Wei et al; Supporting Information Appendix

Table of contents

Description and usage of the k-Seek package:

The two main programs in the package are k_finder.pl and k_counter.pl, which identifies and quantifies kmers, respectively.

Command:

```
perl k_finder.pl input.fastq output
```

k_finder.pl breaks the ~100 bp sequence reads from a fastq file into small fragments which are tallied in a hash table indexed by the sequence of the fragments.

For example:

ATCGAATCGAATCGAATCGAATCGAATCGAATCGAATCGAATCGAATCGA

ATCGAATCGAATCGAATCGAATCGAATCGAATCGAATCGAAGGCTAGATG

5mer hash:

Index	Count	Keep
ATCGA	18	Yes
AGGCT	1	No
AGATG	1	No

To determine the most appropriate fragment length, hash tables are generated for word lengths of 5-10 bps for each read. For each hash table, fragments with very few counts are discarded as they are largely derived from non-repetitive sequences. The cutoff to discard fragments was adjusted empirically for optimal performance when processing 100 bp reads. Hash tables with only one index will be deemed as the correct fragment length.

For example:

Here is ACTGACTGAT repeated 10 times:

ACTGACTGATACTGACTGATACTGACTGATACTGACTGATACTGACTGAT

ACTGACTGATACTGACTGATACTGACTGATACTGACTGATACTGACTGAT

5-10mer hash tables are generated from this sequence:

5mer	
Index	Count
ACTGA	10
CTGAT	10

6mer	
Index	Count
ACTGAC	4
TGATAC	3
TGACTG	3
ATACTG	3
ACTGAT	3
TGAT	1

7mer	
Index	Count
ACTGACT	2
GATACTG	2
ACTGATA	2
CTGACTG	2
ATACTGA	1
CTGATAC	1
TGACTGA	1
TACTGAC	1
TGATACT	1
GACTGAT	1
AT	1

8mer	
Index	Count
ACTGACTG	3
ATACTGAC	3
TGATACTG	2
ACTGATAC	2
TGACTGAT	2
TGAT	1

9mer	
Index	Count
ACTGACTGA	2
TACTGACTG	1
ATACTGACT	1
ACTGATAC	1
GATACTGAC	1
TGATACTGA	1
CTGATACTG	1
ACTGATACT	1
GACTGATAC	1
TGACTGATA	1
CTGACTGAT	1
T	1

10mer	
Index	Count
ACTGACTGAT	10

Only the 10mer hash table contains a single unique index, and therefore produces the best identification.

In cases where the smallest hash table contains two indices, k_finder.pl checks whether the two indices are offsets of the same sequence (e.g. ACTG, CTGA, TGAC, and GACT). This will capture kmers interrupted by indels, as they shift the “reading frame” of sequences.

After the initial identification process, k_finder.pl checks the sequence for internal repetition, by further breaking it into substrings. This allows 2-4mers to be identified. For example ACTGACTG will be identified from the initial 8mer hash. However, this sequence will be tested for internal repeats, revealing two ACTGs and therefore recorded as a 4mer instead of a 8mer. For each read, the program outputs the fastq format with an additional line providing the inferred kmer, and an estimate of its occurrence based on its value in the hash table. The output file here will contain .rep suffix.

Command:

```
perl k_counter.pl output.rep
```

The k_counter.pl program takes the inferred kmer and word-searches the fastq read for the number of occurrences using Perl's regular expression capability. To ensure tandem-ness, it only counts search hits that are either immediately preceded or followed by another hit. If two hits are gapped by a sequence of the same length, k-counter examines that sequence for single bp differences with the inferred kmer. For 4-10mers, if the gap sequence contains only one bp difference from the kmer, it is also counted, thus capturing point mutations and sequence errors. However, gap sequences with more than one bp difference are not counted.

Example:

Here is the 10mer ACTGACTGAT repeated but with an insertion (red) and a substitution (gray background).

ACTGACTGATACTGACCACCTGATACTGACTGATACTGACTGATACTGAC

TGATACTGACTGATACTGACTGATACTAACTGATACTGACTGATACTGAC

k-finder will identify the 10mer TGATACTGAC repeated 7 times as the following 10mer hash table will be generated:

Index	Count	Keep
ACTGACTGAT	1	No
ACTGACCACC	1	No
TGATACTGAC	7	Yes
TGATACTAAC	1	No

k-counter word searches the read with TGATACTGAC, counting all matches (underlined).

ACTGACTGATACTGAC[1]CACCTGATACTGAC[2]TGATACTGAC[3]TGATACTGAC[4]

TGATACTGAC[5]TGATACTGAC[6]TGATACTAACTGATACTGAC[7]TGATACTGAC[8]

TGATACTAAC is also counted as it contains only one substitution. However, because the first match is neither immediately preceded nor followed by another copy it is discarded, yielding a total of 8 counts of the 10mer TGATACTGAC – one more than the counts from k-finder.

k-counter produces two outputs with .sep and .total suffixes. The former is similar to .rep containing the updated counts with incorrect kmer calls removed. The latter is a list of all inferred kmers and their total counts across the entire .sep file.

To obtain estimates of the total kmer quantities from multiple WGSs, use k_compiler.pl.

Command:

```
perl k_compiler.pl folder/ output
```

The k_compiler.pl program will take a folder containing multiple .total files. It outputs a tab delimited table where each column is a kmer, and each line is one sequencing sample. In addition, reverse complements and different offsets of the same kmer (e.g. AAGAG, AGAGA, GAGAA, AGAAG, GAAGA) are summed; the offset of the highest alphabetical order is listed (e.g. AAGAG). Table S2 shows the counts of all kmers for the 84 lines, normalized to a bp per 1x genome sequence (this is why there are fractional counts).

The package is optimized to process 100 bp reads, but can be applied to longer and shorter reads.

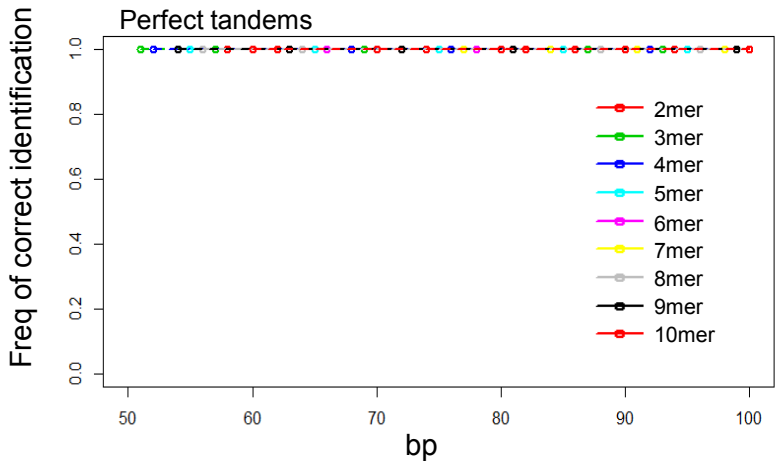
Supplementary table

Table S1. Counts for all possible kmers and identified kmers.

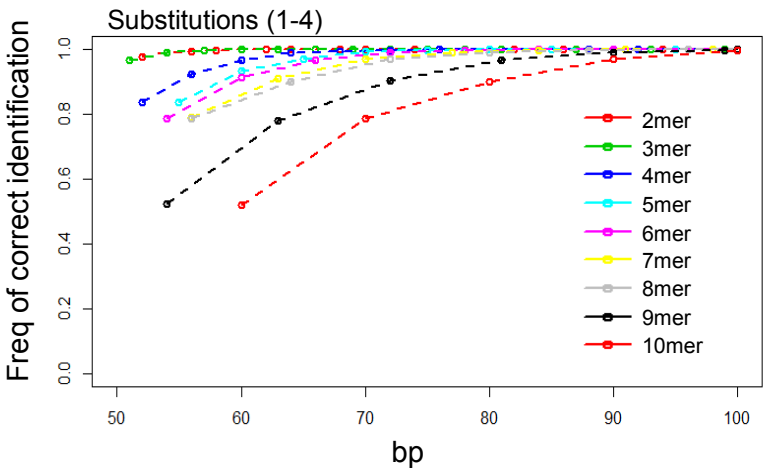
	Possible kmers	Median > 100bp	Top 100	All
2mer	4	3	3	3
3mer	10	5	6	9
4mer	33	1	1	18
5mer	102	17	21	59
6mer	350	12	14	113
7mer	1170	7	13	115
8mer	4140	4	4	77
9mer	14560	6	8	72
10mer	52632	17	30	202
total	73001	72	100	668

Supplementary Figures

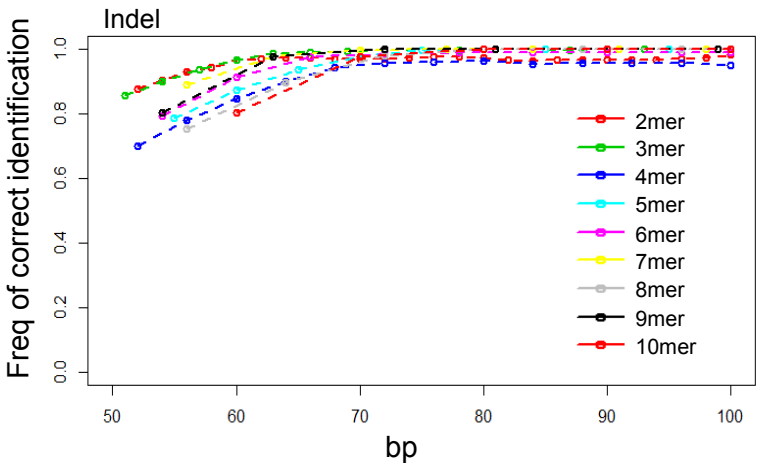
A



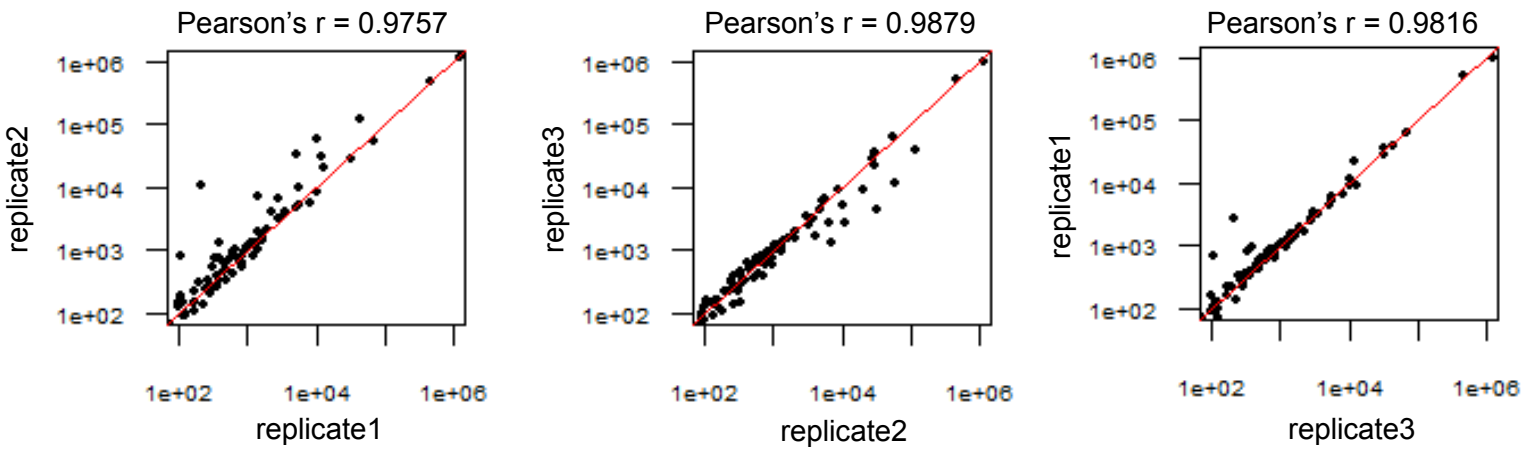
B



C



Supplementary Figure S1. Breakdown of k-Seek performance. Frequency of correct identification is plotted for all kmer sizes, as well as the size of repetition greater than 50bp for perfect tandems (A), tandems with up to 4 substitutions (B), and tandems with indels (C).

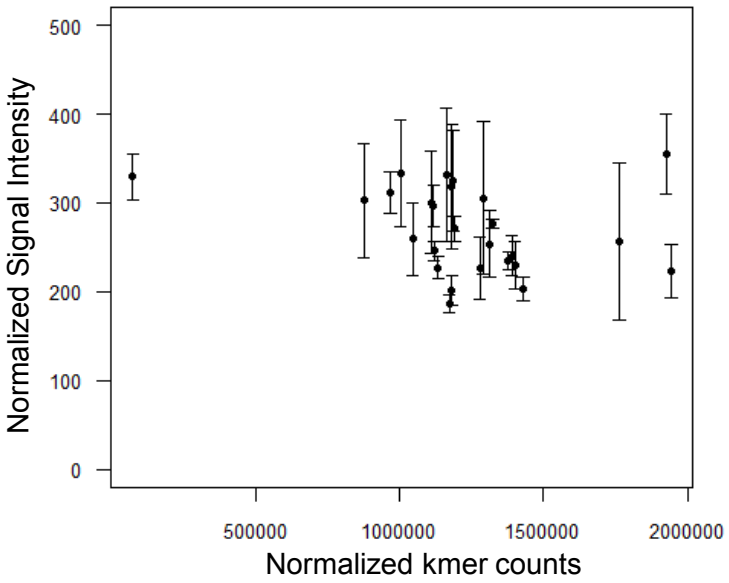


Supplementary Figure S2. Correlation of the ZW155 replicates.

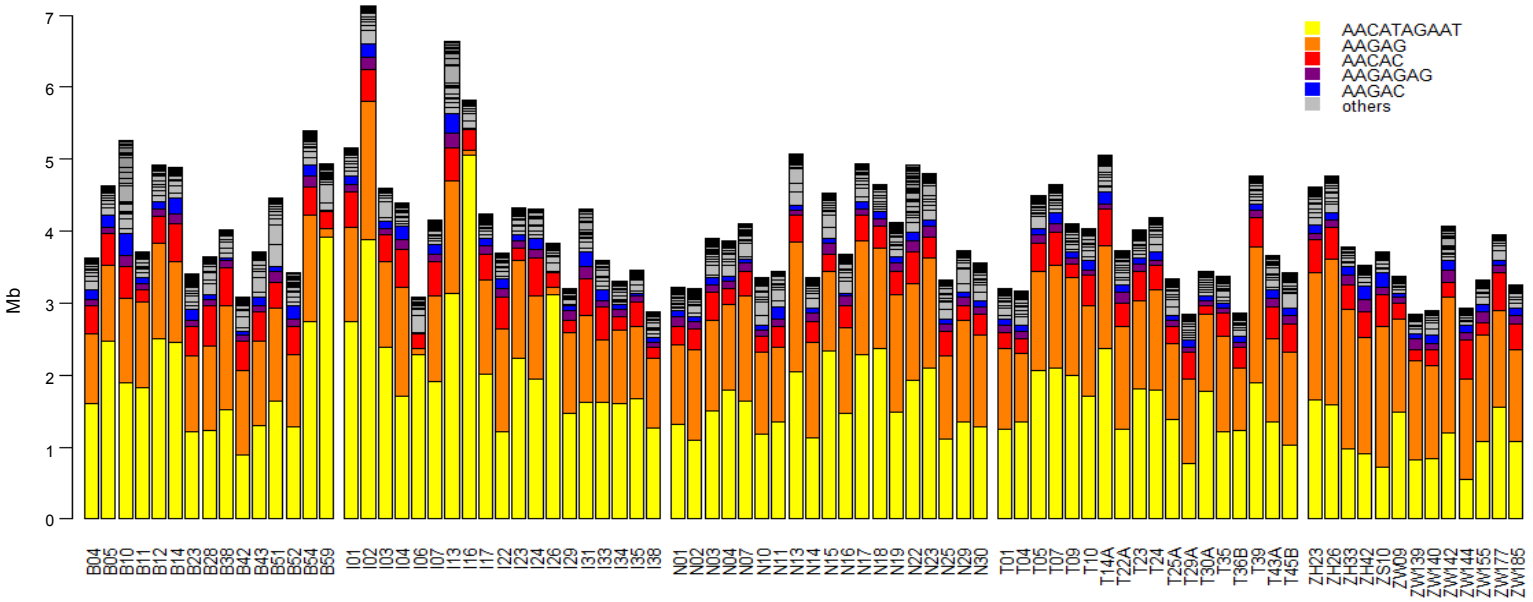
A

Pearson's *r*	rep1	rep2	rep3	rep4
rep1	1.000	0.364	0.042	0.411
rep2	0.364	1.000	0.169	0.279
rep3	0.042	0.169	1.000	-0.028
rep4	0.411	0.279	-0.028	1.000

B

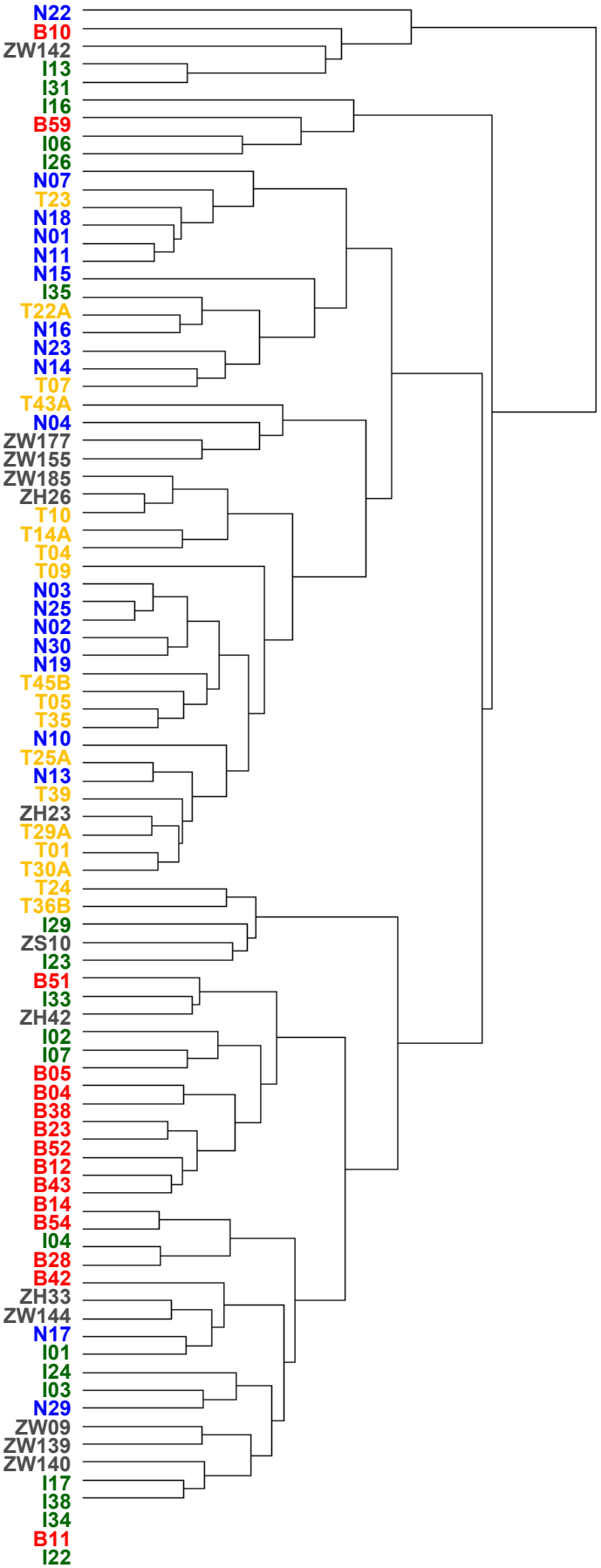


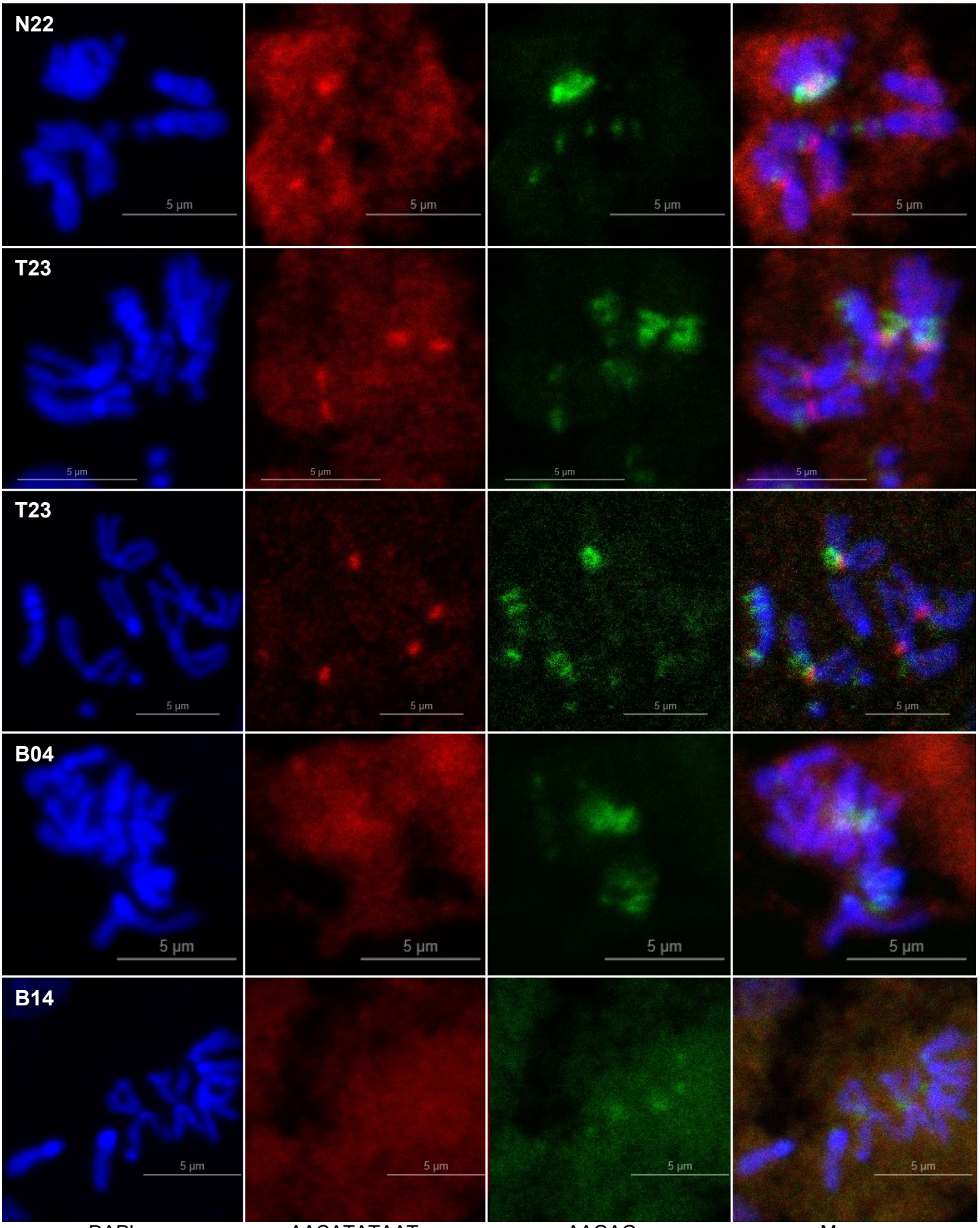
Supplementary Figure S3. Dot Blots with AAGAG probes. A. On each of the 4 membranes (rep1-4) independently hybridized, 86 different DNA samples were blotted randomly along with serial dilutions. Pair-wise correlation values of the normalized signal intensity is tabulated. B. 27 lines were randomly blotted in triplicates onto one membrane. Normalized signal intensity averaged across the triplicates is plotted against the normalized kmer count. No correlation was found.



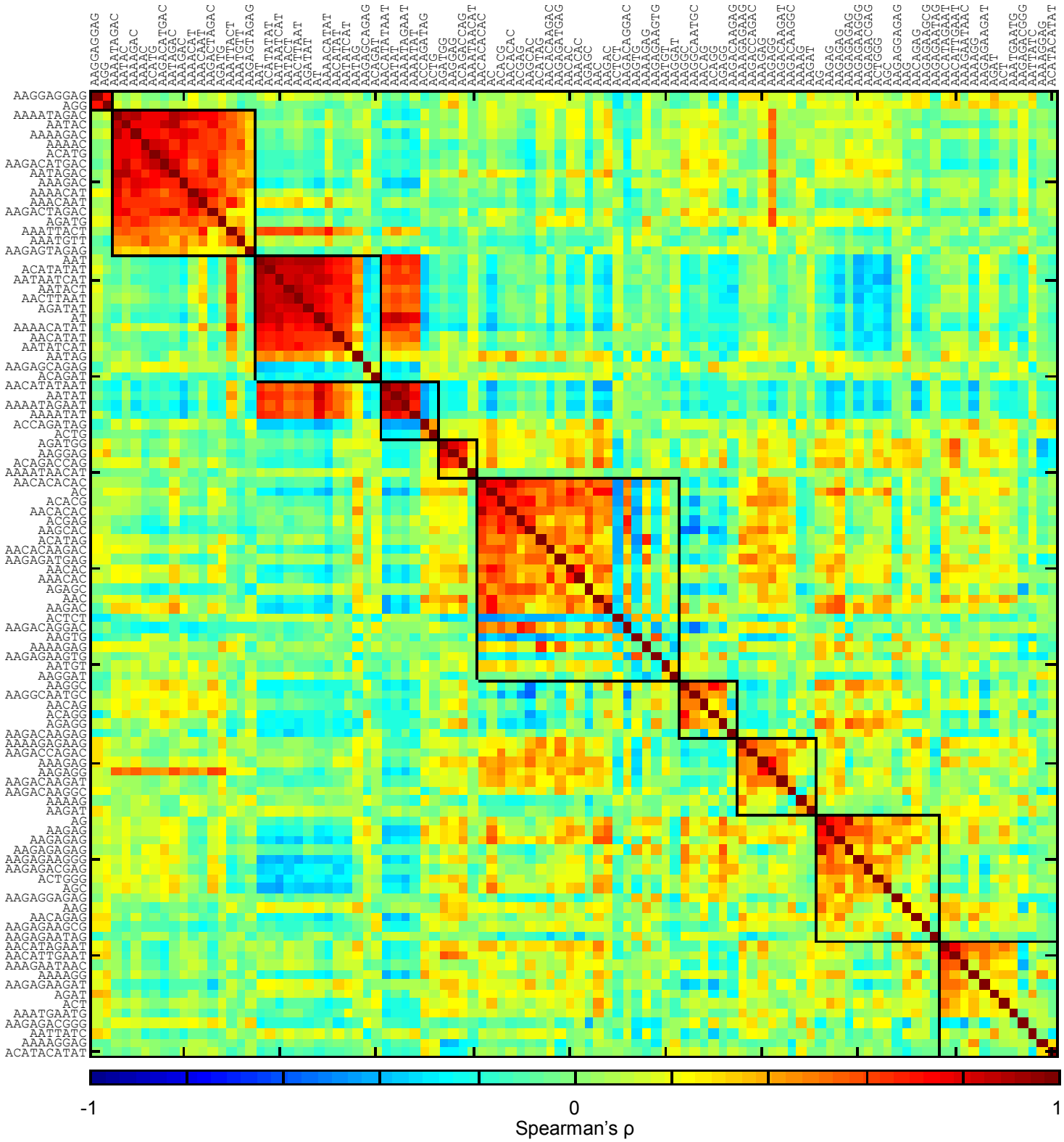
Supplementary Figure S4. Distribution of all kmers across lines.

Supplementary Figure S5. Hierarchical clustering of the lines. Lines are clustered based on euclidean distance. Euclidean distances between lines were calculated based on $\log_{10}$ kmer quantities relative to the mean. The top 100 kmers were used. The different populations are color coded: Beijing (red), Ithaca (green), Netherlands (blue), Tasmania (yellow), and Zimbabwe (gray).

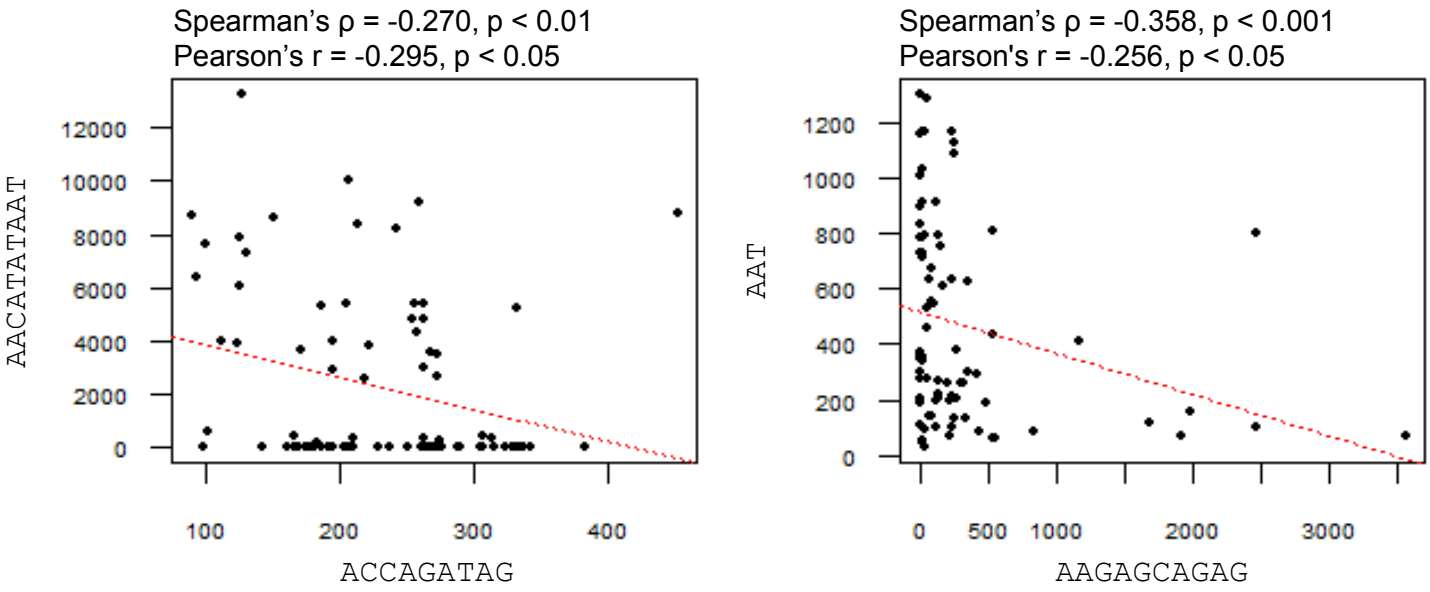




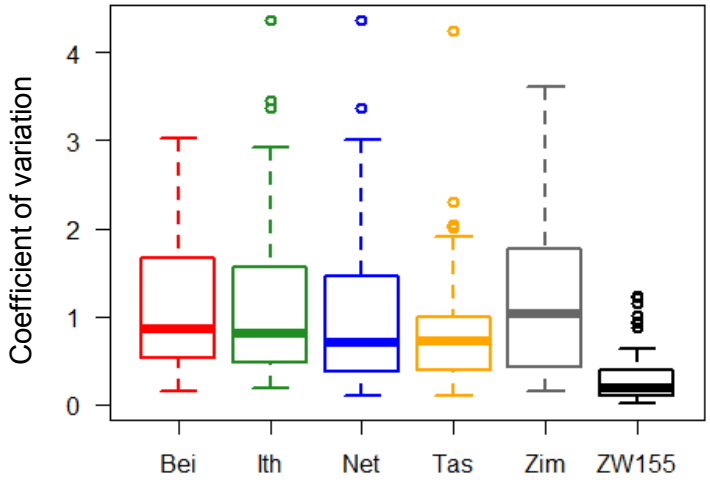
Supplementary Figure S6. Fluorescent in situ hybridization with probes targeting 5mer AAGAG and population-specific 10mer AACATATAAT.



Supplementary Figure S7. Clustered pairwise-correlation matrix of kmers across lines. Same as Fig. 4A but with all kmers labeled.



Supplementary Figure S8. Anti-correlated kmers. Spearman's ρ is calculated with all samples. Pearson's r is calculated after removing samples that have kmer quantities lower than 1% of the average. Red line depicts regression line using all samples.



Supplementary Figure S9. Distribution of coefficient of variation. The coefficient of variation for the top 100 kmers are plotted. The Zimbabwe population has the highest average coefficient of variation ($p = 0.0244$, Kruskal Wallis test.)