



Universidade Federal de Rondonópolis - Laboratório de Geoprocessamento ICAT UFR
Universidade Federal de Mato Grosso do Sul - Laboratório de Geoprocessamento para
Aplicações Ambientais - LabGIS FAENG UFMS

Guia de Uso – Visualização 3D de Terreno (DEM/MDS) em Python/Colab

1. Visão geral

Este notebook faz parte da PRAD Toolbox e foi desenvolvido para possibilitar a visualização em 3D de modelos de elevação (DEM/MDS) a partir de arquivos raster no formato .tif.

O notebook trata valores ausentes, permite aplicar uma suavização opcional e um ajuste da escala vertical. Em seguida gera uma superfície 3D com curvas de nível em formato PNG, para uso em relatórios, aulas e em Projetos de Recuperação de Áreas Degradadas - PRAD.

O fluxo está dividido em etapas numeradas (1) a (7), para que usuários iniciantes possam seguir passo a passo sem precisar conhecer programação.

2. Pré-requisitos

Antes de usar o notebook, você precisa:

- ter uma conta Google (para usar o Google Colab);
- ter um arquivo raster de elevação em formato GeoTIFF (.tif). Exemplos:
Modelo Digital de Elevação (MDE/DEM);

Modelo Digital de Superfície (MDS) derivado de imagens de drone, por exemplo

- ter o arquivo salvo no seu computador para fazer o upload.

O notebook foi pensado para rodar no Google Colab, mas pode ser adaptado para outros ambientes Python que tenham:

- i - rasterio;
- ii - numpy;
- iii - matplotlib;
- iv - scipy (para a suavização, opcional).

3. Estrutura do notebook

O notebook está organizado em blocos:

- instalar dependências (se necessário);
- importar bibliotecas;
- upload do arquivo raster (.tif);
- abrir o raster e tratar valores NoData;
- gerar eixos X e Y;
- processar recorte, suavização e escala vertical + plot 3D com curvas de nível;
- exportar figura 3D como PNG e baixar

Cada bloco possui comentários explicando o que está sendo feito, e as células principais têm o marcador # @title para facilitar a execução no Colab.

4. Passo a passo de uso

4.1. Abrir o notebook no Google Colab

Acesse o Google Colab em seu navegador.

Faça login com sua conta.

Abra o arquivo Visualizacao_DEM_MDS_v2.ipynb no Colab (upload direto ou a partir do Google Drive).

4.2. Etapa 1 – Instalar dependências

Bloco correspondente:

```
#1 Instalar dependências (se necessário)
!pip install rasterio numpy matplotlib
```

Execute esta célula apenas na primeira vez ou se seu ambiente ainda não possuir essas bibliotecas. No Colab, normalmente numpy e matplotlib já estão presentes. rasterio costuma precisar de instalação.

4.3. Etapa 2 – Importar bibliotecas

Bloco correspondente:

```
#2 Importar bibliotecas
import rasterio
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
from google.colab import files
import os
```

Esta célula apenas importa as bibliotecas usadas ao longo do notebook. Basta executar uma vez após a instalação das dependências.

4.4. Etapa 3 – Upload do arquivo raster (.tif)

Blocos correspondentes:

```
#3 Upload do arquivo raster (ex: STRM_rec.tif)
print("Faça o upload do arquivo raster (*.tif)...")
uploaded = files.upload()
```

```
# Detecta o nome do arquivo enviado
raster_path = list(uploaded.keys())[0]
print(f"Arquivo carregado: {raster_path}")
```

O que fazer:

- execute a célula de upload. O Colab abrirá uma janela para selecionar um arquivo no seu computador;
- escolha o raster .tif (ex.: SRTM_rec.tif);
- o código detecta automaticamente o nome do arquivo e atribui à variável raster_path.

Resultado: você terá o caminho do raster (raster_path) disponível para as etapas seguintes.

4.5. Etapa 4 – Abrir o raster e tratar valores NoData

Bloco correspondente:

```
#4 Abrir o raster e tratar valores NoData
with rasterio.open(raster_path) as src:
    altitude = src.read(1) # primeira banda
    nodata = src.nodata
```

```
# Substituir NoData por NaN
if nodata is not None:
```

```

altitude = np.where(altitude == nodata, np.nan, altitude)

# Calcular min e max ignorando NaN
min_altitude = np.nanmin(altitude)
max_altitude = np.nanmax(altitude)

print(f"Altitude mínima: {min_altitude:.2f}")
print(f"Altitude máxima: {max_altitude:.2f}")

```

O que acontece aqui:

O raster é aberto com o rasterio e a primeira banda é lida como matriz altitude. Valores iguais ao nodata do arquivo são substituídos por NaN, para não atrapalharem estatísticas e visualização.

O código calcula altitude mínima e máxima, ignorando NaN, e imprime esses valores.

Interpretação para o usuário:

Se os valores fizerem sentido (ex.: altitudes entre 80 m e 250 m), o raster foi carregado corretamente. Se os valores forem estranhos (muito grandes ou negativos sem sentido), vale verificar o raster original.

4.6. Etapa 5 – Gerar eixos X e Y

O notebook gera malhas de coordenadas para apoiar o gráfico 3D.

Há duas formas:

Uma simples (sem recorte), com:

```
x = np.linspace(0, altitude.shape[1], altitude.shape[1])
```

```
y = np.linspace(0, altitude.shape[0], altitude.shape[0])  
x, y = np.meshgrid(x, y)
```

Outra integrada ao recorte (ver próxima etapa), recalculando X e Y conforme o CROP definido.

Na prática, você não precisa alterar essa parte se está começando; basta executar as células na ordem.

4.7. Etapa 6 – Recorte, suavização e escala vertical + visualização 3D

Esta é a parte mais importante para o usuário ajustar:

Trecho de parâmetros:

```
# --- Parâmetros editáveis ---  
# Janela de recorte (y0:y1, x0:x1). Use None para manter inteiro.  
CROP = (150, 550, 200, 900) # (y0, y1, x0, x1)  
SMOOTH_SIGMA = 1.5          # 0 = sem suavização; 1–3 geralmente bom  
Z_SCALE = 0.35              # <1 "achata" o relevo; >1 exagera
```

CROP: define um recorte da matriz de altitude.

Formato: (y0, y1, x0, x1)

y0:y1 corresponde às “linhas” (vertical) e x0:x1 às “colunas” (horizontal).

Se você quer usar o raster inteiro, pode ajustar para CROP = None (se adaptar o código) ou configurar um recorte que cubra toda a área.

SMOOTH_SIGMA: controla a suavização do relevo via filtro gaussiano.

0 → sem suavização.

1 a 3 → suavizações leves/moderadas, que deixam o relevo mais “fluido” sem perder o padrão geral.

Z_SCALE: controla a exageração vertical.

Valores menores que 1 “achatam” o relevo.

Valores maiores que 1 enfatizam diferenças de altura, úteis em áreas muito planas.

O código:

- aplica o gaussian_filter se o SMOOTH_SIGMA for maior que 0 (instalando scipy se necessário);
- recalcula X e Y com base no recorte;
- aplica Z_SCALE à matriz de altitudes (alt_proc_scaled);
- imprime o intervalo mínimo/máximo após o processamento;
- em seguida, uma célula gera o gráfico 3D:

```
fig = plt.figure(figsize=(12,9))
ax = fig.add_subplot(111, projection='3d')
surf = ax.plot_surface(x, y, alt_proc_scaled, cmap='terrain',
edgecolor='none')
levels = np.arange(np.nanmin(alt_proc_scaled),
np.nanmax(alt_proc_scaled),
max((max_alt-min_alt)/20, 1))
ax.contour(x, y, alt_proc_scaled, levels=levels, colors='red',
offset=np.nanmin(alt_proc_scaled))
fig.colorbar(surf, ax=ax, shrink=0.5, aspect=5)
ax.set_title('Terreno 3D — zoom e suavização')
ax.set_xlabel('Longitude (pixels)')
ax.set_ylabel('Latitude (pixels)')
ax.set_zlabel('Altitude (m, reescalada)')
```

```
plt.show()
```

Você verá:

- a superfície 3D colorida por altitude (cmap "terrain");
- curvas de nível em vermelho projetadas na base;
- barra de cores, indicando o intervalo de altitude;
- eixos X, Y e Z rotulados.

4.8. Etapa 7 – Exportar figura 3D como PNG e baixar

Por fim, o notebook recria o gráfico e salva em PNG, depois faz o download:

```
#7 Exportar figura 3D como PNG e baixar
output_path = "visualizacao3D_terreno.png"
...
plt.savefig(output_path, dpi=300, bbox_inches='tight')
plt.close()

print(f"📄 Figura salva como: {output_path}")
files.download(output_path)
```

O que você precisa fazer:

- executar a célula da Etapa 7;
- a figura será salva com o nome visualizacao3D_terreno.png.

O Colab abrirá uma janela para você baixar o arquivo PNG para o seu computador.

Esse PNG já vem em boa resolução (DPI configurado no código) e pode ser incluído diretamente em relatórios, apresentações ou documentos de PRAD.

5. Recomendações e problemas comuns

a) Figura distorcida ou “achatada demais”

- ajuste o parâmetro `Z_SCALE`;
- se estiver muito baixo, aumente (ex.: de 0.35 para 0.7 ou 1.0);
- se estiver exagerado, reduza.

b) Relevo “muito serrilhado”

- aumente a suavização com `SMOOTH_SIGMA` (ex.: 2.0 ou 3.0);
- lembre-se de que suavizações muito altas podem apagar detalhes finos.

c) Região errada no zoom

- ajuste a tupla `CROP = (y0, y1, x0, x1)` até enquadrar a área de interesse;
- é um processo de tentativa e erro: altere valores, execute, visualize, ajuste novamente.

d) Erro ao importar `gaussian_filter`

- a célula já tenta instalar `scipy` automaticamente com `%pip install scipy`;
- se der erro, execute a célula novamente e verifique sua conexão à internet (no Colab).

6. Aplicações típicas

- visualização 3D didática para aulas de sensoriamento remoto e geoprocessamento;
- geração de figuras para PRAD, estudos de drenagem e morfologia do relevo;
- apoio à interpretação de processos erosivos, vertentes, topos e fundos de vale em áreas de interesse;
- criação de ilustrações para relatórios técnicos e artigos, integrando DEM/MDS com outros produtos da PRAD Toolbox.