Streaming approaches to error detection and trimming in the analysis of short sequencing reads

TBD

January 4, 2015

1 Introduction

K-mer spectral analysis is a powerful approach to error detection and correction in shotgun sequencing data that uses k-mer abundances to determine likely errors (cite Pevzner, Quake, khmer-counting paper). Approaches derived from spectral analysis can be very accurate: Zhang et al. (2014) show that spectral analysis is considerably more effective at finding errors than quality-based approaches (cite). However, spectral analysis is also very compute intensive; most implementations must count k-mers across entire sequencing data sets, which can be memory- or I/O-intensive for large data sets.

Streaming algorithms can offer improved algorithmic and computational efficiency in the analysis of large data sets (cite). Streaming algorithms typically examine the data only once, and scale in memory usage sublinearly with respect to the input data. Streaming algorithms have not been applied to k-mer spectral analysis of reads, although XXX (melsted).

Brown et al. (2012) introduced a streaming algorithm for normalizing k-mer abundance spectra, termed "digital normalization" (abbreviated as "diginorm"). This procedure estimates the k-mer coverage of each read in an online algorithm, by calculating the median k-mer abundance of the read given all previous reads; reads above a certain estimated coverage are set aside and their k-mers are not tracked. This algorithm is both online and *streaming* because it only collects k-mers in reads with a low estimated coverage; for the same reason, it is sublinear in memory for high coverage data sets. The net effect of diginorm is to reduce the data set size that must be considered for downstream processing, such as *de novo* assembly (cite trinity, elijah, etc.)

Here we develop a streaming algorithm for k-mer spectral analysis, based on digital normalization, that can detect and remove errors in sequencing reads. This algorithm operates in sublinear memory, and examines the data at most twice. The approach offers a general framework for making use of locus-specific graph saturation and could potentially be used for error correction, variant calling, and a streaming implementation of assembly. Moreover, it can be applied to data sets with variable coverage such as transcriptomes, metagenomes, and amplified genomic DNA.

Name	Origin	Number of reads	Description
simple genome	10,000	No repeats	
E. coli MG1655	5,000,000	Subset of NCXXXXX	
simple metagenome	2,347	Dynamic range of YYY	
simple transcriptome	568	Dynamic range of ZZZ; multiple exons	

Table 1: Data sets used

2 Results

2.1 Coverage-normalized data can be used to locate and correct errors in high-coverage shotgun sequencing data

Our initial question was whether we could apply spectral error analysis to genomic short read data after digital normalization. We tested this on a synthetic data set and an *E. coli* data set. We then compared the performance of Quake on the raw and digitally normalized counts from the *E. coli data*.

Simulated data: We first applied digital normalization to a simulated data set with known errors. We generated the synthetic data set from a simulated low-complexity genome ("simple genome"; see Methods for generation and Table 1 for data set details). We then applied digital normalization to these synthetic reads, normalizing to a median 20-mer coverage of 20 (k=20, C=20).

The k-mer spectrum before and after digital normalization is shown in Figure 1. While the total number of k-mers decreased in the digitally normalized data set, the separation between the high count k-mers and the low-count k-mers remains clear. The key concept underlying k-mer spectral error analysis is that in a high-coverage data set, these high count k-mers will represent *correct* k-mers, while the low count k-mers are produced by errors in the reads. Simple classification methods suffice to identify and trim or correct these low-count k-mers.

We next used k-mer counts from the downsampled read set to detect errors in the original read set. The algorithm is straightforward: we look for bases at the beginning or ends of low-abundance runs of k-mers in each read, which should signify the locations of errors. We used a "trusted k-mer" cutoff of $C_0 = 3$ as our abundance cutoff, below which we assumed k-mers were erroneous (see Methods). The results are presented in Table 2. Of the 531 simulated reads from the simple genome containing one or more errors, predicted errors matched the known truth exactly for 443 of them (true positives), and 466 reads were correctly predicted to contain no errors (true negatives). 0 reads were falsely predicted to have no errors (false negatives). The errors in 88 reads were miscalled – while the reads each had one or more errors, the positions were not correctly called – and three reads were incorrectly predicted to contain errors, leading to a total of 91 false positives. From this, we calculated the

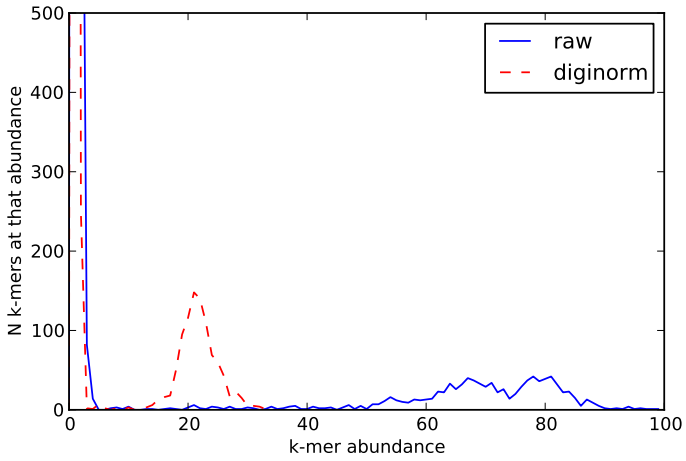

Figure 1: **K-mer spectrum of a simple artificial data set, before and after digital normalization. The peaks at 1 represents erroneous k-mers resulting from (simulated) error; the peaks centered at 80 (raw) and 20 (diginorm) represent k-mers truly present in the genome which are present in many reads.**

prediction sensitivity to be 100% and the prediction specificity to be 83.0%.

When we applied spectral error detection to the unnormalized reads, we saw similar results: 440 TP, 455 TN, 105 FP, and 0 FN, for a sensitivity of 100% and a specificity of 80.7% (Table 2). (For this analysis we used a cutoff of $C_0 = 10$.)

***E. coli* reads:** We next applied digital normalization and k-mer spectral error detection to an Illumina data set from *E. coli* MG1655 (cite Chitsaz). In real reads, we do not know the location of errors; to calculate likely errors, we mapped 5m untrimmed reads to the known *E. coli* MG1655 genome with bowtie2 (cite) and recorded mismatches between the reads and the genome. These mismatches were taken to be errors in the reads. We found 8.0m errors in 2.2m reads, for an overall error rate of 1.60%.

We then compared the results of k-mer spectral error detection with and without digital normalization. We used the same parameters as on the simulated genome ($C_0 = 10$ for unnormalized, $C_0 = 3$ for normalized). The results are presented in Table 3. Using the raw counts, the sensitivities were close between the unnormalized and normalized predictions: using the raw counts, we achieved a sensitivity of 99.8%, versus 99.4% using the counts from the digitally normalized reads. The specificities were also comparable – 47.9% using the raw counts, and 47.5% using the digitally normalized counts.

(Simple genome)	Raw counts	Diginorm counts
Perfect detection (TP)	474	485
No errors (TN)	355	366
Miscalled errors (FP)	159	148
Mispredicted errors (FP)	12	1
Missed errors (FN)	0	0
Sensitivity	100%	100%
Specificity	73.5%	76.5%

Table 2: Spectral error detection on 1000 synthetic reads from a simulated 10kb genome, using raw and digitally normalized counts. The counts are the number of reads where all errors were detected perfectly (TP), no errors were present (TN), one or more errors were miscalled (one type of FP), errors were mistakenly called in an error-free read (the other type of FP), and errors present in a read were missed (FN).

(E. coli)	Raw counts	Diginorm counts
Distinct k-mers	39,677,503	26,296,651 (66%)
Perfect detection (TP)	1,042,325	1,030,787
No errors (TN)	2,782,265	2,782,413
Miscalled errors (FP)	1,133,049	1,140,148
Mispredicted errors (FP)	474	326
Missed errors (FN)	2,135	6,574
Sensitivity	99.8%	99.4%
Specificity	47.9%	47.5%

Table 3: Spectral error detection on 5m *E. coli* reads, using raw and digitally normalized counts.

Sample	N raw reads	N normalized reads
E. coli		

Table 4: Read counts (N) for raw and normalized data sets using a k-mer size of 20 and the specified coverage cutoff. Digital normalization reduces the total number of reads considered informative.

Sample	raw read M	normalized read M
E. coli		

Table 5: Unique k-mer counts for raw and normalized data sets using a k-mer size of 20 and the specified coverage cutoff. Digital normalization reduces the total number of k-mers being tracked.

Sample	raw	diginorm
Erroneous reads	22,902	17,609
Total reads	4,906,469	4,905,853
Total errors remaining	49,823	43,238
Total bp	450,687,634	450,630,551
Per-base error rate	0.011%	0.010%

Table 6: Comparison of Quake results when run on the same *E. coli* data set, using k-mers from either the original data set ("raw") or just the digitally normalized reads ("diginorm"). All numbers are post-error correction; the pre-correction error rate was 1.60%.

***E. coli* error correction with Quake:** While the results above suggest that simple spectral error detection works equally well both before and after digital normalization, we were concerned that we might lose informative reads or k-mers during digital normalization. To evaluate this, we used a more sophisticated spectral error correction algorithm, implemented in Quake, to perform error correction using the digitally normalized counts. Here, because Quake permits the use of a different k-mer counting table from the data being corrected, we could separately use the counts from the raw reads and the digitally normalized reads to correct the raw reads (see Methods for details).

The results of running Quake on the raw data using counts from the raw and digitally normalized data are shown in Table 6. The performance was essentially the same: Quake brought the overall error rate in the data set from 1.60% (8.0m errors) to 0.005%-0.006% (23,000 - 29,000 errors).

These results demonstrate that digitally normalized counts retain all of the information necessary for effective error correction with Quake, despite there being many fewer distinct k-mers.

2.2 Coverage-normalized data can be used to locate errors in variable coverage shotgun sequencing data

Digital normalization works on both genomic data, with even coverage, and variable coverage data such as transcriptome and metagenome data (cite diginorm, elijah). However, one of the drawbacks of spectral abundance analysis is that it does not directly apply to data with uneven coverage. For example, metagenomic or transcriptomic data sets typically contain reads from both high-abundance and low-abundance molecules. This in turn leads to high coverage and low coverage reads in the same data set. This variability in coverage confounds naive spectral analysis for two reasons: first, erroneous k-mers from very high abundance regions can accumulate and increase in abundance over the threshold for trusted k-mers, thus appearing to be correct; and second, correct reads from low coverage regions yield k-mers below the trusted k-mer threshold that appear to be incorrect. In practice, therefore, error analysis for metagenomic and transcriptome data uses other approaches than direct spectral error

analysis (cite).

A key concept developed in digital normalization is that of an estimator of per-read coverage, the median k-mer abundance within a read. Using median k-mer abundance, we developed a general approach that enables spectral error analysis to be used for variable coverage data. We then applied this to two synthetic data sets as well as two real data sets, a mock shotgun metagenome and mRNAseq data from mouse.

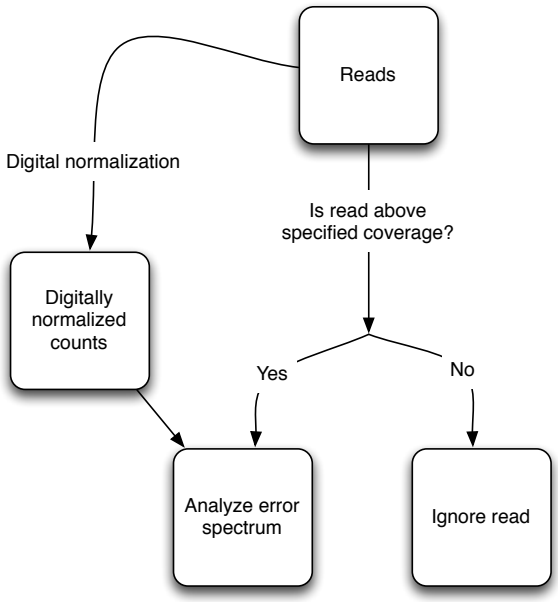

Figure 2: **Coverage-normalized spectral error analysis. Reads are normalized, and high-coverage reads are subjected to spectral error analysis with the normalized counts, while low-coverage reads are ignored.**

Coverage-normalized spectral error analysis: Using digital normalization, we should be able to address both the problem of *too high* coverage and *too low* coverage. First, by applying digital normalization to variable coverage data and then working only with the k-mer counts from the normalized reads, we can avoid counting high abundance errors. Second, by ignoring reads with a low estimated coverage, we can avoid misclassifying true low-abundance k-mers. The process is shown in Figure 2.

Simulated data: To test this approach, we generated two more synthetic data sets, "simple metagenome" and "simple mRNAseq," which contain both high- and low-abundance species (see Table 1 for data set details). After generating synthetic reads with a 1% error rate and applying digital normaliza-

	simple metagenome	simple mRNAseq
K-mer coverage threshold	20	20
Total reads	2347	568
High coverage reads	2254 (96.0%)	524 (92.3%)
Perfect detection (TP)	978	228
No errors (TN)	1098	235
Miscalled errors (FP)	170	52
Mispredicted errors (FP)	6	9
Missed errors (FN)	2	0
Sensitivity	99.8%	100%
Specificity	84.7%	78.9%

Table 7: Variable coverage spectral error detection on two synthetic data sets, a simple metagenome and a simple mRNAseq data set. Per-read coverage was estimated by median k-mer abundance within the read, and only the reads with estimated coverage at or above the specified threshold were analyzed. Digitally normalized counts were used for the spectral error analysis.

tion (k=20/C=20), we again did spectral error detection using the normalized counts. However, this time we modified the algorithm so that only reads with a median k-mer abundance of C or greater were examined (see Methods).

The results of running error detection on the synthetic metagenome and mRNAseq data sets are shown in Table 7. For the simple metagenome data set, 2254 of 2347 reads (96.0%) met the coverage criterion. Of the 2254 reads analyzed, the errors in 978 erroneous reads were called perfectly (TP) and 1098 of the reads with no errors were correctly called as error-free (TN). 2 reads were incorrectly determined to be error free (FN). Of the remaining 176 reads, 170 were miscalled (errors existed but were not exactly called) and 6 were incorrectly called as erroneous when they were in fact correct. We calculated the prediction sensitivity to be 99.9% and the prediction specificity to be 84.7%. For the simple mRNAseq data set, 524 of 568 reads (92.3%) met the coverage criterion, with 228 true positives, 235 true negatives, 0 false negatives, and 61 false positives, for a prediction sensitivity of 100.0% and prediction specificity of 78.9%. In neither case were low-coverage reads included in the statistics.

Importantly, these results are comparable to the results on the synthetic genome (100.0% sensitivity and 83.0% specificity with the same parameters; see Table 2).

mRNAseq data: To evaluate coverage-normalized spectral analysis on real data, we applied variable coverage spectral error analysis to 1m mouse mRNAseq reads (cite Trinity). After calling errors in the reads by mapping them back to the known genomes, we used spectral analysis to identify putative errors. The results are shown in Table 8, third column. We achieved 92.9% sensitivity and 80.7% specificity on the 340,000 high coverage reads in this data set.

	Mock metagenome	mouse mRNAseq
K-mer coverage threshold	10	20
Total reads	4,667,491	791,687
High coverage reads	404,896 (8.7%)	260,641 (32.9%)
Perfect detection (TP)	11,743	90,954
No errors (TN)	366,253	141,009
Miscalled errors (FP)	3,911	20,547
Mispredicted errors (FP)	21,750	1173
Missed errors (FN)	1239	6931
Sensitivity	90.5%	92.9%
Specificity	31.4%	80.7%

Table 8: The results of variable coverage spectral error detection on two real variable coverage data sets, a mock shotgun metagenome and a mouse mRNAseq data set. Per-read coverage was estimated by median k-mer abundance within the read, and only the reads with estimated coverage at or above the specified threshold were analyzed. Digitally normalized counts were used for the spectral error analysis.

Mock metagenome data: We next applied our approach to 5m reads from a diverse mock community data set (Shakya et al., 2013). We used a coverage threshold of 10 for digital normalization, and found that 404,896 reads were at or above this coverage threshold. Here errors were again calculated by mapping the reads to the known reference and finding mismatches. The results are shown in Table 8, second column. We achieve 90.5% sensitivity and 41.4% specificity on the high coverage reads.

Error correcting variable coverage data with Quake: There are many sophisticated error correction algorithms implemented for shotgun genome data, but relatively few work directly on variable coverage data such as mRNAseq. Digital normalization, in theory, enables the use of *any* genomic error correction algorithm to the high coverage components of data sets.

To evaluate this, we used Quake (a genomic error corrector) to correct the high coverage mRNAseq reads using the diginorm counts. We first extracted the 260,641 reads with estimated coverage greater than or equal to 20 from the mouse mRNAseq data set, and then digitally normalized the data. We next applied the Quake error corrector to the unnormalized high-coverage reads using the k-mer counts from the normalized reads, as with the *E. coli* data set. Quake discarded 23,606 reads and corrected the remainder.

We evaluated the corrected reads by mapping them against the mouse transcriptome, and found that, for the retained reads, Quake reduced the overall error rate from 1.28% (230,125 mismatches in 18,012,456 bp total) to 0.43% (74,797 in 17,254,585 total). As with *E. coli*, this suggests that sufficient information remains in the digitally normalized data to do an effective job of error correction.

2.3 A streaming algorithm for error analysis

The spectral error detection approach outlined above is a 2-pass offline algorithm for any given data set - the first pass normalizes the read set and records the k-mer abundances, while the second pass analyzes the reads for low-abundance k-mers. Even with digital normalization reducing the number of k-mers under consideration, this 2-pass approach is time consuming on large data sets. Below, we develop a general streaming approach that considers many of the reads only once.

Streaming analysis of coverage-saturated regions: Shotgun sequencing oversamples most regions – for example, for a 100x coverage genomic data set, we would expect 50% or more of the genome to be represented by more than 100 reads. This is a consequence of the Poisson-random sampling that underlies shotgun sequencing (cite Waterman?). This oversampling provides an opportunity, however: if we regard the read data set as a stream of incoming data randomly sampled from a pool of molecules, high-abundance species or subsequences within the pool will be more highly sampled earlier in the stream than others. For example, in mRNAseq, highly expressed transcripts will on average be highly sampled much earlier than low-expressed transcripts.

We can adapt the same approaches used in previous sections to do *streaming* error analysis by detecting high-coverage reads *during* the first pass. Here we again use the median k-mer abundance within a read to estimate read coverage (cite diginorm); crucially, this can be done at any point in a stream, by using the online k-mer counting functionality of khmer to determine the abundance of k-mers seen thus far (cite khmer-counting).

The conceptual idea is presented in Figure 3. On the first pass, low-coverage reads would be incorporated into the k-mer database and set aside for later analysis, while high-coverage reads would be analyzed for errors. On the second pass, the set aside reads would be checked for coverage again, and either ignored or analyzed for errors. Crucially, this second pass involves *at most* another full pass across the data, but only when the entire data set is below the coverage threshold; the larger the high coverage component of the data, the smaller the fraction of the data that is examined twice.

In Figure 4, we show diginorm-generated coverage saturation curves for both real and error-free simulated reads from *E. coli* MG1655. In both cases, after the first 1m reads, the majority of reads have an estimated coverage of 20 or higher, and hence can be used for error analysis on the first pass through the data.

We first apply this streaming error detection approach to the three synthetic data sets used earlier, and then to the three real data sets.

Streaming error analysis of synthetic data: Using the streaming approach on the "simple genome" reads, we obtain nearly identical numbers to the full two-pass approach: 485 TP, 365 TN, 150 FP, and 0 FN, for a sensitivity

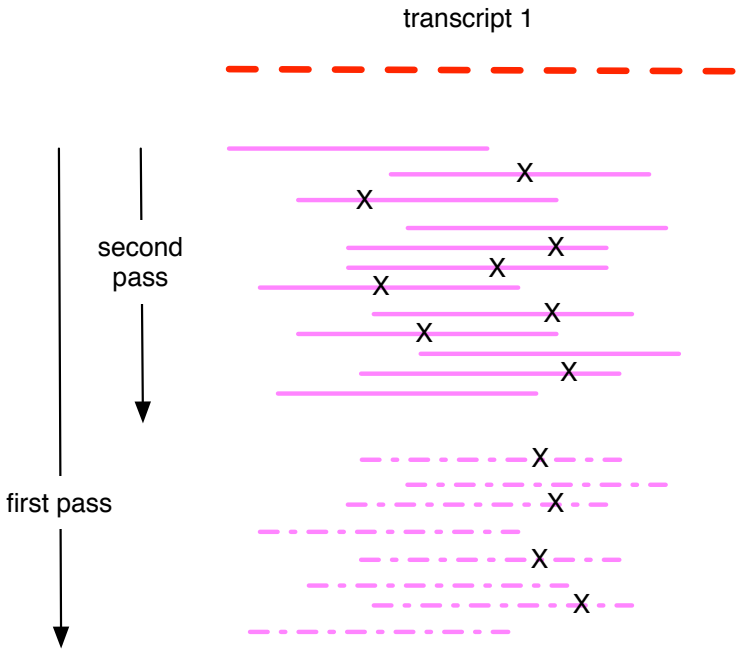

Figure 3: **Diagram of streaming error detection. In a first pass over the read data, reads are loaded in until the graph locus to which they belong is saturated. From that point on, reads are examined for errors and not loaded into the graph. In a second pass, only the subset of reads loaded into the graph are examined for errors.**

of 100% and a specificity of 76.3% (Table 9). However, with the streaming algorithm, only 320 of the 1000 reads are examined twice. Likewise, for the "simple mRNAseq" and "simple metagenome" data sets, we obtain identical and nearly identical results, respectively; due to differences in the order in which reads are examined, the simple metagenome fails to detect one true positive and erroneously finds errors in three extra reads. On the mRNAseq data set, 33.1% of the reads are examined twice, and on the metagenome, 380 of 2347 (16.2%) of the reads are examined twice.

Streaming error analysis of real data: We also get similar quality results on the real data sets when comparing two-pass error detection with streaming error detection (Table 10). For *E. coli*, with streaming error detection we obtain a sensitivity of 99.5% and a specificity of 47.6%, compared to 99.8% and 47.9% with the two-pass approach (Table 3). For the mock metagenome, we have a sensitivity of 90.6% with streaming, vs 90.5% with the two-pass approach; and a specificity of 30.0% with streaming, vs 31.4% with the two pass approach (compare Table 10 and Table 8). And for the mRNAseq data set, we see a

	simple genome	simple metagenome	simple mRNAseq
Number of passes	1.32	1.16	1.33
Perfect detection (TP)	485	977 (-1)	228
No errors (TN)	365 (-1)	1095 (-3)	235
Miscalled errors (FP)	148	171 (+1)	52
Mispredicted errors (FP)	2 (+1)	9 (+3)	9
Missed errors (FN)	0	2	0
Sensitivity	100.0%	99.9%	100.0%
Specificity	76.4%	84.4%	78.9%

Table 9: Results from applying streaming error detection to the same synthetic data sets as in Table 2 and Table 7. Number of passes is the average number of times each read in the data set was examined; numbers in parentheses give the difference between these numbers and the previous results.

	E. coli	mock metagenome	mouse mRNAseq
Number of passes	xx	yy	zz
Perfect detection (TP)	1,033,261	11,726	91,133
No errors (TN)	2,781,961	364,627	140,461
Miscalled errors (FP)	1,138,940	3,951	20,742
Mispredicted errors (FP)	778	23,376	1721
Missed errors (FN)	5308	1216	6584
Sensitivity	99.5%	90.6%	93.3%
Specificity	47.6%	30.0%	80.2%

Table 10: Results from applying streaming error detection to the same real data sets as in Table 3 and Table 8. Number of passes is the average number of times each read in the data set was examined; unless noted in parentheses, numbers were within 1% of non-streaming results.

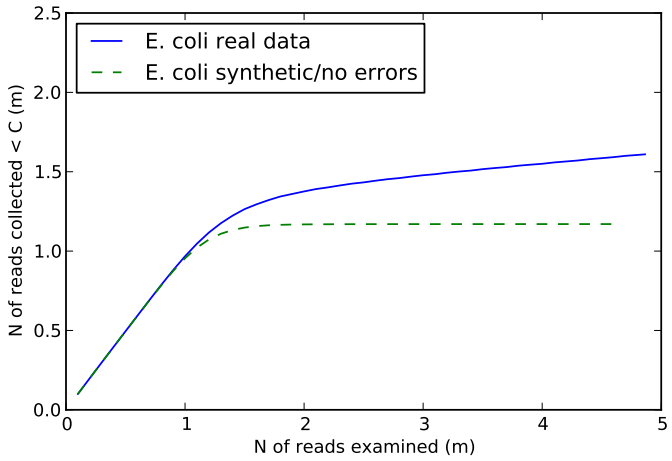

Figure 4: **Saturation curve of a real and a simulated *E. coli* read data set. Reads are collected when they have an estimated coverage of less than 20; in the early phase (< 1m reads), almost all reads are collected, but by 2m reads into the data set, the majority of reads come from loci with an estimated sequencing depth of > 20 and are rejected.**

sensitivity of 93.3% with streaming vs 92.9% with two-pass, and a specificity of 80.2% vs 80.7% for streaming vs two-pass, respectively. However, the streaming approach examined the *E. coli* reads only XX times, the metagenome reads YY times, and the mRNAseq reads ZZ times on average.

2.4 A streaming algorithm for error trimming

Once errors can be *detected* with a streaming algorithm, errors can also be *removed* by trimming reads at the first base predicted to be erroneous in a read. This approach is remarkably effective, but can require considerably more memory than quality-score based trimming, and is currently implemented as an offline algorithm (cite khmer-counting). Below, we apply the same approach shown in Figure [**?**], but instead of predicting errors, we trim each read at the first predicted error.

Streaming error trimming on synthetic data: On the "simple genome" with counts from the digitally normalized reads, this trimming approach eliminates 149 reads entirely due to a starting low-abundance k-mer, and truncates another 392 reads. Of the 100,000 bp in the simulated reads, 31,910 (31.9%) were removed by the trimming process. In exchange, trimming eliminated *all* of the errors, bringing the overall error rate from 0.63% to 0.00%.

Data set	pre-trim error	% bp trim	% reads trim	post-trim error
E. coli	1.60%			0.05%
Mock metagenome	0.32%			0.30%
mouse mRNAseq	2.07%			1.84%
(high coverage only)	1.61%			0.66%

Table 11: A summary of trimming statistics for streaming error trimming. Error rates before and after trimming were estimated by mapping mismatches.

For the simple metagenome we used the variable abundance approach described above and only trimmed reads with estimated coverage of 20 or higher. Here, of 2347 reads containing 234,700 bp, 314 reads (13.4%) were removed and 851 reads (36.3%) were trimmed, discarding a total of 74,321 bases (31.7%). Of 1451 errors total, all but 61 were eliminated, bringing the overall per-base error rate from 0.62% to 0.04%. The simple mRNAseq data set showed similar improvement: 83 of 568 reads were removed, and 208 were trimmed, removing 19,507 of 56,800 bases (34.34%). The initial error rate was 0.65% and the final error rate was 0.07%.

Streaming error trimming on real data: Applying the streaming error trimming to the *E. coli* MG1655 data set used in section XXX, we trimmed 2.0m reads and removed 98,903 reads entirely. Of 8.0m errors, all but 203,345 were removed, bringing the error rate from 1.60% to 0.05%. Trimming discarded 61 Mbp of the original 500 Mbp (12.4%).

On the mouse mRNAseq data set, streaming error trimming removed 28,219 reads and trimmed 70,260 reads, removing 5.52% of the total bases, bringing the overall error rate from 2.1% to 1.8%. When we measured only the error rate in the high-coverage reads, trimming brought the error rate from 1.61% to 0.66%. On the mock metagenome data set, 5823 reads were removed and 21,231 reads were trimmed, removing 0.26% of bases; this low percentage is because of the very low coverage of most of the reads in this data set. XXX

2.5 Illumina error rates and error profiles can be determined from a small sample of sequencing data

K-mer spectral error analysis can also be used to calculate per-position sequencing error for entire data sets (cite khmer-counting). We can adapt the streaming approaches above to efficiently provide estimates for *subsets* of the data. The basic idea is to consume reads until sufficient data has been collected to calculate error rates, and then to calculate those error rates for the new reads based on the k-mer abundances from the old reads. This can aso be done in one pass for data sets with sufficiently high coverage data: as shown above (Figure 4), in some data sets, most of the reads will have sufficient coverage to call errors by the time 20% of the data set has been consumed.

Using the same error detection code as above, we implemented a sublinear

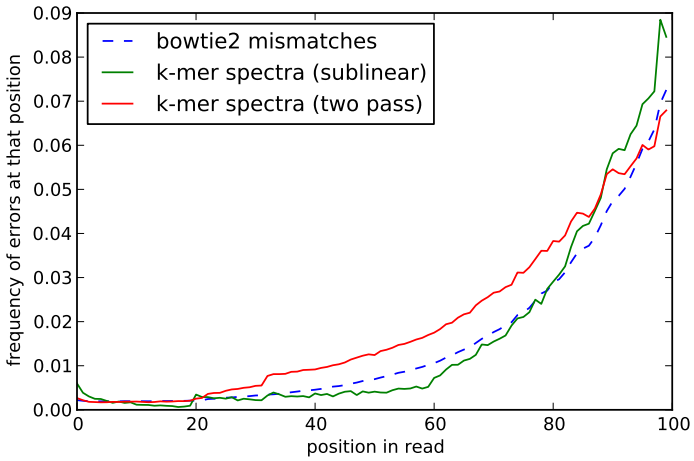

Figure 5: **Error spectrum of reads in the *E. coli* data set. The sublinear k-mer spectrum analysis is calculated based on saturation of a fraction of the data set, while the two-pass spectral analysis uses all of the data, and bowtie2 mismatches are based on all mapped reads. The y values for the k-mer spectral analyses are scaled by a factor of four for ease of comparison.**

memory/sublinear time algorithm that collects reads until some regions have reached 20x coverage, or 200,000 reads have surpassed a coverage of 10x (see Methods for details). In either case, all reads at or above a coverage of 10 are analyzed for errors, with a trusted k-mer cutoff of 3. In Figure 5 and Figure 6 we show the resulting error profiles for the *E. coli* and mouse RNAseq data sets, compared with the profile obtained by examining the locations of mismatches to the references. We also show the error profile obtained with the full two-pass approach (using digital normalization and then error detection as in section XXX) for comparison.

In the *E. coli* data set (Figure 5), we see the increase in error rate towards the 3' end of the gene that is characteristic of Illumina sequencing (cite). All three error profiles agree in shape (Pearson's correlation of 0.99 between each pair) although they are offset considerably in absolute magnitude. The k-mer error profile was calculated from the first XXX reads, but is consistent across five other subsets of the data chosen randomly with reservoir sampling (data not shown); all five subsets had Pearson's correlation coefficients greater than 0.99 with the bowtie2 mapping profile and the two-pass spectral approach.

The RNAseq error profile exhibits two large spikes, one at position 34 and one at position 69. Both spikes appear to be genuine and correlate with large numbers of Ns in those positions in the original data set. The spikes are present in the profiles derived from two-pass spectral analysis as well as the bowtie2

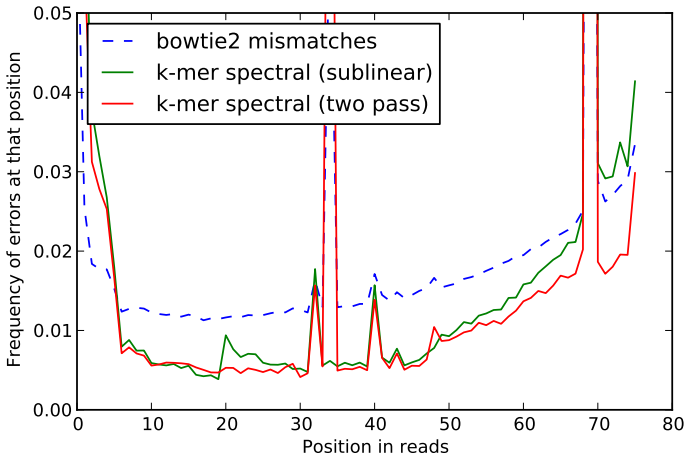

Figure 6: **Error spectrum of reads in the mouse RNAseq data set. The sublinear k-mer spectrum analysis is calculated based on saturation of a fraction of the data set, while the two-pass spectral analysis uses all of the data, and bowtie2 mismatches are based on all mapped reads. The peak of errors at position 34 in the bowtie2 mapping reflects errors that in the first part of the data set are called as Ns, and hence are ignored by the sublinear error analysis; see text for details. Note, the y values for the k-mer spectral analysis are scaled by a factor of four for ease of comparison.**

mismatch calculation. However, the sublinear approach does not detect them when using the first YYY reads. This is because of the choice of subsample: five other subsamples, chosen randomly from the entire data set with reservoir sampling, match the match the two-pass spectral analysis (data not shown). The error profiles calculated from all six subsamples with the sublinear algorithm have a Pearson's correlation coefficient greater than 0.96 with the error profiles from the full two-pass spectral approach and the bowtie2 mismatches.

3 Discussion

3.1 Digital normalization can be applied effectively to short reads prior to error detection and correction.

Tracking k-mer abundances in large short-read data sets is part of many error detection and correction algorithms, but this process can be time and memory intensive. Our results demonstrate that digital normalization can be used to reduce the total number of k-mers under consideration, without strongly affecting

performance. We showed this in both simulated and real data.

With a real *E. coli* data set, digital normalization reduced the number of k-mers by a third (Table 3, Distinct k-mers) yielding essentially the same sensitivity and specificity of error predictions. Moreover, when we ran the Quake error corrector on the reads using unnormalized and normalized counts (Table 6, we achieved nearly identical results, demonstrating that digital normalization does not remove information critical for error correction.

3.2 K-mer counts from digitally normalized short reads can be used to error correct mRNAseq and metagenome data

Spectral error correction approaches typically rely on assumptions of uniform sequence coverage, but these assumptions are violated by several types of data, including mRNAseq and shotgun metagenome data. Digital normalization evens out this coverage, allowing existing spectral error correction approaches to be applied to data from samples with non-uniform abundances. We demonstrated this by using spectral error detection with digitally normalized data to predict errors in both synthetic and real RNAseq and metagenome data (Table 8. We then again used Quake to error correct high-coverage portions of mRNAseq and shotgun metagenome data sets, which yielded good results (Table **??**. This again demonstrates that digital normalized data can retain the information necessary to error correct high coverage reads.

Interestingly, we believe this to be a general approach that could allow any error correction software that separates k-mer counting from error correction to be applied to mRNAseq, shotgun metagenome, and whole-genome amplified samples.

3.3 Short-read error detection and trimming can be done efficiently with a streaming few-pass sublinear-memory algorithm

K-mer spectral error detection, trimming, and correction approaches are typically implemented as a two-pass offline algorithm, in which k-mer counts are collected in a first pass and then reads are corrected in a second pass. While several algorithms that run in sub linear memory do exist (e.g. Lighter), these are still offline algorithms that require at least two full passes across the data. In high coverage data sets it is possible, however, to detect reads that are high coverage in the context of previously encountered reads. By integrating k-mer spectral error analysis directly into the digital normalization algorithm, we showed that on both synthetic and real data sets, we achieved nearly identical predictions to the full two-pass algorithm with an algorithm that is less than two pass (compare Table 8 to Table 10).

We next adapted the error detection algorithm to do streaming error trimming on genomic, metagenomic, and transcriptomic data. On high coverage

components of variable coverage data sets, this led to about a 10x decrease in errors.

3.4 Data-set wide error profiles can be calculated in sub linear time and memory

The ability to analyze high-coverage reads without examining the entire data set offers some intriguing possibilities. One concrete application is the use of high coverage reads to infer data-set wide error characteristics for shotgun data, in a way that is robust to the sample. Can also be used to assess whether the necessary coverage has been obtained in order to truncate sequencing, in e.g. metagenomics workflows.

More generally, the approach of using saturating coverage to truncate computational analysis may have application to streaming sequencing technologies such as Nanopore, where realtime feedback between sequencing and sequence analysis could be useful.

3.5 Concluding thoughts

We describe time- and memory- efficient general algorithmic approaches to k-mer spectral error detection and correction based on read-local analysis of coverage.

These approaches can be applied to variable coverage data, including mRNAseq and shotgun metagenome reads.

Future applications include streaming error correction, reference-free variant calling, and reference-free analysis of streaming sequencing data.