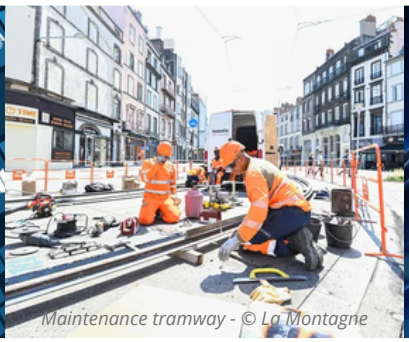
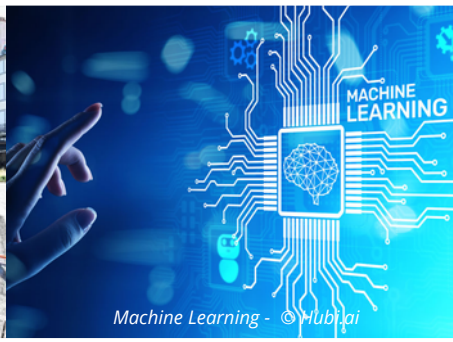
Siège social - © ARTELIA group


Tramway Bordeaux - © Wikipédia

Asset Management - © Training You


Maintenance tramway - © La Montagne



Machine Learning - © Hubi.ai

RAPPORT DE STAGE

MISSION EFFECTUÉE : *MISE EN PLACE D'UN MODELE RAILWISE DE PREDICTION D'USURE DES VOIES DE TRAMWAY ET DES MAINTENANCES ASSOCIEES*

Auteure : Amélie NEDELEC - 5A DAE
Tuteur professionnel : Pierre SOUTY
Tuteur académique : Hervé BAPTISTE

Dates de stage : du 15 février 2025 au 15 août 2025
Organisme d'accueil : Artelia - MI > TAU > STR > GPMR - Bordeaux

SOMMAIRE

TABLEAU DES FIGURES

REMERCIEMENTS

Je tiens tout d'abord à exprimer ma sincère gratitude à l'ensemble de l'équipe de la division GPMR, et plus largement à Artelia, pour m'avoir accueillie et soutenue tout au long de mon stage. Je remercie particulièrement monsieur Pierre SOUTY, mon tuteur de stage, pour sa disponibilité, ses conseils et sa volonté de m'impliquer dans diverses missions dans le monde professionnel et dans cette mission en particulier. Je tiens également à remercier monsieur Sébastien GEISSANT, un collègue qui a su m'expliquer le monde du tramway dans son entièreté.

Par ailleurs, je remercie monsieur Madjid BENALLEGUE, collègue alternant en Intelligence Artificielle, qui a su m'éclairer et m'aider durant la presque intégralité de cette expérience, et sans qui ce projet n'aurait certainement pas abouti en temps et en heures.

Je souhaite en outre remercier l'ensemble de mes collègues, notamment pour leur accueil chaleureux, leur bienveillance à mon égard et leur disposition à m'aider dans diverses tâches administratives ou techniques. Leur professionnalisme a grandement contribué à la réussite de ce stage.

Enfin, j'adresse des remerciements sincères à Hervé BAPTISTE, mon tuteur académique, pour son encadrement et ses conseils, et la vérification de mon épanouissement au sein de mes missions techniques.

GLOSSAIRE

- **Bavette :** Dans les campagnes de mesures d'usure Table, Flanc et Bavette, cette dernière correspond à la zone inférieure latérale du flanc du rail, située en dessous de la surface de roulement. Cette zone est faiblement sollicitée dans des conditions normales de fonctionnement, mais peut subir une usure significative dans certains cas malgré tout, comme dans les courbes serrées, ou lorsque le frottement est excessif, par exemple. Son usure permet donc, parfois, de détecter des anomalies de voie.

- **Files de rail :** Les files de rails sont les deux rails parallèles qui constituent la voie ferrée et qui guident et supportent les roues du tramway.

- **Fraisage** : Le fraisage des rails est un processus essentiel dans l'entretien des voies ferrées, permettant de supprimer les défauts et de restaurer l'état des rails.

- **Machine Learning :** Le Machine Learning, abrégé ML, est une branche de l'intelligence artificielle (IA) qui vise à permettre aux ordinateurs et aux machines d'imiter la façon dont les humains apprennent ; le ML effectue entre autres des tâches de manière autonome, et améliore sa performance et sa précision à mesure qu'il gagne en expérience.

- **Points Kilométriques :** Le point kilométrique (abrégé PK) est une référence linéaire utilisée pour localiser précisément un point le long d'une infrastructure linéaire, comme une ligne de tramway. Il indique la distance cumulée depuis un point de départ conventionnel, exprimée en kilomètres (et parfois en mètres).

- **Reprofilage :** Le reprofilage consiste à restaurer la forme et le profil initial du rail en enlevant une fine couche de métal sur la surface usée ou déformée, généralement par meulage ou fraisage.

- **Voie de retournement :** Il s'agit d'un aménagement ferroviaire permettant à un train, à un métro ou plus précisément dans notre cas à un tramway, de changer de sens de circulation à la fin d'un trajet, ou en cas de besoin.

INTRODUCTION

Selon le groupe de travail n°2 du GIEC (2022), et plus précisément d'après les conclusions du Chapitre 6 - Villes, établissements et infrastructures clés : « une adaptation efficace des infrastructures urbaines aux risques climatiques nécessite une planification intégrée, des connaissances locales et des capacités techniques, soutenues par une gouvernance inclusive et des mécanismes financiers* » *(Chapter 6: Cities, Settlements and Key Infrastructure, s. d.).*

Ainsi, dans le contexte d'intensification des pressions urbaines, d'accélération du dérèglement climatique et de multiplication des aléas qui en découlent, le rôle de l'ingénieure en Aménagement et Environnement est en profonde mutation. De plus en plus sollicités pour proposer des solutions préventives, résilientes et adaptatives, nous devons apprendre à appréhender chaque territoire dans sa singularité. Mieux connaître ses spécificités, c'est mieux anticiper les risques, qu'ils soient naturels, technologiques ou liés aux infrastructures.

Pour y parvenir, et afin d'alléger la charge technique du traitement de l'information, l'ingénieur peut s'appuyer sur des outils de synthèse, de modélisation et d'optimisation. C'est précisément dans cette logique que s'inscrit ce Stage de Fin d'Études (SFE), mené à Artelia, et pensé comme une expérimentation concrète de mise en œuvre d'outils au service d'une ingénierie plus prospective.

L'année passée, j'ai pu bénéficier d'une première expérience à Brest Métropole, dans la fonction publique territoriale. Son objectif était de traiter et regrouper les données pluviométriques de la ville, afin de mieux prédire les risques d'inondation à l'échelle locale. Cette première expérience m'a permis de prendre la mesure des enjeux liés à l'anticipation des aléas climatiques. Cette année, bien que le stage traité dans ce rapport semble à première vue davantage orienté vers un volet technique, il s'inscrit dans une dynamique comparable de prévention des risques. Modéliser l'usure des voies, c'est en effet anticiper une dégradation du réseau, et donc optimiser sa sécurité, sa durabilité et ses performances. Développer un modèle prédictif fiable permet ainsi de limiter les perturbations du réseau (discontinuités de service, insatisfaction des usagers), de maîtriser les coûts liés aux maintenances urgentes, et de mieux gérer les ressources matérielles utilisées pour la maintenance. En somme, modéliser l'usure de façon rigoureuse, c'est contribuer à prévenir les risques d'atteinte aux grands principes du développement durable.

Ainsi, bien que reposant sur des terrains et des données différents, ces deux expériences professionnelles poursuivent un objectif commun : proposer des outils d'aide à la décision, au service d'une gestion plus préventive et raisonnée du monde qui nous entoure. C'est dans cette continuité que s'inscrit le présent rapport, qui s'articulera en 4 parties ; présentation de la structure d'accueil ; contexte technique et territorial ; méthodologie adoptée, projet mené ; et conclusions tirées.

[*En anglais dans le texte].

1. PRÉSENTATION DE LA STRUCTURE D'ACCUEIL

1.1. Artelia dans les grandes lignes

D'après ses propres mots, Artelia est une entreprise ayant pour rôle le conseil, l'ingénierie et la gestion de projets (Artelia Group, 2025). Elle est la fusion de deux entités distinctes, regroupées en 2010 : Sogreah, experte en hydraulique et environnement ; et Coteba, spécialisée dans le bâtiment et l'aménagement urbain. Aujourd'hui, le groupe privé ne se cantonne plus à ces deux pôles d'expertise ; il s'est très largement étendu, puisque répond à des missions dans une grande diversité de secteurs. On peut citer, par exemple, le bâtiment, les mobilités, les infrastructures hydrauliques, l'énergie, l'industrie, ou encore l'aménagement maritime et fluvial.

Cette transversalité d'expertise lui permet d'entreprendre des projets de grandes envergures, en commençant par les phases AVP (AVant Projet) et allant jusqu'aux phases d'après projet. Cette capacité de l'entreprise à couvrir l'ensemble du cycle de vie d'un projet favorise grandement la dynamique de travail des équipes. En effet, en suivant l'opération dans toutes ses phases, celles-ci développent une compréhension approfondie des enjeux techniques, et des contraintes spécifiques relatives à chaque étape. Cela limite les risques d'erreurs ou de pertes d'informations entre les phases, assurant ainsi une meilleure qualité globale des livrables. Cela favorise également le partage de savoir-faire, la montée en compétences multidisciplinaires, et stimule l'innovation par la confrontation de points de vue.

Par ailleurs, en maîtrisant toutes les étapes du projet, Artelia peut adapter ses propositions en temps réel, anticiper les imprévus et ajuster les solutions techniques au fur et à mesure. Cette agilité renforce la satisfaction client, et la réputation du groupe, permettant ainsi à l'entreprise d'atteindre un chiffre d'affaires en 2024 de 1,15 milliards d'euros. Enfin, cette politique implique une forte responsabilisation des individualités. Les collaborateurs se sentent pleinement impliqués dans la réussite des projets, ce qui favorise la motivation et l'esprit d'appartenance à l'entreprise. Artelia adopte d'ailleurs un modèle de gouvernance particulier, dans la mesure où elle est principalement détenue par ses quelque 10 100 collaborateurs *[derniers chiffres de 2025]* (Artelia Group, 2025), faisant d'elle une entreprise à capital majoritairement salarié. De fait, son organisation dite horizontale constitue l'un des atouts majeurs d'Artelia. En capitalisant sur l'intelligence collective de ses collaborateurs, Artelia affirme donc son ambition de répondre de manière durable aux défis contemporains, tout en cultivant un modèle entrepreneurial centré sur l'humain, l'implication et la confiance.

1.2. Le fonctionnement interne d'Artelia

En raison de cette politique particulière, l'entreprise à su se développer sereinement, allant jusqu'à s'implanter aux quatre coins du monde. Aujourd'hui, elle est présente dans 40 pays, en Europe, en Afrique, au Moyen-Orient, en Asie-Pacifique ou encore en Amérique. Son expansion est telle qu'elle appartient même, aujourd'hui, au top 15 des ingénieries européennes de la construction (Artelia Group, 2025).

Pour favoriser l'interconnexion à l'international, Benoît CLOCHERET, président exécutif du groupe, a divisé le groupe en Business Units *[Figure 1]*. Ainsi, il existe actuellement, en tout et pour tout, 10 Business Units, aussi appelées BU ; dont l'organisation interne favorise le développement de projets cohérents. Il est important de souligner que les BU ne sont pas classées ni selon leur positionnement géographique, ni selon le sujet qu'elles traitent, mais plutôt selon un intermédiaire – qui permet aux collaborateurs de situer exactement leur domaine de compétence dans un projet. Cette organisation en Business Units, loin de cloisonner les expertises, est pensée pour encourager la transversalité. En effet, chaque BU travaille en étroite collaboration avec les autres, ce qui favorise le partage de connaissances et de compétences en interne. Ainsi émergent des solutions techniques riches et adaptées à chaque situation.

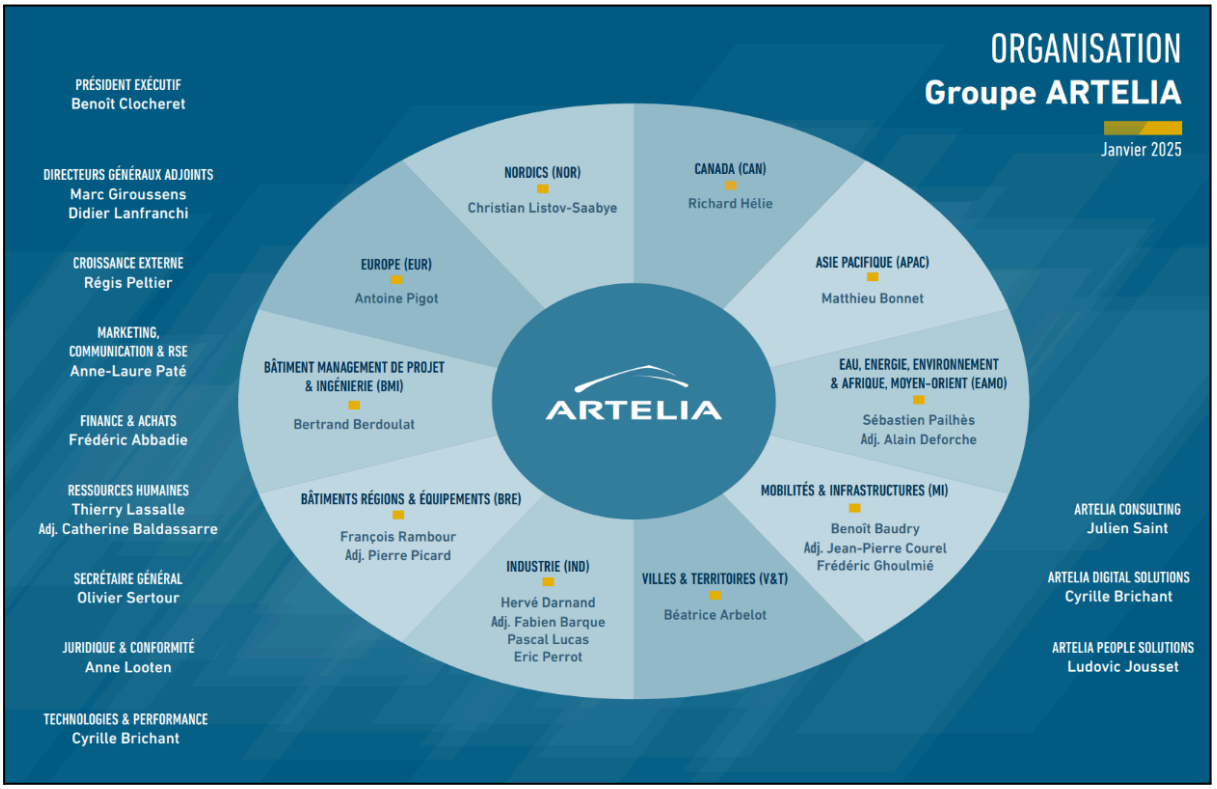


Figure 1 : *Organisation Interne d'Artelia – ARTELIA GROUP*

Chaque Business Unit (BU) fonctionne comme un pôle de compétences spécialisé, mobilisant des expertises variées selon les besoins des projets. Leur organisation n'est pas figée : une révision globale est d'ailleurs prévue à l'horizon 2026.

Mon stage s'inscrit dans la BU actuelle MI – Mobilités et Infrastructures, à la croisée des enjeux de mobilité, d'aménagement du territoire et de développement durable. Elle reflète l'approche intégrée d'Artelia, et se divise en plusieurs sous-structures qui renforcent la technicité de ses interventions.

1.3. Le pôle GPMR

Parmi les sous-structures de la BU, on peut notamment citer le service MI – TAU – STR – GPMR, service dans lequel j'ai effectué mon stage. Par GPMR, on entend Gestion Patrimoniale et Matériel Roulant ;

Bien que cette sous-unité soit souvent méconnue dans le domaine de l'ingénierie, son rôle est crucial pour assurer la pérennité des infrastructures dont elle traite. La gestion patrimoniale consiste à suivre, entretenir et optimiser les actifs tout au long de leur cycle de vie, en anticipant les besoins de maintenance, les renouvellements et les évolutions techniques *[définition mise en avant en interne]* (Artelia Group, 2025). Sans cette démarche, une dégradation prématurée viendrait avec une probabilité importante de toucher lesdites infrastructures, entraînant des coûts de réparation bien plus élevés, des interruptions de service, voire des risques accrus pour la sécurité des passagers.

Qui plus est, dans un contexte où les enjeux environnementaux sont majeurs, la gestion durable des actifs permet de limiter l'empreinte carbone des opérations, en favorisant la réhabilitation plutôt que le remplacement systématique, et en optimisant l'utilisation des ressources. Cela contribue à une meilleure efficacité économique, tout en répondant aux exigences croissantes de responsabilité sociétale. Ainsi, la sous-unité GPMR joue un rôle stratégique, au cœur des projets, en assurant la fiabilité de tout un système. C'est précisément dans cette optique que s'est inscrit le stage réalisé au sein du pôle. Plus spécifiquement, mes travaux portent sur un sujet mêlant données d'exploitation, enjeux de durabilité des infrastructures de transport, et appui à la décision pour la maintenance prédictive d'un réseau de tramways, anonymisé pour la suite.

2. CONTEXTE ET PROBLÉMATIQUE DU STAGE

2.1. Missions du pôle GPMR

Pour bien comprendre les enjeux et le périmètre de mon travail, il convient d'abord de situer précisément les objectifs et les livrables attendus. Ce stage portait sur la problématique suivante : établir une méthode d'analyse des données et de projection des usures de la voie. Avant d'entrer dans les aspects techniques du sujet, remplaçons la mission dans le cadre plus large de l'activité d'Artelia, et du rôle que joue le pôle GPMR dans les projets de transport.

Artelia, en tant qu'entreprise privée d'ingénierie, intervient principalement comme maître d'œuvre (MOE). À ce titre, son rôle est d'accompagner le maître d'ouvrage (MOA) en assurant l'organisation, la coordination et le suivi technique des projets qui lui sont confiés. Sa responsabilité porte sur la mise en œuvre concrète de solutions : autrement dit, *comment répondre aux besoins du MOA* tout en respectant les exigences de qualité, les contraintes techniques, les délais impartis et le budget alloué. Dans cette optique, Artelia a donc pour rôle de fournir au MOA ainsi qu'à l'autorité organisatrice (AO) l'ensemble des informations pertinentes dont elle dispose, notamment celles susceptibles d'impacter le bon déroulement du projet ou son exploitation future.

Dans ce cadre, le pôle GPMR se focalise sur les prestations concernant les transports de type Matériel Roulant à l'instar des métros ou tramways ; exception faite du ferroviaire, qui lui est géré par une autre branche, du nom de TRF. L'objectif final est de permettre au commanditaire d'exploiter le réseau qu'il décide de construire, renouveler ou compléter, grâce une fine analyse technique de ses composantes.

C'est dans cette logique que s'inscrivent les tâches qui m'ont été confiées, centrées sur la mise en place d'une stratégie de maintenance prédictive, appliquée à un réseau anonyme de tramways. En effet, la capacité d'anticiper l'usure des rails et des équipements associés représente un enjeu clé pour assurer la fiabilité et la sécurité du réseau, tout en maîtrisant les coûts de maintenance. Il s'agit là d'une mission typique du MOE dans le cadre plus large de la Gestion Patrimoniale. L'idée est, in fine, de proposer à l'AO un outil opérationnel permettant d'anticiper les interventions possibles pour lui afin d'optimiser sa gestion du réseau. Cette méthode présente donc un double intérêt : pour Artelia, un gain de temps dans la production des livrables ; pour l'AO, une transition vers une maintenance plus proactive, en rupture avec les pratiques actuelles fondées sur des inspections périodiques et des interventions réactives, générant ainsi des économies à la fois en temps et en budget.

2.2. La physionomie des infrastructures liées au tramway

Avant de comprendre quels paramètres de voies étaient primordiaux dans le calcul de l'usure de celles-ci, il a d'abord fallu en apprendre davantage sur le tramway en général. Pour rendre cette acquisition de compétences possible et illustrée, j'ai eu la chance et l'opportunité d'accompagner l'un de mes collègues, dont l'ancien travail consistait justement à effectuer de la maintenance pour l'exploitant tramway de Bordeaux, Kéolis, à la rencontre des anciennes équipes qu'il supervisait ; ces

dernières ayant donc pris la matinée pour m'expliquer pas à pas le contenu de leur travail. Sur le terrain, j'ai eu la possibilité d'observer différentes composantes de la voie, qu'il convient de décrire précisément ici.

2.2.1. Les types de rails

En premier lieu, nous sommes allés observer les différents types de rails existants. Dans le milieu du tramway, deux principaux types cohabitent ; le rail Vignole d'une part, composé de trois grandes parties :

- **Le champignon**, ou tête de rail : il s'agit de la partie supérieure sur laquelle les roues des véhicules roulent, bombée pour supporter les charges et limiter l'usure ;
- **L'âme** du rail : il s'agit de la partie verticale centrale qui relie le champignon au patin et donne au rail sa hauteur et assure la transmission des efforts ;
- **Le patin** : c'est la partie inférieure qui repose sur les traverses et permet la fixation du rail à la voie et qui répartit la charge sur une plus grande surface.

Pour précision, ce type de rail est celui que l'on retrouve également en voie ferrée.

D'autre part, on a le rail à gorge. Contrairement au rail Vignole, il est spécialement conçu pour les réseaux de tramway en milieu urbain, où les rails sont intégrés dans la chaussée. Il permet à la fois le passage des véhicules sur pneus et celui du tramway, en toute sécurité. Dans sa forme, il est asymétrique, composé de :

- **La table de roulement** : la partie surélevée et lisse où circule la roue du tramway ;
- **La gorge** : la partie où un canal latéral creux accueille le boudin de la roue, ce qui permet au tramway de rester bien aligné dans la voie ;
- **Le talon** : la partie inférieure, servant à la fixation sur la plateforme ou le massif béton.

Pour illustration, le schéma suivant *[Figure 2],* qui permet de condenser visuellement l'information discutée précédemment.

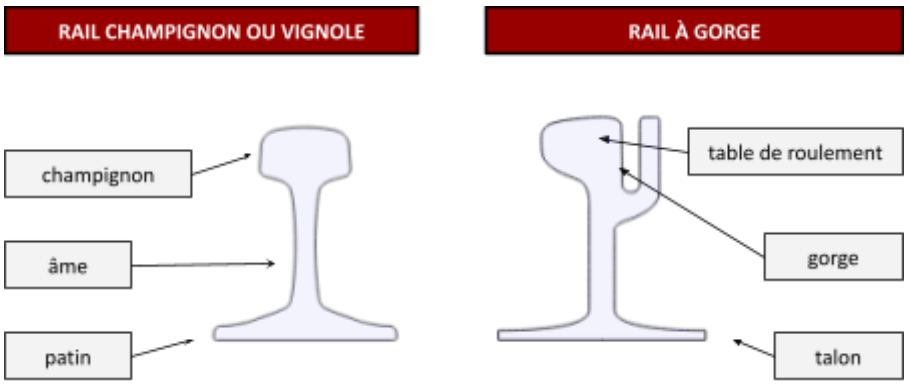


***Figure 2** : Les 2 grands types de rails – NEDELEC Amélie*

2.2.2. Les types de poses

Par la suite, on m'a présenté les divers types de pose conjuguées à ces différents rails ;

- D'une part, la pose sur ballast, qui consiste à installer la voie sur un lit de pierres concassées. Ce lit de pierre joue un rôle essentiel dans la stabilité de la plateforme, l'absorption des vibrations, ainsi que le drainage des eaux de pluie. Ce mode de pose est généralement réservé aux sections de ligne situées en dehors des zones urbaines denses, par exemple en périphérie ou sur des voies de service.

- D'autre part, il existe la pose sur dalle béton, souvent intégrée dans la voirie. Elle concerne une grande partie de la pose de rails en milieu urbain. Ce type de pose est utilisé en site intégré, c'est-à-dire lorsque la plateforme du tramway est partagée avec d'autres usagers (piétons, cyclistes, véhicules), comme c'est le cas en centre-ville. Elle permet un meilleur niveau d'intégration urbaine, avec un revêtement fini (pavage, enrobé, gazon, …), et offre une meilleure stabilité géométrique de la voie dans le temps. Toutefois, elle requiert une mise en œuvre plus précise, et présente des contraintes spécifiques en matière d'accessibilité, d'entretien et de gestion des réseaux enterrés.

Connaître l'existence et la différence entre ces deux types de poses est important dans le contexte du stage car, par expérience métier, on sait que chaque mode de pose à des impacts sur le type de rail associé : la stabilité des bogies, ou encore le confort des passagers; mais également sur propagation des efforts mécaniques dans le rail, et donc sur l'usure de ce dernier. Il faut donc être à même de les prendre en compte pour dresser des constats sur les voies par la suite.

Cette fois encore, pour une compréhension plus accrue de la différence des poses existantes, une petite figure *[Figure 3]* a été conçue par mes soins, la voici donc ci-dessous :

POSE BALLAST

Pose ballast – Futura Sciences
https://cdn.futura-sciences.com/cdn-cgi/image/width=1024,quality=60,format=auto/sources/images/actu/ballast.jpg

POSE BETON

Pose béton – Infociment
https://www.infociments.fr/sites/default/files/styles/image_contenu_article/public/images-contenu-articles/_DSC2184.jpg.webp?itok=U5FYAUgD

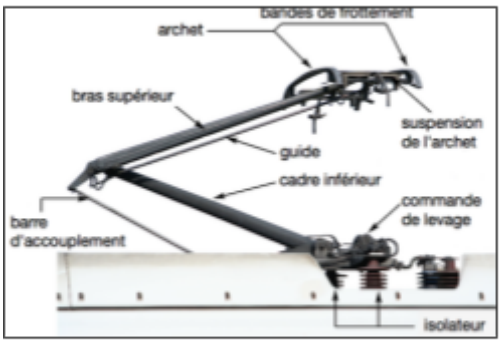
***Figure 3** : Les différents types de poses – NEDELEC Amélie*

Ensuite, j'ai pu voir tous les dispositifs physiques relatifs aux infrastructures ferroviaires ou ferroviaires urbaines, décrits avec précision par les intervenants réseaux rencontrés : appareils de voies (notamment aiguillages), signalisations, exemples d'ouvrages d'art… de quoi alimenter, pour moi, vocabulaire technique et compréhension approfondie du sujet*.

[*Afin de ne pas alourdir le rapport, ces infrastructures ne seront pas détaillées ici. Toutefois, elles m'ont été clairement présentées par les intervenants, ce qui m'a permis de les intégrer à mon champ de compétences, un acquis précieux pour la suite de mon parcours]

Du reste, les éléments relatifs à la LAC (ligne aérienne de contact) m'ont également étaient présentés en détail : le pantographe d'abord, dispositif situé sur le toit des tramways, qui permet de capter l'électricité nécessaire à leur fonctionnement. Il est en contact avec une caténaire, et sa forme articulée lui permet de rester plaqué en permanence contre le fil, même si le véhicule bouge ou si la hauteur du fil varie. Son rôle est d'assurer le transfert du courant électrique depuis la caténaire vers le moteur du véhicule ferroviaire. D'un même fait, les différentes règles de sécurité lui étant

PANTOGRAPHE – LAC



Pantographe – Wikipédia
https://upload.wikimedia.org/wikipedia/commons/thumb/a/a0/Pantographe_ICE_3.png/500px-Pantographe_ICE_3.png

relatives m'ont été présentées, me permettant de saisir tous les enjeux des métiers du domaine.

Par ailleurs, un volet important de la visite terrain a concerné les opérations de maintenance. Les équipes m'ont expliqué les protocoles de surveillance des rails (notamment le contrôle de l'usure de la table et du flanc de rail), les méthodes d'intervention selon les types de pose, et les différents cycles de maintenance préventive ou corrective. Ces informations ont été précieuses pour comprendre les mécanismes d'usure et les contraintes d'exploitation, éléments centraux de mon sujet de stage.

2.3. Les types de maintenance

Ces échanges terrain ont permis de relier les observations concrètes aux stratégies globales d'entretien mises en place sur le réseau. Pour approfondir cette compréhension, il est désormais nécessaire de présenter les différents types de maintenance, afin de cerner les logiques d'intervention et d'anticipation qui structurent la gestion des infrastructures. En première instance, présentons la distinction entre les deux grandes familles de maintenances que l'on connaît : la maintenance préventive et la maintenance corrective.

2.3.1. La maintenance corrective

La maintenance corrective est un regroupement de l'ensemble des actions réalisées à postériori d'un dysfonctionnement constaté sur le réseau. Elle intervient dans le but de remettre en service un système afin qu'il obéisse à son état de fonctionnement normal. Il existe deux sous-types de maintenance corrective :

A. La maintenance curative

L'objectif de cette maintenance est de réparer définitivement la défaillance dudit système. Elle consiste, après diagnostic de la cause du problème, en son élimination, souvent via un remplacement du système défectueux. Le point positif de cette manœuvre réside dans le fait qu'elle soit durable, car une pièce remplacée ne posera de fait plus de problème de sitôt. Cependant, les coûts économiques et environnementaux des nouveaux matériaux employés la rende peu pérenne ; d'autant plus que l'intervention nécessite, la plupart du temps, un arrêt complet du service, rendant souvent les usagers mécontents.

B. La maintenance palliative

Dans la même sous-catégorie, on a donc ce deuxième type de maintenance ; à l'inverse de la maintenance curative, l'intervention est souvent rapide, car relève d'une urgence sur le réseau. Moins contraignante pour les usagers, elle suscite quand même des complications, puisque n'apporte qu'une solution temporaire ou partielle au problème, nécessitant souvent une seconde intervention, assez rapidement, à postériori. Une fois de plus, cette solution n'est pas réellement

intéressante ni au niveau économique, ni au niveau écologique ; il faut donc y avoir recours au minimum possible.

2.3.2. La maintenance préventive

A contrario, la maintenance préventive regroupe quant à elle l'ensemble des actions planifiées en amont d'un événement, visant à réduire la probabilité de panne ou de dégradation de l'équipement, avant que celle-ci n'intervienne. Il s'agit d'une dynamique d'anticipation, qui s'inscrit donc dans le principe de gestion de risques évoquée plus tôt dans ce rapport. La maintenance préventive se déploie en trois sous-catégories, que voici :

A. La maintenance systématique

Cette maintenance est réalisée à intervalles réguliers, suivant un planning fixe. Elle a l'avantage de réduire drastiquement la charge mentale de l'intervenant réseau ; mais le désavantage de prévoir des interventions soit trop fréquentes, et donc des coûts économiques trop importants par rapport au besoin ; soit trop peu fréquentes, rendant donc cette maintenance initialement préventive, corrective in fine.

B. La maintenance conditionnelle

Afin d'être davantage sujet aux observations pratiques qu'à de la théorie, il existe ce que l'on appelle la maintenance conditionnelle. Celle-ci est déclenchée en fonction de l'état réel de l'équipement, via des mesures ou inspections. A l'inverse de la maintenance systématique, elle se base sur des faits et réduit donc les interventions inutiles et coûts associés. Cependant, elle présente également certains défauts, à savoir un coût économique important dédiés à l'intervention de personnes physiques et au matériel qu'elles utilisent ; et donc, de surcroît, rendent les interventions dépendantes d'observations humaines, amplifiant alors les manquements liés aux erreurs possibles d'interprétation.

C. La maintenance prédictive

Ainsi, afin de réduire au maximum lesdites erreurs d'interprétation, il existe un modèle de maintenance basée sur de l'algorithmie, c'est-à-dire sur des principes mathématiques de stockage puis d'apprentissage de comportements de systèmes à partir de données d'entrées : la maintenance prédictive. Ce principe de maintenance, davantage informatisé que les précédents, permet ainsi d'optimiser la planification des interventions, et donc de minimiser les arrêts imprévus et les coûts sous-jacents. Malgré tout, il est important de considérer la forte dépendance du système de maintenance aux données initiales qui lui sont attribuées, rendant de fait la pratique très vulnérables aux défaillances techniques, et demandant donc une attention régulière au modèle mis en place. Par ailleurs, il faut également souligner le paramètre écologique lié au déploiement d'un tel modèle ; la maintenance prévisionnelle reposant en effet sur la collecte et l'analyse de très grands volumes de données, cela nécessite des data centers énergivores, souvent alimentés par des sources d'énergie

non renouvelables, et parfois situés à l'autre bout du monde, ce qui est un point critique majeur du procédé, qui sera détaillé en partie 4.

2.3.3. Occurrence de maintenance

Il est également intéressant de distinguer les différents types de maintenances selon l'occurrence associée à celles-ci ; on a, d'une part, la maintenance courante, d'occurrence hebdomadaire à bi-annuelle, qui concerne notamment le nettoyage des rails, le graissage des aiguillages toutes les semaines, ou le contrôle et ajustement des fixations ; et d'autre part, la maintenance ponctuelle, plus occasionnelle comme son nom l'indique, qui concerne davantage les actions sur lesquelles ce rapport se concentre, c'est-à-dire les rechargements, actions consistant à rajouter de la matière (souvent de l'acier) sur la surface usée du rail, puis à reprofiler* le rail pour retrouver sa forme d'origine ; et les remplacements, actions consistant à retirer un rail ou une portion de rail usée ou endommagée, et à le remplacer complètement par un rail neuf.

[* Reprofiler : Voir le Glossaire]

Après réflexions, discussions et comparaisons, ce stage se spécifie donc sur la mise en place d'un modèle de maintenance préventive – prédictive dédié aux opérations ponctuelles, à savoir rechargements puis remplacements des portions de rails sur les voies. Le but est de prédire l'usure en première instance, en mm de diminution d'épaisseur, avant de conclure sur une année de maintenance pour chaque portion de voie. Avant de développer ledit modèle à proprement parler, il faut distinguer précisément les différents types d'usures qui existent, et sur lesquels nous allons nous concentrer dorénavant.

2.4. Les types d'usures

Les usures de voies de tramways, comme les types de maintenances, peuvent être catégorisées de deux manières différentes ; selon le mécanisme d'usure, c'est-à-dire de la manière où l'usure se produit – et selon la localisation de ladite usure. Voici comment cela se traduit concrètement :

2.4.1. Usures selon le mécanisme

Dans le cadre ferroviaire urbain que représente le monde du tramway, plusieurs mécanismes physiques contribuent à la dégradation progressive des rails. Il convient alors de les distinguer ainsi :

A. Usure abrasive

Il s'agit du mécanisme le plus connu, mais aussi le plus fréquent. Ce type d'usure résulte du frottement mécanique direct entre la roue et le rail. Analogiquement, il serait semblable au

phénomène de ponçage. Ce type de mécanisme survient principalement sur la table de roulement, dû au contact répété entre la roue et la table ; et sur le flanc du rail, lorsque la roue exerce une pression latérale marquée. Ce type d'usure est reconnu comme dépendant des paramètres géométriques et physiques de voie ; et bien qu'aucune équation précise n'existe, il est possible de trouver une liaison factuelle entre ces paramètres en question et leur résultante mécanique, nous le verrons par la suite.

B. Usure ondulatoire

En parallèle, il existe ce que l'on nomme usure ondulatoire, une forme particulière de dégradation, caractérisée par l'apparition de vagues périodiques à la surface du rail. Elle est principalement liée à des phénomènes vibratoires et dynamiques, et peut donc peut générer des bruits de roulement importants ainsi qu'une usure accélérée. En revanche, bien que facilement observable à l'œil nu, elle est bien plus difficile à quantifier, et donc à relier à des paramètres de voies tangibles.

C. Usure par fatigue

Ce dernier type d'usure ne se manifeste pas directement par un enlèvement de matière, mais plutôt par l'apparition progressive de microfissures internes, causée par la répétition d'efforts mécaniques. Elle est donc, en quelque sorte, le prolongement de l'usure abrasive, si cette dernière n'est pas suffisamment prise en compte par les mainteneurs de voies. Il s'agit d'un mécanisme souvent associée aux zones de forte sollicitation, et nécessitant donc une attention particulière et des interventions spécifiques.

2.4.2. Usures selon le mécanisme

Il est important par ailleurs de stipuler que, peu importe le mécanisme concerné, l'usure intervient toujours dans une région du rail. De fait, la localisation de l'usure peut-être, au choix, verticale ou horizontale :

A. Usure verticale

Cette usure, aussi nommée usure de table (ou TABLE dans notre contexte stage), correspond à une diminution de l'épaisseur de la table de roulement ou de la hauteur de champignon en fonction du type de rails analysé. Étant la zone de contact principal roue-rail, c'est à cet endroit que se concentrent les efforts verticaux, dus notamment au poids des rames. Cette usure, à condition d'avoir des appareils de mesure adéquats, est facilement mesurable sur le terrain. Si elle dépasse un certain seuil, la sécurité des usagers du tramway est compromise. Il faut donc attentivement l'observer, afin de maintenir des conditions de fiabilité de service.

B. Usure horizontale

Cette usure-ci concerne la partie interne du champignon, c'est-à-dire la zone en contact avec le boudin de roue lors des circulations en courbe ou dans certaines situations de dévers. Elle est aussi nommée usure du flanc (ou FLANC directement). Concrètement, cette usure est davantage fonction des paramètres de voies (surtout géométriques) que la précédente. Les risques associés à la mauvaise prise en compte de cette usure sont également liés à la sécurité des usagers, car une mauvaise prise en compte donnerait lieu à des risques de déraillement importants.

C. Usure combinée

Le plus souvent, les deux types d'usure coexistent. On appelle souvent cette usure l'usure combinée. Cette situation est fréquente dans les zones courbes soumises à un trafic important. Le cumul de ces deux usures accroît la criticité de la section concernée, justifiant des interventions de maintenance plus fréquentes ou ciblées.

Dans notre cas de figure, on observe principalement une usure abrasive, combinée horizontale et verticale. C'est donc dans l'objectif de prédire ce type d'usure que sera imaginé notre modèle. Pour résumer les informations nécessaires à la compréhension de l'imagination du modèle, un schéma récapitulatif *[Figure 4]* a été conçu ; le voici ci-dessous.

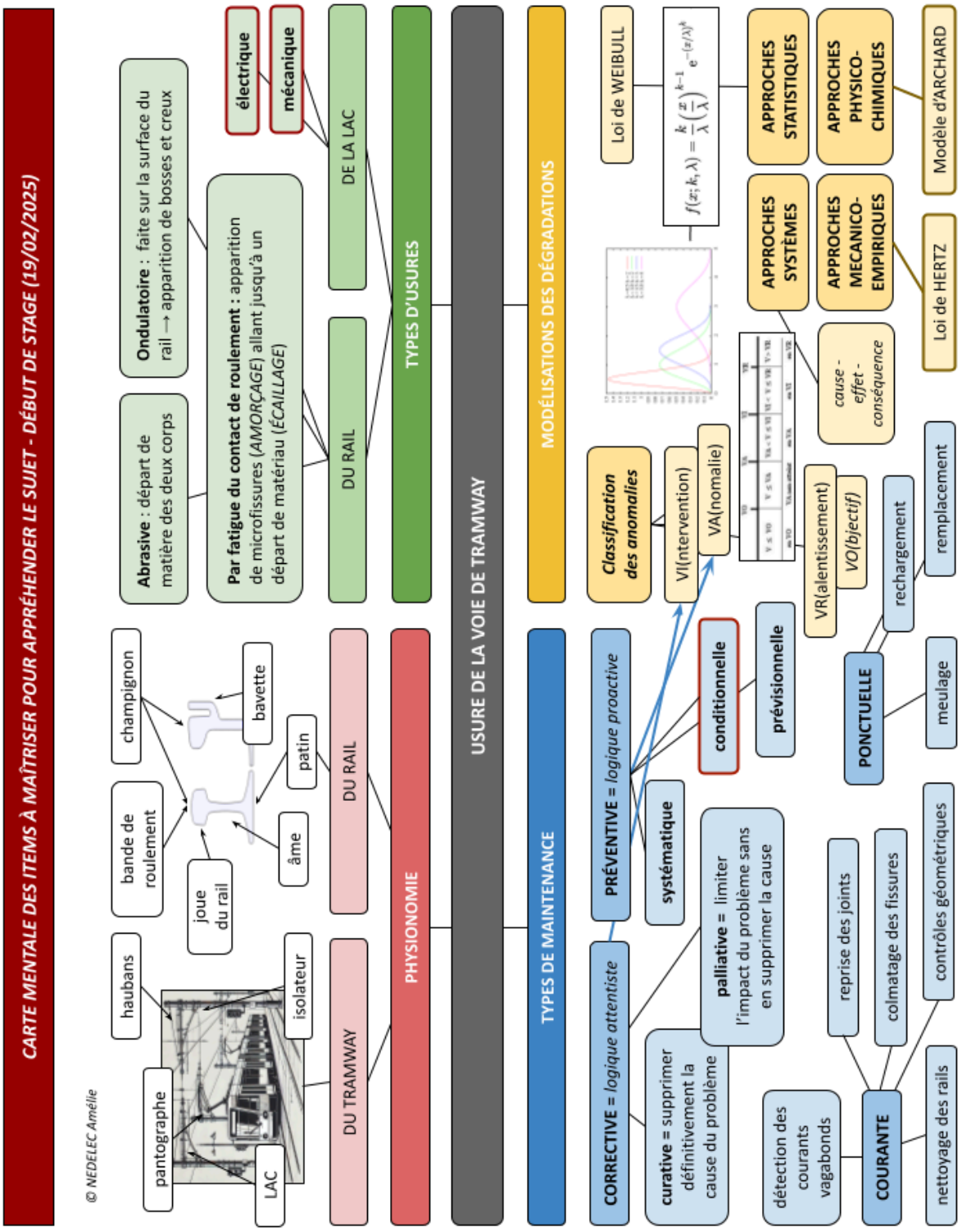


Figure 4 *: Récapitulatif des types d'usures et de maintenances des voies de tramways – NEDELEC Amélie*

3. MÉTHODOLOGIE MENÉE ET OUTILS UTILISÉS

3.1. Démarche adoptée – CRISP-DM

Afin de répondre à cette problématisation autour de l'anticipation des usures combinées, dans un objectif de maintenance préventive – prédictive, il a été nécessaire d'adopter, pour la suite de processus de construction du modèle, une démarche rigoureuse. Parmi les méthodes existantes en recherche scientifique, notamment en data science, il en existe une qui se nomme la ***méthode CRISP-DM***, ou Cross Industry Standard Process for Data Mining, qui est une approche itérative, reposant sur 6 grandes étapes interconnectées (*IBM SPSS Modeler Subscription*, s. d.) :

1. **Compréhension du métier – Business Understanding**, en cernant le problème à résoudre, le contexte et les objectifs à atteindre *[partie étayée précédemment en section 1 et 2] ;*
2. **Compréhension des données – Data Understanding**, en explorant les données disponibles, en évaluant leur qualité, et en identifiant des potentiels problèmes associés ;
3. **Préparation des données – Data Preparation**, dont l'objectif est de transformer les données brutes en données exploitables ;
4. **Modélisation – Modeling**, avec le but d'appliquer des algorithmes pour répondre à la problématique précédemment soulevée ;
5. **Evaluation – Evaluation,** en vérifiant la robustesse du modèle ;
6. **Déploiement – Deployment**, avec l'idée d'intégrer les résultats dans un outil opérationnel et/ou un livrable pour les utilisateurs finaux.

Cette méthode, dans sa forme, illustre la transition progressive des sciences vers une approche davantage centrée sur les données que sur de la théorie pure. Par son articulation itérative, elle interroge les fondements mêmes de la preuve scientifique, en rendant visibles les mécanismes de construction, de validation et parfois de remise en question du savoir. Elle s'inscrit ainsi dans une perspective épistémologique de type constructiviste, où la connaissance se bâtit progressivement à partir de l'expérience, des données, et des ajustements continus. Au-delà de sa capacité à produire des résultats fiables, cette démarche permet d'introduire une réflexion sur la philosophie des sciences, sur la nature même du savoir scientifique, et sur la manière dont celui-ci émerge et évolue dans le cadre concret d'un projet appliqué tel que ce stage.

3.2. Data Understanding : sources, format et qualité de la donnée

La première étape de la méthode, le **Business Understanding**, ayant déjà été menée à bien, la seconde consiste à analyser les données disponibles : identifier leur provenance, examiner leur format, évaluer leur compatibilité pour une éventuelle concaténation, et déterminer ainsi leur potentiel d'utilisation dans une logique de modélisation.

Cette phase préliminaire est cruciale pour diverses raisons : vérifier si l'automatisation d'un traitement est possible ; réfléchir à comment pallier une potentielle hétérogénéité de la donnée ; et orienter, par la suite, les potentiels choix méthodologiques.

Les données rassemblées dans le cadre de ce projet s'inscrivent selon plusieurs formes et formats, contiennent des informations tantôt complémentaires, tantôt antinomiques, et peuvent donc plus ou moins rapidement être mises en corrélation les unes avec les autres. Cela résulte, entre autres, du fait que plusieurs prestataires aient qualifié, géré et partagé cette donnée. Voici comment cela se traduit concrètement :

3.2.1. Les prestataires techniques

Le premier prestataire technique concerné, celui grâce auquel nous avons pu avoir accès à la majeure partie de la donnée exploitable, est DEUTZER. DEUTZER, c'est une entreprise, de taille moyenne d'après ses propres mots, qui fournit aux organisations qui y font appel des mesures des usures et paramètres de voies, à résolution spatiale prédéterminée (ce que l'on nomme Points Kilométriques*), et avec une précision importante (*Offizielle Website Der Deutzer GmbH*, 2024). Elle déploie et met à disposition pour un nombre restreint de participants au projet, un logiciel qui lui est propre. Bien que très riche en informations, ledit logiciel n'est pas très accessible pour une personne non formée, et le guide mis à disposition est principalement en allemand. Qui plus est, la donnée stockée n'est récupérable qu'au format texte, et nécessite donc un traitement préliminaire avant de pouvoir être exploitée. En revanche, l'ensemble des paramètres de voies ayant été relevés à pas de 10 centimètres le long des lignes du réseau, une fois ces manipulations préliminaires effectuées, la donnée acquise par ce premier prestataire nous a été facilement réutilisable pour la suite du traitement. Il convient toutefois de préciser que les données exploitées proviennent d'une unique campagne de mesure réalisée en 2024. Dès lors, un retour d'expérience complet sur l'évolution de l'usure n'a pas pu être établi uniquement à partir des relevés fournis par ce prestataire, et donc notre exploitation a dû également dépendre d'autres intervenants.

---

* Points Kilométriques : voir Glossaire

Bien heureusement, un second prestataire, antérieur à DEUTZER, nous a permis d'acquérir ce fameux retour d'expérience nécessaire. Il s'agit là de TeamFer, une fois de plus d'après ses propres mots spécialiste de la mesure pour le ferroviaire, et concepteur, fabricant et commercial des outils de mesure de géométrie de la voie ferrée (*TeamFer, la Mesure Pour le Ferroviaire*, s. d.). TeamFer nous a permis d'avoir accès à des mesures d'usures sur certains points de certaines courbes de certaines lignes du réseau étudié. On comprend de fait aisément par ce descriptif que la quantité de données et la qualité géographique de celles-ci est relativement faible ; mais, point positif, elle concerne les retours d'usure de FLANC et de TABLE sur les années 2021, 2022 et 2023. Ainsi, bien que les données aient été fournies sous des formats et selon des référentiels PK différents, les deux entités ont transmis suffisamment d'informations pour permettre un croisement pertinent entre les données, et donc une extraction quasiment complète des éléments recherchés.

En complément de ces informations, il a par ailleurs fallu télécharger et exploiter certaines autres données, non présentes dans les relevés des prestataires, mais nécessaires à notre conception d'un modèle. Ces data ont pu être retrouvées dans les informations fournies par la société chargée de délégation de service public des transports en commun du réseau étudié ; certaines au format ShapeFile, donc utilisables exclusivement sur des logiciels de SIG (en l'occurrence, QGis) ; d'autres au format PDF, qu'il a donc fallu extraire, extrapoler puis repositionner suivant certains critères, sur le logiciel qui a permis la concaténation de l'ensemble de ces données.

3.2.2. Le logiciel de stockage – BDD

Ce logiciel en question est Snowflake : une plateforme de data cloud qui permet de stocker, gérer et analyser de grandes quantités de données dans un environnement cloud sécurisé. Il s'apparente à ce que l'on attend traditionnellement d'un logiciel de traitement des bases de données (BDD). A contrario, ce qui fait la différence de cette plateforme avec d'autres, et qui la rend particulièrement intéressante, c'est sa capacité à être adaptée à divers types de besoins, avec la rapidité, la fluidité et simplicité d'utilisation qui est de mise – et ce, même avec des volumes data importants. Dans le cadre de ce projet, Snowflake est donc le logiciel adapté, car nous a permis de centraliser toutes nos données, de les structurer de manière cohérente et surtout de les rendre facilement accessibles à différents services. On a notamment pu :

- importer automatiquement les données collectées sur le terrain ou issues d'outils métiers ;
- croiser facilement plusieurs sources d'informations (mesures d'usure, données d'exploitation, etc.) grâce à un langage de requête unifié ;
- et préparer les données en amont de la modélisation, sans avoir à multiplier les outils ou les transferts.

Finalement, au-delà des aspects purement techniques, l'usage de Snowflake s'inscrit dans une démarche plus large de modernisation des outils internes, avec une vraie volonté de rendre la donnée plus accessible, mieux exploitée, et plus utile dans la prise de décision. Dans cette idée, j'ai d'ailleurs rédigé une notice d'utilisation de la plateforme, dans une optique finale de partage des connaissances. *(ANNEXE 1).*

3.3. Data Preparation : nettoyage, sélection et structuration des données

3.3.1. Harmonisation des systèmes de coordonnées – étape préliminaire

Une fois stockée, la donnée doit, dans une seconde phase, être traitée. La première étape de ce processus a été la recalibration géographique. Il s'agit là d'un réajustement ou d'une correction de la localisation spatiale de données, mais également leur mise en commun dans un système métrique unique, et universel en l'occurrence, et ce afin d'assurer leur alignement précis avec une référence géographique fiable. En effet, les points kilométriques, ou PK, différaient initialement entre les prestataires. Les PK, pour rappel, sont des références spatiales linéaires utilisées pour localiser précisément un point. Ils indiquent la distance entre un point donné et un point de départ officiel, et sont donc très dépendants du système auquel ils appartiennent *(voir Glossaire).* Le traitement relatif

à cette mise en commun a donc consisté à reporter l'ensemble des points sur une carte, puis à en recueillir les coordonnées WGS84 (World Geodetic System 1984) – c'est-à-dire le système de référence géodésique mondial – utilisé principalement pour la géolocalisation par GPS, et qui permet de représenter avec précision n'importe quel point sur Terre à l'aide de coordonnées géographiques. Ainsi convertis dans le même système métrique, les points des deux prestataires ont pu être concaténer dans une base de données unique, stockée sur SnowFlake.

3.3.2. Variables explicatives intégrées dans le modèle : arbitrage et justification

Avant de procéder à toutes autres formes de traitement sur la donnée, il a d'abord fallu comprendre à quels paramètres de voies l'usure des rails étaient liés, en général, et ici en particulier. Pour ce faire, nous avons construit une matrice de corrélation, outil statistique permettant de quantifier et de visualiser les relations linéaires entre plusieurs variables numériques. Chaque cellule de la matrice correspond au coefficient de corrélation entre une paire de variables, généralement mesuré à l'aide du coefficient de Pearson, dont les valeurs s'échelonnent entre -1 et +1. Dans ce cas, une valeur proche de +1 indique une corrélation positive forte (les deux variables évoluent dans le même sens), une valeur proche de -1 traduit une corrélation négative forte (les variables évoluent en sens inverse), tandis qu'une valeur proche de 0 suggère une absence de lien linéaire entre les variables (Rodriguez, 2023).

Via une fonction Python, nous avons donc mis en place cette matrice entre l'ensemble des paramètres de voies existants et disponibles, et nous avons donc pu conclure sur le choix de ceux-ci dans la construction de notre modèle (Sowiński et al., 2021). Ainsi, nous avons décidé de traiter les paramètres suivants, décrits plus en détails encore en *ANNEXE 3*:

- **Le rayon de courbure *:*** Il mesure, comme son nom l'indique, la courbure de la voie ; ainsi, plus le rayon est faible, plus la courbe est serrée, ce qui accentue les contraintes mécaniques exercées sur les rails, en particulier sur les flancs ;
- **La pente *:*** Elle indique l'inclinaison longitudinale de la voie. Une forte pente peut modifier les efforts d'adhérence et d'accélération, et donc influencer l'usure ;
- **Le dévers *:*** Il correspond à l'inclinaison transversale de la voie dans les courbes. Il est conçu pour compenser la force centrifuge, et son mauvais dimensionnement peut accroître l'usure latérale ;
- **La vitesse *:*** Ce paramètre reflète la vitesse maximale autorisée sur le tronçon. Elle influence directement les efforts dynamiques appliqués aux rails, et donc leur dégradation ;
- **La fréquence de passages *:*** Elle désigne le nombre de rames circulant sur un tronçon et sur une période donnée. Il s'agit d'un facteur direct de sollicitation de la voie, et donc un prédicteur important de l'usure ;
- **La distance entre bavettes** ** ***:*** Elle est une mesure liée à la géométrie de la voie, et peut influencer la manière dont les charges sont réparties et transférées au rail ;
- **Le type de pose** : Il désigne la méthode de mise en œuvre des rails (pose sur traverses béton, bois, voie ballastée ou voie en dalle). Chaque type de pose possède des caractéristiques mécaniques différentes, influençant la rigidité verticale et latérale de la voie. Une pose rigide peut engendrer des contraintes plus importantes sur le rail, accélérant

certaines formes d'usure, tandis qu'une pose plus souple peut mieux absorber les efforts, mais s'user différemment.

- **Le type de rail** *:* Il distingue principalement les rails vignole, utilisés surtout en milieu ferroviaire classique, mais pas que, des rails à gorge, plus fréquents en environnement urbain. Le rail à gorge est conçu pour s'insérer dans la chaussée, ce qui le rend plus exposé aux pollutions, aux chocs latéraux et à l'encrassement, pouvant accélérer l'usure. Le rail vignole, quant à lui, offre une géométrie plus adaptée aux circulations à haute vitesse et subit des sollicitations différentes, notamment verticales.

L'ensemble de ces paramètres, pris ensemble, forment donc le cœur des prédicteurs, variables utilisées dans notre modèle d'apprentissage automatique pour prévoir l'usure des rails. Il en demeure d'autres cependant, ici non pris en compte, car non disponibles en l'état – à l'instar par exemple de pluviométrie, du taux d'humidité ou de la température, ou tout autre paramètre liée à la météorologie ; mais également du graissage ou non du rail par l'appareil de mesure, par exemple. Ce manque certain constitue donc une première limite à la perfection de notre modèle.

[*Files de rails : Voir Glossaire]
[**Bavette : Voir Glossaire]

3.3.3. Insertions préliminaires : type de voies, puis limitation de vitesse, et nombre de passages de rames par an

La seconde étape de la Data Preparation a été d'ajouter certaines données manquantes nécessaires au bon déroulé du processus. En effet, après lectures et discussions avec diverses personnes du service, il a été conclu que le nombre de passages de rames et la limitation de vitesse en chaque point – donc les vitesses effectives, car les tramways sont toujours censés rouler au maximum de leur vitesse autorisée – devaient être ajoutées à la base de données initiale.

Pour cela, il a fallu en première instance trouver la voie associée à chaque point issu de la campagne DEUTZER. Pour rappel, une ligne de tramway comporte généralement deux voies distinctes : une pour les tramways qui circulent dans un sens, et une autre pour ceux qui vont dans le sens inverse. DEUTZER possède un logiciel qui classe ses sorties par fichiers. Chacun de ces fichiers en question correspond à un trajet effectué par l'appareil de mesure du prestataire. L'idéal aurait donc été que chaque trajet ait été réalisé sur la portion d'une même voie, soit Voie 1, dans un sens, soit Voie 2, dans l'autre ; mais évidemment, tel n'était pas le cas. Certains fichiers étaient à cheval sur les deux voies, certains hors des voies, certains sur des voies de retournement*, certains sur des voies uniques, certains sur tout cela à la fois… Ainsi, il a fallu, pour chaque fichier DEUTZER (au nombre de 67 en tout), attribuer un identifiant unique à chaque point, classer ces points par ordre croissant via Excel, les sauvegarder en CSV, les reporter sur un fichier QGis, les comparer avec les voies de tramways effectives chargées depuis un dossier fourni par l'AO, et reporter le type de ces voies (classées en V1 pour Voie 1, V2 pour Voie 2, VU pour Voie Unique et VR pour Voie de

Retournement) pour chacun des points du fichier dans la base de données via une requête SQL. Cette partie du travail a été longue et fastidieuse, mais a permis, par la suite, d'établir le nombre de passages de rames par voie entre chaque arrêt, et la limitation par tronçon et par voie également. Grâce à cela, nous avons donc pu aboutir à une limitation de vitesse point par point, ainsi qu'à un nombre de passages de rames pour la même granularité ; et savons, qu'à l'avenir, il sera nécessaire d'exiger la réception de fichiers classés directement par voies.

---

* [Voies de retournement : voir Glossaire]

3.3.4. Insertions complémentaires : types de poses, types de rails, années de mise en service, et années de maintenance

Dans un second temps, d'autres données ont pu être ajoutées ; le type de pose, c'est-à-dire pose sur béton ou pose sur ballast (explicitées en partie 2.2. puis 3.3.2.) – obtenues grâce aux fichiers PDF complémentaires transmis par l'AO. Ensuite, ont été déterminés les types de rails, à savoir champignon ou à gorge (également décrits en 2.2. puis en 3.3.2.), cette fois-ci grâce à un fichier Excel plutôt complet établi par plusieurs personnes du service. Enfin, grâce à des documents tierces fournis par divers participants au projet, les années de mise en service des différentes lignes du réseau, ensuite utilisées pour déterminer les dernières années de maintenance en date, ont également pu être exploitées.

La dernière étape de cette phase complémentaire a donc consisté à recourir à un document Excel annexe, établi par une collègue d'un autre pôle, et regroupant les types de maintenance effectuées sur certains points du réseau, dans un dernier système géométrique par encore rencontré. Il a fallu, une fois de plus, établir une symétrie entre le système utilisé ici et le système WGS 84 tel que présenté précédemment. Grâce au travail de la collaboratrice, du nom de Florence ZUBILLAGA, et après quelques heures passées sur le sujet, une colonne *ANNEE_DERNIERE_MAINTENANCE* a pu être établie. Elle traduit, comme son nom l'indique, les dernières années en date où des opérations de maintenance ont été effectuées sur chaque point du réseau. Pour construire cette colonne, nous avons d'abord utilisé les informations présentes dans le fichier Excel annexe, lorsque celles-ci étaient disponibles. À défaut, la dernière année de maintenance a été supposée correspondre à l'année de mise en service de la ligne concernée — certaines lignes datant de moins de dix ans et pouvant donc être considérées comme relativement récentes Cette colonne *ANNEE_DERNIERE_MAINTENANCE* nous a ensuite permis d'affiner la résolution temporelle du traitement dans la construction de notre modèle.

3.3.5. Construction par vectorisation et similarité

Par la suite, force à malheureusement été de constater qu'il existait peu de données exploitables pour les fichiers de REX obtenus grâce à TeamFer, le second prestataire technique du projet. Pourtant, un bon modèle prédictif doit user d'un nombre de données important pour être fiable. De fait, pour obtenir des données additionnelles, nous avons pensé à une procédure annexe : le principe a été de trouver, pour chaque point dont nous n'avions pas de mesures effectives pour les années 2021, 2022 ou 2023, un autre point, celui qui se rapprochait le plus, en terme de paramètres communs, dudit point.

Pour ce faire, nous avons utilisé ce que l'on appelle une mesure de similarité, c'est-à-dire un outil mathématique nous ayant permis d'évaluer à quel point deux objets sont ressemblants. Contrairement à la distance Euclidienne par exemple, cette mesure est davantage une valeur qui indique le degré de ressemblance entre deux points ; plus cette valeur est élevée donc, et plus les objets sont similaires. Nous avons utilisé, pour notre part, *cosine similarity,* qui évalue en quelque sorte l'angle entre deux vecteurs. Dans notre cas, nos vecteurs ont été composés des paramètres à prendre en compte par notre modèle, tels que détaillés précédemment. Grâce à cette mesure de similarité, on obtient, pour chaque point, le point le plus proche de lui en terme de produit scalaire, donné grâce à la formule :

$$\textit{cosine similarity} = \frac{\vec{A} \cdot \vec{B}}{\left|\vec{A}\right| \cdot \left|\vec{B}\right|}$$

avec $\vec{A}$ et $\vec{B}$ les vecteurs associés à la combinaison de paramètres points

Ainsi, cette méthode d'imputation par similarité a permis d'enrichir notre base de données avec des valeurs estimées à partir des profils les plus proches, en s'appuyant sur des caractéristiques techniques communes. Bien que relative, cette approche reste justifiée dans un contexte de données partiellement manquantes, et offre une solution technique viable permettant de préserver la robustesse de l'apprentissage du modèle. Elle permet in fine de mieux exploiter les données issues du REX et de garantir une meilleure représentativité des cas réels dans l'entraînement de ce modèle.

3.3.6. Nettoyage des données

Enfin, la dernière étape de la préparation des données a consisté en un travail d'épuration, visant à ne conserver que les entrées exploitables par le modèle. La première phase de ce nettoyage a été réalisée directement via une requête SQL sur SnowFlake *[Figure 5].* Celle-ci permettait d'exclure toutes les lignes contenant des valeurs manquantes (NaN) sur les prédicteurs de l'usure. Grâce à cette sélection, seules les données complètes et a priori cohérentes ont été retenues pour la suite du traitement.

Cependant, un second nettoyage, cette fois intégré dans le pipeline du modèle, c'est-à-dire la séquence organisée d'étapes successives qui permettent de transformer des données brutes en résultats exploitables, restait nécessaire. En effet, certaines incohérences subsistent parfois malgré

l'absence de valeurs manquantes : par exemple, des combinaisons de paramètres physiquement impossibles, des doublons non détectés ou encore des valeurs extrêmes qui, bien que techniquement renseignées, n'ont pas de sens opérationnel. Ce nettoyage interne permet donc de fiabiliser davantage les données avant l'entraînement du modèle, et d'éviter ainsi des biais ou erreurs dans les résultats finaux.

Figure 5 : *Requête SQL permettant d'affiner les données – NEDELEC Amélie*

3.4. Modeling

3.4.1. Les familles de modèles

L'étape suivante du travail a été de choisir un modèle de traitement de la donnée. De ce fait, il a fallu parcourir un certain nombre de thèses, d'études ou d'ouvrages afin de choisir proprement et selon nos aspirations le modèle le plus convainquant. Ainsi, nous avons pu classer les méthodes existantes en 3 grandes catégories, à savoir, en premier lieu, les lois mathématiques ; ensuite, nous avons discuté des modèles s'appuyant sur des lois statiques ; et, enfin, des modèles basés sur l'apprentissage informatique. Il convient ici de décrire en quelques mots l'ensemble de ces divers modèles (Morize, 2020).

A. Lois mathématiques ou physiques relatives à l'usure

La particularité des lois physico-mathématiques est que, comme pour l'ensemble des sciences dites dures, elles sont déterministes. En clair, cela signifie que, lorsque que l'on connaît l'entrée, on connaîtra toujours la sortie. Cette approche présente de nombreuses qualités, notamment une prévisibilité forte, pour les raisons évoquées précédemment ; une modélisation simple, car étant une fonction simple avec des coefficients connus ; et de fait, une rapidité de calculs factuelle : il suffit de rentrer la formule dans Excel, de l'appliquer à l'ensemble des entrées fournies, et on aura les sorties en 1 clic. Enfin, il convient de souligner la valeur pédagogique du processus, car facilement explicable et donc transmissible à ses pairs. Une description pareille aurait tendance à démontrer qu'il s'agit d'une très bonne méthode ; cependant, tel n'est pas le cas dans notre situation, pour la simple raison que les lois mathématiques reposent souvent sur des hypothèses idéales : conditions constantes, matériaux homogènes, environnement stable… Or dans la réalité du monde, l'usure dépend d'une multitude de facteurs non linéaires et variables dans le temps : trafic irrégulier, défauts locaux, météo, maintenance passée, etc. Le risque majeur est donc une mauvaise

prédiction dans les cas non standards, une faible robustesse au bruit, et une adaptabilité fragile ; donc difficilement déployables sur un grand réseau hétérogène.

Malgré ces inconvénients certains, ce sont souvent ces lois qui sont choisies par les équipes chargées de la prédiction d'usure, faute de mieux. Certaines d'entre elles sont néanmoins plus représentatives de la réalité que d'autres, et génératrices d'erreurs seulement mineures, comme :

La loi d'Archard, qui modélise l'usure par frottement, via la formule $V = \frac{K * F * s}{H}$

avec $V =$ le volume d'usure (mm³)
$K =$ le coefficient d'usure (dépend matériau + environnement)
$F =$ la force normale (N)
$s =$ la distance de glissement (m)
$H =$ la dureté du rail (MPa)

Avec cette loi, l'usure est proportionnelle :

- à la force normale appliquée (plus on appuies fort, plus ça s'use) ;
- à la distance de glissement (plus il y a de frottement cumulé, plus ça s'use) ;
- et inversement proportionnelle à la dureté du matériau (plus c'est dur, moins ça s'use).

→ Le coefficient K y est est empirique, issu d'essais, et reflète le comportement mécanique du couple de matériaux.

Bien que liée à des propriétés mesurables, et donc étant facilement adaptable, cette loi repose comme explicité auparavant sur des hypothèses très simplificatrices, telles qu'un glissement continu, ou un contact roue-rail simple, ce qui est souvent faux. Elle permet donc une estimation générale utile mais pas extrêmement précise. Elle suppose également avoir la valeur de K, qui s'obtient exclusivement en faisant des essais expérimentaux sur des bancs d'essais, ce qui nécessite à la fois du temps et de l'argent (Guediche, 2017). Il est néanmoins important de relever que cette loi est en partie prise en compte dans le calcul d'usure des voies de train par l'autorité organisatrice (AO) la plus connue à ce jour, à savoir la SNCF (Remillieux, 2010). D'autres lois physico-mathématiques, comme le modèle classique énergétique**,** sont également utilisées par les plus grands du domaine. Plusieurs documents abordent ce sujet, notamment la thèse de Xavier MORIZE, déjà mentionnée précédemment, intitulée « Contribution à une approche patrimoniale de la voie ferrée de tramway », qui traite ce thème de manière très approfondie (Morize, 2020).

B. Lois statistiques relatives à l'usure

Contrairement aux lois déterministes précédemment évoquées, les lois statistiques reposent sur une approche probabiliste. Elles ne cherchent pas à prédire précisément la quantité d'usure à un instant donné en fonction de paramètres physiques, mais plutôt à modéliser la variabilité et l'incertitude liées à la durée de vie ou à la défaillance d'un élément soumis à l'usure. Ces lois sont issues de l'analyse des données historiques, expérimentales ou de terrain, et visent à fournir des informations sur la probabilité que l'usure dépasse un certain seuil au cours du temps.

Parmi les lois statistiques les plus utilisées figure la **loi de Weibull** (Morize, 2020), très répandue en ingénierie pour modéliser la durée de vie des composants soumis à des phénomènes d'usure, fatigue ou rupture.

$$S(t) = exp\left[\left(\frac{t}{\lambda}\right)^{k}\right]$$

avec $S(t)$ = la probabilité de survie à l'instant t
λ = le paramètre d'échelle (représentant une durée caractéristique)
et k = le paramètre de forme qui modifie la distribution (formant la courbe de probabilité d'usure)

L'intérêt majeur de la loi de Weibull est d'intégrer la variabilité intrinsèque des phénomènes d'usure, due à la complexité des conditions d'exploitation (variabilité du trafic, défauts locaux, météo, qualité des matériaux, interventions de maintenance…). En cela, elle permet de mieux gérer le risque lié à la maintenance en fournissant des intervalles de confiance sur la durée de vie restante, et d'optimiser les calendriers d'intervention en fonction de probabilités de défaillance plutôt que de simples prévisions moyennes ; le résultat étant, en effet, un tableau de données donnant, par année, le pourcentage de rails devant être remplacés sur le réseau.

Toutefois, cette approche statistique nécessite une base de données importante et représentative des événements d'usure passés, ce qui peut être contraignant dans certains contextes où les mesures sont rares ou peu fiables. Par ailleurs, elle ne fournit pas directement une compréhension physique du phénomène, mais une modélisation du comportement global, ce qui limite son usage à des applications statistiques et de gestion. Qui plus est, déterminer les paramètres λ et k de la loi nécessite de faire appel à des méthodes statistiques qui cherchent les valeurs desdits paramètres en maximisant la probabilité d'observer les données mesurées. Elle requiert, ainsi, une connaissance intermédiaire des lois statistiques en général et des logiciels permettant de les appliquer, comme Python.

En synthèse, la loi de Weibull, plus amplement décrite en *ANNEXE 2*, et plus largement les lois statistiques en général, complètent les approches déterministes en apportant une vision probabiliste essentielle dans la gestion réaliste de l'usure, surtout pour les systèmes complexes et hétérogènes ; mais ne permet pas de déterminer précisément quelques portions du réseau doit se voir bénéficier du changement. Elle est donc une première étape d'un travail de maintenance, ou peut servir d'outil de comparaison avec un autre modèle, mais ne peut en rien devenir une fin en soi. L'étude de cette loi a, pour ma part, été la première étape dans l'appréhension du sujet en général.

C. Processus de Machine Learning

Enfin, la dernière grande famille de modèles de traitement de données repose sur les techniques dites de Machine Learning, que l'on traduit en français par apprentissage automatique. À la différence des deux approches précédentes, ces modèles ne reposent ni sur des équations

physiques déterministes, ni sur des distributions statistiques préétablies. Ils apprennent directement les relations entre les variables d'entrée (comme la géométrie de la voie, le trafic, la météo, la maintenance...) et la variable de sortie (l'usure) à partir des données historiques disponibles, sans formuler de lois explicites.

Concrètement, cela signifie que l'on entraîne un algorithme informatique à « reconnaître » les formes d'usure passées à partir d'un grand ensemble de données (le jeu d'entraînement), pour qu'il puisse ensuite prédire l'évolution future de l'usure sur d'autres tronçons. Il existe de nombreux types d'algorithmes supervisés pouvant être utilisés à cette fin, les plus connus étant les arbres de décision, les forêts aléatoires (Random Forest), les régressions par Gradient Boosting, ou encore les réseaux de neurones. Le modèle choisi parmi ceux-ci l'a été grâce à une évaluation de métriques, détaillée ci-après *[Partie 3.5].*

D. Le choix

Parmi les propositions précédentes, on retient que chacune d'entre elles présentait à la fois forces et faiblesses. Cependant, dans la dynamique de recherche et d'innovation prônées par Artelia, par mon maître de stage et répondant souvent aux idées d'un stage en générale, c'est le modèle de Machine Learning qui a été retenu comme étant celui à mener à bien.

À ce stade du projet, après un peu plus d'un mois de stage, mes supérieurs m'ont mis en relation avec Madjid BENALLEGUE, alternant en Intelligence Artificielle chez Artelia à Choisy-le-Grand. Son expertise a rapidement contribué à améliorer le processus de Machine Learning, et ce tout au long du développement du modèle, qu'on a nommé RailWise. Cette collaboration s'est avérée particulièrement productive, dans les deux sens.

3.4.2. Le Machine Learning (ML) et l'Intelligence Artificielle (IA)

L'intelligence Artificielle, c'est ce qui désigne l'ensemble des techniques permettant à des machines (ordinateurs, robots,…) de simuler certaines capacités humaines, comme raisonner, apprendre, ou résoudre des problèmes. Elle n'est pas, contrairement à ce que l'on pense généralement aujourd'hui, un simple outil qui permettrait à l'utilisateur d'obtenir des réponses à une demande via un bot, comme ChatGPT en serait le représentant ; elle est plus globalement un ensemble de techniques permettant de faciliter son travail à l'homme en automatisant certaines tâches cognitives, en améliorant la prise de décision à partir de grandes quantités de données, ou encore en optimisant des processus complexes difficilement maîtrisables manuellement.

L'IA se subdivise aujourd'hui en deux grandes sous catégories ; la plus connue, et la plus spectaculaire, l'IA générative, qui contient une notion de création. L'IA générative crée en effet du contenu : elle apprend à partir d'une grande quantité de données, puis génère des sorties nouvelles, cohérentes avec ce qu'elle a appris. Elle repose cependant grandement sur la seconde sous-catégorie de l'Intelligence Artificielle, celle qui nous concerne directement : le Machine Learning (ML). Le ML, c'est un un sous-domaine qui consiste à entraîner une machine à apprendre à partir de données.

Plutôt que de la programmer explicitement, on lui fournit des exemples pour qu'elle généralise des règles ou des comportements.

A. ML non-supervisé

Dans ce processus de ML, on a deux sous-possibilités : le ML supervisé, et non supervisé. Le ML non supervisé, qui consiste à analyser des données sans connaître les réponses attendues, est souvent, dû à la manière dont il traite l'information, moins précis. Le but est en effet de trouver des structures cachées : des regroupements, des similarités, des tendances. Pour faire simple, on ne donne que les entrées au modèle, il doit se débrouiller pour comprendre à quoi ressemblent les sorties. Il est souvent utilisé quand on a aucun retour d'expérience préalable, d'où la conformité très relative aux résultats finaux observables par la suite.

B. ML supervisé et classification

Dans notre cas, nous avions accès à de l'information sur les sorties possibles du modèle (détaillées en partie 3.3.5.) ; ainsi, nous avons pu utilisé le ML supervisé. En quelques mots, cette forme d'apprentissage consiste à entraîner un modèle à partir d'exemples dont on connaît déjà le résultat attendu. Il se subdivise de nouveau en deux sous-catégories ; la première des deux étant la classification, qui se fait sur des données discrètes et catégorielles.

→ *Exemple illustratif :* imaginons un modèle de classification chargé d'identifier si un bâtiment appartient à Paris ou à Marseille, sur la base de ses caractéristiques architecturales, géographiques ou historiques. Si ce modèle est uniquement entraîné avec des données relatives à ces deux villes, il ne pourra jamais prédire qu'un bâtiment appartient à Tours, même si ce dernier est très proche d'un bâtiment marseillais sur certains points.

Autrement dit, le modèle ne sait travailler qu'avec les catégories qu'on lui a fournies, ce qui illustre bien la nature catégorielle (et fermée) des sorties dans ce type d'apprentissage supervisé.

C. ML supervisé et régression

La deuxième sous-catégorie du ML supervisé est la régression, en d'autres termes celle qui nous concerne directement : le modèle est régressif lorsqu'il est en capacité de produire une donnée continue. Dans notre cas, il prédit avant tout une vitesse d'usure, variable totalement continue car pouvant prendre n'importe quelle valeur ; bien que certaines d'entre elles doivent par la suite être exclues car manquant parfois de sens. *Par exemple,* une vitesse d'usure négative n'a pas de sens : cela reviendrait à dire que le rail gagnerait en matière au fil des ans sans opération de maintenance, ce qui est absurde.

Pour résumer, voici un petit schéma *[Figure 6]* montrant l'arborescence de ce qu'est l'intelligence artificielle en générale, et mettant en exergue ce qu'est notre modèle dans tout cela :

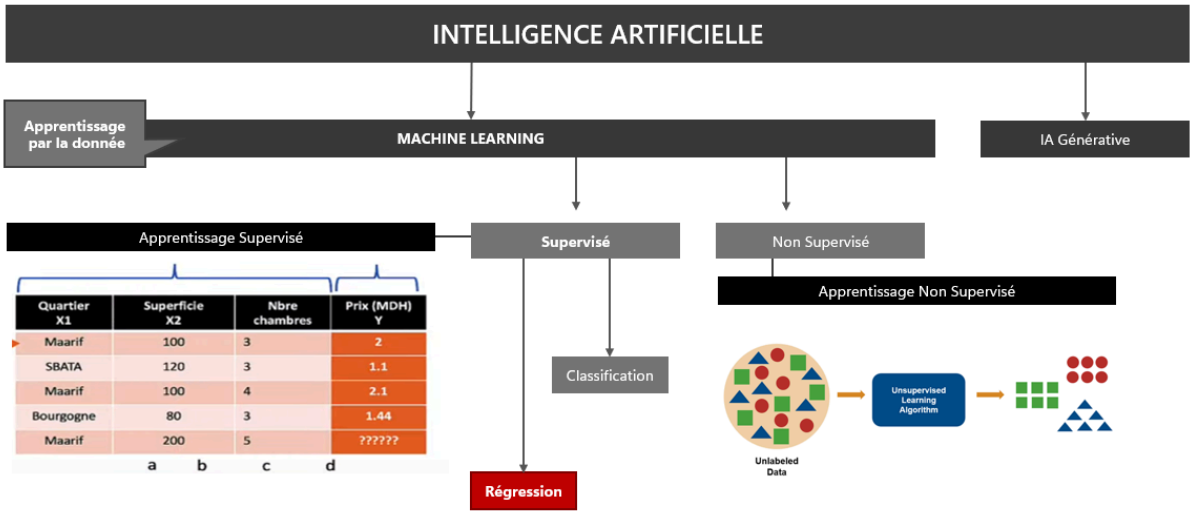


Figure 6 : *L'intelligence Artificielle et ses branches – BENALLEGUE Madjid*

3.5. Evaluation : tests, confrontation de modèles, et choix finaux

Afin d'arriver à un choix final du modèle, des retours et évaluations itératives ont été faits tout au long du processus, permettant in fine de vérifier la robustesse de la démarche. Dans notre cas, les évaluations se structurent autour de plusieurs axes complémentaires, permettant à la fois de juger des performances métriques du modèle mais également juger de sa cohérence métier.

3.5.1. Validation croisée

Avant de retenir l'algorithme final utilisé dans notre cas de figure, plusieurs options ont été explorées. Quatre grandes approches de régression ont ainsi été testées et comparées, afin d'évaluer leur capacité à modéliser correctement l'évolution de l'usure. Ce travail comparatif a permis de mieux cerner les avantages et limites de chaque méthode, et d'identifier celle offrant le meilleur compromis entre performance, robustesse et interprétabilité.

- Tout d'abord, le **Linear Regression,** modèle statistique simple dont l'idée est d'ajuster une droite minimisant les écarts entre valeurs réelles et valeurs prédites, dans notre cas utilisé comme modèle de référence pour évaluer l'impact d'approches plus complexes en comparaisons – et ce bien que nous le sachions peu fiable dans notre cas car modélisant effectivement des relations linéaires, ce qui n'était clairement pas le cas ici, nous l'avons vu plus tôt ;

- Ensuite, le **Random Forest Regressor,** algorithme reconnu pour sa robustesse à des données hétérogènes. Le principe de l'algorithme est de faire exister une forêt, donc un ensemble d'arbres. Chaque arbre de cette forêt est construit à partir d'un échantillon aléatoire du jeu

de données *(méthode dite du bagging),* et à chaque nœud, le modèle sélectionne aléatoirement un sous-ensemble de variables pour déterminer la meilleure séparation. La prédiction finale est ensuite obtenue par moyenne des prédictions de tous les arbres ;

- Et enfin, le **Gradient Boosting Regressor**, plus précisément dans sa version améliorée Hist Gradient Boosting Regressor, qui constitue une méthode d'ensemble itérative et très performante. Le principe fondamental repose sur l'ajustement séquentiel d'arbres de décision faibles, chaque nouvel arbre cherchant à corriger les erreurs résiduelles des arbres précédents en minimisant une fonction de perte spécifique. Contrairement à la forêt aléatoire où les arbres sont construits indépendamment, le gradient boosting construit les arbres de manière dépendante et additive, ce qui permet une convergence plus rapide vers un modèle précis. La version « Hist » optimise cette approche en utilisant une discrétisation intelligente des données continues (histogrammes), ce qui accélère considérablement le processus d'apprentissage tout en maintenant, voire en améliorant, la qualité des prédictions.

Les performances de ces modèles ont ensuite été évaluées à l'aide des métriques évoquées ci-après. De manière générale, Hist Gradient Boosting et Random Forest Regressor sont sortis du lot. Le premier des deux modèles a été choisi au second pour des raisons de performance et de précision accrues, notamment sur des jeux de données comportant des relations complexes et non linéaires. En effet, l'apprentissage séquentiel propre au gradient boosting permet une meilleure capture des résidus et des effets croisés entre variables, là où la forêt aléatoire, bien que robuste, peut parfois lisser excessivement les prédictions. De plus, l'implémentation optimisée de la version *Hist* permet de traiter efficacement de grands volumes de données tout en limitant les temps de calcul, ce qui en fait un candidat de choix dans le cadre de notre application prédictive (Morize, 2020). .

3.5.2. Choix des métriques d'évaluation

Pour juger de la pertinence du choix de modèle comme décrit précédemment, il a fallu en première instance faire tourner ledit modèle en y intégrant une évaluation de métriques. Le choix de celles-ci dépend beaucoup du modèle intrinsèquement ; puisque notre objectif de modèle est de prédire des valeurs numériques continues, en l'occurrence une épaisseur d'usure, le modèle choisi a été, on l'a vu, un modèle de régression. Il existe de fait des indicateurs conçus directement pour évaluer la régression. Pour notre part, nous avons décidé d'en choisir 3 différents :

- La **RMSE**, ou Root Mean Squared Error, traduit par erreur quadratique de moyenne, qui pousse le modèle à éviter de grosses erreurs car pénalise fortement ces dernières en les élevant au carré. Donnée par la formule suivante, la RMSE est très sensible aux valeurs extrêmes ; seule, elle ne suffit donc pas à évaluer pleinement le modèle.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(y_i - \widehat{y}_i\right)^2}$$ avec y_i la valeur réel observée et $\widehat{y}_i$ la valeur prédite

- Le **MAE**, ou Mean Absolute Error, en français l'erreur absolue de moyenne, qui permet de mesurer l'écart moyen, en unités de mesure, entre la prédiction et la valeur réelle. Elle n'amplifie pas les grosses erreurs comme la RMSE, et est donc davantage robuste, mais ne suffit une fois de plus pas seule à l'évaluation du modèle. La formule associée est la suivante :

$$MAE = \frac{1}{n} \sum_{i=1}^{n} \left| y_i - \widehat{y}_i \right|$$

- Le **R^2** enfin, ou coefficient de détermination, permet quant à lui de mesurer la part de variance des données expliquée par le modèle. C'est un indicateur de performance globe, indépendante de l'échelle. Lorsque que le R^2 tend vers 1, cela veut dire que le modèle a tendance à bien expliquer la variabilité de l'usure, ce qui est une bonne chose. Dans la forme, cela donne :

$$R^2 = 1 - \frac{\sum_i \left(y_i - \widehat{y}_i\right)^2}{\sum_i \left(y_i - \bar{y}\right)^2}$$ avec $\bar{y}$ la moyenne des valeurs réelles observées

Dans notre cas de figure, on évalue le modèle deux fois, une pour TABLE et une pour FLANC. Cela donne, pour le modèle définitivement choisi, les valeurs suivantes :

	RMSE	MAE	R^2
TABLE	0.065	0.077	0.935
FLANC	0435	0.287	0.814

En termes d'interprétation, pour l'usure de FLANC, le coefficient de détermination R^2 = 0,814 indique que le modèle explique un peu plus de 81 % de la variance observée. Cela traduit un pouvoir explicatif solide, bien qu'il subsiste une marge pour améliorer la capture de certaines variations plus fines des usures latérales. La MAE de 0,287 mm signifie qu'en moyenne, le modèle se trompe d'environ trois dixièmes de millimètre, un écart relativement faible au regard des amplitudes habituelles d'usure ferroviaire. La RMSE de 0,435 mm met cependant en évidence quelques écarts ponctuellement plus importants, sans remettre en cause la cohérence globale des prédictions.

Pour l'usure de TABLE, les performances sont encore meilleures : R^2 = 0,935 traduit une capacité à expliquer près de 94 % de la variance verticale mesurée. La MAE de 0,077 mm correspond à une erreur moyenne très faible, et la RMSE de 0,065 mm confirme que la dispersion des erreurs est particulièrement réduite. Cela témoigne d'un ajustement très fiable sur cette dimension d'usure.

En résumé, le modèle fournit des résultats robustes sur les deux dimensions : verticale (TABLE) et latérale (FLANC). Si la précision est légèrement supérieure pour la TABLE, les estimations obtenues

pour le FLANC restent pleinement exploitables. Cette double évaluation offre ainsi une vision complète et précise de l'état d'usure des rails, apportant une base solide pour orienter des décisions de maintenance plus ciblées, anticipées et efficaces.

3.5.3. Mise en cohérence métier

Afin de valider non seulement la performance statistique du modèle mais également sa cohérence avec les réalités du terrain, une confrontation a été réalisée avec l'intuition métier d'une personne tierce impliquée dans le projet. Cette experte, qu'on a nommée précédemment, forte de plusieurs années d'expérience dans le domaine de l'usure ferroviaire, a mobilisé à la fois les données techniques parcellaires dont elle disposait (issues notamment de la plateforme TeamFer), et ses connaissances empiriques, issues de l'observation régulière des tronçons et de l'analyse des incidents passés.

Cette phase de validation métier a permis :

- D'abord de de confirmer certaines tendances identifiées par le modèle, notamment l'accélération de l'usure latérale dans les zones en courbe serrée ou fortement sollicitées ;
- De plus, de repérer des cas de sous-estimation ou de surestimation localisée, parfois dus à des effets non pris en compte dans les données ;
- Enfin, d'alimenter une réflexion partagée sur l'interprétation des résultats : les prédictions ont ainsi été perçues non comme des vérités absolues, mais comme des indicateurs d'alerte et d'appui à la décision, à confronter à d'autres sources d'information internes.

En ce sens, cette mise en cohérence métier a joué un rôle crucial dans l'acceptabilité du modèle, en montrant qu'il ne se contentait pas de fournir des chiffres, mais qu'il traduisait des logiques observables par les praticiens du ferroviaire. Ce dialogue entre données et expertise terrain a également contribué à faire émerger des axes d'amélioration pour la suite du projet, comme l'intégration de nouvelles variables explicatives liées aux maintenances ou aux événements perturbateurs (pannes, ralentissements, etc).

3.6. Deployment

3.6.1. Déploiement technique local

Une fois le modèle pré-conçu en algorithmique, c'est-à-dire en idées de déroulement de l'information, il a fallu le retranscrire sous forme de codage informatique. Pour ce faire, nous avons choisi d'utiliser le langage de programmation Python, dans un environnement local, accessible aux équipes via l'éditeur Visual Studio Code (VS Code). Ce choix s'explique par la richesse des bibliothèques scientifiques disponibles, la lisibilité syntaxique du langage, et sa forte intégration dans les pratiques de la data science appliquée à l'ingénierie. Cependant, une fois rédigé en Python, le code a pu, sous couvert de quelques légères modifications, être retranscrites dans des Notebooks Snowflake. L'idée derrière ce processus était de faire en sorte que la prochaine personne à même de l'utiliser pourra le faire de manière simplifiée.

L'architecture du pipeline a alors été conçue de manière modulaire, c'est-à-dire que ses différentes étapes (préparation des données, vectorisation, modélisation, prédiction, détection des interventions, export des résultats) sont organisées sous forme de fonctions indépendantes et réutilisables, comme en témoigne *l'ANNEXE 3.* Cette structure modulaire permet une lisibilité accrue du code, une maintenance facilitée, ainsi qu'une adaptabilité fonctionnelle : chaque composant peut être testé, modifié ou remplacé indépendamment, sans compromettre la stabilité de l'ensemble.

Dans cette continuité de rationalisation des processus, il convient de souligner que le script a été conçu pour intégrer de nouveaux jeux de données (nouvelles courbes, nouveaux tronçons) de manière semi-automatique. Cette flexibilité est essentielle pour une utilisation pérenne sur des réseaux évolutifs ou étendus.

Enfin, à l'issue de l'exécution du code, les résultats peuvent être exportés sous forme de fichiers .csv directement exploitables, structurés par courbe, par type d'usure (usure de table, usure de flanc), et par année prévisionnelle. Ces fichiers peuvent ainsi être mobilisés pour des analyses complémentaires, ou intégrés dans des outils métiers tels que des systèmes d'information géographique (SIG) ou des tableaux de bord de planification de la maintenance.

3.6.2. Jeux de données futurs : concepts RAILADAPT et RAILSTART

Dans la perspective d'un déploiement à plus grande échelle du modèle développé par l'équipe, deux conventions ont été proposées afin de structurer les futurs jeux de données selon leur rôle dans le pipeline de traitement :

- ***RAILADAPT*** désigne un format cible regroupant des données historiques complètes, comportant à la fois les observations d'usure sur plusieurs années et les variables explicatives techniques (trafic, géométrie, matériaux, etc.). Ce format est destiné à l'entraînement, à la calibration et à la validation du modèle.

- ***RAILSTART*** correspond à un format de données partiellement renseignées, souvent sans mesure d'usure, et représente typiquement de nouveaux tronçons ou des extensions du réseau. Ce format est dédié à l'application directe du modèle en prédiction seule, sans réentraînement.

Ces deux structures visent à normaliser l'intégration de données dans des contextes futurs de mise à jour ou d'extension du modèle, selon qu'il s'agisse d'améliorer ses performances (***RAILADAPT***) ou de le mobiliser en exploitation (***RAILSTART***).

3.6.3. Tests sur les données réelles

Avant leur mise en œuvre concrète, ces formats ont été définis en parallèle d'une phase de test sur les données historiques du réseau existant. Le modèle a ainsi été appliqué à des segments

disposant d'observations d'usure sur plusieurs années, enrichies d'informations techniques, comme nous avons pu en discuter tout au long de ce rapport.

Les prédictions générées ont été confrontées aux données observées afin d'évaluer, de manière qualitative, la capacité du modèle à reproduire la dynamique réelle d'usure. L'analyse a notamment permis d'identifier les tronçons où les courbes modélisées reflétaient fidèlement les évolutions constatées.

Il convient néanmoins de préciser que les formats ***RAILADAPT*** et ***RAILSTART***, bien que déjà formalisés à ce stade du projet, n'ont pas encore été mobilisés dans ces tests. Ils ont vocation à structurer les flux de données dans des phases ultérieures, pour le réentraînement du modèle ou son déploiement opérationnel à l'échelle du réseau.

3.6.4. Perspectives d'évolution du déploiement

Ces premières expérimentations ont ainsi permis de valider le fonctionnement du modèle sur des cas concrets, tout en identifiant les pistes d'amélioration possibles pour renforcer sa robustesse et son caractère générique. Dans cette idée, se pose désormais la question de son intégration à plus grande échelle dans les processus métiers, en tirant parti des socles de structuration précédemment évoqués. C'est dans cette optique que plusieurs pistes d'évolution ont été envisagées pour favoriser un déploiement pérenne et automatisé du modèle.

En effet, la mise en œuvre actuelle du modèle reposait initialement sur une exécution locale dans un environnement Python standard, ce qui facilitait le prototypage rapide, les ajustements méthodologiques et la validation des résultats en conditions réelles. Cette approche, efficace en phase de développement, présentait toutefois des limites en termes de passage à l'échelle et de continuité opérationnelle. Dans une logique d'industrialisation, plusieurs axes ont été concrétisés afin de renforcer l'intégration, l'accessibilité et la durabilité du modèle :

- Automatisation du pipeline : Une chaîne de traitement entièrement automatisée, depuis l'ingestion des nouvelles données jusqu'à la génération des prévisions, est désormais opérationnelle. Elle fiabilise la mise à jour du modèle, réduit les risques d'erreurs liés aux interventions manuelles et permet d'augmenter la fréquence des analyses pour suivre l'évolution du réseau en quasi temps réel.

- Intégration dans des outils métiers : Le modèle est maintenant déployé au sein d'une application web dédiée et d'un plugin QGIS interactif. Ces supports permettent aux utilisateurs non techniques de consulter et d'exploiter les résultats de manière autonome (cartographie des prévisions, filtrage par seuils d'usure, simulation d'interventions, etc.), renforçant ainsi son adoption au quotidien par les services techniques.

- Systèmes de suivi et d'alerte : Les seuils métiers définis pour les interventions alimentent à présent des systèmes automatisés de notification et de visualisation anticipée. Ces alertes, fondées sur les dynamiques d'usure prévisionnelles, optimisent les campagnes de

maintenance en dépassant la logique d'intervalles fixes.

Ces avancées marquent une étape clé dans la valorisation des données et l'optimisation de la maintenance ferroviaire, en inscrivant durablement le modèle dans les pratiques de gestion des infrastructures.

3.6.5. Interface métier

L'intégration du modèle dans les processus opérationnels ne se limite pas à un simple déploiement technique ; elle implique également la mise à disposition d'outils accessibles et adaptés aux usages réels des équipes terrain. Dans cette perspective, une interface métier a été développée afin de constituer le point de contact privilégié entre la puissance analytique du modèle et l'expertise métier des utilisateurs finaux.

Cette interface, conçue selon des principes de clarté et de facilité d'usage, regroupe l'ensemble des fonctionnalités nécessaires à la consultation, l'exploration et l'exploitation des résultats. Elle a été conçue en StreamLit, à savoir une bibliothèque Python spécifique, et stockée sur notre plateforme de traitement SnowFlake. Elle permet notamment, comme on le voit sur la figure suivante *[Figure 7] :*

- Visualisation cartographique dynamique : affichage des prévisions d'usure sur l'ensemble du réseau, avec codes couleur et filtres interactifs.
- Recherche et filtrage avancés : accès ciblé par année, possibilité de zoomer pour voir l'information par courbes.
- Simulation d'interventions : estimation de l'impact d'une opération de maintenance sur l'évolution prévisionnelle de l'usure.

Figure 7 : *Cartographie Interactive des usures et années de maintenance prévues – © BENALLEGUE Madjid et NEDELEC Amélie*

En rendant l'information directement exploitable, l'interface favorise la prise de décision rapide et collaborative, tout en réduisant la dépendance à des compétences techniques avancées. Elle constitue ainsi un levier essentiel pour assurer l'appropriation du modèle et son inscription durable dans la chaîne de valeur métier.

3.6.6. Prise de recul

A. Limites identifiées

Le principal frein à l'efficacité du modèle réside dans la qualité et la complétude des données utilisées. Certaines variables clés, telles que la vitesse réelle des rames, les perturbations ponctuelles du trafic, ou encore les données météorologiques, sont actuellement absentes ou insuffisamment renseignées. Cette lacune limite la capacité du modèle à prendre en compte l'ensemble des facteurs influençant l'usure, ce qui contraint sa précision intrinsèque.

Par ailleurs, bien que le modèle ait été calibré sur un historique important relatif au réseau étudié, son application à de nouveaux tronçons ou à d'autres réseaux présentant des caractéristiques géométriques, des profils de trafic ou des types de rails différents peut engendrer un biais ou une perte de performance, du fait d'une représentativité limitée des données d'apprentissage.

B. Perspectives d'évolution

Pour améliorer la robustesse et la pertinence du modèle, plusieurs axes d'évolution sont envisagés. En premier lieu, l'enrichissement du jeu de données constitue une priorité, en intégrant notamment des données complémentaires telles que le trafic réel (en remplacement des estimations théoriques), les historiques détaillés des incidents et des interventions, ainsi que les conditions climatiques locales. Cette démarche vise à mieux refléter la complexité opérationnelle du terrain et à affiner la qualité des prédictions.

Par ailleurs, il est prévu d'élargir le champ d'application du modèle à d'autres réseaux ferroviaires ou zones géographiques, afin d'en tester la généralisation et d'identifier d'éventuels ajustements nécessaires. Une adaptation dynamique du modèle, prenant en compte les spécificités locales, pourrait ainsi être développée pour garantir une utilisation pérenne et fiable.

4. CONCLUSIONS ET OUVERTURE

4.1. Bilan du stage

Pour conclure, notons que ce stage de fin d'études s'est surtout démarqué par la profondeur de l'intégration qu'il m'a procurée dans l'univers professionnel de l'ingénierie. Positionné à la croisée des infrastructures de transport et de la science des données, ce poste m'a donné l'occasion d'améliorer et d'approfondir mes aptitudes dans divers domaines : l'informatique, en structurant des bases de données et en appliquant des algorithmes d'apprentissage automatique, la cartographie, sans oublier l'ingénierie des transports dans son ensemble.

De plus, le respect strict du processus CRISP-DM m'a appris à organiser un projet de sa conception à sa mise en œuvre. Des erreurs ou des imprévus ont pu se produire en chemin, néanmoins, ils ont précisément été des opportunités pour rebondir et apprendre.

Cette expérience de stage, comme décrite ici, m'a offert l'opportunité de relever divers défis, tant sur le plan technique que conceptuel. J'ai dû trouver un équilibre entre la précision informatique et les contraintes professionnelles, tout en apprenant à adapter ma communication en fonction de mon interlocuteur. En affrontant ces défis, j'ai donc pu améliorer ma faculté d'adaptation, ma rigueur méthodologique et mon aptitude à vulgariser des sujets scientifiques.

En outre, le fait de collaborer étroitement avec une personne ayant des compétences différentes des miennes a renforcé ma capacité à travailler en équipe et à faire preuve de flexibilité sur certains aspects ; ces compétences, il ne fait aucun doute, seront essentielles pour la suite de mon parcours. Ces expériences humaines et professionnelles m'ont aidé à forger une approche ingénieure plus holistique, apte à jongler avec les exigences techniques tout en considérant les dynamiques de groupe.

4.2. Apports du stage à la recherche appliquée

En plus des aptitudes personnelles déjà mentionnées, ce stage participe aussi au champ plus large de la recherche appliquée et à l'innovation dans le secteur en général. Effectivement, en considérant les techniques et instruments numériques exploités et élaborés durant cette expérience, on saisit l'importance des ressources technologiques pour appuyer les équipes techniques, présentement et dans le futur. Dans ce contexte, le modèle prédictif élaboré aide effectivement à réduire aussi bien la charge de travail que la charge mentale associée à l'organisation des actions, permettant ainsi aux employés de se focaliser sur des activités plus valorisantes ou d'améliorer leur efficacité.

De plus, en adoptant une approche de solution numérique opérationnelle, ce rendu pave la voie à une incorporation plus approfondie des outils informatiques du Machine Learning dans l'ingénierie des transports, un secteur où l'innovation est essentielle pour faire face aux défis grandissants d'efficacité, de durabilité et de contrôle de l'impact environnemental.

4.3. Réflexion sur le métier d'ingénieure en Aménagement et Environnement

Ce passage à une ingénierie de plus en plus numérique et connectée a suscité diverses considérations internes, notamment concernant le rôle de l'ingénieur en Aménagement et Environnement, qui doit constamment être redéfini. Cette expérience m'a permis de réaliser qu'au-delà des compétences techniques strictement scientifiques et informatiques, il est crucial d'incorporer les contraintes opérationnelles, économiques et humaines dans la gestion de projet. De plus, j'ai remarqué que la curiosité, l'adaptabilité et le désir d'apprendre sont des qualités essentielles pour exceller dans ce domaine. Il s'agit ainsi de compétences à développer progressivement au fil du temps.

En définitive, ce stage, grâce à divers défis techniques que j'ai dû relever, m'a fermement convaincue de l'importance capitale de la formation continue pour rester en phase avec l'évolution incessante de notre profession ; comme nous l'avons souligné en introduction, elle est en perpétuel changement. De plus, un bon ingénieur est celui qui maîtrise la méthode agile, autrement dit « faire petit, tester rapidement, améliorer régulièrement, collectivement », avec une grande compétence.

4.4. Enjeux environnementaux et empreinte écologique

Cette attitude progressive de l'ingénieur ne se restreint pas uniquement aux éléments techniques ou numériques. Elle s'intègre aussi dans une sensibilisation plus vaste aux responsabilités écologiques liées à la planification territoriale et aux infrastructures de transport. Il est donc de notre devoir, en tant que futurs professionnels, d'élaborer des réponses techniques novatrices qui cherchent à réduire au minimum l'impact environnemental qu'elles pourraient avoir sur les générations à venir ; c'est, encore une fois, l'essence même du développement durable.

Par conséquent, bien que l'usage accru de l'intelligence artificielle et des technologies numériques suscite des interrogations valables sur son empreinte environnementale - particulièrement à cause de la consommation d'énergie associée aux calculs et aux systèmes informatiques - il convient de noter qu'au cours de ce stage, l'emploi de l'IA a été envisagé pour optimiser la performance professionnelle, tout en restreignant les conséquences écologiques potentielles.

De fait, grâce à l'automatisation et la systématisation du traitement des données d'usure, notre modèle prédictif facilite la gestion des interventions de maintenance. Cela permet d'éviter les opérations superflues ou mal orientées, prolongeant ainsi la longévité des infrastructures et diminuant les coûts matériels et énergétiques qui y sont liés. Ainsi, l'application de l'IA s'inscrit dans une approche pratique, où la technologie est utilisée comme un outil pour soutenir la transition écologique et améliorer la gestion des ressources, prouvant que le rendement numérique et la préservation de l'environnement peuvent coexister harmonieusement.

LE MOT DE LA FIN

Ce stage a été bien plus qu'une immersion technique ; il a représenté une étape charnière dans mon parcours de future ingénieure en Aménagement et Environnement. Je tiens à exprimer une nouvelle fois ma sincère gratitude à toute l'équipe pour son accueil, sa disponibilité, et la richesse des échanges tout au long de ces mois. Cette expérience, tant humaine que professionnelle, m'a confortée dans mes choix primaires, et m'ouvre des perspectives stimulantes pour la suite de mon parcours.

Forte des compétences acquises et des réflexions engagées, je me projette avec enthousiasme vers un avenir professionnel où je souhaite continuer à allier expertise environnementale, innovation technique et engagement pour des infrastructures plus durables. Pour la suite, j'aspire à renouer avec mes amours inconditionnelles : la gestion des risques sous toutes ses formes, avec un intérêt particulier pour le domaine maritime. Cela étant, je reste pleinement ouverte à de nouvelles expériences, qu'elles soient proches ou éloignées de ce champ, dans la mesure où elles viendraient nourrir ma vision systémique et décentralisée du métier d'ingénieure et du monde en général.

BIBLIOGRAPHIE

Artelia Group. (2025, 24 juin). Accueil - Artelia Group.
https://www.arteliagroup.com/fr/

Chapter 6 : Cities, settlements and key infrastructure. (s. d.). IPCC.
https://www.ipcc.ch/report/ar6/wg2/chapter/chapter-6/

Guediche, M. (2017, 24 octobre). MOdélisation et Simulation de l'Usure des Outils de Coupe au cours du processus d'enlèvement de matière : Approche expérimentale et numérique (MOSUOC).
https://theses.hal.science/tel-02493414

IBM SPSS Modeler Subscription. (s. d.).
https://www.ibm.com/docs/fr/spss-modeler/saas?topic=dm-crisp-help-overview

Morize, X. (2020, 9 juin). Contributions à une approche patrimoniale de la voie ferrée de tramway.
https://pastel.hal.science/tel-03291396

Offizielle Website der Deutzer GmbH. (2024, 17 juin). Deutzer.
https://www.deutzer.de/

Remillieux, A. (2010, 7 juillet). Explicitation et modélisation des connaissances de conduite de changement à la SNCF : vers une gestion des connaissances pré-réfléchies.
https://theses.hal.science/tel-00693957

Rodriguez, P. A. (2023, août 5). Matrice de corrélation. Statorials.
https://statorials.org/matrice-de-correlation/

Sowiński, B., Stelmach, A., & Chudzikiewicz, A. (2021). Simulation Analysis of the Influence of Changes in Track Parameters on Running Safety of a Rail Vehicle. Energies, 14(18), 5882.
https://doi.org/10.3390/en14185882

TeamFer, la mesure pour le ferroviaire. (s. d.).
http://www.teamfer.com/

ANNEXES

L'ANNEXE 1 présente une notice d'utilisation interne du logiciel SnowFlake, et s'étend donc sur plusieurs pages. Ces dernières sont détaillées ci-contre.

→ Les 12 pages suivantes, numérotées de 1 à 11.

L'ANNEXE 2 présente la loi de Weibull, travaillée en détails au début du stage et donc bien explicitée dans un document interne

→ Les 9 pages suivantes, numérotées de 1 à 9

L'ANNEXE 3 est la notice interne de notre modèle, qu'il faut lire attentivement

→ Tout le reste

NOTICE D'UTILISATION INTERNE DU LOGICIEL

SNOWFLAKE

Par ARTELIA - Pôle GPMR
Edition du 4 août 2025

PARTIE 1 : Mise en Contexte

Snowflake est une plateforme de data cloud qui permet de stocker, gérer et analyser de grandes quantités de données dans un environnement cloud entièrement sécurisé. Elle répond aux besoins traditionnels d'un logiciel de traitement des bases de données (BDD), tout en offrant les avantages de la scalabilité, de la flexibilité et de l'accessibilité propres aux solutions cloud.

APPORTS MÉTIERS PRINCIPAUX :

→ **La centralisation des données :**
Snowflake offre un espace unique et partagé où toutes les données, quelle que soit leur source ou leur format, peuvent être regroupées, harmonisées et exploitées. Cela facilite les analyses croisées, garantit la cohérence des informations, et limite la duplication des fichiers.

→ **La sécurisation des données :**
Grâce à des mécanismes de sécurité intégrés (cryptage, gestion fine des accès, journalisation), Snowflake assure la confidentialité, l'intégrité et la traçabilité des données. Cela en fait un outil conforme aux exigences de sécurité des entreprises .

→ **L'automatisation des traitements :**
Snowflake permet de planifier et d'enchaîner automatiquement des traitements (chargement, nettoyage, calculs, export) via des tâches programmées ou des intégrations avec des outils tiers. Cela réduit le besoin d'interventions manuelles et fiabilise les flux de données.

→ **Un gain de temps pour les équipes :**
Grâce à la rapidité d'exécution des requêtes, à la possibilité de travailler en parallèle sans surcharge, et à l'accessibilité depuis n'importe quel poste connecté, Snowflake permet aux équipes de data, de reporting ou d'exploitation de travailler plus vite, plus efficacement, et en toute autonomie.

À QUI S'ADRESSE CE GUIDE ?

Ce guide est destiné principalement aux équipes techniques et métiers impliquées dans la gestion, l'analyse et l'exploitation des données au sein de l'entreprise. Il s'adresse notamment :

- Aux administrateurs des bases de données en question, responsables de l'intégration, la gestion et la sécurisation des données sur la plateforme Snowflake ;
- Ainsi qu'aux responsables métiers et utilisateurs finaux qui bénéficient d'un accès rapide et sécurisé aux données consolidées pour appuyer leurs décisions opérationnelles.

Ce document a pour objectif de présenter les principales fonctionnalités de Snowflake, ses apports métier, et de fournir un cadre commun pour son utilisation efficace au sein de projets data.

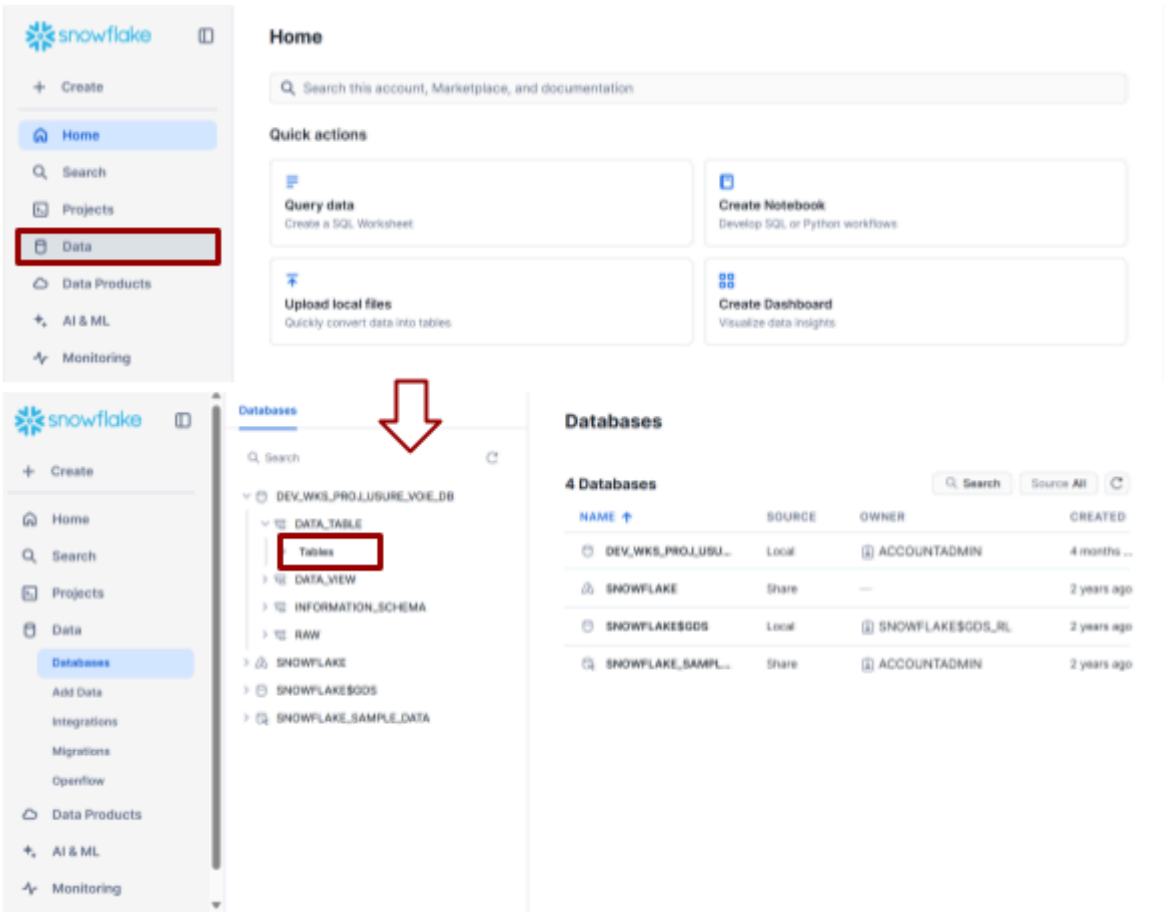
PARTIE 2 : Stocker de l'information dans Snowflake

2.1. Types d'objets pour le stockage

OBJECT	FONCTION	EN DÉTAILS…
Table	Stocker des données structurées	Une table dans Snowflake est comme un tableau Excel : des lignes et des colonnes avec des types de données précis (texte, nombre, date, etc.). On y stocke des données de manière persistante, c'est-à-dire qu'elles restent disponibles même après déconnexion.
View	Sauvegarder une requête	Une vue est une fenêtre sur une ou plusieurs tables. Elle ne contient pas de données en elle-même, mais exécute une requête à chaque fois qu'on l'appelle. Utile pour automatiser des filtres, des jointures ou des calculs.
Stage	Stocker temporairement des fichiers	Une stage est une zone tampon où tu peux déposer des fichiers depuis ton ordinateur (ou depuis le cloud). Ces fichiers peuvent ensuite être chargés dans des tables. C'est comme une salle d'attente pour données.
File Format	Définir le format de lecture des fichiers	Quand tu veux importer un fichier (CSV, JSON, etc.), tu dois expliquer à Snowflake comment lire ce fichier : séparateur, encodage, présence d'un en-tête, etc. C'est ce que fait le file format.

2.1.1. TABLE : Le cœur du stockage

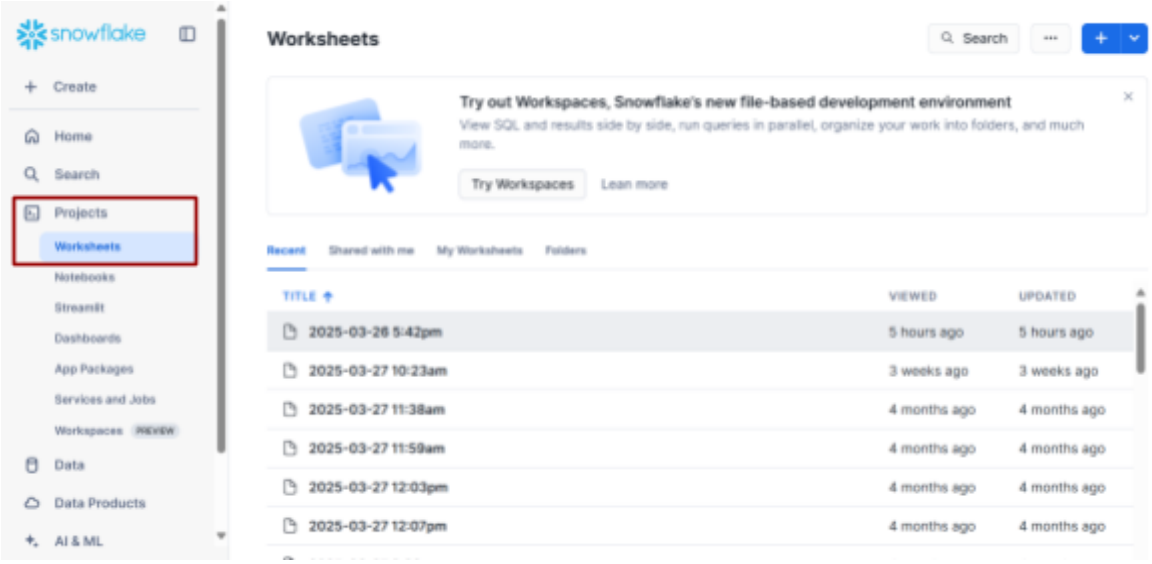
EXEMPLE DE FORME	EXPLICATION
CREATE TABLE employes (id INT, nom STRING, salaire FLOAT);	→ Une table est ***l'objet fondamental de Snowflake*** pour stocker des données structurées. → Elle est organisée en lignes (chaque ligne = un enregistrement) et en colonnes (chaque colonne = un champ, un type de donnée : texte, nombre, date, etc.) ; *c'est un peu comme une base Excel, mais accessible avec du SQL, stockée dans le cloud, interrogeable à haute vitesse.*



→ On peut y accéder simplement en se rendant dans ***Data > Databases > Nom du projet > Tables***

2.1.2. VUE : Une requête toujours à jour

EXEMPLE DE FORME	EXPLICATION
CREATE VIEW employes_salaire_eleve AS SELECT * FROM employes WHERE salaire > 3000;	→ Une vue (ou view) est un objet virtuel dans Snowflake qui permet de ***sauvegarder une requête SQL.*** → Contrairement à une table, une vue ne stocke pas directement de données. Elle agit comme une fenêtre dynamique sur une ou plusieurs tables : à chaque fois qu'on interroge la vue, Snowflake exécute la requête sous-jacente en temps réel. C'est un peu comme un filtre ou un résumé automatique d'une table, toujours à jour, sans dupliquer les données.



→ On peut y accéder simplement en se rendant dans ***Projets > Worksheets***

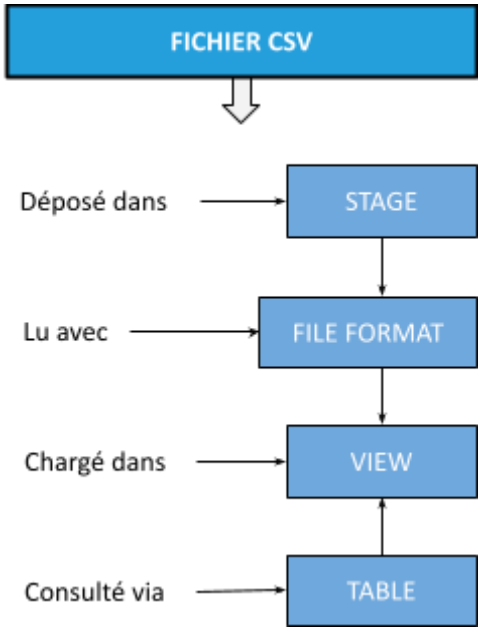
2.1.3. STAGE : Déposer des fichiers pour les charger ensuite

EXEMPLE DE FORME	EXPLICATION
-- Créer une stage (zone de dépôt) CREATE OR REPLACE STAGE ma_stage; -- Ensuite on dépose un fichier .csv (via interface ou ligne de commande) -- Et on charge dans une table COPY INTO ma_table FROM @ma_stage/fichier.csv FILE_FORMAT = (TYPE = 'CSV' SKIP_HEADER=1);	→ Un stage est une zone de stockage temporaire dans Snowflake, utilisée pour déposer des fichiers (CSV, JSON, Parquet, etc.) en vue de les charger dans des tables. → C'est un peu comme un espace tampon entre l'ordinateur (ou un système externe) et la base Snowflake : on y dépose d'abord les fichiers, puis on les importe via des requêtes SQL. C'est une étape clé pour l'intégration de données dans ton entrepôt cloud.

2.1.4. FILE FORMAT: Dire à Snowflake comment lire le fichier

EXEMPLE DE FORME	EXPLICATION
CREATE OR REPLACE FILE FORMAT mon_format_csv TYPE = 'CSV' FIELD_OPTIONALLY_ENCLOSED_BY = '"' SKIP_HEADER = 1;	→ Un file format est un objet de configuration dans Snowflake qui sert à définir les règles de lecture d'un fichier (CSV, JSON, Parquet, etc.) lors de son chargement depuis un stage. → Il permet d'indiquer à Snowflake comment interpréter le contenu d'un fichier : séparateur de colonnes, présence d'un en-tête, encodage, guillemets, types de champs, etc. → C'est presque équivalent à une fiche de lecture que Snowflake va utiliser pour traduire correctement les données brutes dans une table structurée.

2.1.5. RÉSUMÉ VISUEL :

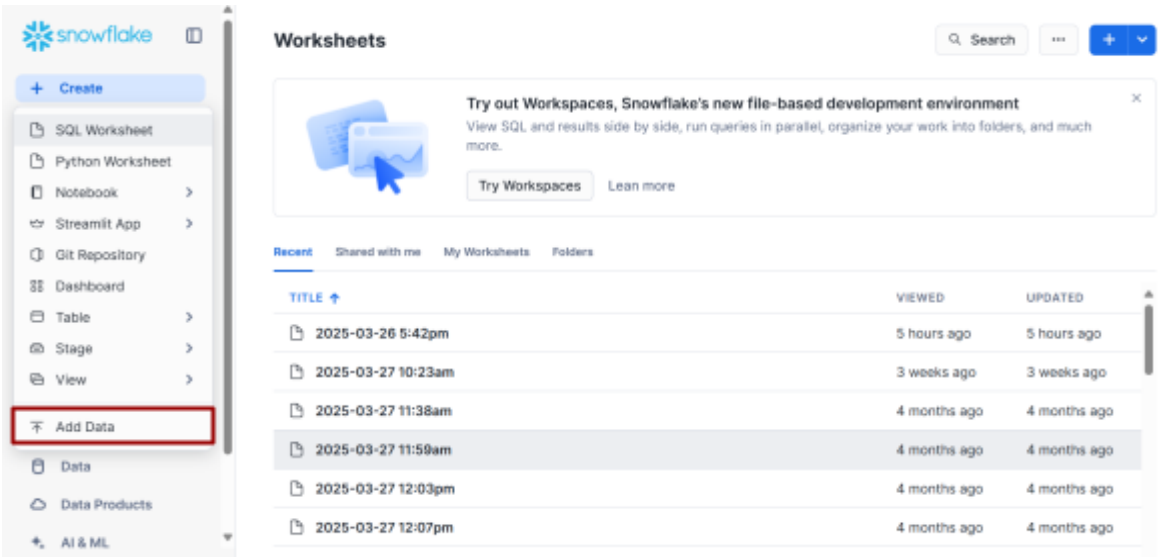


2.2. *Créer une TABLE et y stocker des données*

EXPLICATION	REQUETE
On l'a vu précédemment, pour créer un table, il suffit d'effectuer une requête SQL, et ce dans via ***la manœuvre Projects > Worksheet*** comme pour les Views. *En voici un exemple →*	CREATE OR REPLACE TABLE employes (id INT, nom STRING, salaire FLOAT, date_embauche DATE);

2.3 *Charger un fichier CSV dans une table (étapes)*

EXPLICATION
On peut aussi, plutôt que de créer une table, l'importer directement depuis un format Excel enregistré en CSV. Il suffit d'effectuer ***la manœuvre Create > Add Data > Load Data into a Table > Browse ;*** puis on importe le fichier en question.



PARTIE 3 : Le langage SQL – Les Principes de base

SQL (Structured Query Language) est un ***langage informatique*** standard utilisé pour interagir avec des bases de données relationnelles. Il permet de créer, lire, modifier et supprimer des données organisées en tables. Avec SQL, on peut effectuer des opérations simples comme filtrer des lignes, trier des colonnes, ou plus complexes comme faire des calculs, regrouper des données ou relier plusieurs tables entre elles. C'est un ***langage déclaratif*** : on dit ce que l'on veut obtenir, et le moteur de base de données (comme *Snowflake*) s'occupe de comment l'exécuter efficacement.

3.1. Vocabulaire SQL

En SQL, il est important de connaître certaines requêtes de bases, qui permettent d'effectuer la plupart des opérations possibles grâce à ce langage. En voici les principales :

ELEMENT	DESCRIPTION
SELECT	Lire des données
WHERE	Filtrer des lignes
GROUP BY	Grouper les données
JOIN	Relier plusieurs tables
INSERT	Ajouter des données
UPDATE	Modifier des données
DELETE	Supprimer des données
CREATE	Créer des tables, vues, stages
Le format des colonnes ajoutées aux tables est aussi à connaître :	
INTEGER	Mettre un entier
FLOAT	Mettre un décimal
VARCHAR	Mettre un texte

3.2. Exemples

Ci-dessous, quelques exemples, essentiels pour comprendre le fonctionnement dudit langage :

→ **Lire 10 lignes**	SELECT * FROM pente LIMIT 10;
→ **Filtrer des lignes**	SELECT * FROM courbes WHERE rayon < 300 ;
→ **Faire une moyenne**	SELECT courbe, AVG(pente) FROM table GROUP BY courbe;

PARTIE 4 : Le langage Python – Les Principes de base

Python est un langage de programmation généraliste, très utilisé en sciences des données, automatisation et développement logiciel. Sa syntaxe facilite généralement la lecture et l'écriture des codes générés. Python permet de manipuler des données, d'effectuer des calculs, de créer des visualisations, et de développer des applications grâce à **des bibliothèques** (comme Pandas pour les données, NumPy pour les calculs numériques, ou scikit-learn pour le Machine Learning).

Contrairement à SQL, Python est un ***langage impératif*** et orienté objet, où le programmeur décrit étape par étape comment effectuer une tâche. Sa polyvalence et sa richesse en font un outil essentiel pour le traitement des données dans des projets variés.

4.1. Vocabulaire Python

Voici quelques notions et commandes de base qui sont souvent utilisées dans Python :

ELEMENT	DESCRIPTION
print()	Affiche un message ou une variable à l'écran
for	Boucle permettant d'itérer sur une séquence
if	Condition pour exécuter du code sous certaines circonstances
def	Définit une fonction
import	Permet d'importer des modules externes
list	Structure de données pour stocker des collections ordonnées
pandas.DataFrame	Objet pour manipuler facilement des tableaux de données

4.2. Exemples

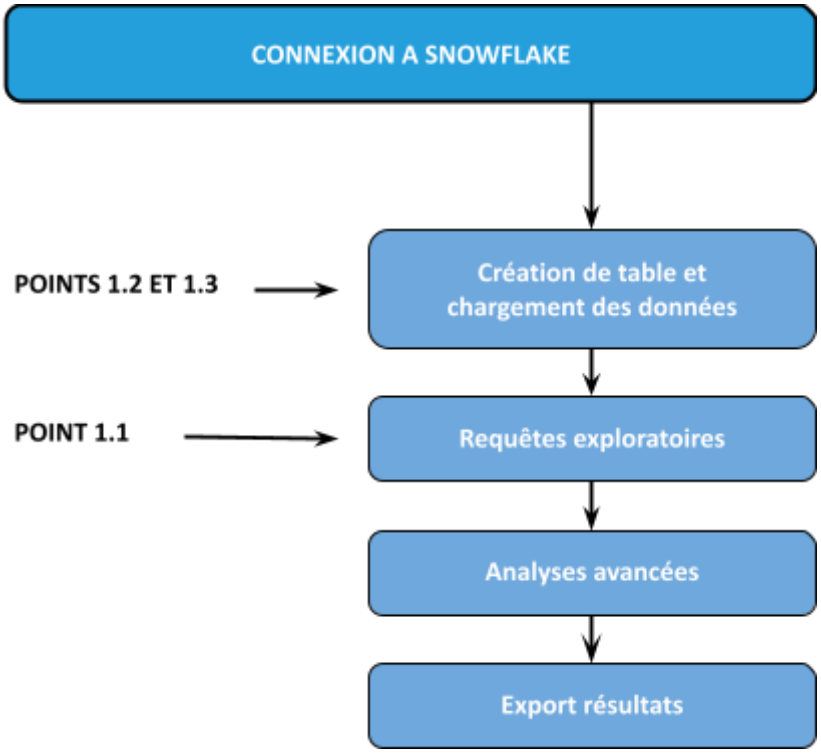
Ci-dessous, quelques exemples, essentiels pour comprendre le fonctionnement dudit langage :

→ Afficher un message	print("Bonjour !")
→ Parcourir une liste et afficher chaque élément	rails = ["vignole", "a gorge"] for i in rails : print(i)
→ Lire un fichier CSV avec Pandas	import pandas as pd df = pd.read_csv("donnees.csv") print(df.head(5)) *# Affiche les 5 premières lignes*
→ Calculer la moyenne d'une colonne	moyenne = df["rayon"].mean() print(moyenne)

PARTIE 5 : Structuration dans un notebook

Un Notebook sur Snowflake est un environnement interactif qui permet d'écrire, exécuter et organiser du code directement connecté à la plateforme Snowflake. Il combine des cellules de code (en SQL, Python ou d'autres langages supportés) avec des éléments de texte explicatif, ce qui facilite l'analyse de données, le prototypage et le partage des résultats.

Grâce au Notebook, les utilisateurs peuvent interroger facilement les données stockées dans Snowflake, visualiser les résultats sous forme de tableaux ou de graphiques, et documenter leur travail dans un même espace collaboratif. C'est un outil puissant pour réaliser des analyses exploratoires, construire des pipelines de données, ou encore développer des modèles d'apprentissage automatique en bénéficiant de la scalabilité et de la sécurité de Snowflake.



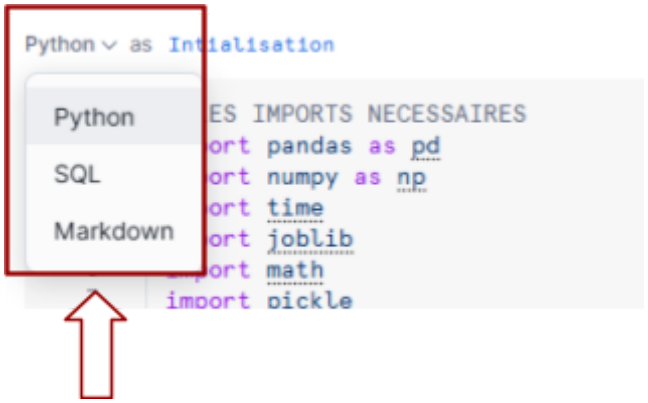
Pour accéder à l'outil Notebook, il faut suivre le parcours ***> Project > Notebook***
⚠️ Et ne pas oublier de spécifier, dans la première cellule, ceci :

```
# L'IMPORT DE SESSION
from snowflake.snowpark.context import get_active_session
session = get_active_session()
```

[A COPIER DIRECTEMENT ↓]

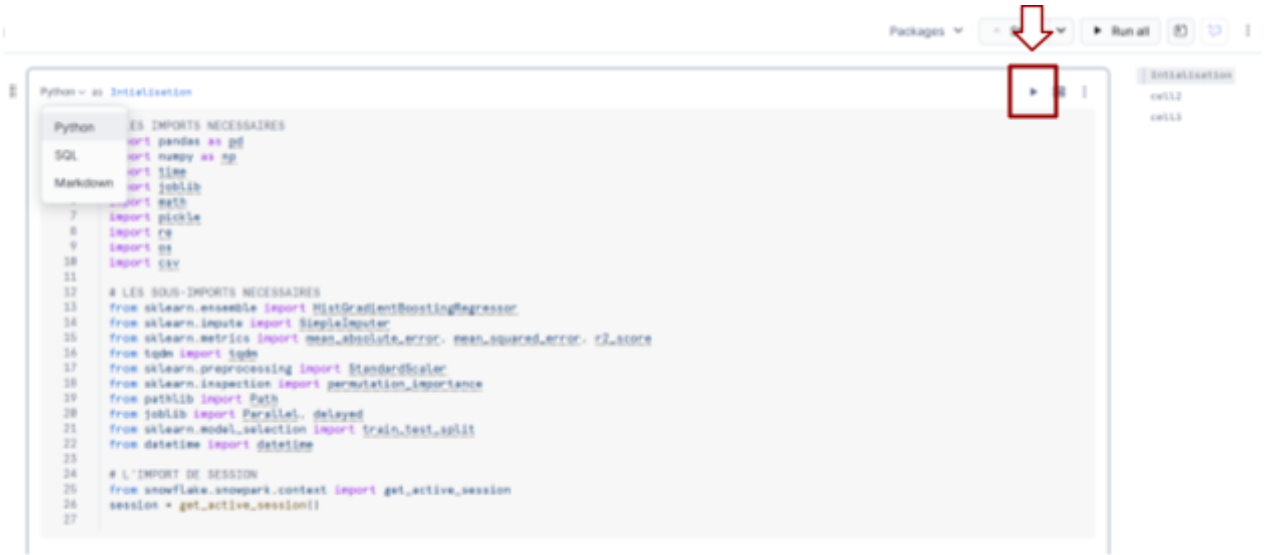
```
from snowflake.snowpark.context import get_active_session
session = get_active_session()
```

→ Chaque nouvelle fonction pourra être copiée dans une nouvelle cellule. Il faudra spécifier le langage utilisé, soit Python, soit SQL, en haut de ladite cellule :



⚠️ ***ATTENTION également à l'exécution des requêtes faites dans les cellules :***

Si certaines actions ne sont qu'à réaliser une seule et unique fois, il vaut mieux ne pas les mettre dans le notebook, sous risque d'écraser ce qui a été fait précédemment. Il vaut mieux utiliser, en conséquence, une Worksheet annexe. Le cas échéant, il est important de cliquer uniquement sur la flèche ASSOCIÉE À LA CELLULE et non pas sur le ❌RUN ALL en haut à droite de l'écran



LOI PARAMÉTRIQUE DE WEIBULL

1. LES LOIS PARAMÉTRIQUES DANS UN CADRE DE MATHÉMATIQUES FONDAMENTALES

La loi de Weibull est avant tout ce que l'on appelle une **loi de probabilité**, c'est-à-dire une règle mathématique décrivant la manière dont une variable aléatoire peut prendre certaines valeurs avec des probabilités associées. Plus particulièrement, la loi de Weibull est une **loi paramétrique**, donc pouvant être décrite par un ensemble fini de paramètres. Ces paramètres gouvernent le comportement de la distribution et permettent de modéliser différents phénomènes aléatoires.

Mathématiquement parlant, une loi paramétrique est définie par une **fonction de densité de probabilité**, que l'on nomme souvent **PDF**, et ou une **fonction de répartition cumulative**, ou **CDF**, dépendant toutes deux d'un vecteur de paramètres. Plus précisément, une PDF, c'est une fonction qui décrit comment une variable aléatoire continue se répartit sur un ensemble de valeurs possibles. En d'autres termes, elle nous dit quelle est la probabilité de tomber dans une certaine plage de valeurs. La CDF est moins précise, puisqu'elle est une fonction rapportant quelle est la probabilité qu'une variable prenne une valeur inférieure ou égale à un certain seuil ; elle est cependant souvent plus utilisée en pratique, notamment lorsqu'il s'agit de prédire des coûts économiques à long terme. Connaître le fonctionnement de ces PDF et CDF pour chaque loi paramétrique étudiée est essentiel dans la mesure où elle permet de comprendre comment obtenir lesdits paramètres de la loi en question *[voir démonstration pour Weibull, ANNEXE 1].*

2. LES LOIS PARAMÉTRIQUES EN PRATIQUE

En pratique, les lois paramétriques sont surtout utilisées pour modéliser des phénomènes aléatoires et réaliser des prédictions de terrain. On y trouve de nombreux avantages, à savoir une **facilité d'interprétation**, étant donné que les paramètres d'une loi ont souvent une signification physique ; et une **réduction de la complexité** d'un système à plusieurs caractéristiques, puisqu'il s'agit en effet d'une modélisation résumant des ensembles de données volumineux en seulement quelques paramètres clés. En revanche, ces lois ont également des limites, dans la mesure où elles requièrent des **hypothèses fortes** (les données doivent suivre une certaine forme pour que le modèle soit valide) et une importante **sensibilité aux erreurs de modélisation**, car une mauvaise estimation des paramètres pourrait induire des prédictions erronées. Ainsi, ces lois sont souvent manipulées afin de **modéliser** grossièrement un système et **prédire** avec des approximations certaines le comportement d'un phénomène.

Leur emploi facilite également les vérifications nécessaires pour la validation du modèle prédictif car permettent d'user dans un second temps et sans complication des **lois statiques** se basant sur les mêmes paramètres.

3. LA LOI DE WEIBULL : ENTRE THÉORIE…

La loi de Weibull étant donc une loi paramétrique, elle dépend logiquement de paramètres précis. Ces paramètres sont ici au nombre de **deux** :

- **Le paramètre de forme, k,** qui détermine comme son nom l'indique la forme de la distribution, c'est-à-dire la pente et le comportement des événements en fonction du temps.
    - Si k = 1, la distribution de Weibull devient équivalente à une distribution exponentielle, où le taux d'événements est constant (absence de vieillissement ou de mémoire)
    - Si k > 1, la distribution devient plus "aiguisée", indiquant que l'intensité des événements augmente avec le temps
    - Si k < 1, la distribution devient plus plate, ce qui signifie que la probabilité d'un événement diminue avec le temps (comportement où les événements rares se produisent plus rapidement au début, puis deviennent moins fréquents).

- **Le paramètre d'échelle, λ,** qui détermine l'étendue de la distribution. Il est lié à la vitesse à laquelle les événements se produisent. En d'autres termes, il ajuste la taille de la distribution.
    - Plus λ est grand, plus les événements sont espacés dans le temps
    - Plus λ est petit, plus les événements se produisent rapidement

La dualité de ces paramètres permet donc à la loi de Weibull d'être suffisamment précise pour être utilisée dans le monde réel (contrairement à la loi exponentielle par exemple, qui n'est utile que dans de très rares cas dans le domaine technique), mais également suffisamment simple pour que lesdits paramètres soient facilement déterminables ; pour ce faire, il existe justement différentes techniques, à savoir :

- **La méthode d'égalisation des valeurs empiriques et théoriques de k et λ** *[voir ANNEXE 1]*, qui permet d'obtenir des paramètres optimaux basés sur l'analyse des données.

- **La méthode MLE (Maximum Likelihood Estimation)**, qui utilise la fonction de vraisemblance. Cette fonction représente la probabilité des données observées en fonction des paramètres du modèle, en supposant que le modèle est correct dès le départ.

- **La méthode informatique, via Python,** qui calcule k et λ à partir des données en utilisant une boîte noire informatique. Cette méthode est souvent considérée comme la plus fiable.

La détermination des paramètres k et λ spécifiques au projet permet ensuite d'utiliser la loi de Weibull à des fins de prédiction, en minimisant les risques d'erreurs liées aux paramètres

4. LA LOI DE WEIBULL : … ET PRATIQUE

4.1. Détermination des paramètres

Les trois méthodes précédemment décrites ont donc été utilisées, afin d'être comparées entre elles et de vérifier leur cohérence interne dans la détermination des paramètres dans le cadre du projet.

Pour rappel, l'idée est de **prédire l'usure des rails en fonction de l'ensemble, ou du moins des plus importantes, caractéristiques du réseau et ainsi prévoir les budgets théoriques de l'exploitant de réseau**. Pour pouvoir déterminer les paramètres k et λ propres à ce réseau et à ce cas de figure, il fallait impérativement avoir accès à un retour d'expérience préalable – à savoir ne prendre en compte, dans le calcul, que des rails ayant d'ores et déjà été renouvelés au moins une fois. Cette information n'était malheureusement disponible dans notre cas que pour 90 observations. Qui plus est, il a fallu distinguer lesdits rails selon leur rayon de courbure (R <= 50 m ; R > 50 m) ; les premiers étant liés à une usure plus rapide en raison des fortes contraintes exercées dessus. De ce fait, on se retrouvait (puisque dans 8 cas on avait un double renouvellement) avec un tableau de **12 données seulement pour les courbes de rayon supérieur à 50 mètres, et 86 pour les autres**.

> *Ce nombre de données est faible ; la détermination des paramètres n'est donc pas aussi fiable qu'on le souhaiterait, car sera systématiquement plus influencée ET par des valeurs extrêmes, ET par une agglomération de données autour d'une valeur précise, agglomération pouvant être expliquée par un comportement anthropique opportuniste de renouvellement de rails.*

Malgré ces limites, on se retrouve avec différentes valeurs obtenues pour k et λ, à savoir :

	MLE		**CDF**			**PDF**			**Python**	
	k	***λ***	***k-W***	***k-V***	***λ***	***k-W***	***k-V***	***λ***	***k***	***λ***
R < = 50 m	3,73	19,76	2,64	3,52	20,56	20,15	4,15	23,64	3,78	19,87
R > 50 m	17,61	23,08	12,19	16,15	23,05	18,94	16,35	23,32	17,61	23,08

→ Ici, le k-W signifie k de Weibull et k-V le k de variance ; la distinction est faite dans la mesure où le k peut être calculé de deux manières [*voir ANNEXE 1*] ; k-W utilisant en effet la fonction solveur d'Excel tandis que k-V n'utilisant que des principes mathématiques mais requérant l'obtention d'un λ en amont. Ainsi, dûes à des imprécisions successives, les k-V ne seront retenus comme fiables dans aucun cas. La méthode MLE étant peu précise sur de petits échantillons, comme ici, elle donnera des résultats qui ne seront pas gardés non plus. Enfin, la valeur aberrante trouvée pour k-W avec la méthode PDF sera proscrite également, bien que la raison de cette imprécision reste trouble. In fine,

les valeurs déterminées par la **méthode Python** seront celles conservées pour la suite du processus de calcul.

4.2. Obtention de résultats

Une fois les coefficients k et λ fixés via les méthodes précédemment décrites, la fonction **LOI.WEIBULL** d'Excel a été utilisée pour prédire l'usure des rails et donc leur remplacement à l'avenir. Ladite fonction se traduit de la manière suivante : **=LOI.WEIBULL ([valeur];[α];[β];[{VRAI ou FAUX}])** avec la "valeur" correspondant aux années de remplacement (allant de 1 à l'infini), α équivalant à notre paramètre k, β à notre paramètre λ – et VRAI ou FAUX dépendant de si l'on veut respectivement avoir une CDF ou une PDF. Dans notre cas, il est préférable de choisir VRAI plutôt que FAUX, car cela permet d'obtenir graphiquement une fonction de distribution cumulée (CDF). Cela favorise une **interprétation plus intuitive** de la situation et permet surtout une **comparaison plus claire des distributions.** En effet, les CDF de deux distributions permettent de voir plus facilement laquelle a une probabilité cumulée plus grande ou plus petite à chaque valeur. En revanche, les fonctions de densité de probabilité (PDF) nécessitent souvent des outils statistiques supplémentaires pour effectuer des comparaisons significatives.

Le but étant d'obtenir un modèle donnant des résultats proches de la réalité, les résultats obtenus ici à partir des paramètres k et λ déterminés par REX vont être comparés aux résultats de l'analyse "à vue d'expert" menée par une agente du service. Des résultats auxiliaires, obtenus par la même LOI.WEIBULL sur Excel, mais à partir de k et λ différents *[voir ANNEXE 2]* seront également comparés aux résultats de la spécialiste ; **afin de conclure sur la pertinence de cette démarche d'analyse statistique parmatérique.**

5. CONCLUSIONS

Graphiquement, les résultats obtenus sont ceux qui figurent page suivante.

Globalement, on constate une **forte dissemblance** entre l'ensemble de ces illustrations. Cependant, on remarque un remplacement plus progressif, donc plus étendu dans le temps, pour les rails relatifs aux rayons de courbure inférieurs à 50 m que pour les rayons de courbure supérieurs à 50 m pour les figures 2 et 3, à l'inverse de ce que relevé sur la figure 1. Ce résultat peut paraître contre-intuitif, dans la mesure où, en raison de cette forme, l'ensemble des rails pour R > 50 m devrait être renouvelé AVANT l'ensemble des rails R <= 50 m. Or, d'un point de vue basé sur un raisonnement logique, il apparaît que les rails de rayons de courbure inférieurs s'usent plus rapidement que d'autres. Comment donc expliquer ces résultats ?

La raison de ces conclusions réside en réalité dans la nature de l'échantillon de départ ; puisqu'en effet, pour R > 50 m, nous n'avons que 12 mesures exploitables, et ce en raison de minces archives concernant ce sujet. Cela explique la montée plus "brusque" de la fonction d'usure, car toutes les mesures se concentrent autour de valeurs avec un Δ très faible entre elles. Malgré ce vice de raisonnement qui malheureusement ne peut être comblé au vu des données exploitables, on peut tout de même apprécier la ressemblance de forme entre les deux dernières figures, et donc **conclure**

sur la pertinence d'utiliser des k et λ en temps que variables d'un réseau à un autre et non comme constantes de départ.

Par ailleurs, nous pouvons aussi conclure sur la nécessité d'avoir un REX complet, avec un nombre de données exploitables important, pour valider précisément la pertinence du calcul de ces k et λ. L'importance d'une base de données plus fournie réside dans l'obtention de conclusions qui pourraient contenir une marge d'erreur, et donc des prévisions faussées entraînant potentiellement des pannes impromptues ou des coûts économiques de révisions trop élevés car inutiles en l'état.

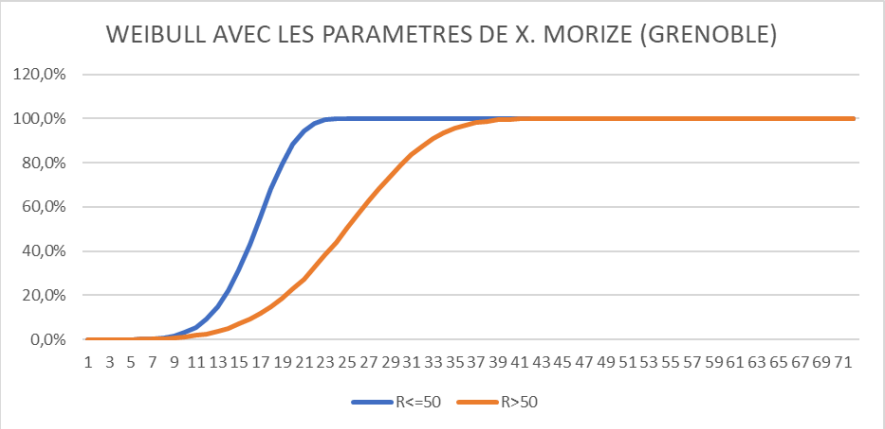


***Figure 1** : Loi de Weibull cumulative à partir des résultats de Grenoble (loi continue)*

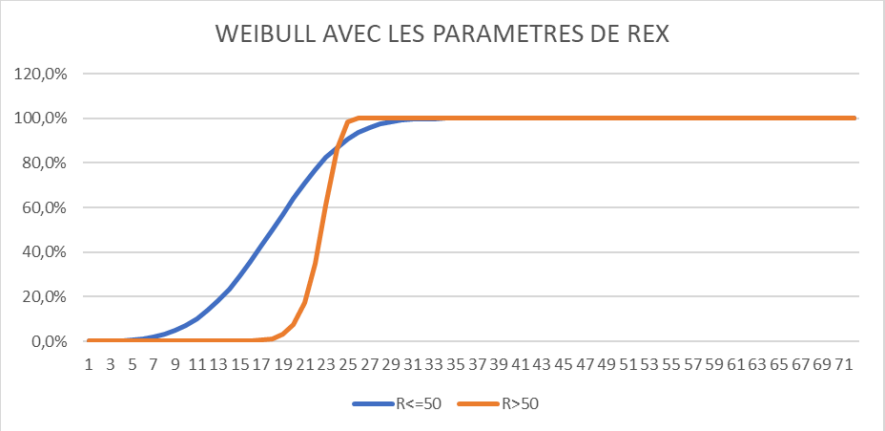


***Figure 2** : Loi de Weibull cumulative à partir des résultats de REX (loi continue)*

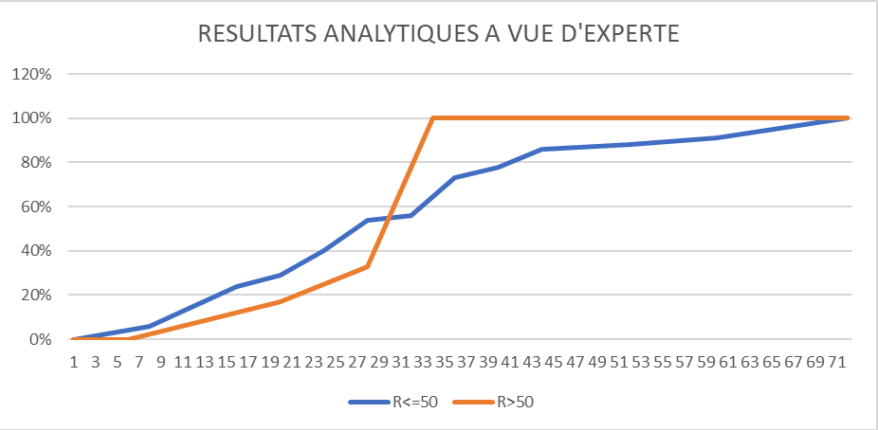


***Figure 3** : Interpolation en loi continue des résultats analytiques venant du REX*

ANNEXE 1

Pour calculer les coefficients k et λ à partir de la loi de Weibull, on va utiliser le principe selon lequel la moyenne de variables aléatoires suivant une même loi de probabilité converge toujours vers l'espérance de cette fonction de densité associée à cette loi. L'espérance, mathématiquement parlant, est la valeur moyenne que l'on peut attendre d'une variable aléatoire après un grand nombre de répétitions de l'expérience. C'est une mesure centrale qui aide à comprendre le comportement attendu d'une variable aléatoire, et c'est de cette espérance que nous allons nous servir pour trouver k et λ. En effet, on va chercher à ce que la somme au carré de la différence entre l'espérance de la loi de Weibull appliquée aux données et leur moyenne converge vers 0, pour minimiser l'écart et de fait obtenir des paramètres optimaux. Pour ce faire, on va utiliser le solveur d'Excel, qui permet d'obtenir des paramètres optimaux pour un résultat demandé.

>>> Mathématiquement, cela se traduit ainsi :

L'espérance de la loi de Weibull se donne sous la forme $E[X] = \lambda\Gamma(1 + \frac{1}{k})$ car :

- L'espérance se donne par la formule générale $E[X] = \int_0^\infty x\, f(x)\, dx$
- La fonction de densité de probabilité f(x) de la loi de Weibull qui se donne par la formule $f(x) = \frac{k}{\lambda}(\frac{x}{\lambda})^{k-1} e^{-(\frac{x}{\lambda})^k}$

De fait, on obtient : $E[X] = \int_0^\infty x \frac{k}{\lambda}(\frac{x}{\lambda})^{k-1} e^{-(\frac{x}{\lambda})^k} dx$

On pose alors un changement de variable, tel que $x = \lambda t^{\frac{1}{k}}$ et on obtient donc $dx = \frac{1}{k}\lambda t^{\frac{1}{k}-1} dt$

Donc $E[X] = \int_0^\infty \lambda t^{\frac{1}{k}} \frac{k}{\lambda}(\frac{\lambda t^{\frac{1}{k}}}{\lambda})^{k-1} e^{-t} \frac{1}{k}\lambda t^{\frac{1}{k}-1} dt$. En simplifiant, on obtient $E[X] = \lambda\int_0^\infty t^{\frac{1}{k}} e^{-t}\, dt$

Or la fonction Gamma d'Euler est de forme $\Gamma(s) = \int t^{s-1} e^{-t} dt$ donc, ici, avec $s = \frac{1}{k} + 1$, on obtient $E[X] = \lambda\Gamma(1 + \frac{1}{k})$

Or, d'après la loi des grands nombres, on sait que lorsque l'on prend un très grand nombre d'observations x1,..., xn issus d'une même loi de probabilité, leur moyenne converge vers E[X]. Or, la moyenne est donnée par $\bar{X} = \frac{1}{n}\sum_{i=1}^{n} x_i = \lambda\Gamma(1 + \frac{1}{k})$. Pour respecter cette convergence, on cherche donc à résoudre $[\frac{1}{n}\sum_{i=1}^{n} - \lambda\Gamma(1 + \frac{1}{k})]^2 \simeq 0$

Comme expliqué précédemment, pour ce faire, on utilise le solveur d'Excel, et c'est ainsi qu'on aboutit aux valeurs de k-W et λ. A partir de cette valeur de λ, une seconde valeur de k peut être trouvée, de la manière suivante :

Puisque l'on sait que la variance d'une variable X est donnée par la formule $Var(X) = E[(X - E[X]^2)] = E[X^2] - (E[X])^2$

Comme $E[X^2] = \lambda^2\Gamma(1 + \frac{2}{k})$ *(résultat obtenu en effectuant une intégration),* on obtient :

$$Var\,(X) = \lambda^2(\Gamma(1 + \frac{2}{k}) - (\Gamma(1 + \frac{1}{k}))^2).$$

Comme précédemment, on cherche à faire converger $Var\,(X) - \lambda^2(\Gamma(1 + \frac{2}{k}) - (\Gamma(1 + \frac{1}{k}))^2)$ vers 0 en utilisant la fonction VAR d'Excel ainsi que la valeur de λ trouvée précédemment, et on détermine de cette manière une nouvelle valeur de k, k-V. On comprend ainsi que k-V est une approximation d'approximation, et donc qu'elle n'est pas vraiment fiable ; d'où le fait qu'in fine elle ne soit pas retenue pour nos calculs suivants.

ANNEXE 2

Résultats obtenus à partir de la thèse de Xavier MORIZE, Contributions à une approche patrimoniale de la voie ferrée de tramway, résultats page 174 – disponibles ici :
https://pastel.hal.science/tel-03291396/file/TH2020PESC1018.pdf

Type de rayon de courbure	η Paramètre d'échelle	β Paramètre de forme	t_ε Jiang's approx.	t_0 Smith's asymptotic approx.
Faible rayon (R<50m)	17,6	6,06	25,17	42,54
Grand rayon (R<300m)	27,11	4,45	38,76	48,53

avec, ici, η correspondant à notre λ et β à notre k

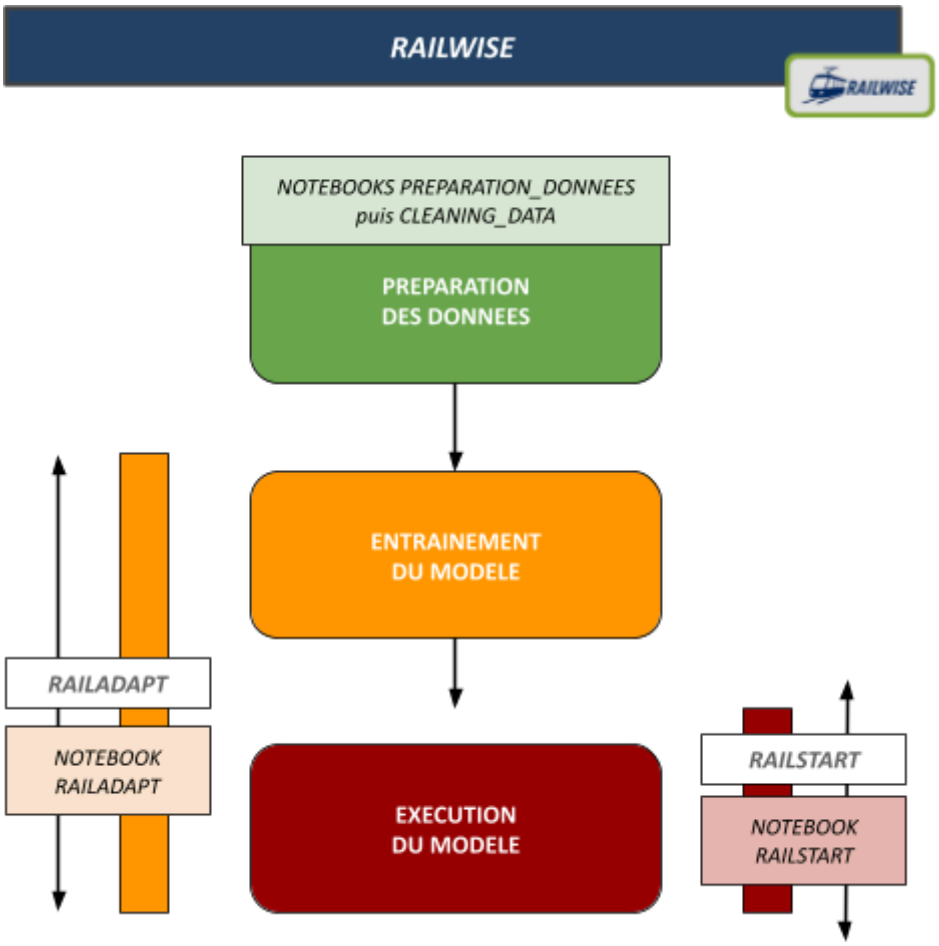
NOTICE D'UTILISATION INTERNE DU MODELE

RAILWISE

Par ARTELIA - Pôle GPMR

Edition du 11 août 2025

RÉSUMÉ VISUEL DU MODÈLE



PARTIE 1 : MISE EN CONTEXTE

1.1. CONTEXTE TECHNIQUE

Pour rappel, la modélisation de l'usure des rails est un enjeu majeur pour la maintenance ferroviaire, y compris urbaine. En anticipant l'évolution de la dégradation des voies, il est possible de planifier des interventions préventives, ce qui réduit les coûts liés aux réparations d'urgence et améliore la sécurité des circulations. Une gestion proactive de l'usure permet également d'optimiser la durée de vie des infrastructures ferroviaires.

1.2. OBJECTIFS DU PROJET

Ce projet vise à développer un modèle prédictif multi-annuel de l'usure des rails, prenant en compte les paramètres d'exploitation essentiels. L'objectif est d'obtenir des prévisions précises permettant d'anticiper les besoins en maintenance et d'adapter les stratégies d'intervention.

1.3. ACTEURS CONCERNÉS

Le projet mobilise une équipe technique interne, des partenaires spécialisés en modélisation et maintenance ferroviaire, ainsi que les exploitants du réseau qui utiliseront les résultats pour optimiser la gestion des infrastructures.

1.4. ENJEUX TECHNIQUES ET STRATÉGIQUES

Sur le plan technique, il s'agit de maîtriser la complexité des données et des phénomènes physiques impliqués dans l'usure des rails. Stratégiquement, la réussite de ce projet permettra de renforcer la performance opérationnelle du réseau, de diminuer les risques d'incidents, et de soutenir la transition vers une mobilité ferroviaire plus durable et économique.

PARTIE 2 : INTRANTS DU MODÈLE ET REQUÊTES ASSOCIÉES

2.1. COMPLÉTUDE DE LA DONNÉE :

Pour garantir la fiabilité de notre modèle, il est essentiel de disposer du maximum de données exploitables. Dans cette perspective, il convient de définir une résolution spatiale adaptée : suffisamment fine pour assurer la précision nécessaire à un modèle fiable, tout en restant raisonnable afin de ne pas compromettre la rapidité du traitement des données. L'exploitant du réseau pourra choisir une résolution spatiale comprise entre le décimètre et le mètre. Concrètement, cela signifie que le système de mesure employé pour recueillir les paramètres de voie doit être capable de fournir les informations demandées tous les décimètres ou mètres.

Le tableau ci-dessous, codé par couleur, précise les paramètres que les prestataires et AO devront fournir, répartis en 4 catégories. Les paramètres ***indispensables*** sont essentiels au fonctionnement du modèle – celui-ci ne doit pas s'exécuter sans eux (il le peut, sous couvert de certaines modifications, mais ne sera pas optimal en ce sens). Les paramètres ***facultatifs*** représentent des compléments susceptibles d'améliorer le modèle, sans en être un prérequis.

Pour meilleure explications, les données en **VERT** devront être fournies par le prestataire technique, en **BLEU** par l'Autorité Organisatrice, et celles en **JAUNE** ne dépendront que de l'application de requêtes manuelles.

	INDISPENSABLES – INTERVENTION PRESTATAIRE
	INDISPENSABLES – INTERVENTION MANUELLE
	INDISPENSABLES – INTERVENTION AO
	FACULTATIFS

2.2. SPÉCIFICATIONS ET FORMATS DES FICHIERS DEMANDÉS :

2.2.1. DEMANDE AU PRESTATAIRE :

Idéalement, les données devraient être fournies sous forme de fichiers structurés en cellules, dans lesquels chaque ligne correspond à une unité géographique unique (plus de détails ci-après), et chaque colonne représente un paramètre d'entrée destiné à alimenter le modèle. Le format CSV est préféré pour sa simplicité et sa compatibilité avec nos outils de traitement, mais tout autre format lisible et structuré pourra également convenir.

Les données MINIMALES attendues par le prestataire sont donc demandées sous cette forme :

LATITUDE	LONGITUDE	PK	FILE	RAYON_ COURBE	PENTE	DEVERS	DISTANCE_ENTRE _BAVETTES
Un point pour chaque mesure faite à la résolution spatiale choisie							

→ On rajoute à ce tableau, si elles sont disponibles, les USURES

Dans le meilleur des cas, le prestataire nous fournira l'ensemble des données précédemment décrites ***AINSI*** que le NOMBRE_PASSAGES par points, le TYPE_POSES par points, le TYPE_RAILS par points, ainsi que les VITESSES par points.

2.2.2. DEMANDE AO :

Il convient également de demander au commanditaire du réseau d'avoir quelques informations supplémentaires susceptibles d'avoir un impact significatif sur les sorties du modèle. Ces fichiers sont les suivants :

> Un ***fichier AO1,*** si possible en CSV, sinon en KML ou SHP (mais demandera des traitements supplémentaires) contenant :

- les coordonnées latitude / longitude (WGS 84),
- le type de point (arrêt, panneau vitesse…),
- la valeur associée (vitesse en km/h ou nom d'arrêt),
- la voie et la ou les lignes concernées.

*Le fichier **AO1** fourni devra ressembler au tableau exemple suivant :*

LATITUDE	LONGITUDE	TYPE DE POINT	VITESSE ASSOCIEE	LIGNE	VOIE
X	Y	Panneau	70	L1	V1

> Un ***fichier AO2***, soit un tableau de correspondance, au format CSV (ou équivalent), où chaque ligne représente une portion de voie ou un point géographique, contenant :

- les coordonnées géographiques (latitude, longitude en WGS84, avec la précision adaptée au pas spatial des mesures),
- la voie et la ou les lignes concernées,
- et le type de rail (ex. vignole, à gorge, type A, type B…).

*Le fichier **AO2** fourni devra ressembler au tableau exemple suivant :*

LATITUDE	LONGITUDE	TYPE DE RAILS	LIGNE	VOIE
X	Y	Type A	L1	V1
X+1	Y+1	Type B	L1	V1
X+2	Y+2	Type B	L1	V1
X+3	Y+3	Type A	L1	V1

> Un ***fichier AO3***, soit une table de données (CSV, Excel ou juste PDF) indiquant, pour chaque segment de ligne entre deux arrêts, le nombre de passages de rames par unité de temps (ex. par jour, par heure, selon la définition du projet), et devant contenir :

- les coordonnées, donc latitude et longitude (en WGS84),
- les noms d'arrêts,
- les lignes et voies concernées

*Le fichier **AO3** fourni devra ressembler au tableau exemple suivant :*

LATITUDE	LONGITUDE	TYPE DE RAILS	LIGNE	VOIE
X	Y	Arret A	L1	V1
X+1	Y+1	Arret B	L1	V1
X+2	Y+2	Arret C	L2	V1
X+3	Y+3	Arret D	L2	V1

> Un ***fichier AO4***, soit une table de données (CSV, Excel ou juste PDF) indiquant, pour chaque position géographique d'une ligne concernée par un arrêt :

- le nom de cet arrêt
- ses coordonnées, donc latitude et longitude (en WGS84),
- et les lignes et voies concernées

*Le fichier **AO4** fourni devra ressembler au tableau exemple suivant :*

LATITUDE	LONGITUDE	ARRET	LIGNE	VOIE
X	Y	Arret A	L1	V1
X+2	Y+2	Arret B	L2	V1
X+5	Y+5	Arret C	L2	V2

> Un ***fichier AO5***, soit un tableau de correspondance, au format CSV (ou équivalent), où chaque ligne représente une portion de voie ou un point géographique, contenant :

- les coordonnées géographiques (latitude, longitude en WGS84, avec la précision adaptée au pas spatial des mesures),
- la voie et la ou les lignes concernées,
- et le type de poses (ballast, béton…)

*Le fichier **AO5** fourni devra ressembler au tableau exemple suivant :*

LATITUDE	LONGITUDE	TYPE DE POSE	LIGNE	VOIE
X	Y	Type A	L1	V1
X+1	Y+1	Type B	L1	V1
X+2	Y+2	Type B	L1	V1
X+3	Y+3	Type A	L1	V1

2.3. LES PARAMÈTRES EXIGÉS :

Pour précision, l'intégralité des informations disponibles dans ce paragraphe le sont également directement en plateforme projet, sur SnowFlake, dans le Notebook PREPARATION_DONNEES .Il conviendra malgré tout de bien lire l'intégralité de ce guide afin de pouvoir travailler en tout état de connaissances.

2.3.1. IDENTIFIANTS UNIQUES, ou ID

Pour garantir la traçabilité et l'intégrité des données dans la nouvelle base, chaque nouvelle ligne insérée se voit attribuer un identifiant unique (ID). Cet ID, généré automatiquement par la base de données, permet de distinguer chaque enregistrement et d'éviter toute ambiguïté lors des analyses ou des mises à jour. Concrètement, cette fonctionnalité est mise en place via une requête SQL **ID** définissant une colonne ID en incrémentation automatique :

ID	
CREATE TABLE DEV_WKS_PROJ_USURE_VOIE_DB.DATA_TABLE.RAW_DATA_{NOM_TABLE} (ID INTEGER AUTOINCREMENT START 1 INCREMENT 1, LATITUDE FLOAT, LONGITUDE FLOAT, PK FLOAT, FILE_RAIL VARCHAR (100), RAYON_COURBURE FLOAT, PENTE FLOAT, DEVERS FLOAT, DISTANCE_ENTRE_BAVETTES FLOAT);	> {NOM_TABLE} sera le nom de notre nouvelle table créée à cette étape

Ainsi, à chaque ajout d'une nouvelle ligne, l'ID s'incrémente automatiquement à partir de 1, assurant une numérotation continue et unique pour l'ensemble des enregistrements.

2.3.2. LES COORDONNÉES GÉOGRAPHIQUES WGS 84

La première demande à formuler auprès du prestataire chargé de la collecte des paramètres nécessaires au modèle concerne les coordonnées géographiques des points relevés. Les appareils de mesure, généralement équipés d'un système RTK GNSS*, doivent impérativement fournir les coordonnées WGS 84 des points.

***RTK GNSS** = Real-Time Kinematic Global Navigation Satellite System.
C'est une technique de positionnement par satellites (GPS, Galileo, etc.) en temps réel qui utilise une station de référence au sol pour corriger les erreurs et obtenir une précision centimétrique.
En clair : c'est du GPS très haute précision grâce à des corrections différentielles.

WGS84 = World Geodetic System 1984.
C'est le système de référence géodésique mondial utilisé par le GPS, qui définit la forme de la Terre, son orientation dans l'espace et un système de coordonnées (latitude, longitude, altitude).
En bref : c'est la "carte de référence" universelle pour positionner un point sur Terre.

La précision exigée dépend directement de la résolution spatiale définie en amont : pour une résolution spatiale d'un mètre, la précision doit être d'au moins 5 chiffres après la virgule ; pour une résolution au décimètre, il faudra 6 chiffres.

Les paramètres X (longitude) et Y (latitude) sont absolument indispensables. Le paramètre Z (altitude) est nécessaire, voire indispensable si aucune donnée sur la pente n'est fournie. Ainsi, les points transmis pourront être sous la forme (X, Y) ou (X, Y, Z), comme dans l'exemple ci-dessous :

EXEMPLE FILE :

LATITUDE	LONGITUDE
48.85890	2.29365
48.85892	2.29366
48.85893	2.29367
48.85894	2.29368

*→ Il est important de noter que les coordonnées GPS sont stockées avec une précision minimale de **5 chiffres après la virgule,** garantissant une précision planimétrique d'environ 1 mètre. Cette résolution est suffisante pour distinguer les deux files de rails d'une même voie. Pour tout calcul de distance, de surface ou de courbure, les données sont projetées dans un système métrique local (par exemple Lambert 93 ou UTM).*

2.3.3. LE POINT KILOMÉTRIQUE (PK)

Pour assurer un suivi longitudinal précis et cohérent de l'usure des rails, chaque point géographique doit être associé à un identifiant linéaire unique appelé PK (Point Kilométrique). Ce PK est exprimé en mètres, calculé à partir d'un point origine géolocalisé dans le référentiel WGS 84, clairement défini pour chaque ligne et chaque voie.

→ Le prestataire doit ***impérativement*** fournir la correspondance entre ce point origine (PK 0) et sa position géographique WGS 84 de référence.

Chaque PK est propre à chaque couple ligne/voie, ce qui signifie que, pour une même position géographique, le PK est distinct selon la ligne et la voie concernées, évitant toute ambiguïté. Le PK doit évoluer de manière strictement incrémentale dans le sens conventionnel de circulation de la voie.

Pour définir la précision des PK, c'est-à-dire le nombre de chiffres après la virgule, on peut s'appuyer sur la résolution spatiale choisie. En divisant la longueur totale de la ligne par la résolution spatiale (distance entre deux points successifs), on obtient le nombre de subdivisions nécessaires, ce qui permet de calculer le nombre minimal de décimales à utiliser pour représenter le PK de manière unique et précise. Par exemple, si la ligne fait 10 km et que la résolution spatiale est de 10 mètres (soit 0,01 km), alors : $\frac{10\ km}{0{,}01\ km} = 1000$ → *Ce qui implique qu'il faut au minimum 3 décimales après la virgule pour exprimer les PK (ex : 0,001 km = 1 mètre).* Toutefois, afin d'anticiper d'éventuelles extensions ou ajouts de portions à la ligne, il est recommandé d'ajouter une décimale supplémentaire. Cela permet de conserver une granularité suffisante pour insérer de nouveaux points sans avoir à renuméroter toute la chaîne de PK, facilitant ainsi la gestion et la maintenance des données dans le temps.

Par ailleurs, pour faciliter la gestion des extensions et les parcours dans les deux sens, il est possible d'adopter une convention où le point origine PK 0 correspond à un repère fixe et stable (par exemple, un arrêt central). Les PK positifs progressent dans un sens (0,0001; 0,0002 ; 0,0003 km…) et les PK négatifs dans le sens opposé (-0,0001 ; -0,0002 ; -0,0003 km…).

Cette approche garantit une représentation linéaire claire et non ambiguë de la position sur la voie, ce qui facilite la fusion de campagnes successives, la comparaison des mesures dans le temps, ainsi que leur exploitation dans un Système d'Information Géographique (SIG).

EXEMPLE FILE POUR LA LIGNE 1 VOIE 1 :

LATITUDE	LONGITUDE	PK
48.85890	2.29365	0
48.85892	2.29366	0
48.85893	2.29367	0.1
48.85894	2.29368	0.1

EXEMPLE FILE POUR LA LIGNE 2 VOIE 2 :

LATITUDE	LONGITUDE	PK
48.85990	2.29375	3
48.85992	2.29376	3
48.85993	2.29377	3.1
48.85994	2.29378	3.1

→ **Si le PK n'est pas disponible ou utilisable :**
Dans ce cas, il est impératif que l'ordre des points dans les fichiers soit strictement ordonné selon leur progression le long de la voie dans le sens de circulation. Cette solution alternative est cependant moins robuste, rend la fusion de données plus complexe, et peut poser des difficultés dans la gestion des voies multiples. Dans cette hypothèse, un identifiant unique (ID) propre à chaque ligne/voie doit être ajouté pour distinguer les données à ***CETTE ÉTAPE***, mais cela ne remplace pas un PK correctement défini.

En résumé, pour garantir la qualité et la facilité de traitement des données d'usure, le PK doit être un identifiant linéaire unique, propre à chaque ligne et voie, avec une origine claire et une progression définie selon le sens de circulation.

2.3.4. LES FILES DE RAILS

Les files de rails devront être indiquées dans le fichier CSV transmis, via une colonne dédiée intitulée « FILE », contenant la mention D ou G selon que la mesure concerne la file de rail droite ou gauche, dans le sens de parcours. Les PK dits uniques ne le seront pas totalement, puisqu'un PK concerna deux points : un D et un G. En effet, les appareils de mesure utilisés sont souvent des bogies instrumentés (type Deutzer) circulant à faible vitesse sur les voies, ce qui permet de collecter au moins deux mesures distinctes par point kilométrique (PK).

EXEMPLE FILE :

LATITUDE	LONGITUDE	PK	FILE
48.85890	2.29365	0	D
48.85892	2.29366	0	G
48.85893	2.29367	0.1	D
48.85894	2.29368	0.1	G

2.3.5. ENSEMBLE VOIE-LIGNE

Pour mieux organiser les données, plusieurs colonnes seront créées dans la base selon les lignes étudiées par le modèle. → Par exemple, si les lignes 1, 2, 3 et 4 d'un réseau sont étudiées, alors les colonnes L1, L2, L3 et L4 seront ajoutées dans la base de données. Concrètement, cela se traduit par une requête similaire à la suivante :

```
ALTER TABLE {NOM_TABLE}]
ADD COLUMN L1 VARCHAR ;
```

← Dans ce cas, L1 peut aussi être L2, L3 ou L4. VARCHAR est le type de la colonne (voir NOTICE SNOWFLAKE), qui en l'occurrence est du texte .

EXEMPLE FILE :

LATITUDE	LONGITUDE	PK	FILE	L1	L2	L3	L4
48.85890	2.29365	0	D				
48.85892	2.29366	0	G				
48.85893	2.29367	0.1	D				
48.85894	2.29368	0.1	G				

La véritable requête associée est la REQUÊTE SQL nommée **LIGNES_AJOUTEES**
Ces colonnes seront ensuite enrichies à partir des observations terrain et des documents fournis, sous la REQUÊTE SQL **VOIES_AJOUTEES**. → Par exemple, en supposant que les points kilométriques (PK) débutant par 0 soient rattachés à la Voie 1 de la Ligne 1, une requête sera exécutée pour compléter la colonne L1 avec la valeur V1 lorsque la condition LEFT(PK, 1) = '0', qui signifie que la première lettre (ou caractère) de la valeur dans la colonne PK est égale à "0", est vérifiée.

EXEMPLE FILE :

LATITUDE	LONGITUDE	PK	FILE	L1	L2	L3	L4
48.85890	2.29365	0	D	V1			
48.85892	2.29366	0	G	V1			
48.85893	2.29367	0.1	D	V1			
48.85894	2.29368	0.1	G	V1			

LIGNES_AJOUTEES	
ALTER TABLE {NOM_TABLE} ADD COLUMN {NUMERO_LIGNE} VARCHAR(100) ;	> Le {NOM_TABLE} désigne le nom de la table > Le {NUMERO_LIGNE} peut être L1, L2… ou même LA, LB ; et correspond au nom de la ligne ou des lignes étudiées. Si il y a plusieurs lignes, alors effectuer la requête plusieurs fois.

VOIES_AJOUTEES	
UPDATE {NOM_TABLE} SET {NUMERO_LIGNE} = '{NUMERO_VOIE} ' WHERE LEFT(CAST(ABS(PK) AS VARCHAR), 1) = '{α}' AND {NUMERO_LIGNE} IS NULL);	> Le {NUMERO_VOIE} désigne, au choix, V1 ou V2 et devra être noté tel quel > Le {α} est la partie entière du PK associé au duo {NUMERO_LIGNE}/{NUMERO_VOIE} De même, la requêtre devra être effectué un nombre de fois équivalent à 2 fois le nombre de lignes

2.3.6. VITESSES

À ce stade, il est impératif que l'autorité organisatrice fournisse la géolocalisation précise (en coordonnées WGS 84, avec la même granularité géographique que précisé en point 2.1) des panneaux de limitation de vitesse, ainsi que celles des stations d'arrêts. Cela permettra de reconstituer, pour chaque tronçon de voie, la vitesse réglementaire applicable, en lien avec les lignes qui y circulent.

Le fichier utilisé à ce stage sera le fichier AO1, c'est–à-dire, pour rappel, le fichier contenant :

- les coordonnées latitude / longitude (WGS 84),
- le type de point (arrêt, panneau vitesse…),
- la valeur associée (vitesse en km/h ou nom d'arrêt),
- la voie et la ou les lignes concernées.

Un traitement dynamique des vitesses sera donc effectué à ce stade, nécessitant de faire appel à une REQUÊTE PYTHON nommée **VITESSES** et qui aboutira à un rendu tel que celui ci-dessous :

EXEMPLE FILE :

LATITUDE	LONGITUDE	PK	FILE	VITESSES
48.85890	2.29365	0	D	12
48.85892	2.29366	0	G	12
48.85893	2.29367	0.1	D	13
48.85894	2.29368	0.1	G	13

VITESSES

```
def haversine_distance(lat1, lon1, lat2, lon2):
    R = 6371000
    phi1, phi2 = np.radians(lat1), np.radians(lat2)
    d_phi = np.radians(lat2 - lat1)
    d_lambda = np.radians(lon2 - lon1)
    a = np.sin(d_phi/2)**2 + np.cos(phi1)*np.cos(phi2)*np.sin(d_lambda/2)**2
    c = 2 * np.arctan2(np.sqrt(a), np.sqrt(1 - a))
    return R * c

def calcul_profil_vitesse_geo(df_points, df_panneaux, acc, dec, step=1):
    # df_points : DataFrame avec colonnes LATITUDE, LONGITUDE, PK
```

```
    # df_panneaux : DataFrame avec colonnes LATITUDE, LONGITUDE, VITESSE_KMH

    # Calculer distances cumulées sur df_points
    distances = [0]
    for i in range(1, len(df_points)):
        dist = haversine_distance(df_points.loc[i-1,'LATITUDE'], df_points.loc[i-1,'LONGITUDE'],
                                  df_points.loc[i,'LATITUDE'], df_points.loc[i,'LONGITUDE'])
        distances.append(dist)
    df_points = df_points.copy()
    df_points['dist_cum'] = np.cumsum(distances)

    # Calculer distances cumulées sur panneaux
    distances_panneaux = [0]
        for i in range(1, len(df_panneaux)):
        dist_p = haversine_distance(df_panneaux.loc[i-1,'LATITUDE'],
df_panneaux.loc[i-1,'LONGITUDE'],
                                    df_panneaux.loc[i,'LATITUDE'], df_panneaux.loc[i,'LONGITUDE'])
        distances_panneaux.append(dist_p)
    df_panneaux = df_panneaux.copy()
    df_panneaux['dist_cum'] = np.cumsum(distances_panneaux)

    # Interpoler vitesse limite sur distance cumulée du tronçon
    positions = df_points['dist_cum'].values
    v_lim_interp = np.interp(positions, df_panneaux['dist_cum'],
df_panneaux['VITESSE_KMH'].values / 3.6)

    # Calcul profil accélération
    v_acc = np.zeros_like(positions)
    v_acc[0] = v_lim_interp[0]
    for i in range(1, len(positions)):
        v_acc[i] = min(np.sqrt(v_acc[i-1]**2 + 2 * acc * (positions[i] - positions[i-1])), v_lim_interp[i])

    # Calcul profil décélération (rétropropagation)
    v_dec = np.zeros_like(positions)
    v_dec[-1] = v_lim_interp[-1]
    for i in range(len(positions)-2, -1, -1):
        v_dec[i] = min(np.sqrt(v_dec[i+1]**2 + 2 * abs(dec) * (positions[i+1] - positions[i])),
v_lim_interp[i])

    # Profil final vitesse théorique
    v_theoretical = np.minimum(v_acc, v_dec) * 3.6  # en km/h

    df_points['VITESSES'] = v_theoretical
    return df_points
```

```
# 1) Charger les données {NOM_TABLE} dans un DataFrame pandas
df_raw =
session.table('{NOM_TABLE}').select('LATITUDE','LONGITUDE','PK','L1','L2','L3').to_pandas()

# 2) Charger les panneaux vitesse AO1
df_panneaux =
session.table('{AO1_TABLE}').select('LATITUDE','LONGITUDE','VOIE','VITESSE_KMH').to_pandas()

# 3) Pour chaque voie présente dans df_raw, on fait le calcul
resultats = []
for voie_col in ['L1','L2','L3']:
   df_troncon = df_raw[df_raw[voie_col].notnull()].copy()
   for voie_val in df_troncon[voie_col].unique():
      df_segment = df_troncon[df_troncon[voie_col] ==
voie_val].sort_values('PK').reset_index(drop=True)
      df_pan = df_panneaux[(df_panneaux['VOIE'] == voie_val)].sort_values('LATITUDE')  # on trie
les panneaux pour cohérence

      if len(df_pan) < 2:
         continue  # impossible d'interpoler si pas au moins 2 panneaux

      # Calculer profil vitesse
      df_calcul = calcul_profil_vitesse_geo(df_segment, df_pan, acc=0.5, dec=-0.5, step=1)
      resultats.append(df_calcul[['LATITUDE','LONGITUDE','PK','VITESSES']])

# 4) Concaténer tout et pousser dans Snowflake
df_result = pd.concat(resultats)
df_result = df_result.drop_duplicates(subset=['LATITUDE','LONGITUDE'])

# 5) Création de la colonne VITESSES si elle n'existe pas déjà
session.sql("""
ALTER TABLE NOM_TABLE ADD COLUMN IF NOT EXISTS VITESSES FLOAT
""").collect()

# 6) Mise à jour dans Snowflake
session.write_pandas(df_result, 'TMP_VITESSES', auto_create_table=True, overwrite=True)
session.sql("""
MERGE INTO {NOM_TABLE} AS target
USING TMP_VITESSES AS source
ON target.LATITUDE = source.LATITUDE AND target.LONGITUDE = source.LONGITUDE
WHEN MATCHED THEN UPDATE SET target.VITESSES = source.VITESSES
""").collect()
```

2.3.7. RAYONS_COURBURE

Le rayon de courbure constitue un paramètre central dans l'évaluation de l'usure des voies de tramway, les zones en courbe étant généralement les plus exposées au vieillissement prématuré des rails. Il est donc essentiel de traiter cette information avec rigueur dès les premières étapes du projet. Cette donnée peut être obtenue soit par acquisition directe auprès du prestataire, via des appareils de mesure spécialisés, soit par traitement manuel, lorsque les données brutes de géolocalisation sont disponibles. La première méthode sera cependant à privilégier.

VERSION PRESTATAIRE :

Pour chaque point mesuré, le prestataire devra fournir la valeur du rayon de courbure, généralement calculée automatiquement par les capteurs embarqués. À défaut, et comme mentionné précédemment, ce calcul pourra être réalisé manuellement à partir des coordonnées géographiques transmises.

EXEMPLE FILE :

LATITUDE	LONGITUDE	PK	FILE	VITESSES	RAYON
48.85890	2.29365	0	D	12	10
48.85892	2.29366	0	G	12	10.5
48.85893	2.29367	0.1	D	13	11
48.85894	2.29368	0.1	G	13	11.4

VERSION MANUELLE :

Le cas échéant, le rayon de courbure pourra être déterminé à partir de trois points des fichiers, selon, le principe mathématique suivant ; soit trois points $A(x_1, y_1)$, $B(x_2, y_2)$, $C(x_3, y_3)$ qui définissent un cercle, alors le rayon de ce cercle est une estimation du rayon de courbure à la position du point B, via la formule $R = \frac{abc}{4A}$:

$$\text{avec } A = \frac{1}{2}|(x_1(y_2 - y_3)) + (x_2(y_3 - y_1)) + (x_3(y_1 - y_2))|$$

et a la distance entre B et C
b la distance entre A et C
c la distance entre B et A

Ce calcul en version manuelle se fera via la REQUÊTE PYTHON nommée **RAYON**
→ *On présuppose ici que les coordonnées sont projetées en système métrique.*
Les LATITUDES et LONGITUDES doivent donc être converties en amont.

RAYONS

```
# 1) Lecture depuis Snowflake
df = session.table("{NOM_TABLE}").to_pandas()

# 2) Conversion WGS84 -> Lambert 93
transformer = Transformer.from_crs("EPSG:4326", "EPSG:2154", always_xy=True)
df['X'], df['Y'] = transformer.transform(df['LONGITUDE'].values, df['LATITUDE'].values)

# 3) Fonction pour calculer le rayon à partir de 3 points
def calc_radius(x1, y1, x2, y2, x3, y3):
    try:
        # Formule géométrique du rayon du cercle passant par 3 points
        A = np.array([[x1, y1], [x2, y2], [x3, y3]])
        a = np.linalg.norm(A[1] - A[2])
        b = np.linalg.norm(A[0] - A[2])
        c = np.linalg.norm(A[0] - A[1])
        s = (a + b + c) / 2
        area = np.sqrt(s * (s - a) * (s - b) * (s - c))
        radius = (a * b * c) / (4 * area)
        return radius
    except:
        return np.nan

# 4) Calcul glissant du rayon
radii = [np.nan]  # Premier point n'a pas de rayon
for i in range(1, len(df)-1):
    x1, y1 = df.loc[i-1, ['X', 'Y']]
    x2, y2 = df.loc[i, ['X', 'Y']]
    x3, y3 = df.loc[i+1, ['X', 'Y']]
    radius = calc_radius(x1, y1, x2, y2, x3, y3)
    radii.append(radius)
radii.append(np.nan)  # Dernier point n'a pas de rayon
df['rayon'] = radii

# 5) Sauvegarde dans une table temporaire sur Snowflake
session.write_pandas(df[['LATITUDE',  'LONGITUDE',  'PK',  'FILE',  'ID',  'rayon_m']],  'TMP_RAYON',
auto_create_table=True, overwrite=True)

# 6) Création de la colonne RAYON_COURBURE si elle n'existe pas déjà
session.sql("""
ALTER TABLE NOM_TABLE ADD COLUMN IF NOT EXISTS RAYON_COURBURE FLOAT
""").collect()
```

```
# 7) Mise à jour dans la table principale avec MERGE
session.sql("""
MERGE INTO {NOM_TABLE} AS target
USING TMP_RAYON AS source
ON target.LATITUDE = source.LATITUDE
  AND target.LONGITUDE = source.LONGITUDE
  AND target.PK = source.PK
  AND target.FILE = source.FILE
  AND target.ID = source.ID
WHEN MATCHED THEN UPDATE SET target.RAYON_COURBURE = source.rayon
""").collect()

print("Colonne RAYON_COURBURE calculée et mise à jour dans NOM_TABLE")
```

2.3.8. DEVERS

Le dévers correspond à la différence de hauteur entre les deux files de rails, mesurée perpendiculairement à la voie. Il joue un rôle crucial dans le confort de roulement, la répartition des charges et la dynamique des véhicules en courbe. Une mauvaise gestion du dévers peut accentuer l'usure des rails, en particulier sur les courbes, et entraîner des sollicitations mécaniques asymétriques.

À ce titre, il constitue également un paramètre clé à intégrer dès les premières étapes du projet de modélisation. Il peut être obtenu :

- soit par acquisition directe à l'aide d'appareils de mesure embarqués capables d'enregistrer la géométrie de la voie avec une haute précision (capteurs inertiels, inclinomètres, etc.) ;
- soit par traitement manuel, à l'aide de la vitesse et du rayon de courbure

Dans notre cas, il est largement préférable que le dévers soit transmis par le prestataire technique. C'est donc cette méthode que l'on privilégiera.

Le cas échéant, le dévers peut être obtenu grâce à la formule : $D = 11.8 * \frac{V^2}{R}$

> avec V^2 la vitesse au carré
> et R le rayon de courbure

Cette formule permet de calculer un dévers idéal, c'est-à-dire celui qui compenserait parfaitement la force centrifuge ressentie dans une courbe. Grâce à ce dévers théorique, on obtiendrait un train ne dérivant pas vers l'extérieur ; des passagers ne subissant aucune poussée latérale, ; et un véhicule en équilibre dynamique sur les rails. Dans notre situation, elle est l'obtention de ce dévers peut-être rendue possible en effectuant la REQUÊTE SQL **DEVERS.**

Cependant, en réalité, cette valeur théorique ne reflète pas les conditions d'exploitation. En effet, le dévers réellement appliqué est souvent réduit, pour s'adapter à la diversité des vitesses de circulation, aux contraintes techniques ou à des exigences de sécurité. C'est pourquoi il reste préférable de s'appuyer sur des données de terrain plutôt que sur cette estimation, qui ne constitue qu'une approximation dans notre modèle.

EXEMPLE FILE :

LATITUDE	LONGITUDE	PK	VITESSES	RAYON	DEVERS
48.85890	2.29365	0	12	10	50
48.85892	2.29366	0	12	10.5	50
48.85893	2.29367	0.1	13	11	55
48.85894	2.29368	0.1	13	11.4	55

DEVERS	
UPDATE {NOM_TABLE} SET DEVERS = 11.8 * (POWER(VITESSE, 2) / RAYON_COURBURE) WHERE VITESSES IS NOT NULL AND RAYON_COURBURE IS NOT NULL;	> VITESSE et RAYON_COURBURE sont des colonnes déjà existantes

2.3.9. PENTE

La pente longitudinale de la voie correspond à la variation d'altitude du rail le long de son tracé.
Ce paramètre influence directement les efforts appliqués sur la voie, en particulier en montée ou en descente, où les contraintes de traction, de freinage et d'adhérence sont modifiées.

Dans le cadre de la modélisation de l'usure des rails, la pente constitue un facteur contributif indirect, notamment sur les sections fortement inclinées ou soumises à des arrêts fréquents (zones de station, terminus, rampes d'accès). Elle peut accélérer certaines formes d'usure, notamment dans les zones à forte sollicitation ou combinées avec des courbes.

Ce paramètre peut être intégré au modèle :

- soit par acquisition directe auprès du prestataire technique, ce que l'on recommande toujours, car permet d'avoir une précision de haut niveau ;
- soit par traitement manuel, à partir de la variation des altitudes Z relevées point par point en calculant le taux de pente entre deux positions successives via la formule :

$P_x = \frac{Z_{x+1} - Z_x}{D} * 1000$ avec D la distance entre les points x et $x + 1$ données en entrée.
Le notion de $* 1000$ est utilisée ici car la pente est souvent en pourmillages.

Dans les faits, ce calcul se traduit par une REQUÊTE PYTHON nommée **PENTE**
Elle donne vie à des données telles que les suivantes :

EXEMPLE FILE :

LATITUDE	LONGITUDE	PK	VITESSES	RAYON	DEVERS	PENTE
48.85890	2.29365	0	12	10	50	5
48.85892	2.29366	0	12	10.5	50	5
48.85893	2.29367	0.1	13	11	55	12
48.85894	2.29368	0.1	13	11.4	55	12

PENTE

```
def calcul_pente_multivoies(df, pk_col='PK', z_col='ALTITUDE', voie_cols=['L1', 'L2', 'L3', 'L4']):
    # Trouver la première voie non nulle par ligne
    def premiere_voie(row):
```

```
        for col in voie_cols:
            if pd.notnull(row[col]):
                return row[col]
        return np.nan

    df = df.copy()
    df['voie_unique'] = df.apply(premiere_voie, axis=1)

    # Trier par voie_unique puis PK pour cohérence du calcul
    df = df.sort_values(['voie_unique', pk_col]).reset_index(drop=True)

    pentes = np.full(len(df), np.nan)  # initialise un array NaN

    # Calcul des pentes pour chaque voie unique
    for voie in df['voie_unique'].dropna().unique():
        mask = df['voie_unique'] == voie
        df_voie = df.loc[mask].copy()

        delta_z = df_voie[z_col].shift(-1) - df_voie[z_col]
        delta_pk = df_voie[pk_col].shift(-1) - df_voie[pk_col]

        pente = (delta_z / delta_pk) * 1000  # pente en ‰
        pente.iloc[-1] = np.nan  # dernière valeur sans pente

        pentes[mask] = pente.values

    df['PENTE'] = pentes
    df.drop(columns=['voie_unique'], inplace=True)
    return df

def calcul_pente_simple(df, pk_col='PK', z_col='ALTITUDE', voie_col='L1'):
    # Filtrer points appartenant à la voie principale
    df_voie = df.loc[df[voie_col].notnull(), [pk_col, z_col]].copy()
    df_voie = df_voie.sort_values(pk_col).reset_index()

    # Calcul des différences
    delta_z = df_voie[z_col].shift(-1) - df_voie[z_col]
    delta_pk = df_voie[pk_col].shift(-1) - df_voie[pk_col]

    pente = (delta_z / delta_pk) * 1000  # pente en ‰
    pente.iloc[-1] = np.nan  # pas de pente au dernier point

    # Initialiser colonne pente avec NaN
    df['PENTE'] = np.nan
```

```
    # Remplir la colonne PENTE dans df aux bons indices
    df.loc[df_voie['index'], 'PENTE'] = pente.values
    return df

# 1) Charger la table depuis Snowflake
df = session.table('{NOM_TABLE}').select('PK', 'ALTITUDE', 'L1').to_pandas()

# 2) Calculer la pente simple (sur la voie L1)
df = calcul_pente_simple(df, pk_col='PK', z_col='ALTITUDE', voie_col='L1')

# 3) Écrire dans une table temporaire
session.write_pandas(df[['PK', 'PENTE']], 'TMP_PENTE', auto_create_table=True, overwrite=True)

# 4) Mettre à jour la table Snowflake
session.sql(f"""
UPDATE {NOM_TABLE} AS target
SET PENTE = source.PENTE
FROM TMP_PENTE AS source
WHERE target.PK = source.PK
""").collect()

# 5) Supprimer la table temporaire
session.sql("DROP TABLE IF EXISTS TMP_PENTE").collect()
```

A la fin de cette étape, une seconde, d'épuration cette fois-ci, sera entreprise. Il s'agit d'une REQUÊTE SQL, appelée **EPURATION**, qui a pour objectif de nettoyer les données en supprimant toutes les lignes où au moins une des valeurs est manquante, à l'exception des colonnes relatives aux lignes (L1, L2 …)

EPURATION	
DELETE FROM {NOM_TABLE} WHERE PK IS NULL OR LATITUDE IS NULL OR LONGITUDE IS NULL OR PENTE IS NULL OR RAYON_COURBURE IS NULL OR DEVERS IS NULL;	> PK, LATITUDE, LONGITUDE, PENTE, RAYON _COURBURE et DEVERS sont des colonnes déjà existantes

2.3.10. DISTANCE_ENTRE_BAVETTES

La distance entre bavettes correspond à l'écart longitudinal séparant deux bavettes successives installées le long de la voie. Les bavettes, placées en bas du rail ou sur l'infrastructure, jouent un rôle de protection contre l'intrusion de corps étrangers, la projection de ballast ou encore l'écoulement d'eau et de débris. Le respect d'un espacement adapté contribue à préserver l'intégrité de la voie et à maintenir de bonnes conditions de sécurité et de confort.

Dans le cadre du suivi de l'usure des rails, la distance entre bavettes est un paramètre utile pour interpréter certains phénomènes locaux, comme l'accumulation de particules abrasives ou les zones d'humidité favorisant la corrosion. Un espacement inadapté peut accentuer ces effets et influencer indirectement la vitesse de dégradation de la voie.

Cette distance peut être déterminée uniquement à partir des relevés fournis par le prestataire technique lors des inspections. Dans les faits, la donnée sera transmise sous le format suivant :

EXEMPLE FILE :

LATITUDE	LONGITUDE	PK	VITESSES	RAYON	DEVERS	PENTE	DISTANCE_ENTRE_BAVETTES
48.85890	2.29365	0	12	10	50	5	0.06
48.85892	2.29366	0	12	10.5	50	5	0.06
48.85893	2.29367	0.1	13	11	55	12	0.06
48.85894	2.29368	0.1	13	11.4	55	12	0.06

2.3.11. DENOMINATION_ZONE

Pour poursuivre le traitement, une fois les identifiants (ID) extraits, il est nécessaire d'attribuer une dénomination claire aux zones de courbes que notre modèle devra analyser. À cet effet, nous mettrons en place une REQUETE PYTHON appelée **DENOMINATION_ZONE**, qui affectera à la colonne du même nom la désignation correspondant à chaque zone.

Cette fonction différenciera les zones selon que le RAYON_COURBURE soit supérieur ou inférieur à 300 m :

- Si le rayon est supérieur à 300, la zone sera qualifiée de LIGNE_DROITE_{X},
- Sinon, elle sera identifiée comme une COURBE_{X},

où X représente un compteur numérique permettant de distinguer chaque zone de façon unique.

Cette approche garantit une précision suffisante pour que les opérations de maintenance soient conduites de manière spatialement cohérente, tout en respectant la logique métier sous-jacente.

DENOMINATION_ZONE

```
def filtre_incoherences(df, diff_max=100):
    indices_a_supprimer = []
    for i in range(1, len(df)-1):
        r = df.loc[i, 'RAYON_COURBURE']
        r_prev = df.loc[i-1, 'RAYON_COURBURE']
        r_next = df.loc[i+1, 'RAYON_COURBURE']
        if (abs(r - r_prev) > diff_max) and (abs(r - r_next) > diff_max):
            indices_a_supprimer.append(i)
    df = df.drop(indices_a_supprimer).reset_index(drop=True)
    return df

def detect_courbe(df, seuil=300):
    df = df.sort_values('PK').reset_index(drop=True)
    denom = []
    etat_courbe = None
    compteur_courbe = 0
    compteur_ligne = 0
    zone_en_cours = None
```

```
    for i in range(len(df)):
        if i < 2:
            etat_courbe = 'LIGNE_DROITE'
            if zone_en_cours != etat_courbe:
                compteur_ligne += 1
                zone_en_cours = etat_courbe
            denom.append(f'LIGNE_DROITE_ZLD_{compteur_ligne}')
            continue
        r1 = df.loc[i-2, 'RAYON_COURBURE']
        r2 = df.loc[i-1, 'RAYON_COURBURE']
        r3 = df.loc[i, 'RAYON_COURBURE']
        if (r1 < seuil) and (r2 < seuil) and (r3 < seuil):
            etat_courbe = 'COURBE'
        else:
            if etat_courbe == 'COURBE' and (r3 >= seuil):
                etat_courbe = 'LIGNE_DROITE'
            elif etat_courbe is None:
                etat_courbe = 'LIGNE_DROITE'
        if zone_en_cours != etat_courbe:
            if etat_courbe == 'COURBE':
                compteur_courbe += 1
            else:
                compteur_ligne += 1
            zone_en_cours = etat_courbe
        if etat_courbe == 'COURBE':
            denom.append(f'COURBE_ZC_{compteur_courbe}')
        else:
            denom.append(f'LIGNE_DROITE_ZLD_{compteur_ligne}')
    df['DENOMINATION_ZONE'] = denom
    return df

# 1) Charger les données depuis Snowflake
df = session.table('{NOM_TABLE}').select('PK', 'RAYON_COURBURE').to_pandas()

# 2) Filtrer les incohérences
df_filtre = filtre_incoherences(df, diff_max=100)
```

```
# 3) Détecter les courbes
df_final = detect_courbe(df_filtre, seuil=300)

# 4) Ecrire les colonnes à mettre à jour dans une table temporaire
# Ici on veut mettre à jour la colonne DENOMINATION_ZONE dans la table
df_update = df_final[['PK', 'DENOMINATION_ZONE']]

session.write_pandas(df_update, 'TMP_DENOMINATION_ZONE', auto_create_table=True,
overwrite=True)

# 5) Mettre à jour la table principale
session.sql(f"""
UPDATE {NOM_TABLE} AS target
SET DENOMINATION_ZONE = source.DENOMINATION_ZONE
FROM TMP_DENOMINATION_ZONE AS source
WHERE target.PK = source.PK
""").collect()

# 6) Supprimer la table temporaire
session.sql("DROP TABLE IF EXISTS TMP_DENOMINATION_ZONE").collect()
```

2.3.12. TYPE_RAILS

Dans la majorité des cas, les prestataires en charge des mesures sur les réseaux ferroviaires urbains précisent au minimum le type de rail observé (vignole ou à gorge) pour chaque point mesuré. Cette méthode sera donc privilégiée ici, ainsi que pour l'étape 2.11.

Lorsque cette information n'est pas fournie, il conviendra de solliciter l'Autorité Organisatrice (AO) afin d'obtenir une cartographie détaillée des types de rails posés selon les portions du réseau.

Pour cela, les données suivantes devront impérativement être transmises :

- Les coordonnées géographiques des points (latitude et longitude en WGS 84), avec la précision indiquée précédemment (nombre de décimales selon le pas spatial) ;
- Le nom de la ligne ;
- Et la voie concernée ;
- Ainsi que, bien évidemment, le type de rail.

Ces informations permettront, grâce à la table de correspondance AO2 définie en début de projet, d'effectuer une jointure géographique pour enrichir notre base avec les types de rails présents à chaque localisation.

→ *Par exemple, si l'AO nous communique qu'entre deux couples type (LATITUDE, LONGITUDE) fournis sur la V1 de la L1, on a un type de rail B, et sinon exclusivement du A, alors on pourra construire un CSV tel que suit :*

EXEMPLE DE CSV À CONSTRUIRE :

LATITUDE	LONGITUDE	TYPE DE RAILS	L1
A FOURNIR		Type A	V1
		Type B	V1
		Type B	V1
		Type A	V1

→ Puis ensuite, à l'aide d'une REQUETE PYTHON nommée **JOIN_TYPE_RAILS**, l'information pour chaque point pourra être sauvegardée dans notre DatFramr.

En outre, pour obtenir une information plus précise, notamment la désignation normalisée complète du rail (référence standard, profil exact, etc.), il est nécessaire de consulter l'autorité organisatrice du réseau. En pratique, cette information est généralement disponible dans le système d'information géographique (SIG) dédié à la cartographie des rails. Ce document, souvent préparé en amont, recense les différents types et références de rails présents sur le réseau. Le cas échéant, il faudra

procéder de manière identique à ce décrit juste au dessus. → Cette approche garantit une description fine et conforme aux standards des rails utilisés sur le réseau, essentielle pour toute analyse technique ou maintenance ciblée.

Grâce à ces techniques, on aboutira à une BDD similaire à l'exemple suivant :

EXEMPLE FILE :

ID	LATITUDE	LONGITUDE	PK	FILE	…	TYPE_RAILS
1	48.85890	2.29365	0	D		A
2	48.85892	2.29366	0	G		A
3	48.85893	2.29367	0.1	D		B
4	48.85894	2.29368	0.1	G		B

JOIN_TYPE_RAILS

```
# 1) Charger les données depuis Snowflake
# Ici on suppose que la table principale contient au moins PK, LATITUDE, LONGITUDE, L1, L2, ...
df_principal = session.table('{NOM_TABLE}') \
   .select('PK', 'LATITUDE', 'LONGITUDE', 'L1', 'L2', 'L3', 'L4') \
   .to_pandas()

# Charger aussi la table AO avec les colonnes nécessaires (LIGNE, VOIE, TYPE_DE_RAIL, LATITUDE, LONGITUDE)
df_AO = session.table('{AO2_TABLE}') \
   .select('LIGNE', 'VOIE', 'TYPE_RAILS', 'LATITUDE', 'LONGITUDE') \
   .to_pandas()

# 2) Filtrer la table AO sur la ligne et voie ciblées
ligne = "L1"
voie = "V1"
df_AO_filtre = df_AO[(df_AO["LIGNE"] == ligne) & (df_AO["VOIE"] == voie)]

# 3) Trier les points AO
df_AO_filtre = df_AO_filtre.sort_values(by=["LATITUDE", "LONGITUDE"]).reset_index(drop=True)

# 4) Construire les segments et associer les types de rails
segments = []
types_rails = []
for i in range(len(df_AO_filtre) - 1):
```

```
    p1 = Point(df_AO_filtre.loc[i, ["LONGITUDE", "LATITUDE"]])
    p2 = Point(df_AO_filtre.loc[i+1, ["LONGITUDE", "LATITUDE"]])
    line = LineString([p1, p2])
    segments.append(line)
    types_rails.append(df_AO_filtre.loc[i, "TYPE_RAILS"])

# 5) Fonction pour trouver le type de rail à partir d'un point
def trouver_type_rail(point):
    for segment, t_rail in zip(segments, types_rails):
        dist = segment.distance(point)
        if dist < 0.0005:  # distance seuil (~50m)
            return t_rail
    return None

# 6) Appliquer la fonction sur chaque point du df principal
voie_col = ligne
def get_type_rail_ligne_voie(row):
    if pd.isna(row[voie_col]):
        return None
    if row[voie_col] != voie:
        return None
    pt = Point(row["LONGITUDE"], row["LATITUDE"])
    return trouver_type_rail(pt)

df_principal["TYPE_DE_RAIL"] = df_principal.apply(get_type_rail_ligne_voie, axis=1)

# 7) Ecrire les données mises à jour dans une table temporaire ---
df_update = df_principal[['PK', 'TYPE_RAILS']].dropna(subset=['TYPE_RAILS'])
session.write_pandas(df_update, 'TMP_TYPE_RAILS', auto_create_table=True, overwrite=True)

# 8) Mettre à jour la table principale dans Snowflake
session.sql(f"""
UPDATE {NOM_TABLE} AS target
SET TYPE_RAILS = source.TYPE_RAILS
FROM TMP_TYPE_RAILS AS source
WHERE target.PK = source.PK
""").collect()

# 9) Supprimer la table temporaire
session.sql("DROP TABLE IF EXISTS TMP_TYPE_RAILS").collect()
```

2.3.13. TYPE_POSES

Pour le type de pose, une méthode en tout point similaire devra être menée, aboutissant donc grâce à un document AO5 et via la REQUETE **JOIN_TYPE_POSES,** à une base de données ressemblant à la suivante :

EXEMPLE FILE :

ID	LATITUDE	LONGITUDE	PK	FILE	…	TYPE_RAILS	TYPE_POSES
1	48.85890	2.29365	0	D		A	A
2	48.85892	2.29366	0	G		A	B
3	48.85893	2.29367	0.1	D		B	B
4	48.85894	2.29368	0.1	G		B	A

JOIN_TYPE_POSES

```
# 1) Charger les données depuis Snowflake
# Ici on suppose que la table principale contient au moins PK, LATITUDE, LONGITUDE, L1, L2, ...
df_principal = session.table('{NOM_TABLE}') \
    .select('PK', 'LATITUDE', 'LONGITUDE', 'L1', 'L2', 'L3', 'L4') \
    .to_pandas()

# Charger aussi la table AO avec les colonnes nécessaires (LIGNE, VOIE, TYPE_DE_POSE, LATITUDE, LONGITUDE)
df_AO = session.table('{AO2_TABLE}') \
    .select('LIGNE', 'VOIE', 'TYPE_POSES', 'LATITUDE', 'LONGITUDE') \
    .to_pandas()

# 2) Filtrer la table AO sur la ligne et voie ciblées
ligne = "L1"
voie = "V1"
df_AO_filtre = df_AO[(df_AO["LIGNE"] == ligne) & (df_AO["VOIE"] == voie)]

# 3) Trier les points AO
df_AO_filtre = df_AO_filtre.sort_values(by=["LATITUDE", "LONGITUDE"]).reset_index(drop=True)

# 4) Construire les segments et associer les types de poses
```

```
segments = []
types_poses = []
for i in range(len(df_AO_filtre) - 1):
    p1 = Point(df_AO_filtre.loc[i, ["LONGITUDE", "LATITUDE"]])
    p2 = Point(df_AO_filtre.loc[i+1, ["LONGITUDE", "LATITUDE"]])
    line = LineString([p1, p2])
    segments.append(line)
    types_poses.append(df_AO_filtre.loc[i, "TYPE_POSES"])

# 5) Fonction pour trouver le type de pose à partir d'un point
def trouver_type_pose(point):
    for segment, t_pose in zip(segments, types_poses):
        dist = segment.distance(point)
        if dist < 0.0005:  # distance seuil (~50m)
            return t_pose
    return None

# 6) Appliquer la fonction sur chaque point du df principal
voie_col = ligne
def get_type_pose_ligne_voie(row):
    if pd.isna(row[voie_col]):
        return None
    if row[voie_col] != voie:
        return None
    pt = Point(row["LONGITUDE"], row["LATITUDE"])
    return trouver_type_pose(pt)

df_principal["TYPE_POSES"] = df_principal.apply(get_type_pose_ligne_voie, axis=1)

# 7) Ecrire les données mises à jour dans une table temporaire
df_update = df_principal[['PK', 'TYPE_POSES']].dropna(subset=['TYPE_POSES'])
session.write_pandas(df_update, 'TMP_TYPE_POSES', auto_create_table=True, overwrite=True)

# 8) Mettre à jour la table principale dans Snowflake
session.sql(f"""
UPDATE {NOM_TABLE} AS target
SET TYPE_POSES = source.TYPE_POSES
FROM TMP_TYPE_POSES AS source
WHERE target.PK = source.PK
""").collect()

# 9) Supprimer la table temporaire
session.sql("DROP TABLE IF EXISTS TMP_TYPE_POSES").collect()
```

2.3.14. NOMBRE_PASSAGES

Le nombre de passages constitue également un paramètre clé dans l'analyse de l'usure des rails, chaque passage générant des sollicitations mécaniques et vibratoires sur la voie. Plus la fréquence est élevée, plus l'usure s'accélère, en particulier dans les zones sensibles (courbes, rampes, zones de freinage). Pour cette raison, il a fallu construire la donnée ;

Pour ce faire, on doit utiliser le document AO4 fourni par l'autorité organisatrice, qui rassemble le nom des arrêts et leur position géographique, par ordre croissant du positionnement des arrêts le long de la ligne et dans le sens de circulation du tramway. Avec ce fichier, il nous est nécessaire d'en acquérir également un second, nommé AO3, qui indiquera le nombre de passages de rames par unité temporelle entre deux arrêts.

De cette manière, on pourra construire une autre table, du nom de **{AO4_TABLE}** qui contiendra LATITUDE, LONGITUDE, ARRET, et autant de LIGNES que concernées par le modèle (L1, L2, L3…) ;

EXEMPLE DE CSV FOURNI PAR L'AO :

LATITUDE	LONGITUDE	ARRET	LIGNE	VOIE
48.85890	2.29365	A	1	1
48.85893	2.29367	B	1	1

L'étape suivante consiste à dédoubler les lignes la REQUETE SQL **NP_ARRET,** ainsi qu'à ajouter un ID nommé ID_ARRET propre à la table. On obtient donc une BDD **{AO4_TABLE}** comme celle-ci :

EXEMPLE FILE ARRETS_{ville} :

ID_ARRET	LATITUDE	LONGITUDE	ARRET	L1
1	48.85890	2.29365	A	V1
2	48.85890	2.29365	A	V1
3	48.85893	2.29367	B	V1
4	48.85893	2.29367	B	V1

NP_ARRETS
SELECT * FROM {AO4_TABLE} UNION ALL SELECT * FROM {AO4_TABLE}

On y ajoute ensuite plusieurs colonnes NOMBRE_PASSAGES_{LXVY}, avec X le numéro de ligne et Y le numéro de voie associé et on complète “à la main” et selon le principe de la requête **NP_COMPLETER,** le nombre de passages de rames entre chaque arrêt :

NP_COLONNES et NP_COMPLETER	
ALTER TABLE {AO4_TABLE} ADD COLUMN NOMBRE_PASSAGES_{LXVY} ; UPDATE {AO4_TABLE} SET NOMBRE_PASSAGES_{LXVY} = x WHERE ARRET = '{NOM_ARRET}' AND '{LIGNE}' = '{VOIE}';	> Le nombre de passages x est à compléter en fonction du document AO3 > De même pour l’arrêt Y et l’ID de l’arrêt Z qui dépendent du tableau précédemment construit > {LIGNE} et {VOIE} sont respectivement le nom de la colonne LIGNE associée à la donnée traitée, et plus spécifiquement de la cellule ‘{VOIE} associée

ID_ARRET	LATITUDE	LONGITUDE	ARRET	L1	L2	NOMBRE_ PASSAGES_ L1V1	NOMBRE_ PASSAGES_ L2V2
2	48.85890	2.29365	A	V1		150	
3	48.85893	2.29367	B	V1		150	
4	48.85893	2.29367	B	V1		200	
5	48.85831	2.29363	C		V2		300
7	48.85832	2.29363	C		V2		300

Cette requête sera alors à appliquer autant de fois qu’il existe de combinaisons LXVY.

Dans notre BDD initiale, on copiera ensuite, via une REQUETE SQL nommée **NP_COPIE,** et après avoir créé ces mêmes colonnes NOMBRE_PASSAGES_{LXVY} ainsi que la colonne ARRET, le contenu de celles-ci. On appliquera de surcroît une autre REQUETE, en PYTHON cette fois, nommée **NP_ASSIGNATIONS_VALEURS**, nous permettant d’extrapoler la donnée sur l’ensemble des points du DataFrame initial, et donc d’obtenir un tableau similaire au précédent, mais s’étendant sur une salve de données plus importante.

NP_COPIE	
UPDATE {NOM_TABLE} a SET a.NOMBRE_PASSAGES_{LXVY} = b.NOMBRE_PASSAGES_{LXVY} FROM {AO4_TABLE} b WHERE a.LATITUDE = b.LATITUDE AND a.LONGITUDE = b.LONGITUDE AND a.{LX}=b.{LX} ;	> {NOM_TABLE} est le nom de notre BDD initiale > {LXVY} est la combinaison des LX (L1, L2, 3…) et VY (V1 ou V2) > {ville} est le nom de la ville traitée

NP_ASSIGNATIONS_VALEURS

```
# 1) Charger les tables en pandas via Snowpark
df_arrets = session.table("{AO4_TABLE}").to_pandas()
df_points = session.table("{NOM_TABLE}").to_pandas()

# 2) Trier les arrêts par ID_ARRET
df_arrets_sorted = df_arrets.sort_values(by="ID_ARRETS").reset_index(drop=True)

# 3) Trouver colonnes LX dans df_arrets
colonnes_L = [col for col in df_arrets.columns if col.startswith("L") and len(col) == 2]

# 4) Initialiser dans df_points les colonnes NOMBRE_PASSAGES_LxVy si absentes
for col_L in colonnes_L:
    voies_uniques = df_arrets[col_L].dropna().unique()
    for voie in voies_uniques:
        col_np = f"NOMBRE_PASSAGES_{col_L}{voie}"
        if col_np not in df_points.columns:
            df_points[col_np] = None

# 5) Parcourir les paires consécutives d'arrêts et propager nb_passages sur df_points
for i in range(len(df_arrets_sorted) - 1):
    arret1 = df_arrets_sorted.loc[i]
    arret2 = df_arrets_sorted.loc[i + 1]

    for col_L in colonnes_L:
        voie1 = arret1[col_L]
        voie2 = arret2[col_L]

        if pd.isna(voie1) or pd.isna(voie2) or voie1 != voie2:
            continue
```

```
    col_np = f"NOMBRE_PASSAGES_{col_L}{voie1}"
    if col_np not in df_arrets.columns:
        continue

    nb_passages = arret1[col_np]
    if pd.isna(nb_passages):
        continue

    lat_min, lat_max = sorted([arret1["LATITUDE"], arret2["LATITUDE"]])
    lon_min, lon_max = sorted([arret1["LONGITUDE"], arret2["LONGITUDE"]])

    mask = (
        (df_points["LATITUDE"] >= lat_min) & (df_points["LATITUDE"] <= lat_max) &
        (df_points["LONGITUDE"] >= lon_min) & (df_points["LONGITUDE"] <= lon_max) &
        (df_points[col_L] == voie1)
    )
    df_points.loc[mask, col_np] = nb_passages

# 6) Ecrire dans une table temporaire dans Snowflake
df_update = df_points[['PK'] + [col for col in df_points.columns if
col.startswith("NOMBRE_PASSAGES_")]]
session.write_pandas(df_update, 'TMP_NB_PASSAGES', auto_create_table=True, overwrite=True)

# 7) Mettre à jour la table POINTS_TABLE dans Snowflake avec ces nouvelles valeurs
set_clauses = ", ".join([f"{col} = source.{col}" for col in df_update.columns if col != "PK"])
session.sql(f"""
UPDATE POINTS_TABLE AS target
SET {set_clauses}
FROM TMP_NB_PASSAGES AS source
WHERE target.PK = source.PK
""").collect()

# 8) Supprimer la table temporaire
session.sql("DROP TABLE IF EXISTS TMP_NB_PASSAGES").collect()
```

L'étape suivante, la dernière, consiste à réaliser une autre REQUÊTE PYTHON, nommée **RAPPROCHEMENT_GEO,** visant à identifier les voies dites communes en rapprochant géographiquement les points appartenant à différentes lignes. L'utilisateur pourra définir lui-même le seuil de proximité considéré comme « proche » ; actuellement, cette distance est fixée à 50 centimètres. Cette requête permet aussi et surtout d'obtenir un cumul du NOMBRE_PASSAGES selon ce rapprochement géographique.

Ainsi, on passera d'une BDD de cette forme → *EXEMPLE FILE :*

LATITUDE	LONGITUDE	PK	L1	L2	NOMBRE_ PASSAGES_ L1V1	NOMBRE_ PASSAGES_ L2V2
48.85890	2.29365	0	V1		150	
48.85892	2.29366	0	V1		150	
48.85893	2.29367	1		V2		200
48.85894	2.29368	1		V2		200

A celle-ci → *EXEMPLE FILE :*

LATITUDE	LONGITUDE	PK	L1	L2	NOMBRE_ PASSAGES_ L1V1	NOMBRE_ PASSAGES_ L2V2	NOMBRE_ PASSAGES
48.85890	2.29365	0	V1	V2	150	200	350
48.85892	2.29366	0	V1	V2	150	200	350
48.85893	2.29367	1	V1	V2	150	200	350
48.85894	2.29368	1	V1	V2	150	200	350

NP_RAPPROCHEMENT_GEO :

```
# 1) Charger la table depuis Snowflake en pandas
df = session.table("{NOM_TABLE}").to_pandas()
# 2) Créer la colonne geometry (Point shapely)
df['geometry'] = df.apply(lambda r: Point(r['LONGITUDE'], r['LATITUDE']), axis=1)

# 3) Extraire coordonnées pour KDTree (numpy array)
coords = np.array([(p.x, p.y) for p in df['geometry']])

# 4) Construire le KDTree spatial
radius_deg = 0.0000005  # ≈ 0.05 m en degrés

tree = cKDTree(coords)

# 5) Repérer colonnes LX (ex: L1, L2, L3, ...)
cols_LX = [col for col in df.columns if col.startswith('L') and len(col) == 2]
```

```
# 6) Repérer colonnes NOMBRE_PASSAGES_ (ex: NOMBRE_PASSAGES_L1V1, ...)
cols_np = [col for col in df.columns if col.startswith('NOMBRE_PASSAGES_')]

# 7) Fonction pour extraire LX depuis un nom de colonne NOMBRE_PASSAGES_LxVy
def extract_LX(col_name):
    parts = col_name.split('_')
    if len(parts) >= 3:
        return parts[2][:2]
    return None

np_to_LX = {col: extract_LX(col) for col in cols_np}

# 8) Pré-calculer pour chaque point les LX remplies et somme des passages par LX
LX_per_point = []
NP_per_point = []

for idx, row in df.iterrows():
    # Ensemble des LX (L1, L2, ...) non nulles pour ce point
    lx_set = {lx for lx in cols_LX if pd.notna(row[lx]) and row[lx] != ''}
    LX_per_point.append(lx_set)

    # Dictionnaire {LX: somme des NOMBRE_PASSAGES correspondants}
    np_dict = {}
    for col_np in cols_np:
        lx = np_to_LX[col_np]
        if lx in lx_set and pd.notna(row[col_np]):
            np_dict[lx] = np_dict.get(lx, 0) + row[col_np]
    NP_per_point.append(np_dict)

# 9) Initialiser la colonne NOMBRE_PASSAGE
df['NOMBRE_PASSAGE'] = 0

# 10) Pour chaque point, sommation des passages de voisins proches avec LX disjointes
for i, (lx_i, np_i) in enumerate(zip(LX_per_point, NP_per_point)):
    voisins = tree.query_ball_point(coords[i], r=radius_deg)
    somme_passages = 0
```

```
    for v in voisins:
        if v == i:
            continue
        lx_v = LX_per_point[v]
        if lx_i.isdisjoint(lx_v) and lx_i and lx_v:
            np_v = NP_per_point[v]
            somme_passages += sum(np_i.values()) + sum(np_v.values())
    df.at[i, 'NOMBRE_PASSAGE'] = somme_passages

# 11) Remplir les valeurs manquantes ou nulles dans NOMBRE_PASSAGE par valeurs uniques dans cols_np
for i, row in df.iterrows():
    if pd.isna(row['NOMBRE_PASSAGE']) or row['NOMBRE_PASSAGE'] == 0:
        vals = [row[col] for col in cols_np if pd.notna(row[col]) and row[col] != 0]
        if len(vals) == 1:
            df.at[i, 'NOMBRE_PASSAGE'] = vals[0]
        elif len(vals) > 1:
            df.at[i, 'NOMBRE_PASSAGE'] = sum(vals)  # Ou moyenne/max selon ta préférence

# 12) Ecrire le résultat dans une table temporaire Snowflake
df_update = df[['PK', 'NOMBRE_PASSAGE']].dropna(subset=['NOMBRE_PASSAGE'])
session.write_pandas(df_update, 'TMP_NOMBRE_PASSAGE', auto_create_table=True, overwrite=True)

# 13) Mettre à jour ta table principale AO4_TABLE dans Snowflake
session.sql("""
UPDATE {NOM_TABLE} AS target
SET NOMBRE_PASSAGE = source.NOMBRE_PASSAGE
FROM TMP_NOMBRE_PASSAGE AS source
WHERE target.PK = source.PK
""").collect()

# 14) Supprimer la table temporaire
session.sql("DROP TABLE IF EXISTS TMP_NOMBRE_PASSAGE").collect()
```

2.3.15. PLUVIOMÉTRIE

La pluviométrie désigne la quantité de précipitations tombées sur une zone donnée, sur une période déterminée. Dans le contexte ferroviaire, elle peut jouer un rôle indirect mais non négligeable dans le processus d'usure des rails. En effet, la présence fréquente d'eau sur la voie, notamment sous forme de pluie ou d'humidité résiduelle, peut accélérer certains phénomènes d'usure ou de corrosion, en particulier lorsqu'ils sont combinés à d'autres facteurs comme la courbure, la pente ou le passage répété des rames.

Dans le cadre de ce travail, les données pluviométriques n'ont pas été intégrées au modèle. Il serait toutefois envisageable, dans une version ultérieure, de les croiser avec les coordonnées géographiques des points de mesure à l'aide de bases de données météorologiques ouvertes (telles que Météo-France ou Copernicus), en sélectionnant les stations les plus proches ou en interpolant les données à partir de grilles régionales. Une telle approche permettrait d'affiner encore davantage l'analyse, notamment sur les segments sensibles à l'humidité et aux variations climatiques. Une collaboration avec les métropoles, qui proposent généralement ce genre d'informations, est également possible.

2.3.16. GRAISSAGE

Le graissage des rails constitue une pratique courante dans l'entretien des voies ferrées, visant à limiter les frottements entre la roue et le rail, en particulier dans les courbes serrées. En réduisant les efforts latéraux et les phénomènes de crissement, il contribue significativement à la diminution de l'usure du flanc des rails, tout en améliorant le confort acoustique et la durabilité des composants.

Son efficacité dépend toutefois de plusieurs facteurs : régularité de l'application, qualité du produit utilisé, emplacement des dispositifs de graissage (applicateurs embarqués ou fixes), et fréquence de passage des rames. Les sections graissées présentent généralement des profils d'usure plus faibles sur le flanc, bien que la table du rail puisse parfois rester exposée à une usure verticale classique.

Dans le cadre de cette modélisation, les données de graissage n'ont pas été intégrées, en raison de leur indisponibilité. Toutefois, si elles venaient à être fournies — par exemple sous forme de localisation des points de graissage ou de planning de maintenance — il serait tout à fait possible de les intégrer dans l'analyse, en croisant leur position avec les points de mesure ou en ajoutant un indicateur binaire (graissé/non graissé) dans la base de données.

Une telle intégration permettrait d'améliorer la prédiction de l'usure latérale, notamment dans les courbes fortement sollicitées.

PARTIE 3 : LES USURES

3.1. VALEURS D'USURES DE FLANCS ET TABLES :

L'usure de la table, ou usure verticale, correspond à la perte de matière sur la surface de roulement du rail, généralement causée par le passage répété des roues et aggravée par des charges élevées ou des défauts de voie. L'usure du flanc, ou usure latérale, résulte principalement des efforts transversaux exercés en courbe, lorsque les boudins de roue frottent contre la face interne du rail.

La combinaison de ces deux phénomènes impacte directement la sécurité et le confort de circulation: une usure excessive de la table modifie le profil de roulement, tandis qu'une usure excessive du flanc diminue la tenue latérale et accélère l'apparition de fissures. Un suivi régulier de ces deux paramètres permet de planifier des interventions ciblées (rechargement ou remplacement) et d'optimiser la durée de vie du rail. C'est l'objectif final du modèle mis en place.

Certains réseaux concernés par notre modèle disposeront déjà de retours d'expérience (REX) sur ces usures, obtenus grâce à des campagnes de mesures antérieures. Dans ce cas, la partie RAILADAPT du modèle sera utilisée afin d'obtenir des simulations plus précises. En revanche, pour un réseau neuf ne disposant d'aucun REX, c'est le mode RAILSTART qui sera appliqué. Cela implique des exigences différentes pour le prestataire :

- En mode RAILSTART, aucune donnée d'usure ne pourra être fournie.
- En mode RAILADAPT, le prestataire devra transmettre les données d'usure déjà étudiées, qui viendront compléter notre base de données actuelle.

Ces données devront indiquer :
- le type d'usure (flanc ou table) ;
- une année antérieure à l'année en cours.

En clair, cela se traduit, dans notre exemple filé, sous la forme suivante, où les usures complétées correspondent au reste de hauteur de table pour TABLE et d'épaisseur de flanc pour FLANC.

EXEMPLE FILE :

LATITUDE	LONGITUDE	PK	…	TABLE_ 2021	TABLE_ 2022	FLANC_ 2021	FLANC_ 2022
48.85890	2.29365	0		5	3	6	4
48.85892	2.29366	0		5	3	6	4
48.85893	2.29367	0.1		5	2	7	4

48.85894	2.29368	0.1		5	2	7	4

3.2. DIFFERENCE RAILADAPT ET RAILSTART :

Le modèle développé repose sur un pipeline modulaire*, conçu pour s'adapter à deux cas d'usage distincts rencontrés dans la gestion des réseaux de tramway :

- RAILSTART, pour les réseaux nouveaux ou récents, sans historique d'usure ;
- RAILADAPT, pour les réseaux disposant d'un à deux ans de retour d'expérience (REX) sur les mesures d'usure.

Ces deux approches partagent une base algorithmique commune, mais diffèrent dans leur manière d'exploiter les données disponibles.

** Un pipeline est une chaîne d'étapes de traitement des données. Lorsqu'on dit qu'un pipeline est modulaire, cela signifie que chaque étape du pipeline est indépendante, interchangeable, adaptable ou réutilisable sans modifier l'ensemble du système.*

Dans les faits, il s'agit seulement de faire appel à des Notebooks différents sur Snowflake, pratique que l'on détaillera pour chacun des modes.

3.2.1. APPROCHE RAILSTART

Dans cette configuration, le modèle fonctionne en mode prédictif pur, à partir des caractéristiques techniques, géométriques et d'exploitation du réseau. Aucun ajustement par des données mesurées n'est possible à ce stade, le modèle s'appuie donc entièrement sur l'assimilation effectuée à partir du réseau source (réseau de référence).

ÉTAPES DU PIPELINE :

1. CHARGEMENT ET PREPARATION DES DONNEES : import des données tabulaires (CSV ou SQL), contrôle des formats et nettoyage initial.
2. STANDARDISATION DES VARIABLES : homogénéisation des échelles pour éviter les biais liés aux ordres de grandeur.
3. PRÉDICTION PAR GRADIENT BOOSTING : application directe du modèle pré-entraîné (HistGradientBoosting) pour estimer les évolutions d'usure.

RÉSULTATS GÉNÉRÉS :

- Évolution simulée de l'usure année par année (TABLE et FLANC) ;
- Détection automatique du franchissement du seuil d'intervention ;
- Estimation de la date de maintenance préventive par extrapolation.

Cette approche permet d'initier une planification intelligente sur un réseau sans historique, en capitalisant sur les similarités structurelles avec le réseau de référence.

Dans ce cas de figure, un Notebook, nommé RAILSTART, devra être utilisé en cliquant sur Run en haut à droite de l'interface SnowFlake.

3.2.2. APPROCHE RAILADAPT

Lorsque des données d'usure réelles sont disponibles (même limitées), le modèle peut être réajusté pour améliorer la précision des prédictions. Cela permet d'affiner la courbe d'évolution, en adaptant les dynamiques à la réalité propre du nouveau réseau.

ÉTAPES SUPPLÉMENTAIRES INTÉGRÉES :

1. Intégration des mesures réelles en tant que points de calibration.
2. Réajustement du modèle
3. Nouvelles prédictions

RÉSULTATS GÉNÉRÉS :

- Amélioration de la courbe d'usure projetée, avec recalage sur les données observées ;
- Renforcement de la robustesse des prévisions à moyen terme ;
- Ajustement dynamique des dates de maintenance prévues en fonction du comportement réel du réseau.

Cette approche permet d'exploiter intelligemment un retour d'expérience partiel pour fiabiliser les décisions de maintenance, sans nécessiter un historique complet.

Dans ce cas de figure, le Notebook à utiliser sera celui nommé RAILADAPT.

3.2.3. CONCLUSION

Cette articulation progressive entre RAILSTART et RAILADAPT permet de couvrir un large spectre de cas rencontrés en exploitation, tout en capitalisant au fil du temps sur les données collectées par l'exploitant. En résumé :

- ➔ ***RAILSTART*** est idéal pour initier une stratégie de maintenance préventive sur un réseau neuf ou récemment numérisé.
- ➔ ***RAILADAPT*** prend le relais dès lors que des mesures d'usure sont disponibles, afin d'affiner les prédictions et d'optimiser les échéances.

3.3. UTILISATION DES USURES DU MODÈLE ACTUEL :

Actuellement, c'est la fonctionnalité RAILADAPT qui a été utilisée pour mettre en place le cœur de notre modèle. En effet, nous disposions de données relatives à l'usure des FLANC et TABLE dans nos jeux de données précédemment traités, bien que leur quantité restait limitée. Afin d'augmenter ce volume sans introduire de biais significatif, nous avons appliqué une méthode de vectorisation via la cosine similarity.

La cosine similarity mesure l'angle entre deux vecteurs et, dans notre cas, ces vecteurs représentent la combinaison des paramètres pris en compte par le modèle (tels que détaillés précédemment). Concrètement, cette mesure de similarité permet, pour chaque point, d'identifier le point le plus proche de lui en termes de produit scalaire, selon la formule :

$$\textit{cosine similarity} = \frac{\vec{A} \cdot \vec{B}}{\left|\vec{A}\right| \cdot \left|\vec{B}\right|}$$

avec $\vec{A}$ et $\vec{B}$ les vecteurs associés à la combinaison de paramètres points

Cette méthode d'imputation par similarité a permis d'enrichir la base de données avec des valeurs estimées à partir des profils les plus proches, en s'appuyant sur leurs caractéristiques communes. Bien que relative, cette approche est pleinement justifiée dans un contexte de données partiellement manquantes et constitue une solution viable pour préserver la robustesse de l'apprentissage. Elle améliore ainsi l'exploitation des données issues du REX et garantit une meilleure représentativité des cas réels lors de l'entraînement du modèle.

PARTIE 4 : LE COEUR DE MODÈLE

Avant d'entamer la modélisation proprement dite, il est essentiel de mettre en place une série de fonctions préliminaires dédiées à la validation, à l'extraction et à la sélection des données pertinentes. Ces étapes garantissent la qualité et la cohérence des données utilisées en aval.

4.1. FONCTIONS PRÉLIMINAIRES DE CLEANING

Parmi ces fonctions préliminaires, certaines visent à garantir la cohérence interne des données. Par exemple, la hauteur de TABLE en année x+1 doit logiquement être supérieure à celle de l'année x, sauf en cas de maintenance déclarée entre les deux ; dans le cas contraire, l'information est considérée comme incohérente et doit être écartée. L'ensemble de ces traitements est regroupé dans le notebook CLEANING_DATA, qui doit impérativement être exécuté avant toute autre requête.

Simples en apparence, ces fonctions constituent pourtant des briques essentielles pour préparer les données à une modélisation fiable et robuste. Elles automatisent le contrôle et la correction des incohérences, sécurisant ainsi la manipulation des jeux de données et prévenant l'introduction d'erreurs, qu'il s'agisse de valeurs invalides ou d'anomalies dans la nomenclature des colonnes.

4.2. ONE-HOT ENCODING

Parmi ces étapes figure également le One-Hot Encoding, qui consiste à convertir les variables catégorielles — généralement non prises en compte directement par ce type de modèle — en un ensemble de variables binaires exploitables.

Concrètement, cette méthode crée une colonne par catégorie distincte, attribuant la valeur 1 lorsque l'observation correspond à la catégorie en question, et 0 sinon. Elle évite ainsi d'introduire un ordre artificiel entre les catégories et permet au modèle de traiter chaque modalité de façon totalement indépendante.

Essentielle à une exploitation optimale des données, cette étape est intégrée dans nos deux modules et figure dans les deux notebooks qui portent son nom.

4.3. STANDARDISATION

Ensuite, il a fallu standardiser les données. La standardisation est une étape essentielle dans la préparation des variables numériques avant leur utilisation dans un modèle. Elle consiste à centrer et réduire les données, c'est-à-dire à transformer chaque variable pour qu'elle ait une moyenne nulle et un écart-type égal à un.

La fonction associée dans notre cas, sous une REQUÊTE PYTHON du nom de **STANDARDISATION**, identifie d'abord un ensemble de colonnes numériques potentielles à standardiser, parmi lesquelles on retrouve des variables techniques comme l'âge depuis la dernière maintenance, la pente, ou encore le rayon de courbure. Seules les colonnes effectivement présentes dans le jeu de données sont prises en compte.

Ensuite, la fonction applique la transformation de standardisation via un objet ***StandardScaler***. Cela permet d'harmoniser les échelles des différentes variables, évitant ainsi qu'une variable avec de grandes valeurs n'écrase l'influence des autres lors de l'apprentissage du modèle.

Pour rappel, Le StandardScaler est un outil de transformation des données qui permet de standardiser les variables numériques en les centrant (moyenne = 0) et en les réduisant (écart-type = 1). Cette transformation facilite l'apprentissage des modèles en mettant toutes les variables sur une même échelle, ce qui évite qu'une variable domine les autres uniquement par son ordre de grandeur.

En sortie, la fonction retourne à la fois le DataFrame avec les colonnes standardisées et le scaler utilisé, ce qui est utile pour appliquer la même transformation à de nouvelles données ultérieurement. La standardisation étant nécessaire au modèle, elle s'applique dans les deux modules possibles, à savoir RAILSTART et RAILADPT, et se retrouve de fait dans les Notebooks des mêmes noms.

STANDARDISATION

```
def standard_scaler(df):
      # Colonnes numériques à standardiser
   numeric_cols = [
      "NBR_PASS_2022", "PENTE", "RAYON_DE_COURBURE", "DEVERS",
      "DISTANCE_ENTRE_BAVETTES", "LIMITATION_VITESSE",
      "USURE_PRECEDANTE_TABLE","USURE_PRECEDANTE_FLANC","ANNEE" ]

   # Colonnes de sortie standardisées
   scaled_cols = [f"{c}_SCALED" for c in numeric_cols]
   # Création du scaler
```

```
scaler = StandardScaler(
    input_cols=numeric_cols,
    output_cols=scaled_cols,
    with_mean=True,
    with_std=True  )

# Fit et transform dans Snowflake
df_scaled = scaler.fit(df).transform(df)
return df_scaled, scaled_cols
```

4.4. CRÉATION / ENTRAÎNEMENT DU MODÈLE SÉQUENTIEL

La création et l'entraînement du modèle séquentiel constituent la phase où l'algorithme apprend, à partir des données historiques, à prédire l'usure future. Cette étape débute par une préparation rigoureuse des données, assurée par plusieurs traitements dédiés.

Une fois les informations nettoyées, imputées et organisées sous forme de séquences, un découpage des données est effectué selon une proportion standard de 80 % pour l'entraînement et 20 % pour le test. Cette répartition permet d'optimiser l'apprentissage tout en conservant un échantillon indépendant pour évaluer la capacité du modèle à généraliser sur des données qu'il n'a jamais vues. Sans ce découpage, il serait impossible de distinguer une bonne performance réelle d'un simple surapprentissage, et l'algorithme risquerait de donner de faux signaux de fiabilité.

Afin de structurer l'information temporelle et de faciliter les prédictions séquentielles, des colonnes supplémentaires sont introduites :

- usure_precedente : valeur d'usure enregistrée l'année précédente pour le même point kilométrique (PK),
- annee : année correspondant à la mesure en cours,
- usure_actuelle : valeur d'usure observée sur l'année courante.

Cette structuration permet au modèle de relier directement chaque état d'usure à son contexte temporel et spatial (via le PK), garantissant ainsi une continuité logique entre les prédictions d'année en année.

Le modèle HistGradientBoostingRegressor est ensuite instancié et entraîné via la méthode .fit() sur le jeu d'apprentissage (X_train, y_train). Lorsque l'option do_retrain=True est activée, le modèle est recalculé puis sauvegardé avec joblib.dump.

Ses performances sont ensuite évaluées à l'aide d'indicateurs tels que le MAE, le RMSE et le R^2, tandis qu'une analyse d'importance par permutation permet d'identifier les variables les plus déterminantes. Cette procédure assure que le modèle final est optimisé, validé de manière fiable et prêt à être appliqué sur de nouvelles données.

4.5. APPLICATION DU MODÈLE SÉQUENTIEL

L'application du modèle séquentiel consiste à exploiter un modèle déjà entraîné pour estimer l'évolution de l'usure sur de nouvelles données.

Pour chaque tronçon, seules les valeurs d'usure pertinentes sont retenues, puis la prédiction démarre à partir de l'année initiale observée. L'usure est estimée itérativement, année après année, en tenant compte de la valeur précédente et en signalant les interventions de maintenance lorsqu'un seuil est dépassé.

Les résultats sont ensuite structurés de manière à ce que chaque ligne décrive une année simulée pour un tronçon, avec la prévision d'usure, les événements détectés et un compteur d'interventions.

Dans le traitement global, cette phase se déroule « ligne par ligne » par paquets de données, avec exécution en parallèle pour accélérer le calcul.

Enfin, la logique de gestion détermine si l'on effectue un réentraînement complet ou une simple application du modèle, cette dernière se limitant alors à la prédiction et à la mise en forme des résultats.

4.6. APPLICATION DU MODÈLE PAR ZONES

La dernière étape du modèle consiste à appliquer ses résultats non pas au niveau de chaque point individuel, mais à une échelle plus large, celle des zones. En effet, le modèle a été calibré pour prédire, année après année, l'évolution de l'usure et les besoins de maintenance au niveau de chaque point précis le long des rails. Cette granularité fine, bien que pertinente du point de vue technique et de la modélisation, ne correspond pas directement à la réalité opérationnelle sur le terrain.

Dans la pratique métier, les équipes de maintenance ne peuvent pas intervenir sur des segments aussi courts que quelques centimètres de rail. Une intervention à une telle échelle serait inefficace et peu rentable, car les opérations de maintenance ou de remplacement nécessitent des zones suffisamment larges pour justifier un déplacement et une mobilisation des ressources.

Par conséquent, il a été nécessaire d'extrapoler les résultats du modèle pour qu'ils s'appliquent à des zones plus vastes, correspondant à des segments exploitables et pertinents pour la planification des interventions. Cette adaptation permet de regrouper les prédictions point par point en unités opérationnelles cohérentes, facilitant ainsi la prise de décision pour la maintenance.

Pour ce faire, nous avons mis en place un système basé sur un seuil d'usure critique à partir duquel une maintenance est jugée nécessaire. Concrètement, ***une intervention est déclenchée dès que X% (dans notre cas – 30 % ; mais peut être changé) des points d'une même zone dépassent un seuil d'usure prédéfini***. Cette règle garantit que les équipes de maintenance interviennent uniquement lorsque la dégradation est suffisamment étendue pour justifier une action collective, évitant ainsi des interventions trop fragmentées et peu efficaces.

Par ailleurs, pour ne pas sous-estimer les risques liés à des points particulièrement dégradés, un système d'alerte signale tout dépassement ponctuel d'un seuil critique plus élevé. Ce flag d'alerte permet de détecter les cas où, même si la majorité des points n'a pas encore atteint le seuil d'intervention, certains points isolés présentent un risque important. Cela facilite l'anticipation et la priorisation de ces zones, évitant ainsi des défaillances graves et des situations d'urgence. Un fichier par année est alors rendu téléchargeable grâce au code ci-après.

Le résultat de cette approche est illustré par la REQUETE PYTHON nommée **MODELE_PAR_ZONE** suivante, qui agrège les données par zone pour identifier les années où une intervention est nécessaire à l'échelle opérationnelle.

MODELE_PAR_ZONE

```
import joblib
import pandas as pd
import numpy as np
from tqdm import tqdm
import datetime
import logging

# Configuration du logger
logging.basicConfig(level=logging.INFO)
logger = logging.getLogger(__name__)

# Paramètres globaux
SEUIL = 50
MAX_ANNEES = 50
POURCENTAGE = 30
SEUIL_USURE = 10
SEUIL_CRITIQUE = 12
ANNEE_DE_BASE = 2023

# --- Fonction principale optimisée ---
def usure_table_par_courbe(df_courbe, features_utiles, model):
    df_courbe_filtre = df_courbe.copy()
    historique = []
    intervention_en_cours = {idx: "RIEN" for idx in df_courbe_filtre.index}
    # Dictionnaire pour suivre les IDs ayant déjà dépassé le seuil critique
    seuil_critique_atteint = {idx: 0 for idx in df_courbe_filtre.index}
    for annee in range(MAX_ANNEES):
        try:
            X_test = df_courbe_filtre[features_utiles]
            usures = model.predict(X_test)
        except Exception as e:
            logger.warning(f"Erreur de prédiction à l'année {annee}: {e}")
            usures = [None] * len(df_courbe_filtre)
        df_courbe_filtre["USURE_PRECEDENTE"] = usures
        df_courbe_filtre["ANNEE"] = annee
        df_courbe_filtre["INTERVENTION"] = df_courbe_filtre.index.map(intervention_en_cours)
```

```
        # Vérifie si l'usure dépasse le seuil critique pour chaque point
        df_courbe_filtre["POINT_CRITIQUE"] = (df_courbe_filtre["USURE_PRECEDENTE"] >
SEUIL_CRITIQUE).astype(int)
        # Mise à jour du flag si le seuil a été dépassé à un moment donné
        for idx in df_courbe_filtre.index:
            if df_courbe_filtre.at[idx, "POINT_CRITIQUE"] == 1:
                seuil_critique_atteint[idx] = 1
        df_courbe_filtre["SEUIL_CRITIQUE_DEPASSE"] =
df_courbe_filtre.index.map(seuil_critique_atteint)
        usure_par_annee = df_courbe_filtre[[
            "ID_SNOWFLAKE", "RAYON_DE_COURBURE_SANS_DIRECTION", "ANNEE",
            "USURE_PRECEDENTE", "INTERVENTION", "POINT_CRITIQUE", "SEUIL_CRITIQUE_DEPASSE"
        ]].values.tolist()
        intervention = verification_si_faut_faire_une_intervention(usure_par_annee)
        if intervention is not None:
            usure_par_annee, usure_prec =
mettre_a_jour_usure_courbe_apres_intervention(intervention, usure_par_annee)
            df_courbe_filtre["USURE_PRECEDENTE"] = usure_prec
            for idx, row in zip(df_courbe_filtre.index, usure_par_annee):
                intervention_en_cours[idx] = row[4]
        historique.append(usure_par_annee)
    return historique

# --- Règle d'intervention ---
def verification_si_faut_faire_une_intervention(usure_par_annee):
    cumul = {"RECHARGEMENT_1": 0, "RECHARGEMENT_2": 0, "RECHARGEMENT_3": 0,
"REMPLACEMENT": 0}
    for row in usure_par_annee:
        if row[3] is None:
            continue
        if row[3] > SEUIL_USURE:
            if row[4] == "RIEN":
                cumul["RECHARGEMENT_1"] += 1
            elif row[4] == "RECHARGEMENT_1":
                cumul["RECHARGEMENT_2"] += 1
            elif row[4] == "RECHARGEMENT_2":
                cumul["RECHARGEMENT_3"] += 1
```

```
            elif row[4] == "RECHARGEMENT_3":
                cumul["REMPLACEMENT"] += 1
    seuil = POURCENTAGE * len(usure_par_annee) / 100
    for intervention, count in cumul.items():
        if count > seuil:
            return intervention
    return None

# --- Mise à jour après intervention ---
def mettre_a_jour_usure_courbe_apres_intervention(intervention, usure_par_annee):
    usure_prec = []
    for row in usure_par_annee:
        if intervention.startswith("RECHARGEMENT"):
            row[3] = 1.0
            row[4] = intervention
        elif intervention == "REMPLACEMENT":
            row[3] = 0.1
            row[4] = "RIEN"
        usure_prec.append(row[3])
    return usure_par_annee, usure_prec

# --- Lancement principal ---
if __name__ == "__main__":
    fichiers = [
        ("FLANC", "features_contenu_extrait_FLANC.csv"),
        ("TABLE", "features_contenu_extrait_TABLE.csv")]
    par_annee = {}
    par_courbe_et_rayon = {}  # <-- ajout du dictionnaire ici
    for suffixe, fichier_csv in tqdm(fichiers, desc=" Traitement des fichiers", position=0):
        logger.info(f"\n Fichier en cours : {fichier_csv}")
        df_permanent = pd.read_csv(fichier_csv)
        features_utiles = [
            col for col in df_permanent.columns
            if col not in ["DENOMINATION_ZONE", "ID_SNOWFLAKE", "LATITUDE", "LONGITUDE"] ]
        model = joblib.load("modele_sequentiel.pkl")
        types_rayon = {
            "RAYON_0_50": lambda x: x <= SEUIL,
```

```
        "RAYON_50+": lambda x: x > SEUIL}
    courbes_uniques = df_permanent["DENOMINATION_ZONE"].dropna().unique()
    for courbe_a_tester in tqdm(courbes_uniques, desc=f"→ Courbes ({suffixe})", leave=False,
position=1):
        for type_rayon, condition in types_rayon.items():
            logger.info(f"\n Traitement : {courbe_a_tester} - {type_rayon}")
            donnees_finales = []
            historique_interventions = {}
            df_courbe = df_permanent[
                (df_permanent["DENOMINATION_ZONE"] == courbe_a_tester) &
                (df_permanent["RAYON_DE_COURBURE_SANS_DIRECTION"].notna()) &
                (df_permanent["RAYON_DE_COURBURE_SANS_DIRECTION"].apply(condition)) ]
            if not df_courbe.empty:
                historique = usure_table_par_courbe(df_courbe, features_utiles, model)
                for annee_data in historique:
                    annee_offset = annee_data[0][2]
                    annee_reelle = ANNEE_DE_BASE + annee_offset
                    for row in annee_data:
                        point_id, rayon, _, usure, intervention, point_critique, seuil_depasse = row
                        intervention_prec = historique_interventions.get((point_id, courbe_a_tester),
"RIEN")
                        changement = 1 if intervention != intervention_prec else 0
                        historique_interventions[(point_id, courbe_a_tester)] = intervention
                        if changement == 1:
                            if annee_reelle not in par_annee:
                                par_annee[annee_reelle] = []
                            ligne_courbe = df_courbe[df_courbe["ID_SNOWFLAKE"] == point_id].iloc[0]
                            latitude = ligne_courbe["LATITUDE"]
                            longitude = ligne_courbe["LONGITUDE"]
                            par_annee[annee_reelle].append({
                                "COURBE": courbe_a_tester,
                                "ID_SNOWFLAKE": point_id,
                                "RAYON": rayon,
                                "LATITUDE": latitude,
                                "LONGITUDE": longitude,
                                "INTERVENTION": intervention,
                                "SUFFIXE": suffixe,
```

```
                        "ANNEE" : annee_reelle, })
                    cle = (courbe_a_tester, type_rayon)
                    if cle not in par_courbe_et_rayon:
                        par_courbe_et_rayon[cle] = []
                    par_courbe_et_rayon[cle].append({
                        "ANNEE": annee_reelle,
                        "ID_SNOWFLAKE": point_id,
                        "RAYON": rayon,
                        "LATITUDE": latitude,
                        "LONGITUDE": longitude,
                        "INTERVENTION": intervention,
                        "SUFFIXE": suffixe})
            else:
                logger.warning(f" Aucune donnée pour la courbe {courbe_a_tester} avec filtre
{type_rayon}")

    # --- Export final : un fichier par année ---
    for annee, lignes in sorted(par_annee.items()):
        df_export = pd.DataFrame(lignes)
        nom_fichier_par_annee = f"interventions_annuelles_{annee}.csv"
        df_export.to_csv(nom_fichier_par_annee, index=False, encoding="utf-8-sig")
        logger.info(f" Fichier annuel exporté : {nom_fichier_par_annee}")

    # --- Export complémentaire : un fichier par courbe et type de rayon ---
    for (courbe, type_rayon), lignes in sorted(par_courbe_et_rayon.items()):
        courbe_nom_fichier = courbe.replace(" ", "_").replace("/", "-")
        type_nom_fichier = type_rayon.replace(" ", "_").replace("/", "-")
        nom_fichier_courbe = f"interventions_{courbe_nom_fichier}_{type_nom_fichier}.csv"
        df_export = pd.DataFrame(lignes)
        df_export.to_csv(nom_fichier_courbe, index=False, encoding="utf-8-sig")
        logger.info(f" Fichier par courbe exporté : {nom_fichier_courbe}")

    # --- Export anticipation annuelle au 1er janvier de l'année en cours ---
    aujourdhui = datetime.datetime.now()
    annee_courante = aujourdhui.year
    # Année pour laquelle on veut anticiper les dépassements (exemple : année courante)
    annee_a_anticiper = annee_courante
```

```
    points_a_anticiper = []
    if annee_a_anticiper in par_annee:
        for ligne in par_annee[annee_a_anticiper]:
            # On garde les points avec une intervention prévue, donc seuil critique dépassé
            if ligne["INTERVENTION"] != "RIEN":
                points_a_anticiper.append(ligne)
    if points_a_anticiper:
        df_anticipation = pd.DataFrame(points_a_anticiper)
        colonnes_ordonnee = ["ANNEE", "COURBE", "ID_SNOWFLAKE", "RAYON", "LATITUDE",
"LONGITUDE", "INTERVENTION", "SUFFIXE"]
        df_anticipation = df_anticipation[colonnes_ordonnee]
        nom_fichier_anticipation = f"anticipation_usure_critique_{annee_a_anticiper}.csv"
        df_anticipation.to_csv(nom_fichier_anticipation, index=False, encoding="utf-8-sig")
        logger.info(f" Fichier anticipation usure critique généré : {nom_fichier_anticipation}")
    else:
        logger.info(f"Aucun point prévu pour dépasser le seuil critique en {annee_a_anticiper}.")
```

RÉSUMÉ

Face aux défis croissants du dérèglement climatique, de l'urbanisation rapide et de la pression sur les infrastructures, anticiper les dégradations est devenu un enjeu stratégique. Ce rapport s'inscrit dans cette dynamique en explorant la maintenance prédictive appliquée à un réseau de tramways. Il démontre comment la modélisation de l'usure des rails peut devenir un outil concret d'aide à la décision, au service de la sécurité, de la durabilité et de l'optimisation des coûts.

Fruit d'un stage de fin d'études mené au sein d'Artelia, ce travail repose sur une méthodologie rigoureuse inspirée du standard CRISP-DM. Il combine analyse métier, compréhension fine des données et développement d'un modèle prédictif fondé sur le Machine Learning. De la collecte à la préparation, en passant par la vectorisation, l'imputation et la sélection des variables, chaque étape a été pensée pour transformer des relevés bruts en projections fiables de l'usure verticale et latérale.

Au-delà de l'aspect technique, cette étude met en lumière l'importance de croiser savoir-faire terrain et outils numériques avancés. Les paramètres géométriques de la voie, la nature des rails, la fréquence de passage ou encore l'historique de maintenance deviennent autant de leviers pour bâtir un modèle capable de simuler, année après année, l'évolution de l'usure et les besoins d'intervention.

Ce rapport illustre ainsi comment l'ingénierie environnementale et l'analyse de données peuvent se rencontrer pour répondre à des enjeux concrets de mobilité durable. En proposant une vision intégrée et prospective de la maintenance, il trace la voie vers une gestion plus préventive, raisonnée et résiliente des réseaux de transport urbains.

ABSTRACT

Facing the growing challenges of climate change, rapid urbanization, and pressure on infrastructure, anticipating degradation has become a strategic priority. This report aligns with this perspective by exploring predictive maintenance applied to a tramway network. It demonstrates how modeling rail wear can become a practical decision-making tool, supporting safety, durability, and cost optimization.

The work is the result of an end-of-study internship at Artelia and is based on a rigorous methodology inspired by the CRISP-DM standard. It combines business analysis, in-depth understanding of data, and the development of a predictive model based on Machine Learning. From data collection and preparation to vectorization, imputation, and feature selection, each step was designed to transform raw measurements into reliable projections of vertical and lateral rail wear.

Beyond the technical aspect, this study highlights the importance of combining field expertise with advanced digital tools. Track geometry parameters, rail type, traffic frequency, and maintenance history all serve as levers to build a model capable of simulating, year after year, wear evolution and intervention needs.

This report thus illustrates how environmental engineering and data analysis can intersect to address concrete challenges of sustainable mobility. By proposing an integrated and forward-looking vision of maintenance, it paves the way toward more preventive, reasoned, and resilient management of urban transport networks.