Module Interface Specification for Slope Stability Analysis Program (SSP)

Henry Frankis and Brooks MacLachlan

November 20, 2018

1 Revision History

Date	Version	Notes
11/12/18	1.0	Initial updates based on template

2 Symbols, Abbreviations and Acronyms

See Section 2 of the SRS Documentation, available in the GitHub repository for the project.

Contents

3 Introduction

The following document details the Module Interface Specifications for SSP, a program for determining the critical slip surface and corresponding factor of safety for a given sloped mass of soil. The document is intended to ease understanding of the design of SSP and should be used as a resource for any maintenance of SSP.

Complementary documents include the System Requirement Specifications and Module Guide. The full documentation and implementation can be found at the GitHub repository for the project.

4 Notation

The structure of the MIS for modules comes from Hoffman and Strooper (1995), with the addition that template modules have been adapted from Ghezzi et al. (2003). The mathematical notation comes from Chapter 3 of Hoffman and Strooper (1995). For instance, the symbol $:=$ is used for a multiple assignment statement and conditional rules follow the form $(c_1 \Rightarrow r_1 | c_2 \Rightarrow r_2 | \ldots | c_n \Rightarrow r_n)$. For quantifiers, this document uses the word "with" to specify constraints on the bound variable.

The following table summarizes the primitive data types used by SSP.

Data Type	Notation	Description
character	char	a single symbol or digit
boolean	$\mathbb{B}$	a value from the set {true, false}
real	$\mathbb{R}$	any number in $(-\infty, \infty)$
integer	$\mathbb{Z}$	a number without a fractional component in $(-\infty, \infty)$

The specification of SSP uses some derived data types: sequences, strings, and tuples. Sequences are ordered lists of elements of the same data type, denoted by brackets enclosing the type of the data elements. If a sequence has fixed dimensions, the notation of the type will include the dimensions in superscript. Strings are sequences of characters. Tuples contain a list of values, potentially of different types, each associated with a field identifier. When a tuple is referenced in this document, a link to an appendix section that specifies the fields of the tuple will be provided. In addition, SSP uses functions, which are defined by the data types of their inputs and outputs. Local functions are described by giving their type signature followed by their specification.

5 Numerical Algorithms

Morgenstern-Price (Section 14)

The non-linear nature of the systems of equations in the Morgenstern-Price solver algorithm requires that the equations for the factor of safety (IM1), the interslice normal-to-shear force ratio (IM2), and the interslice normal forces (IM3) are solved iteratively, with an initial guess for two of the values, typically the factor of safety and interslice normal-to-shear force ratio.

Genetic Algorithm (Section 10)

SSP uses a genetic algorithm to find the coordinates of the critical slip surface vertices that minimize the factor of safety, as described in IM4. The genetic algorithm generates a set of initial potential slip surfaces, and subsequent generations are created by merging and mutating slip surfaces with low factors of safety from the previous generation. The minimum factor of safety after several generations is assumed to correspond to the critical slip surface.

[This section is not on the template. I've left it in for now because the information does seem useful, but maybe this is not the right place for it? Maybe this should go to an appendix? —BM]

6 Module Decomposition

The following table is taken directly from the Module Guide document for this project.

Level 1	Level 2
Hardware-Hiding	
Behaviour-Hiding	Control
	Input
	Output
	Genetic Algorithm
	Kinematic Admissibility
	Slip Weighting
	Slip Slicing
	Morgenstern-Price Calculation
	Slice Property Calculation
Software Decision	Sequence Data Structure
	Random Number Generation
	Plotting

Table 1: Module Hierarchy

7 MIS of the Control Module

7.1 Module

Control

7.2 Uses

Input (Section 8), Output (Section 9), GenAlg (Section 10), Sequence (Section 16)

7.3 Syntax

7.3.1 Exported Constants

N/A

7.3.2 Exported Data Types

N/A

7.3.3 Exported Access Programs

Name	In	Out	Exceptions
Control	string	-	-

7.4 Semantics

7.4.1 State Variables

N/A

7.4.2 Environment Variables

N/A

7.4.3 Assumptions

The access program is called with a string parameter.

7.4.4 Access Routine Semantics

control(*fname*):

- transition:

  Modifies the state of the Input Module, Genetic Algorithm Module, and Output Module.

7.4.5 Local Functions

N/A

8 MIS of the Input Module

8.1 Module

Input

8.2 Uses

Sequence (Section 16)

8.3 Syntax

8.3.1 Exported Constants

N/A

8.3.2 Exported Data Types

coord	=	tuple of (x : $\mathbb{R}$, y : $\mathbb{R}$)
coords	=	[coord]
paramsLayers	=	tuple of (strat : coords, phi : $\mathbb{R}$, coh : $\mathbb{R}$, gam : $\mathbb{R}$, gams : $\mathbb{R}$) (Appendix 19.1.1)
paramsPiez	=	tuple of (piez : coords, gamw : $\mathbb{R}$) (Appendix 19.1.2)
paramsSearch	=	tuple of (Xext, Xetr, Ylim : $[\mathbb{R}]^{1x2}$) (Appendix 19.1.3)
paramsSoln	=	tuple of (ltor, ftype, evnslc, cncvu, obtu : $\mathbb{B}$) (Appendix 19.1.4)

8.3.3 Exported Access Programs

Name	In	Out	Exceptions
load_params	string	-	fileNotExist, badFileExtension, unexpectedInput
verify_params	-	-	badSlopeGeometry, badEffAngleFriction, badCohesion, badDryUnitWeight, badSatUnitWeight, badPiezGeometry, badWatUnitWeight
strat	-	coords	-
slopeX	-	$[\mathbb{R}]$	-
slopeY	-	$[\mathbb{R}]$	-
phi	-	$\mathbb{R}$	-
coh	-	$\mathbb{R}$	-
gam	-	$\mathbb{R}$	-
gams	-	$\mathbb{R}$	-
piez	-	coords	-
piezX	-	$[\mathbb{R}]$	-
piezY	-	$[\mathbb{R}]$	-
gamw	-	$\mathbb{R}$	-
Xext	-	$[\mathbb{R}]^{1x2}$	-
Xetr	-	$[\mathbb{R}]^{1x2}$	-
Ylim	-	$[\mathbb{R}]^{1x2}$	-
ltor	-	$\mathbb{B}$	-
ftype	-	$\mathbb{B}$	-
evnslc	-	$\mathbb{B}$	-
cncvu	-	$\mathbb{B}$	-
obtu	-	$\mathbb{B}$	-

8.4 Semantics

8.4.1 State Variables

slope : paramsLayers
piez : paramsPiez
search : paramsSearch
soln : paramsSoln

8.4.2 Environment Variables

in_file : String

- *in_file* represents a file stored in the file system of the hardware running SSP.

8.4.3 Assumptions

- load_params is called before any of the other access programs.
- The guesses for potential minimum and maximum x and y values of the critical slip surface, as described in *in_file*, lie within the boundaries of the given slope geometry.

8.4.4 Access Routine Semantics

load_params(*fname*):

- transition:

  slope, *piez*, *search*, *soln* := *slope'*, *piez'*, *search'*, *soln'*
  where *slope'*, *piez'*, *search'*, and *soln'* are populated based on the contents of *in_file*.

- exceptions:

  exc := (*fname* does not exist in file system ⇒ fileNotExist
  | *fname*[(|*fname*| - 5)..(|*fname*| - 1)] = ".out" ⇒ badFileExtension
  | *in_file* is not formatted correctly ⇒ unexpectedInput)

verify_params():

- exceptions:

  exc := $(\neg(\forall i \in [0..|slope.strat| - 2])(slope.strat[i].x -$
  $slope.strat[i+1].x \leq 0) \Rightarrow$ badSlopeGeometry
  | $\neg(0 < slope.phi < 90) \Rightarrow$ badEffAngleFriction

$$
\begin{aligned}
&| \neg(0 < slope.coh) \Rightarrow \text{badCohesion} \\
&| \neg(0 < slope.gam) \Rightarrow \text{badDryUnitWeight} \\
&| \neg(0 < slope.gams) \Rightarrow \text{badSatUnitWeight} \\
&| \neg(\forall i \in [0..|piez.piez| - 2])(piez.piez[i].x - \\
&piez.piez[i+1].x \leq 0) \\
&\vee piez.piez[0].x \neq slope.strat[0].x \\
&\vee piez.piez[|piez.piez| - 1].x \\
&\neq slope.strat[|slope.strat| - 1].x \Rightarrow \text{badPiezGeometry})
\end{aligned}
$$

strat():

- output:

$$out := slope.strat$$

slopeX():

- output:

$$
\begin{aligned}
&out := slope.strat[0].x||slope.strat[1].x|| \\
&\ldots||slope.strat[|slope.strat| - 1].x
\end{aligned}
$$

slopeY():

- output:

$$
\begin{aligned}
&out := slope.strat[0].y||slope.strat[1].y|| \\
&\ldots||slope.strat[|slope.strat| - 1].y
\end{aligned}
$$

phi():

- output:

$$out := slope.phi$$

coh():

- output:

$$out := slope.coh$$

gam():

- output:

  $out := slope.gam$

gams():

- output:

  $out := slope.gams$

piez():

- output:

  $out := piez.piez$

piezX():

- output:

  $out := piez.piez[0].x||piez.piez[1].x||$
  $\ldots||piez.piez[|piez.piez| - 1].x$

piezY():

- output:

  $out := piez.piez[0].y||piez.piez[1].y||$
  $\ldots||piez.piez[|piez.piez| - 1].y$

gamw():

- output:

  $out := piez.gamw$

Xext():

- output:

  $out := search.Xext$

Xetr():

- output:

  $out := search.Xetr$

Ylim():

- output:

  $out := search.Ylim$

ltor():

- output:

  $out := soln.ltor$

ftype():

- output:

  $out := soln.ftype$

evnslc():

- output:

  $out := soln.evnslc$

cncvu():

- output:

  $out := soln.cncvu$

obtu():

- output:

  $out := soln.obtu$

8.4.5 Local Functions

N/A

9 MIS of the Output Module

9.1 Module

Output

9.2 Uses

Sequence (Section 16), Plot (Section 17)

9.3 Syntax

9.3.1 Exported Constants

N/A

9.3.2 Exported Data Types

N/A

9.3.3 Exported Access Programs

Name	In	Out	Exceptions
verify_output	$\mathbb{R}$	-	negativeFS
output	$\mathbb{R}$, coords, coords, coords, string	-	-

9.4 Semantics

9.4.1 State Variables

N/A

9.4.2 Environment Variables

out_file : String

- *out_file* represents a file stored in the file system of the hardware running SSP.

screen : $[\mathbb{Z}]$

- *screen* represents the colour values for each pixel on the screen of the hardware running SSP.

9.4.3 Assumptions

N/A

9.4.4 Access Routine Semantics

verify_output(*Fs*):

- exceptions:

  $exc := Fs < 0 \Rightarrow$ negativeFS

output(*Fs*, *crit_slip*, *G*, *X*, *fname*):

- transition:

  out_file is created at path *fname* || ".out". The outputs of Xetr(), Xext(), Ylim(), ftype(), *Fs*, *crit_slip*, *G*, and *X* are written to *out_file*. *screen* is modified to display the outputs of plot(*crit_slip.x*, *crit_slip.y*), plot(*G.x*, *G.y*), and plot(*X.x*, *X.y*).

9.4.5 Local Functions

N/A

10 MIS of the Genetic Algorithm Module

10.1 Module

GenAlg

10.2 Uses

10.2.1 Imported Access Programs

Input (Section 8), MorgPriceSolver (Section 14), Slicer (Section 13), KinAdm (Section 11), SlipWeighter (Section 12), Sequence (Section 16), Rand (Section 18)

10.3 Syntax

10.3.1 Exported Constants

MIN_GENS = 100
NUM_SLIPS = 20
REL_DIFF = 0.00005

10.3.2 Exported Data Types

slip	=	tuple of (surf : coords, Fs : $\mathbb{R}$, G : coords, X : coords, wt : $\mathbb{R}$)
slips	=	[slip]

10.3.3 Exported Access Programs

Name	In	Out	Exceptions
genetic_alg	-	$\mathbb{R}$, coords, coords, coords	-

10.4 Semantics

10.4.1 State Variables

N/A

10.4.2 Environment Variables

N/A

10.4.3 Assumptions

N/A

10.4.4 Access Routine Semantics

genetic_alg():

- output:

  out := slipWeighter(*MSlip*, *slip_surfs*)[0].surf, slipWeighter(*MSlip*, *slip_surfs*)[0].Fs, slipWeighter(*MSlip*, *slip_surfs*)[0].G, and slipWeighter(*MSlip*, *slip_surfs*)[0].X, where *slip_surfs*, of type slips, is developed by:

* using rand to randomly generate coordinates for NUM_SLIPS potential slip surfaces, where the entry and exit x-coordinate for each slip surface are computed according to *generate_slips(Xetr)* and *generate_slips(Xext)*. Corresponding y-coordinates are determined by interpolating on the slope geometry.
* using kinAdm to verify that the geometry of each potential slip surface is physically realizable. If any are not, new slip surfaces are randomly generated until NUM_SLIPS valid slip surfaces have been generated,
* using slicer to redefine each slip surface's coordinates based on the desired number of slices
* using morgPrice to determine the Fs, G, and X fields of each slip surface
* using slipWeighter to determine the wt field of each slip surface
* using rand to generate a new pool of NUM_SLIPS slip surfaces by applying crossovers and mutations to the previous generation, with the more highly-weighted members having a greater likelihood of contributing to the subsequent generations
* applying kinAdm, slicer, morgPrice, and slipWeighter to the new generation
* repeating until at least MIN_GENS have occured and the relative difference between subsequent generations is less than REL_DIFF.

10.4.5 Local Functions

generate_slips() :

$$generate\_slips(Xrange) = (Xrange[0] + rand() * (Xrange[1] - Xrange[0]))$$

11 MIS of the Kinematic Admissibility Module

11.1 Module

KinAdm

11.2 Uses

Input (Section 8), Sequence (Section 16)

11.3 Syntax

11.3.1 Exported Constants

N/A

11.3.2 Exported Data Types

N/A

11.3.3 Exported Access Programs

Name	In	Out	Exceptions
kinAdm	slip	$\mathbb{B}$	-

11.4 Semantics

11.4.1 State Variables

N/A

11.4.2 Environment Variables

N/A

11.4.3 Assumptions

- The *surf* field is populated for every member of the input sequence of slip data.

11.4.4 Access Routine Semantics

kinAdm(*slip_surf*):

- output:

$$
\begin{aligned}
&out := (\ \neg(\forall i \in [0..|slip\_surf.surf|-2])(slip\_surf.surf[i].x - slip\_surf.surf[i+1].x \leq 0) \\
&\vee \neg is\_on\_slope(slip\_surf.surf[0]) \\
&\vee \neg is\_on\_slope(slip\_surf.surf[|slip\_surf.surf - 1|]) \\
&\vee \neg is\_in\_slope(slip\_surf.surf) \\
&\vee (cncvu() \wedge is\_concave\_up(slip\_surf.surf)) \\
&\vee (obtu() \wedge has\_no\_sharp\_angles(slip\_surf.surf)) \\
&\Rightarrow false \\
&|else \Rightarrow true)
\end{aligned}
$$

[Not sure if I'm allowed to use "else" here but don't know how else to express the "else" case succintly —BM]

11.4.5 Local Functions

linSlope : coord × coord → $\mathbb{R}$
linSlope(*point1*, *point2*) = $\frac{point2.y - point1.y}{point2.x - point1.x}$

is_on_slope : coord → $\mathbb{B}$
is_on_slope(*point*) = $(\exists i \in [0..|slope.strat| - 1])(point = slope.strat[i])$
$\vee\ (\exists i \in [0..|slope.strat| - 2])(point.y = linSlope(slope.strat[i], slope.strat[i+1]) * point.x + \frac{slope.strat[i].y}{linSlope(slope.strat[i], slope.strat[i+1]) * slope.strat[i].x})$

is_in_slope : coords → $\mathbb{B}$ is_in_slope(*surf*) = $(\forall i \in [1..|surf| - 2])((\forall j \in [0..|slope.strat| - 2]$ with $slope.strat[j].x \leq surf[i].x < slope.strat[j+1].x)(surf[i].y < (slope.strat[j].y + (surf[i].x - slope.strat[j].x) * linSlope(slope.strat[j], slope.strat[j+1]))))$

is_concave_up : coords → $\mathbb{B}$
is_concave_up(*surf*) = $(\forall i \in [0..|surf|-3])(linSlope(surf[i+1], surf[i+2]) \geq linSlope(surf[i], surf[i+1]))$

distance : coord × coord → $\mathbb{R}$
distance(*point1*, *point2*) = $\sqrt{(point1.x - point2.x)^2 + (point1.y - point2.y)^2}$

has_no_sharp_angles : coords → $\mathbb{B}$
has_no_sharp_angles(*surf*) = $(\forall i \in [0..|surf| - 3])$
$(\arccos \frac{(distance(surf[i], surf[i+1]))^2 + (distance(surf[i+1], surf[i+2]))^2 - (distance(surf[i], surf[i+2]))^2}{2 * distance(surf[i], surf[i+1]) * distance(surf[i+1], surf[i+2])})$

12 MIS of the Slip Weighting Module

12.1 Module

SlipWeighter

12.2 Uses

Sequence (Section 16)

12.3 Syntax

12.3.1 Exported Constants

N/A

12.3.2 Exported Data Types

N/A

12.3.3 Exported Access Programs

Name	In	Out	Exceptions
slipWeighter	slips	slips	-

12.4 Semantics

12.4.1 State Variables

N/A

12.4.2 Environment Variables

N/A

12.4.3 Assumptions

- The *Fs* field is populated for every member of the input sequence of slip data.

12.4.4 Access Routine Semantics

slipWeighter(*slip_surfs*):

- output:

  out := *slip_surfs* sorted by their *Fs* field in ascending order and with the *wt* field of each member populated.

12.4.5 Local Functions

N/A

13 MIS of the Slip Slicer Module

13.1 Module

Slicer

13.2 Uses

Sequence (Section 16)

13.3 Syntax

13.3.1 Exported Constants

N/A

13.3.2 Exported Data Types

N/A

13.3.3 Exported Access Programs

Name	In	Out	Exceptions
slicer	coords, $\mathbb{Z}$	slip	-

13.4 Semantics

13.4.1 State Variables

N/A

13.4.2 Environment Variables

N/A

13.4.3 Assumption

N/A

13.4.4 Access Routine Semantics

slicer(*slip_surf*, *num_slices*):

- output:

  $out := (evnslc \Rightarrow slip\_surf$' obtained by repeatedly applying $slip\_surf$[large_segment($slip\_surf$)] || midpoint($slip\_surf$[large_segment($slip\_surf$)], $slip\_surf$[large_segment($slip\_surf$)+1]) || $slip\_surf$[large_segment($slip\_surf$)+1] until $|slip\_surf'| = num\_slices$ $|\neg evnslc \Rightarrow slip\_surf'$ such that $\forall(i : \mathbb{Z} | i \in [0..|slip\_surf| - 2] : slip\_surf'[i * \frac{num\_slices}{|slip\_surf|-1}..(i+1) * \frac{num\_slices}{|slip\_surf|-1}] = subslice(\frac{num\_slices}{|slip\_surf|-1}, slip\_surf[i], slip\_surf[i+1]))$

13.4.5 Local Functions

large_segment : coords $\rightarrow \mathbb{Z}$
large_segment($surf$) = $index$ such that
$\forall(i : \mathbb{Z} | i \in [0..|surf| - 2] : surf[index + 1] - surf[index] \geq surf[i + 1] - surf[i])$

midpoint : coord $\times$ coord $\rightarrow$ coord
midpoint($point1$, $point2$) = $< \frac{point1.x + point2.x}{2}, \frac{point1.y + point2.y}{2} >$
subslice : int $\times$ coord $\times$ coord $\rightarrow$ coords subslice(n, $point1$, $point2$) = $subslices$ such that
$\forall(i : \mathbb{Z} | i \in [0..n] : subslices[i].x = point1.x + \frac{i}{n} * (point2.x - point1.x) \wedge subslices[i].y = point1.y + \frac{i}{n} * (point2.y - point1.y))$

14 MIS of the Morgenstern Price Solver Module

14.1 Module

MorgPriceSolver

14.2 Uses

Input (Section 8), PropertySorter (Section 15), Sequence (Section 16)

14.3 Syntax

14.3.1 Exported Constants

N/A

14.3.2 Exported Data Types

N/A

14.3.3 Exported Access Programs

Name	In	Out	Exceptions
Morgenstern Price Solver	Sequence; struc; struc; struc; struc	Real	Non Converging; Spurious F_MP

14.4 Semantics

14.4.1 Local Constants

F_MinLim : $\mathbb{R}$	The minimum factor of safety value that the solution must be above to not be considered spurious. [*F_MinLim*=0.5]
max_iter : $\mathbb{R}$	The max number of iterations the algorithm will perform before the solution is considered non converging. [*max_iter*=20]
eps_F : $\mathbb{R}$	The value the absolute difference between the factor of safety calculated by the algorithm between consecutive iterations must be below for the answer to be considered converged. [*eps_F*=1E-6]
eps_Lam : $\mathbb{R}$	The value the absolute difference between the interslice normal to shear force ratio calculated by the algorithm between consecutive iterations must be below for the answer to be considered converged. [*eps_Lam*=1E-6]

14.4.2 State Variables

Lam : $\mathbb{R}$	The interslice normal to shear force ratio. From IM2 of the SRS.
E_force : $[\mathbb{R}]^{1,n+1}$	Sequence of the value of the interslice normal force exerted between slices. A value for each interslice, including ends. Sequence length value n is defined by the input *evalslip*. From IM3 of the SRS.
Del_F : $\mathbb{R}$	The difference between the factor of safety of the current iteration and the previous iteration. When converged the value will not be changing and *Del_F* will be small.
Del_Lam : $\mathbb{R}$	The difference between the interslice normal to shear force ratio of the current iteration and the previous iteration. When converged the value will not be changing and *Del_Lam* will be small.

14.4.3 Access Routine Semantics

Input:

evalslip : $[\mathbb{R}]^{2,n+1}$	Vertex coordinates for the slip surface being evaluated. Identifies shape of the slope, and slice points. Sequence length value of n is defined by the Slicer module (section 13).
params_layers : struc_layers	(Appendix 19.1.1)
piez : struc_piez	(Appendix 19.1.2)
soln : struc_soln	(Appendix 19.1.4)

params_load : struc_load (Appendix **??**)

Exceptions:
A solution which does not converge to a consistent solution, where the change in calculated factor of safety (*Del_F*) between iterations is less than *eps_F*, and the change in interslice normal to shear force ratio (*Del_Lam*) is less than *eps_Lam*, in less than *max_iter* iterations will be considered non converging exception case. A solution with a final calculated a factor of safety less than *F_MinLim* will be considered a spurious factor of safety exception case. Solutions that trigger these exception cases will output a factor of safety (*F_MP*) of 1000.

Output:

F_MP : $\mathbb{R}$	The factor of safety of the slope, as calculated by the Morgenstern Price solution method, measuring the stability of the slope. From IM1 of the SRS.

15 MIS of the Property Sorter Module

15.1 Module

PropertySorter

15.2 Uses

Input (Section 8), Sequence (Section 16)

15.3 Syntax

15.3.1 Exported Constants

N/A

15.3.2 Exported Data Types

N/A

15.3.3 Exported Access Programs

Name	In	Out	Exceptions
Property Sorter	Sequence; struc; struc	struc; struc; struc; struc	None

15.4 Semantics

15.4.1 Access Routine Semantics

Input:

evalslip : $[\mathbb{R}]^{1,n}$	Sequence of vertex coordinates for the slip surface being evaluated. Identifies shape of the slope, and slice points. Sequence length value n is defined by the Slicer module (section 13).
params_layers : struc_layers	(Appendix 19.1.1)
piez : struc_piez	(Appendix 19.1.2)

Exceptions:
There are no potential exceptions for Property Sorter.

Output:

params_internalForce : struc_intForce	(Appendix 19.1.5)
params_angles : struc_angles	(Appendix 19.1.6)
params_soilInterior : struc_soilInt	(Appendix 19.1.7)
params_soilBase : struc_soilBase	(Appendix 19.1.8)

16 MIS of the Sequence Data Structure Module

16.1 Module

Sequence

16.2 Uses

N/A

16.3 Syntax

16.3.1 Exported Constants

N/A

16.3.2 Exported Data Types

[T] = sequence of T, where T is any type

16.3.3 Exported Access Programs

Name	In	Out	Exceptions
[_]	Any number of values of type T	[T]	-
_(_)	[T], $\mathbb{Z}$	T	
_(_:_)	[T], $\mathbb{Z}$, $\mathbb{Z}$	[T]	-

16.4 Semantics

16.4.1 State Variables

N/A

16.4.2 Environment Variables

N/A

16.4.3 Assumptions

N/A

16.4.4 Access Routine Semantics

[_](Any number of values):

- output:

  $out :=$ A sequence containing the arguments passed to the function.

_(_)(*list*, *int*):

- output:

$$out := list[int]$$

_(_:_)(*list*, *int1*, *int2*):

- output:

$$out := list[int1..int2]$$

16.4.5 Local Functions

N/A

16.4.6 Considerations

This module is the sequence data type and operations on sequences implemented by Matlab.

17 MIS of the Plotting Module

17.1 Module

Plot

17.2 Uses

N/A

17.3 Syntax

17.3.1 Exported Constants

N/A

17.3.2 Exported Data Types

N/A

17.3.3 Exported Access Programs

Name	In	Out	Exceptions
plot	[$\mathbb{R}$], [$\mathbb{R}$]	-	-

17.4 Semantics

17.4.1 State Variables

N/A

17.4.2 Environment Variables

screen : $[\mathbb{Z}]$

- *screen* represents the colour values for each pixel on the screen of the hardware running SSP.

17.4.3 Assumptions

N/A

17.4.4 Access Routine Semantics

plot(x, y):

- transition:

  Modifies *screen* to display a plot with x on the horizontal axis and y on the vertical axis.

17.4.5 Local Functions

N/A

17.4.6 Considerations

This module is the plot function implemented by Matlab.

18 MIS of the Random Number Generation Module

18.1 Module

Rand

18.2 Uses

N/A

18.3 Syntax

18.3.1 Exported Constants

N/A

18.3.2 Exported Data Types

N/A

18.3.3 Exported Access Programs

Name	In	Out	Exceptions
rand	-	$\mathbb{R}$	-

18.4 Semantics

18.4.1 State Variables

N/A

18.4.2 Environment Variables

N/A

18.4.3 Assumptions

N/A

18.4.4 Access Routine Semantics

rand():

- output:

  out coloneqq A random number in the interval (0,1).

18.4.5 Local Functions

N/A

18.4.6 Considerations

This module is the rand function implemented by Matlab.

References

Carlo Ghezzi, Mehdi Jazayeri, and Dino Mandrioli. *Fundamentals of Software Engineering*. Prentice Hall, Upper Saddle River, NJ, USA, 2nd edition, 2003.

Daniel M. Hoffman and Paul A. Strooper. *Software Design, Automated Testing, and Maintenance: A Practical Approach*. International Thomson Computer Press, New York, NY, USA, 1995. URL http://citeseer.ist.psu.edu/428727.html.

19 Appendix

19.1 Parameter Tables

19.1.1 Layer Parameters

The elements in the structure of the containers for the parameters of different slope layers. Assumed that the parameters will be entered such that sequence progresses from the uppermost stratigraphic layer at the first index, to the lowest stratigraphic layer at the last index. *nlayer* refers to the number of soil layers in the slope, and is defined by the input file.

Parameter	Description
$strat : [[\mathbb{R}]^{2,\text{nvtx}}]^{1,\text{nlayer}}$	Sequence of coordinate sequences describing the vertexes of each layer. The value *nvtx* is defined by the input file, and can be different for each sequence.
$phi : [\mathbb{R}]^{1,\text{nlayer}}$	Sequence of the effective angle of friction for each stratigraphic layer.
$coh : [\mathbb{R}]^{1,\text{nlayer}}$	Sequence of the effective cohesion for each stratigraphic layer.
$gam : [\mathbb{R}]^{1,\text{nlayer}}$	Sequence of the dry unit weight of soil for each stratigraphic layer.
$gams : [\mathbb{R}]^{1,\text{nlayer}}$	Sequence of the saturated unit weight of soil for each stratigraphic layer.
$E : [\mathbb{R}]^{1,\text{nlayer}}$	Sequence of the Young's modulus for each stratigraphic layer.
$nu : [\mathbb{R}]^{1,\text{nlayer}}$	Sequence of the poissons ratio for each stratigraphic layer.

19.1.2 Piezometric Parameter

The elements in the structure for parameters relating to the piezometric surface existing on the slope. *npz* refers to the number of vertexes describing the piezometric surface, and is defined by the input file.

Parameter	Description
$piez : [\mathbb{R}]^{2,\text{npz}}$	Sequence of vertex coordinates describing the geometry of the water table. If there is no water table than *piez* is an empty array.
$gamw : \mathbb{R}$	The unit weight of water.

19.1.3 Search Range Parameters

The elements in the structure for parameters relating to the range of coordinates the critical slip surface will be searched for in.

Parameter	Description
Xext : $[\mathbb{R}]^{1,2}$	Sequence of the range of x-ordinates that the exit point of the slip will be searched for in. Exit refers to the point of the slip at lower elevation that the slope mass will move towards during failure.
Xent : $[\mathbb{R}]^{1,2}$	Sequence of the range of x-ordinates that the entry point of the slip will be searched for in. Entry refers to the point of the slip at higher elevation that the slope mass will move away from during failure.
Ylim : $[\mathbb{R}]^{1,2}$	Sequence of range of y-ordinates that the slip will be searched for in. The larger value should be greater than the max y-ordinate of the slope. The smaller Ylim value is the deepest the slip surface is expected to descend to.

19.1.4 Solution Parameters

The elements in the structure for parameters relating to method in which the solution method will be approached.

Parameter	Description
ltor : $\mathbb{B}$	Direction the slope is expected to experience failure in. If true than the side of the slope with a greater x-ordinate value is at a lower elevation. If false than the side of the slope with a greater x-ordinate is at a higher elevation.
ftype : $\mathbb{B}$	Switch between functions to use for interslice shear/normal inclination function. If true then the inclination function is a constant (Spencer's method). If false then the inclination function is a half-sine (standard Morgenstern Price method).
evnslc : $\mathbb{B}$	Switch between method of slicing a slip surface to when preparing for analysis. If true then slice slip surface into equal x-ordinate widths. If false then slice distance between vertices into even number of slices.
cncvu : $\mathbb{B}$	Switch for concave slip surface admissibility criterion. If true then an admissible slip surface must be concave upwards towards the surface. If false then an admissible slip surface does not need to pass this criterion.

obtu : $\mathbb{B}$	Switch for angle limit slip surface admissibility criterion. If true then an admissible slip surface must have all interior angles greater than a set limit. If false then an admissible slip surface does not need to pass this criterion.

19.1.5 Internal Force Parameters

The elements in the structure for parameters relating to the forces acting on a slice caused by the slope, and water in the slope acting on itself. n refers to the number of slices composing the evaluation slip surface, and is defined by the Slicer module (section 13).

Parameter	Description
Ub : $[\mathbb{R}]^{1,n}$	Sequence of the force acting on the basal surface of a slice as a result of pore water pressure within the slice. Value for each slice. From DD2 of the SRS.
Ut : $[\mathbb{R}]^{1,n}$	Sequence of the force acting on the upper surface of a slice as a result of pore water pressure standing water on the surface. Value for each slice. From DD3 of the SRS.
W : $[\mathbb{R}]^{1,n}$	Sequence of the downward force acting on the slice caused by the mass of the slice and the force of gravity. Value for each slice. From DD1 of the SRS.
H : $[\mathbb{R}]^{1,n-1}$	Sequence of the force acting into the interslice surfaces as a result of pore water pressure within the adjacent slices. Value for each interslice. From DD4 of the SRS.

19.1.6 Angle Parameters

The elements in the structure for parameters relating to the angles of the slice surfaces. n refers to the number of slices composing the slip surface, and is defined by the input *evalslip* given to the Property Sorter module (section 15).

Parameter	Description
Alpha : $[\mathbb{R}]^{1,n}$	Sequence of the angle that the basal surface of the slice makes with the horizontal. Value for each slice. From DD**??** of the SRS.

Beta : $[\mathbb{R}]^{1,n}$	Sequence of the angle that the upper surface of the slice makes with the horizontal. Value for each slice. From DD**??** of the SRS.

19.1.7 Soil Interslice Properties

The elements in the structure for parameters relating to the soil properties of the slope, as calculated at the interslice interfaces of an evaluation slip. Calculation is based on the ratio of the interface that is in different stratigraphic layers, and the values of the effective angle of friction in the different layers. Interest is only with the interior interslice interfaces therefore for a slope of n slices, there will be $n-1$ interior interslice interfaces. The value n is defined by the input *evalslip* given to the Property Sorter module (section 15).

Parameter	**Description**
phi_IS : $[\mathbb{R}]^{1,n-1}$	Sequence of the vector of the effective angle of friction calculated at each interslice interface.
coh_IS := $[\mathbb{R}]^{1,n-1}$	Sequence of the vector of the effective cohesion calculated at each interslice interface.
E_IS := $[\mathbb{R}]^{1,n-1}$	Sequence of the vector of the Youngs modulus calculated at each interslice interface.
nu_IS := $[\mathbb{R}]^{1,n-1}$	Sequence of the vector of the Poisson ratio calculated at each interslice interface.

19.1.8 Soil Base Properties

The elements in the structure for parameters relating to the soil properties of the slope, as calculated at the basal surfaces of an evaluation slip. Calculation is based on the ratio of the basal surface that is in different stratigraphic layers, and the values of the effective angle of friction in the different layers. An evaluation slip of n slices will have n basal surfaces, and the value of n is defined by the input *evalslip* given to the Property Sorter module (section 15).

Parameter	**Description**
phi_Base : $[\mathbb{R}]^{1,n}$	Sequence of the vector of the effective angle of friction calculated at each slice basal surface in an evaluation slip.

$coh\_Base : [\mathbb{R}]^{1,n}$	Sequence of the vector of the effective cohesion calculated at each slice basal surface in an evaluation slip.
$E\_Base : [\mathbb{R}]^{1,n}$	Sequence of the vector of the Young's modulus calculated at each slice basal surface in an evaluation slip.
$nu\_Base : [\mathbb{R}]^{1,n}$	Sequence of the vector of the Poisson ratio calculated at each slice basal surface in an evaluation slip.