XPRESSyourself: Enhancing the High-Throughput Sequencing Toolkit and Automating Ribosome Profiling and RNA-Seq Analysis

Jordan A. Berg,[1*] Jonathan R. Belyeu,[2] Jeffrey T. Morgan,[1] Alex J. Bott,[1]
Yeyun Ouyang,[1] Aaron R. Quinlan,[2,4,5] Jason Gertz,[3] Jared Rutter[1,6*]

[1]Department of Biochemistry, University of Utah, Salt Lake City, UT, USA, 84112
[2]Department of Human Genetics, University of Utah, Salt Lake City, UT, USA, 84112
[3]Department of Oncological Sciences, University of Utah, Salt Lake City, UT, USA, 84112
[4]USTAR Center for Genetic Discovery, University of Utah, Salt Lake City, UT, USA, 84112
[5]Department of Biomedical Informatics, University of Utah, Salt Lake City, UT, USA, 84112
[6]Howard Hughes Medical Institute, University of Utah, Salt Lake City, UT, USA, 84112

[*]Address correspondence to: jordan.berg@biochem.utah.edu, rutter@biochem.utah.edu.

Nucleic acid sequencing is a routine and powerful research tool; however, computational bottlenecks exist for many users. XPRESSyourself is a ribosome profiling and RNA-Seq analytical pipeline that aims to eliminate these barriers, standardize *in silico* protocols, and decrease time-to-discovery. XPRESSyourself additionally introduces tools missing from current ribosome profiling and RNA-Seq toolkits. Using XPRESSyourself to process publicly available ribosome profiling data, we were able to identify in a matter of hours putative mechanisms that could explain neurodegenerative phenotypes and neuroprotective mechanisms of the small-molecule ISRIB during acute cellular stress, highlighting XPRESSyourself's ability to rapidly uncover novel biological insight from published data.

Keywords

Analysis Pipeline, Ribosome Profiling, RNA-Seq, Automation, Standardization, GTF Truncation, Intron-agnostic Gene Coverage, Quality Control, High-Throughput Sequencing

1 Background

High-throughput sequencing data has revolutionized biomedical, industrial, and basic science research. Specifically, RNA-Seq has become the forerunner technology for high-quality RNA quantification within the last decade. RNA-Seq involves isolating the RNA fragments from a population of cells, converting these fragments into cDNA libraries, and aligning the sequenced reads to a reference genome or transcriptome to measure relative transcript abundance, differential splice variants, sequence polymorphisms, and more [1]. High-throughput sequencing technologies have been developed or adapted for a variety of technologies such as DNA sequencing, ChIP-seq, single-cell RNA-Seq, and ribosome profiling [2].

Although vast strides have been made to implement and perfect these technologies, many bottlenecks still exist. For example, while a basic bioinformatic understanding is more commonplace, learning the intricacies of processing RNA-Seq data can be challenging and problematic. Moreover, many users are not aware of the most up-to-date tools or the appropriate settings for their application [3, 4]. Even for the experienced user, developing robust automated pipelines that accurately process and assess the quality of these datasets can be laborious. The variability that inevitably arises with each lab or core facility designing and using personal pipelines is also a significant challenge in the field.

Though RNA-Seq is a matured technology, there is still an abundance of biases and peculiarities associated with each analytical method or tool, which are often obscured to a user. Additionally, few if any pre-existing pipelines or toolkits offer a thorough set of integrated tools for assessing standard quality control metrics or performing reference curation, particularly in ribosome profiling, which in many aspects is still maturing [5]. For example, a common bias in ribosome profiling libraries is a $5'$ and $3'$ read pile-up [6–8] due to longer kinetics associated with the initiation and termination steps of translation compared to the elongation step. Experts generally recommend that these pile-up-prone regions not be quantified when processing ribosome profiling libraries; however [5, 9], no publicly available computational tools currently exist to facilitate making the appropriate truncations to reference transcripts.

Several computational pipelines for RNA sequencing have emerged that intend to tackle various aspects of these bottlenecks, but many suffer from usability issues, are not easily modifiable, or sacrifice quality for speed. For example, a simple internet search for RNA-Seq pipelines will reveal several classes of pipelines. The first class is a tutorial labeled as a pipeline. Many instances of these are available [10, 11]; however, they are not automated, are often outdated, and can be difficult to implement. The second class is a semi-automated pipeline that requires extensive manual configuration [12–15]. The third class is an automated pipeline but requires programmatic modification to change many common parameters [16, 17]. Perhaps, the most user-friendly example is Galaxy [18], but in cases like its ribosome profiling pipeline, methods are severely outdated and a robust quality control step is lacking. In all the above cases, a thorough, robust, simple pipeline geared to the general user without sacrificing speed or quality is lacking.

In response to these issues surrounding the automation of sequencing technology, we built the XPRESSyourself bioinformatics suite for processing and analyzing high-throughput expression data. This suite was architecturally designed from the ground up to be computationally efficient, without sacrificing quality for speed. Each step of the pipeline utilizes the best performing software package for that task, having been previously vetted by peer-reviewed benchmarking studies where such studies exist for a given tool. Additionally, the pipeline is designed such that updating and testing of a new module are facile tasks for a trained bioinformatician. This enables XPRESSyourself packages to continuously offer the best options available to the entire community, regardless of expertise.

Currently, XPRESSyourself is partitioned into two main software packages. With the XPRESSpipe package, the user is provided with a complete suite of software to handle pre-processing, aligning, and quantifying of sequencing reads, performing quality control via various meta-analyses of pre- and post-processed reads. We also provide access to key quality control measures useful for assessing ribosome profiling and other RNA-Seq experiments. These include read length distributions plots that are particularly helpful for ribosome profiling experiments due to the unique characteristics of the ribosome footprint-sized libraries (usually around 17-33 nt) [19], and a periodicity sub-module that tracks the P-site of ribosome footprints to assess effective capture of the characteristic one codon step of the ribosome. Ribosome profiling also faces the challenge of efficiently depleting ribosomal RNA (rRNA) from samples, as commercial depletion kits are often not selective enough in this experimental context. XPRESSpipe provides a feature that identifies the most abundant rRNA species to target for depletion during library preparation. XPRESSpipe also includes a metagene analysis sub-module that shows the distribution of the relative position of all aligned reads across a representative transcript to show if $5'$ or $3'$ biases in RNA fragment capture occurred during library preparation. Currently, few if any up-to-date computational tools for handling this analysis exist. Additionally, XPRESSpipe includes a module for plotting gene coverage, similar to IGV [36], but where introns are collapsed in order to more clearly see read coverage across exons or coding space. As PCR duplicates can arise during library preparation, a library complexity visualization sub-module is included in the pipeline to assess the level of PCR artifacts in the library and ensure that a robust population of genes was captured during sequencing library creation. While XPRESSpipe summarizes hundreds to thousands of lines of code to one or two lines, we also provide the user with a guided command builder module.

The second package currently available within XPRESSyourself is XPRESSplot, which provides tools to perform the bulk of sequence analysis and generation of figures for publication, where many plot generation protocols that frequently require several hundred lines of code are condensed to a single line with minimal input from the user. XPRESSyourself suite packages are coded in Python and R, the current linguae francae of computational biology and bioinformatics, which allows for easy modification and improvement by the sequencing community at large. XPRESSyourself suite packages are perpetually open source under a GPL-3.0 license at https://github.com/XPRESSyourself.

2 Results

2.1 XPRESSpipe

XPRESSpipe contains automated pipelines for both single-end and paired-end RNA-Seq. The pipeline was designed based upon may characteristics of The Cancer Genome Atlas (TCGA) (https://www.cancer.gov/tcga) alignment standards with appropriate modifications or updates depending on the use. The pipeline handles pre-processing, alignment, and quantification of sequencing reads, after which it will perform essential quality control of each sequence library. In the case of ribosome profiling libraries, default parameters are optimized for this type of data. Within this manuscript, we will focus on ribosome profiling examples to demonstrate the utility of XPRESSpipe, while the majority of statements are also applicable to general single- or paired-end RNA-Seq. More details can be found in the documentation that will be updated as features are added, updated, or modified (https://xpresspipe.readthedocs.io/en/latest/). Table 1 outlines the parameters a user would need to consider modifying based on their sequencing setup or desired output. In the majority of cases, the default parameters for each module will be sufficient.

2.1.1 Inputs

While inputs will vary sub-module to sub-module, a few points of guidance are important to consider. Further information can be found in the documentation (https://xpresspipe.readthedocs.io/en/latest/) or by entering `xpresspipe <sub-module name> --help`. For example, single-end reads should be saved as a FASTQ-formatted file that ends in `.fq`, `.fastq`, or `.txt`. Paired-end reads should additionally include the appropriate mate-pair suffix before the file suffix, such `.read1.fastq` or `.read2.fastq`. Read files can be `.zip` - or `.gz` - compressed. Decompression will be handled automatically by XPRESSpipe. Required input reference files from the user are limited. XPRESSpipe only requires a valid GTF file appropriate for the organism of interest saved as `transcripts.gtf` and the appropriate genomic FASTA file(s). We recommend the genomic FASTA file(s) be placed

Table 1: **Summary of XPRESSpipe pipeline user parameters.**

Arguments	**Description**
Required	
`-i, --input`	Path to input directory
`-o, --output`	Path to output directory
`-r, --reference`	Path to parent organism reference directory
`-g, --gtf`	Path and file name to GTF used for alignment quantification
`-e, --experiment`	Experiment name
Optional	
`--two_pass`	Include option to perform a two-step alignment to map for unannotated splice-junctions
`-a, --adaptors`	Specify adaptor as string – if “None” is provided, software will attempt to auto-detect adaptors – if “POLYX” is provided as a single string in the list, polyX adaptors will be trimmed
`-q, --quality`	PHRED read quality threshold (default: 28)
`--min_length`	Minimum read length threshold to keep for reads (default: 18)
`--umi_location`	Provide parameter to process UMIs – provide the location (see fastp documentation for more details)
`--umi_length`	Provide parameter to process UMIs – provide UMI length (must provide the –umi_location argument)
`--deduplicate`	Include option to quantify alignment files with de-duplication
`--output_bed`	Include option to output BED files for each aligned file
`-c, --quantification_method`	Specify quantification method (default: HTSeq [20])
`--feature_type`	Specify feature type (3rd column in GFF file) to be used if quantifying with HTSeq (default: CDS)
`--stranded`	Specify stranded library preparation method (Varies based on quantification method, see documentation for more information)
`--method`	Provide parameter and method to perform library normalization on samples (options: “RPM”, “TPM”, “RPKM”, “FPKM”)
`--vcf`	Provide full path and file name to VCF file if you would like to detect personal variants overlapping alignments, otherwise not considered
`--batch`	Include path and filename of dataframe with batch normalization parameters
`--sjdbOverhang`	Sequencing read-length - 1 parameter used during reference curation (see STAR documentation for more information)
`--mismatchRatio`	Alignment ratio of mismatches to mapped length is less than this value (see STAR documentation for more information)
`--seedSearchStartLmax`	Adjust this parameter by providing a lower number to improve mapping sensitivity (recommended value = 15 for reads 25 nts) (see STAR documentation for more information)
`--genome_size`	Change if parameter is provided during reference building and using a two-pass alignment
`-m, --max_processors`	Specify number of max processors to use for tasks (default: No limit)

in their own sub-directory within the parent reference directory and that the most up-to-date Ensembl curation be used (https://www.ensembl.org).

2.1.2 User Aids

Several tools and resources are provided to aid in making XPRESSyourself accessible to all users. One such tool is an integrated command builder, accessed by running `xpresspipe build`. This command builder will walk the user through potential considerations based on their library preparation method and build the appropriate command for execution on their personal computer or a supercomputing cluster. If running the command builder on a personal machine, the user will be given a choice to have XPRESSpipe execute the command for them. In addition to this resource, the XPRESSyourself suite provides thorough documentation for each module and tool, along with video walkthroughs (accessible through the README files) and interactive notebooks (found in the home directory of a package.

2.1.3 Automated Reference Curation

One of the first steps of RNA-Seq alignment is curating a reference to which the alignment software will map reads. For the current version of XPRESSpipe, a STAR [21] reference is automatically curated simply by providing the appropriate GTF file saved as `transcripts.gtf` and directory path to the genomic FASTA file(s). Currently, STAR is used within the pipeline as it has been shown consistently to be the best performing read aligner for RNA-Seq [22]. Additional modifications are occasionally recommended to this file, which can be performed within this sub-module or separately, discussed in more detail in the next section. As this can be a time-consuming process, we will leave the `--max_processors` parameter as default in this example to utilize all cores available to the computing unit. This entire process is automatically handled with the `curateReference` sub-module for ease of use to the user. More on GTF modification arguments used in this code block follows in the section below.

Listing 1: curateReference example

```
$ xpresspipe curateReference -o /path/to/reference/ \
                             -f /path/to/reference/fasta_genome/ \
                             -g /path/to/reference/transcripts.gtf \
                             --protein_coding \
                             --longest_transcript \
                             --truncate \
                             --truncate_5prime 45 \
                             --truncate_3prime 15 \
                             --sjdbOverhang 49 \
                             --max_processors None
```

2.1.4 GTF Modification

As ribosomal RNAs and other non-coding RNAs can be highly abundant in RNA-Seq experiments, it is often recommended to not include these sequences for quantification. By providing the `--protein_coding` argument, only protein-coding genes are retained in the GTF file, which acts as a masking step of reads aligning to non-coding regions of the genome.

In most eukaryotes, mRNAs undergo alternative splicing. However, some tools may consider the multiple annotated splice variants of a gene as a multi-mapper since they map to a location where several isoforms of the same gene overlap. These reads are either penalized or discarded. By providing the `--longest_transcript` argument, the longest Ensembl canonical transcript [23] is retained for each gene in the GTF file. However, if using HTSeq with default XPRESSpipe parameters or Cufflinks to quantify reads, this is not necessary. Particularly in the case of Cufflinks, the software is optimized to quantify abundances of the different isoforms of each gene [24].

For ribosome profiling, where read pile-ups at the $5'$ and $3'$ ends of an open reading frame are largely uninformative as to the translational efficiency of the gene, the $5'$ and $3'$ ends of each transcript's total coding region should be truncated as to not be used during read quantification [5,9]. By providing the `--truncate` argument, the $5'$ and $3'$ ends of each coding region will be trimmed by the specified amounts. These values are set to defaults of 45 nt for $5'$ truncation and 15 nt for $3'$ truncation, as is the convention within the ribosome profiling field [5], but these can be modified using the `--truncate_5prime` or `--truncate_3prime` parameters. If generating a GTF for use with general RNA-Seq datasets, the `--truncate` argument should not be provided.

2.1.5 Read Processing

Although all intermediate steps of the pipelines can be run singly, we will describe the outline of the software in the context of the ribosome profiling pipeline. Pipelines and individual sub-modules are capable of being run in a parallel manner for each input file, thus accelerating the overall process. Descriptions of the options can be found in Table 1.

2.1.6 Trimming

Reads generally need to be cleaned of artifacts from library creation. These include adaptors, unique molecular identifier (UMI) sequences, and technical errors in the form of low-quality base calls. By doing so, non-native sequences are removed and reads can align properly to the reference. XPRESSpipe uses fastp, a faster, more accurate trimming package that has improved alignable read output compared to its predecessors [25]. Adaptor sequence, base quality, and read length are all adjustable parameters available to the user. Additionally, features, such as UMIs can be specified and grouped in pre-processing, then removed in post-alignment processing to remove PCR artifacts [26, 27].

2.1.7 Alignment

Reads are aligned to a reference genome. XPRESSpipe uses STAR, which, despite being more memory-intensive, is relatively fast and one of the most accurate sequence alignment options currently available [21, 28]. XPRESSpipe is capable of performing a single-pass, splice-aware GTF-guided alignment or a two-pass alignment of reads wherein novel splice junctions are determined and built into the reference, followed by alignment of reads to the new reference. A coordinate-sorted and indexed BAM file is output by STAR. We abstain from rRNA negative alignment at this step as downstream analysis of these mapped reads could be of interest to some users.

2.1.8 Post-alignment Processing

XPRESSpipe further processes alignment files by optionally parsing files for unique alignments that are then passed on to the next steps. PCR duplicates are detected and marked or removed for downstream processing; however, these files are only used for relevant downstream steps (such as library complexity quality control) or if the user specifies to use these de-duplicated files in downstream steps such as read quantification. Use of de-duplicated alignment files may be advisable in situations where the library complexity profiles (discussed below) exhibit high duplication levels. However, generally the abundance of PCR-duplicates is low in properly-prepared sequencing libraries; thus, doing so may be overly stringent [26]. These steps are performed using samtools [29]. Optionally, BED coverage files can also be output. These conversions are handled by bedtools [30].

2.1.9 Read Quantification

XPRESSpipe quantifies read alignments for each input file using HTSeq with the `intersection-nonempty` method by default [20, 31]. Our rationale for including this quantification method is that it conforms to the current default TCGA standards and is favorable in most applications. If masking of non-coding RNAs is desired, a `protein_coding` modified GTF file should be provided for the `--gtf` argument. HTSeq is recommended for processing ribosome profiling data as it allows selection of feature type across which to quantify, thus allowing for quantification across the CDSs of a transcript instead of the exons. If the user is interested in isoform abundance estimation, Cufflinks is available to perform this method of quantification instead [24, 31].

2.1.10 Normalization

Methods for count normalization are available within XPRESSpipe by way of the XPRESSplot package (described below). For normalizations correcting for transcript length, the appropriate GTF must be provided. Current sample normalization methods available include reads-per-million (RPM), Reads-per-kilobase-million (RPKM) or Fragments-per-kilobase-million (FPKM), and transcripts per million (TPM) normalization [32]. For samples sequenced on different flow cells, prepared by different individuals, or on different days, the `--batch` argument should be provided along with the appropriate metadata matrix, which is then processed by way of XPRESSplot using the ComBat package [33].

2.1.11 Quality Control

A vital step in RNA-Seq analysis is proper quality control of sequencing samples to ensure the interpreted downstream results are reliable. XPRESSpipe performs a variety of quality control measures. For each analysis type, high-resolution, publication-quality summary figures are output for all samples in a given experiment for quick reference to the user.

2.1.12 Read Length Distribution

The lengths of all reads are analyzed by FastQC [34] after trimming. By assessing the read distribution of each sample, the user can ensure the expected read size was sequenced. This is particularly helpful in ribosome profiling experiments for verifying the requisite 17-33 nt ribosome footprints were selectively captured during library preparation [5, 19]. Metrics are compiled and plotted by XPRESSpipe.

2.1.13 Library Complexity

Measuring library complexity is an effective method for analyzing the robustness of a sequencing experiment in capturing various, unique RNA species. As the majority of RNA-Seq preparation methods involve a PCR step, sometimes particular fragments will be favored and over-amplified in contrast to others. By plotting the number of PCR replicates versus expression level for each gene, one can monitor any effects of limited transcript capture diversity and/or high estimated PCR duplication rate on the robustness of their libraries. This analysis is performed using dupRadar [35] where inputs are PCR duplicate-tagged BAM files output by XPRESSpipe by way of samtools [29]. Metrics are then compiled and plotted by XPRESSpipe.

2.1.14 Metagene Estimation Profile

In order to identify any general biases for the preferential capture of the $5'$ or $3'$ ends of transcripts, metagene profiles can be generated for each sample. This is performed by determining the meta-genomic coordinate for each aligned read in exon space. Required inputs are an indexed BAM file and an un-modified GTF reference file. Outputs include metagene metrics, individual plots, and summary plots.

2.1.15 Gene Coverage Profile

Extending the metagene estimation analysis, the user can focus on the coverage profile across a single gene. Although traditional tools like IGV [36] offer the ability to perform tasks such as this, XPRESSpipe provides the ability to collapse the introns to observe coverage over exon space only. This is helpful in situations where massive introns spread out exons and make it difficult to visualize exon coverage for the entire transcript in a concise manner. When running an XPRESSpipe pipeline, a housekeeping gene will be processed and output for the user's reference. Figure S1 provides a comparison with the output of IGV [36] and XPRESSpipe's geneCoverage module over a similar region to demonstrate the compatibility between the methods.

2.1.16 Codon Phasing/Periodicity Estimation Profile

In ribosome profiling, a useful measure of a successful experiment comes by investigating the codon phasing of ribosome footprints [5]. To do so, the P-site positions for each footprint relative the start position of the gene each read was aligned to are calculated using riboWaltz [37]. The same inputs are required as in the `metagene` sub-module.

2.1.17 rRNA Depletion Probe

Ribosomal RNA (rRNA) contamination is common in RNA-Seq library preparation as the bulk of RNA in a cell at any given time is dedicated to rRNA. The sequencing of these RNAs becomes highly repetitive, wasteful, and often biologically uninteresting in the context of gene expression and translation efficiency. The depletion of these sequences is therefore desired to increase the depth of coverage of ribosome footprints. In order to facilitate this depletion, many commercial kits are available that target specific rRNA sequences for depletion or that enrich for poly(A)-tailed mRNAs. However, and especially in the case of ribosome profiling experiments, where RNA is digested by an RNase to create ribosome footprints, many commercial depletion kits will not deplete digested rRNA sufficiently after the footprinting step of ribosome profiling library creation and poly(A)-selection kits are inappropriate as footprints will not have the requisite poly(A) tail. To this end, custom rRNA-depletion probes are recommended [2, 5]. `rrnaProbe` analyzes the over-represented sequences within a collection of footprint libraries that have already undergone adaptor and quality trimming, compiles conserved k-mers across the overall experiment, and outputs a rank-ordered list of these sequences for probe design.

2.1.18 Differential Expression Analysis

XPRESSpipe incorporates DESeq2 for performing differential expression analysis of count data. We refer users to the original publication for more information about uses and methodology [38]. In this module, the user provides the count table output by XPRESSpipe, along with a sample summary table and design formula (as explained in the DESeq2 documentation).

2.1.19 Outputs

While outputs will vary sub-module to sub-module, generally, the user will specify a parent output directory and necessary sub-directories will be created based on the step in the pipeline. Further information can be found in the documentation (https://xpresspipe.readthedocs.io/en/latest/) or by entering `xpresspipe <sub-module name> --help` in the command line. Figure 1 provides an example of the output file scheme for XPRESSpipe. For a complete pipeline run, the user can expect BAM alignment files, a collated count table of all samples in the experiment, and quality control figures and metrics. For almost all sub-modules, a log file will also be written to summarize provided user parameters, track performance, and report errors. An additional log file will be written summarizing dependencies' versions used during the pipeline run. Users are encouraged to provide these files as supplements within publications using XPRESSpipe-processed data to aid in documentation.

2.2 XPRESSplot

Further analysis of ribosome profiling or RNA-Seq data is handled within XPRESSplot. XPRESSplot is a Python library of analysis and plotting tools that build upon existing packages, such as Matplotlib [39] and Seaborn [40] to generate flexible, specific analyses and creates plots frequently used by biological researchers that can each be executed in a single line of code rather than tens to hundreds. Additionally, many included features are currently available in an R or other programming language package but not in a Python package. Brief summaries of key components of this package, as well as descriptions of new or more automated tools are provided below and methods are discussed in subsequent sections. We refer the reader to the documentation

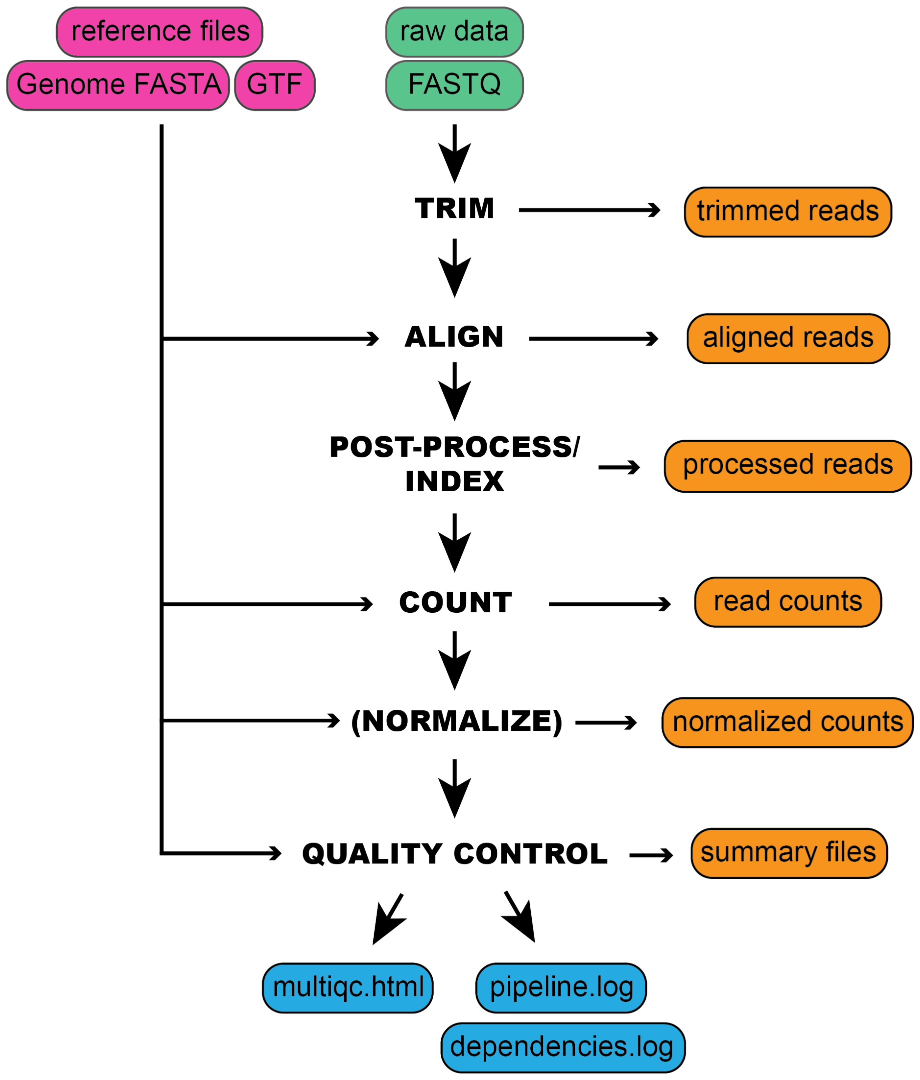


Figure 1: **An example schematic of the inputs required by XPRESSpipe and organization of the outputs.** Representation of the general steps performed by XPRESSpipe and data and log outputs. Steps in parentheses are optional to the user.

(https://xpressplot.readthedocs.io/en/latest) for more detailed instructions for other features within the toolkit. Although XPRESSplot is designed for handling transcriptomic datasets, it is also capable in many cases of handling other -omics datasets, such as microarray, proteomics, or metabolomics data.

2.2.1 Input Data

Generally, two inputs are required for all functions within XPRESSplot:

1. **Expression Matrix**: It is assumed the input data matrix = *i* * *j* where *i* (rows) are genes or other analytes and *j* (columns) are samples.
2. **Metadata Table**: It is assumed the metadata table is a two column, header-less data matrix where column 0 is the sample ID (as specified in *j* of the expression matrix) and column 1 is the sample group (for example, wild-type or treatment).

2.2.2 Normalization

RNA-Seq experiments can be normalized by the user using the reads-per-million (RPM), Reads-per-kilobase-million (RPKM) or Fragments-per-kilobase-million (FPKM), or transcripts per million (TPM) methods [32], as outlined in Equations 1-4 in the Methods. Other normalizations, such as mean centering of *i* features (i.e. genes or other analytes) by scikit-learn's preprocessing module [41] are also available. Count thresholds can also be set to remove lowly expressed genes from an analysis that may be less reliable due to poor ability to be sequenced.

2.2.3 Analyzing Data

Although a litany of analysis tools are included in XPRESSplot package as of the time of writing, we will focus on tools unique to this Python library or that are particularly useful and refer the reader to the documentation for further details and examples of additional analysis features.

2.2.4 Principal Components Analysis

Principal components analysis (PCA) for the data matrix is computed using Python's scikit-learn package [41] and desired principal components are plotted over a scatter plot via Matplotlib [39] and Seaborn [40]. The XPRESSplot PCA module, as in many other analysis modules within XPRESSplot, samples are color-coded for easy visualization of sample groups. Confidence intervals are plotted over the scatterplot using NumPy [42, 43], a feature currently missing from Pythonic PCA packages.

2.2.5 Volcano Plot

Volcano plots are an efficient method for plotting the magnitude, direction, and significance of changes in expression or other data types between two conditions with multiple replicates each. By providing the categorical names for samples of two conditions in the metadata matrix, XPRESSplot will automate the calculation and plotting of this method. When plotting gene expression values, the RNA-Seq-specific volcano plot method should be used which requires a DESeq2-output data table as input. This is essential as RNA-Seq datasets follow a negative-binomial distribution rather than a normal distribution [38]. For other uses, such as for proteomics or metabolomics datasets where the data is normally distributed, the general volcano plot method can be used, which will average the measurements for each analyte between the two conditions and calculate the $\log_2$(fold change). Additionally, for each gene, the p-value between the two conditions is calculated using SciPy's Individual T-test function [44]. The $\log_2$(fold change) and $-\log_{10}$(p-value) is then plotted for each gene between the two conditions. Additional features available are the ability to plot threshold lines, highlight subsets of genes within the plot, and label specific genes by name in order to quickly extract key data from the figure.

2.3 Validation

To evaluate the ability of XPRESSpipe to provide the user with reliable results, we processed publicly available raw sequence files using this automated pipeline. We chose to highlight one ribosome profiling dataset to showcase the utility of XPRESSpipe for rapidly extracting potentially interesting biological insights from sequence data. To further validate the performance of XPRESSpipe, we additionally chose a small subset of TCGA samples, processed their raw read data through XPRESSpipe, and compared the counts to the publicly available TCGA-processed count tables.

2.3.1 New Insights from Published Ribosome Profiling Data

The integrated stress response (ISR) is a signaling mechanism used by cells and organisms in response to a variety of cellular stresses [45]. Although acute ISR activation is essential for cells to properly respond to stresses, long periods of sustained ISR activity can be damaging. These prolonged episodes lead to a variety of diseases, including many that result in neurological decline [46]. A recently discovered small-molecule inhibitor of the ISR, ISRIB, has been demonstrated to have therapeutic potential and a relative lack of side-effects. Interestingly, ISRIB can suppress the damaging chronic low activation of the ISR, while it does not interfere with a cytoprotective acute, high-grade ISR. It has also been shown to be neuroprotective in mouse models of traumatic brain injury, adding to its wide pharmacological interest [47–53].

A recent study (data available under Gene Expression Omnibus accession number GSE65778) utilized ribosome profiling to better define the mechanisms of ISRIB action on the ISR, modeled by 1-hour tunicamycin (Tm) treatment in HEK293T cells [49]. A key finding of this study is that a specific subset of stress-related transcription factors mRNAs display increased translational efficiency (TE) compared to untreated cells during tunicamycin-induced ISR. However, when cells were co-treated with tunicamycin and ISRIB, the TE of these stress-related mRNAs showed no significant increase as compared to untreated cells, which indicates that ISRIB can counteract the translational responses associated with the ISR.

To showcase the utility of XPRESSpipe in re-analyzing ribosome profiling and sequencing datasets, we re-processed and analyzed this dataset using the more current *in silico* techniques included in the XPRESSpipe package to further query the translational mechanisms of the ISR and ISRIB. All XPRESSpipe-processed biological replicate samples exhibited strong correlation between read counts per gene when thresholded similarly to count data available with the original publication (Spearman ρ values 0.991-0.997) (Figure 2B; Figure S2B shows the corresponding plots using the count data provided with the original publication for reference).

Compared to the raw count data made available in the original manuscript, when XPRESSpipe-processed samples were thresholded as in the original published raw count data, samples showed generally comparable read counts per gene between the two analytical regimes (Spearman ρ values 0.937-0.950) (Figure 2A). This is in spite of the fact that the methods section of the original publication outlines the use of now outdated software, such as TopHat2 [54], which has a documented higher false positive alignment rate, generally lower recall, and lower precision at correctly aligning multi-mapping reads compared to STAR [21, 22]. Many of the genes over-represented in the original count data as compared to data processed by XPRESSpipe are genes that have pseudogenes or other paralogs (Figure S2A). As these genes share high sequence similarity with each other, reads mapping to these regions are difficult to attribute to a specific genomic locus and are often excluded from further analyses due to their multi-mapping nature. The benchmarking study [22] that examined these and other aligners described how TopHat2 had a disproportionally high rate of incorrectly aligned bases, or bases that were aligned uniquely when they should have been aligned ambiguously, at least partially explaining the observed overcounted effect with TopHat2. Had TopHat2 marked problematic reads as ambiguous, they would have been excluded from later quantification.

Additionally, when TopHat2 and STAR were tested using multi-mapper simulated test data of varying complexity, TopHat2 consistently suffers in precision and recall. These calls are increasingly more difficult to make with smaller reads as well, and this is evident from Figure 2A, where ribosome footprint samples consistently showed more over-counted genes than the corresponding RNA-Seq samples. When dealing with a ribosome footprint library of about 50-100 million reads, and with TopHat2's simulated likelihood of not marking an ambiguous read as such being about 0.5% higher than STAR, this would lead to 250,000 to 500,000 spuriously aligned reads, which is in line with our observations (all benchmarking details were derived from [22]).

An additional potential contributor to this divergence is that the alignment and quantification within XPRESSpipe uses a current human transcriptome reference, which no doubt contains modifications to annotated canonical transcripts and so forth when compared to the version available when the original study was published. However, in practice, these effects are modest for this dataset (Figure S3). While differences in processing between the outdated and current methods may not always create broad differences in output, key biology may be missed. The analysis that follows is exploratory and only meant to suggest putative targets identifiable by re-analyzing pre-existing, publicly available data.

Similar canonical targets of translation regulation during ISR were identified in the XPRESSpipe-processed data compared to the original study. These targets include ATF4, ATF5, PPP1R15A, and DDIT3 (Figure 3A-C, highlighted in purple) [49]. Of note, the fold-change in ribosome occupancy of ATF4 (6.83) from XPRESSpipe-processed samples closely mirrored the estimate from the original publication (6.44). Other targets highlighted in the original study [49], such as ATF5, PPP1R15A, and DDIT3 also demonstrated comparable increases in their ribosome occupancy fold-changes to the original publication count data (XPRESSpipe: 5.89, 2.47, and 3.93; respectively. Original: 7.50, 2.70, and 3.89; respectively) (Figure 3A). Similar to the originally processed data, all of these notable changes in ribosome occupancy return to untreated levels during Tm + ISRIB co-treatment (Figure 3B). Additional ISR uORF targets highlighted in the study (highlighted in green in Figure 3A-C) also mirrored changes reported in the original study in translational and transcriptional regulation across conditions.

Both the original study and our XPRESSpipe-based re-analysis show that ISRIB is able to counteract the significant increase in TE for a set of genes during ISR. To further explore TE regulation during ISR, we asked if ISRIB has a similar muting effect on genes with significant decreases in TE induced by the ISR. In the original study, genes with significant decreases in TE were reported in a source-data table but not a focus in the study. However, re-analysis of these data with the updated XPRESSpipe methodology identifies genes whose translational down-regulation may play a role in the neurodegenerative effects of ISR and the neuroprotective properties of

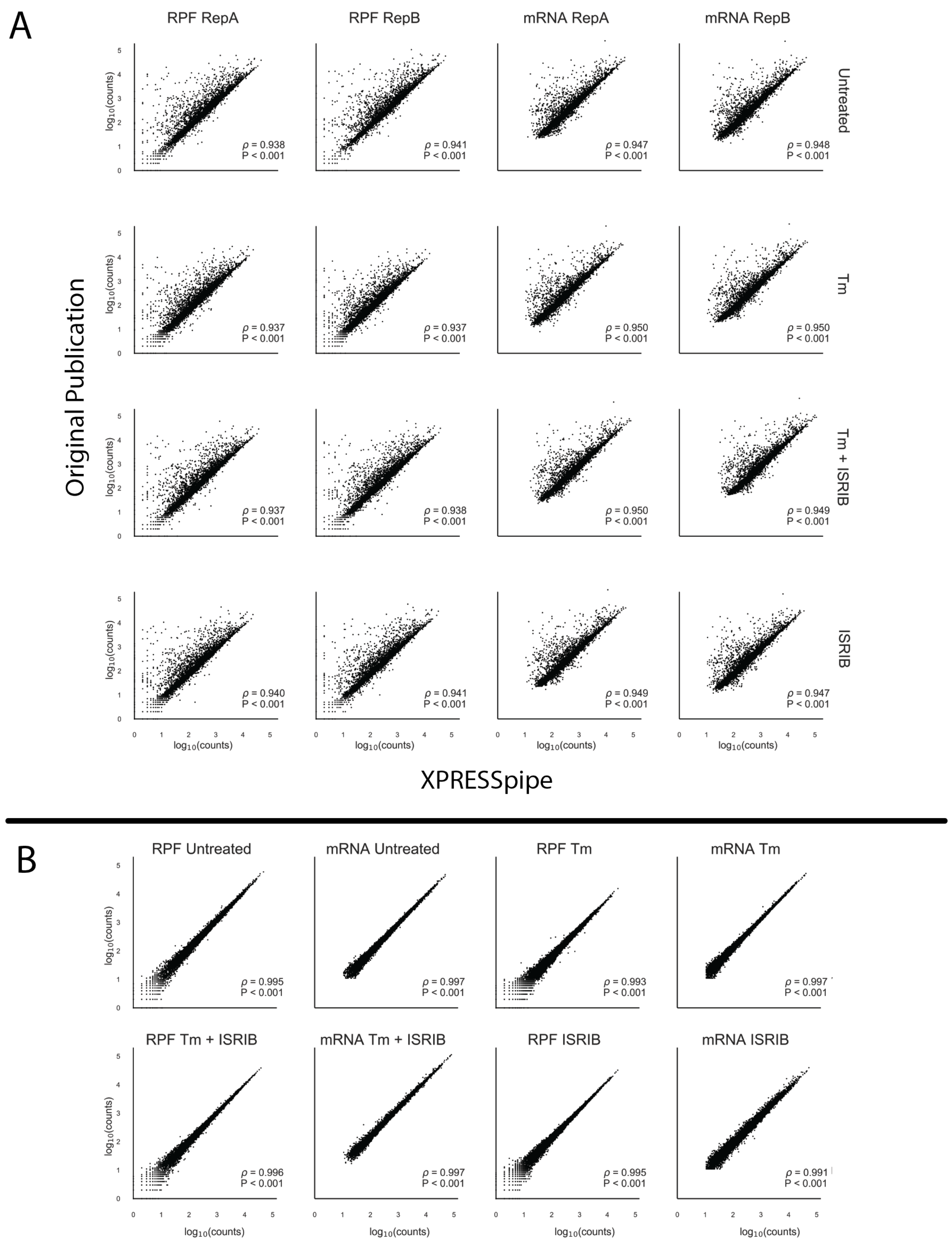


Figure 2: **Comparison between processed data produced by XPRESSpipe and original study.** Genes were thresholded by eliminating any where any RNA-Seq sample had fewer than 10 counts. A) Comparison of read counts per gene between count data from the original study and the same raw data processed and quantified by XPRESSpipe. B) Comparison of biological replicate read counts processed by XPRESSpipe. RPF, ribosome-protected fragments. Tm, tunicamycin. All ρ values reported are Spearman correlation coefficients.

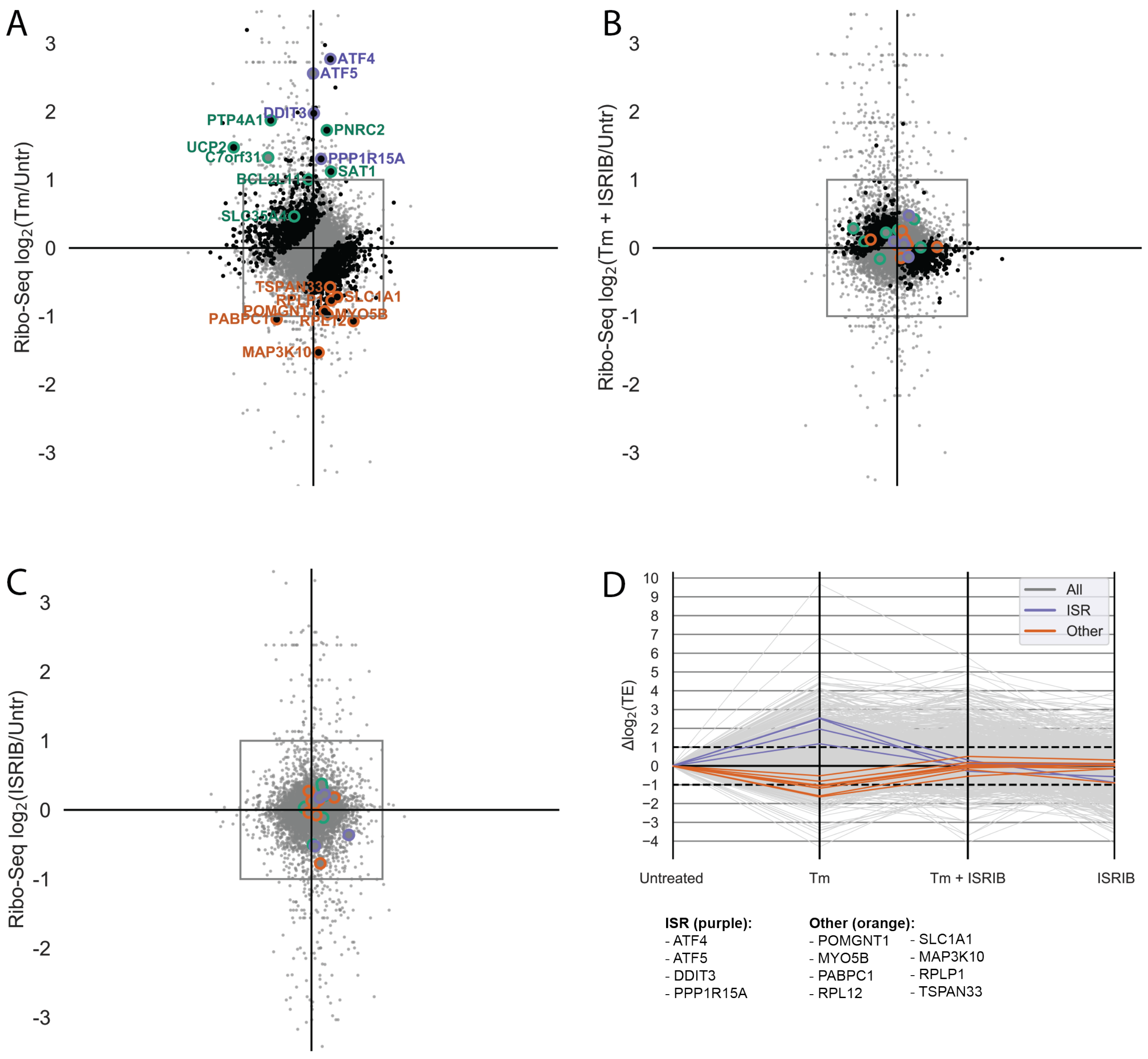


Figure 3: **Analysis of previously published ISR TE data using XPRESSpipe.** A-C) Log_2(Fold Change) for each drug condition compared to untreated for the ribosome profiling and RNA-Seq data. Purple, ISR canonical targets highlighted in the original study. Green, genes with uORFs affected by ISR as highlighted in the original study. Orange, genes fitting a strict thresholding paradigm to identify genes that display a 2-fold or greater increase in TE in Tm + ISRIB treatment compared to Tm treatment. Black, genes with statistically significant changes. Grey, all genes. Changes in ribo-seq and mRNA-Seq were calculated using DESeq2. TE was calculated using DESeq2 (see methods). Points falling outside of the plotted range are not included. D) Changes in $\log_2$(TE) for each drug condition compared to untreated control. Grey, all genes. Purple, ISR targets identified in the original study. Orange, genes fitting a strict thresholding paradigm to identify genes that display a 2-fold or greater increase in TE in Tm + ISRIB treatment compared to Tm treatment.

ISRIB [50–53]. Importantly, several of these genes were not identified as having significantly down-regulated TE in the original analysis, which suggests why a focus on translational downregulation may have been foregone. In all, we identified eight genes with the regulatory paradigm of interest: significant decreases in TE during tunicamycin-induced ISR that are rescued in the ISR + ISRIB condition (Table 2, descriptions sourced from https://www.genecards.org/, https://www.ncbi.nlm.nih.gov/gene/, and https://www.uniprot.org/uniprot/; annotations accessed 27 Jun 2019) (Figure 3D). RNA-Seq and ribosome-footprint coverage across these genes shows that the significant changes in their TE are due to neither spurious, high-abundance fragments differentially present across libraries nor variance from an especially small number of mapped reads (Figure S4). This is an important consideration as the use of CircLigase in the library preparation can bias certain molecules' incorporation in sequencing libraries [55].

Four (POMGNT1, SLC1A1, MAP3K10, TSPAN33) out of the eight identified genes have annotated neurological functions or are integrally tied to a pathway that functions in neurogenesis, which suggests their regulation may be functionally important for the neurodegenerative effects of ISR and the neuroprotective properties of ISRIB. For example, SLC1A1 is a glutamate transporter expressed throughout the brain where it plays vital roles in neurotransmission and glutamate homeostasis. Glutamate transporters, like SLC1A1, have also been implicated in preventing neurotrauma within the first few minutes of insult, and deficits in this transporter can lead to neurotoxic levels of glutamate within a cell [56]. Finally, down-regulation of SLC1A1 has already been implicated in diseases such as neurodegenerative diseases caused by mutations in the eukaryotic translation initiation factor 2B subunit epsilon (eIF2B5) that mimic the effects of phosphorylated eIF2α on cellular stress response and translation [48, 49, 57]. This suggests that TE regulation of SLC1A1 abundance control by translation initiation factors may play a role in the neurodegeneration observed in prolonged ISR conditions. ISRIB's neuroprotective effects may, therefore, stem from a recovery of SLC1A1 protein levels to wild-type levels, which in turn helps regulate glutamate levels. Though speculative, these ISRIB-responsive neuronal targets act as interesting cases for further validation and study in a model more closely mirroring the neuronal environment than the HEK-293T model used in the original study. In all, this comparison demonstrates the utility of XPRESSpipe for rapid, user-friendly analysis and re-analysis of ribosome-profiling experiments in the pursuit of biological insights and hypothesis generation.

Table 2: **Translationally down-regulated genes during acute Tm treatment and recovered regulation during Tm + ISRIB treatment.** Gene names with an asterisk indicate these genes were identified as down-regulated during Tm treatment in the original manuscript.

Gene Name	Relevant Description
POMGNT1	Participates in O-mannosyl glycosylation. Mutations have been associated with muscle-eye-brain diseases and congenital muscular dystrophies. Expressed especially in astrocytes, as well as in immature and mature neurons. Expressed across brain stem cells.
MYO5B*	May be involved in plasma membrane recycling. Identified in the original ISRIB study. No related neurological annotations.
PABPC1	Binds the poly(A) tail of mRNA. Promotes ribosome recruitment and translation initiation. May contribute to mRNA stability. No related neurological annotations.
RPL12	Ribosomal subunit. No related neurological annotations.
SLC1A1	Dense expression in substantia nigra, red nucleus, hippocampus, and cerebral cortical layers. Member of high-affinity glutamate transporter. In the brain, crucial for terminating postsynaptic action of the neurotransmitter glutamate. Responsible for maintaining glutamate concentrations below neurotoxic levels.
MAP3K10*	Functions in JNK signaling, reportedly involved in nerve growth factor induces neuronal apoptosis. Expressed in the cerebral cortex. Activates NEUROD1, which promotes neuronal differentiation.
RPLP1	Ribosome subunit. Evidence for stem cell and embryonic expression in the cerebral cortex.
TSPAN33	Plays a role in normal erythropoiesis and regulates maturation and trafficking of ADAM10, a metalloprotease. Negatively regulates Notch activity by way of its regulation of ADAM10. Notch signaling is vital to neurogenesis.

2.3.2 Performance Validation Using TCGA Data

To further validate the general design and reliability of the XPRESSpipe pipeline, we processed raw TCGA sequence data using XPRESSpipe and compared the output count values to those publicly available through TCGA (https://portal.gdc.cancer.gov/). Spearman ρ values for the selected samples ranged from 0.984-0.986 (Figure 4), indicating XPRESSpipe performs with similar accuracy to the TCGA RNA-Seq processing standards.

The differences in reported counts can be accounted for by a couple of key differences. For instance, the XPRESSpipe-processed files are aligned to the *Homo sapiens* GRChv96 reference transcriptome, while the original count data are aligned to the GRChv79 reference transcriptome. The use of a different transcriptome reference can result in large differences in the final quantified data for several genes (Figure S5). For example, in the four years between these versions, significant advances have been made in our understanding of transcribed regions of the human genome. Between versions 95 and 96 alone (version 95 published 24 Nov 2018, version 96 published 13 Mar 2019), at least 32,259 records were added (quantified by difference in line numbers between the files, although many additional records have been removed or modified).

Another source of dissimilarity in data processing appears to arise if an Ensembl canonical transcripts-only reference is used during quantification. TCGA-processed data used an un-modified transcriptome reference file (all

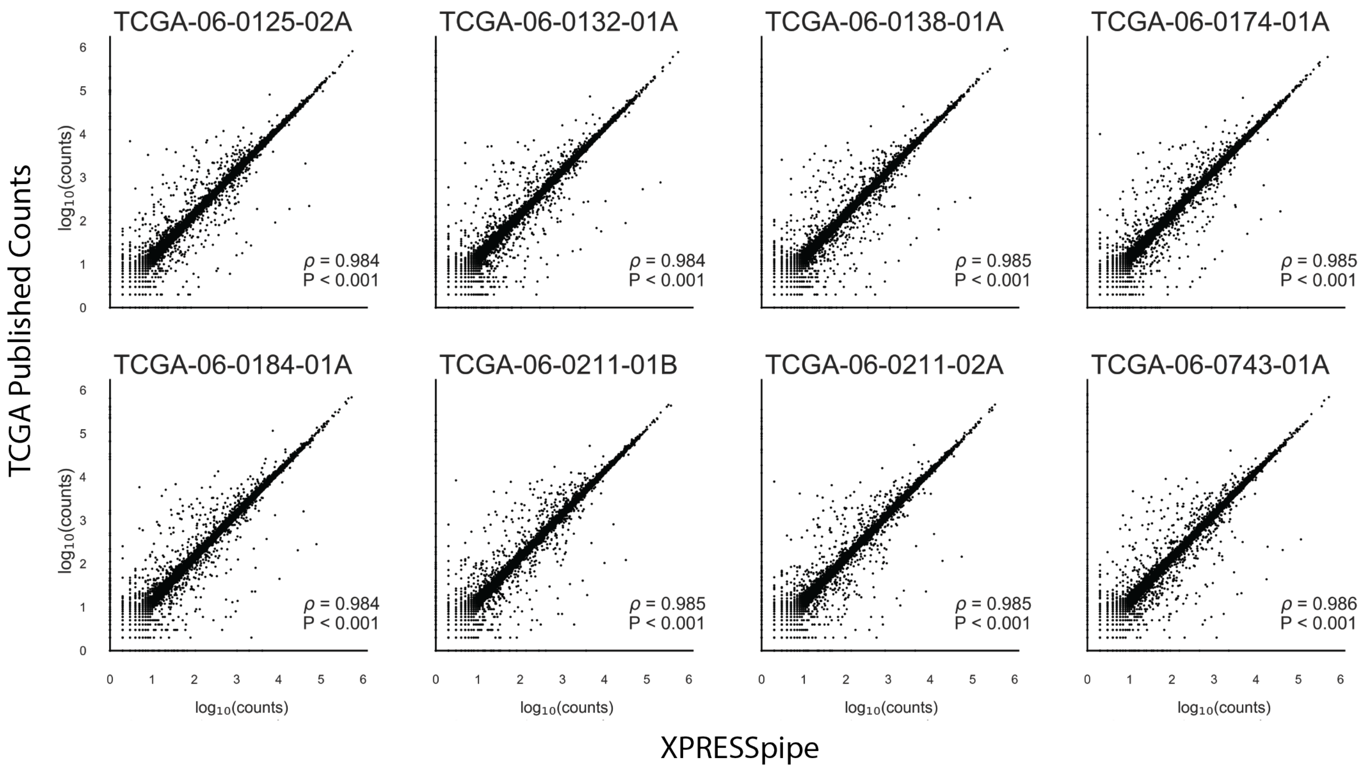


Figure 4: **Pipeline validation using publicly available TCGA count data.** Correlations were calculated between publicly available count data from TCGA samples and the count data processed by XPRESSpipe. Pseudogenes were excluded from the analysis. All reported ρ values are Spearman correlation coefficients.

transcripts); therefore, the use of this modified (longest transcript only) GTF will produce varied quantification for some genes as quantifications are constrained to a single transcript version of a given gene and a read will not be quantified if mapping to an exon not used by the canonical transcript. Even using XPRESSpipe settings closest to the TCGA pipeline and using the same genome and transcriptome version resulted in some variation (Figure S5, plot enclosed in maroon). By performing a more detailed analysis of these differences, it is clear that virtually all genes exhibiting variance between the processing methods are pseudogenes, with the TCGA pipeline broadly attributing more reads to pseudogenes. This can be indicative of the difficultly surrounding the recognition of these reads as multi-mapping to both the original gene and pseudogene (Figure S6, S7, S8. Interactive plots accompanying Figure S8 can be accessed at https://github.com/XPRESSyourself/xpressyourself_manuscript/tree/master/supplemental_files).

2.3.3 Cost Analysis

XPRESSpipe functions can be computationally intensive, and thus, super-computing resources are recommended, especially when dealing with large datasets, or when aligning to larger genomes, such as *Homo sapiens*. Many

Table 3: **XPRESSpipe processing statistics for dataset GSE65778.**

Metric	Value
Elapsed Real Time	09h32m58s
Total CPU Time	1d10h11m42s
Allocated CPUs	16
Allocated Memory per Node	62.50GB
Maximum RAM of all tasks	62.60GB

universities provide super-computing resources to their affiliates; however, in cases where these resources are not available, servers such as Amazon Web Services (AWS) (https://aws.amazon.com/) can be used to process sequencing data using XPRESSpipe. The ISRIB ribosome profiling dataset discussed above contained a total of 32 raw sequence files. Table 3 outlines runtime statistics, providing an example of the time required to process a dataset.

Based on these metrics, if high-performance computing services are not locally available, processing could be performed on a platform such as Amazon Web Services to process the data for relatively little money. For a comparable run, compute cost for a similar computational node for the given elapsed time would cost approximately 7.50 USD using Amazon EC2 On-Demand m5.4xlarge node (however, significantly reduced rates are available if using Spot instances or by using the free tier and storage cost would amount to around 14 USD/month on Amazon S3 storage (calculations were performed 28 Jun 2019). However, in most cases, the final quantified data and quality control metrics are relatively small in size and could thus be stored locally long-term.

3 Discussion

We have described a new software suite, XPRESSyourself, which includes a set of tools to aid in expression data processing and analysis. Although RNA-Seq technologies are becoming more and more established, standardized computational protocols are much less established for some applications. This is problematic when individuals or groups may not be using the most up-to-date methods or be aware of particular biases or measures of quality control required to produce a reliable, high-quality sequencing study. XPRESSpipe handles these issues through on-going curation of benchmarked software tools and by simplifying the required user input. It also outputs all necessary quality control metrics so that the user can quickly assess the quality of their data and identify any systematic problems or technical biases that may be present in their samples.

A particular strong point of XPRESSyourself is that it consolidates, streamlines, and/or introduces many tools specific to ribosome profiling processing and analysis. This includes producing GTF files with $5'$ and $3'$ truncated CDS annotations, rRNA probe design for subtractive hybridization of abundant rRNA contaminants, and automated quality-control analyses to report on ribosome footprint periodicity and metagene coverage. These tools will help democratize many aspects of ribosome profiling that previously had been challenging for many users.

An additional problem XPRESSpipe addresses is the incorrect use of these software tools, which is especially important for those coming from a non-computational background. XPRESSyourself will dissolve this barrier-to-entry for most users so that they can process and analyze their data immediately upon receipt of the raw data and only requires simple programming knowledge covered by simple video walkthroughs, example scripts, and interactive command builders.

We demonstrated the utility of the XPRESSyourself toolkit by re-analyzing a publicly available ribosome profiling dataset. From this analysis, we identified genes that may contribute to the neurodegenerative effects of the integrated stress response (ISR) and how the molecule ISRIB may regulation the translation of these genes to confer neuroprotection. This additionally highlights the importance of re-analyzing older datasets with more current methods, as improved analysis methodologies and updated organism genome references may result in new interpretations and hypotheses.

XPRESSyourself will enable individuals and labs to process and analyze their own data, which will often result in a quicker turnaround of experiments as well as monetary savings. XPRESSyourself will also put missing or incomplete computational tools required for ribosome profiling and RNA-Seq into the hands of the user. Additionally, inclusion of detailed log reports, summaries of software dependencies used during runtime, and containerized versions of the pipeline where dependencies are archived and self-contained will aid in reproducibility and make transparent methods easy to incorporate into associated publications.

4 Conclusions

With the adoption of this flexible pipeline, the field of high-throughput sequencing, particularly within ribosome profiling, can continue to standardize the processing protocol for sequencing data and eliminate some of the variability that comes from using a variety of software packages for various steps during read processing. Additionally, XPRESSpipe consolidates various tools used by the ribosome profiling and RNA-Seq communities. With these tools, such as the GTF modification sub-module, genome reference formatting and curation is automated and accessible to the public. Further, by using this pipeline on publicly available data, we highlight XPRESSpipe's

Table 4: **Summary of dependency software, accession location, and purpose in the XPRESSpipe package.**

Package	**Purpose**	**Reference**
Python	Primary language	
R	Language used for some statistical modules	
fastp	Read pre-processing	[25]
STAR	Reference curation and read alignment	[21]
samtools	Alignment file manipulation	[29]
bedtools	Alignment file manipulation	[30]
Cufflinks	Read quantification (primary)	[24]
HTSeq	Read quantification	[20]
FastQC	Quality Control	[34]
MultiQC	Quality Control	[58]
Pandas	Data manipulation	[59]
NumPy	Data manipulation	[42, 43]
SciPy	Data manipulation	[44]
scikit-learn	Data manipulation	[60]
Matplotlib	Plotting	[39]
XPRESSplot	Normalization and matrix manipulation	This paper
dupRadar	Perform library complexity calculations	[35]
riboWaltz	Perform p-site offset calculations	[37]
DESeq2	Perform differential expression analysis	[38]

utility in being able to re-process publicly available data or personal data to uncover novel biological patterns quickly. Adoption of this tool will aid scientists with quickly accessing their data.

5 Materials and Methods

Methods described in this manuscript apply to the software packages at the time of writing. To obtain the most current methods, please refer to the documentation or source code for a given module.

5.1 Software Dependencies

A list of dependencies required for XPRESSpipe at the time of writing is listed in Table 4. Dependencies for XPRESSplot at the time of writing are listed in Table 5.

5.2 GTF Modification

To parallelize GTF modification, a GTF file is split into approximately proportional chunks equal to the specified number of threads. To avoid an incomplete gene record being included in a chunk, a given chunk endpoint is determined by calculating the size of the GTF, dividing by the number of threads, and advancing one chunk size forward in line number, then advancing line by line until the last line of the current gene record. This is performed for each subsequent chunk. If creating the last chunk, the end of the chunk is automatically the last line of the GTF

Table 5: **Summary of dependency software, accession location, and purpose in the XPRESSplot package.**

Package	**Purpose**	**Reference**
Python	Primary language	
R	Language used for some statistical modules	
Pandas	Data manipulation	[59]
NumPy	Data manipulation	[42, 43]
SciPy	Data manipulation	[44]
Matplotlib	Plotting	[39]
Seaborn	Plotting	[40]
Plotly	Interactive plotting	[61]
scikit-learn	Data manipulation	[60]
SVA	Perform batch correction for known effects with the ComBat function	[33]

record.

Ensembl canonical transcripts are determined according to the Ensembl glossary definition of a canonical transcript [23]. For cases where a tie-breaker exists with equal priority transcripts, the longest is chosen. When there are more than one of these equal priority transcripts of the same length exist, the first listed record is retained. Exon or CDS lengths are calculated by taking the sum of each exon or CDS, not including intron or other space in the calculation.

Protein-coding records are retained by performing a simple string search for the “protein_coding” annotation in the attribute column of a GTF file.

Truncation of records is performed by identifying the $5'$ and $3'$ end of each transcript and modifying the given coordinates to reflect the given truncation amounts. Suggested truncation amounts are 45 nt from the $5'$ end and 15 nt from the $3'$ end, both of which are set as the default truncation amount parameters for the function and do not need to be modified unless the user desires [5]. As a given CDS portion of a given exon may be less than a truncation amount, the function will recursively search CDS by CDS per transcript until the full truncation amount is fully trimmed. Any record smaller than the sum of the $5'$ and $3'$ truncation amounts is removed from the output file. Searching is performed in a strand-specific manner.

5.3 GTF Flattened Record

Flattened transcriptome references are created for meta-analysis modules by providing a GTF. This curation operates in a similar manner to the methods employed by general GTF modification; however, only exon or CDS records are retained with other minimal key information (i.e. strandedness). These records are then divided into a multidimensional array, where each parent array contains all the records for a given chromosome.

5.4 Normalization

Equations 1-4 reflect the design of the normalization functions within XPRESSplot, where *g* is gene *n*, *ge* is cumulative exon space for gene *n*, *r* is total reads, *f* is total fragments, and *l* is length.

$$RPM_{\mathsf{g}} = \frac{1e6 \cdot r_{ge}}{\sum_{g=1}^{n} r_{ge}} \tag{1}$$

$$RPKM_{\mathsf{g}} = \frac{1e9 \cdot r_{ge}}{(\sum_{g=1}^{n} r_{ge}) \cdot l_{ge}} \tag{2}$$

$$FPKM_{\mathsf{g}} = \frac{1e9 \cdot f_{ge}}{(\sum_{g=1}^{n} f_{ge}) \cdot l_{ge}} \tag{3}$$

$$TPM_{\mathsf{g}} = \frac{1e6 \cdot r_{ge}}{(\sum_{g=1}^{n}(\frac{1e3 \cdot r_{ge}}{l_{ge}})) \cdot l_{ge}} \tag{4}$$

5.5 Quality Control Summary Plotting

Summary plots are created using Pandas [59] and Matplotlib [39]. Kernel density plots for library complexity analyses are created using NumPy [42, 43] and SciPy's `gaussian_kde` function [44].

5.6 Metagene Estimation

Metagene calculations are performed by determining the meta-genomic coordinate *M* for each aligned read, where L_e is the leftmost mapped position of the read (strand agnostic) and *r* is the length of the mapped read. *S* denotes the start coordinate for the transcript and l_e is the cumulative length of all exons for the given transcript. The subscripted *e* indicates the coordinate is relative to exon space (intronic ranges within a transcript do not contribute to total space calculation). Required inputs are a BAM file and an un-modified GTF reference file, which is then curated into its longest-transcript, protein-coding-only flattened form, as discussed above. If a longest-transcript, protein-coding-only modified GTF has already been curated, this can alternatively be provided as input, with which the module will flatten (file suffix must be "LC.gtf"). For each mapped coordinate, the metagene position is calculated as:

$$M = \frac{|(L_{\mathsf{e}} \ + \ \frac{1}{2}r) \ - \ S| \ \cdot \ 100}{l_e} \tag{5}$$

In the case where a mapped coordinate falls within multiple genes, a penalty is assigned as:

$$c = \frac{1}{n} \tag{6}$$

Where *c* is the count score for a given meta-position and *n* is the number of different transcripts a given coordinate mapped. To be counted or factored into the penalty, the meta-position coordinate must fall within exon space.

5.7 Gene Coverage Plotting

Gene coverage calculations are performed by determining the exon space of the gene of interest and mapping any read for a given sample to this space. Each nucleotide that maps to these exon regions is counted. During plotting, a rolling window of 20 nucleotides is used to smoothen the plotted coordinates' read coverage. Required inputs are a BAM file and an un-modified GTF reference file, which is then curated into its longest-transcript, protein-coding-only flattened form, as discussed above. If a longest-transcript, protein-coding-only modified GTF has already been curated, this can alternatively be provided as input, with which the module will flatten.

5.8 Periodicity

Ribosome p-site periodicity is calculated using riboWaltz [37]. Required inputs are the path to a directory containing transcriptome-aligned BAM files (ending in "toTranscriptome.out.bam") and the path and file name of the appropriate un-modified GTF.

5.9 rRNA Probe

`rrnaProbe` works on a directory containing FastQC [34] zip compressed files to detect over-represented sequences for each sample. These sequences are then collated to create consensus fragments. One caveat of FastQC is that it collates on exact matching strings, but these strings, or sequences, can be 1 nt steps from each other and a single rRNA probe could be used to effectively pull out all these sequences. In order to handle this situation, XPRESSpipe will combine these near matches. A rank-ordered list of over-represented fragments within the appropriate length range to target for depletion is then output. A BLAST [62] search on these consensus sequences intended for probe usage can then be performed to verify the fragment maps to an rRNA sequence and is thus a suitable depletion probe.

5.10 Confidence Interval Plotting

Confidence intervals within PCA scatterplots generated by XRESSplot are calculated as follows:

1. Compute the covariance of the two principal component arrays, *x* and *y* using the numpy.cov() function.

2. Compute the eigenvalues and normalized eigenvectors of the covariance matrix using the numpy.linalg.eig() function.

3. Compute the θ of the normalized eigenvectors using the numpy.arctan2() function and converting the output from radians to degrees using numpy.deg().

4. Compute the λ of the eigenvalues by taking the square root of the eigenvalues.

5. Plot the confidence intervals over the scatter plot: The center point of the confidence interval is determined from the means of the *x* and *y* arrays. The angle is set equal to θ. The width of the confidence interval is calculated by

$$w = \lambda_x \cdot ci \cdot 2$$

where *ci* is equal to the corresponding confidence level (i.e. 68% = 1, 95% = 2, 99% = 3). The height is similarly computed by

$$h = \lambda_y \cdot ci \cdot 2$$

5.11 Ribosome Profiling Data Analysis

Raw data were obtained from GEO (GSE65778). Reference files were taken from Ensembl Human build GRCh38 version 96. Read alignments were quantified using an XPRESSpipe-modified GTF file that contained only protein-coding records and the $5'$ ends of each CDS truncated by 45 nucleotides and the $3'$ ends truncated by 15 nucleotides. All associated figures and analyses can be reproduced using the associated scripts found at https://github.com/XPRESSyourself/xpressyourself_manuscript (DOI: XXXXXX). See https://github.com/XPRESSyourself/xpressyourself_manuscript/tree/master/isrib_analysis/batch_run_docs for scripts used to process raw data.

Only gene names in common between the original data file and XPRESSpipe output were used for the method comparisons. Correlation between methods or replicates were calculated using a Spearman rank correlation coefficient, performed using the `scipy.stats.spearman()` function [63]. The associated script can be accessed at https://github.com/j-berg/xpressyourself_manuscript/blob/master/isrib_analysis/isrib_analysis.py.

Differential expression analyses were performed using all genes, but with a minimum count of 10 or greater per gene across samples, as recommended by the DESeq2 documentation [38]. Differential expression for ribo-seq and RNA-Seq was performed as detailed in the associated scripts (https://github.com/j-berg/xpressyourself_manuscript/blob/master/isrib_analysis/isrib_analysis.py, https://github.com/j-berg/xpressyourself_manuscript/blob/master/isrib_analysis/isrib_de/isrib_de_analysis.py, https://github.com/j-berg/xpressyourself_manuscript/blob/master/isrib_analysis/isrib_de/run_de.sh). For these analyses, the design formula was such that comparisons were designed as "treated" factor level over "untreated" factor level. Differential expression of translation efficiencies between conditions used the additional incorporation of the "ribosome footprint" factor level over "RNA-Seq" factor level in the design formula [5, 38, 49]. Adjusted p-values (FDRs) in the associated figures were calculated from the differential expression of the translation efficiencies of each gene for a given condition. Those passing an adjusted p-value threshold of less than or equal to 0.1 are highlighted in black.

Intron-agnostic gene coverage profiles were generated using XPRESSpipe's geneCoverage module. Comparison plots were generated using IGV [36]. Interactive scatter plots were generated using Plotly Express [61].

5.12 TCGA Data Analysis

Raw data and processed TCGA count data was obtained from the TCGA Portal (https://portal.gdc.cancer.gov/) via dbGap controlled access (https://www.ncbi.nlm.nih.gov/gap/). Raw data were processed on a protected high-performance computing environment. Correlations between methods or replicates were calculated using a Spearman rank correlation coefficient, performed using the `scipy.stats.spearman()` function [63]. The associated script can be accessed at https://github.com/j-berg/xpressyourself_manuscript/blob/master/tcga_data/tcga_validation.py. Interactive scatter plots were generated using Plotly Express [61] See https://github.com/j-berg/xpressyourself_manuscript/tree/master/tcga_data/batch_process_info for scripts used to process data.

5.13 Cost Analysis

Cost analysis was performed by assessing run logs from the high-performance computing cluster and using published AWS prices (https://aws.amazon.com/ec2/pricing/on-demand/, https://aws.amazon.com/s3/pricing/, accessed 28 June 2019) to calculate the relative cost for a comparable run.

List of abbreviations

AWS - Amazon Web Services, BAM - Binary Sequence Alignment Map, BED - Browser Extensible Data, cDNA - complementary DNA, CDS - coding region of gene, ChIP-seq - chromatin immunoprecipitation sequencing, CPU - central processing unit, dbGaP - Database of Genotypes and Phenotypes, DNA - deoxyribonucleic acid, FDR - false discovery rate, FPKM - fragments per kilobase of transcript per million, GEO - Gene Expression Omnibus, GTF - General Transfer Format, IGV - Integrative Genomics Viewer, ISR - integrated stress response, ISRIB - ISR inhibitor, nt - nucleotide, PCA - principal component analysis, PCR - polymerase chain reaction, RAM - random access memory, RNA - ribonucleic acid, RNA-Seq - RNA sequencing RPKM - reads per kilobase of transcript per million, RPM - reads per million, rRNA - ribosomal RNA, TCGA - The Cancer Genome Atlas, TE - translation efficiency, TPM - transcripts per million, UMI - unique molecular identifier

Ethics approval and consent to participate

Protected TCGA data were obtained through dbGaP project number 21674 and utilized according to the associated policies and guidelines.

Consent for publication

Protected TCGA data were obtained through dbGaP project number 21674 and utilized according to the associated policies and guidelines.

Availability of data and materials

The source code for these packages is perpetually open source and protected under the GPL-3.0 license. The code can be publicly accessed and installed from https://github.com/XPRESSyourself. Updates to the software are version controlled and maintained on GitHub. Jupyter notebooks and video walkthroughs are included within the README files at https://github.com/XPRESSyourself. Documentation is hosted on readthedocs [64] at https://xpresspipe.readthedocs.io/en/latest/ and https://xpressplot.readthedocs.io/en/latest/. The publicly available ribosome profiling data are accessible through GEO series accession number GSE65778. TCGA data are accessible through dbGaP accession number phs000178. Code used to create manuscript figures and analyses can be found at https://github.com/XPRESSyourself/xpressyourself_manuscript (DOI: XXXXXX).

Table 6: **Author ORCIDs**

Author	ORCID
JAB	0000-0002-5096-0558
JRB	0000-0001-5470-8299
JTM	0000-0002-3017-8665
AJB	0000-0003-2273-8922
YO	0000-0001-9523-1044
ARQ	0000-0003-1756-0859
JG	0000-0001-7568-6789
JR	0000-0002-2710-9765

Competing interests

The authors declare that they have no competing interests.

Funding

J.A.B. received support from the National Institute of Diabetes and Digestive and Kidney Diseases (NIDDK) Inter-disciplinary Training Grant T32 Program in Computational Approaches to Diabetes and Metabolism Research, 1T32DK11096601 to Wendy W. Chapman and Simon J. Fisher. J.T.M. received support as an HHMI Fellow of the Jane Coffin Childs Memorial Fund for Medical Research. A.J.B received support from the National Cancer Institute (NCI) Predoctoral to Postdoctoral Fellow Transition Award, K00CA212445.

Contributions

J.A.B. conceptualized and administered the project; performed all investigation, analysis, visualization, and data curation; provisioned resources; acquired funding; and wrote the original draft for this study. J.A.B. and J.R.B. designed and wrote the software. J.A.B., J.T.M., A.J.B., and Y.O. performed software validation. J.A.B., J.R.B., J.T.M., and J.G. and developed the methodology. J.R., A.R.Q., and J.G. supervised the study. All authors were involved in reviewing and editing the manuscript.

Acknowledgments

The authors wish to thank Michael T. Howard for helpful discussions concerning ribosome profiling and sequencing analysis. The authors also wish to thank Mark E. Wadsworth, Ryan Miller, and Michael J. Cormier for helpful discussions on pipeline design. They also wish to thank T. Cameron Waller for helpful discussions related to pipeline design and biological analysis. The support and resources from the Center for High-Performance Computing at the

University of Utah are gratefully acknowledged. The computational resources used were partially funded by the NIH Shared Instrumentation Grant 1S10OD021644-01A1. The results published here are in whole or part based upon data generated by the TCGA Research Network: https://www.cancer.gov/tcga.

References

[1] S. Byron, K. V. Keuren-Jensen, D. Engelthaler, J. Carpten, D. Craig, *Nat Rev Genet* **17**, 392–393 (2016).

[2] N. Ingolia, S. Ghaemmaghami, J. Newman, J. Weissman, *Science* **324**, 218 (2009).

[3] Z. Costello, H. Martin, *NPJ Syst Biol Appl* **4** (2018).

[4] V. Funari, S. Canosa, *Science* **344**, 653 (2014).

[5] N. McGlincy, N. Ingolia, *Methods* **126**, 112 (2017).

[6] M. Gerashchenko, V. Gladyshev, *Nucleic Acids Res* **42** (2014).

[7] C. Artieri, H. Fraser, *Genome Res* **24**, 2011 (2014).

[8] J. Hussmann, S. Patchett, A. Johnson, S. Sawyer, W. Press, *PLoS Genet* **11** (2015).

[9] D. Weinberg, *et al.*, *Cell Rep* **14**, 1787 (2016).

[10] ENCODE, `https://www.encodeproject.org/rna-seq/`.

[11] GDC, `https://docs.gdc.cancer.gov/Data/Bioinformatics_Pipelines/Expression_mRNA_Pipeline/`.

[12] P. Lab, `https://github.com/PavlidisLab/rnaseq-pipeline`.

[13] Nextflow, `https://github.com/nf-core/rnaseq`.

[14] U. Genetics, `https://github.com/UMCUGenetics/RNASeq`.

[15] C. G. Informatics, `https://github.com/cellgeni/rnaseq`.

[16] DNAnexus, `https://github.com/dnanexus/tophat_cufflinks_rnaseq`.

[17] Nextflow, `https://www.nextflow.io/example4.html`.

[18] E. Afgan, *et al.*, *Nucleic Acids Res.* **46**, W537 (2018).

[19] C. Wu, B. Zinshteyn, K. Wehner, R. Green, *Mol Cell* **73** (2019).

[20] S. Anders, P. Pyl, W. Huber, *Bioinformatics* **31**, 166 (2015).

[21] A. Dobin, *et al.*, *Bioinformatics* **29**, 15 (2013).

[22] G. Baruzzo, *et al.*, *Nat Methods* **14**, 135 (2017).

[23] Ensembl, `https://uswest.ensembl.org/Help/Glossary`.

[24] C. Trapnell, *et al.*, *Nat Protoc* **7** (2012).

[25] S. Chen, Y. Zhou, Y. Chen, J. Gu, *Bioinformatics* **34** (2018).

[26] Y. Fu, P. Wu, T. Beane, P. Zamore, Z. Weng, *BMC Genomics* **19** (2018).

[27] T. Smith, A. Heger, I. Sudbery, *Genome Res* **27** (2017).

[28] G. Baruzzo, *et al.*, *Nat Methods* **14**, 135–139 (2017).

[29] H. Li, *et al.*, *Bioinformatics* **25**, 2078 (2009).

[30] A. Quinlan, I. Hall, *Bioinformatics* **26**, 841 (2010).

[31] C. Robert, M. Watson, *Genome Biol* **16** (2015).

[32] C. Evans, J. Hardin, D. Stoebel, *Brief Bioinform* **19**, 776–792 (2018).

[33] J. Leek, W. Johnson, H. Parker, A. Jaffe, J. Storey, *Bioinformatics* **28** (2012).

[34] S. Andrews, Fastqc, `http://www.bioinformatics.babraham.ac.uk/projects/fastqc/` (2010).

[35] S. Sayols, D. Scherzinger, H. Klein, *BMC Bioinformatics* **17**, 428 (2016).

[36] J. Robinson, *et al.*, *Nat Biotechnol* **29**, 24 (2011).

[37] F. Lauria, *et al.*, *PLoS Comput Biol* **14** (2018).

[38] M. Love, W. Huber, S. Anders, *Genome Biol* **15** (2014).

[39] J. Hunter, *Computing In Science & Engineering* **9**, 90 (2007).

[40] M. Waskom, et al (2012).

[41] F. Pedregosa, *et al.*, *Journal of Machine Learning Research* **12**, 2825 (2011).

[42] T. Oliphant, *A guide to NumPy* (Trelgol Publishing, USA, 2006).

[43] S. van der Walt, S. Colbert, G. Varoquaux, *Computing in Science Engineering* **13**, 22 (2011).

[44] E. Jones, T. Oliphant, P. Peterson, *et al.*, Scipy: Open source scientific tools for python, `http://www.scipy.org/` (2001).

[45] H. Harding, *et al.*, *Mol Cell* **11** (2003).

[46] D. Santos-Ribeiro, L. Godinas, C. Pilette, F. Perros, *Drug Discov Today* **23** (2018).

[47] H. Rabouw, *et al.*, *Proc Natl Acad Sci U S A* **116** (2019).

[48] J. Tsai, *et al.*, *Science* **359** (2018).

[49] C. Sidrauski1, A. McGeachy, N. Ingolia, P. Walter, *eLIFE* (2015).

[50] A. Choua, *et al.*, *Proc Natl Acad Sci U S A* **114** (2017).

[51] M. Halliday, *et al.*, *Cell Death Dis* **6** (2015).

[52] C. Sidrauski, *et al.*, *Elife* **2** (2013).

[53] Y. Sekine, *et al.*, *Science* **348** (2015).

[54] D. Kim, *et al.*, *Genome Biol* **14** (2013).

[55] R. Tunney, *et al.*, *Nat Struct Mol Biol* **25**, 577 (2018).

[56] L. Vera-Portocarrero, *et al.*, *Brain Res* **927** (2002).

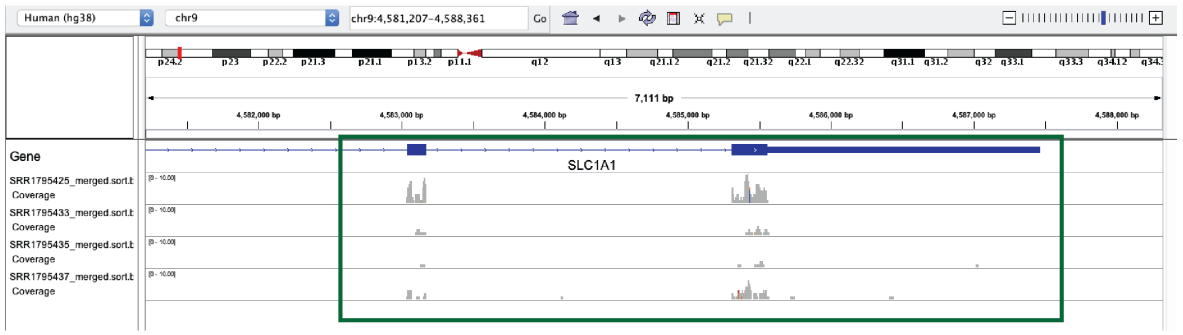
[57] Y. Wong, *et al.*, *Elife* **8** (2019).

[58] P. Ewels, M. Magnusson, S. Lundin, M. Käller, *Bioinformatics* **32**, 3047–3048 (2016).

[59] W. McKinney, *Proc of the 9th Python in Science Conf* pp. 51–56 (2010).

[60] L. Buitinck, *et al.*, *ECML PKDD Workshop: Languages for Data Mining and Machine Learning* (2013), pp. 108–122.

A



B

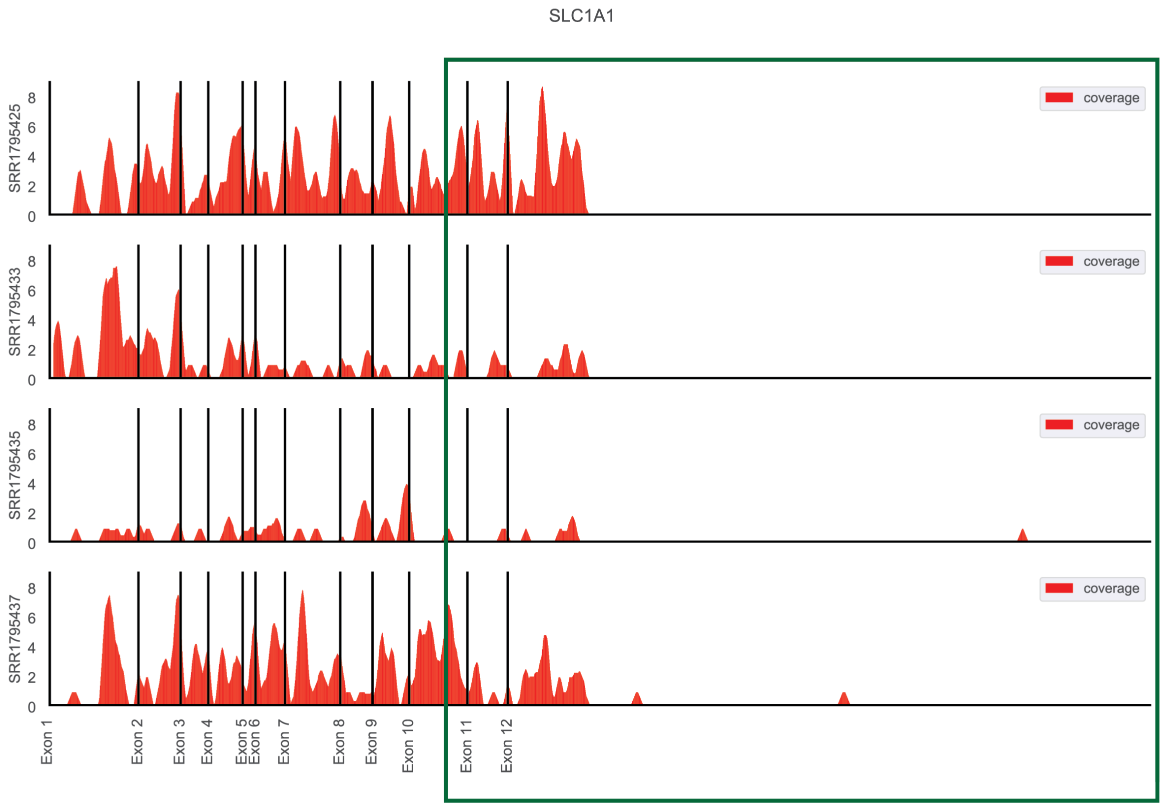


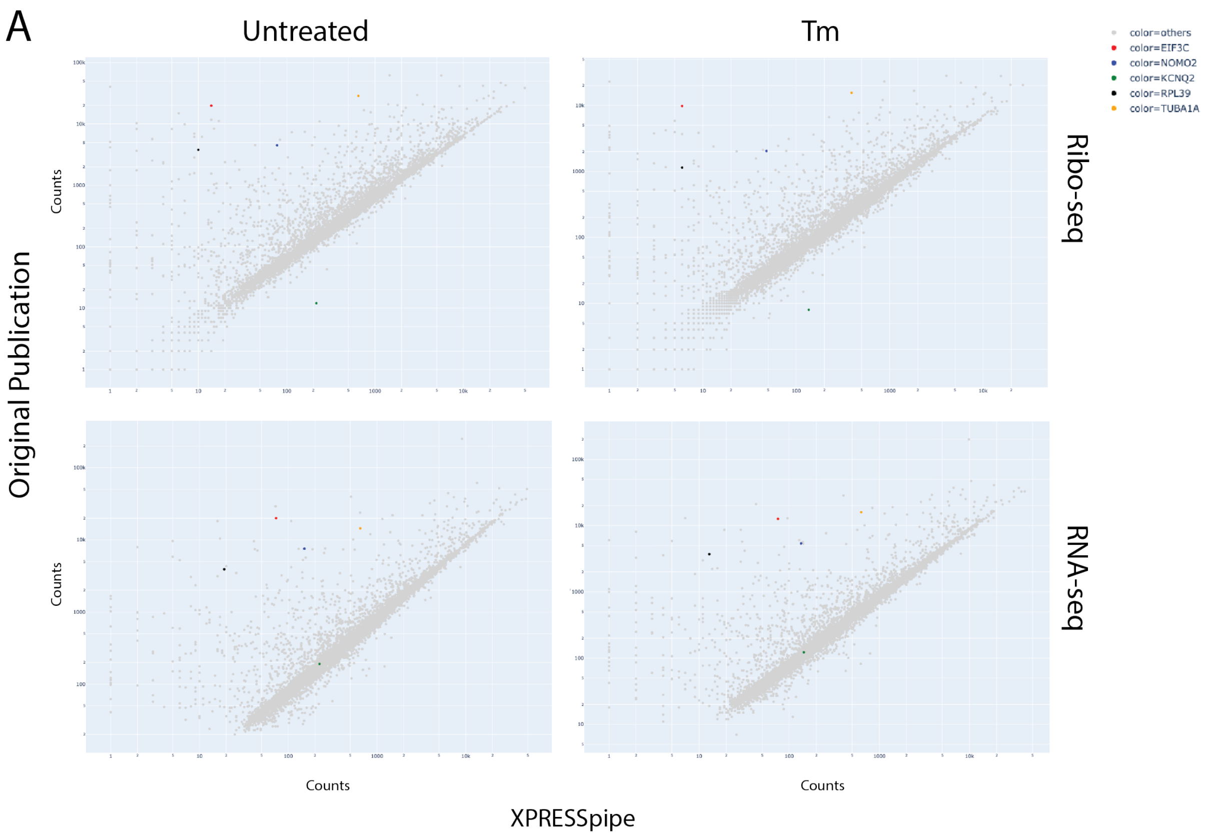
Figure S1: **Comparison between IGV browser and geneCoverage output.** A snapshot of the same samples in IGV (with introns) and the geneCoverage summary (introns collapsed). Green box, exons 11 and 12 of SLC1A1 in human.

[61] P. T. Inc., Collaborative data science (2015).

[62] S. Altschul, W. Gish, W. Miller, E. Myers, D. Lipman, *J Mol Biol.* **215**, 403 (1990).

[63] F. Liesecke, *et al.*, *Sci Rep* **8** (2018).

[64] Read the docs, `https://readthedocs.org/`.



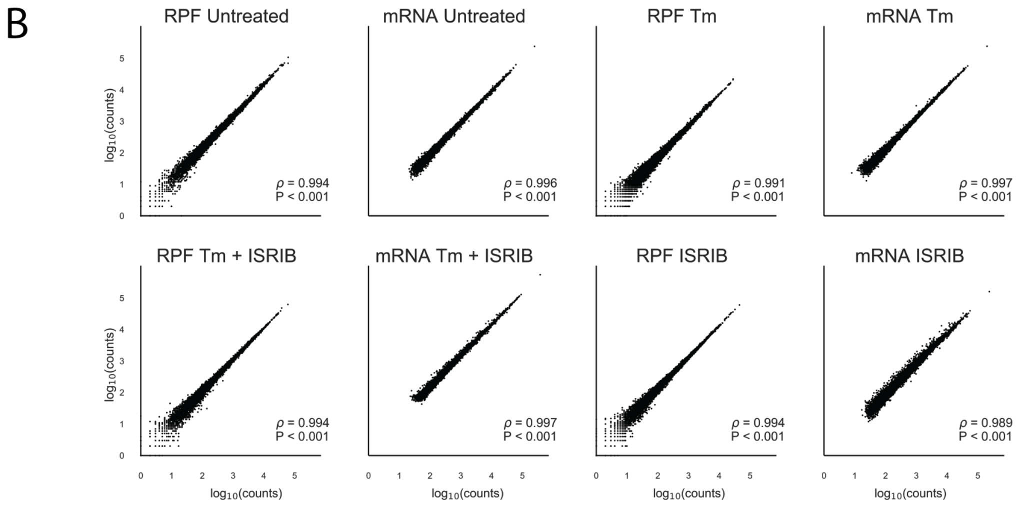


Figure S2: **Original ISRIB count data plotted against XPRESSpipe-processed data reveals systematic differences between the analytical regimes.** A) Selected highlighted genes show consistent differences between processing methods. B) Spearman correlation plots using the data table provided as supplementary data with the original ISRIB manuscript comparing biological replicates. RPF, ribosome-protected footprint. Tm, tunicamycin. All ρ values reported are Spearman correlation coefficients.

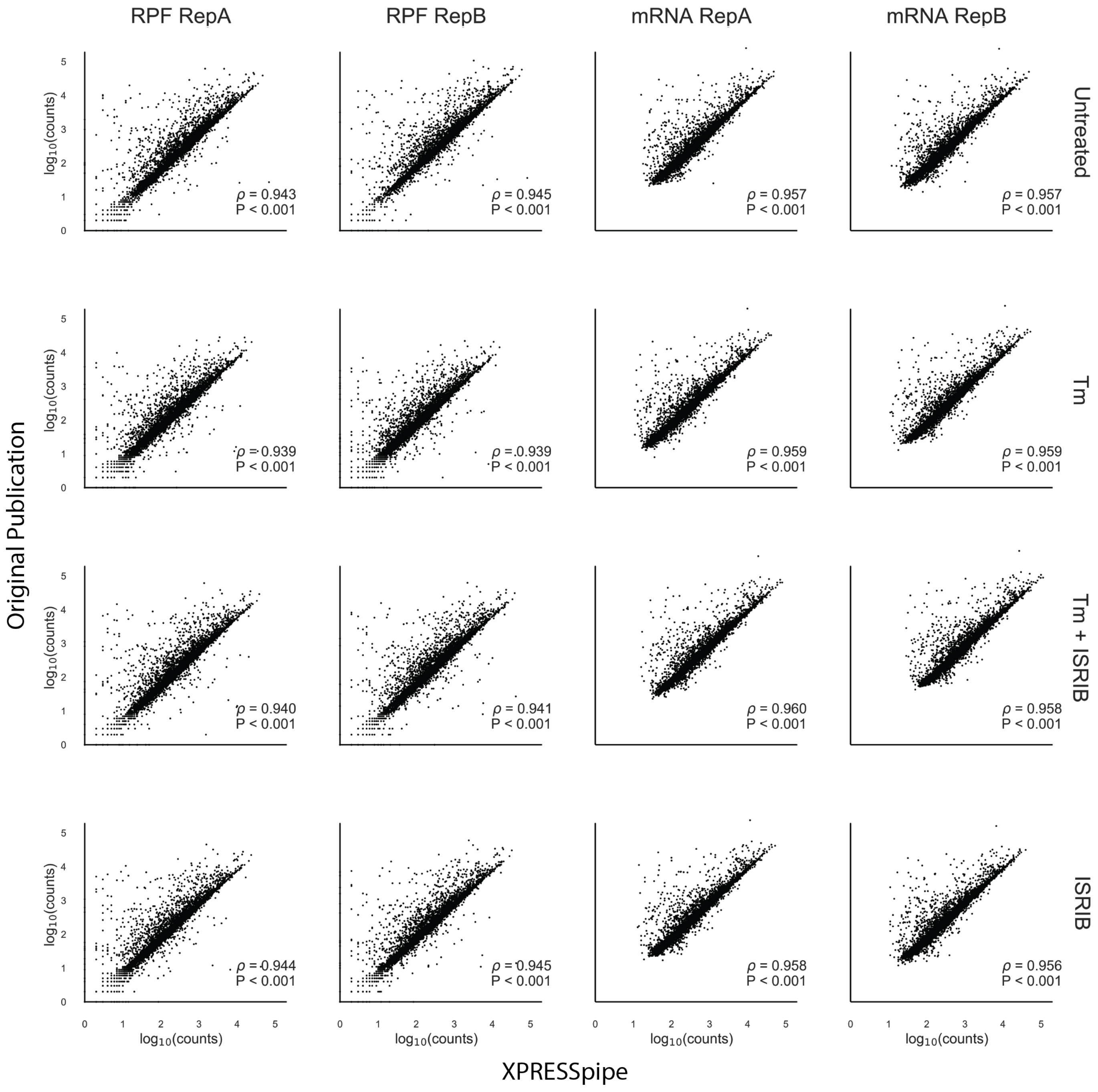


Figure S3: **Original ISRIB count data plotted against XPRESSpipe-processed data quantifying with same reference version reveals negligible improvement in comparability between the analytical regimes.** Original samples were processed using Ensembl human build GRCh38 v72, as in the original manuscript, and compared with the original count data provided with the manuscript. XPRESSpipe-prepared counts were thresholded similarly as the original data (each gene needed to have at least 10 counts across all mRNA samples). RepA, biological replicate A. RepB, biological replicate B. RPF, ribosome-protected footprint. Tm, tunicamycin. All ρ values reported are Spearman correlation coefficients.

SLC1A1

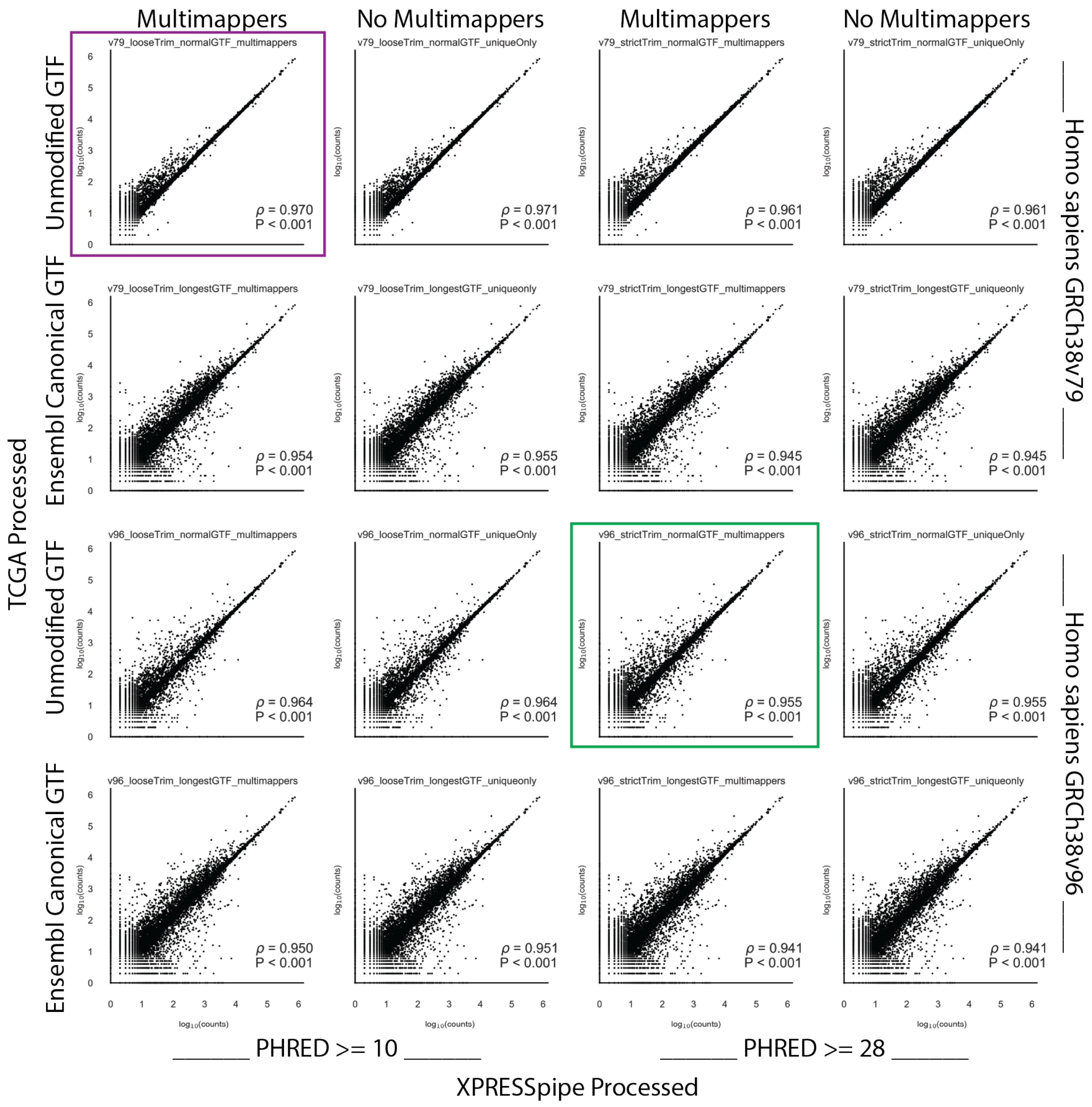


Figure S5: **Sample RNA-Seq count data compared between TCGA count data and various conformations of the XPRESSpipe pipeline.** An overview of how different conformations of the XPRESSpipe peRNAseq pipeline compared to the published TCGA sample TCGA-06-0132-01A count data. The x-axis data in the plot enclosed in maroon most closely mirrors the settings used in the published TCGA RNA-Seq pipeline. The x-axis data in the plot enclosed in green used XPRESSpipe default settings and the most current reference transcriptome at the time of writing. All ρ values reported are Spearman correlation coefficients.

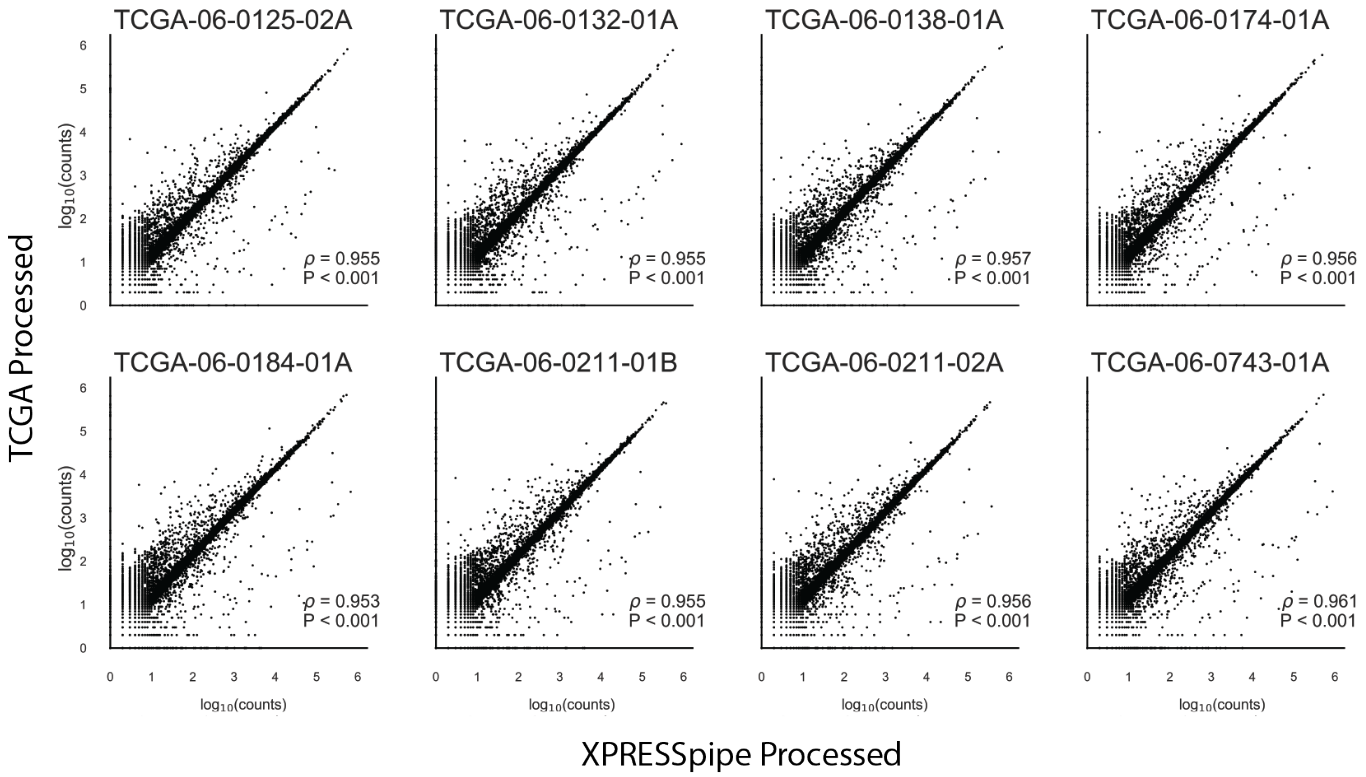


Figure S6: **Effect of pseudogene inclusion on comparability between processing regimes.** Spearman correlations between XPRESSpipe and TCGA-processed count data with pseudogene counts included. All ρ values reported are Spearman correlation coefficients.

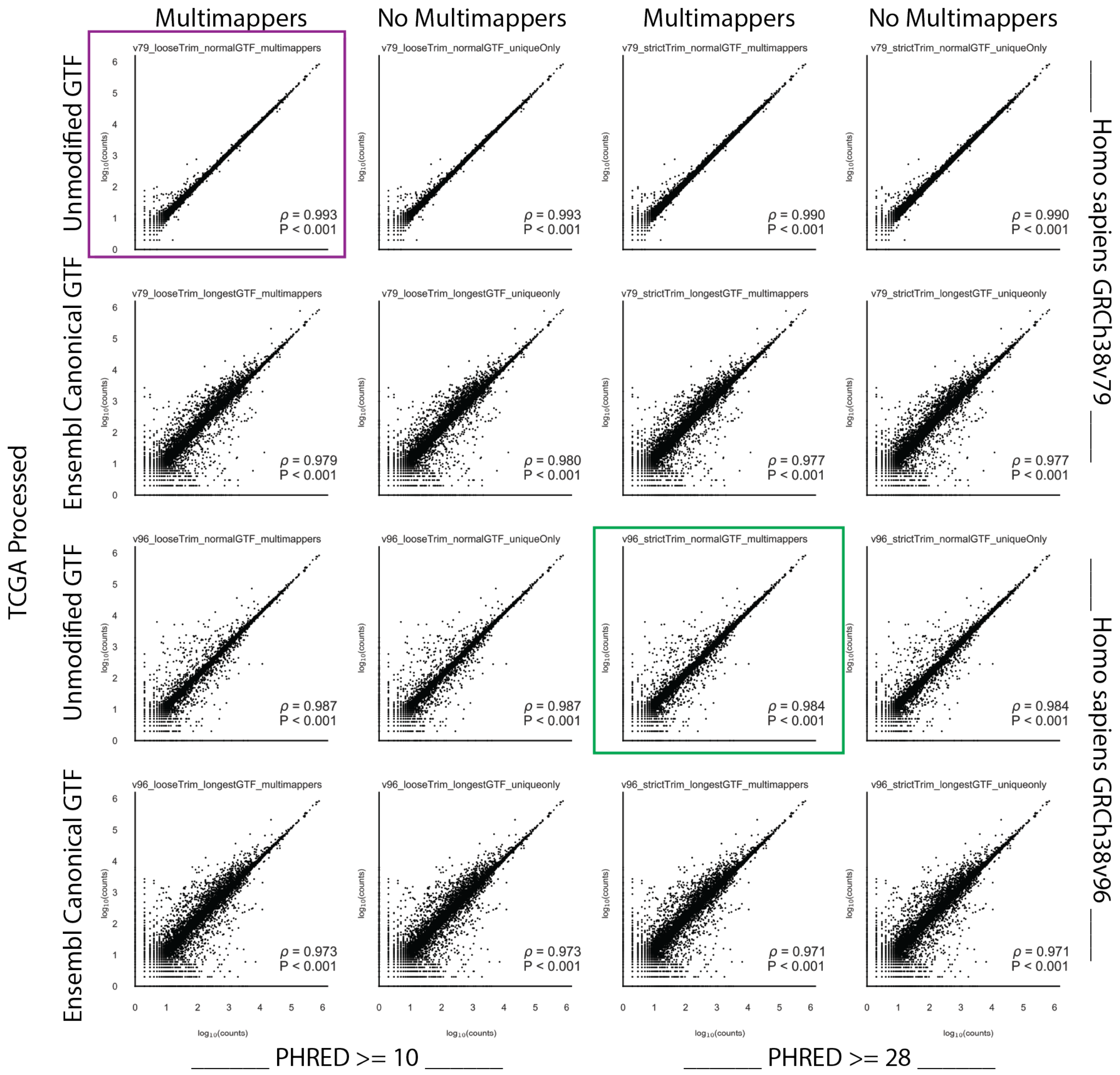


Figure S7: **Removal of pseudogenes counts improve comparability between analytical regimes.** An overview of how different conformations of the XPRESSpipe peRNAseq pipeline compared to the published TCGA sample TCGA-06-0132-01A count data with pseudogenes collapsed. The x-axis data in the plot enclosed in maroon most closely mirrors the settings used in the published TCGA RNA-Seq pipeline. The x-axis data in the plot enclosed in green used XPRESSpipe default settings and a current reference transcriptome. All ρ values reported are Spearman correlation coefficients.

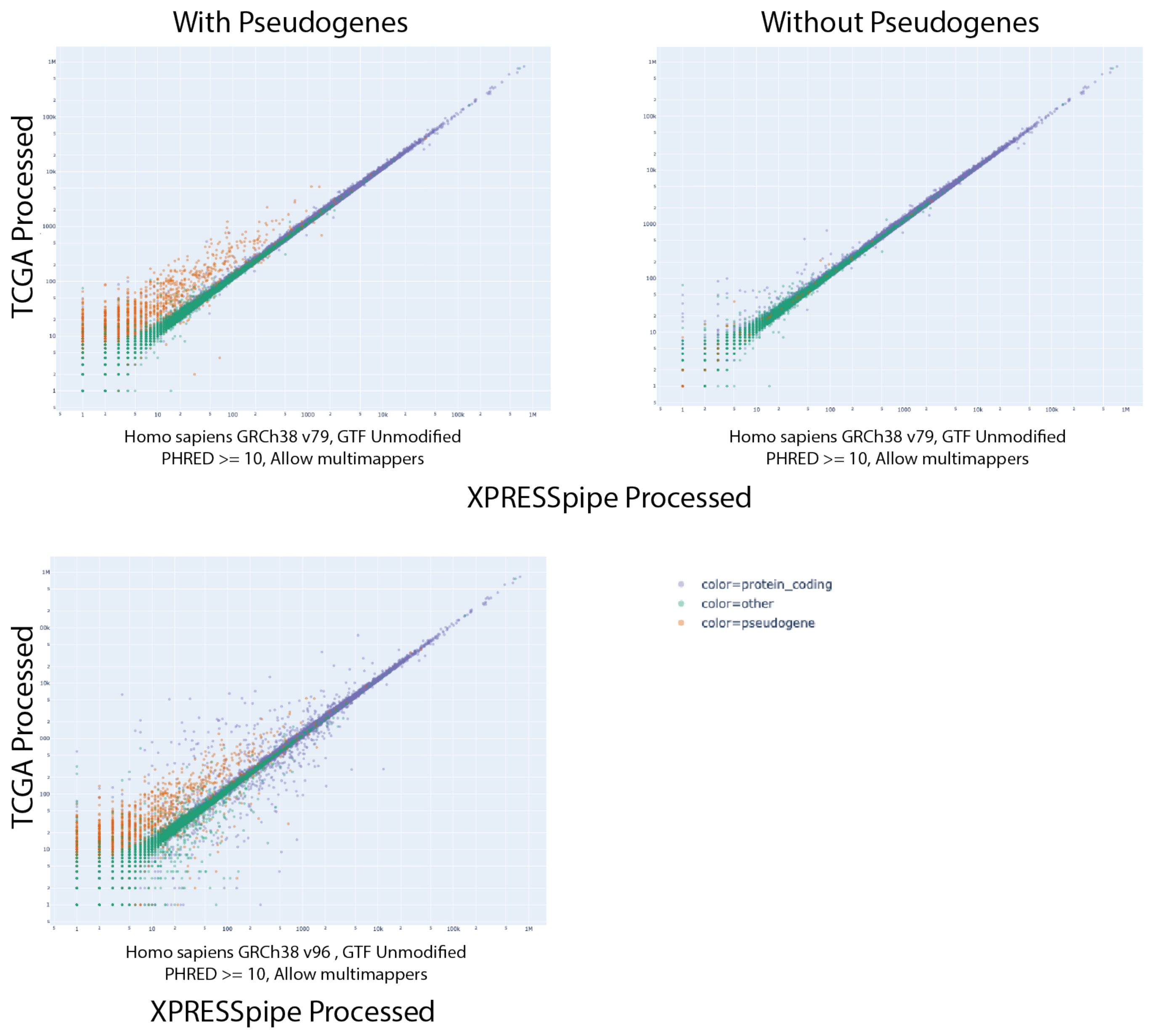


Figure S8: **Pseudogenes counts are over-represented in TCGA-processed data.** An overview of gene-type distributions between transcriptome reference versions. The plots above used GRCh38v79 and the bottom plot used GRCh38v96. Purple points, protein-coding genes. Orange points, pseudogenes. Green points, other gene records. All plots represent sample TCGA-06-0132-01A and were processed the same way except for transcriptome reference used during read quantification.