

École polytechnique de Louvain

Modélisation temporelle du trafic pour des études de résilience sur le réseau routier belge

Auteur : **Timothée JACQUIN**

Promoteur : **Luca SGAMBI**

Lecteurs : **Daniela PERROTTI, Jean-François RONDEAUX**

Année académique 2018–2019

Master [120] : ingénieur civil des constructions

Table des matières

1	L'importance d'un réseau routier en bon état	5
1.1	Introduction	6
1.2	Rôles du réseau routier dans l'économie	6
1.3	Entretien	7
1.4	Impact social	9
1.5	Défis énergétique et environnemental	9
2	Etat de l'art sur la résilience	11
2.1	Définition du concept de résilience	12
2.2	La résilience selon quatre concepts	14
2.2.1	La résilience comme rebond après un traumatisme	14
2.2.2	La résilience synonyme de robustesse	15
2.2.3	La résilience en opposition avec la fragilité	15
2.2.4	La résilience comme adaptabilité durable	15
2.3	Résilience écologique	16
2.3.1	Etude de Robert MacArthur	16
2.3.2	Etude de Stuart L. Pimm	18
2.3.3	Etude de C.S Holling	18
2.4	Résilience économique	19
2.5	Résilience en ingénierie au service des transports	20
2.6	La vulnérabilité et la gestion des risques	21
3	Etat et hypothèses du modèle précédent et pistes d'amélioration	23
3.1	Résumé de la simulation de Joachim L'Hoir et Adrien Tellier	24
3.1.1	Représentation du réseau	24
3.1.2	Hypothèses employées	27
3.2	Amélioration du modèle	27
4	Etude du territoire	29
4.1	Introduction	30
4.2	Enquête sur la mobilité en Belgique	30
4.2.1	MOBEL	30
4.2.2	BELDAM	30
4.2.3	MONITOR	31
4.3	Enquête sur les déplacements domicile travail	32
4.4	Circulation sur le réseau routier	32
4.5	Analyse des données	32

5	Construction du modèle	35
5.1	Détermination des trajets	36
5.1.1	Détermination de la ville de départ	37
5.1.2	Détermination de la ville d'arrivée	40
5.2	Trouver le chemin le plus court	42
5.2.1	Mesure de complexité d'un algorithme	42
5.2.2	Algorithme de Dijkstra	43
5.2.3	Algorithme de Bellman-Ford	47
5.2.4	Algorithme A*	47
5.2.5	La fonction Dijkstra de Matlab	47
5.3	Modélisation de la congestion	48
5.3.1	Static and dynamic traffic assignment	48
5.3.2	Flux de trafic	48
5.4	Heure de départ	52
5.5	Nombre de véhicules à simuler	54
5.6	Prise en compte des camions	55
6	Modélisation et analyse des résultats	57
6.1	Explicatif du code Matlab	58
6.1.1	Impact du CO ₂	58
6.2	Analyse du comportement général	59
6.3	Analyse suite à la réduction du nombre de bandes de circulation	61
6.4	Analyse de la fermeture d'un tronçon	62
6.4.1	Fermeture de la E42 entre Mons et La Louvière	62
6.4.2	Fermeture de la E42 entre La Louvière et Chapelle-lez-Herlaimont	67
6.4.3	Fermeture de la E42 entre Chapelle-lez-Herlaimont et Gosselies	70
6.4.4	Fermeture de la E42 entre Gosselies et Fleurus	71
6.4.5	Fermeture de la E42 entre Fleurus et Sombreffe	72
6.4.6	Fermeture de la E420 entre Nivelles et Gosselies	74
6.4.7	Fermeture de la E411 entre Perwez et Louvain-la-Neuve	75
6.4.8	Fermeture de la E411 entre Wellin et Ciney	77
6.4.9	Fermeture de la E17 entre Gand et Lokeren	78
6.4.10	Récapitulatif	80
7	Critique, pistes d'amélioration et conclusions	82
7.1	Critique du modèle	83
7.2	Pistes d'amélioration	83
7.3	Conclusion	84
A	Distribution statistique	87
A.1	Définition	87
A.2	Exemple de distribution statistique	90
A.2.1	Distribution uniforme	91
A.2.2	Distribution de Bernoulli	91
A.2.3	Distribution binomiale	91
A.2.4	Distribution géométrique	92

A.2.5	Distribution de Poisson	93
A.2.6	Distribution uniforme	93
A.2.7	Distribution normale	94
A.2.8	Distribution Gamma	94
A.2.9	Distribution exponentielle	95
A.2.10	Distribution bimodale	95
B	Théorie des graphes	96
B.1	Définition	96
B.2	Graphes particuliers	97
B.3	Représentation des données	98
B.3.1	Matrice d'adjacence	98
B.3.2	Matrice d'incidence	98
B.3.3	Liste des voisins	99
C	Matrice xyv	100
D	Tutoriel du code Matlab	105
D.1	Initialisation	105
D.2	Détermination des trajets	106
D.3	Simulation temporelle sans fermeture	107
D.4	Simulation temporelle avec fermeture	109
D.5	Analyse	109
E	Code Matlab	111
E.1	Script principal	111
E.2	liste_trajet.m	121
E.3	simulation_temporelle.m	127
E.4	image_reseau.m	134
E.5	Fonction Dijkstra	136
E.6	Fonction distkm.m	144

Table des figures

1.1	Variation des coûts dans le temps	8
2.1	Pluridisciplinarité de la résilience et des concepts associés	12
2.2	Représentation de la communauté étudiée	16
2.3	Représentation d'une communauté	17
2.4	Matrice de risque	22
3.1	Carte de la Belgique	25
3.2	Réseau routier de la simulation	26
4.1	Distribution des distances de déplacement	34
5.1	Répartition de la population	38
5.2	Représentation des villes hôtes	39
5.3	Méthode de sélection des villes	39
5.4	Distribution des villes d'arrivée si Bruxelles est la ville de départ	40
5.5	Graphe du test de l'algorithme de Dijkstra	44
5.6	Courbe du flux f en fonction de la densité de voitures ρ	50
5.7	Courbe modélisée du flux f en fonction de la densité ρ	51
5.8	Courbe modélisée de la vitesse en fonction de la densité ρ	52
5.9	Répartition horaire des déplacements d'un jour ouvrable	53
5.10	Distribution horaire des départs dans la simulation	54
5.11	Axes des flux de poids lourds traversant Belgique	56
6.1	Nombre de véhicules sur chaque tronçon	59
6.2	Zones d'embouteillage sur chaque tronçon	60
6.3	QR code	61
6.4	Vitesse moyenne sur le tronçon	62
6.5	Vue d'ensemble sur la Belgique en tant normal.	64
6.6	Vue d'ensemble sur la Belgique après la fermeture du tronçon.	64
6.7	Zoom sur la situation locale en tant normal.	65
6.8	Zoom sur la situation locale après la fermeture du tronçons.	66
6.9	Vue d'ensemble sur la Belgique	68
6.10	Vue d'ensemble sur la Belgique après la fermeture	68
6.11	Zoom sur la situation locale	68
6.12	Zoom sur la situation locale après la fermeture	68
6.13	Vitesse moyenne entre Ecaussinnes (80) et Chapelle-lez-Herlaimont (88) avant la fermeture	69
6.14	Vitesse moyenne entre Ecaussinnes (80) et Chapelle-lez-Herlaimont (88) après la fermeture	69
6.15	Vue d'ensemble sur la Belgique	70

6.16	Vue d'ensemble sur la Belgique après la fermeture	70
6.17	Zoom sur la situation locale	71
6.18	Zoom sur la situation locale après la fermeture	71
6.19	Vue d'ensemble sur la Belgique	72
6.20	Vue d'ensemble sur la Belgique après la fermeture	72
6.21	Zoom sur la situation locale	72
6.22	Zoom sur la situation locale après la fermeture	72
6.23	Vue d'ensemble sur la Belgique	73
6.24	Vue d'ensemble sur la Belgique après la fermeture	73
6.25	Zoom sur la situation locale	74
6.26	Zoom sur la situation locale après la fermeture	74
6.27	Vue d'ensemble sur la Belgique	75
6.28	Vue d'ensemble sur la Belgique après la fermeture	75
6.29	Zoom sur la situation locale	75
6.30	Zoom sur la situation locale après la fermeture	75
6.31	Vue d'ensemble sur la Belgique	76
6.32	Vue d'ensemble sur la Belgique après la fermeture	76
6.33	Zoom sur la situation locale	77
6.34	Zoom sur la situation locale après la fermeture	77
6.35	Vue d'ensemble sur la Belgique	78
6.36	Vue d'ensemble sur la Belgique après la fermeture	78
6.37	Zoom sur la situation locale	78
6.38	Zoom sur la situation locale après la fermeture	78
6.39	Vue d'ensemble sur la Belgique	79
6.40	Vue d'ensemble sur la Belgique après la fermeture	79
6.41	Zoom sur la situation locale	79
6.42	Zoom sur la situation locale après la fermeture	79
6.43	Temps supplémentaires en fonction du nombre de véhicules im- pactés pour chaque tronçon	81
6.44	Km supplémentaires en fonction du nombre de véhicules impac- tés pour chaque tronçon	81
A.1	Exemple d'histogramme	88
A.2	Fonction de distribution uniforme	91
A.3	Fonction de répartition uniforme	91
A.4	Fonction de distribution binomiale	92
A.5	Fonction de répartition binomiale	92
A.6	Fonction de distribution géométrique	93
A.7	Fonction de répartition géométrique	93
A.8	Fonction de distribution uniforme	94
A.9	Fonction de répartition uniforme	94
A.10	Fonction de distribution normale	94
A.11	Fonction de répartition normale	94
A.12	Fonction de distribution bimodale	95
A.13	Fonction de répartition bimodale	95
B.1	Un multi graphe et un graphe simple	97
B.2	Exemple de graphe particulier	98

B.3	Grappe	99
-----	------------------	----

Liste des tableaux

4.1	Distance domicile-travail moyenne en fonction du lieu de domicile et du lieu de travail	33
4.2	Kilométrage annuel des voitures belges	33
4.3	Distance moyenne d'un déplacement domicile-travail	34
5.1	Liste des villes hôtes	37
5.2	Nombre de voyageurs-kilomètres parcourus par an	41
5.3	Nombre de passages de véhicules sur les routes	55
5.4	Pourcentage de camions et d'autobus sur les routes belges	55
6.1	Fermeture du tronçon entre La Louvière et Mons (E42)	63
6.2	Fermeture du tronçon entre La Louvière et Chapelle-lez-Herlaimont (E42)	67
6.3	Fermeture du tronçon entre Chapelle-lez-Herlaimont et Gosselies (E42)	70
6.4	Fermeture du tronçon entre Gosselies et Fleurus (E42)	71
6.5	Fermeture du tronçon entre Fleurus et Sombreffe (E42)	73
6.6	Fermeture du tronçon entre Nivelles et Gosselies (E420)	74
6.7	Fermeture du tronçon entre Perwez et Louvain-la-Neuve (E411)	76
6.8	Fermeture du tronçon entre Wellin et Ciney (E411)	77
6.9	Fermeture du tronçon entre Gand et Lokeren (E17)	79
6.10	Tableau récapitulatif des résultats	80
C.1	Liste des villes du fichier xyv	104

Tout d'abord, je remercie sincèrement Monsieur Luca Sgambi, mon promoteur, pour son encadrement, ses encouragements et sa disponibilité durant toute cette année.

Je remercie également ma famille et mes amis, toujours prêts à écouter, à donner des conseils et aider, et qui se sont intéressés de près ou de loin à ce mémoire.

Introduction

*Ce n'est pas la destination
mais la route qui compte.*

Proverbe Chinois

Notre société est articulée autour des réseaux de transport. Ces réseaux permettent le déplacement de personnes et de marchandises, par véhicule routier, par train, par avion, etc. Ces réseaux sont complémentaires et indispensable au développement sociaux-économique d'un pays. Ces réseaux génèrent énormément d'emplois, et certains experts estiment que la part de transport dans le PIB varie selon les pays entre 10 à 20 % [1]. De plus, ces réseaux de transport sont la condition même de la liberté des gens de se déplacer et de déplacer leurs marchandises.

Ce mémoire fait suite à celui de Joachim L'HOIR et Adrien TELLIER ayant pour intitulé : "Analyse de la résilience du réseau routier belge"[2]. Ils ont étudié la résilience du réseau routier belge, en vue d'améliorer la gestion des Ouvrages d'art. Les OA sont le point faible des routes, en raison de leur fragilité, de leur coût et du potentiel manque d'alternative. De plus, ils ont tendance à se détériorer plus vite [3]. Dans leur travail, Joachim et Adrien ont tout d'abord étudié un outil d'aide à la gestion des Ouvrages d'Arts, le BMS, puis ils ont abordé la résilience en détails. Ils ont ensuite construit une simulation Matlab qui permet d'analyser les flux de trafic en Belgique suite à la fermeture d'une route.

Ce travail a deux intérêts, il permet d'aider la planification des entretiens routiers en voyant comment le réseau se comporte suite à la fermeture du tronçon. De plus, ce travail permet également de déterminer les Ouvrages d'art à risque, en fonction du trafic journalier et de leur état actuel.

Mon mémoire s'inscrit dans la suite de ce travail, et tout particulièrement avec l'amélioration de la simulation Matlab. Le travail précédent donnait une image du nombre de véhicules passés sur le réseau à la fin de la journée. L'amélioration principal est de rendre la simulation temporelle, c'est-à-dire, à tout instant, de donner aux véhicules une position sur le réseau. D'autre améliorations ont été effectuées, comme la prise en compte des embouteillages.

Ce mémoire ainsi que celui de Joachim et Adrien s'inscrivent dans le cadre d'un grand projet de recherche sur la mobilité. Des recherches futures concernant la mobilité en Belgique ou l'amélioration de ce modèle sont prévues.

Le premier chapitre de ce mémoire abordera l'intérêt économique que procure le réseau routier. Pourquoi les réseaux de transport requièrent-ils des investissements et des études scientifiques ?

Comment le réseau s'adapte-il suite à la fermeture d'une route? Quelles seront les conséquences? Ces questions peuvent être analysées au travers du concept de la résilience. La résilience propose des pistes pour étudier le comportement du trafic sur le réseau suite à une perturbation. Le deuxième chapitre fera une revue de la littérature sur la résilience, et analysera le concept de résilience dans le cadre des transports.

Le troisième chapitre fait état du modèle et de ses hypothèses tel que proposé dans le travail de Joachim et Adrien. Puis seront abordées les améliorations que propose ce travail.

Un modèle représentatif de la réalité nécessite des données actuelles et fiables. Le chapitre 4 propose et critique les données disponibles sur le site internet du SPF mobilité et transport.

Avec ces données, l'écriture de la modélisation peut commencer. Le chapitre 5 détaille tous les paramètres du modèle ainsi que leur justification.

A l'aide du concept de résilience des transports et du modèle, une analyse des résultats de la simulation sera présentée au chapitre 6. Ces analyses portent sur le comportement du trafic suite à la fermeture d'une route ou la réduction du nombre de bandes.

Enfin, une critique du modèle ainsi que ses améliorations possibles pour un travail futur seront détaillées.

Les codes Matlab ainsi qu'un tutoriel d'utilisation est disponible dans les Annexes.

Chapitre 1

L'importance d'un réseau routier en bon état

1.1 Introduction

Ce chapitre traite de l'importance d'un bon réseau de transport, et plus particulièrement du réseau routier. Le réseau routier correspond à 90% du transport de marchandises dans le monde et 83% du transport de personnes [4].

Le développement des routes est indispensable au développement d'une région. Les Romains l'ont compris dès l'Antiquité, avec leurs fameuses voies romaines. Cela leur permettait un trajet plus aisé et facilitait les guerres de conquête, notamment pour le transport de troupes, mais aussi le ravitaillement et le courrier. D'ailleurs, aujourd'hui encore le dicton suivant est utilisé : "toutes les routes mènent à Rome". Preuve que leur réseau était efficace.

Les Chinois ont également développé leur économie en construisant la route de la soie. Cette route permettait le commerce de la soie et d'autres marchandises entre l'Asie et l'Europe.

Aux temps modernes, à l'époque des colonies, une des priorités des colons était de construire des routes. Cela permettait de rapatrier plus facilement les ressources en métropole et d'améliorer les communications pour mieux administrer les colonies.

La première partie de ce chapitre développera les avantages économiques que procure la route. Ensuite, sera abordée l'importance de l'entretien des routes et enfin de l'apport social, et des défis énergétiques et environnementaux.

1.2 Rôles du réseau routier dans l'économie

Les réseaux de transport jouent un rôle fondamental dans l'économie actuelle. En plus de générer énormément d'emplois pour sa construction et sa maintenance, le réseau routier permet notamment l'acheminement des matières premières vers les industries, qui vont transformer ces ressources en produits finis. Puis de transporter ces produits vers les lieux de vente. Réduire les coûts liés au transport est possible en investissant dans les transports. Cela permet de réduire le coût final du produit et donc augmente la productivité. Au plus les coûts de transport à l'intérieur d'un pays seront bas, au plus ce pays sera compétitif sur le plan international, et tourné vers l'exportation [5].

L'économie moderne est basée sur la grande capacité d'échanges, ce qui suppose des transport importants et peu chers. Le développement économique est corrélé avec la rapidité et la densité de circulation des biens. Le plus grand obstacle à ces échanges est la distance. Le développement des réseaux de transport est donc important. Avoir une route en bon état augmente donc la capacité d'échange. Le coût lié au transport est donc une donnée très importante, car le transport n'ajoute pas en soi de valeur au produit fini. [6].

Un réseau rapide permet de réduire les coûts liés au temps de transport. Les prix vont donc diminuer et le pouvoir d'achat va augmenter. Pour l'Etat, il est donc très intéressant de développer son réseau routier, car cela permet de booster son économie. Le développement des transports peut être comparé à l'élargissement du marché. Les marchandises peuvent aller plus loin pour moins cher.

Le transport fluvial est moins cher mais est plus restrictif. En effet, un bateau ne peut pas aller partout. Il suit les cours d'eau naturels ou les canaux. Mais parfois le terrain rend la construction d'un canal impossible. L'avantage des routes, c'est qu'elles peuvent relier même des zones peu accessibles.

Pour un pays développé qui dispose d'un réseau routier important et bien entretenu, le coût de transport est trois fois moins élevé que pour un pays en voie de développement [7]. En effet, les réseaux routiers dans les pays en voie de développement ne sont pas très performants, ce qui entraîne de lourds investissements pour acheminer les ressources. De nombreuses études ont établi que l'amélioration du transport dans ces pays là avait d'énormes conséquences, au niveau des échanges tant intérieurs qu'extérieurs, notamment en comparant les régions reliées au réseau et celles qui ne sont pas, à l'intérieur d'un même pays. Malheureusement, certains pays n'ont pas les moyens de construire et d'entretenir un bon réseau routier [8].

1.3 Entretien

Cependant, construire une infrastructure nécessite un lourd investissement au départ, et un entretien tout au long de sa durée de vie. L'entretien coûte entre 2 et 5% de l'investissement initial par an. De plus, l'amortissement se fait sur le très long terme. De plus, le vieillissement des infrastructures routières augmente les coûts d'entretien et diminue la capacité de la voirie. Une chaussée en mauvais état peut engendrer des coûts supplémentaires si, par exemple, cela augmente le temps de parcours, abîme les marchandises ou provoque des accidents supplémentaires.

A court terme, il peut sembler intéressant de ne pas entretenir la route, mais c'est un mauvais calcul. Au Maroc par exemple, ils ont déterminé qu'en augmentant le budget lié à l'entretien des routes de 2,6 millions d'euros par an, cela faisait économiser 7,5 millions d'euros en 20 ans par rapport au budget initial. Cependant, ce n'est pas toujours possible [1].

Sur la figure 1.1 (source : [1]) est représentée la variation dans le temps des coûts liés à l'entretien. Si la route est trop entretenue, par exemple en la renforçant en avance, cela coûte plus cher. Mais attendre trop longtemps empire la chose, les coûts liés à la remise en état ou à la reconstruction seront beaucoup plus importants.

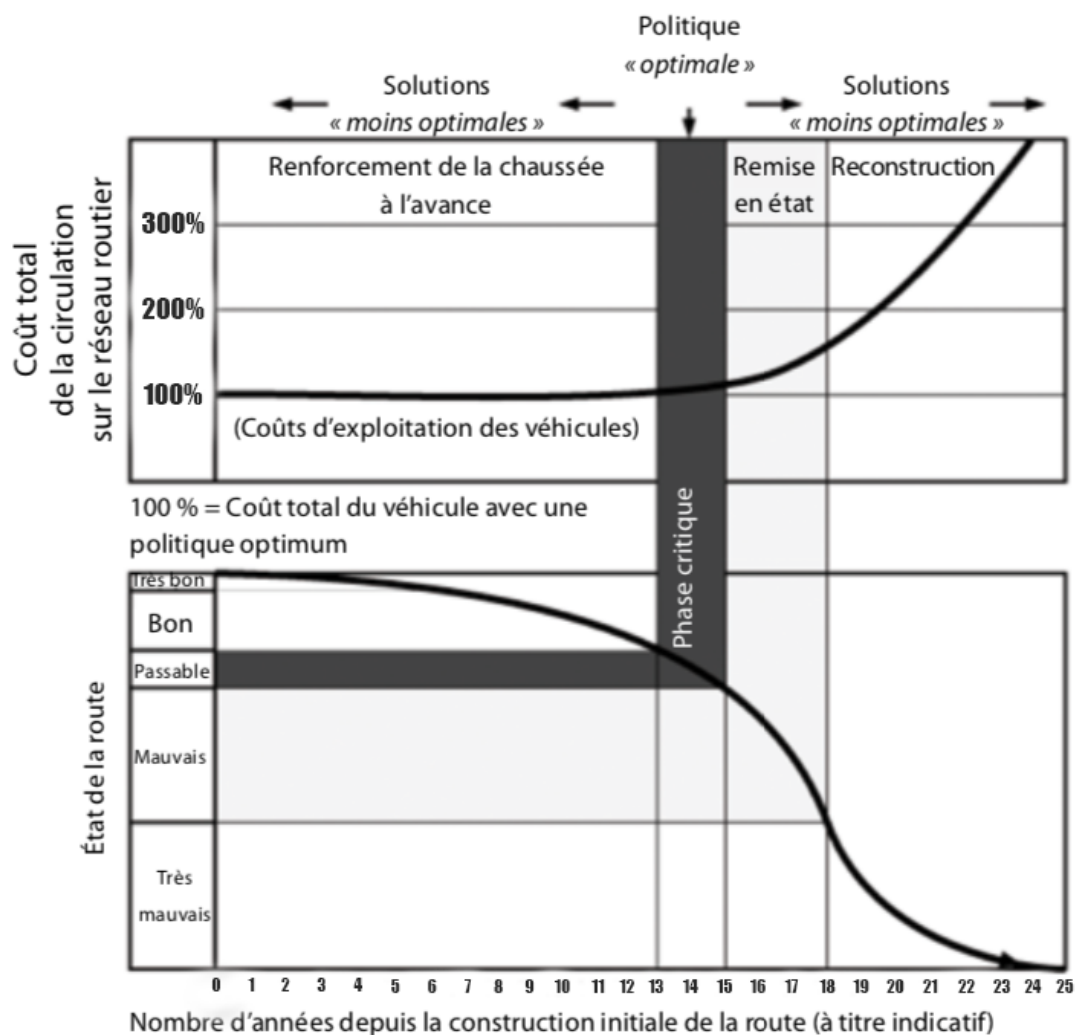


FIGURE 1.1 – Variation des coûts dans le temps

Un autre problème peut survenir si le trafic à long terme est surestimé. Ces infrastructures ne vont pas être utilisées à 100% de leur capacité. Cela revient à faire un gros investissement peu rentable au final. Un exemple assez controversé en Belgique est l'ascenseur à bateaux de Strepv-Thieu. Il a coûté environ 1 milliard d'euros et nécessite des entretiens coûteux. Plusieurs chercheurs s'accordent à dire qu'il ne sera jamais rentable. D'autres en revanche soulignent que cela a énormément développé le transport fluvial.

Historiquement, il n'y avait aucune ou très peu de politique d'entretien. Cela entraîne des économies à court terme seulement. De nos jours, il y a des politiques d'entretien, mais cela reste compliqué à mettre en place car les investissements rentables à court terme sont mieux acceptés par la population. Des outils existent pour aider à la gestion de ces infrastructures, comme le BMS. Le premier chapitre du mémoire de Joachim L'Hoir et Adrien Tellier traite de ce sujet, mais cela ne sera pas abordé ici.

Les ponts et les routes ont une durée de vie entre 50 et 100 ans (à condition

d'avoir été bien entretenus). En Belgique, la plupart des ponts et une partie des routes ont été construits durant la seconde moitié du XXe siècle. Il faudra donc prévoir de les remplacer bientôt.

Étonnamment, les pays pauvres dépensent en moyenne moitié plus que les pays riches par kilomètre de route. C'est que les routes sont déjà construites dans les pays riches et que l'entretien est plus performant. Cela démontre clairement qu'un entretien au bon moment réduit les dépenses futures.

1.4 Impact social

Avoir une route en bon état est aussi nécessaire pour le bien-être des habitants. Cela augmente le pouvoir d'achat comme dit précédemment, mais il y a également d'autres avantages. Les réseaux de transport sont indispensables à la liberté de se déplacer et de déplacer leurs marchandises. Cela permet aussi de désenclaver les zones rurales, une meilleure mobilité, que ce soit pour aller travailler ou pour faire les courses. Une meilleure mobilité permet aussi un accès plus facile à différents services comme l'école ou les soins de santé. Enfin, si la route est en bon état, cela peut également diminuer le bruit pour le confort des habitants. Au niveau sécuritaire, la police ou les pompiers peuvent intervenir beaucoup plus rapidement en cas de besoin [9].

Des dégâts sur la route, comme les "nids de poule", réduisent la vitesse des véhicules, allant même jusqu'à créer des bouchons. Cela augmente les temps de trajet. De plus cela entraîne une augmentation de la consommation de carburant, et donc un coût pour les ménages.

En conclusion, l'intérêt de mener des politiques d'entretien du réseau routier est important du point de vue de la sécurité routière, des économies futures, de l'environnement, et enfin du bien-être des habitants. Néanmoins, il faut investir correctement.

1.5 Défis énergétique et environnemental

Néanmoins, un bon réseau routier ne présente pas que des avantages. En effet, cela incite les gens à utiliser leur voiture, qui n'est pas le moyen de transport le plus propre. De nos jours, il est aujourd'hui primordial de penser à l'impact sur notre planète lors d'études. De plus, les réserves mondiales de pétrole et des matières premières en général diminuent, et la construction d'une route en nécessite beaucoup.

De plus, l'économie locale est de plus en plus valorisée. Cette économie encourage les initiatives locales et la mise en place de petites infrastructures. Le but recherché est de dynamiser l'économie locale, qui peut souffrir de la

concurrence de grandes multinationales qui acheminent des produits sur des milliers de kilomètres.

Chapitre 2

Etat de l'art sur la résilience

Ce chapitre fait le bilan de l'état de la littérature sur la résilience. La première partie de ce chapitre fait état des définitions de la résilience. La seconde partie décline la résilience dans plusieurs domaines, notamment l'écologie ou l'économie. Enfin, la résilience dans le domaine des transports sera abordée pour définir la construction d'un réseau de transport.

2.1 Définition du concept de résilience

Le mot résilience provient du mot latin *resiliere*, qui signifie rebondir. La définition la plus large possible pour la résilience est la capacité d'un système à retrouver un équilibre stable après un événement perturbateur. D'autres auteurs proposent une définition différente. Allenby et Fink définissent la résilience comme *la capacité d'un système de conserver ses fonctions et sa structure face aux changements internes et externes*. Pregenzer la définit comme *la mesure de la capacité d'un système à absorber les changements continus et imprévisibles tout en maintenant ses fonctions vitales*. Haimès la définit comme *capacité du système à résister à une perturbation majeure dans des paramètres de dégradation acceptables et à se rétablir avec une temps approprié, coûts et risques raisonnables*. Depuis de nombreuses années, plusieurs domaines s'approprient une définition de la résilience adaptée.

La résilience s'est développée comme un outil pour la gestion des risques. Le concept devient à la mode dans la communauté scientifique, de plus en plus de travaux traitant de ce sujet sont publiés et couvrent beaucoup de domaines différents. Certains chercheurs travaillent sur la résilience mais sans la nommer, en utilisant d'autres termes, comme la vulnérabilité ou la reconstruction. La figure 2.1 montre la pluridisciplinarité de la résilience ainsi que les concepts associés à ce mot. [10]

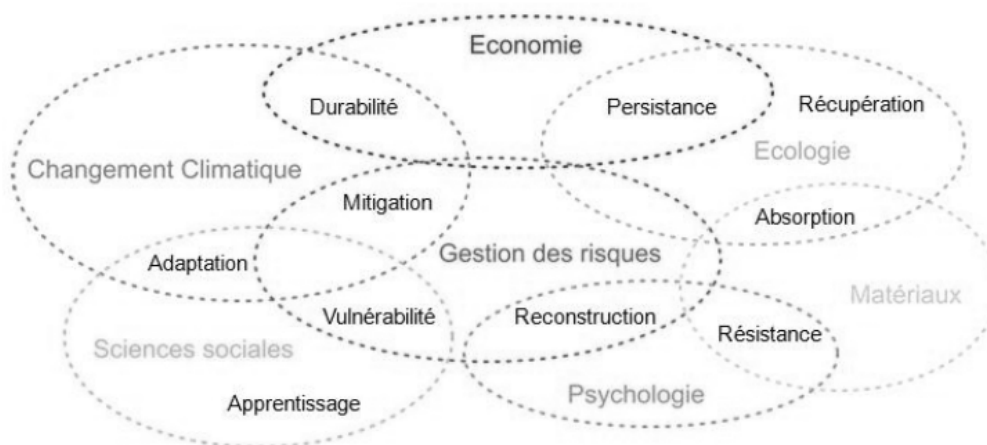


FIGURE 2.1 – Pluridisciplinarité de la résilience et des concepts associés

Notre société repose sur un grand nombre de systèmes, comme le réseau électrique, la production d'électricité, le réseau routier, la gestion des eaux, la production en eau, sa potabilisation, internet, ... Grâce aux nouvelles technologies, ces réseaux sont plus complexes, plus efficaces, plus interconnectés. L'utilisation de cette interdépendance crée des tensions et des risques de défaillance. Cette multiplication de connections rend le système global de moins en moins prévisible. Si un problème arrive sur le réseau, il peut y avoir de multiples répercussions. Par exemple, un souci dans une usine de béton peut retarder une livraison et impacter le bon déroulement du chantier. Autre exemple, en cas de coupure d'électricité, les trams, métros et trains ne peuvent pas circuler. [11]

Pour être efficaces tout en minimisant les coûts, les réseaux fonctionnent à un niveau proche de leur performance maximale. Cela rend ces systèmes vulnérables et sensibles à des perturbations.

L'intérêt qu'a la communauté scientifique à développer et comprendre les propriétés de la résilience, est d'élaborer des stratégies de gestion des risques et de les intégrer dans la conception des systèmes. Il existe différentes stratégies de gestion des risques, soit se concentrer sur la réduction des causes de perturbations, soit se concentrer sur les effets, et essayer de les minimiser. Le risque zéro n'existe pas ; si la menace devient réalité, des dysfonctionnements apparaissent. Cela n'implique pas forcément une catastrophe, mais peut provoquer des défaillances mineures.

Historiquement, lors d'une étude de risque, les questions suivantes se posaient : [12]

- *Qu'est-ce qui peut mal tourner ?*
- *Quelle est la probabilité d'un tel scénario de perturbation ?*
- *Quelles sont les conséquences d'un tel scénario ?*

Les stratégies déployées se basaient sur la réduction de la probabilité de défaillance et sur la réduction des conséquences. Un système était conçu pour éviter et absorber les éléments perturbateurs. De nos jours, certains événements sont considérés comme inévitables, par exemple un tsunami ou un tremblement de terre. La stratégie porte maintenant sur la capacité à se rétablir et au temps de rétablissement.

Un système est dit résilient s'il sait où trouver les capacités à s'adapter aux perturbations et retrouver un état de fonctionnement stable.

Il n'y a un risque qui si l'enjeu est vulnérable. Si la menace devient réelle, cela entraîne des perturbations. Si ces perturbations dépassent un seuil, il y a une crise. En général, la crise entraîne un processus de reconstruction vers un nouvel état stable plus ou moins proche de l'état normal. Ce processus permet d'étudier l'efficacité de la résilience. Dans ce cas-ci, la résilience est considérée comme un processus. Cependant, certains auteurs associent la résilience à une

qualité du système présente avant la menace, et qui apparaît lors de la reconstruction.

Certains systèmes sont plus impactés que d'autres par une perturbation. Par exemple une coupure de courant dans un hôpital causerait des dégâts intolérables. Même si cela ne devrait pas se produire, il est nécessaire de s'en protéger. Les conséquences seraient trop importants, et personne ne peut se le permettre. Les hôpitaux ont donc recours à des groupes électrogènes, qui assurent la fourniture d'électricité à l'instant où la coupure se produit. C'est même une obligation légale en Belgique.

2.2 La résilience selon quatre concepts

David Woods a mis en évidence quatre concepts pour définir la résilience [13], qui sont présents dans plusieurs articles scientifiques. Cette section les passe en revue.

2.2.1 La résilience comme rebond après un traumatisme

La réflexion a pour base cette question : *Pourquoi est-ce que certaines communautés, groupes ou individus récupèrent d'un traumatisme plus rapidement que d'autres ?* A partir de cela, la résilience est définie comme la capacité à rebondir. L'accent n'est pas mis sur le temps nécessaire à la reconstruction mais bien sur la capacité et les ressources présentes avant le traumatisme. Cela part du principe que la capacité à rebondir dépend en grande partie de la structure mise en place en amont. De plus, la capacité à rebondir prend aussi en compte les réactions pendant et après la perturbation.

Un événement est inattendu lorsqu'il sort du spectre des perturbations que le système a la capacité de gérer. La capacité à rebondir n'est donc pas dans les caractéristiques de l'évènement mais dans la manière dont le système va apprendre à gérer la situation inattendue. Cela correspond à la capacité d'adaptation. Plus un système a une forte capacité d'adaptation, moins il est vulnérable.

Le processus d'adaptation aux perturbations, aux défis et aux surprises modifie le système au fil du temps. En étant confrontés à des épreuves, les systèmes retiennent les actions qu'elles ont effectuées et modifient leurs profils pour mieux gérer une autre situation. Même en s'adaptant pour préserver, le processus d'adaptation transforme à la fois le système et son environnement.

2.2.2 La résilience synonyme de robustesse

La résilience peut aussi être définie comme la capacité à encaisser une perturbation. Dans ce cas, la résilience est apparentée à la robustesse. Une augmentation de la robustesse augmente le spectre des perturbations que le système peut encaisser efficacement.

Cette définition de la résilience entraîne de nouvelles interrogations. Que se passe-t-il si la perturbation sort du spectre d'encaissement ? Le système ne peut plus répondre aux demandes. Quelles défaillances cela va entraîner ?

2.2.3 La résilience en opposition avec la fragilité

Le troisième concept considère la résilience comme le contraire de la fragilité. Plutôt que de poser la question de savoir comment et pourquoi les systèmes rebondissent, la question essaie de déterminer *comment les systèmes s'étendent pour gérer les surprises* ? Les systèmes dotés de ressources limitées connaissent toujours des difficultés et s'adaptent aux événements imprévus.

Dans ce cas, la résilience est la capacité d'extensibilité de la zone de performance. C'est donc la capacité des systèmes à s'adapter à des situations imprévues. Cela revient à élargir les performances pour éviter les défaillances. Ce processus de défense est dynamique et modifie le système en lui-même.

2.2.4 La résilience comme adaptabilité durable

C'est la capacité à s'adapter de manière durable et à plus grande échelle. Certaines perturbations peuvent entraîner des défaillances dans le futur car le système n'a pas su retrouver ou garder un état stable après la perturbation. L'idée est que la défaillance peut intervenir longtemps après la perturbation si celle-ci a fragilisé le système. La résilience comme l'adaptabilité durable essaie de répondre à ces trois questions :

- *Quelles caractéristiques dans la structure des systèmes expliquent la différence entre les systèmes qui produisent une capacité d'adaptation durable et ceux qui ne parviennent pas à maintenir la capacité d'adaptation ?*
- *Quels principes et techniques permettraient de concevoir un réseau capable de produire une adaptabilité durable ?*
- *Comment peut-on évaluer efficacement si un système a la capacité de maintenir une adaptabilité dans le temps ?*

Certains systèmes modifient leurs comportements au cours de leur existence. L'adaptabilité durable prend en compte la dynamique d'un système au

cours de sa durée de vie.

Ces quatre types de résilience ne sont pas opposés mais peuvent se compléter. Certains auteurs ont une préférence pour l'un ou l'autre.

2.3 Résilience écologique

L'écologie a été le premier domaine à s'intéresser à la stabilité des systèmes après une perturbation. Le pionnier a été MacArthur en 1955. Ensuite, Pimm et Holling ont également effectué des recherches. La résilience écologique est définie comme la capacité d'un écosystème ou d'une population à retrouver un fonctionnement normal après une perturbation conséquente.

2.3.1 Etude de Robert MacArthur

En 1955, Robert MacArthur étudie la stabilité d'une communauté d'animaux [14]. Pour ce faire, il a représenté un système avec quatre espaces représenté sur la figure 2.2. Une flèche de a vers b signifie que b est un prédateur de a . a est vu comme une source d'énergie pour b . L'énergie est transférée via la nourriture. Dans cette chaîne alimentaire, S_1 mange S_2 et S_3 , et S_2 mange S_3 et S_4 . S_3 et S_4 ont une source d'énergie non représentée ici.

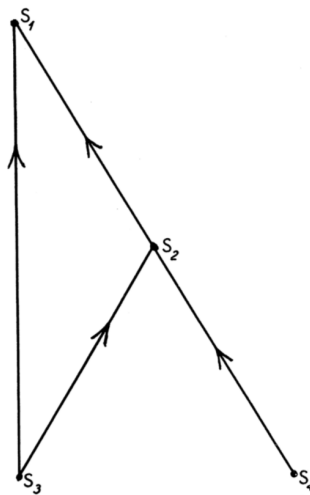


FIGURE 2.2 – Représentation de la communauté étudiée

L'auteur pose trois hypothèses qui doivent être respectées pour que la taille de la population de chaque espèce de varie pas dans le temps :

- La quantité d'énergie entrante est constante dans le temps (énergie prise par S_3 et S_4).

- Le temps où l'énergie est retenue chez une espèce ne varie pas dans le temps.
- La taille d'une population varie uniquement en fonction de la quantité d'énergie disponible.

Tout d'abord, il démontre que l'énergie qui sort du système est égale à l'énergie qui rentre car le temps de transfert de l'énergie est constante entre les espèces, et toute l'énergie provient de l'approvisionnement de S3 et S4. A partir de cela, l'énergie qui rentre et sort de chaque espèce est constante, et donc la population de chaque espèce ne change pas.

Dans certains écosystèmes, la population et la quantité d'espèces différentes tend à rester constante, tandis que dans d'autres, cela varie fortement. Si la population d'une espèce dépend fortement de la population d'une autre espèce, MacArthur définit ce système comme instable. Par contre, si une espèce a plusieurs proies, et donc dépend de plusieurs espèces, c'est une communauté stable. Le stabilité d'un système est proportionnelle au nombre d'interactions entre les espèces du système. En comparant la chaîne alimentaire à un réseau de transport, plus il y a de parcours possibles entre deux points, plus le système est stable.

MacArthur définit la stabilité du système comme suit :

$$S = - \sum_i \sum_j p_{ij} \log(p_{ij}) \quad (2.1)$$

Où p_{ij} est la proportion d'énergie de S_i qui provient de S_j .

Par exemple, dans la communauté représentée à la figure 2.3, la stabilité vaut : $S = -4 \times \frac{1}{4} \log\left(\frac{1}{4}\right) = 0,6$

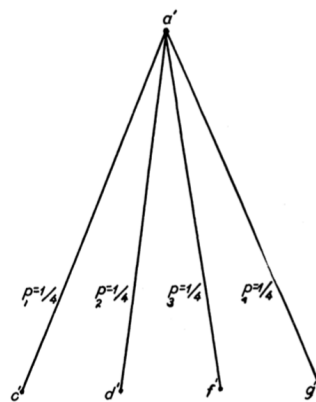


FIGURE 2.3 – Représentation d'une communauté

2.3.2 Etude de Stuart L. Pimm

Dans les années 80, les écologistes utilisaient beaucoup les termes comme "stabilité", "fragilité", "vulnérabilité", pour décrire les phénomènes naturels. Après avoir étudié les différentes significations de ces termes, Stuart L. Pimm a décrit quatre genres de stabilité écologique : stabilité, résilience, variabilité, et persistance. Pour chacun de ces terme, Pimm propose une définition et des pistes de mesures. [15]

- Un système est stable si les variables reviennent à leur valeur initiale après une perturbation.
- La résilience est la mesure du temps nécessaire à un système pour retrouver un état d'équilibre après une perturbation.
- La variabilité mesure la différence de densité de population des espèces au cours du temps.
- La persistance est la capacité d'une espèce à garder constante sa densité de population.

Pimm défini donc la résilience par rapport au temps de rétablissement.

2.3.3 Etude de C.S Holling

C.S Holling a de son côté étudié les différents comportements possibles entre deux populations qui interagissent. [16]

Pour illustrer ses propos, il a pris comme exemple les effets d'une invasion de *tordeuses des bourgeons de l'épinette* dans les forêts du Canada. Les tordeuses sont de petites chenilles brunes, elles se nourrissent des épinettes et des bourgeons du sapin. Les tordeuses tuent les sapins et cela crée une épidémie. Il y a une épidémie tous les 35 ans environ.

Lors d'une épidémie, les sapins sont particulièrement touchés. Il ne reste que les sapins les plus résistants et les bouleaux, qui eux ne sont que peu touchés. La population jeune est beaucoup moins touchée que la population âgée. A la fin d'une épidémie, les sapins et les bouleaux se développent ensemble, ce qui entraîne une surpopulation des deux espèces. Le sapin pousse plus vite, il est donc favorisé dans la compétition avec le bouleau. Pendant ces années, la population de tordeuses est assez basse. S'il y a plusieurs années de saisons sèches consécutives, les prédateurs des tordeuses sont moins présents, ce qui entraîne une augmentation de leur population. Si la population des sapins est importante en même temps, l'épidémie commence. Cela va engendrer la chute de la population de sapins, ce qui va aussi faire chuter la population des tordeuses à la fin de l'épidémie.

Ce procédé est défini comme un équilibre cyclique stable avec une large amplitude par Holling. En suivant la logique de MacArthur, ce système serait très instable, car la population varie considérablement. Holling définit donc deux concepts pour caractériser un comportement, la stabilité et la résilience.

Pour l'auteur, la stabilité représente la capacité de retrouver son état d'équilibre après une perturbation. Le temps de retour ainsi que les fluctuations sont importantes. La résilience quand à elle représente la capacité des systèmes à encaisser les perturbations. Dans le cas des tordeuses, le système est instable mais résilient.

2.4 Résilience économique

La résilience économique a été fort développée par Adam Rose [17][18][19]. Il distingue la résilience statique de la résilience dynamique.

La résilience économique statique est la capacité d'un système à maintenir un haut niveau de fonctionnement en cas de perturbations. Le système doit utiliser les ressources restantes aussi efficacement que possible durant la phase de récupération.

La résilience économique dynamique est la capacité à accélérer la récupération du système. Plutôt que de regarder le stock restant, le système doit renforcer ses capacités et prendre des décisions importantes rapidement.

Il faut aussi distinguer la résilience inhérente et la résilience adaptative.

La résilience inhérente se réfère aux caractéristiques déjà présentes dans le système tels que les stocks disponibles, la capacité d'excédent, la capacité d'accéder aux fournisseurs situés hors de la zone d'importation.

La résilience adaptative se réfère aux décisions prises sous pression et la capacité à diminuer sa situation. Par exemple, travailler sans chauffage.

L'économie peut s'étudier selon trois niveaux :

- La microéconomie, qui étudie les comportements au niveau individuel, comme une entreprise ou un ménage.
- La mésoéconomie, qui concerne une industrie ou un marché.
- La macroéconomie, qui prend en compte les grands agrégats économiques.

Rose a défini une mesure de la résilience au niveau micro et mésoéconomique, la DSER *direct static economic resilience*. Cela correspond à la résilience au niveau de l'entreprise individuelle ou du secteur. Il a également défini une

mesure au niveau macro, la TSER, *total static economic resilience*, qui considère l'économie dans son ensemble.

La DSER est la proportion entre la réduction directe de la production lors d'une perturbation par rapport à la réduction potentielle maximale de la production. Par exemple, à la suite de l'attaque terroriste du 11 septembre 2001 sur le World Trade Center, plus de 95% des entreprises concernées ont évité la faillite grâce à la délocalisation. La perte économique potentielle si les entreprises avait fait faillite aurait été de 43 milliards de dollars. La relocalisation a entraîné une perte de 11 milliards de dollars. Cela revient à estimer que la tactique de résilience avec la relocalisation est $72\% \left(\frac{43-11}{43} \right)$.

Woods utilise le terme robustesse pour faire référence à des systèmes conçus pour gérer efficacement les défaillances connues. La résilience décrit dans quelle mesure le système peut gérer les problèmes imprévisibles. En résumé, la robustesse pour gérer les inconnus et la résilience pour gérer les inconnus inconnus.

2.5 Résilience en ingénierie au service des transports

La résilience dans le domaine de l'ingénierie a été développée pour compléter les approches en gestion des risques, jugé incomplète. En effet, la communauté scientifique trouvait que les travaux précédents en matière de résilience n'étaient plus adaptés. Les ingénieurs voulaient trouver un équilibre entre productivité et sécurité. Le but principal était d'aider les gestionnaires à décider à quel moment ils devaient réduire la production pour réduire les risques de catastrophe [20].

La résilience en ingénierie tient compte de l'effet dynamique des systèmes. Elle part du principe que les systèmes changent constamment et que ce phénomène est encore plus important lors de perturbations [21].

L'ingénierie de la résilience permet de développer les concepts et les méthodes pour mesurer et évaluer la capacité qu'ont les systèmes à ajuster leurs fonctionnements pendant et après des perturbations, de telle sorte qu'ils puissent poursuivre leur fonctionnement. Cette définition met l'efficacité et la prédiction au centre de la conception [22].

Pimm a défini la résilience par rapport au temps de recouvrement tandis que Holling propose la résilience comme la mesure de résistance aux perturbations. Ces deux approches ont influencé Adam Rose pour ses travaux sur la résilience économique, statique et dynamique. Ces travaux ont servi de base à David Woods qui a beaucoup apporté dans la recherche sur la résilience. La résilience en ingénierie est la plus adaptée pour étudier la résilience des transport.

Le réseau de transport supporte beaucoup de pression, il doit être efficace, opérationnel, fiable et sécurisé, comme discuté dans le chapitre 1. Dès lors, il est important d'avoir un réseau de transport résilient. Cela nécessite deux choses :

- Le réseau doit permettre le passage des usagers le plus rapidement possible en cas d'accidents ou de travaux.
- Le réseau doit également prendre soin des ouvrages d'art, qui sont le point faible d'une route, car ils nécessitent plus d'entretien, coûtent plus cher et offrent moins d'alternative en cas de fermeture. L'état d'un ouvrage d'art dépend de beaucoup de paramètres, les flux de trafic, les conditions météorologiques, ... Beaucoup de ses paramètres sont aléatoire, mais connaître certaines propriétés de ces paramètres permettra d'aider à la gestion des ouvrages d'art. [23]

La résilience des transports est la capacité du réseau à déplacer les usagers en cas de perturbations. La modélisation Matlab du réseau, présentée dans les chapitres suivant, permet d'analyser l'impact d'une fermeture ou d'une réduction de flux sur une route.

2.6 La vulnérabilité et la gestion des risques

Un mot qui revient souvent dans la littérature de la résilience est le mot vulnérabilité. Les concepts de vulnérabilité, risque et résilience sont assez proches. C'est pourquoi il est intéressant d'aborder ces concepts. La vulnérabilité d'une entité peut être défini comme le point faible de cette entité.

Dans la gestion des risques, un aléa est la probabilité qu'un certains évènement se produisent. La vulnérabilité représente les éléments ou les enjeux liés à ce même évènement. Enfin, le risque est la combinaison de l'aléa et de la vulnérabilité. La matrice de risque sur la figure 2.4 permet d'évaluer les risques lié à un évènement. Seuls les risques jugés modérés et inacceptables font l'objet d'une analyse et d'une tentative de résolution. [24]

Un risque faible est la combinaison d'un événement rare et présentant peu de conséquences. Par exemple le fait de tomber dans la rue.

Un risque modéré est présent si l'évènement est rare mais est assez grave ou si l'évènement va sûrement se produire mais a peu d'impact. Cela est par exemple un tsunami dans un zone a faible risque, les conséquences sont grandes mais l'évènement a peu de chances de se produire.

Le risque inacceptable combine un grand risque avec une probabilité élevée. Ce genre de cas est à éviter, par exemple personne ne devrait venir s'installer sur un volcan en activité ou manier des éléments radioactifs sans protection.

		Gravité			
		Faible	Moyenne	Elevé	Très élevé
Probabilité	Très rare				
	Rare				
	Probable				
	Très probable				

	Risque faible
	Risque modéré
	Risque inacceptable

FIGURE 2.4 – Matrice de risque

Une étude de risque doit donc évaluer pour chaque scénario de perturbation, la probabilité que ce scénario se produise ainsi que ses conséquences. Ensuite, si c'est nécessaire, de réfléchir à une alternative pour diminuer le risque.

Chapitre 3

Etat et hypothèses du modèle précédent et pistes d'amélioration

La première partie de ce chapitre va présenter le travail de Joachim L'Hoir et Adrien Tellier et leurs hypothèses de travail. La deuxième partie va présenter les améliorations proposées dans ce travail.

3.1 Résumé de la simulation de Joachim L'Hoir et Adrien Tellier

Joachim L'Hoir et Adrien Tellier ont construit une simulation du réseau routier belge. Cette simulation détermine le nombre de voitures qui sont passées sur chaque route à la fin de la journée. L'intérêt du modèle est d'étudier la résilience du réseau en analysant le comportement des véhicules à la suite d'une fermeture d'une route.

3.1.1 Représentation du réseau

Ils ont tout d'abord construit le réseau routier. Ils ont sélectionné les autoroutes et les routes nationales les plus importantes. Ensuite ils ont mis en évidence les intersections de ces routes, qui correspondent aux noeuds. Ces noeuds sont la plupart du temps associées à une ville réelle. Il y a 186 noeuds (villes) et 303 tronçons de route. Les coordonnées géographiques de chaque ville sont enregistrées dans un fichier texte. Ce fichier est présenté dans l'annexe C. Ils ont ensuite créé deux autres fichiers textes, le premier reprenant la distance entre chaque noeud relié par une route, le second reprenant le temps de parcours entre ces noeuds. La distance sur la route entre deux noeuds est considéré comme étant 110% de la distance à vol d'oiseau. ces données sont représentées par une matrice, qui est construite selon le même principe qu'une matrice d'incidence. Par exemple, le fichier contenant les distances est une matrice carrée de 186 lignes et l'élément de la ligne i et colonne j correspond à la distance entre le noeud i et j .

Le réseau a été modélisé à l'aide de la théorie des graphes. Un rappel théorique concernant la théorie des graphes est présenté dans l'annexe B. Ce rappel reprend les notions de base nécessaire à la compréhension de ce travail. Les villes et les routes peuvent être assimilées respectivement à des sommets et des arrêtes dans la théorie des graphes. Le but recherché est de pouvoir utiliser les théorèmes et algorithmes développés pour les graphes dans le modèle.

Ils ont considéré les véhicules se déplaçant par paquets de 1000 véhicules. Dans chaque paquet, les véhicules ont le même trajet. L'intérêt est de diminuer la complexité et le temps de calcul du code Matlab. Cette hypothèse est fiable pour un premier modèle, car il y a beaucoup de véhicules se déplaçant sur le réseau, donc forcément plusieurs véhicules prennent la même route. Pour déterminer le trajet de tous les paquets de véhicules, ils ont utilisé la méthode de Monte-Carlo pour choisir une ville de départ et une ville d'arrivée. La méthode

de Monte-Carlo est une méthode qui propose un algorithme qui permet d'obtenir des résultats de manière aléatoire. Cette méthode tient son nom du célèbre casino du quartier de Monte-Carlo à Monaco. Ils ont sélectionné 20 villes, en fonction de leur importance et de leur position sur la carte pour être les villes de départ ou d'arrivée. Ensuite, la méthode va choisir la ville de départ et d'arrivée en fonction de la population. Plus une ville a d'habitants, plus elle a de chances d'être choisie.

Une fois les villes de départ et d'arrivée choisies, ils ont déterminé le trajet le plus rapide ou le plus court en distance avec l'algorithme de Dijkstra. Cet algorithme sera traité en détail dans le chapitre sur la construction du réseau. Le trajet donne les noeuds successifs que le paquet de véhicules traverse. Ensuite était calculé le nombre de voiture qui sont passées sur chaque route en une journée.

La figure 3.1 montre la carte de la Belgique, trouvée sur Google Maps. Sur la figure 3.2 se trouve l'image du réseau routier belge tels que Joachim et Adrien l'ont modélisé sur AutoCad. Le numéro associé à chaque noeud correspond au numéro du noeud dans la modélisation. Ces numéros sont référencés dans l'annexe C. En rouge sont représentées les autoroutes, en vert sont représentées les routes nationales et en bleu correspondent les routes transfrontalières. Ces dernières ne font pas partie de la simulation.



FIGURE 3.1 – Carte de la Belgique

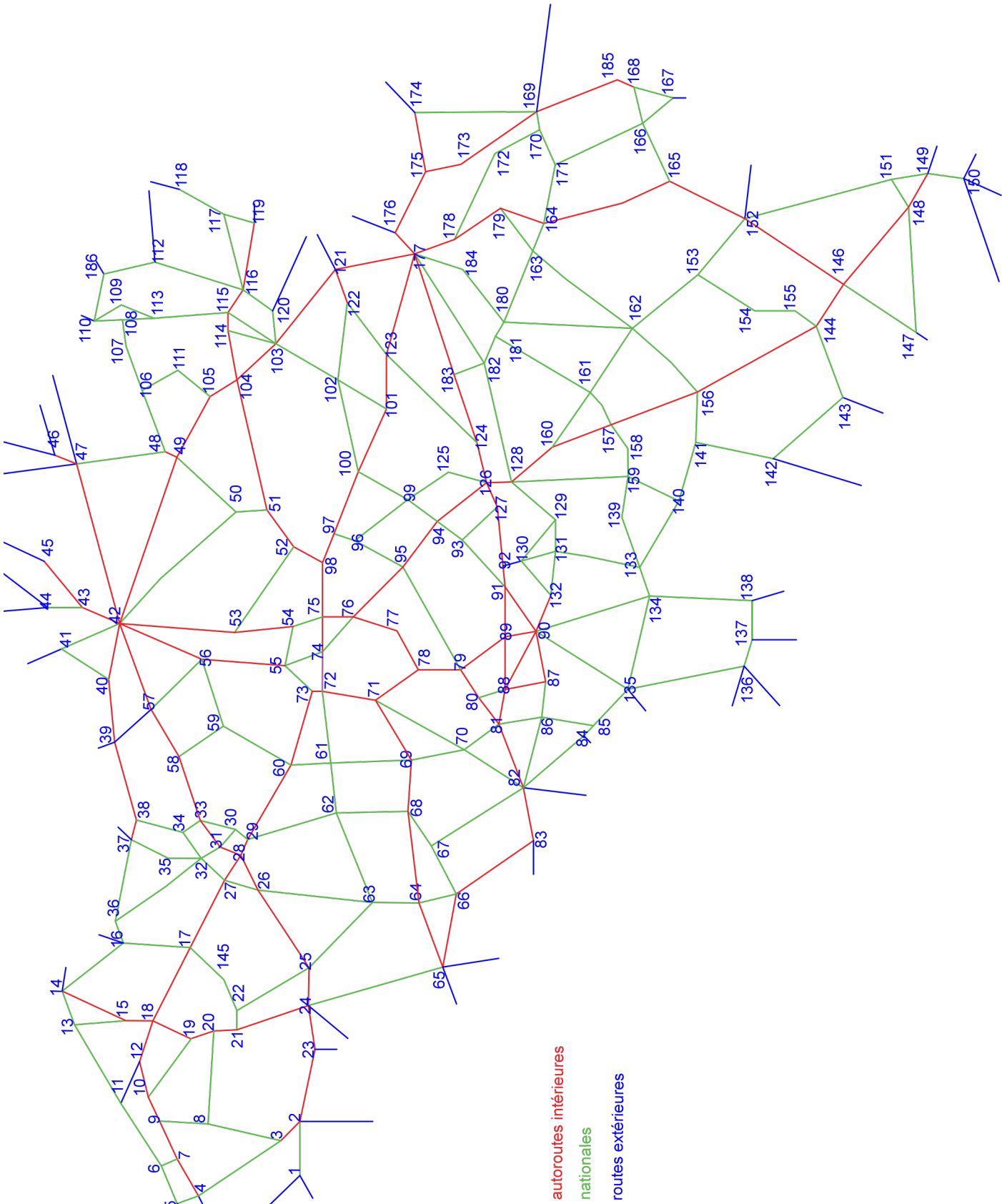


FIGURE 3.2 – Réseau routier de la simulation

3.1.2 Hypothèses employées

Joachim L'Hoir et Adrien Tellier ont utilisé plusieurs hypothèses pour construire leur modèle. Ces hypothèses ont été le point de départ de ce travail, qui visait à améliorer le modèle en levant des hypothèses. Voici la liste des hypothèses utilisées :

- Seules les autoroutes et les nationales les plus importantes sont représentée.
- Le trafic extérieur n'est pas pris en compte.
- Les villes de départ et d'arrivée sont choisies parmi 20 villes.
- Les véhicules se déplacent par paquet de 1000 véhicules.
- La simulation montre une image du nombre de voitures qui sont passées sur chaque tronçon à la fin de la journée.
- La capacité des routes est considérée comme infini, il n'y a donc pas de bouchons.
- Chaque tronçon de route possède une pondération qui va être utilisé pour déterminer le parcours le plus court. Cette pondération est soit la longueur, soit le temps de parcours.
- Tout les véhicules sont assimilés à des voitures.

3.2 Amélioration du modèle

Le nouveau code informatique prends comme fondation l'ancien, en y améliorant les hypothèses, comme suit :

- La fonction qui détermine les parcours des véhicules va être modifiée. La méthode de Monte-Carlo va choisir la ville de départ parmi 50 villes en fonction de la population. Plus une ville a d'habitants, plus elle a de chances d'être choisi. La ville d'arrivée va être choisie par la méthode de Monte-Carlo en fonction de la population ainsi que de la proximité avec la ville de départ.
- La dimension temporelle est ajoutée au modèle. Les véhicules se déplacent sur le réseau durant la journée.
- Le nombre de véhicules par paquet est diminué. Les analyses faites dans ce travail utilisent un véhicule par paquet mais l'utilisateur du programme

peut modifier cette valeur.

- Les paquets de véhicules se voient assigner une heure de départ.
- Chaque paquet de véhicules effectue le trajet aller et retour.
- Les bouchons sont modélisés.
- Les voitures et les camions sont différenciés.
- Le programme va prendre comme fichier en entrée la vitesse maximale autorisée sur chaque tronçon, son nombre de bandes ainsi que le fichier avec les coordonnées géographiques réalisé par Joachim et Adrien.

Chapitre 4

Etude du territoire

4.1 Introduction

Pour construire le modèle, il faut des données, par exemple le nombre de véhicules circulant sur les routes ou la proportion de camions. Ce chapitre fait le point sur ces données.

Le SPF Mobilité et Transport (Service Public Fédéral) a pour mission d'élaborer, de mettre en oeuvre et de soutenir la politique de mobilité et de transport en Belgique. Les principaux objectifs sont le développement économique, la sécurité, l'environnement et les intérêts sociaux. Il s'occupe des quatre réseaux de transport principaux en Belgique, à savoir la circulation routière, le transport aérien, la navigation fluviale et maritime et le transport ferroviaire.

Plus particulièrement, concernant le système de transport routier, le SPF Mobilité et Transport doit assurer le fonctionnement du secteur professionnel du transport routier, la sécurité routière et l'immatriculation des véhicules.

Il a également élaboré plusieurs documents avec des données de la mobilité belge. Ces documents sont analysés dans ce chapitre et sont disponibles sur leur site internet [25]. Toutes les données et figures présentées dans ce chapitre proviennent de ce site. Suite à analyse, les données utiles au modèle seront identifiées et permettront de construire le modèle. Néanmoins, il n'y a pas d'accès aux bases de données brutes, il y a seulement des tableaux "sur mesures".

4.2 Enquête sur la mobilité en Belgique

Pour pouvoir faire une bonne politique en matière de mobilité, il est indispensable de connaître les habitudes des usagers. C'est pour cela que le SPF Mobilité et Transports a commandé des grandes enquêtes nationales sur la mobilité en Belgique. Le but étant de mieux comprendre les pratiques de mobilité et d'identifier des pistes d'action pour une mobilité plus durable. Il y en a eu à ce jour trois : MOBEL (1999), BELDAM (2012) et MONITOR (en cours).

4.2.1 MOBEL

Une première enquête sur la mobilité en Belgique a été réalisée en 1999. Elle visait à déterminer l'organisation des activités des ménages et comment les déplacements vers ces activités étaient effectués. Les enquêteurs ont recueilli les données de 7000 personnes réparties dans toute la Belgique et de tous les milieux.

4.2.2 BELDAM

BELDAM pour BELgian DAily Mobility.

En 2008, les résultats de l'enquête MOBEL commençaient à être trop anciens. Il devenaient donc urgent de recommencer l'enquête. Une nouvelle enquête a donc été commandée pour mettre à jour MOBEL. Ainsi, en 2012 sont publiés les résultats de l'enquête. Elle a été réalisée selon la même méthodologie que MOBEL pour faire ressortir l'évolution des résultats. Au total, 7800 ménages répartis sur l'ensemble de la Belgique ont répondu à l'enquête entre 2009 et 2011. Les questions traitaient notamment des éléments suivants :

- la composition des ménages
- les équipements en matière de transport
- le statut socio-économique
- la répartition des déplacements dans le temps
- les motifs de déplacement
- les modes de déplacement
- la description de l'enchaînement des déplacements

Méthodologie

BELDAM s'intéresse au comportement général des ménages habitant en Belgique. L'enquête se base donc sur les ménages et non pas sur les individus, car elle considère que les décisions en matière de mobilité se décident en famille et non individuellement, la mobilité des enfants étant par exemple fort liée à celle des parents.

Pour que l'échantillon soit fiable et représentatif de la population, il convient de sélectionner les ménages aléatoirement, indépendamment de la région ou du revenu. Il a été déterminé statistiquement qu'il suffisait d'interroger seulement 2400 personnes pour obtenir une marge d'erreur de 2%.

Deux méthodes ont été utilisées pour récolter les données, le porte-à-porte et la voie postale. Le taux de réponses a été de 21%, contre 32% pour MOBEL.

4.2.3 MONITOR

En 2016, une troisième enquête est commandée. Le premier objectif de MONITOR est de mettre à jour MOBEL et BELDAM. Un second objectif est de faire des recherches sur la sécurité routière. Et donc relier les habitudes des usagers avec les accidents de la route. Pour cette enquête, c'est plus de 10 000 ménages belges qui ont été sondés.

Le rapport final n'est pas encore terminé, mais devrait l'être dans le courant de l'année 2019. Néanmoins, un très court rapport avec les premiers résultats a été publié avec l'évolution entre les 3 enquêtes.

4.3 Enquête sur les déplacements domicile travail

Depuis 2005, les employeurs qui occupent plus de 100 personnes ont l'obligation légale de répondre à une enquête de mobilité concernant les déplacements domicile-travail de leurs employés. Un quart des trajets effectués en Belgique se fait dans le cadre du travail. De plus, un trajet domicile-travail aura tendance à être plus long qu'un trajet domestique, comme aller faire les courses ou aller chercher les enfants à l'école par exemple. Notre modélisation ne prend en compte que les routes les plus importantes de Belgique, donc seuls les trajets longs nous concernent. Il est dès lors intéressant d'analyser en détail ces données.

Les entreprises couvertes ne représentent que 1,5 million de personnes, soit un tiers de la population active. De plus, les grosses entreprises sont souvent basées dans les villes, les résultats de l'enquête ne sont donc pas forcément représentatifs de la globalité de la Belgique. Cependant, notre modèle ne reprend que les plus grandes villes, donc ce n'est pas un réel problème pour notre étude.

Les résultats des enquêtes sortent tout les deux ans et la dernière disponible date de 2017.

4.4 Circulation sur le réseau routier

Les derniers documents utilisés dans notre travail sont des relevés des kilomètres parcourus sur les routes belges.

Les centres de contrôle technique répertorient le kilométrage des véhicules à chaque contrôle, ce qui permet d'avoir des données précises du nombre de kilomètres parcourus chaque année par les voitures immatriculées en Belgique. Les véhicules étrangers circulant en Belgique ne sont donc pas repris ici.

Enfin, le dernier document présente le nombre de véhicules par jour qui passent sur chaque tronçon du réseau routier belge. Ces données se basent sur le comptage des véhicules, réalisé par des stations de comptage ou des caméras. Cependant, ce comptage se fait sur des périodes de temps limitées et sur certains tronçons. Les chiffres répertoriés dans ces rapports sont donc partiellement basés sur des hypothèses et des simulations. De plus, la méthode de détermination est différente dans chacune des trois régions.

4.5 Analyse des données

Il y a énormément de données dans ces rapports, mais elles ne sont pas toutes utiles pour ce travail. Il y a quelques difficultés à traiter certaines données, car elles ne correspondent en générale pas exactement à notre recherche. Tous les trajets de mobilité sont considérés, pas seulement ceux en voiture. Il a aussi fallu isoler certaines données. Par exemple, notre modélisation reprend

uniquement les routes les plus importantes, et les données sont pour toutes les routes. La figure 4.1 montre la répartition des déplacements en fonction de la distance. Ces données sont faussées par les trajets sur les routes communales. Le tableau 4.3 reprend la distance moyenne pour rejoindre le lieu de travail, ceux effectué à pied également.

De plus, les données sont fournies sous forme de schéma et pas en tableau de valeur. La lecture de ces schémas est parfois imprécise.

Nous avons donc principalement tiré les données suivantes :

- Profil horaire des départs des usagers.
- Proportion de camions sur les routes.
- Nombre de passages de véhicules sur certains tronçons.
- Distance moyenne effectuée par les véhicules.

D'autres données, comme la distance et le nombre de trajet entre les différentes régions ont également été utilisées pour vérifier les paramètres du modèle, mais avec une incidence moindre.

Néanmoins, il n'existe pas de données concernant les parcours habituels des usagers. Connaître le nombre de passages de véhicules sur chaque tronçon n'est pas suffisant, car il faut connaître le parcours des véhicules pour déterminer le changement de parcours suite à une fermeture. Les parcours des usagers ont donc été déterminés de manière probabilistique.

Le chapitre suivant détermine les paramètres du modèle en analysant les données contenues dans les rapports présentés plus haut.

		Lieu du travail			
		Bruxelles	Flandre	Wallonie	Belgique
Lieu du domicile	Bruxelles	4,3	15,3	32,5	7,2
	Flandre	29,5	12,8	33,6	15
	Wallonie	43,9	41,1	13,4	19,4
	Belgique	24,4	13,3	14,2	15,5

TABLE 4.1 – Distance domicile-travail moyenne en fonction du lieu de domicile et du lieu de travail

	Bruxelles	Flandre	Wallonie	Belgique
Kilométrage annuel	14 111 km	16 655 km	17 773 km	16 853 km

TABLE 4.2 – Kilométrage annuel des voitures belges

	Bruxelles	Flandre	Wallonie	Belgique
Distance moyenne	8,7 km	12,2 km	13,4 km	12,3 km

TABLE 4.3 – Distance moyenne d’un déplacement domicile-travail

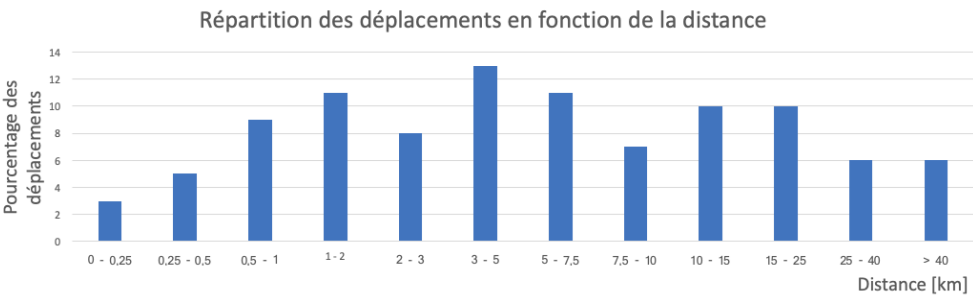


FIGURE 4.1 – Distribution des distances de déplacement

Chapitre 5

Construction du modèle

Ce chapitre explique la construction des différentes hypothèses utilisées pour construire le modèle.

Bien que traités séparément dans le rapport par souci de simplicité, les points 5.1, 5.5 et 5.6 concernant la détermination des trajets, le nombre de véhicules sur les routes et la proportion de camions ont été étudiés en parallèle car les décisions prises à un endroit influencent la solution des autres points.

5.1 Détermination des trajets

La première étape dans la construction du modèle est de déterminer une ville de départ et une ville d'arrivée pour chacun des véhicules. Le modèle ne considère que 50 villes hôtes, c'est-à-dire qui peuvent être choisies comme ville de départ ou d'arrivée. Ces villes ont été déterminées en fonction du nombre d'habitant ainsi que de leur position géographique pour avoir un échantillon répartie sur toute la Belgique. La table 5.1 liste l'ensemble des 50 villes hôtes, leur numéro de ville dans l'ensemble du réseau ainsi que leur population.

Numéro	Ville	Population
2	Ypres	34950
5	La Panne	11129
11	Ostende	71224
14	Knokke	33097
15	Bruges	138116
18	Loppem	5173
21	Roulers	62301
24	Courtrai	76242
27	Gand	259270
36	Eeklo	20890
39	Stekene	18352
42	Anvers	521680
47	Turnhout	41708
49	Tongerlo	7112
53	Malines	79503
57	Saint-Nicolas	73317
60	Alost	82037
63	Renaix	25429
65	Tournai	69493
69	Enghien	13103
74	Bruxelles	1139000
79	Nivelles	26843
81	La Louvière	78895
82	Mons	95265
90	Charleroi	204150

Numéro	Ville	Population
92	Sombreffe	8420
95	Louvain-la-Neuve	30283
100	Tirlemont	33133
102	Saint-Trond	37837
103	Hasselt	74694
107	Lommel	31536
116	Genk	66059
119	Maasmechelen	38193
122	Tongres	30650
128	Namur	110628
134	Philippeville	9239
136	Chimay	9896
141	Beauraing	9050
146	Neufchateau	7533
148	Arlon	29053
150	Aubange	16856
152	Bastogne	15127
159	Dinant	13912
162	Marche en famene	17487
164	Werbomont	4918
168	Saint-Vith	9481
169	Malmédy	12161
177	Liège	195875
182	Huy	21354
186	Hamont	14246

TABLE 5.1 – Liste des villes hôtes

5.1.1 Détermination de la ville de départ

La probabilité de sélection d'une ville est proportionnelle à sa population. Une représentation visuelle de la répartition est représentée sur la figure 5.1. Le réseau avec les villes hôtes dessinées en bleu est à la figure 5.2.

L'algorithme de sélection suit les étapes suivantes. Cet algorithme est du type de Monte-Carlo car le résultat est aléatoire. Une illustration de la méthode est proposée à la figure 5.3.

1. Les villes sont placées sur une ligne droite, la taille allouée à chaque ville est proportionnelle à sa population.
2. La droite est reliée à une échelle de 0 à 1.

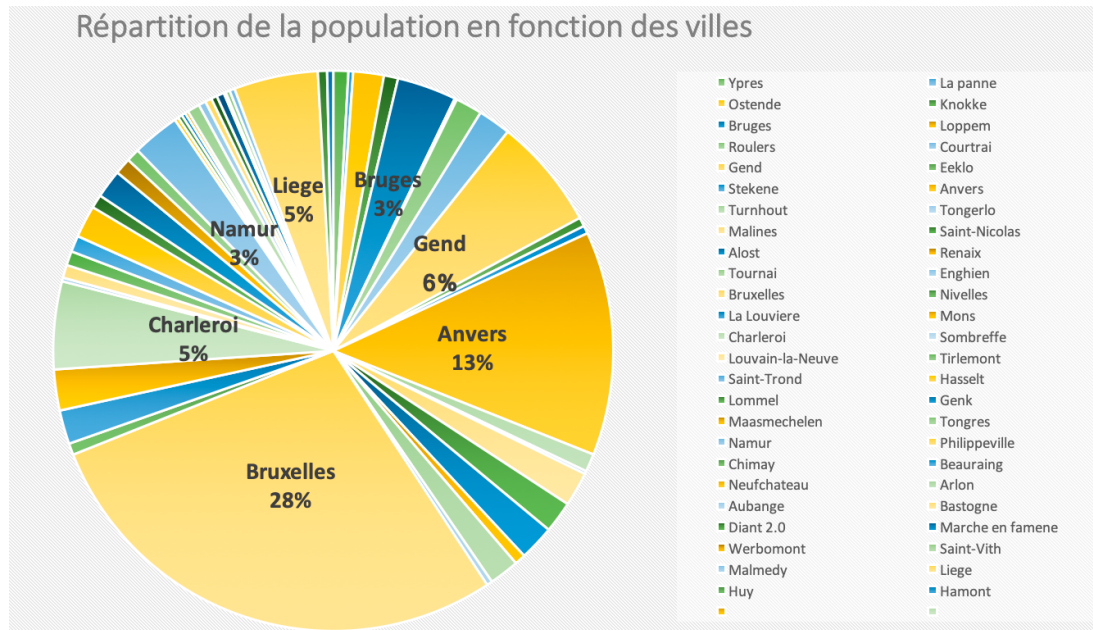


FIGURE 5.1 – Répartition de la population

- Un nombre entre 0 et 1 est tiré au sort, et la ville correspondante est choisie.

Ce mémoire fait appel à certaines notions de probabilité et l'algorithme utilisé provient directement des fonctions de distribution. L'annexe A présente un rappel sur les bases de la probabilité ainsi que les distributions statistiques. L'algorithme crée une distribution statistique des villes en fonction de leur population. Ensuite il crée la fonction de répartition associée qui permet de déterminer la ville de départ.

La distribution de la population peut se calculer avec le pseudo-code suivant.

```

1 pop(i)=population de la ville i
2 pop_total=sum(pop);
3 distribution_pop(1)=pop(1)/pop_tot;
4 n_ville=nombre de ville
5 for i=2:n_ville
6     distribution_pop(i)=pop(i-1)+pop(i)/pop_total;
7 end

```

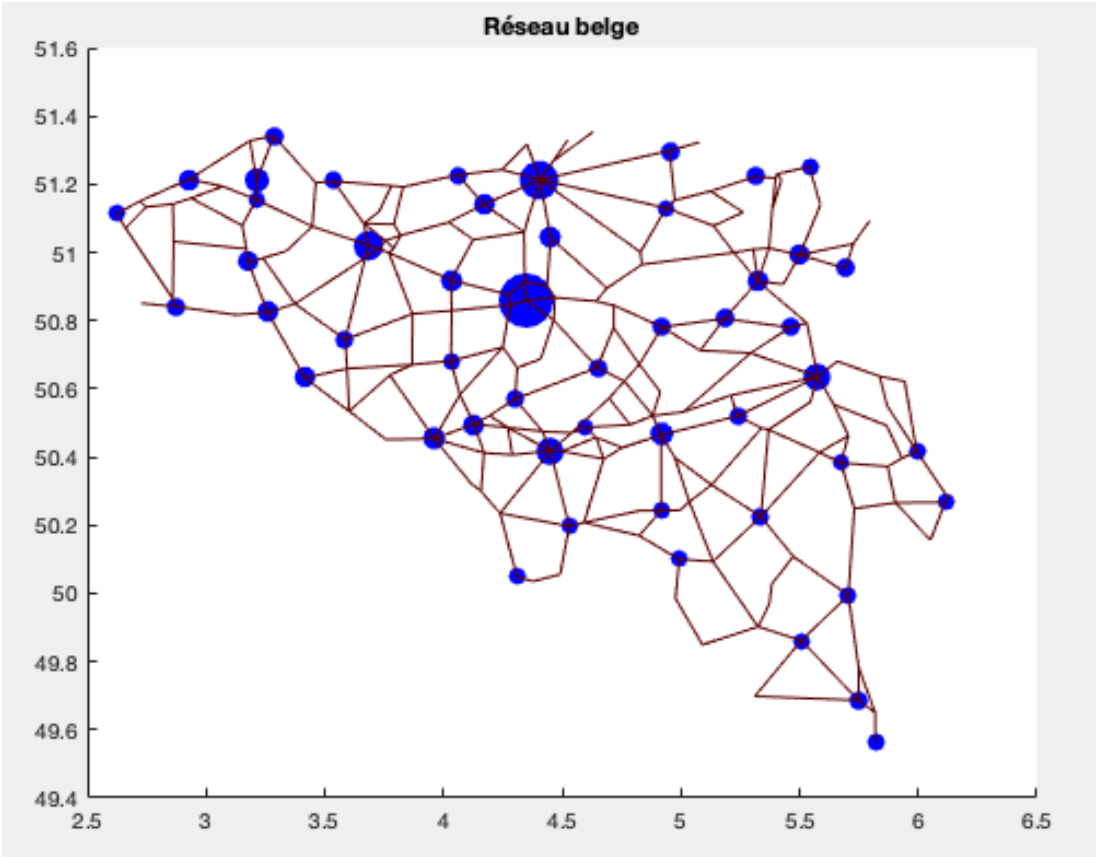


FIGURE 5.2 – Représentation des villes hôtes

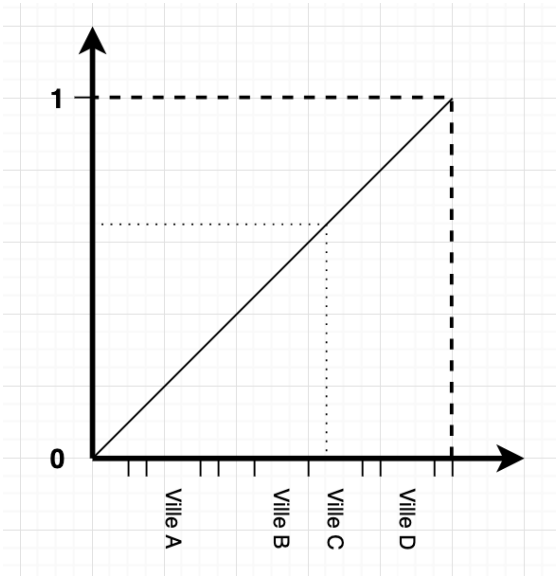


FIGURE 5.3 – Méthode de sélection des villes

5.1.2 Détermination de la ville d'arrivée

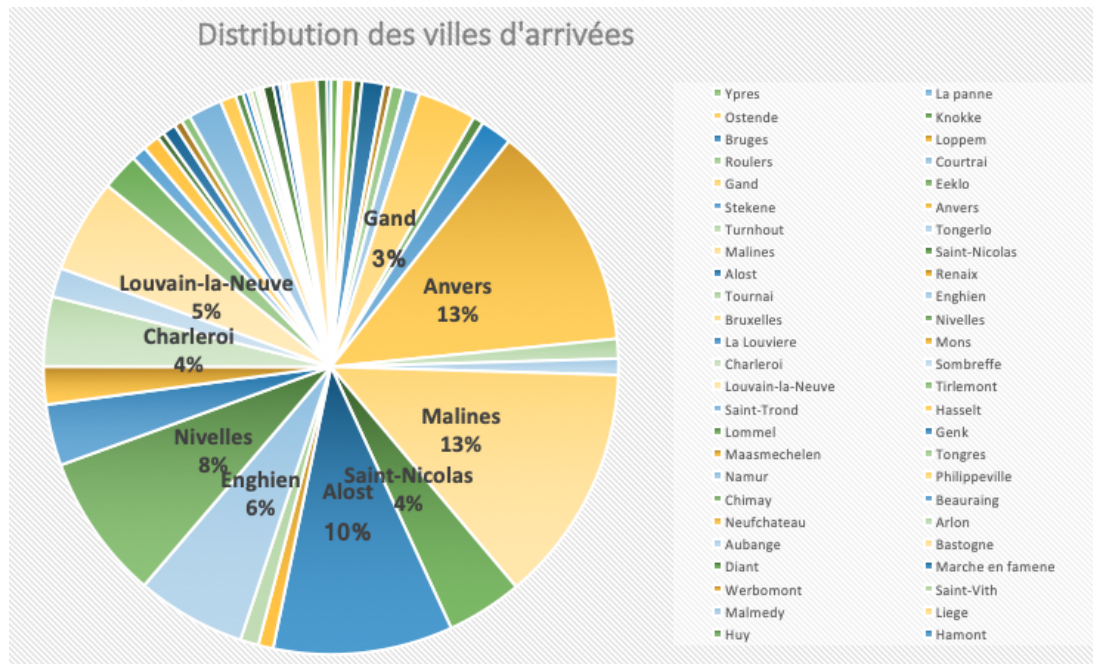


FIGURE 5.4 – Distribution des villes d'arrivée si Bruxelles est la ville de départ

La distribution de la population est la distribution entre 0 et 1 qui est associé aux villes de départ trouvées au point précédent. L'idée est maintenant de déterminer une nouvelle distribution pour déterminer la ville d'arrivée. Cette distribution doit être proportionnelle à la population et à la distance par rapport à la ville de départ. Une nouvelle distribution va être créée en prenant la distance en compte. La distribution finale va être une somme pondérée des distributions de la population et de la distance.

Pour garder la distribution finale entre 0 et 1, il est nécessaire de définir un paramètre λ tel que : Distribution finale = $\lambda \times$ distribution de la population + $(1 - \lambda) \times$ distribution de la distance.

Ces paramètres vont être déterminés en faisant correspondre la longueur moyenne des parcours à la distance moyenne parcourue par les voitures sur les autoroutes.

Les données de Beldam sont faussées par les trajets effectués sur les routes communales, qui représentent plus de la moitié des trajets. Beldam établit la longueur du trajet moyen à 10 kilomètres, ce qui est inutilisable dans notre simulation.

Le tableau 5.2 sépare les données des autoroutes, des nationales et des routes communales. Il donne le nombre de voyageurs-kilomètres sur les routes par an. Avec cela, la distance moyenne d'un trajet pourra être déterminée.

	Wallonie	Flandre
Autoroutes	$15,4 \times 10^9$	24×10^9
Nationnales	$18,2 \times 10^9$	$25,6 \times 10^9$
Routes communales	$9,4 \times 10^9$	$10,8 \times 10^9$

TABLE 5.2 – Nombre de voyageurs-kilomètres parcourus par an

Il y a donc $83,2 \times 10^9$ voyageurs-kilomètres par an sur les autoroutes et nationales. Si le trafic durant le week-end est supposé deux fois moins important, cela donne $265,8 \times 10^6$ voyageurs-kilomètres par jour de semaine. Il y a 1,6 millions de voyageurs journalier en voiture sur notre réseau (voir point 5.5). Cela correspond à 165 km par jour et par voyageur. Si le modèle considère un trajet aller et retour, cela donne 75 kilomètres par trajet.

La distribution de la distance doit donner un poids plus important aux villes proches. Il faut maintenant déterminer l'attribution d'un poids à chaque ville ainsi que λ pour être le plus proche possible de la valeur de 75 km par trajet.

Le poids donné à chaque ville est l'inverse du cube de la distance. Une ville qui est deux fois plus proche de la ville de départ a donc huit fois plus de poids.

Le paramètre λ vaut 0,25. A noter que le programme évite que la ville d'arrivée soit la même que la ville de départ. La figure 5.4 montre la distribution final au cas où Bruxelles a été choisies comme première ville.

En pseudo-code, si la ville k a été choisie en premier :

```

1  for i=1:n_ville % Sauf pour i=k
2      poids_dist(i)=1/(distance(Ville(i), Ville(k)))^3;
3  end
4  poids_total=sum(poids_dist);
5  distribution_dist(1)=poids_dist(1)/poids_total;
6  for i=2:n_ville
7      distribution_dist(i)=distribution_dist(i-1)+ ...
8          poids_dist(i)/poids_total;
9  end
10
11 lambda=0.25
12 distribution_final=lambda*distribution_pop + ...
13     (1-lambda)*distribution_dist;
```

La distribution finale est bien entre 0 et 1.

Un problème a ensuite été rencontré. Si une grande ville est excentrée par rapport aux autres, elle a beaucoup de chance d'être choisie comme ville de départ mais peu comme ville d'arrivée. Cela crée des différences importante

entre les deux sens de circulation d'une route. Cela semble peu logique car la différence est censée être assez faible. De plus, le comptage des véhicules sur les routes ne constate aucune différence de passage entre les deux sens de circulation. Un trajet retour suivant le même parcours en sens inverse que le trajet aller a donc été introduit. Avec cela, il y a exactement le même nombre de véhicules dans les deux sens de circulation dans le modèle.

5.2 Trouver le chemin le plus court

Une fois les villes de départ et d'arrivée déterminées, il faut calculer le plus court chemin entre ces deux villes. La solution la plus simple serait de calculer tous les chemins possibles et prendre le plus court. Cependant, étant donné la quantité importante de noeuds et d'arêtes, cette méthode prendrait énormément de temps de calcul. Il est nécessaire de trouver une méthode plus efficace.

Chaque arrête a un certain poids. Ce poids peut-être soit la distance soit le temps de parcours théorique. Le chemin le plus court est le chemin qui minimise la somme des poids des arrêtes parcourus. La théorie des graphes propose plusieurs algorithmes qui calculent le plus court chemin.

Dans le modèle, le poids des arrêtes est fixe. Cela veut dire que les problèmes de congestion ne sont pas pris en compte dans le choix des parcours.

5.2.1 Mesure de complexité d'un algorithme

La mesure de la complexité d'un algorithme mesure la quantité de ressources nécessaires à l'exécution de l'algorithme. Dans ce travail la ressource va être le temps, mais il est également possible de le faire avec la quantité de mémoire nécessaire. L'intérêt de mesurer la complexité d'un algorithme permet de déterminer son efficacité.

La complexité, noté $O(f(n))$, signifie que le temps de calcul de l'algorithme pour n entrées est proportionnel à $f(n)$. Ce temps de calcul est compté dans le pire des cas.

Par exemple, si la tâche est de trouver un mot parmi une liste de n mots trié par ordre alphabétique et que l'algorithme choisi est de parcourir tous les mots de liste pour trouver le bon, la complexité vaut : $O(n)$. Car, dans le pire des cas le mot recherché est le dernier de la liste et il a fallu parcourir toute la liste. S'il y a deux fois plus de mots, le temps de calcul est deux fois plus long.

Si l'algorithme choisi est de prendre le mot du milieu, regarder si le mot est avant ou après dans l'alphabet et ensuite recommencer avec une demi liste. Dans ce cas ci, la complexité est $O(\log_2(n))$. Si la liste est deux fois plus grande,

le temps de calcul est inférieure à deux fois le temps initial. Le deuxième algorithme est donc plus efficace que le premier.

5.2.2 Algorithme de Dijkstra

L'algorithme de Dijkstra commence par prendre un noeud de départ et va déterminer le chemin le plus court pour rejoindre tous les autres noeuds. C'est considéré comme l'algorithme le plus efficace pour trouver le chemin le plus court dans un graphe non-orienté et sans arrête de poids négatif. La complexité vaut $O(a + n \log n)$, où a est le nombre d'arêtes et n le nombre de noeuds.

Procédure

L'algorithme suit la procédure suivante :

Soit s le noeud de départ et P l'ensemble des noeuds moins le noeud s .

$d(k)$ est la distance provisoire du noeud k .

$l_{j,k}$ est le poids de l'arrête reliant le noeud j et k .

1. Chaque noeud est marqué de façon temporaire : $d(i) = +\infty$
2. La distance provisoire de tous les noeuds i voisins à s est mise à jour à $d(i) = l_{s,i}$
3. Le noeud k est déterminé tel que $d(k) = \min(d(i))$ pour tout $i \in P$
4. Le noeud k est retiré de l'ensemble P
5. La distance $d(k)$ est marquée comme définitive
6. La distance provisoire de tout les noeuds $i \in P$ voisins à k sont mises à jour à $d(i) = \min\{d(i); d(k) + l_{k,i}\}$
7. Si l'ensemble P est non nul, reprendre à l'étape 3.
8. L'ensemble des noeuds i sont marquées par une distance définitive qui correspond au chemin le plus court entre s et i .

Exemple :

Cet algorithme est compliqué à comprendre sans une illustration. L'exemple est appliqué au graphe de la figure 5.5 et a pour point de départ le noeud A.

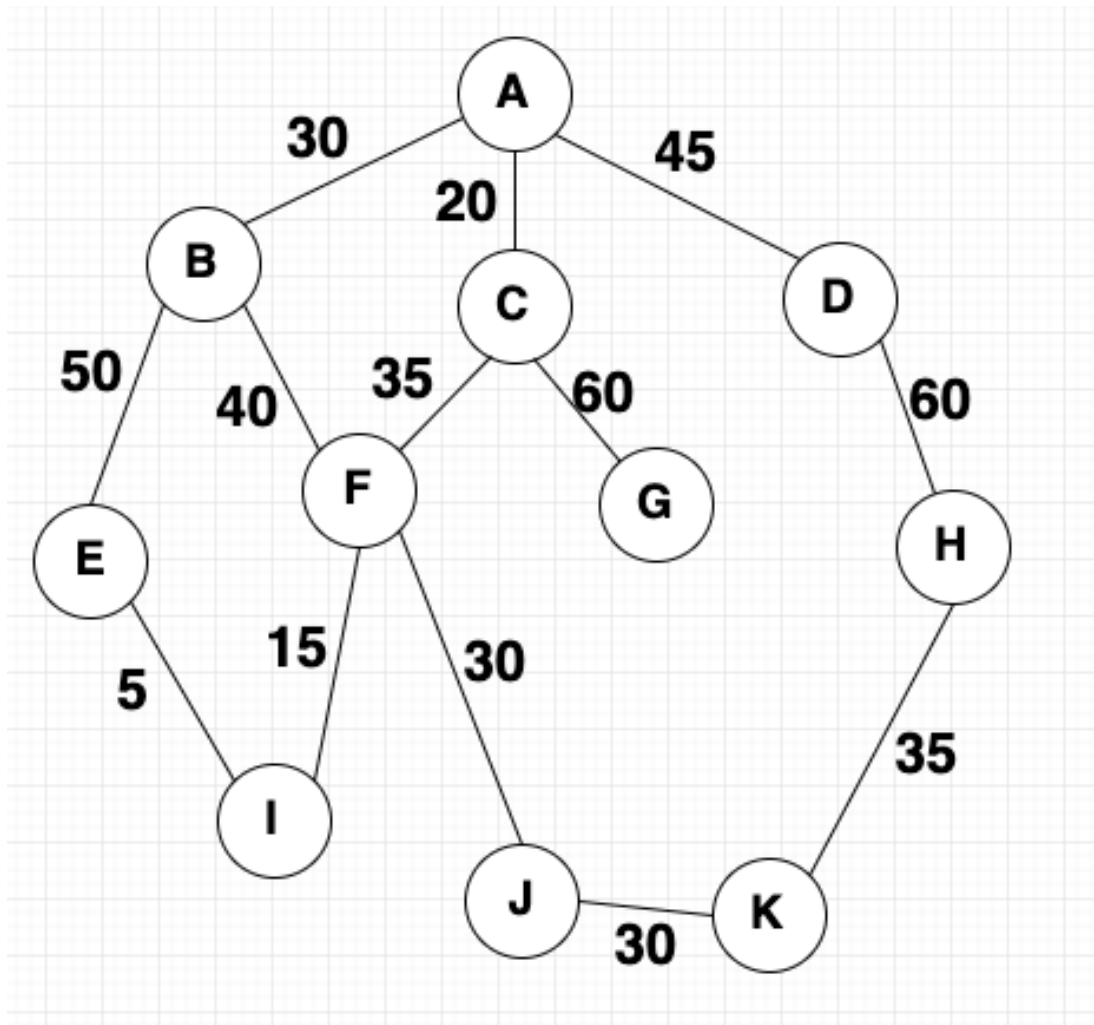


FIGURE 5.5 – Graphe du test de l'algorithme de Dijkstra

Etape 1 :

Première itération, tout les noeuds ont provisoirement une distance infinie.

B	C	D	E	F	G	H	I	J	K
∞	∞	∞	∞	∞	∞	∞	∞	∞	∞

Etape 2 :

Les noeuds voisins du noeud A ont provisoirement la distance de l'arête correspondante.

B	C	D	E	F	G	H	I	J	K
30	20	45	∞	∞	∞	∞	∞	∞	∞

Etape 3 :

La distance provisoire la plus petite est marquée comme définitive, car il est impossible qu'il y ait un chemin plus court car tous les autres chemins passent obligatoirement par une arrête plus longue. Les arrêtes ont obligatoirement une distance positive.

Dans l'exemple, c'est la distance A-C qui est marqué comme définitive. Les noeuds voisins de C sont maintenant marqués provisoirement de la distance par rapport au point A.

B	C	D	E	F	G	H	I	J	K
30	20	45	∞	55	80	∞	∞	∞	∞

Etape 4 :

En suivant la même logique, la distance provisoire la plus petite est marquée comme définitive. C'est la distance provisoire A-B, et la distance des voisins de B sont mis à jour.

Le noeud F est voisin de B mais possède déjà une distance provisoire. La nouvelle distance provisoire de F va être le minimum entre la distance provisoire et la distance du chemin passant par le noeud B. Ici, la distance par le noeud B vaut 70, ce qui est supérieur au 55 actuel. La distance provisoire reste donc 55.

B	C	D	E	F	G	H	I	J	K
30	20	45	80	55	80	∞	∞	∞	∞

Etape 5 :

La distance provisoire A-D est marquée comme définitive et la distance provisoire A-H est maintenant 105.

B	C	D	E	F	G	H	I	J	K
30	20	45	80	55	80	105	∞	∞	∞

Etape 6 :

La distance provisoire A-F est marquée comme définitive.

B	C	D	E	F	G	H	I	J	K
30	20	45	80	55	80	105	70	85	∞

Etape 7 :

La distance provisoire A-I est marquée comme définitive et la distance provisoire A-E devient 75. La distance entre A et E vaut 80 en passant B, et 75 en passant par C, F et I. Donc le trajet par le noeud B est éliminé.

B	C	D	E	F	G	H	I	J	K
30	20	45	75	55	80	105	70	85	∞

Etape 8 :

La distance provisoire A-E est marquée comme définitive.

B	C	D	E	F	G	H	I	J	K
30	20	45	75	55	80	105	70	85	∞

Etape 9 :

La distance provisoire A-G est marquée comme définitive.

B	C	D	E	F	G	H	I	J	K
30	20	45	75	55	80	105	70	85	∞

Etape 10 :

La distance provisoire A-J est marquée comme définitive.

B	C	D	E	F	G	H	I	J	K
30	20	45	75	55	80	105	70	85	115

Etape 11 :

La distance provisoire A-H est marquée comme définitive.

B	C	D	E	F	G	H	I	J	K
30	20	45	75	55	80	105	70	85	115

Etape 11 :

La distance provisoire A-K est marquée comme définitive et la totalité des parcours sont calculés.

B	C	D	E	F	G	H	I	J	K
30	20	45	75	55	80	105	70	85	115

5.2.3 Algorithme de Bellman-Ford

L'algorithme de Bellman-Ford a été développé en 1956 par R. Bellman et L. Ford. Contrairement à celui de Dijkstra, il accepte les arrêtes de poids négatif. Cependant, considérer des arêtes de poids négatif n'a pas de sens dans un réseau routier. La complexité est $O(a \times n)$, où a est le nombre d'arêtes et n le nombre de noeuds. Cela est supérieur à Dijkstra.

L'algorithme de Bellman-Ford est très proche de l'algorithme de Dijkstra. Il va définir la distance provisoire de tous les noeuds à l'infini, puis il va parcourir tout les noeuds et mettre à jour la distance provisoire s'il est inférieur à la précédente. La différence avec Dijkstra réside dans le fait que Bellman-Ford ne marque aucun noeud comme définitif, et donc continue à effectuer des calculs sur tous les noeuds à chaque itération.

5.2.4 Algorithme A*

L'algorithme A* a été crée par P. Hart, N.P Nilsson et B. Raphael en 1959. Il a été développé à partir de celui de Dijkstra, avec l'idée que la première solution trouvée sera la meilleur. Cet algorithme privilégie la rapidité du calcul par rapport à son exactitude. C'est intéressant dans certaines situation mais une solution exacte est préférable dans ce travail.

L'algorithme de Dijkstra va partir d'un point A et étendre son champ de recherche de façon circulaire pour chercher le point d'arrivée B. Par exemple, l'algorithme A* va concentré sa recherche vers le nord si le point B est plus au nord que A, quitte à ne pas trouver la solution optimale. Pour certains réseaux, comme des labyrinthes, l'algorithme A* n'est pas efficace.

5.2.5 La fonction Dijkstra de Matlab

Joseph Kirk a écrit une fonction Matlab qui calcule le plus court chemin a l'aide de l'algorithme de Dijkstra. Cette fonction est disponible gratuitement sur le site *mathworks*, qui est le site de Matlab. Elle est disponible dans l'annexe E.

Si n est le nombre de noeuds, elle va prendre en entrée la matrice d'incidence, c'est-à-dire une matrice $n \times n$ où l'élément $a_{i,j} = 1$ s'il y a une arrête entre i et j et $a_{i,j} = 0$. Elle va également prendre en entrée une matrice $n \times n$ où l'élément $a_{i,j} = d_{i,j}$, où $d_{i,j}$ est le poids de l'arrête $i-j$.

Ella va retourner deux matrices $n \times n$.

La première matrice est la matrice coût. l'élément $c_{i,j}$ vaut le coût du chemin minimum entre i et j .

La deuxième matrice est la matrice des parcours, l'élément p_{ij} est un vecteur qui contient tous les noeuds qui sont traversés par le chemin le plus court entre i et j .

5.3 Modélisation de la congestion

Plusieurs travaux ont été publiés concernant le flux de trafic, en vue d'étudier les interactions entre les usagers. Cette section va reprendre les notions et les études pour modéliser la congestion.

5.3.1 Static and dynamic traffic assignment

Il faut tout d'abord faire la différence entre l'affectation statique et dynamique du trafic [26].

L'affectation statique du trafic assigne à chaque voyageur un trajet qui ne changera pas durant le temps de parcours. Ce trajet peut prendre en compte ou non les prévisions d'embouteillages. Les congestions sont modélisées par un retard sur le temps de trajet final. Ce retard est calculé par une fonction qui dépend de la densité de véhicules qui ont emprunté les routes.

L'affectation dynamique du trafic laisse le choix à chaque voyageur de modifier son trajet en cours de route en fonction des bouchons, des prévisions de bouchons, des accidents et autres. Les congestions sont modélisées en temps réel par une diminution de vitesse en fonction de la densité de véhicules.

Dans ce modèle, le choix a été de prendre un mix des deux méthodes d'assignation. Le parcours des véhicules ne change pas au cours du trajet et ne prend pas en compte les prévisions de congestion. Les embouteillages sont modélisés par une diminution de vitesse. La mise en place de l'affectation dynamique pourra faire l'objet d'une amélioration pour une version future de ce modèle.

5.3.2 Flux de trafic

Le flux est considéré comme libre s'il y a moins de 5 véhicules par kilomètre et par bande. Aucune voiture n'est impactée par le comportement des autres.

Le flux est considéré comme stable s'il y a entre 5 et 20 véhicules par kilomètre et par bande. Les voitures interagissent entre elles mais la vitesse des véhicules reste proche de leur vitesse maximale.

Au delà, le flux est considéré comme instable, de petites perturbations peuvent entraîner des ralentissements importants.

Dans ce travail, l'hypothèse que les usagers commencent à diminuer leur vitesse à partir de 40 véhicules par kilomètre et par bande a été retenue.

Définition des quantités :

Densité ρ : Nombre de véhicules par unité de longueur et par bande, à un temps t fixé.

Flux f : Nombre de véhicules passés par une position fixée par unité de temps.

Vitesse u : distance parcourue par unité de temps.

L'écart temporel h_t est le temps de passage entre deux véhicules à un point fixe.

L'écart spatial h_s est la distance entre deux véhicules à un temps donné.

De ces termes découlent les relations suivantes :

$$f = \frac{\#bande}{h_t}$$

$$\rho = \frac{\#bande}{h_s}$$

$$f = \rho u$$

Adaptation de la vitesse des véhicules

La densité de voitures est facile à calculer dans le code Matlab : il suffit de prendre le nombre de véhicules sur un certain tronçon, puis de diviser ce chiffre par la longueur du tronçons et le nombre de bandes. Dans ce modèle, la longueur du tronçon retenu est de 1 km.

Pour déterminer la diminution de vitesse, il faut trouver la relation qui lie vitesse et densité.

la vitesse u peut être calculée comme :

$$u = \frac{f}{\rho}$$

Ensuite, il faut déterminer le flux f . Expérimentalement, l'allure de la courbe du flux en fonction de la densité de voiture est représentée à la figure 5.6 [27].

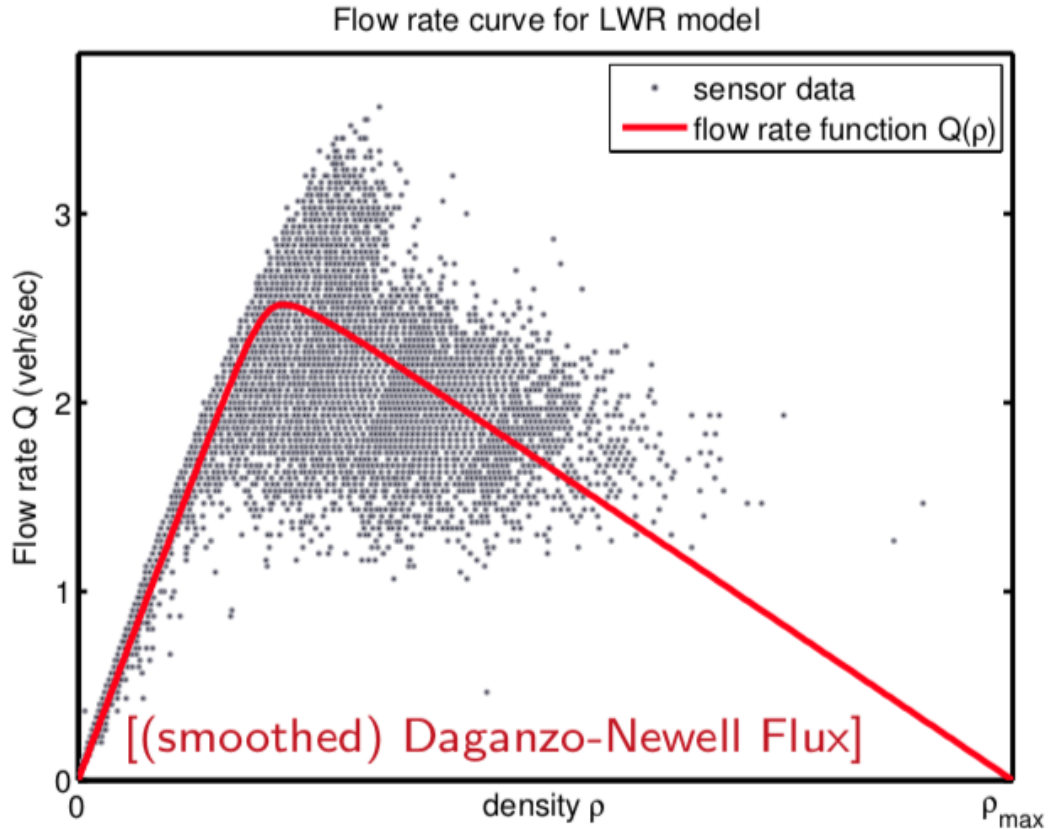


FIGURE 5.6 – Courbe du flux f en fonction de la densité de voitures ρ

Suite à des recherches sur plusieurs documents [26] [27] [28], les paramètres suivants ont été fixés :

$\rho_{crit} = 40$ [Véhicules/km/bande], densité à partir de laquelle la vitesse est diminuée. Cela correspond à une voiture tous les 25 mètres.

$\rho_{max} = 140$ [Véhicules/km/bande], densité maximale des véhicules, les voitures sont à l'arrêt. Cela correspond à une voiture tout les 7 mètres environ.

La figure 5.7 correspond à la relation qui relie le flux f à la densité ρ dans le modèle Matlab. La figure 5.8 correspond à la vitesse des voitures en fonction de la densité.

Concrètement le code Matlab va séparer chaque route en tronçons d'un kilomètre et compter le nombre de véhicules présents. Un camion compte pour 2,5 voitures. Ensuite, il va calculer un coefficient de diminution entre 0 et 1. Les véhicules ont leur vitesse égale à la vitesse maximale autorisée multipliée par

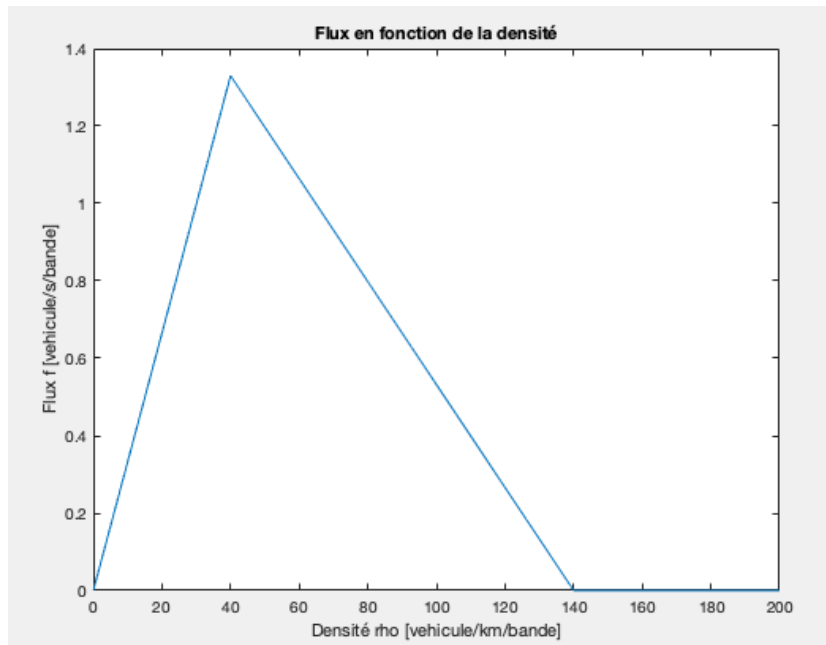


FIGURE 5.7 – Courbe modélisée du flux f en fonction de la densité ρ

le coefficient de vitesse.

Ce coefficient de diminution de vitesse dépend du nombre de bandes et de la quantité de véhicules, mais pas du type de route, autoroute ou nationale. Le modèle considère que les véhicules roulent à 50% de leur vitesse maximale si la densité vaut environ 60 véhicules par bande. Cette hypothèse est justifiée par le fait que les autoroutes disposent de plus de bande, et donc permettent le changement de bande en cas de ralentissement. Les routes nationales semblent moins sécurisantes, et donc la vitesse est inférieure à celle des autoroutes pour une densité équivalente. De même pour les camions, en cas de ralentissement léger, ils gardent une vitesse plus faible que les voitures. Cependant, une amélioration de ces hypothèses peuvent avoir du sens dans un travail futur.

Enfin, le modèle va permettre une vitesse minimale aux véhicules. Cette vitesse minimale est justifiée dans le sens où, si la densité est maximale, les véhicules ne suivent plus la courbe 5.6 et ils continuent d'avancer très lentement. Cela permet d'éviter à certains tronçons de stopper tout le trafic durant la journée. Dans ce modèle, la vitesse minimale est fixé à 10% de la vitesse maximale.

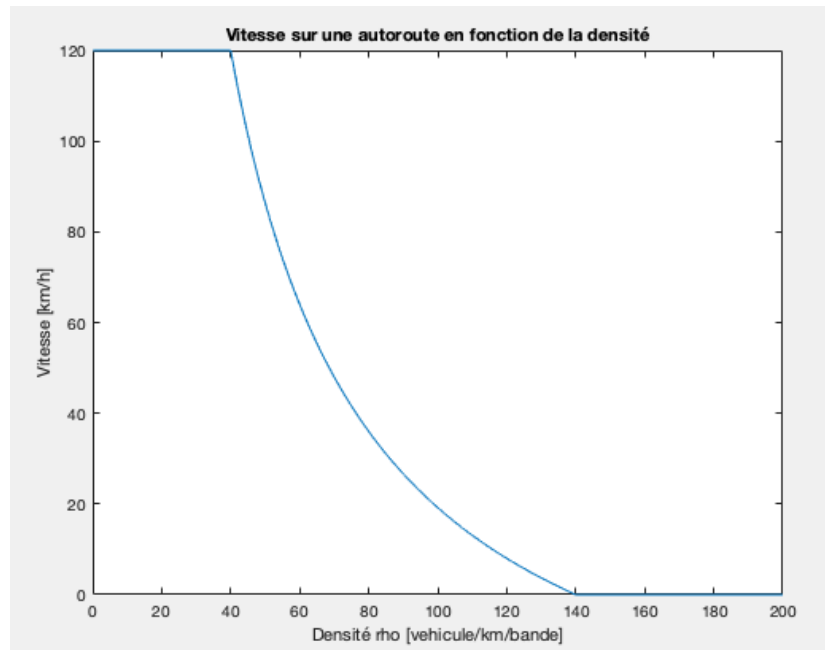


FIGURE 5.8 – Courbe modélisée de la vitesse en fonction de la densité ρ

5.4 Heure de départ

La figure 5.10 provient de Beldam et montre la distribution des heures de départ des usagers pour un jour ouvrable. Pour pouvoir l'utiliser, il faut calculer la fonction de distribution cumulative. Pour effectuer cela, deux choix ont été possibles. Le premier est de modéliser une distribution bimodale des heures de départ. Le deuxième choix est de créer une distribution directement à partir des données du graphique.

La distribution des départs suivant une loi bimodale

La loi bimodale est expliquée dans l'annexe A. La distribution bimodale f , est la somme pondérée de deux distributions normales g_1 et g_2 , et elle est dépend de 5 paramètres :

- μ_1 : Espérance de la distribution g_1
 - σ_1 : Ecart-type de la distribution g_1
 - μ_2 : Espérance de la distribution g_2
 - σ_2 : Ecart-type de la distribution g_2
 - λ , l'intégrale de f doit être égale à 1, donc obligatoirement
- $$f(t) = \lambda \times g_1(t) + (1 - \lambda) \times g_2(t)$$

Cette solution s'est avérée peu adéquate en raison de sa mauvaise représentation de la réalité, notamment à cause du faible nombre de paramètres à faire varier.

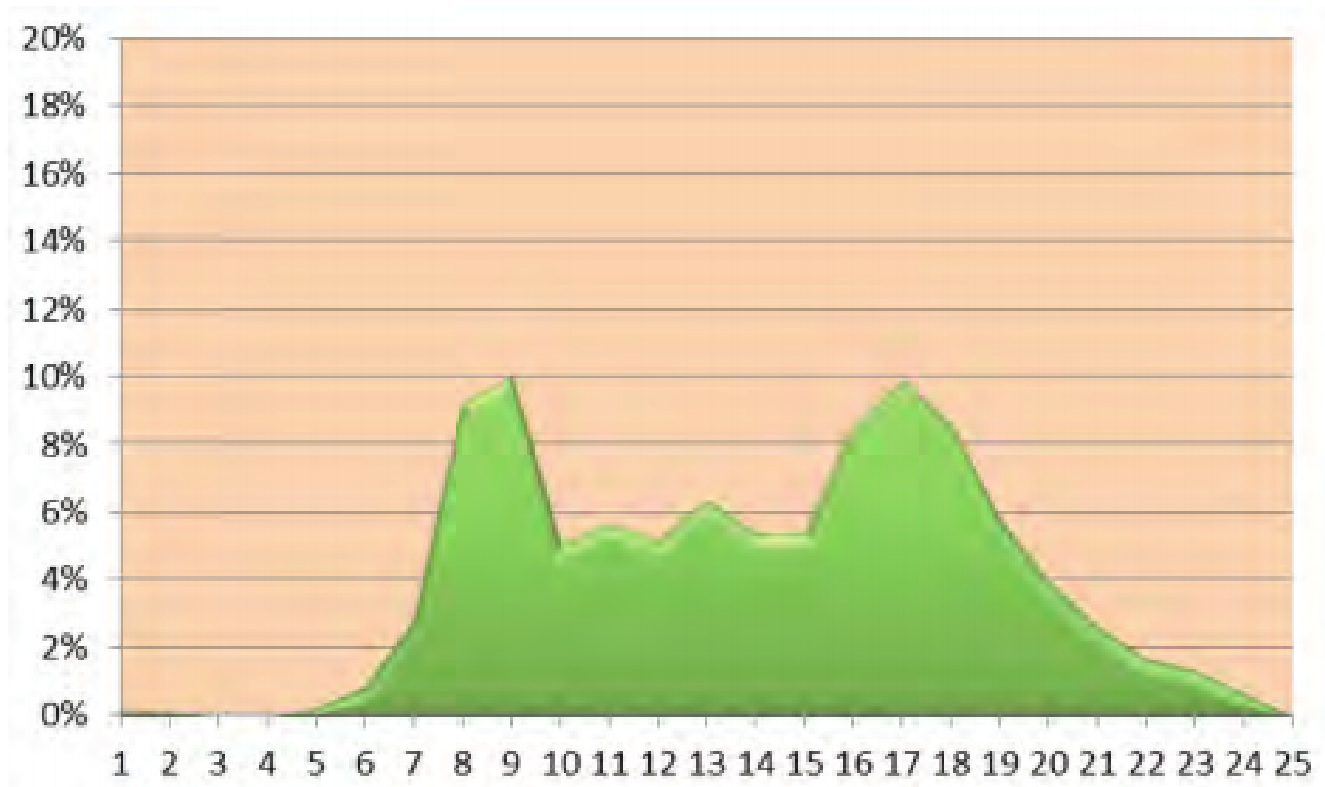


FIGURE 5.9 – Répartition horaire des déplacements d'un jour ouvrable

La distribution des départs suivant le graphe de Beldam

La solution utilisée dans ce modèle a été de recopier les valeurs des heures des départs, et de créer une distribution dans Matlab.

Pour déterminer l'heure de départ du trajet retour, la même distribution est utilisée mais Matlab force l'heure de retour à être séparée de l'heure de départ au minimum du temps théorique du trajet.

Au final, la distribution suivante est obtenue (figure 5.10). Cette distribution ressemble suffisamment à celle de Beldam pour être acceptée.

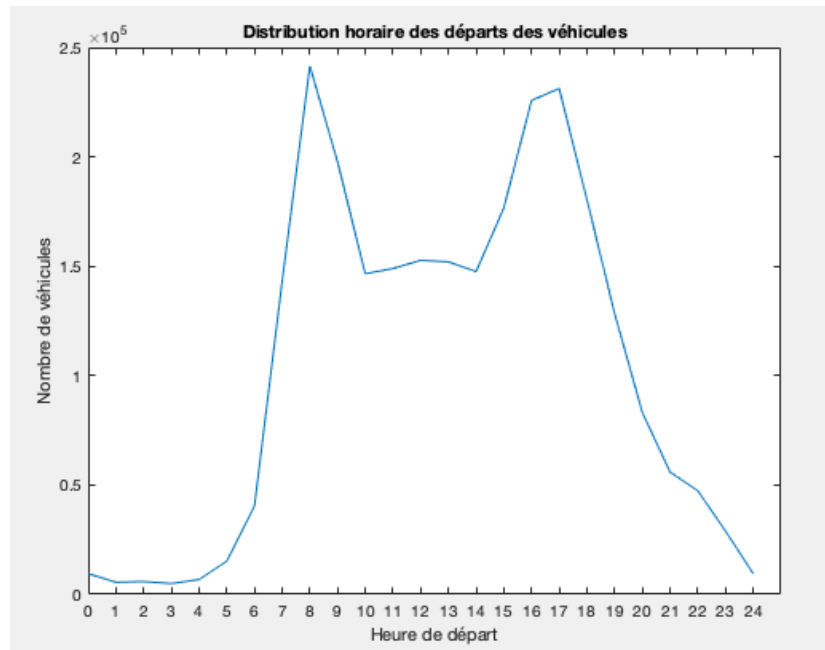


FIGURE 5.10 – Distribution horaire des départs dans la simulation

5.5 Nombre de véhicules à simuler

La détermination de nombre de véhicules à simuler a été effectuée en essayant de faire concorder avec les chiffres de passage de véhicules (voitures et camions) réel quelques tronçons sélectionnés. Les données disponibles ne datent pas de 2019, une extrapolation des données a été effectuée. De plus, les données concernent le nombre de passages entre deux sorties, une valeur moyenne a été calculé pour le tronçon en entier. Le nombre de véhicules de la simulation est calculé en faisant la moyenne de plusieurs itérations du code.

Au final, c'est 1,6 millions de véhicules qui sont simulés. Ces véhicules effectuent le trajet aller et retour.

Les tronçons sur la E 19 sont sur-représentés, tandis que ceux sur la E 411 sont sous-représentés. Sur les nationales, le trafic donné par la simulation n'est pas coordonnée avec le trafic réel. Ces différences sont acceptées dans les hypothèses de ce modèle, mais une amélioration de l'algorithme de recherche des parcours est envisageable dans un travail ultérieur.

Nom de la route	Numéro du tronçon	Nombre réel de passages	Nombre de passages dans la simulation
E 411 (Bruxelles - Louvain-la-Neuve)	252	120 000	85 500
E 411 (Wellin - Libramont)	476	30 000	20 500
E 17 (Courtrai - Warregem)	70	81 000	55 000
E 19 (Bruxelles - Nivelles)	257	96 000	134 500
E 19 (Anvers - Malines)	136	120 000	138 000
E 25 (Chaudfontaine - Aywaille)	583	37 000	24 000
E 40 (Aalter - Loppem)	47	69 000	75 000
N 55 (Enghien - Soignies)	224	7 000	16 000
N 80 (Waremmes - Fernelmont)	409	7 500	1 500
N 86 (Marche-en-Famenne - Ferrières)	531	9 100	1 200

TABLE 5.3 – Nombre de passages de véhicules sur les routes

5.6 Prise en compte des camions

Un camion occupe plus d'espace et roule moins vite qu'une voiture. Il est dès lors intéressant de les différencier des voitures.

	Camions	Bus et autocars
Route nationale	7%	1%
Autoroutes	14,5%	0,5%

TABLE 5.4 – Pourcentage de camions et d'autobus sur les routes belges

Le modèle ne fait aucune distinction entre un camion ou un bus. La simulation qui détermine les trajets considère que 8% des véhicules simulés sont des PL (poids lourds). En plus de cela, le programme ajoute des parcours de camions selon les grands flux de marchandises à travers la Belgique. Ces flux ont été étudiés par Mathieu Strale [29]. La figure 5.11 montre les différents axes utilisés par les camions sur le réseau belge. A partir de cela, quatre axes ont ainsi été distingués. Ils sont :

- Lille - Gand - Anvers - Breda : 5 000 PL
- Valenciennes - Mons - Charleroi - Namur - Liège - Aix-La-Chapelle : 2 000 PL
- Dunkerque - Bruges - Gand - Bruxelles - Namur - Arlon - Luxembourg : 1 000 PL
- Breda - Anvers - Bruxelles - Namur - Arlon - Luxembourg : 1 000 PL

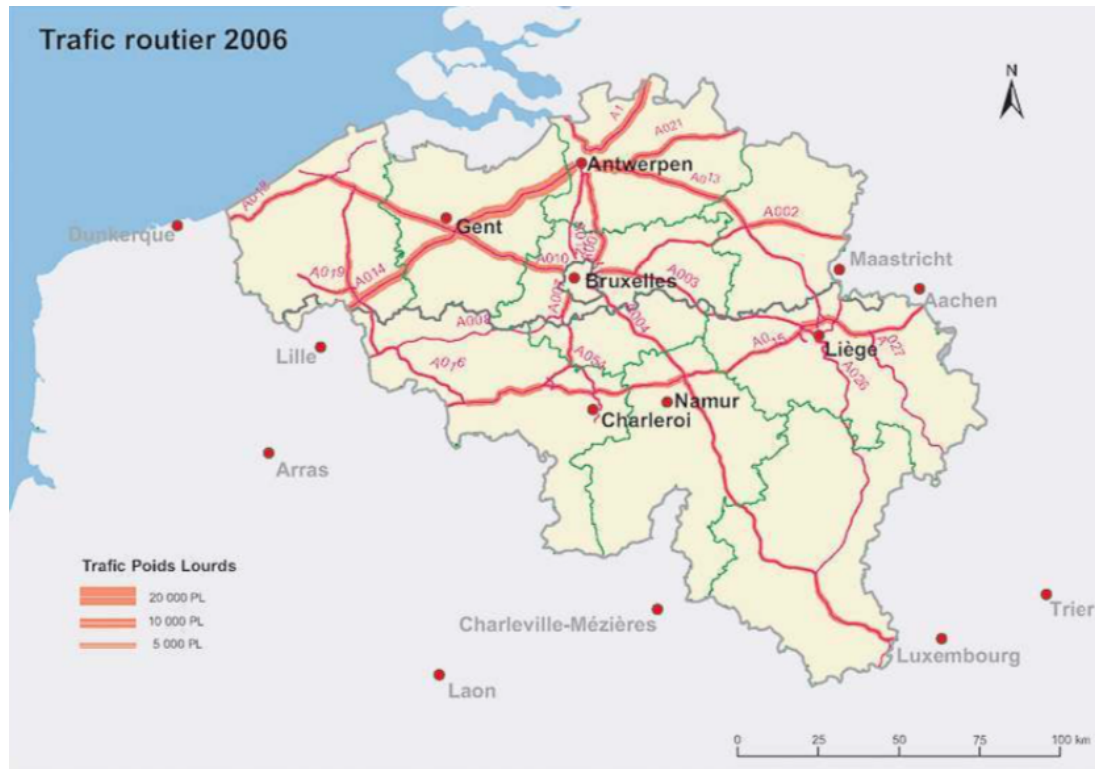


FIGURE 5.11 – Axes des flux de poids lourds traversant Belgique

Le nombre associé à chaque axe correspond au nombre de camions qui emprunte cet axe par jour et par sens de circulation.

Une vérification a été effectuée dans le modèle et il y a 8% de camions sur les nationales et 12,5% sur les autoroutes. Ces valeurs sont suffisamment proches de la réalité pour être acceptés. Enfin, la distribution horaire des départs des camions est supposée la même que celle des voitures.

Chapitre 6

Modélisation et analyse des résultats

6.1 Explicatif du code Matlab

Cette section a pour but d'expliquer le fonctionnement général du code Matlab, sans entrer dans les détails. Une explication détaillée du code se situe dans l'annexe D.

Le code se décompose en quatre parties :

- la détermination des trajets ;
- la simulation sans fermeture ;
- la simulation de la fermeture de la route ;
- l'analyse des résultats.

Le code va tout d'abord lire les fichiers textes en entrée, à savoir les coordonnées géographiques de noeuds (matrice `xyv` - annexe C), le nombre de bandes sur chacun des tronçons et la vitesse maximale autorisée. Ensuite il va construire les matrices utiles, par exemple la matrice des distances entre les villes ou la matrice qui reprends toute les informations des différents tronçons.

Ensuite, Matlab va faire appel à la fonction `liste_trajet.m` qui va déterminer une ville de départ, une ville d'arrivée et une heure de départ pour toutes les voitures et tous les camions du modèle.

La simulation va ensuite faire appel à la fonction `simulation_temporelle.m`. Cette fonction calcule, à chaque pas de temps, la position des véhicules sur le réseau, et détermine une réduction de vitesse si le tronçon est encombré. La fonction renvoie l'heure d'arrivée des véhicules ainsi que la vitesse moyenne à chaque pas de temps et sur chaque tronçon.

Après cela, le code va appeler la même fonction `simulation_temporelle.m` mais en ayant pris soin de fermer le ou les tronçons choisis. La simulation se fait avec les mêmes véhicules que pour le cas sans fermeture de tronçon. Comme pour le point précédent, elle va renvoyer l'heure d'arrivée des véhicules, ainsi que la vitesse moyenne.

Enfin, le code va finir sur une analyse de la situation avant et après la fermeture du tronçon. Bien que la simulation accepte la fermeture de plusieurs tronçons, l'analyse s'effectue pour la fermeture d'un seul tronçon. L'analyse compare l'augmentation du temps de trajet, de la distance et de l'émission de CO_2 pour les véhicules impactés par la fermeture du tronçon.

6.1.1 Impact du CO_2

L'environnement est désormais un facteur important dans le monde actuel. Il est donc intéressant de regarder les émissions de CO_2 dans ce modèle.

L'émission de CO_2 dans l'air dépend de plusieurs paramètres, à commencer par la marque et l'année de la voiture. Un véhicule ancien pollue plus qu'un véhicule neuf, les voitures neuves étant soumises à plus de normes environnementales. L'émission dépend aussi de la vitesse des véhicules.

Dans ce travail, l'émission de CO_2 est comptée uniquement en fonction de la distance. Elle est estimée à 130 grammes par kilomètre pour une voiture et 1 kg par kilomètre pour un camion. Ces nombres correspondent à une moyenne du parc automobile belge.

6.2 Analyse du comportement général

Une première analyse peut se faire lorsqu'aucun tronçon n'est fermé.

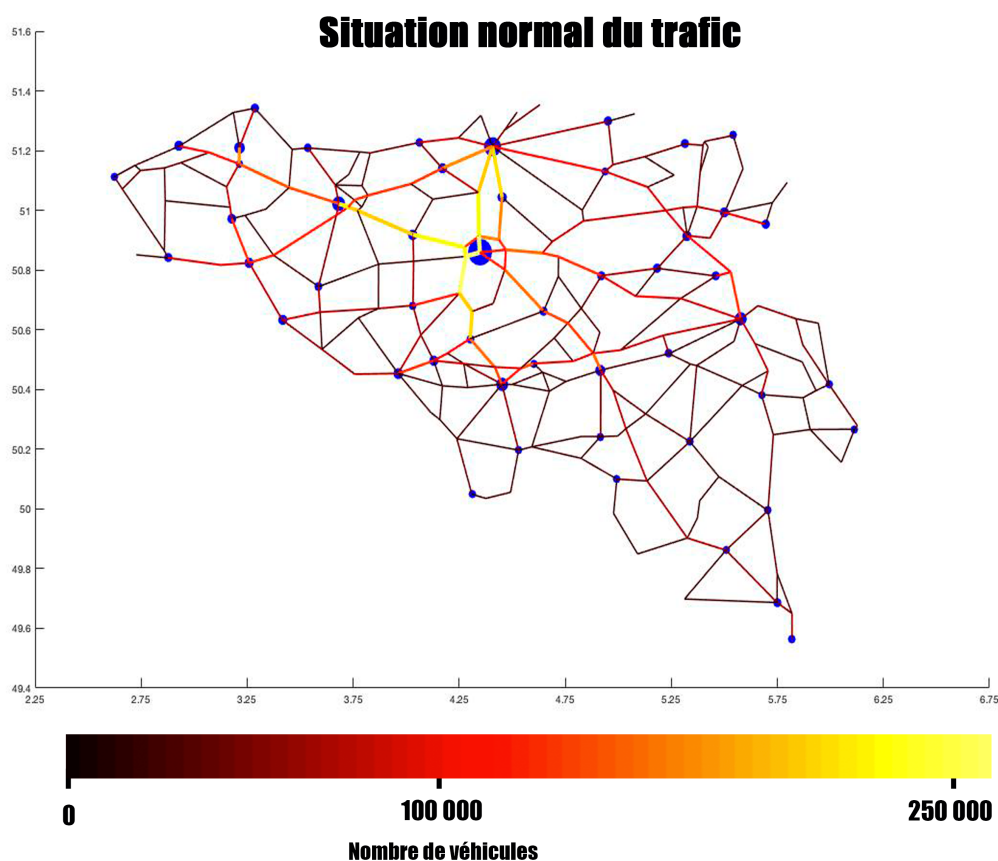


FIGURE 6.1 – Nombre de véhicules sur chaque tronçon

La figure 6.1 montre le nombre de véhicules (voitures et camions) qui ont circulé pendant toute une journée sur chaque tronçon. Plus le tronçon est représenté en jaune, plus il a accueilli de véhicules. Si le tronçon est représenté en noir, il a reçu très peu de véhicules. Sans surprise, le tronçon qui accueille

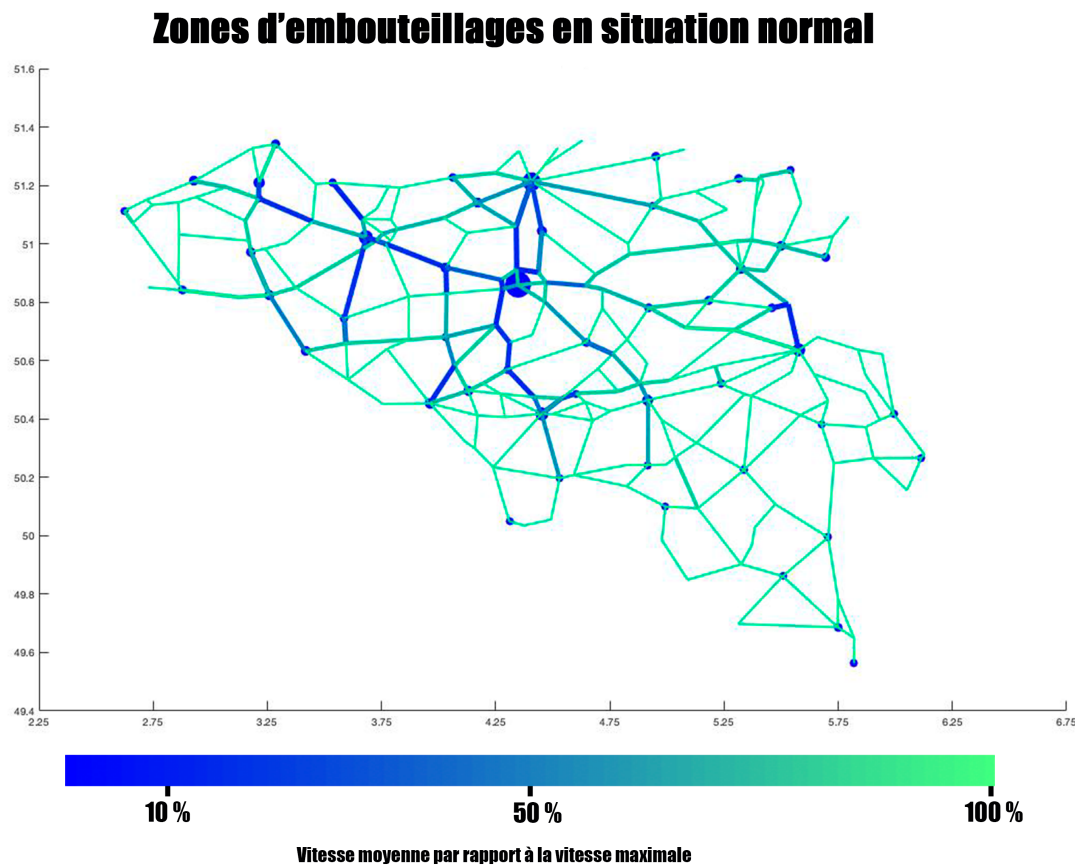


FIGURE 6.2 – Zones d'embouteillage sur chaque tronçon

le plus de véhicules est un de ceux qui rentre dans Bruxelles. Il reçoit chaque jour plus de 258000 véhicules. Il y a également quelques tronçons inutilisés. Comme il n'y a que 50 villes hôtes, certaines routes ne seront jamais utilisées dans le modèle si elles sont excentrées ou s'il y a une route plus rapide à côté. Néanmoins, elles seront utilisées par le modèle en cas de fermeture de certains tronçons.

La durée du trajet moyen sans prendre en compte les bouchons est de 39 minutes. La durée réelle pour les véhicules ayant terminé leur parcours à la fin de la simulation est de 1h49. Le parcours moyen des véhicules est de 69km.

La figure 6.2 représente les embouteillages sur le réseau. Un tronçon vert indique qu'il n'y a eu aucun ralentissement sur la journée et un tronçon représenté en bleu indique que le tronçon a été bloqué durant une très grosse partie de la journée.

Une animation concernant les embouteillages en cours de journée est disponible en scannant le QR code suivant, ou via le lien suivant : https://youtu.be/XnpNYrvT-_g



FIGURE 6.3 – QR code

6.3 Analyse suite à la réduction du nombre de bandes de circulation

Cette section a pour but d'analyser la vitesse des véhicules suite à la réduction du nombre de bandes pour le tronçon 78 entre De Pinte (26) et Gand (28). Ces numéros font référence au numéro de la ville dans le modèle. Pour rappel, toutes les villes ainsi que leurs numéros sont référencées dans l'annexe C.

L'analyse suivante propose de mettre en évidence la vitesse moyenne sur un tronçon en période normale, puis lorsqu'une puis deux bandes sont fermées. Ces fermetures peuvent être le résultat de travaux sur les voies ou d'un accident. Le graphe suivant montre la vitesse des voitures, mais les camions subissent la même diminution de vitesse. Enfin, cette vitesse est calculée en faisant la moyenne sur chaque sous-tronçon de la vitesse des véhicules. ce n'est donc pas la moyenne de tous les véhicules.

Dans l'état d'utilisation normale, le tronçon connaît de légers ralentissements aux heures de pointe, comme représenté sur le premier graphe de la figure 6.4. Lorsque le tronçon passe de trois à deux bandes, les ralentissements sont nettement plus marqués. La vitesse des véhicules descend sous les 80 km/h aux heures de pointe. Ce résultat est logique car, s'il y a une bande de moins, la densité des véhicules augmente et il y a donc plus de risque de congestion. Enfin, lorsqu'il ne reste plus qu'une seule bande, il y a des bouchons en permanence sur le réseau, et les véhicules roulent à la vitesse minimale.

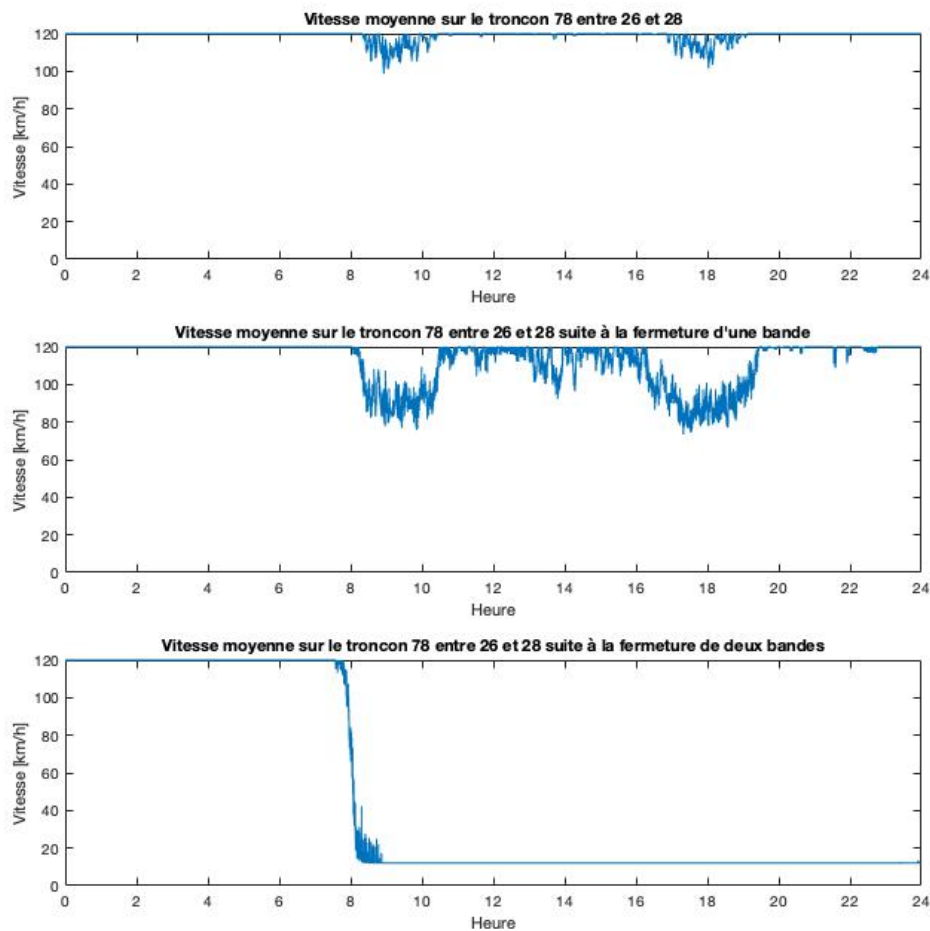


FIGURE 6.4 – Vitesse moyenne sur le tronçon

6.4 Analyse de la fermeture d'un tronçon

La section suivante propose d'analyser le comportement des véhicules en cas de fermeture d'un tronçon. Dans les graphes suivants, seules les véhicules impactés sont représentés. La route fermée est dessinée en vert et le nombre de véhicules qui passe sur un tronçon est écrit en bleu à côté du tronçon.

6.4.1 Fermeture de la E42 entre Mons et La Louvière

La première analyse se fera avec la fermeture du tronçon reliant les villes de La Louvière (81) et Mons (82). Le tronçon dans le sens La Louvière - Mons porte le numéro 267 et le tronçon dans le sens Mons - Louvière porte le numéro 272.

Le tableau 6.1 reprend les différents résultats obtenus. Le temps moyen a été calculé uniquement avec les véhicules ayant terminé leur trajet à la fin de la

simulation (à minuit) tandis que la distance moyenne a été calculé sur tout les véhicules.

Nombre de voitures impactées	49 345
Nombre de camions impactés	8 229
Temps moyen sans la fermeture [min]	60,8
Temps supplémentaire moyen [min]	148,4
Distance moyenne sans la fermeture [km]	94,2
Distance supplémentaire moyenne [km]	8,5
Emissions de CO ₂ [Tonne]	3 904
Emissions de CO ₂ additionnelles [Tonne]	202,5

TABLE 6.1 – Fermeture du tronçon entre La Louvière et Mons (E42)

Comme l’assignation des trajets impose aux véhicules un aller et un retour suivant le même parcours, le nombre de véhicules sur la journée dans les deux sens de circulation est exactement le même. Dans les graphes qui vont suivre, la valeur donnée aux arrêtes équivaut au nombre de véhicules impactés (voitures et camions) passés dans un sens de circulation. Un véhicule impacté par la fermeture est un véhicule qui passe en temps normal par le tronçon fermé.

La figure 6.5 montre la situation du trafic avant la fermeture. Au plus une route est représentée par la couleur jaune, au plus les véhicules impactés passent par cette route. Si la route est représentée en noir, aucun véhicule impacté ne passe par cette route. La figure 6.6 représente le trafic après la fermeture du tronçon. Comme attendu, il y a une redistribution importante du trafic. Les figures 6.7 et 6.8 montre la redistribution au niveau local. Dans ce cas, la redistribution des véhicules est assez importante sur le réseau.

Il y a chaque jour 57 574 véhicules passant par ce tronçon (une fois dans chaque sens) dont 29 930 qui font uniquement le trajet entre ces deux villes. La plupart des véhicules impactés passe par le nord (Soignies). Les autres prennent un autre chemin, parfois très éloigné du chemin initial. Par exemple, les véhicules qui partent de Bruges passe par le ring de Bruxelles. A noter que le ring de Bruxelles est encombré durant une grande partie de la journée.

La hausse importante du temps de parcours s’explique par la création d’embouteillages sur les deux tronçons représentés en jaune sur les figures suivantes, ainsi que le passage par le ring de Bruxelles. La figure 6.2 montre bien que prendre un chemin par Bruxelles augmente considérablement le temps de trajet, car même les routes conduisant à Bruxelles sont ralenties.

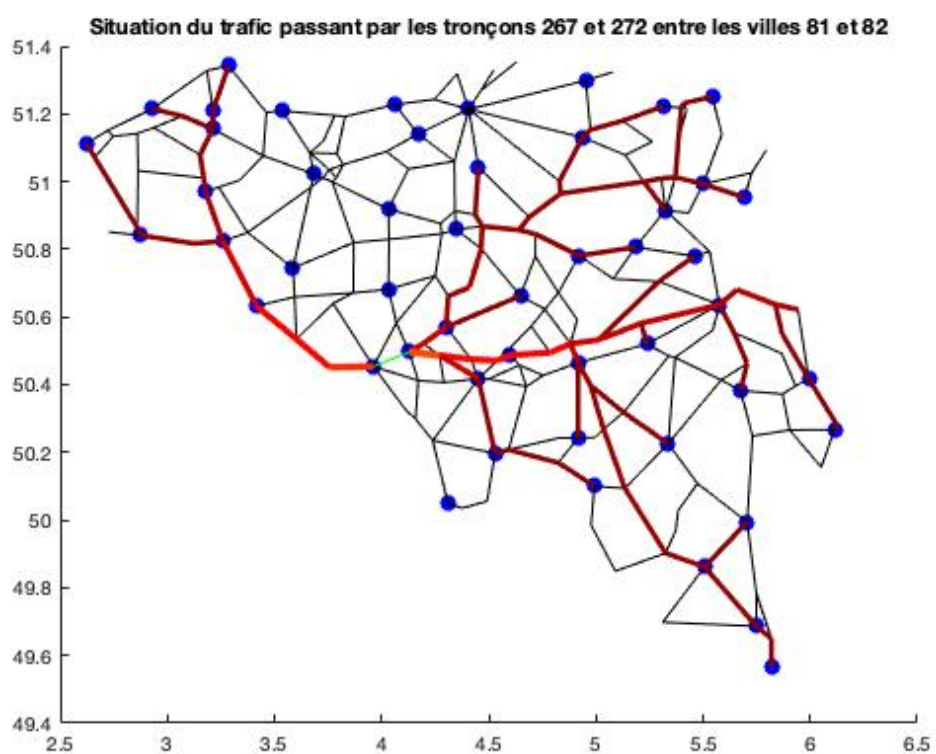


FIGURE 6.5 – Vue d’ensemble sur la Belgique en tant normal.

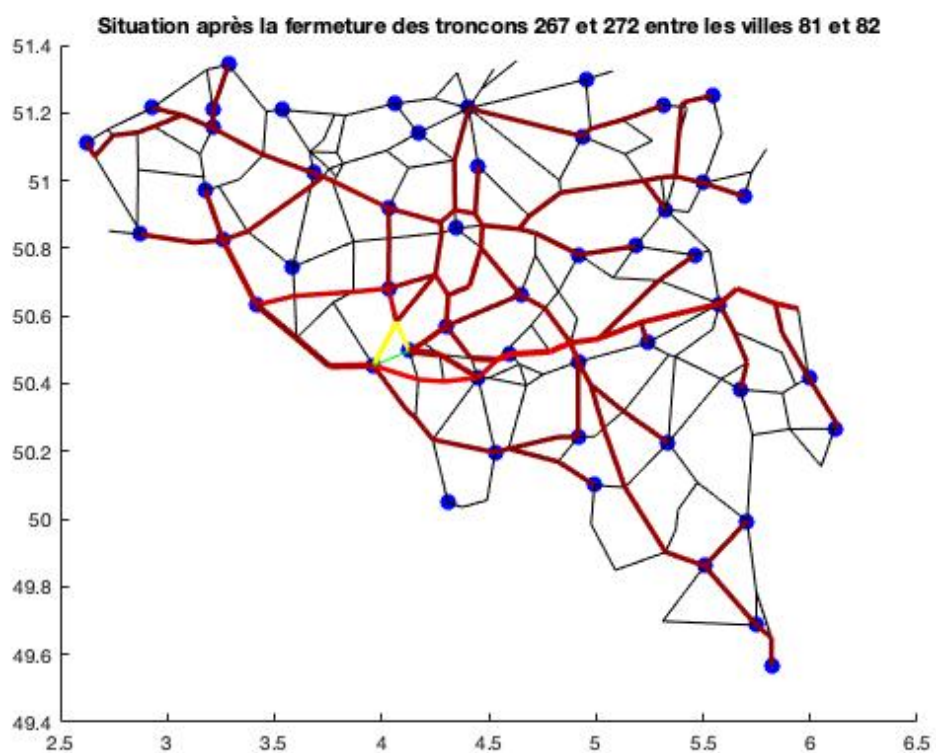


FIGURE 6.6 – Vue d’ensemble sur la Belgique après la fermeture du tronçon.

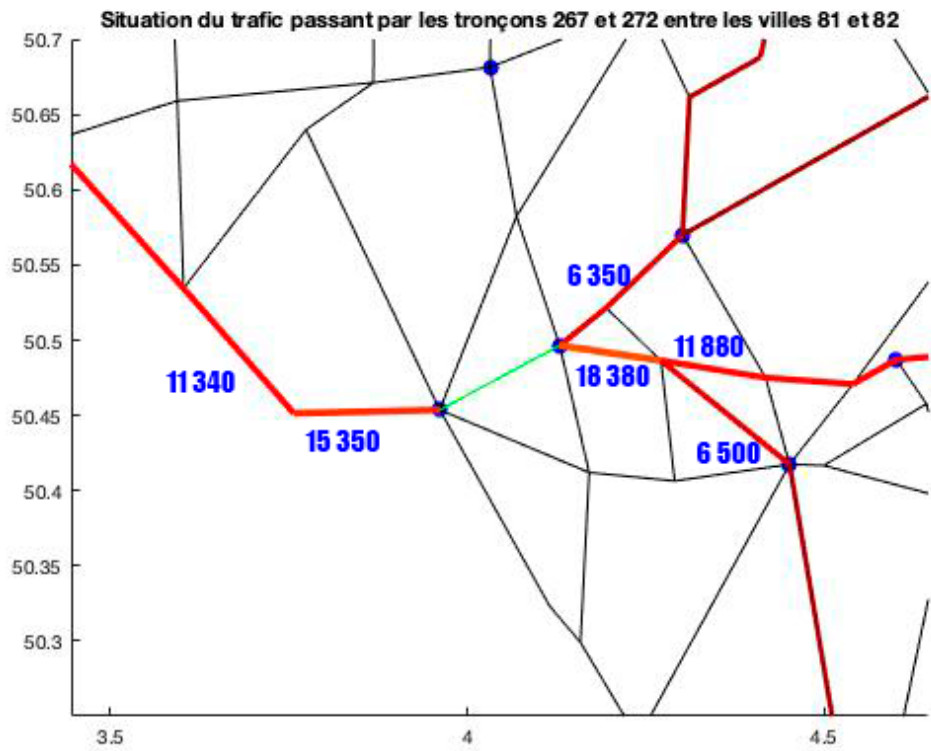


FIGURE 6.7 – Zoom sur la situation locale en tant normal.

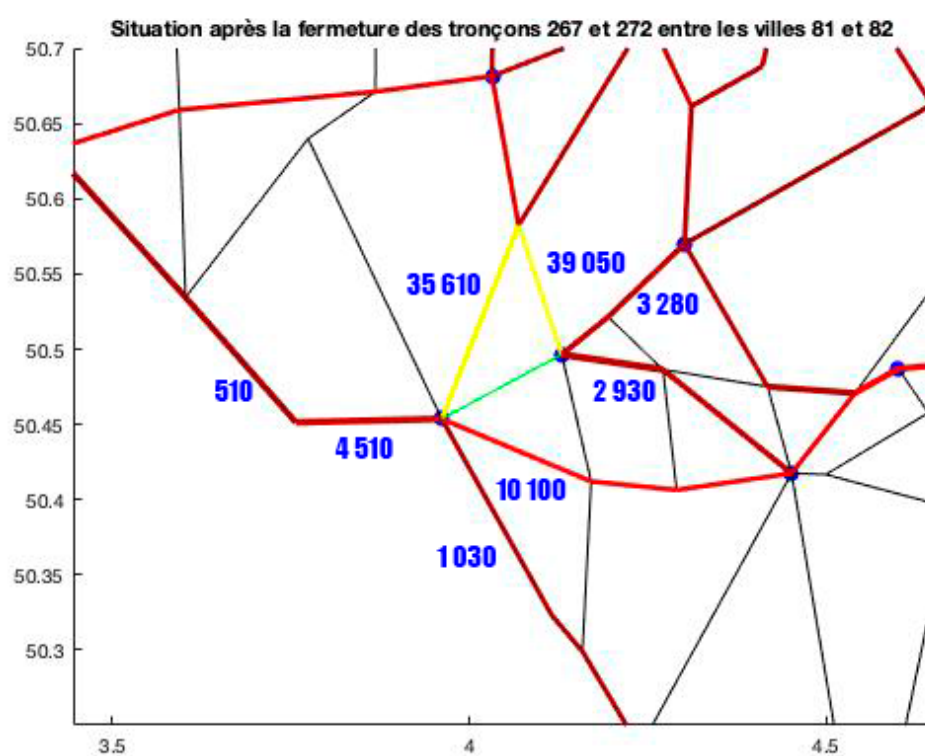


FIGURE 6.8 – Zoom sur la situation locale après la fermeture du tronçons.

6.4.2 Fermeture de la E42 entre La Louvière et Chapelle-lez-Herlaimont

L'analyse suivante porte sur la fermeture du tronçon reliant les villes de La Louvière (81) et Chapelle-lez-Herlaimont (88).

Nombre de voitures impactées	23 450
Nombre de camions impactés	6 020
Temps moyen sans la fermeture [min]	108,6
Temps supplémentaire moyen [min]	134,6
Distance moyenne sans la fermeture [km]	154,3
Distance supplémentaire moyenne [km]	1,8
Emissions de CO ₂ [Tonne]	3 552
Emissions de CO ₂ additionnelles [Tonne]	29,9

TABLE 6.2 – Fermeture du tronçon entre La Louvière et Chapelle-lez-Herlaimont (E42)

La distance supplémentaire à parcourir est très faible, car le passage par Ecaussinnes (80) est très court. Néanmoins, le temps de parcours est nettement plus long, car la déviation fait prendre une route nationale qui est limitée à 90 km/h et des bouchons se forment. La figure 6.13 montre la vitesse moyenne sur ce tronçon (celui en jaune) en temps normale, tandis que la figure 6.14 montre la vitesse moyenne après la fermeture. De plus, la déviation est une nationale constituée de deux bandes, contre trois pour la route fermée

Il y a 29 470 véhicules impactés (voitures et camions), 28 120 modifient leur trajet en passant par Ecaussinnes, et les autres véhicules, soit 1 370 passent par le ring de Bruxelles.

Dans cette situation, la redistribution des véhicules est très locale. Seulement deux déviations sont utilisées et ne s'éloignent pas beaucoup du chemin initial.

Dans ce cas, le calcul des émissions de CO₂ peut être faussé par le fait que le calcul ne tient pas compte de la vitesse. Un tronçon embouteillé émet plus de pollution qu'un tronçon dégagé.

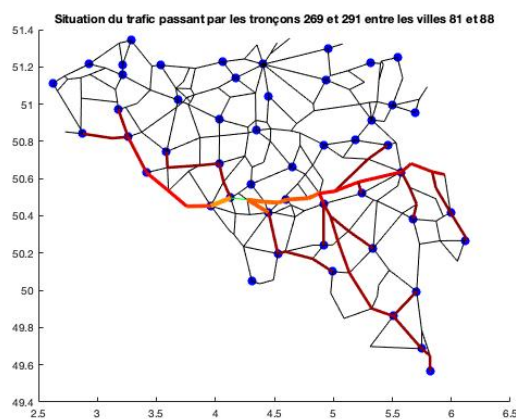


FIGURE 6.9 – Vue d’ensemble sur la Belgique

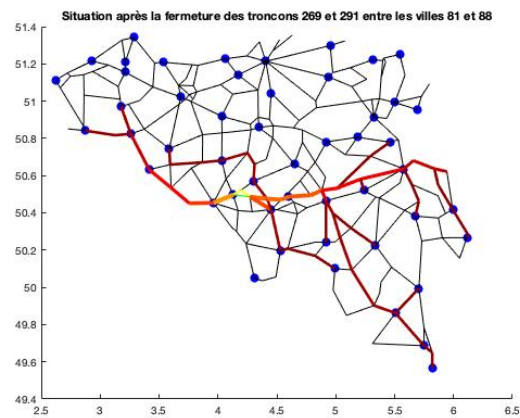


FIGURE 6.10 – Vue d’ensemble sur la Belgique après la fermeture

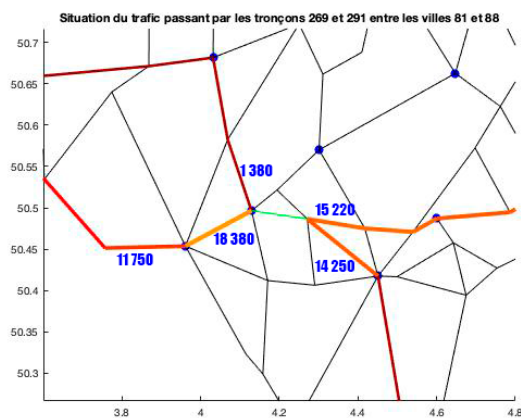


FIGURE 6.11 – Zoom sur la situation locale

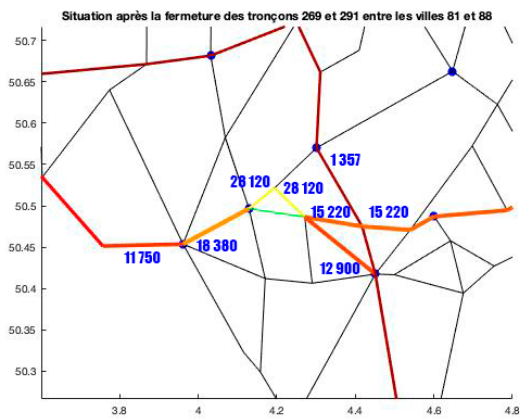


FIGURE 6.12 – Zoom sur la situation locale après la fermeture

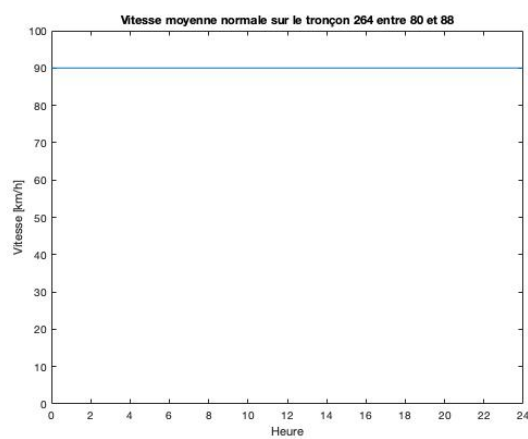


FIGURE 6.13 – Vitesse moyenne entre Ecaussinnes (80) et Chapelle-lez-Herlaimont (88) avant la fermeture

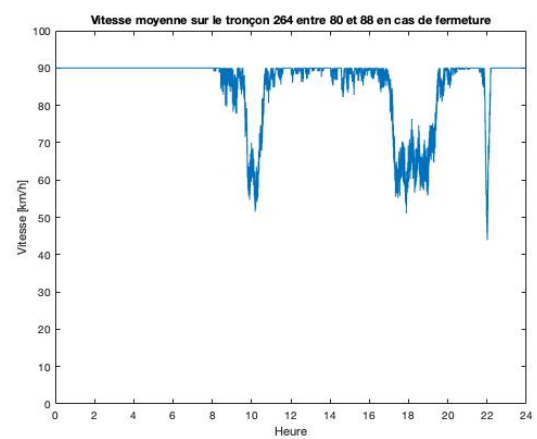


FIGURE 6.14 – Vitesse moyenne entre Ecaussinnes (80) et Chapelle-lez-Herlaimont (88) après la fermeture

6.4.3 Fermeture de la E42 entre Chapelle-lez-Herlaimont et Gosselies

L'analyse porte sur la fermeture du tronçon reliant les villes de Chapelle-lez-Herlaimont (88) et Gosselies (89).

Nombre de voitures impactées	10 347
Nombre de camions impactés	4 874
Temps moyen sans la fermeture [min]	161,1
Temps supplémentaire moyen [min]	50,1
Distance moyenne sans la fermeture [km]	220,3
Distance supplémentaire moyenne [km]	3,3
Emissions de CO ₂ [Tonne]	3 075
Emissions de CO ₂ additionnelles [Tonne]	40,9

TABLE 6.3 – Fermeture du tronçon entre Chapelle-lez-Herlaimont et Gosselies (E42)

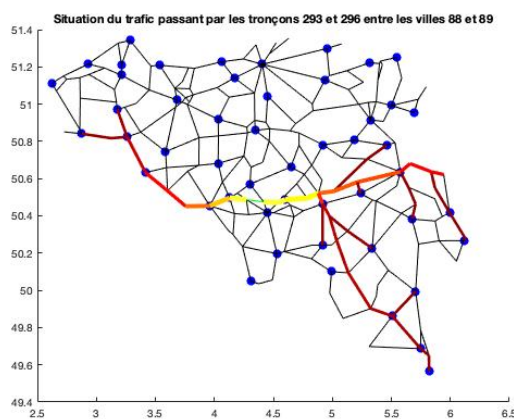


FIGURE 6.15 – Vue d'ensemble sur la Belgique

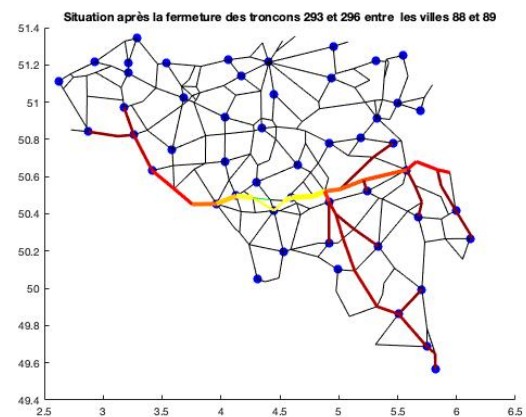


FIGURE 6.16 – Vue d'ensemble sur la Belgique après la fermeture

Comme pour le point précédent, la distance supplémentaire à parcourir n'est pas énorme par rapport au temps supplémentaire, suite au problème de congestion lié à l'augmentation du trafic et au nombre de bande plus petit dans la déviation que dans la route fermée. Il n'y a qu'une seule déviation et la redistribution est uniquement locale. Dans ce cas, la totalité des parcours est déviée par Charleroi.

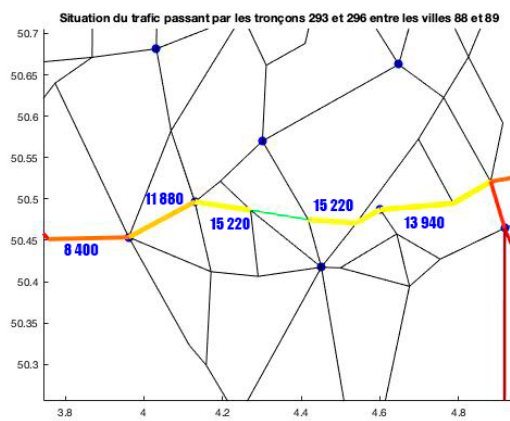


FIGURE 6.17 – Zoom sur la situation locale

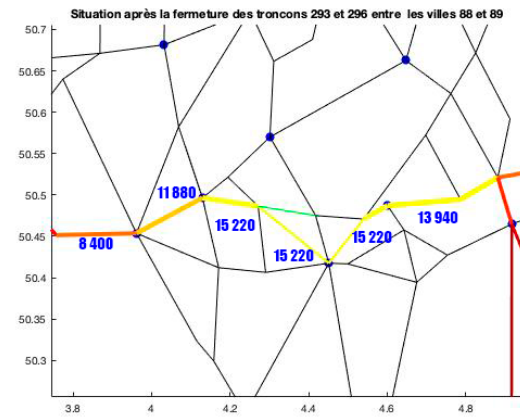


FIGURE 6.18 – Zoom sur la situation locale après la fermeture

6.4.4 Fermeture de la E42 entre Gosselies et Fleurus

L'analyse porte sur la fermeture du tronçon reliant les villes de Gosselies (89) et Fleurus (91).

Nombre de voitures impactées	23 114
Nombre de camions impactés	5 998
Temps moyen sans la fermeture [min]	196,2
Temps supplémentaire moyen [min]	57
Distance moyenne sans la fermeture [km]	168,6
Distance supplémentaire moyenne [km]	4,3
Emissions de CO ₂ [Tonne]	3 695
Emissions de CO ₂ additionnelles [Tonne]	71,2

TABLE 6.4 – Fermeture du tronçon entre Gosselies et Fleurus (E42)

La redistribution des véhicules sur le réseau semble plus marquée que pour les deux points précédents. Environ 1 500 véhicules prennent une déviation éloignée de la route fermée.

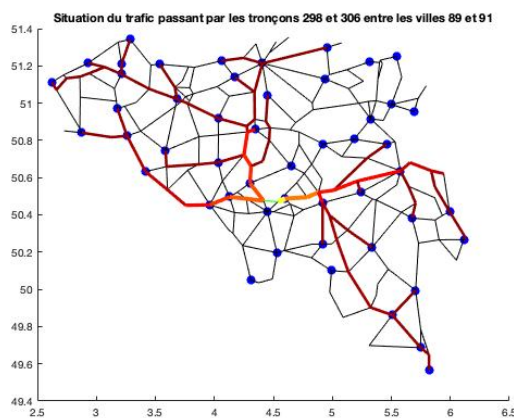


FIGURE 6.19 – Vue d'ensemble sur la Belgique

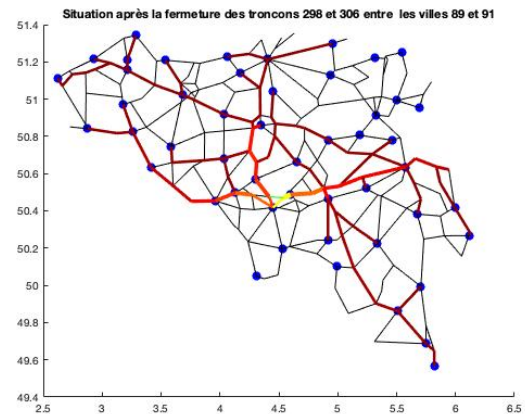


FIGURE 6.20 – Vue d'ensemble sur la Belgique après la fermeture

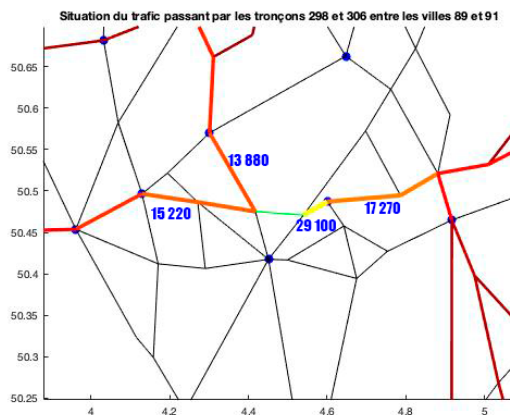


FIGURE 6.21 – Zoom sur la situation locale

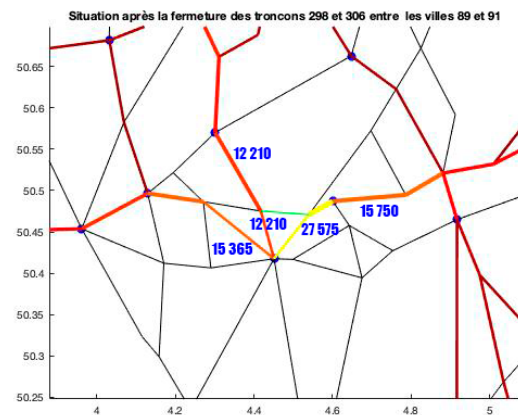


FIGURE 6.22 – Zoom sur la situation locale après la fermeture

6.4.5 Fermeture de la E42 entre Fleurus et Sombreffe

L'analyse porte sur la fermeture du tronçon reliant les villes de Fleurus (91) et Sombreffe (92).

Assez étonnamment, 36% des véhicules impactés ont un temps de trajet plus court en cas de fermeture. Ce taux était de 4% pour la fermeture entre Gosselies et Fleurus (voir 6.4.4), et était inférieur à 1% pour les trois premières analyses. Cette diminution de temps de trajet s'explique par le nombre plus élevé de routes que prennent les usagers, et donc leur nombre est plus réparti sur le réseau. Une meilleure répartition permet de diminuer les bouchons sur les tronçons. De plus, le trajet de certains usagers comprenait des bouchons, et le changement de parcours fait prendre des route moins encombrées. Par exemple, le tronçon reliant les villes 90 et 91 (en jaune juste à gauche du tronçon fermé) est bouché durant toute la journée dans le modèle. Suite à la fermeture,

Nombre de voitures impactées	63 416
Nombre de camions impactés	9 500
Temps moyen sans la fermeture [min]	119,1
Temps supplémentaire moyen [min]	22
Distance moyenne sans la fermeture [km]	93,05
Distance supplémentaire moyenne [km]	8,6
Emissions de CO ₂ [Tonne]	4 448
Emissions de CO ₂ additionnelles [Tonne]	295,3

TABLE 6.5 – Fermeture du tronçon entre Fleurus et Sombreffe (E42)

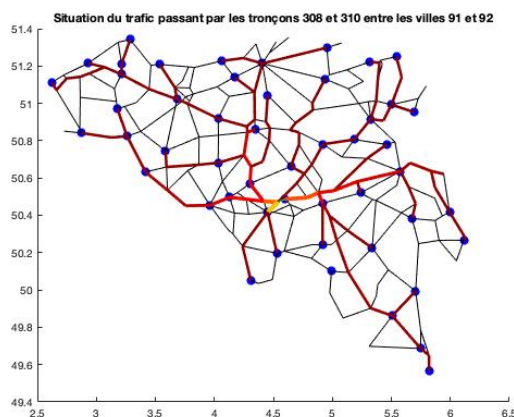


FIGURE 6.23 – Vue d'ensemble sur la Belgique

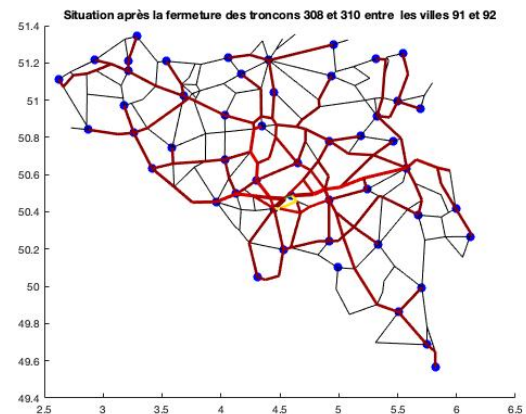


FIGURE 6.24 – Vue d'ensemble sur la Belgique après la fermeture

il y a 42 000 véhicules en moins qui y passe par jour et il n'y a plus aucun bouchon. Cela montre bien que la détermination des trajets ne prend pas en compte les prévisions de congestion. Les figures 6.23, 6.24, 6.25 et 6.26 montrent clairement que beaucoup plus de tronçons sont empruntés suite à la fermeture, tant au niveau local que global.

Il est également intéressant de remarquer qu'au niveau de la Belgique, il y a 33% des usagers qui ont un temps de parcours inférieur à la situation normale. Les véhicules impactés représentent 2% des véhicules simulés. Il y a dans ce modèle des tronçons critiques, qui, de par leur position, incitent le passage pour de nombreux usagers. Cela a pour conséquence des tronçons surchargés, et d'autre vides. Si les tronçons critiques sont fermés, cela réorganise l'ensemble des trajets, et aboutit à une meilleure répartition des véhicules. Dans la réalité, les automobilistes tiennent compte des prévisions de bouchons, et donc fermer une route n'entraînera jamais une diminution du temps de parcours pour un nombre aussi important d'usagers.

Pour finir, sur l'ensemble de la Belgique, il y a 29% des usagers qui ont un temps de parcours strictement identique lors qu'il y a la fermeture ou non.

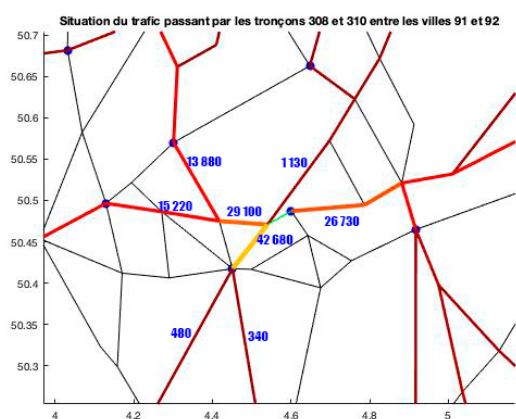


FIGURE 6.25 – Zoom sur la situation locale

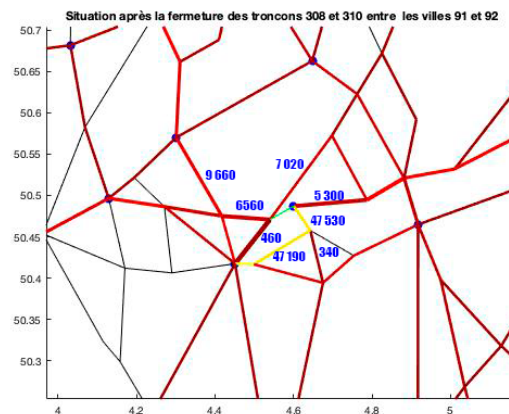


FIGURE 6.26 – Zoom sur la situation locale après la fermeture

Ceux là ne subissent aucune influence lors de la fermeture du tronçon. A l'inverse, cela revient à dire que 71% des usagers ressentent les effets, même très légèrement, de la fermeture d'une route empruntée par seulement 2% du trafic.

6.4.6 Fermeture de la E420 entre Nivelles et Gosselies

L'analyse porte sur la fermeture du tronçon reliant les villes de Nivelles (79) et Gosselies (89).

Nombre de voitures impactées	75 975
Nombre de camions impactés	6 719
Temps moyen sans la fermeture [min]	218,8
Temps supplémentaire moyen [min]	45,7
Distance moyenne sans la fermeture [km]	93,1
Distance supplémentaire moyenne [km]	16,4
Emissions de CO ₂ [Tonne]	3 088
Emissions de CO ₂ additionnelles [Tonne]	543,6

TABLE 6.6 – Fermeture du tronçon entre Nivelles et Gosselies (E420)

Dans cette situation, la distance supplémentaire à parcourir et le nombre de véhicules impactés est très important. Le tronçon qui est fermé est situé sur l'axe Bruxelles - Charleroi. Bien que la plupart des usagers décide de passer par Ecaussinnes (80), les redistributions des autres véhicules restent importantes sur toute la Belgique.

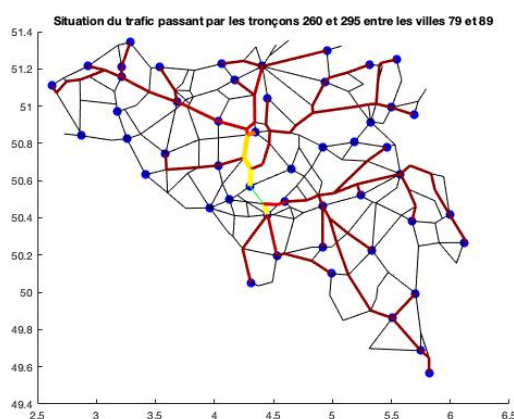


FIGURE 6.27 – Vue d'ensemble sur la Belgique

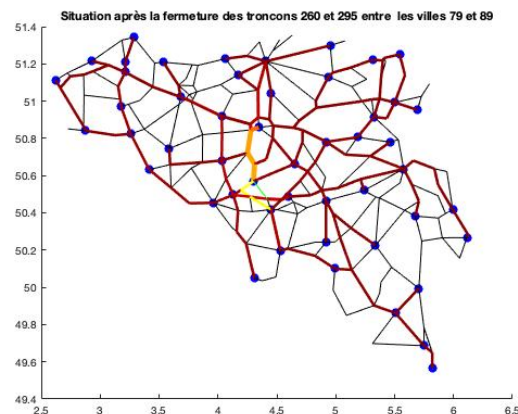


FIGURE 6.28 – Vue d'ensemble sur la Belgique après la fermeture

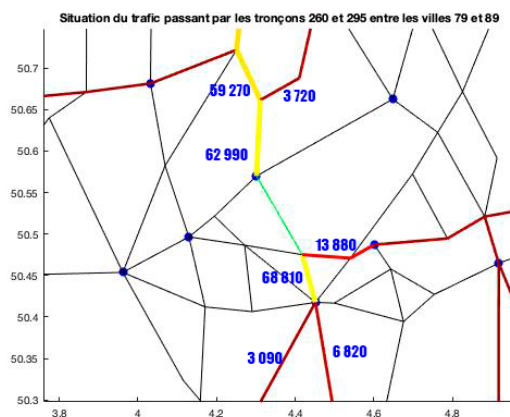


FIGURE 6.29 – Zoom sur la situation locale

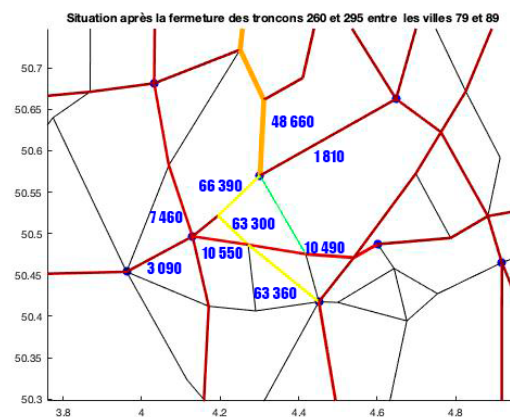


FIGURE 6.30 – Zoom sur la situation locale après la fermeture

La distance supplémentaire à effectuer est assez importante à cause du manque d'alternative rapide à ce tronçon. Le détour par Ecaussinnes est long.

À titre de comparaison, si ce tronçon passe de deux bandes à une seule, la durée supplémentaire est de 19 minutes en moyenne pour les usagers.

6.4.7 Fermeture de la E411 entre Perwez et Louvain-la-Neuve

L'analyse porte sur la fermeture du tronçon reliant les villes de Perwez (94) et Louvain-la-Neuve (95).

Ce tronçon se situe sur l'axe Bruxelles - Namur, et donc beaucoup de véhicules sont impactés. Ce tronçon est fréquemment encombré aux heures de pointe. Ce n'est donc pas une surprise de voir le temps de parcours augmenter

Nombre de voitures impactées	58 853
Nombre de camions impactés	9 148
Temps moyen sans la fermeture [min]	132,3
Temps supplémentaire moyen [min]	187,4
Distance moyenne sans la fermeture [km]	151,6
Distance supplémentaire moyenne [km]	25,5
Emissions de CO ₂ [Tonne]	6 616
Emissions de CO ₂ additionnelles [Tonne]	792,9

TABLE 6.7 – Fermeture du tronçon entre Perwez et Louvain-la-Neuve (E411)

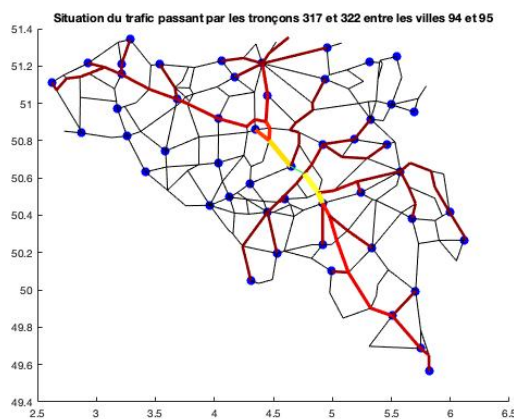


FIGURE 6.31 – Vue d'ensemble sur la Belgique

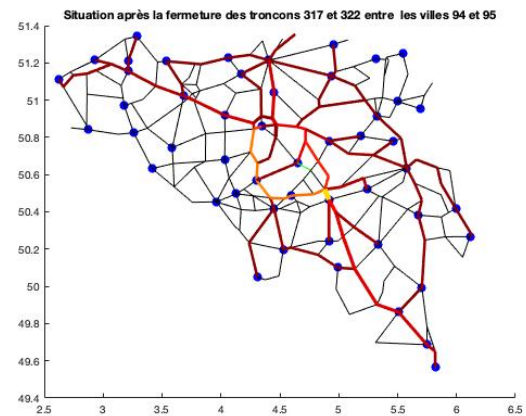


FIGURE 6.32 – Vue d'ensemble sur la Belgique après la fermeture

autant. Si le trafic forme des embouteillages sur le tronçon fermé, ils formeront aussi des embouteillages sur les déviations qui comportent un nombre moins important de bandes. De plus, ils devront se rajouter aux usagers qui utilisent la déviation en temps normal.

La distance à parcourir est également importante. De toutes les déviations, aucune ne passe par Perwez. La E411 est complètement désertée entre Bruxelles (76) et Warisoulx (126) (dernière sortie avant Namur).

Cela encombre bien entendu largement plus les autres tronçons du ring de Bruxelles, et la redistribution est très importante.

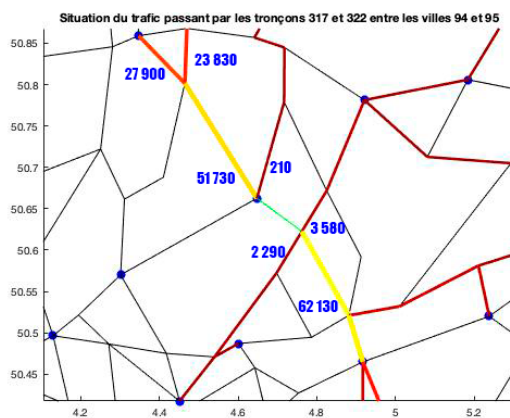


FIGURE 6.33 – Zoom sur la situation locale

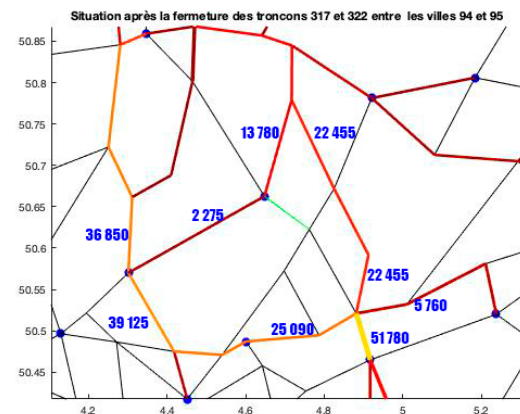


FIGURE 6.34 – Zoom sur la situation locale après la fermeture

6.4.8 Fermeture de la E411 entre Wellin et Ciney

L'analyse porte sur la fermeture du tronçon reliant les villes de Wellin (156) et Ciney (157).

Nombre de voitures impactées	17 604
Nombre de camions impactés	5 571
Temps moyen sans la fermeture [min]	191,3
Temps supplémentaire moyen [min]	25,8
Distance moyenne sans la fermeture [km]	253,8
Distance supplémentaire moyenne [km]	6,2
Emissions de CO ₂ [Tonne]	4 844
Emissions de CO ₂ supplémentaires [Tonne]	179,2

TABLE 6.8 – Fermeture du tronçon entre Wellin et Ciney (E411)

Comme pour l'analyse 5 entre Fleurus et Sombreffe, il y a un certain nombre d'utilisateurs qui ont un temps de trajet inférieur (14%) suite à la fermeture. Cela s'explique par le fait que l'axe Bruxelles - Namur - Arlon est moins fréquenté. Et des routes secondaires non embouteillées accueillent une partie de ce trafic.

La redistribution est très importante sur l'ensemble de la Belgique. Comme les figures suivantes le montre, aucune route n'accueille autant de trafic que le tronçon fermé en temps normal.

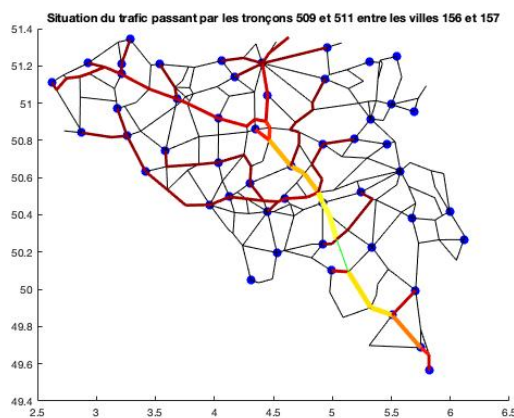


FIGURE 6.35 – Vue d'ensemble sur la Belgique

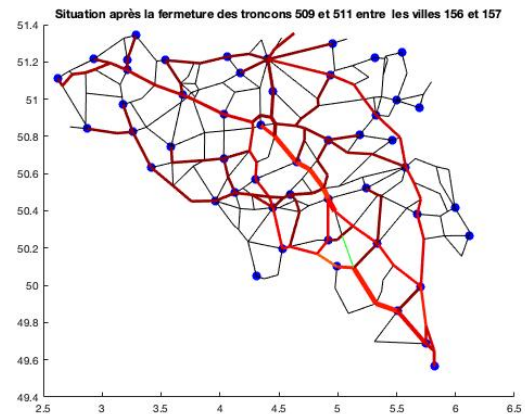


FIGURE 6.36 – Vue d'ensemble sur la Belgique après la fermeture

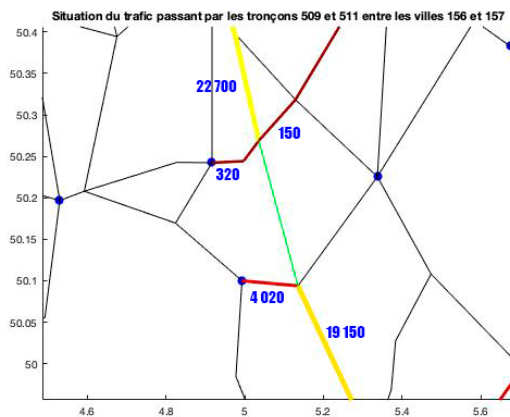


FIGURE 6.37 – Zoom sur la situation locale

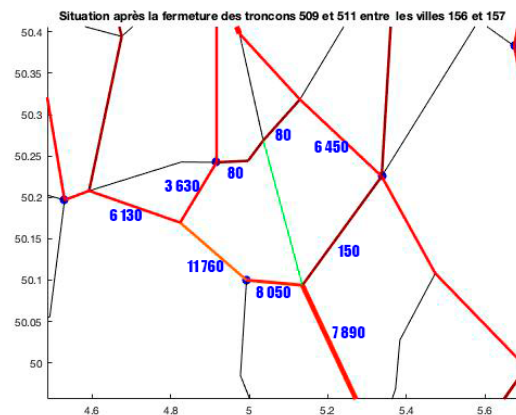


FIGURE 6.38 – Zoom sur la situation locale après la fermeture

6.4.9 Fermeture de la E17 entre Gand et Lokeren

L'analyse porte sur la fermeture du tronçon reliant les villes de Gand (33) et Lokeren (58).

Il est intéressant de remarquer qu'aucun véhicule impacté ne passe par la Wallonie. De par sa position, aucun véhicule ne devant rejoindre la Wallonie n'a intérêt à passer par là.

Le temps de trajet est important car la majeure partie des véhicules passe par le nord (autoroute E34), et crée des embouteillages car cette route n'a pas été conçue pour accueillir autant d'automobilistes. Elle ne fait que deux bandes contre trois pour le tronçon fermé.

Nombre de voitures impactées	46 026
Nombre de camions impactés	13 972
Temps moyen sans la fermeture [min]	132,3
Temps supplémentaire moyen [min]	190,4
Distance moyenne sans la fermeture [km]	165,9
Distance supplémentaire moyenne [km]	13,1
Emissions de CO ₂ [Tonne]	7 627
Emissions de CO ₂ supplémentaires [Tonne]	570,9

TABLE 6.9 – Fermeture du tronçon entre Gand et Lokeren (E17)

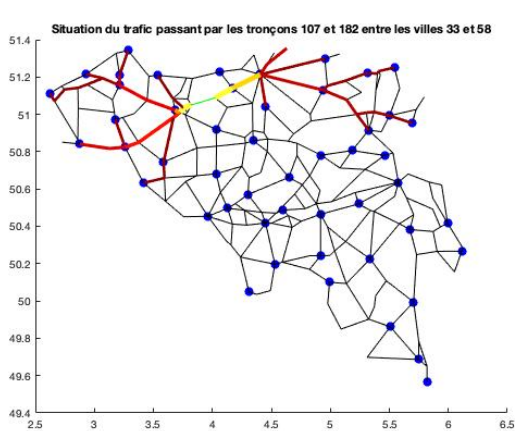


FIGURE 6.39 – Vue d'ensemble sur la Belgique

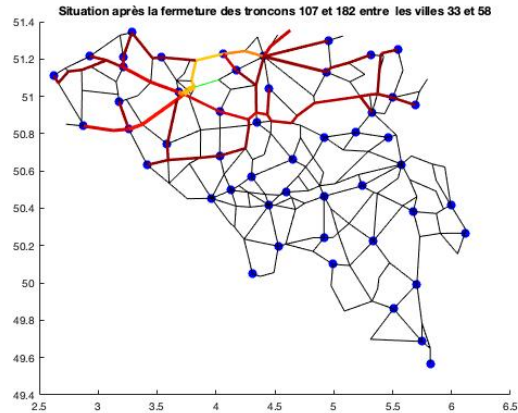


FIGURE 6.40 – Vue d'ensemble sur la Belgique après la fermeture

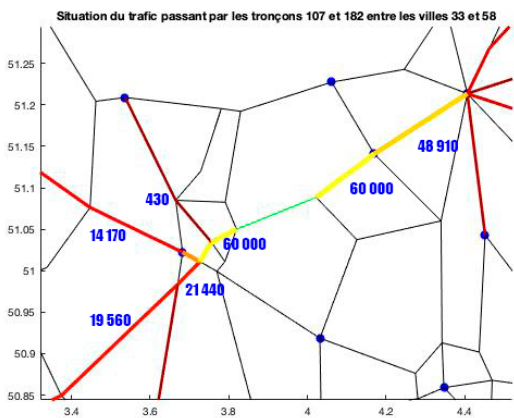


FIGURE 6.41 – Zoom sur la situation locale

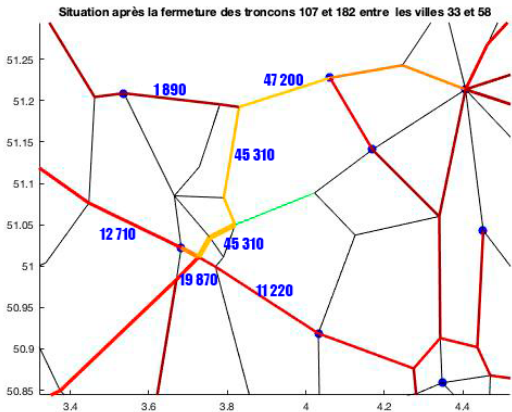


FIGURE 6.42 – Zoom sur la situation locale après la fermeture

6.4.10 Récapitulatif

Tronçons	Véhicules impactés	Temps supplémentaires [min]	Pourcentage du temps de trajet supplémentaires	CO ₂ supplémentaires [Tonne]
81 - 82	57 574	148,4	244%	202,5
81 - 88	29 470	134,6	123%	29,9
88 - 89	15 221	50,1	31%	40,9
89 - 91	29 112	57	29%	71,2
91 - 92	72 916	22	18%	295,3
79 - 89	82 694	45,7	21%	543,6
94 - 95	68 001	187,4	141%	792,9
156 - 157	13 174	25,8	13%	179,2
33 - 58	59 998	190,4	144%	570,9

TABLE 6.10 – Tableau récapitulatif des résultats

Excepté pour le tronçon 91 - 92, au plus de véhicules sont impactés, au plus la prolongation du temps est importante. C'est plutôt cohérent car, plus des usagers doivent changer de parcours, plus la déviation sera encombrée. Si la déviation est une petite nationale, elle n'est pas adaptée pour recevoir tout le trafic d'une autoroute.

Le cas du tronçon 91 - 92 a déjà été abordé plus tôt, le principe est que cette fermeture a entraîné le désengorgement des routes à proximité. Cela a eu pour effet de ne pas faire exploser le temps supplémentaire. Le tronçon 156-157 agit de la même façon, dans une moindre mesure. Il n'est dès lors pas étonnant de remarquer ces deux fermetures sont celles qui impactent le moins le temps de trajet.

De même, plus les kilomètres additionnels sont importants, plus le temps additionnels est important.

Les cinq premières analyses traitent de tronçons voisins sur la E42. Il est intéressant de voir que ces tronçons réagissent très différemment. Les tronçons (81-88) et (88-89) proposent des déviations uniquement locales, tandis que les autres remanient la circulation sur l'ensemble du réseau. Ces tronçons ont chacun un nombre très différent de véhicules impactés, notamment à cause de la proximité avec la ville de Charleroi qui est une grande source de trafic.

Deux tendances se dessinent, soit la redistribution est présente sur l'ensemble de la Belgique (94-95), soit la redistribution est locale (81-88). Sur le tronçon (88-89), il n'y a d'ailleurs qu'une seule déviation.

La figure 6.43 met sur un graphique les temps supplémentaires en fonction du nombre de véhicules impactés. Deux valeurs semblent s'isoler du reste des données, les autres ont une relation linéaire entre elles.

La figure 6.44 montre le nombre de kilomètres supplémentaires en fonction du nombre de véhicules impactés. Bien que le nombre de kilomètres semble augmenter en fonction du nombre de véhicules impactés, aucune tendance claire ne se dessine.

Il fait néanmoins faire attention, ces graphes ne représentent que neuf tronçons sur les trois cents que comptent le modèle.

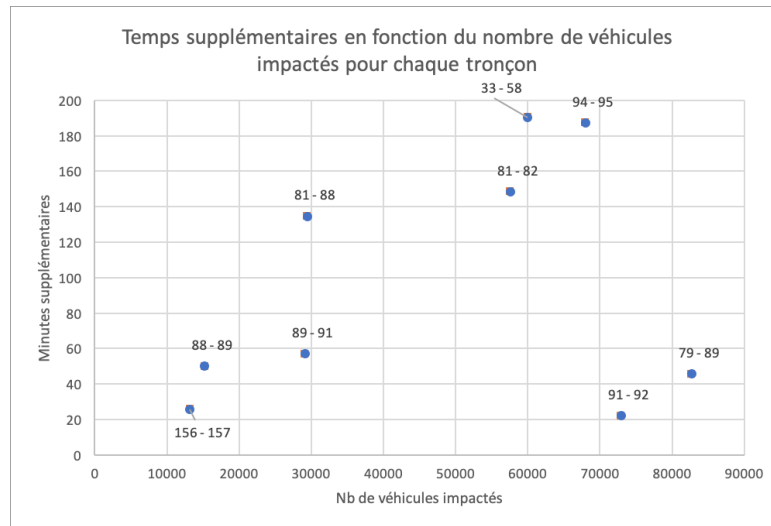


FIGURE 6.43 – Temps supplémentaires en fonction du nombre de véhicules impactés pour chaque tronçon

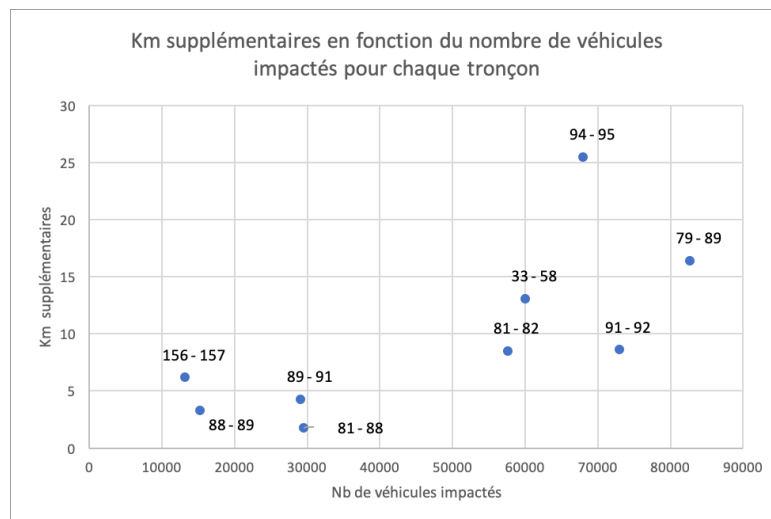


FIGURE 6.44 – Km supplémentaires en fonction du nombre de véhicules impactés pour chaque tronçon

Chapitre 7

Critique, pistes d'amélioration et conclusions

7.1 Critique du modèle

Ce travail est un perfectionnement de celui réalisé par Joachim et Adrien. Il a rempli les objectifs fixés, à savoir d'ajouter une composante temporelle au modèle et de représenter visuellement l'impact d'une fermeture de route. Néanmoins, le modèle est perfectible, car il pose encore quelques hypothèses trop simplificatrices. Cette section a pour but d'expliquer les limitations rencontrées dans ce modèle tandis que la section suivante propose des pistes d'amélioration.

- La détermination des trajets ne prend pas en compte les prévisions de bouchons. Le trajet le plus court change en fonction de la journée. Par exemple, pour aller de Bruges à Charleroi, si le trajet se fait de nuit, le chemin le plus rapide passe par Bruxelles. Néanmoins, à 8h du matin, le ring est toujours encombré et le chemin le plus rapide devient le passage par Tournai.
- Le modèle crée principalement des trajets entre les grandes villes et néglige les routes secondaires ainsi que le trafic international. Il n'y a que 50 villes hôtes, et elles représentent un tiers de la population belge. Tous les habitants des campagnes sont négligés, et donc certains tronçons sont sur-représentés et d'autre sous-représentés. Cela se remarque par le fait que certains tronçons sont inutilisés tandis que d'autre sont saturés.
- La simulation n'accepte pas la fermeture dans un seul sens de circulation. Cela est dû au fait que l'algorithme de Dijkstra, que nous utilisons dans ce modèle, n'accepte pas les graphes orientés.

7.2 Pistes d'amélioration

Quelques améliorations possibles pour un travail futur :

- Un des principaux problèmes du modèle est que les automobilistes ne modifient pas leur trajet en cours de route. Une amélioration possible serait de prendre en compte la diminution de vitesse dans la détermination du parcours le plus court et de le recalculer dès qu'un véhicule doit changer de tronçon.
- Le modèle considère que tout les véhicules roulent à la vitesse maximale autorisée. Or, tous les véhicules ne roulent pas à la même vitesse, et certains sont en excès de vitesse. Il pourrait être intéressant de chercher un profil de vitesse des véhicules pour une modélisation plus réaliste.
- Certaines données du modèle proviennent de sources anciennes. Certaines données seront mises à jour dans le rapport Monitor.

7.3 Conclusion

Dans ce travail, l'importance des réseaux routiers a d'abord été soulignée, tant au niveau économique que social. Puis la résilience a été définie comme la capacité de notre réseau routier à retrouver un état de fonctionnement normal après une perturbation, telle que la fermeture partielle ou totale d'un tronçon suite à un accident ou des travaux.

Ensuite, un modèle a pu être construit pour obtenir les premiers résultats sur la résilience du réseau. Ce modèle a été construit sur base du travail de Joachim et Adrien. Pas à pas, avec les données fournies par le SPF mobilité et transport, des hypothèses et des recherches additionnelles ont permis de mieux représenter la réalité avec le modèle.

La simulation calcule l'état du réseau d'une part pour le réseau normal et, d'autre part, pour le réseau perturbé. Une analyse permet de tirer des informations comme le temps de parcours supplémentaire ou la quantité additionnelle de CO₂ émis.

Ce mémoire a donc permis de créer un modèle du trafic routier. Ce travail va être utilisé en vue de créer un outil d'aide à la gestion des ouvrages d'art. D'une part en aidant le gestionnaire du réseau à mesurer l'impact d'une fermeture de route, d'autre part en prévoyant les ouvrages d'art que le trafic risque de dégrader.

Bibliographie

- [1] Association mondiale de la route. *Importance de l'entretien routier*, 2014.
- [2] L'hoir Joachim et Tellier Adrien. Analyse de la resilience du reseau routier belge. *Mémoire*, 2018.
- [3] Garavaglia E. Sgambi L. Selective maintenance planning based on markovian approach of a steel truss bridge. *Engineering Structures*, 125, pages 532–545, 2016.
- [4] Benoit Barbedette. Le transport routier de marchandise : un maillon indispensable de la vie économique du pays. *Wolters Kluwer France*, Avril 2015.
- [5] Hertince Ntomba. Le rôle du réseau routier dans l'intégration et la croissance économique : impacts des ring et routes de desserte agricole dans la création des richesses en rdc. 2010.
- [6] K.G. Fenelon. The economics of road transport. *Routledge library editions : transport economics*, 1925.
- [7] Gérard Placq. Le developpement du reseau routier belge de 1830 à 1940. *Bulletin de l'Institut de Recherches Economiques et Sociales*, Aout 1951.
- [8] Huybrechts André. Le rôle du progrès des transports dans les économies sous-développées. *Revue économique*, volume 22, pages 140–162, 1971.
- [9] Richard T. T. Forman. Estimate of the area affected ecologically by the road system in the united states. *Conservation Biology*,, Février 2000.
- [10] Serge Lhomme Samuel Rufat Magali Reghezza-Zitt Géraldine Djament-Tran, Antoine Le Blanc. Ce que la résilience n'est pas, ce qu'on veut lui faire dire. *hal-00679293f*, 2011.
- [11] Jenelius E Mattsson, L.-G. Vulnerability and resilience of transport systems – a discussion of recent research. *transport. res. part a*. 2015.
- [12] Jose E. Ramirez-Marquez Seyedmohsen Hosseini, Kash Barker. A review of definitions and measures of system resilience. 2015.
- [13] David D. Woods. Four concepts for resilience and the implications for the future of resilience engineering. 2015.
- [14] Robert MacArthur. Fluctuations of animal populations and a measure of community stability. *Ecological Society of America*, 1955.
- [15] Stuart Pimm. The complexity and stability of ecosystems. *NATURE VOL 307*, pages 321–325, january 1984.
- [16] C.S. Holling. Resilience and stability of ecological systems. 1973.
- [17] Adam Rose. Measuring economic resilience : Recent advances and future priorities. pages 1–5, September 2017.

- [18] Adam Rose. Economic resilience to disasters. november 2000.
- [19] Elisabeth Krausmann Adam Rose. An economic framework for the development of a resilience index for business recovery. pages 73–83, 2013.
- [20] Reggiani et al. Transport resilience and vulnerability : The role of connectivity. transport. res. part a. 2015.
- [21] R. Patriarca et al. Resilience engineering : Current status of the research and future challenges. 2017.
- [22] N Leveson E Hollnagel, D Woods. Resilience engineering : Concepts and precepts. 2006.
- [23] Garavaglia E. Sgambi L. The use of a credibility index in life-cycle assessment of structures. *Structure and Infrastructure Engineering*, 11(5), pages 683–694, 2015.
- [24] E. Foerster. Vulnérabilité : état de l’art sur les concepts et méthodologies d’évaluation. 2009.
- [25] Site internet du SPF mobilité et transport. Consulté en avril 2019. <https://mobilit.belgium.be/fr>.
- [26] J. Bottom M.Mahut-A. Paz R. Balkrishna T. Waller et J. Hicks Yi-Chang, Y. Chiu. Dynamic traffic assignment. page 3, 2011.
- [27] Temple University Benjamin Seibold. A mathematical introduction to traffic flow theory. *Support de cours*, 2015.
- [28] Igor Grabec. Forecasting of traffic jams at disturbed sections of high-ways. pages 1–8, 2014.
- [29] Mathieu Strale. L’évolution récente du transport de marchandises en belgique. pages 1–16, 2011.

Annexe A

Distribution statistique

Cette partie est un rappel pour mieux comprendre les distributions statistiques utilisées dans ce travail. Les probabilités vont être utiles pour modéliser les phénomènes aléatoires. Cette partie s'inspire du cours "LSFAB1105 - Probabilité et statistique", donné par les professeurs Anouar EL GHOUGH et Rainer VON SACHS. La première section de ce rappel donne les définitions de base des probabilités, tandis que la seconde fait le point sur les différentes distributions statistiques existantes.

A.1 Définition

Définition A.1. *La **population** est l'ensemble des données mesurées.*

Définition A.2. *Un **échantillon** est une séquence aléatoire d'une partie de la population.*

Définition A.3. *La **moyenne** peut être définie comme*

$$\bar{x} = \frac{\sum_i x_i}{n}$$

Définition A.4. *Le **percentile** q_p est, pour une valeur de p donnée entre 0 et 1, la valeur telle qu'il y a $p\%$ de valeur plus petite que lui et $(1-p)\%$ de valeur plus grande dans l'ensemble de la population.*

Définition A.5. *Le percentile 0,5 est aussi appelé **médiane**. C'est la valeur au milieu des observations.*

Définition A.6. *L'**écart** est la longueur de la tranche des données :*

$$Ecart = x_{max} - x_{min}$$

Définition A.7. *L'**écart interquantile** :*

$$IQR = q_{0.75} - q_{0.25}$$

Définition A.8. Les *valeurs aberrantes* sont les données qui semblent être séparées du reste des données. Une donnée peut être considérée comme aberrante si elle n'appartient pas à l'intervalle

$$[q_{0,25} - 1,5 \times IQR ; q_{0,75} + 1,5 \times IQR]$$

Définition A.9. Un *histogramme* est un diagramme qui permet de visualiser la distribution d'une série.

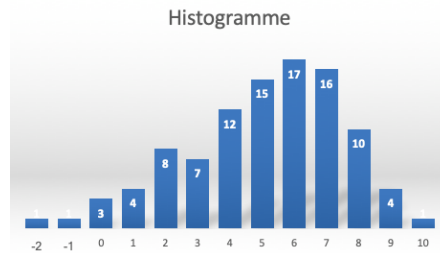


FIGURE A.1 – Exemple d'histogramme

Définition A.10. La *variance* est un moyen de quantifier la dispersion des données par rapport à la moyenne.

$$s^2 = \frac{1}{n-1} \sum_i (x_i - \bar{x})^2$$

Définition A.11. L'*écart-type* est la racine carrée de la variance.

$$s = \sqrt{s^2}$$

Définition A.12. *Coefficient de variation* :

$$cv = \frac{s}{\bar{x}}$$

Définition A.13. Une *expérience* est une opération dont le résultat n'est pas connu avec certitude à l'avance.

Définition A.14. L'*espace des possibles* S décrit tous les résultats possibles pour une expérience

Définition A.15. Un *événement* est une combinaison de résultats possibles. Il peut également être défini comme étant un sous-espace de S .

Proposition A.1. La probabilité que l'événement A se passe : $P(A) = \frac{\text{\#cas favorable}}{\text{\#cas défavorable}}$

Définition A.16. L'événement *complémentaire* de l'événement A est noté $\bar{A}$.

$$P(\bar{A}) = 1 - P(A)$$

Proposition A.2.

$$P(A) = P(A \cap B) + P(A \cap \bar{B})$$

Proposition A.3.

$$P(A \cup B) = P(A) + P(B) - P(A \cap B)$$

Proposition A.4.

$$P(A|B) = \frac{P(A \cap B)}{P(B)}$$

Définition A.17. Deux événements A et B sont **indépendants** si :

$$P(A \cap B) = P(A) * P(B)$$

Proposition A.5. Théorème de Bayes :

$$P(B|A) = \frac{P(A|B)P(B)}{P(A)}$$

Définition A.18. Une **variable aléatoire** (VA) X est un nombre réel associé au résultat d'une expérience aléatoire.

Définition A.19 (Variable aléatoire discrète). La **fonction de distribution** (pmf) $p(x)$ d'une variable aléatoire discrète X , est la fonction $p(x)$ qui associe à chaque valeur x de X , la probabilité $P(X = x)$.

$$p(x) = P(X = x)$$

$$P(X \in I) = \sum_{i: x_i \in I} p(x_i)$$

Définition A.20 (VA discrète). La **fonction de répartition** ou **fonction de distribution cumulative** (cdf) $F(x)$ d'une variable aléatoire X est définie comme

$$F(x) = P(X \leq x)$$

$$F(x) = \sum_{i: x_i \leq x} p(x_i)$$

Définition A.21 (VA continue). La **fonction de densité de probabilité** (pdf) pour une VA continue est la fonction $f(x) \geq 0$ telle que pour tout ensemble I

$$P(X \in I) = \int_I f(x) dx$$

Définition A.22 (VA continue). **Fonction de répartition** (cdf)

$$F(x) = P(X \leq x)$$

$$F(x) = \int_{-\infty}^x f(t) dt$$

$$f(x) = F'(x)$$

Remarques :

$$\sum_i p(x_i) = 1$$

$$\int_{-\infty}^{+\infty} f(x)dx = 1$$

Définition A.23. Espérance $E(X) = \mu_x$

Dans le cas d'un VA discrète :

$$E(X) = \sum_x x \times p(x)$$

$$E(h(X)) = \sum_x h(x)p(x)$$

Dans le cas d'un VA continue :

$$E(X) = \int_{-\infty}^{+\infty} x * f(x)dx$$

$$E(h(X)) = \int_{-\infty}^{+\infty} h(x)f(x)dx$$

Définition A.24. Variance σ_X^2 d'une VA. A ne pas confondre avec la variance s^2 d'un échantillon défini à la définition A.10.

$$\sigma_X^2 = E\left((X - \mu_X)^2\right)$$

Définition A.25. Ecart-type d'une VA.

$$\sigma_X = \sqrt{\sigma_X^2}$$

A.2 Exemple de distribution statistique

Cette section va maintenant traiter des principales distributions statistiques existantes. Une distribution est une fonction qui associe à chaque valeur obtenue une fréquence d'apparition. Une distribution cumulative est l'intégrale de la distribution.

Les distributions discrètes seront abordées, puis les distributions continues.

Distribution discrète

A.2.1 Distribution uniforme

La variable aléatoire X a une distribution uniforme discrète si chaque valeur final $\{x_1, x_2, x_3, \dots, x_n\} \subset \mathfrak{R}$ a une probabilité égale d'être choisie.

$$P(X = x_i) = \frac{1}{n}$$

$$\mu_X = \frac{\sum x_i}{n}$$

Exemple A.1. Une expérience est défini comme un lancer de dé, la variable aléatoire X est le résultat du lancer. Sa distribution est uniforme, comme constaté sur les figures A.2 et A.3.

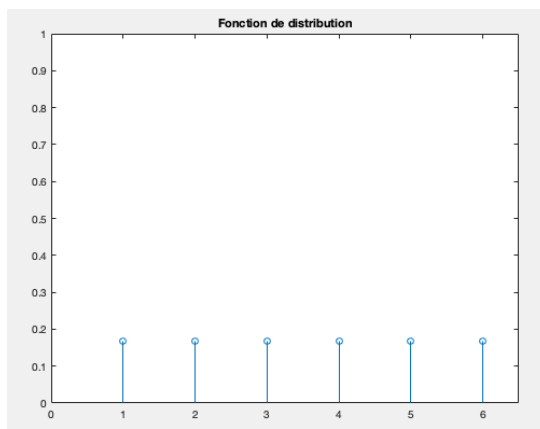


FIGURE A.2 – Fonction de distribution uniforme

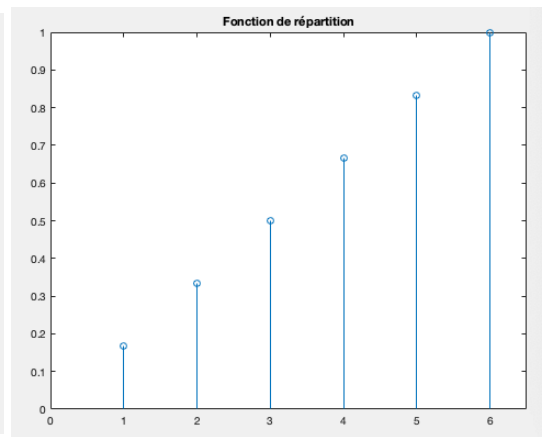


FIGURE A.3 – Fonction de répartition uniforme

A.2.2 Distribution de Bernoulli

Un essai de Bernoulli est une expérience où le seul résultat possible est "réussite" ou "échec", où p est la probabilité d'une réussite.

$$X = \begin{cases} 1 & \text{si succes} \\ 0 & \text{si echec} \end{cases}$$

$$P(X = x) = p^x(1 - p)^{1-x}$$

$$\mu_X = p$$

A.2.3 Distribution binomiale

Si l'essai de Bernoulli est répété n fois. La VA X est le nombre total de succès en n lancer.

$$X \sim Bi(n, p)$$

$$P(X = x) = C_n^x p^x (1 - p)^{n-x}$$

$$\mu_X = np$$

Exemple A.2. Un succès est défini comme étant obtenir un résultat supérieur ou égal à 3 lors d'un lancer de dé. La probabilité p de réussite est $p=2/3$.

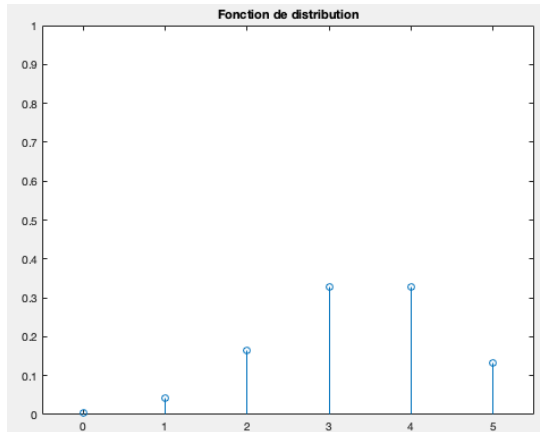


FIGURE A.4 – Fonction de distribution binomiale

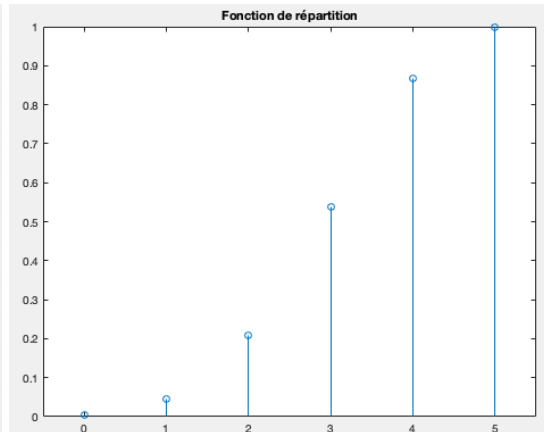


FIGURE A.5 – Fonction de répartition binomiale

A.2.4 Distribution géométrique

L'essai de Bernoulli est répété jusqu'au premier succès. La VA X est le nombre d'essais.

$$P(X = x) = (1 - p)^{x-1} p$$

$$\mu_X = \frac{1}{p}$$

Exemple A.3. Comme pour l'exemple précédent, un succès est défini comme obtenir un résultat supérieur ou égale à 3. La VA X est le nombre de lancers avant d'obtenir le premier succès.

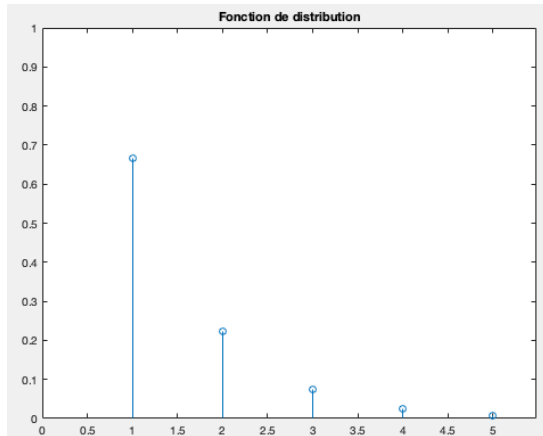


FIGURE A.6 – Fonction de distribution géométrique

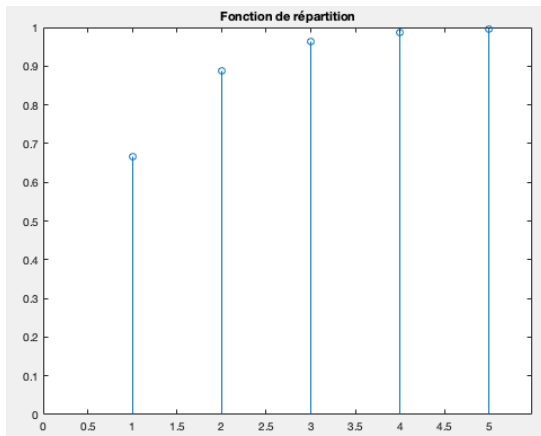


FIGURE A.7 – Fonction de répartition géométrique

A.2.5 Distribution de Poisson

Distribution qui compte le nombre d'événements pendant un certain temps ou un certain espace. Exemple : le nombre d'accidents sur une route pendant une année. Paramètre λ , le temps entre 2 occurrences.

$$P(X = x) = e^{-\lambda} \frac{\lambda^x}{x!}$$

$$\mu_X = \lambda$$

Distribution continue

Passons maintenant à quelques distributions de VA continue.

A.2.6 Distribution uniforme

La VA X suit une distribution uniforme sur l'intervalle $[a, b]$ si elle a une fonction de distribution constante sur l'intervalle.

$$f(x) = \frac{1}{b - a}$$

Exemple A.4. Distribution uniforme sur l'intervalle $[1, 3]$.

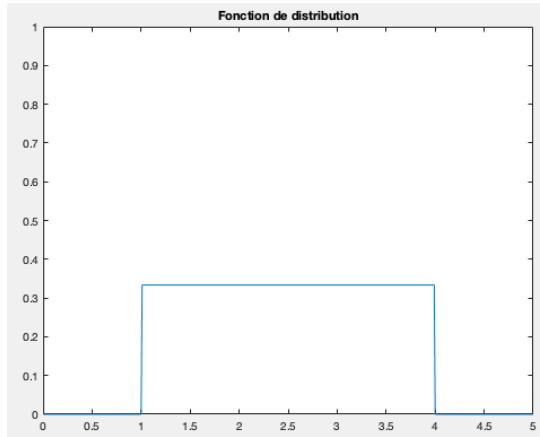


FIGURE A.8 – Fonction de distribution uniforme

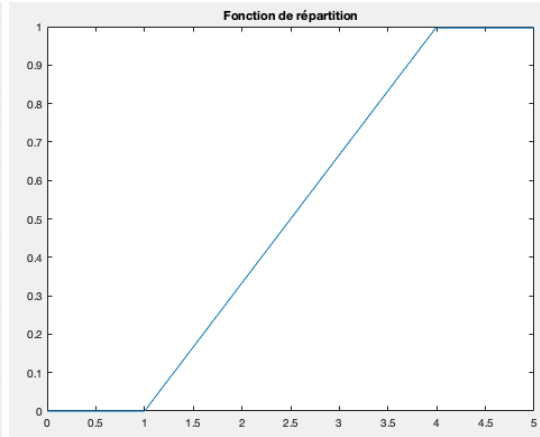


FIGURE A.9 – Fonction de répartition uniforme

A.2.7 Distribution normale

Distribution la plus utilisée. Elle dépends de μ et σ .

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$$

Exemple A.5. *Distribution normale pour $\mu = 0$ et $\sigma = 1$*

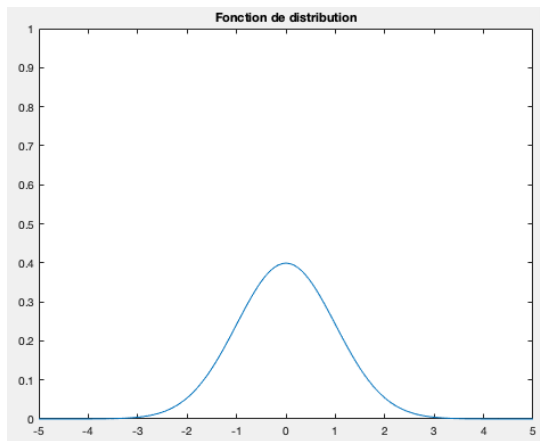


FIGURE A.10 – Fonction de distribution normale

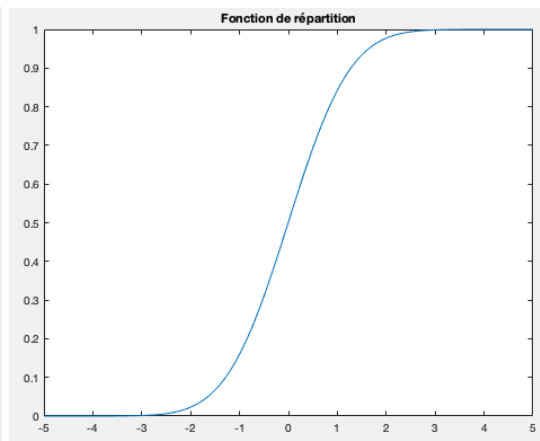


FIGURE A.11 – Fonction de répartition normale

A.2.8 Distribution Gamma

$$X \sim \text{Gamma}(\alpha, \beta)$$

La distribution Gamma est définie comme

$$f(x) = \frac{1}{\Gamma(\alpha)} \frac{x^{\alpha-1}}{\beta^\alpha} e^{-\frac{x}{\beta}}$$

Où $\Gamma(\alpha) = \int_0^\infty t^{\alpha-1} e^{-t} dt$ est la fonction Gamma

A.2.9 Distribution exponentielle

La distribution exponentielle est définie comme

$$f(x) = \frac{1}{\beta} e^{-\frac{x}{\beta}}$$

Où β est l'espérance.

A noter que $\text{Expo}(\beta) = \text{Gamma}(1, \beta)$

A.2.10 Distribution bimodale

Une distribution bimodale est une distribution avec deux pics. Elle est le résultat d'une addition de deux distributions normales. Néanmoins, il faut rester attentif à ce que l'intégrale de la distribution reste égale à un. Il convient donc de définir un paramètre λ tel que :

$$f(x) = \lambda \times g_1(x) + (1 - \lambda) \times g_2(x)$$

Où f est la distribution bimodale f qui dépend des deux distributions normales g_1 et g_2 .

La loi bimodale dépend donc de cinq paramètres.

Exemple A.6. *Distribution bimodale définie comme étant la somme d'une distribution normale avec comme paramètres $\mu = 0$ et $\sigma = 1$ et d'une autre distribution normale avec comme paramètres $\mu = 4$ et $\sigma = 1$, et $\lambda = 0,5$.*

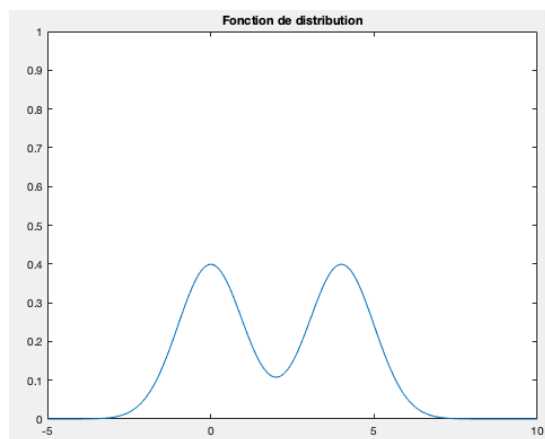


FIGURE A.12 – Fonction de distribution bimodale

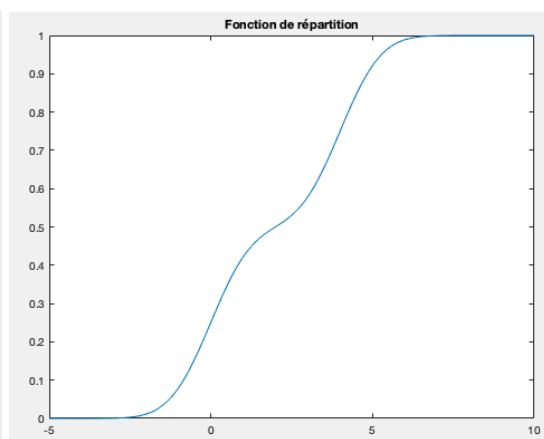


FIGURE A.13 – Fonction de répartition bimodale

Annexe B

Théorie des graphes

Les villes et les routes de notre modèle peuvent être représentés par un graphe. Cette partie étudie la théorie des graphes et explique les notions nécessaires à la compréhension de ce travail. Cette partie s'inspire du cours LINMA1691 - "Mathématiques discrètes I : Théorie et algorithmique des graphes" donnée par les professeurs Vincent BLONDEL et Jean-Charles DELVENNE. La première section de cette annexe donne les définitions de base de la théorie des graphes.

Les images contenues dans cette annexe sont tirées du cours cité ci dessus.

B.1 Définition

Définition B.1. Un **graphe** $G = (V, E)$ est un ensemble V de noeuds et un ensemble fini E d'arêtes. Chaque arête relie deux noeuds du graphes.

Définition B.2. Deux sommets incidents à la même arêtes sont dits **adjacents**.

Définition B.3. Une **boucle** est une arête incidente à un seul noeud.

Définition B.4. Deux arêtes sont dites **multiples** si elles relient les mêmes noeuds incidents.

Définition B.5. Le **degré** d'un sommet et le nombre d'arêtes incidentes à celui-ci. Une boucle compte pour deux.

Définition B.6. Un **graphe simple** est un graphe qui ne possède pas de boucle ni d'arête multiple. Si un graphe n'est pas simple, il est appelé **multi graphe**.

Définition B.7. Le graphe $G' = (V', E')$ est un **sous-graphe** du graphe $G = (V, E)$ si V' et E' sont incluses respectivement dans V et E .

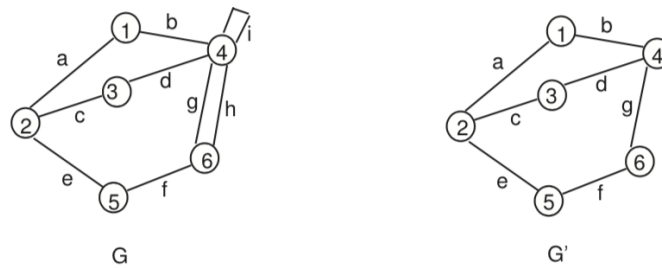


FIGURE B.1 – Un multi graphe et un graphe simple

Définition B.8. Un graphe est **connexe** s'il existe un parcours qui relie tous les sommets du graphe.

Définition B.9. Les **composantes connexes** d'un graphe sont ses sous-graphes connexes maximaux.

Définition B.10. Un **parcours** est une suite de sommets et d'arêtes.

Définition B.11. Un **chemin** est un parcours dont les sommets et arêtes sont tous distincts.

Définition B.12. Un **cycle** est un chemin fermé.

Définition B.13. Un **parcours Eulérien** est un parcours qui visite chaque arête une et une seule fois.

Définition B.14. Un **graphe Eulérien** est un graphe où il existe un parcours Eulérien fermé.

Définition B.15. Un **graphe orienté** $D = (V, A)$ est un graphe dont les arêtes sont dirigées d'un sommet vers un autre.

B.2 Graphes particuliers

Sur la figure B.2, pour pouvez voir quelques exemples de graphes particuliers.

Un graphe $G = (V, E)$ est biparti s'il existe une partition V_1, V_2 des sommets tel que chaque arête ait une extrémité dans V_1 et une extrémité dans V_2 .

Un graphe est dit complet si toutes les arêtes possibles sont présentes.

Un arbre est un graphe connexe qui ne possède pas de cycle.

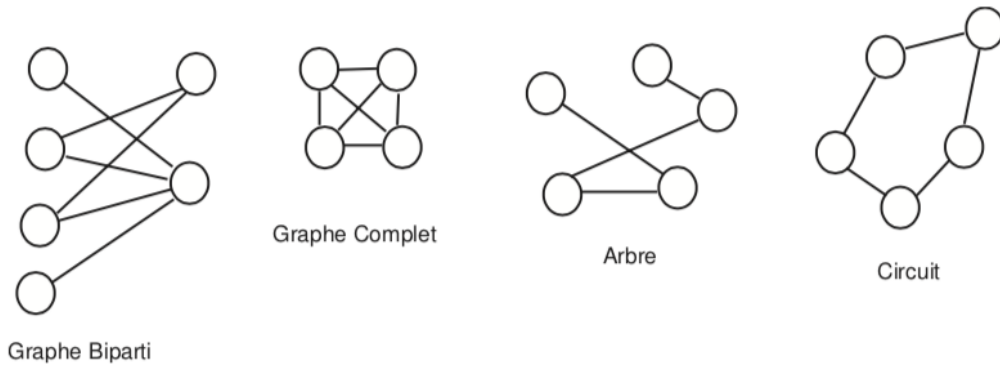


FIGURE B.2 – Exemple de graphe particulier

B.3 Représentation des données

Il existe plusieurs façons de représenter un graphe non-orienté. Dans cette section seront présentées les trois méthodes de représentation les plus utilisées. Dans ce mémoire, la méthode utilisée est la matrice d'adjacence, qui est la plus facile à utiliser et la plus adaptée.

B.3.1 Matrice d'adjacence

Une des méthode est par la matrice d'adjacence, ou matrice noeud-noeud. La matrice d'adjacence est une matrice $n \times n$, ou n est le nombre de sommets. L'élément a_{ij} est égal au nombre d'arêtes liant le sommet i et le sommet j . Cette matrice est symétrique pour un graphe non-orienté.

Prenons le graphe donné a la figure B.3 et trouvons la matrice d'adjacence.

$$\begin{matrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{matrix} \begin{pmatrix} 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 2 \\ 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 2 & 0 & 0 \end{pmatrix} \quad (\text{B.1})$$

B.3.2 Matrice d'incidence

Une autre manière de représenter un graphe est par sa matrice d'incidence, ou matrice noeud - arête. C'est une matrice $n \times m$, ou n est le nombre de sommet et m le nombre d'arête. L'élément $a_{ie} = 1$ si l'arête e est incidente au sommet

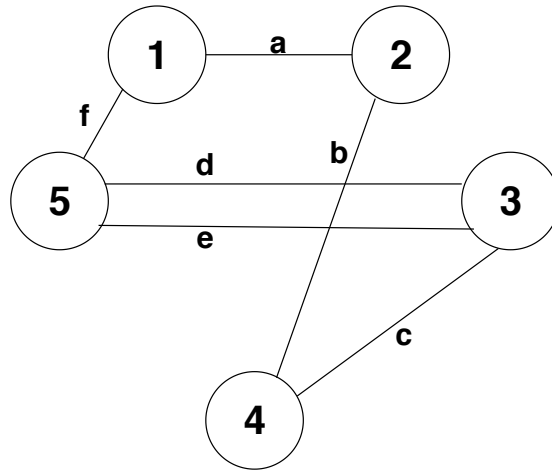


FIGURE B.3 – Graphe

i, 2 si e est une boucle sur i et 0 sinon.

Nous pouvons maintenant représenter la matrice d'incidence de notre exemple

$$\begin{matrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{matrix} \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 \end{pmatrix} \quad (\text{B.2})$$

B.3.3 Liste des voisins

Une représentation du graphe peut aussi se faire en dressant une liste des voisins de chaque noeud. Dans l'exemple il y a donc :

- $V(1) = \{2, 5\}$
- $V(2) = \{1, 4\}$
- $V(3) = \{4, 5, 5\}$
- $V(4) = \{2, 3\}$
- $V(5) = \{1, 3, 3\}$

Cette représentation est plus compacte que la matrice d'incidence pour les graphes creux.

Annexe C

Matrice xyv

Le tableau suivant reprend tout les noeuds associés au modèle. A chaque noeud est associé son numéro dans le programme, le nom de la ville correspondante, ses coordonnées géographiques ainsi que son nombre d'habitants pour les villes hôtes.

Les deux premières colonnes ont été ajoutées ici par soucis de visibilité, seules les trois dernières colonnes font partie du fichier .txt que le code Matlab prend en entrée.

Numéro de la ville	Ville	Latitude	Longitude	Population
1	Poperinge	50.851065	2.727077	0
2	Ypres	50.840809	2.877574	34950
3	<i>Ypres</i>	50.864349	2.859591	0
4	Furnes	51.071442	2.661245	0
5	La Panne	51.112594	2.623599	11129
6	Nieuwpoort-Bad	51.150494	2.720967	0
7	Nieuwpoort	51.133458	2.747459	0
8	Dixmude	51.032085	2.863584	0
9	Middelkerke	51.142095	2.860154	0
10	Gistel	51.159468	2.937191	0
11	Ostende	51.214932	2.929543	71224
12	Jabbeke	51.193856	3.070385	0
13	Zeebruges	51.327764	3.184871	0
14	Knokke	51.341914	3.285551	33097
15	Bruges	51.210453	3.213393	138116
16	Maldegem	51.204373	3.462191	0
17	Aalter	51.076226	3.447567	0
18	Loppem	51.155455	3.211836	5173
19	Baliebrugge	51.07882	3.15294	0
20	Lichtervelde	51.010455	3.169989	0
21	Roulers	50.972113	3.177287	62301
22	Rijssseleind	50.983271	3.239902	0
23	Menin	50.81699	3.126376	0
24	Courtrai	50.824114	3.25988	76242
25	Belgiek	50.849517	3.374413	0

Numéro de la ville	Ville	Latitude	Longitude	Population
26	De Pinte	50.983636	3.670474	0
27	Gand	51.022268	3.682752	259270
28	<i>Gand</i>	51.010909	3.726809	0
29	Merelbeke	50.99932	3.769782	0
30	Melle	51.010909	3.790366	0
31	<i>Gand</i>	51.034679	3.755633	0
32	<i>Gand</i>	51.084934	3.665159	0
33	<i>Gand</i>	51.049986	3.819923	0
34	<i>Gand</i>	51.082627	3.791199	0
35	Evergem	51.12016	3.729129	0
36	Eeklo	51.208908	3.536676	20890
37	Zelzate	51.195496	3.78114	0
38	<i>Zelzate</i>	51.192274	3.830078	0
39	Stekene	51.227464	4.061797	18352
40	<i>Anvers</i>	51.242751	4.246104	0
41	<i>Anvers</i>	51.318101	4.352094	0
42	Anvers	51.213515	4.406728	521680
43	<i>Anvers</i>	51.268503	4.463119	0
44	Maria Ter Heide	51.328985	4.522845	0
45	Brecht	51.353932	4.630503	0
46	Arendonk	51.32316	5.077743	0
47	Turnhout	51.298529	4.952439	41708
48	Geel	51.152986	4.97457	0
49	Tongerlo	51.130426	4.938012	7112
50	Aarschot	51.000933	4.82986	0
51	Nieuwrode	50.964659	4.836152	0
52	Louvain	50.89528	4.688671	0
53	Malines	51.043088	4.451885	79503
54	<i>Bruxelles</i>	50.901974	4.436885	0
55	<i>Bruxelles</i>	50.913092	4.342372	0
56	Willebroel	51.060093	4.339859	0
57	Saint-Nicolas	51.140821	4.169763	73317
58	Lokeren	51.088476	4.02239	0
59	Termonde	51.037496	4.125719	0
60	Alost	50.918684	4.031888	82037
61	Ninove	50.829465	4.034887	0
62	Gapenberg	50.819955	3.869783	0
63	Renaix	50.744733	3.586769	25429
64	Frasnes	50.65914	3.59413	0
65	Tournai	50.632389	3.416022	69493
66	Bury	50.534503	3.602784	0
67	Ath	50.640008	3.774194	0
68	Lessines	50.671213	3.868589	0
69	Enghien	50.681498	4.03183	13103
70	Soignies	50.582437	4.068976	0

Numéro de la ville	Ville	Latitude	Longitude	Population
71	<i>Bruxelles</i>	50.722081	4.250504	0
72	<i>Bruxelles</i>	50.845678	4.280817	0
73	<i>Bruxelles</i>	50.876294	4.274714	0
74	Bruxelles	50.859012	4.347108	1139000
75	<i>Bruxelles</i>	50.86787	4.469903	0
76	<i>Bruxelles</i>	50.801453	4.464678	0
77	<i>Bruxelles</i>	50.687975	4.410077	0
78	Haut Ittre	50.661573	4.311061	0
79	Nivelles	50.569998	4.301324	26843
80	Ecaussinnes	50.52148	4.194324	0
81	La Louvière	50.496549	4.129476	78895
82	Mons	50.453743	3.961429	95265
83	Saint-Ghislain	50.451418	3.757309	0
84	Erquelinnes	50.323315	4.115359	0
85	Erquelinnes	50.299144	4.158481	0
86	Binche	50.412132	4.170664	0
87	Anderlues	50.406486	4.290311	0
88	Chapelle-lez-Herlaimont	50.486519	4.271241	0
89	Gosselies	50.47513	4.417623	0
90	Charleroi	50.417544	4.451205	204150
91	Fleurus	50.470875	4.539965	0
92	Sombreffe	50.487099	4.600751	8420
93	Gembloux	50.572383	4.697937	0
94	Perwez	50.62256	4.762106	0
95	Louvain-la-Neuve	50.66258	4.647784	30283
96	Chaumont-Gistoux	50.779014	4.717151	0
97	<i>Louvain</i>	50.844839	4.717813	0
98	Bertem	50.856896	4.641463	0
99	Incourt	50.671713	4.825272	0
100	Tirlemont	50.781277	4.921252	33133
101	Landen	50.712595	5.081203	0
102	Saint-Trond	50.805905	5.184163	37837
103	Hasselt	50.915294	5.324787	74694
104	Lummen	50.999609	5.22371	0
105	Hulst	51.07813	5.137489	0
106	Mol	51.179731	5.12466	0
107	Lommel	51.22221	5.3139	31536
108	Overpelt	51.218041	5.399258	0
109	<i>Overpelt</i>	51.212234	5.423642	0
110	<i>Overpelt</i>	51.230879	5.40639	0
111	Bourg-Léopold	51.117869	5.263273	0
112	Brée	51.139669	5.587868	0
113	Hechtel-Eksel	51.126789	5.38552	0
114	Hasselt	51.00942	5.305589	0
115	Houthalen-Helchteren	51.0126	5.37167	0

Numéro de la ville	Ville	Latitude	Longitude	Population
116	Genk	50.992795	5.500951	66059
117	Lanklaar	51.025697	5.726632	0
118	Maaseik	51.09363	5.796404	0
119	Maasmechelen	50.953428	5.693982	38193
120	Diepenbeek	50.907101	5.432209	0
121	Ketsingen	50.793262	5.530808	0
122	Tongres	50.779954	5.459893	30650
123	Waremmes	50.704915	5.29381	0
124	Fernelmont	50.531958	5.010334	0
125	Eghezée	50.591757	4.912854	0
126	Warisoulx	50.521038	4.881605	0
127	Rhisnes	50.49462	4.786848	0
128	Namur	50.46487	4.917221	110628
129	Floreffe	50.427294	4.75355	0
130	Sambreville	50.457738	4.642788	0
131	Fosses-La-Ville	50.394451	4.675191	0
132	Chatelet	50.41677	4.499214	0
133	Vodecée	50.207899	4.592195	0
134	Philippeville	50.196789	4.529761	9239
135	Beaumont	50.235111	4.238777	0
136	Chimay	50.048954	4.309659	9896
137	Baileux	50.035265	4.375699	0
138	Couvain	50.055679	4.492093	0
139	Onhayes	50.242714	4.827163	0
140	Heer	50.169495	4.824133	0
141	Beauraing	50.099958	4.993195	9050
142	Gedinne	49.984623	4.977697	0
143	Menuchenet	49.849238	5.091061	0
144	Libramont	49.902502	5.325136	0
145	Tielt	51.004206	3.338194	0
146	Neufchâteau	49.861719	5.507438	7533
147	Florenville	49.698232	5.312021	0
148	Arlon	49.686382	5.748481	29053
149	Messancy	49.650298	5.81869	0
150	Aubange	49.56389	5.819156	16856
151	Martelange	49.782714	5.749069	0
152	Bastogne	49.994446	5.705838	15127
153	Tenneville	50.108134	5.473774	0
154	Saint-Hubert	50.027595	5.38373	0
155	Seviscourt	49.969042	5.372813	0
156	Wellin	50.093795	5.135473	0
157	Ciney	50.268748	5.035812	0
158	Dinant	50.244121	4.99694	0
159	Dinant	50.242285	4.916811	13912
160	Assesse	50.397972	4.975077	0

Numéro de la ville	Ville	Latitude	Longitude	Population
161	Emptinne	50.318252	5.129136	0
162	Marche en famene	50.225383	5.337409	17487
163	Ferrières	50.413577	5.584191	0
164	Werbomont	50.38274	5.674989	4918
165	Baraque de Fraiture	50.248601	5.73078	0
166	Bèche	50.265989	5.904413	0
167	Malscheid	50.156091	6.051588	0
168	Saint-Vith	50.266975	6.113942	9481
169	Malmédy	50.416985	5.993402	12161
170	Stavelot	50.396431	5.932305	0
171	Trois-Ponts	50.37225	5.871001	0
172	Spa	50.492512	5.866797	0
173	Theux	50.55013	5.869331	0
174	Limbourg	50.621849	5.943074	0
175	Herve	50.637015	5.836947	0
176	Herstal	50.680573	5.658757	0
177	Liège	50.635977	5.576956	195875
178	Chaufontaine	50.553467	5.644516	0
179	Aywaille	50.462376	5.705882	0
180	Tinlot	50.480293	5.369528	0
181	Modave	50.485018	5.340406	0
182	Huy	50.519969	5.238531	21354
183	Amon Sotia	50.581144	5.211129	0
184	Gonhis	50.561403	5.545585	0
185	<i>Saint-Vith</i>	50.278531	6.127455	0
186	Hamont	51.250845	5.541983	14246

TABLE C.1 – Liste des villes du fichier xyv

Annexe D

Tutoriel du code Matlab

Cette annexe a pour but d'expliquer en détail le fonctionnement du code Matlab. Le code se décompose en quatre parties, la détermination des trajets, la simulation sans fermeture, la simulation de la fermeture de la route et enfin l'analyse des résultats. Le script principal s'appelle `main.m`. Les codes Matlab se trouvent dans l'annexe E.

D.1 Initialisation

Fichiers d'entrée

Le code va tout d'abord lire 3 fichiers textes contenant les informations du réseau.

- **xyv.txt** : Reprend les coordonnées géographiques des 168 villes ainsi que la population pour les villes hôtes. Correspond au 3 dernières colonnes du tableau de l'annexe C.
- **Vitesse.txt** : Matrice 168×168 , où l'élément i,j vaut la vitesse maximale autorisée sur le tronçon entre i et j , et 0 s'il n'y pas de route entre ces deux villes.
- **Bande.txt** : Matrice 168×168 , où l'élément i,j vaut le nombre de bandes sur le tronçon entre i et j .

Choix des paramètres

Ces paramètres sont laissés au choix de l'utilisateur.

- **choix_poids** : Choix des poids des arêtes dans l'algorithme de Dijkstra. Soit 'temps', soit 'distance'.
- **pack** : nombre de véhicules par paquet. 1 par défaut, les résultats obtenus pour 5 et 10 sont très similaires à ceux pour 1 et nettement plus rapides.
- **dt** : pas de temps pour la simulation temporelle.
- **dx** : longueur de chaque sous-tronçon.

Calcul des variables

Ensuite, Matlab va calculer lui-même quelques valeurs utiles pour la suite :

- **n** : nombre de villes (168).
- **n_route** : nombre de routes (303).
- **D** : Matrice des distances entre les villes (110% de la distance à vol d'oiseau).
- **T** : Matrice des temps théorique entre les villes pour une voiture.
- **Ville** : Liste des villes hôtes.
- **Ville_d** : Liste des villes hôtes ainsi que des villes de départ des camions.
- **E** : Matrice qui va reprendre plusieurs informations pour chaque route.
- **troncons_villes** : Matrice qui relie les villes aux tronçons, l'élément i,j vaut le numéro de la route entre i et j .

D.2 Détermination des trajets

Matlab va d'abord calculer le chemin le plus court entre les villes. Une fois en minimisant le temps de parcours, une fois en minimisant la distance. Pour cela il a besoin de la matrice d'adjacence **A**.

Après cela, il va appeler la fonction `liste_trajet`. Cette fonction détermine une ville de départ, une ville d'arrivée ainsi qu'une heure de départ pour tous les paquets de véhicules.

Création des distributions

La fonction simule 1,6 millions de véhicules, qui font chacun un trajet aller et retour. 8% de ces véhicules sont des camions. Les variables **n_inc_voit** et **n_inc_cam** sont respectivement le nombre de paquets de voitures et de camions simulé.

La ville de départ se détermine en fonction de la population, pour plus d'informations, se référer au chapitre 5. Pour déterminer la ville d'arrivée, la distribution prend en compte la population et la distance.

Ensuite il y a une distribution pour déterminer l'heure de départ. Les valeurs du vecteur **a** proviennent d'un graphe de Beldam. Pour avoir une distribution qui associe une valeur à chaque pas de temps, une extrapolation des données est effectuée.

Assignation des trajets

Pour chaque paquet de véhicules, la fonction va déterminer une ville de départ, une ville d'arrivée et une heure de départ pour les trajets aller et retour.

L'heure de départ du retour doit être séparée de l'heure de départ d'au moins le temps de trajet théorique. En suivant la même procédure, l'assignation des camions est effectuée.

Assignation de camions selon les flux de trafic de marchandises

Ensuite, des trajets de camions sont ajoutés en fonction des grands flux de marchandises. L'heure de départ est calculée selon la même distribution que Beldam. On ne considère pas ces trajets comme allers-retours, car en général les camions traversent les frontières, donc il n'est pas utile de déterminer une heure de retour.

Mise en place des matrices de sortie

Enfin, Matlab va trier les trajets selon les heures de départ, pour que ce soit plus facile pour la suite.

D.3 Simulation temporelle sans fermeture

La variable **T_depart** va recevoir une valeur supplémentaire supérieure à la valeur de fin de la simulation. Cela permet d'éviter les bugs lors de la boucle `while` qui détermine quel trajet va rentrer sur le réseau. Le premier terme est assigné à $dt/2$, pour s'assurer qu'il y ait toujours au moins un véhicule sur le réseau. Toujours pour éviter les bugs.

La variable **time** est la ligne du temps.

La fonction `simulation_temporelle` va être appelée. Cette fonction calcule la position des véhicules sur le réseau à chaque pas de temps. Elle renvoie les heures d'arrivée des véhicules, la vitesse moyenne sur chaque tronçon et à chaque pas de temps.

Détermination des parcours

Pour éviter de devoir lire une cellule pour chaque paquet de véhicules, on va tout d'abord déterminer la liste des noeuds du chemin le plus court entre chaque ville de départ. Chaque ligne de la matrice **M** traite d'un parcours, le premier élément contient le nombre de noeuds parcourus, le reste contient les différents noeuds du chemin le plus court. La matrice **path_ville** associe la ligne de **M** aux villes de départ.

Première itération de la simulation temporelle

positions_v : Chaque ligne correspond à un paquet de véhicules, il y a quatre informations :

- 1er élément : numéro du tronçon en cours
- 2eme élément : distance parcourue sur le tronçon en cours
- 3eme élément : nombre de villes déjà traversées
- 4eme élément : nombre de noeuds que contient le parcours

Lors de la première itération, toutes les informations sont mis-à-jour. Après, les valeurs vont changer lorsque les paquets seront sur le réseau. La même chose est faite pour les camions.

coef_v correspond à la diminution de vitesse de chaque sous-tronçon. La variable **q** correspond aux flux.

Simulation temporelle

La simulation temporelle va commencer, elle effectue la même opération pour chaque pas de temps **dt**.

Tout d'abord elle va lister tous les véhicules qui rentrent sur le réseau. Ce sont ceux qui ont une heure de départ comprise entre **t-dt** et **t**. Cette liste s'ajoute aux paquets qui sont déjà dans la variable **parcours_en_cours_v**.

Ensuite, pour chaque paquet de véhicules présent sur le réseau, le code va calculer la vitesse, en fonction de la densité de voiture du sous-tronçon suivant. Il y a trois cas de figures :

- Le sous-tronçon suivant se trouve sur le même tronçon. Il suffit donc de prendre le coefficient de diminution de vitesse sur le sous-tronçon suivant.
- Le paquet se situe sur le dernier sous-tronçon. Dans ce cas, on prends le coefficient de diminution de vitesse sur le tronçon suivant.
- Le paquet de véhicules est sur le dernier sous-tronçon de son trajet, et donc il va à la vitesse maximale (on considère qu'il prend la sortie si la route est encombrée).

Ensuite Matlab va calculer la distance parcourue, ici aussi trois cas sont possibles :

- Le paquet de véhicules ne change pas de tronçon. On met uniquement à jour la distance parcourue sur le tronçon (2ème colonne de **positions_v**).
- Le paquet de véhicule va changer de tronçon. Dans ce cas, on met à jour les trois premières colonnes de **positions_v**.
- Le paquet a fini son parcours et il quitte le réseau. Il est noté dans la matrice **a_supprimer_prov** pour être enlevé des parcours.

Aussi, pour chaque paquet, sa position dans le réseau a été enregistré dans la matrice **troncons_v** pour pouvoir calculer la densité.

Ensuite, les parcours terminés sont supprimés de la variable **parcours_en_cours_v**. Le morceau de code est long mais est plus rapide que la méthode simple.

La même chose est ensuite faite pour les camions.

Pour certains moments définis par la variable **t_image**, une image de l'état du réseau va être dessinée. Pour cela, le code appelle la fonction **image_reseau**.

Calcul de la diminution de vitesse

Comme dit précédemment, la diminution de vitesse va être fonction de la densité. Attention, ici, $\frac{q}{\rho}$ donne directement le coefficient de diminution de vitesse (et pas la vitesse). La fonction q ne correspond pas exactement au flux tel que défini précédemment.

D.4 Simulation temporelle avec fermeture

L'utilisateur écrit les tronçons qu'il souhaite voir fermés, en respectant le format spécifié dans les commentaires. Il est néanmoins conseillé de ne fermer qu'un seul tronçon.

Matlab va ensuite recalculer les matrices qui ont été modifiées par la fermeture. Ensuite, les parcours les plus courts vont être calculés par Dijkstra. Enfin, On va refaire appel à la fonction **simulation_temporelle**.

D.5 Analyse

Recherche des parcours impactés

Matlab va tout d'abord rechercher pour quelles villes de départ et d'arrivée les véhicules sont impactés. L'élément i,j matrice **path_impact** vaut 1 si le parcours entre la ville i et la ville j est impacté par la fermeture, 0 sinon.

Recherche des véhicules impactés

Matlab va lister tout les paquets de voitures et de camions qui sont impactés par la fermeture dans les vecteurs **impact_v** et **impact_cam**

Calcul du temps de parcours

Le modèle va calculer le temps de parcours tous les véhicules parvenus à destination à la fin de la simulation. Puis il va regarder le temps supplémentaire de parcours suite à la fermeture.

Annexe E

Code Matlab

E.1 Script principal

main.m

```

1  clear all
2  close all
3
4  %% Initialisation
5
6  % Fichier input
7  xyv=dlmread('xyv.txt');
8  V=dlmread('Vitesse.txt');
9  Bande=dlmread('Bande.txt');
10
11 % Choix parametre :
12 choix_poids='temps'; %'temps' ou 'distance', choix de la minimisation
13 pack=100;           % Nombre de vehicules par parquet
14 dt=10;              % Pas de temps
15 dx=1000;           % Pas de distance
16
17 n=length(xyv);      % Nombre de villes
18 % Matrice des distances entre les villes (110% de la longueur a vol
19 %   d'oiseau)
20 D=zeros(n,n);
21 T=zeros(n,n); % Temps theorique entre les villes pour une voiture.
22 k=0;
23 for i=1:n
24     for j=1:n
25         if V(i,j)~=0
26             k=k+1;
27             D(i,j)=distkm([xyv(i,1) xyv(i,2)],[xyv(j,1) xyv(j,2)])...
28                 *1000*1.1; % En m
29             T(i,j)=D(i,j)/V(i,j)/1000*60*60;
30         % En secondes
31     end
32 end

```

```

32 end
33 n_route_init=k/2;           % Nombre de routes avant fermeture
34 n_route=n_route_init;      % Nombre de routes
35 Ville = [];                % Villes hotes
36
37
38
39 for i=1:n
40     if xyv(i,3)~=0
41         Ville = [Ville i];
42     end
43 end
44 Ville_d=sort([Ville 4 149 45 83 174 ]); % Ajout des villes
45                                         % de depart des camions
46
47 nv=length(Ville);           % Nombre de villes hotes
48 nvd=length(Ville_d); % Nombre de villes hotes avec les camions
49
50 % creation de la matrice noeud
51 k=1;
52 E=zeros(n_route,10);
53 % Matrice E qui va reprendre plusieurs informations pour chaque route
54 % E(k,1) : longueur du troncon k
55 % E(k,2) : nombre de bandes
56 % E(k,3) : latitude du premier noeud
57 % E(k,4) : longitude du premier noeud
58 % E(k,5) : latitude du second noeud
59 % E(k,6) : longitude du second noeud
60 % E(k,7) : numero du premier noeud
61 % E(k,8) : numero du second noeud
62 % E(k,9) : vitesse pour une voiture
63 % E(k,10) : vitesse pour un camion
64
65 troncons_villes=zeros(n,n);
66 % Troncons qui relient les villes aux troncons, l'element i,j vaut
67 % le numero de la route entre i et j.
68 for i=1:n
69     for j=1:n
70         if V(i,j)~=0
71             E(k,1)=D(i,j);
72             E(k,2)=Bande(i,j);
73             E(k,3)=xyv(i,1);
74             E(k,4)=xyv(i,2);
75             E(k,5)=xyv(j,1);
76             E(k,6)=xyv(j,2);
77             E(k,7)=i ;
78             E(k,8)=j ;

```

```

79         E(k,9)=V(i , j );
80         if V(i , j )==120;
81             E(k,10)=90;
82         else
83             E(k,10)=min(V(i , j ) ,70);
84         end
85
86         troncons_villes(i , j )=k;
87
88         k=k+1;
89     end
90 end
91 end
92
93 % Retient les infos avant la fermeture pour la comparaison
94 D_init=D;
95 E_init=E;
96 troncons_villes_init=troncons_villes ;
97
98 %% Determination des trajets
99
100 A=1.*(T>0); % Matrice d'adjacence , 1 si route , 0 sinon
101
102 % Trajets plus court , calcule une fois en fonction du temps et une
103 % fois en fonction de la distance .
104 [cost_dist ,path_init_dist] = dijkstra(A,D);
105 [cost_time ,path_init_time] = dijkstra(A,T);
106
107 % Determine les trajets
108 [ville_depart ,T_depart ,n_voiture ,ville_depart_cam ,T_depart_cam ,n_cam]=
109
110
111 %% Simulation temporelle sans fermeture
112
113 if strcmp(choix_poids , 'distance ')
114     cost=cost_dist ;
115     path_init=path_init_dist ;
116 elseif strcmp(choix_poids , 'temps ')
117     cost=cost_time ;
118     path_init=path_init_time ;
119 else % Par default
120     cost=cost_time ;
121     path_init=path_init_time ;
122 end
123
124 T_depart=[T_depart;24*60*60*2]; % Evite les bugs lors de la boucle
125 T_depart(1)=dt /2; % On s'assure d'un parcours des la 1ere

```

```

126 % iteration
127
128 T_depart_cam=[T_depart_cam;24*60*60*2];
129 T_depart_cam(1)=dt/2;
130
131 T_start=0; % En heures
132 T_end=24;
133
134 time=dt:dt:(T_end-T_start)*60*60;
135
136 % Pas de fermeture
137 fermeture=[];
138 [ny, f]=size(fermeture);
139
140 % Fonction de simulation temporelle
141 [T_arrive_v, T_arrive_cam, coef_mem_m, all_path_depart, all_path_depart_cam]
142
143 %% Simulation temporelle avec fermeture
144
145 % Matrice des fermetures,
146 % Mettre une fermeture par ligne, par exemple, si on ferme le
147 % troncon entre les noeuds 81 et 82 et le troncon entre
148 % les noeuds 134 et 135, ecrire :
149 % fermeture=[81 82;134 135];
150 % Si pas de fermeture, mettre fermeture=[];
151 fermeture=[81 82;134,135];
152 [ny, f]=size(fermeture);
153
154 for i=1:ny % On ferme la route
155     V(fermeture(i,1),fermeture(i,2))=0;
156     V(fermeture(i,2),fermeture(i,1))=0;
157     D(fermeture(i,1),fermeture(i,2))=0;
158     D(fermeture(i,2),fermeture(i,1))=0;
159     T(fermeture(i,1),fermeture(i,2))=0;
160     T(fermeture(i,2),fermeture(i,1))=0;
161 end
162
163 n_route=n_route_init-ny;
164 k=1;
165 E=zeros(n_route,10);
166
167 troncons_villes=zeros(n,n);
168 for i=1:n
169     for j=1:n
170         if V(i,j)~=0
171             E(k,1)=D(i,j);
172             E(k,2)=Bande(i,j);

```

```

173         E(k,3)=xyv(i,1);
174         E(k,4)=xyv(i,2);
175         E(k,5)=xyv(j,1);
176         E(k,6)=xyv(j,2);
177         E(k,7)=i;
178         E(k,8)=j;
179         E(k,9)=V(i,j);
180         if V(i,j)==120;
181             E(k,10)=90;
182         else
183             E(k,10)=min(V(i,j),70);
184         end
185
186         troncons_villes(i,j)=k;
187         k=k+1;
188     end
189 end
190 end
191
192 A=1.*(T>0); % 1 si route, 0 sinon
193
194 if strcmp(choix_poids,'distance')
195     [cost_dist,path_init_f] = dijkstra(A,D);
196 elseif strcmp(choix_poids,'temps')
197     [cost_time,path_init_f] = dijkstra(A,T);
198 else % Par default
199     [cost_time,path_init_f] = dijkstra(A,T);
200 end
201
202 % Fonction de simulation temporelle
203 [T_arrive_v_f1,T_arrive_cam_f1,coef_mem_m_f1,all_path_depart_f1,all_pat
204
205
206 %% Analyse
207 % Attention, ne marche que pour une fermeture de route
208
209 % 1. Parcours impactes
210 % Liste tout les parcours entre deux villes hotes qui passent
211 % en temps normal par la route fermee
212
213 path_impact=zeros(n);
214 % L'element i,j de la matrice path_impact vaut 1 si le parcours entre
215 % i et j passe par la fermeture, 0 sinon
216
217 for i=Ville_d
218     for j=Ville_d
219         if i~=j

```

```

220         path=cell2mat(path_init(i,j));
221         k=1;
222         l=0;
223         while (k<=length(path)-1)&& (l==0)
224             if path(k)==fermeture(1,1)
225                 if path(k+1)==fermeture(1,2)
226                     path_impact(i,j)=1;
227                     l=1;
228                 end
229             elseif path(k)==fermeture(1,2)
230                 if path(k+1)==fermeture(1,1)
231                     path_impact(i,j)=1;
232                     l=1;
233                 end
234             end
235             k=k+1;
236         end
237     end
238 end
239 end
240
241
242 % 2. Vehicules impactees
243 % Liste tout les paquets de vehicules qui passent en temps normal
244 % par la route fermee
245
246 % Pour les voitures
247
248 impact_prov_v=zeros(n_voiture,1); % Preallocation provisoire
249 j=0;
250 for i=1:n_voiture
251     if path_impact(ville_depart(i,1),ville_depart(i,2))==1
252         j=j+1;
253         impact_prov_v(j)=i;
254     end
255 end
256 n_impact_v=j; % Nombre de voiture impactees
257 impact_v=impact_prov_v(1:n_impact_v); % Liste des voitures impactees
258
259 % Meme chose pour les camions
260 impact_prov_cam=zeros(n_cam,1);
261 j=0;
262 for i=1:n_cam
263     if path_impact(ville_depart_cam(i,1),ville_depart_cam(i,2))==1
264         j=j+1;
265         impact_prov_cam(j)=i;
266     end

```

```

267 end
268 n_impact_cam=j ;
269 impact_cam=impact_prov_cam(1:n_impact_cam);
270
271
272 % 3. Difference de temps de parcours
273
274 % Temps de parcours, et 0 si le paquet de vehicule n'est pas arrive a
275 % destination a la fin de la simulation.
276 T_parcours_v=(T_arrive_v-T_depart(1:end-1)).*(T_arrive_v>0);
277 T_parcours_cam=(T_arrive_cam-T_depart_cam(1:end-1)).*(T_arrive_cam>0);
278 T_parcours_v_f1=(T_arrive_v_f1-T_depart(1:end-1)).*(T_arrive_v_f1>0);
279 T_parcours_cam_f1=(T_arrive_cam_f1-T_depart_cam(1:end-1)).*(T_arrive_ca
280
281 % Difference de temps des vehicules impactes
282 T_diff_impact_v=zeros(n_impact_v,1);
283 T_diff_impact_cam=zeros(n_impact_cam,1);
284
285 T_impact_v=zeros(n_impact_v,1);
286 T_impact_cam=zeros(n_impact_cam,1);
287 T_f_cam=zeros(n_impact_cam,1);
288 T_f_v=zeros(n_impact_v,1);
289
290 % On ne compte dans la moyenne que les paquets ayant termine
291 % leur trajet
292 k_v=0;
293 for i=1:n_impact_v
294     j=impact_v(i);
295     if T_parcours_v_f1(j)~=0 && T_parcours_v(j)~=0
296         k_v=k_v+1;
297         T_diff_impact_v(k_v)=T_parcours_v_f1(j)-T_parcours_v(j);
298         T_impact_v(k_v)=T_parcours_v(j);
299         T_f_v(k_v)=T_parcours_v_f1(j);
300     end
301 end
302
303 k_cam=0;
304 for i=1:n_impact_cam
305     j=impact_cam(i);
306     if T_parcours_cam_f1(j)~=0 && T_parcours_cam(j)~=0
307         k_cam=k_cam+1;
308         T_diff_impact_cam(k_cam)=T_parcours_cam_f1(j)-T_parcours_cam(j);
309         T_impact_cam(k_cam)=T_parcours_cam(j);
310         T_f_cam(k_cam)=T_parcours_cam_f1(j);
311     end
312 end
313

```

```

314 %Temps moyen des vehicules impactes sans fermeture
315 T_moy=mean([ T_impact_v(1:k_v);T_impact_cam(1:k_cam)])/60;
316 %Temps moyen des vehicules impactes avec fermeture
317 T_f=mean([ T_f_v(1:k_v);T_f_cam(1:k_cam)])/60;
318 % Difference de temps entre la fermeture et sans
319 T_diff=mean([ T_diff_impact_v(1:k_v);T_diff_impact_cam(1:k_cam)])/60;
320
321
322 % 4. Difference de distances
323
324 % Distance parcourue par les vehicules impactes
325 D_impact_v=zeros(n_impact_v,2);
326 D_impact_cam=zeros(n_impact_cam,2);
327
328 % Nombre de passages par les vehicules impactes sur chaque route
329 % du reseau
330 Passage=zeros(n);
331 Passage_f1=zeros(n);
332
333 % Voitures
334 for i=1:n_impact_v
335     j=impact_v(i);
336     k=1;
337     while all_path_depart(j,k+1)~= -1
338         D_impact_v(i,1)=D_impact_v(i,1)+D_init(all_path_depart(j,k),all
339         Passage(all_path_depart(j,k),all_path_depart(j,k+1))=Passage(a
340         k=k+1;
341     end
342
343     k=1;
344     while all_path_depart_f1(j,k+1)~= -1
345         D_impact_v(i,2)=D_impact_v(i,2)+D_init(all_path_depart_f1(j,k)
346         Passage_f1(all_path_depart_f1(j,k),all_path_depart_f1(j,k+1))=
347         k=k+1;
348     end
349 end
350
351 % Camions
352 for i=1:n_impact_cam
353     j=impact_cam(i);
354     k=1;
355     while all_path_depart_cam(j,k+1)~= -1
356         D_impact_cam(i,1)=D_impact_cam(i,1)+D_init(all_path_depart_cam
357         Passage(all_path_depart_cam(j,k),all_path_depart_cam(j,k+1))=P
358         k=k+1;
359     end
360

```

```

361     k=1;
362     while all_path_depart_cam_f1(j,k+1)~= -1
363         D_impact_cam(i,2)=D_impact_cam(i,2)+D_init(all_path_depart_cam,
364         Passage_f1(all_path_depart_cam_f1(j,k),all_path_depart_cam_f1(j
365         k=k+1;
366     end
367 end
368
369 % Distance moyenne sans fermeture , en km
370 D_moy=(mean([D_impact_v(:,1);D_impact_cam(:,1)]))/1000;
371 % Difference de distance
372 D_diff=(mean([D_impact_v(:,2)-D_impact_v(:,1);D_impact_cam(:,2)-D_impact_v(:,1)]))/1000;
373
374 % Emissions de CO2 en Tonne
375 conso_voiture=0.130/1000; % 130 gramme/kilometre
376 conso_camion=1/1000; % 1 kg/kilometre
377 CO2=(sum(D_impact_v(:,1))*conso_voiture+sum(D_impact_cam(:,1))*conso_camion);
378 CO2_f=(sum(D_impact_v(:,2))*conso_voiture+sum(D_impact_cam(:,2))*conso_camion);
379 CO2_diff=CO2_f-CO2;
380
381
382 % 5. Dessin
383
384 % Etat initial
385 figure
386
387 % Affichage des villes hotes
388 for i = 1:nv
389     j=Ville(i);
390     scatter(xyv(j,1),xyv(j,2),70,'b','filled');
391     hold on;
392 end
393
394 % Echelle
395 ecart_x=(1.2-abs(xyv(fermeture(1,1),1)-xyv(fermeture(1,2),1)))/2;
396 ecart_y=(0.45-abs(xyv(fermeture(1,1),2)-xyv(fermeture(1,2),2)))/2;
397 x1=min(xyv(fermeture(1,1),1),xyv(fermeture(1,2),1))-ecart_x;
398 x2=max(xyv(fermeture(1,1),1),xyv(fermeture(1,2),1))+ecart_x;
399 y1=min(xyv(fermeture(1,1),2),xyv(fermeture(1,2),2))-ecart_y;
400 y2=max(xyv(fermeture(1,1),2),xyv(fermeture(1,2),2))+ecart_y;
401 % Si on veut un zoom sur la situation locale
402 % Pour avoir sur l'ensemble de la Belgique, commenter la ligne
403 % suivante
404 axis([x1 x2 y1 y2]);
405
406 linecolors = hot(500); %couleur
407 for k=1:2*n_route

```

```

408     % On dessine toutes les routes en noir
409     plot([E(k,5),E(k,3)], [E(k,6),E(k,4)], 'color', linecolors(1,:));
% creation du rendu
410     hold on;
411 end
412
413 Passage(fermeture(1,1), fermeture(1,2))=0;
414 Passage(fermeture(1,2), fermeture(1,1))=0;
415 maxi=max(max(max(Passage)), max(max(Passage_f1))); % Valeur max de passa
416 for i=1:n
417     for j=1:n
418         if Passage(i, j)~=0
419             a=round(Passage(i, j)/maxi*300+100); % Donne de l'importance
420             b=Passage(i, j)/maxi*2.5+2.5;         % plus de passage
421             plot([xyv(i,1),xyv(j,1)], [xyv(i,2),xyv(j,2)], 'color', lineco
% creation du rendu
422                 hold on;
423             end
424         end
425     end
426
427 % On dessine la fermeture en vert
428 plot([xyv(fermeture(1,1),1),xyv(fermeture(1,2),1)], [xyv(fermeture(1,1)
429 hold on;
430 title(['Situation_du_trafic_passant_par_les_troncons_', num2str(troncon
431
432
433 % 2eme figure apres fermeture :
434 % Meme raisonnement
435 figure
436
437 for i = 1:nv
438     j=Ville(i);
439     scatter(xyv(j,1), xyv(j,2), 70, 'b', 'filled');
440     hold on;
441 end
442
443 axis([x1 x2 y1 y2]); % forcer l'echelle si zoom
444
445 linecolors = hot(500); %couleur
446 for k=1:2*n_route
447     % On dessine toute les routes en noir
448     plot([E(k,5),E(k,3)], [E(k,6),E(k,4)], 'color', linecolors(1,:));
449     hold on;
450 end
451
452

```

```

453 for i=1:n
454     for j=1:n
455         if Passage_f1(i,j)~=0
456             a=round(Passage_f1(i,j)/maxi*300+100);
457             b=Passage(i,j)/maxi*2.5+2.5;
458             plot([xyv(i,1),xyv(j,1)],[xyv(i,2),xyv(j,2)], 'color',linecolor)
459 % creation du rendu
459             hold on;
460         end
461     end
462 end
463
464 plot([xyv(fermeture(1,1),1),xyv(fermeture(1,2),1)],[xyv(fermeture(1,1),2),xyv(fermeture(1,2),2)], 'color',linecolor)
465 hold on;
466 title(['Situation_apres_la_fermeture_des_troncons_',num2str(troncons_v)])
467
468
469 % 6. Affichage des resultats
470
471 Nombre_de_voiture_impacte=n_impact_v/2*pack
472 Nombre_de_camion_impacte=n_impact_cam/2*pack
473
474 T_moy          % En minutes
475 T_diff
476
477 D_moy          % En kilometres
478 D_diff
479
480 CO2             % En tonnes
481 CO2_diff

```

E.2 liste_trajet.m

```

1 function [ville_depart,T_depart,n_voiture,ville_depart_cam,T_depart_cam] = liste_trajet(xyv,IN,Ville,cost_time)
2 % Cette fonction determine une ville de depart, une ville d'arrivee
3 % ainsi qu'une heure pour tous les paquets de vehicules. Les
4 % donnees concernant les camions finissent par _cam.
5 %
6 % IN : - Matrice xyv qui reprend l'ensemble des villes ainsi que
7 %       leur population
8 %       - Ville : Matrice reprenant le numero des villes hotes
9 %       - cost_time : duree de trajet entre deux noeuds
10 %
11 % OUT : - ville_depart : la premiere colonne reprend le numero de

```

```

12 %          la ville de depart, la deuxieme colonne reprend
13 %          le numero de la ville d'arrivee
14 %      - T_depart : matrice colonne qui reprend les heures
15 %          de depart, en secondes
16 %      - n_voiture : nombre de voitures
17 %      - n_cam : nombre de camions
18
19
20 n_sim=1600000; % Nombre de vehicules simules, chaque vehicule
21 %             effectue un trajet aller et retour
22 ratio_camion=0.08; % Pourcentage de camion
23 n_inc_voit=round(n_sim*(1-ratio_camion)/pack); % Nombre de paquets
24 n_inc_cam=round(n_sim*ratio_camion/pack); % Nombre de paquets
25
26 %% Creation des distributions pour determiner les villes hotes
27 % Distribution de la population, plus une ville a d'habitants, plus
28 % elle a de chance d'etre choisie.
29 j=0;
30 pop=zeros(nv,1);
31 for i=Ville
32     j=j+1;
33     pop(j)=xyv(i,3);
34 end
35 pop_tot=sum(pop);
36 pop(1)=pop(1)/pop_tot;
37 for i=2:j
38     pop(i)=pop(i-1)+pop(i)/pop_tot;
39 end
40
41 % Distribution de la distance, le poids donne a chaque ville vaut
42 % l'inverse de la distance au cube. Une ville deux fois plus proche
43 % a deux fois plus de poids.
44
45 distri_dist=zeros(nv);
46 distri_init=zeros(nv);
47
48 for i=1:nv
49     poids_tot=0;
50     for j=[1:i-1,i+1:nv]
51         distri_dist(i,j)=1/(cost_time(Ville(i),Ville(j)))^3;
52         poids_tot=poids_tot+distri_dist(i,j);
53     end
54     distri_dist(i,1)=distri_dist(i,1)/poids_tot;
55     for j=2:nv
56         distri_dist(i,j)=distri_dist(i,j-1)+distri_dist(i,j)/poids_tot;
57     end
58 end

```

```

59
60 % Distribution final
61 lambda=0.25;
62 for i=1:nv
63     distri_init(i,:)=lambda*pop'+(1-lambda)*distri_dist(i,:);
64 end
65
66
67 %% Determination de l'heure de depart
68
69 % Creation de la distribution des horaires
70 % Le vecteur a reprend le poids donne a chaque heure
71 a=[0.5 0.2 0.2 0.2 0.2 0.3 0.8 2.2 9 10 5.4 5.8 5.6 6.2 5.6 5.8 8 9.8
72
73 n_s=60*60/dt;
74 a_1=zeros(24*n_s,1);
75 a_2=zeros(24*n_s,1);
76 for i=1:24
77     for j=1:n_s
78         % Le poids donne a chaque pas de temps est une interpolation
79         % des poids de l'heure d'avant et de l'heure d'apres
80         a_1((i-1)*n_s+j)=a(i)+(a(i+1)-a(i))*(j-1)/n_s;
81     end
82 end
83
84 a_2(1)=a(1);
85 for j=2:24*n_s
86     a_2(j)=a_2(j-1)+a_1(j);
87 end
88
89 % a_3 : distribution finale
90 a_3=a_2./a_2(end);
91
92 %% Preallocation des matrices
93
94 T_depart=zeros(2*n_inc_voit,1);
95 T_depart_cam1=zeros(2*n_inc_cam,1);
96
97 % Elles sont marquees comme provisoires car elles vont etre trieés par
98 % heure de depart
99 villes_depart_prov=zeros(2*n_inc_voit,2);
100 ville_depart_prov_cam1=zeros(2*n_inc_cam,2);
101
102 %% Assignment des voitures
103 for inc=1:n_inc_voit
104
105     % Selection de la ville de depart

```

```
106     a=rand(1);
107
108     i=1;
109     while pop(i)<=a
110         i=i+1;
111     end
112
113     villes_depart_prov(2*inc-1,1)=Ville(i); % Trajet aller
114     villes_depart_prov(2*inc,2)=Ville(i); % Trajet retour
115
116     % Ville d'arrivee
117
118     tarte=distri_init(i,:);
119
120
121     j=i;
122     while i==j
123         b=rand(1);
124         j=1;
125         while tarte(j)<=b
126             j=j+1;
127         end
128     end
129
130     villes_depart_prov(2*inc-1,2)=Ville(j);
131     villes_depart_prov(2*inc,1)=Ville(j);
132
133     % Heure depart
134
135     ran=rand(1);
136     j=1;
137     while a_3(j)<ran
138         j=j+1;
139     end
140
141     t1=j*dt;
142     t2=t1;
143
144     temps_min=cost_time(villes_depart_prov(2*inc-1,1),villes_depart_prov(2*inc,2));
145
146     % L'heure du retour doit etre separee d'au moins le temps de trajet
147     % theorique
148     while abs(t2-t1)<temps_min
149         ran=rand(1);
150         j=1;
151         while a_3(j)<ran
152             j=j+1;
```

```
153         end
154         t2=j*dt;
155     end
156
157     T_depart(2*inc-1)=t1;
158     T_depart(2*inc)=t2;
159 end
160
161 %% Camion
162 % Ce suit le meme raisonnement
163 for inc=1:n_inc_cam
164
165     % Ville de depart
166     a=rand(1);
167
168     i=1;
169     while pop(i)<=a
170         i=i+1;
171     end
172
173     ville_depart_prov_cam1(2*inc-1,1)=Ville(i);
174     ville_depart_prov_cam1(2*inc,2)=Ville(i);
175
176     % Ville d'arrivee
177     tarte=distri_init(i,:);
178
179     j=i;
180     while i==j
181         b=rand(1);
182         j=1;
183         while tarte(j)<=b
184             j=j+1;
185         end
186     end
187
188     ville_depart_prov_cam1(2*inc-1,2)=Ville(j);
189     ville_depart_prov_cam1(2*inc,1)=Ville(j);
190
191     % Heure
192     ran=rand(1);
193     j=1;
194     while a_3(j)<ran
195         j=j+1;
196     end
197
198     t1=j*dt;
199     t2=t1;
```

```

200
201     temps_min=cost_time(ville_depart_prov_cam1(2*inc-1,1),ville_depart_
202
203     while abs(t2-t1)<temps_min
204         ran=rand(1);
205         j=1;
206         while a_3(j)<ran
207             j=j+1;
208         end
209         t2=j*dt;
210     end
211
212     T_depart_cam1(2*inc-1)=t1;
213     T_depart_cam1(2*inc)=t2;
214 end
215
216 %% Assignation de camions selon les flux de trafic de marchandises
217
218 j=2000/pack;
219 all_ville_depart_prov_cam2=zeros(18*j,2);
220 T_depart_cam2=zeros(18*j,1);
221
222 all_ville_depart_prov_cam2(1:1*j,1)=149;
223 all_ville_depart_prov_cam2(1:1*j,2)=4;
224 all_ville_depart_prov_cam2(1*j+1:2*j,1)=4;
225 all_ville_depart_prov_cam2(1*j+1:2*j,2)=149;
226
227 all_ville_depart_prov_cam2(2*j+1:3*j,1)=149;
228 all_ville_depart_prov_cam2(2*j+1:3*j,2)=45;
229 all_ville_depart_prov_cam2(3*j+1:4*j,1)=45;
230 all_ville_depart_prov_cam2(3*j+1:4*j,2)=149;
231
232 all_ville_depart_prov_cam2(4*j+1:6*j,1)=83;
233 all_ville_depart_prov_cam2(4*j+1:6*j,2)=174;
234 all_ville_depart_prov_cam2(6*j+1:8*j,1)=174;
235 all_ville_depart_prov_cam2(6*j+1:8*j,2)=83;
236
237 all_ville_depart_prov_cam2(8*j+1:13*j,1)=2;
238 all_ville_depart_prov_cam2(8*j+1:13*j,2)=45;
239 all_ville_depart_prov_cam2(13*j+1:end,1)=45;
240 all_ville_depart_prov_cam2(13*j+1:end,2)=2;
241
242 % Heure de depart
243 % Pas necessaire d'imposer une contrainte sur l'heure de retour
244 % (car les camions font juste l'aller)
245 for i=1:18*j
246     ran=rand(1);

```

```

247     k=1;
248     while a_3(k)<ran
249         k=k+1;
250     end
251
252     T_depart_cam2(i)=k*dt;
253 end
254
255
256 %% Mise en place des matrices de sortie ( OUT )
257 % Les matrices sont trieés selon l'heure de depart
258
259 % Voiture
260 C=sortrows([ T_depart villes_depart_prov ],1);
261 ville_depart=C(:,2:end);
262 T_depart=C(:,1);
263
264 n_voiture=2*n_inc_voit;
265
266 % Camions
267 T_depart_cam=[T_depart_cam1; T_depart_cam2];
268 ville_depart_prov_cam=[ville_depart_prov_cam1; all_ville_depart_prov_ca
269 C=sortrows([ T_depart_cam ville_depart_prov_cam ],1);
270 ville_depart_cam=C(:,2:end);
271 T_depart_cam=C(:,1);
272
273 n_cam=length(T_depart_cam);
274
275 end

```

E.3 simulation_temporelle.m

```

1 function [T_arrive_v , T_arrive_cam , coef_mem_m , all_path_depart , all_path_d
2 % Cette fonction calcule la position des vehicules sur le reseau
3 % a chaque pas de temps. Elle renvoie les heures d'arrivee
4 % des vehicules , la vitesse moyenne sur chaque troncon
5 % et a chaque pas de temps.
6 %
7 % IN :  – Pack , dt , dx : les parametres du modele.
8 %      – Ville : Matrice reprenant le numero des villes hotes.
9 %      – Ville_d : Les numeros des villes hotes + les villes de
10 %      depart des camions.
11 %      –ville_depart , ville_depart_cam : matrice contenant les
12 %      villes de depart et d'arrivee de tous les vehicules.

```

```

13 %      – path_init : matrice des parcours plus rapides calculés par
14 %          Dijkstra
15 %      – troncons_ville : matrice qui donne le numero de troncon
16 %      entre 2 villes .
17 %      – T_depart , T_depart_cam : heure de depart des vehicules .
18 %      – time : ligne du temps .
19 %
20 % OUT : – T_arrive_v , T_arrive_v_cam : heure d'arrivee des vehicules .
21 %      – coef_mem_m : coeficient de diminution de vitesse .
22 %      – all_path_depart , all_path_depart_cam : liste des noeuds
23 %      parcourus par les vehicules .
24
25
26 %% Determination des parcours
27
28 % Preallocation des heures d'arrivee
29 T_arrive_v=zeros(n_voiture ,1);
30 T_arrive_cam=zeros(n_cam ,1);
31
32 % Preallocation des trajets
33 l_max=33;
34 all_path_depart=-1*ones(n_voiture ,l_max);
35 all_path_depart_cam=-1*ones(n_cam ,l_max);
36
37 % Determination du trajet le plus court entre 2 villes
38 % Matrice M : le 1er element est la longueur du parcours et puis le
39 %   parcours .
40 path_ville=zeros(n);
41 M=zeros(nvd^2 ,30);
42 k=1;
43 for i=1:nvd
44     for j=1:nvd
45         path=cell2mat(path_init(Ville_d(i) , Ville_d(j)));
46         M(k,1)=length(path);
47         M(k,2:length(path)+1)=path;
48         path_ville(Ville_d(i) , Ville_d(j))=k;
49         k=k+1;
50     end
51 end
52
53 %% Simulation temporelle
54
55 % Nombre max de sous-troncons
56 n_dx=ceil(max(E(:,1))/dx); % = 68 pour dx = 1000
57
58 % Nombre de sous-troncons sur chaque troncon .
59 size_troncons=ceil(E(:,1)/dx);

```

```

60
61 % Positions_v , Position_cam :
62 %   - 1er element : numero du troncon en cours
63 %   - 2eme element : distance parcourue sur le troncon en cours
64 %   - 3eme element : nombre de ville parcourues
65 %   - 4eme element : nombre de noeuds que contient le parcours
66 positions_v=zeros(n_voiture,4);
67 positions_cam=zeros(n_cam,4);
68
69 %Premiere iteration
70 for i=1:n_voiture
71     k=path_ville(ville_depart(i,1),ville_depart(i,2));
72     path=M(k,2:M(k,1)+1);
73
74     positions_v(i,1)=troncons_villes(path(1),path(2));
75     positions_v(i,4)=length(path);
76
77     all_path_depart(i,1:M(k,1))=path;
78 end
79 positions_v(:,3)=1;
80 positions_v(:,2)=0.1; % Distance initiale , pour que les vehicules
81 %                     soient deja sur le reseau. Permet d'eviter les bugs.
82
83 % Meme chose pour les camions.
84 for i=1:n_cam
85     k=path_ville(ville_depart_cam(i,1),ville_depart_cam(i,2));
86     path=M(k,2:M(k,1)+1);
87
88     positions_cam(i,1)=troncons_villes(path(1),path(2));
89     positions_cam(i,4)=length(path);
90
91     all_path_depart_cam(i,1:M(k,1))=path;
92 end
93 positions_cam(:,3)=1; % Nombre de villes parcourus
94 positions_cam(:,2)=0.1;
95
96 % Coefficient de diminution de vitesse , pour chaque sous-troncon.
97 coef_v=ones(n_route*2,n_dx);
98
99 % Parametres de la gestion des embouteillages
100 rho_max=140; % =7,14m par voiture
101 rho_crit=40; % =33,33m par voiture
102 q_crit=rho_crit;
103 q=@(x) x.*(x<=rho_crit).*(x>0) + (q_crit-q_crit.*(x-rho_crit)./(rho_max-
104
105 parcours_en_cours_v=[]; % Reprend toutes les voitures en route
106 parcours_en_cours_cam=[]; % Reprend tous les camions en route

```

```

107 count_apd_v=1;
108 count_apd_cam=1;
109
110 % Matrice OUT qui retient les embouteillages
111 coef_mem_m=zeros(length(time),2*n_route);
112
113 t=0;
114 parcours_en_cours_prov=zeros(1,500000/pack);
115 a_supprimer_prov=zeros(5000/pack,1);
116
117 % Nombre de pas de temps entre chaque image du reseau
118 % Si pas envie d'images, mettre une valeur superieure a 8640
119 t_image=500;
120
121 for t_inc=1:length(time)
122     t=t+dt;
123
124     % Voitures
125     n_par=0;
126     % T_depart est trie, et tous les parcours qui commencent au temps t
127     % arrivent sur le reseau.
128     while T_depart(count_apd_v)<=t
129         n_par=n_par+1;
130         parcours_en_cours_prov(n_par)=count_apd_v;
131
132         count_apd_v=count_apd_v+1;
133     end
134     % Liste de toutes les voitures sur le reseau au temps t.
135     parcours_en_cours_v=[parcours_en_cours_v parcours_en_cours_prov(1,
136
137
138     n_all_path=0;
139     % Au cas o il n'y a aucun vehicule.
140     try
141         n_all_path=length(parcours_en_cours_v);
142     end
143
144     n_sup=0;
145
146     % Nombre de voitures sur les differents sous-troncons
147     troncons_v=zeros(n_route*2,n_dx);
148
149     for j=1:n_all_path
150         i=parcours_en_cours_v(j);
151
152         % coef_v : coefficient de diminution de vitesse
153         % Determination de coef_v en regardant la densite sur le

```

```

154      % sous-troncon suivant. S'il est a la fin du troncon, il
155      % regarde sur le troncon suivant.
156      if ceil(positions_v(i,2)/dx)+1<=size_troncons(positions_v(i,1))
157          coef_v_i=coef_v(positions_v(i,1),ceil(positions_v(i,2)/dx));
158
159      elseif positions_v(i,3)+1<positions_v(i,4)
160          coef_v_i=coef_v(troncons_villes(all_path_depart(i,positions_v(i,1)),positions_v(i,2)/dx));
161
162      else
163          coef_v_i=1; % En fin de trajet
164      end
165
166      vitesse=E(positions_v(i,1),9)*coef_v_i;
167      dist_parcouru=vitesse/3.6*dt;
168
169      if positions_v(i,2)+dist_parcouru<=E(positions_v(i,1),1);
170          % S'il reste sur le meme troncon
171
172          positions_v(i,2)=positions_v(i,2)+dist_parcouru;
173          troncons_v(positions_v(i,1),ceil(positions_v(i,2)/dx))=troncons_villes(all_path_depart(i,positions_v(i,1)),positions_v(i,2)/dx);
174
175      elseif positions_v(i,3)+1<positions_v(i,4)
176          % S'il change de troncon
177
178          positions_v(i,3)=positions_v(i,3)+1;
179          positions_v(i,2)=positions_v(i,2)+dist_parcouru-E(positions_v(i,1),1);
180          positions_v(i,1)=troncons_villes(all_path_depart(i,positions_v(i,1)),positions_v(i,2)/dx);
181          troncons_v(positions_v(i,1),ceil(positions_v(i,2)/dx))=troncons_villes(all_path_depart(i,positions_v(i,1)),positions_v(i,2)/dx);
182
183      else
184          % S'il a fini, il est supprime de la liste
185          n_sup=n_sup+1;
186          a_supprimer_prov(n_sup,1)=j;
187          T_arrive_v(i)=t;
188
189      end
190
191  end
192
193  a_supprimer=a_supprimer_prov(1:n_sup,1);
194
195  % Pour supprimer les elements a supprimer de parcours_en_cours_v
196  % Un peu long mais mieux optimise
197  j=1;
198  la=length(a_supprimer);
199  new_parcours=zeros(1,n_all_path-la);
200  for i=1:n_all_path

```

```

201         if j <= la
202             if i ~ a_supprimer(j)
203                 new_parours(i-j+1) = parours_en_cours_v(i);
204             else
205                 j = j + 1;
206             end
207         else
208             new_parours(i-j+1) = parours_en_cours_v(i);
209         end
210     end
211     parours_en_cours_v = new_parours;
212
213     % Meme chose pour les camions:
214
215     troncons_cam = zeros(n_route*2, n_dx);
216     n_par = 0;
217     while T_depart_cam(count_apd_cam) <= t
218
219         n_par = n_par + 1;
220         parours_en_cours_prov(1, n_par) = count_apd_cam;
221
222         count_apd_cam = count_apd_cam + 1;
223
224     end
225     parours_en_cours_cam = [parours_en_cours_cam parours_en_cours_prov];
226     n_all_path_cam = 0;
227     try
228         n_all_path_cam = length(parours_en_cours_cam);
229     end
230
231     n_sup = 0;
232
233     for j = 1:n_all_path_cam
234         i = parours_en_cours_cam(j);
235
236         % Calcul de coef_v
237         if ceil(positions_cam(i, 2)/dx) + 1 <= size_troncons(positions_cam(i, 2))
238             coef_v_i = coef_v(positions_cam(i, 1), ceil(positions_cam(i, 2)/dx));
239         elseif positions_cam(i, 3) + 1 < positions_cam(i, 4)
240             coef_v_i = coef_v(troncons_villes(all_path_depart_cam(i, positions_cam(i, 2))), positions_cam(i, 3));
241         else
242             coef_v_i = 1; % dernier parours
243         end
244
245         vitesse = E(positions_cam(i, 1), 10) * coef_v_i;
246         dist_parcouru = vitesse / 3.6 * dt;
247

```

```

248
249     if positions_cam(i,2)+dist_parcouru<=E(positions_cam(i,1),1);
250         positions_cam(i,2)=positions_cam(i,2)+dist_parcouru;
251         troncons_cam(positions_cam(i,1),ceil(positions_cam(i,2)/dx);
252
253     elseif positions_cam(i,3)+1<positions_cam(i,4)
254         positions_cam(i,2)=positions_cam(i,2)+dist_parcouru-E(
255         positions_cam(i,3)=positions_cam(i,3)+1;
256         positions_cam(i,1)=troncons_villes(all_path_depart_cam
257
258         troncons_cam(positions_cam(i,1),ceil(positions_cam(i,2)
259     else
260
261         n_sup=n_sup+1;
262         a_supprimer_prov(n_sup,1)=j;
263         T_arrive_cam(i)=t;
264     end
265
266 end
267 a_supprimer=a_supprimer_prov(1:n_sup,1);
268
269 % Camions a supprimer
270 j=1;
271 la=length(a_supprimer);
272 new_parcours=zeros(1,n_all_path_cam-la);
273 for i=1:n_all_path_cam
274     if j<=la
275         if i~=a_supprimer(j)
276             new_parcours(i-j+1)=parcours_en_cours_cam(i);
277         else
278             j=j+1;
279         end
280     else
281         new_parcours(i-j+1)=parcours_en_cours_cam(i);
282     end
283 end
284
285 parcours_en_cours_cam=new_parcours;
286
287
288 % Tout les x pas de temps, le programme cree une image
289 if mod(t_inc,t_image)==0
290
291     coef_im=coef_mem_m(t_inc-1,:);
292     image_reseau(n_route,nv,Ville,ny,coef_im,E,xyv,t,fermeture)
293
294 end

```

```

295
296 % Calcule le coefficient de diminution de vitesse pour chaque
297 % sous-troncon.
298 for i=1:n_route*2
299     j=1;
300     while j<=size_troncons(i);
301
302         % densite , les camions comptent pour 2,5 voitures.
303         % Les derniers sous-troncons ont une distance inferieure
304         % a dx, mais qui n'est pas prise en compte.
305         % E(i,2)=nombre de bande du troncon i.
306         rho=(troncons_v(i,j)+2.5*troncons_cam(i,j))/dx/E(i,2)*1000
307
308         if rho==0
309             % Si pas de vehicule
310             coef_v(i,j)=1;
311         else
312             % Vitesse minimale : 10% de la vitesse autorisee
313             coef_v(i,j)=max(q(rho)/rho,0.1);
314         end
315
316         j=j+1;
317     end
318 end
319
320 % Retient la diminution de vitesse.
321 for i=1:2*n_route
322     coef_mem_m(t_inc,i)=mean(coef_v(i,1:size_troncons(i)));
323 end
324 1;
325
326
327 end
328 end

```

E.4 image_reseau.m

```

1 function []=image_reseau(n_route,nv,Ville,ny,coef_im,E,xyv,t,fermeture)
2 % Cette fonction cree une image des embouteillages presents dans
3 % le reseau Chaque troncon a deux valeurs (une dans chaque
4 % sens de circulation). Matlab dessine seulement le deuxieme.
5
6 figure
7

```

```

8      % Dessin des villes hotes
9      for i = 1:nv
10         j=Ville(i);
11         s=60+round(xyv(j,3)/2500);
12         scatter(xyv(j,1),xyv(j,2),s,'b','filled');
13         hold on;
14     end
15
16     % Legende
17     c=colorbar('Location','southoutside','Ticks',[0.1,0.5,1],...
18         'TickLabels',{'10%','50%','100%'});
19     c.Label.String = 'Vitesse_moyenne_par_rapport_a_la_vitesse_maximal';
20
21     linecolors = parula(500); %couleur
22
23     % forcer l'echelle
24     set(gca,'xtick',2.5:.5:6.5);
25     set(gca,'ytick',49.4:.2:51.6);
26     axis([2.5 6.5 49.4 51.6]);
27
28     for k=1:2*n_route
29
30         a=round(500*coef_im(k));
31         b=2;
32         i=E(k,7);
33         j=E(k,8);
34
35         plot([xyv(i,1),xyv(j,1)],[xyv(i,2),xyv(j,2)],'color',linecolors);
36     % creation du rendu
37         hold on;
38     end
39
40     % Route fermee dessinee en vert
41     for i=1:ny
42         plot([xyv(fermeture(i,1),1),xyv(fermeture(i,2),1)],[xyv(fermetu
43         hold on;
44     end
45
46     heure=floor(t/60/60);
47     minute=floor(t/60-heure*60);
48     if minute<10
49         title(['Zones_d''embouteillages_en_Belgique_a_',num2str(heure)
50     else
51         title(['Zones_d''embouteillages_en_Belgique_a_',num2str(heure)
52     end
53     pause(0.01) % Pour qu'il affiche l'image

```

54 `end`

E.5 Fonction Dijkstra

La fonction `Dijkstra.m` a été développée par Joseph Kirk.

La fonction prend en entrée la matrice d'incidence, c'est-à-dire la matrice $n \times n$ où l'élément $a_{i,j} = 1$ si il y a une arête entre i et j et $a_{i,j} = 0$. n étant le nombre de noeuds. Elle va également prendre en entrée une matrice $n \times n$ où l'élément $a_{i,j} = d_{i,j}$, où $d_{i,j}$ est le poids de l'arête i - j .

Ella va retourner deux matrices $n \times n$.

La première matrice est la matrice coût. l'élément $c_{i,j}$ vaut le coût du chemin minimum entre i et j .

La deuxième matrice est la matrice des parcours, l'élément $p_{i,j}$ est un vecteur qui contient tout les noeuds traverses par le chemin le plus court entre i et j .

Cette version de l'algorithme de Dijkstra n'étant plus disponible sur *mathworks*, une autre version de la fonction est disponible à l'adresse suivante : <https://nl.mathworks.com/matlabcentral/fileexchange/36140-dijkstra-algorithm>

```

1 % DIJKSTRA Calculate Minimum Costs and Paths using Dijkstra's
2 %       Algorithm
3 % Filename: dijkstra.m
4 %
5 % Description: Given vertices and edge connections for a graph
6 %       (represented by either adjacency matrix or edge list)
7 %       and edge costs that are greater than zero, this function
8 %       computes the shortest path from one or more starting
9 %       nodes to one or more termination nodes.
10 %
11 % Author:
12 %       Joseph Kirk
13 %       jdkirk630@gmail.com
14 %
15 % Date: 02/27/15
16 %
17 % Release: 2.0
18 %
19 % Inputs:
20 %       [AorV] Either A or V where
21 %       A is a NxN adjacency matrix, where A(I,J) is nonzero (=1)
22 %       if and only if an edge connects point I to point J

```

```

23 %           NOTE: Works for both symmetric and asymmetric A
24 %           V   is a Nx2 (or Nx3) matrix of x,y,(z) coordinates
25 % [xyCorE] Either xy or C or E (or E3) where
26 %           xy   is a Nx2 (or Nx3) matrix of x,y,(z) coordinates
27 %           (equivalent to V)
28 %           NOTE: only valid with A as the first input
29 %           C   is a NxN cost (perhaps distance) matrix, where C(I,J)
30 %           contains the value of the cost to move from point
31 %           I to point J
32 %           NOTE: only valid with A as the first input
33 %           E   is a Px2 matrix containing a list of edge connections
34 %           NOTE: only valid with V as the first input
35 %           E3  is a Px3 matrix containing a list of edge connections
36 %           in the first two columns and edge weights in the
37 %           third column
38 %           NOTE: only valid with V as the first input
39 % [SID] (optional) 1xL vector of starting points
40 %       if unspecified, the algorithm will calculate the minimal
41 %       path from all N points to the finish point(s)
42 %       (automatically sets SID = 1:N)
43 % [FID] (optional) 1xM vector of finish points
44 %       if unspecified, the algorithm will calculate the minimal
45 %       path from the starting point(s) to all N points
46 %       (automatically sets FID = 1:N)
47 % [showWaitbar] (optional) a scalar logical that initializes a
48 %       waitbar if nonzero
49 %
50 % Outputs:
51 % [costs] is an LxM matrix of minimum cost values for the
52 %       minimal paths
53 % [paths] is an LxM cell array containing the shortest path arrays
54 %
55 % Note:
56 %       If the inputs are [A,xy] or [V,E], the cost is assumed to be
57 %       (and is calculated as) the point-to-point Euclidean distance
58 %       If the inputs are [A,C] or [V,E3], the cost is obtained
59 %       from either the C matrix or from the edge weights in the
60 %       3rd column of E3
61 %
62 % Usage:
63 % [costs ,paths] = dijkstra(A,xy)
64 %       -or-
65 % [costs ,paths] = dijkstra(A,C)
66 %       -or-
67 % [costs ,paths] = dijkstra(V,E)
68 %       -or-
69 % [costs ,paths] = dijkstra(V,E3)

```

```

70 %           -or-
71 %   [costs ,paths] = dijkstra( ... ,SID,FID)
72 %           -or-
73 %   [costs ,paths] = dijkstra( ... ,SID,FID,true)
74 %
75 % Example:
76 %   % Calculate the (all pairs) shortest distances and paths
77 %       using [A,xy] inputs
78 %   n = 7; A = zeros(n); xy = 10*rand(n,2)
79 %   tri = delaunay(xy(:,1),xy(:,2));
80 %   I = tri(:); J = tri(:,[2 3 1]); J = J(:);
81 %   IJ = I + n*(J-1); A(IJ) = 1
82 %   [costs ,paths] = dijkstra(A,xy)
83 %
84 % Example:
85 %   % Calculate the (all pairs) shortest distances and paths
86 %       using [A,C] inputs
87 %   n = 7; A = zeros(n); xy = 10*rand(n,2)
88 %   tri = delaunay(xy(:,1),xy(:,2));
89 %   I = tri(:); J = tri(:,[2 3 1]); J = J(:);
90 %   IJ = I + n*(J-1); A(IJ) = 1
91 %   a = (1:n); b = a(ones(n,1),:);
92 %   C = round(reshape(sqrt(sum((xy(b,:) - xy(b',:)).^2,2)),n,n))
93 %   [costs ,paths] = dijkstra(A,C)
94 %
95 % Example:
96 %   % Calculate the (all pairs) shortest distances and paths
97 %       using [V,E] inputs
98 %   n = 7; V = 10*rand(n,2)
99 %   I = delaunay(V(:,1),V(:,2));
100 %   J = I(:,[2 3 1]); E = [I(:) J(:)]
101 %   [costs ,paths] = dijkstra(V,E)
102 %
103 % Example:
104 %   % Calculate the (all pairs) shortest distances and paths
105 %       using [V,E3] inputs
106 %   n = 7; V = 10*rand(n,2)
107 %   I = delaunay(V(:,1),V(:,2));
108 %   J = I(:,[2 3 1]);
109 %   D = sqrt(sum((V(I(:),:) - V(J(:),:)).^2,2));
110 %   E3 = [I(:) J(:) D]
111 %   [costs ,paths] = dijkstra(V,E3)
112 %
113 % Example:
114 %   % Calculate the shortest distances and paths from the 3rd point
115 %       to all the rest
116 %   n = 7; V = 10*rand(n,2)

```

```

117 %      I = delaunay(V(:,1),V(:,2));
118 %      J = I(:,[2 3 1]); E = [I(:) J(:)]
119 %      [costs,paths] = dijkstra(V,E,3)
120 %
121 % Example:
122 %      % Calculate the shortest distances and paths from all points
123 %      to the 2nd
124 %      n = 7; A = zeros(n); xy = 10*rand(n,2)
125 %      tri = delaunay(xy(:,1),xy(:,2));
126 %      I = tri(:); J = tri(:,[2 3 1]); J = J(:);
127 %      IJ = I + n*(J-1); A(IJ) = 1
128 %      [costs,paths] = dijkstra(A,xy,1:n,2)
129 %
130 % Example:
131 %      % Calculate the shortest distance and path from points [1 3 4]
132 %      to [2 3 5 7]
133 %      n = 7; V = 10*rand(n,2)
134 %      I = delaunay(V(:,1),V(:,2));
135 %      J = I(:,[2 3 1]); E = [I(:) J(:)]
136 %      [costs,paths] = dijkstra(V,E,[1 3 4],[2 3 5 7])
137 %
138 % Example:
139 %      % Calculate the shortest distance and path between two points
140 %      n = 1000; A = zeros(n); xy = 10*rand(n,2);
141 %      tri = delaunay(xy(:,1),xy(:,2));
142 %      I = tri(:); J = tri(:,[2 3 1]); J = J(:);
143 %      D = sqrt(sum((xy(I,:) - xy(J,:)).^2,2));
144 %      I(D > 0.75,:) = []; J(D > 0.75,:) = [];
145 %      IJ = I + n*(J-1); A(IJ) = 1;
146 %      [cost,path] = dijkstra(A,xy,1,n)
147 %      gplot(A,xy,'k.:'); hold on;
148 %      plot(xy(path,1),xy(path,2),'ro-','LineWidth',2); hold off
149 %      title(sprintf('Distance from %d to %d = %1.3f',1,n,cost))
150 %
151 % Web Resources:
152 %      <a href="http://en.wikipedia.org/wiki/Dijkstra%27s_algorithm">Dijkstra's algorithm
153 %      <a href="http://en.wikipedia.org/wiki/Graph_%28mathematics%29">Graph theory
154 %      <a href="http://en.wikipedia.org/wiki/Adjacency_matrix">Adjacency matrix
155 %
156 % See also: gplot, gplotd, gplotdc, distmat, ve2axy, axy2ve
157 %
158 function [costs,paths] = dijkstra(AorV,xyCorE,SID,FID,showWaitbar)
159
160     narginchk(2,5);
161
162     % Process inputs
163     [n,nc] = size(AorV);

```

```

164     [m,mc] = size(xyCorE);
165     if (nargin < 3)
166         SID = (1:n);
167     elseif isempty(SID)
168         SID = (1:n);
169     end
170     L = length(SID);
171     if (nargin < 4)
172         FID = (1:n);
173     elseif isempty(FID)
174         FID = (1:n);
175     end
176     M = length(FID);
177     if (nargin < 5)
178         showWaitbar = (n > 1000 && max(L,M) > 1);
179     end
180
181
182     % Error check inputs
183     if (max(SID) > n || min(SID) < 1)
184         eval(['help_' mfilename]);
185         error('Invalid_[SID]_input._See_help_notes_above. ');
186     end
187     if (max(FID) > n || min(FID) < 1)
188         eval(['help_' mfilename]);
189         error('Invalid_[FID]_input._See_help_notes_above. ');
190     end
191     [E,cost] = process_inputs(AorV,xyCorE);
192
193
194     % Reverse the algorithm if it will be more efficient
195     isReverse = false;
196     if L > M
197         E = E(:,[2 1]);
198         cost = cost';
199         tmp = SID;
200         SID = FID;
201         FID = tmp;
202         isReverse = true;
203     end
204
205
206     % Initialize output variables
207     L = length(SID);
208     M = length(FID);
209     costs = zeros(L,M);
210     paths = num2cell(NaN(L,M));

```

```

211
212
213 % Create a waitbar if desired
214 if showWaitbar
215     hWait = waitbar(0,'Calculating_Minimum_Paths_..._');
216 end
217
218
219 % Find the minimum costs and paths using Dijkstra's Algorithm
220 for k = 1:L
221
222     % Initializations
223     iTable = NaN(n,1);
224     minCost = Inf(n,1);
225     isSettled = false(n,1);
226     path = num2cell(NaN(n,1));
227     I = SID(k);
228     minCost(I) = 0;
229     iTable(I) = 0;
230     isSettled(I) = true;
231     path(I) = {I};
232
233     % Execute Dijkstra's Algorithm for this vertex
234     while any(~isSettled(FID))
235
236         % Update the table
237         jTable = iTable;
238         iTable(I) = NaN;
239         nodeIndex = find(E(:,1) == I);
240
241         % Calculate the costs to the neighbor nodes and record
242         % paths
243         for kk = 1:length(nodeIndex)
244             J = E(nodeIndex(kk),2);
245             if ~isSettled(J)
246                 c = cost(I,J);
247                 empty = isnan(jTable(J));
248                 if empty || (jTable(J) > (jTable(I) + c))
249                     iTable(J) = jTable(I) + c;
250                     if isReverse
251                         path{J} = [J path{I}];
252                     else
253                         path{J} = [path{I} J];
254                     end
255                 else
256                     iTable(J) = jTable(J);
257                 end

```

```

258         end
259     end
260
261     % Find values in the table
262     K = find(~isnan(iTable));
263     if isempty(K)
264         break
265     else
266         % Settle the minimum value in the table
267         [~,N] = min(iTable(K));
268         I = K(N);
269         minCost(I) = iTable(I);
270         isSettled(I) = true;
271     end
272 end
273
274 % Store costs and paths
275 costs(k,:) = minCost(FID);
276 paths(k,:) = path(FID);
277 if showWaitbar && ~mod(k, ceil(L/100))
278     waitbar(k/L, hWait);
279 end
280 end
281 if showWaitbar
282     delete(hWait);
283 end
284
285
286 % Reformat outputs if algorithm was reversed
287 if isReverse
288     costs = costs';
289     paths = paths';
290 end
291
292
293 % Pass the path as an array if only one source/destination
294 %     were given
295 if L == 1 && M == 1
296     paths = paths{1};
297 end
298
299
300 % -----
301 function [E,C] = process_inputs(AorV, xyCorE)
302     if (n == nc)
303         if (m == n)
304             A = AorV;

```

```

305         A = A - diag(diag(A));
306         if (m == mc)
307             % Inputs = (A, cost)
308             C = xyCorE;
309             E = a2e(A);
310         else
311             % Inputs = (A, xy)
312             xy = xyCorE;
313             E = a2e(A);
314             D = ve2d(xy, E);
315             C = sparse(E(:,1), E(:,2), D, n, n);
316         end
317     else
318         eval(['help_' mfilename]);
319         error('Invalid_[A,xy]_or_[A,C]_inputs. ');
320     end
321 else
322     if (mc == 2)
323         % Inputs = (V, E)
324         V = AorV;
325         E = xyCorE;
326         D = ve2d(V, E);
327         C = sparse(E(:,1), E(:,2), D, n, n);
328     elseif (mc == 3)
329         % Inputs = (V, E3)
330         E3 = xyCorE;
331         E = E3(:, 1:2);
332         C = sparse(E(:,1), E(:,2), E3(:,3), n, n);
333     else
334         eval(['help_' mfilename]);
335         error('Invalid_[V,E]_or_[V,E3]_inputs. ');
336     end
337 end
338 end
339
340 end
341
342 % Convert adjacency matrix to edge list
343 function E = a2e(A)
344     [I, J] = find(A);
345     E = [I J];
346 end
347
348 % Compute Euclidean distance for edges
349 function D = ve2d(V, E)
350     VI = V(E(:,1), :);
351     VJ = V(E(:,2), :);

```

```

352     D = sqrt(sum((VI - VJ).^2,2));
353 end

```

E.6 Fonction distkm.m

Cette fonction permet de calculer la distance entre deux points avec leur coordonnées géographiques. Cette fonction correspond à la fonction lldistkm.m avec quelques modifications mineures. La fonction lldistkm.m est disponible sur le site *mathworks* à l'adresse suivante : <https://nl.mathworks.com/matlabcentral/fileexchange/38812-latlon-distance>

```

1 function d1km=distkm(latlon1 ,latlon2)
2 % format: [d1km d2km]=lldistkm(latlon1 ,latlon2)
3 % Distance:
4 % d1km: distance in km based on Haversine formula
5 % (Haversine: http://en.wikipedia.org/wiki/Haversine_formula)
6 % d2km: distance in km based on Pythagoras? theorem
7 % (see: http://en.wikipedia.org/wiki/Pythagorean_theorem)
8 % After:
9 % http://www.movable-type.co.uk/scripts/latlong.html
10 %
11 % —Inputs:
12 %   latlon1: latlon of origin point [lat lon]
13 %   latlon2: latlon of destination point [lat lon]
14 %
15 % —Outputs:
16 %   d1km: distance calculated by Haversine formula
17 %   d2km: distance calculated based on Pythagorean theorem
18 %
19 % —Example 1, short distance:
20 %   latlon1=[-43 172];
21 %   latlon2=[-44 171];
22 %   [d1km d2km]=distance(latlon1 ,latlon2)
23 %   d1km =
24 %           137.365669065197 (km)
25 %   d2km =
26 %           137.368179013869 (km)
27 %   %d1km approximately equal to d2km
28 %
29 % —Example 2, longer distance:
30 %   latlon1=[-43 172];
31 %   latlon2=[20 -108];
32 %   [d1km d2km]=distance(latlon1 ,latlon2)
33 %   d1km =
34 %           10734.8931427602 (km)

```

```
35 % d2km =
36 %      31303.4535270825 (km)
37 % d1km is significantly different from d2km (d2km is not
38 % able to work for longer distances).
39 %
40 % First version: 15 Jan 2012
41 % Updated: 17 June 2012
42 %
43 radius=6371;
44 lat1=latlon1(1)*pi/180;
45 lat2=latlon2(1)*pi/180;
46 lon1=latlon1(2)*pi/180;
47 lon2=latlon2(2)*pi/180;
48 deltaLat=lat2-lat1;
49 deltaLon=lon2-lon1;
50 a=sin((deltaLat)/2)^2 + cos(lat1)*cos(lat2) * sin(deltaLon/2)^2;
51 c=2*atan2(sqrt(a),sqrt(1-a));
52 d1km=radius*c; %Haversine distance
53 % x=deltaLon*cos((lat1+lat2)/2);
54 % y=deltaLat;
55 % d2km=radius*sqrt(x*x + y*y); %Pythagoran distance
56 end
```

UNIVERSITÉ CATHOLIQUE DE LOUVAIN
École polytechnique de Louvain

Rue Archimède, 1 bte L6.11.01, 1348 Louvain-la-Neuve, Belgique | www.uclouvain.be/epl