NGS based haplotype assembly using matrix completion

Sina Majidian [ORCID]**, Mohammad Hossein Kahaei** [ORCID]*

School of Electrical Engineering, Iran University of Science & Technology, Narmak, Tehran, Iran. 16846-13114

* kahaei@iust.ac.ir

Abstract

We apply matrix completion methods for haplotype assembly from NGS reads to develop the new HapSVT, HapNuc, and HapOPT algorithms. This is performed by applying a mathematical model to convert the reads to an incomplete matrix and estimating unknown components. This process is followed by quantizing and decoding the completed matrix in order to estimate haplotypes. These algorithms are compared to the state-of-the-art algorithms using simulated data as well as the real fosmid data. It is shown that the SNP missing rate and the haplotype block length of the proposed HapOPT are better than those of HapCUT2 with comparable accuracy in terms of reconstruction rate and switch error rate. A program implementing the proposed algorithms in MATLAB is freely available at https://github.com/smajidian/HapMC.

Introduction

The Single Nucleotide Polymorphism (SNP) is a kind of genetic variation with a frequency greater than 1% in population. In diploid organisms, genomes are organized into pairs of chromosomes, a paternal and a maternal copy. The sequence of SNPs on each copy of a pair of chromosomes is called a haplotype. A genotype is the conflation of two haplotypes on the homologous chromosomes. An SNP is called homozygous, if a pair of alleles at this locus is made up of two identical nucleotides, and is heterozygous, otherwise.

From the evolutionary point of view, the SNP happens as a consequence of mutation. However, since the mutation rate is low, several mutations of a locus rarely occur. Thus, it is usual to assume that the majority of SNPs are bi-allelic, meaning that each SNP can be chosen from just two of the four possible nucleotides, *i.e.*, A, T, C, and G [1]. Accordingly, in this work we similarly use this assumption. The haplotype is widely used in the Genome Wide Association Studies (GWAS), clinical genetics, linkage analysis, drug-design, and personalized medicine [2].

To extract a haplotype, one may use the following three approaches where the last two approaches are mathematical:

1. Applying high-cost experimental and expensive methods for every single individual which is of course not desirable [2].

2. Haplotype phasing wherein the haplotypes are inferred from the genotypes of multiple individuals. As such, a method based on the maximum parsimony assumption [3] and statistical methods like SHAPEIT, developed based on the Hidden Markov Model [1, 4] may be mentioned. Note that using this approach, the haplotype of an individual can not be found separately and also is challenged by the low-frequency and also *de novo* variants [2].

3. Estimating haplotypes from Next Generation Sequencing (NGS) reads *i.e.* nucleotide sequence of fragments. Using this approach, known as the haplotype assembly, haplotyping of a single individual becomes feasible. In this regard, HapCUT2 [5], HapTree [6], and HapSAT [7] are three famous methods developed based on probabilistic models. These methods are sensitive to the selected model and thus fragile to the model error.

A recent method for haplotype assembly is AltHap [8] which has shown accurate results compared to H-PoP [9], SCGD [10], and HapTree [6]. The H-PoP is a heuristic algorithm originated from the Balanced Optimal Partition (BOP) optimization model which benefits from the Minimum Error Correction (MEC) as well as the maximum fragments cut approaches [11]. The SDhaP [12] is also another heuristic method based on correlation clustering and non-convex optimization which does not guarantee reaching the global optimum.

The innovation of this article is threefold. First, the haplotype assembly is mathematically formulated based on matrix completion methods. Secondly, three new algorithms called the Haplotype assembly based on Singular Value Thresholding (HapSVT), Haplotype assembly based on Nuclear norm minimization (HapNuc), and Haplotype assembly based on OPT-SPACE (HapOPT) are proposed. Next, in the section of Results, these algorithms are compared to some benchmark methods in terms of the reconstruction rate and the switch error rate.

Model of haplotypes

To exploit the NGS reads as the raw data, a computational modeling is needed. For this purpose, similar to [10], we first convert the sequence of nucleotides which can be either reads or haplotypes into a sequence of numbers. The SNP nucleotides are converted to 1 and -1 for the wild and rare alleles, respectively. As an example, Table 1 depicts the alleles of the β_2AR gene [3] for which the maternal and paternal haplotypes of an individual are shown by h_m and h_p, respectively. The corresponding codewords based on the above modeling are presented in the last column.

Next, assuming that each read has been aligned to the reference genome, the non-SNP sites of each read are omitted. Then, the reads are coded using the procedure described in Table 1, and are completed by adding zeros for the length of l as shown for 10 aligned reads in Table 2. As seen in this example, for the 1st row, we get {-1 1 1 0 0 0 0 0 0 0} with 3 sites of ± 1 and 7 sites of zeros.

Without loss of generality, by representing the codewords of Table 2 by the vectors r_i, $i = 1, \ldots, N$, we form the read matrix R, where N is the number of reads. In fact, R is an incomplete matrix with the rank of 2 which consists of the maternal and paternal haplotypes in its

Table 1. Haplotypes of β_2AR genes and their corresponding codewords.

	Nucleotides										Codewords
Alleles	G/A	C/A	G/A	C/G	T/C	T/C	T/C	G/A	C/G	G/A	{1/-1,1/-1,...}
h_m	A	C	G	G	C	C	C	G	G	G	{-1,1, 1,-1,-1,-1, 1,-1,1}
h_p	G	C	A	C	T	T	T	A	C	G	{ 1,1,-1, 1, 1, 1, 1,-1, 1,1}

Table 2. Example of aligned reads for β_2AR genes and the considered codewords.

Reads	Nucleotides										Codewords									
1	A	C	G								-1	1	1	0	0	0	0	0	0	0
2			G	G	C	C					0	0	1	-1	-1	-1	0	0	0	0
3			G	G					G	G	0	0	1	-1	0	0	0	0	-1	1
4	G	C	A	C	T	T					1	1	-1	1	1	1	0	0	0	0
5			A	C			T	A	C	G	0	0	-1	0	0	1	1	-1	1	1
6	G	C			T	T					1	1	0	0	1	1	0	0	0	0
7		C			C					G	-1	1	0	0	-1	0	0	0	0	1
8	A	C			C	C	C				-1	1	0	0	-1	-1	-1	0	0	0
9	G			C			T	A	C		1	0	0	1	0	0	1	-1	1	0
10			A	C					C	G	0	0	-1	1	0	0	0	0	1	1

rows. At this stage, we may utilize matrix completion methods to complete this low rank matrix. To do so, by estimating the zero entries of R, we obtain the completed matrix H which has the same dimension as R, *i.e.*, $N \times l$ where l is the haplotype length. According to Table 2, these matrices are given by (1) and (2).

$$R = \begin{bmatrix} -1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & -1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & -1 & 1 \\ 1 & 1 & -1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 1 & 1 & -1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 1 \\ -1 & 1 & 0 & 0 & -1 & -1 & -1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 1 & -1 & 1 & 0 \\ 0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix} \tag{1}$$

$$H = \begin{bmatrix} -1 & 1 & 1 & -1 & -1 & -1 & -1 & 1 & -1 & 1 \\ -1 & 1 & 1 & -1 & -1 & -1 & -1 & 1 & -1 & 1 \\ -1 & 1 & 1 & -1 & -1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & 1 & 1 & 1 & 1 & -1 & 1 & 1 \\ 1 & 1 & -1 & 1 & 1 & 1 & 1 & -1 & 1 & 1 \\ 1 & 1 & -1 & 1 & 1 & 1 & 1 & -1 & 1 & 1 \\ -1 & 1 & 1 & -1 & -1 & -1 & -1 & 1 & -1 & 1 \\ -1 & 1 & 1 & -1 & -1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & 1 & 1 & 1 & 1 & -1 & 1 & 1 \\ 1 & 1 & -1 & 1 & 1 & 1 & 1 & -1 & 1 & 1 \end{bmatrix} \tag{2}$$

From H, one can observe that only two of its rows are different and thus the desired haplotypes are given by

$$\boldsymbol{h}_m = \begin{bmatrix} -1 & 1 & 1 & -1 & -1 & -1 & -1 & 1 & -1 & 1 \end{bmatrix}, \tag{3}$$

$$\boldsymbol{h}_p = \begin{bmatrix} 1 & 1 & -1 & 1 & 1 & 1 & 1 & -1 & 1 & 1 \end{bmatrix}. \tag{4}$$

These vectors can then be decoded to the sequence of nucleotides using the first row of Table 1. To the best of our knowledge, no algorithm has been reported to distinguish between the maternal and paternal haplotypes and therefore $\boldsymbol{h}_p$ and $\boldsymbol{h}_m$ may be interchanged with each other.

It should be noted that the above example is an error-free case to clarify the procedure of data modeling which can be trivially solved. For the erroneous case; which is the subject of our work, $\boldsymbol{R}$ is an incomplete version of $\boldsymbol{H} + \boldsymbol{N}$ where $\boldsymbol{N}$ shows the noise matrix [8].

Proposed methods

We present three new algorithms for haplotype assembly whose general block diagram is illustrated in Fig 1. The goal is to estimate $\boldsymbol{h}_p$ and $\boldsymbol{h}_m$ from the noisy reads. The first two blocks have been explained before. In the third block, we receive an incomplete matrix $\boldsymbol{R}$ with a few known entries where the set of indices of known entries is given by Ω [10]. Then, we intend to estimate the unknown entries based on rank assumption. Mathematically, this is modeled by the following optimization problem:

$$\min_{\boldsymbol{H}} \sum_{(i,j) \in \Omega} (H_{ij} - R_{ij})^2 \text{ subject to } \text{rank}(\boldsymbol{H}) = 2. \tag{5}$$

It is worth mentioning that here we have not only considered the case of all-heterozygous variants, but also included the case of both heterozygous and homozygous variants. This can be realized as a point of this work in comparison to some other methods that are restricted to heterozygous variants. In the all-heterozygous case, the two haplotypes will be the negative of each other, $i.e.$, $\boldsymbol{h}_p = -\boldsymbol{h}_m$ and thus the rank of $\boldsymbol{H}$ will be one (See (5)).

To solve (5), the nuclear norm minimization, Singular Value Thresholding (SVT), and OPTSPACE methods have already been reported [13], based on which we introduce three new algorithms called the HapSVT, HapNuc, and HapOPT.

Haplotype assembly based on Singular Value Thresholding (HapSVT)

To explain the proposed HapSVT algorithm, we first introduce the SVT which is based on Singular Value Decomposition (SVD) [14] defined for the read matrix $\boldsymbol{R}$ as

$$\boldsymbol{R} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^H, \quad \boldsymbol{\Sigma} = \text{diag}(\sigma_i) \quad i = 1, ..., r \tag{6}$$

where H denotes the hermitian operator, and $\boldsymbol{U}$ and $\boldsymbol{V}$ have orthonormal columns with the dimension of $N \times r$ and $l \times r$, respectively. By applying the singular value shrinkage operator
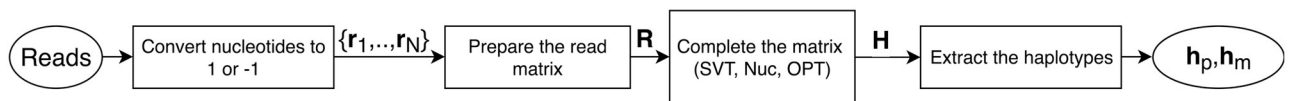

Fig 1. Block diagram of the proposed algorithms.

$D_\tau(\cdot)$ to $\boldsymbol{R}$, we obtain

$$D_\tau(\boldsymbol{R}) = \boldsymbol{U} D_\tau(\Sigma) \boldsymbol{V}^H, \tag{7}$$

where

$$D_\tau(\Sigma) = \mathrm{diag}(\max\{\sigma_i - \tau, 0\}). \tag{8}$$

It is worth noting that $D_\tau(\boldsymbol{R})$ is the optimal value of the optimization problem

$$\min_{\boldsymbol{Z}} \frac{1}{2} \|\boldsymbol{R} - \boldsymbol{Z}\|_F^2 + \tau \|\boldsymbol{Z}\|_*, \tag{9}$$

where $\|\cdot\|_F$ is the Frobenius norm and $\|\cdot\|_*$ shows the nuclear norm as the summation of singular values.

To perform the matrix completion part as shown in Fig 1, we recursively use the SVT in two steps. In the first step, starting with the initial matrix $\boldsymbol{Y}^0 = \boldsymbol{R}$, the singular value shrinkage operator is used as

$$\boldsymbol{X}^k = D_\tau(\boldsymbol{Y}^{k-1}). \tag{10}$$

Then, in the second step, the difference between the projected matrix $\boldsymbol{X}^k$ and the initial matrix is compensated for the known entries using

$$\boldsymbol{Y}^k = \boldsymbol{Y}^{k-1} + \delta \mathcal{P}_\Omega(\boldsymbol{R} - \boldsymbol{X}^k), \tag{11}$$

for $k = 1, 2, \ldots$, where $\mathcal{P}_\Omega(\cdot)$ is an operator which keeps the entries of the matrix corresponding to Ω unchanged, and sets the other entries to zero. The iterations continue until the condition $\|\mathcal{P}_\Omega(\boldsymbol{X}^k - \boldsymbol{R})\|_F < \epsilon \|\boldsymbol{R}\|_F$ is satisfied and the last $\boldsymbol{X}^k$ is reported as the completed matrix $\boldsymbol{H}$.

To extract $\boldsymbol{h}_p$ and $\boldsymbol{h}_m$, we compute the reduced row echelon form of $\boldsymbol{H}$ and by using the first two pivot positions, two independent rows of $\boldsymbol{H}$ are obtained. Then, in order to acquire the paternal and maternal haplotypes the entries are quantized to 1 and −1. The procedures of the HapSVT algorithm is depicted in Algorithm 1.

Algorithm 1: Haplotype assembly using SVT (HapSVT).

```
input: N aligned reads
output: Haplotypes hm, hp
/* Read Matrix Preparation */
1 Convert the sequences of nucleotides (reads) to the sequences of
  numbers.
2 Add zeros to each read to construct ris with the length of l.
3 Construct the read matrix R (N × l).
  /* Matrix Completion (SVT) */
4 Initialize Y0 = R, k = 0, i = 1.
5 while ‖PΩ(Xk − R)‖F < ε‖R‖F do
6     k = k + 1
7     Xk = Dτ(Yk−1)
8     Yk = Yk−1 + δPΩ(R − Xk)
9 end
10 H = Xk
  /* Reduced Row Echelon Form (RREF) Calculation */
11 [Hr, p] = RREF(HT)
  /* Haplotype Extraction */
12 Hq = 2 * (H > 0) − 1
13 hp = Hq(p(1),:)
14 hm = Hq(p(2),:)
15 Convert the entries of hm and hp to the nucleotides.
```

Haplotype assembly based on Nuclear norm minimization (HapNuc)

A popular method for matrix completion is based on relaxing the non-convex rank function to a convex function. Since the number of nonzero singular values determines the rank of a matrix, an approximation of the rank function is defined by the summation of singular values, known as the nuclear norm [15]. In this way, the optimization problem is cast as

$$\min_{\boldsymbol{H}} \|\boldsymbol{H}\|_* \quad \text{subject to } \|\mathcal{P}_\Omega(\boldsymbol{H} - \boldsymbol{R})\|_F < \epsilon. \tag{12}$$

This problem can be solved easily using the CVX, a MATLAB based package [16]. It has been shown that the nuclear norm minimization has strong mathematical guarantees to achieve the optimal solution [15, 17, 18]. To develop the new HapNuc algorithm, we substitute the SVT part of Algorithm 1 by nuclear norm minimization.

Haplotype assembly based on OPTSPACE (HapOPT)

Another method for matrix completion is known as OPTSPACE [19] in which unlike the two previous methods, we assume that the rank of the desired matrix $\boldsymbol{H}$ is known. The OPTSPACE consists of the following three steps: a) trimming, b) projection, and c) cleaning, as explained below.

a) In the trimming step, those columns of $\boldsymbol{R}$ with the degrees larger than $2|\Omega|/l$ are set to zero where $|\cdot|$ shows the cardinality of a set and l is the haplotype length. The degree of a column (or a row) shows the number of its known entries. This step is also performed for the rows of $\boldsymbol{R}$ with the degrees larger than $2|\Omega|/N$ where N is the number of reads.

b) The trimmed $\boldsymbol{R}$ obtained from Step (a) is projected to the space of rank r matrices using

$$P(\boldsymbol{R}) = \frac{Nl}{|\Omega|} \boldsymbol{U} P_r(\Sigma) \boldsymbol{V}^H, \tag{13}$$

where $P_r(\Sigma) = \text{diag}(\sigma_1, \ldots \sigma_r)$ and $\boldsymbol{U}$ and $\boldsymbol{V}$ are given by (6).

c) The cleaning step is performed by solving the following optimization problem,

$$\min_{\boldsymbol{X} \in \mathbb{R}^{N \times r}, \boldsymbol{Y} \in \mathbb{R}^{l \times r}} \min_{\boldsymbol{S} \in \mathbb{R}^{r \times r}} \sum_{(i,j) \in \Omega} \left(\boldsymbol{R}_{ij} - \left(\boldsymbol{XSY}^H \right)_{ij} \right)^2, \tag{14}$$

which contains two minimization parts. The inner part results in a function in terms of $\boldsymbol{X}$ and $\boldsymbol{Y}$. To solve the outer minimization part, we use a gradient based recursive method whose initial matrices are computed from Step (b), i.e., $\boldsymbol{X}_0 = \boldsymbol{U}$ and $\boldsymbol{Y}_0 = \boldsymbol{V}$. Then, this recursive method leads to the optimal solution $\boldsymbol{H} = \boldsymbol{X}_{\text{opt}} \boldsymbol{S}_{\text{opt}} \boldsymbol{Y}_{\text{opt}}^H$. To finalize the third new HapOPT algorithm, we should substitute the SVT part of Algorithm 1 by the above three steps.

Results

Using extensive simulations, we compare the performance of the proposed HapSVT, HapNuc, and HapOPT algorithms with that of the three recent benchmark algorithms AltHap [8], HapCUT2 [5], and SDhaP [12]. It has already been shown that these algorithms outperform some other algorithms like RefHap [20], SCGD [10], HapTree [6], and H-PoP [9]. For comparison purposes, a well-known criterion is the reconstruction rate defined as [21]

$$\text{rr} = 1 - \frac{1}{l} \min \left\{ \mathcal{HD}\left(\hat{\boldsymbol{h}}_m, \boldsymbol{h}_m \right), \mathcal{HD}\left(\hat{\boldsymbol{h}}_p, \boldsymbol{h}_p \right) \right\}, \tag{15}$$

where $\hat{\boldsymbol{h}}_p$ and $\hat{\boldsymbol{h}}_m$ are the reconstructed haplotypes which are compared to the known maternal and paternal haplotypes, $\boldsymbol{h}_m$ and $\boldsymbol{h}_p$. Moreover, $\mathcal{HD}(\cdot, \cdot)$ is the augmented hamming distance between two vectors which counts the number of non-identical sites using

$$\mathcal{HD}(\boldsymbol{a}, \boldsymbol{b}) = \sum_{j=1}^{l} \mathcal{D}(\boldsymbol{a}(j), \boldsymbol{b}(j)), \tag{16}$$

where $\mathcal{D}(\cdot, \cdot)$ is defined as

$$\mathcal{D}(a, b) = \begin{cases} 0 & a = b \\ 1 & \text{otherwise.} \end{cases} \tag{17}$$

To consider another criterion for performance evaluation, we make use of the SWitch Error Rate (SWER), defined as the number of switches divided by the haplotype length [22]. A switch happens when the parental origin of an allele with respect to that of the previous allele differs from one parent to another. For example, by considering $\boldsymbol{h}_p = [1, 1, 1, 1]$ and $\boldsymbol{h}_m = [-1, -1, -1, -1]$ as the grand truth haplotypes and the estimated haplotypes as $\hat{\boldsymbol{h}}_p = [1, 1, -1, -1]$ and $\hat{\boldsymbol{h}}_m = [-1, -1, 1, 1]$, one switch has been occurred.

Simulated data

First, we use the simulated data [21] generated based on real human haplotypes in the HapMap project. This dataset; which contains different read matrices with various error rates and coverage values originated from different haplotype lengths, has vastly been used in previous studies [10, 23, 24]. We choose the longest available haplotype from the dataset with the length of l = 700. The coverage value of the NGS paired-end reads varies from c = 3 to its greatest value c = 10. The average number of reads are N = 561, 936, and 1873 for coverage values of c = 3, 5, and 10, respectively. The number of SNPs covered in each read is a constant value equal to 7.4. Also, 10% (and 20%) of the entries of the read matrix are contaminated by noise with uniform distribution. The results are averaged over 100 independent trials of the experiment.

Table 3 shows the reconstruction rates for different coverage values and error rates. The corresponding SWERs are also depicted in Table 4. In this case, HapCUT2 is not examined, since it needs the Variant Call Format (VCF) file which is not available for this simulated dataset [21]. As seen in both Tables 3 and 4, the proposed HapOPT algorithm outperforms the others in terms of the reconstruction rate as well as the SWER. It is worth reminding that the SDhaP solves a non-convex optimization problem using a heuristic technique with the gradient descent algorithm which does not guarantee reaching the global optimum. Furthermore, as a consequence of increasing the coverage value, a better performance is achieved by a lower SWER and a higher reconstruction rate.

Table 3. Reconstruction rates for different algorithms on simulated data [21]. The best values are in boldface.

coverage	error rate (%)	SDhaP	AltHap	HapOPT(Proposed)	HapSVT(Proposed)	HapNuc(Proposed)
3	10	97.87	99.04	**99.07**	98.38	98.32
5	10	99.19	99.66	**99.72**	97.21	98.82
10	10	99.64	**1**	**1**	99.53	99.64
3	20	96.66	97.32	**97.38**	97.00	97.31
5	20	97.36	98.24	**98.43**	97.47	97.47
10	20	97.02	**99.45**	99.25	98.66	98.6

Table 4. SWERs for different algorithms on simulated data [21]. The best values are in boldface.

coverage	error rate (%)	SDhaP	AltHap	HapOPT (Proposed)	HapSVT (Proposed)	HapNuc (Proposed)
3	10	0.070	0.038	**0.027**	0.111	0.120
5	10	0.019	0.0058	**0.004**	0.207	0.049
10	10	0.0018	**0**	**0**	0.012	0.003
3	20	0.227	0.247	**0.218**	0.350	0.345
5	20	0.136	0.123	**0.101**	0.243	0.266
10	20	0.065	**0.0178**	0.018	0.080	0.121

Real fosmid data

We evaluate the proposed algorithms on the sequence data of the individual NA12878 fabricated based on a fosmid approach [20]. The coverage of this data set is $c = 3$ and the average read length is 40 kb, and hence, is a low-coverage and long-read dataset. For evaluation purposes, we consider the trio-phased haplotype from the GATK resource bundle, as the grand truth containing 1.3 million heterozygous variants in common with fosmid dataset [22, 25]. This dataset has already been used in several studies [5, 8, 22].

In the simulated dataset used in the last section, each read overlaps at least one another read, while for the real data these overlaps do not necessarily occur. In this situation, our algorithm incorporates the overlaps for haplotype estimation, and as a result, the output of each algorithm is some disjoint parts of the whole haplotype, called haplotype blocks. To evaluate a common length for these blocks, we consider their mean and also the AN50 defined as the median of blocks lengths in base pairs weighted by a proportion of correctly estimated alleles [6]. Also, we define the SNP Missing Rate (SMR) for each chromosome as the ratio of the number of missing SNPs in the estimates and the haplotype length [26]. The results on the real fosmid data are shown in Table 5. One can see that both HapOPT and AltHap algorithms achieve lower SNP missing rates in comparison to HapCUT2 and SDhaP. Moreover, HapOPT and AltHap have a better span in terms of AN50.

To assess the accuracy of different algorithms, the corresponding reconstruction rates [5, 22] are presented in Fig 2. Moreover, we have considered both short and long SWERs [5, 22]. By a long switch, we mean that the parental origin does not change for at least two SNPs and if two switches occur one after each other, we consider it as a short switch. These two metrics are reported on real fosmid data in Figs 3 and 4.

From the above results, one can observe that HapOPT outperforms SDhaP and AltHap in terms of the reconstruction rate as well as long and short SWERs with a reasonable runtime as reported in Table 6. Note that although, HapCUT2 achieves the best accuracy, still its SNP missing rate is greater than that of HapOPT. These results on the whole show that HapOPT is a promising tool for haplotype assembly with the best SNP missing rate and a good accuracy in terms of reconstruction rate and SWER.

Conclusion

We have exploited matrix completion methods including SVT, nuclear norm minimization, and OPTSPACE for haplotype estimation. This was led to developing the new HapSVT, Hap-Nuc, and HapOPT algorithms. Our experimental comparison on simulated data revealed that HapOPT is more accurate than SDhaP and AltHap in terms of reconstruction rate and switch error rate. Also, the results on real noisy fosmid data showed that the accuracy of HapOPT is better than that of SDhaP and AltHap and also is comparable to that of HapCUT2 in terms of the reconstruction rate and the short and long SWERs. Moreover, it was shown that HapOPT

Table 5. Mean and AN50 of haplotype blocks lengths for different algorithms on real fosmid data.

| Chr.	SDhaP			HapCUT2			AltHap			HapOPT (Proposed)		
	SMR	Mean	AN50 (kb)	SMR	Mean	AN50 (kb)	SMR	Mean	AN50 (kb)	SMR	Mean	AN50(kb)
1	6.2	71.5	254	6.7	71.1	229	6.2	72.7	234	6.2	72.7	234
2	6.9	68.6	241	8.3	68.3	219	6.9	69.7	223	6.9	69.7	223
3	8.1	69.7	218	8.6	69.3	195	8.0	70.6	204	8.0	70.6	204
4	10.0	63.4	192	10.4	63.1	172	9.9	64.6	177	9.9	64.6	177
5	8.2	69.5	219	8.8	69.0	206	8.2	70.3	210	8.2	70.3	210
6	7.3	82.4	243	7.9	81.9	224	7.3	84.0	236	7.3	84.0	236
7	7.2	69.7	222	7.6	69.5	207	7.1	71.0	212	7.1	71.0	212
8	7.8	75.6	229	8.3	75.2	207	7.7	76.8	220	7.7	76.8	220
9	7.0	79.6	249	7.5	79.2	230	6.9	80.9	235	6.9	80.9	235
10	6.8	83.9	238	7.3	83.4	217	6.7	84.9	220	6.7	84.9	220
11	7.1	77.1	234	7.5	76.8	225	7.0	78.3	228	7.0	78.3	228
12	6.4	73.4	262	7.3	73.0	241	6.7	74.1	249	6.7	74.1	249
13	10.2	69.1	203	10.7	68.7	186	10.1	70.3	191	10.1	70.3	191
14	6.5	77.5	259	7.0	77.1	238	6.3	78.4	246	6.3	78.4	246
15	6.0	73.7	251	6.4	73.2	228	5.9	74.1	234	5.9	74.1	234
16	3.8	96.6	345	4.2	96.2	317	3.7	97.9	327	3.7	97.9	327
17	3.9	70.8	323	4.5	70.4	305	3.9	71.5	310	3.9	71.5	310
18	7.1	75.3	228	7.6	74.9	216	7.0	76.0	223	7.0	76.0	223
19	3.1	90.8	374	3.5	90.4	345	3.0	93.8	360	3.0	93.8	360
20	4.3	92.4	314	4.8	92.0	297	4.2	93.7	304	4.2	93.7	304
21	6.6	81.1	252	7.0	80.8	242	6.4	82.4	242	6.4	82.4	242
22	2.7	123.7	445	3.2	123.2	425	2.6	123.9	426	2.6	123.9	426
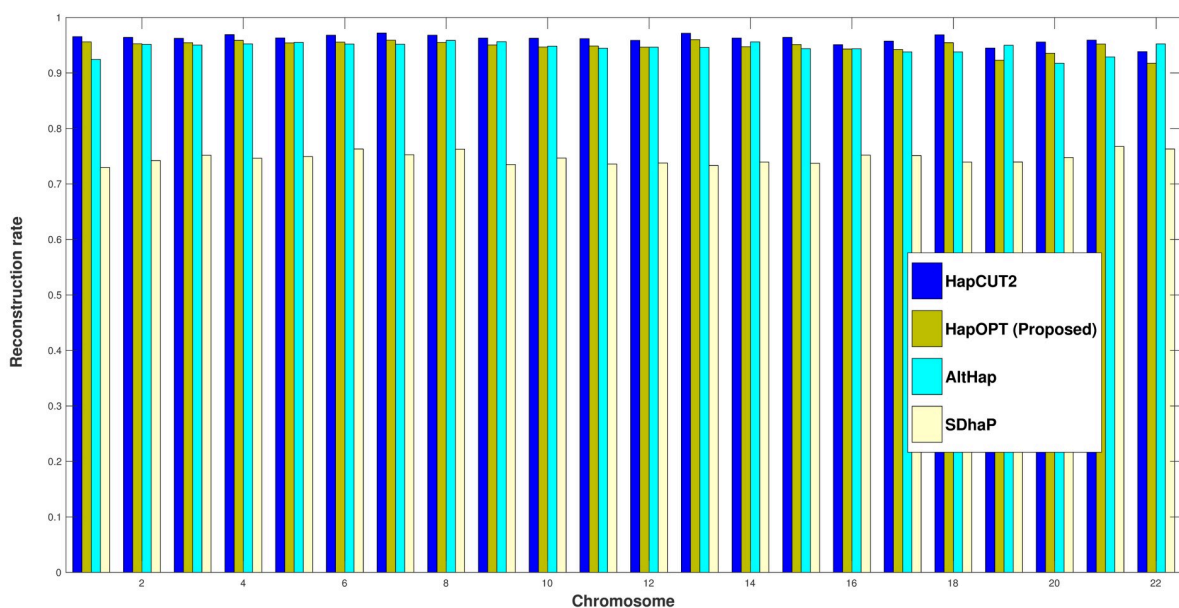
Fig 2. Reconstruction rate of HapOPT, HapCUT2, AltHap, and SDhaP on real fosmid data.
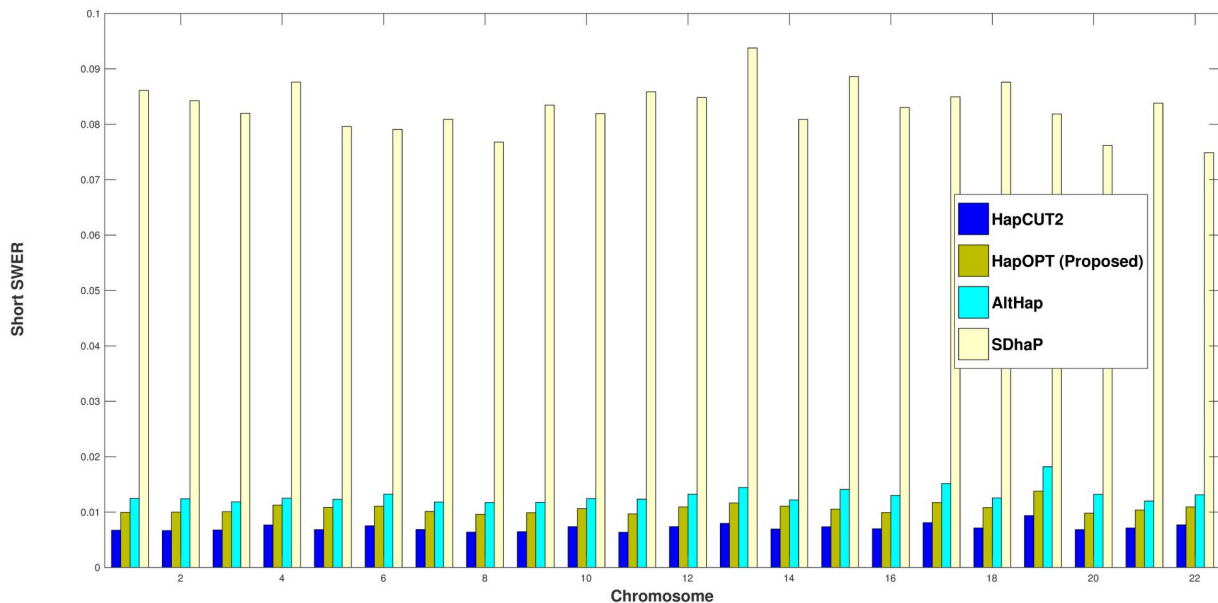
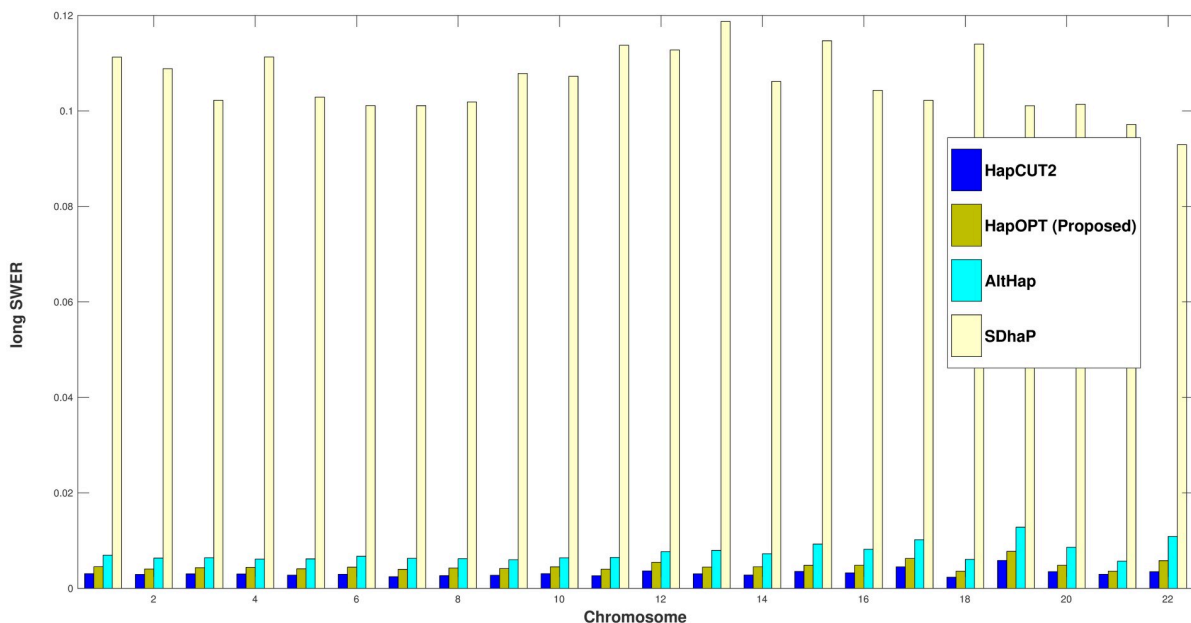**Fig 3. Short SWER of HapOPT, HapCUT2, AltHap, and SDhaP on real fosmid data.**

Fig 4. Long SWER of HapOPT, HapCUT2, AltHap, and SDhaP on real fosmid data.

Table 6. Runtime of HapOPT, HapCUT2, AltHap, and SDhaP on real fosmid data.

	SDhaP	AltHap	HapCUT2	HapOPT (Proposed)
Runtime (Minutes)	5	10	18	355

outperforms the recently addressed algorithms, HapCUT2 and SDhaP, in terms of the mean, SNP missing rate, and AN50 of the haplotype block length. Furthermore, the proposed algorithm is not restricted to the heterozygous assumption, as commonly considered in peer algorithms. On the whole, we can conclude that using the proposed HapOPT, the haplotype is reconstructed more completely and continuously with acceptable accuracy. Also, the proposed optimization problem is capable of estimating haplotypes for different ploidy levels. Our research direction for future is to work on polyploids.

Author Contributions

Formal analysis: Sina Majidian.

Methodology: Sina Majidian, Mohammad Hossein Kahaei.

Supervision: Mohammad Hossein Kahaei.

Validation: Mohammad Hossein Kahaei.

Writing – original draft: Sina Majidian.

Writing – review & editing: Mohammad Hossein Kahaei.

References

1. Delaneau O, Marchini J, Zagury JF. A linear complexity phasing method for thousands of genomes. *Nat Methods*, 9(2):179–181, 2012. https://doi.org/10.1038/nmeth.1785

2. Snyder MW, Adey A, Kitzman JO, Shendure J. Haplotype-resolved genome sequencing: experimental methods and applications. *Nat Rev Genet*, 16(6):344–358, 2015. https://doi.org/10.1038/nrg3903 PMID: 25948246

3. Wang L, Xu Y. Haplotype inference by maximum parsimony. *Bioinformatics*, 19(14):1773–1780, 2003. https://doi.org/10.1093/bioinformatics/btg239 PMID: 14512348

4. O'Connell J, Sharp K, Shrine N, Wain L, Hall I, Tobin M, et al. Haplotype estimation for biobank-scale data sets. *Nat Genet*, 48(7):817–820, 2016. https://doi.org/10.1038/ng.3583 PMID: 27270105

5. Edge P, Bafna V, Bansal V. HapCUT2: robust and accurate haplotype assembly for diverse sequencing technologies. *Genome Res*, 27(5):801–812, 2017. https://doi.org/10.1101/gr.213462.116 PMID: 27940952

6. Berger E, Yorukoglu D, Peng J, Berger B. Haptree: A novel bayesian framework for single individual polyplotyping using NGS data. *PLoS Comput Bioly*, 10(3):e1003502, 2014. https://doi.org/10.1371/journal.pcbi.1003502

7. Mousavi SR, Khodadadi I, Falsafain H, Nadimi R, Ghadiri N. Maximum likelihood model based on minor allele frequencies and weighted max-sat formulation for haplotype assembly. *J Theor Biol*, 350:49–56, 2014. https://doi.org/10.1016/j.jtbi.2014.01.036 PMID: 24491253

8. Hashemi A, Banghua Z, Vikalo H. Sparse tensor decomposition for haplotype assembly of diploids and polyploids *BMC Genomics*, 19(Suppl 4):191, 2018. PMID: 29589554

9. Xie M, Wu W, Wang J, Jiang T. H-PoP and H-PoPG: Heuristic partitioning algorithms for single individual haplotyping of polyploids *Bioinformatics*, 32(24):3735–3744, 2016. https://doi.org/10.1093/bioinformatics/btw537 PMID: 27531103

10. Cai C, Sanghavi S, Vikalo H. Structured low-rank matrix factorization for haplotype assembly. *IEEE J Sel Top Signal Process*, 10(4):647–657, 2016. https://doi.org/10.1109/JSTSP.2016.2547860

11. Xie M, Wu M, Wang J, Jiang T. A fast and accurate algorithm for single individual haplotyping *BMC Syst Biol*, 6(Suppl 2):S8, 2012. https://doi.org/10.1186/1752-0509-6-S2-S8 PMID: 23282221

12. Das S, Vikalo H. SDhaP: Haplotype assembly for diploids and polyploids via semi-definite programming. *BMC Genomics*, 16(1):260, 2015. https://doi.org/10.1186/s12864-015-1408-5 PMID: 25885901

13. Davenport MA, Romberg J. An overview of low-rank matrix recovery from incomplete observations. *IEEE J Sel Top Signal Process*, 10(4):608–622, 2016. https://doi.org/10.1109/JSTSP.2016.2539100

14. Cai JF, Candes EJ, Shen Z. A singular value thresholding algorithm for matrix completion *SIAM J Optim*, 20(4):1956–1982, 2010. https://doi.org/10.1137/080738970

15. Candes EJ, Tao T. The power of convex relaxation: Near-optimal matrix completion. *IEEE Trans Inf Theory*, 56(5):2053–2080, 2010. https://doi.org/10.1109/TIT.2010.2044061

16. Grant M, Boyd S CVX: Matlab software for disciplined convex programming. 2013, Available from: http://cvxr.com/cvx

17. Candes EJ, Recht B. Exact matrix completion via convex optimization. *Found Comut Math*, 9(6):717, 2009. https://doi.org/10.1007/s10208-009-9045-5

18. Recht B, Fazel M, Parrilo PA. Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization. *SIAM Rev Soc Ind Appl Math*, 52(3):471–501, 2010.

19. Keshavan RH, Montanari A, Oh S. Matrix completion from a few entries. *IEEE Trans Inf Theory*, 56(6):2980–2998, 2010. https://doi.org/10.1109/TIT.2010.2046205

20. Duitama J, McEwen GK, Huebsch T, Palczewski S, Schulz S, Verstrepen K, et al. Fosmid-based whole genome haplotyping of a HapMap trio child: evaluation of single individual haplotyping techniques. *Nucleic Acids Res*, 40(5):2041–2053, 2011. https://doi.org/10.1093/nar/gkr1042 PMID: 22102577

21. Geraci F. A comparison of several algorithms for the single individual SNP haplotyping reconstruction problem. *Bioinformatics*, 26(18):2217–2225, 2010. https://doi.org/10.1093/bioinformatics/btq411 PMID: 20624781

22. Kuleshov V. Probabilistic single-individual haplotyping. *Bioinformatics*, 30(17):379–385, 2014. https://doi.org/10.1093/bioinformatics/btu484

23. Deng F, Cui W, Wang L. A highly accurate heuristic algorithm for the haplotype assembly problem *BMC Genomics*, 14:S2, 2013. https://doi.org/10.1186/1471-2164-14-S2-S2 PMID: 23445458

24. Chen ZZ, Deng F, Wang L. Exact algorithms for haplotype assembly from whole-genome sequence data *Bioinformatics*, 29(16):1938–1945, 2013. https://doi.org/10.1093/bioinformatics/btt349 PMID: 23782612

25. DePristo MA, Banks E, Poplin R, Garimella KV, Maguire JR, Hartl C, et al. A framework for variation discovery and genotyping using next-generation DNA sequencing data. *Nat Genet*, 43(5):491–498, 2011. https://doi.org/10.1038/ng.806 PMID: 21478889

26. Motazedi E, Finkers R, Maliepaard C, de Ridder D. Exploiting next-generation sequencing to solve the haplotyping puzzle in polyploids: a simulation study. *Brief Bioinform*, 19(3):387–403, 2017.