

# On a Fixed Haplotype Variant of the Minimum Error Correction Problem

Axel Goblet, Steven Kelk, Matúš Mihalák, and Georgios Stamoulis

Department of Data Science & Knowledge Engineering  
Maastricht University, The Netherlands

a.goblet@student.maastrichtuniversity.nl, {matus.mihalak, steven.kelk,  
georgios.stamoulis}@maastrichtuniversity.nl

**Abstract.** Haplotype assembly is the problem of reconstructing the two parental chromosomes of an individual from a set of sampled DNA-sequences. A combinatorial optimization problem that models haplotype assembly is the *Minimum Error Correction* problem (MEC). This problem has been intensively studied in the computational biology literature and is also known in the clustering literature: essentially we are required to find two cluster centres such that the sum of distances to the nearest centre, is minimized. We introduce here the problem *Fixed haplotype-Minimum Error Correction* (FH-MEC), a new variant of MEC which corresponds to instances where one of the haplotypes/centres is already given. We provide hardness results for the problem on various restricted instances. We also propose a new and very simple 2-approximation algorithm for MEC on binary input matrices.

## 1 Introduction

Humans have, genetically speaking, an extremely high degree of similarity: at the vast majority of positions in our DNA sequence we share the same DNA symbol. The relatively few positions at which we differ are known as Single Nucleotide Polymorphisms (SNP) [5]. In most cases the variation observed at a given position involves two nucleotides (as opposed to three or four). For this reason the SNPs of an individual can be summarized as a string over a binary alphabet, also known as a *haplotype*.

A classical computational challenge in the genomic era is to efficiently infer such haplotypes from a set of overlapping, aligned haplotype fragments which have been obtained by sequencing the DNA at different intervals. This problem is complicated by the fact that humans (and diploid organisms in general) actually have two haplotypes (chromosomes): one inherited from the mother, and one from the father. We do not know easily which haplotype fragment originated from which of the two haplotypes, so the goal is to construct *two* haplotypes and to map the fragments to these two haplotypes. This is the haplotype assembly problem [11]. The *Minimum Error Correction* (MEC) model imposes the following objective function on the selection of the haplotypes: find two haplotypes

such that, summing over all the input fragments, the Hamming distance (interpreted as ‘errors corrected’) from the fragment to its nearest haplotype, is minimized. (When a fragment contains no information about a given position, a ‘wildcard’ character is used which does not contribute to the Hamming distance). The haplotypes can be thought of as cluster centres. This problem, originally introduced in 2001 [10], has been intensively studied in the computational biology literature: we refer to articles [7, 3, 2, 4] and the references therein for a comprehensive overview. We assume that the optimal solution has always cost greater than zero since, otherwise, it is a trivial task to find the optimal solution.

Without restrictions on the use of wildcards it is straightforward to show NP-hardness, and the problem remains hard under a number of natural restrictions. For many years, however, it was unclear whether the problem is NP-hard if there are *no* wildcards in the input: this is the BINARYMEC problem. This was finally settled in 2014 by Uriel Feige, who showed that the equivalent *Hypercube 2-segmentation problem* is NP-hard [8].

Here we present a new variant of the problem: *Fixed-haplotype Minimum Error Correction* (FH-MEC). In this version of the problem one fixed haplotype (not necessarily the optimal one) is given as part of the input, and we are asked to find the other that minimizes the total error correction. This is a quite natural variation which models the situation when one of the haplotypes has already been determined. Fast algorithms for FH-MEC could also be used to heuristically explore the space of solutions to MEC, and thus to provide warm-start upper bounds for MEC algorithms.

It is straightforward (by simply adding many copies of the fixed haplotype to the input) to reduce FH-MEC to MEC in an approximation-preserving way (under preservation of common restrictions on the use of wildcards), so the Binary Fixed-Haplotype variant of MEC, BINARYFH-MEC, inherits the PTAS that via the clustering literature was already known to exist for BINARYMEC [9, 12, 4]. Determining the complexity of FH-MEC is, however, a more involved task since there is no obvious reduction in the opposite direction. We show in this article that FH-MEC is APX-hard by providing an L-reduction from the MAXCUT problem on cubic graphs. Our central result is a proof that BINARYFH-MEC is NP-hard. This is a non-trivial adaption of the elegant proof by Feige [8]. Feige, who reduces from MAXCUT, works purely with  $\ell_1$ -norms, but unfortunately this option is not open to us due to the presence of the fixed haplotype, which cannot be interpreted this way. Another difficulty posed is that we also have to explicitly identify a fixed haplotype sufficient to induce hardness, and deal with a number of subtle technicalities concerning the way Hadamard (sub)matrices, and submatrices encoding the endpoints of graph edges (from the MAXCUT instance), are divided between the fixed haplotype and the variable haplotype. Although the NP-hardness of BINARYFH-MEC implies the NP-hardness of UNGAPPEDFH-MEC (where each haplotype fragment covers a *contiguous* interval of positions), we show an alternative NP-hardness reduction for this problem which is much simpler and potentially easier to manipulate into stronger forms of hardness, and thus of independent interest.

Ending the article on a positive note, we return to BINARYMEC. We follow the trend towards simplification given in [3] and provide another very simple polynomial-time 2-approximation algorithm for this problem. Our algorithm has, compared to [3], lower polynomial dependency on the length of the haplotypes we are constructing (at the expense of higher dependency on the number of fragments).

**Definitions and notations:** A *fragment matrix*  $F$  is a matrix with  $n$  rows and  $m$  columns, every entry of which is in  $\{-1, 1, *\}$ . A  $*$  entry is called a *hole*, and encodes an unknown value.  $F$  is *binary* if  $F$  contains no holes.  $F$  is *ungapped* if, for every row  $r \in F$ , there exists no hole in  $r$  such that there is a non-hole entry somewhere to the left and somewhere to the right of it.

Let  $r_i, r_j$  be two distinct rows of  $F$ . By  $r_i[k]$  we denote the  $k^{\text{th}}$  entry of  $r_i$ . Given two vectors  $r_i, r_j$  of the same dimension their (generalized) Hamming distance is defined as

$$d(r_i, r_j) = |\{k : r_i[k], r_j[k] \in \{-1, 1\}, r_i[k] \neq r_j[k]\}|. \quad (1)$$

i.e.,  $d(r_i, r_j)$  counts in how many positions the two vectors differ, where  $*$  characters in one vector induce no errors, no matter what is the corresponding entry of the other vector. The Minimum Error Correction (MEC) problem is defined as follows.

**Problem:** MEC

**Input:** An  $n \times m$  fragment matrix.

**Output:** Two  $m$ -dimensional vectors  $h_1, h_2 \in \{-1, 1\}^m$ , such that the following sum over all rows of  $F$  is minimized:

$$\sum_{r_i \in F} \min \left\{ d(h_1, r_i), d(h_2, r_i) \right\}.$$

In other words, the goal is to find two haplotypes  $h_1$  and  $h_2$  minimizing the sum of (generalized) Hamming distances of each row of  $F$  to its closest haplotype. This creates a bipartition of the rows into two groups, where rows that share the same closest haplotype are in the same group or *partition*. Ties can be broken arbitrarily. To make a row equal to its closest haplotype, the differing positions (errors) would have to be corrected. By minimizing the sum of these error corrections, the most likely parental haplotypes are found. Observe that a bipartition of the rows immediately induces two haplotypes by a simple majority voting rule on the rows within the same bipartition. Two variants of this problem, UNGAPPEDMEC and BINARYMEC, minimize the same function, but take as input an ungapped and binary fragment matrix, respectively.

In the Fixed-Haplotype MEC (FH-MEC) problem, one of the haplotypes  $h_1, h_2$  is fixed and part of the input:

**Problem:** FH-MEC

**Input:** An  $n \times m$  fragment matrix  $F$  and an  $m$ -dimensional vector  $h_1 \in \{-1, 1\}^m$  (the fixed centre).

**Output:** An  $m$ -dimensional vector  $h_2 \in \{-1, 1\}^m$ , such that the following is minimized:

$$\sum_{r_i \in F} \min \left\{ d(h_1, r_i), d(h_2, r_i) \right\}.$$

For FH-MEC, binary and ungapped variants exist as well. In this paper, the haplotypes  $h_1$  and  $h_2$  are sometimes described as the *fixed* and *variable* haplotypes, respectively.

Given a minimization problem  $\Pi$ , we say that an algorithm  $\mathcal{A}$  is a  $\rho$ -approximation algorithm if for any given instance  $I$  for  $\Pi$  (i)  $\mathcal{A}$  runs in polynomial time in the size of  $I$ , and (ii) it outputs a solution  $sol(I)$  with value at most  $\rho \cdot opt(I)$ . Here  $opt(I)$  corresponds to the optimal solution value for  $I$ . Note that  $\rho \geq 1$ .

## 2 APX-Hardness of FH-MEC

In this section we will prove that FH-MEC is APX-hard by showing that the CUBICMAXCUT problem, where the input graph is *cubic*, L-reduces [13] to our problem. A cubic graph is a graph where every vertex has exactly three adjacent vertices (i.e., the degree of each vertex is exactly three). MAXCUT is APX-hard, even for cubic graphs [1]. Moreover, the value of MAXCUT on cubic graphs has a lower bound of  $2/3$  of the number of the edges [4], which will be used to prove that the proposed reduction is indeed an L-reduction.

**Theorem 1.** FH-MEC is APX-hard.

*Proof.* Let  $G = (V, E)$  be an arbitrary, cubic, connected graph corresponding to an input to the CUBICMAXCUT problem. Let  $F$  be a  $|V| \times 2|E|$  fragment matrix to be constructed as follows: Every edge  $e \in E$  is represented by a block of two columns of  $F$ , and every vertex  $v \in V$  is represented by a row of  $F$ .  $F$  is constructed as follows: First arbitrarily orient the edges of the graph. For every edge  $e = (u, v)$ , set its corresponding columns in  $F$  to  $(1 \ -1)$  in row  $u$ , and to  $(-1 \ 1)$  in row  $v$  and set its corresponding columns in the other rows to  $(* \ *)$ .

A simple example for the cycle graph  $C_3$  on vertices  $\{1, 2, 3\}$  with orientations  $(1, 2), (2, 3), (1, 3)$  is given below.

$$F = \begin{pmatrix} 1 & -1 & * & * & 1 & -1 \\ -1 & 1 & 1 & -1 & * & * \\ * & * & -1 & 1 & -1 & 1 \end{pmatrix}$$

Now, let the fixed haplotype  $h_1$  be the all  $-1$  vector. We will first prove that  $\text{MAXCUT}(G) = c$  if and only if  $\text{UNGAPPEDFH-MEC}(F, h_1) = 2|E| - c$ . Then, we will show that the conditions of an L-reduction are satisfied. There are 2 cases to consider:

**Two vertices connected by an edge are in the same partition:** The rows containing  $(-1 \ 1)$  and  $(1 \ -1)$  will be in the same partition. If both rows are

closest to  $h_1$  (having value  $(-1 -1)$  in the two corresponding columns), then these columns will contribute 2 to the error correction. If both rows are closest to the variable haplotype  $h_2$ , any values on the corresponding columns of  $h_2$  will contribute 2 to the error correction.

**Two vertices connected by an edge are *not* in the same partition:** In every pair of columns representing an edge, there are only two rows filled with numbers. If these rows are placed in separate partitions, the row closest to  $h_1$  will increase the error correction by 1.  $h_2$  can be set to the other row without contributing to the error correction.

For vertices that are not connected by an edge, there is no column pair that is not already covered in the two cases discussed above. For every column pair covered in cases 1 and 2, the rows corresponding to these vertices will be  $(* *)$ . Thus, they do not contribute to the error correction.

For every edge, either case 1 or case 2 will hold. Edges that are split over the partitions (i.e., cut-edges) will contribute 1 to the error correction (case 2). Edges that are not split over the partitions will contribute 2 to the error correction (case 1). The minimum error correction is found by maximizing the number of cut edges. There are  $|E|$  edges in  $G$ . Therefore,  $c$  edges are cut iff the error correction of a solution is  $2|E| - c$ .

To complete the proof, we will show that the conditions of an L-reduction are satisfied. Let  $G$  be an instance of CUBICMAXCUT. Let  $R(G) = (F, h_1 = \{-1\}^{2|E|})$  be the instance of FH-MEC that is constructed from  $G$ . Clearly,  $R(G)$  can be constructed in polynomial time. Let  $Opt(G)$  be the value of the maximum cut of  $G$ . Let  $Opt(R(G))$  be the minimum error correction of  $R(G)$ . Lastly, let  $s$  be a feasible solution of  $R(G)$  and  $S(s)$  the corresponding solution for  $G$  and let  $c(s)$  and  $c(S(s))$  be their respective costs. According to the definition of an L-reduction [13], two conditions need to be satisfied:

$$Opt(R(G)) \leq \alpha Opt(G) \tag{2}$$

$$|Opt(G) - c(S(s))| \leq \beta |Opt(R(G)) - c(s)| \tag{3}$$

where  $\alpha, \beta$  are positive constants. We have that  $Opt(G) \geq 2/3|E|$  for any cubic graph  $G$ . We showed that  $Opt(R(G)) = 2|E| - Opt(G) \leq 2Opt(G)$ . Therefore, taking  $\alpha = 2$  will be sufficient to satisfy Equation (2). For any bipartition  $s$  of  $F$  of cost  $c(s)$ , the cost of the corresponding cut is  $c(S(s)) = 2|E| - c(s)$ . Thus,  $Opt(G) - c(S(s)) = Opt(G) - 2|E| + c(s)$ , and  $Opt(R(G)) - c(s) = 2|E| - Opt(G) - c(s)$ . This shows that  $Opt(G) - c(S(s)) = -(Opt(R(G)) - c(s))$ . Therefore, taking  $\beta = 1$  will satisfy Equation 3) and this completes the proof.  $\square$

### 3 NP-Hardness of BinaryFH-MEC

NP-hardness for a variant of BINARYMEC was proven by a reduction from MAXCUT [8]. In that variant the objective is to maximize the sum of the  $\ell_1$  norms of the vector sums of the rows of each bipartition, rather than to minimize the error correction as we are interested in this paper. Here, we will prove

NP-hardness for BINARYFH-MEC using a reduction inspired by [8]. A vanilla approach does not work and we need to resolve several technicalities that arise from the difference in the objective function and from the presence of a fixed center. In the following we will show that optimizing the one objective function is equivalent to optimizing the other by showing how to translate one objective function value to another. The following allows the conversion of an  $\ell_1$  norm to an error correction.

**Lemma 1.** *Let  $P$  be a subset of  $n_p$  rows of a binary fragment matrix  $F$ . The  $\ell_1$  norm of the sum of the rows of  $P$  is  $l$ , if and only if the contribution to the error correction of the rows is  $(n_p m - l)/2$ .*

*Proof.* Let  $n_{\text{maj}}$  be the number of bits in  $P$  that belong to the majority bit of their column. Let  $n_{\text{min}}$  be the bits that belong to minority bit of their column. If a column contains equal numbers of  $-1$ 's and  $1$ 's, the majority bit can be chosen arbitrarily. Let  $l$  be the  $\ell_1$  norm of the sum of the rows of  $P$ . The contribution of a column to  $l$  is the absolute difference between the number of majority and minority bits in that column. Therefore,  $l = n_{\text{maj}} - n_{\text{min}}$ .

The total number of bits in  $P$  is  $n_p m = n_{\text{maj}} + n_{\text{min}}$ . The contribution of  $P$  to the error correction is equal to the number of minority bits  $n_{\text{min}}$  in  $P$ . We have that  $n_{\text{min}} = n_p m - n_{\text{maj}} = n_p m - n_{\text{min}} - l$  from which we immediately get that  $n_{\text{min}} = (n_p m - l)/2$ .  $\square$

**Corollary 1.** *Given a bipartition of the rows of a binary matrix, the task of maximizing the  $\ell_1$  norm of the sum of the rows of each partition is equivalent to minimizing the error correction.*

The reduction involves the use of *Hadamard* matrices. We recall that an  $M$ -dimensional Hadamard matrix is a set of  $M$  row vectors in  $\{-1, 1\}^M$ , such that the vectors are pairwise orthogonal. This means that every pair of vectors will differ in exactly  $M/2$  positions. Hadamard matrices can be constructed recursively [14] as follows: Let  $H_1 = (1)$ . From here, we can construct  $H_{2M}$  from  $H_M$  by using

$$H_{2M} = \begin{pmatrix} H_M & H_M \\ H_M & -H_M \end{pmatrix}. \quad (4)$$

This construction is also known as Sylvester's construction (James Joseph Sylvester, 1867). Using this recursive construction,  $M$  will be a power of 2. In the proof of our reduction, we use the fact that all columns of a recursively constructed Hadamard matrix  $H_M$  contain  $M/2$  1's:

**Lemma 2.** *Let  $H_M$  be an  $M$ -dimensional Hadamard matrix that is constructed as above. In each column of  $H_M$ , except for the first one, the number of 1's in that column is  $M/2$ .*

*Proof.* Since the first row contains only 1's, and the rows of  $H_M$  are pairwise orthogonal, all other rows contain  $M/2$  1's. Due to the recursive construction by Equation (4),  $H_M = H_M^T$ . Therefore, all columns except the first column contain  $M/2$  1's.  $\square$

Feige showed an upper bound on the  $\ell_1$ -norm of an arbitrary subset of  $q$  vectors of a Hadamard matrix (Proposition 2 in [8]):

**Lemma 3 ([8]).** *Consider an arbitrary set of  $q$  distinct vectors from an arbitrary Hadamard matrix  $H_M$ . Then, the  $\ell_1$  norm of their sum is at most  $\sqrt{q}M$ .*

**Theorem 2.** BINARYFH-MEC is NP-hard.

*Proof.* Given a graph  $G = (V, E)$ , an instance to the MAXCUT problem, an  $M|V| \times M|E|$  fragment matrix  $F$  is constructed where  $M$  will be fixed later on. Every vertex  $v \in V$  is represented by a block of  $M$  rows, and every edge  $e \in E$  is represented by a block of  $M$  columns. Arbitrarily orient all edges  $e \in E$  so every edge is now an ordered pair of vertices. For every edge  $e = (u, v)$ , in the block of columns representing  $e$ , set the block representing vertex  $u$  to all 1's, and set the block representing vertex  $v$  to all  $-1$ 's. Set each block representing one of the remaining vertices (not incident to  $e$ ) to the  $M$ -dimensional Hadamard matrix  $H_M$ , constructed recursively as shown in Equation (4).

The fixed haplotype  $h_1 \in \{-1, 1\}^{M|E|}$  is set as follows: for each block of columns representing an edge, the corresponding coordinates of  $h_1$  are set to  $M/2$  1's followed by  $M/2$   $-1$ 's.

We will first discuss the case where an optimum solution to BINARYFH-MEC on the matrix  $F$  and fixed haplotype  $h_1$  never splits the rows belonging to a single vertex block to two different parts of the bipartition. Such a block of rows will be assigned to either the fixed haplotype  $h_1$  or the variable haplotype  $h_2$ . For each block of columns representing an edge, there are 4 options to consider. Each of these options shows the possible values for the error correction of cut and uncut edges. After that we will calculate the contribution of the blocks of Hadamard matrices to the error correction. In the following, when we say that an edge is cut we mean that the block that corresponds to one if its vertices is assigned to one haplotype but the block corresponding to the other vertex to the other haplotype.

**The edge is cut and the block of  $-1$ 's is closest to  $h_2$ .** The block of 1's corresponding to one of the vertices incident to the edge, will contribute  $M^2/2$  to the error correction. If the block of  $-1$ 's is the only block closest to  $h_2$ ,  $h_2$  can be set to all  $-1$ 's and the block of  $-1$ 's will not contribute to the error correction. If other rows, that include blocks with Hadamard matrices are included in  $h_2$ , the first column of  $h_2$  will be set to 1, and the block of  $-1$ 's will contribute  $M$  to the error correction.

**The edge is cut and the block of 1's is closest to  $h_2$ .** The block of  $-1$ 's will cause an error correction of  $M^2/2$ .  $h_2$  can be set to all 1's on the corresponding entries without contributing to the error correction.

**The edge is not cut and both blocks are closest to  $h_2$ .** If both blocks are assigned to  $h_2$ , the first column among the rows assigned to  $h_2$  will always have 1 as a majority bit. In every other column, there will be no unique majority bit, making the haplotype choice unimportant. The two blocks together will contribute  $M^2$  to the total error correction.

**The edge is not cut and both blocks are closest to  $h_1$ .** In the first  $M/2$  columns, the block of 1's will not contribute to the error correction and the block of  $-1$ 's will contribute  $M^2/2$ . In the second  $M/2$  columns, the block of  $-1$ 's will not contribute to the error correction and the block of 1's will contribute  $M^2/2$ . The total contribution to the error correction for the two blocks will be  $M^2$ .

From the cases above it is clear that, when ignoring the Hadamard blocks, a cut edge will contribute to the error correction either  $M^2/2$  or  $M^2/2 + M$ , while an uncut edge will have contribution of  $M^2$ . Note that every Hadamard block will have no error in the first column and an error correction of exactly  $M/2$  in each one of the remaining columns, regardless which haplotype is assigned to, yielding a total error correction of  $M(M-1)/2$ . Note that for each edge there are  $(|V| - 2)$  Hadamard blocks in that column block representing that edge, thus in total we have  $(|V| - 2)|E|$  Hadamard blocks.

Summing up the terms of (i) the  $c$  cut edges each one contributing either  $M^2/2$  or  $M^2/2 + M$ , (ii) the  $|E| - c$  uncut edges each one with contribution of  $M^2$ , and (iii)  $(|V| - 2)|E|$  Hadamard matrices each one contributing  $M(M-1)/2$ , we see that a cut of size  $c$  will have an error correction  $ec$  in the interval

$$\left[ \frac{((|V||E| - c)M^2 - |E|M(|V| - 2))}{2}, \frac{((|V||E| - c)M^2 - M((|V| - 2)|E| + 2c))}{2} \right]$$

It is straightforward to see that the difference in error correction between cuts of size  $c$  and  $c + 1$  is at least  $M^2/2 - (c + 1)M$ . By taking  $M \geq 2|V|^2|E|^2$  and since  $c \leq |E|$ , knowing  $ec$ , it is always possible to distinguish between cuts of size  $c$  and  $c + 1$ .

On the other hand, it could be possible that splitting rows belonging to a block (representing a vertex) could give us a lower error correction as opposed to not splitting a block. If this potential decrease is less than  $M^2/2 - (c + 1)M$ , it is still possible to distinguish between cuts of size  $c$  and  $c + 1$ .

Assume a Hadamard block is split, and  $q$  of its rows are closest to  $h_2$ . By Lemma 3, the  $\ell_1$  norm of these  $q$  rows is at most  $\sqrt{q}M$ . By Lemma 1, the contribution to the error correction of this subset is at least  $(qM - \sqrt{q}M)/2$ . The crucial observation is that a fixed haplotype  $h_1$  is equal to one of the rows of  $H_M$  i.e., there is a row in  $H_M$  that has  $M/2$  1's followed by  $M/2$   $-1$ 's. This fact can be derived from the recursive construction shown in Equation (4). Abusing slightly notation, we say that that row is equal to  $h_1$ . Thus, all rows, except the one row equal to  $h_1$ , will contribute  $M/2$  to the error correction. Summing up, this will be  $(M - q - 1)M/2$ . From here it follows that the contribution of a split Hadamard codematrix to the error correction is at least  $(M^2 - (\sqrt{q} + 1)M)/2$ . Since  $q < M$  (since we assume splitting of the Hadamard blocks), splitting a Hadamard block can decrease the total error correction by at most  $M^{3/2}$ , by Lemma 3. There are  $(|V| - 2)|E|$  Hadamard blocks in  $F$ , so taking  $M \gg O(|V|^2|E|^2)$ , the total decrease is at most  $M^2/2 - (c + 1)M$ .

Lastly, we investigate whether partially cutting edges can close the gap between cuts of size  $c$  and  $c + 1$ . Assume an edge  $e = (u, v)$  is cut partially, and its

corresponding blocks of 1's and  $-1$ 's are distributed over both partitions. Let  $x_u, x_v$  be the fractions of rows corresponding to vertices  $u, v$ , respectively, which are closest to  $h_2$  (and so  $(1 - x_u), (1 - x_v)$  fractions of rows are assigned to  $h_1$ ). In this bipartition, by majority voting, the contribution to the error correction will be at least  $\min(x_u, x_v)M^2$ . For the rows closest to  $h_1$ , the contribution will be  $(2 - x_u - x_v)M^2/2$ . Thus, the total contribution  $tc$  of the edges to the error correction is

$$\begin{aligned}
tc &= \min(x_u, x_v)M^2 + \left(2 - \min(x_u, x_v) - \max(x_u, x_v)\right)\frac{M^2}{2} \\
&= M^2 - \left(\max(x_u, x_v) - \min(x_u, x_v)\right)\frac{M^2}{2} \\
&= M^2 - |x_u - x_v|\frac{M^2}{2} \\
&= M^2 - y_e\frac{M^2}{2}.
\end{aligned}$$

In the above expression,  $y_e$  is the extent to which  $e$  is cut. Summing up over all edges, the total contribution of the blocks of 1's and  $-1$ 's to the error correction is  $|E|M^2 - M^2/2 \sum y_e$ . The term  $\sum y_e$  can be bounded by observing that local search can always change a fractional cut into an integer cut, which is at least as large. Indeed, within a connected set of fractional vertices we can always either increase them all by some amount or decrease them all by some amount such that in each case at least one of the fractional vertices becomes 1 or 0 respectively. Since we change all the fractional values at the same time by the same amount, this does not alter (i.e., worsen) the value of the term  $y_e = |x_u - x_v|$ . Hence,  $\sum y_e \leq c$ , the value of the cut. Thus, the edges contribute at least  $(|E| - c/2)M^2$  to the error correction, allowing a maximum decrease of  $cM$ , which cannot close the gap between cuts of size  $c$  and  $c + 1$ .  $\square$

## 4 NP-Hardness of UngappedFH-MEC

The NP-hardness of UNGAPPEDFH-MEC is implicitly proven by Theorem 2. The proof that follows does not involve the use of Hadamard matrices, and is therefore less technical and more intuitive. Since little is known yet about the approximability of both UNGAPPEDMEC and UNGAPPEDFH-MEC, this more straightforward proof might see potential use in future research.

**Theorem 3.** UNGAPPEDFH-MEC is NP-hard.

*Proof.* Let  $G = (V, E)$  be an arbitrary, connected graph. Let  $F$  be a  $(M + |V|) \times 4|E|$  ungapped fragment matrix. Here,  $M$  will be a sufficiently large number. Every edge  $e \in E$  is represented by a block of four columns of  $F$ , and every vertex  $v \in V$  is represented by one of the rows of  $F$ .  $F$  is constructed as follows: As usual, each edge is oriented arbitrarily. For every edge  $e = (u, v)$ , set its corresponding columns in  $F$  to  $(1 \ 1 \ -1 \ -1)$  in row  $u$ , and to  $(-1 \ -1 \ 1 \ 1)$  in row

$v$ . Set the corresponding block of columns in all other rows corresponding to vertices to  $(1 \ -1 \ 1 \ -1)$ . Now, set its corresponding columns in  $M/2|E|$  of the last  $M$  rows to  $(1 \ 1 \ -1 \ -1)$ , and in  $M/2|E|$  rows to  $(-1 \ -1 \ 1 \ 1)$ . In the remaining blocks of columns, set all values in these rows to  $*$ . Lastly, let the fixed haplotype  $h_1$  be  $-1$  in all positions.

A simple example of the above construction for the simple triangle graph  $C_3$  with vertices  $\{1, 2, 3\}$  and oriented edges  $\{(1, 2), (2, 3), (1, 3)\}$  is given below.

$$F = \left( \begin{array}{cccccccccccc} 1 & 1 & -1 & -1 & 1 & -1 & 1 & -1 & 1 & 1 & -1 & -1 \\ -1 & -1 & 1 & 1 & 1 & 1 & -1 & -1 & 1 & -1 & 1 & -1 \\ 1 & -1 & 1 & -1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 & 1 \\ 1 & 1 & -1 & -1 & * & * & * & * & * & * & * & * \\ -1 & -1 & 1 & 1 & * & * & * & * & * & * & * & * \\ * & * & * & * & 1 & 1 & -1 & -1 & * & * & * & * \\ * & * & * & * & -1 & -1 & 1 & 1 & * & * & * & * \\ * & * & * & * & * & * & * & * & 1 & 1 & -1 & -1 \\ * & * & * & * & * & * & * & * & -1 & -1 & 1 & 1 \end{array} \right) \left. \vphantom{\begin{array}{c} \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \end{array}} \right\} M/(2|E|) \text{ copies} \quad (5)$$

For every set of columns representing an edge, the variable haplotype  $h_2$  will either be  $(1 \ 1 \ -1 \ -1)$  or  $(-1 \ -1 \ 1 \ 1)$ . By doing this, half of the last  $M$  rows will not contribute to the error correction. The other half can simply be assigned to  $h_1$ , contributing  $M$  to the error correction. Setting  $h_2$  to a value other than  $(1 \ 1 \ -1 \ -1)$  or  $(-1 \ -1 \ 1 \ 1)$  will increase the error correction among the last  $M$  rows by at least  $M/2$ , since  $M/2$  rows contribute 0 to the error correction in these configurations. Since there are only  $4|E||V|$  values in the upper  $|V|$  rows, the decrease in error correction of the new configuration can be at most  $4|E||V|$ . Therefore, when setting  $M > 8|E||V|$ , no different configuration will yield an optimum result.

Any row of the first  $|V|$  rows that is assigned to  $h_1$ , will contribute 2 to the error correction for every edge, since the possible values  $(1 \ 1 \ -1 \ -1)$ ,  $(-1 \ -1 \ 1 \ 1)$  and  $(1 \ -1 \ 1 \ -1)$  all have a hamming distance of 2 to  $(-1 \ -1 \ -1 \ -1)$ .

Let  $e = (u, v)$  be an edge that is part of a maximum cut of  $G$ . If  $u$  is assigned to  $h_2$ , then the  $h_2$  can be set to  $(1 \ 1 \ -1 \ -1)$ , causing row  $u$  not to contribute to the error correction in the columns corresponding to  $e$ . If  $v$  is also assigned to the  $h_2$ , it will contribute 4 to the error correction in the columns corresponding to  $e$ . When assigning  $v$  to  $h_1$  instead, the contribution will be 2. It follows that assigning two vertices that are connected by an edge to the same haplotype will contribute 4 to the error correction in the columns corresponding to that edge. Assigning the vertices to different haplotypes will contribute 2 to the error correction. Since  $c = \text{MAXCUT}(G)$  edges can be split up this way, the total error correction for the edges is  $4|E| - 2c$ . Sequences that do not encode an edge will contribute 2 to the error correction in any haplotype. There are  $|E|(|V| - 2)$  of these sequences, yielding an error of  $2|E|(|V| - 2)$ . Combining the errors of the edges, the values that do not encode an edge, and the last  $M$  rows, the total error correction is equal to  $2|E||V| + M - 2c$ . Since  $M$ ,  $|V|$  and  $|E|$  are known for any

instance, the maximum cut can always be determined based on the minimum error correction.  $\square$

## 5 A Simple 2-Approximation for BinaryMEC

For the BINARYMEC polynomial time approximation schemata (PTAS) are known [9, 12]. In [3] a simple and fast 2-approximation algorithm was shown. The algorithm follows the simple observation that given a “conflict-free” matrix  $M$ , then any heterozygous *column* (i.e., not all 1 or not all -1 column) of  $M$  naturally induces a bipartition of the rows of  $M$ <sup>1</sup>. Their algorithm tries to built from any binary matrix  $M$  a conflict free matrix  $M'$  that is induced by a column of  $M$ .

Here we give an even simpler 2-approximation algorithm for BINARYMEC. We show that it is enough to work directly with rows and, in particular, we show that there always exists a pair of rows of  $M$  that when considered as the two haplotypes  $h_1$  and  $h_2$  induce a 2-approximate solution. The algorithm simply iterates over all pairs of rows and picks the pair inducing the smallest error correction.

**Theorem 4.** *For every binary matrix  $F$ , there exists a pair of rows  $r_1, r_2$  such that taking  $r_1, r_2$  as the haplotypes will yield a 2-approximation to BINARYMEC.*

*Proof.* Let  $h_1$  and  $h_2$  be the haplotypes of an optimum solution to BINARYMEC. Let  $R_1$  and  $R_2$  be the partition of rows induced by  $h_1$  and  $h_2$ , respectively. Thus, the cost of the optimum solution is  $\sum_{r \in R_1} d(r, h_1) + \sum_{r \in R_2} d(r, h_2) =: opt$ . For  $i = 1, 2$ , let  $r_i \in R_i$  be the row from  $R_i$  that is closest to  $h_i$  (in the Hamming distance). Then, by triangle inequality, for every row  $r \in R_i$ ,  $d(r, r_i) \leq d(r, h_i) + d(h_i, r_i) \leq 2d(r, h_i)$ .

Let  $R_1^*, R_2^*$  be the partition of the rows induced by considering  $r_1$  and  $r_2$  as the haplotypes. The cost of this partition is minimum among all partitions, and thus at most the cost induced by the partition  $R_1, R_2$ , which is  $\sum_{r \in R_1} d(r, r_1) + \sum_{r \in R_2} d(r, r_2) \leq \sum_{r \in R_1} 2d(r, h_1) + \sum_{r \in R_2} 2d(r, h_2) = 2opt$ .  $\square$

The running time is  $\mathcal{O}(n^2m)$  since we loop over pairs of rows ( $n^2$  pairs of rows in total on instances with  $n$  rows) and for each pair we compute the error correction which takes  $\mathcal{O}(m)$  time. The algorithm of [3] runs in time  $\mathcal{O}(m^2n)$ . Thus, the new algorithm is quicker for inputs where  $n$  is much smaller than  $m$ . Moreover, if the optimum solution to BINARYMEC is  $k$ , then (after collapsing identical rows) the number of rows in the input will also be at most  $k + 2$  – yet the number of columns could still be large. Hence our algorithm might have a role in *parameterized* approaches to solve or approximate BINARYMEC [6, 3].

To see that this is tight, let  $F = \begin{pmatrix} I \\ 1 - I \end{pmatrix}$ , where  $I$  is the identity matrix. We further replace each 0 in  $F$  by  $-1$ . The two optimum haplotypes will be  $\{-1\}^m$

<sup>1</sup> A binary matrix  $M$  is conflict-free if the rows of  $M$  can be bipartitioned into two sets such that the corresponding entries on each set are identical.

and  $\{1\}^m$ . It is easy to see that each row will contribute 1 to the error correction, for a total error correction of  $n$  on instances of  $n$  rows. The approximation algorithm, on the other hand, will pick one row of  $I$  and one of  $1-I$  as haplotypes. The two chosen rows will not contribute to the error correction. For the  $n - 2$  remaining rows, the contribution will be 2 for a total error of  $2(n - 2)$ .

**Acknowledgements.** The last author acknowledges the support of an NWO TOP 2 grant.

## References

1. Paola Alimonti and Viggo Kann. Hardness of approximating problems on cubic graphs. In *Italian Conference on Algorithms and Complexity*, pages 288–298. Springer, 1997.
2. Vikas Bansal and Vineet Bafna. Hapcut: an efficient and accurate algorithm for the haplotype assembly problem. *Bioinformatics*, 24(16):i153–i159, 2008.
3. Paola Bonizzoni, Riccardo Dondi, Gunnar W Klau, Yuri Pirola, Nadia Pisanti, and Simone Zaccaria. On the minimum error correction problem for haplotype assembly in diploid and polyploid genomes. *Journal of Computational Biology*, 23(9):718–736, 2016.
4. Rudi Cilibrasi, Leo Van Iersel, Steven Kelk, and John Tromp. The complexity of the single individual snp haplotyping problem. *Algorithmica*, 49(1):13–36, 2007.
5. International HapMap Consortium et al. A haplotype map of the human genome. *Nature*, 437(7063):1299, 2005.
6. Rodney G Downey and Michael R Fellows. *Fundamentals of parameterized complexity*, volume 201. Springer, 2016.
7. Maryam Etemadi, Mehri Bagherian, Zhi-Zhong Chen, and Lusheng Wang. Better ILP models for haplotype assembly. *BMC Bioinformatics*, 19(1):52, Feb 2018.
8. Uriel Feige. Np-hardness of hypercube 2-segmentation. *arXiv preprint arXiv:1411.0821*, 2014.
9. Yishan Jiao, Jingyi Xu, and Ming Li. On the k-closest substring and k-consensus pattern problems. In *CPM 2004*, pages 130–144. Springer, 2004.
10. Giuseppe Lancia, Vineet Bafna, Sorin Istrail, Ross Lippert, and Russell Schwartz. SNPs problems, complexity, and algorithms. In *9th European Symposium on Algorithms (ESA 2001)*, LNCS 2161, pages 182–193. Springer, 2001.
11. Ross Lippert, Russell Schwartz, Giuseppe Lancia, and Sorin Istrail. Algorithmic strategies for the single nucleotide polymorphism haplotype assembly problem. *Briefings in Bioinformatics*, 3(1):23–31, 2002.
12. Rafail Ostrovsky and Yuval Rabani. Polynomial-time approximation schemes for geometric min-sum median clustering. *Journal of the ACM*, 49(2):139–156, 2002.
13. Christos H Papadimitriou and Mihalis Yannakakis. Optimization, approximation, and complexity classes. *Journal of Computer and System Sciences*, 43(3):425–440, 1991.
14. Kevin T Phelps, Josep Rifa, and Mercè Villanueva. Rank and kernel of binary hadamard codes. *IEEE Transactions on Information Theory*, 51(11):3931–3937, 2005.