ARSDA: A New Approach for Storing, Transmitting, and Analyzing Transcriptomic Data

Xuhua Xia1
Department of Biology, University of Ottawa, Ontario K1N 6N5, Canada and Ottawa Institute of Systems Biology, Ontario K1H 8M5, Canada
ORCID ID: 0000-0002-3092-7566 (X.X.)

ABSTRACT Two major stumbling blocks exist in high-throughput sequencing (HTS) data analysis. The first is the sheer file size, typically in gigabytes when uncompressed, causing problems in storage, transmission, and analysis. However, these files do not need to be so large, and can be reduced without loss of information. Each HTS file, either in compressed .SRA or plain text .fastq format, contains numerous identical reads stored as separate entries. For example, among 44,603,541 forward reads in the SRR4011234.sra file (from a Bacillus subtilis transcriptomic study) deposited at NCBI’s SRA database, one read has 497,027 identical copies. Instead of storing them as separate entries, one can and should store them as a single entry with the SeqID_NumCopy format (which I dub as FASTA+ format). The second is the proper allocation of reads that map equally well to paralogous genes. I illustrate in detail a new method for such allocation. I have developed ARSDA software that implement these new approaches. A number of HTS files for model species are in the process of being processed and deposited at http://coevol.rdc.uottawa.ca to demonstrate that this approach not only saves a huge amount of storage space and transmission bandwidth, but also dramatically reduces time in downstream data analysis. Instead of matching the 497,027 identical reads separately against the B. subtilis genome, one only needs to match it once. ARSDA includes functions to take advantage of HTS data in the new sequence format for downstream data analysis such as gene expression characterization. I contrasted gene expression results between ARSDA and Cufflinks so readers can better appreciate the strength of ARSDA. ARSDA is freely available for Windows, Linux, and Macintosh computers at http://dambe.bio.uottawa.ca/ARSDA/ARSDA.aspx.

KEYWORDS transcriptomics novel storage solution quantifying expression of paralogous genes sequence format ARSDA

High-throughput sequencing (HTS) is now used not only in characterizing differential gene expression, but also in many other fields, where it replaces the traditional microarray approach. Ribosomal profiling, traditionally done through microarray (Arava et al. 2003; MacKay et al. 2004), is now almost exclusively done with deep sequencing of ribosome-protected segments of messages (Ingolia et al. 2009, 2011), although the results from the two approaches for ribosomal profiling are largely concordant (Xia et al. 2011). Similarly, EST-based (Rogers et al. 2012) and microarray-based (Pleiss et al. 2007) methods for detecting alternative splicing events and characterizing splicing efficiency is now replaced by HTS (Kawashima et al. 2014), especially by lariat sequencing (Awan et al. 2013; Stepankiv et al. 2015). The availability of HTS data has dramatically accelerated the test of biological hypotheses. For example, a recent study on alternative splicing (Vlasschaert et al. 2016) showed that skipping of exon 7 (E7) in human and mouse USP4 is associated with weak signals of splice sites flanking E7. The researchers predicted that, in some species where the splice site signals are strong, E7 skipping would disappear. This prediction is readily tested and confirmed with existing HTS data, i.e., E6-E8 mRNA was found in species with weak splice signals flanking E7, and E6-E7-E8 mRNA in species with strong splice signals flanking E7 (Vlasschaert et al. 2016).

In spite of the potential of HTS data in solving practical biological problems, severe undersusage of HTS data has been reported (GB Editorial Team 2011). One major stumbling block in using the HTS data...
are the large file size. Among the 6472 HTS studies on human available at NCBI/DDBJ/EBI by April 14, 2016, 196 studies each contribute >1 Terabyte (TB) of nucleotide bases, with the top one contributing 86.4 TB. Few laboratories would be keen on downloading and analyzing this 86.4 TB of nucleotides, not to mention comparing this study to HTS data from other human HTS studies.

The explosive growth of HTS data in recent years has caused serious problems in data storage, transmission, and analysis (Leinonen et al. 2011; Kodama et al. 2012). Because of the high cost of maintaining such data, coupled with the fact that few researchers had been using such data, NCBI had planned the closure of the sequence read archive a few years ago (GB Editorial Team 2011), but continued the support only after DDBJ and EBI decided to continue their effort of archiving the data. The incident highlights the prohibitive task of storing, transmitting, and analyzing HTS data, and motivated the joint effort of both industry and academics to search for data compression solutions (Janin et al. 2014; Zhu et al. 2015b; Numanagic et al. 2016).

HTS data files do not need to be so huge. Take, for example, the characterized transcriptomic data for Escherichia coli K12 in the file SRR1536586.sra (where SRR1536586 is the SRA sequence file ID in NCBI/DDBJ/EBI). The file contains 6,503,557 sequence reads of 50 nt each, but 195,310 sequences are all identical (TGGTATCACGGGACACACGCGGGTGTAACGTCCGTCTGAAAGGG), all mapping exactly to sites 929–978 in E. coli 23S rRNA genes (the study did use the MICROBExpress Bacterial mRNA Enrichment Kit to remove the 16S and 235 rRNA, otherwise there would be many more). There are much more extreme cases. For example, one of the 12 HTS files from a transcriptomic study of E. coli (SRR822264.sra), contains a read with 1,606,515 identical copies among its 9,690,570 forward reads. There is no sequence information lost if all these 1,606,515 identical reads are stored by a single sequence with a sequence ID such as UniqueSeqX_1606515 (i.e., SeqID_CopyNumber format which I dub FASTA+ format with file type .fasP). Such storage scheme not only results in dramatic saving in data storage and transmission, but also leads to dramatic reduction in computational time in downstream data analysis. At present, all software packages for HTS data analysis will take the 1,606,515 identical reads separately, and search them individually against the E. coli genome (or target gene sequences such as coding sequences). The SeqID_CopyNumber storage scheme reduces the 1,606,515 searches to a single one.

A huge chunk of SRA data stored in NCBI/DDBJ/EBI consists of ribosome profiling data (Ingolia et al. 2009, 2011), which is obtained by sequencing the mRNA segment (~30 bases) protected by the ribosome after digesting all the unprotected mRNAs segments. Mapping these ribosome-protected segments to the genome allows one to know specifically where the ribosomes are located along individual mRNAs. In general, such data are essential to understand translation initiation, elongation, and termination efficiencies. For example, a short poly(A) segment with about eight or nine consecutive A immediately upstream of the start codon in yeast (Sacharomyces cerevisiae) genes is significantly associated with ribosome density and occupancy (Xia et al. 2011), confirming the hypothesis that short poly(A) upstream of the start codon facilitates the recruitment of translation initiation factors, but long poly(A) would bind to poly(A)-binding protein and interfere with cap-dependent translation. Sequence redundancy is high in such ribosomal profiling data and the FASTA+ format can lead to dramatic saving in the disk space of data storage and time in data transmission.

Aside from the file size problem, HTS data analysis also suffers from the problem of how to allocate multiple-matched reads to paralogous genes (Trapnell et al. 2013; Rogozin et al. 2014). The commonly used options of ignoring such multiple-matched reads or allocating them equally among matched paralogous genes are unsatisfactory. The software ARSDA I present here offers solutions to both the problem of file size and the problem of read allocation to paralogous genes.

**ARSDA**

I developed software ARSDA (for Analyzing RNA-Seq Data, Figure 1A) to alleviate the problem associated with storage, transmission and analysis of HTS data. ARSDA can take input .SRA files or .fastq files of many gigabytes, build an efficient dictionary of unique sequence reads in a FASTA/FASTQ file, keep track of their copy numbers, and output them to a FASTA+ file in the SeqID_CopyNumber format (Figure 1B). Both fixed-length and variable-length sequences can be used as input. In addition, I have implemented functions in ARSDA to take advantage of the new sequence format to perform gene expression, with the main objective of demonstrating how much faster downstream data analysis can be done with data in FASTA+/FASTQ+ format. Furthermore, ARSDA includes a unique feature in assigning shared reads among paralogous genes that I will describe below. ARSDA also includes sequence visualization functions for global base-calling quality, per-read quality, and site-specific read quality (Figure 1, C and D), but these functions are also available elsewhere, e.g., FastQC (Andrews 2017) and NGSPQC (Dai et al. 2010) and consequently will not be described further (but see the attached QuickStart.PDF). ARSDA includes nine programs in the BLAST and sratoolkit from NCBI that enhance part of ARSDA functions.

**Converting HTS data to FASTA+/FASTQ+**

The output from processing the SRR1536586.sra file (with part of the read matching output in Table 1) highlights two points. First, many sequences in the file are identical. Second, although the transcriptomic data characterized in SRR1536586 have undergone rRNA depletion by using Ambion’s MICROBExpress Bacterial mRNA Enrichment Kit (Pobre and Arraiano 2015), there are still numerous reads in the transcriptomic data that are from rRNA genes. This suggests that storing mRNA reads separately from rRNA reads can dramatically reduce file size because most researchers are not interested in the abundance of rRNAs.

While the conversion of FASTA/FASTQ files to FASTA+ files may take a few minutes, it needs to be done only once for data storage, and the resulting saving in storage space, internet traffic, and computation time in downstream data analysis is tremendous. The file size is 1.49 GB for the original FASTQ file derived from SRR1536586.sra, but is only 66 MB for the new FASTA+ file, both being plain text files.

I have applied ARSDA to reduce the file size of transcriptomic data from yeast (S. cerevisiae), nematode (Caenorhabditis elegans), fruit fly (Drosophila melanogaster), zebrafish (Danio rerio), and mouse (Mus musculus), and deposited the resulting reduced data at coevoi.rcd.uottawa.ca in the format of BLAST databases. BLAST reduces sequences further by representing tetranucleotides AAAA, AACG, ..., TTTT by 0, 1, ..., 255 so that each tetranucleotide takes only 1 byte in storage. The sequence ID in these BLAST databases are in the form of SeqID_CopyNumber. These files reduce the computation time for quantifying gene expression from many hours to only a few minutes (~2 min for my Windows 10 PC with an i7-4770 CPU at 3.4 GHz and 16 GB of RAM). This eliminates one of the key bottlenecks in HTS data analysis (Liu et al. 2016), and would make it feasible for any laboratory to gain the power of HTS data analysis. I attach the gene expression characterized by ARSDA
Figure 1 User interface in ARSDA. (A) The menu system, with database creation under the “Database” menu, gene expression characterization under the “Analysis” menu, etc. (B) Converting a FASTQ/FASTA file to a FASTQ+/FASTA+ file. (C) Site-specific read quality visualization. (D) Global read quality visualization.
for the 4321 E. coli K12 coding sequences as Supplemental Material, File S1. A part of it is reproduced in Table 2.

One may ask how quality scores are treated when reads with identical sequences are grouped into the form of SeqID_CopyNumber format. Let me first highlight two observations. First, different reads with the same sequence have similar quality scores. For example, sequence “TGTTATCAGGAGACACACGGCGGGTGCTAACGTCCGTCG TGAAGAGGG” occur many times in file SRR1526586.sra. I took the first six reads with this 50-nt sequence, and computed Pearson correlation among the associated quality scores (each read is associated with a vector of 50 quality scores). The correlation coefficients are all high and positive (Table 3). The same for sequences that occur just twice. Second, the quality scores for this combined sequence are simply average quality scores. For a sequence of length $L$ that occurred twice in the transcriptomic data, the sequence ID will be SeqID_2, and the quality scores for this combined sequence are

$$Q = \frac{Q_1 + Q_2}{2}$$

where $i = 1, 2, \ldots, L$, and $Q_1$ and $Q_2$ are quality scores from the two reads of the same sequence.

Size-reduction differs dramatically with read quality (Figure 2). For high-quality data, e.g., SRR922267 (Figure 2, where an overwhelming majority of bases are at the high-quality end), ARSDA can shrink file size to ~0.05 of the original. However, for poor-quality data, e.g., SRR5484239 (Figure 2), ARSDA can shrink file size only to 0.64 of the original. The reason is that, with high-quality data, reads from the same segment of the transcript are identical, as one would expect, but, with low-quality data, reads from the same segment of the transcript have “mutated” during the amplification and sequencing step, and are often no longer identical. For SRR922267, the most redundant sequence has 2,341,386 identical copies out of 14,872,404 forward reads. In contrast, the most redundant sequence in SRR5484239 has only 1480 identical copies out of 10,702,871 reads. This implies that the paired-end reads, especially long ones, will likely have low size-reduction efficiency because reverse read quality is typically much worse than forward read quality. Size-reduction with the ARSDA approach works best with high-quality reads. Base-calling quality typically decreases rapidly with read length (Figure 1C). Trimming off the low-quality 3’ end of the reads typically leads to dramatically increased size-reduction efficiency.

One of the frequently used sequence-compression scheme is to use a reference genome so that each read can be represented by a starting and an ending number on the genome (Renoi et al. 2015; Kingsford and Patro 2015; Zhu et al. 2015a). This approach has two problems. First, many reads do not map exactly to the genomic sequence because of either somatic mutations or sequencing errors, so representing a read by the starting and ending numbers leads to loss of information.

### Table 1 Part of read-matching output from ARSDA, with 195,310 identical reads matching a segment of large subunit (LSU) rRNA, 86,308 identical reads matching another segment of LSU rRNA, and so on

| Gene       | $N_{copy}$ | Gene | $N_{copy}$ |
|------------|------------|------|------------|
| LSU rRNA   | 195,310    | SSU rRNA | 30,417    |
| LSU rRNA   | 86,308     | LSU rRNA | 29,508    |
| LSU rRNA   | 58,400     | SS rRNA  | 28,187    |
| LSU rRNA   | 47,323     | LSU rRNA | 24,982    |
| LSU rRNA   | 45,695     | SSU rRNA | 23,286    |
| LSU rRNA   | 36,258     | LSU rRNA | 19,991    |
| SS rRNA    | 33,674     | SSU rRNA | 19,268    |

Results generated from ARSDA analysis of the SRR1536586.sra file from NCBI.

Second, RNA-editing and processing can be so extensive that it becomes impossible to map a transcriptomic read to the genome (Abraham et al. 1988; Lamond 1988; Alartorsteve et al. 2008; Li et al. 2009; Simpson et al. 2016). Furthermore, there are still many scientifically interesting species that do not have a good genomic sequence available. One could add additional annotation and indexing for sequence variants resulting from RNA-editing and “mutants” resulting from amplification and sequencing to avoid information loss, but such additional steps reduces the efficiency of compression as well as increases an extra layer of complexity for downstream data analysis.

Software tools for compressing HTS files are often benchmarked against general-purpose GZIP tools (Numanagic et al. 2016). Among nonreference compression tools for FASTQ files, LFQC (Nico1 et al. 2015) was benchmarked to be the most efficient (Numanagic et al. 2016), partly because LFQC uses several compression programs separately for sequence IDs, sequences, and quality scores. I compared file size reduction from FASTQ+ format against compression results from GZIP and LFQC (Table 4). Because FASTQ+ files are simple text files that can be further compressed by GZIP and LFQC, Table 4 also include compression output of GZIP+FASTQ+ and LFQC+FASTQ+. The results (Table 4) confirms the previous finding (Numanagic et al. 2016) that LFQC is much more efficient than GZIP. They also show FASTQ+ to offer a substantial further reduction of file sizes. For SRR1536586, file size reduction efficiency is comparable between LFQC and FASTQ+ (Table 4). However, further compression with GZIP+FASTQ+ and LFQC+FASTQ+ both leads to much reduced file size than using GZIP or LFQC alone (Table 4), the same being true for the paired-end file SRR922270 (Table 4). Furthermore, FASTQ+ has one additional advantage in that it dramatically reduces computation time in downstream data analysis. Take SRR1536586 for example, FASTQ+ would reduce computation time for read-matching (which is the most time-consuming part of any transcriptomic data analysis) to a fraction of roughly 0.075 ($\approx 119,596,093/1,604,183,348$).

### Assigning sequence reads to paralogous genes

One of the most fundamental objectives of RNA-Seq analysis is to generate an index of gene expression (FPKM: matched fragment/reads per kilobases of transcript per million mapped reads) that can be directly compared among different genes and among different experiments with different total number of matched reads (Mortazavi et al. 2008). The main difficulty in quantifying gene expression arises with sequence reads matching multiple paralogous genes (Trapnell et al. 2013;
Rogozin et al. (2014). This problem, which has plagued microarray data analysis, is now plaguing RNA-Seq analysis. Most publications of commonly used RNA-Seq analysis methods (Langmead et al. 2009, 2010; Trapnell et al. 2009, 2012; Roberts et al. 2011, 2013; Langmead and Salzberg 2012; Dobin et al. 2013; Deng et al. 2014) often avoided mentioning read allocation to paralogous genes. The software tools associated with these publications share two simple options for handling matches to paralogous genes. The first is to use only uniquely matched reads, i.e., reads that match to multiple genes are simply ignored. The second is to assign such reads equally among matched genes. These options are obviously unsatisfactory. Here, I describe a new approach which should substantially improve the accuracy of HTS data analysis such as gene expression characterization.

Allocating sequence reads to paralogous genes in a two-member gene family

We need a few definitions to explain the allocation. Let $L_1$ and $L_2$ be the sequence length of the two paralogous genes. Let $N_{U1}$ and $N_{U2}$ be the number of reads that can be uniquely assigned to paralogous gene 1 or 2, respectively (i.e., the reads that match one gene better than the other). Now for those reads that match the two genes equally well, the proportion of the reads contributed by paralogous gene 1 may be simply estimated as

$$P_1 = \frac{N_{U1}}{N_{U1} + N_{U2}} \quad (1)$$

Now, for any read that matches the two paralogous genes equally well, we will assign $P_1$ to paralogous gene 1, and $(1 - P_1)$ to paralogous gene 2. In the extreme case when paralogous genes are all identical, then $N_{U1} = N_{U2} = 0$, and we will assign 1/2 of these equally matched read to genes 1 and 2. We should modify Equation (1) to make it more generally applicable as follows

$$P_1 = \frac{0.01 + N_{U1}}{0.02 + N_{U1} + N_{U2}} \quad (2)$$

where 0.01 in the numerator and 0.02 in the denominator are pseudocounts. The treatment in Equation (2) implies that, when $N_{U1} = N_{U2} = 0$ (e.g., when two paralogous genes are perfectly identical), then a read matching equally well to these paralogous genes will be equally divided among the two paralouges.

One problem with this treatment is its assumption of $L_1 = L_2$. If paralogous gene 1 is much longer than the other, then $N_{U1}$ is expected to be larger than $N_{U2}$, everything else being equal. One may standardize $N_{U1}$ and $N_{U2}$ to number of unique matches per 1000 nt, designated by $SN_{U1,2} = 1000N_{U1}/L_i$ (where $i = 1$ or 2) and replace $N_{U1}$ in Equation (2) by $SN_{U1}$ as follows (Mortazavi et al. 2008):

![Figure 2](image)
Allocating sequence reads in gene family with more than two members

One might, mistakenly, think that it is quite simple to extend Equation (3) for a gene family of two members to a gene family with F members by writing

\[
P_i = \frac{0.01 + SN_{U_i}}{0.02 + SN_{U1} + SN_{U2}} = \frac{0.01 + 1000N_{U_i}}{0.02 + 1000(N_{U_1} + N_{U_2})}
\]

This does not work. For example, if we have three paralogous genes designated A, B, and C, respectively. Suppose that the gene duplication that gave rise to B and C occurred very recently so that B and C are identical, but A and the ancestor of B and C have diverged for a long time. In this case, \(N_{UB} = N_{UC} = 0\) because a read matching B will always match C equally well, but \(N_{UA}\) may be >0. This will result in unfair allocation of many transcripts from B and C to A according to Equation (4). I outline the approach below for dealing with gene families with more than two members.

With three or more paralogous genes, one may benefit from a phylogenetic tree for proper allocation of sequence reads. I illustrate the simplest case with a gene family with three paralogous genes A, B, and C, idealized into three segments in Figure 3. The three genes share one identical middle segment with 23 matched reads (that necessarily match equally well to all three paralogues). Genes B and C share an identical first segment to which 20 reads matched. Gene A has its first segment different from that of B and C, and got four matched reads. The three genes also have a diverged third segment where A matched three reads, B matched six and C matched 12. Our task is then to allocate the 23 reads shared by all three, and 20 reads shared by B and C to the three paralogues.

One could apply maximum likelihood or least-squares method for the estimation, but ARSDA uses a simple counting approach by applying the following

\[
P_A = \frac{3 + 4}{3 + 4 + 20 + 6 + 12} = 0.15556
\]
\[
P_B = \frac{6}{6 + 12} = 0.28148
\]
\[
P_C = \frac{12}{6 + 12} = 0.56296
\]

Thus, we allocate the 23 reads (that matched three genes equally) to paralogous genes A, B, and C according to \(P_A, P_B, \text{ and } P_C\) respectively. For the 20 reads that matched B and C equally well, we allocate \(20 \times 6/(6 + 12) = 12\) to B and \(20 \times 12/(6 + 12) = 20\) to C. This gives the estimated number of matches to each gene as

\[
N_A = 3 + 4 + 23P_A = 10.57778
\]
\[
N_B = 6 + 23P_B + 20 \left(\frac{6}{6 + 12}\right) = 19.14074
\]
\[
N_C = 12 + 23P_C + 20 \left(\frac{12}{6 + 12}\right) = 38.28148
\]

These numbers are then normalized to give FPKM (Mortazavi et al. 2008). The current version of ARSDA assume that gene families with more than two members have roughly the same sequence lengths. This is generally fine with prokaryotes but may become problematic with eukaryotes.

In practice, one can obtain the same results without actually undertaking the extremely slow process of building trees for paralogous genes. One first goes through reads shared by two paralogous genes (e.g., the 20 reads shared by genes B and C in Figure 3) and allocate the reads according to \(P_B = 6/(6 + 12) = 1/3 \) and \(P_C = 12/(6 + 12) = 2/3\). Now genes B and C will have 12.66667 (= 6 + 20\(P_B\)) and 25.33333 \((= 12 + 20P_C)\) assigned reads, i.e., \(N_{UB} = 12.66667\) and \(N_{UC} = 25.33333\). Once we have done with reads shared by two paralogous genes, we go through reads shared by three paralogous genes, e.g., the 23 reads shared by genes A, B, and C in Figure 3. With \(N_{UA} = 7, N_{UB} = 12.66667, N_{UC} = 25.33333\), and \(N = N_{UA} + N_{UB} + N_{UC} = 45\), so we have

\[
P_A = \frac{N_{UA}}{N} = 0.15556; P_B = \frac{N_{UB}}{N} = 0.28148; P_C = \frac{N_{UC}}{N} = 0.56296
\]

\[
N_A = 7 + 23P_A = 10.57778
\]
\[
N_B = 12.66667 + 23P_B = 19.14074
\]
\[
N_C = 25.33333 + 23P_C = 38.28148,
\]

which are the same as shown in Equation (6). This progressive process continues until we have allocated reads shared by the largest number of paralogous genes. The gene expression output in File S1 was obtained in this way.

There are alternative approaches for read allocation among paralogous genes. ARSDA also has an alternative allocation scheme based on BitScores and e-values. For example, when a read exhibits strong homology to \(N\) paralogous genes, but with different e-values or BitScores, I will not assign the read to the paralogous gene with the smallest e-value (or largest BitScore). Instead, all \(N\) paralogous genes will get a share of the read according to sequence similarity reflected in e-value and BitScore. The simplest scheme based on BitScore is to allocate such a read to paralogous gene \(i\) according to

\[
P_i = \frac{\text{BitScore}_i}{\sum \text{BitScore}_i}
\]

which would give a paralogous gene with high BitScore a higher share. An alternative based on e-value is

\[
P_i = \frac{K_{\text{min}}(E)}{E_i}
\]
where \( E \) is e-value and \( K \) is a scaling factor to ensure that \( \sum p_i = 1 \). Equation (10) allocates shared reads more to the paralogous gene with a small e-value than those with large e-value. In practice, Equation (9) is often close to equal allocation, whereas Equation (10) results in more biased allocation favoring the best-match paralogous genes.

**CONTRAST BETWEEN ARSDA AND CUFFLINKS IN CHARACTERIZING GENE EXPRESSION**

The most frequently used software for gene expression is Cufflinks (Trapnell et al. 2012), which is why I am contrasting ARSDA against it. I will use the transcriptomic data for an E. coli wild type (Pobre and Arraiano 2015), archived in NCBI’s SRA database as SRR1536586.sra.
The Cufflinks-quantified gene expression for this file is in file GSM1465035_WT.txt.gz from NCBI Geo DataSets GSM1465035. Gene expression from ARSDA and Cufflinks in mostly concordant (Figure 4), but four points (labeled in Figure 4) stand out as outliers (although many more discordant points will be revealed by a log-log plot). Such dramatic differences demand an explanation. Take rpmJ for example. Either ARSDA severely overestimated, or Cufflinks severely underestimated, the gene expression (Figure 4). I originally expected the discrepancy to be due to different allocation of paralogous genes. The expectation is only partially true.

There are 6426 reads can be mapped unambiguously to rpmJ (which is in fact a single-copy ribosomal protein gene). Although there are rpmA, rpmB, …, rpmJ genes in E. coli, they are not paralogous. One particular read “AGTGCCGAGCAGAAGCATAACAGCTTCAAGGCTGATTTTTGCATATT” occurs 2684 times in SRR1536586.sra. It matches perfectly to the 36 nt at the 3’ end of rpmJ and 14 nt immediately downstream. However, Cufflinks output reported a count of only 2114 reads for rpmJ instead of 6426 (and consequently the much reduced RPKM in Figure 4). I originally suspected that rpmJ may be in an operon with an immediate downstream gene so that some read overlapping rpmJ and the downstream gene would be divided between the two. However, the downstream gene, which is rpsM, is 146 nt away. It is difficult to reconcile 6426 nonambiguous read matches to Cufflinks’ 2114. Similarly, rpmD and rplV (Figure 4) has 14,468 and 22,747 unambiguous read matches, respectively, but the corresponding counts in Cufflinks output are only 8108 and 11,801, respectively. Note that rpmD and rplV are also single-copy genes with no ambiguous read matches. E. coli genes rpmA–rpmJ are not paralogous, neither are rplA–rplY.

The last outlying gene (cpsA in Figure 4) does involve a paralogous gene family (Figure 5). cpsA has 19,776 unambiguous read matches, but Cufflinks output has only 10,957, which resulted in a much lower RPKM than that from ARSDA (Figure 4). Also puzzling are the counts involving cpsF and cpsH. There are 264 unambiguous read matches to cpsD and 58 to cpsH. There are also 55 reads that match well to both cpsD and cpsH, with 27 of them matching cpsD better than cpsH, and 28 matching cpsH better than cpsD. So we may assign (264 + 27) reads to cpsD and (58 + 28) reads to cpsH, with relative proportions of 0.7719 and 0.2281 for cpsD and cpsH, respectively. Twelve reads match cpsF and cpsH equally well (the same e-value and the same BitScore), so we assign them proportionally to the two genes, i.e., 12/0.7719 to cpsF and 12/0.2281 to cpsH. The final counts for cpsF and cpsH are 300.2626 and 88.7374, respectively. However, Cufflinks output shows counts of 2 and 63 for cpsF and cpsH, respectively. The discrepancy is particularly striking given that gene expression from ARSDA and Cufflinks are mostly concordant (Figure 4). The alternative allocation to paralogous

![Figure 4](image-url) Contrast in gene expression (RPKM) between ARSDA and Cufflinks output for the same transcriptomic data in file SRR1536586.sra for E. coli wild type.

![Figure 5](image-url) Phylogenetic relationship among paralogous genes cpsA to cpsI in E. coli, based on coding sequences, with bootstrap values next to internal nodes. Sequences were aligned by MAFFT (Katoh and Toh 2008) with accurate L_INS-i option and a maximum of 16 iterations. Coding sequences were first translated in amino acid sequences, which are aligned with BLOSUM62 matrix. Nucleotide sequences were then aligned against aligned amino acid sequences. Phylogenetic analysis was done with PhyML (Guindon et al. 2010). All these analyses were automated in DAMBE (Xia 2013, 2017).
ACKNOWLEDGMENTS

Nabil Benabou of Center for Advanced Computing of Queen’s University helped me with testing compression software tools on the center’s servers. ARSDA was demonstrated in two workshops, one organized by J. Lu of Peking University, and another by E. Pranckeviciene of University of Vilnius. I thank participants, as well as J. Silke, J. Wang, Y. Wei, and C. Vlasschaert, for their feedback. Two anonymous reviewers offered outstanding feedback leading to substantial improvement of the manuscript. This study is funded by the Discovery Grant from Natural Science and Engineering Research Council (RGPIN/261252) of Canada.

LITERATURE CITED

Abraham, J. M., J. E. Feagin, and K. Stuart, 1988 Characterization of cytomegoc oxi dase III transcripts that are edited only in the 3’ region. Cell 55: 267–272.

Alatortsev, V. S., J. Cruz-Reyes, A. G. Zhelonkina, and B. Sollner-Webb, 2008 Trypanosoma brucei RNA editing: coupled cycles of U deletion reveal proactive activity of the editing complex. Mol. Cell. Biol. 28: 2437–2445.

Andrews, S., 2017 FastQC, Babraham Bioinformatics. Available at: https://www.bioinformatics.babraham.ac.uk/projects/fastqc/.

Arava, Y., Y. Wang, J. D. Storey, C. L. Liu, P. O. Brown et al., 2003 Genome-wide analysis of mRNA translation profiles in Saccharomyces cerevisiae. Proc. Natl. Acad. Sci. USA 100: 3889–3894.

Awan, A. R., A. Manfredo, and J. A. Pleiss, 2013 Lariat sequencing in a unicellular yeast identifies regulated alternative splicing of exons that are evolutionarily conserved with humans. Proc. Natl. Acad. Sci. USA 110: 12762–12767.

Benoit, G., C. Lemaître, D. Lavenier, E. Drezen, T. Dayris et al., 2015 Reference-free compression of high throughput sequencing data with a probabilistic de Bruijn graph. BMC Bioinformatics 16: 288.

Dai, M., R. C. Thompson, C. Maher, R. Contreras-Galindo, M. H. Kaplan et al., 2010 NGSSQC: cross-platform quality analysis pipeline for deep sequencing data. BMC Genomics 11(Suppl. 4): S7.

Deng, Q., D. Ramkold, B. Reinius, and R. Sandberg, 2014 Single-cell RNA-seq reveals dynamic, random monoallelic gene expression in mammalian cells. Science 343: 193–196.

Dobin, A., C. A. Davis, F. Schlesinger, J. Drenkow, C. Zaleski et al., 2013 STAR ultradist universal RNA-seq aligner. Bioinformatics 29: 15–21.

GB Editorial Team, 2011 Closure of the NCBI SRA and implications for the long-term future of genomics data storage. Genome Biol. 12: 402.
Rogozin, I. B., D. Managadze, S. A. Shabalina, and E. V. Koonin, 2014  Gene family level comparative analysis of gene expression in mammals validates the ortholog conjecture. Genome Biol. Evol. 6: 754–762.

Simpson, R. M., A. E. Bruno, J. E. Bard, M. J. Buck, and L. K. Read, 2016  High-throughput sequencing of partially edited trypanosome mRNAs reveals barriers to editing progression and evidence for alternative editing. RNA 22: 677–695.

Stepankiw, N., M. Raghavan, E. A. Fogarty, A. Grimson, and J. A. Pleiss, 2015  Widespread alternative and aberrant splicing revealed by lariat sequencing. Nucleic Acids Res. 43: 8488–8501.

Trapnell, C., L. Pachter, and S. L. Salzberg, 2009  TopHat: discovering splice junctions with RNA-Seq. Bioinformatics 25: 1105–1111.

Trapnell, C., A. Roberts, L. Goff, G. Pertea, D. Kim et al., 2012  Differential gene and transcript expression analysis of RNA-seq experiments with TopHat and Cufflinks. Nat. Protoc. 7: 562–578.

Trapnell, C., D. G. Hendrickson, M. Sauvageau, L. Goff, J. L. Rinn et al., 2013  Differential analysis of gene regulation at transcript resolution with RNA-seq. Nat. Biotechnol. 31: 46–53.

Vlasschaert, C., X. Xia, and D. A. Gray, 2016  Selection preserves Ubiquitin Specific Protease 4 alternative exon skipping in therian mammals. Sci. Rep. 6: 20039.

Xia, X., 2013  DAMBE5: a comprehensive software package for data analysis in molecular biology and evolution. Mol. Biol. Evol. 30: 1720–1728.

Xia, X., 2017  DAMBE6: new tools for microbial genomics, phylogenetics and molecular evolution. J. Hered. 108: 431–437.

Xia, X., V. MacKay, X. Yao, J. Wu, F. Miura et al., 2011  Translation initiation: a regulatory role for poly(A) tracts in front of the AUG codon in Saccharomyces cerevisiae. Genetics 189: 469–478.

Zhu, Z., L. Li, Y. Zhang, Y. Yang, and X. Yang, 2015a  CompMap: a reference-based compression program to speed up read mapping to related reference sequences. Bioinformatics 31: 426–428.

Zhu, Z., Y. Zhang, Z. Ji, S. He, and X. Yang, 2015b  High-throughput DNA sequence data compression. Brief. Bioinform. 16: 1–15.

Communicating editor: C. Myers