v. 25sep2016

Supplemental Material

Improved assembly of noisy long reads by *k*-mer validation

Authors: Antonio Bernardo Carvalho, Eduardo G. Dupim, and Gabriel Nassar

Affiliation:

Departamento de Genética, Universidade Federal do Rio de Janeiro, Caixa Postal 68011, CEP 21941-971, Rio de Janeiro, Brazil.

Corresponding author address:

A. Bernardo Carvalho
Departamento de Genética, Universidade Federal do Rio de Janeiro
Caixa Postal 68011, CEP 21941-971
Rio de Janeiro, Brazil.
e-mail: bernardo@biologia.ufrj.br
phone: 55-21-2260-8820

Table of Contents

Supplemental Methods

Assembly of a model genome

We used as a model genome a small and difficult region: a 44 kb segmental duplication (98% identity between the two copies), which is part of a much larger complex of segmental duplications located in the 10q11 region of the human genome (this 44 kb segmental duplication is shown in brown in Fig. 3 of (Chaisson et al. 2015)). We used the finished sequence to simulate PacBio reads from both copies of the 44 kb segmental duplication, along with ~300 kb of flanking sequence. Specifically, the sequence of the "right" contig (200,828 bp) came from the BAC clone CH17-476B22 (accession AC255477), and the "left" contig (401,520 bp) came from the BAC clones CH17-177M15 and CH17-448F9 (accessions AC255509 and AC255412, respectively). We used simulated reads because we wanted to know which segmental duplication copy they came from (this would not be possible with real data). PacBio reads were simulated with the ReadSim program (https://sourceforge.net/p/readsim ; program settings:100-fold coverage, size distribution from real human PacBio reads, and default PacBio error rates). The resulting 8017 reads were assembled as the other genomes, using either the standard MHAP or the modified version (with *k*-mer validation). The list of the valid *k*-mers was obtained from the concatenated left and right contigs, using the jellyfish program; in the case of L-masking we used all *k*-mers and in the case of LH-masking we used only the single-copy *k*-mers.

Read error correction measurements

For real PacBio reads (*i.e.*, not simulated; Table 1) the analysis was done as follows. We first generated a reference list of correct *k*-mers, either from the assembled genome itself (in *E. coli* and *C. elegans*, which have completely finished genomes), or from Illumina *k*-mers (for the remaining genomes; correct *k*-mers was operationally defined as those with a frequency higher that one seventh of the single-copy peak). In both cases we decomposed each read into its *k*-mers, compared them with the reference list of correct *k*-mers, and counted for each read the number of correct *k*-mers among the total *k*-mers, using custom scripts. The Celera Assembler keeps the original deflines of the reads, so we could track the same read under different read correction schemes (*e.g.*, standard MHAP vs. *k*-mer validation).

In the case of the simulated PacBio reads of the "model genome" the measurement of error correction was more precise, because we know where each read came from. We used one internal file of the Celera Assembler (asm.gkpStore.fastqUIDmap) to match the corrected reads with the segmental duplication copy which they come from, and used *bwa* (Li and Durbin 2009) to align these reads to the proper copy. The resulting bam files were processed with a custom script based on the *pysam* module (Li et al. 2009), to measure the error correction in each site of each read. A match between the read and the source contig was scored as "correct base", a mismatch as "wrong base" and a gap in the read as "deleted base". The results were then sorted for the three types of sites: outside the segmental duplication, conserved within the segmental duplication, and sequence family variant within the segmental duplication ("SFV sites"). There are ~450 SFV sites between the two copies of the segmental duplication, and we randomly choose the same number of the two other types of sites.

Assessment of assembly quality with QUAST

We used QUAST version 3.2 (Gurevich et al. 2013) with the default settings. The reference genomes are: *E. coli*, accession NC_000913.3 ; *S. cerevisae*, accession GCA_000146045.2 ; *A. thaliana*, accessions NC_003070.9, NC_003071.7, NC_003074.8, NC_003075.7, NC_003076.8, NC_001284.2, and NC_000932.1; *C. elegans*, accessions NC_003279.8, NC_003280.10, NC_003281.10, NC_003282.8, NC_003283.11, NC_003284.9, and NC_001328.1); *D. melanogaster*, file dmel-all-chromosome-r6.09.fasta, downloaded from FlyBase (dos Santos et al. 2015).

Supplemental Results

Low frequency and high-frequency cut-offs of Illumina *k*-mers

The immediate purpose of the low-frequency cut-off is to remove error *k*-mers from the Illumina reads. One might expect that this is unimportant because the sequencing errors in the Illumina reads would be different from those of PacBio reads, and hence only real *k*-mers would be shared among them (*i.e.*, a "dirty" Illumina-derived list of valid *k*-mers would work equally well to validate PacBio *k*-mers). However, in practice, we observed both in *Drosophila* and *C. elegans* that removing only very rare *k*-mers (*e.g.*, all *k*-mers that occur less than three times in the Illumina reads) is detrimental to the assemblies (Supplemental Table S14), presumably because intermediate frequency *k*-mers may be error *k*-mers from repeats, and hence may spuriously validate them. This is hinted by the fact that we always found a large number error *k*-mers with frequencies bigger than 1 (*e.g.*, Fig. 1), whereas one would naively expect all error *k*-mers to have frequency of 1. We attempted to obtain a parameter-based cut-off, but when we modeled the single-copy *k*-mer distribution we found that it contains a large, and seemingly unpredictable, non-Poisson component, that differs among genomes (not shown). So by limited trial-and-error we choose as the low cut-off one seventh of the coverage of the single-copy peak. This should be more thoroughly investigated in the future.

The purpose of the high-frequency cut-off is to remove *k*-mers that occur more than once in the genome. The chosen cut-off (1.5 fold the single-copy coverage) was a compromise between the desirable elimination two-copy *k*-mers (which distribution should be centered at two-fold the single-copy peak; Supplemental Fig. S5) and the undesirable elimination of single-copy *k*-mers. Again, this should be more thoroughly investigated in the future.

Finally, species with Y or W chromosomes (in our dataset, *D. melanogaster* and *H. sapiens*) pose a special problem while preparing the valid *k*-mer list from Illumina reads. For example, the *D. melanogaster* Illumina dataset came from males (Gutzwiller et al. 2015), and it can be clearly seen that the X and Y have half the autosomal coverage (Supplemental Fig. S5; the observed peak of single-copy *k*-mers for the sex chromosomes and the autosomes are respectively 46x and 93x). Hence the high frequency cut-off either will not exclude repetitive *k*-mers from the X and Y, or will remove many single-copy *k*-mers from the autosomes. We opted for the former (H=150x), but also tried H=93x. The result is mixed: H=150x yield a higher NG50, but H=93x yield the largest contig size, possibly reflecting a trade-off between X and autosomal assembly (Supplemental Table S14). The ideal solution would be to combine two Illumina valid *k*-mer lists: a female-derived list, used to get the autosomal (and the X-linked) single-copy *k*-mers, and a male-derived list, used to get the single-copy *k*-mers from the X and the Y.

k-mer validation assembly in the presence of coverage bias of Illumina reads

Illumina sequencing is known to have reproducible coverage bias, such as against GC-rich regions found in some mammalian promoters and exons. One of the most extreme cases documented by Ross et al (2013) occurs with the human *NCS1* gene: in a 198x Illumina dataset, the first 72 bases of the *NCS1* first exon were completely uncovered, and the surrounding 2kb has around 20x coverage (see their Fig. 1). Such coverage bias can be a problem for the *k*-mer validation method because most *k*-mers in these regions will not be present in the valid *k*-mer list, which can disturb the self-correction of the long-reads. We addressed this possibility as follows. We first simulated PacBio reads using the same procedures described in the Supplemental Methods (section " Assembly of a model genome"), from a 855 kb contig (accession JSAF02032866) that included the *NCS1* gene. We then assembled the simulated reads with LH *k*-mer validation, using two different valid *k*-mer lists. The first list was obtained from contig JSAF02032866 itself and hence amounts to a bias-free Illumina sequencing. The second list was obtained from a deleted version of contig JSAF02032866 that removed 2kb surrounding the exon 1 of the *NCS1* gene; such list amounts to the strongest possible bias, across 2 kb (much stronger and longer than

reported by Ross et al (2013)). As a control, we also assembled the simulated PacBio reads with the standard MHAP. The result is shown in Supplemental Table S6: The two *k*-mer validation assemblies are very good (zero duplications; zero assembly breaks) and nearly indistinguishable (the nucleotide identity is over 99.99%); both are much better than the standard MHAP. This strongly suggests that LH *k*-mer validation assembly is insensitive to the normally encountered bias in Illumina coverage, which span a few hundred base pairs. The most likely explanation for this insensitivity is that the PacBio read length is much larger than the Illumina coverage gap (2kb in our case), so most read overlaps across the gap will still be detected because the *k*-mers outside it provide enough alignment seeds.

Assembly of additional segmental duplications

The assembly of one human segmental duplication was very clearly improved by *k*-mer validation ("model genome"; Supplemental Table S7). It is desirable to check if this is a general result, since what resolves duplications in some cases may fragment in others. So we tested two additional duplications in the 10q11 region of the human genome, which were longer and harder. They are showed in Fig. 3 of (Chaisson et al. 2015): we picked the light blue and light green segmental duplications, located at the left sides of Contig 1 and Contig 2. Their coordinates in the reference human genome (GRCh38.p7; accession NC_000010.11) are: light green segmental duplication (130 kb), 45,676,384 to 45,806,813 and 49,986,226 to 50,147,389; light blue segmental duplication, 45,896,739 to 45,997,277 and 49,865,872 to 49,966,528. Note that they are close to each other, and in order to study their assembly we build the whole region with the BACs showed in the Fig. 3 of (Chaisson et al. 2015) and extracted two contigs (contig_GB1 and contig_GB2), each one spanning one light blue / light green copy (coordinates of the extracted contigs in NC_000010.11: contig_GB1, 45,550,656 to 46,100,519; contig_GB2 are 49,865,872 to 50,247,286). We then generated PacBio reads and valid k-mer files as described before (Supplemental Methods, section "Assembly of a model genome"), and assembled the PacBio reads with the three methods (M, L, and LH). As in the case of the "model genome", a perfect assembly should result in two contigs; we initially got 27 (M assembly), 27 (L assembly) and 5 (LH assembly). So LH k-mer validation improved the assembly a lot, but still introduced a few assembly breaks. Given the large size of these segmental duplications, it is possible that the assembly breaks were caused by incorrect read overlapping inside the Celera Assembler, and not at the correction of the raw PacBio reads (where we did the k-mer validation). To test this, we took the three corrected read datasets (from M, L, and LH assemblies) and submit them to the Celera Assembler using more stringent assembly parameters (we increased the ovlMinLen parameter of the runCA script from the default 100 bp to 2 kbp). As shown in Supplemental Table S8, the "LH_2000" assembly reconstructed the two original sequences nearly perfectly (99.6% identity with the reference sequences); the M and L assemblies did not improve (Supplemental Table S8, rows 4-6) . Thus, regarding the initial question of this section, it seems safe to conclude that the assembly improvement brought by *k*-mer validation is a general phenomenon. These results also suggest that assembly of repetitive regions will benefit from increased stringency after the initial read correction, as we did by increasing the ovlMinLen parameter.

Assembly of human chromosomes 15 and 17

While the assembling of the whole human genome is beyond the scope of this work, it would be interesting to test *k*-mer validation with human datasets larger than the individual segmental duplications. So we used sorted reads from human chromosomes 15 and 17, obtained as follows (we thank one of the anonymous reviewers for providing these reads). PacBio reads from one individual (HG002; the son of the Ashkenazi Trio; (Zook et al. 2016)) were aligned with BLASR (version rc43) to the human reference genome (GRCh37) using the following parameters:

-clipping subread -sam -minMapQV 30 -advanceExactMatches 10 -maxMatch 25 -sdpTupleSize 13 -sdpMaxAnchorsPerPosition 10 -indelRate 2 -insertion 10 -deletion 10 -mismatch 4 .

Reads aligning to chromosomes 15 and 17 were sorted. We then assembled separately the sorted reads from the two chromosomes using the standard MHAP, L-validation and LH-validation (the valid *k*-mers came from Illumina reads of the same individual (Supplemental Table S12; Supplemental Table S2). As

with other genomes, we found that LH-masking produce more contiguous assemblies (Supplemental Fig. S11), with less misassemblies (Supplemental Table S9). One limitation of this analysis came from the use of sorted reads: there seem to be gaps in read coverage near the telomeres and the centromeres in all assemblies, possibly because reads from repetitive regions were lost during the sorting. This may explain why the results of *k*-mer validation are less compelling here (*e.g.*, compare Supplemental Fig. S6 with Supplemental Fig. S11). At any rate, it is clear that *k*-mer validation improved the assembly of these two human chromosomes.

Hybrid assemblies of the model genome

Throughout this work we have evaluated the *k*-mer validation method using the standard MHAP as the baseline. While this seems reasonable on the grounds that MHAP is the currently recommended overlapper for the PBcR pipeline (Berlin et al. 2015), it may also be argued that the comparison is unfair because *k*-mer validation uses additional information (namely, the Illumina reads used to produce the valid *k*-mer lists). So we also compared *k*-mer validation with two approaches that also use Illumina reads. In the first approach ("hybrid"), we used the standard hybrid assembly as implemented in the PBcR pipeline, in which the long reads are corrected by aligning them to the short reads, and then are directly assembled by the Celera Assembler (Koren et al. 2012). The second approach ("DCA", for "doubly corrected assembly") intend to be as parallel to *k*-mer validation as possible: it combines the Illumina information with the long-reads self-correction. Namely, DCA started with the Illumina-corrected long reads produced in the hybrid assembly, then self-correct them using the standard MHAP algorithm, and finally assemble them with the Celera Assembler. Note that DCA differs from *k*-mer validation because the former directly corrects the PacBio reads with Illumina reads, whereas in the second the Illumina reads information (in the form a valid *k*-mer list) only guides the alignment of PacBio reads for their self-correction; all sequence information came from the PacBio reads themselves. We used the "model genome" (a human segmental duplication) in these assembly comparisons because this genome is at the same time computationally tractable, and a challenging assembly problem; hybrid assembly is too slow to use in a complex genome (e.g., *Drosophila*), and bacterial genomes are too easy to allow useful comparisons between assembly methods (*e.g.*, even the standard MHAP achieves essentially perfect assemblies). To make datasets as parallel as possible, the Illumina reads (100 bp, 100-fold coverage, single-ended) used in the hybrid assembly (and DCA) were simulated without error from the reference sequence of the "model genome" (the valid *k*-mer list used with the model genome was also error-free).

As shown in Supplemental Table S10, DCA is better than both standard MHAP (which only uses self-correction of the long reads) and standard hybrid assembly (which only uses correction of the long reads with Illumina reads). This result is quite expected, since DCA combines the two types of correction. The critical comparison is between LH *k*-mer validation and DCA, and it shows that LH *k*-mer validation is superior both in terms of assembly breaks (*i.e.*, number of contigs produced) and sequence duplications. Indeed, LH *k*-mer validation has zero misassemblies of both types, whereas DCA introduces three assembly breaks and duplicates 7.2% of the sequence.

The above results shows that LH *k*-mer validation uses more efficiently the information provided by the short reads; using the valid *k*-mers (extracted from the short reads) to guide the alignment of the long reads in the self-correction is most beneficial, when compared to direct correction of the long reads with the short reads. The likely cause of this difference is the preservation of the "phase" of the sequence family variants during read correction: Illumina reads can only do this for sites that are less than 100 bp apart (*i.e.*, the read length), whereas long reads (with properly guided alignments) can do this across thousands of base pairs.

PBcR assemblies with different memory parameters.

While using different servers to assemble the same genome with the same method (say, standard MHAP), we found fairly large differences in the assembly results such as N50 and largest contig. The cause traces to the ovlMemory parameter, which specifies the amount of memory allocated to the MHAP step of the PBcR pipeline. MHAP, as most long read aligners, split the raw reads into subsets that fit into

the available memory, and the ovlMemory parameter controls the size of these subsets. Perhaps naively, it seemed to us that this would not affect the assembly result. After running may assemblies, some of them showed in Supplemental Table S13, we found the following: (i) Assemblies done in different servers using the same ovlMemory had very small differences (compare assemblies #1 and # 2; #8, #9 and #10; #11 and #12); (ii) The effect of ovlMemory seems unpredictable: in some cases reducing it seem to improve the assemblies (as judged by N50 and largest contig; #2, #3, #4), whereas the opposite happens in other cases (#5 and #6).

The above reported effects may seem small, but they are relevant for two reasons. First, the initial version of our code stored the valid *k*-mers in a data structure (a HashSet) that consumed a lot of memory (27 Gb for *Drosophila*); when we compared the standard MHAP with *k*-mer validation they were actually using different ovlMemory values, and hence the differences observed between the assemblies were biased in unpredictable directions. The code used in the present work uses a more economical data structure (a BitSet, which uses less than 3 Gb for any genome), so it is possible to use exactly the same ovlMemory (set at 96 Gb) for all runs. Second, we found in *Arabidopsis* that changing ovlMemory from 96 Gb to 56 Gb caused a gross misassembly: a quimeric contig containing sequences from chromosomes 3 and 5 (Supplemental Fig. S7). All *Arabidopsis* assemblies reported in the pre-print of the present work (http://dx.doi.org/10.1101/053256) used ovlMemory=56Gb, and had this and similar misassemblies. This explains why the largest contigs reported there were larger than chromosome arm sizes (*i.e.*, it probably was an artifact, and not the exciting result that the assemblies spanned the centromere).

We do not have an explanation for the effect of ovlMemory parameter, in particular regarding to the Arabidopsis misassembly. As PBcR has been recently upgraded, it seems more useful to investigate this issue with the new pipeline (Canu)(Koren et al. 2016).

Supplemental Discussion

Possible improvements in the *k*-mer validation.

There are several avenues that may lead to further improvements in assembly contiguity. First, nearly all current long read assemblers employ two rounds of read alignment and correction (the first deals with the raw reads, and the second with the corrected reads), and we modified only the aligner of the first round (the MHAP program). It is likely that the second round of read alignment and correction, which uses a different program and is aimed to deal with Sanger-like error rates, would also benefit from the stringent use of single-copy *k*-mers as seeds for overlap detection.

Second, it will be interesting to test the *k*-mer validation procedure in the other long-read aligners (DALIGNER and BLASR) because MHAP is a "probabilistic" aligner that randomly samples a subset of *k*-mers, whereas both DALIGNER and BLASR deal with all *k*-mers (Chaisson and Tesler 2012; Myers 2014; Berlin et al. 2015). It is likely that *k*-mer validation will increase the speed and accuracy of these aligners.

Third, the read correction algorithms (*e.g.*, falconsense and pbdagcon) may be modified to avoid the introduction of non-valid *k*-mers while correcting the reads.

Fourth, there is the issue of the Illumina-derived *k*-mer list. As detailed in the Supplemental Results, we did a limited exploration of the low frequency cut-off (which should remove error *k*-mers) and high frequency cut-off (which should spare only single-copy *k*-mers), and hence a more throughout investigation may be beneficial. Sex-chromosomes also pose particular questions that we have not fully addressed (Supplemental Results).

Fifth, nearly uniform coverage is considered a major advantage of PacBio sequencing, and hence the strong bias we found in the *Drosophila* data warrants further investigations. It will also be interesting to look at Oxford Nanopore datasets for complex genomes.

Finally, recently the PBcR/Celera Assembler pipeline was upgraded to a new pipeline called Canu (Koren et al. 2016) .It will be very interesting to implement the *k*-mer validation procedure on it.

These developments are beyond the scope of the present paper; we are pursuing some of them now.

Supplemental References

Berlin K, Koren S, Chin CS, Drake JP, Landolin JM, Phillippy AM. 2015. Assembling large genomes with single-molecule sequencing and locality-sensitive hashing. *Nature biotechnology* **33**(6): 623-630.

Chaisson MJ, Huddleston J, Dennis MY, Sudmant PH, Malig M, Hormozdiari F, Antonacci F, Surti U, Sandstrom R, Boitano M et al. 2015. Resolving the complexity of the human genome using single-molecule sequencing. *Nature* **517**(7536): 608-611.

Chaisson MJ, Tesler G. 2012. Mapping single molecule sequencing reads using basic local alignment with successive refinement (BLASR): application and theory. *BMC bioinformatics* **13**: 238.

dos Santos G, Schroeder AJ, Goodman JL, Strelets VB, Crosby MA, Thurmond J, Emmert DB, Gelbart WM, FlyBase C. 2015. FlyBase: introduction of the Drosophila melanogaster Release 6 reference genome assembly and large-scale migration of genome annotations. *Nucleic Acids Res* **43**(Database issue): D690-697.

Gurevich A, Saveliev V, Vyahhi N, Tesler G. 2013. QUAST: quality assessment tool for genome assemblies. *Bioinformatics* **29**(8): 1072-1075.

Gutzwiller F, Carmo CR, Miller DE, Rice DW, Newton IL, Hawley RS, Teixeira L, Bergman CM. 2015. Dynamics of Wolbachia pipientis Gene Expression Across the Drosophila melanogaster Life Cycle. *G3* **5**(12): 2843-2856.

Koren S, Schatz MC, Walenz BP, Martin J, Howard JT, Ganapathy G, Wang Z, Rasko DA, McCombie WR, Jarvis ED et al. 2012. Hybrid error correction and de novo assembly of single-molecule sequencing reads. *Nature biotechnology* **30**(7): 693-700.

Koren S, Walenz BP, Berlin K, Miller JR, Phillippy AM. 2016. Canu: scalable and accurate long-read assembly via adaptive k-mer weighting and repeat separation. *bioRxiv*: 071282.

Kurtz S, Phillippy A, Delcher AL, Smoot M, Shumway M, Antonescu C, Salzberg SL. 2004. Versatile and open software for comparing large genomes. *Genome Biol* **5**(2): R12.

Li H, Durbin R. 2009. Fast and accurate short read alignment with Burrows-Wheeler transform. *Bioinformatics* **25**(14): 1754-1760.

Li H, Handsaker B, Wysoker A, Fennell T, Ruan J, Homer N, Marth G, Abecasis G, Durbin R. 2009. The Sequence Alignment/Map format and SAMtools. *Bioinformatics* **25**(16): 2078-2079.

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

Myers G. 2014. Efficient local alignment discovery amongst noisy long reads. In *Algorithms in Bioinformatics*, pp. 52-67. Springer.

Zook JM, Catoe D, McDaniel J, Vang L, Spies N, Sidow A, Weng Z, Liu Y, Mason CE, Alexander N et al. 2016. Extensive sequencing of seven human genomes to characterize benchmark reference materials. *Sci Data* **3**: 160025.