GeoPandas (visualisatie): Achtergrondkaart toevoegen

Dr. Cornelis Stal

April 28, 2022

1 Achtergrondkaart toevoegen aan een plot

Vaak willen we onze data of de resultaten van onze ruimtelijke analyse projecteren op een achtergrondkaart. In deze sectie demonstreren we hoe basisdata als achtergond toegevoegd kunnen worden aan een plot gemaakt met de `GeoPandas.plot()`-methode. Hiervoor maken gebruik van de contextily-bibliotheek, waarmee web mapping tiles (OpenStreetMap, Stamen, topografische kaarten NGI, …) gebruikt worden.

> **Gebruikshandleiding:** in deze sectie zullen we ons beperken tot enkele nuttige basisfunctionaliteiten. Voor meer informatie over de `contextily`-bibliotheek verwijzen we door naar de gebruikshandleiding.

We starten met het importeren van de vereiste bibliotheken:

[ ]:
```
import geopandas as gpd
import contextily as cx
```

We zullen een snede uit de Vlaamse biologische waarderingskaart (BWK, zie Geopunt voor meer info), en meer bepaald alle zones die enigszinds waardevol zijn (`EVAL LIKE '%w%'`), gelegen rond de HOGENT campus Schoonmeersen in Gent. Deze data worden gedownload via een WFS en als GeoJSON ingeladen in een `GeoPandas`-object.

[ ]:
```
import requests

url = 'https://geoservices.informatievlaanderen.be/overdrachtdiensten/BWK/wfs'
params = {'SERVICE': 'WFS', 'REQUEST': 'GetFeature', 'VERSION': '2.0.0',
    'CQL_FILTER': 'EVAL LIKE \'%w%\' AND BBOX(SHAPE,␣
 ↪102000,190500,104000,192500)',
    'TYPENAMES': 'BWK:Bwkhab', 'SRSNAME': 'EPSG:31370',
    'outputFormat': 'application/json'}
r = requests.head(url, params=params)
bwk = gpd.read_file(r.url)

ax = bwk.plot(figsize=(5, 5), alpha=0.5, edgecolor='g', color='#B2DF8A')
ax.set_xlim(102500, 103500)
ax.set_ylim(191000, 192000)
```

1.1 Achtergronddata en CRS

Vooralleer we web mapping tiles toe kunnen voegen aan onze kaart, dienen we de nodige aandacht te besteden aan de gebruikte coördinaat referentiesystemen (CRS) van zowel de achtergrondkaart zelf als de data uit de biologische waarderingskaart. Het heeft de voorkeur dat beide systemen overeen komen, maar dit is niet vereist. Web mapping tiles worden meestal aangeboden in Web Mercator (EPSG 3857).

De CRS van de biologische waarderingskaart hadden we eerder al ingesteld met de `SRSNAME`-attribuut van de WFS. We controleren het systeem door de waarde van het `crs`-attribuut op te vragen:

[ ]:
```
bwk.crs
```

Zoals verwacht worden de gedownloade data niet geprojecteerd in het Web Mercator CRS. De keuze dient daarom gemaakt te worden in welk systeem we de data geprojecteerd willen hebben:

- Gedownloade data projecteren naar Web Mercator;
- Web mapping tiles projecteren naar Lambert '72;
- Beide datasets in het oorspronkelijke CRS laten staan;
- Beide datasets in nog een ander syteem projecteren.

In de meeste gevallen zullen data transformeren met behulp van de `to_crs()`-methode. We zullen de data uit de biologische waarderingskaart projecteren naar het Web Mercator:

[ ]:
```
bwkWM = bwk.to_crs(epsg=3857)
```

> **Opmerking:** wanneer gebruik gemaakt wordt van web services, kunnen de data meestal ook al rechtstreeks in het juiste CRS gedownload worden. Voer een `GetCapabilities`-'request' uit op de service om te zien welke systemen ondersteund worden, bijvoorbeeld voor de biologische waarderingskaart.

Een achtergrondkaart kan nu eenvoudig toegevoegd worden met de `add_basemap()`-functie uit de `contextily`-bibliotheek:

[ ]:
```
ax = bwkWM.plot(figsize=(5, 5), alpha=0.5, edgecolor='g', color='#B2DF8A')
tiles = "https://www.ngi.be/tiles/wmts/cartoweb/1.0.0/topo/default_bw/3857/
 ↪latest/{z}/{y}/{x}.png"
cx.add_basemap(ax, source=tiles)
ax.set_xlim(411000, 412500)
ax.set_ylim(6626250, 6627750)
```

> **Opmerking:** De data die standaard gebruikt worden door `contextily` zijn aangeboven door Stamen Design. Dit werkt deer goed voor kleinschalige kaarten, maar in veel gevallen ook voor grootschalige kaarten. Om de een of andere reden wordt het bereik en zoomniveau van ons studiegebied niet ondersteund. We maken daarom direct gebruik van het `source`-argument, waarmee we kunnen verbinden met de map tiles service van het NGI.

Het CRS van achtergrondkaart kan eveneens aangepast worden door een waarde mee te geven met het `crs`-argument. Voor grote datasets en datasets waarbij de geometrische kwaliteit van de

brondata behouden moet blijven is dit aanbevolen. In onderstaand voorbeeld wordt de CRS van de BWK-dataset rechtstreeks overgenomen. We hadden echter evengoed de string `EPSG:31370` als attribuut in kunnen stellen:

[ ]:
```
ax = bwk.plot(figsize=(5, 5), alpha=0.5, edgecolor='g', color='#B2DF8A')
cx.add_basemap(ax, crs=bwk.crs, source=tiles)
ax.set_xlim(102500, 103500)
ax.set_ylim(191000, 192000)
```

> **Gebruikshandleiding:** het herprojecteren of transformeren van map tiles zal in veel gevallen resulteren in een reductie van de scherpte van het kaartmateriaal. We verwijzen naar de 'contextily's guide on warping tiles' voor meer informatie over dit onderwerk.

1.2 De graad van detail beheersen

Het is mogelijk de graad van detail ('level of detail') van de basiskaart te beheersen met behulp van de optionele `zoom`-attribuut. Meestal volstaat de standaardinstelling van het zoomniveau van de basisdata, maar in sommige gevallen is een reductie of uitbreiding van de graad van detail wenselijk. Deze waarde wordt echter niet te hoog ingesteld, om de leesbaarheid van het resultaat te garanderen en de grootte van de te downloaden data te beperken:

[ ]:
```
ax = bwkWM.plot(figsize=(5, 5), alpha=0.5, edgecolor='g', color='#B2DF8A')
cx.add_basemap(ax, zoom=16, source=tiles)
ax.set_xlim(411000, 412500)
ax.set_ylim(6626250, 6627750)
```

1.3 Een andere achtergrond gebruiken

Zoals eerder aangegeven zal de 'Stamen Terrain'-stijl standaard gebruikt worden door `contextily`, maar hebben we bij aanvang van deze demo al gebruik gemaakt van de topografische kaarten van het NGI. Het is echter niet vereist om een eigen attribuut aan `source` mee te geven. Andere standaardstijlen zijn namelijk ook beschikbaar via `cx.providers`:

[ ]:
```
cx.providers.keys()
```

Maken we bijvoorbeeld gebruik van 'Stamen TonerLite':

[ ]:
```
ax = bwkWM.plot(figsize=(5, 5), alpha=0.5, edgecolor='g', color='#B2DF8A')
cx.add_basemap(ax, source=cx.providers.Stamen.TonerLite)
ax.set_xlim(411000, 412500)
ax.set_ylim(6626250, 6627750)
ax.set_axis_off()
```

1.4 Labels toevoegen aan de achtergrondkaart vanuit tileservices:

In sommige gevallen zal een basiskaart bestaande belangrijke kaartelementen overlappen, zoals labels. Om alsnog de leesbaarheid van het resultaat in de hand te houden, kunnen we gebruik maken van services die deels transparante tiles aanbieden, zoals 'Stamen TonerLabels' of de labels van de topografische kaarten van het NGI. `contextily` bevat enkele functionaliteiten die deze

transparante lagen zal detecteren en automatisch aan de bovenzijde van de lagenstructuur zal plaatsen:

[ ]:
```
ax = bwkSA_WM.plot(figsize=(5, 5), alpha=0.5, edgecolor='g', color='#B2DF8A')
cx.add_basemap(ax, source=cx.providers.Stamen.TonerLite)
cx.add_basemap(ax, source=cx.providers.Stamen.TonerLabels)
ax.set_xlim(411000, 412500)
ax.set_ylim(6626250, 6627750)
ax.set_axis_off()
```

Door lagen op deze manier gescheiden van elkaar in te laden, is het ook mogelijk om voor deze lagen afronderlijke graden van detail te definiëren. De composiet kan hierdoor geheel naar eigen wens worden opgebouwd:

[ ]:
```
ax = bwkSA_WM.plot(figsize=(5, 5), alpha=0.5, edgecolor='g', color='#B2DF8A')
cx.add_basemap(ax, source=cx.providers.Stamen.Watercolor, zoom=16)
cx.add_basemap(ax, source=cx.providers.Stamen.TonerLabels, zoom=15)
```