Facilitated sequence counting and assembly by template mutagenesis

Dan Levy[1] and Michael Wigler[1]

Simons Center for Quantitative Biology, Cold Spring Harbor Laboratory, Cold Spring Harbor, NY 11724

Contributed by Michael Wigler, August 22, 2014 (sent for review June 9, 2014; reviewed by Bud Mishra and Leonid Kruglyak)

Presently, inferring the long-range structure of the DNA templates is limited by short read lengths. Accurate template counts suffer from distortions occurring during PCR amplification. We explore the utility of introducing random mutations in identical or nearly identical templates to create distinguishable patterns that are inherited during subsequent copying. We simulate the applications of this process under assumptions of error-free sequencing and perfect mapping, using cytosine deamination as a model for mutation. The simulations demonstrate that within readily achievable conditions of nucleotide conversion and sequence coverage, we can accurately count the number of otherwise identical molecules as well as connect variants separated by long spans of identical sequence. We discuss many potential applications, such as transcript profiling, isoform assembly, haplotype phasing, and de novo genome assembly.

mutational tagging | expression profiling | copy number variation

Some problems in genomic analysis have remained difficult despite the development of high throughput sequencing methods. Many of these problems arise from the inability to distinguish identical and nearly identical template sequences. Counting molecules of identical composition in an RNA sequencing assay or the copy number of identical stretches of DNA currently depend on quantitative methods that adjust imperfectly for the distortions of data caused by sample processing. Moreover, because read lengths are short, determining the physical connection of distinguishable elements separated by long identical stretches is difficult to impossible and limits our ability to phase single nucleotide variants (SNVs), identify transcript isoforms, and assemble through repetitive genomic regions. We propose a method that solves these problems by randomly mutagenizing the original template molecules. Each template thus bears a unique signature that is imprinted on all of its subsequent copies and the fragments of those copies. Counting molecules becomes a matter of counting unique mutational patterns and assembly a matter of connecting reads with overlapping mutation patterns.

Modifying molecules to facilitate counting is not a new idea. There are several protocols in which a sequence of random nucleotides is appended to the template molecules before amplification and sequencing. This idea has been applied under a variety of names to identify PCR duplicates (1, 2), improve counting of DNA (3, 4) and RNA (5–7) templates, and reduce sequence error (8–10). Each implementation has its own name for the random nucleotide sequences, and we refer to them as varietal tags (11). Counting varietal tags serves the same role as counting unique mutational signatures, mitigating the effects of amplification bias. The advantage of tagging over mutation is that the original message is completely recoverable. The disadvantage is that the tag is confined to one end of the molecule such that identity and count can only be distinguished within one read length of the ends. Further, only reads that include the tag are useful in determining count, and varietal tags provide no solution for assembly and assortment.

Earlier work advocated for the use of PCR mutagenesis as an aid to sequence assembly of regions that are resistant due to base composition or repeat structure (12–14). The method relied on the misincorporation of artificial nucleotides during many rounds of amplification, then sampling the resultant mixture by expanding a set of distinct clones for separate sequencing. Implementation of this method requires many steps and is not amenable to a high throughput process. Even if such technical demands are overcome, as has been proposed (15), mutagenesis during amplification loses track of template count.

In the following, we demonstrate by simulation that template mutagenesis can solve the problems of both counting and assembly. For any application, the order of operation is mutagenesis first, followed by short- or long-range PCR, then fragmentation, if needed, and preparation of sequence libraries. We explore two classes of applications. The first is counting specific DNA or RNA molecules, for assessing genome copy number or profiling a transcriptome (Fig. 1). The second is sequence assembly—for example, establishing haplotypes or distinguishing transcript isoforms (Fig. 2).

Our simulations model partial bisulfite mutagenesis of single-stranded DNA (16): Each mutable position (or "bit") converts (or "flips") independently from a wild-type state to an altered state with a fixed probability (or "flip rate"). We simulate performance under a variety of reasonable parameters for read length and mutation rate and over a range of template lengths and counts. We present the results under an assumption of complete coverage to obtain a theoretical upper limit of performance and then consider the consequences of sampling to various levels of coverage. In our simulations, mutable positions are distributed uniformly throughout the template such that each read contains the same number of bits (or "bit length"). We do not presently incorporate either sequence or mapping error. Variations to these assumptions and procedures are addressed in *Discussion*.

Results

Application 1: Counting Templates. Our first class of applications focuses on the general problem of determining absolute template

Significance

Despite vast improvements in DNA sequencing, many problems of interpretation arise when trying to count or assemble molecules (templates) that are largely identical. We show through simulations that by randomly mutagenizing DNA templates before amplification, many of these problems are resolved. We can obtain more accurate counting of templates by observing the number of unique patterns. By introducing distinctive patterns onto otherwise identical spans, we enhance our ability to correctly assemble sequences. This idea can be implemented with currently available mutagenesis protocols and such techniques can have applications in RNA expression analysis, haplotype phasing, copy number determination, and genome assembly. Template mutagenesis solves counting problems and effectively generates long reads from short-read sequence output.

Author contributions: D.L. and M.W. designed research; D.L. performed research; D.L. analyzed data; and D.L. and M.W. wrote the paper.

Reviewers: B.M., New York University; and L.K., Princeton University.

The authors declare no conflict of interest.

Freely available online through the PNAS open access option.

[1]To whom correspondence may be addressed. Email: wigler@cshl.edu or levy@cshl.edu.

This article contains supporting information online at www.pnas.org/lookup/suppl/doi:10.1073/pnas.1416204111/-/DCSupplemental.

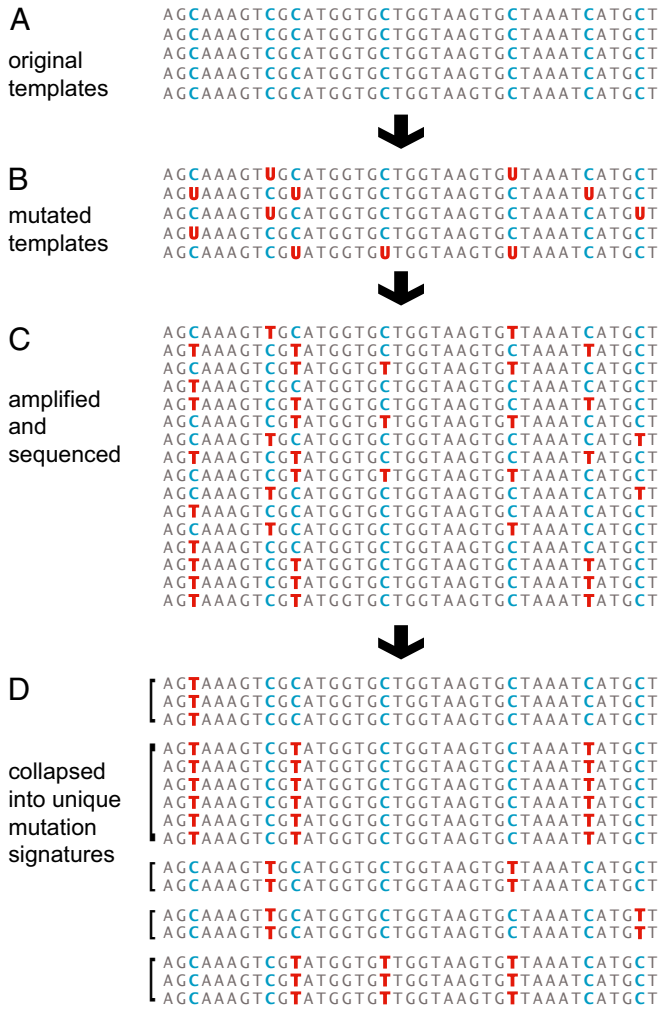

Fig. 1. Enumerating templates over a fixed window. We illustrate the process of counting indistinguishable template molecules (*A*). In the first step, we mutate the molecules by a random process, creating a unique mutation signature to each molecule (*B*). We illustrate here cytidine deamination, a mutational method that converts cytidine (blue) to a uridine (red). Upon amplification, uridine becomes a thymidine (*C*). The sequenced nucleotide strings are aligned, aggregated by their mutational patterns (*D*), and the number of the distinct patterns counted.

count. This is important for determining the copy number of genomic DNA, measuring mRNA expression levels, quantifying allele bias, and detecting somatic mutations. To obtain an accurate count, the protocol requires mutagenesis before amplification. Amplification could be either short- or long-range PCR but must occur before fragmentation if needed for library preparation. The number of possible mutation patterns should exceed the template number to obtain the most accurate count. Hence, we only consider cases where the absolute template count is below the low thousands and save the other cases for *Discussion*.

We begin with the simplest formulation, the case where the templates span a single fixed read length and we have exhaustive coverage (Fig. 3*A*). In this case, our estimate of template count is merely the number of distinguishable mutation patterns observed. The number of possible patterns depends on the number of bits per read, and the probability of observing a given pattern depends also on the flip rate. The optimal flip rate for generating distinct mutational patterns is 0.5, wherein every pattern is equally likely. However, for a window of at least 20 bits, corresponding to a read length of 80 base pairs, a rate of 0.25 is still virtually perfect for template counts in the thousands. Similar efficacy is obtained at a flip rate of 0.15 for a 30-bit window. Templates numbering in the thousands are adequate for genome copy number determination or single-cell transcript profiling. In Dataset S1, we provide code that allows the user to simulate performance under a variety of conditions.

We also demonstrate the recovery of template count subject to varying depth of coverage for a fixed flip rate of 0.35 (Fig. 3*B* and Dataset S2). For each simulation, we mutate T initial template molecules, creating a pool of patterns. At a coverage level of c, we select $c \times T$ patterns with replacement from the pool and record the number of distinct patterns. For high coverage, on the order of 4× per original template molecule, recovery of count is nearly perfect. At lower coverage, the count is underestimated, the inevitable consequence of undersampling. These simulations assume uniform PCR amplification and provide an upper bound on performance.

More often, the length of the template will exceed a single read length. In this case, we do not have a fixed window over which to count distinct mutational patterns. So instead of looking for the number of unique (and hence incompatible) patterns over a fixed window, we seek to find a maximal set of pairwise incompatible reads. This problem can be precisely stated in the language of graph theory. More specifically, we treat reads as vertices and connect them by directed "compatible" edges whenever two reads can derive from the same template. The direction of the edge is determined by the orientation of the reference template. The result is a directed acyclic graph. By Dilworth's theorem (17), the size of the largest set of pairwise incompatible reads is the same as the minimum number of paths covering all vertices of this graph. By Kőnig's theorem (18), this is a problem of maximal matching in a bipartite graph, which is computationally tractable and solvable in polynomial time with the Hopcroft–Karp algorithm (19). Details and code are in *Methods* and Datasets S3–S5. In Fig. 3*C*, we show the results for a fixed mutation rate of 0.35, a template that spans 16 read lengths, for depths of coverage ranging from 1× to 5× per template, and for template counts ranging from 2 to 1,024. Under similar conditions of flip rate and coverage, counting over long templates is comparable to performance in counting over a fixed window, as described above.

The simulations of this section provide guidelines for (*i*) genome-wide copy number determination, (*ii*) transcript profiling, and (*iii*) determining allelic ratios. To determine the copy number, we measure the ratio of count for a given locus to the median count over the remainder of the genome. For transcript profiling, we measure the proportionate counts of each gene transcript. To determine allelic imbalance, we measure the ratio of counts from templates distinguishable by at least one SNV. In the context of RNA, this also enables observation of biased allele expression resulting from chromosome inactivation, imprinting, and the like.

Application 2: Assembling Templates. The second class of applications is to correctly assemble reads by their mutation patterns to recover the proper end-to-end sequence of nearly identical templates, desired when determining haplotype phasing or enumerating transcript isoforms. We consider long templates each tagged uniquely at both ends and simulate the more general task of determining how many initial templates can be correctly assembled from end to end from the mutation pattern alone (Fig. 2).

Following mutagenesis, amplification, fragmentation, and sequencing, we connect reads with overlapping mutation signatures to assemble a path from one tag to the other. Whereas in the previous application we allowed all compatible edges between reads, for this problem we build a subgraph with only the "best edges" between overlapping reads. A pair of tags is "exactly matched" if there is a path in the subgraph that connects them and neither tag is connected to another tag. Such a path is called an "exact path." If two tags originate from the same template, they are a "true match." A "true path" is an exact path for which every read originates from the same initial template.

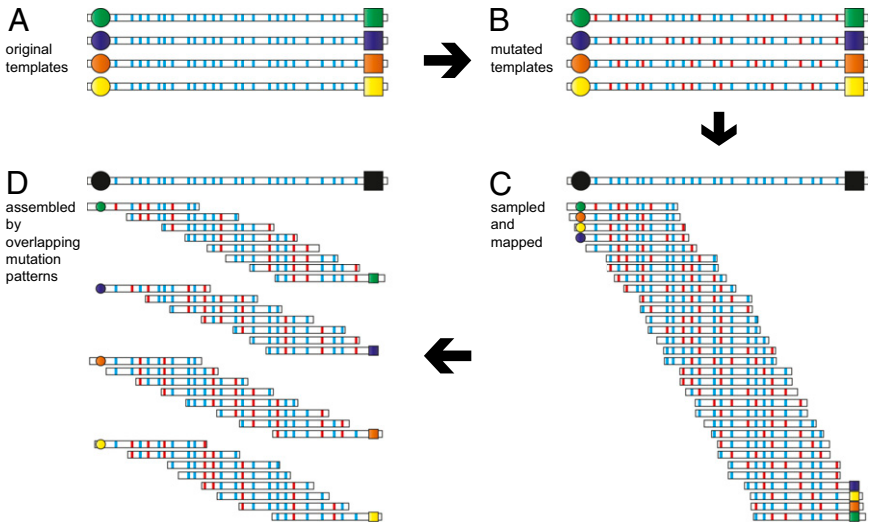

Fig. 2. Assembling long templates. We illustrate the process of connecting distinguishable markers separated by a long span of an indistinguishable sequence. In this example, each template has a unique pair of markers, or "end tags," denoted by the colored circle and square (*A*). Markers of the same color occur on the same template strands and are said to be "in phase." Blue marks on the templates show positions that may mutate. We subject each template to a random mutation process (*B*) that converts some of the positions (converted positions shown in red). After amplification and sequencing, each read is mapped to the reference template (*C*, top strand). Finally, by matching mutation patterns from overlapping reads, we recover the phase of the original templates (*D*).

Determining performance for the general task provides a lower bound on performance for other applications, because if there is an exact path that is also true, then all sequence information for that template was correctly observed. This includes haplotype phasing in the case of genomic data and transcript structure in the case of RNA profiling. In fact, these two applications are less demanding than the general task because there will only be a few template varieties and each template variety provides additional sequence information for distinguishing them.

We first consider the case of exhaustive coverage to establish the best performance we can expect for a given set of conditions. In this case, the best edges are naturally defined as the set of compatible edges that overlap for all but one bit. In Fig. 4*A*, we explore the effect of flip rate on template assembly. At a flip rate of 0.35 and 30 bits per read, performance on 32 template

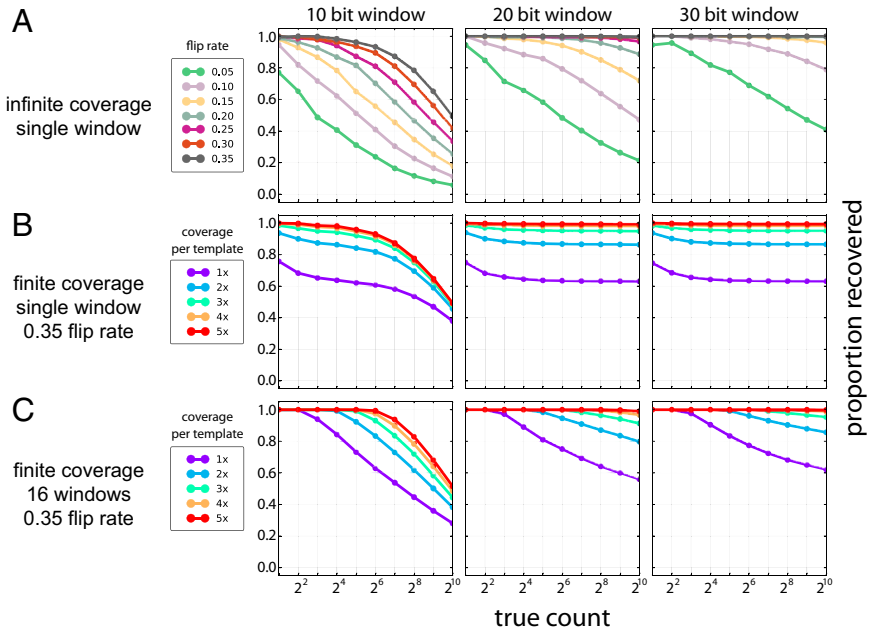

Fig. 3. Recovering counts. The ability to recover the template count is a function of the window size, template length, flip rate, number of templates, and depth of coverage. We show simulation results of template count estimation under a variety of conditions. Each panel has three plots, for window sizes of 10, 20, and 30 bits. The *x* axis shows the true count from 2 to 1,024 ($\log_2$ scale), and the *y* axis shows the average estimated count divided by the true count, or the proportion of templates recovered. *A* simulates recovery when the template is one window long for a range of flip rates for infinite coverage. *B* shows the results from one window template under finite coverage (1× to 5× reads per template) for a fixed flip rate of 0.35. *C* repeats the results of *B* for long templates comprised of 16 read lengths.

molecules is nearly perfect for even the longest span we tested, 2^{10} read lengths, in excess of 100 kilobases.

With exhaustive coverage, best edges are unambiguously defined and have the property that exactly matched pairs and exact paths are also true. However, for finite coverage, there is no natural definition of best edges and exactly matched pairs and exact paths may not be true. For finite coverage, we assign weights to edges by likelihood and use a greedy algorithm to select the best subgraph (detailed in *Methods* and illustrated in Fig. 5). Fig. 4*B* explores the effect of coverage (2× to 14× per template) on recovery of exact matches as a function of template length (2–1,024 read lengths), for 32 templates, a 30-bit read length, and a flip rate of 0.35. In Fig. 4*C*, we fix the template length to 32 read lengths to show recovery as a function of the number of templates, from 2 to 1,024 under the same conditions. Because these are simulations and we know the ground truth, we determine what proportion of exact matches are true and false and show these numbers. All exact matches are connected by an exact path, and for the conditions explored here, virtually all true exact matches are connected by a true path. At the maximum level of coverage, we find nearly all of the true paths even for spans of 2^{10} read lengths. For haplotype phasing, it is sufficient to have a single true path, and this is attainable with high probability at a lower coverage, 10× per template.

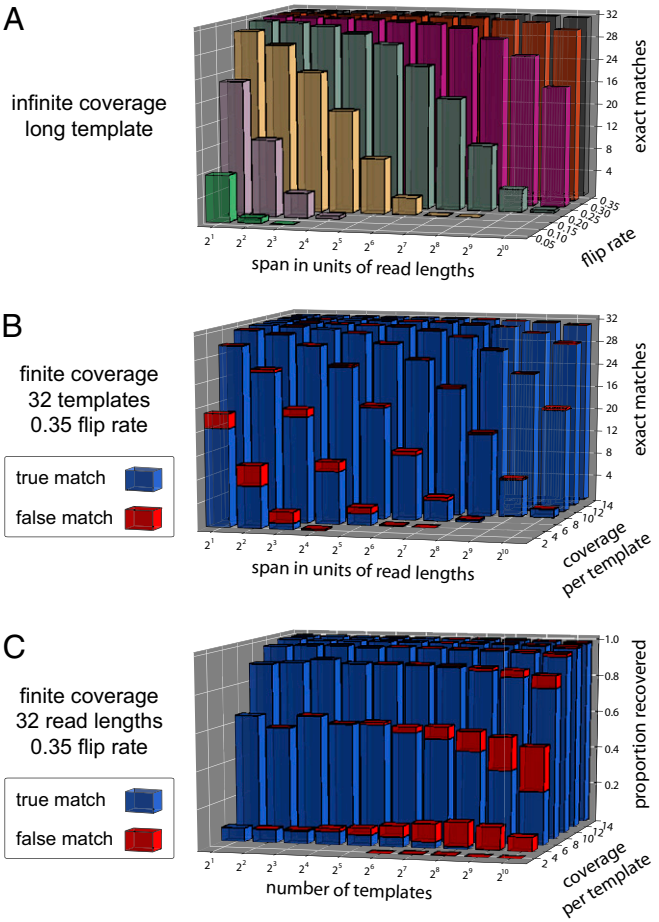

Fig. 4. Recovering phase. Proper phasing of end tags separated by a long span of an identical sequence depends on window size, template length, flip rate, number of templates, and depth of coverage. In *A* and *B*, we show simulation results for the recovery of phase for 32 templates across a span ranging from 2 to 1,024 read lengths for a 30-bit window. In *A*, we assume infinite coverage and examine recovery as a function of flip rate. Because coverage is exhaustive and the assembly graph is well characterized, there are no errors and all phased results are correct. In *B* and *C*, we consider the case of finite coverage, fixing the flip rate at 0.35 over a range of coverages from 2× to 14× per template. In *B*, we fix the number of template molecules at 32 to explore the effect of template length on recovery of phase. In *C*, we fix the template length to 32 read lengths to explore the effect of the number of templates. Because we do not observe every read, we use a greedy algorithm that may return exact matches that are correct (blue) or incorrect (red).

Discussion

We present here a feasibility study and guide for template mutagenesis as an enhancement to sequence-based analysis. Such methods introduce random mutations to create distinguishable patterns in previously identical or nearly identical templates. These patterns are inherited in copies of the template, and portions of patterns remain in fragmented copies. Provided these fragments overlap and there is sufficient diversity in mutational patterns over a read length, the structure of each original template can be inferred, thus overcoming the loss of connectivity resulting from short read lengths. The accuracy of template counting also improves, undistorted by biased amplification. We simulated the applications of this process under assumptions of error-free sequencing, perfect mapping, uniformly distributed reads, and uniform bit distribution. Although these conditions will never be met in practice, the simulations suggest that within readily achievable conditions we can hope for accurately counting the number of otherwise identical molecules, as well as connecting variants separated by long spans of identical sequence. There are many potential applications, ranging from transcription profiling and isoform assembly to haplotype phasing and de novo genome assembly.

To illustrate one application of our results, consider characterizing single-cell gene expression. A mammalian cell has ∼350,000 mRNA transcripts with an average length of 1,500 nucleotides (20). Single-strand cDNA would be randomly mutagenized using cytidine deamination before amplification, fragmenting, and sequencing. Assuming uniformity of sequence coverage and uniformity of PCR amplification, 9 million 100 base pair paired-end reads yields an average of 4× coverage per template. Given the typical distribution of cytidine in mammalian genomes, 30 is a safe estimate for the number of mutable positions in a read pair. From Fig. 3*C*, it is clear that we can count RNA templates with near perfect accuracy for mRNA species of intermediate to scarce expression (<1,000 copies per cell).

Furthermore, many mammalian genes are alternatively spliced, resulting in a diversity of isoforms observable as different patterns of exon inclusion in the mRNA. At a read depth of 10× per template, or 23 million reads, we can count not only transcripts per gene but also expression at the level of individual isoforms: Even without the additional information gained by observing alternative splice junctions, we can assemble a true path from one end of the molecule to the other. This can be accomplished for all but the most abundant genes and very long transcripts (greater than 6,000 nucleotides; Fig. 4*C*). In contrast, varietal tagging can achieve accurate counting of gene transcripts, even long and abundant ones, but it is limited to labeling the end of a molecule and so does not allow counting of isoforms or observing sequence variants, except near the ends of transcripts. The two methods, varietal tagging and mutagenesis, can be seamlessly integrated, achieving the benefits of both methods.

Another direct application of template mutation is discriminating haplotypes in an individual. Using existing short-read technology, we can readily identify many of the heterozygous positions; however, because the polymorphisms are typically more than one read length apart, we are unable to determine the proper phasing of alleles. We propose the following procedure for phase recovery: We first perform partial deamination on high-molecular-weight DNA with a 0.35 flip rate, then dilute it to 30 initial templates per region for each of the two strands. We then amplify with randomly primed PCR and fragment as needed for preparing sequencing libraries. Under the assumption of uniformly distributed coverage (Fig. 4*B*), we observe that coverage of 10× per initial template would be sufficient to

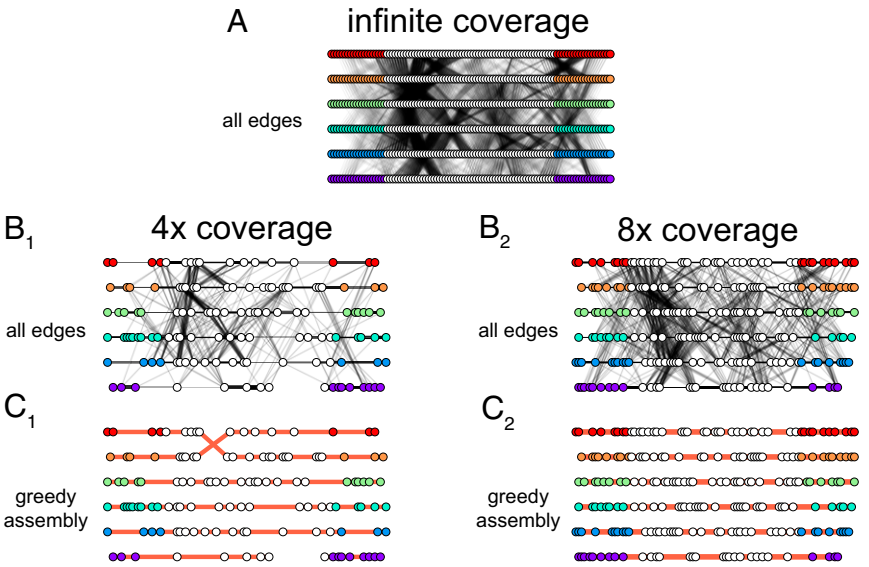

Fig. 5. Greedy path assembly. To carry out the pattern assembly shown in Fig. 2, we first define a graph in which each read is vertex. Some reads contain their end tag (colored circles) and some do not (white circles). We connect two reads with an edge if they agree on their overlap. The weight of an edge reflects the strength of that overlap. *A* depicts the template information assuming exhaustive coverage, drawing all distinct reads and the edges between them. In B_1 and B_2, we finitely sample reads from the templates at a depth of coverage of 4× and 8× per template, respectively. From this information, we then apply a greedy algorithm (C_1 and C_2) to select the best edges, shown in red. When coverage is low (C_1), some paths do not succeed in spanning the length of the template, and of those that do, three determine the correct phasing and two are in error. Under higher coverage (C_2), all paths span the template and all six are correctly phased.

recover phase information for variants separated by hundreds of kilobases.

Further, our ability to establish phase by this method depends on strong concordance between the haplotypes and the reference genome. For those regions where the reference genome is a poor match, due to repeat content, large-scale rearrangements, or novel sequence, the mutation pattern assembly algorithm will fail to generate consistent end-to-end assemblages. Although this presents a problem for direct inference by reference-matched phasing, it provides an opportunity for de novo haplotype assembly. The SUTTA algorithm (21) assembles haplotypes from short-read data by scoring proposed local assemblies based on orthogonal data sources, such as coverage, mate pairs, or physical maps. Template mutagenesis can help. Each local reference genome that SUTTA considers can also be assigned a score based on the number of successful end-to-end mutation pattern assemblies over the region. The result would be a de novo assembly over the human genome for those difficult regions.

In our simulations, we focus on a specific form of mutation: the conversion of cytidine to uridine by deamination. This conversion can be achieved either chemically through bisulfite treatment (16) or enzymatically through activation-induced deaminase (22). One advantage of deamination is that conversion patterns are predictable. Moreover, because bisulfite treatment is widely used in DNA methylation assays, the computational tools for mapping deaminated sequence reads are readily available. Still, other methods of mutagenesis, such as depurination, transposition, alkylating agents, or inducing replication error in the first template copy, might be useful in some contexts.

In our simulations, we assume perfect mapping. In practice, however, our ability to map reads is somewhat degraded by template mutation. For a deamination protocol, a standard practice is to map to a reduced alphabet where all Cs are converted to Ts in both the read and the reference, with two distinct references genomes for each DNA strand (23, 24). Clearly, restricting to a smaller alphabet and doubling the reference genomes impacts our ability to unambiguously map reads, however the effect is surprisingly mild (25). If increased mapping efficiency is needed, we can augment our mapping algorithm with a probabilistic model of the flip rate to prioritize the most likely alignments.

In our simulations, we assume no sequence error, but methods will be necessary for handling these. Aside from its effects on mapping, sequence error may reduce our ability to recover mutation patterns in those cases where the error appears to flip a bit or reverse a flipped bit. Fortunately, sequence error is typically rare. Within a reasonable range for flip rate, window size, and template count, we can expect sparse mutational patterns, well separated so that no two patterns are very much alike. Sequence error will produce a pattern "nearby" an established pattern, and less well covered, and we can use this signature to discount those reads.

Our simulations demonstrate that most applications work best for a low initial template count, less than a few thousand. This is not a problem for many genomic applications and is close to ideal for single-cell RNA analysis. If analysis of greater numbers of template molecules is desired, for example during analysis of bulk mRNA, then after mutagenesis of the first strand of cDNA, multiple separate amplifications reactions can be performed, each with low template count. The products of each reaction can be tagged with barcodes, pooled, and sequenced.

Methods

To enable a wide range of simulations, we developed a library of Python programs (Datasets S1–S7) to simulate mutation, sequencing, counting, and assembly of distinct templates under the assumptions of error-free sequencing, perfect mapping, and uniformity of mutation sites, mutation rate, sequence coverage, and DNA amplification. Our ability to recover template count and assembly depends on the depth of read sampling, typically called "coverage." Coverage usually means the average number of reads overlapping a position in the reference genome, however as we use it, coverage means the average read depth over a position per template.

To measure template count over a single fixed read length for T templates, we first simulate mutation to generate T random template patterns with R bits and a flip rate of p. To measure recovery under infinite coverage, we count the number of distinguishable mutation patterns observed (Dataset S1). To measure recovery under finite coverage, we generate reads by sampling a fixed number of patterns with replacement from the pool before counting distinguishable patterns (Dataset S2). Amplification distortion

could be modeled by altering the sampling procedure; however, our results are restricted to the case of uniform sampling. Our simulations explore mutation rates in the range of 0.05–0.35, template counts in the range of 2–1,024, and read lengths with 10, 20, and 30 mutable bits. For each condition, we perform 100 simulations and record the average recovery under infinite coverage (Fig. 3*A*). For each of those hundred simulations, we perform 100 finite samplings for each coverage level of 1–5× per template and record the mean count observed. Shown in Fig. 3*B* are the results for the mutation rate of 0.35.

When the template length exceeds a single window, we use a few simple graph formalisms. We say that two reads "conflict" if they map to overlapping positions and their overlaps fail to agree. We define a "compatible read graph" to be one in which each distinguishable read is treated as a "vertex" and two vertices are compatible and connected by an "edge" if the reads could have originated from the same template. In the case of finite coverage, two reads are compatible if they do not conflict. In the case of infinite coverage, there is a stronger constraint: Two reads at a distance d mutable bits apart are compatible if we observe all $d - 1$ distinct, nonconflicting, intermediate reads (Dataset S3). All edges inherit a direction from the orientation of the reference template such that each vertex has in-edges extending from one end of the template and out-edges extending toward the other end.

A path in this graph represents a possible partial assembly of an initial template pattern. Consequently, determining the minimum number of templates needed to explain all of the reads is achieved by finding the minimum number of paths such that every vertex in the graph is included in at least one path. This is known as the minimum vertex cover and in general is an NP-hard problem. However, under the assumption of perfect read mapping to a linear genome, our graph is not only directed but also acyclic. By Dilworth's theorem, the minimum number of covering paths is equivalent to the maximum number of elements in an antichain (17). In other words, the minimum number of templates needed to explain the reads is equal to the maximum number of reads that are pairwise incompatible. Using Kőnig's theorem (18), we solve this problem by finding a maximal matching in a new bipartite graph constructed by splitting each vertex in two (an "in" vertex that receives in-edges and an "out" vertex that receives out-edges). Finding a maximal matching in a bipartite graph is then solvable in polynomial time by the Hopcroft–Karp algorithm (19).

As in the case of a fixed read length, we first simulate mutation to generate a pool of template patterns. For a finite coverage level of c, read length R, and T initial templates of length L, we generate $c \times T \times (L/R)$ reads by drawing uniformly with replacement from the set of templates and read start positions. We use these reads to build a compatible read graph, convert to a bipartite graph, and apply Hopcroft–Karp to find a maximal matching (http://code.activestate.com/recipes/123641-hopcroft-karp-bipartite-matching/; Dataset S4). In the case of finite coverage, all reads that do not overlap are by definition compatible. This implies that the maximal antichain (comprised of reads that are all pairwise incompatible) must be comprised of reads which all overlap and therefore all start within a single interval of length R. This fact permits a significant computational simplification: Instead of computing the maximal antichain for the entire graph, we can restrict to smaller subgraphs of reads whose start positions are contained in the interval $[nR, (n + 2)R]$ for $n = 0, \ldots, (L/R) - 1$; identify a maximal antichain over each interval; and compute the maximum size over all intervals (Dataset S5). For a fixed mutation rate of 0.35, we simulate template counts in the range of 2–1,024; reads of 10, 20, and 30 bits; templates that are 16 read lengths long; and a coverage of 1–5×. For each condition, we perform 100 simulations and report the mean number of templates recovered.

In the counting problem, we establish a conservative estimate for the initial number of templates by allowing all compatible edges between reads. For the assembly problem, however, we want to establish high-probability assemblies. Consequently, we restrict our compatible edge graph to a subgraph composed of the best edges. When coverage is exhaustive, we can formulate a very precise definition of best edges. We join two R-bit reads by an edge if they overlap and agree for $R - 1$ bits. Consequently, two tags of a template are exactly matched if and only if every such $R - 1$ bit string across its span has a unique pattern. When the condition is not met—for example, due to too many templates and/or too few flipped bits—we find connected end tags that are not exactly matched (Dataset S3).

When coverage is finitely sampled, there are many ways to select the best edges for a subgraph (Fig. 5). We apply a simple scoring method that assigns a weight to an edge that reflects how unlikely it is that the reads agree on their mutation patterns by chance. Given two reads that overlap and agree on a window of size M with K bits flipped, we weight the edge by $-\log(p^K(1 - p)^{M-K})$, where p is the flip rate. We iterate through all of the edges in order of decreasing weight. An edge e from A to B is selected for inclusion in the subgraph if no edge from A or to B has already been selected that has a weight strictly greater than e. We carry out this procedure for each simulation and extract from the resultant subgraph all exact matches. Unlike the case for exhaustive coverage, exact matches may be incorrect. Because this is a simulation, we know the truth and record the number of exact matches that are correct and incorrect. We also record whether the exact path is also true (Dataset S6).

ACKNOWLEDGMENTS. We thank Lior Pachter, Michael Ronemus, Ivan Iossifov, Jude Kendall, and Talitha Forcier for useful discussions and suggestions. This work was supported by a grant from the Simons Foundation (Award SFARI 235988, to M.W.).

1. McCloskey ML, Stöger R, Hansen RS, Laird CD (2007) Encoding PCR products with batch-stamps and barcodes. *Biochem Genet* 45(11-12):761–767.
2. Miner BE, Stöger RJ, Burden AF, Laird CD, Hansen RS (2004) Molecular barcodes detect redundancy and contamination in hairpin-bisulfite PCR. *Nucleic Acids Res* 32(17):e135.
3. Casbon JA, Osborne RJ, Brenner S, Lichtenstein CP (2011) A method for counting PCR template molecules with application to next-generation sequencing. *Nucleic Acids Res* 39(12):e81.
4. Fu GK, Hu J, Wang PH, Fodor SP (2011) Counting individual DNA molecules by the stochastic attachment of diverse labels. *Proc Natl Acad Sci USA* 108(22):9026–9031.
5. Islam S, et al. (2014) Quantitative single-cell RNA-seq with unique molecular identifiers. *Nat Methods* 11(2):163–166.
6. Jabara CB, Jones CD, Roach J, Anderson JA, Swanstrom R (2011) Accurate sampling and deep sequencing of the HIV-1 protease gene using a Primer ID. *Proc Natl Acad Sci USA* 108(50):20166–20171.
7. Kivioja T, et al. (2012) Counting absolute numbers of molecules using unique molecular identifiers. *Nat Methods* 9(1):72–74.
8. Hiatt JB, Pritchard CC, Salipante SJ, O'Roak BJ, Shendure J (2013) Single molecule molecular inversion probes for targeted, high-accuracy detection of low-frequency variation. *Genome Res* 23(5):843–854.
9. Kinde I, Wu J, Papadopoulos N, Kinzler KW, Vogelstein B (2011) Detection and quantification of rare mutations with massively parallel sequencing. *Proc Natl Acad Sci USA* 108(23):9530–9535.
10. Schmitt MW, et al. (2012) Detection of ultra-rare mutations by next-generation sequencing. *Proc Natl Acad Sci USA* 109(36):14508–14513.
11. Hicks J, Navin N, Troge J, Wang Z, Wigler M (2012) *Varietal Counting of Nucleic Acids for Obtaining Genomic Copy Number Information*. US patent application publication US 2014 and US patent publication 0065609 (March 6, 2014).
12. Keith JM, et al. (2004) Algorithms for sequence analysis via mutagenesis. *Bioinformatics* 20(15):2401–2410.
13. Keith JM, et al. (2004) Unlocking hidden genomic sequence. *Nucleic Acids Res* 32(3):e35.
14. Mitchelson KR (2011) Sequencing of difficult DNA regions by SAM sequencing. *Methods Mol Biol* 687:75–88.
15. Sipos B, Massingham T, Stütz AM, Goldman N (2012) An improved protocol for sequencing of repetitive genomic regions and structural variations using mutagenesis and next generation sequencing. *PLoS ONE* 7(8):e43359.
16. Shortle D, Botstein D (1983) Directed mutagenesis with sodium bisulfite. *Methods Enzymol* 100:457–468.
17. Dilworth RP (1950) A decomposition theorem for partially ordered sets. *Ann Math* 51(1):161–166.
18. König D (1931) Graphen und matrizen. *Mat Fiz Lapok* 38:116–119.
19. Hopcroft J, Karp R (1973) An n/5/2 algorithm for maximum matchings in bipartite graphs. *SIAM J Comput* 2(4):225–231.
20. Alberts B (1989) *Molecular Biology of the Cell* (Garland, New York), 2nd Ed, p 528.
21. Narzisi G, Mishra B (2011) Scoring-and-unfolding trimmed tree assembler: Concepts, constructs and comparisons. *Bioinformatics* 27(2):153–160.
22. Bransteitter R, Pham P, Scharff MD, Goodman MF (2003) Activation-induced cytidine deaminase deaminates deoxycytidine on single-stranded DNA but requires the action of RNase. *Proc Natl Acad Sci USA* 100(7):4102–4107.
23. Krueger F, Andrews SR (2011) Bismark: A flexible aligner and methylation caller for Bisulfite-Seq applications. *Bioinformatics* 27(11):1571–1572.
24. Otto C, Stadler PF, Hoffmann S (2012) Fast and sensitive mapping of bisulfite-treated sequencing data. *Bioinformatics* 28(13):1698–1704.
25. Krueger F, Kreck B, Franke A, Andrews SR (2012) DNA methylome analysis using short bisulfite sequencing data. *Nat Methods* 9(2):145–151.