The Political Methodologist

Editor:
JUSTIN ESAREY, RICE UNIVERSITY
justin@justinesarey.com

Editorial Assistant:
ANDRES CARRIZOSA, RICE UNIVERSITY
acarrizosa@rice.edu

Associate Editors:
RANDOLPH T. STEVENSON, RICE UNIVERSITY
stevenso@rice.edu

RICK K. WILSON, RICE UNIVERSITY
rkw@rice.edu

Contents

Notes from the Editors

The Political Methodologist is still in a transitional period, with the search for a new editorial team (and possibly a new publication structure) still ongoing. But 2016 was a great year for new work in TPM, and that's been reflected in our readership statistics.

In 2016, articles in *The Political Methodologist* were viewed 46,807 times by 34,324 unique visitors. This is slightly less than our viewership for the 2015 year (52,000 views and 37,800 visitors) but still an excellent performance and reflective of the important role that TPM plays as an outlet for discussion of topical and practical issues of interest to the political methodology community.

Our special issue on peer review was a big part of the new content on TPM, and indeed our most viewed post for 2016 came from this special issue. With 3319 views in 2016 (and December of 2015, when the post was originally made), "An Editor's Thoughts on the Peer Review Process" by Sara McLaughlin Mitchell is the most-viewed post on *The Political Methodologist* in 2016. Congratulations!

I also wish to acknowledge that "Making High-Resolution Graphics for Academic Publishing" by Thomas Leeper (originally posted in 2013) is still by far the most popular post on TPM, garnering 20,168 views in 2016 alone. There is no official award or recognition for this distinction, but it is pretty amazing.

On behalf of the (now prior) editorial team, thanks to everyone who contributed to *The Political Methodologist* under our editorship!

-The Editors

By the Numbers: Toward More Precise Numerical Summaries

Gaurav Sood
gsood07@gmail.com

Andrew Guess
New York University
guess@nyu.edu

Unlike the natural sciences, there are few true zeros in the social sciences (p. 960, Gelman, 2011). All sorts of variables are often weakly related to each other. And however lightly social scientists, exogenous events, or other actors intervene, effects of those interventions are rarely precisely zero. When there are few true zeros, categorical statements, like "two variables are significantly related" or "the intervention had a significant effect," convey limited information—about sample size and luck (and manufactured luck). Yet these kinds of statements are the norm in abstracts of the top political science journal, the *American Political Science Review* (*APSR*). As we show later, only 10% of the empirical articles in recent volumes of *APSR* have abstracts

with precise quantitative statements. The comparable number for the *American Economic Review* (*AER*) is 35%.

Informal inspection also suggests that coarse descriptions are common in other sections of social science articles, aside from being exceedingly frequent in the social science vernacular. (We would like to quantify both.) Studies are often summarized as "the study shows this broad phenomenon exists." For instance, Pasek et al. (2015) write, "Moreover, much past research shows that guessing is often biased." Surprisingly, and more problematically, comparison of results often enough also takes the same form. For instance, a study on motivated learning contextualizes the results as follows: "Our finding is consistent with other research showing that even when people agree on factual information, they often still interpret the information in a motivated manner" (from Khanna and Sood 2015). The phrase "finding is consistent with" yields nearly 300,000 results on Google Scholar. And informal inspection suggests that "consistent" covers a surprisingly large range of effect sizes and measures.

None of this is to say that the decision to summarize imprecisely is made without deliberation. Undoubtedly, some resort to coarse summaries because they are not confident about their theories, measures, or models. Others likely use coarse summaries because they think coarse summaries are a more effective way to communicate. In fact, there is some empirical support for the latter thesis. A survey of undergraduate and graduate students found that 77% of the students thought that people preferred to receive verbal expressions of uncertainty over numerical expressions in their everyday lives (Wallsten et al., 1993). (However, the paper also notes that 85% of the students felt comfortable switching to another mode if they thought the other mode suited their needs better.)

Our hunch, however, is that the most common form of coarse summaries in scientific communication—categorical statements around statistical significance—arise as a natural consequence of scientists thinking in terms of the Null Hypothesis Statistical Testing (NHST) framework, which in turn is likely underpinned by a Popperian understanding of science (Gelman and Shalizi, 2013). For instance, hypotheses written in the form of coarse categorical statements around statistical significance, such as, "X will be significantly associated with Y," are exceedingly frequent. These kinds of hypotheses reflect an understanding of science in which scientific progress comes from falsification rather than improvements in measurement.

Whatever the root cause, the use of coarse summaries likely leads to serious problems. First, coarse summarizations risk misinterpretation. Partly because the mapping between verbal phrases and numerical ranges varies between communicators and recipients (Capriotti and Waldrup, 2011), recipients map the same verbal expression to very different numbers (Beyth-Marom, 1982; Bocklisch et

al., 2010; Brun and Teigen, 1988; Simpson, 1944; Simpson, 1963). For instance, Beyth-Marom (1982) elicited numerical mappings of 30 verbal expressions on a 100-point scale and found that the average inter-quartile range of the numerical mapping for a phrase was 14.4. Analogous numbers—standard deviations of the numerical mappings—from Brun and Teigen (1988), based on 27 phrases, and Bocklisch et al. (2010), based on 13 phrases, were 14.2% (translated from a 0–6 scale) and 11.15%, respectively. Relatedly, mapping numerical ranges to verbal phrases in a way that minimizes misclassification error still yields an error rate of nearly 28% (Elsaesser and Henrion, 2013) (see also Bocklisch et al. 2010).

Not only are the mappings variable but the variation is also systematic. Numerical mappings of verbal phrases vary systematically as a function of the phrases used and the characteristics of the recipient. Prominently, numerical mappings of verbal phrases about infrequent events (e.g., "seldom," "rarely," "uncommon") tend to be much less reliable (Wallsten et al., 1986). Interpretation of vague verbal summaries is also subject to cognitive errors. The vaguer a statement, the greater the opportunity to fill in the missing detail. And it is plausible, neigh likely, that people do not resist the opportunity to impute, using common heuristics, such as overweighting accessible information, interpreting evidence in a way that is congenial to their prior beliefs, etc., to impute missing details (Nickerson, 1998; Tomz and Van Houweling, 2009; Brun and Teigen, 1988; Wright et al., 1994).

Use of such cognitive shortcuts is liable to lead to systematic biases in inferences. For instance, in the extreme, confirmation bias implies that people will read an uninformative vague statement as evidence that their priors are correct. It follows that on reading such a statement, people will walk away with yet greater certainty about their priors. For instance, a person who initially believes that a law allowing concealed carry would increase gun crime may optimistically conclude after reading a study summary reporting a positive effect ("the study shows that laws allowing concealed carry increase gun crime") that allowing concealed carry increases gun crime by 20%. The same person reading a summary reporting the opposite effect ("the study shows that laws allowing concealed carry decrease gun crime") may optimistically conclude that the decline is real but of a much lower magnitude.

Finally, coarse summaries may lead to erroneous inferences because of how people use language ordinarily. For instance, a person with flat priors about selective exposure may reasonably interpret a vague summary ("people engage in selective exposure") as implying that most people read news stories from sources that they think are aligned with their party. A more precise numerical statement of the sort that gives the proportion of news stories consumed from ideologically congenial sources would preempt the risk of such

misinterpretation.

Besides misinterpretations of topical effect sizes, coarse summarizations also risk conveying misleading ways of thinking about science—as falsification or simple directional claims rather than as a constant effort to obtain less biased and more precise estimates of actual quantities of interest. Presenting more precise estimates may instill in readers a better appreciation of the point that Donald Green made in an interview in the aftermath of the LaCour scandal: "That's what makes the study interesting. Everybody knows that there's some degree of truth in these propositions, and the reason you do an experiment is you want to measure the quantity."[1]

Making more precise numerical statements may also improve how we understand the results of studies. And over the longer term, by making us think more carefully about our priors, precise numerical summaries may improve how we think about science and interpret scientific results. For instance, presenting precise numerical summaries in abstracts may help us more quickly filter studies in which results appear "too big."[2]

With this preface, we proceed to examine the frequency of vague judgments in social science abstracts.

How Common Are Precise Numerical Summaries in Abstracts?

To assess how common coarse summaries are vis-à-vis more precise numerical summaries of results, we coded 310 abstracts—117 *APSR*, 100 *AER*, and 93 *AER Papers & Proceedings* (*AER P & P*). The *AER* and *AER P & P* samples span June 2014–June 2016 (Vol. 104, 6 through Vol. 105, 6), while the sample of *APSR* abstracts spans February 2013–May 2015 (Vol. 107, 1 through Vol. 109, 2). Given we are only interested in articles in which there are empirical results that can be summarized precisely, we subset on empirical papers. This leaves us with 66 *AER*, 68 *AER P & P*, and 81 *APSR* abstracts.

What we mean by precise numerical summaries of results deserves careful attention. Precision is on a continuum, with summaries ranging from very imprecise to very precise. But for clarity and convenience, our coding scheme captures only one end of the scale. We code summaries of results that take the following form as precise: "A% change in X caused a B% change in Y" or "the intervention caused B% change in Y." For instance, we code the following statements as precise: "The average proportion of 'no' votes is about 40% higher for applicants from (the former) Yugoslavia and Turkey,"

"I find that a one-percentage-point increase in the personal vote received by a gubernatorial candidate increases the vote share of their party's secretary of state and attorney general candidates by 0.1 to 0.2 percentage points." The complementary set includes statements like: "increasing numbers of armed military troops are associated with reduced battlefield deaths," "We find support for these arguments using original data from Uganda," etc.

Figure 1: Proportion of Precise Numerical Statements in Abstracts of Empirical Papers in the *APSR*, *AER*, and the *AER P & P*
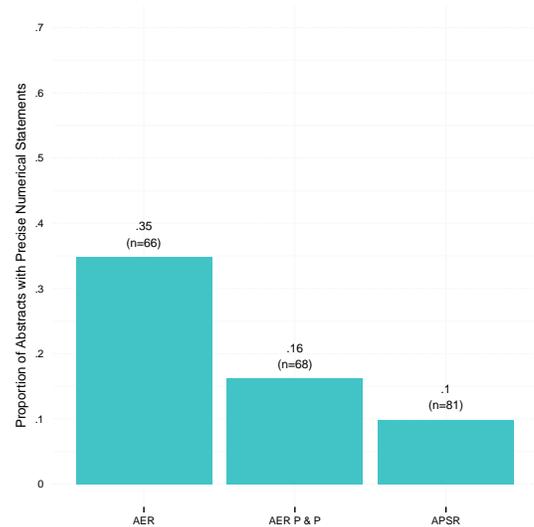
Only about 10% of the empirical articles in recent volumes of APSR have abstracts with precise quantitative statements, similar to the percentage for AER P & P. The comparable number for AER is 35% (see Figure 1). None of the numbers are appealing, but the numbers for APSR stand out. The frequency of coarse summaries of empirical results in abstracts is, however, an imperfect indicator of the dominance of NHST inspired reasoning. The disparity between APSR and AER likely also stems from a lack of widely understood measures in political science. For instance, in Economics, variables like unemployment, inflation, GDP, etc. are widely understood and studied. In political science, only a few variables like turnout come close to being widely understood.

In all, the data shed much needed, but still weak, light on the issue. It is our hope, however, that this note will stimulate discussion about social scientific writing, and increase

---

[1] "An Interview With Donald Green, the Co-Author of the Faked Gay-Marriage Study." Jesse Singal. *New York Magazine*. Published on May 21, 2015. http://nymag.com/scienceofus/2015/05/co-author-of-the-faked-study-speaks-out.html. Green was giving the interview in the aftermath of revelations that Green's co-author had fabricated the data in a highly-publicized study on the persuasive effects of canvassing on attitudes toward gays (Broockman, Kalla, and Aronow 2015; McNutt 2015).

[2] Pushing at an open door: When can personal stories change minds on gay rights? Andrew Gelman. Monkey Cage. The Washington Post. Published on December 19, 2014.

efforts to address (what we contend is) the root cause of a particularly common coarse descriptionâcategorical statements around statistical significance.

Acknowledgments

References

Beyth-Marom, Ruth. 1982. "How probable is probable? A numerical translation of verbal probability expressions." Journal of forecasting 1(3):257–269.

Bocklisch, Franziska, Steen F Bocklisch, Martin RK Baumann, Agnes Scholz and Josef F Krems. 2010. The role of vagueness in the numerical translation of verbal probabilities: A fuzzy approach. In Proceedings of the 32nd Annual Conference of the Cognitive Science Society. pp. 1974–1979.

Broockman, David, Joshua Kalla and Peter Aronow. 2015. "Irregularities in LaCour (2014)." Work. pap., Stanford Univ. http://stanford. edu/ dbroock-/broockman_kalla_aronow_lg_irregularities. pdf .

Brun, Wibecke and Karl Halvor Teigen. 1988. "Verbal probabilities: ambiguous, context-dependent, or both?" Organizational Behavior and Human Decision Processes 41(3):390–404.

Capriotti, Kim and Bobby E Waldrup. 2011. "Miscommunication of uncertainties in nancial statements: a study of preparers and users." Journal of Business & Economics Research (JBER) 3(1).

Elsaesser, Christopher and Max Henrion. 2013. "How Much More Probable is" Much More Probable"? Verbal Expressions for Probability Updates." arXiv preprint arXiv:1304.1501 .

Gelman, Andrew. 2011. "Causality and Statistical Learning1." American Journal of Sociology 117(3):955–966.

Gelman, Andrew and Cosma Rohilla Shalizi. 2013. "Philosophy and the practice of Bayesian statistics." British Journal of Mathematical and Statistical Psychology 66(1):8–38.

Khanna, Kabir and Gaurav Sood. 2015. "Motivated Learning or Motivated Responding? Using Incentives to Distinguish Between the Two Processes." Typescript .

McNutt, Marcia. 2015. "Editorial retraction." Science p. aaa6638.

Nickerson, Raymond S. 1998. "Conrmation bias: A ubiquitous phenomenon in many guises." Review of general psychology 2(2):175.

Pasek, Josh, Gaurav Sood and Job Krosnick. 2015. "Misinformed About the Aordable Care Act? Leveraging Certainty to Assess the Prevalence of Misinformation." Journal of Communication .

Simpson, Ray H. 1944. "The specic meanings of certain terms indicating diering degrees of frequency.".

Simpson, Ray H. 1963. "Stability in meanings for quantitative terms: A comparison over 20 years." Quarterly Journal of Speech 49(2):146–151.

Tomz, Michael and Robert P Van Houweling. 2009. "The electoral implications of candidate ambiguity." American Political Science Review 103(01):83–98.

Wallsten, Thomas S, David V Budescu, Rami Zwick and Steven M Kemp. 1993. "Preferences and reasons for communicating probabilistic information in verbal or numerical terms." Bulletin of the Psychonomic Society 31(2):135–138.

Wallsten, Thomas S, Samuel Fillenbaum and James A Cox. 1986. "Base rate eects on the interpretations of probability and frequency expressions." Journal of Memory and Language 25(5):571–587.

Wright, Daniel B, George D Gaskell and Colm A O'Muircheartaigh. 1994. "How much is 'quite a bit'? Mapping between numerical values and vague quantiers." Applied Cognitive Psychology 8(5):479–496.

The .GOV Internet Archive: A Big Data Resource for Political Science

Emily Gade
University of Washington
ekgade@uw.edu

John Wilkerson
University of Washington
jwilker@uw.edu

Anne Washington
George Mason University
awashi14@gmu.edu

Abstract

We introduce the Internet Archive's .GOV database, which includes more than 1 billion .gov webpage captures in a format that supports large-n systematic analyses. Understanding how to work with .GOV provides an introduction into the ever-expanding world of big data. We offer instructions and tips for using .GOV, and illustrate its potential by exploring attention to several recent issues across U.S. federal government websites. [1]

Introduction

"Big data" will transform social science research. By big data, we primarily mean datasets that are so large that they cannot be analyzed using traditional data processing techniques. However, big data is further distinguished by diverse types of information and the rapid accumulation of that information.[2] We introduce one recently released big data resource, and discuss its promise along with potential pitfalls. For nearly 20 years, governments have used the web to share information and communicate with citizens and the world. .GOV is an archive of nearly two decades of content from .gov domains (US federal, state, local) organized into a database format that is nine times larger than the entire print content of the Library of Congress (90 terabytes, or 90,000 gigabytes).[3] Big data resources like .GOV pose novel analytic challenges in terms how to access and analyze so much data. In addition to the difficulty posed by its size, big data is often messy. Additionally, .GOV is neither a complete nor a representative sample of government presence on the web across time.

The Internet Archive

In 1963, J.C.R. Licklider of the Advanced Research Projects Agency (ARPA) drafted a "Memorandum For Members and Affiliates of the *Inter*galactic Computer *Net*work" (emphasis added). Subsequent discussions ultimately led to the creation of ARPANET in 1968. Soon after, major government departments and agencies were constructing their own "nets" (DOE and MFENet/ HEPNet, NASA and SPAN). In 1989, Tim Berners-Lee proposed (among other things) using hypertext links to enable users to post and search for information on the internet, creating the World Wide Web. The first commercial contracts for managing network addresses were awarded in the early 1990s. In 1995, the internet was officially recognized by the Federal Networking Council, and Netscape Navigator, "the web browser for everyone," went public.

In 1996, a non-profit organization, the Internet Archive (IA) assumed the ambitious task of documenting the public web. The current collection contains more than 450 billion webpage "captures" (downloads of URL linked pages and metadata) dating back to 1995. The best way to quickly appreciate what's in the IA holdings is to visit the WayBack Machine website (`archive.org/web`), where specific historical website captures (e.g. the White House home page from Dec. 27, 1996) can be viewed.

.GOV: Government on the Internet

The Internet Archive also curates sub-collection: .GOV.[4] .GOV contains approximately 1.1 billion page captures of URLs with a .gov suffix (from 1996 through Sept 30, 2013). At the federal level, this includes the official websites of elected officials, departments, agencies, consulates, embassies, USAID missions and much more.[5] Whereas the Wayback Machine makes it possible to view date-specific individual websites, the .GOV collection can be used to investigate patterns across websites and over time.

.GOV offers four types of data from each webpage capture: the link data (the page URL and every other url/hy-

---

[2] For example, see this article on understanding Big Data: Sagiroglu, S., & Sinanc, D. (2013, May). Big data: A review. In Collaboration Technologies and Systems (CTS), 2013 International Conference on (pp. 42-47). IEEE. `https://xa.yimg.com/kq/groups/72986399/1585974627/name/06567202.pdf`

[3] In thinking about using the volume of the Library of Congress as a unit of measure, see: "A "Library of Congress" Worth of Data" by Leslie Johnston, April 25, 2012. `http://blogs.loc.gov/digitalpreservation/2012/04/a-library-of-congress-worth-of-data-its-all-in-how-you-define-it/`

[4] See the Internet Archive's description of their sub-collections here: `https://archive.org/details/additional_collections`

[5] .GOV also includes state and local websites that use the .gov suffix.

perlink found on the page); the parsed text of the page; the full content of the page (the text including html markup language; images; video files etc); and the CDX index file that is used to access the page via the Wayback Machine.

Messy Data

There is no way to download the entire content of the internet, or even a representative sample. The IA (as well as major search firms such as Google) capture content by "crawling" from one page to another. Starting from a limited number of "seed" URLs (web page addresses) a "bot" (software program) collects content from all of the URLS found on the originating page, then all of the URLs on those pages (etc.) until it encounters no more unique pages, or a user defined search constraint tells it to stop. This sequential process inevitably offers an incomplete snapshot of a constantly evolving World Wide Web. In 2008, the official Google blog reported that developers had collected 1 trillion unique URLs in a single concerted effort but also noted that "the number of pages out there is infinite."[6] Crawl results are also incomplete because web pages are sometimes located behind firewalls (the "dark web"), or include scripts that discourage bots from collecting content. The Internet Archive will also delete a website at the owner's request. We have also discovered other limitations of the .GOV data that users should be aware of in designing projects.[7]

The quality of the Internet Archive also improves over time, both because of changes in the way the Web is used and because of changes in the way the Internet Archive conducted its crawls. Figure 1 displays how often the White House website was captured across four different years starting in 1997.[8] The Wayback Machine indicates that `whitehouse.gov` was crawled just 3 times in 1997. In 2001, it was not crawled at all in the month of August and then hundreds of times in the three months following the terrorist attacks on September 11. In 2007, it was captured much more often - at least once a week. And in 2014, `whitehouse.gov` was captured at least once a day.

Figure 1: Frequency of `whitehouse.gov` crawls (selected years)
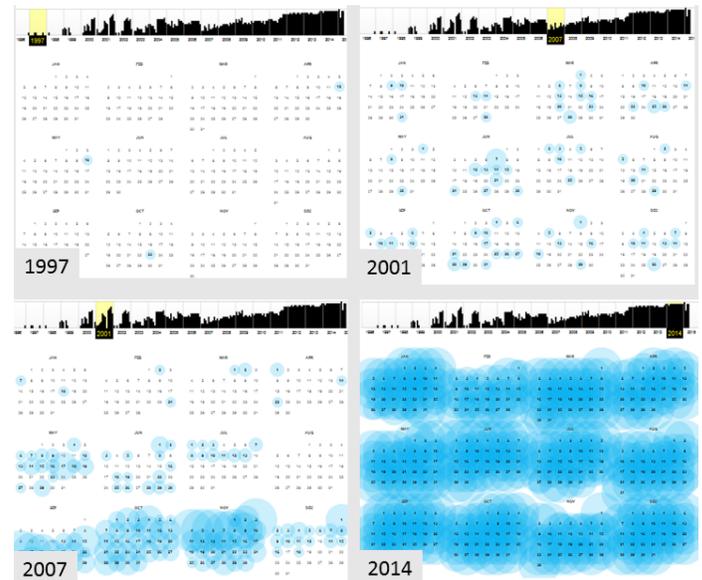
The most complete .GOV crawls occurred during three month time periods (Nov-Jan) of election years starting in 2004.[9] Using congressional websites URLs as the seeds, the IA captured more government web presence than before. Figure 2 indicates spikes in unique .GOV URLs captured during election years. For example, the number triples from about 500 million to 1.5 billion between 2003 and 2004.

Although .GOV is less than ideal as a data resource from a conventional social science perspective, there is no other option for investigating two decades of White House website content, or the content of millions of other pages of government website content. Importantly, because these crawls contain snapshots of each page, researchers could hypothetically examine language that agencies or individuals chose to remove - something scraping those same pages now could not provide. The challenge researchers face is finding the hidden gems in a resource that cannot be easily explored.

---

[6]See Google's Official Blog (July 25, 2008) for discussion at "We knew the web was big..." `http://googleblog.blogspot.com/2008/07/we-knew-web-was-big.html`

[7]These are listed in the on-line Appendix.

[8]The graphs are copied from Wayback Machine search results for `whitehouse.gov`.

[9]According to Vinay Goel, senior data engineer at the IA, the Library of Congress contracted with the IA to systematically capture congressional websites during these time periods.

Figure 2: Total .GOV Unique URLs
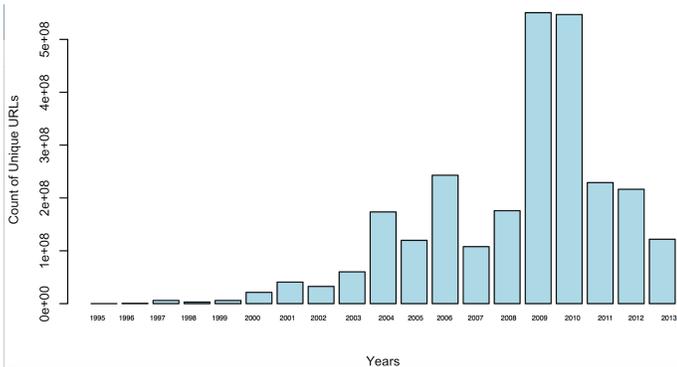
Big Data and Distributed Computing

.GOV is an excellent platform for learning about "big data" analysis techniques. The basic challenge is that the dataset is too large (90,000 gigabytes) to download and explore. Big data is stored and managed differently. Traditional databases (aka "structured data" are organized into neat rows and columns.[10] Big data projects rely on more flexible data storage processes where portions of the data are distributed across a cluster of computers. Each computer in the cluster is a node, and portions of the data are stored in "buckets" or "bins" within each node (see Figure 3). To access the data, researchers use special software to send simultaneous requests to the different nodes. The piecemeal results of these multiple queries are then recombined into a much smaller, single working file.

Figure 3: Hadoop System for .GOV
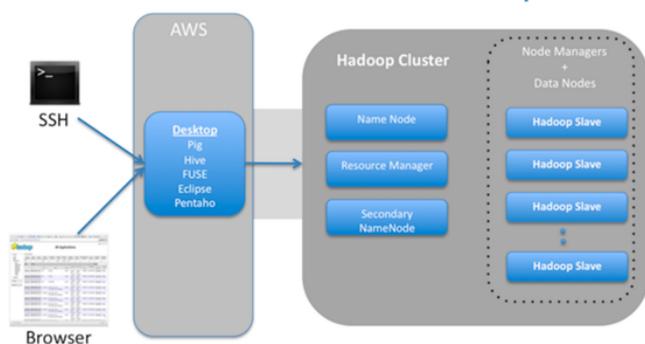
Querying .GOV

The .GOV database is currently hosted on a Hadoop computing cluster operated by a commercial data cloud service,

Altiscale (`www.altiscale.com`). Within the cluster housing .GOV, the data are distributed across nine separate "buckets." Each bucket contains thousands of large (100mb) WARC (Web Archive Container) files (or 'ARC' files for earlier records). Each of these WARC files then contains thousands of individual webpage capture records. As mentioned, each capture record includes the parsed text, the URLs found on the page, the full content of the capture (including images and video files); and the CDX index file. The CDX file includes useful metadata bout specific records that can be used to find and exclude particular records, such as the URL, timestamp, Content Digest, MIME type, HTTP Status Code, and the WARC file where it is located.

The data are accessed using Apache software programs. Apache Pig and Hive are SQL-based languages that can be used for basic data processing such as joining or merging files, searching for specific URLs, and more generally retrieving data of interest. Many Apache commands will be familiar to users with working knowledge of SQL, R or Python. To search all of the capture records in the .GOV database, one must write a query to search thousands WARC files across each of the nine buckets.

Obtaining a key and creating a workbench

Here we describe the big picture process of querying .GOV. In the next section, we present some preliminary findings using the parsed text data. The specific annotated scripts used to accomplish the latter can be found in the on-line Appendix.

Users must first gain access to the Altiscale computing cluster by requesting an "ssh" key (detailed on Altiscale's website: `https://documentation.altiscale.com/getting-started` - you have to email accounts@altiscale.com to request a key). Each key owner is granted a local workbench (an Apache Work Station (AWS)) on the cluster that is similar to the "desktop" of a personal computer and contains the Apache software programs needed to query the database. About 20 gb of storage is also provided (the .GOV database is about 4500 times larger).

Writing scripts to extract information

1. Specifying what is to be collected

Apache Hive and Pig are used execute SQL queries. This can be done on the command line directly (there is no GUI option), but it is easier to write and store scripts on the workbench, and then write a command to execute them across the buckets of interest. For example, one can write an Apache Pig script that requests each parsed text file from a specified URL (e.g. `whitehouse.gov`), separates the parsed

---

[10]See Sagiroglu & Sinanc: `https://xa.yimg.com/kq/groups/72986399/1585974627/name/06567202.pdf`

text fields ("date" "URL" "content" "title" etc.), searches each field for each record for a keyword or regular expression, and then counts how many times a match occurs. The full parsed text could also be downloaded in order to explore the content in more detail later. But with so much data (1.1 billion pages), such a collection can quickly become too large to export.

The functionality of Pig and Hive (like SQL) is limited. For example, Pig will return and count the webpage captures that contain a keyword (true/false) for a date range (e.g., per month/year), but it can't compute the frequency of keyword mentions. To do more detailed or custom analysis, researchers can write user defined functions (UDFs) in Python. The Python script is stored on the workbench as a .py file and then called by the Pig script.[11]

Processing time is a major consideration. Even simple jobs such as keyword counts can take hours or even days to run over so much data. More computationally intensive methods, such as topic models may be impractical. The best way to discover whether a script is going to work and how long it will take is to test it on a subset of the data, such as on just one WARC file in one of the buckets. Running a complete job without testing it is likely to lead to many hours or days of waiting only to discover that it did not work. Linux "Screen" (already installed on the cluster) can then be used to run the script across the cluster remotely (so that your own computer can be used for other things).[12]

2. Providing instructions about where to search on the cluster

The CDX files provide guidance that makes it possible to limit queries to particular URLs, date ranges, WARC files etc.[13] For queries than cannot be restricted in advance (e.g. the research objective is to identify all parsed text files that contain a particular keyword), breaking a job into steps can be more efficient. For example, the first query might identify and produce a list of all of the URLs that contain the keyword. The next query would focus on extracting the relevant information (such as keyword and total word counts) from that more limited set of URLs.

3. Concatenating and exporting the results

Query results for each WARC or ARC file (containing thousands of captures) are stored separately on the cluster. Additional scripts must be written to concatenate them. Whether the results can be exported can also be calculated at this point.[14] If they cannot, Apache Gi-

raffe (`http://giraph.apache.org/`) is designed to facilitate analyses and graphing on the cluster.

Application: Government Attention to the Financial Crisis, Terrorism, and Climate Change

As a starting point to discovering what's in .GOV, we investigate keyword frequencies for three recent issues in American politics. We hope to observe patterns consistent with what is generally known about the issues, and perhaps more novel patterns that begin to illustrate the potential of this new data source.

Collecting the data

We first created a limited list of top level URLs (departments, agencies and political institutions) relevant to the three issues. For example, the regular expression ".house.gov" theoretically captures every webpage of every branch of the official website of the U.S. House of Representatives that the IA collected. This includes, among other things, every Representative's official website (e.g., `pelosi.house.gov`) and every House committee's website (e.g., `agriculture.house.gov`). Aggregating results during the collection process in this way means that we cannot pull out just the results for a particular member or committee's website. That would require a different query using more specific URLs.

We then counted keyword mentions on every subpage of that root URL. We first developed broad lists of keywords related to the three issues. After obtaining results, we created more refined lists by dropping terms that seemed problematic or were used less often (see on-line appendix). For example, we dropped "security" from the terrorism keyword list because it was too general (e.g., financial security). Running the query over all WARC files took about five days of processing time. All together, the results reported below are based on 8.3 billion keyword hits generated by searching about 600 billion words found on the parsed text pages of the specified URLs.

Focusing on raw counts of keywords gives more weight to larger domains. Any changes in attention to terrorism at the much larger State Department will swamp changes at the Bureau of Alcohol, Tobacco and Firearms (ATF). We focus on the proportion of attention given to the issue within an agency or political institution, by dividing the number

---

[11]See `https://wiki.apache.org/Pig/UDFManual` and `https://Pig.apache.org/docs/r0.11.0/udf.html`

[12]For instructions about how to use Screen see: `https://www.rackaid.com/blog/linux-screen-tutorial-and-how-to/#detach`

[13]Instructions for querying the CDX file can be found at `https://webarchive.jira.com/wiki/display/Iresearch/IA+-+GOV+dataset+-+Altiscale`

[14]Instructions for exporting documents from the Altiscale cluster can be found here: `http://documentation.altiscale.com/exporting-data-from-your-altiscale-cluster`

of keyword hits by total website words. A proportion-based approach also does a better job of controlling for the expanding size of government web presence.

Overall Trends

One way to begin to assess the validity of using website content to study political attention is to ask whether changes in content correlate with known events. For each issue in Figure 4, we identified the URLs (federal government organizations) thought to play a role on the issue (see Appendix II for these lists and URLs). The graphs then report average proportions of attention across these URLs.[15] Financial crisis term usage (as a proportion of all terms) spikes upward in 2007-08 as expected (and also in 2001 when there was another stock market decline). Attention to terrorism similarly increases after 9/11/2001, but government-wide attention to terrorism increases most dramatically from 2005 to 2006. Institutionalization is almost certainly part of the explanation. We are capturing attention to terrorism (relative to other issues) on the websites of government agencies. The Department of Homeland Security was not created until 2003 and one of the purposes of its creation was to re-orient the missions of existing agencies (such as FEMA) towards preventing and responding to terrorism. In addition, while 9/11 was an important focusing event for the US, terrorism worldwide continued to increase post 9/11. As we might expect, there is no evidence of equivalent shocks for climate change.

Figure 4: Issue Attention Across .GOV
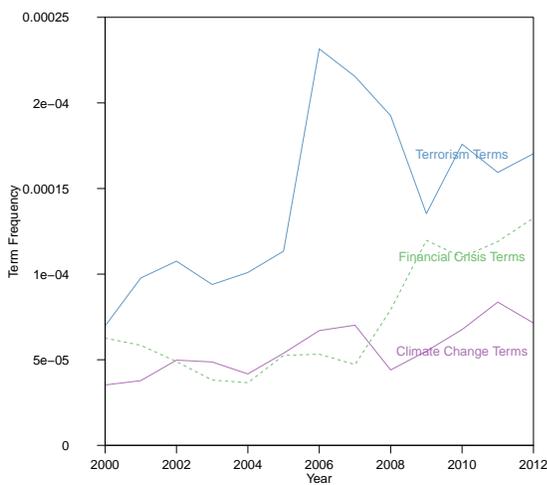
Diffusion of Attention to Terrorism

Political scientists have long been interested in how "focusing events" impact political attention (Birkland 1998). Many studies have examined the impact of 9/11 on the organization and activities of specific government agencies and departments. Here, we ask how attention to terrorism spread across government departments and agencies. Entropy is a measure disorder that is frequently used to study the dispersion of political attention (Boydstun et al. 2014). Figure 5 confirms that attention to terrorism in the federal government became more dispersed post 9/11.[16]

Figure 5: Diffusion in Attention Across .GOV
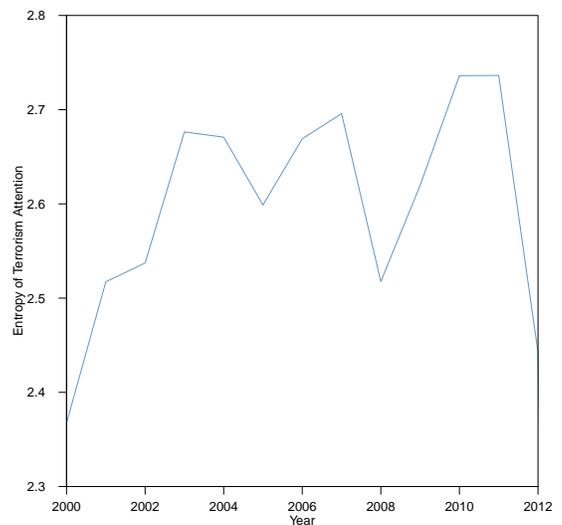
A Financial "Bubble"?

One of the questions raised in congressional hearings after the 2007-08 financial crisis was whether it could have been anticipated and averted. A related question is whether government agencies saw it coming. As an historical archive, .GOV may provide some clues. Here we simply examine "bubble" mentions across organizations (as a proportion of total website words). Figure 6 indicates that references to bubbles spike in the elected branches after the meltdown, whereas bubble mentions at the four agencies most responsible for the economy increase 2-3 years ahead of the crisis. Bubbles also see increased attention at the Federal Reserve before the stock market sell-off in 2001.

---

[15]The different proportions for the different issues are not comparable because they are dependent on the keyword lists.

[16]Our measure is based on the proportion of domain content for 23 departments and agencies, where entropy is based on each domain's proportion of the sum of all agencies' proportions.

Figure 6: Attention to Financial Crisis Across .GOV
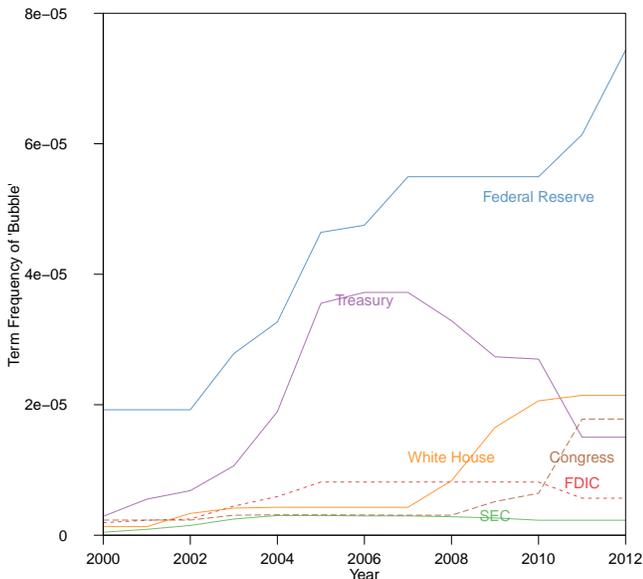
Framing Climate Change

Early in President G.W. Bush's first term of office, pollster Frank Luntz advised Republicans to talk about "climate change" rather than "global warming" because focus groups saw the latter as more of a threat (Leiserowitz et al. 2014, 7). Subsequent academic research also found that the public is somewhat more likely to support action to address global warming. However, it seems as though conservatives also spend much of their time ridiculing global warming. Recently Senator James Inhofe (R-OK) brought a snowball to the Senate floor to question scientists' claims that 2016 was one of the warmest years on record (Bump 2015). If conservatives have discredited global warming in the eyes of the public, then proponents of climate action may have less incentive to use that frame.

In Figure 7, values above .50 indicate that climate change mentions are more common than global warming mentions. "Agencies" refers to the average emphasis on climate change for four agencies with central roles (the EPA, NSF, NOAA, and NASA). According to Figure 7, scientific agencies have always emphasized climate change over global warming, with climate change increasingly favored in recent years. For the elected branches, the patterns are more variable and seem to support the notion that conservatives control the global warming frame. In Congress, global warming has been a more popular frame during periods of Republican control (2001-2008; 2011-2013) and has been used more often over time. The patterns for the White House do not

support what Luntz advised. Global warming receives more attention than climate change for most of the years of the Bush administration. The Obama administration, in contrast, has gone all in for climate change. Although preliminary, these results do suggest that conservatives have defanged what was once the most effective frame for winning public support for climate action.

Figure 7: Attention To Climate Change Across .GOV
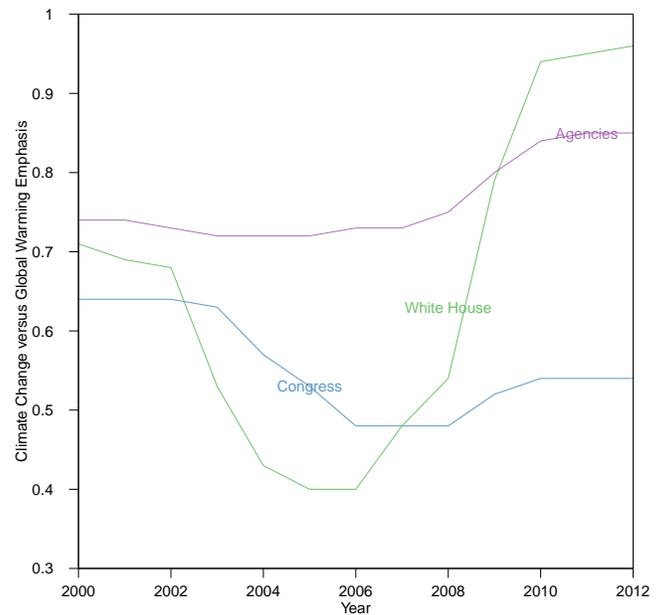
Conclusion

Accessing this big data resource requires new skills and a new mindset. In terms of skills, we hope that our description of the process and working scripts lower the bar. In terms of mindset, political scientists working with statistical methods are used to immediate results. Exploring .GOV in this way is not an option (the Wayback Machine is probably the best way to get a sense of what's in .GOV, and it can take days or even weeks to run a query). On the other hand, .GOV contains insights available nowhere else. Although the current database has important limitations, the Internet Archive recently embarked on a collaboration with many partners to scrape all federal government agencies as completely as possible prior the end of the Obama administration. If this effort is successful and if similar efforts follow in subsequent years, .GOV will be an even more valuable resource for investigating a wide range of questions about the federal bureaucracy and federal programs.

References

"Bash Shell Basic Commands." *GNU Software.* `http://www.gnu.org/software/bash/manual/bash.pdf`

Boydstun, A. E., Bevan, S. and Thomas, H. F. (2014), "The Importance of Attention Diversity and How to Measure It." *Policy Studies Journal*, 42: 173–196. doi: 10.1111/psj.12055

Bump, P. "Jim Inhofe's Snowball Has Disproven Climate Change Once and for All." Washington Post. The Washington Post, 26 Feb. 2015. Web. 28 June 2016.

Birkland, T. A. (1998). "Focusing events, mobilization, and agenda setting. Journal of public policy." 18(01), 53-74.

Edwards, J., McCurley, K. S., and Tomlin, J. A. (2001). "An adaptive model for optimizing performance of an incremental web crawler". *Tenth Conference on World Wide Web* (Hong Kong: Elsevier Science): 106–113.

"The History of the Internet." *The Internet Society.* `http://www.internetsociety.org/internet/what-internet/history-internet/brief-history-internet`

"The Internet Archive." *Internet Archive.* `https://archive.org/`

Kahn, R. (1972). "Communications Principles for Operating Systems." *Internal BBN memorandum.*

Leiner et al. "Brief History of the Internet." `http://www.internetsociety.org/sites/default/files/Brief_History_of_the_Internet.pdf`

Leiserowitz, A. WHAT'S IN A NAME? GLOBAL WARMING VERSUS CLIMATE CHANGE. Rep. Yale Project on Climate Change Communication, May 2014. Web. 28 June 2016.

Licklider, J. C. (1963). "Memorandum for members and affiliates of the intergalactic computer network." M. a. A. to IC Network (Ed.). Washington DC: KurzweilAI. ne.

Najork, M and J. L. Wiener. (2001). "Breadth-first crawling yields high-quality pages." *Tenth Conference on World Wide Web*, (Hong Kong: Elsevier Science): 114–118.

"Pig Manual." *Apache Systems* `https://pig.apache.org/docs/r0.7.0/piglatin_ref1.html`

"The Rise of 3G." *THE WORLD IN 2010.* International Telecommunication Union (ITU)). `<www.itu.int/ITU-D/ict/material/FactsFigures2010.pdf>`.

Sagiroglu, S., and Sinanc, D. (2013, May). "Big data: A review." In Collaboration Technologies and Systems (CTS), 2013 International Conference on (pp. 42-47). IEEE. `https://xa.yimg.com/kq/groups/72986399/1585974627/name/06567202.pdf`

"A "ssh" key (Secure Shell)" (2006). `http://tools.ietf.org/html/rfc4252`

Vance, A. (2009). "Hadoop, a Free Software Program, Finds Uses Beyond Search". *The New York Times.*

Appendix I

The following script flags all web pages that include one or more mentions of the term 'climate change' and stores the full text of those captures. We begin with an overview of the process of running jobs on the cluster, and then provide specific code. For questions, please contact the authors.

Overview

Running scripts on the cluster requires a basic understanding of bash (Unix) shell commands using the Command Line on a home computer (on a Mac, this is the program "Terminal"). For a basic run down of bash commands, see `http://cli.learncodethehardway.org/bash_cheat_sheet.pdf`.

Begin by opening a bash shell on a home desktop, and using a ssh key obtained from Altiscale to log in. Once logged in, you will be on your personal workbench and now have to use a script editor (such as Vi `http://www.catonmat.net/download/bash-vi-editing-mode-cheat-sheet.pdf`). Come up with a name for the script, open the editor, and then either paste or write the desired script in the editor, close and save the file (to your personal workbench on the cluster).

Scripts must be written in Hadoop-accessible languages, such as Apache Pig, Hive, Giraffe or Oozie. Apache languages are SQL-like, which means if you have experience with SQL, MySQL, SQLlite or PostgreSQL (or R or

Python), the jump should not be too big. For text processing, Apache Pig is most appropriate, whereas for link analysis, Hive is best. The script below is written in Apache Pig and a manual can be found at `https://pig.apache.org/`. For an example of some scripts written for this cluster, see `https://webarchive.jira.com/wiki/display/Iresearch/IA+-+GOV+dataset+-+Altiscale`. May be easiest to it "clone" the "archive analysis" file hosted on GitHub from Vinay Goel `https://github.com/vinaygoel` or three basic scripts from Emily Gade `https://github.com/ekgade/.govDataAnalysis` and use those as a launch point. If you don't know how to use GitHub, see here: `https://guides.github.com/activities/hello-world/`(it is actually quite straightforward).

Because Apache languages have limited functionality, users may want to write user defined functions in a program like Python. A tutorial about how to do this can be found at `https://help.mortardata.com/technologies/pig/writing_python_udfs`.

Once a script is written, you will want to run it on a segment of the cluster. This requires another set of Unix style Hadoop shell commands (see `http://hadoop.apache.org/docs/current/hadoop-project-dist/hadoop-common/FileSystemShell.html`). Users must then specify the file path(s), the desired output directory, and where the script can be found.

Getting a Key

As discussed above, this script is run from your workbench on the cluster. To gain access, you will need to set up an SSH "key" with Altiscale (see `http://documentation.altiscale.com/configure-ssh-from-mac-linux`). Once you have obtained and sent your SSH key to Alitscale, you can log in using any bash shell from your desktop with the command "ssh altiscale".

Locating the Data

The Altiscale cluster houses 9 "buckets" of .GOV data. Each bucket contains hundreds or thousands of Web Archive Files (older version are "ARC" files, newer version are "WARC" files, but they have all the same fields). Each WARC/ARC file contains captures from the same crawl, but it (a) won't contain all of the captures from a given crawl, and (b) since the crawl is doing a lot of things simultaneously, captures of a single site can be located in different WARC files.

With so much data, there is no simple "table" or directory that can be consulted to locate a specific web page. The best way to find specific pages is to use Hive to query the CDX database. See Vinay Goel's GitHub for details about how to query CDX: `https://github.com/vinaygoel/archive-analysis/tree/master/hive/cdx`. If a user

knows exactly what he or she wants (all the captures of the whitehouse.gov main page, or all the captures from September 11, 2001), the CDX can tell you where to find them. Otherwise, users will want to query all of the buckets because there is no easy way to learn where results are stored. (Though we advise first testing scripts on a single bucket or WARC file.)

First, use the command line with SSH interface to query the data directories and see which buckets or files to run a job over. This requires the Hadoop syntax to "talk" to the cluster where all the data is stored. The cluster has a user-specific directory where users can store the results of scrapes. A user's local workbench does not have enough space to save them.

Whenever users "talk" from a user's local workbench to the main cluster, users need to use 'hadoop fs -' and then the bash shell command of interest. For a list of Hadoop-friendly bash shell commands, see: `http://hadoop.apache.org/docs/current1/file_system_shell.html`. For example, the line of code

```
hadoop fs -ls
```

pulls a listing of the files in your personal saved portion of the cluster (in addition to the local workbench, each user has a file directory to save the results). As well,

```
hadoop fs -ls /dataset-derived/gov/parsed/arcs/bucket-2/
```

would draw up all the files in Bucket #2 of the parsed text ARCS directory.

Defining Search Terms

Scripts that deal with text are best written in Apache Pig. Hadoop also supports Apache Hive, Giraffe and Spark. To find and collect terms or URLs of interest, users will need to write a script. For example, users might write a script to flag any captures that have a mention of a global warming term, and return the date of the capture, URL, page title, checksum, and the parsed text. This script is saved on your local workbench and needs to have a .pig suffix. Users will need to use some sort of bash editor to write and store the script such as vi (details about how to use vi can be found above). Script is below. The first four lines are defaults and also set the memory.

Script begins:

```
SET default_parallel 100;
SET mapreduce.map.memory.mb 8192;
SET mapred.max.map.failures.percent 10;
REGISTER lib/ia-porky-jar-with-dependencies.jar;
DEFINE FROMJSON org.archive.porky.FromJSON();
DEFINE SequenceFileLoader org.archive.porky.SequenceFileLoader();
DEFINE SURTURL org.archive.porky.SurtUrlKey();
```

The sequence file loader pulls the files out of the ARC/WARC format and makes them readable. Note, when

they were put into the ARC/WARC format, they were run through a HTML parser to remove the HTML boilerplate. However, if the file was not in HTML to begin with, the parser will just produce symbols and this won't fix it. Users will have to deal with those issues separately.

When loading data on the command line (instructions below), give the data a name (here $I_Parsed_Data) and make sure to use the same "name" for the data in the command line command. This is a stand-in for the name of the directory or file over which you will run a script.

```
Archive = LOAD "\$I_PARSED_DATA" USING SequenceFileLoader()
AS (key:chararray, value:chararray);
Archive = FOREACH Archive GENERATE FROMJSON(value) AS m:[];
Archive = FILTER Archive BY m#'errorMessage' is null;
ExtractedCounts = FOREACH Archive GENERATE m#'url' AS src:chararray,
    SURTURL(m#'url') AS surt:chararray,
    REPLACE(m#'digest','sha1:','') AS checksum:chararray,
    SUBSTRING(m#'date', 0, 8) AS date:chararray,
    REPLACE(m#'code', '[^\\p{Graph}]', ' ') AS code:chararray,
    REPLACE(m#'title', '[^\\p{Graph}]', ' ') AS title:chararray,
    REPLACE(m#'description', '[^\\p{Graph}]', ' ')AS description:chararray,
    REPLACE(m#'content', '[^\\p{Graph}]', ' ') AS content:chararray;
```

The above code block says: for each value and key pair, pull out the following fields. Chararray means character array - so a list of characters with no limits on what sort of content may be included in that field. The next line selects the first eight characters of the date string (year, month, day). The full format is year, month, day, hour, second. Unicode errors can wreck havoc on script and outputs. The regular expression pGraph means "all printed characters"– e.g., NOT new lines, carriage returns, etc. So, this query finds anything that is not text, punctuation and white space, and replaces it with a space. Also note that because Pig is under-written in Java, users need two escape characters in these scripts (whereas only one is needed in Python).

```
UniqueCaptures = FILTER ExtractedCounts BY content
MATCHES '.*natural\\s+disaster.*' OR content MATCHES
'.*desertification.*' OR content MATCHES
'.*climate\\s+change.*' OR content MATCHES
'.*pollution.*' OR content MATCHES
'.*ocean\\s+acidification.*' OR content MATCHES
'.*anthropocene.*' OR content MATCHES
'.*anthropogenic.*' OR content MATCHES
'.*greenhouse\\s+gas.*' OR content MATCHES
'.*climategate.*' OR content MATCHES
'.*climatic\\s+research\\s+unit.*' OR content MATCHES
'.*security\\s+of\\s+food.*' OR content MATCHES
'.*global\\s+warming.*' OR content MATCHES
'.*fresh\\s+water.*' OR content MATCHES
'.*forest\\s+conservation.*' OR content MATCHES
'.*food\\s+security.*';
```

This filters out the pages with key words of interest (in this case words related to climate change) and keeps only those pages.

```
STORE UniqueCaptures INTO '\$O_DATA_DIR' USING PigStorage('\u0001');\\
```

This stores the counts the file name given to it. The "using pigstorage" function allows users to set their own delimiters. I chose a Unicode delimiter because commas/tabs show up in the existing text. And, since I stripped out all Unicode above, this should be clearly a new field.

Save this script to your local workbench.

Another option would be to count all the mentions of specific terms. Instead of the above, users would run:

```
SET default_parallel 100;
SET mapreduce.map.memory.mb 8192;
SET mapred.max.map.failures.percent 10;
REGISTER lib/ia-porky-jar-with-dependencies.jar;
```

This line allows you to load user-defined functions from a Python file:

```
REGISTER 'UDFs.py' USING jython AS myfuncs;
DEFINE FROMJSON org.archive.porky.FromJSON();
DEFINE SequenceFileLoader org.archive.porky.SequenceFileLoader();
DEFINE SURTURL org.archive.porky.SurtUrlKey();
Archive = LOAD '\$I\_PARSED\_DATA' USING SequenceFileLoader()
AS (key:chararray, value:chararray);
Archive = FOREACH Archive GENERATE FROMJSON(value) AS m:[];
Archive = FILTER Archive BY m#'errorMessage' is null;
ExtractedCounts = FOREACH Archive GENERATE m#'url' AS src:chararray,
    SURTURL(m#'url') AS surt:chararray,
    REPLACE(m#'digest','sha1:','') AS checksum:chararray,
    SUBSTRING(m#'date', 0, 8) AS date:chararray,
    REPLACE(m#'code', '[\\p{Graph}]', ' ') AS code:chararray,
    REPLACE(m#'title', '[^\\p{Graph}]', ' ') AS title:chararray,
    REPLACE(m#'description', '[^\\p{Graph}]', ' ')AS description:chararray,
    REPLACE(m#'content', '[^\\p{Graph}]', ' ') AS content:chararray;
```

If a user has function which selects certain URLs of interest and groups all other URLs as "other", they would run it only on the URL field. And, if a user has a function that collects words of interest and counts them as well as total words, the user should run that through the content field. Code for using those UDFs would look something like this:

```
UniqueCaptures = FOREACH ExtractedCounts GENERATE myfuncs.pickURLs(src),
    src AS src,
    surt AS surt,
    checksum AS checksum,
    date AS date,
    myfuncs.Threat_countWords(content);
```

In Pig, and the default delimiter is '\n' (new line) but many '\n' appear in text. So one must get rid of all the new lines in the text. This will affect our ability to do text parsing by paragraph, but sentences will still be possible. Code to get rid of the '\n' (new line delimiters) which are causing problems with reading in tables might look something like this:

```
UniqueCaptures = FOREACH UniqueCaptures GENERATE REPLACE(content, '\n', ' ');
```

To get TOTAL number of counts of web pages, rather than simply unique observations, merge with checksum data:

```
Checksum = LOAD '\$I\_CHECKSUM\_DATA' USING PigStorage() AS (surt:chararray,
date:chararray, checksum:chararray);

CountsJoinChecksum = JOIN UniqueCaptures BY (surt,
checksum), Checksum BY (surt, checksum);

FullCounts = FOREACH CountsJoinChecksum GENERATE
    UniqueCaptures::src as src,
    Checksum::date as date,
    UniqueCaptures::counts as counts,
    UniqueCaptures::URLs as URLs;
```

This would sort counts by original "source" or URL:

```
GroupedCounts = GROUP FullCounts BY src;
```

This fills in the missing counts and stores results:

```
GroupedCounts = FOREACH GroupedCounts GENERATE
    group AS src,
    FLATTEN(myfuncs.fillInCounts(FullCounts)) AS (year:int,
    month:int, word:chararray, count:int, filled:int,
    afterlast:int, URLs:chararray);
STORE GroupedCounts INTO '\$O\_DATA\_DIR';
```

The UDFs mention here (pickURLs, Threat_countWords, and FillinCounts) are written in Python and can be seen in at the bottom of this Appendix.

Running the Script

To run this script, type the following code into the command line, after having logged in the Altiscale cluster with your ssh key. Users will select the file or bucket they want to run the script over, and type in an "output" directory (this will appear on your home/saved data on the cluster, not on your local workbench). Finally, users need to tell Hadoop which script they want to run. The I_PARSED_DATA was defined as the location of the data to run the script over in the script above. Here we telling the computer that this bucket is the I_PARSED_DATA. Next, one must load the CHECKSUM data, and finally, give the output directory, and the location of your script.

The following should be run all as one line:

```
pig -p I_PARSED_DATA=/dataset-derived/gov/parsed/arcs/bucket-2/
-p I_CHECKSUM_DATA=/dataset/gov/url-ts-checksum/
-p O_DATA_DIR=place_where_you_want_the_file_to_end_up
location_of_your_script/scriptname.pig
```

Exporting Results

Lastly, to remove results from the cluster users need to open a new Unix shell on their local machine that is NOT logged in to the cluster with their ssh key. Then type the location of the file they'd like to copy and give it a file path for where they'd like to put it on their desktop. For example:

The following should be run all as one line:

```
scp -r altiscale:~/results_location
/location_on_your_computer_you_want_to_move_results_to/
```

For additional scripts and for those with programming experience, see Vinay Goel's GitHub at `https://github.com/vinaygoel/archive-analysis`. For stepwise instruction of a word count script, see Emily Gade's GitHub at `https://github.com/ekgade/.govDataAnalysis`.

Python UDFs:

```
#import packages
from collections import defaultdict
import sys
import re

#define output schema so the UDF can talk to Pig
@outputSchema("URLs:chararray")

# define Function
```

```
def pickURLs(url):
    try:
        # these can be any regular expressions
        keyURLs = [
        'state\.gov',
        'treasury\.gov',
        'defense\.gov',
        'dod\.gov',
        'usdoj\.gov',
        'doi\.gov',
        'usda\.gov',
        'commerce\.gov',
        'dol\.gov',
        'hhs\.gov',
        'dot\.gov',
        'energy\.gov',
        'ed\.gov',
        'va\.gov',
        'dhs\.gov',
        'whitehouse\.gov',
        '\.senate\.gov',
        '\.house\.gov']

        URLs = []
        for i in range(len(keyURLs)):
            tmp = len(re.findall(keyURLs[i], url, re.IGNORECASE))
            if tmp > 0:
                return keyURLs[i]
        return 'other'

# counting words

#define output schema as a "bag" with the word and then the count of the word
@outputSchema("counts:bag{tuple(word:chararray,count:int)}")

def Threat_countWords(content):
    try:
        # these can be any regular expressions
        Threat_Words = [
            '(natural\sdisaster)',
            '(global\swarming)',
            '(fresh\swater)',
            '(forest\sconservation)',
            '(food\ssecurity)',
            '(security\sof\sfood)',
            'desertification',
            '(intergovernmental\spanel\son\sclimate\schange)',
            '(climatic\sresearch\sunit)',
            'climategate',
            '(greenhouse\sgas)',
            'anthropogenic',
            'anthropocene',
            '(ocean\sacidification)',
            'pollution',
            '(climate\schange)']

#if you want a total of each URL or page, include a total count
        threat_counts  = defaultdict(int)
        threat_counts['total'] = 0

        if not content or not isinstance(content, unicode):
            return [(('total'), 0)]
        threat_counts['total'] = len(content.split())

        for i in range(len(Threat_Words)):
            tmp = len(re.findall(Threat_Words[i], content, re.IGNORECASE))
            if tmp > 0:
                threat_counts[Threat_Words[i]] = tmp

        # Convert counts to bag
        countBag = []
        for word in threat_counts.keys():
            countBag.append( (word, threat_counts[word] ) )
        return countBag

## filling in counts using CHECKSUM and carrying over counts
from the "last seen" count

@outputSchema("counts:bag{tuple(year:int, month:int, word:chararray, count:int,
filled:int, afterLast:int, URLs:chararray)}")
```

```
def fillInCounts(data):
    try:
        outBag = []
        firstYear = 2013
        firstMonth = 9
        lastYear = 0
        lastMonth = 0
 # used to compute averages for months with multiple captures
 # word -> (year, month) -> count
        counts = defaultdict(lambda : defaultdict(list))
        lastCaptureOfMonth = defaultdict(int)
        endOfMonthCounts = defaultdict(lambda : defaultdict(lambda:
        dict({'date':0,'count':0})))
        seenDates = {}
#ask for max observed date
        for (src, date, wordCounts, urls) in data:
            for (word, countTmp) in wordCounts:
                year = int(date[0:4])
                month = int(date[4:6])
                if isinstance(countTmp,str) or isinstance(countTmp,int):
                    count = int(countTmp)
                else:
                    continue

                ymtup = (year, month)
                counts[word][ymtup].append(count)

                if date > lastCaptureOfMonth[ymtup]:
                    lastCaptureOfMonth[ymtup] = date
                if date > endOfMonthCounts[word][ymtup]['date']:
                    endOfMonthCounts[word][ymtup]['date'] = date
                    endOfMonthCounts[word][ymtup]['count'] = count

                seenDates[(year,month)] = True

                if year < firstYear:
                    firstYear = year
                    firstMonth = month
                elif year == firstYear and month < firstMonth:
                    firstMonth = month
                elif year > lastYear:
                    lastYear = year
                    lastMonth = month
                elif year == lastYear and month > lastMonth:
                    lastMonth = month

        for word in counts.keys():
 # The data was collected until Sep 2013
 make sure that you aren't continueing into the future
            years = range(firstYear, 2014)
            useCount = 0
            afterLast = False
            filled = False
            ymLastUsed = (0,0)
            for y in years:
                if y > lastYear:
                    afterLast = True
                if y == firstYear:
                    mStart = firstMonth
                else:
                    mStart = 1
                if y == 2013:
                    mEnd = 9
                else:
                    mEnd = 12
                for m in range(mStart, mEnd+1):
                    if y == lastYear and m > lastMonth:
                        pass
                    if (y,m) in seenDates:

 # Output sum, as we will divide by sum of totals later
                        useCount = sum(counts[word][(y,m)])
                        ymLastUsed = (y,m)
                        filled = False
                    else:
 # If we didn't see this date in the capture, we want to use the last capture we saw
 # previously (we might have two captures in Feb, so for Feb we output both,
 # but to fill-in for March we would only output the final Feb count)
 # Automatically output an assumed total for each month (other words
 # may no longer exist)
```

```
                        if endOfMonthCounts[word][ymLastUsed]['date'] ==
                        lastCaptureOfMonth[ymLastUsed]:
                            useCount = endOfMonthCounts[word][ymLastUsed]['count']
                        else:
                            continue
                        filled = True
                    if useCount == 0:
                        continue
                    outBag.append((y, m, word, useCount, int(filled), int(afterLast)
```

Appendix II: Lists of URLs and Terms

Figure 8: URLS

Financial	Terrorism	Climate
cftc\.gov	atf\.gov	doe\.gov
doj\.gov	cdc\.gov	doi\.gov
fanniemae\.com	cia\.gov	dot\.gov
fasb\.org	defense\.gov	eia\.gov
fdic\.gov	dod\.gov	epa\.gov
federalreserve\.gov	doe\.gov	fema\.gov
ffiec\.gov	doj\.gov	fws\.gov
fhfa\.gov	dot\.gov	gao\.gov
fhfb\.gov	eia\.gov	gsa\.gov
finra\.org	faa\.gov	nasa\.gov
freddiemac\.com	fbi\.gov	noaa\.gov
fslic	fema\.gov	nps\.gov
ftc\.gov	gao\.gov	nsf\.gov
gao\.gov	gsa\.gov	occ\.gov
ginniemae\.gov	ice\.gov	state\.gov
gsa\.gov	nsa\.gov	usaid\.gov
hhs\.gov	nsf\.gov	usda\.gov
homeloans\.va\.gov	secretservice\.gov	usgs\.gov
makinghomeaffordable\.	state\.gov	
ncua\.gov	treasury\.gov	
sec\.gov	usaid\.gov	
sipc\.org	usda\.gov	
treasury\.gov	usphs\.gov	

Figure 9: Terrorism Terms   Figure 10: Finance Terms

(9//11*)   (ksts)
(al\-qa*)   (mass\scasualt*)
(alien\ssmuggl*)   (massive\scasualt*)
(arms\sprolifer*)   (military\sforce*)
(arms\ssmuggl*)   (non\-state\sactor*)
(arms\stransfer*)   (pandemic*)
(assassin*)   (proliferat*)
(atrocit*)   (proliferation)
(authoritarian\spopul*)   (radiological)
(ballistic\smissile)   (securitiz*)
(bin\sladen)   (security)
(biological\sweapon*)   (september\s11*)
(biopreparedness)   (suspicious\sactivity)
(bioregulator*)   (taliban)
(biosecurity)   (terror*)
(bioterror*)   (terrorism)
(border\ssecurity)   (terrorist)
(catastrophic\shealth\seve (threat*)
(chemical\sweapon*)   (violat[a-z]+?\s?o?f?\sint
(collateralize*)   (violat[a-z]+?\s?o?f?\su?n
(conventional\sarm*)   (violent\sconflict)
(counterterror*)   (violent\sextremis*)
(critical\sinfrastructure) (weapon[a-z]+?\sof\smass\s
(cyber\-attack*)   (wmd)
(cyber\sattack*)   (zoonotic\sdisease*)
(cyberattack)
(cybersecurit*)
(cyberterror*)
(cyberwar*)
(cyberwarfare)
(dirty\sbomb)   **SELECT TERMS**
(disease*)
(dual\-use\sgood*)   (proliferat*)
(electronic\swar*)   (september\s11*)
(fissile\smaterial)   (terrorism)
(food\ssecurity)   (terrorist)
(fragile\sstate*)   (weapon[a-z]+?\sof\smass\s
(fundamentalis*)   (wmd)
(genocide*)
(hijack*)
(hostile\sstate*)
(if\syou\ssee\ssomething\,
(illegal\smigration)
(improvised\sexplosive\sde
(insurgen*)
(irresponsible\sstate*)
(jihad)
(known\sand\ssuspected\ste

                                                        (adjustable\-rate\smortgag
                                                        (bailout*)
                                                        (bubble)
                                                        (capital\srequirement*)
                                                        (cdo)
                                                        (conservatorship)
                                                        (default)
                                                        (exposure)
                                                        (fannie\smae)
                                                        (financial\sfraud)
                                                        (foreclosure)
                                                        (freddie\smac)
                                                        (ginnie\smae)
                                                        (haircut)
                                                        (home\sprice*)
                                                        (insolvent)
                                                        (lehman)
                                                        (leverag*)
                                                        (liquidity)
                                                        (mortgage-backed)
                                                        (panic)
                                                        (plunge)
                                                        (predatory)
                                                        (receivership)
                                                        (shadow)
                                                        (sluggish\seconomic\sgrowt
                                                        (solvency)
                                                        (speculat*)
                                                        (sub\-prime)
                                                        (subprime)
                                                        (systemic\srisk)
                                                        (toxic)

                                                        SELECT TERMS
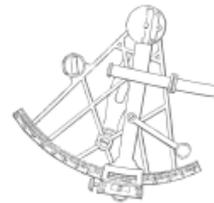   (bubble)
                                                        (default)
                                                        (fannie\smae)
                                                        (foreclosure)
                                                        (freddie\smac)
                                                        (haircut)
                                                        (lehman)
                                                        (liquidity)
                                                        (subprime)
                                                        (systemic\srisk)

Rice University
Department of Political Science
Herzstein Hall
6100 Main Street (P.O. Box 1892)
Houston, Texas 77251-1892

THE SOCIETY FOR POLITICAL METHODOLOGY
ad indicia spectate