The Oyster River Protocol: A Multi Assembler and Kmer Approach For *de novo* Transcriptome Assembly

Matthew D. MacManes[1, *, •, ⋆]

[1] Department of Molecular, Cellular and Biomedical Sciences, University of New Hampshire, Durham NH, USA

[*] E-mail: macmanes@gmail.com

[•] Twitter: @MacManes

[⋆] Mailing Address: 46 College Road, 189 Rudman Hall. Durham NH 03824

Abstract

Characterizing transcriptomes in non-model organisms has resulted in a massive increase in our understanding of biological phenomena. This boon, largely made possible via high-throughput sequencing, means that studies of functional, evolutionary and population genomics are now being done by hundreds or even thousands of labs around the world. For many, these studies begin with a *de novo* transcriptome assembly, which is a technically complicated process involving several discrete steps. The Oyster River Protocol (ORP), described here, implements a standardized and benchmarked set of bioinformatic processes, resulting in an assembly with enhanced qualities over other standard assembly methods. Specifically, ORP produced assemblies have higher `TransRate` scores and mapping rates, which is largely a product of the fact that it leverages a multi-assembler and kmer assembly process, thereby bypassing the shortcomings of any one approach. These improvements are important, as previously unassembled transcripts are included in ORP assemblies, resulting in a significant enhancement of the power of downstream analysis. Further, as part of this study, we show that assembly quality is unrelated to taxonomy, nor is it related to the number of reads generated, above 30 million reads. **Code Availability:** The version controlled open-source code is available at `https://github.com/macmanes-lab/Oyster_River_Protocol`. Instructions for software installation and use, and other details are available at `http://oyster-river-protocol.rtfd.org/`.

Competing Interests

The author declares no competing interests.

1 Introduction

For all biology, modern sequencing technologies has provided for an unprecedented opportunity to gain a deep understanding of genome level processes that underlie a very wide array of natural phenomena, from intracellular metabolic processes to global patterns of population variability. Transcriptome sequencing has been influential, particularly in functional genomics, and has resulted in discoveries not possible even just a few years ago. This in large part is due to the scale at which these studies may be conducted. Unlike studies of adaptation based on one or a small number of candidate genes (e.g. (1; 2)), modern studies may assay the entire suite of expressed transcripts – the transcriptome – simultaneously. In addition to issues of scale, as a direct result of enhanced dynamic range, newer sequencing studies have increased ability to simultaneously reconstruct and quantitate lowly- and highly-expressed transcripts, (3; 4). Lastly, improved methods for the detection of differences in gene expression (*e.g.,* (5; 6)) across experimental treatments has resulted in increased resolution for studies aimed at understanding changes in gene expression.

As a direct result of their widespread popularity, a diverse toolset for the assembly and analysis of transcriptome exists. Notable amongst the wide array of tools include several for quality visualization - `FastQC` (available here) and `SolexaQA` (7), read trimming (e.g. `Skewer` (8), and `Trimmomatic` (9), read normalization (`khmer` (10)), error correction (SEECER (11) and RCorrector (12)), assembly (`Trinity` (13), `SOAPdenovoTrans` (14)), and assembly verification (`TransRate` (15)), `BUSCO` (Benchmarking Universal Single-Copy Orthologs - (16)), and `RSEM-eval` (17)). The ease with which these tools may be used to produce transcriptome assemblies belies the true complexity underlying the overall process. Indeed, the subtle (and not so subtle) methodological challenges associated with transcriptome reconstruction may result in highly variable assembly quality. Production of an accurate transcriptome assembly requires a large investment in time and resources. Each step in it's production requires careful consideration. Here, I propose an evidence-based protocol for assembly that results in the production of the high quality transcriptome assemblies.

This manuscript describes the development of a multi-assembler and multi-kmer protocol. This innovation is critical, as all assembly solutions treat the read data in ways that bias transcript recovery. Specifically, the development of assembly software comes the use of a set of heuristics, that are necessary given the scope of the assembly problem itself. Given each software development team carries with it a unique set of ideas related to these heuristics, individual assemblers exhibit unique assembly behavior. By leveraging a multi-assembler approach, the strengths of one assembler may complement the weaknesses of another. In addition to biases related to assembly heuristics, it is well known that assembly kmer-length has important effects on transcript reconstruction, with shorter kmers more efficiently reconstructing lower-abundance transcripts relative to longer assembly kmer-lengths. Given this, assembling with multiple different kmer lengths, then merging the resultant assemblies may effectively reduce this type of bias. Recognizing these issue, I hypothesize that an assembly that resulted from the combination of multiple different assemblers and lengths of assembly-kmers would be better than each individual assembly, across a variety of metrics.

2 Methods

2.1 Datasets

In an effort at benchmarking the assembly and merging protocols, I downloaded a set of publicly available RNAseq datasets (Table 1) that had been produced on the Illumina sequencing platform. These datasets were chosen to represent a variety of taxonomic groups, so as to demonstrate the broad utility of the developed methods. Because datasets were selected randomly with respect to sequencing center and read number, they are likely to represent the typical quality of Illumina data circa 2014-2017.

Table 1

Type	Accession	Species	Num. Reads	Read Length
Plant	DRR053698	*Cephalotus follicularis*	126M	90bp
Plant	DRR082659	*Aeginetia indica*	69M	90bp
Insect	ERR489297	*Anopheles gambiae*	206M	100bp
Tapeworm	DRR030368	*Echinococcus multilocularis*	73M	100bp
Plant	DRR031870	*Vigna angularis*	60M	100bp
Fish	DRR046632	*Oncorhynchus mykiss*	82M	76bp
Plant	DRR069093	*Hevea brasiliensis*	103M	100bp
Amoebozoa	ERR058009	*Entamoeba histolytica*	68M	100bp
Nematode	ERR1016675	*Heterorhabditis indica*	51M	100bp
Mammal	SRR2086412	*Mus musculus*	54M	100bp
Plant	SRR3499127	*Nicotiana tabacum*	30M	150bp
Mammal	SRR1789336	*Oryctolagus cuniculus*	31M	100bp
Polychaeta	SRR2016923	*Phyllodoce medipapillata*	86M	100bp
Schistosome	ERR1674585	*Schistosoma mansoni*	39M	100bp
Mammal	DRR036858	*Mus musculus*	114M	100bp

Table 1 lists the datasets used in this study. All datasets are publicly available for download by accession number at the European Nucleotide Archive.

2.2 Software

The Oyster River Protocol is implemented as a stand-alone makefile which coordinates all steps described below. All scripts are available at `https://github.com/macmanes-lab/Oyster_River_Protocol`, and run on the Linux platform. The software is version controlled and openly-licensed to promote sharing and reuse. A guide for users is available at `http://oyster-river-protocol.rtfd.io`.

2.3 Pre-assembly procedures

For all assemblies performed, Illumina sequencing adapters were removed from both ends of the sequencing reads, as were nucleotides with quality Phred ≤ 3, using the program `Trimmomatic` version 0.36 (9), following the recommendations from (18). After trimming, reads were error corrected using the software `RCorrector` version 1.0.2 (12), following recommendations from (19). The code for running this step of the Oyster River protocols is available at here. The trimmed and error corrected reads where then subjected to *de novo* assembly.

2.4 Assembly

I assembled each RNAseq dataset using three different *de novo* transcriptome assemblers and three different kmer lengths. First, I assembled the reads using `Trinity` release 2.4.0 (13), and default settings (k=25), without read normalization. Next, the `SPAdes` RNAseq assembler (version 3.10) (20) was used, in two distinct runs, using kmer sizes 55 and 75. Lastly, reads were assembled using the assembler `Shannon` version 0.0.2 (21), using a kmer length of 75. This assembly process resulted in the production of four distinct assemblies. The code for running this step of the Oyster River protocols is available here.

To compare the optimized Oyster River Protocol with a more standard workflow conducted where a single kmer length is used (k=25), trimmed (but not error corrected) reads were assembled using the default settings in `Trinity`, with the exception of digital normalization, which was not performed.

2.5 Assembly Merging via OrthoFuse

To merge the four assemblies produced as part of the Oyster River Protocol, I developed new software that effectively merges transcriptome assemblies. Described in brief, `OrthoFuse` begins by concatenating all assemblies together, then forms groups of transcripts by running a version of `OrthoFinder` (22) packaged with the ORP, modified to accept nucleotide sequences from the merged assembly. These groupings represent groups of homologous transcripts. Of note, the inflation parameter has been increase by default to I=4, to prevent the collapsing of transcript isoforms into a single groups. After `Orthofinder` has completed, a modified version of `TransRate` version 1.0.3 (15) which is packaged with the ORP, is run on the merged assembly, after which the best (= highest contig score) transcript is selected from each group and placed in a new assembly file to represent the entire group. The resultant file, which contains the highest scoring contig for each orthogroup, may be used for all downstream analyses. `OrthoFuse` is run automatically as part of the Oyster River Protocol, and additionally is available as a stand along script, here.

2.6 Assembly Evaluation

All assemblies were evaluated using `ORP-TransRate` and `BUSCO` version 3.0.2. `TransRate` evaluates transcriptome assembly contiguity by producing a score based on contig and mapping metrics, while `BUSCO` evaluates assembly content by searching the assembly for conserved single copy orthologs. In addition to this, final assemblies were compared to the Swissprot protein database using blastX (23) and an e-value of $1e^{-10}$.

2.7 Statistics

All statistics analyses were conducted in R version 3.4.0 (24). Violin plots were constructed using the beanplot (25) and the beeswarm R packages (`https://CRAN.R-project.org/package=beeswarm`). Expression distributions were plotted using the ggjoy package (`https://CRAN.R-project.org/package=ggjoy`). Plots for visualizing the unique content of each assembly were constructed using the UpsetR package (26).

3 Results

Fifteen RNAseq datasets, ranging in size from (30-206M paired end reads) were assembled using the Oyster River Protocol and with `Trinity`. Each assembly was evaluated using the software `BUSCO` and `TransRate`. From these, seven metrics were chosen to represent the quality of the produced assemblies. Of note, all the assemblies produced as part of this work are available here, and will be moved to dataDryad after acceptance.

3.1 Trinity-assembled transcripts

`Trinity` assemblies generally completed on standard a standard Linux server using 24 cores in less than 24 hours. RAM requirement is estimated to be close to 0.5Gb per million paired-end reads. The assemblies on average contained 176k transcripts (range 19k - 643k) and 97Mb (range 14MB - 198Mb). Other quality metrics will be discussed below, specifically in relation to the ORP produced assemblies.

3.2 Oyster River Protocol- assembled transcripts

ORP assemblies generally completed on standard a standard Linux server using 24 cores in three to five days. Typically `Trinity` was the longest running assembler, with the individual `SPAdes` assemblies being the shortest. RAM requirement is estimated to be 1.5Gb - 2Gb per million paired-end reads, with `SPAdes` requiring the most. The assemblies on average contained 153k transcripts (range 23k - 625k) and 64Mb (range 8MB - 181Mb).

3.2.1 Assembly Structure

The structural integrity of each assembly was evaluated using the `TransRate` software package. Using mapping metrics, I evaluated each of the `Trinity` and ORP produced assemblies (Figure 1). As many downstream application depend critically on read mapping, assemblies that maximize this metric are desirable. The split violin plot presented in figure 1A visually represent the mapping rates of each assembly, with lines connecting the mapping rates of datasets assembled with `Trinity` and with the ORP, respectively. The average mapping rate of the `Trinity` assembled datasets was 83% (sd=9%), while the average mapping rates of the ORP assembled datasets was 95% (sd=2%). This test is statistically significant (One sided Wilcoxon rank sum test, $p = 0.0001322$). Figure 1B describes the distribution of assembly scores, which is a synthetic metric taking into account multiple mapping and coverage-based statistics. The `Trinity` assemblies had an average score of 0.22 (sd = .1), while the ORP assembled datasets had an average score of 0.33 (sd = .08). This test is statistically significant (One sided Wilcoxon rank sum test, p-value = 0.01836). Lastly, figure 1C describes the distribution of optimal assembly scores, which is the same synthetic metric as above, but measured after the removal of poorly-supported transcripts. The `Trinity` assemblies had an average score of 0.32 (sd = .09), while the ORP assembled datasets had an average score of 0.45 (sd = .08). This test is statistically significant (One sided Wilcoxon rank sum test, p-value = 0.001351).

Figure 1

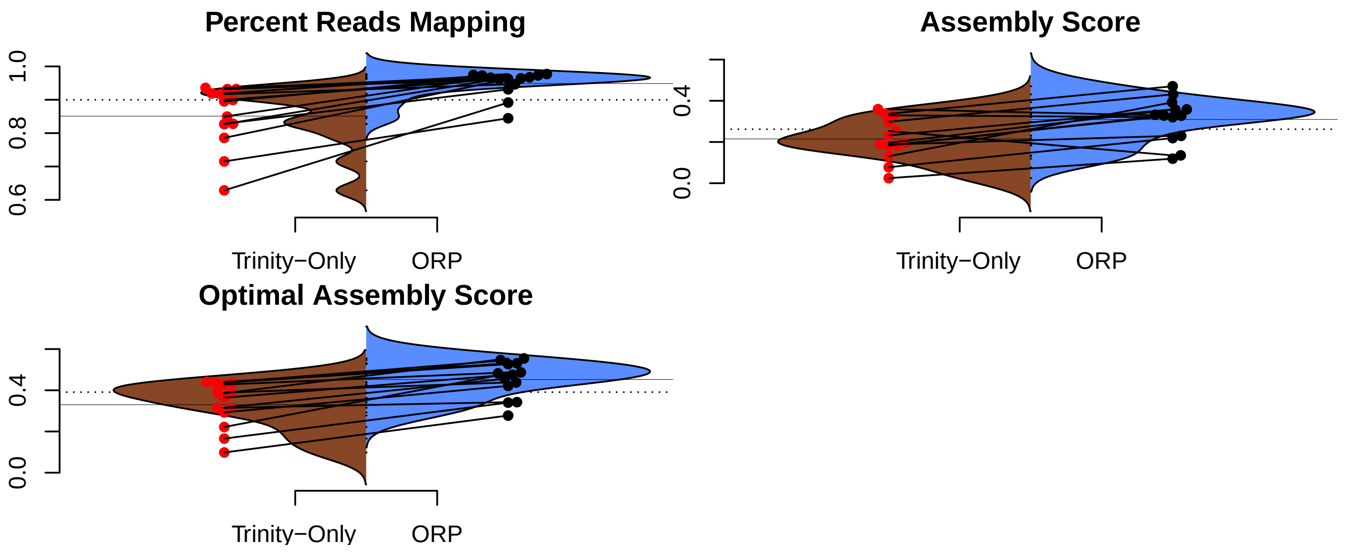

Figure 1. `TransRate` generated statistics. Split violin plots depict the relationship between `Trinity` assemblies (brown color) and ORP produced assemblies (blue color). Red and black dots indicate the value of a given metric for each assembly. Lines connecting the red and black dots connect datasets assembled via the two methods.

3.2.2 Assembly Content

The genic content of assemblies was measured using the software package `BUSCO` version 3.0.2, using the Eukaryota database. `Trinity` assemblies contained on average 86% (sd = 21%) of the full-length orthologs, while the ORP assembled datasets contained on average 85% (sd = 16%) of the full length transcripts. This different is not statistically significant (Figure 2A). Figure 2B depicts the percent of missing transcripts in `Trinity` and ORP assembled datasets. The `Trinity` and ORP assemblies each contained on average 4.4% (sd = 8.7%) missing orthologs. Figure 2C depicts the percent of transcripts that are reconstructed in fragmented (not full length) forms in `Trinity` and ORP assembled datasets. The `Trinity` assembled datasets contained 10% (sd = 17%) of fragmented transcripts while the ORP assemblies each contained on average 10.7% (sd = 13%) of fragmented orthologs. This difference is not statistically significantly different. The rate of transcript duplication, depicted in figure 2D is 47% (sd = 20%) for `Trinity` assemblies, and 34% (sd = 15%) for ORP assemblies. This result is statistically significant (One sided Wilcoxon rank sum test, p-value = 0.02953).

Figure 2

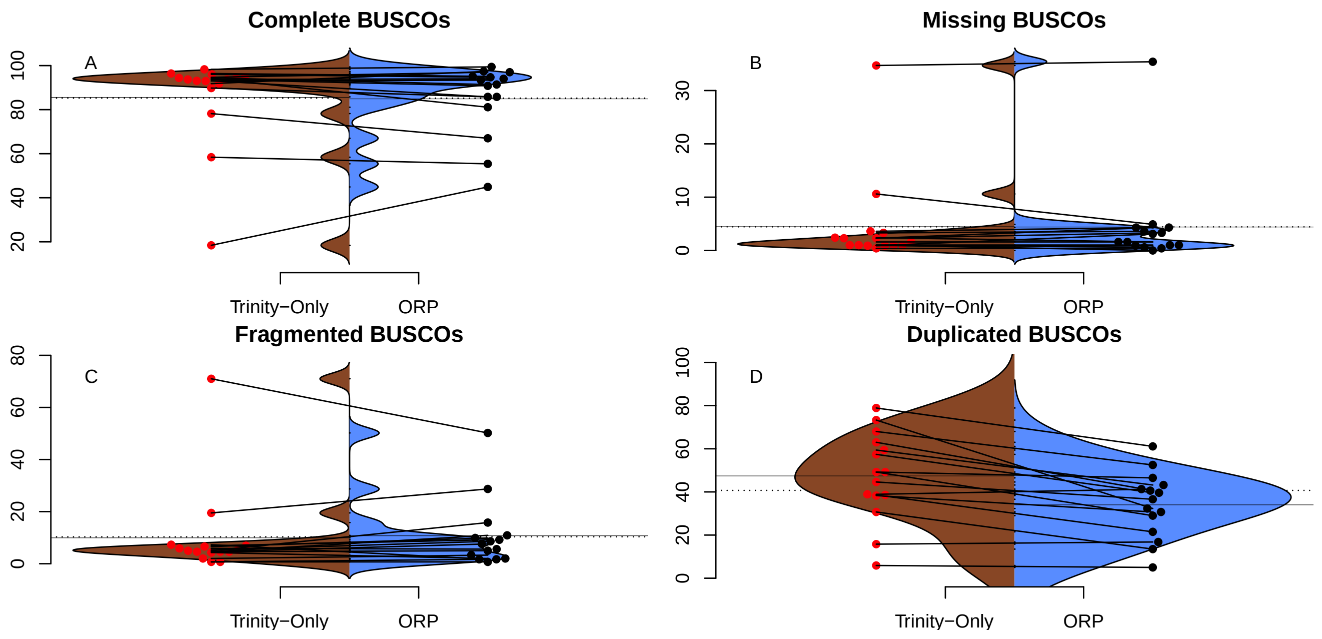

Figure 2. `BUSCO` generated statistics. Split violin plots depict the relationship between `Trinity` assemblies (brown color) and ORP produced assemblies (blue color). Red and black dots indicate the value of a given metric for each assembly. Lines connecting the red and black dots connect datasets assembled via the two methods.

3.2.3 Assembler Contributions

To understand the relative contribution of each assembler to the final merged assembly produced by the Oyster River Protocol, I counted the number of transcripts in the final merged assembly that originated from a given assembler. On average, 33% of transcripts in the merged assembly were produced by the `Trinity` assembler. 18% were produced by `Shannon`, while `SPAdes` produced the remaining 49% of transcripts.

To further understand the potential biases intrinsic to each assembler, I plotted the distribution of gene expression estimates for each merged assembly, broken down by the assembler of origin (Figure 3, depicting four randomly selected representative assemblies). As is evident, most transcripts are lowly expressed, with `SPAdes` and `Trinity` both doing a sufficient job in reconstructing these transcripts. Of note, the `SPAdes` assemblies using kmer-length=75 is biased, as expected, towards more highly expressed transcripts relative to kmer-length 55 assemblies. `Shannon` demonstrates a unique profile, consisting of, almost exclusively high-expression transcripts, given a previously undescribed bias against low-abundance transcripts.

Figure 3

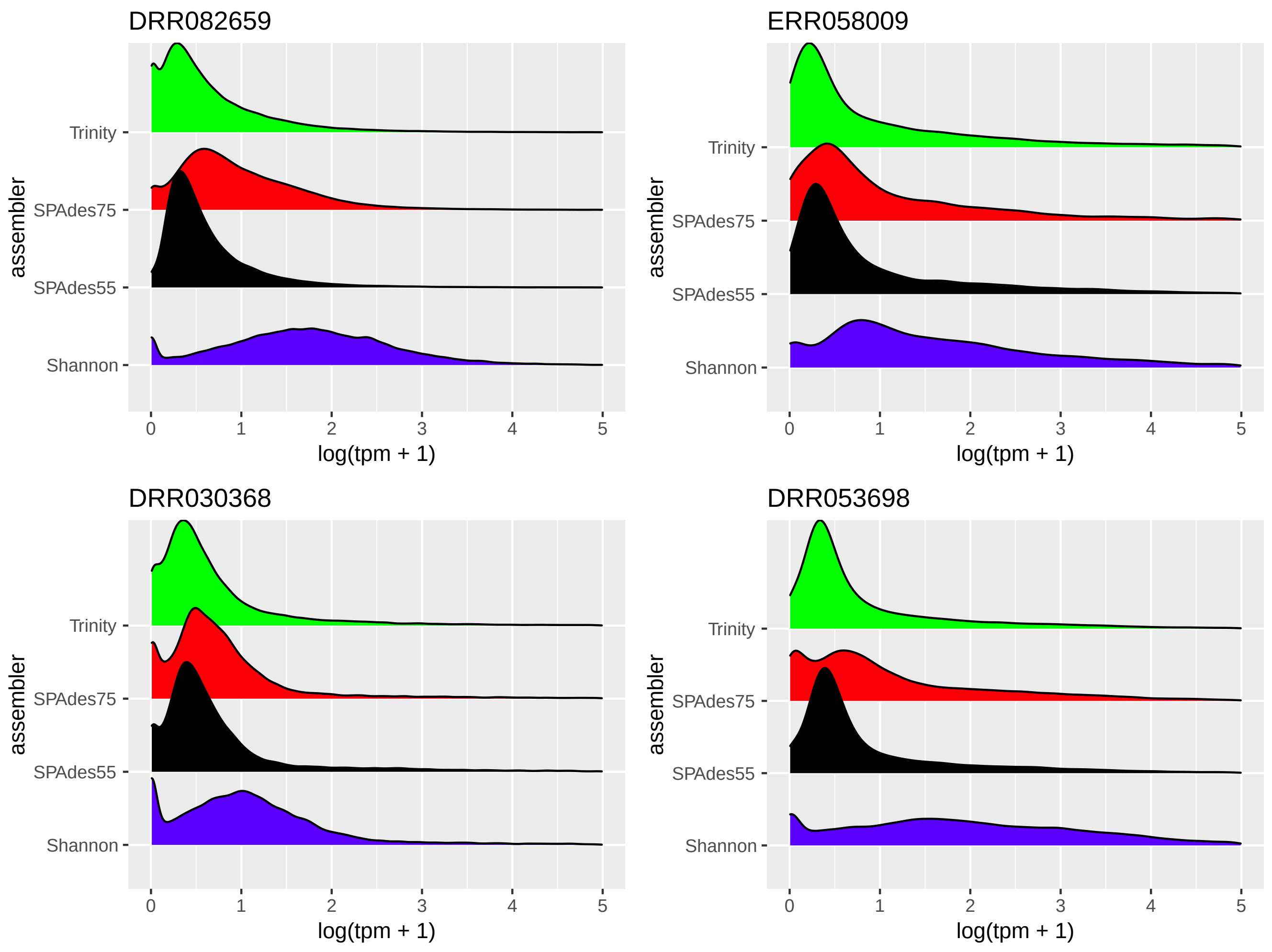

Figure 3 depicts the distribution of gene expression (log(TPM+1)), broke down by individual assembly, for four representative ORP merged final assemblies. As predicted, the use of a higher kmer value with the `SPAdes` assembler resulted in biasing reconstruction towards more highly expressed transcripts. Interestingly, `Shannon` uniquely exhibits a strong bias towards the reconstruction of high-expression transcripts (or away from low-abundance transcripts).

Lastly, though the same read data were assembled, each assembler reconstructed unique transcripts. Using the dataset DRR069093 as an example, across the four different assemblies, a sum of 276852 SwissProt entries were matched. Of these 86% were recovered in all four assemblies. The `SPAdes` assembly using a kmer value of 55 recovered 96% of all transcripts, while the `SPAdes` assembly using a kmer value of 75 recovered 93%. The `Trinity` assembly recovered 96% of the transcripts, while `Shannon` recovered 90%. Depicted in Figure 4, the `SPAdes` assembly using a kmer value of 55 recovered 3749 unique transcripts, `Trinity` recovered 3055, `Shannon` recovered 2526, and `SPAdes` assembly using a kmer value of 75 recovered

775.

Figure 4

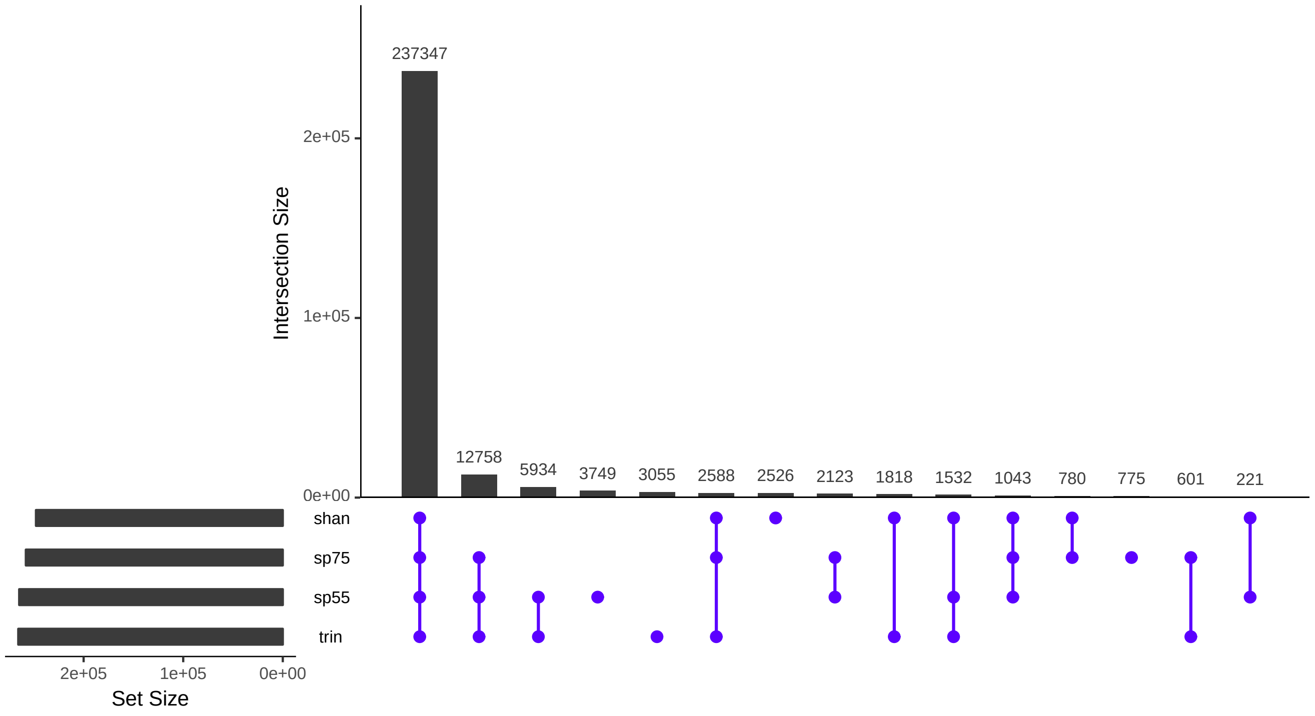

Figure 4 depicts the overlap in identified transcripts between the various assemblies for one representative (DRR069093) dataset.

4 Discussion

For non-model organisms lacking reference genomic resources, the error correction, adapter and quality trimmed reads should be assembled *de novo* into transcripts. While the assembly package `Trinity` (13) is thought to currently be the most accurate stand-alone assembler (17), this study demonstrates that a merged assembly with multiple assemblers (and kmer lengths) results in the highest quality assembly. Specifically, the Oyster River Protocol, which contains a recipe for read error correction, quality trimming, assembly with multiple software packages and merging, resulted in a final assembly, the structure of which was greatly improved.

`TransRate` scores were significantly improved by using the Oyster River Protocol for transcriptome assembly. One metric in particular, the read mapping metric, was vastly improved (Figure 1A). The aspect of quality that this metric assays is critical - specifically measuring how representative of the reads the assembly is. If we assume that the vast majority of generated reads come from the biological sample under study, when reads fail to map, that fraction of the biology is lost. Troublesome, this biology is lost from all downstream analysis and inference. This study conclusively demonstrates that across a wide variety of taxa, assembling with `Trinity` alone may result in a substantial decrease in mapping rate and in turn, the lost ability to draw conclusions from that fraction of the sample.

In contrast to `TransRate` scores, the `BUSCO` metrics were essentially unchanged by assembly with the Oyster River Protocol. The recovery of complete orthologs, the proportion of orthologs reconstructed in fragmented form, and missing orthologs were stable across both assembly methods. The number of orthologs recovered in duplicate (>1 copy), was decreased when using the ORP. Here, we hypothesize that the relative frequency of transcript duplication may have important implications for downstream abundance estimation, with less duplication potentially resulting in more accurate estimation. While gene expression quantitation software (27; 28) probabilistically assigns reads to transcripts in an attempt at mitigating this issue, while not evaluated as part of this work, a primary solution related to decreasing artificial transcript duplication could offer significant advantages.

4.1 Each Assembler Recovers Different Transcripts

The main benefit of the Oyster River Protocol is related to the fact that assemblies are constructed four different ways, using three different assemblers (`Trinity`, `Shannon`, `SPAdes`) and three different values for kmer length (k=25,55,75). As described above, each assembler carries with it a set of heuristics, and these heuristics translate into differential recovery of distinct fractions of the transcript community. Figure 3 depicts this process. Looking at the distribution of gene expression, within the `SPAdes` assemblies, kmer length influences the recovery of transcripts, with longer kmers shifting the distribution to more highly expressed transcripts. Interestingly, `Shannon` seems to have a very different set of expression-based biases, demonstrating an apparent bias against low-abundance transcripts. Trinity exhibits a typical distribution, similar to the `SPAdes` assembler using a shorter value for kmer length.

Taken together, these expression profiles suggest a mechanism by which the ORP outperforms, `Trinity`, and presumably other single-assembler assemblies. While there is substantial overlap in transcript recovery, each assembler recovers unique transcripts (Figure 5), based on expression (and potentially other properties), which when merged together into a final assembly, increases the completeness

4.2 Does Taxonomy Influence Assembly Quality?

Because I was interested in designing a study with broad applicability, I chose read datasets that represented a variety of Eukaryotic groups. Although not originally designed for this purpose, this decision may allow me to understand the influence that intrinsic properties of transcription and transcriptome complexity in different taxonomic groupings may have on assembly. Figure 5 depicts several previously described assembly metrics, broken down by assembly method and by taxonomic group. Given the small sample (n=4 vertebrate, n=5 plant, n=6 invertebrate), it is impossible to draw strong conclusions, but generally, both `Trinity` and the Oyster River Protocol perform equally well across groups. Invertebrate assembly seems to be the most variable in resultant quality, though this may be driven by low sample size coupled with the specific (potentially low quality) datasets chosen at random.

Figure 5

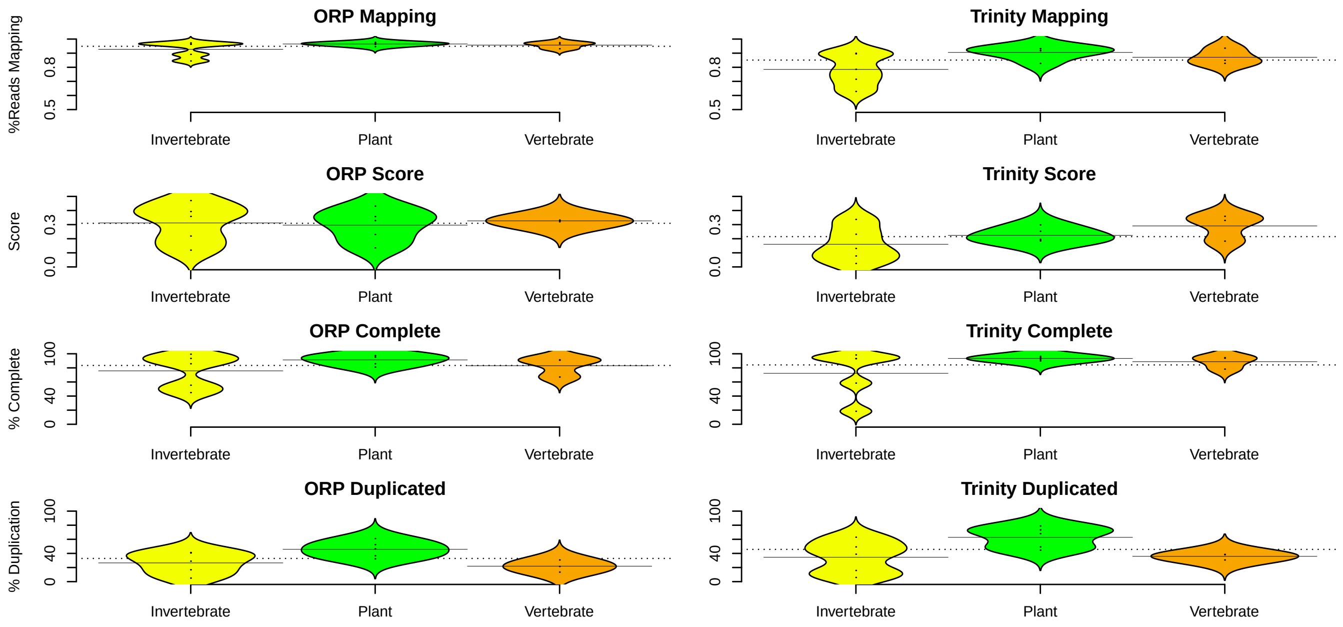

Figure 5 depicts a subset of assembly metrics broken down by taxonomic group. In general, no consistent patterns are observed, suggesting that the described assembly protocol performs well across eukaryotic groups.

4.3 Does Read Depth Influence Quality?

This study included read datasets of a variety of sizes. Because of this, I was interested in understanding if the number of reads used in assembly was strongly related to the quality of the resultant assembly. Conclusively, this study demonstrates that between 30 million paired-end reads and 200 million paired-end reads, no strong patterns in quality are evident (Figure 6). This finding is in line with previous work, (29) suggesting that assembly metrics plateau at between 20M and 40M read pairs, with sequencing beyond this level resulting in minimal gain in performance.

Figure 6

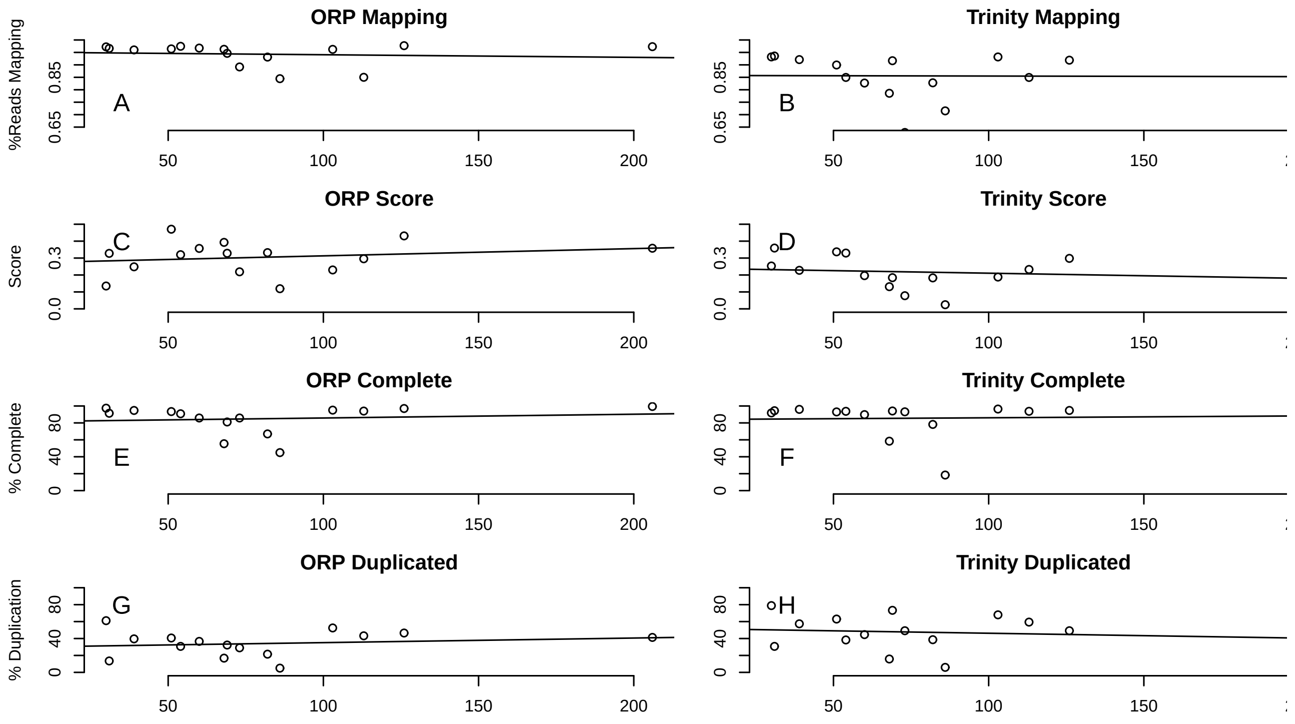

Figure 6 depicts the relationship between the a subset of assembly metrics and the number of read pairs. There is no significant relationship. In all cases the x-axis is millions of paired-end reads.

5 Conclusions

For non-model organisms lacking reference genomic resources, the error correction, adapter and quality trimmed reads should be assembled *de novo* into transcripts. While the assembly package `Trinity` (13) is thought to currently be the most accurate stand-alone assembler (17), a merged assembly with multiple assemblers results in the highest quality assembly. Specifically, use of the Oyster River Protocol, which contains a recipe for read error correction, quality trimming, assembly with multiple software packages, and merging resulted in a final assembly, the structure of which was greatly improved.

Specifically, the improvements in assembly metrics described here are attributed to the multi-way approach, where three different assemblers and three different kmer lengths were used. This approach allows the strengths of one approach to effectively complement the weaknesses of another, thereby resulting in a more complete assembly than otherwise possible. These enhancements are important, as unassembled transcripts are invisible to all downstream analysis.

Acknowledgments

This work was significantly improved by discussions with Richard Smith-Unna, Brian Haas and many others. More generally, the work and it's presentation has been influenced by supporters of the Open Access and Open Science movements.

References

1. Fitzpatrick M, Ben-Shahar Y, Vet L, Smid H, Robinson GE, et al. (2005) Candidate genes for behavioural ecology. Trends In Ecology & Evolution 20: 96–104.

2. Panhuis TM (2006) Molecular evolution and population genetic analysis of candidate female reproductive genes in *Drosophila*. Genetics 173: 2039–2047.

3. Wolf JBW (2013) Principles of transcriptome analysis and gene expression quantification: an RNA-seq tutorial. Molecular Ecology Resources 13: 559–572.

4. Vijay N, Poelstra JW, Künstner A, Wolf JBW (2013) Challenges and strategies in transcriptome assembly and differential gene expression quantification. A comprehensive *in silico* assessment of RNA-seq experiments. Molecular Ecology 22: 620–634.

5. Robinson MD, McCarthy DJ, Smyth GK (2010) edgeR: a Bioconductor package for differential expression analysis of digital gene expression data. Bioinformatics 26: 139–140.

6. Love MI, Huber W, anders S (2014) Moderated estimation of fold change and dispersion for RNA-seq data with DESeq2. Genome Biology 15: 550.

7. Cox MP, Peterson DA, Biggs PJ (2010) SolexaQA: At-a-glance quality assessment of Illumina second-generation sequencing data. BMC Bioinformatics 11: 485.

8. Jiang H, Lei R, Ding SW, Zhu S (2014) Skewer: a fast and accurate adapter trimmer for next-generation sequencing paired-end reads. BMC Bioinformatics 15: 182.

9. Bolger AM, Lohse M, Usadel B (2014) Trimmomatic: a flexible trimmer for Illumina sequence data. Bioinformatics 30: btu170–2120.

10. Pell J, Hintze A, Canino-Koning R, Howe A, Tiedje JM, et al. (2012) Scaling metagenome sequence assembly with probabilistic *de Bruijn* graphs. Proceedings of the National Academy of Sciences 109: 13272–13277.

11. Le HS, Schulz MH, McCauley BM, Hinman VF, Bar-Joseph Z (2013) Probabilistic error correction for RNA sequencing. Nucleic Acids Research 41: 1–11.

12. Song L, Florea L (2015) Rcorrector: efficient and accurate error correction for Illumina RNA-seq reads. GigaScience 4: 48.

13. Haas BJ, Papanicolaou A, Yassour M, Grabherr M, Blood PD, et al. (2013) *De novo* transcript sequence reconstruction from RNA-seq using the Trinity platform for reference generation and analysis. Nature Protocols 8: 1494–1512.

14. Xie Y, Wu G, Tang J, Luo R, Patterson J, et al. (2014) SOAPdenovo-Trans: *de novo* transcriptome assembly with short RNA-Seq reads. Bioinformatics 30: 1660–1666.

15. Smith-Unna R, Boursnell C, Patro R, Hibberd JM, Kelly S (2016) TransRate: reference-free quality assessment of *de novo* transcriptome assemblies. Genome Research 26: 1134–1144.

16. Simão FA, Waterhouse RM, Ioannidis P, Kriventseva EV, Zdobnov EM (2015) BUSCO: assessing genome assembly and annotation completeness with single-copy orthologs. Bioinformatics 31: 3210–3212.

17. Li B, Fillmore N, Bai Y, Collins M, Thomson JA, et al. (2014) Evaluation of *de novo* transcriptome assemblies from RNA-Seq data. Genome Biology 15: 663–21.

18. MacManes MD (2014) On the optimal trimming of high-throughput mRNA sequence data. Frontiers in Genetics 5.

19. MacManes MD, Eisen MB (2013) Improving transcriptome assembly through error correction of high-throughput sequence reads. PeerJ 1: e113.

20. Chikhi R, Medvedev P, Medvedev P (2013) Informed and automated k-mer size selection for genome assembly. Bioinformatics .

21. Kannan S, Hui J, Mazooji K, Pachter L, Tse D (2016) Shannon: An Information-Optimal de Novo RNA-Seq Assembler. bioRxiv .

22. Emms DM, Kelly S (2015) OrthoFinder: solving fundamental biases in whole genome comparisons dramatically improves orthogroup inference accuracy. Genome Biology 16: 157.

23. Camacho C, Coulouris G, Avagyan V, Ma N, Papadopoulos J, et al. (2009) BLAST+: architecture and applications. BMC Bioinformatics 10: 421.

24. R Core Development Team F (2011) R: A Language and Environment for Statistical Computing .

25. Kampstra P (2008) Beanplot: A boxplot alternative for visual comparison of distributions .

26. Conway JR, Lex A, Gehlenborg N (2017) UpSetR: An R Package for the Visualization of Intersecting Sets and their Properties. Bioinformatics .

27. Patro R, Duggal G, Love MI, Irizarry RA, Kingsford C (2017) Salmon provides fast and bias-aware quantification of transcript expression. Nature Methods 14: 417–419.

28. Bray NL, Pimentel H, Melsted P, Pachter L (2016) Near-optimal probabilistic RNA-seq quantification. Nature Biotechnology 34: 525–527.

29. MacManes MD (2015) An opinionated guide to the proper care and feeding of your transcriptome. biorxivorg : 1–23.