An object-oriented Bayesian network for estimating mutation rates

A. P. Dawid
Department of Statistical Science
University College London

Abstract

We describe the use of the object-oriented HUGIN 6 probabilistic expert system software to structure the problem of estimating mutation rates on the basis of family data when paternity can not be regarded as certain.

Key words: Bayesian network; DNA profiling; HUGIN; Mutation; Paternity testing; Probabilistic expert system.

1 Paternity testing and mutation

When conducting paternity testing, we may have available DNA profiles from a triple consisting of the mother m, child c, and putative father pf. Each profile consists of a number of *genotypes*, one for each of the *genetic markers* examined. In turn, the genotype for a given individual on a given marker consists of an unordered pair of *alleles*, one inherited from the individual's mother, and one from the father (although it is not possible to distinguish which is which). The STR (short tandem repeat) markers used in current forensic practice have a finite number (up to 20 or more) of values (*alleles*), each a small positive integer.

Application to a specific case involves two stages. First, we check the DNA profiles of the triple to see whether they are logically compatible with the hypothesis H_0 : "tf = pf", that the putative father pf is the true father tf, and the laws of Mendelian inheritance. If not, this would be a *prima facie exclusion*, and normally the case would be dropped. In the case of compatibility, the specific data on the profiles would be used as the basis of the calculation of an appropriate *likelihood ratio* for comparing the hypothesis H_0 at issue with some alternative hypothesis H_1: for example, that the true father is an unknown person whose genes can be regarded as drawn randomly from the gene-pool of the relevant population.

One feature that can complicate this procedure is the possibility of *mutation* at one or more of the markers observed. If this occurs, then even if in fact tf = pf, we may obtain a seeming exclusion. Since the STR markers used have relatively high mutation rates (Brinkmann *et al.* 1998; Henke and Henke 1999; Sajantila 1999), this possibility may need to be taken seriously if, for example, the triple shows compatibility on all but one (or sometimes two) of the markers. Dawid *et al.* (2001) describe the relevant calculations in simple cases. For more complex cases, for example when the putative father is not available for testing but one can obtain data from a close relative or relatives, one can make use of forensic Bayesian Networks (Dawid *et al.* 2002b). See Cowell *et al.* (1999) for general background on Bayesian networks, also known as Probabilistic Expert Systems.

The above analyses are naturally sensitive to the mutation rates assumed (indeed, to the specific rates of mutation transitions from one allele to another). However, the data from which such rates are estimated typically themselves consist of just such collections of seemingly incompatible triples, for which the evidence in favour of true paternity is regarded as strong. There is then a danger of confounding between the two possible reasons for incompatibility, namely non-paternity, and paternity plus mutation; and this complicates the estimation problem.

Work in progress (Dawid *et al.* 2002a) analyses in detail how this estimation problem can itself be structured and solved using Bayesian networks. The present paper describes the use of the HUGIN 6 software for this purpose.

2 HUGIN 6

Version 6 of the Probabilistic Expert System software HUGIN[1] supports object-oriented construction of

[1]http://www.hugin.com/

Table 1: Glossary

mamg	mother's actual maternal gene
mapg	mother's actual paternal gene
mgt	mother's genotype
pfamg	putative father's actual maternal gene
pfapg	putative father's actual paternal gene
pfgt	putative father's genotype
tfamg	true father's actual maternal gene
tfapg	true father's actual paternal gene
comg	child's original maternal gene
copg	child's original paternal gene
camg	child's actual maternal gene
capg	child's actual paternal gene
cgt	child's genotype
comp?	compatibility of genotypes
tf = pf?	true father is putative father?
lambda	(approximate) average overall mutation rate
rho	ratio of paternal to maternal mutation rate
h	mixing parameter for mutation models

complex Bayesian networks (Koller and Pfeffer 1997). Each network created defines a generic *class* of networks, *instances* of which can be used, as desired, in place of atomic nodes in other networks. This facility is particularly convenient when identical or similar modules recur in different parts of the same global network.

2.1 Top-level network

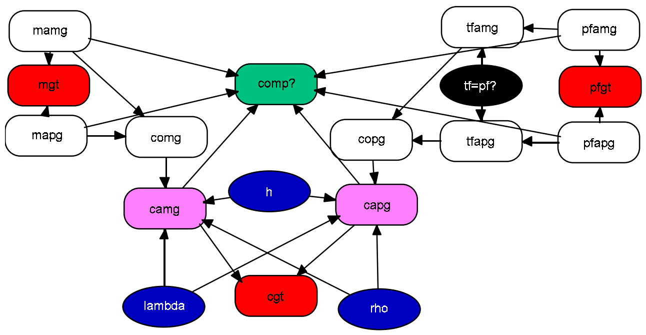

Figure 1: Top-level network **comp6**

Figure 1 is the top-level network **comp6** describing our problem (for a single marker). The symbols are decoded in Table 1. The colours/shadings are purely mnemonic, and do not affect interpretation or calculations.

The oval nodes represent variables entering this network alone; the rounded rectangular nodes are themselves instances of other network classes, to be described below. Arrows between nodes of either kind represent probabilistic dependence of the "child"[2] node on its "parents", roughly as described in Cowell *et al.* (1999). However, for an instance node, itself containing regular nodes, we need further to specify the relevant *interface nodes* (*input* nodes for a "child" instance, *output* nodes for a "parent" instance): These are not shown in Figure 1, but can be displayed by "expanding" an instance node — Figure 2 shows this for nodes `pfamg`, `pfgt`, `copg` and `tfamg`. An arrow into an input node represents a binding of that node to its "parent" node, whereas an arrow out of an output node (and into a non-input node) has the usual probabilistic "parent-child" interpretation. Although only interface internal nodes are displayed on expansion of an instance node, there may also be further internal "hidden" nodes in the corresponding network class, which can be seen by opening that network.

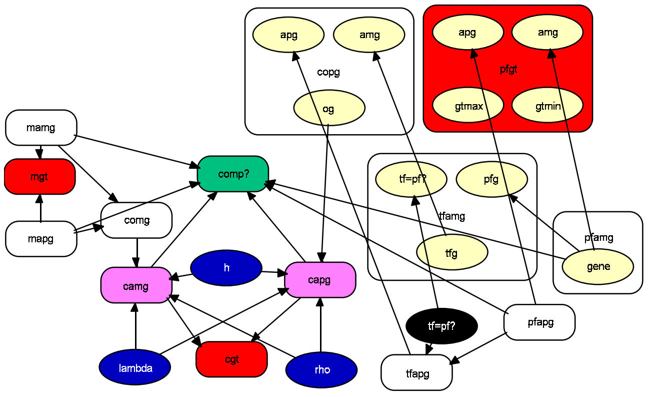

Figure 2: Network **comp6** with nodes `pfamg`, `pfgt`, `copg` and `tfamg` expanded.

In any network we only need to specify probabilistic "parent-child" tables for the oval nodes, since the structure of the instance nodes is determined elsewhere. In the present case, these are `lambda`, `rho`, `h` and `tf=pf?`. As these have no "parents", we just need to specify their prior distributions; and as we shall mostly be concerned with setting values or extracting likelihoods at these nodes, we use uniform priors over a suitable discrete set of values. For `lambda` we use 43 values between 0 and 0.01, and for `rho` and `h`, values $(0, 0.5, 0.7, 0.9, 1)$; node `tf=pf?` is binary, with values 1 and 0 to represent 'yes' and 'no' respectively.

[2]To avoid confusion with genuine biological relationships, we use quotes when these terms are used with their generic network interpretations.

2.2 Lower-level networks

In Figure 1, nodes `mamg`, `mapg`, `pfamg` and `pfapg` are all instances of the class **founder**, represented by the trivial network of Figure 3. This contains a single node `gene`, specifying the distribution of the alleles for the chosen marker in the relevant population. Table 2 shows this distribution for the marker VWA. This information thus needs to be entered only once. The grey edging to the node `gene` signifies that it is an interface node, and the solid boundary that it is in fact an output node; input nodes have dashed boundaries.

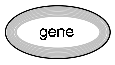

Figure 3: Network **founder** for founder gene

Table 2: Allele distribution for marker VWA

allele	frequency
12	0.0003
13	0.0018
14	0.1009
15	0.1004
16	0.1949
17	0.2834
18	0.2162
19	0.0866
20	0.0137
21	0.0015
22	0.0003

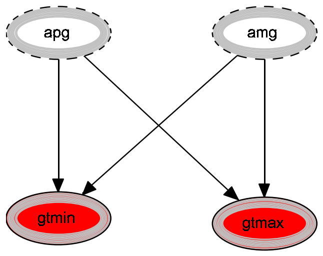

Figure 4: Network **gt** for genotype

Nodes `mgt`, `pfgt` and `cgt` are all instances of the class **gt**, as shown in Figure 4. This destroys the information on the origins of the parental alleles by forming `gtmin` := min(`apg`, `amg`), `gtmax` := max(`apg`, `amg`). These and later relations use the straightforward "expression" syntax of HUGIN, and can be simply entered using its "expression builder" facility.

Nodes `comg` and `copg` are instances of the class **cog**, as shown in Figure 5. This models Mendelian segregation by regarding a child's original (*i.e.* unmutated) gene `og` as chosen at random from the two actual parental genes, `apg` and `amg`, according to the outcome of a fair coin toss `fcoin`. That coin toss is itself an instance of the trivial class **faircoin** shown in Figure 6, containing a single binary node `coin` with prob(1) = prob(0) = 0.5. We thus define `og` := if(`fcoin.coin` == 1, `apg`, `amg`). Here the notation `fcoin.coin` refers to the internal node `coin` in the network class **faircoin** to which the instance node `fcoin` belongs.

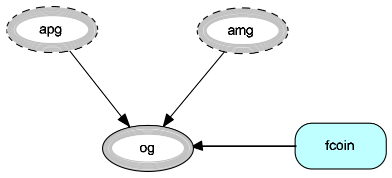

Figure 5: Network **cog** for Mendelian inheritance

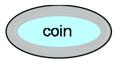

Figure 6: Network **faircoin** for fair coin toss

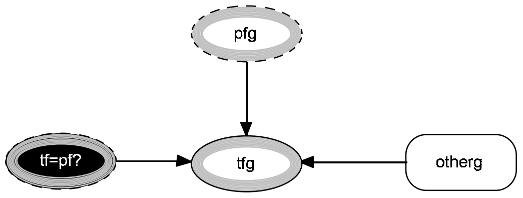

Figure 7: Network **query** for provenance of paternal gene

Nodes `tfamg`, `tfapg` are instances of **query**, as in Figure 7. Here `otherg`, an instance of **founder**, represents a gene contributed by another random member of the population; and `tfg` (true father's gene) is either `pfg` (putative father's gene), or `otherg`, according as `tf=pf?` (true father is putative father?) is 1 or 0: `tfg` := if(`tf=pf?` == 1, `pfg`, `otherg.gene`).

Nodes `camg` and `capg` are instances of class **ag**, as in Figure 8. This models the process whereby an original gene `og`, as contributed by a parent, is possibly

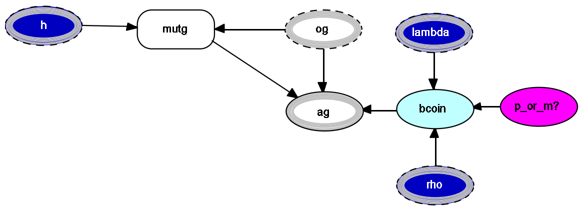

Figure 8: Network **ag** for actual gene

altered by mutation to produce the actual gene `ag` received by the child. We take `ag`:= if(`bcoin` == 1, `mutg.mutg`, `og`), so choosing either the mutated or the original gene according to the outcome of a biased coin-flip, `bcoin`, whose bias is determined by the parameters `lambda` (approximate overall mutation rate) and `rho` (ratio of male to female mutation rates), with the externally specified value of `p_or_m?` deciding the appropriate value: `bcoin` := Binomial(1, 2 * `lambda` * (`rho` * `p_or_m` + (1 - `rho`) * (1 - `p_or_m`))).

The node `mutg` in **ag** is an instance of class **mutg**, as given in Figure 9. This uses a biased coinflip `bcoin`

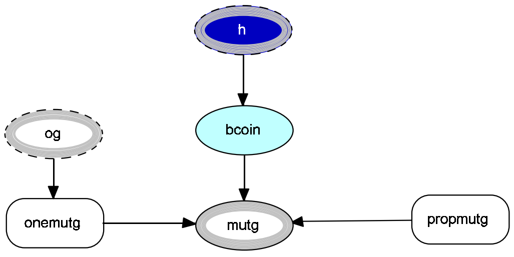

Figure 9: Network **mutg** for result of mutation

:= Binomial(1, `h`) (where `h` is externally specified) to mix over two different models for the mutation process; the *proportional model* of `propmutg`, which is an instance of **founder**, simply forgets the original gene entirely, and resamples from the gene-pool; and the biologically more realistic *one-step* model `onemutg`, an instance of class **onestep** (Figure 10), in which we take `onemutg` := if(`og` == 12, 13, if (`og` == 22, 21, `og` + `faircoin.coin`)). This flips a fair coin to change the allele value by ±1, ignoring any out-of-range transitions.

Under the proportional model, there is a non-zero probability that the result of "mutation" is no change. This is accounted for by the following relationship between the parameter $\lambda \equiv$ `lambda` and the true mutation rate μ:

$$\mu = \left\{(1-h)\left(1-\sum \pi_i^2\right)+h\right\}\lambda, \qquad (1)$$

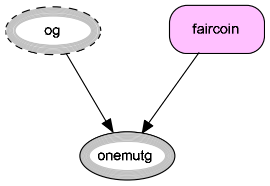

Figure 10: Network **onestep** for one-step mutation model

where the (π_i) are the relevant allele freqencies (as given for VWA in Table 2).

Finally in network **comp6**, node `comp?` describes the compatibility status of the genotypes of the triple. It is an instance of class **compat** (Figure 11), in which

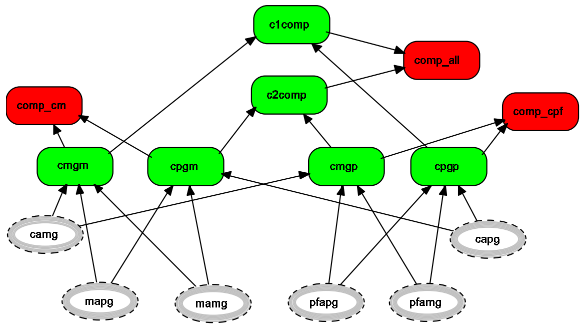

Figure 11: Network **compat** for compatibility

`cmgm`, `cpgm` and `cpgp` are themselves instances of class **nuc_compat** (Figure 12), which indicates the compatibility of individual actual genes of the putative father, mother and child: `comp?` := or(`cag` == `apg`, `cag` == `amg`). Then `comp_cm` [resp. `comp_cpf`] (instances of a trivial 'or gate' class **or**, as shown in Figure 13, having `out` := or(`in_1`, `in_2`)) indicate compatibility of c with m [resp. pf] in the absence of data on the third individual; while `comp_all` (itself an instance of **or** applied to auxiliary nodes `c1comp` and `c2comp`, instances of the unshown trivial 'and gate' class **and**) indicates compatibility of all three genotypes. These

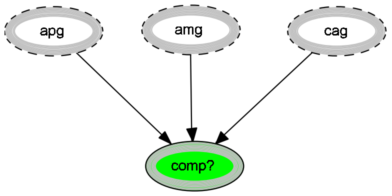

Figure 12: Network **nuc_compat** for simple compatibility

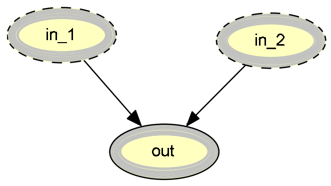

Figure 13: Network **or** for 'or gate'

nodes are used for input of data when only compatibility is reported.

Note that if **comp6** were regarded as a simple network, in the associated junction tree the clique containing `comp?` would have 7 nodes, and — if we take `comp?` itself as binary — $2 \times 11^6 = 3543122$ entries. In addition, the effects of moralisation and triangulation involving "parents" of `comp?` would create further large cliques — one of these, with 22 members, being of size 7412493. In all, the total clique-table size in this case would be 10981777, leading to serious computational inefficiency.

However, with our internal structuring of node `comp?`, there are (after moralisation and triangulation) 8 cliques involving its constituent nodes, having a maximum clique-table size of 234256 and total of 476328. The total clique-table size for the whole network is reduced, by a factor of about 13, to 828739, and computation now proceeds speedily and efficiently.

3 Application

The use of the network **comp6** to construct likelihoods for inference about mutation rates will be described in detail elsewhere. Briefly, at marker VWA there are 4 incompatible triples in our dataset, for which we have complete information on their genotypes; as well as about 1000 triples (or occasionally pairs, with one parent missing), known only to be compatible. For each case other evidence, including the data on other markers, gives a high prior probability of paternity. For each case we set `capg.p_or_m?` := 1, `camg.p_or_m?` := 0, and set desired values for h, `rho` and prob(`tf=pf?` = 1). We then enter the relevant data, and use the software to "propagate", so updating all probabilities in the network. Interrogating the node `lambda` now gives the appropriate contribution, for that case, to the overall likelihood function over the values specified — very good approximations to the continuous likelihood function can also be calculated. Finally, we conduct a sensitivity analysis to the specification of `rho` and `h`.

As an illustration, suppose that we set `h` to 0 (proportional mutation model), `rho` to 0.5 (equal mutation rates in both germ-lines), and, for a certain case, the prior probability of paternity to 0.97. We now enter the evidence that the triple of genotypes is incompatible. On propagating using the HUGIN software, interrogating `lambda`, and adjusting using (1), we find that the associated contribution to the likelihood function for the overall mutation rate μ increases, in an essentially linear fashion, from 0.0180 at $\mu = 0$ to 0.0295 at $\mu = 0.008$

Acknowledgements

This work was supported by the Leverhulme Foundation and the Gatsby Charitable Trust. I am grateful to Hugin Expert A/S for making the HUGIN 6 software available, and to Paola Vicard and Julia Mortera for helpful discussions.

References

Brinkmann, B., Klintschar, M., Neuhuber, F., Hühne, J., and Rolf, B. (1998). Mutation rate in human microsatellites: Influence of the structure and length of the tandem repeat. *American Journal of Human Genetics*, **62**, 1408–15.

Cowell, R. G., Dawid, A. P., Lauritzen, S. L., and Spiegelhalter, D. J. (1999). *Probabilistic Networks and Expert Systems.* Springer, New York.

Dawid, A. P., Lauritzen, S. L., Mortera, J., and Vicard, P. (2002a). Assessing mutation rates for forensic markers. *Abstracts of the Seventh Valencia International Meeting on Bayesian Statistics*, 69. Available from
`ftp://matheron.uv.es/pub/personal/bernardo/Booklet.pdf`.

Dawid, A. P., Mortera, J., and Pascali, V. L. (2001). Non-fatherhood or mutation? A probabilistic approach to parental exclusion in paternity testing. *Forensic Science International*, **124**, 55–61.

Dawid, A. P., Mortera, J., Pascali, V. L., and van Boxel, D. W. (2002b). Probabilistic expert systems for forensic inference from genetic markers. *Scandinavian Journal of Statistics.* To appear.

Henke, L. and Henke, J. (1999). Mutation rate in human microsatellites. *American Journal of Human Genetics*, **64**, 1473.

Koller, D. and Pfeffer, A. (1997). Object-oriented Bayesian networks. In *Proceedings of the 13th Annual Conference on Uncertainty in AI (UAI)*, pp. 302–13.

Sajantila, A. (1999). Experimentally observed germline mutations at human micro- and minisatellite loci. *European Journal of Human Genetics*, **7**, 263–6.