Development of a Geotechnical Engineering Software Package in R and Its Implementation in the Civil Engineering Curriculum

James Kaklamanos[1], Ph.D., A.M. ASCE, and Kyle T. Elmy[2], B.S.C.E., S.M. ASCE

[1] Assistant Professor, Department of Civil Engineering, Merrimack College, 315 Turnpike Street, North Andover, MA 01845; PH (978) 837-3401; FAX (978) 837-5029; email: KaklamanosJ@merrimack.edu
[2] Graduate Student, Department of Civil Engineering, Merrimack College, 315 Turnpike Street, North Andover, MA 01845; PH (978) 837-5299; FAX (978) 837-5029; email: ElmyK@merrimack.edu

ABSTRACT

This study describes the development and implementation of a geotechnical engineering software package within the statistical language and environment R. The software package, aptly named *geotech*, has the capability to perform a range of geotechnical engineering computations, and represents an additional option in the geotechnical engineer's toolbox. R is an object-oriented language and environment for statistical computing that is gaining popularity in science and engineering. In addition to the development and features of the software package, this study also describes its implementation in a junior-level introductory course in geotechnical engineering at Merrimack College in North Andover, Massachusetts. 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. The contribution of this project is an open-source, freely available software package for broad usage in geotechnical engineering education, research, and practice. A similar framework could be followed for the development of R packages for other areas of civil engineering practice.

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 study describes the development and implementation of a geotechnical engineering software package within the statistical language and environment R (R Core Team 2015). The software package, named *geotech* (Elmy and Kaklamanos 2015), has the capability to perform a range of geotechnical engineering computations, and

represents an additional option in the engineer's toolbox.

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 2015). 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 (Elmy and Kaklamanos 2015).

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. The purpose of this paper is to describe the development and implementation of the *geotech* software package in R. In addition to discussing the technical aspects of the software package, we also describe its implementation in a junior-level introductory course in geotechnical engineering at Merrimack College in North Andover, Massachusetts.

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.

Installation and Initialization

R is available for download at the R project website: http://www.r-project.org/ (R Core Team 2015). In this paper, 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 2015) 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. If users wish to view the source code for the calculations, they should visit the *geotech* package website (http://cran.r-project.org/web/packages/geotech/index.html) and download the package source file (with the file extension *.tar.gz). The source code is available in the R subfolder once the *.tar.gz file is unzipped.

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.

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	plot.phase, params.phase, waterContent, relDensity	Plot phase diagrams from weights (or masses) and volumes of a soil sample (plot.phase); calculate a comprehensive list of index parameters from the phase diagram (params.phase); calculate water content from lab results (waterContent); calculate relative density (relDensity).
Grain size distributions and plasticity	plot.grainSize, coefs.grainSize, percentComponents, Dsize	Plot a soil's grain-size distribution (plot.grainSize), and perform associated calculations of percent gravel, sand, and fines (percentComponents), D-sizes (Dsize), and coefficients of uniformity and curvature (coefs.grainSize).
Plasticity	LL, PI, LI, plot.plasticity	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 (plot.plasticity).
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, calc.compaction, plot.compaction, 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 (calc.compaction); plot a soil's compaction curve and the zero air voids curve (plot.compaction); calculate relative compaction (relCompaction).
Groundwater	kConstant, kFalling, kx, kz, wellConfined,	Calculate hydraulic conductivity from constant head and falling head tests (kConstant, kFalling);

Category	Names of functions	Description
Groundwater (cont.)	wellUnconfined, wellMixed, kPump	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 (wellConfined, wellUnconfined, wellMixed); calculate hydraulic conductivities from pumping tests (kPump).
Subsurface stresses	sigmaZ, sigmaZ.profile, plot.sigmaZ, 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, plot.sigmaZ); 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, calc.MohrCircle, plot.MohrCircle, sigma13, tauMax	Calculate normal and shear stresses on an inclined plane (stressTrans); calculate and plot the Mohr circle (calc.MohrCircle, plot.MohrCircle); calculate principal stresses and their orientations (sigma13); calculate the maximum in-plane shear stress and its orientations (tauMax).
Consolidation settlement and rate	plot.cons, deltaPC, deltaSC, Tv.to.U, U.to.Tv, deltaPC.rate, plot.cons.rate	Plot a soil's consolidation curve (plot.cons); 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, plot.cons.rate).
Shear strength	MohrCoulomb, plot.MohrCoulomb, directShear, plot.directShear, triaxial, plotTriaxial, UC, plotUC	Calculate and plot a soil's Mohr-Coulomb failure envelope with a known friction angle and cohesion (MohrCoulomb, plot.MohrCoulomb); calculate and plot the results of a direct shear test (directShear, plot.directShear); calculate and plot the results of a triaxial compression test (triaxial, plotTriaxial); calculate and plot the results of an unconfined compression test (UC, plotUC).
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, plot.sigmaX	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 (plot.sigmaX).
Slope stability	FSinf, FSplanar	Calculate factors of safety against shear failure on slopes using infinite slope analyses (FSinf) and planar failure analyses (FSplanar).

Category	Names of functions	Description
Subsurface explorations	plot.profile, N60, N160	Plot one-dimensional soil profiles (plot.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. The user may access this 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 `Tv.to.U` 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.

EXAMPLE SESSIONS

In this section, we provide two well-documented examples of calculations using the *geotech* package. The first example uses data from a sieve analysis to classify a coarse-grained soil, and the second example develops a subsurface stress profile at a site. These examples are 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.

Example 1: Grain-size analysis and soil classification

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 `plot.grainSize`, which has arguments `sieve`, `size`, `percent`, and `metric`. The documentation for function `plot.grainSize` 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 passing (%)
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:

```
> plot.grainSize(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:

```
> plot.grainSize(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 `plot.grainSize` 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

```
> plot.grainSize(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 `plot.grainSize` function within a script to write the plot directly to a file, hence bypassing the output to the screen. To learn more about the `plot.grainSize` function, users may study the HTML help documentation or view the source code for the function by typing the function name directly into the command prompt with no arguments:

```
> plot.grainSize.
```

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.

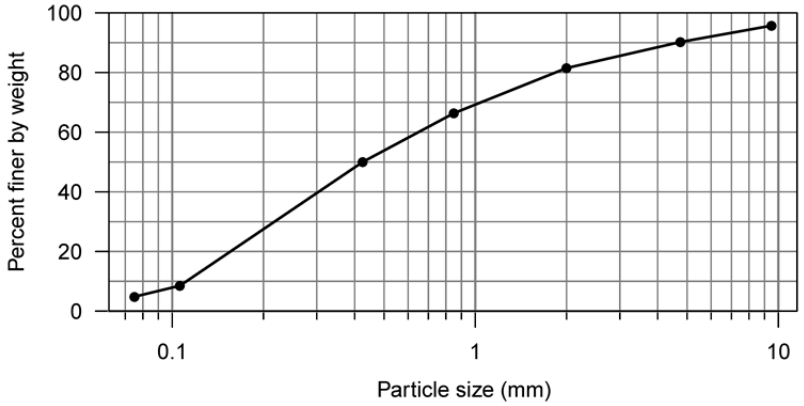

Figure 1. Grain-size distribution for Example 1.

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 `coefs.grainSize` function or from direct calculations using the output of the `Dsize` function: $C_u = 5.83$ and $C_c = 0.65$.

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".

Example 2: Subsurface stress calculations and profile

In this example, we will perform vertical geostatic stress calculations for a clay site with total unit weight γ_t = 108 lb/ft^3 above the groundwater table, saturated unit weight γ_{sat} = 116 lb/ft^3 below the groundwater table, and groundwater table located 15 feet below the ground surface. Stresses will be calculated to a total depth of 40 feet. To calculate the variation of total vertical stress, effective vertical stress, and pore water pressure with depth, the user may employ the `sigmaZ.profile` function. The arguments of this function are unit weights (`gamma`), layer thicknesses (`thk`) or depths (`depth`), the location of the groundwater table (`zw`), and the frequency of output (`zout`) in the same units as the specified depths or thicknesses. If `zout` is unspecified, the stresses will be calculated only at critical points in the profile (the top and bottom of the profile, layer interfaces, and the groundwater table). Note that if a user wanted to calculate the stresses only at a specific depth (i.e., not the variation of stresses with depth), then the user should employ the `sigmaZ` function, instead of the `sigmaZ.profile` function.

The command for executing the `sigmaZ.profile` function and saving the result to a variable named `vertical.stress` is provided below:

```
> vertical.stress <- sigmaZ.profile(gamma = c(108, 116),
>                                   depth = c(15, 40), zw = 15)
```

The output from the function is a four-element list containing vectors of depth, effective vertical stress, total vertical stress, and pore water pressure, respectively:

```
$depth
[1]   0    15    40

$sigmaZ.eff
[1]   0  1620  2960

$sigmaZ.total
[1]   0  1620  4520

$u
[1]   0     0  1560
```

In the output above, units of length are feet, and units of stress are pounds per square foot. Now that these calculations have been performed, the function `plot.sigmaZ` may be used in tandem with the `sigmaZ.profile` function to generate plots of vertical stress versus depth. The `plot.sigmaZ` function has arguments of depth (`depth`), effective stress (`sigmaZ.eff`), total stress (`sigmaZ.total`), pore water pressure (`u`), and a logical variable indicating the desired units for the plot (`metric`). The arguments `sigmaZ.total` and `u` are optional; if they are omitted, then the plot is generated only for vertical effective stress.

Using the prior calculation result, the command to create the plot of vertical stresses versus depth is:

```
> plot.sigmaZ(depth = vertical.stress$depth,
```

```
>              sigmaZ.eff = vertical.stress$sigmaZ.eff,
>              sigmaZ.total = vertical.stress$sigmaZ.total,
>              u = vertical.stress$u, metric = FALSE)
```

The resulting plot is provided in Figure 2. Once the plot is created, users can provide additional commands to modify the plot (for example, by adding additional curves for induced vertical stresses).

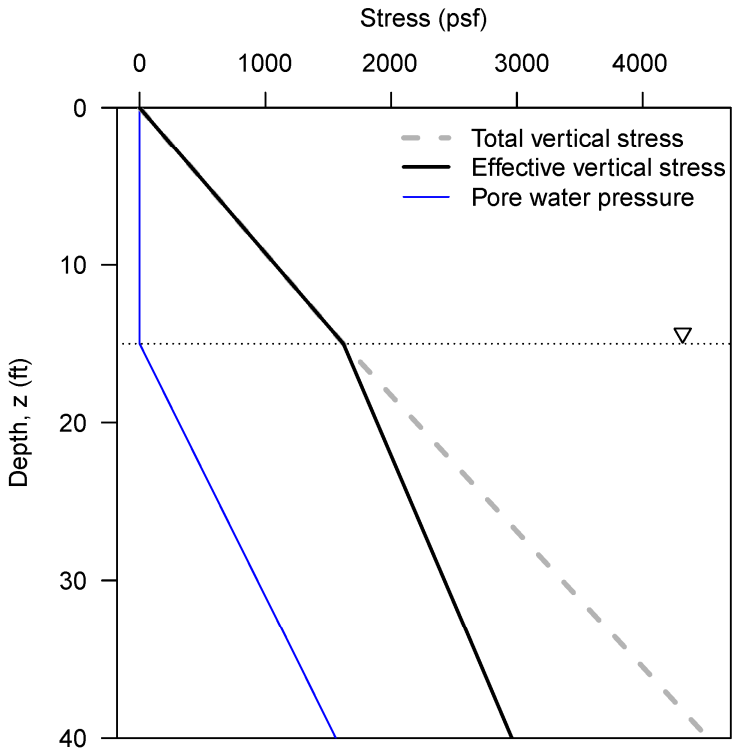

Figure 2. Vertical stress profile for Example 2.

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 planning the pilot implementation of the software into the course, two alternatives were considered:

1. Introduce the software at the beginning of the course, and include at least one problem on each homework assignment throughout the semester that requires students to use the software package.
2. Introduce the software at the end of the course, and have students complete a comprehensive assignment in which they apply the software package on concepts they have learned throughout the semester.

There are advantages and disadvantages to each curricular option. In the first option, students have the benefit of gaining practice with the software program for a longer

period of time throughout the semester. A disadvantage of this approach, however, is students may struggle with simultaneously learning curricular and programming concepts (e.g., mastering the fundamental underpinnings of the Mohr-Coulomb failure criterion while learning how to use the `plot.MohrCoulomb` function). In the second option, students use the software over a shorter period of time, but they have the benefit of having seen all the course material beforehand. For the fall 2015 semester, the latter option was chosen. Throughout the semester, students mastered the course material through traditional hand calculations and occasional usage of spreadsheets, and were then exposed to the *geotech* package through a comprehensive workshop at the end of the semester. By incorporating the software at the end of the semester, our intention was for students to view the software as less of a "black box" and more of a productivity tool. The software is not intended as a substitute for hand calculations (which we believe to be particularly important for students to learn engineering concepts), but rather as a complementary device that allows students to explore concepts and perform parametric studies in ways not possible with pencil and paper (or which would be extremely tedious to work out repetitively by hand).

The software activity took place during the final 2.5-hour lab session of the semester. Prior to the lab session, students were required to complete a reading assignment on the R language and the *geotech* package within R. At the beginning of the session, a 30-minute presentation was given on R, including demonstrations of common commands and the *geotech* R package. Students then completed a two-hour computer activity during lab that introduced them to R and the *geotech* package, and then followed up with an at-home assignment that allowed them to delve more deeply into the software package. The benefit of the timing of this activity is that it also served as an effective review of the course material, leading into the students' final exam preparation. Students were required to revisit course material from earlier in the semester and hopefully comprehend the linkages between various concepts.

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.

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.

ACKNOWLEDGMENTS

The development of the *geotech* R package was facilitated by a special project grant from the United States Universities Council on Geotechnical Education and Research (USUCGER) and a faculty development grant from Merrimack College; this support is gratefully acknowledged. Amy Byrnes (an undergraduate student at Merrimack College) also contributed to the testing of the software and the development of the end-of-the-semester lab activity in Civil Engineering 3020.

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

Elmy, K. T., and Kaklamanos, J. (2015). "*geotech*: Geotechnical Engineering." *R package version 1.0*, <http://cran.r-project.org/web/packages/geotech/index.html> (October 2015).

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.

R Core Team (2015). "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> (October 2015).

RStudio Team (2015). "RStudio: Integrated development for R." RStudio, Inc., Boston, Mass., <http://www.rstudio.com/> (October 2015).