PERFORMANCE OF PERMUTATION TESTS USING SIMULATED GENETIC DATA

A Dissertation
Submitted to the Graduate Faculty
of the
North Dakota State University
of Agriculture and Applied Science

By

Ibrahim Soumare

In Partial Fulfillment of the Requirements
for the Degree of
DOCTOR OF PHILOSOPHY

Major Department:
Statistics

April 2022

Fargo, North Dakota

North Dakota State University
Graduate School

Title

PERFORMANCE OF PERMUTATION TESTS USING SIMULATED GENETIC DATA

By

Ibrahim Soumare

The Supervisory Committee certifies that this ***disquisition*** complies with North Dakota State University's regulations and meets the accepted standards for the degree of

DOCTOR OF PHILOSOPHY

SUPERVISORY COMMITTEE:

Dr. Rhonda Magel
Chair

Curt Doetkott

Dr. Megan Orr

Dr. Changhui Yan

Approved:

04/20/2022
Date

Dr. Rhonda Magel
Department Chair

ABSTRACT

Disease statuses and biological conditions are known to be greatly impacted by differences in gene expression levels. A common challenge in RNA-seq data analysis is to identify genes whose mean expression levels change across different groups of samples, or, more generally, are associated with one or more variables of interest. Such analysis is called differential expression analysis. Many tools have been developed for analyzing differential gene expression (DGE) for RNA-seq data.

RNA-seq data are represented as counts. Typically, a generalized linear model with a log link and a negative binomial response is fit to the count data for each gene, and DE genes are identified by testing, for each gene, whether a model parameter or linear combination of model parameters is zero.

We conducted a simulation study to compare the performance of our proposed modified permutation test to DESeq2 edgeR, Limma, LFC and Voom when applied to RNA-seq data. We considered different combinations of sample sizes and underlying distributions. In this simulation study, we first simulated data using Monte Carlo simulation in SAS and assessed True Detection rate and False Positive rate for each model involved. We then simulated data from real RNA-seq data using SimSeq algorithm and compared the performance of our proposed model to DESeq2 edgeR, Limma, LFC and Voom.

The simulation results suggest that Permutation tests are a competitive alternative to traditional parametric methods for analyzing RNA-seq data when we have sufficient sample sizes. Specifically, the results show that Permutation controlled Type I error fairly well and had a comparable Power rate. Moreover, for a sample size $n \geq 10$ simulation exhibited a comparable True detection rate and consistently kept the False Positive rate very low when sampling from

Poisson and Negative Binomial distributions. Likewise, the results from SimSeq confirm that Permutation tests do a better job at keeping the False Positive rate the lowest.

ACKNOWLEDGMENTS

I first want to thank my advisors, Dr. Rhonda Magel and Curt Doetkott, and committee chair, Dr. Rhonda Magel. I had the unique opportunity to work with two advisors. I could not have asked for a more understanding and supportive advisors to help guide me through this journey and get me where I am today. Dr. Rhonda, in spite of her busy schedule, she always made time for our bi-weekly meeting and truly made this work as smooth as possible. Curt, thank you for lending me your experience in coding and consulting. I am very much indebted to both of you. Thank you for your valuable contributions and advices. I would also like to thank my dissertation committee members: Dr. Megan Orr and Dr. Changhui Yan. Their feedback and advices were very significant in this work.

I would also like to thank my parents for their unconditional love and support. Thank you to my amazing spouse, Djenaba, and son Mohamed for always cheering me up and believing in me. A special thank you to Elhadj Babourema DRAME. Thank you for believing in me and for being one of my great supports. I am eternally indebted to you.

DEDICATION

To my parents, brothers and sisters, my wife and son for their unconditional love and support.

To my grandfather, Elhadj Babourema DRAME, for believing in me and his continued unconditional support and love.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

MC .....Monte Carlo.

C .....Control.

T .....Treatment.

FDR .....False Discovery Rate.

TP .....True Positive.

FP .....False Positive.

FN .....False Negative.

FPR .....False Positive Rate.

TPR .....True Positive Rate.

DE .....Differentially Expressed.

EE .....Equally Expressed.

DGE .....Differential Gene Expression.

SDE .....Significantly Differentially Expressed.

NB .....Negative Binomial.

ZIP .....Zero Inflated Poisson.

Perm .....Permutation.

ZINB .....Zero Inflated Negative Binomial.

ZANB .....Zero-altered Negative Binomial.

cDNA .....Complementary Deoxyribonucleic Acid.

mRNA .....Messenger Ribonucleic Acids.

DS .....Differential Splicing.

LIST OF SYMBOLS

μ .....Population Mean.

σ .....Population Variance.

λ .....Poisson Parameter Lambda.

π .....Proportion of Zero.

α .....Significance Level.

LIST OF APPENDIX FIGURES

CHAPTER 1. INTRODUCTION

Disease statuses and biological conditions are known to be greatly impacted by differences in gene expression levels (Li & Tibshirani, 2013). The recent rise of RNA-seq technology has now supplanted microarrays as the technology of choice for genome-wide Differential Gene Expression (DGE) experiments.

As described by Li and Tibshirani (2013), in each experiment, messenger ribonucleic acids (mRNAs) are shattered and reverse transcribed into complementary deoxyribonucleic acid (cDNA). These short pieces of cDNA are amplified by a polymerase chain reaction and sequenced by a sequencing machine, giving a list of short sequences called reads. These reads are then mapped to the reference genome using an appropriate algorithm, telling us which region each read comes from. Finally, for a set of regions of interest on the genome, such as genes, exons, or junctions, we count the number of reads mapped unambiguously to each of them and use this count as a measure of the expression of the region.

A common challenge in RNA-seq data analysis is to identify genes whose mean expression levels change across different groups of samples, or, more generally, are associated with one or more variables of interest. Such analysis is called differential expression analysis. Differential expression analysis usually involves carrying out a significance test for each gene. Because RNA-seq data generally contain thousands of genes, differential expression analysis involves testing thousands of hypotheses.

Many tools have been developed for Analyzing DGE for RNA-seq data. RNA-seq data are represented as counts and statistical methods that try to identify differential expression – enhanced (“up-regulated”) or suppressed (“down-regulated”) make assumptions about the

statistical properties inherent to the data and they exploit a range of normalization and analysis techniques to compute the magnitude of a DGE result and estimate its significance.

Typically, a generalized linear model with a log link and a negative binomial response is fit to the count data for each gene, and DE genes are identified by testing, for each gene, whether a model parameter or linear combination of model parameters is zero.

It is reported that data from technical replicates can often be well characterized by Poisson distribution, while data from biological replicates have much larger variance and negative binomial models seem to be more appropriate.

Estimates obtained from inferential statistical methods are generally reliable when the underlying assumptions are met. However, when the underlying assumptions of the test statistic are not met, the sampling distribution of the test statistic may deviate substantially leading to inaccurate inferences.

According to Zimmerman and Zumbo (1990), the tendency of researchers to prefer the use of parametric statistics have led many to propose some transformation techniques to satisfy the underlying parametric assumptions. However, others such as Sawilowsky, Blair and Higgins (1985) have shown that transforming data for certain designs can be dramatically non-robust and often produce poor power properties. This controversy calls for the need to better understand statistical procedures available to researchers given an unknown or non-normal population distribution.

Simple permutation tests use rearrangements of the original sample to build the sampling distribution of the test statistic so make minimal assumptions about the data. For clients with modest mathematical or statistical background, permutation tests are often more intuitive than even basic parametric tests such as the two-sample t-test. Inferential methods associated with

RNA-seq data are substantially more mathematically challenging than the two-sample t-test so may be even more difficult to comprehend.

According to Good (1994), permutation tests can be applied to continuous, ordered and categorical data, and to values that are normal, almost normal, and non-normally distributed. For almost every parametric and nonparametric test, one may obtain a distribution-free permutation counterpart. The resulting permutation test is usually as powerful as or more powerful than alternative approaches. And permutation methods can sometimes be made to work when other statistical methods fail.

Permutation tests can take multiple forms. Exact permutation tests compile all possible combinations of treatment and control data for the chosen test statistic. They are called exact because the relevant properties are specifically determined, that is an exact level of significance is determined by a significance test (Walsh, 1968). The moment approximation test uses the continuous probability density function based on the exact lower moments of the test statistic fitted to the discrete permutation distribution. Finally, the approximate randomization test focuses on a random subset of all possible permutations (Mielke & Berry, 2001). In situations where the number of permutations may be overwhelming due to a large sample size, an approximate randomization test can be a viable alternative. Several researchers suggest that permutation and randomization tests help to rehabilitate the power of parametric tests under conditions of non-normality (Potvin & Roff, 1993; Edgington, 1995). And still, others offer permutation tests as preferred alternatives to rank-based tests, citing that rank tests are less powerful than randomization tests on scores (May, Masson, & Hunter, 1989).

The goal of this study is twofold. First, to compare the performance of the permutation test to comparable parametric tests in the two-sample differential expression setting using

simulated data that mimic RNA-seq data. And secondly, to investigate how the sample sizes impact the permutation results. Various scenarios will be explored, some in which the underlying assumptions on the data are met such that the parametric tests perform well and in others where the underlying assumptions on the data for the parametric tests are violated to varying degrees. Our general expectations are that the parametric tests will usually be more powerful, but if simple permutation tests yield reasonably close results, they may be preferred by clients due to their more intuitive nature. A broad study outline follows.

The core of this study will be carried out in two phases. The first phase is further divided into four sub-phases. Firstly, we will simulate RNA-seq data assuming various underlying distributions using Monte Carlo simulation in SAS to mimic real RNA-seq datasets for relatively small sample sizes ($n_1=n_2\leq10$). For our simulated data, we will consider three distributions, namely: Normal, Poisson and Negative Binomial (NB) distributions and then assess Type I error and Power rate for each combination of underlying distribution and sample sizes for the fitted models considered. Secondly, we will repeat the simulation process as describe above but this time we will set twenty percent (20%) of the simulated RNA-seq data to be differential expressed (the DE genes are obtained at 0.5σ and 1σ effect sizes). We will then assess differential expression using permutation, two-sample t-tests, Poisson regression and Negative Binomial regression at varying number of replicates. For each combination of underlying distribution (Normal, Poisson or NB) and number of replicates ($n = 5$, $n = 7$ and $n = 10$), we will obtain the number of differentially expressed (DE) genes for each of the fitted models (T-test, Permutation, Poisson, Negative Binomial). Secondly, from the DE genes obtained we will assess the True Positive (TP) rate and the False Positive (FP) rate and compare these rates across the fitted models.

The process just described was conducted for distributions with mean $\mu = 30$ and standard deviation $\sigma = 5$ for samples simulated from Normal distribution and Lambda=30 for samples simulated from Poisson distribution. For negative binomial, we increased the standard deviation from 5 to 8 and kept the mean the same as that of normal and Poisson distribution at $\mu = 30$. And thirdly, we will increase the number of replicates (greater or equal to 12) and repeat the same process as described above. However, it is important to note for the permutation distribution that when a large number of replicates is considered we did not perform a full permutation test. The number of possible permutations is overwhelming for large samples, therefore we take a random sample of all possible permutated data instead (B=5000). For each of the sub samples of the permuted data we will assess DE genes for each fitted model at varying number of replications and compare the True Positive and False Positive rates across all the fitted models.

After checking the capability of the models to reasonably control false positive rate and detect true DE genes on the simulated data from the theoretical models using Monte Carlo simulation, we will then simulate RNA-seq data from a real dataset in phase 2. The process in phase 2 is much the same as that conducted in phase1 with a slight difference in the simulation procedure. Unlike phase 1, in phase 2 we will simulate RNA-seq dataset from an existing large RNA-seq dataset using SimSeq approach. SimSeq is a data-based simulation algorithm proposed by Sam Benidt and Dan Nettleton (Benidt & Nettleton, 2015). The algorithm is thoroughly described in chapter 4. We will again fit each model (T-test, Permutation, edgeR, limma, LFC and DESeq2) to the simulated data from SimSeq and assess True Positive and False Positive rates across the fitted models and several samples sizes. For large number of replicates (≥ 12),

similarly to phase 1, we will take sub samples (B=5000) of the permuted data and compute the True Positive and False Positive rates.

We expect the parametric tests to perform better in terms of True detection rate (with the false Positive rate fairly low by all approaches). However, our early results suggested that permutation consistently keeps the False Positive rate low compared to EdgeR, Deseq2, Limma, LFC and Voom. Although we expect the detection rate for the parametric tests to be better, we are interested in showing how much poorer the results are using the permutation test. For some clients, the permutation test may be preferable due to its intuitive nature *if* the loss of power is not too great.

Additionally, the two phases as described above were carried out under balanced Two-Sample scenario (Treatment and Control). Moreover, we also applied the scenario described in phase 1 phases to unbalanced sample sizes and see how the results compared to that of balanced sample sizes.

We then attempt to answer the following research questions:

- How do True Positive rate and False Positive (FP) rates compare across T-test, Permutation, Poisson, Negative Binomial, EdgeR, Limma-Voom and DESeq when applied to RNA-Seq data?
- How does the sample size impact the permutation test results?

In the following section, we will first describe more in depth the background of parametric methods used for RNA-Seq data (PoissonSeq, EdgeR, DESeq, Poisson and NB distribution), the use of Monte Carlo Simulation and SAS programming to estimate and test the simulated results, as well as past research. Second, we will discuss the methodology and simulation study approach. We will then present the results from the simulations, along with

some discussion on these results. Following the results, we will elaborate on the importance of the results. The last section of this paper will consist of the summation and the overall thoughts and findings of this simulation study. Figure 1 below summarizes the different stages of this study.

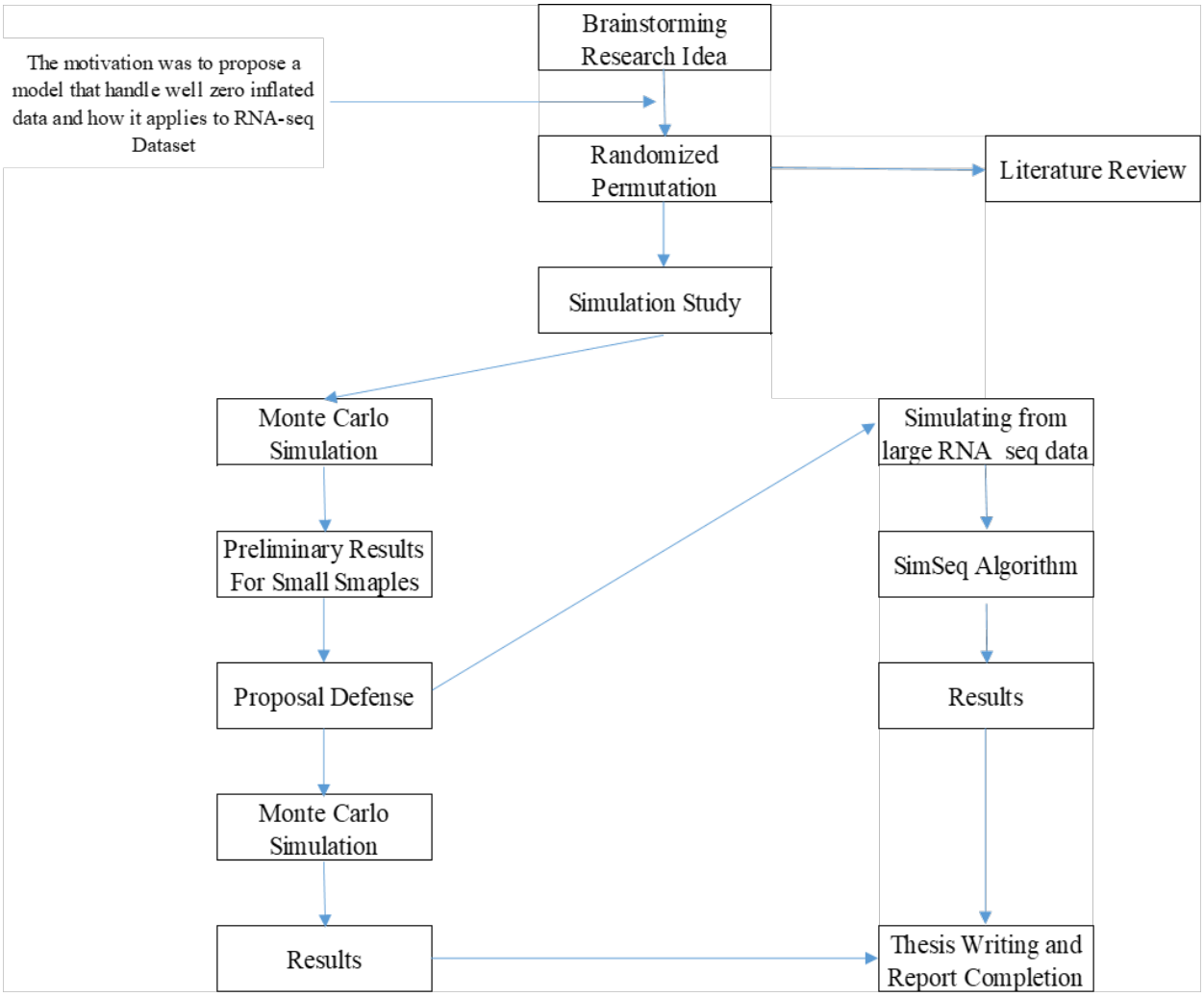

Figure 1. Thesis workflow

CHAPTER 2. LITERATURE REVIEW

In this chapter, we will review some of the historical literature associated with our paper. First, we will review the use of parametric methods assuming particular distribution to identify differentially expressed features from RNA-Seq data such as Poisson, Negative Binomial and Gaussian distribution. Then we will review some non-parametric models as a valid competitor/alternative to the parametric methods. More specifically, we will review the use of simple permutation to detect differentially expressed genes from RNA-Seq data. Finally, we will sum it up with review regarding the best model for RNA-Seq data.

Parametric Methods for RNA-Seq Data

Parametric tests rely on a set of underlying assumptions for the results to be valid. T-test for instance assumes the observations are independent from each other and that the samples are drawn from a normally distributed population. Additionally, it assumes equal variances across the treatment groups (Hunter & May 1993).

Parametric tests can be robust when violation of the assumptions are not severe (Zimmerman, 1987; Sawilowsky & Blair, 1992) and when the sample size is relatively large ($n \geq 30$). In reality, samples used in experiments do not always satisfy underlying assumptions of the models used and researchers should carefully examine and choose appropriate models. Since RNA-seq generates count data, discrete probability distributions are generally appropriate to use to analyze differential gene expression (Robinson & Smyth, 2007). Traditional parametric models used to analyze differential gene expression in RNA-seq data including, but not limited to EdgeR, DESEq2, Limma, Voom, assume Poisson or Negative Binomial distributions (Robinson, McCarthy & Smyth, 2009; Anders & Huber, 2010).

A study conducted by Schurch et al. compared nine DGE tools on a clean dataset of 889 million genes with 42 biological replicates for both control and treatment groups namely: baySeq, cuffdiff, DESeq, edgeR, limma, PoissonSeq, SAMSeq and DEGSeq. Three of the nine tools, EdgeR, DESeq and limma showed excellence performance. They successfully controlled their false positive rate (FPR), maintaining it consistently close or below 5% regardless of fold-change threshold or number of replicates. These are the most widely-used tools, with the exception of limma, suggesting that the majority of the RNAseq DE analyses in the literature are using the most appropriate tools for the job (Schurch et al, 2015). Furthermore, the study suggested that for a low number of replicates ($n \leq 12$) with high capture of significantly differentially expressed (SDE) genes, EdgeR is preferred to limma or DESeq due to its superior TP rate identification. And for sufficient numbers of replicates ($n \geq 12$), in order to ensure that the majority of the true SDE genes is captured it is important to minimize FPR. The slightly better performance of DESeq makes it the best tool of choice. Inversely, PoissonSeq, SAMSeq, DEGSeq, baySeq and cuffdiff all showed inferior performance compared to edgeR, DESeq and limma.

Given the complexity of RNA-seq datasets, the frequency distribution of the read counts does not portray a clear distinction in two classes of genes. Therefore, it is difficult to decide whether low read counts are to be considered expressed genes or not. This problem is addressed by fitting a statistical model that assumes the data is from a mixture of two distributions (Gunter, Koryu & Vincent, 2013). A mixture model of exponential distribution for low read counts (transcripts from inactive genes) and Negative binomial distribution for actively transcribed genes was applied to a number of RNA-seq data sets and the researcher found that the model fitted the data very well (Gunter, Koryu & Vincent, 2013). Gunter, Koryu and Vincent compared

the calculated criterion used for distinguishing between expressed and non-expressed genes and found a consistent results among data sets, which suggests that genes with high TPM values (more than two transcripts per million transcripts) are highly likely from expressed genes. Hence, regression models can sufficiently detect the not actively expressed class of genes and thus, provides a practical criterion to separate genes in expressed and non-expressed groups, smoothing the interpretation of RNA-seq data.

It is well known that the negative binomial distribution often has its largest mass not far from the mean, so, it is very unlikely that the counts follow a negative binomial distribution. If we still treat the distribution of counts as negative binomial, these large counts should be 'outliers'. There are possible reasons for outliers. A gene may be very highly expressed in one individual but not others. In this case, this high expression is a characteristic of this individual, and not related to the outcome.

For univariate count data, zero-inflated negative binomial (ZINB) models have been well accepted and have greater capability than Poisson, zero-inflated Poisson, and negative binomial models in terms of handling augmented zeros and overdispersion. While negative binomial models have been extensively used for bulk RNA-seq data without much zero-inflation (Love et al., 2014, Robinson et al., 2010), ZINB models are typically used for scRNA-seq (single cell RNA-seq) data (van den Berge et al., 2018, Risso et al., 2018).

RNA-seq data sets are not the only count data. In fact, counts data are found in many fields such as business, health, insurance, social sciences, etc. The initial motivation of this study came from an RNA-seq dataset with an excess of zeroes also known as zero inflated data. Models such as zero-inflated Poisson (ZIP) or zero-altered Poisson (or the hurdle model) have been used across several fields to handle excessive zeros in data sets. The difference between

ZIP and the hurdle models resides in how they the treat the types of zeros. The hurdle model assumes a zero-truncated Poisson, i.e. the distribution of the response variable cannot be null, whereas the count process of ZIP can result in zero (Zuur et al, 2009). One main assumption of Poisson regression is the equality of the mean and variance. In reality, this assumption does not always hold. Researchers often turn to Negative Binomial models when faced with over-dispersion in their data even in the non-zero portion of the distribution. Unlike the Poisson distribution with a single parameter (μ), the Negative Binomial distribution contains an additional parameter to account for over-dispersion. Therefore, the zero-inflated negative binomial (ZINB) model and zero-altered negative binomial (ZANB) model are implemented to handle both zero-inflation and over-dispersion.

Non-Parametric Methods for RNA-Seq Data

Nonparametric methods are a way to finesse the difficulty of modelling counts. Without relying on underlying distributional assumptions, they can give reliable results on a vast variety of data sets. As Lehmann stated, because of this property, nonparametric statistics are good alternatives to parametric statistics under non-normal conditions. Although nonparametric tests are robust to departures from normality, they do still require the assumptions of independence of observations, random data selection, and a continuous distribution of data (Kerlinger & Lee, 2000).

As discussed above, existing methods assume an underlying parametric distribution. Issues with parametric distributions arise when the distributional assumptions do not hold which may have negative impact on the performance of the model, and that is often the case especially for large sample sizes where outliers usually exist (Li and Tibshirani, 2013). Yang, Arul and Hui proposed a non-parametric approach (rSeqNP) for testing DE genes and differential splicing

(DS) from RNA-seq data. rSeqNP is an R package that extends non-parametric approach for detecting DE (Li and Tibshirani, 2013) and aims at detecting both DE and DS. Using simulation methods, the authors found that their proposed method controlled Type I error rate and achieved good statistical power for moderate sample and effect sizes (Yang et al, 2015). In addition to not being subject to a parametric underlying distribution, rSeqNP is also flexible in handling various types of experimental designs. One possible limitation of rSeqNP is the fact that it relies on expression estimates for genes. However, the major drawback of rSeqNP is that it exhibits low power for small sample sizes (Yang et al, 2015).

Permutation tests are not without pitfalls especially when sample sizes are unbalanced or data are skewed (Chihara and Hesterberg, 2011). In the case of two--sided hypothesis tests with an unbalance design, when the two treatment groups exhibit different spread, the permutation test is not robust (Chihara and Hesterberg, 2011). Unfortunately, crucial problems related to permutation tests are yet to be addressed, and there is an apparent absence of warnings in the literature about the combined effect of skewness and unbalanced designs (William and Brinljey, 2022).

CHAPTER 3. SIMULATING RNA-SEQ DATA USING MONTE CARLO SIMULATION

In this chapter, we will describe the data generation process and method used to conduct our analysis as well as the results obtained from the Monte Carlo simulation.

Methodology

Data Simulation Overview

Our main goal for conducting this simulation study is to demonstrate that simple permutation is a valid candidate for detecting differentially expressed genes in RNA-seq data sets when compared to the traditional parametric methods. We first started in our preliminary research by assessing Type I error and Power rates (more details are provided below) under various conditions. Afterward, we will be estimating True Positive (TP) and False Positive (FP) rates when the parametric methods' underlying distributions are met. We will then violate the underlying distribution assumptions and obtain the corresponding estimated True Positive (TP) and False Positive (FP) rates.

We simulated 5,000 genes with two groups each with equal sample sizes ranging from 7 to 30. We then set twenty percent (20%) of the simulated genes to be differentially expressed (the two group means are different by a defined effect size) and the other eighty percent (80%) are equally expressed (the two groups have equal means). For clarity, we called the first group the Control group and the second the Treatment group. We used SAS for this simulation and considered four main data-generating distributions namely: Normal, Poisson, Zero Inflated Poisson and Negative Binomial distribution. By definition, the rejection rate is estimated by counting the number of times the null hypothesis was rejected and dividing it by 5000.

Specifically, we simulated data from Normal, Poisson, Negative Binomial and Zero Inflated Poisson distribution with a mean equal to 30. For the normal distribution, we set the

standard deviation to be equal to 5. Moreover, Zero Inflated data was generating as a mixture of two data-generating processes: the Poisson and Uniform distributions. We chose a 20% level of zeroes in the data and then randomly assigned zero to a sample from the uniform distribution with 20% probability of success and the rest of the sample is from Poisson with mean 30. The process just described was carried out for varying levels of sampling effort for each gene. For our study, we are exploring a randomized balanced design (equal sample size for the Control and Treatment group) with sample size levels from 5, 7, 10, 12, 15, 20 and 30.

The list of parameters used for our simulation study is provided in details below with their respective definitions:

- Gene: the number of genes simulated for each underlying distribution; 5,000 genes were used in this study.
- N_C: number of Control specimens.
- N_T: number of Treatment specimens.
- Lambda_C: Poisson parameter for the Control group.
- Lambda_T: Poisson parameter for the Treatment group.
- Mu_C: the mean of the normal distribution for the Control group.
- Mu_T: the mean of the normal distribution for the Treatment Group.
- Sigma_C: the standard deviation of Control gene population.
- Sigma_T: the standard deviation of Treatment gene population
- N_Perms: number of random permutations for the permutation test.

Differential Gene Expression Assessment

Whenever the observed difference or change in read counts or expression levels between the two conditions of an RNA-seq data set (assuming two groups Control and Treatment) is

statistically significant, the gene is declared to be differentially expressed (DE). Therefore, it is important to find the underlying distribution of the data when fitting a parametric method to identify differentially expressed genes. In practice, researchers do not always know the statistical distribution of the data and a violation could lead to an incorrect detection.

In this present study, the focus is to investigate the differential gene expression analysis based on the permutation test and how it compares with traditional parametric method used for gene expression analysis. The framework for this simulation study is as follow: We simulated 5,000 genes of two groups (control and Treatment) each of size n from the same underlying distribution. For our simulated data to exhibit the features of a true RNA-seq dataset, 80% of the data is set to be equally expressed while 20% is set to be differentially expressed. For the differentially expressed genes, we considered 0.5σ and 1σ effect sizes. By 0.5σ effect size we refer to the mean difference between a pair of gene in Control and Treatment is equal to half its standard deviation. For example, suppose that we have a $\mu_C = 30$ and $\sigma_C = 5$ for the Control group; a 0.5σ effect size will correspond to $\mu_T = \mu_C - 0.5\sigma = 30 - 2.5 = 27.5$ and standard deviation $\sigma_T = 5$ for the Treatment group such that $\mu_T - \mu_C = 2.5$. In general, as the effect sizes increase it becomes easier to detect any difference in means; namely, the detection rate of the test increases as well.

To detect DE genes, we then fit the models (T-test, Poisson, Negative Binomial, Permutation test and Zero Inflated Poisson regression) to each gene and count the number of times the null hypothesis for the test below was rejected:

$$H_0: \mu_C = \mu_T$$

$$H_a: \mu_C \neq \mu_T$$

The rejection rate (rejection rate or p-value refer to how often the null hypothesis was rejected) for each underlying distribution is computed by dividing the total count of null hypothesis that was rejected by 5,000 (number of simulated genes). For a 5% significance level we computed the rejection rate for each combination of underlying distributions and sampling efforts.

True Positive Rate

The True Positive rate (TPR; power), also called sensitivity, is the probability that a gene that is declared to be differentially expressed is actually differentially expressed. The rate is computing by tallying the true DE genes from the list of genes declared to be DE genes by a fitted model over the total number of the simulated True DE genes (1000). The TPR was calculated as follow:

$$TPR = \frac{TP}{TP+FN}$$

Where TP is true positive, FN is false negative

False Positive Rate

The False Positive rate, also often called Type I error in statistics, is when an equally expressed (EE) gene is falsely declared as DE gene by a fitted model. It is calculated as the ratio of the number genes wrongly classified as DE genes over the total number of actual negative events (EE genes).

$$FPR = 1 - Specificity = \frac{FP}{FP + TN}$$

Where FP is false positive, TN is true negative

We use the True Positive and False Positive rate to measure the accuracy of our fitted model. If the difference in True Detection rate is not too large, the model that minimizes the false positive rate may be of interest. This may vary from one field to another. For some researchers

and type of studies, controlling the false positive rate is a very important. Therefore, a model that consistently keeps the false positive rate very low is preferred.

Proposed Test

In this study, our main goal is to provide evidence that simple permutation test is a valid and comparable alternative for analyzing RNA-seq data. Permutation and Randomization are often interchangeable; however, the distinction between the two varies among the statistical community (Christensen & Zabriskie, 2021). Some authors consider randomization test as an approximate permutation test that takes only a random sample of all possible permutation (Christensen & Zabriskie, 2021). Others however, differentiate permutation as those based on the assumption of random sampling from two identical population distribution while randomization are based on the assumptions of random assignment of group labels (Onghena, 2018). We refer to permutation test regardless of: (i) whether groups are obtained by random assignment or random sampling, and (ii) whether the groups are obtained by taking full permutation or partial random sample (Christensen & Zabriskie, 2021).

We are simulating data from Normal, Poisson, Negative Binomial and Zero Inflated Poisson distribution and then fit traditional parametric methods used for the simulated genes and then compare the results to that of simple permutation test. We do not expect the permutation test to be superior; if the permutation test yields a power that is close enough to the gold standard then that is sufficient. Particularly, we are interested in how Permutation test compares to the traditional methods with respect to controlling the False Positive rate while yielding a competitive True Detection rate.

For the simple permutation test, we defined the test statistic as follows:

- When sampling from Normal, Poisson distribution and Negative Binomial

Consider our hypothesis:

$$H_0: \mu_C = \mu_T$$

$$H_a: \mu_C \neq \mu_T$$

Where μ_C and μ_T represent the means of the control and treatment groups respectively and $\delta = \mu_C - \mu_T$ the true difference in the means. We define $\bar{C}$ and $\bar{T}$ the means obtained from the samples from the control group (C) and the treatment group (T). The test statistics for our test is given as:

$$D = \bar{C} - \bar{T}$$

Under the null hypothesis, the expected value of D E(D) = 0. Moreover, suppose that the test statistic from our sample is defined as $d = \bar{c} - \bar{t}$. By definition, a two-tailed p-value based on $\bar{C} - \bar{T}$ is:

$$\text{p-value} = \Pr(|\bar{C} - \bar{T}| \geq |\bar{c} - \bar{t}| \mid H_0 \text{ is true}) = \Pr(|D| \geq |d| \mid H_0 \text{ is true})$$

$$= \Pr(D \leq -|d| \mid H_0 \text{ is true}) + \Pr(D \geq |d| \mid H_0 \text{ is true})$$

For our simulation study, we sampled 5,000 genes with two groups (Control and Treatment). Unlike parametric models assuming known distribution, such as the Gaussian or Student's t, to calculate p-value for the permutation test, we first build the sampling distribution for our test statistic $D = \bar{C} - \bar{T}$ by aggregating all (or a sample of) possible values of the test statistic obtained by rearranging the group labels associated with the observations. For small samples, less or equal to 10, we did a full permutation test whereas for large sample size, greater or equal to 12, we did a partial permutation (or randomized permutation) of B size. Here we set B level at 5000 randomly selected permutations. The p-value is then obtained by finding the proportion of the permutation distribution that is at least as extreme as the actual test statistic. Specifically, we first compute the test statistic from the original sample as $d = \bar{c} - \bar{t}$. And then,

the reference distribution is built by computing the B permutation test statistics d_1, …, d_B where $d_i = \overline{c_i} - \overline{t_i}$, and $\overline{c_i}$ and $\overline{t_i}$ are the means of the control and treatment groups respectively when labels have been reassigned according to the i^{th} permutation, or random shuffling, of the group labels. Our two-sided permutation test p-value is then calculated as:

$$p - value = \frac{\sum_1^B I_{(|di| \geq |d|)}}{B}$$

$$P - value = \frac{\sum_1^B I_{(di \leq -|d|)}}{B} + \frac{\sum_1^B I_{(di \geq |d|)}}{B}$$

Where I, the indicator function, is equal to 1 when the condition is true and 0 otherwise.

- When sampling from Zero Inflated Poisson distribution

For genes simulated from the Zero inflated distribution, we had to find a way to incorporate the zero in the mean estimation. We sampled 5,000 genes from a mixture of Poisson and Uniform distribution. We set the probability of success to 20% for the Uniform distribution. In other words, there is a 20% chance of observing a 1 from the Uniform distribution. If the event is 1 then we set y = 0 otherwise y = Pois (Lambda= λ). Now our random variable y follows a modified version of regular Pois (λ) distribution known as Zero Inflated Poisson (ZIP) distribution with a density function defined as:

$$P(Y = k) = \begin{cases} \pi + (1 - \pi)\exp(-\lambda) & \text{if } k = 0 \\ (1 - \pi)\exp(-\lambda)\frac{\lambda^k}{k!} & \text{if } k \in \{1, 2, \dots\} \end{cases}$$

Where $0 \leq \pi \leq 1$ and $\lambda \geq 0$.

The parameter π gives the extra probability thrust at the value 0. When it vanishes, ZIP (π, λ) reduces to Pois (λ).

The mean and variance of the ZIP are:

$$E(Y) = (1 - \pi)\,\lambda$$

$$V(Y) = (1 - \pi)\,(1 + \pi\,\lambda)\,\lambda$$

We can easily derive λ from the expected value of the ZIP as follow:

$$\lambda = \frac{E(Y)}{(1 - \pi)}$$

From the simulated samples we can estimate the mean of the ZIP and the parameter π. Recall that π is the probability that y is 0. Therefore, we could estimate π by dividing the number of 0 by the sample size.

The test statistics for the simple permutation test remains the same as described previously with a slight modification in the permutation process. We quickly realized that a simple full permutation or a partial permutation of the control and treatment group lead to a very poor Type I error. Both Control and Treatment contains 0 level at approximately 20% of their size. Shuffling the two groups could cause the data to be skewed with all the zeros or most of the zeros to be in one group and nothing or very few zeros in the other. After many trial and error, we proposed a modified permutation method. Instead of shuffling all the observations, we held the proportion of zero constant in each group and permuted only the non-zero observations. Once we obtained all the permuted samples from the modified permutation process, we then computed the means using the adjusted mean formula discussed above.

The test statistic is computed in the same fashion as described above. The only difference here is the permutation procedure. When sampling from Normal Poisson and Negative Binomial distribution we shuffled all the data in Control and Treatment group. However, when sampling from Zero Inflated Poisson, we modified the permutation procedure by keeping the proportion of zero constant in each group and permuting only the non-zero values.

Zero Inflated Poisson

Most popular statistical software such as SAS and STATA have implemented packages to fit ZIP and ZINB regression models. However, they are not yet available in SPSS. In SAS, one may use either PROC GENMOD or PROC COUNTREG for ZIP and ZINB models. For our simulation study, we chose to work with SAS and specifically we used the GENMOD procedure to fit the data.

The ZIP model has two components, one component is to model the probability of being the structural zeros ρ using the logistic regression and the other component is to model the Poisson mean μ. Specifically we have the ZIP model defined as follow:

$$\text{Logit}\,(\rho_i) = U_i^T\beta_U,\ \log\,(\mu_i) = V_i^T\beta_v,$$

where the subscript i indicates the i^{th} observation, U and V (which may overlap) represent two sets of explanatory variables that will be linked to ρ and μ, respectively, in the ZIP model, and β_U and β_V are the vectors of parameters for the logistic and Poisson components.

In the ZIP model above, the likelihood of structural zero is model by the logit link function; other link functions can also be used such as probit and complementary loglog. Therefore, the existence of structural zeros not only leads to a more complex distribution, but also provides an additional link function for modeling the effect of explanatory variables for the occurrence of such zeros. In other words, the ZIP model enables us to better understand the effect of covariates by distinguishing the effects of each specific covariate on structural zeros (likelihood for having no expression) and on the count response (mean of Poisson for a non-null expression).

Simulation Study Outline

A synopsis of this simulation study is provided below in detail:

1. Detection rate assessment with two groups: $\mu_C = \mu_T$
   a) Four underlying distributions: Normal, Poisson, Negative Binomial and Zero Inflated Poisson distribution.
   b) Seven levels of sampling effort: $n_C = n_T = 5$, $n_C = n_T = 7$, $n_C = n_T = 10$, $n_C = n_T = 12$, $n_C = n_T = 15$, $n_C = n_T = 20$ and $n_C = n_T = 30$ for each combination of treatments and underlying distribution.
   c) Fitted models: T-test, Permutation test, Poisson, NB, ZIP, ZINB.
   d) Simulation was conducted taking 5,000 genes for each combination of parameters defined above.

SAS Code

The SAS and R code used for this simulation study is provided in Appendix C. We provided the SAS code for the simulation from one underlying distribution since to get the others we just changed the underlying distribution to the desired distribution and everything else remains the same.

Monte Carlo Simulation Results

In this section, we will first cover the results obtained from our preliminary research, and then discuss the results from the detection rate assessment as well as for the TP and FP rate comparison for each combination of underlying distribution, fitted model and sample sizes we considered in the case of two populations scenarios.

Preliminary Research Results

In our preliminary study, we assessed the Type I error as well as the Power for of the fitted models for each underlying distribution we considered in the case of two populations scenarios.

Type I error Assessment: $\mu_C = \mu_T$

Type I error is defined as the probability of rejecting the null hypothesis when in fact it is true. For this simulation study we set our significance level alpha at $\alpha = 0.05$ and expect the estimated Type I error to be in the neighborhood of 5%.

We sampled two random groups (Control and Treatment) from underlying distribution using three different models (Normal, Poisson and ZIP). The estimated Type I error was obtained based on these random samples assuming equals sample sizes for both Control and Treatment. Furthermore, we set the mean of the two groups to be equals to 30 and explored different sampling effort from 5, 7, 10, 12, 15, 20 and 30. For the Normal distribution we set the variance to be equals to 25 and for the Zero Inflated Poisson we set π=0.2; the mean remains the same for all underlying distribution (mean=30).

The estimated Type I error when sampling from Normal Distribution (Normal with mean 30 and standard deviation 5 for both Control and Treatment) is summarized in Table 1 below. The results suggest that, for all combination of sampling efforts and fitted models (T test, Permutation, Poisson and NB), Type I error is maintained near the stated rate of alpha $\alpha = 0.05$. Similar results are obtained when sampling from Poisson distribution. As shown in Table 2, Type I error is maintained near 0.05 significance level with the exception of the Permutation test being a little conservative for sample sizes equal to 5 and 7.

Table 1. Normal samples - rejection rates (%) for fitted models $\mu_C = \mu_T = 30$

Sampling Efforts	Fitted Models			
	T test	Permutation	Poisson	Negative Binomial
$n_C = n_T = 5$	4.73	4.98	5.29	5.38
$n_C = n_T = 7$	4.37	4.70	4.60	4.71
$n_C = n_T = 10$	4.36	4.50	4.48	4.71

Table 2. Poisson samples - rejection rates (%) for fitted models $\mu_C = \mu_T = 30$

Sampling Efforts	Fitted Models			
	T test	Permutation	Poisson	Negative Binomial
$n_C = n_T = 5$	4.37	4.02	4.81	4.81
$n_C = n_T = 7$	4.55	4.18	4.86	4.95
$n_C = n_T = 10$	5.21	4.85	5.37	5.44

Tables 3, 4, 5, 6, 7 and 8 below display estimated Type I error when sampling from Normal and Poisson distribution respectively for large samples (12, 15 and 20) at different random permutation sampling sizes. For small samples ($n_C = n_T \leq 10$) we performed a full permutation on the simulated data set. However, when the sample size is very large ($n_C = n_T \geq 12$) permutation becomes overwhelming. In this case we performed a partial permutation on the simulated data set. For our study, we considered three permutation sizes for our simulated data set: B = 1000, 5000 and 10,000.

For all combination of underlying distributions, sample sizes and number of permutation (B) and fitted models, as shown from table 3 to table 8, Type I error is maintained near the stated significance level of 0.05. Let us note here that for B = 10,000 and sample size of 12 and 15 (Table 8) the permutation test was a little conservative.

Table 3. Normal samples - rejection rates (%) for fitted models $\mu_C = \mu_T = 30$, B=1K

Sampling Efforts	Fitted Models			
	T test	Permutation	Poisson	Negative Binomial
$n_C = n_T = 12$	4.89	5.08	4.91	5.32
$n_C = n_T = 15$	4.87	5.01	4.93	5.20
$n_C = n_T = 20$	5.19	5.36	5.22	5.55

Table 4. Poisson samples - rejection rates (%) for fitted models $\mu_C = \mu_T = 30$, B=1K

Sampling Efforts	Fitted Models			
	T test	Permutation	Poisson	Negative Binomial
$n_C = n_T = 12$	4.90	4.67	5.05	5.13
$n_C = n_T = 15$	5.04	4.76	5.06	5.30
$n_C = n_T = 20$	5.34	5.09	5.38	5.51

Table 5. Normal samples - rejection rates (%) for fitted models $\mu_C = \mu_T = 30$, B=5K

Sampling Efforts	Fitted Models			
	T test	Permutation	Poisson	Negative Binomial
$n_C = n_T = 12$	5.08	5.08	5.14	5.41
$n_C = n_T = 15$	5.38	5.33	5.34	5.64
$n_C = n_T = 20$	5.28	5.28	5.28	5.58

Table 6. Poisson samples - rejection rates (%) for fitted models $\mu C = \mu T = 30$, B=5K

Sampling Efforts	Fitted Models			
	T test	Permutation	Poisson	Negative Binomial
$n_C = n_T = 12$	4.83	4.42	4.83	4.93
$n_C = n_T = 15$	5.16	4.82	5.21	5.43
$n_C = n_T = 20$	4.81	4.38	4.82	5.12

Table 7. Normal samples - rejection rates (%) for fitted models $\mu_C = \mu_T = 30$, B=10K

Sampling Efforts	Fitted Models			
	T test	Permutation	Poisson	Negative Binomial
$n_C = n_T = 12$	5.19	5.39	5.31	5.57
$n_C = n_T = 15$	4.82	4.96	4.83	5.26
$n_C = n_T = 20$	5.18	5.23	5.14	5.62

Table 8. Poisson samples - rejection rates (%) for fitted models $\mu_C = \mu_T = 30$, B=10K

Sampling Efforts	Fitted Models			
	T test	Permutation	Poisson	Negative Binomial
$n_C = n_T = 12$	4.74	4.47	4.86	4.94
$n_C = n_T = 15$	4.75	4.46	4.78	4.99
$n_C = n_T = 20$	4.80	4.52	4.79	4.83

For genes from the ZIP distribution, we only looked at large and B = 1000 number of permutations. From Table 9 below we can conclude that type I error is maintained near the stated alpha value of 0.05 for all fitted model with the exception of NB and ZINB being too conservative.

Table 9. ZIP samples - rejection rates (%) for fitted models $\mu_C = \mu_T = 30$, B=1K

Sampling Efforts	Fitted Models					
	T test	Permutation	ZIP	Poisson	Negative Binomial	ZINB
$n_C = n_T = 12$	5.10	4.78	4.90	3.43	2.10	3.15
$n_C = n_T = 15$	4.41	4.59	5.07	3.41	2.43	3.29
$n_C = n_T = 20$	5.18	4.66	4.85	4.27	3.38	3.18
$n_C = n_T = 30$	5.32	4.94	5.14	4.66	4.04	3.64

The results obtained from the Type I error assessment for each combination of sampling efforts and underlying distributions, suggest that all the models fitted (T test, Poisson, Negative Binomial, Zero Inflated Poisson, Zero Inflated Negative Binomial and Permutation Test) control the Type I errors. Therefore, each of the above models are valid candidates to test the null

hypothesis that the means are equal using RNA-seq data. We will next discuss the results from the power comparison to decide whether a particular model is preferred over the rest.

Power Comparison: $\mu_C \neq \mu_T$

After ensuring that all models maintained Type I error at below or near the stated significance level of α=0.05, we then conducted a power comparison under various conditions to check whether certain models performed better than other did. We considered a 0.5σ effect size for the power comparison. By 0.5σ effect size, we refer to the mean difference between a pair of gene in Control and Treatment is equal to half its standard deviation. For example, suppose that we have a $\mu_C = 30$ and $\sigma_C = 5$ for the Control group; a 0.5σ effect size will correspond to $\mu_T = \mu_C - 0.5\sigma = 30 - 2.5 = 27.5$ and standard deviation $\sigma_T = 5$ for the Treatment group such that $\mu_T - \mu_C = 2.5$. In general, as the effect sizes increase it becomes easier to detect any difference in means; namely, the power of the test increases as well.

- **Effect size: half sigma (0.5σ)**

The power is defined as the probability of rejecting the null hypothesis when in fact it is false. Simulating our observations from two populations with different means and setting the null hypothesis as H_0: $\mu_C = \mu_T$ makes the null hypothesis false. Tallying the number of times each of the models correctly detect the difference in the two groups (rejecting H_0 since there are in fact different and dividing it by 10,000 will give us our estimated powers). This process was repeated for each combination of underlying distribution.

For small sample sizes ($n_C = n_T \leq 10$), when sampling from Normal and Poisson distribution, all fitted models (T test, Permutation, Poisson and NB regression) yielded a comparable power rate as shown in Tables 10-11 and Figures 2-3 respectively. NB is a little bit higher than the other models but the difference is n negligible.

Table 10. Normal samples - rejection rates (%) for fitted models $\mu_C \neq \mu_T$ (Effect size=0.5σ, μ=30)

Sampling Efforts	Fitted Models			
	T test	Permutation	Poisson	Negative Binomial
$n_C = n_T = 5$	9.89	10.45	10.68	10.88
$n_C = n_T = 7$	13.47	14.02	14.04	14.32
$n_C = n_T = 10$	18.20	18.55	18.37	18.98

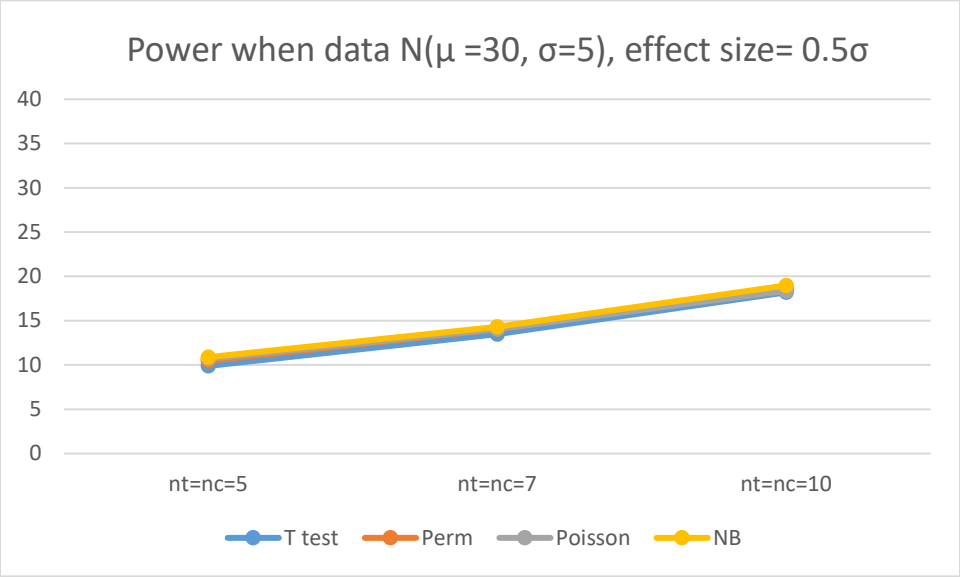

Figure 2. Normal samples - rejection rates (%) for various models $\mu_C \neq \mu_T$ (Effect size=0.5σ, μ=30)

Table 11. Poisson samples - rejection rates (%) for fitted models $\mu_C \neq \mu_T$ (Effect size=0.5σ, μ=30)

Sampling Efforts	Fitted Models			
	T test	Permutation	Poisson	Negative Binomial
$n_C = n_T = 5$	9.96	9.48	11.00	11.12
$n_C = n_T = 7$	14.19	13.47	14.73	14.88
$n_C = n_T = 10$	18.71	17.89	18.99	18.34

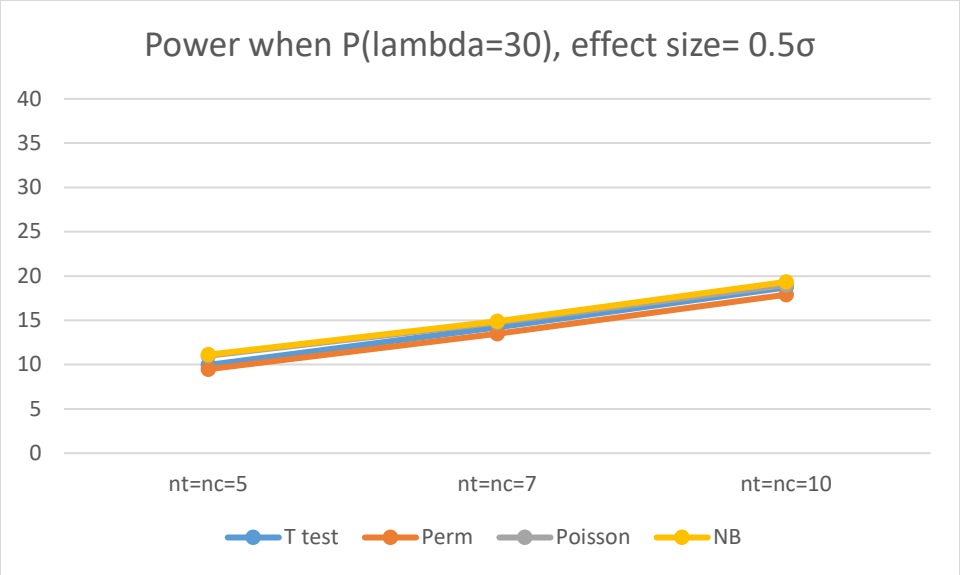

Figure 3. Poisson samples - rejection rates (%) for various models μC≠μT (Effect size=0.5σ, μ=30)

For large samples ($n_C = n_T \geq 12$) we performed a partial permutation for the permutation test (B=1K, 5K, 10K). When sampling from Normal and Poisson distribution, all fitted models (T test, Permutation, Poisson and NB regression) yielded a comparable power rate at all permutation sample sizes level as shown in Tables 12-17 and Figures 4-9.

Table 12. Normal samples - rejection rates (%) for fitted models $\mu_C \neq \mu_T$ (Effect size=0.5σ, μ=30), B=1K

Sampling Efforts	Fitted Models			
	T test	Permutation	Poisson	Negative Binomial
$n_C = n_T = 12$	21.19	21.43	21.13	21.70
$n_C = n_T = 15$	26.10	26.42	26.08	26.78
$n_C = n_T = 20$	32.72	33.04	32.73	33.70

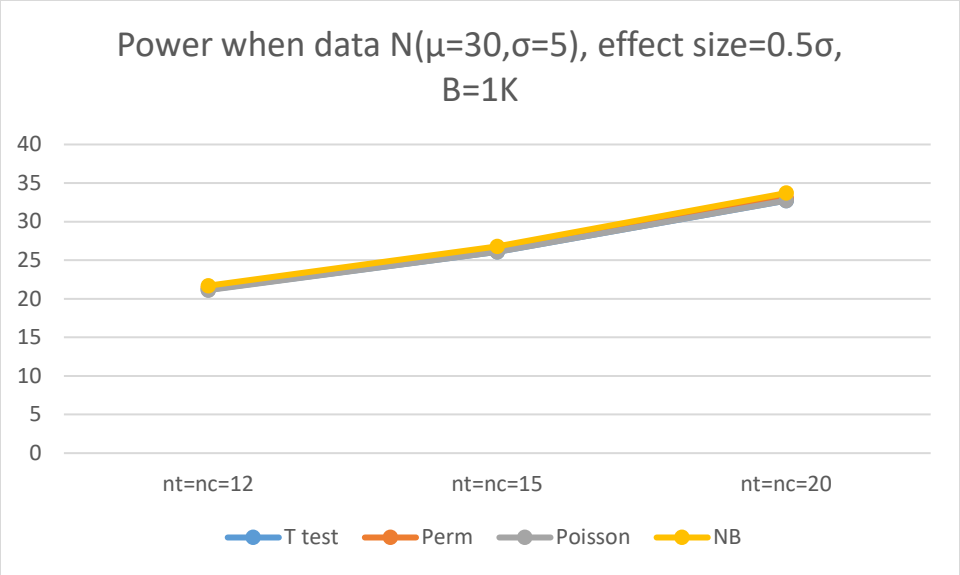

Figure 4. Normal samples - rejection rates (%) for various models μC≠μT (Effect size=0.5σ, μ=30, B=1K)

Table 13. Poisson samples - rejection rates (%) for fitted models $\mu_C \neq \mu_T$ (Effect size=0.5σ, μ=30), B=1K

Sampling Efforts	Fitted Models			
	T test	Permutation	Poisson	Negative Binomial
$n_C = n_T = 12$	22.43	21.57	22.73	23.02
$n_C = n_T = 15$	27.49	26.63	27.73	28.01
$n_C = n_T = 20$	35.09	34.36	35.17	35.52

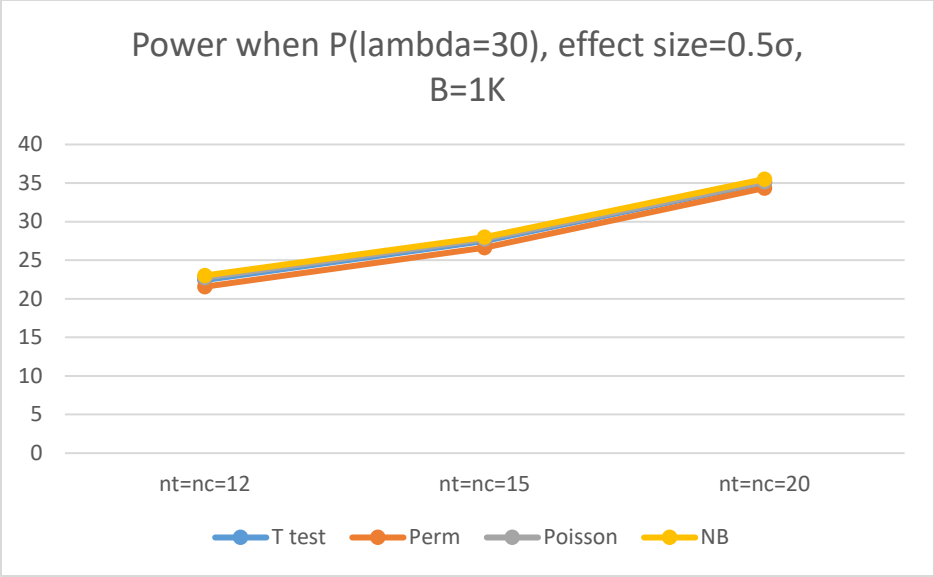

Figure 5. Poisson samples - rejection rates (%) for various models μC≠μT (Effect size=0.5σ, μ=30, B=1K)

Table 14. Normal samples - rejection rates (%) for fitted models $\mu_C \neq \mu_T$ (Effect size=0.5σ, μ=30), B=5K

Sampling Efforts	Fitted Models			
	T test	Permutation	Poisson	Negative Binomial
$n_C = n_T = 12$	20.99	21.35	21.16	21.75
$n_C = n_T = 15$	26.45	26.63	26.43	27.15
$n_C = n_T = 20$	34.11	34.49	34.05	35.26

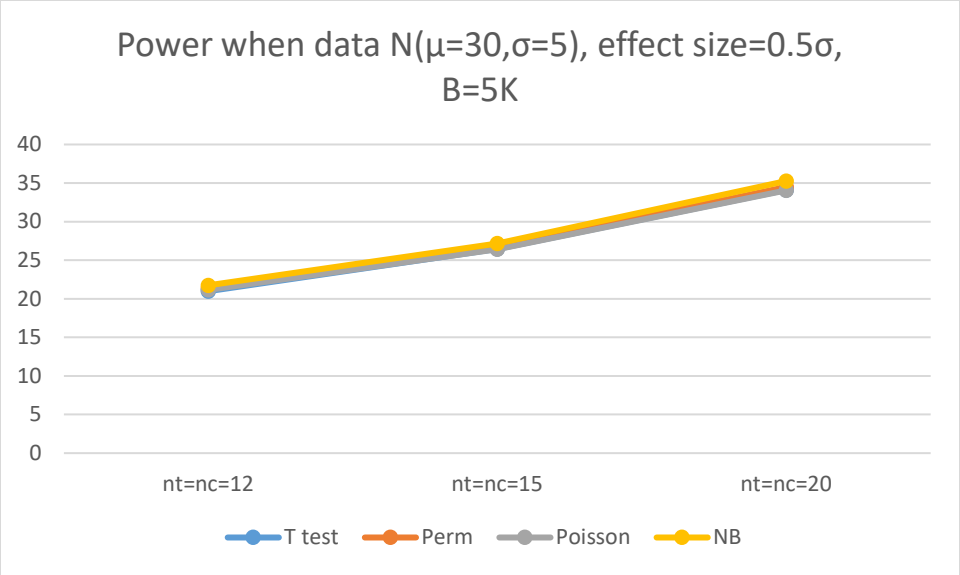

Figure 6. Normal samples - rejection rates (%) for various models μC≠μT (Effect size=0.5σ, μ=30, B=5K)

Table 15. Poisson samples - rejection rates (%) for fitted models $\mu_C \neq \mu_T$ (Effect size=0.5σ, μ=30), B=5K

Sampling Efforts	Fitted Models			
	T test	Permutation	Poisson	Negative Binomial
$n_C = n_T = 12$	22.37	21.59	22.70	22.87
$n_C = n_T = 15$	27.05	26.15	27.13	27.61
$n_C = n_T = 20$	34.89	33.98	34.97	35.49

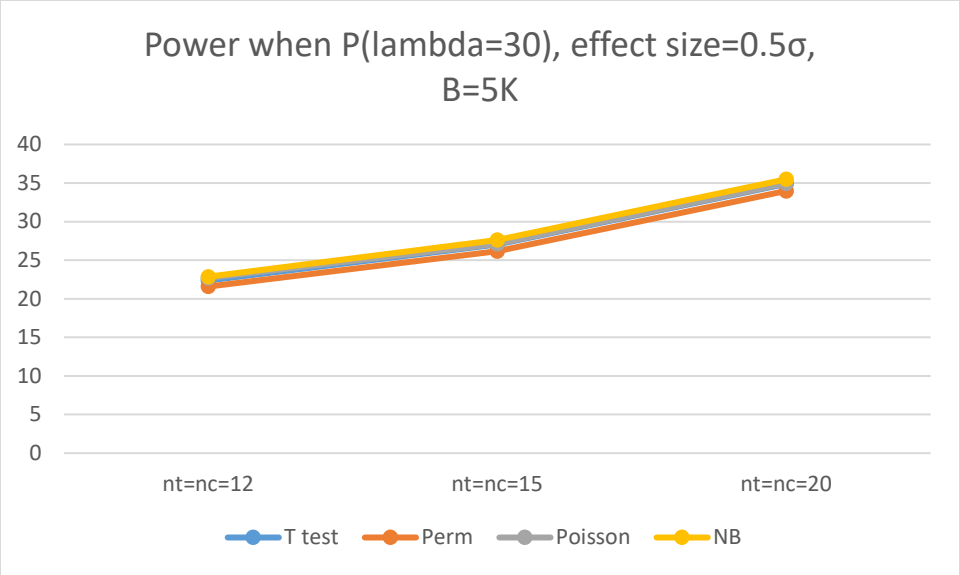

Figure 7. Poisson samples - rejection rates (%) for various models μC≠μT (Effect size=0.5σ, μ=30, B=5K)

Table 16. Normal samples - rejection rates (%) for fitted models $\mu_C \neq \mu_T$ (Effect size=0.5σ, μ=30), B=10K

Sampling Efforts	Fitted Models			
	T test	Permutation	Poisson	Negative Binomial
$n_C = n_T = 12$	21.88	22.11	21.93	22.47
$n_C = n_T = 15$	25.59	25.88	25.65	26.49
$n_C = n_T = 20$	33.43	33.45	33.30	34.23

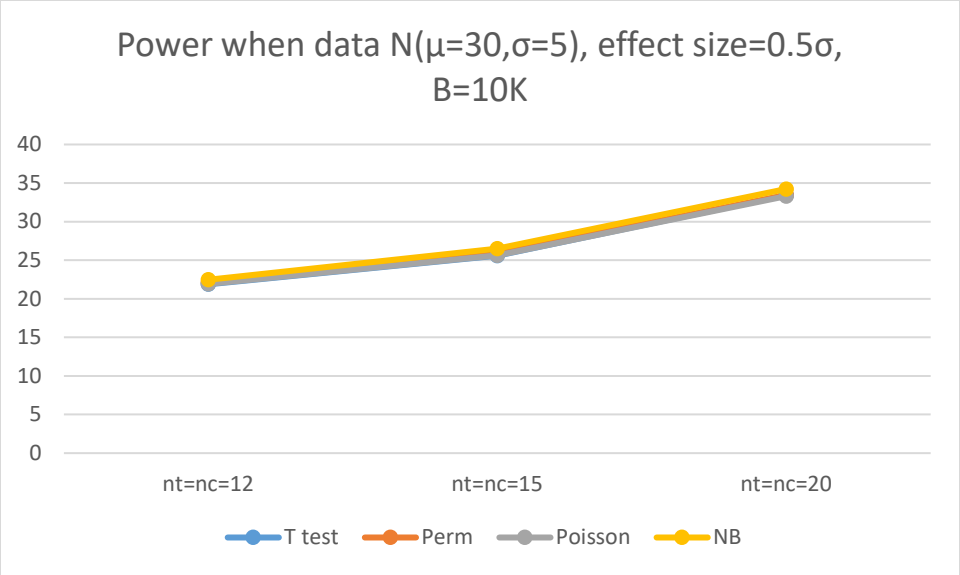

Figure 8. Normal samples - rejection rates (%) for various models μC≠μT (Effect size=0.5σ, μ=30, B=10K)

Table 17. Poisson samples - rejection rates (%) for fitted models $\mu_C \neq \mu_T$ (Effect size=0.5σ, μ=30), B=10K

Sampling Efforts	Fitted Models			
	T test	Permutation	Poisson	Negative Binomial
$n_C = n_T = 12$	22.28	21.49	22.46	22.72
$n_C = n_T = 15$	26.47	25.59	26.73	27.20
$n_C = n_T = 20$	35.04	33.78	35.11	35.28

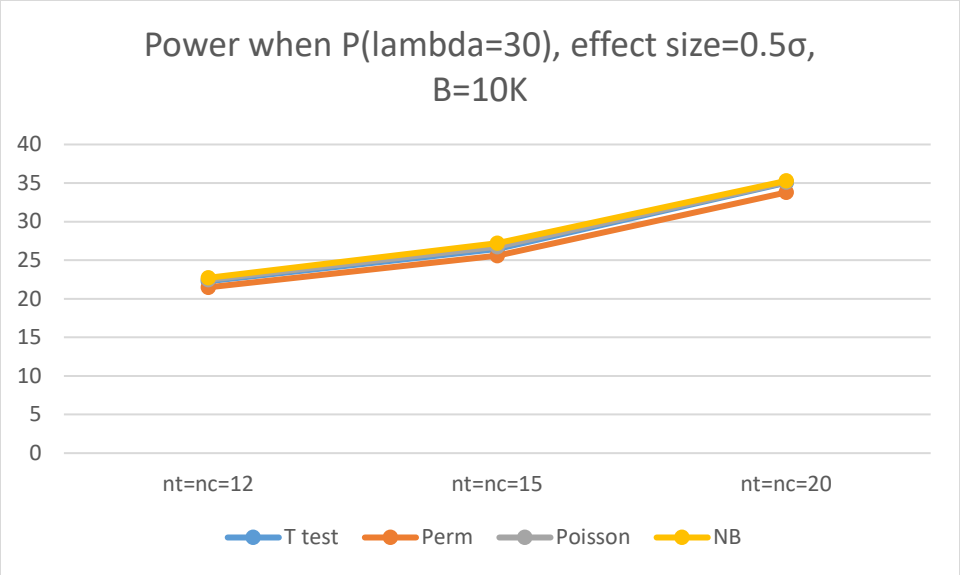

Figure 9. Poisson samples - rejection rates (%) for various models μC≠μT (Effect size=0.5σ, μ=30, B=10K)

Table 18 below summarizes the power rate from all fitted model (T test, Permutation, Poisson and NB regression) when we sample from ZIP distribution. We can see that the permutation test and the ZIP yielded comparable power whereas T test, Poisson, Negative Binomial and ZINB displayed a poor power with T test exhibiting the lowest power rate.

Table 18. ZIP Samples - rejection rates (%) for fitted models $\mu_C \neq \mu_T$ (Effect size=0.5σ, μ=30, π=0.2), B=1K

Sampling Efforts	Fitted Models					
	T test	Permutation	ZIP	Poisson	Negative Binomial	ZINB
$n_C = n_T = 12$	7.65	17.39	20.36	6.09	4.56	14.46
$n_C = n_T = 15$	7.63	21.28	23.56	6.48	5.36	17.19
$n_C = n_T = 20$	8.80	28.00	30.29	7.78	6.91	22.80
$n_C = n_T = 30$	10.71	40.54	42.18	9.99	9.26	33.80

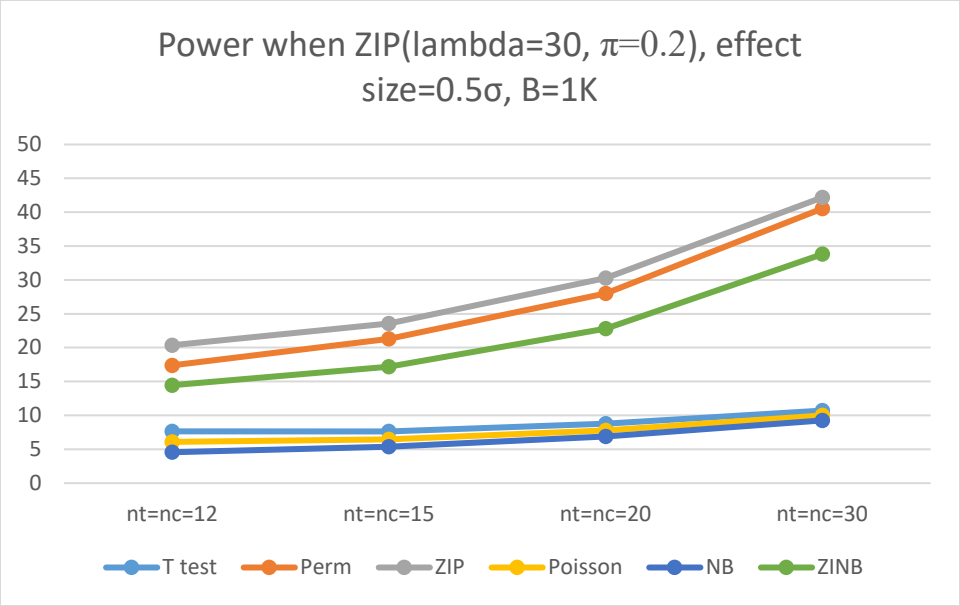

Figure 10. ZIP samples - rejection rates (%) for various models $\mu_C \neq \mu_T$ (Effect size=0.5σ, μ=30, π=0.2, B=1K)

Differential Expressed Genes Assessment

Whenever a fitted model detected a significant difference between the mean of the two conditions of our simulated RNA-seq data (Control vs Treatment), the gene is declared to be differentially expressed. The detection rate is then obtained by tallying the total DE genes over the simulation size (5000).

We sampled two random groups (Control and Treatment) from underlying distribution using three different models (Normal, Poisson and Negative Binomial). The estimated detection rate, true positive and false positive rate were obtained based on these random samples under two different designs: Balanced (equal sample sizes for both Control and Treatment; n1=n2) and Unbalanced (unequal sample sizes for both Control and Treatment; n1≠n2). Furthermore, we set 80% of the simulated data to be equally expressed (μ1=μ2=30) while the other 20% are set to be DE genes ($\mu_1 \neq \mu_2$) with a 1σ effect size.

Various sample sizes were considered from 5, 7 to 30. Our early results suggested that permutation tests suffer from the granularity issue with relatively small sample sizes. Note that

the smallest possible value for the p-value is 1/N where N represents the number of permutation possible. We refer to 1/N as the granularity limit. For large sample sizes, since N is too large for the permutation test to be computationally achievable, we take a partial permutation B for our simulation and therefore the minimum value the p-value can take is 1/B. it is important to note that 1/B can much larger than the permutation limit 1/N. To obtain a small p value, a larger number of B may be required to an accurate estimate of it. Due to these issues, the results are very poor compared to the parametric methods regardless of the design. For the rest of the study we will focus on the following sample sizes: 10, 15, 20, 25 and 30.

Balanced Design: $n_1=n_2$

When sampling from Normal distribution (Normal with mean $\mu=30$ and standard deviation $\sigma=5$) with a 1σ effect size, all fitted models (T test, Permutation, Poisson and NB regression) yielded a comparable detection rate as shown in Table 19. As the sample size increases, the number of genes declared to be differentially expressed increases as well. However, we are interested in the quality of the model to detect the True differentially expressed genes with minimal error. Negative Binomial tend to detect more DE genes compare the other models but the difference is very small. For instance, for sample sizes $n_C= n_T=25$, NB correctly detected 924 genes out of 1000, Poisson was second with 923 and Permutation was third with 922 genes. The difference is about two extra genes. It appears that all fitted models perform relatively well when it comes to detecting True DE genes when the effect size is relatively large.

The last column in Table 19 provides the false positive rate which indicate the proportion of genes that were incorrectly declared to be DE genes. Overall, permutation and Poisson consistently had a lower False Positive rate for all sample sizes compared to Negative Binomial as shown in Figure 11. Permutation and Poisson were somewhat similar, with Poisson slightly

lower when the sample sizes are 15 and 20. Permutation however, had a lower FP rate when the sample sizes are 25 and 30. So it was not consistent to decide which of Permutation and Poisson keep lower FP rate. But clearly both controlled FP rate lower than Negative Binomial. As expected, t-test on the other hand had the overall lower FP rate. This is expected as the underlying distribution is Normal but t-test also had relatively the smallest True Positive rate (slightly lower than the others).

Table 19. Normal samples - detection rates (%) for fitted models

Fitted Models	Sampling Efforts	Actual	Detected	True DE Gene	True Positive rate (%)	False Positive rate (%)
T-test	$n_C = n_T = 10$	1000	722	546	54.60	4.40
	$n_C = n_T = 15$	1000	968	763	76.30	5.13
	$n_C = n_T = 20$	1000	1078	862	86.20	5.40
	$n_C = n_T = 25$	1000	1119	921	92.10	4.95
	$n_C = n_T = 30$	1000	1149	968	96.80	4.53
Permutation	$n_C = n_T = 10$	1000	733	551	55.10	4.55
	$n_C = n_T = 15$	1000	977	765	76.50	5.30
	$n_C = n_T = 20$	1000	1080	866	86.60	5.35
	$n_C = n_T = 25$	1000	1121	922	92.20	4.98
	$n_C = n_T = 30$	1000	1146	967	96.70	4.48
Poisson	$n_C = n_T = 10$	1000	725	543	54.30	4.55
	$n_C = n_T = 15$	1000	968	761	76.10	5.18
	$n_C = n_T = 20$	1000	1076	861	86.10	5.38
	$n_C = n_T = 25$	1000	1124	923	92.30	5.03
	$n_C = n_T = 30$	1000	1149	967	96.70	4.55
Negative Binomial	$n_C = n_T = 10$	1000	748	557	55.70	4.78
	$n_C = n_T = 15$	1000	1001	766	76.60	5.88
	$n_C = n_T = 20$	1000	1091	864	86.40	5.68
	$n_C = n_T = 25$	1000	1138	924	92.40	5.35
	$n_C = n_T = 30$	1000	1180	966	96.60	5.35

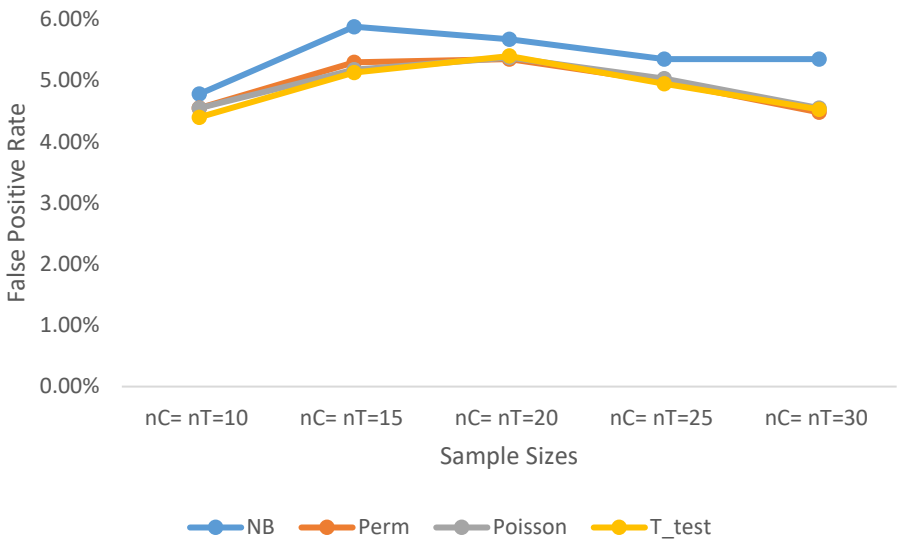

Figure 11. Comparing FP rate across fitted model when sampling from Normal $n_1=n_2$

Table 20 below summarizes the simulation results when the underlying distribution is Poisson (Poisson with mean lambda=30). We see similar pattern as normal distribution samples. The difference in true DE genes is very small across fitted models with Poisson and Negative Binomial closely detecting about the same number. As the sample sizes increase, the models detected more True DE genes and Permutation becomes very close in True detection rate to Poisson and Negative Binomial. The performance of permutation is very satisfying and competitive to its parametric counterparts. Interestingly, Permutation consistently had the lowest False Positive rate across all sample sizes with T-test second as displayed in Figure 12. This is in line with our early studies that showed that Permutation does a better job at controlling False Positive rates - lower than Poisson and Negative binomial.

Table 20. Poisson samples - detection rates (%) for fitted models

Fitted Models	Sampling Efforts	Actual	Detected	True DE Gene	True Positive rate (%)	False Positive rate (%)
T-test	$n_C = n_T = 10$	1000	768	578	87.80	4.75
	$n_C = n_T = 15$	1000	987	778	77.80	5.23
	$n_C = n_T = 20$	1000	1089	888	88.80	4.93
	$n_C = n_T = 25$	1000	1167	958	95.80	5.23
	$n_C = n_T = 30$	1000	1158	976	97.60	4.55
Permutation	$n_C = n_T = 10$	1000	749	569	56.90	4.50
	$n_C = n_T = 15$	1000	959	767	76.70	4.80
	$n_C = n_T = 20$	1000	1078	888	88.80	4.75
	$n_C = n_T = 25$	1000	1152	956	95.60	4.90
	$n_C = n_T = 30$	1000	1151	974	97.40	4.43
Poisson	$n_C = n_T = 10$	1000	780	582	58.20	4.95
	$n_C = n_T = 15$	1000	990	775	77.50	5.38
	$n_C = n_T = 20$	1000	1090	891	89.10	4.98
	$n_C = n_T = 25$	1000	1170	959	95.90	5.28
	$n_C = n_T = 30$	1000	1162	976	97.60	4.65
Negative Binomial	$n_C = n_T = 10$	1000	781	581	58.10	5.00
	$n_C = n_T = 15$	1000	988	775	77.50	5.33
	$n_C = n_T = 20$	1000	1100	893	89.30	5.18
	$n_C = n_T = 25$	1000	1173	955	95.50	5.45
	$n_C = n_T = 30$	1000	1165	976	97.60	4.73

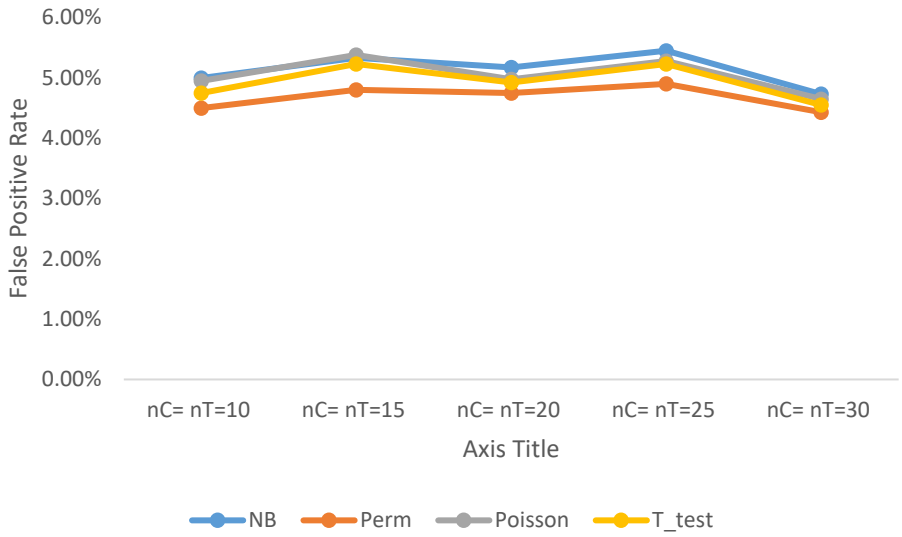

Figure 12. Comparing FP rate across fitted model when sampling from Poisson $n_1 = n_2$

When we sample from Negative Binomial distribution (NB with mean $\mu=30$ and standard deviation $\sigma=8$) we see similar results as Poisson samples discussed above. As exhibited in Table 3 below, all fitted models yielded comparable True DE genes detection rate. Permutation

however consistently kept the False Positive rate lower across all sample sizes (see Figure 13). Negative Binomial tends to overestimate probably due to a larger variation in the data which also lead to slightly higher False Positive rate. Thus far, Permutation not only appeared to be competitive when compared to Poisson and Negative Binomial regression but most importantly it consistently had a better control of the False Positive for all combination of sample sizes and underlying distribution with a few exceptions with Normal data where we saw Poisson slightly lower when the sample sizes were 15 and 20.

Table 21. Negative Binomial samples - detection rates (%) for fitted models

Fitted Models	Sampling Efforts	Actual	Detected	True DE Gene	True Positive rate (%)	False Positive rate (%)
T-test	$n_C = n_T = 10$	1000	874	680	68.00	4.85
	$n_C = n_T = 15$	1000	1042	845	84.50	4.93
	$n_C = n_T = 20$	1000	1143	937	93.70	5.08
	$n_C = n_T = 25$	1000	1164	969	96.90	4.88
	$n_C = n_T = 30$	1000	1205	989	98.90	5.40
Permutation	$n_C = n_T = 10$	1000	869	676	67.60	4.83
	$n_C = n_T = 15$	1000	1032	841	84.10	4.78
	$n_C = n_T = 20$	1000	1127	937	93.70	4.75
	$n_C = n_T = 25$	1000	1155	967	96.70	4.70
	$n_C = n_T = 30$	1000	1193	989	98.90	5.10
Poisson	$n_C = n_T = 10$	1000	881	677	67.70	5.10
	$n_C = n_T = 15$	1000	1036	834	83.40	5.05
	$n_C = n_T = 20$	1000	1143	939	93.90	5.10
	$n_C = n_T = 25$	1000	1167	967	96.70	5.00
	$n_C = n_T = 30$	1000	1203	989	98.90	5.35
Negative Binomial	$n_C = n_T = 10$	1000	882	674	67.40	5.20
	$n_C = n_T = 15$	1000	1039	834	83.40	5.13
	$n_C = n_T = 20$	1000	1143	937	93.70	5.15
	$n_C = n_T = 25$	1000	1168	966	96.60	5.05
	$n_C = n_T = 30$	1000	1204	989	98.90	5.38

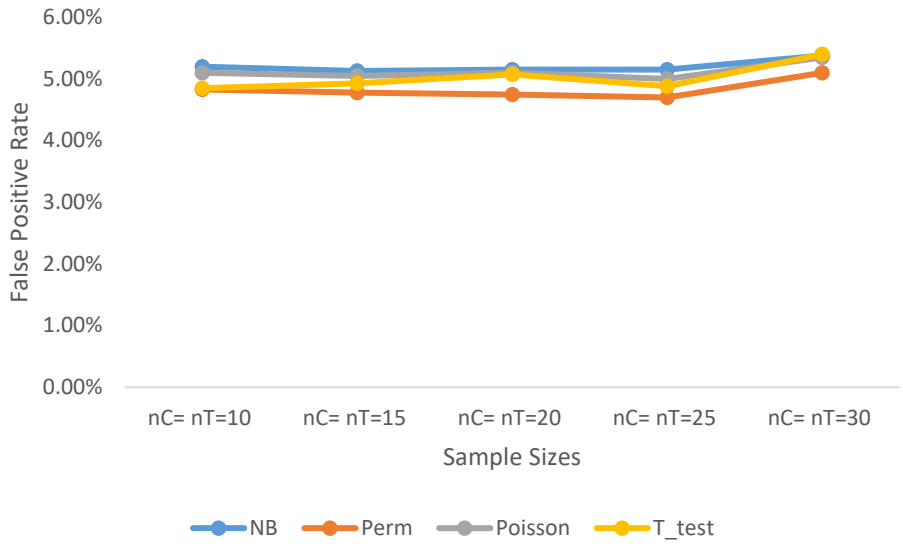

Figure 13. Comparing FP rate across fitted model when sampling from NB $n_1=n_2$

Unbalanced Design: $n_1 \neq n_2$

Given the competitive performance of Permutation test and its capability to control False Positive rate in the samples balanced design scenario, we decided to run a few unbalanced data sets and assess Permutation performance compared to Poisson and Negative Binomial. The results obtained are summarized in Tables 22, 23, 24 when sampling from Normal, Poisson and Negative Binomial distribution respectively. We see similar trend as for the balanced scenario. All models exhibited comparable True DE genes detection rate. Referring to Figures 14, 15 and 16, Permutation kept the False Positive the lowest overall.

Table 22. Normal samples - detection rates (%) for fitted models

Fitted Models	Sampling Efforts	Actual	Detected	True DE Gene	True Positive rate (%)	False Positive rate (%)
T-test	n_C=15 n_T=10	1000	806	621	62.10	4.63
	n_C=20 n_T=10	1000	881	686	68.60	4.88
	n_C=20 n_T=15	1000	989	793	79.30	4.90
	n_C=30 n_T=15	1000	1081	872	87.20	5.23
Permutation	n_C=15 n_T=10	1000	811	628	62.80	4.58
	n_C=20 n_T=10	1000	894	707	70.70	4.68
	n_C=20 n_T=15	1000	989	794	79.40	4.88
	n_C=30 n_T=15	1000	1075	880	88.00	4.88
Poisson	n_C=15 n_T=10	1000	808	628	62.80	4.50
	n_C=20 n_T=10	1000	889	704	70.40	4.63
	n_C=20 n_T=15	1000	985	795	79.50	4.75
	n_C=30 n_T=15	1000	1074	877	87.70	4.93
Negative Binomial	n_C=15 n_T=10	1000	822	629	62.90	4.83
	n_C=20 n_T=10	1000	904	710	71.00	4.85
	n_C=20 n_T=15	1000	1002	796	79.60	5.15
	n_C=30 n_T=15	1000	1094	876	87.60	5.45

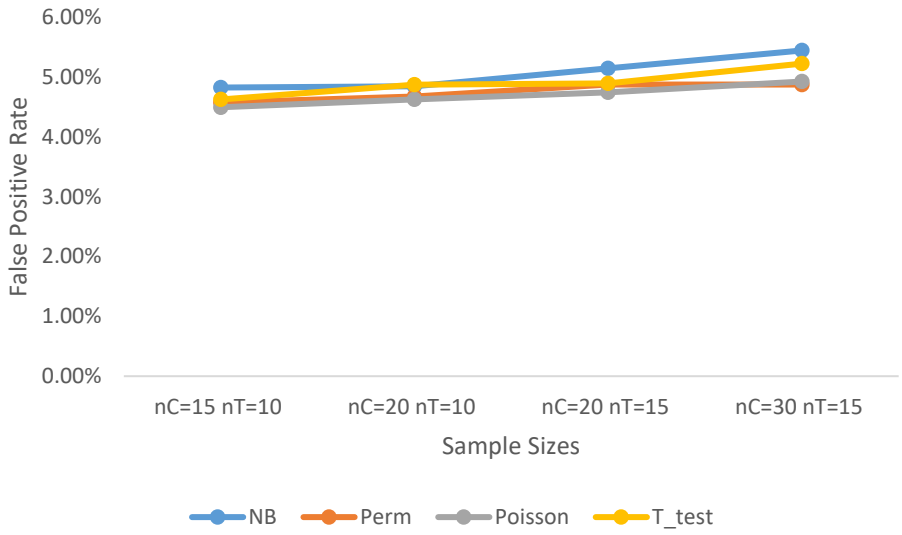

Figure 14. Comparing FP rate across fitted model when sampling from Normal $n_1 \neq n_2$

Table 23. Poisson samples - detection rates (%) for fitted models

Fitted Models	Sampling Efforts	Actual	Detected	True DE Gene	True Positive rate (%)	False Positive rate (%)
T-test	n_C=15 n_T=10	1000	929	723	72.30	5.15
	n_C=20 n_T=10	1000	1047	848	84.80	4.98
	n_C=30 n_T=15	1000	1086	892	89.20	4.85
Permutation	n_C=15 n_T=10	1000	900	706	70.60	4.85
	n_C=20 n_T=10	1000	1038	834	83.40	5.10
	n_C=30 n_T=15	1000	1070	893	89.30	4.43
Poisson	n_C=15 n_T=10	1000	910	711	71.10	4.98
	n_C=20 n_T=10	1000	1049	842	84.20	5.18
	n_C=30 n_T=15	1000	1076	894	89.40	4.55
Negative Binomial	n_C=15 n_T=10	1000	914	711	71.10	5.08
	n_C=20 n_T=10	1000	1049	834	83.40	5.38
	n_C=30 n_T=15	1000	1086	897	89.70	4.73

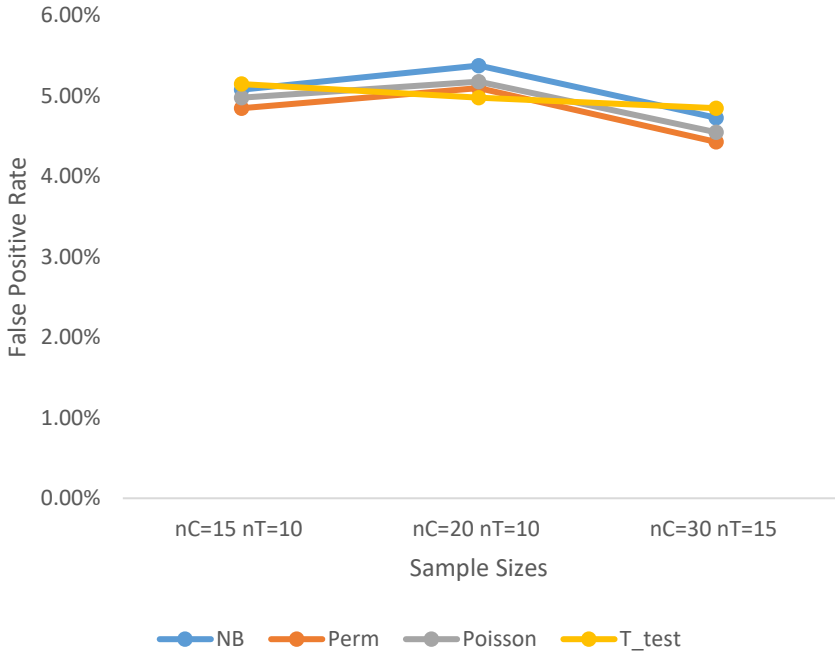

Figure 15. Comparing FP rate across fitted model when sampling from Poisson $n_1 \neq n_2$

Table 24. Negative Binomial samples - detection rates (%) for fitted models

Fitted Models	Sampling Efforts	Actual	Detected	True DE Gene	True Positive rate (%)	False Positive rate (%)
T-test	n_C=15 n_T=10	1000	942	739	73.90	5.08
	n_C=20 n_T=15	1000	1071	885	88.50	4.65
	n_C=30 n_T=15	1000	1142	925	92.50	5.43
Permutation	n_C=15 n_T=10	1000	950	750	75.00	5.00
	n_C=20 n_T=15	1000	1069	890	89.00	4.48
	n_C=30 n_T=15	1000	1131	931	93.10	5.00
Poisson	n_C=15 n_T=10	1000	948	743	74.30	5.13
	n_C=20 n_T=15	1000	1073	886	88.60	4.68
	n_C=30 n_T=15	1000	1145	932	93.20	5.33
Negative Binomial	n_C=15 n_T=10	1000	959	749	74.90	5.25
	n_C=20 n_T=15	1000	1079	889	88.90	4.75
	n_C=30 n_T=15	1000	1149	934	93.40	5.38

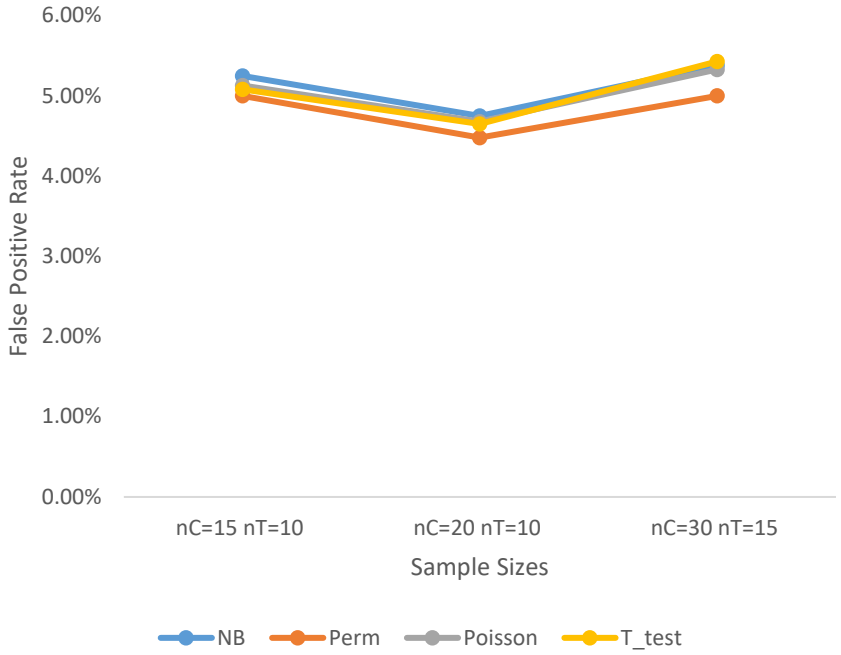

Figure 16. Comparing FP rate across fitted model when sampling from NB $n_1 \neq n_2$

Monte Carlo Simulation Conclusion

Our simulation study using Monte Carlo simulation suggest that permutation is a valid competitive model for analyzing RNA-seq data. Most importantly, the results show for both balanced and unbalanced designs, not only did Permutation yield similar True positive rates as

Poisson and Negative Binomial regression, but it consistently controlled the False Positive rate lower than its parametric counterparts.

RNA-seq data are generally assumed to follow either Poisson or Negative Binomial distribution. And traditional models developed for analyzing such data assume these distributions without providing a way to check whether the underlying assumption are met or not. A slight violation could lead to a substantial wrong estimate. Our theoretical studies provide evidence that for both Poisson and Negative Binomial samples, Permutation is robust and offer a good control of the False Positive rate.

Before we generalize our theoretical findings, in the next chapters we will now simulate RNA-seq data using SimSeq. Recent development in RNA-seq data simulation suggest that given the complexity of RNA-seq data, a simulation method that conserves such complexity is advisable. Our expectation is that the results from SimSeq data will be consistent with Monte Carlo simulation results.

CHAPTER 4. SIMULATING RNA-SEQ DATASET USING SIMSEQ

In this chapter, we will discuss the data generation process and method used to conduct our analysis as well as the results obtained from the SimSeq procedure.

Simseq Simulation Overview

SimSeq is a nonparametric simulation algorithm approach proposed by Benidt and Nettleton for the construction of RNA-seq dataset with two independent treatment groups (Benidt & Nettleton, 2015).

The Simseq Algorithm

The SimSeq algorithm is available as an R package to simulate matrix of RNA-seq read counts. To create differential expression, it subsamples columns from a large existing RNA-seq dataset and then swaps single read counts within genes adjusted by a correction factor (Benidt & Nettleton, 2015).

The SimSeq algorithm takes essentially three main sets of inputs: a large RNA-seq dataset Y with two independent treatment groups (Control & Treatment); a vector c of computed normalization factors with one element for each column of the source dataset; the number of equally expressed (EE) genes G_0 and differentially expressed (DE) genes G_1 in the simulated dataset where $G_0+G_1 \leq G$ and the number of columns n in each of the two treatment groups (Control & Treatment) in the simulated matrix where $n \leq \{N_1, \lfloor N_2/2 \rfloor\}$ where $\lfloor \cdot \rfloor$ is the floor function. The SimSeq algorithm outputs a dataset of RNA-seq read counts with G_0 EE genes and G_1 DE genes with n columns in each of two independent treatment groups (Control & Treatment) (Benidt & Nettleton, 2015). The simulation procedure algorithm as described by Benidt and Nettleton (Benidt & Nettleton, 2015) is given below:

1. For each $g \in \mathcal{G}$, calculate a *P* value from a test of differential expression using the Wilcoxon Rank Sum test.
2. Given the set of calculated *P* values, calculate the local false discovery rate (fdr) for each gene (Strimmer, 2008a, b) using the fdr package.
3. A vector of probability sampling weights *w* is computed as one minus the local fdr for each gene *g* scaled to sum to unity.
4. Randomly select G_1 genes to be DE from $\mathcal{G}$ without replacement according to the vector of probability sampling weights *w* and denote this set $\mathcal{G}_1$.
5. Randomly select G_0 genes to be EE from $\mathcal{G}/\mathcal{G}_1$ without replacement according to equal weights and denote this $\mathcal{G}_0$. Let $\mathcal{G}^* \equiv \mathcal{G}_0 \cup \mathcal{G}_1$ be the set of all EE genes and DE genes chosen in steps 1 and 2.
6. Randomly select one column y without replacement from the first treatment group of Y. Subset y down to the set of genes $\mathcal{G}^*$ to create the column x_1. Assign x_1 to simulated treatment group 1.
7. Randomly select one column without replacement from each treatment group in Y and denote these two columns as Y_1 and Y_2. Let c_1 and c_2 be their corresponding multiplicative normalization factors from c.
8. Subset the two columns Y_1 and Y_2 to the set of genes $\mathcal{G}^*$
9. Create the column x_2 in the following way. For each gene $g \in \mathcal{G}^*$

   Let

$$x_{2g} = \begin{cases} y_{1g} & if\ g \in G_0 \\ \left\lfloor y_{2g} * \frac{c_1}{c_2} + 0.05 \right\rfloor & if\ g \in G_1 \end{cases}$$

Where $\lfloor \cdot \rfloor$ is the floor function, so that $y_{2g} * c_1/c_2$ is rounded to the nearest integer. Let x_2 be the vector whose entries are $\{x_{2g}: g \in \mathcal{G}^*\}$. Assign x_2 to simulated treatment group 2. (Note that c_1/c_2 is a correction factor to allow the read counts in x_2 to have a consistent normalization factor.)

1. Repeat steps 6-9 a total of *n* times with columns sampled without replacement across each iteration.

Source Dataset

We used the GTEx dataset as our source dataset. It contains 17382 genes and about 54 tissues. We retrieve the Pancreas and Stomach tissues and test whether genes are differentially expressed between these two tissues. In the SimSeq simulation algorithm (R package), we simulated 5000 genes and set 20% of the simulated data to be DE genes.

Differential Gene Expression Assessment

For parametric methods, it is crucial to find a distribution to approximate the nature of the differential gene expression data. The traditional R packages generally used to assess DE genes in RNA-seq dataset often assume a Poisson or Negative Binomial underlying distribution. A slight variation could lead to wrong estimates thus inflating the False Positive rate. In this section, we are comparing the performance of popular RNA-seq data DE genes analysis tools namely: edgeR, DESeq2, Limma, Voom and LFC to our proposed Permutation test. Note that edgeR, DESEq2, Limma, Voom and LFC are all available as R package.

edgeR is a Bioconductor software package proposed by Robinson et al. (2010) for analyzing differential expression of replicated count data. It uses an overdispersed Poisson model to account for both biological and technical variability. To improve the reliability of inference, empirical Bayes approach are used to estimate gene specific dispersion parameters. DESeq2, a

successor to DESeq, is very similar to edgeR with a slight difference. DESeq2 is an R package for differential analysis of count data; it uses shrinkage estimation for dispersions and fold changes to improve stability and interpretability of estimates. Deseq2 focuses on the strength rather than the simple presence of differential expression. Limma, another R package for analyzing gene expression. Limma is an acronym for "linear models for microarray data" and comprises functionalities for fitting a larger class of statistical models called: linear models.

We used SAS to run our Permutation test by importing the simulated dataset obtained from SimSeq in R into a csv file and then read the csv file in SAS. Note that we are first filtering the data in R and then use the filtered data for the permutation test to ensure that all considered models are fitted on the same genes.

False Discovery Rate Control

When conducting multiple hypothesis tests there is an increase in the chance of making a False Positive. The higher the number of tests, the higher the probability of making a type I error. In this case the False Discovery Rate (FDR) is preferable error rate measure. FDR is defined as the rate of true nulls among the rejected hypotheses. An α FDR rate implies that on average, the proportion of false discoveries among all discoveries is at most α. Hypothesis testing procedures aiming to control FDR tend to be considerably more powerful than procedures aiming to control Family Wise Error such as Bonferroni. For this study we controlled FDR rate at 10%.

SimSeq Simulation Results

Table 25 below summarizes the results obtained from our fitted model when sampling from SimSeq. When the sample sizes are 10 and 15, DESEq2 has the highest True Detection rate second by LFC. Permutation has a comparable detection rate to Voom. For sample sizes greater than 15, DESEq2 still has the highest True Detection rate however, the difference is very small

compared to Permutation. Permutation, however, exhibited a competitive detection rate compared to edgeR, Limma, and Voom.

The last column of Table 25 provides the False Positive rate for each combination of fitted models sample sizes. As illustrated on Figure 17, Permutation consistently displayed the lowest False Positive Rate across all fitted models and sample sizes. DESEq2 has the highest False Positive rate with LFC the second highest overall.

Although DESEq2 had the highest True Detection rate, it also had the highest False Positive Rate. Permutation not only did turn out to be competitive compared to edgeR, Limma and Voom but it also consistently kept the False Positive very low.

Table 25. SimSeq dataset - detection rates (%) for fitted models

Fitted Models	Sampling Efforts	Actual	Detected	True DE Gene	True Positive rate (%)	False Positive rate (%)
Permutation	$n_C = n_T = 10$	1000	663	645	64.50	0.45
	$n_C = n_T = 15$	1000	695	678	67.80	0.43
	$n_C = n_T = 20$	1000	816	809	80.90	0.18
	$n_C = n_T = 25$	1000	846	841	84.10	0.13
	$n_C = n_T = 30$	1000	895	884	88.40	0.28
edgeR	$n_C = n_T = 10$	1000	795	727	72.70	1.70
	$n_C = n_T = 15$	1000	817	749	74.90	1.70
	$n_C = n_T = 20$	1000	852	820	82.00	0.80
	$n_C = n_T = 25$	1000	926	828	82.80	2.45
	$n_C = n_T = 30$	1000	976	859	85.90	2.93
DESEq2	$n_C = n_T = 10$	1000	1017	884	88.40	3.33
	$n_C = n_T = 15$	1000	995	894	89.40	2.53
	$n_C = n_T = 20$	1000	989	952	95.2	0.93
	$n_C = n_T = 25$	1000	1056	939	93.90	2.93
	$n_C = n_T = 30$	1000	1118	964	96.40	3.85
Limma	$n_C = n_T = 10$	1000	766	706	70.60	1.50
	$n_C = n_T = 15$	1000	774	733	73.30	1.03
	$n_C = n_T = 20$	1000	848	835	83.5	0.33
	$n_C = n_T = 25$	1000	892	845	84.50	1.18
	$n_C = n_T = 30$	1000	938	877	87.70	1.53
Voom	$n_C = n_T = 10$	1000	747	691	69.10	1.40
	$n_C = n_T = 15$	1000	755	710	71.00	1.13
	$n_C = n_T = 20$	1000	838	826	82.60	0.30
	$n_C = n_T = 25$	1000	874	838	83.80	0.90
	$n_C = n_T = 30$	1000	941	872	87.20	1.73
LFC	$n_C = n_T = 10$	1000	929	858	85.80	1.78
	$n_C = n_T = 15$	1000	930	870	87.00	1.50
	$n_C = n_T = 20$	1000	1004	939	93.90	1.63
	$n_C = n_T = 25$	1000	935	877	87.70	1.45
	$n_C = n_T = 30$	1000	987	926	92.60	1.53

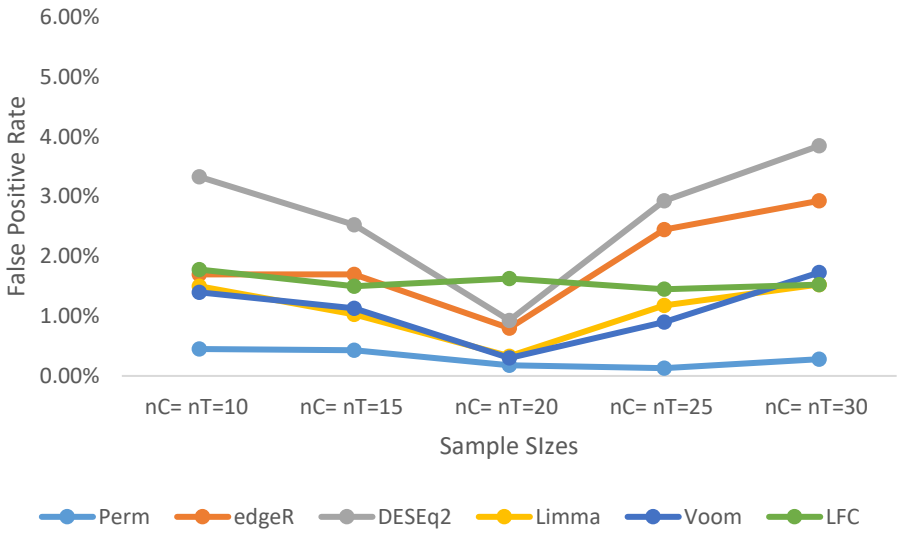

Figure 17. Comparing FP rate across fitted model when sampling SimSeq

CHAPTER 5. CASE STUDY

We obtained a raw RNA-seq dataset using GTEX data. The data comprises 10000 genes from two groups of tissues (pancreas and stomach) each with 15 replicates. Our goal is to test whether genes are differentially expressed between the two tissues. We will fit permutation tests, edgeR, DESeq2 and Limma.

The table below summarizes the results from each fitted model. Out of 10000 genes, DESeq2 detected 3039 to be DE followed LFC with 2662 DE genes and Permutations detected 2554 DE genes. From the Venn diagram in Figure 18, all the fitted models together detected 1967 DE genes. Permutation tests and DESeq2 jointly detected 2252 DE genes; edgeR and Permutation tests jointly detected 2010 DE genes; Limma and Permutations tests detected jointly 2029 DE genes; DESeq2 and edgeR jointly detected 2322.

The results obtained from permutation tests are satisfying. Although DESeq2 detected more DE genes, our simulation study suggests that permutation tests consistently minimize FPR and DESeq2 sometimes tends to overestimate.

Table 26. DE genes per fitted models

	Fitted models				
	Perm	DESeq2	edgeR	Limma	LFC
DE genes	2554	3039	2504	2472	2662

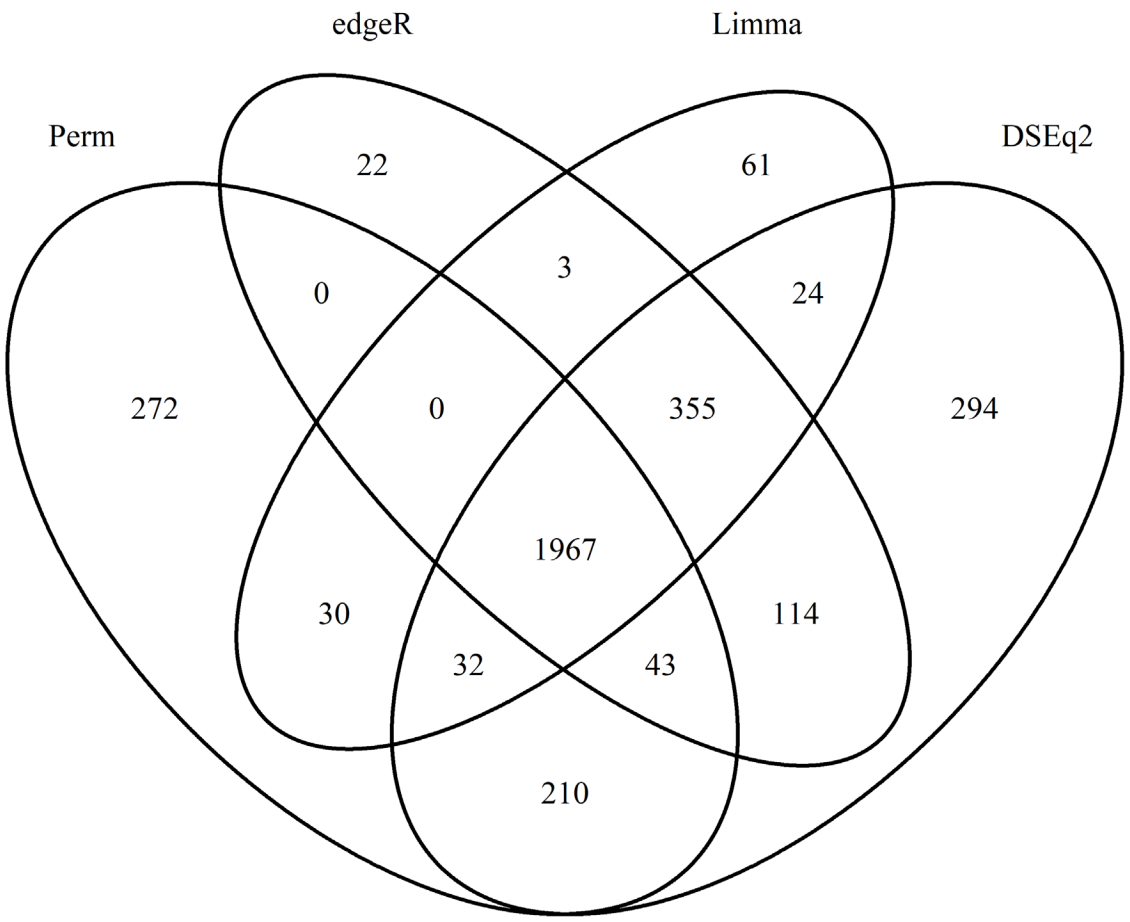

Figure 18. Venn Diagram of DE genes per fitted model

CHAPTER 6. GENERAL CONCLUSION

This dissertation explored two distinct simulation methods for RNA-seq data to compare the performance of Permutation test to traditional models used for analyzing differential gene expression in RNA-seq datasets under different scenarios. We first simulated in Chapter 3 RNA-seq datasets using Monte Carlo simulation in SAS assuming several theoretical distributions namely: Normal (μ=30, σ=5), Poisson (λ=30) and Negative Binomial (μ=30, σ=8). Though many researchers suggest that theoretical distributions are not representative of the complex nature of RNA-seq data, they do provide a reference to which we can test the assumptions made by models developed for analyzing RNA-seq data. Generally, methods used to assess differential gene expression in RNA-seq data assumed a Poisson or Negative Binomial underlying distribution. However, researchers do not always check whether the underlying assumptions of the distribution are met and this could cause biased results with unknown consequences.

For the two sample balanced design case, our preliminary research suggests that all fitted models maintained Type I error near the stated value of 5% with Permutation being conservative for small sample sizes (5 & 7) when sampling from Poisson. Furthermore, we found that permutation exhibited comparable power with Negative binomial being slightly higher. Moreover, the theoretical approach using Monte Carlo simulation suggests that when sampling from a Normal distribution, the true detection rate increases as we increase the sample sizes as well. However, the False positive rate is not always controlled equally the same. Negative Binomial exhibited the highest False Positive rate while Permutation, Poisson and T-test had the lowest False Positive rate with T-test slightly lower than the former two. When sampling from Poisson and Negative Binomial, all fitted models had comparable True Detection rate. Permutation however performed slightly better when sampling from negative Binomial as the

sample sizes increase. Furthermore, Permutation consistently displayed lower False Positive rates compared across all sample sizes. Revisiting our research questions:

- How do True Positive rate and False Positive (FP) rates compare across T-test, Permutation, Poisson, Negative Binomial, EdgeR, Limma-Voom and DESeq when applied to RNA-Seq data?
- Does the sample size from the permutation impact the quality of the results obtained?

From the theoretical simulation, we can deduct that Permutation is a competitive alternative to its parametric counterpart used for gene expression analysis. Additionally, we found that all fitted model controlled false positive rate at 10% with permutation tests being somewhat conservative. It is worthwhile to note that we are only considering sample sizes 10, 15, 20, 25 and 30 as our preliminary results suggested that Permutation suffers from granularity issues with small samples ($n<10$). As the sample sizes increased the performance also improved.

For the two sample unbalanced design, our theoretical results suggest similar outcome observed in the balance design scenario with a slight difference. We considered the following unequal sample sizes: $n_C=15$ & $n_T=10$, $n_C=20$ & $n_T=10$, $n_C=20$ & $n_T=15$, $n_C=30$ & $n_T=15$ when sampling from Normal and $n_C=15$ & $n_T=10$, $n_C=20$ & $n_T=10$, $n_C=30$ & $n_T=15$ when sampling from Poisson and Negative Binomial. All fitted models exhibited comparable true detection rate with Negative Binomial slightly higher and Permutation slightly lower when sampling from Poisson distribution. When sampling from Normal, Permutation and Poisson had comparable false positive rate but lower than Negative Binomial .When sampling from Poisson distribution, Permutation controlled best the False Positive rate except for sample sizes $n_C=20$ & $n_T=10$ where T-test had a slight lower False positive rate. When sampling from Negative Binomial, Permutation's False Positive rate was consistently the lowest for all sample sizes considered.

Overall, Permutation test was not always competitive for detection of DE genes but controlled the false positive rate at 10% for the unbalance scenario. We also saw a positive correlation between the sample sizes and the performance. Specifically, larger sample sizes are encouraged for permutation tests in both two-sample balanced and unbalanced scenarios.

In Chapter 4, we used SimSeq; a data based simulation method proposed by Benidt and Nettlton for simulating RNA-seq dataset from a large source RNA-seq data. Data based simulation methods such SimSeq among many simulate dataset that closely match the complex structure of real RNA-seq data (Benidt & Nettleton, 2015).

The results obtained from the SimSeq dataset under two-sample balance design scenario suggest that overall DESEq2 has the highest True Detection rate second by LFC for all sample sizes considered. For sample sizes 10 and 15, Permutation had the lowest True Detection rate, roughly about 6% lower than DESEq2 and 5% lower than edgeR and Limma and about 3% lower than Voom. When we increase the sample sizes to 20, 25 and 30, Permutation became very competitive. In fact, permutation had a comparable detection rate with Limma and Voom for sample sizes equal to 20 and 25 and slightly higher true detection rate when the sample size was 30. Moreover, Permutation was slightly higher than edgeR for sample sizes 25 and 30. The False Positive however is consistently the lowest for permutation test for all sample sizes with Negative Binomial having the highest FP rate except when sample size was 20 LFC was higher.

From both simulation methods (standard Monte Carlo and SimSeq), we can state that permutation controls fairly well the false positive at 10% and was somewhat conservative for all sample sizes. As we increased the sample sizes permutation had an improved performance and became very competitive at detecting true DE genes. For a researcher concerned with controlling

false positive rate, we will strongly recommend consideration of the permutation test given sufficient sample sizes ($n \geq 10$).

It is noteworthy to point out that during our simulation study we ran into few issues that restricted our ability to control some of the parameters of interest. With Monte Carlo simulation, we were able to control the underlying distributions, the proportion of zeroes present in each sample as well as the effect size. However, RNA-seq data are much more complex and samples obtained from the Monte Carlo simulation do not necessarily mimic such complexity. Therefore, we decided to use SimSeq, which simulate data by subsampling from a larger real RNA-seq dataset. Although SimSeq conserve the complex nature of RNA-seq data, it does present some limitations. With SimSeq, we were unable to control the effect sizes of simulated DE genes as it depends on the presence of available indicator that already indicates differential expression in real dataset. In addition, we were unable to specify the proportion of zeroes in each sample sizes. Furthermore, genes declared to be DE using the SimSeq algorithm are not guaranteed to actually be DE genes as this depend on the quality of the indicator variable. SimSeq works better with homogenous dataset and very poorly with heterogeneous data. We were unable to explore unbalance sample sizes. Despite these limitations, our proposed method outcomes from the Monte Carlo Simulation and the SimSeq were consistent.

REFERENCES

Anders, S., Huber, W. (2010). Differential expression analysis for sequence count data. Genome Biology, 11(10). https://doi.org/10.1186/gb-2010-11-10-r106

Benidt, S., & Nettleton, D. (2015). SimSeq: a nonparametric approach to simulation of RNA-sequence datasets. Bioinformatics, 31(13), 2131-2140.

Blair, R. C. (1981). A reaction to "Consequences of failure to meet assumptions underlying the fixed effects analysis of variance and covariance". Review of Educational Research, 51(4), 499-507.

Blair, R. C., & Higgins, J. J. (1985). Comparison of the power of the paired samples t test to that of wilcoxon's signed-ranks test under various population shapes. Psychological Bulletin, 97(1), 119-128.

Christensen, W. F., & Zabriskie, B. N. (2021). When your permutation test is doomed to fail. The American Statistician, 1-11.

Edgington, E. S. (1995). Randomization Tests. (3rd ed). New York, NY: Marcel Dekker.

Good, P. (1994). Permutation Tests: A Practical Guide to Resampling Methods for Testing Hypotheses. New York, NY: Springer-Verlag.

Hunter, M. A. & May, R. B. (1993). Some myths concerning parametric and nonparametric tests. Canadian Psychology, 34(4), 384-389.

Li, J., & Tibshirani, R. (2013). Finding consistent patterns: a nonparametric approach for identifying differential expression in RNA-Seq data. Statistical methods in medical research, 22(5), 519-536.

Li, J., Witten, D. M., Johnstone, I. M., & Tibshirani, R. (2012). Normalization, testing, and false discovery rate estimation for RNA-sequencing data. Biostatistics, 13(3), 523-538.

Li, W. V. and Li, J. J. (2018). An accurate and robust imputation method scimpute for single-cell rna-seq data. Nature communications 9, 997.

Love, M. I., Huber, W., and Anders, S. (2014). Moderated estimation of fold change and dispersion for rna-seq data with deseq2. Genome biology 15(12), 550.

May, R. B., Masson, E.J., & Hunter, M. A. (1989). Randomization tests: Viable alternatives to normal curve tests. Behavior Research Methods, Instruments, & Computers, 21(4), 482-483.

Mielke, P. W. & Berry, K. J. (2001). Permutation methods: A distance function approach. New York, NY: Springer.

Onghena, P. (2018). Randomization tests or permutation tests? A historical and terminological clarification. Randomization, masking, and allocation concealment, 209-228.

Potvin, C. & Roff, D. (1993). Distribution-free and robust statistical methods: Viable alternatives to parametric statistics? Ecology, 74(6), 1617-1628.

Robinson, M. D., McCarthy, D. J., & Smyth, G. K. (2009). edgeR: a Bioconductor package for differential expression analysis of digital gene expression data. Bioinformatics, 26(1), 139-140.

Robinson, M. D., & Smyth, G. K. (2007). Moderated statistical tests for assessing differences in tag abundance. Bioinformatics, 23(21), 2881-2887.

Sawilowsky, S. S. & Blair, R. C. (1992). A more realistic look at the robustness and type II error properties of the t test to departures from population normality. Psychological Bulletin, 111(3), 352-360.

Schurch, Nick & Schofield, Pietà & Gierliński, Marek & Cole, Christian & Sherstnev, Alexander & Singh, Vijender & Wrobel, Nicola & Gharbi, Karim & Simpson, Gordon & Owen-

Hughes, Tom & Blaxter, Mark & Barton, Geoffrey. (2015). Evaluation of tools for differential gene expression analysis by RNA-seq on a 48 biological replicate experiment. arXiv.

Shi, Y., Chinnaiyan, A. M., & Jiang, H. (2015). rSeqNP: a non-parametric approach for detecting differential expression and splicing from RNA-Seq data. Bioinformatics, 31(13), 2222-2224.

Wagner, G. P., Kin, K., & Lynch, V. J. (2013). A model based criterion for gene expression calls using RNA-seq data. Theory in Biosciences, 132(3), 159-164.

Walsh, E. O. (1968). An introduction to biochemistry. London, England: English Universities.

Wang, L., Feng, Z., Wang, X., Wang, X., & Zhang, X. (2010). DEGseq: an R package for identifying differentially expressed genes from RNA-seq data. Bioinformatics (Oxford, England), 26(1), 136–138. https://doi.org/10.1093/bioinformatics/btp612

William F. Christensen & Brinley N. Zabriskie (2022) When Your Permutation Test is Doomed to Fail, The American Statistician, 76:1, 53-63, DOI:10.1080/00031305.2021.1902856

Zimmerman, D. W. & Zumbo, B. D. (1990a). Effect of outliers on the relative power of parametric and nonparametric statistical tests. Perceptual and Motor Skills, 71, 339-349.

Zimmerman, D. W. & Zumbo, B.D. (1990b). The relative power of the Wilcoxon-Mann-Whitney test and Student t-test under simple bounded transformations. The Journal of General Psychology, 117(4), 425-436.

Zimmerman, D. W. (1987). Comparative power of Student t test and Mann Whitney U test for unequal sample sizes and variances. Journal of Experimental Education, 55, 171-174.

Zuur, A. F., Ieno, E. N., Walker, N. J., Saveliev, A. A., & Smith, G. M. (2009). Mixed effects models and extensions in ecology with R (Vol. 574). New York: Springer.

APPENDIX A. WHEN SAMPLING RNA-SEQ FROM MONTE CARLO SIMULATION

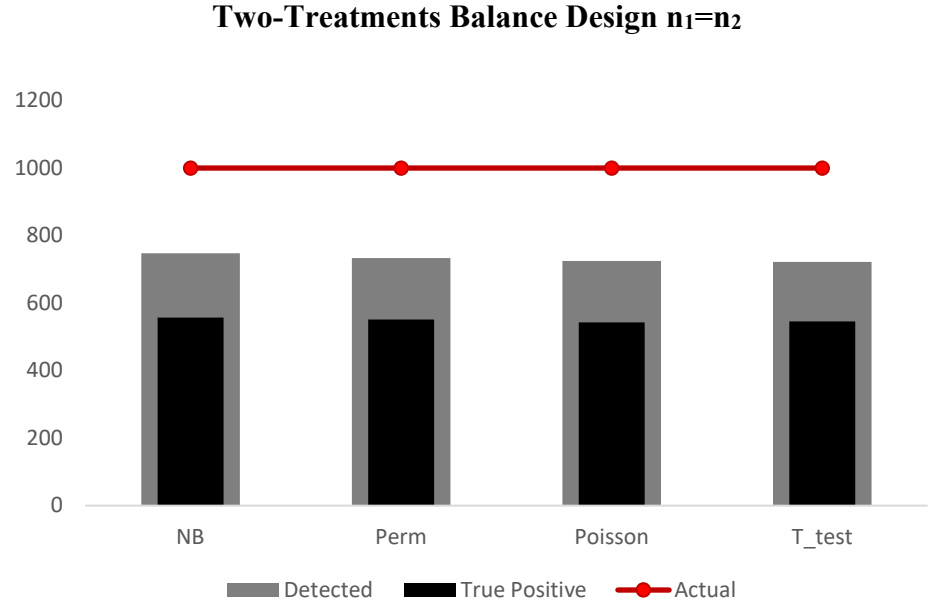

Figure A1. Detection rate when sampling from Normal $n_1=n_2=10$
Each simulation contains 5000 genes and 1000 of them are DE – the red line at 1000 represent the True DE genes; the grey bar represent the total detected genes as DE by a model; the dark bar represent the true detected DE genes; the difference between the grey bar and dark bar represent the false positive.

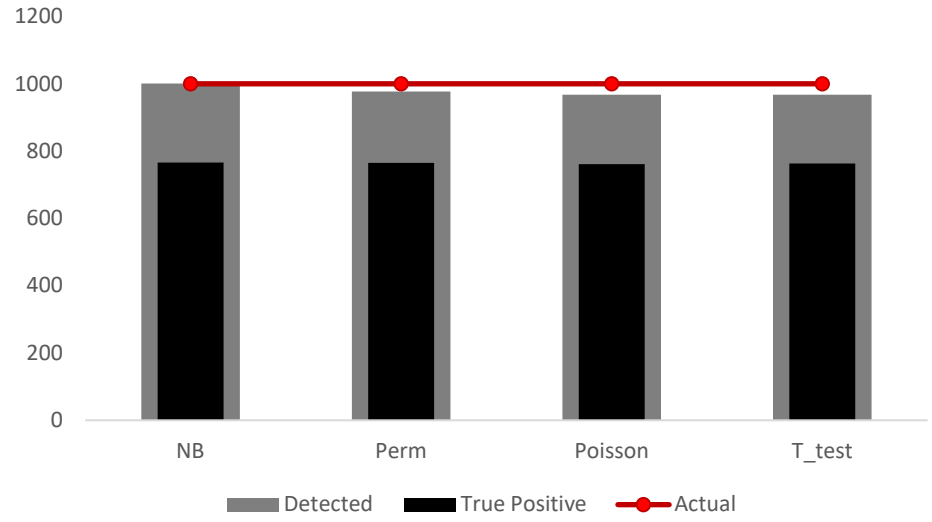

Figure A2. Detection rate when sampling from Normal $n_1=n_2=15$

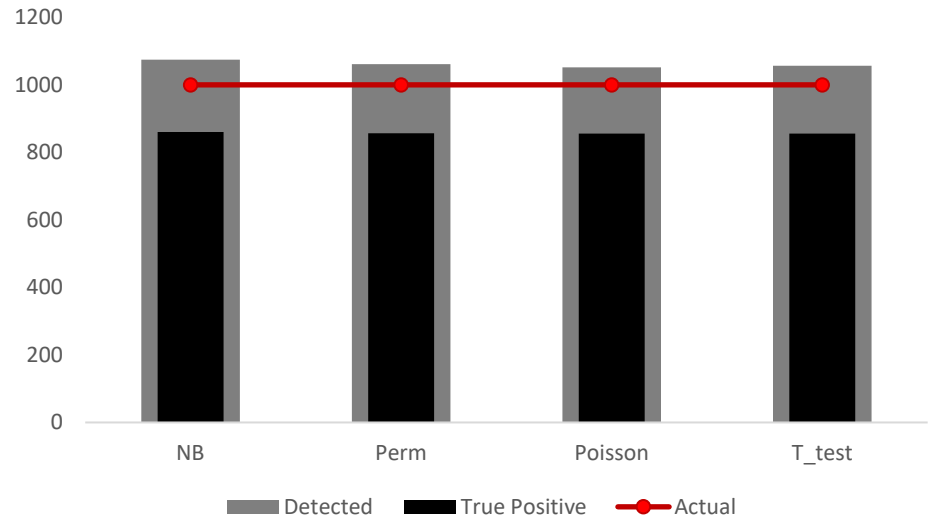

Figure A3. Detection rate when sampling from Normal $n_1=n_2=20$

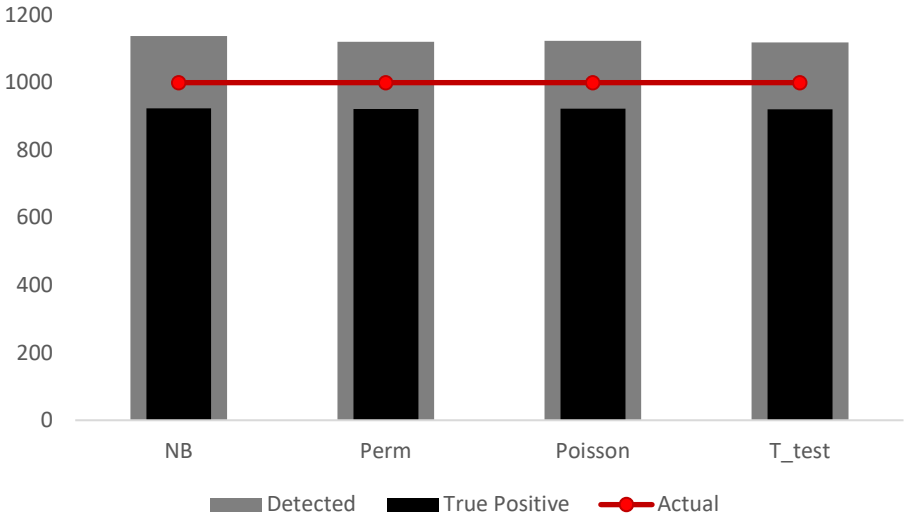

Figure A4. Detection rate when sampling from Normal $n_1=n_2=25$

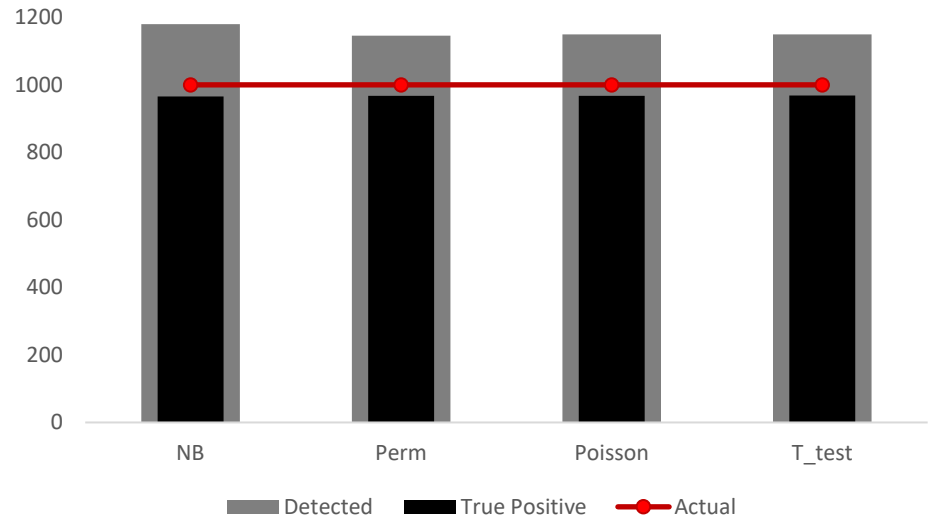

Figure A5. Detection rate when sampling from Normal $n_1=n_2=30$

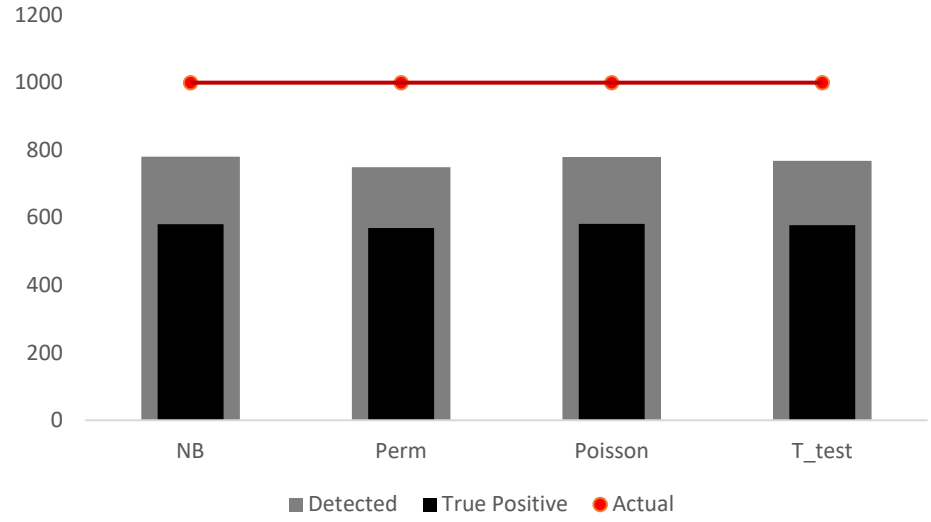

Figure A6. Detection rate when sampling from Poisson $n_1=n_2=10$

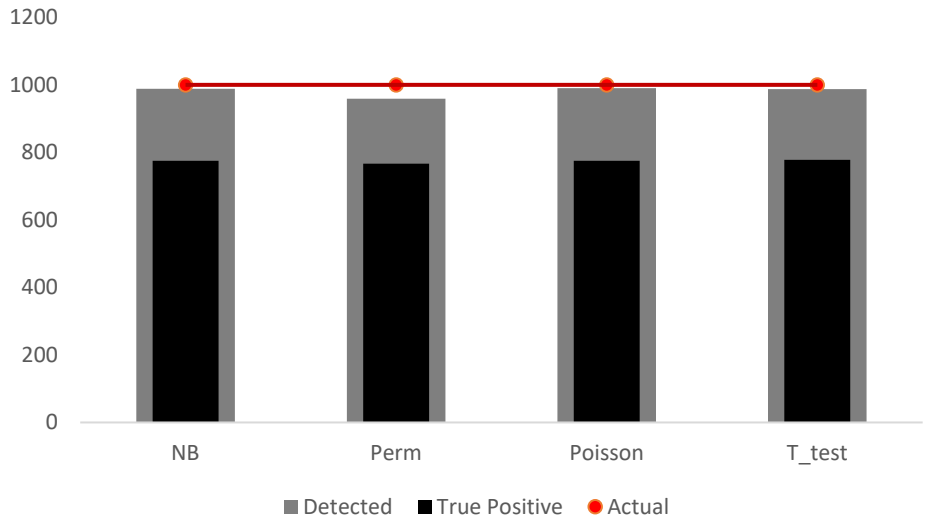

Figure A7. Detection rate when sampling from Poisson $n_1=n_2=15$

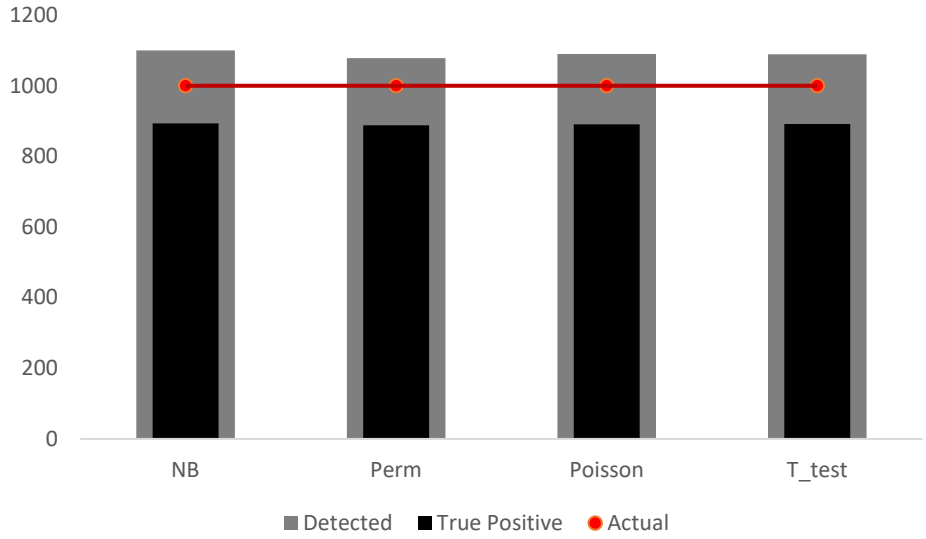

Figure A8. Detection rate when sampling from Poisson $n_1=n_2=20$

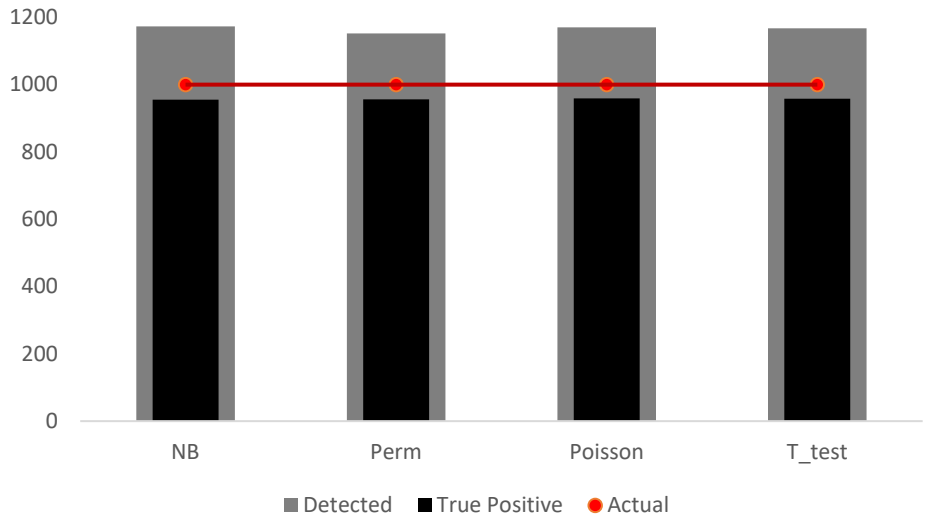

Figure A9. Detection rate when sampling from Poisson $n_1=n_2=25$

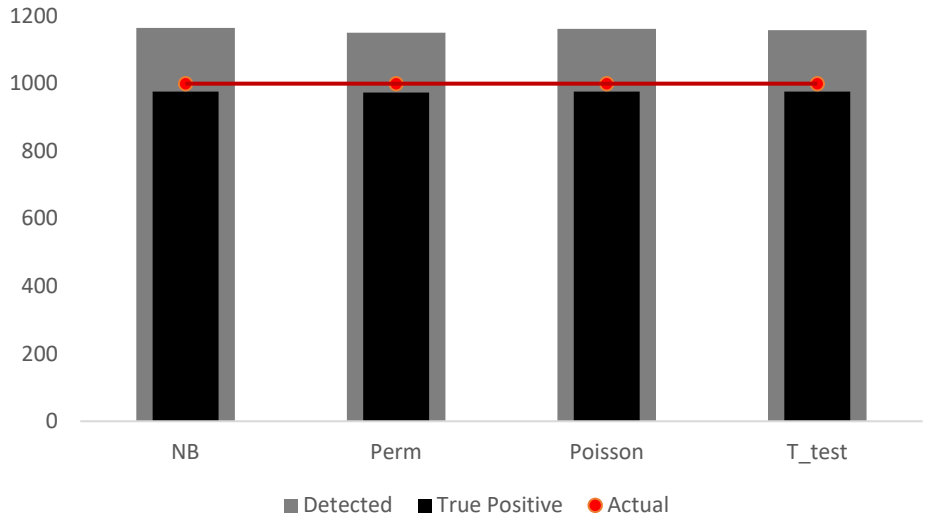

Figure A10. Detection rate when sampling from Poisson $n_1=n_2=30$

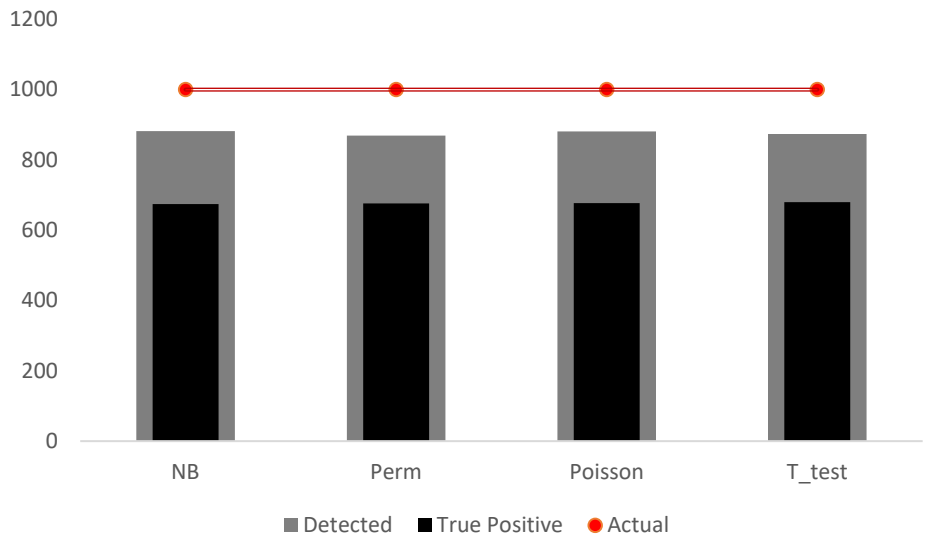

Figure A11. Detection rate when sampling from NB $n_1=n_2=10$

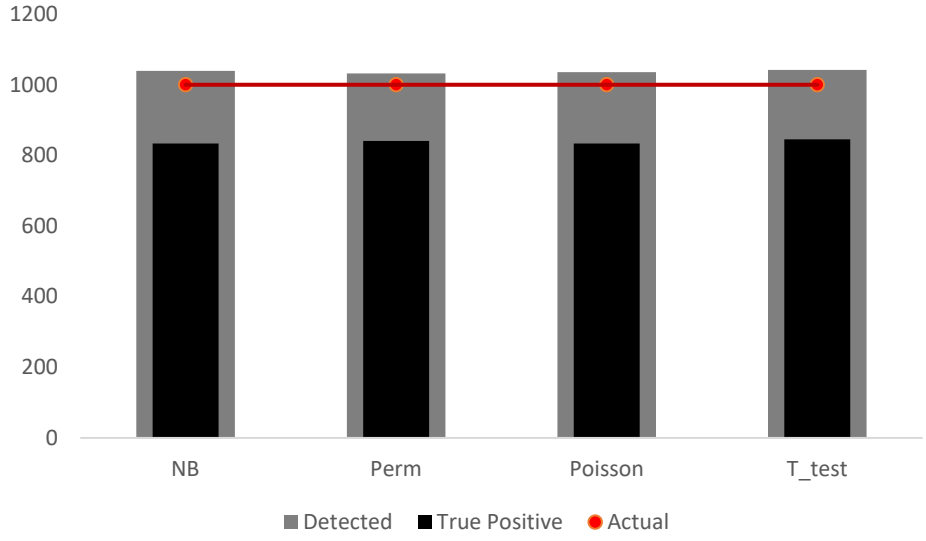

Figure A12. Detection rate when sampling from NB $n_1=n_2=15$

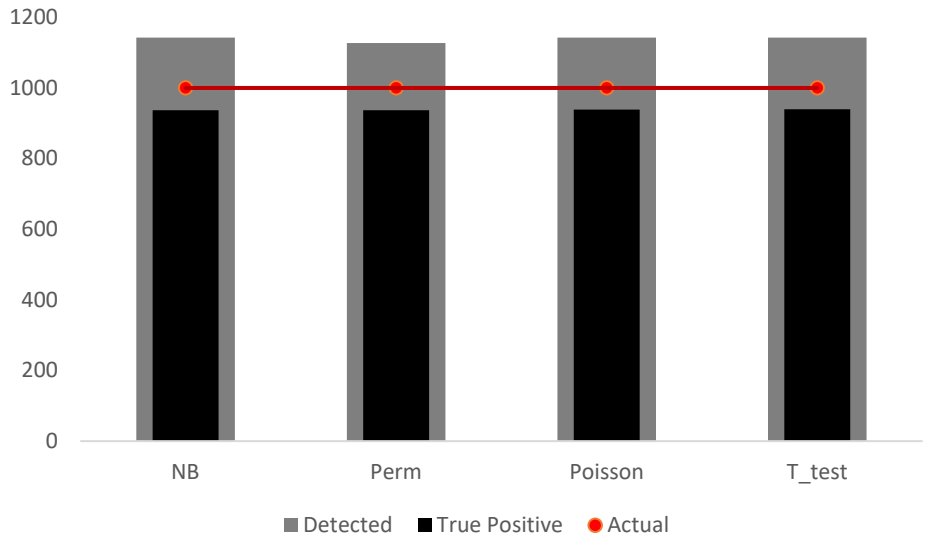

Figure A13. Detection rate when sampling from NB $n_1=n_2=20$

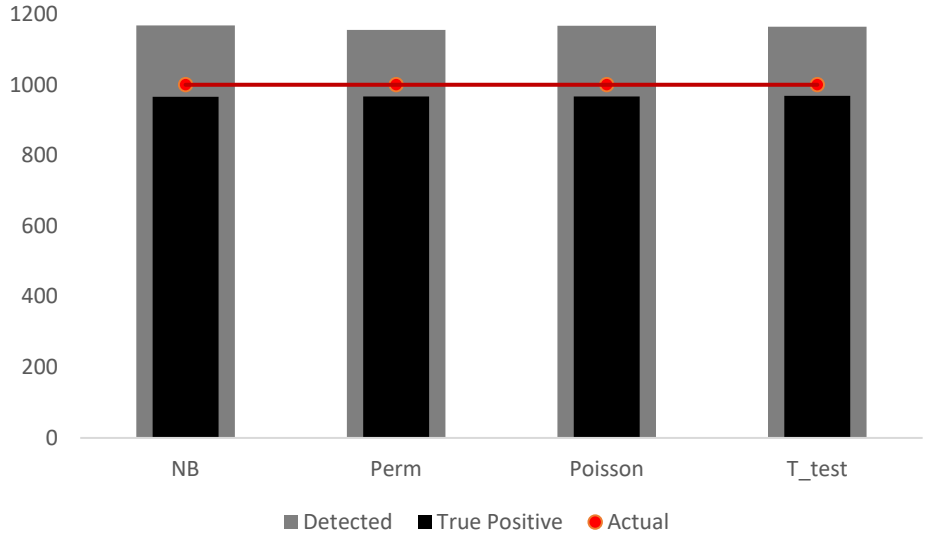

Figure A14. Detection rate when sampling from NB $n_1=n_2=25$

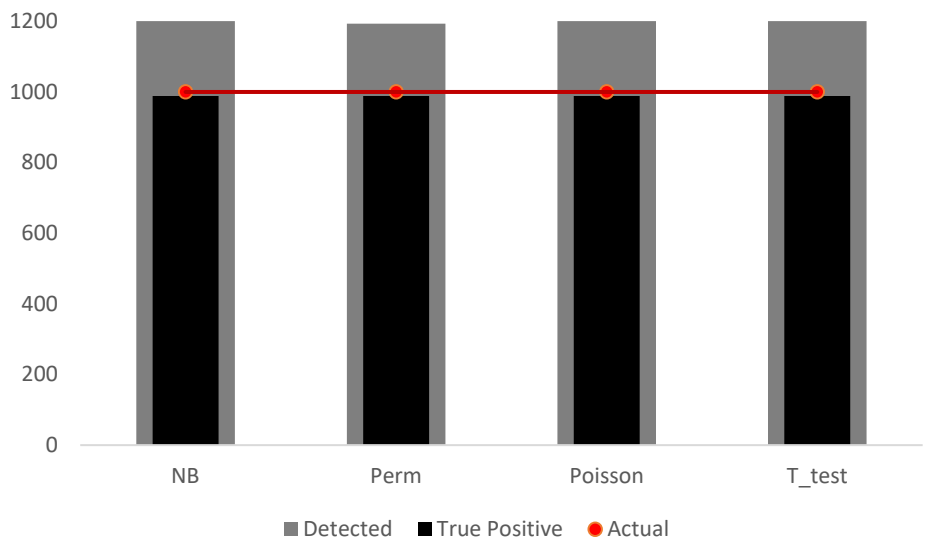

Figure A15. Detection rate when sampling from NB $n_1=n_2=30$

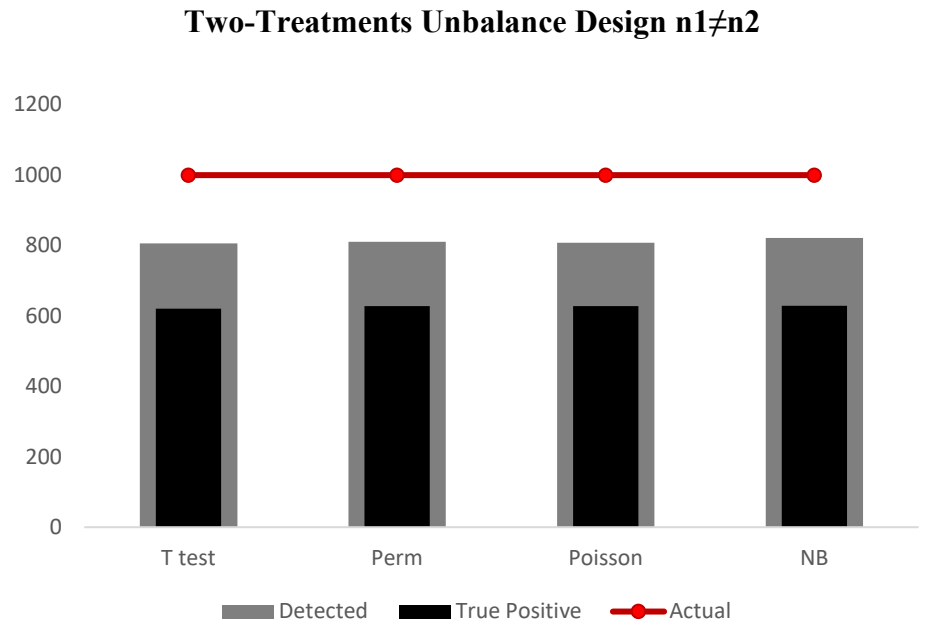

Figure A16. Detection rate when sampling from Normal n1=15 n2=10

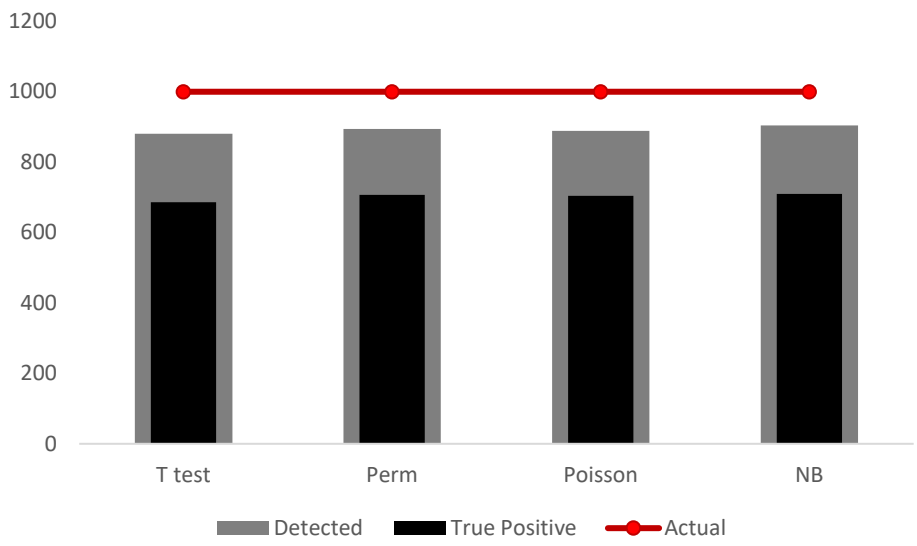

Figure A17. Detection rate when sampling from Normal n1=20 n2=10

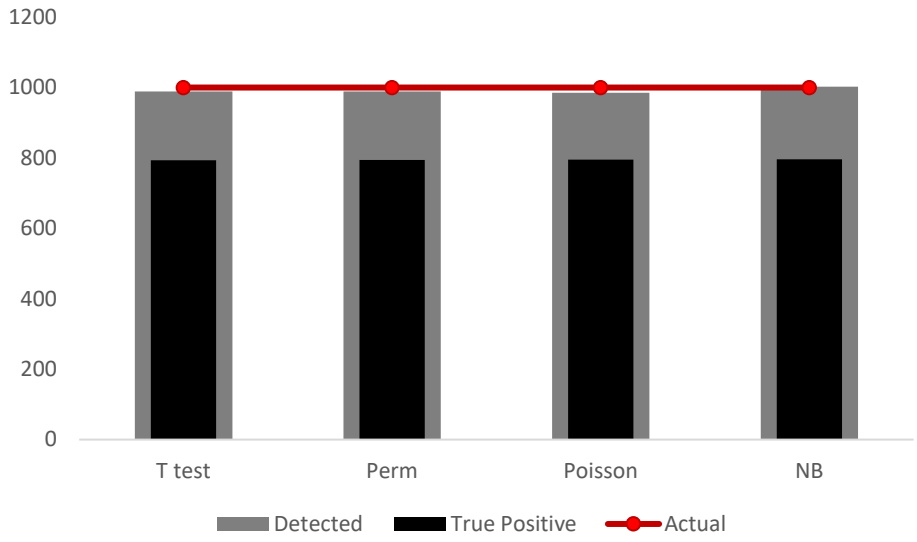

Figure A18. Detection rate when sampling from Normal n_1=20 n_2=15

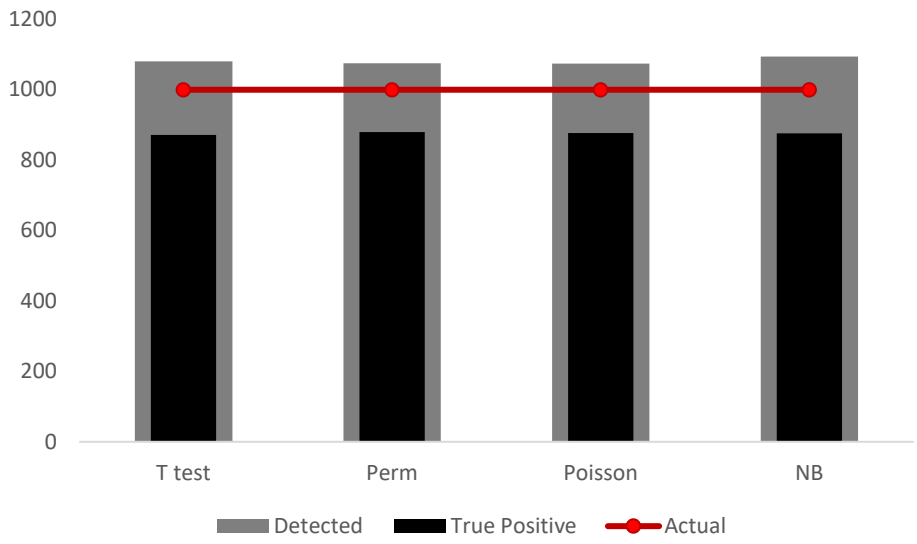

Figure A19. Detection rate when sampling from Normal n_1=30 n_2=15

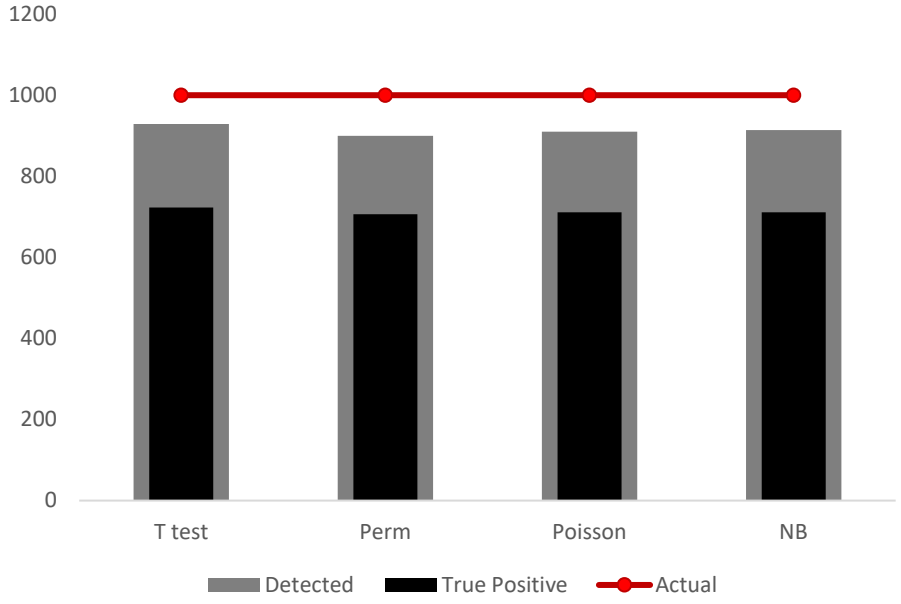

Figure A20. Detection rate when sampling from Poisson n_1=15 n_2=10

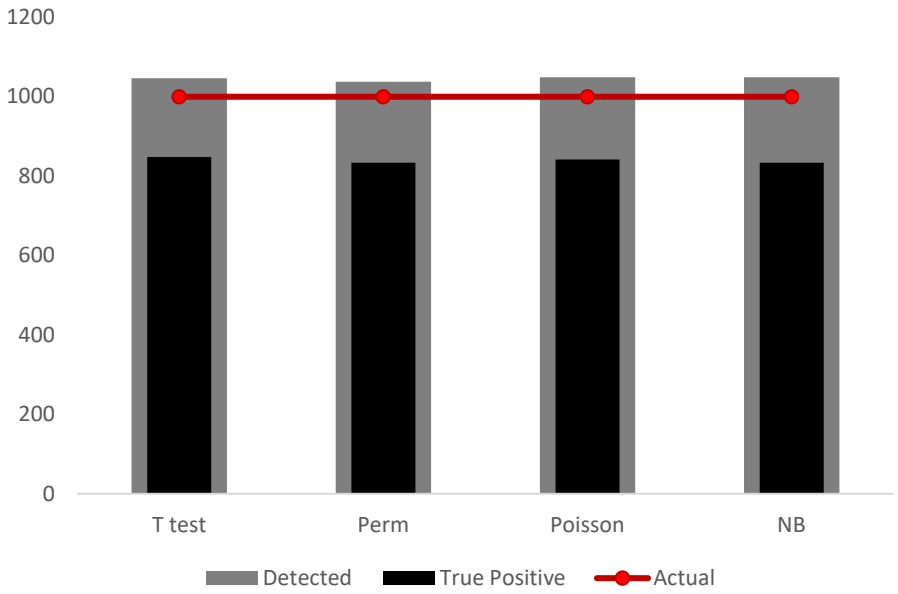

Figure A21. Detection rate when sampling from Poisson n_1=20 n_2=10

Figure A22. Detection rate when sampling from Poisson n_1=30 n_2=15

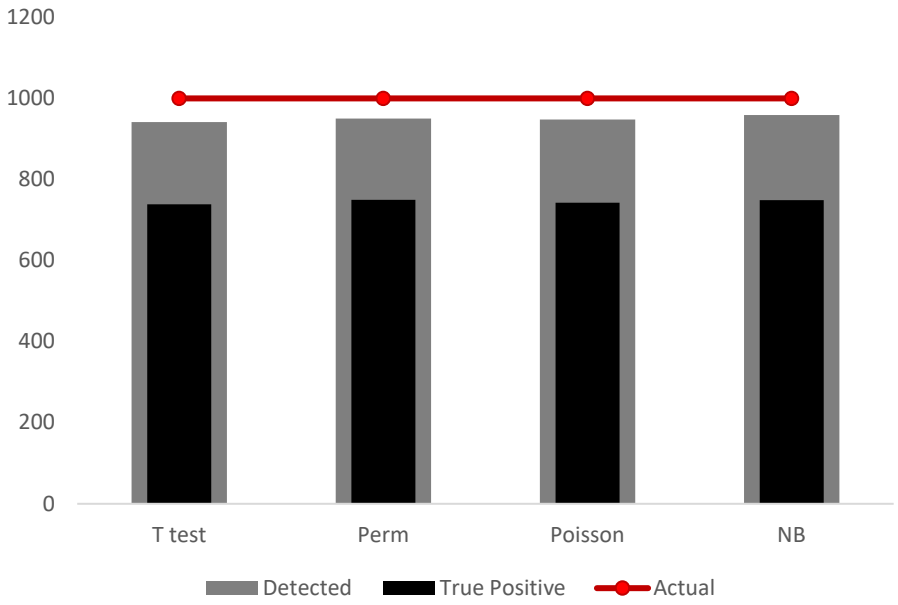

Figure A23. Detection rate when sampling from NB n_1=15 n_2=10

Figure A24. Detection rate when sampling from NB n_1=20 n_2=10

Figure A25. Detection rate when sampling from NB n_1=30 n_2=15

APPENDIX B. WHEN SAMPLING RNA-SEQ FROM SIMSEQ

Two-Treatments Balance DESIGN $n_1=n_2$

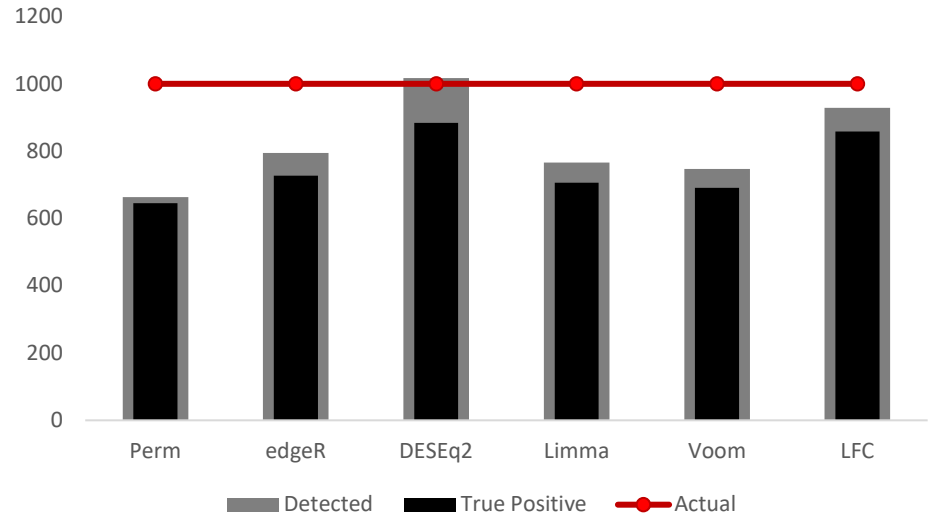

Figure B1. Detection rate when sampling from SimSeq $n_1=n_2=10$

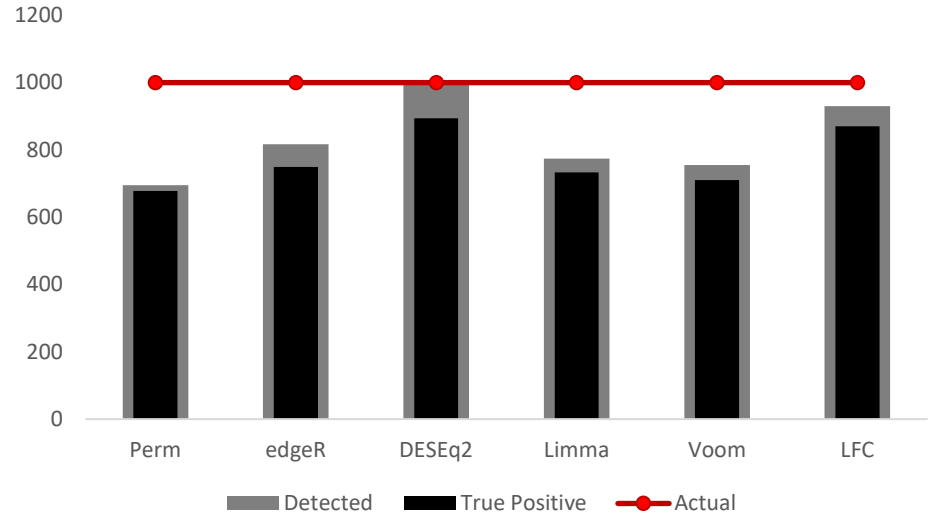

Figure B2. Detection rate when sampling from SimSeq $n_1=n_2=15$

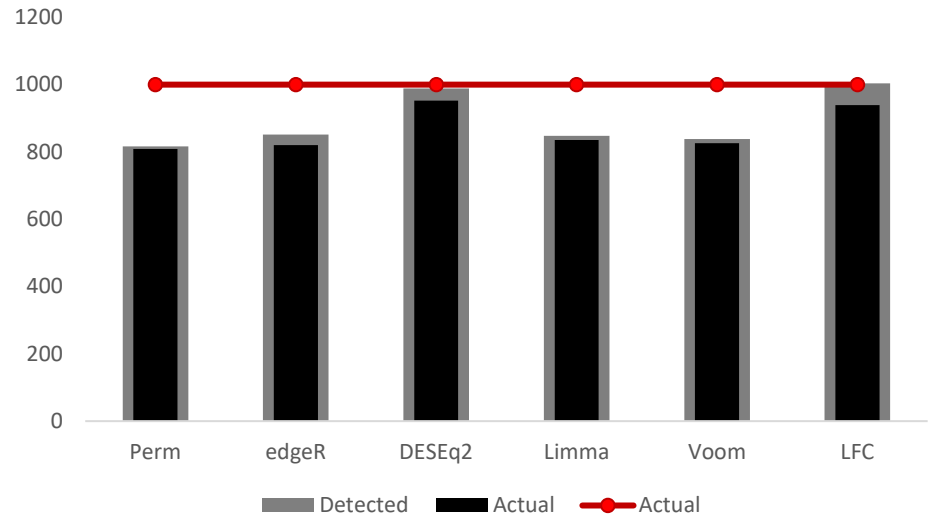

Figure B3. Detection rate when sampling from SimSeq $n_1=n_2=20$

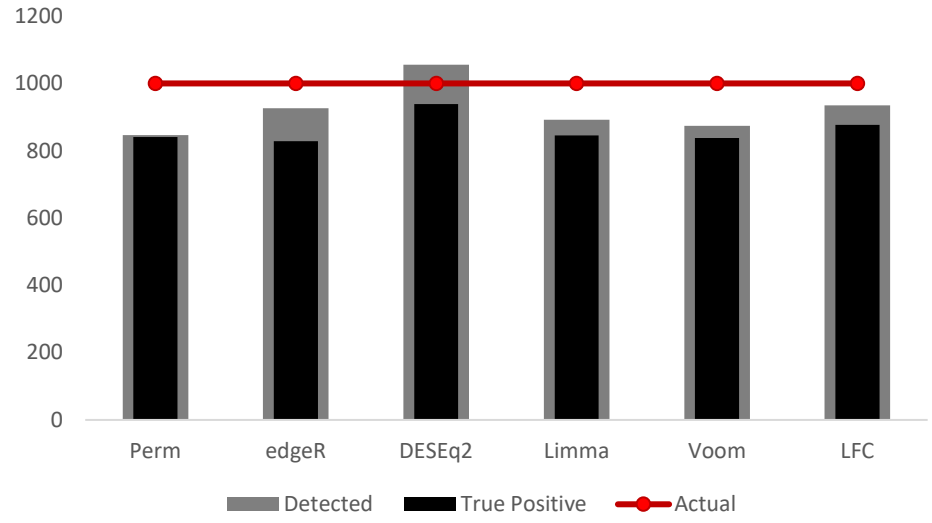

Figure B4. Detection rate when sampling from SimSeq $n_1=n_2=25$

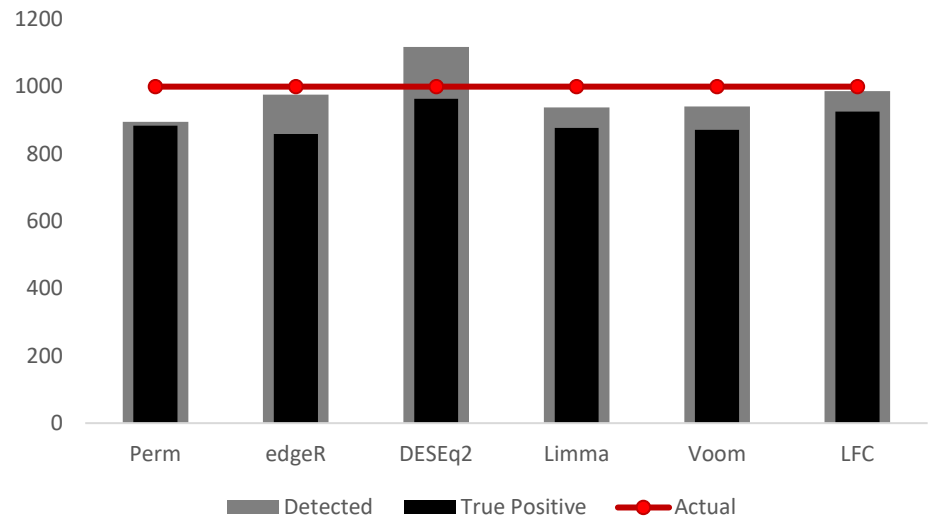

Figure B5. Detection rate when sampling from SimSeq $n_1=n_2=30$

APPENDIX C. SAS CODE

Full Permutation

```
options ls=80 ps=65formchar="|----|+|---+=|-/\<>*";
options mprint symbolgen;

dm 'log;clear;output;clear;';

title1 'Sampling from Normal mu=30, sigma=5, effect size= 1 sigma';

%macro ODSOff();     *** Call prior to BY-group processing ***;
ods graphics off;
ods exclude all;
ods noresults;
%mend;

%macro ODSOn();         *** Call after BY-group processing ***;
ods graphics on;
ods exclude none;
ods results;
%mend;

************************************************************
***** Simgene Macro parameters are:                    *****
*****   NC=number of Control mice                      *****
*****   MuC=Average for Control Gene Population        *****
*****   SD_C=SD for Control Gene Population            *****
*****   NT=number of Treatment mice                    *****
*****   MuT=Average for Treatment Gene Population      *****
*****   SD_T=SD for Treatment Gene Population          *****
************************************************************;
%let NumGene=4000;
%let NumGeneDE=1000;
%let TGenes=5000;

libname norm "C:\Users\bm4817gu\OneDrive for
Business\Desktop\PhD_Project\Power\1_sigma\Normal_m30_s5\DE";

%macro simgene(N_Genes,N_GenesDE,Nmice_C,Mu_C,Sigma_C,NMice_T,Mu_T,Sigma_T);

data generate;
  call streaminit(0);     *** Start Random Number Stream. ***;
  do Gene=1 to &N_Genes;  ***    Loop creating genes.     ***;

   ***** Control Group Loop to generate expression data. ****;
   do Mouse=1 to &Nmice_C;
      Src='C';
        True_DE=0;
      Y=RAND('NORMAL',&Mu_C,&Sigma_C);
      output;
   end;
   ***** Treatment Group Loop to generate expression data. **;
   do Mouse=(&Nmice_C+1) to (&Nmice_C+&Nmice_T);
      Src='T';
        True_DE=0;
```

```
      Y=RAND('NORMAL',&Mu_C,&Sigma_C);
      output;
   end;
   end;
   **** DE Genes ******;
   do Gene=&N_Genes+1 to &N_Genes+&N_GenesDE;  ***    Loop creating genes.
***;
   ***** Control Group Loop to generate expression data. ****;
   do Mouse=1 to &Nmice_C;
      Src='C';
        True_DE=1;
      Y=RAND('NORMAL',&Mu_C,&Sigma_C);
      output;
   end;
   ***** Treatment Group Loop to generate expression data. **;
   do Mouse=(&Nmice_C+1) to (&Nmice_C+&Nmice_T);
      Src='T';
        True_DE=1;
      Y=RAND('NORMAL',&Mu_T,&Sigma_T);
      output;
   end;

  end; ***************** End the Gene Loop. *****************;
  run;

*ods graphics off;
proc univariate data=generate noprint;
  class Src;
  var Y;
  histogram Y / normal(color=black);
  title2 "Comparison of Control and Treatment Populations";
  title3 "NC:&Nmice_C, MuC:&Mu_C, SD_C:&Sigma_C";
  title4 "NT:&Nmice_T, MuT:&Mu_T, SD_T:&Sigma_T";
  run;

proc sort data=generate;
  by Gene;
  run;
proc transpose data=generate
     out=Src_MV(where=(_Name_='Y')
         rename=(Col1=C1 Col2=C2 Col3=C3 Col4=C4 Col5=C5 Col6=C6 Col7=C7
Col8=C8 Col9=C9 Col10=C10
                 Col11=T1 Col12=T2 Col13=T3 Col14=T4 Col15=T5 Col16=T6
Col17=T7 Col18=T8 Col19=T9 Col20=T10));
  by Gene;
  run;
%mend simgene;

***** N_Genes,Nmice_C,Mu_C,Sigma_C,NMice_T,Mu_T,Sigma_T *****;
%simgene(&NumGene,&NumGeneDE,10,30,5,10,25,5);

  ods graphics on; ods exclude none; ods results;
proc transpose data=generate out=raw_data (where=(_name_='Y'));
  by Gene True_DE;
run;

Data norm.raw_data10;
```

```
      set raw_data;
      keep Gene True_DE;
run;

data generate;
      set generate;
      if Y=0 then D=0; else D=1;
run;

data Src_MV;
  set Src_MV;
    Chk_C=C1+C2+C3+C4+C5+C6+C7+C8+C9+C10; *** Create flag for all zeroes for
Cont. *;
    Chk_T=T1+T2+T3+T4+T5+T6+T7+T8+T9+T10; *** Create flag for all zeroes for
Test. *;

    *** Remove obs without any signal. ***;
    if (Chk_T=0) and (Chk_C=0) then delete;


C_bits=compress((C1>0)||(C2>0)||(C3>0)||(C4>0)||(C5>0)||(C6>0)||(C7>0)||(C8>0
)||(C9>0)||(C10>0));

T_bits=compress((T1>0)||(T2>0)||(T3>0)||(T4>0)||(T5>0)||(T6>0)||(T7>0)||(T8>0
)||(T9>0)||(T10>0));


C_sum=(C1>0)+(C2>0)+(C3>0)+(C4>0)+(C5>0)+(C6>0)+(C7>0)+(C8>0)+(C9>0)+(C10>0);

T_sum=(T1>0)+(T2>0)+(T3>0)+(T4>0)+(T5>0)+(T6>0)+(T7>0)+(T8>0)+(T9>0)+(T10>0);

    if  (T_bits='1111111111' and C_bits='1111111111'); *** Select subset ***;
  run;
proc sort;
  by Gene;
run;

proc iml;
  start permtst;
    alldata = Con || Trt;     /* stack data in a single vector */
    N1 = ncol(Con);
    N = N1 + ncol(Trt);

    /** First row of XControl is Observed --- gives obsdiff */
    XControl = allcomb(N, N1);   /* create all combinations */
    NRepl=nrow(XControl);   /* Get number of Combs for Loop */
    XCRef=1:N;                   /* Create reference vector */
  * print Con Trt;
  * print N1 N XCRef;

    nulldist = j(NRepl,1);    /* allocate vector for results */
    do k = 1 to NRepl;
       C_ids=Xcontrol[k,];  /* Select kth row of comb matrix */
       T_ids=setdif(XCRef,C_ids);     /* Select kth anti-row */
     * print C_ids T_ids;
       Control=alldata[,C_ids];   /* Select control elements */
       Treat=alldata[,T_ids];   /* Select treatment elements */
```

```
     * print Control Treat;
       /* difference of means */
       nulldist[k] = mean(T(Control)) - mean(T(Treat));
    end;

    * print nulldist;
    title2 "Histogram of Null Distribution";
    pval = (sum(abs(nulldist) >= abs(obsdiff))) / (NRepl);
    * print pval;
  finish;

*** Main starts here ************************************
*************************************************************;
  use Src_MV;
  nullp = j(&TGenes,1);    /* allocate vector for results */

  *** Each Gene: Con=Control, Trt=Treatment ***;
  do i=1 to &TGenes;
     read point i var{C1 C2 C3 C4 C5 C6 C7 C8 C9 C10} into Con;
     read point i var{T1 T2 T3 T4 T5 T6 T7 T8 T9 T10} into Trt;

     obsdiff = mean(T(Trt)) - mean(T(Con));
   * print obsdiff;

     run permtst;

     nullp[i]=pval;

  end;

* print nullp;
  create p_results(rename=(col1=pvalue)) from nullp;
  append from nullp;

  quit;

data perm_p;
  set p_results;
  file 'perm_p.txt';
  put pvalue;
  run;

%ODSOff;
ods output TTests=sample_ts;
proc ttest data=generate;
  by Gene;
  class Src;
  var Y;
  run;
%ODSOn;

proc print data=sample_ts(obs=100);
  run;

data ts (keep=Gene tValue Probt);
  set sample_ts;
  if method='Satterthwaite'; *** Assume unequal variances. **;
```

```
  file 't_pvals.txt';
  put Gene '09'x Probt;
  format Probt 10.8;
  run;


data t_p;
  infile 't_pvals.txt' missover dlm='09'x dsd;
  input Gene T_pval;
  run;

data perm_p;
  infile 'perm_p.txt';
  input P_pval;
  run;

data tvsp;
  merge t_p perm_p;  *** 1-1 merge on presorted data ***;
  diff_p=T_pval-P_pval;
  run;

ods graphics on;
proc univariate;
  var T_pval P_pval diff_p;
  histogram T_pval P_pval diff_p;
  run;

data ttest_p (keep=Gene ProbF);
      set tvsp;
      rename T_pval=ProbF;
run;

data permut_p (keep=Gene ProbF);
      set tvsp;
      rename P_pval=ProbF;
run;

************************************************************
******* Generalized Linear Model Approach - Poisson Y *******
************************************************************;
%ODSOFF
ods output FitStatistics=PFitStats;
ods output ParameterEstimates=PParmEsts;
ods output Tests3=PFixed3;
ods output LSMeans=PLSMns;
ods output Diffs=PDiffsMns;

proc glimmix data=generate;
  by Gene;
  class Src;
  model Y=Src / d=poisson solution ddfm=kr;
  random _residual_;
  lsmeans Src / diff ilink;
  title2 'Generalized Linear Model Approach';
  run;
%ODSON
```

```
************************************************************
******* Generalized Linear Model Approach - NB Y *******
************************************************************;

%ODSOff
ods output FitStatistics=NBFitStats;
ods output ParameterEstimates=NBParmEsts;
ods output Tests3=NBFixed3;
ods output LSMeans=NBLSMns;
ods output Diffs=NBDiffsMns;

proc glimmix data=generate method=rmpl initglm;
  by Gene;
  class Src;
  model Y=Src / d=NB solution ddfm=kr;
  nloptions tech=nrridg;
  random _residual_;
  lsmeans Src / diff ilink;
  title2 'Generalized Linear Model Approach - Negative Binomial';
  run;
%ODSON

data norm.AllFixed10 (keep=Gene ProbF Approach);
  set PFixed3  (in=in1)
      NBFixed3 (in=in2)
      ttest_p  (in=in3)
      permut_p (in=in4);
  if in1 then Approach='Poisson';
  else if in2 then Approach='NB';
  else if in3 then Approach='T_test';
  else if in4 then Approach='Perm';
  run;
```

Partial Permutation

```
options ls=80 ps=65 formchar="|----|+|---+=|-/\<>*";
options mprint symbolgen;
dm 'log;clear;output;clear;';
%let t0 = %sysfunc(datetime()); *** Collect timing data. ***;
*ods rtf file='Normal_10k_genes_B5k_c30t30.rtf';
title1 'Sampling from Normal mu=30, sigma=5, effect size= 1 sigma';

%macro ODSOff();     *** Call prior to BY-group processing ***;
ods graphics off;
ods exclude all;
ods noresults;
%mend;

%macro ODSOn();         *** Call after BY-group processing ***;
ods graphics on;
ods exclude none;
ods results;
%mend;

*************************************************************;
***** Simulation Macro parameters are:                  *****;
*****                                                   *****;
%let N_Genes=4000;      *** Number of genes to simulate. ***;
%let N_GenesDE=1000; *** Number of DE genes to simulate. ***;
%let Nmice_C=30;        *** Number of Control mice.    *****;
%let Mu_C=30;     *** Mean of Control Gene Population. *****;
%let Sigma_C=5;     *** SD of Control Gene Population. *****;
%let NMice_T=30;        *** Number of Treatment mice.  *****;
%let Mu_T=25;     *** Mean of Treatment Gene Population. ***;
%let Sigma_T=5;     *** SD of Treatment Gene Population. ***;
%let N_Perms=5000;  *** # random permutations for permtest.*;
*************************************************************;
libname norm "C:\Users\bm4817gu\OneDrive for
Business\Desktop\PhD_Project\Power\1_sigma\Normal_m30_s5\DE";
data Generate;
  call streaminit(0);     *** Start Random Number Stream. ***;
  do Gene=1 to &N_Genes;  ***    Loop creating genes.     ***;

   ***** Control Group Loop to generate expression data. ****;
   do Mouse=1 to &Nmice_C;
      Src='C';
        True_DE=0;
      Y=RAND('NORMAL',&Mu_C,&Sigma_C);
      output;
   end;
   ***** Treatment Group Loop to generate expression data. **;
   do Mouse=(&Nmice_C+1) to (&Nmice_C+&Nmice_T);
      Src='T';
        True_DE=0;
      Y=RAND('NORMAL',&Mu_C,&Sigma_C);
      output;
   end;
   end;
   **** DE Genes ******;
```

```
   do Gene=&N_Genes+1 to &N_Genes+&N_GenesDE;  ***    Loop creating genes.
***;
   ***** Control Group Loop to generate expression data. ****;
   do Mouse=1 to &Nmice_C;
      Src='C';
        True_DE=1;
      Y=RAND('NORMAL',&Mu_C,&Sigma_C);
      output;
   end;
   ***** Treatment Group Loop to generate expression data. **;
   do Mouse=(&Nmice_C+1) to (&Nmice_C+&Nmice_T);
      Src='T';
        True_DE=1;
      Y=RAND('NORMAL',&Mu_T,&Sigma_T);
      output;
   end;
  end; ***************** End the Gene Loop. *****************;
  run;

ods graphics on; ods exclude none; ods results;
proc transpose data=generate out=raw_data (where=(_name_='Y'));
  by Gene True_DE;
run;

Data norm.raw_data30;
      set raw_data;
      keep Gene True_DE;
run;

ods graphics off;
proc univariate data=Generate;
  class Src;
  var Y;
  histogram Y / normal(color=black);
  title2 "NC:&Nmice_C, MuC:&Mu_C, SD_C:&Sigma_C";
  title3 "NT:&Nmice_T, MuT:&Mu_T, SD_T:&Sigma_T";
  run;

ods graphics off; ods exclude all; ods noresults;
***** Get permutation means for each group by gene. *****;
proc means data=Generate;
  by Gene;
  class Src;
  ways 1;
  output out=Gene_Mns_UV mean=Observed_mn_Y;
  var Y;
run;

ods graphics on; ods exclude none; ods results;
proc transpose data=Gene_Mns_UV out=Gene_Mns_MV
(where=(_name_='Observed_mn_Y'));
  by Gene;
run;

data Gene_Mns_MV (drop=_Name_);
  set Gene_Mns_MV;
  Observed_Diff_Mns = Col1 - Col2;
```

```
  rename Col1=Observed_mn_C Col2=Observed_mn_T;
  run;


***** Select &N_Perms Random Permutations from each Gene. ***;
proc surveyselect data=Generate
  method = SRS
  reps = &N_Perms
  seed = 0
  N = &Nmice_C
  out = RandomPerms
  outall;
  strata Gene;
run;

ods graphics off; ods exclude all; ods noresults;
***** Get permutation means for each group by gene. *****;
proc means data=RandomPerms;
  by Gene Replicate;
  class Selected;
  ways 1;
  output out=Gene_Perm_Mns_UV mean=mn_Y;
  var Y;
  run;
ods graphics on; ods exclude none; ods results;

proc transpose data=Gene_Perm_Mns_UV
                 out=Gene_Perm_Mns_MV (where=(_name_='mn_Y')
                                       rename=(Col1=mn_C Col2=mn_T));
  by Gene Replicate;
  run;

data p_results;
  merge Gene_Perm_Mns_MV  Gene_Mns_MV;
  by Gene;
  Diff_Mns = mn_C - mn_T;
  if first.Gene then Reject_Count=0;
  if abs(Diff_Mns) >= abs(Observed_Diff_Mns)
    then Reject_Count+1;
  if last.Gene then do;
    pvalue=Reject_Count/(&N_Perms+1);
    output;
    end;
  run;

ods graphics off; ods exclude all; ods noresults;
ods output TTests=sample_ts;
proc ttest data=generate;
  by Gene;
  class Src;
  var Y;
  run;
ods graphics on; ods exclude none; ods results;

data ts (keep=Gene tValue Probt);
  set sample_ts;
  if method='Satterthwaite'; *** Assume unequal variances. **;
```

```
  run;

data tvsp;
  merge ts p_results;  *** 1-1 merge on presorted data ***;
  rename Probt=T_pval
         pvalue=P_pval;
  diff_p=Probt-pvalue;
  run;

ods graphics on;
proc univariate;
  var diff_p;
  histogram diff_p;
  run;

data ttest_p (keep=Gene ProbF);
      set tvsp;
      rename T_pval=ProbF;
run;

data permut_p (keep=Gene ProbF);
      set tvsp;
      rename P_pval=ProbF;
run;

************************************************************
******* Generalized Linear Model Approach - Poisson Y *******
************************************************************;
%ODSOFF
ods output FitStatistics=PFitStats;
ods output ParameterEstimates=PParmEsts;
ods output Tests3=PFixed3;
ods output LSMeans=PLSMns;
ods output Diffs=PDiffsMns;

proc glimmix data=generate;
  by Gene;
  class Src;
  model Y=Src / d=poisson solution ddfm=kr;
  random _residual_;
  lsmeans Src / diff ilink;
  title2 'Generalized Linear Model Approach';
  run;
%ODSON

************************************************************
******* Generalized Linear Model Approach - NB Y *******
************************************************************;

%ODSOff
ods output FitStatistics=NBFitStats;
ods output ParameterEstimates=NBParmEsts;
ods output Tests3=NBFixed3;
ods output LSMeans=NBLSMns;
ods output Diffs=NBDiffsMns;

proc glimmix data=generate method=rmpl initglm;
```

```
  by Gene;
  class Src;
  model Y=Src / d=NB solution ddfm=kr;
  nloptions tech=nrridg;
  random _residual_;
  lsmeans Src / diff ilink;
  title2 'Generalized Linear Model Approach - Negative Binomial';
  run;
%ODSON

data norm.AllFixed30 (keep=Gene ProbF Approach);
  set PFixed3  (in=in1)
      NBFixed3 (in=in2)
      ttest_p  (in=in3)
      permut_p (in=in4);
  if in1 then Approach='Poisson';
  else if in2 then Approach='NB';
  else if in3 then Approach='T_test';
  else if in4 then Approach='Perm';
run;

*ods rtf close;
```

Summarizing Simulation Results

```
libname norm "C:\Users\bm4817gu\OneDrive for
Business\Desktop\PhD_Project\Power\1_sigma\Normal_m30_s5\DE";
ods pdf file="Normal_mu30_sig5_effect_1sig.pdf";
proc format;
  value sig  0-.05 = 'Reject'
           .05<-1 = 'DNR';
  value gen       1='DE'
                  0='EE';
  run;
************************************************************
************************************************************
*******                    n1=n2=10                    *******
************************************************************
************************************************************;

************************************************************
************************************************************
******* Get Rejection Rates (Row 1 Table 11.1 Stroup) *******
************************************************************
************************************************************;
title1 'Sampling from Normal mu=30, sigma=5, effect size= 1 sigma';
proc freq data=norm.AllFixed10;
  tables Approach*ProbF / nopct nocol;
  format ProbF sig.;
  title2 'Compare Rejection Rates Across Approaches when n1=n2=10 when
n1=n2=10';
run;

************************************************************
************************ DE genes ************************
************************************************************;
data DEgenes10;
      set norm.AllFixed10;
      where ProbF<=0.05;
      if Approach="Poisson" then Pois_DE=1; else Pois_DE=0;
      if Approach="NB" then NB_DE=1; else NB_DE=0;
      if Approach="T_test" then Ttest_DE=1; else Ttest_DE=0;
      if Approach="Perm" then Perm_DE=1; else Perm_DE=0;
run;

proc sort data=DEgenes10;
by Gene;
run;

data perm10(keep=Gene Perm_DE) tt10(keep=Gene Ttest_DE) pois10(keep=Gene
Pois_DE) bin10(keep=Gene NB_DE);
      set DEgenes10;
      if Approach='Poisson' then output pois10;
      else if Approach='NB' then output bin10;
      else if Approach='Perm' then output perm10;
      else if Approach='T_test' then output tt10;
run;

data comb_de10;
```

```
      *drop ProbF Approach;
      merge norm.raw_data10 perm10 tt10 pois10 bin10;
      by Gene;
run;

data comb_de10;
   set comb_de10;
   array NumVar _numeric_;
   do over NumVar;
      if NumVar=. then NumVar=0;
   end;
run;

ods output SenSpec=senspec10;
proc freq data=comb_de10 order=formatted;
  tables Perm_DE*True_DE Pois_DE*True_DE NB_DE*True_DE Ttest_DE*True_DE/
senspec;
  format True_DE Perm_DE Pois_DE NB_DE Ttest_DE gen.;
  title2 'True Rejection, Model Accuracy when n1=n2=10';
run;

data stat10;
  set senspec10;
  if Table="Table Perm_DE * True_DE" then table="Permutation";
  else if Table="Table Pois_DE * True_DE" then table="Poisson";
  else if Table="Table NB_DE * True_DE" then table="Neg Binomial";
  else if Table="Table Ttest_DE * True_DE" then table="T test";
  if Statistic="Specificity" then FP=1-Estimate;
run;

proc sort data=stat10;
  by Table;
run;

data sens10 (rename=(Estimate=Sensitivity)) spec10
(rename=(Estimate=Specificity));
  set stat10;
  if Statistic="Sensitivity" then output sens10;
  if Statistic="Specificity" then output spec10;
run;

data stats10;
  merge sens10 spec10;
  by Table;
  keep Table Sensitivity Specificity FP;
run;

proc print data=stats10;
  title2 'Model Sensitivity, Specificity and False Positive Rate when
n1=n2=10';
run;

proc sgplot data=stats10;
  vbar Table/response=Sensitivity;
  title2 'Comparing Model Sensitivity when n1=n2=10';
run;
```

```
proc sgplot data=stats10;
  vbar Table/response=Specificity;
  title2 'Comparing Model Specificity when n1=n2=10';
run;

proc sgplot data=stats10;
  vbar Table/response=FP;
  title2 'Comparing Model False Positive Rate when n1=n2=10';
run;
************************************************************
************************************************************
*******                    n1=n2=15                    *******
************************************************************
************************************************************;

************************************************************
************************************************************
******* Get Rejection Rates (Row 1 Table 11.1 Stroup) *******
************************************************************
************************************************************;

proc freq data=norm.AllFixed15;
  tables Approach*ProbF / nopct nocol;
  format ProbF sig.;
  title2 'Compare Rejection Rates Across Approaches when n1=n2=15';
run;

************************************************************
************************ DE genes **************************
************************************************************;
data DEgenes15;
      set norm.AllFixed15;
      where ProbF<=0.05;
      if Approach="Poisson" then Pois_DE=1; else Pois_DE=0;
      if Approach="NB" then NB_DE=1; else NB_DE=0;
      if Approach="T_test" then Ttest_DE=1; else Ttest_DE=0;
      if Approach="Perm" then Perm_DE=1; else Perm_DE=0;
run;

proc sort data=DEgenes15;
by Gene;
run;

data perm15(keep=Gene Perm_DE) tt15(keep=Gene Ttest_DE) pois15(keep=Gene
Pois_DE) bin15(keep=Gene NB_DE);
      set DEgenes15;
      if Approach='Poisson' then output pois15;
      else if Approach='NB' then output bin15;
      else if Approach='Perm' then output perm15;
      else if Approach='T_test' then output tt15;
run;

data comb_de15;
      *drop ProbF Approach;
      merge norm.raw_data15 perm15 tt15 pois15 bin15;
      by Gene;
run;
```

```
data comb_de15;
   set comb_de15;
   array NumVar _numeric_;
   do over NumVar;
      if NumVar=. then NumVar=0;
   end;
run;

ods output SenSpec=senspec15;
proc freq data=comb_de15 order=formatted;
  tables Perm_DE*True_DE Pois_DE*True_DE NB_DE*True_DE Ttest_DE*True_DE/
senspec;
  format True_DE Perm_DE Pois_DE NB_DE Ttest_DE gen.;
  title2 'True Rejection, Model Accuracy when n1=n2=15';
run;

data stat15;
  set senspec15;
  if Table="Table Perm_DE * True_DE" then table="Permutation";
  else if Table="Table Pois_DE * True_DE" then table="Poisson";
  else if Table="Table NB_DE * True_DE" then table="Neg Binomial";
  else if Table="Table Ttest_DE * True_DE" then table="T test";
  if Statistic="Specificity" then FP=1-Estimate;
run;

proc sort data=stat15;
  by Table;
run;

data sens15 (rename=(Estimate=Sensitivity)) spec15
(rename=(Estimate=Specificity));
  set stat15;
  if Statistic="Sensitivity" then output sens15;
  if Statistic="Specificity" then output spec15;
run;

data stats15;
  merge sens15 spec15;
  by Table;
  keep Table Sensitivity Specificity FP;
run;

proc print data=stats15;
  title2 'Model Sensitivity, Specificity and False Positive Rate when
n1=n2=15';
run;

proc sgplot data=stats15;
  vbar Table/response=Sensitivity;
  title2 'Comparing Model Sensitivity when n1=n2=15';
run;

proc sgplot data=stats15;
  vbar Table/response=Specificity;
  title2 'Comparing Model Specificity when n1=n2=15';
run;
```

```
proc sgplot data=stats15;
  vbar Table/response=FP;
  title2 'Comparing Model False Positive Rate when n1=n2=15';
run;
************************************************************
************************************************************
*******                    n1=n2=20                     *******
************************************************************
************************************************************;

************************************************************
************************************************************
******* Get Rejection Rates (Row 1 Table 11.1 Stroup) *******
************************************************************
************************************************************;

proc freq data=norm.AllFixed20;
  tables Approach*ProbF / nopct nocol;
  format ProbF sig.;
  title2 'Compare Rejection Rates Across Approaches when n1=n2=20';
run;

************************************************************
************************ DE genes **************************
************************************************************;
data DEgenes20;
      set norm.AllFixed20;
      where ProbF<=0.05;
      if Approach="Poisson" then Pois_DE=1; else Pois_DE=0;
      if Approach="NB" then NB_DE=1; else NB_DE=0;
      if Approach="T_test" then Ttest_DE=1; else Ttest_DE=0;
      if Approach="Perm" then Perm_DE=1; else Perm_DE=0;
run;

proc sort data=DEgenes20;
by Gene;
run;

data perm20(keep=Gene Perm_DE) tt20(keep=Gene Ttest_DE) pois20(keep=Gene
Pois_DE) bin20(keep=Gene NB_DE);
      set DEgenes20;
      if Approach='Poisson' then output pois20;
      else if Approach='NB' then output bin20;
      else if Approach='Perm' then output perm20;
      else if Approach='T_test' then output tt20;
run;

data comb_de20;
      *drop ProbF Approach;
      merge norm.raw_data20 perm20 tt20 pois20 bin20;
      by Gene;
run;

data comb_de20;
   set comb_de20;
   array NumVar _numeric_;
```

```
   do over NumVar;
      if NumVar=. then NumVar=0;
   end;
run;

ods output SenSpec=senspec20;
proc freq data=comb_de20 order=formatted;
  tables Perm_DE*True_DE Pois_DE*True_DE NB_DE*True_DE Ttest_DE*True_DE/
senspec;
  format True_DE Perm_DE Pois_DE NB_DE Ttest_DE gen.;
  title2 'True Rejection, Model Accuracy when n1=n2=20';
run;

data stat20;
  set senspec20;
  if Table="Table Perm_DE * True_DE" then table="Permutation";
  else if Table="Table Pois_DE * True_DE" then table="Poisson";
  else if Table="Table NB_DE * True_DE" then table="Neg Binomial";
  else if Table="Table Ttest_DE * True_DE" then table="T test";
  if Statistic="Specificity" then FP=1-Estimate;
run;

proc sort data=stat20;
  by Table;
run;

data sens20 (rename=(Estimate=Sensitivity)) spec20
(rename=(Estimate=Specificity));
  set stat20;
  if Statistic="Sensitivity" then output sens20;
  if Statistic="Specificity" then output spec20;
run;

data stats20;
  merge sens20 spec20;
  by Table;
  keep Table Sensitivity Specificity FP;
run;

proc print data=stats20;
  title2 'Model Sensitivity, Specificity and False Positive Rate when
n1=n2=20';
run;

proc sgplot data=stats20;
  vbar Table/response=Sensitivity;
  title2 'Comparing Model Sensitivity when n1=n2=20';
run;

proc sgplot data=stats20;
  vbar Table/response=Specificity;
  title2 'Comparing Model Specificity when n1=n2=20';
run;

proc sgplot data=stats20;
  vbar Table/response=FP;
  title2 'Comparing Model False Positive Rate when n1=n2=20';
```

```
run;
************************************************************
************************************************************
*******                      n1=n2=25                      *******
************************************************************
************************************************************;

************************************************************
************************************************************
******* Get Rejection Rates (Row 1 Table 11.1 Stroup) *******
************************************************************
************************************************************;

proc freq data=norm.AllFixed25;
  tables Approach*ProbF / nopct nocol;
  format ProbF sig.;
  title2 'Compare Rejection Rates Across Approaches when n1=n2=25';
run;

************************************************************
************************ DE genes **************************
************************************************************;
data DEgenes25;
      set norm.AllFixed25;
      where ProbF<=0.05;
      if Approach="Poisson" then Pois_DE=1; else Pois_DE=0;
      if Approach="NB" then NB_DE=1; else NB_DE=0;
      if Approach="T_test" then Ttest_DE=1; else Ttest_DE=0;
      if Approach="Perm" then Perm_DE=1; else Perm_DE=0;
run;

proc sort data=DEgenes25;
by Gene;
run;

data perm25(keep=Gene Perm_DE) tt25(keep=Gene Ttest_DE) pois25(keep=Gene
Pois_DE) bin25(keep=Gene NB_DE);
      set DEgenes25;
      if Approach='Poisson' then output pois25;
      else if Approach='NB' then output bin25;
      else if Approach='Perm' then output perm25;
      else if Approach='T_test' then output tt25;
run;

data comb_de25;
      *drop ProbF Approach;
      merge norm.raw_data25 perm25 tt25 pois25 bin25;
      by Gene;
run;

data comb_de25;
   set comb_de25;
   array NumVar _numeric_;
   do over NumVar;
      if NumVar=. then NumVar=0;
   end;
run;
```

```
ods output SenSpec=senspec25;
proc freq data=comb_de25 order=formatted;
  tables Perm_DE*True_DE Pois_DE*True_DE NB_DE*True_DE Ttest_DE*True_DE/
senspec;
  format True_DE Perm_DE Pois_DE NB_DE Ttest_DE gen.;
  title2 'True Rejection, Model Accuracy when n1=n2=25';
run;

data stat25;
  set senspec25;
  if Table="Table Perm_DE * True_DE" then table="Permutation";
  else if Table="Table Pois_DE * True_DE" then table="Poisson";
  else if Table="Table NB_DE * True_DE" then table="Neg Binomial";
  else if Table="Table Ttest_DE * True_DE" then table="T test";
  if Statistic="Specificity" then FP=1-Estimate;
run;

proc sort data=stat25;
  by Table;
run;

data sens25 (rename=(Estimate=Sensitivity)) spec25
(rename=(Estimate=Specificity));
  set stat25;
  if Statistic="Sensitivity" then output sens25;
  if Statistic="Specificity" then output spec25;
run;

data stats25;
  merge sens25 spec25;
  by Table;
  keep Table Sensitivity Specificity FP;
run;

proc print data=stats25;
  title2 'Model Sensitivity, Specificity and False Positive Rate when
n1=n2=25';
run;

proc sgplot data=stats25;
  vbar Table/response=Sensitivity;
  title2 'Comparing Model Sensitivity when n1=n2=25';
run;

proc sgplot data=stats25;
  vbar Table/response=Specificity;
  title2 'Comparing Model Specificity when n1=n2=25';
run;

proc sgplot data=stats25;
  vbar Table/response=FP;
  title2 'Comparing Model False Positive Rate when n1=n2=25';
run;

************************************************************
************************************************************
```

```
*******                    n1=n2=30                     *******
************************************************************
************************************************************;

************************************************************
************************************************************
******* Get Rejection Rates (Row 1 Table 11.1 Stroup) *******
************************************************************
************************************************************;

proc freq data=norm.AllFixed30;
  tables Approach*ProbF / nopct nocol;
  format ProbF sig.;
  title2 'Compare Rejection Rates Across Approaches when n1=n2=30';
run;

************************************************************
************************ DE genes **************************
************************************************************;
data DEgenes30;
      set norm.AllFixed30;
      where ProbF<=0.05;
      if Approach="Poisson" then Pois_DE=1; else Pois_DE=0;
      if Approach="NB" then NB_DE=1; else NB_DE=0;
      if Approach="T_test" then Ttest_DE=1; else Ttest_DE=0;
      if Approach="Perm" then Perm_DE=1; else Perm_DE=0;
run;

proc sort data=DEgenes30;
by Gene;
run;

data perm30(keep=Gene Perm_DE) tt30(keep=Gene Ttest_DE) pois30(keep=Gene
Pois_DE) bin30(keep=Gene NB_DE);
      set DEgenes30;
      if Approach='Poisson' then output pois30;
      else if Approach='NB' then output bin30;
      else if Approach='Perm' then output perm30;
      else if Approach='T_test' then output tt30;
run;

data comb_de30;
      *drop ProbF Approach;
      merge norm.raw_data30 perm30 tt30 pois30 bin30;
      by Gene;
run;

data comb_de30;
   set comb_de30;
   array NumVar _numeric_;
   do over NumVar;
      if NumVar=. then NumVar=0;
   end;
run;

ods output SenSpec=senspec30;
proc freq data=comb_de30 order=formatted;
```

```
  tables Perm_DE*True_DE Pois_DE*True_DE NB_DE*True_DE Ttest_DE*True_DE/
senspec;
  format True_DE Perm_DE Pois_DE NB_DE Ttest_DE gen.;
  title2 'True Rejection, Model Accuracy when n1=n2=30';
run;

data stat30;
  set senspec30;
  if Table="Table Perm_DE * True_DE" then table="Permutation";
  else if Table="Table Pois_DE * True_DE" then table="Poisson";
  else if Table="Table NB_DE * True_DE" then table="Neg Binomial";
  else if Table="Table Ttest_DE * True_DE" then table="T test";
  if Statistic="Specificity" then FP=1-Estimate;
run;

proc sort data=stat30;
  by Table;
run;

data sens30 (rename=(Estimate=Sensitivity)) spec30
(rename=(Estimate=Specificity));
  set stat30;
  if Statistic="Sensitivity" then output sens30;
  if Statistic="Specificity" then output spec30;
run;

data stats30;
  merge sens30 spec30;
  by Table;
  keep Table Sensitivity Specificity FP;
run;

proc print data=stats30;
  title2 'Model Sensitivity, Specificity and False Positive Rate when
n1=n2=30';
run;

proc sgplot data=stats30;
  vbar Table/response=Sensitivity;
  title2 'Comparing Model Sensitivity when n1=n2=30';
run;

proc sgplot data=stats30;
  vbar Table/response=Specificity;
  title2 'Comparing Model Specificity when n1=n2=30';
run;

proc sgplot data=stats30;
  vbar Table/response=FP;
  title2 'Comparing Model False Positive Rate when n1=n2=30';
run;

ods pdf close;
```

SimSeq Simulation

```
#if (!requireNamespace("BiocManager", quietly = TRUE))
  install.packages("BiocManager")
#BiocManager::install(c("edgeR","DESeq2","tweeDEseqCountData","tweeDEseqCount
Data"))
BiocManager::install("DescTools")
setwd("X:/Desktop/PhD_Project/PhD_Project/proposal/Recom_reading/Rcode/LFC2/n_
10")
### Load Bioconductor Packages
require(DESeq2)
require(edgeR)
require(limma)
require(tweeDEseqCountData)
library(NOISeq)
### Load CRAN Packages
require(QuasiSeq)
require(samr)
require(fdrtool)
require(SimSeq)
library(fdrtool)
library(DescTools)
```

```
filter.mean <- 10 # lower bound of average read count for simulated genes
filter.nonzero <- 2 # lower bound for nonzero read counts for simulated genes


#data
data<-readRDS("C:/Users/bm4817gu/OneDrive for Business/Downloads/GTEx.StoPan.rds")
genes<-data[,"Gene"]
counts <-subset(data, select = -Gene )  # Matrix of read counts from KIRC dataset
rownames(counts)<-genes
replic <- names(counts) # Replic vector indicating paired columns
treatment <- c(rep("Pan",328),rep("Sto",359)) # Treatment vector indicating Non-Tumor or Tumor columns
### Remove low count genes
keep.counts <- ( rowMeans(counts) >= filter.mean ) & ( rowSums(counts > 0) >= filter.nonzero )
counts <- counts[keep.counts, ]
dim(counts)
### Compute normalization factors to use in SimData function
### from calcNormFactors
lib.sizes <- apply(counts, 2, sum)
nf <- calcNormFactors(counts) * lib.sizes
```

```
### Calculate weights vector beforehand to save run time in
### repeated simulations
sort.list <- SortData(counts = counts, treatment = treatment, replic = NULL,
                      sort.method = "unpaired", norm.factors = nf)
counts <- sort.list$counts

replic <- sort.list$replic
treatment <- sort.list$treatment
nf <- sort.list$norm.factors
probs <- CalcPvalWilcox(counts, treatment, sort.method = "unpaired",
                        sorted = TRUE, norm.factors = nf, exact = FALSE)
weights <- 1 - fdrtool(probs, statistic = "pvalue", plot = FALSE, verbose = FALSE)$lfdr

### Simulate matrix with DE genes having log base 2 fold change greater than 1
# add one to counts matrix to avoid infinities when taking logs
Pan.mean <- rowMeans(log2((counts[, treatment == "Pan"] + 1) %*%
                          diag(1/nf[treatment == "Pan"])))
Sto.mean <- rowMeans(log2((counts[, treatment == "Sto"] + 1) %*%
                          diag(1/nf[treatment == "Sto"])))
lfc <- Pan.mean - Sto.mean
weights.zero <- abs(lfc) < 1
weights[weights.zero] <- 0
```

```
data.sim <- SimData(counts = counts, replic = replic, treatment = treatment,
                    sort.method = "unpaired", k.ind = 10, n.genes = 5000, n.diff = 1000,
                    weights = weights, norm.factors = nf)

counts.simseq <- data.sim$counts # Simulated Count matrix from SimSeq
genes.samp <- data.sim$genes.subset # Genes sampled from source matrix
de.genes <- data.sim$DE.genes # DE genes sampled from source matrix
ee.genes <- genes.samp[ ! (genes.samp %in% de.genes) ] # EE genes sampled from
source matrix
samp.col <- data.sim$col # Columns sampled in SimSeq algorithm
de.genes.sim <- data.sim$genes.subset %in% de.genes # logical vector giving which
genes are DE in simulted matrix
trt<-data.sim$treatment

ee.genes <- sample(which(!de.genes.sim))
de.genes <- sample(which(de.genes.sim))
DE<-counts.simseq[de.genes, ]
counts.simseq_10 <- counts.simseq[c(ee.genes, de.genes), ]
################################################################################
##
write.csv(counts.simseq_10, file = "data.sim_n10.csv")
write.csv(DE, file = "de_genes.sim_n10.csv")
```

DE Genes Assessment Using R packages

```
        ##############################################################################
##########################3
        #if (!requireNamespace("BiocManagerBiocManager", quietly = TRUE))
        #  install.packages("BiocManager")
        #BiocManager::install("edgeR")
        #BiocManager::install("tweeDEseqCountData")
        #install.packages("statmod")
        #BiocManager::install("DESeq2", type="source")
        #BiocManager::install("GenomeInfoDbData", type="source")
        #BiocManager::install("tibble", type="source")
        #BiocManager::install("apeglm")
        setwd("X:/Desktop/PhD_Project/PhD_Project/proposal/Recom_reading/Rcode/LFC2/n_
10")
        library(edgeR)
        library(statmod)
        library(tweeDEseqCountData)
        library(apeglm)
        library(GenomeInfoDbData)
        library(tibble)
        library(DESeq2)
        library(limma)
```

```
        ##############################################################################
##
        count_10<-read.csv("data.sim_n10.csv",header=TRUE, sep = ',')
        gene_10<-count_10[,"X"]
        count_10<-count_10[,-1]
        rownames(count_10)<-gene_10
        head(count_10)
        dim(count_10)

        group<-matrix(c(rep("Pan",10),rep("Sto",10)),ncol=1)
        design<-model.matrix(~group)
        colnames(design)<- c("Pan","Sto")

        ##############################################################################
##
        ### edgeR Analysis
        ##### DGEList

        y <- DGEList(counts=count_10, genes=gene_10,group=group)
        head(y$count)
        head(y$samples)
        head(y$genes)
```

```
##Determine which genes are expressed in a worthwhile number of samples.
isexpr <- filterByExpr(y,group=group)
table(isexpr)

##Keep only expressed genes with defined annotation and
##recompute library sizes
y <- y[isexpr, keep.lib.sizes=FALSE]
dim(y$count)
head(y$count)
head(y$samples)

##Create barplot of library sizes
barplot(y$samples$lib.size*1e-6, names=1:20, ylab="Library size (millions)")

##Apply TMM normalization
y <- calcNormFactors(y)
head(y$samples)

##Estimate dispersion parameters
y <- estimateDisp(y)

##Perform exact test
et <- exactTest(y)
```

```
padj<-topTags(et, n=dim(y$count)[1])$table[,-2:-3]
head(padj)

#saveRDS(padj,"fdr_edgeR10.RDS")

## genes that are DDE when controlling FDR at 5%.
sum(padj$FDR<.05, na.rm=TRUE)
padj<-na.omit(padj)
write.csv(padj, file = "fdr_edgeR10.csv")
DDE<-padj[padj$FDR<0.05,]
name1<-rownames(DDE)
head(name1)

## genes that are DDE when controlling FDR at 10%.
sum(padj$FDR<.1, na.rm=TRUE)
DDE<-padj[padj$FDR<.1,]
de.edgeR<-rownames(DDE)
head(name1)
##############################################################################
##
```

```
### DESeq2 Analysis

##using the DESeq2 procedure with no LFC cutoff.

dim(count_10)


dds <- DESeqDataSetFromMatrix(countData=count_10, colData=group, design=design)

dds


##perform DESeq2 method

dds2 <- DESeq(dds)


##get results

res <- results(dds2)

resultsNames(dds2)

summary(res)

names(res)

presadj<-res[,-1:-4]


#saveRDS(presadj,"fdr_DESeq210.RDS")

#get genes that are DDE

sum(presadj$padj<.1, na.rm=TRUE)

presadj<-na.omit(presadj)

write.csv(presadj, file = "fdr_DESeq210.csv")

DDE3<-presadj[presadj$padj<0.1,]
```

```
dim(DDE3)

de.desq2<-rownames(DDE3)

head(de.desq2)


res1 <- res

##############################################################################
##

##shrink log fold change estimates

res <- lfcShrink(dds2, coef="Sto", type="apeglm")

summary(res)



##get results for an LFC cutoff of 1

resLFC <- lfcShrink(dds2, coef="Sto", lfcThreshold=1, type="apeglm")

summary(resLFC)

pLFCadj<-resLFC[,-2:-3]


#get genes that are DDE

sum(pLFCadj$svalue<.1, na.rm=TRUE)

pLFCadj<-na.omit(pLFCadj)

write.csv(pLFCadj, file = "fdr_LFC10.csv")
```

```
DDE4<-pLFCadj[pLFCadj$svalue<0.1,]
de.lfc<-rownames(DDE4)
head(de.lfc)
##########################################################################
##
##Plot results
plotMA(res1)
plotMA(res)
plotMA(resLFC)
##########################################################################
##
### Limma Voom Analysis

### Limma Voom Analysis

y <- DGEList(counts=count_10, genes=gene_10,group=group)
isexpr <- filterByExpr(y,group=group)
table(isexpr)
y <- y[isexpr, keep.lib.sizes=FALSE]
y <- calcNormFactors(y)

##limma trend
##calculate logCPM values
```

```
##(edgeR function)
lcpm <- cpm(y, log=TRUE)

##perform limma trend
##(limma functions)
fitt <- lmFit(lcpm, design)
efitt <- eBayes(fitt, trend=TRUE)

##p-values for testing difference in gene expression
##between males and females
head(efitt$p.value)
ltp <- efitt$p.value[,2]

##plot mean-variance trendline
plotSA(efitt, ylab="(Standard Deviation)^(1/2)", cex.lab=1.5)

##Get top 10 results
topTable(efitt, coef=2, n=10)
summary(decideTests(efitt))

#get genes that are DDE
limma.Pvalu<-topTable(efitt, coef=2, n=dim(y$count)[1])[,4:5]
sum(limma.Pvalu$adj.P.Val<.1, na.rm=TRUE)
```

```
limma.Pvalu<-na.omit(limma.Pvalu)

write.csv(limma.Pvalu, file = "fdr_limma10.csv")


########################################

########################################

##voom

v <- voom(y, design, plot=TRUE)

fitv <- lmFit(v, design)

efitv <- eBayes(fitv)


##p-values for testing difference in gene expression

##between males and females

head(efitv$p.value)

lvp <- efitv$p.value[,2]


##plot mean-variance trendline

plotSA(efitv, ylab="(Standard Deviation)^(1/2)", cex.lab=1.5)


##Get top 10 results

topTable(efitv, coef=2, n=10)

summary(decideTests(efitv))
```

```
voom.pv<-topTable(efitv, coef=2, n=dim(y$count)[1])[,5:6]

sum(voom.pv$adj.P.Val<.1, na.rm=TRUE)

voom.pv<-na.omit(voom.pv)

write.csv(voom.pv, file = "fdr_voom10.csv")
```