# HaploBlocker: Creation of subgroup specific haplotype blocks and libraries

Torsten Pook[1,2*], Martin Schlather[2,3], Gustavo de los Campos[4], Chris Carolin Schoen[5], Henner Simianer[1,2]

**1** Department of Animal Sciences, Animal Breeding and Genetics Group, University of Goettingen, Goettingen, Germany
**2** Center for Integrated Breeding Research, University of Goettingen, Goettingen, Germany
**3** Stochastics and Its Applications Group, University of Mannheim, Mannheim, Germany
**4** Departments of Epidemiology & Biostatistics and Statistics & Probability, Institute for Quantitative Health Science and Engineering, Michigan State University, Michigan, USA
**5** Plant Breeding, TUM School of Life Sciences Weihenstephan, Technical University of Munich, Freising, Germany

¤University of Goettingen, Animal Breeding and Genetics Group, Albrecht-Thaer-Weg 3, 37075 Goettingen, Germany
* torsten.pook@uni-goettingen.de

## Abstract

The concept of haplotype blocks has been shown to be useful in genetics. Fields of application range from the detection of regions under positive selection to statistical methods that make use of dimension reduction. We propose a novel approach ("HaploBlocker") for defining and inferring haplotype blocks that focuses on linkage instead of the commonly used population-wide measures of linkage disequilibrium (LD) which fail to identify segments shared by individuals in only a subset of the population. We define a haplotype block as a sequence of alleles that has a predefined minimum frequency in the population and only haplotypes with a similar sequence of alleles are considered to be carrying that block, effectively screening a dataset for group-wise identity-by-descent (IBD). Different to most other approaches these blocks are not restricted to shared start or end positions, but can overlap or even contain each other. From these haplotype blocks we construct a haplotype library that represents a large proportion of genetic variability of a population with a limited number of blocks. Our method is implemented in the associated R-package HaploBlocker and provides flexibility to not only optimize the structure of the obtained haplotype library for subsequent analyses (e.g., identification of shared segments between different populations), but is also able to handle datasets of different marker density and genetic diversity. By using haplotype blocks instead of SNPs, local epistatic interactions can be naturally modelled and the reduced number of parameter enables a wide variety of new methods for further genomic analyses. We illustrate our methodology with a dataset comprising 501 doubled haploid lines in a European maize landrace genotyped at 501'124 SNPs. With the suggested approach, we identified 2'851 haplotype blocks with an average length of 2'633 SNPs (compared to 27.8 SNPs per block in HaploView) that together represent 94% of the dataset.

## Author summary

Whereas it is quite easy to identify segments of shared DNA between pairs of individuals, the problem becomes far more complex when analyzing a population. Especially for livestock and crop populations under strong selection one can observe long and possibly favourable segments that are segregating at high frequency. We propose here an adaptive and flexible approach to identify such segments ("haplotype blocks"). The main conceptual difference to other approaches is that we allow haplotype blocks to overlap so that patterns shared by a subset of the population can be mapped adequately. Afterwards, we select a set of those haplotype blocks that form a representation of the whole population ("haplotype library"). This haplotype library can be used similar to a SNP-dataset for subsequent genomic approaches with the advantage of a massive reduction of the number of parameters compared to standard haplotyping approaches. Since many breeding goals (e.g. grain yield, milk production) are known to be caused by complex interactions in genomic regions (or even the whole genome) using haplotype blocks instead of single base pairs provides a natural model for local interactions and enables the use of more complex models to incorporate distant interactions between genes, for instance.


## Introduction

Over the years, the concept of haplotype blocks has been shown to be highly useful in the analysis of genomes. Fields of application range from population genetics, e.g. the mapping of positive selection in specific regions of the genome [1, 2], to statistical applications that make use of dimension reduction [3] to tackle the $p \gg n-$ problem [4]. Existing methods define haplotype blocks as a set of adjacent loci, using either a fixed length of markers/variants per block [5] or population-wide linkage disequilibrium (LD) measures [6–9] in the identification process. The methods and software (e.g., HaploView, [10]) available for inferring haplotype blocks have become increasingly sophisticated and efficient. Although those approaches to infer haplotype blocks have been proven to be useful, existing methods share some key limitations [11]. In particular, the use of population-wide measures of LD limits the ability of existing methods to capture cases of high linkage characterized by the presence of long shared segments caused by absence of crossing over (typically within families or close ancestry). To illustrate this, consider the following toy example of four different haplotypes: 11111111, 10101010, 01010101, and 00000000. If all four haplotypes have the same frequency in the dataset, pairwise LD ($r^2$) of neighboring SNPs is zero and LD-based algorithms would not retrieve any structure. However, in this example, knowledge of the first two alleles fully determines the sequence in the segment. In this work we use the term "haplotype" for a known sequence of alleles of a gamete, and not as often done as a short sequence of alleles.

As the starting point of our approach ("HaploBlocker") we assume a set of known haplotypes which can be either statistically determined as accurately phased genotypes, or observed via single gamete genotyping from fully inbred lines or doubled haploids. When the interest is on inferring the longest possible shared segment between haplotypes, a common approach is to identify segments of identity-by-descent (IBD). A tool for the identification of IBD segments is BEAGLE [12], among others. Since IBD is typically calculated between pairs of individuals an additional screening step is necessary to identify haplotypes that are shared by multiple individuals. This can be done with tools like IBD-Groupon [13] for explicitly defined segments. A method to detect IBD segments directly for groups of individuals has been proposed by Moltke et al. [14], but is not applicable to datasets with hundreds of haplotypes due to limitations

of computation times. A further difficulty is that even minor variation tends to break up IBD segments - this can even be caused by actual calling errors (0.2% on later used Affymetrix Axiom Maize Genotyping Array, [15]).

The imputation algorithm of BEAGLE uses a haplotype library given by a haplotype cluster [16]. The haplotype library in BEAGLE is used to initialize a Hidden Markov Model for the imputing step and is only given in a probabilistic way. This means that there are no directly underlying haplotype blocks that could be used for later statistical application.

Our goal is to provide a conceptualization of haplotype blocks that can capture both population-wide LD and subset-specific linkage, and does not suffer from some of the limitations of IBD-based methods. Unlike common definitions that consider haplotype blocks as sets of adjacent loci, we define a haplotype block as a sequence of alleles and only those haplotypes with a similar sequence are assigned to a specific block. By doing this, different blocks can cover the same regions of the genome but differ in the allele variants they represent. We use here the term "allele" for a variant in the genome which can be a single nucleotide polymorphism (SNP) or other variable sites like short indels. Start and end points of the blocks can vary, thus a recombination hot spot appearing in a subgroup of haplotypes does not affect block boundaries in the remaining sets of haplotypes, leading to overall much longer blocks. Subsequently, we construct a haplotype library, which is defined as a set of haplotype blocks representing the entire dataset. This is done by reducing the set of all previously identified blocks to the most relevant ones. The haplotype library is then a mosaic of a limited number of blocks that serves as a condensed representation of the dataset/genome at hand. Depending on the topic of interest, selection criteria for the relevance of each block can be varied appropriately to identify predominantly longer blocks or focus on segments shared between different subspecies. Based on this library one can create a block dataset that contains dummy variables representing the presence/absence of a given block (0 or 1) or, in case of heterozygotes, a quantification of the number of times (0, 1 or 2) a block is present in an individual. This dataset can be used in a similar way as a SNP-dataset with a massive reduction of the number of parameter.

## Materials and methods

The aim of HaploBlocker is to represent genetic variation in a set of haplotypes with a limited number of haplotype blocks as comprehensively as possible. The main idea of our method is to cluster locally similar haplotypes into groups. To this end, we use a graphical representation ("window cluster") in which each node represents a sequence of alleles in a given segment. An edge indicates which and how many haplotypes transition from node to node, allowing an efficient screening of the dataset. To identify common segments in the haplotypes, we first screen short windows of fixed length for shared allele sequences. The size of these analysed segments is increased in an iterative procedure involving the following steps:

- Cluster-building
- Cluster-merging
- Block-identification
- Block-filtering
- Block-extension
- Fixed-coverage (optional)

For a schematic overview of HaploBlocker we refer to Fig 1. Before we elaborate on each step in the following subsections, we give an outline of the legs. The first leg derives the window cluster. This is done by grouping small chunks of adjacent markers and subsequent cluster-building. By this we incorporate the handling of errors and minor deviations in the dataset. The second leg extracts candidates for the haplotype library from the window cluster. We call this step block-identification and use it to generate a large set of candidate blocks. In the third and last leg (block-filtering) this set is reduced to the most relevant haplotype blocks and thereby generating the haplotype library. Since our haplotype blocks are subgroup specific, we have to, in addition to the physical position, derive which haplotypes are included in each block. This in turn makes a direct identification of the most relevant blocks more complicated and enforced us to split this task into two separate, but closely connected legs (block-identification and block-filtering).

Minor steps in our procedure are the cluster-merging and the block-extension. The former reduces the computation time in the subsequent steps, whereas the latter increases the precision of the result. However, neither step has a major impact onto the final haplotype library. Since various parameters are involved in the procedure, their value might be chosen by means of an optimization approach. We discuss the choice of one of the crucial parameter in the subsection on fixed-coverage.

The last three subsections deal with the graphical depiction of the haplotype library, the information loss through the suggested condensation of genomic data, and the datasets under consideration. Our method is available for users by the correspondent R-package HaploBlocker [17, 18]. There, the default settings of the arguments correspond to the thread of the following subsections.

## Cluster-building

In the first step of HaploBlocker we divide the entire sequence of SNPs into short segments with limited number of variants. We use a window size of 20 SNPs as the default setting and group all haplotypes with at most one allele different to the major variant of that group to allow for some calling errors. We start with the most common sequence in the window and include a new group whenever the current sequence does not fit in any of the previous groups. Since we allow 5% of the sequence to be different, this causes actually different haplotypes to be grouped together in this step. In later steps, we will introduce methods to split these grouped haplotypes into different blocks if necessary. The choice of 20 SNPs as a default is rather arbitrary and should not have a major effect as long as it is much smaller than the genuine block sizes one wants to detect - we will present ways to use flexible window sizes in the block-identification-step.

As an example consider a dataset with the allele sequences for a window of 5 SNPs given in Table 1. The two most common sequences form separate groups, whereas CCCCA is different to CCCCC by only one allele and thus assigned to the same group. For graphical reasons in later steps we assign CCCCC to group 3 even though it is the second group created.

**Table 1.** Exemplary dataset of allele sequences and their assignment according to the cluster-building step.

| Frequency | Allele-sequence | Group |
|---|---|---|
| 101 | AAACC | 1 |
| 54 | CCCCC | 3 |
| 40 | CCCCA | 3 |
| 3 | CAACC | 1 |
| 2 | ACAAC | 2 |

## Cluster-merging

Based on the grouping of the previous step we are able to create a window cluster (cf. Fig 2). Here, each node represents a sequence of alleles of a major variant for a single window and the edges indicate how many of the haplotypes of each node transition into which neighboring node. A window cluster can be simplified without losing any relevant information for later steps of the algorithm by three different techniques:

- simple-merge (SM): Combine two nodes if all haplotypes of the first node transition into the same neighboring node and no other haplotypes are in the destination node.
- split-groups (SG): Split a node into two if haplotypes from different nodes transition into the same node and split into the same groups afterwards.
- neglect-nodes (NN): Remove a node from the cluster if it contains a very small (less than 5 in the default setting) number of haplotypes. Removed haplotypes are still considered when calculating transition probabilities between nodes in later steps.

Since the only actual loss of information in this step stems from neglecting nodes, we first alternately apply SM and SG until no further changes occur, before additionally applying NN. We neglect rare nodes, since a block with few haplotypes (in the most extreme case a block with one haplotype over the whole genome) does not reflect much of the population structure and would have little relevance for genomic prediction (GP) or genome wide association studies (GWAS) anyway. We do not increase the minimum number of haplotypes per node depending on the sample size as is done by using a minor allele frequency filter since long shared segments in only a small number of haplotypes could still be relevant.

As an example for the cluster-merging-step consider a dataset with four windows and five different sequences of groups (104x 1111, 54x 3212, 39x 3223, 2x 2111, 1x 3233, Fig 2). In the first step nodes A3 and B2 are merged by SM. Next, node C1 is split up into two nodes via SG. This triggers additional SM (B1-C1a-D1 and C1b-D2). Afterwards, no SM or SG are possible anymore and NN is performed removing A2 and C3. No further merges are possible after this - consider here that even though D3 is the only node following C2 no SM is possible because removed haplotypes are still considered in later transition probabilities and therefore D3 contains one more haplotype than C2.

## Block-identification

In the third step of HaploBlocker we identify the haplotype blocks themselves. Our suggested approach here is to start with each node and edge of the previously obtained window cluster as a starting block and extend these initial blocks based on transition probabilities to adjacent nodes. A starting block using a node spans over the boundaries of that node and contains its haplotypes. Using an edge is a variant of this procedure using the boundaries of the two connected nodes. A block is extended if at least 97.5% of the haplotypes in a block transition into the same node; deviating haplotypes are removed. Haplotypes filtered out in this step can rejoin the block if their sequence of alleles matches that of the major variant of the final haplotype block in at least 99% of the SNPs. To obtain even more candidate haplotype blocks one can consider computing multiple window clusters under different parameter settings (especially concerning window sizes and minimum probabilities). The use of multiple window clusters based on different initial segment lengths is recommended when the genuine length of the final blocks is not known. Note that not all blocks identified here are part of the final

haplotype library and instead are just used as the candidates blocks when selecting the most relevant ones in the block-filtering-step.

To illustrate the method, consider an excerpt of a window cluster given in Fig 3. Nodes 2, 3, 4 represent the sequence of groups 3223 of Fig 2. When considering the second node as a starting block, we cannot extend the block because there are multiple variants with a non-minor share ($> 2.5\%$) of the transitions in both directions. When using the fourth node of the excerpt, the block can be extended until the second and fifth node of the cluster. One ends up with the same block when using the third node or the edges including 39, 39 and 40 haplotypes. In case all included haplotypes transition into the same node in the first window the block could be extended even further. Note that in this step different variants of a particular group of the cluster-building-step can be in different blocks if they transition into different nodes in later steps (e.g CCCCC (54) and CCCCA (39+1) in the first window (cf. Table 3 & Fig 2).

## Block-filtering

After the identification of haplotype blocks, we reduce the set of all haplotype blocks to a haplotype library of the most relevant blocks representing a high proportion of the dataset with a small number of blocks. To set priorities between the importance of the length of the blocks ($l_b$) and the number of haplotypes ($n_b$) in this selection process we first compute a rating $r_b$ for each block $b$:

$$r_b = {l_b}^{w_l} \cdot {n_b}^{w_n}.$$

Here $w_l$ and $w_n$ represent weighting factors with default values $w_l = 1$ and $w_n = 1$. Note that only the ratio between both parameters matters.

We define a position as an entry of the matrix containing haplotype data. Then, we determine the number of positions in the dataset in which each block is the major block, meaning locally highest $r_b$, covering that position and iteratively remove the block with the least number of major positions in the dataset. This procedure is executed until each block has a minimum number of major positions (MNMP) remaining. For our dataset 5'000 was a suitable value for this but without prior information about the dataset we recommend instead setting a target on what share of the dataset is represented by at least one block ("coverage"). We refer to the fixed-coverage-step below for details. In case of our example given in Fig 3 we end up with a block $b_1$ including 94 haplotypes ranging from node 2 to 3 (including 5 SNPs/node) with a rating $r_{b_1} = 94 \cdot 10 = 940$ and a second block $b_2$ ranging from node 2 to 5 with a rating $r_{b_2} = 39 \cdot 25 = 975$. To simplify the example we assume here that no other blocks have been identified. Block 2 has a higher rating and is therefore the major variant in all $39 \cdot 25 = 975$ covered positions. Block 1 is not the major block in those haplotypes included in both blocks resulting in $(94 - 39) \cdot 10 = 550$ positions as the major block. It has to be noted here that the blocks in the final haplotype library can overlap. In case the MNMP is 550 or smaller, overlap occurs in our example and typically can be observed when a short segment is shared in the majority of the population and a smaller subgroup shares a longer segment which includes the short segment.

## Block-extension

The haplotype blocks that have been identified in the previous step are limited to the boundaries of the nodes of the window cluster. Even though haplotypes are split up into different nodes, these nodes can still represent the same sequence of alleles in adjacent markers. This is caused by the fact that nodes can range over multiple windows. Blocks are extended if haplotypes in a block are similar in neighboring segments.

First, haplotype blocks are extended by full windows if all haplotypes are in the same group in the adjacent window. If the haplotypes of a specific block include multiple variants in the adjacent window the block is still extended if at least the following 20 windows are the same for all haplotypes of a block. By doing this we account for possible errors that could for example be caused by translocations or genotyping/phasing errors. The choice of 20 windows is again somehow arbitrary and should be chosen according to the minimum length of a block one is interested in. In any case, all SNPs with variation in a block are identified and reported in the outcome as a possible important information for later analysis.

Second, blocks are extended by single adjacent SNPs following similar rules as the window extension. As a default we do not allow for any differences here since the adjacent window must have some differences based on the block not being extended in the step before. In case of working with a large number of haplotypes and aiming at identifying the exact end of a block, one might consider allowing for minor differences.

## Fixed-coverage (optional)

In the following we will denote the share of the dataset that is represented by a haplotype library as the coverage of the dataset. To control the coverage we propose an adaptive fitting of the MNMP. Especially for different marker densities the choice of the MNMP is relevant to control the minimum size of each block and thereby the resulting obtained coverage. The MNMP is fitted by iteratively increasing/decreasing the MNMP when the coverage is too high/low. We double/halve the value of the MNMP from step to step. When there are two libraries with coverage below and above the target, respectively, the mean of the two MNMP values (one above/below) of the two haplotype libraries with coverage closest to the target is used next. This procedure is done until the MNMP is 1 or the target coverage is reached. For datasets with a high diversity or a low number of haplotypes a high coverage might not be reached since only blocks with a minimum number of haplotypes in it should be considered. Decreasing this minimum would lead to the identification of long blocks with low frequency in the population which might not be informative in later steps of the analysis.

## Graphical representation of haplotype blocks

We suggest a graphical representation of haplotype blocks to show transition rates between blocks in analogy to bifurcation plots [1]. To this end, we first sort the blocks of the haplotype library according to the physical position of the first SNP of the block. In case of identical starting points the shorter block is considered first. Our aim in sorting the haplotypes is to cluster haplotypes according to their similarity around a specific position (default: SNP in the middle of the dataset). The sorting process itself is executed in two alternating steps:

### Step 1: Adding new haplotypes

In the first iteration of this step we select all haplotypes in the most common block that includes the marker we want to align against. In later iterations, we add the haplotypes of that block with the biggest overlap of haplotypes with the previously considered block. In case no block has overlapping haplotypes, we choose the block with the most haplotypes not considered so far.

### Step 2: Sorting new haplotypes

The newly added haplotypes are ordered according to their presence in neighboring blocks. We do this by iteratively comparing the haplotypes of other blocks starting with

the directly adjacent ones. Whenever only some of the currently considered haplotypes are in the block, we split the group of haplotypes into two and proceed with both groups separately. We stop when every group has either exactly one haplotype left or the end of the haplotype library has been reached.

## Assessment of information content of haplotype blocks

The method described above will provide a condensed representation of the genomic data. We next discuss how to quantify the amount of information lost in the process of condensing genotype data to haplotype blocks. A common method to assess the similarity of two multivariate datasets is to use canonical correlation [19]. A limitation of this approach is that it only assesses the similarity of different datasets and therefore does not provide insight on whether one dataset contains information not included in the other one. Recently, de los Campos [20] proposed three methods for estimating the proportion of variance of an omics set (e.g. high-dimensional gene expression data, methylation or markers) that can be explained by regression on another type of omics data. We used a modified version of the second method proposed by de los Campos [20] to estimate the proportion of variance of the full SNP-set genotypes that can be explained by a regression on the blocks of a haplotype library. For the computations in this work the R-packages sommer [21] and minqa [22] were used with overall very similar results. The methodology can be briefly described as follows:

In traditional SNP-based genomic models [23], a phenotype ($y$) is regressed on a SNP-dataset ($X$) using a linear model of the form:

$$y = Xb + \varepsilon,$$

assuming that the markers have only additive effects $b$. Hence, the vector of genomic values $g = Xb$ is a linear combination of the SNP genotypes. In order to estimate the proportion of $g$ explained by the haplotype library we regress the genomic values $g$ onto the haplotype blocks ($Z$):

$$g = Za + \delta.$$

From this perspective, genomic prediction based on haplotype blocks searches for a vector $Za$ that is optimal in some sense. For instance, in ridge regression, such a vector is obtained by minimizing a penalized residual sum of squares. It has to be noted here that $\varepsilon$ is an error term that includes non-genetic effects whereas $\delta$ is an error term resulting from genetic effects that can not be explained by additive effects ($a$) of single blocks. In random effect models the proportion of the variance of $g$ explained by linear regression on the haplotype library can be estimated using either Bayesian or likelihood methods (e.g. REML, [24]). This proportion of variance explained will vary from trait to trait. We estimate the distribution of the proportion of variance of "genomic vectors" (i.e., linear combinations of SNP genotypes) using a Monte Carlo method. The method proceeds as follows:

1. Sample a vector of weights ($b_s$) completely at random (e.g. from a standard Gaussian distribution)
2. Compute "actual" effects by forming the linear combination: $g_s = Xb_s$
3. Estimate the proportion of variance of $g_s$ that can be explained by regression on haplotype blocks
4. Repeat 1.- 3. for a large number of random vectors $b_s$

As the direct estimation of the heritability using REML variance components has recently been shown to be biased (Schreck and Schlather, [25]), we use their proposed estimator. For the traditional estimates using REML estimates as used in [20] we refer to the supplementary - overall results were similar.

In contrast to penalized canonical correlation [19], this method is asymmetric in that it leads to different results by switching the roles of $X$ and $Z$. In that case the "actual" effect is generated by the block dataset ($g_s = Zb_s$) and is then regressed on the SNP-dataset $X$. Since we compute the share of the variance of one dataset explained by the other dataset, the share of variation that is not explained can be interpreted as previously underused information. An example for underused information are local epistatic interactions that can be modeled via a block but are often not fully captured by linear regression. The use of the traditional genomic relationship matrix [26] in a mixed model indirectly simplifies reality by assuming only additive single marker effects. Consider as a toy example a dataset (cf. Table 2) with three markers, six haplotypes and a genetic effect of 1 occurring in the present of the allele sequence AAA. When assuming no environmental effects, phenotypes are equal to genetic values and fitting an ordinary least squares model (OLS) on single markers (using coding $A\hat{=}1$, $C\hat{=}0$) would assign marker 1 the effect of 0.75, marker 2 the effect of 0.5 and marker 3 the effect of 0.5 with an intercept of -1. This in turn leads to small but nevertheless non-zero residuals to the genetic values showing that a model based on single markers can approximate but not fully explain the genetic effect here.

**Table 2.** Estimated genetic values using an OLS model assuming single marker additive effects.

| Allelic variant | Genetic value | Fitted value in linear model |
|---|---|---|
| AAA | 1 | 0.75 |
| ACC | 0 | -0.25 |
| CAA | 0 | 0 |
| AAC | 0 | 0.25 |
| ACA | 0 | 0.25 |
| CAA | 0 | 0 |

## Genotype data used

We applied HaploBlocker to multiple datasets from different livestock and crop populations. In the following we report results obtained with a dataset of doubled haploid (DH) lines of two European maize (*Zea mays*) landraces ($n = 501$ Kemater Landmais Gelb (KE) & $n = 409$ Petkuser Ferdinand Rot (PE)) genotyped with an Affymetrix Axiom Maize Genotyping Array [15] containing 616'201 markers (609'442 SNPs and 6'759 short indels). Markers were filtered for being assigned to the best quality class (PolyHighResolution, [27]) and having a callrate >90%. As we would not expect heterozygous genotypes for DH lines, markers showing an excess of heterozygosity might result from unspecific binding at multiple sites of the genome. Thus markers were also filtered for having <5% heterozygous calls. This resulted in a dataset of 501,124 usable markers. The remaining heterozygous calls of the dataset were set as NA and imputed using BEAGLE 4.0 [28] with modified imputing parameters (buildwindow=50, nsamples=50). As DH lines conceptually are fully homozygous, haplotype phases were directly observed.

# Results and Discussion

Here, we will focus on the results obtained for chromosome 1 (80'200 SNPs) in maize. All tests were also performed on other chromosomes with similar results. Unless otherwise mentioned, we limit ourselves here to a single landrace (KE). Results for PE were similar and haplotype libraries for the joint set were basically a combination of both individual landrace libraries. All results were obtained by means of the associated R-package HaploBlocker [17, 18].

Using the previously described default settings of HaploBlocker we identified 452 blocks which represent 94.2% of the dataset and have an average length of 2'575 SNPs (median: 1'627 SNPs). For the whole genome we identified 2'851 blocks representing 93.9% of the dataset with an average/median length of 2'634/1'280 SNPs. A graphical representation of the block structure for the first 20'000 markers of the set is given in Fig 4. Haplotypes were sorted according to their similarity around SNP 10'000. Since there is only limited linkage between markers further apart, the graphical representation gets fuzzy with increasing distance from the target SNP. If one is interested in a specific region of the dataset, we recommend orientating the block structure according to that region. A position in the dataset can be covered by multiple haplotype blocks (e.g. if a short segment is present in many haplotypes and this group includes a subgroup with a longer shared segment). Because of this there are dependencies in the presence of different blocks that can be addressed similar to linkage disequilibrium between markers. To reduce this overlap one might consider to remove overlapping sections in a long block when there is a shorter block in that region including all of the haplotypes present in the longer block, however the dependency of the presence/absence between blocks in the dataset will of course still be there. On the other hand, some positions of the dataset are not covered by any block and unlike singletons are not easily included in later analysis. When further investigating these segments in our data we could observe that these segments are often a combination of multiple blocks in that region possibly indicating a recent crossing over. Not obtaining full coverage should not cause major concern since the assignment of effects to that kind of rare segments is generally difficult. These rare variants and especially regions with low coverage can be used as candidates for further investigation. The start and end points of a block can be seen as candidates for positions of ancient (or at least non-recent) recombination. For example, four different blocks start at SNPs 8'572 (green), 8'575 (yellow), 8'575 (purple) and 8'601 (brown), indicating a high tendency for variation around that region (cf. Fig 4).

## Effect of change in the MNMP

The MNMP imposes a weighting between the number of blocks and the coverage of the dataset (cf. Table 3). Higher MNMP lead to a stronger filtering of the haplotype blocks and thereby to a haplotype library with lower coverage and decreased number of blocks. For our data using a parametrization of 5'000 for the MNMP worked fine, ensuring a high coverage while obtaining a haplotype library with a relatively low number of blocks. It has to be noted that this highly depends on the marker density and for less dense data other choices might be more suitable requiring the usage of a target coverage. When choosing a higher value for the MNMP the decrease in coverages becomes stronger in relation to the decrease in the number of blocks. For some analysis optimizing the proportion of the dataset not covered by any block might not be the best indicator for the quality of the haplotype library - instead one could consider preserving a certain share in variation of a SNP-dataset.

**Table 3.** Coverage/number of blocks depending on the MNMP in chromosome 1 of maize.

| MNMP | Coverage | Number of Blocks | Average block length (# of SNPs) |
|---|---|---|---|
| 1 | 96.4% | 1'075 | 1'309 |
| 1000 | 95.9% | 748 | 1'777 |
| 5000 | 94.2% | 452 | 2'575 |
| 20000 | 89.6% | 264 | 3'300 |
| 50000 | 81.1% | 150 | 3'940 |

## Haplotypes out of the sample

To assess how well HaploBlocker identifies haplotype block structures that also pertain to haplotype block structures of other datasets we split our data into a training and testing set and compared the share of both datasets represented by blocks created in the training set only. In all cases the coverage in the test set was below that of the training set, but with higher number of haplotypes in the training set the difference gets smaller. In case of 400 haplotypes in the training set the difference in coverage is down to 2.5% (cf. Fig 5) indicating that analyses done in a sufficiently large dataset can be extended to individuals outside of the sample if they have similar genetic origin. Similar results were obtained when setting a target coverage (90%) for the test set and choosing the MNMP accordingly.

## Controlling length and number of haplotypes per block

The window size chosen in the cluster-building-step has a noteworthy influence on the window cluster and hence on the structure of the haplotype library. By using a shorter window size more haplotypes are classified in the same variant of a window ("group") leading to overall shorter nodes with more haplotypes in the window cluster. Since haplotypes in those nodes tend to split up earlier, the set of haplotype blocks contains more and shorter haplotype blocks, leading to a haplotype library with higher coverage and shorter blocks with more haplotypes per block (cf. Table 4).

**Table 4.** Influence of the window size on the haplotype library.

| Window size | Number of Blocks | Average block length (# of SNPs) | Haplotypes per Block | Coverage |
|---|---|---|---|---|
| 5 | 779 | 1'535 | 148.3 | 95.0% |
| 10 | 579 | 2'112 | 121.6 | 94.5% |
| 20 | 452 | 2'575 | 107.5 | 94.2% |
| 50 | 329 | 2'984 | 94.4 | 92.6% |

In the block-filtering-step the weighting between segment length ($w_l$) and number of haplotypes ($w_n$) in each block also influences the structure of the later obtained haplotype library (cf. Table 5). As one would expect, a higher weighting for the length of a block leads to longer blocks being present in less haplotypes. The effect of a lower relative weighting for the number of haplotypes in each block was found to have only a minor effect in our data. A possible reason for this is that even with $w_l = w_n$ the longest blocks previously identified were still selected in the haplotype library. To identify longer blocks in this case one should consider decreasing the minimum transition probability to extend a block in the block-identification-step and thereby allow for the extension of blocks even when there is variation in a block. Even if $w_l$ or $w_n$ is set to zero there is still an implicit weighting on both the length and the number of haplotypes

since each block is identified using the window cluster and has to contain at least a minimum number of major positions. The overall effect of $w_l$ and $w_n$ is higher when more blocks (starting values) are considered in the block-identification-step. This can be achieved by creating multiple window clusters using different window sizes for instance.

**Table 5.** Influence of the weighting of block length ($w_l$) and number of haplotypes ($w_n$) on the haplotype library.

| $w_l$ | $w_n$ | Number of Blocks | Average block length (# of SNPs) | Haplotypes per Block | Coverage |
|---|---|---|---|---|---|
| 1 | 0 | 456 | 2'765 | 90.2 | 94.1% |
| 1 | 0.5 | 442 | 2'768 | 98.2 | 94.1% |
| 1 | 1 | 452 | 2'575 | 107.5 | 94.2% |
| 0.5 | 1 | 500 | 2'250 | 130.2 | 94.5% |
| 0 | 1 | 875 | 1'258 | 174.6 | 95.9% |

## Information content

We investigated the information content in a simulation study where $b_s$ is sampled from a standard Gaussian distribution and a REML approach is used for fitting the model. We found that 96.0% of the variance of the SNP-dataset can be explained by the default haplotype library (cf. Table 6). As one would expect, the share of variance explained is increasing when increasing the number of blocks in the haplotype library. On the other hand, the share of the variance of the haplotype library that can be explained by the SNP-dataset is 95.2%. Even though the number of parameters in the block dataset ($Z$) is much smaller than in the full SNP set ($X$), the share of the variance explained by the respective other dataset is similar. Additionally to the high share of the variation preserved, the haplotype library provides a natural model for the inclusion of locally interacting SNPs (local epistatic) and simplyfies the inclusion of interactions between distant blocks as the number of parameters is heavily reduced [29]. Here, further analyses are needed to find an ideal weighting between information loss, parameter reduction and ways to account for local and distant epistatic interactions.

An alternative for reducing the number of variables for latter applications is to use a subset of SNPs ($X_s$) instead. A SNP-subset of the same size as the number of haplotype blocks on average explains a slightly lower proportion of the variation of the full SNP-dataset (95.1%) than the block dataset. In contrary to the haplotype library, the variation of the SNP-subset is basically fully explained by the full SNP-dataset (99.99%) which is not surprising since $X_s$ is a genuine subset of $X$. Even though a similar share in variation of the SNP-dataset is preserved, the block dataset should be preferred as it is able to incorporate effects that can not be explained by linear effects of single markers (c.f. Table 2).

**Table 6.** Proportion of variance explained between the full SNP-dataset ($X$), a SNP-subset ($X_s$) and the block dataset ($Z$). For comparability the number of parameters of $X_s$ and $Z$ were chosen equally.

| Number of Blocks/SNPs | $X \sim Z$ | $Z \sim X$ | $X \sim X_s$ | $X_s \sim X$ |
|---|---|---|---|---|
| 1'075 | 99.2% | 97.7% | 98.5% | 99.99% |
| 748 | 98.4% | 96.9% | 97.5% | 99.99% |
| 452 | 96.0% | 95.2% | 95.1% | 99.99% |
| 264 | 92.3% | 93.7% | 90.4% | 99.99% |
| 150 | 86.2% | 92.0% | 82.5% | 99.99% |

## Impact of misplaced SNPs

Haplotypes included in the same block can have minor differences. Reasons for such deviations can be diverse, but by comparing the structure in different blocks some explanations become plausible. If the same SNPs tend to have variation in all blocks of that segment, this can be seen as an indication for a misplacement in the used map (e.g. via a translocation or a duplication not represented in the reference map), whereas differences in only one block likely indicate a recent mutation in that specific subgroup.

We further investigated this and replaced 0.2% of all markers by a binomial distributed random variable with $p = 0.5$ so that these markers are not linked to the other markers in the segment. On average, each block contains 4.16 of the replaced markers and 4.14 (99.6%) of those are reported to have variation. In 4.09 (98.4%) of those SNPs both alleles were present with a frequency above 20%. To compare this to the case of a recent mutation we replaced 0.2% of all markers in specific blocks and fixed these markers in all haplotypes outside of the particular block. These changes resulted in an average of 76.3 alleles per SNP that were replaced by a binomial random variable. In the resulting haplotype library 130.6 haplotypes are in blocks with variation in the replaced markers - 76.0 of those were previously replaced alleles (Type I error: 0.33%, Type II error: 14.81%). When only reporting those blocks with at most 20% minor allele frequency only 93.2 haplotypes are in reported blocks with 75.7 being truly replaced by a binomial random variable (Type I error: 0.76%, Type II error: 4.75%). We refer to Fig 6 for a more detailed overview of the influence of the chosen minimum minor allele frequency for a marker of a block to be reported as a block with variation in that marker on Type I and II errors. Most Type II error are caused by overlapping blocks resulting in actual changes in other blocks as well. When excluding those cases, Type II error is reduced to 0.22%. Since all those allele changes were performed based on the previously identified haplotype library and HaploBlocker is robust against smaller deviations, those numbers should be taken with caution but nevertheless show promise to identify translocations and distinguish those from subgroup specific variation.

## Overlapping segments in multiple landraces

When using HaploBlocker on the joint dataset of both landraces (KE & PE), the resulting haplotype library contains essentially the same haplotype blocks that were identified in the two single landrace libraries as shared segments between landraces are often too short leading to a small rating $r_b$ in the block-filtering-step. To specifically identify those sequences present in both landraces, we added the constraint that each block had to be present in at least five haplotypes of both landraces. This results in the identification of 1'618 blocks which are present in both landraces. Those blocks are much shorter (avg. length: 209 SNPs) and represent only 62.7% of the genetic diversity of the dataset. This is not too surprising since the haplotypes of a single landrace are expected to be much more similar than haplotypes from different landraces. Explicitly this is not an indicator for 62.7% of the chromosome of both landraces to be the same. Shared haplotype blocks can be found across the whole chromosome but only some haplotypes of the landraces have those shared segments.

## Comparison with the results of HaploView

Overall, the structure of the haplotype blocks generated with our approach is vastly different from blocks obtained with LD-based approaches such as HaploView [10]. When applying HaploView on default settings [6] to chromosome 1 of the maize data, 2'666 blocks are identified (average length: 27.8 SNPs, median: 20 SNPs) and 4'865 SNPs (6.1%) are not contained in any block. If one would use a similar coding to the

blocks obtained in HaploBlocker and use a separate variable for each variant of a block, one would have to account for 12'550 different variants (excluding singletons). For the whole genome this would result in 16'904 blocks with 79'718 variants. When using a dataset with both landraces (or in general more diversity), LD-based blocks get even smaller (4'367 blocks, 24'511 variants, average length: 17.3 SNPs, median: 9 SNPs, 4'718 SNPs in no block). In comparison, the haplotype library identified in our approach with multiple landraces is, with minor exceptions, an exact combination of both individual landrace haplotype libraries (1'043 blocks, average length: 2'271 SNPs, median: 1'403 SNPs, coverage: 94.2%). Overall, the potential to detect long range associations between markers and to reduce the number of parameters in the dataset is much higher when using haplotype blocks generated by HaploBlocker. It should be noted that HaploView was developed with slightly different objectives in mind [10].

## Influence of marker density

A common feature of conventional approaches to identify haplotype blocks is that with increasing marker density the physical size of blocks is strongly decreasing [30, 31]. To assess this, we executed HaploBlocker on datasets with different marker densities by only including every second/fifth/tenth/fortieth marker in the model. Since the physical length of a window with a fixed number of SNPs is vastly different, we compared the structure of the obtained haplotype library using the adaptive mode in HaploBlocker (multiple window clusters with window sizes 5,10,20,50 and adaptive MNMP to obtain a target coverage of 95%) instead of default settings. As there are far less markers with possible variation, less blocks are needed to obtain the same coverage in the low-density datasets (cf. Table 7). Additionally, the general structure of the haplotype blocks in the library is changing: since the windows for lower density dataset span over a longer part of the genome, the groups in the cluster-building-step tend to be smaller, leading to less frequent nodes in the window cluster. Since the haplotypes in a node are on average more related to each other, the identified blocks tend to be longer and include less haplotypes.

**Table 7.** Structure of the haplotype library under different marker densities using the adaptive mode in HaploBlocker with target coverage of 95%.

| Density | Number of Blocks | Average block length (# of SNPs on full array) | Haplotypes per Block | Used MNMP |
|---|---|---|---|---|
| Every SNP | 534 | 2'317 | 116.4 | 2'813 |
| Every second SNP | 523 | 2'281 | 112.7 | 1'563 |
| Every fifth SNP | 450 | 2'557 | 96.9 | 945 |
| Every tenth SNP | 401 | 2'811 | 90.6 | 758 |
| Every fortieth SNP | 319 | 3'637 | 79.9 | 294 |

In a second step, we manually adapted the window size (50/25/10/5/5) and the MNMP (5000/2500/1000/500/125) according to the marker density of the dataset. When manually adapting the parameters, the number of blocks in the haplotype library is largely independent of the marker density (cf. Table 8). The length of the blocks is decreasing whereas the number of haplotypes per block is increasing. A possible reason for this is that haplotypes in the same node of the window cluster are less similar in the region than when using bigger window sizes. This will lead to shorter haplotype blocks which are carried by more but less related haplotypes. In case of the dataset where we used every fortieth marker we additionally considered a value of 250 for the MNMP since the resulting coverage was a lot higher indicating that less overall variation is present in the dataset, resulting in fewer blocks needed to obtain similar coverage.

**Table 8.** Structure of the haplotype library under different marker densities when adjusting parameters according to data structure.

| Density | Number of Blocks | Coverage | Average block length (# of SNPs on full array) | Haplotypes per Block |
|---|---|---|---|---|
| Every SNP | 454 | 94.9% | 2'547 | 95.6 |
| Every second SNP | 453 | 94.9% | 2'599 | 101.8 |
| Every fifth SNP | 460 | 94.8% | 2'431 | 109.8 |
| Every tenth SNP | 491 | 95.6% | 2'082 | 134.2 |
| Every fortieth SNP (MNMP=125) | 456 | 97.8% | 2'144 | 141.3 |
| Every fortieth SNP (MNMP=250) | 370 | 96.6% | 2'351 | 139.4 |

Haplotype libraries for all marker densities were similar indicating that a much lower marker density than the high-density chip applied here would be sufficient to use HaploBlocker.

## Conclusions and Outlook

HaploBlocker provides a natural technique to model local epistasis and thereby solves some of the general problems of actual markers being correlated but not causal individually [32,33]. This can be seen as one of the factors contributing to the "missing heritability" phenomenon in genetic datasets [34].

Even though results were only presented for maize, methods are not species-dependent and also tested on human and livestock datasets. However, the opportunities for identifying long shared segments will be higher in datasets from populations subjected to a recent history of intensive selection. When using heterozygous datasets the use of triplets or a highly accurate reference panel is recommended to obtain high phasing accuracy.

It should be noted that by using blocks, an assignment of effects to fixed positions (like in a typical GWAS study) is not obtained. A subsequent analysis is needed to identify which section of the significantly trait-associated haplotype block is causal for a trait and/or which parts of that block differ from the other blocks in that region.

A future topic of research is the explicit inclusion of larger structural variants like duplications, insertions or deletions as is done in methods to generate a pangenome [35]. Since blocks in HaploBlocker are of large physical length most structural variants should still be modelled implicitly and an application to sequence data is perfectly possible.

HaploBlocker provides an innovative and flexible approach to screen a dataset for block structure and to reduce the number of parameters for further statistical applications. It produces a library of DNA segments shared by subsets of individuals and a condensed representation of the genotype data. The latter can mitigate some of the problems regarding typical $p \gg n$– settings in genetic datasets [4] and allows for more complex statistical models that include epistasis or even apply deep learning methods with a reduced risk of over-fitting.

HaploBlocker is available as an R-package [17,18], at https://github.com/tpook92/HaploBlocker, including an extensive user manual and example datasets.

# Supporting information

**S1 Table. Influence of the weighting of block length ($w_l$) and number of haplotypes ($w_b$) on the haplotype library when using multiple window cluster.** Window cluster were generated using a target coverage of 95%, window sizes

of 5,10,20,50 and a maximum of one different allele to the major variant for all window sizes and cases (CSV).

**S2 Table. Proportion of variance explained between the full SNP-dataset ($X$), a SNP-subset ($X_s$) and the block dataset ($Z$) using BLUP variance components (CSV).**

**S1 File. Chromosome 1 Maize** SNP-Dataset containing 80'200 markers and 910 DH-lines(501 Kemater Landmais Gelb & 409 Petkuser Ferdinand Rot) (CSV).

## Acknowledgments

The authors thank the German Federal Ministry of Education and Research (BMBF) for the funding of our project (MAZE - "Accessing the genomic and functional diversity of maize to improve quantitative traits"; Funding ID: 031B0195). Gustavo de los Campos was supported by a Mercator-fellowship of the German Research Foundation (DFG) within the Research Training Group 1644, "Scaling problems in statistics". We acknowledge support by the Open Access Publication Funds of the Goettingen University. We further thank Nicholas Schreck for useful comments on the manuscript and his help regarding unbiased heritability estimation.

## References

1. Sabeti PC, Reich DE, Higgins JM, Levine HZP, Richter DJ, Schaffner SF, et al. Detecting recent positive selection in the human genome from haplotype structure. Nature. 2002;419(6909):832–837.
2. Sabeti PC, Varilly P, Fry B, Lohmueller J, Hostetter E, Cotsapas C, et al. Genome-wide detection and characterization of positive selection in human populations. Nature. 2007;449(7164):913–918.
3. Pattaro C, Ruczinski I, Fallin DM, Parmigiani G. Haplotype block partitioning as a tool for dimensionality reduction in SNP association studies. BMC genomics. 2008;9(1):405.
4. Fan J, Han F, Liu H. Challenges of big data analysis. National science review. 2014;1(2):293–314.
5. Meuwissen THE, Odegard J, Andersen-Ranberg I, Grindflek E. On the distance of genetic relationships and the accuracy of genomic prediction in pig breeding. Genetics Selection Evolution. 2014;46(1):49.
6. Gabriel SB, Schaffner SF, Nguyen H, Moore JM, Roy J, Blumenstiel B, et al. The structure of haplotype blocks in the human genome. science. 2002;296(5576):2225–2229.
7. Daly MJ, Rioux JD, Schaffner SF, Hudson TJ, Lander ES. High-resolution haplotype structure in the human genome. Nature genetics. 2001;29(2):229–232.
8. Taliun D, Gamper J, Pattaro C. Efficient haplotype block recognition of very long and dense genetic sequences. BMC bioinformatics. 2014;15(1):10.
9. Kim SA, Cho CS, Kim SR, Bull SB, Yoo YJ. A new haplotype block detection method for dense genome sequencing data based on interval graph modeling of clusters of highly correlated SNPs. Bioinformatics. 2017;.

10. Barrett JC, Fry B, Maller J, Daly MJ. Haploview: analysis and visualization of LD and haplotype maps. Bioinformatics. 2005;21(2):263–265.

11. Slatkin M. Linkage disequilibrium—understanding the evolutionary past and mapping the medical future. Nature Reviews Genetics. 2008;9(6):477–485.

12. Browning BL, Browning SR. Improving the accuracy and efficiency of identity-by-descent detection in population data. Genetics. 2013;194(2):459–471.

13. He D. IBD-Groupon: an efficient method for detecting group-wise identity-by-descent regions simultaneously in multiple individuals based on pairwise IBD relationships. Bioinformatics. 2013;29(13):i162–i170.

14. Moltke I, Albrechtsen A, vO Hansen T, Nielsen FC, Nielsen R. A method for detecting IBD regions simultaneously in multiple individuals—with applications to disease genetics. Genome research. 2011;21(7):1168–1180.

15. Unterseer S, Bauer E, Haberer G, Seidel M, Knaak C, Ouzunova M, et al. A powerful tool for genome analysis in maize: development and evaluation of the high density 600 k SNP genotyping array. BMC genomics. 2014;15(1):823.

16. Browning BL, Browning SR. Efficient multilocus association testing for whole genome association studies using localized haplotype clustering. Genetic epidemiology. 2007;31(5):365–375.

17. R Core Team. R: A Language and Environment for Statistical Computing; 2017. Available from: `https://www.R-project.org/`.

18. Pook T, Schlather M. HaploBlocker: An R package for the Creation of Haplotype Libraries for DHs and Highly Inbreed Lines; 2018. Available from: `https://github.com/tpook92/HaploBlocker`.

19. Witten DM, Tibshirani R, Hastie T. A penalized matrix decomposition, with applications to sparse principal components and canonical correlation analysis. Biostatistics. 2009;10(3):515–534.

20. de los Campos G. What Fraction of the Information Contained in an Omic Set Can Be Explained by Other Omics? paper # 22987. Plant and Animal Genome Conference, San Diego, California. 2017;.

21. Covarrubias-Pazaran G. Genome-assisted prediction of quantitative traits using the R package sommer. PloS one. 2016;11(6):e0156744.

22. Powell MJD. The BOBYQA algorithm for bound constrained optimization without derivatives. Cambridge NA Report NA2009/06, University of Cambridge, Cambridge. 2009; p. 26–46.

23. Meuwissen THE, Hayes BJ, Goddard ME. Prediction of total genetic value using genome-wide dense marker maps. Genetics. 2001;157(4):1819–1829.

24. Patterson HD, Thompson R. Recovery of inter-block information when block sizes are unequal. Biometrika. 1971;58(3):545–554.

25. Schreck N, Schlather M. From Estimation to Prediction of Genomic Variances: Allowing for Linkage Disequilibrium and Unbiasedness. bioRxiv. 2018;doi:10.1101/282343.

26. VanRaden PM. Efficient methods to compute genomic predictions. Journal of dairy science. 2008;91(11):4414–4423.

27. Pirani A, Gao H, Bellon L, Webster TA. Best practices for genotyping analysis of plant and animal genomes with Affymetrix® Axiom® arrays: 2013:P0997; 2013.

28. Browning BL, Browning SR. Genotype imputation with millions of reference samples. The American Journal of Human Genetics. 2016;98(1):116–126.

29. Martini JWR, Gao N, Cardoso DF, Wimmer V, Erbe M, Cantet RJC, et al. Genomic prediction with epistasis models: on the marker-coding-dependent performance of the extended GBLUP and properties of the categorical epistasis model (CE). BMC bioinformatics. 2017;18(1):3.

30. Sun X, Stephens JC, Zhao H. The impact of sample size and marker selection on the study of haplotype structures. Human genomics. 2004;1(3):179.

31. Kim SA, Yoo YJ. Effects of Single Nucleotide Polymorphism Marker Density on Haplotype Block Partition. Genomics & informatics. 2016;14(4):196–204.

32. He S, Reif JC, Korzun V, Bothe R, Ebmeyer E, Jiang Y. Genome-wide mapping and prediction suggests presence of local epistasis in a vast elite winter wheat populations adapted to Central Europe. Theoretical and Applied Genetics. 2017;130(4):635–647.

33. Akdemir D, Jannink JL, Isidro-Sánchez J. Locally epistatic models for genome-wide prediction and association by importance sampling. Genetics Selection Evolution. 2017;49(1):74.

34. Manolio TA, Collins FS, Cox NJ, Goldstein DB, Hindorff LA, Hunter DJ, et al. Finding the missing heritability of complex diseases. Nature. 2009;461(7265):747–753.

35. Eggertsson HP, Jonsson H, Kristmundsdottir S, Hjartarson E, Kehr B, Masson G, et al. Graphtyper enables population-scale genotyping using pangenome graphs. Nature genetics. 2017;49(11):1654.

**Fig 1. Schematic overview of the steps of the HaploBlocker method.** Steps include (1) Cluster-building: Classify local haplotype variants in short windows into groups. (2) Cluster-merging: Simplify window cluster by merging and neglecting nodes. (3) Block-identification: Identify blocks based on transition probabilities between nodes. (4) Block-filtering: Creation of a haplotype library by reducing the set of blocks to the most relevant ones for the later application. (5) Block-extension: Extend blocks by single windows and SNPs.

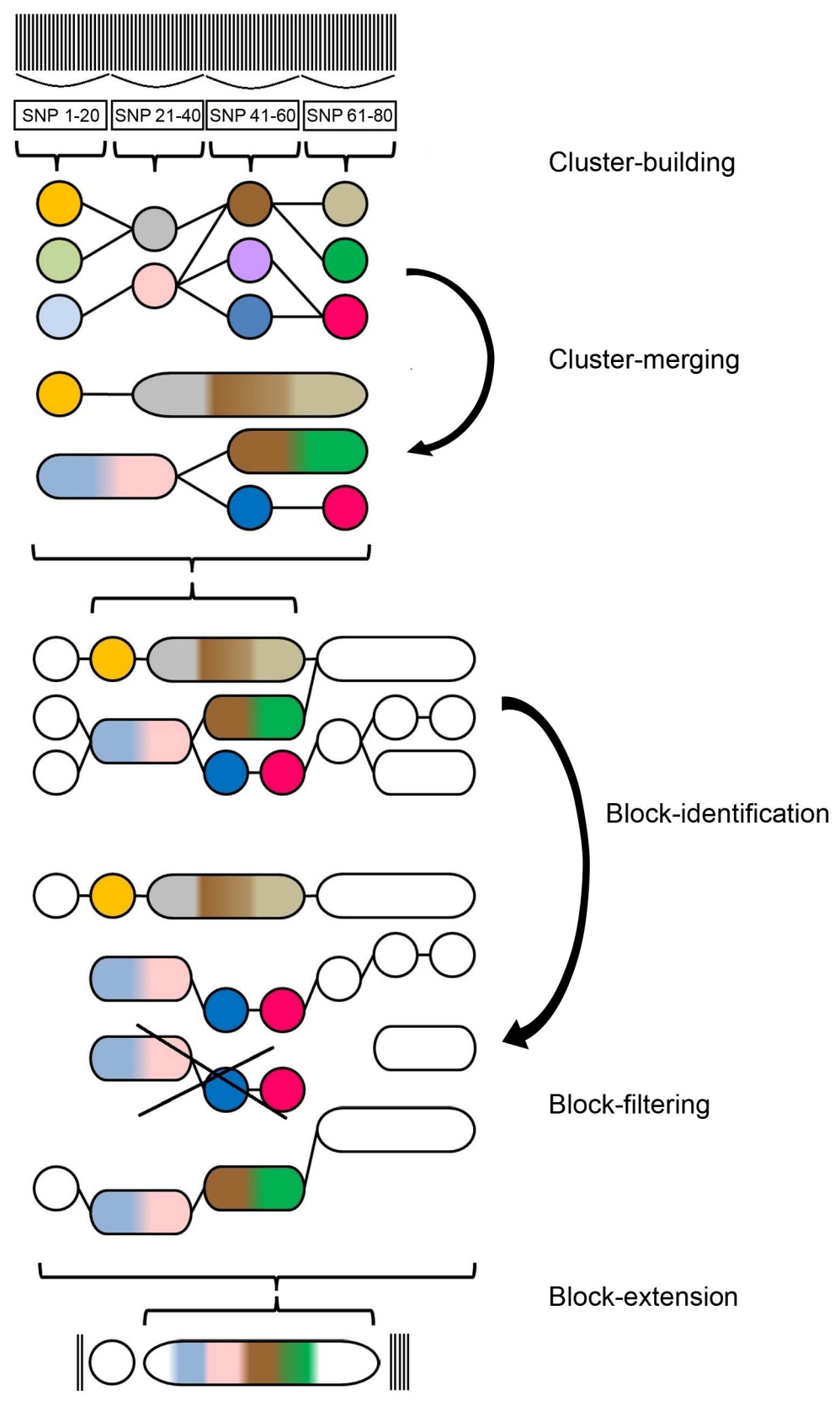
**Fig 2. Cluster-merging-step:** Development of the window cluster in the Cluster-merging-step after each application of SM, SG, NN.

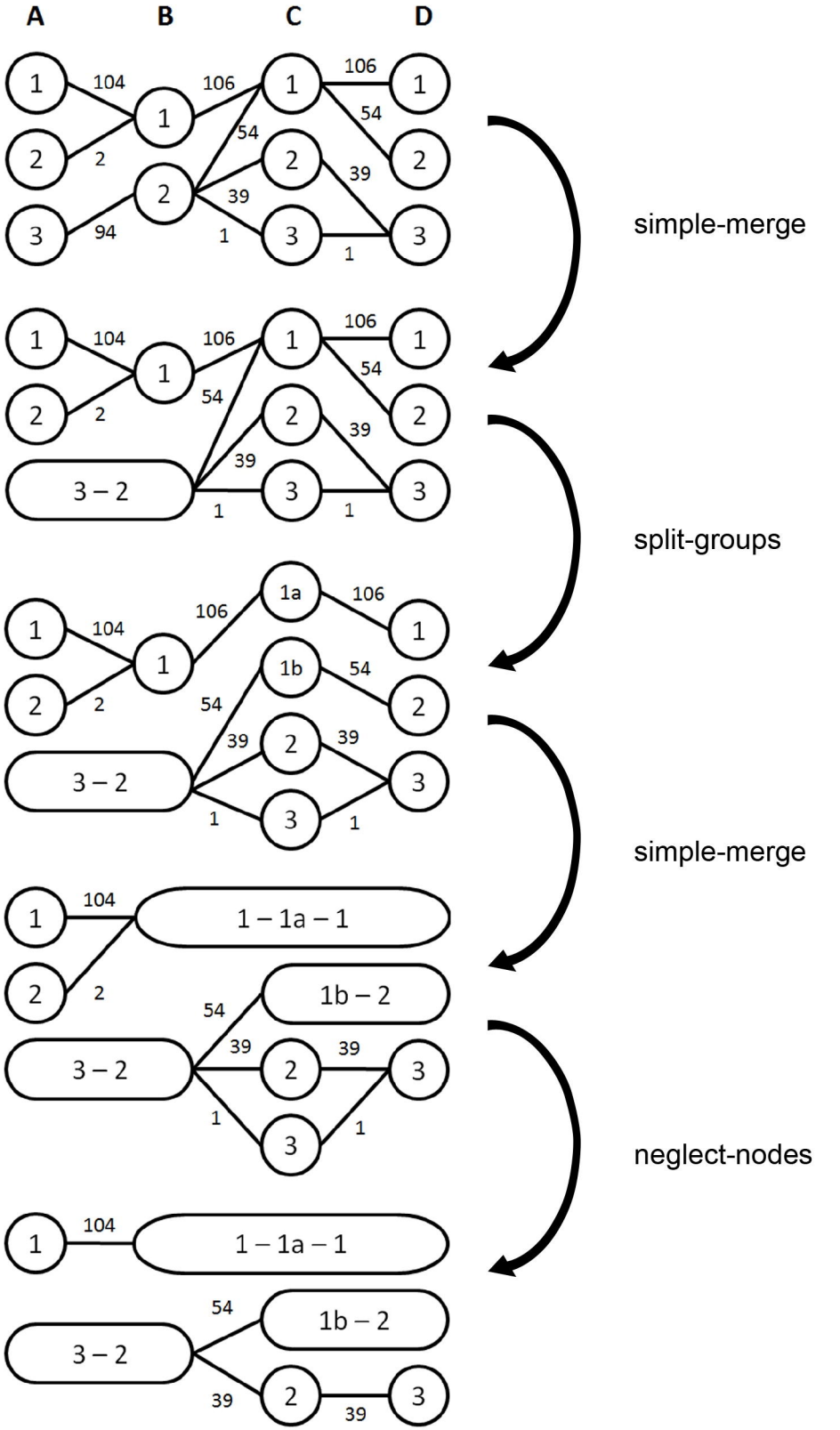
**Fig 3. Excerpt of a window cluster.** This included all edges (transitions) from the nodes of one of the common paths in the example dataset.

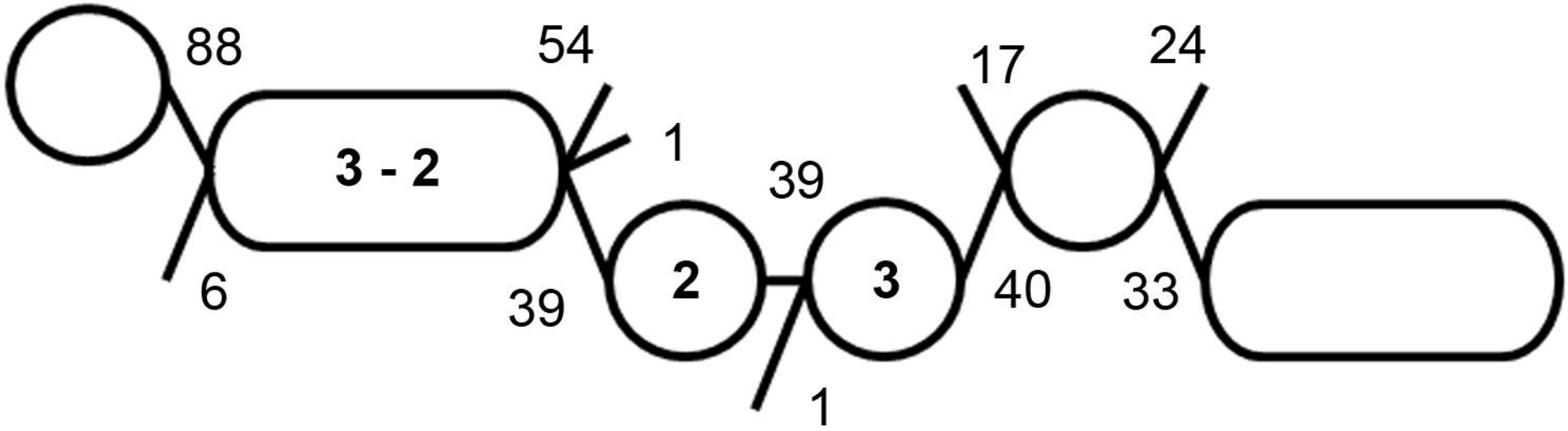
**Fig 4. Graphical representation of the block structure for the first 20'000 SNPs of chromosome 1 in maize.** Haplotypes are sorted for similarity in SNP 10'000. In that region block structures are most visible and transitions between blocks can be tracked easily.

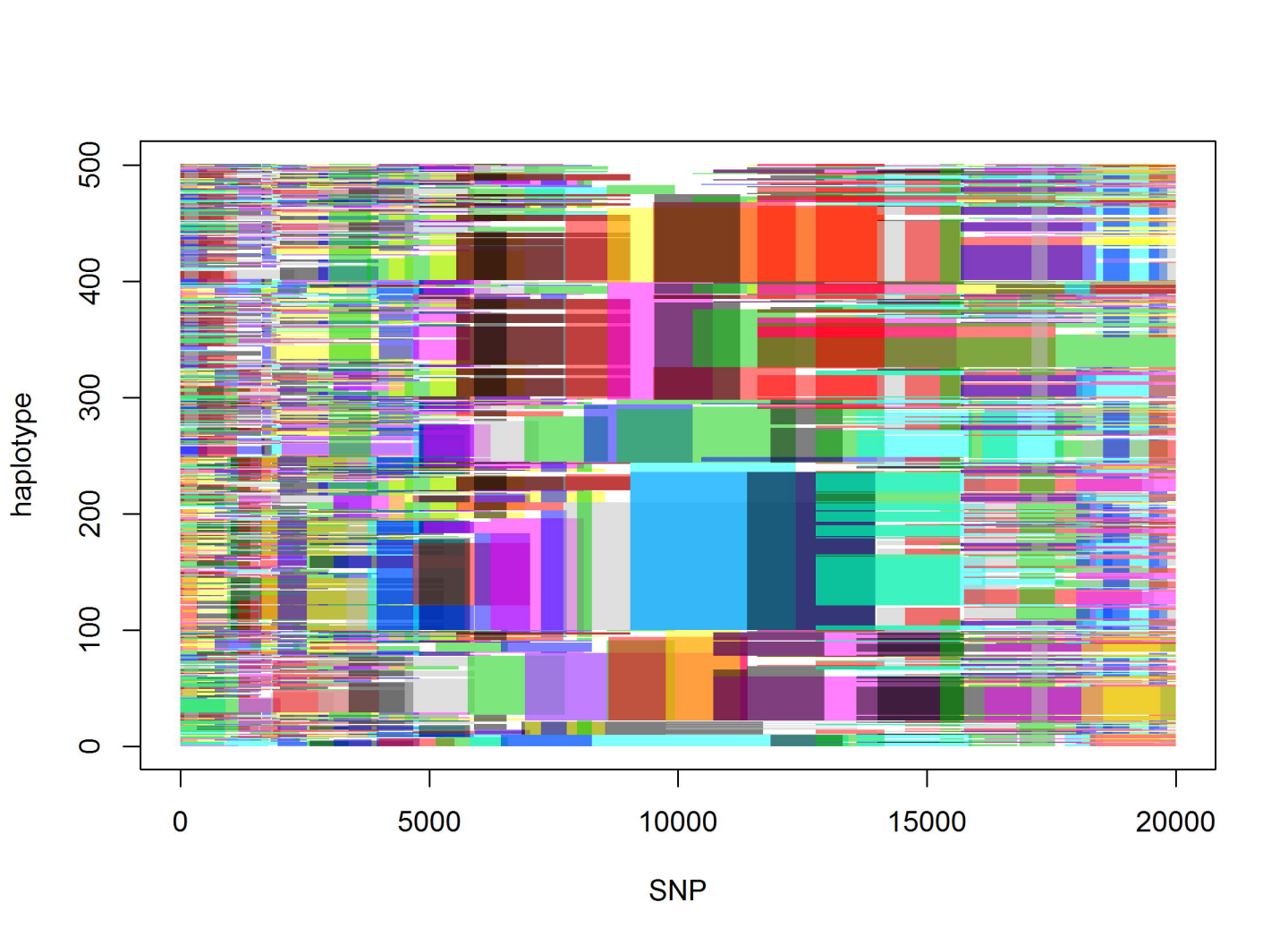
**Fig 5. Proportion of the dataset represented by the haplotype library (coverage) of the training and test set in regard to size of the training set.**

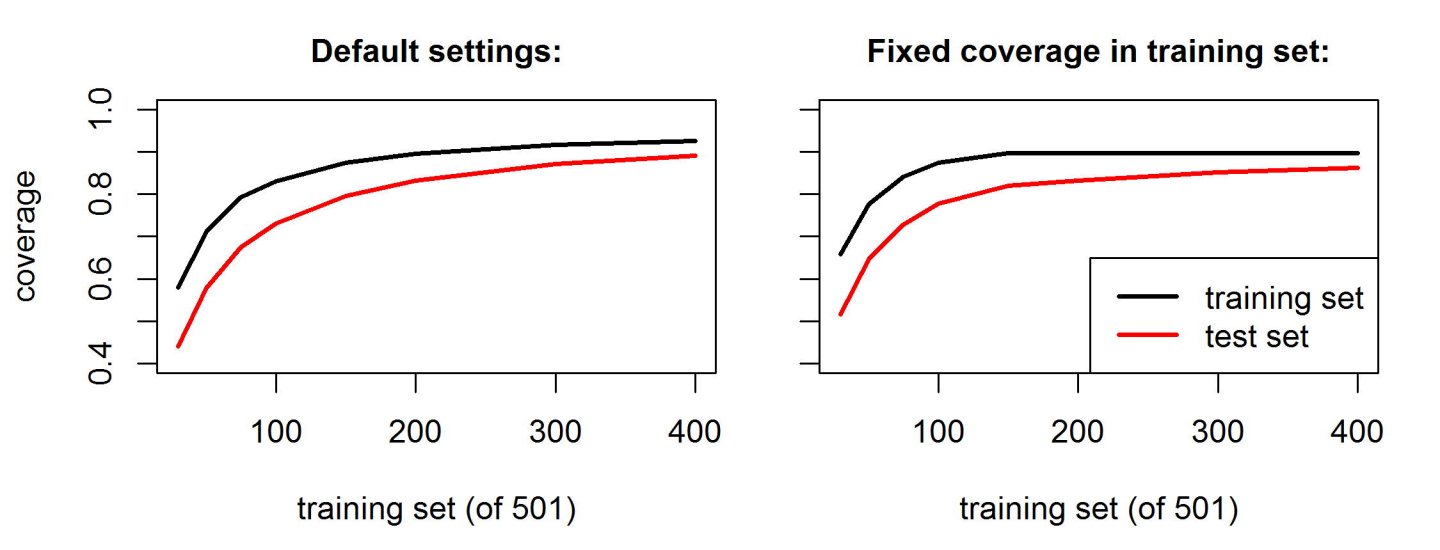
**Fig 6. Influence of the minimum minor allele frequency set to identify subgroups with variable markers.**

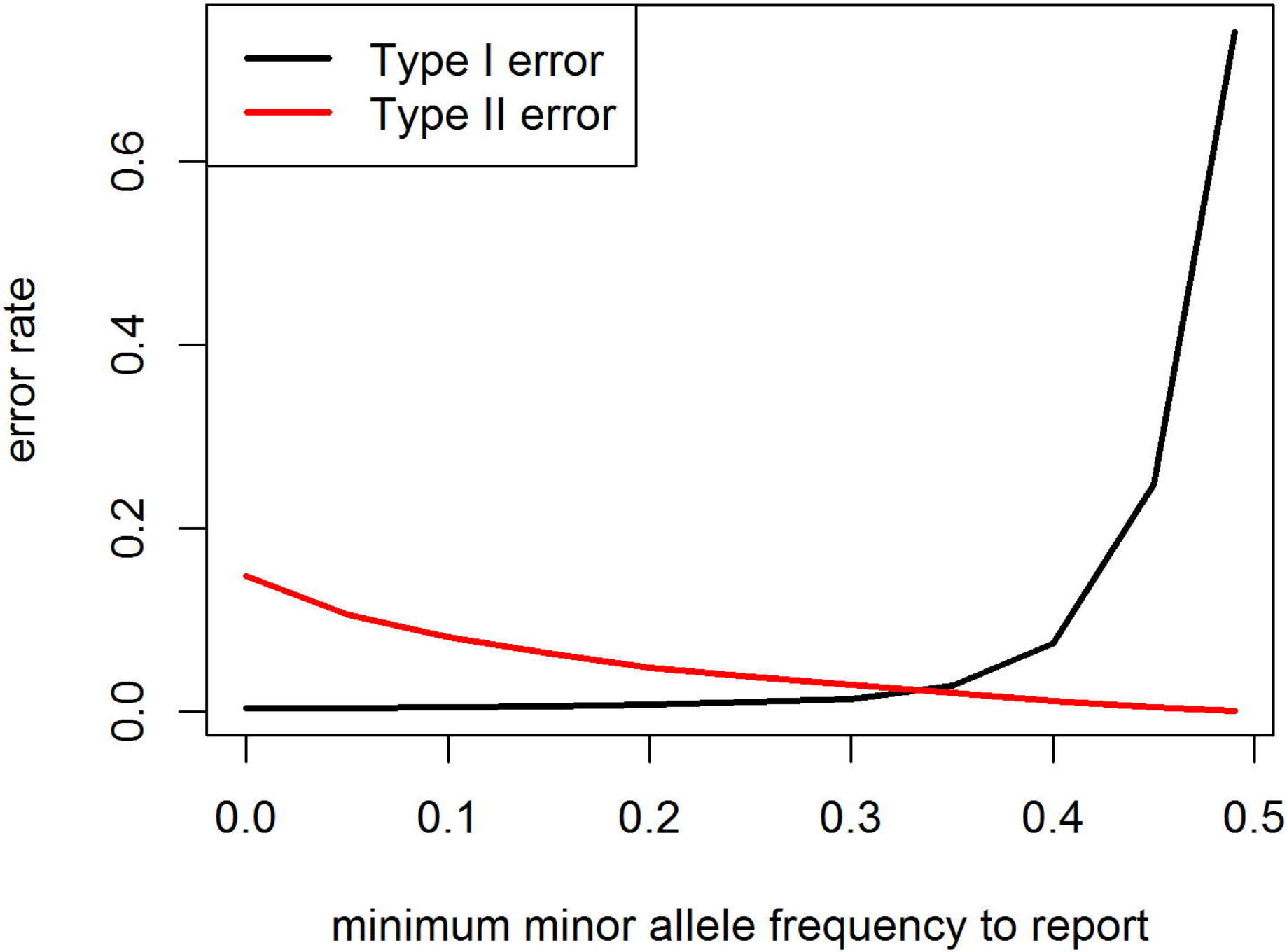

SNP 1-20
SNP 21-40
SNP 41-60
SNP 61-80
Cluster-building
Cluster-merging
Block-identification
Block-filtering
Block-extension


A
B
C
D
1
2
3
104
2
94
106
54
39
1
1
2
3
106
54
39
1
1
2
3
simple-merge
1
2
3 – 2
split-groups
1a
1b
simple-merge
1 – 1a – 1
1b – 2
neglect-nodes


88
54
17
24
3 - 2
1
39
6
39
2
3
40
33
1


haplotype
0
100
200
300
400
500
0
5000
10000
15000
20000
SNP


Default settings:
Fixed coverage in training set:
coverage
0.4
0.6
0.8
1.0
100
200
300
400
training set (of 501)
training set
test set


Type I error
Type II error
error rate
0.0
0.2
0.4
0.6
0.0
0.1
0.2
0.3
0.4
0.5
minimum minor allele frequency to report