Analyse et modélisation de trajectoires d'utilisateurs dans des systèmes réels

Noudéhouénou Lionel Jaderne Houssou

LA ROCHELLE UNIVERSITÉ

ÉCOLE DOCTORALE EUCLIDE

LABORATOIRE L3i

THÈSE présentée par :

Noudéhouénou Lionel Jaderne HOUSSOU

soutenue le : **mardi 06 Juillet 2021**

pour obtenir le grade de : **Docteur de La Rochelle Université**

Discipline : **Informatique et Applications**

Analyse et modélisation de trajectoires d'utilisateurs dans des systèmes réels

---

Rapporteurs

Aline CARNEIRO VIANA	Directrice de recherche	INRIA
Céline ROBARDET	Professeur des Universités	INSA Lyon

Examinateurs

Yacine GHAMRI-DOUDANE	Professeur des Universités	La Rochelle Université
Fabien TARISSAN	Chargé de recherche	CNRS

Direction

Jean-Loup GUILLAUME	Professeur des universités	La Rochelle Université
Armelle PRIGENT	Maîtresse de conférences	La Rochelle Université

Thèse réalisée au Laboratoire L3i
Faculté des Sciences et Technologies
Avenue Michel Crépeau
17042 La Rochelle cedex 01

Tel : +33 5 46 45 82 62
Web : `http://l3i.univ-larochelle.fr/`

Sous la direction de Armelle Prigent — Maîtresse de conférences
Jean-Loup Guillaume — Professeur des universités

Financement Etablissement

Résumé

Les progrès technologiques récents ont accéléré la numérisation de nos sociétés et impulsé un usage intensif de terminaux et de plateformes digitales. L'utilisation de ces outils qui touchent pratiquement à tous les aspects de nos vies induit la production et le stockage massif de traces numériques. Ces dernières correspondent principalement à des séquences ordonnées de localisations, qu'il s'agisse de coordonnées GPS, de pages web visitées ou, plus généralement, de toute description des mouvements de leurs auteurs dans le temps et dans l'espace. Ce sont donc des trajectoires d'utilisateurs. Alors que dans certains environnements ces trajectoires peuvent prendre n'importe quelle forme, dans d'autres, elles doivent suivre un réseau qui limite les déplacements possibles. La question se pose alors de savoir comment analyser, efficacement et adéquatement, les trajectoires lorsqu'elles sont contraintes par un réseau. Dans cette thèse, nos travaux ont en partie été structurés par la recherche de solutions à deux verrous scientifiques relatifs à cette problématique. Le premier concerne le recalage à grande échelle, sur un réseau routier, de trajectoires réelles et bruitées de mobilité urbaine. Le second à trait à la notion de distance entre trajectoires contraintes par un réseau qui est indispensable à l'analyse de données de ce type. Nous avons également abordé la problématique plus appliquée de la fusion de données pour la détection de zones fonctionnelles urbaines. Il s'agit là d'une des applications de l'analyse de trajectoires. La singularité de notre travail réside non seulement dans l'emploi de trajectoires réelles, notamment de mobilité urbaine, mais aussi dans l'utilisation d'outils provenant de la théorie des graphes, outils qui facilitent l'intégration des contraintes et spécificités liées aux réseaux.

Mots clés : réseaux complexes, modélisation par des graphes, analyse de trajectoires, distances entre trajectoires, mobilité urbaine, map-matching, zones fonctionnelles.

Analysis and modeling of users' trajectories in real systems

Abstract

Recent technological advances have accelerated the digitization of our societies and driven the intensive use of digital terminals and platforms. The use of these tools, which affect almost every aspect of our lives, induces a massive production and storage of digital traces. These traces correspond mainly to ordered sequences of locations, such as GPS coordinates, web pages visited or, more generally, any description of their authors' movements in time and space. They are therefore user trajectories. While in some environments these trajectories can take any form, in others they must follow a network that limits the possible movements. The question then arises as to how to analyze, efficiently and adequately, the trajectories when they are constrained by a network. In this thesis, our work was partly structured by the search for solutions to two scientific problems related to this issue. The first one concerns the map-matching, at large-scale of real and noisy urban mobility trajectories. The second one is related to the notion of distance between network-constrained trajectories, which is essential for the analysis of this type of data. We also addressed the more applied problem of data fusion for the detection of urban functional areas which is one of the applications of trajectory analysis. The singularity of our work lies not only in the use of real trajectories, in particular urban mobility trajectories, but also in the employment of tools from graph theory that facilitate the integration of constraints and specificities related to networks.

Keywords : complex networks, graph modeling, trajectory analysis, trajectory distances, urban mobility, map-matching, functional areas.

Remerciements

Noudéhouénou man fè. Kpè di do houè nan Adjalonnon kpo do nouminsin mi ton lè kpo.

Bien que ce document consacre l'aboutissement d'efforts personnels intenses, qu'il me soit humblement permis de rendre un hommage appuyé aux personnes, qui tout le long de mon cursus, m'ont aidé et soutenu.

En premier lieu, je tiens à remercier chaleureusement mes encadrants, Jean-Loup GUILLAUME et Armelle PRIGENT, pour m'avoir offert l'opportunité d'effectuer cette thèse. Je retiens de nos échanges nombre de souvenirs positifs. J'ai particulièrement été marqué par la bienveillance et la compréhension mutuelles qui ont constamment caractérisé nos rapports. Je loue votre patience à mon égard pendant les moments les plus difficiles de la thèse. Vous avez su croire en moi et m'avez ainsi aidé à surpasser mes limites. En outre, vos éloges lors de la soutenance m'ont, à la fois, honoré et profondément ému. Par votre truchement et grâce à Renaud LAMBIOTTE, à qui j'adresse au passage mes sincères remerciements, j'ai eu la chance d'effectuer un séjour de recherche à l'université d'Oxford. Ce fut, pour moi, une expérience exceptionnelle qui restera inoubliable.

Ma gratitude va également à l'endroit des rapporteuses de ma thèse, Aline CARNEIRO VIANA et Céline ROBARDET, ainsi qu'aux autres membres de mon jury, Yacine GHAMRI-DOUDANE, le président, et Fabien TARISSAN, pour avoir accepté d'évaluer mon travail. Vous avez su mettre en valeur mes résultats de par vos appréciations mais aussi à travers vos questions, remarques et suggestions résolument pertinentes.

Cette thèse a été menée au sein du laboratoire L3I où j'ai eu la chance de côtoyer des collègues admirables à l'instar de : Céline, Erlandri, Kais, Muzamil, Nour, Marwa, Salah, Hamza, Mahfoud, Marcella, Viviana, Damien, Valentin ... Merci à tous pour la bonne ambiance, les discussions, le soutien et surtout le magnifique cadeau que vous m'avez offert à l'issue de ma soutenance. Prendre les commandes d'un aéronef est pour moi le parfait présent !

En prélude à mon doctorat, j'ai effectué un master au Vietnam, plus précisément à l'IFI, au cours duquel je me suis lié d'amitié avec des personnes remarquables. Je pense à : Flore et Der membres de la P19 avec moi, mais aussi à Fabrice, Gildas, Félix, Landy, Paul, Olivier, Azise et Paterne. Venus d'horizons

divers et très loin de nos pays respectifs nous avons su forger des liens solides de fraternité qui ont perduré jusqu'à présent. Chers amis félicitations, nous y sommes tous parvenus. Pour la plupart Docteurs ou en passe de le devenir. Ce qui n'était qu'un rêve est à présent notre réalité. Merci à vous et bon vent à tous !

C'est aussi le lieu d'évoquer les noms, à titre de reconnaissance, de personnes rencontrées au Bénin, au Vietnam, en France et en Angleterre qui m'ont encouragé et soutenu. J'ai à l'esprit : Aimn, Anna, Armand, Aristide, Arsène, Hubert, Irène, Léonel, Nhung, Oscar, Subashsingh, Tamègnon, Yannick et plus particulièrement la famille VIADENOU, Symphorien, Ginette, Yanelle et Tobi. De près comme de loin vous avez été présents quand il le fallait. Merci infiniment à tous pour la sollicitude et le soutien.

Je ne saurais finir sans évoquer ceux à qui je dois tant : ma famille. En effet, je me sens profondément redevable envers mes parents, Marilyn HOUNKPONOU et André HOUSSOU. Je souhaiterais vous dire que vos sacrifices n'auront pas été vains. Vous avez été d'un indéfectible soutien et une constante source d'inspiration et de motivation durant toute cette thèse. Soyez-en infiniment remerciés. Cet accomplissement est aussi et d'abord le vôtre ! C'est une bénédiction de vous avoir encore avec moi à cette étape de ma vie et de me sentir porté sur les épaules des géants que vous êtes. À toi Charlaine NOUGBODE ma petite sœur, merci d'être pour moi cette source renouvelée de joie et de motivation. Je m'évertue de mon mieux à être, pour toi, un exemple et une preuve que tes rêves sont réalisables. Merci également aux autres membres de ma famille, dont Pierrette DOSSOU-YOVO, Fanny DAGBA, Pascaline et Virginie HOUSSOU, qui m'ont d'une manière ou d'une autre soutenu.

À tous ceux que j'aurais malencontreusement oublié, soyez assurés de ma profonde gratitude !

Table des matières

Table des figures

Liste des tableaux

Chapitre 1

Introduction générale

Sommaire

1.1 Contexte et motivations

1.1.1 Contexte

Qu'ont en commun un cyclone qui balaie une côte, des gnous qui migrent annuellement en quête de pâturages et d'eau, un passager qui prend un taxi ou un train, une touriste qui visite une ville, un étudiant qui écume les pages de Wikipédia à la recherche d'informations et une cliente qui parcours un site marchand pour dénicher le parfait accessoire allant avec sa dernière tenue ? Ils décrivent tous une trajectoire !

Les trajectoires, telles que nous les concevons, sont des séquences ordonnées de localisations (coordonnées GPS, pages web, points d'intérêts, etc.) qui décrivent un mouvement dans le temps et dans l'espace. Ces localisations, qui en pratique sont des traces numériques, se situent généralement dans un environnement réel, par exemple les positions successives d'un taxi qui se déplace dans une ville ou d'un mammifère dont on veut suivre le mouvement dans la savane. Cependant, elles peuvent aussi appartenir à un environnement virtuel, par exemple un utilisateur qui navigue de page web en page web à la recherche d'une information. L'espace n'est alors plus nécessairement géographique. Comparables à des empreintes, ces localisations recèlent un grand nombre d'informations sur le mouvement qui leur a donné naissance.

Selon l'environnement dans lequel elles se produisent, les trajectoires sont classées en deux principales familles dans la littérature : les trajectoires libres et celles contraintes par un réseau. Les trajectoires libres sont celles décrivant des déplacements dans un environnement ouvert où tous les mouvements sont réalisables : à tout moment il est globalement possible d'emprunter la direction souhaitée dans le respect des contraintes physiques (si on se déplace à une certaine vitesse par exemple il n'est pas possible de s'arrêter et de faire demi-tour instantanément). C'est l'exemple des trajectoires d'animaux dans la nature ou de cyclones. Généralement, on considère qu'une trajectoire dans un espace avec des contraintes physiques légères, par exemple la présence d'arbres dans la nature qui limitent de fait certains mouvements, reste dans cette famille. Contrairement aux trajectoires libres, les trajectoires contraintes par un réseau sont associées à des mouvements réalisés dans des espaces délimités par des réseaux de mobilité ou de communication, qui restreignent les possibilités de déplacement, d'action ou d'échanges. C'est le cas, entre autres, des trajectoires de véhicules qui ne se déplacent que le long

des routes dans une ville, de celles de trains dont le mouvement n'est possible que sur un réseau ferroviaire, ou encore de celles de clients d'un site marchand qui ne peuvent évoluer qu'en suivant les liens hypertextes reliant les pages du site. Toutefois, et bien que cette classification binaire soit largement répandue et acceptée, elle ne permet pas de rendre compte de toute la diversité des types de trajectoires observables, car certaines peuvent être difficilement rangées dans l'une ou l'autre des familles. Les trajectoires d'avions qui a priori paraissent libres, les aéronefs pouvant évoluer dans l'air sans obstacles majeurs à partir d'une certaine altitude, illustrent bien cette particularité. En effet, les appareils évoluant dans un espace aérien donné sont constamment soumis aux instructions des tours de contrôle qui guident et régulent leurs déplacements en leur affectant des couloirs de circulation. Les déplacements des avions, et par conséquent leurs trajectoires, sont donc dans une certaine mesure contraints bien qu'elles se déroulent dans un espace ouvert. On pourrait mener le même raisonnement pour ce qui est de la navigation maritime à quelques exceptions près (le mouvement d'un navire n'étant fortement régulé qu'à l'approche des côtes, des ports ou lors de passages par des canaux et détroits). Nous suggérons de classifier comme trajectoires semi-contraintes, ces trajectoires qui se positionnent à la limite des deux familles précédemment décrites, avec des niveaux de contraintes variables et parfois non exprimables par un réseau. L'étude de ce type de trajectoires n'entre pas dans le cadre de cette thèse mais mériterait à la fois une formalisation et des travaux spécifiques.

Hormis l'environnement, le temps est aussi un paramètre déterminant pour les trajectoires car il régit l'ordre entre les localisations les composant. Notons que le temps peut être implicite ou explicite. A titre illustratif, la trajectoire d'une voiture se déplaçant dans une ville peut être décrite uniquement par la suite de ses positions GPS, auquel cas le temps est implicitement présent car les coordonnées GPS sont mesurées en séquence, mais il n'est pas exploitable dans le traitement de la trajectoire. La même trajectoire pourrait également être représentée en indiquant les horodatages (timestamp en anglais) auxquels les positions GPS ont été enregistrées, ce qui rendrait alors explicite le paramètre temporel. Si une visualisation de ces deux trajectoires sur une carte les ferait apparaître comme identiques, celle avec le temps offre plus d'informations et permet de calculer des vitesses moyennes, des pauses, etc.

1.1.2 Motivations

Historiquement, les travaux relatifs à l'analyse des trajectoires abordaient principalement les problématiques liées aux trajectoires libres. Cependant, les études récentes montrent plutôt une tendance à l'analyse de trajectoires contraintes par un réseau. Ce changement s'explique par deux facteurs essentiels.

En premier lieu, les trajectoires contraintes par un réseau se rapportent plus étroitement aux activités humaines et sont bien plus abondantes de nos jours que les trajectoires libres. En effet, ces deux dernières décennies ont été marquées par un ensemble de révolutions technologiques numériques qui ont donné lieu au développement fulgurant et à la démocratisation de terminaux, de plateformes et d'opérateurs fournissant des services qui touchent pratiquement à tous les aspects de la vie. Ces révolutions technologiques combinées à l'accroissement rapide de la démographie mondiale, avec des populations de plus en plus urbanisées et connectées utilisant frénétiquement les nouveaux terminaux et les services digitaux, ont induit une production et une collecte massives de traces numériques et donc de trajectoires. D'autant plus que ces données sont fortement monétisables. En comparaison, les trajectoires libres sont produites et collectées en proportions nettement plus faibles et dans des contextes qui ne sont pas uniquement associés aux activités humaines comme l'éthologie ou la météorologie. Or, du fait de la transformation socio-économique accélérée de nos sociétés, il importe de disposer de connaissances suffisamment fiables et actualisées provenant des activités des populations, afin de cerner et de répondre efficacement à leurs besoins et attentes. D'où l'engouement pour l'analyse des trajectoires contraintes par un réseau qui sont une mine d'informations sur les utilisateurs qui les génèrent.

En second lieu, jusqu'à très récemment, les trajectoires contraintes par un réseau n'étaient pas exploitées à leur plein potentiel. Et pour cause, elles sont relativement difficiles à obtenir car souvent chasse gardée des plateformes et opérateurs numériques ou protégées par des politiques restrictives (à juste titre) de protection des données. Ce qui, notons-le au passage, poussait les chercheurs à privilégier l'utilisation de données générées par simulation, même si à présent plusieurs jeux de données libres de droits sont accessibles. Par ailleurs, les trajectoires contraintes par un réseau ont été régulièrement traitées comme des trajectoires libres dans la littérature. C'est-à-dire sans prise en compte explicite des informations relatives au réseau sur lequel elles sont réalisées. Cela dans l'optique de ne pas avoir à gérer la complexité supplémentaire qu'induit l'intégration des données du réseau mais

aussi pour capitaliser sur les multiples méthodes et outils déjà développés pour les trajectoires libres.

En somme, l'analyse des trajectoires contraintes par un réseau semble présenter un potentiel en matière d'application socio-économiques et de développements scientifiques supérieur à celui des trajectoires libres. Dans cette thèse nous avons ainsi choisi de nous focaliser sur l'analyse des trajectoires contraintes par un réseau et plus précisément des trajectoires relatives à la mobilité urbaine. Essentiellement en raison des problématiques nouvelles que soulèvent leur traitement et pour leurs riches applications potentielles. Nos travaux ont en partie été structurés par la recherche de solutions à deux verrous scientifiques qui sont transversaux aux opérations concernant l'analyse de trajectoires contraintes par un réseau. Le premier est lié à l'exploitation de données réelles de trajectoires contraintes par un réseau et le second à la comparaison de ces dernières. Plus précisément, il s'agit de :

— l'évaluation de la qualité et de la correction à grande échelle des données obtenues après la phase de prétraitement des trajectoires ;

— l'évaluation des performances des mesures de distances et de similarité dédiées aux trajectoires contraintes par un réseau. Indiquons que ces mesures sont indispensables à la plupart des opérations entrant dans le cadre de l'analyse des trajectoires.

Au-delà de ces deux questions transversales, nous avons également abordé la problématique liée à la fusion de données pour la détection de zone fonctionnelles urbaines qui est une des applications de l'analyse de trajectoires décrivant la mobilité urbaine.

La singularité de notre travail réside non seulement dans l'emploi de trajectoires réelles mais aussi dans l'usage d'outils provenant de la théorie des graphes. Nous optons pour de tels outils parce qu'ils facilitent l'intégration des contraintes et spécificités liées aux réseaux qui se modélisent comme des graphes. En outre, ce choix nous permet de généraliser les méthodes et mesures développées à des trajectoires provenant de réseaux de différentes familles.

1.2 Contributions

Cette thèse aura été l'occasion pour nous de proposer et de développer un certain nombre d'idées et de méthodes concourant à l'analyse de trajectoires contraintes par un réseau. Nos contributions se déclinent en quatre points :

❶ La première s'est focalisée sur l'étape de prétraitement des trajectoires composées de données de géolocalisation. Le problème qui se pose concerne la qualité des résultats issus du processus de recalage des trajectoires sur le réseau routier (map-matching en anglais), c'est-à-dire l'action de transformer une séquence de coordonnées GPS en une séquence de segments routiers. Les solutions existantes ne permettaient pas de déterminer puis de corriger à grande échelle les erreurs issues du processus de prétraitement des trajectoires. Pour y pallier nous avons proposé une solution qui, pour chaque trajectoire, détecte si elle contient des discontinuités (cassures dans la séquence de localisations) puis identifie les boucles qu'elle comporte et distingue celles qui sont légitimes de celles qui ne le sont pas.

❷ La seconde contribution est une revue bibliographique concernant les distances spécifiquement conçues pour les trajectoires contraintes par un réseau afin de savoir, étant données deux trajectoires, si elles sont proches/similaires ou pas. Cet état de l'art vient combler un vide puisque les travaux de même nature ne couvraient que les distances pour trajectoires en espace libre. Nous avons, au terme de notre revue, identifié quatre types de distances à savoir : les distances simples, celles de type nœuds à trajectoires, les distances ensemblistes et les distances intégrant des distorsions.

❸ La troisième contribution de cette thèse porte sur la définition d'une nouvelle distance pour trajectoires contraintes par un réseau qui est intervenue à la suite d'une analyse des limites des distances existantes. La nouvelle distance baptisée EQRP (pour Edit distance with Quasi Real Penalities) est une hybridation entre deux distances préexistantes mais de types différents. Cette nouvelle distance considère les trajectoires à la fois comme des séquences et des ensembles de nœuds pour tirer le meilleur parti des deux mondes. Nous avons proposé une comparaison entre la distance EQRP et d'autres distances pour trajectoires contraintes par un réseau qui a montré qu'elle avait globalement de bonnes performances. Ce qui a confirmé l'intuition qui nous conduit à proposer notre nouvelle distance.

❹ La dernière contribution revêt un caractère fortement applicatif car elle a pour objectif de mieux comprendre la structuration des villes en se basant

sur des trajectoires décrivant la mobilité urbaine. Plus précisément, nous proposons un découpage des villes en zones fonctionnelles urbaines. Ce découpage repose sur la construction de graphes d'interactions générés grâce aux données de mobilité et l'intégration de données socio-économiques collectées dans les villes. Ce découpage en zones fonctionnelles pourra servir d'outil de prise de décision pour guider les politiques d'investissements urbains et pour une restructuration des villes qui ne cesseront de croître au cours des années à venir, en raison de la convergence des dynamiques démographiques et économiques. Il faut dire que si elles devraient maintenir leurs organisations actuelles, les villes répondraient de moins en moins bien aux besoins des citadins qui seraient constamment obligés de se déplacer plus loin pour satisfaire leurs besoins et exercer leurs compétences.

1.3 Organisation de la thèse

Notre manuscrit est organisé en six chapitres. Le présent chapitre 1 qui introduit la thèse, est consacré à l'exposé du contexte et des motivations. Nous y indiquons également nos contributions et l'organisation du document. Le chapitre 2 est un état de l'art qui présente, de prime abord, les notions de base sur les graphes et les trajectoires, indispensables pour une bonne compréhension de la suite. Il y est également détaillé, les techniques employées pour le prétraitement des trajectoires et de leurs limites ainsi que des distances pour trajectoires aussi bien espace libre qu'en espace contraint par un réseau. Le chapitre 3 décrit les jeux de données de trajectoires que nous avons utilisé dans nos différentes expériences, le prétraitement qui en a été fait mais aussi les solutions développées pour nous assurer de la qualité des données utilisées. Dans le chapitre 4 nous définissons une nouvelle distance nommée EQRP pour les trajectoires contraintes par un réseau, puis nous la comparons à plusieurs distances du même type. Le chapitre 5 est dédié à nos travaux sur la délimitation de zones fonctionnelles dans les aires urbaines. Le dernier chapitre conclut la thèse et présente les perspectives.

Chapitre 2

État de l'art

Sommaire

Nous présentons dans ce chapitre, organisé en cinq grandes sections, quelques prérequis sur les graphes et les trajectoires ainsi que l'état de l'art sur les traitements essentiels entrant en ligne de compte pour l'analyse des trajectoires. Dans le détail, la première section traite des graphes et la seconde des trajectoires. Les notions qui y sont abordées préparent le lecteur à une compréhension fluide du reste du document. La troisième section présente les techniques de prétraitement, développées pour le nettoyage et la mise en forme des données brutes de trajectoires. Quant à la quatrième section, elle expose et organise les différentes distances proposées dans la littérature pour la comparaison des trajectoires. Enfin, la cinquième section décrit et structure les travaux liés au clustering de trajectoires. Pour chacun de ces trois traitements nous indiquons les limites actuelles de l'état de l'art et nos contributions ou propositions.

2.1 Notions sur les graphes

2.1.1 Définitions

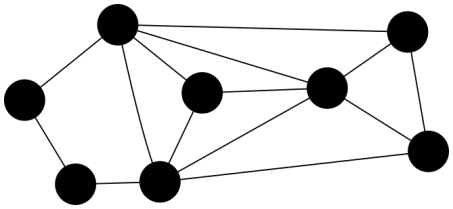
Nous proposons, ci-après, des définitions formelles permettant de mieux appréhender la notion de graphe.

Définition 2.1.1. Un **graphe** $G = (V, E)$ est un couple composé d'un ensemble fini de **sommets** (*vertices* ou *nodes* en anglais) $V = \{v_1, v_2, v_3, ..., v_n\}$ et d'un ensemble d'**arêtes** (*edges* ou *links* en anglais) $E = \{e_1, e_2, e_3, ..., e_m\}$ qui relient les sommets par paires $e = \{v_i, v_j\} \in V^2$. Si $e = \{v_i, v_j\}$ est une arête de G, on dit alors que v_i et v_j sont **adjacents**. Lorsque le sens des arêtes est pris en compte, c'est-à-dire que $\{v_i, v_j\} \neq \{v_j, v_i\}$, le graphe est dit **dirigé**. Les arêtes ne sont donc plus des paires mais des couples (*ordered pair* en anglais) et on parle alors d'**arcs** et non plus d'arêtes. Graphiquement cela se matérialise par l'utilisation de traits pleins pour relier les sommets des graphes non-dirigés alors qu'on représente les arcs par des flèches, comme illustré respectivement sur les figures 2.1a et 2.1b.

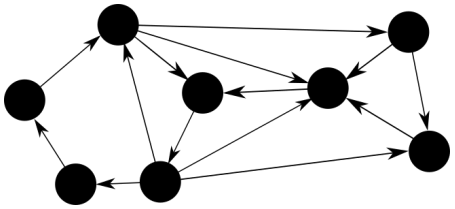
Définition 2.1.2. Un **sous-graphe** $G' = (V', E')$ d'un graphe $G = (V, E)$ est constitué d'un sous-ensemble $V' \subset V$ des sommets de G et d'un ensemble $E' \subset V' \times V' \cap E$ d'arêtes. On parle de sous-graphe induit par V' si E' est exactement l'ensemble $V' \times V' \cap E$, c'est-à-dire que toutes les arêtes de G entre des sommets

de V' sont conservées. La figure 2.1c illustre la notion de sous-graphe.

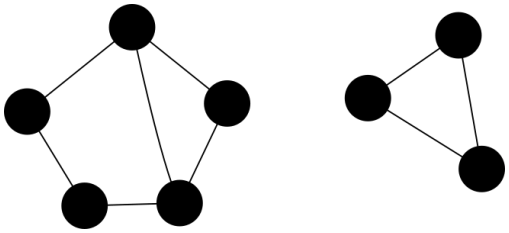
Définition 2.1.3. Un graphe peut être attribué ou pondéré. Les attributs (couleurs, noms, descriptions, identifiants) sont des informations qualitatives ou quantitatives qui peuvent être portées aussi bien par les nœuds que par les arêtes. Les poids quant à eux ne s'appliquent qu'aux arêtes et sont quantitatifs. Ils indiquent généralement l'intensité du lien entre deux nœuds ou la distance les séparant. Un **graphe pondéré** est donc un triplet $G = (V, E, \omega)$ où ω est une fonction associant à chaque arête (ou arc) une valeur numérique. Selon ce que modélise le graphe, ces valeurs seront des entiers ou des réels, uniquement positifs ou pouvant être négatifs.



(a) Exemple de graphe non-dirigé à 8 sommets et 14 arêtes.



(b) Exemple de graphe dirigé à 8 sommets et 14 arcs.



(c) Deux sous-graphes induit du graphe de la figure 2.1a qui correspondent respectivement au 5 sommets de gauche et aux 3 sommets de droite.

FIGURE 2.1 – Exemples de graphes non dirigés, dirigés et de sous-graphes induits.

Définition 2.1.4. Le **voisinage** d'un sommet v est constitué de l'ensemble des sommets partageant une arête avec lui. Autrement dit il se compose de tous les sommets w tels que $\{v, w\} \in E$. Le **degré** d'un sommet correspond à la cardinalité de son voisinage. Pour un graphe dirigé on distingue le **degré sortant** (le nombre

d'arcs qui partent du sommet, c'est-à-dire pour lesquels v est le premier élément du couple) et le **degré entrant** (le nombre d'arcs qui y arrivent).

Définition 2.1.5. Une **chaîne** entre deux sommets v_1 et v_k d'un graphe G correspond à une séquence d'arêtes contiguës $\{e_1, e_2, ..., e_{k-1}\}$ les reliant de proche en proche. C'est-à-dire telles que l'extrémité de e_i correspond au début de e_{i+1}. Elle équivaut de manière équivalente à la suite de sommets adjacents $\{v_1, v_2, ..., v_k\}$ connectés par lesdites arêtes. Un **chemin** est défini de manière similaire dans un graphe orienté. La **longueur** d'une chaîne ou d'un chemin est égale au nombre d'arêtes ou d'arcs la composant.

Définition 2.1.6. Dans un graphe non orienté, un **cycle** est une chaîne dont les extrémités coïncident. Dans un graphe orienté on parle plutôt de **circuit**. Un graphe sans cycle est qualifié d'**acyclique**. Une **boucle** est une arête qui relie un sommet à lui-même, c'est-à-dire dont les deux extrémités correspondent au même sommet. Un graphe $G = (V, E)$ est dit **simple** s'il ne contient pas de boucles et si toute paire de sommets est reliée au plus une fois, c'est-à-dire qu'il ne contient pas d'arêtes multiples [1] : $E \subset \{\{x, y\} \in V^2 | x \neq y\}$. Un graphe G est **connexe** s'il existe une chaîne entre tous ses sommets. Si G n'est pas connexe, chacun des sous-graphes maximaux connexes de G est appelé **composante connexe**. Dans le cas de graphes dirigés on parle de **composante fortement connexe**.

Définition 2.1.7. La **distance** entre deux sommets est la longueur de la plus courte chaîne (ou du plus court chemin pour un graphe dirigé) les séparant. La plus grande des distances entre deux sommets d'un graphe équivaut au **diamètre** de ce graphe. L'**excentricité** d'un sommet est la distance maximale entre ce sommet et tous les autres sommets du graphe. Le **centre** d'un graphe est composé du ou des sommet(s) ayant la plus faible excentricité. Le **rayon** d'un graphe équivaut à l'excentricité d'un sommet appartenant au centre de ce graphe.

2.1.2 Représentation et stockage

La représentation et le stockage des graphes se fait généralement en utilisant :

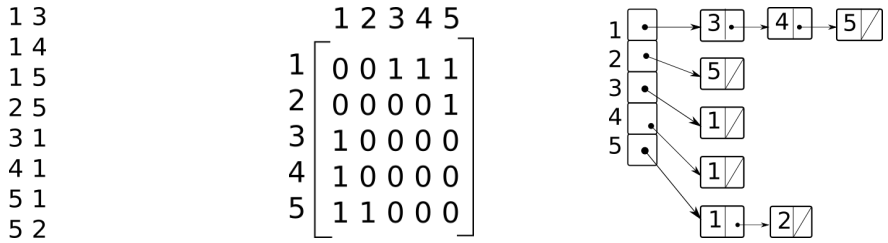
1. Un multigraphe pouvant autoriser les arêtes multiples, l'ensemble des arêtes est alors un multi-ensemble pour permettre les répétitions

— une liste d'arêtes dans laquelle chaque arête est représentée par la paire de nœuds qu'elle relie ;

— une matrice d'adjacence définie telle que $a_{i,j} = 1$ s'il y a une arête entre i et j, 0 sinon ;

— une liste ou un tableau d'adjacence par sommet qui indique pour chacun ses voisins.

La figure 2.2 illustre chacune de ces représentations.

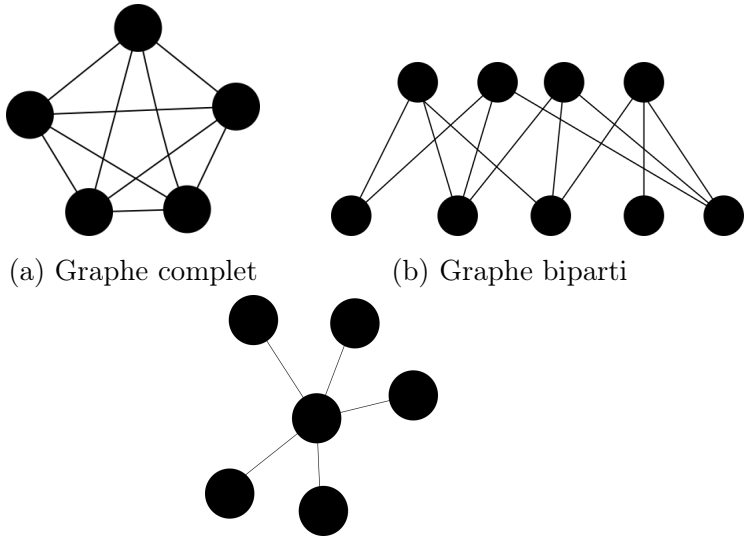


(a) Liste d'arêtes (b) Matrice d'adjacence (c) Liste d'adjacence

FIGURE 2.2 – Représentations des graphes.

2.1.3 Graphes particuliers

Un graphe est dit **complet** lorsque toutes les paires de sommets sont reliées par une arête. Un graphe complet simple contenant n sommets aura $n*(n-1)/2$ arêtes. Un graphe **biparti** est un graphe dont l'ensemble des sommets V se subdivise en deux sous-ensembles disjoints V_1 et $V_2 : V = V_1 \cup V_2$ et $V_1 \cap V_2 = \phi$. Chaque arête du graphe biparti a une extrémité dans V_1 et l'autre dans V_2. Ainsi, les sommets d'un même sous-ensemble ne sont pas reliés entre-eux. Un **arbre** est un graphe connexe acyclique. Chacun de ces trois graphes particuliers est illustré dans la figure 2.3.



(a) Graphe complet (b) Graphe biparti

(c) Arbre

FIGURE 2.3 – Exemples de graphes particuliers.

2.2 Notions sur les trajectoires

2.2.1 Définitions

Comme indiqué en introduction, les trajectoires sont globalement des séquences ordonnées de localisations qui décrivent un mouvement dans le temps et dans l'espace. Cependant, d'un point de vue formel, leur définition varie suivant la nature des localisations qui les composent et en fonction de la façon dont elles sont représentées. Généralement ces localisations sont représentées par des symboles. Nous présentons ici des définitions formelles basées sur les symboles les plus couramment utilisés : les coordonnées GPS et les nœuds et arêtes de réseau.

Définition 2.2.1. Dans un espace géographique, les symboles constituant les trajectoires correspondent souvent aux coordonnées GPS. Chaque symbole représente ainsi une position géographique décrite par une latitude et une longitude. Formellement une trajectoire en espace géographique, avec information temporelle, s'écrit :

$$T = \{(lat_1, lon_1, t_1), (lat_2, lon_2, t_2), ..., (lat_i, lon_i, t_i), ..., (lat_n, lon_n, t_n)\} \\ \text{Avec } t_1 < t_2 < ... < t_i < ... < t_n. \tag{2.1}$$

où lat_i et lon_i indiquent respectivement la latitude et la longitude du point

GPS i tandis que t_i représente l'instant auquel il est enregistré.

Définition 2.2.2. Pour des trajectoires contraintes par un réseau, les coordonnées GPS sont remplacées par les identifiants des nœuds ou des arêtes dudit réseau. Formellement les trajectoires contraintes par un réseau s'écrivent :

$$T = \{(v_1, t_1), (v_2, t_2), ..., (v_i, t_i), ..., (v_n, t_n)\}$$
$$\text{ou} \tag{2.2}$$
$$T = \{(e_1, t_1), (e_2, t_2), ..., (e_i, t_i), ..., (e_n, t_n)\}$$

Avec v_i les nœuds du réseau, e_i ses arêtes et t_i les horodatages associés.

Suivant le contexte, d'autres symboles tels que les identifiants de lecteurs RFID, de cellules GSM, de points d'accès wifi ou encore les clics d'utilisateurs sur une page web (pour ne citer que ceux-là) peuvent être exploités afin de définir formellement les trajectoires.

2.2.2 Modélisations des trajectoires

Dans l'optique de faciliter leur stockage et leur exploitation numérique, les trajectoires peuvent être préalablement modélisées afin d'en synthétiser les informations. Nous décrivons dans cette sous-section quelques-unes des modélisations habituellement associées aux trajectoires.

Modélisation Origine-Destination

C'est une modélisation qui ne retient que l'origine et la destination d'une trajectoire. Cette modélisation permet, entre autres, de générer une matrice Origine-Destination (parfois notée OD) qui décrit les flux entre différents emplacements de l'espace considéré. Bien que simple, elle est suffisante pour nombre d'applications telles que l'identification des principales zones de départ et d'arrivée des déplacements urbains, le découpage de zones urbaines basée sur les données de mobilité, etc. L'écriture formelle d'une trajectoire dont on ne retient que l'origine et la destination est la suivante :

$$T = \{(O, t_1), (D, t_n)\} \tag{2.3}$$

Avec O et D respectivement l'origine et la destination et $t_1 < t_n$. O et D sont des symboles comme définis précédemment tandis que t_1 et t_n correspondent respectivement aux instants de départ et d'arrivée.

Modélisation par morceaux

Dans la modélisation par morceaux (*piecewise* en anglais), les morceaux correspondent à des régions de l'espace ou encore à des portions de la trajectoire. Dans le premier cas, l'espace est découpé en plusieurs zones et chacun des points constituant la trajectoire est associé à une zone. La trajectoire d'origine, initialement constituée de points, est donc transformée en une séquence d'identifiants de zones. Dans le second cas, les morceaux représentent des portions de la trajectoire. Chaque morceau est une approximation de la trajectoire originale (par exemple une moyenne des points).

La modélisation par morceaux permet de réduire la dimension ou le nombre de points des trajectoires. Elle induit l'écriture formelle suivante :

$$T = \{(p_1, t_1), (p_2, t_2), ..., (p_i, t_i), ..., (p_m, t_m)\} \tag{2.4}$$

Avec p_i une portion de la trajectoire ou une région de l'espace et $t_1 < t_2 < ... < t_m$.

2.3 Prétraitement des trajectoires

A l'instar des images, du texte, des séries temporelles et d'autres types de données collectées, les trajectoires brutes recueillies en conditions réelles sont inutilisables en l'état car bruitées et non structurées. Elles nécessitent un traitement préalable ayant pour objectif de les corriger et de faciliter leur exploitation. Ce prétraitement englobe diverses opérations telles que la suppression des données aberrantes, la détection des points d'arrêts, la segmentation, la compression ou encore le map-matching [Zhe15].

La suppression des données aberrantes, dues aux perturbations auxquelles sont sujets les signaux GPS et à leur imprécision intrinsèque, vise à disposer de données cohérentes pour l'extraction de connaissances fiables [YZXS11]. Si certaines données aberrantes sont simples à détecter, telles que l'absence de coordonnées, des coordonnées en dehors du périmètre d'étude..., d'autres peuvent être plus complexes. C'est le cas par exemple de trajectoires dont la séquence de points GPS parait normale mais qui présentent des incohérences telles que des portions parcourues à des vitesses anormalement élevées et inatteignables en pratique, dès que

l'on intègre la dimension temporelle.

La détection des points d'arrêt dans les trajectoires permet d'indiquer les endroits où les déplacements ont été fortement ralentis sinon interrompus [YZZ+11, YZZX12]. De tels points peuvent aider à détecter la formation d'embouteillages dans le trafic, à identifier la localisation de points d'intérêts (stations-service, drives, lieux touristiques, etc.) prisés par les conducteurs ou encore servir à la distinction des trajectoires. En effet, dans certains jeux de données brutes, les séquences de localisations GPS correspondant à un ensemble de trajectoires mélangées, sont simplement listées à la suite sans information permettant de les distinguer les unes des autres. C'est en particulier le cas de trajectoires de taxis qui peuvent être enregistrées comme une seule grande trajectoire qui correspond à la journée de travail d'un conducteur.

La segmentation des trajectoires, a pour but de les découper en morceaux encore appelés sous-trajectoires [ZCL+10]. Ce découpage est plus général que la simple recherche de points d'arrêts car d'autres critères que l'arrêt peuvent être utilisés. Il est en particulier utile dans des applications telles que la détection des modes de transports utilisés ou pour la détection de groupes similaires de sous-trajectoires. Ces segments sont souvent utilisés dans le clustering de trajectoires comme nous le verrons dans la section 2.5.

La compression des trajectoires aide quant à elle à réduire leur taille, en supprimant un certain nombre de points, notamment ceux qui influent peu ou pas sur leur forme ou qui n'apportent que peu d'information comparativement aux autres points [SSZZ14]. Cette technique permet également de diminuer la taille des jeux de données de trajectoires et facilite leur stockage. Il faut noter que la compression se fait généralement en fonction d'une marge d'erreur définie par les opérateurs qui permet de limiter la perte d'information engendrée.

Enfin, le **map-matching**, qui n'a d'intérêt que pour les trajectoires composées de localisations GPS situées sur un réseau routier, a pour objectif de recaler les trajectoires brutes sur les segments routiers [WLH+17, ZHG17]. En effet, les séquences de points GPS qui constituent les trajectoires divergent souvent du tracé du réseau routier en raison de l'imprécision liée à la technologie de localisation par GPS.

De manière générale le prétraitement peut concerner tout ou partie de ces opérations suivant la qualité des données et les applications visées. Dans cette section, nous ne présenterons que le map-matching car les jeux de données que nous cherchons à exploiter ont déjà subi l'essentiel des autres prétraitements. Le map-matching est donc la principale étape nécessaire à leur exploitation.

2.3.1 Le map-matching

Comme indiqué précédemment, l'un des problèmes qui justifie le prétraitement des trajectoires réelles est que, très souvent, elles ne coïncident pas avec le réseau routier en raison des erreurs de mesure intrinsèques du GPS. Par exemple, alors que l'on s'attendrait à ce que les points GPS des trajectoires d'automobilistes dans une ville, soient situés le long des segments routiers, on observe plutôt qu'ils sont généralement répartis de part et d'autre du tracé des routes. Les lignes des trajectoires semblent alors passer par des zones impraticables telles que les bâtiments, les cours d'eau, les terre-pleins et espaces verts. Il est possible de tolérer ces imprécisions pour certaines applications comme par exemple, la détection de zones de forte affluence. Cependant, dans le cas de la gestion du trafic routier où l'on cherche à déterminer le niveau de congestion ou de fluidité des routes, il importe de disposer du parcours exact des trajectoires. La détermination de ce parcours exact se fait en recalant de façon cohérente les points GPS qui composent les trajectoires sur le réseau routier via une opération dénommée map-matching (il s'agit d'un anglicisme qui signifie littéralement faire correspondre sur une carte). La figure 2.4 illustre à la fois les imprécisions d'une trajectoire brute qui est située à plusieurs reprises dans des bâtiments (en rouge) et son map-matching (en vert).

Cette opération de map-matching est cruciale dans la chaîne de traitements des trajectoires contraintes par un réseau car elle permet de les exploiter plus précisément et de prendre explicitement en compte, si besoin, la structure du réseau sous-jacent.

Si le map-matching d'une trajectoire peut paraître une tâche simple pour un opérateur humain, elle est en pratique assez complexe en raison de facteurs liés à la configuration du réseau routier mais aussi à la qualité des données. Au nombre des facteurs dépendant du réseau routier on peut citer : les sens interdits, la présence de tunnels, les échangeurs ou encore les routes parallèles. Quant aux facteurs associés à la qualité des données, ils concernent les points GPS aberrants (dont la position

FIGURE 2.4 – Illustration de l'opération de map-matching (En rouge la trajectoire brute et en vert celle corrigée.)

est incohérente comparativement aux autres points de la séquence à laquelle ils appartiennent), les coupures dans les trajectoires qui génèrent des sous-trajectoires non reliées et les amas de points GPS surtout au niveau des intersections qui indiquent souvent qu'un arrêt a été effectué. Cette complexité du map-matching déjà importante pour un opérateur humain est accrue lors de sa mise en œuvre automatisée. Cela en a fait une problématique scientifique majeure dans le domaine de l'analyse de trajectoires, statut attesté par le nombre important de publications scientifiques sur le sujet. La classification des approches proposées est rendue difficile par la grande variété des méthodes, nous en présentons ici quelques-unes.

2.3.1.1 Classifications des méthodes de map-matching

La classification la plus générale, retenue par [LZG+20], divise les travaux en deux grandes familles : les approches locales (ou incrémentales) et celles dites, globales.

Les approches locales recalent chaque point (ou portion d'une trajectoire) indépendamment des autres. Elles traitent les trajectoires par partie et sont en parti-

culier adaptées au map-matching en ligne qui consiste à recaler les flux de données GPS au fur et à mesure de leur enregistrement. C'est par exemple le type de map-matching qui est utilisé dans les véhicules par les logiciels de géo-localisation en temps réel qui indiquent au conducteur le chemin emprunté ou à emprunter.

Les approches globales, à l'opposé, tiennent compte de l'ensemble des points constituant la trajectoire dans le processus. Elles recherchent le chemin sur le réseau routier qui correspond le mieux à l'entièreté de la trajectoire traitée. Ces approches sont adaptées au map-matching hors ligne qui s'opère sur des données déjà entièrement enregistrées, car la correction d'un point donné peut s'appuyer à la fois sur les points précédents de la trajectoire mais également sur les points suivants.

Bien qu'elle se veuille simple et universelle cette classification ne permet pas de rendre compte assez finement de la richesse de l'état de l'art relatif au map-matching. Nous lui préférons donc la classification proposée par Quddus et al. dans leur revue [QON07], qui sert de référence dans le domaine. Quoiqu'antérieure, cette classification est plus précise. Elle organise les algorithmes proposés en quatre groupes : les approches géométriques, topologiques, probabilistes et avancées.

Les approches géométriques sont les plus anciennes méthodes de map-matching. Elles regroupent des techniques qui ne tiennent compte que de la proximité entre les trajectoires et les segments routiers et se déclinent en trois catégories. La première est celle du map-matching point à point [BK+96], où chaque point GPS est associé au point le plus proche sur un segment du réseau routier. La seconde catégorie dite du map-matching point à courbe [WBK00] fait correspondre chaque point enregistré au segment de route le plus proche. Quant à la troisième catégorie elle concerne le map-matching courbe à courbe, qui rattache des morceaux entiers de la trajectoire à des segments de route [ZHG17].

Les approches géométriques sont des méthodes très rapides à l'exécution. Cependant, elles sont habituellement imprécises car les segments les plus proches des points GPS ne sont pas toujours les plus pertinents. En outre, elles génèrent régulièrement des chemins discontinus c'est-à-dire qui présentent des sauts entre les segments routiers du fait qu'elles traitent chaque point indépendamment des autres.

Les approches topologiques [Y+10, QOZN03] étendent les approches géométriques et les améliorent, en associant le critère de proximité aux propriétés topologiques du réseau routier (c'est-à-dire les connexions entre les segments routiers, leurs courbures, les sens de circulation) dans l'opération de map-matching. Les approches topologiques peuvent être affectées par les données aberrantes au sein des trajectoires et la qualité des données reflétant le réseau routier qui génèrent des erreurs dans l'association des points GPS aux bons segments routiers.

Les approches probabilistes comme celle développée dans [OQN03] définissent une zone d'erreur possible, autour de chaque point d'une trajectoire. La taille de la zone dépend de l'écart-type de l'erreur du GPS utilisé qui est le plus souvent connu d'avance car fournit par le constructeur de l'équipement. On considère alors que le segment routier qui correspond au point est contenu dans cette zone d'erreur ou, inversement, que tous les segments routiers en intersection avec la zone d'erreur sont des solutions potentielles. Pour sélectionner le meilleur segment routier, on utilise des critères comme la direction de la trajectoire, la proximité géographique ou encore la vitesse du véhicule. Ainsi, la définition d'une zone d'erreur peut réduire le nombre de comparaisons à effectuer pour associer un point à un segment routier. Toutefois la dimension de cette zone est critique : trop petite elle pourrait ne correspondre à aucun segment routier et trop grande elle en prendrait un trop grand nombre en compte.

Les approches avancées regroupent des approches plus complexes que celles présentées ci-avant, et qui modélisent l'incertitude liée à l'imprécision des points GPS par le biais d'outils provenant de domaines d'applications variés mais s'adaptant bien à la résolution automatique du problème de map-matching. Au nombre des outils habituellement utilisés dans l'élaboration d'approches complexes pour le map-matching, nous retrouvons : la logique floue, la théorie des croyances de Dempster-Shafer, les filtres de Kalman, les filtres à particules, le modèle de Markov caché ordinairement désigné par son acronyme anglais HMM (Hidden Markov Model) et les champs aléatoires conditionnels (CRF en anglais).

❶ La **logique floue** [FLW04, QNO06] permet de fusionner les informations provenant de différentes sources pour produire une décision. Elle renvoie des valeurs de vérités comprises dans l'intervalle entre 0 et 1 contrairement à la logique booléenne qui génèrent des valeurs binaires (soit 0 soit 1). Elle évalue le

degré de vérité des propositions qui lui sont fournies en entrée et permet ainsi de tenir compte de leurs imprécisions. Dans le contexte du map-matching les propositions peuvent combiner des critères tels que la proximité entre les points GPS et les segments routiers, la vitesse du véhicule, sa direction ou encore la topologie du réseau routier. Tout comme la logique floue, la **théorie des croyances de Dempster-Shafer** [ENB05] est à même de combiner un ensemble de propositions incertaines pour produire un degré de vérité ou, pour reprendre la terminologie du domaine, un degré de croyance. La différence entre les deux approches tient au fait que la théorie des croyances de Dempster-Shafer repose sur l'utilisation de fonctions et de propriétés issues de la théorie des probabilités.

❷ Le **filtre de Kalman** [WLH+17] est une méthode d'estimation de l'état d'un système dynamique à partir de données imprécises (bruitées). Il modélise le système par le biais de matrices de variables aléatoires permettant de relier l'état précédent du système à l'état courant. Son fonctionnement est structuré en deux étapes : la première est consacrée à la prédiction de l'état courant du système et la seconde à une correction des résultats de la prédiction, en exploitant les données d'observation issues du système. La particularité de cette méthode est qu'elle fournit, en plus de l'estimation de l'état du système, une information sur l'erreur moyenne associée à cette estimation. Utilisé dans le contexte du map-matching, le filtre de Kalman prend en entrée les données GPS qui correspondent aux observations du système et prédit les localisations les plus probables qui peuvent leur être associées, sur les segments routiers. A l'instar du filtre de Kalman, le **filtre à particule** [WN16] permet également d'estimer l'état d'un système qui évolue, à partir d'observations bruitées. Toutefois, tandis que le filtre de Kalman détermine analytiquement les valeurs des variables aléatoires décrivant l'état courant du système, le filtre à particule, pour sa part, calcule une approximation de ces valeurs en utilisant des procédés aléatoires d'échantillonnage, usuellement désignés par le terme méthode de Monte-Carlo.

❸ Le **modèle de Markov caché** [NK09, GDM+12] permet de modéliser un système dynamique que l'on suppose régi par un processus de Markov, c'est-à-dire dans lequel chaque état ne dépend que de l'état qui lui est immédiatement antérieur. Le terme "caché" indique que les états du système ne sont pas directement accessibles mais sont décrits par des observations qui

elles sont disponibles. Au fur et à mesure que les observations sont enregistrées, le modèle affecte à chaque état une probabilité dite d'émission qui quantifie la possibilité que l'observation courante provienne de cet état. Les états du modèle sont liés par des transitions auxquelles sont associées des probabilités de transition chiffrant la chance de passer d'un état à l'autre. Les algorithmes utilisant le modèle de Markov caché pour le map-matching associent à chaque point GPS un groupe de segments routiers situés à proximité et leur affectent individuellement une probabilité d'émission, fonction de la distance les séparant du point GPS. Ensuite, ils définissent entre les segments routiers associés à deux points GPS consécutifs des probabilités de transition. La séquence de segments routiers (les états) correspondant le mieux aux points GPS (les observations) est alors celle qui maximise une fonction objectif combinant les probabilités d'émission et de transition. Les **champs aléatoires conditionnels** [BBD+16] sont une amélioration des modèles de Markov cachés définis tels que l'hypothèse de forte indépendance entre les observations est assouplie. Ils aident à mieux modéliser la relation de dépendance existant entre les observations.

❹ En complément des approches qui modélisent l'incertitude, de nouvelles méthodes ont été proposées par des auteurs comme Rappos et al. [RRCM18]. Ces derniers ont développé un algorithme de map-matching s'inspirant des techniques de **dessin de graphes dirigés par la force**. Ils définissent un système de forces attractives et répulsives exercées par les segments routiers qui attirent les points GPS des trajectoires ou les repoussent. L'intensité de la force d'un segment routier est inversement proportionnelle à la distance le séparant d'un point GPS mais proportionnelle à sa longueur et au cosinus de l'angle qu'il forme avec la trajectoire contenant le point. Quant à Lai et al. [XJJF19], ils introduisent une approche qui combine plusieurs algorithmes classiques de map-matching et exploite des **données historiques de trajectoires** afin d'obtenir un résultat optimal. Dans le détail, leur approche consiste à appliquer au moins deux algorithmes différents de map-matching à chaque trajectoire puis à comparer les résultats. Si des divergences sont observées on lève l'incertitude grâce aux résultats historiques provenant du map-matching de trajectoires qui se sont déroulées dans la même région. La trajectoire déjà traitée qui est la plus similaire à la trajectoire courante sert alors de référence.

Les approches avancées offrent généralement de meilleurs résultats, comparées aux autres méthodes. A titre d'illustration, l'approche par modèle de Markov caché a un taux de succès allant jusqu'à 99,89% pour des trajectoires dont les points sont enregistrées à des fréquences supérieures à une fois toutes les 30 secondes [NK09]. Néanmoins, cette qualité a un coût puisque les approches avancées sont moins rapides et bien plus complexes à mettre en œuvre. En outre, elles sont fonction de plusieurs paramètres qu'il faut savoir finement ajuster pour obtenir un résultat optimal, ce qui peut s'avérer difficile à mettre en œuvre en pratique.

Synthèse :

Le map-matching de trajectoires est une tâche complexe mais qui bénéficie d'un intérêt continu des scientifiques. En témoignent les nombreux articles sur le sujet. Les différentes approches de map-matching qui y sont présentées peuvent se classer généralement en approches locales qui traitent les points des trajectoires indépendamment ou en approches globales qui considèrent l'entièreté des trajectoires. Toutefois, cette classification binaire ne rend pas suffisamment compte de la richesse des méthodes développées pour le map-matching. Une classification plus affinée a été proposée par Quddus et al. [QON07]. Elle regroupe les approches de map-matching en quatre catégories :

- les approches géométriques qui n'exploitent que la proximité entre les trajectoires et les segments routiers ;
- les approches topologiques qui étendent les approches géométriques et les améliorent, en associant le critère de proximité aux propriétés topologiques du réseau routier (c'est-à-dire les connexions entre les segments routiers, leurs courbures, les sens de circulation) ;
- les approches probabilistes qui définissent autour des points des trajectoires, des zones d'erreur afin d'identifier les segments routiers leur correspondant ;
- les approches avancées, plus complexes et basées sur l'utilisation d'outils tels que la logique floue, la théorie des croyances de Dempster-Shafer, les filtres de Kalman, les filtres à particules, le modèle de Markov caché, les champs aléatoires conditionnels, les techniques de dessin de graphes dirigés par la force, etc.

Des quatre catégories, les approches avancées offrent généralement de meilleurs résultats mais sont plus lentes et complexes à mettre en œuvre.

2.3.1.2 Gestion des problématiques de qualité et de volume

Indépendamment de leur degré de sophistication, les approches de map-matching présentées jusqu'ici ont de mauvaises performances (en termes de qualité) lorsque les données traitées sont de faible qualité et/ou que leur volume est important. Or, ces deux facteurs caractérisent bien souvent les jeux de données actuels de trajectoires (grande quantité de données fortement bruitées). D'où le développement d'approches intégrant des stratégies pour la gestion des facteurs de volume et de qualité des trajectoires.

Parallélisation. Pour ce qui concerne les approches avancées, l'incapacité à gérer de grandes masses de trajectoires est essentiellement due à leur forte complexité algorithmique, notamment en termes de temps de calcul. Parmi les solutions proposées pour y remédier, on retrouve la mise en œuvre de techniques dédiées au traitement des données massives (*big data* en anglais), telle que la parallélisation des algorithmes de map-matching existants. C'est l'approche développée par Mattheis et al. [MAZE+14] qui consiste à distribuer les calculs de l'algorithme de map-matching basé sur le HMM grâce aux outils de l'écosystème *Apache*. En l'occurrence, ils s'appuient sur *Apache Kafka* [2] pour gérer et distribuer le flux de données de trajectoires à des nœuds de calcul exécutant *Apache Storm* [3].

Le même procédé est mis en œuvre par Francia et al. [FGV19] à la différence que ces derniers optent pour *Apache Spark* afin de distribuer et d'exécuter les calculs de l'algorithme HMM. Dans la même dynamique, Cho et al. [CC15] proposent une méthode inspirée des techniques dédiées aux données massives mais n'impliquant pas une distribution des calculs en tant que telle. Leur solution consiste plutôt à enregistrer les trajectoires et la carte du réseau routier dans le système distribué de gestion de base de données non-relationnelles *Apache Hbase*, puis à les indexer en utilisant la librairie *Geohash*. Par la suite un algorithme de type point à courbe et intégrant les informations sur la topologie du réseau routier, est appliqué aux données ainsi indexées. Le passage à l'échelle est rendu possible dans ce cas grâce à l'indexation des données. Globalement, les approches visant à distribuer les données et/ou les calculs entre plusieurs nœuds, permettent de gérer des volumes importants de trajectoires avec des méthodes classiques de map-matching. Toutefois elles héritent en contrepartie des limites de ces mêmes méthodes.

2. `https://kafka.apache.org/`
3. `https://storm.apache.org/`

Traitement par lots. Hormis, la parallélisation, une autre solution proposée pour le traitement de grands volumes de trajectoires est le traitement par lots. C'est une approche qui diffère du traitement habituel des trajectoires qui est plutôt séquentiel. Sa particularité est que les opérations entrant en ligne de compte pour le map-matching sont réalisées pour plusieurs trajectoires à la fois. Dans leur article [LHK+13], Li et al. proposent par exemple un algorithme de traitement d'une collection de trajectoires qui dans un premier temps sélectionne, en minimisant une fonction objectif, un ensemble de segments candidats, distincts les uns des autres et proches des points GPS des trajectoires. Les segments sélectionnés sont supposés contenir tous les chemins possibles correspondant aux trajectoires. On effectue donc en une fois l'opération de recherche des segments candidats pour l'ensemble des trajectoires. Par la suite le map-matching se fait pour chaque trajectoire en utilisant l'ensemble des segments candidats préalablement déterminés.

Bian et al.[BCW20] proposent également une approche de traitement par lots mais dans laquelle les trajectoires similaires sont d'abord regroupées en clusters. Une trajectoire représentative de chaque cluster est ensuite extraite en déterminant la moyenne glissante (moyenne calculée sur un sous-ensemble de données délimitées par une fenêtre coulissante) des coordonnées GPS issues des différentes trajectoires du cluster. C'est cette dernière qui subit le processus de map-matching. Le résultat du map-matching obtenu pour chaque trajectoire représentative est ensuite affecté à l'ensemble des trajectoires du cluster.

L'approche de traitement par lots rend bien plus efficace le processus de map-matching des trajectoires sans nécessiter l'utilisation des ressources matérielles et logicielles indispensables au calcul distribué. Cependant, les approches par lots ne sont pleinement efficaces que si les trajectoires sont réparties en groupes de tailles conséquentes et suffisamment homogènes ce qui est rarement le cas en pratique.

Gestion du taux d'échantillonnage et de la précision du GPS. Outre les limitations liées au passage à l'échelle, les performances des méthodes de map-matching sont également impactées par la qualité des trajectoires qui dépend essentiellement de la fréquence d'enregistrement (encore appelée taux d'échantillonnage) et de la précision des points GPS. Ainsi, lorsque le taux d'échantillonnage est trop faible, par exemple supérieur à 30 secondes entre deux enregistrements, cela induit une incertitude dans le processus de map-matching. En effet, plus deux

points GPS sont espacés dans le temps et donc dans l'espace, plus le nombre de chemins possibles entre eux est grand. Cela complexifie la détermination de la bonne séquence de segments routiers permettant de les relier. Le traitement par lots de Bian et al. [BCW20], précédemment décrit, gère l'incertitude liée au faible taux d'échantillonnage en tirant parti de la complémentarité des trajectoires se trouvant dans un même cluster. Ces trajectoires prises individuellement sont imprécises, mais une fois agrégées elles se compensent mutuellement. Il en résulte un tracé plus précis et mieux échantillonné permettant d'obtenir un map-matching plus qualitatif.

Toujours dans l'optique de pallier l'incertitude inhérente aux trajectoires à faible taux d'échantillonnage, Yin et al. [YSWZ18] proposent un algorithme de map-matching basé sur un modèle probabiliste combinant plusieurs facteurs. Plus spécifiquement, ils associent le coût des choix de conduite des utilisateurs aux caractéristiques topologiques du réseau routier et à la proximité entre les trajectoires et le réseau. Le paramètre de coût des choix de conduite amène l'algorithme de map-matching à privilégier les segments routiers proches des points GPS et à suivre des routes aussi courtes et rectilignes que possible. Une approche de map-matching par utilisation d'un modèle probabiliste est également proposée par Millard et al. [MBHW19] mais pour traiter des trajectoires contenant des données GPS de faible qualité. Il s'agit de trajectoires enregistrées pendant des épisodes de fortes perturbations voire de coupures des signaux GPS dues à la densité de l'environnement urbain, parsemé d'obstacles tels que des immeubles ou aux passages et parkings souterrains. Le modèle défini dans ce cas intègre un critère temporel fonction des limitations de vitesse sur les segments routiers. Ce critère permet de pénaliser l'apparition de demi-tours qui sont indicateurs d'imprécisions lors du processus de map-matching. Il est combiné aux caractéristiques topologiques du réseau routier et au degré de proximité entre les points GPS et les segments routiers dans le modèle proposé.

2.3.2 Le post-matching

Le post-matching est l'ensemble des actions qui interviennent en aval du map-matching pour en améliorer et/ou évaluer les résultats. A notre connaissance, peu de travaux traitent explicitement du post-matching. Parmi les auteurs ayant abordé le sujet, nous pouvons citer Kruger et al. [KSBE18] qui ont développé un outil visuel et interactif pour la calibration des paramètres de map-matching, le

filtrage des trajectoires et l'édition du réseau routier sous-jacent. De plus, l'outil affiche des statistiques sur la qualité du map-matching. Le retour visuel permet à l'utilisateur de cerner et d'ajuster en temps réel le comportement de l'algorithme de map-matching. Yang et Gidofalvi [YG18] ont également développé un outil interactif pour visualiser les trajectoires, calibrer les paramètres de map-matching et afficher des statistiques sur le processus de map-matching. Leur outil présente l'avantage d'être libre de droits et permet de dessiner à la volée des trajectoires sur la carte. En outre, il supporte plusieurs algorithmes de map-matching. Pour leur part, Blazquez et al. [BRML18] proposent un outil non-visuel de calibration, des paramètres de map-matching. Ce dernier permet de déterminer par la combinaison d'une série d'analyses statistiques et l'utilisation d'un système de logique floue, l'impact des variations des valeurs des paramètres de l'algorithme utilisé sur la qualité du map-matching.

2.4 Distances et similarités entre trajectoires

L'analyse des trajectoires, dont traite la présente thèse, se fonde essentiellement sur leur comparaison. L'objectif étant de rechercher et/ou de regrouper des trajectoires similaires d'après un certain nombre de critères prédéfinis. Cela implique d'évaluer quantitativement le degré de ressemblance ou de différence des trajectoires. Ainsi, une panoplie de mesures de comparaison, aussi bien des distances que des similarités, ont été proposées dans la littérature. Nous en présentons ici quelques-unes, réparties en deux grandes familles : les mesures en espace libre et celles en espace contraint. Le choix des distances et similarités est fait de sorte à couvrir au mieux, mais succinctement, les différents types de mesures existantes.

Avant de présenter ces mesures de distance et de similarité, détaillons préalablement quelques caractéristiques attendues d'une bonne mesure :

❶ La première concerne la **robustesse** aux bruits et aux légères perturbations. D'après ce critère, les distances devraient pouvoir identifier comme similaires deux trajectoires proches et ayant sensiblement la même forme générale, même si elles ne coïncident pas en partie ou totalement.

❷ La seconde est relative à la **complexité**. Les algorithmes permettant d'évaluer les distances devraient générer un coût limité en termes de calculs.

❸ La troisième à trait au **nombre de paramètres** dont dépend la mesure. Ce dernier doit être aussi réduit que possible si ce n'est nul.

❹ La dernière est associée au **caractère métrique** de la distance. Formellement, une distance métrique est une fonction positive $d : E \times E \to [0, +\infty[$ qui respecte les propriétés suivantes pour tout triplet $r, s, t \in E$:

(a) $d(r, s) = 0 \iff r = s$ (principe d'identité des indiscernables)

(b) $d(r, s) = d(s, r)$ (symétrie)

(c) $d(r, s) \leq d(r, t) + d(t, s)$ (inégalité triangulaire)

De ces trois propriétés, l'inégalité triangulaire est certainement la plus intéressante car elle garantit que la distance discrimine correctement les différents éléments comparés. De plus elle permet d'exploiter différentes techniques de filtrage qui accélèrent les comparaisons lors de la recherche de trajectoires similaires.

2.4.1 Distances pour trajectoires en espace libre

2.4.1.1 Distances simples

Pour comparer deux trajectoires en espace libre, l'approche la plus intuitive consiste à calculer la moyenne des distances entre les points de même index des deux trajectoires, comme illustré à la figure 2.5. Cette stratégie qui consiste à créer des paires entre les points de deux trajectoires afin d'évaluer la distance séparant ces dernières, s'appelle l'appariement. Le plus souvent on utilise la distance euclidienne, aussi connue sous le nom de *norme* L_2 [SW80], [FRM94]. C'est une distance qui évalue la longueur du segment reliant deux points du plan ou de l'espace. Toutefois, il est également possible d'utiliser d'autres distances comme la *norme* L_1, encore connue sous le nom de distance de Manhattan, ou la distance du grand cercle qui permet de connaître la distance séparant deux points le long d'une sphère à partir de leurs latitudes et longitudes respectives. Nous ne présenterons ici que la distance euclidienne entre trajectoires.

Distance euclidienne. La distance Euclidienne entre deux trajectoires se définit formellement comme suit :

$$d_{\text{Euclidienne1}}\,(T_1, T_2) = \frac{\sum_{i=1}^{n} d(p_i^1, p_i^2)}{n} \tag{2.5}$$

avec $d(p_i^1, p_i^2)$ la distance Euclidienne entre les points p_i^1 et p_i^2. Les deux trajectoires sont supposées avoir la même taille n.

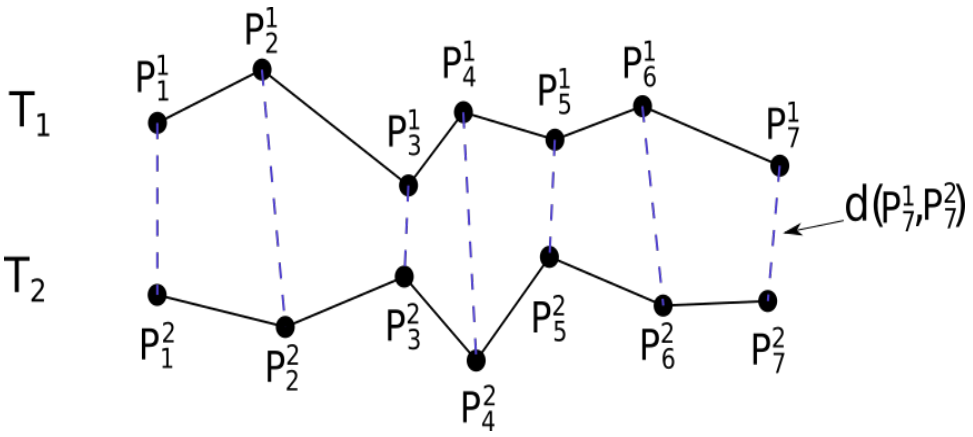


FIGURE 2.5 – Illustration de l'appariement des points avec la distance Euclidienne.

La distance Euclidienne a la particularité d'être métrique, sans paramètre et pouvant se calculer rapidement en $\mathcal{O}(n)$, n étant la longueur de chacune des deux trajectoires. Toutefois, elle est très sensible aux bruits induits par les erreurs de mesures inhérentes à l'utilisation du GPS et qui affectent la précision des coordonnées des points des trajectoires. En outre, la définition précédente ne permet pas de comparer des trajectoires de longueurs différentes puisqu'elle n'apparie que des points de même index. Il est cependant possible d'outrepasser cette limitation en utilisant une fenêtre glissante ayant la taille de la plus petite des deux trajectoires. On retient alors à chaque déplacement de la fenêtre tous les points de la plus petite des trajectoires et autant de points dans la grande. Ensuite, le calcul de la distance entre les points sélectionnés est effectué avec la formule précédemment définie. Au terme du processus, la distance entre les deux trajectoires de tailles différentes est la plus petite des distances calculées à chaque déplacement de la fenêtre glissante. La complexité passe alors à $\mathcal{O}(nm)$ et la formule devient [SLZ+20] :

$$d_{\text{Euclidienne2}}\left(T_1, T_2\right) = \min_{j=0}^{m-n} \frac{\sum_{i=1}^{n} d(p_i^1, p_{j+i}^2)}{n} \tag{2.6}$$

avec n et m les longueurs des trajectoires T_1 et T_2 et en considérant que la trajectoire T_2 est la plus longue, $n \leq m$.

Notons par ailleurs que la distance Euclidienne ne tient pas compte des décalages temporels locaux, plus connus sous l'anglicisme *local time shifting* qui sont fréquemment observés dans les données réelles de trajectoires. Ils correspondent à la présence de sous-séquences de formes similaires, mais décalées l'une par rapport

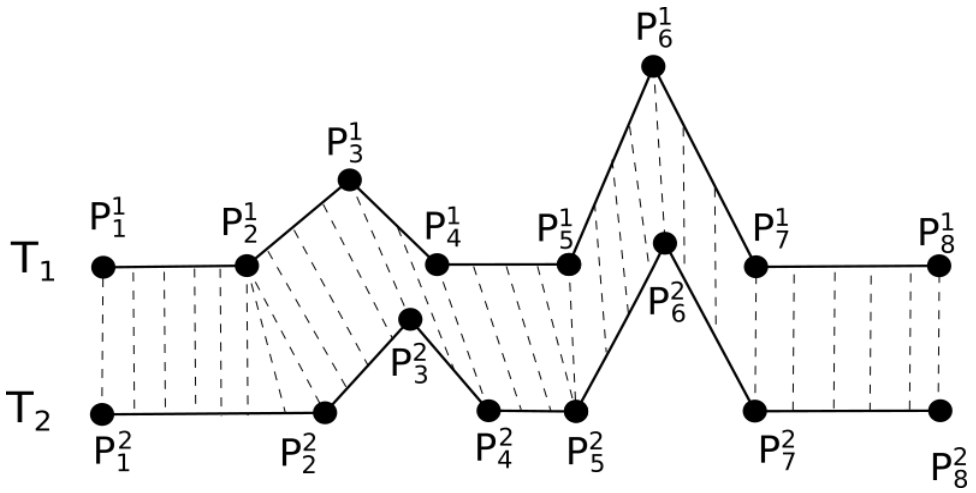


FIGURE 2.6 – Illustration décalage temporel local.

à l'autre, entre deux trajectoires. On parle de sous-séquences parce que l'ordre des points est conservé même si leurs index diffèrent d'une trajectoire à l'autre. Le décalage temporel local s'oppose au décalage temporel global où deux trajectoires, entièrement similaires, sont déphasées. Ici, le caractère local fait référence au fait que seules des portions de trajectoires se ressemblent. La figure 2.6 illustre un décalage temporel local.

2.4.1.2 Distances intégrant les distorsions

Afin de prendre en compte les décalages temporels locaux, des mesures de similarité et de distance, initialement conçues pour les séries temporelles et les chaînes de caractères, ont été adaptées aux trajectoires. Ces mesures dites de distorsions (*warping* en anglais) visent à apparier de façon optimale les points des trajectoires quelle que soit leur taille respective. Elles sont définies par le biais de formules de récurrence que l'on résout efficacement en utilisant la programmation dynamique [Bel66]. Il s'agit d'une méthode de programmation développée pour résoudre les problèmes d'optimisation, en les décomposant en sous-problèmes interdépendants et de moins en moins complexes. Les solutions optimales des sous-problèmes sont stockées et réutilisées au fur et à mesure pour constituer les résultats intermédiaires permettant de trouver la solution finale. Les algorithmes utilisant cette méthode de programmation pour calculer les distances entre trajectoires induisent une complexité d'ordre $\mathcal{O}(nm)$, avec n et m les tailles respectives des trajectoires

comparées. Nous présentons ici quatre mesures intégrant les distorsions à savoir : *LCSS*, *DTW*, *EDR* et *ERP*. Les formules présentées considèrent donc deux trajectoires T_1 et T_2 de tailles respectives n et m.

LCSS (*Longest Common Subsequence*) [KH90], est une mesure de similarité non métrique qui compare deux trajectoires en déterminant la taille de leur plus longue sous-séquence commune. Les points constituant cette sous-séquence ne doivent pas nécessairement occuper des positions consécutives mais doivent apparaître dans le même ordre au sein des deux trajectoires. La particularité de *LCSS* est d'apparier deux points, même s'ils ne coïncident pas, tant que la distance entre eux est inférieure à un seuil ϵ, comme l'illustre la figure 2.7. Cela rend *LCSS* robuste au bruit, mais l'utilisation d'un seuil a pour inconvénient la difficulté d'en définir la valeur optimale. De plus, *LCSS* est une mesure d'appariement partielle. Ce qui signifie qu'elle est uniquement fonction du nombre de paires de points appariés alors que la prise en compte des informations provenant de points non appariés est tout aussi importante pour obtenir une comparaison plus précise des trajectoires.

$$LCSS(T_1, T_2) = \begin{cases} 0, \ Si\ n = 0\ ou\ m = 0 \\ 1 + LCSS(Rest(T_1),\ Rest(T_2)),\ si\ d(Head(T_1),\ Head(T_2)) \leq \epsilon, \\ max(LCSS(Rest(T_1),\ T_2),\ LCSS(T_1,\ Rest(T_2))),\ Sinon \end{cases} \tag{2.7}$$

Avec ϵ la valeur du seuil d'appariement. Head() fait référence au premier point de la trajectoire et Rest() à la partie restante de cette dernière lorsque l'on enlève le premier point. Nous conserverons cette notation dans la suite du document. Pour obtenir une distance à partir de la similarité *LCSS*, on peut appliquer la formule 2.8 :

$$d_{LCSS}(T_1, T_2) = 1 - \frac{LCSS(T_1, T_2)}{min(n, m)} \tag{2.8}$$

DTW (*Dynamic Time Warping*) [BC96], est une mesure d'appariement complet qui ne dépend d'aucun paramètre, contrairement à *LCSS*. Elle permet un étirement ou un rétrécissement des trajectoires, dans le temps, en répliquant des points précédemment traités lorsqu'un nouveau point ne peut être apparié (cf figure 2.8). Cela garantit un appariement, à la fois, optimal et complet entre tous les points

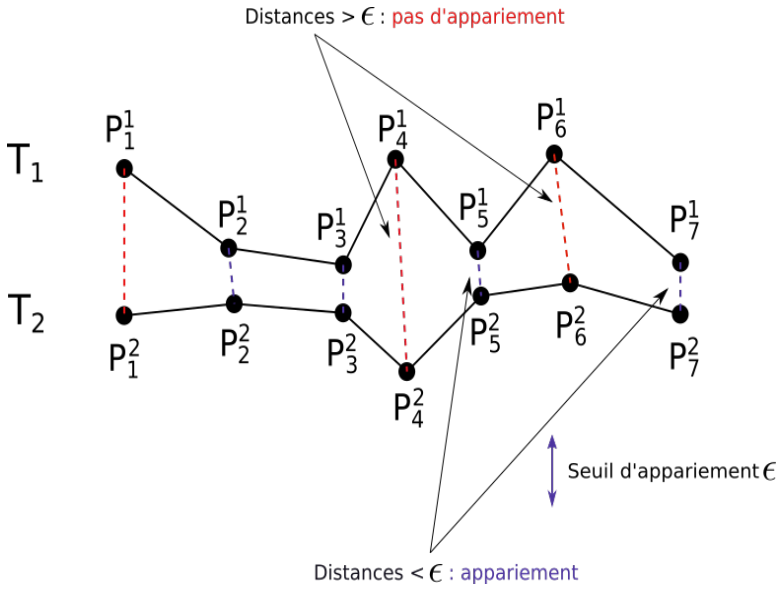


FIGURE 2.7 – Illustration de l'appariement des points avec les distances *LCSS* et *EDR*.

d'un couple de trajectoires. *DTW* n'est ni métrique ni robuste aux bruits. Elle se formule comme suit :

$$DTW(T_1, T_2) = \begin{cases} 0, \ Si\ n = 0\ et\ m = 0 \\ \infty, \ Si\ n = 0\ ou\ m = 0 \\ d(Head(T_1), Head(T_2)) + min \begin{cases} DTW(T_1, Rest(T_2)) \\ DTW(Rest(T_1), T_2) \\ DTW(Rest(T_1), Rest(T_2)) \end{cases} & Sinon \end{cases} \tag{2.9}$$

EDR (*Edit Distance on Real sequence*) [CÖO05], est une distance d'édition. L'idée clé des distances d'édition est d'évaluer le coût total des opérations nécessaires pour transformer une trajectoire en une autre, à savoir : l'insertion, la suppression et la substitution. La substitution correspond à un appariement, tandis que l'insertion et la suppression se produisent lorsqu'un d'appariement n'est pas possible, ce que l'on appelle aussi un gap (un trou dans une des deux trajectoires qui nécessite

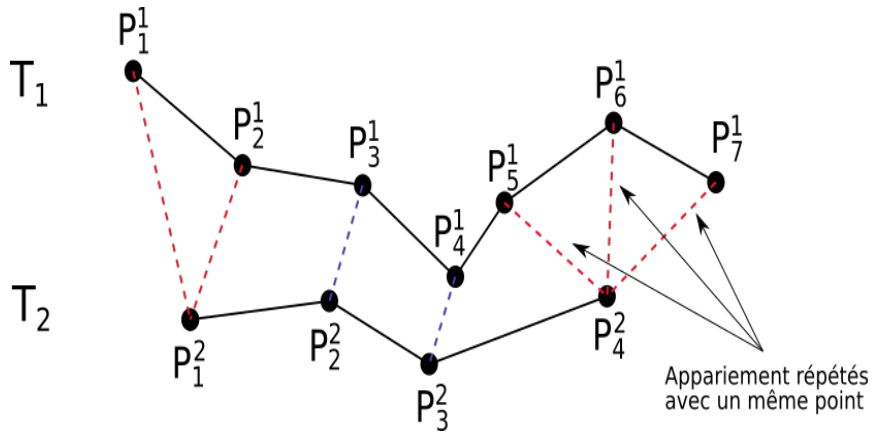


FIGURE 2.8 – Illustration de l'appariement des points avec la distance *DTW*.

d'ajouter, ou de supprimer de manière symétrique, un point dans l'autre trajectoire). Les opérations d'édition engendrent des coûts et la distance d'édition est la somme de ces coûts. *EDR* définit une valeur fixe, généralement 1, pour chaque opération d'édition. Ainsi, calculer la distance *EDR* équivaut à déterminer la distance minimale d'édition entre deux trajectoires. Comme *LCSS*, *EDR* apparie les points en fonction d'un seuil (cf figure 2.7). Par conséquent, elle est également robuste aux bruits, mais non métrique et sensible à la valeur du seuil. L'équation 2.10 permet d'évaluer la distance *EDR* entre deux trajectoires.

$$EDR(T_1, T_2) = \begin{cases} n, \ Si\ m = 0 \\ m, \ Si\ n = 0 \\ min\{EDR(Rest(T_1), Rest(T_2)) + subcost, & Sinon \\ EDR(Rest(T_1), T_2) + 1, EDR(T_1, Rest(T_2)) + 1\} \end{cases} \tag{2.10}$$

avec,

$$subcost = \begin{cases} 0, \ Si\ d(Head(T_1), Head(T_2)) \leq \epsilon \\ 1, \ Sinon \end{cases} \tag{2.11}$$

ERP (*Edit distance with Real Penalty*). Dans l'optique de tirer le meilleur des distances *DTW* et *EDR* mais aussi de les améliorer, Chen et al. [CN04] ont proposé la distance *ERP*. Celle-ci calcule les distances réelles entre les paires de points, en

lieu et place d'une pénalité fixe, et n'utilise pas de condition de seuil pour l'appariement. En cas de substitution, le coût est égal à la distance entre la paire de points appariés. En cas de gap, donc d'absence d'appariement, le coût considéré est la distance entre le point ne pouvant être apparié et un point de référence fixé arbitrairement (cf figure 2.9). Ces ajustements, et en particulier l'utilisation d'un point de référence pour gérer les cas de non-appariement, permettent à *ERP* d'être une mesure de distance métrique. Le point de référence utilisé ne doit pas varier afin de garantir la propriété métrique. En pratique ses coordonnées sont fixées à $(0, 0)$ pour les trajectoires en deux dimensions. *ERP* est invariante aux transformations affines parce que les coordonnées des points formant les trajectoires sont normalisées avant de l'appliquer. Elle est cependant sensible aux bruits car elle est évaluée en fonction des distances réelles (la $normeL_1$) entre les points. Elle cumule ainsi les effets des imprécisions liées au bruit. Sa formule s'écrit :

$$ERP(T_1, T_2) = \begin{cases} \sum_l^n d(p_i^1, g), \ \textit{Si } m = 0 \\ \sum_l^m d(p_i^2, g), \ \textit{Si } n = 0 \\ min \begin{cases} ERP(Rest(T_1), Rest(T_2)) + d(Head(T_1), Head(T_2)) \\ ERP(Rest(T_1), T_2) + d(Head(T_1), g) \\ ERP(T_1, Rest(T_2)) + d(Head(T_2), g) \end{cases} & \textit{Sinon} \end{cases} \tag{2.12}$$

Avec g le point de référence en cas de gap.

2.4.1.3 Distances basées sur la forme

Les mesures présentées précédemment traitent les trajectoires comme des séquences discrètes, en ignorant leur forme. En réalité, et bien qu'elles soient habituellement modélisées comme des séquences de points, les trajectoires sont également constituées de segments qui relient ces points et négliger cette dimension structurelle pourrait biaiser les mesures de distance entre trajectoires. Afin d'intégrer l'information apportée par la forme dans la comparaison des trajectoires, des mesures telles que la distance de *Hausdorff* et la distance *SSPD*, ont été proposées. Les formules présentées ci-après considèrent deux trajectoires T_1 et T_2 de tailles respectives n et m.

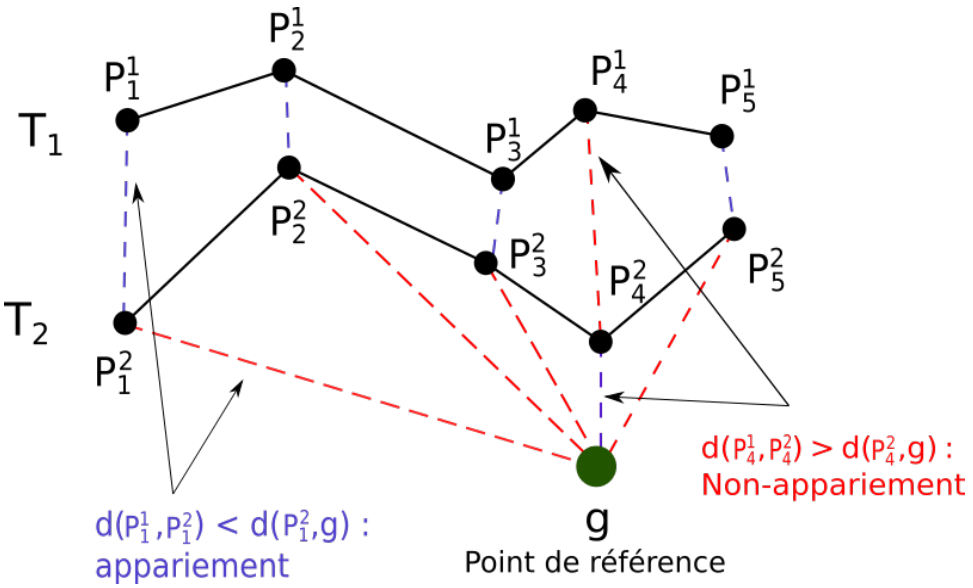


FIGURE 2.9 – Illustration de l'appariement des points avec la distance *ERP*.

Distance de Hausdorff. La distance de *Hausdorff* [Hau14] traite les trajectoires comme des ensembles de points non-ordonnés. Elle est conçue pour déterminer le degré d'éloignement mutuel de deux ensembles. Elle renvoie la distance euclidienne maximale qui sépare un point quelconque d'une trajectoire, du point qui lui est le plus proche dans l'autre trajectoire. Elle est sans paramètre, se calcule par programmation dynamique avec une complexité $\mathcal{O}(nm)$, est robuste aux bruits, métrique et capable de gérer des trajectoires de différentes tailles. Dans sa formulation originale la distance de *Hausdorff* se définit à partir des distances point à segment [Gui16] mais en procédant à des simplifications on aboutit à l'écriture suivante :

$$d_{\text{Hausdorff}}(T_1, T_2) = max \begin{cases} max_{p_i^1 \in T_1} min_{p_j^2 \in T_2} d(p_i^1, p_j^2), \\ max_{p_i^2 \in T_2} min_{p_j^1 \in T_1} d(p_i^2, p_j^1) \end{cases} \tag{2.13}$$

La distance point à segment est la plus courte distance séparant un point d'un segment. Elle est le plus souvent plus résistante aux bruits que la distance point à point car elle ne dépend en réalité que d'un seul des points du segment le plus proche.

SSPD (*Symmetrized Segment-Path Distance*) [BGLR16], est une distance basée

sur la forme qui utilise également la distance point à segment. Elle mesure la moyenne des distances minimales entre chaque point d'une trajectoire et tous les segments d'une deuxième trajectoire puis vice versa. Elle est robuste aux bruits, peut comparer des trajectoires de tailles différentes et n'est fonction d'aucun paramètre. Le coût en termes de calculs pour l'évaluer est de l'ordre $\mathcal{O}(nm)$. Sa formule est :

$$SSPD(T_1, T_2) = \frac{SPD(T_1, T_2) + SPD(T_2, T_1)}{2} \tag{2.14}$$

où

$$SPD(T_1, T_2) = \frac{1}{n} \sum_{i=1}^{n} D_{pt}(p_i^1, T_2) \tag{2.15}$$

et

$$D_{pt}(p_i^1, T_2) = min_{j \in [1,\ldots,m]} D_{ps}(p_i^1, s_j^2) \tag{2.16}$$

Avec $SPD(T_1, T_2)$ la moyenne des distances minimales entre chaque point de T_1 et l'ensemble des segments de T_2. $D_{pt}(p_i^1, T_2)$ la distance minimale entre le point p_i^1 et l'ensemble des segments de la trajectoire T_2. $D_{ps}(p_i^1, s_j^2)$ la distance du point p_i^1 au segment s_j^2.

$$D_{ps}(p_i^1, s_j^2) = \begin{cases} d(p_i^1, p_{i,proj}^1), \ \ Si\ p_{i,proj}^1 \in s_j^2 \\ min(d(p_i^1, s_j^2.p_1), d(p_i^1, s_j^2.p_2)), \ \ Sinon \end{cases} \tag{2.17}$$

où $p_{i,proj}^1$ est le projeté orthogonal de p_i^1 sur le segment s_j^2 tandis que $s_j^2.p_1$ et $s_j^2.p_2$ sont les deux extrémités du segment s_j^2.

Synthèse :

Nous avons proposé dans cette section une classification en trois catégories des mesures de distance pour trajectoires en espace libre, à savoir :

— **les mesures simples** qui rassemblent les *normes* L_p dont la distance euclidienne ou encore la distance du grand cercle. Elles sont rapides à calculer du fait leur complexité linéaire. Elles s'appliquent sur des trajectoires de même longueur et équivalent à la moyenne des distances entre les points de même index. Leur limite commune est de ne pas gérer les décalages temporels locaux ;

— **les mesures intégrant les distorsions**. Ce sont des distances telles que *LCSS*, *DTW*, *EDR* et *ERP* initialement conçues pour les séries temporelles et les chaînes de caractères. Elles ont pour particularité d'apparier de façon optimale les points de deux trajectoires en tenant compte des décalages temporels locaux. Elles sont également définies à partir des distances point à point et se calculent en utilisant la programmation dynamique. Cette méthode de calcul leur confère une complexité quadratique ;

— **les mesures basées sur la forme**. Nous avons retenu dans ce groupe, les distances de *Hausdorff* et *SSPD*. Le propre de ces deux mesures est d'intégrer la structure des trajectoires grâce à l'utilisation des distances point à segment. Elles se calculent également avec des algorithmiques de programmation dynamique avec une complexité quadratique et peuvent s'appliquer à des trajectoires de longueurs différentes.

Nous aurions pu rajouter une quatrième catégorie, celle des mesures dites spatio-temporelles qui intègrent explicitement la dimension temporelle inhérente à chaque trajectoire dans leur définition. Cependant elles sont pour la plupart des extensions de mesures de distance purement spatiales déjà présentées. Il s'agit de : *STLCSS* (*LCSS Spatio-Temporal*) [VKG02], STED (*Spatio-Temporal Euclidean Distance*) [NP06]. Pour plus d'informations sur les distances pour trajectoires en espace libre, nous suggérons les articles de revues [SLZ+20] et [BGLR16].

Le tableau 2.1 résume les principales propriétés des distances pour trajectoires en espace libre, présentées précédemment. On remarque qu'aucune distance ne satisfait à la fois l'ensemble des propriétés énumérées.

TABLE 2.1 – Tableau récapitulatif des caractéristiques des distances en espace libre

Mesures	**Complexité**	**Métrique**	**Sans Paramètres**	**Bruit**	**Décalages temporels locaux**
Distance Euclidienne	$\mathcal{O}(n)$	✓	✓	✗	✗
Hausdorff	$\mathcal{O}(nm)$	✓	✓	✗	✗
SSPD	$\mathcal{O}(nm)$	✗	✓	✓	✗
LCSS	$\mathcal{O}(nm)$	✗	✗	✓	✓
DTW	$\mathcal{O}(nm)$	✗	✓	✗	✓
EDR	$\mathcal{O}(nm)$	✗	✗	✓	✓
ERP	$\mathcal{O}(nm)$	✓	✗	✗	✓

2.4.2 Distances pour trajectoires contraintes par un réseau

Les distances présentées précédemment ont été conçues pour les trajectoires se déroulant dans un espace libre telles que par exemple les trajectoires d'animaux dans la nature. Cependant, les trajectoires sont plutôt couramment collectées sur des objets mobiles tels que des vélos, des voitures, des trains ou des bus se déplaçant sur un réseau (routes, chemins de fer) et donc dans un espace contraint. La prise en compte des contraintes liées aux déplacements sur un réseau a conduit à l'émergence d'une nouvelle classe de mesures de similarités et de distances pour trajectoires. Nous consacrons la présente section à une présentation de quelques-unes de ces mesures dédiées aux trajectoires contraintes par un réseau. Pour la plupart, elles présentent de fortes similitudes avec les distances en espace libre quand elles ne découlent tout simplement pas de ces dernières. Généralement, le passage de l'espace libre à l'espace contraint correspond à un remplacement de la distance euclidienne par la distance réseau. Pour rappel, la distance réseau entre des nœuds correspond à la longueur du plus court chemin les séparant. Dans la suite, nous désignerons la distance réseau entre deux nœuds v_i et v_j par $d_G(v_i, v_j)$. Rappelons que la complexité associée aux algorithmes standards de recherche du plus court est de l'ordre $\mathcal{O}(|V| + |E|)$, $|V|$ et $|E|$ sont respectivement le nombre de sommets et le nombre d'arêtes que contient le réseau. Sauf indication contraire, les distances présentées comparent deux trajectoires T_1 et T_2 de tailles respectives n et m.

2.4.2.1 Distances simples

$\mathbf{SIM_{POI}}$ est, à notre connaissance, l'une des premières mesures intégrant la structure du réseau routier dans sa définition. Elle a été proposée en 2005 par Hwang et al. [HKL05]. Elle compare les trajectoires en fonction d'un ensemble de points d'intérêt (POI) qui peuvent être des intersections de routes ou des lieux. Deux trajectoires sont considérées comme similaires dès lors qu'elles contiennent chacune l'ensemble des POI prédéfinis, dans le cas contraire elles ne sont pas similaires. Ainsi, cette mesure de distance, qui n'est pas métrique, ne peut prendre que deux valeurs 0 ou 1 ce qui est une forte limitation car les mesures de similarité et de distance sont censées prendre des valeurs continues ou discrètes suffisamment variées afin de comparer les trajectoires avec une bonne résolution. De plus, les POI doivent être prédéfinis par des utilisateurs qui peuvent ne pas être des experts ou

ne pas avoir une connaissance approfondie du réseau routier. La complexité des calculs associés à la mesure SIM_{POI} est de l'ordre $\mathcal{O}(|P|)$, avec P l'ensemble des POI et son expression mathématique est :

$$SIM_{POI}(T_1, T_2, P) = \begin{cases} 1, & Si\ \forall p \in P,\ p\ est\ contenu\ dans\ T_1\ et\ dans\ T_2 \\ 0, & Sinon \end{cases} \tag{2.18}$$

D_{NET} s'inspire de la distance euclidienne pour les trajectoires en espace libre. Elle a été définie par Tiakas et al. [TPN+09] comme une distance nœud à nœud pour les trajectoires contraintes par un réseau. Elle correspond à la moyenne des longueurs des plus courts chemins entre les nœuds de même index. Tout comme son pendant en espace libre, elle est métrique, mais ne peut pas gérer les décalages temporels locaux ou les trajectoires de différentes longueurs. Le calcul de la distance D_{NET} a une complexité d'ordre $\mathcal{O}(n(|V| + |E|))$, n est la longueur de chacune des deux trajectoires, c'est à dire le nombre de nœuds les constituant, $|V|$ et $|E|$ sont respectivement le nombre de sommets et le nombre d'arêtes que contient le réseau supportant les trajectoires. D_{net} se formule ainsi qu'il suit :

$$D_{NET}(T_1, T_2) = \frac{1}{n} \sum_{i=1}^{n} d_N(v_i, v_j) \tag{2.19}$$

Avec $d_N(v_i, v_j) = \frac{d_G(v_i,v_j)+d_G(v_j,v_i)}{2D_G}$ la distance réseau normalisée entre les nœuds v_i et v_j. Notons que $v_i \in T_1$, $v_j \in T_2$. D_G est le diamètre du réseau.

2.4.2.2 Distances de type nœud à trajectoire

Les mesures de ce type évaluent la distance minimale séparant chaque nœud d'une première trajectoire et l'ensemble des nœuds de la seconde.

$K - PC$ est une mesure proposée par Evans et al. [EOSH12] dans le but de déterminer efficacement les K principaux corridors (K-PC) d'un réseau routier. Ici, un corridor est une trajectoire représentative d'un groupe de trajectoires (elle est plus proche de toutes les autres). La distance K-PC entre deux trajectoires est ainsi égale à la moyenne des distances réseau minimales entre chaque nœud de la première trajectoire et tous les nœuds de la seconde. Cela garantit que, quelle que soit

la taille des trajectoires, il est toujours possible de calculer la distance les séparant. *K-PC* n'est pas métrique, et ne peut pas gérer les décalages temporels locaux. La complexité associée au calcul de la distance $K-PC$ est $\mathcal{O}((n*m)(|V|+|E|))$. Sa formule est :

$$K-PC(T_1,T_2)=\frac{1}{n}\sum_{v_i\in T_1}\min_{v_j\in T_2} d_G(v_i,v_j) \tag{2.20}$$

STLC est une mesure de type nœud à trajectoire, proposée par [SCW+17]. Elle est obtenue en convertissant la distance réseau en similarité grâce à une fonction exponentielle. Comme *K-PC*, elle n'est pas métrique, et ne peut gérer les décalages temporels locaux mais peut traiter des trajectoires de tailles différentes. Le coût des calculs nécessaires à son évaluation est d'ordre $\mathcal{O}((n*m)(|V|+|E|))$. $STLC$ se définit formellement comme suit :

$$STLC(T_1,T_2)=\frac{\sum_{v_i\in T_1} e^{-d_G(v_i,T_2)}}{n}+\frac{\sum_{v_j\in T_2} e^{-d_G(v_j,T_1)}}{m} \tag{2.21}$$

Avec $d_G(v_i,T_2)=\min_{v_j\in T_2} d_G(v_i,v_j)$, la distance réseau minimale entre le nœud v_i et la trajectoire T_2.

Hausdorff$_G$ a été proposée par Roh et al. [RH10] et est une adaptation de la distance de *Hausdorff* aux trajectoires contraintes par un réseau. Le principe est de remplacer la distance euclidienne par la distance réseau. Ici aussi, l'approche nœud à trajectoire est de mise, puisqu'on évalue les distances minimales entre les nœuds d'une trajectoire et tous les nœuds de l'autre. La version graphe de la distance de *Hausdorff* est aussi métrique. Elle peut traiter des trajectoires de différentes tailles mais ne gère pas les décalages temporels locaux. Sa complexité est $\mathcal{O}((n*m)(|V|+|E|))$ et sa formule :

$$HAUSDORFF_G(T_1,T_2)=max\{TRAJ_{dist}(T_1,T_2),TRAJ_{dist}(T_2,T_1)\} \tag{2.22}$$

On prend le maximum de la distance entre trajectoires $TRAJ_{dist}$, correspondant à la formule 2.23, pour que la mesure obtenue soit symétrique.

$$TRAJ_{dist}(T_1,T_2)=max_{e_i\in T_1} min_{e_j\in T_2} SEG_{dist}(e_i,e_j) \tag{2.23}$$

Avec e_i et e_j les segments respectifs des trajectoires T_1 et T_2.

$$SEG_{dist}(e_i, e_j) = max \begin{cases} min\{d_G(e_{i,d}, e_{j,d}), d_G(e_{i,d}, e_{j,f})\}, \\ min\{d_G(e_{i,f}, e_{j,d}), d_G(e_{i,f}, e_{j,f})\} \end{cases} \tag{2.24}$$

Les indices d et f indiquent respectivement le début et la fin d'un segment et correspondent chacun à un nœud du réseau.

2.4.2.3 Distances ensemblistes

Nous introduisons, à présent, quelques mesures ensemblistes pour les trajectoires contraintes par un réseau. Ces mesures sont utilisées pour des applications, telles que le covoiturage ou l'analyse des flux de mobilité, où des trajectoires similaires sont celles qui partagent des segments de route. Les deux mesures que nous présentons sont basées sur les intersections entre trajectoires. Il s'agit de *SIM_ TRAJ* et *EBD*. Elles peuvent s'appliquer à des trajectoires de tailles différentes mais ne prennent pas en compte les décalages temporels locaux. Elles ont toutes deux une complexité quasi-linéaire $\mathcal{O}(nlogn)$ et sont par conséquent rapides à calculer mais seule *EBD* est métrique.

SIM_TRAJ proposée par [KPZF08] est une similarité qui est égale au rapport entre la longueur de la partie commune de deux trajectoires et la longueur totale d'une d'elle, prise comme référence.

$$SIM\_TRAJ(T_1, T_2) = \frac{L_s(T_1 \cap T_2)}{L_s(T_1)} \tag{2.25}$$

L_s évalue la somme des longueurs des segments. T_1 est dans ce cas la trajectoire fixée comme référence.

EBD (*Edge-Based Distance*) [WBC+19], équivaut à la longueur de la plus longue trajectoire moins la longueur de l'intersection des deux trajectoires.

$$EBD(T_1, T_2) = max(L_s(T_1), L_s(T_2)) - L_s(T_1 \cap T_2) \tag{2.26}$$

2.4.2.4 Distances intégrant les distorsions

Comme pour les trajectoires en espace libre, les mesures intégrant les distorsions sont également utilisées pour comparer les trajectoires contraintes par un réseau. Elles sont principalement des adaptations des distances *LCSS* [WBC+18, YL19], *DTW* [SLB18], *ERP* et *EDR* [KXI20], destinées aux trajectoires contraintes par un réseau. Ces adaptations consistent essentiellement à remplacer la distance euclidienne par la distance réseau. Elles sont également évaluées en usant d'algorithmes basés sur la programmation dynamique dont la complexité est d'ordre $\mathcal{O}((n * m)(|V| + |E|))$. Les formules des mesures intégrant les distorsions ayant déjà été présentées, nous ne les reprenons pas ici hormis les cas particuliers de la mesure LORS et de la version contrainte par un réseau de *ERP*.

LORS (Longest Overlapping Road Segments), proposée par Wang et al. [WBC+18], est une variante de la mesure *LCSS* adaptée aux trajectoires contraintes par un réseau. Toutefois *LORS* apparie plutôt des segments et n'utilise pas de valeur de seuil. Ainsi, deux segments ne sont considérés appariés que s'ils sont identiques.

$$LORS(T_1, T_2) = \begin{cases} 0, \ Si\ T_1\ ou\ T_2\ est\ vide, \\ |e_{1,1}| + LORS(Rest(T_1), Rest(T_2)), \ Si\ e_{1,1} = e_{2,1} \\ max(LORS(Rest(T_1), T_2), LORS(T_1, Rest(T_2))), \ Sinon \end{cases} \tag{2.27}$$

Avec $|e_{1,1}|$ la longueur du segment $e_{1,1}$.

ERP$_\mathbf{G}$ comme indiqué précédemment, *ERP* use d'un point de référence arbitraire dont les coordonnées sont habituellement fixées à $(0, 0)$ pour gérer les cas de non-appariement. Cependant, dans un réseau, on ne saurait procéder pareillement au risque de se retrouver dans l'incapacité de pouvoir calculer la distance réseau entre un nœud donné et le point de référence, au cas où celui-ci ne coïnciderait pas avec un des nœuds du réseau. Il faut alors choisir un des nœuds du réseau comme point

de référence en cas de non-appariement. La formule de *ERP* devient alors :

$$ERP_G(T_1,T_2) = \begin{cases} \sum_l^n d_G(v_i, v_g), \ \ Si\ m = 0 \\ \sum_l^m d_G(v_j, v_g), \ \ Si\ n = 0 \\ min \begin{cases} ERP(Rest(T_1), Rest(T_2)) + d_G(Head(T_1), Head(T_2)) \\ ERP(Rest(T_1), T_2) + d_G(Head(T_1), v_g) \\ ERP(T_1, Rest(T_2)) + d_G(Head(T_2), v_g) \end{cases} & Sinon \end{cases} \tag{2.28}$$

Avec $v_i \in T_1$, $v_j \in T_2$ et v_g le nœud servant de point de référence.

Synthèse :

Notre présentation des distances pour trajectoires contraintes par un réseau, s'est articulée en quatre points :

- **les mesures simples** à savoir les distances SIM_{POI} et D_{NET}. La première compare les trajectoires avec comme support un ensemble prédéfini de points d'intérêt mais s'avère peu précise et subjective. Quant à la seconde elle est une extension de la distance Euclidienne adaptée aux trajectoires contraintes par un réseau ;
- **les mesures de type nœud à trajectoire** ont en commun de rechercher les plus courts chemins entre les nœuds d'une première trajectoire et l'ensemble des nœuds formant une seconde. On retrouve dans cette catégorie, les distances $K - PC$, $STLC$, et $Hausdorff_G$;
- **les mesures ensemblistes** telles que SIM_{TRAJ} et EBD qui évaluent la dissimilarité entre deux trajectoires à partir de leur intersection. Elles s'appliquent sur des trajectoires modélisées comme des séquences de segments routiers et sont rapides à calculer ;
- **les mesures intégrant les distorsions** qui sont essentiellement des extensions des distances *LCSS*, *DTW*, *EDR* et *ERP*, introduites précédemment. Elles gardent les mêmes propriétés que leur version en espace libre.

Les principales propriétés des distances pour trajectoires contraintes par un réseau sont synthétisées dans le tableau 2.2. Le bruit n'est pas pris en compte ici car les trajectoires sont supposées suivre parfaitement le réseau. Tout comme

dans le cas des mesures en espace libre, aucune des mesures en espace contraint ne satisfait simultanément à toutes les propriétés énumérées.

TABLE 2.2 – Tableau récapitulatif des caractéristiques des distances en espace contraint

Mesures	**Complexité**	**Métrique**	**Sans Paramètres**	**Décalages temporels locaux**
SIM_{POI}	$\mathcal{O}(\|P\|)$	✗	✗	✗
D_{NET}	$\mathcal{O}(n(\|V\|+\|E\|))$	✓	✓	✗
K-PC	$\mathcal{O}(nm(\|V\|+\|E\|))$	✗	✓	✗
STLC	$\mathcal{O}(nm(\|V\|+\|E\|))$	✗	✓	✗
HAUSDORFF_G	$\mathcal{O}(nm(\|V\|+\|E\|))$	✓	✓	✗
SIM_{TRAJ}	$\mathcal{O}(nlogn)$	✗	✓	✗
EBD	$\mathcal{O}(nlogn)$	✓	✓	✗
LORS	$\mathcal{O}(nm(\|V\|+\|E\|))$	✗	✓	✓
DTW_G	$\mathcal{O}(nm(\|V\|+\|E\|))$	✗	✓	✓
EDR_G	$\mathcal{O}(nm(\|V\|+\|E\|))$	✗	✗	✓
ERP_G	$\mathcal{O}(nm(\|V\|+\|E\|))$	✓	✗	✓

2.5 Clustering de trajectoires

Analyser les trajectoires implique souvent de pouvoir les rassembler en groupes relativement homogènes afin, par exemple, d'identifier dans une aire urbaine les principaux axes de déplacement, d'analyser les flux de mobilité, de cerner les habitudes et les motivations des déplacements. Cette répartition des trajectoires en groupes se fait à travers l'opération de clustering et les groupes créés sont appelés des clusters. Les méthodes de clustering pour trajectoires sont habituellement dérivées d'algorithmes classiques (k-moyennes ou *K-means* en anglais, DBSCAN, etc.) utilisés en fouille de données. Dans certains cas, il est d'ailleurs possible de regrouper les trajectoires en clusters en appliquant directement les algorithmes classiques de clustering, dès lors que l'on dispose d'une matrice des distances entre les trajectoires. Toutefois, la distance employée doit pouvoir rendre assez fidèlement compte de la proximité et/ou similarité des trajectoires pour un résultat optimal.

Comme pour la présentation des distances entre trajectoires, nous scindons l'état de l'art sur le clustering de trajectoires en deux grandes parties. La première

présente les méthodes pour les trajectoires libres et la seconde celles pour les trajectoires contraintes par un réseau.

2.5.1 Clustering de trajectoires libres

Nous présentons ici les approches de clustering pour les trajectoires réalisées en espace libre ou qui n'ont pas été recalées sur un réseau. Les approches présentées partitionnent les trajectoires autour de centroïdes, suivant leur densité, de façon hiérarchique, d'après des modèles statistiques ou en utilisant des réseaux de neurones.

2.5.1.1 Approches par partitionnement autour de centroïdes

L'idée fondamentale de ces approches est de créer des groupes dont les éléments sont plus proches du centre de leur cluster que du centre des autres clusters. L'algorithme phare de ce type d'approches est celui des k-moyennes. Il se déroule en trois phases. En premier lieu, les centres des clusters sont choisis au hasard. Ensuite, chaque élément du jeu de données est associé au centre dont il est le plus proche. En troisième lieu, le centre de chaque cluster est mis à jour en calculant la moyenne des éléments qui le composent. L'algorithme converge lorsque les clusters ne varient plus. Toutefois, il est plus judicieux d'utiliser des variantes de l'algorithme des k-moyennes telles que l'algorithme des k-médoïdes [KR09] et le Fuzzy C-Means [BEF84] dans lesquelles on définit un centroïde pour chaque cluster, c'est-à-dire l'élément le plus central du cluster, plutôt que d'en calculer la moyenne. Ce choix permet de contourner la forte complexité liée à la détermination de la moyenne d'un groupe de trajectoires de tailles différentes.

Dans [PKK+09], Pelekis et al. proposent ainsi une approche de clustering par partitionnement autour de centroïdes en trois étapes. Ils commencent par modéliser les trajectoires sous forme de vecteurs de même dimension. Puis, ils définissent une distance qui est une extension de la distance ERP, à même de supporter la modélisation en vecteur des trajectoires. Enfin, ils adaptent l'algorithme Fuzzy C-Means et utilisent la distance précédemment définie pour effectuer le clustering des trajectoires.

Les limites des approches par partitionnement autour de centroïdes sont liées à la nécessité de définir à l'avance le nombre de clusters alors même que cette information est souvent inconnue. En sus, la complexité de l'algorithme des k-

médoïdes augmente rapidement avec la taille du jeu de données et le nombre de clusters à trouver. Cela peut entraver son utilisation à grande échelle, d'autant plus que les distances entre trajectoires sont déjà relativement complexes à évaluer. Par ailleurs, ces approches génèrent des clusters de forme sphérique ou quasi-sphérique, alors même que les clusters de trajectoires sont plutôt de formes irrégulières. Notons enfin qu'elles sont sensibles aux distances utilisées et sont plus performantes avec des distances métriques qui rendent plus précisément compte de l'écart relatif entre les éléments comparés.

2.5.1.2 Approches basées sur la densité

Elles se caractérisent par la génération de clusters de densité homogène et sans restriction de forme. Ceci permet de pallier une des limites des approches par partitionnement. Ce sont les approches qui reviennent le plus souvent dans la littérature. Elles découlent de l'adaptation de l'algorithme DBSCAN [EKS+96] au clustering de trajectoires. DBSCAN est fonction de deux paramètres : ϵ qui est un rayon délimitant le $\epsilon - voisinage$ d'un point et $MinPts$, le nombre minimum de points devant appartenir à ce $\epsilon - voisinage$ pour que l'ensemble forme un embryon de cluster, qui grandira par la suite de proche en proche. Plus spécifiquement, l'algorithme vérifie si chaque point du $\epsilon - voisinage$, est lui-même entouré d'un minimum de $MinPts$ points. Si tel est le cas, tous les points du nouveau $\epsilon - voisinage$ ainsi obtenu sont ajoutés à l'embryon de cluster, sinon le point suivant est sélectionné. Il est essentiel d'user de distances métriques pour déterminer de façon fiable les éléments des $\epsilon - voisinages$.

Dans leur article [LHW07], Lee et al. présentent une méthode pour le clustering de trajectoires, basée sur la densité et nommée TRACLUS (*TRAjectory CLUStering*). Avant l'opération de clustering, ils commencent par découper les trajectoires en segments autour de points caractéristiques déterminés grâce à la méthode MDL (*Minimum Description Length*) [GMP05]. Une fois les trajectoires découpées, ils appliquent un algorithme inspiré de DBSCAN aux segments générés. Ils obtiennent alors des clusters de segments et, pour chaque cluster, ils extraient une trajectoire représentative. Notons que la méthode TRACLUS est souvent utilisée comme référence pour l'évaluation des algorithmes de clustering de trajectoires dans la littérature. D'autres travaux [NP06, KM15, LHLG08], présentent également des méthodes de clustering pour trajectoires libres inspirées de DBSCAN ou sa variante OPTICS qui n'a besoin que du paramètre $MinPts$.

La première limite des approches par densité est l'hypothèse de densité constante sur laquelle elles se fondent et qui n'est pas toujours vérifiée avec des jeux de données de trajectoires. La seconde limite est relative aux paramètres ϵ et $MinPts$ qui ne sont pas simples à fixer en pratique puisqu'ils dépendent des jeux de données étudiés.

2.5.1.3 Approches basées sur les modèles statistiques

Ces méthodes se fondent sur les théories relatives aux distributions statistiques et aux probabilités pour partitionner les trajectoires. C'est le cas par exemple de la méthode des mélanges gaussiens utilisée par [ZLL09, BGLR17] pour le clustering de trajectoires. Elle suppose que les trajectoires sont tirées d'une distribution, générée par un mélange pondéré de gaussiennes (distributions normales). Chaque gaussienne est associée à un cluster et est caractérisée par un poids, une moyenne et une variance qui sont inconnus au départ. L'objectif de la méthode est de déterminer les valeurs optimales des paramètres des gaussiennes qui permettent d'approcher au mieux la distribution globale des trajectoires. Pour ce faire, l'algorithme EM (Espérance-Maximisation) proposé par Dempster et al. [DLR77] est utilisé.

Comme pour les approches par partitionnement, il est nécessaire de définir en amont le nombre de clusters devant être produits, ce qui est le principal frein à la mise en œuvre de ces approches.

2.5.1.4 Approches hiérarchiques

Ces approches sont inspirées du clustering hiérarchique qui regroupe les données dans une structure arborescente et permet d'obtenir pour un même jeu de données plusieurs partitionnements à différentes échelles. La spécificité du clustering hiérarchique est qu'il se décline en une version ascendante et une descendante. Dans la version descendante, les trajectoires sont d'abord regroupées dans un seul cluster, ce dernier est ensuite subdivisé en plusieurs clusters qui eux-mêmes sont plus tard subdivisés en de nouveaux clusters, jusqu'à ce que chaque trajectoire constitue à elle seule un cluster. La version ascendante, suit le processus inverse. C'est à dire que les trajectoires commencent par former individuellement des clusters qui sont ensuite fusionnés progressivement jusqu'à ce que toutes les trajectoires appartiennent à un même cluster. Les clusters sont fusionnés en fonction d'un critère d'association (*Linkage Criteria* en anglais). Trois critères sont usuellement

employés :

- le critère d'association minimum (*single-linkage en anglais*) qui détermine la distance minimale entre les éléments de deux clusters et associe les clusters les plus proches selon cette distance ;
- le critère d'association maximum (*complete-linkage en anglais*) qui évalue la distance maximale entre les éléments de deux clusters et associe également les clusters les plus proches selon cette distance ;
- le critère d'association par la méthode de Ward (*ward-linkage en anglais*) qui associe des clusters si l'accroissement de variance au sein du nouveau cluster obtenu est faible.

Zhang et al. ont développé dans [ZLL18] une méthode hiérarchique, dérivée de TRACLUS, pour le clustering de trajectoires. Comme dans TRACLUS, l'approche présentée contient deux phases. La première, a pour but de partitionner les trajectoires en segments et la seconde utilise l'algorithme HDBSCAN pour regrouper les segments. Contrairement à DBSCAN, HDBSCAN est un algorithme de clustering hiérarchique capable de générer des clusters de densités variées. En plus des données de géo-localisation, ils intègrent les informations concernant la vitesse et la direction des trajectoires. Des travaux comme ceux décrits dans [ZZXM09, WQCZ18] exploitent aussi le clustering hiérarchique pour regrouper les trajectoires.

Les approches hiérarchiques présentent l'avantage de ne pas nécessiter la définition a priori du nombre de clusters. Elles peuvent être également utilisées avec des distances entre trajectoires métriques ou non. De plus, il est possible de visualiser le processus de clustering par le biais d'un dendrogramme pour analyser le regroupement des trajectoires à différentes échelles. Cependant, des données aberrantes et un choix inadéquat de distance entre clusters peuvent impacter négativement les résultats de ce type d'approche.

2.5.1.5 Approches utilisant les réseaux de neurones

Bien qu'ils soient initialement conçus pour des tâches de classification par apprentissage supervisé, les réseaux de neurones servent aussi à des fins de clustering. Yao et al. [YZZ+18] proposent, par exemple, une méthode de clustering de trajectoires basée sur l'usage d'un réseau de neurones particulier nommé auto-encodeur de type séquence-à-séquence [SVL14]. Un auto-encodeur est un réseau de neurones structuré en deux blocs : un encodeur et un décodeur. L'encodeur a pour but d'ex-

traire une représentation codée des données fournies en entrée tout en réduisant leur dimension. Le décodeur effectue l'opération inverse en reconstituant les données d'origine à partir de leur version encodée. Dans son fonctionnement, l'auto-encodeur évalue régulièrement les différences entre les données reconstituées et les données originales puis ajuste ses paramètres de sorte à réduire au mieux l'erreur mesurée. L'auto-encodeur de type séquence-à-séquence a la spécificité de prendre en entrée des séquences de tailles diverses et de fournir en sortie une représentation vectorielle uniformisée desdites séquences qui s'accompagne également d'une réduction de dimension. Dans l'approche de Yao et al., une fois que les trajectoires sont encodées, l'algorithme k-moyennes est appliqué aux données vectorielles générées afin d'obtenir les clusters de trajectoires. Outre les auto-encodeurs, les cartes auto-adaptatives (*Self Organizing Map* en anglais) qui sont une autre classe de réseaux de neurones, ont également été utilisées pour le clustering de trajectoires [SBVLK09, NK06]. Le principal inconvénient lié à l'usage des réseaux de neurones est qu'ils nécessitent un nombre important de calculs pour l'ajustement des paramètres dont ils dépendent.

Synthèse :

Nous classons les travaux relatifs au clustering de trajectoires libres, en cinq grandes familles d'approches :

- **les approches par partitionnement** autour de centroïdes qui regroupent les trajectoires avec des d'algorithmes dérivés de l'algorithme des k-moyennes ;
- **les approches basées sur la densité** qui, à partir de variantes de DBSCAN, agglomèrent les trajectoires en tenant compte de leur concentration ;
- **les approches basées sur les modèles statistiques** dont le principe est d'exploiter les notions relatives aux distributions statistiques pour rassembler les trajectoires ;
- **les approches hiérarchiques** qui regroupent les trajectoires dans une structure arborescente permettant une analyse multi-échelle de leur partitionnement ;
- les approches qui utilisent la puissance **des réseaux de neurones** pour répartir les trajectoires en clusters.

Ces différentes approches adaptent les algorithmes classiques de clustering aux

spécificités des trajectoires et partagent avec ceux-ci essentiellement les mêmes avantages et inconvénients.

2.5.2 Clustering de trajectoires contraintes par un réseau

Nous présentons dans cette sous-section les approches de clustering développées spécialement pour les trajectoires contraintes par un réseau. Ces approches intègrent, pour la plupart, la structure du réseau dans le processus de clustering. Nous les classons en quatre familles parmi lesquelles, on retrouve les approches hiérarchiques et celles basées sur la densité comme pour les trajectoires libres. Les deux autres familles d'approches développées pour ce type de trajectoires sont les approches basées graphe et celles par réduction de dimension.

2.5.2.1 Approches hiérarchiques

NNCluster est une approche hiérarchique de clustering de trajectoires contraintes par un réseau, conçue par Roh et al. [RH10]. Elle est structurée en deux parties : la première correspond à la construction de clusters initiaux et la deuxième est relative à l'expansion de ces derniers. Pour construire les clusters initiaux, les trajectoires sont comparées en utilisant la version pour trajectoires contraintes par un réseau de la distance de Hausdorff (cf équation 2.22). L'utilisation de cette distance métrique permet d'exploiter la propriété de l'inégalité triangulaire afin de réduire le nombre de comparaisons entre trajectoires, nécessaires à la construction des clusteurs initiaux. Deux trajectoires sont intégrées à un cluster initial, si leur proportion de voisins communs, parmi leurs k plus proches voisins respectifs, est supérieure à un seuil prédéfini. Dans la phase d'expansion, les clusters initiaux dont les centroïdes sont suffisamment proches sont fusionnés. La fusion se poursuit jusqu'à l'obtention d'un cluster unique et la valeur de l'index de Davies-Bouldin (DB) [DB79] est calculée à chaque étape. Cet index évalue la qualité d'un partitionnement de données et est fonction du rapport entre les distances intra-cluster et celles inter-cluster. Le clustering optimal est alors celui qui minimise l'index DB.

L'utilisation de NNCluster est essentiellement limitée par les paramètres dont elle dépend et dont les valeurs optimales varient suivant les données traitées.

2.5.2.2 Approches basées sur la densité

NETSCAN. Développée par Kharrat et al. [KPZF08], NETSCAN est une méthode de clustering pour les trajectoires contraintes par un réseau, basée sur la densité et qui se déroule en deux phases. Dans sa première phase, NETSCAN qui est inspirée de DBSCAN, identifie les segments routiers les plus denses, puis les regroupe en chemins denses. La densité, ici, est proportionnelle au nombre d'objets mobiles transitant par les segments routiers. Un chemin dense est une suite de segments ayant des densités similaires. Les chemins denses résultants de la première phase correspondent aux centres des clusters de trajectoires. La seconde phase consiste à comparer les trajectoires du jeu de données à ces chemins denses puis à les associer individuellement au cluster dont le chemin dense est le plus proche. La mesure de similarité utilisée est SIM_TRAJ (cf équation 2.25). A terme, le nombre de clusters de trajectoires correspond au nombre de chemins denses.

NETSCAN présente l'avantage d'intégrer le flux de mobilité et la structure du réseau dans la détermination des clusters. Cependant, on notera que seules les trajectoires qui intersectent les chemins denses sont prises en compte dans le clustering et, même pour ces dernières, NETSCAN ne retient que les parties communes entre elles et les chemins denses. En outre, la méthode a un coût important en termes de calculs à effectuer.

NEAT. La seconde approche basée sur la densité que nous présentons est NEAT [HLO13]. Elle a été proposée par Han et al. et est une approche modulable de clustering pour trajectoires contraintes par un réseau. C'est-à-dire qu'elle peut fournir différents types de clustering suivant les applications visées. Elle se décompose en trois phases :

❶ Dans la première phase, les trajectoires sont découpées, suivant les segments du réseau routier, en t-fragments. Plus précisément, un t-fragment est une sous-trajectoire dont tous les points appartiennent au même segment routier. L'ensemble des t-fragments associés à un segment routier forme un cluster de base.

❷ Dans un second temps, les clusters de base sont agrégés en clusters de flux. En pratique, il s'agit de regrouper les t-fragments qui sont associés à des segments routiers adjacents en combinant trois facteurs : la densité de t-fragments dans les clusters de base, le flux des trajectoires d'un cluster de base à l'autre et la limitation de vitesse au niveau des segments routiers.

Ces trois facteurs sont pondérés par des coefficients qui permettent de moduler leur influence et d'obtenir différents types de clustering répondant à des applications spécifiques. Les clusters de flux générés sont des séquences continues de t-fragments qui suivent la même route.

❸ La troisième étape a pour objectif d'affiner les clusters de flux en les regroupant, si nécessaire, grâce à une version modifiée de l'algorithme DBSCAN. Pour évaluer la proximité de deux clusters de flux, on compare les routes qu'ils suivent grâce à une version adaptée de la distance de Hausdorff qui intègre les contraintes du réseau routier. La version de DBSCAN utilisée commence toujours par le cluster de flux suivant la plus longue route afin de rendre l'algorithme déterministe. C'est-à-dire que le résultat du clustering ne change pas pour un même jeu de données. Ce qui n'est pas le cas de la version originale de DBSCAN dont les résultats varient en fonction de l'ordre dans lequel les données sont traitées.

Le principal avantage de NEAT est sa modularité mais il s'applique qu'à des portions de trajectoires et ne permet donc pas un clustering des trajectoires entières.

2.5.2.3 Approches par réduction de dimension

La réduction de dimension consiste à projeter les trajectoires dans un espace euclidien de dimension plus faible que celui des trajectoires. Cela permet d'uniformiser la représentation des trajectoires qui deviennent des points multidimensionnels et de simplifier leur traitement avec les algorithmes classiques de clustering.

DSL-FastMap. Dans [WKBL09], Won et al. ont introduit une méthode de clustering par réduction de dimension nommée DSL-FastMap pour les trajectoires contraintes par un réseau. Leur approche consiste à projeter les trajectoires dans un espace de dimension fixe par le biais de l'algorithme FastMap [FL95] avant de les regrouper par un clustering hiérarchique classique. Les auteurs ont également proposé une distance nommée DSL qui compare deux trajectoires en fonction de la longueur totale des segments qu'ils n'ont pas en commun. Notons que FastMap projette les trajectoires par le biais d'une fonction numérique définie de sorte à conserver le même écart relatif entre trajectoires dans le nouvel espace.

TRACEMOB proposée par Han et al. [HLO17] exploite aussi la réduction de dimension pour le clustering de trajectoires contraintes par un réseau. Elle se structure en trois phases :

❶ La première phase est relative au calcul des distances entre trajectoires, avec une mesure inspirée de l'indice de Jaccard [Jac01] qui prend en compte la proximité des trajectoires, grâce à l'utilisation d'un index spatial. Cette mesure permet ainsi de renvoyer des valeurs faibles pour des trajectoires parallèles (par exemple des trajectoires de voitures se déplaçant en sens opposés le long d'une autoroute) tant que celles-ci sont proches l'une de l'autre. Ce n'est pas le cas avec d'autres mesures du même type. Rappelons que l'indice de Jaccard équivaut au rapport entre la taille de l'intersection de deux ensembles et la taille de leur union. Le résultat de cette phase est une matrice de distance entre trajectoires.

❷ La seconde phase a pour but de projeter les trajectoires dans un espace euclidien en utilisant la matrice de distance précédemment obtenue. Cette transformation se fait via l'application de l'algorithme TrajMap qui conserve l'écart relatif entre les trajectoires même après leur projection. TrajMap est une extension de FastMap et a une complexité linaire. L'objectif de cette phase est de pouvoir ensuite tirer parti des algorithmes de clustering classiques.

❸ Durant la troisième phase, les auteurs appliquent l'algorithme de clustering k-moyennes aux points issus de la projection des trajectoires.

Le challenge avec les approches par réduction de dimension concerne principalement la détermination de la dimension optimale de l'espace dans lequel les trajectoires doivent être projetées. Ceci dans l'optique d'atténuer au mieux les pertes d'information liées à ces opérations.

2.5.2.4 Approches basées graphe

Les trois approches décrites ci-après utilisent des techniques issues de la théorie des graphes pour le clustering des trajectoires. Elles nécessitent une représentation des trajectoires directement sous forme de graphe et ont été proposées par El Mahrsi et al. [EMR12]. Pour chacune d'elles, ils partent d'une représentation des trajectoires et des segments routiers qui les composent sous la forme d'un graphe biparti, avec d'un côté l'ensemble des nœuds correspondant aux trajectoires et

de l'autre l'ensemble des nœuds correspondant aux segments. Les deux premières approches impliquent des projections du graphe biparti sur un des deux ensembles de nœuds tandis que la troisième approche exploite directement le graphe biparti.

Plus précisément dans la **première approche**, les trajectoires sont considérées comme des collections ou sacs de segments. Le poids d'un segment dans une trajectoire est évalué en utilisant une version légèrement modifiée de la mesure TF-IDF (Term Frequency - Inverse Document Frequency) [WLWK08] qui prend en compte la longueur des trajectoires. Initialement, TF-IDF est une mesure conçue pour déterminer l'importance relative d'un terme dans une collection de documents en discriminant négativement les termes qui apparaissent fréquemment (articles, pronoms) dans la plupart des documents. En tenant compte des poids calculés, le graphe biparti est transformé en un graphe de similarité où les nœuds correspondent aux différentes trajectoires. Plus précisément, deux nœuds sont reliés par une arête si la similarité des trajectoires correspondantes est supérieure à zéro. Par la suite, on applique au graphe de similarité un algorithme de détection de communauté basé sur l'évaluation de la modularité pour créer des clusters. Pour chaque cluster, l'algorithme est à nouveau récursivement appliqué sur le sous-graphe associé jusqu'à ce qu'il ne soit plus possible d'obtenir de nouveaux clusters. On obtient ainsi une collection hiérarchique de clusters imbriqués de trajectoires qui permet d'analyser les données à différents niveaux de granularité.

La **deuxième approche** suit les mêmes étapes que la première. Toutefois, la perspective est ici inversée, c'est-à-dire qu'on considère que chaque segment est associé à un ensemble de trajectoires. L'objectif étant d'obtenir des clusters de segments et non de trajectoires. On calcule donc pour chaque segment les poids des trajectoires qui l'ont emprunté en s'inspirant de la mesure TF-IDF. Le graphe biparti est *a posteriori* transformé en un graphe de similarité dont les nœuds sont des segments routiers. Par la suite, un algorithme de recherche de communautés est utilisé comme précédemment pour extraire les clusters de façon hiérarchique.

La **troisième approche** consiste à traiter directement le graphe biparti lui-même en appliquant la méthode de co-clustering MODL [Bou11]. Une méthode de co-clustering vise à réorganiser les lignes et les colonnes de la matrice d'adjacence d'un graphe afin de mettre en évidence des blocs de densité homogène qui correspondent aux clusters. Dans le cas de notre graphe biparti, les lignes de la matrice d'adjacence correspondent aux trajectoires et les colonnes aux segments routiers.

Des trois méthodes présentées, celle du co-clustering est la plus efficace car elle ne nécessite pas l'usage d'une mesure de similarité et produit à la fois un clustering pour les trajectoires et un clustering pour les segments routiers.

Plus généralement, les méthodes de clustering à base de graphe présentent l'avantage de ne pas dépendre de paramètres et de fournir un clustering de trajectoires ou de segments au choix. Toutefois elles sont fortement impactées par la qualité du processus de transformation des données en graphes. Leur autre limite est relative à la forte complexité des algorithmes de traitement des graphes même s'il existe aujourd'hui des heuristiques qui permettent d'obtenir des solutions approchées assez rapidement.

Synthèse :

Les approches de clustering pour trajectoires contraintes par un réseau que nous avons présentées sont regroupées en quatre classes :

— **les approches hiérarchiques** avec pour exemple NNCluster qui regroupe les trajectoires dans une structure arborescente ;

— **les approches basées sur la densité** telles que NETSCAN qui définit des clusters autour de chemins denses et NEAT qui peut générer différents types clusters en fonction de la densité, du flux et de la vitesse des trajectoires ;

— **les approches par changement de dimension** à l'instar de DSL-FastMap et TRACEMOB qui permettent de projeter les trajectoires dans des espaces euclidiens de faible dimension pour en faciliter le clustering ;

— **les approches basées graphe** qui exploitent les outils de la théorie des graphes comme la détection de communauté ou le co-clustering pour déterminer les clusters de trajectoires.

Ces approches diffèrent essentiellement de celles pour trajectoires libres par la prise en compte explicite des éléments du réseau routier comme les segments routiers ou la définition de distances qui intègrent les informations du réseau routier.

2.6 Limites de l'état de l'art

Concernant l'étape du **prétraitement**, nous relevons deux limites essentielles :

— la première est liée à **l'insuffisance de jeux de données libres d'accès et incluant les vérités terrains**. Cette insuffisance a pour conséquence directe de limiter l'évaluation et la comparaison des algorithmes de map-matching. À notre connaissance seul Kubička et al. [KCM+15] ont proposé un jeu de données librement accessible avec vérité terrain mais celui-ci ne contient que 100 trajectoires. D'autres jeux de données existent mais ils sont soit payants soit inaccessibles au public ;

— la seconde a trait à **l'absence de solution de détection et de correction à grande échelle** des erreurs (discontinuités, boucles et détours inutiles) découlant de l'application des algorithmes de map-matching. En effet, bien qu'il existe des outils de post-matching interactifs et visuels pouvant servir à cette tâche, ils ne permettent pas de gérer des jeux de données regroupant de grandes masses (des millions voire des milliards) de trajectoires.

Si les propositions de **distances et similarités** ont été nombreuses dans l'état de l'art, tant pour les trajectoires non-contraintes que pour les trajectoires contraintes, nous avons cependant identifié trois principales limites :

— en premier lieu, **aucune des mesures existantes ne constitue une solution universelle** pour comparer les trajectoires car chacune à ses avantages et ses limites. Il faut ainsi avoir à l'esprit le type de trajectoires disponibles, et l'application visée, afin de choisir en conséquence la mesure la plus appropriée ;

— en second lieu, **les mesures pour espace contraint sont généralement plus complexes à évaluer que celles pour espace libre**, en raison de la recherche du plus court chemin entre nœuds. En effet, le calcul du plus court chemin dépend du nombre de nœuds et d'arêtes du réseau et a une complexité en $\mathcal{O}(|V| + |E|)$, complexité qui est bien plus élevée que le calcul de la distance euclidienne entre des points en espace libre qui se fait en $\mathcal{O}(1)$. Ce coût de calcul plus important limite le plus souvent l'utilisation des mesures pour espace contraint à des jeux de données de petite taille. Des solutions existent pour pallier cette difficulté, par exemple l'utilisation d'index et de techniques de filtrage mais elles sont développées pour comparer les trajectoires en espace libre et s'adaptent difficilement aux cas des trajectoires contraintes. Des efforts restent donc à fournir dans le sens du développement de méthodes d'indexation et de filtrage propres aux trajectoires contraintes

même si on assiste à l'émergence de travaux [SCW+17], [WBC+18], [SLB18] de ce type dans la littérature ;

— en troisième lieu, **il n'existe pas, à notre connaissance, un travail de revue qui compare les performances des distances pour espace contraint** contrairement au cas des distances en espace libre, cf revue [SLZ+20].

Enfin, sur les aspects liés au **clustering** de trajectoires contraintes par un réseau, les limites des approches actuelles concernent :

— tout d'abord, la **forte complexité** des solutions proposées qui entrave leur application à de grandes masses de données. Cette complexité est essentiellement due aux calculs de distances entre trajectoires et aux stratégies utilisées pour leur regroupement ;

— la **sensibilité des approches présentées vis-à-vis des multiples paramètres** dont elles dépendent.

La plupart des travaux qui suivent tentent de répondre à ces limitations. Comme solution à l'insuffisance de données, nous plaidons pour un effort collectif de la communauté visant une libération des données, ne serait-ce qu'à des fins scientifiques. Relativement à la gestion, à grande échelle, des erreurs issues du map-matching, nous proposons une mesure pour détecter automatiquement les trajectoires potentiellement erronées et une solution algorithmique pour leur correction. En vue d'assurer un fonctionnement optimal de notre algorithme, les paramètres dont il dépend sont ajustés en fonction de corrections manuelles, réalisées par des opérateurs humains, via une plateforme web dédiée. Ces travaux sont présentés dans le chapitre 3.

Une analyse comparative des mesures provenant de familles distinctes, montre que certaines ont tendance à compenser mutuellement leurs insuffisances. C'est en particulier le cas des distances de type nœud à trajectoire et de celles intégrant les distorsions. Partant de ce constat, nous proposons une nouvelle distance, pour les trajectoires contraintes par un réseau, qui intègre les points forts des deux familles de mesures tout en inhibant leurs inconvénients. Par ailleurs, nous réalisons également une étude comparative de quelques distances existantes dédiées aux trajectoires contraintes par un réseau. Ces deux contributions se trouvent dans le chapitre 4.

Durant cette thèse nous avons également développé des approches basées graphe pour le clustering de trajectoires contraintes par un réseau. Toutefois, nous ne présentons pas nos résultats car lesdits travaux n'ont pas atteint, à notre sens, une maturité suffisante.

Chapitre 3

Prétraitement des trajectoires

Sommaire

3.1 Introduction

Le présent chapitre est consacré à la description et au prétraitement des données exploitées dans le cadre de nos diverses expérimentations. Ces données contiennent des trajectoires brutes et réelles issues de trajets de taxis dans une ville ainsi que les informations sur le réseau routier de cette dernière. Le prétraitement des trajectoires a consisté, dans un premier temps, à les recaler sur le réseau routier puis, dans un second temps, à corriger les erreurs découlant de ce processus. A cet effet, une solution algorithmique et une plateforme web ont été proposées pour la détection et la correction de trajectoires mal recalées ou contenant des incohérences dans leur tracé. En particulier, la plateforme web a pour but de permettre un traitement des trajectoires par des opérateurs humains en vue de générer des valeurs optimales pour les paramètres de la solution algorithmique. En outre, nous présentons également les choix faits pour le stockage et la gestion des trajectoires avant et après leur prétraitement. La première section est dédiée à la description des données, la seconde au map-matching et à la gestion des trajectoires et la troisième au post-traitement des trajectoires.

3.2 Description des données exploitées

3.2.1 Jeu de données de trajectoires

Les trajectoires que nous exploitons décrivent les trajets effectués par 442 taxis dans la ville de Porto au cours d'une année complète du 01/07/2013 au 30/06/2014. Elles proviennent d'un jeu de données ouvert dénommé "Taxi Service Trajectory" qui a été constitué et publié dans le cadre de la compétition Kaggle ECML/PKDD 2015 par Moreira-Matias et al. [MMGF+13]. La ville de Porto est le cœur de la deuxième plus grande zone urbaine du Portugal, après Lisbonne, avec une population de 237591 habitants [1] et une superficie de 41,42 km^2. Elle est également la capitale de la région nord du Portugal et est bordé par le fleuve Douro et l'océan Atlantique. La ville dispose d'un système de transport intégré combinant bus, tramways, métro et taxis. Certains des taxis sont équipés de terminaux de données mobiles capables de fournir diverses informations à leur sujet et en particulier leur position. Ce sont les informations collectées par ces terminaux qui ont permis

1. recensement de 2011 : `http://mapas.ine.pt`

de constituer le jeu de données dont la structure et un échantillon sont présentés respectivement dans les tableaux 3.1 et 3.2. La figure 3.1 propose quant à elle une visualisation de quelques trajectoires brutes.

TABLE 3.1 – Structure du jeu de données "Taxi Service Trajectory"

Données	**Type**	**Description**
Trip_id	Entier	Identifiant du trajet
Call_type	Caractère	Type de demande (envoi depuis la centrale, demande directe au chauffeur sur une borne, demande directe dans la rue)
Origin_call	Entier	Identifiant anonyme du numéro de téléphone utilisé pour faire la demande
Origin_stand	Entier	Borne d'origine du taxi (si le trajet est demandé directement sur une borne)
Taxi_id	Entier	Identifiant du taxi
Timestamp	Entier	Horodatage du début du trajet
Day_type	Caractère	Type de jour (férié, ouvrable ou week-end)
Missing_Data	Booléen	Absence ou non de données GPS
Polyline	Tableau de tableaux de Réels	Polyligne contenant une liste de coordonnées GPS (latitudes et longitudes) collectées toutes les 15 secondes

TABLE 3.2 – Extrait des données décrivant les trajectoires brutes

Trip_id	**Call_type**	**Origin_call**	**Origin_stand**	**Taxi_id**	**Timestamp**	**Day_type**	**Missing_Data**	**Polyline**
1372637343620000571	A	31508	N/A	20000571	1372637343	A	False	[[8.618868,41.155101],...]
1372641742620000657	B	N/A	22	20000657	1372641742	A	False	[[8.689338,41.168124],...]
1372636858620000589	C	N/A	N/A	20000589	1372636858	A	False	[[8.618643,41.141412],...]

FIGURE 3.1 – Visualisation d'un échantillon de trajectoires brutes.

3.2.2 Réseau routier

Le réseau routier est une donnée indispensable aux opérations entrant dans le cadre du map-matching des trajectoires brutes. Dans nos travaux, il est modélisé sous la forme d'un graphe dirigé dont les nœuds représentent soit des intersections soit des points au niveau desquels les routes changent de forme. Les données cartographiques que nous utilisons pour générer le réseau routier proviennent de *OpenStreetMap*[2] qui est une plateforme collaborative de cartographie sous licence libre. La figure 3.2 présente une visualisation du réseau routier de la région de Porto obtenu à partir de cette plateforme. La difficulté avec ces données vectorielles, est qu'elles sont difficilement exploitables en l'état dans une chaîne de traitement. Elles doivent être préalablement stockées et organisées, ce qui peut se faire au sein d'une base de données. Nous réalisons l'opération d'importation des données vectorielles vers une base de données grâce à l'outil *Osmosis*[3]. Il s'agit d'une application java fonctionnant en ligne de commande et permettant d'importer les informations issues d'*OpenStreetMap* dans une base de données sous *Postgresql*[4] avec l'extension *Postgis*[5] activée. Toutes les opérations nécessaires, y compris la création de la base de données et son peuplement, sont automatiquement gérées par *Osmosis*. Le réseau routier est enregistré sous la forme d'une suite de segments routiers décrits par les informations contenues dans le tableau 3.3.

2. `https://www.openstreetmap.org`
3. `https://wiki.openstreetmap.org/wiki/FR:Osmosis`
4. `https://www.postgresql.org`
5. `https://postgis.net/`

FIGURE 3.2 – Réseau routier de Porto.

TABLE 3.3 – Structure de la base de données contenant le réseau routier

Données	**Types**	**Descriptions**
id	Entier	Identifiant du segment routier
osm_name	Chaîne de caractères	Nom de la route
source	Entier	Identifiant du nœud source
target	Entier	Identifiant du nœud destination
km	Réel	Longueur du segment routier
kmh	Réel	Vitesse maximale autorisée
cost	Réel	Coût associé au parcours du segment routier
x1	Réel	Longitude du nœud source
y1	Réel	Latitude du nœud source
x2	Réel	Longitude du nœud destination
y2	Réel	Latitude du nœud destination
geom_way	LineString	Représentation en polyligne du segment routier

3.3 Map-matching et gestion des trajectoires

3.3.1 Map-matching par modèle de Markov caché

Nous faisons le choix de l'approche par modèle de Markov caché (cf Chapitre 2, section 2.3.1.1), pour le map-matching de nos trajectoires. C'est une approche qui se prête naturellement à la mise en œuvre du map-matching et qui offre de très bonnes performances. Elle est par ailleurs assez régulièrement utilisée dans la littérature. Sur un plan conceptuel, les algorithmes de map-matching utilisant le modèle de Markov caché fonctionnent en deux phases. La première phase a pour but de créer un graphe pondéré, de segments routiers, contenant tous les chemins possibles pouvant correspondre aux mesures GPS de la trajectoire traitée. Quant à la deuxième phase, elle vise à extraire de ce graphe pondéré, le chemin optimal qui correspond à la trajectoire brute. Plus spécifiquement :

❶ Lors de la première phase, un ensemble de segments routiers, encore appelés candidats, sont sélectionnés dans le voisinage de chacun des points GPS de la trajectoire. Chaque candidat se voit affecté d'une probabilité dite d'émission, qui quantifie la vraisemblance qu'il soit le segment routier sur lequel s'effectuait le déplacement au moment de l'enregistrement du point GPS. La probabilité d'émission décroît en fonction de la distance entre le point GPS et le segment routier. Une fois les candidats de tous les points GPS déterminés, un graphe pondéré les connectant est construit. Les nœuds du graphe correspondent aux candidats et les arêtes relient uniquement les candidats d'un point à ceux de son successeur et ainsi de suite. Les candidats d'un même point ne sont pas interconnectés. La pondération associée à une arête correspond à la probabilité de transition entre les candidats qu'elle relie. Cette probabilité quantifie la chance de passer d'un candidat à un autre connaissant les points GPS auxquels ils sont respectivement associés. Le graphe obtenu, constitue dès lors un ensemble de solutions regroupant tous les chemins probables, sur le réseau routier, pouvant correspondre à la trajectoire traitée.

❷ Au cours de la deuxième phase, on extrait le chemin qui est à la fois le plus optimal en terme de proximité par rapport à la trajectoire originale mais aussi le plus cohérent en terme de parcours. Le chemin optimal maximise la probabilité d'observer une certaine séquence de segments routiers connaissant

la séquence de points GPS mesurés. Cette probabilité dépend elle-même des probabilités d'émission et de transition entre candidats.

En pratique, nous avons opté pour l'utilisation de la librairie *Barefoot*. Elle implémente l'algorithme de map-matching basé sur le modèle de Markov caché proposé par Newson et al [NK09]. La probabilité d'émission y est calculée grâce à la fonction suivante :

$$p(z_t|r_i) = \frac{1}{\sqrt{2\pi\sigma_z}} \exp\left[-0.5\left(\frac{||z_t - x_{t,i}||_{grand\_cercle}}{\sigma_z}\right)^2\right] \tag{3.1}$$

Cette fonction suppose que l'erreur du GPS suit une loi normale de moyenne nulle. σ_z représente l'écart type des erreurs de mesure du GPS. $||z_t - x_{t,i}||_{grand\_cercle}$ équivaut à la distance du grand cercle séparant le point z_t mesuré par le GPS à l'instant t et $x_{t,i}$ qui est le point le plus proche de z_t sur le segment routier r_i. Notons que la distance du grand cercle évalue l'éloignement entre deux points à la surface de la terre en tenant compte de la courbure de notre planète. La probabilité de transition entre deux segments routiers r_i et r_j est définie par :

$$p(d_t) = \frac{1}{\beta} \exp\left(\frac{-d_t}{\beta}\right) \tag{3.2}$$

Avec $d_t = ||z_t - z_{t+1}||_{grand\_cercle} - ||x_{t,i} - x_{t+1,j}||_{route}$

$||z_t - z_{t+1}||_{grand\_cercle}$ correspond à la distance du grand cercle entre deux points consécutifs de la trajectoire mesurés au GPS. $||x_{t,i} - x_{t+1,j}||_{route}$ est la distance suivant le réseau routier, c'est à dire le plus court chemin sur le réseau routier, entre $x_{t,i}$ et $x_{t+1,j}$ qui sont les points les plus proches respectivement de z_t et z_{t+1} sur les segments routiers r_i et r_j. d_t compare ainsi la distance en ligne directe à celle suivant le réseau routier. Cette formulation découle d'expérimentations menées sur des données de trajectoires réelles. En effet, on remarque que la distance en ligne directe entre les points GPS consécutifs d'une trajectoire est proche de celle les séparant suivant le réseau routier. En outre, l'écart entre ces deux distances suit une loi exponentielle. Ce qui explique que ce soit la loi utilisée dans la formule de calcul de la probabilité de transition.

La troisième particularité de la méthode de Newson et al. concerne l'utilisation de l'algorithme de Viterbi pour la détermination du chemin optimal. Cet algorithme basé sur le concept de programmation dynamique permet de réduire drastiquement la complexité du calcul du chemin optimal qui autrement serait

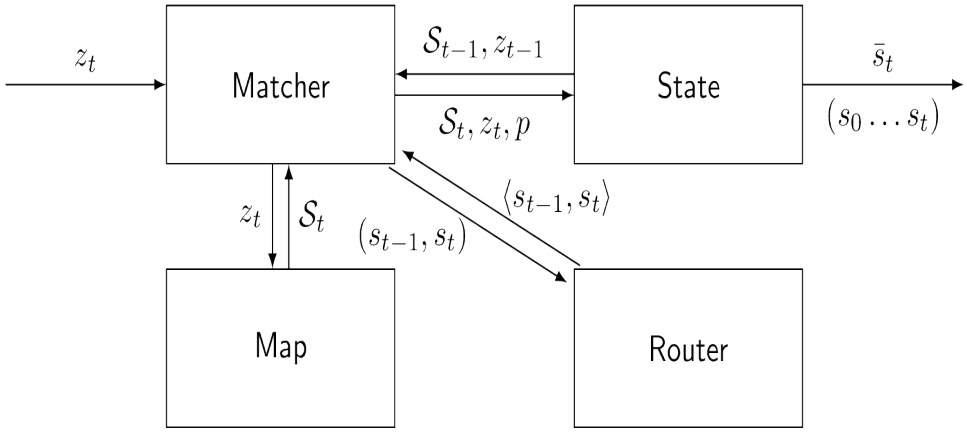


FIGURE 3.3 – Structure de la libraire Barefoot (image tirée de `https://github.com/bmwcarit/barefoot/wiki`).

difficilement déterminable. Pour mieux appréhender le problème, prenons une trajectoire contenant n points GPS, on suppose que chacun des points génère k candidats. Le nombre de chemins possibles s'élèverait alors à k^n. Pour une trajectoire contenant dix points qui génèrent chacun dix candidats cela reviendrait à choisir parmi 10^{10} soit 10 milliards de chemins possibles. L'algorithme de Viterbi réduit le nombre de possibilités à explorer en éliminant au fur et à mesure les transitions entre candidats en fonction des coûts des chemins qu'elles induisent. Dans le cas du map-matching, le coût d'un chemin correspond à la probabilité d'observer la séquence des candidats le constituant connaissant la suite de points GPS enregistrés. Ainsi à chaque étape de l'algorithme, seules les transitions entre candidats qui permettent de former les chemins les plus probables sont conservées.

3.3.2 Implémentation

Barefoot[6] est une librairie développée en Java qui peut être utilisée à la fois pour le map-matching en ligne et hors ligne. Dans le cadre de nos travaux, nous l'exploitons en mode hors ligne, nos données étant déjà entièrement enregistrées. *Barefoot* se compose de quatre blocs inter-dépendants comme illustré à la figure 3.3.

Le bloc *Matcher* orchestre les opérations de map-matching, le bloc State enregistre les candidats pour chacun des points et leurs probabilités respectives, le bloc *Map* permet d'accéder au réseau routier pour rechercher les candidats proches d'un

6. `https://github.com/bmwcarit/barefoot/wiki`

point GPS et le bloc *Router* aide à calculer les plus courts chemins entre paires de candidats.

La librairie *Barefoot* dépend d'un ensemble de paramètres dont certains jouent un rôle déterminant dans la qualité du résultat produit. Nous présentons ces paramètres essentiels ainsi que les valeurs que nous leur avons fixées dans le tableau 3.4.

TABLE 3.4 – Tableau récapitulatif des paramètres essentiels de Barefoot

Paramètres	**Valeurs**	**Descriptions**
matcher.sigma	5	Écart type de l'erreur du GPS. Il s'exprime en mètres et influe sur la probabilité d'émission.
matcher.lambda	0.1	Facteur de normalisation dans la formule de la probabilité de transition.
matcher.radius.max	200	Rayon autour des points GPS dans lequel rechercher les candidats. Il s'exprime en mètres.
matcher.distance.max	15000	Distance maximale en mètres du chemin sur le réseau routier reliant deux candidats.
matcher.distance.min	10	Distance minimale en mètres devant exister entre deux points GPS à traiter.
matcher.interval.min	1000	Temps minimal en millisecondes devant exister entre deux points GPS à traiter.

Parmi ces paramètres, *matcher.sigma, matcher.lambda, matcher.radius.max* influent directement sur la qualité du map-matching car ils interviennent dans le calcul des probabilités d'émission et de transition tandis que *matcher.distance.max, matcher.distance.min, matcher.interval.min* ont plutôt un effet indirect. Ces derniers permettent surtout de moduler la vitesse de traitement des trajectoires en sautant certains points lorsqu'ils sont trop rapprochés, dans l'espace ou le temps, de points déjà traités. Cela est utile quand les trajectoires contiennent un grand nombre de points.

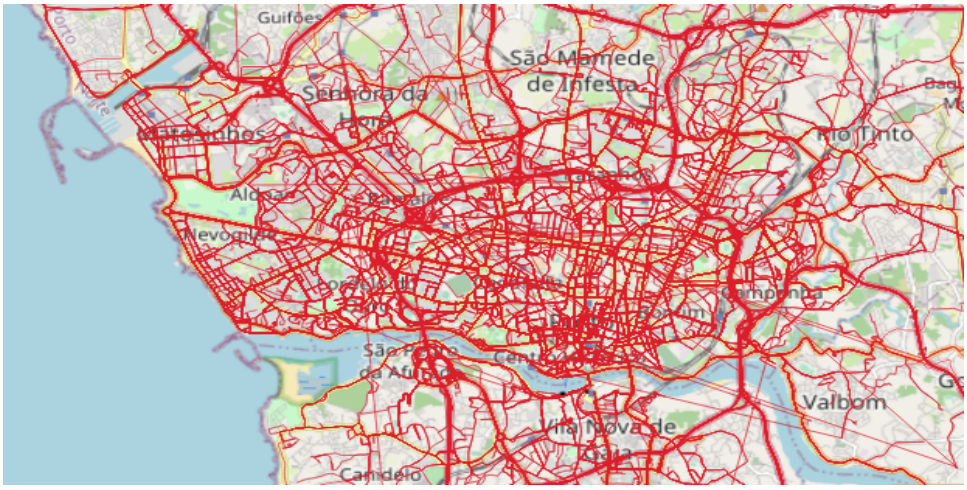


FIGURE 3.4 – Échantillon de trajectoires ayant subi le map-matching.

En raison du volume de trajectoires à traiter dans notre jeu de données, environ 1710670, nous avons été contraints de paralléliser le processus de map-matching. Toutes les implémentations sont réalisées en Java. Le serveur utilisé comporte 32 cœurs Intel Xeon E5-2630 v3 et tourne sous *Linux UBUNTU 16.04* avec 128 GB de RAM. *Barefoot* propose en sortie une représentation en JSON des trajectoires recalées. Partant de cette représentation, nous enregistrons dans notre base de données les trajectoires sous forme de séquences de nœuds et de segments routiers mais aussi sous forme de polylignes. Lors du map-matching un certain nombre de trajectoires aberrantes sont automatiquement exclues. Les trajectoires aberrantes sont soit très courtes (autour d'une ou de quelques dizaines de mètres à peine) ou présentent des incohérences dans les coordonnées GPS, par exemple une liste vide de coordonnées. Au terme du map-matching de notre jeu de données, nous obtenons 1476910 trajectoires recalées soit $86,33\%$ de l'ensemble de départ.

La figure 3.4 montre un échantillon de trajectoires après le map-matching. Comme attendu, on remarque que les trajectoires ayant subi le map-matching suivent bien mieux le réseau routier que les trajectoires brutes même si quelques imprécisions persistent en raison des erreurs liées à la qualité des données et au processus de map-matching en lui-même.

3.3.3 Stockage et gestion des trajectoires

Les trajectoires étant des séquences de localisations GPS décrites par leur latitude et leur longitude, elles peuvent être stockées dans tout système de gestion de base de données (SGBD) classique. Toutefois, il est plus pertinent de disposer d'un SGBD capable de gérer des objets géométriques comme *PostgresSQL*, *Oracle*[7] ou *Mysql*[8] afin de faciliter leur indexation et leur manipulation. Le type de données *LineString* est le plus utilisé pour stocker les trajectoires sous forme d'objets géométriques. Ce type représente à la fois la séquence de points de la trajectoire et les segments de ligne qui les connectent. Avec l'évolution des volumes de trajectoires sont apparues des solutions utilisant des SGBD distribués ou en nuage (*cloud* en anglais) dédiées au stockage des trajectoires. Elles privilégient l'écosystème *Azure*[9] de *Microsoft* pour le cloud [RLB+18, BLYZ16, LRB+17] et l'écosystème *Apache*[10] [MAZE+14, FGV19, CC15] ou la suite *GeoMesa*[11] pour les solutions simplement distribuées. En plus des systèmes déjà énumérés, qui ne sont pas destinés exclusivement aux trajectoires, il existe également des outils développés spécifiquement pour gérer ces dernières. Nous pouvons citer par exemple *TrajBase* [ZADE+17] qui est le module de base de données du logiciel de visualisation de trajectoires *TrajVis*, *TrajStore* [CMWM10], *SharkDB* [WZX+14] et *UlTraMan* [DCG+18]. Ces différents outils permettent de stocker et d'interroger efficacement de grands volumes de trajectoires. Dans nos travaux nous avons fait le choix de stocker nos données avec le système de gestion de bases de données *Postgresql* qui est un outil libre de droits, simple d'usage et offrant de bonnes performances. Par ailleurs, *Postgresql* dispose d'une extension nommée *Postgis* dédiée au traitement des données spatiales qui très riche en termes de fonctionnalités. Il permet également de gérer de grandes masses de données avec fluidité.

3.4 Post-traitement des trajectoires

Indépendamment de l'approche utilisée, le map-matching automatique de trajectoires est sujet à des imprécisions. Il importe donc de pouvoir évaluer la qua-

7. `https://www.oracle.com/index.html`
8. `https://www.mysql.com/`
9. `https://azure.microsoft.com/en-us/`
10. `https://www.apache.org/`
11. `https://www.geomesa.org/`

lité des trajectoires obtenues à l'issue du map-matching afin d'identifier celles qui nécessitent un post-traitement. Une première solution consiste à visualiser les trajectoires originales et leur versions recalées afin d'identifier d'éventuelles erreurs. Nous avons mis en œuvre cette solution grâce à l'outil *QGIS*[12] (Quantum GIS). Il s'agit un logiciel de traitement de données géographiques, libre de droits, qui permet de se connecter à une base de données contenant des données spatiales et de visualiser ces dernières. L'analyse visuelle des trajectoires générées à l'issue du processus de map-matching révèle la présence d'erreurs dans l'association des points GPS aux segments routiers. Ces erreurs prennent essentiellement la forme de discontinuités, de boucles ou de détours dans les trajectoires.

Toutefois, le volume des trajectoires traitées ne permet pas de s'appuyer uniquement sur une analyse visuelle pour la détection des erreurs dues au map-matching. Ceci en raison du temps et des efforts nécessaires. Il faut donc pouvoir trouver une solution algorithmique pouvant être utilisée à grande échelle. Notre proposition a consisté à définir un ratio permettant de comparer les longueurs des trajectoires originales et celles de leurs versions recalées dans le but d'identifier celles qui pourraient présenter des erreurs. L'intuition de cette proposition se fonde sur le fait qu'en dépit de l'imprécision des mesures liées au GPS, les trajectoires originales suivent d'assez près le tracé des routes. Elles ont donc des longueurs proches des longueurs des chemins qui leur sont associés sur le réseau routier. La figure 3.5 montre la distribution des valeurs de notre ratio pour les trajectoires qui n'ont pas été exclues en raison de trop fortes incohérences.

La distribution des valeurs suit une loi proche d'une gaussienne asymétrique et fortement piquée, de moyenne $\mu = 0.977$ et d'écart type $\sigma = 0.187$. Cette moyenne proche de 1 confirme notre intuition concernant la proximité entre les longueurs des trajectoires originales et celles recalées. Au vu de ces données statistiques, nous décidons de considérer comme potentiellement erronées les trajectoires présentant des ratios en dehors de l'intervalle [0,79 ; 1,16] ce qui correspond à la moyenne plus ou moins une fois l'écart-type ($\mu \pm \sigma$). La prise en compte de ce critère, nous amènerait à vérifier 136549 trajectoires soit moins de 10% des trajectoires obtenues après le map-matching.

12. `https://www.qgis.org/fr/site/`

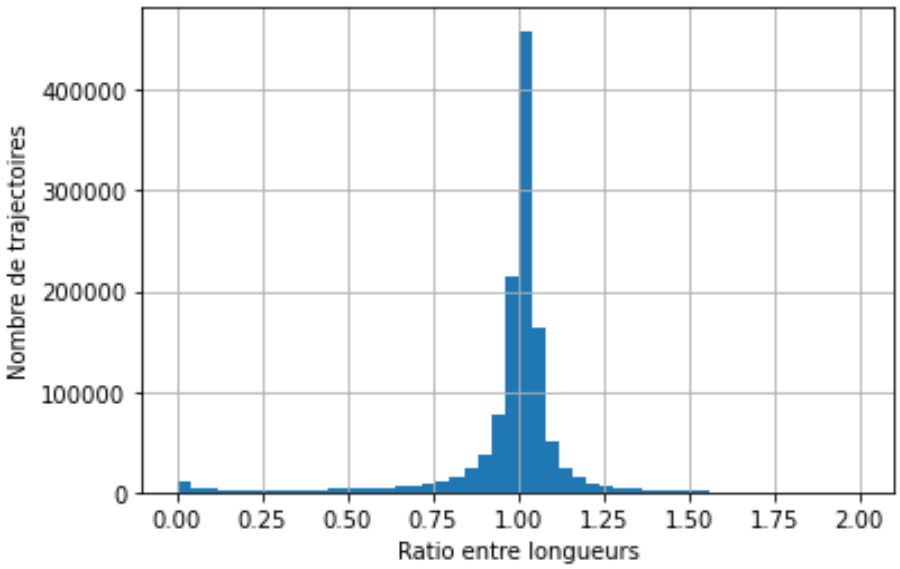


FIGURE 3.5 – Distribution des valeurs du ratio entre les longueurs des trajectoires matchées et leurs versions originales.

3.4.1 Correction des résultats du map-matching

Les erreurs couramment observées au niveau des trajectoires recalées sont de trois principaux types : les discontinuités, les boucles et les détours. Une discontinuité correspond à une cassure dans la séquence de la trajectoire recalée. Cette cassure peut être due à une coupure dans la suite de coordonnées GPS du fait du passage dans un tunnel par exemple ou à un mauvais découpage des trajectoires brutes causant la mise en commun de trajectoires aux tracés non concordants, ou encore à un manque de données au niveau du réseau routier. Une boucle se caractérise par la présence d'un circuit dans la suite de segments formant la trajectoire matchée qui ne correspond pas au tracé indiqué par les points GPS. Les demi-tours sont également considérés comme des boucles. Un détour se manifeste par le choix d'un chemin alternatif entre deux points d'une trajectoire. La particularité (comme pour les boucles) étant que ce chemin diffère du trajet suggéré par les points GPS. Un exemple de discontinuité au sein d'une trajectoire recalée (entre les points 1 et 2) est présenté à la figure 3.6. La figure 3.7 illustre quant à elle l'apparition de boucles et détours après le map-matching. Plus précisément :

— Entre les points 1 et 2 la trajectoire matchée emprunte un détour comparativement au tracé de la trajectoire originale. Cela provient du fait que cette

FIGURE 3.6 – Illustration de la présence de discontinuité après le map-matching.

dernière passe par un tunnel ce qui coupe l'enregistrement des coordonnées GPS.

— Au niveau du point 3 on observe une boucle qui en réalité n'a pas lieu d'être. Deux autres boucles incohérentes sont présentes entre les points 4 et 5 et du point 6 à la fin de la trajectoire.

Les trajectoires contenant des discontinuités sont facilement détectées par une recherche du plus court chemin entre leur origine et leur destination. Le graphe utilisé pour cette recherche de plus court chemin est constitué uniquement des points (nœuds) et segments (arêtes) formant la trajectoire obtenue après le map-matching. Ainsi, si une trajectoire contient une ou plusieurs discontinuités, la recherche du plus court chemin renverra un résultat vide ou une erreur car le graphe utilisé n'est pas connexe.

Les trajectoires qui passent le test de détection des discontinuités peuvent tout de même contenir des boucles ou des détours. Parmi ces deux types d'erreurs, les plus fréquemment observées sont les boucles. Ces dernières peuvent être dé-

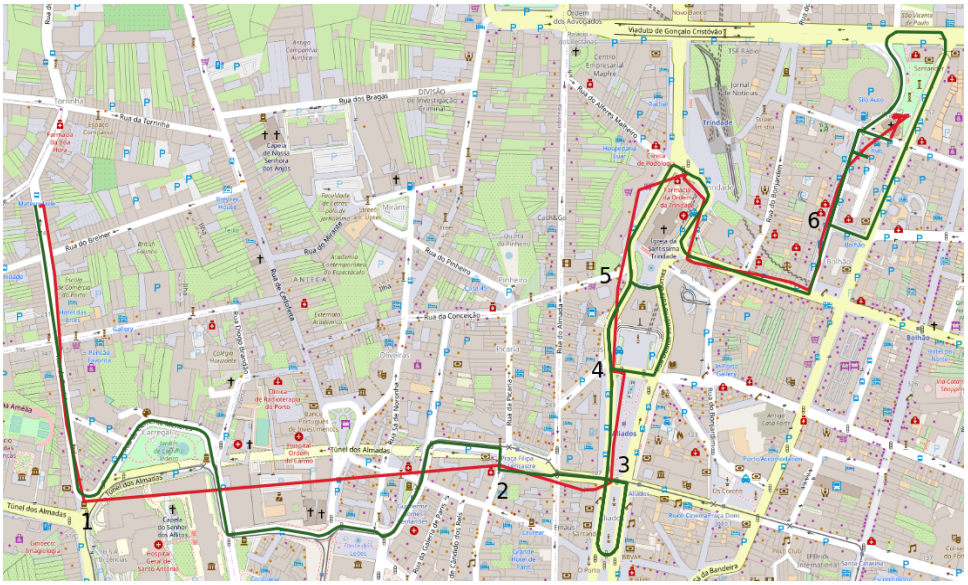


FIGURE 3.7 – Illustration de la présence de boucles et détours après le map-matching. La trajectoire originale est en rouge et celle recalée en vert.

tectées et corrigées automatiquement dans la majorité des cas. A cet effet, nous proposons l'algorithme 1 pour le post-traitement des trajectoires. Ce dernier recherche séquentiellement les différentes boucles présentes dans une trajectoire, puis différencie celles qui sont légitimes (correspondant effectivement au tracé de la trajectoire brute) de celles qui ne le sont pas. Seules les boucles légitimes sont alors conservées. Notre approche consiste principalement à parcourir la trajectoire recalée jusqu'au niveau où se présente une boucle puis à évaluer la vitesse « observée » le long de cette boucle. Nous comparons ensuite cette vitesse à la vitesse moyenne limite correspondant à la boucle sur le réseau routier. La vitesse moyenne limite est égale à la somme pondérée des vitesses maximales des différents segments routiers constituant la boucle. La pondération de chaque vitesse maximale est égale au ratio entre la longueur du segment routier auquel elle s'applique et la longueur totale de la boucle. Nous considérons une boucle comme légitime si la vitesse « observée » est inférieure à une fraction donnée de la vitesse limite moyenne. Plusieurs arguments plaident en faveur de ce critère portant sur la vitesse :

— en premier lieu, l'obligation faite aux automobilistes de respecter les prescriptions du code de la route et par conséquent les limitations de vitesse ;

— en second lieu, la signalisation et la circulation routières, principalement les

feux aux intersections et les interactions avec les autres usagers de la route qui justifient également la nécessité de ralentir que ce soit en ligne droite ou avant et durant un virage ;

— en troisième lieu, l'action de la force centrifuge lors d'un virage ainsi que la topologie des réseaux routiers urbains où les tournants sont souvent à angle droit. Ces facteurs incitent voire obligent naturellement les automobilistes à ralentir.

Une boucle étant constituée habituellement d'une série de virages, elle doit être normalement parcourue à une vitesse moins élevée que celle pouvant être observée le long d'un trajet de même longueur sans virages. Par ailleurs, la vitesse sur le réseau routier étant elle-même limitée et pouvant varier d'un segment à une autre, il peut être généralement admis que la vitesse dans une boucle soit nettement inférieure à la vitesse limite moyenne le long de celle-ci.

En complément de la vitesse observée, l'algorithme proposé exploite deux autres paramètres que sont la longueur des boucles et le nombre de points les composant. Ces paramètres ont été sélectionnés empiriquement car leur prise en compte permet d'affiner la sélection des boucles légitimes. In fine, notre solution algorithmique repose sur l'utilisation de trois paramètres : le facteur de réduction de la vitesse limite moyenne, la longueur minimale d'une boucle et le nombre de points minimum devant la composer. Nous avons implémenté et testé notre solution sur des cas de trajectoires ayant déjà subi le map-matching en fixant les valeurs des paramètres à partir de données issues d'une correction de quelques trajectoires. Les résultats obtenus sont prometteurs comme le montre la figure 3.8. Toutefois, pour une application à grande échelle, il importe de fixer plus précisément les valeurs des paramètres. Une solution pour y parvenir, consiste à annoter manuellement un échantillon conséquent de trajectoires déjà recalées. L'objectif étant de faire identifier par des opérateurs humains les boucles légitimes, afin d'en extraire les valeurs les plus appropriées pour les paramètres. La plateforme conçue à cette fin, est présentée dans la sous-section suivante.

Concernant la détection et la correction des détours, elle peut être réalisée en exploitant des données historiques de map-matching validées par des opérateurs humains. Il faudrait alors disposer de trajectoires très similaires (partageant si possible une grande partie de leurs parcours) avec la trajectoire en cours de traitement. La stratégie consisterait alors à comparer la trajectoire nouvellement recalée aux données historiques puis via un système de vote à sélectionner la solu-

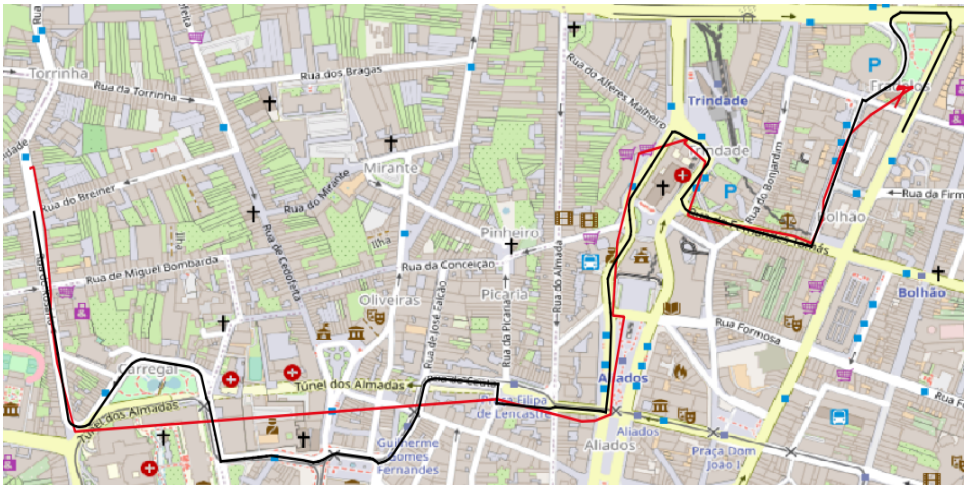


FIGURE 3.8 – Illustration de la correction des boucles illégitimes dans la trajectoires matchée de la figure 3.7. En rouge la trajectoire originale et en noir la trajectoire matchée après correction.

tion la plus appropriée. C'est une solution inspirée de celle proposée par [XJJF19] et décrite dans l'état de l'art. La mise en œuvre de cette solution nécessite de compléter notre plateforme d'annotation des trajectoires présentée ci-dessous afin de prendre en compte la correction complète de ces dernières. Nous n'avons toutefois pas atteint ce niveau de développement dans nos travaux.

3.4.2 Outil d'annotation des trajectoires

En vue d'annoter les trajectoires, nous avons développé une plateforme web permettant de les visualiser et de les corriger. Les trajectoires y sont affichées successivement sur une carte numérique. Pour chacune d'elles, la version originale constituée des points GPS et la version obtenue après map-matching sont projetées sur la carte. L'opérateur peut à tout moment choisir d'afficher ou de cacher une des deux versions. Les boucles contenues dans la version matchée sont identifiées automatiquement. Lorsqu'une boucle est sélectionnée, elle est encadrée par un rectangle de délimitation (bounding box en anglais), l'opérateur compare alors le tracé original à celui généré après le map-matching et décide si la boucle est légitime ou non. Si la boucle n'est pas légitime ou si elle l'est mais ne correspond pas au tracé original, il propose un tracé qui corrige l'erreur observée. Une fois la correction validée, une autre boucle présente dans la base est sélectionnée et

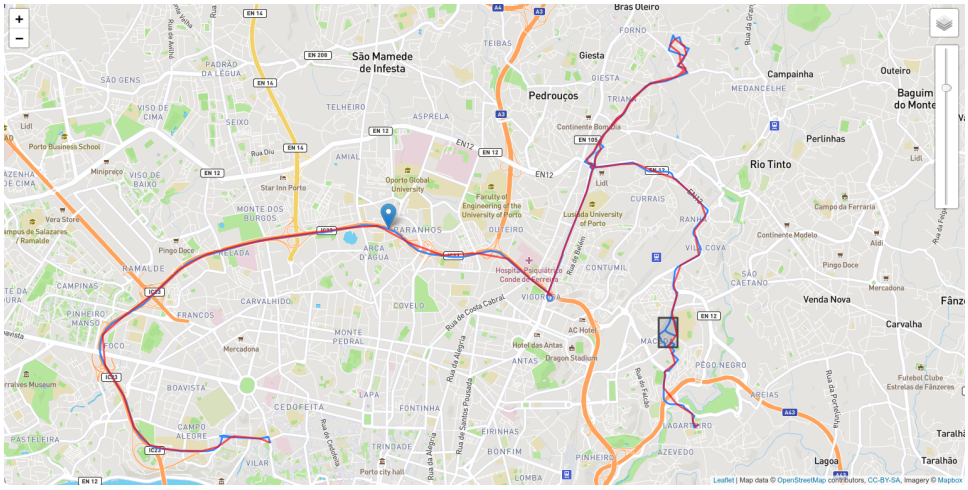


FIGURE 3.9 – Plateforme pour l'annotation des trajectoires. En bleu la trajectoire matchée, en rouge la trajectoire originale, le rectangle gris est la bounding box de la boucle sélectionnée. En haut à droite sont proposés des boutons permettant d'afficher et cacher les différentes trajectoires ainsi qu'un curseur, synchronisé avec le marqueur bleu, permettant de parcourir la trajectoire recalée. La plateforme permet toutes les interactions classiques de zoom et de déplacement.

affichée. Les résultats sont enregistrés dans une base de données conçue à cet effet. Dans certains cas il est difficile de comprendre le tracé de la trajectoire recalée. Aussi, afin d'aider l'opérateur, un marqueur de position dynamique est mis à disposition. Il est contrôlé par un curseur et peut être déplacé de l'origine de la trajectoire à sa destination. Une illustration de la plateforme est proposée à la figure 3.9.

L'utilisation de cet outil permet de voir la diversité des boucles pouvant apparaître dans les trajectoires, juste après le recalage. La figure 3.10 montre 4 exemples réels de boucles identifiées automatiquement sur la trajectoire de la figure 3.9 :

❶ Sur la boucle 1 le taxi arrive de la gauche, sort de la voie rapide puis va vers le nord. Cela ressemble donc à une boucle même si en pratique, comme il s'agit d'un échangeur, une route se trouve en dessous de l'autre. Cette boucle est donc légitime.

❷ Sur la boucle 2, le taxi arrive de la droite, emprunte la boucle externe puis reprend la route centrale qui part vers le sud. En pratique la route centrale est un tunnel et, au vu de la trajectoire rouge et en l'absence de point GPS sur

ou proche de la boucle, il est très probable que cette dernière n'ait jamais eu lieu. En allant voir la configuration réelle des lieux (par exemple en utilisant le mode *street view* de *Google Maps*), on peut confirmer par ailleurs que rien n'impose cette boucle. Cette boucle est donc très probablement illégitime et le trajet direct est le trajet correct.

❸ Sur la boucle 3, le taxi arrive du sud, effectue la boucle dans le sens horaire puis repart vers le sud. Cette boucle se trouve tout en haut de la figure 3.9 et il semble que le taxi ait fait un grand détour, qui se termine par cette boucle. A priori, vu le trajet complet, cette boucle semble légitime et pourrait correspondre à un arrêt intermédiaire. En creusant un peu plus on peut noter que le taxi fait un petit détour qui passe par les marqueurs 1 puis 2. De fait, les marqueurs GPS (rouges) sont un peu décalés vers le haut et il y a deux marqueurs GPS proches au niveau du 2. Cela pourrait laisser penser que le taxi a effectivement fait ce petit détour et a du s'arrêter pendant au moins 15 secondes au carrefour 2. Il est cependant très difficile d'avoir la certitude que le taxi a effectivement emprunté ce petit détour. Aller étudier les lieux n'apporte pas d'aide car la route sous le marqueur 1 a été fermée depuis. Que l'on corrige ou pas ce tout petit détour n'aura probablement que très peu d'impact sur les études ultérieures, un opérateur humain laissera donc probablement la boucle recalée en l'état.

❹ Enfin, sur la boucle 4, le taxi arrive du nord, effectue la boucle indiquée par les marques 1, 2, 3, fait une petite boucle 4, 5 et revient sur le trajet initial en 6 avant de terminer en 7. Cette boucle semble très artificielle, et ne correspond pas à la vision que l'on a des mesures GPS. Les marqueurs GPS n'indiquent, par ailleurs pas, qu'il y ait eu une boucle à cet endroit. Il est donc clair que cette portion de trajectoire est recalée de manière incorrecte. La figure 3.11 illustre le correctif qui pourrait être apporté par un humain à l'aide de la plateforme proposée.

Algorithme 1 : Post-traitement de trajectoires recalées

Entrées :

T **:** Tableau de segments routiers représentant la trajectoire matchée

reduc_vit : facteur de réduction de la vitesse limite moyenne

long_min_bcl : longueur minimale des boucles à traiter

min_pts_bcl : nombre minimal de sommets des boucles à traiter

Sorties :

TR **:** Tableau de segments routiers de la trajectoire corrigée

début

$TR \leftarrow []$

$sous\_traj \leftarrow []$

pour *i allant de 1 à |T|* **faire**

$sous\_traj \leftarrow sous\_traj \cup T[i]$

si *contient_bcl(sous_traj)* **alors**

// recherche de la boucle en allant à rebours

$j \leftarrow |sous\_traj| - 1$

tant que $j \geq 1$ *et* $!(contient\_bcl(sous\_traj[j : |sous\_traj|]))$ **faire**

$j \leftarrow j - 1$

fin

// la partie avant la boucle est ajoutée à la sortie

$pre\_bcl \leftarrow sous\_traj[1 : j - 1]$

$TR \leftarrow TR \cup pre\_bcl$

// la boucle est extraite

$bcl \leftarrow sous\_traj[j : |sous\_traj|]$

// vérification de la longueur et du nombre de points

si *longueur(bcl) > long_min_bcl et nb_pts(bcl) > min_pts_bcl* **alors**

$vit\_lim \leftarrow vit\_lim\_moy(bcl)$

$vit\_reel \leftarrow longueur(bcl)/tps\_parcours(bcl)$

// si la vitesse est cohérente, la boucle est conservée

si *vit_reel <vit_lim * reduc_vit* **alors**

$TR \leftarrow TR \cup bcl$

fin

fin

$sous\_traj \leftarrow []$

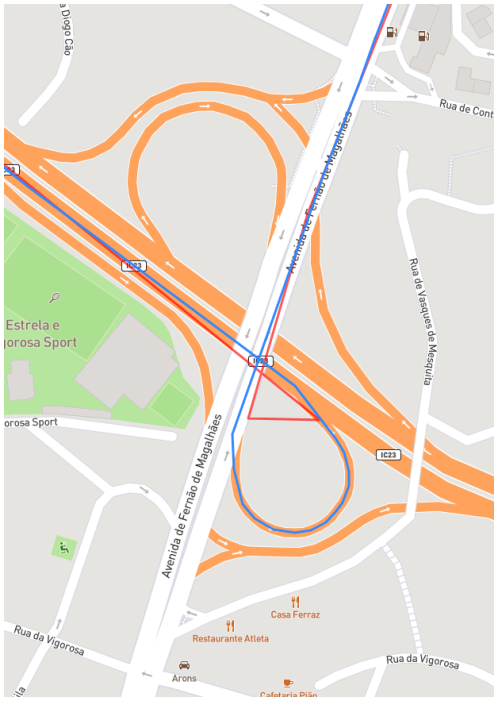
fin

fin

$TR \leftarrow TR \cup sous\_traj$

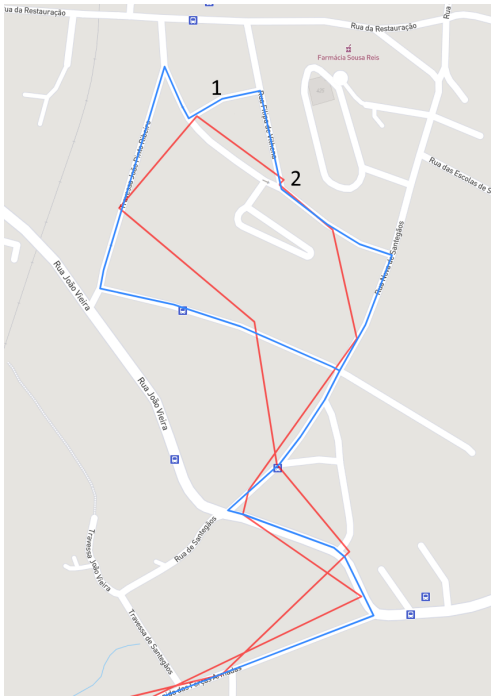
retourner TR

fin



(a) Boucle 1

(b) Boucle 2



(c) Boucle 3

(d) Boucle 4

FIGURE 3.10 – Quatre boucles identifiées après le map-matching.

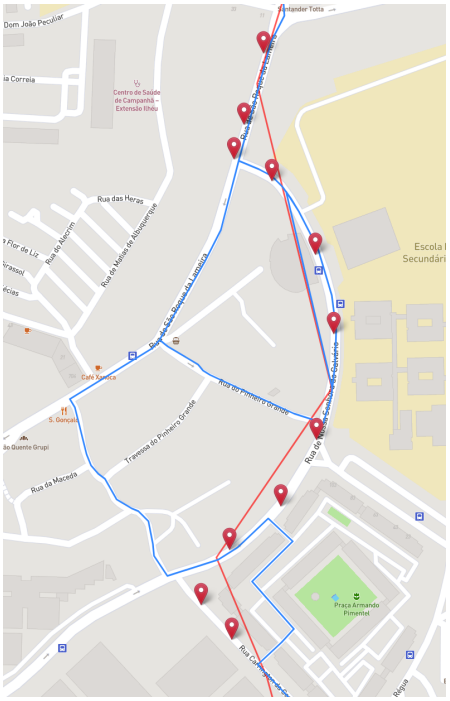


FIGURE 3.11 – Proposition de correction de la boucle 4 par un humain. Les marqueurs sont disposés par le correcteur le long du trajet présumé correct. Ces marqueurs permettent d'identifier sans ambiguïté les segments de route. La plateforme n'autorise pas le placement de marqueurs en dehors des routes pour éviter toute erreur.

3.5 Conclusion

Nous avons présenté dans ce chapitre les jeux de données, principalement les trajectoires et le réseau routier, utilisés pour l'ensemble de nos travaux. Nous avons décrit les étapes nécessaires au prétraitement des trajectoires, les choix fait pour l'implémentation de l'algorithme de map-matching retenu, ainsi que les post-traitements effectués sur les trajectoires recalées et le système de gestion utilisé pour leur stockage.

Concernant plus spécifiquement le post-traitement, nous avons proposé une méthode de détection des discontinuités ainsi qu'un algorithme pour l'identification et la correction des boucles qui sont les erreurs les plus fréquentes sur les trajectoires matchées. Afin d'ajuster au mieux les paramètres de notre algorithme, nous avons également développé une plateforme d'annotation des trajectoires qui permet de visualiser et de corriger les trajectoires déjà recalées.

In fine, il faut garder à l'esprit, qu'on ne saurait concevoir un système entièrement automatisé et totalement infaillible pour le map-matching des trajectoires. Nous pensons que la complexité de cette tâche et les importants volumes de données en jeu imposent de définir des mesures de qualité permettant d'identifier mécaniquement et à grande échelle les trajectoires contenant de potentielles erreurs de traitement. Une fois celle-ci identifiées, il faudrait leur faire subir un post-traitement partiellement basé sur l'intervention d'opérateurs humains pour s'assurer d'une correction aussi optimale que possible.

Chapitre 4

EQRP, une distance hybride pour trajectoires contraintes par un réseau

Sommaire

4.1 Introduction

L'un des objectifs essentiels de l'analyse des trajectoires est de quantifier le degré de similarité ou de différence entre trajectoires par le biais de mesures dédiées. La poursuite de cet objectif s'est traduite, dans la littérature, par la définition d'un ensemble de distances entre trajectoires, aussi bien en espace libre qu'en espace contraint par un réseau. Dans l'optique de répertorier et de comparer les différentes distances proposées, des articles de revue [SLZ+20, MSMEB15] ont été édités mais ils ne traitent, à notre connaissance, que des distances entre trajectoires en espace libre ignorant celles spécifiques aux espaces contraints par un réseau. C'est pour pallier ce manque que nous avons proposé dans notre état de l'art une brève revue des distances entre trajectoires contraintes par un réseau. Dans ce chapitre, nous allons plus loin en effectuant une analyse comparative des distances les plus pertinentes. Nous proposons également une nouvelle distance d'édition pour trajectoires qui considère une trajectoire à la fois comme un ensemble et une séquence de nœuds et dont les performances dépassent celles des autres distances intégrant des distorsions.

L'état de l'art présenté à la section 2.4 du chapitre 2 a fait ressortir une classification des distances entre trajectoires contraintes par un réseau en quatre grandes familles : les distances simples, les distances de type nœud à trajectoire, les distances ensemblistes et celles intégrants les distorsions. Les distances simples comparent les trajectoires en évaluant l'éloignement entre nœuds de même indice ou en fonction d'un ensemble de points d'intérêts. Elles sont sensibles aux moindres déformations (variations dans la forme) des trajectoires et à l'ensemble de points d'intérêts utilisés. Les distances de type nœud à trajectoires déterminent plutôt les espacements minimaux entre les nœuds d'une première trajectoire et l'ensemble des nœuds formant une seconde. Elles parviennent ainsi à mesurer la dissimilarité entre trajectoires tout en étant moins sujettes aux déformations de ces dernières. Les distances ensemblistes comparent quant à elles deux trajectoires à partir de la taille de leur intersection, c'est-à-dire le nombre de nœuds ou de segments qu'elles partagent. Elles ne peuvent donc comparer des trajectoires totalement disjointes. Contrairement aux trois familles précédentes, les distances intégrant les distorsions permettent de tenir compte de l'ordre des nœuds dans les trajectoires. Elles recherchent l'alignement optimal entre les nœuds des trajectoires à comparer. Elles sont à même d'intégrer les décalages temporels locaux qui se manifestent par la présence de sous séquences de nœuds très similaires mais décalées les unes par

rapport aux autres, dans leur calcul. Ces particularités apportent une dimension supplémentaire dans le processus de comparaison entre trajectoires qui n'est pas présente dans les distances des trois autres familles.

Au sein des distances intégrant les distorsions, les distances d'édition, dont les plus connues sont *EDR* et *ERP*, forment une sous-classe et se singularisent par leur mécanique de calcul. En effet, leur principe est d'évaluer le coût total minimal des opérations d'édition de nœuds (insertion, suppression et substitution), nécessaires pour transformer une trajectoire en une autre. En plus des aptitudes inhérentes à l'ensemble des distances intégrant des distorsions, les distances d'édition ont la particularité d'être à appariement complet. Autrement dit, tous les nœuds sont utilisés dans le processus de calcul ce qui n'est pas le cas pour les autres distances intégrant les distorsions. De plus, les distances d'édition peuvent être adaptées pour répondre à des besoins spécifiques en ajustant les coûts des opérations d'édition dont elles dépendent et les stratégies d'appariement entre nœuds.

En dépit de leurs qualités manifestes, les distances d'édition recèlent également des insuffisances. *EDR*, par exemple, dépend d'un seuil d'appariement dont la valeur optimale est souvent difficile à fixer. De plus, elle utilise des sous-coûts fixes, non-corrélés aux distances réelles entre nœuds, pour l'ensemble des opérations d'édition. Ce qui impacte négativement sa capacité à évaluer précisément les distances entre trajectoires. *ERP*, pour sa part, est définie en fonction de distances réelles entre nœuds mais requiert la spécification d'un nœud de référence dans l'évaluation des distances entre trajectoires en cas d'absence d'appariement (concept de gap). Or, on maîtrise mal l'impact de l'usage d'un nœud de référence sur la précision de *ERP* en espace graphe. Par ailleurs, l'usage de distances réelles entre nœuds rend *ERP* sensible à la présence de déformations dans les trajectoires.

En somme, il n'existe pas de panacée pour la comparaison de trajectoires et il faut choisir dès lors la distance la plus appropriée en fonction de l'application visée. Cela dit, il ressort de la littérature que globalement les insuffisances d'une distance sont compensées par les avantages d'autres distances et vice versa. Partant de cette observation nous nous sommes posé la question suivante : est-il possible d'hybrider deux distances qui se compensent mutuellement afin d'obtenir une distance qui réunit leurs qualités tout en inhibant leurs insuffisances ? Dans ce chapitre nous apportons une réponse positive à cette question. Pour y parvenir, nous avons fait le choix de combiner une distance intégrant les distorsions et une distance de type nœud à trajectoire. Ce choix n'est pas anodin car les distances intégrant

les distorsions sont les seules à tenir compte explicitement de l'ordre des nœuds et à gérer les décalages temporels locaux mais elles peuvent être sensibles aux déformations tandis que les distance de type nœud à trajectoire ne tiennent pas compte de l'ordre des nœuds ou des distorsions mais sont moins sensibles aux déformations. Plus précisément nous retenons *ERP* car c'est une distance d'édition qui utilise les distances réelles entre nœuds mais qui est sensible aux déformations et la distance de *Hausdorff* parce qu'elle est relativement moins sujette à l'effet des déformations.

Dans la suite du chapitre, la première section est consacrée à la définition de la distance hybride d'édition *EQRP* (*Edit distance with Quasi Real Penalities*). La seconde décrit la méthodologie adoptée pour la comparaison de quelques distances pour trajectoires contraintes par un réseau y compris celle nouvellement proposée. Les deux dernières sections présentent l'implémentation et les résultats de nos expérimentations ainsi que la discussion que nous proposons relativement à ces résultats.

4.2 Edit distance with Quasi Real Penalities (EQRP)

La mesure que nous proposons diffère des distances d'édition existantes par la façon dont elle traite les nœuds pouvant ou ne pouvant être appariés et pour lesquels il est préférable de procéder à une insertion ou une suppression dans l'une des trajectoires. Plus précisément :

— lorsqu'il y a appariement on conserve la distance réseau réelle entre paire de nœuds ;

— en l'absence d'appariement, on fixe le coût d'insertion ou de suppression à la distance de *Hausdorff* entre les trajectoires comparées. Nous remplaçons ce faisant le coût fixe de *EDR* (valant toujours 1 quelles que soient les trajectoires comparées) par un coût semi-fixe, c'est-à-dire qui n'est figé que pour une paire de trajectoires et qui prend pour valeur la distance de *Hausdorff* entre celles-ci. Ce choix nous permet également de ne plus recourir à l'utilisation d'un nœud de référence.

On obtient ainsi une distance d'édition qui allie le meilleur des deux mondes en considérant à la fois les trajectoires comme des séquences de nœuds et comme des ensembles de nœuds. Cette définition permet de fixer indirectement le seuil d'appariement à la distance de *Hausdorff* entre trajectoires.

La nouvelle distance d'édition que nous introduisons est nommée Edit distance with Quasi Real Penalities ($EQRP$). Elle se définit formellement entre deux trajectoires T_1 et T_2 de tailles respectives $|T_1| = n$ and $|T_2| = m$ par la formule suivante :

$$EQRP(T_1,T_2) = \begin{cases} 0, \ Si\ n = 0\ et\ m = 0 \\ \infty, \ Si\ n = 0\ ou\ m = 0 \\ min \begin{cases} EQRP(Rest(T_1), Rest(T_2)) + d_G(Head(T_1), Head(T_2)) \\ EQRP(Rest(T_1), T_2) + Hausdorff(T_1,T_2) \\ EQRP(T_1, Rest(T_2)) + Hausdorff(T_1,T_2) \end{cases} & Sinon \end{cases} \tag{4.1}$$

Avec $d_G(Head(T_1), Head(T_2))$, la distance réseau entre les nœuds $Head(T_1)$ et $Head(T_2)$ et $Hausdorff(T_1,T_2)$ la distance de *Hausdorff* entre les trajectoires T_1 et T_2.

Telle que nous venons de la définir, la distance $EQRP$ ne respecte pas l'inégalité triangulaire et n'est donc pas métrique. Il est toutefois possible de la rendre métrique en modifiant légèrement sa formulation de la manière suivante :

$$EQRP(T_1,T_2) = \begin{cases} 0, \ Si\ n = 0\ et\ m = 0 \\ \infty, \ Si\ n = 0\ ou\ m = 0 \\ min \begin{cases} EQRP(Rest(T_1), Rest(T_2)) + d_G(Head(T_1), Head(T_2)) + D \\ EQRP(Rest(T_1), T_2) + Hausdorff(T_1,T_2) + D \\ EQRP(T_1, Rest(T_2)) + Hausdorff(T_1,T_2) + D \end{cases} & Sinon \end{cases} \tag{4.2}$$

La différence tient à l'ajout du diamètre D du réseau aux divers sous-coûts. Cet ajustement permet de s'assurer que $EQRP$ respecte l'inégalité triangulaire. C'est-à-dire que pour tout triplet de trajectoires distinctes R, S et T on a :

$$EQRP(R,S) \leq EQRP(R,T) + EQRP(T,S) \tag{4.3}$$

En effet pour qu'une distance d'édition respecte cette inégalité, il suffit que la distance entre éléments sur laquelle elle se base la respecte également même en cas de gap [WSB76]. Or, en ajoutant le diamètre aux différents sous-coûts on s'assure que l'inégalité triangulaire est respectée pour tout triplet de nœuds r, s et t.

Preuve : Lorsqu'il n'y a pas de gap, on se retrouve dans le cas trivial du plus

court chemin entre nœuds qui respecte l'inégalité triangulaire. A l'inverse, si un des nœuds est un gap, l'inégalité demeure respectée car la distance de *Hausdorff* est nécessairement inférieure ou égale au diamètre du réseau.

A l'instar des autres distances intégrant des distorsions, *EQRP* peut être calculée en utilisant un algorithme de programmation dynamique pour résoudre sa formule de récurrence. Ce mode de calcul a une complexité de l'ordre de $\mathcal{O}(nm(|V|+|E|))$ avec n et m, les tailles respectives des trajectoires comparées et $(|V|+|E|)$ le coût du calcul du plus court chemin entre deux nœuds. Cette complexité intègre le coût du calcul de la distance de *Hausdorff* (qui est pré-calculée avant la résolution de la formule de récurrence) et dont la complexité est aussi de l'ordre de $\mathcal{O}(nm(|V|+|E|))$.

4.3 Méthodologie

Après avoir défini formellement notre nouvelle mesure de distance, il importe de nous assurer de sa pertinence en la comparant à d'autres distances pour trajectoires contraintes par un réseau. Toutefois, comparer des distances n'est pas trivial car les valeurs renvoyées par différentes distances pour une même paire de trajectoires ne sont pas toujours cohérentes dans la mesure où chaque distance a une sémantique propre. Il n'est donc pas adéquat de se baser directement sur les valeurs générées par les distances pour les comparer. Il faut procéder dès lors par des approches indirectes. Dans cette section, nous décrivons les méthodes que nous avons développées et adaptées spécifiquement pour comparer les distances pour trajectoires contraintes par un réseau, en l'occurrence la comparaison par évaluation de la résistance aux transformations et celle par clustering.

4.3.1 Comparaison par transformations

La méthode de comparaison par transformation a été initialement proposée par Han et al dans leur revue [SLZ+20]. Elle vise à comparer des distances entre trajectoires libres, en évaluant leurs aptitudes à résister à diverses transformations. Ces dernières, au nombre de six, se déclinent comme suit :

— l'ajout de points d'échantillonnage ;

— la suppression de points d'échantillonnage ;

— le ré-échantillonnage de trajectoire ;

— la dilatation et compression temporelles ;

— la dilatation et compression spatiales ;

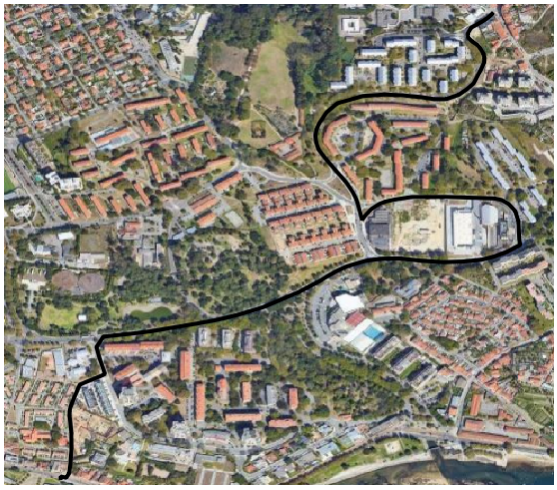
— l'ajout de bruit.

L'ajout de points d'échantillonnage a pour but d'augmenter le nombre de points constituant la trajectoire tandis que la suppression de tels points en réduit la quantité. L'objectif étant simplement d'évaluer l'impact, sur les distances, de la variation du nombre de points constituant les trajectoires, ces deux transformations étant réalisées de sorte à ne pas modifier la forme générale des trajectoires. Pour garantir cela, les auteurs identifient les points d'articulation des trajectoires, points qui déterminent l'allure générale de la trajectoire, et qu'il faut donc veiller à ne pas modifier. Le ré-échantillonnage d'une trajectoire, vise à faire varier l'intervalle de temps entre les points consécutifs. C'est une transformation qui affecte uniquement la dimension temporelle des trajectoires. Les opérations de dilatation et de compression temporelles permettent de faire varier la vitesse d'une trajectoire en augmentant ou en réduisant respectivement l'intervalle de temps entre le premier point et le dernier point de la trajectoire. C'est une transformation proche du ré-échantillonnage de trajectoire et qui comme cette dernière n'affecte que la dimension temporelle. La dilatation et la compression spatiale font agrandir ou rétrécir les trajectoires sans que leur forme ne change. Leur effet est comparable à celui d'une loupe que l'on rapproche ou que l'on éloigne d'un objet. Enfin, l'ajout de bruit permet de déformer spatialement les trajectoires en modifiant plus ou moins la position de certains de leurs points. Cela permet de reproduire l'effet des imprécisions liées à l'utilisation du GPS.

Dans la méthode que nous proposons, nous adoptons la même approche que Han et al. pour comparer les distances entre trajectoires contraintes par un réseau. Toutefois nous ne prenons pas en compte explicitement les données temporelles des trajectoires, car les distances comparées ne les intègrent pas, donc les transformations relatives exclusivement au domaine temporel sont écartées (dilatation et la compression temporelles, ré-échantillonnage de trajectoire). De plus, les trajectoires étant modélisées comme des séquences de nœuds du réseau routier, il n'est pas possible d'ajouter des points d'échantillonnage sans modifier le réseau routier à chaque fois. Les contraintes du réseau routier rendent également très difficile l'agrandissement ou le rétrécissement des trajectoires qui s'y déroulent sans en modifier la forme. En raison de ces limitations nous avons décidé de ne pas inclure dans nos expérimentations les transformations suivantes : ajout de points

d'échantillonnage et dilatation et compression spatiales.

In fine, nous ne retenons que deux des transformations initialement proposées à savoir : la suppression de points d'échantillonnage et l'ajout de bruit qui sont elles mêmes adaptées aux spécificités des trajectoires contraintes par un réseau. Ainsi, nous requalifions la suppression de points d'échantillonnage en suppression de nœuds et nous déclinons l'ajout de bruit en deux versions : l'ajout de boucles et l'ajout de détours. Les nouvelles transformations sont décrites dans les sous-sections à suivre et illustrées à la figure 4.1.



(a) Trajectoire originale

(b) Trajectoire après suppression de nœuds

(c) Trajectoire après ajout de boucles

(d) Trajectoire après ajout de détour

FIGURE 4.1 – Illustrations des transformations effectuées sur les trajectoires.

4.3.1.1 Suppression de nœuds

Dans le contexte des trajectoires contraintes par un réseau, la suppression de points d'échantillonnage revient à retirer certains nœuds de la séquence formant les trajectoires. Tout comme pour les trajectoires libres, nous veillons à ne pas altérer la forme des trajectoires contraintes par un réseau. Pour ce faire, nous identifions avant toute suppression ce que nous appelons les nœuds pivots (nœuds au niveau desquels la trajectoire change sensiblement de forme) au sein des trajectoires, en utilisant l'algorithme de Ramer Douglas Peuchker (RDP) [HG97]. Le principe de cet algorithme, décrit ci-dessous, est de remplacer une polyligne (une ligne brisée) constituée de plusieurs points par une ligne simple si la distance entre la droite reliant les extrémités de la polyligne et son point le plus éloigné est inférieure à un seuil prédéfini.

Algorithme 2 : RDP(ListeDePoints, epsilon)

Entrées : Tableau de points représentant la polyligne $ListeDePoints$,
valeur de seuil $epsilon$
Sorties : Tableau de points contenant les points d'articulation $Resultat$
$dmax \leftarrow 0$
$idx \leftarrow 0$
$taille \leftarrow longueur(ListeDePoints)$
pour *i allant de 2 à taille -1* **faire**
  $d \leftarrow$
   $distance(ListeDePoints[i], Ligne(ListeDePoints[1], ListeDePoints[taille]))$
  si $d > dmax$ **alors**
    $idx \leftarrow i$
    $dmax \leftarrow d$
  fin
fin
si $dmax > epsilon$ **alors**
  $resultatPartiel1 \leftarrow RDP(ListeDePoints[1 : idx], epsilon)$
  $resultatPartiel2 \leftarrow RDP(ListeDePoints[idx : taille], epsilon)$
  $Resultat \leftarrow Concatenation(resultatPartiel1, resultatPartiel2)$
sinon
  $Resultat \leftarrow [ListeDePoints[1], ListeDePoints[taille]]$
fin
retourner $Resultat$

Afin de traiter les trajectoires contraintes par un réseau comme des polylignes, nous ajoutons les coordonnées GPS comme attributs aux nœuds. Une fois les nœuds pivots identifiés, nous supprimons aléatoirement des nœuds parmi ceux restants. Le nombre de nœuds supprimés dépend d'un ratio r dont les valeurs sont comprises entre 0 et 1. Pour une trajectoire T, de taille n, possédant k nœuds pivots, le nombre de points à supprimer est donné par l'équation :

$$NB_{pts} = \min\{n * r, n - k\} \tag{4.4}$$

Cette équation implique que l'on ne peut supprimer, au maximum, que l'ensemble des nœuds qui ne sont pas des nœuds pivots.

4.3.1.2 Ajout de boucles et de détours

L'ajout de bruit à une trajectoire contrainte par un réseau diffère du cas d'une trajectoire en espace libre. En espace libre, cette transformation se traduit par l'ajout d'un bruit gaussien aux coordonnées des points, tandis que dans l'espace du réseau routier, elle correspond à l'ajout de déviations dans le parcours de la trajectoire. La raison est que l'espace du réseau routier est discret ce qui empêche l'ajout des points autres que ceux présents dans l'ensemble des nœuds formant le réseau. Deux types de déviations peuvent être générés : une boucle ou un détour. Une boucle est un circuit dans le réseau routier, une séquence de nœuds commençant et se terminant par le même nœud. Un détour est un chemin alternatif entre deux nœuds distincts d'une même trajectoire. Habituellement, ces deux perturbations se produisent pendant le processus de map-matching lorsque les trajectoires brutes contiennent du bruit ou en raison d'une erreur dans le processus d'assignation des points aux segments routiers. L'objectif est ici, d'essayer de les reproduire artificiellement.

Pour ajouter une boucle, nous choisissons au hasard un nœud u dans la trajectoire et un de ses voisins v (en dehors des nœuds de la trajectoire) dans un rayon prédéfini. Un voisin est donc, dans ce cas, un nœud accessible situé à une distance inférieure au rayon. Nous calculons les chemins les plus courts du nœud u à son voisin v et vice versa. Le chemin obtenu, qui forme une boucle, est ensuite inséré dans la trajectoire au bon endroit : une fois arrivé à u, on parcourt la boucle avant de continuer la trajectoire initiale. Le nombre de boucles à ajouter à une trajectoire est géré par un ratio également compris entre 0 et 1. En multipliant ce

ratio par la taille de la trajectoire on obtient la quantité de nœuds devant porter des boucles.

En ce qui concerne les détours, une procédure similaire à celle utilisée pour ajouter une boucle est adoptée. Dans un premier temps, nous choisissons deux nœuds distincts u et v d'une même trajectoire. Nous appelons le premier nœud dans la séquence de la trajectoire l'origine (supposons que ce soit u) et le second la destination. L'écart entre les deux nœuds est déterminé par un paramètre dit de projection qui équivaut au nombre de nœuds séparant u de v. Plus ce paramètre sera élevé, plus les détours pourront se faire entre une origine et une destination éloignée. Dans un second temps, un nœud intermédiaire, w, est sélectionné parmi les voisins de u situés à une distance inférieure à un rayon prédéfini, w ne pouvant être un nœud de la trajectoire. Par la suite on détermine le chemin le plus court allant de u à v via w et on remplace la sous-séquence de la trajectoire originale entre u et v par celle passant par w. Il faut noter qu'en règle générale le détour n'est pas forcément plus long que le chemin qu'il remplace dans la trajectoire originale, car la sous-trajectoire de u à v n'avait aucune raison d'être un plus court chemin entre ces deux sommets. Le détour est donc juste un chemin alternatif passant par le sommet w.

4.3.1.3 Évaluation de l'impact des transformations

Pour quantifier la robustesse des distances aux transformations, l'approche utilisée est celle décrite dans [SLZ+20]. Elle suit les étapes suivantes :

❶ Choisir au hasard une trajectoire dans le jeu de données original.

❷ Extraire la liste ordonnée des trajectoires les plus proches de la trajectoire choisie dans le jeu de données original. Classer les trajectoires par distance croissante, c'est-à-dire de la plus proche à la plus éloignée. Identifier cette liste comme la *Liste_originale*.

❸ Appliquer les transformations à l'ensemble des trajectoires du jeu de données original hormis la trajectoire initiale. Cette opération génère autant de jeux de données dérivés que de transformations.

❹ Déterminer la liste ordonnée, toujours par distance croissante, des trajectoires les plus proches de la trajectoire choisie dans chacun des jeux de données dérivés. Chacune des listes ainsi générées est une *Liste_transformée*.

❺ Évaluer la similarité entre la *Liste_ originale* et la *Liste_ transformée* correspondante, pour chaque transformation.

L'intuition qui motive cette approche est qu'une distance qui résiste bien aux transformations aura tendance à ordonner de manière similaire les plus proches voisins d'une trajectoire donnée, aussi bien dans le jeu de données orignal que dans les jeux de données dérivés.

L'évaluation de la similarité entre les listes ordonnées de plus proches voisins est faite en usant du coefficient de corrélation de Spearman, encore appelé Rho (ρ) de Spearman. C'est une mesure dédiée à l'estimation de la corrélation entre deux ordonnancements de même taille. Considérons que les ordonnancements correspondent aux valeurs de deux variables X et Y. Elle évalue le rapport entre la covariance desdites variables et le produit de leurs écarts-types respectifs. Formellement elle s'écrit :

$$r_s = \frac{cov(X,Y)}{\sigma_X.\sigma_Y} \tag{4.5}$$

Avec *cov* la covariance et σ l'écart-type.

Les valeurs du ρ de Spearman évoluent dans l'intervalle [-1 ; 1]. Plus la valeur est proche de 1, meilleure est la concordance entre les ordonnancements et par conséquent meilleure est la robustesse des distances face aux transformations. A contrario, une valeur proche de 0 indique une absence de lien entre les deux ordres et donc une faible, ou inexistante, résistance des distances face aux transformations. Une valeur proche de -1 indique une tendance décroissante : les trajectoires proches dans les données originales sont éloignées dans les données dérivées et inversement.

Comparer des distances à partir des corrélations découlant de cette méthode est toutefois source de biais dans le cas des mesures telles que *EDR* et *LCSS* ou de mesures de type ensembliste, en raison de leurs modes de calcul. En effet, ces mesures considèrent toute paire de trajectoires n'ayant aucune paire de nœuds appariés comme étant systématiquement éloignées au maximum. En pratique cela se traduit, par une distance normalisée de valeur 1 dès que les trajectoires sont totalement disjointes. Cela pose un problème de résolution dans la comparaison des trajectoires et biaise l'ordonnancement des plus proches voisins. Plus spécifiquement, on note qu'à partir d'un certain indice, l'ordre des plus proches voisins ne varie plus puisque toutes les trajectoires se retrouvent à distance 1 de la trajectoire

de référence en dépit des transformations. On obtient alors des valeurs de corrélation élevées qui ne reflètent pas l'aptitude de la distance à gérer la transformation courante mais plutôt l'inaptitude de cette dernière à prendre en compte l'effet de ladite transformation. On pourrait éventuellement choisir d'exclure les trajectoires à distance maximale mais ce problème se pose également de manière plus générale quand les distances ne peuvent prendre qu'un ensemble limité de valeurs, faible par rapport au nombre de trajectoires et que de nombreuses trajectoires sont donc à la même distance de la trajectoire de référence.

Pour pallier ce biais nous brisons l'égalité des trajectoires situées à la même distance de la trajectoire de référence par un choix aléatoire. Pour une meilleure compréhension, considérons six trajectoires $T_2, T_3, T_4, T_5, T_6, T_7$ dont les distances respectives vis à vis de la trajectoire de référence T_1 sont : 0,3 ; 1 ; 0,5 ; 1 ; 1 ; 0,2. L'ordonnancement naturel des plus proches voisins de T_1 serait $T_7; T_2; T_4; T_3; T_5; T_6$ si on utilisait l'ordre des indices en cas d'égalité. Si la distance utilisée a une résolution limitée comme expliqué plus haut, l'ordre des trois dernières trajectoires $T_3; T_5; T_6$ a très peu de chance de changer en fonction des transformations. Ici, au contraire, on va briser les égalités par un choix aléatoire et donc les trois trajectoires vont se retrouver ordonnées différemment dans chacun des jeux de données dérivés. Dans un cas extrême, si toutes les trajectoires sont à distance 1 de la trajectoire de référence dans le jeu de données original et dans un jeu de données dérivé, le bris d'égalité va amener à une absence totale de corrélation entre les deux ordres. Par souci d'impartialité nous appliquons cette même technique de classification des plus proches voisins à toutes les distances utilisées.

La comparaison des distances en fonction de leur robustesse à des transformations ne permet pas de rendre compte du pouvoir discriminant (la résolution) de ces dernières. Afin de mieux évaluer ce pouvoir discriminant des distances, c'est-à-dire leur aptitude à rendre compte précisément des écarts relatifs entre différentes trajectoires, nous proposons en complément de la comparaison par transformation, une comparaison par clustering.

4.3.2 Comparaison par clustering

Le pouvoir discriminant des distances peut s'évaluer, entre autres, à travers leur performance lors d'une opération de clustering. L'idée sous-jacente est que si l'on dispose de groupes de trajectoires similaires (proches visuellement par exemple) alors plus une distance est discriminante mieux elle parvient à répartir les trajec-

toires dans leurs clusters d'origine.

Pour mettre en œuvre la comparaison des distances par clustering, nous sélectionnons des trajectoires réparties en groupes bien distincts. Le clustering est effectué par le biais de l'algorithme *Hierarchical Cluster Analysis* (HCA) qui est un algorithme de clustering hiérarchique pouvant s'appliquer aussi bien aux distances métriques qu'à celles non métriques.

La comparaison des résultats du clustering est faite grâce à la mesure *Adjusted Rand Index* (ARI) [HA85] et au coefficient *Silhouette* [Rou87]. ARI évalue la qualité d'un clustering en comparant le résultat obtenu à la vérité terrain, c'est-à-dire la répartition de référence, réalisée manuellement, des trajectoires en clusters. Cette comparaison résulte en une mesure qui a la particularité d'ignorer les permutations. En effet, il peut advenir que les trajectoires soient parfaitement réparties dans des clusters mais que l'ordre de ces clusters diffère de celui fourni par la vérité terrain. Un tel résultat doit pouvoir être comptabilisé comme étant correct, d'où la nécessité d'ignorer les permutations entre clusters. L'autre spécificité de la mesure ARI est qu'elle garantit qu'un clustering aléatoire renvoie une valeur proche de 0 indépendamment de la taille du jeu de données et du nombre de clusters. La formule de la mesure ARI est :

$$RI = \frac{a+b}{C_2^n} \tag{4.6}$$

$$ARI = \frac{RI - E[RI]}{max(RI) - E[RI]} \tag{4.7}$$

— RI correspond à la mesure *Rand Index* dont découle ARI ;

— a est le nombre de paires d'éléments qui appartiennent au même cluster dans la vérité terrain et dans le résultat du clustering ;

— b est le nombre de paires d'éléments qui appartiennent au même cluster dans la vérité terrain et à différents clusters dans le résultat du clustering ;

— E[RI] équivaut à l'espérance du *Rand Index* et s'obtient en calculant cette mesure sur des clustering correspondant à des répartitions aléatoires des éléments dans les clusters.

Le coefficient *Silhouette* mesure, quant à lui, la similarité d'un objet avec son propre groupe par rapport aux autres groupes. Il ne nécessite pas de connaître à

l'avance la vérité terrain. Sa formule est :

$$S = \frac{b - a}{max(a, b)} \tag{4.8}$$

Avec a la distance moyenne entre un élément et tous les autres membres de son cluster et b la distance moyenne entre un élément et tous les éléments du cluster le plus proche. Le coefficient *Silhouette* pour l'ensemble des éléments d'un jeu de données correspond à la moyenne des coefficients *Silhouette* des différents éléments.

Les valeurs de la mesure *ARI* et du coefficient *Silhouette* vont de -1 (pire cas) à 1 (meilleur cas). Les valeurs négatives indiquent généralement une mauvaise affectation des éléments à des clusters. Les valeurs proches de 0 indiquent des clusters qui se chevauchent pour le coefficient *Silhouette* et des assignations aléatoires pour la mesure *ARI*. L'avantage du coefficient *Silhouette* comparé à la mesure *ARI* est qu'il ne nécessite pas de vérité de terrain.

4.4 Implémentation et résultats

4.4.1 Implémentation

Nos expérimentations ont été menées sur un serveur de calcul comprenant 32 processeurs *Intel Xeon E5-2630 v3* et tournant sous *Linux UBUNTU 16.04* avec 128 GB de RAM. Toutes les distances ont été implémentées en langage python et ont été normalisées afin que les valeurs soient comprises entre 0 et 1. Nous comparons la distance *EQRP* à d'autres distances contraintes par un réseau, notamment les versions graphe de *Hausdorff, STLC, DTW, LCSS, EDR et ERP*, sous les angles de la résistance aux transformations et de la qualité du clustering. Nous utilisons la librairie *SciPy* [1] pour le clustering et pour déterminer les coefficients de corrélation. Tous les traitements de données géographiques se font avec l'outil *Quantum GIS (QGIS)*.

Il est à noter que dans le cadre de l'implémentation de la version graphe de *ERP*, il a fallu définir un nœud de référence par rapport auquel les distances doivent être calculées en cas de gap. Parmi les multiples possibilités, nous avons retenu de fonder le choix du nœud de référence sur le critère de l'excentricité.

1. `https://docs.scipy.org`

Rappelons que l'excentricité d'un sommet est la distance maximale entre ce sommet et tous les autres sommets du graphe. Suite à ce choix, deux variantes de la version graphe de *ERP* ont alors été implémentées. Dans la première variante (*ERP_ G1*), le nœud de référence est l'un de ceux ayant l'excentricité minimale et dans la seconde (*ERP_ G2*) c'est un nœud d'excentricité maximale qui est considéré. Un nœud d'excentricité minimale constitue naturellement une référence dans un graphe puisqu'il appartient à son centre. De même un nœud d'excentricité maximale peut faire office de nœud de référence car il est le plus excentré de tous. Concernant les distances *LCSS* et *EDR*, la valeur du seuil de tolérance est fixée à 0, car le bruit qui rendait imprécises les coordonnées des points et imposait la prise en compte d'une marge d'erreur a été supprimé par l'opération de map-matching.

Avant tout traitement, nous extrayons la plus grande composante fortement connexe du graphe représentant le réseau routier de Porto. Ceci afin de garantir que l'on obtienne toujours une valeur de retour lors du calcul des distances entre nœuds. Nous sélectionnons un échantillon de trajectoires de notre jeu de données déjà recalées de sorte qu'elles aient toutes leurs nœuds appartenant à cette plus grande composante fortement connexe. Par la suite, les différentes transformations décrites précédemment sont appliquées aux trajectoires sélectionnées. Nous faisons évoluer la taille des listes ordonnées de plus proches voisins de 20 à 100 par pas de 20. À chaque itération, les corrélations sont calculées entre les listes obtenues pour chaque distance. Les résultats générés sont présentés et analysés dans les sous-sections suivantes. Notons que les calculs de corrélations ont été effectués pour 20 trajectoires sélectionnées aléatoirement dans le jeu de données original, puis les corrélations médianes par transformation et par distance ont été déterminées.

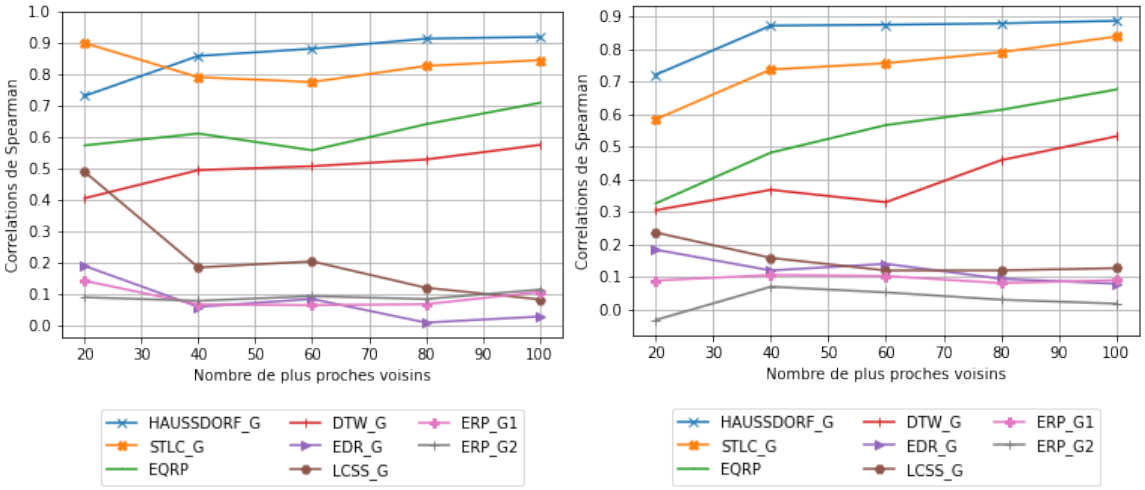
4.4.2 Résultats

Sur chacune des figures ci-après, les courbes présentées montrent l'évolution des valeurs de la corrélation de Spearman en fonction du nombre de plus proches voisins pris en compte dans les listes originales et transformées. La lettre G accolée au nom des distances indique qu'il s'agit des versions graphe.

4.4.2.1 Impacts de la suppression de nœuds

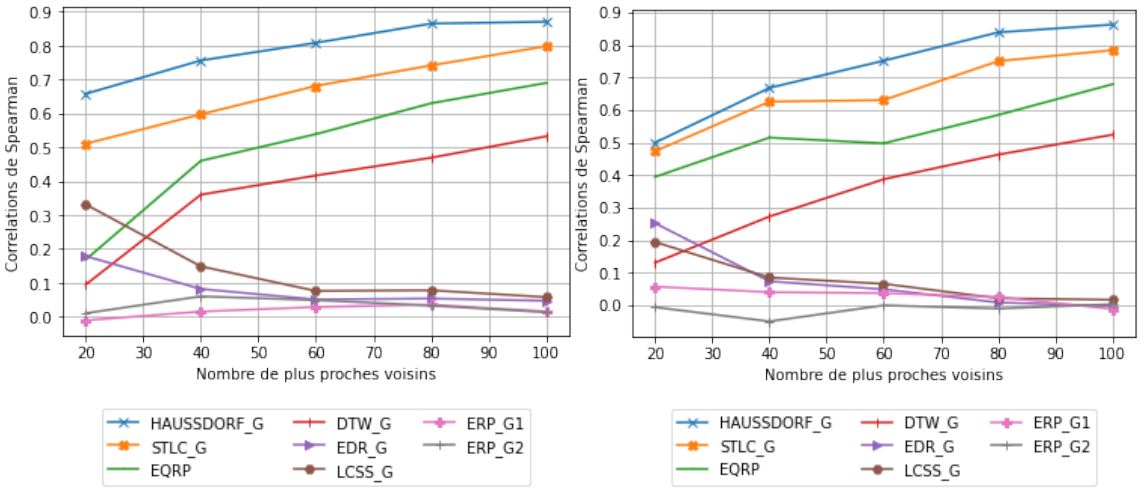
La mise en œuvre de la transformation par suppression de nœuds implique de retirer une certaine proportion de nœuds des trajectoires puis d'évaluer les corréla-

tions entre les listes originales et transformées de plus proches voisins. Durant nos expérimentations, les ratios utilisés pour opérer la transformation par suppression de nœuds sont : 20%, 40%, 60% et 80%. Les résultats obtenus sont présentés à la figure 4.2. L'observation générale qui ressort de ces résultats est une évolution à la baisse des valeurs de corrélations lorsque le nombre de nœuds supprimés augmente.



(a) Ratio = 20%

(b) Ratio = 40%



(c) Ratio = 60%

(d) Ratio = 80%

FIGURE 4.2 – Évolution de la corrélation de Spearman par distance en fonction du nombre de plus proches voisins dans le cadre de la transformation par suppression de nœuds.

Parmi les distances comparées, celle de *Hausdorff* apparaît comme la plus robuste à la suppression de nœuds, quel que soit le ratio considéré, avec des taux de corrélation régulièrement compris entre 80% et plus de 90%. *STLC* se positionne comme la deuxième distance gérant le mieux les effets de la transformation par suppression de nœuds avec des valeurs de corrélation atteignant les 80%. La nouvelle distance que nous proposons *EQRP* se montre également performante et génère des corrélations de l'ordre de 70% dans le meilleur des cas. Elle est surtout bien plus performante que les versions graphe des autres distances intégrant les distorsions que sont *ERP*, *EDR LCSS* et *DTW*. Parmi ces dernières, seule *DTW* conserve des valeurs moyennes de corrélation tandis que les trois autres présentent des corrélations qui évoluent vers 0. Plus particulièrement, les performances de *ERP*, quelle que soit la variante considérée, ressortent comme étant les moins bonnes.

4.4.2.2 Impacts de l'ajout de boucles

La transformation par ajout de boucles est appliquée aux trajectoires originales en fixant le ratio des nœuds sélectionnés à 5% et 10% et le rayon de voisinage à 250 et 500 mètres. Ces ratios permettent de limiter le nombre de boucles générées sur les trajectoires afin d'éviter de trop les dénaturer. Le choix des deux valeurs de rayon vise à produire des boucles de taille intermédiaire (ni trop petites ni trop grandes) et surtout permet de s'assurer de la présence de nœuds voisins dans le graphe représentant le réseau routier.

Les courbes présentées à la figure 4.3 synthétisent les résultats obtenus. Généralement, les résultats montrent que les différentes distances sont plus sensibles à l'ajout de boucles puisqu'elles modifient l'allure globale des trajectoires. En outre, cette sensibilité s'accroît avec le nombre et la taille des boucles parce que les corrélations décroissent avec l'augmentation des valeurs des paramètres gérant le ratio de nœuds affectés et le rayon de voisinage.

Comme dans le cas de la suppression de nœuds, les trois meilleures distances sont la distance de *Hausdorff*, *STLC* et *EQRP*. Toutefois, ici, l'écart entre les valeurs des corrélations de ces trois distances est plus faible que précédemment. La distance de *Hausdorff* conserve les meilleures performances, avec des corrélations atteignant 80% dans la plupart des cas. Les courbes décrivant l'évolution des corrélations de *STLC* et *EQRP* se rejoignent au fur et à mesure que le nombre de plus proches voisins s'accroît. Les deux distances ont donc des capacités similaires

à résister à l'ajout de boucles avec des valeurs des corrélations qui atteignent au maximum 70%. Leur robustesse est cependant moindre comparativement à celle de la distance de *Hausdorff*. Les versions graphe des distances *ERP*, *EDR LCSS* et *DTW* demeurent bien moins performantes que le trio de tête. Surtout *ERP* dont les corrélations sont proches de 0 comme pour la transformation par suppression de nœuds. Toutefois *DTW* se démarque par de meilleures valeurs de corrélations pouvant se chiffrer à plus de 50%. Cela lui confère une résistance moyenne à la transformation par ajout de boucles.

4.4.2.3 Impacts de l'ajout de détours

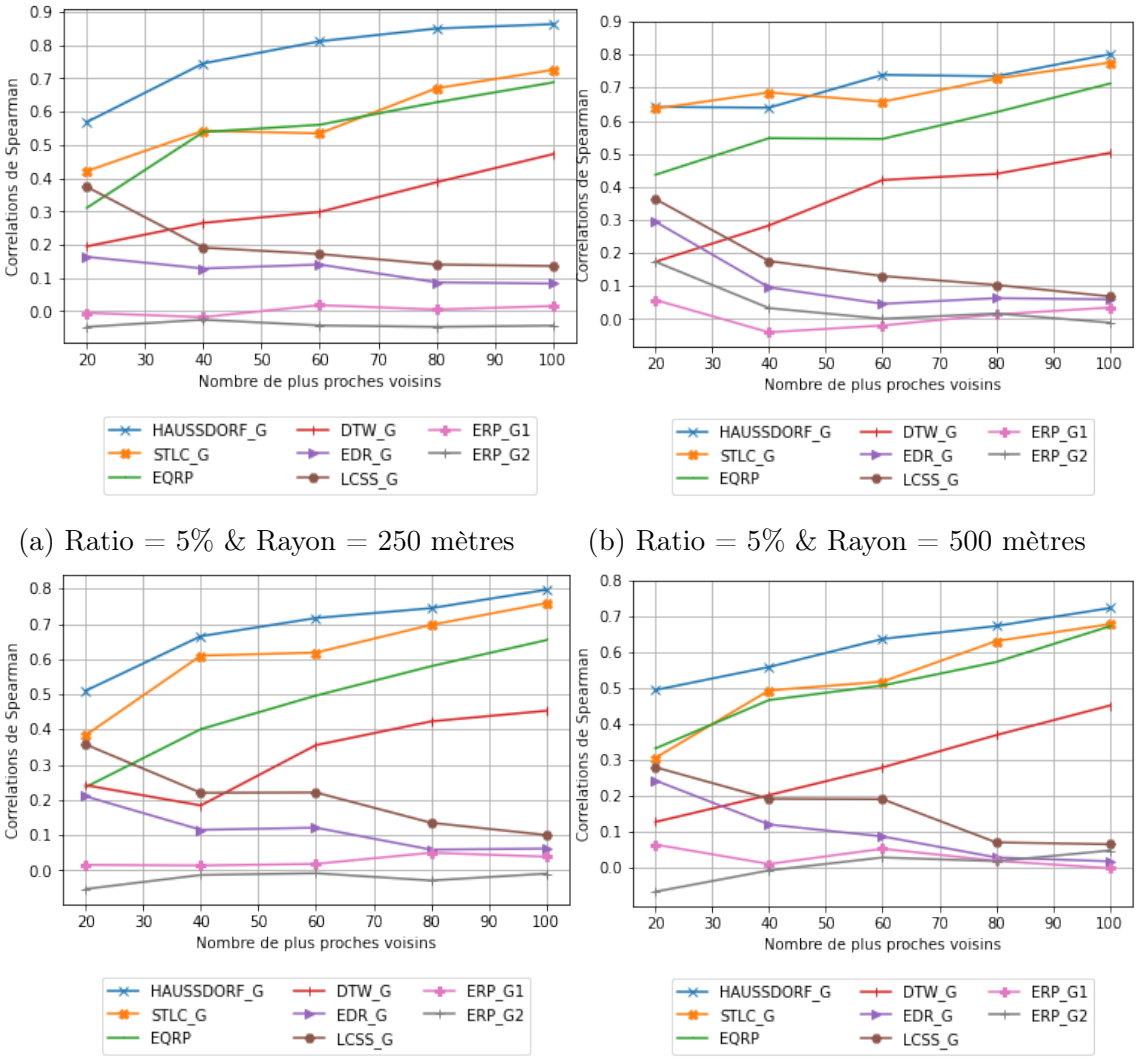
Pour la transformation par ajout de détours nous choisissons les valeurs du paramètre de projection dans l'ensemble $\{3, 5, 7\}$ afin de simuler des détours courts, moyens et plus longs. Le paramètre rayon de voisinage reste fixé à 250 et 500 mètres. Les résultats issus des expérimentations sont présentés à la figure 4.4. On note que l'ajout de détours semble moins impacter les distances dans leur ensemble que l'ajout de boucles.

De toutes les distances, *ERP* est la moins résiliente face aux transformations avec des corrélations proches de 0. Suivent respectivement *EDR* et *LCSS* qui affichent également une faible capacité à gérer cette transformation, au regard de leurs corrélations respectives. Quant à *DTW* elle présente une résistance moyenne aux transformations avec des corrélations atteignant au mieux les 60%.

Les trois meilleures distances demeurent dans l'ordre : la distance de *Hausdorff*, *STLC* et *EQRP*. Indépendamment de l'importance du détour imposé la distance de *Hausdorff* présente des performances régulièrement situées entre 80% et 90%. Ce qui traduit une excellente aptitude à gérer ce type de transformation. *STLC* et *EQRP* atteignent les 70% dans la plupart des cas de figure. Elles ont donc également de bonnes capacités à gérer l'ajout de détours.

4.4.2.4 Résultats du clustering

Afin d'extraire des trajectoires réparties en clusters distincts, nous commençons par créer des régions au sein de la ville de Porto. Au total nous générons cinq régions parfaitement disjointes. La phase suivante consiste à sélectionner, dans la base de données, des trajectoires se déroulant entièrement au sein des régions crées. Nous sélectionnons vingt trajectoires par zone afin d'obtenir un total de cent



(a) Ratio = 5% & Rayon = 250 mètres

(b) Ratio = 5% & Rayon = 500 mètres

(c) Ratio = 10% & Rayon = 250 mètres

(d) Ratio = 10% & Rayon = 500 mètres

FIGURE 4.3 – Évolution de la corrélation de Spearman par distance en fonction du nombre de plus proches voisins dans le cadre de la transformation par ajout de boucles.

trajectoires. La figure 4.5 illustre les zones disjointes ainsi que les trajectoires qui y sont extraites.

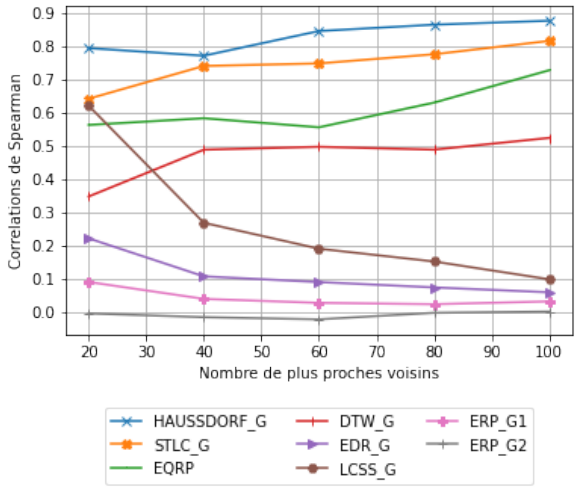
En amont de l'opération de clustering, les matrices de distances entre paires de trajectoires sont calculées. Chaque ligne d'une matrice de distances contient les distances d'une trajectoire à toutes les autres contenues dans le jeu de données.

Une fois les matrices calculées, nous les utilisons pour appliquer l'algorithme de clustering hiérarchique HCA. Le critère de fusion entre clusters que nous utilisons minimise la variance des distances entre les éléments des clusters (méthode de Ward). Les résultats obtenus sont résumés dans le tableau 4.1.

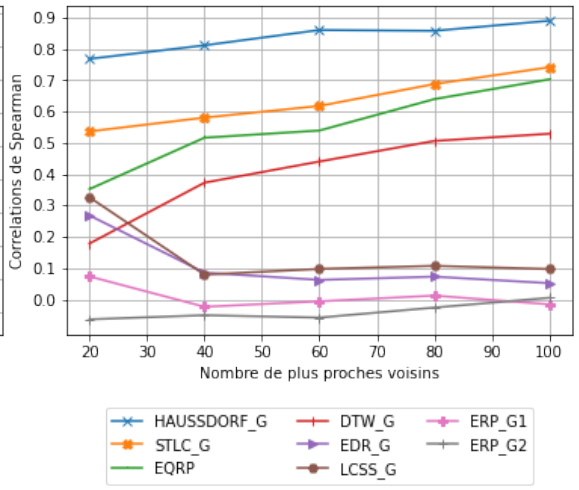
TABLE 4.1 – Évaluation de la qualité du clustering par distances

Distances	**ARI**	**Silhouette**
STLC_G	1.0	0.887
EQRP	1.0	0.593
DTW_G	1.0	0.561
Hausdorff_G	0.975	0.488
LCSS_G	0.283	0.147
EDR_G	0.173	0.106
ERP_{G1}	0.022	0.367
ERP_{G2}	0.006	0.434

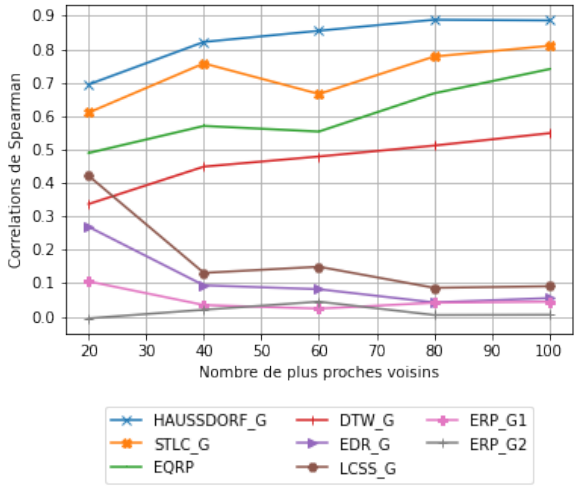
Les résultats issus du clustering corroborent en grande partie les conclusions découlant des expériences liées aux transformations des trajectoires. La distance de *Hausdorff*, *STLC* et *EQRP* parviennent à répartir parfaitement (ou quasi-parfaitement) les trajectoires en clusters distincts. Il en est de même pour DTW qui se démarque ici, par des performances sensiblement égales contrairement à ce qui a été observé pour les transformations. Quant aux distances *LCSS*, *EDR* et *ERP* elles produisent des partitionnements de faible qualité. Notons que *ERP* demeure la distance la moins performante de toutes.



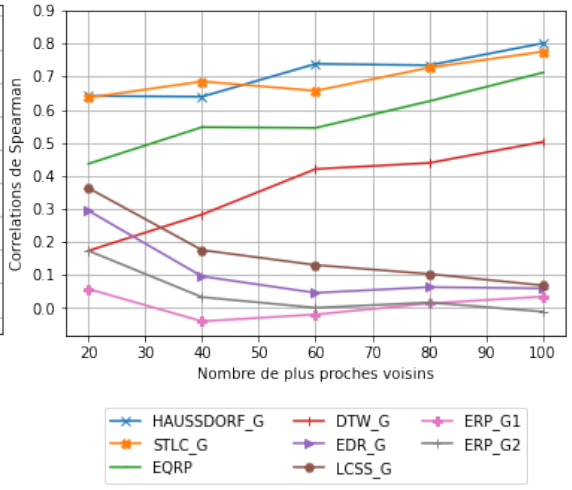
(a) Projection = 3 & Rayon = 250 mètres



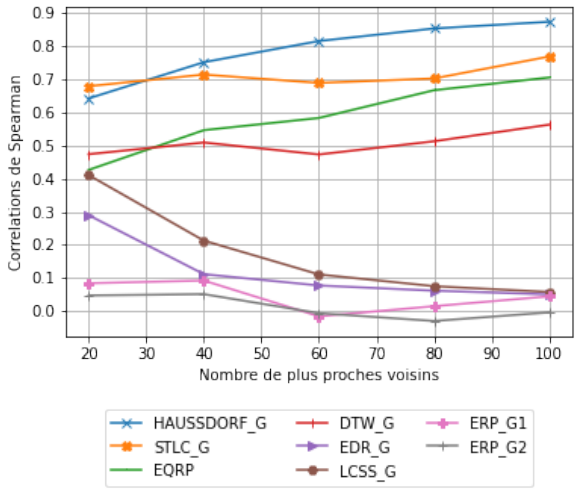
(b) Projection = 3 & Rayon = 500 mètres



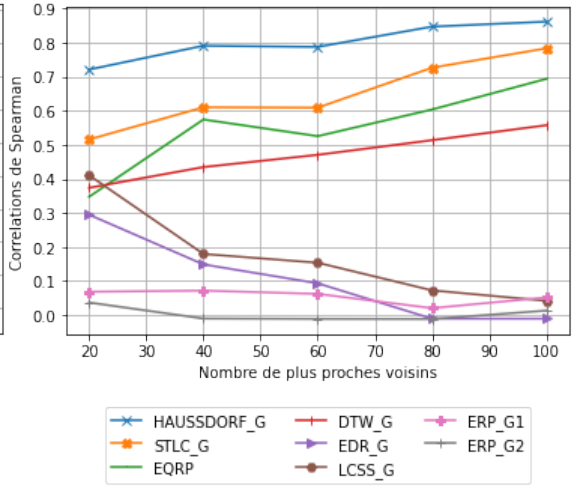
(c) Projection = 5 & Rayon = 250 mètres



(d) Projection = 5 & Rayon = 500 mètres



(e) Projection = 7 & Rayon = 250 mètres



(f) Projection = 7 & Rayon = 500 mètres

FIGURE 4.4 – Évolution de la corrélation de Spearman par distance en fonction du nombre de plus proches voisins dans le cadre de la transformation par ajout de détours.

FIGURE 4.5 – Illustration des zones disjointes avec leurs trajectoires.

4.5 Discussion

Les résultats des expériences liées aux transformations effectuées sur les trajectoires ont fait ressortir une répartition des distances en trois groupes. Le premier est celui des distances résilientes, capables de résister à toutes les transformations avec des valeurs de corrélation pouvant atteindre voire dépasser la barre des 70%. Il rassemble la distance de *Hausdorff*, *STLC* et *EQRP* qui est la distance que nous avons introduite à la lumière de l'analyse de l'état de l'art. Le second groupe ne contient que la distance *DTW* qui affiche des performances moyennes (entre 50% et 60% au mieux) pour les trois transformations. Enfin, *LCSS*, *EDR* et *ERP* forment le groupe des distances les moins robustes. Leurs performances descendent régulièrement en dessous de 20%. Cette répartition des distances est confirmée par les résultats du clustering à la nuance que *DTW* fournit cette fois-ci des résultats similaires aux trois distances les plus performantes.

La distance de *Hausdorff* résiste bien aux diverses transformations en raison de sa formulation basée sur l'évaluation des distances minimales entre nœuds et trajectoires car ces distances ont tendance à rester stables même lorsque les trajectoires sont perturbées. En effet, la distance minimale entre un nœud N et une trajectoire ne varie a priori que si la trajectoire change sensiblement de forme de sorte à modifier son nœud le plus proche de N. Or, il se fait que les transformations n'affectent que localement et aléatoirement la forme des trajectoires, ce qui confère à la distance de *Hausdorff* sa robustesse. En espace euclidien *STLC* apparaissait déjà comme une distance très robuste face aux différentes transformations [SLZ+20]. Cette aptitude est confirmée en espace contraint par un réseau et s'explique par le fait qu'elle s'appuie sur l'utilisation combinée de distances de type nœud à trajectoire et d'une exponentielle négative qui atténuent l'effet des variations liées aux transformations. Le cas de *DTW* est particulier en ce sens que c'est une distance relativement sensible face aux transformations mais qui a un bon pouvoir discriminant. Ce caractère s'explique par le fait que *DTW* exploite les distances réelles séparant les nœuds des trajectoires comparées. Elle renvoie ce faisant une distance relativement précise entre trajectoires mais toute modification dans la position des nœuds est automatiquement répercutée sur la distance calculée. Concernant les distances classiques intégrant les distorsions (*LCSS*, *EDR* et *ERP*) leur faible performance tient pour ce qui concerne *LCSS* et *EDR* à leur mode de calcul qui n'intègre pas l'éloignement réel entre nœuds et à l'utilisation

de seuils d'appariement. Dans le cas de *ERP* nous soupçonnons un impact négatif des nœuds de référence même si cela reste à montrer. Notre choix d'une double implémentation de la version graphe de *ERP*, avait pour objectif de faire ressortir l'impact du choix du nœud de référence sur sa robustesse et sa précision. Toutefois, les résultats obtenus ne permettent pas de conclure.

Bien que les résultats semblent montrer une légère prédominance des distances de type nœud à trajectoire sur la distance *EQRP*, il faut rappeler que celles-ci ne prennent pas en compte l'ordre des nœuds et se retrouvent donc limitées dans leur évaluation des distances entre trajectoires. Cette limitation se manifeste essentiellement dans le cadre de la recherche de trajectoires similaires dans une base de trajectoires. Prenons l'exemple de trois trajectoires de voitures T_1, T_2 et T_3 telles que les deux premières soient disjointes et que T_3 soit identique à T_2 en termes de nœuds mais de sens opposé. L'usage d'une distance de type nœud à trajectoire, comme la distance de *Hausdorff*, ne permettra pas de distinguer les trajectoires T_2 et T_3 qui paraîtrons parfaitement identiques car étant à la même distance de T_1 alors que la distance *EQRP* parviendra à capter la différence liée au sens de circulation des véhicules parce qu'elle aligne au mieux les trajectoires en fonction de l'ordre des nœuds. *EQRP* a donc un pouvoir discriminant supérieur aux distances de type nœud à trajectoire.

In fine, l'intuition qui nous a conduit à formuler la nouvelle distance d'édition *EQRP* s'est avérée fondée puisque validée par les résultats de nos expériences. D'une part, elle prend en compte les distorsions et intègre ce faisant l'ordre des nœuds formant les trajectoires dans leur comparaison. D'autre part elle compense efficacement l'effet des variations de distance entre nœuds en affectant une pénalité dont la valeur équivaut à la distance de *Hausdorff*. Elle est, pour finir, bien plus performante que les autres distances intégrant des distorsions proposées jusqu'ici dans la littérature.

4.6 Conclusion

Ce chapitre nous aura permis d'étendre le travail d'analyse des distances dédiées aux trajectoires contraintes par un réseau, amorcé dans l'état de l'art. Nous y avons proposé la définition d'une nouvelle distance alliant les qualités des distances de type nœud à trajectoire à celles des distances d'édition. Une étude compara-

tive a également été menée afin d'évaluer les capacités des principales distances (contraintes par un réseau) utilisées dans la littérature à résister à des transformations mais aussi à produire des clusters de bonne qualité. Nous pensons qu'il serait intéressant de travailler sur l'utilisation de EQRP pour la résolution de requêtes visant à interroger de grandes bases de données de trajectoires pour extraire des trajectoires répondant à des prédicats. L'objectif final serait de créer une plate-forme d'analyse dédiée aux trajectoires contraintes par un réseau intégrant un moteur de recherche. C'est une perspective qui amènerait à développer et/ou utiliser des architectures parallélisées ou distribuées pour le stockage et le calcul des distances mais aussi à définir des index spécifiques à la distance EQRP, en exploitant par exemple l'inégalité triangulaire.

Chapitre 5

Zones fonctionnelles

Sommaire

5.1 Introduction

D'ici 2050, c'est-à-dire dans moins de 30 ans, les villes auront 2,5 milliards d'habitants de plus qu'aujourd'hui [1] et plus des deux-tiers de la population mondiale sera urbaine. Cette projection impose de repenser les méthodes de gestion des villes pour faire face aux challenges qui en découleront. Dans cette optique, une approche consiste à analyser les activités des citadins pour découper les villes en zones fonctionnelles, capables de fournir une meilleure compréhension de la structuration spatiale et socio-économique des cités.

L'identification de ces zones fonctionnelles est un sujet d'actualité qui a de nombreuses applications telles que la recommandation de circuits touristiques ou l'implantation de nouveaux commerces. Cependant, sa mise en œuvre peut s'avérer complexe car le concept de zone fonctionnelle n'a pas, à l'heure actuelle, de définition harmonisée et les segmentations de villes obtenues dans l'état de l'art sont difficiles à valider [FF11]. Différentes méthodes d'apprentissage supervisé [ZYZ+15, QLL+11] et non-supervisé [YZX12, WGDQ18, FZWZ15, HCL15] ont été proposées. Parmi ces dernières, principalement basées sur du clustering, deux grandes familles émergent : les approches statistiques et celles à base de graphes qui nous paraissent plus pertinentes car elles ont l'avantage d'encapsuler naturellement les interactions.

Dans ce chapitre nous développons une approche à base de graphes pour délimiter les zones fonctionnelles en utilisant à la fois des données de mobilité et des points d'intérêt (POI), sachant que les autres approches basées sur les graphes exploitent seulement les données de mobilité. Notre méthodologie consiste en (i) la construction d'un graphe de régions formelles en utilisant la mobilité, (ii) l'identification de communautés en regroupant des zones formelles et (iii) le contrôle de l'hétérogénéité des communautés identifiées en utilisant l'entropie des POI.

Le reste du chapitre est organisé de la manière suivante : après avoir donné plusieurs définitions de la notion de zone fonctionnelle dans la section 5.2, nous présentons divers travaux proches de notre proposition dans la section 5.3. Ensuite nous décrivons notre méthodologie pour extraire les zones fonctionnelles dans la section 5.4 et terminons par les résultats obtenus en section 5.5.

1. https://www.un.org/development/desa/fr/news/population/2018-world-urbanization-prospects.html

5.2 Le concept de zone fonctionnelle

Le problème de la délimitation des zones fonctionnelles est fréquemment et diversement abordé dans la littérature. Cependant, à notre connaissance, aucune définition consensuelle de zone fonctionnelle n'a pu être donnée jusqu'ici. Plusieurs propositions coexistent, chacune ayant sa particularité. Le plus souvent, elles mettent l'accent sur les interactions socio-économiques au sein des zones fonctionnelles.

Par exemple, l'OCDE [fECoO02] définit une zone fonctionnelle comme une unité territoriale résultant de l'organisation de relations sociales et économiques de sorte que ses frontières ne reflètent pas de particularités géographiques ou historiques. En général, elle est organisée autour d'un ou de plusieurs nœuds de sorte que les zones alentour soient connectées à ce(s) nœud(s) à travers différents systèmes (transports, communications, travail, échanges) [CSI15]. Dans le même ordre d'idées, [Kar07] postule qu'une zone fonctionnelle est caractérisée par une agglomération d'activités et par des infrastructures de transport intrarégionales qui facilitent la mobilité des individus et des produits à l'intérieur de ses frontières.

D'autres auteurs se font plus précis en restreignant les interactions socio-économiques au cadre du marché du travail et donc aux déplacements domicile-travail. Ainsi, selon [Ant05], une zone fonctionnelle est une zone de mobilité domicile-travail. Elle est une agglomération de lieux de travail qui attirent la main d'œuvre des régions environnantes. Sa caractéristique la plus importante est sa capacité à dépasser les frontières administratives. Farmer et al. [FF11], abondent dans le même sens en considérant les zones fonctionnelles comme des régions géographiques au sein desquelles les interactions en termes de trajets domicile-travail sont maximisées et entre lesquelles ces interactions sont plutôt minimisées.

Au-delà des interactions socio-économiques, une zone fonctionnelle est aussi définie comme un territoire possédant une fonction particulière (zone résidentielle, commerciale, touristique, etc.) qui dépend des activités humaines s'y déroulant. Suivant cette conception, une zone fonctionnelle correspond à une certaine utilisation des terres [ZLW+14, GJC17].

En dépit de leur diversité, ces définitions permettent d'identifier des caractéristiques essentielles concernant les zones fonctionnelles. Premièrement et fondamentalement, elles sont délimitées de sorte que les interactions socio-économiques sont plus fortes en leur sein qu'entre elles. Deuxièmement, elles regroupent différents types d'activités et sont donc hétérogènes. Troisièmement, elles ne correspondent

pas nécessairement à un découpage administratif, géographique ou historique. Enfin, elles peuvent être associées à des utilisations spécifiques des terres.

Précisons que, lorsque les zones fonctionnelles sont délimitées à l'échelle de centres urbains, elles sont qualifiées de zones fonctionnelles urbaines et c'est précisément l'identification de ce type de zones qui nous intéresse dans ce chapitre. Par ailleurs, du fait des caractéristiques sus-citées, la détermination des zones fonctionnelles impose de disposer de données (mobilité, communication, commerce ou travail) décrivant des interactions socio-économiques. Dans notre contexte, nous utiliserons des données de mobilité et des POI pour leur identification.

5.3 Etat de l'art

Les travaux récents liés à la délimitation des zones fonctionnelles urbaines font ressortir deux grands types d'approches : les approches statistiques et les approches graphes.

Approches statistiques. Les articles décrivant les approches statistiques associent habituellement la délimitation des zones fonctionnelles urbaines à l'identification des zones résidentielles, commerciales, administratives, touristiques, etc. Les zones fonctionnelles urbaines y sont perçues comme étant des espaces remplissant une fonction spécifique aux yeux des usagers qui les fréquentent. En cohérence avec cette conception, Yuan et al. [YZX12] ont développé une méthode utilisant la DMR (Dirichlet Multinominal Regression) pour identifier les régions fonctionnelles dans la ville de Pékin. Ils combinent les modèles de mobilité extraits des trajets en taxi et les données des POI comme entrée de leur modèle qui est une amélioration du LDA (Latent Dirichlet Allocation). Similairement, Gao et al. [GJC17] ont proposé une méthode également basée sur la LDA pour détecter les zones fonctionnelles dans dix villes américaines mais sans tenir compte des POI. Dans [THL+18], Tu et al. utilisent quant à eux des images de télédétection combinées à des données de positionnement GSM pour identifier les zones fonctionnelles. Ils extraient diverses caractéristiques des deux sources de données puis appliquent un regroupement hiérarchique pour représenter les zones fonctionnelles. D'autres méthodes basées sur la factorisation matricielle [WGDQ18], l'algorithme EM (Expectation Maximization) [LS15], LRA (Low Rank Approximation) ou utilisant le K-means [ZLW+14, ZYZ+15] sont également proposées dans la littérature. Elles

exploitent, selon les cas, les données des courses de taxi, des médias sociaux, des cartes à puce de bus et les POI.

Approches graphes. Cependant, les études sur les zones fonctionnelles urbaines ne sont pas exclusivement liées à l'utilisation des sols. Certains auteurs considèrent les zones fonctionnelles urbaines plutôt comme des régions où les interactions socio-économiques sont plus fortes à l'intérieur de leurs frontières qu'à l'extérieur. Nous partageons ce point de vue car il correspond à notre perception des zones fonctionnelles. Les méthodes décrites dans leurs articles sont généralement basées sur des graphes, car ces derniers modélisent naturellement les interactions. Elles impliquent également la détection de communautés qui vise à trouver un ensemble de nœuds plus fortement interconnectés entre eux qu'avec l'extérieur. Par exemple, dans [FF11], Farmer et Fotheringham formalisent la délimitation des zones fonctionnelles en tant que problème de détection de communautés dans un réseau de flux de déplacements vers les lieux de travail. Leur méthode cherche à maximiser la modularité du processus de détection de communautés en utilisant une approche spectrale. Leur cas d'étude porte sur l'ensemble du territoire irlandais. De même, pour découvrir les zones fonctionnelles urbaines dans la ville de Shanghai, Fan et al. [FZWZ15] appliquent l'algorithme Fast-Newman afin de détecter les communautés dans un graphe de cellules de Voronoï, construit à partir de données de trajectoires de taxis. Ils identifient ensuite les zones fonctionnelles à partir de ces communautés et les étiquettent. Pour leur part, Demsar et al. [DRMB14] s'appuient sur une méthode de détection de communautés recouvrantes (qui se chevauchent) pour l'identification des zones fonctionnelles à Londres.

Ces méthodes basées sur des graphes, n'utilisent que les données de mobilité et ne tiennent pas compte de la composition socio-économique des territoires. C'est une limitation car les zones fonctionnelles ne sont pas seulement des régions de fortes interactions, mais aussi des zones hétérogènes qui regroupent différents types d'activités. Par conséquent, l'intégration des données socio-économiques peut aider à améliorer la qualité du processus de détermination des zones fonctionnelles urbaines. C'est pourquoi nous proposons une méthode, basée sur la détection de communautés et qui enrichit les méthodes existantes en combinant l'utilisation des POI avec les données de mobilité. Nous décrivons cette nouvelle méthode dans la section suivante.

5.4 Méthodologie

En nous référant aux définitions de la section 5.2, l'une des principales caractéristiques des zones fonctionnelles est d'avoir des interactions plus fortes à l'intérieur de leurs frontières qu'à l'extérieur. La recherche de régions présentant de telles caractéristiques a été formalisée par [FF11] comme un problème de détection de communautés pouvant être résolu par l'optimisation de la modularité.

La modularité est une mesure permettant d'évaluer la qualité d'une partition des sommets d'un graphe [New06]. Elle est basée sur l'idée qu'une communauté est un ensemble de sommets plus liés entre eux qu'avec l'extérieur (comme peut l'être un groupe d'amis par exemple). Elle est formellement définie par l'équation 5.1 pour un graphe de matrice d'adjacence A et une partition P des sommets de ce graphe :

$$Q(P) = \frac{1}{2m} \sum_{C \in P} \sum_{i,j \in C} \left[A_{ij} - \frac{k_i k_j}{2m} \right] \tag{5.1}$$

Ici, C est une partie de la partition P, i et j sont deux sommets de C avec k_i et k_j leur degré respectif, m est le nombre d'arêtes dans le graphe.

L'idée sous-tendant cette formulation est de comparer, pour chaque communauté, le nombre de liens internes (somme des A_{ij}) avec le nombre de liens internes attendus dans un modèle de référence (somme des $k_i k_j / 2m$). Dans ce cas, le modèle de référence est le modèle configurationnel [BC78] qui génère un graphe aléatoire en préservant les degrés du graphe original (tous les liens sont mélangés mais le degré des nœuds reste inchangé). Les valeurs prises par la modularité sont des nombres réels compris entre -0,5 et 1.

Notre méthode pour représenter les zones fonctionnelles urbaines s'inspire de [FF11] et repose également sur l'optimisation de la modularité, mais seulement en partie. En effet, contrairement aux auteurs, nous intégrons les informations sur les points d'intérêt pour tenir compte de la sémantique et de l'hétérogénéité des zones traitées. En outre, nous utilisons un algorithme d'optimisation directe comme alternative à la méthode spectrale décrite par Farmer et al. Les trois étapes de notre approche sont : la segmentation du territoire, la construction du graphe d'interaction et l'extraction des zones fonctionnelles. La figure 5.1 illustre le déroulement de notre méthode.

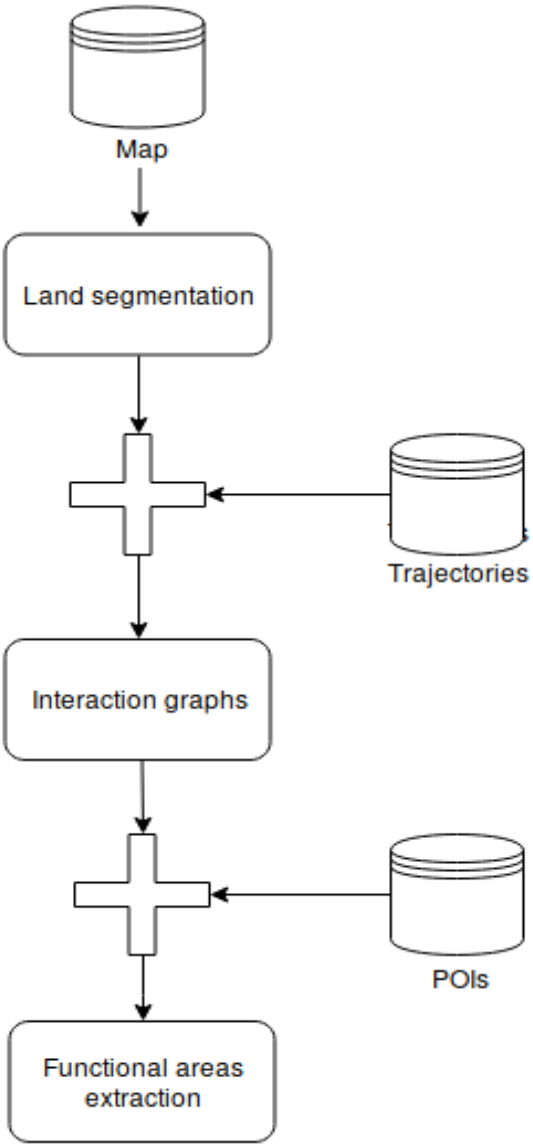


FIGURE 5.1 – Processus de détection de zones fonctionnelles utilisation la détection de communautés.

5.4.1 Segmentation du territoire

Le découpage en grille est une méthode classique de segmentation du territoire qui génère des cellules de surface identique. Il est très utilisé car simple à implémenter [WWT+16, LGGL15, PSR+14, LWXG12]. Cependant, il présente quelques limites. En premier lieu, il dépend d'un paramètre fixant la taille des cellules qui doit être ajusté pour chaque nouvelle zone d'étude. Ensuite, il n'épouse pas l'aménagement urbain, c'est-à-dire qu'il ne tient pas compte de l'organisation spatiale des zones urbaines traitées. Afin de pallier ces insuffisances, nous avons préféré diviser l'espace d'étude en zones disjointes en utilisant le réseau routier. Les unités spatiales obtenues sont *a priori* homogènes d'un point de vue socio-économique et évitent les limitations du découpage en grille. Nous appellerons ces unités spatiales des régions formelles, comme indiqué dans [B+94].

5.4.2 Construction du graphe d'interaction

Les interactions entre les régions formelles sont modélisées par un graphe de flux et représentées sous la forme d'une matrice Origine-Destination (OD). Chaque cellule de la matrice indique le nombre de déplacements effectués entre les deux régions formelles correspondantes pendant une certaine période. La matrice est symétrique car nous ignorons les directions des déplacements. Afin d'obtenir des zones continues, les valeurs des cellules sont ajustées en utilisant une pondération géographique. Les poids sont calculés grâce à la fonction gaussienne, présentée dans l'équation 5.2 et initialement proposée par [FF11] :

$$A_{ij} = W_{ij} \exp(-d_{ij}^2/h^2) \tag{5.2}$$

où A_{ij} est la valeur pondérée de l'interaction entre les régions formelles i et j, W_{ij} représente le nombre de déplacements entre les zones i et j, d_{ij} est la distance euclidienne entre les zones i et j, h est un seuil permettant de contrôler la largeur de bande de la fonction gaussienne ainsi que la compacité des zones fonctionnelles. La fonction gaussienne, ainsi définie, a pour objectif de pénaliser le poids de l'interaction entre les zones formelles dès lors que la distance les séparant augmente au regard de la valeur de seuil fixée.

5.4.3 Extraction des zones fonctionnelles

L'optimisation de la modularité est un problème NP-complet [BDG+06]. Cette limitation a conduit au développement d'heuristiques capables de trouver de bonnes partitions le plus rapidement possible. Selon [For10], l'algorithme de détection de communautés de Louvain [BGLL08] est l'une des heuristiques les plus efficaces de la littérature. Louvain effectue de manière répétée deux étapes. Une étape d'optimisation locale de la modularité, au cours de laquelle les nœuds sont déplacés de communauté en communauté afin d'augmenter la modularité, jusqu'à ce qu'un maximum local soit atteint. Et une étape qui fusionne les nœuds à l'intérieur des communautés pour réduire la taille du réseau. Elle surpasse la plupart des autres solutions tant en termes de temps de calcul que de qualité de la modularité obtenue par l'optimisation. C'est la raison pour laquelle nous l'avons choisie.

Pour extraire les zones fonctionnelles, nous appliquons l'algorithme de Louvain à une série de graphes d'interaction obtenus en faisant varier la valeur de seuil de la fonction gaussienne (dans la formule 5.2). Ensuite, pour chaque partition de graphe obtenue, nous évaluons l'hétérogénéité des régions correspondantes en calculant leur entropie moyenne, telle que définie dans l'équation 5.3 en utilisant la distribution des points d'intérêt.

$$E_{average} = \frac{1}{|R|} \sum_{R \in P} E_R \tag{5.3}$$

où E_R est l'entropie définie par Shannon [Sha48] de la région R et calculée comme :

$$E_R = -\sum_{i \in R} p_i * \log p_i \tag{5.4}$$

avec p_i la proportion de POI de la catégorie i dans la région R.

L'intuition est que, plus une région est hétérogène, plus son entropie est élevée et plus il y a de raisons pour que des interactions spatiales se produisent à l'intérieur de celle-ci. Enfin, nous sélectionnons la partition qui maximise le produit de la modularité par l'entropie moyenne. Cette règle de sélection multicritères garantit que les zones fonctionnelles générées présentent de fortes interactions ainsi qu'une grande hétérogénéité.

5.5 Résultats et discussions

5.5.1 Jeux de données et zone d'étude

Nous utilisons les trajectoires contenues dans le jeu de données de Porto, qui contient 1710670 enregistrements. Après une légère étape de prétraitement pour exclure les trajets dont les données sont manquantes et ceux qui sont trop courts ou trop longs, on obtient 1438924 trajectoires. Les trajectoires très courtes ont leur origine et leur destination associées à la même région formelle tandis que les trajectoires très longues peuvent avoir une origine à l'intérieur de la frontière administrative de Porto et une destination à l'extérieur (et vice versa). Nous stockons les points origine et destination dans une base de données sous *Postgresql* puis les indexons avec un index spatial de type *R-Tree* proposé par l'extension *Postgis*.

Un total de 7710 points d'intérêt ont été collectés à partir de la plateforme *OpenStreetMap*. Le jeu de données des POI contient divers types (distributeur de billets, banque, église, hôtel...) que nous avons agrégés en 10 groupes plus génériques comme résumé dans le tableau 5.1.

Enfin, les régions formelles sont extraites de l'atlas urbain 2012 disponible sur le site de la direction générale du territoire portugais [2]. Il s'agit de données vectorielles contenant des informations cadastrales sur les zones urbaines du Portugal, y compris la ville de Porto. Il n'indique pas le niveau hiérarchique des routes mais contient la plupart des routes observables à Porto. Le processus de segmentation produit 2453 régions formelles. La figure 5.2 illustre la délimitation de Porto en régions formelles.

5.5.2 Expérimentation et résultats

Nos expériences ont été réalisées sur un ordinateur portable *HP Zbook*, contenant un processeur octo-cœur Intel i7-6700HQ cadencé à 8 * 2.60Ghz avec 16Gb de Ram et fonctionnant sous *Linux UBUNTU 16.04*. Les traitements SIG comme l'extraction des zones formelles de l'atlas urbain ou la génération de cartes des zones fonctionnelles ont été effectués avec *QGIS*.

À partir des régions formelles et des données origine-destination, nous avons construit un ensemble de graphes d'interaction en faisant varier la valeur de seuil

2. `http://mapas.dgterritorio.pt/atom-dgt/CDG_atlasurbano2012_Continente_Atom.xml`

TABLE 5.1 – Catégorisation des POI

ID	Types de base	Catégories
1	appartement, hôtel, maison, résidence, dortoir, péniche, chambre d'hôtes, auberge, motel	Hébergement et Résidence
2	tribunal, espace de coworking, ambassade, caserne de pompiers, police, bureau de poste, comptable, entreprise	Lieux de travail et services publics
3	bar, bbq, café, fast food, glacier, pub, mini golf, réserve naturelle, parc, centre sportif, stade	Lieux de restauration et de loisirs
4	aquarium, œuvre d'art, attraction, galerie, musée, parc à thème, point de vue	Tourisme
5	gare ferroviaire, station de bus, taxi, parking	Transport
6	atm, banque, commerce, industrie, vente au détail, entrepôt, kiosque, boutique, tissu	Business et industries
7	collège, maternelle, bibliothèque, archives, école, école de musique, école de langue, université	Education
8	clinique, dentiste, médecin, hôpital, centre infirmerie, pharmacie, centre social, vétérinaire, don de sang	Santé
9	lotissements, terres agricoles, basse-cour, forêt, herbe, terrain vierge, serre horticole, prairie	Terrains vierges et agricoles
10	cimetière, église, chapelle	Lieux de culte

FIGURE 5.2 – Carte des régions formelles de la ville de Porto.

de la fonction gaussienne (paramètre h dans l'équation 5.2) entre 0,05 et 3 avec un pas de 0,05. Les régions n'ayant aucune interaction sont supprimées lors de la phase de construction du graphe.

L'algorithme de Louvain est exécuté dix fois sur chaque graphe d'interaction pour tenir compte de son non-déterminisme et la partition de modularité la plus élevée est sélectionnée. Ensuite, l'entropie moyenne des zones détectées est calculée. Les résultats sont présentés dans la figure 5.3.

La condition d'arrêt de l'expérimentation est la convergence de l'entropie moyenne. Or, d'après la figure 5.3, l'entropie moyenne stagne pour des valeurs de seuil comprises entre 1,5 et 3. Ce qui explique que nous n'ayons pas effectué d'expérimentations au-delà de cette valeur.

La valeur de seuil qui maximise le produit entre entropie moyenne et modularité, d'après la figure 5.3, est $h = 0,35$. La partition induite se compose de onze communautés qui correspondent à des zones fonctionnelles, comme illustré sur la figure 5.4. Chaque communauté est identifiée par un numéro et une couleur. Les figures 5.5 et 5.6 montrent deux exemples de détection de zones fonctionnelles correspondant respectivement aux valeurs de seuil $h = 0,1$ et $h = 2$. On y note une corrélation nette entre la taille des communautés et la valeur de seuil même si nous n'avons pas formalisé cette corrélation.

Après analyse des zones fonctionnelles détectées, la première remarque est que ces zones n'ont pas les mêmes limites que la délimitation administrative de la ville

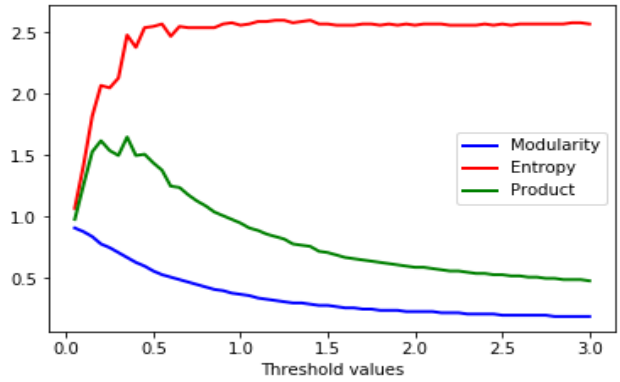


FIGURE 5.3 – Modularité et entropie moyenne pour différentes valeurs du seuil.

FIGURE 5.4 – zones fonctionnelles obtenues avec un seuil $h = 0.35$.

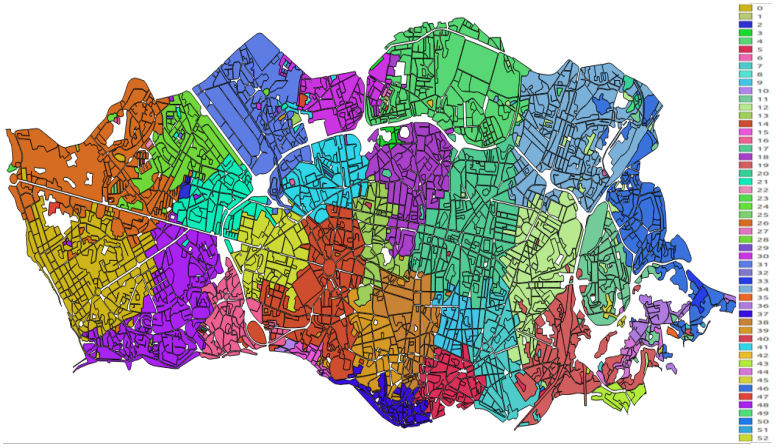


FIGURE 5.5 – zones fonctionnelles obtenues avec un seuil $h = 0.1$.

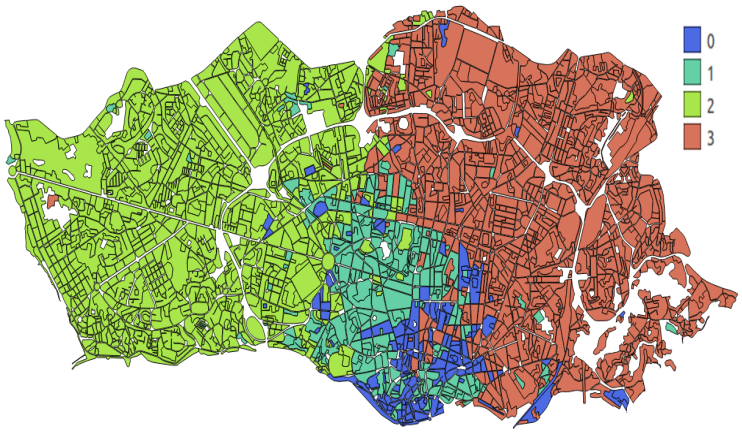


FIGURE 5.6 – zones fonctionnelles obtenues avec un seuil $h = 2.0$.

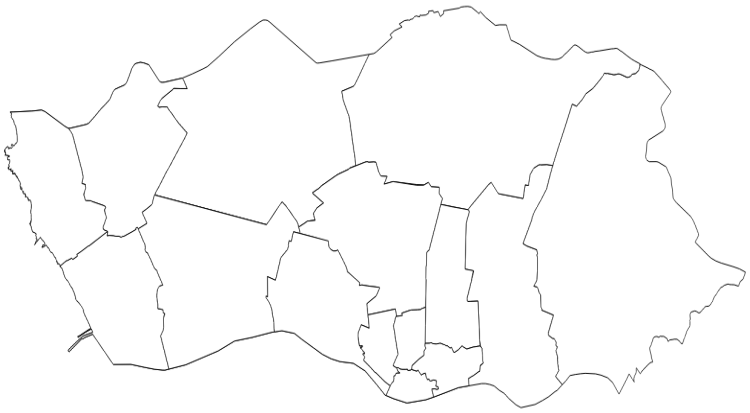


FIGURE 5.7 – Quartiers de Porto.

de Porto en quartiers présentée dans la figure 5.7. Cela confirme que la ville a une structure latente uniquement accessible par le biais des données d'interaction spatiale. La diversité des activités pour chaque zone fonctionnelle est présentée dans la figure 5.8 qui illustre la distribution des POI par communauté. Notons que les couleurs identifiant les zones fonctionnelles détectées et celles associées à la distribution des POI ne sont pas liées.

La figure 5.8 montre que les communautés détectées ont une bonne variété de POI. Les zones résidentielles prédominent dans la plupart des communautés, mais cette situation est normale puisque la fourniture d'un logement est l'une des principales, sinon la principale, fonction d'une ville.

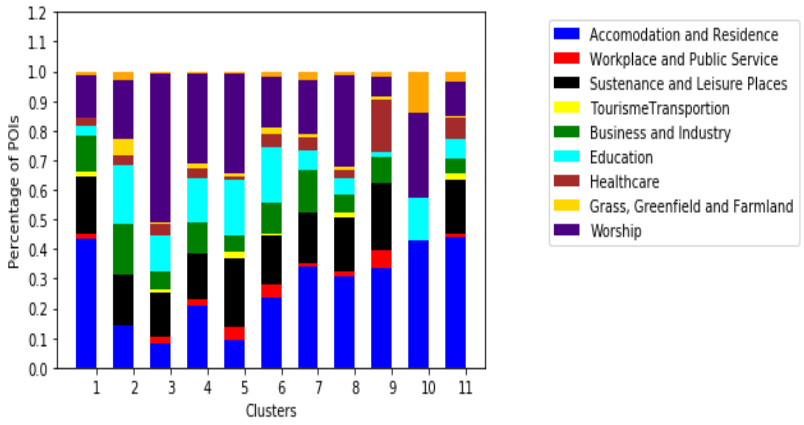


FIGURE 5.8 – Distribution des POIs par zone.

Bien que les zones fonctionnelles détectées consistent en de grands blocs continus, il arrive que certaines de leurs zones formelles se retrouvent en dehors de leurs limites respectives. Cela s'explique par le fait que certaines régions formelles sont fortement interconnectées malgré leur éloignement et conservent une valeur d'interaction importante en dépit de l'application de la fonction gaussienne qui pénalise les liens entre zones formelles éloignées. Pour gérer ces cas particuliers, nous pouvons soit attribuer aux régions formelles dispersées les clusters dominants de leur voisinage, soit les conserver tels quels et les utiliser comme indicateur des liens forts entre les zones fonctionnelles.

D'après les expérimentations de [FF11], la valeur de seuil h de la fonction gaussienne de pondération aurait dû être calculée de manière adaptative pour chaque origine à partir des distances des trajets partant de cette origine et aboutissant à d'autres zones formelles. Cependant, ce mode de détermination de la valeur de seuil h génère des zones fonctionnelles particulièrement fragmentées comme le montre la figure 5.9. Nous en déduisons que la méthode de Farmer et Fotheringham s'avère inefficace pour la détection des zones fonctionnelles à l'échelle d'une ville. Comme solution, nous avons adopté une approche globale de sélection de la valeur de seuil en remplacement de l'approche locale. Cela signifie que nous déterminons une valeur de seuil globale pour toutes les régions formelles en approximant la distribution des distances avec une loi gaussienne. Toutefois, ce mode de calcul génère également des communautés très dispersées, comme le montre la figure 5.10. En raison des mauvais résultats obtenus avec ces deux méthodes de sélection au-

FIGURE 5.9 – zones fonctionnelles avec sélection automatique du seuil pour chaque région formelle.

FIGURE 5.10 – zones fonctionnelles avec sélection automatique du seuil pour toutes les régions formelles.

tomatique de la valeur de seuil, nous n'avons pas pu proposer une comparaison quantitative de notre méthode avec l'approche de Farmer et Fotheringham que nous avons utilisée comme référence.

5.6 Conclusion et travaux futurs

Ce chapitre propose une méthode orientée graphes pour la délimitation des zones fonctionnelles dans les villes et basée sur la détection de communautés. Dans un premier temps, nous divisons la zone d'étude en unités spatiales disjointes appelées régions formelles en fonction du réseau routier. Deuxièmement,

nous construisons des graphes d'interaction pondérés, en utilisant les régions formelles et les trajectoires des taxis. Les sommets de chaque graphe correspondent alors aux régions formelles et deux sommets sont reliés s'il existe des trajets de taxis partant de l'un et se terminant dans l'autre. Les poids sur les arêtes dépendent du nombre de trajets entre les sommets et de leur éloignement. Ils sont calculés grâce à une fonction gaussienne qui pénalise les grandes distances entre régions formelles. Les différents graphes d'interaction sont obtenus en faisant varier la valeur de seuil de la fonction gaussienne. Troisièmement, nous appliquons aux graphes obtenus la méthode de détection de communautés de Louvain afin d'obtenir des partitions qui maximisent la modularité. Ensuite, nous calculons l'entropie moyenne de la partition détectée pour chaque graphe d'interaction. Enfin, nous sélectionnons la partition dont le produit de la modularité par l'entropie moyenne est le plus élevé. Des trajets en taxi et un ensemble de POI de la ville de Porto ont été utilisés pour expérimenter notre méthode avec des résultats cohérents.

Ce travail pourra être approfondi en étudiant la stabilité des zones détectées au cours du temps. À cette fin, l'ensemble de données peut être divisé en sous-ensembles couvrant différentes périodes de temps et les zones fonctionnelles détectées pour chaque sous-ensemble. Ensuite, les cœurs [GCR05] de communautés seront calculés sur les différentes partitions obtenues. En outre, il serait utile de formaliser la relation entre la taille des zones fonctionnelles et les valeurs de seuil de la fonction gaussienne. Une autre perspective est la détermination de communautés qui se chevauchent afin d'avoir une délimitation plus réaliste de la ville car un lieu peut être fortement lié à d'autres lieux appartenant à des zones fonctionnelles disjointes. En outre, nous pouvons nous appuyer sur l'ensemble de la trajectoire et les horodatages associés plutôt que sur les points d'origine et de destination uniquement pour améliorer la méthode. Par exemple, en utilisant l'ensemble de la trajectoire, nous pouvons déterminer la distance réelle parcourue sur le réseau routier et toutes les zones traversées pour effectuer une analyse plus fine. Nous pourrions également comparer les zones fonctionnelles détectées en utilisant, séparément différents types de données de mobilité (trajectoires de taxis, de vélos, de bus, de voitures personnelles) ou en les combinant. Enfin, la dimension temporelle pourrait nous aider à définir une version plus précise du poids des arêtes.

Chapitre 6

Conclusion et perspectives

Sommaire

6.1 Conclusion

Dans cette thèse nous avons entrepris de solutionner quatre problématiques liées à l'analyse des trajectoires contraintes par un réseau : la détection et la correction, à grande échelle, des erreurs issues de leur prétraitement, leur utilisation dans le cadre de l'extraction de zones fonctionnelles, l'identification, la classification et la comparaison des mesures de distance ou de similarité qui leur sont dédiées, la définition d'une nouvelle mesure améliorant celles existantes. La spécificité de nos travaux tient à l'utilisation de données réelles et d'outils de la théorie des graphes pour tenir compte du réseau sur lequel se déroulent les trajectoires mais aussi pour faciliter la généralisation des méthodes et mesures développées.

Dans le chapitre 2 nous avons détaillé l'état de l'art actuel de trois des opérations essentielles à toute analyse de trajectoires : le prétraitement, le calcul de distance ou de similarité et le clustering. Ce travail de revue bibliographique nous a également permis de déterminer les limites actuelles de l'état de l'art pour chacune de ces opérations. Dans le chapitre 3, nous avons présenté le jeu de données de trajectoires et le réseau routier utilisés dans nos travaux, avant de détailler notre mise en œuvre du map-matching pour recaler les trajectoires sur le réseau rou-

tier. En vue de pallier les difficultés liées à la détection et à la correction, pour de grands volumes de données, des erreurs issues du map-matching, nous proposons une mesure de qualité et une solution algorithmique pour le post-traitement des trajectoires. La mesure de qualité équivaut au ratio entre la longueur d'une trajectoire originale et celle de sa version matchée. Grâce à la distribution des valeurs de ce ratio nous fixons une plage en dehors de laquelle les trajectoires sont considérées comme potentiellement erronées. Une fois détectées, ces trajectoires sont scrutées automatiquement dans le but de déceler la présence de discontinuités ou de boucles illégitimes (dont l'existence n'est pas justifiée au regard des données GPS) qui sont systématiquement supprimées. Afin de disposer de valeurs fiables pour les paramètres nécessaires au fonctionnement de la solution algorithmique, nous avons développé un outil de visualisation et d'annotation des trajectoires visant à permettre à un opérateur humain d'évaluer la qualité du map-matching automatique et de proposer les corrections nécessaires le cas échéant. De ces travaux sur le prétraitement des trajectoires, nous retenons que l'automatisation du map-matching doit être associée à l'intervention d'opérateurs humains pour assurer à la fois un traitement efficace de grands volumes de trajectoires contraintes par un réseau routier et un certain niveau de qualité lors de leur recalage.

Disposer de trajectoires bien recalées sur le réseau routier n'est qu'une première étape dans le processus d'analyse, il faut ensuite pouvoir les comparer. À cet effet, nous avons consacré la première partie du chapitre 4 à la définition d'une nouvelle distance nommée EQRP qui comble les lacunes de deux des principales familles de distances pour trajectoires contraintes par un réseau, développées dans la littérature, tout en alliant leurs qualités. L'originalité de cette nouvelle distance, construite sur le modèle d'une distance d'édition, est de comparer les trajectoires en les considérant à la fois comme des séquences et des ensembles de nœuds. En pratique, EQRP apparie de manière optimale les nœuds des trajectoires en fonction de leur ordre d'apparition, puis évalue la distance entre nœuds appariés. Toutefois, lorsque l'appariement n'est pas possible, la distance retenue est calculée en considérant l'ensemble de nœuds formant chacune des trajectoires. Dans la deuxième partie de ce chapitre, nous réalisons une étude comparative de plusieurs mesures de distances pour les trajectoires contraintes par un réseau y compris celle nouvellement définie. Cette étude constitue une première pour ce type de distances. Les résultats obtenus classent EQRP parmi les distances les plus performantes de la littérature. Il est à retenir des travaux de ce chapitre qu'aucune distance ne consti-

tue une solution optimale pour la comparaison des trajectoires : chacune d'elles présente ses avantages et ses limites. Dès lors, il faut savoir choisir la distance la plus appropriée suivant le type d'application visée par l'analyse de trajectoires. En outre, les distances existantes ayant été définies pour des trajectoires se situant dans un espace géographique, il se pose la question de leur adéquation, en l'état, pour des trajectoires se déroulant dans un espace virtuel où les notions de distances n'ont pas nécessairement la même sémantique.

Le chapitre 5, décrit nos travaux concernant la délimitation des zones fonctionnelles dans les aires urbaines. Il s'agit d'une des applications de l'analyse de trajectoires qui a pour but de découper les villes en zones présentant de fortes interactions en leur sein et hétérogènes d'un point de vue socio-économique. La spécificité de la solution que nous proposons est de combiner les données de mobilité, en l'occurrence les trajectoires, aux données socio-économiques que sont les points d'intérêts des zones étudiées. Nous parvenons ce faisant, à faire émerger une structuration géographique latente des aires urbaines qui tient compte de leur composition socio-économique et qui reflète les habitudes de déplacement des habitants. Ce découpage s'oppose à ceux historiquement établis par l'administration qui sont essentiellement politiques. D'un point de vue technique, nous modélisons les interactions entre les unités spatiales composant le territoire étudié à l'aide de plusieurs graphes. Ensuite des communautés correspondant à des zones de fortes interactions sont extraites de ces graphes. Enfin, l'entropie des points d'intérêts au sein de ces communautés combinée à leur modularité sert à sélectionner le meilleur découpage en zone fonctionnelles. Ces travaux nous ont amené à repenser l'organisation des villes à travers une nouvelle approche qui utilise les données relatives à la mobilité des habitants. Il faut noter cependant que nous n'avons exploiter qu'un mode de mobilité, les taxis, et qu'un type de données socio-économiques, les points d'intérêts. Pour affiner les résultats générés, il serait pertinent d'intégrer d'autres modes de mobilité (vélos, transport en commun, marche) mais aussi des données socio-économiques plus variées (densité de population, indicateurs économiques).

6.2 Perspectives

Les réflexions et travaux menés au cours de cette thèse nous ont certes permis de solutionner différents problèmes liés à l'analyse de trajectoires contraintes par un réseau mais ils ont également fait surgir des idées d'améliorations possibles et

de nouveaux questionnements. Par ailleurs, de multiples aspects de ce domaine d'étude n'ont pas été abordés. Conscients de ces limites, nous proposons plusieurs perspectives à ce travail de recherche.

Vers un système semi-supervisé de map-matching.

La première concerne le prétraitement des trajectoires contraintes par un réseau routier et plus précisément l'étape de map-matching. Nous suggérons à cet effet la mise en œuvre d'un système de map-matching semi-supervisé, avec intervention d'un ou de plusieurs opérateurs humains. Le système envisagé serait composé d'un moteur de map-matching tel que ceux que nous avons présenté précédemment, secondé par un module de post-traitement autorisant la visualisation, l'évaluation et la correction des résultats du map-matching. Il comprendrait également une boucle de rétro-action entre le module de post-traitement et le moteur de map-matching afin de pouvoir identifier en amont les trajectoires potentiellement problématiques et d'exploiter les corrections réalisés en post-traitement pour leur recalage. Une étape intermédiaire consiste à corriger manuellement de manière un peu étendue les trajectoires avec les méthodes et l'outil développés dans le chapitre 3 afin de disposer d'un jeu de données de trajectoires parfaitement recalées.

Interactions entre réseau et trajectoires.

Notre seconde perspective est relative à l'étude de l'interaction entre le réseau et les trajectoires qui s'y déroulent. Il s'agirait de pouvoir formaliser l'influence du réseau sur les trajectoires qui y sont observées. Des travaux étudient déjà les liens entre les trajectoires et les propriétés topologiques, notamment en termes de centralité, des segments routiers empruntés. On pourrait aller plus loin en déterminant, par exemple, l'impact de la fermeture de portions de routes sur la fluidité du trafic, mais aussi évaluer comment la topologie des réseaux routiers influe sur la longueur des trajets.

Les jumeaux numériques.

De manière plus générale, nous proposons d'utiliser les trajectoires contraintes par un réseau routier dans le cadre du développement des jumeaux numériques (digital twins en anglais) des villes, concept particulièrement intéressant dans le cadre des villes intelligentes (smart-cities en anglais). Il s'agit de répliques virtuelles des aires urbaines qui permettent de simuler leur fonctionnement dans les moindres

détails, bien au-delà du simple réseau routier. Ces répliques peuvent aider à comprendre comment les villes se transforment et à anticiper les besoins futurs liés à ces transformations. Plusieurs villes françaises ont déjà construit leur jumeau numérique. Les trajectoires seraient dans ce contexte utiles pour reproduire le trafic routier sur une période déterminée grâce à une simulation multi-agent. Cela offrirait une vue d'ensemble de la mobilité urbaine permettant de mieux en déceler les failles pour les corriger.

Comparaison de trajectoires à support non-géographiques.

Ces trois premières propositions, se rapportent à l'analyse de trajectoires contraintes par un réseau routier, cependant ce ne sont pas les seules trajectoires à être contraintes par un réseau. Avec des trajectoires contraintes par un réseau qui ne sont pas liées à un espace géographique, comme celles se rapportant au monde virtuel, plusieurs questions se posent concernant la notion de distances entre trajectoires. Les distances dont nous avons fait état jusqu'ici supposent que les réseaux supportant les trajectoires s'inscrivent dans un espace géographique, ce qui permet d'associer la distance entre trajectoires à un concept d'éloignement géographique tangible. Toutefois, dès que l'on supprime l'hypothèse d'ancrage géographique des réseaux, la sémantique relative à la distance entre trajectoires change. A titre illustratif, prenons le cas d'un réseau de liens hypertextes entre les pages d'un site marchand : comment définit-on une bonne distance entre les trajectoires des visiteurs du site ? Nous proposons, dans un premier temps, de chercher à déterminer si certaines mesures de distance existantes demeurent pertinentes pour des trajectoires non géographiques et, si non, comment les généraliser.

Clustering de trajectoires contraintes par un réseau.

Une des méthodes de clustering utilisée dans la littérature consiste à plonger directement les trajectoires dans un espace euclidien en tenant compte des écarts relatifs entre elles. Cette approche permet de réduire notablement la complexité de l'opération de clustering mais elle est cependant tributaire de la qualité des distances utilisées. Nous proposons plutôt ici de plonger le graphe lui-même dans un espace euclidien, puis de déduire la représentation des trajectoires dans ce nouvel espace, avant de les regrouper via l'utilisation d'algorithmes classiques de clustering. En outre, on pourrait également chercher à savoir si le plongement des réseaux ancrés dans un espace géographique (assimilable à un espace euclidien)

produit de meilleurs résultats que celui des réseaux sans support géographique.

Prise en compte du temps et des données contextuelles

Au-delà des aspects déjà abordés, la dimension temporelle des trajectoires apparaît comme une source d'information essentielle à prendre en compte pour comprendre l'évolution dans le temps des activités qu'elles décrivent. L'intégration du temps dans l'analyse des trajectoires contraintes par un réseau pourrait se faire par le biais d'une représentation en réseaux multicouches, chaque couche correspondant à l'état du réseau pendant une période donnée. Ensuite, les outils adaptés au traitement de ce type de réseau seraient employés pour analyser les trajectoires à travers le temps. D'un autre côté, le temps pourrait également servir simplement de paramètre de filtrage permettant de sélectionner des groupes de trajectoires répondant à une contrainte temporelle prédéfinie avant leur analyse. Plus généralement, il faudrait songer à développer des méthodes d'analyse pour des trajectoires enrichies non seulement par le temps mais aussi par des données contextuelles (commentaires, achats effectués, likes, etc.).

Classification et recommandation de trajectoires.

Pour finir, nous suggérons également la conduite d'études sur la classification et la recommandation de trajectoires. C'est un axe de recherche d'actualité avec de multiples applications dans des secteurs tels que le commerce électronique, le tourisme ou la sécurité publique. Pour cette perspective, la piste des réseaux de neurones pour graphes (Graph neural networks) nous paraît pertinente surtout concernant l'opération de classification. En effet, ils permettent de réaliser efficacement des tâches d'apprentissage supervisé sur des graphes et ont connu un développement rapide ces dernières années.

Je me permets de terminer ce document par un message personnel. L'analyse de trajectoires contraintes par un réseau est un champ d'études à la fois florissant et passionnant scientifiquement, avec des applications qui impactent directement la vie des populations. C'est donc avec entrain que j'y ai consacré mes années de thèse. J'ose espérer que ce document vous aura permis d'en cerner quelques facettes et surtout aura nourrit votre curiosité. En attendant, je vous souhaite une excellente suite de parcours sur vos trajectoires respectives!!!

Annexe A

Publications

Conférences Internationales

— **N.L.J. Houssou**, J.-L. Guillaume, et A. Prigent. A graph based approach for functional urban areas delineation. In Proceedings of the 34rd Annual ACM Symposium on Applied Computing, SAC 2019, Limassol, Cyprus.

Conférences Nationales

— **N.L.J. Houssou**, J.-L. Guillaume, A. Prigent. Une approche basée graphes pour la détection de zones fonctionnelles urbaines. 19ème conférence Extraction et Gestion des Connaissances, EGC 2019, METZ, France. **Nominé pour le prix du meilleur article applicatif.**

— **N.L.J. Houssou**, J.-L. Guillaume, A. Prigent. Review and comparison of similarity measures and community detection algorithm for clustering of network constrained trajectories. 8ème conférence Modèles et Analyses Réseau : Approches Mathématiques et Informatique, Marami 2017, La Rochelle, France.

Posters

— **N.L.J. Houssou**, J.-L. Guillaume, A. Prigent. Analyse et modélisation de trajectoires utilisateurs dans des systèmes réels. Colloque des doctorants de deuxième année 2018, La Rochelle, France.

Bibliographie

[Ant05] J. Antikainen. The concept of functional urban area. findings of the espon project. *Informationen zur Raumentwicklung*, 7 :447–452, 2005.

Voir page 123.

[B+94] S.W. Bednarz et al. *Geography for Life : National Geography Standards*. Distributed by ERIC Clearinghouse (Washington, D.C.), 1994.

Voir page 128.

[BBD+16] Safaa Bataineh, Alfonso Bahillo, Luis Enrique Díez, Enrique Onieva, and Ikram Bataineh. Conditional random field-based offline map matching for indoor environments. *Sensors*, 16(8) :1302, 2016.

Voir page 34.

[BC78] E.A. Bender and E.R. Canfield. The asymptotic number of labeled graphs with given degree sequences. *Journal of Combinatorial Theory, Series A*, 24(3) :296 – 307, 1978.

Voir page 126.

[BC96] Donald J Berndt and James Clifford. Finding patterns in time series : a dynamic programming approach. In *Advances in knowledge discovery and data mining*, pages 229–248. 1996.

Voir page 43.

[BCW20] Wentao Bian, Ge Cui, and Xin Wang. A trajectory collaboration based map matching approach for low-sampling-rate gps trajectories. *Sensors*, 20(7) :2057, 2020.

Voir pages 37 et 38.

[BDG+06] U. Brandes, D. Delling, M. Gaertler, R. Goerke, M. Hoefer, Z. Nikoloski, and D. Wagner. Maximizing modularity is hard, 2006. cite arxiv :physics/0608255 Comment : 10 pages, 1 figure.

Voir page 129.

[BEF84] James C Bezdek, Robert Ehrlich, and William Full. Fcm : The fuzzy c-means clustering algorithm. *Computers & Geosciences*, 10(2-3) :191–203, 1984.

Voir page 57.

[Bel66] Richard Bellman. Dynamic programming. *Science*, 153(3731) :34–37, 1966.

Voir page 42.

[BGLL08] V.D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre. Fast unfolding of communities in large networks. *J. Stat. Mech*, page P10008, 2008.

Voir page 129.

[BGLR16] Philippe C Besse, Brendan Guillouet, Jean-Michel Loubes, and François Royer. Review and perspective for distance-based clustering of vehicle trajectories. *IEEE Transactions on Intelligent Transportation Systems*, 17(11) :3306–3317, 2016.

Voir pages 47 et 49.

[BGLR17] Philippe C Besse, Brendan Guillouet, Jean-Michel Loubes, and Francois Royer. Destination prediction by trajectory distribution-based model. *IEEE Transactions on Intelligent Transportation Systems*, 19(8) :2470–2481, 2017.

Voir page 59.

[BK+96] David Bernstein, Alain Kornhauser, et al. An introduction to map matching for personal navigation assistants. 1996.

Voir page 31.

[BLYZ16] Jie Bao, Ruiyuan Li, Xiuwen Yi, and Yu Zheng. Managing massive trajectories on the cloud. In *Proceedings of the 24th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 1–10, 2016.

Voir page 81.

[Bou11] Marc Boullé. Data grid models for preparation and modeling in supervised learning. *Hands-On Pattern Recognition : Challenges in Machine Learning*, 1 :99–130, 2011.

Voir page 66.

[BRML18] Carola Blazquez, Jana Ries, Pablo Andres Miranda, and Roberto Jesus Leon. An instance-specific parameter tuning approach using fuzzy logic for a post-processing topological map-matching algorithm. *IEEE Intelligent Transportation Systems Magazine*, 10(4) :87–97, 2018.

Voir page 39.

[CC15] Wonhee Cho and Eunmi Choi. A gps trajectory map-matching mechanism with dtg big data on the hbase system. In *Proceedings of the 2015 International Conference on Big Data Applications and Services*, pages 22–29, 2015.

Voir pages 36 et 81.

[CMWM10] Philippe Cudre-Mauroux, Eugene Wu, and Samuel Madden. Trajstore : An adaptive storage system for very large trajectory data sets. In *2010 IEEE 26th International Conference on Data Engineering (ICDE 2010)*, pages 109–120. IEEE, 2010.

Voir page 81.

[CN04] Lei Chen and Raymond Ng. On the marriage of lp-norms and edit distance. In *Proceedings of the Thirtieth international conference on Very large data bases-Volume 30*, pages 792–803, 2004.

Voir page 45.

[CÖO05] Lei Chen, M Tamer Özsu, and Vincent Oria. Robust and fast similarity search for moving object trajectories. In *Proceedings of the 2005 ACM SIGMOD international conference on Management of data*, pages 491–502, 2005.

Voir page 44.

[CSI15] CSIL. *Territorial Agenda 2020 put in practice. Enhancing the efficiency and effectiveness of Cohesion Policy by a place-based approach.* Publications Office of the European Union, 2015.

Voir page 123.

[DB79] David L Davies and Donald W Bouldin. A cluster separation measure. *IEEE transactions on pattern analysis and machine intelligence*, (2) :224–227, 1979.
Voir page 62.

[DCG+18] Xin Ding, Lu Chen, Yunjun Gao, Christian S Jensen, and Hujun Bao. Ultraman : a unified platform for big trajectory data management and analytics. *Proceedings of the VLDB Endowment*, 11(7) :787–799, 2018.
Voir page 81.

[DLR77] Arthur P Dempster, Nan M Laird, and Donald B Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society : Series B (Methodological)*, 39(1) :1–22, 1977.
Voir page 59.

[DRMB14] U. Demsar, J. Reades, E. Manley, and J.M. Batty. Edge-based communities for identification of functional regions in a taxi flow network. In *Extended Abstract Proceedings of the GIScience 2014*, pages 55–60, 2014.
Voir page 125.

[EKS+96] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Kdd*, volume 96, pages 226–231, 1996.
Voir page 58.

[EMR12] Mohamed Khalil El Mahrsi and Fabrice Rossi. Graph-based approaches to clustering network-constrained trajectory data. In *International Workshop on New Frontiers in Mining Complex Patterns*, pages 124–137. Springer, 2012.
Voir page 65.

[ENB05] Maan E El Najjar and Philippe Bonnifait. A road-matching method for precise vehicle localization using belief theory and kalman filtering. *Autonomous Robots*, 19(2) :173–191, 2005.
Voir page 33.

[EOSH12] Michael R Evans, Dev Oliver, Shashi Shekhar, and Francis Harvey. Summarizing trajectories into k-primary corridors : a summary of

results. In *Proceedings of the 20th International Conference on Advances in Geographic Information Systems*, pages 454–457, 2012.
Voir page 51.

[fECoO02] Organization for Economic Co-operation and Development (OECD). Redefining territories, the functional regions, 2002.
Voir page 123.

[FF11] C.J.Q. Farmer and A.S. Fotheringham. Network-based functional regions. *Environment and Planning A : Economy and Space*, 43(11) :2723–2741, 2011.
Voir pages 122, 123, 125, 126, 128 et 135.

[FGV19] Matteo Francia, Enrico Gallinucci, and Federico Vitali. Map-matching on big data : A distributed and efficient algorithm with a hidden markov model. In *2019 42nd International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, pages 1238–1243. IEEE, 2019.
Voir pages 36 et 81.

[FL95] Christos Faloutsos and King-Ip Lin. Fastmap : A fast algorithm for indexing, data-mining and visualization of traditional and multimedia datasets. In *Proceedings of the 1995 ACM SIGMOD international conference on Management of data*, pages 163–174, 1995.
Voir page 64.

[FLW04] Mengyin Fu, Jie Li, and Meiling Wang. A hybrid map matching algorithm based on fuzzy comprehensive judgment. In *Proceedings. The 7th International IEEE Conference on Intelligent Transportation Systems (IEEE Cat. No. 04TH8749)*, pages 613–617. IEEE, 2004.
Voir page 32.

[For10] S. Fortunato. Community detection in graphs. *Physics Reports*, 486(3-5) :75 – 174, 2010.
Voir page 129.

[FRM94] Christos Faloutsos, Mudumbai Ranganathan, and Yannis Manolopoulos. Fast subsequence matching in time-series databases. *Acm Sigmod Record*, 23(2) :419–429, 1994.
Voir page 40.

[FZWZ15] K. Fan, D. Zhang, Y. Wang, and S. Zhao. Discovering urban social functional regions using taxi trajectories. In *Proceedings of the 2015 IEEE 12th Intl Conf on Ubiquitous Intelligence and Computing*, pages 356–359, Aug 2015.

Voir pages 122 et 125.

[GCR05] D. Gfeller, J.-C. Chappelier, and P. De Los Rios. Finding instabilities in the community structure of complex networks. *Phys. Rev. E*, 72 :056135, Nov 2005.

Voir page 137.

[GDM+12] Chong Yang Goh, Justin Dauwels, Nikola Mitrovic, Muhammad Tayyab Asif, Ali Oran, and Patrick Jaillet. Online map-matching based on hidden markov model for real-time traffic sensing applications. In *2012 15th International IEEE Conference on Intelligent Transportation Systems*, pages 776–781. IEEE, 2012.

Voir page 33.

[GJC17] S. Gao, K. Janowicz, and H. Couclelis. Extracting urban functional regions from points of interest and human activities on location-based social networks. *Transactions in GIS*, 21(3) :446–467, 2017.

Voir pages 123 et 124.

[GMP05] Peter D Grünwald, In Jae Myung, and Mark A Pitt. *Advances in minimum description length : Theory and applications*. MIT press, 2005.

Voir page 58.

[Gui16] Brendan Guillouet. Apprentissage statistique : application au trafic routier à partir de données structurées et aux données massives, 2016.

Voir page 47.

[HA85] Lawrence Hubert and Phipps Arabie. Comparing partitions. *Journal of classification*, 2(1) :193–218, 1985.

Voir page 108.

[Hau14] Felix Hausdorff. *Grundzüge der mengenlehre*, volume 7. von Veit, 1914.

Voir page 47.

[HCL15] H. Han, X. Chen, and Y. Long. Discovering functional zones using bus smart card data and points of interest in beijing. *CoRR*, abs/1503.03131, 2015.

Voir page 122.

[HG97] Paul S Heckbert and Michael Garland. Survey of polygonal surface simplification algorithms. Technical report, Carnegie-Mellon Univ Pittsburgh PA School of Computer Science, 1997.

Voir page 103.

[HKL05] Jung-Rae Hwang, Hye-Young Kang, and Ki-Joune Li. Spatio-temporal similarity analysis between trajectories on road networks. In *International Conference on Conceptual Modeling*, pages 280–289. Springer, 2005.

Voir page 50.

[HLO13] Binh Han, Ling Liu, and Edward Omiecinski. Road-network aware trajectory clustering : Integrating locality, flow, and density. *IEEE Transactions on Mobile Computing*, 14(2) :416–429, 2013.

Voir page 63.

[HLO17] Binh Han, Ling Liu, and Edward Omiecinski. A systematic approach to clustering whole trajectories of mobile objects in road networks. *IEEE Transactions on Knowledge and Data Engineering*, 29(5) :936–949, 2017.

Voir page 65.

[Jac01] Paul Jaccard. Distribution de la flore alpine dans le bassin des dranses et dans quelques régions voisines. *Bull Soc Vaudoise Sci Nat*, 37 :241–272, 1901.

Voir page 65.

[Kar07] C. Karlsson. Clusters, Functional Regions and Cluster Policies. Working Paper Series in Economics and Institutions of Innovation 84, Royal Institute of Technology, CESIS - Centre of Excellence for Science and Innovation Studies, February 2007.

Voir page 123.

[KCM+15] Matěj Kubička, Arben Cela, Philippe Moulin, Hugues Mounier, and Silviu-Iulian Niculescu. Dataset for testing and training of map-

matching algorithms. In *2015 IEEE Intelligent Vehicles Symposium (IV)*, pages 1088–1093. IEEE, 2015.
Voir page 68.

[KH90] Joe K Kearney and Stuart Hansen. Stream editing for animation. Technical report, IOWA UNIV IOWA CITY DEPT OF COMPUTER SCIENCE, 1990.
Voir page 43.

[KM15] Jiwon Kim and Hani S Mahmassani. Spatial and temporal characterization of travel patterns in a traffic network using vehicle trajectories. *Transportation Research Procedia*, 9 :164–184, 2015.
Voir page 58.

[KPZF08] Ahmed Kharrat, Iulian Sandu Popa, Karine Zeitouni, and Sami Faiz. Clustering algorithm for network constraint trajectories. In *Headway in Spatial Data Handling*, pages 631–647. Springer, 2008.
Voir pages 53 et 63.

[KR09] Leonard Kaufman and Peter J Rousseeuw. *Finding groups in data : an introduction to cluster analysis*, volume 344. John Wiley & Sons, 2009.
Voir page 57.

[KSBE18] Robert Krüger, Georgi Simeonov, Fabian Beck, and Thomas Ertl. Visual interactive map matching. *IEEE transactions on visualization and computer graphics*, 24(6) :1881–1892, 2018.
Voir page 38.

[KXI20] Satoshi Koide, Chuan Xiao, and Yoshiharu Ishikawa. Fast subtrajectory similarity search in road networks under weighted edit distance constraints. *arXiv preprint arXiv :2006.05564*, 2020.
Voir page 54.

[LGGL15] X. Liu, L. Gong, Y. Gong, and Y. Liu. Revealing travel patterns and city structure with taxi trip data. *Journal of Transport Geography*, 43 :78 – 90, 2015.
Voir page 128.

[LHK+13] Yang Li, Qixing Huang, Michael Kerber, Lin Zhang, and Leonidas Guibas. Large-scale joint map matching of gps traces. In *Procee-*

dings of the 21st ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, pages 214–223, 2013.

Voir page 37.

[LHLG08] Jae-Gil Lee, Jiawei Han, Xiaolei Li, and Hector Gonzalez. Traclass : trajectory classification using hierarchical region-based and trajectory-based clustering. *Proceedings of the VLDB Endowment*, 1(1) :1081–1094, 2008.

Voir page 58.

[LHW07] Jae-Gil Lee, Jiawei Han, and Kyu-Young Whang. Trajectory clustering : a partition-and-group framework. In *Proceedings of the 2007 ACM SIGMOD international conference on Management of data*, pages 593–604, 2007.

Voir page 58.

[LRB+17] Ruiyuan Li, Sijie Ruan, Jie Bao, Yanhua Li, Yingcai Wu, and Yu Zheng. Querying massive trajectories by path on the cloud. In *Proceedings of the 25th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 1–4, 2017.

Voir page 81.

[LS15] Y. Long and Z. Shen. *Discovering Functional Zones Using Bus Smart Card Data and Points of Interest in Beijing*, pages 193–217. Springer International Publishing, Cham, 2015.

Voir page 124.

[LWXG12] Y. Liu, F. Wang, Y. Xiao, and S. Gao. Urban land uses and traffic 'source-sink areas' : Evidence from gps-enabled taxi data in shanghai. *Landscape and Urban Planning*, 106(1) :73 – 87, 2012.

Voir page 128.

[LZG+20] Minshi Liu, Ling Zhang, Junlian Ge, Yi Long, and Weitao Che. Map matching for urban high-sampling-frequency gps trajectories. *ISPRS International Journal of Geo-Information*, 9(1) :31, 2020.

Voir page 30.

[MAZE+14] Sebastian Mattheis, Kazi Khaled Al-Zahid, Birgit Engelmann, Andreas Hildisch, Stefan Holder, Olexiy Lazarevych, Daniel Mohr, Felix Sedlmeier, and Richard Zinck. Putting the car on the map : a scalable

map matching system for the open source community. *Informatik 2014*, 2014.

Voir pages 36 et 81.

[MBHW19] Adam Millard-Ball, Robert C Hampshire, and Rachel R Weinberger. Map-matching poor-quality gps data in urban environments : the pgmapmatch package. *Transportation Planning and Technology*, 42(6) :539–553, 2019.

Voir page 38.

[MMGF+13] L. Moreira-Matias, J. Gama, M. Ferreira, J. Mendes-Moreira, and L. Damas. Predicting taxi-passenger demand using streaming data. *IEEE Transactions on Intelligent Transportation Systems*, 14(3) :1393–1402, Sept 2013.

Voir page 72.

[MSMEB15] Nehal Magdy, Mahmoud A Sakr, Tamer Mostafa, and Khaled El-Bahnasy. Review on trajectory similarity measures. In *2015 IEEE seventh international conference on Intelligent Computing and Information Systems (ICICIS)*, pages 613–619. IEEE, 2015.

Voir page 96.

[New06] M.E.J. Newman. Modularity and community structure in networks. *Proceedings of the National Academy of Sciences*, 103(23) :8577–8582, 2006.

Voir page 126.

[NK06] Andrew Naftel and Shehzad Khalid. Motion trajectory learning in the dft-coefficient feature space. In *Fourth IEEE International Conference on Computer Vision Systems (ICVS'06)*, pages 47–47. IEEE, 2006.

Voir page 61.

[NK09] Paul Newson and John Krumm. Hidden markov map matching through noise and sparseness. In *Proceedings of the 17th ACM SIGSPATIAL international conference on advances in geographic information systems*, pages 336–343, 2009.

Voir pages 33, 35 et 77.

[NP06] Mirco Nanni and Dino Pedreschi. Time-focused clustering of trajectories of moving objects. *Journal of Intelligent Information Systems*, 27(3) :267–289, 2006.

Voir pages 49 et 58.

[OQN03] Washington Y Ochieng, Mohammed Quddus, and Robert B Noland. Map-matching in complex urban road networks. *Revista Brasileira de Cartografia*, 55(2), 2003.

Voir page 32.

[PKK+09] Nikos Pelekis, Ioannis Kopanakis, Evangelos Kotsifakos, Elias Frentzos, and Yannis Theodoridis. Clustering trajectories of moving objects in an uncertain world. In *2009 Ninth IEEE international conference on data mining*, pages 417–427. IEEE, 2009.

Voir page 57.

[PSR+14] T. Pei, S. Sobolevsky, C. Ratti, S.-L. Shaw, T. Li, and C. Zhou. A new insight into land use classification based on aggregated mobile phone data. *International Journal of Geographical Information Science*, 28(9) :1988–2007, 2014.

Voir page 128.

[QLL+11] G. Qi, X. Li, S. Li, G. Pan, and Z. Wang. Measuring social functions of city regions from large-scale taxi behaviors. In *Proceedings of the IEEE International Conference on Pervasive Computing and Communications Workshops*, (PERCOM Workshops), pages 384–388, 2011.

Voir page 122.

[QNO06] Mohammed A Quddus, Robert B Noland, and Washington Y Ochieng. A high accuracy fuzzy logic based map matching algorithm for road transport. *Journal of Intelligent Transportation Systems*, 10(3) :103–115, 2006.

Voir page 32.

[QON07] Mohammed A Quddus, Washington Y Ochieng, and Robert B Noland. Current map-matching algorithms for transport applications : State-of-the art and future research directions. *Transportation research part c : Emerging technologies*, 15(5) :312–328, 2007.

Voir pages 31 et 35.

[QOZN03] Mohammed A Quddus, Washington Yotto Ochieng, Lin Zhao, and Robert B Noland. A general map matching algorithm for transport telematics applications. *GPS solutions*, 7(3) :157–167, 2003.

Voir page 32.

[RH10] Gook-Pil Roh and Seung-won Hwang. Nncluster : An efficient clustering algorithm for road network trajectories. In *International Conference on Database Systems for Advanced Applications*, pages 47–61. Springer, 2010.

Voir pages 52 et 62.

[RLB+18] Sijie Ruan, Ruiyuan Li, Jie Bao, Tianfu He, and Yu Zheng. Cloudtp : A cloud-based flexible trajectory preprocessing framework. In *2018 IEEE 34th International Conference on Data Engineering (ICDE)*, pages 1601–1604. IEEE, 2018.

Voir page 81.

[Rou87] Peter J Rousseeuw. Silhouettes : a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics*, 20 :53–65, 1987.

Voir page 108.

[RRCM18] Efstratios Rappos, Stephan Robert, and Philippe Cudré-Mauroux. A force-directed approach for offline gps trajectory map matching. In *Proceedings of the 26th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 319–328, 2018.

Voir page 34.

[SBVLK09] Tobias Schreck, Jürgen Bernard, Tatiana Von Landesberger, and Jörn Kohlhammer. Visual cluster analysis of trajectory data with interactive kohonen maps. *Information Visualization*, 8(1) :14–29, 2009.

Voir page 61.

[SCW+17] Shuo Shang, Lisi Chen, Zhewei Wei, Christian S Jensen, Kai Zheng, and Panos Kalnis. Trajectory similarity join in spatial networks. 2017.

Voir pages 52 et 69.

[Sha48] C.E. Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27(3) :379–423, 7 1948.

Voir page 129.

[SLB18] Zeyuan Shang, Guoliang Li, and Zhifeng Bao. Dita : Distributed in-memory trajectory analytics. In *Proceedings of the 2018 International Conference on Management of Data*, pages 725–740, 2018.

Voir pages 54 et 69.

[SLZ+20] Han Su, Shuncheng Liu, Bolong Zheng, Xiaofang Zhou, and Kai Zheng. A survey of trajectory distance measures and performance evaluation. *The VLDB Journal*, 29(1) :3–32, 2020.

Voir pages 41, 49, 69, 96, 100, 105 et 118.

[SSZZ14] Renchu Song, Weiwei Sun, Baihua Zheng, and Yu Zheng. Press : A novel framework of trajectory compression in road networks. *arXiv preprint arXiv :1402.1546*, 2014.

Voir page 28.

[SVL14] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.

Voir page 60.

[SW80] Arthur C Sanderson and Andrew KC Wong. Pattern trajectory analysis of nonstationary multivariate data. *IEEE Transactions on Systems, Man, and Cybernetics*, 10(7) :384–392, 1980.

Voir page 40.

[THL+18] W. Tu, Z. Hu, L. Li, J. Cao, J. Jiang, Q. Li, and Q. Li. Portraying urban functional zones by coupling remote sensing imagery and human sensing data. *Remote Sensing*, 10(1), 2018.

Voir page 124.

[TPN+09] Eleftherios Tiakas, AN Papadopoulos, Alexandros Nanopoulos, Yannis Manolopoulos, Dragan Stojanovic, and Slobodanka Djordjevic-Kajan. Searching for similar trajectories in spatial networks. *Journal of Systems and Software*, 82(5) :772–788, 2009.

Voir page 51.

[VKG02] Michail Vlachos, George Kollios, and Dimitrios Gunopulos. Discovering similar multidimensional trajectories. In *Proceedings 18th international conference on data engineering*, pages 673–684. IEEE, 2002.

Voir page 49.

[WBC+18] Sheng Wang, Zhifeng Bao, J Shane Culpepper, Zizhe Xie, Qizhi Liu, and Xiaolin Qin. Torch : A search engine for trajectory data. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 535–544, 2018.

Voir pages 54 et 69.

[WBC+19] Sheng Wang, Zhifeng Bao, J Shane Culpepper, Timos Sellis, and Xiaolin Qin. Fast large-scale trajectory clustering. *Proceedings of the VLDB Endowment*, 13(1) :29–42, 2019.

Voir page 53.

[WBK00] Christopher E White, David Bernstein, and Alain L Kornhauser. Some map matching algorithms for personal navigation assistants. *Transportation research part c : emerging technologies*, 8(1-6) :91–108, 2000.

Voir page 31.

[WGDQ18] Y. Wang, Y. Gu, M. Dou, and M. Qiao. Using spatial semantics and interactions to identify urban functional regions. *ISPRS International Journal of Geo-Information*, 7(4), 2018.

Voir pages 122 et 124.

[WKBL09] Jung-Im Won, Sang-Wook Kim, Ji-Haeng Baek, and Junghoon Lee. Trajectory clustering in road network environment. In *2009 IEEE Symposium on Computational Intelligence and Data Mining*, pages 299–305. IEEE, 2009.

Voir page 64.

[WLH+17] Hongyu Wang, Jin Li, Zhenshan Hou, Ruochen Fang, Wenbo Mei, and Jian Huang. Research on parallelized real-time map matching algorithm for massive gps data. *Cluster Computing*, 20(2) :1123–1134, 2017.

Voir pages 28 et 33.

[WLWK08] Ho Chung Wu, Robert Wing Pong Luk, Kam Fai Wong, and Kui Lam Kwok. Interpreting tf-idf term weights as making relevance decisions. *ACM Transactions on Information Systems (TOIS)*, 26(3) :1–37, 2008.

Voir page 66.

[WN16] Xuemei Wang and Wenbo Ni. An improved particle filter and its application to an ins/gps integrated navigation system in a serious noisy scenario. *Measurement Science and Technology*, 27(9) :095005, 2016.

Voir page 33.

[WQCZ18] Yulong Wang, Kun Qin, Yixiang Chen, and Pengxiang Zhao. Detecting anomalous trajectories and behavior patterns using hierarchical clustering from taxi gps data. *ISPRS International Journal of Geo-Information*, 7(1) :25, 2018.

Voir page 60.

[WSB76] Michael S Waterman, Temple F Smith, and William A Beyer. Some biological sequence metrics. *Advances in Mathematics*, 20(3) :367–387, 1976.

Voir page 99.

[WWT+16] Y. Wang, T. Wang, M.-H. Tsou, H. Li, W. Jiang, and F. Guo. Mapping dynamic urban land use patterns with crowdsourced geo-tagged social media (sina-weibo) and commercial points of interest collections in beijing, china. *Sustainability*, 8(11), 2016.

Voir page 128.

[WZX+14] Haozhou Wang, Kai Zheng, Jiajie Xu, Bolong Zheng, Xiaofang Zhou, and Shazia Sadiq. Sharkdb : An in-memory column-oriented trajectory storage. In *Proceedings of the 23rd ACM international conference on conference on information and knowledge management*, pages 1409–1418, 2014.

Voir page 81.

[XJJF19] LAI Xin, CHEN Jianhua, CAO Jingjing, and XIA Fei. Map matching integration algorithm based on historical trajectory data. In *2019 IEEE Symposium on Product Compliance Engineering-Asia (ISPCE-CN)*, pages 1–6. IEEE, 2019.

Voir pages 34 et 87.

[Y+10] Meng Yu et al. Improved positioning of land vehicle in its using digital map and other accessory information. 2010.

Voir page 32.

[YG18] Can Yang and Gyozo Gidofalvi. Fast map matching, an algorithm integrating hidden markov model with precomputation. *International Journal of Geographical Information Science*, 32(3) :547–570, 2018.

Voir page 39.

[YL19] Haitao Yuan and Guoliang Li. Distributed in-memory trajectory similarity search and join on road network. In *2019 IEEE 35th international conference on data engineering (ICDE)*, pages 1262–1273. IEEE, 2019.

Voir page 54.

[YSWZ18] Yifang Yin, Rajiv Ratn Shah, Guanfeng Wang, and Roger Zimmermann. Feature-based map matching for low-sampling-rate gps trajectories. *ACM Transactions on Spatial Algorithms and Systems (TSAS)*, 4(2) :1–24, 2018.

Voir page 38.

[YZX12] J. Yuan, Y. Zheng, and X. Xie. Discovering regions of different functions in a city using human mobility and pois. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '12, pages 186–194, New York, NY, USA, 2012. ACM.

Voir pages 122 et 124.

[YZXS11] Jing Yuan, Yu Zheng, Xing Xie, and Guangzhong Sun. T-drive : Enhancing driving directions with taxi drivers' intelligence. *IEEE Transactions on Knowledge and Data Engineering*, 25(1) :220–232, 2011.

Voir page 27.

[YZZ+11] Jing Yuan, Yu Zheng, Liuhang Zhang, XIng Xie, and Guangzhong Sun. Where to find my next passenger. In *Proceedings of the 13th international conference on Ubiquitous computing*, pages 109–118, 2011.

Voir page 28.

[YZZ+18] Di Yao, Chao Zhang, Zhihua Zhu, Qin Hu, Zheng Wang, Jianhui Huang, and Jingping Bi. Learning deep representation for trajectory clustering. *Expert Systems*, 35(2) :e12252, 2018.
Voir page 60.

[YZZX12] Nicholas Jing Yuan, Yu Zheng, Liuhang Zhang, and Xing Xie. T-finder : A recommender system for finding passengers and vacant taxis. *IEEE Transactions on knowledge and data engineering*, 25(10) :2390–2403, 2012.
Voir page 28.

[ZADE+17] Ye Zhao, Shamal Al-Dohuki, Thomas Eynon, Farah Kamw, David Sheets, Chao Ma, Yueqi Hu, Xinyue Ye, and Jing Yang. Trajanalytics : A web-based visual analytics software of urban trajectory data. 2017.
Voir page 81.

[ZCL+10] Yu Zheng, Yukun Chen, Quannan Li, Xing Xie, and Wei-Ying Ma. Understanding transportation modes based on gps data for web applications. *ACM Transactions on the Web (TWEB)*, 4(1) :1–36, 2010.
Voir page 28.

[Zhe15] Yu Zheng. Trajectory data mining : an overview. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 6(3) :1–41, 2015.
Voir page 27.

[ZHG17] Lei Zhu, Jacob R Holden, and Jeffrey D Gonder. Trajectory segmentation map-matching approach for large-scale, high-resolution gps data. *Transportation Research Record*, 2645(1) :67–75, 2017.
Voir pages 28 et 31.

[ZLL09] Tianzhu Zhang, Hanqing Lu, and Stan Z Li. Learning semantic scene models by object classification and trajectory clustering. In *2009 IEEE conference on computer vision and pattern recognition*, pages 1940–1947. IEEE, 2009.
Voir page 59.

[ZLL18] Dongzhi Zhang, Kyungmi Lee, and Ickjai Lee. Hierarchical trajectory clustering for spatio-temporal periodic pattern mining. *Expert Systems with Applications*, 92 :1–11, 2018.

Voir page 60.

[ZLW+14] Y. Zhi, Y. Liu, S. Wang, M. Deng, J. Gao, and H. Li. Urban spatial-temporal activity structures : a new approach to inferring the intra-urban functional regions via social media check-in data. *CoRR*, abs/1412.7253, 2014.

Voir pages 123 et 124.

[ZYZ+15] Z. Zhu, J. Yang, C. Zhong, J. Seiter, and G. Tröster. Learning functional compositions of urban spaces with crowd-augmented travel survey data. In *Proceedings of the 23rd SIGSPATIAL International Conference on Advances in Geographic Information Systems*, SIGSPATIAL '15, pages 22 :1–22 :10, New York, NY, USA, 2015. ACM.

Voir pages 122 et 124.

[ZZXM09] Yu Zheng, Lizhu Zhang, Xing Xie, and Wei-Ying Ma. Mining interesting locations and travel sequences from gps trajectories. In *Proceedings of the 18th international conference on World wide web*, pages 791–800, 2009.

Voir page 60.

Analyse et modélisation de trajectoires d'utilisateurs dans des systèmes réels

Résumé : Les progrès technologiques récents ont accéléré la numérisation de nos sociétés et impulsé un usage intensif de terminaux et de plateformes digitales. L'utilisation de ces outils qui touchent pratiquement à tous les aspects de nos vies induit la production et le stockage massif de trajectoires numériques. Ces dernières décrivent les mouvements des utilisateurs dans le temps et dans l'espace et sont souvent contraintes par un réseau qui limite les déplacements possibles. Dans cette thèse, nos travaux ont dans un premier temps, porté sur les problèmes relatifs au recalage sur un réseau routier des trajectoires réelles et bruitées de mobilité urbaine. Dans un second temps, nous nous sommes intéressés à la notion de distance entre trajectoires contraintes qui est indispensable à l'analyse des données de ce type. Nous avons également abordé la problématique plus appliquée de la fusion de données pour la détection de zones fonctionnelles urbaines. La singularité de notre travail réside non seulement dans l'emploi de trajectoires réelles, notamment de mobilité urbaine, mais aussi dans l'utilisation d'outils provenant de la théorie des graphes, outils qui facilitent l'intégration des contraintes et spécificités liées aux réseaux.

Mots clés : réseaux complexes, modélisation par des graphes, analyse de trajectoires, distances entre trajectoires, mobilité urbaine, map-matching, zones fonctionnelles.

Analysis and modeling of users' trajectories in real systems

Abstract : Recent technological progress has accelerated the digitization of our societies and has led to an intensive use of digital terminals and platforms. The use of these tools, which touch almost every aspect of our lives, induces a massive production and storage of digital trajectories. These trajectories describe users' movements in time and space and are often constrained by a network that limits the possible movements. In this thesis, our work was firstly focused on the problems related to the map-matching of real and noisy urban mobility trajectories. Secondly, we have been interested in the notion of distance between constrained trajectories which is essential for the analysis of such data. We have also addressed the more applied problem of data fusion for the detection of urban functional areas. The singularity of our work lies not only in the use of real trajectories, in particular urban mobility trajectories, but also in the employment of tools coming from graph theory, tools that facilitate the integration of constraints and specific features related to networks.

Keywords : complex networks, graph modeling, trajectory analysis, trajectory distances, urban mobility, map-matching, functional areas.

Laboratoire Informatique, Image, Interaction
Faculté des Sciences et Technologies
Avenue Michel Crépeau
17042 La Rochelle - CEDEX 01 - France