The Open Digital Archaeology Textbook

Shawn Graham, Neha Gupta, Jolene Smith, Andreas Angourakis, Andrew Reinhard, Lorna Richardson, Kate Ellenberger, Zack Batist, Joel Rivard, Ben Marwick, Michael Carter, & Beth Compton

2018-08-20

Contents

notice

This volume goes hand-in-glove with Jupyter Notebooks coupled with the Binder service as a way of serving and running interactive computational environments online via 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 list of notebooks and hit the `launch binder` button. To examine or download the code, click the `repository` links instead.

As you read this text and explore the notebooks, feel free to 'fork' (copy) the source text files and to reuse or remix them for your own use. The source files are text files written with minimal markup (headers, images, bold, italics, and so on, in markdown format) that may be opened in any word processor or text editor; the Pandoc program can convert the markdown into Word or other formats if you prefer.

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

Citation Information: A permanent DOI courtesy of zenodo.org will be provided September 2018.

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.

Jolene Smith

Jolene Smith is an archaeologist who manages statewide digital archives and archaeological data at the Virginia Department of Historic Resources. She is interested in open data, preservation, and developing

easy-to-use tools to help small organizations create, use, and manage archaeological data.

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.

Andrew Reinhard

Andrew Reinhard literally wrote the book on video game archaeology: Archaeogaming: An Introduction to Archaeology in and of Video Games (Berghahn Books 2018). He led the team of archaeologists that excavated the Atari Burial ground in Alamogordo, New Mexico (April 2014) as well as the first in-game archaeological expedition via No Man's Sky (Hello Games 2016). He is a PhD candidate at the University of York's (UK) Department of Archaeology where his thesis focuses on evaluating tools and methods for archaeological investigation of digital spaces.

Lorna Richardson

Kate Ellenberger

Zack Batist

Zack Batist is a PhD student at the University of Toronto's Faculty of Information, where his research focuses on the ways in which archaeological practices – in their immense variety and scope – are implemented as part of a broad continuum of work that is inherently interdisciplinary and cooperative in nature. He is also interested in the critical evaluation of open data initiatives and the development of information infrastructures across and among disciplines, as well as public perceptions of archaeology and the representation of archaeology in science fiction, popular media, professional politics and public policy. He also actively tinkers with various computational tools and methods, ranging across the general domains of database management, GIS, statistical computing, network analysis, web development and server administration. Some ramblings and code can be found on github and twitter.

Joel Rivard

Joël Rivard is the replacement GIS and Geography Librarian at the University of Ottawa. Before taking this position in the summer of 2017, Joël spent over 15 years working as the Cartographic Specialist and the GIS Technician at the Carleton University Library. He has a Master of Information Studies as well as an honours degree in Geomatics with a minor in Environmental Studies.

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.

Our hope for this volume is that it will provide you with the ability to work out what you need to know, what you don't know, and how to creatively use the computational power of the devices available to you, to do better archaeology. Once you've outgrown this text, we strongly recommend that you visit the peer-reviewed collection of tutorials at The Programming Historian (which are available in English and Spanish and soon other languages). If you are more interested in the statistical side of digital archaeology, then you should consult Ben Marwick et al How to Do Archaeological Science Using R.

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 are running in the binder environment and *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 our list of notebooks 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, and includes Jupyter notebooks and RStudio by default. 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 they do 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 (for instance, archaeological GIS is well-covered by many excellent texts and online tutorials already - but you *will* find discussions of web mapping here). 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. Suggest - or even better, *write!* - new text or new computational notebooks that could expand the usefulness of this work. When we started this project, there were only four of us; by the time we reached version 1, we were already twelve strong. We would consider this project a success if it can turn its readers into its *writers*.

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

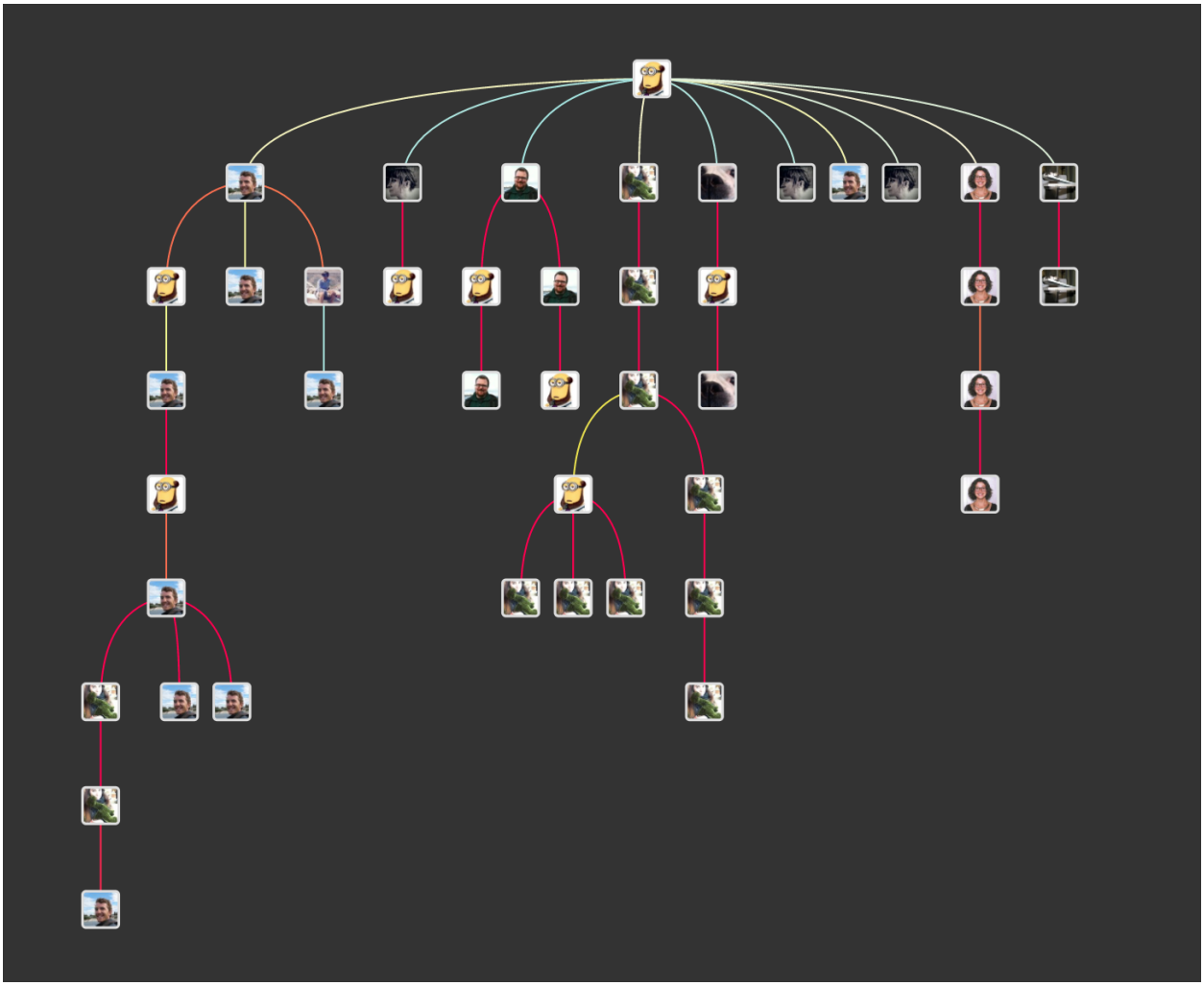


Figure 1.1: A visualization of the conversation tree on Twitter from Graham's query

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?

Huggett, P. Reilly, and Lock (2018) reviewed the landscape of 'digital archaeology', and identified an 'anxiety discourse' that surrounded this thing, 'digital archaeology'. The anxiety around the thoughtful use of computation in archaeology involved the simple - what do we *call* this thing? - to the profound - the effect on archaeology of the use of 'borrowed' technologies:

> This anxiety discourse arose from frustration on three levels. First, digital technologies are transforming the practice of archaeology yet digital archaeologists are frequently seen as little more than technicians or technologists. Secondly, the significance of the transformation of archaeological practice by the digital was largely unrecognised or else taken for granted – archaeologists are simply consumers of the digital like everyone else and so powerless in the face of its inevitability. Thirdly, archaeologists were uncritically accepting of the new tools and opportunities that the digital offered

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 feel a similar frustration and wonder why we feel it necessary to even ask the question. It is important at the outset to state that we do not regard digital archaeology as '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. Cook and Compton (2018), in their survey of digital archaeology as practised in Canada develop a big-tent perspective that perhaps puts it best:

> ....digital technologies are indeed tools, but they are not neutral or passive and therefore the technological ecosystem within which archaeology functions must be connected to broader paradigmatic shifts. Consequently, there is a need for specialisation and focus to fully understand and take advantage of the complexisties of technology, and yet it is so universal that all arcaheologists must take more responsibility for their digital data, analysis, and communications.

All archaeologies are digital, but not all archaeologies are Digital Archaeology.

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. Huggett, P. Reilly, and Lock (2018) develop a typology for understanding the various scenarios under which 'digital archaeology' can be understood to exist. Examine that text, and ask yourself where *this project, ODATE* fits? In terms of 'knowledge practices', where does it lie on the 'established & restricted' versus 'experimental & ubiquitous' continuum? Where does it lie on the 'digital technologies' continuum: from 'open & unfettered' to 'traditional & tethered'? Where does *your* encounter with digital archaeology sit?

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
```

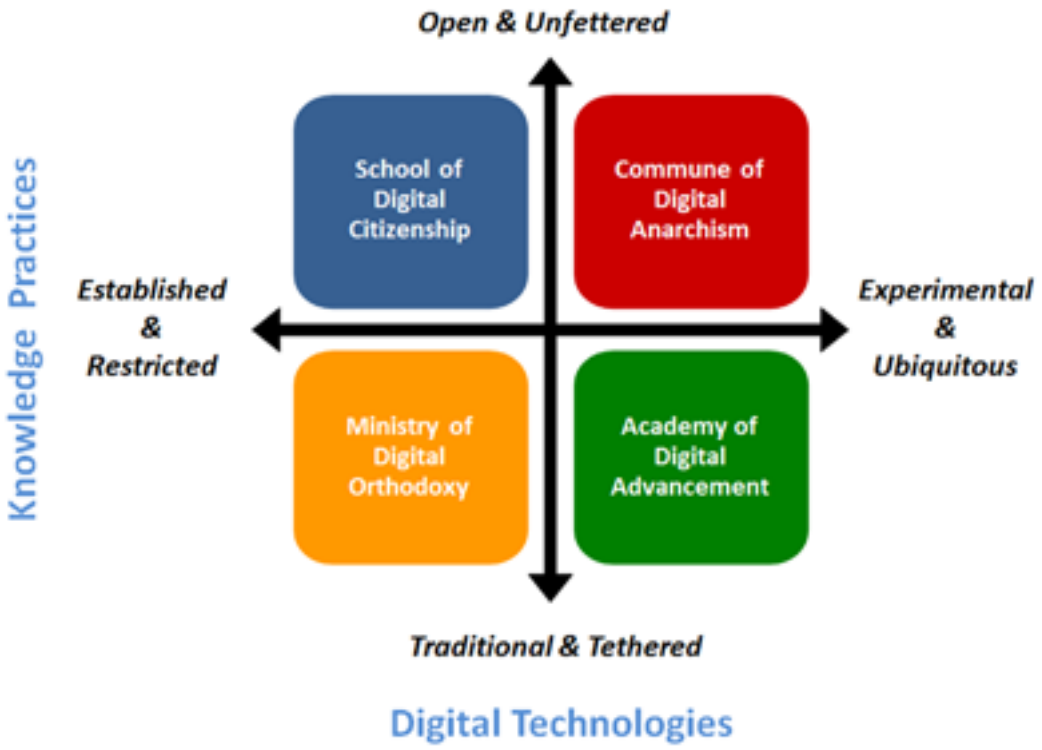


Figure 1.2: Hugget & Locke, 2018, Digital Archaeology and Knowledge Practice Scenarios

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

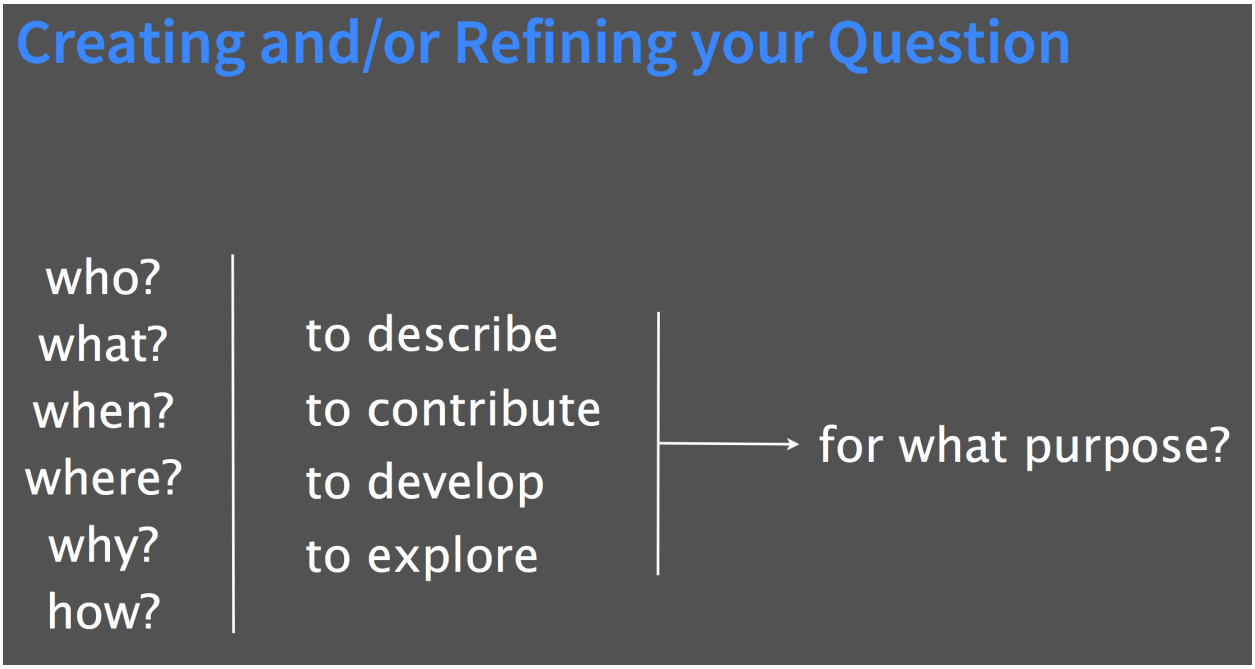


Figure 1.3: Creating and/or refining your research question, per DevDH.org

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 1.4: 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 1.5: 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

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:

```
# what I wanted to do
  - include links to any websites or articles you were reading
    # what I did
```

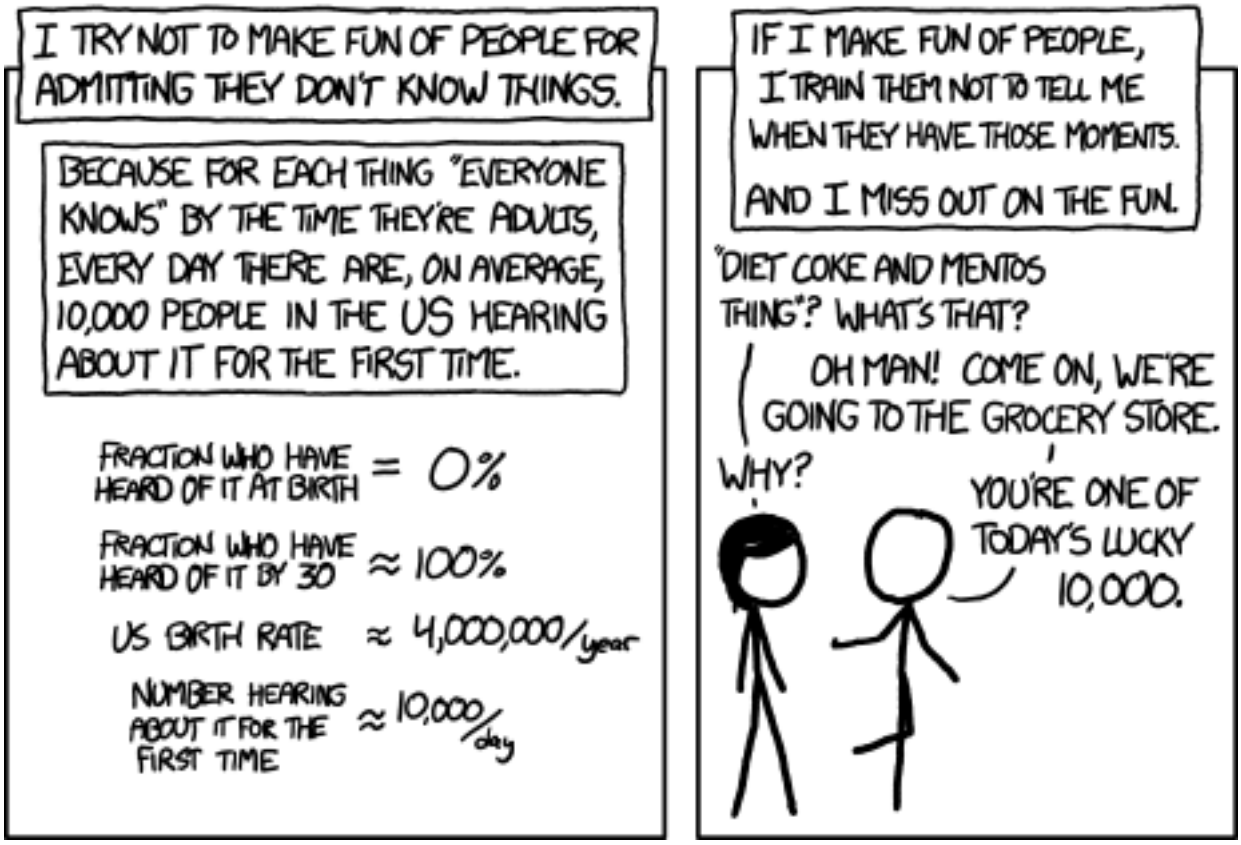


Figure 1.6: XKCD comic 'Ten Thousand'

```
- include the history of commands you've typed by passing your command line history to a new text file
  # 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 the basic 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 (with due consideration to your own privacy concerns) 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'. 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 (Graham, Massie, and Feuerherm 2013). 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:

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 averring 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, as in the Global South, 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

(For more concerning digital things & what they do, see Morgan (2012).)

What's more, *programming is forgetting* (Parrish 2016):

> The process of computer programming is taking the world, which is infinitely variable, mysterious, and unknowable (if you'll excuse a little turn towards the woo in this talk) and turning it into procedures and data. And we have a number of different names for this process: scanning, sampling, digitizing, transcribing, schematizing, programming. But the result is the same. The world, which consists of analog phenomena infinite and unknowable, is reduced to the repeatable and the discrete.
>
> [. . .] In the process of programming, or scanning or sampling or digitizing or transcribing, much of the world is left out or forgotten.

What do we *leave out* when we reduce the messiness of the world to our schemas, our rasters, our compressed images? The decision of what to leave out is always a decision, even if it flows as a consequence of whatever method we're using. In this way, the act of forgetting - even in an era of big data when seemingly everything that *can be recorded is recorded* is still a fundamentally *ethical* and *moral* one. It is not however a recognition that should make us despair. What should we do? In her discussion of the ethics of trying to represent and

interfere with digital representations of the world, Parrish arrives at an ethics that is not so much proscriptive as inquisitive:

- *. . . we should ask "Who gets to use what I make? Who am I leaving out? How does what I make facilitate or hinder access?"*
- *. . . we could ask "What data am I using? Whose labor produced it and what biases and assumptions are built into it? Why choose this particular phenomenon for digitization or transcription? And what do the data leave out?"*
- *. . . we should ask "What systems of authority am I enacting through what I make? What systems of support do I rely on? How does what I make support other people?"*
- *. . . we should ask "What kind of community am I assuming? What community do I invite through what I make? How are my own personal values reflected in what I make?"*
- *. . . You can create art and beauty on a computer.*

(Parrish, 2016)

1.6.1 exercises

- Find the websites for six archaeological projects. Using Parrish's questioning framework, develop a matrix (table) to compare your answers. What commonalities unite the 'most' ethical, by these lights? What might be some unintended consequences for the 'least' ethical projects?
- Use the same framework to evaluate your own online digital presence. You might find the tools and issues discussed in Kelly (n.d.) to be useful in this regard.

1.7 The Human Problem

A key to success in using, remixing, and sharing data openly is communicating effectively with colleagues and other stakeholders. Whether you are utilizing data that already existed or creating a totally new dataset, your work impacts colleagues, and it is important to build not only a technical framework but social supports for your work. A successful digital archaeology project is one which is well-understood and well-regarded by colleagues. In the following section I will describe how to frame digital archaeology work for colleagues and others who may be interested or impacted by it, and share three key tools for establishing boundaries and expectations for your project.

Outline

1. Colleagues

- Can be difficult to move forward when data sharing is not necessarily the norm, and there are not norms around how to do it
- Establish boundaries regarding what you are doing
- Explain why what you are doing is part of, rather than separate from, previous meaningful work in your research realm
- People are hesitant to let the data they spent so much time and effort and money creating be used in ways they have no control over
- Key is being clear about your intentions and the roles of each person in the exchange
- Even if you are creating data "From scratch" there are colleagues whose work will be impacted by your contribution, so thinking about how your work interfaces with theirs will allow you to more carefully plan so that your work is seen as a positive contribution to the literature rather than disruptive in a negative way, or even worse, useless

- The sheer volume of data produced by archaeologists and the time it takes to code and manage our collections leads us to be slow to adopt new technologies, even if we are on board with them. Consider this and do not write people off for being slow to agree to your innovative technology-based ideas.
- Address the importance of this planning for early career and contingent researchers and students in particular

2. Stakeholders Beyond Archaeology

- Example: Cemetery project
- Sometimes stakeholders are individuals, sometimes they are institutions or vague groups/entities; address who to choose and the stakes of deciding this
- The power of informal working groups for getting feedback and direction
- Managing expectations
- Sharing outputs and picking appropriate formats

3. Build a Plan for Yourself
4. Make Your Case

- The Group Meeting
- The Strategy Document
- The Policy Document

Chapter 2

Making Data Useful

Elsewhere in this text we've covered approaches to creating digital archaeology data. Section 1.1 discusses three 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 when trying to use someone else's data or reconstruct their conclusions from field notes and excavation reports. What makes a dataset unwieldy? How can we make it better?

It's not understandable. The dataset may be illegible (poor handwriting, or a poor 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).

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 (a picture of a table in a PDF), when in fact it's got a structure (easily extractable text and numbers). The data may already be processed in ways that make further analysis impossible.

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. And maintaining data in this condition helps protect them from the ravages of time.

Making Data Sensible: Pre-Planning

Creating good data requires planning. It is important to plan the structure of the data, expectations for its use by researchers, and arrange for its storage from the beginning of your research. A reality to confront,

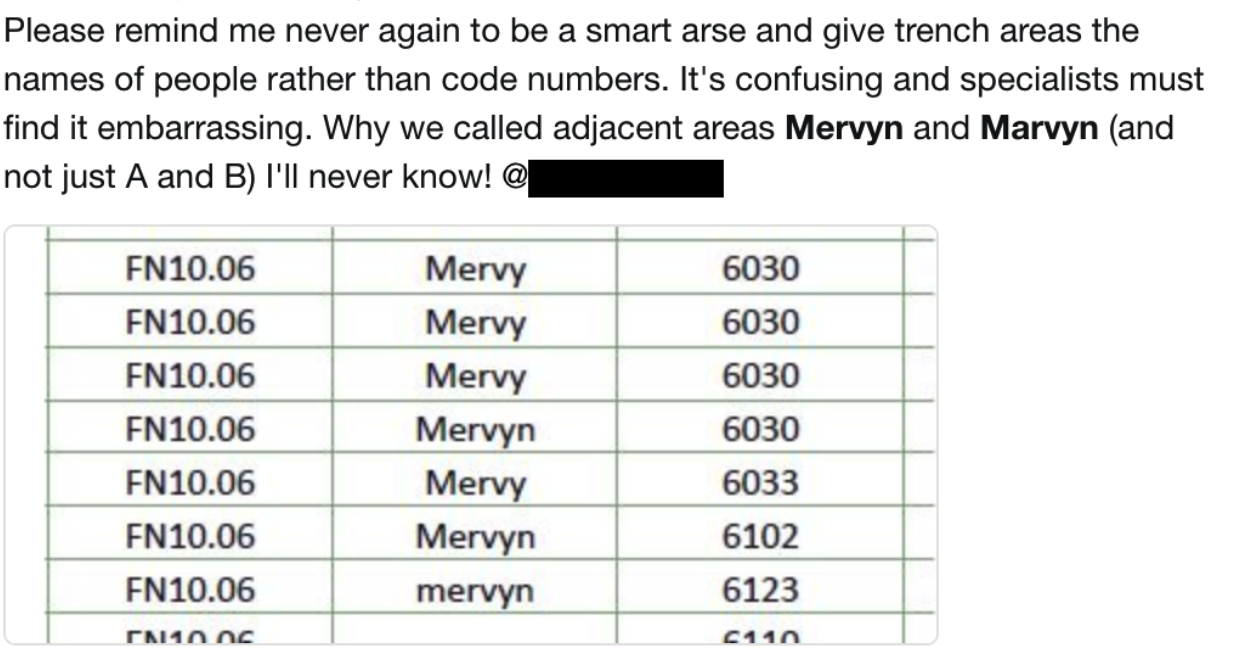


Figure 2.1: A Useful Warning from Kenny Brophy

however, is that there will be tradeoffs.

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? | Data Computational environment Comment

Degree of reproducibility	Data	Analysis	Computational environment	Comment
Not reproducible	Summary statistics of the raw data are presented.	Brief narrative of methods is presented.	No information is provided.	The current status quo for scholarly journal articles.

Degree of reproducibility	Data	Analysis	Computational environment	Comment
Low reproducibility	The reader is invited to contact the author for access to the data.	Brief narrative of methods is presented, names and version numbers of software are stated.	No information is provided.	Frequently seen. Inviting readers to contact the author to access the raw data is no guarantee that the raw data is available.

Degree of reproducibility	Data	Analysis	Computational environment	Comment
Moderate reproducibility	The journal article is accompanied by files of raw data tables in PDF or Excel (i.e., binary) files.	Brief narrative of methods is presented, names and version numbers of software are stated.	No information is provided.	Frequently seen. Having the raw data in supplementary material makes it much more accessible compared to when it must be requested from the author. However, extracting raw data from a PDF or other binary file format can be time-consuming and introduce

Degree of reproducibility	Data	Analysis	Computational environment	Comment
High reproducibility	The journal article is accompanied by plain text files (e.g., CSV format) of raw data.	The journal article is accompanied by script files of R of Python code that demonstrate key parts of the analysis (but do not generate all the results presented in the paper).	No information is provided.	Uncommon. Raw data in plain text format makes reuse highly efficient. Script files with code provide valuable insights into analytical decisions that are not narrated in the text of the article. However, because the code is not complete, substantial effort and skill

Degree of reproducibility	Data	Analysis	Computational environment	Comment
Very high reproducibility	The journal article includes DOIs to an open access repository that contains plain text files (e.g., CSV format) of raw data.	The open access repository linked to from the paper includes version-controlled R package or script files of R or Python code to reproduce all of the analysis output and graphics in the article.	The open access repository linked to from the paper includes a dockerfile that documents the computational environment of the published analysis, and a docker image that allows another person to use that environment.	Currently rarely seen. Other researchers should have a good chance to reproduce, reuse and extend the published results with this combination of plain text data files, code that documents every analysis and visualization in the paper, and details of the computational

Degree of reproducibility	Data	Analysis	Computational environment	Comment

Table reproduced from Marwick (2016).

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. in order to get as close to a perfectly usable and preservable dataset, you should collect both metadata and paradata.

If the concept of metadata is *data about data*, defining attributes and documenting structure, paradata is *data alongside data*, like a data diary (Denard and others 2009). You can see paradata in action at events like the Heritage Jam. The concept can be applied broadly to analytical and technical digital projects to great effect. 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 (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."

Metadata is likewise a critical concept. Some values generated by a machine, like file size or photographic EXIF data, while other *descriptive metadata* is authored by hand. Think of it as the codex that makes the rest of your information useful.

Making Data Durable: Preservation and Longevity

With pre-planning and thought directed at topics like format, structure, and descriptions, you can do a lot to extend the life of your work. From the outset, consider whether you intend to prepare your data for ultimate disposition in a digital repository, maintain it yourself, or both. In the discipline of digital preservation, the LOCKSS (Lots of Copies Keep Stuff Safe) principle recommends storing data in several locations to ensure redundancy.

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

For a deeper dive into digital preservation, see Trevor Owens, *The Theory and Craft of Digital Preservation,* Johns Hopkins University Press, 2018 (full open access preprint available).

Time is your enemy

Archaeologists are well aware that excavation is a destructive act. Conservation is never forever. When we dig, we participate in a process of rearranging physical material in ways that make it both easier to interpret and more vulnerable to inevitable degradation. In a way, the same can be said about archaeological data. Here are some common risks:

Deprecated formats:

After we hit the Save button, it's easy to take for granted that our digital work is safe and sound. But consider file formats. Many of us working with legacy collections and data have come up against old word processing files, databases, or geospatial datasets that just won't open anymore. These are generally binary files [define] in proprietary formats that have since become obsolete. A way to guard against this is to use lossless formats when possible and to regularly convert files to keep up with software versions.

Bit rot and link rot:

Some files can themselves deteriorate over time, especially when copied or compressed repeatedly. Links on websites can become broken or "dead" as resources are moved or reconfigured. Defining a migration schedule

Figure 2.2: An archival box full of legacy media in the section author's workspace, waiting to be transferred to stable formats.

for files to newer versions can help guard against bit rot (don't just save your files somewhere and expect them to work years later). One way to check on file integrity is to periodically generate *checksums* when you store a file. A checksum is long number calculated based on the contents of the bits in the file itself. For example, even a tiny glitch in the pixels of an image will cause changes to the checksum, so a mismatched value will tell you something went wrong with your file. Link rot can be mitigated by linking to archived sources (the Internet Archive's Wayback Machine, for example) or DOIs.

Lost institutional knowledge and unconscious assumptions: What did those codes mean? What unit of measurement is this? How was the equipment calibrated? What were the coordinates of the site datum again? These are critical pieces of information that can all too easily become lost and make related data difficult or impossible to reuse.

Strategies: Have a Plan

Good digital preservation takes some thought in advance. Describe what you're trying to preserve, and to what level. Are you preserving supporting data (tables, images, other files), entire documents (reports, publications, theses), or digital projects (websites, interactive content)? All of these types of information will require different consideration for preservation. Are you planing to curate these digital materials yourself? Then research personal digital archiving. You may also investigate options to find a suitable repository in order to tap into existing infrastructure and expertise. This repository might be part of a university library ecosystem, it might be something specifically designed for archaeological data like Open Context or tDAR, or it could be very broadly defined, like Zenodo. Your choice of repository may dictate the kind of metadata you collect and the way you collect it, so the sooner in your project lifecycle you consider long-term storage and preservation options, the better.

Strategies: Create Metadata

Create thorough metadata as you go. This will help you, those who use your data in the future, and those who are tasked with curating your datasets.

DH projects: "Preservation for the (Digital) Ages: Digital Archivists from Collaborate with Classicists to Improve Database Preservation Methods." ScienceDaily, https://www.sciencedaily.com/releases/2017/10/171017124345.htm. Accessed 7 Feb. 2018.

Xu, Weijia, et al. A Portable Strategy for Preserving Web Applications Functionality. IEEE, 2017, pp. 1–4. CrossRef, doi:10.1109/JCDL.2017.7991578. [Jolene can't access this]

Wordpress to static site

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) writes, 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)

Creating a structure

Choosing legacy_formats

Where to put your data

2.1.1 discussion

2.1.2 exercises

2.2 Cleaning Data

Gathering from many datasets

Define well-formed data

Watch out for many kinds of information in one field Splitting them up

Housekeeping- dates, special characters, encoding, leading/trailing spaces

2.2.1 discussion

2.2.2 exercises

2.3 Arranging and Storing Data for the Long Haul (Databases!)

Once you've collected data according to your plan and structure, you'll need a way to connect it all together in a way that allows you and other researchers to use it, hopefully for some time to come.

So what *is* a database and which one should you use?

Choosing a format: A lot of people have a lot of opinions on good database platforms, horrible ones, and everything in between. But there is no universally right answer and a lot depends on technical resources and level of experience. That is to say, in many cases, a relational database is the strongest option, but handling information well in spreadsheets may met your needs and the needs of other researchers if implemented well. Our organizations may dictate software platforms we're permitted to use, and often that's Microsoft Access or Excel.

Flat tables Excel Pros Cons

CSV or Tab-seperated text Pros Cons

Relational databases Desktop products

Server-based SQL Databases

2.3.1 discussion

2.3.2 exercises

Data Organization in Spreadsheets for Social Scientists

http://www.datacarpentry.org/spreadsheets-socialsci/

SQL for Social Science Data http://www.datacarpentry.org/sql-socialsci/

Cleaning Data with Open Refine

2.4 Using Application Programming Interfaces (APIS) to Retrieve Data

Sometimes it is useful for one program on *this* computer to be able to speak to another program on *that* computer, via the web. This is achieved through the use of 'application programming interfaces' (API). One example you may have come across is sharing a photo from your phone to some kind of social media website - the program on your phone communicates to the site (say, Twitter) and makes that post for you. The API can be thought of as a series of commands or even a kind of language that allows programmatic access to the *other* computer- perhaps a way of specifying how to download the results of an image search, or to retrieve every record in an archaeological database that fits a particular pattern in a format that you can then analyze.

One excellent archaeological database that uses an API to open its records is Open Context. Open Context publishes archaeological data after a rigorous editorial process. Its API is designed to allow faceted search. That is to say, it provides a way to summarize data so that we can understand aggregate patterns in the data. It also provides a path for finding individual records. Like most APIs, it provides the data in a JSON format to allow further computational work. JSON is human-readable text file. JSON organizes data in attribute-value pairs, or lists of such pairs. (Contrast that with the CSV or Excel files you are probably used to, where each row is an entity and each column is an attribute of that entity).

Many websites that have data of interest to archaeologists do not have an API. For many use cases, it can often be sufficient to expose data as simple downloads of table values. An API is valuable however in that it can be used as part of a workflow, updating data as it comes in, for some other application or website. An API can be very useful in cases where multiple persons are updating or contributing data. If the amount of data in the database is very large, an API might be the only practical way of opening the data to the world.

In the Jupyter notebooks, we show you some code for retrieving data from a few different APIs. The first is the 'Chronicling America' website, a repository of historical newspapers from across the United States kept by the Library of Congress. We developed our code for accessing the API by repurposing code that Tim Sherrat used for querying the Trove API from the National Library of Australia (which incidentally underlines again the value of sharing code!). Our version is in this repo. If you study that code, you can see how we built the query in python.

We -

- defined the location of where the APIs 'endpoint' is. That is to say, the URL where we can pass commands to the API;
- we identified one of the parameters that the API uses, `proxtext`, and assigned our search value to it
- we also specified what format to return the results in (eg, json)
- and then we told the python module 'imports' the complete URL that specifies not just the endpoint, but also all of the paramters for our search. The module 'imports' handles the actual getting of information for us, so we don't need to program that from scratch.

The remainder of the code gets the data and puts it into a variable that we can either examine or save to file.

APIs can appear intimidating at first. This is partly because many developers use automatic tools to generate the documentation for their APIs in the first place! The result is a dense soup of terms and jargon that is largely impenetrable to the rest of us. One of the nice things about the Open Context API and its developers is that they also provide examples of how to use the API and some common tasks one might want to do

using it. An example of the kind of scholarship that is enabled when we have large scale data repositories available for programmatic querying is Anderson et al. (2017).

2.4.1 Exercises

Launch the jupyter binder.

1. Open the 'Chronicling America API' notebook. Run through its various steps so that you end up with a json file of results. Imagine that you are writing a paper on the public reception of archaeology in the 19th century in the United States. Alter the notebook so that you can find primary source material for your study. **Going further** Find another API for historical newspapers somewhere else in the world. Duplicate the notebook, and alter it to search this other API so that you can have material for a cross-cultural comparison.
2. Open the 'Open Context API'. Notice how similar it is to the first notebook! Run through the steps so that you can see it work. Study the Open Context API documentation. Modify the search to return materials from a particular project or site.
3. The final notebook, 'Open Context Measurements', is a much more complicated series of calls to the Open Context API (courtesy of Eric Kansa). In this notebook, we are searching for zoological data held in Open Context, using standardised vocabularies from that field that described faunal remains. Examine the code carefully - do you see a series of nested 'if' statements? Remember that data is often described using JSON attribute:value pairs. These can be nested within one another, like Russian dolls. This series of 'if' statements parses the data into these nested levels, looking for the faunal information. Open Context is using an *ontology* or formal description of categorization of the data (which you can see here) that enables inter-operability with various Linked Open Data schemes. Run each section of the code. Do you see the section that defines how to make a plot? This code is called on later in the notebook, enabling us to plot the counts of the different kinds of faunal data. Try plotting different categories.
4. The notebooks above were written in Python. We can also interact with APIs using the R statistical programming language. The Portable Antiquities Scheme database also has an API. Launch this binder and open the 'Retrieving Data from the Portable Antiquities Scheme Database' notebook (courtesy of Daniel Pett). This notebook is in two parts. The first frames a query and then writes the result to a csv file for you. Work out how to make the query search for medieval materials, and write a csv to keep more of the data fields.
5. The second part of the notebook that interacts with the Portable Antiquities Scheme database uses that csv file to determine where the images for each item are located on the Scheme's servers, and to download them. You might find this useful for building an image classifier as described in 4.2.

Going further - the Programming Historian has a lesson on creating a web API. Follow that lesson and build a web api that serves some archaeological data that you've created or have access to. One idea might be to extend the Digital Atlas of Egyptian Archaeology, a gazetteer created by Anthropology undergraduates at Michigan State University. The source data may be found here.

2.5 Linked Open Data and Data Publishing

Digital archaeology is wonderfully messy, and it pulls in many different ways. This messiness can prompt a variety of responses. Huggett, P. Reilly, and Lock (2018) try to put a framework around this messiness so that we, as a field, can understand where we're going and work out where we might like to be. One element they draw attention to is the use of tools from other, non-archaeological, sources and the unintended side-effects that such tool use can have:

> *what must we do in order to know when we know something new?* Without this fundamental capability there is little possibility of locating the so-called 'something new' and determining whether it is really valuable, or merely useful, or have the opportunity to critique, shape or

> incorporate that 'new knowledge' to the benefit of the broader community. Secondly, *how do we identify gaps and obstacles to accessing our total potential pool of knowledge and capabilities?* [...]
>
> Clearly then, this bricolage of different institutional, private and individual knowledge bases makes it very difficult to see the gaps and to know 'when we know something new', whether it be new data, a discovery, a novel approach, an insight or finding and, especially, where, or with whom, this new or potential knowledge resides. This suggests the need for some level of knowledge brokerage [...]

The Linked Ancient World Data Institute is one answer to this problem of data brokerage. Over a series of meetings (and through the work of the Pelagios Commons) the participants developed approaches to using the principles of Linked Open Data to 'fill in the gaps' in establed ontologies (or descriptions of data) for use with Greco-Roman antiquity.

What does that mean, and how does that work? While many databases, services, or museums might expose their data via a web API, there can be limitations. Matthew Lincoln has an excellent tutorial at The Programming Historian that walks us through some of these differences, but the key one is in the way the data is represented. When data is described using a 'Resource Description Framework', RDF, the resource - the 'thing'- is described via a series of relationships, rather than as rows in a table or keys having values. In this approach, 'things' and 'concepts' are linked to unique descriptions of identifiers. In this way, data that is housed in one location can be integrated with data in another on the basis of semantic meanings.

Information is in the relationships. It's a network. It's a *graph*. Thus, every 'thing' in this graph can have its own *uniform resource identifier* (URI) that lives as a location on the internet. Information can then be created by making *statements* that use these URIs, similarly to how English grammar creates meaning: subject verb object. Or, in RDF-speak, 'subject predicate object', also known as a *triple*. In this way, data in *different* places can be linked together by referencing the elements they have in common. This is Linked Open Data (LOD). The access point for interrogating LOD is called an 'endpoint'.

To ask questions of the data, we write queries of the host computer which we access at the endpoit. A common language for writing such queries is *SPARQL* which stands for for SPARQL Protocol and RDF Query Language (yes, it's one of *those* kinds of acronyms).

In the notebook for this section, we're not using Python or R directly. Instead, we've set up a 'kernel' (think of that as the 'engine' for the notebook) that already includes everything necessary to set up and run SPARQL queries. (For reference, the kernel code is here). Both R and Python can interact with and query endpoints, and manipulate linked open data, but for the sake of learning a bit of what one can do with SPARQL, this notebook keeps all of that ancillary code tucked away. The followup notebook shows you how to use R to do some basic manipulations of the query results.

2.5.1 exercises

1. Walk through the notebook to get the hang of writing simple queries of the various endpoints that it is configured to use. Then, modify the existing queries to retrieve answers to your own questions. What are some of the hidden 'gotchas' that you encounter, and what else do you need to know in order to make efficient use of the resources that *do* exist for linked open archaeological data?
2. Ethan Gruber is a leading authority on using linked open data for archaeological work. In this post he explores ResearchSpace from the British Museum. Consider his criticisms and explore ResearchSpace for yourself.

2.6 Scraping Data

We have discussed a variety of ways that we can obtain and manipulate archaeological data. These have depended on the owners or controllers of that data to make positive choices in how that data is structured

and exposed to the web. But often, we will encounter situations in which we want to use data that has been made available in ways that don't allow for easy extraction. The Atlas of Roman Pottery for instance is a valuable resource - but what if you wanted to remix it with other data sources? What if you wanted to create a handy offline reference chart for just the particular wares likely to be found on your site? It's not immediately clear, from that website, that such a use is permitted, but it's easy to imagine someone doing such a thing. After all, it is not uncommon on a site to have well-thumbed stacks of photocopies of various reference works on hand (which also likely is a copyright violation).

In this gray area we find the techniques of web scraping. Every web page you visit is downloaded locally to your browser; it is your browser that renders the information so that we can read the page. Scraping involves freeing data of interest from the browser into a format that is useable for your particular problem (often to a txt or csv file). This can be done manually via copying and pasting, but automated techniques are more thorough, less error-prone, and more complete. There are plugins for browsers that will semi-automate this for you; one commonly used plugin for Chrome is Scraper. When you have Scraper installed, you can right-click on the information you're interested, and select 'scrape similar'. Scraper is examining the underlying code that generates the website, and is extracting all of the information that appears in a similar location in the 'document object model'. An excellent walkthrough of the technical and legal issues involved in using Scraper is covered in this Library Carpentry workshop, and is well worth your time. (Another excellent tutorial to writing your own scraper that works in a similar vein (identifying patterns in how the site is built, and pulling the data that sits inside those patterns) is available on the Programming Historian).

Other times, we wish to reuse or incorporate the published data from an archaeological study in order to complement or contrast with our own work. For instance, in Lane, K., Pomeroy, E. & Davila, M. (2018). Over Rock and Under Stone: Carved Rocks and Subterranean Burials at Kipia, Ancash, AD 1000 – 1532. Open Archaeology, 4(1), pp. 299-321. doi:10.1515/opar-2018-0018 there are a number of tables of data. Their Table 1 for instance is a list of calibrated dates. How can we integrate these into our own studies? In many journal pdfs, there is a hidden text layer over top of the image of the laid out and typeset page (pdfs made from journal articles from the mid 1990s or earlier are typically merely images and do not contain this hidden layer). If you can highlight the text, and copy and paste it, then you've grabbed this hidden layer. But let's do that with table one - this is what we end up with:

```
Lab Nō.MAMSSample Nō.SectorPitStratigraphic Unit[Cut]C14 Date±13C Cal 1 Sigmacal ADCal 2 Sigmacal AD158
```

Not entirely useful. If you paste directly into a spreadsheet, you end up with something even worse:

The answer in this case is a particular tool called 'Tabula'. This tool allows you to define the area on the page where the data of interest is located; it then tries to understand how the text is laid out (especially with regard to white space) and enable you to paste into a spreadsheet or document cleanly (assuming that the pdf has that hidden text layer). In the Jupyter Notebook for this section we have an R script that calls Tabula, and then passes the result into a dataframe enabling you to immediately begin to work with the data. To run this script, launch the binder and then click 'new -> RStudio session'. Within RStudio, click on 'file - open' and open 'Extracting data from pdfs using Tabulizr.R'. This script is commented out explaining what each line achieves; you run each line from top to bottom by pressing the 'run' button (or using the keyboard shortcut).

This script works well with tabular data; but what if we wanted to recover information or data from other kinds of charts or diagrams? This is where Daniel Noble's 'metaDigitise' package for R comes into play. This package enables you to bring an image of the graph or plot into rStudio, and then trace over things like scale bars and axes to enable the package to work out the actual values the original graph plotted! The results are then available for further manipulation or statistical analysis. Since so many archaeological studies do not publish their source data (perhaps these studies were not written at a time when reproducibility or replicability were as feasible as they are today) this tool is an important element of a digital archaeological approach to pulling data together. In the binder launched above (link here there is a script called 'metaDigitise-example.R'. Open that within rStudio and work through its steps to get a sense of what can be accomplished with it.

A full walk through of this package (which is conceived of as being useful for meta-analyses) see the original documentation. Another just-released package that is worth exploring is 'datapasta' by Miles McBain.

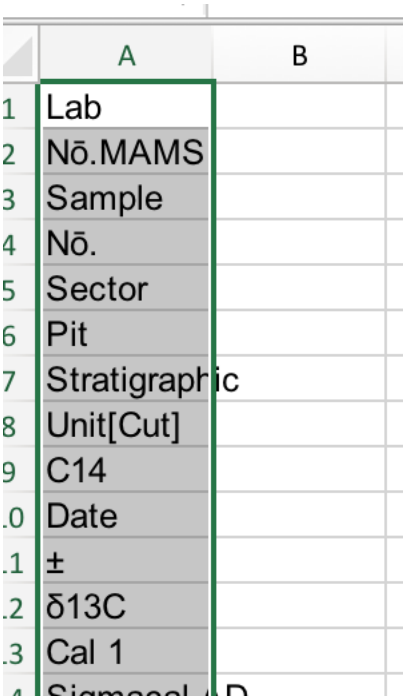


Figure 2.3: Pasting Lane et al table 1 into excel

Finally, there is the extremely powerful Scrapy framework for python that allows you to build a custom scraper for use on a wide variety of sites. We also include a walk through of the this tutorial that builds a scraper from scratch on Finds.org.uk data.

2.6.1 Exercises

1. Complete the Scrapy walkthrough. Then, try again from start on another museum's collection page (for instance, these search results from the Canadian Museum of History). What do you notice about the way the data is laid out? Where are the 'gotchas'? Examine the terms of service for both sites. What ethical or legal issues might scraping present? To what degree should museums be presenting data for both human *and* machines to read? Is there an argument for *not* doing this?
2. Read Christen (2012) and consider the issues of access to digital cultural heritage from this viewpoint. Where should digital archaeology intersect?

Chapter 3

Finding and Communicating the Compelling Story

Data never speaks for itself. In digital work, this problem is compounded by the kinds of stories that the *tools* want to tell. Many different kinds of content management systems are available to mount the materials online. Each one however has built in presets and categories (and *ways* of categorizing) that reveal how their makers imagined the world working. Even the division of content into 'posts' and 'pages' on the ubiquitious cms Wordpress is a division of information by priority - and search engines for instance take that division as a signal in how results are returned, for instance.

Imagine that you are working with a First Nation in Canada, designing an online exhibition concerning their cultural heritage. Imagine that you've been asked to design the exhibit in such a way as to allow different members of that community (or outsiders) different kinds of access - that different elements of the materials should be recombined in particular ways. Wordpress's simple pages and posts would clearly be not enough.

That is one aspect that should be considered in the public-facing work of communicating the compelling story. Given that so much of digital archaeology involves working on the open web, ethical concerns permeate everything we do. Who could this harm? That is the question that needs to be asked at every juncture. It might not be apparent that harm could result from our decisions - what harm is there in making our data analysis open with a jupyter notebook or shared code or dataset? Sometimes, the harm is in keeping the work closed; sometimes making the work open could allow it to be appropriated by bad-faith actors (witness the ways Classical scholarship is appropriated in the service of racism).

In this chapter, you will find a variety of tools and approaches for increasing the reach of, and engagement with, your scholarship. Choices have consequences. Who is hurt, who is helped, by yours?

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 Data Driven Documents

Given that a lot of archaeological data is represented in JSON format when we retrieve it from an API or from a database, it is worth your time to consider ways of representing this data natively on the web. By 'natively' we mean, as part of the document object model approach that underpins most webdesign. By making the data part of the way the site is generated (and manipulated), we free our data and our visualizations. This opens up the possibility of generating archaeological visualizations (and compelling narratives) on-the-fly as data is updated. It opens up the possibility of remixing the data, or of linking it into other data.

But as Mike Bostock (editor for interactive graphics at the New York Times and creator of D3) points out, there is a danger of getting lost in the woods of any one particular technology; we always need to keep in mind,

> The purpose of visualization is insight, not pictures.
>
> - Mike Bostock

In this section, we introduce two approaches to data driven documents: D3.js, and the Processing language.

3.2.1 D3

Let's make an interactive bar chart of pottery counts using D3.

> D3.js is a JavaScript library for manipulating documents based on data. D3 helps you bring data to life using HTML, SVG, and CSS. D3's emphasis on web standards gives you the full capabilities of modern browsers without tying yourself to a proprietary framework, combining powerful visualization components and a data-driven approach to DOM manipulation.

What this means is that D3 gives us some pre-packaged tools for taking our JSON file and tying elements of that data to different kinds of graphical elements in the html of a webpage, in a way that is relatively straightforward. Take a few moments to explore the example gallery which demonstrates - with the code - ways of achieving various kinds of visualizations (including dynamic and interactive ones).

To use D3.js, we have to specify three particular parts of the puzzle - the data, the styling, and the html that pulls it altogether. Within the html, we use SVG - 'scalable vector graphics' ie mathematical descriptions of the lines, rather than the coloured pixels of raster graphics - to describe how the deifferent elements of the visualization will be built. The *size* and *placement* of these elements depends on the aspects of the data.

One way of experimenting with D3 is to use Mike Bostock's bl.ocks.org service. This site shows off many different kinds of visualizations. To use it for yourself requires us to put our html, css, and json files somewhere where bl.ocks.org can find it - in this case, we will use a Github gist. A gist is part of the Github website that allows anyone to quickly post a code snippet online, with a url, for sharing. If we are logged in, then the gist ceases to be anonymous - for instance, Shawn's gists are all at gist.github.com/shawngraham. To run a gist through bl.ocks.org, we copy the URL path *after* gist.github.com and paste it _after bl.ocks.org - hence Shawn's gists could be visualized at `http://bl.ocks.org/shawngraham/<string-of-random-letters-and-numbers-identifying-the-exact-gist`. A gist can have more than one file.

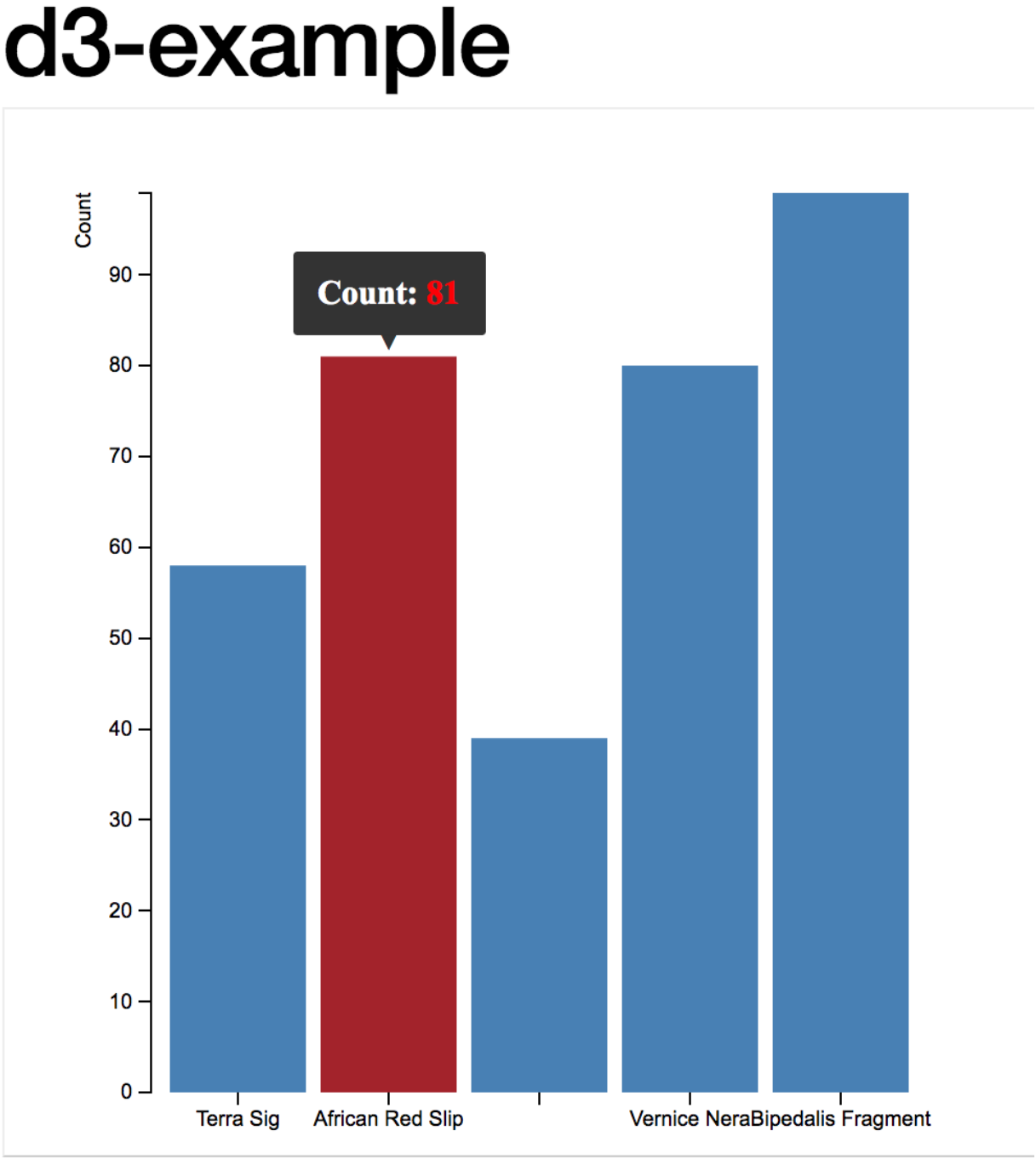


Figure 3.1: A screenshot of the final product

On your own webserver, you would have an `index.html` file, a `style.css` file, and a data file, either in csv, tsv, or in json. Let's build a simple bar chart with an interactive tool tip, that visualizes some pottery counts.

Our pottery counts data is represented by the following .json file:

```
[
  {"PotteryType": "Terra Sig","Count": 58},
  {"PotteryType": "African Red Slip","Count": 81},
  {"PotteryTyp": "Dressel24","Count": 39},
  {"PotteryType": "Vernice Nera","Count": 80},
  {"PotteryType": "Bipedalis Fragment","Count": 99}
]
```

The next step is to create the style sheet. The one we're going to use can be inspected here. A stylesheet describes how each element referred to in the html should be rendered. Can you work out which part of the stylesheet is governing the colour for our barchart? Which bit will take care of the axes?

The final step is to create the html that will pull everything together. Because we're going to use the tools in the D3 package to control how this all works, we have to tell this to the html. The opening of our HTML looks like this - see how we're telling the brower where our css is, and where to find the D3? The D3 *could* live in, for instance, a folder on our own server, but here we're going to use the most up-to-date version available at 3djs.org:

```
<!DOCTYPE html>
<meta charset="utf-8">
<head>
<link rel="stylesheet" type="text/css" href="barchart.css">
</head>
<body>
    <script src="https://d3js.org/d3.v3.min.js"></script>
     <script src="d3.tip.v0.6.3.js"></script>
<script>
```

Next, we're going to define a series of variables that will control or describe how we want the data to be displayed, and the kind of *interactivity* that our chart will have. The second script in the html code above calls a related package, 'd3.tip' by Justin Palmer. This is a small bundle that enables mouse-over tool-tips on the elements in our visualization. It's not loading it from somewhere else on the web, but rather from the same location where the index.html file can be found.

In the definition of the variables below, any that begin with `d3` are referring to the D3 package which was loaded by the `<script>`. For more on the wide variety of things D3 can do, we invite you to check out the D3 tutorials.

```
var margin = {top: 40, right: 20, bottom: 30, left: 70},
    width = 460 - margin.left - margin.right,
    height = 500 - margin.top - margin.bottom;

var x = d3.scale.ordinal()
    .rangeRoundBands([0, width], .1);

var y = d3.scale.linear()
    .range([height, 0]);

var xAxis = d3.svg.axis()
    .scale(x)
    .orient("bottom");

var yAxis = d3.svg.axis()
```

```
    .scale(y)
    .orient("left")
    .ticks(10, "");
```

This next variable contains the information telling the browser what to do if the user mouses-over one of the bars in our chart. When that happens, the browser is to turn the bar red and to display the value for that bar (which is held in d.Count - look at the JSON again. Do you see the 'Count' data? If you wanted to adapt this chart to show *weights* of the pottery, what might you have to change?)

```
var tip = d3.tip()
  .attr('class', 'd3-tip')
  .offset([-10, 0])
  .html(function(d) {
    return "<strong>Count:</strong> <span style='color:red'>" + d.Count + "</span>";
  })
var svg = d3.select("body").append("svg")
    .attr("width", width + margin.left + margin.right)
    .attr("height", height + margin.top + margin.bottom)
  .append("g")
    .attr("transform", "translate(" + margin.left + "," + margin.top + ")");

svg.call(tip);
```

Now we add the actual data:

```
d3.json("pottery.json", function(error, data) {

  if (error) throw error;

  x.domain(data.map(function(d) { return d.PotteryType; }));
  y.domain([0, d3.max(data, function(d) { return d.Count; })]);
```

We just tell it that the json data is in the `pottery.json` file. If that json file was on the web somewhere, we could just provide the url (as long as it ends in a json file!). Then we map the data in the json file to our x and y axes - PotteryType and Count. This mapping pushes that data to `d.PotteryType` and `d.Count`; anytime we need that data, we can now just refer to those variables.

We finish up by pulling it all together:

```
  svg.append("g")
      .attr("class", "x axis")
      .attr("transform", "translate(0," + height + ")")
      .call(xAxis);

  svg.append("g")
      .attr("class", "y axis")
      .call(yAxis)
    .append("text")
      .attr("transform", "rotate(-90)")
      .attr("y", -36)
      .attr("dy", ".71em")
      .style("text-anchor", "end")
      .text("Count");

  svg.selectAll(".bar")
      .data(data)
    .enter().append("rect")
```

```
      .attr("class", "bar")
      .attr("x", function(d) { return x(d.PotteryType); })
      .attr("width", x.rangeBand())
      .attr("y", function(d) { return y(d.Count); })
      .attr("height", function(d) { return height - y(d.Count); })
      .on('mouseover', tip.show)
      .on('mouseout', tip.hide);
});
```

You can see our gist that contains our interactive bar chart at https://gist.github.com/shawngraham/dc506c6be872625101a1163ad59e4d68 (interestingly, all the files in our gist are listed alphabetically. Scroll down to see all of the files). Let's make this visualization live - pass the filepath to https://bl.ocks.org/ to see the finished result!

(That is, copy the bit *after* gist.github.com/ and paste it so that it's now after bl.ocks.org in your browser's address bar.)

3.2.2 discussion

What might this particular visualization be telling you about this 'site'? How might you integrate this visualizations with other kinds of visualizations (check the example gallery for inspiration) to tell a compelling story?

This walkthrough has only barely skimmed the surface of what D3 can do. But now that you know how the different pieces fit together, you can try fitting your own data into the visualizations that *other* people have made. This can be frustrating - you really have to know how your data is structured, and you must read closely the index.html files of the visualization you want to repurpose to make sure that the code is calling the data correctly. For instance, in our example above, Graham kept getting an error where the bars themselves would not display. He chased this error down by right-clicking on the barchart and selecting 'inspect element'. This opens the browser's inspector tool, and allows us to see what errors are occuring in our html. In Graham's case, the error message was `error invalid value for rect attribute height NaN`.

Error messages are often inscrutable. Copying the error into a Google search usually finds *other* people who have had the same error and who have posted their problem to sites like Stack Overflow, a coding question and answer site. Reading the responses, graham realized that when he repurposed the example from this walkthrough he made the following mistake. Can you spot it?

Original code:

```
svg.selectAll(".bar")
      .data(data)
    .enter().append("rect")
      .attr("class", "bar")
      .attr("x", function(d) { return x(d.Employee); })
      .attr("width", x.rangeBand())
      .attr("y", function(d) { return y(d.Salary); })
      .attr("height", function(d) { return height - y(d.Salary); })
      .on('mouseover', tip.show)
      .on('mouseout', tip.hide);
```

Graham's mistake:

```
  svg.selectAll(".bar")
      .data(data)
    .enter().append("rect")
      .attr("class", "bar")
      .attr("x", function(d) { return x(d.PotteryType); })
```

```
.attr("width", x.rangeBand())
.attr("y", function(d) { return y(d.PotteryType); })
.attr("height", function(d) { return height - y(d.Count); })
.on('mouseover', tip.show)
.on('mouseout', tip.hide);
```

It was very frustrating! It wasn't until he took a break and looked at the code again with fresh eyes that he spotted it. Have you?

3.2.3 exercises

1. Ben Christensen has a very simple line graph using D3 at http://bl.ocks.org/2579599, and the source code at https://gist.github.com/2579599. Login to Github, and then take a fork (a copy) of his source code. Study the code and make a list, of *every* line you would have to modify in order to turn this graph *with its current data* into an illustration of archaeological data.

2. Now try to do it. Click the 'edit' button on your gist, and make your changes. To save your changes, you need to hit the green 'commit' button at the bottom of the page. Display your changes using bl.ocks.org. **Nb** Your changes can take a few minutes to show up - it takes time for things to propagate across the web. View your bl.ocks.org in *incognito mode* so that you're not viewing the cached (old) version of your graph. If your changes show - great! If they don't - use the inspector. What are your errors telling you?

3. Try representing your data in a .json file. Use the examples at the top of this page to guide you. Search for the error messages you will get, and keep track of what you've done, and where you've sought help. This will be more difficult, and frustrating, but you will discover that you are learning throughout.

4. The Data Visualization Catalogue has a wide variety of examples of different kinds of charts along with suggestions for their most appropriate use-cases. Code examples are provided for all of them, and many use D3. Select an example that might be useful for archaeology, and explore its code. Can you make it work? Where do you run into roadblocks: what kinds of questions would you use to remove the roadblock? The point here is to map out *what you don't know* so that you can begin to ask the *right* questions.

3.3 Storytelling and the Archaeological CMS: Omeka, Kora, and Mukurtu

The city of Amersterdam has been building a new north south metro line, under the Amstel River. This massive infrastructure project of course necessitated an impressive amount of archaeological research.

> Rivers in cities are unlikely archaeological sites. It is not often that a riverbed, let alone one in the middle of a city, is pumped dry and can be systematically examined. The excavations in the Amstel yielded a deluge of finds, some 700,000 in all: a vast array of objects, some broken, some whole, all jumbled together. Damrak and Rokin proved to be extremely rich sites on account of the waste that had been dumped in the river for centuries and the objects accidentally lost in the water.

(Department of Archaeology and Archaeology (MenA) 2018)

The Archaeological Department of the City of Amsterdam has created an impressive website that showcases the 700 000 plus finds, but they've done it in an intriguing way. There is a beautiful photo-catalogue called *Stuff* (details here). There is a documentary.

Figure 3.2: screenshot of Stuff book

They are also in the progress of making all of their data downloadable (check in with their dataset page). But what is fundamentally important for the *public communication* of the compelling stories of their work is the interactive component of their site. Their site allows the visitor to arrange one's own display and narrative.

Every object is available, in high-resoultion photography, with associated metadata. The interface allows the visitor to position interesting items in ways to tell a story - for example, a story of 'useless versus re-usable' objects.

By encouraging visitors to adopt the role of curator in a light and engaging way, the site is an excellent example of a 'rabbit hole' down which one begins to really engage with the way the Amstel River is not just an archaeological site but a human dump. It invites the visitor to reflect on ways we humans use and abuse our environment, and the long time-depth of humanity's impact on the environment.

3.3.1 The Content Management System

How did they build this site? If you right-click on the site in whatever browser you are using, one of the option in the pop-up menu will be something like 'Inspector'. The Inspector tool lets you explore (and modify if you wish) the underlying code. It gives us a peak under the hood at what is driving and building a particular site.

Below the Surface is, it appears, custom coded. Can we do that? Yes.... for a given value of 'yes'. Modern websites are a combination of html, css, and js for the most part. When developing a site, a developer will create a 'document object model' that represents programmatically the different parts of a website. The css - cascading style sheet - describes how the different elements of the DOM will look when they become rendered in the browser, and the JS - javascript - does the heavy lifting of retrieving the content that gets inserted into the DOM to create the site. For us, as archaeologists, we will rarely have the luxury of designing and building a site like *Below the Surface*. But fortunately there are 'Content Management Systems' that allow us to create these public-facing sites with a minimum of fuss. If you've ever blogged using Wordpress.com or built a website for a class using a free website builder, then you've used a content management system.

Wordpress for instance uses PHP, a scripting language, to create a 'back-end' that allows users to focus on writing blog posts or pages. When a user posts a new blog post, the text of that post is retrieved from a SQL

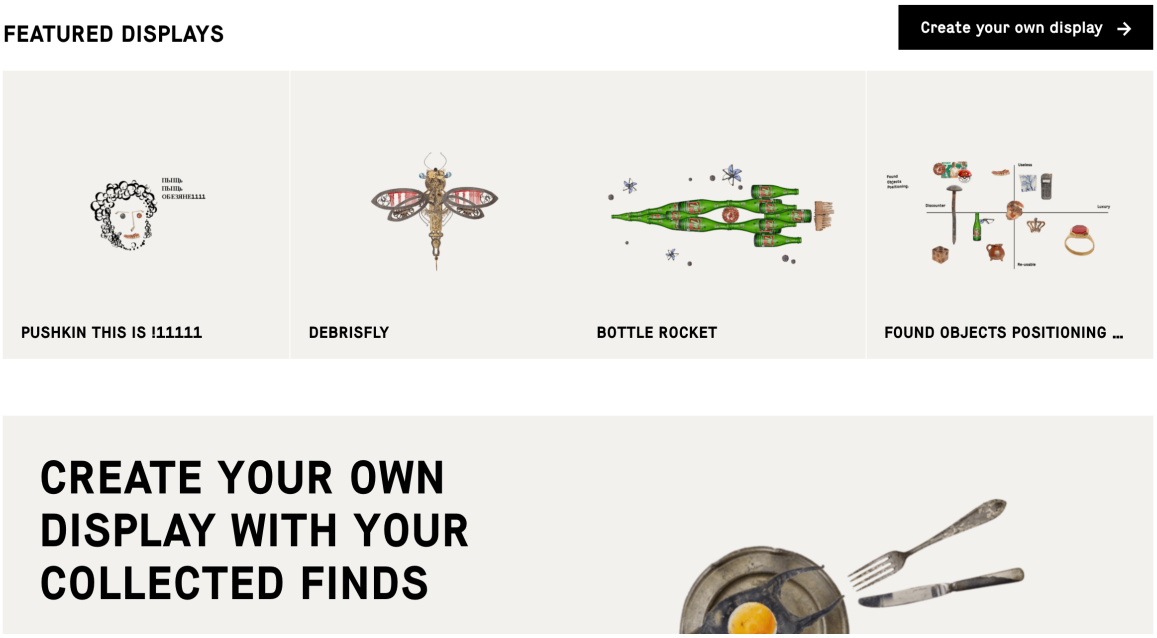


Figure 3.3: Screenshot of the Interactive section form the Below the Surface website

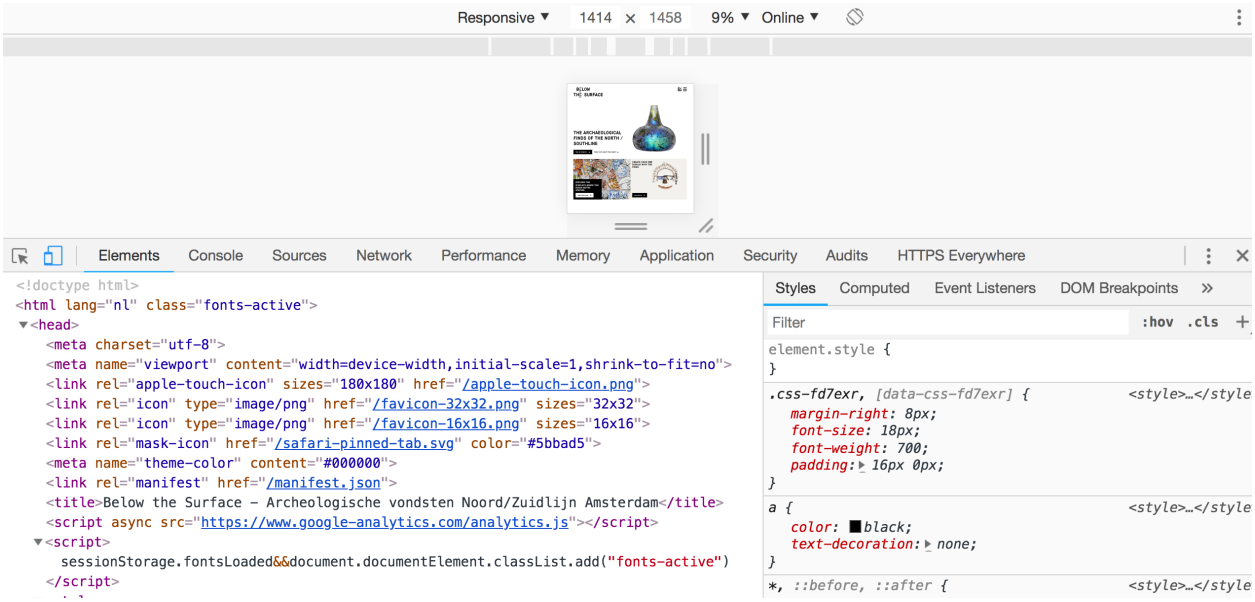


Figure 3.4: inspector screenshot

database and is rendered in the browser via those PHP scripts. Wordpress has a wide variety of themes or pre-built templates of CSS and PHP. Posts can be tagged or categorized in various ways, and template pages can be set up to only display materials that meet certain tags or categories.

Many well-developed archaeological projects use Wordpress as a system for managing their content. The Day of Archaeology used Wordpress; incidentally, because of the way the particular theme used by *Day of Archaeology* worked, which organized content in logical and regular ways in terms of the *page paths* (not all themes do this!) Ben Marwick was able to script a series of analyses using R to understand just what do archaeologists talk about when they talk about their work.

3.3.2 Omeka

Omeka is a Swahili word meaning to display or layout wares; to speak out; to spread out; to unpack; it was chosen by the team at the Roy Rosenzweig Center for History and New Media, George Mason University to describe their project. At the time of its creation, there was a lack of CMS that managed cultural heritage materials. Omeka was built around the metaphor of the archive and the item within that archive. Instead of an editor for making blog posts that are tagged or categorized, Omeka uses Dublic Core metadata to describe individual items from a collection. Then, the user could combine these items to make exhibits, to make webpages, or to make browsable collections. It's worth keeping in mind that *displaying* materials is not the same thing as *exhibiting* materials; there's an element of narrative involved in an exhibition. The back-end structure of Omeka makes exhibits and the items or collections within those exhibits highly discoverable and citable (in contrast to Wordpress). The Programming Historian currently has three lessons on working with Omeka. Omeka is customizable with a wide variety of themes; becaus it is open source, it can also be hacked to (relatively) easily to improve its visual attractiveness.

3.3.3 Kora

Another open-source content management system of interest to archaeologists is Kora, developed by Matrix: The Center for Digital Humanities & Social Sciences in the Anthropology Department at Michigan State University. Kora allows for a wider variety of metadata schemes, and the use of different ontologies to describe cultural heritage materials. There is a further elaboration of Kora called mbira, currently in 'open beta' (meaning, you can use it but you will encounter bugs) that enables one to build mobile (and geolocated) experiences on your collection.

3.3.4 Mukurtu

Mukurtu is a content management system *explicitly* designed to foreground indigenous ontologies and classificatory schemes. It is managed by the Center for Digital Scholarship and Curation at Washington State University. It too is open source.

> In 2007, Warumungu community members collaborated with Kim Christen and Craig Dietrich to produce the Mukurtu Wumpurrarni-kari Archive. Mukurtu is a Warumungu word meaning 'dilly bag' or a safe keeping place for sacred materials. Warumungu elder, Michael Jampin Jones chose Mukurtu as the name for the community archive to remind users that the archive, too, is a safe keeping place where Warumungu people can share stories, knowledge, and cultural materials properly using their own protocols. Growing from this community need, Mukurtu CMS is now an open source platform flexible enough to meet the needs of diverse communities who want to manage and share their digital cultural heritage in their own way, on their own terms. - (Mukurut, 'About')

Mukurtu puts the emphasis on what it calls the '3cs', the 'three structural elements [that] are required to create digital heritage items. You will need at least one community, one cultural protocol within that community, and one category before you can start working with content'. By defining and building the cms

around communities, Mukurtu builds in control and knowledge-keeping around the needs of those groups, who then define the cultural protocols.

> There are two types of cultural protocols: open and strict. Digital heritage items within an open protocol can be viewed by anyone (including anonymous site visitors), while items within a strict protocol can only be viewed by members of that protocol. Multiple protocols can be layered to ensure that users can only view items appropriate for them, and to provide very granular access. For example, if an item is part of the two strict protocols 'Women Only' and 'Elders Only,' then only users who are members of both the 'Women Only' and 'Elders Only' protocols can view that item.
>
> Some examples of cultural protocols could be gender-based (male only, female only), age-based (elders only, no youth), seasonal access only, clan or tribal affiliation, secret/sacred, community only, or public access/open.

3.3.5 Exercises

Content management systems make it much easier to mount cultural heritage materials in various ways online by separating the tasks of hosting and displaying materials from the questions of narrative or access.

1. Search out examples of archaeological, anthropological, or cultural heritage projects using Wordpress, Omeka, Kora, and Mukurtu. Contrast and compare the ways the *platforms* are constraining or enabling access, control, and narrative.
2. Miriam Posner has two lessons at the Programming Historian on using Omeka.net to mount an exhibit. Follow those two lessons (first;second) to mount an exhibit on the material culture in the room in which you are currently sitting. How do the metadata categories and the exhibition templates constrain what story you can tell? How do they push your story in particular ways?
3. If you have access to your own webspace, you can install any of these content management systems yourself. (If you don't have access to your own webspace, one good option is Reclaim Hosting which also has one-click installs for many different CMS). Select one of these CMS and install it, keeping careful notes about all the *tacit* assumptions about your background knowledge concerning the web. Where are the hidden 'gotchas'? What do these imply about the ability for different groups to take control of their own cultural heritage on the web?

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 et al. 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 Virtual Archaeology

3.5.1 Theory

-blah

3.5.1.1 Making, Wayfaring, Tacking and Archaeological Cabling

-blah

3.5.1.2 Materiality

-blah

3.5.1.3 Phenomenology

-blah

3.5.2 Method

-blah

3.5.2.1 The London & Seville Charters

-blah

3.5.2.2 Agency

-blah

3.5.2.3 Authority

-blah

3.5.2.4 Authenticity

-blah

3.5.2.5 Transparency

-blah

3.5.2.6 Paradata

-blah

3.5.3 Practice

-blah

3.5.3.1 Paul Reilly

3.5.3.2 Anthony Masinton

-blah

3.5.3.3 Grant Cox

-blah

3.5.3.4 Richard Allen

-blah

3.5.3.5 Daniel Pletinckx

-blah

3.5.3.6 Nicole Beale

-blah

3.5.3.7 Alice Watterson

blah

3.5.3.8 Costas Papadopoulos

blah

3.5.3.9 Ed Gonzalez-Tennant

-blah

3.5.4 Discussion

-blah

3.5.5 Exercises

-blah

3.6 Archaeogaming

"Archaeogaming" is the study of archaeology both in and of games, largely focused on digital games (arcade cabinets, computer games, console games, etc.) but also including analog games (tabletop, pen-and-paper, cards, dice, etc.). Two major branches comprise archaeogaming: 1) representations of archaeology and archaeologists in games, and 2) treating games as archaeology themselves.

3.6.1 Archaeological Reception

Archaeologists and archaeological settings lend themselves easily to game design. Archaeologists have reasons to travel to exotic places, to go on adventures, to look for artifacts, and to engage with indigenous cultures. Game developers appropriate archaeological tropes for adventure storytelling, encouraging looting and commerce in artifacts. It is rare to find a game that accurately represents archaeologists and archaeology, and players must ask themselves how what is represented on-screen reflects modern interpretations of the field and its practitioners. One can use these games as entrypoints to talking with the general public about archaeology. One can also attempt to work with game developers on improving archaeology's representation. Lastly one can create archaeology-themed games as a way to introduce more accurate portrayals of what archaeology is and what archaeologists actually do. Ethics is central to archaeological reception, supported by the work of L. Meghan Dennis (University of York).

3.6.2 Games as Archaeology

Because games are created by people for other people to use, the become part of modern day material culture. As seen in the list of projects below, archaeologists treat game media as both physical and digital artifacts. Games are also archaeological sites when one considers various files recorded to game media, each with a specific location and function. Games are landscapes, too, inviting players to engage with the content.

3.6.3 Archaeogaming Projects Past and Present

3.6.3.1 The Atari Burial Ground

The first archaeogaming project actually happened in a desert landfill in Alamogordo, New Mexico. In 1983 Atari, Inc. buried hundreds of thousands of unsold/returned video game cartridges, which was the cheapest and most expedient way to dispose of its e-waste. Although the dumping was documented by the Alamogordo Daily News and the New York Times, it became an urban legend in the video game community. The legend stated that Atari's game E.T.: The Extraterrestrial (1982) was the worst video game of all time, and Atari in its shame decided to dump millions of copies in a landfill, covering them with a layer of concrete and then with 10 m of dirt and rubbish. In 2013, a Canadian company, Fuel Entertainment, received the rights from the city of Alamogordo to excavate the landfill in search of the games, filming the project. City waste management expert Joe Lewandowsky conducted photogrammetry to pinpoint the location of where to dig. Archaeologists were invited to assist in the recovery and analysis of the games in April 2014. While for some people the excavation would prove that Atari did dispose of their games in the desert, the archaeologists were interested to learn where the archaeological evidence and details of the urban legend both agreed and diverged. This was the first deposition of an assemblage of 20th-century video games, which for many people growing up in the 1980s had become their cultural heritage. About 800,000 games were actually buried (not millions), and E.T. proved to be one of over 40 different titles dumped in 1983. After the excavation ended, the city donated some of the games to museums worldwide and then auctioned the rest on Ebay, using the proceeds to pay back Alamogordo for equipment and labor, and also to rehabilitate the local museum.

3.6.3.2 Buried

The Twine game Buried (URL: http://taracopplestone.co.uk/buried.html), created by Tara Copplestone and Luke Botham for the University of York's 2014 Heritage Jame, is an example of ergodic literature (a text-based game) that interprets the concept of burial in both academic and personal contexts. You play as an archaeologist, and the game is faithful to actual day-to-day archaeology and the archaeologist's routine. Compare this to the over-the-top action of Tomb Raider where archaeology is treated more as set-dressing and an excuse for advancing the series' story.

3.6.3.3 The No Man's Sky Archaeological Survey

All games lend themselves to archaeological investigation, but only a few contain permanent human populations of players who have made game-spaces their own, investing them with history and lore independent of the game developers' intentions. One such game is Eve Online. Another is No Man's Sky. At its inception, the No Man's Sky Archaeological Survey intended to explore the procedurally generated, universe-sized universe of the game, documenting examples of "machine-created culture" created through the game's algorithms and code. What happened instead became an investigation of the human-settled region of the "Galactic Hub", an enclave of hundreds of citizen scientist-players forced to migrate en masse to another region of the universe because of a catastrophic climate change event. On August 11, 2017, Hello Games released the version 1.3 update, "Atlas Rises", which reset all of the planetary climates and biomes on a quintillion worlds overnight. Players with bases and farms on "Paradise"-class worlds awoke to extreme cold, heat, and toxicity. They abandoned their habitations and left behind messages for others to read, some being notes of farewell, and others with forwarding addresses. Learning about what happened, who these players were, how they settled the galaxy, and what they left behind became the renewed focus of the NMSAS project. Think of it as Roanoke or Herculaneum in space.

3.6.3.4 Reverse-Engineering Digital Games as Archaeological Practice

Prof. John Aycock (University of Calgary) works at the intersection of archaeology and computer science. His work in reverse-engineering games of the 1980s and 1990s explores the necessity of understanding how

hardware and software interface with each other to create a gaming experience, while also deconstructing disk images to understand how games actually work from a technical perspective, identifying precisely where files live on the original game media. His current projects include analyzing full-motion video formats in games, understanding game development through analyzing underlying code, and early attempts at digital rights management (DRM, or copy protection) on physical game media.

3.6.3.5 Purpose

Digital games specifically (and software generally) are the new habitations of the late 20th and early 21st centuries. Billions of people enter synthetic spaces every day to work, communicate, make a living, be entertained, to create, all by passing through the modern day looking glass of the smartphone screen, desktop display, or television. The nature of these digital habitations, their use, construction, and disposal, lends itself to archaeological interpretation. It is a new field within archaeology where we can attempt to understand human occupation and development in digital built environments.

3.6.4 Is Archaeogaming Archaeology? A Future of the Discipline.

When most people think of archaeology they consider it to be focused on ancient cultures and artifacts. However, in the past 30 years, archaeology of the recent past (and of Late Capitalism) has become more mainstream. Archaeology is the study of how people interact with things and within their environment regardless of time period. William Rathje's Tucson Garbage Project (begun in 1973) studied people's everyday household rubbish, comparing the material evidence with anonymous interviews with the residents about what they threw out. In 2018 Justin Walsh and Alice Gorman lead the project recording the history of use of the International Space Station, using tens of thousands of photographs taken from inside the ISS to reconstruct how the interior changed over time.

With archaeogaming, one can work with physical artifacts such as game tapes, cartridges, CDs, and hardware, but one can also conduct archaeological investigation within the games contained on that media. Software programs are built environments and through their daily use by billions of people become contemporary examples of material culture and built heritage. Games contain art and architecture, a history of use, object biographies, game- and user-created histories, and change over time through use and development. Archaeogaming is an entrypoint to understanding the wider world of software archaeologically. When studying the past, archaeologists rely on material evidence, context, and primary sources to reconstruct an understanding of people and how they lived. With software, similar archaeological questions are asked of populations of creators and users, towards an archaeology of the future.

3.6.5 Exercises

1. Play a game featuring an archaeologist or archaeology and describe how the archaeology as represented differs from actual practice.
2. Write your own mini-game that includes either archaeology or a character who is an archaeologist. An accessible tool with which to craft a game is Twine. Tara Copplestone has a guide to twine for archaeologists that can get you started; the Twine Cookbook contains code examples for achieving particular effects. One in particular that might appeal to archaeologists is the potential for triggering passages of text or parts of a game based on the players' actual physical locations
3. Pick a game that has neither archeologists nor archaeology (or cultures, monuments, artifacts, etc.) in it and describe how it is an archaeological artifact, site, and/or landscape.
4. Deconstruct a piece of physical game media as an archaeologist might, and write a site report based on what you find.

3.6.6 Further Reading

The following works will help situate 'archaeogaming' for you in broader archaeological practice: Aycock and Reinhard (2017), Meyers and Reinhard (2017), Reinhard (2015), Reinhard (2017), and Reinhard (2018).

3.7 Social media as Public Engagement & Scholarly Communication in Archaeology

This section is currently under development. Check back soon!

3.7.1 discussion

3.7.2 exercises

3.8 Computational Creativity

Once archaeological data becomes digital, it can escape the narrow purview of scholarly, academic engagement. Should it? Once it's out in the world, we lose our academic authority over it because it can be remixed, re-assembled, and taken to pieces for various ends - not all of them politically, ethically, or morally valid. But given that so much of archaeology is publicly funded, we have to recognize that these things will happen, and that we *should* be equipping our readers, our publics, to engage with these materials in ways that do justice to the peoples of the past, and also, the needs of the future. One way that this can be accomplished is to provide data and tools so that this archaeology can become fodder for art or activist inspired work.

This is part of the rationale for the open access journal *Epoiesen: A Journal for Creative Engagement in History and Archaeology*. From its *about* page, we learn:

> There is a perception that archaeology is for the archaeologists, history for the historians. On our side, there is perhaps a perception that speaking to non-expert audiences is a lesser calling, that people who write/create things that do not look like what we have always done, are not really 'serious'. In these vacuums of perception, we fail at communicating the complexities of the past, allowing the past to be used, abused, or ignored, especially for populist political ends. The 'know-nothings' are on the march. We must not stand by.
>
> In such a vacuum, there is a need for critical creative engagement with the past. In *Succinct Research*, Bill White reminds us why society allows archaeologists to exist in the first place: 'it is to amplify the whispers of the past in our own unique way so they can still be heard today'. We have been failing in this by limiting the ways we might accomplish that task.

In this section, we explore ways of using generative computational approaches to remix archaeological data in a variety of ways - from a whimsical archaeological Twitterbot, to sonification, to glitch photograph, to world building.

To close this introduction to creative engagement, we shall leave the final word to James Ryan. Ryan is a PhD student at the University of California Santa Cruz who is using similar methods to generate a podcast that tells the story of a fictitious town in 'Sheldon County'. A good discussion of his project can be found on eurogamer.net. Ryan says,

> I view coding as a kind of literacy that works a lot like regular literacy: just as writing enables modes of human expression that span from the boring, mundane, functional to the creative, poetic, profound, coding does as well. So, to me, creative coding is to coding as creative writing is to writing. I try to make sure that each day I spend working on a simulation leads to interesting new possibilities being generable by the time I stop working that day. Again, the whole process

> feels kind of like sculpting to me, as opposed to writing, for instance, where the work is more linear. The artist is simply working in a different material context. Working with oil paints is a lot different than working with clay which is a lot different than working on a typewriter, and all of these are different than working with code. But these are all tools of expression."

What could creative engagement and expressive use of coding with digital archaeological data, in terms of the ways the public learns about archaeology? Or how to view the world through an archaeological lense?

3.8.1 Twitterbots with Tracery

this section reuses, adopts and modifies portions of Graham's Programming Historian lesson on Tracery

Strictly speaking, a twitter bot is a piece of software for automated controlling a Twitter account. When thousands of these are created and are tweeting more or less the same message, they have the ability to shape discourse on Twitter which then can influence other media discourses. Bots of this kind can even be seen as credible sources of information. Projects such as Documenting the Now are creating tools to allow researchers to create and query archives of social media around current events - and which will naturally contain many bot-generated posts. Here we demonstrate how one can build a simple twitterbot using the tracery.io generative grammar and the Cheap Bots Done Quick service. Tracery exists in multiple languages and can be integrated into websites, games, bots. You may fork it on github here.

The impetus for building a Twitterbot (or any social media automation) in this way is to use the affordances of digital media to create *things* that could not otherwise exist to move us, to inspire us, to challenge us. Archaeologist Sara Perry for instance is leading the 'Emotive: Storytelling for cultural heritage' project that makes use of bots to such ends. In the uses of such automatons, there is room for satire; there is room for comment. With Mark Sample, we believe that there is a need for 'bots of conviction'.

These are protest bots, bots so topical and on-point that they can't be mistaken for anything else. Per Sample, such bots should be

topical – "They are about the morning new - and the daily horrors that fail to make it into the news."

data-based – "They draw from research, statistics, spreadsheets, databases. Bots have no subconscious, so any imagery they use should be taken literally"

cumulative – "The repetition builds on itself, the bot relentlessly riffing on its theme, unyielding and overwhelming, a pile-up of wreckage on our screens."

oppositional – "protest bots take a stand. Society being what it is, this stance will likely be unpopular, perhaps even unnerving"

uncanny – "The appearance of that which we had sought to keep hidden."

What would such protest bots look like, from an archaeological perspective? Historian Caleb McDaniel's *every 3 minutes* bot shames us with its unrelenting reminder that every three minutes, a human being was sold into slavery in the Antebellum South.

Individuals on Twitter who responded to a query about what the bots of conviction for archaeology might look like suggested some ideas:

> @electricarchaeo a bot tweeting full-resolution images of cultural heritage locked behind tile viewers and fraudulent copyright claims by their holding inst? — Ryan Baumann (@ryanfb) April 22, 2017

> @electricarchaeo A bot tweeting pictures of Native American sacred places that have been desecrated in the name of corporate greed. — Cory Taylor (@CoryTaylor_) April 22, 2017

> @electricarchaeo A bot tweeting the identities of historical assets given inheritance #tax exemption because they are "available" to public view — Sarah Saunders (@Tick_Tax) April 22, 2017

> @electricarchaeo Every time someone says "since the beginning of time, humans have" automatically responding BULLSHIT — Colleen Morgan (@clmorgan) April 22, 2017

Given that so much historical or archaeological data is expressed on the web as JSON, a bit of digging should find you data that you can actually fold into your *own* bot. Graham has built several bots powered by different services. He writes,

> My most successful bot has been @tinyarchae, a bot that tweets scenes from a horrible dsyfunctional archaeological excavation project. Every archaeological project deals with problems of sexism, abuse, and bad faith; @tinyarchae pushes the stuff of conference whispers to a ridiculous extreme. It is a caricature that contains a kernel of uncomfortable truth. Other bots I have built glitch archaeological photography; one is actually useful, in that it is tweeting out new journal articles in archaeology and so serves as a research assistant. (For more thoughts on the role bots play in public archaeology, see this keynote from the Public Archaeology Twitter Conference).

3.8.1.1 Planning: What will your bot do?

We begin with pad and paper. As a child in elementary school, one activity we often did to learn the basics of English grammar was called 'mad-libs' (as in, 'silly - mad - ad libs'). The teacher performing this activity would ask the class to, say, list a noun, then and adverb, then a verb, and then another adverb. Then on the other side of the sheet there would be a story with blank spaces like this:

"Susie the _noun_ was _adverb_ _verb_ the _noun_."

and students would fill in the blanks appropriately. It was silly; and it was fun. Twitterbots are to madlibs what sports cars are to horse and wagons. The blanks that we might fill in could be values in svg vector graphics. They could be numbers in numeric file names (and thus tweet random links to an open database, say). They could be, yes, even nouns and adverbs. Since Twitterbots live on the web, the building blocks that we put together can be more than text (although, for the time being, text will be easiest to work with).

We are going to start by sketching out a *replacement grammar*. The conventions of this grammar were developed by Kate Compton (@galaxykate on Twitter); it's called Tracery.io. It can be used as a javascript library in webpages, in games, and in bots. A replacement grammar works rather similarly to the madlibs you might remember as a child.

In order to make it clear what the grammar *is doing, we are going to* not *create a working bot for the time being. We are making clear first what the grammar does, and so we will build something surreal to surface how that grammar works.*

Let's imagine that you would like to create a bot that speaks with the voice of a potted plant - call it, *plantpotbot*. What kinds of things might *plantpotbot* say? Jot down some ideas-

- I am a plant in a pot. How boring it is!
- Please water me. I'm begging you.
- This pot. It's so small. My roots, so cramped!
- I turned towards the sun. But it was just a lightbulb
- I'm so lonely. Where are all the bees?

Now, let's look at how these sentences have been constructed. We are going to replace words and phrases with *symbols* so that we can regenerate the original sentences. There are a number of sentences that being with 'I'. We can begin to think of an 'being' *symbol*:

```
"being": "am a plant","am begging you","am so lonely","turned towards the sun"
```

This notation is saying to us that the symbol "being" can be replaced by (or is equivalent to) the phrases "am a plant", "am begging you" and so on.

We can mix symbols and text, in our bot. If we tell the bot to start with the word "I", we can insert the *symbol* 'being' after it and complete the phrase with "am a plant" or "turned towards the sun" and the

sentence will make *grammatical* sense. Let's build another symbol; perhaps we call it 'placewhere':

```
"placewhere": "in a pot","on the windowsill","fallen over"
```

("placewhere" is the *symbol* and "in a pot" and so on are the *rules* that replace it)

Now, in our sentences from our brainstorm, we never used the phrase, 'on the windowsill', but once we identified 'in a pot', other potential equivalent ideas jump out. Our bot will eventually use these *symbols* to make sentences. The symbols - 'being', 'placewhere' - are like our madlibs when they asked for a list of nouns, adverbs and so on. Imagine then we pass the following to our bot:

```
"I #being# #placewhere#"
```

Possible outcomes will be:

- I am so lonely on the windowsill
- I am so lonely in a pot
- I turned toward the sun fallen over

With tweaking, and breaking the units of expression into smaller symbols, we can fix any expressive infelicities (or indeed, decide to leave them in to make the voice more 'authentic'.)

3.8.1.2 Prototyping with a Tracery editor

There is a Tracery editor at www.brightspiral.com/tracery/. We will use that to work out the kinks in *plantpotbot*. The editor visualizes the way the symbols and rules of the grammar interact (how they are nested, and the kinds of output your grammar will generate). Open the editor in a new window.

The dropdown menu at the top-left, marked 'tinygrammar', has some other example grammars that one can explore; they show just how complicated Tracery can become. For the time being, remain with 'tinygrammar'. One of the nice things about this editor is that you can press the 'show colors' button, which will color code each symbol and its rules, color-coding the generated text so that you can see which element belongs to what symbol.

If you double-click on a symbol in the default grammar (`name` or `occupation`) and hit your delete key, you will remove the symbol from the grammar. Do so for 'name' and 'occupation', leaving only 'origin'. Now, add a new symbol by clicking on the 'new symbol' button. Click on the name (`symbol1`) and rename it `being`. Click the `+` sign and add some of our rules above. Repeat for a new symbol called `placewhere`.

As you do that, the editor will flash an error message at the top right, 'ERROR: symbol 'name' not found in tinygrammar'. This is because we deleted `name`, but the symbol `origin` has as one of its rules the symbol `name`! This is interesting: it shows us that we can *nest* symbols within rules. Right? We could have a symbol called 'character', and character could have sub-symbols called 'first name', 'last name' and 'occupation' (and each of these containing an appropriate list of first names and last names and occupations). Each time the grammar was run, you'd get e.g. 'Shawn Graham Archaeologist' and stored in the 'character' symbol

The other interesting thing here is that `origin` is a special symbol. It's the one from which the text is ultimately generated (the grammar is *flattened* here). So let's change the `origin` symbol's rule so that *plantpotbot* may speak. (When you reference another symbol within a rule, you wrap it with `#` marks, so this should read: `#being# #placewhere#`).

It still is missing something - the word 'I'. You can mix ordinary text into the rules. Go ahead and do that now - press the `+` beside the rule for the `origin` symbol, and add the word 'I' so that the origin now reads `I #being# #placewhere#`. Perhaps your plantbot speaks with a poetic diction by reversing `#placewhere# #being#`.

If you press 'save' in the editor, your grammar will be timestamped and will appear in the dropdown list of grammars. It's being saved to your browser's cache; if you clear the cache, you will lose it.

Before we move on, there is one last thing to examine. Press the JSON button in the editor. You should see something like this:

```
{
    "origin": [
        "I #being# #placewhere#"
    ],
    "being": [
        "am so lonely",
        "am so lonely",
        "am begging you",
        "am turned towards the sun"
    ],
    "placewhere": [
        "in a pot",
        "in a windowsill",
        "fallen over"
    ]
}
```

Every Tracery grammar is actually a JSON object consisting of key/value pairs, which is what Tracery calls symbols and rules. (For more on manipulation JSON, please see this tutorial by Matthew Lincoln). This is the format we will be using when we actually set our bot up to start tweeting. JSON is finicky. Note how the symbols are wrapped in `"` as are the rules, but the rules are also listed with commas inside `[` and `]`. Remember:

```
{
  "symbol": ["rule","rule","rule"],
  "anothersymbol": ["rule,","rule","rule"]
}
```

(of course, the number of symbols and rules is immaterial, but make sure the commas are right!)

It is good practice to copy that JSON to a text editor and save another copy somewhere safe.

3.8.1.3 Exercise

1. Build a grammar for your bot, following the sequence above. Save the .json file that expresses your bot. Remember that some of the power of Twitterbots comes from their serendipitous placement with other tweets in *your* timeline, *and others'*. The potential for juxtaposition of your bot's message(s) against other people's tweets will also influence the relative success of the bot. How can you plan for these collisions?
2. Get a twitter account for your bot and connect it to Cheap Bots Done Quick

You can plumb a bot into your own, current, account, but you probably don't want a bot tweeting *as* you or *for* you. In which case, you need to set up a new Twitter account. When you set up a new Twitter account, Twitter will want an email address. You can use a brand new email address, or, if you have a Gmail account, you can use the `+tag` trick, ie instead of 'johndoe' at gmail, you use `johndoe+twitterbot` at gmail. Twitter will accept that as a distinct email from your usual email.

Normally, when one is building a Twitterbot, one has to create an app on twitter (at apps.twitter.com), obtain the consumer secret and key, and the access token and key. Then you have to program in authentication so that Twitter knows that the program trying to access the platform is permitted.

Fortunately, we do not have to do that, since George Buckenham created the bot hosting site 'Cheap Bots Done Quick'. (That website also shows the JSON source grammar for a number of different bots, which can

serve as inspiration). Once you've created your bot's Twitter account - and you are logged in to Twitter as the bot account- go to Cheap Bots Done Quick and hit the 'sign in with Twitter' button. The site will redirect you to Twitter to approve authorization, and then bring you back to Cheap Bots Done Quick.

The JSON that describes your bot can be written or pasted into the main white box. Take the JSON from the editor and paste it into the main white box. If there are any errors in your JSON, the output box at the bottom will turn red and the site will try to give you an indication of where things have gone wrong. In most cases, this will be because of an errant comma or quotation mark. If you hit the refresh button to the right of the output box (NOT the browser refresh button!), the site will generate new text from your grammar.

Underneath the JSON box are some settings governing how often your bot will tweet, whether your source grammar will be visible to others, and whether or not your bot will reply to messages or mentions:

Decide how often you want your bot to tweet, and whether you want the source grammar visible. Then... the moment of truth. Hit the 'tweet' button, then go check your bot's twitter feed. Click 'save'.

Congratulations, you have built a Twitterbot!

3. Tracery can power more than bots. This site: https://lovecraftian-archaeology.glitch.me/ uses Tracery to generate archaeological site reports to lovecraftian effect. The source code is at https://glitch.com/edit/#!/lovecraftian-archaeology. Here, the grammar is in the file `grammar.js`. In this grammar, we are generating site names and other names that we want to reuse across the generated story. Beginning with the `origin`, diagram out how the elements interrelate to create the site report. Then, once you see how the grammar is working, adapt the code to create your own work of fiction-meets-archaeology.

3.8.1.4 Good bot citizenship

As Cheap Bots Done Quick is a service provided by George Buckenham out of a spirit of goodwill, do not use the service to create bots that are offensive or abusive or that otherwise will spoil the service for everyone else. As he writes,

> If you create a bot I deem abusive or otherwise unpleasant (for example, @mentioning people who have not consented, posting insults or using slurs) I will take it down

Other pointers for good bot citizenship are provided by Darius Kazemi, one of the great bot artists, are discussed here.

3.8.2 Chatbots

Chatbots are conversational robots designed to mimic human interaction - with varying degrees of success.The EMOTIVE project at York University is exploring such chatbots for the purposes of storytelling and cultural heritage. The poet and programmer Allison Parish, of New York University, designs and builds various algorithmic explorations of language including such chatbots. In the creativity binder there is a notebook by Parrish ("semantic_similarity_chatbot") that walks you through programming such a chatbot by training it on a database of movie dialogue. It works by calculating the 'semantic similarity' between what you type and what exists in its database of similar responses.

The notebook is very well commented out and takes you through the concepts in some detail. At the end of the notebook, Parrish makes some suggestions on where to take it next-

> Use the metadata file that comes with the Cornell corpus to make a chatbot that only uses lines from a particular genre of movie. (How is a comedy chatbot different from an action chatbot?) Use a different corpus of conversation altogether. Your own chat logs? Conversational exchanges from a novel? Transcripts of interviews on news programs?

3.8.2.1 Exercise

1. Train the chatbot on 'archaeological' movies. What does this say about the public consciousness of archaeology?
2. Train the chatbot on a corpus of writing by a particular archaeologist. Explore the responses - what does your virtual archaeologist reveal about their practice?

3.8.3 Sonification

this section resuses, adopts and modifies portions of Graham's Programming Historian lesson on the sound of data

While there is a lot of research into archaeo-acoustics (much of which involves digital manipulation of the sound files), in this section we want to take the opposite tack - of *creating* aural representations of our data.

Pleasing or discordant sound from data can be used to signal many different things - from warning tones that something is wrong, to artistic interventions meant to draw attention to environmental degradation, to economic inequality over space. Sonifying archaeological materials could be an effective strategy in a work of public archaeology or advocacy.

To begin with, sonification is a useful exploratory technique to make your data unfamiliar again. By translating it, transcoding it, remediating it, we begin to see elements of the data that our familiarity with visual modes of expression have blinded us to. This deformation, this deformance, is in keeping with arguments made by for instance Mark Sample on breaking things, or Bethany Nowviskie on the 'resistance in the materials'. Sonification moves us along the continuum from data to capta, social science to art, glitch to aesthetic. So let's see what this all sounds like.

Sonification is the practice of mapping aspects of the data to produce sound signals. In general, a technique can be called 'sonification' if it meets certain conditions. These include reproducibility (the same data can be transformed the same ways by other researchers and produce the same results) and what might be called intelligibility - that the 'objective' elements of the original data are reflected systematically in the resulting sound (see Hermann 2008 for a taxonomy of sonification). Last and Usyskin (2015) designed a series of experiments to determine what kinds of data-analytic tasks could be performed when the data were sonified. Their experimental results (Last and Usyskin 2015) have shown that even untrained listeners (listeners with no formal training in music) can make useful distinctions in the data. They found listeners could discriminate in the sonified data common data exploration tasks such as classification and clustering. (Their sonified outputs mapped the underlying data to the Western musical scale.)

Last and Usyskin focused on time-series data. They argue that time-series data are particularly well suited to sonification because there are natural parallels with musical sound. Music is sequential, it has duration, and it evolves over time; so too with time-series data (Last and Usyskin 2015: 424). It becomes a problem of matching the data to the appropriate sonic outputs. In many applications of sonification, a technique called 'parameter mapping' is used to marry aspects of the data along various auditory dimensions such as pitch, variation, brilliance, and onset. The problem with this approach is that where there is no temporal relationship (or rather, no non-linear relationship) between the original data points, the resulting sound can be 'confusing' (2015: 422).

In the notebook on sonification, we walk through the process of mapping time-series data against the 88 key keyboard. In the code, you will see a number of different methods that can be used.

The data are represented in json:

```
my_data = [
    {'event_date': datetime(1792,6,8), 'magnitude': 3.4},
    {'event_date': datetime(1800,3,4), 'magnitude': 3.2},
    {'event_date': datetime(1810,1,16), 'magnitude': 3.6},
    {'event_date': datetime(1812,8,23), 'magnitude': 3.0},
```

```
    {'event_date': datetime(1813,10,10), 'magnitude': 5.6},
    {'event_date': datetime(1824,1,5), 'magnitude': 4.0}
]
```

The name of the second value here is 'magnitude', but it could be anything - counts, percentage, averages. The first value, in the sample code, uses python's `datetime` format and so has to be written as in the example. Unfortunately, `datetime` only works for dates from 1 CE onwards. This means that for any dates of interest to many archaeologists, this approach would either seem to be not useful, or require too much bother in order to get the dates correctly expressed.

This is not as big a problem as it first may appear. The actual calendar date is not important; it's the internal relationship that matters. That is to say, we're interested in the evolution over time as represented by the evolution of the sonification as we listen to it. In which case, say we had the following data we wanted to sonify:

```
Phase 1: 234 sherds of pottery A
Phase 2a: 120 sherds of pottery A
Phase 2b: 45 sherds of pottery A
Phase 3: 47 sherds of pottery A
Phase 1: 120 sherds of pottery B
Phase 1: 34 sherds of pottery B
Phase 2a: 200 sherds of pottery B
Phase 2b: 180 sherds of pottery B
Phase 3: 87 sherds of pottery B
```

and we know that there are approximately 20 years between Phases 1 and 3, for pottery A we could express this like so:

```
my_data = [
    {'event_date': datetime(1010,1,1), 'magnitude': 234},
    {'event_date': datetime(1020,1,1), 'magnitude': 120},
    {'event_date': datetime(1025,1,1), 'magnitude': 45},
    {'event_date': datetime(1030,1,1), 'magnitude': 47}
]
```

. . . run the code, save the output, and then do the same thing again for pottery B.

3.8.3.1 Exercise

1. Collect some data that you would like to sonify. Construct an array as above to express each potential 'voice' in your final composition. What do you think you might 'hear'?
2. Try the Programming Historian sonification tutorial using Musicalgorithms to represent that data. What is the key difference between that approach, and the approach detailed here using miditime? How does it differ, and what does that *sound* like? Try sketching out how you could use these differences to explain key concepts in archaeology to a public audience.
3. Launch the jupyter notebook to explore miditime's approach to sonifying time-series data. Build mutliple voices, and mix them together using Garageband or Audacity. What creative decisions do you have to take in order to make this work? How do they change the data, or what you are able to hear? What *do* you hear - and *what* might it mean?

3.8.4 Worldbuilding

There is an approach to video games called 'procedural content generation' to create the world of the game so that it is always unique when you play it. *Minecraft* is one game that uses procedural content generation; *No Man's Sky* is another. In the creativity2 binder there is a subfolder called 'WorldBuilding' that contains a

notebook, 'model demo'. This notebook builds a world by simulating certain environmental processes, where the terrain is modeled as a series of Voronoi polygons with elevation and simulated water erosion along the edges, after Martin O'Leary's tutorial on Creating Fantasy Maps. Nomads move into this terrain and travel around it, from which movement trails and cities emerge. These cities then interact following the well known Axelrod Tribute Model. The final elements of this demonstration generate a kind of chronicle that tells the 'history' of this world.

Another kind of procedural history generator can be found in the same binder in the 'history generator' folder, which also permits the visualization of the 'history', like so:

Figure 3.5: Output of the script

This script contains a very simple model of state formation, fusion and fision. It also incorporates a Tracery grammar to 'liven up' the resulting chronicle. As a final feature, it uses the graphviz library to visualization the relationships through time.

3.8.4.1 Exercises

1. Examine the underlying models for interaction. What are their theoretical underpinnings? What would have to be modified to make these generative worlds/histories more 'real'?
2. How could such generative histories be used for public archaeology?

Chapter 4

Eliding the Digital and the Physical

Digital data overlays the world we live in. Archaeologically, this has been the case since the emergence of computational approaches to archaeology in the 1970s. Practically, this has meant better visualizations, better analyses, better ways to cope with the data. The last twenty years however have seen digital data *permeate* the physical world in ways hitherto impossible. Perhaps you've walked down the street and been curious about a building you've seen. A few moments with your smartphone, and the information is there. A notification pops up, telling you about other similar buildings nearby. A bicycle rental stand is nearby - you free one with the app on your phone (which has automatically adjusted the price based on the density of other bicycles nearby that are being used).

It makes no practical difference, these days, to make a hard-and-vast distinction between the digital and the physical. A car crashes into a telephone pole, and internet service goes out for a neighborhood. A bug in the code causes the airline reservation system to go down. In this section we consider some of the ways digital techs of use to archaeologists smear across the digital and physical realms. Once you become familiar with some of the ways these technologies work, you will be better able to judge whether a given use at a site, in a project, in an article, is a valid or useful intervention.

4.1 Photogrammetry

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

Photogrammetry, as the name implies, is the derivation of measurements from photographs. In our case, we are talking about triangulation. We identify from a series of photographs of an object a series of 'control points' (the same features across multiple photographs) and 'rays' (lines-of-sight). Then we work out where the rays intersect via triangulation. (Our eyes and brain do this naturally and we call this 'depth perception'). Once we've done this enough times, we end up with a cloud of points which are the three-dimensional position in space *relative* to the camera that took the photographs. Various algorithms can then join-up the points into a meshwork, onto which we can project the image information. The quality of the result depends on the software and algorithms, the quality of the camera, background lighting, and the skill of the photographer.

Nowadays, visually appealling models can be generated by low-cost smart-phone apps and shared immediately with services such as Sketchfab (check out their Cultural Heritage & History category). For recording purposes or for 3d printing artefacts afterwards for study, higher-power cameras and software are generally used (Agisoft Photoscan is an often-used product in this regard). Open source software is quite powerful, but packages like VisualSFM (one of the best known) can be difficult to set up. (If you are familiar with Docker, Ryan Baumann has simplified some of the process. More images and more computational power does not always lead to better results, however (Baumann 2015).

In general, it takes practice to develop the necessary photographic and technological skill/intuition to get the best out of one's photographs and one's software. In the exercise below, we introduce you to a workflow using Regard3d, a graphical user interface for working with a number of algorithms at each step of the process. It is also worth noting that 3d models can be generated from high-quality drone or other video; a workflow for this (which also uses Regard3d) may be found here.

The general process runs like this:

- image capture: take overlapping images; you want a high degree of overlap. 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 with regard to the points of overlap in the images.
- image matching: tie points are matched and camera orientations are deduced
- dense point cloud generation. The intersection of rays then allows us to work out the location of these points in space
- secondary product generation
- analysis / presentation

4.1.2 Further Readings

The following will challenge your sense of what is possible with 3d photogrammetry, and how/why archaeologists should think critically about this technology.

Eve, S. (2018). Losing our Senses, an Exploration of 3D Object Scanning. Open Archaeology, 4(1), pp. 114-122. Retrieved 7 Aug. 2018, from doi:10.1515/opar-2018-0007

Reilly, P. (2015). Additive Archaeology: An Alternative Framework for Recontextualising Archaeological Entities. Open Archaeology, 1(1), pp. -. Retrieved 7 Aug. 2018, from doi:10.1515/opar-2015-0013

Verdiani, G. (2015). Bringing Impossible Places to the Public: Three Ideas for Rupestrian Churches in Goreme, Kapadokya Utilizing a Digital Survey, 3D Printing, and Augmented Reality. Open Archaeology, 1(1), pp. -. Retrieved 7 Aug. 2018, from doi:10.1515/opar-2015-0007

4.1.3 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. Try with just the default values. If the system cannot compute matches, try again but this time slide the keypoint density sliders (two sliders) all the way to 'ultra'. Using 'ultra' means we get as many data points as possible, which can be necessary given our source images (warning: this also is computationally very heavy and if your machine does not have enough memory the process can fail). 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. **Warning** 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.

4.1.3.1 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* - uppercase/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:

myphone;LG3;6

Save the file. Now you can open Regard3d, 'add a picture set', select these images, and Regard3d will have all of the necessary metadata with which to work.

4.2 3D Printing, the Internet of Things and "Maker" Archaeology

Anna Esther Heckadon (2018)

4.2.1 3d Printing - a Workflow

In the workflow described below, it is entirely possible to 3d print a wide variety of data that are not, strictly speaking '3d' in the sense of existing in three dimensions in the 'real world'. For instance, one could extract the hue, colour, and saturation values of a collection of site photographs and 3d print an object that represents several hundred photographs as a surface where x = hue, y = colour, and z = saturation.

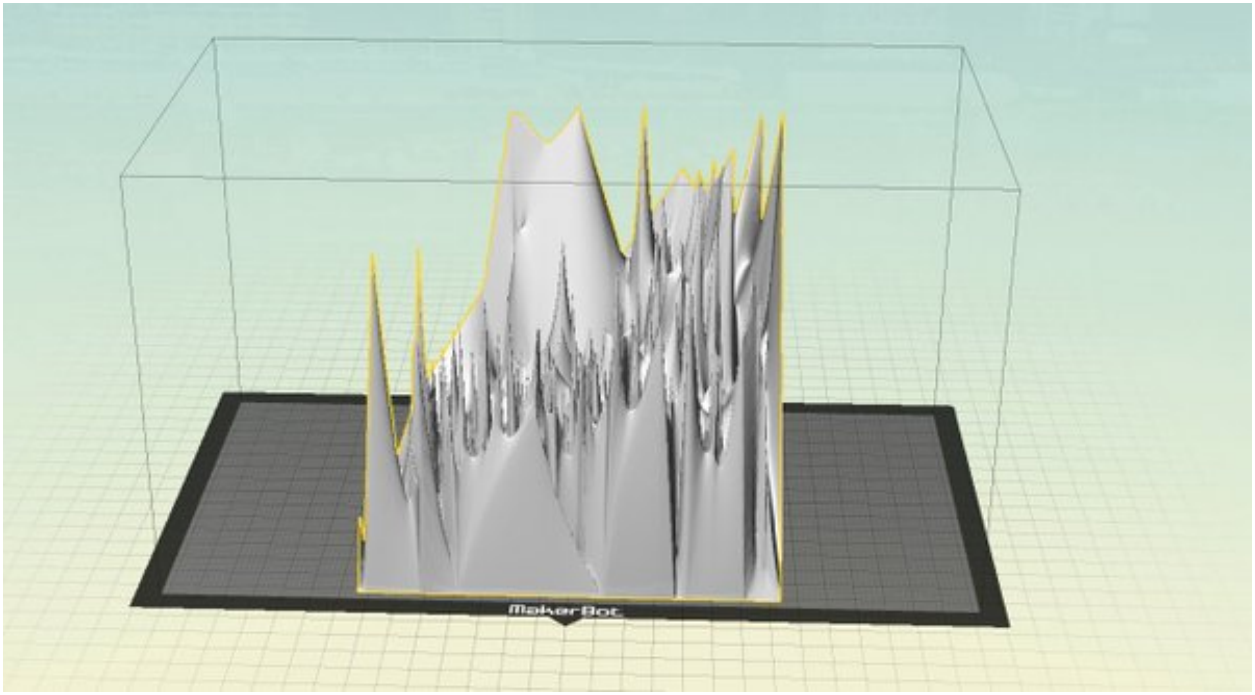


Figure 4.1: Hue, color, and saturation values for 9000 images depicting human remains for sale collected from Instagram

Users will obtain values for x,y,z and represent those values as points using the values from x,y. The z value will remain in the attribute table and will be used later. Once the points are represented in x,y space, we'll create a digital elevation model (DEM). This essentially will take the x,y points and interpolate or mathematically guess all of the values between the points while assigning a pixel value to each location from the z attribute table. Typically this z value represents elevation (hence the name of the DEM), but this is typically colour coded in various shades of grey/black/white. The last step will be to use a tool in QGIS to

convert the DEM to an STL file, which is a standard 3d object file format that is used for makerbot. This will create the 3D object, which can then be printed using the 3D printer.

Let's walk through the process.

You have data values that you'd like represent in a 3D space. To display values into 3D space, you'll need values inputed in x,y and z. x,y will represent your data in a 2D space, while the addition of the z value will add the 3rd dimension. The method outlined below is typically done to display geographic data, but it can also be used with other types of data. The method below will use non-geographic data to create a 3D object file.

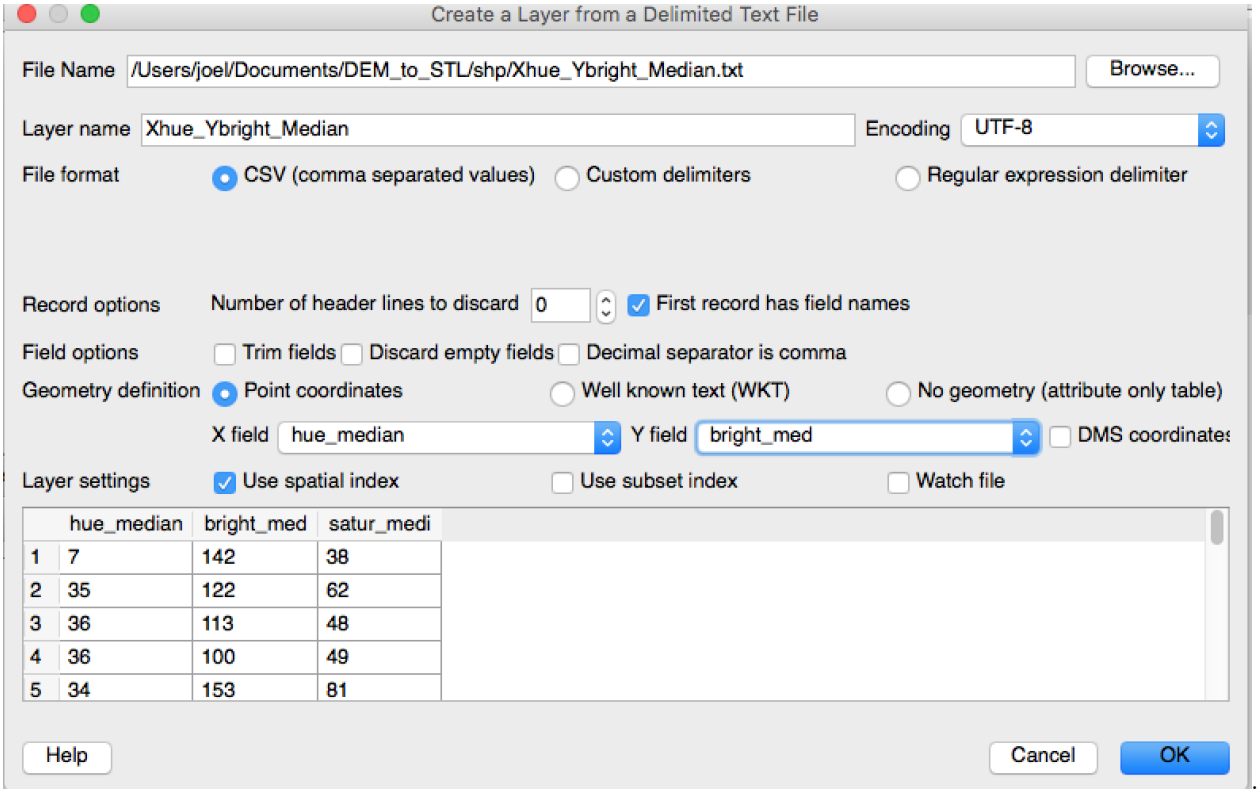
The software used in this workflow is QGIS. QGIS is a free open source desktop geographic information systems (GIS) application that allows users to view, edit and analyze geographic information. Users can download the tool for Windows, MAC OS X or Linux at the following website. The workflow below was done using QGIS 2.14.

4.2.1.1 Create a point shapefile

1. Collect or ensure that you have a comma-separated file (.csv) with values of x,y,z. This is displayed in a text editor, but it can also be viewed as tabular data.

```
hue_median,bright_med,satur_medi
7,142,38
35,122,62
36,113,48
36,100,49
34,153,81
31,148,61
29,120,129
29,176,96
24,169,78
29,127,122
27,249,38
28,202,81
```

2. Open QGIS and in the menu, click on Layer > Add Layer > Add delimited text layer... This will allow us to add our .csv file in QGIS



You should now see some points displayed in QGIS. You'll notice in the layers panel on the left that there is a point file. This point file is considered to be a vector feature with its geometry represented as a point. If you can't see the Layers Panel, select in the menu View > Panels > Layers Panel. In the layers Panel, turn the layer on/off by clicking on the box next to it. Keep in mind that this point file is only temporary and should now be saved properly.

3. In the Layers Panel, right click on the file and select Save as... and change to the settings as outlined in the screenshot. Ensure that you change the CRS to EPSG 3857 WGS 84 / Pseudo Mercator.

4. In the Layers Panel, remove the temporary point file by right clicking on the file and clicking remove. You'll want to have only the pointfilepseudo in the Layers Panel.

Your data is now viewed in 2D space and represented as points. Each point (x,y) has a z value associated to it, even if we can't see it. You can view this z value by clicking on identify tool in QGIS and then clicking on a point. The underlying attribute table will appear, indicating the value of x,y and most importantly the z.

4.2.1.2 Create a Digital Elevation Model

Our next step is to fill in the gaps between the points. To do so, we'll convert the vector point geometry to raster pixels while we use interpolation tools in QGIS to do this. This is quite common in a GIS software, where users attempt to interpolate the elevation of the area with use the spot height points. The output product from this operation is called a digital elevation model (DEM).

1. In QGIS, click on View > Panels > Toolbox. This will open up the Processing Toolbox window on the right. We'll be using one of the interpolation tools from this toolbox to create our Digital Elevation Model (DEM).
2. In the search box, type v.surf. This will bring up various interpolation tools that can be used for your interpolation with the use of GRASS tools. GRASS (Geographic Resources Analysis Support System) are open source tools that can be used in QGIS or as a standalone application. In this case, we're using the GRASS tools in QGIS.

3. Double click on the v.surf.idw - surface interpolation by... The idw in this particular case refers to the interpolate method of Inverse-Distance Weighted. This method essentially gives less weight to known points that are furthest when trying to interpolate values.
4. Fill in the information as seen in the screenshot. The algorithm will process and your output will be added to the Layers Panel.

You'll see various shades of grey/black/white sections when looking at the DEM raster in QGIS. Each shade of grey represents a different value of z.

4.2.1.3 Create a 3D object (.stl)

For 3D printing, users must have specific file formats. An .obj or .stl are some of the common formats that can be used to create 3D models. Meshlab is a free tool that can be used to view and convert between multiple types of 3D object formats. In our particular case, we'll be printing our final 3D object on a makerbot printer. We'll need the file format to be in .stl. Luckily for us, QGIS can export directly from a DEM to STL file.

1. In QGIS, click plugins > Manage and Install Plugins...
2. In the search box, type DEMto3d. This should bring up the plugin that's required for our conversion. Click on the plugin and install it. Exit the plugins menu when complete.
3. In the QGIS menu, click on Raster > DEMto3D > DEM 3D Printing. Select the InterpolateIDW layer and the remainder of the settings as outlined in the screenshot below. Once complete, click on Export to STL. Once prompted to save the file, select the appropriate directory.

4.2.1.4 Print 3D object or create AR

This will export an stl file that is rather large in file size. This stl file can be used to print a 3D representation of the model or can be used to view the model as an augmented reality object.

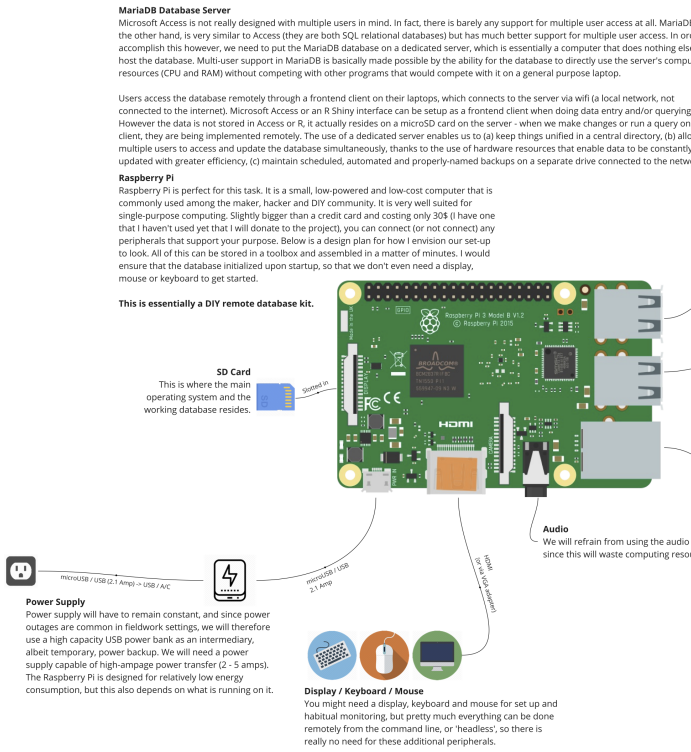
4.2.1.5 discussion

4.2.1.6 exercises

4.2.2 Using Raspberry Pi in the Field

Archaeology is an inherently collaborative process that happens in various places simulataneously. It is therefore often challening to ensure that archaeological databases, which serve to store and organize inter-related information obtained through various methods and practices, are kept up to date and syncronized across research environments. In order to deal with these challenges it might be necessary to set up a database hosted on a network, which would enable multiple users to simultaneously engage with the database while ensuring that the data remains orderly and complete.

In order to implement such a system we need to set up a network, set up a server, set up a database along with user-friendly interfaces, and ensure that the system is aligned with and contributes positively to the goals of the overall project.



The system that we're going to build will look something like this:

It consists of: - a local network generated and managed by a wireless router - a Raspberri Pi mini-computer upon which the database will reside and be served to other devices across the network - the software configured to host the database and ensure that the data is secured and backed up - user interfaces that talk to the database and that encourage desired user behaviour - other useful services such as a file sharing server and data visualization portal that may help encourage collaborative and informed research

The system that we're going to build may be suitable for some, but not for others. It is therefore important to remember that this all requires some degree of flexibility so that what you make suits the overall goals of the project you're contributing to. So you may copy these instructions directly or tinker around to suit your needs!

Some Notes on Hardware: This is meant to be a very portable, low energy use and inexpensive setup. While I recommend that you use a dedicated server - a computer that is configured to run the database server, and only the database server - you can actually host the database on a regular laptop that you also serves as the client. I will be including notes on how to configure this to work all on one device, but I recommend that you use this kind of setup for educational and testing purposes only, and not in a production environment.

Raspberry Pi is available online for around 45$ CAD at https://www.raspberrypi.org/products/, but your local makerspace will likely be happy to lend you one to play with. You will also need a SD card with a capacity of at least 8GB and with adequate read/write speeds (see https://www.raspberrypi.org/documentation/installation/sd-cards.md for more info), a wireless router (this one is 40$ and tiny - perfect for portability! https://amzn.to/2HoOM6W), a power source that is capabale of high-ampage output of at least 2.4A (a wall socket will suffice, but in the field I recommend an external battery pack to be used, to allow for increased portability and as a safeguard against power failures - I use this one at 43$: https://amzn.to/2Ht8quv) and a couple USB flash drives with varying storage capacities that suit your needs.

This guide is designed with unix-users in mind - so those who use MacOS or Linux. Sorry Windows users :(

4.2.2.1 Networking Basics

The goal of this project is to enable multiple computers to communicate with a database on a central server. In order to accomplish this we first need to set up a network, which enables this communication to happen. **A network is a set of interconnected computers, which communicate with each other using cables or wireless connections as well as protocols that manage the connections.**

The internet is one kind of network that relies on immense cabling infrastructure and DNS protocols that have global reach (DNS, or Domain Name Servers, distribute codes to every computer on the internet, which are comparable to phone numbers in a phone network). However in our case we are going to set up a local network (also known as a local area network, LAN or intranet) that has a limited scope and range. **A local network distributes IP addresses to connected devices that enable them to communicate with each other, but it does not necessarily connect these computers to the outside world of the internet.** It can be imagined as a neighbourhood with roads and walkways of its own that serve the local community; the neighbourhood may be connected to the rest of the world via a regional or national highway. **The local network is managed by a router, which is a specialized computer that assigns local IP addresses and also helps direct traffic to and from the broader internet.** You can think of the router as a local post office, which receives mail directed to the town and distributes the mail to every household.

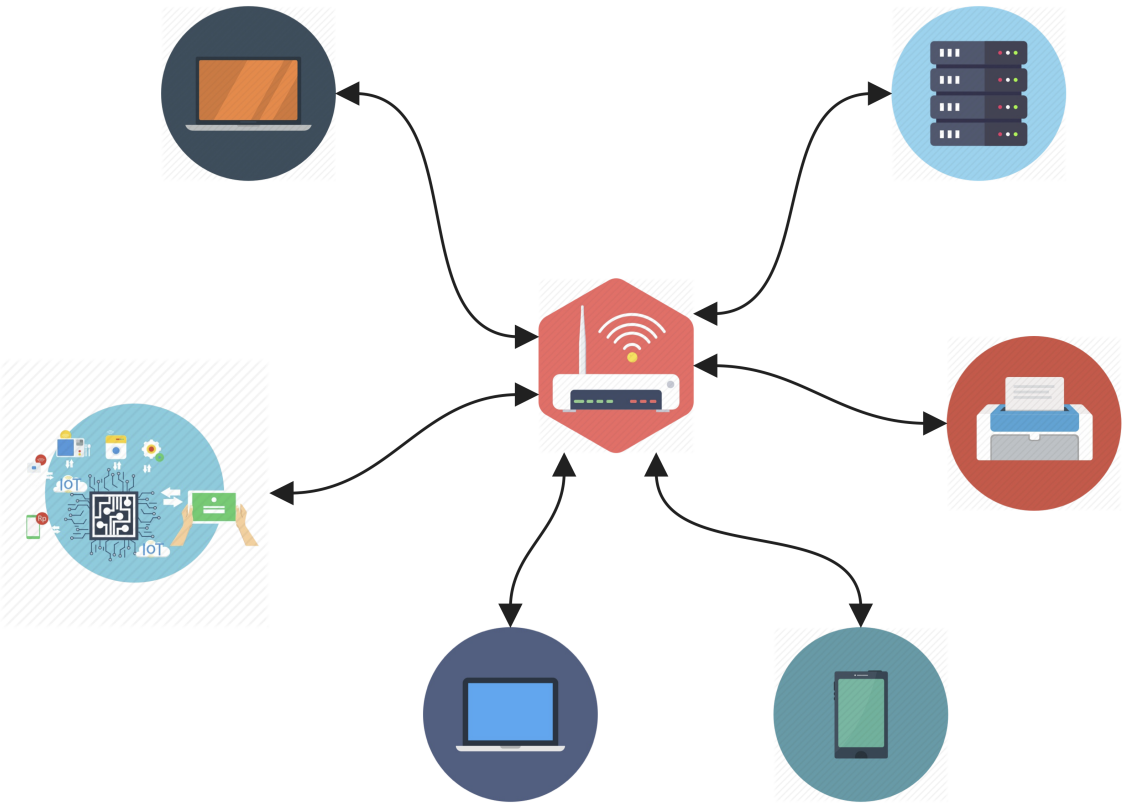


Figure 4.2: local network

Other kinds of specialized buildings might exist in the town as well. You might do all sorts of things in

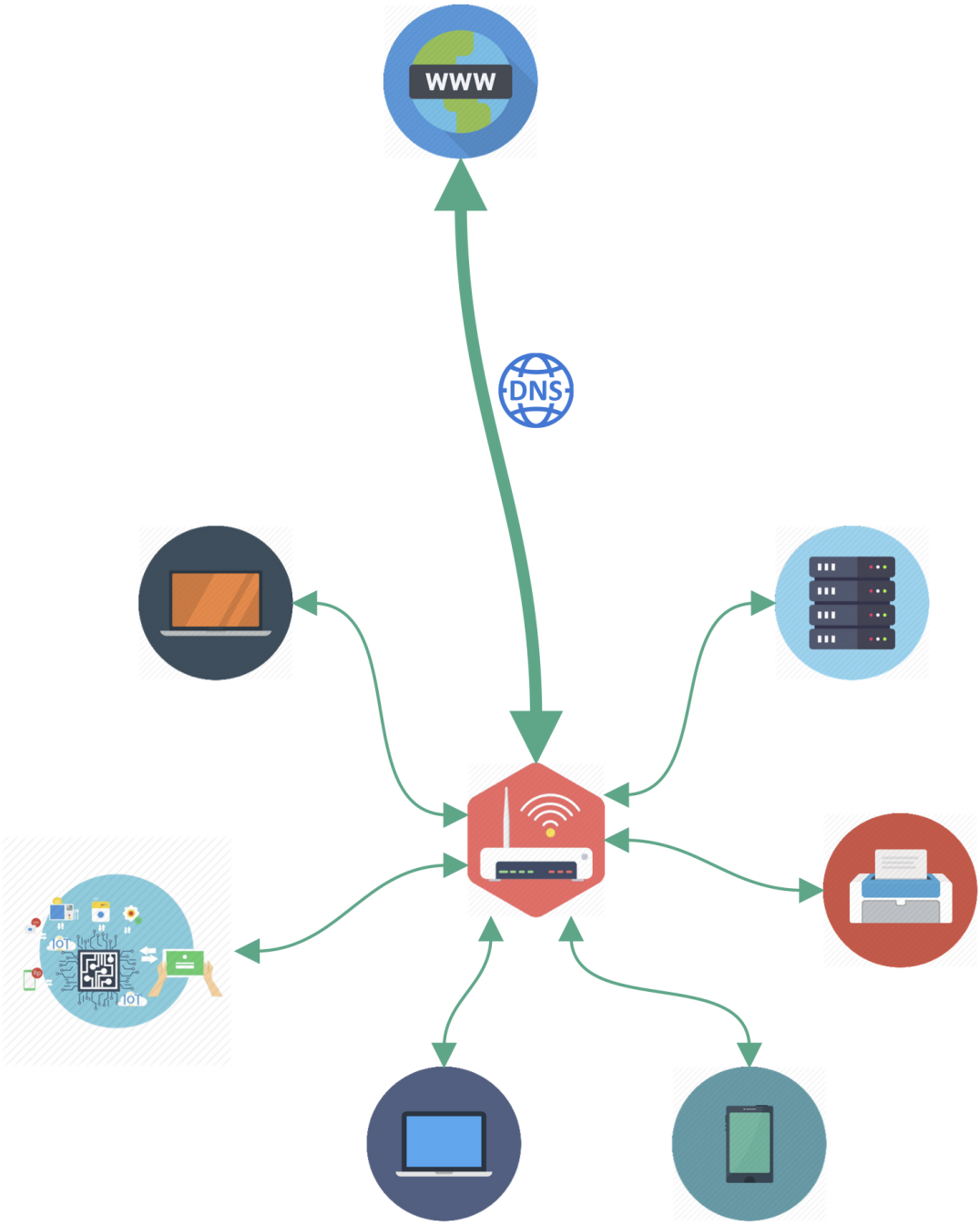


Figure 4.3: internet connected network

your home, such as cook, water the garden, wash the dog, or host a party, which each consume varying degrees of food and water (including some waste) but services like grocery store or water reclamation plants need to be streamlined and focused in order to keep up with the inflow and outflow of material that they need to process, which is immense because they engage with each and every household in town. These represent specialized computers on a network called servers (also referred to as hosts, clusters or 'the cloud'). **Servers are centralized computers that manage and distribute information to and from clients across the network. They are designed to be as lightweight and efficient as possible,** such as by minimizing flare in user interfaces to enable more effective use of computing resources directed towards the task at hand, or by keeping the temperature cool to prevent shutdowns and slowdowns due to overheating. The hardware places hard limits on how efficient a computer may operate, but configuring the software can go a long way in stretching a computer's capabilities.

Finally, **a user (also referred to as a client, terminal or workstation) is any other computer on the network that is served information or passes along information to be served to others.** The things that they do on their own computer are local relative to other computers on the network. The terms 'local' and 'remote' are therefore commonly used to designate things that occur close to home base, and things that occur further away, respectively.

4.2.2.2 Raspberry Pi and Raspbian

Raspberry Pi is an open source, inexpensive and low-power mini-computer that includes all the necessary components needed to do a wide variety of creative things. It's very popular among makers and DIY/DIWO hackers due to its extensibility and flexibility of use. Raspberry Pi has its own Linux distribution called Raspbian OS, which is designed to be lightweight and consume relatively little power.

While you *could* connect a display, keyboard and mouse to the Raspberry Pi and interface with it directly, **it's actually very common to run it 'headless' - by running commands via the command line or terminal over a network, without the use of a visual user interface**. We do this by establishing a SSH connection, by typing commands in a terminal window on a remote computer that are actually implemented on the local computer. The network setup enables all of this to happen.

It may seem a little intimidating, but using the terminal is actually quite easy and systematic once you get the hang of it - it just takes some getting used to! Moreover, experimenting with a Raspberry Pi is arguably the best way to learn how to do this, since the operating system can be wiped and reinstalled very easily. So if you mess up (as everyone inevitably does) you'll be back up and running in a matter of minutes!

So let's get started by installing Raspbian on a SD card. Start by inserting the SD card into your computer using an adapter (these tend to come with the microSD card if you just bought a new one, but makerspaces and photographers may have one lying around for you to borrow). Once it's in it will mount automatically and you should see it pop up on your finder or file explorer. It might have a different name, but be sure you are able to identify it and associate it with what you just inserted into your laptop or desktop computer.

Now we'll have to wipe and format the microSD card using an SD Card Formatter tool. Use the one made available by the SD Association, available at: https://www.sdcard.org/downloads/formatter_4/.

[more details and screenshots here]

Once the microSD card is wiped and formatted, we want to etch the Raspbian operating system onto the card. First, download Raspbian OS from https://www.raspberrypi.org/downloads/raspbian/, and then use Etcher (available at https://etcher.io/) to write it to the microSD card. This may take a few minutes. Notice that the mounted volume is re-named as 'boot'.

[more details and screenshots]

There are a couple other things that we need to do to ensure that we can interface with the Raspberry Pi headless. First we need to create a file in the OS that enables SSH interface. Open terminal (/Applications/Utilities/Terminal.app, or by hitting Command+Space and then typing Terminal.app) and type the following command, then hit enter:

```
touch /Volumes/boot/ssh
```

This command (`touch`) creates an empty text file in a location specified after the space. In this case, we are creating a text file called 'ssh' in the main directory of the boot volume. What this file does, in simple terms, is specify that it's okay to allow other computers to interface with the Raspberry Pi over the network.

So now that we've specified that it's okay to enable SSH connections over a network, we need to enable the Raspberry Pi to connect to the network in the first place. If we were using a graphical user interface, we would go to a menubar, select the network and input the password. Only then would we be able to connect to the Raspberry Pi headless. However because we are not using a graphical user interface, we need to devise a way to make the Raspberry Pi connect to the network automatically, without prior intervention on our part. We can do this by creating another file, `wpa_supplicant.conf`, similarly to how we made the `ssh` file above.

```
touch /Volumes/boot/wpa_supplicant.conf
```

Now we need to edit this file to specify the login credentials for the network that we want to join. We'll use a terminal-based text editor called `nano` to edit this file.

```
nano /Volumes/boot/wpa_supplicant.conf
```

Pro Tip: alternatively, we could navigate to the directory where `wpa_supplicant.conf` is located and then run nano, breaking this up into two separate steps:

```
cd /Volumes/boot
nano wpa_supplicant.conf
```

The `cd` command moves you around directories. Check what files and sub-directories are in your current directory with `ls`. See the cheatsheet provided by the Raspberry Pi Foundation that lists some common terminal command and explains how to use them: https://www.raspberrypi.org/documentation/linux/usage/commands.md

After the `nano` command you should see a blank file like this:

[screenshot]

Type the following into `wpa_supplicant.conf`. Remember to change the generic country code, network SSID and network password, including the <> characters, to suit your setup. The quotes should remain in place.

```
country=<XX>
ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev
update_config=1

network={
    ssid="<network name>"
    psk="<network password>"
}
```

Note: this is a relatively insecure method, since we are writing out our network credentials in plain text. Anyone who has read access to this file can see this information. I recommend that you not use login credentials that are personal to your university account, since they are typically used to access other crucial university or employee services. Ask your network administrator or IT department to create a new generic or institutional user and password combination for such testing and tinkering experiences!

Exit nano by hitting `control + x`, and then pressing the `y` key when asked if you'd like to save the buffer (which means save the file). This should return you to the normal terminal window.

Now eject the microSD card, place it in the Raspberry Pi and boot it up. Wait a couple minutes for it to finish booting up. The red LED light specifies power consumption (solid means power is flowing, flickering means inadequate power supply, i.e. needs a supply with better ampage), and the green LED light specifies data processing. The operating system will be finished installing once the green LED calms down.

Now that we've installed the OS and enabled network-based control of it, let's go ahead and establish such a connection. This is called SSH, or Secure SHell connection. To establish an SSH connection we will need to know the location of the target computer on the network. We can do this by going into the router's settings and checking all connected computers' IP addresses, or we can run the following command on your laptop or desktop computer:

`arp -a` [more details]

Now that we know the target computer's ip address, we can type the following command, replacing the generic placeholder (including the <> characters) with the actual address:

```
ssh pi@<ip address>
```

You will be prompted for a password. The default password on fresh installations of Raspbian is 'raspberry'. No astericks will be visible as you type it, just hit enter when you are finished. Your window should look something like this:

[screenshot]

Congratulations, you're in! :tada:

In the command we just ran, 'pi' and 'raspberry' specify the default username and password on the host computer, and the host computer itself is specified by the IP address that locates it on the network.

This terminal window should now be considered a remote client to the raspberry pi. More specifically, everything that you do run that follows `pi@raspberrypi:` in the terminal window is being implemented on the Raspberry Pi. If you were to close the terminal window, you will be prompted to end the session - this is fine, but the Raspberry Pi will still be running, without any SSH connections still active. You will need to re-establish a new SSH connection to get back in control of the host computer, and to shut it down.

You may also get the following warning when trying to establish an initial connection via SSHL

[screenshot]

This is triggered because each remote computer that establishes an SSH connection with a specific target is identified using a cryptographic key, for security reasons. When you re-install Raspbian on the same microSD card, the old key slot is removed and replaced with a new one. However, when your remote laptop tries to enter the computer located on the same location on the network it reaches for the old key that works with the lock that has since been removed. We therefore need to delete the old key and replace it with a new one that fits the new lock.

To do this, navigate to the file where the keys are stored, and edit the file to delete the corresponding keys:

```
sudo nano ~/.ssh/known_hosts
```

Remove all the data stored in this file by going to the end and holding backspace, then save and exit with `control + x` and then `y`.

Now that we are in the remote host, let's install updates. This requires that the local network be connected to the internet.

```
sudo apt-get update -y
sudo apt-get upgrade -y
```

Linux software is installed and managed through the use of package managers, in this case one called apt. `apt-get update` updates the listing of software (along with version numbers and compatibility info) that is managed on a trusted and audited series of servers out on the internet. `apt-get upgrade` installs updates to outdated software on your device based on comparisons of the local version numbers with those on the external listing. Including the `-y` flag is optional, and specifies that any prompt to confirm an update with a yes/no response should always respond with 'yes'.

For the purpose of this tutorial it is not necessary to update the kernel or firmware. The most up-to-date versions are included in the raspbian disk images. Using `sudo rpi-update` may install bleeding-edge updates or untested firmware that might break our intended setup.

Note: There is no power switch on the Raspberry Pi, and power-supply can therefore be a bit finicky. **You will have to shut down the computer by running a specific command. DO NOT JUST UNPLUG THE POWER CORD! That can corrupt the SD card and you'll have to re-image it and re-install the OS, and all data will be lost in the process!**

To shut down the Raspberry Pi, use the following command:

```
sudo shutdown -h now
```

To restart the Raspberry Pi, simply re-connect the power source.

4.2.2.3 Setting up Static IP Addresses

To make our lives a little easier, especially when working on a network that many other devices will be connected to, it will be helpful to configure a static IP address for the Raspberry Pi on the network. Each device connected to the network is assigned a unique IP address, typically as a sequence following the order in which they connected. So the router might have IP address `192.168.0.100`, and subsequently connected computers will have `192.168.0.101`, `192.168.0.102`, `192.168.0.103`, etc. Most routers enable up to 200 connections by default, distributing IP addresses up to `192.168.0.199`. In order to maintain consistency when connecting to the Raspberry Pi (which may be necessary when writing instructions for others to follow, or when pre-configuring connections to the Raspberry Pi on others' computers), we can reserve an IP address from the upper end of the range, such as `192.168.0.198`. You will no longer have to lookup the target computer's IP address in the router options or via the terminal - you could simply run `ssh pi@192.168.0.198` to establish a connection to the Raspberry Pi from any remote computer on the network.

In order to do this we will have to find some information about the network configuration. Start by SSH-ing into the Raspberry Pi.

We first need to get the Raspberry Pi's IP address on the network, on both wifi and ethernet interfaces.

```
ip -4 addr show | grep global
```

The output should look something like this:

```
inet 169.254.39.157/16 brd 169.254.255.255 scope global eth0
inet 192.168.0.102/24 brd 192.168.0.255 scope global wlan0
```

`eth0` refers to the ethernet interface, and `wlan0` refers to the wireless interface. Information pertaining to the ethernet interface will not appear if the Raspberry Pi is not currently connected to the router via an ethernet cable.

Next we need to determine the router's gateway:

```
ip route | grep default | awk '{print $3}'
```

The result should look something like this:

```
192.168.0.1
```

Then we need to determine the DNS server or namespace, which is often the same as the router's gateway:

```
cat /etc/resolv.conf
```

The result should look something like this:

```
# Generated by resolvconf
nameserver 192.168.0.1
```

Finally, we have to add this retrieved information to the file located at `/etc/dhcpcd.conf`:

```
sudo nano /etc/dhcpcd.conf
```

Pro Tip: when editing system files or doing anything that reconfigures the computer setup in a serious way, you will have to include the `sudo` before the command. This grants 'super user' privileges, which are required for system adminstration.

Add the following, substituting the `static routers` and `static domain_name_servers` values with the one's that were just determined above. The values for `static ip_address` should be our new consistent IP address. For the numbers following the `/`, use the values from the IP addresses we identified above.

```
interface eth0
static ip_address = 192.168.0.197/16
static routers = 192.168.0.1
static domain_name_servers = 192.168.0.1

interface wlan0
static ip_address = 192.168.0.198/24
static routers = 192.168.0.1
static domain_name_servers = 192.168.0.1
```

Save and close the file, and you should be good to go! You might have to re-establish the connection, however, since the host may now be situated at a different location on the network.

___Pro Tip:___ You may need to make a direct connection with your computer to download updates (When in the field, I tend to tether my cullular connection to my laptop and share the connection over ethernet). In such cases, you will need to edit `dhcpcd.conf` and comment out the `eth0` specifications to enable an alternative ethernet interface with your laptop that is not specific to the router that it is normally connected to. You may not even need an `eth0` static IP, but it helps to have that direct ethernet connection in place if you are using the Pi as a network attached storage (NAS) device, i.e. sharing files over the network, as this gives a significant speed boost to file transfers.

4.2.2.4 Setting up the database management system

Now that we have the Raspberry Pi set up, let's set up a database management system (DBMS for short). In this tutorial we're going to be using MariaDB, which is for all intents and purposes identical to MySQL (MariaDB is a fork of MySQL that is more committed to open source principles). In fact, all MySQL commands work in MariaDB, and the community of users cross-polinate and use the terms synonymously in their documentation and troubleshooting. We will be using many commands that were designed for use in MySQL and which were ported over to MariaDB when it was forked.

In our current application, we will be installing a database server on the Raspberry Pi, which hosts one or more databases. The database, and all information stored within it, resides on the SD card slotted into the Raspberry Pi, which users connect to over the network as clients. The fact that the database exists in one centralized place ensures that the information accessed by clients is constantly up to date and consistent. In order to connect to the database as a client, users will have to know the server's IP address, the port that is used by MariaDB on the server, the name of the database to be accessed, and the users' login credentials to access the database.

To install MariaDB Server, first SSH into the Raspberry Pi and then use `apt-get` to install `mariadb-server` from the package manager:

```
sudo apt-get install mariadb-server
```

Once it is installed, you will have to re-configure a configuration file to enable access to the database across the network. First access the config file:

```
sudo nano /etc/mysql/mariadb.conf.d/50-server.cnf
```

Find the line that configures `bind-address` to `127.0.0.1` and change its value to `0.0.0.0`. This changes the addresses that the MariaDB server will listen to for connections. If we had left it at `127.0.0.1`, which is typically synonymous with `localhost`, only modifications made by the Raspberry Pi's local users would be able to access any of the databases managed by the MariaDB server. By changing the value to `0.0.0.0` we enable the server to accept connections from any computer on the network, if they have valid credentials. You can also configure specific IP addresses in order to allow only computers with those locations on the network to access the database server.

Once your changes are made, save the file using `control + x` and then restart the MariaDB service:

```
sudo service mysql restart
```

Now let's login to the server so that we can create a database. By default there is only one user named 'root', whose password is blank. Root has the broadest set of privileges, and can create or drop databases and users freely. You can only log in as root on the machine on which the MariaDB server resides - you can not log in as root from a remote connection. After establishing a SSH connection to the Raspberry Pi, initiate the server and login as root:

```
sudo mysql
```

You'll notice that the terminal window looks a little different. We are now in the MariaDB server environment, and we will have to escape it in order to run commands outside of it. The command to escape is `control + c`. You can also establish another SSH connection in a new terminal window so that you can use one window for database management tasks, and the other for working with other aspects of the operating system.

MariaDB commands are SQL, or Structured Query Language. They are generally written using upper case letters, and every command must end with a semicolon. The semicolon essentially indicates the boundaries of a SQL statement, which is essentially a self-contained logical statement. If you do hit enter after an incomplete statement is input then any further input will be tacked on to the end of the incomplete statement. The semicolon indicates that the statement is finished and ready to be executed.

Now let's create a database called `odate1`:

```
CREATE DATABASE odate1;
```

Next create a new user named `lara` with password `croft`:

```
CREATE USER 'lara'@'localhost' IDENTIFIED BY 'croft';
```

We can also make a restriction that would only recognize a user named lara if she attempts to access the server from a specific IP address by replacing `localhost` with the IP address that you are expecting her to connect from:

```
CREATE USER 'lara'@'12.34.56.78' IDENTIFIED BY 'croft';
```

Different users can have different permissions granted to them. Groups of specific permissions can be lumped together and labelled as roles, which can also be assigned to users, which may make database management slightly easier. In this tutorial we will not fret too much with permissions or roles, and we will simply assign all permissions to our user lara:

```
GRANT ALL PRIVILEGES ON odate1.* TO 'lara'@'%' IDENTIFIED BY 'croft';
FLUSH PRIVILEGES;
```

We could also restrict lara's permissions to specific tables on the database by replacing the `*`, which means 'any' or 'wild card' in logical or boolean terms, with the name of a specific table contained within the database.

Now let's close the MariaDB service (using `control + c`) and try logging in as lara:

```
mysql -u lara -p
```

The -p flag indicates that the terminal should prompt us for a password. As lara, if we try creating a database we will get an error because lara does not have permission to do so.

Creating or dropping tables, columns, indexes and relationships, as well as selecting, inserting, updating and deleting rows, is done via SQL. More comprehensive guides on SQL can be found at the following links, but here are a few useful snippets that you can try out.

To create a table:

```
CREATE TABLE `odate1`.`contexts` (
`id` INT(11) UNSIGNED NOT NULL AUTO_INCREMENT,
`Context` VARCHAR(255) NOT NULL DEFAULT '',
`Trench` VARCHAR(255) NOT NULL DEFAULT '',
`ContextDescription` VARCHAR(255) NOT NULL DEFAULT '',
`Munsell` VARCHAR(255) NOT NULL DEFAULT '',
`SedimentColour` VARCHAR(255) NOT NULL DEFAULT '',
`NumberOfBuckets` INT(11) NOT NULL DEFAULT '',
PRIMARY KEY (`id`)
);
```

To insert data into the table:

```
INSERT INTO `contexts` (id, Context, Trench, ContextDescription, Munsell, SedimentColour, NumberOfBucke
VALUES (1, 0001, 001, "The first context of the season. Loamy sand, very herbaceous, characteristic of a
```

to select data from the table:

```
SELECT `Context`, `Munsell`, `SedimentColour` FROM `contexts` WHERE `contexts`.`Trench` = 001;
```

To update data in the table (https://www.w3schools.com/sql/sql_update.asp):

```
UPDATE `contexts` SET `SedimentColour` = "Dark Brown" WHERE `Munsell` = "5YR 3/3";
UPDATE `contexts` SET `SedimentColour` = "Brown" WHERE `Munsell` = "5YR 4/3";
UPDATE `contexts` SET `SedimentColour` = "Brown" WHERE `Munsell` = "5YR 5/4";
UPDATE `contexts` SET `SedimentColour` = "Reddish Yellow" WHERE `Munsell` = "5YR 7/6";
```

To delete data from the table (https://www.w3schools.com/sql/sql_delete.asp):

```
DELETE FROM `contexts` WHERE `Context` = "0005";
```

To make a relationship between two tables:

```
CREATE TABLE `odate1`.`trenches` (
`id` INT(11) UNSIGNED NOT NULL AUTO_INCREMENT,
`Trench` VARCHAR(255) NOT NULL DEFAULT '',
`Supervisor` VARCHAR(255) NOT NULL DEFAULT '',
`Reason` VARCHAR(255) NOT NULL DEFAULT '',
`LocalDatum` VARCHAR(255) NOT NULL DEFAULT '',
PRIMARY KEY (`id`)
);

ALTER TABLE `contexts` ADD CONSTRAINT `FK_contexts_Trench` FOREIGN KEY (`Trench`) REFERENCES `contexts`
```

To run a join query (https://www.w3schools.com/sql/sql_join.asp):

```
SELECT `contexts`.Context, `contexts`.`Trench`, `contexts`.`NumberOfBuckets`, `Trench`.`Reason` FROM `c
```

- do all this from the command line but show how it can also be done from a GUI like Sequel Pro or SQL Workbench, which actually shows the queries being run

4.2.2.5 Client side stuff

Manipulating a database can be done from the command line or from a graphical user interface running on a remote client. There are a wide variety of database client software available (either open source or

proprietary) but I highly recommend Sequel Pro (mac only) and MySQL Workbench (cross platform).

As mentioned above, in order for a client to access a database on a remote server they will need the following information: * the server's IP address * the port on which MariaDB listens to connections * the database's name * a user's username and password

Assuming that the Raspberry Pi has the static IP we assigned to it earlier, we can access our test database using the following information: * IP address: 192.168.0.198 * Port: 3306 (this is the default for MariaDB and MySQL) * Database: odate1 * User: lara * Password: croft

Everything that happens in this interface can be expressed as a logical SQL statement, which is passed on to the server and executed there as if it were done in the terminal, as demonstrated above. In MySQL Workbench this is explicitly shown to the user so that they may verify the actions that they take.

It is also possible to generate your own interfaces and data entry forms using Microsoft Access, PHP or R. In broad terms, a connection to the database is established using the same information just listed above, and buttons next to text fields or table renderers are used to execute SQL statements that insert, update, select delete or otherwise query the database accordingly.

For more information on how create data entry forms using Microsoft Access, PHP and R Shiny, please refer to the following links: * Microsoft Access: * PHP: * R:

4.2.2.6 Scheduling automatic backups

It is wise to backup your databases regularly, but this is often overlooked. Thankfully we can automate this process! We accomplish this by running a basic script that wraps the database as a single file and copies it over to a specified directory that corresponds with a folder on a mounted USB storage drive.

First we will have to configure the system so that the USB storage device is mounted consistently and automatically each time the device is connected. While Raspberry Pi does support auto USB mounting when starting the machine, it can be a bit flaky. Therefore we will use the device ID (referred to as UUID in the linux system) to ensure that there is more certainty of a proper mount. So we need to determine the UUID of the USB device by first inserting it in the Raspberry Pi and then cross-checking the outputs of the following commands:

```
ls -al /dev/disk/by-uuid
lsblk
```

Infer the correct UUID based on the drive's capacity and through elimination of other possibilities, and note it down somewhere.

Now create the directory where we want to mount the drive:

```
sudo mkdir /media/backup
```

Give the directory adequate permissions:

```
sudo chmod -R 777 /media/backup
```

Note: This command gives unrestricted read, write and execute permissions to all users! Configure your permissions to suit your needs!

Next open the `fstab` file, which is a system file that handles the way Raspbian mounts USB devices:

```
sudo nano /etc/fstab
```

At the end of that file, add the following command, replacing the UUID with the one pertaining to the specific device and the directory with the intended mount point:

```
UUID=783A-120B /media/backup auto noatime,nofail,umask=000 0 0
```

Note: This is the correct code for FAT32 formatted drives only. FAT32 handles permissions is a slightly different way than NTFS and linux-based filesystems (i.e. ext4) and you will need to configure them accordingly. However, I recommend the use of FAT32 in general, since it is interoperable between virtually all operating systems, whereas others are not, and zipped archaeological databases probably won't be more than 4GB each, which is the file size limit for FAT32-formatted devices.

Now the USB drive will be auto mounted and ready to use. Don't forget to restart your Raspberry Pi to test it out:

```
sudo shutdown -h now
```

After you have gotten the USB drive to mount automatically, we need to create a script called `dbbackup.sh` in the home directory:

```
cd ~
sudo nano dbbackup.sh
```

In that file, enter the following:

```
#!/bin/bash
$OUTPUT_FILE = /media/backup/mysqldump/odate1_$(date +"%Y%m%d_%H%M%S").gzip
$USERNAME = lara
$PASSWORD = croft
$DATABASE_NAME = odate1

sudo mysqldump -u$USERNAME -p$PASSWORD $DATABASE_NAME | gzip > $OUTPUT_FILE
```

The username, password, database name and file path can also be modified to suit your unique situation. Ensure that the permissions are properly set to allow you to write to the specified directory.

Then we need to set the script to be executable:

```
sudo chmod 755 dbbackup.sh
```

Now we need to automate the execution of this script using `cron`. `cron` is a time-based scheduling utility in Unix/Linux operating systems such as Raspbian. It reads a configuration file, commonly referred to as `crontab` where jobs are scheduled using a special syntax. To access and edit the crontab simply enter `sudo crontab -e` and add a line at the end of the file to schedule a job.

To run dbbackup.sh every 30 minutes, add the following line at the end of the crontab:

```
*/30 * * * * /home/pi/dbbackup.sh
```

___Pro Tip:___ crontab.guru is a great resource for determining the correct syntax for cron schedules.

For more information, please refer to the following link: https://www.codingepiphany.com/2013/06/12/raspberry-pi-lamp-server-tuning-and-automated-db-backup/

4.2.2.7 Configuring a file sharing server

It may be useful for participants on a project to assemmble photos, spreadsheets, reports or various other digital files with minimal hassle. Luckily, the Raspberry Pi can also be used to set up a network attached storage (NAS) device. In simpler terms, network-connected users can transfer files between their own computers and a USB storage device connected to the Raspberry Pi. This effectively renders a USB storage drive as a wireless drive.

To begin, the system must be confured to automount the USB drive. Instructions on how to do this have been provided in section 4.2.2.5. I recommend that you use a separate USB drive than the one used for backups. For the purpose of this tutorial, I will assume that the file sharing drive is mounted at `/media/FileShare`.

We will be using a software package called Samba to enable the primary NAS functions. Install Samba using the following command:

```
sudo apt-get install samba samba-common-bin
```

Next, edit the Samba configuration file:

```
sudo nano /etc/samba/smb.conf
```

Add the following at the end of the file:

```
[FileShare]
Comment = Pi shared folder
Path = /media/FileShare
Browseable = yes
Writeable = Yes
only guest = no
create mask = 0777
directory mask = 0777
Public = yes
Guest ok = yes
```

Ensure that `Path =` the directory you want to share, such as the mount point of an external drive.

This configuration allows anyone to read, write, and execute files in a volume named share, either by logging in as a Samba user or as a guest. If you do not want to allow guest users, omit the `guest ok = yes` line.

Please refer to the following link and the Samba documentation in order to set up usernames and passwords. The method just described allows free-for-all access, and it should be communicated to users that they can add, delete or modify any file in the shared directory, for better or worse. You may feel inclined to schedule automated backups of your shared drive to yet another connected drive, though this may congest the network and occupy lots of disk space, depending on the size of the shared directory. You may wish to schedule such backups during off-hours, when volumes of traffic across the network are expected to be low. * * *

4.2.2.8 Tinkering and Improvising in the Field

- working in multiple locations without any internet connection (and sqldump workaround)
- ensure that people, especially higher up, understand the constraints and limitations you face
- show people rather than tell
- be respectful of their data and devices
- power outages and working off-grid
- migrating to the cloud and back
- keep notes, watch your posture, etc

4.2.2.9 Discussion

4.2.2.10 Takeaways

4.2.2.11 Excercises

4.2.2.12 Further Reading

https://www.raspberrypi.org/documentation/linux/usage/commands.md

4.3 Place-based Interpretation with Locative Augmented Reality

Augmented reality is the use of various technologies to overlay data onto one's visual or aural perception of the environment. Augmented reality does not need to be a digital technology - it can be as simple as a sheet of plexiglass/perspex framing a view of an archaeological site, etched so that the 'missing' structure(s) can be seen, as at Carnuntum in Austria.

Figure 4.4: Carnuntum

(Image Wikimedia Commons, by user Gryffindor)

Augmented reality does not need to be a visual technology, either.

What AR could be - the following two videos were winning entries in the HeritageJam at York University in recent years. They are prototypes for how AR might be used archaeologically.

https://youtu.be/wAdbynt4gyw

https://youtu.be/Y_yE-TkDn5I

4.3.1 Projection Mapping

Projection mapping is a kind of AR that uses digital video projectors and specialized software on your computer to block out areas of light so that the different areas show different videos or images at the same time. Instead of a rectangle of light, one can 'map' the boundaries of the projection areas to match the architecture of a building facade (for example). Other terms for this include 'spatial augmented reality' or 'video mapping'.

The artist Craig Winslow used the technology to bring the faded historic signage of Winnipeg's downtown back to life.

Get started:

Optoma Projection Mapper(5$ android, ios app) http://projection-mapping.org/tools/optoma-projection-mapper/

Mappmap: https://mapmapteam.github.io/

Hololens, ARkit, all the rest of it.

.

4.3.2 exercises

1. The best introduction to producing virtual reality is currently Jacob Greene's Creating Mobile Augmented Reality Experiences in Unity at the Programming Historian. In this tutorial, he demonstrates how to use the built in features of the Unity 3d game engine to create smartphone AR applications. Complete the tutorial, and then extend the application by adding 3d models (there are creative-commons licensed models at Sketchfab in their 'cultural heritage' category) to it.
2. The interactive fiction engined Twine can be hacked to create webpages that only display certain text passages or audio or images if the user is standing in a particular location. See the Twine Cookbook's 'Geolocation' section. A walk through of using this feature is here. Play with this to plan out a hand-held guide to your campus.

4.4 Artificial Intelligence in Digital Archaeology

To speak of 'artificial intelligence' in archaeology may be to speak too soon yet. We do have machine learning in the service of archaeology (neural networks for classificatory purposes, for instance), and there is a well-established body of work in terms of simulation that could fall under the rubric of 'artificial intelligence'.

Then why use the term? We think it is still useful to use the term because it reminds us that, in using computational power for simulation or image classification we are off-loading some of our expertise and abilities to a non-human actor. In the case of machine learning and neural networks, we really *can't* see inside the 'black box'. But we can examine the training data, for it is in the selection of training data that we introduce biases or agendas into the computation. By thinking of the machine in this case as something non-human, our hope is that we remind you to not accept the results or methods of AI in archaeology blindly, as if the machine was not capable of racist or colonialist results.

Machine learning: a series of techniques that endeavour to train a computer program to identify and classify data according to some previously determined values. We will in this chapter discuss image recognition using the neural network model trained by Google, the Inception3 model, which we will query using the Tensorflow package.

Agent based simulation: a series of techniques that create a population of software 'agents' who are programmed with contextual rules (e.g., *if this happens, do that*) governing the behaviour of individual agents. The context can be both in terms of the simulated environment (GIS data, for instance) or the social environment (social relationships as a network). Simulation experiments iterate over multiple combinations of parameters' values, the 'behaviour space'. Simulation results are then used by the investigator to explain the 'real world' phenomenon as an emergency of a population of agents following a set of rules under certain situations.

The value of machine learning: it makes us think carefully about what we are looking for and at in the first place; and then it can be scaled massively.

The value of agent-based modeling: it forces us to think carefully about what it is we think actually *happened* in the past such that it can be expressed as a series of contextual rules of behavior, functioning under certain conditions.

4.4.1 Agent-based modeling (ABM)

This section is an overview of agent-based modeling (***ABM***), as it is used within archaeology, anthropology, and behavioral sciences. Before tackling the "agent-based", we start with a brief introduction to what "modeling" means. To place ABM in a bigger picture, we introduce the reader to several dimensions of the

variability of simulation models. We describe the necessary and optional parts of ABM models, including their characteristic type of element, i.e. the agent. At this point, the student must be aware that ***simulation models*** in our context involve computer code (they are *digital*) and the iteration of processes (they are *dynamic*).

Following the definition of ABM, we describe the process of creating a model or *formalizing* an informal model. Although the reality is messier, this process can be divided into *definition* of research question and phenomenon, *design*, *implementation*, and *verification*. *Validation*, which is considered a fifth step by most of the modeling literature, is regarded here as an extra, lengthy process, often separable from the tasks involved in creating and exploring a model. Furthermore, we introduce two additional steps that are not normally acknowledged: *understanding* and *documenting* the model. In explaining all modeling steps, we keep instructions general enough for them to be valid when dealing with any platform or programming language.

Next, we cover the *why* and *how* of the application of ABM in archaeology, history, and social sciences. We discuss some key foundational works, including models that predate the development of ABM as it defined today. Specifically regarding archaeological applications, we illustrate the diversity of ABM approaches by presenting several examples of models with varying goals, spatial and temporal scales, and theoretical backgrounds.

Last, we walkthrough the steps involved creating an ABM model. Inspired in the theme of emergence and collapse of territorial integration, we designed a model, the ***Pond Trade model***, showcasing many elements that are common in ABM models in archaeology. The coupled process of design and programming was intentionally broken down into smaller, progressive steps to illustrate an organized pace of code development, from simple to increasingly complex algorithms. The Pond Trade model is implemented in ***NetLogo*** (https://ccl.northwestern.edu/netlogo/), which is free, easy to install and use, and widely used in academia. The exercise assumes no previous programming knowledge, though practice with NetLogo is required to fully undertand the later versions of the model. (A Jupyter Notebook that uses an agent based model written in Python as the basis for an exploration of computational creativity can be found in the 'Worldbuilding' subfolder here; please also see Ben Davis' Agent-based modelling prehistoric landscape use with R and NetLogo).

Throughout this section, several concepts will probably sound *alien* for most history and archaeology students; and some may remain obscure long after completing this lesson. Beyond any practical application, ABM has strong roots in mathematics and logic. ***Don't panic!*** Most ABM modelers don't go through formal introductions nor are well-versed in algebra and calculus. As in most digital approaches in humanities and social sciences, ABM is mostly done by self-taught experts with hybrid and tortuous academic profiles. Also, because ABM modellers in archaeology are not often computer scientists, the community lacks conventions about how to communicate models and simulation results, though proposals do exist (**REF ODD**). In this sense, the content of this section should NOT be considered the standard among the ABM-archaeology community.

To the date, there is no easy way to begin doing ABM. We encourage students to engage in modelling the sooner, the better, by following our exercises, other tutorials, or their interests and creative anxieties. The full comprehension of ABM will require years of practice and transdisciplinary readings; certainly, a greater mastery in programming. Despite the rarity of ABM in history and archaeology mainstream curricula, there are many publications that offer introductions to ABM in archaeology, writen by authors with different backgrounds and perspectives (e.g., Breitenecker, Bicher, and Wurzer (2015), Cegielski and Rogers (2016), Romanowska (2015); also visit *The ABM in Archaeology Bibliography*, for an extensive and constantly updated list of references).

4.4.1.1 What is ABM?

> "Essentially, all models are wrong, but some are useful"
> George Box, 1987
> Box and Draper (1987), p. 424

The first thing to consider is that ABM is about models. A model is a representation of a phenomenon as a set of elements and their relationships; the key term being *representation*. It is a generalization/abstraction/simplification of what we consider as *THE* phenomenon. Think about a pineapple. The mental representation of the phenomenon, "the pineapple", is already a model. It consists of a set of traits or visual cues, and their relative positions (e.g., crown of spicky leafs on top, thorns covering the oval body). Empirically, every pineapple has different visual properties, but we still are able to identify any pineapple we encounter by using this model.

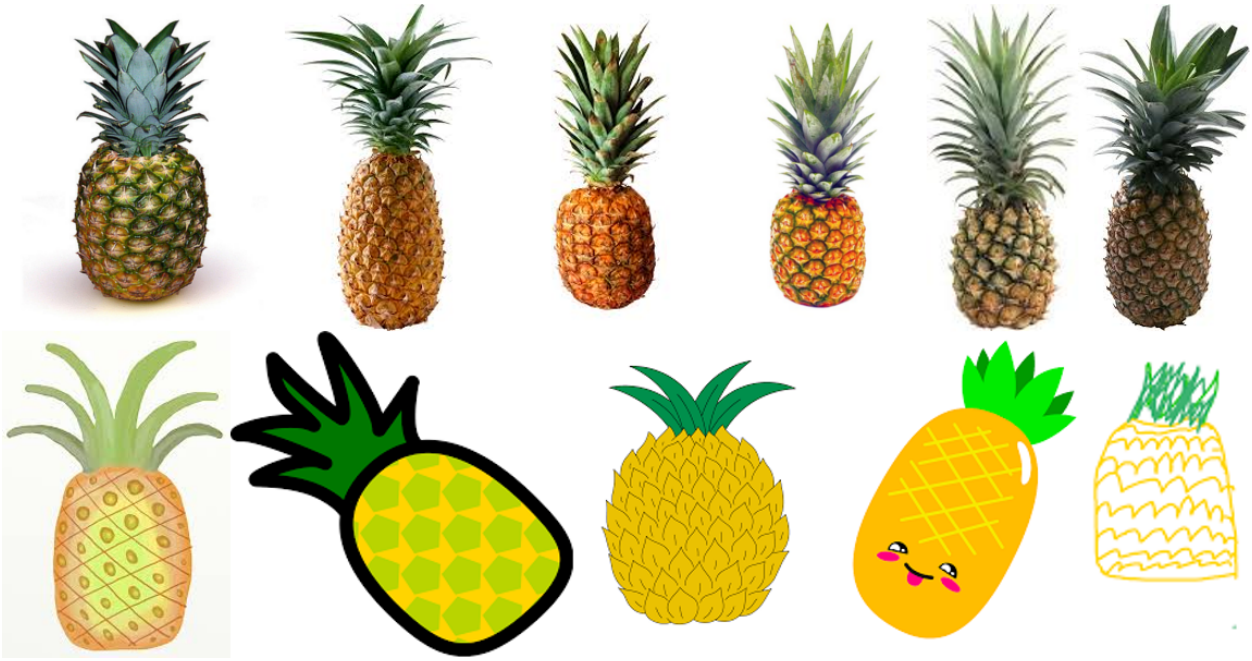


Figure 4.5: Pineapples and their (visual) models

Another important statement about models is that they are–as any mental process–a phenomenon in its own right. That puts us in the realm of epistemology and cognitive sciences. It follows that one's model is *a* model and that there are many models representing a phenomenon as there are minds that recognise this phenomenon. Observe the diversity of pineapple drawings. Adding further complexity to this picture, minds are constantly changing with the integration of new experiences–creating, modifying, merging, and forgetting models–and we often exchange and contrast alternative models through expressing them to ourselves and others. As the last straw of confusion, the ability for communicating models give them the status of *'material objects'*, with a *existence* of their own, reproduced through learning, checked by the clash of new experiences, and in perpetual change through innovation and creativity. For example, a child with little or no experience with the 'real pineapple' is able to learn–and graciously modify–a model of pineapple.

Well, what do pineapple drawings have in common with doing models in archaeology? You may imagine that, if your mental representation of a pineapple is a model, human minds are bursting with models about everything, including *models of models*, *models within models*, and *models of ourselves*. If an archaeologist, there will be many archaeologically-sensitive models buried deep inside such mental apparatus. *"Why did people in this site buried food remains in small pits?" "They wanted to avoid attracting critters." "They did not stand the smell of decomposition." "One person with high status started doing it and them the rest just followed." "They were offering sustainance to the land spirit."* These explanations derive directly from models of *why people behave as they do*. Often tagged as *social* or *behavioral*, this kind of model is as key in archaeology as it is neglected. Given that archaeological practice orbits around materials, archaeologists tend to relegate social models to a second plane, as if the goal is actually to understand the material remains of human behavior rather than the behavior itself. More importantly, many archaeologists are hardly cognizant that they are using social models even when being avowedly 'non-theoretical'.

Another common practice is to present authoritative models as mantras while unconsciously hidding the models used. Social models are indeed more permeable to *personal*, *subjective* perspectives (in contrast with a model of stratification, for example) because we use them daily to interact with others, apart from archaeological affairs. Thus, they are strongly moulded to our life experiences and beliefs, which vary depending on many socio-cultural factors, including those less obvious such as age and gender. Even though there are shared traits between them, these models are as many and diverse as pineapple drawings.

Why bother then doing formalized models in our research? Ultimately, we are searching for knowledge, statements with at least some *value of truth*, independent of any individual. Remember that models of a phenomena, as diverse they may be, are contrasted with experiences of that phenomena, which are much less variable. As George Box's famous phrase states, models are NOT equivalent reproductions of a phenomenon, but tools for looking and interacting with it. What we could define as *imaginative models*, such as the smiling pineapple, though still able to exist for other purposes, can be clearly distinguished from models with *some degree of truth*, because they fail to match our sensible experience and are incompatible with other successful models. Those that cannot be distinguished, should be considered *valid*, *competing* models; their study, validation, and improvement being the main goal of scientific endeavours.

Unfortunately, models involving human behavior are less straightforward to validate or dismiss than a "smiley pineapple". Still, when having many competing models, progress can be made by ordering them by 'probability of being true', following many criteria. A warning though: almost-certain models might be eventually proven false, and marginal (even discredited) models can end up being mainstream. To be alive (read, *useful*) models must be communicated, used, re-used, abused, recycled, hacked, and broken. A well-documented model, preserved in many locations and formats, could as easily be considered *hibernating*, awaiting new evidence for validation or a new generation of scholarship.

Models are not only inevitable for any living mind, but they are also the keystone for the generation of knowledge. However, if models can be so simple, spontaneous, and intuitive as a pineapple drawing, what is the need of doing strange models with complicated equations, painstaking specifications, and fathomless designs? Archaeologist use models to define their research agenda and guide their interpretations. A lot of ink has been spilled in presenting, discussing, and revisiting archaeological and archaeologicaly-relevant models. Nonetheless, few of these works are able to define unambiguosly the model(s) in question, not because they lack in effort but due to the intrinsic limitation of the media objectifying the model: human, natural, verbal language.

For example, we may write a book on how we believe, given known evidence, that kinship relates to property rights among, say, the Armenians contemporaneous to the Early Roman Empire. Our reasoning and description of possible factors can be lengthy and impeccable; however, we are bound to be misunderstood at some point, because readers may not share the same concepts and certainly not all connotations that we may have wanted to convey. A significant part of the model in our minds would be writen between the lines or not at all. Depending on their background, readers can potentially miss altogether that we are referring to *a model*, and consider the elements of our model as loose speculations or, worse, as given facts.

Formal models, or rather *formalizing* informal models, reduce ambiguity, setting aside part of the risk of being misunderstood. In a formal model everything must be defined, even when the elements of our model are still abstract and general. Our imaginary book about ancient Armenians could be complemented with one or several formal models that crystalize the main features of our informal model(s). For instance, if our informal model about property inheritance includes the nuclear patriarchal family as the main kinship structure, we must define it in *null* terms, explaining to ourselves and to others what "family", "nuclear", and "patriarchal" means, at least in the context of our model. *Does a model assume that family implies co-habitation? Does our model consider adopted members? What happens if a spouse dies?* As it is often the case, we may end up changing our informal model through the effort of formalization; by realizing, for example, that we can replace "patriarchal" with "patrilineal" because our model does not assume a particular power structure within families.

Formalization often demands the use of formal logic and to some extent, quantification. Seen from the perspective of the human natural language, mathematics and logic contain lots of repetitions, fixed vocabulary, and no nuances. Nevertheless, computational systems function strictly with this kind of language, and they

are both empowered and limited by it. A formal definition is a disclaimer saying: *"This is what X means in this model, no more, no less. You can now criticize it at will."* Whitout the formal definition, we would probably spend a chapter reviewing everything we read and thought about "nuclear patriarchal family" and still would not be "at the same page" with all our readers. The terms of this definition are assumptions of the model. At any point, new evidence or insight can suggest us that an assumption is unjustified or unnecessary, and we may change it or remove it from the model. It is in the elaboration and exploration of the unintended consequences of these rigorous definitions that the power of simulation emerges.

SG: at this point, send the reader to a jupyter notebook with a simple ABM in python in it, also point to more complicated tutorials elsewhere.

- 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 passing of time as the driving variable of a model. Other kinds of models may share the goal of inferring new data from a given knowledge or dataset. For instance, we can design a curve that fits the number of archaeological sites in a region for all known periods, and use it to estimate this value during periods not represented in any sites. However, this equation is not representing the mechanisms that are supposely causing the phenomenon, i.e. the distribution of human activity in space and time. The idea of step, iteration, or repetition as the representation of passing time is the keystone of any simulation model.
- Equation-based modeling, System Dynamics - ODE systems as models (mention Lorenz attractor, end with the idea of iteration of a process)
- Algorithm-based modeling, Algorithms as behavioral models (mention cellular automata, Van Neumann, Schelling, etc)
- 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 (multiple processes are iterated contextually by multiple entities).

b. Schedule: spectrum between fully-scheduled (processes are initiated in a predetermined order) and event-driven (processes are initiated given a certain input).
ABMs fall somewhere in-between these two spectra, though normally leaning towards distributed (e.g., using Monte Carlo methods) and fully-scheduled. Applications in several disciplines (see Wikipedia)

- Parts of an ABM model (re-visit presentations)

4.4.1.2 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). mention paradox Occam's razor vs. emergence
- 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 several times before jumping into the next step. The end result of development and exploration of a model may be relatively stable, often in the form of publications. However, this product potentially feeds a new modeling effort (and that is actually the main usefulness of models in the long run).
- ABM in python, Java, C#, or C++? in R? in NetLogo? (Any language can do it, really). Pros and cons, talk in terms of higher/lower level programming languages (still, they are all quite high-level).
- remember to talk about https://www.openabm.org/

4.4.1.3 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

4.4.1.4 ABM tutorial: the Pond Trade model

brief overview disclaimer: MY modelling strategy is highly focused in theory-building (general, explorative), not hypothesis-testing and prediction (specific, data-driven).

4.4.1.4.1 Phenomena of interest and conceptual model

- phenomena: cycles of growth and collapse, i.e. flutuations in the scale of site occupation, out of phase with environmental change. Focus on coastal settlements around a water body (e.g., lake, bay, sea).
- Belief or main assumption: topography, transport technology, trade, settlement size and wealth, and cultural exchange are intertwined.

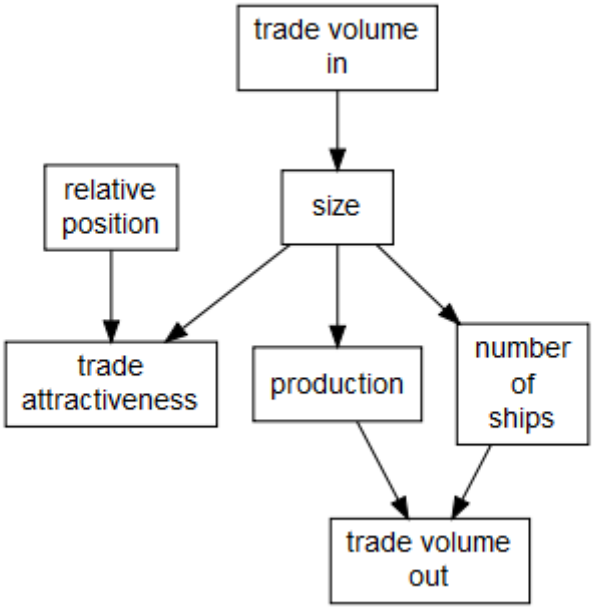


Figure 4.6: andros-first-diagram

```
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 greater cost-benefit ratio.
- 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.

4.4.1.4.2 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

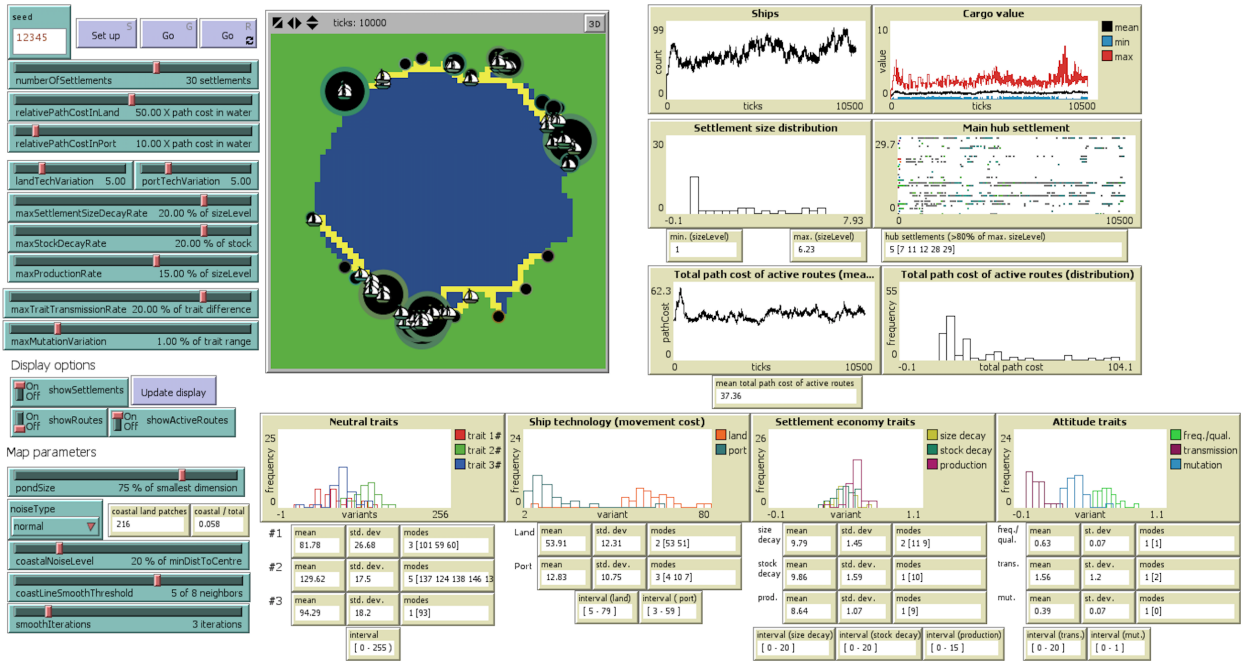


Figure 4.7: The model interface in step 13

4.4.1.4.3 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

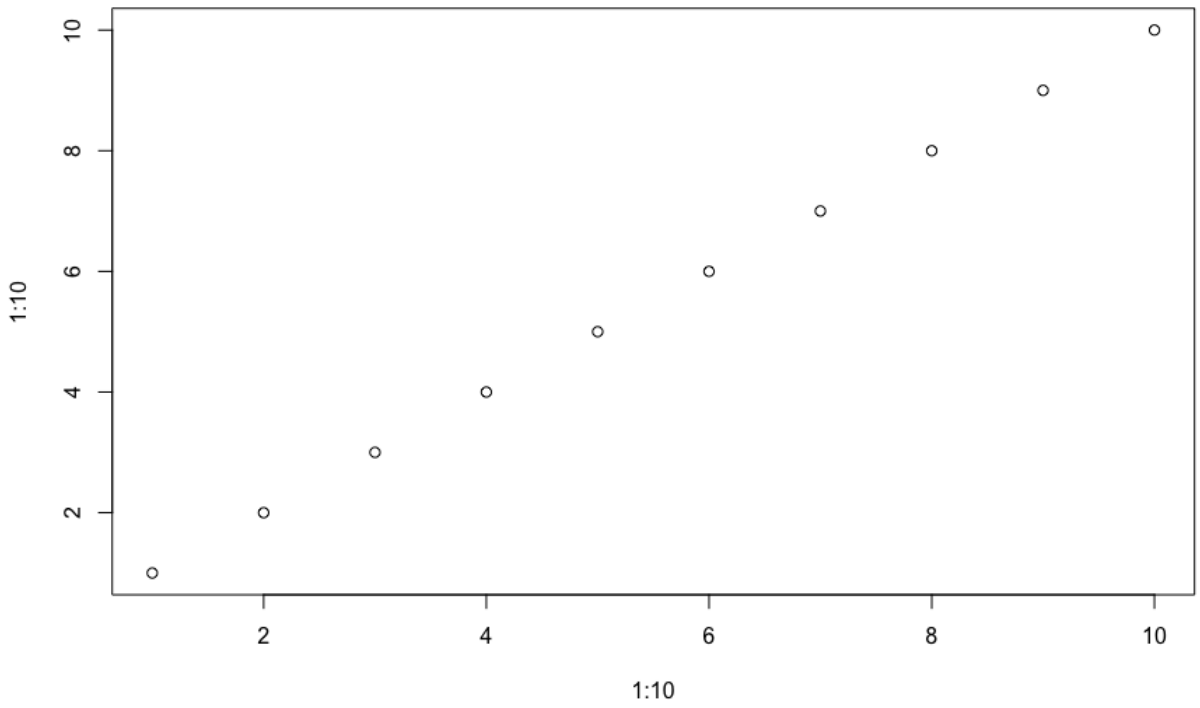


Figure 4.8: andros-second-diagram

4.5 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.

4.5.1 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.

4.5.2 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). Huffer and Graham (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.

These technologies are rapidly being put to uses that we regard as deeply unethical. Amazon, for instance, has a facial recognition service called 'Rekognition' that it has been trying to sell to police services (Winfield, n.d.), which can be considered a kind of digital 'carding' or profiling by race. In China, massive automated computer vision is deployed to keep control over minority populations ("China Has Turned Xinjiang into a Police State Like No Other" 2018). Various software companies promise to identify 'ethnicity' or 'race' from store security camera footage, in order to increase sales (and a quick search of the internet will find them for you). In Graham and Huffer's Bone Trade project, one possible mooted outcome is to use computer vision to determine descendent communities to which belong the human bones being trade online. Given that many of these bones probably were removed from graves in the first place to 'prove' deplorable theories on race (see Redman (2016) on the origins) such a use of computer vision runs the risk of re-creating the sins of the past.

Before deploying computer vision in the service of archaeology, or indeed, any technology, one should always ask how the technology could be abused: who could this hurt?

4.5.3 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.
3. If you are feeling adventurous, explore Matt Harris' signboardr, an R package that uses computer vision to identify and extract text from archaeological photos containing a sign board, and then putting that data into the metadata of the photos. Harris' code is a good example of the power of R and computer vision for automating what would otherwise be time consuming.

Chapter 5

Digital Archaeology's Place in the World

Whether or not digital archaeology consitutes, in your view, a distinct subfield, we need to consider the ways it is complicit or resists the dominant digital platforms that permeate our lives. The current digital ecosystem requires that people's behaviour online (their clicks, their likes, their follows, their browsing) by monetized (and weaponized). Archaeology - and the popular conception of archaeology promulgated by so-called 'alternative archaeologies' - generates a lot of content, hence clicks, and thus money for Facebook, Amazon, Google.

How do we, as digital archaeologists, deal with this? Subvert it? Capitalize on it?

Or is this none of our business, not our worry?

5.1 Marketing Digital Archaeology

Digital archaeology exists as both a scholarly pursuit but also a 'tactical term'. We mean this in the way that Matthew Kirschenbaum described the Digital Humanities in his contribution to the 2012 edition of the Debates in the Digital Humanities. He writes,

> To assert that digital humanities is a "tactical" coinage is not simply to indulge in neopragmatic relativism. Rather, it is to insist on the reality of circumstances in which it is unabashedly deployed to get things done—"things" that might include getting a faculty line or funding a staff position, establishing a curriculum, revamping a lab, or launching a center. At a moment when the academy in general and the humanities in particular are the objects of massive and wrenching changes, digital humanities emerges as a rare vector for jujitsu, simultaneously serving to position the humanities at the very forefront of certain value-laden agendas—entrepreneurship, openness and public engagement, future-oriented thinking, collaboration, interdisciplinarity, big data, industry tie-ins, and distance or distributed education—while at the same time allowing for various forms of intrainstitutional mobility as new courses are approved, new colleagues are hired, new resources are allotted, and old resources are reallocated.

We live in a current moment where, to get things done, we have to deploy terms in ways that capture the imagination of decision makers *and the public* in ways that affect change. In a sense, it is a kind of marketing. But it is worth thinking about the ways digital archaeology fits into the frameworks of public archaeology as discussed in Moshenska (2017). In particular, we are thinking of the ways in which the public form their views of archaeology. The work of academic archaeologists is not the primary vector through which the public learns about archaeology. How then can we deploy our work in ways that we *infiltrate* the places we have hitherto abandoned? Some time ago, Graham argued on the basis of scraping the network of links

surrounding the so-called 'blogosphere' related to archaeology that we were 'teaching' the search engines what was important, what constituted archaeology. That was in 2011. By 2014, he was no longer so certain, and again missing the boat by a few years, argued that archaeologists needed to engage with writing the Wikipedia (Graham 2015). Ironically enough, a few years later still it would not have been unreasonable to argue that Twitter and Facebook should be the locus for our marketing of archaeology, but with the rise of bad-faith actors whose concerted gaming of the algorithms of social media are working to undermine basic foundations of trust, we arrive at a moment when Wikipedia is our last online bastion of common ground (Madrigal 2018).

Does that mean we should *not* engage with Twitter and Facebook in terms of the *marketing* or knowledge mobilization that we do? Of course not. But it does mean that we can't naively put materials there and expect them to have any real impact. 'Marketing' implies a budget, it implies active engagement, and it implies working to understand the arms-race that this advertising-powered web has created. If there is a business school at your institution, have you ever taken a course there? Have you ever tried to buy advertising on google, and mount an ad-words campaign?

This is related to the discussion of the economics of public archaeology, which Burtenshaw explores CITATION. When we consider the economics of *digital* archaeology, we are confronted with the hard question of how do we measure its value? In the physical world, we can consider that archaeology provides value through:

- tourism
- urban regeneration
- direct sale of material (antiquities trading)
- marketing and branding
- jobs created by archaeological research and conservation.
- Burtenshaw (2017), 37

These create direct impact in terms of monies spent, and indirect in terms of this money being re-circulated by the initial suppliers. Burtenshaw suggests that we should understand the economic impact of public archaeology by looking at its 'magnitude, multiplication and distribution within a certain area' Burtenshaw (2017), 38. For digital archaeology, this directly ties back to the tactical usage of the term, because when we deploy it tactically, we are implicitly making an argument about economic value and economic impact. In online advertising, such things are measurement by 'engagement' or click-bait. Indeed, 'clickbait archaeology' (see e.g., this thread by Erin Thompson) can be considered archaeology done - or promoted - with the express purpose of monetizing outrage in some register because people are more likely to click on negatively or outrageously framed stories (eg Hensinger, Flaounas, and Cristianini (2013), Maldonado (2016)). Thus, projects that use a digital aspect tied into a current high-profile issue (such as eg the destruction of cultural heritage by terrorists) as a way of generating traffic to a website _for the express purpose of increasing the profile of the organization rather than the scholarly dimension of the work is actually undermining the archaeology. (Some research makes a connection between how such stories make the reader *feel* for whether or not something will achieve 'virality' or wide-spread sharing, Guerini and Staiano (2015)). Maldonado has an excellent blog post on this matter.

The marketing of archaeology online is largely a function of the economic value to be gained through engagement with the ad-based ecosystem created by Facebook, Google, and to a lesser extent, Twitter. The economic value of traffic to archaeological websites is not yet mapped out, but the role of archaeological click-bait in generating value for dubious purposes (see also Kristina Killgrove's excellent 'Don't share that Daily Mail Link about Archaeology... Just Don't') is only starting to be explored. That 'digital archaeology' as a term can be deployed tactically for strategic purposes *within* the academy is also a kind of marketing. The question is: what kind of 'digital archaeology' do *you* want to see out in the world?

5.1.1 exercises

The Documenting the Now project is a collaborative project that is developing tools to enable researches and activists to keep track of developing issues as they play out over social media. One of these tools is TWARC -

'twitter archiving' tool. Its repository is here. *The instructions below were first published in our workbook for crafting digital history.*

NB Twitter is changing how it approves and manages this process. The steps below might well be out of date, so proceed with caution. We will update once we see what the new process is, and whether it is worth the bother.

1. First of all, you need to set up a Twitter account, if you haven't already got one. Do so, but make sure to minimize any personal information that is exposed. For instance, don't make your handle the same as your real name.
2. Turn off geolocation. Do not give your actual location in the profile.
3. View the settings, and make sure all of the privacy settings are dialed down. For the time being, you *do* have to associate a cell phone number with your account. You can delete that once you've done the next step.
4. Go to the Twitter apps page and click on new app.
5. On the new **application** page, just give your app a name like my-twarc or similar. For website, use the site URL for your university or institution (although for our purposes any website will do). You don't need to fill in any of the rest of the fields.
6. Continue on to the next page (tick off the box saying you've read the developer code of behaviour). This next page shows you all the details about your new application.
7. Click on the 'Keys and Access Tokens' tab.
8. Copy the consumer key, the consumer secret to a text file.
9. Click on the 'create access tokens' button at the bottom of the page. This generates an access token and an access secret.
10. Copy those to your text file, save it. **Do not put this file in a github repo or similarly leave it online anywhere**.

Now, right click on this link to launch the binder in a new page; we've pre-installed TWARC for you:

WARNING this binder is not secure; it is possible that someone might be able to intercept. That said, the session is destroyed after your use and so the liklihood of someone intercepting is vanishingly small.

11. On the right hand side of the screen, select 'new >> terminal'. At the `$` prompt type `twarc configure`, give it the information it asks for (your consumer secret etc) and follow the prompts.

You're now ready to search. For instance, `$ twarc search canada150 > search.jsonl` will search Twitter for posts using the canada150 hashtag.

Wait! Don't run that command! (Force-stop the search if necessary by hitting ctrl+c.)

If you search for canada150 , there are, what, 36 million Canadians? How many tweets is that likely to be? Quite a lot — and the command will run quietly for days grabbing that information, writing it to file, and you'll be sitting looking at the screen wondering if anything is happening.

Try something smaller and more contained for now: `$ twarc search archaeogaming > search.jsonl`.

Note that Twitter only gives access to the last two weeks or so via search. For grabbing the stream as an event happens you'd use the twarc stream command — see the Twarc documentation for more.

It might take some time for the search to happen. You can always force-stop the search by hitting ctrl+c. If you do that though there could be an error in the text formatting of the file which will throw an error when you open it with a tool expecting perfectly formatted json. You can still open the JSON in a text editor though, but you will have to go to the end of the file and fix the formatting.

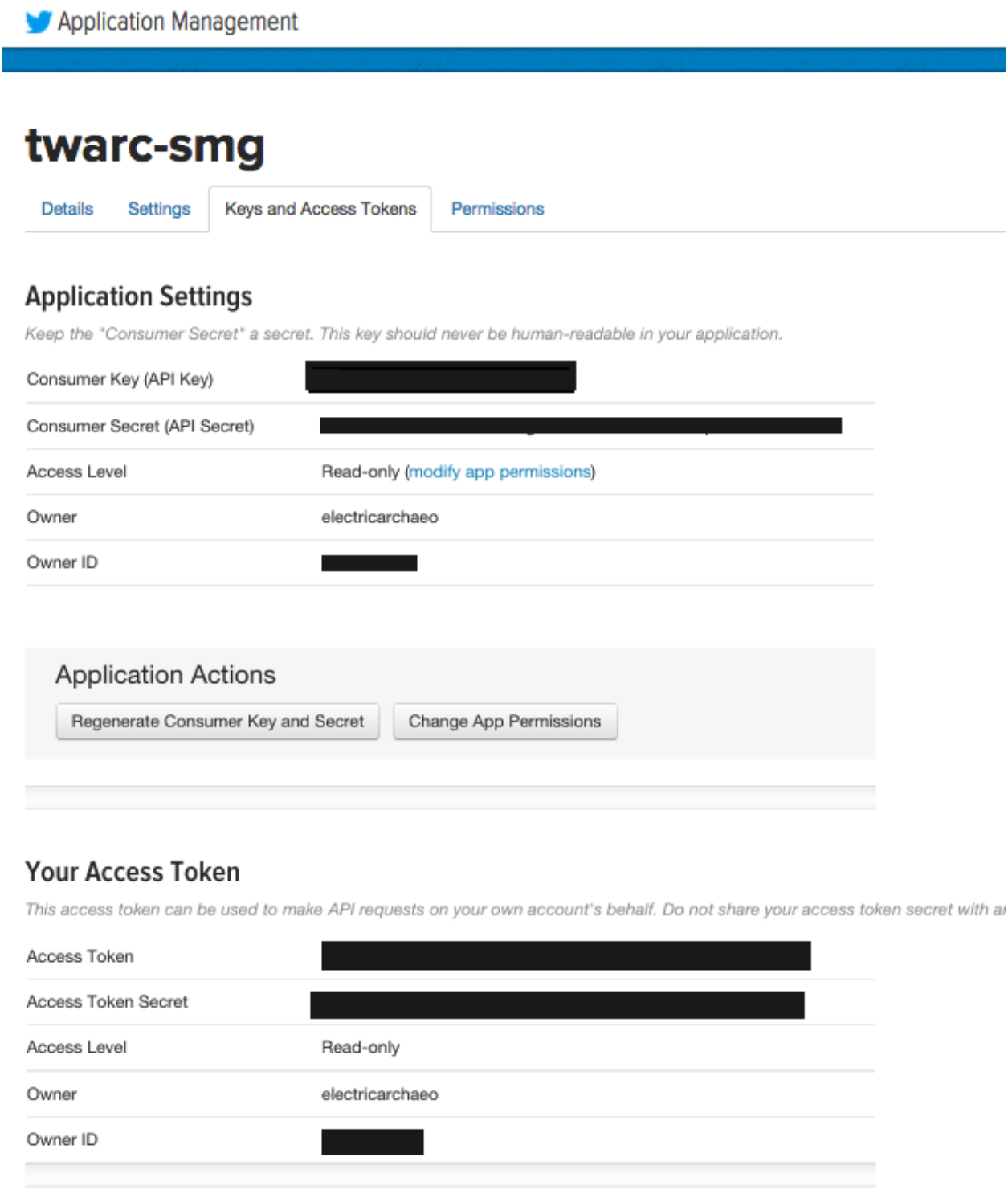


Figure 5.1: Twitter Application Management page as of August 2018

The data being collected is in JSON format. That is, a list of 'keys' and 'values'. This is a handy format for computers, and some data visualization platforms require data in this format.

On the home screen for the binder, you can download the json file to your computer. There are some utilities that come with TWARC that allow you to build visualizations of the data. To get these utilities, at the terminal in the binder, use `git clone` to get the source for TWARC:

```
git clone https://github.com/DocNow/twarc.git
```

Then, you can do things like build a tweet wall:

```
python twarc/utils/wall.py tweets.jsonl > tweets.html
```

or a wordcloud:

```
python twarc/utils/wordcloud.py tweets.jsonl > wordcloud.html
```

You can use the `network.py` command to get the tweets organized for network analysis in a program such as Gephi like so:

```
utils/network.py --users tweets.jsonl tweets.gexf
```

... which would enable you to identify influential accounts and so on. Consult the TWARC documentation for other things you can do with this package.

5.2 Sustainability & Power in Digital Archaeology

This section is currently under development. Check back soon!

5.2.1 discussion

5.2.2 exercises

Chapter 6

On the Horizons: Where Digital Archaeology Might Go Next

As an undergraduate in the mid 1990s, Graham was asked to go on the 'world wide web' to develop an annotated bibliography of websites concerning the Etruscans. After being disappointed by the results of a search on Alta Vista, he wrote: "this so-called the world wide web will never be of use for academics".

Our ability to predict the future is thus suspect.

Given that historical lack of vision, we are hesitant to proclaim where digital archaeology might go next. So why don't you tell us? Annotate this page, these words, with links to your own digital archaeology projects and explorations. Help us improve and expand this text. Collaborate with us.

Where digital archaeology might go next depends on *you*.

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.

Anderson, David G., Thaddeus G. Bissett, Stephen J. Yerka, Joshua J. Wells, Eric C. Kansa, Sarah W. Kansa, Kelsey Noack Myers, R. Carl DeMuth, and Devin A. White. 2017. "Sea-Level Rise and Archaeological Site Destruction: An Example from the Southeastern United States Using Dinaa (Digital Index of North American Archaeology)." *Sea-Level Rise and Archaeological Site Destruction: An Example from the Southeastern United States Using DINAA (Digital Index of North American Archaeology)*. https://doi.org/10.1371/journal.pone.0188142.

Andrienko, G., N. Andrienko, P. Jankowski, M.-J. Kraak, D. Keim, A. M. MacEachren, and S. Wrobel. 2007. "Geovisual Analytics for Spatial Decision Support. Setting the Research Agenda." *International Journal of Geographical Information Science* 21 (8): 839–57.

Anna Esther Heckadon, Kayla Hartemink, Kaylynne Sparks. 2018. "Interactive Mapping of Archaeological Sites in Victoria." *Interactive Mapping of Archaeological Sites in Victoria* 2.

Aycock, John, and Andrew Reinhard. 2017. "Copy Protection in Jet Set Willy: Developing Methodology for Retrogame Archaeology." *Internet Archaeology* 45. https://doi.org/10.11141/ia.45.2.

Baumann, Ryan. 2015. "Qualitative Photogrammetry Comparisons Gallery." /etc (blog). https://ryanfb.github.io/etc/2015/07/27/qualitative_photogrammetry_comparisons_gallery.html.

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.

Box, George EP, and Norman R Draper. 1987. *Empirical Model-Building and Response Surfaces*. John Wiley; Sons.

Breitenecker, Felix, Martin Bicher, and Gabriel Wurzer. 2015. "Agent-Based Simulation in Archaeology: A Characterization." In *Agent-Based Modeling and Simulation in Archaeology*, edited by Gabriel Wurzer, Kerstin Kowarik, and Hans Reschreiter, 53–76. Springer, Cham. doi:10.1007/978-3-319-00008-4_3.

Burtenshaw, Paul. 2017. "Economics in Public Archaeology." In *Key Concepts in Public Archaeology*, edited by Gabriel Moshenska, 31–42. UCL Press. http://discovery.ucl.ac.uk/1574530/1/Key-Concepts-in-Public-Archaeology.pdf.

Caraher, William. 2012. "Archaeological Glitch Art." *The Archaeology of the Mediterranean World*.

https://mediterraneanworld.wordpress.com/2012/11/21/archaeological-glitch-art/.

Cegielski, Wendy H., and J. Daniel Rogers. 2016. "Rethinking the role of Agent-Based Modeling in archaeology." *Journal of Anthropological Archaeology* 41 (March). Academic Press: 283–98. doi:10.1016/J.JAA.2016.01.009.

"China Has Turned Xinjiang into a Police State Like No Other." 2018. https://www.economist.com/briefing/2018/05/31/china-has-turned-xinjiang-into-a-police-state-like-no-other.

Christen, Kimberly. 2012. "Does Information Really Want to Be Free? Indigenous Knowledge Systems and the Question of Openness." *International Journal of Communication* 6 (0). http://ijoc.org/index.php/ijoc/article/view/1618.

Cook, Katherine, and Mary E. Compton. 2018. "Canadian Digital Arcaheology: On Boundaries and Futures." *Canadian Digital Arcaheology: On Boundaries and Futures* 42: 38–45.

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.

Denard, Hugh, and others. 2009. "The London Charter for the Computer-Based Visualisation of Cultural Heritage." *No. February*, 1–13.

Department of Archaeology, Monuments, and in cooperation with the Chief Technology Office (CTO) Archaeology (MenA) City of Amsterdam. 2018. "Below the Surface - the Archaeological Finds of the North/South Line." https://belowthesurface.amsterdam/en/pagina/de-opgravingen-0.

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.

Gitelman, Lisa, ed. 2013. *Raw Data Is an Oxymoron*. Cambridge, Massachusetts ; London, England: The MIT Press.

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

Graham, Shawn. 2015. "Mapping the Structure of the Archaeological Web." *Internet Archaeology* 39 (1). http://dx.doi.org/10.11141/ia.39.1.

———. 2017. "Cacophony: Bad Algorithmic Music to Muse To." https://electricarchaeology.ca/2017/02/03/cacophony-bad-algorithmic-music-to-muse-to/.

Graham, Shawn, Guy Massie, and Nadine Feuerherm. 2013. "The Heritagecrowd Project: A Case Study in Crowdsourcing Public History." *Writing History in the Digital Age*.

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.

Guerini, Marco, and Jacopo Staiano. 2015. "Deep Feelings: A Massive Cross-Lingual Study on the Relation Between Emotions and Virality." *CoRR* abs/1503.04723. http://arxiv.org/abs/1503.04723.

Hensinger, Elena, Ilias Flaounas, and Nello Cristianini. 2013. "Modelling and Explaining Online News Preferences." In *Pattern Recognition - Applications and Methods*, edited by Pedro Latorre Carmona, J. Salvador Sánchez, and Ana L.N. Fred, 65–77. Berlin, Heidelberg: Springer Berlin Heidelberg.

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.

———. 2018. "Fleshing Out the Bones: Studying the Human Remains Trade with Tensorflow and Inception." *Journal of Computer Applications in Archaeology* 1 (1). Ubiquity Press, Ltd.: 55–63.

Huggett, J., P. P. Reilly, and G. Lock. 2018. "Whither Digital Archaeological Knowledge? The Challenge of Unstable Futures." *Journal of Computer Applications in Archaeology,* 1 (1): 42–54. http://doi.org/10.5334/jcaa.7.

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.

Kelly, Noah. n.d. "A Diy Guide to Feminist Cybersecurity: Take Control of Your Digital Spaces." Hack*Blossom. https://hackblossom.org/cybersecurity/.

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.

Madrigal, Alexis. 2018. "Wikipedia, the Last Bastion of Shared Reality." https://www.theatlantic.com/technology/archive/2018/08/jeongpedia/566897/.

Maldonado, Adrián. 2016. "The Serialized Past: Archaeology News Online." *Advances in Archaeological Practice* 4 (4). Cambridge University Press: 556–61. doi:10.7183/2326-3768.4.4.556.

Marwick, Ben. 2016. "Computational Reproducibility in Archaeological Research: Basic Principles and a Case Study of Their Implementation." *Journal of Archaeological Method and Theory*, 1–27. http://link.springer.com/article/10.1007/s10816-015-9272-9.

Meyers, Emery, and Andrew Reinhard. 2017. "Trading Shovels for Controllers: A Brief Exploration of Archaeology in Video Games." *Public Archaeology* 14 (2): 137–49.

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.

Morgan, Colleen Leah. 2012. *Emancipatory Digital Archaeology*. University of California, Berkeley.

Moshenska, Gabriel. 2017. "Key Concepts in Public Archaeology." web. https://www.ucl.ac.uk/ucl-press/browse-books/key-concepts-in-public-archaeology.

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.

Parrish, Allison. 2016. "Programming Is Forgetting: Toward a New Hacker Ethic - Transcript of Keynote at the Open Hardware Summit 2016." opentranscripts.org. http://opentranscripts.org/transcript/programming-forgetting-new-hacker-ethic/.

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

Redman, Samuel J. 2016. *Bone Rooms: From Scientific Racism to Human Prehistory in Museums*. Harvard University Press.

Reinhard, Andrew. 2015. "Excavating Atari: Where the Media Was the Archaeology." *Journal of Contemporary Archaeology* 2 (1): 86–93.

———. 2017. "Video Games as Archaeological Sites: Treating Digital Entertainment as Built Environments," In *The Interactive Past: Archaeology, Heritage, and Video Games*, edited by A. A. Mol, Ariese-Vandemeulebroucke C. E., Boom K. H. J., and A. Politopoulos, 99–106. Sidestone Press.

———. 2018. *Archaeogaming: An Introduction to Archaeology in and of Video Games*. Berghahn Books.

Romanowska, Iza. 2015. "So You Think You Can Model? A Guide to Building and Evaluating Archaeological Simulation Models of Dispersals." *Human Biology Open Access Pre-Prints* 79. http://digitalcommons.wayne.edu/humbiol{_}preprints/79.

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.

Winfield, Nick. n.d. "Amazon Pushes Facial Recognition to Police. Critics See Surveillance Risk." https://www.nytimes.com/2018/05/22/technology/amazon-facial-recognition.html.