ELSEVIER Computational Statistics & Data Analysis 28 (1998) 225–232

COMPUTATIONAL STATISTICS & DATA ANALYSIS

A method for calculating probability convolution using "ternary" numbers with application in the determination of twin zygosity

J.H. Zhao, P.C. Sham

Department of Psychological Medicine and Department of Biostatistics & Computing, Institute of Psychiatry De Crespigny Park, Denmark Hill, London SE5 8AF, UK

Received 1 August 1997; received in revised form 1 November 1997; accepted 2 April 1998

Abstract

It often happens that an efficient algorithm can be designed by considering some special properties of the problem. In this short paper we describe a problem in twin zygosity determination that involves a convolution of probability distributions. Our solution to this problem was to develop an algorithm for counting in the base-3 or "ternary" number system.

Keywords: Trinary numbers; Twin zygosity; Recursive algorithm

1. Introduction

Human geneticists differentiate between two kinds of twins. If two twins are derived from a single fertilised egg then they are called monozygote (MZ) twins. Otherwise they are derived from two separate fertilised eggs, and are called dizygotic (DZ) twins. These two kinds of twins differ in that MZ twins have exactly the same genetic material as each other, whereas DZ twins are genetically as alike to each other as ordinary brothers and sisters. The genetic overlap between two DZ

*Corresponding author. E-mail: j.zhao@iop.bpmf.ac.uk, p.sham@iop.bpmf.ac.uk.

0167-9473/98/$19.00
PII S0167-9473(98)00036-X

twins is therefore 50%, on average. The greater genetic similarity of MZ twins as compared to DZ twins is the basis of the twin method, which aims to evaluate the relative importance of genetic and environmental factors in causing individual differences in the population. The application of this method requires the zygosity of each twin pair in a sample to be determined.

We have recently developed a program to determine twin zygosity according to DNA marker data. In developing this program, we came across an interesting problem that we solved by an algorithm for counting in the base-3 or "ternary" number system.

2. The problem

Each person has two sets of chromosomes, one transmitted from the father and the other from the mother. Each chromosome can be thought of as a linear array of genes, each having a particular function. Interspersed between the genes are "markers" that have no known biological function but are identical to genes in all other respects. The order of genes (and markers) on a chromosome is fixed, so that a particular gene is always found at the same position, called a locus. Each person has two genes at each locus, one in the set of chromosomes inherited from the father, and one in the set of chromosomes inherited from the mother. At each locus, the genes can take several alternative forms, each of which is called an allele. Two genes that are of the same form are said to be identical-by-state (IBS). Two genes may be IBS because they are descended from the same ancestor and are replicates of the same ancestral gene. This kind of gene sharing is called identity-by-descent (IBD). Genes can also be IBS not because they are IBD but as a matter of coincidence, as there are only a finite number of alleles at each locus.

MZ twins have both genes IBD at every locus in the genome. In contrast, DZ twins have a 25% chance that both genes are IBD, a 50% chance that one gene is IBD, and a 25% chance that no gene is IBD, at each locus. Our aim is to exploit this difference between MZ and DZ twins in order to determine zygosity. If there is any locus at which a twin pair is not IBS for both genes, then the twin pair can be designated as DZ. On the other hand, even DZ twins have a non-zero probability of being IBS for both genes at every locus. We wish to calculate this probability, which gives an indication of the proportion of DZ twins which might be misclassified as MZ.

The problem is complicated by the fact that the IBD status of a twin pair is not directly available. Instead, the IBD status must be inferred indirectly from IBS status. A few possible patterns of IBS relationships between two twins are shown in Table 1.

Suppose that data are available for n markers on different chromosomes (so that they are statistically independent). For convenience we imagine them as forming a virtual line array.

1	2	... ...	n

Table 1
Possible patterns of IBS sharing between a twin pair

Twin1	Twin 2	IBS
11	11	2
12	12	2
11	12	1
12	23	1
11	22	0
11	23	0
12	34	0

The number of markers IBS at each location is either 0, 1 or 2. The total number of markers IBS can therefore range from 0 to $2n$. For MZ twins, each locus must have both markers IBS, so that the total number of markers IBS must be $2n$, unless there are errors in the data. For DZ twins, the probability distribution of the number of alleles IBS at each locus is a function of the frequencies of the alleles. Let the number of alleles at locus i be n_i, and the allele frequencies be p_{ij}, for $j = 1, \ldots, n_i$. The IBS distribution of the ith locus in DZ twins, denoted as q_{i0}, q_{i1}, q_{i2} were derived by Lange (1986) as follows.

$$q_{i0} = \frac{1}{4}\left[\sum_j p_{ij}^2(1-p_{ij})^2 + 2\sum_{j<k} p_{ij}p_{ik}(1-p_{ij}-p_{ik})^2\right],$$

$$q_{i2} = \frac{1}{4} + \frac{1}{2}\sum_j p_{ij}^2 + \frac{1}{4}\left[2\left(\sum_j p_{ij}^2\right)^2 - \sum_j p_{ij}^4\right],$$

$$q_{i1} = 1 - q_{i0} - q_{i2},$$

From the IBS distributions of the n individual loci, we wish to obtain the probability distribution of the total number of markers IBS for DZ twins. This will enable us to determine the probability that a pair of DZ twins will have 2n markers IBS, and therefore be misclassified as MZ. It also allows a goodness-of-fit test between the observed IBS distribution of DZ twin pairs for a sample with an expected IBS distribution, in order to assess whether the assumptions of the model are valid.

3. The algorithms and implementation

Our problem is therefore to obtain the convolution of n probability distributions, where n may be variable among twin pairs. Each component random variable has three possible values, 0, 1, or 2, so that there are 3^n joint outcomes. To consider all these outcomes in standard programming languages requires n loop indices, one for

each component random variable. Each index starts at 0 and ends at 2, and represents the IBS status of a location. This generates 3^n loops each representing a joint outcome. Each loop involves the calculation of a total IBS value (i.e. the sum of the IBS value of all locations) and a probability (i.e. the product of the IBS probabilities of all locations). The probability of a certain total IBS value is the sum of the probabilities of all outcomes associated with that total IBS value. This, however, presents us with the problem that the number of loop indices is both large and variable. Most high-level programming languages can only handle a limited number of loop indices.

However, on closer examination of this problem we find that we can take advantage of the regular pattern of the indices. The 3^n possibilities can be enumerated by counting through a base-3 (or "ternary") number sequence. For $n = 2$ the sequence is 00, 01, 02, 10, 11, 12, 20, 21, 22. In other words, the n digit of each "ternary" number can be used to represent the IBS status of the n marker loci. Two alternative algorithms are immediate.

3.1. *Algorithm 1*

We first get the most significant trinary digit and subtract the number it represents to get the remainder. We then proceed to get the ternary digit for the remainder and so on until the remainder is less than 3. For example, decimal number 80 is less than $3^4 = 81$ but bigger than $3^3 = 27$. the most significant ternary digit is therefore 2 in the place of 3^3. We then subtract 2×3^3 and get decimal 26 that falls within interval $[3^2, 3^3]$, and so we have 2 in the place of 3^2. Repeating the same procedures four times we obtain the ternary number 2222. We can store numbers that are powers of three in advance, so that we only need to compute them once.

3.2. *Algorithm 2*

We simply count directly in base 3. Starting from 0 and increasing 1 at a time until the required number is obtained. Each increment of 1 involves adding 1 to the least significant digit and "carrying over" to the next digit if the digit exceeds 2.

Subroutines implementing the above algorithms are appended. We present the program in QBASIC, as it is included in nearly all the MS-DOS versions and hosts many features that traditional programming languages do not normally have.

Subroutine 1 is more suitable for converting an arbitrary decimal number to base 3. It is a "greedy-type" algorithm. However, it is restricted by machine precision to $n = 33$ for double precision and n = 15 for single precision.

Subroutine 2 is easily applied to number any base and is particularly efficient for generating a consecutive sequence of numbers, as is required for our current problem. Besides, it may serve as a simple example of recursive algorithm. Better known examples of recursive algorithms include the calculation of factorials, Hanoi tower and related routines.

Table 2
IBS distributions for DZ twins

IBS	Probability	Truncated Probability
0	0.0000001	0.0000001
1	0.0000047	0.0000047
2	0.0000745	0.0000746
3	0.0007020	0.0007026
4	0.0043547	0.0043587
5	0.0187297	0.0187469
6	0.0573779	0.0574304
7	0.1268247	0.1269407
8	0.2030294	0.2032151
9	0.2345095	0.2347241
10	0.1929296	0.1931061
11	0.1100498	0.1101505
12	0.0413198	0.0413576
13	0.0091795	0.0091879
14	0.0009140	*** Truncated ****

Using either of these routines, we can easily enumerate all the 3^n possibilities, recording the IBS value as well as the probability of each instance. The IBS distribution is obtained by summarizing the probabilities over all instances that produce the same IBS value, for each possible IBS value.

Note that the digits from the two subroutines differ by one digit shift and that the DECLARE statements in both routines could be added by QBASIC automatically.

4. An example

For illustration we give an example of seven markers. Table 2 shows the calculated IBS probabilities for DZ twins given these seven loci each with five alleles of equal frequencies. The probability of a DZ twin pair sharing all $2n$ markers IBS provides an indication of how likely it is for a DZ twin pair to be misclassified as MZ. In this example, the chance that a DZ twin pair will share all $2n$ markers IBS and be misclassified as MZ is less than one in a thousand. As a check of model assumptions (such as the random mating and the absence of genotyping errors), the observed IBS distribution of the sample, given that IBS is incomplete, can be compared to the theoretical truncated IBS distribution, which is obtained by dividing each IBS probability (for IBS of between 0 and $2n - 1$) by the probability that IBS is less than $2n$. The statistical significance of the differences between the observed and theoretical distribution can be evaluated by the usual Pearson Chi-square test.

5. Other applications

These algorithms can be used to obtain the probability distribution of the sum of a finite number of discrete random variables $X_1, X_2, \ldots, X_n$, provided that the possible values that each random variable can take are the same.

6. Acknowledgements

We thank two anonymous referees and editors for their helpful comments which greatly improved our first draft, and the Wellcome Trust for financial support.

Appendix A

Here we give the QBASIC source and sample driver for each routine. Interested readers may change them to other form, say C, without loss of generality. This also applies to the situation when we have discrete probabilities with similar pattern.

Subroutine 1.

```
SUB radix (y, n, save, d)
REM ZHAO JH 16/10/96 (UK)
PRINT "# # ="; y
x = y
FOR i = 1 TO n
  d(i) = 0
NEXT
kmax = 0
FOR i = 1 TO n
  FOR k = 0 TO n
    IF x = save(k) GOTO L0
    IF x > save(k) AND x < save(k + 1) GOTO L1
  NEXT
L0: IF kmax < k THEN kmax = k
  d(k + 1) = 1
  GOTO done
L1: IF kmax < k THEN kmax = k
REM change here to allow for other base
  IF x ≥ 2 * save(k) THEN j = 2 ELSE IF k = 0 THEN j = x ELSE j = 1
  x = x − j * save(k)
  d(k + 1) = j
  IF x = 0 THEN GOTO done
NEXT
done:
```

```
sum = 0
FOR i = kmax TO 0 STEP - 1
  sum = sum + d(i + 1) * save(i)
  PRINT USING "############ ############
  ##################";
  d(i + 1); save(i); d(i + 1) * save(i); sum
NEXT
END SUB
```

Subroutine 2.

```
SUB digit (radix1, n, d, i)
REM subroutine to express decimal number to (radix1 + 1)-base
REM Zhao JH, 21/10/96 (UK)
  IF d(i) < radix1 THEN
  d(i) = d(i) + 1
  GOTO ok
ELSE
  d(i) = 0
  d(i + 1) = d(i + 1) + 1
  IF d(i + 1) ≤ radix1 GOTO ok
END IF
CALL digit(radix1, n, d, i + 1)
ok:
END SUB
```

Sample driver program for **Subroutine 1.**

```
DECLARE SUB radix (y#, n!, save#, d#)
DEFDBL A-H, O-Z
n = 35: maxn = 3^n
DIM SHARED save(0 TO n), d(0 TO n + 1)
FOR i = 0 TO n
  save(i) = 3^i
NEXT
FOR i = 25 TO 35
  PRINT "power i = "; i
  x = 3^i - 1
  CALL radix(x, n, save, d)
NEXT
END
```

Key word SHARED is simply used to communicate array with the subroutines, the lower bounds of the arrays are explicitly declared as starting from 0. As noted above, we calculated the powers of three in advance to relieve the computing burden.

Sample driver program for **Subroutine 2.**

```
DECLARE SUB digit (radix!, n!, d!, i!)
n = 4: radix = 3: maxn = radix ^ n - 1
DIM SHARED d(0 TO n)
FOR i = 0 TO n
  d(i) = 0
NEXT
FOR i = 1 TO maxn
  x = i
  PRINT USING "x = # # # # # # #"; x
  CALL digit(radix - 1, n, d, 0)
  FOR j = n - 1 TO 0 STEP - 1
    PRINT USING "# #"; d(j);
  NEXT
  PRINT
NEXT
END
```

Reference

Lange, K., 1986. The affected sib-pair method using identity by state relations. Am. J. Hum. Genet. 39(1), 148–150.