


## Genome analysis

# WGA-LP: a pipeline for whole genome assembly of contaminated reads

N. Rossi <sup>1,\*</sup>, A. Colautti<sup>2</sup>, L. Iacumin<sup>2</sup> and C. Piazza<sup>1</sup>

<sup>1</sup>Department of Mathematics, Computer Science, and Physics, University of Udine, 33100 Udine, Italy and <sup>2</sup>Dipartimento di Scienze Agroalimentari, Ambientali e Animali, University of Udine, 33100 Udine, Italy

\*To whom correspondence should be addressed.

Associate Editor: Can Alkan

Received on July 27, 2021; revised on September 22, 2021; editorial decision on October 11, 2021; accepted on October 15, 2021

## Abstract

**Summary:** Whole genome assembly (WGA) of bacterial genomes with short reads is a quite common task as DNA sequencing has become cheaper with the advances of its technology. The process of assembling a genome has no absolute golden standard and it requires to perform a sequence of steps each of which can involve combinations of many different tools. However, the quality of the final assembly is always strongly related to the quality of the input data. With this in mind we built WGA-LP, a package that connects state-of-the-art programs for microbial analysis and novel scripts to check and improve the quality of both samples and resulting assemblies. WGA-LP, with its conservative decontamination approach, has shown to be capable of creating high quality assemblies even in the case of contaminated reads.

**Availability and implementation:** WGA-LP is available on GitHub (<https://github.com/redsnc/WGA-LP>) and Docker Hub (<https://hub.docker.com/r/redsnc/wgalp>). The web app for node visualization is hosted by shinyapps.io (<https://redsnc.shinyapps.io/ContigCoverageVisualizer/>).

**Contact:** olocin.issor@gmail.com

**Supplementary information:** [Supplementary data](#) are available at *Bioinformatics* online.

## 1 Introduction

A currently active challenge in the context of whole genome assembly (WGA) for bacterial genomes is to produce reliable WGAs that are contaminant free (Chun *et al.*, 2018; Steinegger and Salzberg, 2020; Del Angel *et al.*, 2018). In this context, we built WGA-LP, a pipeline that includes different strategies to guide the users in producing higher quality WGAs of prokaryotic genomes, by also including specific features to control possible contamination. Moreover, its workflow is structured to assist in the quality evaluation of the results of each step of the pipeline by providing useful plots and summaries. The current state-of-the-art for decontamination consists in the use of Kraken2 (Wood *et al.*, 2019), a software for read origin imputation, and of pipelines like ProDeGe (Tennessen *et al.*, 2016) and SIDR (Fierst and Murdock, 2017). This last is, however, meant for eukaryotic genomes.

## 2 Software description

WGA-LP software is built to be used from the command line. The procedures of the pipeline are organized by functionality and have a consistent syntax for argument passing. More details are available in the [Supplementary Material](#), on the GitHub and Docker Hub web pages of the tool.

WGA-LP performs many steps that can be run independently. In order to execute the whole workflow, the user is required to provide the raw reads (.fastq) and, optionally, the references that should be used for decontamination (.fasta). All the other input files can be produced using WGA-LP commands. Check the [Supplementary Material](#) for a complete explanation of all the input parameters for WGA-LP.

The first step of WGA-LP has the role of assessing the quality of the input reads and detecting possible contamination sources. To this end, WGA-LP relies on Trimmomatic (Bolger *et al.*, 2014), FastQC (Andrews, 2010), Kraken2 and Bracken (Lu *et al.*, 2017). The trimming step is fully configurable so that the user can choose the right approach for his data.

A novel contribution of WGA-LP is its decontamination procedure, that exploits a custom script including calls to three programs: BWA mem (Li, 2013), Samtools (Li *et al.*, 2009) and Bazam (Sadedin and Oshlack, 2019). The inputs for the decontamination are the raw reads and two sets of references, one for the target organism and one for the contaminants. We first determine all the reads that map to any contaminant reference, then among such reads we filter the ones that map to any reference genome of the target organism. This gives us the set of reads that we consider to be from the contaminant and we remove them from the original set. The combination of BWA mem, Samtools (view) and Bazam allows us to simply perform a loop in which fastq

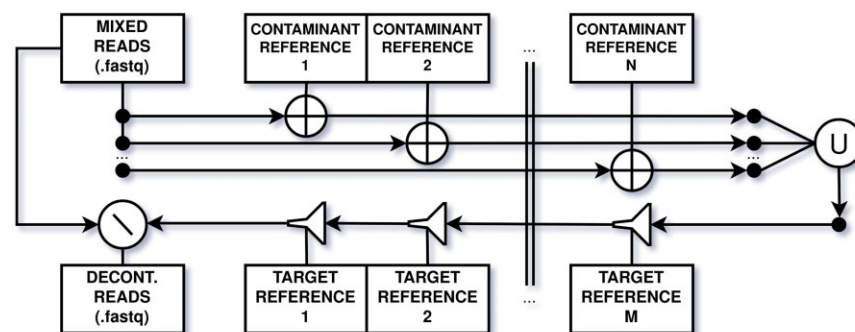


Fig. 1. The decontamination procedure. Input reads are mapped against each reference of the contaminant independently [first three wires from left to are then merged together (Union,  $\cup$ )] and gradually filtered (last wire from right to left), with the effect of removing all the reads that map to any target reference. The final decontaminated reads are extracted by set difference ( $\setminus$ ) using the original input set

reads are mapped to a reference obtaining a bam file. Such file is then processed with Samtools to extract mapped/nonmapped reads. The mapped reads are finally converted back to fastq format through Bazam. The presented decontamination approach is conservative and reduces the probability of discarding reads of the target organism. More details about this approach are presented in Figure 1 and in the Supplementary Materials.

This part of the pipeline can be used as a standalone program and can be combined with any other program for WGA.

WGA-LP natively supports SPAdes (Bankevich *et al.*, 2012) and Minia (Chikhi and Rizk, 2012) assemblers. SPAdes is currently a common choice for bacterial WGA, while Minia is a very simple and fast assembler. The other steps of WGA-LP can support any assembler that includes in its outputs a fasta formatted assembly and a fastq assembly graph [required only for putative plasmid search with Recycler (Rozov *et al.*, 2017)].

We use the term node to refer to an assembled segment of contiguous DNA (either a scaffold or a contig) produced by an assembler. WGA-LP includes custom scripts to help in the visualization of node coverage by postprocessing the output of Samtools depth. This allows to produce coverage plots (computed by remapping the reads to the assembled genome) that can be helpful in finding anomalies, such as prophage insertions in the genome. Moreover, WGA-LP provides a web app and tools for nodes (and reads) selection that can improve the decontamination results. These act by exploiting the assembly process as it tends to assemble nodes with reads of the same organism. Such procedures are well fitted to be combined with Kraken2, since this tool can point out problematic nodes, that can be then further evaluated with BLAST alignment (Altschul *et al.*, 1990) in order to validate user selections.

For node reordering, WGA-LP uses the ContigOrderer option from Mauve aligner (Rissman *et al.*, 2009). This step requires to provide a reference for the target organism.

WGA-LP offers interfaces to two programs that extract putative plasmids: plasmidSPAdes (Antipov *et al.*, 2016) and Recycler. It is highly recommended to check the results of these tools using BLAST.

WGA-LP includes three programs to evaluate the quality of the final result of the pipeline: Quast (Gurevich *et al.*, 2013) CheckM (Parks *et al.*, 2015) and Merquy (Rhie *et al.*, 2020). Especially, CheckM is useful to verify the completeness and contamination of the produced assembly.

For the annotation, WGA-LP interfaces with Prokka (Seemann, 2014) in order to create NCBI compliant assemblies. This can be considered as the final output of the pipeline and can be used for downstream analysis.

### 3 Results

We tested WGA-LP pipeline on real and simulated data (see Section 4) and we have shown how its workflow was effective in producing a high quality WGA even in the challenging scenario of a contaminated genome, with improvements in comparison with less curated

approaches (see Supplementary Material). Finally, we extended the comparison to include ProDeGe, another state-of-the-art decontamination procedure. ProDeGe alone was not able to filter large nodes of the contaminant; however, it was possible to use WGA-LP procedures based on kraken2 classification to refine the resulting assembly, achieving comparable results with our pipeline. However, also in this case, our tool performed better on the elimination of the shorter nodes, keeping those that, in a further check, were classified from the target genome by BLAST alignment.

Relying on ART (Huang *et al.*, 2012), we ran a set of simulations to assess the performance of our decontamination procedure in two different settings. In the first, we investigated the impact of the phylogenetic distance of the contaminant on the effectiveness of our approach, while in the second, we addressed the effect of different contamination levels. In every setting, WGA-LP has proven to be effective in removing the reads of the contaminant while preserving the reads from the target. More details about these simulations can be found in the Supplementary Material.

Both the decontamination procedure and the node selection, that are the core of our pipeline, can be integrated in any other pipeline for WGA, in the preprocessing and postprocessing phases.

### 4 Data availability

The testing reads, from the organism *Lactocaseibacillus rhamnosus*, heavily contaminated with *Pediococcus acidilactici*, are available in the NCBI's Sequence Read Archive at <https://www.ncbi.nlm.nih.gov/bioproject/?term=prjna749304> and can be accessed with the accession number SRR15265000, associated to the BioProject PRJNA749304. WGA-LP includes utilities to quickly access all the resources needed to reproduce the tests presented in this paper. All website and links in this paper and in the Supplementary Material were accessed on the July 25, 2021.

### Funding

This work was supported by the Ministero dell'Università e della Ricerca, Rome, Italy, FIRB n. RBFR107VML.

Conflict of Interest: none declared.

### References

- Altschul, S.F. *et al.* (1990) Basic local alignment search tool. *J. Mol. Biol.*, **215**, 403–410.
- Andrews, S. (2010) Babraham bioinformatics-FastQC a quality control tool for high throughput sequence data. <https://www.bioinformatics.babraham.ac.uk/projects/fastqc> (25 July 2021, date last accessed).
- Antipov, D. *et al.* (2016) plasmidSPAdes: assembling plasmids from whole genome sequencing data. *Bioinformatics*, **32**, 3380–3387.
- Bankevich, A. *et al.* (2012) SPAdes: a new genome assembly algorithm and its applications to single-cell sequencing. *J. Comput. Biol.*, **19**, 455–477.

- Bolger, A.M. et al. (2014) Trimmomatic: a flexible trimmer for illumina sequence data. *Bioinformatics*, 30, 2114–2120.
- Chikhi, R. and Rizk, G. (2012) Space-efficient and exact de Bruijn graph representation based on a bloom filter. In: *WABI, Lecture Notes in Computer Science*, Vol. 7534. Springer, pp. 236–248.
- Chun, J. et al. (2018) Proposed minimal standards for the use of genome data for the taxonomy of prokaryotes. *Int. J. Syst. Evol. Microbiol.*, 68, 461–466.
- Del Angel, V.D. et al. (2018) Ten steps to get started in genome assembly and annotation. *F1000Research*, 7, 148.
- Fierst, J.L. and Murdock, D.A. (2017) Decontaminating eukaryotic genome assemblies with machine learning. *BMC Bioinform.*, 18, 1–16.
- Gurevich, A. et al. (2013) Quast: quality assessment tool for genome assemblies. *Bioinformatics*, 29, 1072–1075.
- Huang, W. et al. (2012) Art: a next-generation sequencing read simulator. *Bioinformatics*, 28, 593–594.
- Li, H. (2013) Aligning sequence reads, clone sequences and assembly contigs with bwa-mem. arXiv:1303.3997.
- Li, H. et al. (2009) The sequence alignment/map format and Samtools. *Bioinformatics*, 25, 2078–2079.
- Lu, J. et al. (2017) Bracken: estimating species abundance in metagenomics data. *PeerJ Comput. Sci.*, 3, e104.
- Parks, D.H. et al. (2015) CheckM: assessing the quality of microbial genomes recovered from isolates, single cells, and metagenomes. *Genome Res.*, 25, 1043–1055.
- Rhie, A. et al. (2020) Merqury: reference-free quality, completeness, and phasing assessment for genome assemblies. *Genome Biol.*, 21, 1–27.
- Rissman, A.I. et al. (2009) Reordering contigs of draft genomes using the Mauve aligner. *Bioinformatics*, 25, 2071–2073.
- Rozov, R. et al. (2017) Recycler: an algorithm for detecting plasmids from de novo assembly graphs. *Bioinformatics*, 33, 475–482.
- Sadedin, S.P. and Oshlack, A. (2019) Bazam: a rapid method for read extraction and realignment of high-throughput sequencing data. *Genome Biol.*, 20, 1–6.
- Seemann, T. (2014) Prokka: rapid prokaryotic genome annotation. *Bioinformatics*, 30, 2068–2069.
- Steinegger, M. and Salzberg, S.L. (2020) Terminating contamination: large-scale search identifies more than 2,000,000 contaminated entries in genbank. *Genome Biol.*, 21, 1–12.
- Tennessen, K. et al. (2016) ProDeGe: a computational protocol for fully automated decontamination of genomes. *ISME J.*, 10, 269–272.
- Wood, D.E. et al. (2019) Improved metagenomic analysis with kraken 2. *Genome Biol.*, 20, 1–13.