Large sets of edit-metric sequence identification tags to facilitate large-scale multiplexing of reads from massively parallel sequencing

Brant C. Faircloth[1] and Travis C. Glenn[2]

[1]Department of Ecology and Evolutionary Biology, University of California, Los Angeles, CA 90095 USA; [2]Department of Environmental Health Science and Georgia Genomics Facility, University of Georgia, Athens, GA 30602 USA

Correspondence: Brant C. Faircloth (brant.faircloth@gmail.com), 621 Charles E. Young Drive, Dept. of Ecology and Evolutionary Biology, University of California, Los Angeles, CA 90095 USA

Abstract

Background

Massively parallel DNA sequencing technologies provide exponential increases in the amount of data returned from the sequencing process, relative to traditional (Sanger-based) techniques. Use of unique, synthetic oligonucleotides (identifying sequence tags) on each sample enables the deconvolution of samples pooled prior to massively parallel sequencing. To counteract oligonucleotide synthesis and sequencing errors, sequence tags should be drawn from combinations with sufficient differences and with an appropriate error-correcting code over the alphabet [A,C,G,T]. This method ensures that errors within the tags do not cause sequences to be assigned to the wrong sample while also enabling correction and recovery of incorrectly-sequenced or incorrectly-synthesized tags. The set of available tags should be large, allowing sample multiplexing to scale with rapid changes in sequencing platform output. The set of tags should also account for errors possible during both the oligonucleotide synthesis and sequencing processes. Most tags in current use have been designed to maintain a particular Hamming distance: a scheme that ensures the distance between tag sequences is maintained in the presence of substitutions. However, sequence identification tags conforming to the edit-metric are more appropriate: edit-metric sequence tags are robust to insertions, deletions, and substitution errors.

Results

We present *eddittag*, a python package containing several tools to facilitate the design of edit-metric-based sequence identification tags, check existing sets of sequence identification tags for conformance to the edit-metric, and apply sequence identification tags to primers and/or adapters. We use *edittag* to design several large sets of edit-metric sequence tags ranging from four to 10 nucleotides in length and edit distance three to nine. Finally, we test a set of fusion primers designed with the software developed here, demonstrating high levels of successful amplification.

Conclusions

Researchers using sequence identification tags should consider using edit-metric-based sequence tags in place of the more common, Hamming-distance-based alternatives. Edit-metric sequence tags are robust to insertion, deletion, and substitution errors, thus more robust to all forms of error present during the DNA sequencing process. *Edittag* facilitates the generation and application of these robust sequence tags.

Background

Synthetic, oligonucleotide sequence identification tags or “sequence tags” can be attached to individual pieces of DNA allowing sample tracking during massively parallel sequencing (MPS). Sequence tags enable efficient distribution of the output from these platforms among smaller stretches of DNA from many individuals rather than extensive, deep sequencing across entire (2-3 Gbp) genomes of one to several individuals. Thus, the ability to tag and track sequenced DNA from many individuals in multiplex increases the efficiency of NGS when the genomes being sequenced are small [1] or when researchers want to apportion the output of NGS platforms among smaller genomic regions of many individuals [2-4].

Prior work introduced the idea of sequence tagging by incorporating tags to sequence reads using PCR primers and DNA ligation [5-7]. Yet, early sequence tags were designed with weak defenses against the effects of sequencing error on the uniqueness of each tag sequence. Sequencing errors occur on all MPS platforms, but the type of errors and the error rates vary across MPS platforms [8-15]. Sequencing errors on platforms from Roche 454 and Pacific Biosciences largely consist of insertion and deletion errors, whereas sequencing errors on platforms from Illumina and Applied Biosystems are generally substitutions [16]. Single-read sequencing error rates vary from 0.5 - 5% [9, 10, 14, 15] on Roche, Illumina, and Applied Biosystems platforms to 18% on the Pacific Biosciences platform [12].

Sequencing errors negatively impact the utility of sequence tags because they change the basepair composition of sequence tags by inserting bases to, substituting bases within, or deleting bases from sequence tags. Thus, sequencing errors can cause one tag to appear identical to another ("crossover") or sufficiently alter a sequence tag such that it is unrecognizable ("loss") and untraceable to the source material. A uniformly distributed error rate of 1.0% during an MPS sequencing run producing 10^6 reads, each having an 8 bp sequence tag, produces approximately 77,000 reads (8 %) having more than one error within the sequence tag (Figure 1). Probability ensures that longer sequence tags, which allow multiplexing of more samples, are affected by sequencing error to a greater degree.

Using error-correction schemes, researchers can construct sequence tags that are robust to sequencing errors (i.e., minimize crossover) while also allowing the correction of certain types of errors. Hamady et al. [17] used Hamming codes [18] to develop a set of error-correcting sequence tags which they used to successfully track a large number of reads in multiplex (see also [19]). However, Hamming codes assume that the errors occurring within each sequence tag are only substitutions [20, 21]. Thus, insertion and deletion errors violate the codeword scheme and reduce the utility of Hamming-based tags on sequencing platforms where sequencing errors may include insertions and deletions [22].

Sequence tags based on the "edit metric" or Levenshtein distance [23, 24] provide a more robust alternative usable across the entire group of MPS platforms, because edit-metric sequence tags are robust to the types of sequencing errors occurring on all MPS platforms:

insertions, deletions, and substitutions. Edit metric sequence tags allow for the correction of sequencing errors according to the following formulas [23-25]:

$$\text{Required Edit Distance} = 2 * (\text{Errors}) + 1$$

or

$$\text{Correctable Errors} = (\text{Edit Distance} - 1)/2$$

Thus, we can correct up to two sequencing errors in sequence tags from a set having an edit distance of five. Although edit-metric sequence tags are provided by several commercial (Roche 454, Inc.) and non-commercial sources [25, 26], there are few available methods (*c.f.* [27]) of generating sets of edit metric-based sequence tags. However, current methods may generate tags that do not correctly follow the edit-metric, and current methods are best suited to generating sequence tag sets comprising tags of shorter length (≤ 8 nt). The continually increasing output of MPS platforms suggests that large collections of edit metric sequence tags will be essential to distributing output across smaller genomes, select genomic regions, and populations of individuals.

Here, we introduce *eddittag*, a collection of tools for generating edit metric sequence tags, testing previously designed sequence tags for conformance to the edit metric, and programmatically applying sequence tags to amplification primers and platform-specific sequencing adapters. *Eddittag* differs from similar programs by providing: (1) a method to generate edit metric sequence tags of arbitrary length; (2) a method to check the conformity of previously designed tags, adapters, linkers, or primers to the edit metric; (3) methods for applying sequence tags to amplification primers and platform-specific sequencing adapters, and (4) multiprocessing support to speed tag generation when tag lengths are long (≥ 8 nucleotides).

Implementation

Eddittag provides Python (http://www.python.org) programs for designing edit metric sequence tags, testing tags for conformation to the edit metric, and convenience programs for applying sequence tags to amplicons or platform-specific sequencing adapters. We describe implementation details of these methods separately.

Sequence Tag Design

Technically, designing error-correcting sequence tags is a matter of generating all n-length combinations of [A,C,G,T]; removing those combinations containing homopolymer runs; removing combinations with GC content outside an appropriate range (approximately 40-60 %); removing tags that are perfect self-complements; and iteratively comparing each tag in the remaining group against all other tags in the remaining group to create the largest set that maintains some minimum edit distance.

Practically, the process is more complex because the creation of edit metric sequence tag sets requires comparison of all tags in the candidate set to all other tags in the candidate set. Given sufficiently long sequence tags, this requirement rapidly approaches the limits of desktop computation. For example, the full set of 10 nucleotide tags contains 1,048,576 members, which requires 550 billion pairwise edit-distance comparisons across all tags in the candidate set. If storage of each result requires 8 bits, then storing the entire array requires approximately 500 GB - a daunting object with which to work. Additionally, this considers only the first stage of processing and ignores the additional computational and storage overhead required to select and test subsets of edit metric sequence tags.

Thus, we modified the approach used by the lexicode algorithm [28] to speed processing, reduce memory consumption, and enable parallelization of jobs across multiple processors. Briefly, our approach generates all *n*-length combinations of sequence tags. Then, if the remaining group is sufficiently large, we apportion tags into discrete batches of 25,000 tags, and we distribute each batch among the available number of processing cores to (optionally) remove those tags having problematic composition (homopolymers, improper GC, perfect self-complements). After filtering, we rebuild the set of candidate tags returned from each processing core, and we create the following data structure, where the 0th position of each "row" below is the "key" to which we pair a "value" comprising a list of all tags:

```
(
        (tag0, [(tag0), (tag1), (tag2), (tag3)]),
        (tag1, [(tag0), (tag1), (tag2), (tag3)]),
        ...
)
```

If this data structure is sufficiently long (more than 500 members or "rows" as illustrated above), we apportion the tuple into batches containing 500 "rows", and we distribute each batch among the available number of processors. Iterating over each row, we then compute the edit distance between the "key" and all sequence tags in the value list using either a C-based Python module (http://pylevenshtein.googlecode.com) or a pure-Python method. To reduce memory consumption when iterating over millions of tags, we produce a summary vector for each key giving the count of all other sequence tags having values that fall within edit distance categories (0,1,2...,*n*), and we use the position of the count in the vector to denote the edit distance. Thus, the vector:

([1, 12, 124, 5])

corresponds to a key having a single tag edit distance 0 from itself (the tag compared to itself), 12 tags edit distance one from itself, 124 tags edit distance two from itself, and five tags edit distance three from itself. We then reduce the data by keeping only those keys having the maximum count of comparisons at the minimum desired edit distance, a technique that allows us to reduce the needed number of pairwise comparisons over the entire data set by

approximately 99% (estimated from the generation of eight nucleotide, edit distance three tags).

After reducing the data, for each key, we compute the edit distance between the key and all sequence tags in the value; we drop any tags in the value less than the desired edit distance; and we iterate over the remaining tags in the value, retaining only those tags that are also the desired edit distance from one another. Finally, we determine the count of remaining tags in the value list for each key, and we return the key (and its values) having the largest value list. Additionally, we include an option that quickly returns subsets of keys within this final set having edit distances from the key at values greater than the minimum desired edit distance.

We used this approach to design sets of edit metric sequence tags from four to 10 nucleotides in length and having edit distances of three. We used the shortcut method described above to select subsets, within each of these sets, having edit distances from four to nine.

Sequence Tag Checking

We provide a program for validating an existing set of sequence tags, primers, sequencing adapters, etc. for conformance to the edit metric by comparing each tag against all others and computing the minimum edit distance between members of the set. In short, the program iterates through the set of tags input; computes the pairwise edit distance between all tags in the set using either a C-based Python module or a pure-Python method; and outputs either the minimum distance of the set, those tag pairs having an edit distance less than the minimum expected, or the edit distance between all members of a set. This program is also capable of computing the Hamming distance between inputs in similar fashion.

We used this method to validate the edit distance between all members of each set of sequence tags generated above.

Sequence Tag Application

We provide two convenience programs for integrating sequence tags to platform-specific adapters and PCR primers. The first program simply integrates designed sequence tags to adapters and/or primers by inputting the list of desired sequence tags, the adapter/primer sequence 5' of the sequence tag location, and the adapter/primer sequence 3' of the sequence tag location. This program is largely meant to reduce mistakes when manually positioning sequence tags within large numbers of adapters and primers.

The second program is meant primarily for integration of sequence tags to PCR amplicons. In brief, this program adds sequence tags to the 5' ends of both upper and lower PCR primers, removes common bases between each sequence tag and primer sequence, optionally prepends both primers with a sequence (GTTT) promoting +A addition [29], uses primer3 [30] to evaluate tagged primers for complementarity issues and the presence of hairpins, and outputs all tagged primers to an sqlite (http://www.sqlite.org) database for subsequent evaluation and selection.

To test this design process and the resulting utility of PCR primers sequence tagged in this way, we integrated the entire set (n = 164) of 10 nucleotide, edit distance five sequence tags (Supplementary Table 1) to primers amplifying the rbcL locus in land plants [31, 32]. We used the resulting database to select five hairpin-free, sequence tagged primers which we used, along with an untagged primer set, to amplify the rbcL locus from 12 tropical forest tree species in a reaction mixture containing two uL CTAB-extracted DNA [33], 0.5 uM each primer, 1X BSA, 1X Standard (NEB) PCR Buffer, 0.5 U NEB Taq Polymerase and the following touchdown PCR thermal profile: 95°C for 30 s; 20 cycles of 95°C for 30 s, 60°C for 30 s minus 0.5°C per cycle, 68°C for 1.5 min; 20 cycles of 95°C for 30 s, 50°C for 30 s, 68°C for 1.5 m; 72°C for 15 min.

Results

We designed sequence tags of four to 10 nucleotides in length and having a minimum edit distance of three (https://github.com/baddna/edittag/downloads). Additionally, we selected all subsets of these sequence tags having edit distances at values greater than the minimum distance within each length category. Thus, we designed a large set (Table 1) of sequence tags ranging from four to 10 nucleotides in length and having edit distance three to nine. We validated the conformance of our sequence tags to the edit metric by analyzing all resulting tag sets using our method to compute the minimum edit distance between members of a given tag set. All tag sets we designed contained members having observed edit distances equal to or greater than the minimum expected edit distance.

We successfully amplified the *rbcL* locus using 10 nucleotide, sequence tagged primers in all 12 (100 %) tropical forest tree species (Figure 2).

Discussion

We designed a large set of edit metric sequence tags falling into several edit distance categories. Although the number of tags within each edit metric set is large, our methodology likely did not yield the largest potential set of edit metric tags for two reasons. First, given sufficient numbers of sampling draws, evolutionary algorithms are likely to produce larger sets of edit metric sequence tags relative to Conway's lexicode algorithm [22, 34]. However, the approach proposed by Ashlock et al. [22, 34] depends on a genetic algorithm that we felt would be slower and more computationally demanding than our lexicode-based approach when evaluating longer sequence tags. Additionally, evolutionary algorithms are sampling-based and unlikely to return identical tag sets across small numbers of runs, which may be problematic depending on the purpose of tag design.

Second, our computational shortcut to find tags of edit distance greater than the minimum desired increases speed, but it does not return the largest sets of edit metric sequence tags within these greater edit distance categories. One can easily maximize the size of each edit metric sequence tag set returned by running the program for only the tag length and minimum edit distance desired. We believe that this shortcut method is generally sufficient for most

applications, and the amount of computational time this approach saves is large, particularly when evaluating sequence tags over eight nucleotides in length.

Our testing of sequence-tagged amplicons suggests that the integration of primer design software to the amplicon-tagging process may increase the success of the tagging process by allowing researchers to avoid primers having potentially problematic secondary structures resulting from the placement of sequence tags. However, we realize that our sample of primers having integrated tags was small, and we recognize that additional trials using other primers in varied organisms will provide a better understanding of the utility of this approach.

Typically, we consider sequencing error as a single error term, identical to the approach we used in the simple models presented in Figure 1. It is important to remember, however, that sequencing error is actually a composite of several, different sources of error: errors arising during the sequencing process, errors arising during the sequence replication process (e.g. PCR), and errors arising during oligonucleotide synthesis. Each source of error has a potentially unique bias that contributes to the entire error term. For example, incomplete coupling during oligonucleotide synthesis results in a preponderance of deletion errors in the final oligonucleotide pool. Thus, even if a sequencing technology free from deletion errors is used, deletions will still be present in any sequence tagged data, and the presence of deletions violates certain code-word schemes. This observation suggests that we should use sequence tags robust to insertions, deletions, and substitutions, and that future work on the sources, rates, and effects of sequencing error across the MPS platforms would be desirable.

Conclusions

Synthetic, oligonucleotide, sequence identification tags provide researchers with the ability to tag and track DNA from many individuals in multiplex during MPS. As MPS technologies advance, it will be useful to have larger sets of sequence identification tags to efficiently distribute the output of these platforms among larger numbers of samples. Current sequence tags are often designed according to an error-correcting scheme that is robust only to substitution errors – a common form of error on MPS platforms. However, sequence tags designed using an alternative scheme, the edit-metric, offer more robust error-correction in the presence of insertions, deletions, and substitutions. We provide a software package, *edittag*, to: generate edit-metric sequence tags of arbitrary length, validate previously designed tags for conformance to the edit metric, and integrate edit-metric tags to PCR primers. We use our software to generate several large sets of edit metric sequence tags, which we make publicly available, and we validate a random set of edit-metric sequence tags incorporated to fusion PCR primers.

Availability and Requirements

We make all computer code discussed above available from https://github/baddna/edittag under a BSD-license. Documentation is available at http://baddna.github.com/edittag/. We have additionally prepared an Amazon Machine Instance for the Elastic Compute Cloud

(http://aws.amazon.com/ec2) ready for immediate tag design, analysis, and integration of tags to primers – please see http://baddna.github.com/edittag for the most up-to-date AMI number. We provide TXT, CSV files, and an sqlite database containing the edit metric sequence tags under a Creative Commons Attribution License from: https://github.com/BadDNA/edittag/downloads. *Edditag* has numpy (http://numpy.scipy.org/) as a dependency and two optional dependencies: py-levenshtein 0.10.1, which is available from http://pylevenshtein.googlecode .com under the GNU General Public License, and a modified version of primer3 2.2.3 (http://primer3.sourceforge.net), which is available from https://github/baddna/mod-primer3.

Authors' contributions

BCF and TCG were responsible for initially planning to generate large, edit metric sequence tag sets. BCF wrote all computer code, performed the tag validation, and conducted laboratory verification of amplicon tagging. BCF and TCG wrote the manuscript.

Acknowledgements and Funding

We thank PA Gowaty (PAG), SP Hubbell (SPH), and M Reasel for their support and comments on this manuscript.

Startup funds from the University of California to PAG and SPH, a Smithsonian Scholarly Studies Grant to SPH, NSF grant # DEB-0948585 to SPH, and NSF grant # DEB-0614208 to TCG provided financial support for this work.

References

1. Morin PA, Archer FI, Foote AD, et al. **Complete mitochondrial genome phylogeographic analysis of killer whales (Orcinus orca) indicates multiple species**. *Genome Research* 2010.
2. Jumpponen A, Jones KL: **Massively parallel 454 sequencing indicates hyperdiverse fungal communities in temperate Quercus macrocarpa phyllosphere.** *The New Phytologist* 2009, **184**:438-488.
3. Cummings N, King R, Rickers A, et al. **Combining target enrichment with barcode multiplexing for high throughput SNP discovery**. *BMC Genomics* 2010, **11**:641.
4. Price LB, Liu CM, Johnson KE, et al. **The effects of circumcision on the penis microbiome.** *PLoS ONE* 2010, **5**:e8422.
5. Binladen J, Gilbert MTP, Bollback JP, et al. **The use of coded PCR primers enables high-throughput sequencing of multiple homolog amplification products by 454 parallel sequencing.** *PLoS ONE* 2007, **2**:e197.
6. Meyer M, Stenzel U, Myles S, Pru K, Hofreiter M: **Targeted high-throughput sequencing of tagged nucleic acid samples**. *Nucleic Acids Research* 2007, **35**:1-5.
7. Meyer M, Stenzel U, Hofreiter M: **Parallel tagged sequencing on the 454 platform**. *Nature Protocols* 2008, **3**:267-278.
8. Margulies M, Egholm M, Altman WE, et al. **Genome sequencing in microfabricated high-density picolitre reactors**. *Nature* 2005, **437**:376-381.
9. Huse SM, Huber JA, Morrison HG, Sogin ML, Welch DM: **Accuracy and quality of massively parallel DNA pyrosequencing**. *Genome Biology* 2007, **8**:R143.
10. Bentley DR, Balasubramanian S, Swerdlow HP, et al. **Accurate whole human genome sequencing using reversible terminator chemistry.** *Nature* 2008, **456**:53-9.
11. Dohm JC, Lottaz C, Borodina T, Himmelbauer H: **Substantial biases in ultra-short read data sets from high-throughput DNA sequencing.** *Nucleic acids research* 2008, **36**:e105.
12. Chin C-shan, D P, Sorenson J, et al. **The Origin of the Haitian Cholera Outbreak Strain**. *South Asia* 2011:33-42.
13. Harismendy O, Ng PC, Strausberg RL, et al. **Evaluation of next generation sequencing platforms for population targeted sequencing studies.** *Genome Biology* 2009, **10**:R32.
14. Hillier LW, Marth GT, Quinlan AR, et al. **Whole-genome sequencing and variant discovery in C . elegans**. *Nature Methods* 2008, **5**:183-188.
15. McKernan KJ, Peckham HE, Costa GL, et al. **Sequence and structural variation in a human genome uncovered by short-read, massively parallel ligation sequencing using two-base encoding.** *Genome Research* 2009, **19**:1527-41.
16. Shendure J, Ji H: **Next-generation DNA sequencing.** *Nature biotechnology* 2008, **26**:1135-45.
17. Hamady M, Walker JJ, Harris JK, Gold NJ, Knight R: **Error-correcting barcoded primers for pyrosequencing hundreds of samples in multiplex**. *Nature Methods* 2008, **5**:2007-2009.
18. Hamming RW: **Error detecting and error correcting codes**. *Bell Systems Technical Journal* 1950, **29**:147-160.
19. Erlich Y, Chang K, Gordon A, et al. **DNA Sudoku--harnessing high-throughput sequencing for multiplexed specimen analysis.** *Genome research* 2009, **19**:1243-53.

20. Cole R, Gottlieb L-A, Lewenstein M: **Dictionary matching and indexing with errors and don't cares**. *Proceedings of the thirty-sixth annual ACM symposium on Theory of computing - STOC '04* 2004:91.
21. Stephen GA: *String Searching Algorithms*. River Edge, NJ: World Scientific Publishing; 2000:245.
22. Ashlock D, Houghten SK: **DNA error correcting codes: No crossover.** *2009 IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology* 2009:38-45.
23. Gusfield D: *Algorithms on strings, trees, and sequences*. Cambridge, UK: Cambridge University Press; 1999:535.
24. Levenshtein VI: **Binary codes capable of correcting deletions, insertions, and reversals**. *Soviet Physics-Doklady* 1966, **10**:707-710.
25. Qiu F, Guo L, Wen T-jung, et al. **DNA Sequence-Based "Bar Codes" for Tracking the Origins of Expressed Sequence Tags from a Maize cDNA Library Constructed Using Multiple mRNA Sources 1**. *Plant Physiology* 2003, **133**:475-481.
26. Adey A, Morrison HG, (No Last Name) A, et al. **Rapid, low-input, low-bias construction of shotgun fragment libraries by high-density in vitro transposition**. *Genome Biology* 2010, **11**:R119.
27. Meyer M, Kircher M: **Illumina sequencing library preparation for highly multiplexed target capture and sequencing.** *Cold Spring Harbor protocols* 2010, **2010**:pdb.prot5448.
28. Conway J, Sloane N: **Lexicographic codes: Error-correcting codes from game theory**. *IEEE Transactions on Information Theory* 1986, **32**:337-348.
29. Brownstein MJ, Carpten JD, Smith JR: **Short Technical Reports**. *Biotechniques* 1996, **20**.
30. Rozen S, Skaletsky H: **Primer3 on the WWW for General Users and for Biologist Programmers**. In *Methods in Molecular Biology*. edited by Krawetz S, Misener S Totowa, NJ: Humana Press; 2000, **132**:365-386.
31. Kress WJ, Erickson DL: **A two-locus global DNA barcode for land plants: the coding rbcL gene complements the non-coding trnH-psbA spacer region.** *PloS one* 2007, **2**:e508.
32. Kress WJ, Erickson DL, Jones FA, et al. **Plant DNA barcodes and a community phylogeny of a tropical forest dynamics plot in Panama**. *PNAS* 2009.
33. Doyle JJ, Doyle JL: **A rapid DNA isolation procedure for small quantities of fresh leaf tissue.** *Phytochemical Bulletin* 1987, **19**:11-15.
34. Ashlock D, Guo L, Qiu F: **Greedy Closure Evolutionary Algorithms**. In *Proceedings of the 2002 Congress on Evolutionary Computation*. Honolulu, HI: 2002:1296-1301.

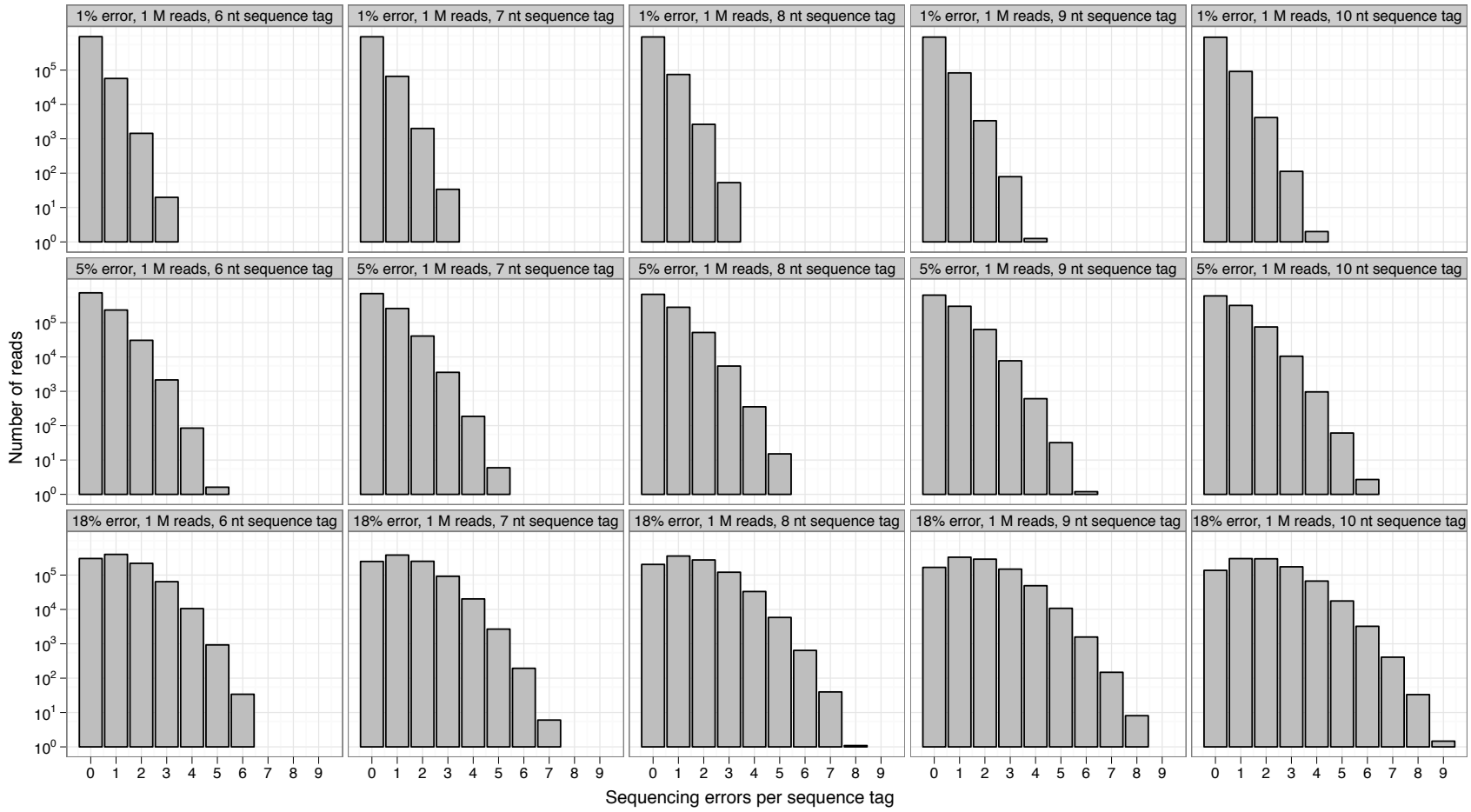

Figure 1 | Simulated count of reads containing barcode errors assuming uniformly distributed sequencing error rates of 1%, 5%, and 18%. Counts are the average across 1000 simulated runs, each returning 10^6 reads per run. We simulated reads making the simplifying assumption that sequencing errors are a Bernoulli process – thus, sequencing error across the sequence tag follows the binomial distribution where the barcode length is equivalent to the number of trials and sequencing error rate is equivalent to the probability of "success" in each trial. We used a single error term, which incorporates error from sequencing plus oligonucleotide synthesis.

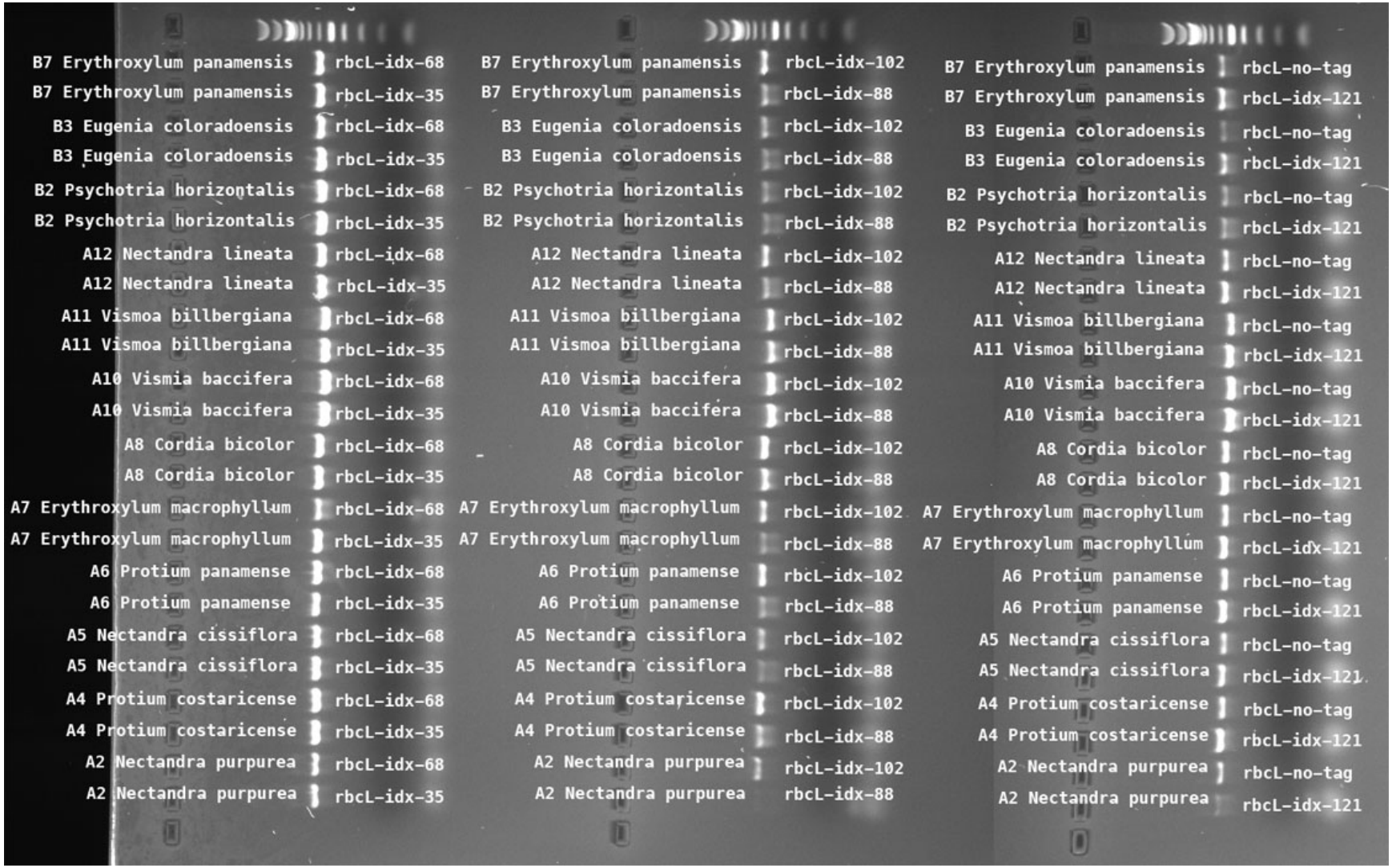

Figure 2 | Amplicons from the *rbcL* locus of 12 tropical forest tree species generated from five, randomly-selected, sequence tagged primers and one untagged primer [31, 32]. Sequence tags were 10 nucleotides in length and edit distance five.

Table 1 | Counts of four to 10 nucleotide, 3+ edit distance sequence tags sets designed using *edittag*. We did not include, in any set, sequence tags having >2 homopolymers, GC content outside the range 40 % < GC < 60 %, or perfect self-complementarity.

Code Sizes		Distance						
		3	**4**	**5**	**6**	**7**	**8**	**9**
Length	**4**	7	-	-	-	-	-	-
	5	25	7	-	-	-	-	-
	6	61	15	5	-	-	-	-
	7	211	41	11	4	-	-	-
	8	531	103	24	8	3	-	-
	9	1036	301	62	18	6	3	-
	10	7198	971	164	40	14	5	3

Supplementary Table 1 | Sequences of *rbcL* primers, modified to include 10 nucleotide, edit distance five sequence tags and a GTTT "pigtail". Underlined, italicized bases within the primer sequence indicate base-sharing between the locus-specific primer and the sequence tag.

Name	Upper (5' - 3')	Lower (5' - 3')
rbcL-idx-35	GTTTccatatgaacATGTCACCACAAACAGAGACTAAAGC	GTTTccatatgaacGTAAAATCAAGTCCACCRCG
rbcL-idx-68	GTTTaataccagg*a*TGTCACCACAAACAGAGACTAAAGC	GTTTaataccaggaGTAAAATCAAGTCCACCRCG
rbcL-idx-88	GTTTgttgccacctATGTCACCACAAACAGAGACTAAAGC	GTTTgttgccacctGTAAAATCAAGTCCACCRCG
rbcL-idx-102	GTTTccatcgcagtATGTCACCACAAACAGAGACTAAAGC	GTTTccatcgca*gt*AAAATCAAGTCCACCRCG
rbcL-idx-121	GTTTatcctaggcgATGTCACCACAAACAGAGACTAAAGC	GTTTccatatgaacgtAAAATCAAGTCCACCRCG