GMCheck: Bayesian error checking for pedigree genotypes and phenotypes

Alun Thomas

Department of Medical Informatics and Center for High Performance Computing, University of Utah, 391 Chipeta Way Suite D, Salt Lake City, UT 84108, USA

Received on November 19, 2004; revised on April 29, 2005; accepted on May 3, 2005

Advance Access publication May 6, 2005

ABSTRACT

Summary: GMCheck uses graphical modeling to find the posterior probabilities of data errors given genotypes or phenotypes in a specified pedigree structure.

Availability: The Java classes and Javadoc pages for GMCheck can be obtained from bioinformatics.med.utah.edu/~alun, which also has information on use, parameter settings and file formats.

Contact: alun@genepi.med.utah.edu

Lange et al. (1988) and O’Connell and Weeks (1988) have detailed the importance of precise error checking for genetic marker data in pedigrees used for linkage analysis, and produced programs, MENDEL and PedCheck, for finding errors by detecting implied allele segregations that violate Mendelian rules of inheritance. Unlike previous programs which use ad hoc heuristics and only part of the data available at a locus, both these programs calculate the posterior probabilities for genotypes given the entire pedigree structure and observed data. Each, however, has implementational drawbacks: MENDEL cannot compute posterior probabilities for loci with large numbers of alleles and approximates these with conditional posterior numbers of alleles and approximates these with conditional posteri-
or; PedCheck is unable to deal with looped pedigrees efficiently (O’Connell and Weeks, 1988). Loki (Heath, 1998) is an efficient program that handles arbitrarily structured pedigrees but exits at the discovery of the first error. Merlin (Abecasis et al., 2001) uses exact computation to find the posterior probabilities of genotypes given multilocus data, but only for small pedigrees. SimWalk2 (Sobel et al., 2002) addresses the same problem using Markov chain Monte Carlo integration which takes considerable time.

GMCheck is a new program that frames detecting and reporting errors from single locus genetic data as a Bayesian network and uses the methods of Lauritzen and Spiegelhalter (1988) for efficient exact calculation. The novel aspect is the introduction of an explicit indicator variable to represent the occurrence or the non-occurrence of an error for each datum.

The probability of a genotype configuration is represented as a product of simple factors which defines a Markov random field and corresponding Markov graph. This graph is triangulated using the heuristic greedy algorithm, and a sequence of maximal cliques that correspond to parent–offspring triplets. If there are alleles at a locus,

and Stewart (1971) and Cannings et al. (1978). We can also reverse the peeling order and perform a distribute evidence step which computes the posterior marginal distributions for each clique and hence for each variable. Replacing the summation operation by maximization gives a dynamic program that will find a state of maximal posterior probability (Dawid, 1992).

Let \( x_i \) be the true unobserved genotype of the \( i \)-th individual, let \( y_i \) be the corresponding observation. For each observation we have a binary variable \( e_i \) indicating the presence or absence of an error. Let \( x = \{x_i\}, e = \{e_i\} \). The factors of the Markov random field are

- \( \pi(x_i) \): The population frequency for genotype \( x_i \).
- \( \tau(x_i|x_f,x_m) \): The probability that a child inherits genotype \( x_i \) from parents with genotypes \( x_f \) and \( x_m \) defined by the usual Mendelian rules.
- \( \rho(y_i|x_i) \): The probability that an individual with genotype \( x_i \) has observation \( y_i \).
- \( \rho^*(y_i|x_i,e_i) \): The observational error model which is \( \rho(y_i|x_i) \) if \( e_i = 0 \), but constant if \( e_i = 1 \). Thus, if \( e_i = 0 \) there is no error and we use the usual penetrance function, whereas if \( e_i = 1 \) an error has occurred and the \( i \)-th observation is deemed uninformative.
- \( \epsilon(e_i) \): The prior probability of an observational error.

Our graphical model is then defined by the product

\[
f(x,e) = \prod_{i \in \mathcal{P}} \pi(x_i) \prod_{i \in \mathcal{F}} \tau(x_i|x_f,x_m) \prod_i \rho^*(y_i|x_i,e_i) \epsilon(e_i),
\]

where \( F \) is the set of founders of the pedigree. We first compute \( \sum_i \sum_f f(x,e) \) and \( \sum_i f(x,e) = \{0,0,\ldots\} \), whose ratio gives the probability that the data are error free. If this is less than a threshold value, we then find \( \hat{x} \) and \( \hat{\epsilon} \) such that \( f(\hat{x},\hat{\epsilon}) = \max_x \max_{\epsilon} f(x,e) \) to obtain a configuration of variables of maximal posterior probability. The values of \( \hat{\epsilon} \) indicate the combination of genotypes most probably in error, and computing \( \sum_i f(\hat{x},\hat{\epsilon}) \) gives us the posterior probability of this. Finally, we compute posterior marginals and report genotypes with high posterior error probability and the most probable correct states.

The time and storage needed for these computations are determined by the product of the number of states of variables in each clique of the Markov graph. In the case of zero loop pedigrees these cliques correspond to parent–offspring triplets.
each genotype has \(|a(a + 1)|/2 \) states. A generic graphical modeling program would, therefore, need time and storage of order \(o(a^2)\). However, for each parental-genotype pair there are at most four possible offspring genotypes. We take advantage of this to reduce the computational requirements to order \(o(a^4)\). GMCheck will also compute probabilities for looped pedigrees but will behave as a generic graphical modeling program when dealing with cliques that are not parent-offspring triplets.

To further speed up computations we consider only the alleles observed in the pedigree being checked, even though other alleles may occur in other pedigrees. Although this parsimonious approach can alter the posterior probabilities slightly, it greatly increases tractability.

As an illustration, the figure shows a fictitious pedigree of seven linked nuclear families that is problematic for heuristic methods as all seven need to be considered jointly in order to detect the error. It is also problematic for programs that cannot handle loops. GMCheck gave the following output in 4.05 s on the author’s laptop computer.

**Pedigree (1)** locus (1)

\[
P(\text{at least one error}) = 1.0
\]

**Most probable individual(s) in error:**

- Individual (28) 1 28, with probability 0.8286

**Individuals with high error probability:**

- **Observation** = 1 14
- **P(Error)** = 0.077
- **Probable genotype:**
  - \(P(2,3) = 0.922\)
  - **Individual** (28) 1 28
  - **Observation** = 2 4
  - **P(Error)** = 0.842
  - **Probable genotype:**
    - \(P(1,2) = 0.364\)
    - \(P(1,1) = 0.258\)
    - \(P(2,4) = 0.157\)
    - \(P(2,2) = 0.105\)

This indicates that there must be at least one error and that it is most probably for individual 28. Although it could be explained by an error for individual 14 this is far less probable. The posterior probabilities of likely genotypes are also given and since for 28 none of these is by far larger than the others, this is a case where the observation might be deleted rather than corrected.

The program has also been used on larger datasets, for example, a set of 8839 individuals in 132 pedigrees of up to 326 individuals, including a looped pedigree of 61 individuals, typed at 25 biallelic loci, was checked in 185 s.

The current default parameter values for GMCheck are set to report cases where the overall probability of error-free data is \(<0.05\), and where the marginal probability of a particular genotype error is \(>0.05\). The prior probability of an error is set at 0.01. All these values can be changed using command line arguments. These defaults were chosen informally based on experience with a small but diverse collection of datasets. For a more complete discussion of this issue see Douglas et al. (2002).

This approach provides exact computation of posterior error probabilities using all available genotyping data in pedigrees of arbitrary size and complexity. It is efficient, and for zero looped pedigrees its requirements grow linearly with the size of the pedigree. It also handles looped pedigrees although the resources required to do so can grow quickly if there are several intersecting loops. The output from the program is informative and should enable straightforward correction or deletion of unreliable data. Thus, heuristics, partial analyses or simulation methods for checking single locus data should be needed for only the most complex pedigrees.

If the genotypes of a particular individual over a broad range of loci are indicated to be in error this may point to problems in the pedigree data or a possible sample mix up. A full multilocus analysis would be a more reliable way of detecting this, however, and also of addressing the problem of individually uninformative loci, such as single nucleotide polymorphisms in linkage disequilibrium. This is still an open question for large pedigrees.

**ACKNOWLEDGEMENT**

This work was supported by NIH NIGMS grant R21 GM070710 to A.T.

**REFERENCES**


