ITERATIVE TWO-PASS ALGORITHM FOR MISSING DATA IMPUTATION IN SNP ARRAYS

CHRISTINE SINOQUET
Computer Science Institute of Nantes-Atlantic (Lina)
U.M.R. C.N.R.S. 6241, University of Nantes
2 rue de la Houssinière, BP 92208
44322 Nantes Cedex, France
christine.sinoquet@univ-nantes.fr

Received 27 January 2009
Revised 10 April 2009
Accepted 21 May 2009

Though nowadays high-throughput genotyping techniques’ quality improves, missing data still remains fairly common. Studies have shown that even a low percentage of missing SNPs is detrimental to the reliability of down-stream analyses such as SNP-disease association tests. This paper investigates the potentiality for improving the accuracy of an SNP inference method based on the algorithm formerly designed by Roberts and co-workers (Nature, 2007). This initial algorithm performs a single scan of an SNP array, inferring missing SNPs in the context of sliding windows. We have first designed a variant, KnnWinOpti, which fully exploits backward and forward dependencies between the overlapping windows and thus restores the genuine dependency of inference upon direction scanning. Our major contribution, algorithm SnpShuttle, therefore iterates bi-directional scanning to predict SNP values with more confidence. We have run simulations on realistic benchmarks built after the high resolution map of mouse strains published by the Perlegen Project. For each of the 20 mouse chromosomes and for missing data percentage varying in range 5%-30%, SnpShuttle has always been shown to increase yet high KnnWinOpti’s accuracies.

Keywords: Missing genotype inference; SNP imputation; sliding window; bi-directional scan of SNP array; nearest neighbor search.

1. Introduction

The simplest type of genetic polymorphism, Single Nucleotide Polymorphism (SNP), involves only changes in one nucleotide, which occurred generations ago within the DNA sequence. SNP variants are not to be confounded with mutations: they must be present in more than 1% of the population. To clarify ideas, we emphasize that one single individual can be uniquely defined by only 30 to 80 independent SNPs and unrelated individuals differ in about 0.1% of their 3.2 billion nucleotides. Compared with other kinds of DNA markers, SNPs are appealing because they are abundant, genetically stable and amenable to high-throughput automated analysis.
Consistently, advances in high-throughput SNP genotyping technologies lead the way to various downstream analyses.

However family- or population-based, association studies compare the frequencies of SNP variants in affected subjects and controls, in order to identify the SNPs related to a disease and therefore map the causal gene responsible for a monogenic disease. The great importance of SNPs in biomedical research extends to multifactorial etiology, that is the inheritance of a phenotypic characteristic (trait) that can be attributed to the interactions between two or more genes and their environment. In this case, the quantitative phenotype induced typically varies along a continuous gradient. Though not necessarily genes themselves, quantitative trait loci (QTLs) are stretches of DNA which are closely linked to the genes underlying the phenotype studied. The issue at stake is identifying combinations of genetic determinants which should accumulate among affected subjects, without being causal by themselves, when considered alone.

Moreover, DNA polymorphism is known to affect the reaction to pathogens, chemicals, as well as to drugs and vaccines in the case of animals. In particular, regarding the pharmacogenomics field, SNPs are thought to be key enablers in designing customized therapy: the purpose is discarding ineffective or toxic drugs and prescribing the optimal dose of the most appropriate medication, on the basis of patients’ SNP profiles.

In addition to medical uses, important economical profits depend on the reliability of SNP collections and further analyses of the latter. An illustration of this is gaining an understanding of multigenic and induced systemic resistance in plant breeding. Another example is livestock breeding programs, for which elucidating the genetic component of phenotypic variation aims at facilitating stable changes in production, fertility and health through selection.

The key to dissecting the genetic susceptibility of complex diseases, SNP data may unfortunately contain around 5% to 10% of undetermined markers — or missing calls. Together with other factors known to introduce a bias in association studies, ignoring the missing data may considerably undermine the identification of SNP–phenotype associations. Nonetheless, despite increasing improvements in reliability, generation delay and cost of genotyping techniques, re-genotyping the missing calls is not a practical alternative to dismissing subjects concerned with missing calls or discarding offending markers. Hence, missing data inference — or imputation — remains a hot research area.

The impact of missing SNPs on downstream analysis quality has mainly been evaluated for association studies. In their study, Croiseau and co-authors first check that missing data might lead to an important loss in power to detect association when the family-based linkage test studied is performed on the remaining complete data. Besides, another marker in linkage disequilibrium with the disease susceptibility site may be predicted as the most relevant site. An imputation strategy relying on trios has been shown to be efficient to increase the test power, thus pleading in favor of SNP inference. Analysing the performance of a strategy designed for SNP imputation-based association studies, Dai and co-workers incorporate information
on disease status and other covariates to improve the imputation.\textsuperscript{5} These authors evaluate the impact of missing data (5\%, 10\% and 15\%) on the prediction of interaction between two loci involved in developmental dyslexia. This impact is shown to exist for the lowest percentage. Through simulations, Lin \textit{et al.} demonstrated that missing SNP imputation improves power over other existing population-based methods dedicated to association identification.\textsuperscript{5} In addition, accurate control of the type I error has been provided. Finally, in the domain of genetic linkage map construction, a simulation study has shown that the presence of missing values produces shorter map lengths for more widely spaced markers.\textsuperscript{7} Thus, the impact of missing data is undoubtedly detrimental to the quality of downstream analyses.

Besides, many softwares devoted to association studies require that the completeness criterion is satisfied for the data sample used to estimate their model parameters. To warrant a sufficiently high confidence in such estimations, among other features, it is crucial that the sample should be large enough.\textsuperscript{8,9} This additional point confirms the need for computational SNP inference methods.

Usually, when authors review the different strategies proposed to infer missing genotype data, not only do they refer to algorithms dedicated to this latter task; they also mention approaches where missing data imputation is \textit{handled simultaneously} with a task of interest such as genotype phasing\textsuperscript{10–12} or disease association study.\textsuperscript{6,13–15} Henceforth, algorithms confining to missing SNP value inference will be referred to as off-line methods. Accuracy results are only available for the latter.

In this paper, our aim is to improve SNP inference accuracy, on the basis of the former work of Roberts and co-authors.\textsuperscript{16} Complementary motivation for investing in research on SNP imputation originates in the current accuracy limitations of (off-line) inference methods. A review dedicated to the description and comparison of some eight off-line methods specifically focuses on \textit{k}-nearest neighbor (KNN) algorithms, as well as various regression methods (the reader is referred to Ref. 17 for more details). Some other works resort to the following strategies or domains: expectation maximization,\textsuperscript{18} Bayesian framework,\textsuperscript{19} Decision Forest pattern recognition,\textsuperscript{20} Gibbs sampling combined with tree-based regression\textsuperscript{5} and neural networks.\textsuperscript{17,21} Besides, the haplotype block structure of eukaryotic genomes has been evidenced.\textsuperscript{22,23} Hence, some algorithms attempt to exploit the corresponding constraints induced on genotypes through local clustering of haplotypes (i.e. phased genotypes) into small groups. In this line, SNP imputation has been implemented using various strategies: entropy measure combined with dynamic programming, to partition haplotypes into blocks\textsuperscript{24}; cluster membership modeled by a hidden Markov model\textsuperscript{12} and KNN imputation within a window sliding along the chromosome.\textsuperscript{16}

Wishing to apprehend the current trends about off-line inference accuracy, we could find no such compilation in the literature. Thus, we would fill this lack by providing the trends presented in Table 1. Indeed, this table underlines that most methods are still likely to confine to accuracies close to 90\% when run on difficult benchmarks, even if some of them are in the meanwhile also able to predict SNPs with an accuracy close to such a high percentage of 97\%. As a conclusion, any
attempt to improve accuracy in missing value estimation seems all the more valuable as the quality of down-stream analyses is the challenging issue.

The algorithm of Roberts and co-workers, NPUTE, processes an SNP array characterized by \( M \), the number of markers (or lines) and \( N \) the number of subjects (or columns), where \( M \) is much higher than \( N \) (up to \( 10^4 \) times higher). The SNP

Table 1. Accuracy percentages for various off-line imputation methods. \( p_{\text{miss}} \): percentage of missing data. SVDR: SeattleSNPs Variation Discovery Resource; AA: African American population; ED: population of European descent; CEPH: Utah residents with ancestry from northern and western Europe; CEU: 60 founders from the Centre d’Etude du Polymorphisme Humain; J/C: 45 Japanese from Tokyo, Japan, and 45 Han Chinese from Beijing, China; YRI: 60 founders from the Yoruba in Ibadan, Nigeria; LD: linkage disequilibrium; GENOA: Genetic Epidemiology Network of Arteriopathy.

<table>
<thead>
<tr>
<th>Reference of study</th>
<th># of SNPs</th>
<th># of subjects</th>
<th>( p_{\text{miss}} ) (%)</th>
<th>Accuracy (%)</th>
<th>Comment</th>
</tr>
</thead>
</table>
| I
|ipi \( ^{24} \) | 103 | 387 | 1 | 94.92 | benchmark \( ^{25} \) across a 500-kb region \( ^{a} \) |
| Haplotype block partitioning based on entropy measure | ** | ** | 10 | 92.02 | |
| | 5200 | 20 | 1 | 92.31 | benchmark \( ^{22} \) |
| | ** | ** | 5 | 90.77 | |
| | ** | ** | 10 | 90.58 | |
| II
|fastPHASE \( ^{12} \) | 216 | 24 | 5 | 94.7 | SVDR, AA |
| Hidden Markov Model | ** | 23 | ** | 97.6 | **, ED |
| | ** | 47 | ** | 96.3 | **, AA + ED |
| | 41018 | 60 | 10 | 96.6 | CEPH HapMap data, chrom. 7 |
| | ** | ** | 25 | 95.9 | across a 159-Mbp region |
| | 15532 | ** | 10 | 96.7 | CEPH HapMap data, chrom. 22 |
| | ** | ** | 25 | 96.1 | across a 35-Mbp region |
| III
|fastPHASE, evaluation in Ref. 17 | 100 | 60 | 5 | 91.8–95.1 \( ^{b} \) | CEPH, SNPs in strong LD \( ^{c} \) |
| | ** | ** | ** | 88.4–93.0 | **, SNPs in weak LD \( ^{d} \) |
| | ** | ** | ** | 61.5–63.5 | **, SNPs in no LD \( ^{e} \) |
| | 100 | 90 | 5 | 92.6–95.6 | J/C \( ^{f} \) |
| | ** | ** | ** | 89.3–94.4 | \( ^{d} \) |
| | ** | ** | ** | 64.9–66.1 | \( ^{f} \) |
| | 100 | 60 | 5 | 87.7–90.8 | \( ^{d} \) |
| | ** | ** | ** | 83.0–86.1 | \( ^{d} \) |
| | ** | ** | ** | 67.9–69.1 | \( ^{d} \) |
| IV
|Linear regression, with backward elimination, evaluation in Ref. 17 | 100 | 60 | 5 | 89.1–93.1 \( ^{b} \) | CEPH \( ^{c} \) |
| | ** | ** | ** | 85.2–90.6 | \( ^{d} \) |
| | ** | ** | ** | 52.3–64.1 | \( ^{d} \) |
| | 100 | 90 | 5 | 88.4–93.8 | J/C \( ^{c} \) |
| | ** | ** | ** | 86.6–92.7 | \( ^{d} \) |
| | ** | ** | ** | 61.2–66.7 | \( ^{d} \) |
| | 100 | 60 | 5 | 83.4–87.1 | YRI \( ^{f} \) |
| | ** | ** | ** | 78.9–83.9 | \( ^{d} \) |
| | ** | ** | ** | 60.4–69.5 | \( ^{d} \) |
Table 1. (Continued)

<table>
<thead>
<tr>
<th>Reference of study</th>
<th># of SNPs</th>
<th># of subjects (%)</th>
<th>$p_{\text{miss}}$ (%)</th>
<th>Accuracy (%)</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>V LARS, Linear regression</td>
<td>100</td>
<td>60</td>
<td>5</td>
<td>89.2–94.2</td>
<td>HapMap data, chrom. 22</td>
</tr>
<tr>
<td>with Least Angle Regression, evaluation in Ref. 17</td>
<td>**</td>
<td>**</td>
<td>**</td>
<td>85.3–91.6</td>
<td>CEU&lt;sup&gt;c&lt;/sup&gt;</td>
</tr>
<tr>
<td></td>
<td>**</td>
<td>**</td>
<td>**</td>
<td>62.7–64.5</td>
<td>**&lt;sup&gt;d&lt;/sup&gt;</td>
</tr>
<tr>
<td></td>
<td>**</td>
<td>**</td>
<td>**</td>
<td>86.8–93.1</td>
<td>**&lt;sup&gt;d&lt;/sup&gt;</td>
</tr>
<tr>
<td></td>
<td>**</td>
<td>**</td>
<td>**</td>
<td>66.2–66.9</td>
<td>**</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>60</td>
<td>5</td>
<td>88.4–94.3</td>
<td>J/CEU&lt;sup&gt;c&lt;/sup&gt;</td>
</tr>
<tr>
<td></td>
<td>**</td>
<td>**</td>
<td>**</td>
<td>79.2–83.6</td>
<td>**&lt;sup&gt;d&lt;/sup&gt;</td>
</tr>
<tr>
<td></td>
<td>**</td>
<td>**</td>
<td>**</td>
<td>68.9–69.9</td>
<td>**</td>
</tr>
<tr>
<td>VI Neural network&lt;sup&gt;21&lt;/sup&gt;</td>
<td>1000</td>
<td>$10^4$</td>
<td>1</td>
<td>95.9</td>
<td>Coalescent model generated by the ms program&lt;sup&gt;26&lt;/sup&gt;</td>
</tr>
<tr>
<td></td>
<td>**</td>
<td>**</td>
<td>5</td>
<td>94.7</td>
<td>recombination and mutation</td>
</tr>
<tr>
<td></td>
<td>**</td>
<td>**</td>
<td>10</td>
<td>94.7</td>
<td>rates both equal to $10^{-8}$ (*)</td>
</tr>
<tr>
<td></td>
<td>680</td>
<td>$10^4$</td>
<td>1</td>
<td>93.1</td>
<td>across a 6-Mbp region</td>
</tr>
<tr>
<td></td>
<td>**</td>
<td>**</td>
<td>5</td>
<td>92.9</td>
<td>$r^2 &lt; 0.8$ selected from (*)</td>
</tr>
<tr>
<td></td>
<td>**</td>
<td>**</td>
<td>10</td>
<td>92.1</td>
<td>**</td>
</tr>
<tr>
<td></td>
<td>552</td>
<td>$10^4$</td>
<td>1</td>
<td>92.5</td>
<td>$r^2 &lt; 0.5$ selected from (*)</td>
</tr>
<tr>
<td></td>
<td>**</td>
<td>**</td>
<td>5</td>
<td>92.7</td>
<td>**</td>
</tr>
<tr>
<td></td>
<td>**</td>
<td>**</td>
<td>10</td>
<td>91.6</td>
<td>**</td>
</tr>
<tr>
<td></td>
<td>288</td>
<td>$10^4$</td>
<td>1</td>
<td>93.2</td>
<td>$r^2 &lt; 0.2$ selected from (*)</td>
</tr>
<tr>
<td></td>
<td>**</td>
<td>**</td>
<td>5</td>
<td>92.3</td>
<td>**</td>
</tr>
<tr>
<td></td>
<td>**</td>
<td>**</td>
<td>10</td>
<td>92.0</td>
<td>**</td>
</tr>
<tr>
<td></td>
<td>1962</td>
<td>90</td>
<td>1</td>
<td>96.2</td>
<td>CEPH HapMap chrom. 22</td>
</tr>
<tr>
<td></td>
<td>**</td>
<td>**</td>
<td>5</td>
<td>95.4</td>
<td>**</td>
</tr>
<tr>
<td></td>
<td>**</td>
<td>**</td>
<td>10</td>
<td>95.1</td>
<td>**</td>
</tr>
<tr>
<td></td>
<td>126</td>
<td>1458</td>
<td>1</td>
<td>86.8</td>
<td>Chrom. 2, GENOA&lt;sup&gt;27&lt;/sup&gt;</td>
</tr>
<tr>
<td></td>
<td>**</td>
<td>**</td>
<td>2</td>
<td>86.5</td>
<td>Data from Ref. 30</td>
</tr>
<tr>
<td></td>
<td>**</td>
<td>**</td>
<td>5</td>
<td>83.1</td>
<td>**</td>
</tr>
<tr>
<td>VII Neuter&lt;sup&gt;16&lt;/sup&gt;</td>
<td>1024</td>
<td>46</td>
<td>5</td>
<td>$\sim 97^d$</td>
<td>150 k benchmark</td>
</tr>
<tr>
<td>Nearest neighbor method combined with window sliding</td>
<td>**</td>
<td>**</td>
<td>10</td>
<td>$f$</td>
<td>Combined SNPs from the</td>
</tr>
<tr>
<td></td>
<td>**</td>
<td>**</td>
<td>15</td>
<td>$f$</td>
<td>140 k Broad/MIT mouse&lt;sup&gt;28&lt;/sup&gt;</td>
</tr>
<tr>
<td></td>
<td>**</td>
<td>**</td>
<td>20</td>
<td>$f$</td>
<td>and the 10 k GNF</td>
</tr>
<tr>
<td></td>
<td>**</td>
<td>**</td>
<td>25</td>
<td>$\sim 96^d$</td>
<td>mouse dataset</td>
</tr>
<tr>
<td></td>
<td>1024</td>
<td>16</td>
<td>5</td>
<td>94.1</td>
<td>Perlegen mouse dataset&lt;sup&gt;29&lt;/sup&gt;</td>
</tr>
<tr>
<td></td>
<td>**</td>
<td>**</td>
<td>10</td>
<td>94.2</td>
<td>(<a href="http://mouse.perlegen.com">http://mouse.perlegen.com</a>)</td>
</tr>
<tr>
<td></td>
<td>**</td>
<td>**</td>
<td>15</td>
<td>93.5</td>
<td>Consecutive SNPs extracted</td>
</tr>
<tr>
<td></td>
<td>**</td>
<td>**</td>
<td>20</td>
<td>93.4</td>
<td>from a high-resolution set of</td>
</tr>
<tr>
<td></td>
<td>**</td>
<td>**</td>
<td>25</td>
<td>92.8</td>
<td>8.3 million SNPs</td>
</tr>
</tbody>
</table>

<sup>a</sup>Density of markers; <sup>b</sup>Accuracy range over various tuning parameter values; <sup>c</sup>The 100 top-ranked SNPs showing minor allele frequencies above 5% and $p$-values for the Hardy–Weinberg equilibrium test greater than 0.01 were considered as SNPs in strong LD; <sup>d</sup>Weak LD, $r^2 < 0.1$; <sup>e</sup>No LD, $r^2 < 10^{-4}$; <sup>f</sup>Accuracy percentages are reported from a low-resolution plot; the total accuracy decrease between 5% and 25% missing data percentages is estimated to be around 1%. 

Iterative Two-Pass Algorithm for Missing Data Imputation in SNP Arrays 837
array is scanned from top to bottom, through a sliding window of fixed width. Imputation is achieved for all the missing calls located in the central row of the current window. Thoroughly examining Npute algorithm, we identified possible optimizations and therefore designed KnnWinOpti, a variant of Npute. Central to our contribution is the sensitivity of SNP inference to the scanning direction of the SNP array. Henceforward, in our approach, two scans performed in opposite directions (that is from top to bottom and from bottom to top) will provide two sets of inferred values for the same missing calls; the confrontation of these two sets will definitely fix SNPs inferred with identical values and delay the resolution of other SNPs. In counterpart, SNPs whose inference has been delayed will further benefit from an inference context enriched with previously fixed SNP values. The design of a high-level algorithm, SNPShuttle, is straightforward: inferring missing calls with reinforced confidence is implemented by iterating bi-directional scans until no more SNP value can be fixed. Section 2 presents the basic concepts common to algorithms Npute, KnnWinOpti and SNPShuttle. Section 3 describes the schemes of algorithms KnnWinOpti and SNPShuttle. Section 4 focuses on the comparison of KnnWinOpti and SNPShuttle, relying on realistic benchmarks.

2. Preliminaries

The original algorithm, Npute, and our variant KnnWinOpti, both implement nearest neighbor SNP inference within sliding windows. The input parameters for both algorithms are the SNP panel, a matrix $S[0 \cdots M-1][0 \cdots N-1]$ of $M$ markers and $N$ subjects, and $L$, the “half-width” of the sliding window. The elements of the SNP matrix belong to $\{0, 1, 2, ?\}$. For each genetic marker, the code 0 denotes homozygoty for the variant with major frequency, 1 indicates homozygoty for the least frequent variant, 2 specifies heterozygoty and “?” is the label for “missing data”.

**Definition 1.** Given two subjects $s$ and $s'$ respectively characterized with variants $v$ and $v'$ of a given marker $m$ ($v = S[m][s]$, $v' = S[m][s']$), we define $\delta$, the distance between the two variants $v$ and $v'$, such that

$$\delta(v, v') = \begin{cases} 
0 & \text{if } (v \neq ? \text{ and } v' \neq ? \text{ and } v = v') \\
1 & \text{if } (v = ? \text{ or } v' = ?) \\
2 & \text{if } (v \neq ? \text{ and } v' \neq ? \text{ and } v \neq v') 
\end{cases}$$

We conform to Roberts and co-workers’ choice of the elementary distance between two variants of a given genetic marker. $\delta$ is an adaptation of the Hamming distance $\delta_H$, for which we recall that $\delta_H(v, v') = 0$ if $v = v'$ and $\delta_H(v, v') = 1$ in the case of a mismatch. To implement a nearest neighbor strategy, it could be envisaged to restrain the comparison of two subjects, within a given window, to the SNPs for which no missing call is observed in neither subject. Then, $\delta$ would coincide with $\delta_H$. But simply ignoring missing calls in the comparison of two subjects would amount to systematically considering these missing calls equal, which would
entail a bias. Therefore, a more refined comparison is required. If at least one of \( v \) and \( v' \) is a missing call, \( \delta_H(v, v') \) might be equal to 0 or 1. Thus, \( \delta_H(\ (?, v), \delta_H(v, \ ?) \) and \( \delta_H(\ ?, \ ?) \) are approximated as \( \frac{1}{2} \). Actually, for convenience, values \( \frac{1}{2} \) and 1 are scaled by a factor of 2.

**Definition 3 (Inference context).** In the general case of a symmetric win-
row vector of distances, where subjects are pairwise-compared within a window centered on row \( m \) a vector of pairwise comparisons between subjects, not between loci, as is the case me-
gy. If need be, that is when there is a missing call in a row \( N \) due to linkage disequilibrium, is indeed the very point exploited by algorithm for conventional linkage metrics. Nevertheless, the existence of haplotype blocks, \( S \) for which \( C \) entail a bias. Therefore, a more refined comparison is required. If at least one of \( v \) and \( v' \) is a missing call, \( \delta_H(v, v') \) might be equal to 0 or 1. Thus, \( \delta_H(\ (?, v), \delta_H(v, \ ?) \) and \( \delta_H(\ ?, \ ?) \) are approximated as \( \frac{1}{2} \). Actually, for convenience, values \( \frac{1}{2} \) and 1 are scaled by a factor of 2.

**Definition 2 (PMD vector).** Given a marker \( m \) and the \( N \) subjects considered, the elementary pairwise mismatch distance vector \( PMD(m) \) is defined as a vector of \( N(N - 1)/2 \) entries such that \( PMD(m)[(s, s')] = \delta(S[m][s], S[m][s']) \), \( 0 \leq s < s' < N \).

We now link the notion of elementary \( PMD \) vector to the inference context of row \( m \), itself a \( PMD \) vector. \( PMD \) is a vector of distances, where subjects are pairwise-compared for row \( m \) alone. In contrast, the inference context of row \( m \) is a vector of distances, where subjects are pairwise-compared within a window centered on row \( m \) (in the general case).

**Definition 3 (Inference context).** In the general case of a symmetric win-
row centered on row \( m \) \( (L \leq m \leq M - L - 1) \), \( Context(m) \) is a vector of \( N(N - 1)/2 \) entries verifying \( Context(m)[(s, s')] = \sum_{i=m-L,i\neq m} PMD(i)[(s, s')] \), \( 0 \leq s < s' < N \).

Thus, each element of the \( PMD \) vector \( Context(m)[(s, s')] \) represents the total “number of mismatches” (according to distance \( \delta \)), between subjects \( s \) and \( s' \), within a window spanning from row \( m - L \) to row \( m + L \).

**Remark 1.** Row \( m \) is excluded from the calculation of its proper context \( Context(m) \).

The concept of context is to be related to that of haplotype block. \( Context(m) \) is a vector of pairwise comparisons between subjects, not between loci, as is the case for conventional linkage metrics. Nevertheless, the existence of haplotype blocks, due to linkage disequilibrium, is indeed the very point exploited by algorithm NPUTE, in the line of all imputation algorithms relying on a nearest neighbor strategy. If need be, that is when there is a missing call in a row \( m \), for a subject \( s \), it is then possible to identify a nearest neighbor for \( s \), within the concerned context. Given the \( PMD \) vector \( Context(m) \), inference for any missing \( S[m][s] \) is straightforward: the list of the \( N - 1 \) relevant entries \((s, s') (s < s') \) and \((s', s) (s' < s) \) in \( Context(m) \) is sorted in increasing order. That is, the subjects showing the small-
est distances with subject \( s \) appear at the top of this list. The nearest neighbor \( s_n \) for which \( S[m][s_n] \) is not missing itself is identified in the list. Then, \( S[m][s_n] \) is assigned to \( S[m][s] \).

The key idea of NPUTE is performing fast imputation over overlapping sliding windows. The scan of the SNP panel is implemented by shifting the current window one range further at each step. Thus, in the symmetric case, the \( PMD \) vector \( Context(m) \) relative to a window centered on row \( m \) is merely the \( PMD \) relative to the previous overlapping window centered on row \( m - 1 \), from which the contribution
Fig. 1. Window sliding along SNP array and updating of $C_{\text{ontext}}$. $M = 15$; $L = 3$ (“half-width” of symmetric window) (see Sec. 2 “Preliminaries” for definitions of $C_{\text{ontext}}$ and $PMD$). In each figure (a), (b) or (c), the left and the right subfigures respectively display $C_{\text{ontext}}(m-1)$ and $C_{\text{ontext}}(m)$. The annotations on the right subfigure highlight which operations are performed to update $C_{\text{ontext}}(m-1)$ into $C_{\text{ontext}}(m)$. (a) Top section of SNP array including first symmetric window: through window sliding, window upper section widens until the window becomes symmetric; (b) Median section including last symmetric window: a perfectly symmetric window of constant width moves down median section; (c) Bottom section: window lower section shrinks as window moves downwards.

of row $m - L - 1$ must be subtracted and to which that of row $m + L$ must be added [see Fig. 1(b)]. However, in the simple case of a symmetric window, $C_{\text{ontext}}(m)$ is not merely computed as $C_{\text{ontext}}(m-1) - PMD(m - L - 1) + PMD(m + L)$. Row $m$ itself does not contribute to the calculation of $C_{\text{ontext}}(m)$, which is indeed computed as follows: $C_{\text{ontext}}(m) = C_{\text{ontext}}(m-1) + PMD(m - 1) - PMD(m) - PMD(m - L - 1) + PMD(m + L)$.

In addition, note that there are $2 \times L + 1$ windows to be processed apart, among which $2 \times L$ are not symmetric. Figure 2 depicts the variation in shape of the sliding window as it moves down the marker array.

Remark 2 (Non-symmetric windows). We arbitrarily call the “top” section of the marker array the section with lowest marker indexes. Besides, given the current

Fig. 2. Sliding window. $M = 15$; $L = 3$ (“half-width” of symmetric window). As the sliding window moves downwards the SNP array, depending on the row $m$ under inference, the window is either centered on $m$ or is non-symmetric.
location of the sliding window, we refer to its upper and lower sections respectively as the sections with the lowest and highest indexes.

For all rows \( m \) in the top section of array \( S \) \((0 \leq m \leq L - 1)\), the lower sections of the corresponding windows have constant width \( L \). In contrast, the upper section width increases from 0 to \( L - 1 \) as the window slides downwards.

The second case of non-symmetry arises for any row in the “bottom” section of the marker array \((M - L \leq m \leq M - 1)\). There, the upper section maintains a constant width \( L \), whereas the lower section width decreases from \( L - 1 \) to 0.

**Property 1 (Recurrence relation).** (see Fig. 1)
We recall that row \( m \) is excluded from the calculation of its proper context \( C_{\text{context}}(m) \), whatever the conformation of the current window (symmetric or not). Therefore, context updating will always include contribution \( \text{PMD}(m - 1) - \text{PMD}(m) \).

**Initialization:**
\[
C_{\text{context}}(0) = \sum_{i=1}^{L} \text{PMD}(i)
\]

(A) \( 1 \leq m \leq L \):
\[
C_{\text{context}}(m) = C_{\text{context}}(m - 1) + \text{PMD}(m - 1) - \text{PMD}(m) + \text{PMD}(m + L)
\]

(B) \( L + 1 \leq m \leq M - L - 1 \):
\[
C_{\text{context}}(m) = C_{\text{context}}(m - 1) + \text{PMD}(m - 1) - \text{PMD}(m) - \text{PMD}(m - L - 1) + \text{PMD}(m + L)
\]

(C) \( M - L \leq m \leq M - 1 \):
\[
C_{\text{context}}(m) = C_{\text{context}}(m - 1) + \text{PMD}(m - 1) - \text{PMD}(m) - \text{PMD}(m - L - 1)
\]

However, due to backward dependencies [\( \text{PMD}(m - L - 1) \)] and forward dependencies [\( \text{PMD}(m + L) \)], Npute computation of successive contexts is not fully optimized. Indeed, in Property 1 [lines (B) and (C)], \( \text{PMD}(m - L - 1) \) has already been inferred. Besides, \( \text{PMD}(m + L) \) [lines (A) and (B)] will be computed again as contribution \( \text{PMD}(m') \) in \( C_{\text{context}}(m') \) when \( m' \) is equal to \( m + L \).

3. Towards Iterative Bi-Directional Imputation

3.1. Management of backward and forward dependencies combined with SNP block loading

Property 1 advocates the need for maintaining two backward and forward FIFO lists, denoted by \( B \) and \( F \). Besides, it is attractive to design software independent of memory-availability requirements. To meet this second purpose, during an SNP array scan, we choose to successively load contiguous “small” SNP blocks. Thus, the memory needs only be allocated for a SNP block rather than for the whole SNP array. It now remains to combine such dependency management with the loading of successive SNP blocks of \( K \) rows (\( K \) is an input parameter tuned to a default
The sketch of this novel version, KNNWinOpti, is described as Algorithm 1, where we annotated lines and blocks of lines.

Since the combination of the two modifications (dependency management, SNP block loading) brings complexity in the description of the novel version, we will carefully comment it below. Points (A) to (C) that establish the recurrence relation (Property 1) are respectively described as Loop I (line 4), Meta-Loop II (lines 6 to 9) and Loop III (line 11). In the original NPUTE version and in the general case (B) of the recurrence relation, three PMD vectors need to be computed for row m inference: $PMD(m)$, $PMD(m + L)$ and $PMD(m - L - 1)$. Instead, when our variant KNNWinOpti processes row m, $PMD(m)$ is expected as the head of $\mathcal{F}$ (block 24, line 1). $PMD(m + L)$ is computed as a contribution to $\mathcal{C}_{ontext}(m)$ and is added to $\mathcal{F}$ (block 24, end of line 2), for further reuse as $PMD(m')$ at the time of inference of row $m'$ ($m' = m + L$). $PMD(m - L - 1)$ is expected as the head of $\mathcal{B}$ (line 33) and contributes to the calculus of $\mathcal{C}_{ontext}(m)$ (line 34). After row $m$ inference (line 35, referring to block 21), the now updated vector $PMD(m)$ is added to $\mathcal{B}$ (block 21, line 2). Thus, this latter PMD vector may be reused as $PMD(m' - L - 1)$, opportunely as the head of $\mathcal{B}$ at the time of inference for row $m'$ ($m' = m + L + 1$) (line 33). Figure 3 shows using our running example ($M = 15; L = 3$) how the two FIFO lists are synchronized.

Finally, SNP block loading and SNP inference must also be synchronized. The preliminary loading of a block of $2L + 1$ rows (line 13) is required by inference of row 0 and rows involved in Loop I; the SNP block loading manager (line 30) ensures that row $i + L$ is available at the time of row $i$ inference (Meta-Loop II). Depending on the total number of rows imputed in Meta-Loop II, the last “meta”-iteration in

**Algorithm 1 - KNNWinOpti**($M$, $N$, $S$, $L$, $K$, $nbLoadings$)

**INPUT:**
- $M$, the number of genetic markers;
- $N$, the number of subjects;
- $S$, an SNP array of size $M \times N$, with values belonging to \{0, 1, 2, ‘?’\}, where each line corresponds to a genetic marker and each column corresponds to a subject;
- $L$, half-width of the sliding window;
- $K$, the size of the SNP blocks successively loaded (in number of rows);
- $nbLoadings$, the number of loadings of SNP blocks necessary to run Meta-Loop II.

**OUTPUT:** the array $S$, where each previous value ‘?’ has been replaced with 0, 1 or 2.

**AUXILIARY VARIABLES:**
- $\mathcal{B}$ and $\mathcal{F}$, two FIFO lists dedicated to the memorization of backward and forward dependencies.
- In the following, suffixes appended to the variable name $PMD$ are meant for a quick understanding;
- $PMD_{m-}$, pairwise mismatch distance vector for current row, say $m$, before SNP inference;
- $PMD_{m+}$, PMD vector for current row, after SNP inference;
- $PMD_{m-L-1}$, PMD vector for row $m - L - 1$ (inferred);
- $PMD_{m+L}$, PMD vector for row $m + L$ (not inferred yet);
- $\mathcal{C}_{ontext}$: PMD vector resulting from the summation, column by column, of the PMD vectors related to the rows of current window.
Algorithm 1 - KNNWinOpti($M, N, S, L, K, nbLoadings$) (continued)

1. $\mathcal{F} \leftarrow \emptyset$; $\mathcal{B} \leftarrow \emptyset$ /* initialization of the two FIFO lists */
2. inferenceOfRow0($\emptyset$)

3. /* Loop I: processing of all rows (but 0) located at top section of SNP array $S$ */
4. inferenceOfRows1ToL($\emptyset$)

5. /* Meta-Loop II: processing of rows whose windows are symmetric (but row $L$) */
6. start $\leftarrow L + 1$
7. for $i = 1$ to $nbLoadings$
8. inferenceOfKRows($\emptyset$)
9. endfor
10. /* Loop III: processing of rows located at the bottom section of array $S$ */
11. inferenceOfRowsM-LToM-1($\emptyset$)

12. procedure inferenceOfRow0()
13. $S \leftarrow$ loadRows($2L + 1$)
14. /* computation of the first context (row 1 to $L$) */
15. $\mathcal{C} \leftarrow 0$
16. for $i = 1$ to $L$
17. $PM^m \leftarrow$ computePMD($\emptyset$); push($\mathcal{F}, PM^m$); $\mathcal{C} \leftarrow \mathcal{C} + PM^m$
18. endfor
19. /* inference for row 0 */
20. $m \leftarrow 0$
21. (A) $\begin{cases} PM^m \leftarrow \text{inferMissingSNPSsForRow}(\mathcal{C}, m) \\ PM^{m-1} \leftarrow PM^m \end{cases}$

22. procedure inferenceOfRows1ToL() /* Loop I */
23. for $m = 1$ to $L$
24. (B) $\begin{cases} PM^m \leftarrow \text{remove}(\mathcal{F}) \\ PM^{m+L} \leftarrow \text{computePMD}(m + L); \text{push}(\mathcal{F}, PM^{m+L}) \\ \mathcal{C} \leftarrow \mathcal{C} + PM^{m-1} = PM^m + PM^{m+L} \end{cases}$
25. $\mathcal{C} \leftarrow \mathcal{C} + PM^{m-1} = PM^m + PM^{m+L}$
26. process as in (A)
27. endfor

28. procedure inferenceOfKRows() /* Meta-Loop II */
29. /* For conciseness, we suppose here that the last iteration in Meta-Loop II requires exactly the loading of $K$ rows. The adaptation required when $M - 2L - 1 \mod K$ differs from 0 is trivial and is not shown here. */
30. $S \leftarrow$ loadRows($K$)
31. for $m = \text{start}$ to $\text{start} + K - 1$
32. process as in (B)
33. $PM^{m-L-1} \leftarrow \text{remove}(\emptyset)$
34. $\mathcal{C} \leftarrow \mathcal{C} + PM^{m-1} = PM^m - PM^{m-L-1} + PM^{m+L}$
35. process as in (A)
36. endfor
37. start $\leftarrow start + K$

38. procedure inferenceOfRowsM-LToM-1() /* Loop III */
39. for $m = M - L$ to $M - 1$
40. $PM^m \leftarrow \text{remove}(\mathcal{F})$; $PM^{m-L-1} \leftarrow \text{remove}(\emptyset)$
41. $\mathcal{C} \leftarrow \mathcal{C} + PM^{m-1} = PM^m - PM^{m-L-1}$
42. $PM^{m} \leftarrow \text{inferMissingSNPSsForRow}(\mathcal{C}, m)$
43. $PM^{m-1} \leftarrow PM^m$
this loop may require the loading of less than $K$ rows, which explains a specific (trivial) treatment not shown here. Notice that Loop III does not refer to forward dependencies.

3.2. Iterative two-pass imputation

In their method, before performing inference, Roberts and co-authors compute all \textit{PMD} vectors. Instead, we are careful that any newly imputed row is added to $\mathcal{B}$ list (second line of block 21). Therefore, in the cases where distance $\delta(v,v')$ was approximated as 1 if $v = v'$, a recent inference for $v$ or $v'$ value would possibly switch $\delta(v,v')$ to the accurate value of 0 or 2. Thus, not only is \textsc{KnnWinOpti} a variant of \textsc{Npute} augmented with FIFO list management and block loading, it also allows further optimization. Since algorithm \textsc{Npute} does not update $\mathcal{Context}(m)$ according to the results of previous inferences for rows $m - L$ to $m - 1$, scanning the SNP array from top to bottom (TB) and scanning it from bottom to top (BT) provide the same inferred SNP values. In contrast, in the case when $\mathcal{Context}(m)$ is computed on the fly, the result of a TB pass may be compared with that of a BT pass, in order to resolve the missing data with more confidence.

\textsc{SnpShuttle} (see Algorithm 2) iterates the following process: procedures scanFromToBottom (line 4) and scanFromBottomToTop (line 5) respectively implement independent scans of current marker array $S$, in opposite directions. \textsc{KnnWinOpti} serves this double purpose: scanFromBottomToTop is easily implemented by running \textsc{KnnWinOpti} on $S$ inverted row per row. The sets of newly inferred SNPs are compared (line 6). The SNP variants inferred with identical values by these scans are fixed and array $S$ is updated accordingly (lines 7 to 10).
This process is repeated until all remaining missing calls cannot be resolved, at which point the algorithm switches to KNNWinOpti process.

4. Experimental Results and Discussion

Our objective in this paper is to focus on checking whether iterative bi-directional scanning increases imputation accuracy with respect to mono-directional scanning. We just mention here that as far as accuracy is concerned, NPUTE’s authors have shown that this algorithm performs similarly as fastPHASE, a software reputable for its accuracy. Moreover, due to the quasi-linearity with respect to $M$, it was also shown that NPUTE is about ten times faster than fastPHASE.

4.1. Simulation of realistic benchmarks

In our case, evaluation may only be conducted on simulated data. To control $p_{miss}$, the percentage of missing genotypes, we generated benchmarks from genuine mouse genotype files made available online by the Perlegen project. The Perlegen 8.27-million SNP set provides a high resolution map for 15 strains of inbred laboratory mice. Given the number $M$ of markers specified for the benchmark, we scanned the file relative to each chromosome with a sliding window of size $M$, shifting it by one row at each step. Thus, we identified the region of $M$ contiguous markers showing the lowest percentage of missing data, $p_0$. We extracted this region and replaced at random SNP variants with missing calls, in the proportion specified by $p_{miss}$. We checked that the simulated missing calls were not already missing in the original file. We observed that the average missing data percentage calculated over a whole chromosome ranges between 10.81% (chromosome 17) and 15.32% (chromosome 11). We chose $M$ equal to 2500. Percentage $p_0$ varies in interval [2.69%, 7.54%] over the chromosomes analysed (minimum and maximum respectively obtained for chromosomes 14 and 11). Then, we generated twenty benchmarks of 2500 markers.

---

Algorithm 2 - SNPShuttle($M, N, S, L, K, nbLoadings$)

```plaintext
INPUT and OUTPUT: same parameters as for KNNWinOpti

1. modified ← true;
2. while (modified)
3.     modified ← false
4.     $SNP_1, inferred$ ← scanFromTopToBottom(S)
5.     $SNP_2, inferred$ ← scanFromBottomToTop(S)
6.     solvedSNPs ← compare($SNP_1, inferred, SNP_2, inferred$)
7.     if (solvedSNPs ≠ $\emptyset$)
8.         updateMarkerArray($S, solvedSNPs$)
9.         modified ← true
10.    endif
11. endwhile
12. if there still exist ‘?’s in $S$
13.    call KNNWinOpti($M, N, S, L, K, nbLoadings$)
14. endif
```
for each of the 20 mouse chromosomes and each $p_{\text{miss}}$ percentage value in the range [0\%-30\%] (step 5\%).

4.2. **SNP variant assignment**

This procedure has been described in Sec. 2. When there exist several nearest neighbors with SNPs $m$ available, assignment is implemented by adopting a majority-voting scheme.

4.3. **Window size tuning**

In NPUTE’s implementation, window tuning requires much more time than inference itself, which scales in $O(MN^2)$. NPUTE achieves the tuning of parameter $L$ through the comparison of inference accuracies computed for the $n_L$ candidate values of $L$ in interval [5, 30] ($n_L = 26$). The $L$ value retained is that corresponding to the highest accuracy. Since observed and predicted values must be compared to calculate accuracies, the following trick is implemented: each SNP which is not a missing call is temporarily considered missing and is inferred. On the subject of time, the ratio of tuning complexity to inference complexity is $n_L \times (100 - p_{\text{miss}}) / p_{\text{miss}}$.

In this paper, where we are only interested in investigating whether our intuition regarding bi-directional scanning is sound, we tested our algorithms on the same genetic data used for NPUTE’s evaluation. Thus we could shortcircuit window size tuning. We reused the $L$ value computed by Roberts et al., for each chromosome. These values range in the interval [14, 20]. Our decision is justified by observing that the accuracy is quasi-constant when $L$ ranges in [14, 20] (at least) (see Fig. 7 from Ref. 16).

4.4. **Accuracy comparison**

The accuracies of KNNWinOpti and SNPShuttle are compared in Fig. 4.

![Fig. 4. Comparison of KNNWinOpti and SNPShuttle’s accuracies for various missing data percentages. $M = 2500$ markers. KWO and SSH respectively refer to KNNWinOpti and SNPShuttle algorithms.](image)
Not surprisingly, for each algorithm, we observe that the inference accuracy decreases when the missing percentage increases. Above all, our results show that SnpShuttle’s accuracies are always higher than KnnWinOpti’s. Table 2 recapitulates the averages, standard deviations, minima and maxima observed over all 20 chromosomes, for the two algorithms.

In Table 2, we also provide the results obtained with the reference algorithm Npute. The trend observed consists in a slight improvement of KnnWinOpti’s accuracy over that of Npute. The difference ranges in the interval [0.20, 0.60]. This narrow difference between the two algorithms’ performances was foreseeable. KnnWinOpti guarantees that when row \( m \) is being inferred, all missing calls in the upper section of the sliding window have been assigned a value. Suppose for instance that \( p \) SNPs were unknown in the upper section of the sliding window for at least one of two subjects \( s \) and \( s' \). Then, on this window upper section, the distance between \( s \) and \( s' \), which would be \( \alpha + p \times 1 \) when approximated by Npute, takes its accurate value inside \([\alpha + p \times 0, \alpha + p \times 2]\). Thus, on average, KnnWinOpti’s expected distance should tend towards Npute’s distance. However, in a nearest-neighbor strategy, the KnnWinOpti distance between \( s \) and its nearest neighbor \( s_n \) is more likely to take its accurate value in the lower section of interval \([\alpha + p \times 0, \alpha + p \times 2]\) than in the upper section. The slight discrepancy observed in favor of KnnWinOpti is an indirect consequence of this bias. This remark confirms that the very crucial point to gain marginal accuracy is actually the iterative bi-directional scan of the SNP array. To implement SnpShuttle, rewriting Npute into KnnWinOpti was a necessity.

The accuracy gain brought by SnpShuttle may seem moderate. Nonetheless, it must be highlighted that KnnWinOpti already provided quite high accuracies, especially in comparison with the trends compiled in Table 1. Fairly poor gains are observed for \( p_{\text{miss}} \) values exceeding 20% (1.21% on average, for this latter percentage). We would emphasize that our investigation relative to missing percentages above 20% (included) is mainly justified by study completeness. Indeed, the tendency for high-throughput genotyping techniques is to provide data characterized with missing data percentages in the range [5%, 10%]. Thus, an interesting improvement is observable under realistic conditions (1.81% and 1.61% on average for 5% and 10% respectively). For illustration, a focus on the 5% case shows that the number of chromosomes characterized with accuracies over 94% increases from 5 (with KnnWinOpti) to 18 (with SnpShuttle). This number increases from 0 to 12 for accuracies over 95%. Finally, SnpShuttle even enables the increase of four chromosomes’ accuracies over 96%.

4.5. Discussion

A gain in accuracy of SnpShuttle over KnnWinOpti has been evidenced in our study. However, the significance of the admittedly modest gain observed is questionable: for already high gains, would an improvement of about 2% be enough to
Table 2. Comparison of the performances of KnnWinOpti and SnpShuttle.

<table>
<thead>
<tr>
<th></th>
<th>Accuracy (%)</th>
<th>Missing data percentage (p&lt;sub&gt;miss&lt;/sub&gt;)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>5%</td>
<td>10%</td>
</tr>
<tr>
<td>Npute</td>
<td>average (A&lt;sub&gt;NPU&lt;/sub&gt;)</td>
<td>92.62 ± 0.98</td>
</tr>
<tr>
<td></td>
<td>min</td>
<td>90.68</td>
</tr>
<tr>
<td></td>
<td>max</td>
<td>94.43</td>
</tr>
<tr>
<td>KnnWinOpti</td>
<td>average (A&lt;sub&gt;KWO&lt;/sub&gt;)</td>
<td>93.22 ± 1.03</td>
</tr>
<tr>
<td></td>
<td>min</td>
<td>90.04</td>
</tr>
<tr>
<td></td>
<td>max</td>
<td>94.56</td>
</tr>
<tr>
<td>SnpShuttle</td>
<td>average (A&lt;sub&gt;SSH&lt;/sub&gt;)</td>
<td>95.03 ± 1.19</td>
</tr>
<tr>
<td></td>
<td>min</td>
<td>91.49</td>
</tr>
<tr>
<td></td>
<td>max</td>
<td>96.26</td>
</tr>
<tr>
<td>A&lt;sub&gt;KWO&lt;/sub&gt; - A&lt;sub&gt;NPU&lt;/sub&gt;</td>
<td>0.60</td>
<td>0.49</td>
</tr>
<tr>
<td>A&lt;sub&gt;SSH&lt;/sub&gt; - A&lt;sub&gt;KWO&lt;/sub&gt;</td>
<td>1.81</td>
<td>1.61</td>
</tr>
</tbody>
</table>

Note: For each missing SNP percentage p<sub>miss</sub>, the average, standard deviation, minimum and maximum are calculated from the 20 chromosome mean accuracies. Each chromosome mean accuracy was computed from 20 benchmarks of 2500 markers characterized by a common p<sub>miss</sub> value (with missing calls distributed at random). We also provide the results observed with the reference algorithm Npute.
reveal an association otherwise not predicted or predicted borderline? In an afore-
cited analysis of the impact of missing data on association prediction, an effect has
been shown, even for the low percentage of 5%. A fine evaluation of such an impact
would require intensive simulations, including weak to strong interactions, which
lies beyond the scope of our paper. Yet, even in the ideal case where few data are
missing, the difficulty to exhibit associations is such that nowadays standards for
publishing SNP-disease associations compel authors to confirm their results on two
different benchmarks, in order to discard false positive results. In addition, endeavor-
ning for a supplementary accuracy gain of 2% seems all the more worthwhile, as
lowering the risk of false-negative results is crucial for public health.

When inferring real missing calls, which corresponds to setting \( p_{\text{miss}} \) to 0, accur-
acies cannot be computed. Roberts et al. defined an indicator, the confidence score,
as follows: for each \( S[m][s] \) imputed, they calculate the percentage of matches in
\( \text{Context}(m) \) (before imputation) between subject \( s \) and the nearest neighbor \( s_n \)
used for this inference. The confidence score is computed as the average of such
percentages over all missing calls. Consistent with Roberts and co-workers’ results
(Fig. 8 from Ref. 16), we check that this score is highly correlated with the accuracy:
for the Perlegen mouse data and \( p_{\text{miss}} \) values in the range [5%, 15%], correlation
coefficients are always higher than 0.86.

\( \text{NPUTE} \)'s memory and time complexities both scale in \( O(MN^2) \). With block load-
ing, \( \text{KNNWinOpti} \)'s requirements in memory allocation decrease down to \( O(KN^2) \),
where \( K \) is the number of rows per block. As a counterpart, the temporal com-
plexity is slightly augmented [by \( O(M/K) \) with respect to that of \( \text{NPUTE} \). Relying
on the missing calls that are easiest to guess, in order to infer the other ones with
more confidence, is an attractive idea. In theory, the most accurate results should
be obtained through the following algorithm: (i) score each row of the SNP panel
with the percentage of missing calls in the window centered on the row; (ii) sort
the rows in increasing score order and obtain list \( L \); (iii) remove the lowest-scored
row \( m \) from \( L \), infer missing calls for row \( m \) and update accordingly row \( m \), sub-
sequently update “knowledge” for the \( 2 \times L \) windows overlapping row \( m \), then
update sorted list \( L \); iterate (iii) until some termination condition is verified as in
\( \text{SNPShuttle} \). This method is untractable in the general case since its time complexity
scales in \( O(M (M \log(M) + N^2)) \). \( \text{SNPShuttle} \) may be viewed as but a very rough
approximation of the previous untractable method.

5. Conclusion

In this paper, we investigated the accuracy-improvement potentiality of the algo-


alarm designed by Roberts and co-workers. We showed that iterative two-pass scan-
ing of SNP arrays constantly brings an accuracy gain over \( \text{KNNWinOpti} \), a variant
of the yet performing algorithm \( \text{NPUTE} \). Bi-directional parsing is meant to delay
the prediction of the SNPs most difficult to resolve, until their environment is
enriched enough with previously inferred SNPs. Up to the missing data percentage
of 15%, we observed that the accuracy increase ranges between 1.56 and 1.81%. Further improvement of imputation quality is worthwhile with regard to challenging down-stream analyses such as identifying disease risk or reducing the trial-and-error approach in the choice of treatments through personalized medicine. In this exploratory study, we implemented a very rough approximation of the untractable ideal algorithm; as a compromise, in the future, we plan to design and test strategies explicitly conditioning inference relative to sub-regions on the quality of their flanking regions. Finally, it is not a trivial problem to examine how the automatic tuning of the optimal window width can be boosted.

Acknowledgments

The author wishes to thank F. Fani for his contribution to software implementation as a trainee of the Conservatoire National des Arts et Métiers, in Nantes. The author is also grateful to an anonymous referee for substantive comments that have significantly improved this manuscript.

References

Christine Sinoquet is an Associate Professor of Computer Science at the University of Nantes, France, where she works in the area of bioinformatics and computational biology at the Computer Science Institute of Nantes-Atlantic. She holds a M.Sc. in Computer Science from the University of Rennes 1 and received her Ph.D. in Computer Science from this same institution. During her Ph.D. position at the Inria Centre of Rennes, she specialized in bioinformatics. She has initiated two Master degree programs in Bioinformatics (University of Clermont, France, and Nantes). She currently serves as the Head of this second Master degree program since 2005. Her research activities have been focused on various topics including data correction prior to molecular phylogeny inference, motif discovery in biological sequences, comparative genomics and missing genotype imputation. At the present time, her research interests are algorithmic and statistical aspects of complex data analysis, with a major application to genome-wide dissection of the genetic susceptibility of complex diseases.