PedCheck: A Program for Identification of Genotype Incompatibilities in Linkage Analysis

Jeffrey R. O'Connell[1] and Daniel E. Weeks[1,2]

[1]Department of Human Genetics, University of Pittsburgh, Pittsburgh; and [2]The Wellcome Trust Centre for Human Genetics, University of Oxford, Oxford

Summary

Prior to performance of linkage analysis, elimination of all Mendelian inconsistencies in the pedigree data is essential. Often, identification of erroneous genotypes by visual inspection can be very difficult and time consuming. In fact, sometimes the errors are not recognized until the stage of running linkage-analysis software. The effort then required to find the erroneous genotypes and to cross-reference pedigree and marker data that may have been recoded and renumbered can be not only tedious but also quite daunting, in the case of very large pedigrees. We have implemented four error-checking algorithms in a new computer program, PedCheck, which will assist researchers in identifying all Mendelian inconsistencies in pedigree data and will provide them with useful and detailed diagnostic information to help resolve the errors. Our program, which uses many of the algorithms implemented in VITESSE, handles large data sets quickly and efficiently, accepts a variety of input formats, and offers various error-checking algorithms that match the subtlety of the pedigree error. These algorithms range from simple parent-offspring–compatibility checks to a single-locus likelihood-based statistic that identifies and ranks the individuals most likely to be in error. We use various real data sets to illustrate the power and effectiveness of our program.

Introduction

When marker genotype data are generated, one often must go through a tedious iterative process of correcting errors that are incompatible with Mendelian inheritance. To this end, if visual inspection is not sufficient to identify the problem, the researcher often runs a diagnostic program, such as UNKNOWN of the LINKAGE package (Lathrop and Lalouel 1984; Lathrop et al. 1984, 1986) or the Genetic Analysis System (GAS), which use genotype-elimination techniques (Lange and Goradia 1987; Lange and Weeks 1989), to try to identify the source of the error. However, as Stringham and Boehnke (1996) recently pointed out, these programs are not helpful enough. For example, in the interest of speed, only a partial genotype-elimination algorithm was implemented in GAS; so, it may fail to identify all Mendelian inconsistencies. Likewise, not only does UNKNOWN sometimes fail to provide helpful diagnostic information (see discussion in Stringham and Boehnke 1996), but it also can take a painfully long time to run as the number of alleles at the marker under consideration becomes large.

Stringham and Boehnke (1996) recently developed an algorithm to calculate the posterior probability of genotyping error for each member of a pedigree and implemented it by using the MENDEL package (Lange et al. 1988). In their first method, each genotyped pedigree member is allowed to have every possible genotype, with each genotype weighted by the probability that the genotype is in error, and then the posterior probability of the individual's genotype is computed. Since this method is too computationally intensive in the presence of highly polymorphic markers, they also developed a faster algorithm, employing a simpler model, in which only one genotyped pedigree member at a time is allowed to have every possible genotype and the other pedigree members are assigned their original genotypes. They then computed the probability for each genotyped pedigree member, conditioned on all other genotyped members being correct. Their faster algorithm identifies those individuals for whom elimination of their genotypes from the data will eliminate the inconsistency in the pedigree data. Still, their two approaches are not automated to handle more than one marker at a time and slow down in the presence of highly polymorphic markers.

Thus, there is still a clear need for a rapid and efficient program to preprocess marker data, to determine if there are any typing errors and to assist the user in identifying

Received October 2, 1997; accepted for publication April 17, 1998; electronically published May 22, 1998.

Address for correspondence and reprints: Dr. Jeffrey R. O'Connell, Department of Human Genetics, University of Pittsburgh, 130 DeSoto Street, Pittsburgh, PA 15261. E-mail: jeff@sherlock.hgen.pitt.edu

0002-9297/98/6301-0036$02.00

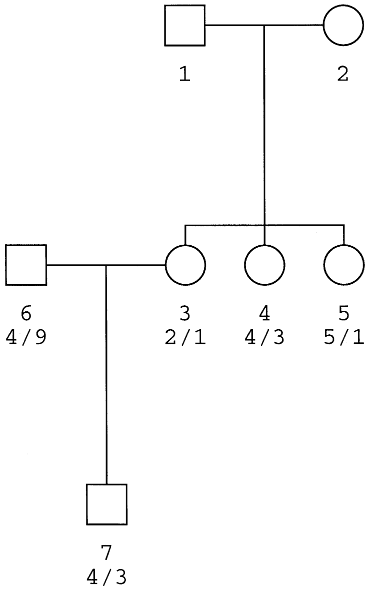


Figure 1 Pedigree with level 1 errors

them so that these errors can be resolved quickly. Ideally the program would be fast and efficient, would handle large data sets with perhaps hundreds of markers and thousands of individuals, would give detailed diagnostic information on the source of these errors, and would identify the individuals involved. It also would offer different degrees of checking, from a very quick check based on simple parent-offspring matching and allele counting, within component nuclear families, to a more complex and powerful check using full-pedigree likelihoods to help identify more-subtle errors.

Therefore, to provide researchers with such a tool, we have written a new program, PedCheck, for identification of marker-typing incompatibilities in pedigree data, that offers the capabilities described above and that also overcomes some of the difficulties and limitations present in existing packages. In particular, it relies on the set-recoding algorithm of VITESSE (O'Connell and Weeks 1995), to avoid the speed bottleneck that hinders UNKNOWN in its genotype elimination and MENDEL in its likelihood calculations; it processes one locus at a time, so that it is not limited by the number of markers it can handle; and it uses full genotype elimination to offer a check that is more comprehensive than that of GAS.

Error-Checking Algorithms

We introduce four error-checking algorithms to handle different types of data sets and different degrees of difficulty in the identification of an error. Each successive algorithm employs a more rigorous and powerful technique designed to identify the more subtle errors. For example, to check raw laboratory data that might have numerous data-entry or genotype-scoring errors, a likelihood-based check would be inefficient and may offer no additional information if simple parent-child relations can reveal these errors. Likewise, if a large pedigree has a subtle error that simple parent-child relationships cannot identify, then a more powerful check is required.

Nuclear-Family Algorithm

For each marker, the nuclear-family algorithm uses the known genotypes to check for inconsistencies between parents and offspring. An error is flagged if one or more of the following conditions is true: the alleles of a child and a parent are incompatible; the child is compatible with each parent separately but not when both parents are considered simultaneously; there are more than four alleles in a sibship; there are more than three alleles in a sibship with a homozygous child; there are more than two alleles in a sibship with two different homozygotes among the sibs; an allele is out of bounds of any specified range; at an X-linked locus, a male is not coded as "homozygous," as required by the LINKAGE programs; or an individual has only one allele defined in an autosomal system (most programs expect both alleles to be specified). Figure 1 illustrates three of these errors. First, there are more than four distinct alleles among the genotypes of siblings 3, 4, and 5; second, the genotypes of parent 3 and child 7 are incompatible; and third, under the assumption that there were only seven alleles specified for this marker, the 9 allele for individual 6 is out of range. Note that the nuclear-family algorithm is very rapid, because it involves only nuclear-family information and no genotype elimination.

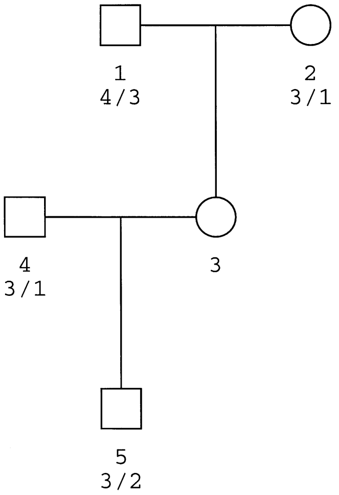


Figure 2 Pedigree without level 1 errors but with a level 2 error

The nuclear-family algorithm is appropriate as a first check, particularly for large data sets that have not been checked previously. If there are no errors or if the errors have been corrected, then there still may be inconsistencies due to the revised information or to the genotype relations inferred by analysis of pedigree members other than the sibs, offspring, or parents of an individual. Moreover, with regard to the presence of loops, the nuclear-family algorithm ignores loop information in the pedigree file, since only nuclear-family information is used. Thus, to identify errors that may arise in these situations, we introduce a second error-checking algorithm.

Genotype-Elimination Algorithm

Genotype elimination is performed via our extended version of the Lange-Goradia algorithm (Lange and Goradia 1987; Lange and Weeks 1989) for set-recoded genotypes (O'Connell and Weeks 1995). The Lange-Goradia algorithm recursively uses the nuclear-family relationships to eliminate invalid genotypes in the pedigree; the recursion is continued until no more genotypes can be eliminated. The genotype-elimination algorithm is more powerful than the nuclear-family algorithm because it can pinpoint subtle inconsistencies resulting from the elimination of certain genotypes, on the basis of more complex pedigree relations. For each pedigree and locus, the genotype-elimination algorithm identifies the first component nuclear family that is found with an error that has not been identified already by the nuclear-family algorithm, and it outputs the inferred-genotype lists for each member of that nuclear family. Only one nuclear family is reported, because a genotype-elimination error may propagate to other component nuclear families, and, thus, ascertainment of whether the pedigree contains only one such error or multiple distinct errors is difficult. Figure 2 illustrates a pedigree that is internally inconsistent but for which the nuclear-family algorithm will detect no errors. Each component nuclear family is consistent, but individual 4 determines phase in individual 5, who in turn forces individual 3 to have a 2 allele, which means that the parents of individual 3 must have a 2 allele, which is not the case.

We say that a genotype-elimination algorithm is complete if it can detect that the set of given genotypes violates Mendelian laws of inheritance. Thus, if a complete genotype-elimination algorithm finds no errors, then the genotypes are consistent with Mendelian laws of inheritance, and linkage analysis can be performed. Although the original Lange-Goradia algorithm is guaranteed to be complete only for pedigrees without loops, we have extended the algorithm and have proved that our extended algorithm is complete for all pedigrees (authors' unpublished data). Thus, our genotype-elimination algorithm is guaranteed to determine if the pedigree is consistent.

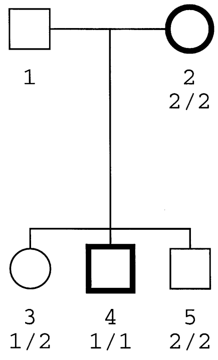


Figure 3 Pedigree with two critical genotypes, in individuals 2 and 4. The symbols for these individuals have a boldface border.

Although our genotype-elimination algorithm will always find the subtle errors, the diagnostic output of the inferred-genotype lists does not always permit easy identification of the source of the problem, since the genotype lists for untyped individuals may be long. Even if there is only one error, the individual involved may be difficult to identify by examination of only the genotype lists, since either more than one individual may be the possible source of that error or the error may not be in the particular nuclear-family data appearing in the output. Thus, we implemented two additional error-checking algorithms that can be useful for finding the source of these more subtle errors, as exemplified in the two examples of extended pedigrees given in the report by Stringham and Boehnke (1996).

These two additional error-checking algorithms focus on "critical genotypes," which we define as those genotypes of an individual that eliminate the pedigree inconsistency when removed from the data—that is, when scored as unknown. Although removal of a critical genotype may eliminate the error, this does not necessarily imply that the genotype is erroneous. For example, untyping a correctly typed parent may produce a consistent pedigree when the error actually is caused by a child's genotype. The concept of critical genotype can be extended easily to "degree *n* critical genotypes," which are defined as an *n*-tuple of genotypes of scored individuals, having the property that, when all *n* individuals in the set are untyped simultaneously, the inconsistency is eliminated. Critical genotypes are analogous to critical points and extrema of functions. The set of extrema is a subset of the critical points. Here, the set of erroneous genotypes is a subset of the critical genotypes. As mentioned in the Introduction, Stringham and Boehnke's (1996) faster algorithm determines the critical genotypes of a pedigree by using their likelihood-based statistic, whereas our method uses genotype elimination. We now describe the two additional error-checking algorithms.

Critical-Genotype Algorithm

In particularly complicated cases, one might want to invoke the critical-genotype algorithm, which attempts to identify the critical genotypes, if any, in the pedigree. This is done by "untyping" one typed individual at a time, by scoring him/her as having an unknown genotype and applying the genotype-elimination algorithm to determine if the inconsistency has been eliminated. For example, in figure 3 there are two critical genotypes: the 2/2 of individual 2 and the 1/1 of individual 4. Untyping either of these individuals will eliminate the inconsistency. There may be one or more critical genotypes, or there may be none. If there are none, higher-degree critical genotypes could be investigated, at the cost of increased processing time, but we have not yet encountered real data for which this has been necessary. Note that if the critical-genotype algorithm finds only one critical genotype, then that genotype represents the error.

Odds-Ratio Algorithm

If the critical-genotype algorithm identifies several critical genotypes at a locus, then we have no way of deciding a priori which critical genotype is most likely to be erroneous. To help distinguish between alternative critical genotypes, we have implemented an odds-ratio statistic based on single-locus likelihoods of the pedigree. First, for each individual with a critical genotype, we identify the valid typings that eliminate the inconsistency. Note that we restrict our genotypes to contain only alleles that appear in the pedigree under consideration, although there may be other alleles at that particular locus in other pedigrees. Since, by definition, untyping an individual with a critical genotype results in a consistent pedigree, we know that after genotype elimination the individual must have at least one alternative valid genotype. Second, for the particular locus, we compute and store the likelihood of the pedigree data for each alternative valid typing at each critical genotype, holding all other critical genotypes at their original value; that is, for each alternative genotype, compute the likelihood L(pedigree and that alternative genotype). Let $L_{\max}$ be the largest likelihood obtained. Note that several different genotypes may have the value $L_{\max}$. Then, for each alternative genotype, we form the quantity $L_{\max}/L$(pedigree with alternative genotype), which gives an odds ratio *against* that alternative genotype versus any genotype with likelihood $L_{\max}$. Any genotype with a value of $L_{\max}$ will have an odds ratio of 1, and, in general, the best-supported genotypes will have an odds ratio close to or equal to 1. The algorithm returns each alternative typing together with its odds ratio. Note that only the list of alternative genotypes could be used to screen any newly suggested typing for its validity (a new typing that is reliable but fails to remove the inconsistency may indicate a nonpaternity error). Note also that the odds ratio can be defined in terms of conditional probabilities: Suppose we have a pedigree in which two individuals have critical genotypes; if we let P1 and P2 represent the two individuals with critical genotypes, D the original pedigree data (with P1 and P2 set to unknown), G1 and G2 the original genotypes of P1 and P2, respectively, and A1 and A2 any two alternative genotypes for P1 and P2, respectively, then the odds ratio is $P(\mathrm{D}|\mathrm{P1} = \mathrm{G1}, \mathrm{P2} = \mathrm{A2})/P(\mathrm{D}|\mathrm{P1} = \mathrm{A1}, \mathrm{P2} = \mathrm{G2})$. Since we are attempting to determine which individual is most likely to have a genotyping error, comparing the probabilities for any two individuals, when alternately scored with their original and alternative genotypes, is a logical approach.

For example, with regard to figure 3, if we assume that there are two alleles at the locus, then individual 2 has only one consistent alternative typing, namely, 1/2. Individual 4, however, has two alternative consistent typings, 1/2 and 2/2. For the odds-ratio algorithm, we then would compute the likelihood of the pedigree, using each of these three genotypes separately, find the maximum value, and compare the maximum value to each of the three likelihoods. In two separate examples in the Applications section, we illustrate the use of this odds ratio as an indicator of the individual most likely to be the source of the incompatibility. Since computation of the likelihood requires that allele frequencies be specified, our odds-ratio algorithm has three variations: (1) use the user-defined allele frequencies; (2) assume that all alleles are equally frequent; and (3) use allele frequencies estimated from all typed individuals in the input file, by counting the number of times the allele appears at a locus, divided by the total number of alleles present. The last two variations are useful for checking raw data without having to carefully specify allele frequencies as required for linkage analysis. In general, use of estimated instead of equally frequent allele frequencies leads to a bigger spread in the odds ratios.

Implementation

Algorithms

We have implemented our four error-checking algorithms in a computer program, PedCheck. In PedCheck, we identify the algorithms in terms of levels: level 1 checking uses the nuclear-family algorithm; level 2 checking uses the genotype-elimination algorithm; level 3 checking uses the critical-genotype algorithm; and level 4 checking uses the odds-ratio algorithm.

Timing Results

To illustrate the performance of PedCheck at each level, we used an unloaded Sun Ultra 170 computer and a test data set typical of a genomewide study, consisting of 1,095 individuals and 289 markers (134,561 genotypes). The data set was drawn from the U.K. Arthritis and Rheumatism Council repository of material from families with two or more affected siblings. For our genomewide test data set, level 1 checking took <1 min to run and found seven inconsistencies, all in a single pedigree. The genotype-elimination algorithm is more computationally intensive than the nuclear-family algorithm but still is quite fast. Level 2 checking took 14 min to run, for all 289 markers, and found two additional errors, in two pedigrees not identified in level 1 checking. Note that the level 2 option automatically runs level 1 first. Level 3 checking took 15 min and identified a total of 23 critical genotypes. Again, the level 3 option automatically runs levels 1 and 2 first. Although our odds-ratio statistic involves computation of the likelihood of the pedigree, it is very fast to compute, since it involves only single-locus likelihoods. Finally, level 4 checking took 15 min to identify and perform a full-pedigree likelihood calculation for each of the total of 170 alternative genotypes across the 23 critical genotypes.

With regard to checking data, level 1 is recommended as a first check, particularly for large data sets that have not been checked previously. If there are no level 1 errors or if the errors have been corrected, then level 2 should be run to detect more-subtle errors. Level 3 should be run to determine the critical genotypes, and level 4 should be run to pinpoint the most likely source of error. Level 2 should be rerun after any change in the data, until PedCheck indicates that the pedigree is consistent.

Input Formats

Often, a researcher finds a problem with the pedigree data only at the time of performing linkage analysis, which means that both the raw allele-size data and the individual IDs may have been already renumbered; in such a situation, finding the error may involve tedious cross-referencing of the new numbering against the original raw data. Since error checking ideally is performed on the raw data, PedCheck also has the capability of handling pedigrees in LINKAGE format or pre-makeped format and allele data in integer base-pair sizes, without

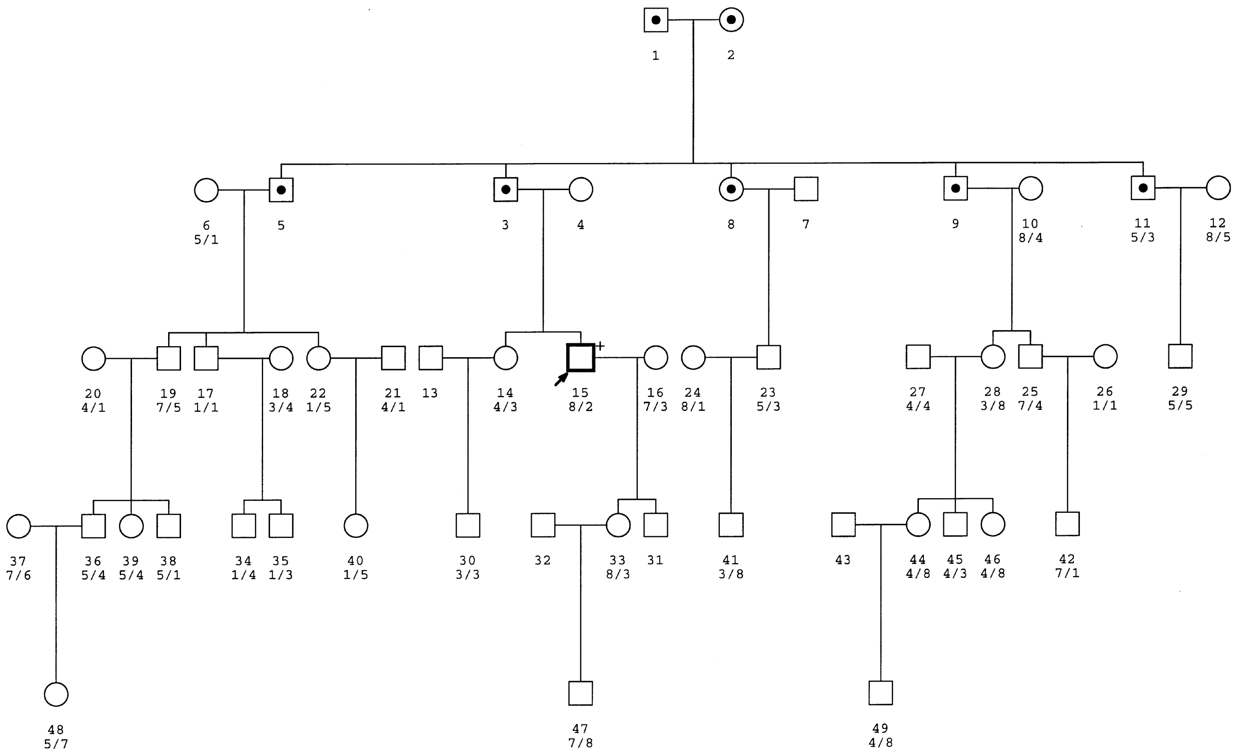


Figure 4 Pedigree from Wijsman and Guo (Ott 1993), with one genotyping inconsistency. The symbols with a bullet in the center indicate members of the nuclear family, identified by use of level 2 checking; the symbol with a boldface border indicates the individual with a critical genotype; the plus sign (+) indicates the individual with the actual genotype error; and the arrow points to the individual identified by PedCheck as most likely to have an erroneous genotype.

```
*********** LEVEL 2 ERRORS ****************

##### GENOTYPE ERROR: Pedigree: 1   Locus:  1.   Name: Marker_1. #####

ORDERED GENOTYPE LISTS: Any allele greater than 8 is set_recoded.
(U) Father 1:    3|1  5|1  3|2  5|2  1|3  2|3  3|3  4|3  5|3
           6|3  7|3  8|3  3|4  5|4  1|5  2|5  3|5  4|5  5|5
           6|5  7|5  8|5  3|6  5|6  3|7  5|7  3|8  5|8
(U) Mother 2:    3|1  5|1  3|2  5|2  1|3  2|3  3|3  4|3  5|3
           6|3  7|3  8|3  3|4  5|4  1|5  2|5  3|5  4|5  5|5
           6|5  7|5  8|5  3|6  5|6  3|7  5|7  3|8  5|8

(U) Child 3:    3|2  4|2  2|3  8|3  2|4  8|4  3|8  4|8
(U) Child 5:    7|1  1|7
(U) Child 8:    3|11  5|11  11|3  3|3  5|3  11|5  3|5  5|5
(U) Child 9:    7|3  3|7
(T) Child 11:    5|3  3|5

 ****** CHECKING HIGHER LEVEL ERRORS. *******

Untyping any person listed will result in a consistent pedigree at the given
locus.

 Using estimated allele frequencies.

 ##### Pedigree: 1 #######

 Locus:  1.   Name: Marker_1

                  Valid Typings     Odds Against (vs. best)
 Person 15:   2/ 8 (ORIGINAL)
                     3/ 8           1.0
                     5/ 8           2.0
                     7/ 8           2.0
```

Figure 5 Diagnostic output from PedCheck, for pedigree in figure 4.

the need to specify allele frequencies. No LINKAGE-format data file is required, except when user-defined allele frequencies are used in level 4 checking; otherwise, PedCheck requires only a file containing the names of the markers. Note that, by design, PedCheck does not check disease loci, since any inconsistencies at a disease locus do not involve genotyping errors. VITESSE (O'Connell and Weeks 1995) will identify any errors due to an incorrect disease model.

Applications

We illustrate the performance of our PedCheck program on the two challenging pedigrees analyzed recently by Stringham and Boehnke (1996), in which the source of the error is difficult to discern. The first pedigree, displayed in figure 4, is from the research of Wijsman and Guo (Ott 1993). When Stringham and Boehnke (1996) applied their posterior-probability method (which took 40 min at our workstation), they concluded that individual 15 had a .995 posterior probability of genotyping error. When we ran this pedigree through PedCheck, using level 4 checking (which took 10 s), we obtained the output shown in figure 5.

From this diagnostic output, obtained by use of the higher-level options, we see that there were no level 1 errors, but there was a single level 2 error. If we consider only the level 2 output, it is clear that both child 5 and child 9 have a single unordered genotype—1/7 and 3/7, respectively—after genotype elimination ("U" within parentheses indicates that they originally were untyped [fig. 5]). Child 8 has either a 3 allele or a 5 allele in all of her possible genotypes. Thus, as Stringham and Boehnke (1996) pointed out, children 5, 8, 9, and 11 are consistent with the parents having the four alleles (1, 3, 5, and 7). However, child 3 has either a 2 allele or an 8 allele in all of his possible genotypes, which indicates that the most likely source of the error is among his descendants (who are forcing him to have the 2 allele and the 8 allele). In fact, if we now examine the output from levels 3 and 4, we see that there is only one critical genotype, in individual 15, who is a descendent of individual 3. Thus, PedCheck clearly identified individual 15 as the source of the error and indicated that genotype 3/8 has the best odds of being the correct typing. However, this pedigree actually includes a nonpaternity error; so, the alternative valid genotypes are not relevant. Note that nonpaternity errors are quite difficult to distinguish from actual genotype errors, on the basis of any single-locus statistic alone.

Figure 6 presents the second pedigree from the report by Stringham and Boehnke (1996), for which the true genotyping error was later revealed, by retyping, to be in individual 28. When we ran this pedigree through PedCheck, using level 4 checking (which took 5 s, compared with 25 min for Stringham and Boehnke's program), we obtained the output shown in figure 7.

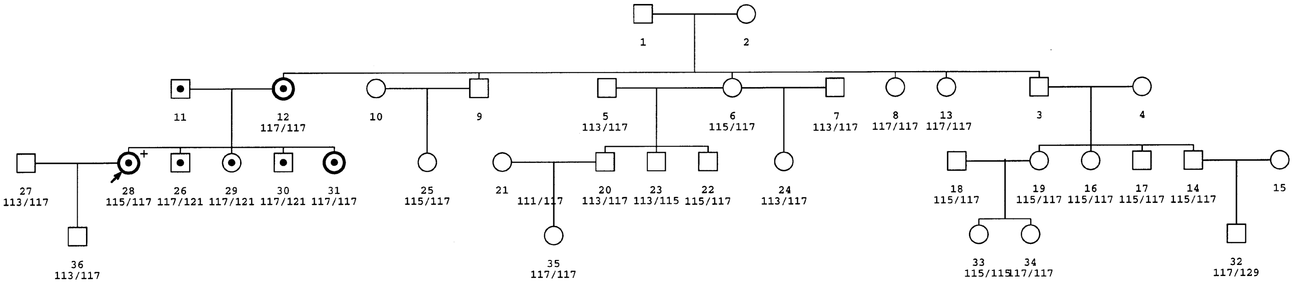
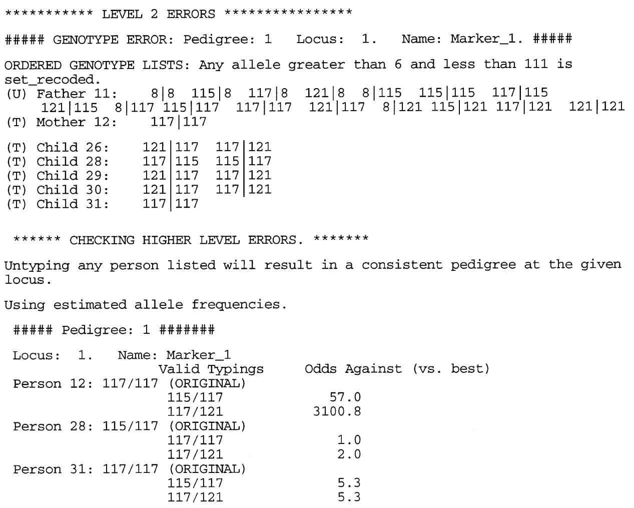


Figure 6 Eye-disease pedigree from the report by Stringham and Boehnke (1996), with one genotyping inconsistency. Alleles are given in base-pair sizes. The symbols with a bullet in the center indicate members of the nuclear family, identified by use of level 2 checking; the symbols with a boldface border indicate the individuals with a critical genotype; the plus sign (+) indicates the individual with the actual genotype error; and the arrow points to the individual identified by PedCheck as most likely to have an erroneous genotype.



```
*********** LEVEL 2 ERRORS ****************

##### GENOTYPE ERROR: Pedigree: 1   Locus:  1.   Name: Marker_1. #####

ORDERED GENOTYPE LISTS: Any allele greater than 6 and less than 111 is
set_recoded.
(U) Father 11:    8|8  115|8  117|8  121|8  8|115  115|115  117|115
    121|115  8|117 115|117  117|117  121|117  8|121 115|121 117|121  121|121
(T) Mother 12:    117|117

(T) Child 26:    121|117  117|121
(T) Child 28:    117|115  115|117
(T) Child 29:    121|117  117|121
(T) Child 30:    121|117  117|121
(T) Child 31:    117|117


 ****** CHECKING HIGHER LEVEL ERRORS. *******

Untyping any person listed will result in a consistent pedigree at the given
locus.

Using estimated allele frequencies.

 ##### Pedigree: 1 #######

 Locus:  1.   Name: Marker_1
                  Valid Typings     Odds Against (vs. best)
 Person 12: 117/117 (ORIGINAL)
                   115/117             57.0
                   117/121           3100.8
 Person 28: 115/117 (ORIGINAL)
                   117/117              1.0
                   117/121              2.0
 Person 31: 117/117 (ORIGINAL)
                   115/117              5.3
                   117/121              5.3
```

Figure 7 Diagnostic output from PedCheck, for pedigree in figure 6.

This output indicates that there are no level 1 errors, indicating a more subtle genotype error. One level 2 error appeared, but the genotype lists of the parents do not easily reveal who in the family might have the erroneous genotype. However, the level 3 output indicates that there are three critical genotypes, in individuals 12, 28, and 31, and the level 4 output indicates that each of these individuals has two alternative consistent genotypes. The level 4 odds ratios were determined by computing the likelihood of the pedigree, for each of these six genotypes separately; determining the highest likelihood; and then comparing the likelihoods, to form the odds ratios. So, for example, the odds against individual 12 being typed 117/121 versus individual 28 being typed 117/117 are 3,100.8:1. Thus, level 3 checking immediately identified three pedigree members to consider further, and, in fact, these individuals belong to the same nuclear family: individuals 28 and 31 are children of individual 12. Therefore, untyping either child 28 or child 31 will eliminate the inconsistency, and, in fact, these two individuals had the highest posterior probability of error (.679 for individual 28 and .315 for individual 31), when analyzed by use of Stringham and Boehnke’s approach. However, level 3 checking also showed that untyping only the mother, individual 12, also would eliminate the inconsistency, but she is not indicated as a likely source of error, by Stringham and Boehnke’s approach, since her posterior probability of error was only .008. The reason is that all five of her children inherited her 117 allele. The high odds ratios generated by PedCheck also indicated that individual 12 is not the likely source of error. Thus, we would conclude that individual 28 most likely has the erroneous genotype and that the correct genotype is 117/117. Retyping revealed that this is in fact the true answer (M. Boehnke, personal communication).

Thus, in these two difficult examples, our approach not only identified the same individual that Stringham and Boehnke (1996) identified as being the most likely source of the error but also, in the second example, predicted the correct typing, on the basis of the odds ratios. We, however, are not advocating that the odds-ratio statistic be used to “correct” erroneous genotypes (this has to be done by use of the real data) but, rather, that it be used as an indicator of which individuals are the best candidates for follow-up in the lab. On the other hand, researchers often choose to perform an analysis, without correcting the error in the lab, by untyping enough individuals until the pedigree is consistent. In such a case, we recommend untyping an individual who has an alternative typing with an odds ratio of 1 (or as close as possible to 1).

Discussion

We have developed a new computer program, PedCheck, to assist researchers with quick and efficient identification of non-Mendelian inconsistencies in pedigrees. PedCheck uses four different error-checking algorithms, ranging from a very fast check of the pedigree errors to a more sophisticated check based on a likelihood odds ratio.

Throughout this discussion we have implicitly assumed that, if a pedigree is consistent with Mendelian laws of inheritance, then there are no genotype errors. Thus, we are not addressing genotyping errors that may be present in pedigrees even though all the genotypes are consistent. For example, the algorithm used by Stringham and Boehnke (1996) computes prior probabilities for every typed individual, not just for individuals who have a critical genotype, so that consistent genotypes also might be flagged as sources of error, particularly when those genotypes have rare alleles. Also, since PedCheck performs only single-locus checks, it cannot detect those errors that may become apparent only through unusual linkage results or an excess of multiple recombinants. To detect such errors, multipoint analysis is often more useful, and therefore various approaches have been implemented in software programs (Lathrop et al. 1983*a,* 1983*b;* Sobel et al. 1995; Ehm et al. 1996). Other types of pedigree error that may or may not produce an inconsistent pedigree are paternity errors or the presence of half sibs treated as sibs. If these pedigree errors are detected through genotype errors, then PedCheck can assist the researcher, since there is an option to give summary statistics indicating the number of loci at which each individual has a critical genotype; thus, if an individual has a critical genotype at many loci, this may indicate a pedigree error. Again, morespecialized software has been developed for these types of situations (Stivers et al. 1996; Boehnke and Cox 1997; Göring and Ott 1997; Ehm and Wagner 1998). Finally, although PedCheck was developed to find errors that lead to inconsistent genotypes, the higher-level–checking algorithms in the program can be easily adapted to identify any typed individuals who have valid alternative genotypes that have an odds ratio greater than some predefined threshold. If an individual had alternative genotypes that were significantly more likely than the original genotype, then this might be an indication that the original genotype was an error. For future work, it would be of interest to perform a detailed study of how useful the odds-ratio statistic would be in the pinpointing of these types of errors in pedigrees consistent with Mendelian inheritance.

In routine genotyping, a large majority of genotyping errors will be detected and diagnosed by simple level 1 error checking, which is computationally extremely fast. A small fraction of genotyping errors will require a more sophisticated level of diagnostic analysis, involving examination of the underlying genotype lists generated by the genotype-elimination algorithm produced by level 2 checking. When this output does not indicate the source of the error, level 3 and level 4 checking can be used to identify the individual(s) most likely to be involved. Thus, PedCheck should be able to make a significant contribution, by virtue of its ability to quickly analyze many loci with large numbers of alleles, using four powerful error-checking algorithms at once. Occasionally, there still will arise cases for which the program developed by Stringham and Boehnke (1996) will prove to be very useful, particularly when a pedigree contains many loops, since the algorithm to handle loops, in MENDEL, is quite efficient.

Acknowledgments

This work was supported, in part, by National Institutes of Health grant HG00932; the Wellcome Trust Centre for Human Genetics at the University of Oxford; the University of Pittsburgh; Association Française Contre Les Myopathies; and the W. M. Keck Center for Advanced Training in Computational Biology at the University of Pittsburgh, Carnegie Mellon University, and the Pittsburgh Supercomputing Center. We thank Paul Wordsworth for providing the test data set, the collection of which was funded by the U.K. Arthritis and Rheumatism Council. We also thank Carol Haynes, Meg Cooper, Tara Matise, Janet Sinsheimer, and the reviewers, for providing useful comments. This program is not to be confused with PEDCHK of S.A.G.E. (Case Western Reserve University 1994).

Electronic-Database Information

URLs for programs discussed in this article are as follows:

Genetic analysis system (GAS), version 2.0, http://users.ox.ac.uk/~ayoung/gas.html

PedCheck may be obtained by anonymous ftp to ftp://watson.hgen.pitt.edu or to the European Bioinformatics Institute mirror site, http://ftp.ebi.ac.uk

References

Boehnke M, Cox NJ (1997) Accurate inference of relationships in sib-pair linkage studies. Am J Hum Genet 61:423–429

Case Western Reserve University, Department of Epidemiology and Biostatistics (1994) Statistical analysis for genetic epidemiology (S.A.G.E.), release 2.2, Cleveland

Ehm MG, Kimmel M, Cottingham RW Jr (1996) Error detection for genetic data, using likelihood methods. Am J Hum Genet 58:225–234

Ehm MG, Wagner M (1998) A test statistic to detect errors in sib-pair relationships. Am J Hum Genet 62:181–188

Göring HHH, Ott J (1997) Relationship estimation in affected sib pair analysis of late-onset diseases. Eur J Hum Genet 5: 69–77

Lange K, Goradia TM (1987) An algorithm for automatic genotype elimination. Am J Hum Genet 40:250–256

Lange K, Weeks D, Boehnke M (1988) Programs for pedigree analysis: MENDEL, FISHER and dGENE. Genet Epidemiol 5:471–472

Lange K, Weeks DE (1989) Efficient computation of lod scores: genotype elimination, genotype redefinition, and hybrid maximum likelihood algorithms. Ann Hum Genet 53:67–83

Lathrop GM, Hooper AB, Huntsman JW, Ward RH (1983*a*) Evaluating pedigree data. I. The estimation of pedigree error in the presence of marker mistyping. Am J Hum Genet 35: 241–262

Lathrop GM, Huntsman JW, Hooper AB, Ward RH (1983*b*) Evaluating pedigree data. II. Identifying the cause of error in families with inconsistencies. Hum Hered 33:377–389

Lathrop GM, Lalouel JM (1984) Easy calculations of lod scores and genetic risks on small computers. Am J Hum Genet 36:460–465

Lathrop GM, Lalouel JM, Julier C, Ott J (1984) Strategies for multilocus linkage analysis in humans. Proc Natl Acad Sci USA 81:3443–3446

Lathrop GM, Lalouel JM, White RL (1986) Construction of human linkage maps: likelihood calculations for multilocus analysis. Genet Epidemiol 3:39–52

O'Connell JR, Weeks DE (1995) The VITESSE algorithm for rapid exact multilocus linkage analysis via genotype set-recoding and fuzzy inheritance. Nat Genet 11:402–408

Ott J (1993) Detecting marker inconsistencies in human gene mapping. Hum Hered 43:25–30

Sobel E, Lange K, O'Connell JR, Weeks DE (1995) Haplotyping algorithms. In: Speed TP, Waterman MS (eds) Genetic mapping and DNA sequencing. Springer-Verlag, New York, pp 1–22

Stivers DN, Zhong Y, Hanis CL, Chakraborty R (1996) RELTYPE: a computer program for determining biological relatedness between individuals based on allele sharing at microsatellite loci. Am J Hum Genet Suppl 59:A190

Stringham HM, Boehnke M (1996) Identifying marker typing incompatibilities in linkage analysis. Am J Hum Genet 59: 946–950