ISCI, Volume *18*

Supplemental Information

Kevlar: A Mapping-Free Framework for Accurate Discovery of *De Novo* Variants

Daniel S. Standage, C. Titus Brown, and Fereydoun Hormozdiari

Supplemental Information
Kevlar: a mapping-free framework for accurate discovery of *de novo* variants

Daniel S. Standage, C. Titus Brown, Fereydoun Hormozdiari

Transparent Methods

Assessing diagnostic utility of novel k-mers

We expect that a *de novo* mutation will result in numerous novel *k*-mers, given a sufficiently large value of *k*. We also expect that these novel *k*-mers will be present in high abundance, given sufficiently deep sampling of the proband genome. Intuitively, we can use these novel *k*-mers to identify reads that span the *de novo* variant—see Figure 1a.

We assessed this intuition by traversing the human reference genome (GRCh38) base by base, simulating variants (SNVs and 5 bp deletions) at each position. For each simulated mutation, we determined the fraction of *k*-mers spanning the mutation that exist nowhere else in the genome, and thus act as a diagnostic signature of that particular variant. We then aggregated over the entire genome the probability that *k*-mer spanning a mutation (in this case 31-mers) will be novel—see Figure 1b and 1c.

Based on the results of this experiment, we formulate the *de novo* variant discovery problem as a search for putatively novel *k*-mers that are abundant in the proband and effectively absent in each parent. For sake of simplicity, we are using the term *proband* to refer generally to the subject or focal individual, and *parent* to refer generally to control individuals.

Here, *abundant* and *effectively absent* are defined in terms of a simple threshold model. Let X be the *absence threshold*, and Y be the *presence threshold*, and $A = \{A_p, A_m, A_f\}$ be the abundances of a *k*-mer in the proband, mother, and father. We designate this *k*-mer as "*interesting*" (putatively novel) if and only if $A_p \geq Y$, $A_m \leq X$, and $A_f \leq X$. Based on our experience, the values $Y = 5$ and $X = 1$ produce desirable results for 30x sequencing coverage.

Kevlar workflow

The steps of the Kevlar workflow, summarized at a high level in Figure 1d, are described in detail in the subsequent sections.

Step 0: Compute k-mer counts

Preliminary to identifying novel *k*-mers, the abundance of each *k*-mer in each sample must be counted. Storing exact counts of every *k*-mer requires a substantial amount of space (dozens of gigabytes or more per sample), so Kevlar exploits several strategies to reduce the space required for keeping *k*-mer counts in memory.

First, Kevlar stores approximate *k*-mer counts in a Count-Min sketch, a probabilistic data structure similar to a Bloom filter that operates in a fixed amount of memory, exchanging accuracy for space efficiency (Zhang et al., 2014). A separate Count-Min sketch is used for each sample. The accuracy of each Count-Min sketch depends on its size and the number of distinct elements (*k*-mers in this case) being tracked. The Count-Min sketch exhibits a one-sided error, meaning that *k*-mer counts are sometimes overestimated but never underestimated. The extent of inaccuracy in the *k*-mer counts is summarized by the *false discovery rate (FDR)* statistic computed from the occupancy of the Count-Min sketch.

Second, Kevlar uses a masked counting strategy in which *k*-mers present in the reference genome and a contaminant database (composed of bacterial, viral, vector, and adapter sequences) are ignored. This substantially reduces the number of *k*-mers to be stored in the Count-Min sketch, and as a consequence the desired level of accuracy can be maintained using a smaller amount of space.

Third, *k*-mer counts are recomputed with exact precision in subsequent steps of the Kevlar workflow, which means any *k*-mer retained erroneously due to an inflated count can be compensated for at a later stage. As a consequence, it is possible to reduce the size of the Count-Min sketch even further, resulting in a FDR of 0.5 or greater.

Kevlar's *k*-mer counting operations are invoked with the `kevlar count` command, and rely on bulk sequence loading procedures and an implementation of the Count-Min sketch data structure from the khmer library (Crusoe

et al., 2015; Standage et al., 2017; Zhang et al., 2014). Note that several alternative *k*-mer counting libraries and tools (Marçais and Kingsford, 2011; Rizk et al., 2013) have been developed and utilized to solve a variety of different biological problems (Bray et al., 2016; Patro et al., 2014; Rahman et al., 2018; Sun and Medvedev, 2018).

Step 1: Identifying novel k-mers and reads

To identify sequences spanning *de novo* variants, Kevlar scans each read sequenced from the proband. The per-sample abundances of each *k*-mer are queried from the Count-Min sketches computed in previous steps. If a *k*-mer is present in high abundance in the proband and absent from the parents (that is, it satisfies user-specified abundance thresholds), it is designated as "interesting" or putatively novel. This operation is similar to the selection of "novo" *k*-mers by NovoBreak (Chong et al., 2017) and "significant" *k*-mers by HAWK (Rahman et al., 2018). Any read containing one or more interesting *k*-mers is retained for subsequent processing. This step is implemented in the `kevlar novel` command.

Step 2: Contamination, reference, and abundance filters

Reads containing putative novel *k*-mers are filtered prior to subsequent analysis. This filtering step serves two purposes.

First, Kevlar re-computes the abundance of each interesting *k*-mer in the proband sample. The relatively small volume of these reads allows Kevlar to re-compute *k*-mer counts with perfect accuracy in a small amount of memory and time. Any *k*-mer whose corrected count no longer satisfies the required abundance threshold is discarded. Note that since only proband reads are retained, only the proband *k*-mer abundances can be recomputed. This filtering step will not recover a *k*-mer that is erroneously discarded in the previous step due to an erroneously inflated *k*-mer abundance in one of the control (parent and sibling) samples.

Second, if for any reason *k*-mers from the reference genome and contaminants are not ignored in the initial *k*-mer counting step, this filtering step provides another opportunity to discard these *k*-mers.

After these filters are applied, any read that no longer contains any novel *k*-mers is discarded, and the remainder of the reads are retained for subsequent analysis.

The `kevlar filter` command is used to execute these contamination, reference, and abundance filters.

Step 3: Partitioning reads using shared novel k-mers

Interesting reads spanning the same variant are expected to share numerous interesting *k*-mers. These shared novel *k*-mers provide a mechanism for grouping the reads into disjoint sets reflecting distinct variants.

To be precise, we define a *read graph* G as follows: every read containing one or more novel *k*-mers is represented by a node in G, and a pair of nodes is connected by an edge if they have one or more novel *k*-mers in common. With this formulation, if two reads share a novel *k*-mer they are part of the same connected component in G. Overall G is sparse, but typically each connected component of the graph is highly connected. In subsequent steps, each component or partition $p \in G$ is analyzed independently.

The `kevlar partition` command implements this partitioning strategy.

Step 4: Contig assembly and reference target selection

For each connected component $p \in G$, we assemble the corresponding reads using the overlap-based algorithm implemented in the fermi-lite library (Li, 2017a). Briefly, fermi-lite performs error correction, trims reads at unique *l*-mers, constructs an FM-index of the trimmed reads, and constructs a transitively reduced overlap graph. The optimal path in the final graph is output as a contig C_p suitable for variant calling.

Next, we select a target reference sequence (or set of candidate targets) for the contig C_p. Briefly, Kevlar decomposes the contig into overlapping subsequences of length l (*seeds*; $l = 51$ by default), and uses BWA MEM (Li, 2013) to identify locations of exact matches for each seed sequence in the reference genome. The genomic interval that spans all seed exact matches, plus Δ nucleotides in each direction ($\Delta = 50$ by default), is then selected as the target reference sequence for C_p. If any adjacent seed matches are separated by more than D nucleotides ($D = 10,000$ by default), then the seed matches are split at that point and multiple reference targets are selected. The set of reference target sequences corresponding to contig C_p is denoted T_{C_p}.

Read assembly is invoked with the `kevlar assemble` command, and reference target selection is invoked with the `kevlar localize` command.

Step 5: Contig alignment and variant annotation

The contig C_p is aligned to each reference target sequence $t \in T_{C_p}$ using the ksw2 library (Li, 2017b)—specifically its implementation of Green's formulation of dynamic programming global alignment and extension (`ksw2_extz`). If there are multiple candidate targets, only the highest scoring alignment is retained. When a contig aligns to multiple locations with the same optimal score, all optimal alignments are retained for variant calling.

Prior to variant calling, kevlar right-aligns any gaps at the right end of the alignment to minimize the number of alignment blocks/operations. Next, Kevlar inspects the alignment path (represented as a CIGAR string) of each alignment and tests for matches against expected patterns. Alignments matching the pattern `^(\d+[DI])?\d+M(\d+[DI])?$` are classified as SNV events, and the "match" block of the alignment is scanned for mismatches between the contig and the reference target. Any mismatch is reported as a single nucleotide variant. Alignments matching the pattern `^(\d+[DI])?\d+M\d+[ID]\d+M(\d+[DI])?$` are classified as indel events. In addition to reporting the internal gap of this alignment as an indel variant, the flanking "match" blocks are also scanned for mismatches between the contig and the target to be reported as putative SNVs. Any alignment not matching the two patterns described above is designated as an uninterpretable "no-call" and listed in the output along with the corresponding contig sequence.

In some cases, there is a possibility that kevlar will report two or more calls in close proximity. While the probability of two *de novo* variants occurring in close proximity is effectively nil, it is common for an inherited variant to occur proximal to a *de novo* variant. Occasionally one of these inherited variants will not be spanned by any interesting *k*-mers, in which case it can immediately be designated as a "passenger" variant call. However, in cases where an inherited variant is spanned by one or more interesting *k*-mers, we rely on subsequent examination of *k*-mer abundances to distinguish novel variants from inherited variants.

The `kevlar call` command computes the contig alignments and makes preliminary variant calls.

Step 6: Likelihood scoring model for ranking and filtering variant calls

Given the filters already discussed, false *interesting* *k*-mer designations are rare throughout the genome overall. Redundancy from a high depth of sequencing coverage prevents sequencing errors from driving the reported abundance of *k*-mers present in the parents to 0. If a *k*-mer is present in either parent, it is disqualified from the *interesting* or novel designation.

We observed false *interesting* *k*-mer designations are enriched around inherited mutations. It is very common for variants present in one parent to be absent from the other parent. If by chance the depth of sequencing coverage is low at such a locus in the donor parent, there may not be enough redundancy to compensate for sequencing errors. As a result, some *k*-mers that are truly present in the donor parent will have a reported abundance of 0. Being truly absent from the other parent, these *k*-mers are erroneously designated as unique to the proband.

A related complication occurs when a novel variant is proximal to an inherited variant. Both variants are reflected in the alignment of the associated contig (assembled from proband-derived *interesting* reads) to the reference genome. In both of these cases, distinguishing novel variants from inherited variants benefits from examination of the abundances of all *k*-mers containing each variant, as well as the corresponding reference *k-mers*.

We utilize a likelihood based model to score and rank the predicted *de novo* variants. We consider the abundance of the interesting *k*-mers to calculate the likelihood of the event observed being *de novo*, inherited, or a false call. Using these likelihood probabilities, we calculate a score for each variant being truly a *de novo* variant based on ratio of likelihoods.

First, for each variant we define a set of alternate *k*-mers $\mathbf{A}$ as the *k*-mers indicating existence of the variant (alternate genotype). We consider only *k*-mers that are unique to this variant (that is, they don't appear in any other location in the reference genome). We assume that there are a total of *n* alternate *k*-mers.

Let the random variables v_c, v_f, and v_m indicate the genotype (i.e. $\{0/0, 0/1, 1/1\}$) of the putative variant in the proband/child, father, and mother respectively. The random variable $\mathbf{A}_c = \{A_{c_1}, A_{c_2}, ..., A_{c_n}\}$ denotes the counts of the alternate allele *k*-mers in the proband, $\mathbf{A}_m = \{A_{m_1}, A_{m_2}, ..., A_{m_n}\}$ the alternate allele *k*-mer counts in the mother, and $\mathbf{A}_f = \{A_{f_1}, A_{f_2}, ..., A_{f_n}\}$ the alternate allele *k*-mer counts in the father. The likelihood that a putative variant is *de novo* can be calculated as follows.

$$
\begin{aligned}
L(\text{dn} = 1) &= P(\mathbf{A}_c, \mathbf{A}_m, \mathbf{A}_f \mid \text{dn} = 1) \\
&= P(\mathbf{A}_c, \mathbf{A}_m, \mathbf{A}_f \mid v_c = 0/1, v_m = 0/0, v_f = 0/0) \\
&= P(\mathbf{A}_c \mid v_c = 0/1) P(\mathbf{A}_m \mid v_m = 0/0) P(\mathbf{A}_f \mid v_f = 0/0)
\end{aligned}
$$

We note that there are dependencies between *k*-mer counts within a sample. However, to simplify the calculation of likelihoods, we assume independence between the *k*-mer counts and provide an approximation of the likelihoods. For calculating the probability of an observed *k*-mer count conditioned on a 1/1 genotype, we assume a normal distribution where parameters are learned empirically for each sample using only exonic *k*-mers that occur only once in the reference genome. For the genotype 0/0 we use binomial distribution to calculate the likelihood of the observed *k*-mer abundance assuming the *k*-mer is generated by, e.g., sequencing error. Similarly, we calculate the likelihood that a putative variant is a false positive prediction by conditioning on the variant's non-existence (genotype $0/0$) in all three samples, i.e. $L(fp = 1) = P(\mathbf{A}_c, \mathbf{A}_m, \mathbf{A}_f \mid v_c = 0/0, v_m = 0/0, v_f = 0/0)$.

Finally, we calculate the likelihood of observed *k*-mer counts under the inheritance assumption. As there are several different valid scenarios to represent variant inheritance the likelihood calculation requires additional steps as explained below (again assuming independence of *k*-mer abundances as an approximation).

$$
\begin{aligned}
L(\text{ih} = 1) &= P(\mathbf{A}_c, \mathbf{A}_m, \mathbf{A}_f \mid \text{ih} = 1) \\
&\approx \prod_{i=1}^{n} P(A_{c_i}, A_{m_i}, A_{f_i} \mid \text{ih} = 1) \\
&= \prod_{i=1}^{n} \frac{P(\text{ih} = 1 \mid A_{c_i}, A_{m_i}, A_{f_i}) P(A_{c_i}, A_{m_i}, A_{f_i})}{P(\text{ih} = 1)} \\
&= \prod_{i=1}^{n} \frac{P(A_{c_i}, A_{m_i}, A_{f_i})}{P(\text{ih} = 1)} \times P(\text{ih} = 1 \mid A_{c_i}, A_{m_i}, A_{f_i})
\end{aligned}
$$

We calculate the $P(\text{ih} = 1 \mid A_{c_i}, A_{m_i}, A_{f_i})$ as summation of probability of possible trio-genotype combinations representing inheritance scenarios (e.g., $(v_c = 1/0, v_f = 1/0, v_m = 0/0)$ or $(v_c = 1/0, v_f = 0/0, v_m = 1/0)$). Furthermore, we assume a constant prior value for $P(\text{ih} = 1)$ based on all possible valid inheritance scenarios.

Finally, we utilize a heuristic score motivated from the likelihood ratio test to score and rank any predicted variant as being a *de novo* variant. Note that, as numerical calculation of the likelihoods is numerically prone to error we consider the logarithm of the score. Thus, we formally define the score assigned to each variant for being *de novo* as $S_L = \log L(dn = 1) - \max\{\log(L(ih = 1)), \log(L(fp = 1))\}$. The `kevlar simlike` command computes likelihoods for preliminary variant calls, sorts the calls, and filters out low scoring and otherwise problematic calls.

Data simulations

We simulated whole-genome shotgun sequencing for a hypothetical trio (father, mother, and proband) to evaluate the accuracy of our *de novo* variant discovery algorithm. Using the human reference genome (GRCh38) and a catalog of common variants (dbSNP), we constructed two independent diploid genomes representing the two parents. We randomly selected SNPs and indels from dbSNP and assigned the variants to each parental haplotype at a rate of 1 for every 1000 bp.

We then constructed the diploid proband genome through recombination of the parental diploid genomes and simulated germline mutation. SNVs and indels ranging from <10 bp to 400 bp in length were simulated as heterozygous events unique to the proband, representing *de novo* variation.

Finally, we used wgsim (Li, 2011) to simulate whole-genome shotgun sequencing of each individual. This produced sequences resembling Illumina 2x150bp paired-end reads with low sequencing error rate. The sequencing was repeated at four different average depths of sequencing coverage: 10x, 20x, 30x, and 50x.

References

Bray, N. L., Pimentel, H., Melsted, P., and Pachter, L. (2016). Near-optimal probabilistic RNA-seq quantification. *Nature Biotechnology*, 34(5):525.

Chong, Z., Ruan, J., Gao, M., Zhou, W., Chen, T., Fan, X., Ding, L., Lee, A. Y., Boutros, P., Chen, J., et al. (2017). novobreak: local assembly for breakpoint detection in cancer genomes. *Nature methods*, 14(1):65.

Crusoe, M. R., Alameldin, H. F., Awad, S., Boucher, E., Caldwell, A., Cartwright, R., Charbonneau, A., Constantinides, B., Edvenson, G., Fay, S., et al. (2015). The khmer software package: enabling efficient nucleotide sequence analysis. *F1000Research*, 4.

Li, H. (2011). wgsim: Read simulator for next generation sequencing. https://github.com/lh3/wgsim.

Li, H. (2013). Aligning sequence reads, clone sequences and assembly contigs with BWA-MEM. arXiv:1303.3997.

Li, H. (2017a). fermi-lite: Standalone C library for assembling illumina short reads in small regions. https://github.com/lh3/fermi-lite.

Li, H. (2017b). KSW2: Global alignment and alignment extension. https://github.com/lh3/ksw2.

Marçais, G. and Kingsford, C. (2011). A fast, lock-free approach for efficient parallel counting of occurrences of k-mers. *Bioinformatics*, 27(6):764–770.

Patro, R., Mount, S. M., and Kingsford, C. (2014). Sailfish enables alignment-free isoform quantification from rna-seq reads using lightweight algorithms. *Nature Biotechnology*, 32(5):462.

Rahman, A., Hallgrímsdóttir, I., Eisen, M., and Pachter, L. (2018). Association mapping from sequencing reads using k-mers. *eLife*, 7:e32920.

Rizk, G., Lavenier, D., and Chikhi, R. (2013). DSK: k-mer counting with very low memory usage. *Bioinformatics*, 29(5):652–653.

Standage, D. S., Aliyari, A., Cohen, L. J., Crusoe, M. R., Head, T., Irber, L., Joslin, S. E., Kingsley, N. B., Murray, K. D., Neches, R., Scott, C., Shean, R., Steinbiss, S., Sydney, C., and Brown, C. T. (2017). khmer release v2.1: software for biological sequence analysis. *The Journal of Open Source Software*, 2(15):272.

Sun, C. and Medvedev, P. (2018). Toward fast and accurate SNP genotyping from whole genome sequencing data for bedside diagnostics. *bioRxiv*, page 239871.

Zhang, Q., Pell, J., Canino-Koning, R., Howe, A. C., and Brown, C. T. (2014). These are not the k-mers you are looking for: Efficient online k-mer counting using a probabilistic data structure. *PLoS ONE*, 9(7):1–13.

Supplementary Figures

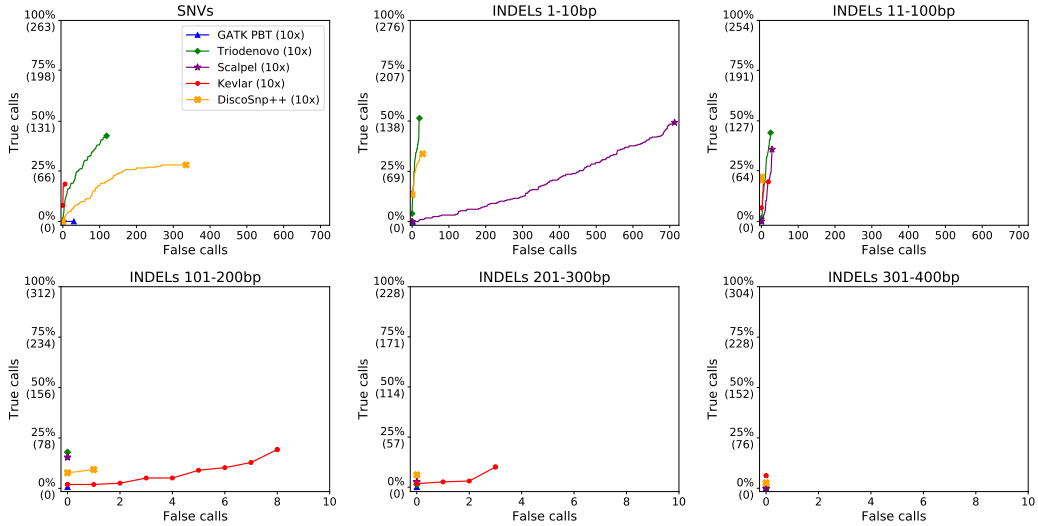

Figure S1 Accuracy of five *de novo* variant prediction algorithms at 10x coverage, Related to Figure 2.

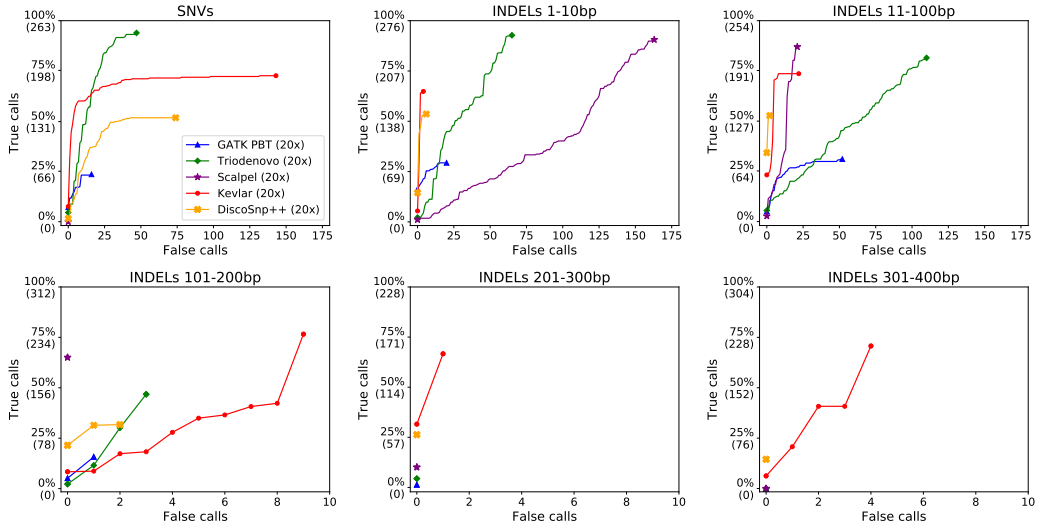

Figure S2 Accuracy of five *de novo* variant prediction algorithms at 20x coverage, Related to Figure 2.

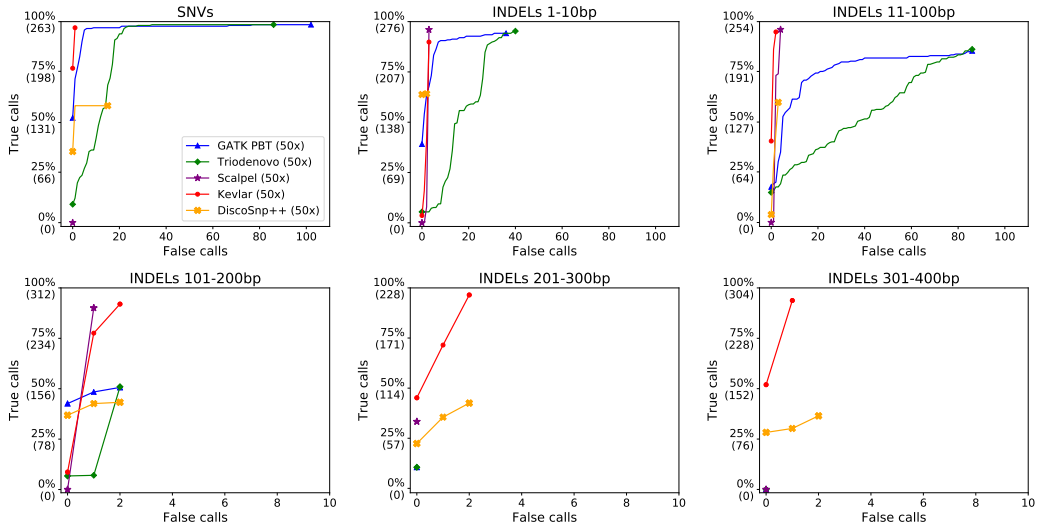

Figure S3 Accuracy of five *de novo* variant prediction algorithms at 50x coverage, Related to Figure 2.