MDGE610: kallisto assignment

Alex Gao

Feb 2025

Part 1: Conceptual understanding

(1) The EM Algorithm. What is the EM algorithm? Describe how it works in your own words:

a. What objective function does it maximize? The EM algorithm maximizes the observed-data log-likelihood, represented in Dempster et al. (1997) by equation (2.4):

$$L(\phi) = log\ g(y|\phi)$$

$L(\phi)$ is the log-likelihood – a scalar value that measures how well model parameters explain the observed data. Specifically, it is the logarithmic probability of observing data y under the statistical model g with parameters ϕ. The EM algorithm iteratively updates ϕ to increase the log-likelihood until it reaches a maximum.

b. What are the E-step and M-step, conceptually? The E (expectation)- and M (maximization)-steps are repeating phases of the EM algorithm. The challenging part of maximizing the log-likelihood for empirical data is contending with hidden (latent) variables that affect the model but are not directly observed, thus making the data "incomplete." EM repeatedly "completes" the data, allowing iterative maximization:

In the E-step, the algorithm uses the current parameter values to estimate latent variable values. This effectively "completes" the data – in the sense that the latent variables are now "known" – and allows the calculation of a log-likelihood value.

In the M-step, the algorithm treats the estimated latent variable values as known, and calculates new parameter values which increase the log-likelihood. These updated parameters are then used in the next E-step to re-estimate the latent variables.

By iterating between these steps, the log-likelihood increases, or remain the same, at each iteration until convergence.

c. What constitutes the "missing data" in kallisto's application of EM? The missing data (latent variables) are transcript identities – i.e., which transcript actually generated each read. While pseudoalignment identifies the set of transcripts compatible with each read, it does not reveal the exact transcript of origin.

(2) kallisto's Objective Function. Examine the kallisto paper and/or source code to under stand the specific likelihood function being maximized.

a. What is the mathematical form of the objective function? Kallisto maximizes a likelihood function for RNA-seq, represented in Bray et al., (2016) as:

$$L(\alpha) = \prod_{f \in F} \sum_{t \in T} \frac{\alpha_t}{l_t} y_{f,t}$$

Where F is the set of reads (from sequencing output), T is the set of transcripts (from input reference transcriptome), l_t is transcript length, and α_t is transcript abundance. y is a binary indicator for if a transcript is compatible with transcript t – this collapses the equation to:

$$L(\alpha) = \prod_{e \in E} \left(\sum_{t \in e} \frac{\alpha_t}{l_t} \right)^{c_e}$$

Where c_e is counts (c) per equivalence group (e). Equivalence classes group reads that map to exactly the same set of transcripts, so the contribution to the likelihood is computed once per class instead of once per read. This is enabled by pseudoalignment, and greatly speeds up the EM algorithm.

b. How does kallisto's EM implementation maximize it? Kallisto's EM algorithm follows the same conceptual framework discussed in question 1b. The parameters being estimated are transcript abundances (α_t), which contribute to the log-likelihood – a measure of how well these abundances explain the observed reads. The latent variable is the "transcript identity" of each read, defined as the fractional assignment of the read to all compatible transcripts.

In practice, kallisto initializes arbitrary transcript abundance estimates. In the E-step, the algorithm uses the current abundance estimates to determine transcript identities for all reads. In the M-step, these transcript identities are treated as known, and the abundances are updated to maximize the log-likelihood. The updated abundances are then used in the next E-step to recompute transcript identities. This iterative process continues until the log-likelihood converges, at which point the final transcript abundances are reported.

c. What convergence criteria does kallisto use to decide when to stop iterating? Where are these specified in the code? Kallisto stops iterating when the estimated transcript abundances (α_t) are stable between successive EM iterations. Convergence is assessed only for transcripts whose abundances exceed a minimum threshold ($\alpha_t N > 0.01$), effectively ignoring very low-abundance transcripts. For the remaining transcripts, EM terminates when the relative change in abundance between iterations is less than 1%. This ensures that the abundance estimates have stabilized while avoiding unnecessary computation for negligible transcripts. Hard limits for minimum and maximum iterations are also specified to promote early exploration of new solutions, or stop computation if a "stable" solution cannot be reached.

The thresholds are hard-coded and appear to be chosen heuristically. From `kallisto-master/src/EMAlgorithm.h`:

```
    const double alpha_change_limit = 1e-2; // ignore very low abundance transcripts
    const double alpha_change = 1e-2;       // relative change threshold (1%)

    size_t n_iter = 10000                   // hard upper limit for number of iterations
    size_t min_rounds = 50                  // hard lower limit for number of iterations
```

Based on these thresholds, the stopping condition for EM is implemented from line 173 of `EMAlgorithm.h` as:

```
bool stopEM = false;
int chcount = 0;

// Count the number of transcripts whose relative change exceeds the threshold
```

```
for (int ec = 0; ec < num_trans_; ec++) {
    if (next_alpha[ec] > alpha_change_limit &&
        (std::fabs(next_alpha[ec] - alpha_[ec]) / next_alpha[ec]) > alpha_change) {
        chcount++;
    }
}

// If no transcripts exceed the threshold (chcount == 0) and
// the minimum number of EM rounds has passed, stopEM is set to true
if (chcount == 0 && i > min_rounds) {
    stopEM = true;
}
```

d. How might these criteria be justified? These convergence criteria are best justified on practical rather than statistical grounds. In principle, EM can continue making arbitrarily small updates, but beyond a point the gains in accuracy diminish rapidly while runtime continues to increase. For most RNA-seq applications, especially differential expression analysis of moderately to highly expressed genes, a 1% change in abundance estimates is unlikely to meaningfully affect downstream results. Ignoring extremely lowly expressed transcripts further reduces computation without substantially impacting typical analyses, since such transcripts are often filtered out later anyway. Taken together, these choices make kallisto well suited for large datasets and routine differential expression analyses focused on robust signals.

(3) Local vs. Global Maxima. The EM algorithm is guaranteed to find a local maximum, but not necessarily the global maximum. Different starting points could, in principle, lead to different solutions.

a. Examine how kallisto initializes its abundance estimates. Abundance estimates are initialized arbitrarily. From line 38 of `EMAlgorithm.h`:

```
alpha_(num_trans_, 1.0/num_trans_), // uniform distribution over targets
```

Here, the current transcript abundances are stored in the vector `alpha_` and begin as:

$$\alpha_t = \frac{1}{\text{number of transcripts}}$$

This uniform initialization ensures that EM starts without bias toward any transcript. Subsequent iterations then update these estimates based on read assignments.

b. Do you think initialization matters for this problem? Why or why not? Initialization is unlikely to significantly affect kallisto's results for typical RNA-seq data. When most reads map uniquely, the EM algorithm converges to essentially the same maximum regardless of starting values. Conversely, if a majority of reads were ambiguous multi-mappers, different initializations could lead to different local maxima. For standard RNA-seq with 75–150 bp reads, most reads provide clear evidence of transcript origin, so local maxima generally coincide with the global maximum. Low-depth or short-read datasets, however, could make abundance estimates more sensitive to initialization, in which case higher depth or longer reads may be needed during sequencing.

Part 2: Empirical Investigation

Testing Convergence Criteria. You will empirically test how kallisto's EM convergence criteria affect estimation accuracy.

2a. Modifying the code After downloading the kallisto source code, I initialized a git repository and did an initial commit. I then edited files in `kallisto-master/src` to expose the following parameters to the command line:

- `alpha_change`: core convergence check, max relative change in transcript abundances. Default 1e-2 (1%).
- `alpha_change_limit`: abundance threshold for convergence check. Default 1e-2.
- `alpha_limit`: cleanup step, transcripts with abundances below this threshold are zeroed out. Default 1e-7.
- `n_iter`: hard iteration limit. Default 10,000.
- `min_rounds`: minimum number of iterations before calling converged. Default 50.

The first three parameters seem most relevant to quantification sensitivity, while the last two should only have a major effect on runtime.

All changes were tracked with git (see below) and code was successfully compiled in `kallisto-master/build`.

```
commit 67a930b750a2e9486e5842a7bd0c629317cfd534 (HEAD -> master)
Author: Alex <alexg@alex17.localdomain>
Date:   Sun Feb 8 00:46:56 2026 -0700

    redefined alpha_change and alpha_change_limit as separate options

commit 487bd38b69d96571c0b5bfb311abbc554aed3941
Author: Alex <alexg@alex17.localdomain>
Date:   Sun Feb 8 00:04:21 2026 -0700

    exposed convergence parameters to command line

    - added new fields to ProgramOptions in common.h
    - added command line options to ParseOptionsEM in main.cpp
    - changed run() in EMAlgorithm.h to use newly implemented options
    - updated usage help text in usageEM() function of main.cpp

commit 95ff8a2c638910b302d3b49e53d7d493f2d370c2
Author: Alex <alexg@alex17.localdomain>
Date:   Sat Feb 7 23:44:02 2026 -0700
```

2b. Investigation After confirming that the modified program behaved as expected, several considerations emerged for downstream investigation. A natural first option would be a grid-search based optimization, where all parameter combinations are systematically evaluated. However, the complexity of this search would be unmanageable with five tunable parameters and this approach was ultimately deemed to be outside the scope of this report.

Instead, a direct hypothesis-driven approach was adopted, outlined below:

Hypothesis 1. Effect of convergence criteria on accuracy: Decreasing `alpha_change` and `alpha_change_limit` reduces estimation error for low-abundance transcripts by enforcing stricter EM convergence.

Hypothesis 2. Effect of cleanup threshold on accuracy: Increasing `alpha_limit` causes increasingly aggressive cleanup. Increasing this threshold will progressively reduce false positive detection at the expense of true low-abundance transcript sensitivity.

—

Effect of convergence criteria on accuracy Exhaustive kallisto runs were performed across all combinations of `alpha_change` and `alpha_change_limit`, with values sampled logarithmically from 1e-4 (0.01%) to 1e-1 (10%). Although many parameter combinations are effectively redundant, the small dataset and short runtimes made a comprehensive evaluation feasible.

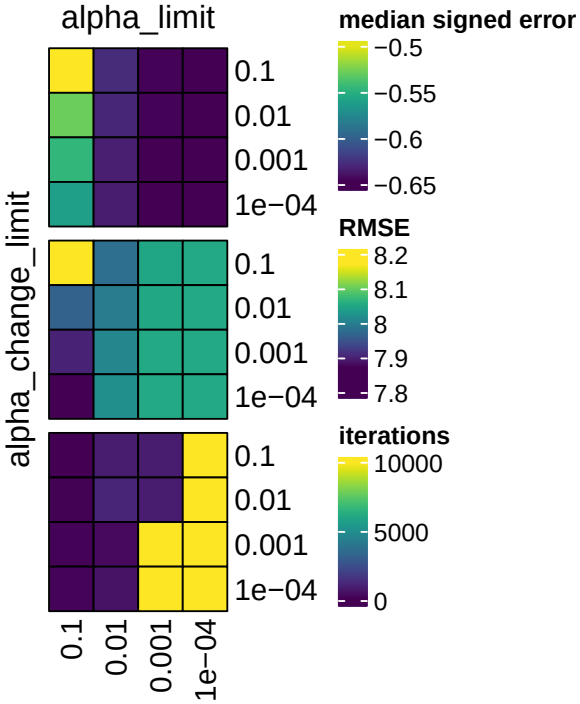
Quantification accuracy was summarized by two complementary metrics: **(i)** median of the signed error ($count_{estimate} - count_{real}$) to capture direction of error, and **(ii)** root mean squared error (RMSE) to measure overall error magnitude. To focus on expressed transcripts only, transcripts with true count == 0 were filtered out. Computational cost was summarized as number of EM iterations for each run.

Across all combinations of `alpha_change` and `alpha_change_limit`, median signed error was consistently negative, indicating systematic underestimation of transcript counts. Importantly, tightening either parameter did not reduce this bias. Instead, the median error became slightly more negative as convergence criteria were made more stringent, plateauing at approximately -0.65 counts (top heatmap). This indicates that stricter convergence does not move estimates closer to the true values, and, if anything, slightly reinforces an existing bias.

RMSE showed a similarly flat pattern across the parameter grid (bottom heatmap). Error magnitude varied modestly and only showed slight decrease when more transcripts were assessed for convergence (`alpha_change_limit` = 1e-4) with weaker criteria (`alpha_limit` = 0.1).

Runtime spiked sharply as convergence criteria were tightened. Given the negligible gains in quantification accuracy, these increases represent a clear trade-off: substantially higher computational cost for minimal or no improvement in accuracy.

Together, these results suggest that kallisto's EM algorithm converges rapidly to a stable solution whose accuracy is primarily constrained by data limitations (*e.g.*, fragment length, sequencing depth) rather than by premature EM termination. Tightening convergence criteria beyond default-like values does not meaningfully improve quantification accuracy for expressed transcripts and yields diminishing returns in accuracy while substantially increasing runtime.



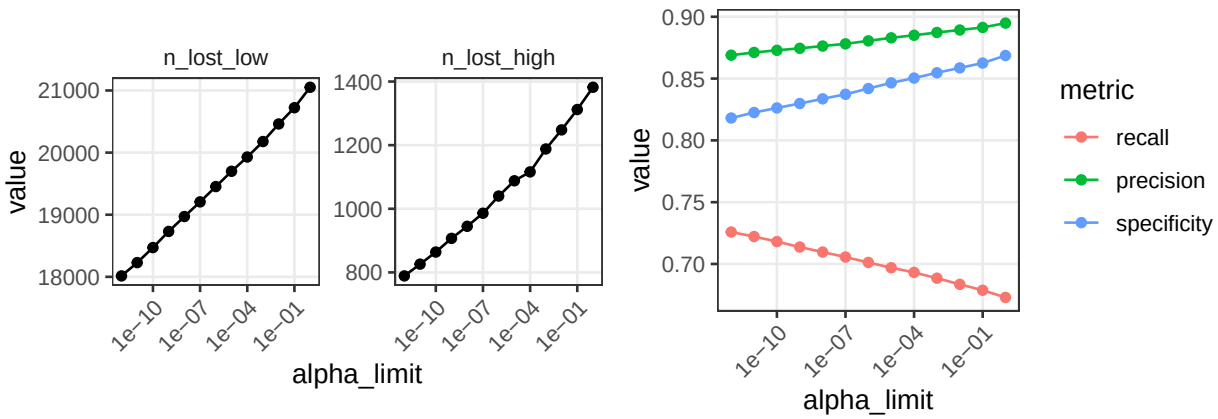
—

Effect of cleanup threshold on accuracy Given that varying the convergence parameters had minimal effect on accuracy, subsequent kallisto runs were performed using default convergence criteria (`alpha_change` = `alpha_change_limit` = 1e-2). The cleanup threshold (`alpha_limit`) was progressively increased from permissive (1e-12) to aggressive (1e0) to explore the effects of altering the cleanup threshold.

Quantification of performance was summarized using multiple complementary metrics: **(i)** recall, the fraction of truly expressed transcripts that were detected, **(ii)** precision, the fraction of detected transcripts that were truly expressed, and **(iii)** specificity, the fraction of truly absent transcripts that were left undetected.To assess sensitivity to low-abundance transcripts, the number of lost transcripts **(iv)** were stratified by expression level (low for transcripts with true counts < 10, high for all others). Computational cost was summarized as number of EM interations for each run.

Across the tested range of `alpha_limit` values, recall decreased modestly from ~0.75 to ~0.65, reflecting progressive loss of detectable transcripts as the cleanup threshold became more aggressive. Precision increased from ~0.85 to ~0.90, and specificity increased from ~0.80 to ~0.85, indicating that more stringent cleanup reduces false positives and improves confidence in detected transcripts. The number of lost transcripts rose with increasing `alpha_limit`, with low-abundance transcripts showing the largest effect (~3,000 lost), compared with high-abundance transcripts (~600 lost). These trends demonstrate a clear tradeoff: permissive `alpha_limit` values maximize sensitivity to rare transcripts but allow more false positives, whereas more aggressive values reduce false positives but sacrifice detection of low-abundance transcripts. Overall, the effect of `alpha_limit` on high-abundance transcripts is modest, suggesting that the choice of threshold primarily affects low-abundance transcript detection.

The runtime for all conditions were virtually identical, with convergence occurring at 95 iterations. This is expected, as post-EM zeroing should have no effect on the upstream EM run.



—

Conclusion Across all tested conditions, kallisto's default convergence parameters (`alpha_change` = `alpha_change_limit` = 1e-2) are sufficient for accurate quantification of expressed transcripts in bulk RNA-seq. Tightening convergence criteria yields minimal improvement in error metrics while substantially increasing runtime, indicating that EM convergence is already robust under default settings.

Adjusting the cleanup threshold (`alpha_limit`) primarily affects low-abundance transcripts: more permissive values preserve sensitivity but increase false positives, whereas more aggressive thresholds reduce false positives at the cost of losing low-abundance transcripts. High-abundance transcripts are largely unaffected.

Taken together, these results suggest that for standard bulk RNA-seq of moderately to highly expressed transcripts, kallisto's default parameters are appropriate, and aggressive tuning provides little benefit. The optimal strategy for detecting very low-abundance transcripts remains unclear and may require additional data-specific considerations.