The Thesis Committee for Nalbeat Kwon
Certifies that this is the approved version of the following thesis:

Resurrecting Legacy Code to Revitalize Software for Groundwater Research: Reproducibility and Robustness for the Barton Springs Case, Texas

APPROVED BY
SUPERVISING COMMITTEE:

Supervisor:

____
Suzanne A. Pierce

____
Charles W. Kreitler

____
Yolanda Gil

Resurrecting Legacy Code to Revitalize Software for Groundwater Research: Reproducibility and Robustness for the Barton Springs Case, Texas

by

Nalbeat Kwon, B.S.

Thesis

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

Master of Science in Energy & Earth Resources

The University of Texas at Austin

December 2016

Acknowledgements

Thanks to:

The research team:

> Suzanne Pierce for her insightful guidance and moral support through our unexpectedly rocky journey. You have been the best mentor I could ask for. I would give you a million boxes of hangwa if I could.
>
> John Gentle for instilling in me a sense of awareness for the real-life struggles of computing. A lot of this research would have been impossible without you. May this be an ode to our failures and a small toast to our successes.

Yolanda Gil for her inspiring work in the forefront of reproducible science. The second million boxes of Korean confections are for you.

Charlie Kreitler for the best groundwater policy course of my life and helpful comments to steer me in the right direction.

Michael Passarello and Natalie Ballew for their work before me, setting the framework for this research.

Becky Teasley and Kate Sinner for their wonderful contributions and our brainstorming sessions together.

Cyrus Proctor for creating the custom makefile for MODFLOW-96 on Wrangler.

Michael Ciarleglio, Will Cain, Marcel Dulay, Dave Eaton, Jack Sharp, and the rest of the GWDSS team for their vision that started it all.

The Jackson School of Geosciences and the EER faculty for continuing the tradition of excellence.

My family and friends, whom I love very much.

And lastly, Charlie Chaplin for these words of wisdom that taught me how to deal with my hardships: “Life is a tragedy when seen in close-up, but a comedy in long-shot.”

Abstract

Resurrecting Legacy Code to Revitalize Software for Groundwater Research: Reproducibility and Robustness for the Barton Springs Case, Texas

Nalbeat Kwon, MS EER

The University of Texas at Austin, 2016

Supervisor: Suzanne A. Pierce

Advanced computing is becoming an indispensable part of geosciences, the interdisciplinary nature of which often requires large-scale and data-intensive numerical modeling. Groundwater in Texas is one such area that can greatly benefit from advanced decision support for understanding aquifer systems, uncertainty analysis, and policy making. However, software developed for research is often used for a relatively short period of time before it is abandoned or lost. The unintentional abandonment of software within the fast changing technological landscape makes model simulation results difficult to replicate, hindering widespread reusability and causing significant effort to be lost on redeveloping new software for researchers pursuing similar or adapted studies. These legacy codes are potentially important assets and may be resurrected and moved to an archive for long-term reuse. This research develops and tests methodologies to inform the design of best practices for documenting and preserving reproducible workflows and scientific software. Methodologies were tested with an existing codebase and assets from the Groundwater Decision Support System (GWDSS), originally developed in 2006 for participatory decision making and groundwater management. The original GWDSS provided a hybrid architecture for integrated assessment models by combining a numerical simulation code for groundwater (MODFLOW) with other systems dynamics and optimization components. Prior attempts to resurrect GWDSS were unsuccessful due to problems commonly experienced with scientific software, such as insufficient documentation and backward compatibility issues. This research experimented with two resurrection strategies: 1) Initially, a virtual machine (VM) approach to handle compatibility issues, which found similar obstacles in addition to the lack of provenance that would yield questionable results, and possibly inherent problems with the codebase due to uncurated changes made in the past. 2) Then efforts were redirected to writing a new application that replicates and improves many of the old functionalities of GWDSS, leveraging high-performance computing for batch processing of data while seeking to integrate new web-based technologies for data visualization. Ultimately, research efforts informed design and preparation of an ideal architecture that uses an open source framework and technology stack that enables users to easily access and use distributed data systems.

Table of Contents

List of Tables ix

List of Figures x

Chapter 1: Introduction 1

Exploring Robust and Reproducible Decision Support Systems 1

Texas Groundwater Case an an Example 3

Summary of Chapters 9

Chapter 2: Resurrecting Legacy Code 11

Towards Reproducible Science 11

Documentation and Curation for a Reproducible Workflow 12

Case Study—Background on GWDSS 15

Reproducibility for the GWDSS Case 18

Past Attempts at Resurrection of GWDSS 19

Phase II: Virtual Machine and Code Provenance 21

Phase III: Reconstructing GWDSS Functions and Key Features 24

Simulation Modeling Code Compatibility 25

Repurposing Concepts from GWDSS 27

Software Documentation 31

Chapter 3: Proposed Architecture Design for
Advanced Decision Support System and Federated Data Services 34

Overview 34

Key Components of the Idealized Architecture 35

Agave Core 38

ADAMA (Araport Data and Microservices API) 38

Workflow Tools 39

Watermark 39

Example Job Submission Procedure 39

Advantages 41

Leveraging High Performance Computing Systems 43

The Need for High Performance Computing 43
Implementing the FDDSS Workflows on HPC 44
Chapter 4: Testing Scientific Uncertainty—Analysis of Model Results 50
Previous Work 51
Results and Discussion 56
Total Pumping 60
Storage 61
Springflow 63
Chapter 5: Conclusions 68
Appendices 73
Appendix A: Chapter 1 Supplement 73
Appendix B: Chapter 2 Supplement 76
References 77

List of Tables

Table 1.1: Summary of the range of groundwater system characteristics in the state of Texas 4
Table 1.2: Summary of the 41 existing Texas Groundwater Availability Models...6
Table 2.1: Notable attempts and timeline related to the evolution and efforts to revive GWDSS 20
Table 2.2: File structure needed for data preparation to bridge scalars to pumping values 30
Table 3.1: Approximate summative stats on the big data 43

List of Figures

Figure 1.1: Map of major aquifer systems in the state of Texas (from George et al., 2011) 4
Figure 2.1: Diagram of GWDSS architecture (modified from Pierce et al., 2006) 17
Figure 2.2: Virtual machine specifications for recreating GWDSS environment 21
Figure 2.3: Build successful in IDE code editor with 0 errors and 473 warnings 23
Figure 2.4: Application unable to launch on Java web server (last step) 24
Figure 2.5: Left) rch.dat in the correct format. Right) rch.dat format as processed by GWDSS 24
Figure 2.6: Conceptual diagram of the automated workflow, modified from Kwon et al. (2015) 28
Figure 2.7: The 11 pumping zones for reallocation and pumping rate decision variable 29
Figure 2.8: OntoSoft metadata checklist for GWDSS 33
Figure 2.9: Figure 2.9 New scripts on GitHub, a public repository, which also includes user guides 33
Figure 3.1: Idealized workflow architecture, modified from Gentle et al., (in prep) 37
Figure 3.2: Command line interface to use HPC 45
Figure 3.3: Comprehensive HPC workflow for running batch MODFLOW simulations, modified from Gentle et al., (in prep) 47
Figure 3.4: The order of reference for execution of files 48
Figure 4.1: Comparison of average monthly minimum springflow values from the two sets of simulations using different recharge scenarios (Original and Modified datasets, calculated using TROLD and TRBAR recharge interpretations, respectively) (modified from

Ballew, 2015) 54
Figure 4.2: Springflow difference between two sets of simulations (shown in Figure 4.1) using the two recharge interpretations (modified from Ballew, 2015) 55
Figure 4.3: Left) cumulative scalars by hydraulic conductivity zones. Right) location of the 11 zones 57
Figure 4.4: Average scalar values by hydraulic conductivity zones 58
Figure 4.5: Average scalar values (of bin 3; see Figures 4.1 and 4.2) by hydraulic conductivity zones 59
Figure 4.6: Performance of simulated pumping by recharge interpretation....60
Figure 4.7: Performance of simulated storage by recharge interpretation 62
Figure 4.8: Performance of simulated springflow by recharge interpretation....64
Figure 4.9: Drought status for Barton Springs, modified from BSEACD (2016) 65

Chapter 1: Introduction

Exploring Robust and Reproducible Decision Support Systems

The Earth resources so critical to humans are best understood through scientific sources of knowledge such as data and models. Yet the translation or communication of scientific understanding and the nuanced implications are frequently misunderstood, misinterpreted, or simply not conveyed to stakeholders or decision makers that can benefit from science-informed choices (Barbour et al., 2008; Liu et al., 2008; Policansky, 1998). A decision support system (DSS) is an interactive computer application or set of tools designed to aid experts and non-experts in making informed management decisions using the best available science. Effective decision support can be a major bridge between science and policy (Michaels, 2009). Yet constructing a DSS to provide useful insight for science-based decision-making, particularly for complex Earth resource problems, requires integration of knowledge across disciplines and the use of advanced computing resources (McIntosh et al., 2005; Parker et al., 2002; Rizzoli and Young, 1997).

Environmental problems are directly linked to our quality of life (Rizzoli and Young, 1997), but they are often complex problems in which the interactions between numerous parameters over varying spatial and temporal scales do not display easy-to-predict, linear behaviors (Pierce, 2006). Therefore an external modeling tool capable of fast and powerful computations is important to enable policymakers and stakeholders to quickly see the consequences of their decisions over longer time scales. As a result, advanced computing applications are becoming an important part of interdisciplinary geosciences research and advanced decision support.

Simultaneously, there are various challenges to implementing decision support systems. Firstly, the environmental problem that the decision maker is trying to solve with the aid of DSS is difficult, mainly due to the uncertainty of knowledge and complexity of scales, both spatial and temporal (Poch et al., 2004). To understand these complex systems and formulate a solution pool, the developer of the DSS will need to integrate various knowledge realms (Rizzoli and Young, 1997), but in reality there is fragmentation not only between the scientists in different disciplines, but also between scientists and developers, and between scientists and policymakers (Parker et al., 2002). For its application in negotiation settings with stakeholders who are not scientists, a DSS needs to be flexible and easy to use, presenting significant challenges for implementation and design (McIntosh et al., 2005). There are very few off-the-shelf tools to streamline DSS design and implementation, making it a very expensive and time-consuming undertaking. However, despite the difficulties encountered during the development process, a DSS may be abandoned by the user after learning about or solving a particular problem (McCown, 2002). Ensuring reusability of the model and DSS components is vital to prevent extraneous efforts and resources spent on duplicating models or tools developed by other researchers in the past (Rizzoli and Young, 1997). Since the model and data (including minor properties such as configurations, settings, versions, builds, formatting, etc) are necessary to replicate the experiment, reproducibility of science can only be achieved after reusability of the tools has been established.

TEXAS GROUNDWATER CASES AS AN EXAMPLE

Groundwater in Texas is one such area that can greatly benefit from advanced decision support to understand the physical aquifer systems as well as socioeconomic and political systems. Groundwater problems can be viewed as common pool resources (CPR), exhibiting a wide range of complex yet typical challenges for management and policymaking for CPR systems (Pierce et al., 2013). The groundwater systems in the state of Texas have an extensive range of characteristics for physical aspects, climatic conditions, and risk/vulnerability profiles (Pierce et al., 2016; George et al., 2011; Sanger and Reed, 2000). Pierce (2006) identifies approximately 40 parameters that can be used to characterize the behavior and characteristics of aquifer systems. Figure 1.1 shows a map of the major aquifer systems in the state of Texas, and Table 1.1 lists a subset of the physical characteristics that describe an aquifer system, highlighting the broad ranges and variation found in the major aquifers that make Texas an excellent selection of groundwater cases for evaluation.

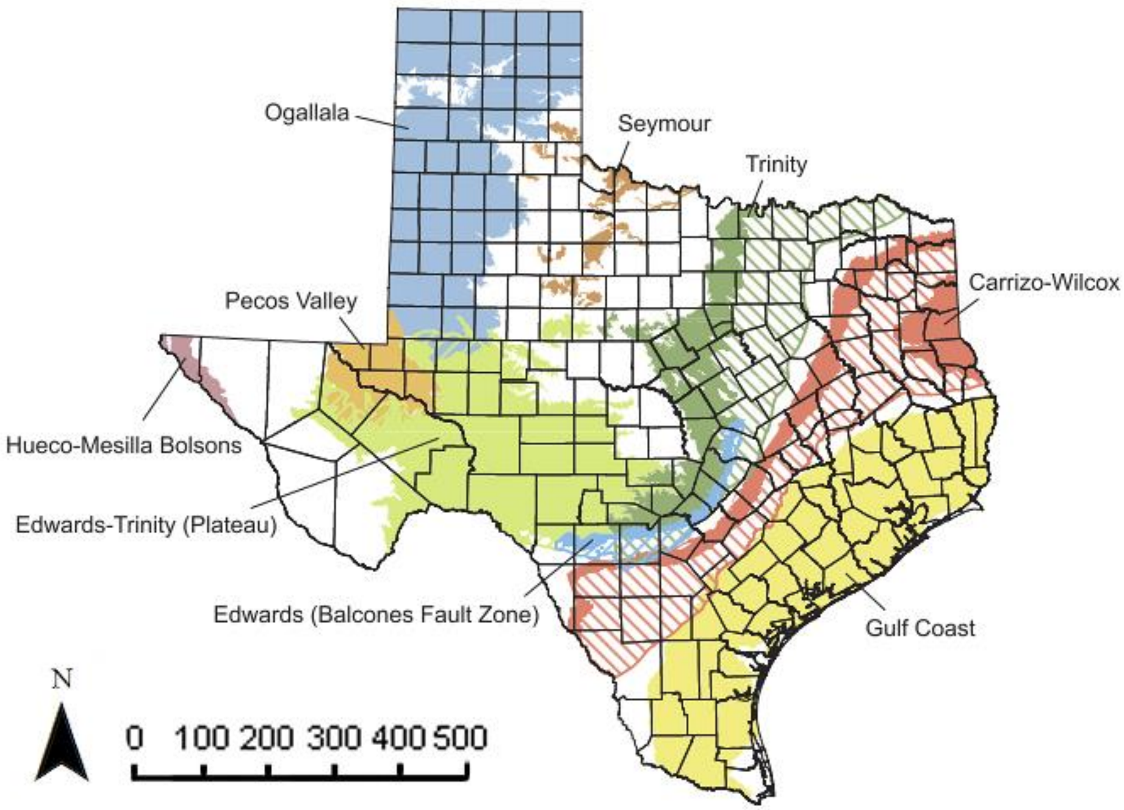


Figure 1.1 Map of major aquifer systems in the state of Texas (from George et al., 2011)

Summarized Range of Physical Characteristics in the Major Aquifers of Texas		
Property	**Min**	**Max**
Areal extent	1,370 mi^2 (Hueco-Mesilla)	41,879 mi^2 (Gulf Coast)
Max Aquifer Thickness	110 m (Seymour)	2,740 m (Hueco)
Average Saturated Thickness	29 m (Ogallala)	580 m (Trinity)
Average Recharge	2.54 mm/yr (Trinity)	31.24 mm/yr (Gulf Coast)
Hydraulic Conductivity	~1 m/day (various)	~2,200 m/day (Edwards BFZ)
Total Dissolved Solids	<100 mg/L (various)	>10,000 mg/L (various)

Table 1.1 Summary of the range of groundwater system characteristics in the state of Texas. Note that the values are for the minimum and maximum reported across major aquifers in the state with the name of the system shown in parentheses (). (Collected from Pierce et al., 2016; George et al., 2011; Sanger and Reed, 2000).

In addition, the state of Texas has a unique program that 1) establishes science-vetted groundwater models and 2) engages stakeholders and planning groups in development of desired future conditions (DFCs) for each groundwater system in the state, and 3) requires use of the models to quantify and assess impacts in support of DFC planning (Mace et al., 2008). In Texas, this model-based planning process depends on simulation models called Groundwater Availability Models (GAMs). These GAMs are the preferred management tool for groundwater as mandated by the Texas State Legislature (Senate Bill 2), and are approved by the Texas Water Development Board (TWDB). The TWDB GAM program represents an ongoing effort to simulate the aquifer systems of Texas and determine the available yield for Texas communities using a standardized and science-informed process (Dougal, 2006; Pierce, 2006; Mullican and Schwartz, 2004).

The most commonly used numerical simulation code for modeling groundwater is MODFLOW, a modular-finite difference groundwater flow model (Harbaugh and Banta, 2005; Harbaugh and Banta, 2000; Harbaugh and McDonald, 1996; McDonald and Harbaugh, 1988; McDonald and Harbaugh, 1984). MODFLOW is recognized as one of the most robust and widely used codes for simulating groundwater flow (Loudyi et al., 2014; Pierce 2006).

Table 1.2 shows the model versions in use for all GAMs in Texas. The complete table with external software platforms, scale, and level of complexity can be found in Appendix A.

GAM	State	Simulation program / version
Carrizo-Wilcox (northern)	Superseded by Carrizo-Wilcox, Queen City, Sparta GAM	MODFLOW-96
Carrizo-Wilcox (central)	Superseded by Carrizo-Wilcox, Queen City, Sparta GAM	MODFLOW-96
Carrizo-Wilcox (southern)	Superseded by Carrizo-Wilcox, Queen City, Sparta GAM	MODFLOW-96
Carrizo-Wilcox, Queen City, and Sparta (northern)	Current	MODFLOW-96
Carrizo-Wilcox, Queen City, and Sparta (central)	Current	MODFLOW-96
Carrizo-Wilcox, Queen City, and Sparta (southern)	Current	MODFLOW-96
Edwards BFZ (northern)	Current	MODFLOW-96
Edwards BFZ (Barton Springs)	Current	MODFLOW-96, MODFLOW-DCM
Edwards BFZ (San Antonio)	Current	GWSIM-IV (1979-), MODFLOW-96 MODFLOW-2000
Edwards-Trinity (Plateau) and Pecos Valley	Current	MODFLOW-96
Gulf Coast (GMAs 15 and 16)	To be completed	
Gulf Coast (northern)	Current	MODFLOW-2000
Gulf Coast (central)	Current	MODFLOW-96
Gulf Coast (southern)	Current	MODFLOW-96
High Plains	Current	MODFLOW-NWT
Hueco-Mesilla Bolsons	Current	MODFLOW-96
Ogallala (northern)	Superseded by High Plains GAM	MODFLOW-96
Ogallala (southern)	Superseded by High Plains GAM	MODFLOW-96
Seymour and Blaine	Current	MODFLOW-96
Seymour (Haskell, Knox, Baylor counties)	Current	MODFLOW-2000

Table 1.2 Summary of the 41 existing Texas Groundwater Availability Models, evaluated in April 2016 (detailed table included in Appendix A).

GAM	State	Simulation program / version
Trinity (northern) and Woodbine	Current	MODFLOW-96, MODFLOW-NWT
Trinity (Hill Country)	Current	MODFLOW-96
Blossom	To be completed	
Brazos River Alluvium	To be completed	
Capital Reef Complex	To be completed	
Edwards-Trinity (High Plains) and Ogallala (southern)	Superseded by High Plains GAM	MODFLOW-2000
Dockrum	Superseded by High Plains GAM	MODFLOW-2000
Lipan	Current	MODFLOW-96
Llano Uplift	To be completed	
Nacatoch	Current	MODFLOW-2000
Rustler	Current	MODFLOW-NWT
West Texas Bolsons (Wild Horse Flat, Michigan Flat, Ryan Flat, and Lobo Flat) and Igneous Aquifers	Current	MODFLOW-96
West Texas Bolsons (Red Light, Green River, and Eagle Flat)	Current	MODFLOW-2000
West Texas Bolsons (Presidio and Redford Bolsons)	Current	MODFLOW-2000
Yegua-Jackson	Current	MODFLOW-2000
Alternative Models (Recalibration / Updates from 96 to 2K)		
Bone Spring-Victorio Peak Model	Current	MODFLOW-2000
Dockrum Alternative Model	Current	MODFLOW-2000
Edwards BFZ (Barton Springs) Alternative Model	Current	MODFLOW-2000
Edwards-Trinity (Plateau) One Layer Model	Current	MODFLOW-2000
GMA 16 Model	Current	MODFLOW-2000
Kinney County Model	Current	MODFLOW-2000

Table 1.2, cont.

As shown in Table 1.2, 41 GAMs can be found on the TWDB website, 7 of which are superseded by newer models and 5 of which are still in the process of development. Counting GAMs with multiple platform overlaps, out of the 29 total current GAMs, 15 use

MODFLOW-96, 13 use MODFLOW-2000, and 4 use specialized modules or variants of MODFLOW-2005 (MODFLOW-DCM and MODFLOW-NWT). The 6 alternative models, included in the 29, are recalibrations of older GAMs and are all in MODFLOW-2000. The 5 that are scheduled to be completed in 2015-2016 are also in MODFLOW-2000.

The majority of the GAMs are built for MODFLOW-96. However, USGS currently supports only MODFLOW-2005, and many optimization and visualization tools by USGS that are specifically built for MODFLOW do not support MODFLOW-96, and the GAMs for MODFLOW-96 and MODFLOW-2000 are not cross-compatible with the newer platform. Many of the current MODFLOW utilities built by USGS are compatible with the superseded MODFLOW-96 if they are only dealing with the binary MODFLOW output data. However, more complex utilities such as FloPy, a Python package for creating, running, and post-processing MODFLOW (Bakker et al., 2016), or ModelMuse, a graphical user interface for MODFLOW (Winston, 2009), are not compatible with the older platforms.

While newer GAMs are being developed in subsequent versions of MODFLOW, these mostly apply to minor aquifers that have had less priority than the major aquifers and are only now being created, or select models for which recalibrations and updates have occurred. Consequently, there is a strong need for updates to established GAMs of many major aquifers of Texas. With each new version of MODFLOW, mathematical improvements to the model as well as additional packages and processes (which define model hydrological capabilities) are developed (Loudyi et al., 2014). Thus maintaining the superseded GAM version for major aquifers equates to not utilizing the higher degree of model precision and the best science currently available to manage our groundwater resources.

SUMMARY OF CHAPTERS

The goal of this research is to create a reproducible workflow for running groundwater models in Texas, to be used as a part of decision support systems and to determine the robustness of policy recommendations and management options. This research can be summarized as follows:

Chapter 1 introduces decision support systems and briefly discusses the benefits and challenges of DSS use in the context of environmental resource problems and policymaking. The difficult and expensive development process of DSS as well as the need to secure reproducibility of science makes reusing the already developed tools the most logical option. This research focuses on Texas groundwater, as the science-vetted policy setting and the diverse physical aquifer systems make Texas a unique laboratory with the right conditions.

Chapter 2 summarizes the efforts to resurrect a legacy codebase of GWDSS (Groundwater Decision Support System) to be reused for practical and theoretical research applications. As the nature of such pursuit does not guarantee success, the chapter provides descriptions of different approaches to rebuild and reconstruct the application. A brief history of the initial development of GWDSS is also provided, along with some best practices towards software reproducibility. The GWDSS code and application are based on the Barton Springs segment of the Edwards Aquifer, located in Texas and providing a useful instance with lessons that can be translated readily for use in other settings.

Chapter 3 proposes an idealized, forward-looking architecture and cyber-infrastructure to establish powerful model capabilities, meaningful visualization tools, relatively painless configurations, and reproducible workflows. Ultimately, this research recognizes that a system of flexible federated services will best accommodate the fast changing modern technological landscape. The description of a workflow leveraging High Performance Computing (HPC) resources, specifically the Wrangler system at the Texas Advanced Computing Center (TACC), is provided.

Chapter 4 analyzes a collection of nearly 40,000 model results achieved using the workflow described in Chapters 2 and 3 with the aid of HPC resources to automate the entire set of runs. The subsequent analysis builds on prior studies and is used to test the robustness of management solutions for a groundwater case study in Texas. Pumping, storage, and springflow metrics were analyzed for each of the four scientifically unique interpretations of recharge for the Barton Springs segment of the Edwards.

Finally, Chapter 5 summarizes and concludes this research by discussing possible next steps for this GWDSS-descendant and providing an outlook for the future of reproducible groundwater model workflows in an era of fast-changing technology.

Chapter 2: Resurrecting Legacy Code

TOWARDS REPRODUCIBLE SCIENCE

Reproducibility is a cornerstone of the scientific process; all science should be testable and reproducible. Any result that cannot be replicated by other peers in the scientific community is not considered stable science. Despite being the simple and widely accepted idea that it is, in the realm of computer-assisted research, it is quite common for even the original scientist to be unable to perform the same model simulation and reproduce identical results after a few years. This problem is especially relevant in geoscience fields, due to the interdisciplinary nature that often requires large-scale and data-intensive numerical modeling. This issue may occur on various levels, including the lack of backward compatibility for software dependencies, complicated workflows that are not adequately curated or documented, data with no provenance (origin and processing history), or restricted access to data/software needed to replicate the experiment.

For software, reproducibility does not mean rewriting the codebase or redeveloping the program from scratch. Instead, a researcher should be able to set up specific configurations for the model runs as described in a publication and achieve the same results; reproducibility lies in reusing the tools, rather than recreating them. Quite often, software developed for research is used for a relatively short period of time before it is abandoned or lost, due to a number of causes: a paper finally makes it to publication, a researcher retires, a graduate student finishes his or her defense, or funding is cut. Once abandoned, the software quickly becomes outdated in the digital landscape that spares little penchant for backward compatibility. This unintentional abandonment of code hinders widespread reusability and causes significant effort to be lost on redeveloping new software for researchers who pursue similar studies in the future. Potentially important assets, these legacy codes can sometimes be resurrected and moved to an archive for long-term reuse.

The vast majority of us habitually save our work—we are aware of the consequences because at some point in our lives we have been exposed to software crashes, hardware failures, and power outages. Although tolerable in small portions during development stages, any final version of a piece of software should be backed up not just onto a local physical hard drive, but redundantly on non-local persistent storage. Historically hundreds of programs are likely to have been lost because they were reliant upon locally stored physical copies (or even a single original version). Ideally, the finished or resurrected code should be openly accessible in a public repository under a version control system for other researchers to share, use, and improve, and indexed for community discovery.

Documentation and Curation for a Reproducible Workflow

No journey to reproducibility is feasible without curation and documentation. Curation refers to a specific set of metadata that tracks the lifecycle of the data or application, such as when and where it was created, where it has been, and all the changes made to it since its origination. Documentation is a more general term that refers to instructions for users and administrators, such as manuals and FAQs that explain what the data/application is and how it is used.

Every program will have its minor bugs and hiccups that may not be discernible through initial scrutiny of the code, but usually a developer who has spent much time with the application will know how to deal with or get around these idiosyncrasies. On the other hand, a passing researcher who simply needs to use the tool will likely have a substantially more difficult time trying to debug an error, even if it is potentially insignificant. Since the scientist cannot be expected to debug the application, he or she can no longer depend on the application's calculations in his or her research, and thus the application is now rendered unreliable simply through lack of proper documentation and maintenance, even if the calculations performed are sound despite the errors being reported (a common occurrence). These insights into how the application operates can best be gained through first-hand experience, but a well-documented guide will greatly lessen the burden. Documentation should be both inside and outside of the code, as comments within the source code of the application and in administrator and user manuals that are made available with the application. Documentation should go beyond the simple listing of dependencies and steps for installation; it should contain unambiguous and comprehensive instructions on many aspects of using the program, because even with ample curation, unforeseen circumstances may rise. It is impossible to account for every error, but ambiguity is best prevented if written with feedback from a scientist from an outside field. In geoscience and other interdisciplinary fields, a layman (non-programmer) should be able to understand the process given relatively short exposure to the software and its documentation. If the software fails with little or no curation to reference, this presents an almost impossible problem to tackle for the scientist without a programming background, as the learning curve of diving into the back end of the application is not one that can be overcome with a few months' effort. Even for a professional programmer, trying to debug or reconstruct a codebase developed by another programmer is not an easy task, especially if the code is not well-commented or curated.

The importance of documentation and commenting the codebase is acutely acknowledged in the developer world as the first and foremost best practice (Sanchez-Rosado et al., 2009). However, the same discipline and background required in software engineering to adhere to best practices and ensure the principles of sound software development may not always be present for the scientist-programmer, a researcher who is simply programming to accomplish a task. There are a number of reasons why the researcher-programmer may not go through the documenting and sharing process. First, documenting takes effort. It is very time-consuming to write easily-understandable documentation that accounts for every possible bug, especially when the efforts of writing a manual for the software are not commonly recognized or valued as those spent writing a scientific paper. But a thorough record should always be kept, and it takes less effort if written sooner. Trying to document subtleties after a couple of years is almost impossible, at which point any meaningful documenting process quickly becomes too hard or too expensive to undertake.

Second, the developer may be so acquainted with the program that he or she does not feel the need to comment the code or write an extensive user manual. However, this familiarity with the codebase may not be true in a few years' time, and for every transition to a new researcher or programmer without proper documentation, the transferred knowledge base becomes thinner and thinner, until no one fully understands the program and it becomes unused.

Third, the developer may be concerned about licensing issues for sharing data or software. If the data generated is sensitive (either proprietary or classified) or the application being developed relies on another third party application with licensing constraints, it may be difficult or even illegal to release the software or data publicly. While the need for security or privacy may be paramount to the organizational goals of the researcher, documenting the software being developed and its usage should still remain a top priority in any project that is expected to continue being used in the future.

This research follows general best practices for preserving reproducible workflows by adhering to the standards set by the NSF-funded EarthCube OntoSoft project to make science more open and facilitate reproducibility (Gil et al., 2015). These include documenting metadata about software on the OntoSoft portal, sharing the code on a public repository under a version control system, and assigning digital object identifiers (DOIs) to all artifacts produced by the research, ensuring that they will be trackable, publishable, and freely available for a reasonably long period of time.

CASE STUDY—BACKGROUND ON GWDSS

For this study, the research team attempted to revive the codebase for the Groundwater Decision Support System (Pierce, 2006), originally developed for participatory decision making to aid urban planning and groundwater allocation strategies for the Barton Springs segment of the Edwards Aquifer, a highly productive karstic aquifer sustaining endangered species and rapid urban development simultaneously. The Barton Springs segment was chosen as the alpha test case because it is a well-studied aquifer with comprehensive historical and current datasets. The GWDSS application was a tightly coupled simulation-optimization program for stakeholders to use in a real-time negotiation setting to compare management strategies and observe how groundwater management or land use choices affect the aquifer system (Pierce, 2006).

GWDSS was designed as a java-based wrapper that uses numerical groundwater simulation code (MODFLOW) with loosely federated commercial and open source software components for systems dynamics integration, optimization, a simple graphical user interface, and database. Figure 2.1 shows a conceptual architecture for the application. The system included interfaces for three levels of users, stakeholder, decision makers, and subject matter experts (either computer scientists or domain scientists). GWDSS connected data through a web-client and server architecture and enabled data storage with a MySQL database. Application Program Interfaces (APIs) enabled connectivity with various simulation and algorithmic capabilities, including numerical groundwater codes, systems dynamics models, and an optimization search engine.

GWDSS could run the full Groundwater Availability Model (GAM) for the Barton Springs segment of the Edwards Aquifer with 7,036 cells as well as a simplified version that uses 11 cells representing aggregated hydraulic conductivity zones. The detailed model with 7,036 cells is useful for research purposes or determining the aquifer system's effective yield, whereas the simpler, high level model is best suited for real-time negotiations and determining consensus yield (Pierce, 2006).

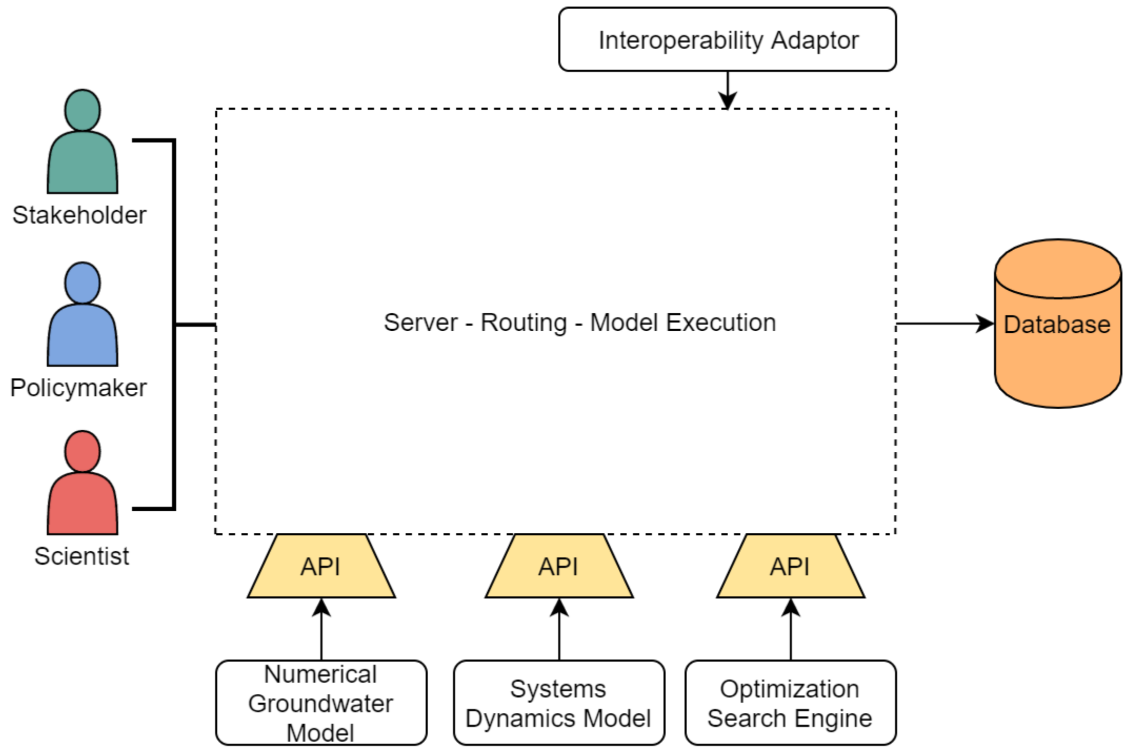


Figure 2.1 Diagram of GWDSS architecture (modified from Pierce, 2006).

The original GWDSS displayed a multimodel capacity to link numerical simulations with the socioeconomic aspect of systems dynamics. It used a non-exhaustive but efficient metaheuristic optimization algorithm capable of searching a subset of a solution space to quickly evaluate alternatives and identify higher performing decision variable settings. It had a basic graphical user interface such as user log-ins, dashboards and map panes for assigning land use distribution and adjusting pumping rates, and multi-metric visualization of 2D graphs and a time sequence of groundwater response. GWDSS was used to produce several datasets in 2006 and as a DSS in a live stakeholder setting[1] in 2008. If successfully revitalized, GWDSS can be useful for both practical applications as a teaching tool and case study for groundwater management, as well as informing theoretical research.

Reproducibility for the GWDSS Case

The GWDSS project for the Barton Springs District saw its fair share of risks related to local storage and lack of documentation, which were discussed in a previous section of this Chapter. During initial development, the codebase was maintained using a version control system that only a small team of developers and researchers were able to access. However, when active development and the project funding ended, the application was partially packaged for installation with limited instructions and the version control was shut down. For years, GWDSS operated on a single machine, principally used for the original development from before 2004 through 2008. Beginning in about 2009, the codebase and application were redundantly stored on the development system and on external hard drives, none of which were truly persistent means of storage.

In 2015, efforts to resurrect the GWDSS repository began. One of the strategies used involved recreating the GWDSS deployment using a virtual machine (VM, which is an emulation of an operating system), but the solid state drive in the laptop on which it was built suffered a total mechanical failure due to a manufacturer flaw. This effort was

[1] GWDSS was used in a real-time negotiation among participating stakeholders. Although it was not an official event for any governing agency, the participants were all part of the Regional Water Quality Planning process and were voting members for that steering committee.

partially successful and the desktop environment was preserved using virtual machine capabilities (Kwon et al., 2016). Had it not been for the virtual machine image backups that were shared online with contributors, this project would have suffered yet another stumbling block on its already rocky journey.

PAST ATTEMPTS AT RESURRECTION OF GWDSS

After active work on GWDSS was paused in 2008, the application slowly retreated into an abandonment state with no direct curation of any possible changes that may have occurred in the process. Since then, several attempts have been made to resurrect the application. Notable endeavors have been summarized in Table 2.1 for easy viewing. This project deals with the phased attempts from 2015 through 2016.

2006	The original GWDSS created and used to produce several datasets and run initial optimization searches for single and multi-objective problem formulations.
2007	An attempt by Sandia National Laboratories to set up a version of GWDSS on their servers with the help of the original development team. The server failed repeatedly, most likely due to conflicting security issues. After months of effort, GWDSS was able to run on the Sandia servers under a rebranded name and modified interface design.
2008	Live GWDSS tested and used for real-time stakeholder dialogue sessions (Pierce et al., 2006). Active usage paused as a decision support tool and codebase extended for use in teaching experiments and studies related to water markets (Broadbent et al., 2008).
2009	The first attempt to resurrect the software at the Center of Agile Technology (CAT) at the University of Texas at Austin was largely unsuccessful due to budget constraints; eventually a basic functioning version was able to run on a CAT application server with a fragmented interface.
2010	GWDSS revamped and installed on a few local machines at the Jackson School of Geosciences at The University of Texas at Austin to be used for graduate research projects (Passarello et al., 2014; Passarello et al., 2011), after which it started again to slowly revert into an uncurated state.
2014	Phase I this project: An attempt to revive the codebase was made, but the technological landscape had evolved so rapidly that the software libraries and architectures required to develop and deploy the application were no longer supported. Subsequently, the existence of GWDSS was identified on old development machines from 2006, and the entire suite of files related to the codebase was backed up onto an external storage device.
2015	Phase II this project: An attempt to reconstruct GWDSS using old settings in an integrated development environment (IDE) inside of a virtual machine. Problems in the GWDSS application arose that were not identified in the documentation and the underlying causes were never effectively discovered (Kwon et al., 2015; Kwon et al., 2016).
2016-	Phase III this project: Efforts switched to repurposing the original concepts from GWDSS and creating a new platform using current technologies, including shell scripting and batch processing on High Performance Computing (HPC) resources and various modern web technologies.

Table 2.1 Notable attempts and timeline related to the evolution and efforts to revive GWDSS.

PHASE II: VIRTUAL MACHINE AND CODE PROVENANCE

The first task was to capture the original codebase and store it in an accessible location for the research team. From the start issues with backward compatibility were recognized, so the initial attempt in 2015 was to launch a working version of GWDSS with an approach to replicate the old development environment using a virtual machine (VM) so that a working state of the software is frozen and encapsulated for reusability. Once the application was able to load, build, and run inside the IDE code editor and the team could validate that the software was working correctly, a virtual image could be captured and transferred to any machine with a VM installed. This Phase II attempt used Oracle VirtualBox, but any other VM solution is viable.

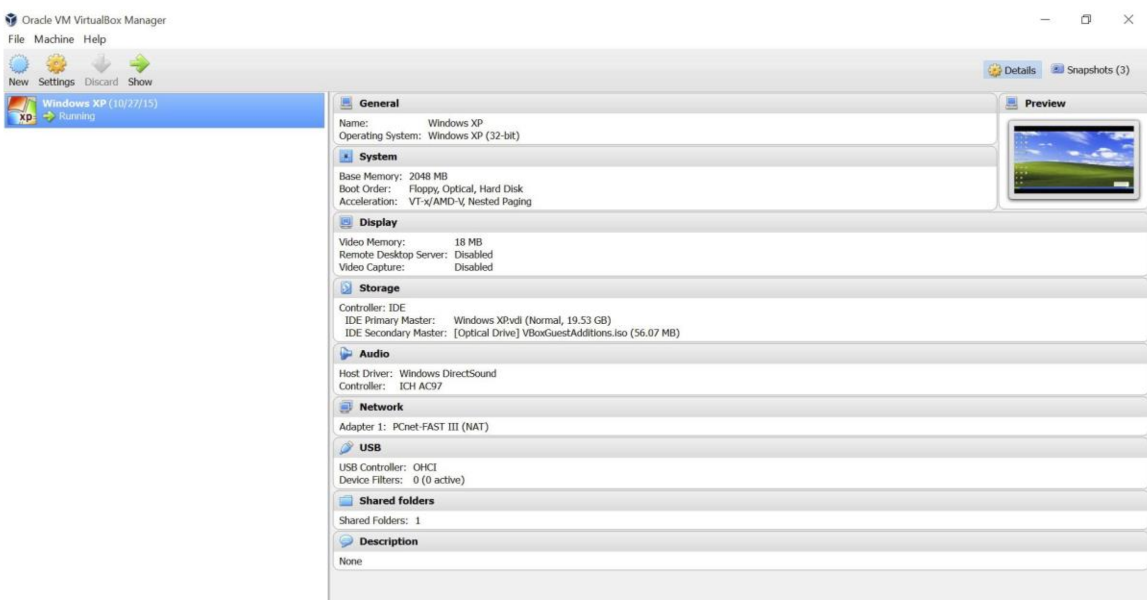


Figure 2.2 Virtual machine specifications for recreating GWDSS environment.

The codebase files were copied to a laptop on which a virtual partition was created. All existing files related to GWDSS were copied, but admittedly there were several different versions of GWDSS with many duplicate files, and possibly even some missing. The specifications for dependencies are listed in Appendix B, as indicated in the notes from the original programmer. These are the specific versions used in the development stage of the program, but not necessarily the only versions compatible.

GWDSS was imported into the Eclipse IDE code editor. When imported all at once, the many versions of GWDSS seemed to create conflicts between themselves within the IDE. Eclipse gave us over 14,400 errors, though most were duplicate errors and could be reduced to about 300 by properly configuring the development environment. Once separately imported, each version of GWDSS had a varying number of errors, but they were significantly fewer. There were only two critical errors for the version that was believed to be the most suitable one to resurrect, which was the most versatile and was not limited to a single case study if resurrected. The code did not build or run at this point. The two errors are listed in Appendix B.

The next step was troubleshooting these errors in Eclipse to get it to build. Most troubleshooting processes required programmer knowledge because they were not explicitly stated in the notes. The author could not have made any progress alone and had significant help from Mr. John Gentle, a professional programmer at the Texas Advanced Computing Center (TACC). Eventually, the build phase was successful with no errors and 473 warnings.

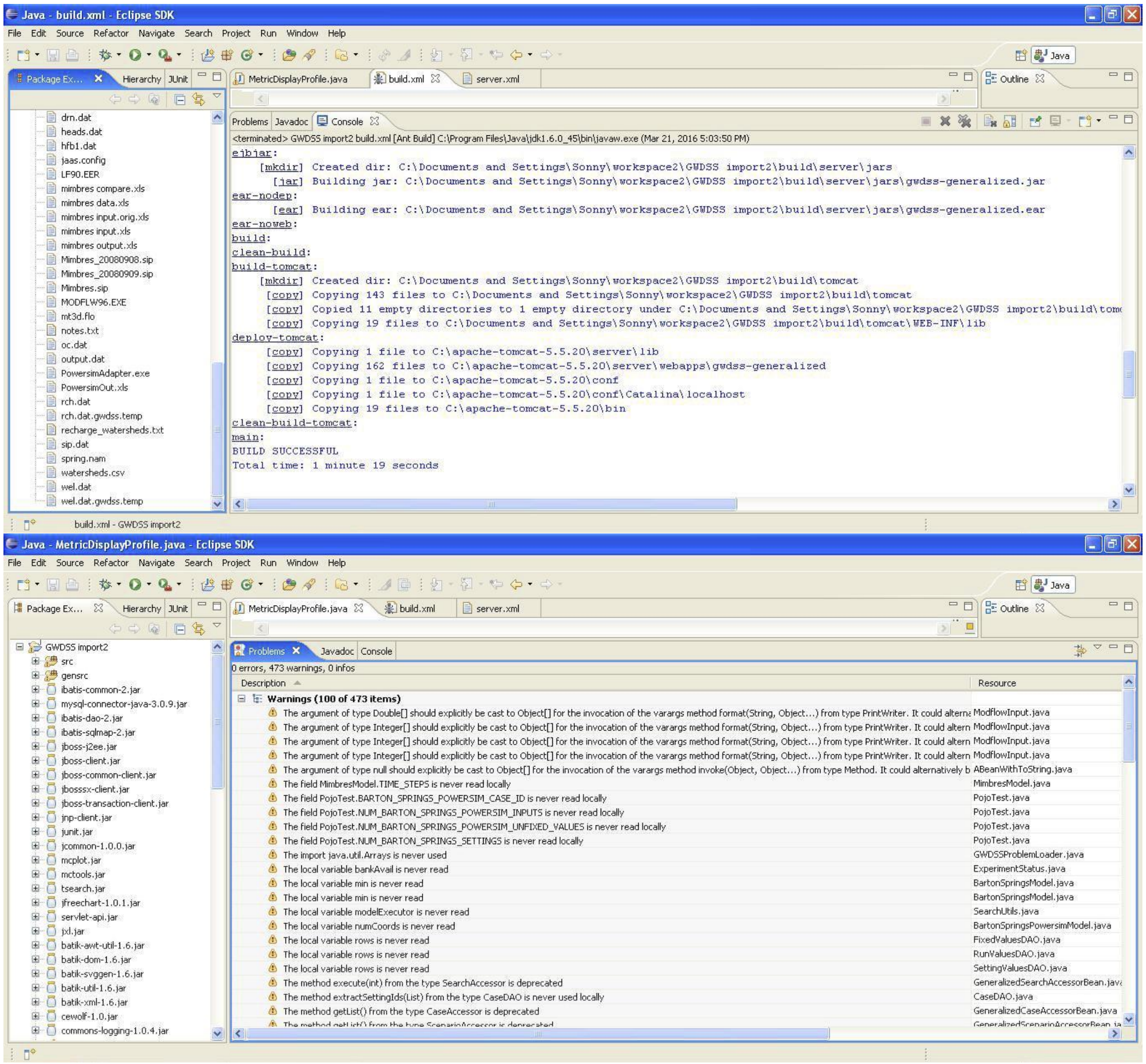


Figure 2.3 Build successful in IDE code editor with 0 errors and 473 warnings.

More troubleshooting was needed to get the application to run. However, the codebase was never debugged in full for a number of reasons, including insufficient documentation, lack of Java-specific expertise, and possibly inherent problems within the codebase. After months of investigating the causes of errors, the Phase II virtual machine resurrection and reuse of the original codebase for GWDSS were deemed unworthy of any further effort.

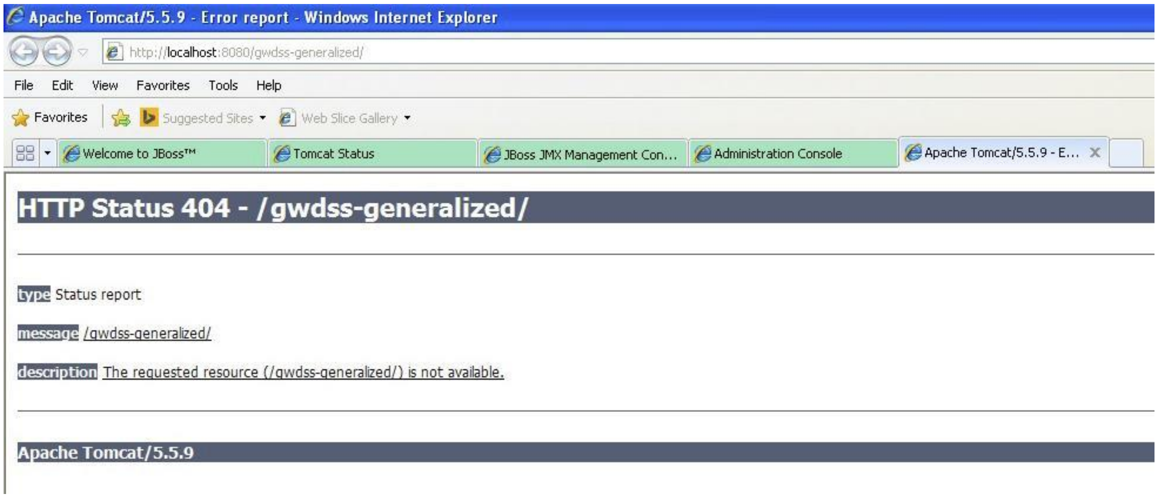


Figure 2.4 Application unable to launch on Java web server (last step).

PHASE III: RECONSTRUCTING GWDSS FUNCTIONS AND KEY FEATURES

Given the development history of GWDSS, which had many branching versions of the codebase containing undocumented and uncurated changes by many researchers, it was decided that GWDSS lacked reliable provenance at this point in its history. Thus the team evaluated options for a path forward and began Phase III of the resurrection. The team redirected its efforts to document the codebase and began repurposing concepts from GWDSS to create a new application. This new GWDSS descendant will be significantly different in both design and architecture, but many of the original features and capabilities can be replicated and improved, such as data visualization and exposing its logic through web services. An updated concept will include a new web-based approach for leveraging High Performance Computing (HPC) systems to run MODFLOW batch processing and data analysis. This is achieved by salvaging pieces of data from GWDSS and creating an automated workflow in a Python shell scripted environment, which replicates the core feature of GWDSS without the participatory modeling aspect (multi-user GUI, visualization, socioeconomic feedbacks, and optimization algorithm). Additional efforts are needed to complete the resurrection/recreation to restore all the former capabilities of GWDSS.

Phase III began with the key challenge in the GWDSS replication process of evaluating the core simulation component, which is a numerical code written in FORTRAN and based on partial differential equations used to calculate groundwater response, called MODFLOW.

Simulation Modeling Code Compatibility

As a large part of the resurrection project depends on actually running the core model (MODFLOW), it is worthwhile to note on the issues confronted when working with this specific modeling platform. As described in Chapter 1, the majority of the state of Texas still relies on an older version of MODFLOW (e.g. 1996). Out of the five main versions of MODFLOW (1983, 1988, 1996, 2000, and 2005), only MODFLOW-2005 is currently supported by USGS.

MODFLOW deals with ASCII text files as inputs and outputs. The file formats are not user-friendly because they have no headings but only numbers and spaces. Aside from both the user manual and programmer documentation for MODFLOW being in general very difficult to understand, the file structures are vaguely defined in the MODFLOW guide; the team was often unable to find explicit documentation for the definitive structure on formats for many input files. Not knowing which arrays of numbers (going on for hundreds of pages) from GWDSS translated to which array in the equivalent file for MODFLOW made data preparation very difficult.

MODFLOW uses FORTRAN code and subsequently the errors it reports are in FORTRAN 77. One particular error encountered by the research team was never fully elucidated by studying the documentation and searching online.

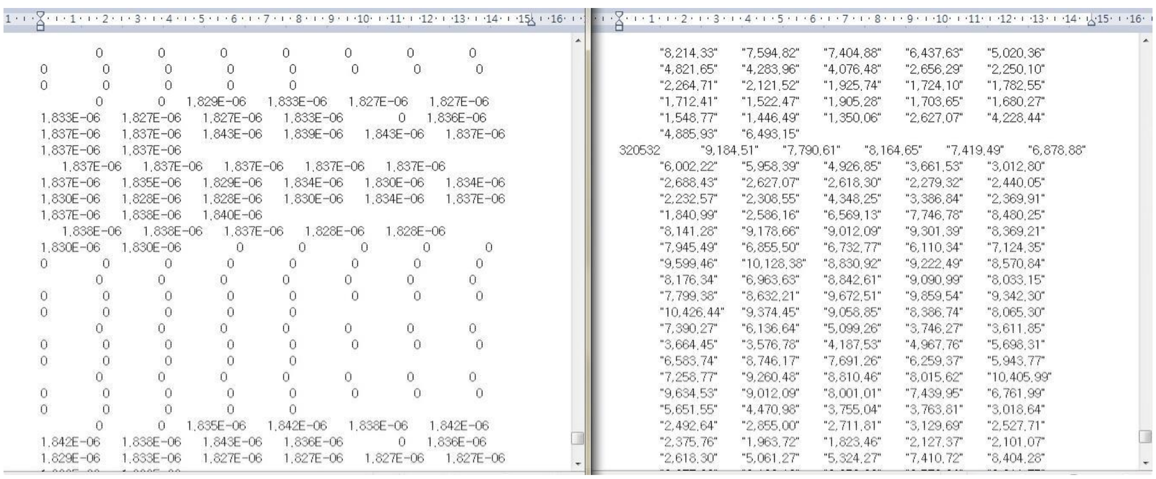


Figure 2.5 Left) rch.dat in the correct format. Right) rch.dat format as processed by GWDSS.

The discrepancies between TWDB's GAMs (designed for MODFLOW-96 or MODFLOW-2000) and USGS's currently supported platform (MODFLOW-2005) creates a compatibility issue that requires a complicated file conversion process to recreate the old groundwater model for MODFLOW-96 for use in the newer versions of the application. USGS has two conversion programs for this purpose: MF96toMF2K and MF2KtoMF05UC. With the help of a partnering research team at the University of Minnesota Duluth, the initial 1996 to 2000 conversion was possible, but the 2000 to 2005 conversion was ultimately unsuccessful, which renders the products from USGS and TWDB incompatible for all practical purposes.

Repurposing Concepts from GWDSS

The team was able to salvage bits of data from the old GWDSS files that could act as starting points for creating new inputs to run the simulations. The team managed to link together the file pieces to create the required input format. Out of the MODFLOW input files required to run the Barton Springs GAM, pumping and recharge files are affected by the decision variable. The aim of this research was to analyze four different recharge scenarios (Passarello et al., 2014). Each recharge scenario entailed running nearly 10,000 pumping scenarios, which were the best performing scenarios chosen by the old optimization algorithm of GWDSS out of the hundreds of thousands of possible runs using the fully detailed model. The near 10,000 solutions were the result of the optimization algorithm running continuously for days, but the decision to stop close to 10,000 was arbitrary.

With nearly 40,000 runs to complete the study of the Barton Springs scenario used during the initial development of GWDSS, automation was crucial; leveraging HPC resources at the Texas Advanced Computing Center (TACC) will make this approach not only feasible but scalable to larger and more complex simulations than previously executed with MODFLOW. Three distinct tasks were identified for the automation process, each as a separate automated process: 1) the batch generation of pumping values and recharge interpretations as input, 2) executing the MODFLOW simulation using the generated input data, and 3) post-processing the simulation outputs for metric extraction, aggregation, and analysis. Python was chosen as the main programming language, primarily because it is the de facto language used in most High Performance Computing environments to connect the various systems, but also because of its huge and growing user base and all-around accessibility.

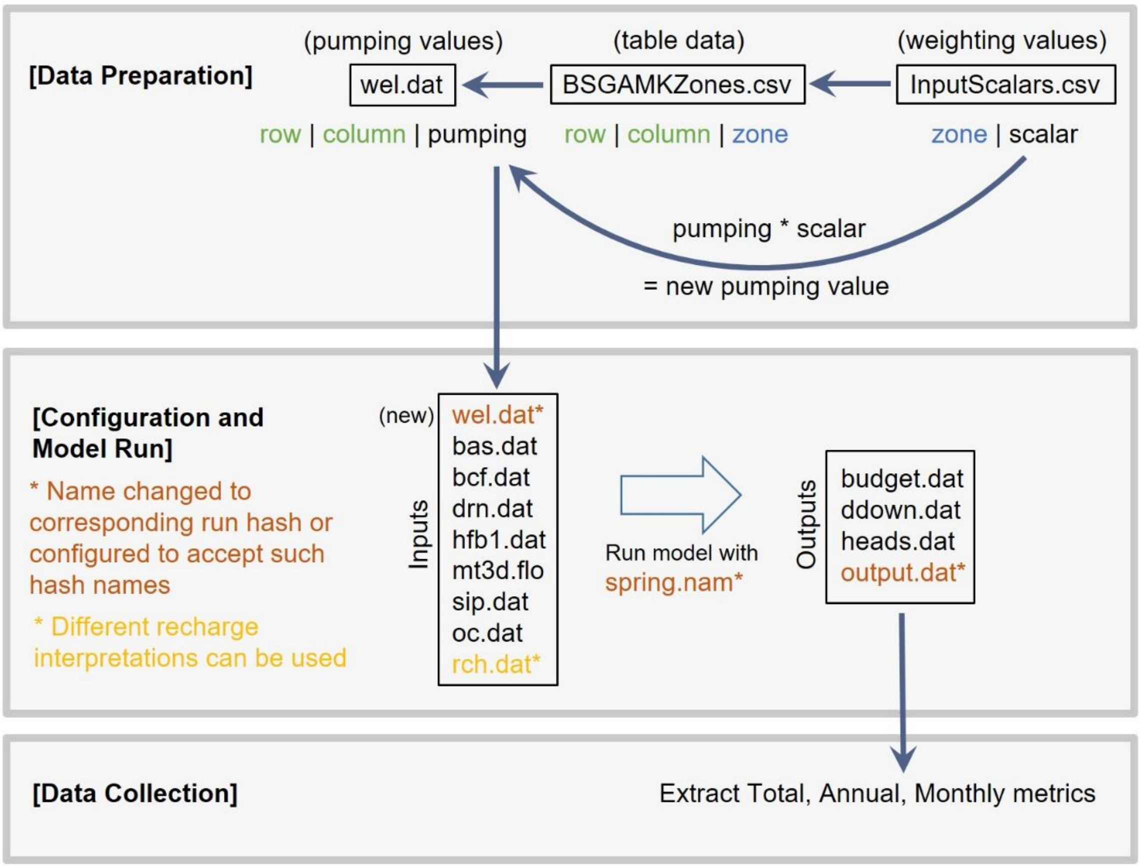


Figure 2.6 Conceptual diagram of the automated workflow, modified from Kwon et al. (2015).

For each groundwater pumping scenario, the pumping values were weighted differently for the 11 spatial zones into which the aquifer grid area is divided, roughly based on actual hydraulic conductivity values of the region to realistically evaluate pumping within meaningful management contexts (Pierce, 2006).

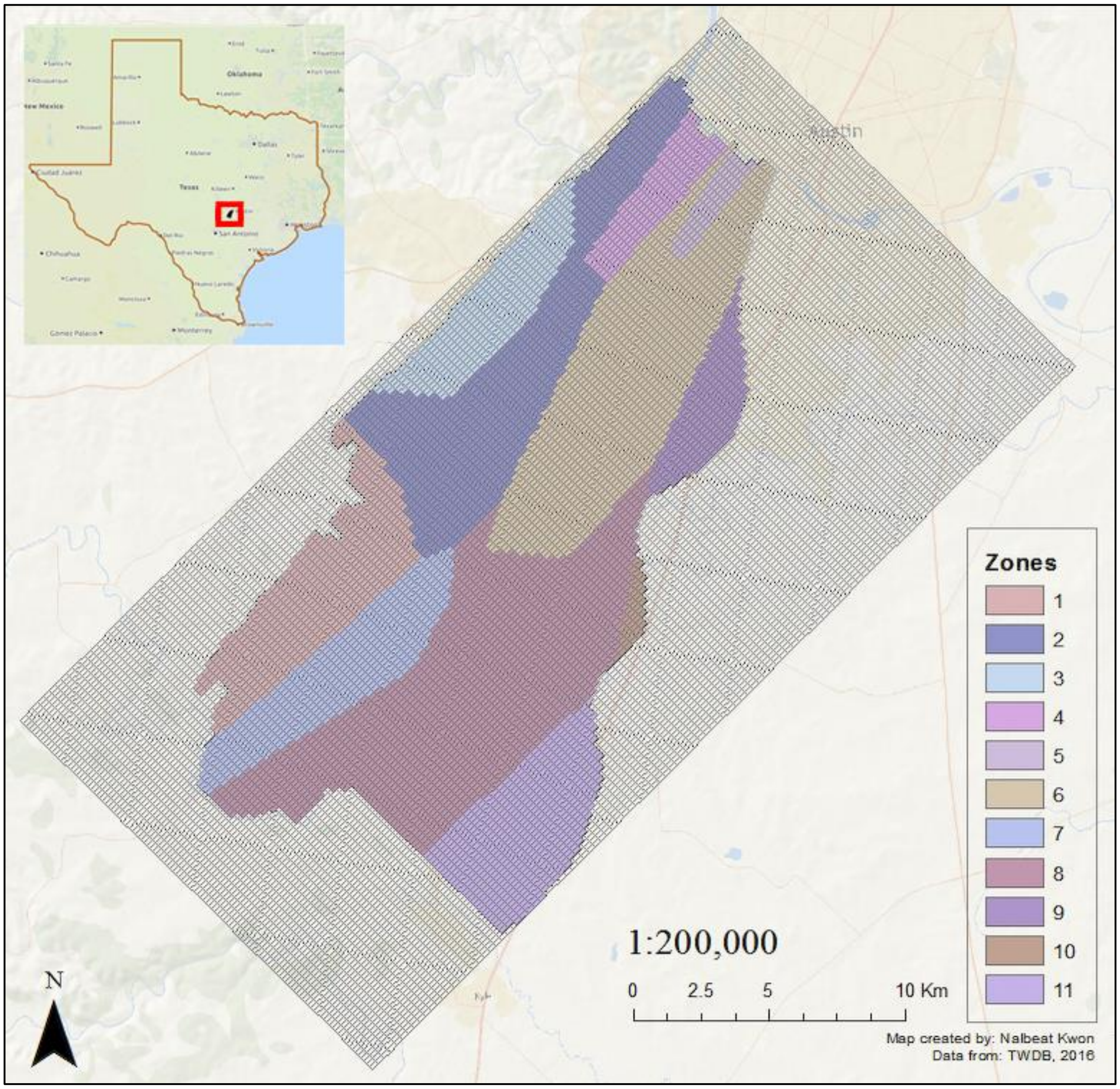


Figure 2.7 The 11 pumping zones for reallocation and pumping rate decision variables. The rectangular grid represents the 14,400 total cells. Only the colored cells (7,036) are active.

Pumping values can be found as wel.dat, a normal MODFLOW binary input file. The baseline wel.dat was available from the original Barton Springs GAM files. The file that had the different weighting scalars for the 11 zones for all ~10,000 simulation runs was found in the suite of GWDSS files. The input scalars, or numerical weighting values ranging from 0.5 to 2, were extracted from the top optimal runs identified by the original GWDSS's metaheuristic optimization algorithm as potential best candidate solutions for resiliency within the aquifer model based on pumping, storage, and recharge interpretation. Because the ~10,000 pumping scenarios are the top performing simulation runs, these scalars represent machine-learned value multipliers for each cell to achieve the optimal pumping results.

File	**Necessary data columns**
wel.dat	row | column | pumping
scalars	zone | scalar
(Missing link)	row | column | zone

Table 2.2 File structure needed for data preparation to bridge scalars to pumping values

Scalars, which are different for each zone, needed to be linked to individual cells in order to be multiplied to the corresponding pumping values. The Barton Springs GAM has 120 rows and 120 columns, or 14,400 cells, out of which only 7,036 are active during the model calculations. Row & column combinations are unique to each cell, and each cell can only fall into one zone. For this process to work, the correct relationships between cells and zones (which zones each of the row & column combinations fell into) had to be known. After an archaeological excavation of files, these values were found contained in the attribute table of a GIS shapefile for the Barton Springs segment of the Edwards Aquifer. Some of the difficulty was due to the lack of headings in all the files.

The code for automation is being developed and maintained under version control in a GitHub repository, which can be found at https://github.com/jgentle/msgdp along with the necessary documentation.

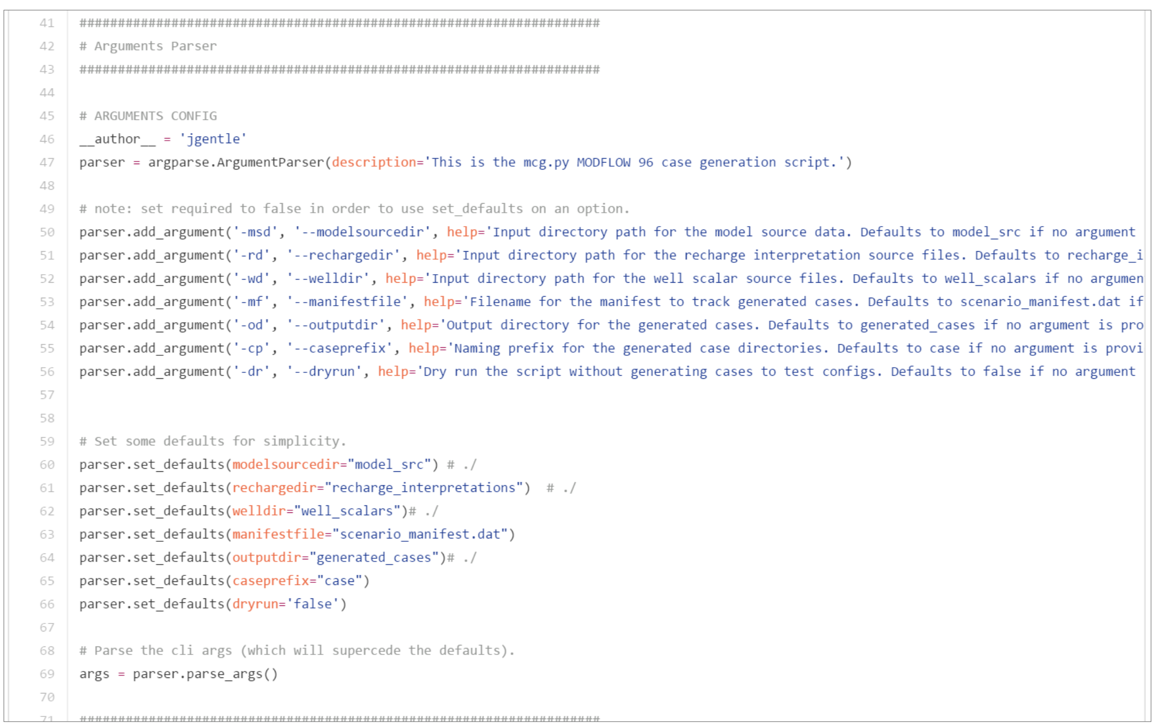
Automation consistency can be verified by running the same process again in a different location and then comparing the results to see if they are identical. Finally, data validity may be tested by comparing the newly created datasets with the original GWDSS results that already exist. Since numerical outcomes of applications may be affected by updating compilers, architectures, and build flags, numerical differences were tolerated within a minor range.

These scripts enable the model simulation functionality of the old GWDSS. The next steps for creating a widely usable groundwater application involve integrating new web-based technologies for data visualization and exploration, and building out a RESTful API for water scientists to access MODFLOW programmatically, generate configurations and workflows, develop scenarios and design models, and publish and curate data from a persistent, secure, and customizable workspace. The next chapter will discuss them in greater detail.

SOFTWARE DOCUMENTATION

Using the knowledge-sharing approaches documented by the NSF-funded OntoSoft project, digital documentation of GWDSS is underway, from conception to development, deployment, characterization, integration, composition, and dissemination through open source communities and geosciences modeling frameworks. Metadata about the software has been completed within the OntoSoft portal to provide descriptive curation, make GWDSS searchable, and complete documentation of the scientific software lifecycle. Information assets, documentation, and examples are shared using open platforms for data sharing and assigned unique DOIs. These artifacts include: 1) the GWDSS VM image in an "as is" state, which includes the full codebase of the original versions of GWDSS, the executable installation files for required third party dependencies, and all developer documentation written by the original developers, 2) newly created Python scripts for automated workflows available via an open GitHub repository, and 3) all other products of this research, including newly generated data, presented posters, etc.

Figure 2.8 OntoSoft metadata checklist for GWDSS. OntoSoft has 6 criteria for documenting software metadata: Identify, Understand, Execute, Do Research, Get Support, and Update. Note that Execute, Do Research, and Update portions are red because GWDSS is currently inactive.


```
########################################################################
# Arguments Parser
########################################################################

# ARGUMENTS CONFIG
__author__ = 'jgentle'
parser = argparse.ArgumentParser(description='This is the mcg.py MODFLOW 96 case generation script.')

# note: set required to false in order to use set_defaults on an option.
parser.add_argument('-msd', '--modelsourcedir', help='Input directory path for the model source data. Defaults to model_src if no argument
parser.add_argument('-rd', '--rechargedir', help='Input directory path for the recharge interpretation source files. Defaults to recharge_i
parser.add_argument('-wd', '--welldir', help='Input directory path for the well scalar source files. Defaults to well_scalars if no argumen
parser.add_argument('-mf', '--manifestfile', help='Filename for the manifest to track generated cases. Defaults to scenario_manifest.dat if
parser.add_argument('-od', '--outputdir', help='Output directory for the generated cases. Defaults to generated_cases if no argument is pro
parser.add_argument('-cp', '--caseprefix', help='Naming prefix for the generated case directories. Defaults to case if no argument is provi
parser.add_argument('-dr', '--dryrun', help='Dry run the script without generating cases to test configs. Defaults to false if no argument


# Set some defaults for simplicity.
parser.set_defaults(modelsourcedir="model_src") # ./
parser.set_defaults(rechargedir="recharge_interpretations")  # ./
parser.set_defaults(welldir="well_scalars")# ./
parser.set_defaults(manifestfile="scenario_manifest.dat")
parser.set_defaults(outputdir="generated_cases")# ./
parser.set_defaults(caseprefix="case")
parser.set_defaults(dryrun='false')

# Parse the cli args (which will supercede the defaults).
args = parser.parse_args()

########################################################################
```
Figure 2.9 New scripts on GitHub, a public repository, which also includes user guides.

Chapter 3: Proposed Architecture Design for Advanced Decision Support Systems and Federated Data Services

OVERVIEW

Advancing decision support capabilities for groundwater systems requires powerful computing capabilities with complex data and simulation model integration. This chapter describes an idealized architecture that improves on the original Groundwater Decision Support System (GWDSS) design while offering support for running multiple iterations of groundwater simulations, along with the initially intended DSS functionality for scenario generation and optimization tools. This framework of advanced decision support and federated data services will enable both technical and non-technical users to easily access High Performance Computing (HPC) systems from various platforms (desktops, mobile, or instrumentation), while ensuring reusable and reproducible workflows. Leveraging current technology trends such as RESTful[2] APIs[3] and Linux Containers (LXCs)[4] allows the application components to be scaled as required to support various usage levels within

---

[2] **RESTful API:** an API that uses HTTP methodologies to interact with data. REST stands for Representational State Transfer. Browsers use HTTP, which allows them to interact with remote systems through a RESTful interface.

[3] **API (Application Program Interface):** a specification comprised of various protocols, methods, and tools that defines how software components interact with other software components. For example, when an end user visits a website, the API defines the possible ways the user can query the database via the user interface and what types of processing can be done on that data.

[4] **Linux Containers (LXC):** Virtualization method for running multiple sandboxed virtual processes that all share a single Linux kernel, as opposed to traditional virtual machines which run multiple virtual operating system that share a single common hardware resource pool. Linux containers enable the reliable publication and distribution of modern scalable web architectures and microservices.

the research community across a wide array of hardware platforms, and with the ability to extend the application to incorporate or transition to new or better technology solutions as they become available in the future.

This framework will integrate four major components: 1) Agave Core API to link with fundamental computing abilities, 2) ADAMA data federation strategy for building community web services, 3) automated workflow capabilities to parallelize and distribute jobs on the HPCs, and 4) Watermark application for interactive data visualization. Only the third component (HPC parallelization) is implemented for this research.

The latter part of this chapter discusses the steps taken by the research team to run the Barton Springs Groundwater Availability Model (GAM) on the High Performance Computing systems at TACC. It describes efforts to establish and implement the federated data and DSS (FDDSS) framework for the reader to use the tools already developed to replicate the original GWDSS performance and reproduce the set of candidate solutions (e.g. Pierce et al, 2016).

KEY COMPONENTS OF THE IDEALIZED ARCHITECTURE

The four key components of an ideal cyberinfrastructure for a powerful GWDSS-descendent are shown in Figure 3.1. Conceptually, these components provide the following services for a DSS: 1) HPC & cloud computing capabilities, 2) access to community data sources, 3) project or case database, and 4) knowledge base. Note that the primary cyberinfrastructure, or Agave Core, is not explicitly drawn in this diagram as it is an abstract aspect that underlies the rest of the system components that are shown in the diagram. Workflow and Knowledge Base (blue rectangle) is where users and developers will code in any automated reasoning, objective functions, or constraints for a case study, and it represents the Barton Springs case study workflow discussed in Chapter 2 (Figure 2.6).

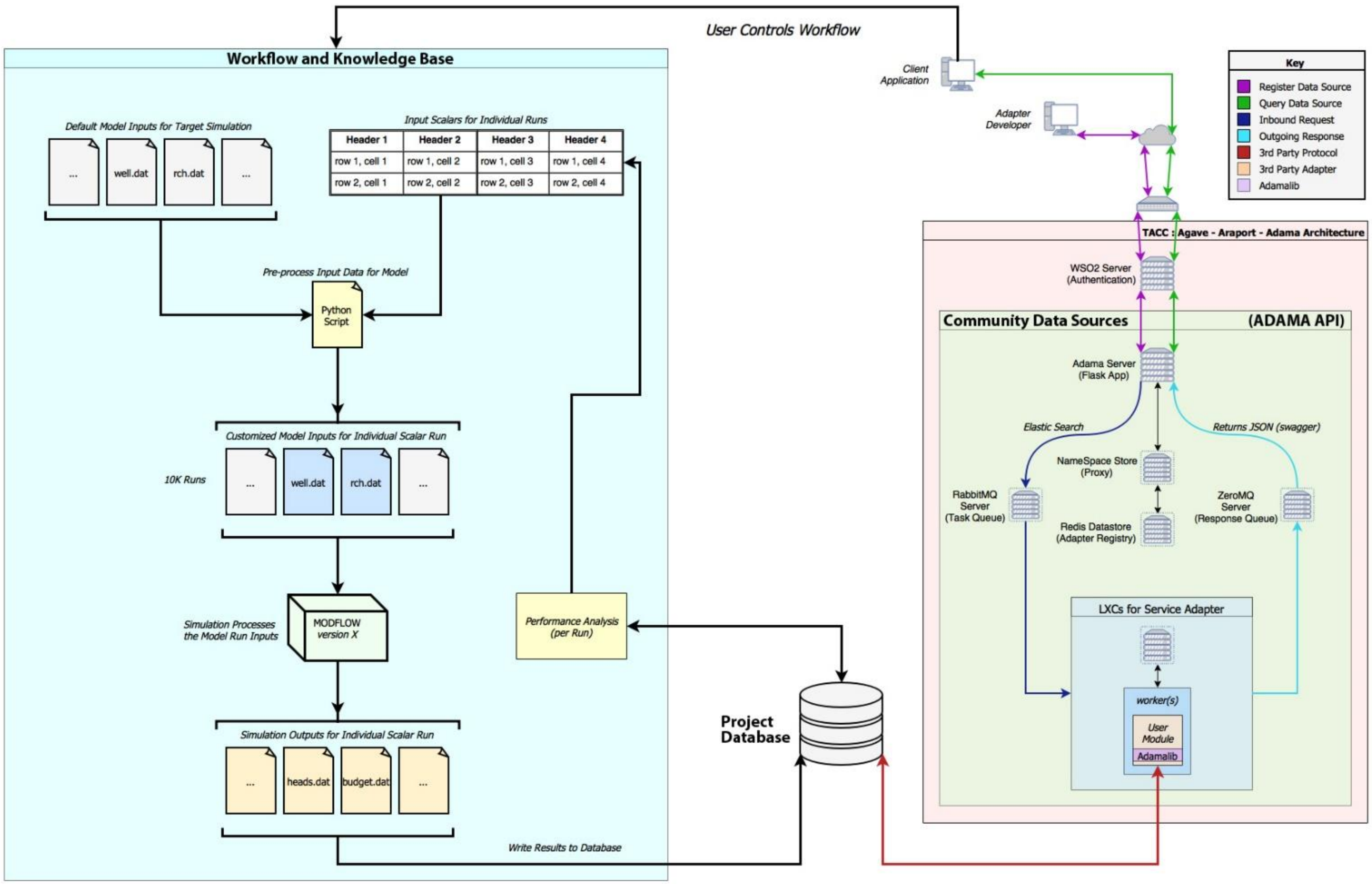


Figure 3.1 Idealized workflow architecture, modified from Gentle et al., (in prep).

Agave Core

Agave Core is an open source framework that provides hosted services allowing researchers to manage data, conduct experiments, and publish and share results from anywhere in a gridded virtual environment. Agave enables the user to focus on using the systems built on top of the grid instead of dealing with the tedious task of configuring all the individual boxes and trying to get them to communicate with each other. Agave was created as part of the Arabidopsis Information Portal, or Araport, an open access online community for plant science research. Agave is an open source, platform-as-a-service solution for hybrid cloud computing. It provides a full suite of services covering everything from standards-based authentication and authorization to computational, data, and collaborative services.

ADAMA (Araport Data and Microservices API)

ADAMA (or Adama) is a data federation strategy for the Arabidopsis Portal. Originally developed to integrate several distributed datasets published in various formats, it allows for innovative web service publication and discovery for scientific data. Adama allows researchers to put a front-end interface on their data (i.e., wrap their datasets in a RESTful API) so that their data can be accessed by other researchers or collaborators in a consistent manner using standard web protocols. Through a REST interface provided by ADAMA, data providers can document their data while users can discover the data and its provenance.

Workflow Tools

Workflows are an important part of any DSS, and the precise tool used to implement a workflow can vary. This research used several batch files that execute via SLURM (Simple Linux Utility for Resource Management), an open-source job scheduler, for designing parallelized application execution on remote computer systems.

Watermark

Developed at the Texas Advanced Computing Center, Watermark is an application geared towards visualization of multi-criteria datasets offering interactive capabilities for data in geospatial, tabular, or other graphical formats (Noll, 2013). It is intended to simplify the negotiated management of common pool resources (CPR) by combining quantifiable scientific data with qualifiable factors and social drivers derived from a topical analysis of relevant literature. Visualizing the resulting intersection between these methodologies exposes the connections between otherwise disparate aspects of the system being analyzed.

Example Job Submission Procedure

The federated data and DSS framework can be implemented and used in the following sequence:

<u>Step 1</u>: A user signs up for an account on the Arabidopsis portal and is granted an access token. The user will be authenticated through the TACC infrastructure to use the

HPC systems, or through XSEDE (an NSF program supported by TACC) for users outside of the University of Texas system.

Step 2: The user can create a namespace (an endpoint such as a URL) when leveraging ADAMA and expose his or her data through a discoverable service adapter. Anyone in the world can access this URL and consume this data via the RESTful API.

Step 3: The user will initiate a job using SLURM scripts, which may run a simulation code such as MODFLOW, or other Python or shell scripts on the HPC resources.

An adapter[5] for accessing the data in that database is coded in Python for ADAMA. An adapter is a piece of specialized code within ADAMA that specifically handles the translation of data from various formats to JSON[6] format. At a given namespace endpoint, the RESTful API will give the user a few basic mechanisms to push (POST), pull (GET), delete (DELETE), and update (PUT/PATCH) the data. When the user queries this endpoint, he or she will get the clean, structured version of the data. When a query is made against an ADAMA adapter and no worker[7] is currently available to handle that request, it will automatically build a new worker on the fly to do the job, then kill the worker process back off to conserve system resources. Everything that happens in an ADAMA adapter is

[5] An adapter may be constructed in 4 different ways, including a passthrough endpoint that will give the user raw data formats, if necessary (e.g., as unprocessed input for another program).

[6] **Javascript Object Notation (JSON):** an open-standard, language-independent data format

[7] **Worker:** a process, or application thread, that is executing on the system

ephemeral and stateless; the only things that are persistent are the data itself, the listening adapter, and the namespace endpoint for querying the adapter.

The adapter (which lives on the TACC ADAMA production server inside its own Linux Container (LXC)) retrieves information from where the data resides (which can be anywhere as long as it is accessible online), parses it into the expected format, and passes it back through the API to the requesting application as structured data in the expected JSON format.

After data is processed, it is pushed into a persistent storage solution (by the application that is generating the data, not ADAMA). ADAMA does not process data in any sense; it simply formats the return query and passes the JSON file to the endpoint request. For example, an adapter that converts text file outputs from MODFLOW to structured data that is in clean, organized key value pairs may be written and exposed through the endpoint.

After data is post-processed and stored in the database (or exposed via an ADAMA adapter), the user will navigate to Watermark and query against the ADAMA Adapter Service to populate the Watermark dashboard with the retrieved data.

ADVANTAGES

The abstraction of data presentation allows the user to access data in uniform formats and allows developers to reduce the complexity of their application-specific data structures away from the users. The scientist collecting and generating the data will only be concerned about how data goes in and how it is generated, while the user consuming the data will only be interested in what comes back and what can be done with the returned data. Adapters handle all transformations of data going in and out. The types of transformations would be domain- and application-specific, as part of the user's query. An adapter will be written that handles the formatting of data being passed back from Watermark from a given endpoint.

Another major advantage is scalability. There will no longer be a need to wrap up a myriad of moving components under any one application, which is an ordeal for development, maintenance, and support. Each component performs its respective tasks while none of them know about each other directly; they will access common data resources, talk indirectly through messaging systems, or hyperlink to one another. This kind of modular design, called microservices, is critical for ensuring the system scales under load, which is best achieved when each piece of the technology stack can be orchestrated individually.

This Agave-ADAMA-SLURM-Watermark architecture will allow users to link non-standard data sources and abstract them into a standardized format so that they can be passed easily between requesting applications or users, making it easier to reuse applications or models through a systematic, scalable, and reproducible methodology. However, this ideal architecture is currently very high-level and will require further investigation in the near future for full implementation.

Leveraging High Performance Computing Systems

The Texas Advanced Computing Center (TACC) houses more than a dozen High Performance Computing (HPC) systems, such as Stampede, Lonestar 5, Wrangler, Corral, Maverick, Stallion, Rustler, Jetstream, Chameleon, and Ranch, each tailored to specific needs a user may have, such as cloud computing, visualization, or archival storage.

The Need for High Performance Computing

For the full computation, which was not practical on a single local machine, leveraging HPC was vital. The compute times required to generate data on an individual system (measured and extrapolated using a 2015 MacBook Pro Retina laptop) are listed in Table 3.1.

Job name	**Quantity generated**	**CPU time per file**	**Total CPU time**	**Total file size**
Input generation	9,382 files	3 minutes	470 hours	120 gigabytes
Input assembly	37,528 files	50 milliseconds	30 minutes	1.36 terabytes
Output execution	150,112 files	0.3 seconds	13 hours	4.12 terabytes

Table 3.1 Approximate summative stats on the big data. Input generation refers to generating wel.dat files from scalars (described in Chapter 2); input assembly refers to combining the select wel.dat and rch.dat (decision variables) with the other MODFLOW input files to be ready for running the simulations; output execution refers to the actual simulation runs. Per simulation run, 4 new files are generated (37,528×4=150,112).

For instance, leveraging HPC resources with reservations for multiple nodes, the team was able to run the scalar generation job (20 days' worth of computation time on a single machine) in about 2 hours.

Implementing the FDDSS Workflows on HPC

Setting up the data and system federation required multiple steps and leveraged use of various HPC resources.

This research leveraged Wrangler and Corral, two TACC systems designed for data analysis and storage, respectively. For detailed information on the architecture and how to use these systems, see the User Guides at the TACC User Portal (portal.tacc.utexas.edu/user-guides).

Compiling FORTRAN and installing MODFLOW on Wrangler was the first step necessary. A custom makefile was created in order to install the Windows-based MODFLOW-96 on the Unix-based CentOS of HPCs. A new user trying to reproduce the results of this research could take advantage of the scripts already created when using TACC resources. Refer to https://github.com/jgentle/modflow-build-scripts for instructions on building MODFLOW-96.

To use the TACC system, the user would need to set up login credentials, receive allocations, and make node reservations to run jobs. Login to HPC is done through secure shell (SSH), which allows connection to a remote computer using secure shell protocols, and through a command line interface (Figure 3.2).

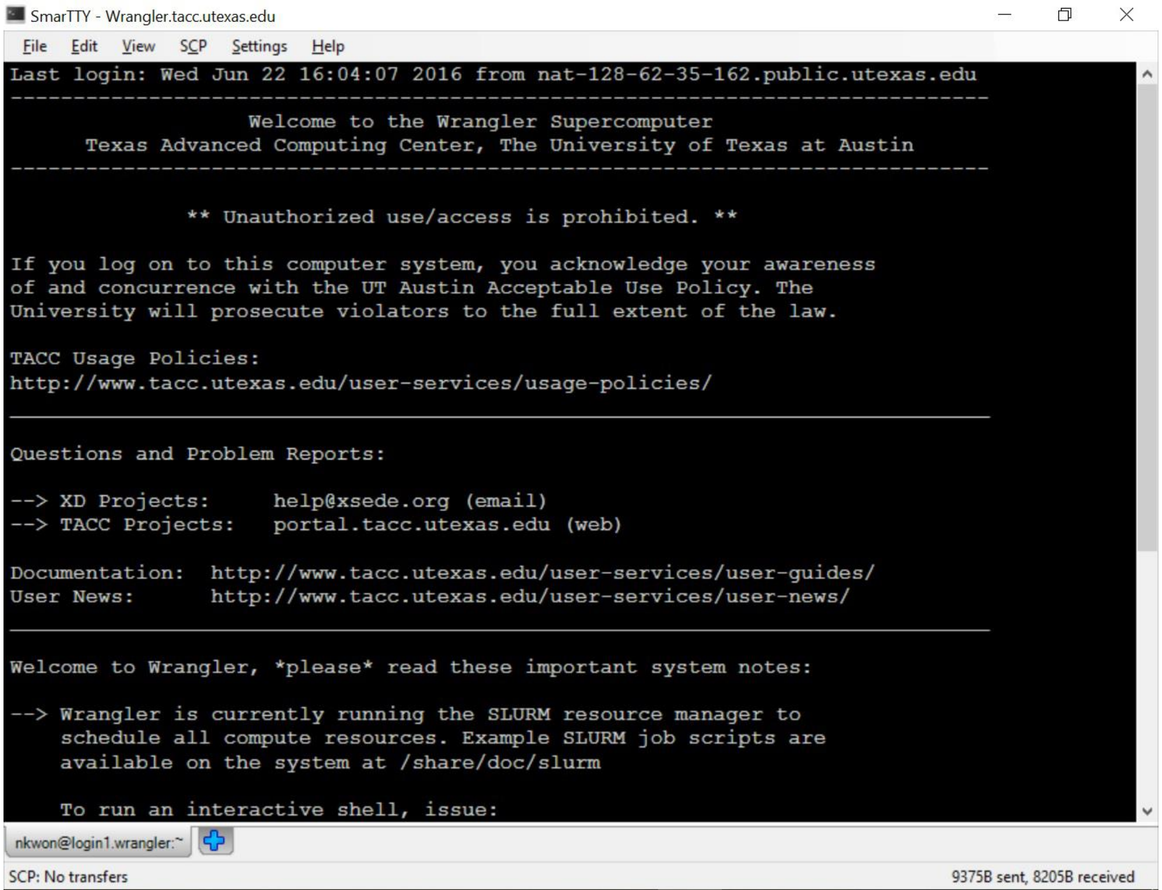


Figure 3.2 Command line interface to use HPC. On a Windows platform, an SSH client is used.

Cloning the code from the Git repository will make a local copy of the MODFLOW-96 source files via SSH into the specified directory on the local system and execute the application. Note that updating compilers and architectures will result in different numerical outputs from the executable; results were compared between HPC-built MODFLOW outputs, locally built MODFLOW outputs, and USGS-provided MODFLOW outputs. Slight numerical discrepancies exist between all of them, but differences were negligible and determined to be acceptable for the purposes of this research project which focuses heavily on the computational tool development and implementation. Figure 3.3

describes the comprehensive HPC workflow from input preparation to output generation, and all other related components. Scripts created for the Barton Springs case study and used to test the FDDSS are publicly available in a code repository at the following address: https://github.com/jgentle/msgdp.

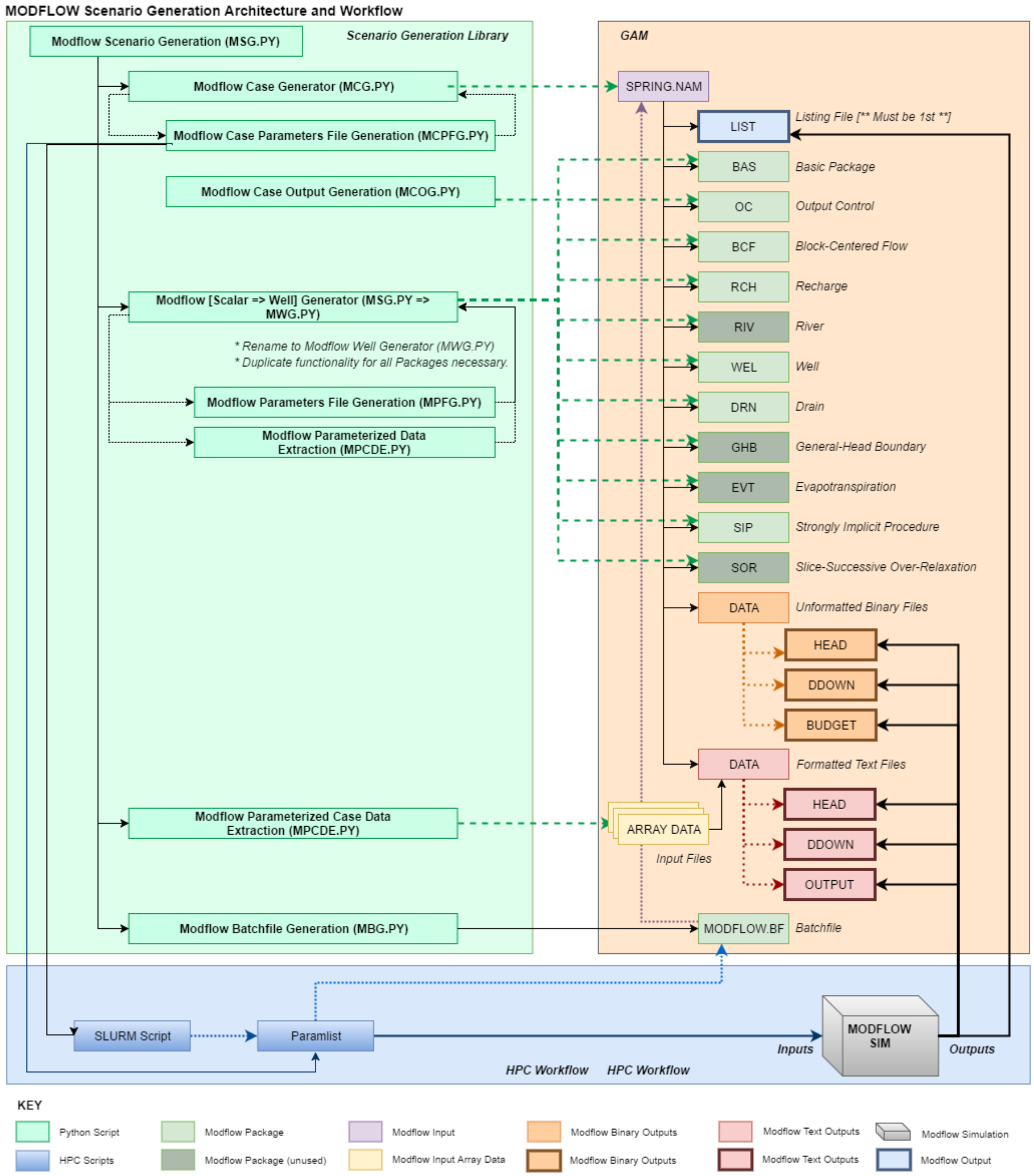


Figure 3.3 Comprehensive HPC workflow for running batch MODFLOW simulations, modified from Gentle et al., (in prep).

The first step in the workflow (generating new input pumping files from scalars) was described in the previous chapter.

Assembling the right combinations of input files was the second part of the workflow. Since only the pumping and recharge files needed to be replaced for each simulation while the other suite of MODFLOW input files remained constant for the GAM, instead of copying redundant files into ~40,000 directories, a combination of scripts that rename and relocate files was written to coordinate paths. Each simulation case is uniquely named, using the newly created scalar hash and recharge name combination. Inversely, the uniquely named wel.dat and rch.dat files were renamed to the generic format within each case for ease of processing.

Figure 3.4 shows how each file references another in a hierarchy in order to execute this workflow.

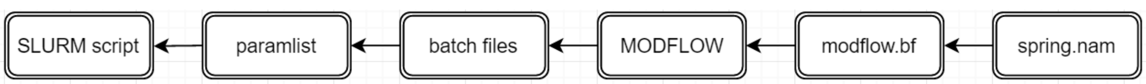


Figure 3.4 The order of reference for execution of files

The SLURM script sets up the job to run on the HPC system and defines the configurations for the job, such as the number of Normalized Units/Service Units, number of cores, duration of job, and account to charge for computation.

The paramlist file is a text file that lists the jobs that are to be run. It will point to batch files (in this case 240 to spread evenly across 240 cores[8]) and distribute the simulation execution task across the reservation resources defined in the SLURM script. Each of the

[8] There are 24 cores per node. The maximum number of nodes a user can reserve at TACC is currently 10; thus the maximum number of cores is 240.

240 batch files executed approximately 160 simulation cases (to run a total of 37,528 cases). Each executed line of the batch file changes into the directory of the target case and calls the MODFLOW-96 executable.

The MODFLOW executable finds and executes the modflow.bf (batch file). The modflow.bf file contains the name of the input file for the case (spring.nam for all cases). Normally when MODFLOW executes, it will ask for the name of the input file. Using the modflow.bf[9] file removes the step for being prompted for input file names so the user does not have to enter the name for each simulation run (or create a script that does so).

These combined Python scripts for running MODFLOW-96 on the TACC HPC systems (or any HPC system that uses Python and SLURM) will be available in the form of a Python library here: https://github.com/jgentle/msgdp. This repository will allow anyone to use the combined scripts readily from a command line interface and will be usable on any system that supports Python (for generating inputs; Unix, OS X, Windows) and SLURM (for processing; Unix).

The completed set of model results were stored on the Corral system. Finally, a script was written to parse through the output data to gather the essential metrics for case data uncertainty analyses, discussed in detail in Chapter 4.

---

[9] Although modflow.bf is a batch file designed to run multiple simulations without prompting for input, it cannot reference anything outside its own local context; it can only see a case if it is located inside the same directory. This design is mostly likely due to the computational capabilities 20 years ago when MODFLOW-96 was developed, when renaming and relocating multiple files inside a directory took much less effort than actually running the simulations. Today, (especially for the purpose of this research) it is impractical to rename and lump together tens of thousands of input files to use this batch file. Instead, a normal batch file was created to run the MODFLOW batch file, which then runs the GAM case without prompting for input.

Chapter 4: Testing Scientific Uncertainty—Analysis of Model Results

Management decisions that fail to address the uncertainties in a model can often result in unexpected or undesired outcomes (Regan et al., 2005). This research framed the issue of determining robust groundwater management and policy in the context of understanding the level of scientific certainty about a system under study. Primary efforts throughout the project were geared toward establishing the computational capabilities to enable testing and replication of simulated results for robustness. This chapter evaluates the actual robustness for the Barton Springs case study, uses the computational workflows and federated systems to extract the target metrics from a comparative set of processed MODFLOW output data to evaluate their performance based on different scientific interpretations of recharge. The results present an initial view and potential for generating robust, science-based recommendations from reusable and reproducible DSS.

The Barton Springs segment of the Edwards Aquifer Groundwater Availability Model (GAM) is a two dimensional MODFLOW model over a 10-year period (120 monthly timesteps) that combines springflow from Barton Springs and Cold Springs, which comprise 94% and 6% of total flow, respectively (Scanlon et al., 2001). Decisions about groundwater availability for the Barton Springs segment are strongly influenced by minimum flow requirements for the endangered Barton Springs salamander (*Eurycea sosorum*), whose habitable environment is restricted to aqueous areas immediately adjacent to the spring outlets and within a narrow temperature range. Water policies curtail usage and pumping depending on drought conditions. Competing interests that encompass a range of preferences from limiting water usage for the protection of endangered species to pumping more water to support the growing local population necessitate careful resource management decisions.

This study applies the computational workflows presented in previous chapters to evaluate the sensitivity of springflow to recharge input parameters using the GAM for the Barton Springs segment of the Edwards Aquifer. Sensitivity analysis is useful for comparing model uncertainties for aquifer parameters and boundary conditions (Anderson and Woessner, 1992). Out of the four parameters for which sensitivity analyses were conducted by Scanlon et al. (2001) (recharge, hydraulic conductivity, pumping, and spring drain conductance), the original GAM model responded most sensitively to changes in recharge and hydraulic conductivity and insensitively to changes in pumping and drain conductance. The model's high sensitivity to recharge, an input parameter that is directly influenced by land use decisions, results in the need to compare multiple interpretations of recharge to determine a fuller scope of management outcomes.

PREVIOUS WORK

The aim of this portion of the research was to evaluate the robustness of science-based policy and management recommendations by generating and then comparing simulation results across various pumping scenarios and recharge interpretations for the Barton Springs segment. Additionally, the research looked at how to produce a proof of concept analysis of the automation workflow to generate the inputs and outputs necessary to replicate the simulation capability of the original GWDSS. Model runs were carried out using permutations of four recharge interpretations and 9,382 pumping scenarios (total of

37,528 simulation runs). The four recharge interpretation files for the case study originated from a study by Passarello (2011), which are:

1. Recharge 1 (TROLD): Baseline scenario (Scanlon et al., 2001) calibrated with historical springflow data from 1999 to 2009. (The baseline recharge that is included in the original BS GAM simulated stress periods from 1989 to 1999, and an updated BS GAM (Smith and Hunt, 2004) simulated 1950 to 1959 to incorporate historical drought data.) This only included inputs from precipitation and losing streams.
2. Recharge 2 (TRNAT): Natural recharge scenario with newly calculated inputs from precipitation and losing streams, based on land use surveys and NEXRAD precipitation data.
3. Recharge 3 (TRMAC): Natural recharge (Recharge 2) combined with anthropogenic inputs of leaky pipes and return flow from irrigation, calculated using methods from Wiles (2007) and Hauwert (2009).
4. Recharge 4 (TRBAR): Combines natural inputs with anthropogenic inputs but alters a portion of natural inputs (diffuse recharge) using methods from Barrett and Charbeneau (1997).

The set of candidate solutions with varied pumping regimes across the Barton Springs segment first originated from a simulation-optimization setup in the original Groundwater Decision Support System (GWDSS) (Pierce, 2006). With the amount of water extraction as the input variable, the optimization algorithm had sought out solutions that maximized pumping volumes while preserving high springflow responses. As a result, a set of approximately 10,000 candidate solutions were generated. Except for the Original and Modified [10] datasets generated in 2010 before GWDSS's reversion into an abandonment state, these runs were lost with the failure to resurrect the application, but a blueprint could be salvaged from the suite of GWDSS files in the form of ~10,000 scalar[11] values, which could be multiplied to the single pumping input file (wel.dat) in the original GAM to reverse engineer the ~10,000 optimal pumping scenarios.

Using the Original and Modified datasets that GWDSS had generated before its death, subsequent research by Ballew (2015) compared monthly minimum springflow metrics from the 10,000 runs, based on two recharge interpretations, TROLD and TRBAR (Passarello, 2011) (Figure 4.1), out of which 9,382 converged[12] for both cases. Ballew (2015) plotted the differences in minimum springflow values between the two recharge interpretations from least to greatest (Figure 4.2). The area with the highest differences denotes the highest disagreement in springflow values between the two recharge scenarios. This analysis was important in the reverse engineering process of this research to target the

[10] These were the combinations of the 10,000 pumping scenarios and two recharge interpretations (TROLD and TRBAR) (Passarello, 2011). These two datasets were later analyzed more closely by Ballew (2015) and were referred to as Original (using TROLD) and Modified (using TRBAR) datasets by Ballew (2015).

[11] The ~10,000 optimal pumping scenarios generated by GWDSS were a product of a single pumping file (wel.dat) in the original BS GAM multiplied by ~10,000 different combinations of weighted values, which will be referred to in this research as scalars. See Chapter 2 for more information on scalars.

[12] If MODFLOW's solver package determines that the model solution is mathematically inadequate, the run will not converge.

simulation runs that were of most interest out of the 10,000 runs in terms of model uncertainty.

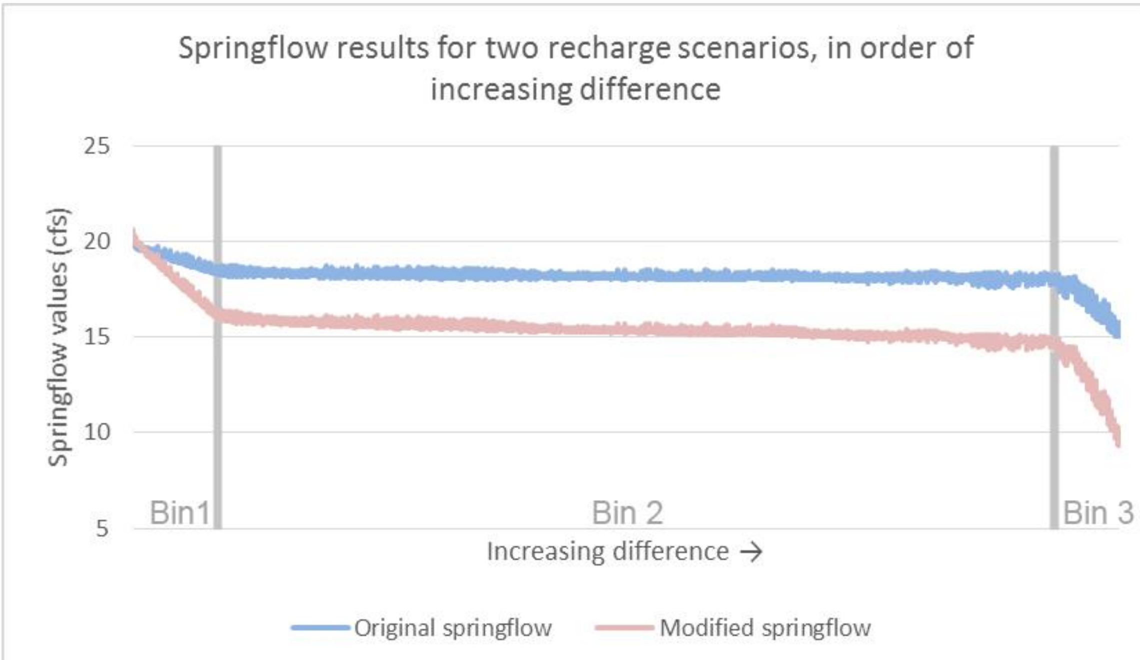


Figure 4.1 Comparison of average monthly minimum springflow values from the two sets of simulations using different recharge scenarios (Original and Modified datasets, calculated using TROLD and TRBAR recharge interpretations, respectively) (modified from Ballew, 2015). The red and blue lines each contain 9,382 average minimum springflow values from the converged simulation runs. Data points are arranged in an order of increasing difference; the left side shows little difference in minimum springflow between the two sets of simulations, while the difference becomes greater to the right. The grey vertical lines separate the plot into three bins, based on a frequency analysis of differences, which can be observed more pronouncedly in Figure 4.2.

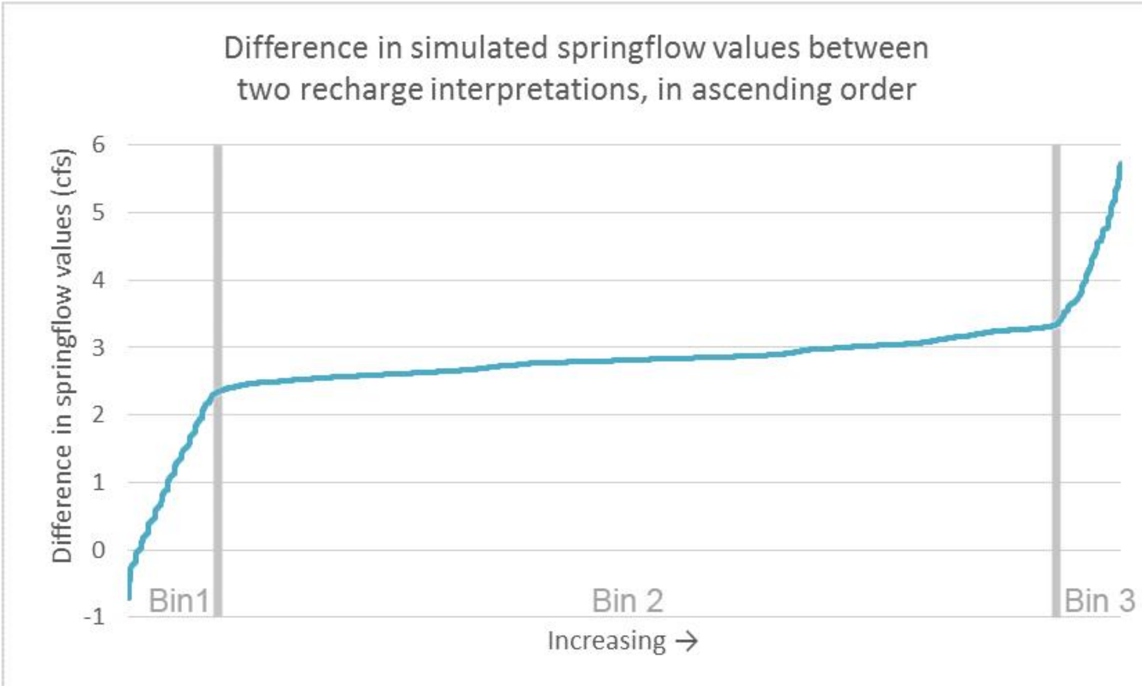


Figure 4.2 Springflow difference between two sets of simulations (shown in Figure 4.1) using the two recharge interpretations (modified from Ballew, 2015). While Figure 4.1 shows the average minimum springflow values from Original and Modified datasets, Figure 4.2 only shows the difference between them. Values are arranged on the horizontal axis in an ascending order (from least to greatest difference). The grey vertical lines separate the plot into three bins, based on a frequency analysis.

The plot was then divided into three bins (Figures 4.1 and 4.2). The cutoffs were calculated using a frequency analysis by Ballew (2015). Some difference (as can be seen in bin 2 of Figures 4.1 and 4.2) is natural because one interpretation of recharge (TROLD) only accounts for inputs from losing streams and precipitation, while the other (TRBAR) also incorporates anthropogenic inputs. The subsets of solutions in the graph that demonstrate greater difference or divergent responses (bins 1 and 3) include response levels with the greatest relevance on policy and management settings, and are of particular interest. Bin 1 shows the greatest discrepancy between the two sets over higher springflow values (Figure 4.1), which may be more relevant to flood frequency studies. Bin 3 shows high levels of discrepancy at low springflow rates. This research is more concerned with groundwater management in relation to endangered species protection and drought planning. Therefore candidate solutions in bin 3 are evaluated more closely because they indicate the modeled settings most likely to maintain minimum springflow levels necessary to sustain endangered species. These lower springflow value simulation runs with higher differences between recharge settings (bin 3) correlate to 2,544 runs out of the 37,528 total runs achieved in this research, recognized by their unique hash code, and are the most important simulation disagreements with regard to resource management decisions.

RESULTS AND DISCUSSION

The Barton Springs test case was configured with 11 decision variables for pumping rates across the aquifer (Pierce, 2006). These pumping zones were designed to match the actual hydraulic conductivity zones for Barton Springs and varied using the machine learning optimization algorithm, or metaheuristic search engine in the GWDSS (Pierce, 2006), to generate a set of pumping regimes (multiplying the original pumping settings by a numerical value ranging from 0.5 to 2, referred to as a scalar in this study**)** that can be used to test performance across scenario settings.

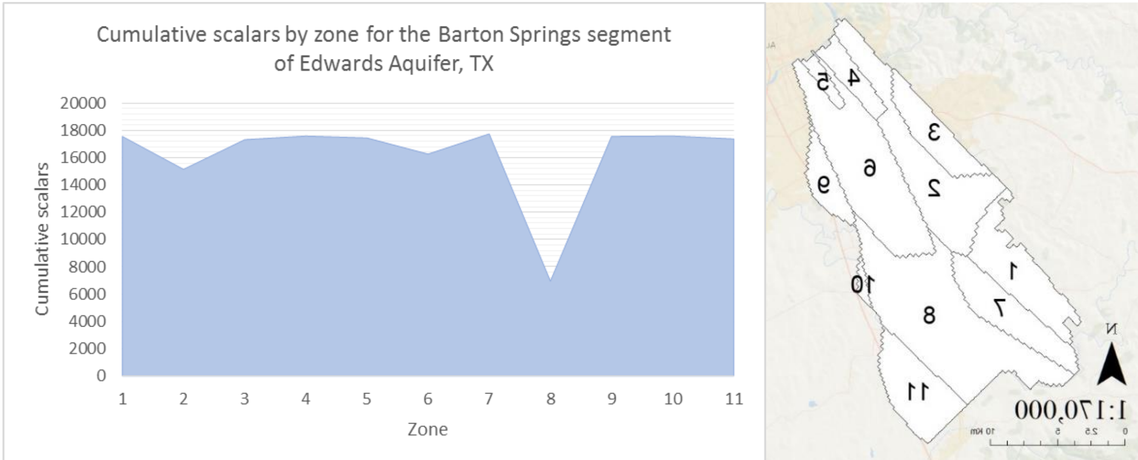


Figure 4.3 Left) cumulative scalars by hydraulic conductivity zones. Right) location of the 11 zones.

Figure 4.3 shows the pumping zone map for the case study and a graph of cumulative scalars for the 9,382 pumping inputs by zone. The actual scalar multiplier settings for each individual zone ranged from 0.5 to 2 (i.e., between 50% and 200% of the baseline pumping setting), and the graph represents the sum of all 9,382 scalar settings, or pumping scenarios, for each zone. Since the 9,382 pumping scenarios used in this study represent the top performing simulation results out of the hundreds of thousands of possible runs, the scalars are a result of machine learning of the behaviors of the best performing cases; thus a higher cumulative value signifies the zone's ability to withstand higher pumping. Zone 8 displays the least amount of cumulative scalars, meaning that the original GWDSS's optimization algorithm had learned that pumping more from zone 8 generally yielded unsatisfactory overall results. In other words, zone 8 is the most vulnerable to pumping. Zones 2 and 6 also show some sensitivity to pumping, but other zones are mostly unaffected by pumping increases. To put this into context, Figure 4.4 shows the average scalar value for each zone.

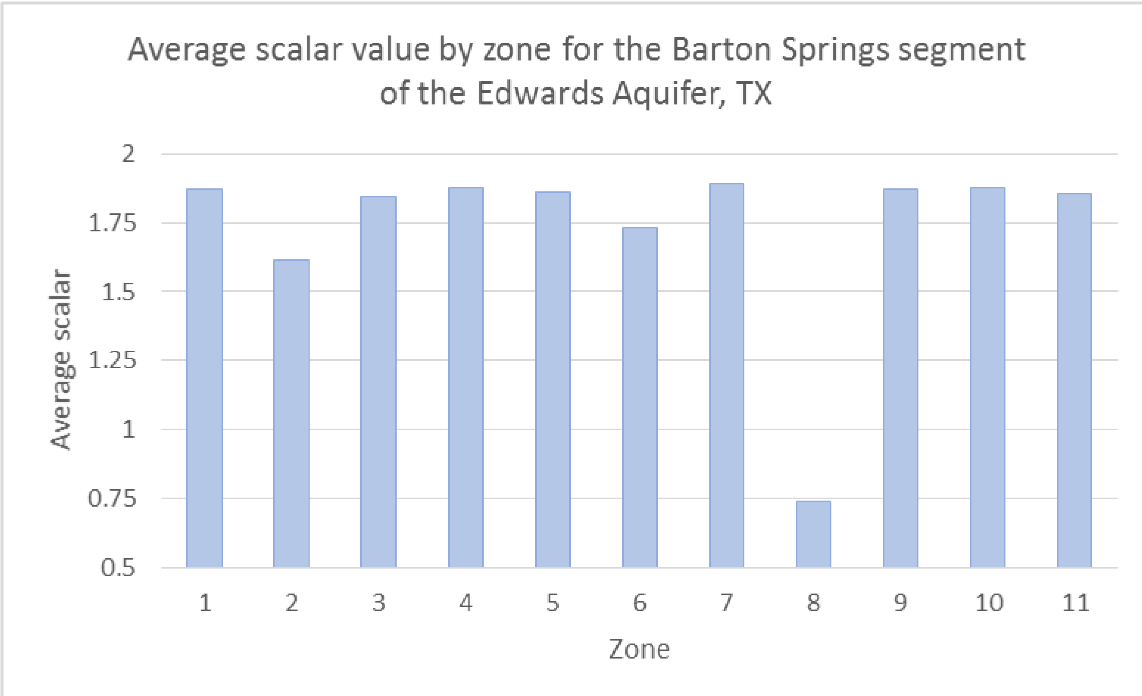


Figure 4.4 Average scalar values by hydraulic conductivity zones

The scalars range from 0.5 to 2 and are applied to the baseline pumping file (wel.dat) provided in the original Barton Springs GAM (Scanlon et al., 2001); a scalar value of 0.5 denotes a reduction in pumping by 50%, whereas a scalar value of 2 means pumping is doubled. While zones 2 and 6 do show slightly lower scalar values compared to other zones, they are over 1.5 (50% increase in pumping). It can be observed that the study area is generally not sensitive to pumping changes, and the only reduction in pumping necessary in order to achieve the optimal results was in zone 8. For the map of the 11 zones, see Figure 2.7 or Figure 4.3 above. Although not explored in this research, the observed pumping response for this particular zone may be a starting point for future discussions.

Pumping scenarios that only converged across all recharge scenarios are included in the analysis for an impartial comparison. Out of the 37,528 total runs, 30,632 converged over all four recharge scenarios. Out of the 2,544 runs that are of most interest (bin 3), 436 converged across all four recharge interpretations. This means that only a portion of the selected pumping scenario results were mathematically viable in combination with the four recharge scenarios. The average scalar values were taken from this selection (Figure 4.5). Compared with the average scalars from the entire set of 9,382 (Figure 4.4), the two graphs are nearly identical with very minor differences, signifying that pumping has little influence on the actual divergence of model behavior in estimating springflow.

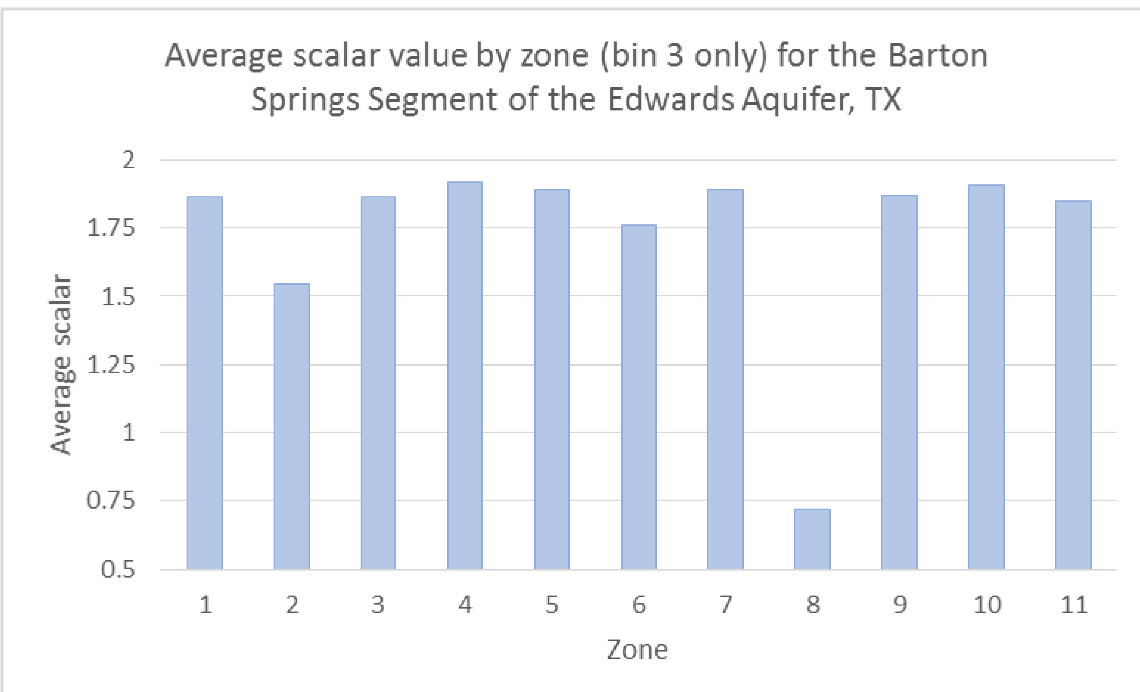


Figure 4.5 Average scalar values (of bin 3; see Figures 4.1 and 4.2) by hydraulic conductivity zones. Since bin 3 consists of simulation runs with lower minimum springflow values (Figure 4.1) and the highest degree of disagreement between the two datasets with different recharge interpretations (Figure 4.2), the similarity between the average scalars

of the entire set and the average scalars of the vulnerable subsection (bin 3) signifies that pumping has little influence on the divergence of model behavior.

Total Pumping

The main purpose of Groundwater Availability Models (GAMs) is to estimate the available yield, or the amount of water that can be extracted, for the communities that depend on the aquifer. The maximum amount of water pumped is an input variable and is distributed over the 11 spatial zones. Figure 4.6 displays the cumulative pumping over the 10-year model period for each of the four recharge scenarios for bin 3.

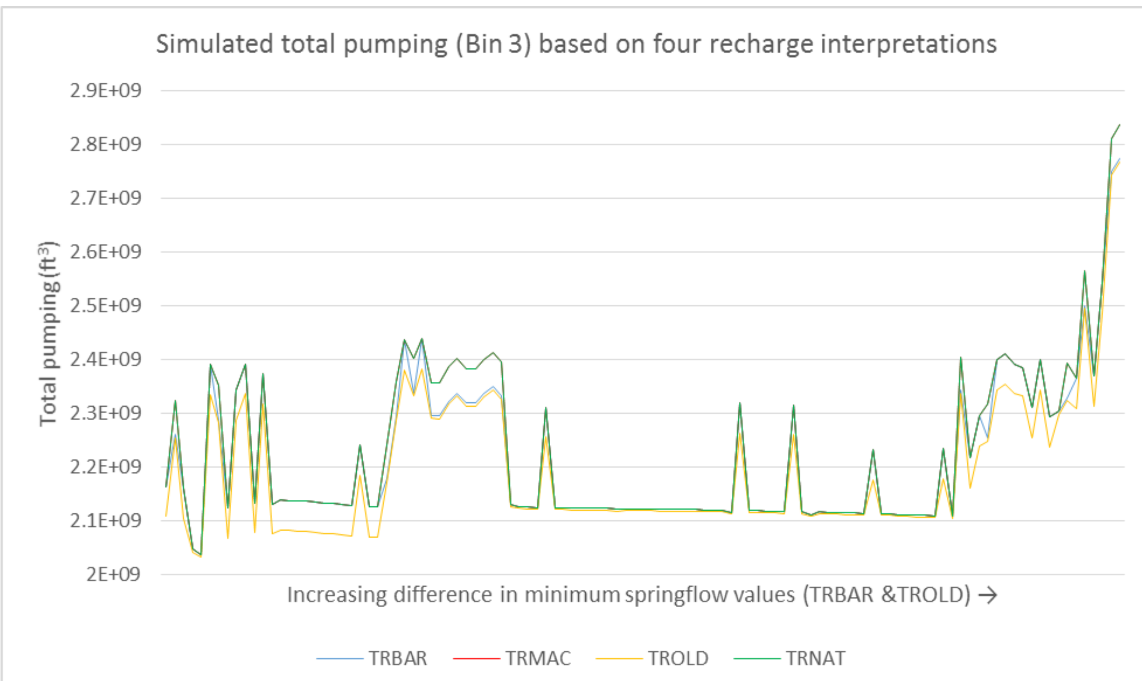


Figure 4.6 Performance of simulated pumping by recharge interpretation. Each line consists of 436 values of total pumping (volume) from the converged simulation runs. The 436 simulations correspond to a subsection (bin 3, which displayed high uncertainty over low springflow values) from the 37,528 total simulations. New data points for TRMAC and TRNAT datasets are added to the existing order of increasing difference between the minimum springflow values for the Original (using TROLD) and Modified (using TRBAR) datasets as described in Ballew (2015) and Figures 4.1 and 4.2.

Simulation runs are plotted in the same ascending order of difference between springflow responses across TRBAR and TROLD (see Figures 4.1, 4.2, and Ballew, 2015). Note that the same order of difference in minimum springflow values may not be true for TRNAT and TRMAC; the means to extract the monthly minimum metrics from the 37,528 processed outputs was not developed for this study. Instead, this addition of new data points to the existing comparative analysis should be viewed as providing greater depth of comparison to the high degree of discrepancy found in a prior study (Ballew, 2015). TRNAT and TRMAC overlap so much that TRMAC is frequently not visible. TRBAR and TROLD also overlap, but fall behind TRNAT and TRMAC slightly in the high fluctuation areas. However, most of the data points show all four overlapping (i.e., pumping the same amount of water) despite the differences in recharge. This is important because it demonstrates that in many cases across all four recharge interpretations, the storage and the springflow levels (response variables) are adequate enough to not curtail pumping (an input variable). This is consistent with the findings of Scanlon et al. (2001) that the original Barton Springs GAM is not sensitive to changes in pumping.

Storage

Storage refers to the amount of water left in the system at the end of the model period. Together with total pumping, they make up the available yield for the aquifer system.

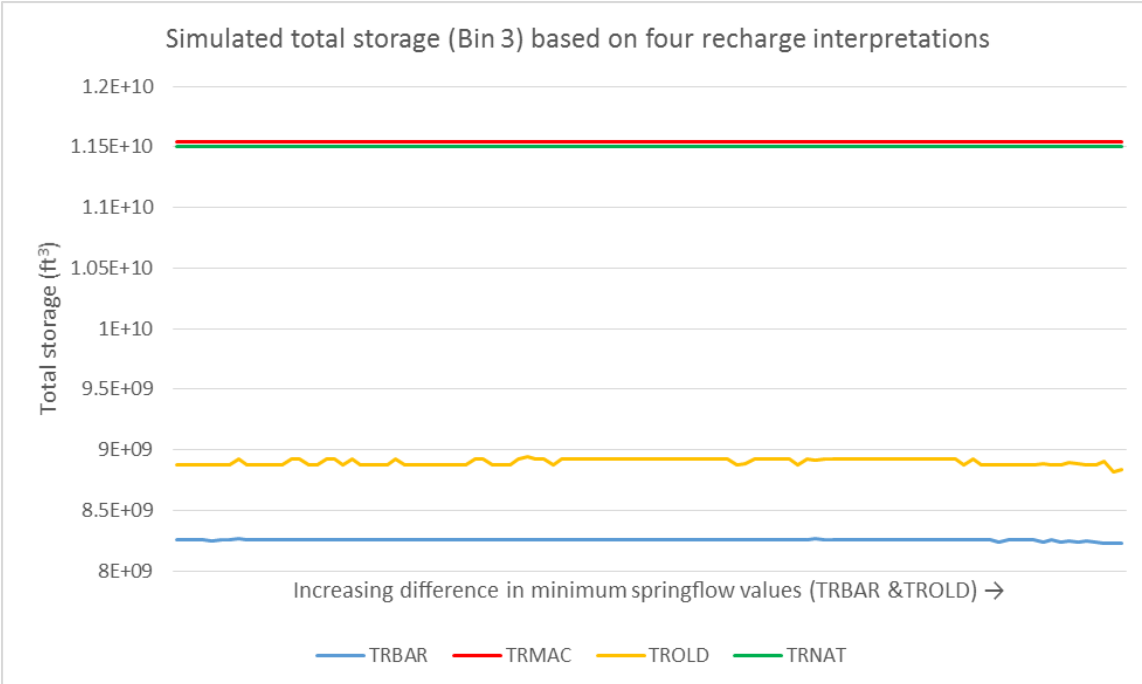


Figure 4.7 Performance of simulated storage by recharge interpretation. Each line consists of 436 values of total storage (volume) from the converged simulation runs. The 436 simulations correspond to bin 3 from the 37,528 total simulations. TRMAC and TRNAT datasets are overlaid on the existing order created from the TRBAR and TROLD datasets (increasing difference between the springflow values for TROLD and TRBAR datasets as described in Ballew (2015) and Figures 4.1 and 4.2.

Water availability across recharge scenarios for the case study aquifer can be ranked in order as follows: TRMAC ≥ TRNAT > TROLD > TRBAR. TROLD shows slight fluctuations in water level while the other three show consistent storage in the aquifer, signifying that TROLD is the most affected by pumping changes. Again, TRMAC and TRNAT performances are very similar, showing overlap in graphed results. Interestingly, TRBAR sees less availability in the aquifer than TROLD, but TRBAR results in higher overall pumping or more water extraction in some runs than TROLD (Figure 4.6). Also, there is a stark difference in the storage response levels between the higher recharge scenarios (TRMAC and TRNAT) and the lower recharge scenarios (TROLD and TRBAR).

In the decision problem formulation, pumping rates would be determined by the stakeholders and only curtailed by the DSS if the optimization algorithm found them leading to unsatisfactory results. The simulations show that while nearly the same desired amount of water can physically be extracted for the 10-year model period across all four recharge interpretations, there is high uncertainty in the amount of water that is left in the aquifer after the model period.

Springflow

Springflow is perhaps the most important performance metric for the Barton Springs case study because it is directly related to drought conditions and survival of endangered species. It is also one of the regularly monitored model metrics that can be measured in the field for direct comparison with the model results. In the case study dataset presented here, mean springflow values for the modeled period were collected for the simulations in bin 3.

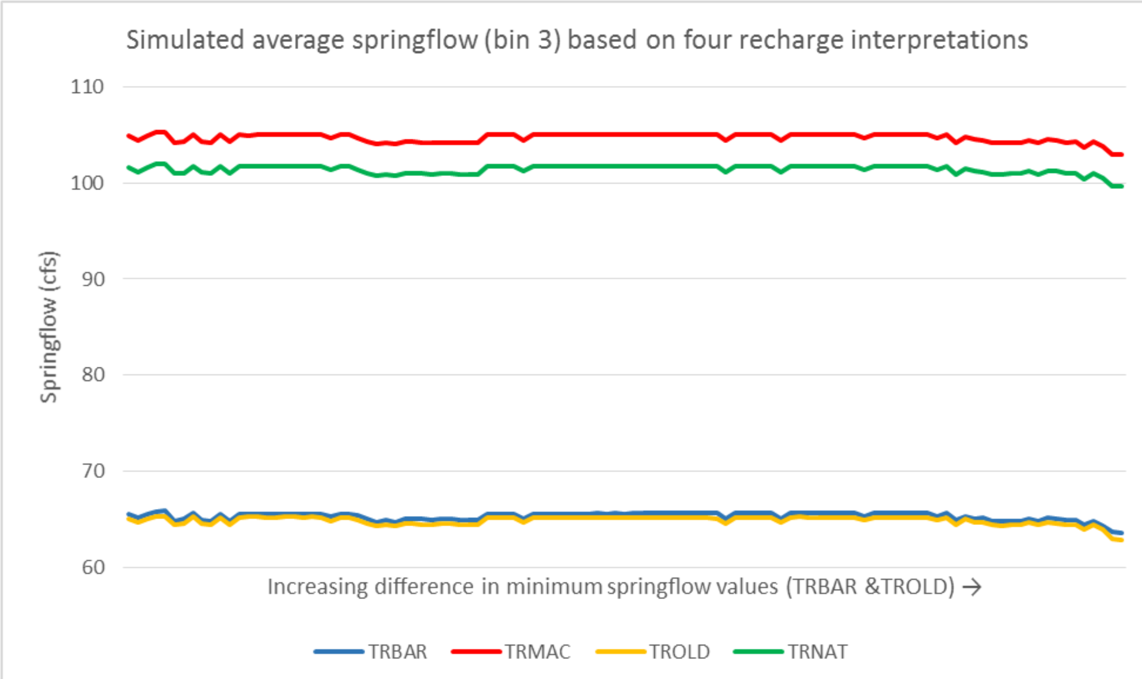


Figure 4.8 Performance of simulated springflow by recharge interpretation. Each line consists of 436 values of average springflow (rate) from the converged simulation runs. The 436 simulations correspond to bin 3 from the 37,528 total simulations. Again, TRMAC and TRNAT datasets assumed the order already created using the ranked springflow difference between TRBAR and TROLD datasets as described in Ballew (2015) and Figures 4.1 and 4.2. Note that the order is based on minimum springflow, while this graph shows mean springflow.

TRMAC and TRNAT show much higher mean springflow values than TRBAR and TROLD (Figure 4.8). Average springflow rates for TRBAR and TROLD come to about 65 cfs (1.84 m^3 s^{-1}) and around 105 cfs (2.97 m^3 s^{-1}) for TRMAC and TRNAT. These values are averaged from yearly data; actual springflow will be lower in drier months and has been known historically to dip down into the Alarm and Critical Stages in various months (Smith et al, 2013). As a frame of reference, the average simulated springflow during drought-of-record conditions of the 1950s was 53 cfs (1.50 m^3 s^{-1}) (Smith and Hunt, 2004; Hunt et al., 2010). Figure 4.9 shows the current settings for drought stage triggers based on Barton

Springs springflow as managed by the Barton Springs/Edwards Aquifer Conservation District. Drought stage 2 (Alarm Drought) is triggered if the 10-day average discharge falls below 38 cubic feet per second (cfs) (1,076 $m^3 s^{-1}$), stage 3 (Critical Drought) is triggered below 20 cfs (0.57 $m^3 s^{-1}$), and stage 4 (Exceptional Drought) when under 14 cfs (0.40 $m^3 s^{-1}$).

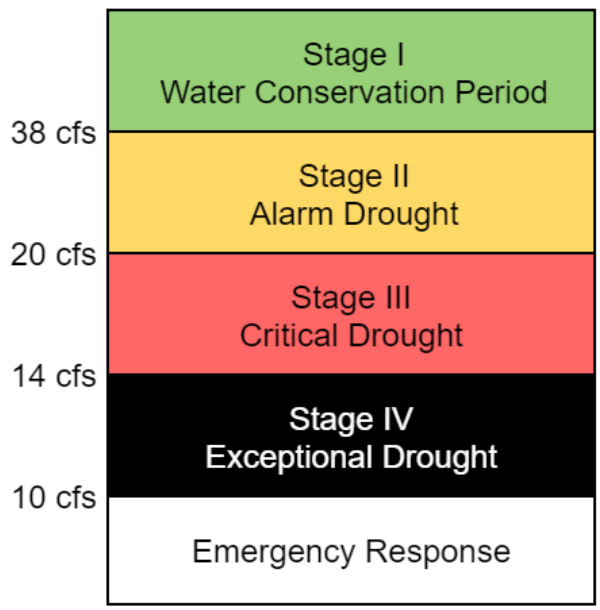


Figure 4.9 Drought status for Barton Springs, modified from BSEACD (2016)

In the analysis of monthly minimum springflow values for TROLD and TRBAR by Ballew (2015), 30 of the 120 resulting time steps fell below Alarm Stage, and 10 fell below Critical Stage. The same level of vulnerability may be assumed for TRBAR and TROLD scenarios in this study. Since based on two recharge interpretations these selected runs (bin 3) displayed the highest degree of disagreement, this study sought to compare the same subset of runs using the rest of the recharge interpretations estimated by Passarello

(2011) and their performance in order to better understand the certainty of management decisions that may rely on these GAM results. Ballew (2015) concluded that based on the findings for minimum springflow values, current policies for Barton Springs may overestimate groundwater availability. This study found that the other two recharge interpretations (TRMAC and TRNAT) by Passarello (2011) displayed a stark difference to those compared by Ballew (2015) (TRBAR and TROLD). Of the four recharge scenarios, the mean values for TRBAR and TROLD come closer to the actual observed values, while TRMAC and TRNAT seem to greatly overestimate the amount of springflow and water that is still available in the system. Based on the results found in this research, total pumping remained mostly constant across recharge scenarios and springflow differences (Figures 4.4 and 4.5), indicating that springflow did not decrease enough to curtail pumping in most zones, and thus suggesting that springflow values for all four recharge interpretations, while different, may be sufficient. This coincides with the findings of Scanlon et al., (2001) that the Barton Springs GAM is not sensitive to pumping and the fact that the ~10,000 optimal runs generated by GWDSS are the scenarios that maximized pumping while preserving a high springflow response. The range of uncertainty for the estimated storage and springflow values, however, is extremely wide (Figures 4.7 and 4.8).

Because the Barton Springs GAM was defined to be most sensitive to recharge by Scanlon et al., (2001), recharge was considered an important variable in need of multiple interpretations to find a range of values for model performance metrics. For this research, the majority of the efforts was focused on establishing a reusable workflow and computational approach that can streamline and enable future testing of various scenarios for policies and uncertainty interpretations, but future work could build on advances reported here to create a script for extracting monthly data to compare minimum flows at more resolved timesteps for detailed analyses. In conjunction with more refined temporal analyses, spatial variability may also require , especially for zone 8, which displayed a reduction in pumping to about 70% of the baseline pumping settings (Figures 4.4 and 4.5). As can be seen in Figures 4.7 and 4.8, using a single interpretation of model inputs would most likely misguide policymakers in managing our groundwater resources. Considering that both TRMAC and TRNAT used scientifically viable, peer-reviewed methodologies to calculate recharge, these findings of the high range of uncertainty between recharge interpretations attest the importance of such uncertainty analyses and calls for prudence in policy recommendations, which are, as this study suggests, unlikely to be robust. The Barton Springs case study reflects instances with high spatial heterogeneity, sensitivity to changes in surface and karst recharge features, and extreme uncertainty.

Chapter 5: Conclusions

At the highest level, this research addressed two major challenges faced by advanced computational geoscientists everywhere: reproducibility and robustness.

Reproducibility is a founding pillar of science, but one that is surprisingly difficult to uphold in the realm of computer-assisted research. Obstacles to reproducibility may occur on various levels, including the lack of backward software compatibility, insufficient documentation or curation, and restricted access to research assets necessary to replicate the experiment. Without a plan of action that coincides with the best practices for ensuring reproducible workflows, the fast-changing world of technology will make this process more and more difficult over time. As advanced technology becomes more accessible and available, parallel improvements need to occur in both the methods of scientific publications and in the perception and understanding of the importance of these best practices. Although reproducibility itself should not be the goal, it is a fundamental component of science.

Using computational hydrogeology as a topical example, this research found that the computational tools available to groundwater scientists were far from malleable to the different needs a researcher may have, other than the basic inputs around which the models were originally developed. A survey of the existing Groundwater Availability Models of the state of Texas demonstrated that the majority of GAMs for the major aquifers are still in a twice-superseded version of MODFLOW, and any meaningful conversion (in order to use some of the modern supplemental tools specifically developed for MODFLOW by USGS), even with the USGS-provided conversion utility, could not be achieved with some cases. The preparation of custom inputs for adaptable research is difficult due to the complicated input/output formats and poor documentation. In Texas, where the usage of GAMs is the result of the push towards science-vetted policymaking, the findings and lessons of this research are both appropriate and timely.

This project experienced significant challenges during the attempted resurrection process of a legacy codebase (e.g., GWDSS), demonstrating the consequences of the abandonment of important scientific software that ultimately emphasized the message of reproducibility. Upon realization that generating straightforward documentation and implementing a simple refactoring of the legacy codebase for GWDSS would not be possible, the research advanced in a revised direction to devise a reusable and reproducible computational architecture and approach with similar capabilities to the original decision support system. The newly conceptualized cyberinfrastructure aims to integrate the processing power of High Performance Computing with constantly improving web-based visualization tools while ensuring reproducibility and scalability. The original GWDSS sought to create an adaptable and flexible architecture that is open source and compatible with various data formats (Pierce, 2006). With the proposed Agave-ADAMA-SLURM-Watermark framework of services, the approach devised and presented in this research comes ever closer to the initial goals of the GWDSS effort.

Generating robust recommendations for decision makers based on scientific knowledge and data is a difficult challenge. Scientists require software architectures and utilities that can assure reuse of analytical components and data access. In this research, the modular and reusable tools created for the project were used to evaluate a complex case study. Model inputs and outputs were successfully generated using fully reproducible scripts and computational components in a new generation of loosely federated DSS. The four different but scientifically viable recharge interpretations for the Barton Springs Segment of the Edwards Aquifer were tested in order to better understand implications of scientific uncertainty as it relates to the robustness of the current policy recommendations. In conclusion, the exceptional sensitivity to recharge of the Barton Springs GAM yielded results that were highly uncertain, and consequently the policy recommendations based on modelled results for Barton Springs may not reflect scientifically robust solutions. The comparative approach used to evaluate this initial set of scenarios and simulated solutions should be further explored for the Barton Springs segment of the Edwards Aquifer and replicated for other aquifers, particularly GAMs for different aquifers in the state of Texas that can reproduce[13] the workflows, leverage the scripts, and use the HPC resources presented in this research.

The issue of outdated GAMs that are incompatible with the currently supported MODFLOW platforms is a pressing matter that needs to be resolved in the very near future. A recent Python module created by USGS called FloPy (Bakker et al., 2016) is a tool for constructing, executing, and post-processing MODFLOW models, by translating the entire model into Python code. Although it does not support MODFLOW-96, FloPy may provide a means to gain programmatic access to executing more recent versions of MODFLOW simulations. Since scripts are not prone to becoming outdated, FloPy may be a potential

[13] The tools developed for this research are usable for MODFLOW-96. Future projects will aim to expand the usability to other versions of MODFLOW, especially if the GAMs in Texas see updates to the newer model platforms.

solution for developing flexible and reproducible MODFLOW-supporting workflows in future studies.

The ultimate vision for this research is that once the computational workflows are established and the data/resources are exposed via a simple web application (Watermark) as a reusable tool, the results of which are easy to access and visualize across spatial and regulatory contexts, the testing of policy decisions against desired future conditions will become more systematic and transparent, thus facilitating the communication of science to improve the management of our groundwater resources.

Remaining challenges abound on the architectural aspect of this research, the most prominent of which is the constant battle against the technological complexities that are bound to come. Although this project sought to make the overall task of using the proposed cyberinfrastructure and reproducible workflow as simple as possible for the target user, implementing an absolutely clean, automated workflow that will operate as a single magical turnkey and still function usefully across users and domains with varying needs and interests remains an aspiration rather than a reality. As technology progresses, certain technological components will have to change, or be maintained or updated. Somewhere in between the components a human will need to understand the full complexity of the system and make necessary adjustments. Since any and all scientific workflows require alignment of the myriad moving parts, there is always complexity that can never be completely eliminated.

Following best known practices for documenting and sharing data, code, and models will be key to achieving reproduciblity. However, reproducibility in science cannot be achieved by a single team of researchers; it must be a community effort. This research has adequately demonstrated the need for such effort and hopes to aid many groundwater scientists who recognize the same challenges by laying an exemplary groundwork and providing powerful computing capabilities for meaningful analyses.

Appendix A: Chapter 1 Supplement

Summary[14] of Texas GAMs (evaluated April 2016)

GAM	State	Simulation program / version	Front end software	Cell size	Active cells	Layers
Carrizo-Wilcox (northern)	Superseded[15]	MODFLOW-96	PMWIN 5.3	1 mi^2	137,942	6
Carrizo-Wilcox (central)	Superseded[1]	MODFLOW-96	PMWIN 5.3	1 mi^2	120,477	6
Carrizo-Wilcox (southern)	Superseded[1]	MODFLOW-96	PMWIN 5.3	1 mi^2	82,896	6
Carrizo-Wilcox, Queen City, and Sparta (northern)	Current	MODFLOW-96	PMWIN 5.3	1 mi^2	170,533	8
Carrizo-Wilcox, Queen City, and Sparta (central)	Current	MODFLOW-96	PMWIN 5.3, Groundwater Vistas	1 mi^2	173,257	8
Carrizo-Wilcox, Queen City, and Sparta (southern)	Current	MODFLOW-96	PMWIN 5.3	1 mi^2	100,883	8
Edwards BFZ (northern)	Current	MODFLOW-96	PMWIN 5.3	0.25 mi^2	15,076	1
Edwards BFZ (Barton Springs)	Current	MODFLOW-96, MODFLOW-DCM	PMWIN 5.0.54	500,000 ft^2	7,046	1
Edwards BFZ (San Antonio)	Current	GWSIM-IV (1979-), MODFLOW-96 MODFLOW-2000	Groundwater Vistas	0.25 mi^2	53,129	1
Edwards-Trinity (Plateau) and Pecos Valley	Current	MODFLOW-96	PMWIN 7.0.18	1 mi^2	63,398	3 (2 active)

14 If reported in the GAM documents
15 By Carrizo-Wilcox, Queen City, Sparta GAM

Gulf Coast (GMAs 15 and 16)	To be completed					
Gulf Coast (northern)	Current	MODFLOW-2000	Groundwater Vistas	1 mi^2	134,260	4
Gulf Coast (central)	Current	MODFLOW-96	PMWIN 5.2.1, 5.3	1 mi^2	56,736	4
Gulf Coast (southern)	Current	MODFLOW-96	PMWIN 5.0.54	1 mi^2	27,007	4
High Plains	Current	MODFLOW-NWT	Groundwater Vistas	2640 ft^2		4
Hueco-Mesilla Bolsons	Current	MODFLOW-96	MODFLOWP	500,000 m^2		10
Ogallala (northern)	Superseded[16]	MODFLOW-96		1 mi^2	24,207	1
Ogallala (southern)	Superseded[2]	MODFLOW-96		1 mi^2	28,992	1
Seymour and Blaine	Current	MODFLOW-96	PMWIN 5.3	1 mi^2	23,437	2
Seymour (Haskell, Knox, Baylor counties)	Current	MODFLOW-2000		660 ft^2		
Trinity (northern) and Woodbine	Current	MODFLOW-96, MODFLOW-NWT	PMWIN	0.25 mi^2		8
Trinity (Hill Country)	Current	MODFLOW-96	PMWIN 7.0.18	1 mi^2	12,976	4
Blossom	To be completed					
Brazos River Alluvium	To be completed					
Capital Reef Complex	To be completed					
Edwards-Trinity (High Plains) and Ogallala (southern)	Superseded[2]	MODFLOW-2000	Groundwater Vistas 5.17	1 mi^2	57,873	4
Dockrum	Superseded[2]	MODFLOW-2000	Groundwater Vistas 4	1 mi^2	150,270	3
Lipan	Current	MODFLOW-96	PMWIN 5.3	0.5 mi^2	8,280	1
Llano Uplift	To be completed					
Nacatoch	Current	MODFLOW-2000	Groundwater Vistas 5.19	0.25 mi^2	75,320	
Rustler	Current	MODFLOW-NWT	Groundwater Vistas 6	0.25 mi^2	226,240	2

[16] by High Plains GAM

West Texas Bolsons (Wild Horse Flat, Michigan Flat, Ryan Flat, and Lobo Flat) and Igneous Aquifers	Current	MODFLOW-96	PMWIN 5.3	0.5 mi^2	53,829	3
West Texas Bolsons (Red Light, Green River, and Eagle Flat)	Current	MODFLOW-2000	Groundwater Vistas 5	0.5 mi^2	17,909	3
West Texas Bolsons (Presidio and Redford Bolsons)	Current	MODFLOW-2000	Groundwater Vistas 6	0.5 mi^2		3
Yegua-Jackson	Current	MODFLOW-2000	Groundwater Vistas 5.41	1 mi^2	149,882	5
Alternative Models (Recalibration / Updates from 96 to 2K)						
Bone Spring-Victorio Peak Model	Current	MODFLOW-2000		4,000,000 ft^2	48,051	1
Dockrum Alternative Model	Current	MODFLOW-2000				3
Edwards BFZ (Barton Springs) Alternative Model	Current	MODFLOW-2000		500,000 ft^2	14,400	1
Edwards-Trinity (Plateau) One Layer Model	Current	MODFLOW-2000				1
GMA 16 Model	Current	MODFLOW-2000				2
Kinney County Model	Current	MODFLOW-2000				4

Appendix B: Chapter 2 Supplement

Specifications for needed dependencies for setting up GWDSS

Dependency name and version	Function
Windows XP 32-bit with 2GB RAM	Operating System
Eclipse 3.2.1 running with Java 1.8.0	Integrated Development Environment
Java SE 6.0	Programming language
Apache Ant 1.6.5	Compile/build/deploy tool
Apache TomCat 5.5.20	Java web server environment
MySQL-Essential-5.0.21	Database
MySQL Administrator and GUI Tools	Database supplementals
JBoss 4.0.2	Java Enterprise Edition application server

Critical errors in Eclipse IDE with GWDSS import

1. The project was not built since its build path is incomplete. Cannot find the class file for java.lang.Object. Fix the build path then try building this project.
2. The type java.lang.Object cannot be resolved. It is indirectly referenced from required .class files.

References

Anaya, R., and Jones, I., 2009, Groundwater availability model for the Edwards-Trinity (Plateau) and Pecos Valley Aquifers of Texas: Texas Water Development Board, Report 373, 103 p.

Anderson, M.P., and Woessner, W.W., 1992, Applied groundwater modeling, simulation of flow and advective transport: New York, Academic Press, 381 p.

Bakker, M., Post, V., Langevin, C.D., Hughes, J.D., White, J.T., Starn, J.J., and Fienen, M.N., 2016, FloPy v3.2.4: U.S. Geological Survey Software Release, doi:10.5066/f7bk19fh.

Ballew, N.J., 2015, Data visualization as a tool for groundwater management: bridging science and policy [Master's thesis]: Austin, The University of Texas at Austin, 81 p.

Barbour M.T., Poff, N.L., Norris, R.H., and Allan, J.D., 2008, Perspective: communicating our science to influence public policy: Journal of the North American Benthological Society, vol. 27, no. 3, p. 562-569.

Barton Springs/Edwards Aquifer Conservation District, 2016, Drought Status: www.bseacd.org/aquifer-science/drought-status/ (accessed August 2016).

Beach, J.A., Ashworth, J.B., Finch, S.T., Chastain-Howley, A., Calhoun, K., Urbanczyk, K.M., Sharp, J.M., and Olson, J., 2004, Groundwater availability model for the igneous and parts of the West Texas Bolsons (Wild Horse Flat, Michigan Flat, Ryan Flat and Lobo Flat) Aquifers: prepared for the Texas Water Development Board, 407 p.

Beach, J.A., Burton, S., and Kolarik, B., 2004, Groundwater availability for the Lipan Aquifer in Texas: prepared for the Texas Water Development Board, 246 p.

Beach, J.A., Huang, Y., Symank, L., Ashworth, J.B., Davidson, T., Vreugdenhil, A.M., and Deeds, N.E., 2009, Final report: Nacatoch Aquifer groundwater availability model, prepared for the Texas Water Development Board, 304 p.

Beach, J.A., Symank, L., Huang, Y., Ashworth, J.B., Davidson, T., Collins, E.W., Hibbs, B.J., Darling, B.K., Urbanczyk, K., Calhoun, K., and Finch, S., 2008, Final report: groundwater availability model for the West Texas Bolsons (Red Light Draw, Green River Valley, and Eagle Flat) Aquifer in Texas: prepared for the Texas Water Development Board, 320 p.

Blandford, T.N., Blazer, D.J., Calhoun, K.C., Dutton, A.R., Naing, T., Reedy, R.C., Scanlon, B.R., 2003, Groundwater availability of the Southern Ogallala Aquifer in Texas and New Mexico: numerical simulations through 2050: Texas Water Development Board, 160 p.

Blandford, T.N., Kuchanur, M., Standen, A., Ruggiero, R., Calhoun, K.C., Kirby, P., and Shah, G., 2008, Groundwater availability model of the Edwards-Trinity (High Plains) Aquifer in Texas and New Mexico: Texas Water Development Board, 282 p.

Broadbent, C.D., Brookshire, D.S., Coursey, D., Cain, W., McIntosh, M., Tidwell, V., and Williams, A.A., 2008, Progress report on the Mimbres Basin: establishing a proto-type water leasing market, University of New Mexico.

Chowdhury, A.H., and Mace, R.E., 2007, Groundwater resource evaluation and availability model of the Gulf Coast Aquifer in the Lower Rio Grande Valley of Texas: Texas Water Development Board, Report 368, 119 p.

Chowdhury, A.H., Wade, S., Mace, R.E., and Ridgeway, C., 2004, Groundwater availability model of the Central Gulf Coast Aquifer system: numerical simulations through 1999: Texas Water Development Board, 108 p.

Dougal, L. H., 2006, Water availability modeling - recent developments and legal considerations, Jackson Walker L.L.P.

Deeds, N.E., Harding, J.J., Jones, T.L., Singh, A., Hamlin, S., and Reedy, R.C., 2015, Final conceptual model for the High Plains Aquifer System groundwater availability model: prepared for the Texas Water Development Board, 590 p.

Deeds, N.E., and Jigmond, M., 2015, Numerical model report for the High Plains Aquifer System groundwater availability model: prepared for the Texas Water Development Board, 640 p.

Deeds, N., Kelley, V., Fryar, D., Jones, T., Whallon, A.J., and Dean K.E., 2003, Groundwater availability model for the Southern Carrizo-Wilcox Aquifer: prepared for the Texas Water Development Board, 452 p.

Deeds, N.E., Yan, T., Singh, A., Jones, T.L., Kelley, V.A., Knox, P.R., Young, S.C., 2010, Final report: Groundwater availability model for the Yegua-Jackson Aquifer: prepared for the Texas Water Development Board, 582 p.

Dutton, A.R., Harden, B., Nicot, J.-P., and O-Rourke, D., 2003, Groundwater availability model for the central part of the Carrizo-Wilcox Aquifer in Texas: Bureau of Economic Geology, The University of Texas at Austin, 2001-483-378, 405 p.

Dutton, A.R., Reedy, R.C., and Mace, R.E., 2001, Saturated thickness in the Ogallala Aquifer in the Panhandle Water Planning Area—simulation of 2000 through 2050 withdrawal projections: Bureau of Economic Geology, The University of Texas at Austin, UTA01-462, 62 p.

Ewing, J.E., Harding J.J., and Jones, T.L., 2016, Final conceptual model report for the Brazos River Alluvium Aquifer groundwater availability model: prepared for the Texas Water Development Board, 514 p.

Ewing, J.E., Jones, T.L., Pickens, J.F., Chastain-Howley, A., Dean, K.E., and Spear, A.A., 2004, Groundwater availability model for the Seymour Aquifer: prepared for the Texas Water Development Board, 533 p.

Ewing, J.E., Jones, T.L., Yan, T., Vreugdenhil, A.M., Fryar, D.G., Pickens, J.F., Gordon, K., Nicot, J.-P., Scanlon, B.R., Ashworth, J.B., and Beach, J., 2008, Final report: groundwater availability model for the Dockrum Aquifer: Texas Water Development Board, 510 p.

Ewing, J.E., Kelley, V.A., Jones, T.L., Yan, T., Singh, A., Powers, D.W., Holt, R.M., and Sharp, J.M., 2012, Final groundwater availability model report for the Rustler Aquifer: prepared for the Texas Water Development Board, 460 p.

Fryar, D., Senger, R., Deeds, N., Pickens, J., Jones, T., Whallon, A.J., and Dean, K.E., 2003, Groundwater availability model for the Northern Carrizo-Wilcox Aquifer: prepared for the Texas Water Development Board, 529 p.

George P.G., Mace, R.E., and Petrossian, R., 2011, Aquifers of Texas: Texas Water Development Board, Report 380, 172 p.

Gentle, J.G., 2016, MSGDP: MODFLOW96 scenario generation and data processing Python module for TACC: https://github.com/jgentle/msgdp, doi:10.5281/zenodo.60735.

Gentle, J.G., Kwon, N., and Pierce, S.A., Idealized workflow and architecture for running groundwater simulations in High Performance Computing environments (in prep).

Gil, Y., Ratnakar, V., and Garijo., D., 2015 OntoSoft: Capturing Scientific Software Metadata, *in* Proceedings, Eighth ACM International Conference on Knowledge Capture, Palisades, NY, October 2015.

Harbaugh, A. W., Banta, E. R., Hill, M. C., and McDonald, M. G., 2000, MODFLOW-2000, the U.S. Geological Survey modular ground-water model—user guide to modularization concepts and the ground-water flow process: U.S. Geological Survey, U.S. Geological Survey Open-File Report 00-92, 121 p.

Harbaugh, A. W., and Banta, E. R., 2005, MODFLOW-2005, the U.S. Geological Survey modular ground-water model—the ground-water flow process: U.S. Geological Survey Techniques and Methods, book 6, chap. A16, 253 p.

Harbaugh, A. W., and McDonald, M. G., 1996, User's document for MODFLOW-96, an update to the U.S. Geological Survey modular finite-difference ground-water flow model, US Geological Survey Open-File Report 96-485, 56 p.

Heywood, C.E., and Yager, R.M., 2003, Simulated ground-water flow in the Hueco Bolson, an alluvial-basin aquifer system near El Paso, Texas: U.S. Geological Survey Water-Resources Investigations Report 02-4108, 73 p.

Hutchison, W.R., 2008, Preliminary groundwater flow model: Dell City area, Hudspeth and Culberson Counties, Texas: El Paso Water Utilities Hydrogeology Report 08-01, 435 p.

Hutchison, R.W., and Hill, W.R., 2011, Report: recalibration of the Edwards (Balcones Fault Zone) Aquifer—Barton Springs segment—groundwater flow model: Texas Water Development Board, 115 p.

Hutchison, W.R., Hill, M.E., Anaya, R., Hassa, M.M., Oliver, W., Jigmond, M., Wade, S., Aschenbach E., 2011, Groundwater Management Area 16 groundwater flow model: Texas Water Development Board, 296 p.

Hutchison, W.R., Jones, I.C., Anaya, R., 2011, Update of the groundwater availability model for the Edwards-Trinity (Plateau) and Pecos Valley Aquifers of Texas: Texas Water Development Board, 59 p.

Hutchison, W.R., Shi, J., and Jigmond, M., 2011, Groundwater flow model of the Kinney County area: Texas Water Development Board, 219 p.

Hunt, B.B., Smith, B.A., and Holland, W.F., 2010, Information in support of the Drought DFC and Drought MAG: BSEACD Technical Note, 2011-0707, 5 p.

Jigmond, M., Hutchison, W.R., and Shi, J., 2014, Final report: groundwater availability model of the Seymour Aquifer in Haskell, Knox, and Baylor Counties: Texas Water Development Board, 185 p.

Jones, I.C., 2003, Groundwater availability modeling: northern segment of the Edwards Aquifer, Texas: Texas Water Development Board, Report 358, 75 p.

Jones, I.C., 2016, Groundwater availability model: eastern arm of the Capitan Reef Complex Aquifer of Texas: Texas Water Development Board, 480 p.

Jones, I.C., Anaya, R., Wade, S., 2009, Groundwater availability model for the Hill Country portion of the Trinity Aquifer System, Texas: Texas Water Development Board, 195 p.

Kasmarek, M.C., 2012, Hydrogeology and simulation of groundwater flow and land-surface subsidence in the northern part of the Gulf Coast Aquifer System, Texas, 1891-2009: U.S. Geological Survey Scientific Investigations Report 2012-5154, 55 p.

Kelley, V.A., Deeds, N.E., Fryar, D.G., and Nicot, J.-P., 2004, Groundwater availability models for the Queen City and Sparta Aquifers: prepared for the Texas Water Development Board, 867 p.

Kelley, V.A., Ewing, J., Jones, T.L., Young, S.C., Deeds, N., and Hamlin, S., 2014, Updated groundwater availability model of the Northern Trinity and Woodbine Aquifers: prepared for North Texas GCD, Northern Trinity GCD, Prairielands GCD, and Upper Trinity GCD, 990 p.

Kwon N., Gentle, J.G., and Pierce, S.A., 2015, Resurrecting Legacy Code Using OntoSoft Knowledge-Sharing and Digital Object Management to Revitalize and Reproduce Software for Groundwater Management Research, *in* technical session IN53B Informatics and Intelligent Systems in Reproducible Geoscience Research III Posters, American Geophysical Union Fall Meeting, San Francisco, CA, December 2015, doi:10.6084/m9.figshare.2014917.

Kwon N., Gentle, J.G., and Pierce, S.A., 2016, Virtual machine image file for GWDSS, doi:10.5281/zenodo.60733.

Lindgren, R.J., Dutton, A.R., Hovorka, S.D., Worthington, S.R.H., and Painter, S., 2004, Conceptualization and simulation of the Edwards Aquifer, San Antonio region, Texas: U.S. Geological Survey Scientific Investigations Report 2004-5277, 143 p.

Liu, Y., Gupta, H., Springer, E., and Wagener, T., 2008, Linking science with environmental decision making: experiences from an integrated modeling approach to supporting sustainable water resources management: Environmental Modeling & Software, v. 23, p. 846-858, doi:10.1016/j.envsoft.2007.10.007.

Loudyi, D., Falconer, R., and Lin, B., 2014, MODFLOW: an insight into thirty years development of a standard numerical code for groundwater simulations, *in* Proceedings, International Conference on Hydroinformatics, 11th, New York, NY, August 2014.

Mace, R.E., Chowdhury, A.H., Anaya, R., and Way, S.-C., 2000, Groundwater availability of the Trinity Aquifer, Hill Country area, Texas: numerical simulations through 2050: Texas Water Development Board, Report 353, 117 p.

Mace, R.E., Petrossian, R., Bradley, R., Mullican, W. F., and Christian, L., 2008, A streetcar named desired future conditions: the new groundwater availability for Texas (revised), in 9th Annual: The changing face of water rights in Texas, Bastrop, May 2008.

McCown, R.L., 2002, Changing systems for supporting farmers' decisions: problems, paradigms, and prospects: Agricultural Systems, vol. 74, no. 1, p. 179-220, doi: 10.1016/s0308-521x(02)00026-4.

McDonald, M.G., and Harbaugh, A.W., 1984, A modular three-dimensional finite difference ground-water flow model: U.S. Geological Survey, Open File Report 83-875.

McDonald, M.G., and Harbaugh, A.W., 1988, A modular three-dimensional finite difference ground-water flow model: U.S. Geological Survey, Techniques of Water Resource Investigations, book 6, chapter A1.

McIntosh, B.S., Jeffrey, P., Lemon, M., and Winder, N., 2005, On the design of computer-based models for integrated environmental science: Environmental Management, vol. 35, no. 6, p. 741-752, doi:10.1007/s00267-004-0172-0.

Michaels, S., 2009, Matching knowledge brokering strategies to environmental policy problems and settings: Environmental Science & Policy, vol. 12, p. 994-1011, doi: 10.1016/j.envsci.2009.05.002.

Mullican, W.F., and Schwartz, S., 2004, 100 years of rule of capture: from east to groundwater management: Texas Water Development Board, Report 361, 174 p.

Noll, D.J., 2013, A spatial decision support framework for web-based, multi-stakeholder engagement: case study of geothermal power project siting in Idaho [Master's thesis]: Austin, The University of Texas at Austin, 140 p.

Oliver, W.A., and Hutchison, W.R., 2010, Modification and recalibration of the groundwater availability model of the Dockrum Aquifer: Texas Water Development Board, 114 p.

Parker, P., Letcher, R., Jakerman, A., Beck, M.B., Harris, G., Argent, R.M., Hare, M., Pahl-Wostl, C., Voinov, A., Janssen, M., Sullivan, P., Scoccimarro, M., Friend, A., Sonnenshein, M., Barker, D., Matejicek, L., Odulaja, D., Deadman, P., Lim, K., Larocque, G., Tarikhi, P., Fletcher, C., Put, A., Maxwell, T., Charles, A., Breeze H., Nakatani, N., Mudgal, S., Naito, W., Osidele, O., Eriksson, I., Kautsky, U., Kautsky, E., Naeslund, B., Kumblad, L., Park, R., Maltagliati, S., Giarrdin, P., Rizzoli, A., Mauriello, D., Hoch, R., Pelletier, D., Reilly, J., Olasfsdottir, R., and Bin, S., 2002, Progress in integrated assessment and modelling: Environmental Modelling & Software, vol. 17, p. 209-217, doi: 10.1016/s1364-8152(01)00059-7.

Passarello, M.C., 2011, New methods for quantifying and modeling estimates of anthropogenic and natural recharge: a case study for the Barton Springs segment of the Edwards Aquifer, Austin, Texas [Master's Thesis]: Austin, The University of Texas at Austin, 185 p.

Passarello, M.C., Pierce, S.A., and Sharp, J.M., 2014, Uncertainty and urban water recharge for managing groundwater availability using decision support: Water Science & Technology Journal, vol. 70, no. 11, p 1888-1896, doi:10.2166/wst.2014.437.

Pierce, S.A., Sharp, J.M, and Eaton, D.J., 2016, An overview of groundwater decision support systems, *in* Jakeman, A.J., Barrateau, O., Hunt, R.J., Rinaudo, J.-D., and Ross, A., eds., Integrated groundwater management: concepts, approaches and challenges: Springer International Publishing, p. 639-665.

Pierce S.A., Dulay, M., Sharp, J.M., Lowry, and T.S., Tidwell, V.C., 2006, Defining tenable groundwater management: integrating stakeholder preferences, distributed parameter models, and systems dynamics to aid groundwater resource allocation, *in* Proceedings, MODFLOW and More, 4th, Golden, CO, May 2006, p 409-413.

Pierce, S.A., 2006, Groundwater decision support: linking causal narratives, numerical models, and combinatorial search techniques to determine available yield for an aquifer system [Ph.D. thesis]: Austin, The University of Texas at Austin, 313 p.

Pierce, S.A., Sharp, J.M., Guillaume, J.H.A., Mace, R.E., and Eaton, D.J., 2013, Aquifer-yield continuum as a guide for science-based groundwater management: Journal of Hydrogeology, vol. 21, no. 2, p 331-340, doi:10.1007/s10040-012-0910-y.

Poch, M., Comas, J., Rodriguez-Roda, I., Sanchez-Marre, M., Cortes, U., 2004, Designing and building real environmental decision support systems: Environmental Modeling & Software, vol. 19, p. 857-873, doi:10.1016/j.envsoft.2003.03.007.

Policansky, D., 1998, Science and decision making for water resources: Ecological Applications, vol. 8, no. 3, p. 610-618.

Regan, H.M., Ben-Haim, Y., Langford, B., Wilson, W.G., Lundberg, P., Andelman, S.J., and Burgman, M.A., 2005, Robust decision-making under severe uncertainty for conservation management: Ecological Applications, vol. 15, no. 4, p. 1471-1477.

Rizzoli, A.E., and Young, W.J, 1997, Delivering environmental decision support systems: software tools and technique: Environmental Modeling & Software, vol. 12, nos. 2-3, p. 237-249.

R.W. Harden & Associates, Inc., 2004, Northern Trinity / Woodbine Aquifer groundwater availability model: prepared for the Texas Water Development Board, 391 p.

Scanlon, B.R., Mace, R.E., Smith, B.A., Hovorka, S.D., Dutton, A.R. and Reedy, R.C., 2001, Groundwater availability of the Barton Springs segment of the Edwards Aquifer, Texas: numerical simulations through 2050: Bureau of Economic Geology, The University of Texas at Austin, UTA99-0.

Shi, J., Boghici, R., Kohlrenken, W., and Hutchison, W., 2016, Conceptual model report: minor aquifers in Llano Uplift Region of Texas: Texas Water Development Board, 291 p.

Smith, B.A., Hunt, B.B., and Holland, W.F., 2013, Drought Trigger Methodology for the Barton Springs Aquifer, Travis and Hays Counties, Texas: BSEACD Report of Investigations, 2013-1201, 35 p.

Wade, S.C., Hutchison, W.R., Chowdhury, A.H., and Coker, D., 2011, A conceptual model of groundwater flow in the Presidio and Redford Bolsons Aquifers: Texas Water Development Board, 98 p.

Wade, S.C., and Jigmond, M., 2013, Groundwater availability model of West Texas Bolsons (Presidio and Redford) Aquifer: Texas Water Development Board, 100 p.

Winston, R.B., 2014, Modifications made to ModelMuse to add support for the saturated-unsaturated transport model (SUTRA): U.S. Geological Survey Techniques and Methods, book 6, chap. A49, 6 p., doi:10.3133/tm6a49.

Winston, R.B., 2009, ModelMuse—a graphical user interface for MODFLOW-2005 and PHAST: U.S. Geological Survey Techniques and Methods, book 6, chap. A29, 52 p.