United States Universities Council on Geotechnical Education and Research
Special Project Grant 2015
Final Summary Report

geotech: Development of a Geotechnical Engineering Software Package in *R*

James Kaklamanos, Ph.D.

Assistant Professor
Department of Civil Engineering
Merrimack College
315 Turnpike Street, North Andover, MA 01845
Tel: 978-837-3401 | Fax: 978-837-5029
Email: KaklamanosJ@merrimack.edu

Project Start Date: 1 October 2014
Project End Date: 30 September 2015

Submitted to:
USUCGER Board of Directors
February 2016

Abstract

This report describes the result of the first special project grant awarded through the United States Universities Council on Geotechnical Education and Research (USUCGER). In this project, undertaken by Prof. Jim Kaklamanos at Merrimack College and his student Kyle Elmy, we developed a geotechnical engineering software package within the statistical language and environment R (Kaklamanos and Elmy, 2016a). The software package, aptly named *geotech*, has the capability to perform a range of geotechnical engineering computations, and is freely available for use in the geotechnical engineering field. In geotechnical engineering education, research, and practice, there is a range of technology available for performing engineering calculations, and this software package represents one additional option in the engineer's toolbox.

Table of Contents

1 Introduction

Ranging from geotechnical engineering education to professional practice, geotechnical engineers and students face similar decisions about the optimal methodology for performing engineering calculations. For example, they may perform hand calculations, create a spreadsheet using Microsoft Excel, create a Mathcad worksheet, or utilize commercial geotechnical engineering software available through their educational institution or company. A well-rounded engineer should be comfortable with multiple alternatives for completing a certain task. This project concerns the development of a geotechnical engineering software package within the statistical language and environment R (R Core Team, 2016). The software package, named *geotech* (Kaklamanos and Elmy, 2016a), has the capability to perform a range of geotechnical engineering computations, and is available for broad usage within geotechnical engineering education, research, and practice.

R is an object-oriented language and environment for statistical computing that is gaining popularity in the scientific and engineering communities. Derived from the S language and environment developed at Bell Laboratories, R is an open-source language that is freely available at http://www.r-project.org/ (R Core Team, 2016). R covers much of the functionality of MATLAB, which is more commonly used in engineering courses. However, because R is freely available and can be installed on a wide variety of computer platforms, individuals can install and use R without worrying about the hassles or costs associated with commercial software licenses. An additional benefit of R is the wide availability of add-on user-contributed software codes, referred to as "packages," which perform calculations in specific areas of study. Users of R can easily install and load these add-on packages once they have installed R on their computers. There are over 7000 user-contributed packages available in R (http://cran.r-project.org/), and the *geotech* package is the first to broadly cover the types of calculations that would be necessary in an introductory geotechnical engineering course (Kaklamanos and Elmy, 2016a). This project builds upon prior efforts by the PI, who is the author of the *nga* R package (Kaklamanos and Thompson, 2010), which implements the Next Generation Attenuation of Ground Motions (NGA-West 1) earthquake ground-motion prediction equations (Power et al., 2008).

R has a command-line interface that contrasts with the point-and-click type interface of other commonly used software programs. Users who have experience with the commercial software MATLAB generally find that the syntax of R is easy to learn due to the similarities. However, in order to use specific R packages such as the *geotech* package, it is not necessary to invest a great deal of time to become an expert in the language. All that is necessary is a basic understanding of some simple commands and an awareness of the extensive help documentation available within R. The quality and extent of the R documentation is unsurpassed by many other programs. The *geotech* package could therefore be reasonably incorporated into undergraduate geotechnical engineering courses without a large amount of additional instruction on R, and the package can be used by geotechnical engineering practitioners and researchers without requiring a herculean initial effort to understand the program.

2 Project Overview

The development of the *geotech* R package was performed at Merrimack College by Prof. Jim Kaklamanos (the Principal Investigator) and an undergraduate student research assistant, Mr. Kyle Elmy. Mr. Elmy graduated with a B.S. in Civil Engineering at Merrimack 2015, and has since continued at Merrimack to pursue an M.S. in Civil Engineering. During the summer between his undergraduate and graduate careers, Mr. Elmy worked full-time on the development of the software package in R, with his summer salary being supported by this USUCGER special project grant (see Section 5: Budget Information). He also continued this work during an independent study during the fall 2015 semester. This project offered this student with an opportunity to collaborate with the PI on a useful contribution to the field of geotechnical engineering. The results of this project are being presented at the 2016 American Society of Civil Engineers (ASCE) Geotechnical and Structural Engineering Congress in Phoenix, Arizona, in February 2016. The conference presentation and paper are entitled, "Development of a Geotechnical Engineering Software Package in R and Its Implementation in the Civil Engineering Curriculum" (Kaklamanos and Elmy, 2016b). The conference paper contains similar information to this project report, but also contains additional details beyond the scope of this report. The most useful resource, however, is the online help documentation associated with the *geotech* package.

The project was divided into three tasks, which are briefly described below, and expanded upon in the Results section of this report.

1. Identify and outline the geotechnical engineering calculations to be included.
2. Program and validate the geotechnical engineering functions in R.
3. Compose the help documentation and distribute the package.

2.1 Task 1: Identify and outline the geotechnical engineering calculations to be included

The first task was to identify and outline the geotechnical engineering calculations to include in the *geotech* R package. In the first version of the software package, an emphasis has been placed on calculations that arise within the context of an introductory geotechnical engineering course. These topics and the associated R functions are discussed in the Results section. In future versions of the *geotech* R package (beyond the scope of the current project), the functionality of the package could be extended to include more advanced calculations in the design of foundations and earth-retaining structures, and the analysis of slope stability.

2.2 Task 2: Program and validate the geotechnical engineering functions in R

Once the calculations were outlined, Mr. Elmy worked with the PI to write structured functions for the geotechnical engineering calculations in R. Users are able to perform calculations by calling the function name and specifying a list of arguments appropriate to each function, as described in the Results section. After the functions were written, validations were performed by comparing the R output against hand calculations and output from other programs (such as Microsoft Excel). To perform these validations, Mr. Elmy and the PI used course material from *Civil Engineering 3020 – Geotechnical Engineering* at Merrimack College, and cross-checked the results with additional examples from textbooks such as Coduto et al. (2011), Holtz et al. (2011), and Das and Sobhan (2014).

2.3 Task 3: Compose the help documentation and distribute the package

The final task was to compose an extensive library of help documentation, following the format specified in the R package guidelines. The help documentation includes some background on the purpose of each function, details on the input and output of each function, and unique examples. An important component of Task 3 is the dissemination of this software package to the broader community. Because the result of this project is a software package, the primary mode of distribution is the Internet (specifically via the R project website and/or the PI's research website, as described in Section 3.1). The presentation given by the PI at the American Society of Civil Engineers (ASCE) Geotechnical and Structural Engineering Congress will also help serve this purpose. The long-term goal is for geotechnical engineering students, professors, researchers, and practitioners across the world to have access to and benefit from the *geotech* package.

3 Results

In this section of the report, we provide details on the results of this project: the *geotech* software package in R. The availability of the software, its installation and initialization, and the functions and documentation of the software will be discussed. An example calculation session will also be provided, but users are referred to the help documentation associated with the project for more comprehensive details and examples.

3.1 The *geotech* R package

In this section, we provide an overview of how the *geotech* package is employed within the statistical language and environment R.

3.1.1 Availability and installation

R is available for download at the R project website: http://www.r-project.org/ (R Core Team, 2016). In this report, we provide some basic examples on using R (within the context of the *geotech* package), but a large amount of general documentation is available on the R project website, including details on how to install and become acquainted with R for the most common operating systems. The user can run R directly from the command-line interface once installed, or may choose to employ a user interface environment, such as RStudio (RStudio Team, 2016) or R Commander (Fox, 2005).

Once R has been installed on your system, the next step is to install the *geotech* package. This package can be installed by using the package manager within R (accessed via the drop-down menus), or by typing

```
> install.packages("geotech")
```

into the R command prompt. Commands intended to be entered at the R command line are indicated by lines that begin with the greater-than symbol. All functions in R have parentheses, and the function arguments, if any, are placed inside the parentheses; here, the argument is a package name, which is placed in quotes because it is a character string.

Alternately, the package may be installed by downloading the appropriate binary file (Windows or MacOS X) from the *geotech* R package website: http://cran.r-project.org/web/packages/geotech/index.html, which also can be accessed through the research website of the PI (http://www.kaklamanos.com). To install the package locally using the binary file, the relevant files are geotech_<version>.tgz for Unix variants (including Mac OSX), and geotech_<version>.zip for Windows operating systems. Download the appropriate file into a directory on your computer (do not decompress the file), and follow the relevant instructions below. To install the *geotech* R package on a Microsoft Windows system, select the drop-down menu Packages from the R GUI, and then select "Install package(s) from local .zip files." You will then be prompted to locate the .zip file on your computer. To install a package on Mac OS X, select the drop-down menu Packages & Data from within the Mac R GUI, click Package Installer, and select the Local Source Package option in the Packages Repository panel. To install the package on a Unix-based system at the R command prompt, type

```
> install.packages("XYZ/geotech/<version>.tar.gz, repos = NULL),
```

where XYZ indicates the path to the directory where the .tar.gz file is located.

Note that an additional file, geotech_<version>.tar.gz, is the package source file, which can be viewed or compiled on any operating system. In order for Windows users to unzip this file, we recommend the program 7Zip, which is available at http://www.7-zip.org/. An important use of the .tar.gz file is to view the source code for the calculations; once the file is unzipped, navigate to the R subfolder and open the respective source-code files. Alternatively, users may view the source code for a specific function by typing that function's name into the console (e.g. `install.packages` for the function used at the beginning of this paragraph).

3.1.2 Initialization and documentation

Once installed, the *geotech* package must be loaded into the active R session in order to access its functions. The user may load the package into the R session by accessing this option from the drop-down menu or by typing

```
> library(geotech).
```

For users to familiarize themselves with R and the *geotech* package, the command

```
> help.start()
```

will open the extensive HTML documentation for R. If you navigate to the "Packages" hyperlink, you will see a list of the packages currently installed on your system. Click on the *geotech* link to see a list of the functions that are available within the *geotech* package. The standard R help documentation for a function contains a short description of the function, its usage and syntax, a definition for each argument, a description of the value returned by the function, a series of examples for using the function, and various notes and references intended to make the function as transparent as possible to the user. When learning how to use a function, users may benefit from copying and pasting the example codes into the R command line to observe the output. A PDF version of the reference manual is also available at the R project website and the PI's research website; this contains the same information as the HTML documentation. Anyone interested in learning more about the *geotech* R package without actually installing R is encouraged to review the PDF reference manual, as it contains much more information than possible in this summary report.

3.1.3 Functions

The *geotech* package is composed of sets of functions that are used to perform calculations and create plots that commonly arise in geotechnical engineering. In this section, we provide an overview of the functions available within the initial version of the *geotech* package. Note that the current version of the *geotech* package has focused upon calculations that arise within the context of an introductory geotechnical engineering course. The types of calculations that are currently included are: (1) phase diagrams and index parameters, (2) grain-size distributions, (3) plasticity, (4) soil classification, (5) compaction, (6) groundwater, (7) subsurface stresses (geostatic and induced), (8) Mohr circle analyses, (9) consolidation settlement and rate, (10) shear strength, (11) bearing capacity, (12) lateral earth pressures, (13) slope stability, and (14) subsurface explorations. The latter four categories include basic applications to subsurface explorations, foundations, earth-retaining structures, and slope stability; future versions of the package may be extended to include more advanced calculations in geotechnical design. Table 1 provides an outline of the functions currently included with the *geotech* package.

Table 1. Outline of functions included with the *geotech* R package

Category	Names of functions	Description
Phase diagrams and index parameters	phase.plot, phase.params, waterContent, relDensity	Plot phase diagrams from weights (or masses) and volumes of a soil sample (phase.plot); calculate a comprehensive list of index parameters from the phase diagram (phase.params); calculate water content from lab results (waterContent); calculate relative density (relDensity).
Grain size distributions	grainSize.plot, grainSize.coefs, percentComponents, Dsize, size.from.sieve	Plot a soil's grain-size distribution (grainSize.plot), and perform associated calculations of percent gravel, sand, and fines (percentComponents), D-sizes (Dsize), and coefficients of uniformity and curvature (grainSize.coefs); determine the grain size corresponding to a specific sieve number (size.from.sieve).
Plasticity	LL, PI, LI, plasticity.plot, A.line	Calculate and plot a soil's liquid limit (LL) using the flow curve; calculate plasticity index (PI) and liquidity index (LI) from Atterberg limits and in-situ water content; plot a soil's plasticity parameters on Casagrande's plasticity chart (plasticity.plot); calculate the PI associated with the A-line on Casagrande's plasticity chart (A.line).
Soil classification	AASHTO, USCS, GI, USCS.coarse. symbol, USCS.fine.symbol	Classify a soil using grain-size and plasticity data using the American Association of State Highway and Transportation Officials (AASHTO) classification system and the Unified Soil Classification System (USCS). There are also some sub-functions that perform parts of the calculations; for example, determining AASHTO group index and USCS group symbols.
Compaction	gammaD.exp, gammaD.theo, compaction.calc, compaction.plot, relCompaction	Calculate the dry unit weight from experimental results (gammaD.exp) or theoretically from a specified degree of saturation and specific gravity (gammaD.theo, which is used for plotting the zero air voids curve); calculate a soil's compaction curve as well as its maximum dry unit weight and optimum moisture content (compaction.calc); plot a soil's compaction curve and the zero air voids curve (compaction.plot); calculate relative compaction (relCompaction).
Groundwater	kConstant, kFalling, kx, kz, wellFlow, wellDrawdown, kPump	Calculate hydraulic conductivity from constant head and falling head tests (kConstant, kFalling); calculate equivalent horizontal and vertical hydraulic conductivities for layered soils (kx, kz); calculate flow rates and groundwater levels for wells in confined aquifers, unconfined aquifers, and mixed aquifers (wellFlow, wellDrawdown); calculate hydraulic conductivities from pumping tests (kPump).
Subsurface stresses	sigmaZ, sigmaZ.profile, sigmaZ.plot, induced.point, induced.point.profile, induced.area, induced.area.profile	Calculate and plot total vertical stresses, effective vertical stresses, and pore water pressures (sigmaZ, sigmaZ.profile, sigmaZ.plot); calculate induced stresses due to point loads (induced.point, induced.point.profile) and area loads (induced.area, induced.area.profile). The functions ending in ".profile" calculate the variation of stresses with depth.
Mohr circle analyses	stressTrans, MohrCircle.calc, MohrCircle.plot, sigma13, tauMax	Calculate normal and shear stresses on an inclined plane (stressTrans); calculate and plot the Mohr circle (MohrCircle.calc, MohrCircle.plot); calculate principal stresses and their orientations (sigma13); calculate the maximum in-plane shear stress and its orientations (tauMax).

Category	Names of functions	Description
Consolidation settlement and rate	cons.plot, deltaPC, deltaSC, Tv.to.U, U.to.Tv, deltaPC.rate, rate.plot	Plot a soil's consolidation curve (cons.plot); calculate amounts of primary consolidation settlement (deltaPC) and secondary compression settlement (deltaSC); calculate the time factor from percent consolidation (and vice versa) using the simplified method for consolidation rate calculations (Tv.to.U, U.to.Tv); calculate and plot settlement versus time curves (deltaPC.rate, rate.plot).
Shear strength	MohrCoulomb.calc, MohrCoulomb.plot, directShear.calc, directShear.plot, triaxial.calc, triaxial.plot, UC.calc, UC.plot	Calculate and plot a soil's Mohr-Coulomb failure envelope with a known friction angle and cohesion (MohrCoulomb.calc, MohrCoulomb.plot); calculate and plot the results of a direct shear test (directShear.calc, directShear.plot); calculate and plot the results of a triaxial compression test (triaxial.calc, triaxial.plot); calculate and plot the results of an unconfined compression test (UC.calc, UC.plot).
Bearing capacity	bearingPressure, bearingCapacity, Nc, Nq, Ngamma	Calculate bearing pressure and ultimate bearing capacity for shallow foundations (bearingPressure, bearingCapacity), as well as theoretical bearing capacity factors (Nc, Nq, Ngamma).
Lateral earth pressure	K, Ko, Ka, Kp, sigmaX, sigmaX.profile, sigmaX.plot	Calculate at-rest, active, and passive lateral earth pressure coefficients using various methods (K, Ko, Ka, Kp); calculate horizontal stresses in the subsurface (sigmaX, sigmaX.profile); plot horizontal stresses versus depth (sigmaX.plot).
Slope stability	FSinf, FSplanar	Calculate factors of safety against shear failure on slopes using infinite slope analyses (FSinf) and planar failure analyses (FSplanar).
Subsurface explorations	soil.profile, N60, N160	Plot one-dimensional soil profiles (soil.profile) and perform corrections to standard penetration test data for field procedures (N_{60}) and effective overburden stress ($N_{1,60}$).

Table 1 is intended to serve as a broad overview of the calculations currently associated with the *geotech* package. Comprehensive details of these functions are available in the HTML help documentation associated with the *geotech* package, and in the PDF reference manual associated with the package. The user may access the HTML help documentation from the `help.start()` command mentioned earlier, or by typing

```
> help(FunctionName)
```

or

```
> ??FunctionName,
```

where `FunctionName` is the name of a specific function (such as `compaction.calc` or `sigma13`). Queries on specific topics (other than the names of functions) may be performed using the built-in help documentation in R by typing

```
> help.search("Topic"),
```

where Topic is the name of the topic for which you are searching (it is necessary to enclose your search in quotes). There is also a wide array of helpful Internet resources on R that may be accessed by any search engine.

3.2 Example sessions

In this section, we provide an example session of the R package, using data from a sieve analysis to classify a coarse-grained soil. This basic example is intended to illustrate typical usage in R, the flexibility of input and output, and some fundamental syntax and programming concepts that are helpful to know prior to starting R. Many additional examples are provided with the HTML help documentation available with the package.

For this example, we will analyze data from a sieve analysis to plot the grain-size distribution and classify a coarse-grained soil using the Unified Soil Classification System (ASTM 2011). The grain-size distribution, obtained from a laboratory sieve test, is given in Table 2. The function used for creating the grain size distributions is grainSize.plot, which has arguments sieve, size, percent, and metric. The documentation for function grainSize.plot informs us that either sieve or size must be used for the data on the horizontal axis; sieve is a list of the sieve numbers according to ASTM D422 (ASTM 2007) and size is the corresponding particle sizes in inches or millimeters. Only one of these parameters (sieve or size) needs to be entered. The percent argument gives the corresponding percentage of soil finer by weight, and metric is a logical variable (either TRUE or FALSE) to indicate whether metric or English units are used. The documentation for each function clearly indicates the required units of the arguments if there are any restrictions. In creating the *geotech* package, we have designed the inputs to the functions to be as flexible as possible (e.g., by allowing different types of input data [here, either the sieve numbers or the particle sizes must be specified] and different units).

Table 2. Grain-size distribution for Example 1.

Sieve No.	Grain size (mm)	Percent finer (%)
3/8-in	9.50	95.72
No. 4	4.75	90.23
No. 10	2.00	81.49
No. 20	0.850	66.36
No. 40	0.425	50.00
No. 140	0.106	8.61
No. 200	0.075	4.82

The user may define the input data as follows, using variable names of his or her choice:

```
> ## Input Parameters
> sieve.example <- c(3/8, 4, 10, 20, 40, 140, 200)
> percent.example <- c(95.72, 90.23, 81.49, 66.36, 50.00, 8.51,
>                      4.82)
```

The first line of code is a comment (comments in the R language are specified with the number sign, #), and the second and third lines assign data to the variables `sieve.example` and `percent.example`. Assignment in R is performed using the *assign* command: `<-` (i.e., an arrow-like symbol). An equal sign (=) may also be used to assign variables in most situations; however, the equal sign is usually used to assign arguments within function calls.

The variables `sieve.example` and `percent.example` are vectors, and vectorized values have been assigned using the "c" (concatenate) operator. Once assigned, individual elements of vectors or matrices may be accessed using brackets. For example, typing `percent.example[1]` would access the first element of the `percent.example` vector, and would output `95.72`. Note that R is a case-sensitive language, and typing a variable with different capitalization (such as `PERCENT.example`) would return an error.

An example function call for generating the grain-size distribution would be:

```
> grainSize.plot(sieve = sieve.example, size = NA,
>               percent = percent.example, metric = TRUE).
```

The order of the function call does not matter as long as each mandatory argument is named. One can save space by omitting the argument's name (and inputting the values directly), as in:

```
> grainSize.plot(sieve.example, NA, percent.example, TRUE).
```

When the argument names are omitted, the arguments are matched by their order, as listed in the function documentation. The argument `size` does not need to be specified (because `sieve` has been specified instead). Any arguments that are left unspecified will be assigned their default value, as described in the function's documentation. For the `grainSize.plot` function, the default value of `size` is `NA`, which indicates that no inputs have been assigned to this variable (provided that `sieve` has been assigned). Therefore, an acceptable function call would also be

```
> grainSize.plot(sieve = sieve.example,
                percent = percent.example, metric = TRUE).
```

When the function is called, the grain size distribution will appear in a new window, illustrated in Figure 1. This figure can be saved in a number of different formats by selecting "Save as" from the file menu. Alternatively, more advanced users can use the `grainSize.plot` function within a script to write the plot directly to a file, hence bypassing the output to the screen. To learn more about the `grainSize.plot` function, users may study the HTML or PDF help documentation, or they may view the source code for the function by typing the function name directly into the command prompt with no arguments:

```
> grainSize.plot.
```

The function may also be executed by pasting its source code into the R command terminal. Users are welcome to modify the source code of this function to further adapt the plot if desired; for example, to plot multiple curves on the same plot, change the titles, and/or to adapt the colors. The plotting functions in our software package are provided with the intention that they will serve as a base for users' specific applications.

Besides creating the plot, some additional calculations can be performed using the data from the grain-size distribution. For example, the percent components of gravel, sand, and fines may be computed using the `percentComponents` function; using the output from this function, we find that this sample is composed of 9.77% gravel, 85.41% sand, and 4.82% fines. The D-sizes of the distribution (corresponding to certain percent-passing values) may be calculated by the function `Dsize`. Finally, the coefficients of uniformity and curvature (C_u and C_c) are calculated by the `grainsize.coefs` function or from direct calculations using the output of the `Dsize` function: $C_u = 5.83$ and $C_c = 0.65$.

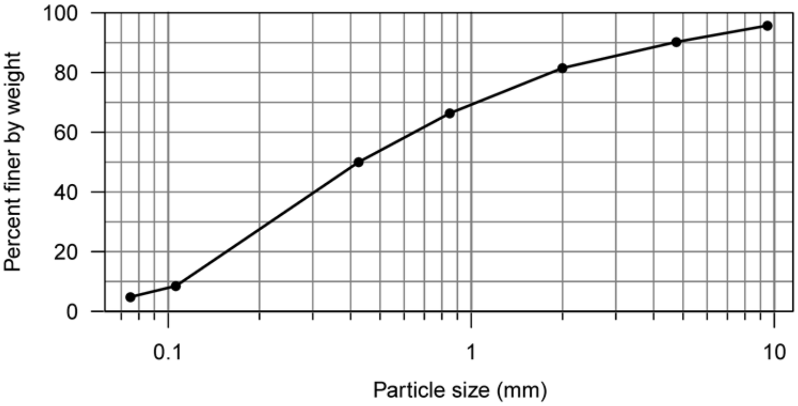

Figure 1. Grain-size distribution for Example 1.

To classify the soil according to the Unified Soil Classification System, the `USCS` function is employed. The function requires the grain-size distribution and/or plasticity data (i.e., the liquid and plastic limits), depending on the percentage of fines present in the sample. Because the soil sample has a fines content of less than 5%, only the grain-size distribution is necessary to classify this soil (Coduto et al. 2011). Either the raw grain-size distribution (using the variables `sieve` [or `size`] and `percent`) or the parameters calculated in the preceding paragraph (percent gravel, percent sand, percent fines, C_u, and C_c) must be supplied as inputs to the function.

The function calls

```
> USCS(sieve = sieve.example, percent = percent.example)
```

and

```
> USCS(pg = 9.77, ps = 85.41, pf = 4.82, Cu = 5.83, Cc = 0.65)
```

would give the same result. This function outputs a list with two elements: `symbol` and `name`, corresponding to the USCS group symbol and group name of the soil ("SP" and "Poorly-graded sand" for this example, respectively). If the percent fines in the sample were greater than 5%, the liquid and plastic limits would need to be specified to the `USCS` function; if they are omitted, an error message will result.

Several of the *geotech* functions (such as `USCS`) return lists of more than one element. In R, a list is a flexible data structure in which different elements of the list can have unequal lengths and different data types (character, numerical, logical, etc.), not unlike the different columns of a spreadsheet. For example, say we have assigned the output from the `USCS` function to a variable `Soil.Class`:

```
> Soil.Class <- USCS(sieve = sieve.example,
>                    percent = percent.example)
```

To see the names of the components of the list `Soil.Class`, type

```
> names(Soil.Class).
```

The "str" (structure) command provides additional information about the type and structure of an R object:

```
> str(Soil.Class).
```

The dollar sign ($) is used to access elements of a list; for example, typing `Soil.Class$symbol` would access the "symbol" element of the list, and would return "SP".

An additional example session is described in Kaklamanos and Elmy (2016b), and many more examples are available using the help documentation associated with the R package.

3.3 Implementation in the civil engineering curriculum

The *geotech* R package was incorporated into *Civil Engineering 3020 – Geotechnical Engineering* at Merrimack College during the fall 2015 semester. Civil Engineering 3020 is a required core course of all civil engineering majors, and is typically taken in the junior year. In this pilot implementation, students completed a comprehensive laboratory assignment in which they applied the software package on concepts learned throughout the semester. Ms. Amy Byrnes, a junior civil engineering major at Merrimack College, helped design the lab activity as part of her honors project. Following their initial exposure to the *geotech* R package in their junior year, students will be encouraged to continue using the software on assignments and projects in their geotechnical design electives during their senior year (e.g., Foundation Engineering and Earth Slopes and Retaining Structures), as well as in their capstone design project during the spring semester of their senior year. More details on the implementation of the *geotech* R package in the civil engineering curriculum are available in Kaklamanos and Elmy (2016b).

4 Summary

Geotech is an open-source, freely available software package for broad usage in geotechnical engineering education, research, and practice. The implementation of the *geotech* software package at Merrimack College helped improve students' understanding of geotechnical engineering concepts and served as an effective course review at the end of the semester. It is hoped that others may use the *geotech* package as an example for the development of R packages for other areas of civil engineering practice. In the current version of the software package, an emphasis has been placed on calculations that arise within the context of an introductory geotechnical engineering course. Because these fundamental geotechnical engineering calculations also arise frequently in professional practice and research, we anticipate that the broader geotechnical engineering community will also find this software to be a helpful contribution.

5 Budget Information

The entire project budget ($5,000) was allocated to the summer salary and fringe benefits of the undergraduate student researcher who worked on this project during summer 2015 (1 June 2015 to 31 August 2015). The student researcher was paid at a rate of $10/hr, and this grant supported 464.5 work hours, or approximately 13 weeks of full-time work at 35-36 hours per week. A summary of the expenditures are provided in Table 3.

Table 3. Summary of grant expenditures

Cost category	Amount
Student salaries	$4,644.68
Student fringe benefits (FICA) @ 7.65%	$355.32
Total expenditures	$5,000.00

The financial support of this project by USUCGER is gratefully acknowledged, as is the support of a faculty development grant from Merrimack College to partially support the PI during the summer in which this work occurred. We appreciate USUSCGER's dedication to advancing geotechnical engineering education and research, and we hope the results of this project help contribute towards that goal.

References

ASTM (2007). “Standard test method for particle-size analysis of soils.” *ASTM D422-63(2007)e2*. ASTM International, West Conshohocken, Pa.

ASTM (2011). “Standard practice for classification of soils for engineering purposes (Unified Soil Classification System).” *ASTM D2487-11*. ASTM International, West Conshohocken, Pa.

Coduto, D. P., M. R. Yeung, and W. A. Kitch (2011). *Geotechnical Engineering: Principles and Practices*, 2nd ed., Prentice Hall, Upper Saddle River, N.J., 794 pp.

Das, B. M., and K. Sobhan (2014). *Principles of Geotechnical Engineering*, 8th ed., Cengage Learning, Stamford, Conn., 768 pp.

Fox, J. (2005). “The R Commander: A basic-statistics graphical user interface to R.” *Journal of Statistical Software*, 14(9), doi: 10.18637/jss.v014.i09.

Holtz, R. D., W. D. Kovacs, and T. C. Sheahan (2011). *An Introduction to Geotechnical Engineering*, 2nd ed., Prentice Hall, Upper Saddle River, N.J., 864 pp.

Kaklamanos, J., and K.T. Elmy (2016a). “*geotech*: Geotechnical Engineering.” *R package version 1.0*, <http://cran.r-project.org/web/packages/geotech/index.html> (February 2016).

Kaklamanos, J., and K.T. Elmy (2016b). Development of a geotechnical engineering software package in R and its implementation in the civil engineering curriculum, *American Society of Civil Engineers (ASCE) Geotechnical and Structural Engineering Congress 2016*, Phoenix, Arizona, 14–17 February 2016.

Kaklamanos, J., and E. M. Thompson (2010). “*nga:* NGA earthquake ground motion prediction equations.” *R package version 1.5*, <http://cran.r-project.org/web/packages/nga/index.html> (February 2016).

Power, M., B. Chiou, N. Abrahamson, Y. Bozorgnia, T. Shantz, and C. Roblee (2008). An overview of the NGA project, *Earthquake Spectra* **24,** 3–21.

R Core Team (2016). “R: A language and environment for statistical computing.” R Foundation for Statistical Computing, Vienna, Austria, ISBN 3-900051-07-0, <http://www.R-project.org> (February 2016).

RStudio Team (2016). “RStudio: Integrated development for R.” RStudio, Inc., Boston, Mass., <http://www.rstudio.com/> (February 2016).