Albert Ludwig University of Freiburg
&
Novartis Institutes for Biomedical Research

Master's Thesis

Assessment and Comparison of RNA-seq Quantification Methods

Author:

Daniel Desirò

Supervisor:

apl. Prof. Dr. Sven Schuierer

May 2016

Imprint

Author	Daniel Massimo Maria Desirò
Timeframe	December 1st, 2015 to June 1st, 2016
Reviewer	apl. Prof. Dr. Sven Schuierer Novartis Institutes for Biomedical Research Novartis Pharma Switzerland
	Prof. Dr. Rolf Backofen Chair for Bioinformatics Albert Ludwig University of Freiburg
Supervisor	apl. Prof. Dr. Sven Schuierer Novartis Institutes for Biomedical Research Novartis Pharma Switzerland
Examination Regulations	The presented master's thesis is written according to the examination regulations of the Albert Ludwig University of Freiburg from August 19th, 2005 for the degree Master of Science. It consists of the main examination regulations for the course of studies in Informatics and corresponding enclosure regulations for the course of studies in Bioinformatics and Systems Biology.
Examination Office	Chair for Bioinformatics Department of Computer Science Albert Ludwig University of Freiburg

Statement of Authorship	I hereby testify that the presented thesis is written by myself without the use of sources or aids other than those named. Passages taken verbatim or in substance are marked as such and are referenced accordingly. I further testify that this thesis was not presented in another examination.

Freiburg, May 31th, 2016

Daniel Desirò

Summary

The goal of this thesis was, as the title suggests, the comparison of different quantification methods for RNA-seq data. In order to achieve this, the python based ARQ pipeline was created. The purpose of ARQ is to automate the whole workflow from the alignment of paired-end RNA-seq data and the subsequent generation of gene or transcript counts with a RNA-seq quantification tool. ARQ was then applied to a data set (SEQC) [33] with two different sample types, one from paired-end Universal Human Reference RNA and one from paired-end Human Brain RNA. In order to evaluate the different quantification methods, the resulting counts were then used as input for an R based statistic script. Furthermore smoothed and averaged versions of the results from the Cufflinks [11] quantification method were used to generate a simulated data set (SEQS) with the Flux Simulator [16] simulation tool. ARQ was then also applied to this second data set and the results were again evaluated with the R based statistic script. In addition the results of the SEQC data set were also evaluated with two qPCR goldstandards (Bio-rad and TaqMan) [33]. For the SEQS data set, the true counts were given by the corresponding profile files from the Flux Simulator.

Acknowledgements

I want to thank Sven Schuierer for the interesting and exercising subject of my thesis and also the great opportunity to do my thesis at such a major company as Novartis and its research branch, the Novartis Institutes for BioMedical Research. It was a great work experience and insight in (computational) life science from a health care perspective. And I am also grateful for his good guidelines and great support as my supervisor. I also want to thank Guglielmo Roma for accepting me as a master student at the company and giving me the freedom of focusing on the elaboration of my thesis. A special thanks also belongs to Damir Bucar as a good friend and helping me out on various questions. I am also thanking all my friends and my girlfriend for reviewing my thesis and my parents for providing the support and opportunity to pursue my academic education.

Contents

List of Figures

List of Tables

Abbreviations

FASTA	nucleotide sequence storage format
FASTQ	nucleotide sequence and quality storage format
GTF	**G**ene **T**ransfer **F**ormat
BAM	**B**inary sequence **A**lignment/**M**ap format
SAM	**S**equence **A**lignment/**M**ap format
BED	**B**rowser **E**xtensible **D**ata
qRT-PCR	**q**uantitative **R**eal-**T**ime **P**olymerase **C**hain **R**eaction
RNA-Seq	**RNA**-**Seq**uencing
SEQC	**Se**quencing **Q**uality **C**ontrol Consortium
SEQS	**Se**quencing **Q**uality Control **S**imulation
HBR	**H**uman **B**rain **R**eference
UHRR	**U**niversal **H**uman **R**eference **R**NA
FPKM	**F**ragments **P**er **K**ilobase per **M**illion
DESeq	**D**ifferential **E**xpression for **Seq**uence count data
cDNA	**c**omplementary **DNA**
ncRNA	**n**on-**c**oding **RNA**
qsub	**q**ueue **sub**mission

Chapter 1

Introduction

In recent years RNA-sequencing has become a widely used and appreciated technology for various genomic experiments and studies [1]. It is a highly parallel method [1] for genome-wide analysis of the transcriptome and provides several viewpoints on transcriptomic diversity [2] as well as the regulation of gene expression. This includes alternative splicing, allele-specific expression [2] and polyadenylation [3]. It is therefore not surprising, that RNA-seq has been slowly surpassing and overtaking the role of microarrays [28] and has become the most commonly used technology in biomedical research [1].

The most important and common usage of RNA-seq is the quantification of gene expression [2] and from this the determination of differentially expressed genes between multiple conditions [1]. To achieve this, there have been various approaches from bioinformaticians to invent methods and algorithms which explore the quantification of RNA-seq experiments from different perspectives. The challenge here is, that most of the currently used RNA-seq machines yield relatively short reads with about 200 nt [26]. In contrast to this are eucaryotic transcripts of which mammalian transcripts are on average 2.2 kb long [26]. This results in sequencing reads usually being assigned to multiple isoforms [3]. There exists one approach from Pacific Biosciences that can sequence full-length cDNAs [27] but with the drawback of having relatively high error rates and a low throughput [3]. It is therefore not feasible to use such an approach.

To address the problems with the difference in sequenced reads and transcript length, two main steps have emerged. After the reads produced by the RNA-seq machines are stored in FASTQ files, the reads are first aligned to a reference genome or transcriptome [1–3, 8]. In the second step, these alignments are used to estimate the expression level for each transcript [1–3, 8]. For this the various tools have different strategies to distribute the expression of alternative transcripts within each gene [1]. This led to the development of two different types of approaches.

In the first method, reads overlapping with the feature reference sequence are simply counted. The approach is less optimal for getting the expression values for each transcript and more suited to determine the expression values of the genes, exons and junctions. This is due to the high possibility that a read mapped to a gene is shared between multiple transcripts.

The second method is designed to overcome this problem in calculating expression estimates by quantifying the transcript isoform abundance [8]. Here reads are distributed among transcripts with the use of statistical models and optimization algorithms [8]. To calculate the expression of a gene, the expression estimates of all transcripts belonging to this gene can then simply be added up [8].

Recently a third approach has been developed, which entirely skips the whole alignment step of the RNA-seq expression estimation pipeline. This method pseudoaligns reads to a reference transcriptome to identify possible transcripts from which the reads could have originated [15]. By doing this it only searches for compatible transcripts and avoids the alignment of single bases [15].

In order to successfully compare the three different approaches and to provide a fair evaluation, only gene counts will be considered in this work. With this none of the quantification approaches has an advantage over the others. Previous works have already thoroughly examined the effects of different alignment methods for a quantification method and vice versa. Therefore and also for the reason that kallisto [15] doesn't have alignments at all, the effect of different alignment methods on a quantification method is not evaluated. Different combinations of aligners and quantifiers are classified as one method and judged according to the final resulting gene counts.

Additionally the memory and time consumption of the algorithms will not be considered. This is because some were already analysed thoroughly by Kanitz et al. and although the runtime substantially varies among methods [3], usually compromises can make be made if an algorithm is better suited for a task but requires more resources.

The focus of this thesis lies on four different alignment methods [4–7] and eight different quantification scripts [8–15] with different approaches. Some of the quantification scripts are counting based and some are transcript isoform abundance based. Two different data sets were used to test these tools, from which one was an experimental RNA-seq data set. The problem when evaluating experimental RNA-seq data is that there usually does not exist a ground truth [2]. Therefore a second simulated data set was used to also evaluate the various methods. Its expression values were based on a portion of the quantified data from the experimental data set.

Chapter 2

Overview

The alignment tools used in this work were TopHat2 [4], HISAT [5], STAR [6] and Bowtie2 [7]. The resulting alignments of the first three were used in combination with the three count based methods htseq-count [9], featureCounts [10] and EQP-QM [8], and with the two transcript abundance based methods Cufflinks [11] and FluxCapcitor [12]. The last alignment tool, Bowtie2, was used to obtain alignments on the transcriptome level for the further use with BitSeq [13] and RSEM [14]. The last quantification tool, kallisto [15], is a novel method which does not require any preceding alignment and is based on the concept of transcript abundance estimates.

The experimental data set [33] consists of two parts. The first part, SEQC-A consists of the Universal Human Reference RNA (UHRR) and the second part, SEQC-B consists from the Human Brain Reference (HBR). Flux Simulator [16] was used to generate the simulated data set. The expression values for the simulations were based on a slightly smoothed and averaged version of the quantified Cufflinks results. Furthermore TaqMan [33] and Bio-Rad [33] qRT-PCR data were used as goldstandards to assess the expression estimates from the SEQC [33] data set. All produced transcript estimations were transformed into gene counts and normalized with the R DESeq [17] package. This normalized results were then analysed from various perspectives.

Chapter 3

Aligner

3.1 Settings

To ensure a fair evaluation of all tools, parameters which were similar in multiple alignment programs were set to the same values. The settings for these parameters can be found in Figure 3.1. All other tool specific settings can be found in the corresponding tool section. All alignments were produced with the use of multiple cores.

```
minimum intron length:
        MIN_INTRON_LEN   = 20
maximum intron length:
        MAX_INTRON_LEN   = 500000
maximum allowed multihits:
        MAX_MULTI_HITS   = 100
inner mate distance option:
        MATE_DIST_OPT    = 500
maximum allowed mismatches:
        MAX_MISMATCHES   = 10
read length:
        READ_LENGTH      = 100
```

FIGURE 3.1: Alignment command line settings for similar parameters.

3.2 Functionality

3.2.1 TopHat2

TopHat2 [4] is an alignment tool, designed to work with RNA-seq reads from the Illumina Genome Analyzer. It uses parts of the Bowtie2 [7] read mapping program and incorporates major improvements to its predecessor, TopHat [29]. Like TopHat, TopHat2 first detects possible splice sites for introns and then uses these potential sites to correctly align reads overlapping multiple exons. One of the major differences and improvements to TopHat, is the inclusion of Bowtie2. It addresses the problem, that transcripts from the target genome may contain structural differences, like insertions and deletions, and therefore vary vastly in contrast to the reference genome [30, 31]. Bowtie2 can discover insertions and deletions due to sequencing errors very efficiently.

As a further improvement TopHat2 comprises new algorithms, which can address more various types of RNA-seq reads. The inclusion of Bowtie2 means, that in addition to generating a TopHat index from the reference genome file and the corresponding GTF annotation file, a Bowtie2 index has to be constructed. This Bowtie2 index has to be created prior to the TopHat2 index. TopHat2 uses the Bowtie2 index for generating the TopHat2 index and also needs the reference FASTA genome file inside the Bowtie2 index folder.

Afterwards the alignments for multiple FASTQ RNA-seq files can be generated. TopHat2 then writes its alignments inside two output BAM files. One with reads which were accepted from TopHat2, `accepted_hits.bam` and one with unmapped reads, `unmapped.bam`. This is not optimal, because the quantifiers normally need these, too. Therefore the two output files have to be merged. Sadly this is not possible by simply combining both files, because of a faulty BAM header information of the `unmapped.bam` file which results in the BAM header information of a merged read file becoming corrupted. Luckily this can be fixed with the use of a simple script invented by Brueffer et al. [18].

The `tophat-recondition` script takes the Tophat2 output files and creates a fixed unmapped reads file, `unmapped_fixup.bam`. It is then possible to extract the header information from this fixed BAM file with samtools [19]. In combination with this extracted header information, samtools can then be used to merge the `accepted_hits.bam` and the `unmapped_fixup.bam` files to create a complete alignment file. The resulting alignment file was then name sorted with the samtools sort function. The full workflow is presented in Figure 3.2.

3.2.2 HISAT

HISAT (hierarchical indexing for spliced alignment of transcripts) [5] is based on the Bowtie2 alignment tool with the focus on a low memory and time consumption. Like Bowtie2, it uses an indexing strategy based on the Burrows-Wheeler transform and the Ferragina-Manzini (FM) index. This new indexing scheme, called hierarchical indexing, first constructs a global FM index which represents the whole genome. Afterwards it generates multiple individual local FM indexes across the whole genome, which will cover the entire genome when joined. These local FM indexes span two exons and are divided into three categories: long-anchor reads (16 bp in each exon), intermediate-anchored reads (8-15 bp in one exon) and short-anchored reads (1-7 bp in on exon). With this HISAT can efficiently address the spliced-alignment problem.

In order to construct these different indexes, HISAT is divided into three separate steps. The first step is used to provide HISAT with a GTF genome annotation file. For this, the GTF file has to be processed with the python script `extract_splice_sites.py` which is included in the HISAT package. The resulting file is then used to align reads with small anchors. Additionally a HISAT index has to be constructed with the hisat-build script. It generates a Bowtie2 like index from the genome FASTA file. Afterwards the files constructed in these two steps can be applied to the HISAT alignment tool when processing RNA-seq FASTQ read files. The full workflow is presented in Figure 3.3.

copy genome file:

```
        cp data/<Genome File> indexes/BOWTIE_index/<Genome File>
```

Bowtie2 index:

```
        bowtie2-build -f data/<Genome File> indexes/BOWTIE_index/
        <Genome Name>
```

TopHat2 index:

```
        tophat2 -G data/<GTF File> --transcriptome-index indexes/
        TOPHAT_index/<Genome Name> indexes/BOWTIE_index/
        <Genome Name>
```

TopHat2 alignment:

```
        tophat2 --min-intron-length 20 --max-intron-length 500000
        --max-multihits 100 --mate-inner-dist 500
        --read-mismatches 10 --read-gap-length 10 --read-edit-dist
        10 --microexon-search --keep-fasta-order --num-threads 6
        --output-dir alignments/<Dataset Name>/TOPHAT_align/
        <Sample Name> --transcriptome-index indexes/TOPHAT_index/
        <Genome Name> indexes/BOWTIE_index/<Genome Name>
        fastq_data/<FASTQ File 1> fastq_data/<FASTQ File 2>
```

Tophat-recondition fix:

```
        python tophat-recondition.py -q alignments/<Dataset Name>/
        TOPHAT_align/<Sample Name>
```

samtools header extraction:

```
        samtools view -H alignments/<Dataset Name>/TOPHAT_align/
        <Sample Name>/unmapped_fixup.bam > alignments/
        <Dataset Name>/TOPHAT_align/<Sample Name>/
        unmapped_fixup-header.sam
```

samtools merge:

```
        samtools merge -h alignments/<Dataset Name>/TOPHAT_align/
        <Sample Name>/unmapped_fixup-header.sam alignments/
        <Dataset Name>/TOPHAT_align/<Sample Name>/
        merged_hits-unsorted.bam alignments/<Dataset Name>/
        TOPHAT_align/<Sample Name>/accepted_hits.bam alignments/
        <Dataset Name>/TOPHAT_align/<Sample Name>/
        unmapped_fixup.bam
```

samtools sort:

```
        samtools sort -n -m 20000000000 alignments/<Dataset Name>/
        TOPHAT_align/<Sample Name>/merged_hits-unsorted.bam
        alignments/<Dataset Name>/TOPHAT_align/<Sample Name>
```

FIGURE 3.2: Alignment workflow of TopHat2.

3.2.3 STAR

The STAR (Spliced Transcripts Alignment to a Reference) software [6] was developed to successfully align a vast number of RNA-seq reads to a reference genome

HISAT splice sites:
```
python extract_splice_sites.py data/<GTF File> indexes/
HISAT_index/<Genome Name>.splice
```
HISAT index:
```
hisat-build -f data/<Genome File> indexes/HISAT_index/
<Genome Name>
```
total mate length:
```
MATE_TOTAL_LENGTH=‘expr MATE_DIST_OPT + READ_LENGTH
+ READ_LENGTH‘
```
HISAT alignment:
```
hisat --min-intronlen 20 --max-intronlen 500000 -k 100
--maxins $MATE_TOTAL_LENGTH --phred33 --reorder --threads
6 -S alignments/<Dataset Name>/HISAT_align/<Sample Name>/
<Sample Name>.sam -x indexes/HISAT_index/<Genome Name>
--known-splicesite-infile indexes/HISAT_index/
<Genome Name>.splice -1 fastq_data/<FASTQ File 1> -2
fastq_data/<FASTQ File 2>
```
samtools sort:
```
samtools view -Shb alignments/Dataset Name/HISAT_align/
<Sample Name>/<Sample Name>.sam | samtools sort -n -m
20000000000 - alignments/<Dataset Name>/HISAT_align/
<Sample Name>
```

FIGURE 3.3: Alignment workflow of HISAT.

with the focus on a high mapping speed while improving alignment sensitivity and precision. It is based on a novel heretofore unused RNA-seq alignment algorithm which was designed to address the many problems of RNA-seq read mapping. The algorithm features two major steps, a seed searching and a clustering, stitching and scoring step.

In the first step, the seed search step, STAR sequentially searches for a Maximal Mappable Prefix (MMP). In contrast to other alignment tools [mummer and Mauve], STAR uses the sequential search of the MMP only on the unmapped part of the read, which gives STAR a substantial advantage in mapping speed. Hereby STAR can easily find splice junctions and use the MMPs as anchors in the genome to mark mismatches and indels.

The created seeds are then clustered together and stitch combinations are created and scored. Afterwards, the stitched combination with the highest score is then chosen as the best alignment. With this STAR can, without bias detect canonical junctions, non-canonical splices and chimeric (fusion) transcripts.

To perform an alignment, first a STAR index from the genome reference FASTA and the genome GTF annotation file, has to be created. This index can then be used in the STAR alignment step in combination with the FASTQ RNA-seq read files to generate the alignments. The full workflow is presented in Figure 3.4.

```
STAR index:
        STAR --runMode genomeGenerate --runThreadN 4 --genomeDir
        indexes/STAR_index --genomeFastaFiles data/<Genome File>
        --sjdbGTFfile data/<GTF File> --sjdbOverhang 99
STAR alignment:
        STAR --genomeLoad NoSharedMemory --alignIntronMin 20
        --alignIntronMax 500000 --outFilterMultimapNmax 100
        --alignMatesGapMax 500 --outFilterMismatchNmax 10
        --outFilterMismatchNoverLmax 0.3 --outSAMorder
        PairedKeepInputOrder --runThreadN 6 --outFileNamePrefix
        alignments/<Dataset Name>/STAR_align/<Sample Name>/
        <Sample Name> --genomeDir indexes/STAR_index --readFilesIn
        fastq_data/<FASTQ File 1> fastq_data/<FASTQ File 2>
samtools sort:
        samtools view -Shb alignments/<Dataset Name>/STAR_align/
        <Sample Name>/<Sample Name>Aligned.out.sam | samtools sort
        -n -m 20000000000 - alignments/<Dataset Name>/STAR_align/
        <Sample Name>
```

FIGURE 3.4: Alignment workflow of STAR.

3.2.4 Bowtie2

Bowtie2 [7] is a fast RNA-seq aligner with a combination of high speed, sensitivity and accuracy by combining the full-text minute index [23] with hardware-accelerated dynamic programming algorithms. With the extension of the full-text minute index, which is also used in Bowtie [24], Bowtie2 is able to find gapped alignments. The algorithm is split into an ungapped seed-finding part using the full-text minute index and a gapped finding part which is based on dynamic programming and benefits from single-instruction multiple-data (SIMD) parallel processing.

To process a read, Bowtie2 first extracts substrings from the read and its complement which are then aligned without gaps by the full-text minute index approach.

Afterwards the positions of the substrings are then determined from the index and extended into full alignments by using SIMD-accelerated dynamic programming.

To use Bowtie2 for transcriptome alignments, first a Bowtie2 index has to be created from the transcriptome FASTA file and the transcriptome FASTA file has to be transferred to the Bowtie2 index folder. Afterwards, an alignment can be created with the Bowtie2 index and the RNA-seq read FASTQ files. In this work, Bowtie2 was only used to create transcript alignments for BitSeq [13] and RSEM [14].

Because RSEM only worked with a specific setting and not tolerating various alignments, two different Bowtie2 transcript alignments had to be constructed. The BitSeq Bowtie2 alignment only uses the standard settings and the ones which were also used for the other alignments. The RSEM Bowtie2 alignment uses settings which are described in the RSEM manual. The full workflow is presented in Figure 3.4.

copy transcriptome file:
```
        cp data/<Transcriptome File> indexes/BWTRS_index/
        <Genome Name>
```
Bowtie2 index:
```
        bowtie2-build -f data/<Transcriptome File> indexes/
        BWTRS_index/<Genome Name>
```
total mate length:
```
        MATE_TOTAL_LENGTH=‘expr MATE_DIST_OPT + READ_LENGTH
        + READ_LENGTH‘
```
Bowtie2 alignment (BitSeq):
```
        bowtie2 --maxins $MATE_TOTAL_LENGTH --threads 4 -q -x
        indexes/BWTRS_index/<Genome Name> -1 fastq_data/
        <FASTQ File 1> -2 fastq_data/<FASTQ File 2> -S alignments/
        <Dataset Name>/BWTRS_align/<Sample Name>/<Sample Name>.sam
```
samtools sort (BitSeq):
```
        samtools view -Shb alignments/<Dataset Name>/BWTRS_align/
        <Sample Name>/<Sample Name>.sam | samtools sort -n -m
        20000000000 - alignments/<Dataset Name>/BWTRS_align/
        <Sample Name>
```
Bowtie2 alignment (RSEM):
```
        bowtie2 --maxins $MATE_TOTAL_LENGTH --threads 4
        --no-discordant --no-mixed --sensitive --dpad 0 --gbar
        99999999 --mp 1,1 --np 1 --score-min L,0,-0.1 -q -x
        indexes/BWTRS_index/<Genome Name> -1 fastq_data/
        <FASTQ File 1> -2 fastq_data/<FASTQ File 2> -S alignments/
        <Dataset Name>/BRSEM_align/<Sample Name>/<Sample Name>.sam
```
samtools sort (RSEM):
```
        samtools view -Shb alignments/<Dataset Name>/BRSEM_align/
        <Sample Name>/<Sample Name>.sam | samtools sort -n -m
        20000000000 - alignments/<Dataset Name>/BRSEM_align/
        <Sample Name>
```

FIGURE 3.5: Alignment workflow of Bowtie2.

Chapter 4

Quantifier

4.1 Settings

All quantifiers were applied using the default parameters if possible. The quantifier htseq-count [9], featureCounts [10], EQP-QM [8], Cufflinks [11] and FluxCapacitor [12] were all used in combination with HISAT [5], TopHat2 [4] and STAR [6] alignments. FeatureCounts and EQP-QM were also used in the different modes described below.

4.2 Functionality

4.2.1 htseq-count

The htseq-count [9] quantification script is part of the HTSeq package and counts the reads overlapping with the respective exon. The focus of htseq-count lays on differential expression analysis and discards ambiguously mapped genes for this reason. Therefore reads mapped to multiple positions or genes are not counted.

Htseq-count consists of three different modes which can be used to better adjust reads overlapping with multiple exons. The `union` option counts all reads aligning to only one gene and discards genes slightly overlapping with other genes. The `intersection_strict` option counts only reads completely mapped to a gene. It discards reads with gaps, but includes reads which are completely mapped to one gene while overlapping to another. The last mode, `intersection_nonempty`, works like the union option, except it also allows the overlapping to another gene from the `intersection_strict` option. Reads completely contained in two genes are discarded in all three modes. All of these modes were evaluated in this work. The htseq-count tool accepts alignments in SAM format and further needs the GTF annotation file to successfully count the mapping reads. The full workflow is presented in Figure 4.1.

```
samtools view:
       samtools view -h alignments/<Dataset Name>/
       <Aligner Name>_align/<Sample Name>.bam > alignments/
       <Dataset Name>/<Aligner Name>_align/
       <Sample Name>.bam.tmp.sam
htseq-count quantification:
       htseq-count --stranded no --type exon --idattr gene_id
       --mode intersection-nonempty --samout quantifications/
       <Dataset Name>/HTSEQ_quant/<Quantification Name>/
       <Quantification Name> --order name --format bam
       alignments/<Dataset Name>/<Aligner Name>_align/
       <Sample Name>.bam.tmp.sam data/<GTF File> >
       quantifications/<Dataset Name>/HTSEQ_quant/
       <Quantification Name>/<Quantification Name>.quant
remove SAM file:
       rm -f alignments/<Dataset Name>/<Aligner Name>_align/
       <Sample Name>.bam.tmp.sam
copy count file:
       cp quantifications/<Dataset Name>/HTSEQ_quant/
       <Quantification Name>/<Quantification Name>.quant
       quantifications/<Dataset Name>/HTSEQ_quant/
       <Quantification Name>.cnt
```

Figure 4.1: Quantification workflow of htseq-count.

4.2.2 featureCounts

The featureCounts [10] quantifier classifies entries in the GTF annotation file as features and meta-features. A feature is simply one entry, e.g. exon, and a meta-feature contains all features having the same `gene_id` attribute. A count can be performed on feature or meta-feature level. To assign a read to a feature, featureCounts first compares every base in the read with the region on the genome contained in each feature. A read is accepted and assigned to a feature if an overlap is found between the read and the feature. Different mismatches, like insertions, deletions, exon-exon junctions and fusions are permitted.

FeatureCounts also by default does not count multi-mapping reads, like htseq-count, but has the option to also count these reads. Because of featureCounts overcounting reads by assigning multi-mapping reads to all features when using this option, only the default unambiguous version was used in this work. To perform a quantification, featureCounts simply needs the GTF annotation file and the alignment file. The full workflow is presented in Figure 4.2.

```
featureCounts quantification:
        featureCounts -t exon -g gene_id -s 0 -S fr --donotsort -p
        -a data/<GTF File> -o quantifications/<Dataset Name>/
        FEATC_quant/<Quantification Name>/
        <Quantification Name>.quant alignments/<Dataset Name>/
        <Aligner Name>_align/<Sample Name>.bam
copy count file:
        cp quantifications/<Dataset Name>/FEATC_quant/
        <Quantification Name>/<Quantification Name>.quant
        quantifications/<Dataset Name>/FEATC_quant/
        <Quantification Name>.cnt
```

FIGURE 4.2: Quantification workflow of featureCounts.

4.2.3 EQP-QM

EQP-QM (exon quantification pipeline - quantification module) is the quantification module of EQP [8]. To count the aligned reads, EQP-QM first calculates weights for all reads by intersecting the alignments with the transcript, genome and junction coordinates from the GTF file. This is done via BEDTools [25], for which the BAM alignment file and GTF annotation file are first transformed into BED format. A read from the intersection file is then only counted if the read is completely contained within the exon boundaries and therefore does not overlap with an intron. This assures, that only completely matured expressions are counted. EQP-QM has the option to produce gene, exon and junction counts from alignment files created by splice-aware alignment tools.

As an option EQP-QM can quantify genes ambiguously and unambiguously, both options were used in this work. To quantify an alignment, EQP-QM first needs to construct an index from the genome annotation GTF file. This was done with the `eqp-setup.sh` module. Afterwards the BAM alignment files can be quantified in combination with the generated index by the `eqp-quantify.sh` script. The full workflow is presented in Figure 4.3.

```
EQP-QM index:
       eqp-setup.sh data/<GTF File> indexes/EQPQM_index
EQP-quantification:
       eqp-quantify.sh -g --nosort -o <Quantification Name> -d
       indexes/EQPQM_index quantifications/<Dataset Name>/
       EQPQM_quant/<Quantification Name> alignments/
       <Dataset Name>/<Aligner Name>_align/<Sample Name>.bam
copy count file:
       cp quantifications/<Dataset Name>/EQPQM_quant/
       <Quantification Name>/<Quantification Name>-gene.cnt
       quantifications/<Dataset Name>/EQPQM_quant/
       <Quantification Name>.cnt
```

Figure 4.3: Quantification workflow of EQP-QM.

4.2.4 Cufflinks

Cufflinks [11, 20] was originally designed for finding novel transcripts and splicing. It first composes a parsimonious set of transcripts from the RNA-seq alignments and the estimate of the relative transcript abundance of these sets. The assemblage of the parsimonious transcripts is done via an algorithm which only reports a minimum number of transfrags (full length transcript fragments) which is needed to justify all splicing occurrences. Afterwards, in the expression step, the transfrags are quantified and background or artificial transfrags are filtered. This is done via a strict statistical model. Cufflinks reports the estimated transcript abundances via FPKM (expected fragments per kilobase of transcript per million fragments mapped).

To perform a quantification with Cufflinks, the script needs a sorted BAM alignment file, as well as the genome GTF annotation file. Cufflinks then reports its transcript results as `isoforms.fpkm_tracking`. To provide a standardized evaluation of all quantification tools, the total read counts were then extracted from the Cufflinks standard output and used to recalculate the count values from the FPKM values. The transcript counts were then later combined into gene counts. The full workflow is presented in Figure 4.4.

```
samtools sort:
        samtools sort -m 20000000000 alignments/<Dataset Name>/
        <Aligner Name>_align/<Sample Name>.bam quantifications/
        <Dataset Name>/CUFFL_quant/<Quantification Name>/
        <Quantification Name>
Cufflinks quantification:
        Cufflinks --library-type fr-unstranded --no-update-check
        --quiet -G data/<GTF File> -o quantifications/
        <Dataset Name>/CUFFL_quant/<Quantification Name>
        --num-threads 4 quantifications/<Dataset Name>/
        CUFFL_quant/<Quantification Name>/
        <Quantification Name>.bam
copy count file:
        cp quantifications/<Dataset Name>/CUFFL_quant/
        <Quantification Name>/isoforms.fpkm_tracking
        quantifications/<Dataset Name>/CUFFL_quant/
        <Quantification Name>.cnt
remove sorted alignment file:
        rm -f quantifications/<Dataset Name>/CUFFL_quant/
        <Quantification Name>/<Quantification Name>.fpkm
get total counts:
        grep 'Normalized Map Mass' qsub_data/CUFFL_quant-*.qout |
        sed 's/>.*:\s//' | sed 's/\.\..*qsub_data\///' | sed
        's/\(.*\)\-\(.*\)_\(.*\)_\(.*\)\.qout:/\1\t\2\t\3\t\4\t/' |
        sort -k2,2 -k3,3n > data/cuff_read_mass.data
transform fpkm to count:
        python get_cuff_counts.py data/cuff_read_mass.data
        quantifications/CUFFL_quant fpkm cnt
```

FIGURE 4.4: Quantification workflow of Cufflinks.

4.2.5 FluxCapacitor

The FluxCapacitor [12] transcript abundance quantification script was developed to investigate annotated alternatively spliced transcripts. The algorithm uses a modified version of a splicing graph to distribute reads mapping to exonic area between the transcripts. To construct this graph, first read distribution profiles from non-overlapping transcripts are generated. These profiles then operate as capacities for the edges of the flow network graph, together with a deviation factor for statistical interference. The resulting graph then represents a linear optimization problem and can be solved with a linear solver. Transcripts are quantified by their resulting edge value.

To use the FluxCapacitor for quantification, the genome annotation GTF file and the alignment BAM file have to be sorted. The BAM file can be sorted via samtools, for the GTF file, the FluxCapacitor software provides an extra GTF sorting option. The FluxCapacitor capacitor module then takes these sorted files as input to generate the transcript abundance quantification. The full workflow is presented in Figure 4.5.

4.2.6 BitSeq

To provide transcript expression estimations, the BitSeq (Bayesian interference of transcripts from sequencing data) [13] quantification algorithm uses a generative model which represents every read individually according to the relative abundance of transcript fragments and with a noise parameter. An alignment is then represented as the probability that a read aligns to a specific transcript, including indels. Afterwards, a Markov chain Monte Carlo algorithm for Bayesian inference is used to create samples from the posterior probability distribution. The MCMC is implemented via a collapsed Gibbs sampler. The samples provide an estimate of the transcript abundances. The first step in applying BitSeq on RNA-seq reads is to generate the alignment probabilities with the parseAlignment script. The script uses the FASTA transcriptome annotation file and the transcript alignment

```
FluxCapacitor sort GTF:
        flux-capacitor -t sortGTF --force -i data/<GTF File> -o
        indexes/FLXCP_index/<GTF File>
samtools sort:
        samtools sort -m 20000000000 alignments/<Dataset Name>/
        <Aligner Name>_align/<Sample Name>.bam quantifications/
        <Dataset Name>/FLXCP_quant/<Quantification Name>/
        <Quantification Name>
FluxCapacitor quantification:
        flux-capacitor -t capacitor --force --tmp-dir
        quantifications/<Dataset Name>/FLXCP_quant/
        <Quantification Name> -o quantifications/<Dataset Name>/
        FLXCP_quant/<Quantification Name>/
        <Quantification Name>.quant -i quantifications/
        <Dataset Name>/FLXCP_quant/<Quantification Name>/
        <Quantification Name>.bam -a indexes/FLXCP_index/
        <GTF File> --minilen 20
copy count file:
        cp quantifications/<Dataset Name>/FLXCP_quant/
        <Quantification Name>/<Quantification Name>.quant
        quantifications/<Dataset Name>/FLXCP_quant/
        <Quantification Name>.cnt
remove sorted alignment file:
        rm -f quantifications/<Dataset Name>/FLXCP_quant/
        <Quantification Name>/<Quantification Name>.bam
```

FIGURE 4.5: Quantification workflow of FluxCapacitor.

BAM files, generated with Bowtie2 [7], as input.

The resulting probability file can then be used in the estimateExpression step to calculate the mean transcript expression levels represented as thetaMeans. Afterwards the included python script `getCounts.py` can be used in combination with the probability file from the first script to generate the transcript counts. The resulting count file can then be combined with the information file generated in the first script to combine transcript names and counts. The full workflow is presented in Figure 4.6.

4.2.7 RSEM

RSEM (RNA-seq by Expectation Maximization) [14] is an RNA-seq transcript abundance estimation assessment tool which is capable of processing reads that

BitSeq alignment probabilities:
```
        parseAlignment --outFile=quantifications/<Dataset Name>/
        BTSEQ_quant/<Quantification Name>/
        <Quantification Name>.prob --trSeqFile=data/
        <Transcriptome File> --trInfoFile=quantifications/
        <Dataset Name>/BTSEQ_quant/<Quantification Name>/
        <Quantification Name>.info --procN=4 --unstranded
        --verbose alignments/<Dataset Name>/BWTRS_align/
        <Sample Name>.bam
```
BitSeq estimate expressions:
```
        estimateExpression --outPrefix=quantifications/
        <Dataset Name>/BTSEQ_quant/<Quantification Name>/
        <Quantification Name> --outType=theta --procN=4 --seed=0
        --verbose quantifications/<Dataset Name>/BTSEQ_quant/
        <Quantification Name>/<Quantification Name>.prob
```
BitSeq generate counts:
```
        python getCounts.py -o quantifications/<Dataset Name>/
        BTSEQ_quant/<Quantification Name>/
        <Quantification Name>.count -p quantifications/
        <Dataset Name>/BTSEQ_quant/<Quantification Name>
        quantifications/<Dataset Name>/BTSEQ_quant/
        <Quantification Name>/<Quantification Name>.thetaMeans
```
add transcript names:
```
        tail -n +2 quantifications/<Dataset Name>/BTSEQ_quant/
        <Quantification Name>/<Quantification Name>.info | cut
        -d" " -f2 | paste -d" " - quantifications/<Dataset Name>/
        BTSEQ_quant/<Quantification Name>/
        <Quantification Name>.count > quantifications/
        <Dataset Name>/BTSEQ_quant/<Quantification Name>.cnt
```

Figure 4.6: Quantification workflow of BitSeq.

map ambiguously between multiple genes and between multiple isoforms. A key feature of RSEM is, that it allows RNA-seq quantification without the use of a FASTA reference genome file. For this, RSEM first needs to prepare the reference sequences from the FASTA reference transcriptome file, to create a set of preprocessed transcript sequences for a more precise read analysis. Afterwards RSEM uses the Bowtie2 alignment program with parameters precisely adjusted for the later quantification with RSEM. The quantification is then done with the use of an extended previously designed directed graph model [32]. RSEM then calculates maximum likelihood abundance estimates with the Expectation-Maximization algorithm from the previously described model.

The first step in assessing the transcript abundance estimates with RSEM is to prepare the reference sequences with the `rsem-prepare-reference` script. In the second step, `rsem-calculate-expression` can be applied on transcript alignment BAM files in combination with the prepared reference sequences to estimate the transcript abundances. The Bowtie2 parameter values were set exactly like the values that RSEM would have used to make the Bowtie2 alignments. The reason for this is, that RSEM is unable to process indel, local or discordant alignments. The full workflow is presented in Figure 4.7.

RSEM index:
```
      rsem-prepare-reference data/<Transcriptome File> indexes/
      RSEM_index/<Genome Name>
```
RSEM quantification:
```
      rsem-calculate-expression --seed 0 --alignments
      --paired-end alignments/<Dataset Name>/BRSEM_align/
      <Sample Name>.bam indexes/RSEM_index/<Genome Name>
      quantifications/<Dataset Name>/RSEM_quant/
      <Quantification Name>/<Quantification Name>
```
copy count file:
```
      cp quantifications/<Dataset Name>/RSEM_quant/
      <Quantification Name>/<Quantification Name>.genes.results
      quantifications/<Dataset Name>/RSEM_quant/
      <Quantification Name>.cnt
```

FIGURE 4.7: Quantification workflow of RSEM.

4.2.8 kallisto

The kallisto [15] RNA-seq quantification tool works without a preceding alignment from an RNA-seq aligner. Kallisto achieves this by pseudoaligning reads to a reference, which results in a list of transcripts for each read, from which the read could have originated. To quantify the reads, kallisto first constructs a colored de Bruijn graph from the transcriptome. Each node corresponds to a k-mer which is marked with all the corresponding transcripts. In this contigs are defined as paths in the graph, having identical marks. A contig therefore corresponds to a transcript. Each k-mer is then mapped within a hash table to the contig it belongs to. In the second step, kallisto pseudoaligns the reads via hashing against the constructed T-DBG index. The transcript abundances are then quantified from these pseudoalignments by a likelihood function for RNA-seq which is iteratively optimized using an expectation-maximization algorithm.

To generate an alignment, the kallisto index can be constructed by applying the kallisto index module to a transcriptome FASTA file. Thereafter, the resulting index file can be used in combination with the RNA-seq FASTQ files to create the transcript abundance estimations. The full workflow is presented in Figure 4.8.

```
kallisto index:
      kallisto index -i indexes/KLLST_index/<Genome Name>.idx
      data/<Transcriptome File>
kallisto quantification:
      kallisto quant -i indexes/KLLST_index/<Genome Name>.idx -o
      quantifications/<Dataset Name>/KLLST_quant/
      <Quantification Name> --seed=0 --plaintext --threads=4
      fastq_data/<FASTQ File 1> fastq_data/<FASTQ File 2>
copy count file:
      cp quantifications/<Dataset Name>/KLLST_quant/
      <Quantification Name>/abundance.tsv quantifications/
      <Dataset Name>/KLLST_quant/<Quantification Name>.cnt
```

Figure 4.8: Quantification workflow of kallisto.

Chapter 5

Data

5.1 Genome Annotation and Reference

The FASTA genome reference file and the GTF genome annotation file for this work originated from the Ensemble genomic interpretation system [21]. It is the Ensemble 78 version based on the GRCh38 genome build. The chromosome comment lines in the genome FASTA file had to be slightly modified because of Bowtie2 [7] only accepting genome and GTF files with identical identifiers. To utilize this data for the simulations in the Flux Simulator [16], single chromosome FASTA files had to be created from this genome reference file.

Because of the Flux Simulator trying to utilize all entries in the GTF genome annotation file, a slightly modified version of this GTF file only containing annotations which are also in the FASTA genome reference file, had to be generated. Furthermore, a combination of the FASTA genome cDNA and ncRNA files was created for use in the transcriptome alignments. The files can be found in the supplementary section of this work.

5.2 SEQC - RNA-Seq Illumina

The RNA-Seq data used in this thesis consists of two different sample types. Both sample types were created in the context of the RNA Sequencing Quality Control (SEQC) Consortium [33]. The first experimental RNA-Seq data set originates from the Universal Human Reference RNA (UHRR) and is denoted SEQC-A. The second experimental RNA-Seq data set consists of samples from the Human Brain Reference (HBR) and is labeled SEQC-B. Both RNA-Seq data sets consist of 100bp long paired-end reads with four technical replicates and were part of GEO series GSE47774. Each replicate was sequenced on eight multiplexed lanes of an Illumina HiSeq 2000. The resulting FASTQ files had around 10M reads each and a total of 64 FASTQ file pairs with more than 320M reads per sample were created.

The used UHRR and HBR RNA-Seq benchmarking data sets are especially valuable for tasks like the assessment of RNA-Seq quantification tools. These utilized samples were extensively analysed in the MicroArray and the RNA Sequencing Quality Control projects [34] and also have corresponding TaqMan and Bio-Rad data.

5.3 SEQS - Flux Simulator

5.3.1 Samples

This data set was created with the Flux Simulator [16] and consists of two sample types SEQS-A and SEQS-B. A total of 8 replicas were simulated with 10M reads for each replica, to be able to compare the SEQC data set with the SEQS data set. The expression values for each of the sets was based on the average quantification results from the Cufflinks [20] quantification for the respective SEQC-A and SEQC-B set. This count data was slightly smoothed out before creating the average expression values, because of Cufflinks sometimes over-counting reads. This was done mainly to prevent counts occurring excessively only in one of the 32 replicas from a sample type, which happened occasionally for Cufflinks in combination with Tophat2 [4].

5.3.2 Functionality

The Flux Simulator that was used to simulate the RNA-Seq data sets, is a simulation pipeline consisting of various universally applicable models. The models cover RNA-Seq components like reverse transcription, fragmentation, adapter ligation, PCR amplification, gel segregation and sequencing. The Flux Simulator first creates randomized expression levels for transcriptomes according to general laws of gene expression from the input FASTA reference genome and the input GTF annotation genome. The library step is followed by the fragmentation step and the reverse transcription step which has three optional randomization methods. In this work, the library was created using a uniformal random fragmentation option. In the last step, the Flux Simulator simulates read sequences together with the genomic mappings.

To simulate the data sets according to the two different sample sets, the first step is to generate a model profile file with the Flux Simulator expression step. The expression values in this pattern profile file were then modified with the `make_flx_conf.py` python script to create profile files for SEQS-A and SEQS-B, according to the respective expressions of SEQC-A and SEQC-B from the Cufflinks quantification files. These were then applied to the Flux Simulator library step to create a library for each data set. The resulting profile files were then duplicated to simulate multiple different RNA-seq FASTQ read data files with the Flux Simulator Simulation step. A sample workflow is shown in Figure 5.1.

5.4 qPCR - Gold Standards

5.4.1 TaqMan

The TaqMan [33] qRT-PCR data, used in this work, consists of 1000 genes for both sample types, the UHRR and the HBR.

5.4.2 Bio-Rad

The Bio-Rad [33] qRT-PCR data consists of around 20K genes for each of the two sample sets, UHRR and HBR.

Flux Simulator initial expression values:
```
       flux-simulator -t simulator -x --threads 4 -p flx_sim/
       flx_sim.conf
```
create expression values:
```
       python make_flx_conf.py quantifications/SEQC/CUFFL_quant/
       cnt 1,2,3,4,5,6,7,8,17,18,19,20,21,22,23,24,33,34,35,36,
       37,38,39,40,49,50,51,52,53,54,55,56 flx_sim/flx_sim.pro
       flx_sim/flx_sim_A.pro > flx_sim/flx_sim_A.ref
       python make_flx_conf.py quantifications/SEQC/CUFFL_quant/
       cnt 9,10,11,12,13,14,15,16,25,26,27,28,29,30,31,32,41,42,
       43,44,45,46,47,48,57,58,59,60,61,62,63,64 flx_sim/
       flx_sim.pro flx_sim/flx_sim_B.pro > flx_sim/flx_sim_B.ref
```
create config files (A/B):
```
       sed -e 's/#PRO_FILE_NAME\t/PRO_FILE_NAME\t\/flx_sim\/
       flx_sim_<A/B>.pro/' -e 's/#TMP_DIR\t/TMP_DIR\t\/flx_sim\/
       tmp_<A/B>/' flx_sim/flx_sim.conf > flx_sim/
       flx_sim_<A/B>.conf
```
Flux Simulator library creation (A/B):
```
       flux-simulator -t simulator -l --threads 4 -p flx_sim/
       flx_sim_<A/B>.conf
```
make temporal directory:
```
       mkdir tmp_<A/B>_1
```
copy profile (A/B):
```
       cp flx_sim_<A/B>.pro flx_sim_<A/B>_<Simulation Number>.pro
```
create config files (A/B):
```
       sed -e 's/#LIB_FILE_NAME\t/LIB_FILE_NAME\t\/flx_sim\/
       flx_sim_<A/B>.lib/' -e 's/#PRO_FILE_NAME\t/
       PRO_FILE_NAME\t\/flx_sim\/
       flx_sim_<A/B>_<Simulation Number>.pro/' -e 's/#TMP_DIR\t/
       TMP_DIR\t\/flx_sim\/tmp_<A/B>_<Simulation Number>/'
       flx_sim/flx_sim.conf > flx_sim/
       flx_sim_<A/B>_<Simulation Number>.conf
```
Flux Simulator sequencing (A/B):
```
       flux-simulator -t simulator -s --threads 4 -p flx_sim/
       flx_sim_<A/B>_<Simulation Number>.conf
```
split FASTQ files:
```
       python split_flux_reads.py flx_sim/ flx_sim
       <Simulation Number> data/flux_simulations/
```
zip FASTQ files:
```
       gzip data/flux_simulations/flx_sim_*_*-*.fastq
```

FIGURE 5.1: Simulation workflow of Flux Simulator.

Chapter 6

Methods

6.1 Overview

The ARQ (Assessment and Comparison of RNA-seq Quantification Methods) pipeline was generated for the purpose of comparing multiple RNA-Seq quantification methods. It automates the process of generating different quantifications from different alignments of multiple samples. The `compare-counts.R` tool is designed in regard to the data structure resulting from running the `ARQ.py` python script and can be used for statistically analysing the counts produced. Each tool got assigned a unique naming string to make management between tools easier. The different setting shortcuts for tools are given in 6.1 and the different name shortcuts for tools are given in Figure 6.2 and .

`ambig`	`A`	EQP-QM ambiguous setting
`unambig`	`U`	EQP-QM unambiguous setting
`union`	`N`	htseq-count union setting
`istrict`	`S`	htseq-count intersection-strict setting
`inonempt`	`E`	htseq-count intersection-nonempty setting

Figure 6.1: List of tool setting strings and corresponding shortcut for the tables presented in the appendix.

`TOPHAT`	TopHat2 genome alignment
`HISAT`	HISAT genome alignment
`STAR`	STAR genome alignment
`BWTRS`	Bowtie2 genome transcript alignment
`BRSEM`	Bowtie2 genome transcript alignment for RSEM
`AFREE`	Alignment free
`SIMED`	Simulated
`QPCR`	qRT-PCR
`HTSEQ`	htseq-count quantification
`FEATC`	FeatureCounts quantification
`EQPQM`	EQP-QM quantification
`CUFFL`	Cufflinks quantification
`FLXCP`	FluxCapacitor quantification
`BTSEQ`	BitSeq quantification
`RSEM`	RSEM quantification
`KLLST`	kallisto quantification
`FLXSM`	Flux Simulator profile
`TQMAN`	TaqMan data
`BIORD`	Bio-Rad data

FIGURE 6.2: List of tool naming strings used in the ARQ tool. These are also used in the tables and figures of this thesis.

6.2 Create Counts

6.2.1 Functionality

The python tool ARQ that can be used for creating the count data is divided in several small scripts which are all accessed by the `ARQ.py` main script. Following this (Figure 6.3), is a short description of the functionality for each individual script.

`ARQ.py` needs a configuration file and a project name as input (Figure 6.4). The configuration file has to contain all tool, output and data paths, as well as the desired tool configuration. An example can be found in the supplementary. ARQ then first creates all indexes for the desired tool, continues with creating the desired alignments and afterwards all quantifications. The three steps are each parallelized via the UNIVA Grid Engine [22] for every sample and tool call. This implies that, presumed there is a sufficient amount of available cores and memory, all tools and samples in each of the steps will be processed simultaneously. This can be deactivated with the SINGLE_TOOL_JOB option. If this is deactivated, the script will still call all tools simultaneously, but only for one sample at a time. This can be used if there is not enough processing power available for all samples to run at once.

If ARQ is called together with the nohup command, the IGNORE_QS_ERROR has to be enabled and the USER_SET_OVER, USER_SET_SKIP, USER_SET_NEW parameters have to be set. Otherwise ARQ demands user input while running, which will not work in combination with the nohup command. The three parameters USER_SET_OVER, USER_SET_SKIP and USER_SET_NEW accept a combination of the form <tool_abbreviation>_index, <tool_abbreviation>_align and <tool_abbreviation>_quant, depending on the desired job. USER_SET_OVER will set all defined jobs to overwrite mode and every job which is already done will be deleted and recalled afterwards. USER_SET_SKIP will skip all defined jobs completely. USER_SET_NEW will only make jobs for each tool, which were not already processed.

6.2.2 Project Structure

ARQ creates its own project structure with folders for indexes, alignments, quantifications and other. It first creates a folder with the project name and inside all folders that are needed for the script. The indexes for the various tools can be found in the indexes folder. They are named <tool_name>_index (e.g. HISAT_index).

All alignments can be found in the alignments folder, in which there are folders for each new FASTQ dataset. A new dataset can be specified in the configuration file under DATA_SET_NAME. Inside the dataset folder there will be a folder for each alignment tool, named <aligner>_align (e.g. STAR_align). Alignment files will be named according to the name given in the FASTQ links file.

The quantifications can be found in the quantifications folder. Like in the alignment folder a folder with the dataset name will be generated. Inside this, a folder for each quantification tool will be made, named <quantifier>_quant

`ARQ.py`
: Main script for calling all intermediate steps. It accesses all other ARQ scripts and coordinates the construction of indexes, alignments and quantifications for all tools and data files.

`arq_dirs.py`
: Generates all main tool directories, needed for the call.

`arq_get.py`
: Contains definitions for processing the configuration data as well as the input FASTQ files and other several informations.

`arq_check.py`
: Validates the input files, including the configuration file and checks the progress of the project folder.

`arq_tools.py`
: Contains the configuration for all tools used in the `ARQ.py` count tool. The configuration for new tools can be added here. It also contains the basic bash script template which is used for the tool calls with the UNIVA Grid Engine.

`arq_qsub.py`
: Constructs, writes and calls the final bash scripts for all requested jobs.

`arq_read.py`
: Contains the help section of all tools.

`arq_helper.py`
: Contains various small functions.

`arq_debug.py`
: Script with debug functions.

FIGURE 6.3: Overview of the `ARQ.py` python script.

`ARQ.py` command line call:

```
python ARQ.py -p <Project Name> -c <Configuration File>
```

FIGURE 6.4: Command line call of the `ARQ.py` python script.

(e.g. `EQPQM_quant`). Quantifications inside are named according to the alignment they originated from, the reference number of the FASTQ file and a naming string which can be defined in the configuration option under `QUANT_NAME_LIST`. The `default` name string will be given when there is no naming string defined at `QUANT_NAME_LIST`.

A file named `<dataset>_fastq.ref` can be found inside the quantification dataset folder which contains all reference number to FASTQ mappings. Unzipped FASTQ files are stored in the `fastq_data` folder. The `qsub_data` folder contains all the bash scripts that have been submitted with the`qsub` command from the UNIVA Grid Engine as well as respective `<Qsub Name>.qout` files.

6.2.3 Tool Configuration

The command line configuration is described in `arq_tools.py`. This file can be used to add new tools or parameters to the ARQ program according to a specific template. The functionality of the template is described in the following part. The substitution parameters in the `bash_script` string will be substituted with the parameters inside the `bash_config` dictionary. After that a standard tool script will be constructed according to the `script_order` dictionary key inside the `job_config_data` dictionary. The script will combine the tool parameters in this order to create a command line call for the tool. The `script_options` key parameters will be inserted in the `options` part of the `script_order` list. This will be the template for one of the three bash script parts. `bash_main_script`, `bash_pre_script` and `bash_seq_script`.

The first one contains the main tool call with all requested options. The later two are prequel and sequel bash scripts. These can contain commands to prepare the data before the main tool call or refine it afterwards. The `&&` syntax was used as a new line for multiple commands or lines. Readability was preserved through separation of multiple commands by a `+` into multiple lines. The created script parts will then be substituted with the previous generated basic bash script. This approach is used in every tool. The final bash script for each individual call will then be generated by inserting the specific parameters from the `input_parameters` dictionary. The already existing tool settings can be used as a guideline.

6.2.4 Tool Parameters

The following is a guideline to add new parameters to the `ARQ.py` script which can then be used in the configuration file. A few parameters, that can be used when integrating new tools, are automatically generated and are described in Figure 6.5. Other parameters can simply be added into the `arq_tools.py` file, while obeying the following rules. New parameters have to be inserted into the `input_parameters` dictionary key of the the `return_dict`. To define the dependencies and the tool type, the new tool has to be inserted into the `possible_jobs` dictionary key of the `return_dict`. Finally, to define the possible job combinations, like which alignment can be used with which quantifier, the combination of both has to be inserted into the `quant_align` dictionary key of the `return_dict`.

```
main project path:
        <proj_dir_path>  = <proj_dir_prefix>/<p_name_suffix>/
path to the index directory:
        <index_dir_path> = <proj_dir_path>/indexes/<data_set_name>/
path to alignment directory:
        <align_dir_path> = <proj_dir_path>/alignments/
                           <data_set_name>/
path to quantification directory:
        <quant_dir_path> = <proj_dir_path>/quantifications/
                           <data_set_name>/
path to alignment:
        <comb_in_file>   = <proj_dir_path>/alignments/
                           <data_set_name>/<align_out_prefix>.bam
path to TOOL index directory (one automatically generated for each tool):
        <TOOL_index_dir> = <index_dir_path>/TOOL_index/
path to TOOL alignment directory (automatically generated for each tool):
        <TOOL_align_dir> = <index_dir_path>/TOOL_align/
path to alignment directory (changes according to each FASTQ file):
        <align_out_dir>  = <align_dir_path>/<TOOL_align>/
                           <align_out_prefix>/
path to quantifier directory (changes according to each alignment file):
        <quant_out_dir>  = <quant_dir_path>/<TOOL_quant>/
                           <quant_out_prefix>/
alignment prefix name (from the names inside the FASTQ links file):
        <align_out_prefix>
quantification prefix name (according to the alignment name):
        <quant_out_prefix>
first FASTQ file:
        <fastq1_file>
second FASTQ file:
        <fastq2_file>
```

Figure 6.5: Automatically defined parameters of the `ARQ.py` script.

6.3 Utility

The following are four small utility scripts which were made to process the Cufflinks [11] output data, extract some information used in the comparison script and to prepare input data for the Flux Simulator.

Almost all quantification tools are providing their output in a simple count format. This is not true with Cufflinks, it only represents its output as FPKM and having different output formats is inconvenient for comparison. The scripts `cuff_read_mass.sh` and `get_cuff_counts.py` were designed to overcome this problem. Together they can transform the Cufflinks output FPKM files into count files by reverse calculation.

The first file can extract the needed total read counts, which were used internally from Cufflinks to transform the counts into FPKM. The total counts are reported as standard output from Cufflinks and are therefore written into the respective `CUFFL_quant-<aligner>_<fq_ref_num>_default.qout` inside the `qsub_data` folder. The counts are denoted as Normalized Map Mass and can be extracted at once for all Cufflinks `<Qsub Name>.qout` files with this script. The script then outputs a tab separated file with the columns `CUFFL_quant`, `<aligner>`, `<fq_ref_num>`, `default`, `<total_count>`, which can be used as input for the second script. This uses the Cufflinks total count file generated from the previous script, the folder of the Cufflinks output files and the file extension for the input (`fpkm`) and the desired output (`cnt`) files as parameters. It will then generate Cufflinks counts and store them inside the input folder with the new file extension.

One of the other two scripts is `trans2gene.py` which generates a genome to transcript mapping file out of a GTF file. This mapping file will be used to change transcript to gene IDs inside the statistics script. The last script is `geno2chrom.py` which needs the genome reference FASTA file and an output folder as parameters and generates a new FASTA file for each chromosome according to the name of the chromosome. This chromosome FASTA files were then used in the Flux Simulatior simulations.

6.4 Simulations

The steps for the simulations made with the Flux Simulator [16] are described in the `flx_sim_qsub.sh` script and an example call is already described in Figure 15. The two small python scripts used in this workflow are `make_flux_conf.py` and `split_flux_reads.py`.

The first is used to process a flux.pro file generated in the Flux Simulator expression step. After the first call of the Flux Simulator expression step, this script can be used to replace the generated expressions with desired expressions from multiple Cufflinks count files. The input parameters are the Cufflinks count folder, the file extensions, a list of reference numbers, which indicate which files will be used for the new expressions, the input Flux Simulator profile file and an output profile file.

Cufflinks sometimes randomly produces relatively high counts. These counts will be smoothed according to the following criteria. If a count for a specific transcript is more than 10% of the sum of this transcript over all samples and the count is more than 0.1% of the sum of all transcripts in this sample, it will be smoothed. It is important to have a sufficient number of samples for the first smoothing argument. For the smoothing, the probable erroneous count will be subtracted from the total count sum over all samples for this transcript. This mainly removes counts with high values which are present in only one of all sample replicates. If there are two or more samples with very high counts, only the highest one will be subtracted.

The second script, `split_flux_reads.py` is used to split an output FASTQ read file. All from the Flux Simulator generated paired-end FASTQ files consist of consecutive paired reads. These can be extracted into separate files with this script. The script uses the folder with the Flux Simulator FASTQ reads, the prefix of the reads, the file numbers and an output folder for the split reads. Therefore the names of the reads have to be in the following format: `<prefix>_<A/B>_<number>` (e.g. `flux_sim_A_1.fastq`).

The tool will then search for all A,B and number combinations with the provided prefix and will generate `<prefix>_<A/B>_<number>_<1/2>.fastq` files (e.g. `flux_sim_A_1_1.fastq` and `flux_sim_A_1_2.fastq`). Also the last 4 characters of the name of each entry will be removed, because of the Flux Simulator distinguishing the paired-end reads with `:S/1` and `:A/2`.

One script not used in the Flux Simulator workflow, but which is important for the simulations itself is the `prep_flux_gtf.py` script. This tool modifies a genome annotation GTF file for usage in the Flux Simulator. It removes all genome entries from the GTF file that have no corresponding genome entry in the FASTA reference genome file and requires a FASTA file, an input GTF file and an output GTF file as input.

6.5 Compare Counts

The tool to compare the counts produced with the `ARQ.py` script consists of two parts. One utility script, the `util-lib.R`, which contains all functions for the statistical analysis and plotting and the `compare-counts.R` main script. The main script can directly access the `ARQ.py` folder structure and extract reads with the help of a configuration file. The preferred analysis steps and file paths can be defined at the beginning of the script. It has to be provided with a tab separated configuration file with the following columns `<aligner>`, `<quantifier>`, `<naming_string>`, `<combinations>` and `<count type>`. An example can be found in the supplementary. The script will then search for quantifications inside the provided paths according to the configuration file. When setting `read.results.bool`, `make.txt.bool` and `write.count.matrix.bool` to `TRUE`, the quantifications will be read and a count matrix for each quantifier and alignment combination will be generated. This matrix can afterwards be simply scanned when setting the `read.cnt.matrix` parameter to `TRUE`.

Chapter 7

Literature

Some recent studies [1–3] have already addressed the assessment of RNA-Seq quantification tools. Every one of them used a different strategy to achieve this, which demonstrates the difficulty of choosing the right evaluation technique to approach this subject. This is due to the shortage of precise and sensitive methods to evaluate this type of data. Resulting from the lack of an underlying ground truth for the data. Comparing the performance of solutions from different algorithms is rather difficult, when the true values are not know. One can only try to find a suitable metric to compare the tools according to variation and similar statistical measurements. Obviously this is not true with simulated data sets, but most of the time, simulations do not reflect the real nature of things. Some of the following projects have applied metrics and other comparisons to overcome the mentioned problems.

Norel et al. [36] described another problem resulting from this lack of good metrics, the phenomenon of the self assessment trap [36]. This happens when authors of new methods choose a scoring scheme favouring their own methods and producing results proclaiming superiority for their new designed tool. Teng et al. [1] approached this problem with the construction of two specific data sets. One experimental data set derived from a microarray experiment, which compared two cell lines to define true biological differences. The other data set was artificially designed through altering sample data from another project [40]. The samples were computationally modified to create two different sets with differently expressed genes between the two sets. They then applied these data sets to the aligner STAR [6], TopHat2 [4] and Bowtie2 [7] and the quantifier Cufflinks [11], eXpress [47], FluxCapacitor [12], kallisto [15], RSEM [14], Sailfish [35] and Salmon [38]. The experiment therefore focused on the comparison of transcript abundance quantification scripts.

The result of their experiments was, that none of the tools really performed well with eXpress and FluxCapacitor being a little bit worse and RSEM a little bit better. A notable part in the paper is their evaluation of the correlation measurement, which became a popular measurement in life sciences [39, 41, 42]. They demonstrated the case of correlation not being the most revealing measurement when trying to represent reproducibility and argued that a metric based on the standard deviation is much more applicable.

This issue with correlation was also observed by Fonseca et al. [2], where they observed a high median Spearman correlation of their produced data. In this context they also examined individual pairs of outcomes and came to the conclusion that the gene expression estimates sometimes vary vastly. They compared on four experimental data sets and one simulated data set with different characteristics, on fifty gene profiling pipelines. The pipelines consisted of different combinations, modes and versions of the alignment tools TopHat, Star, Bowtie, Osa [43], GSNAP [44], Smalt [45], and the quantifiers HTSeq [9], Cufflinks and FluxCapacitor. They used the simulated data, generated with the Flux Simulator, to explore the impact from alignment tools on the error rate of the pipelines.

One observation was, that the error was increasing with read length when using unspliced aligners like Bowtie and BWA. The spliced aligners Tophat, OSA and STAR produced similar low median error rates, which suggests that they have a similar quality. They further came to the overall conclusion, that the choice of the right spliced aligner does not make much of a big difference because they reached the peak of their improvements when profiling genes. On the other hand the quantification tools can make a difference in the analysis of RNA-Seq data.

Kanitz el al. [3] also evaluated RNA-Seq estimation pipelines in the context of runtime and memory consumption. They evaluated the transcript estimation abundance methods BitSeq [13], CEM [46], Cufflinks, eXpress [47], IsoEM [48], MMSeq [49], RSEM, rSeq [50], Sailfish, Scripture [51] and TIGAR2 [52] on a simulated and an experimental data set. They simulated data sets with different sizes, to test the performance of the different tools.

The observation was, that all tools except for Sailfish have increasing run times with increasing data sets. With IsoEM being the fastest and RSEM the slowest, when allowing multi-threading, which TIGAR2 lacks. Half of the tools, namely CEM, eXpress, MMSEQ, Sailfish, Scripture and TIGAR2, had a seemingly independent memory usage in the case of the sample size. The other half, which included BitSeq, Cufflinks, IsoEM, RSEM and rSeq were dependent on the sample size and increased with increasing sample size.

Apart from the runtime and memory analysis, their outcomes in terms of accuracy on a gene expression level suggest that a compiled estimated transcript isoform abundance based approach is more accurate than a count based approach. It should be noted, that the analysis of the count based methods was done on an "union exon" and "transcript" count based gene expression estimate, which were implemented by the group themselves. Furthermore, out of their tested eleven quantification programs, nine of those methods had a really good Spearman correlation coefficient with values above 0.90. The two methods that had a slightly poorer performance, Cufflinks and Scripturer, had a Spearman correlation coefficient of around 0.75. They also positively answered the question if higher expression levels and a larger library size will lead to a more accurate estimation. Apart from some other investigations, for example into GC content and polyadenylation, they also investigated the impact of the aligner on the estimation accuracy with the conclusion that the alignment tool has neither a high influence on transcript estimation nor on gene estimation.

Following this, some of the analysis done in this thesis will try to confirm or refute these statements.

Chapter 8

Results

8.1 Overview

All assessments were made with respect to gene counts. This enabled the comparison between quantifier operating solely on a gene level and quantifier producing transcript estimates. Concluding from this, all transcript IDs of the SEQC [33] and the SEQS data sets were first converted into gene IDs. These were then merged with equal gene IDs, to obtain a proper comparison. This resulted in 64253 different gene IDs for the statistical comparison. The gene counts were then normalized over all samples with the DESeq [17] R tool to obtain an average total count of 1×10^6. The TaqMan [33] and Bio-Rad [33] data sets were also scaled with respect to the average over all samples and methods of their matching genes. This resulted in 1046 different genes for the TaqMan comparison and 18325 different genes for the Bio-Rad comparison.

The data was then assessed in view of gene counts, mean (M), fold change (FC), standard deviation (SD), coefficient of variation (CV), Pearson's product-moment correlation coefficient (P CC), Spearman's rank correlation coefficient (S CC) and absolute logarithmic ratio of the standard deviations (R SD). Only a few plots are shown in this chapter, to retain readability. All other plots are inside the appendix section, denoted with "A:".

8.2 Precision

8.2.1 Assessment Techniques

The following comparisons were made to access the reproducibility of individual methods among different samples. The comparison of total or mean counts for individual samples are provided as boxplots to show the general distribution of counts among samples. Boxplots were also used to show the effects of normalization on the data. Further comparisons were made on the mean, the standard deviation and the coefficient of variation over all samples. The mentioned evaluations were also done considering fold changes, because usually one is interested in the detection of differential expressed genes when comparing RNA-Seq data from different sample sets.

8.2.2 Raw Counts

The summary boxplots in Figure 8.1 and A:1 are showing the raw total gene counts of all SEQC samples. One can easily see that the FluxCapacitor [12] produces about three times more samples than the other tools. The results from the SEQS simulations in Figure 8.2 and A:2 show similar results.

Here, the FluxCapacitor counts vary between 2.5×10^7 and 3.5×10^7. All other samples, with some exceptions were around the 1×10^7 count marker. It is interesting to see that the TopHat2 [4] alignment tool, which is the recommended alignment tool for Cufflinks [20] produces rather inconsistent estimations among samples. The boxplot of the samples from this combination is rather widespread in comparison to the boxplots from the other alignment and quantification combinations. This can be the reason for the rather bad performance assessment described from Kanitz el al. [3], because of them using the TopHat2 alignment tool for their comparison of Cufflinks. Another quantifier for which the alignment tool had an impact, was the featureCounts [10] quantifier. Here the combination with STAR [6] provides an average amount of counts compared the other methods. When combined with HISAT [5] and TopHat2, featureCounts produces only half of the counts.

Apart from this, the alignment tools had also a somewhat nonexistent effect on the outcome of the other quantifications. This is consistent with previously reported results [3]. The impact of different tool settings from the HTSeq quantifier on the total counts was also relatively low. Compared to this, the different settings from the EQP-QM tool had a little more impact on the total counts. This reports are almost consistent with the raw counts from the simulated data (Figure 8.2 and A:2). The only difference is the variation of the Cufflinks Samples when combined with TopHat2. But a variation between samples was seemingly nonexistent for all methods when applied to the simulated data.

Noticeable are also the total counts from the Flux Simulator profile file, which are the real counts for the simulated SEQS data sets. The counts are at 2×10^7, which is doubled comparing to most of the other counts. The reason for this is presumedly, Flux Simulator [16] counting each of the paired-end reads in then construction of the profile file. Nevertheless setting the `READ_NUMBER` to 2×10^7 produces the desired average 1×10^7 total count (for most of the methods).

8.2.3 Normalized Counts

The normalization with DESeq, shows a difference in total expression between A (Figure A:3) and B (Figure A:4) samples. This is true for all except BitSeq and the combination of Cufflinks and TopHat2. All BitSeq samples are in close proximity to the 1×10^6 marker, while the Cufflinks-TopHat2 combination is more widespread with one sample hovering around the 3×10^6 marker. All other methods had its A counts around 0.85×10^6 and the B counts around 1.15×10^6 supposing an overall increased expression of the HBR data sets. The observed differential expression was also visible in the simulated data, shown in Figure A:5 and A:6. The Cufflinks-TopHat2 combination is also showing a similar overall expression. Also the BitSeq [13] counts were again in the proximity of 1×10^6. Corresponding boxplots for the simulated data can be found at Figure and in the "Additional

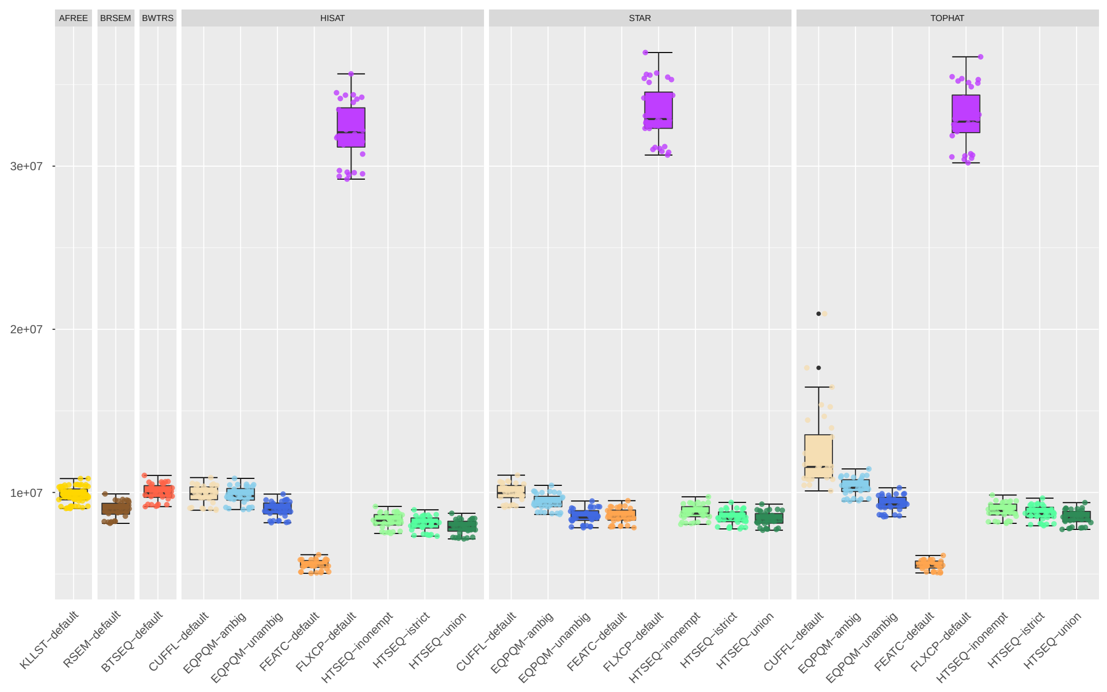

FIGURE 8.1: SEQC-A raw counts boxplot. This boxplots are showing the distribution of total raw counts of the SEQC-A data set between all 32 different samples. The colors provide a guideline for different quantifiers and settings of quantifiers. A description of tool abbreviations can be found at 6.2.

Material" section.

Table A:12 is showing a summary over all samples and genes for each method. All comparisons, except the "SD FC" column, are describing the applied function over all samples for each gene. The table entries then show the mean of this value over all genes. The "SD FC" column, shows the standard deviation over all genes of their respecting mean fold change over all samples. All fold changes are compared with respect to their absolute value after applying the binary logarithm.

The values are mostly consistent except for one outlier. Although the FluxCapacitor [12] had a rather high distribution of sample counts, its mean and standard deviation fold change values are average.

The previously mentioned bad combination of Cufflinks and TopHat2 is still visible. The standard deviation over all counts in the "SD A" and "SD B" columns show a high deviation compared to the other methods. Another outlier in a different direction is also shown by BitSeq. It has an average standard deviation for the counts, but low low values when observing the mean and the standard deviation

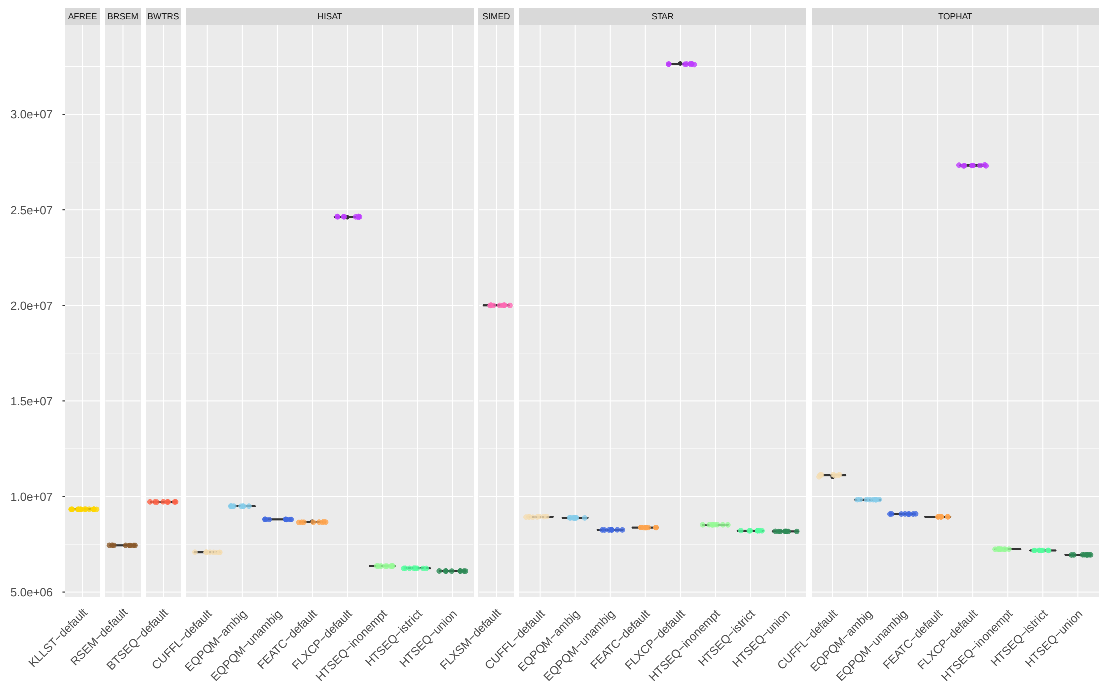

FIGURE 8.2: SEQS-A raw counts boxplot. This boxplots are showing the distribution of total raw counts of the SEQS-A simulated data set between all eight different samples. The colors provide a guideline for different quantifiers and settings of quantifiers. The "SIMED" column with "FLXSM-default" is showing the raw total count of the simulations from the Flux Simulator profile file. A description of tool abbreviations can be found at 6.2.

of fold changes. A similar discovery was also made in the boxplot summary.

The results from the simulated SEQS data in Table A:13 shows a similar distribution, with a suggesting slightly greater overall differential gene expression. This can be observed when comparing the mean counts of A and B samples. The BitSeq values are also similar to the one seen in the SEQS table. They have again relatively low fold change values.

The consistency among samples can be further investigated when comparing the standard deviation over all samples for each gene to the mean over all samples for each gene. The resulting measurement is the coefficient of variation when dividing the standard deviation through the mean count. A mean over all genes is shown in Table A:12 and corresponding for SEQS at Table A:13. The mean of the measurement is a bit vacuous as seen in the mentioned Tables. A better way to investigate this measurement is to plot the standard deviation versus the mean.

All settings and alignment combinations of the methods RSEM [14], EQP-QM

[8], featureCounts, HTSeq [9] had a similar plot like the one shown from the featureCounts and HISAT combination in Figure 8.3. Kallisto had a small variation among samples (Figure A:7). The FluxCapacitor, Cufflinks and BitSeq counts were as anticipated. However Cufflinks combined with STAR and HISAT, and all FluxCapacitor plots still showed a relatively good consistency with only some outliers (Figure A:8 and A:9).

The plots for Cufflinks-TopHat2 combination and BitSeq can be seen in Figure A:11 and Figure A:10. It can be easily seen that both tools produce some outliers which are correlating with the mean. Cufflinks has a conspicuous line of outlying genes with an increasing standard deviation, whereas BitSeq's outliers are in the lower section. The results from the SEQS data set were similar.

8.3 Accuracy

8.3.1 Assessment Techniques

This section assesses then results with respect to their mutual similarity and their correlation with reference data sets. This are TaqMan and Bio-Rad for SEQC and the Flux Simulator profile files for SEQS. All comparison are done on the mean over all samples for each gene. The data is quantified via comparing counts and absolute values (ABS) of log_2 fold changes for each gene. This is done with the Pearson correlation coefficient, Spearman correlation coefficient and absolute values of log_2 ratio (R) between standard deviations over all genes.

8.3.2 Method Comparison

Assessing counts alone is usually not sufficient for an accurate description of the performance of a tool. The reason for this is, that the usual goal of RNA-Seq experiments is the discovery of differentially expressed genes between two sample sets. This is also the main cause of using two different sample sets. Differential

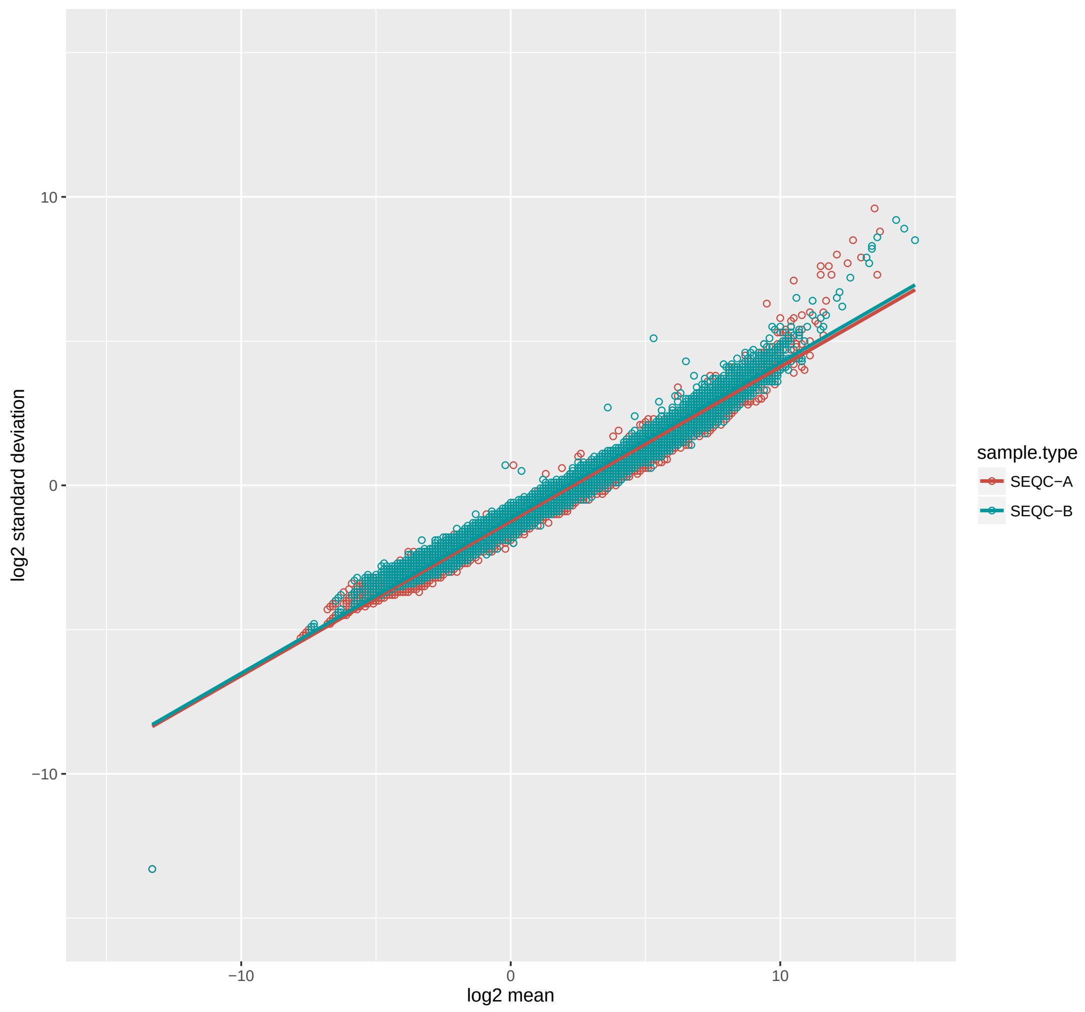

FIGURE 8.3: SEQC SD-mean scatterplot for featureCounts-HISAT. This scatter plot shows the logarithmic correlation of the standard deviation with the mean for each gene count over all samples of the SEQC data set. The plot shows the SEQC-A samples as red dots and the SEQC-B samples as cyan dots. The overall shape of the plot is representative for the results obtained from RSEM, EQP-QM, featureCounts and HTSeq.

expressed genes can be discovered when comparing the log_2 fold change values for each gene in two data sets. A comparison of the absolute log_2 fold changes for the SEQC data is shown in Figure 8.4.

Most of the tools had a similar distribution with an upper quantile around 2.5 and a upper whisker around 6. An outlier is again the BitSeq quantifier with its upper quantile around 1. Although the Cufflinks-Tophat2 combination and all FluxCapacitor fold changes had ordinary mean fold change values, their outliers are easy noticeable in the boxplot. The case for this could be high fold changes in both directions, which balance themselves out across genes. The SEQS fold change boxplots (Figure A:19) are showing a similar picture with an overall lower upper quantile but, therefore slightly higher outliers. The methods were further compared with Pearson, Spearman and their standard deviation.

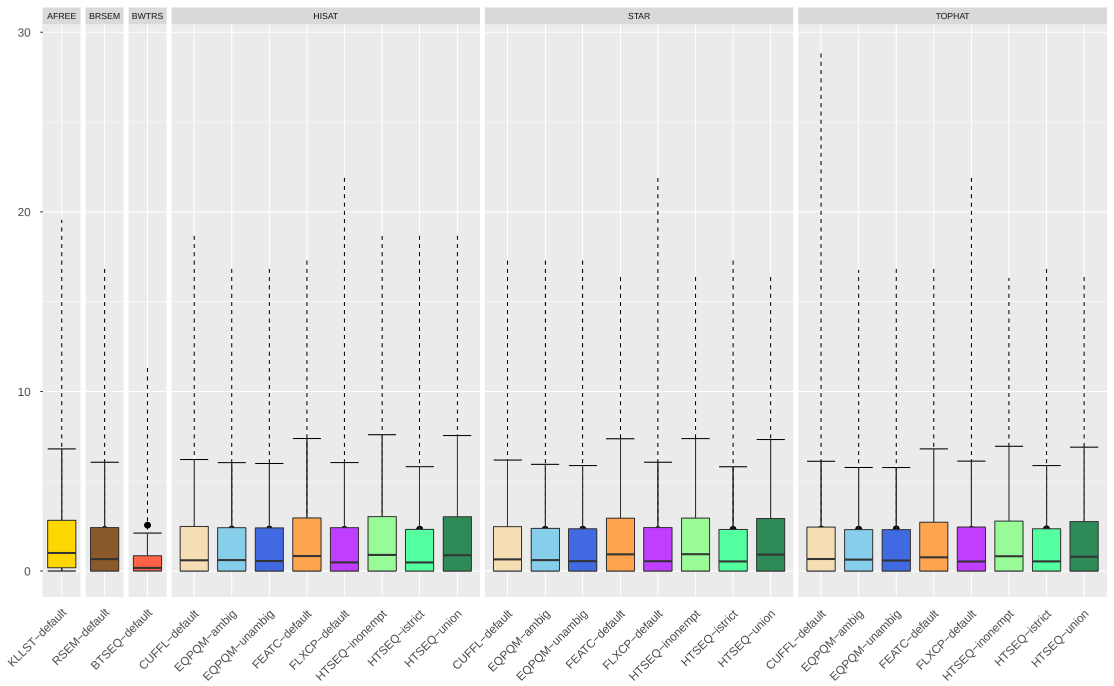

FIGURE 8.4: SEQC ABS FC boxplot. This boxplots are showing the distribution of absolute logarithmic fold change values of the SEQC data set between all 64253 different genes. The colors provide a guideline for different quantifiers and settings of quantifiers. A description of tool abbreviations can be found at 6.2.

The comparison of standard deviations is based on the suggestion from Norel et al. [36]. The values are computed with $|log_2\sigma_n - log_2\sigma_m|$ with σ_n and σ_m denoting the standard deviation for method n and m. This is 0 if the fold changes from both samples have the same distribution. The comparison of this for all method combinations is shown in Figure A:14 for SEQC and Figure A:15 for SEQS.

The BitSeq values have an average of 1.31 with this type of comparison. In contrast, the highest difference of standard deviations from other methods is achieved when comparing EQP-QM-TopHat2 with the ambiguous setting to featureCounts-HISAT. The value here is only 0.23, which concludes a still very similar deviation.

BitSeq is clearly standing out. Most of the other compared methods have values below 0.1. An exception is slightly visible when looking at featureCounts and HT-Seq with the union setting. The relating heatmap for the SEQS data set (Figure A:15) shows a similar overview with a slightly better overall correlation. BitSeq is again attracting attention with an average of 1.43.

To not solely rely on the findings of this measurement, the investigation was further enhanced with the inclusion of commonly used tools for comparison. Namely Pearson's product-moment correlation coefficient, which measures linear correlation of data sets. The respective plot can be found at Figure 8.5.

Pearson shows a similar picture to the one obtained from the ratio of standard deviations. The overall counts are a little more diverse and patterns are more visible. This includes the previously spotted pattern of featureCounts and HTSeq with the union setting. Similar almost constant low correlations can be observed with the HTSeq setting intersection-nonempty and kallisto. The correlation of these samples to other methods is around 0.76 on average. BitSeq has the lowest correlation of all methods, with average values of 0.56. This was also visible with the standard deviation ratio comparison. Most of the other methods have values above 0.8. The same assessment for the SEQS data set can be seen in figure A:17. The results have an overall better correlation and BitSeq is still standing out.

A further measurement of correlation is Spearman's rank correlation coefficient. The related plot can be found in Figure A:16. BitSeq's correlation with other methods is much better than with Pearson and the standard deviation ratio. It has an average correlation of almost 0.8. The previous observed patterns of low correlation are consistent with the Spearman correlations. The Spearman correlations for the SEQS data set (Figure A:18) are similar to the results from the Pearson correlation of the SEQS data set. This excludes BitSeq which has a similar improvement as BitSeq's SEQC Pearson and Spearman comparison.

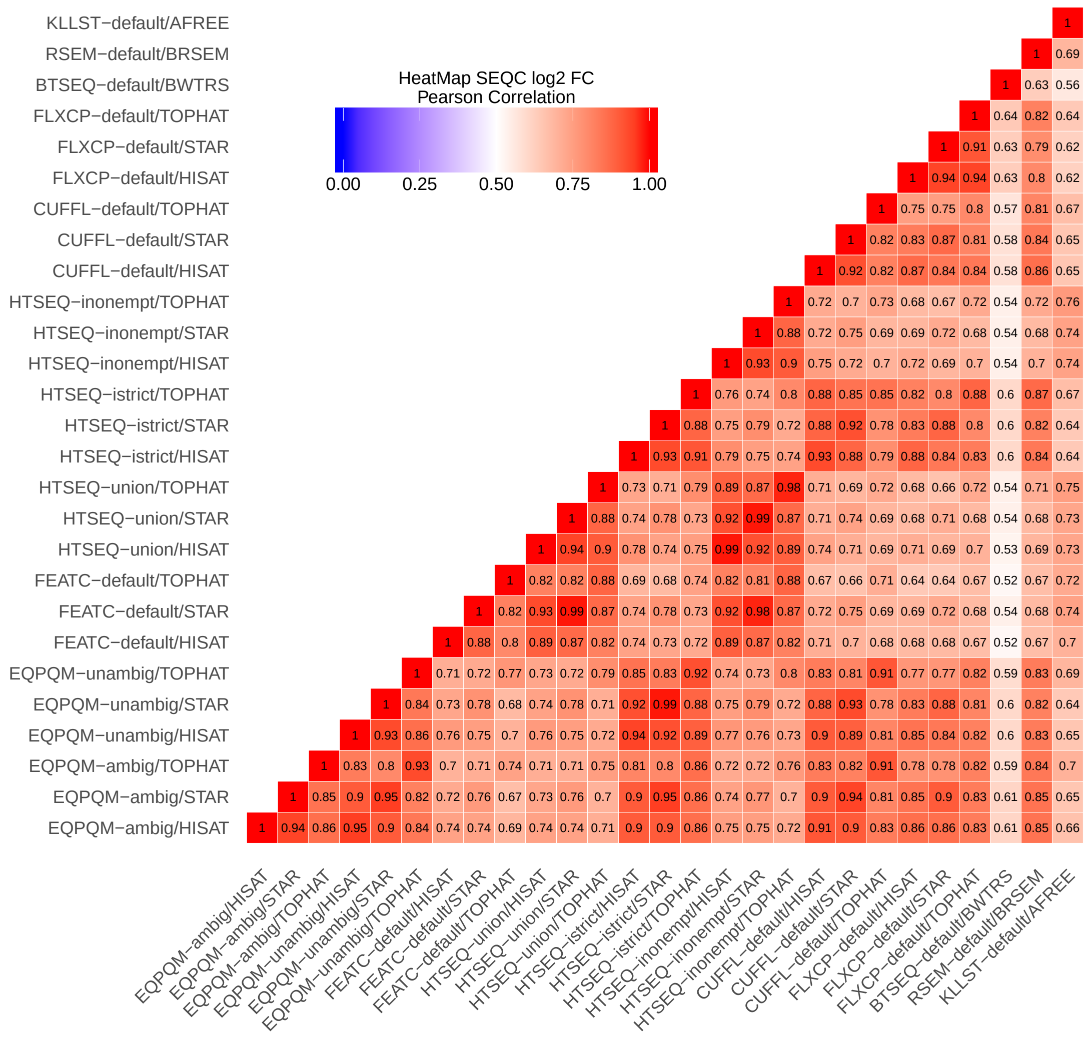

FIGURE 8.5: SEQC Pearson FC heatmap. This heatmap shows then logarithmic fold change Pearson correlations of all methods to one another with the SEQC data set. A darker red reveals a higher correlation. A description of tool abbreviations can be found at 6.2.

8.3.3 Method Accuracy

The previous comparisons are all based on the data itself without an underlying ground truth. They provide a rough overview of how similar different tools are, but not if they are correct. To address this, the following comparisons are based on SEQC against the qRT-PCR TaqMan and Bio-Rad data sets and SEQS against the Flux Simulator profile data, which is the underlying ground truth for the simulations.

To give an overview, the first assessment is again the comparison of different absolute log_2 fold change values. The boxplots in Figure 8.6 and A:20 are showing the TaqMan and Bio-Rad genes compared to corresponding SEQC genes. All Bio-Rad boxplots have a consistent upper quantile of around 2.5. The only exception is BitSeq, with a lower upper quantile. There is also a little trend between Bio-Rad an FluxCapacitor visible, their upper quantiles, whiskers and outliers are slightly better matching than the other methods compared to Bio-Rad.

The TaqMan boxplot is a little bit more widespread and almost all methods have a slightly higher differential expression than the TaqMan samples. BitSeq is again underperforming and FluxCapacitor has more heightened fold change compared to Bio-Rad and all other methods. The SEQS fold change boxplots (Figure A:19) were mostly described in Section 8.3.2. The overall expression was a little bit lower than underlying ground truth obtained from the FLux Simulator profile files.

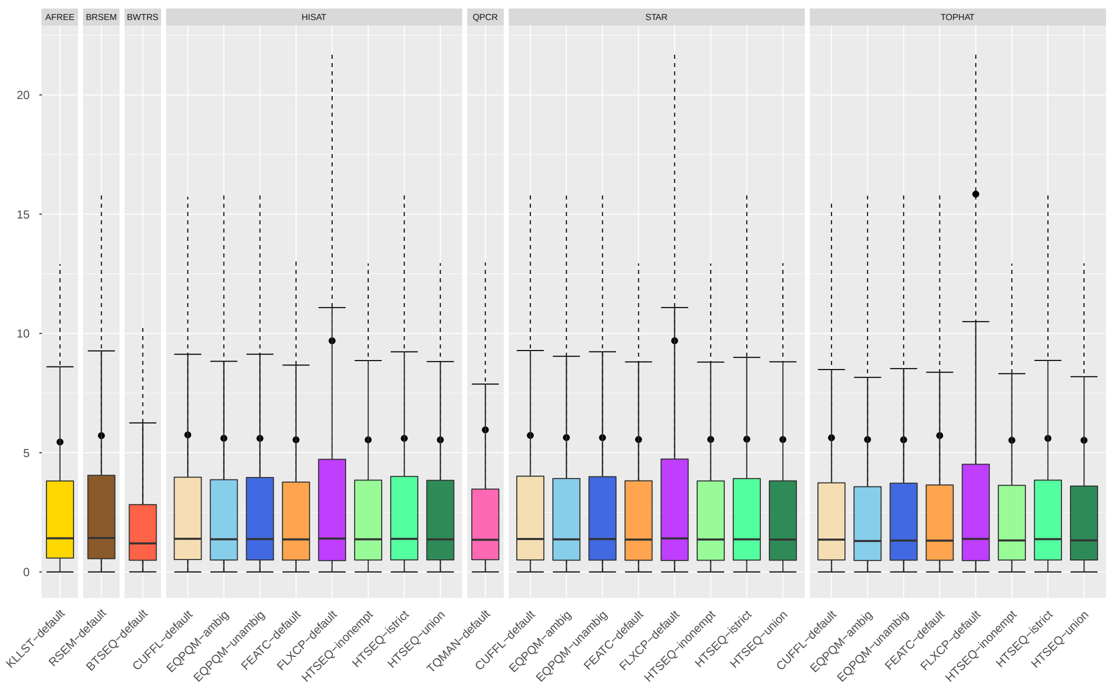

FIGURE 8.6: TaqMan ABS FC boxplot. This boxplots are showing the distribution of absolute logarithmic fold change values of the SEQC data set between all 1046 different genes represented in the TaqMan data set. The colors provide a guideline for different quantifiers and settings of quantifiers. The "QPCR" column with "TQMAN-default" is showing the fold changes of the TaqMan data. A description of tool abbreviations can be found at 6.2.

Almost all counts had similar correlations when compared to Bio-Rad, which is shown in Table A:21. The correlation with Pearson showed an overall low performance with the SEQC-A data set. The values spanned from 0.41 for fluxCapacitor-HISAT to 0.60 HTSeq-HISAT with the union setting. In contrast to this is the SEQC-B data set, with Pearson correlation values reaching from 0.69 for fluxCapacitor-STAR and fluxCapacitor-TopHat2 to 0.82 for Cufflinks-STAR.

Opposing to this are the values of the Spearman correlation, which ranged from 0.85 to 0.87 for SEQC-A and from 0.86 to 0.87 for SEQC-B. Applying the previously mentioned ratio of the standard deviation to the comparison of SEQC and Bio-Rad counts, shows an overall bad correlating deviation in counts for SEQC-A and SEQC-B sets. The best ratios are 1.33 in SEQC-A for BitSeq and 1.75 in SEQC-B for EQP-QM-STAR in the unambiguous setting and HTSeq-STAR in the intersection-strict setting, while the worst rations are 1.63 for Cufflinks-TopHat2 in SEQC-A and 2.01 for FluxCapacitor-HISAT in SEQC-B. These values assume a three- and fourfold difference in standard deviations.

The comparison of SEQC to TaqMan counts A:22 resulted in a slightly better performance for Pearson. The data sets achieved values from 0.57 to 0.62 for SEQC-A and values from 0.82 to 0.84 for SEQC-B. Making the comparisons with Spearman resulted in a range similar to the one obtained with Bio-Rad. The minimum and maximum values were 0.81 and 0.87 for SEQC-A and 0.82 and 0.87 for SEQC-B. Really different are the values from the standard deviation ratio. They had an overall vast improvement with values ranging from 0.60 for BitSeq to 0.99 for Cufflinks-TopHat2 in the SEQC-A data set and from 0.27 for EQP-QM-STAR with the unambiguous setting and HTSeq-STAR with the union to 0.66 for Cufflinks-TopHat2 in the SEQC-B data set.

Pearson and Spearman correlation values were mostly uninteresting for the simulated SEQS data sets. All sample types except Cufflinks-TopHat2 and BitSeq had values reaching from 0.93 to almost 1.00. BitSeq had with Spearman values of 0.87 on both sample sets a lower correlation than with the Pearson values of 0.99 for SEQS-A and almost 1.00 for SEQS-B. Cufflinks-TopHat2 was definitely underperforming with a Pearson correlation of 0.43 for SEC-A and 0.78 for SEQS-B, despite the fact, that the expression values for the simulations were based on the Cufflinks counts. The ratio of different standard deviations had overall good values. Most values were in the range of close to 0.00 and 0.11. An exception were the SEQS-B results from FluxCapacitor, BitSeq and Cufflinks-Tophat2 with values from 0.20 to 0.28. SEQS-A had a little heightened value of 0.20 with BitSeq and an absolute outlier of 1.12 with Cufflinks-TopHat2.

Assessing only counts is not sufficient when looking at RNA-Seq data. The differential gene expression presented through fold changes is a more important measurement for RNA-Seq data. The previous quantifications were therefore also applied to the logarithmic fold change values of the results (Table A:24). The fold changes compared to Bio-Rad had an overall acceptable performance for Pearson and Spearman. Pearson achieved values between 0.75 for FluxCapacitor and 0.79 for most of the other samples. The Spearman correlation was a little bit better with overall values from 0.83 to 0.85. Almost all standard deviation ratios had values below 0.26, with BitSeq making an exception of 0.76.

A further investigation was done for the fold change values to access the validity of the differential gene expression. The "TPR" and "FPR" columns in Table A:24 show the true positive rate or specificity and the false positive rate for the obtained fold change values for each method compared to Bio-Rad. The threshold for declaring a gene "differential expressed" was set to 2.0 which presumes a twofold differential expression. The overall specificity was between 0.72 and 0.77 with BitSeq making an exception with a true positive rate of only 0.54. This assumes that only half of the BitSeq samples were quantified right in terms of differential gene expression. Still the false positive rates are at an almost constant 0.23 with FluxCapacitor-TopHat2 and BitSeq being at 0.25.

The SEQC fold changes compared to TaqMan (Table A:25) had an overall better performance than when compared to Bio-Rad. The Pearson correlations were between 0.87 for FluxCapacitor and 0.94 for BitSeq. Even less allocated were the Spearman values with values between 0.90 and 0.92. These almost constant values are probably resulting from the low number of different genes in the TaqMan data set. Opposed to this are the ratios of the standard deviations which are spread between 0.08 for kallisto and 0.52 for FluxCapacitor-HISAT. The other FLuxCapacitor combinations had a similar bad ratio.

Also a little bit increased were the BitSeq values with 0.32, while all other tools being below 0.19. The small sample size of different genes in the TaqMan set was also beneficial for the specificity of BitSeq. The specificity with a value of 0.77 was still quite low compared to the other values being above 0.82 and up to 0.88, but an improvement when compared to the specificity of 0.54 obtained in the Bio-Rad comparison. The whole false positive rate was also a little bit higher than the rates obtained with Bio-Rad. They stretched from 0.28 for most of the methods to 0.32 for BitSeq.

The same measurements were also applied to the SEQS results (Table A:26). All values except the ones from BitSeq and FluxCapacitor were above 0.75 for the Pearson correlations. EQP-QM-TopHat2 with the unambiguous setting had the best Pearson correlation of 0.88.

The Spearman correlations were similar to the ones obtained with SEQC and Bio-Rad. BitSeq, FluxCapacitor and RSEM were in the lower section with values between 0.79 and 0.81 while the other tools achieved correlations from 0.83 with Cufflinks-HISAT and HTSeq-HISAT in the union and intersection-strict settings to 0.91 with EQP-QM-TopHat2 and the unambiguous setting.

All standard deviation ratios, except BitSeq were below 0.09. The high value of 1.46 from BitSeq is consistent with bad performance of BitSeq in the previous SEQC Bio-Rad and TaqMan comparisons. The accuracy of BitSeq is even worse when looking at the true positive rate. BitSeq achieves a rate of only 0.30 which is even worse than the rate of 0.54 obtained from the SEQC and Bio-Rad specificity calculation. Also the false positive rate of 0.12 is the highest observed in all results from the SEQS data set. Second worst in specificity was FluxCapacitor-HISAT with a value of 0.67. The best true positive rate of 0.87 was achieved by Cufflinks-TopHat2 which is presumably influenced due to the simulation expression values being based on the Cufflinks count data. The false positive rates, apart from the previous mentioned rate for BitSeq, were all below 0.11.

Chapter 9

Discussion

The quantification of RNA-Seq data is accompanied by many different obstacles. Although the constant development of new sequencing techniques and machines has made the RNA-Seq readings more and more easy, some problems still exist. These are mostly caused by the fact, that RNA-Seq analysis has to be fast, cheap and precise. One can not achieve more than two of these basic principles without compromising the other remaining one.

Because it is generally difficult to sequence long reads, it became common practice to analyse short 100 nt long paired-end reads. This is nothing compared to the on average 2.2 kb long [26] mammalian transcripts. Nevertheless this technique has been improved much and has become the method of choice when sequencing RNA reads. Machines based upon these techniques are producing a vast amount of unsorted data, to compensate for these relatively short reads. This data has to be assessed and the reads have to be mapped and counted, which creates new problems.

The reads have a high probability to be shared between multiple transcripts when mapped to a gene. Many authors have developed several different sophisticated methods to address this and other problems.

The two commonly used basic principles are the direct counting of gene and the estimation of transcript abundances. Both quantification methods are usually accompanied by a preceding alignment step in which the reads are aligned to the genome or transcriptome. Although many different alignment tools exist, the impact of using one specific aligner for a quantifier is usually negligible [2]. This was also the overall outcome of this thesis with one major exception, the combination of Cufflinks [11] and TopHat2 [4]. This tool combination was highly inconsistent across different technical replicas which resulted in a low overall precision. This stands in contrast of being the authors recommended combination [20]. However this did not affect the overall precision of Cufflinks.

All other tools had an even performance when used with different aligners. The choice of the right quantification tool is more important. This is also seen when assessing the two quantification tools BitSeq [13] and RSEM [14]. These tools require alignments to the transcriptome instead of the genome, which was performed using two different settings for Bowtie2. The reason for this was the strictness of RSEM which is described in Section 4.2.7. BitSeq in contrast has no special preferences for the alignments. Although one might argue that the restrictions could result in a lower performance than the aforementioned, the opposite happened. RSEM had an overall good performance when compared to other tools whereas BitSeq provided a poor performance. BitSeq achieved only an acceptable performance when compared to the TaqMan [33] data. But this was presumably only because of the low count of different genes in this set. The comparison of BitSeq with Bio-Rad [33] and with the simulated data had a specificity of only 0.54 and 0.30. However this work does not claim that this assessment is an absolute marking for BitSeq and a trained user of BitSeq could achieve better results when adjusting the parameters, but an untrained user might do well to be wary when using BitSeq.

All other tools had a relatively even true positive and false positive rate. The comparison of fold changes with Bio-Rad resulted in an overall specificity of 0.23 (Table A:24) and the comparison with TaqMan (Table A:25) resulted in around 0.29, while the simulated data had a false positive rate around 0.10. This indicates, that simulated sets cannot resemble the diversity of real data and should not be solely used to assess the performance of a tool, when one does not have the real underlying expression values. But the simulation of RNA-Seq data would be obsolete if this ground truth of real data were be available. Yet another option would be to find a good estimation of expression values obtained from the results of multiple tools.

The assessment with Spearman and Pearson proved to be more comparable for simulated data with a similar result as the Bio-Rad Pearson and Spearman correlations. As well as it is not recommended to quantify a tool only on simulations, it is also not recommended to only rely on a small sample set like TaqMan because one might neglect variations in the data which is not covered by a small sample set. Almost all methods had an even correlation of 0.90 for Pearson and Spearman. This might be true for the set, but not for a larger set as shown with Bio-Rad where the correlations had more variation.

The suggestion from Norel et al. [36] to also assess the standard deviation proved to be a good addition to the comparison of samples. A combination of all three methods might be the most reliable choice of quantifying a tool. On both, when observing counts of samples and fold changes. The standard deviation can show aspects of the data which are not visible with Pearson or Spearman. An example is the high correlation of the Spearman SEQC-B [33] counts when compared to Bio-Rad. Whereas the standard deviations of the results from the various methods compared to the standard deviation of Bio-Rad is rather high.

Also the overall impact of different tool settings was rather low. Still there is a pattern of HTSeq [9] with the intersection-strict setting visible when looking at all different comparisons. The setting had a slightly improved performance in contrast to the other settings. Also the novel tools kallisto [15] and EQP-QM [8] showed similar results, compared to other well performing methods like RSEM, featureCounts [10] and HTSeq. EQP-QM could perform better when used with its underlying alignment tool EQP, but the focus of this thesis was on the performance of quantification tools on an equivalent basis.

Chapter 10

Conclusion

The quantification of RNA-Seq data is a difficult task with many variables to consider. Because of a missing ground truth, this is also true when assessing the methods which try to quantify this data. Although some of the presented tools, like EQP-QM [8], kallisto [15], RSEM [14] and HTSeq [9] with the intersection-strict setting, performed relatively well, there is still room for improvement. This is visible when looking at the average specificity of only 0.78 and the average Spearman correlation of 0.85 for this tools in the Bio-Rad [33] and SEQS comparisons. It is therefore not wise to select and advice the usage of a specific tool set and this thesis is designed to provide an overview of possible aspects to consider when trying to choose the most appropriate tool for a data set. Still the presented assessments can also be considered as a guideline for this choice. This work shows possible common methods to evaluate the consistency of the produced data. One could assess the RNA-Seq reads with different tools and then compare the results with some of the presented comparison methods.

Appendix

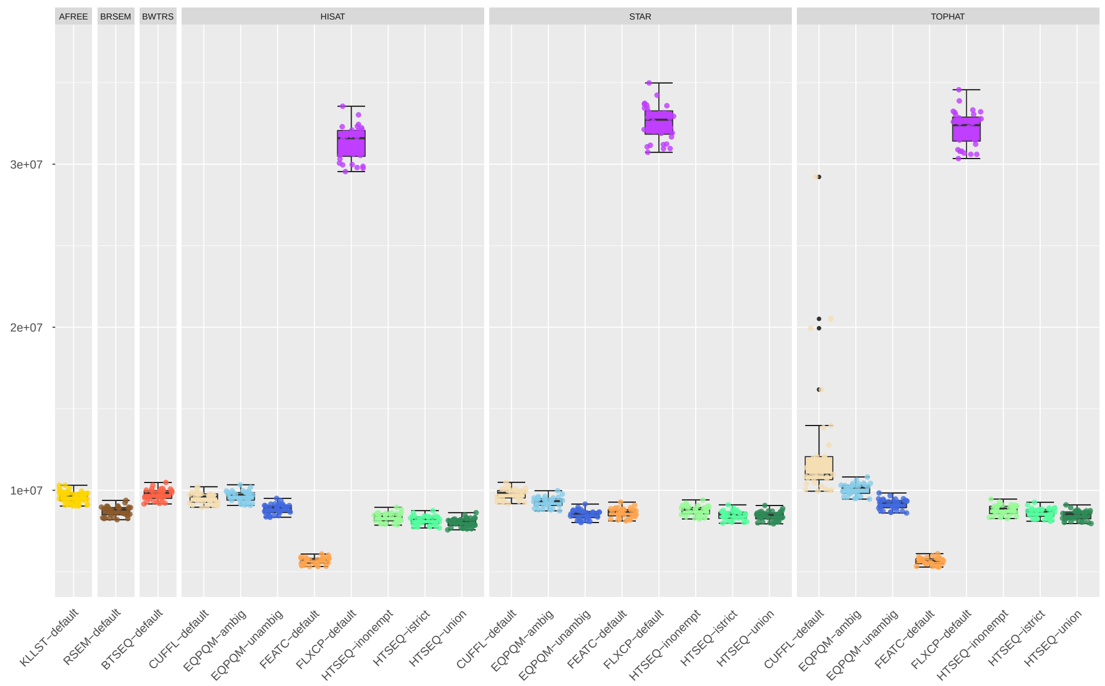

FIGURE A:1: SEQC-B raw counts boxplot. These boxplots are showing the distribution of total raw counts of the SEQC-B data set between all 32 different samples. The colors provide a guideline for different quantifiers and settings of quantifiers. A description of tool abbreviations can be found at 6.2. The plot is referred to in Section 8.2.2.

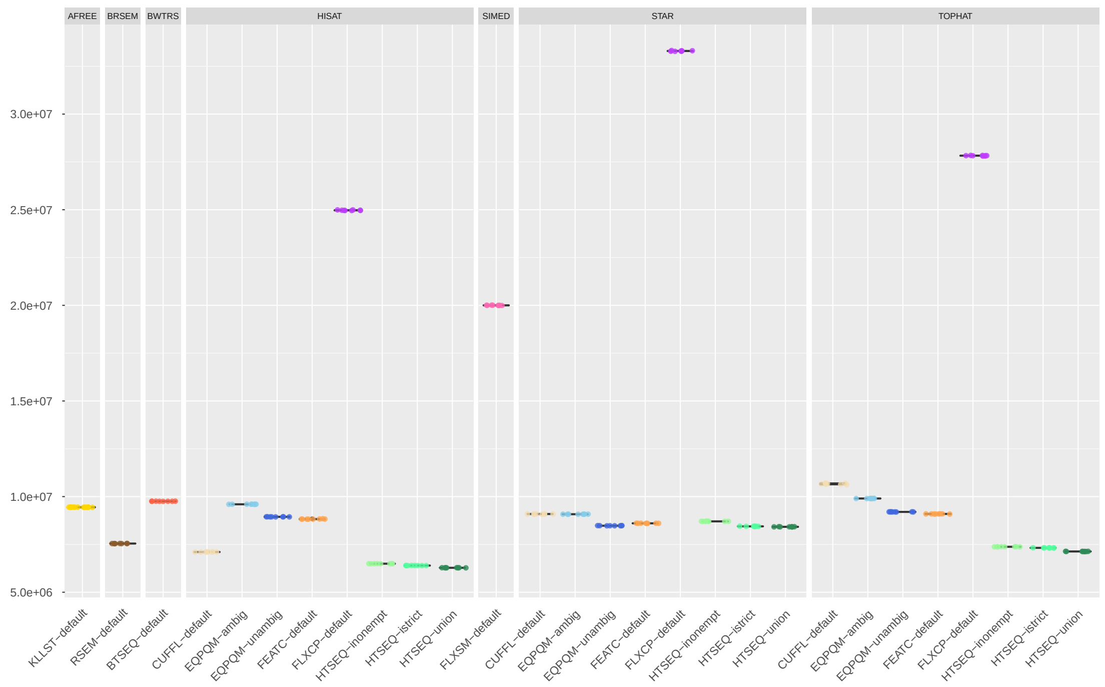

Figure A:2: SEQS-B raw counts boxplot. These boxplots are showing the distribution of total raw counts of the SEQS-B simulated data set between all eight different samples. The colors provide a guideline for different quantifiers and settings of quantifiers. The "SIMED" column with "FLXSM-default" is showing the normalized total count of the simulations from the Flux Simulator profile file. A description of tool abbreviations can be found at 6.2. The plot is referred to in Section 8.2.2.

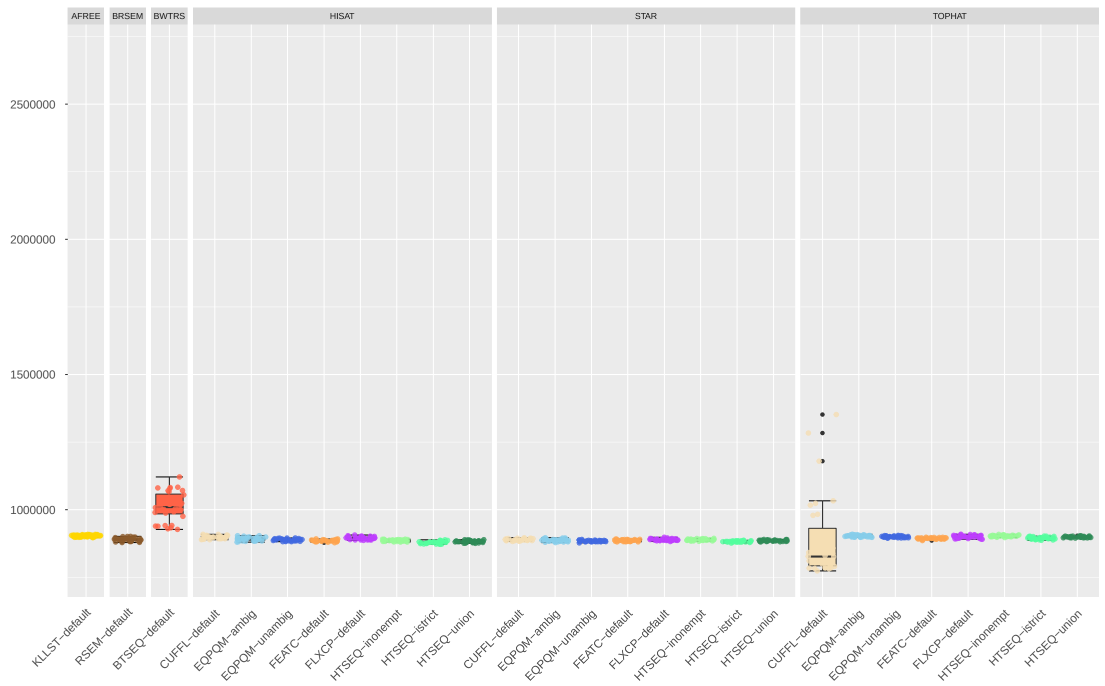

Figure A:3: SEQC-A normalized counts boxplot. These boxplots are showing the distribution of total normalized counts of the SEQC-A data set between all 32 different samples. The colors provide a guideline for different quantifiers and settings of quantifiers. A description of tool abbreviations can be found at 6.2. The plot is referred to in Section 8.2.3.

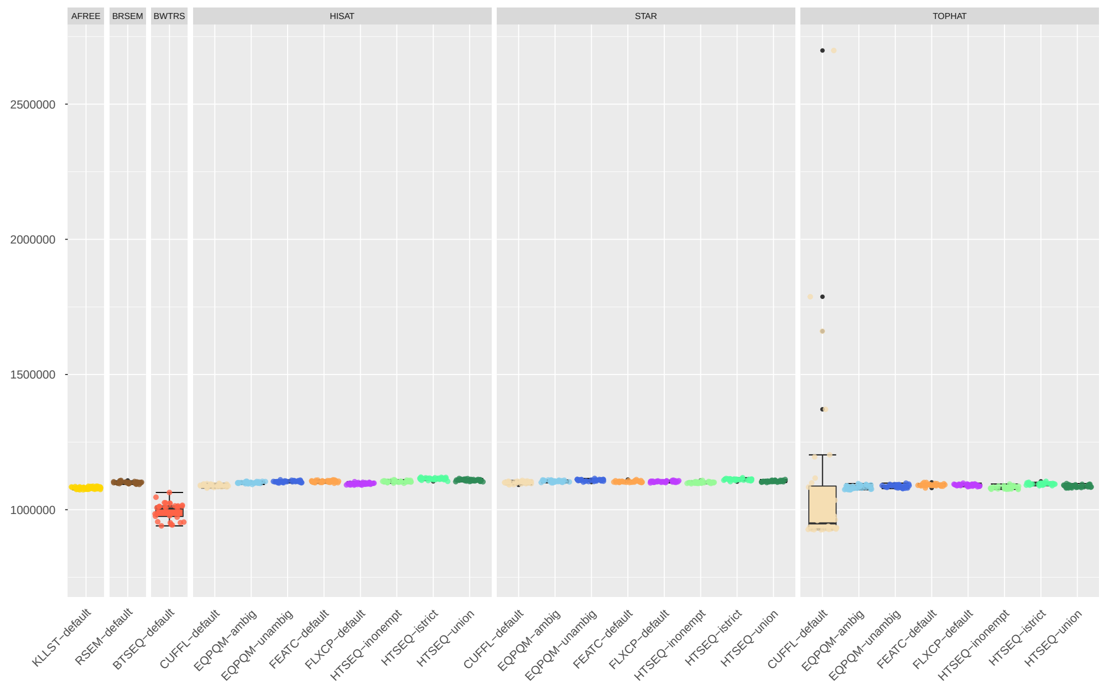

FIGURE A:4: SEQC-B normalized counts boxplot. These boxplots are showing the distribution of total normalized counts of the SEQC-B data set between all 32 different samples. The colors provide a guideline for different quantifiers and settings of quantifiers. A description of tool abbreviations can be found at 6.2. The plot is referred to in Section 8.2.3.

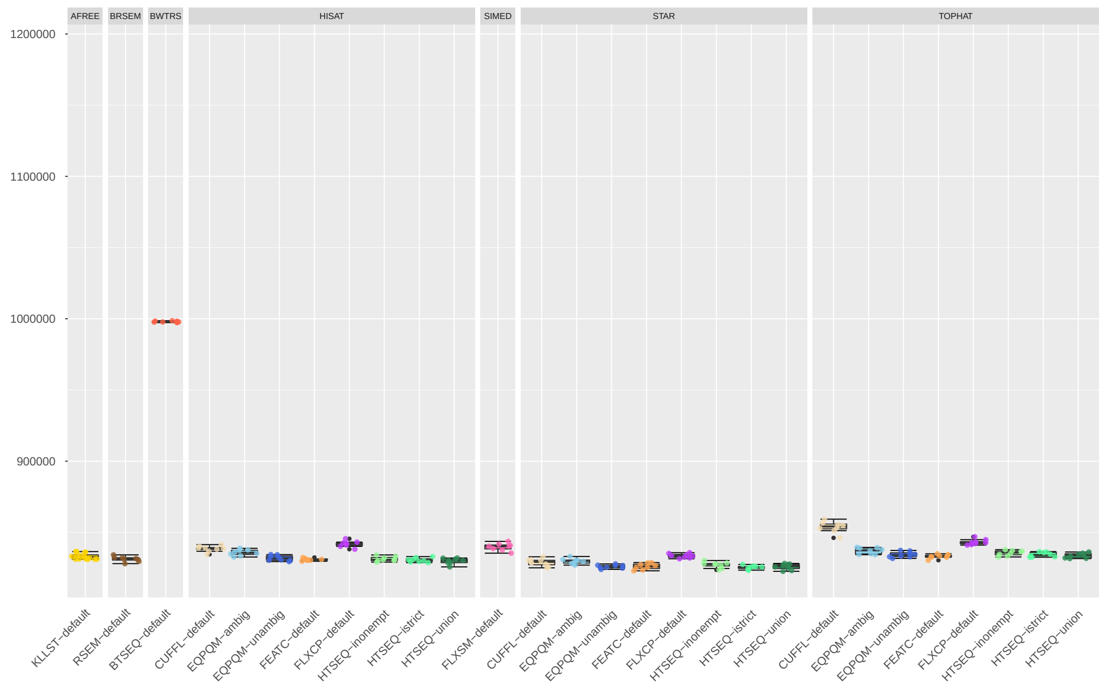

FIGURE A:5: SEQS-A normalized counts boxplot. These boxplots are showing the distribution of total normalized counts of the SEQS-A simulated data set between all eight different samples. The colors provide a guideline for different quantifiers and settings of quantifiers. The "SIMED" column with "FLXSM-default" is showing the normalized total count of the simulations from the Flux Simulator profile file. A description of tool abbreviations can be found at 6.2. The plot is referred to in Section 8.2.3.

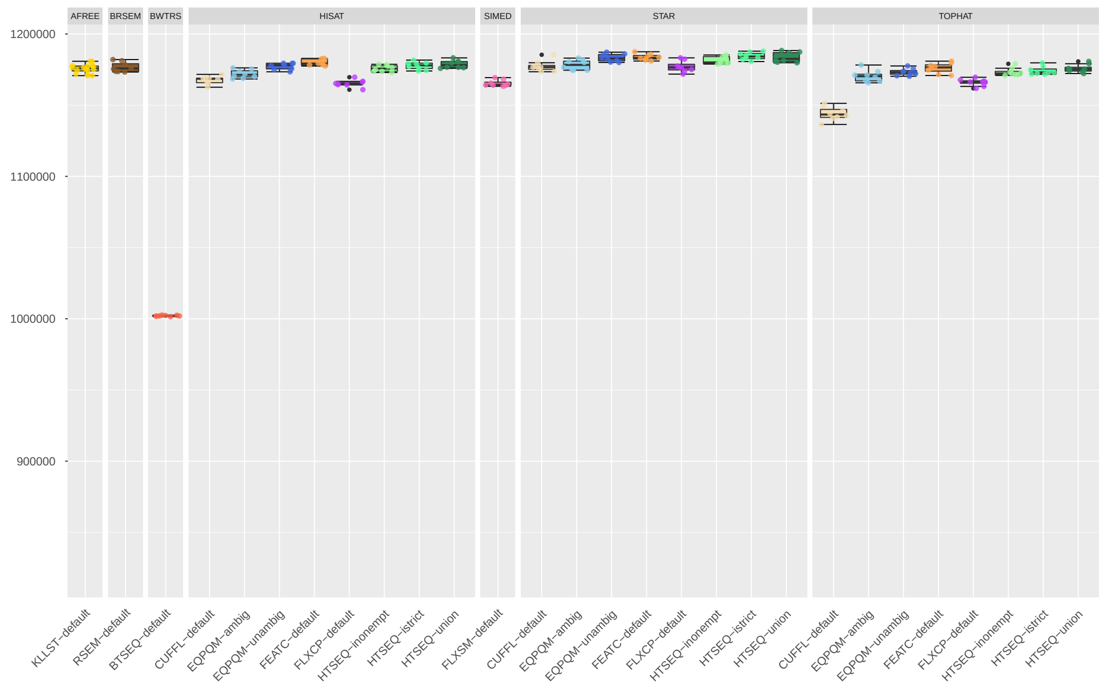

Figure A:6: SEQS-B normalized counts boxplot. These boxplots are showing the distribution of total normalized counts of the SEQS-B simulated data set between all eight different samples. The colors provide a guideline for different quantifiers and settings of quantifiers. The "SIMED" column with "FLXSM-default" is showing the normalized total count of the simulations from the Flux Simulator profile file. A description of tool abbreviations can be found at 6.2. The plot is referred to in Section 8.2.3.

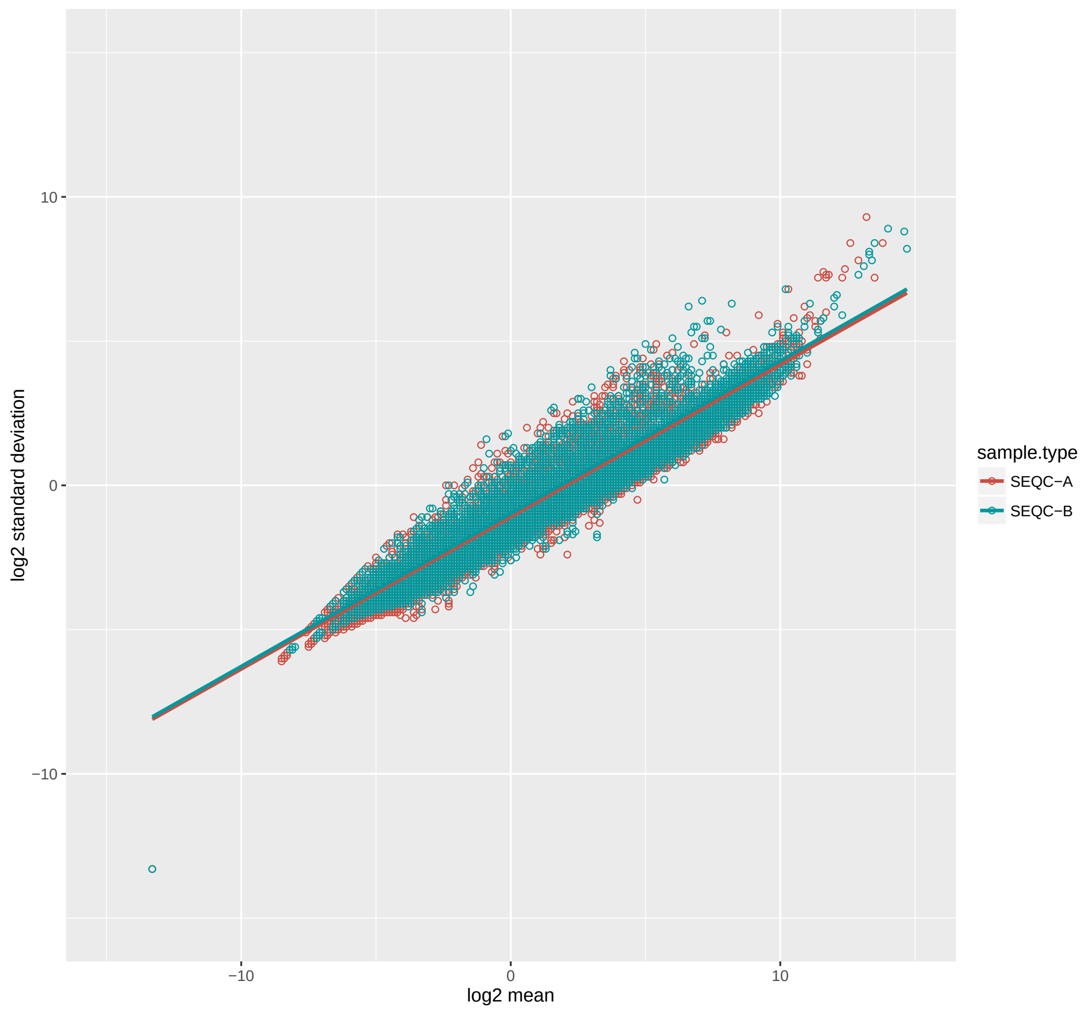

Figure A:7: SEQC SD-mean scatterplot for kallisto. This scatter plot shows the logarithmic correlation of the standard deviation with the mean for each gene count over all samples of the SEQC data set. The plot shows the SEQC-A samples as red dots and the SEQC-B samples as cyan dots. The plot is referred to in Section 8.2.3.

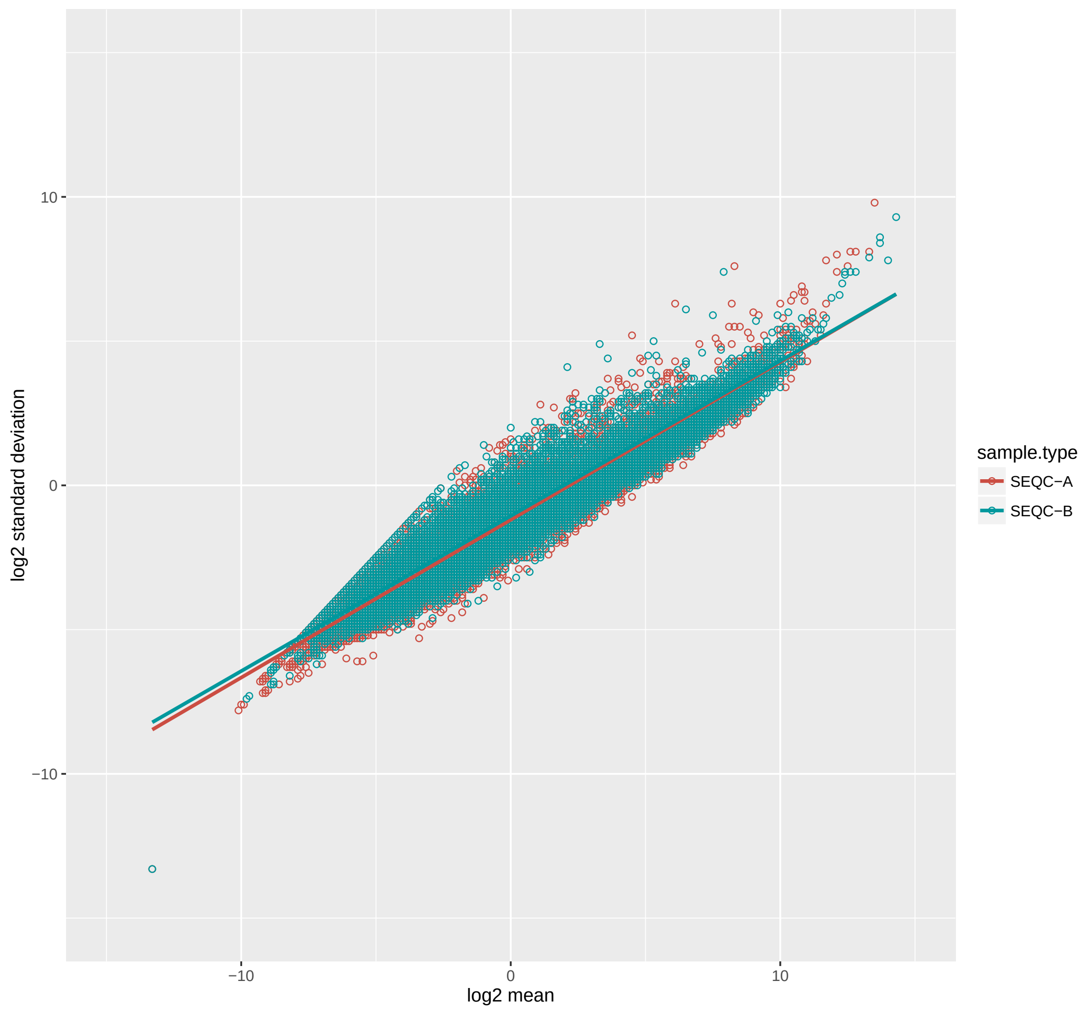

FIGURE A:8: SEQC SD-mean scatterplot for FluxCapacitor-TopHat2. This scatter plot shows the logarithmic correlation of the standard deviation with the mean for each gene count over all samples of the SEQC data set. The plot shows the SEQC-A samples as red dots and the SEQC-B samples as cyan dots. The overall shape of the plot is representative for the results obtained from all FluxCapacitor aligner combinations. The plot is referred to in Section 8.2.3.

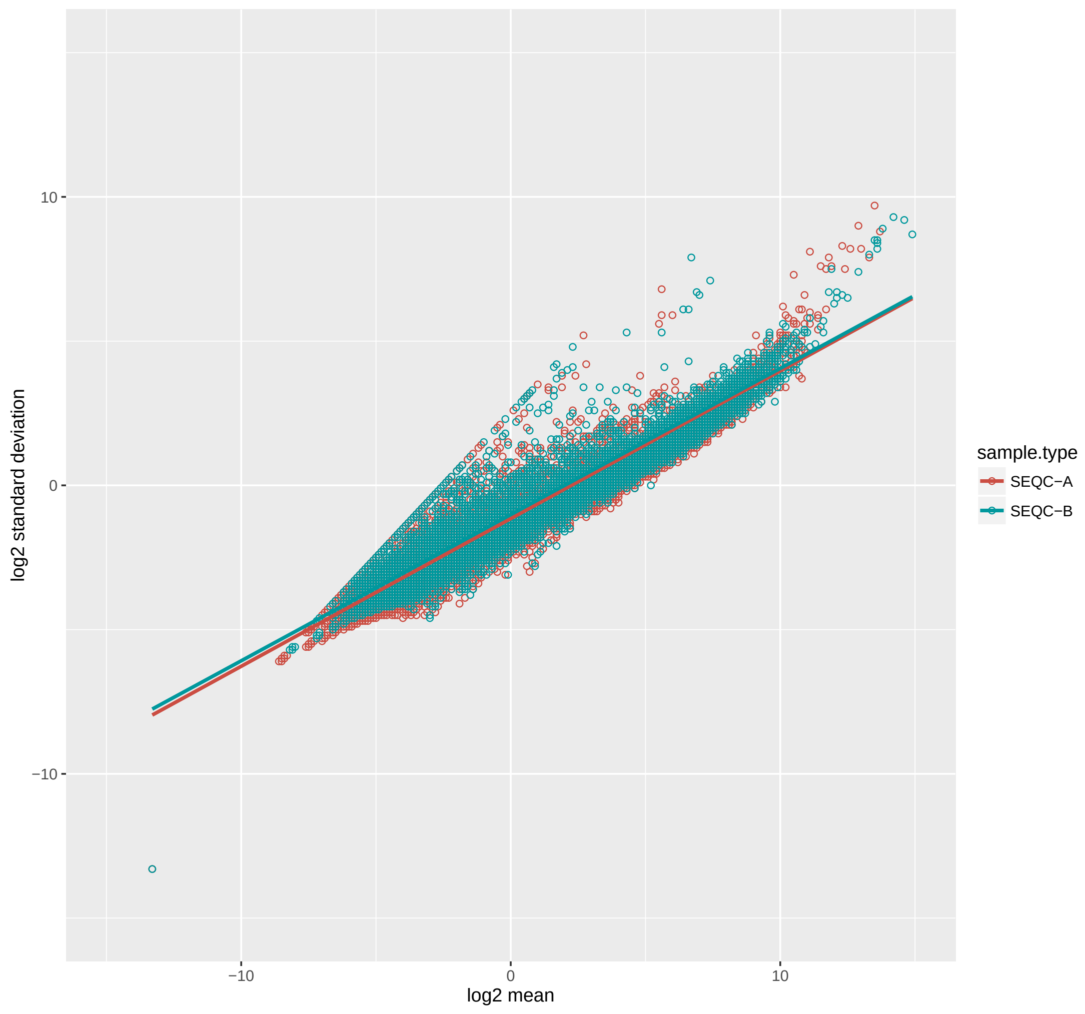

Figure A:9: SEQC SD-mean scatterplot for Cufflinls-STAR. This scatter plot shows the logarithmic correlation of the standard deviation with the mean for each gene count over all samples of the SEQC data set. The plot shows the SEQC-A samples as red dots and the SEQC-B samples as cyan dots. The overall shape of the plot is representative for the results obtained from Cufflinks-STAR and Cufflinks-HISAT. The plot is referred to in Section 8.2.3.

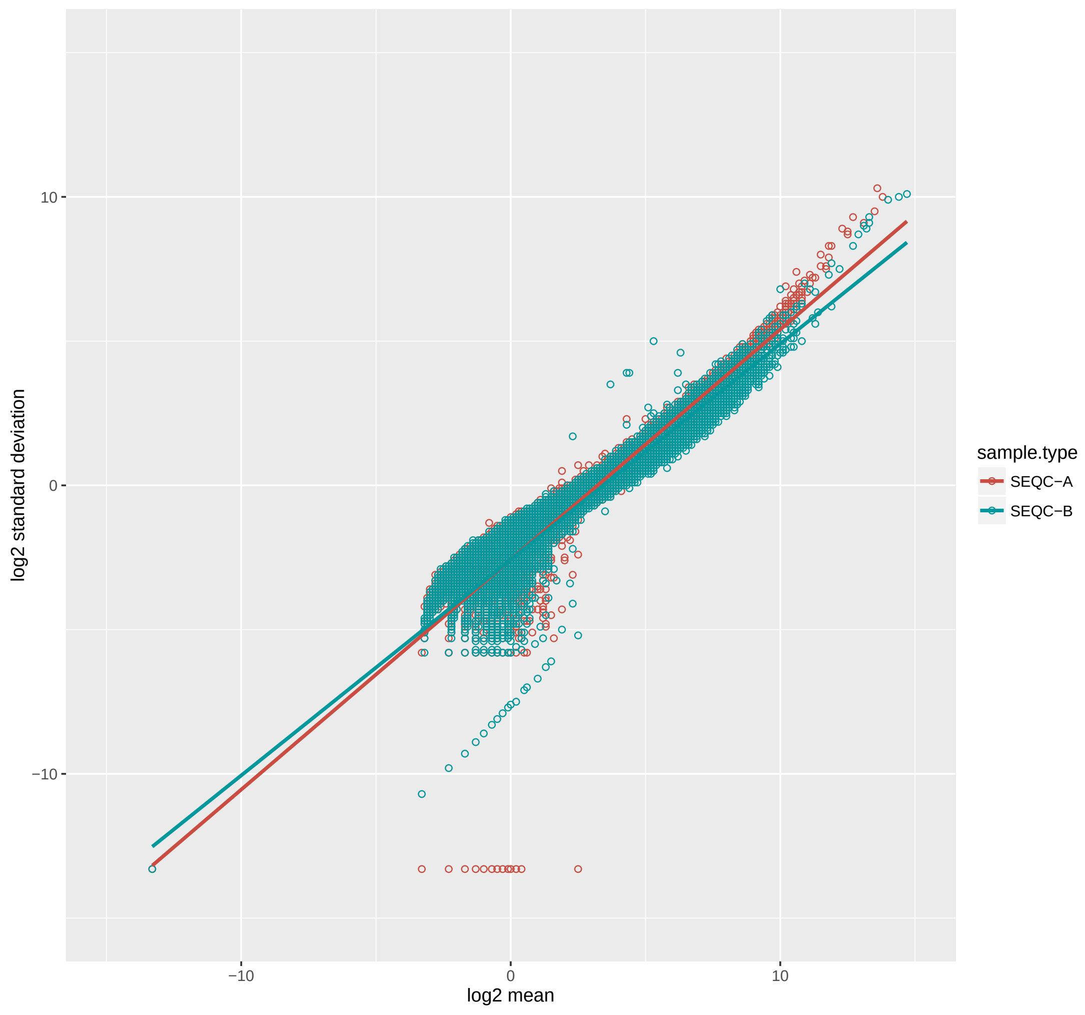

FIGURE A:10: SEQC SD-mean scatterplot for BitSeq. This scatter plot shows the logarithmic correlation of the standard deviation with the mean for each gene count over all samples of the SEQC data set. The plot shows the SEQC-A samples as red dots and the SEQC-B samples as cyan dots. The plot is referred to in Section 8.2.3.

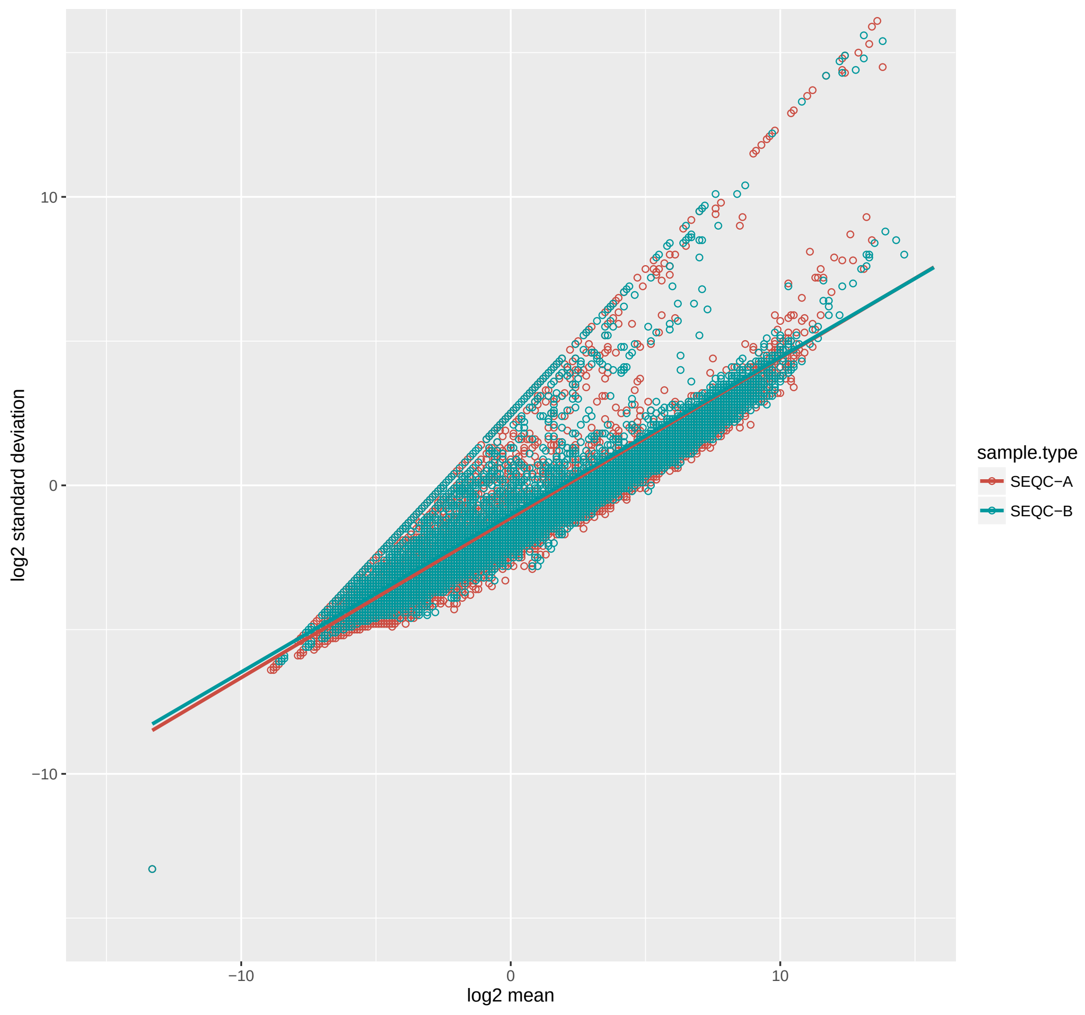

Figure A:11: SEQC SD-mean scatterplot for Cufflinks-TopHat2. This scatter plot shows the logarithmic correlation of the standard deviation with the mean for each gene count over all samples of the SEQC data set. The plot shows the SEQC-A samples as red dots and the SEQC-B samples as cyan dots. The plot is referred to in Section 8.2.3.

	M A	M B	M FC	SD FC	SD A	SD B	CV A	CV B
STAR								
EQPQM A	13.80	17.20	1.75	2.50	0.63	0.73	1.12	1.13
EQPQM U	13.76	17.26	1.73	2.52	0.63	0.74	1.17	1.17
FEATC	13.79	17.18	2.03	2.59	0.65	0.75	1.27	1.27
HTSEQ N	13.78	17.20	2.03	2.60	0.65	0.75	1.27	1.27
HTSEQ S	13.73	17.30	1.72	2.51	0.63	0.74	1.16	1.17
HTSEQ E	13.83	17.13	2.03	2.59	0.64	0.74	1.26	1.27
CUFFL	13.83	17.13	1.84	2.64	0.66	0.77	1.18	1.18
FLXCP	13.84	17.18	1.79	2.69	0.68	0.79	1.17	1.17
HISAT								
EQPQM A	13.88	17.12	1.75	2.48	0.61	0.72	1.14	1.14
EQPQM U	13.83	17.18	1.74	2.51	0.60	0.72	1.19	1.18
FEATC	13.78	17.19	2.08	2.75	0.73	0.87	1.34	1.34
HTSEQ N	13.73	17.26	2.05	2.66	0.64	0.77	1.30	1.28
HTSEQ S	13.68	17.34	1.72	2.55	0.62	0.76	1.17	1.17
HTSEQ E	13.79	17.18	2.06	2.64	0.63	0.76	1.29	1.28
CUFFL	13.98	16.96	1.84	2.66	0.65	0.77	1.19	1.19
FLXCP	13.93	17.06	1.78	2.72	0.68	0.81	1.18	1.17
TOPHAT								
EQPQM A	14.05	16.86	1.70	2.39	0.61	0.72	1.12	1.12
EQPQM U	14.00	16.91	1.70	2.42	0.61	0.72	1.18	1.17
FEATC	13.91	16.99	1.98	2.68	0.75	0.88	1.33	1.31
HTSEQ N	13.99	16.91	1.93	2.55	0.63	0.75	1.26	1.25
HTSEQ S	13.93	17.05	1.72	2.50	0.62	0.74	1.18	1.17
HTSEQ E	14.05	16.83	1.94	2.54	0.63	0.74	1.25	1.24
CUFFL	13.84	17.07	1.82	2.62	8.60	14.18	1.18	1.17
FLXCP	14.01	16.98	1.79	2.68	0.69	0.80	1.19	1.17
BWTRS								
BTSEQ	15.74	15.48	0.66	1.09	1.12	0.83	0.17	0.17
BRSEM								
RSEM	13.85	17.12	1.78	2.49	0.63	0.75	1.15	1.15
AFREE								
KLLST	14.06	16.81	2.00	2.45	0.66	0.79	1.31	1.31

TABLE A:12: SEQC count precision. ”A” and ”B” are describing sample sets. ”M” describes the mean. ”SD” describes the standard deviation. ”CV” describes the coefficient of variation. A description of tool abbreviations can be found at 6.2 and 6.1. The plot is referred to in Section 8.2.3.

	M A	M B	M FC	SD FC	SD A	SD B	CV A	CV B
STAR								
EQPQM A	12.91	18.34	1.55	2.73	0.44	0.58	0.84	0.85
EQPQM U	12.85	18.42	1.54	2.76	0.43	0.58	0.87	0.88
FEATC	12.86	18.42	1.58	2.80	0.43	0.57	0.87	0.88
HTSEQ U	12.85	18.42	1.57	2.80	0.43	0.58	0.87	0.88
HTSEQ S	12.85	18.43	1.53	2.75	0.43	0.58	0.87	0.87
HTSEQ E	12.87	18.39	1.60	2.80	0.43	0.57	0.87	0.88
CUFFL	12.90	18.32	1.64	2.91	0.50	0.65	0.87	0.88
FLXCP	12.97	18.32	1.57	2.93	0.46	0.61	0.87	0.88
HISAT								
EQPQM A	13.01	18.24	1.60	2.75	0.43	0.56	0.84	0.85
EQPQM U	12.95	18.32	1.60	2.79	0.42	0.56	0.88	0.88
FEATC	12.93	18.37	1.58	2.83	0.44	0.58	0.89	0.89
HTSEQ N	12.92	18.34	1.52	2.84	0.50	0.65	0.89	0.89
HTSEQ S	12.93	18.33	1.52	2.83	0.49	0.65	0.89	0.89
HTSEQ E	12.94	18.30	1.57	2.87	0.49	0.65	0.89	0.89
CUFFL	13.05	18.17	1.64	2.97	0.54	0.70	0.89	0.90
FLXCP	13.10	18.14	1.55	3.01	0.54	0.70	0.89	0.89
TOPHAT								
EQPQM A	13.03	18.22	1.64	2.78	0.42	0.56	0.84	0.85
EQPQM U	12.99	18.26	1.62	2.81	0.42	0.55	0.88	0.88
FEATC	12.97	18.30	1.57	2.82	0.48	0.63	0.89	0.89
HTSEQ N	12.98	18.30	1.52	2.81	0.47	0.61	0.88	0.89
HTSEQ S	12.99	18.27	1.54	2.81	0.46	0.61	0.88	0.89
HTSEQ E	13.00	18.26	1.57	2.83	0.46	0.61	0.88	0.89
CUFFL	13.29	17.81	1.73	2.98	0.56	0.71	0.87	0.87
FLXCP	13.12	18.15	1.57	2.99	0.51	0.66	0.88	0.89
BWTRS								
BTSEQ	15.53	15.60	0.59	1.06	0.5	0.47	0.12	0.11
BRSEM								
RSEM	12.94	18.31	1.61	2.82	0.47	0.63	0.86	0.87
AFREE								
KLLST	12.97	18.30	1.67	2.84	0.44	0.59	0.88	0.89

Table A:13: SEQS count precision. ”A” and ”B” are describing sample sets. ”M” describes the mean. ”SD” describes the standard deviation. ”CV” describes the coefficient of variation. A description of tool abbreviations can be found at 6.2 and 6.1. The plot is referred to in Section 8.2.3.

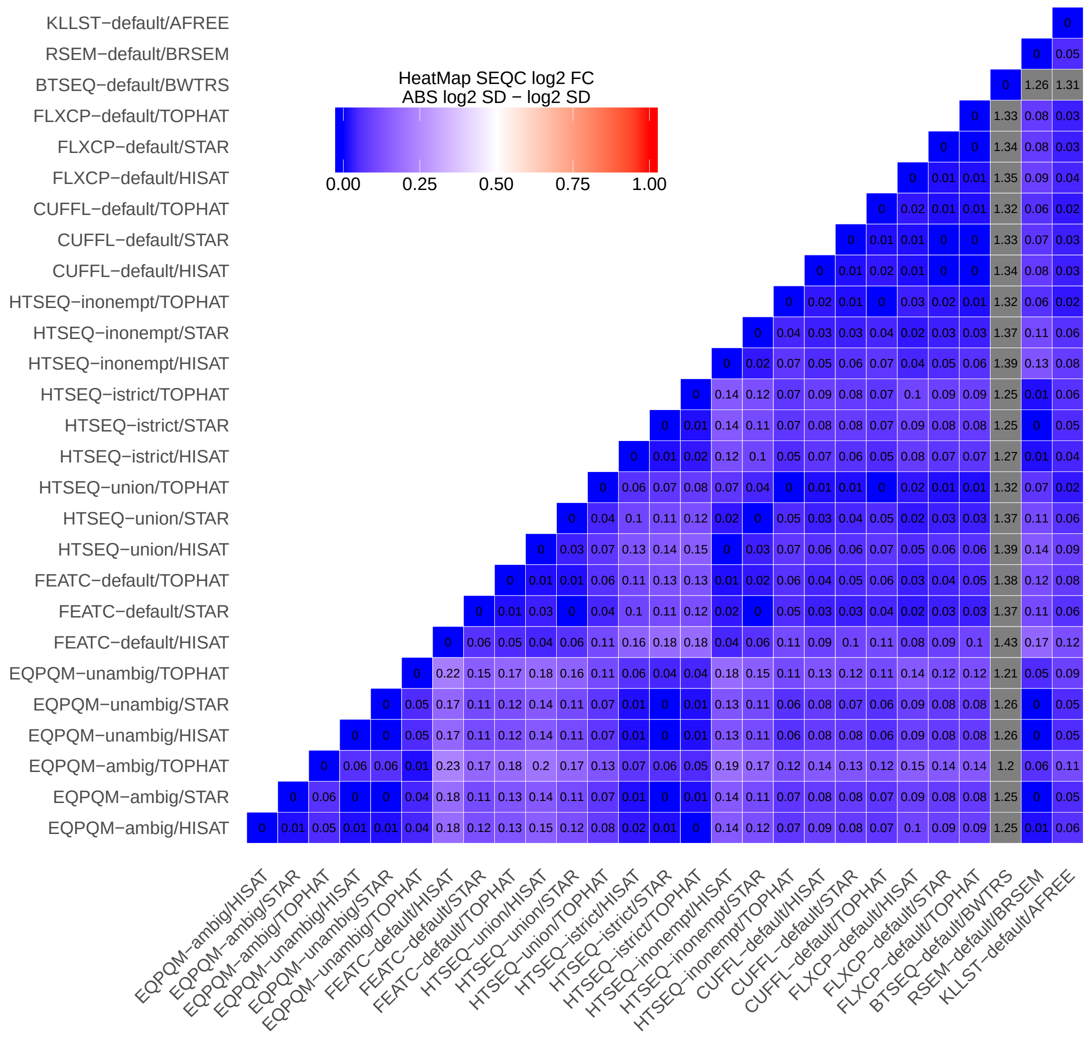

Figure A:14: SEQC SD FC ratio heatmap. This heatmap shows then logarithmic ratio between the standard deviation over fold change values of all methods to one another with the SEQC data set. A darker blue reveals a better similarity of the standard deviations. A description of tool abbreviations can be found at 6.2. The plot is referred to in Section 8.3.2.

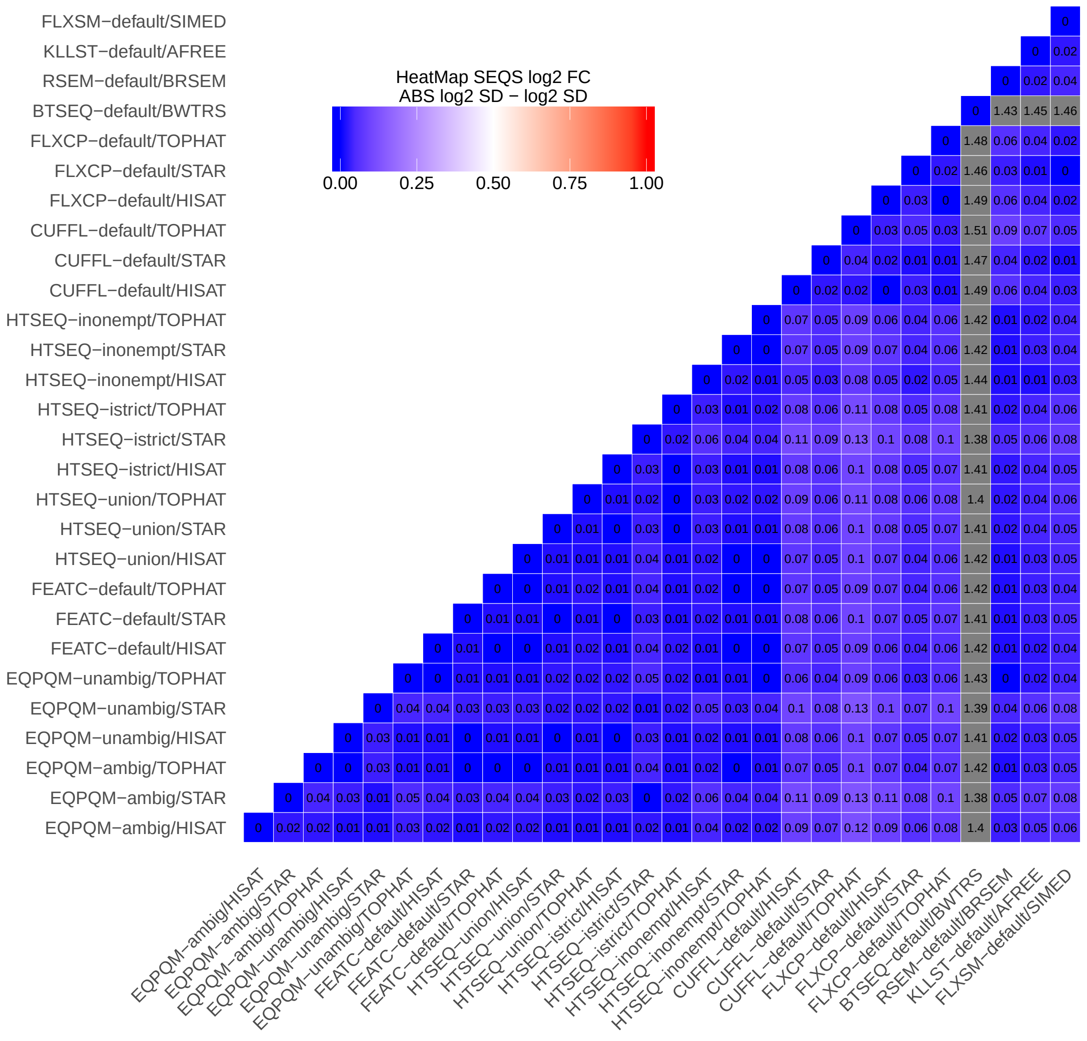

Figure A:15: SEQS SD FC ratio heatmap. This heatmap shows then logarithmic ratio between the standard deviation over fold change values of all methods to one another with the SEQS data set. A darker blue reveals a better similarity of the standard deviations. A description of tool abbreviations can be found at 6.2. The plot is referred to in Section 8.3.2.

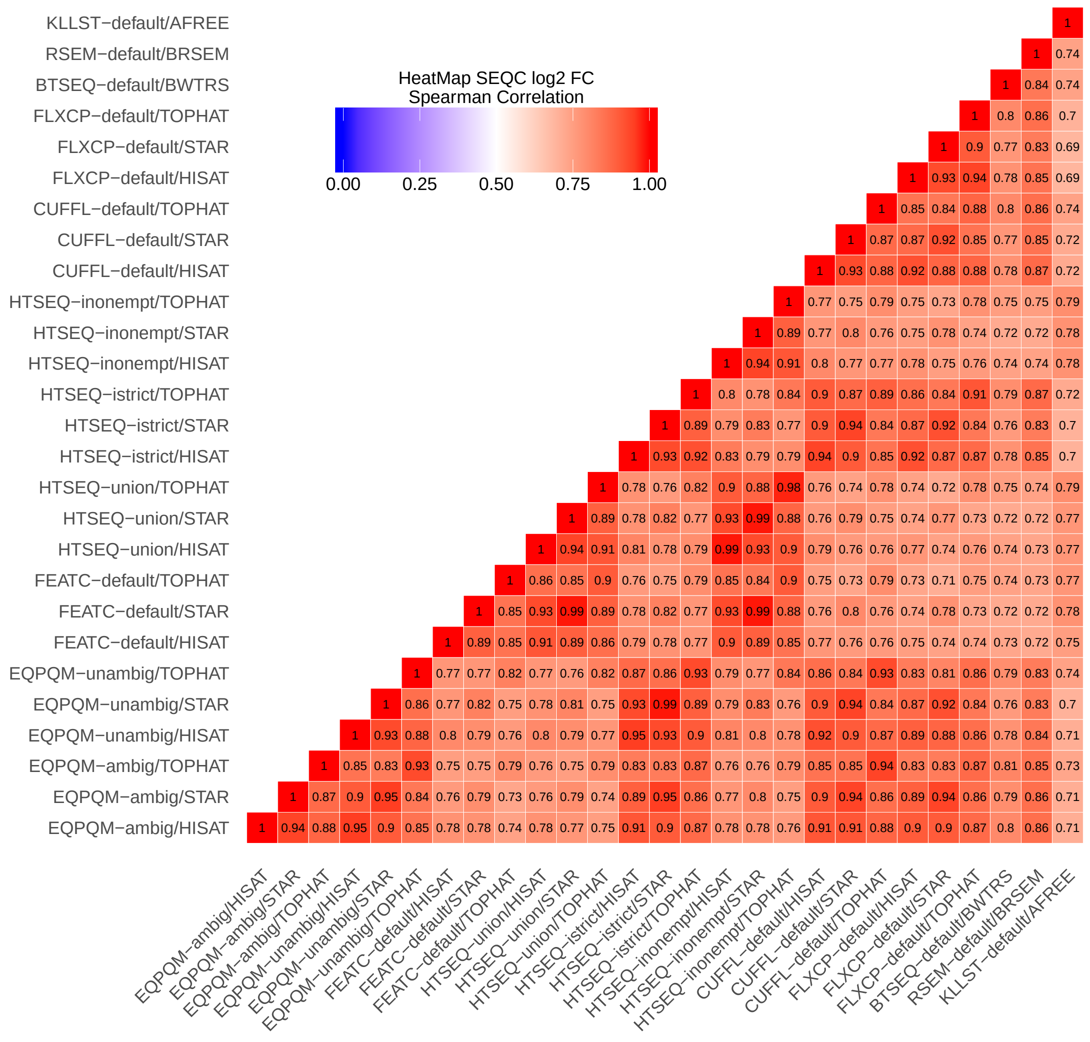

Figure A:16: SEQC Spearman FC heatmap. This heatmap shows then logarithmic fold change Spearman correlations of all methods to one another with the SEQC data set. A darker red reveals a higher correlation. A description of tool abbreviations can be found at 6.2. The plot is referred to in Section 8.3.2.

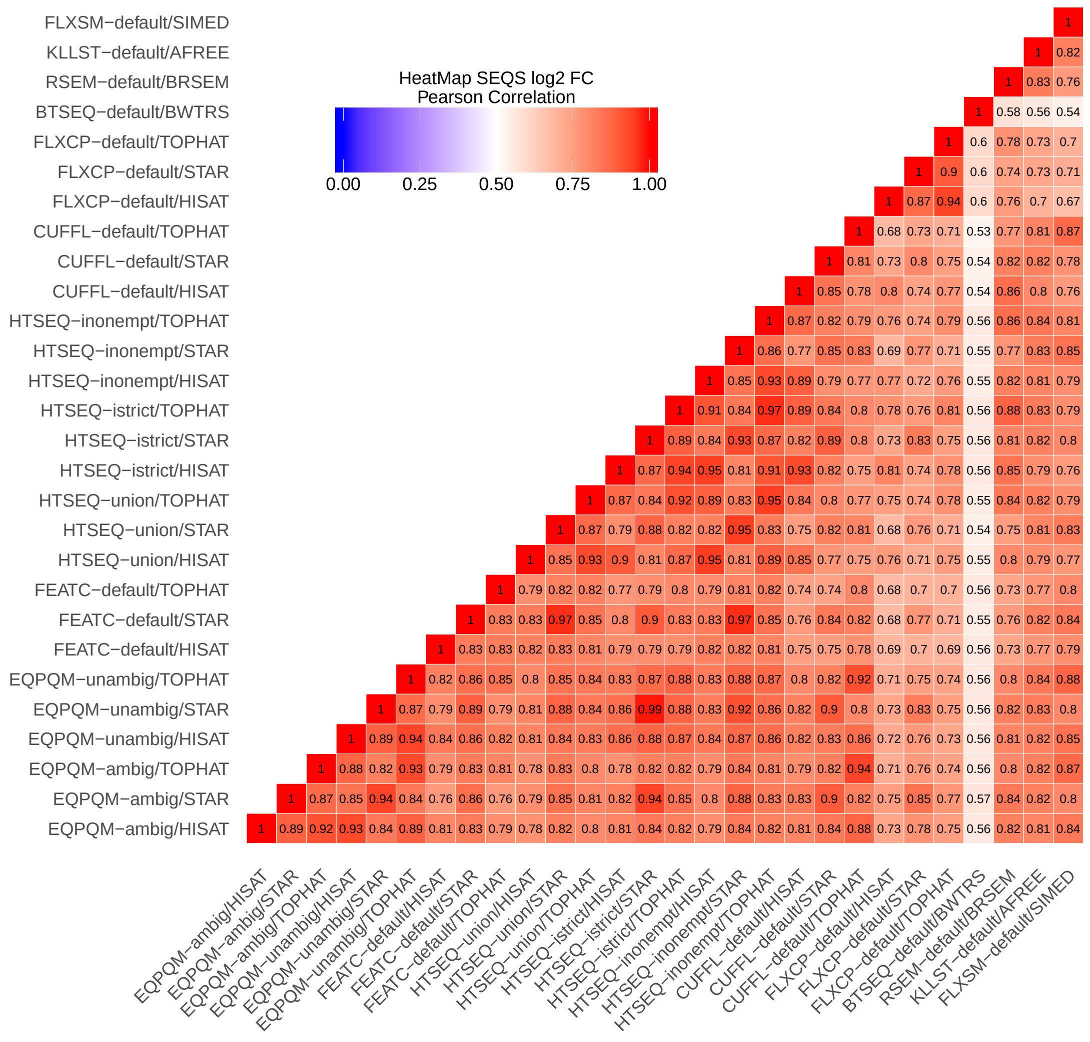

FIGURE A:17: SEQS Pearson FC heatmap. This heatmap shows then logarithmic fold change Pearson correlations of all methods to one another with the SEQS data set. A darker red reveals a higher correlation. A description of tool abbreviations can be found at 6.2. The plot is referred to in Section 8.3.2.

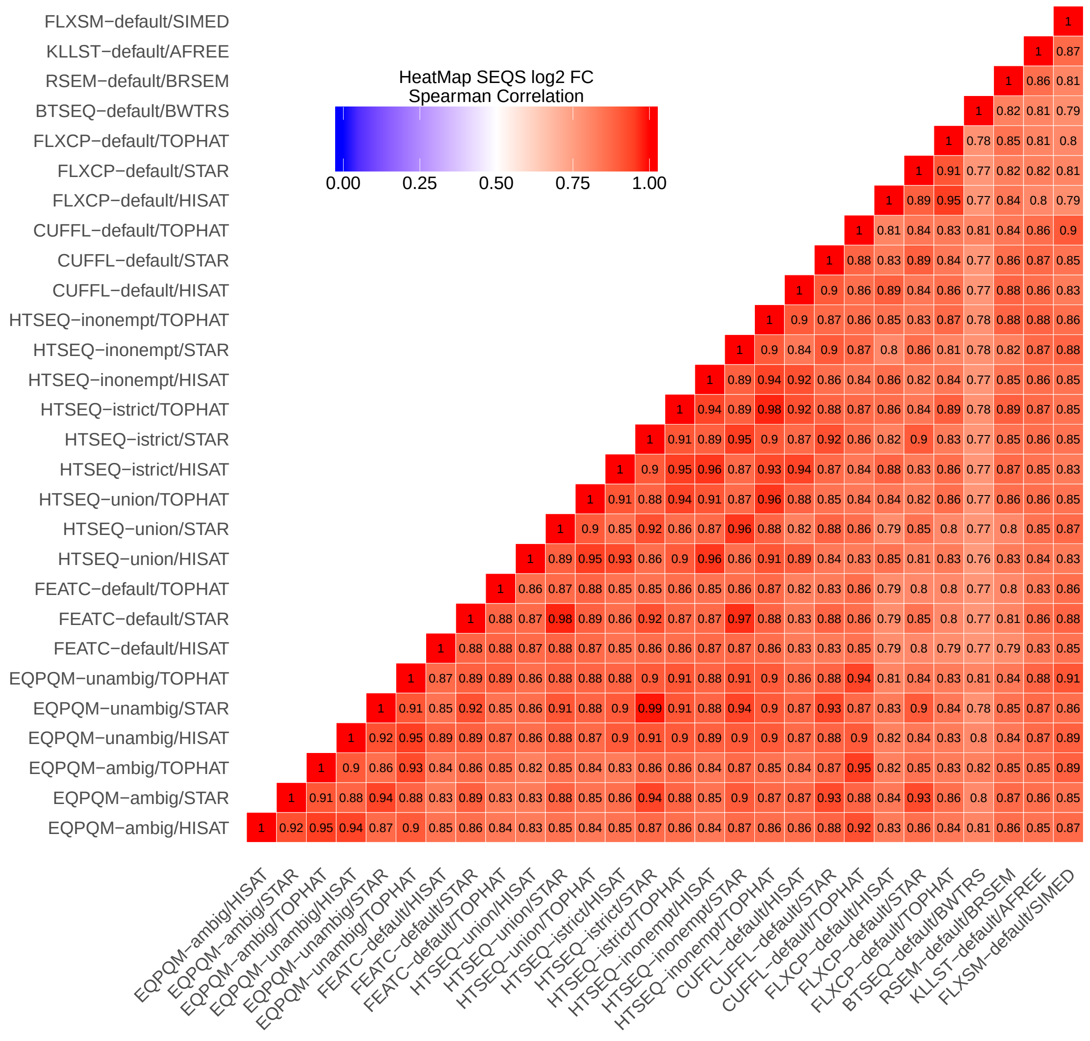

FIGURE A:18: SEQS Spearman FC heatmap. This heatmap shows then logarithmic fold change Spearman correlations of all methods to one another with the SEQS data set. A darker red reveals a higher correlation. A description of tool abbreviations can be found at 6.2. The plot is referred to in Section 8.3.2.

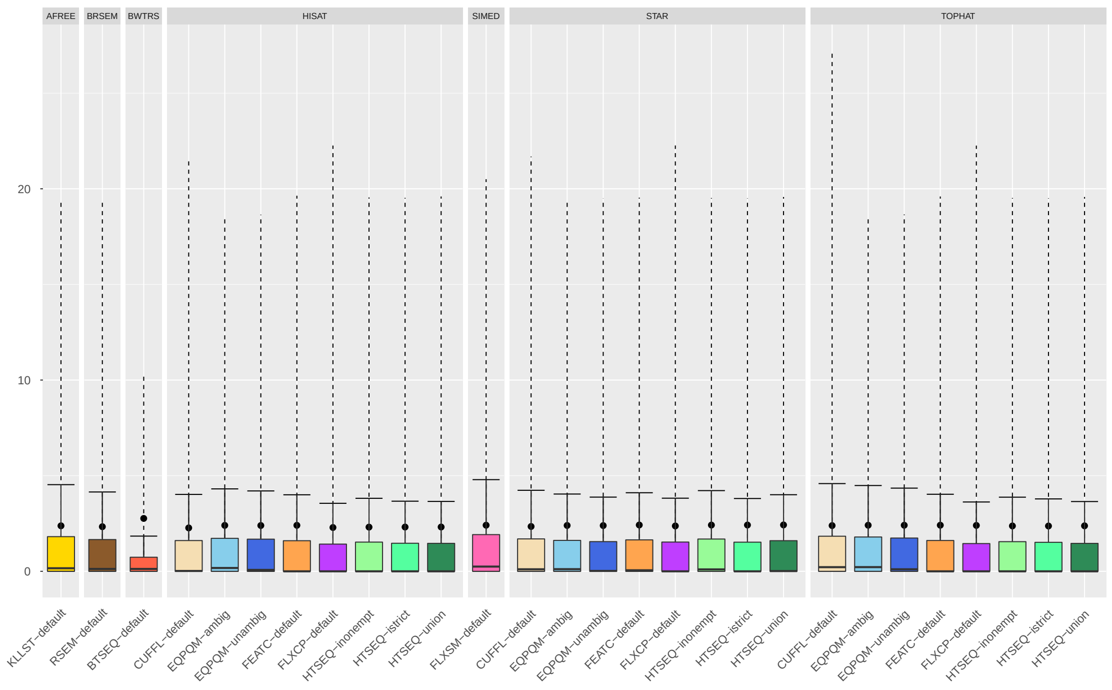

Figure A:19: SEQS ABS FC boxplot. These boxplots are showing the distribution of absolute logarithmic fold change values of the SEQS data set between all 64253 different genes. The colors provide a guideline for different quantifiers and settings of quantifiers. The "SIMED" column with "FLXSM-default" is showing the fold changes of the simulations from the Flux Simulator profile file. A description of tool abbreviations can be found at 6.2. The plot is referred to in Section 8.3.3.

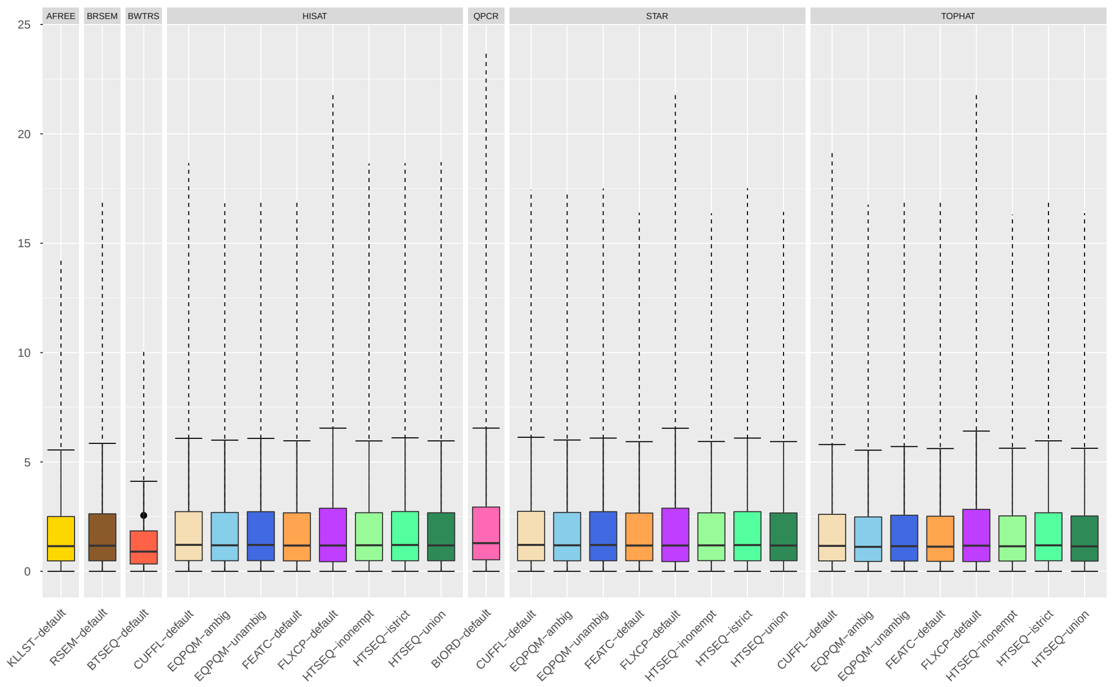

FIGURE A:20: Bio-Rad ABS FC boxplot. These boxplots are showing the distribution of absolute logarithmic fold change values of the SEQC data set between all 18325 different genes represented in the Bio-Rad data set. The colors provide a guideline for different quantifiers and settings of quantifiers. The "QPCR" column with "BIORD-default" is showing the fold changes of the Bio-Rad data. A description of tool abbreviations can be found at 6.2. The plot is referred to in Section 8.3.3.

	P CC A	P CC B	S CC A	S CC B	R SD A	R SD B
STAR						
EQPQM A	0.52	0.77	0.86	0.86	1.54	1.83
EQPQM U	0.53	0.78	0.87	0.86	1.47	1.75
FEATC	0.56	0.79	0.87	0.86	1.52	1.77
HTSEQ N	0.56	0.79	0.86	0.86	1.51	1.77
HTSEQ S	0.54	0.78	0.87	0.86	1.50	1.75
HTSEQ E	0.54	0.78	0.87	0.86	1.51	1.79
CUFFL	0.55	0.82	0.87	0.87	1.41	1.67
FLXCP	0.45	0.69	0.86	0.86	1.60	1.98
HISAT						
EQPQM A	0.48	0.76	0.87	0.86	1.49	1.80
EQPQM U	0.49	0.77	0.87	0.87	1.40	1.80
FEATC	0.51	0.78	0.87	0.86	1.41	1.79
HTSEQ N	0.60	0.78	0.86	0.86	1.62	1.78
HTSEQ S	0.58	0.77	0.87	0.87	1.60	1.77
HTSEQ E	0.58	0.77	0.87	0.86	1.61	1.80
CUFFL	0.52	0.81	0.87	0.87	1.37	1.76
FLXCP	0.41	0.68	0.86	0.86	1.51	2.01
TOPHAT						
EQPQM A	0.50	0.76	0.86	0.85	1.53	1.92
EQPQM U	0.51	0.77	0.87	0.86	1.45	1.83
FEATC	0.53	0.78	0.86	0.86	1.46	1.82
HTSEQ N	0.52	0.78	0.86	0.86	1.44	1.82
HTSEQ S	0.51	0.77	0.87	0.86	1.43	1.80
HTSEQ E	0.51	0.77	0.87	0.86	1.44	1.85
CUFFL	0.54	0.81	0.87	0.86	1.63	1.98
FLXCP	0.43	0.69	0.86	0.86	1.53	2.00
BWTRS						
BTSEQ	0.50	0.77	0.87	0.86	1.33	2.00
BRSEM						
RSEM	0.50	0.76	0.87	0.86	1.46	1.81
AFREE						
KLLST	0.49	0.76	0.87	0.86	1.43	1.86

Table A:21: Bio-Rad count accuracy. ”A” and ”B” are describing sample sets. ”P CC” describes Pearson correlation coefficient. ”S CC” describes Spearman correlation coefficient. ”R SD” describes the ratio of standard deviations. A description of tool abbreviations can be found at 6.2 and 6.1. The plot is referred to in Section 8.3.3.

	P CC A	P CC B	S CC A	S CC B	R SD A	R SD B
STAR						
EQPQM A	0.59	0.83	0.83	0.83	0.76	0.40
EQPQM U	0.58	0.83	0.83	0.83	0.68	0.27
FEATC	0.57	0.83	0.82	0.82	0.73	0.30
HTSEQ N	0.57	0.82	0.81	0.82	0.73	0.27
HTSEQ S	0.57	0.83	0.83	0.83	0.71	0.29
HTSEQ E	0.57	0.83	0.83	0.83	0.73	0.32
CUFFL	0.62	0.84	0.84	0.84	0.76	0.41
FLXCP	0.57	0.82	0.82	0.82	0.65	0.37
HISAT						
EQPQM A	0.59	0.83	0.83	0.83	0.81	0.43
EQPQM U	0.59	0.83	0.83	0.83	0.73	0.31
FEATC	0.57	0.82	0.82	0.82	0.76	0.31
HTSEQ N	0.57	0.82	0.81	0.82	0.74	0.28
HTSEQ S	0.57	0.83	0.83	0.83	0.73	0.29
HTSEQ E	0.57	0.83	0.83	0.83	0.74	0.33
CUFFL	0.62	0.84	0.84	0.84	0.82	0.45
FLXCP	0.58	0.82	0.82	0.82	0.72	0.43
TOPHAT						
EQPQM A	0.59	0.83	0.83	0.82	0.78	0.45
EQPQM U	0.58	0.83	0.83	0.83	0.68	0.31
FEATC	0.57	0.83	0.82	0.82	0.73	0.32
HTSEQ N	0.57	0.82	0.81	0.82	0.73	0.31
HTSEQ S	0.57	0.83	0.83	0.83	0.72	0.32
HTSEQ E	0.57	0.83	0.82	0.82	0.73	0.36
CUFFL	0.61	0.84	0.84	0.83	0.99	0.66
FLXCP	0.57	0.82	0.82	0.82	0.66	0.41
BWTRS						
BTSEQ	0.59	0.83	0.86	0.86	0.60	0.56
BRSEM						
RSEM	0.60	0.83	0.87	0.87	0.76	0.33
AFREE						
KLLST	0.59	0.83	0.87	0.86	0.72	0.38

Table A:22: TaqMan count accuracy. ”A” and ”B” are describing sample sets. ”P CC” describes Pearson correlation coefficient. ”S CC” describes Spearman correlation coefficient. ”R SD” describes the ratio of standard deviations. A description of tool abbreviations can be found at 6.2 and 6.1. The plot is referred to in Section 8.3.3.

	P CC A	P CC B	S CC A	S CC B	R SD A	R SD B
STAR						
EQPQM A	0.99	0.99	0.96	0.96	0.07	0.06
EQPQM U	0.99	0.99	0.96	0.96	0.02	0.01
FEATC	0.99	1.00	0.95	0.95	0.01	0.04
HTSEQ N	0.99	0.99	0.94	0.94	0.02	0.02
HTSEQ S	0.99	0.99	0.95	0.95	0.02	0.02
HTSEQ E	0.99	0.99	0.96	0.96	0.03	0.01
CUFFL	0.99	0.98	0.96	0.96	0.01	0.05
FLXCP	0.97	0.95	0.94	0.93	0.11	0.20
HISAT						
EQPQM A	0.99	0.99	0.97	0.97	0.07	0.09
EQPQM U	0.99	0.99	0.97	0.97	0.01	0.02
FEATC	0.99	0.99	0.95	0.95	0.00	0.01
HTSEQ N	0.98	0.99	0.94	0.94	0.04	0.02
HTSEQ S	0.98	0.99	0.95	0.95	0.06	0.03
HTSEQ E	0.98	0.99	0.96	0.95	0.06	0.04
CUFFL	1.00	1.00	0.96	0.96	0.03	0.04
FLXCP	0.96	0.94	0.93	0.93	0.09	0.24
TOPHAT						
EQPQM A	1.00	0.99	0.97	0.97	0.08	0.09
EQPQM U	0.99	1.00	0.97	0.97	0.02	0.02
FEATC	0.99	0.99	0.95	0.95	0.01	0.00
HTSEQ N	0.99	0.99	0.94	0.94	0.01	0.01
HTSEQ S	0.99	0.99	0.95	0.96	0.01	0.03
HTSEQ E	0.99	0.99	0.96	0.96	0.01	0.04
CUFFL	0.43	0.78	0.98	0.98	1.12	0.27
FLXCP	0.97	0.94	0.94	0.93	0.08	0.22
BWTRS						
BTSEQ	0.99	1.00	0.87	0.87	0.20	0.28
BRSEM						
RSEM	0.99	0.99	0.91	0.91	0.05	0.06
AFREE						
KLLST	0.99	0.99	0.94	0.94	0.03	0.03

Table A:23: SEQS count accuracy. ”A” and ”B” are describing sample sets. ”P CC” describes Pearson correlation coefficient. ”S CC” describes Spearman correlation coefficient. ”R SD” describes the ratio of standard deviations. A description of tool abbreviations can be found at 6.2 and 6.1. The plot is referred to in Section 8.3.3.

	P CC FC	S CC FC	R SD FC	TPR	FPR
STAR					
EQPQM A	0.79	0.84	0.13	0.76	0.23
EQPQM U	0.79	0.85	0.12	0.77	0.23
FEATC	0.79	0.85	0.16	0.76	0.23
HTSEQ N	0.79	0.85	0.16	0.76	0.23
HTSEQ S	0.79	0.85	0.12	0.77	0.23
HTSEQ E	0.79	0.85	0.16	0.76	0.23
CUFFL	0.79	0.85	0.12	0.77	0.23
FLXCP	0.75	0.83	0.15	0.76	0.23
HISAT					
EQPQM A	0.79	0.84	0.14	0.76	0.23
EQPQM U	0.79	0.85	0.13	0.77	0.23
FEATC	0.78	0.84	0.13	0.75	0.23
HTSEQ N	0.79	0.85	0.15	0.76	0.23
HTSEQ S	0.79	0.85	0.11	0.77	0.23
HTSEQ E	0.79	0.85	0.15	0.76	0.23
CUFFL	0.79	0.85	0.11	0.77	0.23
FLXCP	0.75	0.83	0.17	0.76	0.23
TOPHAT					
EQPQM A	0.77	0.84	0.23	0.72	0.23
EQPQM U	0.78	0.84	0.20	0.74	0.23
FEATC	0.77	0.84	0.20	0.72	0.23
HTSEQ N	0.78	0.84	0.21	0.73	0.23
HTSEQ S	0.79	0.85	0.14	0.76	0.23
HTSEQ E	0.78	0.84	0.21	0.73	0.23
CUFFL	0.79	0.84	0.19	0.75	0.23
FLXCP	0.75	0.83	0.14	0.75	0.25
BWTRS					
BTSEQ	0.76	0.83	0.76	0.54	0.25
BRSEM					
RSEM	0.79	0.84	0.16	0.75	0.23
AFREE					
KLLST	0.78	0.84	0.25	0.73	0.23

Table A:24: Bio-Rad fold change accuracy. "P CC" describes Pearson correlation coefficient. "S CC" describes Spearman correlation coefficient. "R SD" describes the ratio of standard deviations. "TPR" describes the true positive rate. "FPR" describes the false positive rate. A description of tool abbreviations can be found at 6.2 and 6.1. The plot is referred to in Section 8.3.3.

	P CC FC	S CC FC	R SD FC	TPR	FPR
STAR					
EQPQM A	0.90	0.91	0.16	0.87	0.28
EQPQM U	0.90	0.91	0.17	0.87	0.28
FEATC	0.91	0.90	0.12	0.86	0.29
HTSEQ N	0.91	0.90	0.12	0.85	0.29
HTSEQ S	0.90	0.91	0.17	0.87	0.28
HTSEQ E	0.91	0.90	0.12	0.86	0.29
CUFFL	0.90	0.90	0.17	0.88	0.28
FLXCP	0.87	0.90	0.50	0.86	0.29
HISAT					
EQPQM A	0.90	0.91	0.15	0.87	0.28
EQPQM U	0.91	0.91	0.16	0.87	0.28
FEATC	0.91	0.90	0.12	0.86	0.28
HTSEQ N	0.91	0.90	0.13	0.85	0.29
HTSEQ S	0.90	0.91	0.18	0.87	0.29
HTSEQ E	0.91	0.90	0.14	0.85	0.29
CUFFL	0.90	0.90	0.18	0.87	0.28
FLXCP	0.87	0.90	0.52	0.86	0.29
TOPHAT					
EQPQM A	0.90	0.91	0.06	0.82	0.28
EQPQM U	0.91	0.91	0.08	0.84	0.28
FEATC	0.90	0.90	0.07	0.83	0.28
HTSEQ N	0.91	0.90	0.08	0.84	0.29
HTSEQ S	0.91	0.90	0.16	0.86	0.29
HTSEQ E	0.91	0.90	0.09	0.84	0.28
CUFFL	0.91	0.90	0.10	0.85	0.28
FLXCP	0.87	0.90	0.49	0.86	0.30
BWTRS					
BTSEQ	0.94	0.92	0.32	0.77	0.32
BRSEM					
RSEM	0.90	0.91	0.17	0.87	0.29
AFREE					
KLLST	0.89	0.90	0.08	0.84	0.31

Table A:25: TaqMan fold change accuracy. "P CC" describes Pearson correlation coefficient. "S CC" describes Spearman correlation coefficient. "R SD" describes the ratio of standard deviations. "TPR" describes the true positive rate. "FPR" describes the false positive rate. A description of tool abbreviations can be found at 6.2 and 6.1. The plot is referred to in Section 8.3.3.

	P CC FC	S CC FC	R SD FC	TPR	FPR
STAR					
EQPQM A	0.80	0.85	0.08	0.78	0.07
EQPQM U	0.80	0.86	0.08	0.77	0.07
FEATC	0.84	0.88	0.05	0.81	0.06
HTSEQ N	0.83	0.87	0.05	0.80	0.07
HTSEQ S	0.80	0.85	0.08	0.77	0.07
HTSEQ E	0.85	0.88	0.04	0.82	0.06
CUFFL	0.78	0.85	0.01	0.78	0.07
FLXCP	0.71	0.81	0.00	0.72	0.09
HISAT					
EQPQM A	0.84	0.87	0.06	0.82	0.07
EQPQM U	0.85	0.89	0.05	0.83	0.06
FEATC	0.79	0.85	0.04	0.77	0.07
HTSEQ N	0.77	0.83	0.05	0.73	0.08
HTSEQ S	0.76	0.83	0.05	0.72	0.08
HTSEQ E	0.79	0.85	0.03	0.76	0.07
CUFFL	0.76	0.83	0.03	0.75	0.08
FLXCP	0.67	0.79	0.02	0.67	0.09
TOPHAT					
EQPQM A	0.87	0.89	0.05	0.85	0.06
EQPQM U	0.88	0.91	0.04	0.86	0.05
FEATC	0.80	0.86	0.04	0.78	0.07
HTSEQ N	0.79	0.85	0.06	0.75	0.07
HTSEQ S	0.79	0.85	0.06	0.76	0.07
HTSEQ E	0.81	0.86	0.04	0.78	0.07
CUFFL	0.87	0.90	0.05	0.87	0.06
FLXCP	0.70	0.80	0.02	0.70	0.09
BWTRS					
BTSEQ	0.54	0.79	1.46	0.30	0.12
BRSEM					
RSEM	0.76	0.81	0.04	0.74	0.10
AFREE					
KLLST	0.82	0.87	0.02	0.82	0.08

TABLE A:26: SEQS fold change accuracy. "P CC" describes Pearson correlation coefficient. "S CC" describes Spearman correlation coefficient. "R SD" describes the ratio of standard deviations. "TPR" describes the true positive rate. "FPR" describes the false positive rate. A description of tool abbreviations can be found at 6.2 and 6.1. The plot is referred to in Section 8.3.3.

Supplementary

Create Counts Scripts

`ARQ.py`
Main script for calling all intermediate steps. It accesses all other ARQ scripts and coordinates the construction of indexes, alignments and quantifications for all tools and data files.
`arq_dirs.py`
Generates all main tool directories, needed for the call.
`arq_get.py`
Contains definitions for processing the configuration data as well as the input FASTQ files and other several informations.
`arq_check.py`
Validates the input files, including the configuration file and checks the progress of the project folder.
`arq_tools.py`
Contains the configuration for all tools used in the ARQ count tool. The configuration for new tools can be added here. It also contains the basic bash script template which is used for the `qsub` tool calls of the UNIVA Grid Engine [22].
`arq_qsub.py`
Constructs, writes and calls the final bash scripts for all requested jobs.
`arq_read.py`
Contains the help section of all tools.
`arq_helper.py`
Contains various small functions.
`arq_debug.py`
Script with debug functions.

Utility Scripts

`cuff_read_mass.sh`
Extracts Cufflinks read mass (total read counts) from qsub.qout files for later FPKM to count calculations.
`get_cuff_counts.py`
Transforms Cufflinks FPKM reads into read counts.
`trans2gene.py`
Processe a GTF file and produces a sorted transcript ID to gene ID reference file.
`geno2chrom.py`
Generates a new FASTA file for each chromosome in a genome FASTA file.

Simulation Scripts

`flx_sim_qsub.sh`
Command list for simulating FASTQ reads with the Flux Simulator.
`make_flux_conf.py`
Processes a flux profile file generated in the Flux Simulator expression step and replaces the expression values with medium expression values over all Cufflinks transcript count files.
`prep_flux_gtf.py`
Modifies a GTF file for the usage in the Flux Simulator.
`split_flux_reads.py`
Splits flux FASTQ paired end files into two separate files.

Compare Counts Scripts

`compare-counts.R`
Main script for generating a statistical analysis and plots. Can directly access the ARQ folder structure and extract reads with the help of a configuration file. The analysis steps can be defined at the beginning of the script.
`util-lib.R`
Contains functions for the statistical analysis and plotting.

Configuration Files

`arq_proj_test.conf`
This is a sample configuration file fir the ARQ pipeline.
`SEQC-sample-sequencing-information-1.txt`
Tabular files with information about the FASTQ files ARQ should use. (first read)
`SEQC-sample-sequencing-information-2.txt`
Tabular files with information about the FASTQ files ARQ should use. (second read)
`make_statistics.conf`
This is the basic configuration file for the statistics script.
`flx_sim.conf`
This is the used configuration file for the flux simulations.

Data Files

`cuff_read_mass.data`
Contains Cufflinks total read mass for all SEQC runs. Used in recalculation of Cufflinks counts.
`cuff_read_mass_sim.data`
Contains Cufflinks total read mass for all SEQS runs. Used in recalculation of Cufflinks counts.
`Mean_Gene_Count_Summary_for_SEQC_BIORD.txt`
Contains the mean gene count summary for all Bio-Rad gene counts.
`Mean_Gene_Count_Summary_for_SEQC_TQMAN.txt`
Contains the mean gene count summary for all TaqMan gene counts.
`ng-human-GRCh38-Ensembl-glen.txt`
Contains gene lengths for alle genes.
`ng-human-GRCh38-Ensembl-trans2gene.txt`
Transcript ID to gene ID reference file.

Genome Files (not provided)

`ensemble-Homo_sapiens-extended-mod.fa`
Modified genome reference FASTA file.

`ensembl-Homo_sapiens.GRCh38.78.transcript.fa`
Transcriptome file combined from the cDNA and ncRNA files.
`ng-human-GRCh38-Ensembl.gtf`
GTF genome annotation file.

SEQC Count Files

```
SEQC_Gene_Count_Matrix_EQPQM-ambig-HISAT.txt
SEQC_Gene_Count_Matrix_EQPQM-ambig-STAR.txt
SEQC_Gene_Count_Matrix_EQPQM-ambig-TOPHAT.txt
SEQC_Gene_Count_Matrix_EQPQM-unambig-HISAT.txt
SEQC_Gene_Count_Matrix_EQPQM-unambig-STAR.txt
SEQC_Gene_Count_Matrix_EQPQM-unambig-TOPHAT.txt
SEQC_Gene_Count_Matrix_FEATC-default-HISAT.txt
SEQC_Gene_Count_Matrix_FEATC-default-STAR.txt
SEQC_Gene_Count_Matrix_FEATC-default-TOPHAT.txt
SEQC_Gene_Count_Matrix_HTSEQ-union-HISAT.txt
SEQC_Gene_Count_Matrix_HTSEQ-union-STAR.txt
SEQC_Gene_Count_Matrix_HTSEQ-union-TOPHAT.txt
SEQC_Gene_Count_Matrix_HTSEQ-istrict-HISAT.txt
SEQC_Gene_Count_Matrix_HTSEQ-istrict-STAR.txt
SEQC_Gene_Count_Matrix_HTSEQ-istrict-TOPHAT.txt
SEQC_Gene_Count_Matrix_HTSEQ-inonempt-HISAT.txt
SEQC_Gene_Count_Matrix_HTSEQ-inonempt-STAR.txt
SEQC_Gene_Count_Matrix_HTSEQ-inonempt-TOPHAT.txt
SEQC_Gene_Count_Matrix_CUFFL-default-HISAT.txt
SEQC_Gene_Count_Matrix_CUFFL-default-STAR.txt
SEQC_Gene_Count_Matrix_CUFFL-default-TOPHAT.txt
SEQC_Gene_Count_Matrix_FLXCP-default-HISAT.txt
SEQC_Gene_Count_Matrix_FLXCP-default-STAR.txt
SEQC_Gene_Count_Matrix_FLXCP-default-TOPHAT.txt
SEQC_Gene_Count_Matrix_BTSEQ-default-BWTRS.txt
SEQC_Gene_Count_Matrix_RSEM-default-BRSEM.txt
SEQC_Gene_Count_Matrix_KLLST-default-AFREE.txt
```

SEQS Count Files

```
SEQS_Gene_Count_Matrix_EQPQM-ambig-HISAT.txt
SEQS_Gene_Count_Matrix_EQPQM-ambig-STAR.txt
SEQS_Gene_Count_Matrix_EQPQM-ambig-TOPHAT.txt
SEQS_Gene_Count_Matrix_EQPQM-unambig-HISAT.txt
SEQS_Gene_Count_Matrix_EQPQM-unambig-STAR.txt
SEQS_Gene_Count_Matrix_EQPQM-unambig-TOPHAT.txt
SEQS_Gene_Count_Matrix_FEATC-default-HISAT.txt
SEQS_Gene_Count_Matrix_FEATC-default-STAR.txt
SEQS_Gene_Count_Matrix_FEATC-default-TOPHAT.txt
SEQS_Gene_Count_Matrix_HTSEQ-union-HISAT.txt
SEQS_Gene_Count_Matrix_HTSEQ-union-STAR.txt
SEQS_Gene_Count_Matrix_HTSEQ-union-TOPHAT.txt
SEQS_Gene_Count_Matrix_HTSEQ-istrict-HISAT.txt
SEQS_Gene_Count_Matrix_HTSEQ-istrict-STAR.txt
SEQS_Gene_Count_Matrix_HTSEQ-istrict-TOPHAT.txt
SEQS_Gene_Count_Matrix_HTSEQ-inonempt-HISAT.txt
SEQS_Gene_Count_Matrix_HTSEQ-inonempt-STAR.txt
SEQS_Gene_Count_Matrix_HTSEQ-inonempt-TOPHAT.txt
SEQS_Gene_Count_Matrix_CUFFL-default-HISAT.txt
SEQS_Gene_Count_Matrix_CUFFL-default-STAR.txt
SEQS_Gene_Count_Matrix_CUFFL-default-TOPHAT.txt
SEQS_Gene_Count_Matrix_FLXCP-default-HISAT.txt
SEQS_Gene_Count_Matrix_FLXCP-default-STAR.txt
SEQS_Gene_Count_Matrix_FLXCP-default-TOPHAT.txt
SEQS_Gene_Count_Matrix_BTSEQ-default-BWTRS.txt
SEQS_Gene_Count_Matrix_RSEM-default-BRSEM.txt
SEQS_Gene_Count_Matrix_KLLST-default-AFREE.txt
SEQS_Gene_Count_Matrix_FLXSM-default-SIMED.txt
```

References

[1] Teng M, Love MI, Davis CA, Djebali S, Dobin A, Graveley BR, Li S, Mason CE, Olson S, Pervouchine D, Sloan CA, Wei X, Zhan L, Irizarry RA (2016). *A benchmark for RNA-seq quantification pipelines.* Genome Bio. 17(1):74.

[2] Fonseca NA, Marioni J, Brazma A (2014). *RNA-Seq Gene Profiling - A Systematic Empirical Comparison.* PLoS One 9(9):e107026.

[3] Kanitz A, Gypas F, Gruber AJ, Gruber AR, Martin G, Zavolan M (2015). *Comparative assessment of methods for the computational inference of transcript isoform abundance from RNA-seq data.* Genome Bio. 16:150.

[4] Kim D, Pertea G, Trapnell C, Pimentel H, Kelley R, Salzberg SL (2013). *TopHat2: accurate alignment of transcriptomes in the presence of insertions, deletions and gene fusions.* Genome Bio. 14(4):R36.

[5] Kim D, Langmead B, Salzberg SL (2015). *HISAT: a fast spliced aligner with low memory requirements.* Nat Methods. 12(4):357-60.

[6] Dobin A, Davis CA, Schlesinger F, Drenkow J, Zaleski C, Jha S, Batut P, Chaisson M, Gingeras TR (2013). *STAR: ultrafast universal RNA-seq aligner.* Bioinformatics. 29(1):15-21.

[7] Langmead B, Salzberg SL (2012). *Fast gapped-read alignment with Bowtie 2.* Nat Methods. 9(4):357-9.

[8] Schuierer S, Roma G (2015). *The Exon Quantification Pipeline (EQP): a comprehensive approach to the quantification of gene, exon, and junction expression from RNA-seq data.* Novartis Report.

[9] Anders S, Pyl PT, Huber W (2015). *HTSeq A Python framework to work with high-throughput sequencing data.* Bioinformatics. 31(2):166-169.

[10] Liao Y, Smyth GK, Shi W (2014). *featureCounts: an efficient general purpose program for assigning sequence reads to genomic features.* Bioinformatics. 30 (7):923-930.

[11] Trapnell C, Williams BA, Pertea G, Mortazavi A, Kwan G, van Baren MJ, Salzberg SL, Wold BJ, Pachter L (2010). *Transcript assembly and quantification by RNA-Seq reveals unannotated transcripts and isoform switching during cell differentiation.* Nat Biotechnol. 28(5):511515.

[12] Montgomery SB, Sammeth M, Gutierrez-Arcelus M, Lach RP, Ingle C, Nisbett J, Guigo R, Dermitzakis ET (2010). *Transcriptome genetics using second generation sequencing in a Caucasian population.* Nature. 464(7289):773-7.

[13] Glaus P, Honkela A, Rattray M (2012). *Identifying differentially expressed transcripts from RNA-seq data with biological variation.* Bioinformatics. 28(13):1721-1728.

[14] Li B, Dewey CN (2011). *RSEM: accurate transcript quantification from RNA-Seq data with or without a reference genome.* BMC Bioinformatics. 12:323.

[15] Bray NL, Pimentel H, Melsted P, Pachter L (2016). *Near-optimal probabilistic RNA-seq quantification.* Nat Biotechnol. 34(5):525-7.

[16] Griebel T, Zacher B, Ribeca P, Raineri E, Lacroix V, Guigó R, Sammeth M (2012). *Modelling and simulating generic RNA-Seq experiments with the flux simulator.* Nucleic Acids Res. 40(20):10073-83.

[17] Anders S, Huber W (2010). *Differential expression analysis for sequence count data.* Genome Biol. 11(10):R106.

[18] Brueffer C, Saal LH (2016). *TopHat-Recondition: a post-processor for TopHat unmapped reads.* BMC Bioinformatics. 17(1):199.

[19] Li H, Handsaker B, Wysoker A, Fennell T, Ruan J, Homer N, Marth G, Abecasis G, Durbin R (2009). *The Sequence Alignment/Map format and SAMtools.* Bioinformatics. 25(16):2078-9.

[20] Trapnell C, Roberts A, Goff L, Pertea G, Kim D, Kelley DR, Pimentel H, Salzberg SL, Rinn JL, Pachter L (2012). *Differential gene and transcript expression analysis of RNA-seq experiments with TopHat and Cufflinks.* Nat Protoc. 7(3):562-78.

[21] Cunningham F, Amode MR, Barrell D, Beal K, Billis K, Brent S, Carvalho-Silva D, Clapham P, Coates G, Fitzgerald S, et al. (2015). *Ensembl 2015.*

[22] Tommaso PD, Ramirez AB, Palumbo E, Notredame C, Gruber D (2011). *Benchmark Report: Univa Grid Engine, Nextfow, and Docker for running Genomic Analysis Workfows.* Univa Report.

[23] Ferragina P, Manzini G (2000). *Proceedings of the 41st Symposium on Foundations of Computer Science.* IEEE Comput Soc. 390398

[24] Langmead B, Trapnell C, Pop M, Salzberg SL (2009). *Ultrafast and memory-efficient alignment of short DNA sequences to the human genome.* Genome Biol. 10(3):R25.

[25] Quinlan AR, Hall IM (2010). *BEDTools: a flexible suite of utilities for comparing genomic features.* Bioinformatics. 26(6):841-2.

[26] Milo R, Jorgensen P, Moran U, Weber G, Springer M (2010). *BioNumbers–the database of key numbers in molecular and cell biology.* Nucleic Acids Res. (Database issue):D750-3.

[27] Au KF, Sebastiano V, Afshar PT, Durruthy JD, Lee L, Williams BA, van Bakel H, Schadt EE, Reijo-Pera RA, Underwood JG, Wong WH (2013). *Characterization of the human ESC transcriptome by hybrid sequencing.* Proc Natl Acad Sci U S A. 110(50):E4821-30.

[28] Wang Z, Gerstein M, Snyder M (2009). *RNA-Seq: a revolutionary tool for transcriptomics.* Nat Rev Genet. 10(1):57-63.

[29] Li H1, Homer N (2010). *A survey of sequence alignment algorithms for next-generation sequencing.* Brief Bioinform. 11(5):473-83.

[30] Xing J1, Zhang Y, Han K, Salem AH, Sen SK, Huff CD, Zhou Q, Kirkness EF, Levy S, Batzer MA, Jorde LB (2009). *Mobile elements create structural variation: analysis of a complete human genome.* Genome Res. 19(9):1516-26.

[31] Levy S1, Sutton G, Ng PC, Feuk L, Halpern AL, Walenz BP, Axelrod N, Huang J, Kirkness EF, Denisov G, et al. (2007). *The diploid genome sequence of an individual human.* PLoS Biol. 5(10):e254.

[32] Li B, Ruotti V, Stewart RM, Thomson JA, Dewey CN (2010). *RNA-Seq gene expression estimation with read mapping uncertainty.* Bioinformatics. 26(4):493-500.

[33] SEQC/MAQC-III Consortium (2014). *A comprehensive assessment of RNA-seq accuracy, reproducibility and information content by the Sequencing Quality Control Consortium.* Nat Biotechnol. 32(9):903-14.

[34] MAQC Consortium, Shi L, Reid LH, Jones WD, Shippy R, Warrington JA, Baker SC, Collins PJ, de Longueville F, Kawasaki ES, et al. (2006). *The MicroArray Quality Control (MAQC) project shows inter- and intraplatform reproducibility of gene expression measurements.* Nat Biotechnol. 24(9):1151-61.

[35] Patro R, Mount SM, Kingsford C (2014). *Sailfish enables alignment-free isoform quantification from RNA-seq reads using lightweight algorithms.* Nat Biotechnol. 32(5):462-4.

[36] Norel R, Rice JJ, Stolovitzky G (2011). *The self-assessment trap: can we all be better than average?* Mol Syst Biol. 7:537.

[37] Roberts A, Pachter L (2013). *Streaming fragment assignment for real-time analysis of sequencing experiments.* Nat Methods. 10(1):71-3.

[38] Rob Patro, Geet Duggal, Carl Kingsford (2015). *Salmon: Accurate, Versatile and Ultrafast Quantification from RNA-seq Data using Lightweight-Alignment.* bioRxiv. 2015.

[39] Mortazavi A, Williams BA, McCue K, Schaeffer L, Wold B (2008). *Mapping and quantifying mammalian transcriptomes by RNA-Seq.* Nat Methods. 5(7):621-8.

[40] Lappalainen T, Sammeth M, Friedlnder MR, 't Hoen PA, Monlong J, Rivas MA, Gonzàlez-Porta M, Kurbatova N, Griebel T, Ferreira PG, et al. (2013). *Transcriptome and genome sequencing uncovers functional variation in humans.* Nature. 501(7468):506-11.

[41] Kumamaru H, Ohkawa Y, Saiwai H, Yamada H, Kubota K, Kobayakawa K, Akashi K, Okano H, Iwamoto Y, Okada S (2012). *Direct isolation and RNA-seq reveal environment-dependent properties of engrafted neural stem/progenitor cells.* Nat Commun. 3:1140.

[42] Marinov GK1, Williams BA, McCue K, Schroth GP, Gertz J, Myers RM, Wold BJ (2014). *From single-cell to cell-pool transcriptomes: stochasticity in gene expression and RNA splicing.* Genome Res. 24(3):496-510.

[43] Hu J, Ge H, Newman M, Liu K (2012). *OSA: a fast and accurate alignment tool for RNA-Seq.* Bioinformatics. 28(14):1933-4.

[44] Wu TD, Nacu S 2010). *Fast and SNP-tolerant detection of complex variants and splicing in short reads.* Bioinformatics. 26(7):873-81.

[45] Ponstingl H (2010). *SMALT An efficient and accurate aligner for DNA sequencing reads.* Sanger Report.

[46] Li W, Jiang T (2012). *Transcriptome assembly and isoform expression level estimation from biased RNA-Seq reads.* Bioinformatics. 28(22):2914-21.

[47] Roberts A, Pachter L (2013). *Streaming fragment assignment for real-time analysis of sequencing experiments.* Nat Methods. 10(1):71-3.

[48] Nicolae M, Mangul S, Mndoiu II, Zelikovsky A (2011). *Estimation of alternative splicing isoform frequencies from RNA-Seq data.* Algorithms Mol Biol. 6(1):9.

[49] Turro E, Su SY, Gonçalves Â, Coin LJ, Richardson S, Lewin A (2011). *Haplotype and isoform specific expression estimation using multi-mapping RNA-seq reads.* Genome Biol. 12(2):R13. doi: 10.

[50] Jiang H1, Wong WH (2009). *Statistical inferences for isoform expression in RNA-Seq.* Bioinformatics. 25(8):1026-32.

[51] Guttman M, Garber M, Levin JZ, Donaghey J, Robinson J, Adiconis X, Fan L, Koziol MJ, Gnirke A, Nusbaum C, et al. (2010). *Ab initio reconstruction of cell type-specific transcriptomes in mouse reveals the conserved multi-exonic structure of lincRNAs.* Nat Biotechnol. 28(5):503-10.

[52] Nariai N, Kojima K, Mimori T, Sato Y, Kawai Y, Yamaguchi-Kabata Y, Nagasaki M (2014). *TIGAR2: sensitive and accurate estimation of transcript isoform expression with longer RNA-Seq reads.* BMC Genomics. 15 Suppl 10:S5.