Open Access Repository
www.ssoar.info

Historical Software Issue 1: Introduction

Thaller, Manfred

Veröffentlichungsversion / Published Version
Zeitschriftenartikel / journal article

Zur Verfügung gestellt in Kooperation mit / provided in cooperation with:
GESIS - Leibniz-Institut für Sozialwissenschaften

Empfohlene Zitierung / Suggested Citation:
Thaller, M. (1981). Historical Software Issue 1: Introduction. *Historical Social Research*, *6*(3), 75-82. https://doi.org/10.12759/hsr.6.1981.3.75-82

Nutzungsbedingungen:
Dieser Text wird unter einer CC BY Lizenz (Namensnennung) zur Verfügung gestellt. Nähere Auskünfte zu den CC-Lizenzen finden Sie hier:
https://creativecommons.org/licenses/by/4.0/deed.de

Terms of use:
This document is made available under a CC BY Licence (Attribution). For more Information see:
https://creativecommons.org/licenses/by/4.0

Leibniz-Institut
für Sozialwissenschaften

Mitglied der
Leibniz-Gemeinschaft

Diese Version ist zitierbar unter / This version is citable under:
https://nbn-resolving.org/urn:nbn:de:0168-ssoar-52386

HISTORICAL SOFTWARE SECTION(1)

"Historical Software" is a term which certainly needs explanation, particularly if one starts to open up a regular section in a newsletter about it. Historians have used computers for quite some time now, but the views about what the typical use of such a machine by a historian should be are widely conflicting. Some of the classics that have been published under titles like "the historian and the computer" tend to treat them as a kind of very large pocket calculators, which crunch numbers very quickly and are therefore suitable aids for all historians which encounter so many of them in their research that paper and pencil or pocket calculators would be too slow.(2) On the other extreme, similiar titles may cover descriptions of very complex attempts to formalize logical concepts of the content of historical documents(3) or applications which use the capabilities of sort programs and/or the more sophisticated output devices to produce editions of historical source material which are cheaper than the ones produced by more conventional techniques. They contain more or less refined sets of computer produced registers but do not differ methodologically from editions which would have been produced by historians using enlarged clerical staffs instead of electronical equipment.(4)

So one tends to say that historians use either software which has been produced by Social Science projects to perform numeric calculations or such developed by philologists to work with literary material. The exceptions to this rule are roughly divided between two classes:

- there exist the "very large projects" which developed programs for tasks of tabulation and other purposes, which in principle could be fulfilled meanwhile by generally available software. The reason for such developments was either, that such programs did not exist at that time, or, that the amount of data to be analysed by those large projects could not be processed cheaply enough by standard software which is usually designed to work upon datasets with a couple of hundred or thousand cases rather than hundred thousands of them(5) - reasons which are by no means specific for historical research projects, but have been leading to similiar "private" developments in big and long running projects in the social sciences.(6)
- there exist quite a few attempts to use special programs in higher programming languages for the preparation of data to be analysed later by existing packages. Such developments usually originated from the feeling, that for the historian it is very hard to translate sources immediately into the rectangular matrices of code numbers required by the prominent packages. The range of such attempts is very wide, indeed: it includes numerous rough and ready programs which can be used just for one particular type of historical source(7) right through to general purpose programs(8), providing for complex sets of input formats which overcome difficulties usually arising from any attempt to use the machine to

smooth up the process of translating expressions of uncertain wording into clearcut categories of statistical analysis.

This last class of applications then, seems to be what constitutes the most "historical" use of the computer in the sense, that it cannot - or not completely - count upon the use of programs produced by other disciplines. "Historical Software", then, as a first definition, would consist of all programming efforts which are designed to treat historical source material with as much care for the potentially important detail as the tradition of history as a discipline asks for, while preparing it at the same time for analysis by methods from the social sciences which see wisely developed generalizations as more central than fascinating details.

Still, a section in a newsletter dedicated to the improvement of methodological relations between historical and social sciences will not restrict itself to this narrow interpretation of "Historical Software"; considering itself as a service to QUANTUM members, rather than as a set of theoretical utterances, it will follow three aims:

- to inform in regular intervalls about some of the less known features of existing packages in the social sciences which are of particular importance for anyone working in the described interdisciplinary area.
- to inform about some of the packages from literary and linguistic computing, which, while at first look relatively far from quantification, can be potentially very useful for sophisticated approaches, including things like quantitative content analysis.
- to improve the cooperation between producers and users of "Historical Software" in the sense mentioned above.

The first two of this aims will clearly be fulfilled best by short documentations plus answers to concrete problems brought to the knowledge of the author; this "plus" being at the same time an invitation for proposals about which techniques/programs/methodologies QUANTUM members would like to hear about in these pages.

The last attempt shall develop into an additional service offered by QUANTUM via the Zentrum für Historische Sozialforschung; its detailed description fills the remainder of this section.

As already mentioned, a rather large number of locally developed programs for the preparation and analysis of historical source material already exist. Source material has been made machine readable to be processed by them. Usually the conventions of such programs are so particular to the special developments that the machine readable data sets can scarcely be processed at any other installation or by any other system, if one cannot master the programming skill to write conversion routines in a higher programming language. As in many projects, the "private" software has been created at the beginning by a programmer who was hired for a relatively short time, such skill usually gets lost during the later stages; exactly at that time it becomes quite often clear, though, that it might be very helpful to use programs developed elsewhere. Besides that, the effort put into preparing sources in a way which includes as much of the information contained in the original, even though not all of it may have been needed for the immediate analyses planned, is often justified by the hope, that such material may prove useful for other researchers later on, who could then analyse a particular source without going once again through the dreary business of creating a machine readable data set. In reality, it very often happens, that after the end of such a project the person responsible for the dataset changes his/her institutional affiliation and when, after a couple of years, somebody actually wants to do some secondary analysis, it is discovered, that the programs originally written have meanwhile become inoperable. To improve this situation we herewith announce the following plan:

In the context of QUANTUM a set of programs will be developed, which will allow the quick and easy reformatting of all source materials made machine readable under any of the following classes of format specifications, which, to our best knowledge, should cover almost all input conventions described during the last 20 years in the relevant journals. This means, that everybody, who has created data according to specifications developed at his computing centre, can have them converted to almost any format necessary for their processing at another one or with a different package. This does not include any guarantee that institutional access to programs developed elsewhere is possible for a given project. It should be mentioned, though that most projects which have developed their own programs during the last years have shown readiness to cooperate with other ones; means to provide a network for such cooperation, at least within the German speaking countries, are under investigation. As far as CLIO, the system developed by the author, is concerned, the offer to provide its services to other projects is repeated herewith.

The following prospect of input formats which are supported(9) is necessarily rather abstract and many historians, who might think it a good idea to take some of their data, have them converted to another format, process them by a specialised system somewhere else, reconvert the changed set and process it further with their own programs, may feel uncertain, whether a given data set falls under the conventions described. So, the service offered, shall be summarized first of all in non-technical language:

- If any member of QUANTUM has prepared a data set, which was made machine readable according to the rules of non-standard programs, containing stuff like hierarchical data structures, specialized spelling conventions, nominative material or complex textual variables, and wants to perform anything with such a data set that is possible by existing programs, but not at his own local computing centre - as translating it into rectangular matrices suitable for statistical analysis, creating sophisticated output for printing, nominative record linkage and so on - he can have his data set converted to the input specifications of the program needed to perform the requested operation.

- The Zentrum für Historische Sozialforschung will arrange for the required software. This will be done in three stages. From October 1st of this year, access to suitable software will be provided. As soon as the demand for this service makes it necessary, all components needed will be made operative within the context of the Zentrum itself. If requests tend to be centered at particular universities, the necessary programs plus assistance for its implementation at the local site will be provided.
- If you think you would profit from such a service, but are not sure whether your data qualify, ask; we will in any case try to be helpful.

Now, after these preliminaries, here is a description of the input formats to be covered. While necessarily technical, it has been tried to keep it understandable for everybody who has had some exposure to the language of EDP - actually this is a kind of zero draft of an internal working paper, which shall describe the design of the planned conversion routines; if you are interested in this lengthier and more technical description which shall be developed during the next months please request a copy.

Basically, any system for the processing of historical sources can use a combination of three input "philosophies".

1.1 A source text can be input in a way we will call "structured". I.e., words, names, dates or numbers are taken from the original context, and rearranged according to specified rules, as "on every card there is the surname of a person, starting in its first column; a slash and the Christian names follow ... if any text remaines, which cannot be treated according to the preceding rules, enter it unchanged on as many additional cards as you need, starting each of them with an ampersand (&) in the first column."

1.2 A source text can be input in a way we will call "predited". I.e., the text is transcribed verbatim onto some medium. Later, special signs are added, to clarify the meaning of given words, names and so on, according to rules as: "Prefix any calendar date in the text with an asterisk. ... Every surname shall be suffixed by a slash, followed by a number taken from the following list. ... Every personal pronoun shall be suffixed by a backslash, followed by the number assigned to the person it's refering to."

1.3 Finally a source text can be entered verbatim and remain unchanged. This is usually the case, if linguistically oriented analyses are performed.

In principle conversions from and to all of these "philosophies" - and the mixtures created out of them - will be supported. It should be made clear, though, that the following restrictions exist:

- a structured text can be transformed to preedited or unchanged text only if and so far as it contains markers preserving the original order of the text.
- no support for grammar oriented transformations of natural language will be provided.

Independent of these principles of input, computer services for a given project are usually developed under one of the following orientations:

2.1 Compiler oriented. In this case rules such as "any string of characters, which is limited by the constructs '(N=' and 'N)' contains the surname of a person, followed by as many Christian names, as may appear in the source, separated from it and from each other by an arbitrary number of blanks" imply that the historian asking for any result concerning information connected with names will write a series of commands advising the machine to get to the next '(N=', take whatever follows up to the next 'N)' and process it in some specific way. Such commands can either be macros of a string processing system or quite simply statements in any higher programming language which has a string handling capability.

2.2 Package oriented. Here the input data are, before any computations are performed, processed by a program, usually known as "parser", which reformats the data. The historian afterwards specifies a variable name such as PNAME to advise another program to perform a given computation regarding nominative information. If any corrections have to be made, they are done with the help of the local editor. The researcher himself is responsible to provide all the commands necessary to instruct the Operating System of his computer to provide access to the data sets needed.

2.3 Data bank oriented. Everything just said about "parsing" and variable names holds. Additionally such systems usually contain their own updating modules; in any case they do all computations on the basis of sets of files which are not accessible for the user outside of the system and are, potentially, even created and maintained without his knowledge.

Principally the conversion of input formats is independent from the orientation of the system they were supposed to be processed with. The following restrictions may prove rather crucial, though:

- data coming from compiler oriented systems have usually not run through any implicit validations of the input format and contain very often a large number of ad hoc conventions. ("Every string consisting only of digits is a price; if it starts with a zero it has not been in the original source, but found somewhere else; if it starts with two zeros,though, it was in the original source, but is measured in another currency.") Such conventions mainly result from the discovery of additional problems during data input, when the simple primary rule ("Every string ...") should not be invalidated. They will cause headache to most package or data bank oriented systems, which usually provide their own solutions to such problems. To convert such data we therefore need a list of all such "special cases" - please don't believe, that what seemed to be self-evident within the environment of one string handling system is at all reasonable within another one. The missing implicit validation will sometimes provide weird output from package or data bank oriented systems, as they tend to assume, that all data have gone through system-specific logical checks. - So here, too, a complete set of all assumptions about system behaviour in the case of format violations is needed. (Even if you are sure, that you have not overlooked any errors; you almost certainly have.)

- data coming from package oriented systems are easiest to convert; if they shall be processed by a compiler oriented system, please note, though, that you have to do a lot more of "just technical" things in the future. We would advise you to make detailed contacts with somebody knowing very intimately the system you are going to use, before you tell us, into which conventions we shall translate your data.
- data coming from data bank systems have all the qualities of the ones from packages, just described. Many of the computations in such systems are usually made possible, though, by data representations, which are completely internal and you are not informed about; we can only convert input data for such systems, not their system files.

Finally, data from all input philosophies and program orientations can employ several formats. The following ones are going to be supported:

3.1 Basic data representations:

3.1.1 Fixed format: the limitations of any item of information are given by its starting position (column) and its length on a specific input medium.

3.1.2 Freefield format: every item is delimited from the next by the same delimiter. Which value belongs to which variable is known from the fixed sequence of the items.

3.1.3 Tag-Content logic: every item starts with a symbol that carries the information to which variable the value of this item shall be assigned.

3.1.4 Paired inclusion symbols: the input text is interspersed with symbols which form logical pairs. Which variable gets the string between those two symbols as value is decided by the starting and/or closing symbol of the pair. Such pairs of symbols may form hierarchies.

3.2 Legal sign sets:
All printable ASCII characters are going to be supported; no signs are reserved. Three classes of sign modification for the representation of characters not in ASCII can be handled:

3.2.1 Representation of single signs by arbitrary strings of signs. (":a means ä.")

3.2.2 Modification of one sign by another, announced by a selected character: ("Every construct x&y means: treat y as superscript to x.")

3.2.3 Protypes, announced by a selected character: ("Every x&nnn means: treat X&nnn as one character." This kind of sign representation is usually interesting only if data shall be converted for the benefit of programs preparing them for computer setting afterwards.)

3.3 Data structures:
Any mixture of the following structures can be converted from. It should be mentioned though that very many programs which may be interesting for special applications have quite a few restrictions with regard to the structures they can handle.

3.3.1 Rectangular data matrices.

3.3.2 Hierarchies ("Treat all records describing estates, which appear after the record describing person n and before the one describing person n+1 as belonging to person n.")

3.3.3 Networks. ("Treat all persons which have the value 'PERSON121' in a given variable as connected by some specific relation; e.g. treat them as identical, i.e. store all information regarding this person only once and use it in every context in which the said value appears.")

3.4 Miscellaneous Conventions:

3.4.1 Multiple values: Translation from systems allowing a variable to have multiple values to such allowing a variable to appear as often within a case as it can be assigned a value is possible.

3.4.2 Repetition signs: Coding conventions, which allow the use of special symbols to indicate, that a variable shall get the same value as the same variable in the preseding "case" (or structural element, or knot in a network) did, are acceptable.

FOOTNOTES

[1] Address all communications to: Manfred Thaller, Max-Planck-Institut für Geschichte, Hermann-Föge Weg 11, D 3400 Göttingen or QUANTUM.

[2] E.g. Edward Shorter: The Historian and the Computer, Eaglewood Cliffs, N.J., 1971.

[3] E.g. Rolf Gundlach und Carl August Lückerath, Historische Wissenschaften und elektronische Datenverarbeitung, Frankfurt a.M. ect., 1976.

[4] E.g. Karl Schmid (Ed.): Die Klostergemeinschaft von Fulda im frühen Mittelalter (= Münstersche Mittelalterschriften 8/1-8/3), München 1978, or, if one looks at the edition only and not at what's done with the sources besides editing them, Alan MacFarlane (Ed.): Records of an English Village: Earls Colne 1400-1750, Cambridge, 1980 sqq.

[5] Quite a few of the programs developed by the Philadelphia History Project should meanwhile belong to this category, though not all of them. See Historical Methods Newsletter 9 (1976),43-181.

[6] E.g. TASUS developed by the VASMA project in Mannheim. See Johann Handl: TASUS: Ein Programmsystem zur Verarbeitung sozialwissenschaftlicher Massendaten. In: Tagungsband der 9. WASCO-Tagung am 10./11. April 1980 in Mannheim, Ed. by WASCO (Wissenschaftlich-technische Anwender von Siemens Computern e.V.), Mannheim 1980. 274-282.

[7] E.g. most of the developments described and quoted in: Ad van der Woude and Anton Schuurmann (Eds.): Probate Inventories, A new source for the historical study of wealth, material culture and agricultural development (= A.A.G. Bijdragen 23), Wageningen 1980, passim.

[8] E.g. the Earls Colne Projekt programs (see: Timothy J. King: The use of computers for storing records in historical research, in: Historical Methods 14 (1981), 59-64) or the author's own CLIO (HSR 15 (July 1980), 40-65).

[9] Care will be taken to ensure, that the input formats of all programs can be handled, which have come to the knowledge of QUANTUM as having been developed by its members. The same holds for the following systems, which provide the potential user with full sets of solutions to a large class of problems which typically arise when inputing historical source material. (The specifications of these systems, too, can be taken care of only as far, as standards out of manuals and/or other documentations could be acquired.)

TU-STEP (Specifications are available from Dr. Wilhelm Ott, Zentrum für Datenverarbeitung, Brunnenstraße 23, Tübingen.)
CONVRT (see: Lawrence Glasco and Reginald Baker, in: Historical Methods Newsletter 7 (1974),125-128).
The input system of the Cambridge Group for the History of Population and Social Structure (see: Roger Schofield and Ros Davies, in: Historical Methods Newsletter 7 (1974) 114-124).
The input system of the Earls Colne project in Cambridge (see: note 8 above).
CLIO (see: note 8 above).