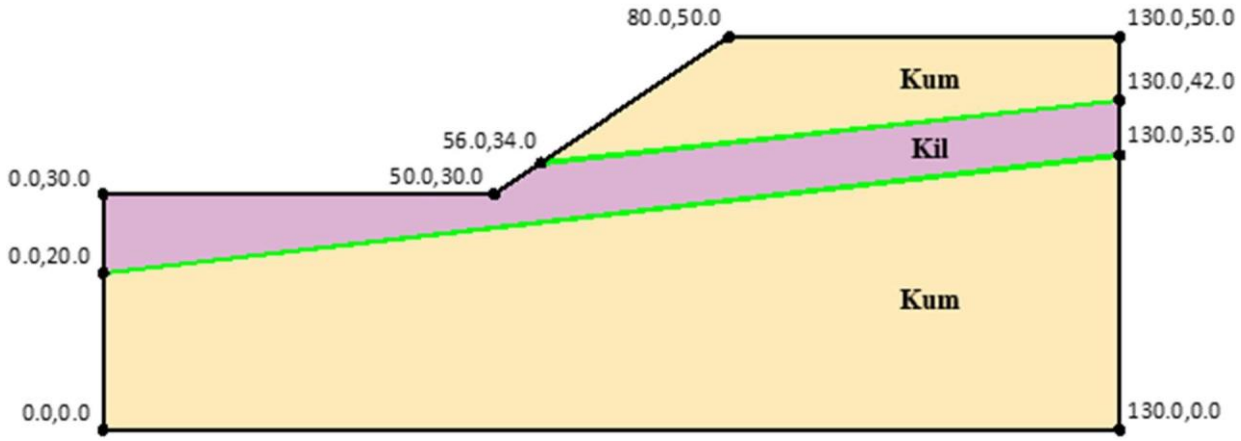
HYRCAN'da Python Kullanımı

Hazırlayan: **Roozbeh Geraili Mikola, PhD, PE**
Çevirmen: **Umut Dağar**
E-posta: **hyrcan4geo@outlook.com**
Web Sayfası: **www.geowizard.org**



Şekil 1. Model Geometrisi

Bu eğitim kılavuzu, **Python**'un **HYRCAN**'da nasıl kullanılacağını göstermektedir. Python, bilimsel ve sayısal programlama için kullanılan genel amaçlı bir programlama dilidir. Python programlama dili, **HYRCAN**'ın içine yerleştirilmiştir ve modellerin Python programı tarafından çalıştırılmasına izin verecek şekilde uygulanmıştır. Şu anda Python aşağıdaki uzantı modüllerini içemektedir:

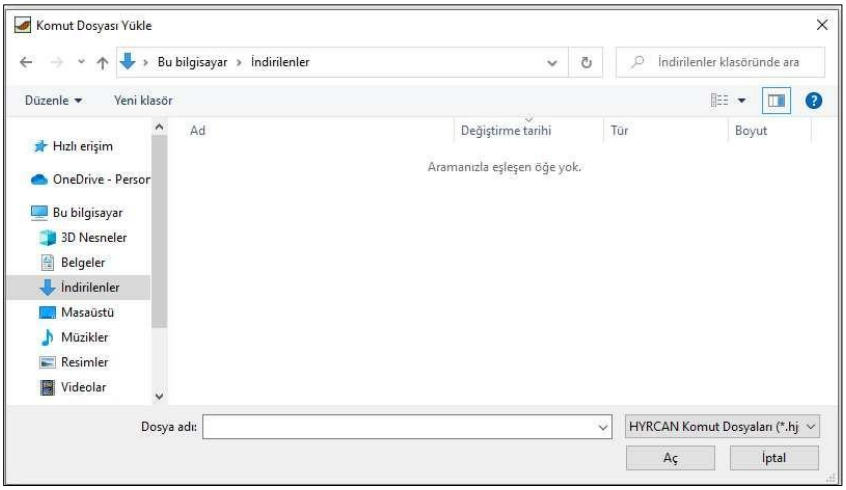
- numpy 1.22.4 Array operations
- scipy 1.8.1 Collection of scientific libraries
- XlsxWriter 3.0.3 Writing files in the Excel 2007+ XLSX file format

Python Kodu Yazma

Python konsolunda yazmak veya komut dosyası metnini yüklemek üzere **HYRCAN**'da Python kodu yazmanın iki yolu vardır. Konsolda Python komutlarını birer birer yazıp, Enter'a basarak çalıştırabilirsiniz, komut dosyası ise yalnızca yüklendiğinde çalışanve birkaç satırdan oluşan biraz daha karmaşık kodlar içermektedir. Bu eğitim kılavuzunda ikinci yöntemi kullanacağız. Herhangi bir metin düzenleyiciyi (örneğin; Notepad++) kullanarak yeni bir dosya oluşturabilir ve tüm python komutlarını buraya yazıp dosyayı py uzantılı (örneğin; script_tut09.py) olarak kaydedebilirsiniz. Python yorumlayıcısı tarafından çalıştırılan komut dizisini artık **HYRCAN**'a yükleyebileceksiniz. Lütfen **HYRCAN**'ın yalnızca **Python** ve **JavaScript** komut dosyalarına karşılık gelen ***.py** ve ***.hjs** uzantılı komut dosyalarını tanıdığını unutmayın.

Seçim: *Dosya* →

Komut Dosyası Yükle



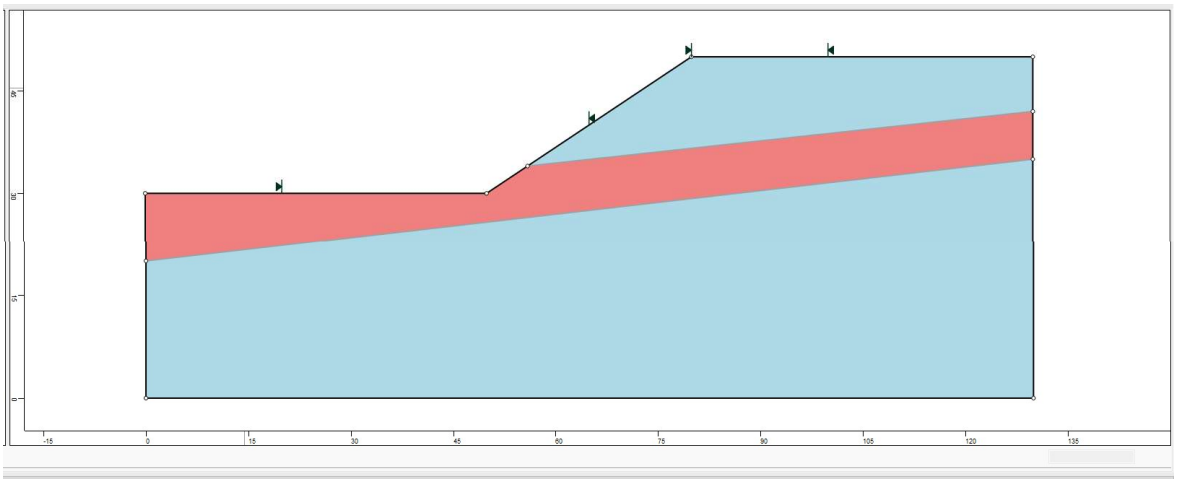
Şekil 2. Komut Dosyası Yükleme Penceresi.

Modüllerle Çalışmak

İlk olarak "import" deyimi ile hyrcan modülünü içe aktarıyoruz. Hyrcan modülü, **Python** ve **HYRCAN** arasındaki etkileşimi tanımlamaktadır. Ayrıca, ***hy*** kısaltması ile hyrcan modülünü içe aktarabilirsiniz.

```
import hyrcan as hy
hy.command("""
newmodel()
set('failureDir','r21')
extboundary(0,0,130,0,130,50,80,50,50,30,0,30,0,0)
matboundary(0,20,130,35)
matboundary(56,34,130,42)
definemat('ground','matID',1,'matName','Sand','uw',18,'cohesion',5,'friction',38)
definemat('ground','matID',2,'matName','Clay','uw',17,'cohesion',57,'friction',0)
assignsoilmat('matid',2,'atpoint',85,30)
definelimits('limit',20,65,'limit2',80,100)
set('Method','BishopSim','on')
""")
```

Yukarıdaki komut dosyası, Python komut dosyası kullanarak basit bir şev geometrisi oluşturur. hy.command (hyrcan.command) fonksiyonu, bir dizi **HYRCAN** komutu vermek için kullanılır. Üçlü tırnak işaretleri, çok satırlı bir dize tanımlamak için kullanılır. Şekil 1, Python komut dizisi tarafından oluşturulan şev geometrisinin şematik görünümünü göstermektedir.



Şekil 3. Dış Sınırın Oluşturulması.

Şimdi diyelim ki, kil tabakasının özelliğini değiştirmek ve kohezyon parametreleri için güvenlik faktörünü tahmin etmek amacıyla analiz yapmak istiyoruz. Aşağıdaki satırlar, kil tabakasının özelliklerinin nasıl değiştirileceğini ve her değişiklikten sonra hesaplamanın nasıl gerçekleştirileceğini göstermektedir. "for" deyimi işlemi tekrarlamak için kullanılır. Lütfen, programın her hesaplama döngüsünün sonunda bilgi iletişim kutusunu göstermesini önlemek için "compute" komutuyla beraber "silence" argümanının kullanıldığını unutmayın. Aksi takdirde, program ilk döngüden sonra süreci durdurur ve "for" döngüsünden çıkar.

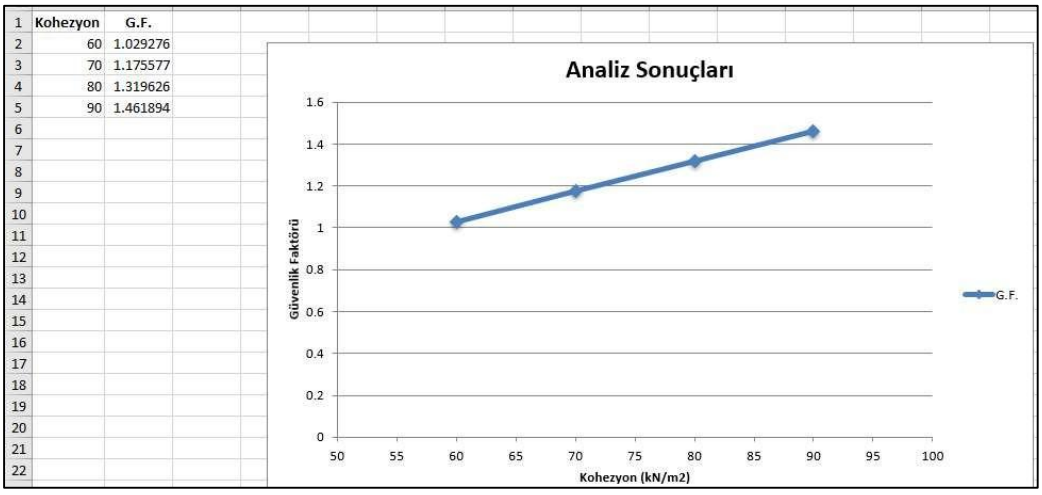
```
fos_array = [-1,-1,-1,-1]
coh_array = [60,70,80,90]
cmd =
"definemat('ground','matID',2,'matName','Clay','uw',17,'cohesion',{coh},'friction'
,0)"
for i in range(0, 4):
    hy.command(cmd.format(coh=coh_array[i]))
    hy.command("compute('silence')")
    fos_array[i] = hy.min_fos('BishopSim')
```

"hy.min_fos" komutu, belirtilen analiz yöntemi (yani Basitleştirilmiş Bishop) için minimum güvenlik faktörünü hesaplamak için kullanılır. Yöntemlerin tam adını yazarak kullanabilirsiniz. Örneğin; "Bishop Simplified", "Janbu Simplified", "Spencer", "GLE/Morgenstern-Price" olarak yazılabilir. Komut olarak yazmak isterseniz "BishopSim", "JanbuSim", "Spencer", "GLE/M-P" komutlarını kullanarak yazabilirsiniz. Hyrcan modülündeki tüm fonksiyonların listesi kılavuzda listelenmiştir. (Help -> Scripting Language -> Python)

Artık sonuçları "XlsxWriter" modülünü kullanarak excel dosyasına aktarabileceksiniz. "XlsxWriter", bir "Excel 2007+ XLSX" dosyasında birden çok çalışma sayfasına metin, sayı, formül ve köprü yazmak için kullanılabilen bir Python modülüdür. "XlsxWriter" modülü, programın Python ortamında zaten kuruludur. XlsxWriter modülü hakkında daha fazla bilgi https://xlsxwriter.readthedocs.io/ adresinde bulunabilir. Aşağıdaki komutlar, programdan çıktının excel dosyasına nasıl aktarılacağını göstermektedir. Şekil 4, "XlsxWriter" modülü tarafından oluşturulan excel dosyasını çıktısını göstermektedir..

```
import xlsxwriter

# Create a new XlsxWriter Workbook object
workbook = xlsxwriter.Workbook('coh_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 (kN/m2)', 'FOS']
# Write a row of data
worksheet.write_row('A1', headings, bold)
# Write a column of data
worksheet.write_column('A2', coh_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 (kN/m2)', 'min': 50, 'max': 100})
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()
```



Şekil 4. Kohezyon Değerlerine Karşılık Gelen Güvenlik Faktörlerini Özetleyen Excel Dosyası

Komut Dizisi

Bu örnek için kullanılan Python kaynak kodları aşağıda listelenmiştir. (script_tut09.py).

```
# Name: script_tut09.py
# Import hyrcan and xlsxwriter modules
import hyrcan as hy
import xlsxwriter

# Issue series of commands to create the initial slope geometry
hy.command("""
newmodel()
set('failureDir','r21')
extboundary(0,0,130,0,130,50,80,50,50,30,0,30,0,0)
matboundary(0,20,130,35)
matboundary(56,34,130,42)
definemat('ground','matID',1,'matName','Sand','uw',18,'cohesion',5,'friction',38)
definemat('ground','matID',2,'matName','Clay','uw',17,'cohesion',57,'friction',0)
assignsoilmat('matid',2,'atpoint',85,30)
definelimits('limit',20,65,'limit2',80,100)
set('Method','BishopSim','on')
""")

# Initialize the fos and cohesion arrays
fos_array = [-1,-1,-1,-1]
coh_array = [60,70,80,90]
# Initialize the command text with the placeholder using curly brackets {}
cmd = "definemat('ground','matID',2,'matName','Clay','uw',17,'cohesion',{coh},'friction',0)"
# Loop through the cohesion values and perform the computation and store the fos value
for i in range(0, 4):
    hy.command(cmd.format(coh=coh_array[i]))
    hy.command("compute('silence')")
    fos_array[i] = hy.min_fos('BishopSim')

# Create a new XlsxWriter Workbook object
workbook = xlsxwriter.Workbook('coh_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 (kN/m2)', 'FOS']
# Write a row of data
worksheet.write_row('A1', headings, bold)
# Write a column of data
worksheet.write_column('A2', coh_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 (kN/m2)', 'min': 50, 'max': 100})
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()
```