Automatic generation of preliminary genealogies using the UK's local BMD registers

Stephen Bush

Cite as: Stephen J. Bush, "Automatic generation of preliminary genealogies using the UK's local BMD registers," *Annals of Genealogical Research*. Vol. 5, No. 1 (2009) at http://www.genlit.org/agr/viewarticle.php?id=27.

Family trees that date back to the early 19th century can be assembled with surprising ease, thanks to the availability of vital records. This article presents a novel means of assembling 'outline' family trees, by automatically searching, extracting and presenting information from the birth, marriage and death registers.

An Introduction to the BMD databases

By UK law, civil registration of births, marriages and deaths (BMD) commenced in England and Wales from 1 July 1837, in Scotland from 1 January 1855, and in Ireland from 1 January 1864, although in this latter case, an earlier registration date of 1845 is applicable for marriages alone[1]. BMD databases provide a significant body of information, which is increasingly being made available electronically (www.ukbmd.org.uk).

The standardised format of each database entry renders the entire BMD database amenable to searches as conducted not by a human but by a program designed to implement a particular pattern for doing so (and structure the output in 'family tree' format accordingly). The motivation to accomplish this for BMD records is quite recent - specifically, it is reliant upon the provision of *local* BMD records. There are actually two sets of birth, marriage and death indexes in the UK: the original indexes as held by the local register offices and a secondary index created by the General Register Office (GRO). Quarterly, these original indexes were re-transcribed and then re-indexed by the central government, and as such, an element of data loss is applicable - errors of transcription or omission are applicable in the secondary, GRO, records. It is always preferable, therefore, to search the local indexes[2], although as a consequence of this, local interest must be established and maintained in the respective areas to even ensure they are made widely available in the region they cover. Volunteer-led projects under the umbrella term of 'LocalBMD' in nine areas of the UK - Bath, Berkshire, Cheshire, Lancashire, North Wales, Staffordshire, West Midlands, Wiltshire and Yorkshire - are transcribing the respective hard-copy local registers into BMD databases to be made available online. With the exception of Bath, whose BMD registers

[1] This article, will, however refer only to automated searching of the BMD databases of England and Wales, given that only the local copies of these records are being made freely available online, albeit on a restrictive county-by-county basis at present. See main text for further details, noting that the principles here outlined may work on data from all sources, providing the format is consistent throughout. Currently, Scottish and Irish records remain distinct from the UK BMD databases, and as such, their access also differs. Scottish registration indexes for births from 1855 to 2006, marriages from 1855 to 1932 and deaths from 1855 to 2006 can be searched online at http://www.scotlandspeople.gov.uk/ on a pay-per-view basis. For records of registration in Ireland and Northern Ireland (after 1922) there are currently no online indexes. These records are instead held in the General Register Offices in Dublin and Belfast.

[2] Local records have only recently become publicly available not only online, but actually to the public in the relevant locality. Prior to this, the secondary source - the GRO records - were the only BMD registers permissible to public scrutiny; whilst microfiche copies were available locally, the hard copies themselves were originally located only at Somerset House, London, until they were moved in 1972 to St. Catherine's House and finally in 1997 to the Family Records Centre.

are now complete, all remain ongoing.

It is proposed that automated data mining of a set of BMD databases, of standardised format, will provide an effective and convenient means by which preliminary genealogies may be determined. A program, BMDTreeGen, has been written to this end; its usage, benefits and limitations will accordingly be discussed. However, the benefits of such a program, in principle, are twofold: (a) as a guideline to further research, which may be prompted in part by checking the validity of this program's output, and (b) as an effective means by which the home user may add detail to the 'branches' of their own family tree. This latter is of use should the meticulous perusal of the records necessary to do so not have quite the emotional connection or appeal that originally inspired research into their family name or bloodline; an immediate application of this is in the lines of descent that led to uncles and aunts by marriage.

Such a standardised format of the UK's BMD databases, as required for automated use, does not exist in all cases, being restricted to the nine large-scale LocalBMD projects thus far attempted; however, independent volunteer projects are at least systematic in devising and maintaining their own data structures in collaboration with local register offices, even if those are not necessarily adopted by volunteers transcribing the BMD indices of a neighbouring county - not all local indexing falls within the remit of the LocalBMD project, after all. In order to devise suitable software, an extensive and complete set of existing BMD databases was searched for, regardless of their size in comparison to larger, though unfinished, projects. It is perfectly feasible that these smaller-scale projects can fit seamlessly into larger-scale efforts to index the national record set (the Digitisation of Vital Events [DOVE] Project, for example), of which these databases represent a part. Even if this ultimately proves untrue, the BMDTreeGen tool herein presented, created with small, complete databases in mind, may still garner some use amongst those with local interest in BMD records.

Databases generated by the Bath BMD (www.bathbmd.org.uk) transcribing project were chosen to represent the practical example with which BMDTreeGen is to be used. While the principal tenets of this program can be applied to databases of theoretically indefinite size, in practical terms, the data format employed was ready-made and thus the database used is limited to those indexes which have been transcribed in obedience with it. However, it must be noted that the Bath BMD was chosen for its robustness, for advantageous choices made by the transcribing team (described below) and the fact, above all, that as a project, the Bath BMD is complete[3]. Thus, BMDTreeGen represents a data mining utility whose databases, although limited to the Bath district, nevertheless represent a complete set of the covered area and hence the principle reason why any record may not be found in any of the local BMD databases (that it hasn't been transcribed yet) is removed. The same search criteria as used by any Internet user at www.bathbmd.org.uk, and the layout of any information obtained from online database access can be easily mimicked for a downloaded copy of the entire Bath BMD. As such, the search process can also be readily automated. In addition, it is easier to fit the task of computer programming to the existing data format, than enforce a novel one upon an already enormous series of databases.

How do you derive family relationships from BMD data?

The convenience of online database searching may ultimately promote the growth of more powerful tools still: programs designed to facilitate automated family tree generation. A database searchable by computational methods is one that can also be searched automatically by the same. BMDTreeGen employs the same methodical approach that any human would bring to extracting meaningful relationships given the BMD databases.

For instance, from a human perspective, to determine your ancestry given your own name as a

[3] The Bath BMD announced the complete transcription of all locally-held registers in February 2008, with a final tally of 490,498 birth, 175,565 marriage and 347,252 death records.

staring point (and assuming you are initially ignorant of all but this), you would have to conduct the following 'bottom up' searches of the B, M and D databases in turn.

The human search protocol, ultimately adaptable into BMDTreeGen, is as follows:
1. Search B for your name, knowing a suitable range of years with which to narrow down this initial enquiry. The record thus obtained will state your mother's maiden name. An initial accurate record is all that is required to start off the search process - the 'suitable range of years' within which to look for ancestral records is otherwise based on various existing assumptions. For instance, given marriage year *x*, most assume the birth of the two individuals listed must surely fall within the years *x* minus 50 and *x* minus 20 and are typically similar to each other.
2. Search M for the marriage record of your father and mother, using as your search criteria your surname (which is also that of your father) and your mother's maiden name, obtained in the previous step. It is assumed that your parents were both married and that their marriage is no later than the year of your birth (given that it is only in relatively modern times that unmarried couples bore children). The record thus obtained will state both your father's and your mother's forenames.
3. Search B for your father's name, given that his forename is now known. When this methodical search pattern is adapted to BMDTreeGen, this requires boundaries for a 'search window,' derived from the oldest age at which a couple may marry - one of BMDTreeGen's initial questions to the user - and the known year of marriage itself[4]. The record thus obtained will state his mother's maiden name.
4. Search B for your mother's name, now that both her fore- and maiden name are known. The record thus obtained will state her mother's maiden name.
5. Search B for (other) children to this marriage, using as search parameters the child's surname (as belonging to the father and obtained from the marriage record) and mother's maiden name. It is assumed that all children are born after the parent's marriage date. Crucial to the success of this particular search is knowledge of the mother's maiden name.

With the surname and mother's maiden name obtained in both steps 3 and 4, step 2 - a search of the marriage database - can be repeated to continue this process one generation further back in time. The 'search window' stipulated in step 3 can be narrowed down with regard to ancestors by searching the death database at any time after both their fore- and surnames are known, accepting that married women may die with a name different to that in their birth record. The age at death thus listed in either case can be reconciled with the year of marriage and allow for some calculation as to the year of birth, accepting the following: (a) in many instances, the age of death represents an estimate as made by whomsoever pronounces death, and (b) each database notes the year of the particular event and not its specific date; thus, dependent upon the calendar position of the event itself as relative to the day the register was updated, an error margin must be introduced of, at minimum, 1 year upon either side of any year calculated from this figure.

To translate this search protocol from human-driven to automatic, it's necessary to note two important points. Firstly, there are various stages where assumptions must be made by the human user, although these must be made specific (and therefore restrictive) for an automatic approach. Secondly, this schema can be conveniently inverted in its entirety to allow "top-down" searching along much the same lines, given as starting point the oldest ancestral name. When

[4] In BMDTreeGen, should this search window be too broad - i.e. if accepting of surprisingly late marriages (usually a more modern occurrence) - then the consequently high value of 'oldest age at marriage' will ensure BMDTreeGen searches for birth records *earlier* (given that this parameter is used to calculate the lower bound). There is a greater risk, therefore, of BMDTreeGen erroneously selecting the first record meeting the required name within that search window. This is not a problem that can be avoided, nor does it represent a lapse of logic in BMDTreeGen's programming - rather, caution can only be given as to too broad a search of years within which people marry, and from the perspective of the program itself, the number of multiple birth records bearing the same name within the search window (if any) are noted in a separate report file: the user must be aware of this margin of error.

this protocol for database searching is rendered automatic by BMDTreeGen, all human assumptions must be instead incorporated as parameters, and therefore defined from the outset by the user (these shall be detailed later).

What are the pros and cons of using BMDTreeGen to automate this database search technique?

Pros: speed, convenience and exhaustiveness.

The search protocol detailed above incorporates many steps for a human analyst; for rapidity and convenience, it is necessary to automate this. Moreover, it may be argued that the most significant obstacle to be overcome in genealogical data mining is actually tedium; the work done in implementing this protocol is of a repetitive and uninspiring nature, given it is the outcome alone which spurs the anecdotes and stories that are the larger goal, of family history compilation. To automate this, BMDTreeGen requires the input of one starting name only yet ultimately yields as output both genealogical data and a means by which each record is related to each other. It is the logical continuation of the work done in having made available large electronic databases: to parse them with a suitable methodology.

BMDTreeGen's approach possesses an additional novelty in the respect of operating, unlike the human ‘bottom-up’ database search previously introduced, from the top down. Whilst it is common practice to work backwards through ones ancestors, and stop only when the relevant records do, it is thought preferable to instead identify the oldest potential ancestor available in the databases, thus permitting an exhaustive 'at all levels' search[5]. This is not only computationally more convenient, it guarantees of any database search that if no matching records are found, the generational level at which cessation of automatic tree generation occurs is plainly identified - one can manually correct or otherwise add records, so as to continue working downward (i.e. towards the present day).

By contrast, adding an individual 'up' from the last identified level necessitates moving both ways in time - both towards an older ancestor (i.e. towards a region characteristically less documented still) and, potentially, to follow an additional downward branch. For instance, if you add a parent to a genealogy’s current oldest individual, you tacitly admit to the existence of that person's family, which necessitates searches for siblings and therefore, many potential downward branches to explore, should any be found. Searching from the top down bypasses the myriad confusions of simultaneous 'up' and 'down' searching, whilst ensuring progression of the tree is made always towards the modern day - where records are typically fuller, more accurately annotated and more readily available. Alongside this profusion of modern-era records, one must be aware of the fact that there are increasing levels of both individual and family migration. As the current UK BMD records are restricted to specific geographical regions, it becomes the case that if searching for a specific name, it is easier to go from an older record to a modern one than from any modern record to an ancestor. Being a late 20th century Smith in Bath (for instance) is no guarantee that your ancestors were also resident in the area. Modern-era records are characteristically full of individuals unlinked to a chain of previous generations, having migrated to the area only relatively recently, and it is folly to pick modern names of interest and attempt to derive their ancestry automatically, unless you have prior knowledge that this will be worthwhile. Any persons for whom a suitably aged ancestor cannot be found will regrettably not find use with BMDTreeGen - their ancestors are not resident in this geographical area; the indexing of BMD records for another locality will be necessary in their case.

This also best illustrates the principal disadvantage to BMDTreeGen, although this disadvantage

[5] A supplemental program, ancestor_finder, has been written to automatically identify the oldest potential ancestor to a query name. This is of particular use if you wish to derive a genealogy starting only with a recent name (i.e. your own). See the sub-section "if BMDTreeGen works from the top down only, how do I know which record to start from?"

is true of human-driven database searches also: that as the databases vary in completeness of content and accuracy with regard to any given family name, the scenarios in which you can search them must be specific. This is true of human-led searches; however, it is much more so for any automated process.

Con: the scenarios within which BMDTreeGen best operates must be specific.

Computer-driven searches cannot be broad, and they can only be as flexible as you allow them to be. As such, transforming the 'common sense' search protocol, above, into BMDTreeGen requires that the user firmly specify any assumptions.

The practical usage of BMDTreeGen requires setting values for both the youngest and oldest age at which a person may marry, and the time period post-marriage within which children are produced by the union. These boundaries are imposed by necessity, given the nature of the databases: many records exist with the same name; how is a person, much less a program, to distinguish between what is possible and what is actually plausible? Conveniently, legal limits exist upon marital age, which provides a lower boundary[6], but although it's not unheard of that people may (re)marry at an advanced age, the likelihood of automatically identifying the correct spouse if the 'search window' for the marriage database is too broad diminishes sharply. There is a significant chance that a younger person with the same name is erroneously selected as the correct spouse; BMDTreeGen must exercise the same degree of caution a human may, and respect a 'middle ground' chronology. Naturally, however, the sharp clamping down upon extremities may cause the program to simply discard otherwise possible (albeit unlikely) results, suggesting that human analysis of BMDTreeGen's output is necessary, thus obviating the need for BMDTreeGen anyway. In response to this, it can only be re-iterated that BMDTreeGen rapidly provides a framework within which much more detailed, human-driven, analysis should be performed.

How does BMDTreeGen present its findings?

BMDTreeGen assigns an identification code to each record that it extracts from the B, M and D databases. It is intentionally simplistic, although aims to be robust and providing of an at-a-glance representation of familial relationships[7]. In essence, it is constructed of alternating letter-and-number pairs. The initial query name is given the character A; the first spouse of this person, if found, is given both a copy of the last code and an extension to it - the number 1 appends their code *with their relationship to the last person seen*, i.e. John Smith, the query name, has code A, and his wife, Jane Doe, has code A1. Should John remarry, his second wife would have code A2, and so forth. Smith's children (by spouse number 1) are given letter codes, in order to append the current code still further. In order of age, Arthur, Betty and Charles are A1A, A1B and A1C, respectively. John's children by a second wife, should there be a second wife, will be A2A, A2B, and so forth. The pattern emerges with little effort: spouses of Arthur, Betty and Charles will be A1A1, A1B1 and A1C1, and the firstborn child of each union will be referred to as A1A1A, A1B1A and A1C1A. The pattern of the code is always one of pairs - take the following example, broken up into its constituents and accepting that if the latter character has no trailing number, it

[6] In 1929, it became illegal for people under 16 to marry; prior to this, girls could marry at 12 and boys at 14, although in both cases, parental consent was required if they were under the age of 21 (reduced to 18 in 1969). Naturally, people don't respect rules as clearly as computers do. Falsified records, minors marrying with parental consent (which won't be noted in a simple database, even if true) and unregistered marriages abound, to complicate the issue. Admittedly, this is to confuse the matter for a human. A computer program is set to ignore it; the human is to be aware of this!

[7] It is the at-a-glance aspect that was focussed upon. Although computer methods may lend themselves more immediately to genealogical numbering systems such as the Ahnentafel, the fact that any genealogies derived from the BMD databases can only extend, at the earliest, from 1837, suggests that a far less complicated system can be much more satisfactory.

represents that there is no known marriage. The codes for each individual determine their relationship to other members of the genealogy:

Code: A1B2A1C.
Interpret the code as follows: [A1] [B2] [A1] [C].
Read from right to left: This is the third child (C) born to the first spouse of the eldest child (A1) of the second spouse of the second child (B2) of the query individual and his/her first spouse, i.e. John and Jane (A1).

In the Smith lineage, as could be obtained by BMDTreeGen, two more points can be determined from examining these codes. Firstly, and quite neatly, the generational number of each individual in the lineage is equal to the number of letters in their code. John, the founding member, has a one-letter code, A. His children, generation 2, have two-letter codes, and his grandchildren have three-letter codes, and so forth. Secondly, any individual whose code ends with a letter is a member of that lineage by blood; any whose code ends with number is a member by marriage. In the latter instance, as males do not marry *into* a lineage but are traditionally only *connected* to it, the children must be sought for in a new lineage, using the code of this male as a query for a new search by BMDTreeGen. For instance, Betty Smith, above, bearing code A1B marries John Jones, A1B1. As we know Betty to be female, she may have children, but these will not appear in the Smith lineage - they will be children with the surname Jones. A1B1A and A1B1B, for instance, will *only have these codes in the Smith lineage if they have the surname Smith*. We know from marital tradition that they will not; as such, BMDTreeGen will not find them. We must instead, upon inspection of BMDTreeGen's output and realising that A1B1 is a male name, start a fresh search using John Jones as the query name. Only in this respect can Betty's children be found, albeit that they are now members of the Jones lineage.

From a computer programming perspective, it is necessary to explore the 'depths' of each generation and so exhaustively search from any given record. Finding John and Jane's children, Arthur, Betty and Charles, gives you three lines of enquiry to pursue - it is necessary for BMDTreeGen to explore *every* branch descending from Arthur before returning to a 'basal level' and repeating the process with Betty and Charles. It is the only feasible way to ensure rigour in the search protocol; however, its side-effect is that the lineage thus created is not immediately neat to the human eye; grouping by generation must thus occur manually. BMDTreeGen will of course list all records in the lineage, and provide codes to link them together by immediate relationship (explained above), but this rigorous search protocol means that Arthur's great-grandchildren (for example) will be listed before Betty's children since BMDTreeGen explores the one line of descent before the other. It poses an unnecessary and complex programming task to automatically construct a lineage for human reading convenience when BMDTreeGen's primary goal has always been to construct a lineage with the convenience of speed. Regardless, having a distinct code for each individual allows the output file to be re-organised without any loss of integrity. Furthermore, irrespective of relationships within each generation, the generation number applicable to each individual – in essence, the 'search depth' – is listed just before that person's identification code on each output line.

How does BMDTreeGen cope with the fact that the B, M and D databases are old, incomplete and error-prone?

Before BMDTreeGen can be tested, it is necessary to discuss the original data from which online databases have been compiled. Given that it is by definition antiquated, and in being digitized now succumbs to an additional transcription stage, the B, M and D databases as used contain abundant sources of error. If BMDTreeGen automatically searches databases, it is only as good as the database content – as such, what are the scenarios BMDTreeGen cannot accommodate?

What BMD database entries are not amenable to automated searching?

Certain scenarios may arise when interpreting the available information in the B, M and D

databases. However, some cannot yet be accommodated by BMDTreeGen - without a human analyst to search the BMD databases, no computer program yet possesses the wherewithal to 'imagine' creative solutions to these problems. As such, records of the following type will simply not be found when the stepwise search protocol, defined above, is undertaken automatically, and thus caution must always be taken with the program's output. As yet, no truly efficacious replacement exists for the human element, nor a substitute for the imagination required in the interpretation of aging records, which are often increasingly inaccurate, if they exist at all. These complications are those of *poor* or *confusing* data; note that the most pressing problem of all is, however, *absent* data (discussed in the following section). The following searches will fail in all cases if automatic methods are employed, either because (a) what is requested violates an implicit assumption made by BMDTreeGen, which - if made permissible in all cases - will increase the number of 'false positive' records to an extent greater than that of these 'false negatives' or (b) the programming of BMDTreeGen is insufficient for the task. It is hoped that the latter cases are much fewer; however, they are not negligible and will be noted below.

- No records will be found if exact spelling is not maintained between birth, marriage and death records; this point is most prominent and a significant cause of inadequacy in BMDTreeGen's output. Given historical levels of illiteracy and the fact that records were not self-maintained but that clerks recorded dictation, often from strong regional accents, confusion over the spelling of ones name (not an uncommon occurrence) often compelled clerks to make an educated guess as to what it was. Aside from the additional possibility of transcription errors arising in the transition between paper and digital copies, it was also not an uncommon practice for people to be known by their middle name (or something else entirely); as a consequence, relative to the order of names on the birth certificate, marriage and death records occasionally show an inversion, taken by this author to mean the person's preference for their usage (the birth records only shows the parent's preference as to the child's name).
- Birth records will not be found of children born to unmarried parents, even if they later marry.
- Records of death will not be found where one member of a married couple, or records of birth for any children born to that union, double-barrel their surname.
- Birth records will not be found where a child does not take the father's surname.
- Death records will not be found where a married woman dies after having reverted to her maiden name (i.e. cases of divorce).
- Multiple marriage records (i.e. bearing the same reference number) may be found for the same marriage if the female has been married previously; in this instance, separate records may exist for her new marriage using both her maiden and her former married name. BMDTreeGen will not, however, count these separate records of the same person having married as two distinct marriages for that person; instead, a note is made in the output log as to the potential previous marriage of the spouse, for future enquiry.
- Marriage records will not be found where a person re-marries before his/her former spouse(s) death record(s) are found. Given that later marriages occur either after death or divorce and the BMD databases cannot account for the latter, it is a necessary, but knowingly flawed, assumption of BMDTreeGen that death is required before further marriage records may even be considered for the remaining party.

What BMD database entries are absent?

In several situations, BMDTreeGen will fail to compile an accurate genealogy as the required information is simply not present. The BMD databases are not, nor is it likely that will ever be, wholly accurate. There are several reasons for this, some of which are listed below:

- Although mandatory civil registration commenced in England and Wales in 1837, the onus was actually on the parish registrar, rather than the parent, to update the relevant records; it was only in 1874 that fines were introduced for non-compliance with civil registration, so it is expected that until this time, disinterest is responsible for pronounced

gaps. Furthermore, children not registered within six months of birth could never be registered, and a state of confusion with regard to whether un-baptised children could be registered may account for the fact that a sizable proportion of the birth database is non-existent.

- Divorce was practically impossible for the poor, so it was often the case where people wishing to do so simply left their spouse and established a new family with someone else; in these instances, they would have claimed marriage to their new partner upon the birth record for any children but no marriage certificate could ever be found.
- Rather prominently, a certain proportion of deaths during wartime tend not to occur locally: male records of the conscript generations are not maintained at home.
- The 1926 Legitimacy Act permitted illegitimate children to be re-registered upon the subsequent marriage of the parents; before this time, illegitimate children were always registered with the mother's surname.

How do you install and use BMDTreeGen?

BMDTreeGen is made available as a script, a single file called tree_gen.pl, written in the programming language Perl and thus requiring for usage local installation of the latest release of ActivePerl, version 5.10.0, available freely from http://www.activestate.com/activeperl/. After installation, BMDTreeGen may be run from the command line simply by switching to the directory the tree_gen.pl script has been placed in, typing "perl tree_gen.pl" (omitting quotation marks) and following the on-screen prompts to define search parameters. Output will be in two files. One will be a comment file, report.txt, which lists any and all complications arising from BMDTreeGen's search-and-create-tree protocol. (The presence of multiple records bearing the same name, the presence of multiple marriages with the same reference, etc., all of which may provide valuable information for later, human-led, searching.) The other file will be an automatically constructed genealogy itself. This will be in plain text format, in a file named after the surname of the initial query, the oldest known ancestor of that lineage. This ancestor must always be male; it is for the user to know this - BMDTreeGen cannot obtain this information. As the BMD databases themselves do not state the gender of each record, and as the assumption has always been that the children of a marriage take the father's name, BMDTreeGen's search protocol is always to search, for instance, the birth database of the letter H for the names of children born to the union of John Hancock and Jane Smith, accepting that the latter name is obtained in the previous step of the search scheme (i.e. obtain a marriage record if the name of the spouse is known). While this necessitates separate searches for the children of married female children in the query lineage (e.g. Hancock), these children would bear a different surname by marriage and fully justify such a separate search anyway.

Finally, BMDTreeGen requires a copy of the databases within which it is to search; all three databases - for births, marriages and deaths - are downloaded letter-by-letter from www.bathbmd.org.uk, and stored in subfolders Births, Marriages and Deaths, respectively, of the folder Databases[8]. This folder must be present in the directory from which BMDTreeGen is run. Each file, representing every entry by surname initial, must be in CSV (comma-separated value), rather than plain text, format. This allows for the file to be opened in spreadsheet format, but more pertinently, the inclusion of commas provides a means by which BMDTreeGen can distinguish between different data-types when the program reads each record; this is vital for the program's success. Further to the discussion of complicating factors with regard to the BMD database content itself, it is sometimes necessary to edit particular entries if obvious misspellings or inaccuracies exist. The databases thus used diverge slightly from those made available online by the BathBMD group; however, this is only so for the required accuracy (the original databases

[8] Each database is, quite simply, a plain text file: A.txt, B.txt, C.txt, etc. This is of relevance if users of BMDTreeGen wish to download these resources directly from the BathBMD website (the databases constitute a sizable amount (~16Mb) of plain text, and copies are not hosted by the *Annals of Genealogical Research*).

are also provided, as is a list of the errata, for comparative purposes).

The **online supplementary material**[9] for this article comprises BMDTreeGen and example output, for reference alongside the discussion below. This output represents sections of family trees for illustrative purposes, all as created by BMDTreeGen using the same query name but different search parameters. Note that as the BMD databases extend from 1837 to the present day, BMDTreeGen will create genealogies that can easily incorporate people alive today. The example output of BMDTreeGen as supplied has been edited in respect of this, thus represents only a sample of older content. More importantly, however, the data format is consistent throughout; this intentional editing does not detract from its illustrative purpose.

Additional resources are not hosted online due to space constraints. The databases themselves, essential for the use of BMDTreeGen, may be obtained by either contacting the author or directly from www.bathbmd.org.uk, whilst following the instructions above as to how they are to be organized into folders[10].

If BMDTreeGen works from the top down only, how do I know which record to start from?

A supplemental program, ancestor_finder.pl, has been created to identify the oldest possible ancestor to any query individual, by implementing a basic search protocol. This (a) finds the birth record of the query individual and identifies the mother's maiden name, (b) searches the marriage database for marriages of the provided surname and the newly known maiden surname, and (c) searches the birth database for the ancestral parent and repeats this process. Ultimately, the output of ancestor_finder will be full names and birth years (if available) for a chain of parents[11]. This provides all the necessary information to use BMDTreeGen in its 'top down' capacity, as described above. Its operating instructions are no different from that of BMDTreeGen, and the same cautions must be applied with regard to the integrity of the databases (see "How does BMDTreeGen cope with the fact that the B, M and D databases are old, incomplete and error-prone?", above). Inherent flaws in the available data will become apparent in the output.

Testing BMDTreeGen

Given that BMDTreeGen is designed to impose rapid order upon imprecise data, its usage should only be promoted given the convenience of this speed. It is cumbersome and slow for a human analyst to pore over the databases and imagine where gaps may be filled, given poorly transcribed, misspelled or absent records; BMDTreeGen will succeed in automating family tree generation only if it does not introduce additional errors upon these. Similarly, its functionality can be apologised for if it cannot circumvent the problem of poor quality data (primarily, misspellings of certain records - these will simply not be identified by the program, unless a 'sounds similar' algorithm is introduced); however, this aside, caution must always be exercised in the usage of BMDTreeGen. How accurate it is with regard to an inherently flawed dataset can only be tested in comparison to a family tree derived by human examination of the databases. However, BMDTreeGen responds favourably; its automatically created genealogy is strongly in line with that obtained manually. Errors are few, but are typically inherent in the database content themselves.

The comparison of mistakes will further explore how they arose, and either vindicate the use of

[9] These materials may also be obtained directly from the author, by emailing sjbush@gmail.com.
[10] The author can, in addition, supply not only the databases as used in the example search discussed below, but unedited copies of the same (i.e. the databases as originally downloaded) with a further text document detailing any amendments (necessary typographical corrections) made.
[11] The 'chain of parents' in ancestor_finder's output file is listed numerically. Unlike BMDTreeGen, no identification code need apply as the relationships are not complicated. Numbers in parentheses prior to each listed name state the number of that person's child.

BMDTreeGen, if it is to be used with knowledge of such limitations, or suggest that the program itself is no better than a human, but that it can make more mistakes, faster. The latter can be ruled out - BMDTreeGen, if not relied upon with exclusivity, has the double advantage of providing additional, often unwanted, instruction as to the types of human error that must be accommodating for when trawling old records. Testing of a tree produced by BMDTreeGen against one produced manually is conducted using the descendants of Edward Hancock, born 1849. The search parameters employed are initially for marriages between the ages of 16 and 40, and for children to be born within 20 years of marriage. These windows are used to search for appropriate birth records, accepting that multiple records with the same name may exist within such a window[12], and as such there must also be allowance for 'follow-through error', whereby a death record cannot be found as the age at death as stated upon the otherwise-correct record would no longer tally with the correct birth-year (an earlier birth being incorrectly selected by BMDTreeGen)[13].

The following errors in BMDTreeGen's output are noted; however, explanations may be given which refer to the content of the databases themselves. This is not unashamedly apologetic; rather, BMDTreeGen automatically identifies these records as being potential sources of error too, to allow for BMDTreeGen to be used in conjunction with a more detailed human analysis of the databases. Aside from records not found as a cause of spelling error[14] and with awareness of the complications detailed above, the only errors BMDTreeGen has to accommodate for are as follows:

1. Late marriage missed/early birth error. A broad search window for years in which two people may marry translates into an equally broad search for the births of each person. The more immediate error, of course, is that late marriages are simply missed if this broad search is not conducted[15]. However, even with the broad search window intact, the 'early birth error' is introduced as a consequence of it - a younger person with the same name may be falsely selected as representing the appropriate candidate; the incorrect birth-year is thus introduced into BMDTreeGen. For instance, a search for marriages between the ages of 16 and 70 will cause Emily May (correct b. 1871) to be falsely assigned a birth-year of 1845, if using as upper bound her marriage year of 1900[16]. Confusingly (and irritatingly), the broad search window for marriages is justified in this case - Emily is the third wife of Edward Hancock, b. 1849, and 22 years his

[12] BMDTreeGen will log a warning message for any such occurrences. The convenience of a broad search window also increases the risk that the youngest of multiple people with the same name are erroneously incorporated into a genealogy. However, restricted search windows rule out true late marriages just as surely as false ones; without further sources of information, there is no immediately convenient way to determine what is plausible and what is not. Using BMDTreeGen with a 'standard' 16-40 yr. marriage search will inadvertently rule out a late marriage known to be true, for instance: of Edward Hancock (b. 1849) to his third wife, Emily May (22 yrs. his junior) in 1900 (when he was 51). However, increasing the search parameters to wilfully include them may introduce another source of error: 'false positive' late marriages.

[13] It is easily possible for BMDTreeGen to appear correct when, compared to a human-created database, any instance of 'no death record found' in its automated genealogy also corresponds to a similarly absent death record from the manual genealogy. In these cases, BMDTreeGen doesn't find a death record not by follow-through error, but because there is no record to find. The automatic and the manual genealogy will correspond by coincidence.

[14] Refer to the supplementary file, "BMD Errata," for modifications made to the downloaded databases that corrected spelling errors and suchlike

[15] See the example output file for a search of marriages between the ages of 16 and 40 yrs; Edward's third wife, Emily, is missed as he married her at age 51. This inadvertently removes from the lineage six children, six grandchildren, six great-grandchildren and two great-great-grandchildren - noting these are only the records both present and accurate in the Bath BMD, too!

[16] BMDTreeGen reports that there are 5 records of 'Emily May' between 1830 and 1900, of which only the earliest was selected - however, this warning sounds a cautionary note as to the interpretation of BMDTreeGen's automatic genealogy.

junior. The early birth error can be mitigated by altering search parameters: if the max. age at which a person can marry is high (e.g. 70), then in the case of Mary Ann Gill, the birth-year is falsely determined to be 1841; if somewhat more reasonable (e.g. 40), the birth-year is found to be 1865 (much more in line with her husband‘s birth year, known to be 1873). In the case of the 1841 birth, BMDTreeGen at least reports that 4 records for Mary Ann Gill exist between 1824 and the year of her marriage to Albert Edward Hancock, 1894. Three of these must precede 1854 (i.e. 1894-40), however - BMDTreeGen makes no such cautionary report with regard to the 1865 birth. Indeed, 1865 does seem plausible.

2. Wrong or absent death record, caused by the early birth error. This is an error of false inference, looking for a death record in the wrong place, given that although the record has the correct name, the associated age at death will no longer correspond to the birth year BMDTreeGen believes that individual to have.

3. Recent death. Based on their updating schedule, the death databases do not necessarily account for very recent deaths, i.e. within the last year, or thereabouts.