Internal Report CS Bioinformatics Track January 2019

Leiden University

Computer Science

Bioinformatics Track

Performance Evaluation of
Transcript-level RNA-Seq Aligners
'HISAT vs STAR'

Mina Parsania

MASTER'S THESIS

Leiden Institute of Advanced Computer Science (LIACS)
Leiden University
Niels Bohrweg 1
2333 CA Leiden
The Netherlands

Performance Evaluation of Transcript-level RNA-Seq Aligners

‘HISAT vs STAR’

Mina Parsania (1321463)
Supervisor: Dr. E. M. Bakker
01.20.2019

Abstract

Nowadays RNA-seq is a widely-used technique being applied in an extensive variety of sequence-based research. RNA-seq facilitates the analysis and discovery of novel genes and transcripts. Most notably RNA-seq can be used to identify differentially expressed genes or transcripts by differential statistical analysis. For example, RNA-seq used in a study done on Chinese hamster ovary all lines identified several differentially expressed genes. The experiment outputs show a total of 572 upregulated and 924 downregulated protein-coding genes in CHO-K1 cells when comparing with CCL39 cells [15]. RNA-seq has been used in many cancer researches [14], hereby RNA-seq shows a higher throughput than classical microarray analysis [16]. As RNA-seq does not rely on pre-designed complement probes, a significant advantage of RNA-seq is avoidance of technical issues existing in Microarray-technique such as cross-hybridization [18]. RNA-seq data sets can be very extensive leading to several challenges w.r.t. performance. For example, read-alignment, a preprocessing stage of RNA-seq, requires a lot of memory and computations. Pertea et al. proposed an efficient and effective protocol for analyzing RNA-seq data using HISAT2, StringTie and Ballgown [1]. In their study and evaluations, they used down-sampled data. In this work, we aim to re-evaluate the whole pipeline of Pertea et al. on the same down-sampled data as well as the full set of data to study the scalability of their protocol. Furthermore, a comparison of the impact of the aligners HISAT2 and STAR on performance, quality and scalability in the protocol proposed by Pertea et al. is presented.

1. Introduction

DNA sequencing (DNA-Seq) plays an essential role in mapping out the human genome. In DNA-Seq it is possible to determine the complete DNA sequence of an organism's genome. DNA sequencing is also the most efficient way to determine the sequence at RNA or proteins. However, the transcriptome specifically holds what genes are expressed at a certain moment in time. Thus RNA-seq, sequencing the transcriptome, allows us to analyze various important aspects such as alternative gene splicing, gene fusion and changes in gene expression leading to more detailed insights in all kinds of processes in the cells.

The RNA-seq technique is applied to many different experimental designs for instance, experiments done under different biological conditions, as well as experiments periodically during a period of time (time-series data). For example, in [3] dynamic biological processes on gene expression data is studied and modelled. In [4] several methods that can be applied to model time-course RNA-seq data are discussed. One of the popular approaches in RNA-seq is differential expression analysis [1,14,15]. Differential expression analysis can be used to study stages of which genes or transcripts are expressed at a significantly higher or lower number in different tumor tissues versus healthy tissues.

In RNA-seq studies typically many reads of the transcriptome are used. The RNA-seq processing pipeline for analyzing the data consists of aligning these reads to the genome, assembling the transcripts and merging them to cover the unknown parts of the reads mapped on the genome by those already discovered. The output data is used as input for further downstream analysis.

Next Generation Sequencing technique produces millions of small segments called reads to map them throughout the whole reference genome using different bioinformatic tools. Two of well-known and mostly used transcriptomic aligners are HISAT and STAR. The goals of the current work is to firstly validate Pertea et al's work as well as performing comparisons between HISAT2 and STAR. We do this since it is essential to find a suitable aligner among many by considering the characteristics of the data we intend to analyze (for example paired-end reads/single end reads, the length of the reads, etc.). In [1] it is emphasized that HISAT is the fastest aligner among the rest. This encourages us to find out if HISAT2 gives still better performance rather than STAR (STAR is a popular aligner with a relatively high rate of accuracy, however it is memory-intensive). STAR is an ultrafast two-step alignment that first splice junctions are detected then they are used to further map the reads. it helps in increasing the alignment accuracy.

Pertea et al. [1] proposed a so-called *"New Tuxedo"* RNA-seq analysis pipeline consisting of transcriptomic analysis of reads I) using HISAT to map the reads,II) StringTie, to count the abundances and III) Ballgown to find differentially expressed genes and transcripts. In [1] the

reads are mapped against only chromosome X to make the protocol faster and simpler. It is notable that StringTie is also used to cover the unknown information of reads by those of known reads and. As a result of all these steps, genes and transcripts lowly or highly expressed are discovered.

In the proposed *New Tuxedo* pipeline HISAT2 is applied to map RNA-seq paired-end samples to the given genome. STAR is studied as an alternative aligner in the *New Tuxedo* protocol by Pertea et al. The STAR aligner is compared on performance, scalability and accuracy. We also choose STAR (as one of the well-known spliced transcripts aligners) to work with since it outperforms many other aligners, such as GSNAP, in speed on human samples while it preserves sensitivity and precision [2]. Another advantage of STAR is that non-contiguous reads are directly mapped to the genome with the help of a read-clipping technique called soft-clipping. Soft clipping helps to improve the accuracy of the mapping. Furthermore, STAR uses the Maximal Mappable Prefix (MMP) search algorithm and uncompressed Suffix Arrays (SA) rather than compressed SA (as many short-read aligners use) to increase the accuracy rate [2]. Also, STAR alignments are deterministic. However, it is changed to a non-deterministic aligner if for example multiple threads are used. In that case different executions of the same data may lead to different orderings of the reads. Alternative aligners such as GSNAP are much faster than many other aligners but still slower than STAR. The Tophat aligner is also much slower than STAR and does not apply read clipping for partial reads (reads which are not completely matched with a section/sections on the reference genome) leading to increase of the unmapped reads rates (the percentage of reads which are unmapped).

The STAR aligner shows a competitive run in comparison with HISAT2 in some studies. For instance, Baruzzo et al. [9] observed in an experiment they performed that STAR outperforms HISAT2 when working on human data set samples with a recall of >=97%. Compared to the observed recall of HISAT2 in Baruzzo et al.'s [8] STAR gives a better performance than HISAT2 on human data sets. The better performance consists of both a higher percentage of reads aligned correctly, as well as a higher precision and recall rate on a simulated human data set. Although there is an exception on one set of human data showing no significant difference between the results for both mappers. This inspired us to process the data used in Pertea's paper by STAR and see if this re-map leads to with respect to any advancements in the results with respect to alignment accuracy performance.

The goal of our study is to critically evaluate the proposed Pertea's New Tuxedo RNA-seq pipeline (PNT). Pertea et al. [1] use Hisat2, StringTie and Ballgown in their suggested processing pipeline. The goals of our study is to evaluate the performance of STAR in PNT as well as to optimize STAR parameters, Alexander Dobin et al. [2] imply, when aligning to genomes other than those containing smaller introns, STAR's default parameters seem to operate well, but when this is not the case, we need to change parameters to obtain the optimal

state for a specific data set. Additionally, the RNA-seq design is limited by the budget dedicated to the analysis. To achieve the highest power by choosing the optimal parameters and still being under the defined budget is also one of the goals. Here we investigate several parameters to evaluate any improvements in the mapping results. We change the parameters depending on the structure of data sets being studied, different parameter optimization is needed as explained above. For example, if the genome to be mapped to has a small size, then the appropriate parameter will be scaled similarly if we process data with smaller intron sizes.

Pertea et al. [1] suggest that Hisat2, StringTie and Ballgown is the optimal software packages to be used to analyze the human transcriptomic paired-end data. Specifically, in contrast to other mapping software packages, their proposed protocol pipeline decreases the calculation time a lot while maintaining a good level of quality of the alignment and hence results.

In this study we conducted several experiments to evaluate the performance of HISAT2 in PNT compared to the STAR aligner used in PNT instead. We found that HISAT2 is still more rapid than STAR. However, the difference is small for data sets indicating they work well for smaller data sets, but the time spent to map increases as data grows. We also tested this on two different comparing systems of 8GByte (intel core i7) and 64GByte (intel core i7). main memory respectively. It is clear that more available main memory plays a positive role in decreasing the mapping time for STAR, but HISAT2 is still aligning the reads faster than STAR. Although HISAT2 is faster, we observed that STAR represents higher rates of correctly aligned reads. This encourages us to remove false positive rates out of the results. So, parameter optimization could be done to get true positive rates. Finally, we conclude that STAR is better choice to work with in this specific protocol PNT. Since by tuning parameters correctly mapped reads rates as well as mismatch rates are respectively higher and lower than when we use HISAT2 in PNT, executed on full data.

In our study, we take the same pipeline of Pertea et al., PNT to work with. The software packages used work well without any restrictions. The pipeline is applied on the same data as they used in their work to verify their findings. Furthermore, as mentioned above, we increased the volume of data by using the full data sets in order to analyze and evaluate the scaling of the pipeline in various settings.

The rest of the paper is organized as follows. In Section 2, we describe the new tuxedo pipeline and explain each software's function used in the pipeline as well as its strategy to work. In Section 3 the experimental setup of Pertea's work [1] as well as of our experiment are explained. Notably, there are two different phases in our experiment. In the first phase PNT (Pertea's New Tuxedo) is implemented. In the second phase, the alternative STAR aligner is used in PNT as well as parameter optimization for both aligners. It should be mentioned that during the first phase the data are the same as used by Pertea [1] however in the second phase we use the same down-sampled data as well as down-sampled data with different sizes of

20%,40%,80% of full data, and the full data. In the next section hardware specifications are mentioned. Both phases of the experiment are explained in Sections 5 and 6. In Section 7, parameters influencing the alignment results of HISAT2 and STAR are studied. The results are given in Section 8. Finally, CPU and memory usage are investigated for both the aligners.

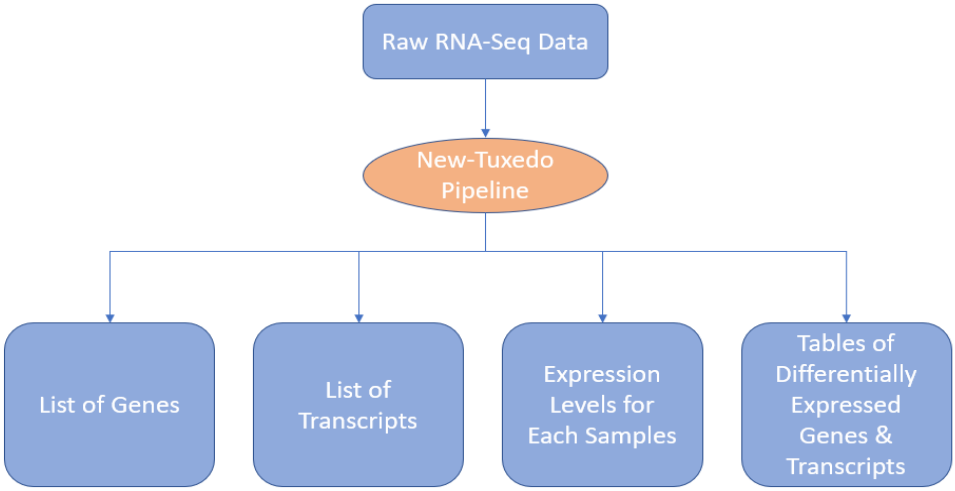


Figure 1 Overview of PNT pipeline.

Before we start to explain two phases of our experiment, it is required to describe some some statistical terms and definitions used in the current work:

- Fold-change: in order to discover how a quantity changes from an initial value to a final value the fold-change is analyzed. For instance, if a gene is upregulated by 2 fold change, it means that the final value is two times bigger than the initial value.
- p-value: the probability value for a given statistical model (in case of true null hypothesis) is equal or greater than the actual observed results. For example, p-val 0.05 means a strong evidence against the null hypothesis. So, we may reject the null hypothesis.
- q-value the minimum *False Discovery Rate* at which a test may be called significant. In q-value the traditional threshold is 0.01-0.05. 0.01 indicates that 1% of the significant results is null.

2. Overview of the "New Tuxedo" Pipeline using HISAT, StringTie and Ballgown

In [1] the new tuxedo for the processing pipeline PNT for RNA-seq data is proposed. PNT consists of the following processing steps:

0) The sample reads, RNA-seq data, given as a GTF file, are inputs of PNT.
1) Hisat2 aligns the reads in order to map them on the genome.
2) The mapped reads are counted by StringTie to have an insight on the abundances of mapped reads. Subsequently, merging function is applied for covering the reads. merging function is also applied for covering the reads.
3) Gffcompare in PNT is used to determine how many assembled transcripts match annotated genes as well as discovering entirely novel transcripts.
4) The Ballgown analysis software is used to find which genes and transcripts are up and downregulated.
5) The results are given as tables of up and downregulated genes and transcriptomes.

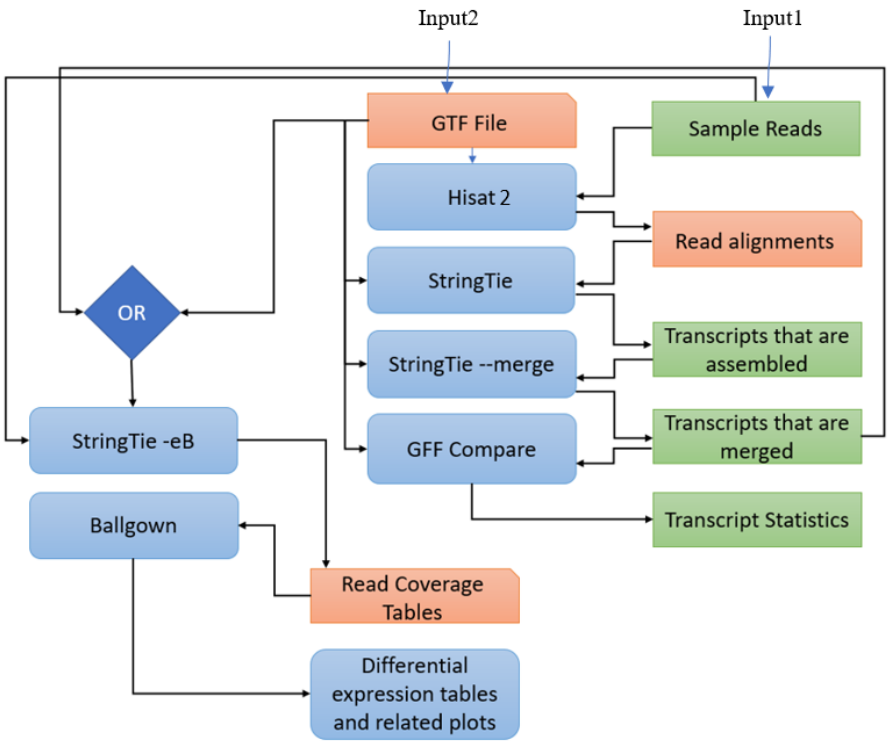


Figure 2 An overview of PNT, the "new tuxedo" protocol [1]. Sample reads and GTF files enter to the process. Mapping the reads against a genome is the first step in the processing pipeline. In the following, StringTie calculates the abundances of reads. In the end, Ballgown performs the statistical analysis and gives the lists of down and up regulated genes and transcripts.

2.1 Mapping Reads: HISAT

HISAT2 performs the method of exon-first (such as Mapsplice [6]) on samples reads. The exon-first method consists of two steps. First, it applies the unspliced method to map all the reads to the whole genome, next, it splits the remaining reads to shorter ones and align them again to the genome [7].

Indexing techniques in HISAT2: Before starting to map the reads to reference genome, it is required to have genome indices beforehand (Each index contains a genomic region). HISAT2 uses two kinds of indexing for aligning, Burrows-Wheeler transform (BWT) and the Ferragina-Manzini (FM) index. Global and local FM indexing strategies help anchoring and extending of the alignments faster. However, to speed up the execution, HISAT2 uses a multi-threading model that helps the package accomplish the job within a short time [17]. In FM index-based aligners, the exact matches are found and then the exact alignment are created from the exact matches. The FM indices cover the whole genome to improve mapping to the reference genome.

Locations of annotated genes: PNT uses HISAT2 for mapping of the RNA reads from the transcriptome to the genome. HISAT2 has two inputs the *reference annotation* and *the RNA reads* (see Appendix A). Notably, to speed up the mapping time, positions of known genes/transcripts annotations can also enter the procedure as an extra input leading to a speedy mapping process by skipping to find the locations of annotated genes as they are already included in the third input (also available for STAR). If those locations are not already offered to HISAT2, then HISAT2 will find them by itself. Although it will take longer time to map the reads.

2.2 Transcript Assembly: StringTie

Generally, power of StringTie in accurate reconstruction of genes is already tested [8]. Level of estimation of StringTie in comparison with Cufflinks, IsoLasso, Scripture and Traph is higher in both real and simulated data sets.

After aligning reads to the genome, StringTie program is used to I) assemble the transcripts mapped on the genome. Similarly, in this step reference annotation is required. The GTF file and mapped RNA reads as inputs are entered to StringTie. Then StringTie assembles transcripts from mapped RNA-Seq reads. II) Estimating the expression level of each gene is also done by StringTie. After the assembly process III) the *--merge* function merges all the gene structures in each of the samples to create a unifies set of transcripts across different samples. In other words, we may use this feature to create a consensus transcriptome reference GTF from multiple samples. The StringTie's merge function finds and fixes those transcripts which are not well-covered in the previous performance. So, this function merges all transcripts of all the samples together. Since the covered transcripts may improve the results. The result

is again fed to the StringTie to IV) re-estimate abundances of transcripts once again. Finally, it passes transcripts and abundances to Ballgown for statistical analysis.

StringTie takes the advantages of the network (see Figure 3) to assemble and quantify the reads by building graphs. The paths of the graph represent the weights of coverages. It consists of amounts of abundances for the reads. In other words, reads including more repeat times in the samples are heavier in the network of StringTie as well (see Figure 3) [8]. Next, the assembled reads are counted by FPKM. The FPKM counts the total number of fragments instead of number of reads (RPKM). Subsequently, the results are entered to StringTie's merge function to merge all data. It is notable that gffcompare serves to compare the GTF file and transcript query obtained from the previous step of the experiments.

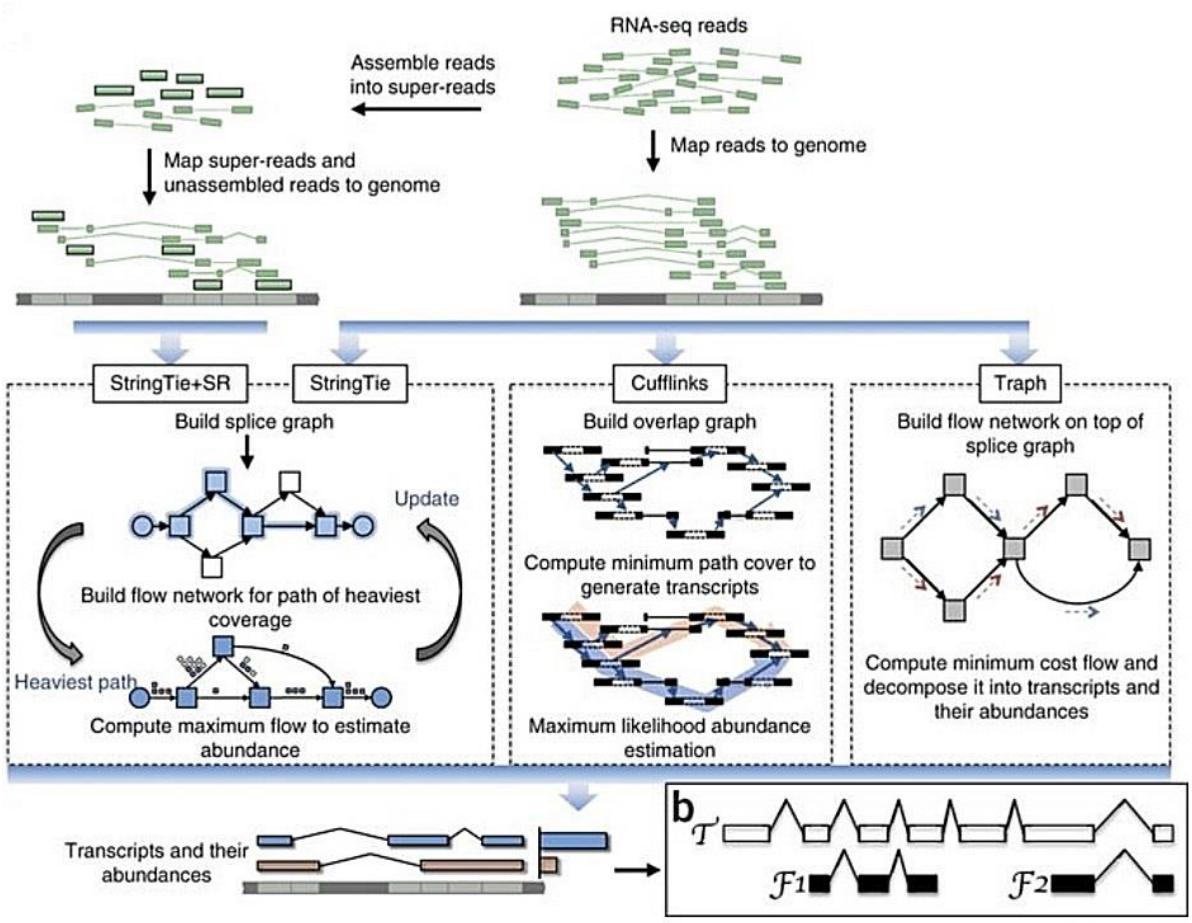


Figure 3 an overview of StringTie works, picture from [8]

2.3 Differential expression of genes and transcripts: Ballgown

In this step, read-counts and transcripts are both fed to the Ballgown. Default parameters are used in this study. After grouping data to two different conditions, Ballgown outputs the differentially expressed genes as tables (Table1 and Table2). As mentioned, the Ballgown the R package takes the output of StringTie (abundance of transcripts) as inputs. It performs some statistical analysis on them such as flexible differential expression analysis, visualization of transcript structures, and matching of assembled transcripts to annotation. The results are summarized and visualized in downstream analysis. As shown in Tables, 9 transcripts are differentially expressed on chromosome X between two sexes (q-val threshold 0.05) as well as 10 genes at the gene level with the same cut-off of 0.05.

3. Experimental Setup

As already mentioned, the goal of our study is to validate the PNT as well as to compare HISAT2 and STAR. The study is done in two different phases. In the first phase, PNT is run and the same results as [1] output. However, in the second phase a comparison is performed on HISAT2 and STAR performances considering different factors such as speed, memory usage, accuracy of mapping (percentage of uniquely mapped reads).

Pertea et al. [1] used 12 RNA-seq transcriptomic paired-end human samples. They are all obtained from Illumina hiseq 2000 including two sets of British from England (GBR) and Yoruba from Ibadan in Nigeria (YRI). Each group consists of 6 samples to be assessed in which half is male. Notably, in the library preparation, the experiment is done on 465 lymphoblastoid cell lines from the 1000 Genomes for the certain sample of ERR188337. All the samples are human samples (organism, homo sapiens). Both the forward and reverse reads lengths are 75 base pairs each (for example in ERR188337). The reads worked on in this study are of modest sizes. However shorter reads can save the resources, the longer reads improve splicing detection. Thus, a suitable read-length depends on the ultimate goal of the study. Similarly, in PNT a modest size of data is used to save time and resources as well as to provide scalable read-alignment. Moreover, to generate robust results, the number of biological replicates should not be less than 3 [5]. Considering these factors, the ultimate results is more reliable helping to get more real output.

The first part of our analysis (re-doing the pipeline exactly as the Pertea's paper) showed the same results as [1]. Similarly, the number and name of the genes and transcripts differentially expressed are the same as the paper (see Tables 1 and Table 2).

In our experiments the data is used in the two different phases of the experimental setup. The first sets of data are those used in Pertea et al.'s work [1]. In the second phase we test 10 percent of full data set and we increase the amount of reads by 20, 40, 80 to finally 100 percent of the full data set. It is done for all the 12 samples using both HISAT2 and STAR.

3.1 Phase I: Experimental Setup

In PNT [1] the sub-samples are tested using HISAT2, StringTie and Ballgown (their suggested protocol pipeline). Mapping process is done by HISAT2. StringTie calculates the abundances of reads mapped on the reference genome. Then it passes the results of analysis to Ballgown. Ballgown statistically investigate the results and highlights those genes and transcripts which are up and downregulated. Our experiment starts with re-performing the whole pipeline of PNT on the same data as Pertea et al. used. In the end, the same differentially expressed genes and transcripts are reported.

4. Hardware

Pertea et al. [1] recommend using a machine with the following hardware specifications:

- Hardware: 64-bit computer running either Linux or Mac OS X (10.7 Lion or later); 4 GB of RAM (8GByte preferred).

In our experiment, the evaluation and experiments are performed on two machines with the following specifications (see Appendix D):

- Intel(R) Core (TM) i7-7700HQ CPU, 4 Cores, 2.80GHz 8GByte of main memory and Ubuntu 17.04 64-bit
- Intel Core i7-5820K 3.3 GHz, 12 Cores, 64GByte main memory and Ubuntu 16.04 LTS 64-bit.

5. Experimental Results of Phase I: Evaluating the Performance of STAR vs HISAT2 in (PNT)

As mentioned earlier, our experiment consists of two phases. In the first phase the work of Pertea et al. [1] is performed and evaluated. In this phase, exactly the same set of data they used is being tested in our experiment.

5.1 Mapping

We implemented the PNT pipeline of [1] in the first phase of our experiment. Similar to PNT, HISAT2 aligns the reads of 12 subsamples (two groups of GBR and YRI already mentioned) against the reference genome (chromosome X). Data are paired-end reads with 75 bp long. All are obtained from 465 lymphoblastoid cell lines from the 1000 Genomes.). After aligning the reads, StringTie calculates the abundances and finally Ballgown statistically investigate the results of the last step to find which genes and transcripts are highly or lowly expressed between two groups. Like PNT, the results we obtained shows that among 9 differentially expressed transcripts 3 of which (XIST, TSIX, PNPLA4) correspond to isoform of known genes. Notably, like PNT, we also use the 8 processors of our system to run the HISAT2 for

each of the sub-samples as [1]. We used the measure *Fold-change* to describe how much a quantity of expression changes.

There are several important factors influencing the results of RNA-seq techniques. The two features of sensitivity and precision of an aligner play an important role in the quality of the true positive results of the final step. *Sensitivity* or *recall* means for instance the percentage of genes which are correctly identified as highly-expressed as having the condition. Additionally, *precision* is the proportion of positive results. There are several factors that lead to bad alignments of the reads: for example, there is a false positive rate factor that should be controlled by aligner. It is claimed that HISAT2 is able to cope with such an issue. Furthermore, if a bad aligner is chosen for mapping to the genome, the unaligned significant percentage of reads will lead to fake outcome when it consumes more time to map on the genome as well.

Clearly, multimapped reads affect the mapping results. In HISAT2 as well as in many other aligners, they are ignored to map on the reference genome. Solving this issue will highly increase the final rate of mapped reads leading to have a better understanding from data [1].

Based on what we observe from samples report using HISAT2, the overall mapping rates are above 95% in which the rates of uniquely mapped alignments are between 70 and 80 percent (see Figure 4). Moreover, the multi-mapped (see Figure 5) and unmapped (see Figure 6) reads rates except two samples which exceed 20 % are between 10 and 20 percent.

5.2 Assembly

PNT uses an annotation file to do the assembly of reads. The annotation file is in HISAT2. Notably, it is recommended that the user takes this file in the analysis if it is available. Since it will improve the assembly process and leads to better accuracy. It contains the locations of all known genes with their introns and exons boundaries and splicing events. It clearly also improves the speed of mapping, since the known genes will not again be sought. The annotation file is also used in the merge function of the protocol. In PNT the annotation file contains 2098 transcripts from 1086 genes on chromosome X. Similar to PNT, in our experiment, the transcriptome assembly statistics represent 908 assembled genes when it is used without annotation. However, it is 1258 when supplying the annotation file. The statistics of transcripts matching annotation also show 661 transcripts while it is 1408 transcripts when it is merged with annotation.

To observe how the assembled transcripts, differ from the given GTF file, *gffcompare* is used in PNT [1]. The *Gffcompare* compares the genome annotation file using the result of StringTie. Such options can assess the accuracy of pipelines by comparing the results with the known reference annotation.

5.3 DE analysis

To find out how much of each transcript is expressed we can use the FPKM factor. As mentioned earlier, the assembled reads are counted by FPKM. The FPKM counts the total number of fragments in genes instead of number of reads (RPKM) by using log2 of the values. As a result, to stabilize the variance, log 2 transformation is applied to FPKM results. There is only one difference between RPKM and FPKM that is FPKM considers that two reads can map to one fragment. As a result, a fragment is not counted twice. After transformation we see that there are a very few fractions of genes that are highly expressed. We already know that, generally genes with lower level of expression will be challenging to be reconstructed. Since such reads are not well-covered and as a result are not informative to study.

The result tables of differentially expressed genes and Transcripts (see Tables 1 and 2 respectively) are the same as PNT. It shows that three known genes (XIST, TSIX and PNPLA4) are existed in both the males and females. Although the expression level of PNPLA4 is higher in females than males. In the biology of genes, TSIX is a non-coding gene that attaches to XIST to inactivate chromosome X [1]. In our experiment, we can decide which threshold to be used to extract differentially expressed genes and transcripts. Here the q-value of 0.05 is chosen as the threshold. Notably, the tables show that there are no large differences between sexes. If there would large differences, then we witnessed many p-values near zero.

Table 1 Differentially expressed transcripts between sexes. The table below represents the lists of up or down regulated transcripts. Chromosome X has 9 differentially expressed transcripts between the two groups. By considering a q-val of 0.05. three of them are isoforms of the known genes which are XIST, TSIX and PNPLA4 as well. The fold change of each of them is also shown. P-values are representing the significance level of them as well. The structure of the table is exactly the same as [1] reported. Likewise, the precision of the values is the same as well. the fold change under 1 means that transcripts are expressed at lower levels in males. Notably, in geneIDs column, the transcripts and gene identifier assigned by StringTie are generated in order of the bundle of reads that are processed. So, in different runs different results may output.

geneNames	geneIDs	feature	id	fc	pval	qval
XIST	MSTRG.506	transcript	1729	0.00	7.04	1.61
.	MSTRG.506	transcript	1728	0.01	1.25	1.43
TSIX	MSTRG.505	transcript	1726	0.08	2.49	1.90
.	MSTRG.506	transcript	1727	0.04	3.71	2.13
.	MSTRG.590	transcript	1919	7.31	9.39	3.77
PNPLA4	MSTRG.56	transcript	203	0.46	9.86	3.77
.	MSTRG.506	transcript	1731	0.04	2.13	6.99
.	MSTRG.594	transcript	1923	9.18	3.50	1.00
.	MSTRG.510	transcript	1744	11.97	4.44	1.13

Table 2 Differentially expressed genes between sexes. The table below represents differentially expressed genes between two conditions being studied. Statistical measures of significance are also included in the table. Genes ID, fold change, p-value and q-value are represented as well. Chromosome X has 10 differentially expressed genes at the q-val of 0.05. gene-level p-values seem better than transcript-level. This means better statistical power to detect differential expression of genes rather than transcripts. Notably, in geneIDs column, the transcripts and gene identifier assigned by StringTie are generated in order of the bundle of reads that are processed. So, in different runs different results may output.

	feature	id	fc	pval	qval
1	gene	MSTRG.506	0.00	6.80	6.75
2	gene	MSTRG.56	0.54	3.66	1.81
3	gene	MSTRG.590	7.28	6.99	2.31
4	gene	MSTRG.505	0.08	1.16	2.89
5	gene	MSTRG.349	0.56	1.71	3.40
6	gene	MSTRG.594	9.14	3.21	5.31
7	gene	MSTRG.510	12.23	4.37	6.20
8	gene	MSTRG.492	0.65	1.98	2.45
9	gene	MSTRG.596	7.71	2.29	2.53
10	gene	MSTRG.788	1.74	3.57	3.55

6. Phase II : STAR vs HISAT Performances Analysis

In the second phase our main goal of study is examined: a comparison of the performance of HISAT2 vs STAR in the PNT pipeline. We analyze the running time scalability of the accuracy of the aligners. Furthermore, we will study of the impact of the different parameters and try to establish an optimal set.

Besides we specifically switch the mapping software used in the protocol with STAR and analyze performances. We re-do mapping with HISAT2 and STAR once again on a machine including higher RAM capacity. Furthermore, we do the mapping process on different sub-sets of the full data as well as full data to see if it affects the performance and time of aligning. It should be mentioned that all the sub-samples are mapped in both the machines to see how RAM capacity influences the mapping speed in both the aligners. In the end, parameter optimization of aligner with the worse results will be optimized to observe any improvement in the final output. Here the *Uniquely mapped reads* rates and *time* efficiency will be investigated.

The experiment is done on two different machines since the STAR aligner is memory intensive the available memory is expected to have an impact on the speed of mapping the reads to the reference genome. In order to evaluate them we use machines with 8GB and 64GB main memory and use data sets of increasing sizes.

6.1 Performances of Aligners: Alignment Accuracy

The accuracy of an aligner is one of the most important factors when choosing an aligner to map. A suitable aligner is the one with higher accuracy and lower running time. However, as to be expected, it is often a trade-off between the two. In our experiment, the parameters of *uniquely mapped reads* and *multimapped reads* rates define how accurate both aligners perform. The accuracy of an aligner is defined by a higher rate of uniquely mapped reads as well as lower multimapped reads.

6.1.1 Alignment Performance on Machine with 8GByte RAM

After running the protocol with reference file on the machine with 8GByte RAM, we replaced HISAT2 for STAR. Results show that STAR shows more uniquely mapped and less multi-mapped reads rates (see Figures 4 and 5). The percentage of reads that do not align is less than for HISAT2 (see Figure 6). The percentages of reads which are aligned exactly one time for STAR differs between 96.15 and 97.75 % While in HISAT2 bars roughly hit 80%. They differ between 71.85 and 80.77 %. The average rates of the uniquely aligned reads are 76.5 and 97.10 in HISAT2 and STAR respectively.

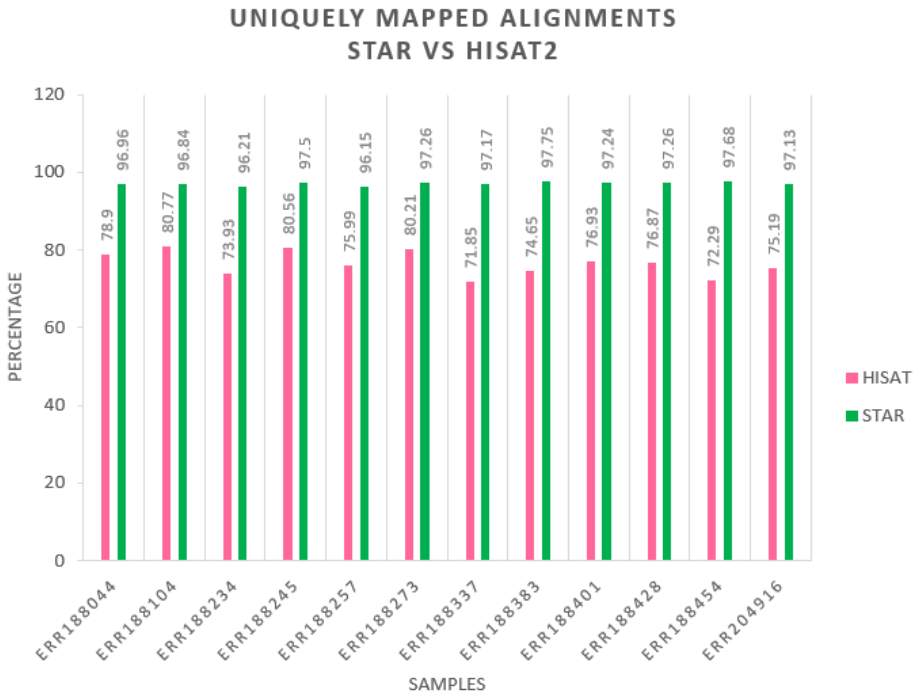


Figure 4 Uniquely mapped reads between STAR and HISAT2. STAR outperforms by longer bars showing improved percentages of uniquely aligned reads in down-sampled data of [1]. This criterion represents a read which mapped only once in the genome. In all the samples STAR outperforms than HISAT2 in the percentage rates of uniquely mapped reads.

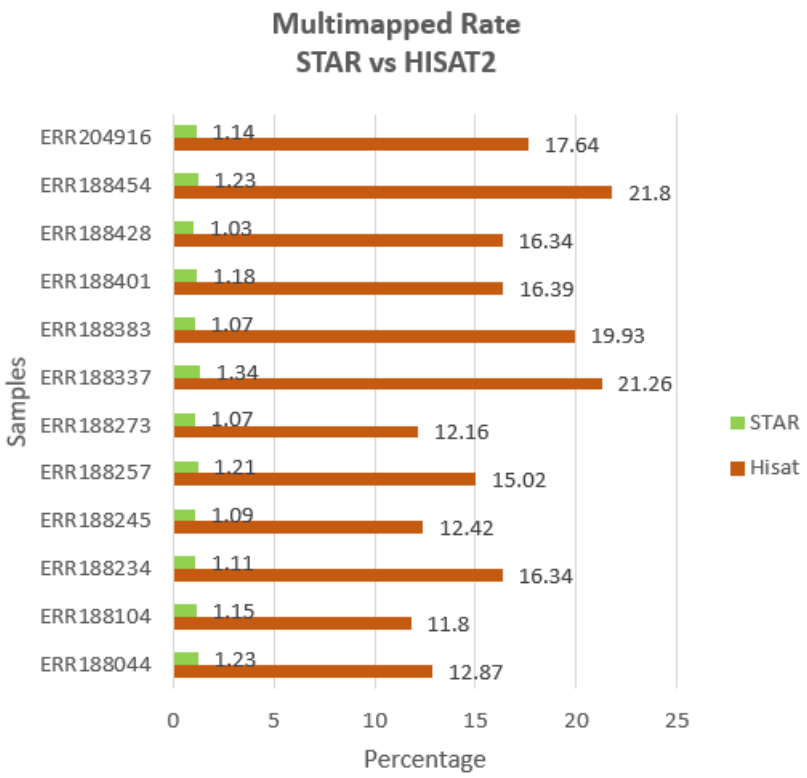


Figure 5 The percentages of multi-mapped reads using STAR and HISAT2 are depicted. An extra analysis of multi-map rates investigation is also done in our experiment with the use of reference file for both the aligners. The percentages of reads for STAR are much smaller than for HISAT2.

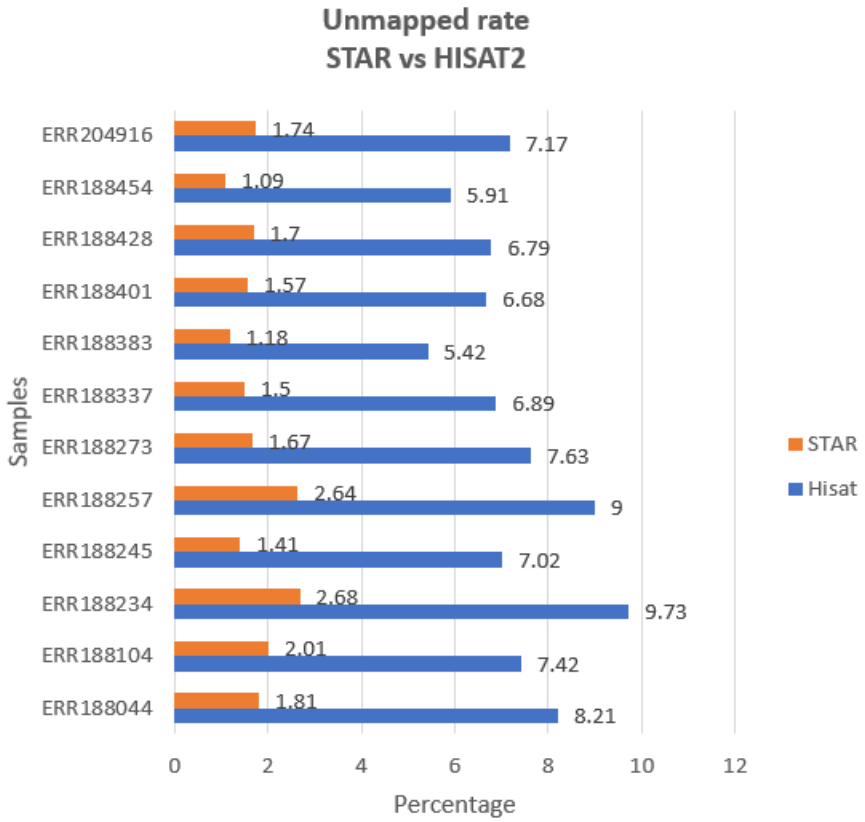


Figure 6 The percentage of reads which STAR and HISAT2 did not map to the genome. The extra analysis of unmapped are also depicted. Percentages of reads that fail to map to known exons are smaller for STAR than for HISAT2.

6.1.2 Alignment Performance on Machine with 64GByte RAM:

In the performance analysis, we observe lower percentages of uniquely mapped read rates for HISAT2 than those for STAR (Tables 3 and 4). Running HISAT2 on the full data represents the mapping rates approximately less than 4% of the whole data mapping on only chromosome X but this measure is above 12% in STAR. However, the results imply there are higher rates of mismatches included in the uniquely mapped reads rates.

Table 3 Uniquely mapped reads rates for HISAT2 on machine with 64 GB of RAM memory. The second column indicates 12 samples names tested during the experiment. They are from two groups of YRI and GBR. They are included both males and females equally in each group.

		0.1	0.2	0.4	0.8	1
1	ERR188044	3.98	3.98	3.97	3.96	3.94
2	ERR188104	4.24	4.25	4.24	4.24	4.24
3	ERR188234	4.26	4.26	4.25	4.24	4.23
4	ERR188245	3.72	3.72	3.71	3.7	3.7
5	ERR188257	3.54	3.54	3.53	3.54	3.53
6	ERR188273	4.17	4.18	4.17	4.15	4.16
7	ERR188337	3.41	3.41	3.4	3.39	3.39
8	ERR188383	3.92	3.91	3.91	3.89	3.89
9	ERR188401	3.85	3.85	3.86	3.85	3.84
10	ERR188428	3.62	3.62	3.62	3.61	3.61
11	ERR188454	3.79	3.77	3.76	3.75	3.75
12	ERR204916	4.08	4.08	4.08	4.06	4.06

Table 4 Uniquely mapped reads rates for STAR on machine with 64GB of RAM memory. The second column indicates 12 samples names tested during the experiment. They are from two groups of YRI and GBR. They are included both males and females equally in each group.

		0.1	0.2	0.4	0.8	1
1	ERR188044	12.46	12.46	12.48	12.47	12.47
2	ERR188104	12.98	12.99	13	13.01	13
3	ERR188234	13.99	13.99	14	14	14
4	ERR188245	12.16	12.15	12.16	12.17	12.17
5	ERR188257	11.73	11.72	11.72	11.72	11.72
6	ERR188273	12.81	12.82	12.84	12.85	12.85
7	ERR188337	11.65	11.65	11.65	11.64	11.64
8	ERR188383	13.22	13.22	13.23	13.23	13.23
9	ERR188401	12.62	12.6	12.62	12.61	12.61
10	ERR188428	11.88	11.9	11.91	11.91	11.91
11	ERR188454	12.78	12.78	12.76	12.74	12.75
12	ERR204916	13.25	13.23	13.24	13.23	13.24

6.1.3 Data Growth Effects on Aligners Accuracies

In the following Figures the effects of data growth are investigated in both the aligners. The Figures represent the aligners uniquely mapped reads rates are affected by increasing the samples sizes. In STAR aligner we observe improvements rather than HISAT2. It implies that the accuracy of HISAT2 very slightly decreases as the size of samples increases. In HISAT2 uniquely mapped reads rates drop down while for STAR they remain virtually stable. It means as data grow, the accuracy decreases very slightly for HISAT2 and remains stable for STAR.

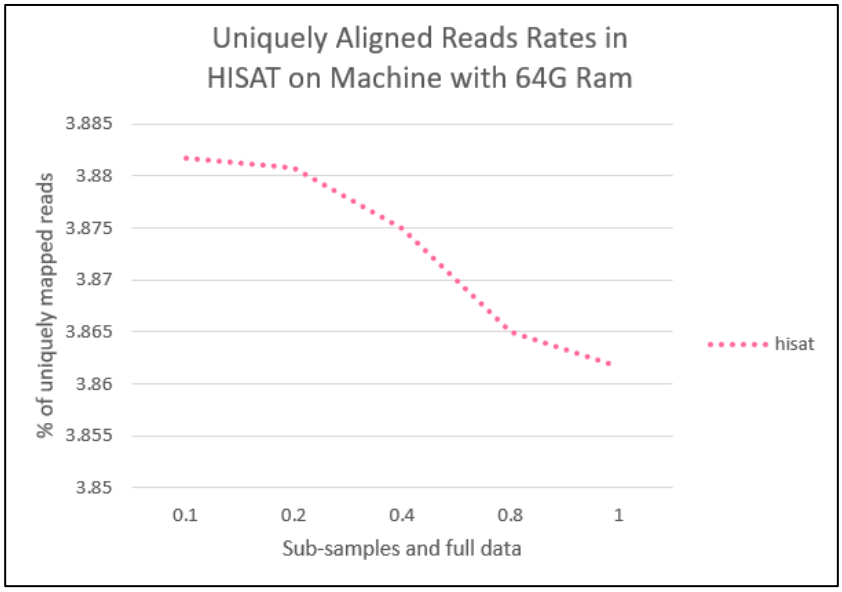


Figure 7 Depiction of data growth affects HISAT2 uniquely mapped rates. The percentages represent very small changes as the data increases. Without parameter optimization.

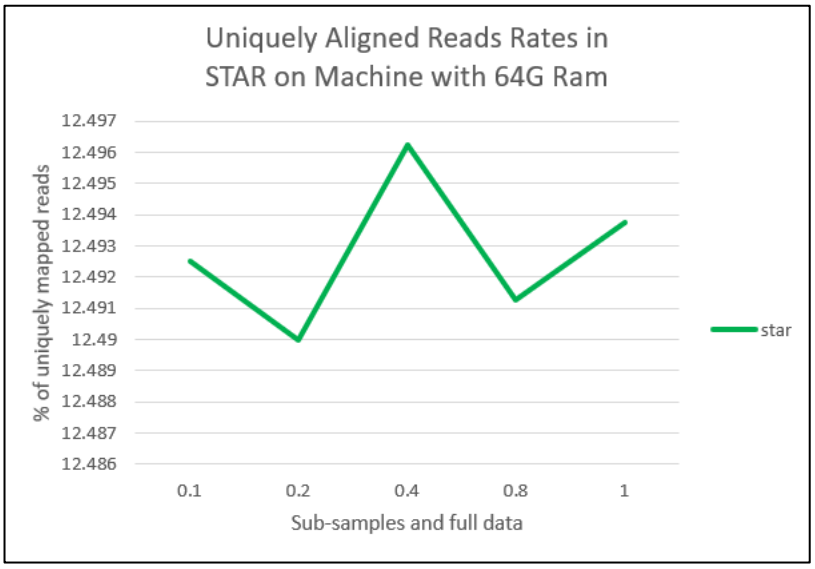


Figure 8 Depiction of percentages of uniquely mapped reads when using STAR. The percentages remain stable on bigger data sets. Without parameter optimization.

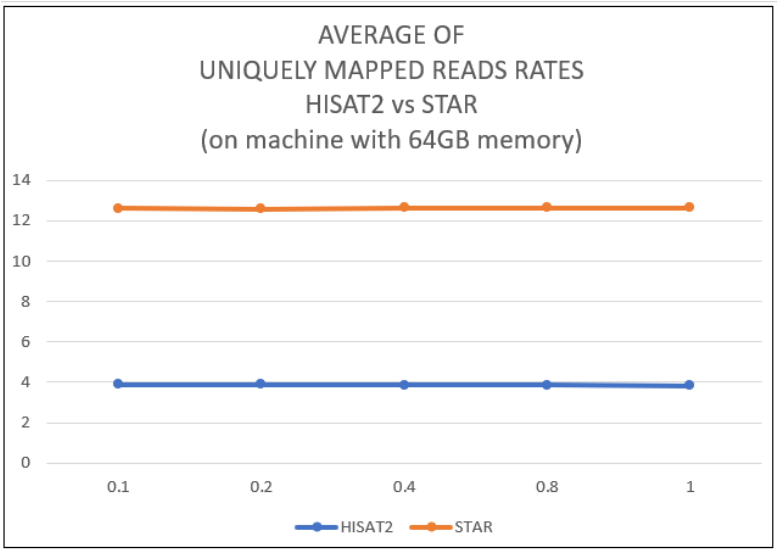


Figure 9 the average of uniquely mapped reads HISAT2 vs STAR on a machine with 64GB main memory.

6.2 Comparative Time Efficiency of HISAT2 vs STAR

In the following, time efficiency of each of the aligners in our experiment is tested. The same data (12 sub-samples) analyzed in PNT is examined here to observe if HISAT2 is still timely-efficient. The analysis id performed on two machines with 8 and 64GB of main memories (Figure 10).

Note that only some small differences between the execution time of HISAT2 and STAR is observed on machine with 8GB memory.

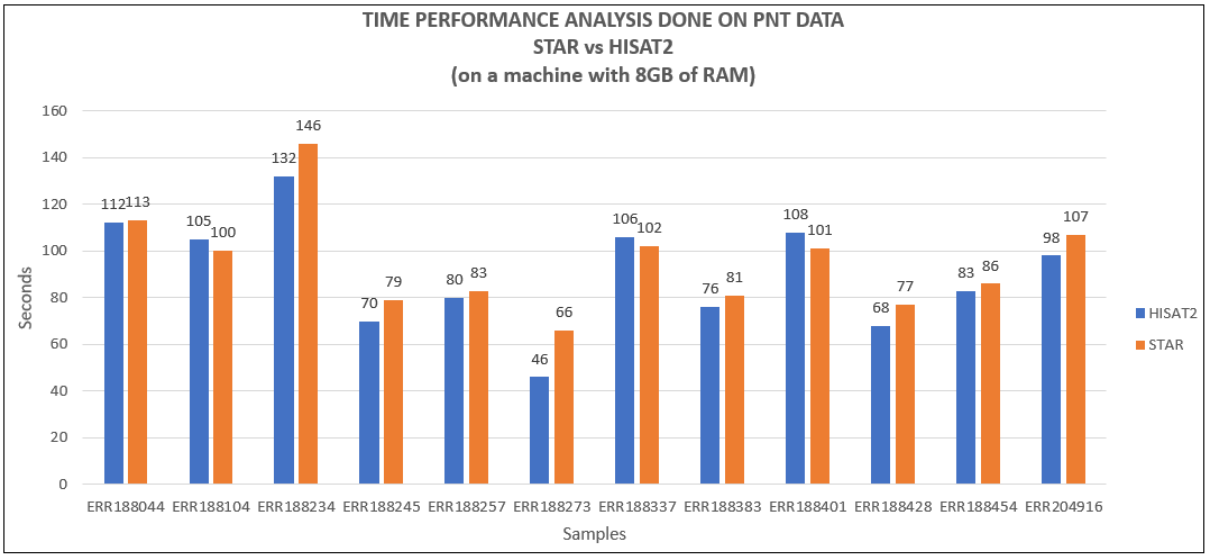


Figure 10 Comparison of time performance between STAR & HISAT2. HISAT2 shows a slightly better performance when running on a system with 8GByte of RAM. It should be mentioned that this Figure belongs to the data of PNT aligned with HISAT2 and STAR as well. Among the samples above, the sample of ERR188273 shows a bigger difference between two.

6.2.1 RunTime of STAR vs HISAT2 on Machine with 8GB RAM vs 64GB:

By running the STAR aligner on our machine with 8GB vs 64 GB RAM we find out that the STAR aligner tends to run faster when RAM memory capacity increases, whereas the availability of more memory has almost no impact on the runtime of HISAT2.

The results of mapping for each of the down-sampled data sets as well as full data sets show that although the STAR requires more time to map bigger data sets, we observe improvement in runtime of STAR when running on machine with 64GB of RAM memory rather than 8GB of memory (see Figure 11). Thus, we confirm that with a larger RAM memory, we can speed up STAR performances when aligning the human paired-end transcriptomic samples.

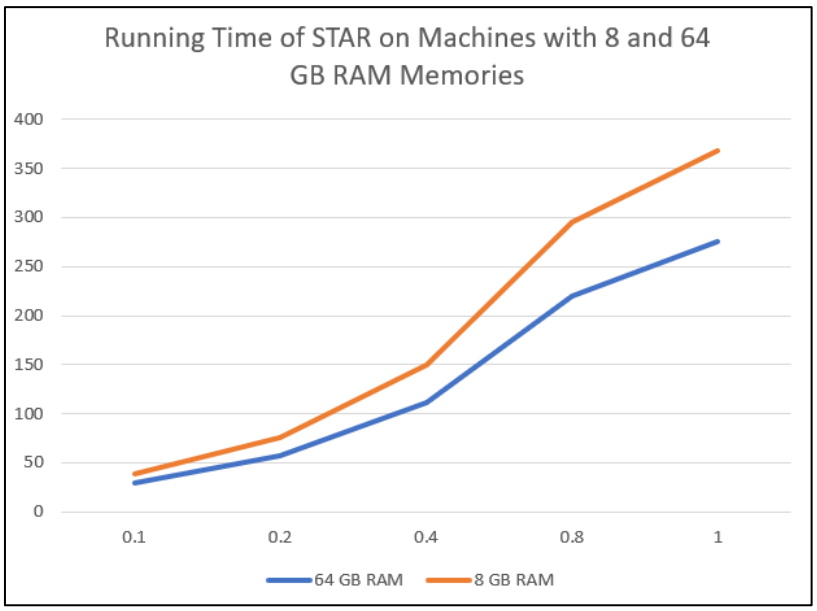


Figure 11 Influences of RAM capacities on aligning speed in STAR. Running STAR aligner on two different machines with 8 and 64GByte RAM memory is tested. The Figure above represents improvements in mapping speed when the RAM capacity is 64GByte.

We run the HISAT2 on our machine with 64GByte RAM to map the different sub-samples on the reference genome of chromosome X (see Table 5). The results of read mapping for each of the down-sampled data show that while data grows, run time increases almost linearly (see Figure 11). It means that HISAT2 is still far more timely-efficient than STAR even if the mapped data is large. It verifies that HISAT2 is a fast method to map reads on the genome. The indexing technique is the reference set used in HISAT2 plays a great role to get HISAT2 run faster.

Table 5 Runtime of HISAT2 vs STAR on machine with 64GB of RAM memory

		HISAT2					STAR				
SAMPLES		0.1	0.2	0.4	0.8	1	0.1	0.2	0.4	0.8	1
YRI-Male	ERR188044	0.41	1.19	2.37	5.16	6.35	34.53	69.39	140.1	273.34	342.03
YRI-Male	ERR188104	0.38	1.14	2.28	5.01	6.15	33	67.32	127.2	248.31	312.52
YRI-Female	ERR188234	1.03	1.23	2.5	5.4	7.06	35.33	71.12	141.3	282.12	351.34
GBR-Female	ERR188245	0.29	0.58	1.57	3.51	4.53	28.56	51.24	97.01	194.33	246.13
GBR-Male	ERR188257	0.30	1.01	2	4	5.08	30.03	52.39	99.45	198.39	246.20
YRI-Female	ERR188273	1.04	0.32	1.05	2.06	2.43	15.45	27.41	50.32	101.19	126.57
GBR-Female	ERR188337	0.38	1.19	2.37	5.15	6.29	32.38	63.05	121.5	241.05	301.04
GBR-Male	ERR188383	1.22	0.58	1.54	3.48	4.47	26.17	51.52	98.19	188.44	237.33
GBR-Male	ERR188401	1.22	1.17	2.36	5.13	6.34	35.17	69.37	140.2	272.50	340.55
GBR-Female	ERR188428	0.56	1.19	2.37	5.14	6.27	28.57	56.09	114.6	225.25	278.47
YRI-Male	ERR188454	1.26	1.30	2.02	4.02	5.09	32.09	55.58	104.6	209.15	259.37
YRI-Female	ERR204916	1.11	1.01	2.06	4.07	5.20	31.27	54.58	103.1	204.33	259.36

7. Parameter Optimization in STAR and HISAT2

As Dobin et al. [10] showed that utilizing different parameters depending on the type of experiments result in improvement in aligning. Similarly, we tune the parameters of STAR to observe if we can obtain any improvements. Parameters tested in the current experiment are selected based on their effects on the mapping performance. In STAR aligner the effects of the following parameters are studied:

- The most influential parameter on data using STAR is --*outFilterMismatchNmax*. It controls the maximum number of mismatches allowed per alignment. Since it depends on data structure to optimize parameters, we need to change the default value to fit the situation of data sets.
- Another important parameter for STAR mapping optimization is *sjdbOverhang*. It influences sensitivity of junction detection. This value (default:100) is generally used for the reads longer than 50 bp. In our experiment, we strictly change the default value to readlength-1 which is the 74 bp in our study (readlength is 75bp).
- *SeedsearchstartLmax* is one of the parameters to tune the mapping sensitivity. As our data set is modest (not very short or very long lengths of reads analysis). The parameter determines the length of blocks a read can be split in (the parameters list tested in our experiments are in the Appendix B).

7.1 Parameter optimizations

As already mentioned in the introduction section, one of the goals is by considering the financial limitation of the analysis a researcher gets the optimal results out of the experiment. In our experiment, tuning the parameters is recommended to get the maximum accuracy and speed from the mapper's performance. In the following, the parameters of HISAT2 and STAR is tuned to achieve any improvement in mapping process. Notable, there is always a trade-off between accuracy, sensitivity and precision. On one hand the rates might be improved by tuning the parameters, on the other hand it may negatively affect other rates such as multimapped reads rates. Additionally, in human sample analysis, a modest mismatch rate between samples is natural due to naturally occurring SNPs. However, the more mismatches are allowed, the more spurious the alignments will become. So, by estimation of the rate of mismatches on the sequence of the length of the read we may find the optimal choice for mismatch rate.

7.1.1 Parameter optimization in STAR

Parameter optimization is performed for the 12 samples (samples which Pertea et al. studied on) of Human data. Adjusting parameters are done when mapping paired-end reads using STAR aligner. (see Appendix B). The results are illustrated in Figure 12. The chart shows that uniquely mapped alignments rates are increased in comparison with those without changes in

parameter values. All the rates are well higher than 98 percent while without parameter optimization, they are less than 98 percent. As an example of the parameter that we adjust in our experiment, *outfiltermismatchNmax* controls the number of Mismatches. The mismatches include RNA-editing, SNPs and sequencing errors. It is notable that, beside these values changes, the no-mapped rates are all decreased as well. Most of them decreased to 0.01 while a couple of samples results show even the rate of 0. Furthermore, to increase overall mapping sensitivity we applied *seedsearchstartlmax* to limit the length of reads to map on the reference genome. This leads to define the number of reads to consider as *too short* reads. The aim of parameter optimization is to tailor the reads as best as possible based on the sample's specifications. Otherwise the results are skewed leading to unreal information of transcripts. Finally, after applying parameter optimization on sub-samples, we tweak the parameters of HISAT2 and STAR for the full data.

The parameters are selected based on their influences on the outputs. Although rates in STAR hit higher points than HISAT2, we should not rely on these rates without optimization of parameters. By looking at the results we see mismatch rates are not trivial in STAR. So, there should be reduced to make sense. However, it is not true to get a mismatch rate of zero. Since we have at least some SNPs affecting the mismatch rates. As our analysis of the STAR and HISAT2 aligners maps only on chromosome X and the volume of data is gradually increased in the sample sorts, so we expect fewer but not no-mismatches. Keeping this in mind, the values with higher rates of uniquely mapped as well as lower mismatch rates are considered as the best option.

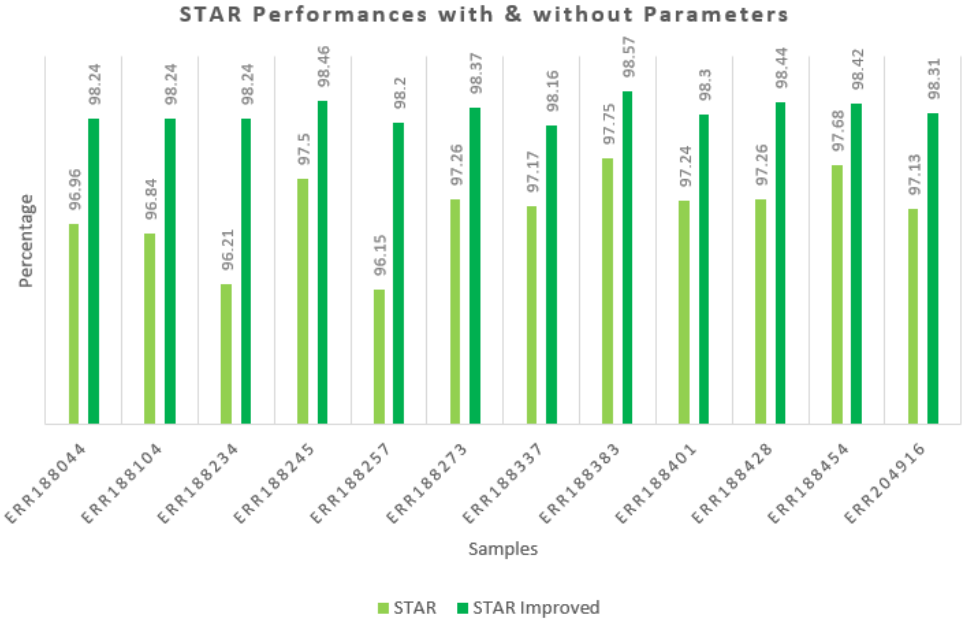


Figure 12 STAR with/without parameter optimization" on machine with 8GByte RAM done on PNT data

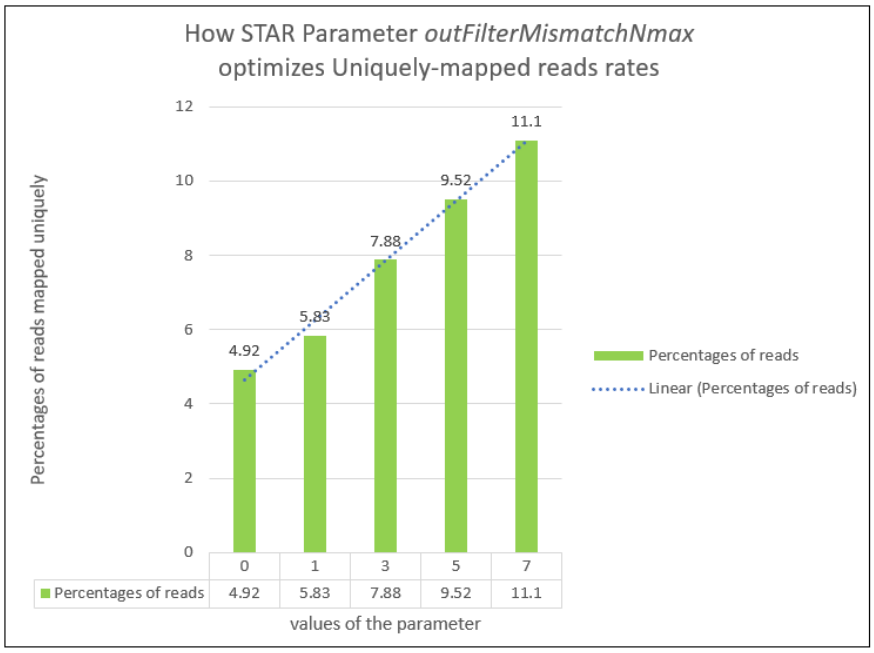


Figure 13 the test is done on a very small portion of the reads to show how the parameter affects the rate

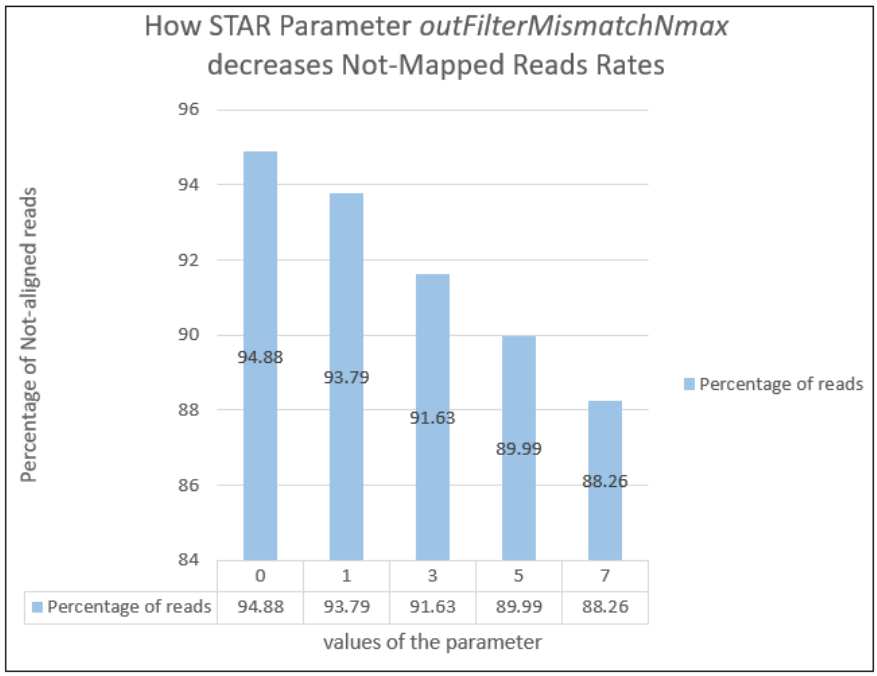
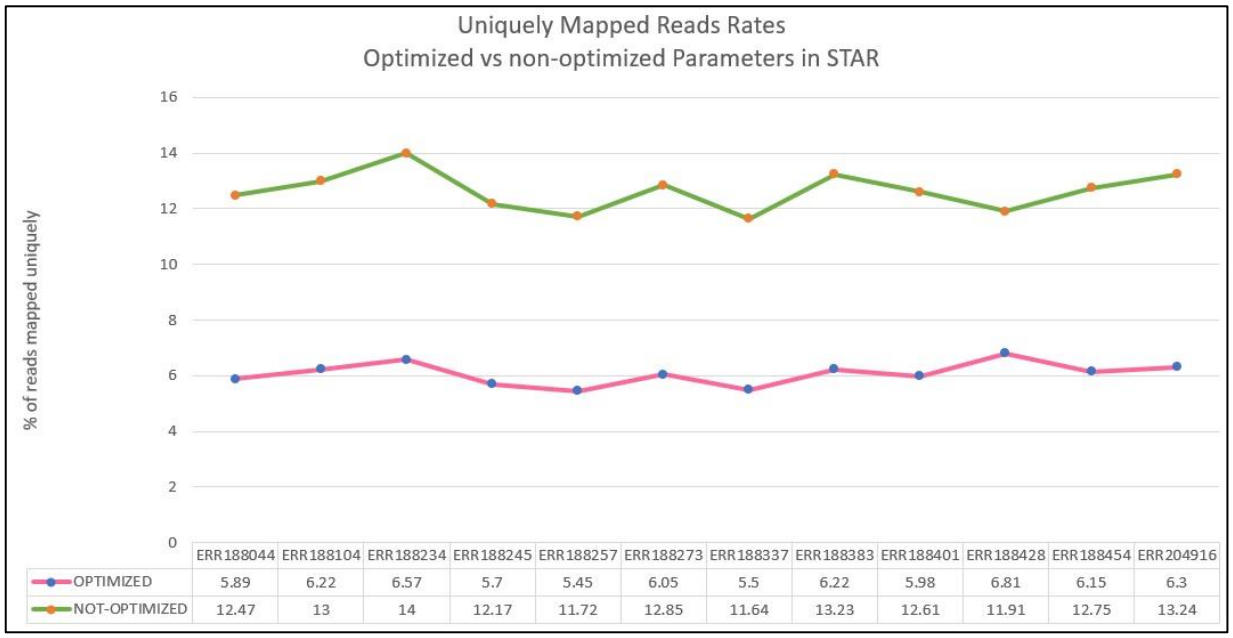


Figure 14 the test is done on a very small portion of the reads to show how the parameter affects the rate

By considering two measures of mismatch rates and uniquely mapped reads rate, tuning the parameters is performed on the full samples. The only parameter affecting the results is *outFilterMismatchNmax*. By decreasing the default number of the parameter from 10 to 1, we see that the mismatch rats go significantly down. It is also important to know that result of parameter optimization is data structure dependent. As an example, the paired-end read optimization is different than single end reads. similarly, length of the read also has a strong role in selection of the parameters to tweak (Appendix B). The two figures below represent how the parameter leads to decrease in both graphs of mapped and mismatch reads rates.



	ERR188044	ERR188104	ERR188234	ERR188245	ERR188257	ERR188273	ERR188337	ERR188383	ERR188401	ERR188428	ERR188454	ERR204916
OPTIMIZED	5.89	6.22	6.57	5.7	5.45	6.05	5.5	6.22	5.98	6.81	6.15	6.3
NOT-OPTIMIZED	12.47	13	14	12.17	11.72	12.85	11.64	13.23	12.61	11.91	12.75	13.24

Figure 15 Rates differ between 11.50 to 14 before tweaking the parameters. While, after performing the STAR with new values for parameters, they are decreased a lot. Rates are now between 5.5 to about 6.80. There is a huge gap between two sets as shown. One of the reasons might be that there is a huge room for improvement on the data. It should also be considered that parameter tuning must be done cautiously. In some cases, as the percentage of mapped reads increases, on the other hand the rate of multimapped reads increases as well.

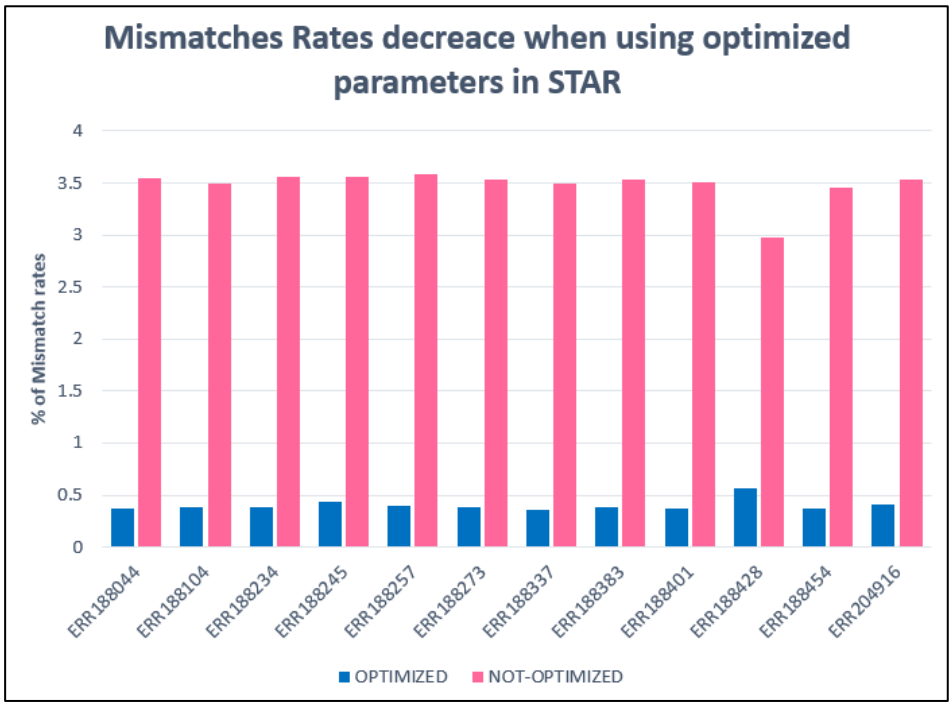


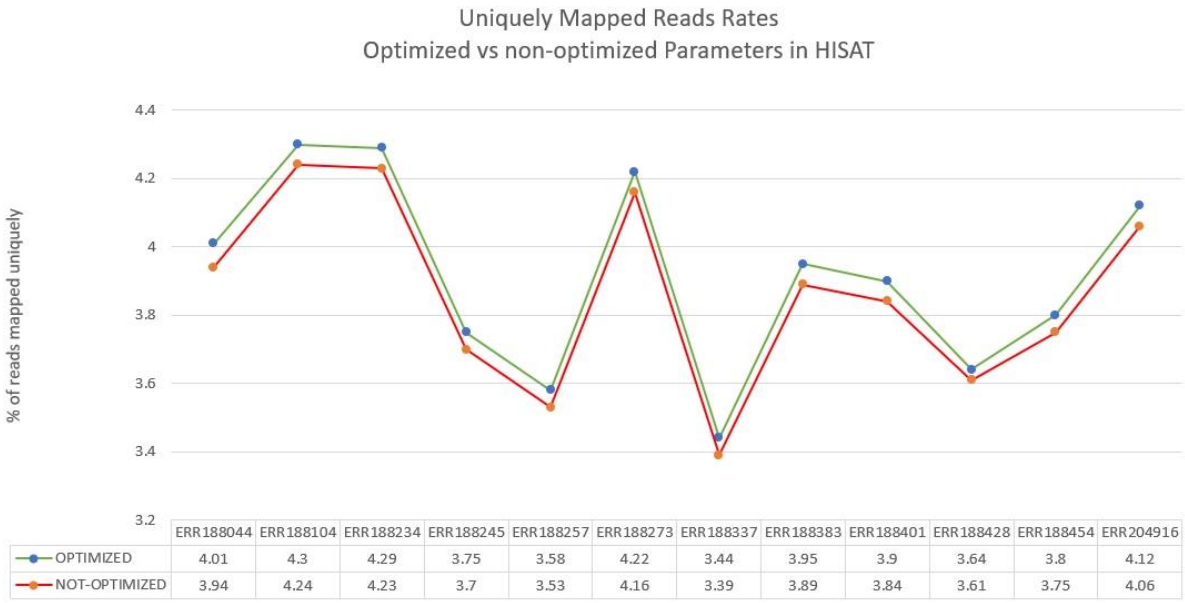
Figure 16 Comparison of mismatch rates before and after optimization of parameters in STAR. Mismatches are hugely decreased when parameter of -outFilterMismatchNmax is set to 1 (default=10)

7.1.2 Parameter optimization in HISAT2

The HISAT2 default parameters are also switched with different values. However, it is notable that default values are the suitable values as the authors of both aligners have mentioned in the manuals. The Figure below represents a trivial change in the values of uniquely mapped reads rates. For example, using default parameters in HISAT2, when aligning the sample ERR188044, 3.94% of the reads are uniquely mapped to the reference genome. However, an improvement is observed when we optimize the parameters in HISAT2. It implies how optimization of parameters affect the mapping results.

HISAT2 parameters affecting *uniquely mapped reads rates* are two parameters of *–pen-noncansplice* & *--pen-cansplice* among the variety of parameters we test. These parameters set the penalties for each pair of canonical and non-canonical splice sites (see Appendix B). Generally, it is observed that practically all introns contain regions of two highly conserved dinucleotides, GT at donor and AG at acceptor regions. Canonical splice sites are considered GT, AG and non-canonical are non-GT, AG splice sites. We consider penalties in case non-canonical splice sites are observed in the mapping process [19]. By default, the non-canonical splice sites value is 3. By increasing the penalty of this option, we observe that the uniquely mapped reads rates are all increased in all the sample. We increase the accuracy of samples by

strictly setting the non-cansplice sites option. For instance, the first sample as ERR188044 is optimized from 3.94 to 4.01 % of full data. Notably, in some versions of HISAT we see that non-cansplice value is even 12.



	ERR188044	ERR188104	ERR188234	ERR188245	ERR188257	ERR188273	ERR188337	ERR188383	ERR188401	ERR188428	ERR188454	ERR204916
OPTIMIZED	4.01	4.3	4.29	3.75	3.58	4.22	3.44	3.95	3.9	3.64	3.8	4.12
NOT-OPTIMIZED	3.94	4.24	4.23	3.7	3.53	4.16	3.39	3.89	3.84	3.61	3.75	4.06

Figure 17 HISAT2 uniquely mapped reads rates is improved when parameters are optimized

8. Results

In PNT [1], the experiment is done on the sub-samples of full-data. Pertea et al. [1] confirms that the process of mapping is extremely fast by HISAT2. However, they have also tested the full-data set of the same sub-samples and mapped the reads with HISAT2. In their experiment multi-mapped reads can severely affect the expression levels of genes implying that a significant percentage of reads are mapped to more than one place in the reference genome. The result of their analysis represents that multi-mapped reads rates in different samples are changing between 20.9 and 33.2% except one sample that is 57.7%. The rates of uniquely aligned reads are also between 65.2 and 75.8% except one rate showing 39.7% in 12 full human samples. Similarly, in the first phase of our experiment, we perform the same analysis as [1] and observe the same results (see Tables 1&2) as Pertea et al. [1] observed (the same genes and transcripts names with p-value, q-value). In the second phase of our experiment, we test data with different sizes. In below, results are investigated based on two factors of uniquely mapped reads rates as well as the time both aligners need to map.

9. CPU & Memory Usage in STAR & HISAT2

Running a HISAT2 command in terminal requires almost the same CPU usage as STAR (using *top* command to check) when using optimized parameters. However, memory usage differs a lot as expected. Since STAR consumes more memory capacity to run a command. The memory usage in STAR is 6.4. although HISAT2 requires 0.5 percent of memory. STAR needs almost the same percent of CPU as HISAT2. Based on what we observe, this rate differs between 700 to 800 for STAR and 680 to less than 800 for HISAT2 (only a trivial difference, it means HISAT2 performs a command with a bit less CPU usage). Similarly, when the parameters are not optimized, the required CPU usage by STAR is the same as when it is optimized. In HISAT however there is some trivial changes when we use the parameters with default values. It differs between 700 and 800. Parameter optimization has no effects on memory usage when using HISAT. Although in STAR it increases by 6.5.

It should also be mentioned that *-p* parameter with the value of 8 is included in all the HISAT2 commands in this work. Using multi-threading option is also a help for the speed of the process. However, in this case the order of the runs is not deterministic anymore.

10. Discussion

We have run the "New Tuxedo" pipeline [1] in the first phase of our experiment and switched HISAT2 with STAR to evaluate their performances in the second phase. We focus on HISAT2 performance by running HISAT2 and STAR on the full and sub-sampled data. The results of running the pipeline in the first phase are the same as Pertea's [1]. In the second phase we specifically focus on aligners performances as well as tuning their parameters.

We used STAR since we find it suitable to switch with HISAT2. Soft-clipping technique is one of the advantages of STAR aligner. Since it leads to higher accuracy of aligning. Furthermore, STAR analyses draft genes leading to increase the accuracy. Additionally, it has verified its ability in mapping human data in many other experiments already done (refer to introduction for more details). Hence, it is investigated in our experiment if STAR outperforms HISAT2 as expected.

We observe improvements in mapping performance when using STAR, however the outcome shows HISAT2 is far faster than STAR in running the human (RNA-seq paired-end reads, 75bp) samples used in this work. One reason is that HISAT2 is not resource demanding. The other would be indexing technique HISAT2 applies.

Here, we try tuning the parameters of both HISAT2 and STAR to remove unreal large uniquely mapped reads rates of STAR as well as improvement in HISAT2 mapping performance. It is

obvious that larger rates do not necessarily mean that they are aligned correctly. The uniquely mapped reads rate is the common objective measure in the both aligners. However, we consider the mismatch rates to decrease in STAR as well.

Considering that the performance is tightly dependent on the data and requirements, the results imply that STAR represents an improved mapping accuracy rather than HISAT2. Since it still shows higher rates of mapped reads. However, we can take more factors to be changed in optimization process. For instance, depending on the research goals, multi-mapped reads can also be decreased as well. However, in our work this rate was moderate. Mainly, there are factors influencing the STAR performance such as the technique used in STAR to map the unaligned reads. STAR performs read-clipping while trying to re-align unmapped reads. Soft-clipping technique improves the robustness of the STAR aligner as well as maximizing the alignment score. If we decrease the mismatch penalty, we may end up with less soft clipping. Additionally, in some cases, the genomes contain gap or ambiguity symbols leading to poor or low mapping rates. However, using STAR in such cases seems to be a good option. Since it applies soft-clipping and increases the mapping rate. Hence, it seems STAR suits much better to low-quality genomes.

11. Conclusion

We perform our experiment in two different phases with two sets of data. In the first phase as PNT [1] we implement the pipeline on 12 down-sampled data. The 5% of data which map to the chromosome X are analyzed. In the second phase however, we do the experiment by 20,40,80 and 100% of full data for each of the samples. Notably, in parameter optimization only the full samples are optimized. The process is run on two different machines. When performing the first phase, the machine with 8GByte of RAM (core i7 (7th Gen, Intel), windows 10, 64 bits) is used. However, in the second phase both the machines with 8 and 64GBytes of RAM (Intel Core i7-5820K 3.3 GHz 12 cores, Ubuntu 16.04 LTS 64-bit) are employed.

In the current experiment we have analyzed different aspects of the experiments. We have investigated data growth effects, accuracy of the aligners, the influence of RAM capacity, the run time of the aligning process, the effects of parameter optimization on aligners performances for both HISAT2 and STAR aligners and finally CPU and memory usages of both. We observe that RAM capacity plays an important role when applying STAR to map to the reference genome, it is very trivial in HISAT2 though. Another observation is that as data grows STAR puts an improved performance on view by specifically considering the uniquely mapped reads rates. Accuracy in STAR depends on three factors in our experiment. We have observed the roles of RAM capacity, the sizes of samples and finally parameter optimization

on transcriptomic paired-end human samples with different sizes. Furthermore, we have also noticed the parameter optimization affect the mappers performance when aligning with STAR. It is the same for HISAT2, it is very trivial though. For example, STAR maps the full sample of ERR188044 with 5.89 % of uniquely mapped reads rates with default parameters while it is 12.47 when we optimize the parameters. These values are respectively 3.94 and 4.01 using HISAT2. Finally, CPU and memory usages of both the aligners are investigated showing that CPU usages are the same for both however STAR consumes more memory usage to map.

Depending on the data structure optimization of parameters is done. it is very important to not skew the results. There are some trade-offs that should be considered before tuning the parameters. Sensitivity and precision of the aligners should both be considered as well. The aim is to get reliable and the optimal or near optimal results representing and conserving the biological features of the samples.

12. Future Work

Sequencing technology is growing fast. Similarly, the need for parallel improvement of the mappers is growing as well. Hence, different new strategies for aligning the reads is also required. As [2] implies, the same framework of STAR can be used as a basic guidance. Additionally, as an example in presence of pseudogenes HISAT maps a portion of the reads in these regions on the reference genome. To build well-developed mappers, the researchers can specifically strengthen the well-performing aligners algorithms by improving the weaknesses for aligning.

References

[1] Pertea M, Kim D, Pertea GM, Leek JT, Salzberg SL, Transcript-level expression analysis of RNA-seq experiments with HISAT2, StringTie and Ballgown. (2016), doi: 10.1038/nprot.2016.095.

[2] Dobin A, Davis CA, Schlesinger F, Drenkow J, Zaleski C, Jha S, Batut P, Chaisson M, Gingeras TR, STAR: ultrafast universal RNA-seq aligner, (2013), doi:10.1093/bioinformatics/bts635.

[3] Bar-Joseph Z, Gitter A, Simon I, Studying and modelling dynamic biological processes using time-series gene expression data, Nat Rev Genet, 13 (2012), pp. 552-564, 10.1038/nrg3244.

[4] Oh S, Song S, Grabowski G, Zhao H, Noonan JP. Time series expression analyses using RNA-seq: a statistical approach, BioMed Res Int (2013), pp. 1-16, 10.1155/2013/203681.

[5] Conesa A, Madrigal P, Tarazona S, Gomez-Cabrero D, Cervera A, McPherson A, Szcześniak MW, Gaffney DJ, Elo LL, Zhang X, Mortazavi A, A survey of best practices for RNA-seq data analysis, (2016), doi: 10.1186/s13059-016-0881-8.

[6] Mikkelsen TS1, Wakefield MJ, Aken B, Amemiya CT, Chang JL, Duke S, Garber M, Gentles AJ, Goodstadt L, Heger A, Jurka J, Kamal M, Mauceli E, Searle SM, Sharpe T, Baker ML, Batzer MA, Benos PV, Belov K, Clamp M, Cook A, Cuff J, Das R, Davidow L, Deakin JE, Fazzari MJ, Glass JL, Grabherr M, Greally JM, Gu W, Hore TA, Huttley GA, Kleber M, Jirtle RL, Koina E, Lee JT, Mahony S, Marra MA, Miller RD, Nicholls RD, Oda M, Papenfuss AT, Parra ZE, Pollock DD, Ray DA, Schein JE, Speed TP, Thompson K, VandeBerg JL, Wade CM, Walker JA, Waters PD, Webber C, Weidman JR, Xie X, Zody MC; Broad Institute Genome Sequencing Platform; Broad Institute Whole Genome Assembly Team, Graves JA, Ponting CP, Breen M, Samollow PB, Lander ES, Lindblad-Toh K. Genome of the marsupial Monodelphis domestica, reveals innovation in non-coding sequences. Nature 447, 167–177, (2007).

[7] Garber M, Grabherr MG, Guttman M, Trapnell C, Computational methods for transcriptome annotation and quantification using RNA-seq, (2011), doi:10.1038/nmeth.1613.

[8] Pertea M, Pertea GM, Antonescu CM, Chang TC, Mendell JT, Salzberg SL. StringTie enables improved reconstruction of a transcriptome from RNA-seq reads, (2015), doi:10.1038/nbt.3122.

[9] Baruzzo G, Hayer KE, Kim EJ, Di Camillo B, FitzGerald GA, Grant GR, Simulation-based comprehensive benchmarking of RNA-seq aligners, (2016), doi:10.1038/nmeth.4106.

[10] Dobin A, Gingeras TR, Optimizing RNA-Seq Mapping with STAR. In: Carugo O., Eisenhaber F. (eds) Data Mining Techniques for the Life Sciences. Methods in Molecular Biology, vol 1415. Humana Press, New York, NY, (2016), doi.org/10.1007/978-1-4939-3572-7_13.

[11] Smolka M, Rescheneder P, Schatz MC, von Haeseler A, Sedlazeck FJ, Teaser: Individualized benchmarking and optimization of read mapping results for NGS data, (2015), DOI 10.1186/s13059-015-0803-1.

[12] Ching T, Huang S, Garmire LX, Power analysis and sample size estimation for RNA-Seq differential expression, (2014), doi: 10.1261/rna.046011.114.

[13] Dobin A, STAR manual 2.4.0.1, (2014).
[14] Cieślik M, Chinnaiyan AM, Cancer transcriptome profiling at the juncture of clinical translation, (2018), doi:10.1038/nrg.2017.96.

[15] Monger C, Motheramgari K, McSharry J, Barron N, Clarke C, A Bioinformatics Pipeline for the Identification of CHO Cell Differential Gene Expression from RNA-Seq Data, (2017), Part of the Methods in Molecular Biology book series (MIMB, volume 1603).

[16] Cheng Z, Sun, Niu X, Shang Y, Ruan J, Chen Z, Gao S, Zhang T, Gene expression profiling reveals U1 snRNA regulates cancer gene expression, (2017), doi: 10.18632/oncotarget.22842.

[17] Kim D, Langmead B, Salzberg SL, HISAT2: a fast-spliced aligner with low memory requirements, (2015), doi: 10.1038/nmeth.3317.

[18] Zhao S, Fung-Leung WP, Bittner A, Ngo K, Liu X, Comparison of RNA-Seq and Microarray in Transcriptome Profiling of Activated T Cells, (2014), doi: 10.1371/journal.pone.0078644.

[19] Burset M, Seledtsov IA, Solovyev VV, Analysis of canonical and non-canonical splice sites in mammalian genomes, (2000), Nucleic Acids Res. 2000 Nov 1; 28(21): 4364–4375.

[20] Pai Zhang Ling-Hong Hung Wes Lloyd Ka Yee Yeung, Hot-starting software containers for STAR aligner, (2018), doi.org/10.1093/gigascience/giy092. Reference for STAR overview

Appendix A

In the Appendix A we include the codes we run in the Ubuntu script. In the first phase of our experiment we use the same option as PNT did. The following is the list of topics run and explained in appendix A:

1. Mapping the reads
2. Sort and convert the SAM files to BAM
3. Genes and transcripts assembling and quantification
4. Comparison of StringTie's outputs with GTF file
5. Creation of table of counts and investigate transcript abundance
6. Differential expression analysis
7. Data visualization
8. Mapping with STAR:
9. STAR parameters tested on Pertea et al.'s data

1) Mapping the reads

Here we map the reads for each sample to the reference genome. The codes below in the Table map the reads of the samples to the reference genome. The options used here are explained in the help section of the software.

-1 <m1>	Files with #1 mates, paired with files in <m2>
-2 <m2>	Files with #2 mates, paired with files in <m1>
-p	the number of processors used when running the commands (in help of HISAT2: number of alignment threads to launch)
--dta	Reports alignments tailored for transcript assemblers
-x	Index filename prefix
-S	skip the first <int> reads/pairs in the input

Here is only one example of the codes run in HISAT2:

```
$ HISAT22 -p 8 --dta -x chrX_data/indexes/chrX_tran -1
chrX_data/samples/ERR188044_chrX_1.fastq.gz -2
chrX_data/samples/ERR188044_chrX_2.fastq.gz -S
ERR188044_chrX.sam
```

2) Sort and convert the SAM files to BAM

For Samtools older than 1.3 the following two steps are executed (The Samtools version I used: older than 1.3):
Firstly, the SAM files are converted to binary BAM files:

View	SAM<->BAM conversion
-bs	Import SAM to BAM when @SQ lines are present in the header

Here is one example of codes run by samtools:

```
$ samtools view -bS ERR188044_chrX.sam >
ERR188044_chrX_unsorted.bam
```

Now the BAM files should be sorted:

Sort	sort alignment file
-@	number of CPUs to be used

Here is one example of the code run by samtools:

```
$ samtools sort -@ 8 ERR188044_chrX_unsorted.bam
ERR188044_chrX
```

If your Samtools version is not older than 1.3, you should type the following commands to sort and convert in one step:

```
$ samtools sort -@ 8 -o ERR188044_chrX.bam
ERR188044_chrX.sam
```

3) Genes and transcripts assembling and quantification

Transcripts assembly

As we have already in the overview mentioned, we do need to assemble transcripts. Since they might be covered partially. Like the HISAT22 package, there are options available to utilize:

-G	reference annotation to use for guiding the assembly process (GTF/GFF3)
-p	number of threads (CPUs) to use (default: 1)
-l	name prefix for output transcripts

Here is one example of the code run by StringTie:

```
$ stringtie -p 8 -G chrX_data/genes/chrX.gtf -o
ERR188044_chrX.gtf –l ERR188044 ERR188044_chrX.bam
```

Transcript merging

we merge the gene and transcript models using the StringTie merge operation by the following:

```
$ stringtie --merge -p 8 -G chrX_data/genes/chrX.gtf -o
stringtie_merged.gtf chrX_data/mergelist.txt
```

If you run the stringtie from a different directory, then the clear path should be inserted in the function above.

4) Comparison of StringTie's outputs with GTF file

If you wish to compare the outputs of StringTie package with the reference annotation, you can use the gffcompare function:

-r	reference annotation file (GTF/GFF)
-G	it tells gffcompare to compare all transcripts in the input transcripts .gtf file
-o	output prefix

Here is one example of the code run by gffcompare:

```
$ gffcompare –r chrX_data/genes/chrX.gtf –G –o merged
stringtie_merged.gtf
```

The reason to this comparison is:
"This allows the user to quickly check how the predicted transcripts relate to an annotation file" [1].

5) Creation of table of counts and investigate transcript abundance

-B	enable output of Ballgown table files which will be created in the same directory as the output GTF (requires -G, -o recommended)
-e	-e only estimate the abundance of given reference transcripts (requires -G)

Here is an example of codes run by StringTie:

```
$ stringtie –e –B -p 8 -G stringtie_merged.gtf -o
ballgown/ERR188044/ERR188044_chrX.gtf ERR188044_chrX.bam
```

6) Differential expression analysis

The packages used in R are Ballgown, RSkittleBrewer, genefilter, dplyr and devtools. Download them via the Bioconductor website. After installation all should be called to execute.

Needed R packages

After running R, and installation of packages, they load by library command:

```
> library(ballgown)
> library(RSkittleBrewer)
> library(genefilter)
> library(dplyr)
> library(devtools)
```

Phenotype data file

```
> pheno_data = read.csv("geuvadis_phenodata.csv")
```

StringTie's expression estimation reading

```
> bg_chrX = ballgown(dataDir = "ballgown", samplePattern =
"ERR",pData=pheno_data)
```

Genes with low abundance are removed
Here we filter out all the genes that their abundance values are low.

```
> bg_chrX_filt = subset(bg_chrX,"rowVars(texpr(bg_chrX))
>1",genomesubset=TRUE)
```

Differentially expressed transcripts

The following will test each of the gene feature for differential expression.

If n<4 per group, another package called limma should be used.

```
> results_transcripts =
stattest(bg_chrX_filt,feature="transcript",
covariate="sex", adjustvars =c("population"), getFC=TRUE,
meas="FPKM")
```

Differentially expressed genes

```
> results_genes = stattest(bg_chrX_filt,
feature="gene",covariate="sex", adjustvars =
c("population"), getFC=TRUE,meas="FPKM")
```

Gene name, Gene ID feature in the table

```
>
results_transcripts=data.frame(geneNames=ballgown::geneName
s (bg_chrX_filt),geneIDs=ballgown::geneIDs(bg_chrX_filt),
results_transcripts)
```

From the small to large p-value

We can sort the results based on what we determine to be shown.

```
> results_transcripts = arrange(results_transcripts,pval)
> results_genes = arrange(results_genes,pval)
```

Output file creation

```
> write.csv(results_transcripts,
"chrX_transcript_results.csv",row.names=FALSE)
> write.csv(results_genes,
"chrX_gene_results.csv",row.names=FALSE)
```

Q-value<0.05 in both genes and transcripts

In the tables below, all differentially expressed transcripts and genes are listed. In transcript level (see Table 1) there are 9 transcripts to be shown differentially expressed between males and females and 10 genes (see Table 2) are shown so (with the cutoff of 0.05 q-value). In table 2 the known names of genes are isoforms related to the known genes (see Tables 1&2).

```
> subset(results_transcripts,results_transcripts$qval<0.05)

> subset(results_genes,results_genes$qval<0.05)
```

7) Data visualization

How to create colorful plots

In order to get better understanding of data, colorful plots can be visualized by the following:

```
> tropical= c('darkorange', 'dodgerblue',
'hotpink','limegreen', 'yellow')
> palette(tropical)
```

FPKM in different sex groups

To evaluate the abundance feature of the genes between females and males, we use FPKM measurement in the below commands.
There are number of options to be compared via Ballgown. Here we only measure the FPKM of the transcripts.
To facilitate the drawing process, we should transform the FPKM to log2 (see Figure 18). The second command does so. As there is no log2(0) we add it up with 1.

```
> fpkm = texpr(bg_chrX,meas="FPKM")
> fpkm = log2(fpkm+1)
>
boxplot(fpkm,col=as.numeric(pheno_data$sex),las=2,ylab='log
2(FPKM+1)')
```

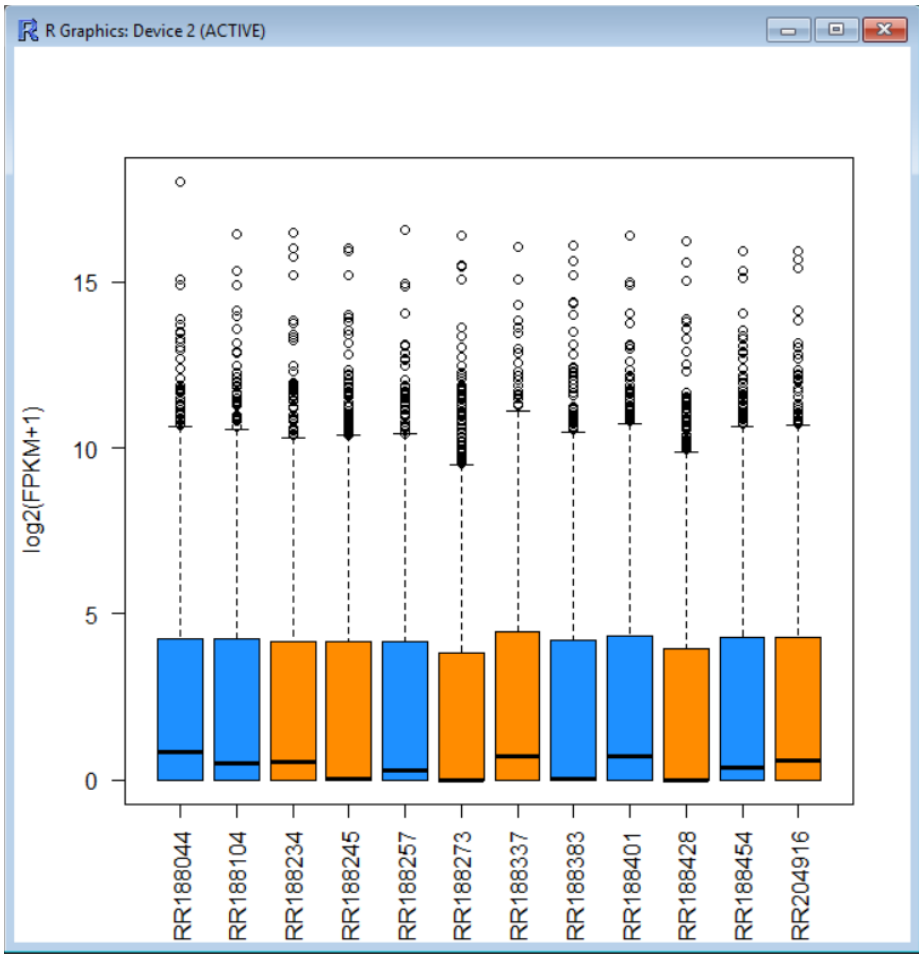


Figure 18 Distribution of FPKM values across the 12 samples

How to plot a specific transcript in samples

To plot a specific transcript, we can first get its name and then the gene that contains this transcript (see Figure 19).

```
> ballgown::transcriptNames(bg_chrX)[12]
> ballgown::geneNames(bg_chrX)[12]
> plot(fpkm[12,] ~ pheno_data$sex, border=c(1,2),
main=paste(ballgown::geneNames(bg_chrX)[12],' : ',
ballgown::transcriptNames(bg_chrX)[12]), pch=19,
xlab="Sex",
ylab='log2(FPKM+1)')
> points(fpkm[12,] ~ jitter(as.numeric(pheno_data$sex)),
col=as.numeric(pheno_data$sex))
```

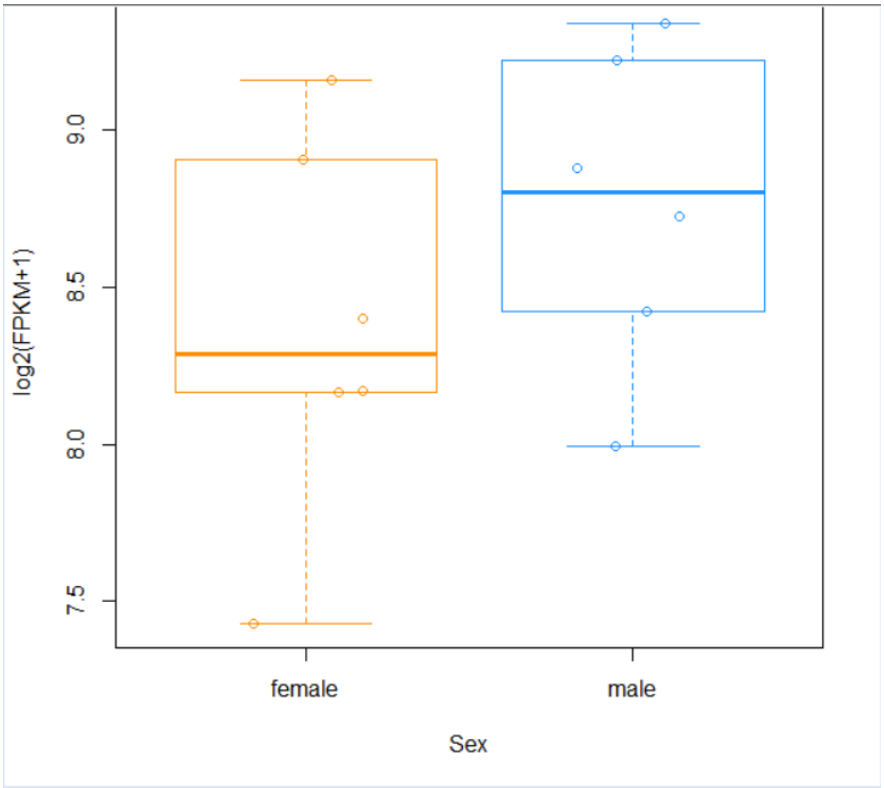


Figure 19 FPKM distributions in males and females for transcript NM_012227 from gene GTPBP6

Transcripts expression with common gene locus appeared in all samples
The level of expression and the structure of the NR_001564 is depicted by the command below. This isoform is grouped as known isoform as I already mentioned above. This is highly expressed in females rather than males. The other isoforms related to gene of this transcript is listed in the table 2 as the differentially expressed ones (see Figure 20).

```
> plotTranscripts(ballgown::geneIDs(bg_chrX)[1729],
bg_chrX,main=c('Gene XIST in sample ERR188234'),
sample=c('ERR188234'))
```

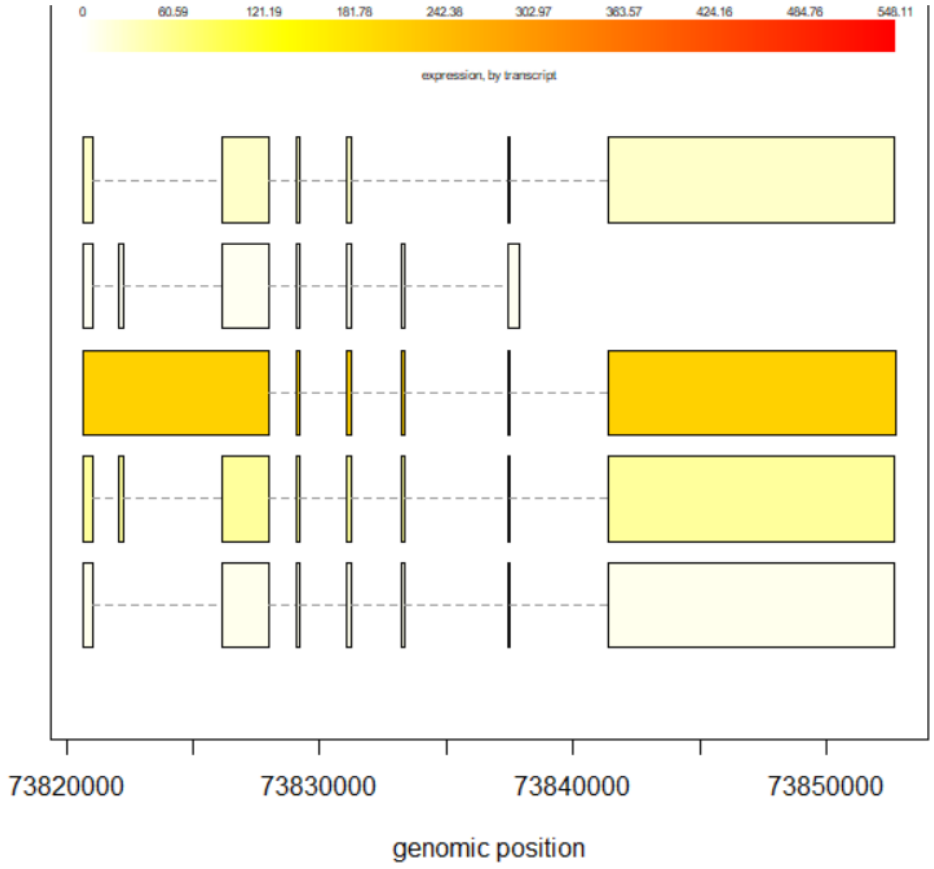


Figure 20 Structure and expression levels of five distinct isoforms of the XIST gene in sample ERR188234.

Average expression values of all transcripts

Here MSTRG.56 is an example: (see Figure 21)

```
>
plotMeans('MSTRG.56',bg_chrX_filt,groupvar="sex",legend=FAL
SE)
```

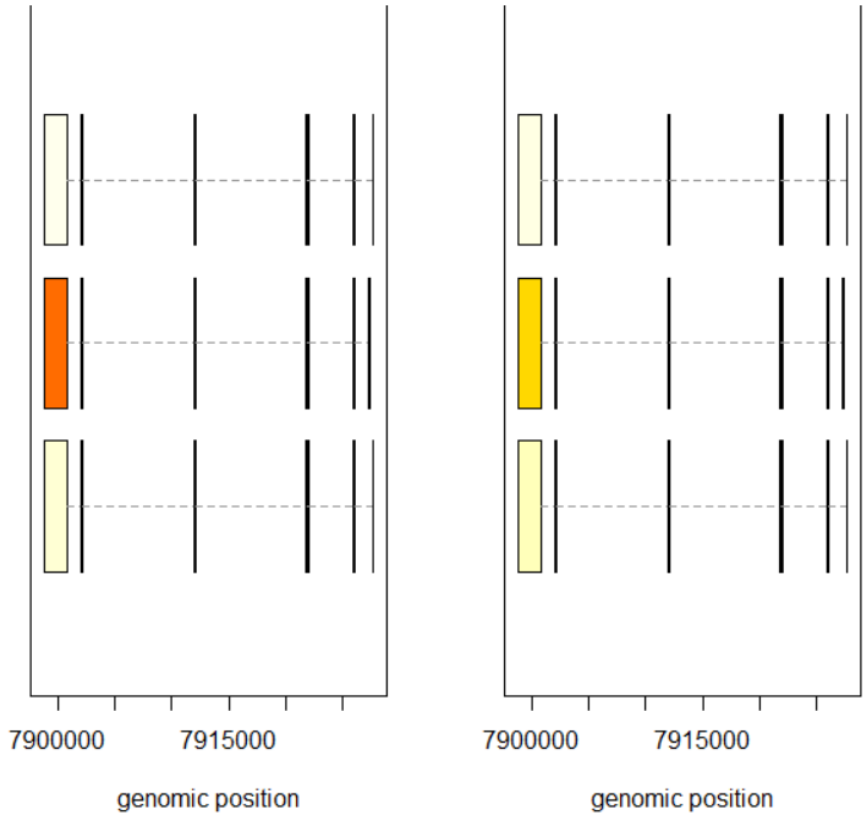


Figure 21 Average level of expression in gene MSTRG.56 for all transcripts in all groups

8) Mapping with STAR:

To align the reads the commands below run:

--runThreadN	number of threads to be used
--genomeDir	path to the genomeDir folder including indices
--readFilesIn (fastq format)	reads path (read1path space read2path) in paired-end, read1 (forward) read2 (reverse)
--outFilesPrefix	path to where outputs are going to be saved/prefix

All reads are unzipped beforehand, and then called by the command.

Here is only one example of the code run by STAR:

```
STAR --runThreadN 8 --genomeDir /home/mina/genomeDir --
readFilesIn
/home/mina/UnzippedSamples/ERR188044_chrX_1.fastq
/home/mina/UnzippedSamples/ERR188044_chrX_2.fastq --
outFileNamePrefix
/home/mina/my_rnaseq_exp/STAR_Outputs/ERR188044
```

9) STAR parameters tested on PNT data

```
STAR --runThreadN 8 --genomeDir /home/mina/genomeDir --
readFilesIn
/home/mina/UnzippedSamples/ERR188044_chrX_1.fastq
/home/mina/UnzippedSamples/ERR188044_chrX_2.fastq --
outFileNamePrefix
/home/mina/my_rnaseq_exp/STAR_Outputs/ERR188044 --
outFilterMatchNmin 20 --seedSearchStartLmax 30 --
outFilterScoreMinOverLread 0 --outFilterMatchNminOverLread
0 --outFilterMismatchNoverLmax 9
```

Appendix B

In Appendix B, we describe the experiments we performed as well as the parameter optimization of the mappers. Firstly, we run both the aligners on different machines of 8 and 64G of RAM to map different sizes of data. We consider the uniquely mapped reads rates and running time in the experiment. The following tables represent time, uniquely mapped reads rates and the impacts of data growth on the speed of both aligners as well as the influence of RAM capacity on running time of aligners, tested on two different machines of 8G and 64G RAM.

1. Tables of results
2. How parameter optimization affects the mapping rates in STAR
3. STAR and HISAT2 parameters to optimize the mapping results
4. Parameters tested in STAR on PNT data
5. Parameters tested in STAR on full data
6. Parameters tested in HISAT2 on full data
7. HISAT2 parameter optimization results
8. STAR parameter optimization results

1) Tables of results

-Uniquely mapped reads rates, HISAT2 vs STAR on a machine with 8G of RAM:

The numbers are percentage of uniquely mapped reads.

Table 6 percentage of uniquely mapped reads rates, HISAT2 vs STAR tested on PNT data (8GB RAM)

		HISAT2					STAR				
SAMPLES		0.1	0.2	0.4	0.8	1	0.1	0.2	0.4	0.8	1
YRI-Male	ERR188044	3.98	3.98	3.97	3.96	3.94	12.46	12.46	12.48	12.47	12.47
YRI-Male	ERR188104	4.24	4.25	4.24	4.24	4.24	12.98	12.99	13	13.01	13
YRI-Female	ERR188234	4.26	4.26	4.25	4.24	4.23	13.99	13.99	14	14	14
GBR-Female	ERR188245	3.72	3.72	3.71	3.7	3.7	12.16	12.15	12.16	12.17	12.17
GBR-Male	ERR188257	3.54	3.54	3.53	3.54	3.53	11.73	11.72	11.72	11.72	11.72
YRI-Female	ERR188273	4.17	4.18	4.17	4.15	4.16	12.81	12.82	12.84	12.85	12.85
GBR-Female	ERR188337	3.41	3.41	3.4	3.39	3.39	11.65	11.65	11.65	11.64	11.64
GBR-Male	ERR188383	3.92	3.91	3.91	3.89	3.89	13.22	13.22	13.23	13.23	13.23
GBR-Male	ERR188401	3.85	3.85	3.86	3.85	3.84	12.62	12.6	12.62	12.61	12.61
GBR-Female	ERR188428	3.62	3.62	3.62	3.61	3.61	11.88	11.9	11.91	11.91	11.91
YRI-Male	ERR188454	3.79	3.77	3.76	3.75	3.75	12.78	12.78	12.76	12.74	12.75
YRI-Female	ERR204916	4.08	4.08	4.08	4.06	4.06	13.25	13.23	13.24	13.23	13.24

-Uniquely mapped reads rates, HISAT2 vs STAR on a machine with 64GByte of RAM:

The numbers are percentage of uniquely mapped reads.

Table 7 percentage of uniquely mapped reads rates, HISAT2 vs STAR (64GB RAM)

		HISAT2					STAR				
SAMPLES		0.1	0.2	0.4	0.8	1	0.1	0.2	0.4	0.8	1
YRI-Male	ERR188044	3.98	3.98	3.97	3.96	3.94	12.46	12.46	12.48	2.47	12.47
YRI-Male	ERR188104	4.24	4.25	4.24	4.24	4.24	12.98	12.99	13	13.01	13
YRI-Female	ERR188234	4.26	4.26	4.25	4.24	4.23	13.99	13.99	14	14	14
GBR-Female	ERR188245	3.72	3.72	3.71	3.7	3.7	12.16	12.15	12.16	12.17	12.17
GBR-Male	ERR188257	3.54	3.54	3.53	3.54	3.53	11.73	11.72	11.72	11.72	11.72
YRI-Female	ERR188273	4.17	4.18	4.17	4.15	4.16	12.81	12.82	12.84	12.85	12.85
GBR-Female	ERR188337	3.41	3.41	3.4	3.39	3.39	11.65	11.65	11.65	11.64	11.64
GBR-Male	ERR188383	3.92	3.91	3.91	3.89	3.89	13.22	13.22	13.23	13.23	13.23
GBR-Male	ERR188401	3.85	3.85	3.86	3.85	3.84	12.62	12.6	12.62	12.61	12.61
GBR-Female	ERR188428	3.62	3.62	3.62	3.61	3.61	11.88	11.9	11.91	11.91	11.91
YRI-Male	ERR188454	3.79	3.77	3.76	3.75	3.75	12.78	12.78	12.76	12.74	12.75
YRI-Female	ERR204916	4.08	4.08	4.08	4.06	4.06	12.46	12.46	12.48	12.47	12.47

-Time, HISAT2 vs STAR on a machine with 8G of RAM:

The numbers show time spent to map by both the aligners (*minutes: seconds* in HISAT2, *minutes* in STAR).

Table 8 running time analysis, HISAT2 vs STAR (on a machine with 8GB RAM)

		HISAT2					STAR				
SAMPLES		0.1	0.2	0.4	0.8	1	0.1	0.2	0.4	0.8	1
YRI-Male	ERR188044	0:33	1:11	2:30	5:22	6:25	46	95	189	366	453
YRI-Male	ERR188104	0:34	1:08	2:23	5:01	6:21	44	88	180	334	417
YRI-Female	ERR188234	0:37	1:19	2:43	5:26	6:54	48	96	190	379	473
GBR-Female	ERR188245	0:23	0:51	1:51	3:46	4:47	35	66	130	260	323
GBR-Male	ERR188257	0:25	0:54	1:55	3:59	4:59	36	68	133	265	332
YRI-Female	ERR188273	0:13	0:27	0:59	2:12	2:40	18	35	67	136	170
GBR-Female	ERR188337	0:35	1:12	2:31	5:32	6:31	41	82	165	326	406
GBR-Male	ERR188383	0:24	0:51	1:51	3:57	4:40	33	66	130	252	318
GBR-Male	ERR188401	0:35	1:11	2:31	5:15	6:28	46	92	187	363	457
GBR-Female	ERR188428	0:33	1:10	2:26	5:16	6:16	37	74	155	301	376
YRI-Male	ERR188454	0:26	0:55	1:58	4:07	5:01	38	71	142	281	352
YRI-Female	ERR204916	0:25	0:55	1:58	4:12	5:01	37	69	135	285	345

-Time, HISAT2 vs STAR on a machine with 64GByte of RAM:

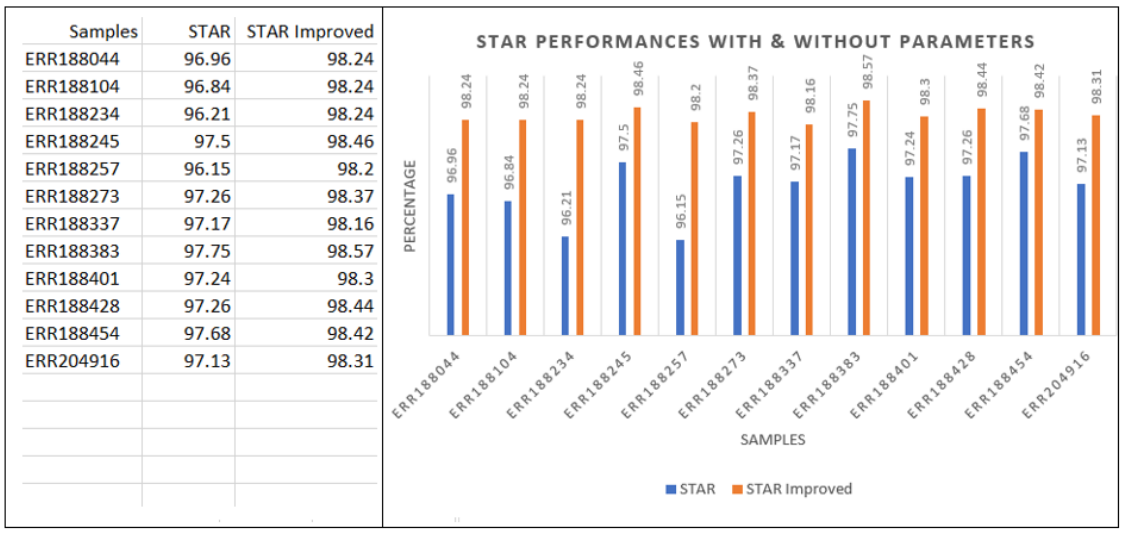
The numbers show time spent to map by both the aligners (*minutes. Seconds* or only *minutes* in HISAT2 and STAR).

Table 9 running time analysis, HISAT2 vs STAR (on a machine with 64GB RAM)

		HISAT2					STAR				
SAMPLES		0.1	0.2	0.4	0.8	1	0.1	0.2	0.4	0.8	1
YRI-Male	ERR188044	0.41	1.19	2.37	5.16	6.35	34.53	69.39	140.1	273.34	342.03
YRI-Male	ERR188104	0.38	1.14	2.28	5.01	6.15	33	67.32	127.2	248.31	312.52
YRI-Female	ERR188234	1.03	1.23	2.5	5.4	7.06	35.33	71.12	141.3	282.12	351.34
GBR-Female	ERR188245	0.29	0.58	1.57	3.51	4.53	28.56	51.24	97.01	194.33	246.13
GBR-Male	ERR188257	0.30	1.01	2	4	5.08	30.03	52.39	99.45	198.39	246.20
YRI-Female	ERR188273	1.04	0.32	1.05	2.06	2.43	15.45	27.41	50.32	101.19	126.57
GBR-Female	ERR188337	0.38	1.19	2.37	5.15	6.29	32.38	63.05	121.5	241.05	301.04
GBR-Male	ERR188383	1.22	0.58	1.54	3.48	4.47	26.17	51.52	98.19	188.44	237.33
GBR-Male	ERR188401	1.22	1.17	2.36	5.13	6.34	35.17	69.37	140.2	272.50	340.55
GBR-Female	ERR188428	0.56	1.19	2.37	5.14	6.27	28.57	56.09	114.6	225.25	278.47
YRI-Male	ERR188454	1.26	1.30	2.02	4.02	5.09	32.09	55.58	104.6	209.15	259.37
YRI-Female	ERR204916	1.11	1.01	2.06	4.07	5.20	31.27	54.58	103.1	204.33	259.36

2) How parameter optimization affects the mapping rates in STAR

We run the STAR on the PNT RNA-seq data. Then we optimize the parameters used in the STAR and compare the results with those of not optimized. The following table and graph represent the results of with and without parameter tuning in STAR.



Samples	STAR	STAR Improved
ERR188044	96.96	98.24
ERR188104	96.84	98.24
ERR188234	96.21	98.24
ERR188245	97.5	98.46
ERR188257	96.15	98.2
ERR188273	97.26	98.37
ERR188337	97.17	98.16
ERR188383	97.75	98.57
ERR188401	97.24	98.3
ERR188428	97.26	98.44
ERR188454	97.68	98.42
ERR204916	97.13	98.31

Figure 22 STAR uniquely mapped rates comparison on PNT data

--outFilterMismatchNoverLmax	which is max number of mismatches per pair relative to read length. for 2x100b, max number of mismatches is 0.06*200=8 for the paired read. So, in our case it is 9 as the readlength is 75, 2x75b. 0.06x150=9 [13].
--outFilterMatchNmin 20	options that help us to control identity between reads and genome. In other words, they control the minimum number of bases which are matched.Default value is 0. This makes stricter decision on the output of mapping as it restrict the mapping value to those which at least have the number of matched bases higher than this value. It allows alignment of shorter reads.
--outFilterMatchNminOverLread 0	Default is :0.66 It can be used to control min identity (number of base/bases normalized to the read length)
-- outFilterScoreMin 0	The absolute value of the score. If it sets closer to read length, it may result in reads with higher quality.
--outFilterScoreMinOverLread 0	Default: 0.66 The score divided by the length of the read pair. Although the default value determines that only reads with the alignment score more than two third of the read length will be output, we change it to zero. (closer to 1 means selection of reads with higher quality, more stringent!) However, we may expect that accuracy might decrease, this might be a good option being used in our case (fewer data).
--seedSearchStartLmax 30	as we have reads of 75 lengths, so we can define a new set of this parameter by 30. The default one is 50. In our case 30 would be a better option to deal with. The starting point via the reads. The default value is 50.

`--outFilterMultimapNmax`	Multi-mapped reads should be controlled well. This is a good help to deal with. There is always a trade-off between the values and the quality of results.
`--genomeSAindexNbases`	If you are using a very small genome, it should be set in the related parameter
`--alignIntronMin &` `--alignIntronMax`	The allowed minimum and maximum sizes of introns.

3) STAR and HISAT2 parameters to optimize the mapping results

Here**,** the parameters and their influences on the study is written as well as the tables of mapping results representing before and after optimization changes in both aligners.

4) Parameters tested in STAR on PNT data

To observe how parameter optimization affects the mapping results, we chose only a small portion of the full data for a quick check.

5) Parameters tested in STAR on full data

STAR tested parameters. Only outFilterMismatchNmax is optimized to 1.

outFilterMismatchNmax	0	1	3	5	7
Uniquely Mapped Reads	4.92	5.83	7.88	9.52	11.1
Multimapped Reads	0.2	0.38	0.5	0.5	0.64
Not Mapped	94.88	93.79	91.63	89.99	88.26
Mismatch Rate	0	0.38	1.13	1.86	2.65
outFilterMismatchNoverLmax	0	0.1	0.2	0.4	
Uniquely Mapped Reads	4.6	12.94	12.91	12.91	
Multimapped Reads	0.12	0.5	0.5	0.5	
Not Mapped	95.29	86.56	86.59	86.59	
Mismatch Rate	0	3.61	3.52	3.52	
seedSearchStartLmax	20	30	40	60	
Uniquely Mapped Reads	12.94	12.86	12.91	12.91	
Multimapped Reads	0.85	0.82	0.5	0.5	
Not Mapped	86.21	86.33	86.59	86.59	
Mismatch Rate	3.55	3.54	3.52	3.52	
seedSearchLmax	1	3	5	7	
Uniquely Mapped Reads	12.91	12.91	12.91	12.91	
Multimapped Reads	0.5	0.5	0.5	0.59	
Not Mapped	86.59	86.59	86.59	86.5	
Mismatch Rate	3.52	3.52	3.52	3.53	

winAnchorMultimapNmax	10	20	30	40	60		
Uniquely Mapped Reads	12.77	12.77	12.8	12.83	12.91		
Multimapped Reads	0.53	0.59	0.56	0.53	0.5		
Not Mapped	86.71	86.65	86.65	86.65	86.59		
Mismatch Rate	3.5	3.5	3.5	3.51	3.5		
outFilterMatchNminOverLread	0.2	0.4	0.6	0.7	0.8	0.85	1
Uniquely Mapped Reads	12.91	12.91	12.91	12.45	11.63	10.66	12.91
Multimapped Reads	0.5	0.5	0.5	0.47	0.47	0.41	0.5
Not Mapped	86.59	86.59	86.59	87.09	87.91	88.93	86.59
Mismatch Rate	3.52	3.52	3.52	3.43	3.21	2.96	3.52
outFilterScoreMinaoverLread	0.6	0.69	0.75	0.8			
Uniquely Mapped Reads	14.41	12.59	11.07	10.22			
Multimapped Reads	3.84	0.47	0.44	0.41			
Not Mapped	84.98	86.94	88.49	89.37			
Mismatch Rate	3.84	3.44	3.03	2.79			
alignSjOverhangMin	1	3	4	7	10		
Uniquely Mapped Reads	12.83	12.86	12.88	12.91	12.94		
Multimapped Reads	0.59	0.56	0.53	0.5	0.44		
Not Mapped	86.59	86.59	86.59	86.59	86.62		
Mismatch Rate	3.51	3.51	3.52	3.52	3.52		

alignSjOverhangMin	1	3	4	7	10
Uniquely Mapped Reads	12.83	12.86	12.88	12.91	12.94
Multimapped Reads	0.59	0.56	0.53	0.5	0.44
Not Mapped	86.59	86.59	86.59	86.59	86.62
Mismatch Rate	3.51	3.51	3.52	3.52	3.52
limitoutSJcollapsed	900,000	700,000	500,000	200,000	50,000
Uniquely Mapped Reads	12.91	12.91	12.91	12.91	12.91
Multimapped Reads	0.5	0.5	0.5	0.5	0.5
Not Mapped	86.59	86.59	86.59	86.59	86.59
Mismatch Rate	3.52	3.52	3.52	3.52	3.52
limitSJdbInsertNSj	700,000	500,000	50,000		
	12.91	12.91	12.91		
	0.5	0.5	0.5		
	86.59	86.59	86.59		
	3.52	3.52	3.52		
outFilterType	BySJout				
	12.91				
	0.41				
	86.68				
	3.5				

6) Parameters tested in HISAT2 on full data

The following parameters made no changes on the results except
--PEN-NONCANSPLICE (default:3) & --PEN-CANSPLICE

`--pen-cansplice 1,2,3,4,5,6,7`	sets the penalty for each of the pair that is canonical splice site
`--pen-noncansplice (default:3)`	sets the penalty for each of the pair that is not canonical splice sites.
`--qc-filter (default:off)`	in reads with format of --qseq filters those reads with qseq filter field of non-zero.
`--no-discordant`	by default, concordant reads pairs are found by hisat, otherwise discordant ones will be found.

`--minins`
 	
`--maxins 0,100 0,200`
`0,800 0,1000`
	minimum and maxmum fragment length (default0 and 500 resp.)

`--no-mixed`	if no concordant or discordant reads are found then hisat looks for individual mates.
`--sp MX,MN` `1,2 2,1 3,2 4,3 5,4`	maximum and minimum soft-clipping per base

`--min-intronlen 15,20,25,30`
 	
`--max-intronlen`
`100,000 200,000`
`700,000 800,000`
	minimum and maximum intron length

`--seed 1,3,5,10,15,20`	number of seeds to be used for pseudo-random number generator
`--non-deterministic`	two identical reads may not be reported as the same alignments if we specify this parameter in command.

pen-noncansplice

Values tested	1	2	5	7	8	9	10
Uniquely-mapped-reads %	4.3	4.3	4.3	4.3	4.3	4.3	4.3
Multi-mapped-reads %	1.28	1.28	1.27	1.27	1.27	1.27	1.27
No-mapped reads %	94.42	94.42	94.42	94.42	94.42	94.43	94.43
X*	1.7	1.7	1.69	1.69	1.68	1.68	1.68

pen-cansplice

Values tested	1	2	3	4	5	6	7
Uniquely-mapped-reads %	4.32	4.32	4.32	4.32	4.33	4.32	4.32
Multi-mapped-reads %	1.25	1.25	1.24	1.24	1.24	1.24	1.24
No-mapped reads %	94.43	94.43	94.43	94.44	94.44	94.44	94.44
X*	1.68	1.68	1.68	1.68	1.68	1.68	1.68

sp

Values tested	1,2	2,1	3,2	4,3	5,4
Uniquely-mapped-reads %	4.3	4.44	4.48	4.62	4.51
Multi-mapped-reads %	1.27	0.75	0.6	0.43	0.54
No-mapped reads %	94.43	94.81	94.42	94.95	94.95
X*	1.68	1.89	1.96	2.05	2.02

X*= Of the total percentage of pairs aligned 0 times concordantly or discordantly, X % are aligned exactly 1 time

7) HISAT2 parameter optimization results:

The optimization process is done using HISAT2 on all the 12 samples.
in the next table, only the uniquely mapped rates are compared (with and without parameter optimization).

Table 10 the mapping results when no optimization on parameters in HISAT2 is applied on PNT data (8GB RAM)

	Uniquely mapped	Multimapped	Not mapped	Mismatch
ERR188044	4.01	1.22	94.77	1.53
ERR188104	4.3	1.24	94.46	1.67
ERR188234	4.29	1.59	94.12	1.74
ERR188245	3.75	1.44	94.8	1.65
ERR188257	3.58	1.3	95.11	1.55
ERR188273	4.22	1.21	94.57	1.52
ERR188337	3.44	1.5	95.06	1.49
ERR188383	3.95	1.64	94.41	1.63
ERR188401	3.9	1.39	94.71	1.58
ERR188428	3.64	3.29	93.06	1.28
ERR188454	3.8	1.76	94.44	1.49
ERR204916	4.12	1.6	94.28	1.61

-HISAT2 uniquely mapped reads rates optimized vs non-optimized:

Table 11 the mapping results when optimization on parameters in HISAT2 is applied on PNT data (8GB RAM). Only uniquely mapped reads rates are shown.

	OPTIMIZED	NOT-OPTIMIZED
ERR188044	4.01	3.94
ERR188104	4.3	4.24
ERR188234	4.29	4.23
ERR188245	3.75	3.7
ERR188257	3.58	3.53
ERR188273	4.22	4.16
ERR188337	3.44	3.39
ERR188383	3.95	3.89
ERR188401	3.9	3.84
ERR188428	3.64	3.61
ERR188454	3.8	3.75
ERR204916	4.12	4.06

8) STAR Parameter optimization results

Table 12 the mapping results when no optimization on parameters in STAR is applied on PNT data (8GB RAM)

	Uniquely mapped	Multimapped	Not mapped	Mismatch
ERR188044	5.89	0.24	93.88	0.37
ERR188104	6.22	0.27	93.52	0.39
ERR188234	6.57	0.26	93.18	0.39
ERR188245	5.7	0.27	94.03	0.44
ERR188257	5.45	0.25	94.31	0.4
ERR188273	6.05	0.26	93.7	0.39
ERR188337	5.5	0.28	94.22	0.36
ERR188383	6.22	0.23	93.55	0.38
ERR188401	5.98	0.26	93.77	0.37
ERR188428	6.81	0.54	92.65	0.57
ERR188454	6.15	0.26	93.6	0.37
ERR204916	6.3	0.26	93.45	0.41

Table 13 the mapping results when optimization on parameters is applied on PNT data in HISAT2 (8GB RAM). Only uniquely mapped reads rates are shown.

Uniquely mapped reads		
	OPTIMIZED	NOT-OPTIMIZED
ERR188044	5.89	12.47
ERR188104	6.22	13
ERR188234	6.57	14
ERR188245	5.7	12.17
ERR188257	5.45	11.72
ERR188273	6.05	12.85
ERR188337	5.5	11.64
ERR188383	6.22	13.23
ERR188401	5.98	12.61
ERR188428	6.81	11.91
ERR188454	6.15	12.75
ERR204916	6.3	13.24

Table 14 the mapping results when optimization on parameters is applied on PNT data in STAR (8GB RAM). Only mismatch rates are shown.

MISMATCH		
	OPTIMIZED	NOT-OPTIMIZED
ERR188044	0.37	3.54
ERR188104	0.39	3.49
ERR188234	0.39	3.56
ERR188245	0.44	3.56
ERR188257	0.4	3.58
ERR188273	0.39	3.53
ERR188337	0.36	3.5
ERR188383	0.38	3.53
ERR188401	0.37	3.51
ERR188428	0.57	2.97
ERR188454	0.37	3.45
ERR204916	0.41	3.53

Appendix C

Installation of different software packages used in the study

In the current Appendix, we explain how to install the different packages used in the experiment. Additionally, how data are downloaded is also described. The commands are written into the terminal of Linux operating system.

1. Equipment setup
2. Downloading software packages
    a. Samtools
    b. HISAT2
    c. StringTie
    d. Gffcompare
    e. Ballgown
3. HISAT2 indices
4. Download, Install and run STAR
    a. Download STAR
5. Generating genome indices in STAR

1) Equipment Setup

In the upstream analysis of data, Ubuntu environment is used as mentioned. Terminal window will do the job to run the commands. We consider a folder named as my_rnaseq_data to put all the requirements in. Be careful to make a bin folder in the root as your PATH to put all the executable files of the programs in.

2) Downloading software packages

Data is accessible via ftp://ftp.ccb.jhu.edu/pub/RNAseq_protocol/chrX_data.tar.gz. After downloading data should be unzipped with the following command:

```
$ tar xvzf chrX_data.tar.gz
```

You will see that there are two files and four folders of samples, *inexes, genomes* and *genes* in the chrX_data. In sample folder there are 24 files related to samples. Each sample has 2 files of forward and reverse reads, since the reads are all paired-end. It should be mentioned that samples are from two populations of GBR (British from England) and YRI (Yoruba from Ibadan, Nigeria). Half of the samples are males' and the other half females'. Genes folder contains the annotation file as.GTF file. The indexes file contains 8 files named *chrX_tran.1.ht2, chrX_tran.2.ht2, chrX_tran.3.ht2, chrX_tran.4.ht2, chrX_tran.5.ht2, chrX_tran.6.ht2, chrX_tran.7.ht2, and chrX_tran.8.ht2*. All are pre-built indexes of

chromosome X downloaded. The genome directory including a file named chrX.fa, which is the file of sequences for chromosome X in human (GRCh38 build 81).

Make a folder to put all executable files in, add to your PATH. By typing mkdir in the Terminal, we can make directory in each path we consider. The following command creates a bin folder that will be used later as the source of all executable files.

```
$ mkdir $HOME/bin
```

Add the created folder as your PATH to the environment variable:

```
$ export PATH=$HOME/bin:$PATH
```

Environmental variables are variables that are defined for the current shell and are inherited by any child shells or processes.

a) Samtools

Get the Samtools package via the link in the Equipment part, then unpack the file:
$ tar jxvf samtools-0.1.19.tar.bz2
In the samtools unzipped file directory write the following commands:

```
$ cd samtools-0.1.19
$ make
$ cd ..
```

Then the Samtools binary is copied in the PATH just created:

```
 cp samtools-0.1.19/samtools $HOME/bin
```

b) HISAT2

After downloading HISAT2, the file is unzipped:

```
$ unzip HISAT22-2.0.1-beta-OSX_x86_64.zip
```

The same goes for the HISAT2 package too, the executable files should be copied to the PATH:

```
cp HISAT22-2.0.1-beta/HISAT22* HISAT22-2.0.1-beta/*.py
$HOME/bin
```

c) StringTie

The below command followed by downloading and unpacking the StringTie:

```
$ tar xvzf stringtie-1.2.2.OSX_x86_64.tar.gz
```

cd to the unpacked directory:

```
$ cp stringtie-1.2.2.OSX_x86_64/stringtie $HOME/bin
```

d) Gffcompare

Downloading is done, then file is unpacked:

```
tar xvzf gffcompare-0.10.1.Linux_x86_64.tar.gz
```

And cd to the folder:

```
$ make
```

the gffcompare file is copied into the PATH:

```
$ cp gffcompare-0.10.1.Linux_x86_64/gffcompare $HOME/bin
```

e) Ballgown

After installation of R software, with the version of 3.2.2, Ballgown package should be installed. The command is written in the R console:

```
> install.packages("devtools",repos="http://cran.us.r-
project.org")
> source("http://www.bioconductor.org/biocLite.R")
>biocLite(c("alyssafrazee/RSkittleBrewer","ballgown","genef
ilter","dplyr", "devtools"))
```

3) HISAT2 index

There are two options to get the HISAT2 indexes available. The first is downloading them via the HISAT2 website. And the second is creating the indexes if they are not available. The following commands make the HISAT2 indexes:
Before making indexes, we run a python script in the HISAT2 folder that determines splice sites:

```
$ extract_splice_sites.py chrX_data/genes/chrX.gtf >
chrX.ss
$ extract_exons.py chrX_data/genes/chrX.gtf > chrX.exon
```

- ✓ The Shebang on top of python script should be changed to the python executable current PATH, i.e. #!/home/mina/anaconda2/bin/python

Then we build the HISAT2 indexes:
Options:

--ss	splice site file name
--exon	exon file name

```
$ HISAT22-build --ss chrX.ss --exon chrX.exon
chrX_data/genome/chrX.fachrX_tran
```

4) Download, install and run STAR

The process of alternating STAR with HISAT2 is begun by firstly downloading the STAR package. All the steps necessary to download the package is in Appendix A. The package is downloaded and installed in terminal of Linux. The "make" function should be run after unzipping the files.

The codes are run in the terminal and by using the options to make indices, we then start to map by STAR.

a) Download STAR:

To download STAR package, the following command is written in Terminal of the Linux:

```
wget
https://github.com/alexdobin/STAR/archive/2.5.3a.tar.gz
```

When downloading is finished, the file should be unzipped:

```
tar -xzf 2.5.3a.tar.gz
```

to build the STAR we need to cd to the unzipped file:

```
cd STAR-2.5.3a
```

To get STAR source using git:

```
git clone https://github.com/alexdobin/STAR.git
cd STAR/source
```

In the source file:

```
make
```

5) Generating Genome Indices in STAR

Before starting to map the reads, genome index files are required. If you wish to build them yourself:

```
STAR --runThreadN 8 --runMode genomeGenerate --genomeDir
/home/mina/genomeDir --genomeFastaFiles
/home/mina/my_rnaseq_exp/chrX_data/genome/chrX.fa --
sjdbGTFfile
/home/mina/my_rnaseq_exp/chrX_data/genes/chrX.gtf --
sjdbOverhang 74
```

As you see, there are options used in the command:

--runThreadN	Number of threads to be used
--runMode generation	commands STAR to run genome indices
--genomeDir	specifies genome directory path
--genomeFastaFiles	Fasta file (or files)
--sjdbGTFfile	Annotated transcripts file
--sjdbOverhang	readlength-1

- ✓ It should be mentioned that genomeDir folder must be made before starting the run. All the indices will put here.

- ✓ You should be careful that options are case sensitive. So, each should be written the same as above.

Appendix D

For the experiment, the required hardware and software packages names and specifications are included in the current Appendix. To get the same results as [1] we need to download and run the same software and versions as PNT [1].

1. Hardware
2. OS
3. Software
4. Extra installations

1) Hardware:

- 64-bit computer running either Linux or Mac OS X (10.7 Lion or later)
- 4 GB of RAM (8GByte preferred)

2) OS:

- Upstream process of analysis is performed in Ubuntu environment (for example version of 17.04).
- Downstream process is performed on windows 10, 64 bits.

3) Software:

- HISAT22 software (https://anaconda.org/bioconda/HISAT22/2.0.1beta/download/linux-64/HISAT22-2.0.1beta-py34_0.tar.bz2) (version 2.0.1 or later).
- StringTie software (http://ccb.jhu.edu/software/stringtie/dl/stringtie-1.2.2.Linux_x86_64.tar.gz), version 1.2.2 or later.
- SAM tools (https://kent.dl.sourceforge.net/project/samtools/samtools/0.1.19/samtools-0.1.19.tar.bz2) (version 0.1.19 or later).
- Gffcompare (http://ccb.jhu.edu/software/stringtie/dl/gffcompare-0.9.5.Linux_x86_64.tar.gz) version -0.10.1
- R (https://www.r-project.org) (version 3.2.2 or later).
- STAR aligner version 2.5.3a

4) Extra installations

- Zlib in Ubuntu should also be installed. Since some programs need it to run. It can be installed by apt-get in Terminal.
- Seqtk software needed for down-sampling