Historical Software Issue 5: Literary and Linguistic Computing - FAMULUS, OCP, COCOA, LEXICO, COBAPH

Thaller, Manfred

Historical Social Research, Vol. 7 — 1982 — No. 3, 78-87

software

HISTORICAL SOFTWARE SECTION[+]

When you ask historians why they started using the computer, the vast majority will probably NOT start with some reflections upon the methodological virtues inherent in statistical reasoning, but very many people will honestly admit, that they were lured into doing so by two perspectives: being able to perform a LARGE number of routine tasks QUICKLY and, furthermore, being able to perform a large number of routine tasks quickly. Starting with such bright hopes, many people get very disturbed, when they finally discover, that "using the computer" amounts quite often to the definition of very exact theoretical frameworks even for relatively unimportant sidetracks of the line one wants to follow, operationalising those frameworks into rather elaborate codesystems and finally studying, which statistical methods might be most useful for a given application.

Not all of us are happy about this situation: in the very first of these sections we quoted a number of systems that were specifically designed to use a computer not only in the later stages of work - that is, for the production of statistical computations - but during the initial stages of data preparation and definition of categories as well. Another approach to get the "large and quick" just quoted is the development of retrieval systems in connection with a data bank specific for some topic of research.

Quite beyond that there exists the large community of "Literary and Linguistic Computing" which has quite a few historians among its members. Indeed one might say, that computerusing historians are split into two camps: users of the computer as a tool to support statistical methodology and users of those machines as instruments for improved and cheap editions of sources along classical lines.

Such splits - not free of animosity - are always unfortunate. Still one has to live with them. In our special case, though, we think, that the situation is bound to change by a number of developments which have taken place during the last few years and tend to become increasingly important.

The first of these developments is of course the micro-computer. It shall not be praised without qualification in this place as, in my opinion, happens all too often already. Still those machines constitute a major breakthrough - even archives, not usually the hastiest institutions when it comes to technological innovation, have accepted users who set up small micros for data entry in the room reserved for users in general.

The second major event seems to be the appearence of the Kurzweil Date Entry Machine - finally able to read printed books and the writing of ALL typewriters (and not just the OCR heads).

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

Both developments point, in my opinion, towards the same direction: it makes less and less sense to enter data in numerical form; it becomes easier and easier instead to enter data as texts "just as they appear in the original". That seems to ask for two things: first of all, the interest in specialised software for the preparation of data for statistical analysis, which have not been entered in a form particularly suited to it, becomes increasingly important. In addition to that, the borderline between the two camps in "historical computing" which we already mentioned, should become much less clear. Users of statistical methodology will increasingly encounter data which are machine readable already, without having been prepared for statistical analysis. - On the other hand the preparation of textual corpora for EDP purposes should become increasingly cheaper, making it much more desirable to use formalized approaches for the (at least) secondary analysis of material acquired initially for more conservative purposes.

In the first of these software sections we announced our intention to provide a complete conversion capability between the different input formats. This service - used that far by a small number of our members only, but used already - is available now in principle and will be improved as the demand increases. Additionally we will try in future to improve - at least on the technical level - the relations between "quantifiers" and other classes of people in the historical sciences using the computer. As a first token for this promise we will dedicate this issue to a short outlook on some specimens of the programs used in "Literary and Linguistic Computing".

The reason why we do this now is not only the general intention indicated; the current issue seems to be particularly suitable for this purpose - at least in Germany - as the "Informationszentrum Sozialwissenschaften" is currently just finishing the publication of a survey of software available for "Linguistische Datenverarbeitung". This documentation was done in the tradition of the one on statistical software available for the Social Sciences published earlier (1), being initiated by the former SIZSOZ project (Software Information Centre for the Social Sciences).

The new documentation was certainly much more problematical to begin with: There's much less professional software available in the area of Literary and Linguistic Computing as there is in the realm of statistics, so the standard of the descriptions of available programs had to be much more cumbersome to define. It can not yet be definitely said how far these difficulties will be finally overcome when the documentation is complete - the following remarks were written on the base of a preliminary offprint of the material collected.

"Software for Literary and Linguistic Computing" as understood by the contributors to the emerging documentation covers mainly five areas:

- programs generating several classes of wordlists - indices, counts of word frequencies, KWIC's or concordances. In many cases more or less developed features for lemmatizing the texts exist.
- programs which allow solutions of problems in linguistic research, e. g. systems to analyze certain classes of grammars of other sets of rules, when applied to given corpora of text.
- systems out of the research done in Artificial Intelligence, e. g. systems to answer questions in natural language.
- information retrieval systems.
- utilities for various projects.

The last category mentioned - "utilities" - is probably the one, where the greatest effort will be needed by the editors of the final documentation, as the material is on extremely different levels just here: so among the descriptions received, there was one of a system that is able to write magnetic tapes in a particular format, that's useful only at the machines of one manufacturer and only if you intend to analyse those tapes further with a certain, very specialised, program system.
This example probably shows, what is most problematic about the whole documentation: practically all the systems were described by their authors - and some peple have a very high regard for the importance of their work.
I do not envy the editors.
Let's hope they will be able to iron out the enormous differences between the descriptions that were delivered to them - in any case the final documentation should be very useful for any historian who intends to start working on a computer with continous texts.
Particularly for one purpose: to show what already exists and to convince the beginner, that in this field so much has been done already, that it would be perfectly pointless to start from scratch without attempting to get at least an idea which problems have already been solved elsewhere.
From the groups the available software can be divided into, probably only two are relevant for most historians: concordance generating programs and retrieval systems. That utilities are mostly restricted to use within the hardware/software environment of one particular computing centre we have already pointed out. "Linguistic software" in a narrow sense is so specialised, that it will probably be interesting only for historians which are very much oriented towards philological analysis. Software from the area of Artificial Intelligence is generally designed for experimental usage - that is, it has been mainly developed as an object of research, not as a tool for other researchers; furthermore it usually has been written to handle only very small amounts of data. (2)
Even retrieval systems, as described in the forthcoming documentation, seem to me to be interesting mainly for showing how much is available in that area by now. None of the systems described seem to have any features which are particularly suited for historical applications (special precautions for multiple currency systems, routines for the comparison of names, routines for the handling of networks as they appear in genealogical studies, handling of "vague" or "fuzzy" queries and so on). Most of them are strictly oriented towards bibliographical documentation with quite simple data structures and main emphasis on the handling of abstracts with few restrictions upon the wording of them. I doubt if most of them offer much more to an historian than is already made available by FAMULUS which - available at most Computing Centres meanwhile - is old, kind of clumsy, but very easy to use for the typical down-to-earth applications of historians. Still, if one thinks that system to be insufficient, it would be a very wise move to check with the documentation we are talking about, if there doesn't exist another one already, that will run on the local mainframe and provide features more suitable to the respective application before one starts the development of yet another retrieval system.
Systems which generate lists of words and/or concordances can be quite useful for many historical applications. Concordances and KWIC's have become a major result of nonquantitative applications of EDP in the humanities. One can definitely not doubt, that they are useful tools,

if one wants to make an intensive analysis of a larger text or a collection of smaller ones. As a matter of fact, while having developed a system myself, that contains quite a lot of retrieval options, I would be the last one to doubt, that a simple rough-and ready KWIC of, say a collection of letters, which is printed at the local computing centre, may be a much more efficient way to provide access to the information contained within the material than the attempt to create yet another "data bank". On the other hand the production of concordances in itself is scarcely less "dangerous" than the undue enthusiansm for on line retrieval systems rightly characterised as "infoholics" by some reviewers. Dangerous, that is, if working about e. g. a collection of letters, one believes necessarily to have to publish the resulting concordance, ordering the word along the lines of a competely new approach, providing means for the treatment of variant possibilities to read illegibly scribled notes and takling all the other endless minutiae that stand between a rough-and-ready printout that answers 99 % ot the questions and a concordance worthy to be published. If one falls into that pitfall, producing a concordance which is scarcely sold - nothing to say about being used - it will be a very costly and prolonged endeavour; if one does not, the rough-and-ready KSIC can come extremely cheap, currently available hardware and software being what they are. Commercial houses in Germany offer currently to make printed texts machine readable at a rate of 2 - 3 DM per 1000 characters. (3) If one has access to some microprocessor that has a reasonable editor and a reliable link to the local mainframe, costs can compete with that. There exists software, which will produce a complete concordance out of a machine readable text when given a few metacommands with a syntax that's even easier than the one employed by SPSS. And such programs can have facilities to restrict the concordance produced to a small number of keywords or certain predefind contexts - within the same command language; such packages furthermore can provide facilities for lemmatizing the texts stored, so you can, if necessity arises, step by step improve your rough-and-ready KWIC into somewhat more sophisticated. If - and I would strongly emphasize and only if - access to the necessary hardware and software exists, I would consider for very many historical research projects the production of such ambitionless concordances as a possibility that's much superior to the design of another data bank which, after months of data input - and many more months of development if the programs themselves are written anew -, is finally consulted twice a month.

Three such systems we will describe in this section. One of them - OCP (4) (an acronym for Oxford Concordance Program) has been specifically designed as successor to COCOA, an early concordance generator that has been - and is - very widely available indeed. OCP has the advantage of having been designed from the very start to be transportable. It is supposed to run on most brands of computers and it is relatively small. (Something like 11.000 lines of FORTRAN, written in the 66 standard. The authors claim, it should be possible to compile it with few - if any - changes with all compilers which follow the 77 standard.) So you should have few difficulties in convincing your local computing centre that the acquisition of this package would not put to much strain into the resources available. Indeed, if you have to initiate the acquisition of a concordance generating system at your installation, OCP is what I would recommend.

As an example of what a different and more complex approach might offer, we will describe LEXICO (5). This system is certainly more comfortable that OCP in that it hides the operating system almost com-

Historical Social Research, Vol. 7 — 1982 — No. 3, 78-87

pletely from the user: it contains its own filehandling and backup system (that implies tapehandling) plus specialised routines for data input, furthermore its own editor, extremely lengthy explanations of errors made by an user and even a scheduling mechanism that submits complete batch programs created out of user requests, allowing the user to give all commands interactively, though doing everything in batch which is to expensive for dialogue. Unfortunately this system (some 27.000 lines of FORTRAN 66 plus a number of routines in UNIVAC assembler language) is related extremely closely to the architecture of the machine (and obviously: the operating system) it has been written for (an UNIVAC 1110). So, while I would say that the overall design of this system would come pretty close to what I consider the optimal one possible at the current time of development, you should only try to acquire it if you have access to an UNIVAC computer of the 1100 series.
Last, not least, we will describe COBAPH (6), a system of COBOL programs which is organised as a set of separate modules and requires slightly more background knowledge from the user than the ones mentioned previously. The size of the source programs is not known precisely to the author - one should add, that the development of the last version of this system was not completed and development has stopped. This system can not be recommended for acquisition therefore: one should study it quite carefully nevertheless, as it is a very good example how much comfort a program system can provide for the user even when it has been designed for an environment with very scarce resources.
This selection is admittedly biased: the systems we describe here were selected according to the following criteria:

- they should be true concordance generators, that is, they should produce some kind of word list by default. Therefore we did not include systems which allow the creation of concordances if relatively complicated programs in the respective control languages are written.
- They should be independent of specialised hardware, beyond the dependency from one mainframe. So we did specifically not include systems which contain (assembly language) routines for screen handling or assume a distinct type of behavior of the video terminals available at an installation.
- Software section is short, the three systems selected are examples: complete coverage of the field "concordance generators" cannot even be attempted.

Nor do we try to give in the following descriptions detailed introductions into the respective command languages: we just want to show that programs of that kind may be useful for a historian who is not linguistically-philologically oriented and what kind of highlights they provide.

OCP do begin with, has certainly the most complete defaults: if you call the program and add the command line *GO, the system will assume that the standard input unit contains text on 80 column records and produce a word index in the English alphabetical order.
For data input the system provides for:

- "masking out" any fixed part of the input records,
- defining symbols indicating "comments" which shall not be processed along with the text,

- continuation line and multiple line indicators.

Portions of the input data can be addressed by:

- explicit or implicit line numbering,
- references abbreviated to 1 character ("Z" = "TITLE", "A" = "AUTHOR" and so on) which may but need not form hierarchies.

The data to be processed can be restricted to:

- any range of input lines,
- any portion of input text that can be defined by a logical expression using the defined references (PROCESS WHERE A = "SHAKESPEARE" EXCEPT WHERE T = "HENRY"). This is for historians definitely a highlight of the system, as it makes it usable as a primitive retrieval system for collections of (very simple) documents.

Order of sorting and definition of the character set (multiple character representations of "letters" which can not be represented by one character on the available keyboard, diacritics and so on) are completely under user control. (Actually the system was designed with the specific aim of being able to handle non-european alphabets.)
Which words are selected for processing,can be controlled by a quite powerful pattern oriented command. One can select words that are listed explicitly, word which contain certain patterns of letters, phrases constituted by such words and/or cooccuring in a defined distance, words and/or phrases of a given frequency and so on.
Lists of words and forms of such can be defined to belong to one "headword" (that is, a lemma).
For the parts of the input data and/or the specific word selected one can request either lists of words appearing, an index of those words, a concordance (i. e. a KSIC) and word frequencies being output for further processing by the usual statistical packages.
The format of the output produced can be controlled very well, e. g. with regard to page titling, page numbering, splitting a page into multiple columns and so on. The extension of the context printed within the KWIC's is under user control too.
OCP operates always upon an input file that has been created and administrated by other programs. No equivalence to a "system file" is known.

The last sentence was put at the end of the remarks about OCP as it defines its biggest difference to the second system we are talking about, LEXICO. LEXICO is probably inferior to OCP in most of the qualities which we just have described. It asks for a quite specific input format, allows less control over the character set, the collating sequence and the output formats a user can specify (you have even to make special arrangements to distinguish between upper and lower case letters). Furthermore the user is restricted with regard to the identifications he can assign to different units of his or her texts and it is much more difficult to select only a subset of the words for processing - LEXICO lacks cefinitely the quality of an extremely simple retrieval system that OCP as certainly owns.
Why then one could read from the basic descriptions of the systems we are speaking about, that this reviewer thinks LEXICO to be - from an historians point of view - at least as useful? Its great virtue is, that it is defined around a whole set of interrelated system files which have been written to implement an integrated concept of what working with continous texts should be like. As this concept does not see so much the production of a concordance as its aim, but "lexico-

graphic processing", one might indeed say, that we compare two different classes of programs being potentially unfair against both of them. We do not intend, though, a comparative evaluation of those systems: what we try to illustrate is just the different kinds of use a historian can make of systems which have not been written for him in the beginning.

LEXICO assumes texts not to be something to be "concorded", but as raw material, to be eventually turned into a database containing information about their constituents. The system assumes, that this is done in dialogue, all jobs requiring a larger share of the ressources available (as e. g. the execution of complex transformations which can be specified interactively) being executed "in the background" as batch runs.

All that happens in five steps, each of which can be repeated as often as necessary.

First of all one has to define a "collection". This kind of entity is described by a number of defaults applicable to all the "texts" contained therein - such as word delimiters and the like. These defaults can be changed by the user at any stage.

Into this collection, which can be understood as a database, one enters "texts" now, that is, distinct units of the whole corpus to be processed, which can be accessed individually later on, as the need arises.

Any of these texts can then be edited with the help of an editor that's inherent to the system. It is quite flexible though it does not contain any macro definition possibilities - what's usually giving their power to the more modern editors available. (On the other hand LEXICO's editor allows quite a number of ways of subdividing a text into discretely addressable segments (lines, paragraphs, chapters and so on).)

Any subset of the texts within a collection can at any point of time be "concorded" - producing optionally a printed concordance and at the same time a internal copy, that's stored for further use.

One of the uses such a concordance is put to, is lemmatizing: LEXICO provides a whole number of ways how to do this. Basically you are entering rules which either respell single words or indicate, that a set of words belong to a "basetype" (i. e. a lemma). When you have finished a set of definitions you ask the system to apply the rules to the "collection" in question and check then, which words could not be attributed to a lemma - if necessary you add additional rules, change existing ones and so on. Within this process you can furthermore interactively resolve homographs - to get them listed under different lemmata in the next concordance you are going to produce.

LEXICO finally adds another step: the creation of slips containing the context a word appears within, which can be handled further manually.

The process just described may be carried on over a very long time - different participants in a research project may work upon a corpus successively. The system will take care of the creation of most of the files needed and assist, if necessary, in writing them to tapes for longterm storage; at many occasions you get a chance to create backups, which enable you to return to earlier stages if necessary.

The very great advantage of LEXICO lies in its ability to make effort additive - you do not have to write down all the specifications for a list of words at once, but you start with some simple steps, have them executed and proceed with the results, having to care only for further refinements - not for what happened before. Its disadvantage, lets stress it again, is the very large effort that would be needed to implement it.

So after speaking about a system, that can be made available with relative small effort and another one, that represents a different approach and can be used only by a very small number of our readers, we should finally come to a third, that might be an excellent representative for the kind of software one can hope to have access to right now at a computer centre, trying to get the most out of the resources available locally.
COBAPH (an acronym for the German words for "general basic program for EDP usage in linguistics and philological subjects") could indeed be interpreted as a model of a concordance generating program. Notable, by the way, that in the introduction to the users manual the authors say explicitly, that they themselves do not understand word lists and/or concordances as the final aim of the studies their system is supposed to support, but rather as an intermediate step, which shall prepare the final analysis of the texts.
If we want to compare COBAPH with the other systems quoted here, we should compare it primarily with OCP: indeed one might describe the facilities provided as a subset of OCP, comprising the possibilities for the description of the sets of signs that are used (multiple representations of one letter by several characters (7) etc.), influencing the collating sequence and arranging for the most convenient format of output. The kind of output available is roughly the same as with OCP (lists of words, word counts, KWICs); additionally "Reimwörterbücher" (i. e. the same kind of output restricted to the last words of the lines) are available. Missing are features for segmenting a text - besides defining hierarchical identification numbers for the lines - and the possibilities to restrict the resulting lists to words which are defined as skeletons (e. g. all words with a given ending) or to restrict the processing to phrases.
The system is very good at defining the usage a character is put to - while the relevant parts of the command language are less comfortable, I doubt that OCP contains many possibilities for e. g. defining a collating sequence for combinations of letters and diacritics, which cannot be realized with COBAPH as well.
While somewhat less flexible with regard to the formatting of output, COBAPH contains enough options for e. g. constructing title lines and divide a page into columns to let me rest assured that they could fulfill the wishes of most of the typical applications within historical projects.
COBAPH output can be treated further by a couple of postprocessors: notably COLA which supports lemmatizing - though batch oriented and less comfortable than LEXICO. As mentioned, COBAPH had to be realized in an environment of pretty scarce resources: so the system was organized to faciliate usage of tapes for sorting and contains its own checkpoint logic, making it easy to divide a lengthy task between several jobs - something everybody will value highly, who ever had to repeat the first 10 stages of an expensive job, just because the computer crashed one second before the results were complete.
COBAPH assumes that tapes are used for holding the data; therefore the system contains a number of possibilities to maintain files on tape - many readers who are hampered by missing mass storage when handling really big files will think it was a most commendable decision to include such components.

We introduced these three systems as examples: COBAPH shows, that even now you have some hope, that software exists at your installation or might be provided by a nearby university, which should make simple KWICs a good alternative to the use of a more or less sophis-

ticated retrieval system; if you can get access to recent developments like OCP (which definitely should be possible to be acquired by most university computing centres, being cheap and easy to implement) you can indeed, with a minimum of data preparation, use KWICs along with a primitive, but certainly not powerless retrieval system. If your university is willing to invest real effort into the acquisition of a system as sophisticated as LEXICO (though not all too many of them are lying around for every brand of computer) you can have at your disposal a system which is able to answer many routine questions quickly with data that need scarce preparation to begin with, but can succesively be used with increasing sophistication.

The next edition of the software section will bring it back to the fold of quantifying methodology, its subject being GRADAP. I would like to remind you, that I'm ready and willing to switch this topic on short notice, if there is some piece of software, somebody would like to have an evaluation of.

FOOTNOTES

1 Informationszentrum Sozialwissenschaften: Sozialwissenschaftliche Anwendersoftware/Social Science Application Software, Bonn, 3rd edition, 1980.

2 This, by the way, is equally true of RESEDA, an Historical Data Base System developed under the guidance of Gian Piero Zarri at the Laboratoire d'Informatique pour les Sciences de l'Homme. (On RESEDA see e. g.: Gian Pierro Zarri: The Use of Artificial Intelligence Techniques in the Conception and Utilization of a Historical Data Base, in: Joseph Raben and Gregory Marks (Edd.): Data Bases in the Humanities and Social Sciences.)

3 One of the next software sections will actually deal with hardware - talking about new possibilities of data input, as the arise from micro-processors and data entry machines. As the performance of some of the commercial houses quoted is currently just being tested, this will be spared for a later edition.

4 The system and the manual can be acquired from Mrs. Susan Hockey, Oxford computing Service, 13 Banbury Road, Oxford, OX2 6NN, England. The cost of the release tape is quoted at 100 Pound St. (Assuming you send an empty tape.) The system will be distributed to Computing Centres only - not to single institutes - and the user has to sign a somewhat lengthy license agreement which specifically prohibits any local changes to the programs. An overview of the system can be found in the ALLC Bulletin, vols. 7 (1979), 35-43, 155-164, 268-275 and 8 (198) 28-35.

5 For a short introduction see: Richard L. Vanetzky et al.: LEXICO: A System for Lexicographic Processing, CHum 11 (1977) 127-137. The manual is available from: The program Librarian, Madison Academic

Computing Centre, University of Wisconsin, Madison, Wisconsin, as: Computer Sciences Department: Computer Sciences Technical Reports, nos. 280 and 283 thru 288 (all 1976).
Unfortunately the main author has left the MACC meanwhile, so there is scarcely any maintenance. The system is available for a nominal handling fee from the address given above. Before you ask for the system, though, you have to get a written consent of Mr. Nathan Relles, Sperry Univac, P.O.B. 500, Blue Bell, Penns. 19424. As the tape is distributed by MACC the system unfortunately contains a couple of calls to nonstandard software components, so in its present shape its not even interchangeably between UNIVAC mainframes. This is currently being changed, though, by Mr. G. Kock of the Gesellschaft für Wissenschaftliche Datenverarbeitung, Am Faßberg, 34 Göttingen. A copy of this version, which will at least be able to run on all UNIVACs, will be available from him within the next weeks. Attempts to make the system somewhat less machine dependent are under consideration.

6 Developed at the Universität Regensburg. A XEROXed users manual is available on request from Dr. Ludwig Hitzenberger, Universität Regensburg, FB Sprach- und Literaturwissenschaften, Universitätsstr. 31, 84 Regensburg. As mentioned in the text COBAPH is not maintained, so no procedure for acquisition exists.

7 This very feature, called "tuple processing" is the only significant extension to the system that was hampered by its new version not being completed. The other facilities implied by the present remark - masking characters out of processing, using them as diacritics etc. - are in the implemented version.