

Cloud-based Approach on Genetic Data Imputation Parameters' Optimization

Pavlo Horun^{1,*†} and Chirstine Strauss^{2†}

¹ Lviv Polytechnic National University, Kniazia Romana St. 5, 79000 Lviv, Ukraine

² University of Vienna, Oskar Morgenstern Platz 1, 1090 Vienna, Austria

Abstract

The imputation process for genetic data is cost and time-intensive, primarily due to the high complexity of the methods involved, and the substantial volume of data processed. A thorough performance evaluation of the imputation algorithms such as Beagle, AlphaPlantImpute, LinkImputeR, MACH and others shows that while some algorithms are highly accurate, they are often computationally expensive. Being widely used, they have multiple input parameters which impact the quality and accuracy of the imputation. Traditional machine learning techniques for parameter optimization like grid search and randomized search become inefficient in high-dimensional parameter spaces, leading to prohibitive computational costs, especially in large-scale applications. Our study proposes the cloud-based approach for input parameters optimization by using Bayesian optimization with consecutive Domain Reduction Transformer (DRT). Described algorithm and developed library allow users to find the optimal input parameters for the data imputation in a more flexible way.

Keywords

Bayesian optimization, parameters optimization, data imputation, Beagle, cloud technologies, distributed calculations, bioinformatics

1. Introduction

The modern sequencing technologies made the analysis of genetic variations to be significantly advanced. However, the vast amounts of data generated during these processes often contain missing values, particularly in the field of plant genetics. To address these gaps, a variety of machine learning and statistical tools have been employed to impute missing genetic data (GD) efficiently [1-5]. These methods vary in their performance, and each comes with its strengths and limitations.


The imputation process for genetic data is cost and time-intensive, primarily due to the high complexity of the methods involved, and the substantial volume of data processed. Imputation tools such as Beagle [6-7], HBImpute [3], Impute, MACH [5], AlphaPlantImpute [8-9], MissForest, and LinkImputeR [10] offer flexibility in their input parameters. Being powerful and widely used, they have multiple input parameters, and each one impacts the quality and accuracy of the imputation [9, 11]. Furthermore, these parameters often have a wide range of acceptable values, and the relationships between these parameters and the resulting accuracy can be highly nonlinear and difficult to model. Therefore, performing a comprehensive search across all possible parameter combinations to determine the optimal settings would be computationally prohibitive, requiring almost infinite amount of time and resources.

Traditional machine learning techniques for parameter optimization, such as grid search or even randomized search, quickly become inefficient due to the high dimensionality of the parameter space and the large number of evaluations needed. Such approaches often scale poorly

* Corresponding author.

† These authors contributed equally.

 pavlo.p.horun@lpnu.ua (P. Horun); christine.strauss@univie.ac.at (C. Strauss)

 0009-0008-4296-5560 (P. Horun); 0000-0003-0276-3610 (C. Strauss)



© 2024 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

with the number of parameters and their wide ranges, leading to significantly increased computational costs, which is unfeasible in large-scale applications.

In the article [12] authors addressed the challenges of GD imputation in bioinformatics, specifically focusing on the limitations of existing software tools such as Beagle, AlphaPlantImpute and others. The core issues are:

1. The complexity and unclear nature of input parameters, which often make optimization time-consuming. There is some lack of proper investigation of time and computational resource requirements in existing literature.
2. High time and resource costs for GD imputation. Traditional tools for GD imputation have high computational costs, especially when using large datasets.
3. The accuracy of missing GD imputation is crucial for subsequent bioinformatics analysis. However, optimizing accuracy often results in increased computation time, which the article attempts to balance in their “Algorithm A – Time Optimization”. This algorithm focuses on reducing the imputation time using cloud-based distributed computing, which showed significant speed improvements ranging from 47% to 87% based on the size of the dataset. The use of parallelization in the cloud environment is well-detailed, showing how splitting the data by chromosomes and distributing processing across multiple virtual nodes leads to significant time savings.

Based on described above, the next goal is to optimize input parameters in a more efficient way. One of the biggest issues here is time required for each single imputation run. In such case Bayesian optimization offers a more efficient alternative way by incorporating prior knowledge and sequential learning to iteratively select promising parameter combinations. Bayesian optimization builds a probabilistic model of the objective function, typically a Gaussian process, which helps in approximating the relationship between the input parameters and the output (e.g., imputation accuracy). By balancing exploration (testing unknown regions of the parameter space) and exploitation (focusing on regions known to yield good results), Bayesian optimization can converge to an optimal or suboptimal solution with significantly fewer evaluations compared to exhaustive or random search methods.

Using Bayesian optimization in this context allows for a more targeted exploration of the parameter space, thereby reducing the overall computational cost and time required for parameter tuning. It focuses on the most promising regions of the parameter space, which leads to a faster convergence toward effective parameter settings. This is particularly advantageous given the constraints on computational resources and the need for timely imputation results in large-scale genetic studies.

2. Input parameters optimization

In [12] authors suggested the approach to reduce imputation time in distributed cloud environment. To continue this research we developed the library that uses Bayesian optimization (BO) [13] with consecutive Domain Reduction Transformer (DRT) [14] to automate input parameters optimization using benefits of distributed environment. In addition, it is possible to restore parameters optimization process from the last successful or even failed step. It can lead to near-optimal settings for tools like Beagle without the prohibitive costs associated with exhaustive parameter searches. Consequently, researchers can achieve higher accuracy in their imputations while maintaining a reasonable resource usage that facilitates more efficient genetic analysis.

The main flow consists of the following steps:

- take original file;
- prepare set of files by modifying original one (split it by chromosomes, randomly remove some data for different missing ratio (MR) etc.);

- upload the artifacts on AWS S3 storage (storage type and cloud provider could be different);
- in a distributed environment run Bayesian Optimization using Beagle with consecutive DRT;
- iteratively collect and process the logs (optimization, imputation metrics, beagle logs);
- analyze obtained results.

This high-level algorithm is depicted on the Figure 1.

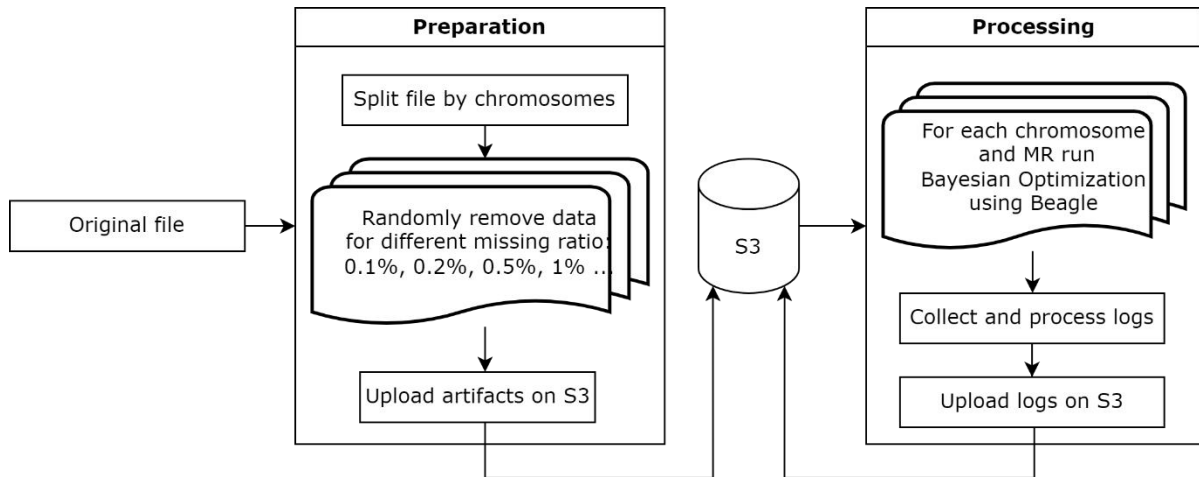


Figure 1. High-level parameters optimization flow

2.1. File preparation phase

During the file preparation phase it's important to split original file by chromosomes (based on the performance comparison conducted in [12], it decreases imputation time significantly) and collect set of files with the different MR (like 0.1%, 0.2%, 0.5%, 1%, 3%, 5%, 10% etc.). At the end, it will be useful for more comprehensive analysis and comparison to get more valuable insights.

So, the key steps on this phase are:

- Splitting original file by chromosomes (this will decrease further imputation time).
- For each MR randomly remove the data several times (this allows us to reduce bias and have statistically independent results that could be averaged with a higher level of confidence). Doing this 3 times for each case should be enough for further averaging.
- Upload prepared artifacts on S3.

Described algorithm represented on the **Figure 2**.

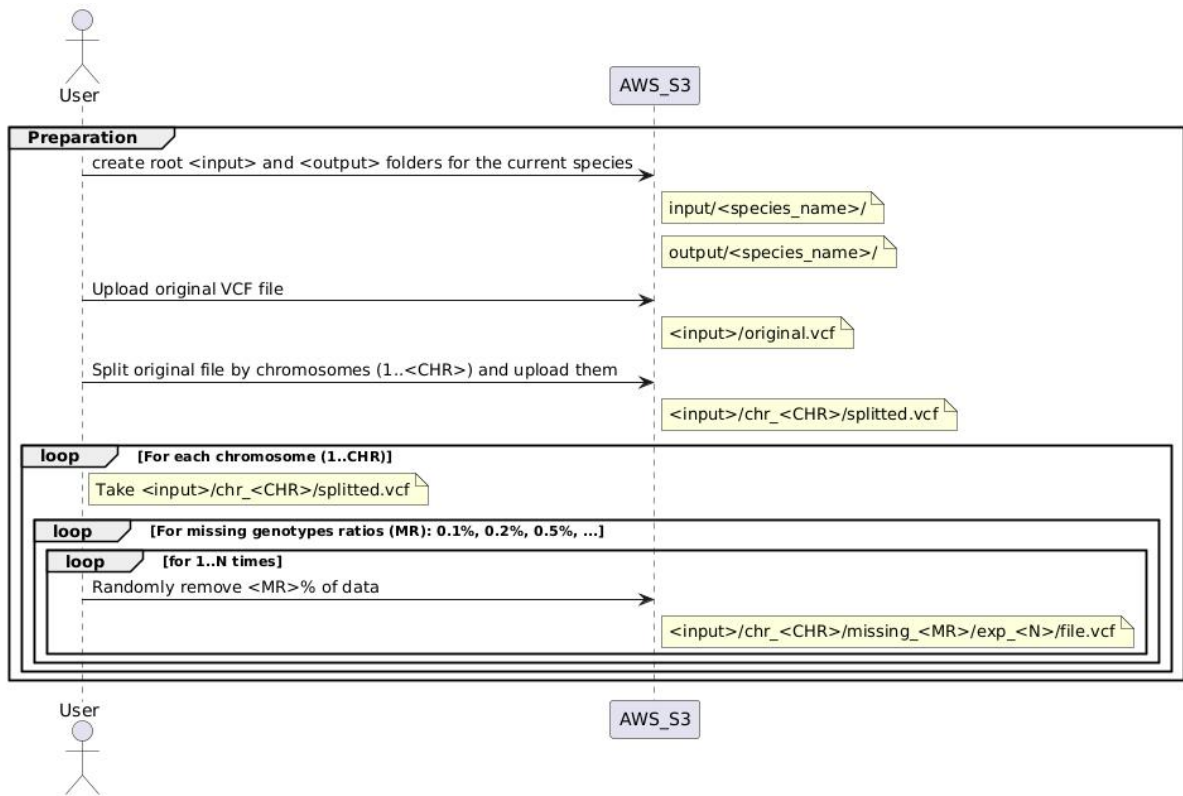


Figure 2. Input files preparation algorithm

2.2. Parameters' optimization phase

After uploading prepared artifacts on storage, user needs to configure dedicated cloud environment (you may refer to the subsection 2.2 from [12]). During this phase each file is downloaded on each virtual host and run parallel Bayesian optimization task using DRT with multiple Beagle runs. When done – upload results with the corresponding logs on storage.

Described algorithm is demonstrated on the Figure 3.

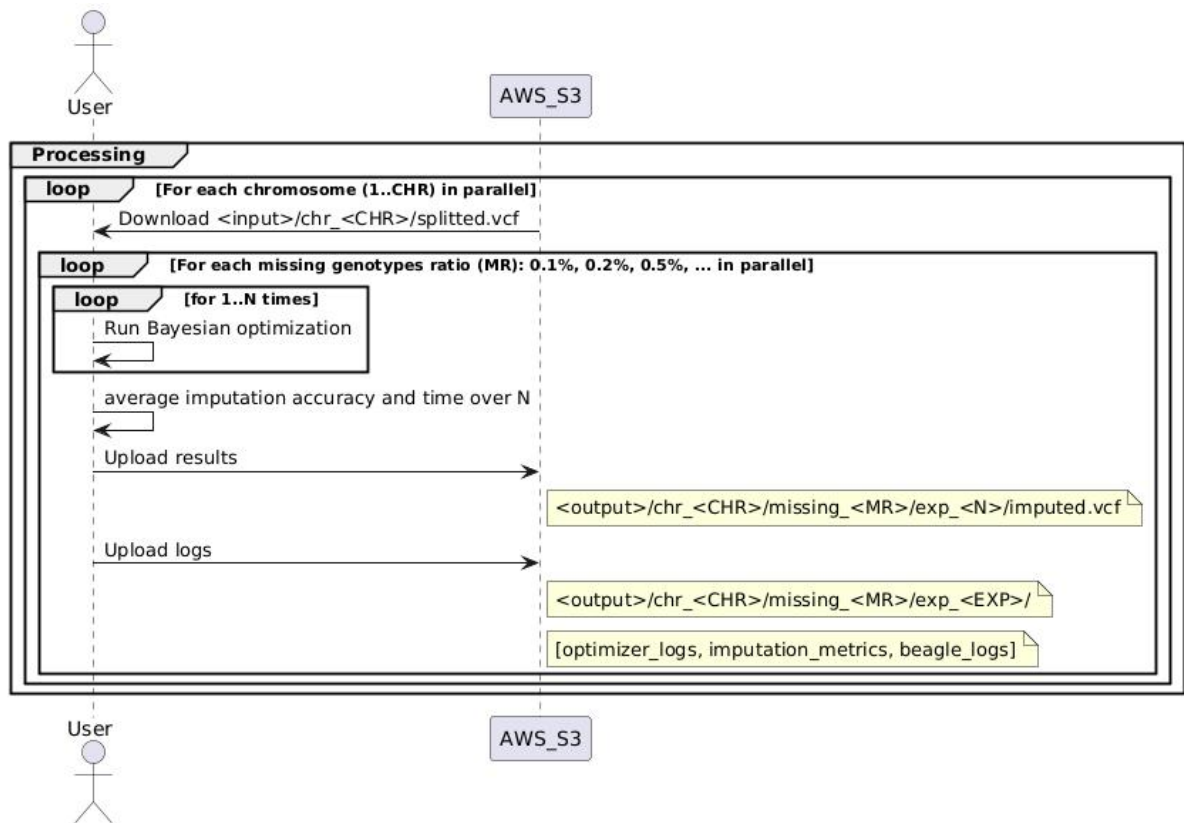


Figure 3. Parameters' optimization algorithm

The developed library performs such actions by itself without user's interruption. One of the key advantages of it is that user can continue parameters optimization even after successful or failed steps. This plays crucial role since accuracy achieved after X epochs could be still insufficient.

2.3. Beagle parameters selection

Being widely used, Beagle tool accepts numerous input parameters and each one may significantly influence the quality and accuracy of the imputation [9, 11]. The latest version of Beagle 5.4 (release 06Aug24.a91) accepts:

- Data parameters – most of them are required and contain file paths (nothing to optimize here).
- Phasing parameters – used to control phasing accuracy (could be specified by the user).
- Imputation parameters. It worth to focus on the limited subset and specify rest of them based on each specific case.
- General parameters. The most impactful are “ne” and “window” (see description below). Reducing the window parameter will reduce the amount of memory required for the analysis.

Most of the mentioned parameters were investigated in [9] and then compared Beagle and AlphaPlantImpute2. Unfortunately, there is a lack of description for exact optimal parameters investigation. Nevertheless, authors admitted that some parameters (like “ne” and “iterations”) may cause too long time for imputation and based on their results such approach is not very suitable for the large-scale datasets. Potentially, using developed library it is possible to perform more versatile parameters optimization to find other optimal (or sub-optimal) points that provides similar accuracy within reasonable amount of time. The set of Beagle parameters that worth to optimize is listed in the Table 1. Rest of the parameters should be specified depending on the current use case.

Table 1

Beagle input parameters for optimization

Parameter	Description	Type, default
burnin	The maximum number of burnin iterations used to estimate an initial haplotype frequency model for inferring genotype phase	> 0, default = 3
iterations	The number of iterations used to estimate genotype phase	> 0, default = 12
phase-states	The number of model states used to estimate genotype phase	> 0, default = 280
imp-states	The number of model states used to impute ungenotyped markers	> 0, default = 1600
imp-segment	The minimum cM length of haplotype segments that will be incorporated in the Hidden Markov Model state space for a target haplotype	> 0, default = 6.0
imp-step	The length in cM of the step in centiMorgans used for detecting short IBS segments	> 0, default = 0.1
imp-nsteps	The number of consecutive steps (see imp-step argument) that will be considered when detecting long IBS segments	> 0, default = 7
cluster	The maximum cM distance between individual markers that are combined into an aggregate marker when imputing ungenotyped markers	≥ 0 , default = 0.005
ne	The effective population size (for unphased input genotypes)	> 0, default = 100000
window	The cM length of each sliding window	> 0, default = 40.0
overlap	The cM length of overlap between adjacent sliding windows	> 0, default = 2.0

Based on official Beagle 5.4 documentation, shorter “window” requires less memory, while “iterations” controls the trade-off between compute time and phasing accuracy.

3. Discussion

As mentioned before, one more useful (even crucial) feature introduced here is an ability to continue optimization from the last run. In case of long running imputation jobs this allows users to decide if existing imputation accuracy is sufficient or they want to continue parameters optimization. The core library that performs Bayesian optimization from [13] allows to use existing logs for further optimization. So, in the last case such algorithm will avoid running imputation

with already checked parameters values that saves time tremendously. In some cases, it could be very beneficial to continue parameters optimization with the modified general parameters.

Additional issue is the cost saving. To address it, so-called spot EC2 instances could be used as suggested in [12] (this saves up to 90% of costs compared to so-called "on-demand instances"). For example, users may stick with r5a.2xlarge (4 vCore and 32 Gb RAM) EC2 type. It worth to mention that the target EC2 type has no impact on the overall process – its main benefit is to increase optimization (and imputation) time performance. And having regular backups (dumps of the logs) allows users to continue optimization from the latest observed state. Above feature to continue optimization is extremely valuable since spot instances could be terminated at any time and whole progress will be lost.

Continuous failures caused by sporadic spot instances termination could be managed by the appropriate cloud orchestration. It was briefly described in [12] to address network instability and synchronization issues. More advanced orchestration layer could enhance the robustness and fault tolerance of whole optimization flow. From one side, it increases costs due to systems' complexity and more resources to be used. But from the other side, including orchestration, such as Kubernetes, could improve reliability and further optimized resource allocation and synchronization.

One more interesting topic is a broader species scope. Extensive testing across other recent imputation models and different crop species could provide more insights into the general applicability and scalability of the proposed methods. This is a part of the future research.

4. Conclusions and future work

The optimization of genetic data imputation using distributed cloud technologies represents a significant step forward in the analysis of large-scale biological datasets. The proper selection of optimal input parameters is a key to improve imputation accuracy. Described method and library allow to automate their tuning using cloud-based environment and Bayesian optimization with consecutive Domain Reduction Transformer. This allows users to find the optimal input parameters in a more flexible way for different datasets.

More generic scenario consists of conducting preliminary experiments for predefined input files using proposed approach and developed library to find set of optimal imputation parameters (like "profiles") for Beagle. This may give some insights on how accuracy and time depend on input file characteristics (for instance, species type, variants and lines count, missing genotypes ratio, heterozygosity frequency (HF), minor allele frequency (MAF) etc.). Such "profiles" could be used then on a regular basis for similar input files that saves time and resources to achieve optimal (or at least satisfied for the user) imputation accuracy.

As a next step it is reasonable to focus efforts addressing above question, refining suggested algorithms and exploring additional methods for adaptive parameter tuning using distributed environments.

Declaration on Generative AI

During the preparation of this work, the authors used ChatGPT in order to: Grammar and spelling check, Paraphrase and reword. After using this tool/service, the authors reviewed and edited the content as needed and takes full responsibility for the publication's content.

References

- [1] N. Bhandari, R. Walambe, K. Kotecha, and S. P. Khare, "A comprehensive survey on computational learning methods for analysis of gene expression data," *Front. Mol. Biosci.*, vol. 9, p. 907150, Nov. 2022, doi: 10.3389/fmolb.2022.907150.

- [2] T. Emmanuel, T. Maupong, D. Mpoeleng, T. Semong, B. Mphago, and O. Tabona, "A survey on missing data in machine learning," *J Big Data*, vol. 8, no. 1, p. 140, Oct. 2021, doi: 10.1186/s40537-021-00516-9.
- [3] T. Pook, A. Nemri, E. G. Gonzalez Segovia, D. Valle Torres, H. Simianer, and C.-C. Schoen, "Increasing calling accuracy, coverage, and read-depth in sequence data by the use of haplotype blocks," *PLoS Genet*, vol. 17, no. 12, p. e1009944, Dec. 2021, doi: 10.1371/journal.pgen.1009944.
- [4] T. Pook *et al.*, "Improving Imputation Quality in BEAGLE for Crop and Livestock Data," *G3 Genes/Genomes/Genetics*, vol. 10, no. 1, pp. 177–188, Jan. 2020, doi: 10.1534/g3.119.400798.
- [5] H. Alipour, G. Bai, G. Zhang, M. R. Bihamta, V. Mohammadi, and S. A. Peyghambari, "Imputation accuracy of wheat genotyping-by-sequencing (GBS) data using barley and wheat genome references," *PLoS ONE*, vol. 14, no. 1, p. e0208614, Jan. 2019, doi: 10.1371/journal.pone.0208614.
- [6] B. L. Browning and S. R. Browning, "Genotype Imputation with Millions of Reference Samples," *The American Journal of Human Genetics*, vol. 98, no. 1, pp. 116–126, Jan. 2016, doi: 10.1016/j.ajhg.2015.11.020.
- [7] B. L. Browning, Y. Zhou, and S. R. Browning, "A One-Penny Imputed Genome from Next-Generation Reference Panels," *The American Journal of Human Genetics*, vol. 103, no. 3, pp. 338–348, Sep. 2018, doi: 10.1016/j.ajhg.2018.07.015.
- [8] L. Chen *et al.*, "Genotype imputation for soybean nested association mapping population to improve precision of QTL detection," *Theor Appl Genet*, vol. 135, no. 5, pp. 1797–1810, May 2022, doi: 10.1007/s00122-022-04070-7.
- [9] T. Niehoff, T. Pook, M. Gholami, and T. Beissinger, "Imputation of low-density marker chip data in plant breeding: Evaluation of methods based on sugar beet," *The Plant Genome*, vol. 15, no. 4, Dec. 2022, doi: 10.1002/tpg2.20257.
- [10] N. Munyengwa *et al.*, "Optimizing imputation of marker data from genotyping-by-sequencing (GBS) for genomic selection in non-model species: Rubber tree (*Hevea brasiliensis*) as a case study," *Genomics*, vol. 113, no. 2, pp. 655–668, Mar. 2021, doi: 10.1016/j.ygeno.2021.01.012.
- [11] A. Palanivinayagam and R. Damaševičius, "Effective Handling of Missing Values in Datasets for Classification Using Machine Learning Methods," *Information*, vol. 14, no. 2, p. 92, Feb. 2023, doi: 10.3390/info14020092.
- [12] L. Mochurad and P. Horun, "Improvement Technologies for Data Imputation in Bioinformatics," *Technologies*, vol. 11, no. 6, p. 154, Nov. 2023, doi: 10.3390/technologies11060154.
- [13] Fernando Nogueira, *Bayesian Optimization: Open source constrained global optimization tool for Python*. (2014). Python. Accessed: Jan. 10, 2024. [Online]. Available: <https://github.com/bayesian-optimization/BayesianOptimization>
- [14] N. Stander and K. J. Craig, "On the robustness of a simple domain reduction scheme for simulation-based optimization," *Engineering Computations*, vol. 19, no. 4, pp. 431–450, Jun. 2002, doi: 10.1108/02644400210430190.