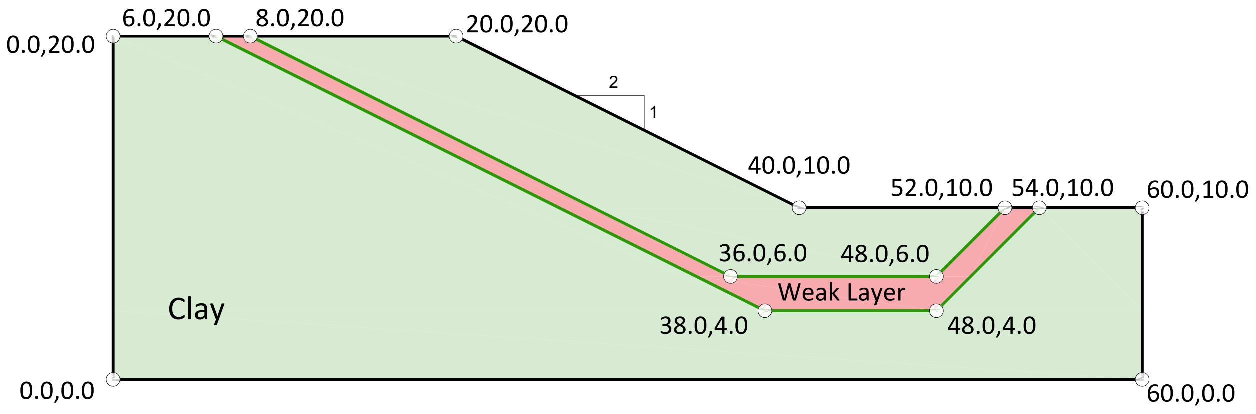
Parametric Study in ADONIS using Python

Created By: Roozbeh Geraili Mikola, PhD, PE
Email: adonis4geo@outlook.com
Web Page: www.geowizard.org



This tutorial will demonstrate how to use ***Python*** in ***ADONIS***. Python is a general-purpose programming language with good support for scientific and numerical programming. The Python programming language is embedded inside ***ADONIS*** and extended to allow models to be manipulated from Python programs. Currently Python environment contains the following extension modules:

- numpy 1.22.4 Array operations
- scipy 1.8.1 Collection of scientific libraries
- XlsxWriter 3.0.3 Writing files in the Excel 2007+ XLSX file format

An undrained clay slope failure with a thin weak layer

This example demonstrates a stability analysis of a slope of undrained clay. The example is taken from Griffiths and Lane's paper (1999). The slope model consists of a thin layer of week material. The weak layer runs parallel to the slope and then turns to be horizontal in the toe zone. The presence of this thin weak layer in the slope influences the stability of the slope. The geometry of the slope model is presented above. The height of the slope is 10 m and the slope is inclined at an angle of 26.57° (2:1) to the horizontal. Table 1 presents the material properties for the slope model. In this example, the ***Python*** scripting is used to carry out the parametric analyses using a constant value of undrained shear strength of the soil (c_{u1}) and five different values of undrained shear strength of the thin layer (c_{u2}) with ratios c_{u2}/c_{u1} equal to 1.0, 0.8, 0.6, 0.4, and 0.2. Figure 2 gives a discretization of the domain unstructured triangular (T3) elements. The right and the left sides of the slope were horizontally constrained, and the underside was completely fixed.

Table 1- Soil Parameters for undrained clay slope with a thin weak layer.

Parameters	**Value**
Unit Weight, *γ* (kN/m^3)	20
Young's Modulus, *E* (kPa)	100000
Poisson's Ratio, *v*	0.3
Cohesion (undrained), c_{u1} (kPa)	50
Friction angle (undrained), (°)	0

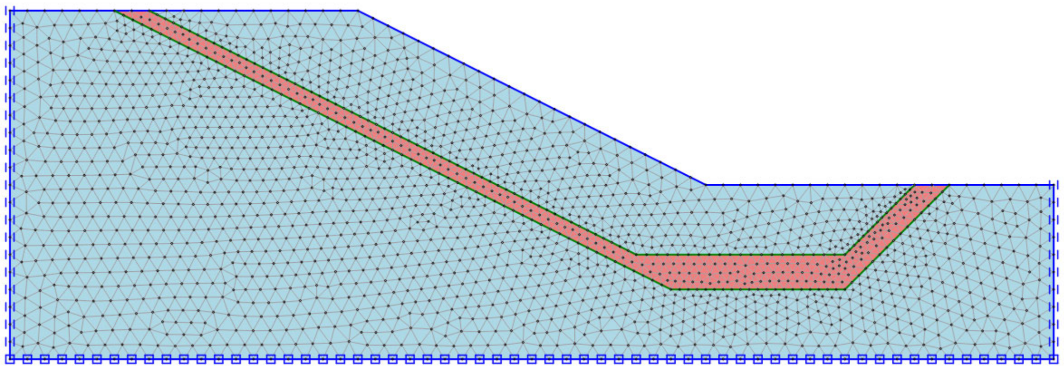


Figure 1- Typical finite element mesh for undrained clay slope with a thin weak layer.

Writing Python Code

There are two ways to write Python code in ***ADONIS***. In the Python console or load the script text. In the console you write Python commands one by one, executing them by pressing ***Enter***, while script can contain more complex code made up of several lines, executed only when loaded. In this tutorial we use the latter. You can simply create a new file using any text editor (Notepad++ for instance) and type all the python commands in there and save the file with *py* extension (i.e. script_tut10.py). Now you will be able to load the script into ***ADONIS*** which gets executed by embedded Python interpreter. Please note that ***ADONIS*** only recognizes the script files with extension ****.py*** and ****.ajs*** which corresponds to ***Python*** and ***JavaScript*** script files.

Select: *File* → *Load Script*

Working with Modules

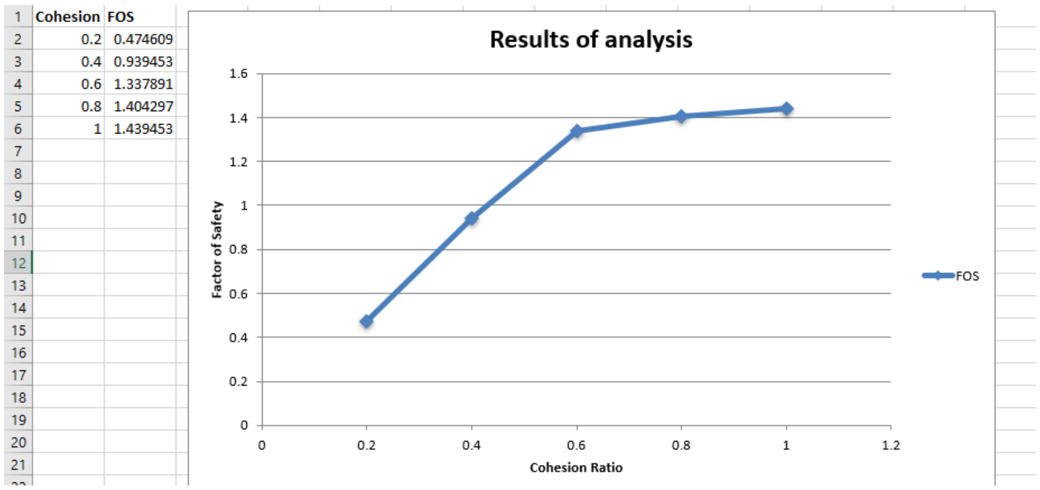
First, we import the *adonis* module with the import statement. The *adonis* module defines the interaction between ***Python*** and ***ADONIS***. You also will be able to import the *adonis* module as the shortened name like *ad*. The following script creates a simple slope geometry using Python scripting. The *ad.command* (*adonis.command*) function is used to issue a series of ***ADONIS*** commands. The triple quotes are used to define a multi-line string. The objective is to modify the property for the weak layer and perform the sensitivity analysis to estimate the factor of safety for multiple cohesion parameters. The string formatting is used to modify the cohesion value in each cycle. The following lines show how to modify the properties for the clay layer and perform the computation after each modification. *For* statement is used to repeat the process. Please note that '*dispinfodlg'* argument should be turned *'off'* with *solve* command to prevent program from showing the information dialog at the end of each computation cycle. Otherwise, program would stop the process after first loop and steps out of the for-loop.

```
import adonis as ad

cu1 = 50000
fos_array = [-1,-1,-1,-1,-1]
coh_ratio_array = [0.2,0.4,0.6,0.8,1.0]

cmd = "material('create','Mohr-Coulomb','matid',2,'matname','Material
2','density',2038,'shear',3.84615e+07,'bulk',8.33333e+07,'coh',{coh},'fric',0,'dil',0,'tens
',0)"

for i in range(0, 5):
    ad.command("""
    newmodel()
    set('unit','stress-pa')
    line('startPoint',0,0,'endPoint',0,20)
    line('startPoint',0,20,'endPoint',20,20)
    line('startPoint',20,20,'endPoint',40,10)
    line('startPoint',40,10,'endPoint',60,10)
    line('startPoint',60,10,'endPoint',60,0)
    line('startPoint',60,0,'endPoint',0,0)
    line('startPoint',6,20,'endPoint',38,4)
    line('startPoint',38,4,'endPoint',48,4)
    line('startPoint',48,4,'endPoint',54,10)
    line('startPoint',8,20,'endPoint',36,6)
    line('startPoint',36,6,'endPoint',48,6)
    line('startPoint',48,6,'endPoint',52,10)
    discretize('maxedge',1)
    segment('id',14,11,'numedge',60)
    segment('id',15,12,'numedge',20)
    segment('id',16,13,'numedge',15)
    gmsh('maxedge',1,'elemtype','T3','useNMD','on')
    material('create','Mohr-Coulomb','matid',1,'matname','Material
1','density',2038,'shear',3.84615e+07,'bulk',8.33333e+07,'coh',50000,'fric',0,'dil',0,'tens
',0)
    """)

    new_coh = coh_ratio_array[i] * cu1
    ad.command(cmd.format(coh=new_coh))

    ad.command("""
    material('assign','matid',1,'region',42,8)
    material('assign','matid',2,'region',42,5)
    material('assign','matid',1,'region',42,2)
    applybc('xfix','xlim',59.743,61.035,'ylim',-1.499,10.698)
    applybc('xfix','xlim',-1.654,0.103,'ylim',-1.499,20.621)
    applybc('xyfix','xlim',-1.602,61.035,'ylim',-1.240,0.103)
    set('gravity',0,9.8)
    solve('fos','fosLBLimit',0.25,'fosUBLimit',2.0,'dispinfodlg','off')
    """)

    fos_array[i] = ad.fos()
```

ad.fos command is used to extract the minimum factor of safety for the model. List of all the functions in the *adonis* module is listed in the manual (Help -> Scripting Language -> Python)

Now you will be able to export the result into excel file using *XlsxWriter* module. XlsxWriter is a Python module that can be used to write text, numbers, formulas and hyperlinks to multiple worksheets in an Excel 2007+ XLSX file. XlsxWriter module is already install in the program's Python environment. More information about the XlsxWriter module can be found at https://xlsxwriter.readthedocs.io/. The following commands shows how to export the output from the program into the excel file. Figure 3 illustrates the output excel file created by XlsxWriter module.

```python
import xlsxwriter

# Create a new XlsxWriter Workbook object
workbook = xlsxwriter.Workbook('coh_ratio_vs_fos_plot.xlsx')
# Add a new worksheet to a workbook
worksheet = workbook.add_worksheet()
# Create a new Format object to formats cells in worksheets
bold = workbook.add_format({'bold': 1})
# Add the worksheet data that the charts will refer to.
headings = ['Cohesion Ratio', 'FOS']
# Write a row of data
worksheet.write_row('A1', headings, bold)
# Write a column of data
worksheet.write_column('A2', coh_rati_array)
worksheet.write_column('B2', fos_array)
# Create a scatter chart sub-type with straight lines and markers.
chart = workbook.add_chart({'type': 'scatter',
                             'subtype': 'straight_with_markers'})
# Configure the first series.
chart.add_series({
    'name': '=Sheet1!$B$1',
    'categories': '=Sheet1!$A$2:$A$7',
    'values': '=Sheet1!$B$2:$B$7',
})
# Add a chart title and some axis labels.
chart.set_title({'name': 'Results of analysis'})
chart.set_x_axis({'name': 'Cohesion Ratio', 'min': 0, 'max': 1.2})
chart.set_y_axis({'name': 'Factor of Safety'})
# Set an Excel chart style.
chart.set_style(11)
# Insert the chart into the worksheet (with an offset).
worksheet.insert_chart('D2', chart, {'x_offset': 25, 'y_offset': 10, 'x_scale': 1.5,
'y_scale': 1.5})
# Close the Workbook object and write the XLSX file
workbook.close()
```



	Cohesion	FOS
1	Cohesion	FOS
2	0.2	0.474609
3	0.4	0.939453
4	0.6	1.337891
5	0.8	1.404297
6	1	1.439453

Figure 3- Exported excel file that summarizes cohesion ratio versus fos values.

The FOS obtained using various methods with different ratios of c_{u2}/c_{u1} are illustrated in Figure 4.

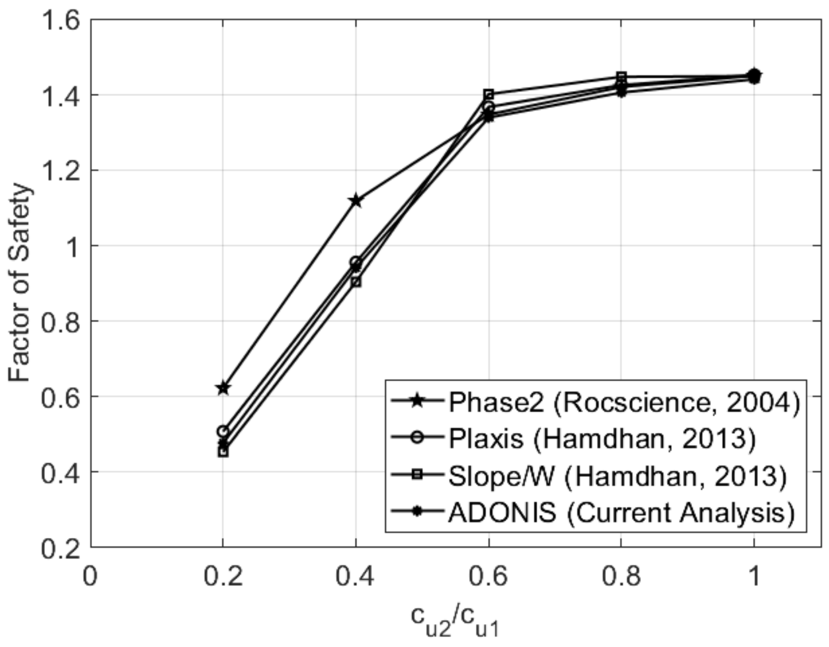


Figure 4- Comparison of FOS for undrained clay slope with a thin weak layer for different c_{u2}/c_{u1} ratio.

Script

The Python source codes for this example are listed below (script_tut10.py).

```python
# Name: script_tut10.py
# Import adonis and xlsxwriter modules
import adonis as ad
import xlsxwriter

# Initialize the fos and cohesion arrays
cu1 = 50000
fos_array = [-1,-1,-1,-1,-1]
coh_ratio_array = [0.2,0.4,0.6,0.8,1.0]

# Initialize the command text with the placeholder using curly brackets {}
cmd = "material('create','Mohr-Coulomb','matid',2,'matname','Material 
2','density',2038,'shear',3.84615e+07,'bulk',8.33333e+07,'coh',{coh},'fric',0,'dil',0,'tens',0)"
# Loop through the cohesion values and perform the computation and store the fos value
for i in range(0, 5):
    ad.command("""
    newmodel()
    set('unit','stress-pa')
    line('startPoint',0,0,'endPoint',0,20)
    line('startPoint',0,20,'endPoint',20,20)
    line('startPoint',20,20,'endPoint',40,10)
    line('startPoint',40,10,'endPoint',60,10)
    line('startPoint',60,10,'endPoint',60,0)
    line('startPoint',60,0,'endPoint',0,0)
    line('startPoint',6,20,'endPoint',38,4)
    line('startPoint',38,4,'endPoint',48,4)
    line('startPoint',48,4,'endPoint',54,10)
    line('startPoint',8,20,'endPoint',36,6)
    line('startPoint',36,6,'endPoint',48,6)
    line('startPoint',48,6,'endPoint',52,10)
    discretize('maxedge',1)
    segment('id',14,11,'numedge',60)
    segment('id',15,12,'numedge',20)
    segment('id',16,13,'numedge',15)
    gmsh('maxedge',1,'elemtype','T3','useNMD','on')
    material('create','Mohr-Coulomb','matid',1,'matname','Material 
1','density',2038,'shear',3.84615e+07,'bulk',8.33333e+07,'coh',50000,'fric',0,'dil',0,'tens',0)
    """)

    new_coh = coh_ratio_array[i] * cu1
    ad.command(cmd.format(coh=new_coh))

    ad.command("""
    material('assign','matid',1,'region',42,8)
    material('assign','matid',2,'region',42,5)
    material('assign','matid',1,'region',42,2)

    applybc('xfix','xlim',59.743,61.035,'ylim',-1.499,10.698)
    applybc('xfix','xlim',-1.654,0.103,'ylim',-1.499,20.621)
    applybc('xyfix','xlim',-1.602,61.035,'ylim',-1.240,0.103)

    set('gravity',0,9.8)

    solve('fos','fosLBLimit',0.25,'fosUBLimit',2.0,'dispinfodlg','off')
    """)

    fos_array[i] = ad.fos()

# Create a new XlsxWriter Workbook object
workbook = xlsxwriter.Workbook('coh_ratio_vs_fos_plot.xlsx')
# Add a new worksheet to a workbook
worksheet = workbook.add_worksheet()
# Create a new Format object to formats cells in worksheets
bold = workbook.add_format({'bold': 1})
# Add the worksheet data that the charts will refer to.
headings = ['Cohesion Ratio', 'FOS']
# Write a row of data
worksheet.write_row('A1', headings, bold)
# Write a column of data
worksheet.write_column('A2', coh_ratio_array)
worksheet.write_column('B2', fos_array)
# Create a scatter chart sub-type with straight lines and markers.
chart = workbook.add_chart({'type': 'scatter',
                             'subtype': 'straight_with_markers'})
# Configure the first series.
chart.add_series({
    'name': '=Sheet1!$B$1',
    'categories': '=Sheet1!$A$2:$A$7',
    'values': '=Sheet1!$B$2:$B$7',
})
# Add a chart title and some axis labels.
chart.set_title({'name': 'Results of analysis'})
chart.set_x_axis({'name': 'Cohesion Ratio', 'min': 0, 'max': 1.2})
chart.set_y_axis({'name': 'Factor of Safety'})
# Set an Excel chart style.
chart.set_style(11)
# Insert the chart into the worksheet (with an offset).
worksheet.insert_chart('D2', chart, {'x_offset': 25, 'y_offset': 10, 'x_scale': 1.5, 'y_scale': 
1.5})
# Close the Workbook object and write the XLSX file
workbook.close()
```