Laboratoire 1
Vue d'ensemble des sources de données et des outils en transport

Module : Introduction à la recherche et à l'analyse des transports

Durée : 180 minutes

Contents

Objective 2

1 Fichiers & outils 3

2 Activité 1 – Explorer les catégories & sources de données de transport 3

3 Activité 2 – Vérification d'accès aux outils & Hello World 5

4 Activité 3 – Premiers pas sous MATLAB avec un jeu de données de transport 6

5 Activité 4 – Bases de Python avec `Python_Introduction.ipynb` 8

6 Activité 5 – Bases Excel pour les données de transport 9

7 Activité 6 – Découverte de VISUM et VISSIM 9

8 Devoir – Analyse comparative des données de transport sous MATLAB et Python 10

Objectif

Dans ce TP, vous allez :

- classer les principaux types de données de transport et les relier aux formats de fichiers et portails courants,
- explorer au moins un jeu de données réel de transport (comptages, GTFS, ou similaire),
- vérifier l'accès à trois outils d'analyse essentiels : MATLAB, Python/Colab et Excel,
- effectuer un chargement de données et un tracé simples sous MATLAB et Python,
- pratiquer des opérations de base sur tableur dans Excel à partir de données de transport,
- comprendre le rôle de VISUM et VISSIM via une visite guidée de l'interface (GUI) ou une démonstration,
- se préparer à un devoir comparant MATLAB et Python pour l'analyse de données.

L'objectif est de suivre clairement les étapes pratiques et de se familiariser avec les outils et les jeux de données utilisés tout au long du reste du cours.

1 Fichiers & outils

1.1 Fichiers utilisés dans ce TP

Les noms exacts des fichiers peuvent varier selon les universités, mais vous aurez généralement :

- **Jeux de données de transport (exemples)**, p. ex.
  - un exemple de comptages de trafic ou de vitesses (`traffic_counts_example.csv`),
  - un extrait d'horaires de transport public ou un fichier de comptage de passagers,
  - tout petit jeu de données fourni par votre enseignant sur Moodle ou par e-mail.
- **Notebook Python :**
  - `Python_Introduction.ipynb` (syntaxe Python de base et exemples).

1.2 Plates-formes logicielles

- **MATLAB** (bureau ou MATLAB Online).
- **Python** via Anaconda (local) ou **Google Colab** (navigateur).
- **Excel** (bureau) ou Excel Online / Google Sheets.
- **PTV VISUM** et **PTV VISSIM** (souvent sur les PC du laboratoire ; certains étudiants peuvent installer une version étudiante).

Réflexion & points clés

Ce TP est volontairement léger en théorie : l'objectif est de toucher chaque outil au moins une fois, afin que les TP suivants puissent se concentrer sur la modélisation plutôt que sur l'installation ou les problèmes du type où dois-je cliquer ? .

2 Activité 1 – Explorer les catégories & sources de données de transport

Dans les diapositives d'introduction, vous avez vu huit grandes catégories de données de transport (réseau, service, demande, trafic, fret, mobilité partagée, environnement, contexte socio-économique) ainsi que des portails d'exemple tels qu'OpenStreetMap, des répertoires GTFS, Eurostat et des sites nationaux de comptage du trafic.

Tâche 1 : Associer des variables aux catégories de données

Objectif : Relier des variables du monde réel à la bonne catégorie de données de transport.

Étapes :

1. Dans vos notes (ou sur un support imprimé), créez un tableau à trois colonnes :

Catégorie	**Variables typiques**	**Formats d'exemple**
Réseau & infrastructure		
Service / offre		
Demande & habitudes de mobilité		
Trafic & exploitation		
Fret & logistique		
Mobilité partagée & micromobilité		
Environnement & état des actifs		
Socio-économique / usage du sol		

2. À partir des exemples des diapositives, complétez chaque ligne avec :
   - 2 à 3 variables typiques (p. ex. débit horaire, TMJA (AADT) pour trafic & exploitation),
   - 1 à 2 formats ou spécifications courants (p. ex. Shapefile, GeoJSON, GTFS, CSV).
3. Comparez vos réponses avec vos voisins et résolvez les différences.

Point de contrôle : Vous devez pouvoir prendre n'importe quelle variable (p. ex. fréquence des bus ou tonnage weigh-in-motion) et la placer dans la bonne catégorie.

Tâche 2 : Explorer au moins un portail open data

Objectif : Découvrir où obtenir de vrais jeux de données de transport.
Étapes :

1. Avec votre enseignant, visitez brièvement un ou deux portails mentionnés dans les diapositives, tels que :
   - OpenStreetMap / téléchargements Geofabrik,
   - répertoire de données GTFS,
   - portail national de comptage du trafic ou d'enquête déplacements ménages.
2. Pour chaque portail, notez :
   - le type de données fourni,
   - les formats disponibles (p. ex. fichiers texte GTFS zippés, CSV, Shapefile),
   - la couverture temporelle et spatiale (quelles années, quelles régions).
3. Notez un jeu de données qui vous semble intéressant pour un futur projet (nom, lien et description en une ligne).

Optionnel : Si le temps le permet, téléchargez un petit fichier CSV (p. ex. débit horaire à une station) que vous pourrez réutiliser plus tard dans MATLAB et Python.

3 Activité 2 – Vérification d'accès aux outils & Hello World

Avant de commencer l'analyse des transports, nous devons confirmer que MATLAB, Python et Excel fonctionnent sur votre appareil (ou sur un PC du laboratoire).

Tâche 3 : Accès MATLAB / Octave

Objectif : Vérifier que vous pouvez exécuter un script MATLAB simple.
Étapes :

1. Lancez MATLAB. Si vous n'avez pas la version bureau, utilisez :
   - **MATLAB Online** dans votre navigateur, ou
   - **GNU Octave** comme alternative gratuite avec une syntaxe similaire.
2. Dans la fenêtre de commande, tapez :

```
x = linspace(0, 4*pi, 100);
y = sin(x);
plot(x, y);
title('Hello␣Transport␣World');
xlabel('x');
ylabel('sin(x)');
```

3. Vérifiez qu'une fenêtre de graphique apparaît et que vous voyez une courbe sinusoïdale.

Point de contrôle : Si vous voyez le graphique sans erreurs, votre installation MATLAB est prête pour les TP suivants.

Tâche 4 : Accès Python / Colab

Objectif : Vérifier que vous pouvez exécuter du code Python depuis le notebook `Python_Introduction.ipynb`.

Étapes :

1. Ouvrez https://colab.research.google.com et connectez-vous.
2. Importez `Python_Introduction.ipynb` dans Colab (File → Upload Notebook).
3. Exécutez les premières cellules. Si le notebook contient `print("Hello, world!")`, vérifiez que l'exécution se fait correctement.
4. Dans une nouvelle cellule de code, tapez :

```
x = 5
y = 3
print("Sum␣=", x + y)
```

5. Exécutez la cellule et confirmez que `Sum = 8` s'affiche.

Tâche 5 : Accès Excel / Sheets

Objectif : Vérifier que vous pouvez ouvrir et modifier un tableur.

Étapes :

1. Ouvrez Excel (bureau) ou Excel Online / Google Sheets.
2. Créez un petit tableau avec trois colonnes :
   - `time_hour`, `volume`, `lane_id`,
   - et 5 à 10 lignes factices avec des valeurs inventées.
3. Insérez un graphique linéaire simple de `volume` en fonction de `time_hour`.

Point de contrôle : Vous pouvez rapidement créer, enregistrer, rouvrir et modifier un fichier tableur de base.

4 Activité 3 – Premiers pas sous MATLAB avec un jeu de données de transport

Dans cette activité, vous allez charger un petit jeu de données de transport et produire un tracé simple dans MATLAB.

Tâche 6 : Charger et inspecter un jeu de données dans MATLAB

Objectif : Lire un fichier CSV, inspecter sa structure et identifier les variables pertinentes.

Étapes :

1. Placez le jeu de données exemple (p. ex. `traffic_counts_example.csv`) dans le dossier de travail MATLAB.
2. Dans MATLAB, exécutez :

```
T = readtable('traffic_counts_example.csv');
head(T)
summary(T)
```

3. À partir de la sortie, identifiez :
   - le nombre de lignes (observations),
   - les principales colonnes (p. ex. temps, volume, ID de station ou de lien).

Point de contrôle : Vous devez maintenant savoir ce que représente chaque colonne et laquelle convient pour un tracé en série temporelle ou un diagramme en barres par catégories.

Tâche 7 : Créer au moins un graphique dans MATLAB

Objectif : Tracer une relation pertinente à partir du jeu de données.

Étapes :

1. Choisissez une variable numérique à visualiser (p. ex. débit horaire).
2. S'il existe une colonne de temps (p. ex. `timestamp` ou `hour`), créez un tracé en ligne :

```
plot(T.timestamp, T.volume);
xlabel('Time');
ylabel('Volume␣(veh/h)');
title('Hourly␣traffic␣volume');
```

3. Si votre jeu de données est plutôt catégoriel (p. ex. IDs de station), vous pouvez agréger et tracer :

```
G = groupsummary(T, "station_id", "sum", "volume");
bar(G.station_id, G.sum_volume);
xlabel('Station');
ylabel('Total␣volume');
title('Total␣volume␣by␣station');
```

4. Ajustez les étiquettes des axes et le titre selon le contexte de votre jeu de données.

Réflexion & points clés

MATLAB est particulièrement efficace pour les opérations matricielles et le tracé rapide. Ne vous inquiétez pas si votre graphique n'est pas encore parfait : concentrez-vous sur le chargement correct des données et la production d'une figure lisible.

5 Activité 4 – Bases de Python avec `Python_Introduction.ipyn`

Les diapositives ont présenté des notions Python telles que les variables, les types de données, les entrées/sorties, les conditions et les boucles. Le notebook `Python_Introduction.ipynb` reprend ces idées sous forme de code exécutable.

Tâche 8 : Exécuter et modifier des exemples Python de base

Objectif : Se familiariser avec l'exécution et l'édition de cellules Python dans Colab.
Étapes :

1. Dans Colab, exécutez toutes les cellules de `Python_Introduction.ipynb` une fois, du début à la fin.
2. Repérez une cellule qui définit des variables comme :

```
my_int = 7
my_float = 7.0
my_list = [1, 2, 3]
```

3. Modifiez les valeurs (p. ex. changez la liste) et réexécutez la cellule. Vérifiez que la sortie affichée se met à jour.
4. Trouvez une cellule qui définit une fonction, par exemple :

```
def add(a, b):
    return a + b
```

5. Ajoutez une nouvelle cellule où vous appelez cette fonction plusieurs fois avec des arguments différents et affichez les résultats.

Point de contrôle : Vous devez être à l'aise pour modifier des nombres, des chaînes, des listes simples et relancer le notebook sans erreurs.

Tâche 9 : Entrées

Objectif : Pratiquer les structures de contrôle Python de base que vous réutiliserez dans les TP suivants.
Étapes :

1. Repérez dans le notebook la partie où `input()` et `print()` sont utilisés.
2. Modifiez le code afin qu'il :
   - demande à l'utilisateur un nombre,
   - indique si le nombre est positif, négatif ou nul.
3. Ensuite, créez une boucle simple qui affiche les nombres de 1 à 10 et leurs carrés :

```
for n in range(1, 11):
    print(n, n**2)
```

4. Exécutez vos cellules et vérifiez que les sorties correspondent à ce qui est attendu.

Réflexion & points clés

Ces bases (variables, fonctions, conditions, boucles) sont les briques essentielles pour tout, du nettoyage simple des données aux pipelines avancés de machine learning dans les cours suivants.

Tâche 10 : (Optionnel) Charger le même jeu de données en Python

Objectif : Reproduire votre analyse MATLAB en utilisant Python et pandas.
Étapes :
1. Importez dans Colab le même fichier CSV que vous avez utilisé dans MATLAB.
2. Dans une nouvelle cellule du notebook, exécutez :

```
import pandas as pd

df = pd.read_csv('traffic_counts_example.csv')
print(df.head())
print(df.describe())
```

3. Créez un tracé simple, par ex. volume total par station :

```
grouped = df.groupby('station_id')['volume'].sum()
grouped.plot(kind='bar');
```

4. Comparez ce graphique à votre version MATLAB.

6 Activité 5 – Bases Excel pour les données de transport

Tâche 11 : Ouvrir et explorer un jeu de données dans Excel

Objectif : Effectuer un nettoyage simple et une visualisation à l'aide d'un tableur.
Étapes :
1. Ouvrez votre jeu de données (CSV ou XLSX) dans Excel.
2. Vérifiez que :
   - la ligne d'en-tête est correctement interprétée comme des noms de colonnes,
   - les colonnes numériques (p. ex. volumes) ne sont pas stockées en tant que texte.
3. Activez les filtres (Data → Filter) et :
   - filtrez sur une station ou un itinéraire,
   - triez par heure ou par date.
4. Insérez un graphique simple (Insert → Line or Column) pour montrer volume vs. temps ou des comptages par station.

Optionnel : Si vous connaissez les tableaux croisés dynamiques, créez un TCD résumant le volume total par station et par jour.

7 Activité 6 – Découverte de VISUM et VISSIM

VISUM et VISSIM sont des outils spécialisés que vous verrez davantage dans les modules ultérieurs. Dans ce TP, l'objectif est seulement de comprendre *à quoi* ils servent et à quoi ressemblent leurs interfaces.

Tâche 12 : Observer l'interface VISUM

Objectif : Reconnaître les principales parties de l'interface VISUM.
Étapes (souvent guidées par l'enseignant) :
1. Sur un PC de laboratoire (ou via un projecteur), ouvrez un petit projet VISUM.
2. Identifiez :
   - la fenêtre d'édition du réseau affichant les liens et les nœuds,
   - la liste des objets (liens, nœuds, zones, lignes),
   - où les matrices OD et les résultats d'affectation sont affichés.
3. Notez une question type à laquelle VISUM répond bien (p. ex. débits sur les liens, ratios V/C à l'heure de pointe).

Tâche 13 : Observer l'interface VISSIM

Objectif : Distinguer VISUM (macroscopique) de VISSIM (microscopique).
Étapes (souvent guidées par l'enseignant) :
1. Ouvrez un modèle simple de corridor dans VISSIM.
2. Regardez une courte simulation et notez :
   - des véhicules individuels qui se déplacent, s'arrêtent et font la queue,
   - des carrefours à feux et des changements de voie.
3. Discutez en une phrase :
   - Quand choisir VISUM plutôt que VISSIM ?
   - Quand une vue microscopique (VISSIM) est-elle nécessaire ?

Réflexion & points clés

Règle approximative : VISUM pour la planification des réseaux à grande échelle (flux sur de nombreux liens) ; VISSIM pour les opérations détaillées sur des corridors et des carrefours.

8 Devoir – Analyse comparative des données de transport sous MATLAB et Python

Pour le devoir, vous allez répéter une partie du travail en TP de manière plus systématique et rédiger un court rapport.

Tâche 14 : Choix du jeu de données et description

Objectif : Obtenir un jeu de données de transport réel depuis un portail open data.
Étapes :
1. Choisissez un portail de données (p. ex. flux GTFS, Eurostat GISCO, comptages FHWA, site national open data).
2. Téléchargez un jeu de données qui contient :
   - soit des volumes de trafic ou des vitesses, *ou*
   - des horaires de transport public ou des comptages de passagers, *ou*
   - des informations origine-destination / demande de déplacement.
3. Enregistrez le jeu de données au format CSV si possible.
4. Rédigez un court paragraphe décrivant :
   - ce que contient le jeu de données (variables, couverture spatiale/temporelle),
   - où vous l'avez obtenu (portail et URL).

Tâche 15 : Visualisation des données sous MATLAB

Objectif : Produire au moins deux graphiques utiles dans MATLAB.
Étapes :
1. Chargez le jeu de données avec `readtable()` ou une fonction similaire.
2. Effectuez un nettoyage minimal (p. ex. supprimer les valeurs manquantes, filtrer sur une ligne ou une station).
3. Créez au moins deux graphiques, par exemple :
   - un tracé en série temporelle (p. ex. volume ou fréquence au fil du temps),
   - un diagramme en barres comparant des catégories (p. ex. stations, lignes, types de véhicules).
4. Exportez les graphiques en images (PNG ou JPEG) pour les utiliser dans votre rapport.

Tâche 16 : Visualisation des données sous Python

Objectif : Répéter une analyse similaire avec Python/pandas.
Étapes :
1. Dans un nouveau notebook Colab, importez le jeu de données via `pandas.read_csv()`.
2. Utilisez `df.head()`, `df.info()` et `df.describe()` pour comprendre la structure.
3. Créez au moins deux graphiques qui reproduisent (autant que possible) vos graphiques MATLAB (avec `matplotlib` ou le tracé via `pandas`).
4. Enregistrez le notebook et exportez les graphiques en images.

Tâche 17 : Comparer MATLAB et Python

Objectif : Réfléchir aux différences entre outils et présenter vos conclusions.
Étapes :

1. Rédigez une courte discussion comparative (env. 1 à 2 pages) abordant :
   - Quel outil vous a semblé le plus simple pour ce jeu de données ?
   - Lequel offrait le plus de flexibilité pour le tracé et le nettoyage ?
   - Y a-t-il eu des différences dans la manière de visualiser ou d'interpréter les résultats ?
2. Suivez la structure de rapport suggérée :

1	Page de titre (titre du TP, cours, votre nom, ID, date)
2	Objectifs (ce que vous vouliez faire)
3	Introduction (contexte bref sur le jeu de données et les outils)
4	Méthodologie (acquisition des données, étapes MATLAB, étapes Python)
5	Résultats (graphiques et principaux résultats numériques)
6	Discussion (comparaison MATLAB vs. Python, difficultés)
7	Conclusion (principales leçons retenues)
8	Références (jeux de données, logiciels, tutoriels)
9	Annexes (listings de code, graphiques supplémentaires, captures d'écran)

3. Soumettez le rapport PDF et votre notebook Python (et le script MATLAB, si l'enseignant le demande).

Réflexion & points clés

Ne cherchez pas à faire le graphique parfait . Concentrez-vous sur une histoire claire : ce que montre votre jeu de données et comment MATLAB et Python vous ont aidé chacun à mettre cette histoire en évidence.