DEVELOPMENT OF COMPUTER PROGRAMS FOR FAST COMPUTATION OF G-FUNCTIONS AND AUTOMATED GROUND HEAT EXCHANGER DESIGN

By

JONATHAN CHRISTOPHER COOK

Bachelor of Science in Mechanical Engineering.
Oklahoma State University
Stillwater, OK
2019

Submitted to the Faculty of the
Graduate College of
Oklahoma State University
in partial fulfillment of
the requirements for
the Degree of
MASTERS OF SCIENCE
December, 2021

DEVELOPMENT OF COMPUTER PROGRAMS FOR FAST COMPUTATION OF G-FUNCTIONS AND AUTOMATED GROUND HEAT EXCHANGER DESIGN

Thesis Approved:

Dr. Jeffrey D. Spitler

Thesis Advisor

Dr. Christian K. Bach

Dr. Ronald D. Delahoussaye

ACKNOWLEDGMENTS

Advances in knowledge build on previous work. A collection of individuals likely can make more progress than a single individual. There are a number of people I would like to acknowledge; both those who have come before me and those who had a positive impact on this work. First though, I must give the glory to God. For without His creation, this work would not have been possible. "So whether you eat or drink, or whatever you do, do all to the glory of God." 1 Corinthians 10:31

I'd like to thank those who have shed blood sweat and tears so that I can work on a computer, live in an air conditioned structure and store my food in a refrigerator. These things we often take for granted. Lots of hard work and sacrifices were required to provide us with the productivity tools we have today.

Without successful completion of my undergraduate Bachelors of Science degree in Mechanical Engineering, I likely wouldn't have had this opportunity. My undergraduate colleagues were a huge help in studying and performing well in group projects. I'd specifically like to acknowledge Gerardo Toth and Garrett Weber for their hard work, often serving as encouragement. My senior design teammates (Jake McDonald, Cole Straight, Lujia Wang) and our mentors Dr. Craig Bradshaw, Dr A. Santhanakrishnan and Mrs. Laura Southard played a significant role in winning multiple senior design awards.

I'd like to acknowledge specific faculty at Oklahoma State University (OSU) who I believe are excellent. I garnered a foundation in programming in Dr. Ronald Delahoussaye's computer methods course in Python, and went on to work with him

Acknowledgments reflect the views of the author and are not endorsed by committee members or Oklahoma State University.

as an undergraduate teaching assistant. I was introduced to C++ while performing undergraduate research with Dr. Omer San and his then PhD candidate Dr. Romit Maulik. I worked on multiple projects with Dr. Bradshaw where I was able to write code, and much of my understanding of thermodynamics comes from him. He recognized my potential and recommended me for this work. Dr. Dan Fisher was helpful on more than one occasion during my time as an undergraduate, and wrote a letter of recommendation to the graduate college based on my previous performance (e.g. winning multiple senior design awards). Much thanks to the many mentors I have had while at OSU.

I'd like to thank my father for providing me with a "dad" scholarship, which required that I study engineering, maintain a predetermined GPA and finish within 4 years. Throughout my 4 years of undergrad at OSU, I benefited greatly from having my grandparents within an hour drive away.

During the 1980's and 1990's, Prof. Johan Claesson of Lund University in Sweden and his research assistants pioneered the work on thermal response functions (g-functions). Dr. Massimo Cimmino of the Polytechnique Montreal studied their work, recreated it, furthered it and then released his program as an open-source project. A monumental feat. My gratitude goes out to those who have made prior progress in this field.

My adviser, Dr. Jeffrey Spitler, has been vital to the project. I am grateful for his guidance over the last two and a half years. Dr. Cimmino has collaborated on more than one occasion to enhance *pygfunction* in ways that have greatly benefited this project. Dr. Xiaobing Liu performed extensive testing on a code base that is the predecessor to the *Ground Heat Exchanger Design Toolbox* (GHEDT) I have developed. Mr. Timothy West was hired as an undergraduate research assistant after gaining funding based on my work with cpgfunction and the adaptive discretization

Acknowledgments reflect the views of the author and are not endorsed by committee members or Oklahoma State University.

scheme, and has been a major help in what I call "proof of concept" ideas. While I was busy developing code so that I could meet deadlines, Mr. West assisted in much needed research. On more than one occasion, his and Dr. Spitler's results guided what I would develop next. This work has greatly benefited from more than 281 years of computing time on a recently built high performance computing cluster (HPCC) here at OSU. I do not believe the OSU HPCC would be as powerful if it were not for Mr. Jesse Schafer's work. On a system that large, there is an enormous amount of maintenance and upkeep that is required, especially to get it setup in the first place. There were many occasions that Mr. Schafer performed maintenance and improved functionality of the cluster so that I and others could continue research. Drs. Edwin Lee and Matt Mitchell were crucial to the integration of *cpgfunctionEP* into EnergyPlus. Both my father Mr. Steven C. Cook and my brother Mr. James A. Cook have provided ideas to further this project along. Dr. Christian Bach was very thorough in reading drafts of this thesis, and gave excellent feedback during the editing process. Mrs. Evin Cook also helped with editing of this thesis near the end when time was limited. I am very thankful for the team that has helped overcome the many challenges faced in this project.

At the start of COVID-19 (Spring of 2020) lock downs my parents welcomed me back in their home free of rent. I was able to save money, and was more productive. My first major boost of productivity occurred over a 2 month period of time in the summer of 2020, when I first developed *cpgfunction.* My mother kept me fed, and the refrigerator full (just 10 steps from my room). Therefore, during those 2 months I was able to stay "plugged in to the matrix". Both my immediate family and my grandparents provided great support during this period of time.

I'm thankful for the people I met while living back at home with my parents on my daily bike ride up to the local track and field. Along the way I would often get

Acknowledgments reflect the views of the author and are not endorsed by committee members or Oklahoma State University.

to see Mr. and Mrs. (Paul and Jill) Bukowski; two wonderful and hard working individuals. I met and then saw Ed Sorensen nearly every day at the track. He gave me a lot to laugh about and could take my mind off work for a bit. His wisdom is nearly unparalleled. I'm thankful for all the wonderful people back home I got to see while living there.

Soon after returning to live in Stillwater, OK in February 2021, I met my wife, Mrs. Evin Cook. Evin's parents, Greg and Traci, have taken me in like their own, and have been very supportive. My second and final major boost of productivity in this project occurred after Evin and I got married in July. The significant amount of development time put into GHEDT was made possible in large part because she kept me fed, and I could stay focused on my work.

The work detailed in this thesis and computer programs I have written have been helped along by many people. I am thankful for those who have played a role.

Finally, I would not have partaken in this research had it not been funded. The work was initially financed by the U.S. Department of Energy (DOE) through a research subcontract from Oak Ridge National Laboratory (ORNL). The contract was initially year-to-year. At the end of the first year, there was a break in the contract. Thankfully, the OG&E Energy Technology Chair and Oklahoma State University via return of indirect costs to Dr. Spitler provided funding during the lapse in contract, which ended up lasting for 2 months. During these 2 months, the initial version of *cpgfunction* was written as well as an adaptive discretization scheme. The work completed throughout these 2 months were of interest and resulted in new funding from the DOE through a research subcontract from the National Renewable Energy Laboratory and by OSU through the Center for Integrated Building Systems (CIBS). Thankfully, the DOE through ORNL resumed funding after that 2 month lapse. I greatly appreciate the support of OSU and the U.S. DOE, along with our project

Acknowledgments reflect the views of the author and are not endorsed by committee members or Oklahoma State University.

managers (Arlene Anderson, Xiaobing Liu, Edwin Lee and Luigi Gentile Polese, Sr.).

Acknowledgments reflect the views of the author and are not endorsed by committee members or Oklahoma State University.

To know what you know and what you do not know, that is true knowledge. —Confucius

You call yourself students? — David C. Ullrich

It is not that I'm so smart. But I stay with the questions much longer.
— Albert Einstein

The content on this page reflects the views of the author and are not endorsed by committee members or Oklahoma State University.

Name: JONATHAN CHRISTOPHER COOK

Date of Degree: December, 2021

Title of Study: DEVELOPMENT OF COMPUTER PROGRAMS FOR FAST COMPUTATION OF G-FUNCTIONS AND AUTOMATED GROUND HEAT EXCHANGER DESIGN

Major Field: MECHANICAL ENGINEERING

Abstract: Ground-source heat pump systems utilizing vertical borehole ground heat exchangers are energy efficient. The most accurate method for sizing the boreholes is based on simulations which make use of superposition of a unit-step thermal response g-function. Due to long calculation times, a library of g-functions was constructed in the 1990's and is still utilized in ground heat exchanger design tools. Peer-reviewed literature has been silent concerning the expansion of the library or further improvements to the closed source g-function calculation program.

g-Functions can now be computed by utilizing a recently developed open-source program, *pyfunction*. Version 1.1 of *pygfunction* had excessive memory consumption for large borehole fields, motivating development of an alternative program written in C++, named *cpgfunction*. The program, when coupled with an adaptive discretization scheme, provided the ability to compute a new expanded library of g-functions. A pure C++ fork of *cpgfunction*, named *cpgfunctionEP*, has been integrated into the whole building energy simulation program, EnergyPlus™, as a third-party g-function calculation tool. Early versions of *cpgfunction* were faster than *pygfunction* for most cases, though recent developments in *pygfunction* have broken the computation barrier for g-function calculation and reduced memory consumption. g-Functions are computed live time in a newly developed *Ground Heat Exchanger Design Toolbox* (GHEDT), written in Python. As part of the work described in this thesis, many additional developments in *pygfunction* have been made.

A fast monthly hybrid time step simulation is developed. This simulation is enhanced and automated compared to an existing program capable of ground heat exchanger sizing; *Ground Loop Heat Exchanger Professional* (GLHEPRO). The GHEDT sizing results are validated against GLHEPRO. GHEDT provides a novel development of automated selection of a borehole field by placing boreholes within user-defined geometric constraints. The advanced methodology searches borehole fields constructed in domains. When simulated at a constant height over the domain, a discontinuous unimodal excess temperature function is formed. An integer bisection routine enables fast selection of a desired field. The most capable design routine automatically places boreholes within a constrained polygonal available land description excluding a polygonal unavailable drilling zone, where a building or utilities may be located.

TABLE OF CONTENTS

Chapter Page

LIST OF TABLES

Table Page

LIST OF FIGURES

Figure Page

Figure Page

Figure Page

Figure Page

Figure Page

Figure Page

ABBREVIATIONS

ASHP	Air source heat pump
GHE	Ground Heat Exchanger
GHEDT	Ground Heat Exchanger Design Tool
GLHEPRO	Ground Loop Heat Exchanger Professional
GSHP	Ground Source Heat Pump
EED	Earth Energy Designer
SBM	Superposition Borehole Model
UBWT	Uniform borehole wall temperature
UHF	Uniform heat flux
UIFT	Uniform inlet fluid temperature
WSHP	Water source heat pump

NOMENCLATURE

VARIABLES	UNITS	DESCRIPTION
B	m	Borehole spacing
c_p	J/(kg K)	Specific heat
D	m	Burial depth of borehole
D_b	m	Borehole diameter
H	m	Height of borehole
r_b	m	Borehole radius
R_b	m K/W	Local borehole thermal resistance
R_b^*	m K/W	Effective borehole thermal resistance
$\Delta \bar{T}_b$	K	Average change in borehole wall temperature
$\dot{Q}$	W	Rate of heat rejection/extraction
$\dot{Q}'$	W/m	Rate of heat rejection/extraction per unit length

GREEK SYMBOLS	UNITS	DESCRIPTION
α	m^2/s	Thermal Diffusivity

MATERIAL SUBSCRIPTS	
f	Fluid
g	Grout
s	Soil

CHAPTER I

Introduction

1.1 Background

Energy engineering is the practice of improving existing systems to more efficiently consume energy, and in some cases improve living conditions. Commercial and residential buildings consume nearly 50% of primary energy consumed in the United States (`https://www.eia.gov/consumption/`), and are a common place for energy engineers to replace existing equipment with more efficient devices. To enable energy engineering for design of new buildings or retrofits, significant research has been put into building energy analysis and simulation programs. A substantial portion of energy consumed by buildings falls into the category of heating ventilation and air-conditioning (HVAC), and there is room to improve the design of devices that transfer heat in and out of buildings so that energy can be more efficiently consumed.

A heat pump makes efficient use of electricity to transfer heat in the opposite direction of natural flow[1]. Reversible heat pumps contain internal valves that allow heating or cooling to be provided, utilizing an external heat source or heat sink. The source/sink medium can be air, water or ground (Ingersoll et al., 1948). Air source heat pumps used for heating can become ineffective due to frost formation on the heat exchanger that inhibits heat transfer with the air. However, a water source heat pump (WSHP) interfacing with an adequate volume of water, such as a large pond, lake, or river can operate in cold weather, and can reduce energy consumption year

[1] The second law of thermodynamics states that the natural path of heat transfer is always from hot to cold

round due to higher thermal properties present in the fluid and less fluctuation[2] in temperature than the outdoor air. Large bodies of water or rivers are not available at every building site. However, the ground is an infinite media with a large thermal mass at a consistent favorable temperature below a depth of 5 m (16.4 ft) (Florides and Kalogirou, 2007).

A WSHP can be coupled to a ground heat exchanger (GHE), or a network of pipes buried in the ground, to form a ground source heat pump (GSHP). A heat carrier fluid is pumped throughout the GHE to transfer heat to or from the ground via convection heat transfer. In the summer months while the heat pump is in cooling mode, heat is rejected to the ground, while in the winter months in heating mode heat is extracted from the ground. GHEs can be open or closed systems. An example of an open system is a extraction/injection well, where water is extracted from a well and then injected back into the ground near the well. Closed systems include horizontal, slinky or vertical networks (Florides and Kalogirou, 2007).

In North America, the most common GHE consist of vertically drilled boreholes with single U-tube pipes connected in parallel flow. Double or multiple U-tubes can be used but are less common. Concentric or co-axial heat exchangers are sometimes used for deep boreholes. U-tubes are typically made of high density polyethylene (HDPE). The installation process includes drilling a vertical borehole, feeding the U-tube(s) into the hole and back filling with grout. The tubes can be weighted to help with the downward feed, and a tube through which grout is pumped is attached to the U-tube, but retracted as the borehole is backfilled. The purpose of the grout is to protect the groundwater from surface infiltration and provide good thermal contact between the tubes and the borehole wall. The circulating heat carrier fluid in a vertical borehole GHE is commonly water or an anti-freeze/water solution.

[2] By Carnot's equations for maximum theoretical heat pump efficiency, it can be determined that the ideal heat pump performance can be achieved when the external body's temperature approaches the indoor temperature.

This thesis focuses on the recent developments of computer programs that enable the design of improved vertical borehole GHE systems to increase energy efficiency of HVAC systems in buildings. Throughout the remainder of this manuscript, the terms vertical borehole GHE and GHE are interchanged.

1.2 Ground Heat Exchanger Potential

GSHPs have a high installation cost when compared to ASHPs. The high initial installation cost of vertical borehole GHEs is typically viewed as the primary hurdle behind widespread adoption (Hughes, 2008). The costs associated with the installation of a GSHP include one or multiple water source heat pump(s), circulation pump, grout, high density polyethylene pipe, drilling costs[3], and labor for piping and heat pump installation (Fyffe et al., 2011).

GSHP systems save energy and in return money and resources each year, where the average payback time period was from 8.6 to 12 years in 2006 (Liu, 2010). Installation of a large-scale ground source heat pump system during construction of a new neighborhood could reduce payback time due to economies of scale (Fyffe et al., 2011). A Residential Energy Consumption Survey in 2015 (Figure 1.1) found that space heating, water heating and air conditioning account for more than 70% of energy consumption in households. A GSHP could reduce the water heating energy consumed in the summer months by rejecting heat to indoor water storage as well as the ground. Liu (2010) conducted a national GSHP retrofit study where he found that replacement of space-heating, water-heating and air-conditioning with GSHPs could reduce energy consumption associated with HVAC and water heating by 45.1%, a savings of 4.2 quadrillion BTUs at the time.

The data plotted in Figure 1.1 make it clear that there is a great dependence on petroleum and gas for space and water heating in homes. Transition from gas heating

[3] The drilling costs per foot for large and small systems is between \$5-\$7 and \$8-\$10, respectively (Fyffe et al., 2011).

to efficient heating with electricity can make a home or building safer and cleaner. If electricity is produced by gas, the primary energy consumption of the GSHP system will only be lower than the gas furnace if the GSHP system performance factor multiplied by the power plant efficiency is higher than the gas furnace efficiency. GSHP systems that are well-designed will usually meet this criterion, but there are also poor GSHP systems that won't. Therefore, GSHP technology provides the potential to reduce total primary energy consumption in residential homes and relinquish the dependence on gas product combustion for heating purposes.

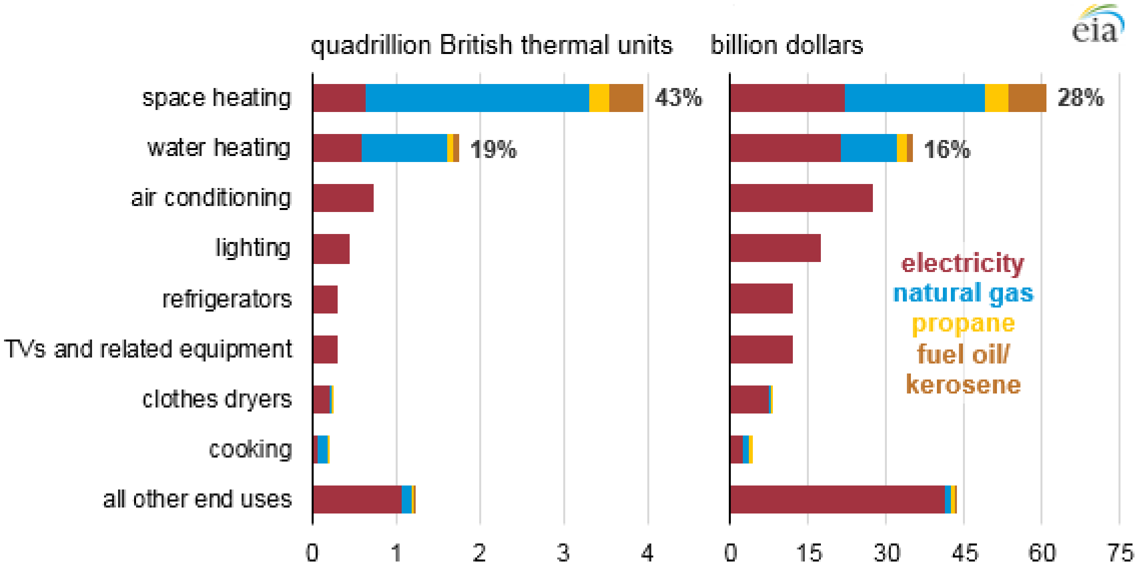


Figure 1.1: US household energy consumption by end-use in quadrillion British thermal units (BTUs) and billion dollars in 2015 (https://www.eia.gov/todayinenergy/detail.php?id=37433).

Design and installation of GSHPs during initial construction and retrofits are not limited to residential housing. To enable energy efficiency research for commercial buildings, a database of files that describe fifteen commercial building types and one multifamily residential building for pre-1980, post 1980 and new construction have been created (Deru et al., 2011). There are weather data files for 1042 locations in the USA. The files are formatted to be input into a whole building energy simulation program, EnergyPlus. These building reference models represent more than 60% of commercial buildings. They do not represent any building in particular, but can be used as reference for design and modified to more closely resemble a specific building. A building reference model and a weather data file can be simulated to create hourly building loads for the year. Hourly building heating and cooling loads are a necessary input for the design of GHE's. Other inputs for GHE design will be discussed in Section 1.3.

1.3 Borehole Thermal Interaction

The circulating fluid exits the borehole field and interfaces with the heat pump. The heat pump moves heat out of the building in the summer and moves heat into the building in the winter. The heat pump extracts or rejects heat to the circulating fluid, which causes the fluid entering the GHE to be different than the exiting temperature. That temperature difference causes convection heat transfer to occur inside of the GHE piping. Conduction heat transfer then ensues inside of the borehole through the grout. There are three heat flow paths inside of a borehole heat exchanger with two pipes: heat transfer between the fluids in opposite flow, heat transfer from the pipe with entering fluid to the borehole wall and heat transfer from the pipe with exiting fluid to the borehole wall (Hellström, 1991). The resistance network can be drawn as a triangle with two vertices located at each pipe, and one at the borehole wall. A top view of a single U-tube and its associated Δ-circuit is shown in Figure 1.2. The internal borehole thermal resistance network approximately reaches steady state at the breaking time, $t_b = 5r_b^2/\alpha_g$ (Eskilson and Claesson, 1988).

The heat is then conducted through the ground (commonly referred to as soil to reduce confusion in subscripts between grout). In a field of boreholes, thermal interaction can play a significant role in the performance of the GHE. A borefield too small and of too tightly packed boreholes could allow the circulating fluid to overheat or freeze. A borefield too large would be unnecessarily expensive. Thus,

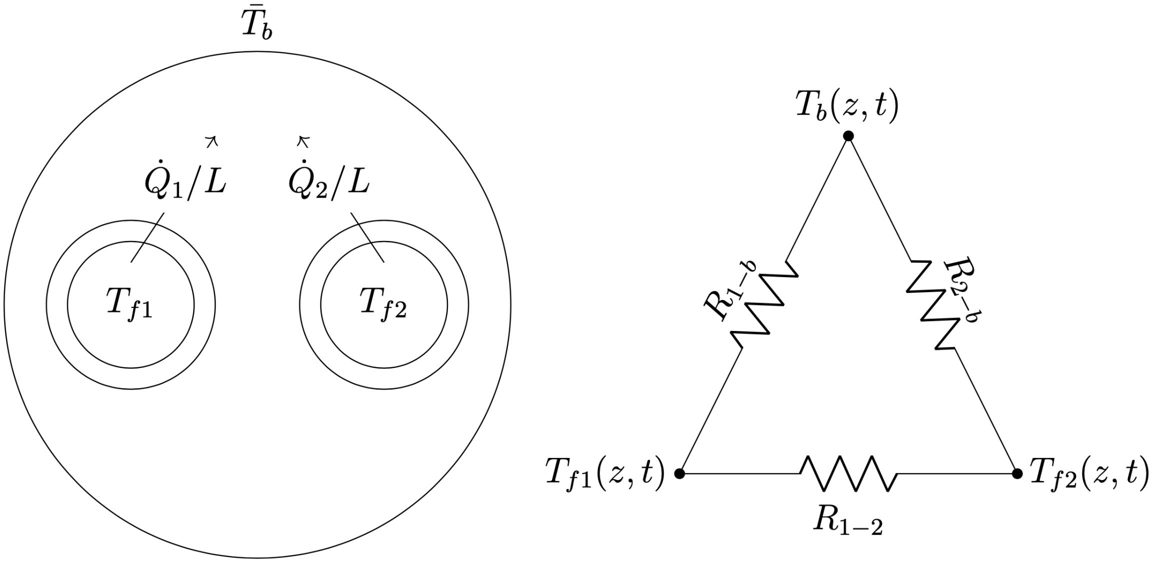


Figure 1.2: Top view of a single U-tube borehole heat exchanger with its corresponding delta circuit.

there is a need for design tools that can simulate the circulating fluid temperature, and account for the short-term internal borehole heat transfer, and the long-term thermal interaction of boreholes. More details about existing design tools are given in Section 1.4.

Eskilson (1987) developed techniques and a 3600 line Fortran program (Eskilson, 1986), named *Superposition Borehole Model* (SBM), to quantify long-term thermal buildup in a ground heat exchanger. The theory was based on a continuous point source (Ingersoll et al., 1948) integrated over a finite length, located in a semi-infinite homogeneous medium. The point source integrated over a finite length has been referred to as a finite line source. Eskilson's solution was derived from Fourier's linear partial differential equation describing conduction of heat in solids. The solution could determine a temperature at any radial distance from the finite line source, at any depth in the medium and any time. The temperature rise at a single location in the medium could be found by summation, or superposition, of multiple finite line sources. Eskilson tabulated the average change in borehole wall temperature due to a constant rate of heat rejection. The thermal response functions were made dimensionless by careful consideration of the rate of heat rejection rate and thermal conductivity of the soil. Eskilson's thermal response g-functions quantified the dimensionless average change in borehole wall temperature, and have been used in design tools by superimposing a time varying heat extraction/rejection rate (building heating and cooling loads) (Cimmino, 2019).

$$\Delta\bar{T}_b = \frac{1}{2\pi k_s}\int_0^t \frac{d\dot{Q}_b'(\tau)}{dt} g(t-\tau)d\tau \tag{1.1}$$

Eskilson's g-functions are applicable to the thermal interaction in the medium between the outer borehole walls and have been referred to as the long-time step g-function. Methods for quantifying the short-term thermal response inside of the borehole were developed by Yavuzturk and Spitler (1999). Yavuzturk and Spitler (1999) developed a short-term dimensionless response function that utilized a two-dimensional transient finite volume numerical model for a borehole containing a single U-tube. The short-term response model was used to extend the g-function to shorter time steps. The computation time of these short-time step g-functions was quick enough that they were not tabulated for interpolation. Xu and Spitler (2006) simplified the numerical grid used in the calculation of the short-time step to be one dimensional by considering the single U-tube borehole heat exchanger as a single pipe. The equivalent single pipe was determined by conserving the volume of the fluid, pipe and the borehole thermal resistance. Short and long-term g-functions are shown in Figure 1.3. More details on the short-term g-functions can be found in (Young, 2004; Claesson and Javed, 2011).

Further details for calculation of the long-term g-function are described in Chapter II. The short-time step of (Xu and Spitler, 2006) is utilized in the new Ground Heat Exchanger Design Tool and its use is discussed in Chapter III.

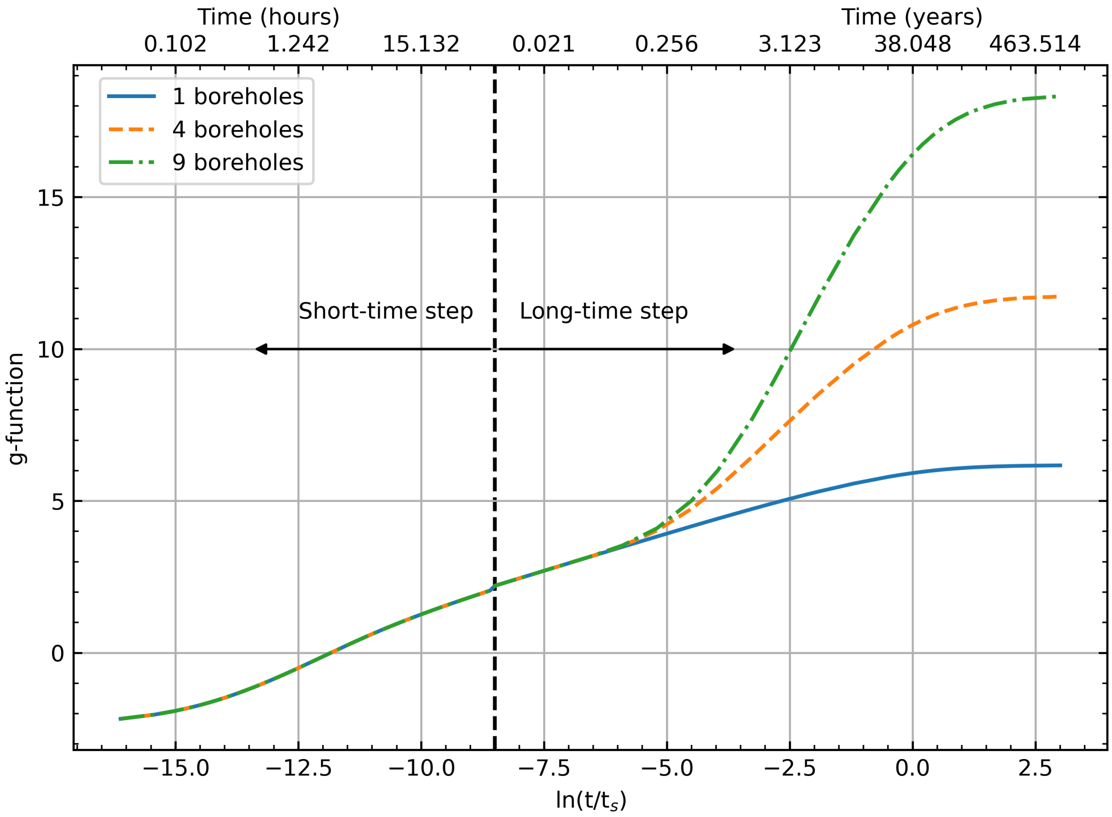


Figure 1.3: Short (Xu and Spitler, 2006) and long-time step g-functions (Cimmino, 2015, 2018b) for a single U-tube with 0.2 L/s per borehole flow rate, 3/4 inch nominal high density polyethylene piping, a borehole radius of 150 mm. The long term g-functions are computed with a burial depth of 2m, a uniform borehole spacing of 5m and a borehole length of 96m.

1.4 Existing Design Tools and Limitations

Eskilson and Claesson (1988) discussed the SBM simulation model for a ground heat exchanger with options to pipe the network in parallel, series or a hybrid of the two. The purpose of the simulation model is to determine the evolution of fluid temperatures throughout time. The heat pump entering (or the heat exchanger exiting) fluid temperatures are commonly used as the design constraints for sizing a system. Typically, a heating dominated building (ground extraction dominated) is constrained by the peak minimum temperature, whereas a cooling dominated building (ground rejection dominated) typically[4] is constrained by the peak maximum temperatures. Taking a cooling-dominated building as an example, for a specific borehole configuration, moving boreholes further apart means reducing the peak heat pump entering fluid temperature. The sizing process for simulation-based design varies the spacing between or height of the boreholes so that the peak temperature does not exceed a user defined constraint.

Simulation based design tools like that of Eskilson and Claesson (1988) are not the only method for sizing GHEs. Cullin (2008) discussed a simplified design approach developed by Kavanaugh (1992); Kavanaugh and Rafferty (1997) that can be worked by hand. The method has been presented in the American Society of Heating, Refrigeration and Air-conditioning Engineers (ASHRAE) HVAC Applications Handbook (ASHRAE, 2011) and has been referred to as the "handbook method". However, Cullin et al. (2015) found the handbook method to have low accuracy. Cullin et al. compared the sizing results procured by the simulation based design of Spitler (2000) coupled with the hybrid time step procedure of Cullin and Spitler (2011) (GLHEPRO) to that of Kavanaugh and Rafferty (1997). Measurement devices installed on four ground heat exchangers provided the heating and cooling loads and the fluid temperatures. The premises of the work was that using measured loads and measured peak heat pump entering fluid temperature limits should result in sizing of ground heat exchangers of the actual depth. The result was that the simulation-based design depth was predicted within 6% of the actual for all four cases, whereas the handbook method ranged from -21% (undersized) to 103% (over-sized).

Hellström and Sanner (1994) developed a ground heat exchanger design program named *Earth Energy Designer* (EED) that utilized Eskilson's SBM. The original Version of EED touted several new features, including a graphical user interface written

[4] Depending on the choice of design minimum temperature and the date of the start of operation, it is also possible that a cooling-dominated system may be constrained by the minimum temperature limit being reached in the first year of operation.

in Pascal, a database of ground thermal properties, estimation of the borehole thermal resistance based on a pipe material database, the ability to control the circulating fluid flow rate and access to a database of g-functions that could be quickly accessed. The program consists of proprietary closed-source software. EED is commercially available and has seen many updates. While there have been no improvements to the SBM, or an expansion of the original database discussed in the peer-reviewed literature, the computer manual for EED Version 3 (BLOCON, 2015) contained more configurations and introduced an optimization procedure that searches over configurations in the database. The optimization procedure can be done on a height or cost basis. The procedure is not well described in the manual, but it appears to perform an exhaustive search over all configurations that will fit within a user-specified rectangular land area, using a user-specified range of borehole spacing and borehole depth with a maximum number of boreholes. A constant expansion factor for the borehole spacing search can be user defined or automatically determined. The determined configurations appear to be limited to contain uniform spacing in the x- and y-directions (all of the configurations in the database contain a uniform borehole spacing). A detailed list of results for various configurations is shown in an example, where the detailed list can be sorted by category (e.g. total length or cost). The optimization of cost is based on inputs for fixed cost of the system and each borehole as well as the cost per length of drilling the boreholes and the trenching between the boreholes. EED version 4 (BLOCON, 2017) introduced a mapping procedure that can map irregular configurations to regular configurations in the library. It is stated that the number of boreholes for the mapped field will typically be near the same as the input field. Therefore, the size of the irregular configurations are limited to what exists in the library computed by SBM. Examples provided appear to show accurate results for the mapping procedure (within ±0.4 °C for both examples), though the specific mathematical methodology remains undisclosed.

The database of g-functions, computed by Hellström via the SBM program, is also utilized for simulation of ground heat exchangers in the *Ground Loop Heat Exchanger Professional* (GLHEPRO) (Spitler, 2000) program. Malayappan and Spitler (2013) implemented the simplified single integral analytical solution of Claesson and Javed (2011) for computing the g-functions in GLHEPRO. However, the conclusion was that the method greatly over approximated the g-functions for larger configurations, so the method is recommended for a maximum of 36 boreholes.

EnergyPlus™ (DOE, 2021) can perform a whole building energy simulation that includes a ground coupled heat pump. Prior to Version 9.6.0, the two methods for g-functions in EnergyPlus were either to compute a g-function utilizing the solution of Claesson and Javed (2011), or to input a computed g-function. Prior to this work, the recent monumental development of an open-source g-function calculation toolbox by Cimmino (2018b), named *pygfunction*, could have been utilized to compute g-functions for input to EnergyPlus. As part of the work described in this thesis, EnergyPlus Version 9.6.0 is improved with the g-function calculation of Cook and Spitler (2021) and an adaptive discretization scheme discussed in Chapter II.

In summary, the calculation of thermal responses between the boreholes in existing design tools rely on low-accuracy manual methods, libraries, g-function calculations with constraints on the number of boreholes, or in the case of EED some undisclosed proprietary mapping procedure. To date, GLHEPRO requires a manual description of the locations of boreholes, rather than a description of the available land area with an optimized field being returned. EED contains the ability to perform an exhaustive search over all (or a subset of) configurations available in the library. The required inputs are a rectangular land area description, borehole spacing range (minimum and maximum), borehole height range and a maximum number of boreholes. All configurations that can be sized within the specified bounds are sized and placed into a table. The table can be sorted in descending or ascending order by the category of primary interest (e.g. total length or cost). EED only utilizes borefields with equal spacing in both directions, and the outer land description can only be described by a rectangle.

This thesis describes a new program with a novel borehole configuration selection methodology. The new program contains improved optimization in many aspects when compared to EED, such as: a live-time g-function calculation that considers the heat distribution in a more realistic manner (via *pygfunction*), ability to automatically determine configurations with unequal x- and y-spacing, the outer land description can be defined by polygonal vertices and inner polygonal no-drilling zones can be defined. Additionally, the authors desire for this work is that it be open-source where researchers can directly build off of the work to improve design methodologies in the future. This desired approach is significantly different from the existing proprietary GHE design programs.

1.5 Objectives

The primary objective of this work is the development of a new *Ground Heat Exchanger Design Tool* (GHEDT) that can can find an optimal configuration and determine the required depth *quickly*[5]. GHEDT is both a standalone program and serves as an automated design tool for quantifying economic benefits for use in an online techno-economic program (Liu et al., 2020). To quantify the energy savings, GHEDT generates input data for EnergyPlus. The capabilities of GHEDT include:

- Compute the effective borehole thermal resistance for any borehole containing a single U-tube, multiple U-tubes or a coaxial tube.
- Compute an equivalent single U-tube based on any multiple U-tube or coaxial GHE so that EnergyPlus can simulate the system (EnergyPlus currently can

[5] The original goal of this work was to automatically determine a configuration and size it within 5 minutes, though all but one routine completes in less than 30 seconds, and the longer routine completes in around 1.5 minutes.

only accept single U-tube piping arrangements).

- Perform an accurate and fast simulation of ground heat exchangers using g-functions computed on the fly. The long-time step g-functions are computed by the method of Prieto and Cimmino (2021), and the short-time step g-functions are computed by the method of Xu and Spitler (2006).

- Develop a final sizing methodology to determine the required borehole depth for a specific configuration.

- Automatically place boreholes within a user-defined land area. This is coupled with a domain search to find an optimal configuration for a target drilling depth. The more advanced design routines allow unequal spacing in each direction, the ability to define an outer polygonal available land area and inner polygonal unavailable areas.

1.6 Outline

The objective of this work is to develop a new ground heat exchanger design program that can perform the simulation and sizing tasks like existing design tools (GLHE-PRO, EED) in addition to novel optimized placement and sizing of boreholes based on an outer polygonal land constraint with the optional ability to define inner no-drilling zones.

- Chapter II Discusses the development of *cpgfunction* and why it was necessary, an adaptive discretization scheme, a new g-function calculation for EnergyPlus (*cpgfunctionEP*), a library of g-functions computed for use in GHE design tools and an interpolation procedure utilized in GHEDT. Recent improvements to the g-function calculation of *pygfunction* and *cpgfunction* are discussed.

- Chapter III presents the simulation and sizing methodology of GHEDT. The ability to compute equivalent single U-tubes based on any multiple U-tube or coaxial tube configuration is discussed. Sizing results of GHEDT are validated against the existing design tool, GLHEPRO.

- Chapter IV discusses recent improvements to pygfunction that allow fast and accurate on the fly g-function calculation. Novel design algorithms, implemented in GHEDT, for automated borehole placement are presented and proven to be useful and robust.

- Chapter V contains conclusions and recommendations for future work.

CHAPTER II

Ground Thermal Response g-Functions

2.1 Introduction

Ground-source heat pump (GSHP) systems are an energy-efficient, low-emission means of heating and cooling buildings. Particularly for larger systems serving commercial and institutional buildings, ground heat exchangers consisting of multiple vertical boreholes are commonly used to provide the heat source and/or heat sink. Ground heat exchangers represent a significant portion of the system first cost, and methods to optimize the design by reducing the amount of drilling required are discussed in Chapter IV.

A vertical borehole ground heat exchanger (GHE) consists of holes drilled into the ground with high density polyethylene piping inserted that circulates a heat exchange fluid to transfer heat between the fluid and the ground. The piping may be configured as a single U-tube, double U-tube, or a coaxial arrangement. In most of North America and much of the rest of the world, the holes are backfilled with grout to prevent infiltration of polluted surface water into the ground and any aquifers present. As a secondary effect, the grout facilitates conduction heat transfer, though it is common for the grout to have a lower thermal conductivity than the surrounding ground. In Scandinavia, GHEs are commonly filled with groundwater, relying on a well top and steel casing into the bedrock to protect the groundwater from surface pollutants. Common heat exchange fluids are alcohols (methyl or ethyl) and glycols (propylene or ethylene) mixed with water.

The long-time scales associated with heat transfer in the ground make estimating the borehole-to-borehole thermal influence a complex process. Accurate prediction of the ground response to heat input is crucial for determining a proper size of a GHE. An undersized GHE may fail by overheating or freezing the circulating fluid, which could cause increased operating costs through poor heat pump performance, heat pump failure or circulating pump failure. An oversized GHE could be too large of an initial investment to be installed. Given the significant cost of a GHE, a "right-sized" GHE is highly preferable.

Prof. Johan Claesson and his PhD student Per Eskilson at Lund University developed methods (Eskilson, 1986, 1987; Claesson and Eskilson, 1988; Eskilson and Claesson, 1988) to quantify the long-term thermal interaction of boreholes and the associated ground heat transfer using a dimensionless borehole wall temperature response function known as a g-function. g-Functions are used within a simulation to predict the evolution of temperatures returning to the heat pump over time. The results of a multi-year simulation can provide insight as to whether the drilling depth should be varied, more boreholes should be added to the system, or the boreholes should consist of a different configuration[1]. Within GHE sizing programs such as *Ground Loop Heat Exchanger Professional* (GLHEPRO) (Marshall and Spitler, 1994; Manickam et al., 1996; Spitler, 2000; Cullin, 2008; Grundmann, 2016) and *Earth Energy Designer* (Hellström and Sanner, 1994) g-functions are used along with information about ground thermal properties, heat pump performance, borehole configuration and the building heating and cooling loads to predict temperature of the fluid returning from the GHE. Both programs incorporate libraries (databases) of g-functions for standard configurations, e.g. rectangles, L-shapes, U-shapes, etc. originally computed by Hellström using a program written by Eskilson (1986). To the best of the authors knowledge, this program for calculating g-functions is not open-source and it is not publicly available.

[1] The term "configuration" is used in this thesis to mean the horizontal arrangement (x,y coordinates) of boreholes as seen in a plan view.

The limited set of borehole geometrical configurations available in the library has served as the basis for many installed GHEs. Nevertheless, a more comprehensive library would be useful in supporting design. At the onset of this work, a tool that could calculate g-functions for large borehole fields with acceptable speed and memory requirements was also highly desired, both for use in designs and research of faster methods for calculating g-functions. This chapter discusses the theory of the g-function, the development of a C++ g-function calculation tool, *cpgfunction* (Cook and Spitler, 2021) used to develop a new library with over 34k configurations containing up to 1024 boreholes (Spitler et al., 2021) and an adaptive discretization scheme utilized to further decrease computation time. *Cpgfunction* has been forked and made into a pure C++ project and been integrated in to EnergyPlus. The dimensioning rules of Eskilson are utilized to interpolate g-functions.

In many respects, the need that drove development of *cpgfunction* has now been met by the equivalent borehole method (EBM) of Prieto and Cimmino (2021). However, *cpgfunction* has proven to be useful for library development, and for custom calculation of g-functions within EnergyPlus. It is also likely that it may be useful in other design tools, in the form of a precompiled callable executable.

2.2 Background

2.2.1 Theory of the g-Function Calculation

This section summarizes aspects of the g-function calculation theory. First, Fourier's 2nd order partial differential equation governs all heat conduction problems (Ingersoll et al., 1948). Although other phenomena (e.g. saturated and unsaturated moisture transport) may be present, g-functions are calculated assuming pure conduction heat transfer. Because the partial differential equation is linear, the responses of individual segments may be superimposed in space and time. The domain is assumed to be isothermal[2] for purposes of g-function calculation. The isothermal ground surface boundary is modeled with a method of images – mirror image segments above the surface of the ground are included in the calculation.

To understand the theory of the g-function, it is important first to consider a heat point source (Ingersoll et al., 1948). A point source is an infinitesimal point that begins emitting heat at a rate of $\dot{Q}$ at time $t = t_1$ in a medium that initially is of temperature $T = 0$ and contains no heat transfer. A radial flow of heat will begin, and the temperature at any radial distance at any point in time can be found. The solution for temperature at time t, radius r in a medium with conductivity k and thermal diffusivity α based on a constant heat rejection rate $\dot{Q}$ was given by Ingersoll et al. (1948) as

$$T(r,t) = \frac{\dot{Q}}{4\pi k r}\frac{2}{\sqrt{\pi}}\int_{\frac{r}{2\sqrt{\alpha t}}}^{\infty} e^{-\beta^2} d\beta \tag{2.1}$$

where $\beta = r/2\sqrt{\alpha(t-\tau)}$. A solution based on a time varying heat extraction rate was also presented.

$$T(r,t) = \frac{1}{2(\pi^{3/2})\alpha r}\int_{\frac{r}{2\sqrt{\alpha t}}}^{\infty} \dot{Q}(t - \frac{r^2}{4\alpha\beta^2})e^{-\beta^2} d\beta \tag{2.2}$$

Eskilson (1987) integrated a point source from burial depth, D to the height of the borehole H ($D < z < D + H$). Eskilson referred to the integration as a continuous point source integrated along the length of the borehole, though more recently the solution has been referred to as the finite line source. The solution included a mirror source above the ground surface with negative temperature response to impose an isothermal boundary at the ground surface, located at $r = 0$ and $-D-H < z < -D$. Eskilson's finite line source was a solution for the rise in temperature at any radial

[2] g-Functions are commonly used assuming the undisturbed ground is isothermal. This assumption is usually justified by the near constant average temperature of the ground over the depth of the borehole. It's possible to superimpose an undisturbed ground temperature that changes with time and depth. This might be useful if modeling very short boreholes.

distance r, any depth z and after a period t.

$$T(r,z,t) = -\frac{\dot{Q}}{4\pi kH}\int_{D}^{D+H}\left(\frac{1}{r_+}erfc\left(\frac{r_+}{\sqrt{4\alpha t}}\right) - \frac{1}{r_-}erfc\left(\frac{r_-}{\sqrt{4\alpha t}}\right)\right) \tag{2.3}$$

The values of r_+, r_- and r are given as

$$r_+ = \sqrt{r^2 + (z-s)^2)} \qquad r_- = \sqrt{r^2 + (z+s)^2} \qquad r = \sqrt{x^2 + y^2} \tag{2.4}$$

and $erfc$ is the error function. Eskilson (1986) computed a two-dimensional numerical grid of temperature responses in his computer program, called the Superposition Borehole Model (SBM). The first position of the horizontal mesh array, located at the borehole radius, was allocated for the borehole boundary condition, which considered the borehole wall temperature to be uniform, or equal along the length. Further discussion of the numerical mesh is given in Chapter IV. The g-function then, was the average rise in temperature at a borehole wall due to a heat rate of $\dot{Q}/(2\pi kH) = 1°C$, computed by utilizing a finite difference method to sum up all the contributions of thermally interacting boreholes.

Claesson and Javed (2011) derived a single integral analytical expression for the average temperature rise along a borehole length. The solution is derived with a constant heat rejection rate of $\dot{Q}/H$ along the borehole and includes a mirror response above the ground surface to impose an isothermal ground surface. The average temperature rise at the borehole wall is given as $\bar{T}_{ls}$,

$$\bar{T}_{ls}(r,t) = \frac{\dot{Q}}{4\pi kH}\int_{\frac{1}{\sqrt{4\alpha t}}}^{\infty} e^{-r^2s^2}\frac{I_{ls}(Hs, Ds}{Hs^2}ds \tag{2.5}$$

where,

$$I_{ls}(h,d) = 2ierf(h) + 2ierf(h+2d) - ierf(2h+2d) - ierf(2d) \tag{2.6}$$

$$ierf(X) = \int_0^X erf(x')dx' = Xerf(X) - \frac{1}{\sqrt{\pi}}(1 - exp(-X^2)) \tag{2.7}$$

therefore, the long-term response for single borehole could be computed at the borehole radius, r_b.

$$T_{b,1} = \bar{T}_{ls}(r_b, D, H, t) \tag{2.8}$$

The temperature response at borehole i in a field of multiple boreholes was given as

$$\bar{T}_{b,i}(t) = \sum_{j=1}^{N} \bar{T}_{ls}(d_{ij}, t) \tag{2.9}$$

$$d_{ij} = \begin{cases} r_b & i = j \\ \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} & i \neq j \end{cases} \tag{2.10}$$

The average temperature rise at the borehole wall could then be determined.

$$\bar{T}_b = \frac{1}{N} \sum_{i=1}^{N} \sum_{j=1}^{N} \bar{T}_{ls}(d_{i,j}, t) \tag{2.11}$$

Therefore, the unit step response g-function could be determined.

$$g(t) = \frac{1}{N} \sum_{i=1}^{N} \sum_{j=1}^{N} \frac{1}{2} \int_{\frac{1}{\sqrt{4\alpha t}}}^{\infty} e^{-di,j^2 s^2} \frac{I_{ls}(Hs, Ds)}{Hs^2} ds \tag{2.12}$$

Claesson and Javed (2011) concluded that the method was in good agreement with Eskilson's g-functions for short-time scales, though it was stated that as the number of boreholes grew the differences also grew. The largest field compared in their paper was a square layout with 9 boreholes.

A g-function calculation program utilizing the analytical expression would necessitate far less finite line source evaluations when compared to the numerical grid of SBM. With that understood, the method was of interest and was implemented into EnergyPlus (DOE, 2021) and GLHEPRO. However, the solution of Claesson and

Javed (2011) is based on a constant heat flux applied to the borehole wall. This boundary condition solution is not the same as Eskilson's, who considered an average borehole wall temperature equal for all boreholes. Malayappan and Spitler (2013) concluded that the method should only be used for less than around 35 boreholes.

Cimmino and Bernier (2014) set out to develop a computer program that utilized the analytical finite line source solution and could approximate Eskilson's g-functions with accuracy. The finite line source response of Claesson and Javed (2011) was enhanced by deriving a solution that allowed the two thermally interacting line sources to be of different height and depth. Therefore, the boreholes could now be discretized into a number of segments, n_q along the length. The change in temperature at segment v along borehole j due to a rate of heat rejection over the length $\dot{Q}'$ of segment u along borehole i was given.

$$\Delta T_{i\to j,u\to v}(t) = \frac{\dot{Q}'_{i,u}}{2\pi k} h_{i\to j,u\to v}(t) \tag{2.13}$$

$$h_{i\to j,u\to v}(t) = h_{FLS}(t, d_{ij}, H_{i,u}, D_{i,u}, H_{j,v}, D_{j,v}) \tag{2.14}$$

$$\begin{gathered}
h_{i\to j,u\to v}\left(t\right) = \tfrac{1}{2H_{j,v}} \int_{\frac{1}{\sqrt{4\alpha t}}}^{\infty} \tfrac{1}{s^2} \exp\left(-d_{ij}^2 s^2\right) [ierf\left(\left(D_{j,v} - D_{i,u} + H_{j,v}\right)s\right) \\
-ierf\left(\left(D_{j,v} - D_{i,u}\right)s\right) \\
+ierf\left(\left(D_{j,v} - D_{i,u} - H_{i,u}\right)s\right) - ierf\left(\left(D_{j,v} - D_{i,u} + H_{j,v} - H_{i,u}\right)s\right) \\
+ierf\left(\left(D_{j,v} + D_{i,u} + H_{j,v}\right)s\right) - ierf\left(\left(D_{j,v} + D_{i,u}\right)s\right) \\
+ierf\left(\left(D_{j,v} + D_{i,u} + H_{i,u}\right)s\right) - ierf\left(\left(D_{j,v} + D_{i,u} + H_{j,v} + H_{i,u}\right)s\right)]
\end{gathered} \tag{2.15}$$

For the borehole wall temperature to remain equal in the field, it was found that the heat extraction rates had to vary from segment to segment. The change in temperature $\Delta T_{i\to j,u\to v}$ based on a time varying heat rejection rate $\dot{Q}'_{i,u}(t)$ was shown to be possible via temporal superposition of the finite line source solution.

$$\Delta T_{i\to j,u\to v}\left(t_k\right) = \sum_{p=1}^{k} \left[\frac{\dot{Q}'_{i,u}\left(t_k\right) - \dot{Q}'_{i,u}\left(t_{k-1}\right)}{2\pi k} h_{i\to j,u\to v}(t_k - t_{p-1})\right] \tag{2.16}$$

Where $\dot{Q}'_{i,u}(t_0) = 0$. The change in temperature of $T_{b,j,u}$ based on a time varying heat rejection rate can be found by summation of the changes in temperature from all borehole segments, at all previous points in time.

$$\Delta T_{b,j,u} = \sum_{p=1}^{k}\sum_{i=1}^{n_b}\sum_{j=1}^{n_q} \frac{\dot{Q}'_{i,u}(t_k) - \dot{Q}'_{i,u}(t_{k-1})}{2\pi k} h_{i\rightarrow j,u\rightarrow v}(t_k - t_{p-1}) \tag{2.17}$$

The total rate of heat extraction applied to the boreholes, $\bar{Q}'$ is made constant throughout time.

$$\frac{\sum_{i=1}^{n_b}\sum_{j=1}^{n_q} H_{i,u}\left(\dot{Q}'_{i,u}(t_k) - \dot{Q}'_{i,u}(t_{k-1})\right)}{\sum_{i=1}^{n_b}\sum_{j=1}^{n_q} H_{i,u}} = \bar{Q}'\delta(t_k - t_1) = \begin{cases} \bar{Q}' & t = t_1 \\ 0 & t_k \neq t_1 \end{cases} \tag{2.18}$$

The heat rejection applied to the borehole walls can be made dimensionless by considering a nominal heat rejection rate $\tilde{Q} = \dot{Q}'/\bar{Q}'$.

$$\frac{\sum_{i=1}^{n_b}\sum_{u=1}^{n_q} H_{i,u}\left(\tilde{Q}_{i,u}(t_k) - \tilde{Q}_{i,u}(t_k)\right)}{\sum_{i=1}^{n_b}\sum_{u=1}^{n_q} H_{i,u}} = \delta(t_k - t_1) = \begin{cases} 1 & t = t_1 \\ 0 & t_k \neq t_1 \end{cases} \tag{2.19}$$

The change in borehole wall temperature can then be made non-dimensional.

$$\Theta_{b,j,u}(t_k) = \sum_{p=1}^{k}\sum_{i=1}^{n_b}\sum_{u=1}^{n_q} \left(\tilde{Q}_{i,u}(t_k) - \tilde{Q}_{i,u}(t_{k-1})\right) h_{i\rightarrow j}(t_k - t_{p-1}) \tag{2.20}$$

Where $\Theta_{b,j}(t_k) = \frac{\Delta T_{b,j}(2\pi k)}{\bar{Q}'}$. The case of $\Theta_b(t_k) = \Theta_{b,j}(t_k)$ is uniform borehole wall temperature for all boreholes.

Cimmino and Bernier (2014) did the following to compute g-functions that well approximated the original g-functions of Eskilson:

- Applied a constant total heat rejection rate to the boreholes.
- Discretized the finite line source segments into 12 vertically stacked segments

for each borehole.

- Adjusted the heat distributed to each segment to vary in time to ensure the borehole wall temperatures were uniform.

Cimmino (2018a) went on to show how to build a system of equations to solve the g-functions and introduced numerical methods to decrease calculation time. The numerical methods consisted of:

- Similarity identification – Algorithms were presented for determining groups of borehole segments that contain the same segment response. These segments were all located at the same horizontal and radial distances from one another.
- Load history reconstruction – For an uneven time step, the heat rejection rate is unknown at times. An accumulated heat extraction rate function was created that interpolates the previous heat rejection rates. The accumulated heat extraction rate function was linearly interpolated to find the rate of heat rejection at the time step.
- Multithreading – The calculation of the segment response factors was performed in parallel.

Cimmino (2018b) published an open-source python package, *pygfunction*, capable of computing g-functions with boundary conditions of uniform heat flux (UHF), uniform borehole wall temperature (UBWT) and uniform inlet fluid temperature (UIFT) for all boreholes (Cimmino, 2015, 2016). More discussion of the g-function background up through the current day is given in Chapter IV.

2.2.2 Performance of *pygfunction* (v1.1.1)

Ground source heat pump systems commonly use vertical borehole ground heat exchangers to serve as a heat sink and source. Since the ground heat exchanger (GHE) represents a significant portion of the system cost, it is important to accurately "right-size" the GHE. This is particularly the case as large systems become more common. Current interest in GSHP technology began in the 1970s with residential applications, but there are many large commercial systems with hundreds of boreholes, and some "district-scale" systems with more than a thousand boreholes. Two large district-scale examples are the Ball State University system with more than 3600 boreholes and the Epic Systems system with more than 6000 boreholes (Florea et al., 2017). For both large systems serving commercial and institutional buildings, g-functions are needed for design and simulation purposes. Calculation of these g-functions, whether done for a specific design, or as part of development of a library, is of interest. Due to available computing resources, the work in this chapter is focused on borehole configurations with a maximum number of boreholes around 1000.

The first task to address in this process was how to compute g-functions, given that for many years the only program that could do so was the SBM. The open-source g-function calculation tool, *pygfunction*, proved to be accurate and was the quickest to get started with (given that the techniques were already programmed and available).

In studying the results of the g-function calculation in *pygfunction*, it was found that the convergence was dependent on the number of segments and the height of the boreholes. Cimmino and Bernier (2014) discussed the idea of convergence based on the number of segments used, but did so at one height. Figure 2.1 shows the error of the g-function curve (over the 27 calculated points in dimensionless time points shown in Appendix B), based on the number of equal segments along the borehole length and shown for various heights. The reference cases here are g-functions computed with 96 equal segments along the borehole. g-Functions were computed using a 12x13 borehole layout. What is noticeable is that a borehole height of 24m is adequately converged (<1%) with 6-8 segments. As the depth increases, the required number of segments to reach a mean percent error of less than 1% increases. The required number of segments for adequate convergence at 48m is around 13 segments, and then for 384m the number of segments required for less than 1 mean percent error over the g-function would be more than 50 segments along the borehole length.

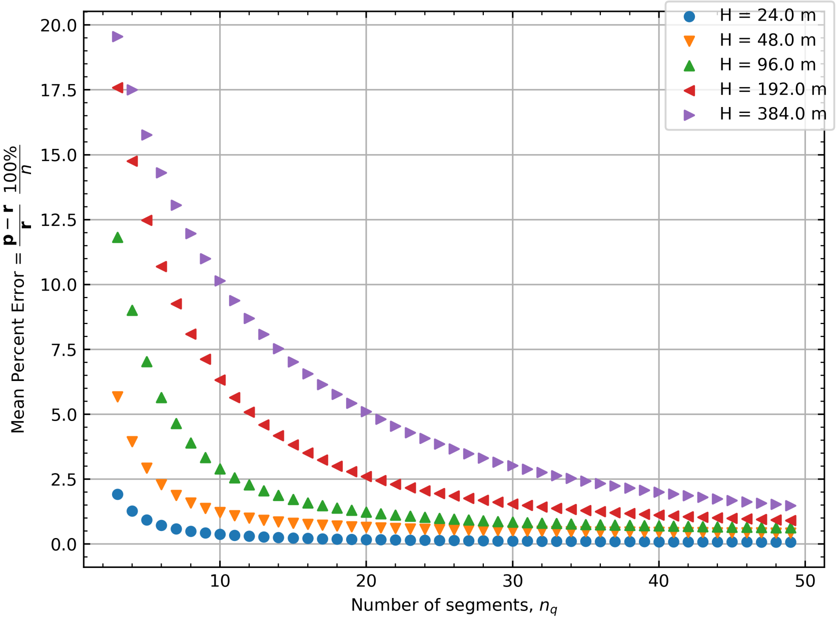


Figure 2.1: The mean percent error of a g-function curve based on an equal number of segments used along the borehole length, plotted at various heights.

Pygfunction version 1.1.1 was available at the start of this work. Increasing the number of segments along the borehole resulted in memory allocation failures on typical personal computers (PCs). The culprit was found to be that the memory consumed became so large that the program could not execute. *Pygfunction* is a very powerful tool for calculating g-functions, but for larger borehole fields with hundreds of boreholes and thousands of finite line sources, the required memory and computational time can be quite high, exceeding several hours and requiring hundreds of Gigabytes (GB) of random access memory (RAM). Figure 2.2 shows the computational time and memory requirements for calculation of g-functions for square borehole fields up to 1024 boreholes, with 24 finite line sources per borehole. The computational time and memory are plotted against the total number of finite line source segments, $n_{bh} \times n_q$, for each ground heat exchanger. (I.e. n_{bh} is the number of boreholes and is multiplied by the number of finite line source segments per borehole n_q.) Cimmino and Bernier (2014) approximated Eskilson's g-functions with 12 finite line sources per borehole. As shown in Figure 2.1, the g-functions have some sensitivity to the discretization and for larger borefields and deeper boreholes, more than 12 finite line sources are needed.

The performance of *pygfunction* version 1.1.1, shown in Figure 2.2, was captured on Oklahoma State University's High Performance Computing Cluster OSU HPCC. All compute nodes on the cluster use dual Intel Skylake 6130 CPUs (OSUHPCC, 2021). There are 164 "batch" nodes with 96 GB RAM, 12 "bigmem" nodes with 768 GB RAM and one "supermem" node with 1.5 terabytes (TB) RAM. For larger fields with several hundred boreholes or more, the memory required can easily exceed what's available on most desktop computers (typical desktop computers currently have 8-16GB of RAM), and even exceed the 96 GB available on the most commonly available OSUHPCC "batch" nodes.

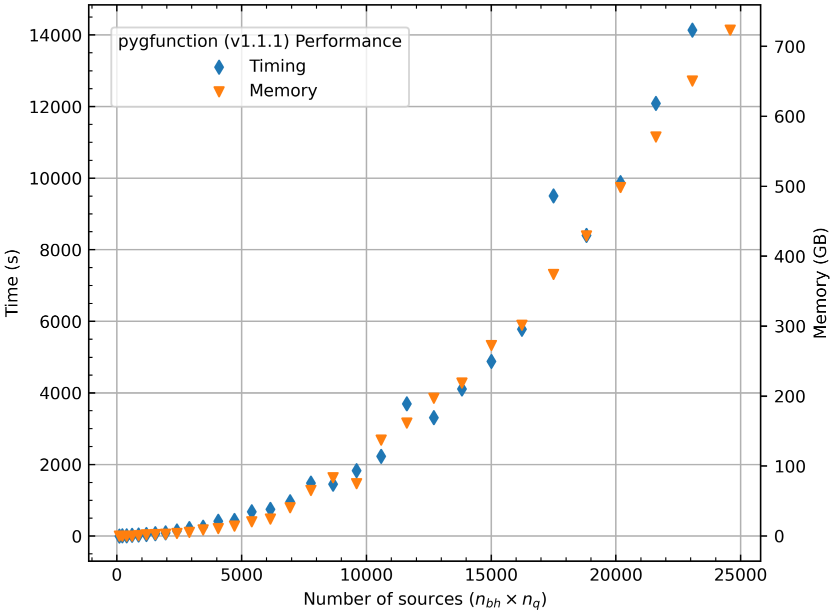


Figure 2.2: The timing and memory performance of *pygfunction* (version 1.1.1) based on the total number of sources used in the calculation.

Therefore, the amount of memory required can be very important. With a new expanded library of g-functions in mind that contain configurations up to 1000 boreholes, it would be beneficial for the consumption of memory to be below 96 GB so that the "batch" nodes could be utilized on the cluster. A new g-function calculation program written in C++, named *cpgfunction*, was developed with an eye towards reducing memory consumption. The result is a nearly 8-fold reduction in memory.

2.3 Methodology

2.3.1 Development of *cpgfunction*

This section gives an overview of the *cpgfunction* methodology, much of which is derived directly from *pygfunction*. In order to benefit from the increased speed of a compiled language, *cpgfunction* was written in C++. Both programs require calculation of segment-to-segment response functions, using Cimmino and Bernier's generalization (2014) of Claesson and Javed's (2011) single integral analytical finite line source formulation. Letting $h_{ij}\left(t_k\right)$ represent the dimensionless mean temperature rise of the jth segment due to a unit heat flux input at the ith segment, after time t_k, a matrix is $H\left[i,j,t_k\right]$ formed in *pygfunction* and later used to calculate the g-function. This matrix is responsible for much of *pygfunction*'s memory consumption.

The H matrix has a form of symmetry across the diagonal; if L_i, L_j are the lengths of the ith and jth segments, respectively:

$$L_i \cdot H\left[i,j,t_k\right] = L_j \cdot H\left[j,i,t_k\right] \tag{2.21}$$

Given the need to reduce the memory consumption, only the lower triangular half of the matrix is stored; when needed, values from the upper triangular half are recalculated using a reordered form of Equation 2.21. This process has been referred to as matrix "packing". The lower half of the matrix is stored in a vector using an index transformation:

$$I_{dx}\left(i,j\right) = \frac{i*\left(2*n_q*n_{bh}-i-1\right)}{2}+j \tag{2.22}$$

Where $n_q * n_{bh}$ is the total number of finite line sources, and I_{dx} is the index for the vector.

In order to calculate the UBWT and UIFT boundary conditions, *pygfunction* (v1.1.1) allocates memory and computes variants of the segment response matrix H, which are used in the spatial and temporal superposition processes. *Cpgfunction* does not allocate additional space for these matrices, but accesses the packed matrix H, which is a vector. The resulting memory savings of using a single triangular packed matrix and eliminating the two additional matrices gives approximately a six-fold reduction in memory required. Beyond this reduction, the containers (specifically the vector container available in the standard C++ library) used in *cpgfunction* are minimized. C++ containers handle the memory allocation and deallocation that would have to be explicitly described in C. Each container is only allocated once, and the locations in those arrays overwritten with new values rather than replaced. The net impacts, as shown later, are, in practice, reductions between five-fold and nine-fold.

Having spoken of the general symmetry of the H matrix, it is important to note that both programs take advantages of other symmetries for reducing the number of calculations of segment-to-segment response functions. In large rectangular-grid configurations, there are many, many identical pairs of segments. (By "identical", it is meant that they have the same horizontal distance and same vertical distance, within some tolerance.) As discussed by Cimmino (2018a), computing a single segment response for boreholes in similarity groups can greatly speed the g-function computation where there are large numbers of identical pairs.

For smaller problems, the speed improvement of the similarity identification process in *cpgfunction* results in faster performance. Though, as the problem becomes large enough, the lack of vectorization in the temporal and spatial superposition process in *cpgfunction* (version 1.0) results in *pygfunction* becoming slightly faster for the largest cases benchmarked. *Cpgfunction* makes extensive use of multithreading to perform calculations in parallel.

Regarding the boundary conditions, it should be noted that *cpgfunction* currently only implements calculation of the UBWT boundary conditions; calculation of UIFT boundary conditions would be a desirable capability.

2.3.2 Adaptive Discretization Scheme

In order to further reduce the memory and time requirements, the discretization can be adjusted for each case. This section describes the methodology.

2.3.2.1 Sensitivity to Discretization and Boundary Conditions

As Cimmino and Bernier (2014) discussed, the accuracy of the semi-analytical (stacked analytical finite line source) solution was dependent on the number of vertical segments. Additionally, the uniform inlet fluid temperature boundary condition of Cimmino (2015) is the closest model to reality for a borefield piped in parallel. Therefore, the desired g-function to be calculated was a UIFT boundary condition with many segments along the borehole. Due to long calculation times, a library of typical UIFT g-functions were computed utilizing *pygfunction* version 1.1.1 (this was done prior to development of *cpgfunction*) for use in an automated borefield selection process for square fields ranging from 4 to 1024 boreholes (Liu et al., 2020). However, even with the use of the OSU HPCC, calculation of 5 B/H ratios for each configuration took significant time due to there only being 8 nodes with more than 96 GB of RAM. The RAM consumption of *pygfunction* (version 1.1.1) was a major bottleneck. This bottleneck could be overcome if memory consumption was reduced enough to allow the 160 nodes with up to 96 GB of RAM on OSU HPCC to be utilized.

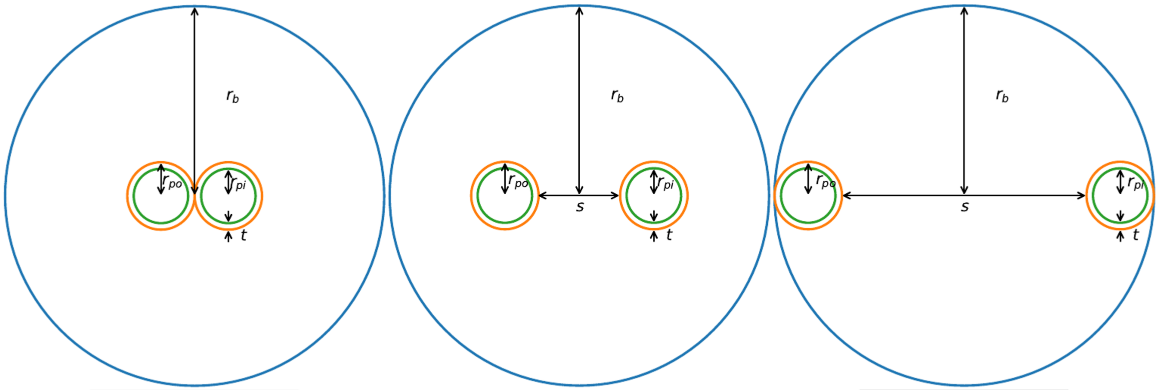
The UBWT boundary condition makes the assumption that the flow rate is high enough along the borehole such that the temperature along the borehole is equal, and that the temperature along all the boreholes are also equal. Hence, a uniform borehole wall temperature. However, the UIFT boundary condition proposed by Cimmino (2015) is closer to reality. The solution is available in *pygfunction* and requires additional input details. The UIFT solution requires u-tube placement and properties. The properties include ground, grout and pipe thermal conductivities, mass flow rate of fluid and fluid thermal properties. The UIFT solution is dependent on borehole resistance. Borehole resistance is based on grout thermal conductivity and u-tube pipe placement. There are three configurations for single U-tubes from the literature worth mentioning, comprised of the A, B, and C geometries originally described by Paul (1996), and shown in Figure 2.3:

- A configuration – the tubes are touching one another. This is the worst case scenario, where the temperature of the water in the downward flowing tube could be mostly transferring heat to/from the upward flowing tube to the heat exchanger rather than the surrounding ground. This leads to a very high borehole resistance.

- B configuration – the tubes are evenly spaced in the borehole. This is considered to be the average or typical result of weighting and dropping U-tubes into the borehole without controlling their position. This results in what is considered typical borehole resistance.

- C configuration – the pipes are touching the outside wall. This is a best case scenario, and can be accomplished with placing borehole spacers in the holes. However, spacers increase the cost of installation, so this low borehole resistance scenario is normally forgone in practice.

The UIFT g-function falls in between the UHF solution of Claesson and Javed



Variable	Value	Units
r_b	75	mm
r_{po}	13.33	mm
r_{pi}	10.77	mm
s	0	mm

Variable	Value	Units
r_b	75	mm
r_{po}	13.33	mm
r_{pi}	10.77	mm
s	32.22	mm

Variable	Value	Units
r_b	75	mm
r_{po}	13.33	mm
r_{pi}	10.77	mm
s	96.66	mm

Figure 2.3: Borehole configuration A (left), B (middle) and C (right) using U-tube made from nominal 3/4 inch SDR-11 tubing.

(2011) and the UBWT solution of Cimmino and Bernier (2014). The UIFT solution tends upwards towards the UHF solution as the effective borehole resistance is increased, and downwards towards the UBWT solution as the effective resistance is decreased. The effective borehole thermal resistance essentially quantifies the ability of the circulating fluid inside of the pipes to transfer heat to the surrounding ground. Configuration A of the boreholes results in a high resistance because of the distance between the tubes and the borehole walls, exacerbated by the short-circuiting heat transfer from the downward fluid to the upward fluid. The B configuration is the typical case. The C configuration is only in practice possible by using spacers and represents the lowest borehole resistance because the thermal interaction between the upward and downward flow is reduced, and the tubes are against the borehole wall. Table 2.1 shows three borehole configurations, utilizing shank spacings from A, B and C configurations with varied conductivity to exemplify the desired borehole resistance. The variation of the g-function curves are shown in Figure 2.4.

The UIFT g-function falls in between the UHF and the UBWT, where the UHF over approximates and the UBWT under approximates. Given that the stacked finite line source g-function tends upwards as the number of vertical segments is decreased,

Table 2.1: Effective borehole thermal resistance values used at H=96m for a 12x13 borefield.

Values varied	High R_b^*	Average R_b^*	Low R_b^*
Grout conductivity, k_g (W/(m·K))	0.64	1.0	2.77
Shank spacing, s (m)	0.0	0.022	0.066
Eff. Borehole Resistance, R_b^* ((m·K)/W)	0.2707	0.1643	0.0841

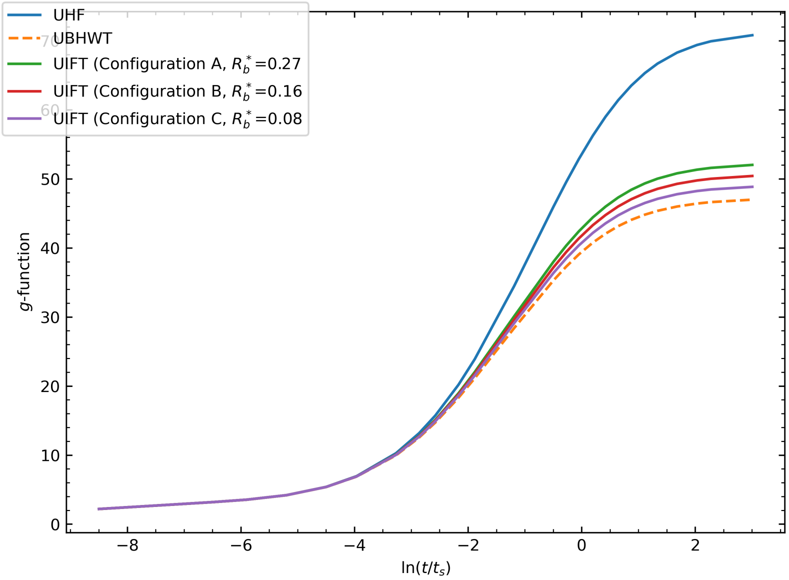


Figure 2.4: Boundary condition variation with a varied effective borehole thermal resistance for the UIFT g-function.

the UBWT boundary condition with less segments could approximate a UIFT g-function with many segments.

2.3.2.2 Typical Parameters

The original g-function library developed by Hellström contained g-functions for 27 dimensionless points in time, and about 8 curves at various B/H ratios for each configuration. Computing an exhaustive library of UIFT g-functions would require more degrees of freedom, and more computation time, given that the UIFT g-function is largely impacted by the effective borehole thermal resistance. In an attempt to create an expanded library utilizing UIFT g-functions, typical parameters are used. These values are only typical, not recommended, and are seldom correct. However, an average case UIFT g-function is believed to be a better approximation of reality than a g-function computed with the UBWT boundary condition.

The mass flow rate $\dot{m}$ is a required input to the UIFT g-function A typical recommended system flow rate for a GHE is 2.5 gallons per minute (GPM) per peak cooling ton (2.5 GPM/ton). Though drilling depth per cooling ton varies widely, a roughly typical value is 250 ft/ton. From these two values, a typical rate of 1 GPM/100 ft of borehole height may be found. This corresponds to about 0.2 kg per second for every 100 m of drilling for water (0.2 kg/s per 100 m).

Grouting material is backfilled in boreholes to protect groundwater and provide thermal contact. Though the grouting material intends to facilitate heat transfer from the pipes to the ground, the grout often has a lower thermal conductivity than the surrounding ground. Grouting materials that are higher in thermal conductivity and easier to pump (like graphite-enhanced mixtures) can be more expensive than acceptable. Table 2.2 contains grouting material and their thermal conductivity ranges. A typical thermal conductivity of the ground (or soil) used is 2.0 W/(m·K), and the typical thermal conductivity for grout is about 1.0 W/(m·K).

Table 2.2: Typical geothermal grouting materials and their ranges of thermal conductivity (Collins and McClain, 2009; GeoPro, 2021).

Grouting Material	Thermal Conductivity (W/(m·K))
Pure Bentonite	0.64-0.74
Bentonite/Sand Mixture	0.90-2.0
Graphite-Based Mixture	1.36-2.76

Standard high-density polyethylene (HDPE) SDR-11 pipe sizes are selected from, and are tabulated in Appendix A. The smallest pipe is chosen, and when the head loss exceeds 15 ft, a larger pipe is selected. The head loss, h_L is dependent on the Darcy-Weisbach friction factor, f, the diameter of the pipe, D, the average velocity

V and the gravitational constant g_c.

$$h_L = f \frac{L}{D} \frac{V^2}{2g_c} \tag{2.23}$$

The friction factor simplifies to $f = 64/Re$ for laminar flow. The friction factor inside a pipe in turbulent flow is a function of Reynolds number, Re and the relative roughness ϵ/D. Colebrook's iterative equation served as the basis for the Moody chart and is used to calculate the friction factor (Çengel and Ghajar, 2015).

$$\sqrt{f} = -2.0log_{10}\left(\frac{\epsilon/D}{3.7} + \frac{2.51}{Re\sqrt{f}}\right) \tag{2.24}$$

A pipe roughness for HDPE piping, $\epsilon = 2.0 \times 10^{-5}$ is made use of. Table 2.3 provides the discrete heights that are used in the generation of a database. The associated pipe sizes, which do not exceed 15 ft of pressure drop, are given. The intermediate values of friction factor and Reynolds number are provided for reproducibility. The mass flow rates provided are on a per borehole basis, $\dot{m}_b$ so to get the system mass flow rate the number of boreholes would need to be multiplied.

Table 2.3: Discrete heights to be computed in database, the associated pipe sizes, the head loss and intermediate values.

H (m)	Nominal Size (in)	D_i (mm)	D_o (mm)	f	Re	h_L (ft)	$\dot{m}_b$ (kg/s)
24	0.75	21.83	26.67	0.045	2795	2.70	0.048
48	1	27.33	33.4	0.039	4465	6.10	0.096
96	1.25	34.49	42.16	0.035	7077	13.41	0.192
192	2.5	59.75	73.03	0.033	8170	6.56	0.384
384	3.5	83.13	101.6	0.030	11744	9.14	0.768

2.3.2.3 Determination of Optimal Segments

With a g-function library in mind, 5 heights are selected to be computed; 24m 48m, 96m, 192m and 384m. Typical borehole layouts are made use of and displayed in Table 2.3. The mass flow rates are increased to ensure the Reynolds number inside of the borehole stays turbulent.

Converged UIFT g-functions (utilizing 4m segment lengths) are computed for the 5 heights at configurations ranging from 2x2 to 32x32 boreholes and are stored. These are all computed using a uniform borehole spacing of 5m. These reference cases, computed with the UIFT boundary condition, are aimed to be replicated using UBWT with a reduced number of segments. To discover the number of segments that should be utilized in the UBWT g-function calculation to approximate the typical case UIFT g-functions, the UBWT g-functions are computed over a range of segments at each height. The mean percent error over the entire g-function curve is computed with the reference being the converged UIFT case. The points plotted in Figure 2.5 represent the specific mean percent error points based on the number of segments used in the UBWT calculation, where the lines in between the points represent a linear interpolation curve. To determine the ideal number of segments to approximate a UIFT g-function with the UBWT calculation, a linear interpolation is used at each height to solve for the number of segments that makes the mean percent error equal to 0. In other words, the root is found. These intersection points are marked as red X's in Figure 2.5, and a vertical dashed line is drawn down to where the optimum number of segments per borehole is. These results are stored. This process of finding the ideal number of segments per borehole at each height is done for each field.

The process of determining the ideal number of segments is done at each height and for each of the borefields with reference curves (2x2 to 32x32). The optimum segment lengths are shown in Figure 2.6a and the optimum number of segments are shown in Figure 2.6b. There are trends associated with each height over the different borefields. At each height, once the total drilling depth becomes sufficiently large, the optimum number of segments appears to approach a set number of segments.

Once the combination of number of boreholes and borehole depth becomes great enough, it is not feasible to calculate the reference case because the memory require-

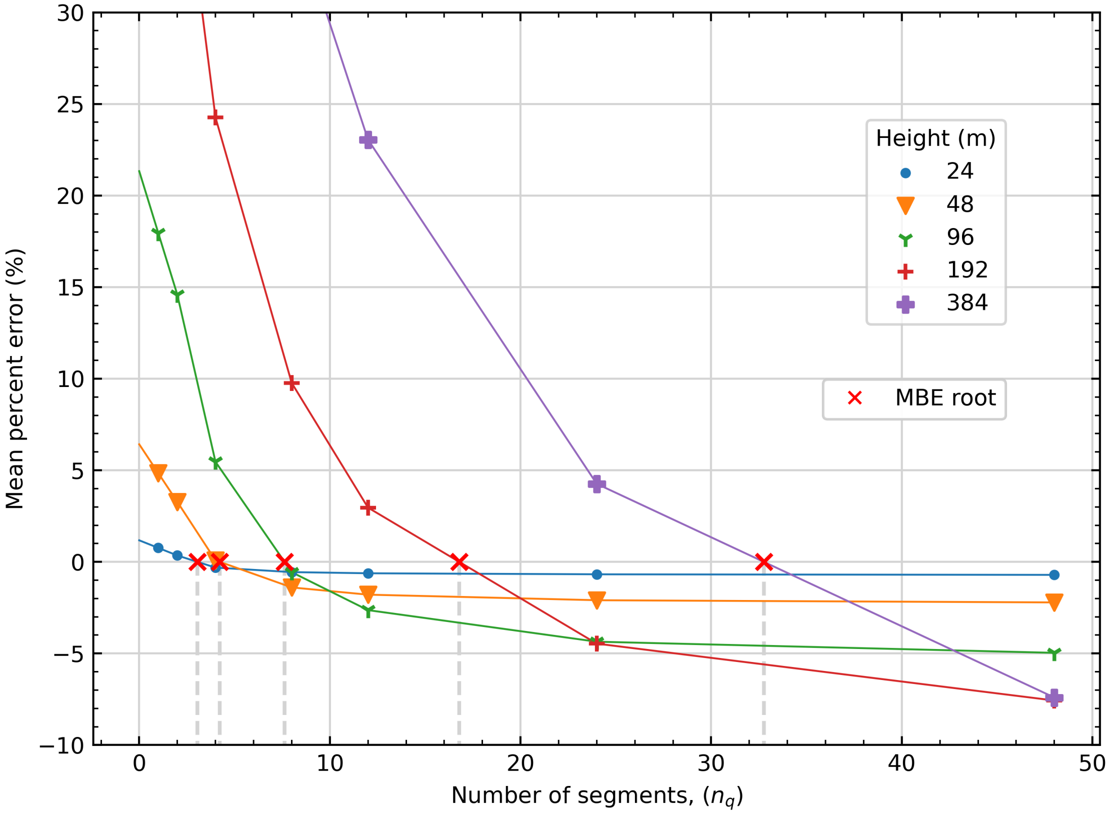
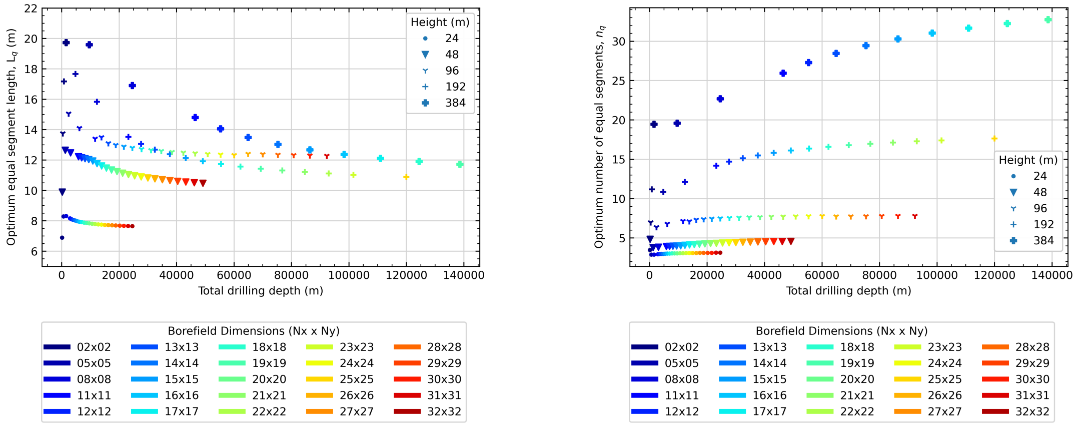


Figure 2.5: Finding the root of the mean bias error for a 19x19 borefield at 5 different heights.

ments exceed the available memory on the available cluster nodes. For this reason, reference cases cannot be computed for all combinations of number of boreholes and borehole depth. Therefore, not all of the optimum segment lengths (Figure 2.6a) are plotted. The optimum segment length (left) up to 96m can be computed for all the fields, while the 192m height cases can only be compared up to a 25x25 and the 384m up to a 20x20. For example, a 32x32 field at a height of 384m made up of 4m long segment lengths for 30 time steps would consume on the order of 2.3 terabytes of RAM using the *pygfunction* (version 1.1.1). The cluster has no compute nodes available with this much memory.

Table 2.4 describes the adaptive segment scheme. Each of the depths computed for the library creation were chosen to double at each step and be divisible by 4.



(a) (b)

Figure 2.6: Optimum segment lengths (a) and number of segments (b) given total drilling depth and a specific height in a field.

This was done because all of the g-functions are computed with a burial depth of 2m, and similarities can be increased. Based on a depth and a range of boreholes, the adaptive discretization scheme "snaps" to a segment length selection which will both increase similarities and very closely approximate the UIFT calculation. A depiction of how the 2m burial depth and 4m segment discretization can increase similarities is displayed in Figure 2.7.

Table 2.4: Discretization scheme based on number of boreholes and height.

Depth (m)	Range (NBH)	Segment Length (m)	Number of Segments / BH
24	All	8	3
48	All	12	4
96	All	12	8
192	NBH <120 BH	16	12
192	NBH >= 120 BH	12	16
384	NBH <220 BH	16	24
384	NBH >= 220 BH	12	32

2.3.3 Interpolation

Though not part of *cpgfunction*, GHEDT generates g-functions for specific heights and a spline-based interpolation scheme is used based on the number of computed

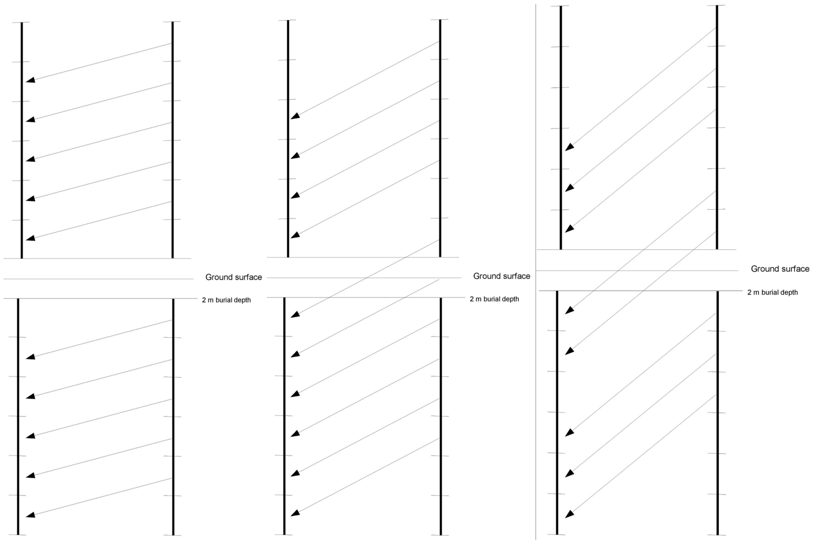
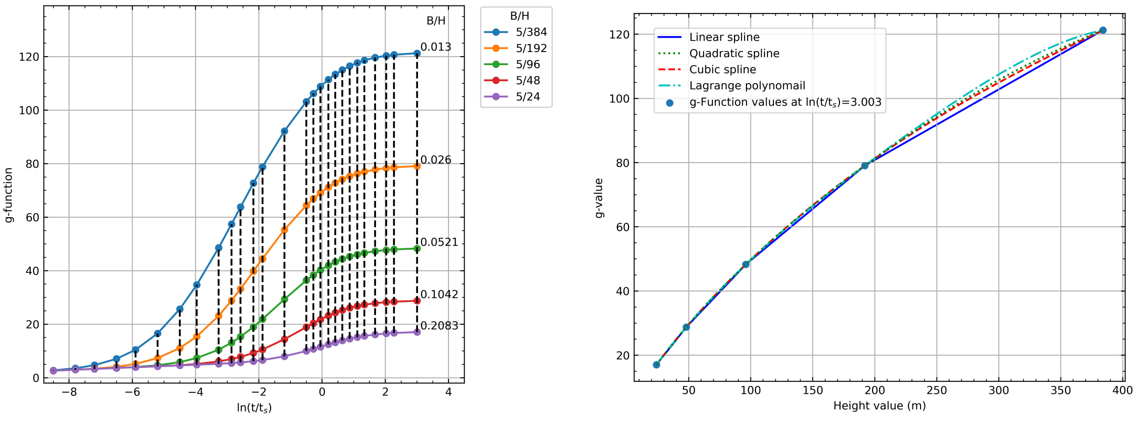


Figure 2.7: Depiction of similarities across the ground surface with 4m discretization increments based segment lengths with 2m burial depths.

curves. The interpolation procedure utilizes a spline fit at each of the dimensionless points in time. At each of the points in time computed, the g-function points over the range of heights are fitted to a spline. For example, g-functions at various B/H ratios are shown in Figure 2.8a, and a linear interpolating spline is drawn between the height values at each dimensionless point in time. Figure 2.8b takes a close look at the interpolation scheme for the last dimensionless value, $\ln\left(\frac{t}{t_s}\right) = 3.003$, over the range of height values. Various spline curves (linear, quadratic, and cubic) are shown, as well as a Lagrange polynomial fit. SciPy's (Virtanen et al., 2020) (version 1.6.2) interpolation package (*scipy.interpolate*) is utilized. The *interp1d* function is used for the spline interpolation and the *lagrange* function it utilized to create the Lagrange polynomials. The minor error introduced by the interpolation procedure, when compared to a reference g-function, is discussed in subsection 2.4.3.



(a) (b)

Figure 2.8: A visualization of a linear spline interpolation (left) and a visualization of various spline interpolation (linear, quadratic and cubic) curves and a Lagrange polynomial at ln(t/ts)=3.003.

2.4 Results

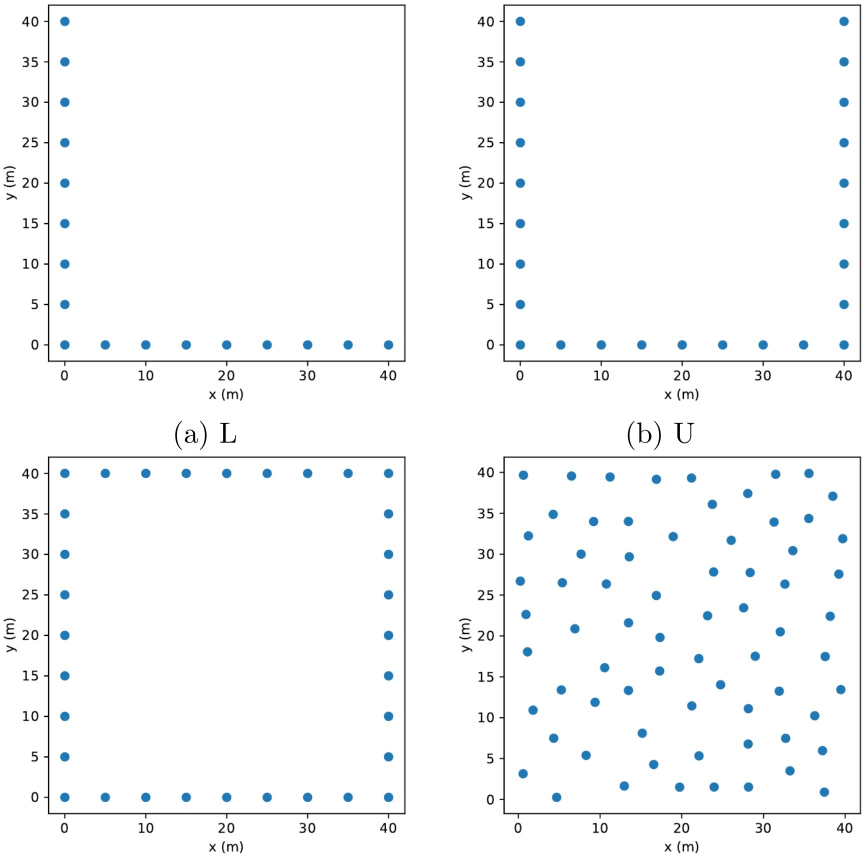
2.4.1 Validation and Performance of *cpgfunction*

This section shows a validation of *cpgfunction* to *pygfunction* for example cases and compares the performance of the two programs.

2.4.1.1 Verification and Timing for Example Cases

The results include a validation of accuracy; comparisons of RAM utilized; and comparisons of timing for high-similarities and low-similarities cases.

Cpgfunction has been verified against *pygfunction* for a variety of cases. The UBWT boundary condition in *cpgfunction* matches the results of *pygfunction* quite accurately. Consider the four cases shown in Figure 2.9 – an L-shape, U-shape, open rectangle, and a configuration generated with Poisson disk sampling. A Poisson disk sampling algorithm implemented by Hill (2017), based on Bridson (2007), is used to generate this field and other fields (below) with reduced similarities. Compared to a true random placement of boreholes, using the Poisson disk sampling algorithm eliminates the possibilities of having boreholes closer together than a minimum distance, or farther apart than twice that distance. A minimum distance of 5m is used. Figure 2.10 shows the g-functions computed with each program; the root mean square error (RMSE) differences between the 27 points of each g-function, is less than 0.1%.



(a) L

(b) U

(c) Open

(d) Poisson disk

Figure 2.9: Borehole configurations - illustrative test cases.

Table 2.5 summarizes the time required by the two programs; the last column gives the ratio of the time required by *pygfunction* to that required by *cpgfunction*. It could be considered the “speed ratio”, quantifying how many times faster *cpgfunction* is than *pygfunction*. The g-function is represented as a series of g-values, computed for specific values of $\ln(t/t_s)$. The amount of time to compute a g-function varies linearly with the number of g-values computed. All g-functions in this Chapter are represented by 27 g-values. The speed ratio is related to the relative prevalence of

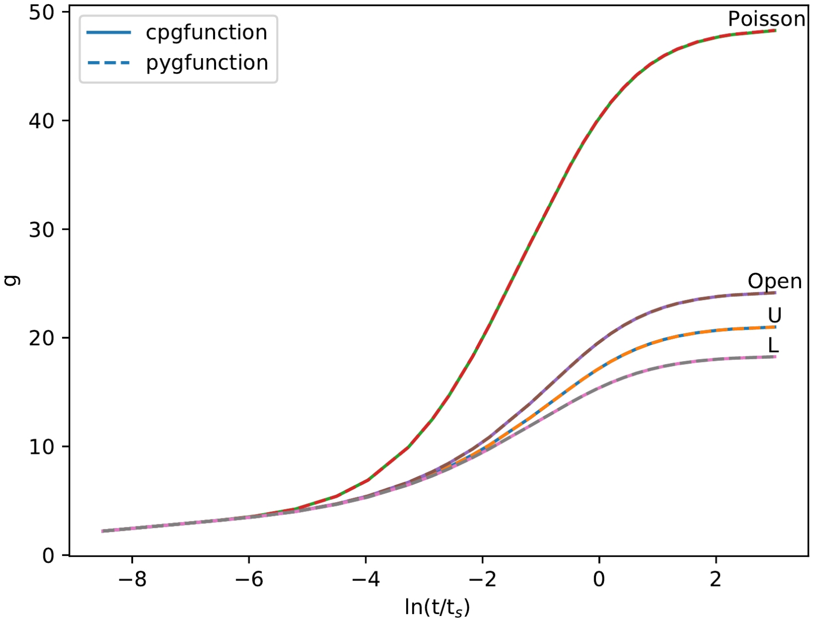


Figure 2.10: Comparison of g-functions calculated with *cpgfunction* and *pygfunction*.

similar segment pairs. Cases with higher numbers of similar pairs have lower ratios. Ordering the cases as Poisson Disk, L, U, Open rectangle, orders the ratio from highest to lowest and the prevalence of similar pairs from lowest to highest.

Table 2.5: Summary of example cases.

Config.	**Number sources**	**Pygf. time (s)**	**Cpgf. time (s)**	**Ratio**
L	204	19.7	1.6	12.7
U	300	22.9	2.3	9.9
Open	384	23.3	3.7	6.4
Poisson Disk	840	168.0	11.0	15.3

This trend can be explained as follows. The g-function calculation occurs in two parts: (1) computing segment-to-segment responses and filling the segment matrix, and (2) building and solving systems of equations. In process (1), *cpgfunction* calculates segment-to-segment response factors notably faster than *pygfunction* (version 1.1.1), due to the advantage of using a compiled language (C++ instead of Python.) Both programs incorporate similarity identification presented by Cimmino (2018a) to reduce the number of necessary segment-to-segment responses (finite line source integrations) to be computed. Segment-to-segment responses that are similar contain pairs of segments with identical or near-identical horizontal and vertical offsets, such that the segment-to-segment responses are identical or near-identical. However, a high prevalence of similar segments reduces the advantage of *cpgfunction*. This occurs because less time is spent in process (1), and the remainder of *cpgfunction* contains less vectorization than *pygfunction*. Therefore, the building and solving systems of equations that adjust the distribution of heat are comparatively slower in *cpgfunction* (version 1.0) than in *pygfunction*. The segment response matrix must be indexed by Equation 2.21, thus *cpgfunction*'s speed relies heavily on multithreading. The superposition process is of order $O(n!)$ (Big O notation), and is therefore very computationally intensive. *pygfunction* contains vectorization in the superposition process, where *cpgfunction* only contained multi-threading. However, in later development (*cpgfunction* v2.0), a vectorized linear algebra function, written in Fortran, is utilized that can perform matrix vector multiplication with a packed symmetric matrix. Vectorization is significantly faster than sole parallelization. This is discussed in subsection 2.5.1 *cpgfunction* and *pygfunction*.

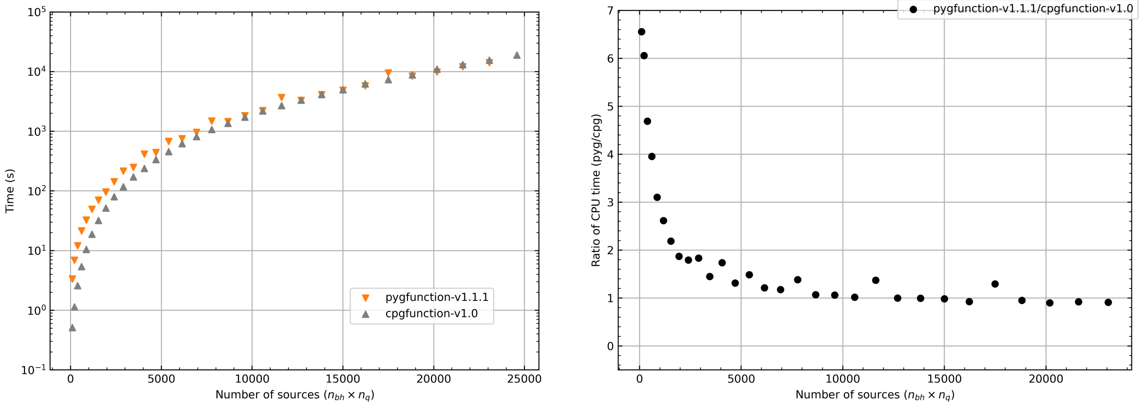
2.4.1.2 Results for Fields with High Similarities

As discussed in the last section, large rectangular borehole configurations have large numbers of identical segment pairs, which reduces the speed advantage of *cpgfunction* compared to *pygfunction*. In this section, *cpgfunction* v1.0 and *pygfunction* v1.1.1 are compared. Later developments (in both *pygfunction* and *cpgfunction*) are discussed and compared in subsection 2.5.1 *cpgfunction* and *pygfunction*. The comparison in this section is without the adaptive discretization scheme.

To compare the performance of *cpgfunction* and *pygfunction* for cases with high similarities, g-functions were calculated for uniformly spaced square configurations from 2x2 to 32x32 boreholes. A horizontal spacing of 5m and a depth of 96m were utilized, with 24 segments per borehole. For each case, the computation time was determined within the code by polling the system clock at the beginning and end of the calculation. The RAM consumption is provided by the Simple Linux Utility for Resource Management (SLURM) (Yoo et al., 2003), which OSU HPCC makes use of for job scheduling.

Figure 2.11a shows a comparison of the computational time for the 30 of the 31 cases, plotted against the total number of sources $n_{bh} \times n_q$ in the field. (Even using a compute node with 768 GB of RAM, *pygfunction* ran out of RAM for the 32x32 borehole case, so *pygfunction* results are not shown for this case.) The ratio of the time required for *pygfunction* to that required by *cpgfunction* is plotted in Figure 2.11b. Up to about 10,000 sources, *cpgfunction* is faster than *pygfunction*; above 10,000 sources the two programs have similar computational speed, with *cpgfunction* being slightly slower than *pygfunction* for some cases.

As can be seen in Figure 2.11a, there is some "wobble" in the results – slight deviations from a smooth curve with *pygfunction.* The reason is not known, but it seems likely that other processes on the compute node may have small effects on the computational time.



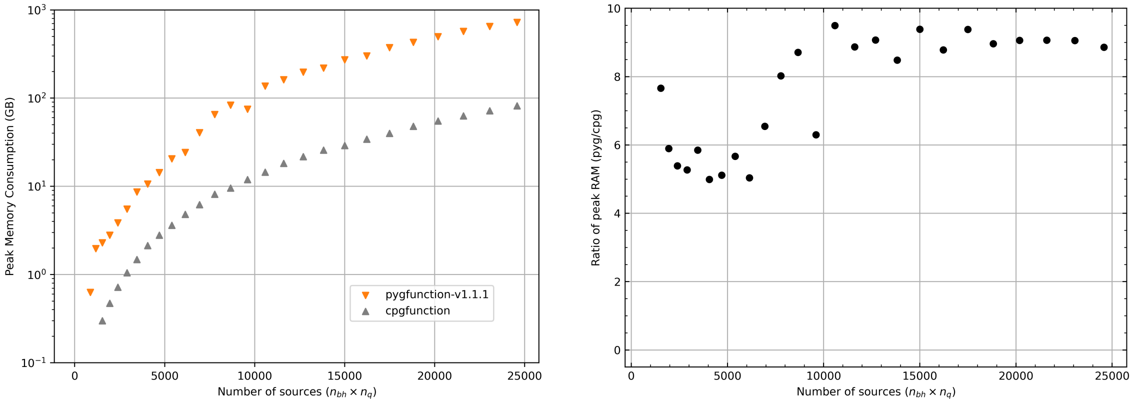
(a)

(b)

Figure 2.11: Timing comparisons for square configurations (left) and ratios (*pygfunction*/*cpgfunction*) of computing time.

Pygfunction uses the NumPy package (Harris et al., 2020) which provides high level access to low-level vectorized matrix operations. Here, the computational speed of *pygfunction* exceeds that of *cpgfunction* with configurations containing high number of sources and similarities in large part due to the similarity identification methods of Cimmino (2018a), which reduce the amount of time spent in pure Python code while calculating segment-to-segment response factors. (Again, the more similar segment pairs, the greater the possibility for *pygfunction* to have a computational edge in speed.) After the segment responses are computed, the remainder of *pygfunction*'s UBWT g-function calculation depends heavily on NumPy matrix operations which provide access to low-level highly optimized Fortran functions.

Figure 2.12a compares the memory requirements for the square borehole configurations. Figure 2.12b shows the ratio of peak RAM consumed by *pygfunction* to that consumed by *cpgfunction*. For larger fields where the RAM becomes a limiting factor, *pygfunction* uses about seven to nine times as much RAM. Picking just one example – the case with 5400 sources corresponds to a borehole field with 225 boreholes in a 15x15 configuration. For this case, *pygfunction* requires 20.5 GB of RAM – a value that will exceed the available RAM on all but the highest spec desktop computers. *Cpgfunction* will only require 3.6 GB – a value that is commonly available on modern desktop computers.



(a)

(b)

Figure 2.12: Comparison of peak RAM usage (left) and the ratio (*pygfunction*/*cpgfunction*) of peak RAM usage.

2.4.1.3 Results for Fields with Low Similarities

To investigate the speed of the two programs for more irregular cases, a series of Poisson disk configurations were calculated. These fields were chosen based on having the same surface area as the rectangular cases above, but the aspect ratio (width to length of the field) was chosen as 4. Then, for each rectangle, a Poisson disk distribution was created. A sample field of 900 m^2 area, with 28 boreholes is shown in Figure 2.13. Figure 2.14a shows a comparison of the computation time required for the two programs with the Poisson disk fields, where there are comparatively low numbers of similar pairs. As shown in Figure 2.14b, *cpgfunction* is significantly faster than *pygfunction* for all cases; as the number of sources gets high, *cpgfunction* is about 4 times faster than *pygfunction*. A later version of *pygfunction* (version 2.0) vectorized the similarity identification as well as the calculation of the segment response integrals, which proved to be much faster than solely multi-threading (see subsection 2.5.1 *cpgfunction* and *pygfunction*).

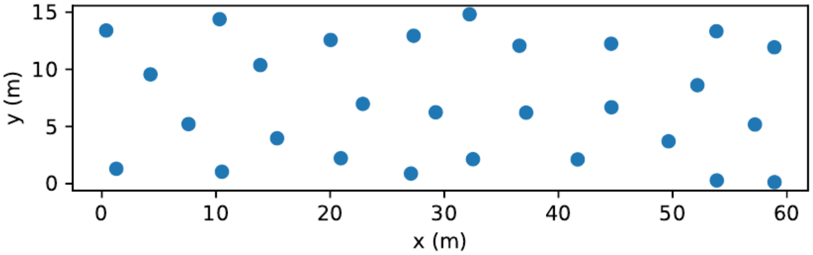
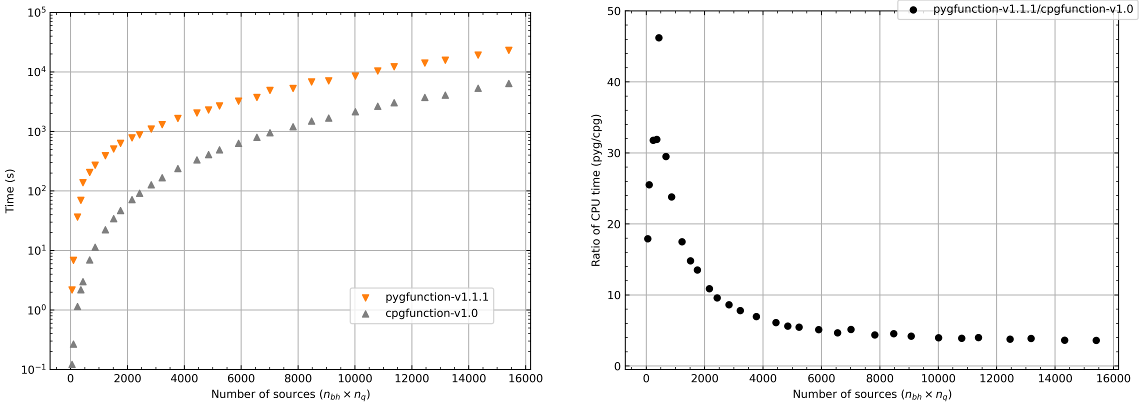


Figure 2.13: Sample Poisson disk field.

2.4.1.4 Example

For purposes of example, a g-function for a system installed in the 1990s was calculated with both *pygfunction* and *cpgfunction*. The 320 borehole system (Dinse, 1998) was installed at Daniel Boone High School in Washington County, Tennessee. The boreholes are 46m deep and are laid out in 16 groups of 20 boreholes each. Each group is a 4x5 rectangle with 4.6m spacing. The field consists of two rows of 8 groups,



(a) (b)

Figure 2.14: Timing results for the Poisson disc fields (left) and the ratio (*cpgfunction*/*pygfunction*) of computing time.

with spacing of 6.1m between each group, as illustrated in Figure 2.15. Thus, there are considerably more similarities than there would be in a 320-borehole Poisson disk field, but less than there would be in a uniformly-spaced 32x10 rectangular field. For both programs, 12 segments per borehole were used for a total of 3840 segments.

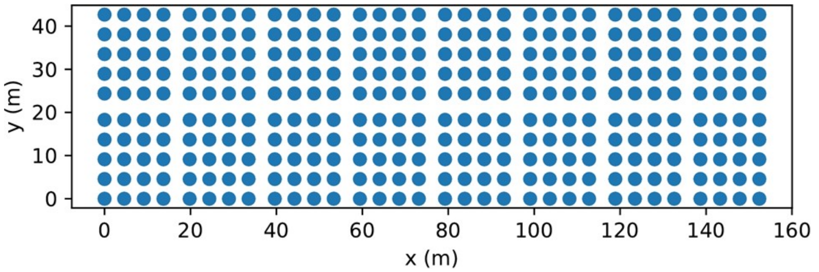


Figure 2.15: Example borehole field, a real world system installed at Daniel Boone Highschool in Washington County, Tennessee (Dinse, 1998).

The two g-functions are shown in Figure 2.16. The RMSE difference between the two g-functions, each with 27 points, is less than 1%. For this case, *pygfunction* required 10.8 GB of RAM and *cpgfunction* required 2.0 GB. The computation time for each step is summarized in Table 2.6. As can be seen, *cpgfunction* has a significant advantage in calculating the segment-to-segment response factors. Also noticeable is the fact that *pygfunction*, by being able to take fuller advantage of very fast linear algebra libraries is faster at building and solving the system of equations.

Table 2.6: Summary of 320-borehole example.

Step	**Pygf. time (s)**	**Cpgf. time (s)**	**Ratio**
Identifying similarities	258.5	29.9	8.6
Calculating segment-to-segment response factors	293.1	2.1	139.6
Building and solving system of equations	108.3	197.7	0.55
Total	662	229.8	2.8

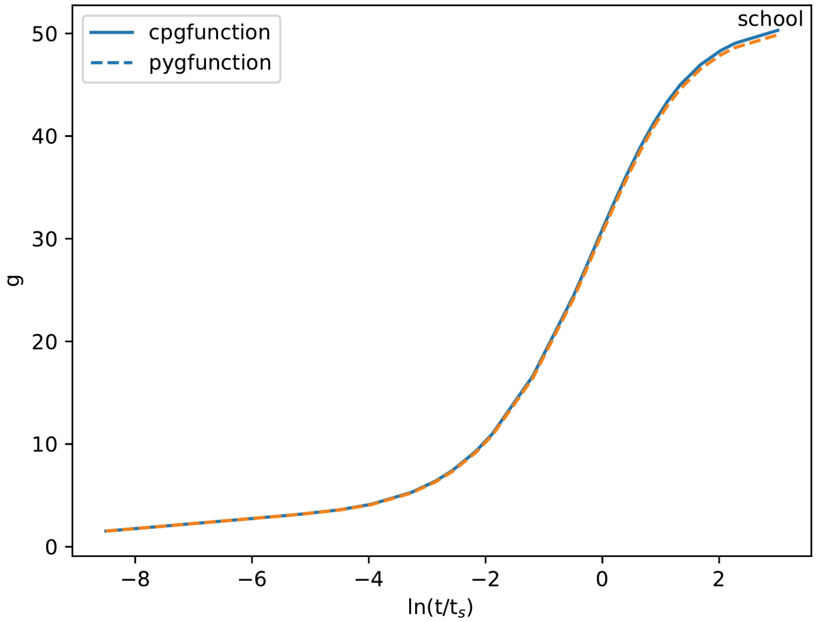


Figure 2.16: Example borehole field g-functions.

2.4.2 Performance of *cpgfunction* with the Adaptive Discretization Scheme

The adaptive discretization scheme is implemented for the fields ranging from 2x2 to 32x32. The scheme is implemented, and the mean percent errors are computed and plotted in Figure 2.17. The mean percentage is computed over the entire g-function curve at all 27 dimensionless points.

Figure 2.17 shows the mean percentage error of *cpgfunction* using the adaptive discretization scheme. The range of errors shown (-1.5% to 0.7%) validates that the scheme has been implemented correctly, and that the discretization of Table 2.4 keeps the results in an acceptable range. However, this plot is made for the same fields as were used to develop the adaptive discretization scheme, as described in subsubsection 2.3.2.3 Determination of Optimal Segments. To investigate the accuracy for other configurations, new fields are generated using the Poisson-disc sampling routine. These fields use the same areas that the 2x2 to the 32x32 does, but the ratio of length to width is changed to ensure the discretization scheme does not only apply to fields on a square layout with regular spacing. These fields are configured as follows:

$$A_1 = A_2 \tag{2.25}$$

$$l_1 w_1 = A_1 \tag{2.26}$$

$$l_2 = 1/2 l_1 \tag{2.27}$$

Thus $2w_1 = w_2$. The Poisson disc sampling is applied so that no boreholes fall within a radius of 5m to one another. The reference cases using UIFT at 4m segment lengths are computed, and the UBWT g-functions are computed using the adaptive discretization scheme for these fields. The results are displayed in Figure 2.18, where the land areas and number of boreholes can be seen in the lower legend. For the vast majority of cases the results are quite good: -2.5% to +0.5%. The few larger errors (less than -4%) are well within the range of typical uncertainties for design (about ±10%). Further research could be done to improve the scheme in the future. Later research into an adaptive discretization scheme by other members in the research group[3] proved that even less segments were necessary for use in an adaptive discretization scheme by utilizing segments of unequal length. Shorter segment lengths were utilized near the top and bottom of the borehole. This author later contributes to *pygfunction* version 2.1 to enable unequal segment length functionality (discussed in Chapter IV Borehole Configuration Selection in the GHEDT).

The following is an example to show the increase in performance provided by the

[3] The research group consists of an undergraduate research assistant, Timothy West, the author and the author's advisor, Dr. Jeffrey Spitler.

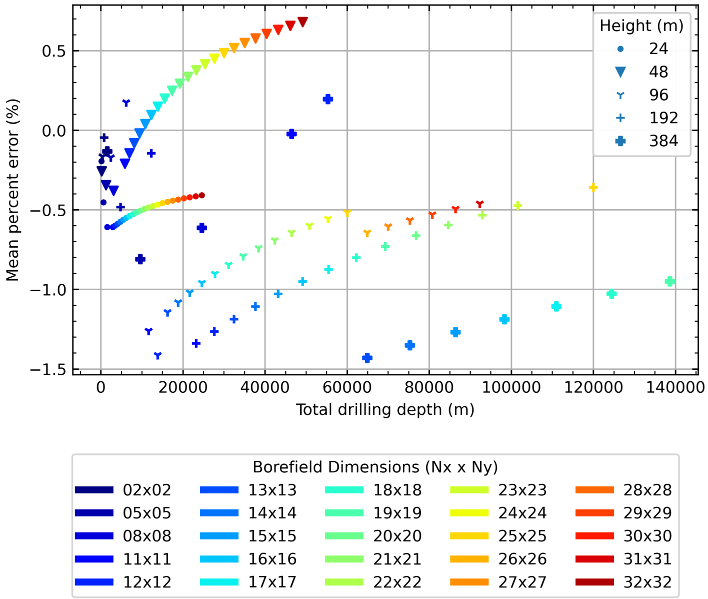


Figure 2.17: Segment length discretization scheme validation for square fields with uniform spacing.

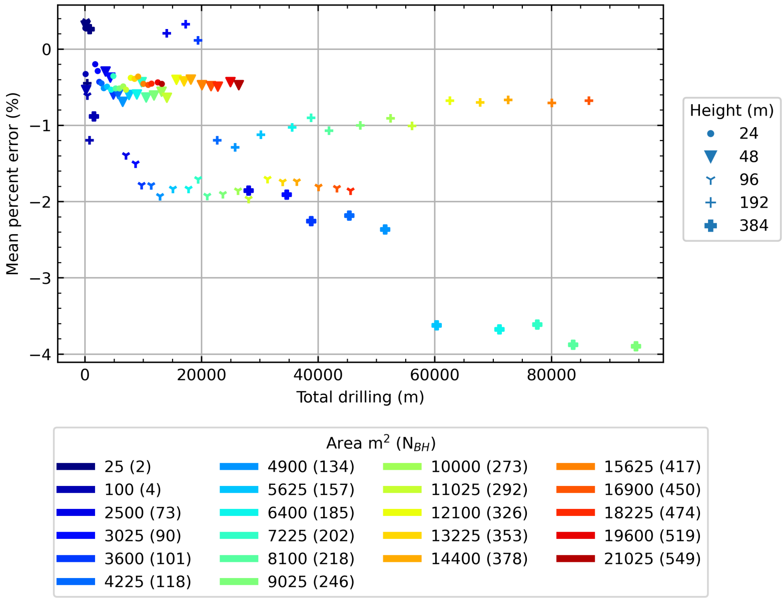
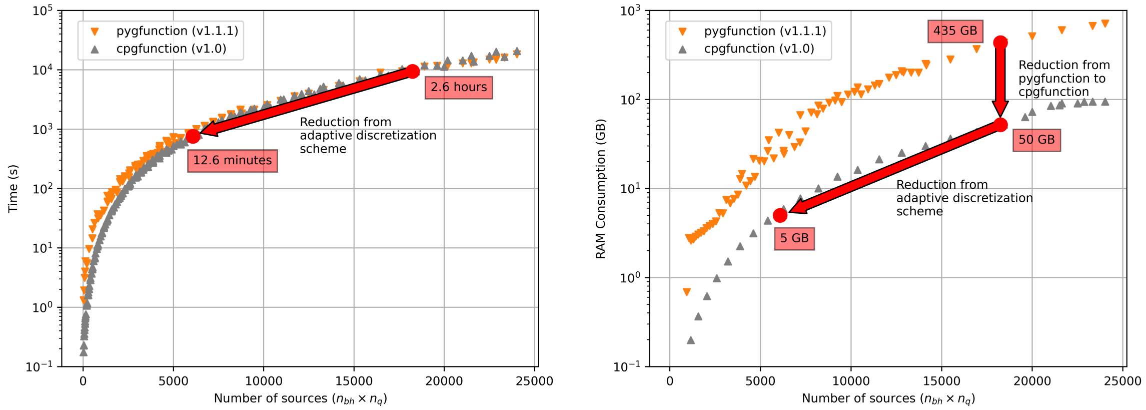


Figure 2.18: Validation of the discretization scheme for the Poisson-disc sampled fields ranging small to large land areas.

adaptive segment scheme. Consider a 19x20 field (380 boreholes) at a depth of 192m computed with 4m segment lengths for a UIFT reference case. This would be a total of 18,240 sources. The amount of RAM consumed would be 435 GB using *pygfunction*, but only 50 GB for *cpgfunction* as can be seen in Figure 2.19b (an 8-fold reduction of RAM). The amount of time saved in going from *pygfunction* to *cpgfunction* in this case is little to none. Then the adaptive discretization scheme is applied to the UBWT boundary condition so that it can approximate the highly accurate converged UIFT solution. There is a reduction in RAM from 50 to 5 GB and a decrease in time from 2.6 hours to 12.5 minutes. The time reduction from the adaptive discretization scheme is given in Figure 2.19a. The number of segments used is reduced from 18,240 to 6080.



(a)

(b)

Figure 2.19: The performance increase by use of *cpgfunction* and adaptive discretization scheme; significantly faster computation and less memory consumed.

2.4.3 Performance of Interpolation Procedure

For the database of g-functions, discrete values of borehole spacing, height and burial depth are used for the calculations to give a set of g-functions for each configuration. To then use the database in design, an interpolation scheme must be implemented. This section presents and compares various interpolation fits and examines the implications of the interpolation. As discussed by Eskilson (1987) and Cimmino and Bernier (2014), the g-function scales based on four dimensionless variables: t/t_s, r_b/H, D/H and B/H. This allows, for example, computing the response of a configuration with a specific value of B, and by interpolation between g-functions for different B/H ratios, utilizing the results for other B spacings. Though Claesson and Eskilson

(1988) claimed that the g-function had a weak dependence on D/H, the dimensionless value is considered in this analysis.

An example here is based on a rectangular 7x10 borefield, for which g-functions are computed for a borehole spacing of 5 and 20 meters. Even though the borefields have different borehole spacings, the inputs that contain the same dimensionless parameters should produce equal g-functions. These are computed using the methodology of Cimmino and Bernier (2014) and could been seen as a verification that the dimensionless ratios apply the way Claesson and Eskilson (1988) originally discussed them. Furthermore, an interpolation procedure is introduced, and the error associated with interpolation is presented.

The dimensional and dimensionless values for the g-functions computed are given in Table 2.7 and Table 2.8. The g-functions containing the same dimensionless values are nearly identical. The errors are summarized in Table 2.9, confirming that the g-functions do indeed scale with *B/H.* The g-functions are plotted in Figure 2.20.

Table 2.7: Summary of dimensional and non-dimensional parameters for the 7x10 borefield with uniform spacing of B=5m.

B (m)	D (m)	H (m)	r_b (mm)	B/H	D/H	r_b/H
5	2	24	75	0.20833	0.08333	0.00313
5	2	48	75	0.10417	0.04167	0.00156
5	2	96	75	0.05208	0.02083	0.00078
5	2	192	80	0.02604	0.01042	0.00042
5	2	384	87.5	0.01302	0.00521	0.00023

Table 2.8: Summary of dimensional and non-dimensional parameters for the 7x10 borefield with uniform spacing of B=20m.

B (m)	D (m)	H (m)	r_b (m)	B/H	D/H	r_b/H
20.0	0.66667	32.0	0.016	0.6250	0.0208	0.0005
20.0	1.0	48.0	0.024	0.4167	0.0208	0.0005
20.0	2.0	96.0	0.048	0.2083	0.0208	0.0005
20.0	4.0	192.0	0.096	0.1042	0.0208	0.0005
20.0	8.0	384.0	0.192	0.0521	0.0208	0.0005

To perform a check that the interpolation procedure that has been implemented

Table 2.9: Mean percent errors for g-functions with the same dimensionless values.

B (m)	H_0 (m)	H_1 (m)	H_2 (m)	H_3 (m)	H_4 (m)
5	96	48	24	12	8
20	384	192	96	48	32
MPE (%)	-3.76E-14	-9.51E-14	5.63E-14	1.47E-13	1.23E-05

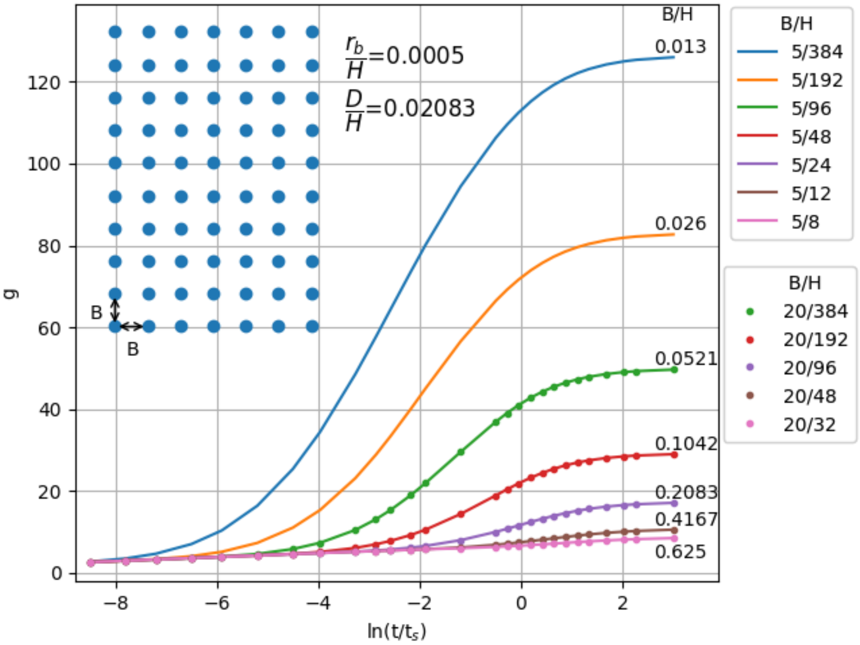


Figure 2.20: g-Functions computed for a borehole configuration of 7x10 with the same dimensionless values.

is accurate, the dimensionless borehole radius and burial depth values are made equal so that the only variation is the B/H value. This will not be the case in the library, but from the following example it can be assumed that the interpolation is properly implemented. Additionally, the error associated with using fewer g-function curves than have been computed for the library is investigated. This information will be useful later - when a live-time g-function needs to be computed, it will be known how many bracketing g-function curves should be computed.

Based on Figure 2.8b it appears that the maximum difference between the interpolating procedures is when the height values are furthest apart and will produce the maximum error in interpolation. The maximum error is quantified by computing a reference g-function with equal dimensionless values other than the B/H ratio. A reference g-function is computed at a height of 288m, and the various interpolation procedures are utilized. The mean percent errors are computed over the 27 dimensionless points in time. Table 2.10 shows the mean percent error based on a number of g-function curves utilized, and the interpolation procedure used. For the first row of Table 2.10, 5 g-functions computed at 24, 48, 96, 192, and 384 m are used. The second row uses the last four depths; the third row uses the last three depths, and the 4^{th} row interpolates between 192 and 384m depths. Interpolation cannot be performed with less than 2 curves. Here, the error of the quadratic spline interpolation is the least. The error of the curve will change based on where the interpolation is being done, and how far apart the curves are. This is an extreme case; most errors will be lower.

Table 2.10: MPE values associated with the kind of interpolation technique used based on the number of reference g-functions.

# of g-Functions	Linear spline error (%)	Quadratic spline error (%)	Cubic spline error (%)	Lagrange polynomial error (%)
5	-0.727	0.052	0.111	-0.736
4	0.759	1.271	1.294	1.294
3	3.313	3.760		3.760
2	6.950			6.950

The errors are quantified over a range of heights utilizing all 5 g-function curves, and are shown in Figure 2.21. The absolute error for all the methods is less than 1%, and therefore any are acceptable for use. The quadratic spline maintains accuracy the best over the whole range. Errors caused by interpolation at the calculated height values (24m, 48m, 96m, 192m and 384m) are nearly 0. The absolute average errors are 0.36, 0.04, 0.05 and 0.33 for the linear, quadratic, cubic and Lagrange fits respectively.

The mean percent error over the entire g-function curve is useful to quantify, though the more relevant error to quantify is the error in sizing. In subsection 3.7.3 Borehole Heat Exchanger Effects on Sizing, Hellström's g-functions are interpolated

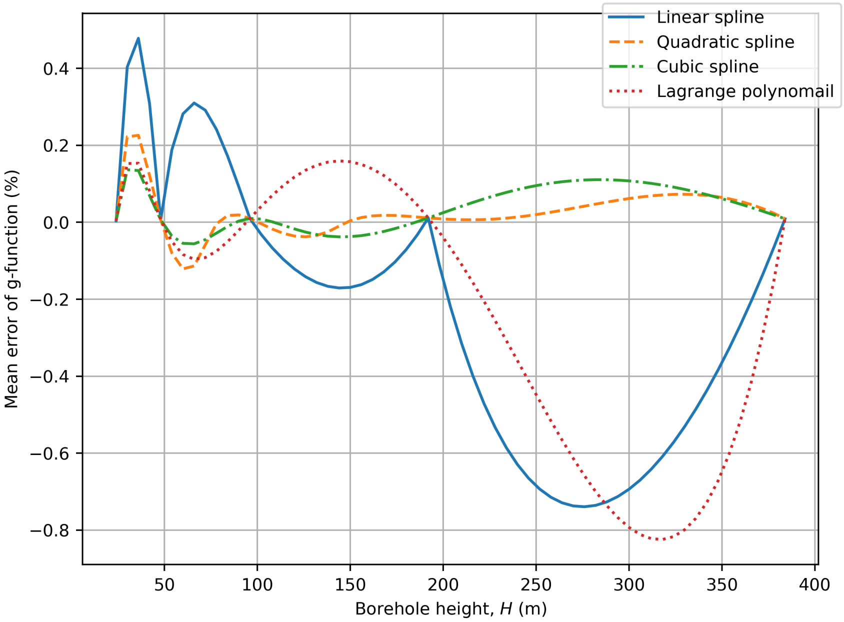


Figure 2.21: The mean percent error of the interpolated g-function curve at height values ranging from 24 to 384m utilizing various kinds of interpolation.

over and sized compared to GLHEPRO. GHEDT, for a single U-tube, is shown to be within less than 0.5%. In subsection 4.3.2 Integer Bisection Search, going from 5 to 3 curves is shown to have less than a 0.1% effect on the sizing of a system. The interpolation procedure is utilized in GHEDT. During the design process of GHEDT, after a field is selected, it must be sized. The user defines a minimum and maximum height range. Three g-function curves are computed. The quadratic spline is utilized for interpolation. The function will utilize a quadratic spline for interpolation as long as there are at least three g-function curves provided, will utilize linear if 2 curves are provided and is capable of working with one g-function curve as long as the requested height is the same as what is in the interpolation table.

2.4.4 Library

The use of *cpgfunction* coupled with the adaptive discretization scheme computed on the OSU HPCC “batch” nodes served as the basis for a new library containing over 34,000 configurations. Currently available libraries, implemented in eQuest (Liu and Hellström, 2006), GLHEPRO (Spitler, 2000), and EED (BLOCON, 2015) have less than 1000 possible configurations and are proprietary. Spitler et al. (2021) gives a brief description of this new, publicly available library containing g-functions for 34,321 configurations at 5 depths. Some of the configurations are available in existing libraries; others are new. The new configurations are C-shapes, lopsided-U-shapes and zoned rectangles.

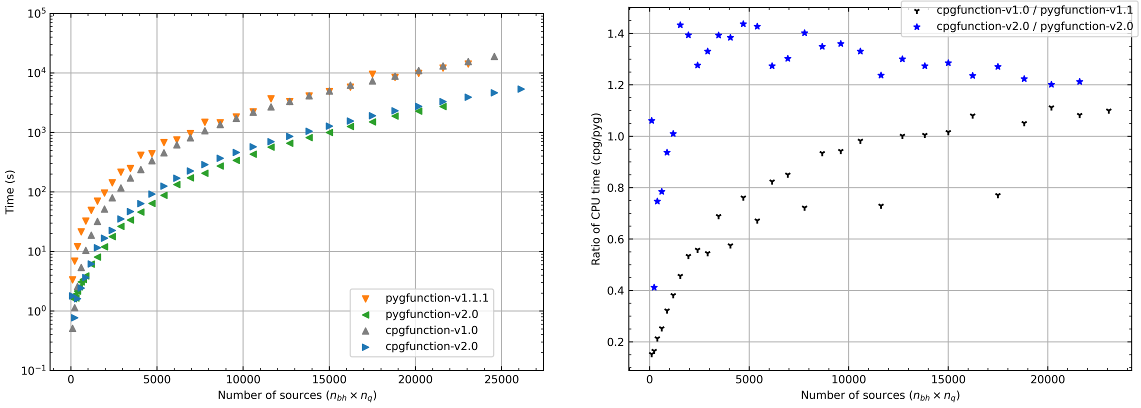
2.5 Recent Developments

2.5.1 *cpgfunction* and *pygfunction*

Recent developments have occurred for both *pygfunction* and *cpgfunction*. This section briefly describes the developments made in 2021 to both programs. Speed increases have occurred by further utilization of the processor. The speed improvements discussed here are on a pure computational basis. The code being utilized makes better use of the computer chip at each clock cycle. The NumPy package in Python provides high level interface for array programming, and is fast for matrix operations because it wraps the OpenBLAS library; an optimized Basic Linear Algebra Subprogram (BLAS) library that contains the Linear Algebra Package (LAPACK), written mostly in Fortran. The speed increase of *cpgfunction* from version 1.0 to 2.0 was garnered by utilizing an OpenBLAS function call that can do a symmetric packed matrix vector (SPMV) multiplication. The SPMV function accepts a packed symmetric matrix, and requires that the matrix be defined as an upper or lower packed matrix. The SPMV function can multiply the symmetric packed matrix by a vector, and the OpenBLAS function performs the operation with computational efficiency. The SPMV function is utilized in the temporal and spatial superposition process of *cpgfunction*, and resulted in on average more than a 60% increase in speed. Though the increase in speed of *cpgfunction* was over shadowed by the increase in speed from *pygfunction* version 1.1.1 to version 2.0. Cimmino focused on improving the speed for the identification similarities, and the calculation of the segment response factors. These processes were where *cpgfunction* previously had the advantage, as shown in Table 2.6. The similarity identification was previously done in pure Python syntax for *pygfunction* version 1.1.1. The finite line source integration of *pygfunction* version 1.1.1 was done utilizing the SciPy package. The SciPy package wraps the Quadrature Package (QUADPACK); a Fortran package for integration. The Gaussian-Kronrod quadrature integration is specifically the integration that is utilized. *Pygfunction* version 1.1.1 multithreaded the calculation of the similarities. For *pygfunction* version 2.0, Cimmino vectorized the similarity identification process, and it is now performed using NumPy array linear algebra. Additionally, the finite line source integration is vectorized by utilizing a SciPy function named "quad_vec".

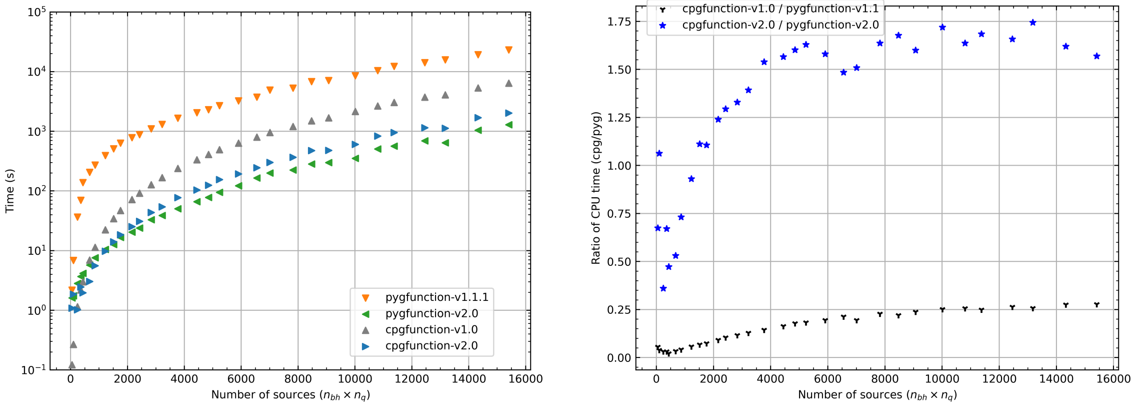
The speed of each of the versions for each program are shown in Figure 2.22a for square cases. The ratio of *cpgfunction* to *pygfunction* is plotted in Figure 2.22b, which quantifies how many times faster *pygfunction* is than *cpgfunction.* The version 1 comparison is shown to approach unity for very large cases, and for the largest cases *pygfunction* has the edge. Though, the version 2 comparison, *pygfunction* outperforms for nearly all cases. The speed comparisons for Poisson disc sampled fields is shown in Figure 2.23a. Figure 2.23b is the ratio of how many times faster *pygfunction* is than *cpgfunction.* The comparison for version 1 of each program is that *pygfunction* is about $\frac{1}{4}$ as fast for larger cases, though *pygfunction* version 2 gains an even greater edge compared to *cpgfunction* version 2 for fields with low similarities.

The memory consumption of *pygfunction* version 2.0 is reduced. Cimmino reduced the repetition of the segment response matrix definitions that were present in version 1.1.1. *Cpgfunction* still consumes less memory (the memory for *cpgfunction* version 1.0 and version 2.0 are the same), though the current usage of the packed symmetric matrix and the SPMV function makes the assumption that the segments along the borehole are equal. Even more recent developments to *pygfunction* (version 2.1)



(a) (b)

Figure 2.22: Timing comparisons for square configurations (left) and ratios (*cpgfunction/pygfunction*) of computing time for versions 1 and 2.



(a) (b)

Figure 2.23: Timing comparisons for Poisson configurations (left) and ratios (*cpgfunction/pygfunction*) of computing time for versions 1 and 2.

have proven that unequal segments can both reduce computation time (by reducing the total number of segments utilized) and increase the accuracy. Unequal segment lengths along with even further speed improvements of *pygfunction* are discussed in Chapter IV Borehole Configuration Selection in the GHEDT. While the speed improvements discussed here were on a purely computational basis, the improvement of Prieto and Cimmino (2021) is an entirely new way of considering how to solve the problem. *Pygfunction* benefits from the vectorization present in each process. The speed performance of equivalent borehole method (Prieto and Cimmino, 2021) is in part due to the previous successful attempts at vectorizing the code in *pygfunction*.

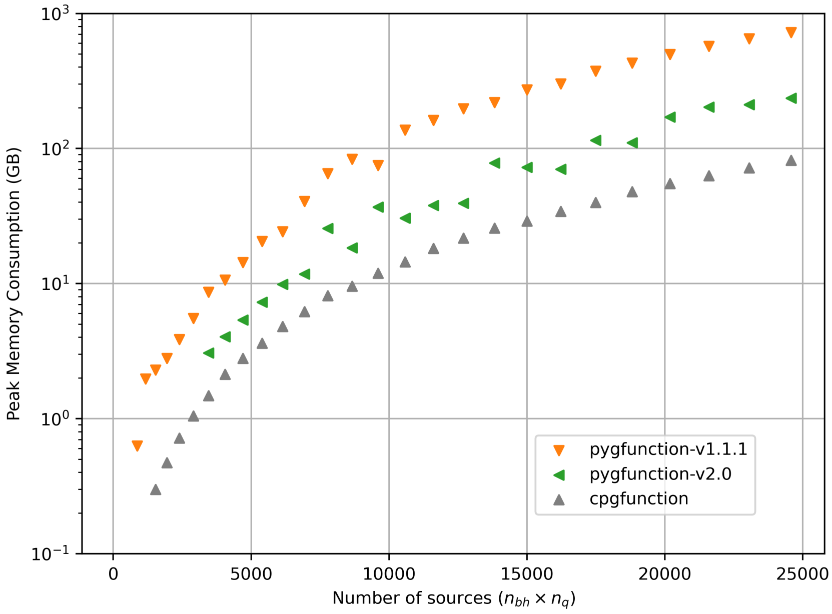


Figure 2.24: Memory comparisons for *pygfunction* v1.1.1, *pygfunction* v2.0 and *cpgfunction*.

2.5.2 *cpgfunctionEP*

The EnergyPlus development team requested that the cpgfunction project solely utilize C++ code. EnergyPlus is currently working to remove any remaining Fortran code, and the developers weren't keen on the author introducing Fortran libraries into the mix. *CpgfunctionEP* version 0.6 is a fork of *cpgfunction* version 2.0 and is available in the EnergyPlus version 9.6.0 release. The differences between the programs are described in this section.

The first modification to *cpgfunctionEP* was the removal of the OpenBLAS dependency. There were two optimized linear algebra Fortran functions in OpenBLAS that allowed cpgfunction to fully utilize the central processing unit (CPU). One of the functions was the SPMV function of BLAS, and the other was a function that solved general equations (GESV) that performs LU-decomposition with partial pivoting. Both of these function calls are located in the "building and solving equations" section of the code. NumPy and SciPy utilize the OpenBLAS library, and the author believes it to be the fastest CPU based linear algebra library. The differences in the speed of each program based on the calculation step and the ratio quantifying how many times faster cpgfunction version 2.1 (the most recent release) is than cpgfunctionEP is given in Table 2.11. The performance is captured on an 11th Generation Intel Core i9-11900K at 3.50GHz, and all 16 logical threads are used for multithreading. The g-function is calculated for a borehole field with a 12x13 configuration with 12 equal vertical segments. Calculating segment-to-segment response factors in *cpgfunctionEP* is done with the integration approximation of Cimmino (2021), while *cpgfunction* utilizes the Gauss-Kronrod quadrature integration implemented by Munoz (2014). There appears to be a slight increase in speed by utilizing the approximation.

The steps of computation for building and solving the systems of equations is shown in Table 2.12. The 3 times decrease in speed of the temporal superposition is a result of the replacement of the SPMV function with a hand rolled C++ implementation that contains multi-threading via OpenMP. The decrease in performance by deprecating the call to the GESV function is more than that of the SPMV function. For this case, the result is a 73 times decrease in speed. The LU decomposition utilized in cpgfunctionEP is currently Crout's method with implicit pivoting, inspired by Press et al. (2002). The Eigen library was planned to be utilized as a replacement of the OpenBLAS linear algebra, but segment fault errors occurred when building on Linux Ubuntu when utilizing Eigen's LU decomposition. A thorough analysis of the inputs and outputs of the Eigen function that caused the segment fault was performed. The conclusion was that the problem was inside of Eigen. The Eigen library is not currently a dependency of *cpgfunctionEP*. The reason for the segment fault was not identified.

There are more computationally savvy methods than Crout's method, with implicit pivoting, currently utilized in *cpgfunctionEP*. There was a deadline to get *cpg-*

functionEP ready for the release of EnergyPlus version 9.6.0. The next improvements to *cpgfunctionEP* should be the development or utilization of a pure C++ function that competes in speed performance with the OpenBLAS GESV function. The calculation of segment-to-segment responses could also be vectorized via OpenMP's single input multiple data (SIMD) instruction set. With these two improvements, *cpgfunctionEP* might have a chance at competing with the performance of *pygfunction*.

Table 2.11: Summary of timings for a 12x13 field with 12 segments based on the calculation step in cpgfunctionEP version 0.6 and cpgfunction version 2.1.

Step	cpgfunctionEP-v0.6 time (s)	cpgfunction-v2.1 time (s)	Ratio
Identifying similarities	2.13	2.05	1.04
Calculating segment-to-segment response factors	0.92	1.26	0.73
Building and solving system of equations	341.30	15.50	22.02

Table 2.12: Section breakdown of the building and solving systems of equations for 27 time steps and a 12x13 borehole field.

Step	cpgfunctionEP (v0.6)	cpgfunction (v2.0)	Ratio
Time to build matrices	4.44	5.00	0.89
Load history reconstruction	0.07	0.07	0.94
Temporal super position	7.65	2.48	3.08
LU Decomposition	325.00	4.44	73.26

2.6 Conclusions

At its initial development, *cpgfunction* version 1.0 was a great improvement for the purposes of generating a library of g-functions on the OSU HPCC. *Cpgfunction*, in combination with an adaptive discretization scheme, was able to generate a new expanded library more than 34 times the size of Hellström's proprietary library. Later developments to the adaptive discretization scheme led to the idea of unequal segment lengths, which are proven to be both more accurate and faster with fewer segments, as shown in Chapter IV Borehole Configuration Selection in the GHEDT. *Cpgfunction* was forked and a pure C++ version was implemented into EnergyPlus, named *cpgfunctionEP*. It is likely that *cpgfunctionEP* will not be replaced anytime soon, though it may be further developed to increase the speed of the calculation. *cpgfunctionEP* does not perform as fast as *cpgfunction* due to lack of external optimized vectorized code.

Given that *cpgfunction* is written in a compiled language, it may be easier for GHE sizing tools not written in Python to utilize the calculation. It is currently unclear if an executable could be created from *pygfunction*. *Cpgfunction*, at its current speed, could be suitable for use in sizing tools for up to several hundred boreholes. Additionally, *cpgfunction* may be further developed to utilize the equivalent borehole method of Prieto and Cimmino (2021). See Chapter IV Borehole Configuration Selection in the GHEDT for a discussion of the equivalent borehole method.

The status of *cpgfunction* is that it currently only compiles on Linux. The reason for that is the OpenBLAS library does not appear to utilize CMake yet. Though, the library does claim to provide pre-compiled binaries for Windows, so a developer familiar with Windows and C++ should be able to quickly compile a version on Windows. *cpgfunctionEP* is not currently recommended for use in sizing tools for it is slow when compared to *cpgfunction*. Though, *cpgfunctionEP* does compile on all three operating systems (Mac, Linux, Windows). *Cpgfunction* and *cpgfunctionEP* are both open-source programs and are available at `https://github.com/j-c-cook`. Also, *cpgfunctionEP* is available in the EnergyPlus repository `https://github.com/NREL/EnergyPlus/tree/develop/third_party/cpgfunctionEP-0.6`.

CHAPTER III

Sizing and Simulation in the GHEDT

3.1 Introduction

The ground is a widely accessible medium with large thermal capacity and can be used as thermal storage for heating and cooling applications. For larger commercial and institutional buildings, the most common technique to extract or reject heat to the ground is via a vertical borehole heat exchanger (BHE). In North America, a vertical BHE typically contains a single U-tube made of high-density polyethylene and is backfilled with grout. In North American applications, the borehole is commonly 50-125 m in depth, but shallower and deeper boreholes are sometimes used. A vertical borehole ground heat exchanger (GHE) can consist of one or many BHEs that receive fluid in a (usually) parallel piping arrangement with the header piping buried 1-1.5 m below the ground surface. The circulating fluid or heat carrier fluid exits the boreholes and is supplied to one or more water source heat pumps (WSHPs), which then rejects heat to or extracts heat from the circulating fluid, depending on whether the WSHP is in heating mode or cooling mode. The design of the BHE plays a significant role in a GHEs performance in the short-term (ranging from 5 minutes to 2 days). Over longer periods of time (days to decades) the ground properties and arrangement of boreholes becomes increasingly important for the performance. Understanding the short and long-term performance of a GHE prior to installation is crucial. A GHE designed too small can suffer from long-term temperature build-up or draw-down, causing poor system performance or failure. A GHE that is too large could exacerbate an already expensive initial installation cost, resulting in a waste of money and resources, or cause a lower first-cost system being chosen. This chapter presents a Python package named *Ground Heat Exchanger Design Toolbox* (GHEDT), which provides the tools necessary to quantify the short and long-term thermal interaction in a GHE. GHEDT can predict heat pump entering fluid temperatures over the system design life. It can determine the required borehole depth of a specified borehole field to ensure the heat pump entering fluid temperature remains within specified bounds. GHEDT is novel in its ability to select a borehole field layout based on a target depth, though this chapter focuses on the simulation and sizing methodologies.

3.2 Background

The GHEDT contains a g-function based simulation procedure. The g-function, originally developed by Prof. Johan Claesson and his PhD student, Per Eskilson (1987), quantifies the average borehole wall temperature, $\bar{T}_b$ due to a constant total but time varying heat rejection rate applied over the length of the borehole wall $\dot{Q}_b'$ (Cimmino, 2019).

$$\Delta\bar{T}_b = \frac{1}{2\pi k_s}\int_0^t \frac{d\dot{Q}_b'(\tau)}{dt} g(t-\tau)d\tau \tag{3.1}$$

The g-functions are an intermediate solution to a pure heat conduction model that is derived from Fourier's differential equation. The long-term thermal buildup is accounted for by integrating the point source solution (Ingersoll et al., 1948) over the length of the borehole and superimposing the response of all boreholes (Claesson and Javed, 2011). The g-function is dependent on dimensionless time and dimensionless geometric ratios (Claesson and Eskilson, 1988). The equation describing the use of the g-function to find the average borehole wall temperature is simplified by applying a constant heat rejection rate.

$$\bar{T}_b = T_g + \frac{\dot{Q}_b}{2\pi k_s H} g\left(\frac{t}{t_s}, \frac{B}{H}, \frac{r_b}{H}, \frac{D}{H}\right) \tag{3.2}$$

Where:

T_g is the undisturbed ground temperature,

k_s is the soil/ground thermal conductivity,

t is the time,

t_s is the time scale, $H^2/(9\alpha_s)$, and t/t_s is the dimensionless time,

H is the borehole "height" - that is the depth beyond the depth of the horizontal piping,

B is the borehole spacing, and B/H is the dimensionless spacing,

r_b is the borehole radius and r_b/H is the dimensionless borehole radius, and

D is the burial depth (depth of the horizontal piping) and D/H is the dimensionless burial depth.

Historically, existing design tools like *Ground Loop Heat Exchanger Professional* (GLHEPRO) (Marshall and Spitler, 1994; Manickam et al., 1996; Spitler, 2000; Cullin, 2008; Grundmann, 2016) and *Earth Energy Designer* (Hellström and Sanner, 1994) have made use of a library of precomputed long-term g-functions. The time to compute g-functions can be quite large, and therefore, a precomputed library has been preferred in previously developed GHE sizing tools. The g-function library was computed by Hellström, a colleague of Eskilson. Prior to the developments described in Chapter IV, an enhanced library was computed for use in the GHEDT (Spitler et al., 2021). GHEDT contains methods for interpolating g-functions based on the dimensionless borehole height, B/H and can correct g-functions for the dimensionless borehole radius, r_b/H. The GHEDT can interpolate g-functions computed by the open source g-function calculation toolbox, *pygfunction* (Cimmino, 2018b), but in this chapter Hellström's g-functions are used so that the simulation of GHEDT can be validated against GLHEPRO.

The long-time step accounts for the long-term heat buildup but does not account for short-term thermal processes inside of the borehole. A system can be dominated either by short term peak loads or long-term loads as discussed by Young (2004). The short-term thermal processes were initially represented as being quasi-steady state, using only a borehole thermal resistance. Yavuzturk and Spitler (1999) developed a dynamic 2D (radial-angular) model of the borehole to extend the g-functions to short time steps. Xu and Spitler (2006) simplified this to a one-dimensional radial geometry - a cylinder of fluid, surrounded by a single pipe, surrounded by grout. Xu and Spitler showed this approach worked very well if the parameters of the one-dimensional radial model conserve the thermal capacity of each element (water[1] , pipe, grout), the effective borehole thermal resistance, and the thermal resistance between the fluid and the grout. Xu and Spitler formulated this for a single U-tube, but it can be readily extended to double U-tubes and co-axial heat exchangers by computing equivalent single U-tube BHEs. This extension is described in this document, and is implemented with compatibility to EnergyPlus in mind, so that the resulting systems defined by GHEDT can be used in a whole building simulation for predicting energy consumption. The one-dimensional redial numerical model implemented by Mitchell (2019) was pulled from his Ground Loop Heat Exchanger (GLHE) code base, edited to accept objects that are children of (inherit from) *pygfunction* pipe objects and vectorized to increase the calculation speed (by about 22x). *Pygfunction* provides the effective borehole thermal resistance calculation utilized in GHEDT. *Pygfunction*

[1] The working fluid is referred to as "water" here, though in practice it may be a water-antifreeze mixture.

had existing single and multiple U-tube models, and a coaxial pipe model has been developed and committed to *pygfunction* (in collaboration with Cimmino). Borehole thermal resistance is discussed in detail in the next section.

GHEDT can perform both an hourly simulation that considers the load on the ground at each hour and a hybrid time step simulation that greatly decreases the computation time to predict the evolution of the heat pump entering fluid temperature. An enhanced and automated hybrid time step procedure that extends the work of Cullin and Spitler (2011) is presented.

The necessary inputs to describe a GHE in GHEDT are:

- Hourly rejection/extraction loads on the ground for 1 year duration – these are usually based on simulation using EnergyPlus or a similar whole-building energy simulation (WBES) program. The WBES program predicts the hourly heating and cooling loads and estimates the resulting heat rejection/extraction with a model of the WSHP.

- Soil thermal properties – this consists of the thermal conductivity, the volumetric heat capacity of the soil and the undisturbed ground temperature. For projects in the final design stage, these values are determined with a thermal response test (Spitler and Gehlin, 2015).

- Fluid properties – this consists of the thermal properties of the fluid and the system flow rate. The thermal properties can be computed by an open source, state of the art, Python fluid property calculation package, CoolProp (Bell et al., 2014). The system flow rate is determined in conjunction with the heat pump selection.

- Pipe and grout thermal properties – these properties are typically provided by the manufacturers.

- Configuration of boreholes – this consists of the x,y coordinates of the boreholes, as well as the depth.

The remainder of this document describes the simulation and sizing procedures of GHEDT. The next section describes computation of borehole resistance for single U-tube, double U-tube, and coaxial tube BHEs. The computation of the effective borehole thermal resistance is validated by comparing to GLHEPRO. Determining equivalent single U-tube borehole heat exchanger geometries is then discussed. The short time step g-function response for each geometry is then compared to GLHEPRO. The hourly and hybrid time step simulation procedures are discussed and compared to one another. Then the sizing results of the three BHEs are compared to GLHEPRO for a 12x13 configuration with 5m equal spacing, in an attempt to validate the simulation procedure of GHEDT.

3.3 Borehole Resistance

3.3.1 Theory

The performance of the borehole heat exchanger often involves representation as a thermal network, which has:

- Local borehole thermal resistance, R_b: at any depth, the resistance to heat transfer between the fluid and the borehole wall.

- Local borehole internal resistance, R_a: at any depth, the resistance to heat transfer between the upward and downward flowing fluid streams.

- Effective borehole thermal resistance, R_b^*: approximating the average borehole fluid temperature as the simple mean of the entering and exiting fluid temperatures, the effective borehole thermal resistance is the resistance to heat transfer between the simple mean fluid temperature and the borehole wall temperature for the entire borehole.

The calculation of these resistances is described in some detail by Javed and Spitler (2016, 2017). Use of the effective borehole thermal resistance allows a conceptually simple model of the borehole. It can be treated as a single pipe with fluid entering at temperature, T_{in}, and exiting at temperature, T_{out} where the pipe is rejecting heat to its surroundings at a constant rate, as shown in Figure 3.1 (left).

$$T_{out} = T_{in} + \frac{\dot{Q}_b}{\dot{m}c_p} \tag{3.3}$$

The equivalent resistance network is given in Figure 3.1 (right). Considering temperature to be analogous to voltage, and heat flow analogous to current, the governing equation for the average borehole wall temperature, $\bar{T_b}$ can be written:

$$\bar{T_b} = \bar{T_f} - \frac{\dot{Q}_b}{L} R_b^* \tag{3.4}$$

Where the average fluid temperature $\bar{T_f}$ refers to the simple average of the inlet and the outlet temperatures. In practice, constant ground heat rejection/extraction loads are not present, but the solution to the actual loads over time is found by superimposing step-functions representing the changing loads. See Section 3.6 Ground Heat Exchanger Simulation and Spitler and Bernier (2016) for further discussion.

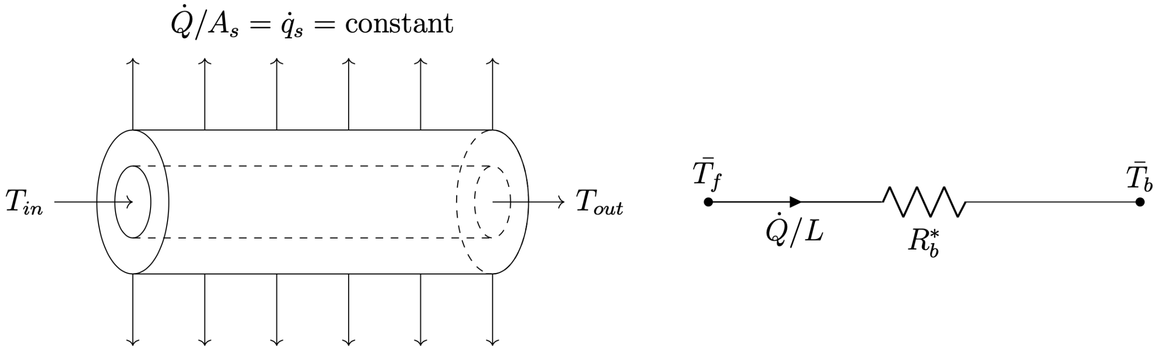


Figure 3.1: An equivalent borehole heat exchanger pipe with constant surface heat flux (left), and an associated effective borehole resistance network (right).

A single U-tube borehole heat exchanger has fluid at an inlet temperature, T_{f1}, flowing downwards and an outlet temperature, T_{f2} flowing upwards. The tubes each reject or extract heat to their surrounding grout via convection heat transfer at the tube wall and conduction heat transfer through the pipe wall over the length of the pipe, giving a heat transfer rate of $\dot{Q}/L$. The ability of each pipe to transfer heat to the borehole wall is given by the local borehole thermal resistance, R_b. The most accurate technique for computing the local borehole resistance of any arbitrarily positioned pipe in a borehole is the multipole method (Claesson and Hellström, 2011). For calculating the effective borehole thermal resistance, a Δ-resistance network (Figure 3.2) is often used as an approximation for the heat flow in the borehole. In this case, the local internal thermal resistance, R_a, is found by considering the network between the two pipes (Javed and Spitler, 2016) as shown in Equation 3.5. Then, the internal borehole energy balance for Figure 3.2 is given by Equation 3.6 and Equation 3.7, where the convention is that z is positive going downwards.

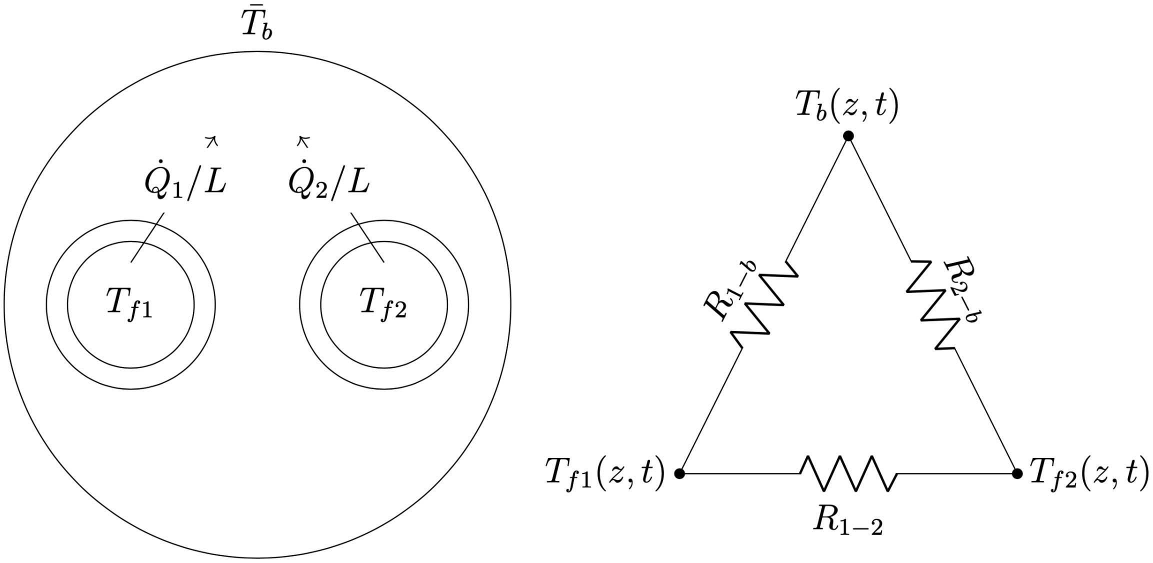


Figure 3.2: Top view of a single U-tube borehole heat exchanger with its corresponding delta circuit.

$$R_a = \left[\frac{1}{R_{1-2}} + \frac{1}{R_{1-b} + R_{2-b}}\right]^{-1} \tag{3.5}$$

$$-\dot{m}c_f\frac{dT_{f1}}{dz}=\frac{\dot{Q}_1}{L}(z)=\frac{T_{f1}-T_b}{R_{1-b}}+\frac{T_{f1}-T_{f2}}{R_{1-2}} \tag{3.6}$$

$$\dot{m}c_f\frac{dT_{f2}}{dz}=\frac{\dot{Q}_2}{L}(z)=\frac{T_{f2}-T_b}{R_{2-b}}+\frac{T_{f2}-T_{f1}}{R_{1-2}} \tag{3.7}$$

Cimmino (2019) uses the multipole solution for any n number of tubes arbitrarily positioned to determine the delta-circuit thermal resistances and presents a method adapted from Hellström (1991); Claesson and Hellström (2011) to calculate the effective borehole thermal resistance in *pygfunction*.

3.3.2 Development

Pygfunction (Cimmino, 2018b) is utilized for computing effective borehole thermal resistances and is enhanced[2] where need be. The BHE objects of GHEDT are children of (inherit from) parent objects that are defined in *pygfunction*. Cimmino developed effective borehole resistance calculations for use in computing g-functions, but this feature is also used in GHE simulation, where the effective borehole resistance plays an important role in the performance of a system. The GHEDT BHE objects compute intermediate parameters, such as pipe thermal resistance and fluid convection coefficient, upon initialization. Objects are defined that hold the fluid, soil, grout, and pipe thermal properties to reduce the number of required parameters for input to other functions and objects. Variations of these improvements to the interface to *pygfunction* could and will likely work their way into the program itself.

Single and multiple U-tube models were available in pygfunction (Cimmino, 2015, 2016, 2019). The modularity of the implementation allowing multiple objects able to interface with a single g-function calculation (polymorphism) is a desirable trait for borehole heat exchanger models. Polymorphism is a desired characteristic of the BHE models so that the objects can be passed into reusable functions to compute equivalent

[2] The author has collaborated with Cimmino to enhance pygfunction for use in GHE design and has contributed to pygfunction.

single U-tube boreholes. A concentric (otherwise referred to as coaxial) borehole heat exchanger model has been added to pygfunction for use in UIFT g-function calculations, and borehole thermal resistance calculations can now be computed as well. Figure 3.3 shows a pie section of a concentric borehole heat exchanger with the resistance network. A fluid at an inlet temperature, $T_{f,1}$ enters the top of the pipe in the center, flows to the bottom, and returns in the annulus as $T_{f,2}$. $R_{cyl,3}$ represents the grout; $R_{cyl,2}$ represents the outer pipe, and $R_{cyl,1}$ represents the inner pipe.

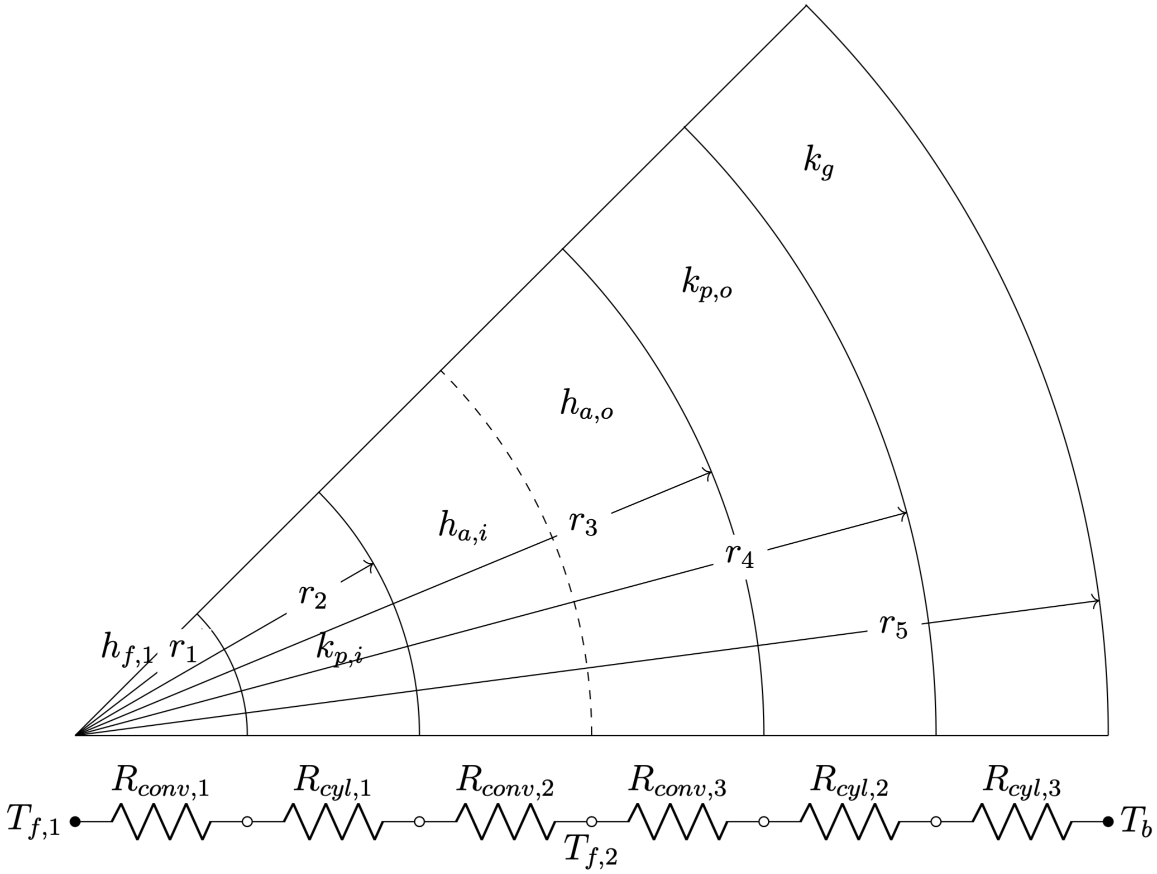


Figure 3.3: Resistance network for concentric pipe in borehole.

The concentric model has a downward flow and an upward flow and Hellström's (1991) expressions, shown in Figure 3.1, can be used for calculating effective borehole thermal resistance. With the simpler co-axial network, the internal thermal resistance can be defined as the sum of resistances between the two fluids.

$$R_a = R_{1-2} = R_{conv,1} + R_{cyl,1} + R_{conv,2} \tag{3.8}$$

The thermal interaction between the fluid in the annulus and the borehole wall is defined as the sum of the resistances between the fluid temperature, $T_{f,2}$ and the borehole wall, T_b.

$$R_{2-b} = R_{conv,3} + R_{cyl,2} + R_{cyl,3} \tag{3.9}$$

The resistance between the inner fluid, $T_{f,1}$ and the borehole wall T_b is set to infinity because there is no direct connection between the temperatures.

$$R_{1-b} = \infty \tag{3.10}$$

The conduction resistances in cylinders, R_{cyl} is a direct calculation given the inner, r_i and outer, r_o radii, and the thermal conductivity, k_p.

$$R_{cyl} = \frac{ln(r_o/r_i)}{2\pi k_p} \tag{3.11}$$

Computing the convection resistances, R_{conv} requires knowledge of the convection coefficient. The Nusselt number is required to determine the convection coefficient. The first step in computing the Nusselt number is calculation of the Reynolds number.

$$Re = \frac{4\dot{m}}{\mu \pi D_h} \begin{cases} \text{laminar} & Re \leq 2300 \\ \text{critical} & 2300 < Re \leq 4000 \\ \text{turbulent} & Re > 4000 \end{cases} \tag{3.12}$$

Where $\dot{m}$ is the mass flow rate, μ is the dynamic viscosity of the fluid and D_h is the hydraulic diameter of the pipe. The hydraulic diameter is a ratio of the cross-sectional area to the wetted perimeter, $D_h = 4A_c/p$. The hydraulic diameter is used for the annular region; for the inner pipe, the hydraulic diameter simplifies to the inner diameter.

Circular tubes: $$D_h = \frac{4A_c}{p} = \frac{4\pi D^2/4}{\pi D} = D$$

The hydraulic diameter of the annulus region, given the outer diameter, D_o and inner diameter, D_i is given by:

Concentric tubes: $$D_h = \frac{4A_c}{p} = \frac{4\pi(D_o^2 - D_i^2)/4}{\pi(D_o + D_i)} = D_o - D_i = 2(r_o - r_i)$$

The (Gnielinski, 1976) correlation is used for computing the Nusselt number for turbulent flow (Çengel and Ghajar, 2015). The Nusselt number in laminar flow for a constant surface temperature boundary condition inside a circular tube is reduced to $Nu = 3.66$. The Darcy-Weisbach friction factor is a necessary input for determining the Nusselt number. The Colebrook-White equation is used for turbulent flow, and the laminar flow friction factor reduces to $f = 64/Re$ (Çengel and Ghajar, 2015).

In determining the Nusselt number, a one-dimensional interpolation function is used in the critical region to ensure there are no dramatic jumps in determination of the Nusselt number during design (Gnielinski, 2013). Hellström (1991) gave correlations for the Nusselt number while in laminar flow in the inner annulus, $Nu_{a,i}$ and the outer annulus, $Nu_{a,o}$ regions of a concentric tube heat exchanger.

$$Nu_{a,i} = 3.66 + 1.2\left(\frac{r_i}{r_o}\right)^{-0.8} \tag{3.13}$$

$$Nu_{a,o} = 3.66 + 1.2\left(\frac{r_i}{r_o}\right)^{0.5} \tag{3.14}$$

The inner and outer Nusselt numbers in the annulus region are equivalent for turbulent flow.

After the Nusselt number in a region is solved for, the convection coefficient of the fluid, h_f, given the thermal conductivity of the fluid, k_f can then be found.

$$h_f = \frac{k_f Nu}{D_h} \tag{3.15}$$

From there, the convection resistances can be solved for.

$$R_{conv} = \frac{1}{h_f 2\pi r_i} \tag{3.16}$$

3.3.3 Borehole Resistance Validation

Given the existing borehole heat exchanger models of *pygfunction*, the single U-tube and multiple U-tubes, and the added coaxial model, code can readily be written to compute the borehole resistances. *Pygfunction*'s borehole heat exchanger models are wrapped to provide a new application programming interface (API) that reduces the number of inputs required to call the models and update the resistances during design when the flow rate to each borehole or height of the boreholes is modified. The API defines the way a function is called, and GHEDT simplifies the API for BHEs. It is important to ensure that the BHE models of GHEDT don't introduce any errors in the effective borehole resistance calculation. To verify the implementation, the effective borehole resistance calculations for the three borehole heat exchanger models are compared to GLHEPRO. A series of test cases are defined and described as follows.

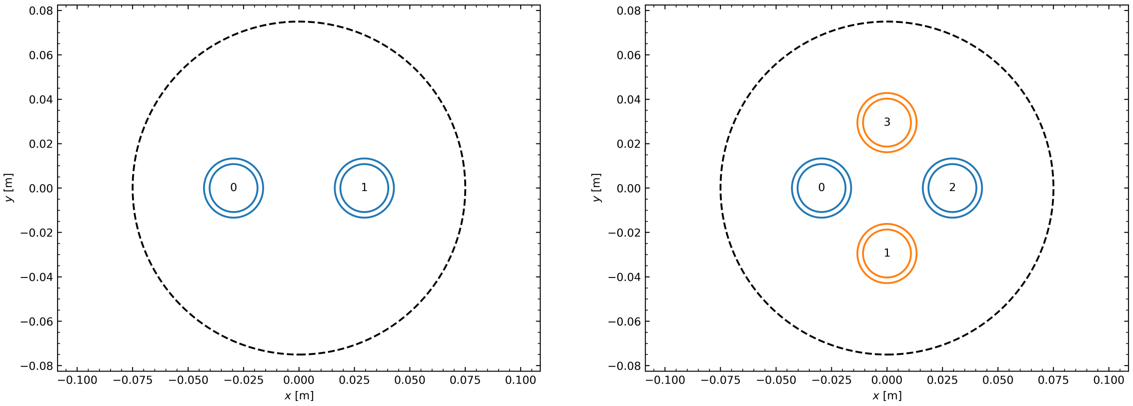
The test case inputs that do not pertain to geometry in the borehole heat exchangers are given in Table 3.1. The single and double U-tube borehole heat exchanger geometry is summarized in Table 3.2 and shown (top view) in Figure 3.4. The two U-tubes are different colors for the double U-tube shown in Figure 3.4b. The coaxial borehole heat exchanger geometry is defined in Table 3.3 and the top view is shown in Figure 3.5.

Table 3.1: Non geometric parameters necessary for computing borehole resistances

Volumetric flow rate per borehole, $\dot{V}_b$ (L/s)	0.2
Height of boreholes, H (m)	100
Ground thermal conductivity, k_s (W/(m·K))	2
Grout thermal conductivity, k_g (W/(m·K))	1
Pipe thermal conductivity, k_p (W/(m·K))	0.4
Water temperature, T (°C)	20
Pipe roughness, ϵ (m)	1.00E-06

Table 3.2: Single and double U-tube borehole geometry

Borehole Diameter, D_b (mm)	150
U-tube Inside Diameter, $D_{p,i}$ (mm)	21.6
U-tube Outside Diameter, $D_{p,o}$ (mm)	26.6
Shank Spacing[3], s (mm)	32.3



(a) Single U-tube

(b) Double U-tube

Figure 3.4: Top views of U-tube borehole heat exchangers, based on geometrical inputs given in Table 3.2.

Table 3.3: Coaxial borehole geometry

Borehole Diameter, D_b (mm)	175
Inner Tube Inside Diameter, $D_{p,i}$ (mm)	44.2
Inner Tube Outside Diameter, $D_{p,a,i}$ (mm)	50
Outer Tube Inside Diameter, $D_{p,a,o}$ (mm)	97.4
Outer Tube Outside Diameter, $D_{p,o}$ (mm)	110

[3] "Shank spacing" is the distance between the two legs of the U-tube measured from the outer tube walls. For a double U-tube, it is the distance between the two opposite tubes, again measured from the outer tube walls. The shank spacing is considered to be constant over the height of the borehole.

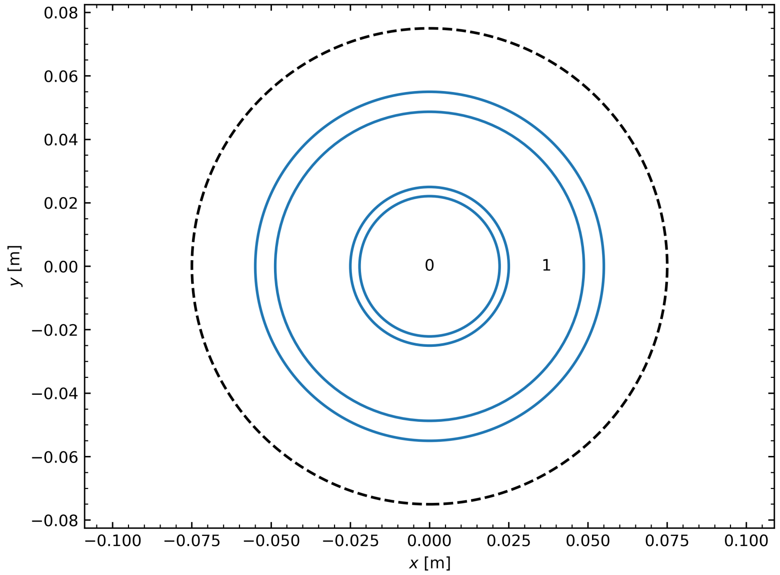


Figure 3.5: Top view of a concentric borehole heat exchanger, based on the geometrical inputs given in Table 3.3.

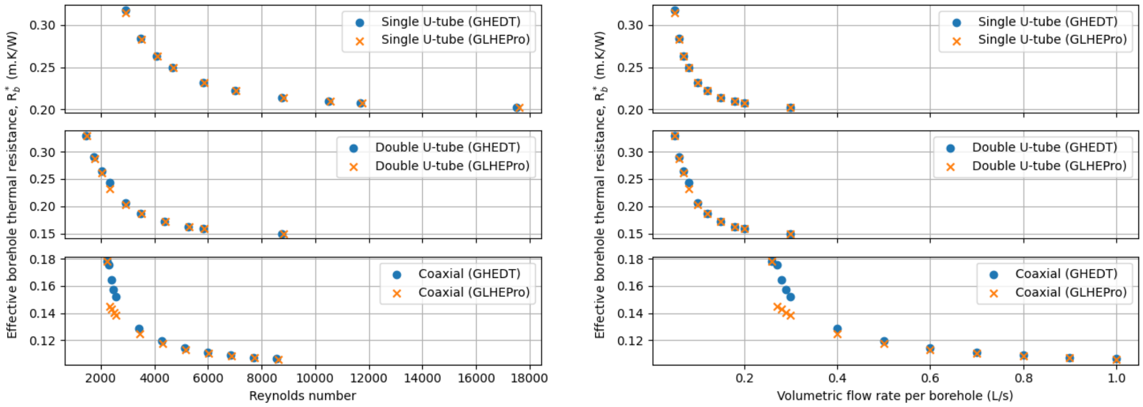
Given the borehole heat exchanger inputs described in Table 3.1, Table 3.2 and Table 3.3 for single U-tube, multiple U-tubes and coaxial models, the intermediate values are presented in Table 3.4 to help with reproducibility: Reynolds number, pipe resistance, convection coefficient and convective resistance. For the coaxial heat exchanger, the Reynolds number is for the annulus; the convection coefficient and resistance apply to the inside of the outer pipe wall, and the pipe resistance is given for the outer pipe. The resulting effective borehole resistance given the borehole volumetric flow rate described in Table 3.1 is also provided in Table 3.4.

Table 3.4: Effective borehole thermal resistance values and intermediate values to the calculation

Field	Single U-tube	Double U-tube (parallel)	Coaxial
Reynolds number	11667	5833	2291[4]
Pipe resistance (K/(W/m))	0.083	0.083	0.083
Convection coefficient (W/m2·K)	2522	1292	57.1
Convective resistance (K/(W/m))	0.0897	0.0952	0.106
Effective borehole resistance (K/(W/m))	0.2073	0.1597	0.1929

[4] The coaxial Reynolds number given is for the outer annulus region.

The effective borehole thermal resistances of GHEDT are now compared to GLHEPro over a range of flow rates. Figure 3.6a presents the effective borehole thermal resistances over a range of Reynolds numbers for the three borehole heat exchangers described above. The only input value that is varied to provide the range of Reynolds numbers is the volumetric flow rate per borehole. The flow rates are provided in Figure 3.6b. As can be seen in the Figure 3.6, the effective borehole thermal resistances calculated by GHEDT match those calculated by GLHEPRO quite well, with some discrepancy in the transition region, due to minor differences in how the transition region is modeled. The transition region is $2300 < Re < 4000$, and the coaxial tube has the largest discrepancies Figure 3.6b.



(a)

(b)

Figure 3.6: The effective borehole thermal resistances versus Reynolds number (a) and versus volumetric flow rate per borehole (b) for single U-tube, multiple U-tube and concentric borehole heat exchangers

3.4 Generation of Equivalent Single U-tube Representations

3.4.1 Background

It is desirable for GHEDT to be compatible with EnergyPlus. That is, a simulation of a specific GHE in GHEDT will result in the same heat pump exiting fluid temperatures as in EnergyPlus. EnergyPlus has three options for calculation of g-functions: uniform heat flux boundary condition, the uniform borehole wall temperature boundary condition of Cook and Spitler (2021) coupled with an adaptive discretization scheme, or the g-function can be input rather than computed. Therefore, the g-function described in a EnergyPlus input file is an optional output of GHEDT. However, EnergyPlus must still compute the effective borehole thermal resistance for the simulation. Currently, the only BHE model available for input to EnergyPlus is the single U-tube. Therefore, with compatibility to EnergyPlus in mind, GHEDT goes beyond determining the equivalent parameters for the one-dimensional radial model. Instead, GHEDT determines the equivalent parameters for a single U-tube that can be input into EnergyPlus.

For compatibility with the current EnergyPlus version, multiple U-tube and coaxial BHE models are transformed to an equivalent single U-tube BHE and can be determined for most cases[5]. Therefore, when a multiple U-tube or coaxial BHE is described for simulation and sizing in GHEDT, an equivalent single U-tube must be determined and output for EnergyPlus to properly simulate the design input provided by GHEDT. This is done by preserving the volume of the fluid and pipe, the sum of the convection and pipe resistance and the total effective borehole thermal resistance. The equivalent single U-tube parameters can all be readily determined algebraically, except for finding the equivalent pipe conductivity and thermal conductivity of the grout. Brent's method, a root finding procedure implemented in SciPy (Virtanen et al., 2020) is twice employed for finding the equivalent pipe thermal conductivity and the equivalent grout conductivity. The pipe conductivity is varied such that the total pipe resistance is made equal, and then the effective borehole thermal resistance is set equal.

[5] While researching this, it was found that there are valid co-axial configurations that cannot be rendered as an equivalent single U-tube and still conserve the volume of water while maintaining the borehole diameter. This is an inherent limitation in this approach. To solve this problem, the borehole diameter is increased when necessary. In the longer term, EnergyPlus will need to treat double U-tube and co-axial configurations.

The volume and resistance calculations for multiple U-tube and coaxial BHEs are described in the following two sections. The pipe conductivity can be varied such that the summation of the pipe and convection resistances are equal for both the original and equivalent geometries. The grout thermal conductivity can then be varied to set the effective borehole thermal resistance values equal. In the case that the equivalent geometry does not fit inside of the borehole, the borehole radius is enlarged to fit the two pipes with reasonable spacing in the borehole.

3.4.2 Multiple U-Tubes

In this section, n represents the number of pipes, where a double U-tube contains 4 pipes. The first step, calculation of the equivalent single U-tube requires computation of surface area at the fluid-pipe interface, gives the surface area (in m^2/m):

$$A_s = n\pi D_{p,i}^2 \tag{3.17}$$

Then, using the convection coefficient at that surface, the convective resistance for the double U-tube is:

$$R_{conv} = \frac{1}{hA_s} \tag{3.18}$$

The volume of the fluid, pipe and grout are found by considering a 1 m length section of the pipe (volumes here are in units of m^3/m):

$$V_{fluid} = n\pi\left(\frac{D_{p,i}}{2}\right)^2 \tag{3.19}$$

$$V_{pipe} = n\pi\left(\frac{D_{p,o}}{2}\right)^2 - V_{fluid} \tag{3.20}$$

$$V_{grout} = n\pi\left(\frac{D_b}{2}\right)^2 - V_{pipe} - V_{fluid} \tag{3.21}$$

The resistance of the pipes can be found:

$$R_{pipe} = \frac{\left(\dfrac{D_o}{D_i}\right)}{n 2 \pi k_p} \tag{3.22}$$

These parameters will be applied to find the equivalent single U-tube description.

3.4.3 Concentric

The volumes to be considered for computing an equivalent single U-tube given a concentric BHE are provided below. The volume of the fluid and the pipe are required to be computed, the grout volume is unnecessary, but included. The grout volume doesn't need to be determined because the borehole diameter is already defined. The variables used in this section are based on Figure 3.3.

$$V_{fluid} = \pi \left[\left(\frac{D_1}{2}\right)^2 + \left(\frac{D_3}{2}\right)^2 - \left(\frac{D_1}{2}\right)^2 \right] \tag{3.23}$$

$$V_{pipe} = \pi \left[\left(\frac{D_2}{2}\right)^2 - \left(\frac{D_1}{2}\right)^2 + \left(\frac{D_4}{2}\right)^2 - \left(\frac{D_3}{2}\right)^2 \right] \tag{3.24}$$

$$V_{grout} = \pi \left[\left(\frac{D_5}{2}\right)^2 - \left(\frac{D_4}{2}\right)^2 \right] \tag{3.25}$$

The convection and outer pipe resistance thermally separate the water from the grout portion of the borehole. The inner pipe and fluid thermal resistances are included in the R_b^* calculation, but don't have an effect on the equivalent U-tube pipe and fluid thermal resistance. The Xu and Spitler (2006) model treats the fluid as a single lump. That is, it's all at one temperature with no internal thermal resistance. Therefore, the outer annulus convective resistance and the outer pipe resistance of the concentric heat exchanger determine the needed convective resistance and pipe resistance of the equivalent single U-tube, as follows.

$$R_{conv} = \frac{1}{h_{a,o} A_{s,3}} \tag{3.26}$$

$$R_{pipe} = \frac{\left(\frac{D_4}{D_3}\right)}{n 2\pi k_{p,o}} \tag{3.27}$$

3.4.4 Equivalent Single U-tube

After the total volumetric capacitance of the fluid and pipe and the resistances for convection and pipe have been found, an equivalent single U-tube can be solved for. The value n in this section is held constant at 2 because the parameters for an equivalent single u-tube are being found. Any equivalent values will be marked with an equivalent, eq, subscript.

The inner and outer equivalent single U-tube diameters are solved for.

$$D_{p,i,eq} = 2\sqrt{\frac{V_{fluid}}{n\pi}} \tag{3.28}$$

$$D_{p,o,eq} = 2\sqrt{\frac{(V_{fluid} + V_{pipe})}{n\pi}} \tag{3.29}$$

A horizontal shank spacing where the tubes are equal distance from each other and the walls is solved for. It could happen that the equivalent pipes do not fit in the borehole; this will only occur for the coaxial BHE. If the pipes do not fit within the existing borehole radius, the borehole radius is adjusted so that the pipes will fit within the borehole, plus an additional 1/10 borehole radius. The equivalent pipe thermal conductivity is solved for.

$$k_{p,eq} = \frac{ln(D_{p,o}/D_{p.i})}{2\pi(nR_{pipe})} \tag{3.30}$$

Now, an initial grout conductivity guess is used from the existing BHE, which completes the inputs and allows creation of a single U-tube BHE. A root finding procedure is then implemented twice. The first procedure's cost function is one that solves for the equivalent pipe thermal conductivity that drives the difference between the sum of the convection and pipe resistance between the BHEs to 0. The equivalent pipe thermal conductivity is varied. This is necessary to match the resistance between the fluid and the grout. The mass flow rate could be varied to ensure an equivalent convection coefficient, but the mass flow rate is a sensitive parameter that should not be varied just for the purposes of an equivalent U-tube.

Following the pipe conductivity being found, the second root finding procedure is implemented where the cost function is the difference between the effective borehole thermal resistance of the original and equivalent BHEs. The value of grout thermal conductivity that makes the cost function 0, or solves the root, ensure that the effective borehole thermal resistance of the equivalent BHE is equal to the original. The equivalent BHE grout conductivity is varied.

The multiple U-tube from Table 3.2 and the concentric heat exchanger from Table 3.3 have a single equivalent borehole heat exchanger computed. The intermediate volumetric and resistances as well as the final equivalent single U-tube results for the two borehole heat exchangers are given in Table 3.5. The equivalent single U-tubes are visualized in Figure 3.7.

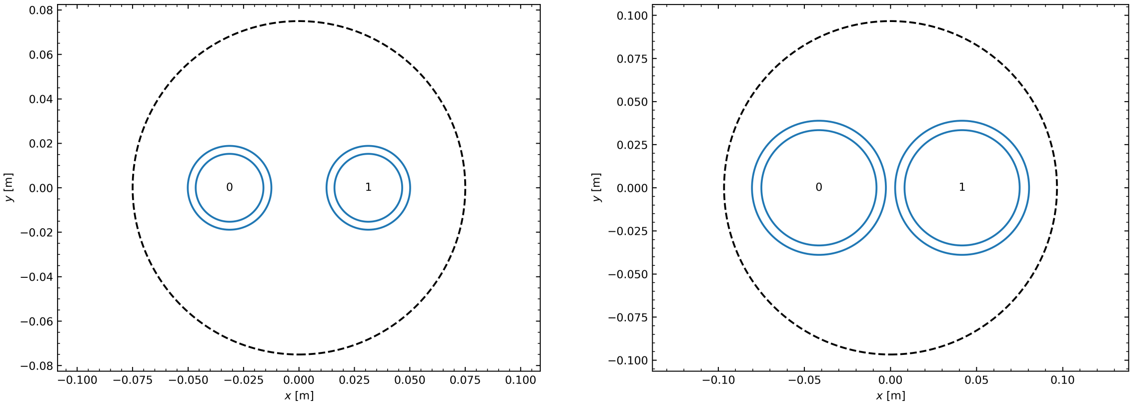
These equivalent BHEs must be computed any time the mass flow rate to each borehole or the height of the boreholes change. The volumetric flow rate for the system is usually set early during the design process. Because the boreholes are piped in parallel, the mass flow rate to each borehole is the system flow rate divided by the number of boreholes. The change in flow rate affects the local and effective borehole thermal resistances. The height is varied in the sizing process, affecting the effective borehole thermal resistance. The equivalent BHE is specific to the flow rate and height but will serve its purpose well in EnergyPlus because EnergyPlus is

Table 3.5: Table for intermediate results in solving equivalent single U-tube borehole heat exchanger

Equivalent parameters	Double U-tube	Coaxial
Fluid volume per meter, V_{fluid} (m3/m)	0.0015	0.007
Pipe volume per meter, V_{pipe} (m3/m)	0.0008	0.0025
Total Convective Resistance, R_{conv} (K/(W/m))	0.132	0.0095
Total Pipe Resistance, R_{pipe} (K/(W/m))	0.021	0.0484
Fluid volumetric flow rate, $\dot{V}_b$ (L/s)	0.2	0.2
Radius of inner pipe, $r_{p,i}$ (m)	0.0152	0.0334
Radius of outer pipe, $r_{p,o}$ (m)	0.0188	0.0389
Shank spacing, s (m)	0.0248	0.0054
Convection coefficient, h (W/m2.K)	1287	247.4
Pipe thermal conductivity, k_p (W/(m·K))	0.2317	0.249
Grout thermal conductivity, k_g (W/(m·K))	1.6966	0.522
Effective borehole resistance, R_b^* (m·K/W)	0.1597	0.1929

only doing a simulation. The single U-tube equivalent borehole is valid at a specific effective borehole thermal resistance. Therefore, in the design of a system in GHEDT, the equivalent single U-tube is computed many times. The calculation is fast enough to not be concerned with several calls throughout a sizing procedure.

The short-term response g-functions can now be solved for. The object accepts a single U-tube borehole heat exchanger object as an input – this is the equivalent single U-tube BHE in the case of double U-tubes and co-axial BHE.



(a) Double U-tube equivalent

(b) Coaxial tube equivalent

Figure 3.7: Equivalent single U-tube for double U-tube (a) and coaxial (b) (Note: the borehole radius grew for the coaxial heat exchanger).

3.5 Short Time Step

The short-term response of the borehole is largely due to the interaction inside of the borehole. The short-term response accounts for around the first 2 days of the response of a GHE due to a constant heat rejection load applied. A building that is dominated by short peak loads could be greatly undersized if only the long-term response is simulated.

The short-term radial numerical model of Xu and Spitler (2006) implemented in GHEDT was originally implemented by Mitchell (2019) and modified for code optimization. The short-term radial numerical model has been edited to accept a borehole heat exchanger object of GHEDT (which are children of pygfunction objects). The calculation is vectorized (by this author) by implementing the one-dimensional grid into NumPy (Harris et al., 2020) arrays, and the tridiagonal solver of an optimized Basic Linear Algebra Subprograms (BLAS) library (van de Geijn and Goto, 2011), OpenBLAS, is directly called via SciPy's (Virtanen et al., 2020) wrapper.

The short-term response g-functions for the three BHEs described in Table 3.1, Table 3.2 and Table 3.3 are plotted in Figure 3.8. The double U-tube and coaxial tube short-term response g-functions utilize the equivalent single U-tube BHEs found in the previous section. The minor differences between the implementations lead to minor differences in sizing results. This will be explored in the next section.

3.6 Ground Heat Exchanger Simulation

3.6.1 Overview

The purpose of the ground heat exchanger (GHE) simulation in GHEDT is primarily to determine the minimum and maximum heat pump entering fluid temperatures over the design life. For purposes of sizing, the design life is often taken to be 20-30 years. While the piping network in the ground will last much longer, the building and

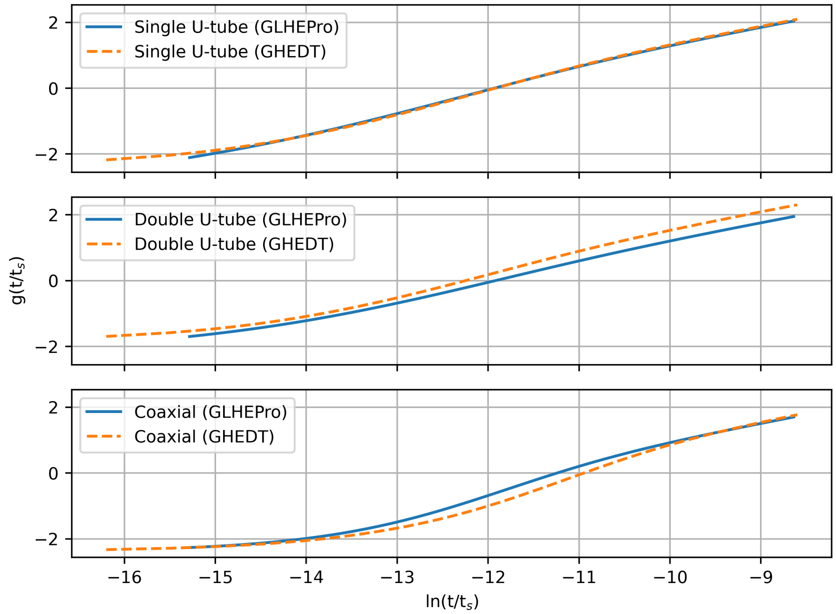


Figure 3.8: Short-term response g-functions for the single U-tube, double U-tube and coaxial tube borehole heat exchangers. The double U-tube and coaxial tube BHEs utilize the equivalent single U-tube calculations.

building loads are likely to change in the future. Finding the minimum and maximum heat pump entering fluid temperatures can be done with an hourly simulation, but in order to achieve adequate computational speed, a hybrid time step approach (Cullin and Spitler, 2011) is commonly used. Hourly heat rejection/extraction loads, like those shown in Figure 3.9, are necessary to simulate a GHE with an hourly time step. These are typically assumed to repeat every year over the design life. For the hybrid time-step approach, the hourly loads are represented with monthly values and monthly peak values.

3.6.2 Hourly Simulation

The heat pump entering fluid temperature, $T_{f,entering}$, can be found by performing an hourly simulation of the GHE. The first step in an hourly simulation is to determine the average borehole wall temperature, $\bar{T}_b$ at an hour, n. This is accomplished by superimposing the g-function with a time varying heat rejection rate, $\dot{Q}$ applied to a

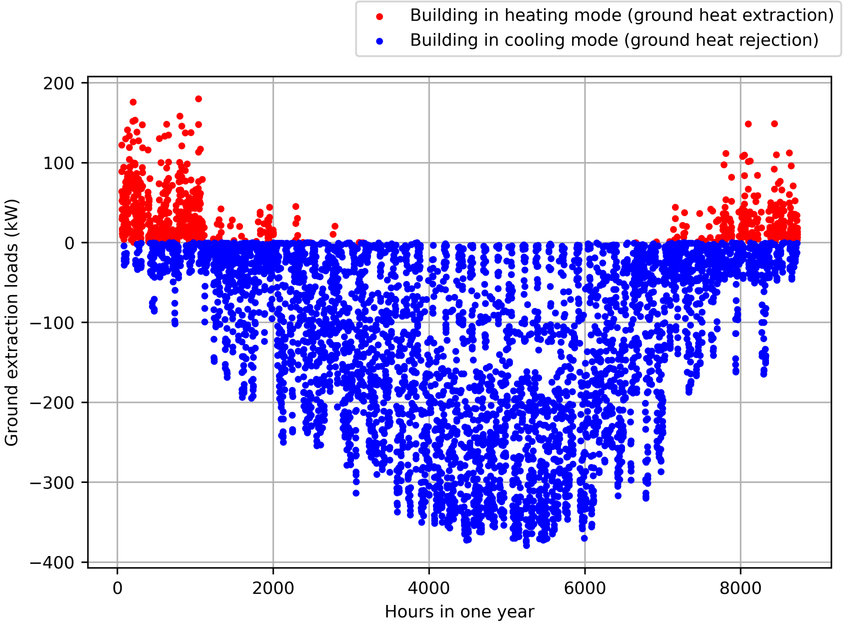


Figure 3.9: Atlanta office building one-year hourly ground extraction load profile.

borefield with n_{bh} number of boreholes.

$$\bar{T}_{b,n} = T_g + \sum_{i=1}^{n} \frac{\dot{Q}_i - \dot{Q}_{i-1}}{2\pi k_s H n_{bh}} g\left(\frac{t_n - t_{i-1}}{t_s}\right) \tag{3.31}$$

Equation 3.31 represents a superposition of step pulses. The superposition can be done because the pure conduction heat transfer model[6] is based on linear differential equations.

To solve for the change in fluid temperature given an effective borehole wall temperature and a constant heat extraction along the borehole, the effective borehole thermal resistance is used as discussed in the Borehole Resistance section above. Given the effective borehole thermal resistance network, the simple mean fluid tem-

[6] The model does not handle the effects of groundwater flow. As discussed by Chiasson et al. (2000), for groundwater flow to have a significant effect, high hydraulic conductivity and high hydraulic gradient must be present. The vast majority of locations do not meet these criteria. Future research is needed to develop fast and accurate design models that can be used for locations with high hydraulic conductivities and gradients.

perature between the inlet and the outlet, $\bar{T}_f$, can be determined:

$$\bar{T}_f = \bar{T}_b + \frac{\dot{Q}_b}{H} R_b^* \tag{3.32}$$

Since the effective borehole thermal resistance is derived based on the simple mean fluid temperature, the heat pump entering fluid temperature can be readily determined as:

$$T_{f,entering} = \bar{T}_f + \frac{\dot{Q}_b}{2\dot{m}c_p} \tag{3.33}$$

Therefore, the heat pump entering fluid temperature at hour n, h_n, can be determined by the following equation.

$$T_{f,entering,n} = \frac{\dot{Q}_{b,n}}{2\dot{m}c_p} + \frac{\dot{Q}_{b,n}}{H} R_b^* + \sum_{i=1}^{n} \frac{\dot{Q}_{b,i} - \dot{Q}_{b,i-1}}{2\pi k_s H} g\left(\frac{(h_n - h_{i-1}) * 3600}{t_s}\right) \tag{3.34}$$

The number of operations required to determine the heat pump entering fluid temperature in big O notation is given by $O(n!)$, where n is the number of time steps used in the simulation. This results in relatively high computation times, as shown in Figure 3.10, where the times for the hybrid time step simulation, discussed in the next section, are shown to be 3 orders of magnitude lower for a 20-year simulation. The hybrid time step procedure still makes use of a detailed simulation (Equation 3.34), but vastly reduces the number of time steps used by representing the load profile with monthly average and monthly peak loads. Therefore, in the design process where, say, 10 simulations might be done to size a ground heat exchanger, the hybrid time step is preferred. However, the availability of the hourly simulation in GHEDT allows us to confirm that the hybrid time step simulation has sufficient accuracy.

Using the loads shown in Figure 3.9, and the single U-tube BHE of Table 3.1 and Table 3.2, a three-year simulation of a 12x13 borehole field with 5m spacing is performed. The hourly simulation provides the heat pump entering fluid temperatures

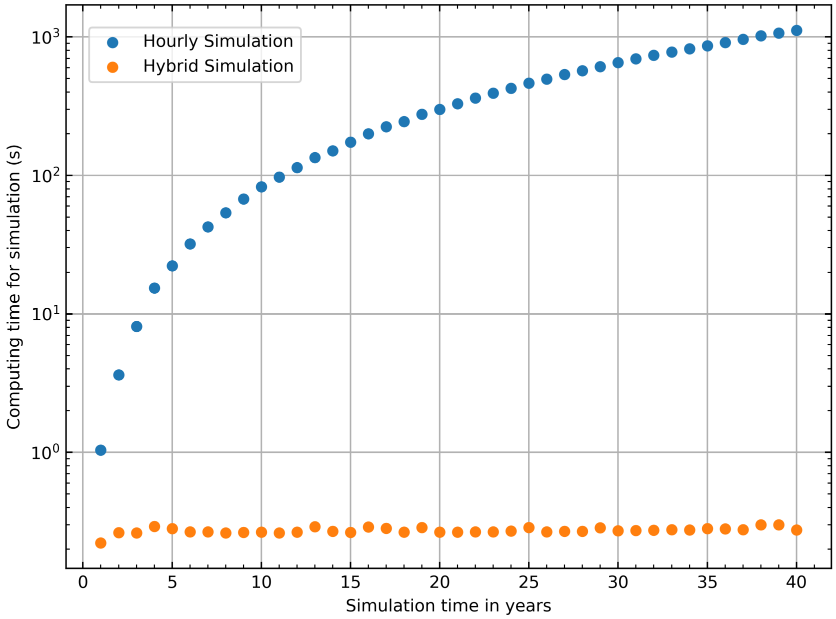


Figure 3.10: Computation times for various simulation periods (1-40) years – hourly and hybrid time steps.

at each hour are shown in Figure 3.11. The maximum and minimum temperatures are marked, as for the designing sizing procedure, these are the critical values.

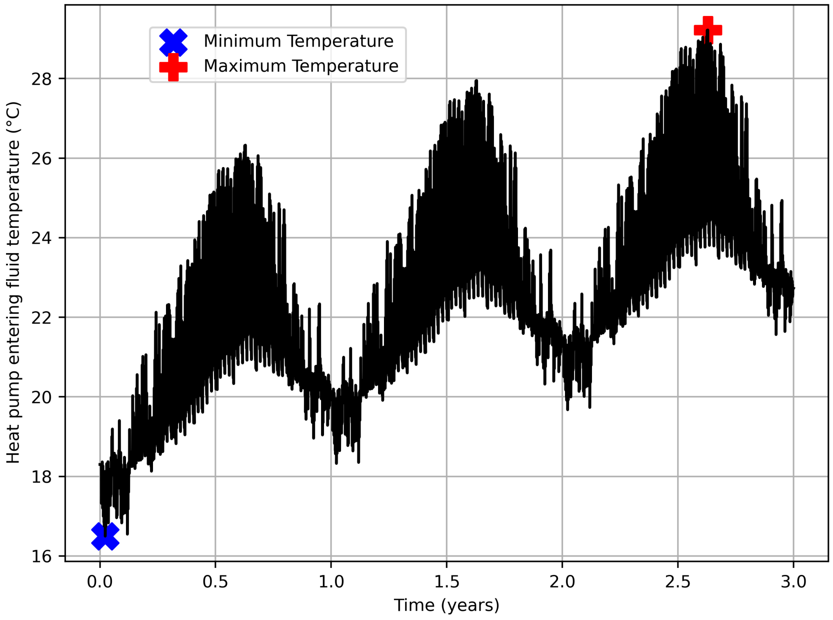


Figure 3.11: Hourly heat pump entering fluid temperatures determined with an hourly simulation for 3 years.

3.6.3 Monthly Hybrid Time Step

The hybrid time step procedure utilizes monthly average and monthly peak time steps, rather than using the 8760 hourly loads each year. The hybrid time step concept dates back at least until 1996 (Manickam et al., 1996). However, until Cullin (2008) developed a peak load analysis procedure and implemented it in an Excel spreadsheet, the duration of the square pulse for use in GHE design tools was left up to the engineer's judgement. Cullin and Spitler (2011) described a more complete investigation of the peak load analysis procedure, finding that compared to an hourly simulation for 48 building/climate combinations, the peak load analysis approach had an effect on the size between +8% overprediction to 6% underprediction. They deemed these errors small enough to consider the method viable for sizing, given uncertainties about the ground, and that the long-term g-function considers the ground to be homogeneous and only contain pure conduction. They also identified several limitations: (1) the peak load duration was the same for each month; (2) the calculated duration was based on only on the 24-hour period in which it occurred – a better estimate would take into account at least the previous day; (3) peak loads were always assumed to occur at the end of the month. Furthermore, the actual implementation required iterative manual analysis by the program user to find the correct peak length.

One of the new contributions of the work here is to improve the approach to remove the three limitations described above, and to completely automate the process, so that calculation of the loads and hybrid time steps is completely automated.

The improved and automated procedure first splits the ground loads into heating (ground heat rejection) and cooling (ground heat extraction) loads. The total, peak and average loads are found for each month. The day that the peak load occurs on is stored, and the 48-hour load containing that day and the prior day load profile is stored. If the peak load day occurs on the first day of the month, the last day of the previous month is used. If the peak load day occurs on January 1, December 31 will be used as the prior day.

The peak load duration for each month is found by simulating the 48-hour period (on an hourly basis) and pulses of a range of durations (1-hour, 2-hour, 3-hour, etc.) with two load profiles (a peak and nominal), and then a comparison is made. The first load 48-hour load profile is a constant peak load, $\dot{Q}_{peak}$, that is the difference between the monthly average and the actual peak load.

$$\dot{Q}_{peak,i} = max(\dot{Q}_i) - avg(\dot{Q}_i) \tag{3.35}$$

Where the subscript i refers to the month. The second simulation is a nominal load, $\dot{Q}_{nom}$ that finds the difference between each hour, denoted subscript j, and the average for the month. The load is then scaled by the maximum load for the month.

$$\dot{Q}_{nom,i,j} = \frac{\dot{Q}_{i,j} - avg(\dot{Q}_i)}{max(\dot{Q}_i)} \tag{3.36}$$

The duration that gives the same peak as the 48-hour hourly simulation is bracketed, and the actual non-integer duration in hours is found by interpolation. The curves used to determine the peak load duration for heating and cooling are shown in Figure 3.13 for the month of June given the hybrid load profile discussed below.

To ensure the described methodology performs as desired, a synthetic load profile is created, and shown in Figure 3.12. The synthetic load contains a monthly average load, and a monthly peak load for heating and cooling. The automated hybrid time step analysis should therefore find the same monthly average loads, and monthly peak loads as the synthetic load profile input. Essentially, given an hourly load profile, the hybrid analysis determines monthly average and monthly peaks. Therefore, the hybrid analysis performed on the hybrid load should result in the same hybrid load profile. This is done to ensure that the program is behaving as expected.

The first month's building heating load (ground heat extraction) has a constant load of 1 kW, with a peak of 2 kW applied at the 7th day for the first month for a duration of 1 hour. The peak load is then doubled each month, and the peak day, constant load and duration of peak load are incremented by one each month. Therefore, the second months heating load has a 2 kW constant load, a peak of 4 kW and the peak is applied at the 8th day of the month, the peak load duration is 2 hours and the peak load day one incremented by one day. The building cooling load profile (ground heat rejection) is applied opposite that of the heating load. The first months building cooling mode contains a peak load of 24 kW, a constant heating load of 12 kW and a peak load day at the 18th day of the month, and a peak load duration of 12 hours.

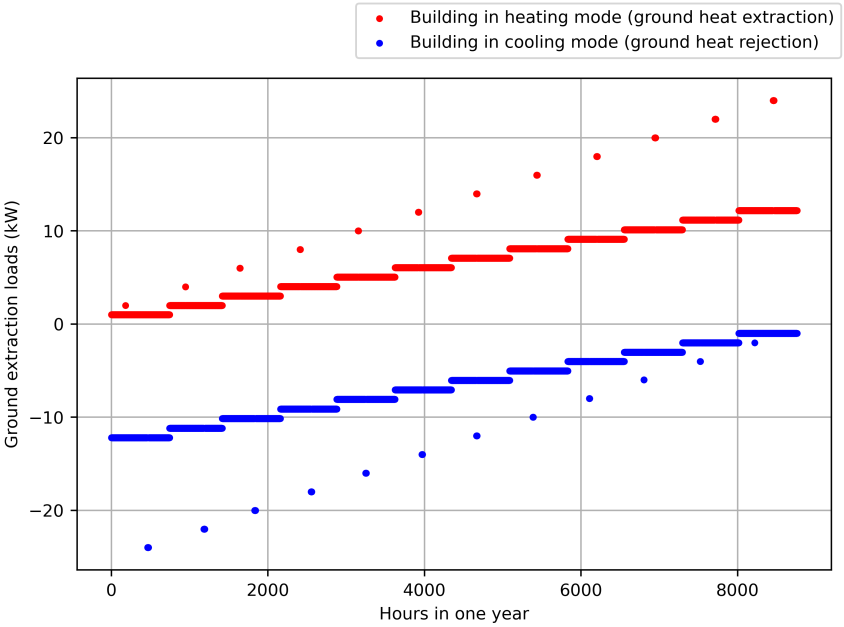


Figure 3.12: Synthetic ground extraction load profile with doubling peak load each month.

The peak load analysis for the synthetic load profile is illustrated for the month of June in Figure 3.13. The number of peak hours found are 6 and 7 for heating and cooling respectively. The heating and cooling loads both have a peak simulation, and a 48-hour nominal simulation. The peak hour is determined by drawing a horizontal line (shown in yellowish/orange) to intersect with the peak simulation. The time at which the horizontal line intersects is the peak duration for the month and heating/cooling load. The peak load analysis procedure results for the synthetic load were checked for each month (Table 3.6) confirming that the procedure works correctly. The errors all fall within 1%.

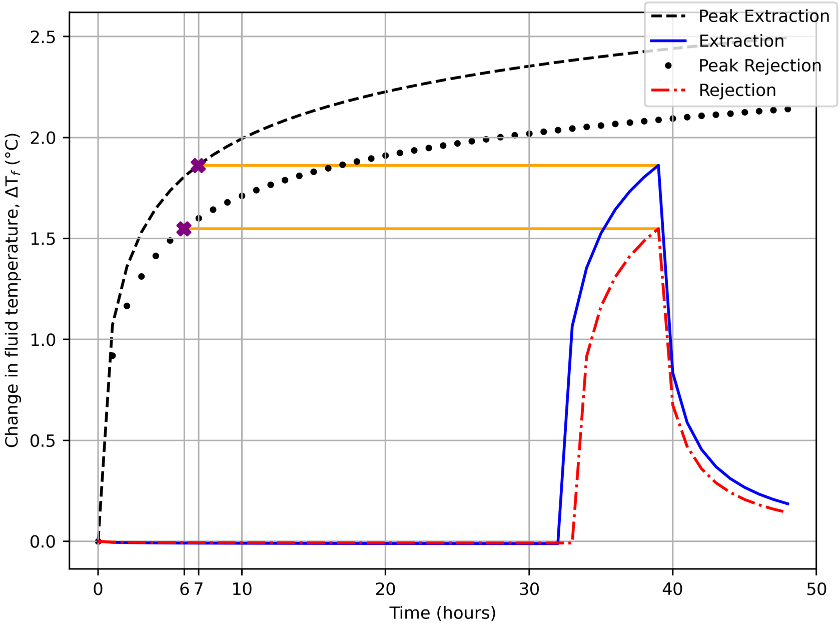


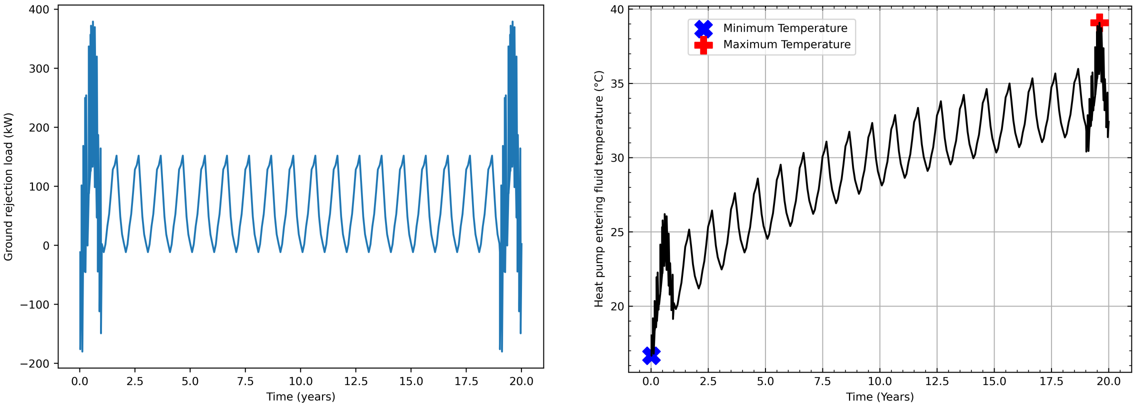
Figure 3.13: The peak load analysis to determine peak duration for heating and cooling loads in the month of June for a synthetic load profile.

Table 3.6: The output of the peak load analysis procedure on the synthetic load.

Fields	Total		Peak		Average		Peak Day		Peak Duration	
Load Type	Clg	Htg	Clg	Htg	Clg	Htg	Clg	Htg	Clg	Htg
January	9072	745	24	2	12.2	1	19	7	11.89	1
February	7513	1348	22	4	11.2	2	18	8	10.89	1.99
March	7540	2241	20	6	10.1	3	17	9	9.92	2.99
April	6561	2896	18	8	9.1	4	16	10	8.93	3.98
May	6016	3745	16	10	8.1	5	15	11	7.94	4.97
June	5089	4356	14	12	7.1	6.1	14	12	6.95	5.96
July	4500	5257	12	14	6	7.1	13	13	5.96	6.95
August	3745	6016	10	16	5	8.1	12	14	4.97	7.94
September	2896	6561	8	18	4	9.1	11	15	3.98	8.93
October	2241	7540	6	20	3	10.1	10	16	2.99	9.92
November	1444	8041	4	22	2	11.2	9	17	1.99	10.9
December	745	9072	2	24	1	12.2	8	18	1	11.89

In most cases where a single year's loads are repeated, the peak temperatures usually occur in the last year of operation, though peak minimum temperatures sometimes occur in the first winter, as shown in Figure 3.14. These loads are for a cooling dominated office building in Atlanta, and the temperatures would tend downward over time. Since computational time taken by the hybrid time step simulation is dependent on the number of steps, one way to save time is to not model the peak loads during the years between the first year and last year.

The ground heat extraction/rejection loads are converted to a hybrid time step representation and the result is shown in Figure 3.14a. Then, a hybrid time step simulation is performed for a single U-tube with a 12x13 rectangle with 5m uniform spacing. This gives an estimated temperature at the end of each time step, as shown in Figure 3.14b.



(a) 20-year hybrid load

(b) Heat pump entering fluid temperatures

Figure 3.14: Hybrid loads and resulting heat pump entering fluid temperatures for a 20-year simulation.

3.6.4 Validation

The results of the hybrid time step solution are checked against the results of the hourly simulation. The two sets of loads are shown in Figure 3.15 for a three-year period. As discussed in the previous section, peak loads are only applied during the first and last year. Visually, the hybrid peak loads match the actual peak loads quite closely, as expected. The hybrid procedure was validated in subsection 3.6.3, and a numerical error of the peak load representation was always less than 1% in Table 3.6. Figure 3.16 shows the results of a three-year simulation using these loads. For the first and third years, the results match within about 0.14°C. The error this causes in sizing is addressed in subsection 3.7.3.

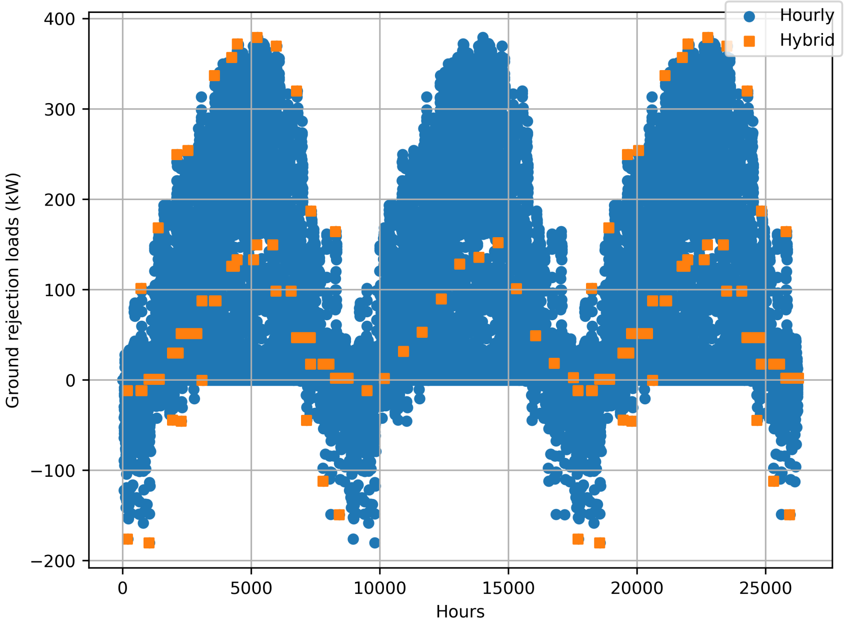


Figure 3.15: Hourly and hybrid time-step loads for the Atlanta office building over 3 years.

3.7 Sizing

3.7.1 Cost Function

The sizing procedure is aimed at finding the root of a “cost function” that is referred to as “excess temperature.” This excess temperature cost function may be unreferred to in literature up until this point. This is formulated by taking the maximum of the difference between the actual maximum heat pump EFT and the design maximum,

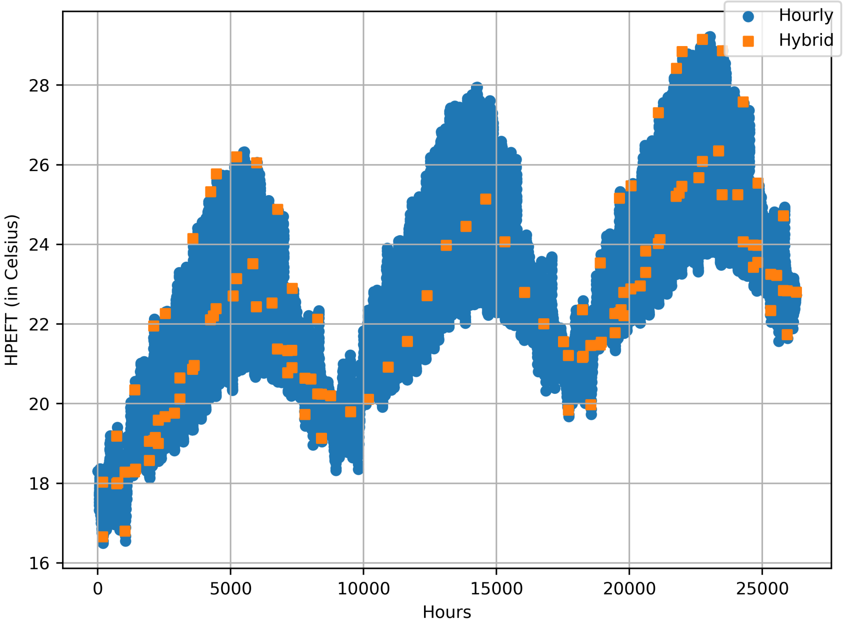


Figure 3.16: Hourly and hybrid time-step predictions of peak heat pump entering fluid temperatures.

and the difference between the minimum heat pump EFT and the design minimum.

$$\Delta T_{max} = max(EFT) - EFT_{max,allow} \tag{3.37}$$

$$\Delta T_{min} = EFT_{min,allow} - min(EFT) \tag{3.38}$$

The excess temperature then represents the maximum amount by which either design limit is exceeded.

$$T_{excess} = max([\Delta T_{max}, \Delta T_{min}]) \tag{3.39}$$

The goal then is to find the root of the excess cost function, which is the minimum depth at which the design limits are met. This is illustrated for one particular case in Figure 3.17 – the root, representing the minimum depth is indicated with an with a red “x” and corresponds to a design depth of about 136m. A random selection of a field, and a random minimum and maximum height range may not bracket the root. The material discussed in Chapter IV Borehole Configuration Selection in the GHEDT automates the selection process so that less hand calculation is necessary in the design process.

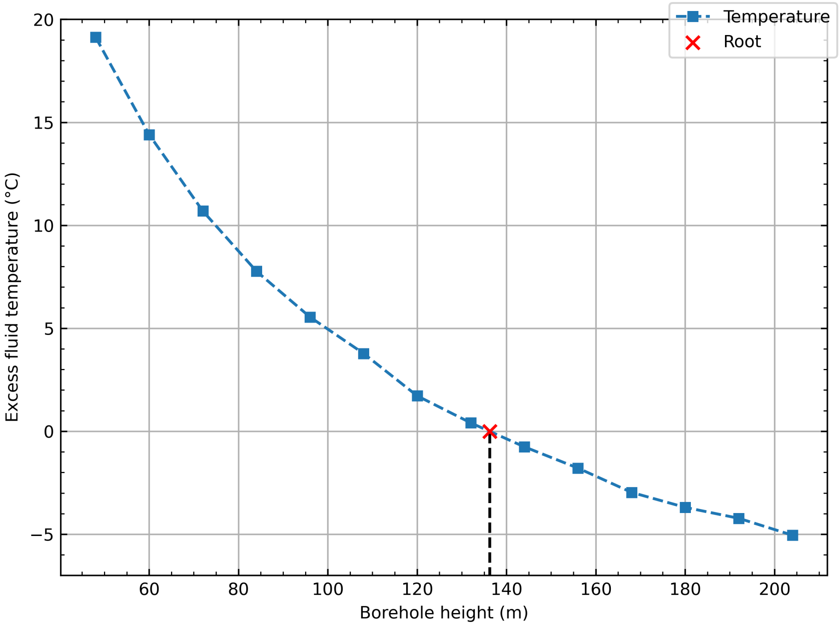


Figure 3.17: Hourly and hybrid time-step predictions of peak heat pump entering fluid temperatures.

3.7.2 Procedure

The sizing procedure utilizes the previously described simulation procedure to predict the minimum and maximum heat pump entering fluid temperature for a height, which then gives the excess fluid temperature. A search routine is used to adjust the height iteratively via interpolation until the root is found. The following section will now demonstrate the use of the sizing procedure for different borehole types.

3.7.3 Borehole Heat Exchanger Effects on Sizing

The sizing procedure is exercised for the three different types of BHE described in Table 3.1, Table 3.2 and Table 3.3, with a range of flow rates using the loads shown in Figure 3.9, are now sized with a long-term g-functions based on a 12x13 rectangular layout with uniform borehole spacing. The long-term g-functions for this case are output from GLHEPRO so that when comparisons are made, the differences are only due to the short-time step and the peak load analysis procedure. The system flow rate is varied from 0.05 to 0.5 L/s per borehole to provide an idea of how the BHE selected will play a role in the height. For reference purposes, the sizing is also done with the hourly simulation.

To start though, Figure 3.18 shows the results using the hybrid time step procedure in GHEDT. The results are counter intuitive, and some general comments are in order. The required borehole length depends partly on several factors – but in these cases, the main differences are: (1) the effective borehole thermal resistance, and (2) there is a secondary effect of increasing the flow rate on the design. This secondary effect can be explained by the fact that increasing the mass flow rate results in the exiting fluid temperature more closely approaching the mean fluid temperature. During summer peak conditions, the exiting fluid temperature is lower than the mean fluid temperature and increasing the flow rate then raises the exiting fluid temperature. Since this is the critical design variable, raising it has the effect of increasing the depth to counter this rise. Increasing the flow should decrease the effective borehole thermal resistance, which will decrease the size, but increasing the flow also causes the secondary effect just discussed. The balance of influences may increase or decrease the size as the flow rate increases.

This is a brief explanation of some of the perceived irregularities in Figure 3.18. To understand the results, it is important to first understand the flow regimes. For each of the flow rates and heat exchangers, the Reynolds numbers are plotted for both programs in Figure 3.19, along with the critical Reynolds number (2300) and the upper bound on our approximation of the transition region (Re=4000). For the co-axial heat exchanger, the annulus Reynolds numbers are clearly in the laminar region or transition region all the way up to the flow rate of 0.5 L/s. The double U-tube is in the transition region at the lowest flow rate.

In order to validate the design procedure of GHEDT, the sizing results are com-

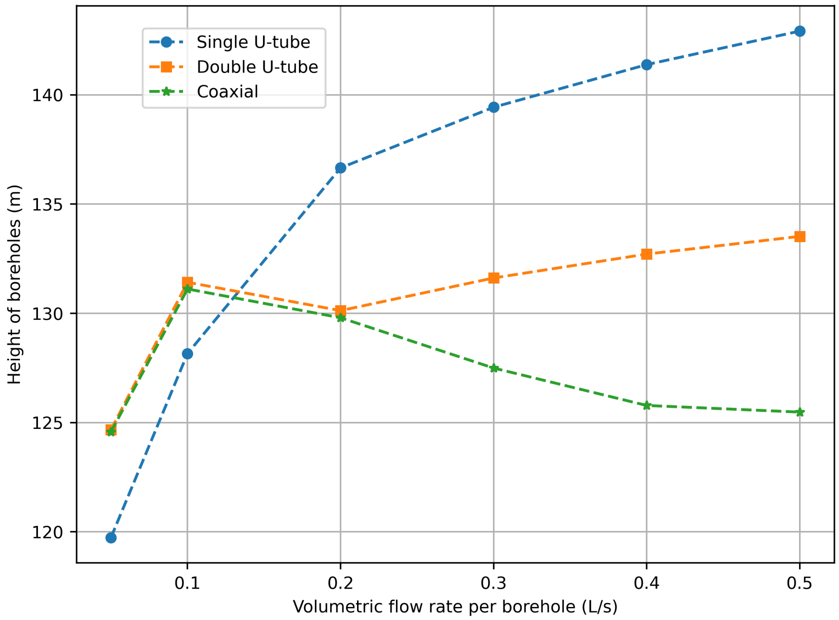


Figure 3.18: Variation of design heights calculated with GHEDT.

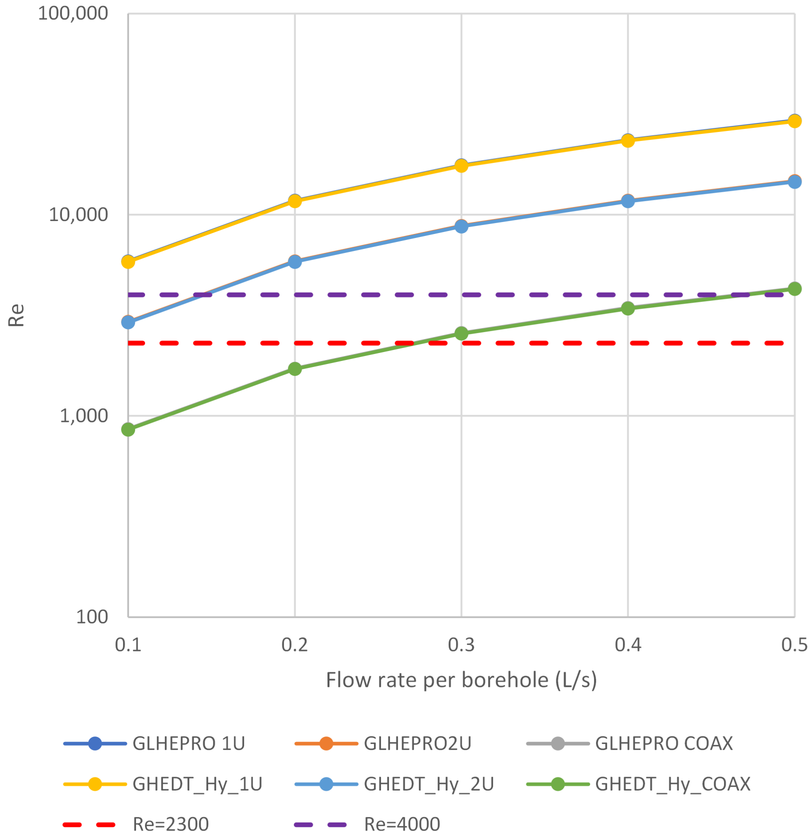


Figure 3.19: Reynolds numbers for the different BHE and flow combinations.

pared to GLHEPRO. Figure 3.20 shows the percent error of sizing with the reference case being the hourly time step simulation of GHEDT. The simulation procedures of GHEDT appear to perform as desired, resulting in borehole heights that are close to those found with GLHEPro. With a flow rate of 0.1 L/s per borehole neither of the hybrid time step procedures perform well. However, the typical design flow rate is 0.2-0.3 L/s per borehole for every 100m of drilling depth. These cases are sized for well over 100m depth per borehole. However, if a designer was concerned with the accuracy of a design produced by GHEDT, a 20 year hourly simulation (about 5 minutes of computation time) at the design height selected by the hybrid procedure could be utilized to check heat pump entering fluid temperatures.

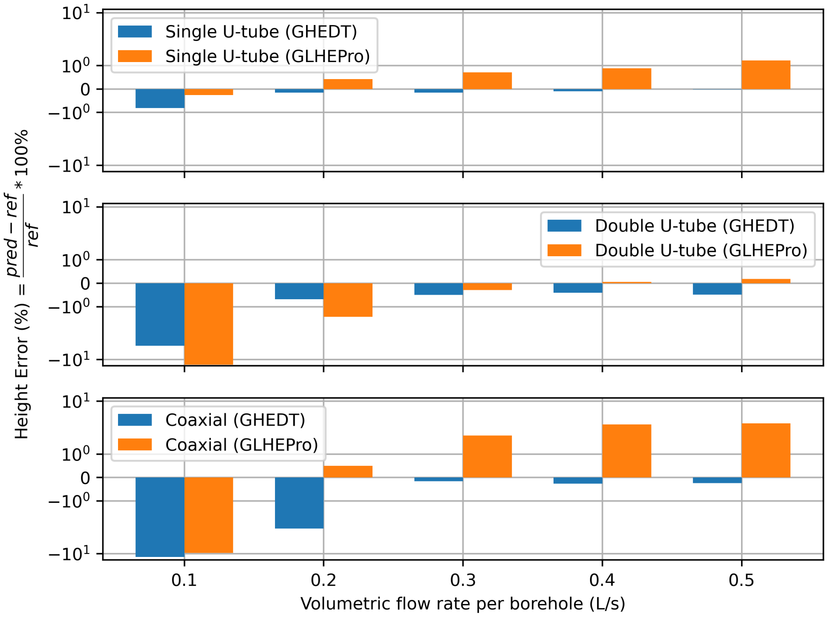


Figure 3.20: Quantifying error in the hybrid time step calculation for GHEDT and GLHEPRO when compared to the hourly simulation of GHEDT.

While for most cases, the improved hybrid time step procedure of GHEDT appears to perform better than Cullin's original implementation, though the short-time step simulation results are not identical. Additionally, for the coaxial heat exchanger, the transition region appears to be treated differently based on what was seen in Figure 3.6 of the Borehole Resistance Validation section, which results in slight differences in the effective borehole thermal resistance. Further investigation of the short-time step procedures and the treatment of the transition region would be useful to resolve all of the smaller differences.

3.8 Conclusions

GHEDT is a powerful new tool for use in designing ground heat exchangers. This chapter emphasized the treatment of different types of borehole heat exchangers with an approach that both allows them to be sized within the tool, but also provides an equivalent single-U-tube specification that can be used within EnergyPlus to simulate double U-tubes and co-axial BHE. This feature has not been previously available.

GHEDT has an hourly time step procedure, that can be used to validate the heat pump entering fluid temperature results provided by the hybrid time step procedure. Due to faster computation times, the hybrid time step can be used to size a system. An hourly simulation could then be executed to verify the hybrid procedures results.

GHEDT also makes use of a new and improved hybrid time step procedure. Though a fuller demonstration of the improvements, like that of Cullin and Spitler (2011) is still forthcoming.

The next chapter, Chapter IV Borehole Configuration Selection in the GHEDT, presents the most powerful new feature of GHEDT – automated selection of borehole configurations. Spitler and Cook (2021) previously described selection of borehole configurations for regularly shaped borehole heat exchangers, with the use of precomputed g-functions available in the g-Function Database (Spitler et al., 2021; Cook, 2021). The next chapter presents improvements to existing design procedures by computing accurate g-functions during selection – a feat not possible until very recent. With the use of the fast g-function calculation, the next chapter also presents a design algorithm where the outer available land area can be described by any polygonal shape, and sections of the available land area can be marked as “no drilling zones.”

CHAPTER IV

Borehole Configuration Selection in the GHEDT

4.1 Introduction

Ground source heat pump systems are an energy efficient means of providing heating and cooling to buildings. The best technique for prediction of borehole depths in a vertical ground heat exchanger (GHE) involves superimposing a unit step-response g-function with ground loads (Cullin et al., 2015). The g-function is an intermediate solution of a pure conduction model. g-Functions were historically pre-computed and stored in libraries out of necessity due to the long calculation times required. The g-function quantifies an average rise in non-dimensional temperature due to a constant rate of heat input to the borehole(s).

The original theory of the g-function dates to the work of Prof. Claesson of Lund University and his PhD student Eskilson (1987). The original g-function calculation program, *Superposition Borehole Model* (SBM) (Eskilson, 1986), was utilized by Hellström (1991) to generate a library of g-functions. The library is accessed by GHE sizing tools, such as the *Ground Loop Heat Exchanger Professional* (GLHE-PRO) (Marshall and Spitler, 1994; Manickam et al., 1996; Spitler, 2000; Cullin, 2008; Grundmann, 2016) and *Earth Energy Designer* (EED) (Hellström and Sanner, 1994). The SBM is closed source. Since the generation of the library in the 1990's, the peer-reviewed literature has been silent as to any further developments in the SBM methodology or expanding the content of the original library. The limited set of borehole heat exchanger geometrical configurations available in the library has served as the basis for many installed GHEs. GLHEPRO relies on the design engineer to manually select a configuration, perform a sizing routine and then iterate the process until an acceptable sized borehole height is found. The sizing routine varies the height of the boreholes in a selected configuration to keep the circulating fluid temperature within an acceptable range (e.g. 5°C-35°C) over a design period (e.g. 20-30 years). Building heating and cooling loads are a required input. EED (BLOCON, 2015) contains functionality to consider user defined inputs for a rectangular area, a borehole spacing range and height range. All the configurations in the library that fit in the rectangular area, and can be sized within the user specified height range, are sized and presented to the user. The user is then given the option to sort by category (e.g. total height or total cost).

This chapter presents a Python package, *Ground Heat Exchanger Design Toolbox* (GHEDT), that has the ability to automatically select borehole configurations based on a user defined target depth and geometric constraints. This chapter describes a new methodology for optimization of borehole fields based on a user-defined target depth. GHEDT contains novel abilities when compared to EED. These abilities include: automated placement of boreholes with different x- and y-spacing, live-time g-function calculation during design, consideration of an outer land described as a polygon by its vertices and the ability to treat inner polygonal no-drilling zones. The live time g-function calculation in GHEDT is dependent on the open source g-function calculation toolbox, *pygfunction* (Cimmino, 2018b). The default g-function calculation utilizes the uniform inlet fluid temperature g-function boundary condition (Cimmino, 2015, 2019), a newly added[1] feature to *pygfunction* that allows the segment lengths to be unequal and the equivalent borehole method (EBM) of Prieto and Cimmino (2021). g-Functions from Hellström's library are shown to over approximate significantly more than the live calculation of GHEDT when compared to a reference case. Using unequal

[1] The unequal segment feature was proposed by the author (`https://github.com/MassimoCimmino/pygfunction/issues/135`) and was implemented in collaboration with Cimmino (`https://github.com/MassimoCimmino/pygfunction/pull/136`).

segments is shown to both decrease computation time and result in more accurate g-functions with the use of less segments.

4.2 Background

Eskilson's SBM computed a two-dimensional (radial-axial) finite difference method (FDM) model, where each discrete point on the grid contained the temperature rise based on heat input to a line segment[2]. The temperature rise at a node in the mesh can be determined by summing the individual contributions of all the sources (boreholes). This process, called superposition, is possible because the pure conduction model is derived from Fourier's linear partial differential equation that describes the transfer of heat in solids. This gives, in effect, a three-dimensional solution. The vertical mesh was calculated by increasing at a rate of $\sqrt{2}$ towards the middle height of the borehole. The end segment length was a fraction of the height, α.

$$\alpha = \frac{1}{2}\frac{\sqrt{2}-1}{\sqrt{2}^{m}-1} \tag{4.1}$$

Where m is half the number of vertical segments. While the vertical segment lengths chosen were based on a ratio α that could be readily calculated, the horizontal radial mesh lengths were determined on a case-by-case basis. The first position, or zeroth index of the horizontal mesh array was reserved for the value of the boundary condition, and is located at the borehole radius. Therefore, the boreholes were uncoupled from the solution and maintained at a uniform temperature along the length (Cimmino, 2019). Each borehole wall temperature was equal in the field. Recommendation given for the remainder of the mesh was to vary the size of the cells until a solution independent of the mesh was found. The computer manual for SBM, by Eskilson (1986), dedicates an entire chapter to the accuracy of the calculation based on the radial mesh cell sizes, and external reports that discussed the rules to choose

[2] The line segment is a numerical representation of a finite line source.

a radial mesh are referred to. Later, Eskilson and Claesson (1988) describe rules for the horizontal mesh. The recommended fineness of the grid (or number of grid cells) increases as the number of boreholes increase.

Claesson and Javed (2011) derived a single integral analytical finite line source solution by integrating a point heat source over the length of the borehole. A mirror borehole was placed above the ground with negative temperature rise to impose an isothermal ground surface. This analytical solution greatly decreased the computation time of the g-function when compared to Eskilson's numerical SBM. This decrease in computation time resulted from temperature responses between boreholes being computed, rather than a grid of responses, greatly reducing the number of finite line source evaluations necessary. The methodology was implemented into EnergyPlus for simulating ground heat exchangers and into GLHEPRO by Malayappan and Spitler (2013) for the use of live time g-function calculations. Though Malayappan and Spitler (2013) found that for borehole fields containing more than about 30 boreholes, the solution over approximated Eskilson's unit-response functions.

Cimmino and Bernier (2014) reformulated the solution of Claesson and Javed (2011) in order to represent each borehole with multiple stacked finite line sources. These sources represent the boreholes with each segment having its own burial depth and height. This stacked finite line source model now makes the g-function calculation semi-analytical – the solution depends partly on the vertical grid. The formulation of the segment response is dimensionless, and the response between each pair of segments is referred to as a segment-to-segment response. Cimmino and Bernier found that the handling of the distribution of heat in the borehole field had a significant impact on the g-function. The following are four boundary conditions for distribution of heat used to solve the g-functions:

- Uniform heat flux (UHF) - A constant heat flux is prescribed at each borehole wall for all boreholes in the field. This solution is what Claesson and Javed

(2011) presented.

- Uniform borehole wall temperature (UBWT) - At every point in time, the borehole wall temperature in the field is uniform, or equal. The heat rejection to each borehole must vary with time, but the total average heat flux prescribed to the boreholes is constant.

- Uniform inlet fluid temperature (UIFT) - As described by Cimmino (2015), the entering fluid temperatures are uniform for all boreholes but increase with time. The internal borehole thermal interaction must be considered. The borehole wall temperature is equal along each segment, but the borehole wall temperatures can vary along the borehole (considering the borehole is broken into multiple segments), and throughout the field. The heat rejection to each borehole can vary with time, but the total average heat flux prescribed to all boreholes is constant.

- Mixed inlet fluid temperature (MIFT) - As described by Cimmino (2019), a mixed arrangement of series and parallel connected boreholes can exist. The inlet fluid temperature of each borehole is determined either by the inlet fluid temperature to the borefield, or the exiting fluid temperature from another borehole. The borehole wall temperature and distribution of heat is prescribed in the same manner as UIFT. A network of parallel connected boreholes solved with the MIFT boundary condition simplifies to a UIFT solution.

Cimmino and Bernier (2014) constructed a system of equations that ensured a UBWT boundary condition. They were able to compute g-functions that fell within a root mean square error of $\pm5\%$ (excluding boreholes in a line) when compared to Eskilson's g-functions, by enforcing a UBWT boundary condition and discretizing the boreholes into 12 equal segment lengths. Cimmino and Bernier (2014) discussed the numerical accuracy of the solution by computing g-functions with 256 equal segments along the borehole and comparing the error using less segments. They showed that the stacked FLS solution has a dependency on the vertical grid, leading them to also wonder if the grid utilized by Eskilson was fine enough to achieve grid independence in all cases. Cimmino (2015) went on to implement a UIFT g-function calculation for boreholes containing single U-tubes. As can be seen in Figure 4.1, the UIFT solution tends upwards towards a g-function computed with the UHF boundary condition as the (effective) borehole thermal resistance increases. The UIFT solution tends downwards towards the UBWT boundary condition as the borehole thermal resistance decreases. Cimmino (2016) introduced UIFT solutions with multiple U-tubes in the borehole. Cimmino (2019) implemented a mixed inlet fluid temperature (MIFT) boundary condition that allows for both series and parallel connected networks to be modeled, which simplifies to UIFT when the boreholes are piped in parallel.

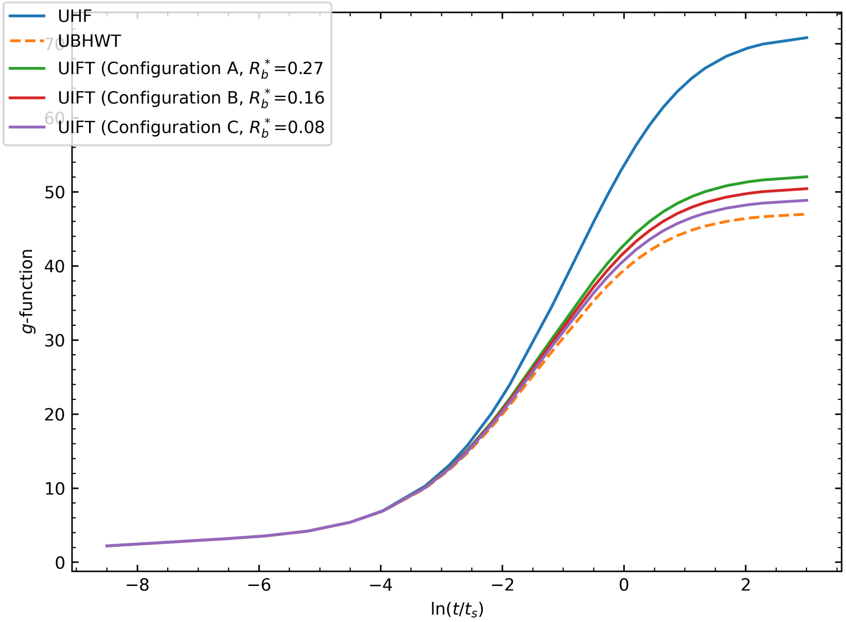


Figure 4.1: The g-functions computed for UHF, UBWT and UIFT boundary conditions, with varied UIFT boundary condition g-functions based on effective borehole resistance.

Cimmino (2018a) increased the computational speed of the g-function calculation by pre-determining boreholes that contain similarities or have the same segment responses. A symmetric group of segments all contain the same radial distance and burial depth from a reference borehole segment. Cimmino (2018a) presented algorithms that when implemented can determine symmetric borehole groups. The similarity solver, when compared to the previous detailed solver, resulted in a 308x speed increase for a 12x12 rectangular borefield. Cimmino (2018b) released an open-source g-function calculation toolbox, named *pygfunction*.

With the task of computing g-functions for borefields containing large numbers of boreholes, up to 1024, it was found by Cook and Spitler (2021) that the memory requirements of *pygfunction* (version 1.1.1) were more than typical high-performance computers could handle. The size of the segment response matrix in *pygfunction*, based on the number of boreholes n_{bh}, number of equal vertical segments along the length of the boreholes n_q and the number of points in time n_t, was $(n_{bh}n_q \times n_{bh}n_q) \cdot n_t$. Cook and Spitler packed[3] the segment response matrix, which is symmetric by reciprocity, to create a response matrix of the size $(n_{bh}n_q \times n_s) \cdot n_t$.

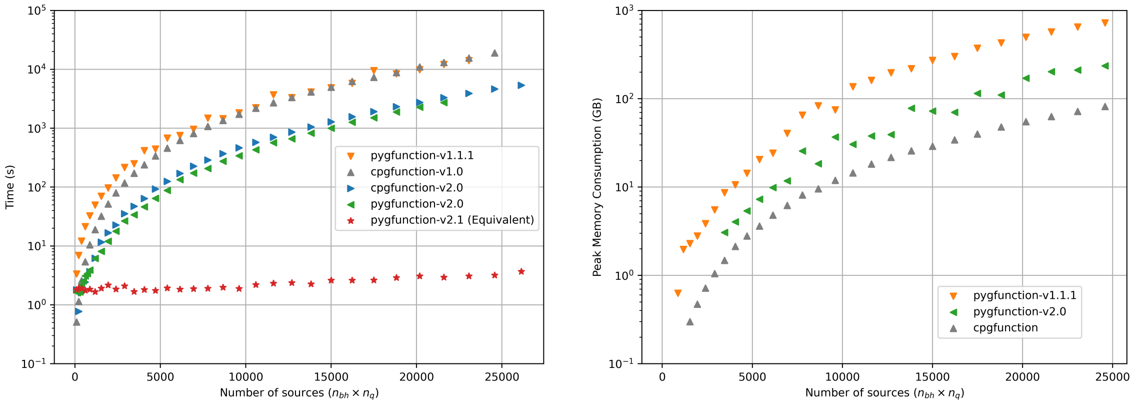
$$n_s = \frac{n_q(n_q+1)}{2} \tag{4.2}$$

The result of Cook and Spitler (2021) was a low-level C++ implementation of Cimmino's UBWT system of equations that used on average 8-9 times less memory than *pygfunction*. The UBWT system of equations are described in Cimmino and Bernier (2014) and Cimmino (2018a). Congruent with the converged solution discussion of Cimmino and Bernier (2014), it was found that calculated g-functions tend to shift upwards as the number of finite line source segments along the height of the borehole is decreased. To further decrease the memory consumption, this fact was utilized to try to approximate converged (many segments along the length of the borehole) UIFT g-functions by decreasing the number of segments used in the UBWT calculation.

[3] James A. Cook helped in the process of deriving a function that could index the packed matrix corresponding to its (i, j) position in the original full matrix.

The reference UIFT g-functions were chosen based on typical installation practices in North America. The result was that most cases consumed less than 96GB of RAM, which allowed Spitler et al. (2021) to compute a large library of g-functions on the Oklahoma State University High Performance Computing Cluster (OSUHPCC, 2021). A fork of *cpgfunction*, named *cpgfunctionEP*, was later modified to be a pure C++ project for integration into EnergyPlus (available since the 9.6.0 release on September 23, 2021). The deprecation of external Fortran libraries, which are optimized for vectorization, resulted in a decrease in speed performance. The number of segments used by default in the g-function calculation of *cpgfunctionEP* are those that provide approximate UIFT solutions for average use cases.

The original version of *cpgfunction* version 1.0 outperformed *pygfunction* version 1.1.1 for most problems, though when the problem was very large, *cpgfunction* was slower. A speed increase was obtained in a second version of *cpgfunction* version 2.1 by vectorizing the spatial and temporal superposition of the time varying heat extraction rates with the segment responses. This vectorization was already being done in *pygfunction*. However, the next release of *pygfunction* version 2.0 was significantly faster than *cpgfunction*, due to Cimmino vectorizing the identification of similarities as well as the calculation of segment responses. Prieto and Cimmino (2021) proposed an equivalent borehole method (EBM), that vastly reduced the number of segment responses required to be calculated by determining equivalent borehole groups using a hierarchical clustering scheme for determination. The EBM determines groups boreholes that have similar borehole wall temperatures. A single steady state finite line source solution containing one segment is utilized in the initial calculation of the dimensionless borehole wall temperature in the field. From there, a hierarchical agglomerative clustering method is performed, and a minimum number of groups, G_{min} are determined by a prescribed cutoff threshold of ½ applied to a dendrogram. The number of vertical lines (groups) that are intersected by the horizontal line is the minimum number of groups necessary. Prieto and Cimmino find that setting the number of groups G to be one greater than the minimum number of groups provides sufficient accuracy. The size of the segment response matrix was reduced to $(Gn_q \times Gn_q) \cdot n_t$, and the resulting system of equations was also reduced. Figure 4.2a shows the time comparison[4] for *pygfunction* and *cpgfunction* over the range of versions based on the number of sources in the calculation. Figure 4.2a shows a significant speed improvement provided by the equivalent method of Prieto and Cimmino (2021), available in *pygfunction* version 2.1. The speed increase can be over 1000x. Figure 4.2b shows the memory consumption for *pygfunction* and *cpgfunction* based on the number of sources used. The memory consumption of the equivalent method does not appear on Figure 4.2b because its consumption is always less than 0.1 GB.



(a) Square fields timing

(b) Square fields memory

Figure 4.2: Timing and memory comparisons for pygfunction and cpgfunction on the basis of the number of sources.

A recent development of *pygfunction* has been made to allow the segment lengths along the borehole to be unequal. Unequal segments lengths are shown in this chapter to improve the speed and accuracy of *pygfunction*. Cimmino and the author collaborated on a function to determine unequal segments in the vertical direction. The method was finalized by Cimmino. The function returns segment ratios along the

[4] Dual Intel "Skylake" 6130 CPUs available on the Oklahoma State University High Performance Computing Center cluster, named Pete, were utilized.

borehole based on a required number of segments input and an optional end length ratio input. The function discretizes the borehole by utilizing a geometrically expanding mesh. While Eskilson proposed a constant expansion factor of $\sqrt{2}$, Cimmino's expansion factor is solved for based on the number of segments and the end segment ratio provided. The default end segment length ratio was selected to be 0.02. Optimal vertical discretization is a subject of future[5] work for *pygfunction*. This chapter shows that a 2% end length ratio and 8 unequal segments that expand towards the center of the borehole can reduce computation time and increase accuracy when compared to equal vertical segment lengths. The 8 unequal segment length discretization is abbreviated here as 8USL. The segment discretization of Eskilson (unequal), Cimmino (equal) and the current[6] 8USL schemes are shown in Figure 4.3. The equivalent borehole method combined with 8 unequal segments along the borehole serves as the basis for live UIFT g-function calculations in the GHEDT optimization process.

4.3 Methodology

4.3.1 Validation

Unequal stacked finite line source segment lengths are undiscussed as of current in the literature. Cimmino is the progenitor of the stacked finite line source methodology and

[5] The optimal vertical discretization of the stacked finite line source segments is an open issue (`https://github.com/MassimoCimmino/pygfunction/issues/153`) for pygfunction.

[6] If a name was to be put on the "current" scheme, it would be fitting for it to be a combination of names: Cook, Cimmino, Spitler and West. The idea of unequal segment lengths had been discussed between Cook and Spitler some time ago, but it wasn't until his undergraduate research assistant Timothy West edited an offline pygfunction version 1.1 that an unequal segment length stacked finite line source solution had been computed. West and Spitler developed a "3 segment scheme" that could approximate typical UIFT g-functions using the UBWT boundary condition with unequal segments, one step improved from the equal segment length adaptive discretization scheme of Cook and Spitler. Cook understood the speed improvements in the similarity identification of pygfunction version 2+ and began working on improving the more complicated pygfunction version 2 to handle unequal segments. The initial development of unequal segments into pygfunction version 2+, was started with the 3-segment scheme in mind. However, upon completion, the EBM method was available, and orders of magnitude faster than a UBWT g-function computed with the similarity method and 3 (unequal) segments. After Cook showed some initial promising results to Cimmino, he was a huge help in wrapping up the unequal segment length functionality for use and compatibility with all available g-function calculation methods of pygfunction.

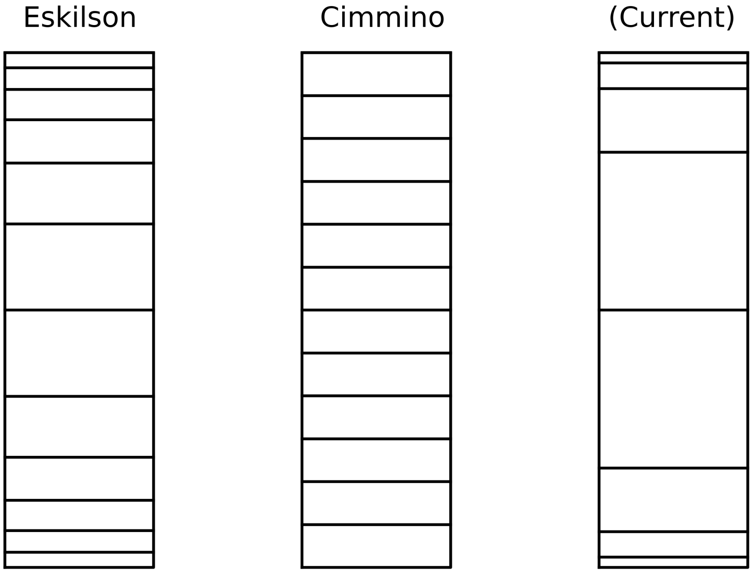


Figure 4.3: Eskilson's 12 segment discretization scheme, Cimmino's uniform discretization and the current segment ratio discretization using an end length ratio of 0.02 with 8 segments.

until recently *pygfunction* has always considered uniform segment lengths along the borehole. The author proposed unequal segment length functionality and collaborated extensively with Cimmino to provide full unequal segment length functionality to all boundary conditions and solver methods of *pygfunction*. The unequal segment length function of *pygfunction*, finalized by Cimmino, does differ from Eskilson's vertical grid. The function requires the number of vertical segments, n_q and has an optional end length ratio, α which is set to a default of 0.02. To determine the segment ratios along the borehole, the required number of segments is 3. Though, if 1 or 2 segments are requested then the user is warned that the end length ratio is over-ridden. If 3 segments are described, the end length ratios are set at the top and bottom of the borehole, and the remainder of the ratio makes up the center segment. If the end length ratio multiplied by the number of segments is 1, then the segment ratios are equal along the borehole. If the number of segments provided is greater than 3 and even, the following polynomial expression is used to solve for an adaptive expansion factor, f. By adaptive, it is meant that the expansion factor is not constant along the borehole, where Eskilson used a constant expansion factor of $\sqrt{2}$.

$$0 = (1 - 2\alpha) - fx + 2\alpha x^m \tag{4.3}$$

Where $m = n_q/2$. The expansion factor is solved for based on the real root that ensures the sum of the segment ratios are equal to 1. The segment ratios along the borehole, where $r_0 = \alpha$, are given by the following equation, where $0 \leq i < m$, and the ratios are symmetric across the center of the borehole.

$$r_i = f^i \cdot \alpha \tag{4.4}$$

If the number of segments is greater than 3 and odd, then the adaptive expansion factor is solved by determining the root of the following polynomial.

$$0 = (1 - 2\alpha) - fx + \alpha x^n + \alpha x^{n+1} \tag{4.5}$$

Where $n = (n_q - 1)/2$. The ratios along the borehole are set using Equation 4.4, though the ratios for this case do not meet at the center. The center segment ratio is set to be $f^m \cdot \alpha$.

The impact of varying the end segment ratio and the number of sources is analyzed. To analyze the impact of varying the end segment ratio and the number of sources, a reference UIFT g-function for a 12x13 configuration with 5m equal spacing at a borehole height of 96m utilizing 96 equal segment lengths is computed using the similarities solver[7]. Other inputs are described in Table 4.1 and Table 4.2, and

[7] The "similarities" solver (Cimmino, 2018a) made use of similarities between pairs of segments to reduce the amount of computation time for computing segment-to-segment response factors. The EBM solver makes additional approximations not made by the similarities solver. It is expected that, with the same discretization, the similarities solver should be more accurate, but slower than the EBM solver. In this work, the similarities solver was used to compute the reference cases and also cases with the 12ESL discretization. Unequal segments lengths have full functionality with all

the g-functions are solved at Eskilson's 27 logarithmic dimensionless points in time (shown in Appendix B). A g-function with unequal segments that utilizes the EBM is computed with various combinations of number of sources n_q and end segment ratios α. The number of sources is ranged from 5 to 25, and the end segment ratio is ranged from 0.01 to 0.05. The mean percent error based on the end segment ratio and number of sources utilized is given in Figure 4.4. The mean percent error is computed over the entire g-function curve and quantifies the average error. As the end segment ratio is decreased and the number of sources increased, the solution better approximates the reference case.

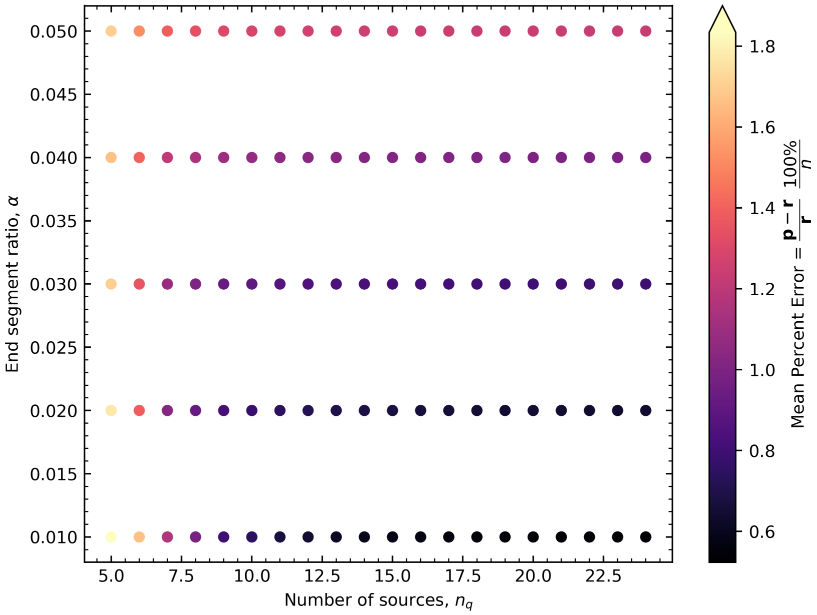


Figure 4.4: Mean percent error of the equivalent borehole method with unequal segments based on the number of segments used and end segment ratio.

Now, an end segment ratio is selected as 0.02 and held constant. The number of sources are varied from 3 to 20. The mean percent error for the EBM with unequal segments, and the similar borehole method (SBM) with equal segments are compared

solvers in pygfunction (detailed, similarities, equivalent) and all boundary conditions (UHF, UBWT, UIFT). However, the resulting increase in accuracy from utilizing unequal segment lengths is focused on the UIFT boundary condition with the EBM. Unequal segment lengths can be used with the similarities solver, but the resulting increase in accuracy is not shown in this thesis.

in Figure 4.5. The EBM is shown to be more accurate than the SBM with fewer segments when utilizing unequal segment lengths. The slope appears to nearly flatten out for the EBM unequal segment curve at around 8 segments. Using more than 8 segments for an end segment ratio of 0.02 is futile.

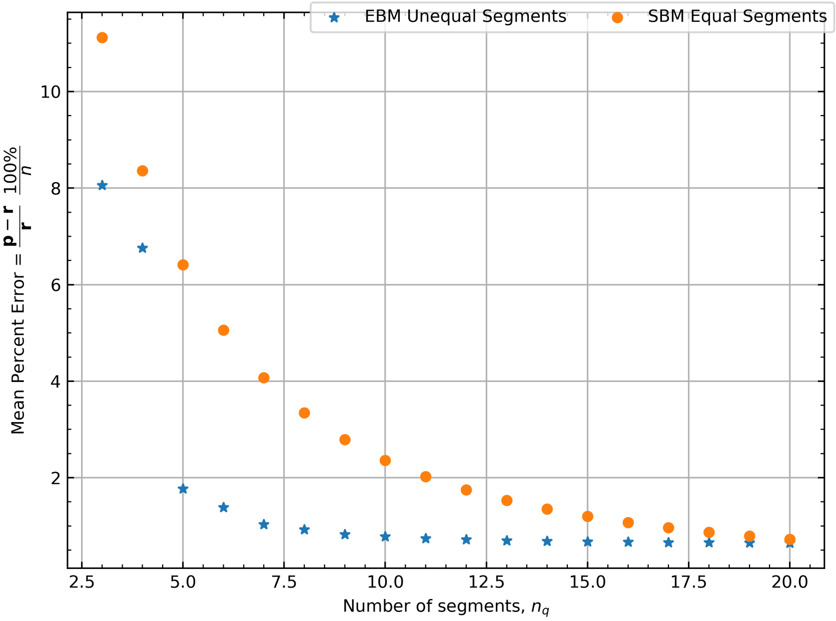


Figure 4.5: Mean percent error of the equivalent borehole method utilizing unequal segments with an end segment ratio of 0.02, and the similar borehole method using equal segment lengths.

The equivalent borehole method was recently proposed by Prieto and Cimmino (2021) and is validated here for use in the GHEDT. The unequal segment length g-function computed with the EBM is validated by quantifying the mean percent error in relation to a reference UIFT g-function. This is done over a range of borehole configurations. The g-Functions are computed at 96m depths for a single U-tube geometry described in Table 4.1. The remainder of the inputs are described in Table 4.2, and the g-functions are solved at Eskilson's 27 logarithmic dimensionless points in time (shown in Appendix B). The reference g-function is computed using the UIFT boundary condition (Cimmino, 2019) with the similarities solver (Cimmino, 2018a) that contains 96 equal segments along the borehole, equaling 1 meter segment lengths.

Cases for existing methods are computed, using UBWT (Cimmino and Bernier, 2014; Cimmino, 2018a) and UIFT (Cimmino, 2015, 2019) boundary condition at 12 equal segments (abbreviated 12ESL) along the borehole. The EBM (Prieto and Cimmino, 2021) is used to compute g-functions with 12 equal segment lengths. Finally, the 8 unequal segment length (8USL) discretization is coupled with the equivalent borehole solver, proposed here to serve as the basis for live time g-function calculation in GHEDT. The mean percent error over the curves are computed to the reference case. This is done for a set of 20 square cases ranging from 1 borehole to 400 boreholes. The size of the problem, or the number of sources used in the "converged" case, is dependent on the amount of RAM available. These cases were computed with so called bigmem nodes on the OSUHPCC (2021) that have 760GB of RAM. Cases containing more than 400 boreholes with 96 equal segments along the length for each borehole resulted in memory allocation failures. Computation of these reference cases is not feasible for most users, given that the average current workstation probably contains 8-16 GB of RAM (in December of 2021).

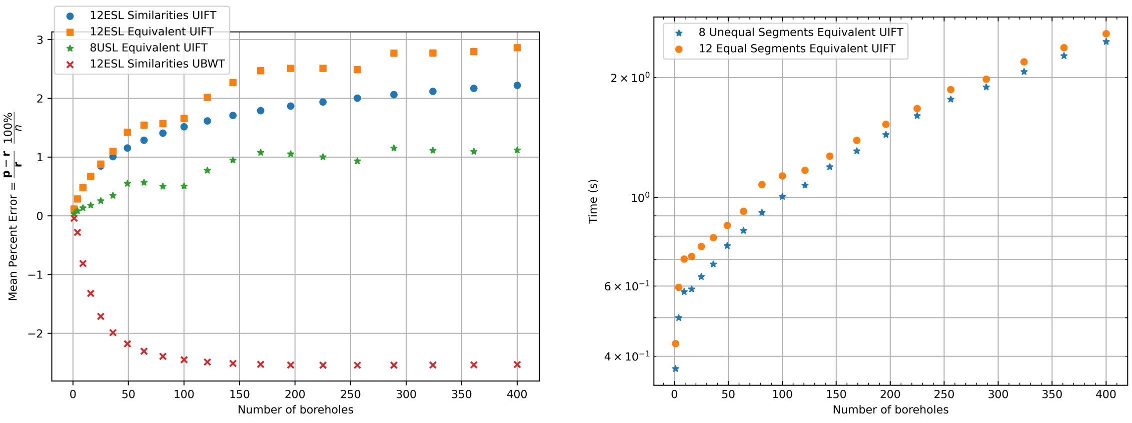
The mean percent error results for the square cases are given in Figure 4.6a. The 8USL discretization is shown to provide an improvement to the accuracy of the g-function calculation. For the square cases, the 8USL UIFT EBM g-functions are more accurate than the 12ESL UIFT EBM g-functions. The 8USL discretization not only improves the accuracy compared to the 12ESL discretization, but also decreases the computation time, as seen in Figure 4.6b. The accuracy of g-functions computed with the similarities solver is increased with the use of unequal segment lengths as well, though those results are not pertinent to this work.

Table 4.1: Single U-tube borehole geometry.

Borehole Diameter (mm)	150
U-tube Inside Diameter (mm)	21.6
U-tube Outside Diameter (mm)	26.6
Shank Spacing[8](mm)	32.3

Table 4.2: Non geometric parameters input for GHE sizing.

Volumetric flow rate per borehole, $\dot{V}_b$ (L/s)	0.2
Burial Depth, H (m)	2
Ground thermal conductivity, k_s (W/(m·K))	2
Grout thermal conductivity, k_g (W/(m·K))	1
Pipe thermal conductivity, k_p (W/(m·K))	0.4
Ground volumetric heat capacity, $(\rho c_p)_s$ (kJ/(m·K))	2345.49
Grout volumetric heat capacity, $(\rho c_p)_g$ (kJ/(m·K))	3901
Pipe volumetric heat capacity, $(\rho c_p)_p$ (kJ/(m·K))	1542
Undisturbed ground temperature, T_g (°C)	18.3
Water temperature, T (°C)	20
Pipe roughness. ϵ (m)	1.00E-06
Max Heat Pump Entering Fluid Temperature, T_{max} (°C)	35
Min Heat Pump Entering Fluid Temperature, T_{min} (°C)	5
Simulation duration (years)	20



(a) Square fields timing

(b) Square fields memory

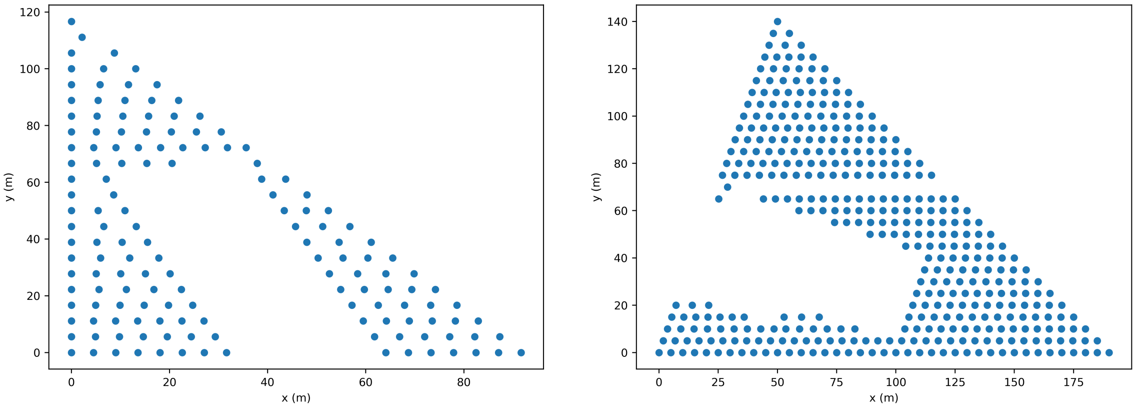
Figure 4.6: The mean percent error between g-functions computed with Eskilsons 27 logarithmic dimensionless time points for square fields ranging from 1x1 to 20x20.

The square cases are highly regular and may be easier to approximate than those that are non-uniform in spacing and have cutouts, or missing boreholes in a large area. Therefore, it's desirable to make some comparisons of accuracy for more irregular cases. An additional 10 configurations computed using a RowWise[9] borehole

[8] "Shank spacing" is the distance between the two legs of the U-tube measured from the outer tube walls. For a double U-tube, it is the distance between the two opposite tubes, again measured from the outer tube walls.

[9] The RowWise borehole placement method was developed by Timothy N. West, an undergraduate research assistant who worked under Dr. Jeffrey Spitler.

placement method, ranging from 150 to 389 boreholes, are also compared. The RowWise scheme utilizes a target spacing within each row but increases the spacing to make full use of the land area available. The land area can be described by an outer polygon, with inner polygons describing locations where boreholes cannot be placed. Take as an example, a row that stretches 18m between the property boundary and the building edge, with a user-defined target spacing of 5m. The row could hold four boreholes spaced 5m apart, if the target spacing was maintained. The RowWise scheme automatically adjusts the spacing to fully use the 18m, placing the four boreholes 6m apart. For these cases, reference g-functions are computed at four heights: 48m, 96m, 192m and 384m. The reference cases all utilize 96 equal finite line source segments along the length of the borehole. The number of segments used is limited by the memory consumption, which grows to near 760GB for the largest case. A couple of the fields used in this analysis that are computed with the RowWise generation technique are given in Figure 4.7.



(a)

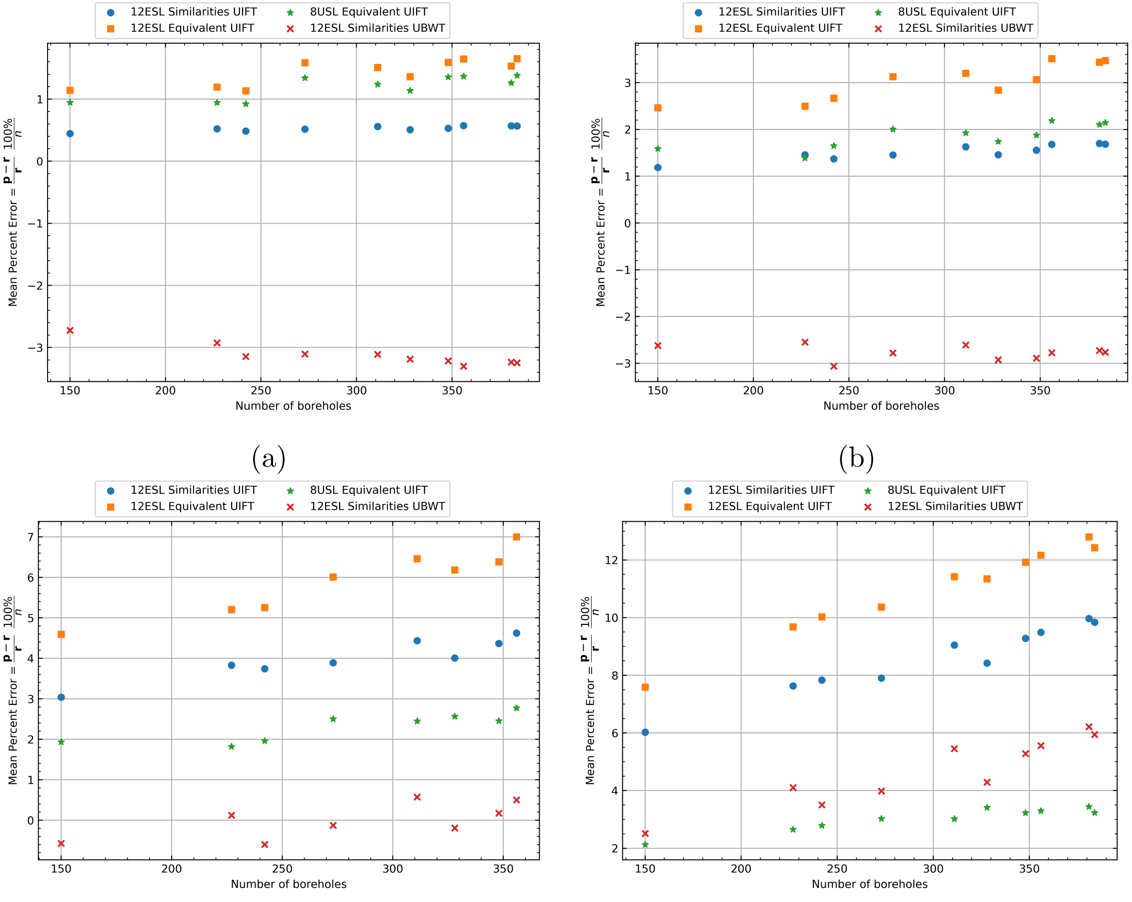
(b)

Figure 4.7: Two RowWise layouts used in verifying the equivalent borehole method, one with 150 boreholes (a) and one with 384 boreholes (b).

The mean percent error results of 8USL and 12ESL g-functions compared to the reference g-functions are given in Figure 4.8a for 48m results, Figure 4.8b for 96m results, Figure 4.8c for 192m results, and Figure 4.8d for the 384m results. These results are a bit more nuanced than for the square cases. At 48m and 92m depths, the results for the 12ESL discretization with the similarities solver are slightly better than the EBM with the 8USL discretization. At the 192m depth, the errors in the UIFT 12ESL cases shift noticeably upwards, while the 12ESL UBWT solution seemingly becomes a good approximation. At 384m, all cases have shifted upwards substantially, with the 8USL maintaining its accuracy, and proving the best approximation at this height. It is noteworthy that Eskilson considered the UBWT boundary condition to be a good approximation for a network of boreholes connected in parallel when the flow rate becomes sufficiently high. In practice the flow rate is often not sufficiently high enough for UBWT to be a good approximation.

The UIFT g-function is capable of better approximating physical reality by considering the actual flow rate. The UBWT approximation underestimates the g-function (for most cases). The UIFT approach has not been used in design tools in the past. The UIFT g-function boundary condition (Cimmino, 2015) was only recently developed, and made available in an open-source program (Cimmino, 2018b) that was recently deployed. While the more accurate UIFT method has been an available resource, the live calculation of g-functions has not been feasible. Therefore, UIFT was sidelined in favor of an approximate UBWT approach that was tabulated for lookup in a database. The EBM makes the g-function calculation fast enough to be done on the fly during the design process. That allows the preferred UIFT boundary condition to be utilized in a live g-function calculation. Using the EBM with UIFT boundary condition and the 8USL discretization gives slight overprediction (<3%), which is acceptable for design programs, and preferable to underprediction.

GHEDT depends on pygfunction and contains a function that wraps all the possible g-function calculation configurations. A user of GHEDT could specify the boundary condition used, the number of segments and whether the segments are equal or unequal along the length. The recommended g-function for calculation, that is default, is computed using the EBM with the UIFT boundary condition and the 8USL discretization.



(a)

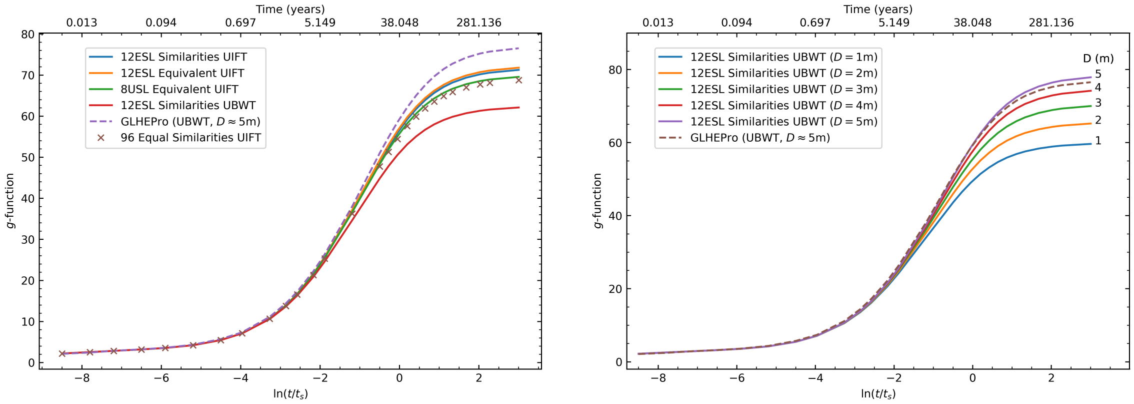
(b)

(c)

(d)

Figure 4.8: Mean percent error of various of g-function calculation methods over a range of RowWise borehole configurations at 48m (a), 96m (b), 192m (c) and 384m (d) depths compared to the reference g-functions.

Various g-functions for a 12x13 configuration with an equal borehole spacing of 5m and borehole lengths of 96m are shown in Figure 4.9a. The various boundary conditions and solver methodologies utilized result in differing g-functions. The previously described g-functions are plotted, in addition to a g-function from Hellström's library that was obtained by utilizing GLHEPRO. The red x's are the 96 equal segment length reference case. Even though Cimmino and Bernier (2014) were able to approximate Eskilson's g-functions with high accuracy using 12 equal segments, the calculated curve appears much lower than the one found in the GLHEPRO lookup tables. This can be explained by the difference in the burial depth. Eskilson (1987) stated that varying the burial depth from the surface to 5m had little impact on the g-function, and computed his g-functions for somewhere between 4-5 meters. Figure 4.9b shows the variation of the UBWT g-function with 12 equal segments over a range of burial depths D from 1 to 5m. The UBWT g-functions (computed with *pygfunction*) containing 4 and 5m burial depths bracket Eskilson's g-function.



(a)

(b)

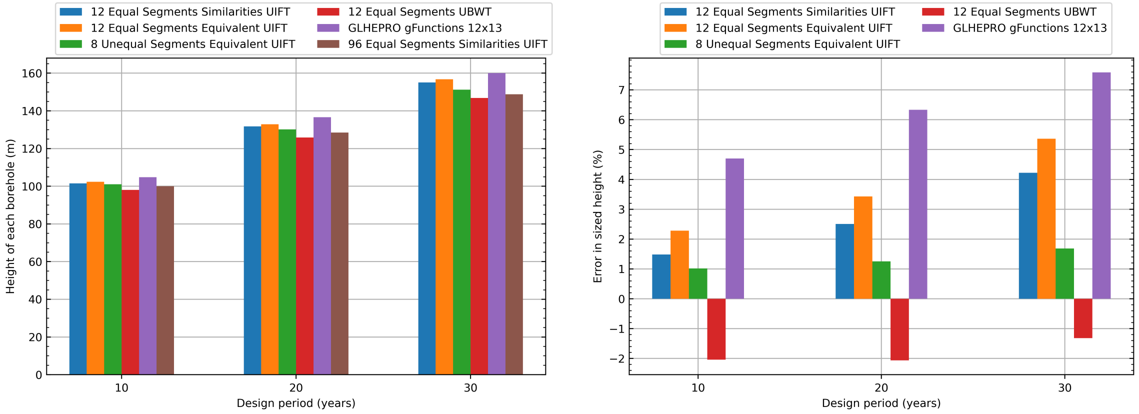
Figure 4.9: The g-functions plotted for a configuration of 12x13 with uniform spacing of 5m and a borehole height of 96 meters (a). The g-function is shown to vary with the burial depth in (b).

The comparison of the mean percent error (MPE) of the entire g-function curve, as done in Figure 4.6 and Figure 4.8 is a useful way of quantifying g-function error. However, these results don't directly correspond to the error in sizing a GJE. Eskilson's dimensionless time goes out to 3.003. The dimensionless time value of 3.003 corresponds to 767 years for the soil conductivity, soil volumetric heat capacity and height of the boreholes for this case. For a 20-30 year design life, a significant portion of the g-function that contributes to the MPE is not used in a GHE simulation. The time in years associated with this case can be seen on the top x-axis in Figure 4.9a. While quantifying the error of the g-function curve is useful for understanding overall trends, the g-function comparison plot of Figure 4.9 does not mean much by itself, without determining the impact on the GHE size.

To investigate the effects on sizing, loads from an Atlanta office building (Spitler et al., 2020) are utilized to size the configurations described above. The loads are heavily cooling-dominated, so over time more heat is put into the ground than is extracted from. In order to make the final sizing procedure (determining the depth for a specific configuration) fast, g-functions for borehole heights of 24m, 48m, 96m, 192m and 384m are pre-computed. Borehole heights are iteratively interpolated for in the sizing procedure. The sizing procedure is done for the design temperature limits given in Table 4.2, for periods of 10, 20, and 30 years. The g-functions used correspond to the g-functions plotted in Figure 4.9a.

The sizing results are presented in Figure 4.10a. As the design period is increased, the total amount of land area necessary is substantially increased. This is primarily a result of the unequal loads (cooling-dominated). The errors in sizing, when compared to the reference g-function with the converged 96 equal segment similarity case, are shown in Figure 4.10b. The sizing height determined with the GLHEPRO g-functions gives the highest sizes, due to the GLHEPRO g-functions being based on burial depths of 4-5m and the UBWT boundary condition. The g-functions calculated with the EBM, 8USL discretization, and UIFT boundary condition are the "best" – that is, they are the most accurate while also slightly overestimating the required depth rather than underestimating the required depth. The 8USL EBM g-function overestimates the required height by about 1%, 1.3% and 1.65% at 10, 20 and 30 years respectively, when compared to the reference g-function (96 equal segments with the similarity method).

The conclusion of the analysis of the accuracy of the g-functions is that the EBM with 8USL discretization and a UIFT boundary condition is the best approximation for calculating g-functions quickly. These g-functions are very accurate and properly account for borehole thermal resistance and actual burial depths.



(a) (b)

Figure 4.10: Height of boreholes (a) based on g-functions computed using a range of calculations, and errors in reference to a 1m equal segment length discretization (b).

4.3.2 Integer Bisection Search

In the design of GHEs, the acceptable drilling depth is often defined by local conditions – both the geology and available drilling rigs. For large systems with many boreholes, it is feasible to bring in other drilling rigs, but the geology may still limit the economically feasible drilling depth. The cost of moving the drilling rig to a new location must be balanced against declining rates of penetration[10] (ROP) as depths increase. Therefore, a maximum or target drilling depth is a required input for the configuration selection algorithms in GHEDT. A likely future improvement would be to develop cost models that account for moving drill rigs, ROP varying with depth, the cost of trenching, etc. For now, this information is encapsulated in a user-specified target depth. GHEDT first provides the ability to quickly determine the configuration based on a target depth and can size the configuration to determine a final actual depth. This means that designers using the tool can quickly evaluate alternative target depths, which is highly useful for GHE designers. GHEDT's methodology differs from existing design tools. GLHEPRO does not currently contain an automated methodology. EED's methodology is an exhaustive procedure, rather than selection

[10] The rate of penetration quantifies the speed a drill bit can break rock. In certain geological conditions, the speed, or rate of penetration, of the drilling may be slower.

of a single configuration.

The selection of configurations is made possible via construction of domains that, when searched over at the target height, form a discontinuous unimodal[11] excess temperature function. These are referred to as unimodal domains throughout this chapter. Points on the discontinuous unimodal function are produced by simulating borehole fields in the domain at the target height. The values of the unimodal function are excess temperature values (the simulation procedure is described in Section 3.6 and the excess temperature cost function is described in Section 3.7). The function is discontinuous because adding a borehole results in a jump in excess temperature. An integer bisection search routine was implemented to decrease the required number of borehole fields to be simulated to make a borehole field selection. After a field is selected, a size is determined by varying the depth of the boreholes in between the described target depth and minimum depth. GHEDT provides a number of search routines, with increasing degrees of complexity. The simplest search routine, covering only square and near square fields, is described in this section.

A GHE simulation requires building heating and cooling loads. The heating and cooling load profile, shown in Figure 4.11, is used for the simulation and design routine results shown in this chapter. The load profile is produced by simulating an office building[12] in EnergyPlus using weather conditions for Atlanta, GA. The building layout has a rectangular footprint of 50m in length by 33.3m in width.

A square and near-square domain is described, and the integer bisection routine is utilized to select a field. The square and near-square domain is created by

[11] The "unimodal functions" in this thesis have a single minimum over the range of interest, and the slope constantly descends from left to right. Increasing total drilling depth, while maintaining similar geometrical spacing, results in lower excess temperatures. The domains in this thesis are constructed to produce a unimodal excess temperature curve. The unimodal excess curve is created by simulating the fields in the domain at a constant height. GHEDT contains a function that can check unimodality of a domain. The required inputs to the function that checks whether a domain produces a unimodal excess temperature curve are a single height and a domain.

[12] This building is further described in (Spitler et al., 2020).

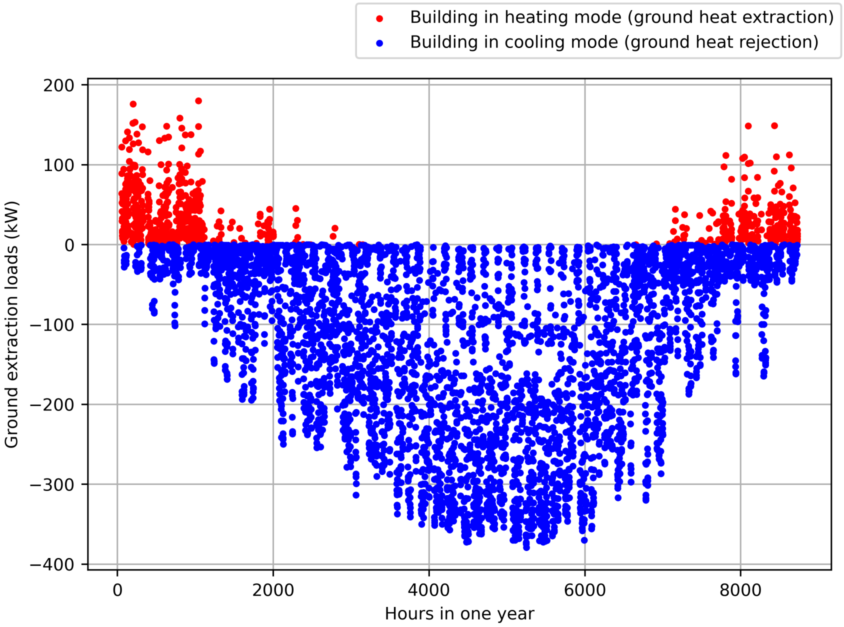


Figure 4.11: An Atlanta office building ground heat extraction hourly load profile for one year.

placing successively larger square and near-square[13] fields in order. The square and near-square domain, $\mathbb{D}$ contains configurations from one borehole to a square field containing 1024 boreholes (e.g., 1x1, 1x2, 2x2… 32x32). Creation of the square and near-square fields is given by a function $r(N_x, N_y, B_x, B_y)$, where N_x is the number of boreholes in the x-direction, N_y is the number of boreholes in the y-direction, B_x is the spacing in the x-direction and B_y is the spacing in the y-direction. Therefore, the domain, based on a user defined uniform borehole spacing, B=5m is constructed.

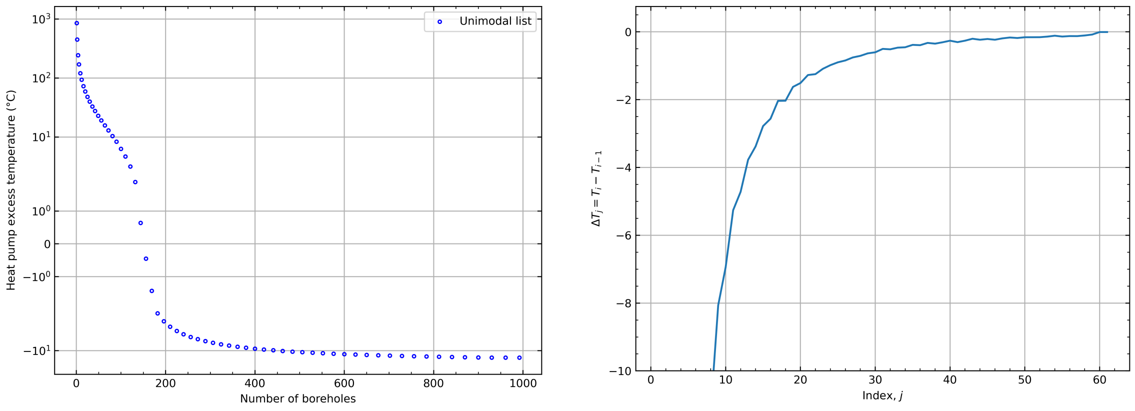
$$\mathbb{D} = \left\{ \begin{matrix} r(1,1,B,B) \\ r(1,2,B,B) \\ r(2,2,B,B) \\ \vdots \\ r(32,32,B,B) \end{matrix} \right\} \tag{4.6}$$

[13] Near-square would be a configuration with (N) x (N+1) boreholes – the number of rows and columns is different by one.

The resulting excess temperatures for each configuration in the domain, $\mathbb{D}_i$ are simulated for 20 years at the maximum user defined height, $H = 135$m, and the excess temperature is calculated based on the minimum and maximum heat pump entering fluid temperature over the simulation. Figure 4.12a shows the excess temperatures plotted based on the 20 year simulation results of the square and near-square fields. To verify the unimodality, the successive differences along the domain are computed to ensure each resulting configuration contains less excess temperature than the previous. Though this domain is easily visually verifiable, a systematic method to ensure unimodality is useful.

$$\Delta T_j = T_i - T_{i-1} \tag{4.7}$$

Figure 4.12b shows proof of unimodality, where none of the ΔT_j values are above 0. If $\Delta T_j \leq 0$ for all j, then the discontinuous excess temperature function is unimodal. Proof of unimodality, or a smooth discontinuous excess temperature function, is important for the integer bisection search, described later in this section.



(a)

(b)

Figure 4.12: A unimodal function of excess temperatures produced by square and near-square fields (a) and proof that the discontinuous excess temperature function produced is unimodal (b).

The configurations are simulated at the target height, so that when a field is selected, the height can then be decreased to drive the excess temperature to 0.

Because an excess temperature of 0 is desired, the configuration that results in an excess temperature closest to 0, but negative is the configuration of interest. If the user were to set the minimum height very near the maximum height, it's possible that the two heights would not bracket the root of the excess temperature. The selected configuration would not be able to meet the minimum height limit. This could be remedied by decreasing the minimum height, or possibly adjusting the borehole spacing.

To accelerate the selection process, in addition to only simulating the fields at the target height, an integer bisection search is implemented. The integer bisection search starts by simulating the first configuration, $S(\mathbb{D}_0, H_{min})$ at the minimum height and maximum height, $T_0 = S(\mathbb{D}_0, H_{max})$. Where S a function that performs a 20-year hybrid time step simulation and returns the heat pump excess temperature. If the sign of the resulting simulations are opposite one another, then the first configuration can be sized between the specified height range. The last value in the domain is simulated, $T_{n-1} = S(\mathbb{D}_{n-1}, H_{max})$. If the signs of T_0 and T_{n-1} are opposite one another, then the integer bisection search can be performed, because the solution is bracketed. If both the resulting excess temperatures are less than 0, then a warning message is displayed that informs the user that either the loads are unrealistically small, or the configuration in the first domain, $\mathbb{D}_0$ needs to contain less boreholes. If both the resulting excess temperatures are above 0, then a warning message is displayed that informs the user that the loads are either excessively large, or the last configuration in the domain, $\mathbb{D}_{n-1}$ should contain more boreholes. The function being searched over is a discontinuous function, $S(\mathbb{D}_i, H_{max})$ with i on the interval $[a, b]$, where $a = 0$ initially and $b = n - 1$, where n is the number of elements in $\mathbb{D}$. Each iteration performs the following steps:

- Calculate c, the integer midpoint of the indexing interval, $c = ceil(\dfrac{a+b}{2})$
- Calculate the excess temperature, $T_c = S(\mathbb{D}_c, H_{max})$

- Convergence is satisfied if $c = a$ or $c = b$
- Examine the sign of T_c, and if the sign is equal to the sign of T_a, then $a = c$, otherwise $b = c$.

The result of the integer bisection search are two values, T_a and T_b where $b = a + 1$, and the sign of T_a does not equal the sign of T_b. The field that corresponds to $T_b = S(\mathbb{D}_b, H_{max})$ is selected; the final depth will be determined in the next step, sizing. This selected field contains the closest minimum excess temperature to 0 of the configurations in the domain. The sizing function then decreases the depth (which increases the excess temperature) from the target depth to drive the excess temperature to 0. Once a configuration is selected, g-functions are computed at the target depth, minimum depth, and the average of the two. The depth is then found by interpolating the g-functions so that the excess temperature is driven to 0.

A maximum depth of 135m and a minimum depth of 60m is used to select a square or near square design with other inputs described in Table 4.1 and Table 4.2. The entire domain is plotted in Figure 4.13, and the points that are simulated during the bisection search routine are marked in red. The numbers marking the simulated fields are labeled in the order they were simulated during the integer bisection search. A zoomed in portion of the plot is shown, so that it can be easily seen that the last field simulated, marked number 8, is the field with excess temperature closest to 0, but negative. The amount of time it takes to select a field is about 16 seconds on a desktop PC with an 11th Gen Intel Core i9-11900K at 3.50GHz. The time to compute the three g-functions for interpolation and then size takes around 4 seconds. Therefore, using the 8 unequal segment EBM g-function calculation method and the bisection integer search routine, a square or near square field ranging from 1 to 1024 boreholes can be selected and sized in about 20 seconds. The maximum height was described so that a borefield containing a 12x13 layout would be selected, so that the results can be compared to what was previously found in Figure 4.10a; a way of validating the integer bisection search results. The sizing result here is 130.1818m, where the result given in Figure 4.10a for the same field is 130.1339. This 0.036% difference in sizing height is due to the difference in the number of g-functions used in the interpolation.

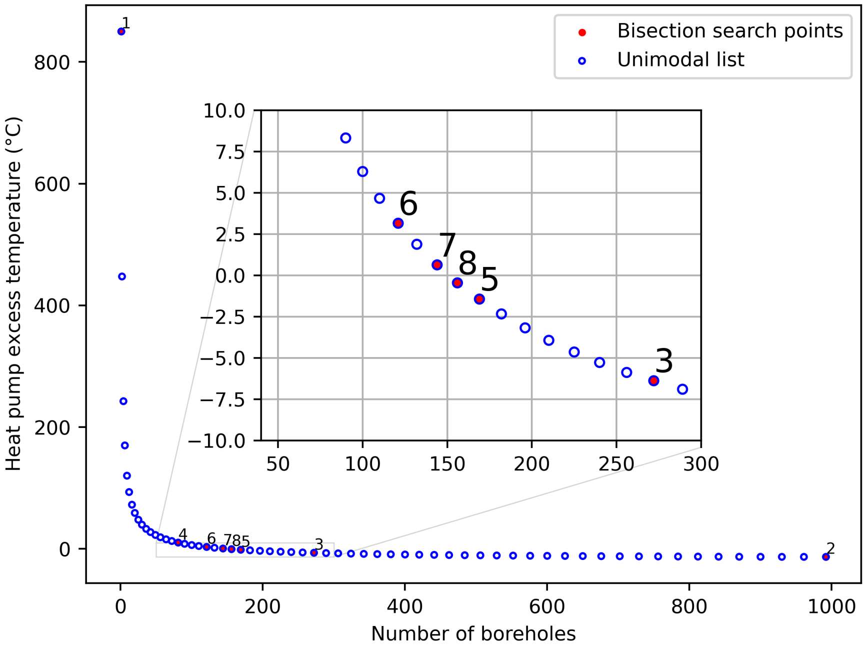


Figure 4.13: The square and near square search domain plotted with the integer bisection search simulation points plotted.

4.4 Design Routines

4.4.1 Uniform Constrained Rectangular Search

The previous example showed the sizing and selection of an unconstrained square field with a target depth of 135m. However, in practice square designs may not be desirable as they are more prone to thermal buildup, and the design will more than likely be constrained by land area. A common borehole layout for a GHE is one that is rectangular. For example purposes, the Atlanta office building with a 50m by 33.3 m footprint is considered to be on a lot 85m by 80m, and the bottom left corner of the building is shifted 15m right and 36.5m up from the bottom left corner of the property, as shown in Figure 4.14. This leaves a rectangular area (below the building on the figure) that can be used for a rectangular borehole field.

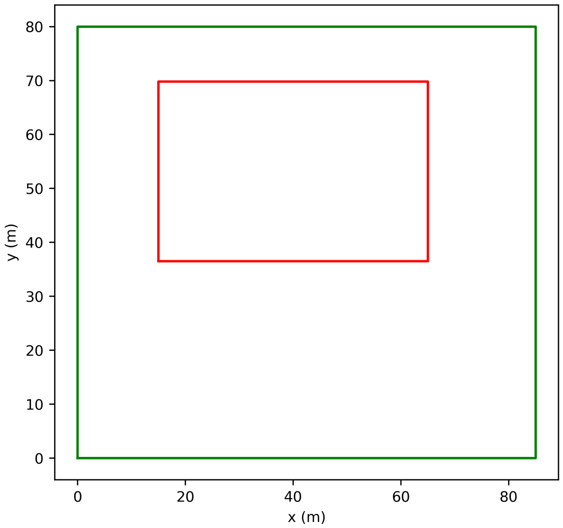


Figure 4.14: A rectangular land description with an unavailable no-drill zone where a rectangular building is located.

A search routine has been developed that takes in the area (length x width) constraints and a minimum and maximum uniform borehole spacing. The GHE designed with this routine will closely resemble what a GHE designer has had the ability to come up with by iteratively adjusting the number of boreholes in the x and y directions, varying the borehole spacing and then checking the design with the use of g-functions in Hellström's library.

The search procedure is similar to the square and near-square search described in the previous section. First, a domain of rectangles, that when simulated at the maximum height result in a unimodal discontinuous function, is created. The first step in this process is to determine a minimum number, N_{min} and maximum number, N_{max} of boreholes using the length of the longest side (L_x) and the spacing constraints, B_{min} and B_{max}, supplied:

$$N_{max} = \frac{L}{B_{min}} + 1 \tag{4.8}$$

$$N_{min} = \frac{L}{B_{max}} + 1 \tag{4.9}$$

However, the number of boreholes found in these equations, for most cases, will be floating point numbers. Therefore, the minimum number of boreholes is rounded up (using a ceil function) to the nearest integer, and the maximum is rounded down (using a floor function) to the nearest integer. The number of boreholes used in the longest direction then falls between $N_{min} \leq N \leq N_{max}$, where $N_0 = N_{min}$ and $N_{n-1} = N_{max}$. The uniform B spacing utilized is one that ensures the land is fully used in the direction of longest specified length. The number of boreholes then used for the shorter length is rounded down to the nearest integer and based on the uniform B spacing. Though for every number of boreholes in the x-direction, N_i, multiple rectangles could have the same number of boreholes in the y-direction . For example, if there are 10 boreholes in the x-direction, there might be multiple rectangles that have, say, 5 boreholes in the y-direction. So, then there could be multiple 10x5 rectangles with slightly longer uniform borehole spacing. This would cause the discontinuous excess function to creep up slightly until another row of boreholes could fit in the y-direction. Therefore, multiple fields containing the same number of boreholes in the y-direction are disallowed from the domain. A lower end of the domain is added in case a user defines much more land area than necessary. The lower end steps from one borehole to a line of boreholes at the maximum spacing based on the minimum number of boreholes, and then rows are added in the vertical direction, maintaining a uniform maximum borehole spacing. The rectangles forming uniform rectangular domain for this specific case are shown in Table 4.3.

A discontinuous unimodal function of excess temperature can be created by simulating each of the rectangles in the uniform rectangular domain. The domain in its entirety, as well as the fields selected for simulation during the integer bisection search is shown in Figure 4.15a. The proof of unimodality for the uniform spaced constrained rectangle domain is shown in Figure 4.15b. The large jump at index of 10 is when the domain goes from a single line to two lines. This proof of unimodality

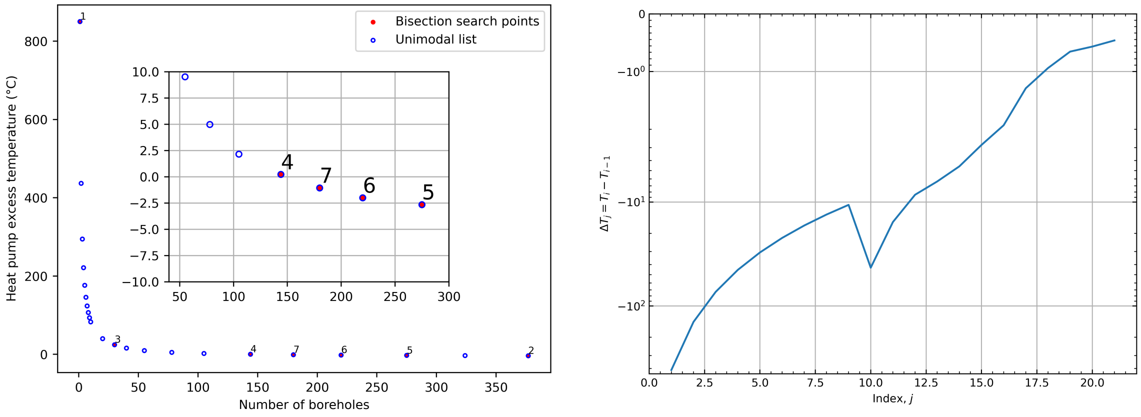
Table 4.3: The uniformly spaced rectangular domain, for the inputs previously described.

Index	Nx	Ny	B
0	1	1	9.44
1	2	1	9.44
2	3	1	9.44
3	4	1	9.44
4	5	1	9.44
5	6	1	9.44
6	7	1	9.44
7	8	1	9.44
8	9	1	9.44
9	10	1	9.44
10	10	2	9.44
11	10	3	9.44
12	10	4	9.44
13	11	5	8.50
14	13	6	7.08
15	15	7	6.07
16	18	8	5.00
17	20	9	4.47
18	22	10	4.05
19	25	11	3.54
20	27	12	3.27
21	29	13	3.04

is not limited to this case, the method by which the domains are constructed will be unimodal no matter the land or spacing constraints.

For the example scenario, a borehole field that contains 20 boreholes in the x direction and 9 boreholes in the y direction with a uniform spacing of 4.47 meters is selected by the integer bisection search (marked "7" in Figure 4.15a). The target depth used is 135m. The uniform spaced rectangular field is shown in Figure 4.16. The time to perform this search on a desktop PC is around 16 seconds. The sized height is 123.13m, which results in a total drilling depth of 22,164 m.

The rectangular design routine is both powerful and a novel development for GHE designers to be able to quickly determine a uniformly spaced rectangular layout with land area, height and uniform borehole spacing constraints. The uniform layout



(a) (b)

Figure 4.15: The unimodal excess temperature function and the bisection search points (a) for the uniform rectangular domain, and the proof of unimodality (b).

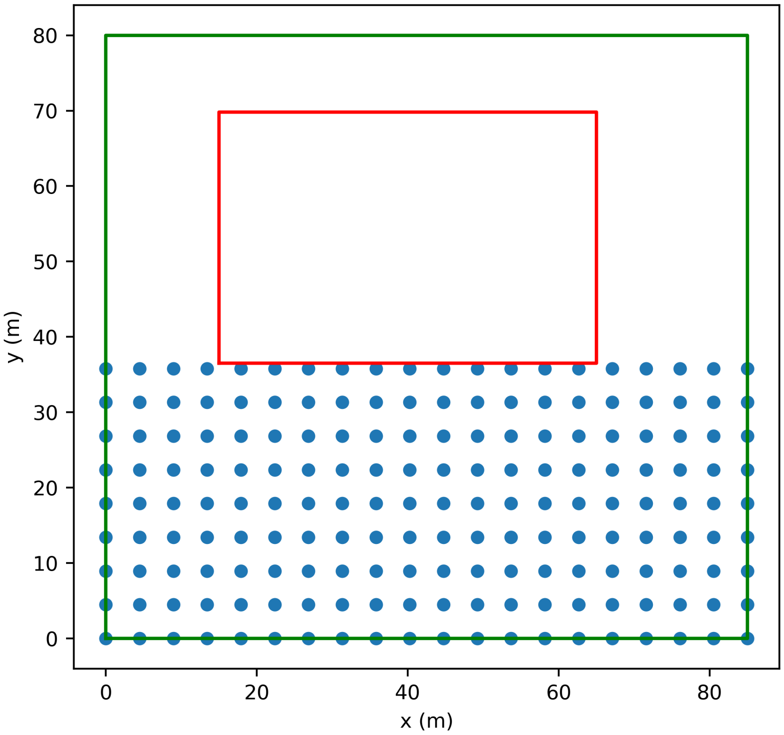


Figure 4.16: A borefield with a 20x9 layout and a uniform spacing of 4.47 meters found by searching a rectangular domain consisting of uniform borehole spacings.

cannot always fully make use of the land area; therefore, the next section will present a bi-uniform constrained rectangular search routine, which can select a rectangle with different spacing in the x and y directions, with an additional input of maximum spacing in the direction of the shorter length.

4.4.2 Bi-Uniform Constrained Rectangular Search

The Bi-uniform search requires one additional input when compared to the rectangular search routine: a maximum borehole spacing in the direction of shortest length, N_y. For the remainder of this report, the x direction will be considered the long direction, and the y-direction the short direction, a line of boreholes in the x-direction to be a row and a line of boreholes in the y-direction to be a column. If the maximum spacing in the y direction was unconstrained, the solution for problems with plenty of land area would be to space the rows of boreholes as far apart as possible, until the thermal response between the rows were independent of each other, and the solution of the g-function became like a single line. However, the further apart boreholes are spaced, the more head loss in the pipe and the higher the required pumping power. Additionally, the heat transfer of the horizontal piping would need to be accounted for if it became a significant enough portion of the network. Therefore, the designer is given the ability to specify a land area, minimum total borehole spacing, maximum borehole spacing in the x-direction, maximum spacing in the y-direction and height constraints.

The function that creates a bi-rectangular domain takes in the inputs described above. The routine is similar to the rectangular domain generator, though this time the spacing in the y-direction is set as what is input as the maximum y-spacing. The number of the boreholes used in the y-direction are found by rounding Equation 4.9 up to the nearest integer value. This is done because the least number of boreholes that fit along the y-direction is desired. For example, a bi-uniform domain with L_x = 85m, L_y = 36.5m a minimum borehole spacing of B_{min} = 4.45m, a maximum borehole spacing along the x-direction $B_{x,max}$ = 10m and a maximum borehole spacing along the y-direction $B_{y,max}$ = 12m provides the following domain, shown as a column vector.

$$\mathbb{D} = \left\{ \begin{matrix} r(1,1) \\ r(1,2,9.4,9.1) \\ \vdots \\ r(10,1,9.4,9.1) \\ \vdots \\ r(10,5,9.4,9.1) \\ r(11,5,8.5,9.1) \\ \vdots \\ r(20,5,4.5,9.1) \end{matrix} \right\} \tag{4.10}$$

However, this domain uses a fixed number of rows in the y-direction, based on a specified maximum borehole spacing in the y-direction. A designer could conceivably manually (and iteratively) adjust the maximum y-spacing, run a bisection search on the domain created, record the size, and then adjust the y-spacing again until discovering the bi-uniform rectangular field that produces the minimum drilling. For fields that are much longer in the x-direction, than the y-direction, adding an additional row in the y-direction causes a large jump in the excess temperature. Therefore, the goal is to find the minimum number of boreholes in the y-direction first (a course adjustment), then fine tune the number of boreholes in the x-direction (creates columns along the shorter direction). To automate what could be done by hand, the bi-uniform domains, where the number of boreholes in the y-direction are held constant while the number of boreholes in the x-direction are increased, are searched over by only looking at the last value in each domain. The bi-rectangular domain selected to be fine tuned searched over, is the domain where the last configuration (the one with the most excess temperature) is closest to 0, but negative. The methodology then, is that the integer bisection search is twice employed.

To solve the problem, a nested domain is created. The nested domain, denoted $\mathbb{N}$, is shown in Equation 4-10 as a matrix of horizontally stacked bi-rectangular column vector domains. The matrix is specific to the inputs described.

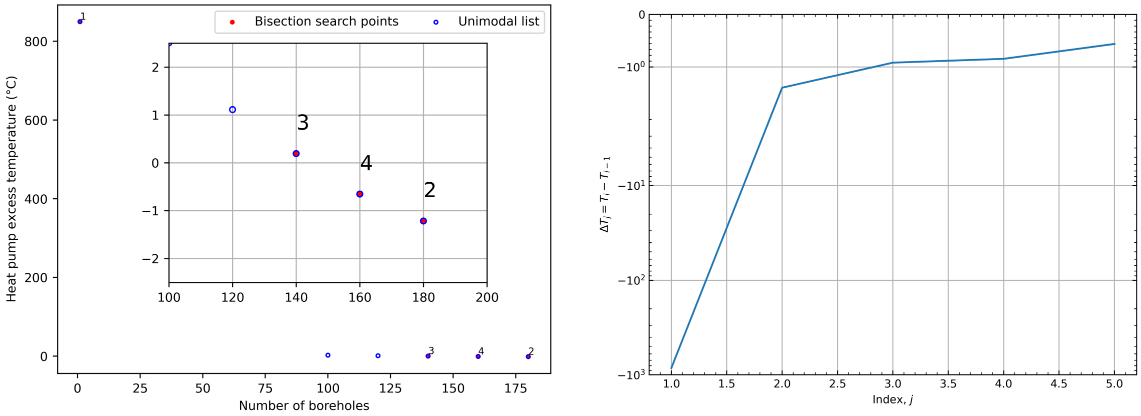
$$\mathbb{N} = \begin{Bmatrix} r(1,1) & r(1,1) & & r(1,1) \\ r(2,1,9.4,9.1) & r(2,1,9.4,7.3) & & r(2,1,9.4,4.6) \\ \vdots & \vdots & & \vdots \\ r(10,1,9.4,9.1) & r(10,1,9.4,7.3) & & r(10,1,9.4,4.6) \\ \vdots & \vdots & \cdots & \vdots \\ r(10,5,9.4,9.1) & r(10,6,9.4,7.3) & & r(10,9,9.4,4.6) \\ r(11,5,8.5,9.1) & r(11,6,8.5,7.3) & & r(11,9,8.5,4.6) \\ r(20,5,4.5,9.1) & r(20,6,4.5,7.3) & & r(20,9,4.5,4.6) \end{Bmatrix} \tag{4.11}$$

The first bisection search is performed on an “outer domain” that is constructed by taking the last configuration in each domain. A field containing one borehole is inserted in the first location to provide a high heat pump excess temperature.

$$\mathbb{N}_{n-1} = \begin{Bmatrix} r(1,1) \\ r(20,5,4.5,9.1) \\ r(20,6,4.5,7.3) \\ r(20,7,4.5,6.1) \\ r(20,8,4.5,5.2) \\ r(20,9,4.5,4.6) \end{Bmatrix} \tag{4.12}$$

This outer search successively increases the number of rows. For outer land constraints where $L_x/L_y \gg 1$, the jumps in excess temperature by adding on another row is going to result in larger jumps between excess temperatures, compared to adding another column. Therefore, the outer domain search performs the process of locating the desired number of boreholes in the y-direction, N_y and the inner search then varies the number of boreholes in the x-direction, N_x.

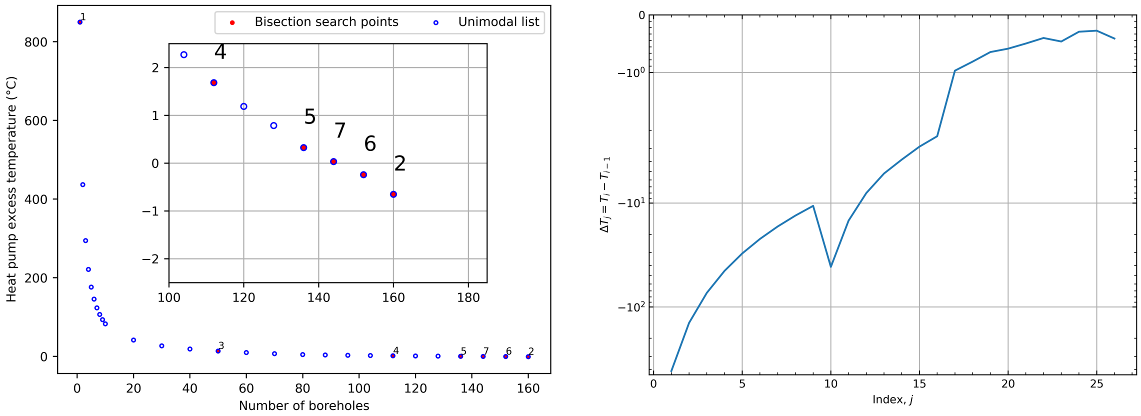
The outer search is shown in Figure 4.17a, and pertains to the domain given in Equation 4.12. Proof can be seen in Figure 4.17b. The configuration selected in the outer search contains 160 boreholes, and is the rectangle $r(20, 8, 4.5, 5.2)$, and is of the domain located in the second to last column, $\mathbb{N}_{:,n-2}$. The configuration contains an excess temperature closest to 0, but negative. Then the bisection search is employed on the selected domain. The integer bisection search for this inner domain is given in Figure 4.18a and the proof of unimodality is given in Figure 4.18b. The largest configuration contains 160 boreholes in the inner search. The result is that one column can be reduced. The rectangle chosen is defined as $r(19, 8, 4.72, 5.2)$.



(a)

(b)

Figure 4.17: The outer domain bisection search (a) and its proof of unimodality (b).



(a)

(b)

Figure 4.18: The inner domain bisection search (a) and its proof of unimodality (b).

The resulting field for the bi-rectangular search routine is given in Figure 4.19 and is determined in 24 seconds on a desktop PC. The borehole arrangement is a 19x8 configuration, equaling 152 boreholes, and contains an x-spacing of 4.72m and a y-spacing of 5.21m, with a required height of boreholes at 132.29m. The resulting total drilling depth to meet the heating and cooling demands is 20,108 m. Comparing this bi-rectangular case to the base case, there is a 9.28% saving in drilling depth.

The subsection 4.4.6 Additional Examples section provides a discussion of how the search algorithms work when the land area described is much more than necessary to meet the building loads.

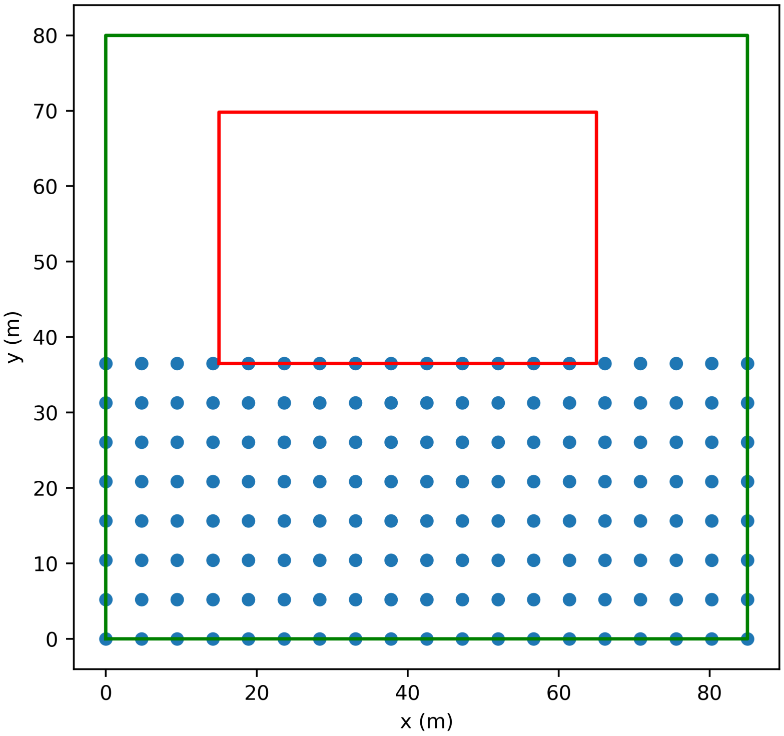


Figure 4.19: A borefield with a 19x8 layout and a bi-uniform spacing of 4.72m in the x-direction and 5.21m in the y-direction found by utilizing the bi-rectangular search routine.

4.4.3 Bi-Uniform Constrained Zoned Rectangular Search

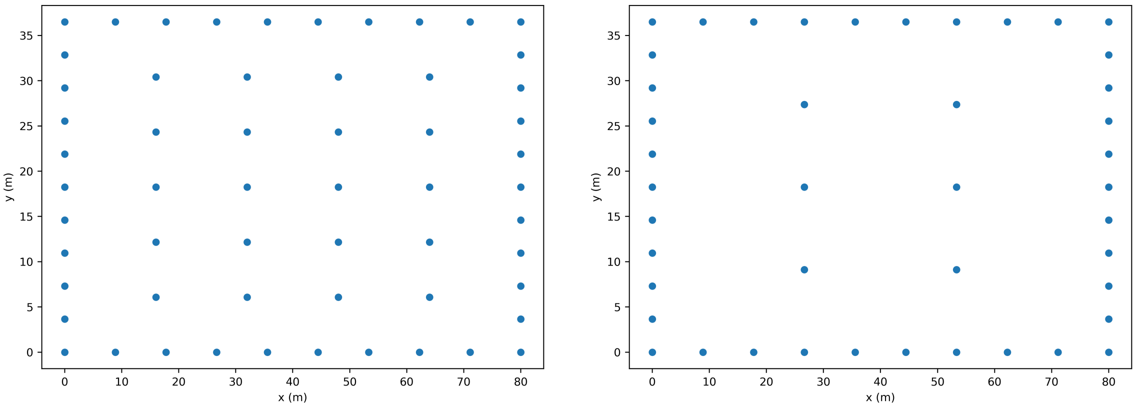
The inner boreholes in large rectangular fields can become thermally saturated over time if the annual heat rejection/extraction loads are significantly imbalanced. The ground in contact with the inner boreholes can reach a state where these boreholes are mostly ineffective, and most of the heat (rejection or extraction) is exchanged with the surrounding ground via the perimeter boreholes. Therefore, to decrease the disparity between boreholes, the concept of a zoned rectangle (Spitler et al., 2020) was conceived. A zoned rectangular borehole field consists of two zones: an open rectangle around the perimeter, and an inner rectangle with increased borehole spacing compared to the perimeter. A zoned rectangle is described by the number of boreholes in the x and y directions for the perimeter, the perimeter x and y borehole spacing, and the inner number of boreholes in the x-direction, $N_{i,x}$ and the inner number of boreholes in the y-direction, $N_{i,y}$ directions. The inner spacing for the x, $B_{i,x}$ and y, $B_{i,y}$ are determined by:

$$B_{i,x} = \frac{(N_x - 1)B_x}{(N_{i,x} + 1)} \tag{4.13}$$

$$B_{i,y} = \frac{(N_y - 1)B_y}{(N_{i,y} + 1)} \tag{4.14}$$

The origin of the inner rectangle is placed at coordinate $(B_{i,x}, B_{i,y})$ from the origin of the outer open rectangle. An example for a zoned rectangle with $N_x = 10$ boreholes, $N_y = 11$ boreholes, a length in the x-direction, $L_x = 80$m and a length in the y-direction, $L_y = 36.5$m is shown in Figure 4.20, where the figures contain an inner rectangle ($N_{i,x} x N_{i,y}$) of 4x5 (Figure 4.20a) and 2x3 (Figure 4.20b).

A domain for a zoned rectangle begins with an open rectangle with one borehole in the center, and works its way up to a bi-uniform rectangle. The decision to increase a row or column in the inner rectangle is based on which reduction will keep the inner ratio $B_{i,x}/B_{i,y}$ closest to the outer borehole spacing ratio, B_x/B_y. The goal is to maintain an inner ratio that is similar to the outer borehole ratio. That is, not all possible combinations of row and column dimensions in the interior zone are considered in the domain.

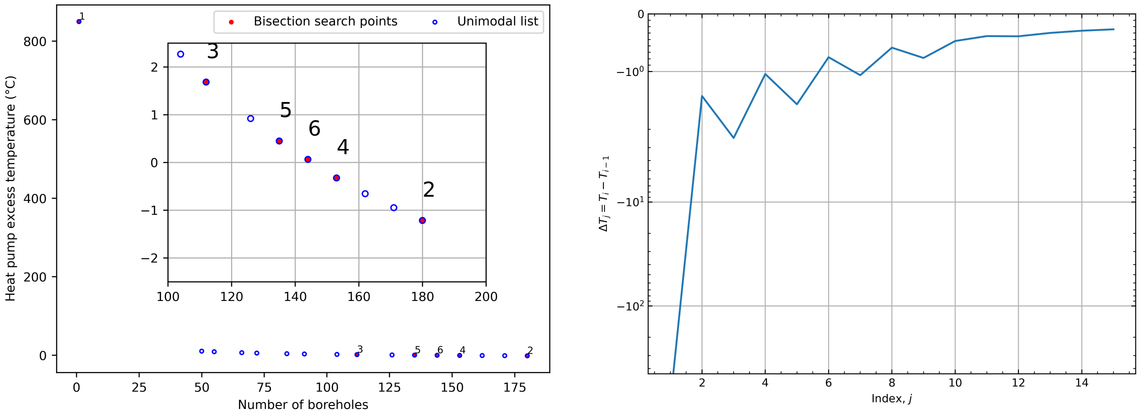


(a) (b)

Figure 4.20: Bi-uniform zoned rectangles with an inner rectangle of 4x5 (a) and an inner rectangle of 2x3 (b).

Given the number of boreholes in the x- and y-directions along the perimeter, and the x- and y- spacing, a unimodal domain can be created with the stepwise reduction process. However, this leaves open the question of what the best design is for the perimeter. Therefore, a nested search is utilized for this process. The inputs necessary to create a nested zoned rectangular domain are the same as those described for the bi-uniform constrained case: L_x,L_y,B_{min},$B_{x,max}$,$B_{y,max}$. A list of unimodal domains is created by incrementing the number of boreholes in the x and y directions iteratively, until there is room to increase only one dimension. For example, if the x dimension, based on user specified inputs, can range from 5-8 boreholes in the x direction and 5-6 boreholes in the y-direction, the zoned rectangle domains created will have outer dimensions of 5x5, 5x6, 6x6, 7x6, 8x6. The nested domain bisection search can then be employed on the nested zoned domain. The outer search is given in Figure 4.21a, and proof of unimodality Figure 4.21b. The sharp jagged edges in Figure 4.21b imply large jumps in excess temperature but is not an issue for the design process, given that each of the domains cover such a large magnitude of excess temperature.

However, upon searching the selected zoned rectangular domain in Figure 4.22, the largest field in the domain is selected (marked with a “2”). It is understood that thermal saturation in the middle of the borefield does occur, and the goal of this design



(a)

(b)

Figure 4.21: The outer domain zoned rectangle search (a) and proof of unimodality (b).

routine is to procure a zoned rectangle, the conclusion can be drawn that the nested bisection routine utilized to in the bi-uniform rectangular design process needs an adjustment for this zoned rectangular search. Though, this outer search does provide a good first guess of where to start looking. The excess temperatures that are below 0

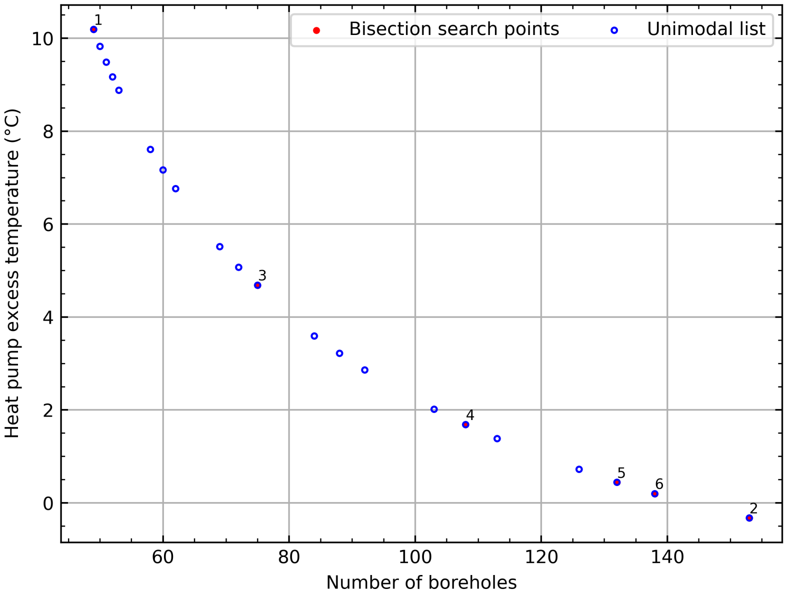


Figure 4.22: The inner domain search based on the field selected in Figure 4.21a.

in the outer zoned search shown in Figure 4.21a are at points -0.32, -0.56, -0.94, -1.215, where points a, b, c and d are respectively plotted in Figure 4.23 along the x-direction with their excess temperatures. The y-magnitude of the points in Figure 4.23 is the sizing result of the field selected with the inner domain integer bisection search. The result is that diverging further from an excess temperature of 0 is more ideal, to an extent. Moving boreholes to the perimeter and decreasing the number of boreholes in the interior is an improvement, up to a certain point. Therefore, the bi-uniform zoned rectangular search first performs an outer domain search, then successively solves for heights in the direction of negative excess temperature for the outer domain for as long as the total depth of the borefield is decreasing, or a maximum iteration of 7 is hit. Each of these exhaustive searches employs a bisection search to select a field in the domain, and then sizes the field to determine the sizing result. The time to perform the nested bisection search on a desktop PC is 95 seconds. The additional time when compared to the bi-uniform rectangular search is the additional successive search and sizing procedures. Each additional successive search raises the calculation time by a scalar.

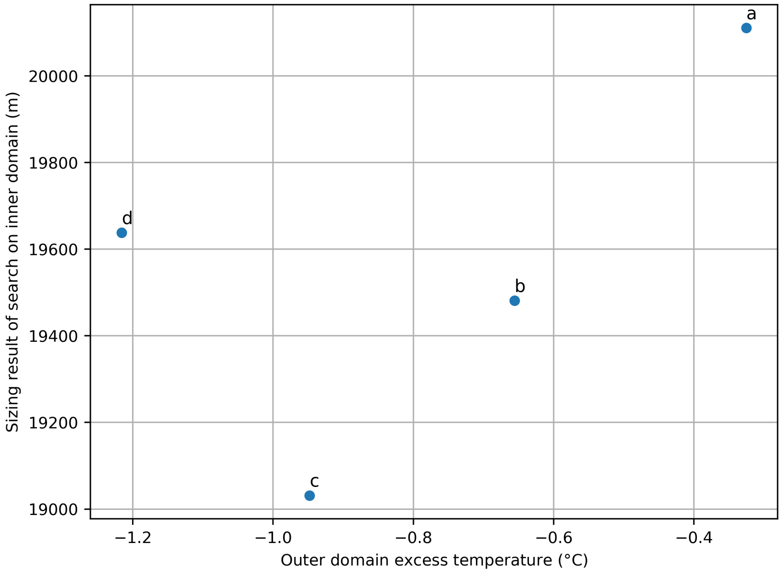


Figure 4.23: The resulting size of the search over each outer domain result that has a negative excess temperature.

The result of the bi-uniform zoned rectangular search, shown in Figure 4.24 is a zoned rectangle that contains 142 boreholes (the sizing result is marked “c” in

Figure 4.23). The outer dimensions are 19x9 with 4.72m x 4.56m spacing and the inner dimensions are 15x6 with 5.31m x 5.21m spacing. The drilling depth for each borehole is found to be 134.02m, with a total drilling depth of 19,031.5 m. This provides an additional saving of 5.3% when compared to the bi-uniform rectangular search, and a total of 14.15% when compared to the base case.

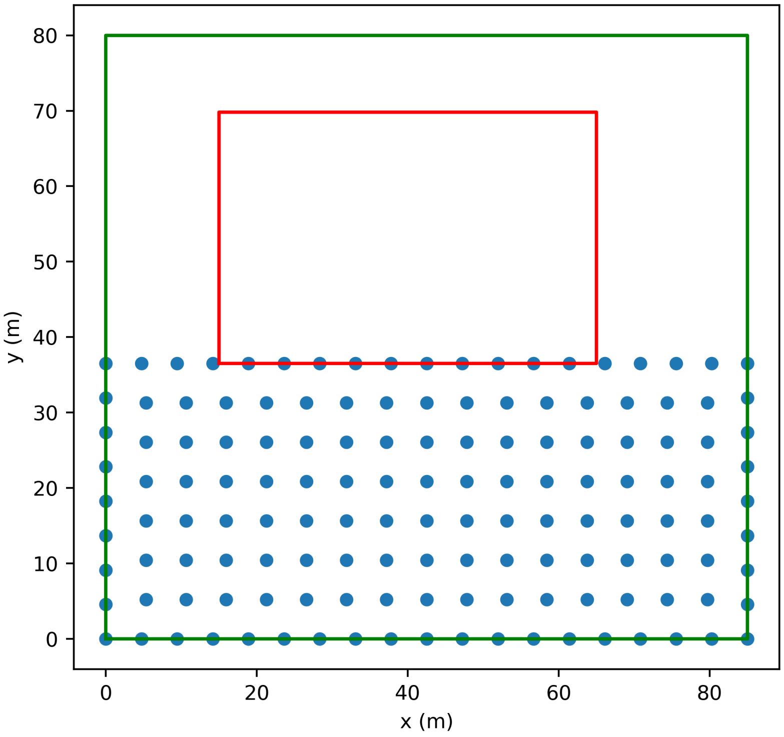


Figure 4.24: Bi-rectangular zoned case found with the bi-uniform zoned rectangle search routine.

4.4.4 Bi-Uniform Constrained Rectangular Search with No-Drilling Zones

The bi-uniform rectangular and zoned rectangular searches are constrained to a subset of the land area, when a placement algorithm that could utilize the whole land area would be a more ideal solution. E.g., as shown (Spitler et al., 2020) earlier, a borehole field that wraps around the building can be a considerably better solution than a rectangular field on one side of the building. One can imagine an optimization that adjusts the position of every borehole, but this is unlikely to be computationally feasible with larger borehole fields. Furthermore, when considering the optimization of borehole placement, there is a certain level of practicality that must be considered. For the results to be applicable to real world installations, the borehole locations should be able to be marked by a human[14], and the piping network not overly complicated. With that idea in mind, the bi-uniform constrained rectangular search is utilized now for the whole land area, but boreholes that fall within a specified polygonal zone of no-drilling are removed. This maintains the borehole placement in rows, but still can give significant improvement in the design. The result, show in Figure 4.25, is a field containing 91 boreholes with x-spacing, $B_x = 6.53$m and y-spacing, $B_y = 11.42$m with no boreholes placed inside of the no-drilling zone. The design, determined in 22 seconds, has a required drilling depth of each borehole at 127.62m. The total drilling depth found is therefore 11,613.7 m. This results in a savings of 47.6% when compared to the base case.

Table 4.4 shows the various design procedures and their percent savings compared to the base case. For the same amount of land area, the bi-zoned search proves to be a better solution than the bi-rectangular, when compared to the base case. A design algorithm that can consider "no-drilling zones" results in nearly half the total drilling depth.

Table 4.4: Comparison of design routine sizing to a typical design with uniform spacing in the x and y-direction, otherwise referred to as the "base case".

Design Routine	Shown in	Number of boreholes	Height per borehole (m)	Total Drilling (m)	Savings to base case (%)
Rectangular	Figure 4.16	180	123.13	22164.3	0.00%
Bi-Rectangular	Figure 4.19	152	132.29	20108.08	-9.30%
Bi-Zoned	Figure 4.24	142	134.02	19031	-14.15%
Bi-Rectangular w No-Drill Zones	Figure 4.25	91	127.62	11613.42	-47.60%

An additional example is created with an L shaped building that can be described

[14] Perhaps drones might be used in the future to mark the borehole locations in more geometrically complex designs.

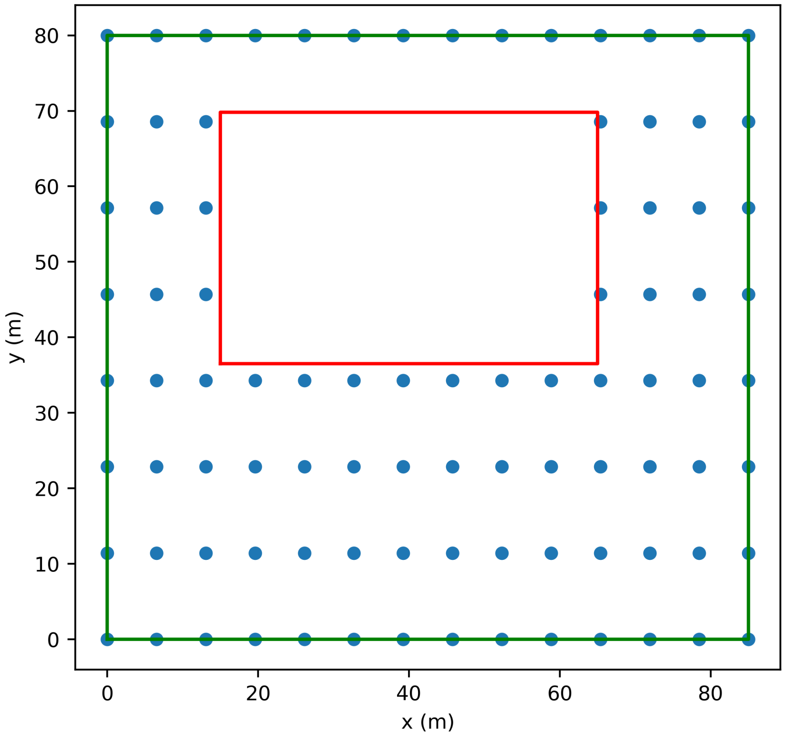


Figure 4.25: A field determined with the bi-uniform constrained rectangular search with no-drilling zones.

by 5 coordinates is now utilized. The building is also rotated by 45 degrees and then translated so that the building is located in the first quadrant. The unrotated, as well as the final rotated and translated building coordinates are given in Table 4.5. The total building area footprint is 1664 m^2, which is about the same footprint area as the original Atlanta office building. The outer building land area is set to be 68x100 m^2 so that the total amount of drilling area is similar to that of the previous examples.

Table 4.5: L-shaped building unrotated and rotated and translated coordinates.

	Unrotated		Rotated and Translated	
Points	x	y	x	y
1	0	0	1	50
2	60	0	43.43	7.57
3	60	16	54.74	18.89
4	16	16	23.63	50.00
5	16	60	54.74	81.11
6	0	60	43.43	92.43

The bi-uniform rectangular design routine with no-drilling zones selected the borefield shown in Figure 4.26 in 28 seconds. The field contains 90 boreholes, with an x-spacing of 9.7m and a y-spacing of 7.69m. The height of each of the boreholes is sized to be 126.45 m, totaling 11,380.5 m. These results are comparable to those shown in Figure 4.24, though the drilling depth is less because the footprint of the building is more spread out (less dense). The large rectangular footprint of the original Atlanta office building doesn't allow as much heat transfer below the building, where the L shaped building is skinnier and more of the land area can be utilized.

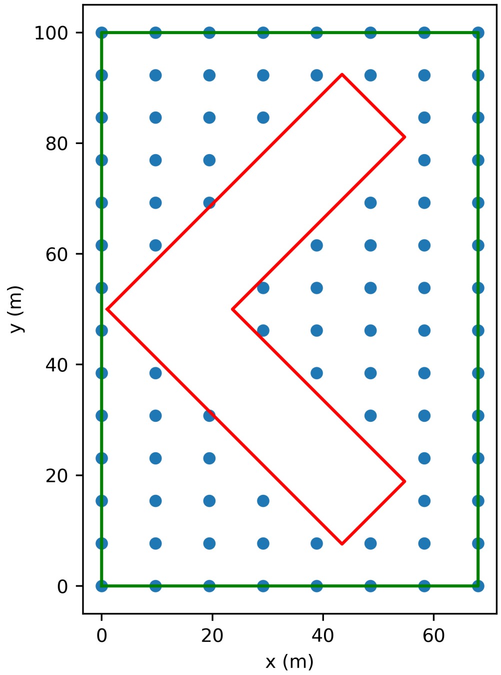


Figure 4.26: A borefield determined by the bi-uniform constrained rectangular design where the no-drilling zone is a polygon.

4.4.5 Bi-Uniform Polygonal Constrained Rectangular Search with No-Drilling Zones

The outer boundary describing the land area will not always be a rectangle, therefore the bi-uniform rectangular search with no-drilling zones is extended to handle polygonal land constraints. The L-shaped building is used again, though this time is shifted

10m to the right. The outer coordinates describing the polygonal land constraint is given in Table 4.6. The total available land area is maintained at 6800 m^2.

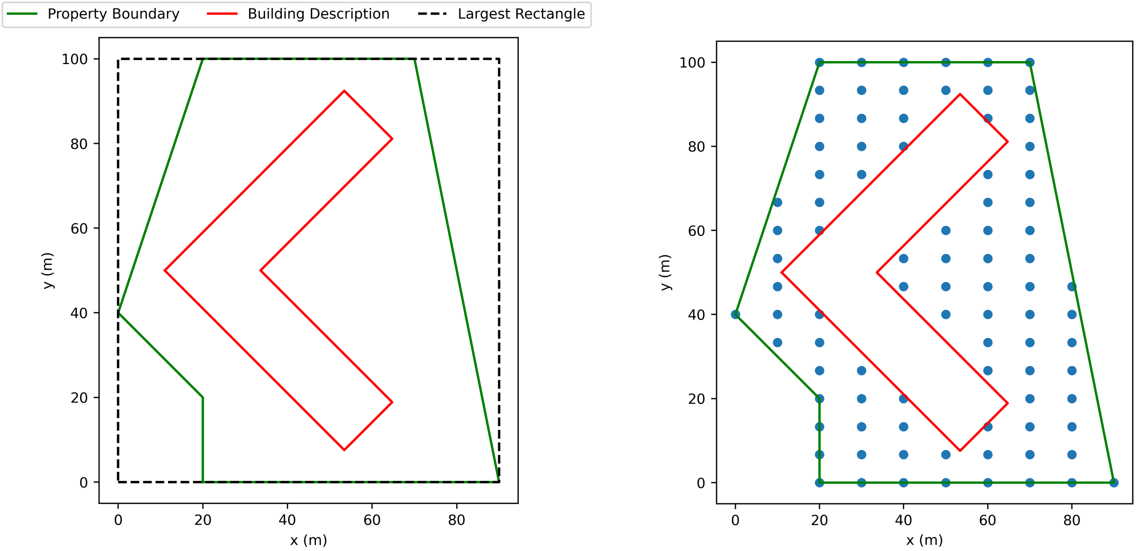
Table 4.6: Outer polygonal land area constraint coordinates.

Points	x	y
1	20	0
2	90	0
3	70	100
4	20	100
5	0	40
6	20	20
7	20	0

This design process first determines an outer rectangle that encompasses the land area, as shown in Figure 4.27a. The bi-uniform constrained nested domain is constructed. The domain is then processed to remove any boreholes that lie outside of the land area, but keep boreholes that lie along the perimeter. Then, the boreholes that are inside of, or lie on top of the described no-drilling zone are removed, leaving only borefields with boreholes inside of the described drilling area. Due to the clipping procedure, a rare, but possible occurrence is that more boreholes could be located in a field than the one before it (in a domain). Therefore, a final sorting process is done to sort the domains by the number of boreholes to ensure the domain produces a unimodal discontinuous excess temperature function.

The field determined by the routine, shown in Figure 4.27a, contains 88 boreholes with an x-spacing of 10m and a y-spacing of 6.67m. The height of each borehole is 132.21m, resulting in a total drilling depth of 11,634.3 m. This is comparable to the rectangular building results of Figure 4.19, even though the building is more elongated. The increase in drilling depth, when compared to the results of Figure 4.24 can be explained by the cutout procedure, where the boreholes don't fall along the perimeter.

A future improvement of this algorithm could be to place boreholes considering the available land, and constraints. The method could start placing boreholes at the



(a) (b)

Figure 4.27: The initial process of determining the largest rectangle (a) and the results for a bi-uniform rectangular search with no-drilling zones and polygonal land description (b).

perimeter, and if the next borehole in the row is to fall inside of the no-drilling zone, then the borehole spacing could be reduced back to fit if and only if the minimum borehole spacing could be maintained. An additional degree of freedom could be the rotation of the layout of the rows, or to rotate the polygonal descriptions.

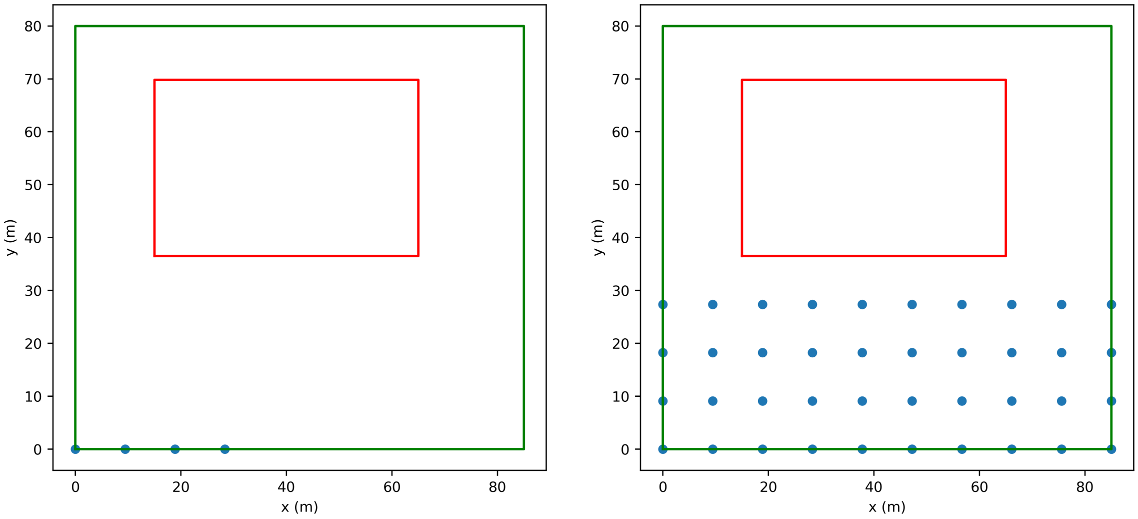
4.4.6 Additional Examples

The additional examples presented in this section show the robustness of the design algorithms. It could be that a designer is unsure of how much land area they may need, so too much land or not enough land could be described. This section presents results of the various sizing algorithms based on either scaled loads, or modified land area. The loads can be scaled at each hour of the year by a constant. If the constant is less than 1, the required borefield will contain less boreholes. If the constant is greater than one, the required borefield will require more boreholes. There are also miniscule and astronomical loads that are created by multiplying the load at each hour by 1/1000 and 1000 respectively. The algorithms can handle any combination of loads and available land area, and if the inputs described don't result in any possible field selections, then the user is informed on how to modify the inputs. The handling of miniscule and astronomical loads is handled by the "Integer Bisection Search". For the case of the loads being miniscule, or the lower and upper end of the domain results in too much excess temperature (both positive), the user is informed that the domain should have less numbers of boreholes. If the lowest part of the domain contains one borehole, then the loads are miniscule and one borehole at minimum depth would solve this designer's problem. For the case of the loads being astronomical, or the higher and lower ends of the domain result in not enough excess temperature (both negative), the user is informed that more land area should be provided, or the minimum spacing should be reduced.

The domains created result in a discontinuous unimodal excess temperature function, and always work themselves down to one borehole. The rectangular based domains start at one borehole, work out to a line of boreholes, and then add lines of boreholes all spaced at the maximum possible spacing. Therefore, if the land area prescribed is excessive, the rectangular based designs will work themselves to one long row of boreholes. The bottom end of the zoned rectangular domain works from one borehole to a line, then to an L, then a U, then closes off the shape with an open rectangle prior to the zoned rectangle domain beginning with one borehole placed inside of an open rectangle.

The results for the equal spacing rectangular search with scaled loads is first looked at. The results for the 1/16 loads in Figure 4.28a is 4 boreholes in a line, each at 121.8 m, while the ½ load profile shown in Figure 4.28b results in 4 rows of 10 boreholes (40 total boreholes). The spacing for each of the cases is uniform and the maximum available given the inputs, which is 9.4 m.

The equal spacing rectangular design routine is now employed for the original Atlanta office buildings, but the available land in the x-direction is moved from 85m



(a) (b)

Figure 4.28: Field selection results for rectangular equal spacing design routine with 1/16 loads (a) and 1/2 loads (b).

to 200m. The result, shown in Figure 4.29, is a borefield containing 84 boreholes all at 10m spacing, and sized for 126.47 m for each borehole. For this design routine, as the land area is stretched in one direction, the selected configuration will tend to a single line of boreholes with maximum spacing.

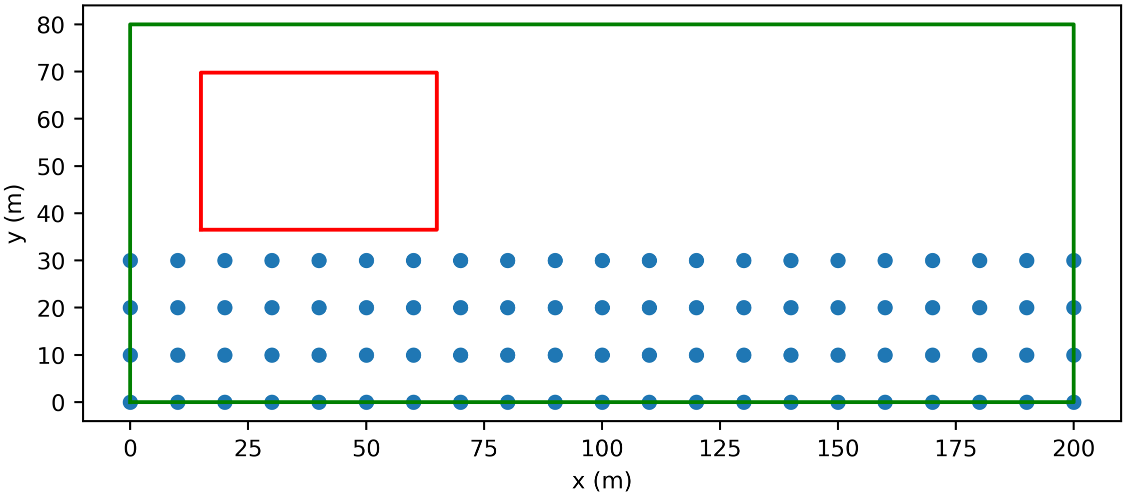
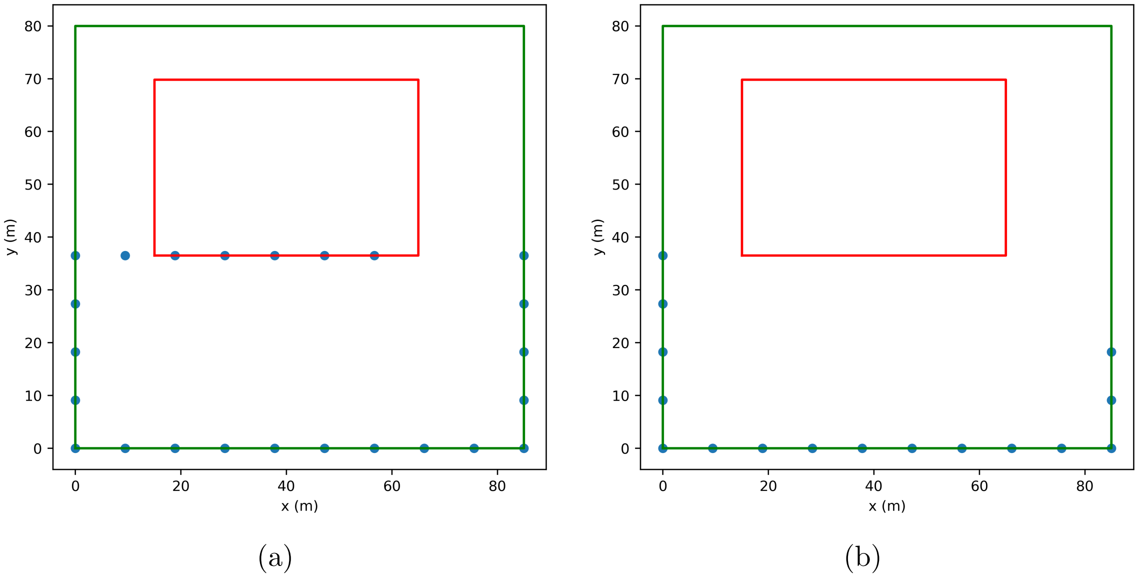


Figure 4.29: Result for uniform constrained rectangular search with original Atlanta loads and a stretched available land area.

The lower end of the domain for the bi-uniform rectangular search is the same as the rectangular search, with the difference being that the B spacing in the y-direction can vary.

The bi-uniform zoned routine is executed with 3/8 scaled loads and 1/4 scaled loads. The result for the 3/8 scaled loads is a "C" shape, which is nearly an open rectangle, and shown in Figure 4.30a. The result of the design routine with loads that are one quarter of the original Atlanta loads is a lopsided U, seen in Figure 4.30b.



(a)

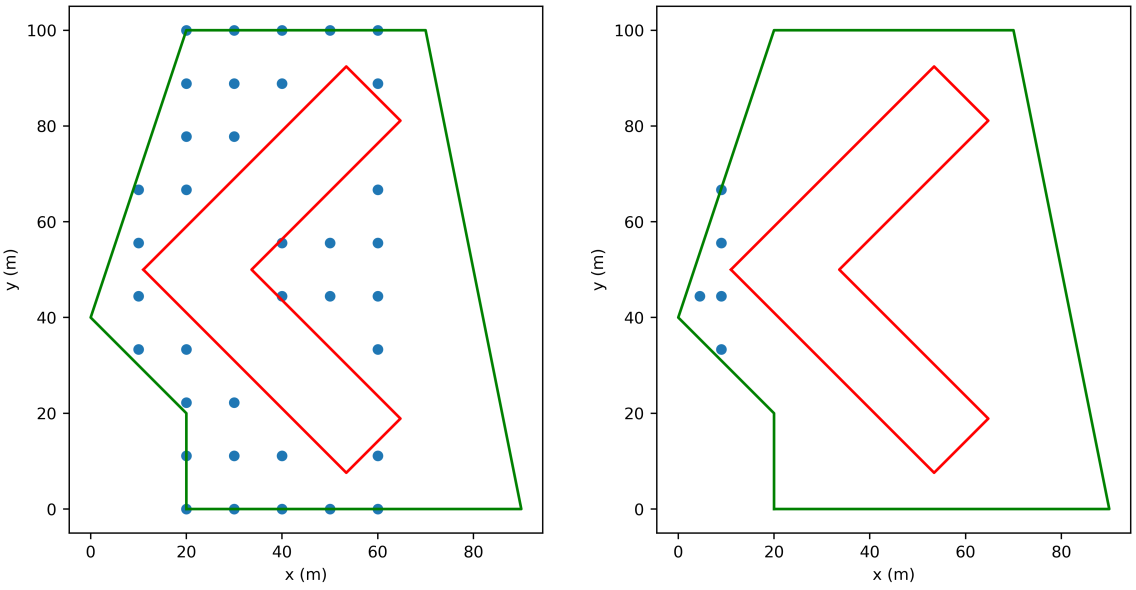
(b)

Figure 4.30: Results of bi-uniform zoned rectangular design procedure with 3/8 scaled loads (a) and 1/4 scaled loads (b).

The bi-uniform rectangular search with a polygonal outer description and no-drilling zones is shown to handle 1/2 scaled loads in Figure 4.31a and 1/16 scaled loads in Figure 4.31b.

4.5 Conclusions

Unequal segments for use in a stacked finite line source solution is shown to both increase accuracy and can decrease speed. The equivalent borehole method of Prieto and Cimmino (2021) is validated for use in ground heat exchanger design. The equivalent borehole method can utilize unequal segments along the borehole for higher accuracy while utilizing less segments. The burial depth is shown to be a sensitive



(a)

(b)

Figure 4.31: Results of bi-uniform zoned rectangular design procedure with 3/8 scaled loads (a) and 1/4 scaled loads (b).

parameter in the design process, and access to a live time g-function calculation provides the ability to control the burial depth used in the g-function calculation. The live time g-function calculation also allows use of the more physically-realistic uniform inlet fluid temperature boundary conditions, specific to the borehole resistance and flow rate for the design in question. For a burial depth of 2m, Hellström's g-functions (calculated with a 4.5m burial depth) for a 12x13 rectangular layout are shown to over approximate the required design depth by more than 6% for a 20-year simulation, while the unequal segment length equivalent borehole method g-functions over approximate by slightly more than 1%, when compared to reference converged uniform inlet fluid temperature g-functions.

Novel design procedures for automated selection of practically placed borefields are presented. An integer-based bisection method is presented for quickly searching and selecting fields in a domain. The domains are constructed so that the resulting discontinuous excess temperature function is unimodal over the entire curve. The more complicated design procedures utilize an outer domain search and then an inter domain search over a nested domain of configurations.

It is shown that for the same land area constraint, bi-uniform rectangular design and bi-uniform zoned rectangular design algorithms can respectively save 9.3% and 14.15% when compared to a human iterated design utilizing Hellström's g-functions. As was shown by Spitler et al. (2020), utilizing a larger section of land area that wraps around a building provides significant savings. The bi-uniform algorithm is utilized to select wrap around configurations and is shown to save 47.6% when compared to the base case. The bi-uniform algorithm is also shown to be applicable to outer land constraints described by any polygonal shape. The design algorithms are shown to be robust, where the routines will select and size a field given any description of inputs or inform the user on how to adjust the inputs so that the algorithm can determine a field.

These novel design procedures provide significant new design capabilities not previously available in any tool. These design procedures rely on user inputs that, in some cases, serves as a surrogate for economic optimization. E.g. maximum spacing is intended to limit the amount of trenching that would occur if the goal to minimize drilling depth were unconstrained. Development of accurate cost models for drilling and trenching could support better economic optimization as part of the design. Future work could refine these design procedures or provide additional design procedures. Development of accurate cost models for drilling and trenching could support better economic optimization as part of the design.

CHAPTER V

Conclusions and Future Work

The work in this thesis has been based on the thermal response functions (g-functions) originally developed by Prof. Claesson of Lund University and his graduate students, using calculation procedures developed by Dr. Cimmino of Polytechnique Montreal. Before getting to the main conclusions, a couple interesting findings on the nature of the calculated g-functions should be noted.

- g-Functions are scaled on 4 dimensionless parameters. According to Eskilson (1987), the ratio of burial depth to borehole length is relatively unimportant for burial depths between 1 and 5m. However, as shown in Figure 4.9b, there is significant sensitivity to burial depth in that range. For the case of 12x13 equally spaced boreholes at 5m and borehole heights of 96m, the value of g at ln(t/t_s)=3.03 is 36.5% higher for the 5m burial depth than the 1m burial depth. This difference makes a significant difference in the resulting design, and burial depth is a parameter worth considering for the calculation of g-functions.

- Cimmino and Bernier (2014) suggested that 12 segments per borehole were an adequate (good enough) approximation of a converged case that used 256 segments. The conclusion appears to have been made based on a 10x10 borehole field at one B/H value of 0.05. However, as shown in Figure 2.1, the required number of segments varies significantly with borehole height. The required number of segments appears to have a correlation with the B/H value. Additionally, the required number of segments appears to be dependent on the number of boreholes and the density of the boreholes. The required number of segments increases as the number of boreholes increase, shown in Figure 2.6. This is exacerbated at deeper borehole heights. The results of the adaptive discretization scheme for the Poisson disc sampled cases in Figure 2.18 were less accurate than the square cases in Figure 2.17. The density, or proximity, of boreholes is different for square and Poisson cases.

The following additional conclusions were reached in this work:

- A g-function calculation tool, written in C++, named *cpgfunction*, has been developed for computation of g-functions on high performance computing clusters. The methodology is similar to *pygfunction*, but with significant changes to reduce memory consumption by packing the segment response matrix and utilizing an adaptive discretization scheme to approximate converged UIFT g-functions utilizing fewer vertical segments in the UBWT calculation. The accuracy of *cpgfunction* was validated against *pygfunction*. The speed of *cpgfunction*, for most cases, was faster. The resulting memory consumption was reduced by approximately a factor of 8. This allows almost all g-function calculations on the OSU HPCC to be done on the most widely available compute nodes with 96 GB of RAM. As a result, substantially more computing hours can be performed in less real time. *Cpgfunction* coupled with an adaptive discretization scheme made it possible to utilize the many OSU HPCC "batch" compute nodes to create a new more expansive library of g-functions.

- A pure C++ fork of *cpgfunction* was developed for EnergyPlus. The project, named *cpgfunctionEP*, provides EnergyPlus with an enhanced third-party g-function calculation tool. A user of EnergyPlus can specify the number of segments per borehole, but the default option is to utilize the adaptive discretization scheme. The new calculation is significantly faster and more accurate than the previously available uniform heat flux boundary condition.

- *Pygfunction* has been contributed to and enhanced, namely with the addition of a common g-function interface, coaxial pipes, unequal segment lengths along the vertical length of the borehole. This was open-source collaboration with Dr. Massimo Cimmino (the owner and creator of *pygfunction*). Computing g-functions with various g-function boundary conditions (possibly for comparison purposes) is greatly simplified by a single common g-function interface API. The use of unequal segment lengths in *pygfunction* has been proven to be more accurate and faster while using fewer segments. The stacked finite line source solution developed by Cimmino and Bernier (2014) had previously only utilized equal segment lengths along the borehole. Unequal segment lengths are a major new development and now default in *pygfunction*.

- The proposed equivalent borehole method of Prieto and Cimmino (2021) coupled with unequal segment lengths has proven to be an accurate method for use in g-function calculation. The coupled method is default for g-function calculation in GHEDT and *pygfunction* (version 2.1).

- A powerful new sizing tool, GHEDT, has been created, and validated against GLHEPRO. The tool can consider single U-tubes, multiple U-tubes and coaxial tubes in the borehole. A single equivalent U-tube methodology is implemented so that double-U and co-axial GHEs sized with GHEDT can be simulated in EnergyPlus using an equivalent single U-tube. GHEDT can perform both hourly and hybrid time step simulations. The hourly simulation can be utilized to verify the accuracy of the hybrid time step procedure. The hybrid time step procedure developed for GHEDT is an improved version of the hybrid time step simulation described by Cullin and Spitler (2011). The simulation has been improved by implementing the recommendations of Cullin and Spitler: using different peak load durations for each month; using 48-hour periods to determine the peak load durations; and placing the peak loads on the correct day each month. In addition, the original Cullin and Spitler implementation only used integer peak load durations and required extensive user interaction. The new approach implemented here uses non-integer durations and automates the entire process converting hourly loads to the hybrid time step representation.

- A novel design methodology for automated selection of borehole fields is developed and presented. The methodology performs optimization based on a target drilling depth and is a capability of GHEDT. An integer bisection search routine is developed to quickly search domains of boreholes for an optimized configuration. GHEDT has the novel ability to automatically position boreholes given a constrained land area and no-drilling zones. This feature can give excellent designs, though there is room for further optimization.

The following items are recommended as future work:

- The performance of *cpgfunctionEP* should be increased by utilizing vectorization for the linear algebra, the finite line source integration approximation of Cimmino (2021) and the identification of similarities. The goal is for the pure C++ version to become as fast as *cpgfunction*, so that the projects can be merged. The project should incorporate the following additional enhancements:
  - Equivalent borehole method of Prieto and Cimmino (2021) for g-function calculation.
  - The ability to compute g-functions with unequal segment lengths.
  - Computation of a g-function with the MIFT boundary condition.

- *Pygfunction* is currently limited to vertical boreholes. Drilling non-vertical boreholes, e.g. to make use of the ground beneath buildings, is common in Scandinavia. Pygfunction should be enhanced to compute g-functions with tilted boreholes. A design algorithm could then be developed for GHEDT that allows the boreholes to be tilted as a part of the optimization process.

- A graphical user interface (GUI) should be developed (potentially using the GUI toolkit; Qt), that wraps GHEDT. At each new release of GHEDT, the GUI should have executables (applications) made for Windows, Mac and Linux operating systems.

- GHEDT was made possible by the previous development and availability of an open-source g-function calculation program. The result of the open-source collaboration could be a great benefit to the area of ground heat exchanger design. The code of GHEDT should be online and available for engineers to analyze and modify. Development of a funding mechanism, perhaps a membership fee, to support further development of GHEDT should be considered.

- GHEDT currently relies on user input of heat rejection/extraction loads. This requires some approximation for the heat pump performance. Development and integration of a model for the heat pump(s) that would allow building heating and cooling loads to be the input is recommended.

- GHEDT currently only optimizes based on height of the boreholes. Input for vertical drilling and horizontal trenching drilling costs could be utilized to develop a optimization procedure based on system installation cost.

- An algorithm should be implemented that can perform automatic circuiting of the piping network to help with cost prediction and real-world installation.

- • The short-time step g-functions generated with GHEDT have minor differences from those generated with GLHEPRO. The difference in resulting sizing could be important (perhaps as high as a few %) for cases where the annual heat rejection and extraction are well-balanced and the flows are very low. This should be further investigated. The development and utilization of both a detailed numerical model and experimental measurements are recommended. A numerical model that can model the internal borehole thermal interaction for all geometries (coaxial, single U-tube, or multiple U-tubes) would allow a systematic investigation to either validate or help with correction of GHEDT and/or GLHEPRO. Experimental testing would be useful to validate the numerical model and determine the accuracy of the convection correlation approximations that are made in the co-axial model.

- GHEDT has powerful search capabilities not available in commercial design tools such as GLHEPRO and EED. These features should be made available in commercial design tools. Additionally, commercial design tools should be updated to consider the burial depth in the g-function.

References

ASHRAE (2011). *The ASHRAE Handbook 2011: HVAC Applications: I-P and SI Editions*, chapter 34. ASHRAE.

Bell, I. H., Wronski, J., Quoilin, S., and Lemort, V. (2014). Pure and pseudo-pure fluid thermophysical property evaluation and the open-source thermophysical property library CoolProp. *Industrial & Engineering Chemistry Research*, 53(6):2498–2508.

BLOCON (2015). *Earth Energy Designer (EED) Version 3.2 Manual.* Retrieved from `https://buildingphysics.com/manuals/EED3.pdf` on November 27, 2021.

BLOCON (2017). *Earth Energy Designer (EED) Version 4 Update Manual.* Retrieved from `https://buildingphysics.com/manuals/EED4.pdf` on November 25, 2021.

Bridson, R. (2007). Fast poisson disk sampling in arbitrary dimensions. *ACM SIGGRAPH.*

Çengel, Y. A. and Ghajar, Afshin, J. (2015). *Heat and Mass Transfer : Fundamentals & Applications.* McGraw-Hill, New York, N.Y, 5th edition.

Chiasson, A. D., Rees, S. J., and Spitler, J. D. (2000). A preliminary assessment of the effects of groundwater flow on closed-loop ground source heat pump systems. Retrieved from `https://www.osti.gov/biblio/20104801` on November 25, 2021.

Cimmino, M. (2015). The effects of borehole thermal resistances and fluid flow rate on the g-functions of geothermal bore fields. *International Journal of Heat and Mass Transfer*, 91:1119–1127.

Cimmino, M. (2016). Fluid and borehole wall temperature profiles in vertical geothermal boreholes with multiple u-tubes. *Renewable Energy*, 96:137–147.

Cimmino, M. (2018a). Fast calculation of the g-functions of geothermal borehole fields using similarities in the evaluation of the finite line source solution. *Journal of Building Performance Simulation*, 11(6):655–668.

Cimmino, M. (2018b). pygfunction: an open-source toolbox for the evaluation of thermal response factors for geothermal borehole fields. In *Proceedings of eSim 2018, the 10th conference of IBPSA-Canada*, pages 492–501. IBPSA Canada.

Cimmino, M. (2019). Semi-analytical method for g-function calculation of bore fields with series- and parallel-connected boreholes. *Science & Technology for the Built Environment*, 25(8):1007–1022.

Cimmino, M. (2021). An approximation of the finite line source solution to model thermal interactions between geothermal boreholes. *International Communications in Heat and Mass Transfer*, 127:105496–.

Cimmino, M. and Bernier, M. (2014). A semi-analytical method to generate g-functions for geothermal bore fields. *International Journal of Heat and Mass Transfer*, 70:641–650.

Claesson, J. and Eskilson, P. (1988). Conductive heat extraction to a deep borehole: Thermal analyses and dimensioning rules. *Energy*, 13(6):509–527.

Claesson, J. and Hellström, G. (2011). Multipole method to calculate borehole thermal resistances in a borehole heat exchanger. *HVAC&R Research*, 17(6):895–911.

Claesson, J. and Javed, S. (2011). An analytical method to calculate borehole fluid temperatures for time-scales from minutes to decades. *American Society of Heating, Refrigerating, and Air-Conditioning Engineers, Inc*, 117(2):279–288. `https://research.chalmers.se/en/publication/525719`.

Collins, R. and McClain, T. (2009). An overview of geothermal grouts. https://www.thedriller.com/articles/87744-an-overview-of-geothermal-grouts.

Cook, J. (2021). *gFunctionDatabase – A g-function database with a database management system for data retrieval, updates and application to be used in geothermal ground heat exchanger design.* https://pypi.org/project/gFunctionDatabase/.

Cook, J. C. and Spitler, J. D. (2021). Faster computation of g-functions used for modeling of ground heat exchangers with reduced memory consumption. In *Proceedings of Building Simulation 2021 Conference*. International Building Performance Simulation Association.

Cullin, J. R. (2008). Improvements in design procedures for ground source and hybrid ground source heat pump systems. Master's thesis, Oklahoma State University, Stillwater, OK.

Cullin, J. R. and Spitler, J. D. (2011). A computationally efficient hybrid time step methodology for simulation of ground heat exchangers. *Geothermics*, 40(2):144–156.

Cullin, J. R., Spitler, J. D., Montagud, C., Ruiz-Calvo, F., Rees, S. J., Naicker, S. S., Konečný, P., and Southard, L. E. (2015). Validation of vertical ground heat exchanger design methodologies. *Science & Technology for the Built Environment*, 21(2):137–149.

Deru, M., Field, K., Studer, D., Benne, K., Griffith, B., Torcellini, P., Liu, B., Halverson, M., Winiarski, D., Rosenberg, M., Yazdanian, M., Huang, J., and Crawley, D. (2011). U.S. Department of Energy Commercial Reference Building Models of the National Building Stock. Technical report, National Renewable Energy Laboratory. https://doi.org/10.2172/1009264.

Dinse, D. R. (1998). Geothermal system for school. *ASHRAE journal*, 40(5):52–55.

DOE (2021). *EnergyPlus*. `https://energyplus.net/`.

Eskilson, P. (1986). Superposition borehole model. *Manual for Computer Code*.

Eskilson, P. (1987). *Thermal analysis of heat extraction boreholes*. PhD thesis, University of Lund, Lund, Sweden.

Eskilson, P. and Claesson, J. (1988). Simulation model for thermally interacting heat extraction boreholes. *Numerical Heat Transfer*, 13:2:149–165.

Florea, L. J., Hart, D., Tinjum, J., and Choi, C. (2017). Potential impacts to groundwater from ground-coupled geothermal heat pumps in district scale. *Groundwater*, 55(1):8–9.

Florides, G. and Kalogirou, S. (2007). Ground heat exchangers—a review of systems, models and applications. *Renewable Energy*, 32(15):2461–2478.

Fyffe, J. R., Clayton, M. E., Grosvenor, C. E., and Webber, M. E. (2011). Analysis of Large-Scale Ground Source Heat Pump Systems for Residential Heating and Cooling in Austin, TX. volume Volume 4: Energy Systems Analysis, Thermodynamics and Sustainability; Combustion Science and Engineering; Nanoengineering for Energy, Parts A and B of *ASME International Mechanical Engineering Congress and Exposition*, pages 717–728.

GeoPro, I. (2021). PowerTEC. `https://geoproinc.com/products/powerTEC.html`.

Gnielinski, V. (1976). New equations for heat and mass transfer in turbulent pipe and channel flow. *Int. Chem. Eng.*, 16(2):359–368.

Gnielinski, V. (2013). On heat transfer in tubes. *International Journal of Heat and Mass Transfer*, 63:134–140.

Grundmann, R. M. (2016). Improved design methods for ground heat exchangers. Master's thesis, Oklahoma State University, Stillwater, OK.

Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., van Kerkwijk, M. H., Brett, M., Haldane, A., Del Río, J. F., Wiebe, M., Peterson, P., Gérard-Marchant, P., Sheppard, K., Reddy, T., Weckesser, W., Abbasi, H., Gohlke, C., and Oliphant, T. E. (2020). Array programming with numpy. *Nature (London)*, 585(7825):357–362.

Hellström, G. (1991). *Ground heat storage : thermal analyses of duct storage systems.* PhD thesis, University of Lund, Lund, Sweden.

Hellström, G. and Sanner, B. (1994). Software for dimensioning of deep boreholes for heat extraction. In *Proceedings of Calorstock*, pages 195–202.

Hill, C. (2017). *Poisson disk sampling in Python.* Retrieved 29 January, 2021 `https://scipython.com/blog/poisson-disc-sampling-in-python/`.

Hughes, P. (2008). Geothermal(ground-source)heat pumps: Market status, barriers to adoption, and actions to overcome barriers. Technical report. https://doi.org/10.2172/948543.

Ingersoll, L. R., Zobel, O. J., and Ingersoll, A. C. (1948). *Heat Conduction, with Engineering and Geological Applications.* International series in pure and applied physics. McGraw-Hill Book Co., New York, 1st edition.

Javed, S. and Spitler, J. (2016). Calculation of borehole thermal resistance. In Rees, S. J., editor, *Advances in Ground-Source Heat Pump Systems*, chapter 3, pages 63–95. Woodhead Publishing.

Javed, S. and Spitler, J. (2017). Accuracy of borehole thermal resistance calculation methods for grouted single u-tube ground heat exchangers. *Applied Energy*, 187:790–806.

Kavanaugh, S. (1992). Simulation of ground-coupled heat pumps with an analytical solution. In *Proceedings of the ASME International Solar Energy Conference*, volume 1, pages 395–400.

Kavanaugh, S. P. and Rafferty, K. D. (1997). *Ground-source heat pumps : design of geothermal systems for commercial and institutional buildings.* American Society of Heating, Refrigerating and Air-Conditioning Engineers, Atlanta.

Liu, X. (2010). Assessment of national benefits from retrofitting existing single-family homes with ground source heat pump systems. Final report, Energy and Transportation Science Division, Oak Ridge National Laboratory. Report number 1023843.

Liu, X., Spitler, J. D., DeGraw, J. W., Cook, J. C., New, J. R., Adams, M. B., and Holladay, S. (2020). FY20 Third Milestone Report for Advanced Techno-Economic Modeling for Geothermal Heat Pump Applications in Residential, Commercial, and Industry Buildings.

Liu, X. A. and Hellström, G. B. (2006). Enhancements of an integrated simulation tool for ground-source heat pump system design and energy analysis. `https://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.509.9391`.

Malayappan, V. and Spitler, J. (2013). Limitations of using uniform heat flux assumptions in sizing vertical borehole heat exchanger fields. In *Proceedings of Clima 2013*. IBPSA Canada. `https://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.714.7613`.

Manickam, A., Dharapuram, M., Delahoussaye, R., and Spitler, J. (1996). *GLHEPRO for Windows. The Professional Ground Loop Heat Exchanger Design Software. Version 2.06 Users Guide., Oklahoma State University.*

Marshall, C. and Spitler, J. (1994). *GLHEPRO: The Professional Ground Loop Heat Exchanger Design Software. Users Guide., Oklahoma State University.*

Mitchell, M. S. (2019). *Enhanced Vertical Ground Heat Exchanger Model for Whole Building Energy Simulation.* PhD thesis, Oklahoma State University, Stillwater, OK.

Munoz, A. (2014). Higher order ray marching. *Computer graphics forum*, 33(8):167–176. `http://giga.cps.unizar.es/~amunoz/projects/CGF2014_higherorder/`.

OSUHPCC (2021). *OSU's newest supercomputer "Pete" is available for all OSU researchers.* https://hpcc.okstate.edu/pete-supercomputer.html.

Paul, N. D. (1996). *The effect of grout thermal conductivity on vertical geothermal heat exchanger design and performance.* PhD thesis, Mechanical Engineering Department, South Dakota State University.

Press, W. H., Teukoksky, S. A., Vetterling, W. T., and Flannery, B. P. (2002). *Numerical Recipes in C++ : The Art of Scientific Computing.* Cambridge University Press, Cambridge, UK, 2nd edition.

Prieto, C. and Cimmino, M. (2021). Thermal interactions in large irregular fields of geothermal boreholes: the method of equivalent boreholes. *Journal of Building Performance Simulation*, 14(4):446–460.

Spitler, J. and Bernier, M. (2016). Vertical borehole ground heat exchanger design methods. In Rees, S. J., editor, *Advances in Ground-Source Heat Pump Systems*, chapter 3, pages 63–95. Woodhead Publishing.

Spitler, J., Cook, J., West, T., and Liu, X. (2021). G-function library for modeling vertical bore ground heat exchanger [data set]. Technical report, Oak Ridge National Laboratory. `https://dx.doi.org/10.15121/1811518`.

Spitler, J. D. (2000). Glhepro : a design tool for commercial building ground loop heat exchangers. https://www.osti.gov/etdeweb/biblio/20144646.

Spitler, J. D. and Cook, J. C. (2021). Sizing ground heat exchangers with rectangular constraints - zoned rectangular configurations. Technical report, Submitted to Oak Ridge National Laboratory, February 26, 2021.

Spitler, J. D., Cook, J. C., and Liu, X. (2020). A Preliminary Investigation on the cost Reduction Potential of Optimizing Bore Fields for Commercial Ground Source Heat Pump Systems. Stanford University, Stanford, California. Proceedings of the 45th Workshop on Geothermal Reservoir Engineering. February 10-12.

Spitler, J. D. and Gehlin, S. E. (2015). Thermal response testing for ground source heat pump systems—an historical review. *Renewable & Sustainable Energy Reviews*, 50:1125–1137.

van de Geijn, R. and Goto, K. (2011). *BLAS (Basic Linear Algebra Subprograms)*, pages 157–164. Springer US, Boston, MA.

Virtanen, P., Gommers, R., Oliphant, T., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., Walt, S., Brett, M., Wilson, J., Millman, K., Mayorov, N., Nelson, A., Jones, E., Kern, R., Larson, E., and Pingel, T. (2020). Scipy 1.0: fundamental algorithms for scientific computing in python. *Nature Methods*, 17(3):261–272.

Xu, X. and Spitler, J. (2006). Modeling of vertical ground loop heat exchangers with variable convective resistance and thermal mass of the fluid. Pomona, NJ. Proceedings of the 10th International Conference on Thermal Energy Storage-Ecostock.

Yavuzturk, C. and Spitler, J. D. (1999). Short time step response factor model for vertical ground loop heat exchangers. *ASHRAE Transactions*, 105.

Yoo, A. B., Jette, M. A., and Grondona, M. (2003). SLURM: Simple Linux Utility for Resource Management. In Feitelson, D., Rudolph, L., and Schwiegelshohn, U., editors, *Job Scheduling Strategies for Parallel Processing*, pages 44–60, Berlin, Heidelberg. Springer Berlin Heidelberg.

Young, T. R. (2004). Development, verification, and design analysis of the borehole fluid thermal mass model for approximating short term borehole thermal response. Master's thesis, Oklahoma State University, Stillwater, OK.

APPENDIX A

High Density Polyethylene Pipe Dimensions

SDR 11

Nominal Size	Outer Diameter			Inside Diameter			Area	
in.	in.	ft	mm	in.	ft	mm	ft^2	m^2
3/4	1.050	0.0875	26.67	0.848	0.0707	21.54	0.0039	0.00036
1	1.315	0.1096	33.40	1.062	0.0885	26.96	0.0061	0.00057
1 1/4	1.660	0.1383	42.16	1.340	0.1117	34.03	0.0098	0.00091
1 1/2	1.900	0.1583	48.26	1.534	0.1278	38.96	0.0128	0.00119
2	2.375	0.1979	60.33	1.917	0.1598	48.70	0.0200	0.00186
2 1/2	2.875	0.2396	73.03	2.321	0.1934	58.96	0.0294	0.00273
3	3.500	0.2917	88.90	2.825	0.2355	71.77	0.0435	0.00405
3 1/2	4.000	0.3333	101.60	3.229	0.2691	82.02	0.0568	0.00528
4	4.500	0.3750	114.30	3.633	0.3027	92.27	0.0719	0.00669
5	5.563	0.4636	141.30	4.491	0.3742	114.07	0.1099	0.01022
6	6.625	0.5521	168.28	5.348	0.4457	135.85	0.1559	0.01449
8	8.625	0.7188	219.08	6.963	0.5802	176.86	0.2643	0.02457
10	10.750	0.8958	273.05	8.678	0.7232	220.43	0.4105	0.03816
12	12.750	1.0625	323.85	10.293	0.8577	261.44	0.5775	0.05368

Schedule 40

Nominal Size	Outer Diameter			Inside Diameter			Area	
in.	in.	ft	mm	in.	ft	mm	ft^2	m^2
3/4	1.050	0.0875	26.67	0.810	0.0675	20.59	0.0036	0.00033
1	1.315	0.1096	33.40	1.033	0.0861	26.23	0.0058	0.00054
1 1/4	1.660	0.1383	42.16	1.363	0.1136	34.61	0.0101	0.00094
1 1/2	1.900	0.1583	48.26	1.593	0.1327	40.46	0.0138	0.00128
2	2.375	0.1979	60.33	2.049	0.1707	52.04	0.0229	0.00213
2 1/2	2.875	0.2396	73.03	2.445	0.2037	62.09	0.0326	0.00303
3	3.500	0.2917	88.90	3.042	0.2535	77.26	0.0504	0.00469
3 1/2	4.000	0.3333	101.60	3.524	0.2937	89.52	0.0677	0.00629
4	4.500	0.3750	114.30	3.998	0.3331	101.54	0.0871	0.00809
5	5.563	0.4636	141.30	5.016	0.4180	127.41	0.1372	0.01274
6	6.625	0.5521	168.28	6.032	0.5026	153.21	0.1983	0.01843
8	8.625	0.7188	219.08	7.942	0.6619	201.74	0.3439	0.03195
10	10.750	0.8958	273.05	9.976	0.8314	253.40	0.5426	0.05041
12	12.750	1.0625	323.85	11.889	0.9908	301.99	0.7706	0.07159

APPENDIX B

Eskilson Logarithmic Points

Table B.1: Eskilson's 27 logarithmic points used for the long time-step g-function calculation.

Index	ln(t/ts)
1	-8.5
2	-7.8
3	-7.2
4	-6.5
5	-5.9
6	-5.2
7	-4.5
8	-3.963
9	-3.27
10	-2.864
11	-2.577
12	-2.171
13	-1.884
14	-1.191
15	-0.497
16	-0.274
17	-0.051
18	0.196
19	0.419
20	0.642
21	0.873
22	1.112
23	1.335
24	1.679
25	2.028
26	2.275
27	3.003

VITA

JONATHAN CHRISTOPHER COOK

Candidate for the Degree of

Master of Science

Thesis: DEVELOPMENT OF COMPUTER PROGRAMS FOR FAST COMPUTATION OF G-FUNCTIONS AND AUTOMATED GROUND HEAT EXCHANGER DESIGN

Major Field: Mechanical Engineering

Biographical:

Education:

Completed the requirements for the degree of Master of Science with a major in Mechanical Engineering at Oklahoma State University in December 2021.

Received a Bachelors of Science in Mechanical Engineering at Oklahoma State University in May 2019.

Experience:

Employed as a graduate research assistant (GRA) for the Oklahoma State University Building and Environmental Thermal Systems Research Group (BETSRG) from August 2019 to present.

Open-source contributions:

A contributor to Massimo Cimmino's *pygfunction* and the Department of Energy's (currently managed by the National Renewable Energy Laboratory) *EnergyPlus*.