Abstract

Overview

The brain is foremost the most important organ in the human body, with a vast number of tissue and cell types. Brain cell types include neurons, oligodendrocytes and their precursory forms, astrocytes with the help of epithelial cells, microglia and glia cells and sometimes a mixture or hybrid of many cell types.

Single cell RNA sequencing has provided a new method of studying the heterogeneity of cell types within a single tissue. However, analysis of highly dimensional data can prove challenging and meaningful analysis is often difficult to determine, and results are difficult to convey to the reader in a meaningful way.

Aims and results

Here, the data and methodology of Darmanis *et al.,* (2015) has been reanalysed by replicating the pipeline outlined, as well as processing the data through a new pipeline. Additionally, the results of the analysis have been integrated into a web interface (http://bioinf6.bioc.le.ac.uk/~srpgrpb/pages/) allowing for flexibility when viewing a range of three-dimensional graphs as well comparison of different clustering and preprocessing analysis approaches. PCA and tSNE clustering approaches were examined in both pipelines and the shared data allowed for some comparisons between the different methods of clustering the data. Additionally, the numbers of predicted clusters varied between different approaches.

Conclusions

The study has provided insight into the challenges associated with replication of bioinformatics analysis, and presents a new method of presenting highly dimensional data on an online interface as an alternative to more traditional presentations of clustering analysis.

Introduction

The brain is composed of various cell types including neurons, oligodendrocytes and microglia. There have been many attempts to classify the brain tissue by cell type and producing a cellular atlas for visualisation of the structure of the brain. This would also help determine how the different cell type are expressed across different aged groups for example.

Single cell RNA sequencing (scRNAseq) is a next generation sequencing technique that examines genomes of individual cells, but is more time consuming than bulk sequencing. It is a highly sensitive technique that is now being used to study tissues and populations, and enhance cell function studies. Bulk cell sequencing technologies are commonly used for visualising transcriptomic and genomic data but is ineffective for the specificity needed in single cell sequencing (Li and Li, 2018). Sequencing tools and technologies are constantly being refined and pioneered as the resources available will continue to improve (Hwang, Lee and Bang, 2018).

However, the bioinformatics analysis pipelines available to process the quantity of data produced by an scRNAseq pipeline have not been perfected and as such newly available pipelines are regularly introduced to the scientific community. Determining the effectiveness of data analysis pipelines on the highly dimensional data that scRNAseq produces can be a particular challenge, alongside variable clustering methodologies making it difficult to elucidate meaningful conclusions from the data in a replicable manner.

Presenting scRNAseq data can be a challenging process and interactive visualisation can make this easier. Interactive data presentation should always use user friendly interfaces allowing for customisation to overcome accessibility limitations such as colour blindness, and should always provide simple and easy to use interfaces to minimise user learning time.

This study aimed to reanalyse the data from the Darmanis et al. (2015) paper by following the original methodology, as well as using a new analysis pipeline to reanalyse the data and investigate the differences different clustering approaches may have. 466 cells were used which were obtained from adult and foetal brain tissue, used to begin developing a 'cellular atlas' for the human brain (Ding *et al*, 2016). The analysis results have been presented on a user friendly website (http://bioinf6.bioc.le.ac.uk/~srpgrpb/pages/) to demonstrate key differences between many different preprocessing and downstream analysis steps.

Methods

Obtaining the data:

Data was available from the Gene expression omnibus (GEO accession GSE67835), which illustrated an overview of the whole experiment. Using the SRA run selector, an overall run table could be downloaded along with an accession list. There were 466 samples remaining after the original quality control process from the paper. Some samples were excluded when they contained less than 400,000 reads, reducing the number of samples from 482 to 466.

FastQ files were downloaded using a perl scripted asperaconnect approach (Gerth, 2018), which was faster than relying on ncbi sratools. First aspera was downloaded and installed from asperasoft.com (Zhu, 2015). The perl script was used with the following command perl sra_download.pl --ascp test_list.txt (with the text file containing SRR names). The gene expression for each cell sample were obtained to allow for replication of the original expression matrix. The metadata from the run-info files provided experimental information on all cells. The same python script from the mrna processing method was used to combine the individual expression files together and replicate the gene expression matrix.

Original Pipeline

Data Preprocessing

The first step was to try and replicate the mRNA sequencing data method. If the original data was completely raw and unprocessed, then the following steps could have been carried out. Figure 1 shows a flowchart of the preprocessing steps outlined by the authors.

The C++ version of Prinseq was first downloaded and installed (Cantu, Sadural and Edwards, 2019). A Bash script was created to run the prinseq options used in the original paper (prinseq_script2.sh). Then FastQC and fadapa were installed to look for and remove overrepresented sequences (Andrews, 2017). Orphan pairs of less than 30bp’s in length had to be removed using prinseq. Any Nextera adaptors within the sequences had to be removed using TrimGalore after it was installed (Krueger, 2012).

The remaining reads were aligned to the hg19 genome downloaded from UCSC (Kent *et al., 2002*) using STAR. Table browser was used to produce an annotated GTF file to create a genome index for STAR alignment (Karolchik *et al*, 2004). However, the gene names in this file were in the wrong format and needed to be altered to SRR format. The genome index was generated using HPC spectre due to the high RAM requirement for the process.

STAR was installed from the GitHub repository and ran to align the reads to the genome using the options that had previously been stated in their paper (Dobin, 2019).
HTSeq was then used to convert these reads to counts for the genes. It was installed and ran using the options stated in the paper (Anders, Pyl and Huber, 2015).
The output of these files were combined using the data_combination.py script and the counts were converted to counts per million in libre-office calc.

Clustering and visualisation

The data was split into 4 files based on the origin of the donated cells (foetal cells, adult neurons, foetal quiescent neurons and foetal replicating neurons) and the full dataset as shown in the data analysis flowchart in Figure 2. First, pairwise analysis was performed on the full dataset using the sdce package (version 1.99.4 as provided on the github page. This version requires flexmix (Leisch, 2004) to be at 2.3-13 before installation), as implemented in the scde_analysis.R script, where direct drop out weighting was used to determine the genetic distances between cells. Clustering analysis was then carried out on these distances using the Mclust package version 5.4.5 (Fraley et al, 2012) .

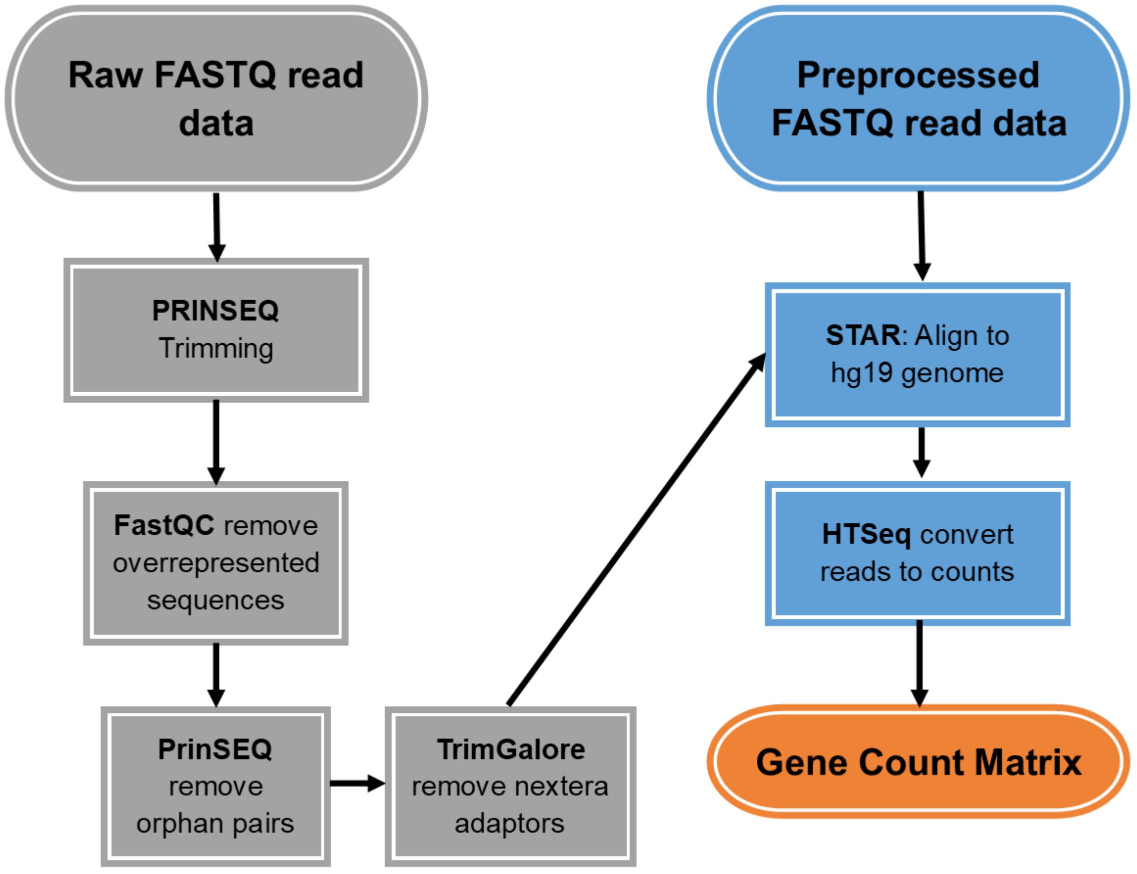
Raw FASTQ read data
PRINSEQ Trimming
FastQC remove overrepresented sequences
PrinSEQ remove orphan pairs
TrimGalore remove nextera adaptors
Preprocessed FASTQ read data
STAR: Align to hg19 genome
HTSeq convert reads to counts
Gene Count Matrix

Figure 1: Pipeline flowchart for the mRNA preprocessing to produce a gene count matrix using the original author's methodology. When processing the FASTQ data provided from the original study it became apparent that many steps may have been carried out on the files prior to them being made available, but the extent of the processing was not made clear. As such, redundant steps (marked in grey) were not able to be carried out, and preprocessed FASTQ files which were available were used to generate the gene count matrix.

The tsne package (version 0.1-3), was used to reduce the data to 3 dimensions. This could then be fed into the Mclust function to determine the presence and number of clusters using the Bayesian Information Criterion (BIC), and any errors that may have occurred in previous clustering approaches (Maaten and Hinton, 2008).

Direct dropout analysis was repeated for the adult neuron and foetal cell datasets, then minimum spanning trees were created for each of these sets of cells using the igraph package (version 1.2.4.1) within R by calculating a distance graph and using it to produce a minimum spanning tree. The vertices of the minimum spanning tree were coloured in accordance to the walktrap cluster membership guidelines (Pons and Csardi, 2005). The longest path was determined as the largest distance within the distance matrix. The individual neuron group datasets were combined into a set containing all neurons (adult and foetal). Again, direct dropout was performed to produce a distance matrix. In this case, the distance matrix was used for principal components analysis (PCA), carried out using the FactoMineR package (version 1.42) within R (Husson *et al*, 2019).

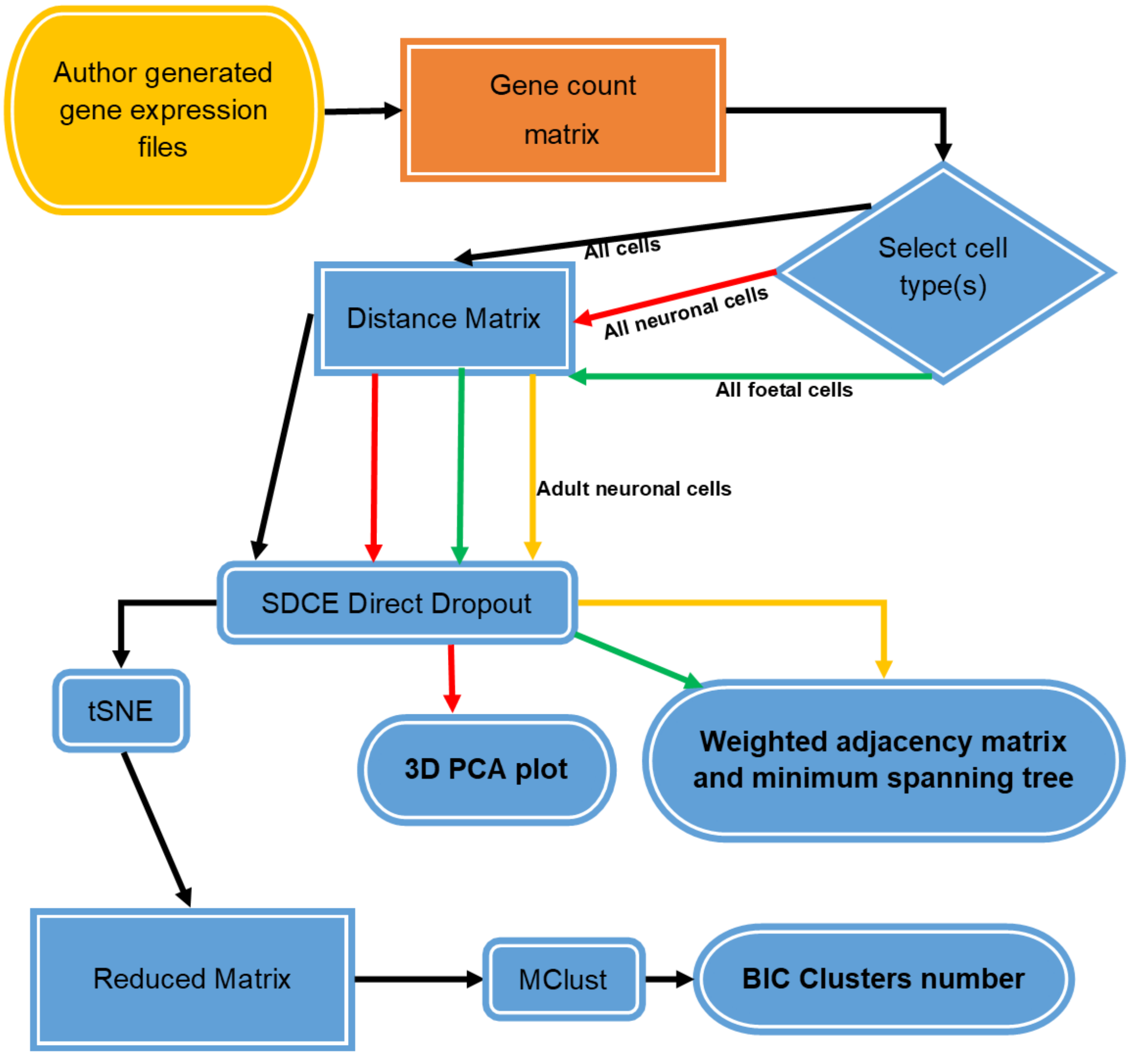

Figure 2: Pipeline flowchart outlining the data analysis portion of the authors original pipeline. The analysis was carried out using the author provided gene count matrix, and also the matrix generated in an attempt to replicate the pipeline. Various subsets of cells individually underwent pairwise analysis using scde, followed by scde directed dropouts. Minimum spanning trees were created using the iGraph package on the foetal cells and adult neuronal cells. PCA plots were created using the FactoMineR R package, and MClust was used to determine cluster numbers using the Bayesian Information Criterion.

Alternative pipeline Data analysis

All alternative pipeline analysis was carried out using R version 3.6.1 and bioconductor version 3.9. The new analysis pipeline developed in this project is outlined in Figure 3.

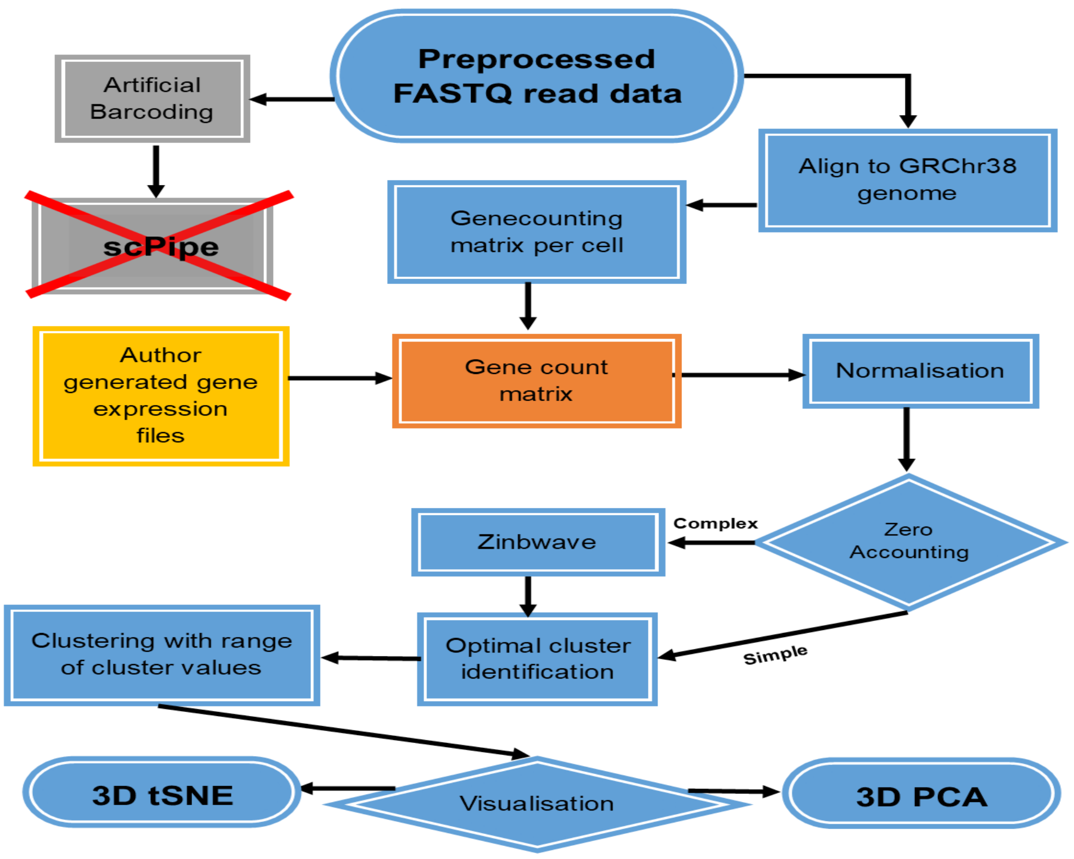
Preprocessed FASTQ read data
Artificial Barcoding
scPipe
Align to GRChr38 genome
Genecounting matrix per cell
Author generated gene expression files
Gene count matrix
Normalisation
Zero Accounting
Complex
Zinbwave
Simple
Optimal cluster identification
Clustering with range of cluster values
Visualisation
3D tSNE
3D PCA

Figure 3: Pipeline flowchart demonstrating the newly implemented pipeline. The pipeline was initially going to use scPipe and further analysis by Scater and SC3 as recommended by the scPipe package, but due to technical difficulties caused by the provided FASTQ files having undergone some processing, scPipe was replaced with Rsubread to carry out the alignment and gene counting steps. Normalisation was carried out using the bioconductor Scater package, and two approaches for dealing with zero inflation were used. The first was simply using the Scater optional argument preserve_zeroes=FALSE during normalisation, whereas Zinbwave was a more computationally demanding purpose built method of accounting for zero inflation. SC3 was used to predict an optimal K value for the number of clusters, and clustering was carried out on a range of values around the predicted value. Finally, the plotly package was used to visualise the clustered cells in three dimensions using both tSNE and PCA approaches.

Rsubread data analysis.

Rsubread (Liao, Smyth and Shi, 2019) is a bioconductor package available for R, and allows for the processing of FASTQ files and alignment to a reference genome in order to generate a gene count matrix. For this analysis a python program has been created (see: Rsubreadmultiproc.py on github) to implement the Rsubread portion of our scRNAseq analysis pipeline using a list of SRA accessions. Files were aligned to a reference index built from the gencode assembly of GRChr38 (primary assembly fasta file and gencode v31 GTF file for annotation). Each sample was then aligned to the index to generate an alignment bam file, and the Rsubread featurecounting function was used to generate a gene count matrix for the individual cell (see Rsubread_multi.R on github).

This gene count matrix could then be processed into a total experiment gene count matrix using data_combination.py for downstream analysis. The gene count matrix produced by the authors were also utilised within the downstream analysis to identify whether the differences in clustering produced by the authors were the result of the matrix production or due to the clustering pipeline itself.

SC3

The Scater pipeline (Kiselev *et al*, 2017) normalised the count matrix on a $\log_2$ scale to account for the proportional changes in values as opposed to the additive changes. We accounted for differences in library size, as the number of genes expressed between cells would not be consistent.

To account for the zero inflation found within scRNAseq data, Zinbwave was utilised as per the suggested pipeline analysis outlined by scPipe. As Zinbwave requires a cluster number, also known as a k value, the SC3 function ‘sc3_estimate_k’ estimated the optimal cluster number. Zinbwave was then implemented using the ‘BiocParallel’ package to account for the computational density of Zinbwave. As the effects of ZInwave on cluster were unknown, cluster analysis with and without Zinbwave were completed for further insight.

Once the optimal k value was identified, clustering of the data was completed using a range of k values around optimal number and the full SC3 clustering pipeline was completed as per the SC3 documentation example (Kiselev *et al.,* 2017). Clusters will be visualised using the ‘plotTSNE’ and ‘plotPCA’ functions from the scater package utilising the ‘ncomponent’ arguments to ensure that a 3D graph can be produced. Plotly (Sievert, 2018) utilised the 3D plot data from the scater outputs and produced a pure 3D graph to represent the data.

Website

Interface development:

The website interface was developed using a number of languages. For text and layout, html was used, with css and javascript used to implement styles and menu interactivity respectively. PHP utilising the mysqli query functions was used to allow users to query a database created from the final results.

R Shiny graphs:

Plotly (Sievert, 2018) was used to create 3D interactive Rshiny graphs for the clustering analysis. Plotly was also utilised to create the 3D graphs generated by the reanalysis of the authors methodology. This allowed for the graphs produced to be more comparable between pipelines due to the consistency in its usage. Rshiny was also used to provide interactivity to other graphs along the pipeline, allowing comparison between the results of the original pipeline when performed on count matrices obtained through different methods.

Cluster specific gene querying:

In order to allow users additional data visualisation of the clustering effects of the new pipeline a small database was generated from the reanalysis pipeline data. This was implemented using a mysql database allowing the user to query individual genes with k value from the final range used for clustering (5-25). Users are able to query a full list of gencode genes from the analysis pipeline, and information on the individual cells with recorded expression of that gene, alongside information about the cell type, tissue and donor for each cell provided by the metadata table provided by the original authors on the SRA website.

Results

Website:

The analysis and reanalysis has been presented with an online website (found at http://bioinf6.bioc.le.ac.uk/~srpgrpb/pages/) which presents interactive graphs for PCA, clustering and minimum spanning trees, as well as an additional gene query interface.

Data was analysed using both pipelines to create a number of interactive graphs. The colours in these graphs can be controlled by colour pickers, which allows customisation of the graphs for cosmetic and accessibility reasons. Various datasets can be chosen from among a provided set, as well as allowing different clustering methods to be shown. The original pipeline determined the optimal clustering number by BIC, and the BIC graph, as well as uncertainty graphs for each of the datasets. The clusters calculated by that method as well as the cell types provided in the SRA run table were coloured on the tSNE plot, to show how well the calculated clustering and the run table matched.

The PCA plots illustrate the differences in the principal components among the three neuron groups in the data. Minimum spanning trees show communities of cells, and likely relations between cells in the sample. For the new pipeline implementation, PCA and tSNE plots are presented using both the author generated gene count matrix and the Rsubread generated gene count matrix, and is designed in such a way that has been designed to make comparisons between the variety of graphs created for future analysis. The level of interactivity allows for graphs to be visualised in 3D space, the clusters to be coloured as per the users selection and also allows for variable k-means to be selected to view comparisons between different values.

Additionally, individual cell information based on expression of a gene can be queried in a separate search tool, which allows a user to query an individual gene and have a table of cells and properties of the cells returned.

Original Pipeline:

mRNA processing

Of the 466 cells that were downloaded and underwent processing, 5 cells were removed from the expression matrix was generated due to zero recorded gene expression within the matrix. The cells that were removed were SRR1974691, SRR1974705, SRR1974706, SRR1974707 and SRR1974950.

Clustering

The clustering was quite variable between all the different count matrices produced, with each method showing a different number of clusters (8-10). The differences in clustering are shown in figure 4, along with the uncertainty of the clusters in the dataset produced by the original pipeline. Some of this variation can be explained by the stochastic nature of the clustering methods. All of the clustering was relatively consistent with the cell types provided in the SRA run table, as shown in figure 5. The dataset which provided only eight clusters (the one originally produced by the authors), appeared to place many of the "hybrid" cells, the ninth group, in the same cluster as the neurons. Data from the new pipeline has generated nine clusters resembling the cell type clustering, and following the author clustering methods has also produced similar results.

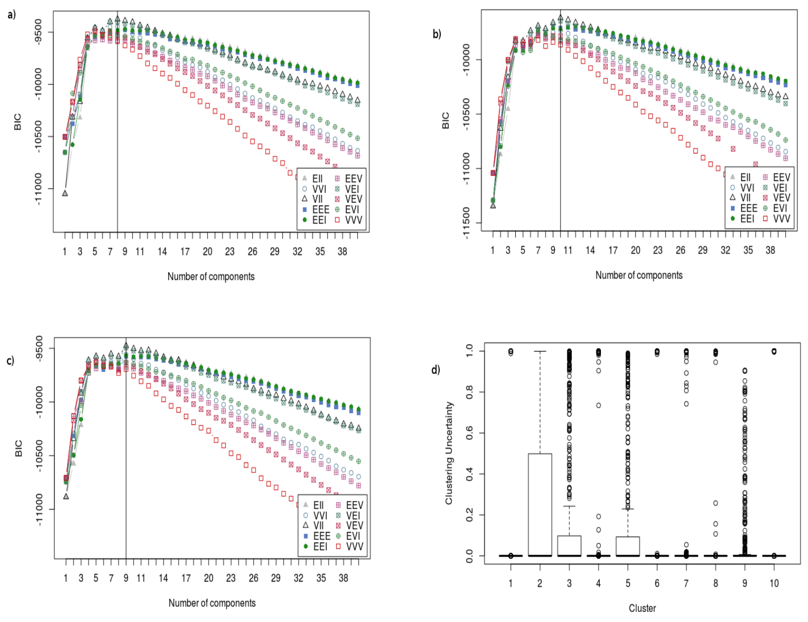

Figure 4. BIC graphs for each count matrix. a) the author generated count matrix showed 8 clusters through BIC analysis, b) the matrix produced through the same process as the original pipeline showed 10 clusters, c) the matrix produced through a new pipeline showed 9 clusters. d) shows the clustering uncertainty for b).

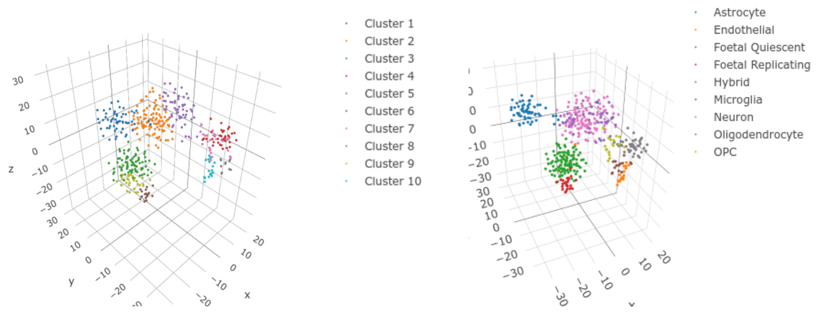

Figure 5. tSNE plots comparing predicted clustering against cell types. The dataset used was produced by the original pipeline.

Spanning Trees

The spanning trees produced a variable number of communities, with the adult neurons showing either 13 or 18 communities, dependent on the sample. 18 communities found within the newer pipeline dataset, and 13 communities were found within the other two. The foetal neuron spanning tree showed between 9 and 13 clusters, with no consistency between the groups produced by the same method. The longest paths on these trees were calculated as the greatest distance between cells in the distance matrix, so while they may not look like the longest path on the spanning trees, this may be due to extra dimensions of community formation, not shown by the trees produced. Both spanning trees for the original pipeline are shown in figure 6.

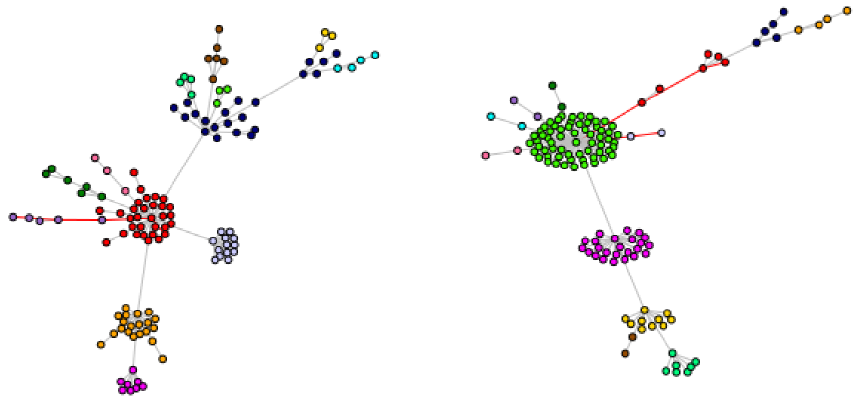

Figure 6. Spanning trees of the adult (left) and foetal (right) neurons, showing 13 communities in each case. The dataset used was produced by the original pipeline.

PCA

The PCA results among the neuron groups in the sample were very similar across all three of the datasets used, and that similarity extends to the PCA graph in the original paper, all of them show the three different cell types as differentiated groups, as would be expected for three sets of cells that are known to have differences in expression. The PCA for the original pipeline is shown in figure 7.

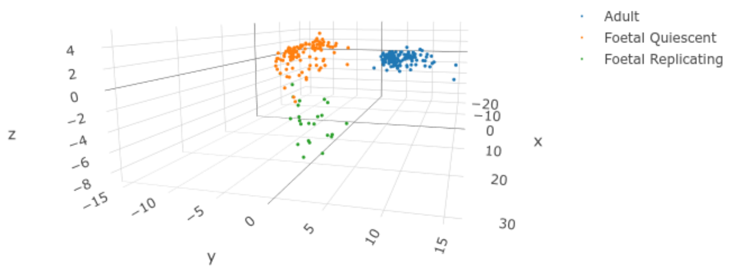

Figure 7. PCA plot showing clustering of the three groups of neurons present in the sample. The dataset used was the one produced by the original pipeline.

Alternative Pipeline:

Clustering

After the formation of our gene count matrix, the "sc3_esitmate_k" function within the SC3 package suggests an optimal k value of 14 for non zinbwave transformed data and the zinbwave altered self generated gene count matrix. The zinbwave transformed author generated gene count matrix was estimated to have an optimal k value of 12. In attempting to validate the optimal k values provided by SC3, k values from 5 to 25 were selected to investigate how different cluster numbers affect the groupings of cells by the pipeline.

The interactive clustering plots showed certain conserved clusters were present in the full range of k values used for clustering when using SC3, one indicator that some of these clusters have biological significance. As seen in figure 8 , changes in K value only affect specific clusters within the tSNE, suggesting that those clusters are not as defined as the clusters found by the authors. Zinbwave did not affect the PCA plots between the same datasets. As PCA plots are non-stochastic compared to tSNE, having identical plots with and without zinbwave would highlight that zinbwave did not affect the clustering.

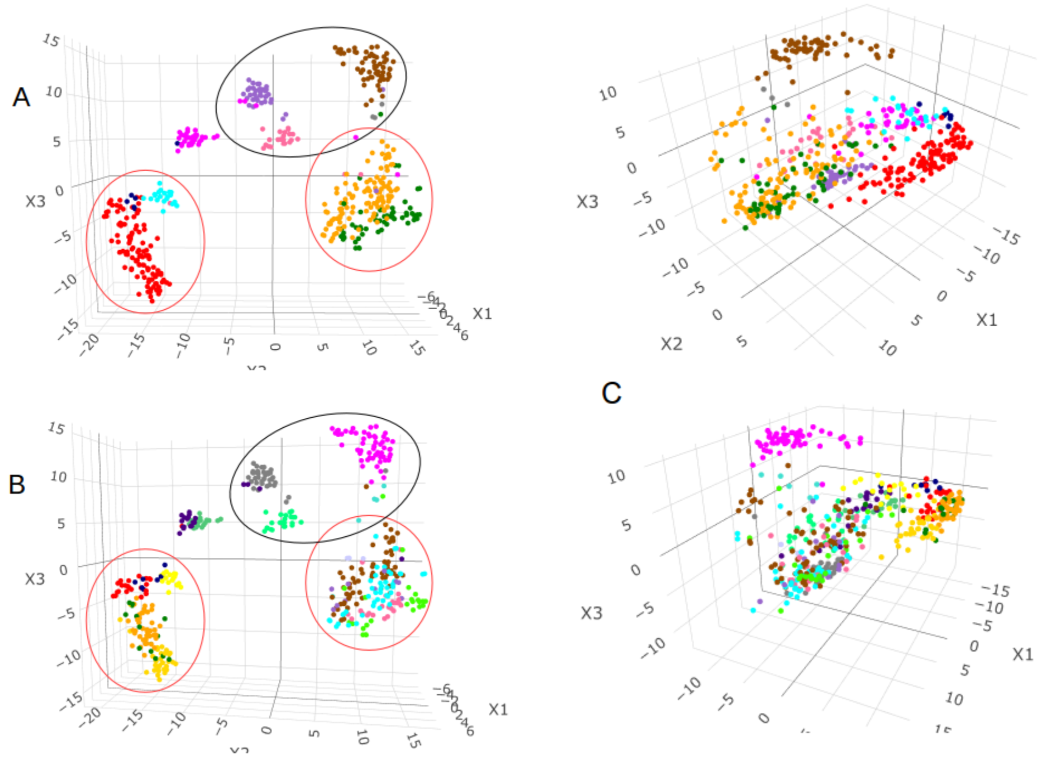

Figure 8. A comparison between a 10 and 18 cluster tSNE plots using Author generated gene count matrix and without zinbwave. The black circles represent clusters that have been conserved between the two k values, whereas the red circles show clusters that change significantly. **A**; The 10 cluster tSNE plot. **B;** The 18 cluster tSNE plot. **C**; The PCA plots for the corresponding k values. The 10 cluster plot is the top image while the 18 cluster plot is on the bottom. While the PCA plot is non-stochastic, the clusters are not clearly represented when compared to tSNE.

When compared to the clusters produced by the authors and our clustering plots, it is apparent that the clusters are significantly different. As seen in the cell type plots on the cluster plot tab, the 9 distinct cell types identified by the authors are not all defined within our self generated clusters. Some cell types were independently clustered and conserved regardless of changes to the k value used to cluster the data, but the hybrid and neuron cell types appear to be clustered together. This is the case with the author or self gene expression matrices and with or without zinbwave. A single endothelial cell is found within the aforementioned cluster regardless of dataset used as well. This implies that either the SC3 did not cluster the cells correctly or that the cell types identified by the authors may not be accurate. When colouring the clusters using a k value of 9 (the same number of clusters identified by the authors), the resulting tSNE and PCA plot looks nearly identical to the cell type plots. This, along with the quality of the clusters, suggest the former.

Quality

As shown by the Quality graphs, the SC3 clustering of both the author and self generated gene count matrices were successful, however there were no k values where all of the clusters were of sufficient quality. While more subjective than the silhouette plot, the consensus matrix highlights that the cell similarities are the cause of how SC3 grouped the cells together. In contrast, the silhouette plots provide an objective, quantitative measure of the consensus matrix' diagonality. As seen in all silhouette plots, the majority of clusters produced high scores, but there are at least one cluster that is significantly low. Overall, SC3 appears to group cells together effectively, but improvements are still possible. Example of these graphs can be seen in figure 9.Based on these quality metrics, the issues observed when colouring the cells by the cell types identified by the authors are likely caused by SC3 itself.

Challenges

mrna processing:

One issue that occurred was that no overrepresented sequences were found in the 'Raw' data using FastQC. However an error occurred within the expression matrix where all the genes were producing zero expression results. This was a good indicator that the 'raw' data was not actually raw.

The Rsubread analysis of the raw data for the alternative pipeline implementation was mostly successful, however, it became apparent that a single cell of data was unavailable from the authors in a usable format due to an unknown file error on both the forward and reverse reads from the paired end data. The SRR in question (SRR1947800) was therefore omitted from the final gene count matrix output, and the alternative pipeline proceeded with only 465 out of the total 466 cells data provided by the original authors (Liao, Smyth and Shi, 2019). This gene count matrix can be found on our GitHub repository in the alternative pipeline folder (count_matrix).

After normalisation with scater, SC3 identified the optimal cluster count on our self generated gene count matrix was 14 for both gene count matrices and regardless of zinbwave usage. After SC3 clustered the cells and identified any possible groups sized between 5 and 25, plottSNE was used within the scater package of R to produce a visual representation of the SC3 clusters.

This led to the production of 2D graphs that sufficiently covered the K values that were determined within the SC3 analysis. The ncomponent argument within plottSNE was changed from 2 to 3 but did not produce any readable 3D graphs, so was replaced with Plotly.
Plotly is a visualisation tool that was used to create 3D interactive graphs in both the original methodology within the paper, and the alternative pipeline for a direct comparison, which may not always be possible when using different visualisation tools due to different options needed by the tools to work correctly. The interactive graphs produced for the author generated gene counting matrix and our self-generated gene counting matrix are present on the website. The graphs being fully interactable, allow for the colours of each cluster to be changed, reorientation of the graphs for viewing at different positions and a graph selector slider to pick which graph to view.

Discussion

Our success/achievements in meeting project requirements

Overall, we have presented a comprehensive website with a range of interactive graphs to allow the user to explore our full set of results for both pipelines. Particular challenges were met when attempting the reanalysis of the data, in part due to the unknown nature of the available FASTQ files, but also due to insufficient documentation to replicate the authors final results, but the pipeline was implemented in a successful manner. Furthermore, the secondary pipeline analysis has demonstrated that there is a significant range of scRNAseq analysis pipelines available.

Challenges in Analysis

Zinbwave

Zinbwave was originally chosen to account for the zero inflation within samples and possibly allow for a more accurate clustering to occur. It utilised the Bioconductor package within R (Risso et al, 2018). The expression matrix and SRA run –info files were converted into a summarised experiment object with rows containing non-essential information or genes with 0 expression in all cells removed. 21625 genes remain from the original 22085. Zinbwave dealt with the zero inflation within the truncated SummarisedExperiment object using k = 14 and an epsilon of 21625. The documentation of Zinbwave specifies that the pipeline is resource intensive with no intent of optimising the code forthcoming. As shown in on the cluster plots under the ‘newer pipeline’ tab, only using Scater to normalise the gene count matrix results in tSNE clusters that are slightly better compared the clusters that utilised zinbwave modified count matrices. This may be the result of tSNE’s stochastic nature, however the clusters produced within our analysis suggests that the zinbwave zero inflation accounting produced less pronounced clusters. This does not discount the importance of zero inflation on scRNAseq data, but only the implementation of zinbwave on multiple datasets would provide further insight into its usefulness.

scPipe

Initially the pipeline chosen for the alternative pipeline analysis of the data provided by the author was scPipe, and the further analysis recommended for clustering and normalisation (SC3 and scater respectively) (Tian et al, 2018). Implemented correctly, scPipe would have been used to strip barcodes and combine paired end reads into a single FASTQ file, before using Rsubread to generate a reference genome index and align the reads to the reference as well as aligning the reads to the annotated exons. scPipe could then have been used to demultiplex the data given the barcode data for the cell, and then generate a gene count matrix using the gene counting function. However, during the implementation of the scPipe pipeline several issues were encountered.

Firstly, the raw data provided by the authors was not accompanied by any information as to the degree of preprocessing it had undergone prior to upload, nor whether sequencing barcodes or UMIs remained in the data. This meant that an artificial barcode had to be added and then subsequently removed from the data, which drastically increased the processing time per cell for the pipeline. Additionally, errors within the gene counting function resulted in gene matrices with no gene expression recorded to be produced, an error replicated with the scPipe package example data and is currently under investigation by a contributor to the scPipe repository. As such, an alternative pipeline was substituted in place of scPipe in order to complete analysis within the timeframe for the project.

When developing the alternative pipeline, various clustering approaches were tested before the actual alternative pipeline was decided on. Some of these methods included Cloud virtual resource (CloVR) sequencing through either VirtualBox or the online resource is normally used to provide a single automated pipeline that is easy for installation and usage (Angiuoli *et al*, 2011). Granatum is a free, user-friendly, interactive online resource that was developed for people to use who have had little to no programming experience to analyse data and hopefully produce clusters. Using the expression matrix provided by the raw data, the genes within the datasets that had 0 expressions were automatically removed causing the genes to go from 22085 to 21360 genes for 466 samples. All the clustering methods that could have been selected produced a different number of random clusters and was only used as a pure comparison basis to see if actual secondary pipeline was clustering in a similar way (Zhu *et al.*, 2017).

Documentation

Particular challenges were met during both reanalysis and implementation of the new pipeline due to difficulty locating sufficient documentation for software. The authors did not provide the original scripts used for analysis nor the particular arguments used which is one factor in understanding the discrepancies between the published results and what was replicated following their pipeline. Additionally, the implementation of the secondary pipeline was repeatedly hindered by insufficient documentation on possible arguments (or values that could be assigned to the arguments) being available from the linked documentation for the software packages.

Future improvements

To the website

To develop a website that could be useful to others, expansion of the interface to allow for users to submit their own data to run through our pipeline could be implemented. Full implementation of a data analysis pipeline could prove beneficial, providing a set environment for scRNAseq gene expression matrix analysis, although some steps in analysis (such as Zinbwave zero accounting) can be time consuming and would not be practical to implement on a reactive web interface.

To analysis

With additional time the knowledge we have gained on scRNAseq analysis could be applied to carry out additional analysis on the results. For example, since 2015 a number of new clustering techniques have been introduced such as the Seurat package (Stuart *et al.,* 2019), for this data Louvain clustering on the PCA space could be completed. Subsetting the samples further for individual group feature selection and PCA for clustering could also be an option, which would eliminate the necessity of multiple spanning trees within the analysis.

Comparison to currently available resources

Due to increasing popularity of scRNAseq analysis, interactive online visualisation tools are becoming commonplace. The scope of many of these tools is significantly larger than what has been presented in our web resource. For example Clustvis (Metsalu and Vilo, 2015) is an online PCA visualisation tool which allows for import of user data. Additionally comprehensive resources such as iDEP were found covering full analysis of both example data and allows for users to input their own datasets (Ge, Son and Yao, 2018). The larger scope of this online tool makes for a more comprehensive analysis, as alongside our website functionality heatmaps, examine expression for entire gene pathways and more. However, despite resources with these functionalities being available, obtaining and maintaining grant funding to provide web based applications for analysis can be challenging due to the relatively small number of researchers in the field.

Comparing clustering approaches

Cluster formation is governed by the similarity of expressed genes between each cell and grouping the most similar. This can result in outliers or incorrectly identified genes affecting which clusters those cells identified within. Bootstrapping provides a method of identifying the effects of specific genes on the clustering algorithm. Ideally, a number of genes will be used to categorise each cell into their clusters and the removal of one gene should not have a significant effect on those cluster assignments. The reference genome annotation file utilised in the self generated gene count matrix contains 59050 genes, therefore bootstrapping each gene would be time consuming and resource intensive. Given additional time, this would provide additional insight regarding the cluster formations that is not currently known.

Gene matrix counts

The clusters generated by both the author and self generated gene count matrices are effectively identical. Comparing the PCA plots for both datasets show minimal differences and the tSNE plots appear to differ only due to the plot's stochastic nature. This is particularly of note as the author's gene count matrix contains 22,085 genes, whereas our gene count matrix accounts for isoforms of each gene, resulting in 59,050 genes being present. The lack of clustering differences isn't surprising as the reads would align to a specific isoform of a gene as opposed to a general gene construct. Differences in clustering could have arisen, but were not seen in our investigations. Overall, no issues can be directed at the author's matrix as a gene count matrix utilising a different reference genome version, reference annotation file and pipeline produced significantly similar clusters using SC3.

K mean determination

Completing the SC3 analysis using a wide range of k values provided insight into highly conserved clusters within the data. If clusters are conserved within a range of K values, then it can be assumed that the formation of those clusters are accurate. The stability plots shown, the example shown in figure 9C, within the quality plots highlight whether a specific cluster at a K value is present when a K value is changed. The stability plots show that there are small numbers of clusters at each K value that have low stability scores. Ideally, as the value of K increases, the clusters should gradually change to compensate. As figure 8 shows, specific clusters are conserved within the author gene count matrix tsne plot, however certain clusters fragment more rapidly as the value of K increases. This would suggest a disparity in the clustering quality, however the cell number appears sparse. The small cell number will increase the effect of an increased K value. As a result, utilising a larger sample set is advisable when identifying clusters. With more samples, larger K values can be included to see the incremental effect of increasing K on the cluster formations.

The files available on the public repository provided by the authors ranged in size from 100MB to 1.6GB per cell. The true raw data was unlikely to be this small, and as no overrepresented sequences were found after the first prinseq step, then it's possible that the data is not available on the public repository in a true raw fastq format. Most likely, some of the mRNA preprocessing steps were carried out on the data prior to uploading. This is a possible reason why not all of the steps in the papers methodology needed to be carried out. The only steps carried out in the reanalysis were STAR, HTSeqand the count conversion to counts per million. The overall expression matrix seems to be similar to the expression matrix that was produced within the paper.

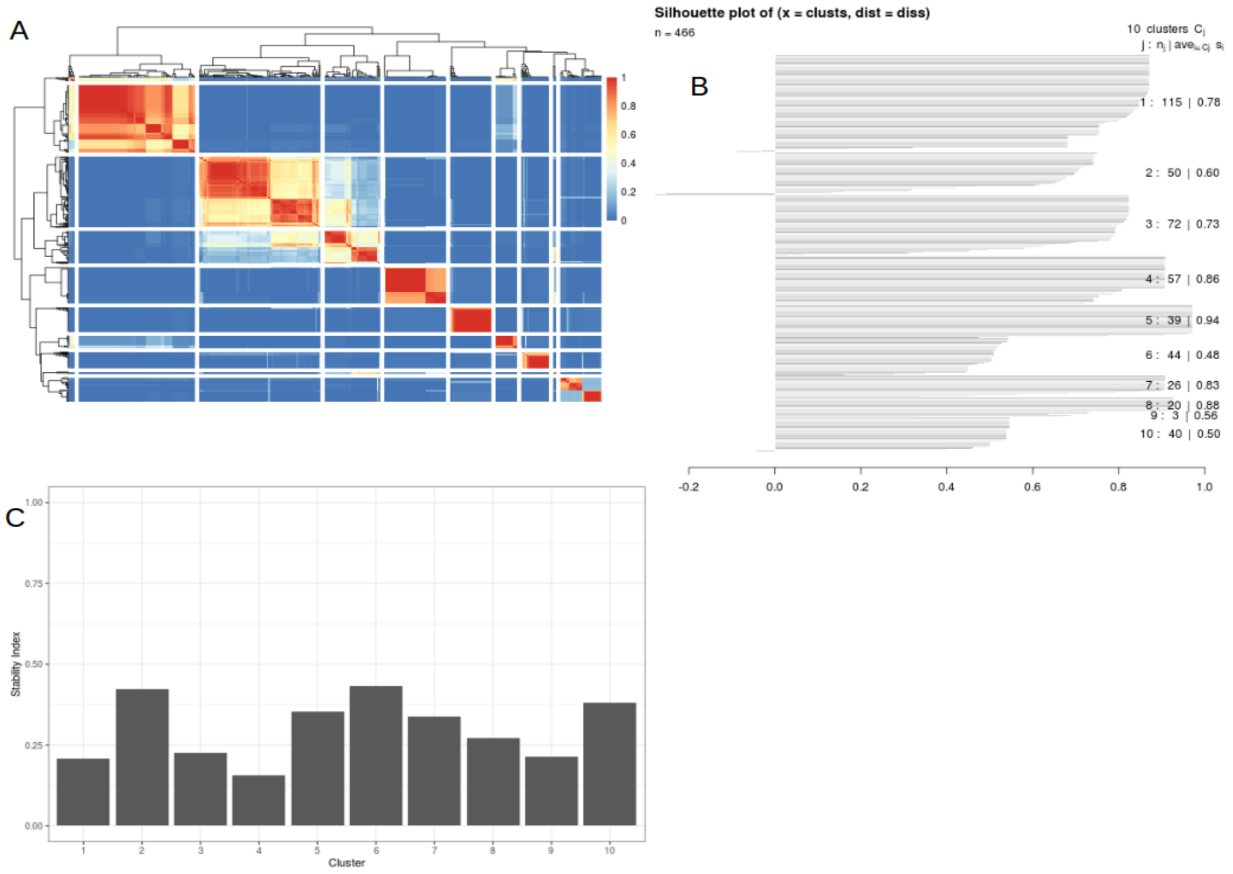

Figure 9. The quality plots generated by SC3 for the author generated gene count matrix, without zinbwave and using a k value of 10. These plots are available under the 'newer pipeline' tab on our website. **A**; The consensus matrix has each cell on the X and Y axis, with the red side of the colour bar highlighting that the cells are similar whereas the blue colour denotes the cells are different. **B**; The silhouette plot illustrates the diagonality of the consensus matrix, with a score of 1 describes a perfect block diagonal consensus matrix and 0 highlights no block-diagonal structure. **C**; The cluster stability plot which highlights whether the clusters generated at a specific k value are conserved within different k values. A score of 1 illustrates the cluster is conserved exactly in other K values and 0 denotes the cluster is not present in any other K value.

A comparison between the expression matrix produced within the mrna processing and the raw data expression matrix already available showed that the results were similar for the most expressed gene across samples, but the amount of expression was different. There was also a large difference in the number of genes being expressed. With the 'Raw' expression matrix there were 22085 that were having their expression observed. But the mrna expression matrix contained 28518 genes. This was probably down to the methodology not being exact enough to 100% know if all the steps within their methodology could be carried out on data that was 'Raw'. Another reason for this difference could be that the genome reference we used for alignment was not the same version as the one they used in the paper. All genome versions contain different amounts of data and could affect the results of the alignment.

Differentially expressed genes, marker genes and expression matrices were also produced and viewable on the website, but will not be focused upon. It is beneficial to investigate the specific genes utilised to create each cluster and locate the cell type closest to each cluster's gene expression profile. Insufficient time prevented further analysis and is something to investigate in the future.

Conclusion

Attempting a reanalysis of Darmanis *et al.*, (2015)'s research has revealed many of the challenges in day to day scRNAseq data analysis, as well as issues in methodology documentation in bioinformatics research preventing true reproducibility. Both the reanalysis of the original pipeline and the alternative pipeline failed to replicate the results produced within the actual paper.

The website that we have produced demonstrates the high dimensionality of the data, and provides opportunities to directly compare the results between different clustering methodologies and k means. This clearly displays the wide variances between clustering methodologies, and exposes the difficulties in determining the "true" results when comparing to noise generated by the experimental methods.

References

Anders, S., Pyl, P.T. and Huber, W. (2015) 'HTSeq—a Python framework to work with high-throughput sequencing data', Bioinformatics, 31(2), pp. 166-169.

Andrews, S. (2017) , FastQC: a quality control tool for high throughput sequence data.2010,.

Angiuoli, S.V., Matalka, M., Gussman, A., Galens, K., Vangala, M., Riley, D.R., Arze, C., White, J.R., White, O. and Fricke, W.F. (2011) 'CloVR: a virtual machine for automated and portable sequence analysis from the desktop using cloud computing', *BMC bioinformatics,* 12(1), pp. 356.

Cantu, V.A., Sadural, J. and Edwards, R. (2019) 'PRINSEQ , a multi-threaded tool for fast and efficient quality control and preprocessing of sequencing datasets', PeerJ Preprints, 7, pp. E27553v1.

Dobin, A. (2019) *STAR.* Available at: https://github.com/alexdobin/STAR/blob/master/README.md (Accessed: July 4th 2019).

Ding, S., Royall, J.J., Sunkin, S.M., Ng, L., Facer, B.A., Lesnar, P., Guillozet-Bongaarts, A., McMurray, B., Szafer, A. and Dolbeare, T.A. (2016) 'Comprehensive cellular-resolution atlas of the adult human brain', *Journal of Comparative Neurology,* 524(16), pp. 3127-3481.

Fraley, C., Raftery, A.E., Murphy, T.B. and Scrucca, L. (2012), mclust version 4 for R: normal mixture modeling for model-based clustering, classification, and density estimation.

Ge, S.X., Son, E.W. and Yao, R. (2018) 'iDEP: an integrated web application for differential expression and pathway analysis of RNA-Seq data', *BMC bioinformatics,* 19(1), pp. 534.

Gerth, M. (2018) *perlscripts/sra_download.pl*
. Available at: https://github.com/gerthmicha/perlscripts/blob/master/sra_download.pl (Accessed: 04/07/2019).
Husson, F., Josse, J., Le, S., Mazet, J. and Husson, M.F. (2019) 'Package ‘FactoMineR’', *Package FactorMineR,* .

Hwang, B., Lee, J.H. and Bang, D. (2018) 'Single-cell RNA sequencing technologies and bioinformatics pipelines', Experimental & molecular medicine, 50(8), pp. 96.

Karolchik, D., Hinrichs, A.S., Furey, T.S., Roskin, K.M., Sugnet, C.W., Haussler, D. and Kent, W.J. (2004) 'The UCSC Table Browser data retrieval tool', *Nucleic acids research,* 32(suppl_1), pp. D496.

Kent WJ, Sugnet CW, Furey TS, Roskin KM, Pringle TH, Zahler AM, Haussler D. The human genome browser at UCSC. *Genome Res.* 2002 Jun;12(6):996-1006.

Kiselev, V.Y., Kirschner, K., Schaub, M.T., Andrews, T., Yiu, A., Chandra, T., Natarajan, K.N., Reik, W., Barahona, M. and Green, A.R. (2017) 'SC3: consensus clustering of single-cell RNA-seq data', Nature methods, 14(5), pp. 483.

Krueger, F. (2012) 'Trim Galore: a wrapper tool around Cutadapt and FastQC to consistently apply quality and adapter trimming to FastQ files, with some extra functionality for MspI-digested RRBS-type (Reduced Representation Bisufite-Seq) libraries', URL http://www.bioinformatics.babraham.ac.uk/projects/trim_galore/.(Date of access: 28/04/2016), .

Leisch, F. (2004) 'Flexmix: A general framework for finite mixture models and latent glass regression in R', .

Li, W.V. and Li, J.J. (2018) 'An accurate and robust imputation method scImpute for single-cell RNA-seq data', Nature Communications, 9(1), pp. 997.

Liao, Y., Smyth, G.K. and Shi, W. (2019) 'The R package Rsubread is easier, faster, cheaper and better for alignment and quantification of RNA sequencing reads', *Nucleic acids research,* 47(8), pp. e47.

Maaten, L.v.d. and Hinton, G. (2008) 'Visualizing data using t-SNE', Journal of machine learning research, 9(Nov), pp. 2579-2605.

Metsalu, T. and Vilo, J. (2015) 'ClustVis: a web tool for visualizing clustering of multivariate data using Principal Component Analysis and heatmap', *Nucleic acids research,* 43(W1), pp. W570.

Pons, P.&. & Csardi, G. (2005) *igraph.* Available at: https://igraph.org/r/doc/cluster_walktrap.html.

Risso, D., Perraudeau, F., Gribkova, S., Dudoit, S. and Vert, J. (2018) 'A general and flexible method for signal extraction from single-cell RNA-seq data', Nature Communications, 9(1), pp. 284.

Sievert, C. (2018) 'plotly for R', *Available under: https://plotly-book.cpsievert.me,* .

Stuart, T., Butler, A., Hoffman, P., Hafemeister, C., Papalexi, E., Mauck III, W.M., Hao, Y., Stoeckius, M., Smibert, P. and Satija, R. (2019) 'Comprehensive Integration of Single-Cell Data', *Cell,* .

Tian, L., Su, S., Dong, X., Amann-Zalcenstein, D., Biben, C., Seidi, A., Hilton, D.J., Naik, S.H. and Ritchie, M.E. (2018) 'scPipe: a flexible R/Bioconductor preprocessing pipeline for single-cell RNA-sequencing data', PLoS computational biology, 14(8), pp. E1006361.

Zhu, J.e.a. (2015) *Aspera.* Available at: https://downloads.asperasoft.com/en/downloads/8?list (Accessed: 3/7/19).

Zhu, X., Wolfgruber, T.K., Tasato, A., Arisdakessian, C., Garmire, D.G. and Garmire, L.X. (2017) 'Granatum: a graphical single-cell RNA-Seq analysis pipeline for genomics scientists', Genome medicine, 9(1), pp. 108.

Code Appendix:

All code used in this investigation can be found on our github repository at
https://github.com/MichaelKubiak/Group_project