Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Engineering
Department of Civil and Environmental Engineering

Magnus Nordin

Utilizing Preprogrammed GIS-Functions through User Friendly Python Scripted Map Solutions

Master's thesis in Engineering and ICT
Supervisor: Terje Midtbø, IBM
Co-supervisor: Hege Le Brun, Ambita AS
April 2022

NTNU
Norwegian University of Science and Technology

Magnus Nordin

Utilizing Preprogrammed GIS-Functions through User Friendly Python Scripted Map Solutions

Master's thesis in Engineering and ICT
Supervisor: Terje Midtbø, IBM
Co-supervisor: Hege Le Brun, Ambita AS
April 2022

Norwegian University of Science and Technology
Faculty of Engineering
Department of Civil and Environmental Engineering

Utilizing Preprogrammed GIS-Functions through User Friendly Python Scripted Map Solutions

Magnus Nordin

TBA4925 - Geomatics

2022

Master's Thesis

Abstract

Based on the vast amount of different geographical analyses, this thesis explores if there is a market for preprogrammed analysis with the use of GIS-functions. By interviewing active customers of Ambita's map solution products, we can gain insight about the interest for new and more advanced products. This thesis presents a general background of GIS, an introduction in the use of scripts in geographical analysis with examples of studies using scripted workflows within GIS. The interview phase includes a survey with carefully chosen questions in order to answer important research questions in the thesis. After analyzing the results of the interviews, multiple programs were developed as scripts designed with Python. These scripts were used in timed analyses to compare how much time that is saved by interacting with my scripts compared to using a regular GIS-program. The thesis concludes that scripted analyses can be valuable for the future of GIS, as long as the raw data is of good quality and the analyses are well arranged.

Sammendrag

Basert på den enorme mengden av forskjellige geografiske analyser undersøker denne rapporten om det er et marked for forhåndsprogrammerte analyser med bruk av GIS-funksjoner. Ved å intervjue aktive kunder som benytter Ambitas kartløsningsprodukter, kan vi få innsikt i kundenes interesse for nye og mer avanserte produkter. Denne masteroppgaven presenterer en generell bakgrunn om GIS og en introduksjon til bruk av skripting i geografiske analyser med eksempler på studier hvor arbeidsflyten er styrt av forhåndsdefinerte kommandoer (skript). Intervjufasen inneholder en spørreundersøkelse med nøye utvalgte spørsmål for å finne svar på problemstillinger i oppgaven. Basert på resultatene fra intervjufasen ble det utviklet flere programmer i form av Python-skript. Disse skriptene ble brukt i analyser hvor det ble målt tidsforbruk for å sammenligne hvor mye tid som ble spart av å bruke mine script i motsetning til et vanlig GIS-program. Oppgaven konkluderer med at forhåndsprogrammerte analyser kan være verdifulle for framtiden til GIS, så lenge rådataene er av god kvalitet og at analysene er godt tilrettelagt.

Preface

This paper is the product of my master's thesis at NTNU, Department of Civil and Environmental Engineering.

I would like to thank my supervisor Terje Midtbøe for the support and guidance through my studies.

I would also like to thank Hege Le Brun, my supervisor at Ambita, who wanted to engage in the thesis.

Most of all, I want to thank my wife Marte. Without her guidance and motivation, I wouldn't have made it this far. She truly is a wonder!

Eidsvoll, April, 2022

Magnus Nordin

Contents

Introduction

In the early years of the twentieth century, cartography was a time consuming profession, where the creation of a map was not something that was done "in a day". To generate a proper world map, many different expeditions across the world were launched to carefully measure and calculate distances and heights (Werenskiold et al., 1935). When the first large scale maps were created, there were a lot of inaccuracies, and many areas were poorly mapped or not at all. Rainforests, desserts and other woodlands were often left out due to the level of difficulty in exploring and measuring these areas. Also, there was a lesser need to accurately map these far off places.

Fast forward to the invention of electronic equipment. Eventually there were machines that could automatically measure distance and angle from one point to another and, with the inclusion of the GPS satellite system which became fully operational in 1993 (Mai, 2012), could measure the exact position and height for all locations on the planet. These advancements made the making of maps much easier and a lot more detailed and accurate.

Today there are even more options available. With the development of programmable drones with cameras, together with aerial photogrammetry by the use of airplanes, it is possible to map almost any surface on the planet in a much shorter time than before.

In the past, maps were hand drawn and later printed on paper and collected into atlases (Briney, 2021). Updating the maps with new data would mean manually entering the new info, or reprinting an entire atlas in a new version. At the time, this time consuming process might not have been a real concern. There was no alternative. Now the maps and data are available online. If something is found to be wrong, it is possibly fixed with the click of a button. If something is updated, for instance an orthophoto, an areal photograph of the ground which is compensated for distortion, it simply replaces the old one in an instant. Thus the users get the updated version immediately.

There is a large array of map solutions available to the public today, based on many different forms of data collection and processing of geospatial data (Lwin et al., 2012). Organizing all the existing data, and simplifying the analysis of map data is a task that is growing in importance, as the shear amount of data is growing as never before. Map data can be useful in many different circumstances. Terrain models, land mass and nesting places for endangered species are some examples of useful information. With that much data available, there has to be a way to filter out and extract the information that is needed for different tasks. Programs for doing such calculations already exist, but they are heavy, complicated and some times unstable to use (Sipe and Dale, 2003). One of the goals is to figure out how the process can be simplified, or even automated, for the average person. By programming scripts to execute parts of the analysis, the users can avoid having to perform repeatable tasks multiple times. The scripts can range from executing a single geoprocessing tool, to performing a complete analysis consisting of several processes executed in sequence. However, it must be considered whether or not the solution is something that the average person would want to utilize. The best option might still be to let GIS-experts do the analysis, considering there are a lot of ways to calculate and analyze geographical data.

In this thesis, the importance of geographical analysis is explained and further discussed how scripting, in the form of preprogramming GIS-functions with these analyses, is the next big step in making map data available for the average map user. During this process, there has been an interview phase with active customers of Ambita AS, gathering information on the current needs for scripted geographical analyses, and to see how content GIS-users are with the current solutions. After the interview phase, a user interactive test was created based on the answers received in the interviews. The test was created with Python designed scripts, developed in this thesis, and conducted on a small group of people given the time constraint of the thesis and the willingness of the invited users. During the test, the test users had to perform a specific analysis in the GIS-program QGIS, then use the scripts with preprogrammed workflows to create the same result. The test users were timed based on how long they interacted with QGIS, and compared with how long they interacted with the interfaces of the scripts.

The thesis aim to answer the following research questions:

1. How far has GIS become automated with the help of scripted workflows?
2. What kind of GIS-analyses do users want or need scripted?
3. How much time can a GIS-user save from using scripted workflows?

The rest of the thesis is structured as follows: The main history of cartography and GIS, as well as the road to digitization of maps and the current use of scripted workflows are described in Chapter 1. In Chapter 2 preprogramming and possible automation by using scripts is explained and how Python can play a key role in achieving this. Chapter 3 presents the setup of the interview phase, and based on these results, the geographical analyses that were chosen for the timed user test. The results from the interviews, along with the results from the user interaction with the GIS-program and Python scripts, are located in Chapter 4. Discussion about the results and the process of the thesis is placed in Chapter 5, with a conclusion ending in Chapter 6. Appendix A contains the survey used in the thesis. Appendix B, C and D contain the code for the different preprogrammed scripts designed with Python, while Appendix E display the raw time data from the user interaction tests.

Chapter 1

Background and History

In this chapter, the history of cartography and the present maps and map making processes are explained. It includes examples from a range of studies that utilize preprogramming in graphical analyses.

1.1 Cartography

Cartography is the study and practice of making maps. Map making is thought to have started over 16 500 years ago (Whitehouse, 2000), where the stars were depicted rather than land, but those maps have little in common with modern maps. There are a lot of records of cartography being used to measure borders throughout history. In early Egyptian culture, triangles were a central element of measuring borders of land areas (Blackburn and Holister, 1988), as shown in Figure 1.1. The borders of different farm land had to be redrawn after every cycle of the Nile flooding the farms. The location was ideal because of the nutritional deposition from the flood, but the overflowing water also erased the borders, which meant they had to be redrawn.

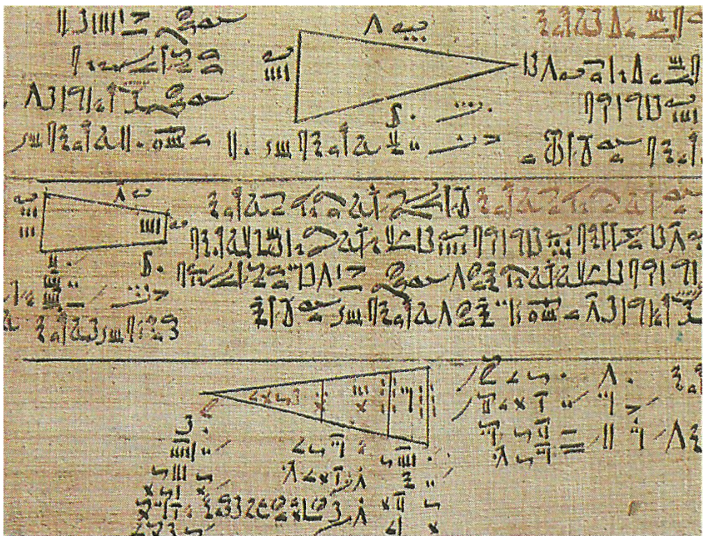


Figure 1.1: Land surveying document from ancient Egypt, estimated age 1 600 BCE.

Credit: Encyclopedia of Modern Technology, Volume 9, page 33.

Already in the beginning of the 20th century, every bit of a land was discovered and mapped by manual labor. This includes countless of hours spent walking and carefully mapping the surface of the earth, using the principles of triangulation. From a world atlas published by Cappelen (Werenskiold et al., 1935) there is a quote:

> It sure would be comfortable for them (cartographers)
> if they could get a secure viewpoint high above the surface,
> so high that the land could be observed, strongly downsized, as on a map.
> Up there they could, in peace and quiet, depict the green meadows,
> the dark forests, the rivers that sparkles and the smoke filled cities. (p.1)

This dream became a reality with the commercialization of air planes. Photogrammetry, the science of measuring objects pictured in photographs, can be utilized to map 3D images of buildings (Nilsson and Grundberg, 2009) by using overlapping pairs of images. It is also used on a large scale with air planes to map large areas in a very short amount of time. These aerial photographs are captured from a direct top-down view. They overlap in such a way that every point on the ground is represented in at least two different images. This way, two overlapping images can be used to create an illusion of depth in a way that makes the image appear three dimensional. With these images, and the use of known coordinates and heights for specific features in the images, it is possible to calculate the position and height for every point of interest in the images.

1.2 Digital Maps

Before the twentieth century, it was a rare sight to see commoners in possession of maps, especially land maps. In the beginning, these were used for military purposes only, and were handled as classified documents. Only the officers got the privilege to study the projection of earth from above. Take for instance the Norwegian public enterprise "Statens Kartverk", founded as "Norges Geografiske Oppmåling" in 1773. This enterprise had a strictly military purpose, which was to measure the area along the Norwegian-Swedish border from Enningdalen in the south, to Stene Skanse in Verdalen in the north. Another quote from Cappelen's World atlas reads:

> Now (1935) we have, as known, a different view on this case, and maps
> have a big and steadily increasing utilization among the modern man.
> There is a need for more and more specialized maps, like railway,
> sea, weather, city and historical maps, and usually there is no problem
> gaining access to what we need.
> One can safely assert that today's humans have become quite spoiled,
> in regards to maps. (p.1)

Cartographers in 1935 felt that humans were spoiled in regards to maps, yet today over half of all smartphone users reportedly used Google Maps, a multifunctional digital map, already in 2013 (Smith, 2013). This application can tell you the shortest route to your destination, dependent on your choice of travel, whether it be by driving a car, riding a bike, walking or using public transportation. It can also recommend alternate routes if there is traffic or if the shortest path is otherwise obstructed. Consumers, both back in the 1930s and now, expect precision in maps. They expect that the map shows the tiniest detail about how the rivers flow and that cities has the same location in regards to the surroundings as in reality. As mentioned in Section 1.1, triangulation has always been a solid basis of calculation, and the World atlas from Cappelen also mentions this:

> What is it now that gives our time's maps the security that characterizes them? In reality there are multiple cartographic methods, because map representation is a process with many parts. What is the basis for everything is the triangulation or the triangle-measuring, that rests on the simple relationship of only needing to know the two angles and the length of a single side in order to calculate the other two sides. (p.2)

Today we have a combination of several different methods of measurement and calculation. Manual labor is still a part of the grand picture, but now with the help of GPS and aerial photogrammetry, the measurements and calculations are a lot more precise. This is due to the introduction of extremely precise benchmarks, both in terms of position and height. However, most of the land masses are in motion, although very slowly, meaning that the benchmarks rarely will have to be corrected.

Many people fail to realize how different parts of cartography are linked together. Google has created a functionality called Google Street View, where users can view the map from street level in a 360 degree view. Creating these views is mostly done by driving a car with an advanced camera setting on the roof, but there has been several other camera mounted vehicles to reach more remote places (Abhijit Ogale et al., 2010), and even a backpack for covering very remote places like mountain trails (Crump, 2015).

The coverage of Google Street View varies a lot. In many remote cities, only the main road is mapped. In the larger cities many side streets and trails are mapped. When people are used to everything being readily available, it's easy to forget how much work that has been done to pave the way in the first place, and they can end up being disappointed whenever they cannot view the street they are looking for.

1.2.1 Projections

The world is an uneven ellipsoidal sphere, but as for visualization purposes, it is spherical. This makes a globe the best way to represent it properly. This visualization is not practical for measurements and calculations, and does not adapt well onto a flat surface or digital screen unless the area of interest is very limited. On the other hand, when it is important to include an entire country, or even the whole world at once, there are several problems when trying to replicate the spherical shape. It is impossible to replicate the shapes from the sphere identically onto a 2D-plane (Werenskiold et al., 1935). What actually is projected onto the map, is a grid of circles that encapsulates the globe. These circles, and the crossings between them, are then used as reference points to draw an estimation of the globe onto a flat surface. If there was any possibility to reproduce the grid without any form of error from the globe, the projection would be length-, area-, and angle-indifferent.

This would mean that any distance on the globe to be exactly the same distance in the projection, while every area and angle would have the correct size. However, this ideal-projection doesn't exist and it never will (Deetz and Adams, 1921; Pearson, 1990), just as it is impossible to perfectly unfold a cut up rubber ball onto the ground.

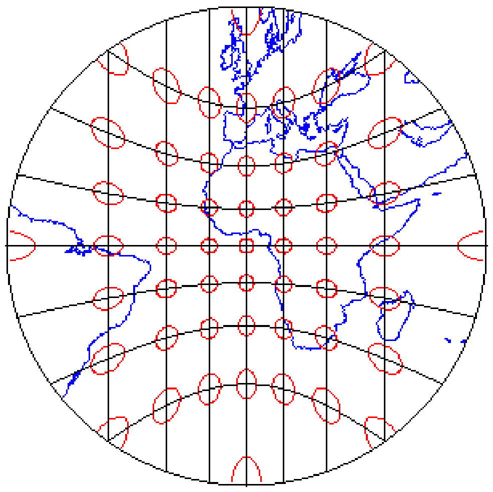


Figure 1.2: Gnomonic projection.

Credit: Department of Geography, The Pennsylvania State University.

This means that there has to be made a choice, whether the most important thing is correct distance, angle, or area. Some map projections have a very distorted view of the world, while having the benefit of every straight line drawn from a point A to a point B representing the shortest possible travel route across the Earth's surface. An example can be seen in Figure 1.2 where every line drawn would replicate a straight line drawn on a globe of the Earth. This makes for a great way to determine shortest travel path for air and boat travel (Albrecht, 2004). Some nations even have their own projections, especially made to create precise measurements within the border, both in regard to absolute position, or just a separate height datum (Solheim, 2000). Constructing a bridge while using inaccurate data or wrong reference points can become a costly project if the end result is a misalignment at the meeting point (Leigh, 2021). When performing GIS-analysis it is important that the relevant data is using a projection that is centered on the area of interest, to avoid distortions from having different reference points (Albrecht, 2004).

1.2.2 Data Collection

In order to produce digital maps, there is a need for geospatial data. This data can be generated through field surveying, remote sensing, or by digitizing existing analog maps and map data (Lwin et al., 2012). Many of the data sources today is based on remote sensing in the form of aerial photography and high-resolution satellite imagery, and all of it is stored digitally, ready to be displayed or further processed.

1.3 Geographic Information Systems

Geographic Information Systems (GIS) are widely used in many work environments today. A GIS is useful for displaying one or more types of data together on a base map. In order to utilize GIS software, it is necessary to have access to geographically referenced information. Many types of data can be geographically referenced, whether they are built up of points, lines or polygons. The main difference between an analog cartographic document and a GIS map, is that it is possible to add and remove as many layers of information as needed with a GIS and perform analysis on the data, creating new or additional data in a way ordinary maps can not. Even though it is possible to apply GIS principles to analog maps, it could not have gained as much growth and importance as it has today without the digitization of maps, as mentioned in Section 1.2.

1.3.1 History of GIS

The development of GIS have been through four distinct phases (Dempsey, 2012a). The first phase ranges from the early 1960s to mid 1970s, where a few key individuals were shaping this new discipline into a new way of improving research and development. From the mid 1970s to early 1980s, national agencies started to adopt the technology, which led to the focus on best practice development. In the rest of the 1980s, GIS gained momentum in the commercial market, and in the last phase, from the end of 1980s and forward, the focus has been on making the GIS technology more user friendly and widely available. However, information about the development of GIS has been withheld from many national departments. Some government functions have actively refused to give information surrounding the adaptation and deployment of their GIS. Others have removed spacial data from the general public in order to ensure public safety (Salkin, 2005).

The early stages of GIS development are also reflecting different instances with disparate goals. At first the major instances were the Canada Geographic Information System, the Harvard Laboratory for Computer Graphics, the Environmental Systems Research Institute (ESRI) and the Experimental Cartography Unit. The first time that GIS gained a common direction of development, was after satellite imaging technology increased the commercial activity around GIS. At that time, ESRI became the major actor in the field, creating a basis for mass applications for both business and private use.

Although GIS today is thought to be digital applications and programs, the first usage of spatial analysis from the principles of GIS can be dated all the way back to 1832, where a French geographer, Charles Picquet, created a representation of cholera epidemiology in Paris, by using a color graded map of the 48 districts of Paris (Dempsey, 2012a). The same disease was responsible for yet another appliance of spatial analysis, shown in Figure 1.3, by an Englishman named John Snow, who made an even better representation of the outbreak, marking every instance of cholera deaths with a distinct position on a map of London in 1854 (Aguirre, 2014). This work resulted in displaying a high number of deaths surrounding a public water pump on Broad Street. Before this work, the common perception was that cholera spread by "bad air", but now it was discovered that the real cause was contaminated water.

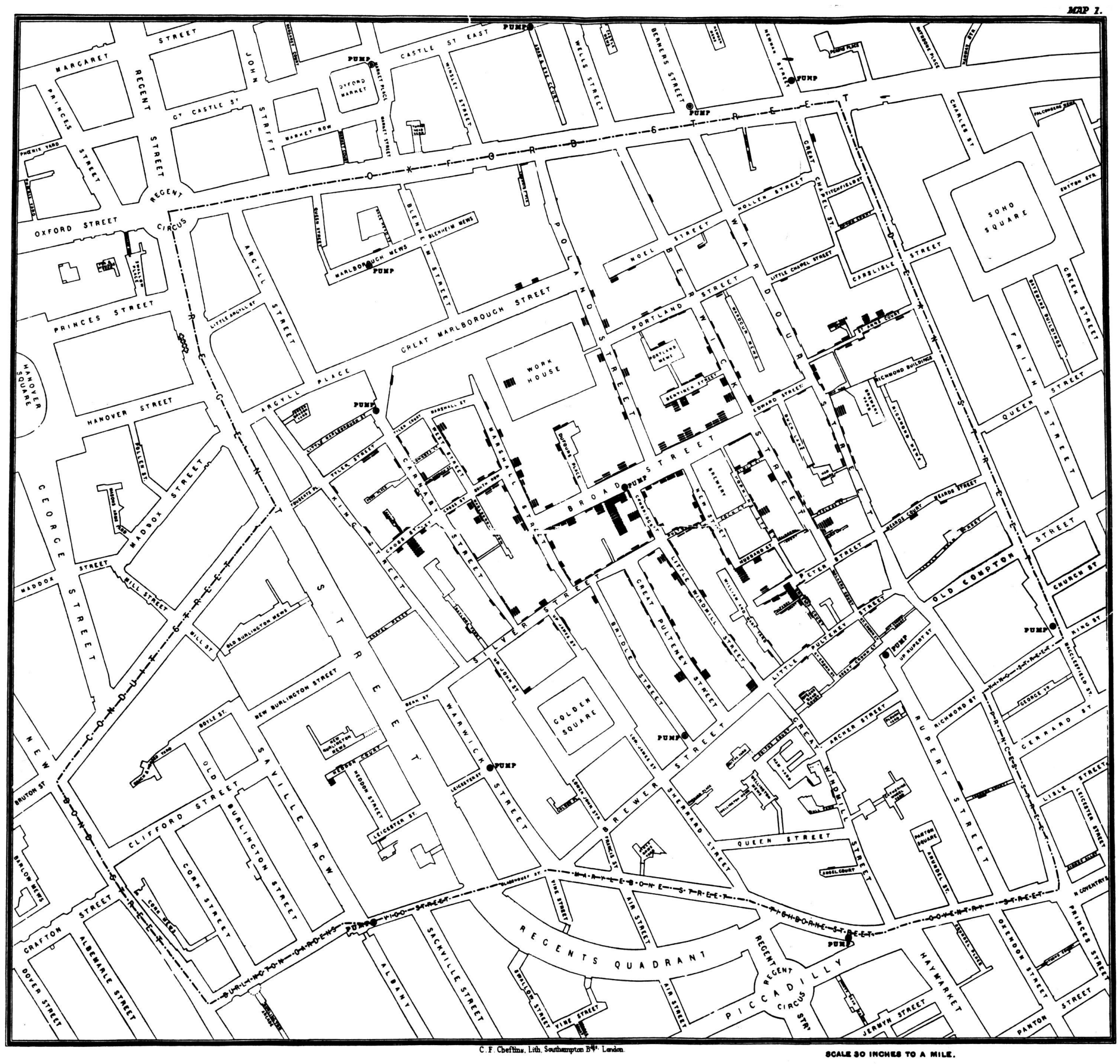


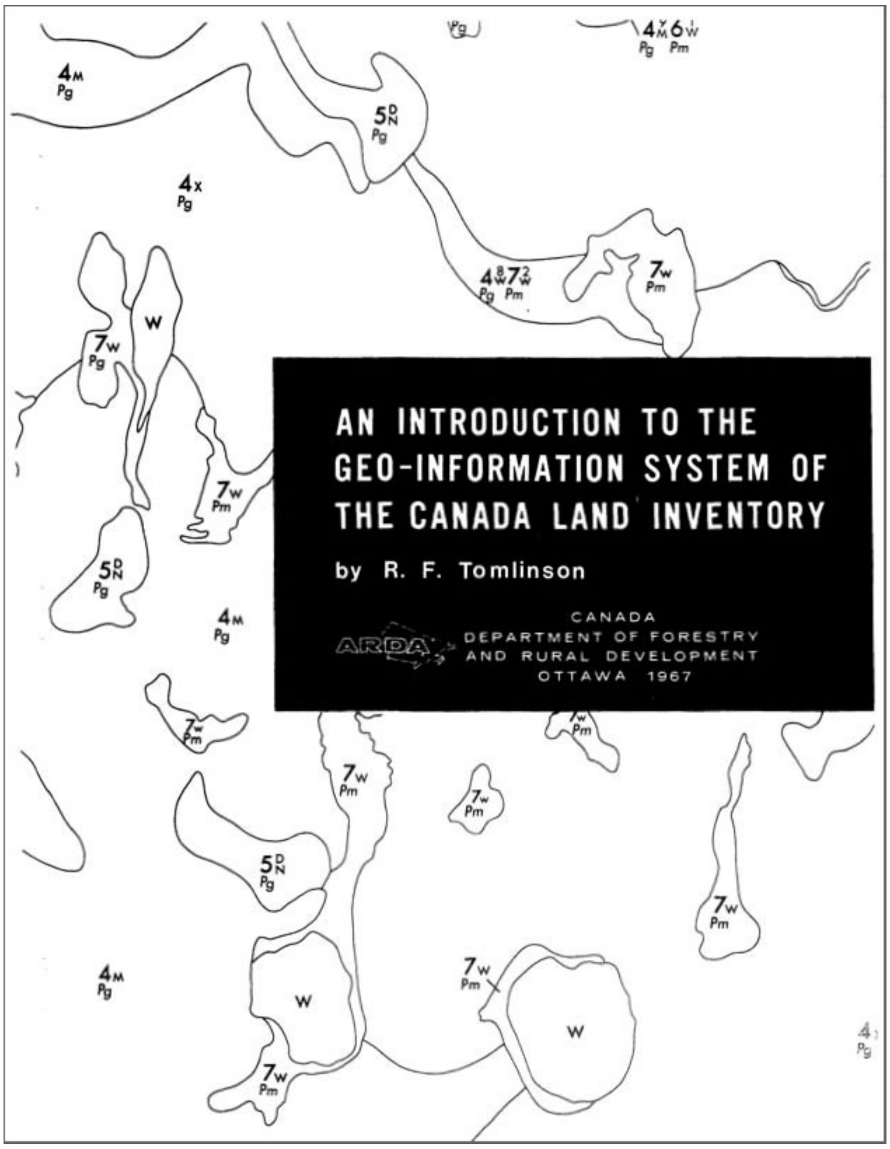
Figure 1.3: John Snow's cholera outbreak map, made with GIS principles.

Credit: Public domain.

The first operational GIS was made in 1962 and contained information about land usage in Canada (Tomlinson, 1999). It was called Canada Land Inventory (CLI) and was the base for the Canada Geographic Information System (CGIS), released in 1967. The name "Geographic Information System" was made a standard by the CGIS. There were a lot of different proposals for naming convention, but a 1967 paper entitled "An Introduction to the Geo-Information System of the Canada Land Inventory", shown in Figure 1.4, gave the idea for the now well known and widely accepted phrase "Geographical Information System".

1.3.2 GIS-programs

Today there are many different programs that revolve around geospatial data handling. Two of the major platforms are ArcGIS and QGIS (Dempsey, 2012b). These programs allow users to manipulate geospatial data and perform advanced analyses,



AN INTRODUCTION TO THE GEO-INFORMATION SYSTEM OF THE CANADA LAND INVENTORY

by R. F. Tomlinson

CANADA
DEPARTMENT OF FORESTRY
AND RURAL DEVELOPMENT
OTTAWA 1967

Figure 1.4: The first appearance of the term geo-information system or Geo-IS.

Credit: ESRI's ArcNews (Tomlinson, 1999).

by using a large array of built-in GIS functions. These platforms have both adopted Python as their main language for scripting (Altaweel, 2017). The use of Python was further increased with the inclusion of PostGIS (Brennan, 2020). With the use of PostGIS, a spatial database extender to PostgreSQL, it is possible to store geospatial objects and execute complex workflows that can be stored and rerun at a later time. PostGIS is included in both ArcGIS and QGIS, as well as other popular platforms as OpenStreetMap and GRASS.

1.4 Studies of Preprogrammed GIS Workflows

The growth of geospatial analysis has increased a lot in recent years, especially with the utilization of Python (Altaweel, 2020). There are a lot of different ways to make geospatial analysis processes execute automatically, creating a workflow of one or more processes to run in quick succession. This makes a potential user able to do certain types of geospatial analysis without the need to know explicitly how every part of the GIS-programs work. This section will focus on two main sources of making scripted analysis with geospatial workflows.

1.4.1 ArcGIS ModelBuilder

ArcGIS has a built in programming tool called ModelBuilder (ArcGIS, 2004). This is a visual programming language used to create geoprocessing workflows. A more detailed explanation of the ModelBuilder follows in Section 2.1.

The ModelBuilder has been used in several different areas of study. Fernandez and Moya-Delgado (2017) mentions examples from landslide-susceptibility research (Jiménez-Perálvarez et al., 2009), soil erosion analysis (Csáfordi et al., 2012), eco-environmental sensitivity assessments along railways (Dong et al., 2011), land-use and land-cover change projects (Lee et al., 2008; Borna et al., 2011) and transport route planning (Gohari et al., 2012; Mohtashami et al., 2012).

While the ModelBuilder provides some degree of automation on its own, there are some studies that try to implement it in a way that require less interaction with the program. A landslide-susceptibility study from the University of Granada, Spain (Fernandez and Moya-Delgado, 2017) tries to implement an easy-to-use ModelBuilder workflow of the preprocessing part of an advanced landslide analysis to be shared and reused. In order to check that the scripted preprocessing works as intended, it is run on a set of data that previously has been manually altered to suit the following analysis, thereby checking that the output is identical. The conclusion of the aforementioned study mentions some advantages to the scripting of the preprocessing. Given several intermediate steps, a single human error can cascade down upon the final result. Programming these steps to make sure they are executed the same way every time eliminates the possibility of human error. The time spent performing the preprocessing can vary a lot based on the amount of knowledge and training the user has, as well as the different steps taking variable amounts of time to execute.

Time usage will be a central part of a user test later in this thesis, which will aim to answer the third research question mentioned in the introduction.

1.4.2 Scripted GIS Workflows with Python

Similar to how the ModelBuilder is utilized, there is a widespread use of scripting tools to preprogram workflow processes. A lot of these scripts are based on the programming language Python. Some of the major platforms have a built-in Python extension, which will be further detailed in Section 2.2.1.

A study on weather effects on building thermal performance utilized a workflow using packages from Python and R (Skeie and Gustavsen, 2021). The workflow was used to obtain and handle research data. The process preparing the data for further analysis was automated, requiring only longitude and latitude to acquire and process input data. Later in the study, Python scripts were also used in different types of geographical calculations, some using the gdal Python library. Among other findings, the study concludes that using scripting tools to automate geoprocessing tasks effectively helped in utilizing open geospatial data from web map services.

Some of the advantages of using Python for geospatial analysis includes all the tools and libraries that exist in the programming language (Toms et al., 2018). Anaconda is an open source platform for using Python and R. It is popular because it includes many of the tools used in data science and machine learning from a single install (G, 2017). GeoPandas is another open source project that uses ArcGIS API to process geospatial data for use in the cloud (Jordahl et al., 2021). It extends the data types used by Pandas, a data analysis tool built on Python, to conduct spatial operations on geometric object types. Using GeoPandas eliminates the need to have a spatial database like PostGIS.

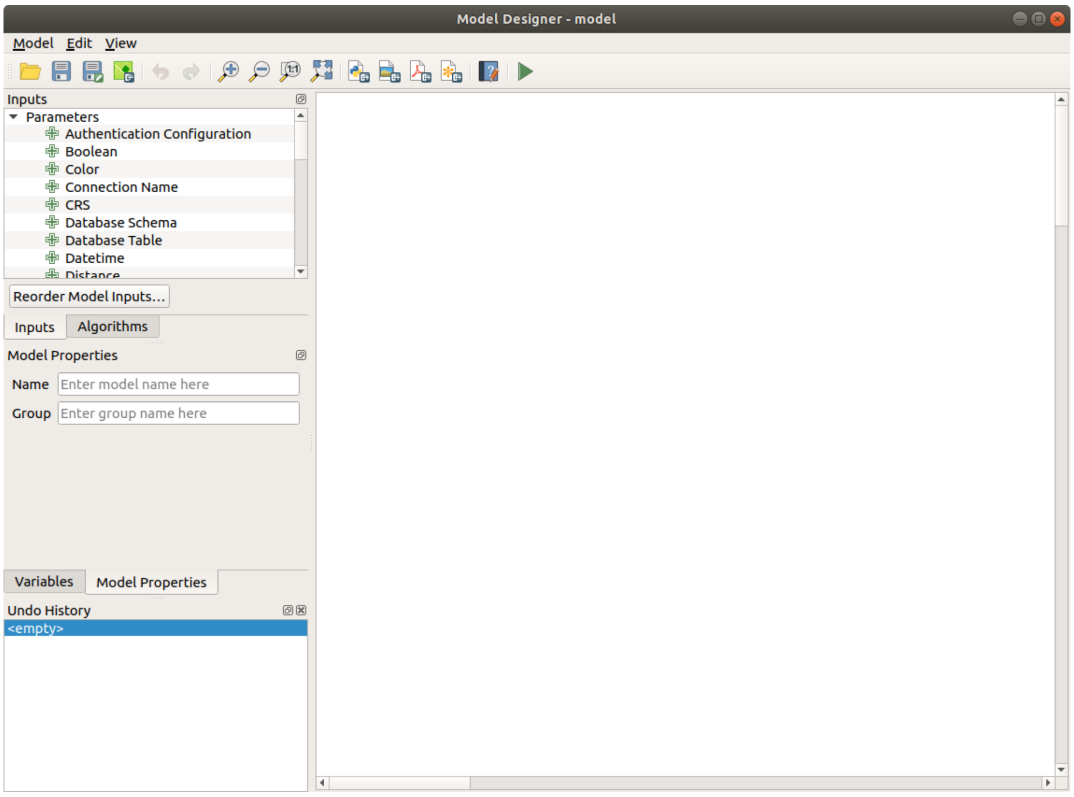


Figure 1.5: QGIS' Graphical Modeler

Credit: QGIS Development Team (2021).

QGIS has a Python based framework called "Processing", which aim to integrate geoprocessing tools from a wide range of software libraries (Graser and Olaya, 2015). Within the framework exist a graphical tool named the "Graphical Modeler", shown in Figure 1.5. It can be used to create geoprocessing workflows and export them as Python scripts, making all the different steps and processes packaged into a single algorithm on a Python script file. The modeler has been used in a case study

(Cosentino and Pennica, 2015) to automate a commonly used analysis in relation to identifying unstable zones in seismic microzonation maps. The model takes a range of inputs, performs several operations and ends with extracting areas prone to instability from thematic maps. The case study concludes that the tools found in QGIS and the graphical modeler helps simplify and speed up the processes surrounding spatial analysis in regards to their studies. This modeler works similarly to ArcGIS's ModelBuilder, but does not feature as frequently in academic works as of today.

Chapter 2 further details different ways to script GIS processes and in what way Python can be utilized for this purpose, as well as how Python will be used later in this thesis.

1.4.3 Graphical User Interface

Computer interaction is usually handled by the means of a Graphical User Interface (GUI), from the basis operating system by using a mouse cursor (Levy, 2018), to more specialized GIS-oriented software (Olivera et al., 2006) that can be designed with Python (Baykal and Colak, 2022).

Different types of GUIs are created for different purposes, but the main reason for creating them is easier interaction with a computer program and handling of data (Berry et al., 2014).

Both of the aforementioned modelers from ArcGIS and QGIS are GUIs. They have the ability to perform a large range of operations, and have a generalized purpose that includes a lot of the many GIS-operations that exists. While these GUIs can be utilized for many purposes, it is more focused on having a lot of options. When trying to preprogram and speed up interaction, which is what this thesis will examine, a more narrow and customized GUI might be a better choice.

In order to benefit from scripted workflows in the GIS environment, the users must be able to interact with and utilize the program efficiently. One possible way of achieving this, is by creating a user-friendly and intuitive GUI that only handles the requirements for the scripted workflow, which will be tested and utilized in Section 3.2.2.

1.5 Ambita

This thesis will be conducting a survey, mentioned in Section 3.1, in cooperation with Ambita, which is a Norwegian tech company specializing in national property data. Ambita was founded in 1987 under the name "Norwegian property information", with the purpose of digitizing the national land register.

Their main focus today is to help customers digitize and automate processes. They offer specific packages for real estate agents that covers everything needed to sell a property, electronic registration of property, insight solutions to help analyzing customer data and map solutions and analysis. Ambita is the largest supplier of Municipal map data in Norway.

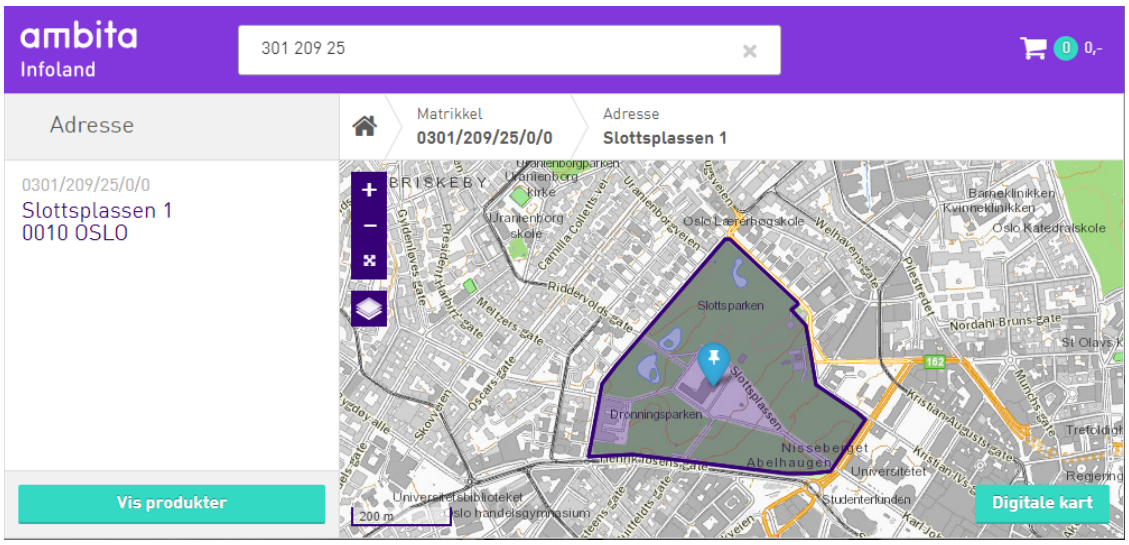


Figure 1.6: Ambita infoland store.

Credit: infoland.ambita.com - Ambita AS.

Figure 1.6 shows their online product distribution page, which is called "Ambita Infoland". Products that can be found here includes, but is not limited to, the "real estate package", information about property, map data, roads, regulation plans and zoning, cadastral data and orthophotos.

Chapter 2

Preprogrammed Geoprocessing with Python Scripts

This chapter will explain the meaning of preprogrammed geoprocessing in geographical analysis with the use of scripting, and how Python plays a big part in the process.

2.1 Scripting

GIS-programs usually contain a big variety of processing tools. This is to cover every possible need for the advanced user, but for most projects and tasks, only a few select tools are needed to get the desired output. Scripting every single process as they are needed is a difficult and often unnecessary task. Choosing to script tasks that are certain to be used again, or that are supposed to be used on several different data, is the most cost effective way to go. It is especially effective to script processes that are to be repeated in a specific time interval. If there is a dataset that is certain to be updated on a monthly basis, it would be redundant to manually perform the necessary analysis in a GIS, if the process can be stored and restored for the next iteration. With input and output management on the aforementioned task, the process could be completely automated without need for human interaction.

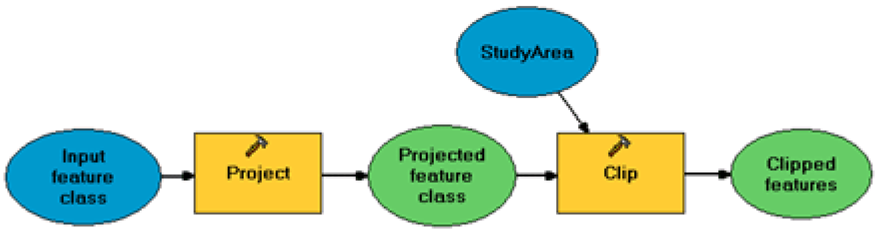


Figure 2.1: Example of a ModelBuilder workflow in ArcGIS.

Credit: ESRI's ArcGIS Resources.

Take a simple example: A geographical analysis is to be done containing four different sets of data. Figure 2.1 shows the ArcGIS ModelBuilder workflow for the example. In order to perform geographical analysis, the data need to be defined in the same projection and covering the same area of interest. The first thing to do in this case, is to convert every dataset to one common projection. If there exist data without a connection to any projection, it is not a valid dataset. When all the valid data are defined in the same projection, they can be cut to exclude data outside the scope of the geographical boundary. This action is done with a tool that is often called a "Clip tool". All datasets are then "clipped" in accord to a static polygon, called StudyArea in Figure 2.1, which return only the data covering the area of interest.

If this example was completed manually, it would take four identical iterations to reach the goal. Given that the four iterations are identical, it is possible to make a scripted workflow that would do the iteration on every given input. The script is created in advance, and is fed the four different datasets and the boundary polygon. The script could then do the whole task in the same time as manually doing a single iteration.

In an example like this, the time spent interacting with the script is $\frac{1}{x}$ the time it would take to interact with the processes separately, where x is the amount of datasets needed to process. The more datasets to process, the more time saved with scripting.

A variety of candidate cartographic products for automated map production workflows is summarized by Buckley and Watkins (2009) as:

- *many maps with the same theme but different extents,*
- *many maps with the same extent but different themes, and*
- *many maps with the same extent and the same theme, but the data for the theme changes over time.*

The easiest geoprocessing workflow to automate, with the use of scripting, is when only the input data changes over time, as this would simply require to rerun the original script. This process can be set to run at specific intervals, or whenever the new data is updated.

In some cases, there might be a need to change the requirements as a project advance. If a specific task is to find the optimized spot for a fire station, there might be a requirement to have it localized near the citizens to minimize travel time, but not too close to an existing fire station. It should also be located near a main road, and not in some tight neighborhood. In reality it would be impossible to meet all these requirements perfectly, thus it is necessary to give each of them a weight. It serves as a priority list, without having to totally exclude anything. For projects with one specific case, the weight list would only be generated and used once, but if several fire stations were to be built, this analysis could be used in many different places. Each and every location might have different priorities of the requirements listed, and therefore would be processed with different weights. With scripted workflows it is not necessary to go through all the steps over and over, but have an editable variable for the weights that is changed every time the scripted processes execute. Figure 2.2 shows an example of how the output from a weighted geographical analysis with two requirements can be presented.

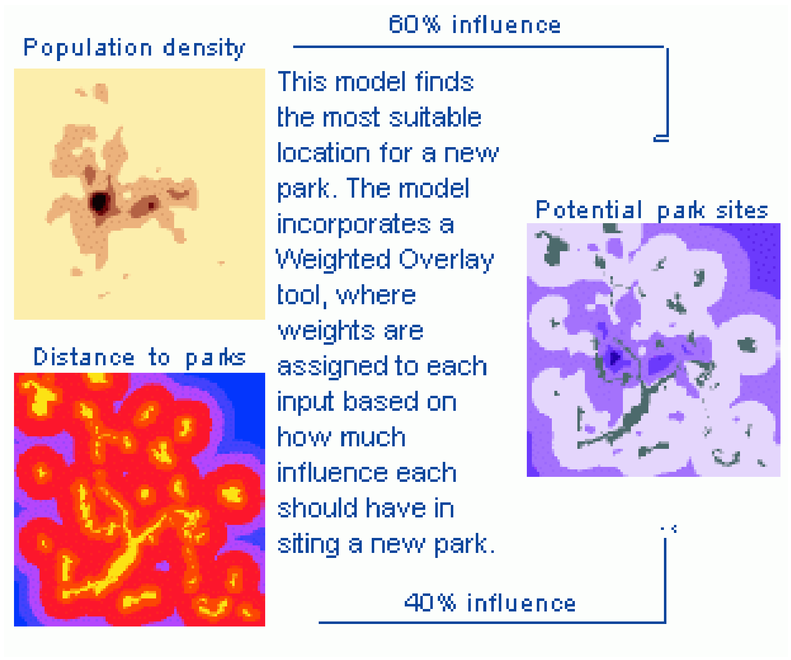

60% influence
Population density
This model finds the most suitable location for a new park. The model incorporates a Weighted Overlay tool, where weights are assigned to each input based on how much influence each should have in siting a new park.
Potential park sites
Distance to parks
40% influence

Figure 2.2: Example of a weighted overlay in ArcGIS.

Credit: ESRI's ArcGIS Resources.

2.2 Python in GIS-Programs

Python is a lightweight open source programming language that is fairly easy to learn and use compared to the more high-level programming languages such as C, C++ and Java. Despite being a lightweight, there are a lot of community generated libraries and packages that give it a very comprehensive application area.

Given Pythons more intuitive syntax, it is easier for a larger community of users to fully embrace it. With the large number of libraries, Python is still able to be used to create a large variety of new applications, such as integration of mapping features with web programs, GIS functionality for mobile and new tools that require server and cloud based systems (Bahgat, 2015).

ArcGIS and QGIS are two of the major platforms in modern GIS, as mentioned in Section 1.3.2. ArcGIS is ESRI's commercial GIS platform, used by many big corporations, while private users often choose the open source QGIS, given that it does not require a license.

Both ArcGIS and QGIS have embraced Python, and have chosen to build it into their products. They have their own Python extensions called ArcPy and PyQGIS respectively.

2.2.1 ArcPy and PyQGIS

ArcPy is a module that let Python code access the ArcToolbox in ArcGIS (Toms, 2015), where all the major geospatial analysis tools are located. The tools are not written in Python, but the ArcPy module works around that obstacle. In this way, it is possible to write geospatial analyses in Python code while utilizing the tools needed from ArcGIS. The module also contains some tools that are not in the ArcToolbox, which makes it an even more powerful tool than using generic ArcGIS.

Using ArcPy it is possible to perform spatial analyses in a manner more similar to regular coding.

ArcGIS already have the ModelBuilder, explained in Section 2.1, and while it is possible to discuss which one is better, ModelBuilder or ArcPy, it is important to include that they can be used together. The ModelBuilder can utilize Python scripting, as well as ArcPy can incorporate models from the ModelBuilder.

Python in QGIS works a little different than in ArcGIS. To utilize Python in QGIS, simply open the built-in Python console window and start writing code. Although this is the easiest option to utilize Python in QGIS, it still requires users to interact with QGIS itself. In this thesis, PyQGIS is used externally as a Standalone Application. This means that it can be run as a separate program, without the need to manually interact with QGIS, as long as it is installed on the computer. A full documentation of PyQGIS is written by Dobias (2010), and a more detailed guide to using Python in QGIS is found in the PyQGIS Developer Cookbook (Qgis, 2013).

PyQGIS is more easily available compared to ArcPy. There is not a clear answer to which is the easiest or best to use, but there are more examples and varied documentation about ArcPy. This results in the fact that it is easier for new programmers to find information for ArcPy, rather than PyQGIS.

Chapter 3

Methods

In order to determine the importance of scripted geographical analysis, interviews including a survey with potential map application users were conducted before determining how to set up a scripted user test. These users were selected from a list of active customers of Ambita's map solutions. Initially 52 users were invited to participate, where 24 responded, and 13 showed interest in participating. 10 interviews were conducted, another 32 potential users were contacted, but none resulted in interviews. Time limits prevented further potential users to be invited to participate. For the interviews, the age of the participants ranges from age 32 to 55, with the average being 42.

Based on the results of the interviews, a range of suitable geographical analyses will be listed for use in a user test of scripted GIS workflows. In the user test these analyses will be conducted using a regular GIS-program QGIS, and by using scripted workflows that I have designed with Python specifically for this thesis, as detailed in Section 3.2.

3.1 Survey

A residential survey conducted in Shanghai, China, became the baseline for creating a suitable questionnaire for this survey phase (Che et al., 2013). The survey in this thesis was adapted to better suit the research goals of this thesis. Much like the original survey, the first part consisted of gathering personal information to see if it was possible to draw any conclusions based on age, gender etc. The second part was adapted with the cooperation and knowledge of Ambita, creating pairs of questions with hypotheses on a correlation between the answers given in each respective pair. The basis for the hypotheses came from a subjective point of view. As the paired questions have similar meanings, and therefore estimated to give a correlated answer, it would be conflicting to get an opposite response than expected. Given the principles of probabilistic reasoning under uncertainty, and that Bayesian reasoning doesn't handle conflicting results well, the hypotheses were created from subjective logic (Jøsang, 2016). The third part of the process contained the answers of the participants, while the final step was checking if the proposed hypotheses matched with the participants attitudes towards the GIS-solutions in question. In order to understand deviations from the expected correlations, a few detailed and open-ended questions were asked in the end of the survey, where the participants could give a deeper explanation of their needs. The survey contained pairs of yes or no questions. As previously mentioned, these had predetermined hypotheses bound to the possible outcome of answers. The hypotheses are displayed in Table 3.1, Table 3.2 and Table 3.3.

Question 1 and 2 are assumed to be inversely correlated, meaning that if a participant answers the first question "yes", it is assumed that the answer to the second question is "no", and vice versa. Hypotheses listed in Figure 3.1

Question 1:	Are you satisfied with the currently existing map solutions?
Question 2:	Do you miss anything in the list of current map solutions?

Table 3.1: Hypotheses for the possible outcome of Question 1 and 2.

	Q1: Yes	**Q1**: No
Q2: Yes	Indifference to efficiency, covers the necessary.	Awareness of problems, obvious flaws in map solution.
Q2: No	Absolute satisfaction.	Dissatisfied with map solutions in general. Need for simpler solutions without more options.

Question 3 and 4 are assumed to be slightly directly correlated, meaning that if a participant answers the first question "yes", it is assumed that the answer to the second question also is "yes", and vice versa. Hypotheses listed in Figure 3.2

Question 3:	Would it be beneficial to have other types of map data than you can get today?
Question 4:	Do you want to be able to choose your own analysis of the data presented to you?

Table 3.2: Hypotheses for the possible outcome of Question 3 and 4.

	Q3: Yes	**Q3**: No
Q4: Yes	Absolute need for extended map service.	Need for specific analysis, competent users who wants to be in control
Q4: No	Need for better service, want more in the easiest way possible.	Absolute satisfaction.

Question 5 and 6 are assumed to be directly correlated, meaning that if a participant answers the first question "yes", it is assumed that the answer to the second question also is "yes", and vice versa. Hypotheses listed in Figure 3.3

Question 5:	Does it take too long to gain the information you need from maps in general?
Question 6:	Would you want to gain information from maps faster than before?

Table 3.3: Hypotheses for the possible outcome of Question 5 and 6.

	Q5: Yes	**Q5**: No
Q6: Yes	Want better efficiency, willingness to try something new to get it.	Always interested in enhanced efficiency despite being satisfied with current time spent.
Q6: No	Maps in general feel complicated, not worth it to learn something new.	Absolute satisfaction.

Interviews

The interviews were conducted through physical meetings, over the internet by means of Skype and through regular phone calls. As a part of the survey, this gathered information about age and profession, in order to create a basic demographic for the answers, as well as information about how the subjects utilize map solutions from Ambita and how they handle the map data.

Building on the second goal from the introduction of this thesis, one of the questions asked to the users was "Would you like to have access to scripted workflows for complete geographical analyses?". This is a form of preprogramming on a level that requires a predetermined knowledge of the users needs, before the scripted workflows are implemented. This is not an option for customers who are using ad-hoc analysis, as the users should simply be able to chose a type of analysis on a dataset, and get it without any other specifications. From this, it can be classified as a "one-click-analysis". The full list of questions used in the interview and survey is included in Appendix A.

3.2 Standalone Application for Scripted Analysis

With the usage of Python, as mentioned in Section 2, we will look at some examples of analyses that can benefit from using the QGIS extension PyQGIS in standalone applications as scripts. These examples will show the results of using multiple GIS-functions in succession. Section 3.2.1 will show the results using the GIS-program QGIS, while Section 3.2.2 show the results from the scripts that I have developed in order to help the users perform the analyses faster. After developing these scripts based on the interview phase, they will be used in a timed user test with a smaller subset of participants. Section 3.2.3 explain the final part of the user test where the participants will be performing single geographical analyses, timing each one, to compare time spent on basic operations. All of these scripts are designed with Python, including intuitive graphical user interfaces(GUI), explained in Section 1.4.3, for easier use of the application.

The manual analyses will require the users to control that all the input data has the same projection as one another, while the script includes a function called "reprojectlayer", which ensures that all the input data is setup in the same projection to avoid projection errors in the form of distortions as mentioned in Section 1.2.1.

3.2.1 Manual Analyses in QGIS

Performing geographical analysis require geospatial data to be processed. The data used in this example are publicly available municipal data from Norway which has been produced by means mentioned in Section 1.2.2.

Suitable Building Grounds

Given the detailed feedback from the survey, it would seem one of the most wanted analysis is to locate convenient building grounds. Based on multiple factors that are directly excluding the possibility to build on a given location, it is possible to end up with an overlay that shows the valid building locations.

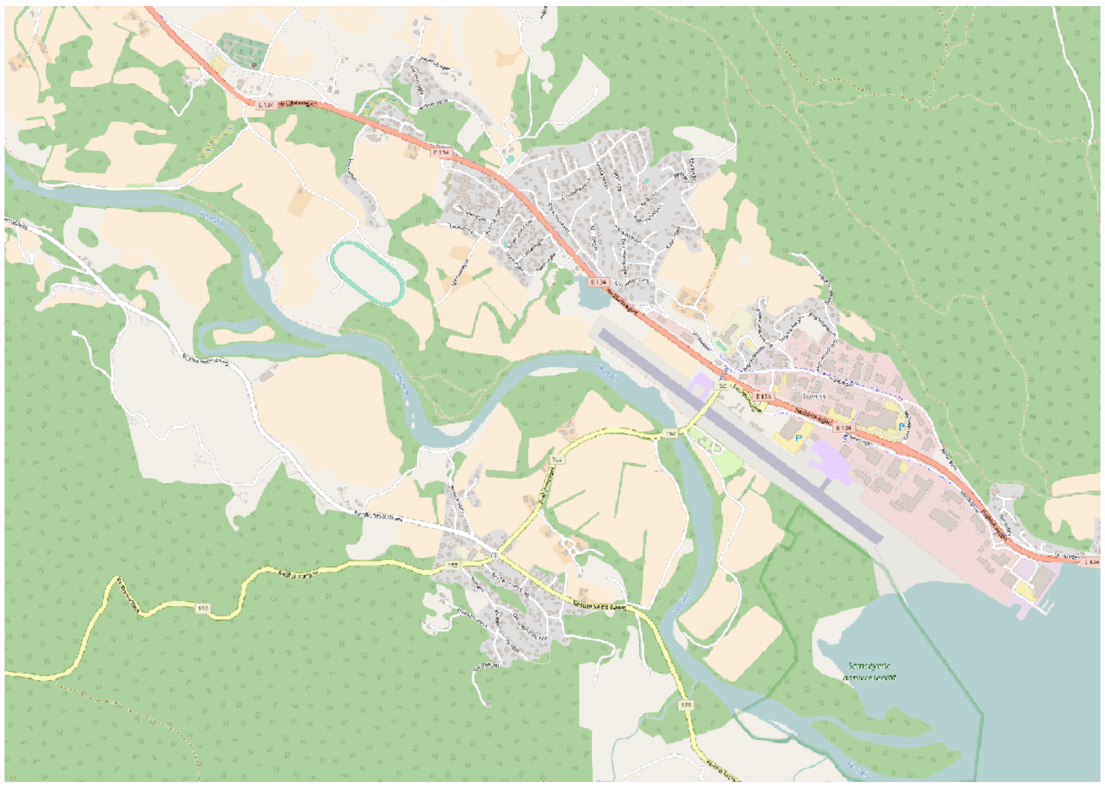


Figure 3.1: Background map of Notodden, Telemark, Norway.

Figure 3.1 shows a basic OpenStreetMap (OpenStreetMap-contributors, 2021) background map of the city of Notodden in Telemark, Norway. In an example of a scripted analysis, there would have to be a clear and precise "recipe". If this was supposed to be an analysis to determine valid building grounds for a new large mall, the recipe could be the following:

- The building can not be closer than 5 meter from a currently established road.
- The building should be closer than 250 meter from a currently established road.
- The building has to be at least 10 meter from the water.
- The building can not be built on an area of Natural Reserve.
- The building can not be build on any Conservation area or Protected buildings.
- The building 'can' replace existing housing.

The listed numbers and requirements are not accurate representation for building a large mall, they are meant for demonstration only.

The geographical analysis known as "buffer" creates an encapsulation of a point, line segment or polygon in a way that makes a new border that has a specified width around the point, line or area. Figure 3.2 shows the result of a three step process, where the road network has been used in two different buffer analyses. The first and second step, is a buffer of the size 250 meter, and a buffer of the size 5 meter. The third step is creating the geographical difference between the 250 meter buffer and the 5 meter buffer. The results of these three steps are the gray fields in the figure.

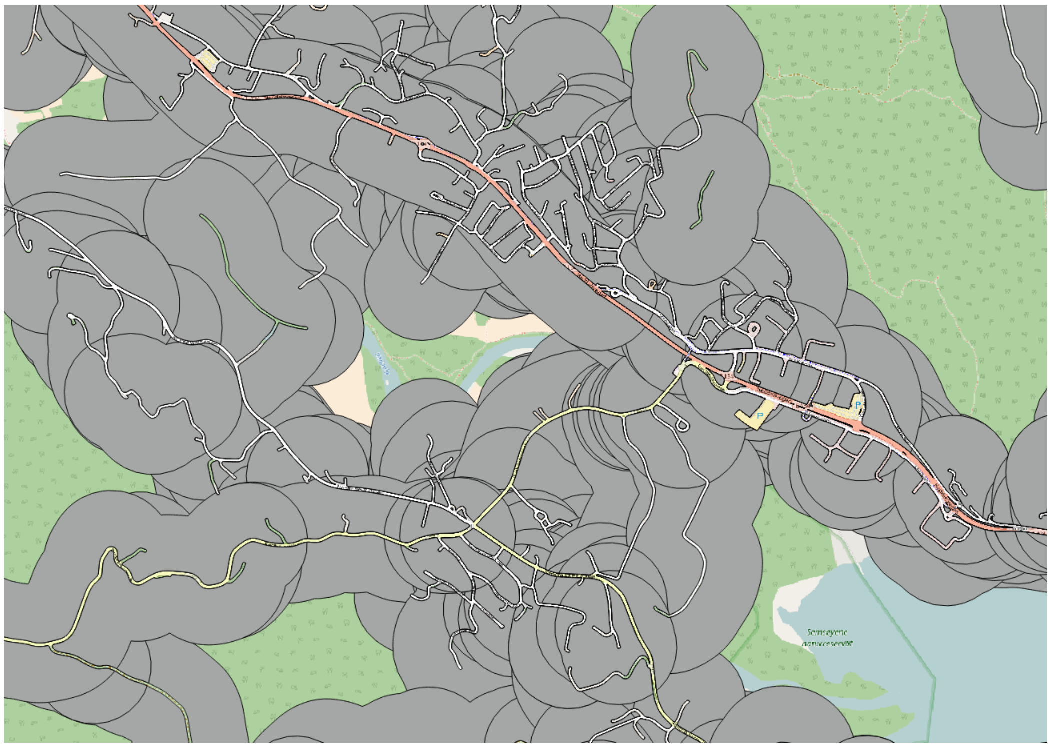


Figure 3.2: Buffer zone around all road segments ranging from 5 to 250 meters.

The next steps involve creating buffer zones around existing water and protected buildings, as the new building can not be directly next to the protected buildings. Figure 3.3 shows the surrounding buffer zone on protected buildings in pink, the buffer zone of the water in blue and the Natural Reserve in green. In a complete

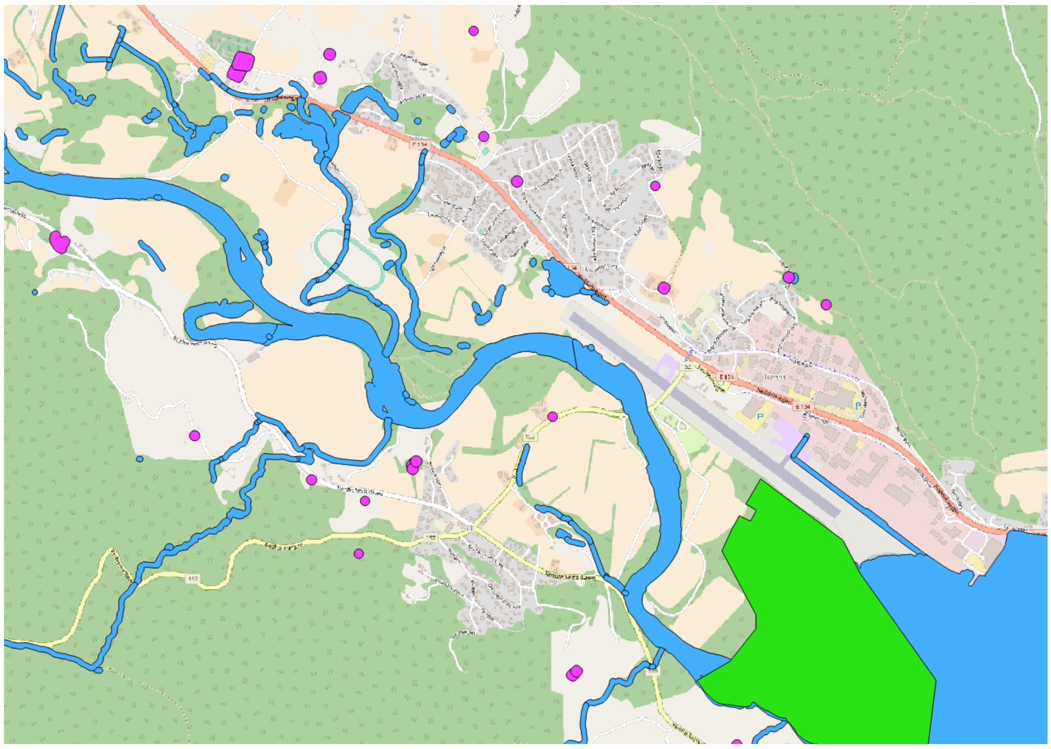


Figure 3.3: Polygons covering a Natural Reserve (green) and buffer zones around water segments (blue) and protected buildings (pink).

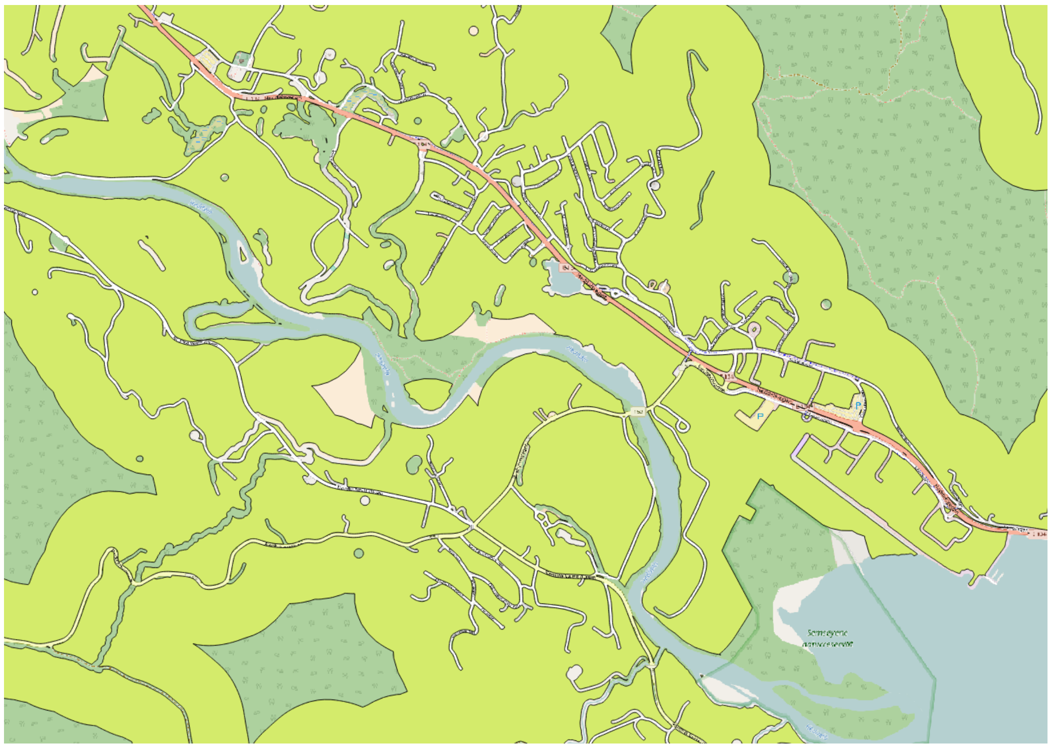


Figure 3.4: Valid building grounds (yellow) for a mall given the example specifications.

analysis there would probably be a lot more considerations to make, for instance the gradient of the field, possible danger zones, like flooding or land masses that contain a lot of quick clay and protected nesting zones, to mention a few. The complete list of consideration can only be determined by further investigations towards the potential users of the analyses.

Continuing with the example, all of the areas shown in Figure 3.3 are deleted from the preliminary area from Figure 3.2, which results in the yellow area shown in Figure 3.4. Given the prerequisites of this example, the yellow area represents all eligible places to build a large mall. Further analysis can also be implemented if the size of the mall is predetermined. With a size input function, it is possible to exclude all areas where the mall is unable to fit.

Sidewalk Planning

When planning to construct a sidewalk on an existing road, one of the main concerns is whether or not there are any buildings obstructing the path. If any building, residential or not, lies too close to the road, it prevents the possibility to simply construct the sidewalk next to the road. Having a simple scripted analysis could easily check whether or not there are any buildings obstructing the construction of a sidewalk.

Figure 3.5 shows a simple example of a rural main road with some buildings close by. In order to check whether or not there are any obstructions, one simply follows the same procedure as in the last example, by applying buffer zones to the two parts.

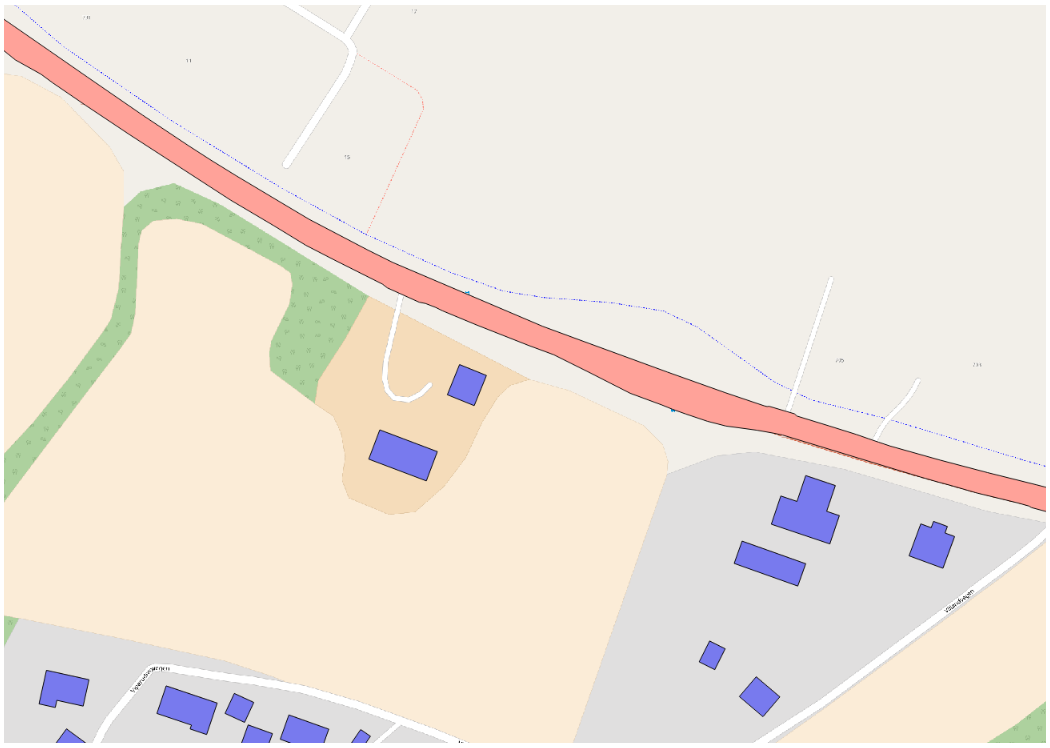


Figure 3.5: Rural main road marked in red, with buildings marked in blue.

In this example, the requirements are as follows:

- The sidewalk spans 2 meter outward from the road
- The outer side of the sidewalk must be at least 2 meters from existing buildings

The buffer around the house ensures that there is still room for a fence or a hedge between the sidewalk and the buildings, if desired.

Figure 3.6 shows the buffer zones around the buildings and the road. In this example there are no overlapping zones. In order to check this, QGIS has a function that checks for intersecting areas. This would result in a feedback telling the user that there are no buildings in the path of a sidewalk in the specified area. If, however, there was an obstruction, the output would show the specific area of the obstruction, and whether or not it is only an obstruction of a building's buffer zone, or if the building itself is within the path of the sidewalk.

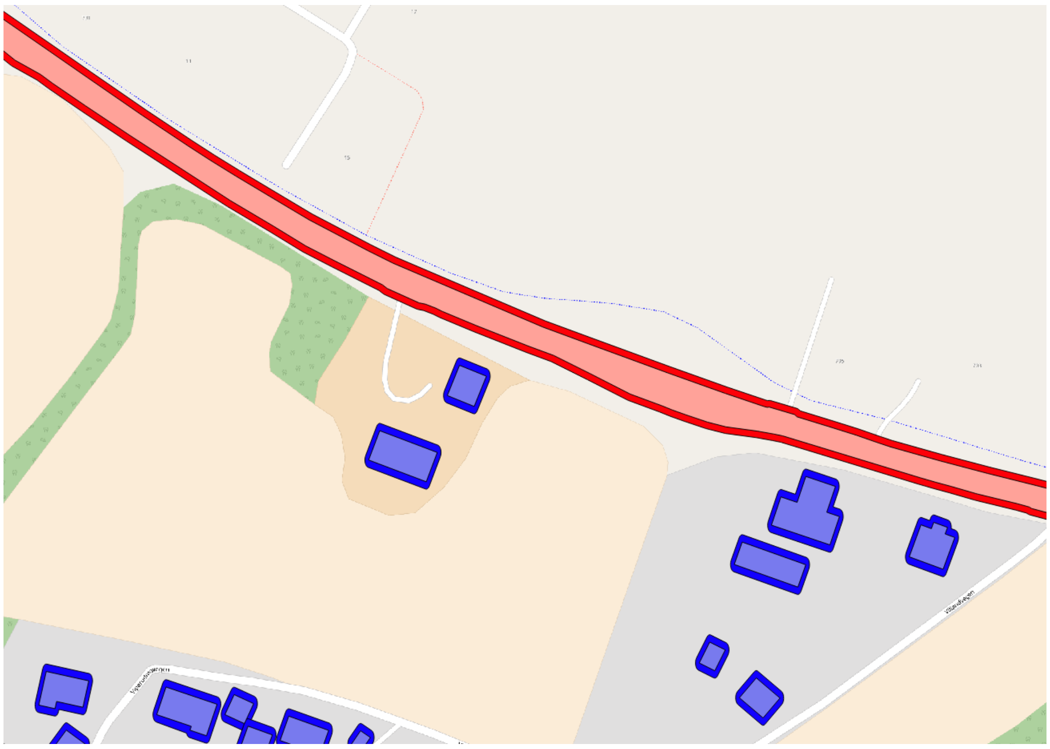


Figure 3.6: Rural main road marked in red, buffer in deep red, with buildings marked in blue, buffer in deep blue.

These types of analysis might not seem to be too difficult to perform without the help of scripted processes, but gathering the right information, collecting it within a GIS-program and performing the different analysis without getting any unforeseen errors can end up requiring a lot of time. As long as these types of analyses are set up and programmed properly, it has the potential to save a lot of unnecessary time usage.

3.2.2 Scripted Analyses with PyQGIS

This section explains the scripted versions of the examples described in the previous section. This is done with the use of Python programming with the extension PyQGIS. The scripts are made specifically for this thesis, in a way that they can be run as standalone applications without any other requirements than the users geographical data, as long as QGIS is installed on the computer as mentioned in Section 2.2.1.

Much like the model builders mentioned in Section 1.4, these scripted analyses behave like a modeled workflow. The difference is that the workflow code is hidden behind a simplified GUI, which makes the users able to strictly focus on the data they want to have analyzed. The GUI is also written in Python, and is kept as simple as possible to let the users efficiently utilize the scripts, as proposed in Section 1.4.3.

Suitable Building Grounds

Given the complexity of the example, the scripted process has to be created with hard-coded variables, meaning that someone with GIS experience and the knowledge of specific requirements for building these types of buildings has to create the process in the first place.

To fully preprogram the GIS-functionality in this example, the user is only asked to supply input data files specifying which one contains roads or other types of data. In this example, for simplicity, it is assumed that the data supplied is limited to the geographical building area in question, meaning that it will not require the user to enter a bounding box, limiting the area in question.

Figure 3.7 shows the result of the scripted analysis for the example Suitable Building Grounds. In the scripted analysis, there is no background image, as this would not be saved with the resulting data set. When comparing it to Figure 3.4, there is a clear similarity between the two. Any differences are strictly due to the example being analyzed in a different map projection, ref Section 1.2.1, in the scripted analysis than in the original analysis manually in QGIS.

The code to the script for the "Suitable Building Grounds" example is included in Appendix B.

Sidewalk Planning

The example concerning Sidewalk Planning is somewhat less complicated, as the only goal is to check whether or not two buffers intersect.

In order to preprogram the GIS-functionality in this example, the only requirement would be the input data of roads and housing. The buffer size in this example is hard coded into the script, but the script could be altered to accept a custom buffer size, with a default value if the user decline to enter a value.

When processing this example, there are no overlapping areas in the buffers, and therefore gives no error to the user. Should there be any overlapping areas, then the areas in question would be highlighted to the user, making it easier for the user to identify where the construction would have to be reconsidered.

Figure 3.8 shows the result of the scripted analysis for the example Sidewalk

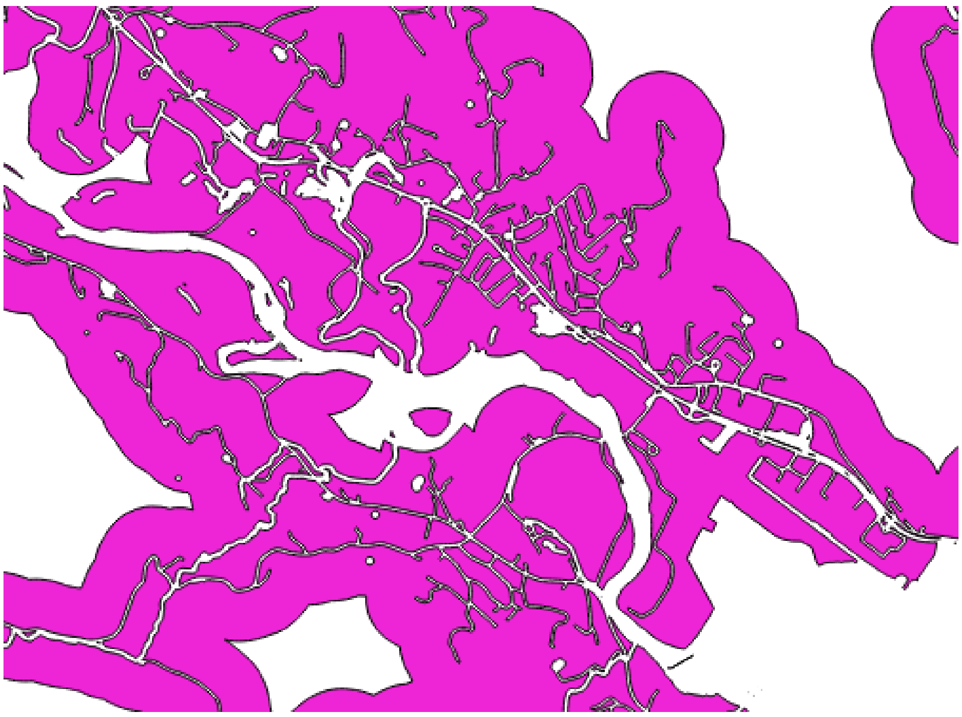


Figure 3.7: Result of the scripted analysis for the Suitable Building Grounds case. Same case as Figure 3.4.

Planning. Compared to Figure 3.6 it looks to be slightly different, but this is again strictly due to the scripted analysis needing to use a different map projection, ref Section 1.2.1, than the one originally used in the manual process in QGIS.

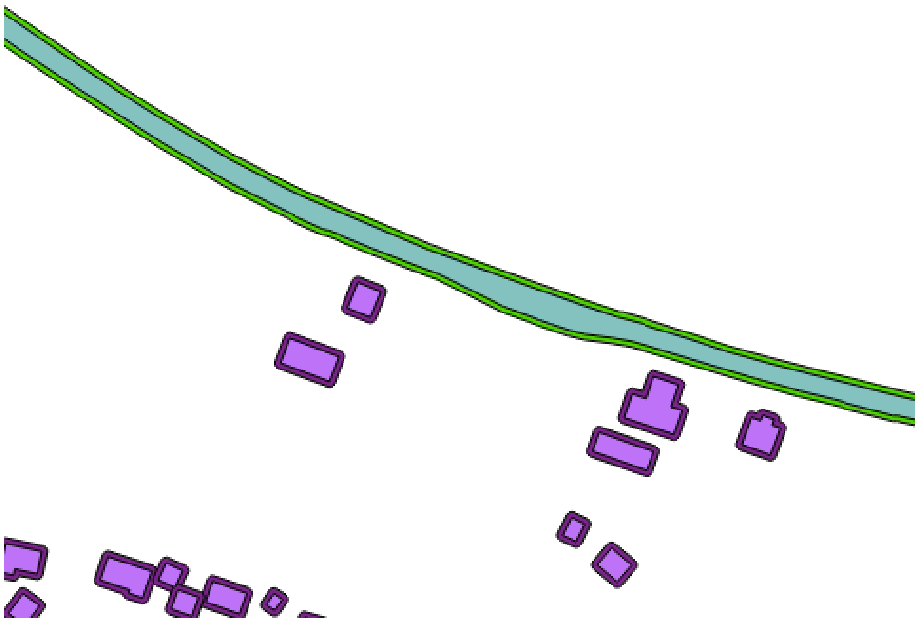


Figure 3.8: Result of the scripted analysis for the Sidewalk Planning case. Same case as Figure 3.6.

The code to the script for the "Sidewalk Planning" example is included in Appendix C.

3.2.3 Single Geographic Processing Tool

When having a custom need for a certain type of geographical processing tool, a standalone application can supply the user with a range of different options of tools to choose from, and then apply one or more specific processes, one by one, to a data file the user provides. When the user is done with processing the data file, the application is terminated. This makes it possible for the user to customize the processing requirements based on what the user is trying to achieve. Even though scripting is most efficient when given multiple tasks to handle in a single request, it may still be beneficial with single tasks for the more inexperienced GIS-user.

Appendix D contains the code for the "Single Geographical Processing Tool", where the user supplies an input data file, and chooses what type of process to apply to the data. Depending on what type of process the user choose, the program may ask for a second input data file, or the size of a few variables. The resulting data will be displayed for the user, and can be saved as a new data file in order to keep performing analysis or store the end result.

Chapter 4

Results

4.1 Survey

The participants were all familiar with acquiring map data, but not everyone were familiar with processing the data. Some of the participants informed that they mainly acquired the data, only to forward it, without performing any geographical analysis on their own. Of the participants, five worked as land-use planners, four worked as architects, and one as an engineer. Half of the participants previously had a job in the public sector, and two were in their first relevant job.

Out of the ten interviews, only four answered that it would be interesting to have fully scripted analysis, this includes both the participants in their first relevant job. On the other hand, only two out of ten answered that they would be interested in doing the simplified analysis themselves, and these two also answered yes to the one-click-analysis. It would seem that the users were more interested in getting the work done for them, instead of having more freedom to specify the analysis to their specific requirements.

4.1.1 Question Analysis

Using the Tables 3.1, 3.2 and 3.3, the results of the yes or no questions of the questionnaire are placed into the new Tables 4.1, 4.2 and 4.3.

Table 4.1: Outcome of Question 1 and 2.

Frequency	**Outcome**
7	Absolute satisfaction.
2	Indifference to efficiency, covers the necessary.
1	Dissatisfied with map solutions in general. Need for simpler solutions without more options.
0	Awareness of problems, obvious flaws in map solution.

Given that question one and two, listed in Section 3, are hypothesized to be inversely related, it would be reasonable to assume most of the results to be one yes and one no. Table 4.1 shows that there was no instances of anyone saying no to being satisfied and yes to something missing. There are, however, instances of every other outcome.

Table 4.2: Outcome of Question 3 and 4.

Frequency	**Outcome**
7	Absolute satisfaction.
2	Absolute need for extended map service.
1	Need for better service, want more in the easiest way possible.
0	Need for specific analysis, competent users who wants to be in control

Table 4.2 contradicts one of the main ideas of this thesis. Only 2 out of 10 wanted to be included in the choice of analysis, while 3 out of 10 just wanted a wider range of products or map data. The interest of being in control of the type of data to receive, and to be able to get help doing their own analysis, looks to be a lot less than first anticipated.

Table 4.3: Outcome of Question 5 and 6.

Frequency	**Outcome**
8	Absolute satisfaction.
2	Want better efficiency, willingness to try something new to get it.
0	Always interested in enhanced efficiency despite being satisfied with current time spent.
0	Maps in general feel complicated, not worth it to learn something new.

The last two questions were hypothesized to be directly correlated, and Table 4.3 shows that for this survey, every answer followed this hypothesis. Eight participants answered no to both questions about time consumption, and two participants answered yes.

Out of the 10 participants, 6 was determined to have absolute satisfaction on all three question pairs. However, every single participant had some remarks despite their satisfaction, but not enough to answer no to whether or not they were satisfied. Feedback like these remarks follows in Section 4.1.2.

4.1.2 Detailed Interviews

Not long before the interviews were conducted, Ambita had launched a new version of their online map solution store. This made the interviewees initially assume that most of the questions were related to the new version. Despite explaining that the questions were based on the general use of map solutions, there is reason to believe that the participants might have still been influenced by this initial assumption.

The most common feedback, is the need to gather all types of map data from several different sources, and how this would be a lot less hassle if everything were to be placed at a single source. Even though everyone questioned agreed that Ambita is better than any other map distributor in the Norwegian market, they still had problems understanding why it is such a difficulty getting all the necessary data to Ambita. Gathering everything is a goal for Ambita, but per usual, money and politics dictate the results.

When new height systems and file formats are created, it is usually to improve the quality of functionality, but converting every type of data into the new systems takes time. One user remarked that in some cases, it happens that the different sets of map data are created in different formats, which makes importing them into another software a tedious task.

4.2 User interface with the Application

Following the examples mentioned in Section 3.2, a total of 25 candidates who previously showed interest in the survey were invited to participate in a test of the scripted analyses. In the end, only five test users agreed to participate in the experiment, which was conducted individually and locally by the test users. All five test users were moderately experienced in performing GIS-analysis. The users got the specifications as listed in the examples, and were supposed to perform the geographical analyses manually in QGIS, and then with the assistance of the scripts described in Section 3.2.2.

When recording the time spent using both the methods of analysis, it is assumed that the user already have the geographical data for use in the analysis. The time only reflect the amount of time the users are interacting with the program or script, and not the time spent waiting for the processes to complete, as these processes can take a variable amount of time based on different background processes running on the computer at the time of execution and the software itself, as mentioned by Sipe and Dale (2003) referenced in the introduction of this thesis.

The raw data from the results in the following subsections are presented in Appendix E.

4.2.1 Fully Scripted Analyses

In the example **Suitable Building Grounds**, outlined in Section 3.2.2, the average user spent 57.9% less time conducting the analysis using the script instead of manually using QGIS, with a minimum of 51.5% and a maximum of 64.8%.

Test users informed that even though using the script on its own is faster, what is even more beneficial is that the processes are all running at once. This eliminates the need to check if one process is done, before starting the next, given that every single process usually take quite some time to complete.

In the example **Sidewalk Planning**, outlined in Section 3.2.2, the average user spent 60.4% less time conducting the analysis using the script instead of manually using QGIS, with a minimum of 51.5% and a maximum of 68.1%.

Despite having less actions and datasets to handle, Sidewalk Planning had a greater improvement compared to Suitable Building Grounds. This was mostly due to the very short time needed to perform the scripted analysis of the smaller example.

4.2.2 Single Process Scripted Analyses

As mentioned in Section 3.2.3, it is possible to let the user perform scripted analyses by choosing type and variables for each process. As this type of scripting does not necessarily run several different types of processes in succession, it is going to require more user interaction.

The results of using the Single Geographic Processing Tool script, versus manual interaction, are displayed in Table 4.4.

Table 4.4: Percent performance increase with script on Single Process Analyses
**Remote buffer consists of more than one GIS-function when performed manually.*

Geographical tool	**Avg**	**Min**	**Max**
Buffer	27.4%	18.5%	34.5%
Remote buffer*	51.9%	45.2%	57.1%
Difference	28.5%	27.0%	30.2%
Clip	35.8%	28.6%	41.7%

The *Remote buffer* analysis is a combination of two *Buffer* analysis and one *Difference* analysis. As such, it requires more interactions in QGIS than the others, and resemble the *Fully Scripted* analyses from Section 4.2.1 more than the others.

The average performance increase of the remaining three analyses *Buffer*, *Difference* and *Clip* is 30.6%, with an average minimum of 24.7% and an average maximum of 35.5%.

Chapter 5

Discussion

This thesis started with the question of whether or not there is a growing market for scripted GIS-processes for geographical analyses, and how an intuitive GUI assisted application designed to perform these analyses could speed up the processes. The introduction chapter listed a few research questions, with the first one being:

1. How far has GIS become automated with the help of scripted workflows?

From Section 1.4 it shows that there is many successful attempts to use scripted workflows to decrease workload or time usage. Most of the studies automated only parts of the processes, either in the form of preprocessing geographical data or data acquisition, while others tried to recreate and automate entire workflows, for instance to determine unstable seismic zones, as mentioned in Section 1.4.2.

The second research question:

2 What kind of GIS-analyses do users want or need scripted?

is answered through the interviews. The survey was specifically designed to find an answer to this research question. The answer would further help determine what types of analyses to be included in the timed user tests in order to answer the third research question. The results of the survey and interviews might give some insight into what could be the next step to bringing scripted analyses into the commercial market. Automating processes is often not a question of "if", but rather a question about "when?" and "how?".

Considering Table 4.1 in context with Table 3.1, it shows that the single participant who answered no to being satisfied, also answered no to whether there was anything missing from the existing solutions. At the same time, the only two participants who answered that they were missing something from the solutions, also answered that they were satisfied with the solutions available. From this information it would seem that even though there are room for improvements in how to handle map data distribution, no one is clearly saying that the current system is unacceptable.

The option of having user interference on the map data, in order to choose their own simple analysis without having to use third party software, was met with excitement only by a few users. Table 4.2 shows that three out of ten participants wanted other types of map data, and only two of these thought it might be advantageous to be able to choose simple analysis on the data. These results could simply be the lack of information about what other types of map data that exist, and what that could be of benefit for the participants.

The result shown in Table 4.3 gives an indication that the participants are very aware of the amount of time they spend, compared to how the work has to be done. The fact that eight participants were totally satisfied could also mean that they are too comfortable with their current work routine, to be able to imagine how the workflow could improve without adding complexity to their routines.

Most of the participants explained that when it is already known how to do spatial analysis, they would feel more secure about doing the actual analysis themselves. In many cases, scripted analysis on a general level would be very difficult to set up, as it would require a lot of different tools and functions for every buyers different purposes. In the end, it could cloud up the product list, making it much more difficult for the customers to find what they actually need.

A different solution might be to choose a small set of standardized analyses that many different customers are known to conduct, creating fully scripted analyses for a given area that everyone can get to know and rely on, automating most of the process.

People are colored by their experience. If the interviews were conducted solely on newly educated, the answers might have been different since they would, hopefully, not yet be set in their ways. This thesis is a result of interviewing users with multiple years of experience, users who might not want to change their world to something they see as far away from their normal day to day work life. With that in mind, the two participants who were in their first relevant work after studies both answered yes to wanting fully scripted analysis as a way to decrease work load.

One other factor that could distort the results of these interviews, is the fact that some of the interviewed are contractors who get paid by the hour, and thus might not want the work to go faster, as they would loose income. One interviewee, who previously worked as self-employed, gave this as an explicit reason for why the work didn't go "too slow".

Map solutions that work fast, like Google Maps, where regular people can figure out the driving distance, which road to take and expected arrival time, should be part of the aim for the future. This service is popular and many people find it easy to use and a necessary tool. If there is a way to implement geographical analysis that is as practical as Google Maps, it would make for a lot of new and exciting analyses.

The results from the user interaction, mentioned in Section 4.2, answers the third research question:

3 How much time can a GIS-user save from using scripted workflows?

They show a significant increase in analysis speed, with both the fully scripted analysis examples reducing more than 50% of the time spent interacting with the program. This was also the case for the only scripted multi-process analysis tool, *Remote buffer*, that consisted of multiple single analysis tools. The reason being when using this scripted process, there is only one action needed to set off the analyses, while in QGIS, several actions are required in order to reach the same goal.

Looking at the single process scripted analyses, there is also an increase in performance, but a lot less than for the fully scripted analyses. If these types of analysis tools require nearly the same amount of actions in both the scripted program as in QGIS, then why is there still an average increase of 30.6%? The test users mentioned the user interface as the deciding factor, as both loading the data sets and performing the analysis required a lot less clicks and searching in menus. The actual time spent in these analyses were all very low to begin with, meaning that a slight variation will generate a large outcome in percentage. All in all, the test subjects didn't feel that the single process scripted analyses gave them any significant advantage over the manual way. While the users didn't feel it, the experiment proved it to be beneficial as expected from the background in Section 1.4.3. Another time saver with these single process analyses, which was not included in the experiment, is the lack of needing the open the full GIS-program to execute a minor task.

It would have been beneficial to have a larger pool of test users performing the analysis. With different groups of test users with different levels of expertise in GIS-analysis it would have been possible to see if there is a difference in how much scripted workflows could benefit the casual GIS user compared to a GIS expert. The test users who participated in this experiment were all moderately experienced with GIS-analysis, which is assumed to give results somewhere in between the inexperienced and the experts.

One downside to scripting and automation is of course the lesser need of human involvement. As a company starts automating more of their usual tasks, it is important to engage the workers such that they do not fear for their jobs which could lead them to stagger the advancement of the company. Many workplaces now have employees that have lost their tasks, but not their jobs. This gives them lots of free time on their hands, but instead of finding tasks that will accommodate the progress, some tend to work against it. For a company, such as Ambita, it is important to lend a hand during the transition from manual GIS-processing to automating processes. Companies in transition must give their employees the chance to evolve as part of the process, to help them develop new skills that the employee and employer finds useful.

Automating GIS-functions are a part of the evolution of maps. Maps have always evolved, and even though they have come far, they have not, and may never, reach their final form. As mentioned in Section 1.2, cartographers in 1935 felt humans were spoiled in regards to maps. If humans had been satisfied with maps drawn by hand, with all their inaccuracies, we wouldn't have come as far as we have today. If an analysis is requested routinely, it should become an automated process. In the short term this might not be cost-effective, but in the long term, this will provide developers more time on hand to develop tomorrows solutions.

While there were made several attempts to increase the number of participants both in the interview phase and the user test, it ended up being relatively small compared to what was initially expected. It is difficult to draw specific conclusions from the interviews based on occupation, as there are too many different occupations involved with a comparably small sample size. The same applies to age, as there were no clear indication to whether young or old had any particular pretense of the situation. Both of these categories could have been further explored with a larger test group.

Based on the detailed feedback from the interviews, every different occupation, and company, has their own specific needs when it comes to geographical analysis, making it difficult to find common ground when developing advanced automation in the GIS-sector.

Chapter 6

Conclusion

In this thesis, research was conducted on the current state of automation in GIS. An interview round gathered information about the map solution needs from customers of Ambita and a user test with scripted geoprocessing workflows, developed in this thesis, were conducted to answer the research questions from the introduction. The conclusions are as follows:

1. The amount of studies that utilize preprogrammed GIS-workflows have grown in recent years, as shown in Section 1.4. Most of the studies have chosen to automate the gathering of input data or script the preprocessing of geographical data, not entire geographical analyses.
2. In this thesis, the interviews revealed an interest for fully scripted analyses for complex tasks. The basic GIS-processes were left wanted handled manually, given that the interviewed users were familiar with their own GIS-usage.
3. Scripting some of the processes in geographical analysis can be beneficial, done the right way. It shows from the results of the user interface test in Section 4.2.1 that fully scripted analyses eliminate more than half the time spent performing them. And this does not even include the greater freedom to multi task, given that all the processes run in succession, without the need to check in every time a single process is done.

Even if it is possible to fully script or automate every aspect of a process, it will always be limited by economy. Automation is only worth pursuing if it can be used to limit the amount of money that has to be spent on the surrounding process. That being said, automating GIS-analysis is most beneficial to the users having to buy geographical analyses, and a lot less to the users generating the analyses as a paid per hour service.

There is a greater desire for fully scripted analyses from the interview phase, and the results show a significantly better efficiency with fully scripted analyses compared to simplifying single analysis. Based on this, there is a possibility to conduct a bigger testing round to see if the improvements are consistent, and if the improvements increase when the test users' GIS-abilities are less proficient.

Conducting a new survey of needs from map solution customers and developers could reveal which types of graphical analyses are the best candidates for automation by rating them based on complexity and cost-to-benefit ratio.

Bibliography

Abhijit Ogale, Christian Frueh, Carole Dulong, Daniel Filip, Luc Vincent, Josh Weaver, Dragomir Anguelov, Stephane Lafon, and Richard Lyon (2010). Google Street View: Capturing the World at Street Level. *Computer*, 43:32–38.

Aguirre, J. C. (2014). The Unlikely History of the Origins of Modern Maps. *Retrieved (2022, January 26) from https://www.smithsonianmag.com/history/unlikely-history-origins-modern-maps-180951617/.*

Albrecht, J. (2004). Choosing a projection. *Retrieved (2022, January 26) from http://www.geo.hunter.cuny.edu/˜jochen/gtech201/lectures/lec6concepts/map%20coordinate%20systems/how%20to%20choose%20a%20projection.htm.*

Altaweel, M. (2017). The Use of Python in GIS. *Retrieved (2022, January 29) from https://www.gislounge.com/use-python-gis/.*

Altaweel, M. (2020). Python and Geospatial Analysis. *Retrieved (2022, January 27) from https://www.gislounge.com/python-and-geospatial-analysis/.*

ArcGIS (2004). *ESRI ArcMapTM 9.0 license type: ArcInfo.* ESRI Inc.

Bahgat, K. (2015). *Python Geospatial Development Essentials.* Packt Publishing Ltd.

Baykal, T. M. and Colak, H. E. (2022). Producing climate boundary maps using GIS interface model designed with Python. *Progress in Physical Geography*, 46(1):61–83.

Berry, P., Bonduá, S., Bortolotti, V., Cormio, C., and Vasini, E. M. (2014). A GIS-based open source pre-processor for georesources numerical modeling. *Environmental Modelling and Software*, 62:52–64.

Blackburn, D. and Holister, G. (1988). *Vitenskapens verktøy (Encyclopedia of Modern Technology).* Norsk Fogtdal, [Oslo].

Borna, K., Rajabi, M. R., Hamrah, M., Mansourian, A., and Ebrahimi, F. (2011). Design and development of an event-based model for geo-spatial information systems in accordance to the concept of dynamic version. *International Review on Computers and Software*, 6(1):140–149.

Brennan, J. (2020). Linking python with PostGIS. *Retrieved (2022, January 28) from https://james-brennan.github.io/posts/postgis_python/.*

Briney, A. (2021). What Is an Atlas? *Retrieved (2022, January 25) from https://www.thoughtco.com/what-is-an-atlas-1435685.*

Buckley, A. and Watkins, D. (2009). Automated Map Production Workflows. *Proceedings of the 24th International Cartographic Conference*, pages 1–11.

Che, Y., Yang, K., Jin, Y., Zhang, W., Shang, Z., and Tai, J. (2013). Residents' concerns and attitudes toward a municipal solid waste landfill: Integrating a questionnaire survey and GIS techniques. *Environmental Monitoring and Assessment*, 185(12):10001–10013.

Cosentino, G. and Pennica, F. (2015). QGIS geoprocessing model to simplify first level seismic microzonation analysis. *Retrieved (2022, February 14) from https://qgis.org/en/site/about/case_studies/italy_rome.html.*

Crump, E. (2015). Take a tour of Snowdonia - with Google Street View. *Retrieved (2022, January 26) from https://www.dailypost.co.uk/whats-on/trips-breaks/take-tour-snowdonia-google-street-10577102.*

Csáfordi, P., Pődör, A., Bug, J., and Gribovsyki, Z. (2012). Soil Erosion Analysis in a Small Forested Catchment Supported by ArcGIS Model Builder. *Acta silvatica & lignaria Hungarica*, 8(1):39–56.

Deetz, C. H. and Adams, O. S. (1921). *Elements of Map Projection with Applications to Map and Chart Construction.* US Government Printing Office.

Dempsey, C. (2012a). History of GIS. *Retrieved (2022, January 26) from https://www.gislounge.com/history-of-gis/.*

Dempsey, C. (2012b). QGIS versus ArcGIS. *Retrieved (2022, January 28) from https://www.gislounge.com/qgis-versus-arcgis/.*

Dobias, M. (2010). PyQGIS Documentation. *Release 1.4*, 1:3–10.

Dong, R. ., Sun, X. ., Li, C. ., Liu, T., and Gou, Y. . (2011). GIS model for eco-environmental sensitivity assessment of the areas along Qinghai-Tibetan railway. *Chinese Journal of Ecology*, 30(9):2093–2098.

Fernandez, P. and Moya-Delgado, S. (2017). Slopes geometric model: An automatic Pre-DInSAR analysis. *Environmental Modelling & Software*, 92:119–124.

G, Y. (2017). Which Python package manager should you use? *Retrieved (2022, January 30) from https://towardsdatascience.com/which-python-package-manager-should-you-use-d0fd0789a250.*

Gohari, A., Ahmad, H. A., Hashim, M. G., Kheirandish, S., and Kumar, L. (2012). Towards the design of GIS-based routing system. *International Journal of Geoinformatics*, 8(2):63–69.

Graser, A. and Olaya, V. (2015). Processing: A python framework for the seamless integration of geoprocessing tools in QGIS. *ISPRS International Journal of Geo-Information*, 4(4):2219–2245.

Jiménez-Perálvarez, J. D., Irigaray, C., El Hamdouni, R., and Chacón, J. (2009). Building models for automatic landslide-susceptibility analysis, mapping and validation in ArcGIS. *Natural Hazards*, 50(3):571–590.

Jordahl, K., den Bossche, J. V., Fleischmann, M., McBride, J., Wasserman, J., Badaracco, A. G., Gerard, J., Snow, A. D., Tratner, J., Perry, M., Farmer, C., Hjelle, G. A., Cochran, M., Gillies, S., Culbertson, L., Bartos, M., Ward, B., Caria, G., Taves, M., Eubank, N., Sangarshanan, Flavin, J., Richards, M., Rey, S., Maxalbert, Bilogur, A., Ren, C., Arribas-Bel, D., Mesejo-Leon, D., and Wasser, L. (2021). geopandas/geopandas: v0.10.2. Zenodo. https://doi.org/10.5281/zenodo.5573592.

Jøsang, A. (2016). *Subjective Logic: A Formalism for Reasoning Under Uncertainty.* Springer.

Lee, C.-L., Huang, S.-L., and Chan, S.-L. (2008). Biophysical and system approaches for simulating land-use change. *Landscape and Urban Planning*, 86(2):187–203.

Leigh, G. E. (2021). The High-precision Transcontinental Traverse: Improving the Scale of the U.S. Survey Network. *Retrieved (2022, February 20) from https://celebrating200years.noaa.gov/magazine/tct/welcome.html#survey.*

Levy, S. (2018). Graphical User Interface. Encyclopedia Britannica. *Retrieved (2022, February 20) from https://www.britannica.com/technology/graphical-user-interface.*

Lwin, K. K., Estoque, R. C., and Murayama, Y. (2012). Data Collection, Processing, and Applications for Geospatial Analysis. *Progress in Geospatial Analysis*, 9784431540:29–48.

Mai, T. (2012). Global Positioning System History. *Retrieved (2022, January 23) from https://www.nasa.gov/directorates/heo/scan/communications/policy /GPS_History.html.*

Mohtashami, S., Bergkvist, I., Löfgren, B., and Berg, S. (2012). A GIS approach to analyzing off-road transportation: A case study in Sweden. *Croatian Journal of Forest Engineering*, 33(2):275–284.

Nilsson, H. F. and Grundberg, D. (2009). *Plane-Based Close Range Photogrammetric Reconstruction of Buildings.* Master's thesis, Umeå University.

Olivera, F., Valenzuela, M., Srinivasan, R., Choi, J., Cho, H., Koka, S., and Agrawal, A. (2006). ArcGIS-SWAT: A geodata model and GIS interface for SWAT. *Journal of the American Water Resources Association*, 42(2):295–309.

OpenStreetMap-contributors (2021). Main Page - OpenStreetMap Foundation. *Retrieved (2022, January 26) from https://wiki.osmfoundation.org /w/index.php?title=Main_Page&oldid=9045.*

Pearson, F. (1990). *Map Projections: Theory and Applications.* Taylor & Francis.

Qgis (2013). PyQGIS Developer Cookbook. *Handbok.*

QGIS Development Team (2021). QGIS Desktop 3.16 User Guide.

Salkin, P. E. (2005). GIS in an Age of Homeland Security : Accessing Public Information to Ensure a Sustainable Environment. *William & Mary Environmental Law and Policy Review*, 30(1).

Sipe, N. G. and Dale, P. (2003). Challenges in using geographic information systems (GIS) to understand and control malaria in Indonesia. *Malaria Journal*, 2:1–8.

Skeie, K. and Gustavsen, A. (2021). Utilising open geospatial data to refine weather variables for building energy performance evaluation—incident solar radiation and wind-driven infiltration modelling. *Energies*, 14(4).

Smith, C. (2013). Google+ Is The Fourth Most-Used Smartphone App. *Retrieved (2022, January 26) from https://www.businessinsider.com/google-smartphone-app-popularity-2013-9?r=US&IR=T.*

Solheim, D. (2000). New height reference surfaces for Norway. *Symposium of the IAG*, 7:154–158.

Tomlinson, R. (1999). Origins of the Canada Geographic Information System. *Retrieved (2022, January 23) from https://www.esri.com/news/arcnews/fall12articles/origins-of-the-canada-geographic-information-system.html.*

Toms, S. (2015). *ArcPy and ArcGIS–Geospatial Analysis with Python.* Packt Publishing Ltd.

Toms, S., Rees, E. V., and Crickard, P. (2018). *Mastering Geospatial Analysis with Python.* Packt Publishing.

Werenskiold, W., Isachsen, F. E., and J.W. Cappelens forlag (1935). *Cappelens verdens atlas.* J.W. Cappelens forlag.

Whitehouse, D. (2000). BBC News - Ice Age star map discovered. *Retrieved (2022, January 26) from http://news.bbc.co.uk/1/hi/sci/tech/871930.stm.*

Appendix A

Survey

Background:

- How old are you?
- What is your occupation?
- How long have you had your current job?
- Where did you work before?

Use of maps:

- What kind of maps do you use?
- Are you satisfied with the currently existing map solutions?
- Do you miss anything in the list of current map solutions?
- Would it be beneficial to have other types of map data than you can get today?
    - o What other types of map data would you like to get?
- Do you want to be able to choose your own analysis of the data presented to you?
    - o What kind of analysis would you like to perform?
- Would you like to have access to fully automated geographical analysis?
    - o What kind of analysis would you like to have automated?
- Does it take too long to gain the information you need from maps in general?
    - o Why does it take too long?
- Would you want to gain information from maps faster than before?
    - o How many different places do you need to search for information?
- Does your company have a GIS-expert who is able to perform geographical analysis?

If yes:

- o Does this person work exclusively with GIS?
- o Have this person had this position a long time?

If no:

- o Do you feel the company could benefit from hiring a GIS-expert?

- Are there any map-related needs in your company that is not met?

Appendix B

```
'''
Scripted Analysis example Suitable Building Grounds
'''
from qgis.core import *
from qgis.gui import *
from qgis.utils import *
from PyQt4.QtCore import *
from PyQt4.QtGui import *


import processing
import tkinter as tk

from processing.core.Processing import Processing
from tkinter import filedialog
from tkFileDialog import askopenfilename

filename = ""
fileArray = []
def browse_limits():
    global fileArray
    fileArray.append(askopenfilename(initialdir = "/", title = "Select file",
                                     filetypes = (("Shape files", "*.shp"),
                                                  ("all files","*.*"))))
    button2 = tk.Button(text="Open data files containing limiting factors",
                        command=browse_limits, width=25,
                        height=5).grid(row=0, column=1)

def browse_roads():
    global master
    master = tk.Tk()
    global filename
    filename = askopenfilename(initialdir = "/",title = "Select file",
                               filetypes = (("Shape files", "*.shp"),
                                            ("all files","*.*")))
    root.destroy()
    button2 = tk.Button(text="Open data files containing limiting factors",
                        command=browse_limits, width=25,
                        height=5).grid(row=0, column=1)
    master.destroy()
    master.mainloop()

root = tk.Tk()
button = tk.Button(text="Open data file containing roads",
                   command=browse_roads, width=25,
                   height=5).grid(row=0, column=1)
root.mainloop()

app = QgsApplication([], True)
app.setPrefixPath("C:/Program Files/QGIS 2.18/apps/qgis-ltr", True)
app.initQgis()

Processing.initialize()
Processing.updateAlgsList()

canvas = QgsMapCanvas()
canvas.setWindowTitle("PyQGIS Standalone Application Example")
canvas.setCanvasColor(Qt.white)

print(fileArray[0])
print(fileArray[1])
layer1 = QgsVectorLayer(filename, "Veg", "ogr")
result = processing.runalg("qgis:reprojectlayer",layer1,"EPSG:25832", None)
layer1 = processing.getObject(result['OUTPUT'])
bufferLayer = layer1

layer2 = QgsVectorLayer(fileArray[0], "Limit", "ogr")
result = processing.runalg("qgis:reprojectlayer",layer2,"EPSG:25832", None)
layer2 = processing.getObject(result['OUTPUT'])

print(layer2.extent().toString())
print(layer2.featureCount())

for x in fileArray:
    if x == fileArray[0]:
        continue
    layer3 = QgsVectorLayer(x, "Limit", "ogr")
    result = processing.runalg("qgis:reprojectlayer",layer3,"EPSG:25832", None)
    layer3 = processing.getObject(result['OUTPUT'])
    unionLayer = processing.runalg("qgis:union",layer2,layer3, None)
    layer2 = processing.getObject(unionLayer['OUTPUT'])

result = processing.runalg("qgis:difference",bufferLayer,layer2,True,None)
finalLayer = processing.getObject(result['OUTPUT'])
QgsMapLayerRegistry.instance().addMapLayer(finalLayer)
```

```
canvas.setExtent(finalLayer.extent())
canvas.setLayerSet([QgsMapCanvasLayer(finalLayer)])
canvas.freeze(True)
canvas.show()
canvas.refresh()
canvas.freeze(False)
canvas.repaint()
exitcode = app.exec_()
QgsApplication.exitQgis()
sys.exit(exitcode)
```

Appendix C

```
'''
Scripted Analysis example Sidewalk Planning
'''
from qgis.core import *
from qgis.gui import *
from qgis.utils import *
from PyQt4.QtCore import *
from PyQt4.QtGui import *


import processing
import tkinter as tk

from processing.core.Processing import Processing
from tkinter import filedialog
from tkFileDialog import askopenfilename

roadFile = ""
housingFile = ""
def browse_housing():
    global housingFile
    housingFile = askopenfilename(initialdir = "/", title = "Select file",
                                  filetypes = (("Shape files", "*.shp"),
                                               ("all files","*.*")))
    global master
    master.destroy()

def browse_roads():
    root.destroy()
    global master
    master = tk.Tk()
    global roadFile
    roadFile = askopenfilename(initialdir = "/",title = "Select file",
                               filetypes = (("Shape files", "*.shp"),
                                            ("all files","*.*")))
    button2 = tk.Button(master, text="Open data file containing housing",
                        command=browse_housing, width=25,
                        height=5).grid(row=0, column=1)
    master.mainloop()

root = tk.Tk()
button = tk.Button(text="Open data file containing roads",
                   command=browse_roads, width=25,
                   height=5).grid(row=0, column=1)
root.mainloop()

app = QgsApplication([], True)
app.setPrefixPath("C:/Program Files/QGIS 2.18/apps/qgis-ltr", True)
app.initQgis()

Processing.initialize()
Processing.updateAlgsList()

canvas = QgsMapCanvas()
canvas.setWindowTitle("PyQGIS Standalone Application Example")
canvas.setCanvasColor(Qt.white)

layer1 = QgsVectorLayer(roadFile, "Veg", "ogr")
result = processing.runalg("qgis:reprojectlayer",layer1,"EPSG:25832", None)
layer1 = processing.getObject(result['OUTPUT'])
QgsMapLayerRegistry.instance().addMapLayer(layer1)
result2 = processing.runalg("qgis:fixeddistancebuffer",layer1,2,5,True,None)
bufferLayer = processing.getObject(result2['OUTPUT'])
QgsMapLayerRegistry.instance().addMapLayer(bufferLayer)

layer2 = QgsVectorLayer(housingFile, "Housing", "ogr")
result = processing.runalg("qgis:reprojectlayer",layer2,"EPSG:25832", None)
layer2 = processing.getObject(result['OUTPUT'])
QgsMapLayerRegistry.instance().addMapLayer(layer2)
result2 = processing.runalg("qgis:fixeddistancebuffer",layer2,2,5,False,None)
bufferLayer2 = processing.getObject(result2['OUTPUT'])
QgsMapLayerRegistry.instance().addMapLayer(bufferLayer2)

result = processing.runalg("qgis:clip",bufferLayer,bufferLayer2,None)
finalLayer = processing.getObject(result['OUTPUT'])
QgsMapLayerRegistry.instance().addMapLayer(finalLayer)

if finalLayer.featureCount() > 0:
    canvas.setExtent(finalLayer.extent())
else:
    canvas.setExtent(bufferLayer.extent())
canvas.setLayerSet([QgsMapCanvasLayer(finalLayer), QgsMapCanvasLayer(layer1),
                    QgsMapCanvasLayer(bufferLayer), QgsMapCanvasLayer(layer2),
                    QgsMapCanvasLayer(bufferLayer2)])
canvas.freeze(True)
canvas.show()
canvas.refresh()
canvas.freeze(False)
canvas.repaint()
exitcode = app.exec_()
QgsApplication.exitQgis()
sys.exit(exitcode)
```

Appendix D

```
'''
Single Geographic Processing Tool
'''
from qgis.core import *
from qgis.gui import *
from qgis.utils import *
from PyQt4.QtCore import *
from PyQt4.QtGui import *


import processing
import tkinter as tk

from processing.core.Processing import Processing
from tkinter import filedialog
from tkFileDialog import askopenfilename, asksaveasfile, asksaveasfilename

root = tk.Tk()
e1 = 0.0
e2 = 0.0
innerBufferDist = 0
outerBufferDist = 0
varDir = askopenfilename(initialdir = "/",title = "Select file",
                         filetypes = (("shape files","*.shp"),
                                      ("all files","*.*")))
varDir2 = ""
processType = ""
def set_buffer_dist():
    global outerBufferDist
    global innerBufferDist
    global e1
    outerBufferDist = int(e1.get())
    global e2
    if (e2 != 0.0):
        innerBufferDist = int(e2.get())
    global master
    master.destroy()
```

```
def get_difference_layer():
    global varDir2
    varDir2 = askopenfilename(initialdir = "/",title = "Select file",
                              filetypes = (("shape files","*.shp"),
                                           ("all files","*.*")))

def buffer_tool():
    root.destroy()
    global master
    master = tk.Tk()
    tk.Label(master, text="Buffer size: ").grid(row=0)

    global e1
    e1 = tk.Entry(master)
    global outerBufferDist
    outerBufferDist = e1.get()

    e1.grid(row=0, column=1)

    tk.Button(master, text='Next',
              command=set_buffer_dist).grid(row=3, column=0,
                                            sticky=tk.W, pady=4)

    master.mainloop()

def difference_tool():
    root.destroy()
    global master
    master = tk.Tk()
    global varDir2
    varDir2 = askopenfilename(initialdir = "/",title = "Select file",
                              filetypes = (("shape files","*.shp"),
                                           ("all files","*.*")))
    global processType
    processType = "difference"
    master.destroy()
    master.mainloop()

def buffer2_tool():
    root.destroy()
    global master
    master = tk.Tk()
    tk.Label(master, text="Inner limit: ").grid(row=0)
    tk.Label(master, text="Outer limit: ").grid(row=1)

    global e1
    e1 = tk.Entry(master)
    global e2
    e2 = tk.Entry(master)
    global innerBufferDist
    innerBufferDist = e2.get()
    global outerBufferDist
    outerBufferDist = e1.get()

    e2.grid(row=0, column=1)
    e1.grid(row=1, column=1)

    tk.Button(master, text='Next',
              command=set_buffer_dist).grid(row=3, column=0,
                                            sticky=tk.W, pady=4)

    master.mainloop()

def clip_tool():
    root.destroy()
    global master
    master = tk.Tk()
    global varDir2
    varDir2 = askopenfilename(initialdir = "/",title = "Select file",
                              filetypes = (("shape files","*.shp"),
                                           ("all files","*.*")))
    global processType
    processType = "clip"
    master.destroy()
    master.mainloop()

tk.Label(root, text="""Choose a geographical analysis tool:""",
         justify = tk.LEFT, padx = 20).pack()
tk.Button(root, text="Buffer", command=buffer_tool,
          width=25).pack(anchor=tk.W)
tk.Button(root, text="Difference", command=difference_tool,
          width=25).pack(anchor=tk.W)
tk.Button(root, text="Remote Buffer", command=buffer2_tool,
          width=25).pack(anchor=tk.W)
tk.Button(root, text="Clip", command=clip_tool,
          width=25).pack(anchor=tk.W)

root.mainloop()

app = QgsApplication([], True)
app.setPrefixPath("C:/Program Files/QGIS 2.18/apps/qgis-ltr", True)
app.initQgis()

Processing.initialize()
Processing.updateAlgsList()

canvas = QgsMapCanvas()
canvas.setWindowTitle("PyQGIS Standalone Application Example")
canvas.setCanvasColor(Qt.white)

if (e1 != 0.0):
    layer1 = QgsVectorLayer(varDir, "Punktdata", "ogr")
    result = processing.runalg("qgis:reprojectlayer",layer1,"EPSG:25832", None)
    layer1 = processing.getObject(result['OUTPUT'])
    result1 = processing.runalg("qgis:fixeddistancebuffer",layer1,
                                outerBufferDist,5,True,None)
    bufferLayer1 = processing.getObject(result1['OUTPUT'])
    result2 = processing.runalg("qgis:fixeddistancebuffer",layer1,
                                innerBufferDist,5,True,None)
    bufferLayer2 = processing.getObject(result2['OUTPUT'])
    result3 = processing.runalg("qgis:difference",bufferLayer1,
                                bufferLayer2,True,None)
    bufferLayer = processing.getObject(result3['OUTPUT'])
    QgsMapLayerRegistry.instance().addMapLayer(layer1)
    QgsMapLayerRegistry.instance().addMapLayer(bufferLayer)

elif (processType == "difference"):
    print(varDir)
    print(varDir2)
    layer1 = QgsVectorLayer(varDir, "Punktdata", "ogr")
    if not layer1.isValid():
        print "Layer %s did not load" % layer1.name()
    projectLayer1 = processing.runalg("qgis:reprojectlayer",layer1,
                                      "EPSG:25832", None)
    layer1 = processing.getObject(projectLayer1['OUTPUT'])
    if not layer1.isValid():
        print "Layer %s did not load" % layer1.name()
    layer2 = QgsVectorLayer(varDir2, "Punktdata2", "ogr")
    if not layer2.isValid():
        print "Layer %s did not load" % layer2.name()
    projectLayer2 = processing.runalg("qgis:reprojectlayer",layer2,
                                      "EPSG:25832", None)
    layer2 = processing.getObject(projectLayer2['OUTPUT'])
    if not layer2.isValid():
        print "Layer %s did not load" % layer2.name()
    print("test")
    resultLayer = processing.runalg("qgis:difference",layer1,layer2,True,None)
    print("test2")
    bufferLayer = processing.getObject(resultLayer['OUTPUT'])
    QgsMapLayerRegistry.instance().addMapLayer(bufferLayer)
```

```
elif (processType == "clip"):
    print(varDir)
    print(varDir2)
    layer1 = QgsVectorLayer(varDir, "Punktdata", "ogr")
    if not layer1.isValid():
        print "Layer %s did not load" % layer1.name()
    projectLayer1 = processing.runalg("qgis:reprojectlayer",layer1,
                                      "EPSG:25832", None)
    layer1 = processing.getObject(projectLayer1['OUTPUT'])
    if not layer1.isValid():
        print "Layer %s did not load" % layer1.name()
    layer2 = QgsVectorLayer(varDir2, "Punktdata2", "ogr")
    if not layer2.isValid():
        print "Layer %s did not load" % layer2.name()
    projectLayer2 = processing.runalg("qgis:reprojectlayer",layer2,
                                      "EPSG:25832", None)
    layer2 = processing.getObject(projectLayer2['OUTPUT'])
    if not layer2.isValid():
        print "Layer %s did not load" % layer2.name()
    print("test")
    resultLayer = processing.runalg("qgis:clip",layer1,layer2,None)
    print("test2")
    bufferLayer = processing.getObject(resultLayer['OUTPUT'])
    QgsMapLayerRegistry.instance().addMapLayer(bufferLayer)

canvas.setExtent(bufferLayer.extent())
canvas.setLayerSet([QgsMapCanvasLayer(bufferLayer)])
canvas.freeze(True)
canvas.show()
canvas.refresh()
canvas.freeze(False)
canvas.repaint()
exitcode = app.exec_()
QgsApplication.exitQgis()
sys.exit(exitcode)
```

Appendix E

Table 6.1: Raw data for Suitable Building Grounds analysis

Automatic	Manual	Time saved (%)
00:01:40	00:03:26	51.5
00:01:59	00:04:46	58.4
00:02:01	00:05:44	64.8
00:02:37	00:05:32	52.7
00:02:39	00:06:57	61.9

Table 6.2: Raw data for Sidewalk Planning analysis

Automatic	Manual	Time saved (%)
00:00:32	00:01:06	51.5
00:00:46	00:01:50	58.2
00:00:52	00:02:43	68.1
00:00:57	00:02:31	62.2
00:01:23	00:03:40	62.2

Table 6.3: Raw data for Buffer analysis

Automatic	Manual	Time saved (%)
00:00:22	00:00:27	18.5
00:00:25	00:00:35	28.6
00:00:27	00:00:39	30.8
00:00:34	00:00:45	24.4
00:00:38	00:00:58	34.5

Table 6.4: Raw data for Remote Buffer analysis

Automatic	Manual	Time saved (%)
00:00:23	00:00:42	45.2
00:00:28	00:00:59	52.5
00:00:36	00:01:23	56.6
00:00:39	00:01:15	48.0
00:00:48	00:01:52	57.1

Table 6.5: Raw data for Difference analysis

Automatic	Manual	Time saved (%)
00:00:27	00:00:37	27.0
00:00:34	00:00:47	27.7
00:00:34	00:00:48	29.2
00:00:35	00:00:49	28.6
00:00:37	00:00:53	30.2

Table 6.6: Raw data for Clip analysis

Automatic	Manual	Time saved (%)
00:00:20	00:00:28	28.6
00:00:22	00:00:36	38.9
00:00:23	00:00:33	30.3
00:00:26	00:00:43	39.5
00:00:28	00:00:48	41.7