Brian Hayes

Victor Willing, Navigation, *1977*

Reassembly Required

Last September two scholars at Hebrew Union College in Cincinnati surprised the world of biblical archaeology by publishing a bootleg version of twenty-four manuscripts from the Dead Sea Scrolls. The manuscripts were not obtained through some Indiana Jones adventure in archaeolarceny. They were reconstructed from a concordance to the scrolls—that is, a list of all the words appearing in the documents, with their surrounding context.

The unauthorized edition was assembled by Ben Zion Wacholder, a professor of Talmudic studies, and Martin G. Abegg, Jr., a doctoral candidate. They (and others) had grown impatient with the official custodians of the scrolls, who had delayed publication of a large collection of manuscripts for more than thirty years. Following the release of Wacholder and Abegg's edition, there was a brief debate over the propriety of their action, and then two weeks later the entire issue became moot. The Huntington Library in San Marino, California, announced that it would grant scholars access to its photographs of the scrolls, and shortly afterward the Israel Antiquities League in Jerusalem adopted a similar policy. Then, in November, another set of scroll photographs, of unknown provenance, was published in a facsimile edition by the Biblical Archaeology Society in Washington, D.C., which was also the publisher of Wacholder and Abegg's work. Thus the materials needed for preparing translations and interpretations of many of the documents are now publicly available.

Whatever the content of the scrolls, Wacholder and Abegg's clever method of extracting the text is of interest as a computational process. Many people would not have guessed that a complete text could be recovered from a concordance. As it turns out, the process is a particularly easy instance of a technique that has uses in many other fields, from navigation to molecular biology to seismology.

In calling the reconstruction of the unpublished Dead Sea Scrolls a computationally easy process, I do not mean to diminish the labors of Wacholder and Abegg. The concordance on which they based their text had been prepared in the 1950s by a team of four young scholars in Jerusalem; copies of the concordance began to circulate in 1988. Every instance of every word in a given document has a separate entry in the concordance; there are about 60,000 entries altogether. Each one gives a few words of context, as well as the physical location of the word in the manuscript. All the unpublished scrolls are badly fragmented, and so the location is given by identifying the fragment number and the line number on the fragment. Wacholder and Abegg made use only of the contextual information, not the location numbers.

Wacholder and Abegg's first task was to type the concordance into a computer. The computer they had available was an Apple Macintosh equipped with typefaces for displaying Hebrew and Aramaic, the languages of the scrolls. There are word-processing programs that can accommodate right-to-left-reading languages such as Hebrew and Aramaic, but Wacholder and Abegg did not have one. Consequently, Abegg (who did all the typing) had to enter each line of text backwards. The subsequent reassembly of the original text was a computer-aided process rather than a computer-automated one.

It might seem at first that a concordance would not provide enough information for a complete reconstruction of a document. Only the local context of each word is given, and so one is left with a jumble of disconnected phrases, which have to be assembled like the pieces of a jigsaw puzzle. But the puzzle can be solved in a systematic way. Because the pieces of the puzzle overlap, there is a method of assembling them that requires no guesswork or intuition and quickly converges on a correct reading.

Here is a simple concordance to a twenty-nine-word verse in the Book of Ezra:

1. **and** when
2. my beard **and** sat
3. my garment **and** my
4. my head **and** of
5. my mantle **and** plucked
6. sat down **astonied**
7. of my **beard** and
8. and sat **down** astonied
9. rent my **garment** and
10. off the **hair** of
11. of my **head** and
12. when I **heard** this
13. and when **I** heard
14. this thing **I** rent
15. and my **mantle and**
16. and of **my** beard
17. garment and **my** mantle
18. hair of **my** head
19. I rent **my** garment
20. head and **of** my
21. the hair **of** my
22. and plucked **off** the
23. mantle and **plucked** off
24. thing I **rent** my
25. beard and **sat** down
26. plucked off **the** hair
27. heard this **thing** I
28. I heard **this** thing
29. and **when** I

The words in boldface are the keywords; each entry includes two words of context preceding the keyword and one of context following it. For keywords at or near the beginning and the end of the passage, the contextual positions are left blank.

To reconstruct the original text, begin at the beginning. Note that the keyword in line 1 has no preceding context, and so it must represent the start of the text. Accordingly, the first word of the verse is *and*, and the second word is *when*. Now, looking up the keyword *when* (line 29) reveals that the third word of the text is *I*. On looking up *I*, you find two entries (lines 13 and 14), but the ambiguity is immediately resolved by checking the context; the words *and when* in line 13 match the text already discovered, and so the next word must be *heard*. You can continue in this way, adding one more word to the reconstructed text at each step, until you come to the end of the passage, signaled by a keyword with no following context (line 6). The procedure for reconstructing the text is hardly more difficult than ordinary reading. The complete verse (Ezra 9:3) is as follows: "And when I heard this thing, I rent my garment and my mantle, and plucked off the hair of my head and of my beard, and sat down astonied."

Is this method guaranteed to work in all cases? At least two conditions have to be met for the algorithm to operate. First, the concordance must be a complete one, with a keyword entry for every occurrence of every word in the original text. The Dead Sea Scrolls concordance is apparently complete—at least to the extent that the texts themselves are known completely—but many other concordances are not exhaustive; they omit "noise" words such as *the, a* and *of.* The second condition is that each entry in the concordance must include enough context to resolve all possible ambiguities. The concordance to Ezra 9:3 needs two words of context preceding the keyword to distinguish the phrase "and of my beard" (line 16) from "hair of my head" (line 18). A concordance to King Lear would have to provide at least six words of context, lest the algorithm become trapped in a never-ending loop when it reaches Lear's lament on the death of his daughter: "Never, never, never, never, never!"

The real challenge when it comes to practical examples of text reconstruction is dealing with errors. A large corpus of Hebrew and Aramaic that has been typed from left to right is unlikely to be free of typographical mistakes. And even if the transcription were perfect, the fragmentary nature of the underlying text would leave many gaps and discontinuities. The algorithm described above has little tolerance for error. If an inconsistent spelling causes the algorithm to miss a keyword, the process comes to an abrupt halt; if a misspelling leads it to the *wrong* keyword, the algorithm could either skip a segment of text or get stuck in an infinite loop. In dealing with all such irregularities, Wacholder and Abegg relied on the human touch—on reading and understanding the text. Thus it seems the concordance supplied enough information to allow two experts, aided by a computer, to recover the original text; there was probably not enough information for the computer to do it alone.

Did the custodians of the scrolls realize that the concordance would allow the full text to be extracted? It is hard to see how they could have missed the possibility, because the method of reassembling the text from a concordance is essentially the same as the method for assembling the text from scattered fragments of parchment in the first place. Perhaps that is why the concordance was kept secret for thirty years.

Molecular biologists face a problem almost identical to that of scholars reconstructing a text from a concordance. The biologists can read the sequence of nucleotides in a segment of DNA that is fifty or a hundred nucleotides long. But a typical gene has a few thousand nucleotides, and the complete human genome runs to about three billion. To decipher a long sequence, biologists have to break the DNA molecule into many fragments, read the individual fragments and then try to reassemble the pieces on the basis of clues from overlapping context.

The fragments are created by "digesting" DNA with enzymes called endonucleases, which cut the molecule at various specific places. There are several endonucleases, and each one makes its cuts at different points along the DNA strand. As a result, fragments produced by different endonucleases have regions of overlap; the set of all such fragments constitutes a concordance to the complete molecule. The fragments are reassembled in a process known as a shotgun sequence alignment. The reconstruction is almost always done with the aid of a computer and relies less on human intuition than does the analogous task in textual analysis. The reason is simply that human intuition doesn't have much to say about a text that looks like "TATAGCTCGCC"; whether you read left to right or right to left, meaning does not leap to the eye.

As in the Dead Sea Scrolls project, the most difficult part of a shotgun alignment is dealing with errors and uncertainties in the supplied data. The chemical techniques for reading the nucleotide sequence of a DNA fragment have error rates of a few percent, and so any large collection of fragments will have hundreds of incorrect nucleotide assignments. The computer software employed in doing shotgun alignments copes with those errors by taking a probabilistic approach. Rather than give up entirely when it cannot find an exact match, it assigns probabilities to various candidate alignments. Given enough fragments with enough overlaps, a single "consensus" sequence eventually emerges.

Both the reconstruction of a manuscript and the reconstruction of a gene are one-dimensional problems. Words and nucleotides always line up in sequence, one after the other. Things get more complicated in two or more dimensions.

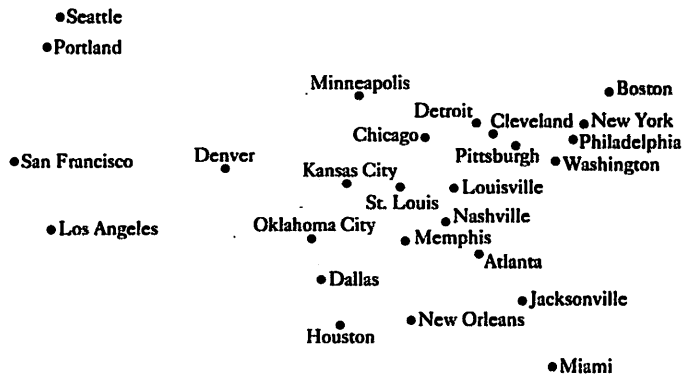
Suppose you are lost in an unfamiliar country (Indiana Jones, indeed!), and your only resource is a page torn from an atlas—a page that shows not a map but merely a table of distances between large cities. Can you reconstruct a map from the table? This problem looks much harder than the textual one. It would be easy enough to go the other way—to construct the table from the map—but transforming distances into locations seems problematic. The mileage table tells you only how far apart the cities are; to figure out where they are, you need some indicator of direction as well as distance, and that information was thrown away when the spatial tableau of the map was reduced to a mere matrix of numbers. Or was it?

It turns out the mileage table provides more than enough information for you to draw the map. Indeed, only a few rows or columns of the table are needed to put every city in its place. As an exercise in ruler-and-compass geometry, the procedure works like this: First set up perpendicular x and y axes, and mark them off in miles at a scale that will accommodate the largest distance between cities. Now pick any city and place it at the origin of the coordinate system, where the axes cross. Then choose a second city, look up the distance to the first city, and place the second city at the corresponding point on the positive x axis. For a map of the United States, San Francisco might be the first city, placed at $x=0$, $y=0$, and Washington the second city, with coordinates $x=2,442$, $y=0$.

Now try adding a third city, say Minneapolis. According to the distance table, Minneapolis is 1,584 miles from San Francisco, and so you know it must be somewhere on a circle of that radius centered on San Francisco. Draw such a circle. Consulting the table again, you can also draw a circle with a radius of 934 miles, centered on Washington. Minneapolis must lie wherever the two circles intersect. But where *do* they intersect?

One possibility is that the circles don't intersect at all. In that case you can conclude that the region being mapped has a non-Euclidean geography, and you should always travel from San Francisco to Washington via Minneapolis, since the trip is shorter that way than it is by the direct route. A second possibility is that the circles "kiss" each other at a single point; that would imply that Minneapolis lies directly on the line between San Francisco and Washington. The third possible outcome—the actual one here—is that the circles intersect at two points, one above the San Francisco–Washington baseline (positive y) and the other below it (negative y). A bit of algebra shows that the coordinates of the two points are $x=1,556$, $y=296$ and $x=1,556$, $y=-296$. Which value of y should you select? It's up to you; the choice is arbitrary. Suppose, then, you take the positive value.

At this point you have in fact made three arbitrary decisions. You have given San Francisco the arbitrary coordinates $x=0$, $y=0$; you have decreed that Washington lies in the positive x direction from San Francisco; and you have placed Minneapolis in the positive y direction. Those choices may appear capricious, but there is an equal amount of arbitrariness in other maps as well. After all, the world would not change—only maps would change—if the prime meridian passed through San Francisco instead of Greenwich. If you need to make your map conform to the arbitrary conventions of the rest of the world, you can readily do so after the fact. For example, you can slide your map along a world map until San Francisco is at the correct latitude and longitude, and then rotate your map around San Francisco until the axis passing through Washington is correctly oriented. As for placing Minneapolis above or below the x axis: if your choice differs from the standard, just use a mirror image of your map instead of the map itself.



Once three cities have been placed on the map, the arbitrary choices are done with; all other positions are fully determined. For each additional city, the distances from San Francisco and Washington are plotted by drawing circles; whenever the procedure generates two candidate positions—as it almost always does—the ambiguity is resolved by checking the distance to Minneapolis. In that way a complete map is drawn from just the three columns of distances to San Francisco, Washington and Minneapolis; the rest of the mileage matrix is superfluous. Of course there is nothing special about the three chosen cities. The method would work as well with other cities, provided only that they are not all on a line or too close together.

The mileage-to-map algorithm sounds good in principle, but I felt a need to test it on real data. The illustration above is a map of twenty-six U.S. cities, drawn by a computer program based on the algorithm. The pattern of the cities looks familiar, although the ship of state is listing slightly to starboard. An eyeball comparison with a conventional map yields a reasonably close match. For a more systematic check, I took the calculated coordinates of all the cities and regenerated a table of intercity mileages, then compared the resultant table with the original one from which the map was made. Distances to San Francisco and Washington would be expected to match exactly, of course, but there was no such guarantee for all the other distances. The correspondence was remarkably close, with errors generally less than five miles.

I must confess that this successful map was not the first one that my program drew. The first map was based on a mileage table in a Rand McNally road atlas. I ran a preliminary test with just a handful of cities, and the result looked fine. Then I typed in the rest of the numbers and generated another map. In broad outline it looked like the U.S., but on inspection there were some disturbing oddities. Stretching off to the southeast from Washington was an archipelago of cities: Baltimore, Philadelphia, New York, Boston. Meanwhile, Richmond and Norfolk had been transposed to the New England coast. Neither North nor South would approve of this rearrangement. To find the cause I went looking for a bug in the program, but that was not the source of the problem. In a sense the map was correct: It showed a true geography of the nation as measured by highway miles. Because the Great Lakes get in the way, Boston is a longer drive from Minneapolis than Richmond is.

The successful map I drew is based on a table of airline distances from the Coast and Geodetic Survey. In one respect its accuracy is surprising, for it ignores the curvature of the earth. But one can go only so far without spherical trigonometry. When I tried drawing a world map with the same program, the result looked like no planet I have ever visited. Most of the cities were on the perimeter of a large ellipse. Even local constellations of cities could not be trusted: London and Paris were close neighbors, but New York and Chicago were at antipodes.

Producing a map from a mileage table is not a totally frivolous exercise. The algorithm has important applications. It is similar to the method employed in the loran system of marine navigation, developed during the Second World War. A loran receiver calculates the difference in travel time for signals emitted by two or more transmitters of known location. Thus when you fix your position by loran, you do not know the actual distances to the transmitting stations; you know only the difference in distance—how much farther it is to one point than to the other. With this information you can determine that your position must lie somewhere along a hyperbola; a second pair of stations specifies a second hyperbola, and your location is at the intersection of these two curves.

Navigation with the newer Global Positioning System satellites is even more closely related to the map algorithm. Each satellite continually broadcasts reports of its own position in three-dimensional space, along with a highly accurate indication of the time of each report. A receiver that has an accurate time reference of its own can immediately calculate the travel time for each signal and hence the distance to the corresponding satellite. If three satellites are visible, the receiver can determine an unambiguous position on the earth's surface. If a fourth satellite is available, the receiver can make do without the internal time reference.

In the 1970s the biochemist Donald M. Engelman and the chemist Peter B. Moore of Yale University employed the mileage-to-map algorithm in a series of experiments that aimed to decipher the structure of the ribosome, the cellular organelle that translates nucleic acids into proteins. The ribosome is an assembly of some fifty-five protein molecules and three strands of RNA; it is too small for microscopy but too large and complicated for the brute-force methods used to study individual biological molecules. Engelman and Moore set out to learn how the protein molecules are arranged in three-dimensional space.

To do so they assembled ribosomes in which the deuterium isotope of hydrogen replaced ordinary hydrogen atoms in specific pairs of proteins. With neutron-diffraction techniques they then measured the distance between the selected proteins and compiled the measurements into a table analogous to a table of intercity mileages. In this case the reconstruction was fully three-dimensional, and so distance was defined not from the two points at the ends of a baseline but from the three points at the vertices of a base triangle. A fourth point was needed to resolve the ambiguity between positions above and below the base triangle.

Finally, the mapping algorithm also has a place in what may well be the largest single consumer of computing resources outside the defense establishment: the interpretation of seismic survey data for petroleum exploration. The seismic data are records of signals reflected from subsurface structures and received by sensors at the earth's surface. If the sensors could be made to look only straight down, interpreting the data would be comparatively easy: The depth of a structure would be correlated in a fairly simple way with the travel time of the reflected signal. But the sensors are omnidirectional, so that structures at many depths, but all having the same slant range, contribute to a single peak in the data.

Again, the distinctive characteristic of the problem is that distances are known, but not directions. In this case, however, there really is too little information for an easy and efficient solution. The reflected signals bear no labels to indicate their source, and so there is no direct way to correlate signals reflected by the same structure but received by different sensors. In effect, one is given not a table of mileages between cities but merely a list of cities and an unordered list of mileages, with no indication of how they go together. In practice, the seismic data are interpreted through iterative methods: a computer program makes a guess about the subsurface structures, simulates the seismic data that would be generated by those structures, then adjusts the hypothetical features and repeats the entire process until the simulated data resemble the real seismic returns. The equivalent problem in textual analysis might be to reconstitute a book not from a concordance but from a rather sketchy index. ●

BRIAN HAYES is the editor of AMERICAN SCIENTIST.

THE NATIONAL TRANSLATIONS CENTER (NTC)

at the

LIBRARY OF CONGRESS

Get Copies of Translations Selected by Leading Research Scientists and Engineers from Government and Industry...

■ **100% in English** ■ **Only scientific topics** ■ **Only available through NTC** ■ **24-hour turnaround**

English translations available on the world's scientific advances in communicable disease prevention, toxicology, chemical processes, semiconductors, metal treatments, paper and cardboard production, and *much more.* Originally in Japanese, German, Russian and other languages.

Check Our Inventory of Translations Online

The National Translations Center is a clearinghouse for high tech articles, patents, and conference papers. Check the NTC index electronically through Dialog, OCLC, NASA, DTIC, NERAC, ISI, FTD, or CISTI or the print version of *World Translations Index.*

Help Sharpen America's Competitive Edge: Add Your Translations to the NTC

The NTC depends upon voluntary cooperation of federal agencies, private concerns, universities, and individuals who contribute the articles they've had translated. Complete confidentiality is guaranteed for the source of contributed translations.

Write to:
The National Translations Center
Library of Congress
Washington, DC 20541-5017

Telephone: 202-707-0100
FAX: 202-707-6147

Just $35
per translation (includes shipping)