The Open Digital Archaeology Textbook Environment

Shawn Graham, Neha Gupta, Jolene Smith, Andreas Angourakis, Michael Carter, & Beth Compton

2018-07-03

Contents

notice **5**

About the Authors **7**

Getting Started **9**
Students: How to use this text . . . 9
How to contribute changes, or make your own version . . . 10
Colophon . . . 11

Welcome! **13**

1 Going Digital **15**
1.1 So what is Digital Archaeology? . . . 17
1.2 Project Management Basics . . . 23
1.3 Github & Version Control . . . 25
1.4 Open Notebook Research & Scholarly Communication . . . 30
1.5 Failing Productively . . . 37
1.6 The Ethics of Big Data in Archaeology . . . 40

2 Making Data Durable **43**
2.1 Designing Data Collection . . . 45
2.2 Cleaning Data with Open Refine . . . 45
2.3 Linked Open Data and Data Publishing . . . 45
2.4 Introduction to Digital Libraries, Archives & Repositories . . . 45
2.5 Command Line Methods for Working with APIs . . . 46

3 Finding and Communicating the Compelling Story **47**
3.1 Statistical Computing with R and Python Notebooks; Reproducible code . . . 47
3.2 D3, Processing, and Data Driven Documents . . . 48
3.3 Storytelling and the Archaeological CMS: Omeka, Kora . . . 49
3.4 What is Web Mapping? . . . 49
3.5 Place-based Interpretation with Locative Augmented Reality . . . 53
3.6 Virtual Archaeology . . . 54
3.7 Archaeogaming . . . 54
3.8 Social media as Public Engagement & Scholarly Communication in Archaeology . . . 54

4 Eliding the Digital and the Physical **55**
4.1 3D Photogrammetry & Structure from Motion . . . 55
4.2 Computer Vision and Archaeology . . . 62

5 Digital Archaeology's Place in the World **65**
5.1 Marketing Digital Archaeology . . . 65
5.2 Sustainability & Power in Digital Archaeology . . . 65

6 On the Horizons: Where Digital Archaeology Might Go Next **67**

References **69**

notice

This volume goes hand-in-glove with a computational environnent that uses Jupyter Notebooks coupled with the Binder service as a way of serving and running the notebooks online in a browser. This relieves both instructors and students of the problems of installing software in different environments and the troubleshooting that this entails. Instead, students and instructors can concentrate on the learning and writing literate code that explores archaeological issues. To launch the ODATE notebooks, please go to the basic notebook repository and hit the `launch binder` button (after reading the information there).

THIS IS A DRAFT VERSION; it's not even version 0.1 yet. It is filled with errors, omissions, and non-sequiturs.

Figure 1: This book will, with time, have many branches and off-shoots. It is meant to be customized and expanded; there will never be a canonical version. Photo by Brandon Green, *unsplash.com*.

About the Authors

Shawn Graham

Shawn Graham trained in Roman archaeology but has become over the years a digital archaeologist and digital humanist. He is currently an associate professor in the Department of History at Carleton University in Ottawa Canada. He keeps an open lab notebook of his research and experiments in digital history and archaeology at his research blog, electricarchaeology. He can be found on Twitter at electricarchaeo.

Neha Gupta

Neha Gupta is a broadly trained archaeologist and recent postdoctoral fellow at Memorial University. Her research programme addresses geospatial and digital methods in post-colonial and Indigenous archaeology. Her specialties include geovisualization and GIS, landscape and settlement archaeology and the archaeology of India and Canada (Ontario). She recently launched MINA | Map Indian Archaeology, a public digital project to promote the archaeology of India and to encourage collaboration in the development of digital tools and technologies appropriate for archaeology. Recent scholarship centers on themes of colonial practices, web maps, Indigenous peoples and archaeology's relationship with society.

Michael Carter

Michael Carter is an Assistant Professor in the School of Creative Industries at Ryerson University. He is also Director of Industry Relations - Master in Digital Media Program/Yeates School of Graduate Studies. His research is primarily centred around the Theory, Method and Practice of Virtual Archaeology, which flows naturally from his previous career in animation and visual special effects. More about his research can be explored on his Research Gate profile

Beth Compton

Beth Compton is a PhD student at the University of Western Ontario's Department of Anthropology. Her research examines critically the implications and assumptions of digital approaches to repatriation, and uses archaeological ethnographic approaches to evaluate artifact sharing policies. She study archaeological representation and replication, particularly the use of digital photography, 3D modeling, and 3D printing. She is interested in all aspects of meaning-making and knowledge sharing in digital archaeological heritage practice. She is a Co-Founder of the DH Maker Bus project and is an Ontario Trillium Scholar.

0.0.1 Jolene Smith

0.0.2 Andreas Angourakis

Andreas Angourakis is a PhD student at the Department of History and Archaeology, University of Barcelona. He is a generalist that underwent training in humanistic disciplines, social sciences, and biology. He is self-taught in software and programming languages, including R and NetLogo. His research in archaeology has been focused on the development of simulation models of socio-ecological dynamics in the past, mainly using agent-based modeling, and assembling multivariate statistical protocols for analyzing and interpreting archaeological data. As digital archaeology's bones of the trade, he believes that creativity and science are unmistakably intertwined. He can be found on Twitter as (**???**)(https://twitter.com/AndrosSpica), GitHub as Andros-Spica, and is present on both Research Gate and Academia.

Editorial Board

Getting Started

We wrote this text with a particular kind of student in mind. We imagined this student as having already taken a first year introductory course to archaeology, of the kind that surveys the field, its history, and its principle methods and theoretical positions. Very few courses of that kind include any depth in digital methods and theory, which is understandable when we look at the incredible variety of archaeological work, skills, and interests! Digital work is every bit as diverse as other kinds of archaeology, but it also presents its own particular challenges. One of these is the anxiety that comes when one first approaches the computer for anything more complex than word processing or a bit of social media. 'What happens if I break it?'; 'I'm not techy!'; 'If I wanted to do computers, I wouldn't have gone into this!' are all actual student concerns that we have heard in our various classrooms.

It'll be ok.

We take a pedagogical perspective that focuses on the learning that happens when we make things, when we experiment or otherwise play around with materials and computing, and especially, when/if things break. It's a perspective that finds value in 'failing gloriously', in trying to push ourselves beyond our comfort level. The only thing that you will need therefore to be successful in learning some of the basic issues around digital archaeology is a willingness to consider why things didn't work the way they ought to have, and a web browser. We built this textbook with its very own digital archaeology computer built right in! There's nothing you can break on your own machine, nor does your machine have to be very powerful.

Students: How to use this text

Each section in this book is broken down into an overview or discussion of the main concepts, and then followed up with skill-building exercises. The computational environment - provided via a Jupyter notebook- is running on someone else's servers. When you close the browser, it shuts down.

Warning! The computational notebooks that we provide that are running in the [binder][http://mybinder.org] environment *will time out* after **ten** minutes' inactivity.

Your work can be saved to your own machine and reloaded into the environment later, or you can 'push' your changes to your own online repository of work (to learn how to push work to a Github repository, see the sections on Github & Version Control and Open Notebook Research & Scholarly Communication so you'll be able to get your work and data out of the Jupyter Notebooks and onto space that you control). The best way to use this book is to make sure you have at least one hour blocked out to read through a section, and then two hours to go through the section again if you're working on the exercises. We find that the work goes better if those blocks are interspersed with breaks every twenty minutes or so.

Do you notice that stripe down the side of the screen at the right? That's a tool-bar for annotating the text, using a tool called `Hypothes.is`. If you highlight any text (go ahead, highlight that phrase right now by right-clicking and dragging your mouse!) a little pop-up will ask you if you want to annotate or highlight the text. If you choose annotate, a writing pane will open on the right. Using the Hypothesis tool requires a reader to create a login and account with Hypothesis, which is managed by the Hypothesis site, not us.

By default, such annotations are made public. Private annotations can only be viewed by the particular individual who made them. All annotations (both public and private) have their own unique URL and can be collated in various ways using the Hypothesis API (here's an example). Please tag your annotation with `odate` to allow easy curating of the public annotations.

Please note that any public annotations can be read by any other reader. These can also be responded to, as well - which might make a great classroom activity! A class can create group annotations which are only visible to participants in that group (instructions here). Annotation is a tool for research; personal reaction to anything we've written in ODATE should be done via the reader's blog while leaving an annotation on ODATE linking to the blog piece. Because the API or 'application programming interface' for Hypothesis allows one to retrieve annotations programmatically, there is a growing world of scripts and plugins for managing or retrieving those annotations. Kris Shaffer has created a Wordpress plugin to pull annotations to a new Wordpress post (details are linked here), which might be another great option for a class blog working through ODATE.

Over time, the parts of the text that are heavily annotated will look as if someone has gone over them with a yellow highlighter. You can use this to help guide your reading - perhaps that's a part where many people had problems, or perhaps it's a part that sparked a really interesting discussion! Group annotation like this promotes 'active reading', which means that you're more likely to retain the discussion.

Finally, if you'd rather not read this as a web page, you can grab a pdf copy by pressing the download button above (the downwards-facing arrow icon) and printing out just the bits you want or need. If you'd rather read this text via an e-reader or iBooks or similar, the download link will also give you an ePub version. Individuals who use a screenreader or other assistive device might prefer to work with the pdf or epub versions. Please do let us know if there is any way we can make this text more accessible for users with particular needs. Since this text is fundamentally a series of plain-text files that we then manipulate to create these different outputs, it should be straightforward for us to adapt accordingly!

How to contribute changes, or make your own version

Perhaps your professor has assigned a portion of this text to your class, with the instruction to improve it. Do you see the edit button at the top of the screen (it looks like a little square with a pencil)? If you click on that, and you have an account on Github (and you're signed in), you will grab a copy of this entire text that you can then edit. If you want, you can also make a `pull-request` to us, asking us to fold your changes into our textbook. We welcome these suggestions! Since this book has a creative-commons license, you are welcome to expand and build upon this as you wish, but do cite and link back to the original version.

Instructors can take a copy of this repository as well, and re-edit or recombine the text in whatever ways will serve their learning goals best. We knit the markdown together inside R Studio with the Bookdown package. References are kept in the bibtex format in a `.bib` file. (For more information, see the colophon below) . ## How to access and use the computational environment {-}

There are two options. We created a set of Jupyter Notebooks coupled with the Binder service as a way of serving and running the notebooks online in a browser. This relieves both instructors and students of the problems of installing software in different environments and the troubleshooting that this entails. Instead, students and instructors can concentrate on the learning and writing literate code that explores archaeological issues. To launch the ODATE notebooks, please go to the basic notebook repository and hit the `launch binder` button (after reading the information there). Students can use the built in file manager to download or upload notebooks into the environment.

Alternatively, instructors might want to use the DHBox, which is a linux based computer accessible through a browser. The DHBox has a bit more flexibility in the sense that the terminal available inside is not as locked down as what is available through Binder. The DHBox perists for as long as a month, and so students don't have to be as mindful of saving their work to their *own* machines as perhaps with the Binder service. At the DHBox site, students create a new user account. They should select the maximum amount of time available

for the box. Each time a person logins, the box will tell them how much time remains for the account. Students can use the built-in file manager to download their work from the box to their own machine.

It is worth noting that any jupyter notebook can be read within either environment (and of course, students can install Jupyter onto their own machines if they so desire). Jupyter notebooks can be written in both python or R. Jupyter notebooks are quickly becoming a standard for 'literate programming', where the reflective text and the code are interwoven for the purposes of reproducibility and replicability.

Colophon

This text was created using the Bookdown package for R Markdown. R Markdown is a variant of the simple Markdown format created by John Gruber. That is to say, at its core this text is a series of simple text-files marked up with simple markers of syntax like `#` marks to indicate headings and so on. R Markdown allows us to embed code snippets within the text that an interpreter, like R Studio, knows how to run, such that the results of the calculations become embedded in the surrounding discussion! This is a key part of making research more open and more reproducible, and which you'll learn more about in chapter one.

The sequence of steps to produce a Bookdown-powered site looks like this:

1. create a new project in RStudio (we typically create a new project in a brand new folder)
2. run the following script to install Bookdown:

```
install.packages("devtools")
devtools::install_github("rstudio/bookdown")
```

3. create a new textfile with `metadata` that describe how the book will be built. The metadata is in a format called `YAML` ('yet another markup language') that uses keys and values that get passed into other parts of Bookdown:

```
title: "The archaeology of spoons"
author: "Graham, Gupta, Carter, & Compton"
date: "July 1 2017"
description: "A book about cocleararchaeology."
github-repo: "my-github-account/my-project"
cover-image: "images/cover.png"
url: 'https\://my-domain-ex/my-project/'
bibliography: myproject.bib
biblio-style: apa-like
link-citations: yes
```

This is the only thing you need to have in this file, which is saved in the project folder as `index.Rmd`.

4. Write! We write the content of this book as text files, saving the parts in order. Each file should be numbered `01-introduction.Rmd`, `02-a-history-of-spoons.Rmd`, `03-the-spoons-of-site-x.Rmd` and so on.

5. Build the book. With Bookdown installed, there will be a 'Build Book' button in the R Studio build pane. This will generate the static html files for the book, the pdf, and the epub. All of these will be found in a new folder in your project, `_book`. There are many more customizations that can be done, but that is sufficient to get one started.

Welcome!

Digital archaeology as a field rests upon the creative use of primarily open-source and/or open-access materials to archive, reuse, visualize, analyze and communicate archaeological data. This reliance on open-source and open-access is a political stance that emerges in opposition to archaeology's past complicity in colonial enterprises and scholarship; digital archaeology resists the digital neo-colonialism of Google, Facebook, and similar tech giants that typically promote disciplinary silos and closed data repositories. Specifically, digital archaeology encourages innovative, reflective, and critical use of open access data and the development of digital tools that facilitate linkages and analysis across varied digital sources.

To that end, this document you are reading is integrated with live open code notebooks that can be re-used, altered, or extended. Part of our inspiration comes from the 'DHBox' project from CUNY (City University of New York, (link), a project that is creating a 'digital humanities laboratory' in the cloud. While the tools of the digital humanities are congruent with those of digital archaeology, they are typically configured to work with texts rather than material culture in which archaeologists specialise. The second inspiration is the open-access guide 'The Programming Historian', which is a series of how-tos and tutorials (link) pitched at historians confronting digital sources for the first time. A key challenge scholars face in carrying out novel digital analysis is how to install or configure software; each 'Programming Historian' tutorial therefore explains in length and in detail how to configure software. The present e-textbook merges the best of both approaches to create a singular experience for instructors and students: a one-click digital laboratory approach, where installation of materials is not an issue, and with carefully designed tutorials and lessons on theory and practice in digital archaeology.

This is not a textbook about learning how to code. Rather, it is about instilling the habits of thought that will enable success when confronted with digital novelty, the habits of thought that will enable you to determine how to work with digital materials, and the habits of thought that permit you to see where and when digital approaches will make the difference in your research. Skills change; techniques evolve; new tools emerge. Habits of thought are hard to cultivate but have staying power!

Through this textbook, we aim to offer a learners'-perspective-view on digital methods in archaeology, that is, how we might think with, and through, digital sources of information, digital tools and technologies and their relationship with society. We are deeply aware of how rapidly both digital sources and technologies can change, particularly on the Web; we therefore present this e-textbook and open-learning environment as a guide to best practices when working with available digital data and digital tools, what kinds of analysis are possible, how to perform these analytical techniques, and how you might publish your data, making them re-usable for another scholar and ethics and ethical issues in doing digital archaeology.

We have not elected to try to *cover* every possible topic. By design, this book is meant to grow, branch, and change with time. It is meant to foster *uncoverage* and be used to supplement or complement an instructor's own situated approach. Annotate the text using the Hypothes.is. Take it to bits. Use and adapt the parts that make most sense in your own particular learning context.

Chapter 1

Going Digital

> Digital archaeology should exist to assist us in the performance of archaeology as a whole. It should not be a secret knowledge, nor a distinct school of thought, but rather simply seen as archaeology done well, using all of the tools available to and in better recovering, understanding and presenting the past. In the end, there is no such thing as digital archaeology. What exists, or at least what should exist, are intelligent and practical ways of applying the use of computers to archaeology that better enable us to pursue both our theoretical questions and our methodological applications. (Evans, Daly, and MyiLibrary 2006)

While we agree with the first part of the sentiment, the second part is rather up for debate. We believe that there *is* such a thing as digital archaeology. Digital tools exist in a meshwork of legal and cultural obligations, and moreso than any other tool humans have yet come up with, have the capability to exert their own agency upon the user. Digital tools and their use are not theory-free nor without theoretical implications. There is no such thing as neutral, when digital tools are employed. This is why digital archaeology is - or should be - a distinct subfield of the wider archaeological project.

In a conversation initiated on Twitter on March 10, 2017, Graham asked the question (the thread for which discussion starts here), 'is digital archaeology the same as using computers in archaeology?' The resulting conversation ranged widely over everything from the topic of study to the ways in which computational power enables the researcher to ask questions that were not previously feasible to ask. Other researchers sounded a note of caution against the kind of 'technological fetishism' that digital work can often fall pray to, especially given the larger issues of gender and 'solutionitis' that emerge given the white, 20-35 year old demographic of many tech workers (for criticisms of technological solutionism or utopianism in archaeology, see the work of Colleen Morgan (2012) Joyce, Tringham, Morozov, Kansa). Others sounded a warning that to think of digital archaeology as something distinct from archaeology risks 'going the way of DH' and instead appealed for a holistic understanding.

Hanna Marie Pageau succintly captured these issues, when over a series of tweets beginning here she wrote,

> 'Digital archaeology has an obvious digital component. However, saying it's simply using a computer is like saying being a computer scientist means you use a computer to do science. There is an implied addition [to the] topic of specific methods that brings you from an archaeologist using a computer to being an archaeologist who studies digital archaeology. I would argue that archaeogaming is the most straight forward example. Because while gaming is usually thought of as digital, it could study table top gaming and not technically be digital in nature. However if you're studying ethics of representation in games you're going from just using a computer as a tool to it being THE medium.'

In which case, an important aspect of digital archaeology that differentiates it from the use of computing power to answer archaeological questions is this question of purpose. In this section, we take up this question beginning with the question of *teaching* digital approaches. We progress by suggesting that digital archaeology

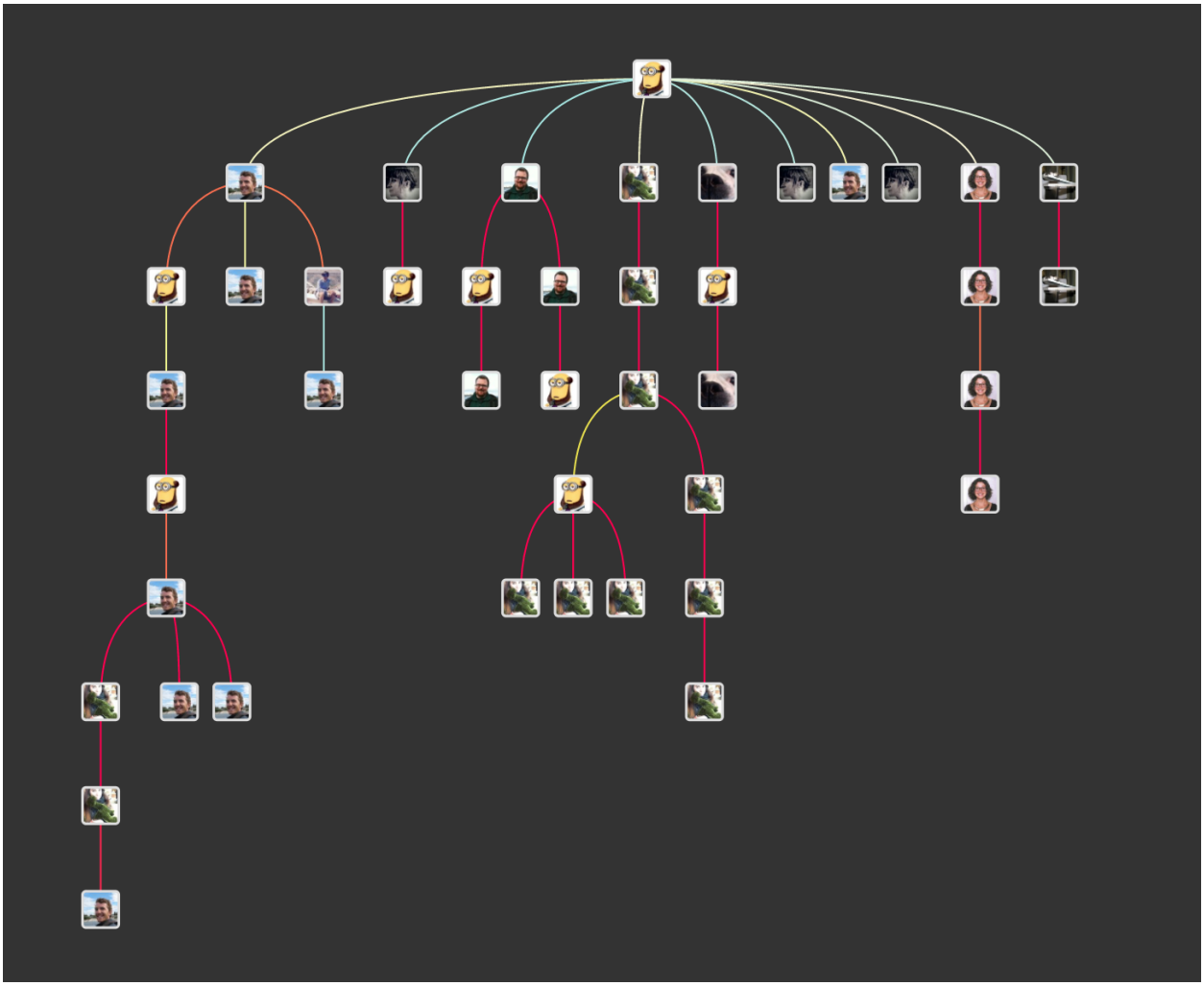


Figure 1.1: A visualization of the conversation tree on Twitter from Graham's query

is akin to work at the intersection of art and public archaeology and digital humanities. We provide you the necessary basics for setting up your own digital archaeological practice. Entrance into the world of digital archaeology requires organizational ability and facility with versioning files. It is allied with the practice of open notebook science, and it attempts to future-proof by using the simplest file formats and avoiding proprietary software where possible. These are the basics on which the rest of digital archaeological practice is founded.

1.1 So what is Digital Archaeology?

If you are holding this book in your hands, via a device or on paper, or looking at it on your desktop, you might wonder why we feel it necessary to even ask the question. It is important at the outset to make the argument that digital archaeology is not about 'mere' tool use. Andrew Goldstone in *Debates in the Digital Humanities* discusses this tension (Goldstone 2018). He has found (and Lincoln Mullen concurs with regard to his own teaching,(Mullen 2017)) that our current optimism about teaching technical facility is misplaced. Tools first, context second doesn't work. Alternatively, theory first doesn't seem to work either. And finally, for anything to work at all, datasets have to be curated and carefully pruned for their pedagogical value. We can't simply turn students loose on a dataset (or worse, ask them to build their own) and expect 'learning' to happen.

Our approach in this volume is to resolve that seeming paradox by providing not just the tools, and not just the data, but also the computer itself. Archaeologically, this puts our volume in dialog with the work of scholars such as Ben Marwick, who makes available with his research the code, the dependencies, and sometimes, an entire virtual machine, to enable other scholars to replicate, reuse, or dispute his conclusions. We want you to reuse our code, to study it, and to improve upon it. We want you to annotate our pages, and point out our errors. For us, digital archaeology is not the mere use of computational tools to answer archaeological questions. Rather, it is to enable the audience for archaeological thinking to enter into conversation with us, and to *do* archaeology for themselves.

Digital archaeology is necessarily a public archaeology. This is its principal difference with what has come before, for never forget, there has been at least a half-century of innovative use of computational power for archaeological knowledge building.

Geospatial, digital and Web-based tools are now central to carrying out archaeological research and to communicating archaeological information in a globalized world. Until recently, the accumulation and effective management of digital archaeological data have been the primary focus of archaeologists (Evans and Daly 2006). Under this model, scholars emphasize the 'integration' into archaeology of computing technologies, and how, by utilizing current available computing memory and processer speed, one does archaeology, only better (Daly and Evans 2006: 1). This situation in turn demonstrates the 'marriage between the two', archaeology and computing (Daly and Evans 2006: 2).

For Evans and Daly (2006), writing in the first decade of the 21st century, digital archaeology was synonymous with the use of Information and Communication Technology or ICT, and reflected wider efforts at that moment to transform education through newly available digital tools. Some scholars and policy makers believed that digital technologies were the answer to pressing global social issues such as poverty, a point that we will discuss later.

More recently, in his inaugural editorial for the open-access journal, Frontiers in Digital Humanities, Costopoulos (2016) argues that 'digital archaeology has been [here] a while'. Computing in archaeology, that is 'doing archaeology digitally' as Costopolous remarks, constitutes a 'state of normality' in archaeological practice. This view places emphasis on the availability of digital tools and their use in institutional contexts, overlooking the highly structured nature of social groups that employ these tools, and where, how and why these technologies are created and used. While fruitful, these views tend to obscure broader developments in the social sciences and humanities, of which archaeology is a part, and underestimate the changing relationship between archaeology and society.

1.1.1 A distant view

Ethan Watrall has drawn the history of computational archaeology/digital archaeology all the way back to the pioneering work of James Deetz in the 1960s, who used computers at MIT to perform stylistic analyses of Arikara ceramics (Ethan Watrall 2017, Deetz (1965)). Most early interest in computation for archaeology was centred on the potential for computational databases, although ambition often out-stripped capability. By the 1970s, serious efforts were being put into work to build the infrastructural knowledge necessary to make and usefully query archaeological datasets. One can see this concern play out by considering a `topic model` (Shawn Graham 2014) of the early volumes of the Computer Applications in Archaeology (a topic model is a way of deducing latent patterns of discourse within text, based on patternings of words (See Graham, Weingart, and Milligan 2012)):

topic 1 – computer, program, may, storage, then, excavation, recording, all, into, form, using, retrieval, any, user, output, records, package, entry, one, unit

topic 6: but, they, one, time, their, all, some, only, will, there, would, what, very, our, other, any, most, them, even

topic 20: some, will, many, there, field, problems, may, but, archaeologists, excavation, their, they, recording, however, record, new, systems, most, should, need

The beginnings of the CAA are marked by hesitation and prognostication: what *are* computers for, in archaeology? There is a sense that for archaeologists, computation is something that will be useful insofar as it can be helpful for recording information in the field. By the 1980s desktop computing was becoming sufficiently widespread that the use of geographic information systems was feasible for more and more archaeologists. The other 'killer app' of the time was computer-aided design, which allowed metric 3d reconstructions from the plans drawn on site by excavators. Yet, computational resources were still limited enough that computing was not something that one could merely 'play' with. Software was costly, computation took time, and training resources were put into learning the proprietary packages that existed (rather than coding knowledge). By the 1990s, the introduction of the cd-rom and the shift in PC gaming technologies from primarily text-based to graphical based games led to teaching simulations for archaeology, most notably T. Douglas Price and Anne Birgitte Gebauer's *Adventures in Fugawiland*. Watrall identifies the emergence of the web as being not so much a boon for computational archaeology as it was for public archaeology (although the pioneering journal *Internet Archaeology* was first published in 1996); nevertheless, the birth of the web (which it must be remembered is *distinct from* and *overlays* the internet) allowed for a step-change in the effectiveness of the dissemination of open-source software and code, including practices for remote collaboration on code that are now beginning to percolate into scholarly publication.

The 2000s have seen, insofar as digital archaeology is concerned, a replay of the earlier episodes of computational archaeology, concommitant with each subsequent web 'revolution' (ie, so-called web 2.0, web 3.0 etc). Works such as (Evans, Daly, and MyiLibrary 2006) and (E. C. Kansa, Kansa, and Watrall 2011) are broadly concerned more with questions of infrastructure and training, while the more recent *Mobilizing the Past* deal with problems of training, and the ethical issues that the emerging digital surveillance permitted by our networked society presents to the practice of archaeology (and public archaeology). Perhaps the most promising new digital technologies to emerge in recent years include methods for linking open archaeological data via the web (ie, freeing various 'silos' of disciplinary knowledge so that the semantic connections between them can be followed and queried) and various mixed-reality approaches (virtual reality, augmented reality, 3d printing, and the so-called internet of things or the practice of wiring everything that can be wired to the web). The 2000s have also seen a growing realization that our digital tools and their algorithmic biases not only permit interesting questions to be asked about the past, but also inhibit points of view or impose their own worldviews upon the past in ways that may damage communities and/or scholarship. This reflective critique of computation in the service of archaeology marks digital archaeology within the ambit of the digital humanities (despite the division between anthropological and humanistic archaeologies).

1.1.2 Is digital archaeology part of the digital humanities?

In recent years - certainly the last decade - an idea called 'the digital humanities' has been percolating around the academy. It is a successor idea to 'humanities computing', but it captures that same distinction between discussed above. Digital archaeology has developed alongside the digital humanities, sometimes intersecting with it (notably, there was a major archaeological session at the annualinternational Alliance of Digital Humanities Organizations (ADHO) DH conference in 2013).

The various component organizations of the ADHO have been meeting in one form or another since the 1970s; so too the Computer Applications in Archaeology Conference has been publishing its proceedings since 1973. Archaeologists have been running simulations, doing spatial analysis, clustering, imaging, geophysicing, 3d modeling, neutron activation analyzing, x-tent modeling , etc, for what seems like ages. Happily, there is no one definition of 'dh' that everyone agrees on (see the various definitions collected at http://definingdh.org/; reload the page to get a new definition). For us, a defining *characteristic* of DH work is that public use we discussed above. But, another characteristic that we find useful to consider is the *purpose* to which computation is put in DH work. This means that digital work also has to be situated in the contexts of power and access and control (which sometimes means that digital work is mis-characterised as being part of a 'neo-liberal' agenda to reduce knowledge work to base profit motifs, eg Brouiellet; more thoughtful work about the confluence of the digital with neoliberalism may be found in Caraher xxxx and Kansa xxxx and Greenspan xxx. We discuss the ethical dimensions to digital work more fully in The Ethics of Big Data in Archaeology.)

For us, a key difference between the kind of computational archaeology of the last years of the twentieth century versus the emerging digital archaeology of the last decade lie in the idea of the purpose behind the computing power. Trevor Owens, a digital archivist, draws attention to the purpose behind one's use of computational power – generative discovery versus justification of an hypothesis (tjowens 2012). Discovery marks out the digital humanist whilst justification signals the humanist who uses computers. Discovery and justification are critically different concepts. For Owens, if we are using computational power to deform our texts, then we are trying to see things in a new light, to create new juxtapositions, to spark new insight. Stephen Ramsay talks about this too in Reading Machines (Ramsay 2011, 33), discussing the work of Samuels and McGann, (Samuels and McGann 1999): "Reading a poem backward is like viewing the face of a watch sideways – a way of unleashing the potentialities that altered perspectives may reveal". This kind of reading of data (especially, but not necessarily, through digital manipulation), does not happen very much at all in archaeology. If 'deformance' is a key sign of the digital humanities, then digital archaeologists are not digital humanists. Owen's point isn't to signal who's in or who's out, but rather to draw attention to the fact that:

> When we separate out the the context of discovery and exploration from the context of justification we end up clarifying the terms of our conversation. There is a huge difference between "here is an interesting way of thinking about this" and "This evidence supports this claim."

This is important in the wider conversation concerning how we evaluate digital scholarship. We've used computers in archaeology for decades to try to justify or otherwise connect our leaps of logic and faith, spanning the gap between our data and the stories we'd like to tell. We believe, on balance, that 'digital archaeology' sits along this spectrum between justification and discovery closer to the discovery end, that it sits within the digital humanities and should worry less about hypothesis testing, and concentrate more on discovery and generation, of 'interesting way[s] of thinking about this'.

Digital archaeology should be a prompt to make us 'think different'. Let's take a small example of how that might play out. It's also worth suggesting that 'play' as a strategy for doing digital work is a valid methodology (see Ramsay (2011)). (And of course, the ability to play with computing power is a function of Moore's law governing the increase in computing power time: computing is no longer a precious resource but something that can be 'wasted'.)

1.1.3 Archaeological Glitch Art

Bill Caraher is a leading thinker on the implications and practice of digital archaeology. In a post on archaeological glitch art (Caraher 2012) Caraher changed file extensions to fiddle about in the insides of images of archaeological maps. He then looked at them again as images:

> The idea ... is to combine computer code and human codes to transform our computer mediated image of archaeological reality in unpredictable ways. The process is remarkably similar to analyzing the site via the GIS where we take the "natural" landscape and transform it into a series of symbols, lines, and text. By manipulating the code that produces these images in both random and patterned ways, we manipulate the meaning of the image and the way in which these images communicate information to the viewer. We problematize the process and manifestation of mediating between the experienced landscape and its representation as archaeological data.

Similarly, Graham's work in representing archaeological data in sound (a literal auditory metaphor) translates movement over space (or through time) into a soundscape of tones (Graham 2017). This frees us from the tyranny of the screen and visual modes of knowing that often occlude more than they reveal (for instance, our Western-framed understanding of the top of the page or screen as 'north' means we privilege visual patterns in the vertical dimension over the horizontal (Montello et al. 2003)).

These playful approaches force us to rethink some of our norms of communication, our norms of what archaeology can concern itself with. It should be apparent that digital archaeology transcends mere 'digital skills' or 'tool use'; but it also suffers from being 'cool'.

1.1.4 The 'cool' factor

Alan Liu (Liu 2004) wondered what the role of the arts and humanities was in an age of knowledge work, of deliverables, of an historical event horizon that only goes back the last financial quarter. He examined the idea of 'knowledge work' and teased out how much of the driving force behind it is in pursuit of the 'cool'. Through a deft plumbing of the history of the early internet (and in particular, riffing on Netscape's 'what's cool?' page from 1996 and their inability to define it except to say that they'd know it when they saw it), Liu argues that cool is 'the aporia of information... cool is information designed to resist information... information fed back into its own signal to create a standing interference pattern, a paradox pattern' (Liu 2004, 179). The latest web design, the latest app, the latest R package for statistics, the latest acronym on Twitter where all the digital humanists play: cool, and dividing the world.

That is, Liu argued that 'cool' was amongst other things a politics of knowledge work, a practice and ethos. He wondered how we might 'challenge knowledge work to open a space, as yet culturally sterile (coopted, jejune, anarchistic, terroristic), for a more humane hack of contemporary knowledge?' (Liu 2004, 9). Liu goes on to discuss how the tensions of 'cool' in knowledge work (for us, read: digital archaeology) also intersects with an ethos of the unknown, that is, of knowledge workers who work nowhere else somehow manage to stand outside that system of knowledge production. (Is alt-ac 'alt' partially because it is the cool work?). This matters for us as archaeologists. There are many 'cool' things happening in digital archaeology that somehow do not penetrate into the mainstream (such as it is). The utilitarian dots-on-a-map were once cool, but are now pedestrian. The 'cool' things that could be, linger on the fringes. If they did not, they wouldn't be cool, one supposes. They resist.

To get that more humane hack that Liu seeks, Liu suggests that the historical depth that the humanities provides counters the shallowness of cool:

> The humanities thus have an explanation for the new arts of the information age, whose inheritance of a frantic sequence of artistic modernisms, postmodernisms, and post-postmodernists is otherwise only a displaced encounter with the raw process of historicity. Inversely, the arts offer the humanities serious ways of engaging – both practically and theoretically- with "cool". Together, the humanities and arts might be able to offer a persuasive argument for the humane arts in the age of knowledge work. (Liu 2004, 381).

In which case, the emergence of digital archaeologists and historians in the last decade might be the loci of the humane hacks – if we move into that space where we engage the arts. Indeed, the seminal anthropologist Tim Ingold makes this very argument with reference to his own arc as a scholar, 'From Science to Art and Back Again':

> Revisiting science and art: which is more ecological now? Why is art leading the way in promoting radical ecological awareness? The goals of today's science are modelling, prediction and control. Is that why we turn to art to rediscover the humility that science has lost?

We need to be making art. Digital archaeology naturally pushes in that direction.

1.1.5 Takeaways

- Digital archaeology is a public archaeology
- Digital archaeology is often about deformance rather than justification
- In that deformative practice, it is in some ways extremely aligned with artistic ways of knowing
- Digital archaeology is part of the digital humanities, and in many ways, presaged current debates and trends in that field.

All of these aspects of digital archaeology exist along a continuum. In the remainder of this chapter, we give you a 'boot-camp' to get you to the point where you can begin to wonder about deformation and the public entanglement with your work.

1.1.6 Exercises

The first steps in going digital are quite easy. They are fundamentally a question of maintaining some basic good habits. Everything else flows from these three habits:

```
1. separate _what_ your write/create from _how_ you write it.
2. keep what you write/create under version control.
3. break tasks down into their smallest manageable bits
```

Have you ever fought with Word or another wordprocessor, trying to get things just right? Word processing is a mess. It conflates writing with typesetting and layout. Sometimes, you just want to get the words out. Othertimes, you want to make your writing as accessible as possible... but your intended recipient can't open your file, because they don't use the same wordprocessor. Or perhaps you wrote up some great notes that you'd love to have in a slideshow; but you can't, because copying and pasting preserves a whole lot of extra gunk that messes up your materials. Similarly, while many archaeologists will use Microsoft Excel to manipulate tabular data (artifact measurements, geochemistry data, and so on), Excel is well known for both corrupting data and for being impossible to replicate (ie, the series of clicks to manipulate or perform an analysis differ depending on the individual's particular installation of Excel).

The answer is to separate your content from your tool, and your analytical processes separate from your data. This can help keep your thinking clear, but it also has a more nuts-and-bolts practical dimension. *A computer will always be able to read a text file.* That is to say: you've futureproofed your material. Any researcher will have old physical discs or disc drives or obsolete computers lying around. It is not uncommon for a colleague to remark, 'I wrote this in Wordperfect and I can't open this any more'. Graham's MA thesis is trapped on a 3.5" disc drive that was compressed using a now-obsolete algorithm and it cannot be recovered. If, on the other hand, he had written the text as a .txt file, and saved the data as .csv tables, those materials would *continue* to be accessible. If the way you have manipulated or cleaned the data is written out as a `script`, then a subsequent investigator (or even your future self) can re-run the exact sequence of analysis, or re-write the script into the equivalent steps in another analytical language.

A .txt file is simply a text file; a .csv is a text file that uses commas to separate the text into columns. Similarly, a .md file is a text file that uses things like `#` to indicate headers, and `_` to show where italicized text starts and stops. A script, in a play, tells you what to say and do. A `script` for a language like `R` or `Python`

does the same thing for the computer, and has the advantage that it is human-readable and annotatable as well, because its format *is still a simple text file*. Scripts you might encounter could have the `.r` or `.py` or `.sh` file extensions. You can open these in a text editor and see what the computer is being instructed to do. Annotations or comments in the script can be set off in various ways, and help the researcher know what is happening or is intended to happen at various points. Let's begin by creating some simple text files to document our research process, in the Markdown format.

1. A nice place to practice writing in markdown that shows you immediately how your text might be rendered when turned into html, pdf, or Word doc is Dillinger.io. Go there now to try it out. Write a short piece on why you're interested in Digital Archaeology.

a. Include a blockquote from the introduction to this book.
b. Include two links to an external site.
c. Embed an image.

Here are two short demonstration videos:

https://youtu.be/Gxb88ujao-U

https://youtu.be/ip7HFi8zozY

Click the 'export as' dropdown, and select 'markdown'. Keep a note of where the file has downloaded to on your machine.

2. Sign up for a github account

Once you're logged in, we will create a new repository called `scratchpad`. Click on the + at the top right of the screen, beside your avatar image.

Write a short description in the 'description box', and tick off the 'initialize the repository with a readme'. You can also select a license from the drop down box — this will put some standard wording on your repository page about the conditions under which someone else might use (or cite) your code.

Click 'Create repository'.

At this point, you now have a folder — a repository — on the GitHub website into which you can deposit your files. It will be at http://github.com//scratchpad. So let's put some materials into that repository.

Notice, when you're on your repository's page, that there is a button to 'create new file' and another for 'upload files'. Click on 'create new file'.

3. The file editor window will open. In the new file name box, type in `todo-list.md`. The `.md` is important, because github recognizes this as a text file using markdown conventions. It displays markdown translated into the appropriate html - `#` becomes `<h1>` which will be bolded and larger text in our browser. In the editor window, wse bullet points to break down what else you need to do this week. Each bullet point should have a sub-bullet with an actual `ACTION` listed, something that you can accomplish to get things done.

4. Click on the green 'commit' button at the bottom of the page when you're done.

Now we'll upload a file into your repository.

5. Find the markdown file on your machine that you created with Dillinger. You can drag-and-drop this onto the list of files in your repository; Github will know that you want to upload it. **OR** you can click on the 'upload files' button, and then 'select files' and click on the file that way (much like adding an attachment to an email).

6. Github will upload the file; you can upload more files at this point, or click on the green 'commit' button at the bottom.

As you work through this book, we encourage you to write your thoughts, observations, or results in simple text files. This is good practice whether or not you embark on a full-blown digital project, because ultimately, if you use a computer in your research, you *have* gone digital.

1.2 Project Management Basics

A digital project, whether in archaeology or in other fields, iterates through the same basic steps. There is

```
1. finding data
2. fixing data
3. analyzing the data
4. communicating the story in the data
```

Eighty percent of your time on any digital project will be invested in cleaning up the data and documenting what you've done to it. But in truth, a digital project begins long before we ever look at a data set (or are given data to work with, if we're part of a larger project). How do we formulate a research question or our exploration more generally? How do we translate a gut feeling or intution or curiosity into something that is *operable*? REF Moretti on operationalizing things

The four steps we identified above are cyclical; at any one time you might be at a different stage of the process. Indeed, those four steps could easily be subsumed under what Simon Appleford and Jennifer Guiliano of devdh.org identify as the 'Best Practice Principles Of Designing Your First Project.' For Appleford and Guiliano, the outline of a project involves figuring out:

1. the question, problem, or provocation
2. sources (primary, secondary)
3. analytical activity
4. audience
5. product

Note that 4, audience, comes before 5, product. You must think of your reader/user!

Let us imagine that we were inspired by Allison Mickel's piece, 'Tracing Teams, Texts, and Topics: Applying Social Network Analysis to Understand Archaeological Knowledge Production at Çatalhöyük' (Mickel 2016).

We could frame a question: 'What role do social networks play in the development of knowledge production at my site?'

We could frame a problem: 'Mickel's exploration of social networks considered x, but not y.'

We could frame a provocation: 'Social Network Analysis promises to revolutionize our knowledge of the social contexts that underpin archaeological fieldwork, putting this power in the hands of everyone from the site director on down.'

Following Appleford and Guiliano, we can refine our question, or our problem, or our provocation down to its essence in order to figure out the next parts of the the process. Knowing exactly what kind of question, problem, or provocation we're after, we then have a better sense of what to do when confronted with a mass of data (for instance, the excavation diaries from Kenen Tepe held in OpenContext.org, deposited by Parker and Cobb, 2012). Once the question is well-drawn out, questions 3 and 4 take care of themselves.

One other element that we might add is 'collaboration'. How do you plan to collaborate? While many digital archaeology projects are done by a single individual working in the quiet of their own space, most projects require many different skill sets, perspectives, and stakeholders. It is worth figuring out at the outset how you plan to work together. Will you use email? Will you use a private slack or messaging client? What about Kanban boards? (A Kanban board can be as simple as a whiteboard with three columns on it, marked 'to do', 'doing', and 'done'. Tasks are written on post-it notes and moved through the columns as necessary. A popular software implementation of a Kanban board is Trello.) We would also recommend that you write down the ideal division of labour and areas of responsibility for each of the participants, *along with a mechanism for resolving disputes.*

Finally, how much time would you have to work on your digital archaeology project? All of us have multiple demands on our time. Let's be realistic about how much time you have available. How many hours, total, do you spend in class, at work, asleep, and socializing? Add that up for a week, then multiply by the number of

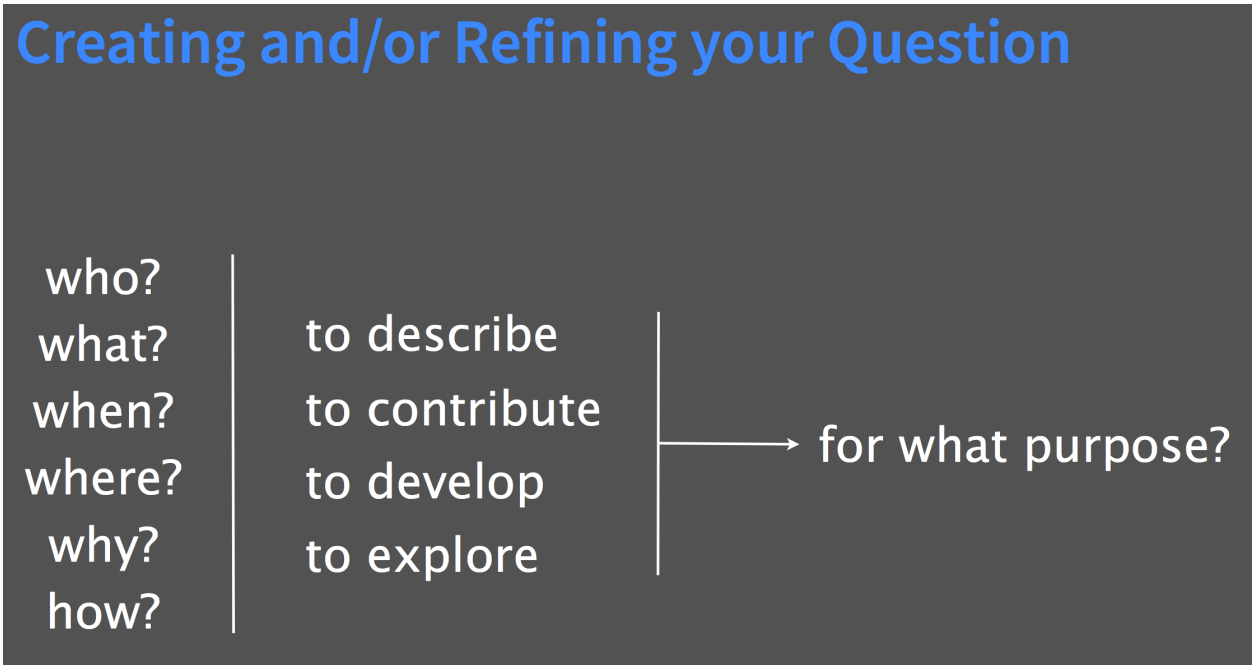


Figure 1.2: Creating and/or refining your research question, per DevDH.org

weeks in your term. There are 384 hours in a 16 week term. Subtract to find out how many 'spare' hours you can devote to homework, this project, or a hobbby.

Divide that by the number of weeks your course runs. That's how many hours per week you can spend on all your non in-class course work. Then, divide those hours by the number of courses you have.

That's how much time you have for your project. It's not an awful lot, which means that the more energy you put into planning, the more effective your labour is going to be.

1.2.1 Take-aways

- be explicit about how collaboration will be managed
- be explicit about how your research goals intersect with your audience
- be brutally honest about your time and guard it jealously

1.2.2 exercises

1. Create a new markdown file in your `scratchpad` repository. Call it 'initial-project-idea.md'. Using `#` to indicate headings, sketch out a question, problem, or provocation of your own that occurs to you as you browse the Kenen Tepe materials housed at OpenContext.org. Save that file.
2. Read this piece by Ben Marwick What kind of project management plan did he have in terms of the digital work? Reflect on his case study and identify where the trouble points were in the light of what you've read in this section. Can you find a project management plan for *any* archaeological project generating digital data?

1.3 Github & Version Control

It's a familiar situation - you've been working on a paper. It's where you want it to be, and you're certain you're done. You save it as 'final.doc'. Then, you ask your friend to take a look at it. She spots several typos and that you flubbed an entire paragraph. You open it up, make the changes, and save as 'final-w-changes.doc'. Later that day it occurs to you that you don't like those changes, and you go back to the original 'final.doc', make some changes, and just overwrite the previous version. Soon, you have a folder like:

```
|-project
    |-'finalfinal.doc'
    |-'final-w-changes.doc'
    |-'final-w-changes2.doc'
    |-'isthisone-changes.doc'
    |-'this.doc'
```

Things can get messy quite quickly. Imagine that you also have several spreadsheets in there as well, images, snippets of code... we don't want this. What we want is a way of managing the evolution of your files. We do this with a program called Git. Git is not a user-friendly piece of software, and it takes some work to get your head around. Git is also very powerful, but fortunately, the basic uses to which most of us put it to are more or less straightforward. There are many other programs that make use of Git for version control; these programs weld a graphical user interface on top of the main Git program. It is better however to become familiar with the basic uses of git from the command line *first* before learning the idiosyncracies of these helper programs. The exercises in this section will take you through the basics of using Git from the command line.

1.3.1 The core functions of Git

Figure 1.3: Alice Bartlett's summary of what Git does

At its heart, Git is a way of taking 'snapshots' of the current state of a folder, and saving those snapshots in sequence. (For an excellent brief presentation on Git, see Alice Bartlett's presentation here; Bartlett is a senior developer for the Financial Times). In Git's lingo, a folder on your computer is known as a `repository`. This sequence of snapshots in total lets you see how your project unfolded over time. Each time you wish to

take a snapshot, you make a `commit`. A commit is a Git command to take a snapshot of the entire repository. Thus, your folder we discussed above, with its proliferation of documents becomes:

```
|-project
    |-'final.doc'
```

BUT its commit history could be visualized like this:

Figure 1.4: A visualization of the history of commits

Each one of those circles represents a point in time when you the writer made a commit; Git compared the state of the file to the earlier state, and saved a snapshot of the `differences`. What is particularly useful about making a commit is that Git requires two more pieces of information about the git: who is making it, and when. The final useful bit about a commit is that you can save a detailed message about *why* the commit is being made. In our hypothetical situation, your first commit message might look like this:

```
Fixed conclusion

Julie pointed out that I had missed
the critical bit in the assignment
regarding stratigraphy. This was
added in the concluding section.
```

This information is stored in the history of the commits. In this way, you can see exactly how the project evolved and why. Each one of these commits has what is called a `hash`. This is a unique fingerprint that you can use to 'time travel' (in Bartlett's felicitous phrasing). If you want to see what your project looked like a few months ago, you `checkout` that commit. This has the effect of 'rewinding' the project. Once you've checked out a commit, don't be alarmed when you look at the folder: your folder (your repository) looks like how it once did all those weeks ago! Any files written after that commit seem as if they've disappeared. Don't worry: they still exist!

What would happen if you wanted to experiment or take your project in a new direction from that point forward? Git lets you do this. What you will do is create a new `branch` of your project from that point. You can think of a branch as like the branch of a tree, or perhaps better, a branch of a river that eventually merges back to the source. (Another way of thinking about branches is that it is a label that sticks with these particular commits.) It is generally considered 'best practice' to leave your `master` branch alone, in the sense that it represents the best version of your project. When you want to experiment or do something new, you create a `branch` and work there. If the work on the branch ultimately proves fruitless, you can discard it. *But*, if you decide that you like how it's going, you can `merge` that branch back into your master. A merge is a commit that folds all of the commits from the branch with the commits from the master.

Git is also a powerful tool for backing up your work. You can work quite happily with Git on your own machine, but when you store those files and the history of commits somewhere remote, you open up the possibility of collaboration *and* a safe place where your materials can be recalled if -perish the thought- something happened to your computer. In Git-speak, the remote location is, well, the `remote`. There are many different places on the web that can function as a remote for Git repositories. You can even set one up on your own server, if you want. One of the most popular (and the one that we use for ODATE) is Github. There are many useful repositories shared via Github of interest to archaeologists - OpenContext for instance shares a lot of material that way. To get material *out* of Github and onto your own computer, you `clone` it. If that hypothetical paper you were writing was part of a group project, your partners could clone it from your Github space, and work on it as well!

You and Anna are working together on the project. You have made a new project repository in your Github space, and you have cloned it to your computer. Anna has cloned it to hers. Let's assume that you have a

very productive weekend and you make some real headway on the project. You `commit` your changes, and then `push` them from your computer to the Github version of your repository. That repository is now one commit *ahead* of Anna's version. Anna `pulls` those changes from Github to her own version of the repository, which now looks *exactly* like your version. What happens if you make changes to the exact same part of the exact same file? This is called a `conflict`. Git will make a version of the file that contains text clearly marking off the part of the file where the conflict occurs, with the conflicting information marked out as well. The way to `resolve` the conflict is to open the file (typically with a text editor) and to delete the added Git text, making a decision on which information is the correct information.

1.3.2 Key Terms

- repository: a single folder that holds all of the files and subfolders of your project
- commit: this means, 'take a snapshot of the current state of my repostiory'
- publish: take my folder on my computer, and copy it and its contents to the web as a repository at github.com/myusername/repositoryname
- sync: update the web repository with the latest commit from my local folder
- branch: make a copy of my repository with a 'working name'
- merge: fold the changes I have made on a branch into another branch
- fork: to make a copy of someone else's repo
- clone: to copy an online repo onto your own computer
- pull request: to ask the original maker of a repo to 'pull' your changes into their master, original, repository
- push: to move your changes from your computer to the online repo
- conflict: when two commits describe different changes to the same part of a file

1.3.3 Take-aways

- Git keeps track of all of the differences in your files, when you take a 'snapshot' of the state of your folder (repository) with the `commit` command
- Git allows you to roll back changes
- Git allows you to experiment by making changes that can be deleted or incorporated as desired
- Git allows you to manage collaboration safely
- Git allows you to distribute your materials

1.3.4 Further Reading

We alluded above to the presence of 'helper' programs that are designed to make it easier to use Git to its full potential. An excellent introduction to Github's desktop GUI is at this Programming Historian lesson on Github. A follow-up lesson explains the way Github itself can be used to host entire websites! You may explore it here. In the section of this chapter on open notebooks, we will also use Git and Github to create a simple open notebook for your research projects.

You might also wish to dip into the archived live stream; link here from the first day of the NEH funded Institute on Digital Archaeology Method and Practice (2015) where Prof. Ethan Watrall discusses project management fundamentals and, towards the last part of the stream, introduces Git.

1.3.5 Exercises

Take a copy of the ODATE Binder. Carefully peruse the readme so that you can create your own version of the *executable* version. Once you've done that, launch the binder. It might take five or six minutes to launch; be patient.

Once it has launched, click the `new` dropdown and select terminal.

1. Because the Jupyter notebook is being served from a github repository, the `git` program is already running in the background and is keeping track of changes. If you were working on your own machine (and had git installed), you could turn *any* folder into a repository by telling Git to start watching the folder, by typing `git init` at the command prompt inside that folder.

Periodically, we tell Git to take a snapshot of the state of the files in the folder by using the command 'git commit'. This sequence of changes to your repo are stored in a *hidden* directory, `.git`. Most of the time, you will never have reason to search that folder out. (But know that the config file that describes your repo is in that folder. There might come a time in the future where you want to alter some of the default behaviour of the git program. You do that by opening the config file, which you can read with a text editor. Google 'show hidden files and folders' for your operating system when that time comes.)

2. Let's make a new file; we can do this by selecting the kind of file we want from the `new` dropdown menu. Select 'text file'. Jupyter will open its text editor and create a new file for you called `untitled.txt`. Click on the name to change it to `exercise1.md`. Type in the editor to add some information in it - perhaps a note about what you're trying to do- then hit save. **Do not hit logout**.

a. From the Jupyter home screen, hit `new` and select terminal. At the prompt type `$ git status`
b. Git will respond with a couple of pieces of information. It will tell you which `branch` you are on. It will list any untracked files present or new changes that are unstaged. We now will `stage` those changes to be added to our commit history by typing `$ git add -A`. (the bit that says `-A` adds any new, modified, or deleted files to your commit when you make it. There are other options or flags where you add *only* the new and modified files, *or* only the modified and deleted files.)
c. Let's check our git status again: type `$ git status`
d. You should see something like this:

```
On branch master
Initial commit
Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   exercise1.md```
```

e. Let's take a snapshot: type `$ git commit -m "My first commit"`. Nothing much seems to have happened; a new `$` is displayed. In this kind of environment, no news is good news. It's only often when something goes wrong that you'll see the terminal print out information. For some users (especially if you are approaching these exercises via DHBox rather than our Binder environment) there could be an error at this point. Git keeps track not only of the changes, but *who* is making them. Git might ask you for your name and email. Helpfully, the Git error message tells you exactly what to do: type `$ git config --global user.email "you\@example.com"` and then type `$ git config --global user.name "Your Name"`. Now try making your first commit.

Congratulations, you are now able to track your changes, and keep your materials under version control!

3. Go ahead and make some more changes to your repository. Add some new files. Commit your changes after each new file is created. Now we're going to view the history of your commits. Type `$ git log`. What do you notice about this list of changes? Look at the time stamps. You'll see that the entries are listed in reverse chronological order. Each entry has its own 'hash' or unique ID, the person who made the commit and time are listed, as well as the commit message eg:

```
commit 253506bc23070753c123accbe7c495af0e8b5a43
Author: Shawn Graham <shawn.graham@carleton.ca>
Date:   Tue Feb 14 18:42:31 2017 +0000

Fixed the headings that were broken in the about section of readme.md
```

a. We're going to go back in time and create a new branch. You can escape the `git log` by typing `q`. Here's how the command will look: `$ git checkout -b branchname <commit>` where `branch` is the

name you want the branch to be called, and `<commit>` is that unique ID. Make a new branch from your second last commit (don't use < or >).

b. We typed `git checkout -b experiment 253506bc23070753c123accbe7c495af0e8b5a43` (**Don't** copy *our* command, because our version includes a reference to a commit that doesn't exist for you! Select a commit reference from your own commit history, which you can inspect with `git log`). The response: `Switched to a new branch 'experiment'` Check git status and then list the contents of your repository. What do you see? You should notice that some of the files you had created before seem to have disappeared - congratulations, you've time travelled! Those files are not missing; but they *are* on a different branch (the master branch) and you can't harm them now. Add a number of new files, making commits after each one. Check your git status, and check your git log as you go to make sure you're getting everything. Make sure there are no unstaged changes - everything's been committed.

4. Now let's assume that your `experiment` branch was successful - everything you did there you were happy with and you want to integrate all of those changes back into your `master` branch. We're going to merge things. To merge, we have to go back to the master branch: `$ git checkout master`. (Good practice is to keep separate branches for all major experiments or directions you go. In case you lose track of the names of the branches you've created, this command: `git branch -va` will list them for you.)

a. Now, we merge with `$ git merge experiment`. Remember, a merge is a special kind of commit that rolls all previous commits from both branches into one - Git will open your text editor and prompt you to add a message (it will have a default message already there if you want it). Save and exit and ta da! Your changes have been merged together.

5. One of the most powerful aspects of using Git is the possibility of using it to manage collaborations. To do this, we have to make a copy of your repository available to others as a `remote`. There are a variety of places on the web where this can be done; one of the most popular at the moment is Github. Github allows a user to have an unlimited number of `public` repositories. Public repositories can be viewed and copied by anyone. `Private` repositories require a paid account, and access is controlled. If you are working on sensitive materials that can only be shared amongst the collaborators on a project, you should invest in an upgraded account (note that you can also control which files get included in commit; see this help file. In essence, you simply list the file names you do not want committed; here's an example). Let's assume that your materials are not sensitive.

a. Now we push your work in the repository onto the version that lives at Github.com:

```
git push -u origin master
```

NB If you wanted to push a `branch` to your repository on the web instead, do you see how you would do that? If your branch was called `experiment`, the command would look like this:

```
$ git push origin experiment
```

b. Because your repository is on the web at Github, it is possible that you might make changes to the repository directly there, or from your local machine. To get the updates to integrate into where you are currently working, you could type

```
$ git pull origin master
```

and then begin working.

1.3.6 Warnings

This only scratches the surface of what Git and Github can do. (Remember - git is the program that keeps snapshots of your work and enables version control; Github is a place that lets you share that sequence of snapshots and files so that others can contribute changes). More information about Git and some exercises conceived for the DHBox/Linux/Mac are available here. Remember, although it is possible to make changes to files directly via the edit button on Github, you should be careful if you do this, because things rapidly

can become out of sync, resulting in conflicts between differing versions of the same file. Get in the habit of making your changes on your own machine, and making sure things are committed and up-to-date (`git status`, `git pull origin master`, `git fetch upstream` are your friends) before beginning work. At this point, you might want to investigate some of the graphical interfaces for Git (such as Github Desktop). Knowing as you do how things work from the command line, the idiosyncracies of the graphical interfaces will make more sense. For further practice on the ins-and-outs of Git and Github Desktop, we recommend trying the Git-it app by Jessica Lord.

For help in resolving merge conflicts, see the Github help documentation. For a quick reminder of how the workflow should go, see this cheat-sheet by Chase Pettit.

1.4 Open Notebook Research & Scholarly Communication

SG HAVE THE STUDENTS USE RSTUDIO FROM THE BINDER AND THEN PASTE LINCOLN'S FILE INTO THAT

Digital archaeology necessarily generates a lot of files. Many of those files are data; many more are manipulations of that data, or the data in various stages of cleaning and analysis. Without any sort of version control or revision history (as detailed in the previous section), these files quickly replicate to the point where a project can be in serious danger of failing. Which file contains the 'correct' data? The correct analysis? Even worse, imagine coming back to a project after a few months' absence. Worse still, after a major operating system update of the kind foisted on Windows users from Windows 7 to Windows 10. The bad news continues: magnetic storage can fail; online cloud services can be hacked; a key person on the project can die.

Even if the data makes it to publication, there is the problem of the data not being available to others for re-interrogation or re-analysis. Requests for data from the authors of journal articles are routinely ignored, whether by accident or design. Researchers may sit on data for years. We have all of us had the experience of working on a collection of material, and then writing to the author of the original article, requesting an explanation for some aspect of the data schema used, only to find out that the author has either died, kept no notes, left the field entirely, or simply doesn't remember.

There is no excuse for this any longer. `Open notebook science` is a gathering movement across a number of fields to make the entire research process transparent by sharing materials online as they are generated. These include everything from the data files themselves, to the code used to manipulated it, to notes and observations in various archives. Variations on this 'strong' position include data-publishing of the materials after the main paper has been published (see for instance OpenContext or the Journal of Open Archadological Data). Researchers such as Ben Marwick and Mark E. Madsen are leading the field in archaeology, while scholars such as Caleb McDaniel are pushing the boundaries in history. The combination of simple text files (whether written text or tabular data such as .csv files) with static website generators (ie, html rather than dynamically generated database websites like Wordpress) enables the live publishing of in-progress work. Carl Boettiger is often cited as one of the godfathers of this movement. He makes an important distinction:

> This [notebook, not blog] is the active, permanent record of my scientific research, standing in place of the traditional paper bound lab notebook. The notebook is primarily a tool for me to do science, not communicate it. I write my entries with the hope that they are intelligible to my future self; and maybe my collaborators and experts in my field. Only the occasional entry will be written for a more general audience. [...] In these pages you will find not only thoughts and ideas, but references to the literature I read, the codes or manuscripts I write, derivations I scribble and graphs I create and mistakes I make. (Boettiger)

Major funding bodies are starting to require a similar transparency in the research that they support. Recently, the Social Sciences and Humanities Research Council of Canada published guidance on data management plans:

> All research data collected with the use of SSHRC funds must be preserved and made available for use by others within a reasonable period of time. SSHRC considers "a reasonable period" to be within two years of the completion of the research project for which the data was collected.

Annecdotally, we have also heard of work being denied funding because the data management plan, and/or the plan for knowledge mobilization, made only the briefest of nods towards these issues: 'we shall have a blog and will save the data onto a usb stick' does not cut it any more. A recent volume of case-studies in 'reproducible research' includes a contribution from Ben Marwick that details not only the benefits of such an approach, but also the 'pain points'. Key amongst them was that not everyone participating in the project was on board using scripted code to perform the analysis (preferring instead to use the point-and-click of Excel), the duplication of effort that emerged as a result, and the complexities that arose from what he calls the 'dual universes' of Microsoft tools versus the open source tools. (MARWICK REF). On the other hand, the advantages outweighed the pain. For Marwick's team, because their results and analysis can be re-queried and re-interrogated, they have an unusually high degree of confidence in what they've produced. Their data, and their results have a complete history of revisions that can be examined by reviewers. Their code can be re-used and re-purposed, thus making their subsequent research more efficient. Marwick goes on to create an entire 'compendium' of code, notes, data, and software dependencies that can be duplicated by other researchers. Indeed, we will be re-visiting their compendium in Section XXXXXXXXX.

Ultimately, McDaniels says it best about keeping open notebooks of research in progress when he writes,

> The truth is that we often don't realize the value of what we have until someone else sees it. By inviting others to see our work in progress, we also open new avenues of interpretation, uncover new linkages between things we would otherwise have persisted in seeing as unconnected, and create new opportunities for collaboration with fellow travelers. These things might still happen through the sharing of our notebooks after publication, but imagine how our publications might be enriched and improved if we lifted our gems to the sunlight before we decided which ones to set and which ones to discard? What new flashes in the pan might we find if we sifted through our sources in the company of others?

A parallel development is the growing practice of placing materials online as pre-prints or even as drafts, for sharing and for soliciting comments. Graham for instance uses a blog as a place to share longer-form discursive writing in progress; with his collaborators Ian Milligan and Scott Weingart, he even wrote a book 'live' on the web, warts and all (which you may still view at The Macroscope). Sharing the draft in progress allowed them to identify errors and ommissions as they wrote, and for their individual chapters and sections to be incorporated into class syllabi right away. In their particular case, they came to an arrangment with their publisher to permit the draft to remain online even after the formal publication of the 'finished' book - which was fortunate, as they ended up writing another chapter immediately after publication! In this, they were building on the work of scholars such as Kathleen Fitzpatrick, whose *Planned Obsolescence* was one of the first to use the Media Commons 'comment press' website to support the writing. Commentpress is a plugin for the widely used Wordpress blogging system, which allows comments to be made at the level of individual paragraphs. This textbook you are currently reading uses another solution, the hypothes.is plugin that fosters communal reading and annotation of electronic texts. This points to another happy by-product of sharing one's work this way - the ability to generate communities of interest around one's research. The Kitz et al. volume is written with the Gitbook platform, which is a graphical interface for writing using Git at its core with markdown text files to manage the collaboration. The commit history for the book then also is a record of how the book evolved, and who did what to it when. In a way, it functions a bit like 'track changes' in Word, with the significant difference that the evolution of the book can be rewound and taken down different branches when desired.

In an ideal world, we would recommend that everyone should push for such radical transparency in their research and teaching. But what is safe for a group of (mostly) white, tenured, men is not safe for everyone online. In which case, what we recommend is for individuals to assess what is safest for them to do, while still making use of the affordances of Git, remote repositories, and simple text files. Bitbucket at the time of writing offers free private repositories (so you can push your changes to a remote repository without fear of others looking or cloning your materials); ReclaimHosting supports academic webhosting and allows one to

set up the private 'dropbox' like file-sharing service Owncloud.

In this exercises below, we will explore how to make a simple open notebook via a combination of markdown files and a repository on Github. Ultimately, we endorse the model developed by Ben Marwick, of creating an entire 'research compendium' that can be installed on another researcher's machine, but a good place to start are with the historian Lincoln Mullen's simple notebook templates. This will introduce to you another tool in the digital archaeologist's toolkit, the open source R programming language and the R Studio 'IDE' ('integrated development environment).

Far more complicated notebooks are possible, inasmuch as they combine more features and ways of compiling your research. Scholars such as Mark Madsen use a combination of Github pages and the Jekyll blog generator (for more on using Jekyll to create static websites, see Amanda Visconti's Programming Historian tutorial.) A simple Github repository and Wordpress blog can be used in tandem, where the blog serves for the *narrative* part of a notebook, the part that tries to make sense of the notes contained in the repository. This aspect of open notebook science is critically important in that it serves to signal your *bona fides* as a serious scholarly person. Research made available online is *findable*; given the way web search works, if something cannot be found easily, it might as well not exist.

Ultimately, you will need to work out what combination of tools works best for you. Some of our students have had success using Scrivener as a way of keeping notes, where Scrivener writes to a repository folder or some other folder synced across the web (like Dropbox, for instance). In this workflow, you have one Scrivener file per project. Scrivener uses the visual conceit of actual 3 x 5 notecards. Within Scrivener, one would make one card per note, and keep them in a 'research' folder. Then, when it becomes time to write up the project, those notecards can be moved into the draft and rearranged as necessary so that the writing flows naturally from them.

1.4.1 How to Ask Questions

"I tried it and it didn't work", read the email. Tried what? What didn't work? What did the code actually say? What did you actually type? There is an art to asking questions when code or computing is involved. A lot of the issues involved in trying to get help ultimately stem from our natural reluctance, our natural reticence, to appear foolish or ignorant. Admitting in a public forum, or to a classmate or peer, or professor or colleague that you don't know how to x is intimidating. In which case, we invite you to reflect on this comic by Randall Munroe:

Learning 'how to ask questions' involves also learning 'how to answer questions'. This in turn is related to what Kathleen Fitzpatrick calls 'Generous Thinking':

> Generous Thinking [begins] by proposing that rooting the humanities in generosity, and in particular in the practices of thinking with rather than reflexively against both the people and the materials with which we work, might invite more productive relationships and conversations not just among scholars but between scholars and the surrounding community - Kathleen Fitzpatrick

This means that when your peer or colleague has a question or issue in their code or their process (or in their journey to become more digitally inflected in their work) you egage with them *at face value*, in the same spirit that the stickpeople in Munroe's cartoon do. Nothing will be more detrimental to the progression of digital archaeology than the appearance of gatekeepeers and hidden knowledge.

When you run into a problem - when you first become aware that something is going awry - stop. Collect your breath. Do not click madly about in all directions, hoping that something might work.

1. Step away from your computer. Shut it down, put it to sleep, go outside. Sometimes, what you need more than anything else is just fresh eyeballs. Come back to the machine after a 30 minute break. You will be surprised at how often all that you really needed was a break.
2. If that doesn't solve the issue, your next step is to help other people **reproduce** what's happening on your machine. There are a variety of ways of doing this. To start, open your text editor and make a new 'date-my-problem.md' file with the following headings:

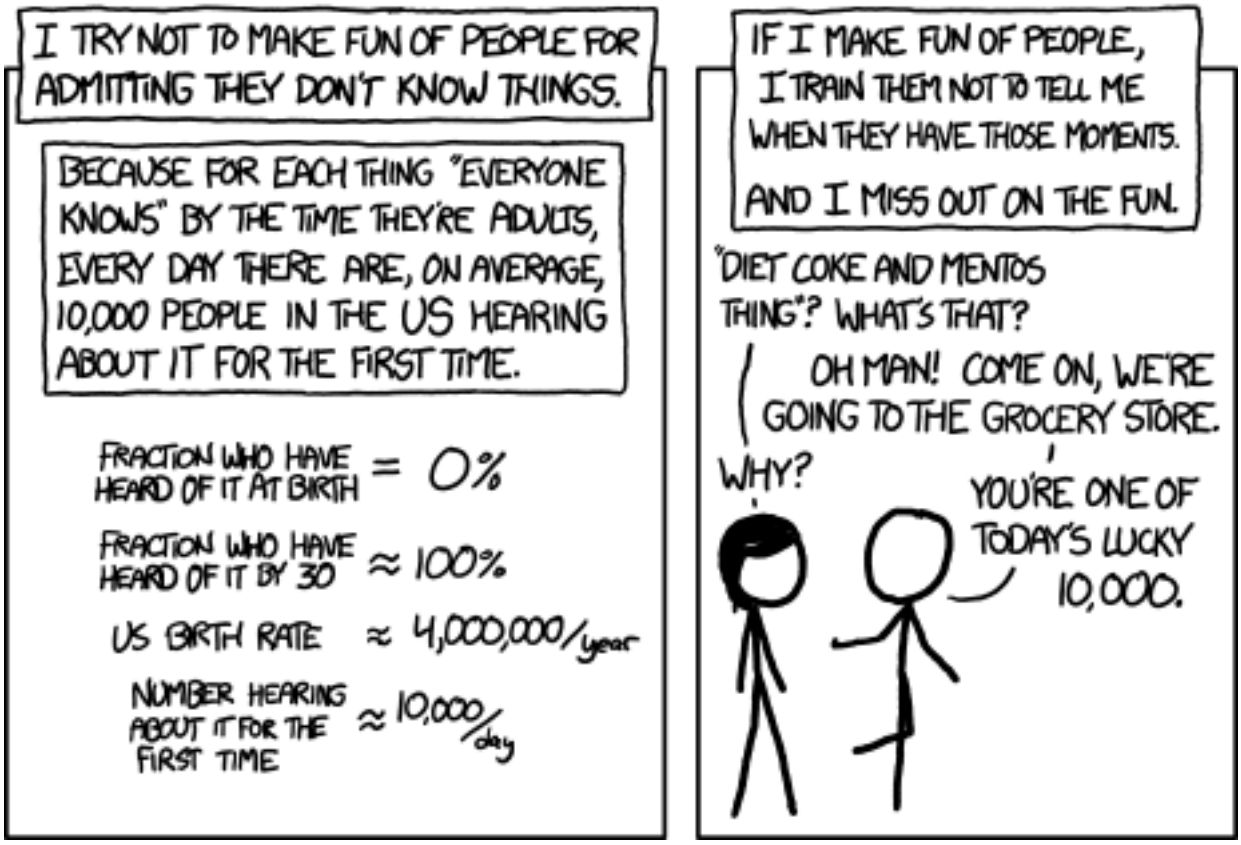


Figure 1.5: XKCD comic 'Ten Thousand'

```
# what I wanted to do
  - include links to any websites or articles you were reading
    # what I did
  - include the history of commands you've typed by passing your command line history to a new text fil
    # expected outcome
  - describe what you think ought to be happening
  - describe or list the software, the libraries, the packages you're working with
  - share the actual code you're using
    # actual outcome
  - what actually seems to be happening.
  - copy all error messages
  - take screenshots and put them online somewhere; include here the URL to the screenshot
  - sometimes even a video can be handy; screen-cast-o-matic.com lets you take a video with narration o
```

With time, you will become better at describing the issue succinctly. You will also learn that many of your issues have been encountered before - one good strategy is to Google the actual text of your error message. You will discover Stack Overflow, who also have excellent advice on asking for help. There is even code to help you create code that shows other people what you're experiencing!

Your help query can also go into your open notebook. In this way, your open notebook becomes not just the record of your research, but also an invitation to others to join you.

1.4.2 discussion

Questions for discussion:

1. Search the archaeological literature (via jstor or Google Scholar) for examples of open notebook science 'in the wild'. Are you finding anything, and if so, where? Do there seem to be impediments *from the journals* regarding this practice?
2. What excites you about the possibilities of open notebook archaeology? What are the advantages?
3. What frightens you? What are the disadvantages?
4. Search online for the 'replicability crisis in science'. Is there any such thing in archaeology?
5. Study Marwick's paper REF and compare it to its supporting Github repository. What *new* questions could be asked of this data?
6. In what ways are terms like 'open access', 'open source', and 'open science' synonyms for a similar approach, and in what ways are they different?

1.4.3 Take-aways

Keeping an open notebook (or if necessary, a closed notebook; see below) is a habit that must be cultivated. As a target to aim for, try to have

- each experiment|project in its own folder
- each experiment|project with regular pattern of subfolders `data` and `figures` and `text` and `bib` etc
- the experiments|projects under version control.
- a plan for data publishing. One option is to submit the repository to zenodo or similar to obtain digital object identifiers (DOIs) for the repository
- a plan to write as you go, on a fail log or blog or what-have-you. Obtain a DOI for this, too.

We haven't mentioned DOIs in this section, but when your notebook and your narrative about your research has a DOI, it becomes easier for your colleagues to cite your work - even this work in progress!

1.4.4 Further Reading

Baker, James. 'Preserving Your Research Data', *The Programming Historian* http://programminghistorian.org/lessons/preserving-your-research-data

1.4.5 On Privilege and Open Notebooks

While we argue for open notebooks, there may be circumstances where this is not desireable or safe to do. Readers may also want to explore an Evernote alternative, Laverna which stores your notes in your web-browser's cache hence making them private, but also allows sync to services such as Dropbox (versioning and backup are still absolutely critical). If you work primarily on a Mac computer, nvAlt by Brett Terpstra is an excellent note-taking application that can sync remotely. Another possibility is Classeur a web abb that integrates with various blogging platforms, allows for syncing and collaboration, the choice of what to make public and what to keep private, and includes the ability to sort notes into various notebooks. It does *not* save locally, so be warned that your information is on their servers. There is an API (application programming interface) that allows you to download your materials (for more on APIs, see Introduction to Digital Libraries, Archives & Repositories).

A final word on the privilege involved in keeping an open notebook is warranted. To make one's research available openly on the web, to discuss openly the things that worked, the things that haven't, the experiments tried and the dead ends explored, is at the current moment something that depends on the perceived race, class, and gender of the person doing it. What passes without comment when I (Shawn Graham, a white, tenured, professor) do something, could attract unwarranted, unwanted, and unfair attention if a woman of colour undergraduate tried. This is not to say this always happens; but disgracefully it happens far too often. It is important and necessary to fight back against the so-called 'internet culture' in these things, but it is not worth risking one's safety. To those who benefit from privilege, it is incumbent upon them to make things safe for others, to recognize that open science, open humanities, represents a net boon to our field. In which case, it is up to them to normalize such practices, to make it safe to try things out. We discuss more in the following section on what Failing Productively means, why it matters, and why it is integral not only to digital archaeology, but the culture of academic research, teaching, and outreach more generally.

1.4.6 exercises

In this series of exercises, we are going to take you through the process of setting up an open research notebook, where you control all of the code and all of the data. A good rule-of-thumb in terms of keeping a notebook is 'one notecard per thought', here adapted as 'one file per thought, one folder per project'.

1. Launch your version of the ODATE Binder (here is our version again).
2. Create a folder called 'research project'. The simplest open research notebook can then be a series of text files that you create inside that project, where one file = one idea. Each file should have the markdown extension, `.md`. You can then cross-link files by making basic links in the text, eg `I found the ideas in [Graham 2016](graham-2016-reading-notes.md) made me think of....`
3. You can also create new python or R notebooks within the folder. Create a new R notebook in your `research project` folder. In the first cell, we're going to write some code that lets you install useful packages for R. R is a language that is open and has a very active ecosystem of researchers creating useful packages that do various things. This ecosystem is peer-reviewed. Once a package has been peer reviewed, we can install it in R with the command `install.packages("name-of-the-package")`. But sometimes we want to install something that is not in the formal ecosystem. Right now, we would like to install a useful package by archaeologist Sebastian Heath called 'cawd' ("Collected Ancient World data sets for R."). In the first cell of your new R notebook, type the following:

   ```
   install.packages("devtools")
   devtools::install_github("sfsheath/cawd")
   ```

```
install.packages("sp")
```

Select the cell, and hit the `run` button.

4. Let's visualize some of this data. Create a new cell. We're going to tell R to use this new library (this new package) we installed, and we're going to look at some data in it.

```
library("sp")
library("cawd")
par(mai=c(0,0,0,0))
plot(awmc.roman.empire.200.sp)
```

This should plot a map of the Roman Empire's extent in 200. Let's see what other data sets are in here:

```
data()
```

Any of the datasets that end with `.sp` can be plotted the same way as we did above.

Play around with the data; more information about the `cawd` see Sebastian Heath's repo here. We're not expecting you to do much yet with this data. Instead, add markdown cells as you play. Because you know how to add links to your markdown, you can also link your work to articles in JSTOR for instance, or other websites - library permalinks, data repositories, and so on. Remember to save your work, and to open the terminal so that you can `git add`, `git commit`, and `git push` your work to your repository.

5. Another option for building your open notebook is to make your markdown files on your own machine, in a text editor like Atom or Sublime Text , and then putting them online so that a templating engine turns them into a navigable website. In this particular case, we are going to use something called `mdwiki`. Mdwiki involves a single html file which, when put in the same folder as a series of markdown files, acts as a kind of wrapper to turn the markdown files into pages of a wiki-style website. There is a lot of customization possible, but for now we're going to make a basic notebook out of a mdwiki template.

a. Fork the minimal mdwiki template to your Github account; md wiki template is linked here

b. At this point, any markdown file you create and save into the `mdwiki-seed\ll_CC\` folder will become a webpage, although *the .md extension should still be used in the URL* . If you study the folder structure, you'll see that there are pre-made folders for pages, for pdfs, for images, and so on (if you clone the repo, you can then add or remove these folders as you like using the file manager). Remembering to frame any internal links as relative links. That is to say, if you saved a markdown file in `ll_CC/pages/observation1.md` but wanted to link to `ll_CC/pages/observation2.md`, it is enough to just add `[Click here](observation2.md)`. Because the `mdwiki-seed` you forked was *already* on a `gh-pages` branch, your notebook will be visible at `YOURUSERNAME.github.io/mdwiki-seed/`. But note: the page will reload and you'll see `#!` or 'hashbang' inserted at the end of the URL. This is expected behaviour.

c. Let's customize this a bit. Via Github, click on the `ll_CC` directory. One of the files that will be listed is `config.json`. If you click on that file, you'll see:

```
{
 "additionalFooterText": "All content and images &copy; by Your Name Goes Here ",
 "anchorCharacter": "#",
 "lineBreaks": "gfm",
 "title": "Your wiki name",
 "useSideMenu": true
}
```

Change the title so that it says something like `Your-name Open Research Notebook`. You can do this by clicking on the pencil icon at the top right of the file viewer (if you don't see a pencil icon, you might not be logged into github). Scroll to the bottom and click on the 'commit changes' button when you're done.

d. Let's add notes to this notebook. You can do this in two ways. In the first, you clone your mdwiki-seed via the command line, and use the text editor to create new pages in the appropriate folder (in this case, `ll_CC\pages`), then `git commit`, `git add .`, and `git push` to get your changes live online. You can create a kind of table of contents by directing the `ls` command into a new file, like so:

```
$ ls > index.md
```

and then editing that file to turn the filenames into markdown links like so: `[display text for link](filename.md)`.

Alternatively, a more elegant approach to use in conjunction with `mdwiki` is to use Prose.io and keep your notebook live on the web. Prose.io is an editor for files hosted in Github. You log into Prose.io with your github credentials, and select the repository you wish to edit, in this case, `mdwiki-seed`. Then, click on the 'new file' button. This will give you a markdown text editor, and allow you to commit changes to your notebook! **Warning** do not make changes to `index.html` when using `mdwiki`. If you want a particular markdown file to appear as the default page in a folder, call it `index.md` instead. You could then periodically update your cloned copy on your own machine for back up purposes.

Either way, add some notes to the notebook, and make them available online.

1.5 Failing Productively

We have found that students are very nervous about doing digital work because, 'what if it breaks?' and 'what if I can't get it to work?' This is perhaps a result of high-stakes testing and the ways we as educators have inculcated all-or-nothing grading in our courses. There is no room for experimentation, no room for trying things out when the final essay is worth 50% of the course grade, for instance. Playing it safe is a valid response in such an environment. A better approach, from a pedagogical point of view, is to encourage students to explore and try things out, with the grading being focused on documenting the *process* rather than on the final outcome. We will point the reader to Daniel Paul O'Donnel's concept of the unessay; more details behind the link.

Our emphasis on open notebooks has an ulterior motive, and that is to surface the many ways in which digital work sometimes fails. We want to introduce to you the idea of 'failing productively' because there is such a thing as an unproductive failure. There are ways of failing that do not do us much good, and we need - especially with digital work - to consider what 'fail' actually can mean. In the technology world, there are various slogans surrounding the idea of 'fail' - fail fast; move fast and break things; fail better; for instance.

When we talk about 'failing productively' or failing better, it is easy for critics of digital archaeology (or the digital humanities; see Allington et al 2016 but contra: Greenspan 2016) to connect digital work to the worst excesses of the tech sector. But again, this is to misunderstand what 'fail' should mean. The understanding of many tech startup folks that valorizing failure as a license to burn through funding has caused a lot of harm. The tech sector failed to understand the humanistic implication of the phrase, and instead took it literally to mean 'a lack of success is itself the goal'. But where does it come from?

The earliest use of the 'fail better' idea that we have found outside the world of literature and criticism seems to occur during the first tech boom, where it turns up in everything from diet books to learning to play jazz, to technology is in *The Quest for a Unified Theory of Information.* CITATION (Wolfgang Hofkirchner) That book according to Google Scholar at the time of writing has been cited 13 times, but the things that cite it have themselves been cited over 600 times. We are here merely speculating on where this mantra of the fast fail, fail better, comes from and how it spreads, but it would be a very interesting topic to explore.

Perhaps a better understanding of what 'fail' should mean is as something akin to what Nassim Taleb called 'antifragility'. The fragile thing breaks under stress and randomness; the resilient thing stays the same; and the anti-fragile thing actually gets stronger as it is exposed to randomness. Kids' bones for instance need to be exposed to shocks in order to get stronger. Academia's systems are 'fragile' in that they do not tolerate fail; they are to a degree resilient, but they are not 'antifragile' in Taleb's sense. The idea that 'fail' can break

that which is 'fragile' is part of the issue here. So silicon valley really means 'fail' in the sense of 'antifragile' but they frequently forget that; academia sees 'fail' as the breaking of something fragile; and so the two are at loggerheads. Indeed, the rhetorical moves of academe often frame weak results - fails - as actual successes, thus making the scholarship fragile (hence the fierceness of academic disputes when results are challenged, sometimes). To make scholarship anti-fragile - to extract the full value of a fail and make it be productive, we need remember only one thing:

A failure shared is not a failure.

Not every experiment results in success; indeed, the failures are richer experiences because as academics we are loathe to say when something did not work – but how else will anybody know that a particular method, or approach, is flawed? If we try something, it does not work, and we then critically analyze why that should be, we have in fact entered a circle of positive feedback. This perspective is informed by our research into game based learning. A good game keeps the challenges just ahead of the player's (student's) ability, to create a state of 'flow'. Critical failure is part of this: too hard, and the player quits; too easy, and the player drops the controller in disgust. The 'fails' that happen in a state of flow enable the player to learn how to overcome them. Perhaps if we can design assessment to tap into this state of flow, then we can create the conditions for continual learning and growth (see for instance Kee, Graham, et al. 2009). As in our teaching, so too in our research. Presner writes,

> Digital projects in the Humanities, Social Sciences, and Arts share with experimental practices in the Sciences a willingness to be open about iteration and negative results. As such, experimentation and trial-and-error are inherent parts of digital research and must be recognized to carry risk. The processes of experimentation can be documented and prove to be essential in the long-term development process of an idea or project. White papers, sets of best practices, new design environments, and publications can result from such projects and these should be considered in the review process. Experimentation and risk-taking in scholarship represent the best of what the university, in all its many disciplines, has to offer society. To treat scholarship that takes on risk and the challenge of experimentation as an activity of secondary (or no) value for promotion and advancement, can only serve to reduce innovation, reward mediocrity, and retard the development of research. PRESSNER 2012 cite

1.5.1 A taxonomy of fails

There are fails, and then there are fails. Croxall and Warnick identify a taxonomy of four kinds of failure in digital work:

1. Technological Failure
2. Human Failure
3. Failure as Artifact
4. Failure as Epistemology

. . . to which we might add a fifth kind of fail:

5. Failing to Share

The first is the simplest: something simply did not work. The code is buggy, dust and grit got into the fan and the hardware seized. The second, while labeled 'human failure' really means that the *context*, the framework for encountering the technology was not erected properly, leading to a failure to appreciate what the technology could do or how it was intended to be used. This kind of failure can also emerge when we ourselves are not open to the possibilities or work that the technology entails. The next two kinds of failure emerge from the first in that they are ways of dealing with the first two kinds of failure. 'Failure as Artifact' means that we seek out examples of failures as things to study, working out the implications of why something did not work. Finally, 'Failure as Epistemology' purposely builds the opportunity to fail into the research design, such that each succeeding fail (of type 1 or type 2) moves us closer to the solution that we need. The first two refer to what happened; the second two refer to our response and how we react to the first two (*if* we react at all). The key to productive failure as we envision it is to recognize when one's work is

suffering from a type 1 or type 2 fail, and to transform it to a type 3 or type 4. Perhaps there should be a fifth category though, a failure to share. For digital archaeology to move forward, we need to know where the fails are and how to move beyond them, such that we move forward as a whole. Report not just the things that work, but also the fails. That is why we keep open research notebooks.

Lets consider some digital projects that we have been involved in, and categorize the kinds of fails they suffered from. We turn first to the HeritageCrowd project that Graham established in 2011. This project straddled community history and cultural history in a region poorly served by the internet. It was meant to crowd-source intangible heritage via a combination of web-platform and telephony (people could phone in with stories, which were automatically transcribed and added to a webmap). The first write-up of the project was published just as the project started to get underway (CITATION GRAHAM ETAL). It's what happened next that is of interest here.

The project website was hacked, knocked offline and utterly compromised. The project failed.

Why did it fail? It was a combination of at least four distinct problems (GRAHAM CITATION):

1. poor record keeping of the *installation* process of the various technologies that made it work
2. computers talk to other computers to persuade them to do things. In this case, one computer injected malicious code into the technologies Graham was using to map the stories
3. Graham ignored security warnings from the main platform's maintainers
4. Backups and versioning: there were none.

Graham's fails here are of both type 1 and type 2. In terms of type 2, his failure to keep careful notes on how the various pieces of the project were made to fit together meant that he lacked the framework to understand how he had made the project vulnerable to attack. The actual fail point of the attack - that's a type 1 fail, but could have been avoided if Graham had participated more in the spirit of open software development, eg, read the security warnings in the developer forum! When Graham realized what had happened to his project, he was faced with two options. One option, having already published a piece on the project that hailed its successes and broad lessons for crowdsourcing cultural heritage, would have been to quietly walked away from the project (perhaps putting up a new website avverring that version 2.0 was coming, pending funding). The other option was to warn folks to beef up the security and backups for their own projects. At the time, crowdsourcing was very much an academic fashion and Graham opted for the second option in that spirit. In doing this, the HeritageCrowd project became a fail of type 3, an artifact for study and reflection. The act of blogging his post-mortem makes this project also an instance of type 5, or the communication of the fail. It is worth pointing out here that the *public* sharing of this failure is not without issues. As we indicated in the Open Notebook Research & Scholarly Communication section, the venue for sharing what hasn't worked and the lessons learned is highly contingent on many factors. Graham, as a white male tenure-track academic on the web in 2012, could share openly that things had not worked. As the web has developed in the intervening years, and with the increasing polarization of web discourse into broad ideological camps, it may well not be safe for someone in a more precarious position to share so openly. One must keep in mind one's own situation and adapt what we argue for here accordingly. Sharing fails can be done with close colleagues, students, specialist forums and so on.

If we are successful with ODATE, and the ideas of productive fail begin to permeate more widely in the teaching of digital archaeology, then a pedagogy that values fail will with time normalize such 'negative results'. We are motivated by this belief that digital archaeology is defined by the productive, pedagogical fail. It is this aspect of digital archaeology that also makes it a kind of public archaeology, and that failing in public can be the most powerful thing that digital archaeology offers the wider field.

We implore you to do your research so that others can retrace your steps; even a partial step forward is a step forward! When you find a colleague struggling, give positive and meaningful feedback. Be open about your own struggles, but get validation of your skills if necessary. Build things that make you feel good about your work into your work.

1.5.2 Exercises

What are the nature of your own fails? Reflect on a 'fail' that happened this past year. Where might it fit on the taxonomy? Share this fail via your open notebook, blog, or similar, with your classmates. How can your classmate convert their fails into types three or four?

1.6 The Ethics of Big Data in Archaeology

In its 2017 global survey of digital usage, We Are Social, a British marketing firm reported that mobile subscriptions in the Americas, Europe and the Middle East now outnumber their respective resident populations. Overall, the firm concluded that across 239 countries that were surveyed, Web usage and the number of social media users continues to grow, with many residents accessing Web content on smart phones and tablets rather than on personal computers. These are compelling statistics, and make clear that no region in the world is fully digital, and that across the 'digital world', there exists considerable unevenness. Yet, we are unable to discern real barriers to these tools and technologies, and the survey does not shed light on their place in social life.

For example, awareness that rural communities in the Global North (ref), as in the Global South (ref), often do not have equitable access to educational and health resources, facilities and infrastructure spurred initiatives to create low-cost, internet ready devices that can potentially address these shortcomings. One Laptop per Child launched in 2006, sought to 'transform education' by providing the world's poorest children with a low-energy, rugged laptop for under $100 (USD). Its current deployment ranges from schools in the United States (Miami, Florida and Charlotte, North Carolina), Indigenous youth in Canada, and students in the war-torn regions such as Nagorno Karabakh and Armenia, Rwanda, Gaza and Ramallah, and Afghanistan, as well as in Nicaragua, Peru, Paraguay, Uruguay, Kenya, Madagascar, India and Nepal.

In the same vein, in 2011, Aakash, a low-cost, Android-based tablet developed in India, by Indian engineers, and sponsored by the Indian government, was released into public space to bring the digital classroom into the hands of the most marginalized and remote communities in India (Phalkey and Chattapahyay 2016; Chattapadhyay and Phalkey 2016).

These well-intentioned, ambitious ICT projects have in common the belief that digital technologies solve social problems where ever they exist. The projects also share another element: digital tools are thought to be intuitive and empowering thus, they do not require 'teachers' or 'teaching'. As Audrey Watters (2012) suggests, 'parachute' technologies i.e. devices that are dropped into school environments assume that children will 'use [them], hack [them], and prosper'. Yet, these deployment efforts typically do not evaluate student academic achievement through time, nor do they seek to build upon, and expand existing educational infrastructure and resources (Warschauer and Ames 2010: 33-34).

We live, work and play in a globalized world that has serious inequalities in terms of wealth, basic necessities for human life such as clean water, food, housing, and safety, as well as access to education and health services and the opportunity to make a living and found a family (UN Declaration 1949). These are not new concerns; yet, they inform the international and national (often post-colonial) contexts within which archaeology is practiced. That these societal issues transcend the range and scope of any one discipline means that we cannot underestimate the influence of social and political factors on the practice of archaeology.

We believe that digital archaeology can offer insights into the social context of archaeology, and a deeper understanding of its the impact on the practice of archaeology. This in turn can guide us to how we can begin to address social inequalities in the discipline. We offer the following provocations (building on Graham, forthcoming):

1) the ethics of digital public archaeology are the ethics of archaeology
2) the ethics of making digital 'things' are the ethics of labour, power and access/control
3) a digital 'thing' is built and reflects the culture of its maker(s) and thereby invites critical study by anthropologists, archaeologists, historians, etc

4) digital things are entangled in the practices and ethics of contemporary society

.. concerning digital things & what they do, see Morgan 2011 https://www.academia.edu/2997156/Emancipatory_Digital_Archaeology esp p160

1.6.1 discussion

- include discussion of privacy
- ethics of crowdfunding
- ethics of crowdsourcing (& dubious ethics of using amazon's mechanical turk)
- ethics of public archaeology & social media

1.6.2 exercises

- maybe some sort of personal audit of what traces the student is leaving online
- some sort of case study?

Chapter 2

Making Data Durable

Elsewhere in this text we've covered approaches to creating digital archaeology data. Section 1.1 discusses three great habits to maintain while working digitally ref back to "The first steps in going digital are quite easy. They are fundamentally a question of maintaining some basic good habits. Everything else flows from these three habits:" section 1.1.6. These principles will help you and future researchers use your data, reproduce your conclusions and "future-proof" your digital work.

Prepare yourself for a little journey. In a dreamlike state, you find yourself in a time machine, noticing that you have traveled to a point in the far or not-so-distant future. You arrive in your own lab to find a group of researchers puzzling over the information you created in the time before, trying to reconstruct your conclusions and make some kind of sense of it all.

```
"What are these strange codes?"
"Does this thing go with that? It looks like there's a bit missing, but we can't be sure."
"What was on all those corrupted flash drives? Does anyone even have a flash-drive-to-skull-jack conver
"WHAT WAS THIS PERSON THINKING?"
```

It doesn't have to be this way. Most archaeological researchers have encountered "bad" data in the past when trying to understand or reconstruct someone's conclusions from field notes and excavation reports. What makes a dataset bad?

It's not understandable. The dataset may be illegible (poor handwriting, or a bad scan of a hard copy document). It might be made up of codes with no way to decipher their meaning. It might be a digital binary file that can no longer be read by available software (or on a physical format that's now obsolete). do

It can't be easily accessed. The dataset might be saved in a format that only expensive, proprietary software can read. It might be comprised of a lot of fascinating information but challenging to extract from a format like PDF.

It's difficult to reuse. The dataset might use inconsistent terminology. The data might be saved as an image, when in fact it's got a structure. [should a definition of structured data go here or somewhere else?]

With these frustrations in mind, the qualities of "good" data become apparent. Good data are stable, human readable, and accessible. They can be rearranged and remixed. As an added benefit, those same qualities that make data useful and conclusions reproducible can also help to protect against the ravages of time.

Pre-Planning

You'll notice that quality data (however you define the term) take planning. The structure of your data and the practicalities of how you'll handle and store it should be woven into your research design. A reality to confront, however, is that there will be tradeoffs.

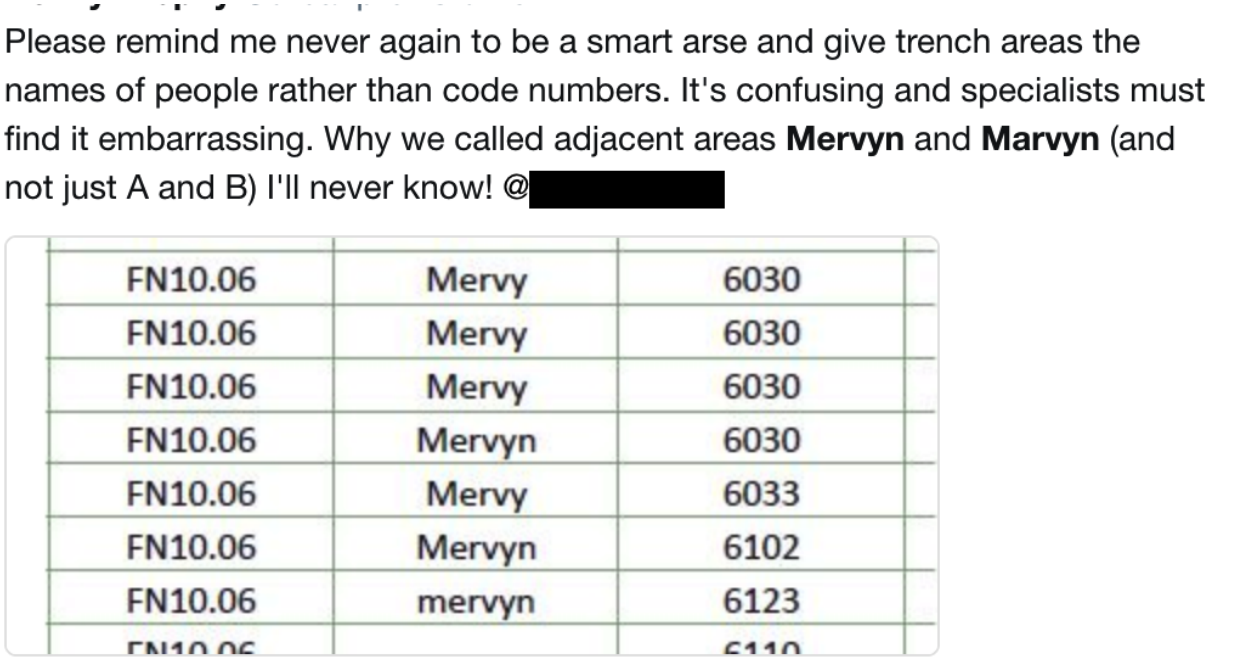


Figure 2.1: A Useful Warning from Kenny Brophy

Up front, right at the beginning, when you're working out your research questions and goals, ask yourself some questions. What do I want my data to do? How much detail do I need to collect? Do I have a plan for curating datasets I've created? Is there space somewhere to do so? Will I need to be selective?

[Cite the grid in Computational Reproducibility in Archaeological Research: Basic Principles and a Case Study of Their Implementation]

Perfection is unattainable. No dataset will ever be truly "complete." You could collect, measure, and produce data at finer and finer resolutions, ad infinitum. But you don't want to do that. And even if you think you're being comprehensive, all data you collect is only a specific, bounded representation of the real world.

The concept of Paradata as implemented in events like the Heritage Jam [link/reference] can be applied broadly to analytical and technical digital projects to great effect. Originally documented in the London Charter for Computer-based Visualization of Cultural Heritage [source and link], paradata (literally data alongside the data) are narrative descriptions of process. Keeping a paradata file can help to illustrate some of the assumptions we naturally make when producing and grooming data. Data are never "raw". [Gitelman, Lisa, ed. Raw data is an oxymoron. MIT Press, 2013.] In the context of an archaeological excavation with published analysis, this information might take the form of sections of a gray literature report or an appendix to a publication. Or it might be a text document that travels along side a collection of data files as a "readme."

Preservation and Longevity

For advice on specific types of data consult *Guides to Good Practice* published by the Archaeology Data Service and Digital Antiquity, 2011.

Time is your enemy - bit rot and link rot - deprecated formats - disappearing web resources and problems with the "cloud" - lost institutional knowledge and unconscious assumptions

Tools - metadata creation - fixity

2.1 Designing Data Collection

In teaching, there is a concept called 'backwards design', where you design your lesson from the end point you wish your students to achieve. The same is true of data. Knowing that archaeology is destructive, and that the processes of archaeology *creates* new kinds of artefacts (drawings, photos, record sheets, labels, and so on), we need to start at the end. Ideally, it is an end point that will allow someone else to reconsider the archaeological record that you have created. As Gavin Lucas (Lucas 2012), the idea of an 'archaeological record' is a three-fold, trinity-like concept-

- the material culture broadly understood
- the 'formation theory' of how that material culture came to be
- and the materialization of the construction of that material culture in the present: our archive

And so, we will begin this section on data collection by thinking through where the *things* are going to go.

Where to put your data

LOCKSS Choosing a repository Does it have room for your stuff? Is your stuff within the purview of its digital archivists? How long is this place going to be around? Does it have plans for migrating and maintaining your files? Choosing what you want to curate Simple formats = easier migration Convert databases to xml or csv tables with relationships Lossless files. TIFF over JPEG CAD- no archival alternative Geospatial- GeoJSON is text-based and human readable 3D- a new frontier in digital preservation (we don't know how to do it)

2.1.1 discussion

2.1.2 exercises

2.2 Cleaning Data with Open Refine

blahde blah blah

2.2.1 discussion

2.2.2 exercises

2.3 Linked Open Data and Data Publishing

yargble blarble floss

2.3.1 discussion

2.3.2 exercises

2.4 Introduction to Digital Libraries, Archives & Repositories

yadda linked open data in here?

- there should be a discussion here on the shape of data; not everything comes in flat files (explain what that means); also json, geojson,

SG June 18: maybe here's where the archdata, finds.org.uk notebooks could go?

2.5 Command Line Methods for Working with APIs

[todo] + simple api call to chronicling america + more complex to open context + some simple plots in the data from open context + calling leaflet from notebook, mapping open context <- but maybe do this in 3.4

see jupyter notebooks, https://github.com/o-date/open-context-jupyter

2.5.1 Working with Omeka

yadda

2.5.2 Working with tDAR

yadda

2.5.3 Working with ADS

2.5.4 Exercises

yadda

Chapter 3

Finding and Communicating the Compelling Story

blah blah blah

3.1 Statistical Computing with R and Python Notebooks; Reproducible code

When one of us (Graham) was a graduate student, he was tasked with teaching undergraduates how to do a chi-squared test of archaeological data. This was in the late 1990s. He duly opened up Excel, and began to craft a template there. While it was possible to use Excel's automatic chi-square stastical test on a table of data, simply showing the students how to use the function seemed undesirable. If he used the function, would the students really *know* what the test was doing? In any event, after several hours of working with the function, he determined that he couldn't actually figure out what exactly the function was doing. The dataset was from Mike Fletcher and Gary R. Lock's very clear *Digging Numbers: Elementary Statistics for Archaeologists* (spearheads and the presence/absence of a loop feature; Fletcher and Lock, 1991: 115-125). The answer Excel was providing was not the answer that Fletcher and Lock demonstrated. Was there some setting somewhere in Excel that was affecting the function? Time was ticking. Graham elected instead to build the spreadsheet so that each step - each calculation - in the statistic was its own cell. He had the students perform these steps on their own spreadsheet. When even that went awry, he saved his sheet onto a floppy disc, and loaded it one computer at a time for the students. This time, working from the original spreadshet, the results agreed with the text. The association between material and period was stronger than the association of material and presence of a loop.

Excel is a black box. When we use it, we have to take on faith that its statistics do what they say they are doing. Most of the time, this is not an issue - but Excel is a very complicated piece of software. Biologists, for instance, have known for years that Excel's automatic date-time conversions (that is, Excel detects the *kind* of data in a cell and changes its formatting or *the actual data itself*) could affect gene names (Zeeberg et al 2004) link. Marwick discusses other issues that occur when using Excel to manipulate our archaeological data. Archaeological data can seldom be used in the format in which they are first recorded. Much effort has to be expended to clean the data up into the neat and tidy tables that a spreadsheet requires. What decisions were made in the tidying process? What bits and pieces were left out? What bits and pieces were transformed as they were entered into the spreadsheet? If someone else were to try to confirm the results of the study - to reproduce it - and there is no record of the manipulations of the data (the specific transformations), reproducibility is unlikely. When researchers discuss a new method, without the data being available and the kinds of transformations and analyses not clearly and fully specified, the researchers have created a barrier to moving archaeology forward. As an exercise right now to understand just how difficult a problem this is,

make a table in Excel. Visualize the following data: 12 bronze spear heads, 18 iron spear heads; 6 of the bronze spear heads have a loop, 10 of the iron spear heads have a loop. Write down all of the steps you took to visualize this information. Save your work; close the spreadsheet. Now hand your list over to a peer. Have them follow the steps exactly, but do not clarify or otherwise explain the list.

Compare what they've created, with what you created. How close of a match is there? Did they reproduce your visualization? What elements or steps did you forget to include in your list?

Now you might begin to see some of the issues involved in relying on a point-and-click graphical user interface and a black-box like Excel with your archaeological data.

Marwick (2017) writes with regard to this problem,

> This is a substantial barrier to progress in archaeology, both in establishing the veracity of previous claims and promoting the growth of new interpretations. Furthermore, if we are to contribute to contemporary conversations outside of archaeology (as we are supposedly well-positioned to do, cf. Kintigh et al. (2014)), we need to become more efficient, interoperative, and flexible in our research. We have to be able to invite researchers from other fields into our research pipelines to collaborate in answering interesting and broad questions about past societies.

Marwick lists four principles for reproducible archaeology. Note that 'reproducible' here means being able to re-do the analysis and obtain the same results; 'replicable' means using the same methods on a new data set collected and analyzed with the published methods of an earlier study. The principles that we should follow are:

- make the data and the methods that generated the results openly available
- script the stastical analysis (that is, no more point-and-click statistics). More on this below.
- use version control
- record the computational environment employed

todo:

- how R addresses these issues
- how R works with version control
- point to Marwick's more complex book on R for archaeologists
- R markdown and jupyter notebooks : literate programming
- what goes in 'discussion' below?
- marwick's compendium

3.1.1 discussion

3.1.2 exercises

3.2 D3, Processing, and Data Driven Documents

blerg

3.2.1 discussion

3.2.2 exercises

3.3 Storytelling and the Archaeological CMS: Omeka, Kora

blargle

3.4 What is Web Mapping?

A web map is a geographic visualization that is supported by computational infrastructures. This form of mapping is fundamentally *powered* by the Web (Axismap, 2018). It should come as no surprise that the ubiquity of web maps correlates with the development of Web 2.0, a marked transformation in the way that that we engage with the Internet (Aced 2013).

As Darcy DiNucci (DiNucci 1999) remarked in 1999, content loading into a browser as 'static screenfuls is only an embryo of the Web to come'. Web 2.0, thus is characterized by 'interactivity', 'two-way communication' often through devices that are 'Internet-enabled' and a greater awareness amongst developers of 'user experience' and 'interface'. These interests are correlated with the diversification of screen sizes on mobile hand-held devices such as smart phones, tablets and laptops, all of which require responsiveness, appropriate scaling for size and interaction primarily through touch and swipe. Whereas the first iteration of the Web required advanced training to create a website, Web 2.0, as is often claimed, requires little or no expert knowledge to make a Web-ready 'thing' such as a website, a blog, or a Wiki. This is the context in which Web mapping, that is the Web-based visualization of geographic information became popularized.

One can argue that web maps narrow the distance between professional map makers and non-specialist makers; yet some scholars have expressed concerns on the quality of geographic data that are now frequently and widely shared on the Web (M. van Exel 2010). Collaborative online platforms such as Open Street Map, for example, have drawn attention to the social dimension of knowledge making; how accurate is information that a lay-person or particular interest community has uploaded? Can we make real-world decisions based on those data? (Daniel Bégin 2013) suggest that volunteered geographic information reflects 'contributors' motivation and individual preferences in selecting mapped features and delineating mapped areas', a situation that can enable scholars in examining and assessing the quality of those data.

For archaeologists, web maps and publishing geographic information present challenges and opportunities. Archaeologists are aware that the data they collect through field studies often contain sensitive location information. These concerns are sometimes heightened in contexts where there are tensions between archaeologists and local communities or ethnic and linguistic minorities who might be socially, politically and economically marginalized in that society. Archaeologists often express concern that publishing location information on sites of archaeological and historical interest can facilitate, if not result in, the destruction of those sites through looting. Looting and illegal trafficking of archaeological artefacts and human bones is an issue observed in many places (Neil Brodie 2001, Huffer and Graham (2017)).

Recent developments in geovisual analytics that leverage the spatial dimension in data suggest that scholars can work with, and meaningfully analyze these data even where they contain sensitive location information (G. Andrienko 2007). This situation opens enormous opportunities for archaeologists to develop tools that are appropriate for meaningful analysis and publication of archaeological data. In the next section, we build upon the ethos of 'openness' and present an overview of map services, followed by a guide to making an interactive web map with the Leaflet library. For an example of a Leaflet web map, check out Open Context.

3.4.1 Overview of Map Services

Tiled map service and Web Map Service are two forms of Web-based mapping. A WMS is an interface that enables us (the clients) to request specific maps i.e. visual representations of geographic information from a geospatial database. The WMS server is called via a Universal Resource Link (URL) on an Internet-enabled desktop GIS. A request typically consists of the geographic layer (e.g. theme) and geographic area of interest. The response to a request results in *geo-registered map images* that are displayed and queried within a browser. Because the map is dynamically drawn upon request, and because the server typically uses the most current information from several layers in the geospatial database, WMS maps tend to load slowly. [Toporama](http://wms.ess-ws.nrcan.gc.ca/wms/toporama_en "target="_blank) is an example of a WMS server for Canadian topographic themes. (add image for WMS architecture?)

Tile map services such as [TillMill Project](http://tilemill-project.github.io/tilemill/docs/crashcourse/introduction/ "target="_blank), [OpenStreetMap](https://www.openstreetmap.org/ "target="_blank), [CartoDB](www.carto.com "target="_blank), and [Stamen](http://maps.stamen.com "target="_blank) all use one or more vector layers that have been rasterized into an image. This rasterized image is divided into 256 x 256 adjacent pixel images or 'tiles'. This is usually the base layer in a web map.

Each tile is an image on the Web, which means that you can link to it. For example, the following URL points to a specific tile on the Web:

```
https://tile.openstreetmap.org/7/63/42.png
```

The three elements in the URL are: 1. `tile.openstreetmap.org`, the tile server name; 2. `7`, the zoom level or `z` value of the tile; and finally 3. `63/42`, the `x` and `y` values in the grid where the tile lives

Z values have a range between 0 and 21, where 21 returns a tile with greatest detail (and smallest sized tile).

Once generated, the set of tiles are stored on disk, ready to be distributed rapidly to large numbers of simultaneous requests. Tiled maps load quickly precisely because they are pre-generated. They shift attention to map aesthetics and smooth map navigation, trading in functionality such as layer order, map scale and projection. [Alex Urquhart](http://alexurquhart.github.io/free-tiles/ "target="_blank) maintains a list of tile services. (add image for tiles?)

Data layers are typically added on top of the base layer. Data layers can be points, lines and polygons. These data layers are saved as [GeoJSON](http://geojson.org/ "target="_blank), a format designed for representing on the Web, geographic features with their non-spatial attributes.

3.4.2 Making a web map with Leaflet

[Leaflet](http://leafletjs.com/ "target="_blank) is a JavaScript library developed by [Vladimir Agafonkin](https://vimeo.com/106112939 "target="_blank) for use with tiled maps. Launched in 2008, Leaflet has become widely used in tile web mapping because the library's low-barrier customization and interactivity with map elements, and because of its simplified setup when compared to a WMS served map. Moreover, Leaflet's compatibility with other Web 2.0 technologies and code-sharing platforms such as [GitHub](www.github.com "target="_blank) has encouraged an active community of 'makers'.

In keeping with this e-book's ethos of 'openness' and with a motivation to encourage low-cost and low-barrier web mapping, below is an outline to get started on our own interactive web map with Leaflet. We will need:

1. some point data, ideally geocoded (lat/long) data saved as CSV;
2. a text editor installed on local machine, such as Atom;
3. a hosting service, such as GitHub (public account);
4. a tile map service, such as OpenStreetMap, MapBox (free account) or other;
5. a curious you

3.4.3 Exercises

Making a 'digital thing' can be exciting and intimidating, and yet seeing one's creation on the Web can be gratifying. It is important to realize that much of the work to make that happen takes place on a local machine. Therefore, setting up a local environment with the tools we need is highly encouraged. We recommend installing a local web server such as [MAMP](https://www.mamp.info/en/ "target="_blank) for Macs or [WampServer](http://www.wampserver.com/en/ "target="_blank) for Windows.

For this exercise, we'll use [Prepros](https://prepros.io/ "target="_blank), a temporary preprocessor application that reloads our local browser as we make changes to our HTML file, and enables us to see what's happening.

3.4.3.1 A simple Leaflet web map

1. Download (web-map) and unzip to known location. Which files and folders do you see? It is helpful to see files and sub-folders within their hierarchical structure before we start editing. Bring the web-map folder into Atom. We do this by 'Add a Project Folder'.
2. Locate the file named `index.html` and open it. What do you see in the file? The first line

   tells us that this document written in **html**, a language for creating web pages.

Go ahead and change the title

```
<title>title when you hover over tab</title>
```

3. We now want to have a look at our web page on a local browser. Let's fire up Prepros (this may take a few minutes) to get a preview. Add the web-map folder as a project, either by drag and drop or use the + at the bottom left of the Prepros window.

Right click on the folder to 'Enable Live Preview'. Then click on the globe icon to see the web page in the browser.

4. On the `index.html`, locate the tag `<head>` and `<body>`. In the anatomy of a web page, these sections are most important for creating and loading content.

Within our `<head>` section, we have added two tags, `<link>` and `<script>` to the Leaflet library. You will notice that the URL points to `CDN` or a Content Delivery Network. These are servers that host Web content based on our geographic location. In this case, we request a specific hosted CSS or Cascading Style Sheet for Leaflet and the Leaflet script that adds interactivity to the web map.

The CSS gives us pre-defined styles and elements to format the content of a webpage i.e. we get the look and feel of a Leaflet map. It includes fonts, size, colour, line spacing, and Leaflet elements like the map, and an icon for zoom.

It is key that that CSS loads before the script, and that both of them are within the `<head></head>` tags.

5. Next, in the `<body>` section of our web page, we add a `<div>` element that will contain a thing called `map`.

```
<div id="map"></div>
```

We then call the `<script>` and initialize our map using `L.map`.

Examine the code

```
<script>
var map = L.map('map').setView([40.5931,-75.5265], 12);
```

`.setView` centers our web map on specific coordinates and at a particular zoom level. Change the coordinates and hit save. Have a look at the live preview. What do you see?

6. We now want to load a tile server for our base map, using `L.tileLayer`

```
L.tileLayer('https://{s}.tile.openstreetmap.org/{z}/{x}/{y}.png', {
    maxZoom: 19,
    attribution: '&copy; <a href="http://www.openstreetmap.org/copyright">OpenStreetMap</a>'
    }).addTo(map);
</script>
```

This code tells us the following: 1. `https://{s}.tile.openstreetmap.org`, tile server we want, 2. `19`, the maximum zoom level, and 3.`'&copy; <a href="http://www.openstreetmap.org/copyright">OpenStreetMap</a>'` the attribution for that tile provider

Take note of `.addTo(map);` that actually adds the layer to our web map, and the `</script>` closes this particular script. To load additional layers or attribute data, we would add our code within the tags `<script></script>` which we outline in the next subsection.

Congratulations! We have our first Leaflet map. Examine the map in the Prepros preview window.

3.4.3.2 Loading point data onto a Leaflet map

Now that we have a base map set up, we want to load some feature data i.e. points, lines, polygons, onto it. We have a few different ways to add feature data. They can be loaded as a Common Separated Value (CSV) file or a GeoJSON.

In this exercise, we will work with a GeoJSON, a small file (point-data.geojson) with about 20 potential excavation sites. The original CSV had four fields, all of which were converted into GeoJSON using an online tool [here](http://www.convertcsv.com/csv-to-geojson.htm/ “target=”_blank).

1. Fire up a text editor and examine the contents of `point-data.geojson`. What do you see? Take note of **type**, **geometry**, and **properties**. How many properties or attributes are there, what are they?
2. Next, open `index.html`. We will now add several lines of code that enable us to grab our GeoJSON data, load them and represent them as markers.

Locate the `<head>` tag, and the script for loading Leaflet. Below it, add a script called jQuery. This Javascript library is widely used to enable interactivity, animations, plug-ins and widgets. We load jQuery on our web page using the following code after Leaflet.js :

```
<!-- loads Leaflet.js -->
<script src="http://cdn.leafletjs.com/leaflet-0.7.3/leaflet.js"></script>
<!-- loads jQuery -->
<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.1/jquery.min.js"></script>
```

3. We are now ready to add a few lines to get our GeoJSON (point-data.geojson). In the `<body>` section, let’s add the following code below your tile layer:

```
// load a tile layer
  L.tileLayer('https://{s}.tile.openstreetmap.org/{z}/{x}/{y}.png', {
    maxZoom: 19,
    attribution: '&copy; <a href="http://www.openstreetmap.org/copyright">OpenStreetMap</a>'
    }).addTo(map);

// load GeoJSON and save it as a thing called `data`
  $.getJSON("point-data.geojson", function(data) {
```

followed by :

```
// adds GeoJSON objects to layer
   data = L.geoJson(data  ,{
```

```
// converts point feature into a map layer
    pointToLayer: function(feature,latlng){
      return L.marker(latlng);
      }
    }).addTo(map);
  });
```

4. Save the file and preview it in the browser. Congratulations! We've added our own point feature data to a Leaflet map.

5. It would be great to have interaction beyond panning and zooming on our web map. One way is to add pop-ups to each of our markers that we can click on.

Locate `pointToLayer` which we called passed a function. We will create a variable called marker and bind a pop-up to each marker:

```
// creates a variable called marker
pointToLayer: function(feature,latlng){
var marker = L.marker(latlng);

// binds a popup to marker, assigns properties to display
  marker.bindPopup(feature.properties.Label + '<br/>');
  return marker;
}
}).addTo(map);
    });
```

6. That's it! We've created a web map with our own point data, and we have markers with pop-ups to click on.

3.4.4 Resources

Below are examples in archaeology that use Leaflet that you can try out, and that have repositories that you can fork for you own projects:

1) The Digital Atlas of Ancient Egypt developed at the Department of Anthropology, Michigan State University: https://msu-anthropology.github.io/daea/

Repository: https://github.com/msu-anthropology/daea

2) TOMB: The Online Map of Bioarchaeology developed by Lisa Bright (Michigan State University): http://brightl1.github.io/TOMB/

Repository: https://github.com/brightl1/TOMB/

3) MINA | Map Indian Archaeology developed by Dr Neha Gupta (Memorial University of Newfoundland): http://dngupta.github.io/mina.github.io/

Repository: https://github.com/dngupta/mina.github.io

3.5 Place-based Interpretation with Locative Augmented Reality

yep.

3.5.1 discussion

3.5.2 exercises

3.6 Virtual Archaeology

What is Virtual Archaeology (VA)? ### Theory blah

3.6.0.1 Making, Wayfaring, Tacking and Archaeological Cabling

blah #### Materiality blah #### Phenomenology blah ### Method blah

3.6.0.2 The London & Seville Charters

blah #### Agency blah #### Authority blah #### Authenticity blah #### Transparency blah #### Paradata blah ### Practice blah #### Paul Reilly blah #### Anthony Masinton blah #### Grant Cox blah #### Richard Allen blah #### Daniel Pletinckx blah #### Nicole Beale blah #### Alice Watterson blah #### Costas Papadopoulos blah #### Ed Gonzalez-Tennant blah

3.7 Archaeogaming

Guest section by Andrew Reinhard, Meghan Dennis, Tara Copplestone, Colleen Morgan?

3.7.1 Discussion

blah ### Exercises blah

3.8 Social media as Public Engagement & Scholarly Communication in Archaeology

boo socmed

3.8.1 discussion

3.8.2 exercises

Chapter 4

Eliding the Digital and the Physical

yadda

4.1 3D Photogrammetry & Structure from Motion

In recent years, faster and more powerful computers have made it feasible to do complex 3d model building by extracting points of overlap in multiple photographs, then extrapolating from the camera metadata embedded in those images the distance from the points to the camera's image sensor. This information allows the reconstruction of where those points were *in space* relative to the camera. Thus astonishingly good 3d models can be created at rather low cost.

Laser scanning, on the other hand, involves shooting rays of light onto an object (or space) and counting the time it takes for the light to return to the scanner. Laser scanners are able therefore to take detailed micro-millimetre scans of an object's surface and texture. For some archaeological purposes, laser scanning is to be preferred. For other purposes, 3d photogrammetry or 'structure from motion' (sfm) is entirely appropriate, and the level of resolution good enough

In this chapter, we'll cover some of the basic principles of how sfm works while pointing you to more detailed discussions. We also provide links in the 'further readings' to some recent case studies using 3d photogrammetry in novel ways.

4.1.1 Basic principles

- image capture
- image matching
- dense point cloud generation
- secondary product generation
- analysis / presentation

take overlapping images; you want a high degree of overlap

tie points are matched

camera orientations are deduced

dense point cloud generated

chief ray is the line from the object, through the lens, to the image plate

intersection of many of these allows the software to work out the location of the point in space

- knowing the 'interior and exterior' orientation of the camera - its internal arrangements, including lens distortion, focal length and so on from the metadata bundled with the image, allows software to work out the position of the camera viz points of overlap in the images
- the intersection of rays then allows us to work out the location of these points in space
- resulting point cloud has an arbitrary scale and coordinate frame (ways around this)

sfm process - identifies control points that are visible in multiple images (default in agisoft photoscan is 40 000 *per image*) - best control points are then matched - then triangulation (more or less) to work out the relative position & orientation of every image

this is the sparse reconstruction

now dense reconstruction

- repeats the process above but for all possible control points
- a 'triangulated irregular network' TIN is generated, a mesh, onto which textures can be mapped using regular graphics software

then what?

- download to your computer; clean up

from regard3d: To obtain a 3D model, the following steps are performed:

For each image, features (sometimes also called keypoints) are detected. Features are points in an object that have a high probability to be found in different images of the same object, for example corners, edges etc. Regard3D uses A-KAZE for this purpose. For each feature, a mathematical descriptor is calculated. This descriptor has the characteristic that descriptors of the same point in an object in different images (seen from different viewpoints) is similar. Regard3D uses LIOP (Local Intensity Order Pattern) for this purpose. The descriptors from different images are matched and geometrically filtered. The result of this step is a collection of matches between each image pair. Now "tracks" are calculated. For each feature that is part of a match in an image pair, it is searched also in other images. A track is generated from features if these features satisfy some conditions, for example a track is seen in at least 3 images. The next step is the triangulation phase. All the matches of all the image pairs are used to calculate: The 3D position and characteristic of the "camera", i.e. where each image was shot and the visual characteristics of the camera The 3D position of each "track" is calculated The result of the triangulation phase is a sparse point cloud. In order to obtain a more dense point cloud ("densification"), several algorithms can be used. The last step is called "Surface generation". The point clouds are used to generate a surface, either with colored vertices or with a texture.

4.1.2 exercises

While there are command-line applications (like VSFM) for photogrammetry, running such things from a Jupyter notebook does not work very well. VSFM *can* be installed in something like DHBox (but it's not for the faint-of-heart, see eg this). Roman Hiestand built a graphical user interface around a series of open-source modules that, when combined in a workflow, enables you to experiment with photogrammetry. With a bit of hacking, we can also make it work with photographs taken from smartphone or tablet.

Download and install the relevant version of Regard3d for your operating system.

1. Try the Heistand's tutorial using the images of the Sceaux Castle. This tutorial gives you a sense of the basic workflow for using Regard3d.
2. Take your own photographs of an object. Try to capture it from every angle, making sure that there is a high amount of overlap from one photograph to another. Around 20 photographs can be enough to make a model, although more data is normally better. Copy these photos to your computer. A note on smartphone cameras: While many people now have powerful cameras in their pockets in the form of smartphones, these cameras are not in Regard3d's database of cameras and sensor widths. If you're using a smartphone camera, you will have to add this data to the metadata of the images, and then

add the 'camera' to the database of cameras. **If you've taken pictures with an actual digital camera, chances are that this information is already present in the Regard3d database**. You'll know if you need to add information if you add a picture set to Regard3d and it says 'NA' beside the picture.

Open Regard3d and start a new project. Add a photoset by selecting the directory where you saved the photos.

Click ok to use the images. **If Regard3d doesn't recognize your camera, check the Adding metadata to images section below**.

Click on compute matches. Slide the keypoint density sliders (two sliders) all the way to 'ultra'. You can try with just the default values at first, which is faster, but using 'ultra' means we get as many data points as possible, which can be necessary given our source images. This might take some time. When it is finished, proceed through the next steps as Regard3d presents them to you (the options in the bottom left panel of the program are context-specific. If you want to revisit a previous step and try different settings, select the results from that step in the inspector panel top left to redo).

The final procedure in model generation is to compute the surfaces. When you click on the 'surface' button (having just completed the 'densification' step), make sure to tick off the 'texture' radio button. When this step is complete, you can hit the 'export' button. The model will be in your project folder - .obj, .stl., and .png. To share the model on something like Sketchfab.com zip these three files into a single zip folder. On Sketchfab (or similar services), you would upload the zip folder. These services would then unzip the folder, and their 3d viewers know how to read and display your data.

3. Cleaning up a model with Meshlab Building a 3d model takes skill, patience, and practice. No model ever appears 'perfect' on the first try. We can 'fix' a number of issues in a 3d model by opening it in a 3d editing programme. There are many such programmes out there, with various degrees of user-friendliness. One open-source package that is often used is Meshlab. It is very powerful, but not that intuitive or friendly. It does not 'undo'.

Once you have downloaded and installed Meshlab, double-click on the .obj file in your project folder. Meshlab will open and display your model. The exact tools you might wish to use to enhance or clean up your model depends very much on how your model turned out. At the very least, you'll use the 'vertice select' tool (which allows you to draw a box over the offending part) and the 'vertice delete' tool.

SG: screenshots, paths to frequently used useful tools in this regard.

4.1.2.1 Adding metadata to images

1. Go to https://www.sno.phy.queensu.ca/~phil/exiftool/index.html and download the version of the Exiftool appropriate to your computer.

- **Windows users** you need to fully extract the tool from the zipped download. **THEN** you need to rename the file to just `exiftool.exe`. When you extract it, the tool name is `exiftool(-k).exe`. Delete the `(-k)` in the file name.
- **Move** the file `exiftool.exe` to the folder where your images are.
- **Mac users** Unzip if you need to, double click on the dmg file, follow the prompts. You're good to go.

2. Navigate to where your images are store. Windows users, search your machine for `command prompt`. Mac users, search your machine for `terminal`. Run that program. This opens up a window where you can type commands into your computer. You use the `cd` command to 'change directories', followed by the exact location (path) of your images. On a PC it'll probably look something like `cd c:\users\yourname\documents\myimages`. When you're in the location, `dir` will show you everything in that director. Mac users, the command `ls` will list the directory contents. Make sure you're in the right location, (and windows users, that `exiftool.exe` is in that directory).

3. The following commands will add the required metadata to your images. Note that each command is saying, in effect, exiftool, change the following metadata field to this new setting for the following image. the `*.jpeg` means, every single jpeg in this folder. **NB** if your files end with .jpg, you'd use `.jpg`, right?

```
exiftool -FocalLength="3.97" *.jpeg
```

This sets the focal length of your image at 3.97 mm. You should search for your cellphone make online to see if you can find the actual measurement. If you can't find it, 3.97 is probably close enough.

```
exiftool -Make="CameraMake" *.jpeg
```

You can use whatever value you want instead of `CameraMake`. E.g., `myphone` works.

```
exiftool -Model="CameraModel" *.jpeg
```

You can use whatever value you want, eg `LG3`.

If all goes according to plan, the computer will report the number of images modified. Exiftool also makes a copy of your images with the new file extension, `.jpeg_original` so that if something goes wrong, you can delete the new files and restore the old ones by changing their file names (eg, remove `_original` from the name).

4. Regard3d looks for that metadata in order to do the calculations that generate the point cloud from which the model is created. It needs the focal length, and it needs the size of the image sensor to work. It reads the metadata on make and model and compares it against a database of cameras to get the size of the image sensor plate. Oddly enough, this information is **not** encoded in the image metadata, which is why we need the database. This database is just a text file that uses commas to delimit the fields of information. The pattern looks like this: `make;model;width-in-mm`. EG: `Canon;Canon-sure-shot;6`. So, we find that file, and we add that information at the end of it.

- **windows users** This information will be at this location:

```
C:\Users\[User name]\AppData\Local\Regard3D
```

eg, on my PC:

```
C:\Users\ShawnGraham\AppData\Local\Regard3D
```

and is in the file "sensor_database.csv".

- ***mac users** Open your finder, and hit shift+command+g and go to

```
/Applications/Regard3D.app/Contents/Resources
```

- Do not open `sensor_database.csv` with Excel; Excel will add hidden characters which cause trouble. Instead, you need a proper text editor to work with the file (notepad or wordpad are not useful here). One good option is Sublime Text. Download, install, and use it to open `sensor_database.csv`
- Add whatever you put in for camera make and camera model (back in step 3) *exactly* - upper-case/lowercase matters. You can search to find the size of the image sensor for your cell phone camera. Use that info if you can find it; otherwise 6 mm is probably pretty close. The line you add to the database then will look something like this:

```
- Save.

Now you can open Regard3d, 'add a picture set', select these images, and Regard3d will have all of the
```

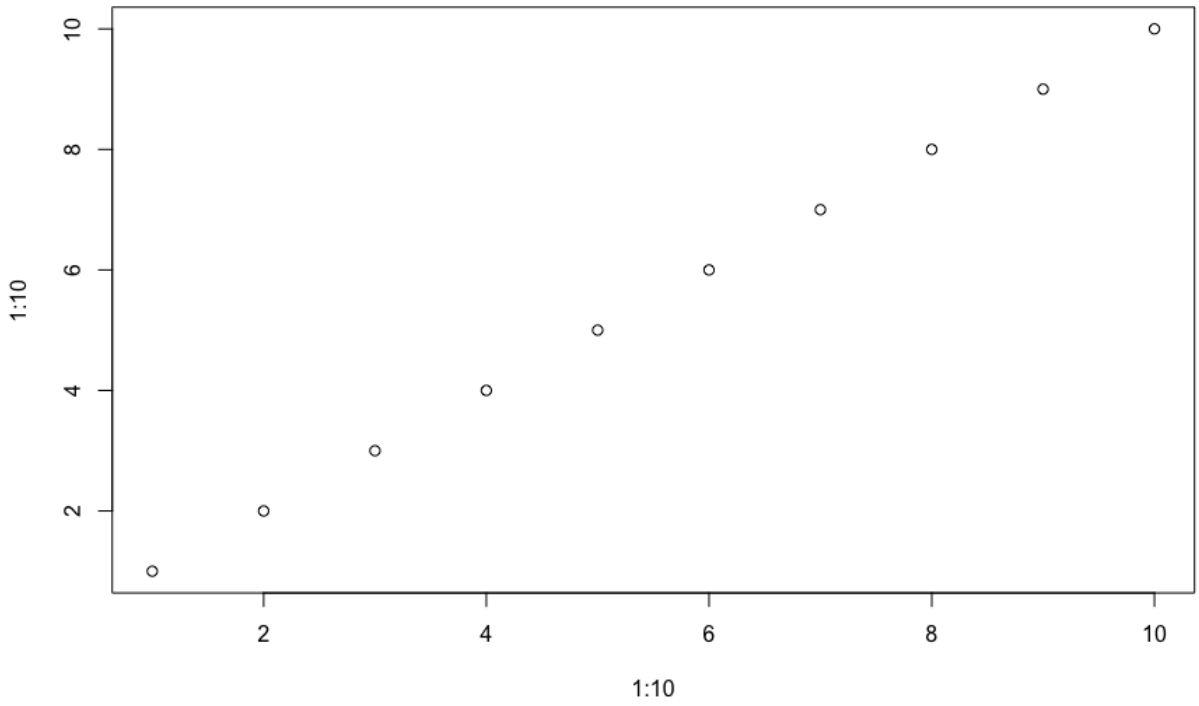
```
<!--chapter:end:04.1-3d-photogrammetry.rmd-->

## 3D Printing, the Internet of Things and "Maker" Archaeology

yay

### discussion

### exercises

<!--chapter:end:04.2-3d-printing.rmd-->


## Artificial Intelligence in Digital Archaeology

To speak of 'artificial intelligence' in archaeology may be to speak too soon yet. We do have machine le

Then why use the term? We think it is still useful to use the term because it reminds us that, in using

Machine learning: a series of techniques that endeavour to train a computer program to identify and clas

Agent based simulation: a series of techniques that create a population of software 'agents' who are pro

The value of machine learning: it makes us think carefully about what we are looking for and at in the i

The value of agent-based modeling: it forces us to think carefully about what it is we think actually *k


### Agent-based modeling (ABM)

This section is an overview of agent-based modeling (**_ABM_**), as it is used within archaeology, anthr

Following the definition of ABM, we describe the process of creating a model or *formalizing* an informa

Next, we cover the *why* and *how* of the application of ABM in archaeology, history, and social science

Last, we walkthrough the steps involved creating an ABM model. Inspired in the theme of emergence and co

Throughout this section, several concepts will probably sound *alien* for most history and archaeology s

To the date, there is no easy way to begin doing ABM. We encourage students to engage in modelling the s

#### What is ABM?

>"Essentially, all models are wrong, but some are useful"
>George Box, 1987
>*Empirical Model-Building and Response Surfaces*, p. 424

The first thing to consider is that ABM is about models. A model is a representation of a phenomenon as

![Pineapples and their (visual) models](images/andros_pineapple.png)

Another important statement about models is that they are--as any mental process--a phenomenon in its ou
```

```
Well, what do pineapple drawings have in common with doing models in archaeology? You may imagine that,

Why bother then doing models in academia? Ultimately, we are searching for knowledge, staments with at l

Unfortunately, models involving human behavior are less straightforward to validate or dismiss than a "s

Fine. Models are not only inevitable for any living mind, but they are also the keystone for the generat

Formal models--or rather *formalizing* informal models--reduce ambiguity, setting aside part of the risk

Formalization often demand the use of formal logic and some extend of quantification. Seen from the pers

A formal definition is a disclaimer saying: *"This is what X means in this model, no more, no less. You



- Why model with ABM? open with the idea that we are always modeling
- Mention review article 'Why model?'

- def system, model, formal model
- The origins of computer simulation
- 3d models and statistical models (e.g., regression) are not simulation models. Simulation implies the
- Equation-based modeling, System Dynamics - ODE systems as models (mention Lorenz attractor, end with t
- Algorithm-based modeling, Algorithms as behavioral models (mention cellular automata, Van Neumann, Sch
- The variety of algorithm-based models, much coincides with the variety of programming paradigm.
  a. Flow: spectrum between centralized (iteration of a single process, e.g. event) and distributed (mul
  b. Schedule: spectrum between fully-scheduled (processes are initiated in a predetermined order) and c
ABMs fall somewhere in-between these two spectra, though normally leaning towards distributed (e.g., usi
- Parts of an ABM model
 (re-visit presentations)

#### How to model with ABM?

- Define hypotheses, questions or simply the phenomenon of interest (identify the system)
- Define the elements and processes (you believe are) required to address the system (model the system)
- Express the chosen elements and processes as computer code (implement the model)
- Modeling is a process of constant iteration. Each stage is an iteration loop that is often repeated se
- ABM in python, Java, C#, or C++? in R? in NetLogo? (Any language can do it, really). Pros and cons, ta
- remember to talk about https://www.openabm.org/

#### ABM in Archaeology and Social Sciences

- Emergence... Life game, deterministic chaos
- Saving the gap between atomism and holism, or individual and social structure
- Virtual laboratory for social sciences
- Examples
- Mention the cycles of enthusiasm and skepticism, and the tendency to develop closed circles
- Challenges and pitfalls

#### ABM tutorial: the Pond Trade model

brief overview
disclaimer: MY modelling strategy is highly focused in theory-building (general, explorative), not hypot
```

```
##### Phenomena of interest and conceptual model

- phenomena: cycles of growth and collapse, i.e. flutuations in the scale of site occupation, out of pha
- Belief or main assumption: topography, transport technology, trade, settlement size and wealth, and cu

![andros-first-diagram](images/andros-diagram.png)
```

library(DiagrammeR) grViz(“diagrams/boxes.dot”)

```
- Elements: "pond" or water body, settlements, ships, routes, goods.
- Rules:
  - coastal settlements of variable size around a pond.
  - each settlement has a "cultural vector".
  - trade ships travel between the settlements.
  - once in their base settlement, ships evaluate all possible trips and choose the one with the greate
  - ships carry economic value and cultural traits between the base and destination settlements.
  - settlements size depends on the economic value they receive from trade.
  - the economic value produced in a settlement depends on its size.
  - the number of ships per settlement depends on its size.

##### Implementation

- Implement the model based on general definition (agents, variables, parameters)
- Walk-through the implementation in NetLogo

Pond Trade model repository: https://github.com/Andros-Spica/PondTrade

![The model interface in step 13](images/interface-andros.png)

##### Simulation experiments and analysis

- Simulations and experiment design
- Analysis and display of results (R example)
```



plot(1:10, 1:10) “‘

SG: code chunks didn’t render correctly through the bookdown generation process, which is baffling. check the original bookdown book for how to sort this out. removed the executable code chunk, put it into a different script, ran it, copied the output here. this is not optimal.

AA: didn’t find anything useful about that... Anyhow, I can continue by loading images and leaving r chunks as generic code

4.2 Computer Vision and Archaeology

It has become practical in recent years to use neural networks to identify objects, people, and places, in photographs. This use of neural networks *applied to imagery* in particular has seen rapid advancement since 2012 and the first appearance of ‘convolutional’ neural networks (Deshpande (2016) provides an accessible guide to this literature). But neural networks in general have appeared sporadically in the archaeological literature since the 1990s; Baxter (2014) provides a useful overview. Recent interesting uses include Benhabiles and Tabia (2016) which uses the approach to enhance pottery databases, and Wang et al. (2017) on stylistic analysis of statuary as an aid to restoration. In this section, we provide a gentle introduction to how convolutional neural networks work as preparation, and then two jupyter binders that may be repurposed or expanded with more data to create actual working classifiers.

4.2.1 Convolutional Neural Networks

Neural networks are a biological metaphor for a kind of sequence of computations drawing on the architecture of the eye, the optic nerve, and the brain. When the retina of the eye is exposed to light, different structures within the retina react to different aspects of the image being projected against the retina. These ‘fire’ with greater or lesser strength, depending on what is being viewed. Subsequent neurons will fire if the signal(s)

they receive are strong enough. These differential cascades of firing neurons 'light up' the brain in particular, repeatable ways when exposed to different images. Computational neural networks aim to achieve a similar effect. In the same way that a child eventually learns to recognize *this* pattern of shapes and colour as an 'apple' and *that* pattern as an 'orange', we can train the computer to 'know' that a particular pattern of activations *should be* labelled 'apple'.

A 'convolutional' neural network begins by 'looking' at an image in terms of its most basic features - curves or areas of contiguous colour. As the information percolates through the network the layers are sensitive to more and more abstraction in the image, some 2048 different dimensions of information. English does not have words to understand *what* precisely, some (most) of these dimensions are responding to, although if you've seen any of the 'Deep Dream' artworks [SG insert figure here] you are seeing a visualization of some of those dimensions of data. The final layer of neurons predicts from the 2048 dimensions what the image is supposed to be. When we are training such a network, we know at the beginning what the image is of; if at the end, the network does not correctly predict 'apple', this error causes the network to shift its weighting of connections between neurons back through the network ('backpropogation') to increase the chances of a correct response. This process of calculation, guess, evaluation, adjustment goes on until no more improvement seems to occur.

Neural networks like this can have very complicated architectures to increase their speed, or their accuracy, or some other feature of interest to the researcher. In general, such neural networks are composed of four kinds of layers. The first is the **convolutional** layer. This is a kind of filter that responds to different aspects of an image; it moves across the image from left to right, top to bottom (whence comes the name 'convolutional'). The next layer is the layer that reacts to the information provided by the filter; it is the **activation** layer. The neural network is dealing with an astounding amount of information at this point, and so the third layer, the **pooling** layer does a kind of mathematical reduction or compression to strip out the noise and leave only the most important features in the data. Any particular neural network might have several such 'sandwiches' of neurons arranged in particular ways. The last layer is the **connected** layer, which is the layer with the information concerning the labels. These neurons run a kind of 'vote' on whether or not the 2048-dimension representation of the image 'belongs' to their particular category. This vote is expressed as a percentage, and is typically what we see as the output of a CNN applied to the problem of image identification.

4.2.2 Applications

Training a neural network to recognize categories of objects is massively computationally intense. Google's Inception3 model - that is, the final state of the neural network Google trained - took the resources of a massive company to put together and millions of images. However, Google *released* its model to the public. Now anyone can take that *finished* pattern of weights and neurons and use them in their own applications. But Google didn't train their model on archaeological materials, so it's reasonable to wonder if such a model has any value to us.

It turns out that it does, because of an interesting by-product of the way the model was trained and created. **Transfer learning** allows us to take the high-dimensional ways-of-seeing that the Inception3 model has learned, and apply them to a tweaked final voting layer. We can give the computer mere thousands of images and tell it to learn *these* categories: and so we can train an image classifier on different kinds of pottery relatively quickly. Google has also released a version of Inception3 called Mobilnet that is much smaller (only 1001 dimensions or ways-of-seeing) and can be used in conjunction with a smartphone. We can use transfer learning on the smaller model as well and create a smartphone application trained to recognize Roman pottery fabrics, for instance.

The focus on identifying objects in photographs does obscure an interesting aspect of the model - that is, there are interesting and useful things that can be done when we dismiss the labeling. The second-to-last layer of the neural network is the numerical representation of the feature-map of the image. We don't need to know what the image is of in order to make use of that information. We can instead feed these representations of the the images into various kinds of k-means, nearest-neighbour, t-sne, or other kinds of statistical tools to look for pattern and structure in the data. If our images are from tourist photos uploaded to flickr of archaeological sites, we might use such tools to understand how tourists are framing their photos (and so,

their archaeological consciousness). Graham and Huffer (2018) are using this tool to identify visual tropes in the photographs connected to the communities of people who buy, sell, and collect photos of, human remains on Instagram. Historians are using this approach to understand patterns in 19th century photographs; others are looking at the evolution of advertising in print media.

4.2.3 Exercises

1. Build an image classifier. The code for this exercise is in our repo; launch the binder and work carefully through the steps. Pay attention to the various 'flags' that you can set for the training script. Google them; what do they do? Can you improve the speed of the transfer learning? The accuracy? Use what you've learned in section 2.5 to retrieve more data upon which you might build a classifier (hint: there's a script in the repo that might help you with that).
2. Classify similar images. The code for this exercise is in Shawn Graham's repo; launch the binder and work through the steps. Add more image data so that the results are clearer.

Chapter 5

Digital Archaeology's Place in the World

blerg

5.1 Marketing Digital Archaeology

blag

5.1.1 discussion

5.1.2 exercises

5.2 Sustainability & Power in Digital Archaeology

the big ticket item.

5.2.1 discussion

5.2.2 exercises

Chapter 6

On the Horizons: Where Digital Archaeology Might Go Next

blargble

References

Aced, Cristina. 2013. "Web 2.0: The Origin of the Word That Has Changed the Way We Understand Public Relations." *International PR 2013 Conference. Images of Public Relations*. https://www.researchgate.net/publication/266672416_Web_20_the_origin_of_the_word_that_has_changed_the_way_we_understand_public_relations.

Baxter, Mike. 2014. "Neural Networks in Archaeology." https://www.academia.edu/8434624/Neural_networks_in_archaeology.

Benhabiles, Halim, and Hedi Tabia. 2016. "Convolutional Neural Network for Pottery Retrieval." *Journal of Electronic Imaging* 26. https://doi.org/10.1117/1.JEI.26.1.011005.

Boettiger, Carl"Welcome to My Lab Notebook - Reloaded." Website. *Lab Notebook*. http://www.carlboettiger.info/09/28/Welcome-to-my-lab-notebook.html.

Caraher, William. 2012. "Archaeological Glitch Art." *The Archaeology of the Mediterranean World*. https://mediterraneanworld.wordpress.com/2012/11/21/archaeological-glitch-art/.

Costopoulos, Andre. 2016. "Digital Archeology Is Here (and Has Been for a While)." *Frontiers in Digital Humanities* 3: 4. doi:10.3389/fdigh.2016.00004.

Daniel Bégin, S Roche, Rodolphe Devillers. 2013. "Assesing Volenteered Geographic Information (Vgi) Quality Based on Contributors' Mapping Behaviours." *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*. https://www.int-arch-photogramm-remote-sens-spatial-inf-sci.net/XL-2-W1/149/2013/isprsarchives-XL-2-W1-149-2013.pdf.

Deetz, James. 1965. *The Dynamics of Stylistic Change in Arikara Ceramics*. Urbana: University of Illinois Press.

Deshpande, Adit. 2016. "The 9 Deep Learning Papers You Need to Know About." https://adeshpande3.github.io/adeshpande3.github.io/The-9-Deep-Learning-Papers-You-Need-To-Know-About.html.

DiNucci, Darcy. 1999. "Fragmented Future." *Print Magazine*. http://darcyd.com/fragmented_future.pdf.

Ethan Watrall. 2017. "Archaeology, the Digital Humanities, and the 'Big Tent'." In *Debates in the Digital Humanities*, 2016th ed. Accessed February 23. http://dhdebates.gc.cuny.edu/debates/text/79.

Evans, Thomas L., Patrick T. Daly, and MyiLibrary, eds. 2006. *Digital Archaeology: Bridging Method and Theory*. London ; New York: Routledge. http://proxy.library.carleton.ca/login?url=http://www.myilibrary.com?id=29182.

G. Andrienko, P. Jankowski, N. Andrienko. 2007. "Geovisual Analytics for Spatial Decision Support. Setting the Research Agenda." *International Journal of Geographical Information Science* 21 (8): 839–57.

Goldstone, Andrew. 2018. "Teaching Quantitative Methods: What Makes It Hard 9in Literary Studies)." In *Debates in the Digital Humanities*.

Graham, Shawn. 2017. "Cacophony: Bad Algorithmic Music to Muse To." https://electricarchaeology.ca/

2017/02/03/cacophony-bad-algorithmic-music-to-muse-to/.

Graham, Shawn, Scott Weingart, and Ian Milligan. 2012. "Getting Started with Topic Modeling and MALLET." *Programming Historian*, September. http://programminghistorian.org/lessons/topic-modeling-and-mallet.

Huffer, Damien, and Shawn Graham. 2017. "The Insta-Dead: The Rhetoric of the Human Remains Trade on Instagram." *Internet Archaeology*. doi:https://doi.org/10.11141/ia.45.5.

Kansa, Eric C., Sarah Whitcher Kansa, and Ethan Watrall. 2011. *Archaeology 2.0: New Approaches to Communication and Collaboration*. Cotsen Digital Archaeology. http://escholarship.org/uc/item/1r6137tb.

Liu, Alan. 2004. *The Laws of Cool: Knowledge Work and the Culture of Information*. 1 edition. Chicago: University of Chicago Press.

Lucas, Gavin. 2012. *Understanding the Archaeological Record*. Cambridge University Press.

M. van Exel, S. Fruijtier, E. Dias. 2010. "The Impact of Crowdsourcing on Spatial Data Quality Indicators." In *Proceedings of Giscience 2011, Zurich, Switzerland, 14–17 September 2010*. https://www.researchgate.net/publication/267398729_The_impact_of_crowdsourcing_on_spatial_data_quality_indicators.

Mickel, Allison. 2016. "Tracing Teams, Texts, and Topics: Applying Social Network Analysis to Understand Archaeological Knowledge Production at Çatalhöyük." *Journal of Archaeological Method and Theory* 23 (4): 1095–1126. doi:10.1007/s10816-015-9261-z.

Montello, Daniel R., Sara Irina Fabrikant, Marco Ruocco, and Richard S. Middleton. 2003. "Testing the First Law of Cognitive Geography on Point-Display Spatializations." In *International Conference on Spatial Information Theory*, 316–31. Springer. http://link.springer.com/chapter/10.1007/978-3-540-39923-0_21.

Mullen, Lincoln. 2017. "A Confirmation of Andrew Goldstone on 'Teaching Quantitative Methods'." *The Backward Glance*. http://lincolnmullen.com/blog/a-confirmation-of-andrew-goldstone-on-teaching-quantitative-methods/.

Neil Brodie, Colin Renfrew, Jennifer Doole, ed. 2001. *Trade in Illicit Antiquities: The Destruction of the World's Archaeological Heritage*. Cambridge: McDonald Institute for Archaeological Research.

Ramsay, Stephen. 2011. *Reading Machines: Toward an Algorithmic Criticism*. 1st Edition edition. Urbana: University of Illinois Press.

Samuels, Lisa, and Jerome J. McGann. 1999. "Deformance and Interpretation." *New Literary History* 30 (1): 25–56. doi:10.1353/nlh.1999.0010.

Shawn Graham. 2014. "A Digital Archaeology of Digital Archaeology: Work in Progress." https://electricarchaeology.ca/2014/11/06/a-digital-archaeology-of-digital-archaeology-work-in-progress/.

tjowens. 2012. "Discovery and Justification Are Different: Notes on Science-Ing the Humanities." *Trevor Owens*. http://www.trevorowens.org/2012/11/discovery-and-justification-are-different-notes-on-sciencing-the-humanities/.

Wang, Haiyan, Zhongshi He, Yongwen Huang, Dingding Chen, and Zexun Zhou. 2017. "Bodhisattva Head Images Modeling Style Recognition of Dazu Rock Carvings Based on Deep Convolutional Network." *Journal of Cultural Heritage* 27: 60–71. doi:https://doi.org/10.1016/j.culher.2017.03.006.