Yanagi, Kallisto and Kevlar: Differential Expression Methods

Walter Báez Laguna Humberto Ortiz-Zuazaga

May 16, 2019

In Bioinformatics, RNA transcript abundance estimation from RNA-Seq data seeks to describe the effects of genetic or environmental changes on gene expression. Over the years, differential expression analysis, either with genes or transcripts, is used to find the set of genes or transcripts with different expression levels between X conditions. After a few decades of development, transcript level analysis still faces major challenges that makes the problem of transcript expression quantification quite challenging [B. Hagen, 2001]. Different approaches, alignment pipelines and techniques like RapMap, Yanagi, Kevlar, Kallisto, Salmon have been emerging to reduce the time and difficulty of RNA transcript estimation for differential expression while maintaining comparable, or superior, accuracy with older methods. In this literary review we will be discussing three different modern tools for differential expression, Yanagi, Kallisto and Kevlar. We will explain how each one works and probable pros and cons of each one.

To begin, we will discuss first about Yanagi. Yanagi constructs a set of segments that can be used in the read-alignment-quantification step instead of the whole transcriptome without loss of information. Builds a segment library from a reference transcriptome based on the possible splicing variations. These segment libraries are amenable for usage with lightweight pseudo-alignment methods for segment quantification. Yanagi thrives to illustrate the utility of the segmentation approach using the differential analysis of exon, any part of a gene that will encode a part of the final mature RNA, skipping events across samples from two conditions of interest. Exons that are shorter than parameter L will make junction segments miss reads that span more than two of such short exons. Let's say we have exons E1, E2, and E3, the two exons E2 and E3 of width k are both shorter than L, no segments will capture a read that span E1, E2, and E3 for instance. The tool also shows that exonic regions of a messenger RNA precursor can be combined differently through alternative splicing to form distinct isoforms. The overview of the algorithm behind Yanagi is quite complex. It starts with a given the transcriptome annotation (GTF format file) and the transcript sequences (FASTA format files) as input, Yanagi generates the set of segments and its sequences (as a FASTA file) as the output of the segmentation process. The process can be summarized into three steps: (1) Preprocessing the transcriptome annotation in order to obtain disjoint transcriptome regions, (2) Constructing a Segments Graph (SgG), and finally (3) Generating the segment library. [Gunady et al., 2018]

In the first step, the tool applies a Preprocessing step to eliminate exon overlaps present in the transcriptome reference from events involving alternative 3'/5' splice sites, or transcription start/end sites. This step ensures that any splicing event is at the beginning or the end of a genomic segment, which makes the process of generating L-disjoint and max covering segments easier, is independent from the parameter L, so it can be done only once per transcriptome reference. It is implemented based on the GenomicRanges package in R, specifically the disjoint function. In the next step, Yanagi builds a separate segment graph for each gene, since there are no alternative splicing events between transcripts of different genes. However, future work may use segment graphs that connect different genes sharing regions of identical sequence length L or greater. A segment graph G is an acyclic directed graph defined by the pair (N, E), where N is a set of nodes representing segments, and E is the set of directed edges between the nodes. The tool has a very important parameter, the L value. It is the only parameter required by the segmentation algorithm. To understand the impact of the L parameter on the generated segments library, it is

important to mention that the choice of L is based on the expected read length of the sequencing experiment, using small values of L tends to shred the transcriptome while high values ends up generating segments such that each segment represents a whole transcript. [Gunady et al., 2018]

Moving to our next modern differential expression tool, Kallisto uses RNA-Seq files to estimate isoforms based on k-mer abundance (sequences of k bases). One of the most noticeable traits of the tool is the improvement in speed and memory usage when it is compared to older or similar providing similar accuracy. Most of the previous methods for differential expression were based on mapping RNA-Seq sequenced reads to a reference genome, assigning positions to the reads with the genome and counting the number of overlapping reads. Kallisto on the other hand, uses pseudoalignment with the transcriptome sequence instead of the whole genome to potentially identify the original transcript for each read instead of identifying the position of the read in it. The first step before initializing the analysis is to build an index, a de Bruijin Graph from the k-mers of a given transcriptome. The nodes in the graph correspond to a k-mer, retains the information of the transcripts bases on a color, lineal stretches with the same color are contigs or transcripts. After the graph is built, the tool stores the mapping for each k-mer with the position of it in the corresponding transcripts of origin. Kallisto then decomposes the reads into k-mers and finds a path in the de Bruijin Graph which generates compatibility classes for each k-mer. While skipping redundant k-mers, since this usually means adjacent k-mers belong to the same transcript, the pipeline finds the potential transcripts of origin for a read based on the intersections of the k-mers compatibility classes previously mention. Afterwards, Kallisto optimizes the RNA-Seq likelihood function with the Expectation Maximization Algorithm. The algorithm will maximize the equivalent RNA-Seq probability trying different values of probabilities for each transcript in order to get the highest one. In simple words, Kallisto uses ECs to derive a count statistic for each EC. Transcripts compatibility counts (TCCs) are produced per each EC, representing the number of fragments that appeared compatible with the corresponding set of transcripts during the pseudo-alignment step. In the final step, TCCs are used to perform gene-level differential analysis by skipping the quantification step using logistic regression. [Bray et al., 2016]

The last tool to be discussed is Kevlar. The most recent of the three, Kevlar is use to develop reference-free variant discovery methods for genomics. The project started with a focus of de novo germline variants in simplex pedigrees and the developers are working on supporting wider range experiments and case/control studies. Kevlar is a method used to identify variants spanning reads or to assemble these reads into contigs without the use of a reference genome. As the progress of the methods continues, it is still dependable of a reference genome for making the final variant calls by aligning the contigs to a small cutout of the given genome. In order to use this method it is advice to dedicate a virtual environment using venv or conda and it also requires a Python 3 package and several dependencies that are not in the standard libraries like pysam module, pandas, scipy, khmer, etc. The source code is open source and anyone can clone it from Github to contribute to the development or work around. Kevlar overview consists of the count command which is used to count the k-mers for each sample as well as for the reference genome and the mask. The novel command uses pre-computed k-mers counts to find reads containing novel k-mers using the specified thresholds. Following the last command the filter recomputes k-mers and filters out the ones with insufficient abundance or from contaminated sources, reads without interesting k-mers are discarded. Reads spanning the same variant usually have k-mers worth paying attention and with partition reads command the reads can be grouped based on the shared novel k-mers. Contig Assembly takes each partition from the previous command and assembles them into contigs for variant annotation. Subsequently, the localize command identifies the appropriate targets in the given reference genome in order to align each contig for variant annotation. The call command then computes a full dynamic programming alignment for each contig to the corresponding reference target and calls variants based on the alignment path. Finally the simlike command computes the likelihood score for each variant prediction and ranks them based on the score. [Standage et al., 2019]

There is a comparison between Yanagi and Kallisto stated by [Gunady et al., 2018] in their paper that talks about both segment-based and TCC-based approaches. Both methods successfully avoid the quantification step when targeting gene-level analysis. This is an advantage in efficiency, speed, simplicity, and accuracy, as previously discussed. One difference between the methods is that segment-based approach is agnostic to the alignment technique used, while TCC-based approach

is a Kallisto specific approach. Since segments are formed to preserve the genomic location and splicing structure of genes, SCs can be directly mapped and interpreted with respect to the genome coordinates. ECs do not have a direct biological meaning in this sense, all k-mers that belong to the same transcript yet originated from different locations over the genome will all fall under the same EC, making TCCs less interpretive, while on the contrary, with SCs these k-mers will appear in different segments depending on the transcriptome structure using Yanagi, crucial for a biologist who analyzes the outcome of the differential analysis. As for Kevlar, not much information was gathered since it's still very recent and under development. A good source material for a paper would be a comparison of this method which does not need a reference genome until the final steps to Kallisto and Yanagi in order to compare accuracy, efficiency and running time.

References

[B. Hagen, 2001] B. Hagen, J. (2001). The origins of bioinformatics. *Nature reviews. Genetics*, 1:231–6.

[Bray et al., 2016] Bray, N. L., Pimentel, H., Melsted, P., and Pachter, L. (2016). Near-optimal probabilistic rna-seq quantification. *Nature biotechnology*, 34(5):525.

[Gunady et al., 2018] Gunady, M. K., Mount, S. M., and Bravo, H. C. (2018). Fast and interpretable alternative splicing and differential gene-level expression analysis using transcriptome segmentation with yanagi. *bioRxiv*.

[Standage et al., 2019] Standage, D. S., Brown, C. T., and Hormozdiari, F. (2019). Kevlar: a mapping-free framework for accurate discovery of de novo variants. *bioRxiv*.