Taylor & Francis
Taylor & Francis Group

Interdisciplinary Science Reviews

ISSN: 0308-0188 (Print) 1743-2790 (Online) Journal homepage: https://www.tandfonline.com/loi/yisr20

‘Software and Scholarship’ — Editorial

Tara Andrews

To cite this article: Tara Andrews (2015) ‘Software and Scholarship’ — Editorial, Interdisciplinary Science Reviews, 40:4, 342-348, DOI: 10.1080/03080188.2016.1165456

To link to this article: https://doi.org/10.1080/03080188.2016.1165456

Published online: 07 Jun 2016.

INTERDISCIPLINARY SCIENCE REVIEWS, Vol. 40 No.4, December 2015, 342–348

‘Software and Scholarship’ — Editorial

Tara Andrews

The thematic focus of this issue is to examine what happens where software and scholarship meet, with particular reference to digital work in the humanities. Despite the some seven decades of its existence, Digital Humanities continues to struggle with the implications, in the academic ecosystem, of its position between engineering and art. The parallel drawn here between software and engineering on the one hand will be less controversial than the one between scholarship and art on the other — to what extent should these really be cast as parallel? Mathematicians are scholars, and physicists; so too are philosophers, theologians, biologists, and historians. Each of these fields tends to evoke a very different perception of the balance between logic and intuition, however; some are perceived as pursuing a form of unambiguous truth within the systems they inhabit, others as pursuing a more pluralistic understanding. Although the lines between disciplines are so often more arbitrary than real, and it is difficult to draw a firm distinction that will satisfy both the mathematicians and the philosophers, the label of ‘scholar’ seems to be something that scientists are keen to avoid.[1]

For indeed the divide between ‘science’ and ‘humanities’ is not as sharp as is often assumed in discussion. The supposed clarity of difference is exacerbated in the Anglophone world where vocabulary sharpens the division, and both words are needed, in opposition to each other, to translate the sense of the German ‘Wissenschaft’ and its cognates in other languages. Perhaps due to the feeling — endemic among scholars in the humanities — that their disciplines are often compared against the sciences and found to be wanting, or the pronounced focus of policymakers on ‘STEM’ subjects that bring with them an apparent devaluing of humanities subjects, or simply a sense that scientific truth is not a thing that ought to be sought after in English departments, quite a bit of reflection has occurred throughout the humanities about the proper function of what is now usually termed ‘digital humanities’ with respect to the wider humanities, and in particular whether the adoption of the quantitative and somewhat empirically natured digital methods will bring more of the STEM-like respect that is perceived to be lacking among policymakers and the public.

And yet this sort of soul-searching in the humanities tends to elide the issues, discussion, and debates that have indeed cropped up in fields considered by most to be ‘pure science’, such as physics and biology, and it leaves aside a discussion of why and how computation has been adopted in such an evidently smooth trajectory by fields that belong unarguably to humanities departments, such as

 DOI 10.1080/03080188.2016.1165456

linguistics and archaeology, as well as many fields in that middle ground between humanities and science known as the social sciences.

What is often not acknowledged is that even in the so-called hard sciences, the use of computers was (and is) a gradual phenomenon, often marked by heated methodological debate among its practitioners. Galison (1992) provides an admirable case study for physics, a field that is considered by most to be one of the paragons of 'big data'-enabled computation-heavy sciences. He describes the enthusiasm that took hold in the 1950s for experiments in particle physics that required some capability for processing images in bulk; laboratories that wished to do this sort of research must have either an ever-expanding army of skilled lab technicians to examine the photos and select those that were 'interesting', or else some sort of machine that could do this selection automatically. This quickly led to a split over which methodology was more appropriate — an 'interactive' approach in which the work of the machine was monitored and channelled by a human, or a 'segregated' approach in which humans did the necessary preparatory work before leaving the ultimate detection of interesting patterns to the machine. At issue for these communities of physicists was a question that is very familiar in the humanities today: is it better to rely on human intuition, with its specialised capability for pattern recognition, for the detection of scientifically interesting phenomena, or does the logical rigour of a machine free from the effects of confirmation bias produce better results?

Perhaps the most striking feature of Galison's study was the strong parallels that can be drawn between the experiences of particle physicists and those of practitioners who use computational methods in the humanities; here I will take philology, a field with which I am perhaps the most familiar, as an example. As a field, philology is a very instructive example of how a tension can crop up between logic and intuition when computational methods are applied to research. Different practitioners will describe their work in different ways — a classical philologist will describe the purpose of her work very differently than a modern textual scholar, both in terms of the working methods and of the expected scholarly outcome. Nevertheless the commonality could be expressed thus: philologists are trying to make a maximum of sense out of the texts with which they work, be they second- or third-century fragments of the Bible, medieval translations of animal fables, or successive drafts of a nineteenth-century classic.

'Sense' is a broad word, however. It appeals both to our reason and our emotions; it relies on deduction and interpretation in equal measure. Therein lies the great challenge of philology in the digital age: suddenly we have a plethora of possibilities to create tools that support our deductive capacities. As a result a great number of tools, techniques, and methods have sprung up around digital philology. In many ways the computer is an indispensable aid to the work of comparing, reconciling, and producing new versions of texts, but the degree to which philology is a computationally tractable task remains open to fierce debate, and so the different tools and methods have different balances between the rational and the interpretative, between quantification and qualitative work. The less rigorously deterministic art of interpretation is not directly supported in this world of computer tools. Semantic meaning is usually not an inherent part of the

computational model; although many tools do allow room for expressions of this meaning, to be retained and displayed to the human user, it tends to be difficult or impossible to capture that meaning in a way that lends itself to any sort of programmatic analysis.

Thus a divide has sprung up among philologists, very similar to the one Galison has described in particle physics. On one side, the process is thoroughly interactive. The machine is merely a sort of repository of raw information, to be examined by the scholar when necessary, and a useful means to prepare the result of the scholar's work for publication. Software packages such as Classical Text Editor and the erstwhile COLLATE bear witness to this approach: the job of the computer in these cases is to keep a record of the textual facts, and to allow the scholar to arrange these facts, interactively, in a form familiar enough that the use of the computer is more of a convenient filing system than a methodological departure. On the other end of the spectrum are tools such as CollateX for textual collation, or statistical methods for the construction of a stemma (that is, a graph of the copying history of the manuscripts of a text) that have a significant overlap with evolutionary biology and historical linguistics. In these latter cases the work is much less interactive; rather, the data are fed to computer algorithms whose purpose is not to organise, but to analyse it. The mark of success of these algorithms, in the eyes of their creators and users, is the extent to which their results are accepted by scholars without interaction, and with minimal alteration.

The question, as it is normally presented in discussions among philologists, is one of the extent to which human judgment is an indispensable part of scholarship, or alternatively to which human bias can lead scholars into error. Is the computer a 'dumb machine' that cannot be involved in questions of judgment and interpretation, or is it an impartial aggregator of evidence that cannot be easily led into confirmation bias? Here the debate touches uncomfortably on the pressure felt within the humanities to be 'more like science' — more quantifiable, more reproducible, more falsifiable, more logically rigorous.

But the humanities have a troubled history with empiricism, not helped by the fact that empiricism and positivism are easily conflated. A few years ago I was speaking to a respected scholar of pre-Islamic Arab history, who much to my surprise began to lament the drive toward formalisation and scientific method in history. The trouble, he declared, was that just about anything can be 'proven' by marshalling the correct evidence and ignoring what does not fit, and this 'scientific method' as he understood it was simply a process of legitimation of an argument that made it difficult, rhetorically, to refute.

While this scholar did not fully grasp the principle of falsification that underlies a correct application of the scientific method, his experience of scholarly arguments that draw on the rhetoric of scientific method to produce positivistic arguments that are anything but testable is perfectly real. I myself have seen several examples of this sort of misguided empiricism, with its roots in a positivistic view of history. The debate between positivism and historicism and the development of modern critical theory has been a particular mark of twentieth century historical studies, but to work the scientific method into these debates is indeed problematic. The canonical scientific method, with its emphasis on data

collection and its presumption that an experiment can be constructed that will clearly confirm or falsify any given hypothesis, is often a poor fit for fields within the humanities that rely on incomplete data and evidence that is open to interpretation, such as history, or that work entirely in the realm of intellectual creativity and hermeneutics, such as philosophy or literature. The advantage of a critical theoretical approach in these fields is that it provides a framework for a logical rigour that lends substance to argument, without relying for its function on the production of evidence that may well never have existed.

Into this environment comes computational method and the digital humanities, along with the pressure for the somewhat subjective and introspective humanities to become 'more like science' — more quantifiable, more reproducible, more amenable to a cost-benefit analysis for any particular avenue of inquiry. Many modern scholars are concerned about the perception that their discipline and its methods are seen as out-dated and superfluous to the modern global society and economy; it is very easy to see how their suspicion is easily aroused against digital methods as a neat solution to this credibility problem. Is this influx of digital method and excitement about 'big data' simply positivism in a new form? The question is asked repeatedly in venues such as the Humanist mailing list: how, precisely, do computational methods add value to the humanistic process of rhetorical reasoning, productive theory, critical interpretation, and hermeneutics?

Despite the doubt and suspicion, and despite the serious practical difficulties of using computational methods in the humanities (such as the questions of access, preservation, and sustainability) it is much easier to imagine that computational methods will become mainstream than that they will be abandoned. Although some information and interpretation is easily lost when a hand-annotated copy of a particular edition of a novel is reduced to a digital transcription, philologists will go on working with digital surrogates of texts. Although it is extremely difficult to create a computer-based visualisation of a phenomenon in the humanities without falling into the trap of artificial precision and misguided normativity (c.f. Drucker 2011), visualisation has proven to be far too useful a tool for humanists of all stripes to investigate a collection of data and look for patterns. The allure of statistical methods to discover patterns in an increasingly large corpus of digitised text, records, and other artefacts is irresistible, even as the statistical methods tend to expect us to treat as anomalous any data that does not conform to a neat pre-defined pattern.

In all of these cases we are, implicitly or explicitly, constructing models of our objects of study; all such models contain a certain amount of domain knowledge, and all of our computational tools operate on the basis of that domain knowledge. These facts, self-evident though they are, directly give rise to perhaps the two most pressing concerns of scholars in the humanities when faced with digital methods. First, the black box question: can we truly know what models, assumptions, and inferences are made within the source code of a particular software tool? If so, how? If not, how can we justify a blind use of it? Second, the question that has led finally to the theme of this special issue: how do we evaluate what contribution to scholarship has been made by the creator of that tool?

Perhaps, however, it would help to consider parallels to the traditional process of creation, adaptation, and evaluation of knowledge and interpretation within the humanities. We might imagine each field within which we work as a system of its own, with its own axioms and emphases and consequences out of which our interpretations are derived and our conclusions are constructed — these are the theoretical and abstract tools of our trade. In any field, we will need some way to evaluate the scholarly contribution of any theory that is advanced, or any claim that is made, though these generally come in a discursive written form. Moreover, we will have to do this without a full knowledge of the axioms and theoretical frameworks that this scholarship might draw on. In the case of software, the sticking point for many scholars is that they are not qualified to understand how the programme is doing what it does, or to discover where its flaws might be. And yet in the case of discursive scholarship we often find ourselves needing to do that very thing.

Any field worthy of that designation is far too vast and complex for a single person to understand it all in both breadth and depth. An individual scholar cannot hope to examine everything by herself, and in order to continue with her research, whether as a historian, archaeologist, literary scholar, scholar of art, philosopher, or anything else, she will always need to rely to some extent on exactly the sort of conceptual 'black boxes' that are causing such consternation in DH at the moment. I am a historian first and a textual scholar second, and claim no expertise at all in the other fields; I do not have the practical capacity, for example, to engage with the axioms and methodological theory of art history. To what extent, then, should I be allowed to incorporate the findings and conclusions of art historians into my own research? To what extent should an archaeologist turn his talents to textual historical source forensics, rather than relying on the work I have done on that topic, in order to identify and make sense of material artifacts?

And so we come back to software systems, which have grown to be large enough that we cannot claim fully to comprehend their entirety, and this makes some scholars uncomfortable. There are valid reasons for discomfort, of course: code together with its execution environment has the capability to transform itself along the way into something unexpected, and software already has a very poor reputation for the sort of reliability, safety, and stability that is expected in other engineering disciplines (Leveson and Turner 1993; Bogost 2015). This nevertheless has parallels to the situation we have always faced in any discipline that must draw on another, or that must draw on assertions that would be impractical to check. When a digital philologist writes a system for variant text analysis she must take on faith that the database software works as advertised, that the XML parser does its job properly, that the Javascript libraries for manipulating SVG will put the correct image in the web browser. As she works with these tools to build her own system, she will naturally have some expectations concerning the results she will see; if the results are substantially different then she will pause and try to bridge the gap between expectation and reality.

But here perhaps there is a difference between historical disciplines and software. We typically understand (more or less) the conclusions of art historians or

archaeologists or textual scholars by reading their arguments and reflecting on them, applying the conclusions to the mental models we have built up from our own knowledge and experience, and considering the consequences of these conclusions. In theory one could do the same with computer code — read it, mentally run through the algorithms therein, understand the argument, even read the source code for the sub-systems upon which other code depends — but in practice no one does this for more than a very small subset of any given programme. And no wonder — code in its static text form is not a good medium for human-to-human communication! This is why, as any software developer well knows, he or she will be pressured to document, document, document. By and large it is the documentation of a programme that sets our expectations for what the code does. Only then do we grasp its logic and its argument, and only then can we put it to the test. In theory a scholar could take the enormous trouble to replicate the software-encoded scholarship, but in practice we rely on the summary — the documentation — to set our expectations.

The other difference that arises with software comes in how we put the work to the test. Where in the historical and philosophical disciplines we read and reflect, using the new information as a new set of mental building blocks, in computing one must take the 'building block' metaphor rather more literally. If a scholar wishes to evaluate the code, he must try to make something with it. If the result is unexpected, then the scholar must take a closer look at the code and the documentation, and try to work out whether the fault lies in his expectations or in the building block itself.

Perhaps the dissonance we see surrounding the digital humanities arises from the sense that, as long as scholars are building software systems — making things — they are not evaluating them or critiquing them. This corresponds to the 'tensions between theoretical critique and productive theory' that were observed by Hayles (2012), writing about the relationship between the digital humanities and the field of comparative media studies. It is, however, perhaps a false dichotomy after all; to incorporate others' work into one's own should necessarily involve a certain amount of study and critique of that work.

The papers written in response to the call for this special issue look at the relationship between software and scholarship from three valuable perspectives. Joris van Zundert reflects on the role that authorship plays both in traditional scholarship and in the creation of computer programmes, and how the misappropriation of this authorship creates grave difficulties in the evaluation of its contribution. Rebecca Sutton Koeser writes about the dynamics of the personal relationship between developers and researchers, and broaches some of the same topics as Van Zundert, from a complementary perspective. Aris Xanthos dissects a particular piece of text-research software to examine the scholarship inherent in the tool, and in so doing provides a case study for how this task might be done.

I will end with my own reflection on this topic of evaluation. Peer review of scholarly works of software continues to pose a particularly vexed challenge — who is qualified to carry it out? By what criteria should they evaluate the work? How will the reviewing process essentially function? The amount of time and effort it takes for a peer reviewer to read a discursive journal article and think

about its implications tends to exactly fill the amount of time the reviewer has to devote to the task. The act of making proper use of a piece of software, on the other hand, and especially the act of incorporating it into something else being built, has a rather higher minimum cost in terms of time and effort. The number of people who are in a position to provide a good review of any particular piece of scholarly software — those who actually have a use for the software (or at least have suitable digital data on hand to experiment with) as opposed to those who might be able to spare a little theoretical consideration but no more — will always be rather small. Yet the alternative — to disregard the scholarly contribution of a piece of software on the basis that it is too difficult to evaluate — is perhaps the greatest crippling threat to the future of the digital humanities. It is my great hope that the papers collected in this issue will prompt scientists and scholars throughout the wider world of the digital humanities to reflect more productively on the issue of how the authors of scholarly software can be truly and substantially be given the recognition that their work deserves.

Note

[1] E.g. the reference of Galison (1992, 252) to a 'dreadful transformation from scientist to scholar' that has 'frightened' some physicists.

ORCID


Tara Andrews http://orcid.org/0000-0001-6930-3470

References

Bogost, Ian. 2015. Programmers: stop calling yourselves engineers. *The Atlantic*, November 5. http://www.theatlantic.com/technology/archive/2015/11/programmers-should-not-call-themselves-engineers/414271/.

Drucker, Johanna. 2011. Humanities approaches to graphical display. *Digital Humanities Quarterly* 5 (1). http://www.digitalhumanities.org/dhq/vol/5/1/000091/000091.html.

Galison, Peter. 1992. Fortran, physics, and human nature. In *The invention of physical science*, ed. Mary Jo Nye, Joan L. Richards, and Roger H. Stuewer, 225–260. Boston Studies in the Philosophy of Science 139. Netherlands: Springer. http://link.springer.com/chapter/10.1007/978-94-011-2488-1_10.

Hayles, N. Katherine. 2012. How we think; the digital humanities. In *How we think: digital media and contemporary technogenesis*, 1–54. Chicago, IL: University of Chicago Press. http://www.press.uchicago.edu/ucp/books/book/chicago/H/bo5437533.html.

Leveson, N. G., and C. S. Turner. 1993. An investigation of the therac-25 accidents. *Computer* 26 (7): 18–41. doi:10.1109/MC.1993.274940.