Documentation technique

Description générale

Ce programme permet de détecter automatiquement les ouvrages en terre, tels que les remblais et les déblais, le long d'une route à partir d'un Modèle Numérique de Terrain (MNT) fourni. Un module supplémentaire permet de générer des visualisations des profils perpendiculaires à la route.

Architecture du programme et processus de traitement

Le programme est structuré en trois classes principales, qui sont exécutées les unes après les autres lors de l'exécution.

Main Ouvrage Detector

Lorsque le programme principal (main_ouvrages_detector) est exécuté, l'utilisateur est invité à saisir le code de la route, par exemple « A33 ». Ensuite, un dossier de sortie est créé avec le nom output_coderoute, par exemple « output_A33 ».

Les seuils de différence de niveau importants pour la classification des ouvrages en remblai ou déblai sont ensuite définis en interne. Par défaut, le programme utilise une différence de hauteur de 2 mètres pour détecter un profil en remblai (+2 mètres) ainsi que pour détecter un profil en déblai (-2 mètres).

1. ProfileAnalyzer

Fonctions générales :

- Analyse les profils d'élévation perpendiculaires à la route chaque mètre
- Calcule une courbe linéaire pour modéliser le terrain naturelle sans l’existence de la route
- Classifie les profils en remblai ou déblai
- Calcule les caractéristiques des profils en remblai ou déblai

Remarques générales :

Un log est créé et des résultats intermédiaires sont affichés dans la console pour suivre l'avancement du programme et détecter les erreurs si besoin. En outre, lorsque cela est pertinent, des contrôles des entrées et des retours des fonctions sont inclus pour vérifier leur bon format.

Entrées :

La classe utilise le code de la route saisi par l'utilisateur pour charger automatiquement les données concernant la route en question via une API internet. Les données utilisées proviennent de l'IGN et sont disponibles sous la licence ouverte Etalab. Les données « Tronçons de route » de la BD TOPO® sont utilisées afin d'avoir accès aux caractéristiques de la route. Seuls les tronçons de route

avec un numéro correspondant au code de la route spécifiée par l'utilisateur sont chargés. Ensuite, un contrôle est exécuté pour filtrer les tronçons pertinents, de type autoroutier, et éviter d'inclure des pistes avec un même numéro.

Les données altimétriques doivent être fournies par l'utilisateur sous le nom « mnt » avec l'extension « .tif » dans un dossier appelé « data ». Le fichier est lu par la classe pour obtenir les altitudes nécessaires.

Première boucle :

Vu que la route consiste en plusieurs tronçons, on utilise une boucle **`for`** pour itérer sur chacun d'eux. Les tronçons sont traités en série. Pour commencer, un contrôle est effectué pour s'assurer que les données ont le bon format (LineString ou MultiLineString).

Deuxième boucle :

Pour chaque tronçon de route, on traite chaque mètre. Une boucle **`while`** est utilisée à cette fin, commençant à zéro et ajoutant un mètre à chaque itération, jusqu'à ce que la longueur du tronçon soit atteinte.

Calcul de la ligne perpendiculaire :

Pour chaque mètre d'un tronçon, une ligne de 120 mètres perpendiculaire à la route est calculée. D'abord, l'angle de direction de la route est calculé. Pour les 15 premiers mètres de la route, l'angle est calculé entre le point actuel et un point situé 10 mètres plus loin sur la route. Pour le reste du tronçon, l'angle est calculé entre le point actuel et un point situé 10 mètres avant sur la route.

L'ajout de 90 degrés à l'angle permet de calculer un déplacement perpendiculaire en utilisant des fonctions trigonométriques (dx = 60 * cos(angle + 90°) et dy = 60 * sin(angle + 90°), la valeur 60 est utilisée pour obtenir une ligne d'une longueur totale de 120 mètres). Ces valeurs permettent de calculer deux points extrêmes (start_point et end_point) qui définissent les extrémités de la ligne perpendiculaire.

Enfin, la ligne perpendiculaire est créée en connectant les deux points définis, en utilisant la fonction LineString de la bibliothèque **Shapely**.

Calcul de la hauteur moyenne de la route :

Sept points sont déterminés sur la ligne perpendiculaire à la route, de 3 mètres à gauche du point milieu de la chaussée jusqu'à 3 mètres à droite. Pour tous ces points, l'élévation est déterminée à partir du fichier data/mnt.tif. Ensuite, l'élévation moyenne est calculée en divisant la somme des élévations par le nombre de points valides. Le choix d'utiliser l'élévation moyenne sur une partie de la route de 6 mètres a été fait pour minimiser l'impact des artefacts d'altitude qui pourraient fausser les résultats.

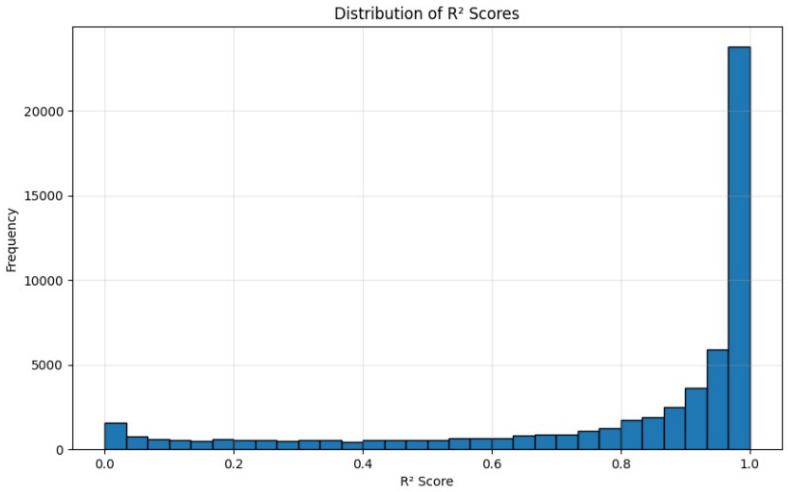
Calcul de la pente naturelle du terrain :

Dans cette étape, on essaie de simuler l'élévation du terrain naturel comme si la route n'était pas présente. Ces données ne sont pas disponibles directement. Pour simuler cela, on prend les deux

zones, à gauche et à droite de la route, comme référence afin de pouvoir interpoler les élévations au niveau de la route. L'estimation est faite en utilisant une régression linéaire simple.

Les scores R^2 pour l'autoroute A33 :

- moyenne : 0,796
- médian : 0,945
- percentile 25 : 0,724
- percentile 75 : 0,988.



Ce résultat est jugé suffisant pour nos objectifs à ce stade, mais il pourra être amélioré en utilisant des méthodes de modélisation plus avancées.

La pente naturelle globale du terrain est alors estimée en prenant le coefficient β1 de la régression. Les coefficients sont sauvegardés en interne pour pouvoir être réutilisés dans les étapes de calcul suivantes.

Estimation de la hauteur naturelle du terrain au niveau de la route :

Le modèle de régression linéaire entraîné lors de l'étape précédente est réutilisé pour estimer l'élévation au point milieu de la chaussée, comme si la route n'existait pas. Ici, l'élévation d'un point suffit, car, par définition, la moyenne des élévations prises sur une courbe linéaire sera toujours égale à la valeur au milieu.

Calcul de la différence d'hauteur :

La différence entre l'élévation de la route et l'estimation du terrain original est calculée en soustrayant l'élévation estimée du terrain original de l'élévation calculée de la chaussée.

Classification des profils :

Si la différence de hauteur est égale ou supérieure au seuil de remblai, spécifié dans le programme principal, le profil est classé comme « remblai ». Si la différence de hauteur est inférieure ou égale au seuil de déblai, également spécifié dans le programme principal, le profil est classé comme « déblai ». Sinon, le profil est classé comme « rasant ».

Calcul des caractéristiques supplémentaires :

Seulement si le profil est classé comme remblai ou déblai, des caractéristiques supplémentaires sont calculées. Les calculs se déroulent légèrement différemment pour les deux types d'ouvrages en terre.

Calcul des attributs d’un profil de déblai :

Tout d'abord, le point de départ est déterminé. À cette fin, on essaie de trouver le point avec l'altitude minimale sur la ligne perpendiculaire entre le milieu de la route et 15 mètres à droite. Ce point minimal constitue le début de l'ouvrage en terre de déblai et se trouve généralement dans le fossé drainant l'eau de la route.

Pour trouver le point de fin de l'ouvrage, on vérifie, à l'aide d'une boucle **`while`**, si une intersection se produit. Une intersection peut être détectée lorsque l'élévation prédite, utilisant la fonction de régression déterminée auparavant, devient plus petite que l'élévation réelle déterminée à partir du Modèle Numérique de Terrain (MNT).

La différence de hauteur maximale est ensuite calculée comme l'altitude au point maximal diminuée de l'altitude au point minimal.

Nous avons développé trois manières différentes pour calculer la pente de l'ouvrage :

1. La pente de l'ouvrage total : calculée par la différence de hauteur maximale divisée par la distance entre le point minimal et le point maximal.
2. La pente d'une section de l'ouvrage commençant à une distance de 2 mètres après le point minimal et s'arrêtant à 2 mètres avant le point maximal : calculée de la même manière, en divisant la différence de hauteur de ces deux points par la distance entre eux.
3. La pente d'une section d'une longueur déterminée (par défaut de 2 mètres) au milieu de l'ouvrage, seulement si la distance entre le point minimal et le point maximal est supérieure à la longueur de la section : calculée de la même manière, en divisant la différence de hauteur entre le point de début de la section et le point de fin de la section par la distance entre eux.

Calcul des attributs d’un profil de remblai :

Trouver le début d'un ouvrage en terre de remblai est plus difficile. Il ne suffit pas de trouver le point avec l'élévation maximale, car celui-ci peut se trouver sur la chaussée et pas nécessairement au bord de celle-ci. À la place, on définit comme début de l'ouvrage le point où l'inclinaison commence et où la pente devient plus grande. Une route a une pente maximale sur la chaussée de 2,5 %, mais sur les bords, la pente peut augmenter jusqu'à 4 %. Avec une marge pour éviter les artefacts dus aux fissures par exemple, on considère que l'ouvrage commence lorsque la pente devient plus grande que 8 %.

Pour trouver le point de fin de l'ouvrage, on vérifie, à l'aide d'une boucle `while`, si une intersection se produit. Une intersection peut être détectée lorsque l'élévation prédite, utilisant la fonction de régression déterminée auparavant, devient plus grande que l'élévation réelle déterminée à partir du Modèle Numérique de Terrain (MNT).

La différence de hauteur et la pente sont calculées de la même manière que pour les ouvrages en terre de déblai.

Sorties :

Un fichier, appelé « classified_profiles.gpkg », au format GeoPackage est créé. Il contient deux objets. Le premier contient tous les points classifiés sur la route avec leurs caractéristiques. Le

deuxième contient des informations sur les points utilisés pour les calculs des attributs des ouvrages en terre de remblai et de déblai, à des fins de vérification.

2. SegmentConstructor

Fonctions générales :

- Regroupe les profils classifiés en segments continus
- Construit les géométries des ouvrages et calcul des attributs supplémentaires
- Les segments sont nommés

Remarques générales :

Des résultats intermédiaires sont affichés dans la console pour suivre l'avancement du programme et détecter les erreurs si besoin. En plus, lorsque cela est pertinent, des contrôles des entrées et des retours des fonctions sont inclus pour vérifier leur bon format.

Entrées :

La classe utilise le code de la route saisi par l'utilisateur pour charger automatiquement les données concernant la route et les points de repère via une API internet. Les données utilisées proviennent de l'IGN et sont disponibles sous la licence ouverte Etalab.

Les données « Route numérotée ou nommée », ainsi que les données « Points de repère » de la BD TOPO® sont utilisées. Seules les données concernant la route spécifiée par l'utilisateur sont chargées.

Les résultats, c'est-à-dire les profils classifiés de l'étape précédente, sont également utilisés.

Boucles principales :

Nous utilisons une boucle `for` pour traiter toutes les géométries présentes dans les données de la route. Après la vérification du format, une deuxième boucle `for` traite tous les *LineStrings* présents dans les *MultiLineStrings*.

Sélection des points de repère :

Nous sélectionnons uniquement les points de repère (PR) pertinents, en utilisant un buffer autour du segment de route, puis en réalisant une intersection entre ce segment tamponné et les PR.

Boucle sur la longueur de la route :

Le paramètre *i* est initialisé à zéro pour commencer la boucle au début d’un segment de route.

Recherche du point classifié le plus proche :

On cherche le point classifié dans les profils classifiés calculés pendant la phase précédente, qui est le plus proche du point actuel sur la route.

Pour accélérer la recherche, on crée une zone de buffer de 5 m autour du point actuel sur la route. Ensuite, on sélectionne uniquement les points parmi les profils classifiés qui intersectent cette zone. Puis, on boucle sur ces points sélectionnés, en calculant à chaque fois la distance au point actuel sur la route, afin de trouver le point le plus proche (c'est-à-dire celui avec la distance minimale).

Détermination de la classification :

Le type de l'ouvrage en terre (*remblai*, *deblai* ou *rasant*) est déterminé à partir des attributs du point du profil classifié sélectionné.

Comparaison avec le(s) prochain(s) point(s) :

Le point situé un mètre plus loin sur la route est interpolé, et de la même manière que pour le premier point, le point le plus proche est ensuite recherché. Si le type d'ouvrage de ce point classifié est différent de celui du premier point, *i* est incrémenté, puis on sort de cette boucle et retourne à la boucle de la longueur de la route.

Si les types sont identiques, on ajoute les attributs du profil à une liste pour ensuite calculer les attributs de l'ouvrage dans son ensemble. Les attributs suivants sont calculés :

- hauteur maximale : la valeur maximale dans la liste des hauteurs.
- hauteur moyenne : la somme de toutes les hauteurs dans la liste divisée par le nombre d'éléments.
- pente maximale : la valeur maximale dans la liste des pentes.
- pente moyenne : la somme de toutes les pentes dans la liste divisée par le nombre d'éléments.

Création d'un segment et calcul des attributs :

S'il y a au moins deux points du même type qui se suivent, on construit un LineString et on calcule des attributs supplémentaires.

- Recherche du dernier PR avant le début et avant la fin du LineString : On crée d'abord un buffer autour du point (de début et de fin respectivement) et on récupère les PR qui intersectent cette zone. Parmi ces PR, on isole les deux plus proches, puis, parmi eux, on sélectionne celui ayant le chiffre minimal.
- Calcul du nombre de mètres sur la route depuis le dernier PR jusqu'au début et jusqu'à la fin du LineString :
    - Étant donné que les coordonnées des PR ne se trouvent pas exactement sur la route, mais parfois juste à côté, et que l'on veut la distance sur la route et non en survol, il faut d'abord trouver le point sur la route le plus proche du PR. Ce point est déterminé en projetant le PR sur la route, ce qui fournit la distance mesurée depuis le début de la ligne jusqu'à la projection orthogonale de ce point sur la ligne. Ensuite une interpolation est réalisée pour obtenir le point sur la route le plus proche du PR.

    - Pour obtenir la distance entre les points de début et de fin du segment et les PR précédents, on calcule les projections de ces points pour obtenir les distances depuis le début de la ligne. Ensuite, ces deux distances sont soustraites l'une de l'autre, ce qui donne une distance en mètres.
- Attribution d'un nom au LineString : Le nom est constitué des éléments suivants :
    - le code de la route
    - le PR du début
    - le nombre de mètres sur la route entre le point de début du LineString et le dernier PR précédent
    - la direction

Finalement, chaque LineString possède les attributs suivants :

- geometry : les coordonnées de tous les points constituant la ligne
- length : la longueur du segment
- classification : le type d'ouvrage (remblai, deblai ou rasant)
- hauteur_max : la hauteur maximale, comme calculé ci-dessus
- pente_max : la pente maximale
- hauteur_moyenne : la hauteur moyenne
- pente_moyenne : la pente moyenne
- PR_start : le dernier PR avant le début du segment
- PR_end : le dernier PR avant la fin du segment
- abcisse_start : le nombre de mètres après PR_start, où commence le segment
- abcisse_end : le nombre de mètres après PR_end, où s'arrête le segment
- nom : le nom du segment
- route : le code de la route

Sorties :

Un fichier nommé « ouvrages_{*codedelaroute}*.gpkg », au format GeoPackage, est créé.

3. OuvragesSelector

Fonctions générales :

- Filtre les ouvrages selon des critères spécifiques :
    - ne garde que les ouvrages en terre de remblai et de déblai
    - supprime les zones des ouvrages qui chevauchent avec des ponts

- supprime les ouvrages trop courts

Remarques générales :

Des résultats intermédiaires sont affichés dans la console pour suivre l'avancement du programme et détecter les erreurs si besoin.

Entrées :

La classe charge les données concernant les ponts via internet à l'aide d'une API. Les données utilisées proviennent de l'IGN et sont disponibles sous licence ouverte Etalab. Les représentations des ponts se trouvent dans la catégorie « Construction surfacique », ainsi que dans la catégorie « Construction linéaire » de la BD TOPO® (avec la nature *Pont*). Un *bounding box* est utilisé pour ne charger que les données situées dans la zone autour de la route spécifiée par l'utilisateur.

Les résultats de l'étape précédente, à savoir les segments d'ouvrages, sont également utilisés.

Sélection des ouvrages :

Seuls les segments avec une classification « remblai » ou « deblai » sont conservés.

Suppression des zones qui chevauchent des ponts :

Pour supprimer les zones chevauchant des ponts, une fonction est créée permettant de filtrer les ponts au formats MutliPolygon, MultiLineString ou LineString.

Pour chaque élément, c'est-à-dire chaque pont, un buffer de 2 m par défaut est créé. Toutes les parties d'un ouvrage qui intersectent cette zone tamponnée sont ensuite supprimées.

Cette action est effectuée deux fois :

1. Une fois pour supprimer les zones chevauchant des ponts représentés par des polygones.
2. Une fois pour supprimer celles chevauchant des ponts représentés par des lignes.

Enfin les ouvrages vides sont supprimés entièrement.

Fusion des ouvrages proches :

Afin de fusionner les ouvrages proche les uns des autres, deux GeoDataFrames séparés sont créés :

4. Un pour les ouvrages classifiés « remblai ».
5. Un pour les ouvrages classifiés « deblai ».

Les ouvrages situés à 10 m ou moins les uns des autres sont ensuite fusionnés.

Pour effectuer la fusion, on crée d'abord des zones de buffer autour des ouvrages, puis on les fusionne. Ensuite, le résultat doit être converti en segments avec une mise à jour des attributs.

Enfin, les deux GeoDataFrame sont concaténés pour obtenir un GeoDataFrame contenant tous les ouvrages en terre de remblai et déblai.

Suppression des ouvrages courts :

Les ouvrages dont la longueur est inférieure ou égale à 20 m sont supprimés.

Sorties :

Un fichier, nommé « selected_ouvrages.gpkg », au format GeoPackage est créé.

4. Get data functions

Module contenant les fonctions pour l'acquisition et le traitement des données géographiques depuis une base de données PostgreSQL via une API.

Fonctions générales :

- récupère des données vectorielles concernant la route à partir du service WFS
- récupère des données vectorielles concernant les ponts à partir du service WFS
- [récupère des données raster représentant le modèle numérique du terrain (MNT) à partir du service WMS : fonction non-fonctionelle]

Remarques générales :

En raison d'un problème de résolution encore non résolu, la fonction *get_mnt()* n'est pas utilisée. Les données LiDAR HD de l'IGN ne sont pas encore disponibles en flux à la date du présent stage.

Entrées :

Pour la fonction *get_data()* et la fonction *get_ponts()*:

- filter : le filtre qui doit être appliqué pour le filtrage des données. Au format '*attribut*='*valeur*''. Peut être vide. Par exemple : 'numero='A33''.
- type_of_data : l'endroit où se trouvent les données. Au format '*BDD:catégorie*'. Par exemple : 'BDTOPO_V3:route_numerotee_ou_nommee'. Ne peut pas être vide.

Spécifications de l'URL et des paramètres :

URL est spécifiée. Les données vectorielles sont accessibles via l'adresse suivante « *https://data.geopf.fr/wfs/ows* ».

Le protocole de communication utilisé est le Web Feature Service (WFS), un moyen standardisé d'accéder à des données géographiques vectorielles (comme des points, des lignes ou des polygones représentant des routes, rivières, bâtiments, etc.) directement via Internet. Ce protocole, défini par l'Open Geospatial Consortium (OGC), est un standard international. Le service WFS du Géoportail de l'IGN implémente la version 2.0 de ce protocole et donne accès à la base de données BD TOPO®, indispensable dans le cadre de ce projet.

Les paramètres suivants sont spécifiés pour la requête WFS :

- SERVICE : WFS (le protocole utilisé)

- REQUEST : GetFeature (le type de requête, ici nous voulons « lire », obtenir des données)
- VERSION : 2.0.0 (version du protocole utilisée)
- TYPENAMES : indique le type de données à interroger, défini par le paramètre d'entrée *type_of_data* (spécifié la base de données et la catégorie d'objet)
- OUTPUTFORMAT : application/json (les données sont retournées au format GeoJSON)
- CQL_FILTER : facultatif, ce paramètre permet de filtrer les données en amont pour optimiser le temps de traitement. Il est modifiable via le paramètre d'entrée *filter*
- SRSNAME : la projection utilisée, ici EPSG:2154 (Lambert-93)
- BBOX : la fonction *get_ponts()* utilise également le paramètre *bbox* pour définir un *bounding box* (ou boîte englobante), qui délimite une zone d'intérêt sur la carte. Cette boîte en forme d'un rectangle est définie par deux coins opposés. Elle permet de réduire le volume de données à charger et ainsi d'optimiser les performances du serveur.

La fonction *get_data()* :

Cette fonction charge les données en utilisant le protocole et les paramètres décrits ci-dessus. Uniquement si le chargement est réussit, les données au format GeoJSON sont converties en un GeoDataFrame, avec une projection explicitement spécifiée pour garantir la compatibilité avec les autres jeux de données. Le GeoDataFrame est ensuite retourné par la fonction.

En cas d'échec du chargement, un message d'erreur avec le code de statut HTTP est affiché dans la console afin de faciliter le débogage.

La fonction *get_ponts()* :

Les ponts étant trop nombreux pour être chargés en une seule fois, cette fonction applique une étape supplémentaire : elle définit d'abord une boîte englobante (*bounding box*) pour restreindre la zone géographique d'intérêt

Dans une première phase, les données liées à la route d'intérêt sont chargées via le protocole et les paramètres précédemment décrits. Un *buffer* (zone tampon) est créé autour de cette route afin de s'assurer que toutes les données pertinentes seront incluses. Autour de la route tamponnée, un *bounding box* est généré, et les coordonnées de ses deux coins sont stockées en interne.

Dans une seconde phase, ce *bounding box*, combiné aux autres paramètres, est utilisé pour interroger les données situées dans la zone délimitée. Si le chargement réussit, les données GeoJSON sont converties en GeoDataFrame avec une projection explicitement définie pour garantir leur compatibilité. Un filtrage supplémentaire permet de ne conserver que les objets représentant des ponts. Le GeoDataFrame est ensuite retourné par la fonction.

Si un des chargements échoue, un message d'erreur avec le code de statut HTTP est affiché dans la console.

Sorties :

Lorsque le chargement est réussi, les fonctions *get_data()* et *get_ponts()* retournent un objet de type GeoDataFrame.

Optionnelle : Main Profiles Constructor

Cette partie du code peut être exécutée séparément, mais nécessite la sortie « ouvrages_*codedelaroute*.gpkg » créée pendant l'exécution du Main Ouvrages Detector.

Ce programme génère des visualisations des profils d'élévation perpendiculaires le long d'un segment routier défini entre deux points de repère (PR). Il utilise un Modèle Numérique de Terrain (MNT) pour extraire les élévations.

Fonctions générales :

- Génère des visualisations graphiques des profils

Remarques générales :

Des remarques sont affichés dans la console pour suivre l'avancement du programme et détecter d'éventuelles erreurs. Plusieurs résultats intermédiaires sont sauvegardés à des fins de vérification des résultats dans un logiciel SIG. Des contrôles sont également effectués à chaque étape pour vérifier le format des résultats intermédiaires.

Entrées :

Plusieurs informations sont demandées à l'utilisateur :

- Code de la route (par exemple A33) : étant donné que ce programme peut être exécuté de manière indépendante, l'utilisateur doit saisir le code de la route.
- PR de début et la distance en mètres après ce PR où l'utilisateur souhaite commencer.
- PR de fin et la distance en mètres après ce PR où l'utilisateur souhaite s'arrêter.
- Espacement : distance souhaitée entre les profils.

Ensuite, le fichier ouvrages_{codedelaroute}.gpkg, créé pendant l'exécution du programme précédent, est lu, et ses données sont assignées à une variable.

Les points de repère sont chargés directement via internet à l'aide d'une API. Les données utilisées proviennent de l'IGN et sont disponibles sous licence ouverte Etalab.

Les données altimétriques doivent être fournies par l'utilisateur sous le nom « *mnt* » avec l'extension « *.tif* » dans un fichier appelé « *data* ». Ce fichier est lu par la classe pour obtenir les altitudes nécessaires.

Connexion des segments :

Les informations concernant la route créées précédemment contiennent des segments de route. Pour obtenir des résultats cohérents, notamment lorsque la partie de la route qui nous intéresse s'étend sur plusieurs segments, il est nécessaire de créer une ligne continue reliant ces segments.

Pour connecter des segments non-ordonnés :

1. On prend le premier segment, on l'ajoute à une liste et on le supprime de la copie des données.
2. Ensuite on crée des zones de buffer autour des points d'extrémité. Par défaut le buffer s'étend sur 5 m autour des extrémités, mais cette valeur peut être ajustée.
3. À l'aide d'une boucle `for`, on vérifie pour tous les segments restants si:
    - La fin d'un segment se trouve dans la zone de buffer du début du premier segment.
    - Le début d'un segment se trouve dans la zone de la fin du premier segment.
4. Si l'un des cas est vérifié, le segment est ajouté à la liste et supprimé des segments restants.
5. Si aucun segment restant peut être connecté, la liste, formant une chaîne, est ajoutée à une liste de chaînes.
6. Ensuite on sélectionne le premier segment encore disponible et on répète la procédure, jusqu'à ce qu'il n'y ait plus de segments restants.
7. Toutes les chaînes ainsi créées sont alors converti en **shapely** LineStrings.

Pour générer des profils transversaux centrés sur la route, une ligne représentant le centre de la route est requise. Actuellement, les segments représentent une chaussée et sont situés approximativement au centre de chaque chaussée. Une route à deux chaussées est donc représentée par deux lignes distinctes.

Afin de fusionner ces lignes et de créer une seule ligne centrale :

1. On crée un buffer autour des lignes.
2. Ces buffers sont fusionnés, produisant un polygone.
3. Enfin, une ligne centrée est calculée avec la fonction *centerline* de la bibliothèque **pygeoops**.
4. Cette ligne est convertie en GeoDataFrame pour permettre le traitement spatial.
5. Elle est ensuite sauvegardée pour vérification dans un logiciel SIG.

Recherche des PR :

On commence par chercher les PR de début et de fin dans les données provenant de l'IGN, en utilisant les entrées fournies par l'utilisateur. Puis on identifie les points sur la route les plus proches des PR correspondants. Ces points sont sauvegardés à des fins de vérification.

Création du segment d'intérêt :

Pour créer le segment correspondant à l'intérêt de l'utilisateur, on détermine les deux points d'extrémité :

1. On mesure la distance sur la route jusqu'au PR de début, en utilisant une projection.

2. À cette distance, on ajoute le nombre de mètres après le PR de début défini par l'utilisateur.
3. Enfin, le point de départ est calculé par interpolation de la distance sur le route depuis le début.
4. Pour déterminer le point de fin, la même méthode est appliquée.

Enfin, une ligne est créée, suivant la trajectoire de la route, entre ces deux points d'extrémité.

Détermination des lignes perpendiculaires :

Des lignes perpendiculaires à la route sont calculées de la même manière que précédemment, avec l'espacement défini par l'utilisateur. Si aucun espacement n'est spécifié, le programme applique par défaut une valeur de 25 m.

Visualisation des profiles :

Pour chaque mètre sur la ligne perpendiculaire à la route, l'élévation est chargée depuis le fichier data/mnt.tif, fourni par l'utilisateur.

Ensuite une visualisation graphique du profil est générée à l'aide de la bibliothèque **matplotlib.pyplot**.

- Sur l'axe horizontal, les distances depuis le début de la ligne perpendiculaire sont tracées.
- Sur l'axe vertical, les élévations sont affichées.

Le nom de chaque profil contient le code du PR et le nombre de mètres (arrondi à la dizaine) après le PR où le point sur la route est situé.

À chaque itération, le profil est sauvegardé au format .png, dans un dossier nommé *profiles*, sous-dossier de output_{codedelaroute}.

Sorties :

Dans le sous-dossier « profiles » du dossier output_{codedelaroute}, se trouvent tous les profils créés durant l'exécution du programme, au format .png. Le nombre de fichiers créés dépend de la longueur du segment défini par l'utilisateur.

Par ailleurs, des résultats intermédiaires sont sauvegardés dans output_{codedelaroute}, afin de faciliter les vérifications dans un logiciel SIG.

Structure globale des données

Entrées

- MNT au format GeoTIFF (mnt.tif), fourni par l'utilisateur et sauvegardé dans un fichier data
- Code de la route (ex: "A33"), à saisir par l'utilisateur
- Données du BD TOPO® de l'IGN, disponibles sous la licence ouverte Etalab et chargées automatiquement via un API (connexion internet obligatoire), en utilisant le module *get_data_functions.py*

- Pour la création des visualisations : code de la route, PR du début, nombre de mètres après le PR début, PR de fin, nombre de mètres après le PR fin, et espacement entre profiles (tout saisi par l'utilisateur)

Sorties

Les fichiers sont sauvegardés dans un dossier output_{route} :

- classified_profiles.gpkg
- ouvrages_{route}.gpkg
- selected_ouvrages.gpkg
- centerline_{route}.gpkg
- pr_points_{route}.gpkg
- segment_start_{route}.gpkg
- chosen_segment_{route}.gpkg
- perpendicular_lines_{route}.gpkg
- profiles/profile_{route}_PR{X}-{Y}m.png

Dépendances

Les bibliothèques utilisées principalement sont :

- geopandas
- shapely
- rasterio
- numpy
- matplotlib
- math
- pygeoops
- numpy
- tqdm
- scikit-learn
- requests

Une liste complète des dépendances se trouve dans le fichier *requirements.txt*.

Utilisation

1. Placer le MNT dans mnt.tif
2. Exécuter main_ouvrages_detector.py
3. Saisir le code de la route

4. Les résultats sont générés dans le dossier output_{route}
5. Les fichiers au format .gpkg (GeoPackage) peuvent être ouverts dans un logiciel SIG (par exemple QGIS ou ArcGIS)
6. Optionnelle : exécuter main_profils_contructor.py
7. Saisir les spécifications demandées
8. Les résultats sont également générés dans le dossier output_{route}

Notes techniques

- Les seuils de classification peuvent être ajustés selon les besoins.
- Le programme gère automatiquement les systèmes de coordonnées.
- Les résultats sont sauvegardés au format GeoPackage pour compatibilité SIG.
- Par définition, on utilise pas d'accent dans la programmation. L'accent sur « déblai » n'est alors pas utilisé dans le code.
- Pour la visualisation des profils en format .png, la distance entre profils est paramétrable.