Densidade e quantidade de áreas úmidas

Importando pacotes e inicializando *geemap*

```
import os
import ee
import geemap
```

```
geemap.ee_initialize()
```

O índice *Wetland score* é uma combinação da densidade de áreas úmidas localmente com a densidade e quantidade de áreas úmidas regionalmente. *Wetland score* entra no cálculo da diversidade da paisagem em locais planos e úmidos, com baixa variedade de *landforms* e baixa amplitude de elevação. Nesses locais a variação microclimática seria baixa devido a baixa variabilidade topográfica e geomorfológica, mas como há alta densidade de áreas úmidas, esses locais atuam regulando a variabilidade microclimática localmente, como tampões climáticos, e regulando a emissão de gases de efeito estufa.

Existem três cenários de distribuição de áreas úmidas, os locais (1) estão presentes em áreas com alta densidade de áreas úmidas no entorno, (2) os locais estão situados em áreas com baixa densidade de áreas úmidas localmente, mas alta densidade regionalmente e (3) os locais estão presentes em áreas com alta quantidade de áreas úmidas, mas baixa densidade devido a sua distribuição espacial. Desta forma, *wetland score* é composto pelos três cenários citados anteriormente, primeiro é calculado a densidade local, regional e a quantidade regional e, para cada métrica, é calculado um valor de Z, subtraindo pela média e dividindo pelo desvio padrão. A densidade de áreas úmidas é a média ponderada dos valores de Z da densidade local e regional (peso duplo para a densidade local). Nos locais onde os valores de Z da quantidade de áreas úmidas regional é maior que a densidade média calculada anteriormente, o índice torna-se a média ponderada da densidade local, densidade regional e quantidade de áreas úmidas regional (duplo peso para a densidade local).

$$wetland.score = \begin{cases} \frac{2 \times densidade_{local}(Z) + densidade_{regional}(Z)}{3} & \text{se densidade média é maior ou igual à quantidade regional} \\ \frac{2 \times densidade_{local}(Z) + densidade_{regional}(Z) + quantidade_{regional}(Z)}{4} & \text{se a quantidade regional for maior a densidade média} \end{cases}$$

Nesse capítulo demonstraremos como as densidades e a quantidade de áreas úmidas foram calculadas e em outro capítulo mostraremos como calculamos os valores de Z e o *wetland score*.

Banco de Dados

Nós utilizamos como base de dados de áreas úmidas o *Global Wetlands database* (Gumbricht et al. 2017). Nós reprojetamos o *raster* de áreas úmidas para a mesma resolução do modelo digital de elevação usado nas etapas anteriores (~ 90 metros).

Para calcularmos as densidades e quantidade de áreas úmidas, nós retiramos as áreas úmidas classificadas como sistemas lacustres e ribeirinhos (*riverines* e *lacustrines*) e reclassificamos o *raster* como sendo área úmida (1) ou não sendo área úmida (0). Depois, calculamos a densidade de áreas úmidas dentro de uma vizinhaça de 450 metros (5 células) de raio de um *kernel* circular (densidade local). Calculamos também a densidade e a quantidade de áreas úmidas na vizinhança de ~ 1170 metros (13 células) (regional).

Código para calcular as densidades e quantidade de áreas úmidas

As análises foram rodadas no *Google Earth Engine* (Gorelick 2017), devido a demanda computacional do projeto, usando o pacote **geemap** (Wu 2020) em *Python* (Python Software Foundation 2023) como interface pela facilidade na documentação e reprodutividade das análises.

Primeiro, nós importamos os polígonos dos biomas do Brasil e extraímos suas coordenadas geográficas máximas e mínimas para delimitar a região de análise. Importamos o modelo digital de elevação e o *raster* de áreas úmidas e reprojetamos a resolução das áreas úmidas para a resolução do modelo digital de elevação.

```
# Importando mapa de biomas do IBGE para extrair as coordenadas
# mínimas e máximas do Brasil
bioma = ee.FeatureCollection("projects/ee-lucasljardim9/assets/Biome")

def func_cmp(feature):
    return feature.bounds()

# Extraindo as coordenadas mínimas e máximas do Brasil
bioma_box = bioma.map(func_cmp).geometry().dissolve(**{'maxError': 1}).bounds()

# Extraindo a resolução do mapa
DEM = ee.Image("MERIT/DEM/v1_0_3")

escala = DEM.projection().nominalScale()
```

```
# Reprojetando áreas úmidas
wetlands = (ee.Image("projects/ee-lucasljardim9/assets/Cifor_wetlands")
            .reproject(**{'crs': "EPSG:4326",
                          'scale': escala}))
```

Em seguida, retiramos os sistemas ribeirinhos e lacustres do *raster* de áreas úmidas e reclassificamos as classes do raster em presença e ausência de áreas úmidas.

```
# Criando uma máscara para rios e lagos

rivers = wetlands.mask(wetlands.neq(10))

# Retirando rios e lagos das áreas úmidas

wetlands = wetlands.mask(rivers)

# Transformando áreas úmidas em um raster binário
# de presença de áreas úmidas

wetlands_binary = wetlands.where(wetlands.gt(0), 1).unmask()
```

Posteriomente, nós calculamos a densidade de áreas úmidas, localmente, dentro de um *kernel* circular de ~450 metros (5 células). Primeiro, transformamos as células da vizinhança de cada célula focal em bandas de uma imagem. Assim, cada células vizinha da célula focal fica empilhada como uma coluna. Para cada coluna, somamos os valores das células (0 ou 1) como a quantidade de áreas úmidas na vizinhança. Depois, dividimos a quantidade de áreas úmidas pelo número de células na vizinhança, resultando na densidade de áreas úmidas.

```
radius_pixels = 5

# Tranforme as células da vizinha em bandas

neighbors = wetlands_binary
            .neighborhoodToBands(ee.Kernel.circle(ee.Number(radius_pixels)))

# Conte a quantidade de áreas úmidas na vizinhança
wetlands_count = neighbors.reduce(ee.Reducer.sum()).toDouble()

# Conte o número de células totais na vizinhança

neighbors_amount = neighbors.bandNames().length()
```

```
#Divida a quantidade de áreas úmidas pelo
# número de células para calcular a densidade
wetlands_density_local = wetlands_count.divide(ee.Number(neighbors_amount))
```

Repetimos o mesmo procedimento da densidade local para calcularmos a densidade e quantidade de áreas úmidas regional.

```
radius_pixels = 13

# Transforme as células vizinhas em bandas

neighbors = wetlands_binary
           .neighborhoodToBands(ee.Kernel.circle(
                                    ee.Number(radius_pixels)
                                    )
                                )

# Conte a quantidade de áreas úmidas
wetlands_count = neighbors.reduce(ee.Reducer.sum()).toDouble()

# Conte o número de células totais na vizinhança
neighbors_amount = neighbors.bandNames().length()

# Calcule a densidade dividindo a quantidade
# de áreas úmidas pelo número de células
wetlands_density_regional = wetlands_count.divide(ee.Number(neighbors_amount))
```

Por fim, exportamos a densidade de áreas úmidas local (*wetlands_density*), a densidade regional (*wetlands_density_1000*) e a quantidade regional (*wetlands_count*) como *assets* no *Google Earth Engine*.

```
assetId_quantidade = "projects/ee-lucasljardim9/assets/wetlands_count"

assetId_densidade_local = "projects/ee-lucasljardim9/assets/wetlands_density"

assetId_densidade_regional = "projects/ee-lucasljardim9/assets/wetlands_density_1000"

geemap.ee_export_image_to_asset(
    wetlands_count,
    description='wetlands_count',
    assetId=assetId_quantidade,
```

```
    region=bioma_box,
    scale=escala, maxPixels=1e13
)

geemap.ee_export_image_to_asset(
    wetlands_density_local,
    description='wetlands_density_local',
    assetId=assetId_densidade_local,
    region=bioma_box,
    scale=escala, maxPixels=1e13
)

geemap.ee_export_image_to_asset(
    wetlands_density_regional,
    description='wetlands_density_regional',
    assetId=assetId_densidade_regional,
    region=bioma_box,
    scale=escala, maxPixels=1e13
)
```