iQuant: A fast yet accurate GUI tool for transcript quantification

Tin Chi Nguyen[1], Nan Deng[1], Guorong Xu[2], Zhansheng Duan[2] and Dongxiao Zhu [1,*]
[1]*Department of Computer Science, Wayne State University, Detroit, MI 48202; dzhu@wayne.edu*
[2]*Department of Computer Science, University of New Orleans, New Orleans, LA 70148*
** To whom correspondence should be addressed.*

***Abstract*—Transcript quantification using RNA-seq is central to contemporary and future transcriptomics research. The existing tools are useful but have much room for improvement. We present a new statistical model, a fast yet accurate transcript quantification algorithm. Our tool takes RNA-seq reads in fasta or fastq format as input and output transcript abundance through a few mouse clicks. Our method compares favorably with the existing GUI tools in terms of both time complexity and accuracy. Availability: Both simulation data used for method comparisons and the GUI tool are freely available at http://asammate.sourceforge.net/.**

***Keywords*-Transcriptome quantification; alternative splicing; RNA-seq; GUI; constrained least square;**

I. Introduction

Transcript quantification using RNA-seq is central to a wide range of transcriptomics research. The problem itself is challenging due to the fact that the observed exonic expression signal can be aggregated from a set of sibling transcripts encoded by the same gene with diverse alternative splicing mechanisms. In essence, the problem finds its root in latent variable models where we infer the latent variables (transcript expression) from the observed variables (exonic expression).

Several computational approaches have been developed to utilize high throughput gene expression profiling data collected from microarray and RNA-seq experiments. In earlier studies, an iterative Expectation-Maximization (EM) type of algorithm using Expression Sequence Tags (ESTs) ([1]) and a Nonnegative Matrix Decomposition (NMF) based algorithm ([2]) using exon and exon-exon junction microarrays were developed to solve this problem. Both approaches belong to latent variable model family and they represent some of the more pioneering efforts to tackle the transcript quantification problem. However, both iterative approaches suffer from non-unique solutions that are sensitive to initialization. Moreover, the performance was also limited by data quantity (ESTs) and quality (microarrays).

RNA-seq technology gives base-level exonic expression signal with unprecedented dynamic range and sensitivity. Both data quantity and quality for solving this problem are substantially improved. Despite a number of approaches to transcript quantification ([3]–[8]), few attempt has been made towards solving the problem using base-level signal. Bohnert and Ratsch ([9]) developed a quadratic programming based web tool (rQuant.web) to exploit base-level expression signal for transcript quantification, but significant advantages of pursuing this direction are yet to be demonstrated. In addition, this algorithm does not exploit the increasing availability of multiple samples (replicates), which can be used to reduce the effect of random variability.

In this paper, we present a new statistical model, a fast yet accurate transcript quantification algorithm using base-level signal and a user-friendly tool with a GUI. Using real-world simulation studies, we compared our tool with three other GUI tools in terms of both time complexity and accuracy.

II. Methods

We propose a new model to explain how the observed base-level RNA-seq expression signal (observed read covereage) is aggregated from a mixture of sibling transcripts. We first introduce the model for a single sample, then we extend the model to process multiple samples.

Given a set of short reads of a sample, we first align the reads to the reference genome, then we estimate the abundance of annotated transcripts per gene locus. We denote the number of exonic positions by m. The number of annotated transcripts is denoted by n. The vector of observed read coverage is denoted by $\underline{\mathbf{e}}$ = $[e_1, e_2, ..., e_m]^T$ where e_i is the observed read coverage at the i^{th} exonic position. The gene-level expression abundance is denoted by r. The gene expression abundance can be estimated either using read counts (e.g. RPKM) or average base-level exonic signal over the shared exonic regions among all the sibling transcripts. The vector of transcript proportions is denoted by $\underline{\mathbf{p}} = [p_1, p_2, ..., p_n]^T$ where p_j is the proportion of j^{th} transcript , $\sum_{j=1}^{n} p_j = 1$ and $0 \leq p_j \leq 1$ for all j. The splicing matrix can be denoted by $\mathbf{S} \ \epsilon \ \{0, 1\}^{m \times n}$. $\mathbf{S}$ is a matrix of 0's and 1's, each row i represents a single base and each column j represents a sibling transcript. S_{ij}=1 indicates that the j^{th} sibling transcript contributes to the exonic signal observed at i^{th} base, S_{ij}=0 otherwise.

An example is illustrated by Figure 1. In this example, the 1^{st} transcript skips no exon, and thus the all the elements of the 1^{st} column of $\mathbf{S}$ take the value of 1 (S_{j1} = 1 for all $j \epsilon [1..m]$). In the 2^{nd} column of $\mathbf{S}$, the elements corresponding to the 2^{nd} exon take the value of 0 because the 2^{nd} transcript skips the 2^{nd} exon. In the 3^{rd} column of

978-1-4577-1613-3/11/$26.00 ©2011 IEEE

$\mathbf{S}$, the elements corresponding to the 3^{rd} exon take the value of 0 because the 3^{rd} transcript skips the 3^{rd} exon.

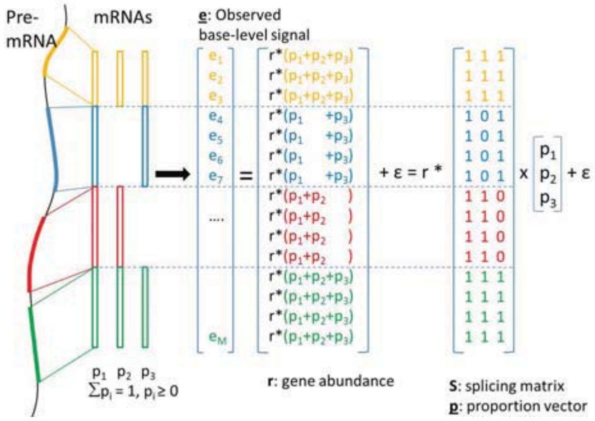

Figure 1. Transcript abundance estimation using observed read coverage

At each exonic position i, the expected read coverage is $r\sum_{j=1}^{n} S_{ij}p_j$. Our goal is to minimize the difference between the observed read coverage and the the expected read coverage. We can write the observed read coverage as a sum of the expected coverage and error vector (ε) as following:

$$\begin{bmatrix} e_1 \\ e_2 \\ e_3 \\ ... \\ e_m \end{bmatrix} = \begin{bmatrix} r(S_{11}p_1 + ... + S_{1n}p_n) \\ r(S_{21}p_1 + ... + S_{2n}p_n) \\ r(S_{31}p_1 + ... + S_{3n}p_n) \\ ... \\ r(S_{m1}p_1 + S... + S_{mn}p_n) \end{bmatrix} + \varepsilon$$

$$= r \begin{bmatrix} (S_{11} + ... + S_{1n}) \\ (S_{21} + ... + S_{2n}) \\ (S_{31} + ... + S_{3n}) \\ ... \\ (S_{m1} + ... + S_{mn}) \end{bmatrix} \times \begin{bmatrix} (p_1) \\ (p_2) \\ ... \\ (p_n) \end{bmatrix} + \varepsilon$$

So we have the following equation:

$$\underline{\mathbf{e}} = (r\mathbf{S}) \times \underline{\mathbf{p}} + \varepsilon. \qquad (1)$$

For each gene locus, the relationship between the observed base-level coverage and the latent transcript proportion can be modeled as in equation (1), where ε is the estimation error. In other words, our algorithm solves the following constrained linear least square problem:

$$\begin{cases} \min_{\underline{\mathbf{p}}} \|\underline{\mathbf{e}} - (r\mathbf{S}) \times \underline{\mathbf{p}}\|^2 \\ \underline{\mathbf{1}}^T \times \underline{\mathbf{p}} = 1 \\ 0 \leq \underline{\mathbf{p}} \leq 1 \end{cases} \qquad (2)$$

where $\underline{\mathbf{1}}^T = \underbrace{[1, 1, 1, ...1]}_{\text{n}}$

In case of multiple samples, the same logic can be applied to estimate the transcript proportions. For each gene locus, each sample (replication) has different expression abundances but share the same transcript proportion vector. Let k be the number of the samples. After proper normalization, we have k vectors of observed read coverage: $\underline{\mathbf{e}}_1 = [e_{11}, e_{12}, ..., e_{1m}]^T, \underline{\mathbf{e}}_2 = [e_{21}, e_{22}, ..., e_{2m}]^T, ..., \underline{\mathbf{e}}_k = [e_{k1}, e_{k2}, ..., e_{km}]^T$. The gene expression abundances are denoted by $r_1, r_1, ..., r_k$. We have k equations:

$$\begin{cases} \underline{\mathbf{e}}_1 = (r_1\mathbf{S}) \times \underline{\mathbf{p}} + \varepsilon_1 \\ \underline{\mathbf{e}}_2 = (r_2\mathbf{S}) \times \underline{\mathbf{p}} + \varepsilon_2 \\ \underline{\mathbf{e}}_3 = (r_3\mathbf{S}) \times \underline{\mathbf{p}} + \varepsilon_3 \\ ... \\ \underline{\mathbf{e}}_k = (r_k\mathbf{S}) \times \underline{\mathbf{p}} + \varepsilon_k \end{cases}$$

or

$$\begin{bmatrix} \underline{\mathbf{e}}_1 \\ \underline{\mathbf{e}}_2 \\ \underline{\mathbf{e}}_3 \\ ... \\ \underline{\mathbf{e}}_k \end{bmatrix} = \begin{bmatrix} r_1\mathbf{S} \times \underline{\mathbf{p}} \\ r_2\mathbf{S} \times \underline{\mathbf{p}} \\ r_3\mathbf{S} \times \underline{\mathbf{p}} \\ ... \\ r_k\mathbf{S} \times \underline{\mathbf{p}} \end{bmatrix} + \varepsilon$$

where ε is the error we want to minimize. The equation above can be rewritten as:

$$\begin{bmatrix} \underline{\mathbf{e}}_1 \\ \underline{\mathbf{e}}_2 \\ \underline{\mathbf{e}}_3 \\ ... \\ \underline{\mathbf{e}}_k \end{bmatrix} = \begin{bmatrix} r_1\mathbf{S} \\ r_2\mathbf{S} \\ r_3\mathbf{S} \\ ... \\ r_k\mathbf{S} \end{bmatrix} \times \underline{\mathbf{p}} + \varepsilon \qquad (3)$$

If we denote $\underline{\mathbf{y}} = [\underline{\mathbf{e}}_1^T, \underline{\mathbf{e}}_2^T, \underline{\mathbf{e}}_3^T, ..., \underline{\mathbf{e}}_k^T]^T$ and $\mathbf{W} = [(r_1\mathbf{S}^T), (r_2\mathbf{S})^T, (r_3\mathbf{S})^T, ..., (r_k\mathbf{S})^T)]^T$, the transcript proportion can be estimated by solving the following constrained linear least square:

$$\begin{cases} \min_{\underline{\mathbf{p}}} \|\underline{\mathbf{y}} - \mathbf{W} \times \underline{\mathbf{p}}\|^2 \\ \underline{\mathbf{1}}^T \times \underline{\mathbf{p}} = 1 \\ 0 \leq \underline{\mathbf{p}} \leq 1 \end{cases} \qquad (4)$$

where $\underline{\mathbf{1}}^T = \underbrace{[1, 1, 1, ...1]}_{\text{n}}$

In fact, the optimisation problem described in (4) is the generalized form of (2), since it formulates the optimisation problem for one or multiple samples. This is a classic medium-scale convex quadratic programming, which can be solved easily by the active set method ([10]) .

III. Results

Using real-world simulation studies, we demonstrate the accuracy and speed of iQuant by comparing with RAEM ([8], implemented in aSAMMate suite) and two widely used GUI tools, Cufflinks ([6]) and rQuant ([9]). We used FluxSimulator to simulate the whole transcriptome sequencing experiments with the Illumina Genome Analyzer.

15 million and 30 million single end reads with length of 50 and 100 were simulated using Ensembl database, including around 100,000 annotated human transcript structures. To account for the read errors in real-world data, we estimated an error model from realworld RNA-seq data sets and provided as an input for FluxSimulator to simulate reads with errors([11]).

A. *Time complexity*

Table I shows the time complexity on the same hardware platform (MacPro, two Intel Xeon DualCore 2.66GHz, 4GB RAM). It is clear from Table I that iQuant is much faster than its competitors.

Table I
COMPARISON OF RUNNING TIME OF THE FOUR GUI TOOLS

	RAEM	iQuant	Cufflinks	rQuant.web
15 millions	3377s	820s	2060s	> 1d
30 millions	6610s	1036s	2795s	> 1s

B. *Accuracy of isoform quantification*

We proceed to compare the accuracy of isoform quantification for the four GUI tools. An ultimate goal for transcriptome quantification is to estimate sibling isoform abundance proportions, summing up to one for each gene. From the ground truth where we simulated RNA-seq reads using FluxSimulator, we know the true copy numbers of all the isoforms. Thus the more accurate isoform quantification method will give the vector of isoform proportion least divergent from the ground truth. We used Jensen-Shannon (JS) divergence to capture both linear and nonlinear relationships. Values closer to zero indicate a better performance.

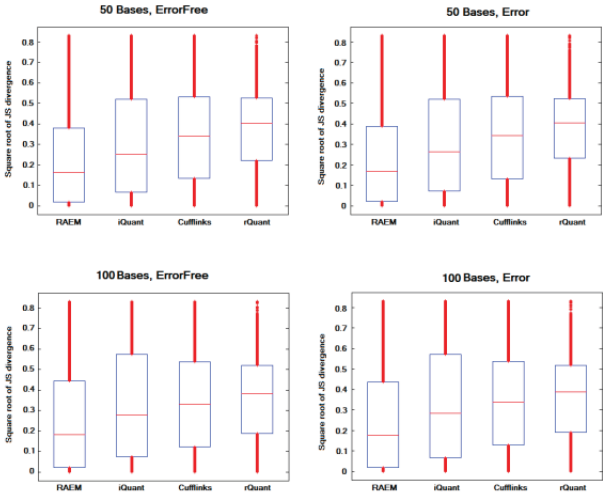

Figure 2. Comparison of accuracy using J-S divergence.

From Table I and Figure 2, iQuant demonstrates an impressive speed without compromise of accuracy.

IV. CONCLUSION

In this paper, we have presented a way to accommodate multiple samples and thus give rise to a better estimate. Beside accuracy, our GUI tool also demonstrates an impressive speed. It is particularly important prospectively with the faster pace of increase in both sequencing depth and sample size in the near future.

REFERENCES

[1] Y. Xing, T. Yu, Y. N. Wu, M. Roy, J. Kim, and C. Lee, “An expectation-maximization algorithm for probabilistic reconstruction of full-length isoforms from splice graphs,” *Nucleic Acids Research*, vol. 34, no. 10, pp. 3150–3160, 2006.

[2] M. A. Anton, D. Gorostiaga, E. Guruceaga, V. Segura, P. Carmona-Saez, A. Pascual-Montano, R. Pio, L. M. Montuenga, and A. Rubio, “Space: an algorithm to predict and quantify alternatively spliced isoforms using microarrays genome biology,” *Genome Biology*, vol. 9, no. 2, p. R46, 2008.

[3] H. Jiang and W. H. Wong, “Statistical inferences for isoform expression in rna-seq,” *Bioinformatics*, vol. 25, no. 8, pp. 1026–1032, 2009.

[4] X. Wang, Z. Wu, and X. Zhang, “Isoform abundance inference provides a more accurate estimation of gene expression levels in rna-seq,” *Journal of Bioinformatics and Computational Biology*, vol. 8, no. Suppl 1, pp. 177–192, 2010.

[5] B. Li, V. Ruotti, R. M. Stewart, J. A. Thomson, and C. N. Dewey, “Rna-seq gene expression estimation with read mapping uncertainty,” *Bioinformatics*, vol. 26, no. 4, p. 493500, 2010.

[6] C. Trapnell, B. A. Williams, G. Pertea, A. Mortazavi, G. Kwan, M. J. van Baren, S. L. Salzberg, B. J. Wold, and L. Pachter, “Transcript assembly and quantification by rna-seq reveals unannotated transcripts and isoform switching during cell differentiation,” *Nature Biotechnology*, vol. 28, no. 5, pp. 511–515, 2010.

[7] M. Nicolae, S. Mangul, I. Mandoiu, and A. Zelikovsky, “Estimation of alternative splicing isoform frequencies from rna-seq data,” *Algorithms Mol Biol.*, vol. 6, no. 1, p. 9, 2011.

[8] N. Deng, A. Puetter, K. Zhang, K. Johnson, Z. Zhao, C. Taylor, E. K. Flemington, and D. Zhu, “Isoform-level microrna-155 target prediction using rnaseq,” *Nucleic Acids Research*, vol. 39, no. 9, p. e61, 2011.

[9] R. Bohnert and G. Ratsch, “rquant.web: a tool for rna-seq-based transcript quantitation,” *Nucleic Acids Research*, vol. 38, no. suppl 2, pp. W348–W351, 2010.

[10] J. Nocedal and S. J. Wright, “Numerical optimization,” *Springer*, p. 449, 2006.

[11] G. Xu, N. Deng, Z. Zhao, T. Judeh, E. Flemington, and D. Zhu, “Sammate: a gui tool for processing short read alignments in sam/bam format,” *Source Code Biol Med.*, vol. 6, no. 1, p. 2, 2011.