Projet Rhéologie

Lisa Cortey - Xavier Watine

Novembre 2019 - Mai 2020

Commanditaire : Guillaume Maitrejean
Laboratoire Rhéologie et Procédés

Tutrice : Adeline Leclercq Samson
Laboratoire Jean Kuntzmann

Université Grenoble Alpes

Remerciements

Dans un premier temps, nous tenons à remercier notre commanditaire *Guillaume MAITRE-JEAN*, de nous avoir fait confiance pour ce projet, de nous avoir fournit les données tout au long du projet en fonction des suggestions faites en réunion et d'avoir pu répondre rapidement à toutes nos questions.

D'autre part nous remercions tout particulièrement notre tutrice *Adeline LECLERCQ SAMSON* pour ses précieux conseils et sa disponibilité tout au long du projet.

Et enfin nous remercions l'Université Grenoble Alpes - UGA, qui nous donne l'opportunité de pouvoir mettre en application nos connaissances sur des projets comme celui-ci.

Sommaire

Introduction

La rhéologie est l'étude de la déformation et de l'écoulement de la matière sous l'effet d'une contrainte appliquée. La mesure des propriétés rhéologiques notamment la mesure de la viscosité, est cruciale tant du point de vue recherche que industriel. Pour déterminer les propriétés rhéologiques complexes des fluides de nombreux appareils, appelés rhéomètres, sont couramment utilisés.

Le Laboratoire Rhéologie et Procédés (UMR 5520) est une Unité Mixte de Recherche entre le CNRS, Grenoble-Institut National Polytechnique (G-INP) et l'Université Grenoble Alpes (UGA). Il a pour principales activités la rhéologie et le génie des procédés. Le laboratoire a mis en place un projet dont l'objectif est de développer un nouveau type de rhéomètre. Pour se faire le Laboratoire de Rhéologie et Procédés collabore avec le Laboratoire Jean Kuntzmann qui a pour domaines d'activités les Mathématiques Appliquées et d'Informatique. En effet l'objectif de cette collaboration est de construire un modèle statistique afin de prédire le nombre de Reynolds d'un jet en utilisant à la fois de grands ensembles de données et une approche Data Science.

Lors de l'éjection d'un fluide, le procédé d'impression continue à jet d'encre CIJ met en compétitions trois forces : inertielle (vitesse), visqueuse (viscosité) et d'interface (tension superficielle). Ces forces affectent la morphologie du jet obtenu. Sous certaines conditions, la morphologie du jet est unique et directement liée aux propriétés rhéologiques du fluide. On observe des morphologies très différentes même pour deux fluides très proches rhéologiquement parlant.

Ainsi en éjectant un fluide à l'aide d'un dispositif CIJ adapté, et en comparant sa morphologie à une base de données contenant une vaste gamme de morphologies de jet, il est possible de déterminer le nombre de Reynolds du fluide, et donc indirectement sa viscosité. En effet le nombre de Reynolds est un nombre sans dimension liant la viscosité, la masse volumique, la vitesse et une longueur de référence : Re = ρ * U * D / nu, avec ρ la masse volumique, U la vitesse à l'entrée, D le diamètre et nu la viscosité. Comme ρ, U et D sont constants, Re ne dépends que de nu. On pourrait de ce fait remplacer la fonction du rhéomètre.

Objectif : Prédire le nombre de Reynolds d'un fluide à partir de la forme/morphologie du jet.

Le but est de développer un modèle statistique fiable au moyen de modèles linéaires et de réseaux de neurones afin de mesurer la viscosité d'un fluide de manière précise à l'instar des instruments de mesure utilisés actuellement. Ce faisant nous avons à notre disposition un dispositif expérimental qui permet de simuler un jet informatiquement en choisissant l'amplitude de stimulation et le nombre d'image au cours du temps d'écoulement du jet. L'utilisation d'une simulation numérique nous permet d'avoir un nombre plus important et plus vaste de données pour un coût et un temps expérimental réduis.

Quelle approche Data Science utilisez pour prédire le nombre de Reynolds ?

1 Données

1.1 Présentation des données

Les données qui nous ont été fournies sont les coordonnées des interfaces fluide/air pour un jet à un instant T, qui ont été simulés par ordinateur. Il s'agit donc des coordonnées des points situés à la limite du fluide. Pour chaque simulation de jet, nous avons un fichier de type Reynolds_Amplitude.gnu, suite à l'évolution du projet nous avons utilisé des fichiers de type Reynolds_Amplitude_Temps.gnu comportant les coordonnées.

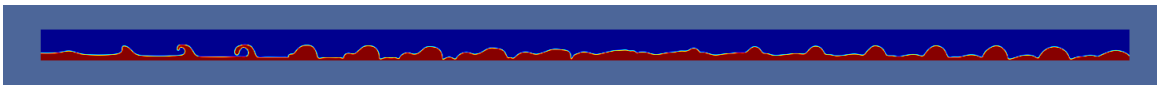


FIGURE 1 – Interface fluide/air

L'amplitude correspond à la perturbation émise sur le jet. Nous disposons d'interface pour un nombre d'amplitude allant de 0,01 à 0,15.

Le temps est en seconde la période à laquelle a été prise l'interface pour un jet.

Les coordonnées de l'interface sont les cordonnée de chaque point en x et y formant une interface, elles sont en axisymétrique centré sur l'axe des abscisses. La longueur de l'interface (en x) diffère entre 140, 200 et 240 selon les jeux de données.

Le nombre de Reynolds est un nombre sans dimension utilisé en mécanique des fluides. Il caractérise un fluide en particulier la nature de son régime (laminaire, transitoire, turbulent).Nous disposons d'interface pour un nombre de Reynolds allant de 100 à 900.

$$Re = \frac{\rho U D}{nu}$$

- ρ la masse volumique,
- nu la viscosité,
- U la vitesse du fluide,
- D le diamètre de la buse.

La vitesse du jet et le diamètre de la buse sont fixe.

1.2 Indicateurs

L'objectif est de réaliser un modèle statistique de régression capable de prédire le nombre de Reynolds d'un fluide à partir des propriétés morphologiques de celui-ci.

Pour cela nous avions deux possibilités. La première aurait été de créer les modèle à partir des coordonnées de toutes les interfaces. Et la seconde était de calculer les propriétés morphologiques de chaque interface en amont, puis de créer le modèle à partir du jeu de données obtenu.

La seconde option s'est avéré bien plus simple. En effet, la première était complexe à réalisé de part le temps de calcul nécessaire pour créer un modèle.

Nous avons donc calculé des indicateurs morphologiques des fluides à partir de ces coordonnées, afin de pouvoir réaliser des modèles de prédiction à partir des indicateurs.

Nous avons donc commencé par discuter des indicateurs qui seraient utiles et possibles à calculer avec les coordonnées.

1.2.1 Les césures/brisures

1.2.1.1 La première brisure

La première brisure est la coordonnée en abscisses où le jet se brise.

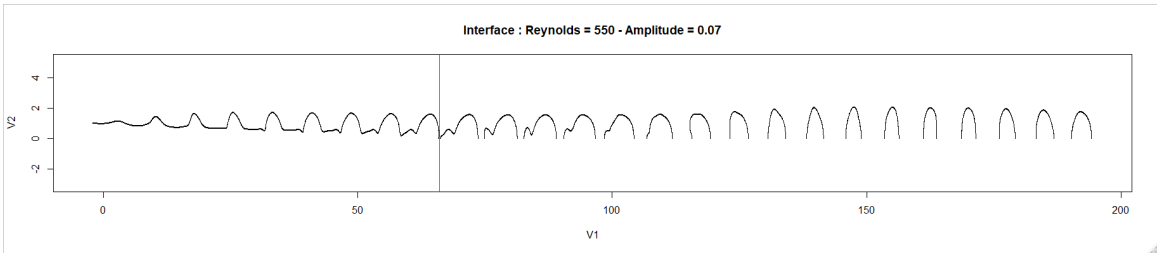


FIGURE 2 – Interface indicateur première brisure

Pour obtenir cette coordonnée, on parcourt tous les points et on regarde la distance entre le point actuel et le point suivant. La première césure sera donc la coordonnée en abscisse du point précédant la césure. La largeur de la césure a été définie à partir d'un écart supérieur à 0.3.

1.2.1.2 Le nombre de césures

On effectue la même opération pour détecter les autres brisures. Ensuite on compte le nombre de césures détectées.

Voici ce que l'on voit lorsque l'on affiche toutes les césures :

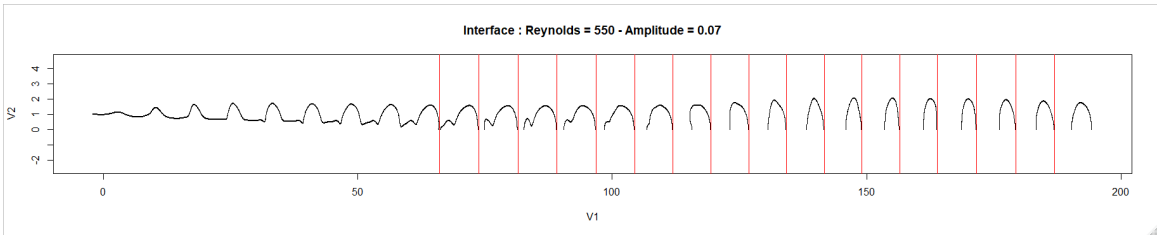


FIGURE 3 – Interface indicateur nombre de césures

Il est possible qu'une interface ne présente pas de césure. Cela est dû au fait que l'interface n'est pas assez étendue. Le jet se brise forcément. Pour gérer ce problème, si aucune césure n'est détectée, on donnera la valeur en abscisses du dernier point de l'interface.

1.2.2 Volume et surface du jet

Le volume et la surface ainsi que le ratio volume/surface sont des mesures qui vont nous donner de l'information sur les propriétés du jet.

Pour les calculer il faut se rappeler que les interfaces sur lesquelles on travaille sont en axi-symétrique, centré sur l'axe des abscisses.

1.2.2.1 Volume

Pour calculer le volume, on va utiliser l'intégration par la méthode des trapèzes.

On va donc calculer l'aire des trapèzes comme sur l'illustration ci-contre. Une fois l'aire d'un trapèze calculée, on va la multiplier par l'aire du cercle ayant pour rayon la moyenne entre les points $f(x_i)$ et $f(x_i + 1)$. Ainsi on obtiendra le volume de nos trapèzes (on perd légèrement en précision en calculant l'aire du cercle sur la moyenne des 2 points mais ça n'est pas significatif car les points sont très rapprochés). On calcule la somme de tous nos volumes de trapèzes pour obtenir le volume total du jet.

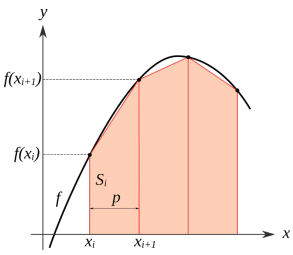


FIGURE 4 – Méthode des trapèzes

1.2.2.2 Surface

Le calcul de la surface va être très similaire à celui du volume. En effet on aura toujours besoin de calculer l'intégrale de la même manière. La surface est l'aire de la partie extérieure du jet. Pour calculer cette surface, on va "dérouler" la partie extérieure du trapèze puis calculer son aire. Ce qui nous donnera un trapèze isocèle comme suit :

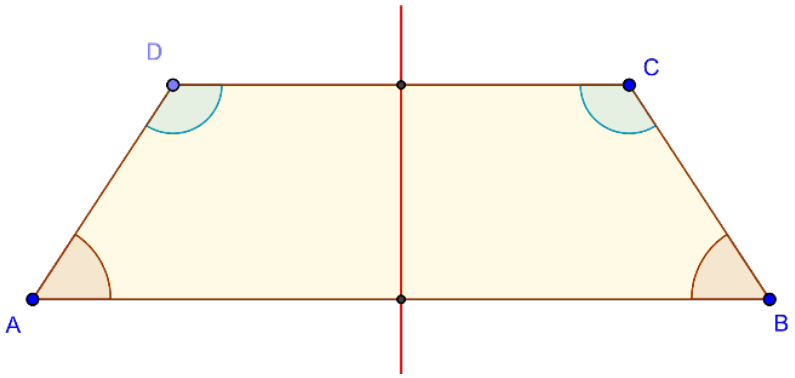


FIGURE 5 – Trapèze "déroulé"

Pour calculer l'aire de ce trapèze "déroulé", on effectuera la moyenne des aires des rectangles que l'on obtient à partir des points C et D (le petit rectangle) et des points A et B (le grand rectangle).

Les aires des rectangles seront donc égales au produit de la hauteur [1] et largeur du trapèze initial, multiplié par 2π.

1.2.3 Les gouttes

Nous avons calculé 3 indicateurs à partir du comportement du jet, que ce soit au niveau de la formation d'une goutte, c'est-à-dire avant la première brisure, ou après. Pour se faire il faut commencer par les identifier. On définira une goutte [2] par le point le plus haut de la bosse dans l'interface.

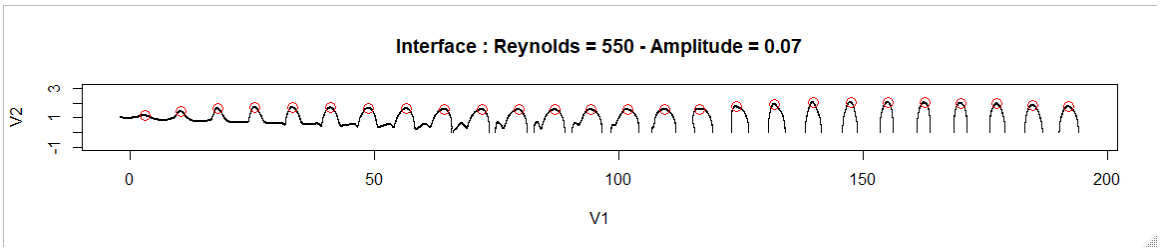


FIGURE 6 – Interface indicateur sommet gouttes

On itère sur tous les points de l'interface. On se fixe un nombre de points à analyser (on a choisi 150). Pour dire qu'un point est le sommet d'une goutte, il faudra que celui-ci soit plus haut que tous les points autour de lui. Pour nous, il faut qu'il soit plus haut que les 150 points précédents et suivants.

A partir du moment où on a identifié les gouttes, cela nous ouvre les portes pour nos indicateurs.

1.2.3.1 Les satellites

Un satellite est une petite goutte qui s'est détachée de la goutte principale.

1. Au point le plus haut pour le grand rectangle, et au point le plus bas pour le petit rectangle
2. ou future goutte

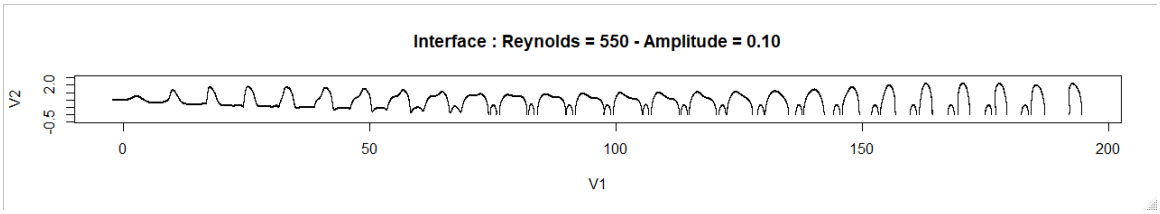


FIGURE 7 – Interface indicateur satellite

Comme on le voit sur le graphique ci-dessus, un satellite s'est créé. On peut suivre sa formation à partir de la coordonnée 50 en abscisses. Une fois formé, un satellite peut avoir 3 comportements différents. Il peut soit se faire rattraper par la goutte qui suit, soit rattraper la goutte suivante, soit rester entre les 2 gouttes. L'analyse de ces comportements est complexe, alors on se contentera de détecter la présence des satellites. Ainsi, l'indicateur sera égal à 1 s'il y a un satellite, 0 sinon.

Pour cela on va regarder si entre 2 gouttes on trouve 2 césures. En effet, normalement entre 2 gouttes (ou bosse si elle n'est pas encore formée), on peut avoir 1 césure au maximum. Si c'est plus, alors il y a un satellite.

1.2.3.2 La longueur d'onde

La longueur d'onde est la distance entre les deux gouttes qui précèdent la première brisure. Étant donné que l'on a les coordonnées de chaque goutte, cette opération ne nécessite qu'une soustraction.

1.2.3.3 Polynôme de hauteur de goutte

L'objectif ici est de trouver les coefficients d'un polynôme qui suivrait au mieux les hauteurs du fluide. Nous nous sommes arrêtés sur un polynôme de degré 4, car les résultats obtenus avec plus de coefficients n'étaient pas significativement meilleurs. Comme on peut le voir sur le graphique suivant, le résultat obtenu est plutôt satisfaisant.

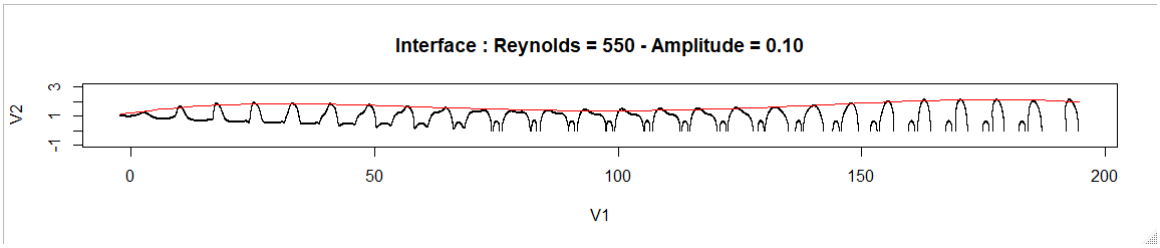


FIGURE 8 – Polynôme de degré 4 suivant les hauteur de goutte

1.3 Filtres

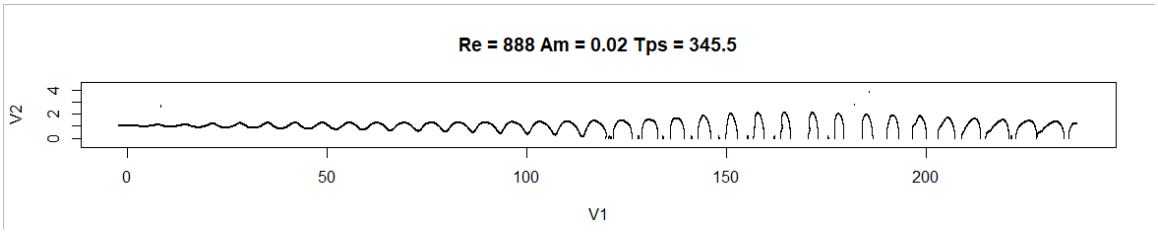
Les interfaces sont issues de simulations informatiques. On récupère les cordonnées des points entre le fluide et l'air. Malheureusement on ne peut pas éviter les erreurs informatiques. Il se trouve que certaines interfaces présentent des points qui ne sont pas censés être là.

Étant donné que les calculs des indicateurs sont effectués en parcourant tous les points, les points parasites posent un gros problème. Parfois ces points empêchent le calcul des indicateurs, car la fonction les calculant ne peut pas gérer la position trop excentrée du point. 2 000 des 43 000 interfaces ne nous donnent donc aucune information .

De plus, il se trouve qu'un nombre important d'interfaces présentant ces points parasites passent dans la fonction calculant des indicateurs.

On a donc ainsi des données erronées. C'est pour cela qu'il nous a semblé impératif de gérer le problème.

Voici un exemple d'interface avec des points parasites :

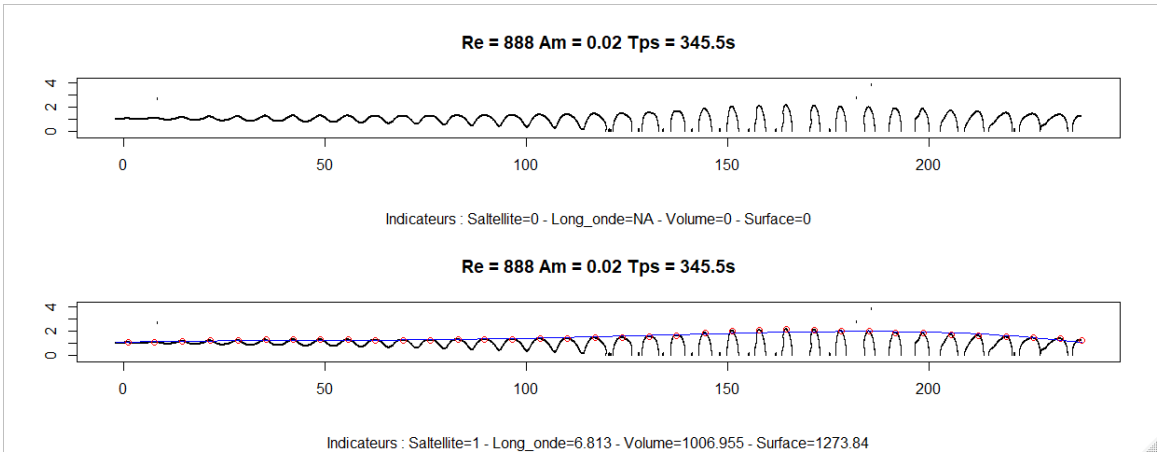


Comme on peut le voir il y a 3 points (ou amas de points) situés au-dessus du jet. Ici la fonction n'a pas calculé d'indicateurs car le point le plus haut est au-delà de la limite de hauteur de jet.

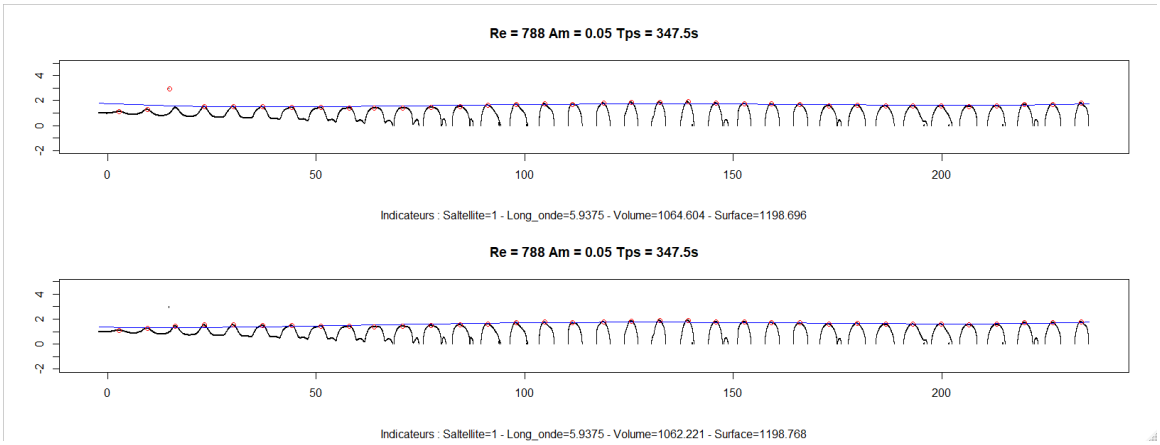
Si on supprime ces points, on pourra obtenir les indicateurs de cette interface. On va donc devoir filtrer les points. On ne peut malheureusement pas les filtrer directement dans la fonction. On va donc devoir le faire en amont, ce qui va prendre du temps.

Pour se faire, on va calculer la matrice de distances de chaque interface. Pour chaque point au-dessus du quantile 0.90 (pour limiter le temps de calcul), on va regarder les valeurs des distances avec leurs plus proches voisins. Si cette valeur est trop élevée, alors on mettra le point de côté pour l'analyse. Il se peut que certains points soient supprimés à tort. Ça n'influe pas vraiment sur les calculs d'indicateurs. En effet, les interfaces disposent d'en moyenne 10 000 points. Donc 3 points en moins sur une goutte ne changent pas significativement les indicateurs.

Voici quelques graphiques nous montrant les indicateurs calculés. On pourra voir les points les plus haut de chaque bosse/goutte, ainsi que la courbe du polynôme de degré 4 suivant ces points. D'autres indicateurs seront disposés sous le graphique. Le premier graphique nous montre les points calculés sans filtre, et le second, avec filtre. On pourra ainsi comparer les indicateurs obtenus.

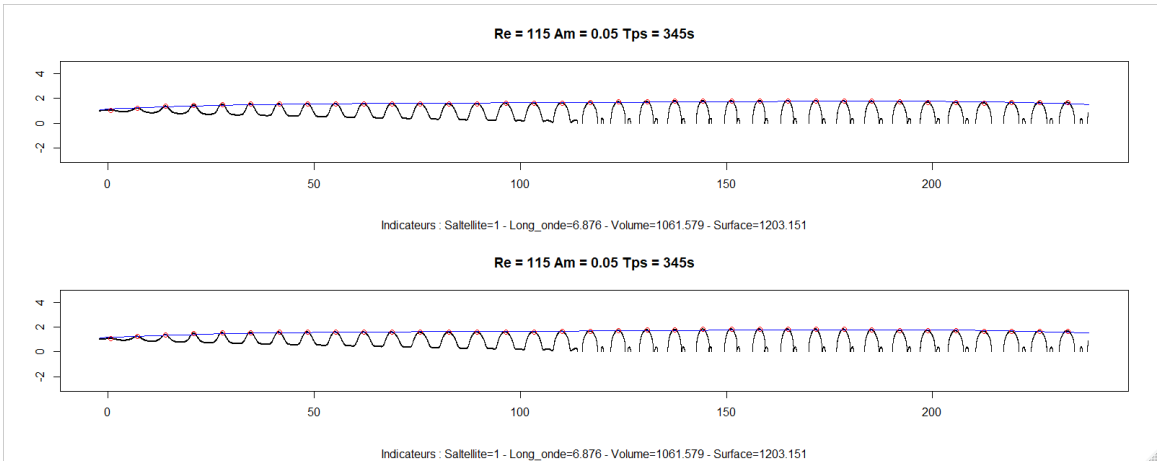


Ici on peut voir que le filtre a permis de calculer les indicateurs. En effet sans le filtre on peut voir que les indicateurs sont tous à 0 ou en valeur manquante.



Cette interface sans filtre passe dans la fonction calculant les indicateurs, malgré la présence de points parasites. Cependant on peut voir qu'elle considère le point parasite comme le point le plus haut d'une goutte. Cela fausse les indicateurs, et notamment le polynôme qui suit les points les plus haut des gouttes. Ici le point parasite a fait oublier une seule bosse. Parfois, il se peut que plusieurs bosses/gouttes soient oubliés.

Avec le filtre, aucun souci, le point parasite a bien été ignoré et la courbe suit bien mieux les gouttes au début de l'interface. On peut également voir que les indicateurs de l'interface figure 1.3 ont légèrement changé. Le volume a logiquement diminué, et le polynôme suivant les hauteurs de gouttes est bien mieux ajusté . Sur un grand nombre d'interfaces, ces amélioration sont loin d'être négligeables.



Ces deux derniers graphiques montrent une interface qui ne nécessitait aucun changement. Aucun point n'a été supprimé. Les valeurs des indicateurs restent donc les mêmes, comme voulu.

2 Méthodologie

On dispose désormais d'un ensemble de mesures de variables(indicateurs) sur la morphologie d'un jet ainsi que son nombre de Reynolds. L'objectif étant de prédire le nombre de Reynolds d'un jet inconnu, on suppose qu'il existe une relation déterministe entre ces variables et le nombre de Reynolds.

Pour répondre à cet objectif nous utiliserons deux méthodes statistiques en abordant dans un premier temps leurs fonctionnements. En effet nous allons effectuer une modélisation par régression linéaire (modèle linéaire) de manière à obtenir des informations sur notre modèle, puis nous utiliserons un réseau de neurone afin de prédire de manière la plus précise le nombre de Reynolds.

2.1 Modèle linéaire

2.1.1 Qu'est ce qu'un modèle linéaire

Un modèle linéaire est un modèle statistique avec lequel on cherche à exprimer une variable aléatoire Y dite variable à expliquée en fonction d'une ou plusieurs variables explicatives X auxquelles sont associées les paramètres inconnus β du modèle selon la formule :

$$Y = X\beta + U$$

- Y : est le vecteur d'observations des valeurs du Nombre de Reynolds {y1,...yn},
– X : est la matrice (n,k) de rang k, contenant les n valeurs mesurées des k variables/indicateurs explicatifs,
- β est le vecteur de contenant les k coefficients à estimer du modèle,
- U est le vecteur des erreurs du modèle.

Ici nous sommes dans le cas de la régression linéaire multiple (X est alors composée de la variable constant 1 et des k variables explicatives). Elle est utilisée entre une variable quantitative (à expliquer) et plusieurs autres variables quantitatives (explicatives), la mise en œuvre d'une régression impose l'existence d'une relation de cause à effet entre les variables prises en compte dans le modèle.

Aussi l'utilisation de modèles statistiques peut répondre à plusieurs besoins : descriptif, explicatif, prédicatif. Dans notre cas nous utilisons ce modèle statistique dans un premier temps dans un but de prédiction. L'accent est alors mis sur la qualité des estimateurs et des prédicteurs qui doivent, dans notre cas, minimiser la mae [3]. L'erreur absolue moyenne (MAE) est moyenne arithmétique des valeurs absolues des écarts, elle sert à comparer plusieurs modèles ou prévisions par rapport à une série d'observations, mais aussi différentes méthodes de prédiction entre elles. Elle nous donne une meilleure idée de la qualité de prédiction, en revanche il n'est pas possible de savoir si le modèle a tendance à sous ou sur-estimer les prédictions.

Le bon modèle n'est donc plus celui qui explique le mieux les données au sens d'un R^2 très élevé au prix d'un nombre important de variables pouvant introduire des effets de colinéarités. Le bon modèle est celui qui conduit aux prédictions les plus fiables et donc une mae minimale. Ceci conduit à rechercher un modèle optimal c'est-à-dire avec un nombre convenable de variables explicatives en utilisant des méthodes de sélection de variables si nécessaire.

2.1.2 La démarche

Nous établissons dans un premier temps notre modèle avec les variables à notre dispositions.

Comme énoncé précédemment notre objectif est d'obtenir le modèle optimal, nous procédons alors à une sélection de variable pas à pas en utilisant la méthode Stepwise. Cet algorithme introduit une étape d'élimination de variable après chaque étape de sélection afin de retirer du modèle d'éventuels variables qui seraient devenues moins indispensables du fait de la présence de celles nouvellement introduites.
Sous R le critère de comparaison des modèles est l'AIC qui est comparé pour chaque modèle en supprimant ou en ajoutant la variable qui conduit à la meilleure amélioration de l'AIC (plus petit AIC).

3. Mean absolute error : erreur absolue moyenne

L'AIC utilise le maximum de vraisemblance, il pénalise les modèles comportant trop de variables, qui pourrait entraîner un sur-apprentissage des données et généralisent mal.

$$AIC = -2lnL(\theta) + 2k$$

Nous évaluons ensuite :

— La significativité globale du modèle (Test de Fisher)
H0 : absence de significativité globale du modèle
H1 : significativité globale du modèle

— La significativité des coefficients (Test de Student)
H0 : absence de significativité de l'effet du coefficient
H1 : significativité de l'effet du coefficient

Suite à la vérification de nos hypothèses nous pouvons analyser le R^2 ajusté :

$$R^2 = \frac{SSR}{SST} \qquad R^2 ajuste = 1 - \frac{n-1}{n-p-1}(1-R^2)$$

- SSR : somme des carrés des résidus,
- SST : somme totale des carrés,
- n : nombre d'observations,
- p : nombre de variables explicatives.

Le R^2 est le coefficient de détermination, il représente la proportion de la variabilité des données qui est expliquée par le modèle. Plus le R^2 est proche de 1, meilleur est le modèle c'est un indice de qualité mais qui a la propriété d'être monotone croissant en fonction du nombre de variables.Le R^2 ajusté est une correction du R^2 qui permet de prendre en compte le nombre de variables utilisées dans le modèle.

Nous calculerons ensuite la moyenne de l'erreur moyenne absolu sur 20 germes différents appliqués lors de la sélection du jeu de test et d'entraînement, afin d'évaluer la qualité du modèle.

2.2 Réseaux de neurones artificiel

2.2.1 Qu'est ce qu'un réseau de neurone

Un réseau de neurones artificiel est un modèle, qui est inspiré de la structure en réseau des neurones biologiques. Ils fonctionnent avec des couches de neurones connectés. Il peuvent servir notamment à la classification et à la régression. Pour ce projet, seule la régression nous intéresse. Les réseaux de neurones nous permettent d'atteindre des résultats bien meilleurs qu'avec la régression par modèles linéaires multiples. Cependant un réseau de neurones est plus coûteux en terme d'énergie et de temps.

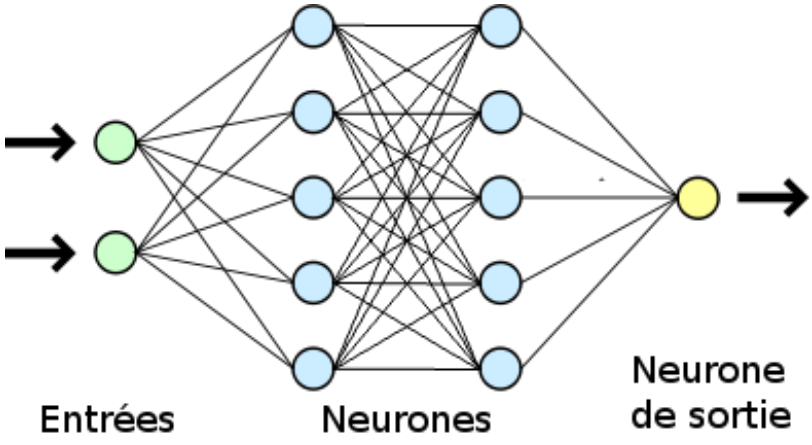


FIGURE 9 – Schéma d'un réseau de neurones

De plus, il est beaucoup plus complexe de comprendre le modèle obtenu avec cette méthode. Lorsque l'on regarde la structure du réseau de la figure 9 [4], on peut voir qu'il y a les entrées, la sortie, et entre les deux il y a 2 couches de 5 neurones. Ces deux couches sont appelées hidden layers [5]. Ces couches sont comme leur nom l'indique, complexes à interpréter. C'est ce qui rend les réseaux de neurones si opaques.

Les réseaux de neurones sont initialisés en donnant un coefficient/poids aléatoire à chaque connexion entre neurones. Les poids seront ensuite affinés à chaque itération grâce à une descente de gradient, de manière à avoir l'EQM [6] la plus proche de 0. L'EQM est un bon indicateur d'écart pour le modèle. En effet, il est plus punitif pour les gros écarts entre la prédiction et la valeur attendue, et il saura récompenser les prédictions très proches de la bonne réponse.

2.2.2 La démarche

La première question que l'on se pose lors de la création d'un réseau de neurones est la suivante : comment paramétrer le réseau ? En effet, il est possible de mettre autant de couche de neurones que l'on veut, avec dans chacune de ces couches, n'importe quel nombre de neurones.

Dans la majorité des réseaux de neurones que l'on peut trouver, le nombre de neurones par couches est une puissance de deux [7]. Nous avons donc testé différentes configurations à partir des puissances de deux. On a pu voir que les configurations avec moins de 64 neurones ne nous donnait pas de résultats très précis, et mettaient beaucoup de temps à se stabiliser.

Étant donné que le fait de mettre peu de neurones dans le réseau ne nous permet pas de gagner du temps, ni d'avoir de meilleurs résultats, l'intérêt de d'utiliser de petits réseaux de neurones est nul. Chaque couches du réseau de neurones sera donc composée d'au moins 64 neurones.

On dit qu'un modèle se stabilise lors ce que la fonction d'erreur avec laquelle on évalue le modèle au fil des itérations n'évolue plus. On arrêtera à ce moment là l'apprentissage du réseau.

En ce qui concerne le nombre de couches du réseau, en général deux ou trois couches cachées suffisent. Nous avons tout de même testé d'augmenter leur nombre jusqu'à 5, et nous n'avons pas noté de réduction de la mae sur le jeu de test par rapport à un réseau composé de 3 couches cachées. De plus le risques de sur-apprentissage augmente au delà de 5 couches cachées. On préférera donc un réseau de neurone à 3 couches cachées.

On testera donc sur chacune des couches le nombre de neurones suivants : {64, 128, 256}. Du au fait que la puissance de calcul de nos ordinateurs personnels, nous ne pouvons pas tester de plus gros réseaux.

On a donc 27 différentes configurations de réseaux de neurones à tester. Pour les tester nous avons réalisé 5 modèles pour chaque configuration de réseau. Grâce à cela nous avons pu comparer les résultats de chacune des configurations. On en remarque une qui sort du lot. Il s'agit de la configuration suivante : {64, 256, 256}.

4. C'est un exemple, un réseau de neurones peut être composé d'autant de neurones qu'on veut
5. Couches cachées
6. Erreur Quadratique Moyenne
7. 2, 4, 8, 16, 32, 64 ...

3 Résultats

Dans cette partie, nous allons présenter les jeux de données que nous avons utilisé. Nous verrons ensuite les résultats que nous avons obtenu grâce ces mêmes jeux de données, d'abord avec les modèles linéaires, puis avec les réseaux de neurones.

3.1 Les jeux de données

Nous allons construire différents modèles avec différentes variables afin de permettre de prédire le nombre de Reynolds sans avoir nécessairement besoin de réaliser les expériences. Il est donc essentiel de choisir les bonnes variables/indicateurs afin d'assurer la véracité de notre modèle.

Nous séparons les jeux de données en deux parties, afin de réaliser l'apprentissage sur le jeu d'entraînement et les prédictions sur le jeu de test. Le jeu d'entraînement prendra en compte de manière aléatoire 80% et le jeu de test 20% du jeu de données initial. Rappelons que la qualité de chaque modèle issue d'un jeu de donnée sera évalué par l'erreur moyenne absolue (mae) sur le jeu de test. Pour le modèle linéaire nous calculerons la moyenne de la mae du modèle sur 20 échantillons de test/entraînement différents.

- 1er jeu de données :

Données : Interface fluide/air pour chacun des jets, au dernier pas de temps sur un domaine de 140. Pour un nombre de Reynolds de 100 à 900 par pas de 10 et pour une amplitude de 0.01 à 0.15 par pas de 0.01. Suite aux calculs des indicateurs nous disposons 1183 observations et 12 variables.

Equation :
$Reynolds = Amplitude + Premiere\_cesure + nb\_cesures + long\_brisures + integrale + Amplitude * Premiere\_cesure + Amplitude^2 + Premiere\_cesure^2 + nb\_cesures^2 + long\_brisures^2 + Amplitude * Premiere\_cesure^2$

Nous avons calculé la longueur de brisure qui est la somme des distances de séparation du fluide sur toute l'interface. On ajoute également une interaction entre la variable Amplitude et Premiere_cesure afin de prendre en compte l'influence de ces deux paramètres couplés. Nous avons également ajouté les variables explicatives au carré pour les ajuster à la variable dépendante Reynolds de manière non-linéaire.

- 2ème jeu de données :

Données : Interface fluide/air pour chacun des jets, sur les 20 dernières secondes de simulation (125s-145s) sur un domaine de 140. Pour un nombre de Reynolds de 100 à 900 par pas de 10 et pour une amplitude de 0.01 à 0.15 par pas de 0.01. Suite aux calculs des indicateurs nous disposons de 25389 observations et 22 variables.

Equation :
$Reynolds = Amplitude + Temps + Premiere\_cesure + nb\_cesures + long\_brisures + integrale + Amplitude * Premiere\_cesure + Amplitude^2 + Temps^2 + Premiere\_cesure^2 + nb\_cesures^2 + long\_brisures^2 + integrale^2 + Amplitude * Premiere\_cesure^2 + Amplitude^3 + Temps^3 + Premiere\_cesure^3 + nb\_cesures^3 + long\_brisures^3 + integrale^3 + Amplitude * Premiere\_cesure^3$

On ajoute la variable Temps à notre modèle c'est le temps en seconde à laquelle à été prise la photo de l'interface. On aura donc plusieurs Temps pour un même jet. De plus nous avons ajouté la variable intégral afin d'avoir un indicateur en plus sur la morphologie du jet. Nous avons également ajouté les variables explicatives au cube pour les ajuster encore plus à la variable dépendante Reynolds de manière non-linéaire.

- 3ème jeu de données :

Données : Interface fluide/air pour chacun des jets, sur les 2 secondes de simulation par pas de 0.2s (223s-225s) soit 11 interfaces par jet. Pour un nombre de Reynolds de 100 à 790 par pas de 10 et pour une amplitude de 0.01 à 0.15 par pas de 0.01. Cette fois ci le domaine fait 200 de long (contre 140 précédemment) et la simulation dure 225s. Suite aux calculs des indicateurs nous disposons de

11256 observations et 34 variables.

Equation :
$Reynolds = Amplitude + Temps + Premiere\_cesure + nb\_cesures + volume + surface + ratio\_sv + satellite + long\_onde + poly4 + poly3 + poly2 + poly1 + poly0 + Premiere\_cesure * Amplitude + Amplitude^2 + Temps^2 + Premiere\_cesure^2 + nb\_cesures^2 + volume^2 + surface^2 + ratio\_sv^2 + long\_onde^2 + Premiere\_cesure * Amplitude^2 + Amplitude^3 + Temps^3 + Premiere\_cesure^3 + nb\_cesures^3 + volume^3 + surface^3 + ratio\_sv^3 + long\_onde^2 + Premiere\_cesure * Amplitude^3$

Suite aux résultats de nos précédentes prédictions, nous avons décidé de supprimer la variable intégrale et de la remplacer par des variables plus représentatives du jet, les coefficients du polynôme de degré 4, son volume et sa surface ainsi que son ratio. D'autre part nous avons remplacé la variable longueur_brisure par long_onde (distance entre 2 "bosses" avant que le jet ne se brise). Nous avons aussi ajouté une variable concernant la préséance de satellite ou non (codée 0 ou 1).

- 4eme jeu de données :

Données : Interface fluide/air pour chacun des jets, sur les 6 secondes de simulation par pas de 0.5s (344s-350s). Pour un nombre de Reynolds de 100 à 900 par pas de 1 et pour une amplitude de 0.01 à 0.05 par pas de 0.01 et avec un pas de 5 Reynolds pour les amplitudes de 0.06 à 0.1 par pas de 0.01. Cette fois ci le domaine fait 240 de long (contre 200 précédemment) et la simulation dure 225s. Suite aux calculs des indicateurs nous disposons de 42367 observations et 34 variables.

Equation :

$Reynolds = Amplitude + Temps + Premiere\_cesure + nb\_cesures + volume + surface + ratio\_sv + satellite + long\_onde + poly4 + poly3 + poly2 + poly1 + poly0 + Premiere\_cesure * Amplitude + Amplitude^2 + Temps^2 + Premiere\_cesure^2 + nb\_cesures^2 + volume^2 + surface^2 + ratio\_sv^2 + long\_onde^2 + Premiere\_cesure * Amplitude^2 + Amplitude^3 + Temps^3 + Premiere\_cesure^3 + nb\_cesures^3 + volume^3 + surface^3 + ratio\_sv^3 + long\_onde^3 + Premiere\_cesure * Amplitude^3$

Nous avons supprimé les points parasites de nos interfaces fluide/air présent lors de la simulation informatique (cf : correction/nettoyage)

3.2 Modèle Linéaire

Pour chaque jeu de donnée différent nous allons appliquer la démarche mis en place pour réaliser la régression linéaire et analyser les résultats du modèle.

3.2.1 1er Jeu de données

Nous allons réaliser une sélection de variables via la méthode Stepwise sur l'équation de base du jeu de donnée, nous obtenons ce modèle :

```
Call:
lm(formula = Reynolds ~ Amplitude + Premiere_cesure + nb_cesures +
    long_brisures + integrale + data.Amplitude...data.Premiere_cesure +
    Amplitude.2 + Premiere_cesure.2 + nb_cesures.2 + long_brisures.2 +
    data.Amplitude...data.Premiere_cesure.2, data = Train)

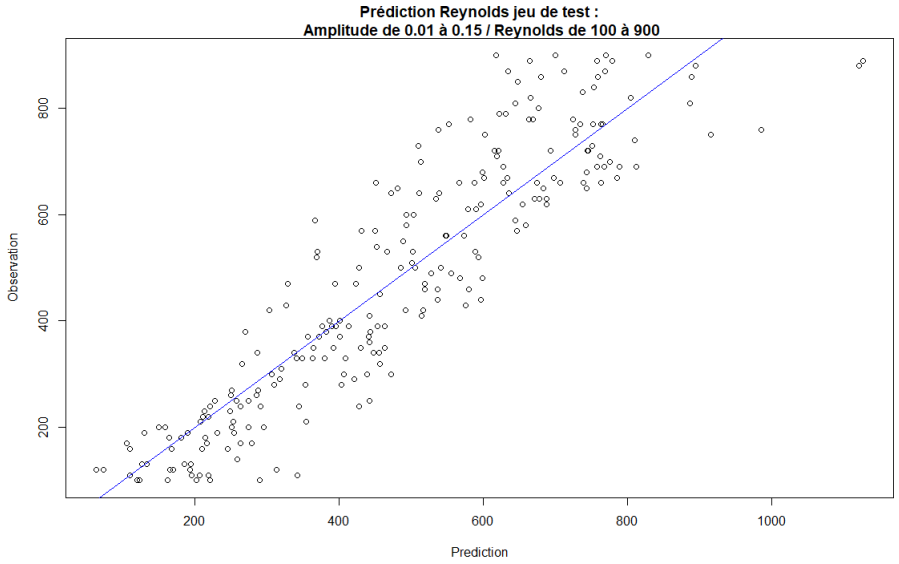
Residuals:
    Min      1Q  Median      3Q     Max
-386.28  -65.15   -7.07   54.59  402.43

Coefficients:
                                        Estimate Std. Error t value Pr(>|t|)
(Intercept)                           -1.769e+03  3.862e+02  -4.582 5.24e-06 ***
Amplitude                             -4.629e+04  3.015e+03 -15.353  < 2e-16 ***
Premiere_cesure                       -7.498e+01  4.612e+00 -16.257  < 2e-16 ***
nb_cesures                            -4.811e+01  9.566e+00  -5.029 5.90e-07 ***
long_brisures                          5.065e+01  5.195e+00   9.750  < 2e-16 ***
integrale                              5.787e+01  1.642e+00  35.255  < 2e-16 ***
data.Amplitude...data.Premiere_cesure  4.086e+02  2.538e+01  16.098  < 2e-16 ***
Amplitude.2                            1.317e+05  1.059e+04  12.445  < 2e-16 ***
Premiere_cesure.2                      2.329e-01  1.927e-02  12.081  < 2e-16 ***
nb_cesures.2                           1.919e+00  4.365e-01   4.397 1.22e-05 ***
long_brisures.2                       -1.242e+00  1.750e-01  -7.098 2.50e-12 ***
data.Amplitude...data.Premiere_cesure.2 -8.465e+00  7.231e-01 -11.707  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

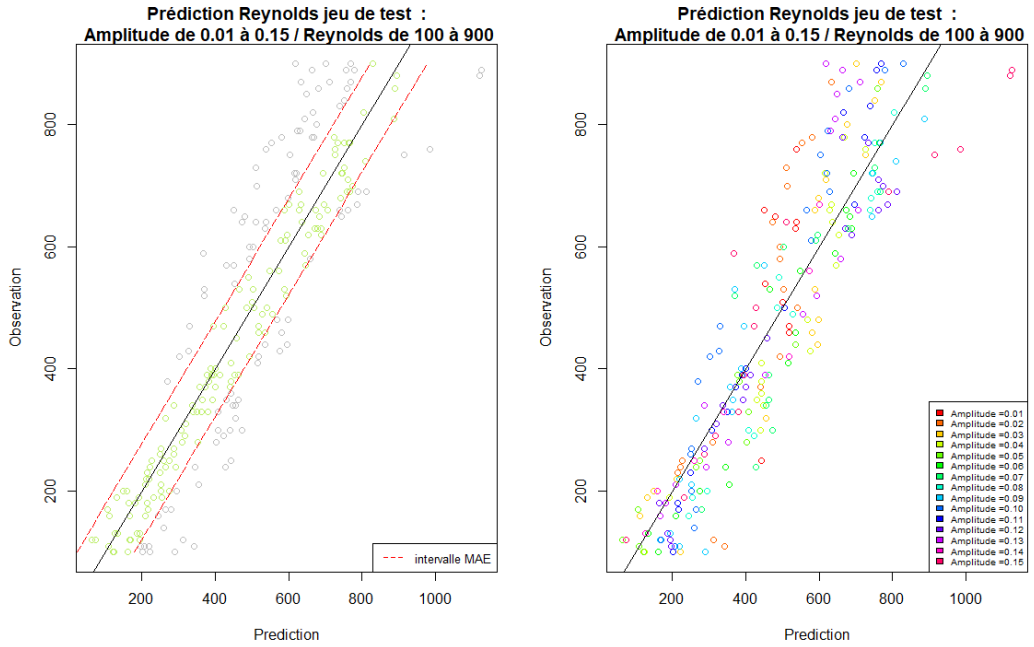
Residual standard error: 106.9 on 934 degrees of freedom
Multiple R-squared:  0.7819,    Adjusted R-squared:  0.7793
F-statistic: 304.4 on 11 and 934 DF,  p-value: < 2.2e-16
```

FIGURE 10 – Régression modèle 1

Aucune variable n'a été exclu lors de la sélection de variables.
Le modèle est globalement significatif avec une p-value associée inférieure à 1%.
Tous les coefficients sont bien significativement différents de zéro, avec une p-value associée inférieure à 1%.
On obtient un R^2 ajusté de 0.78, il y a 78% de la variabilité des données qui est expliquée par le modèle.



On calcul une erreur absolue moyenne sur le jeu de test de 78.21, soit une précision à 78 Reynolds près.



On observe que pour la prédiction des Reynolds compris entre 500 et 900, ce sont les observations avec une très faible amplitude qui sont sous-estimées. En revanche on constate que pour la prédiction des Reynolds supérieur à 700, ce sont les observations avec une forte amplitude à 0.15 qui sont surestimées tandis que pour les observations avec une amplitude à 0.11/0.12 sont sous-estimées.

En testant sur 20 germes différents pour la répartition du jeu d'entraînement et de test on obtient une erreur absolue moyenne sur le jeu de test de 80.20, soit une précision à 80 Reynolds près.

3.2.2 2ème Jeu de données

Nous allons réaliser une sélection de variables via la méthode Stepwise sur l'équation de base du jeu de donnée, nous obtenons ce modèle :

```
Call:
lm(formula = Reynolds ~ Amplitude + Temps + Premiere_cesure +
    nb_cesures + long_brisures + integrale + Amplitude...Premiere_cesure +
    Amplitude.2 + Temps.2 + Premiere_cesure.2 + nb_cesures.2 +
    long_brisures.2 + integrale.2 + Amplitude...Premiere_cesure.2 +
    Amplitude.3 + Temps.3 + Premiere_cesure.3 + nb_cesures.3 +
    long_brisures.3 + integrale.3 + Amplitude...Premiere_cesure.3,
    data = Train)

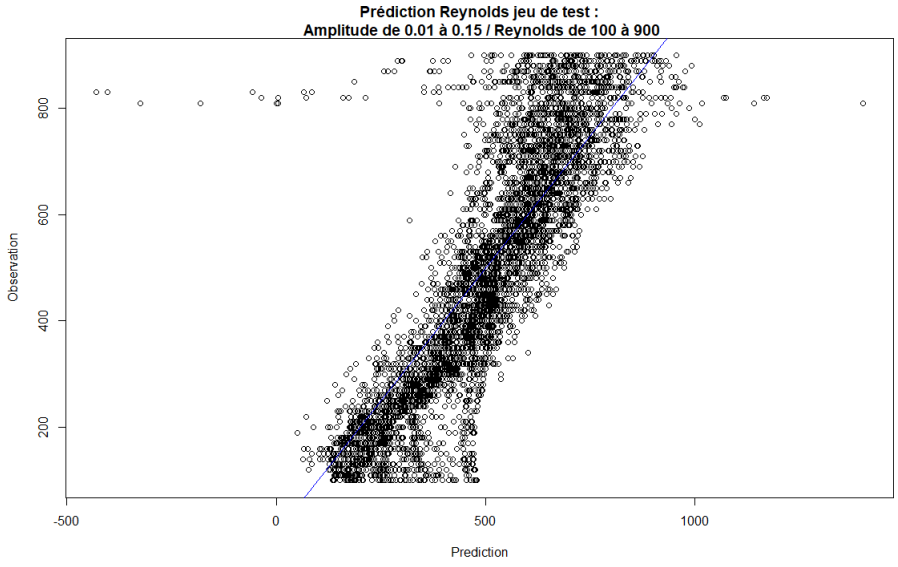
Residuals:
    Min      1Q  Median      3Q     Max
-568.90  -77.91  -19.80   55.70 1255.46

Coefficients:
                                Estimate Std. Error t value Pr(>|t|)
(Intercept)                   -4.163e+04  1.409e+04  -2.955  0.00313 **
Amplitude                     -8.651e+04  2.860e+03 -30.244  < 2e-16 ***
Temps                          3.482e+03  3.172e+02  10.976  < 2e-16 ***
Premiere_cesure               -8.460e+01  5.469e+00 -15.469  < 2e-16 ***
nb_cesures                     2.009e+01  4.148e+00   4.842 1.29e-06 ***
long_brisures                  4.993e+01  2.407e+00  20.742  < 2e-16 ***
integrale                     -2.582e+03  1.071e+02 -24.108  < 2e-16 ***
Amplitude...Premiere_cesure    7.064e+02  2.361e+01  29.919  < 2e-16 ***
Amplitude.2                    4.384e+05  1.969e+04  22.264  < 2e-16 ***
Temps.2                       -2.655e+01  2.351e+00 -11.293  < 2e-16 ***
Premiere_cesure.2              1.440e-01  4.952e-02   2.908  0.00364 **
nb_cesures.2                  -4.663e+00  5.426e-01  -8.594  < 2e-16 ***
long_brisures.2               -4.143e+00  2.482e-01 -16.694  < 2e-16 ***
integrale.2                    2.137e+01  8.717e-01  24.520  < 2e-16 ***
Amplitude...Premiere_cesure.2 -2.762e+01  1.315e+00 -20.997  < 2e-16 ***
Amplitude.3                   -9.273e+05  5.549e+04 -16.712  < 2e-16 ***
Temps.3                        6.684e-02  5.802e-03  11.520  < 2e-16 ***
Premiere_cesure.3              8.252e-04  1.595e-04   5.173 2.33e-07 ***
nb_cesures.3                   2.471e-01  2.256e-02  10.956  < 2e-16 ***
long_brisures.3                1.209e-01  7.620e-03  15.868  < 2e-16 ***
integrale.3                   -5.799e-02  2.361e-03 -24.568  < 2e-16 ***
Amplitude...Premiere_cesure.3  4.431e-01  2.889e-02  15.337  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

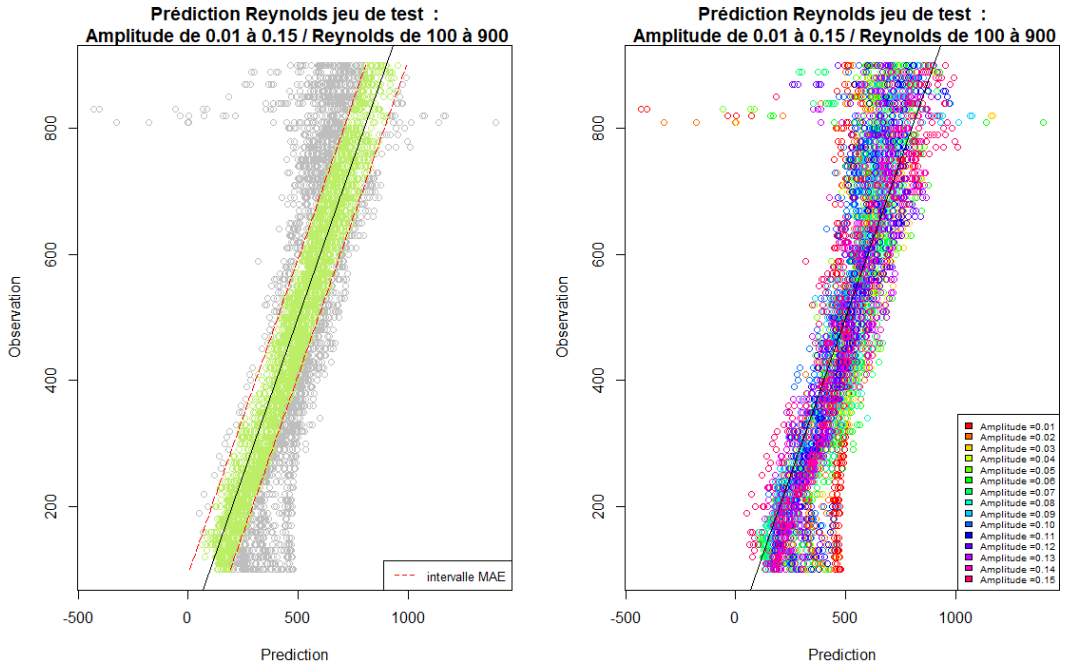
Residual standard error: 139.2 on 20289 degrees of freedom
Multiple R-squared:  0.6429,    Adjusted R-squared:  0.6426
F-statistic:  1740 on 21 and 20289 DF,  p-value: < 2.2e-16
```

FIGURE 11 – Régression modèle 2

Aucune variable n'a été exclu lors de la sélection de variables.
Le modèle est globalement significatif avec une p-value associée inférieure à 1%.
Tous les coefficients sont bien significativement différents de zéro, avec une p-value associée inférieure à 1%.
On obtient un R^2 ajusté de 0.64, il y a 64% de la variabilité des données qui est expliquée par le modèle.



On calcul une erreur absolue moyenne sur le jeu de test de 92.51, soit une précision à 93 Reynolds près.



On observe que pour la prédiction des Reynolds compris entre 100 et 400, ce sont les observations avec une très faible amplitude qui sont surestimées. En revanche on constate que pour la prédiction des Reynolds supérieur à 750, certaines observations sont fortement surestimées et sous-estimées quel que soit l'amplitude.

En testant sur 20 germes différents pour la répartition du jeu d'entraînement et de test on obtient une erreur absolue moyenne sur le jeu de test de 93.48, soit une précision à 93 Reynolds près.

3.2.3 3ème Jeu de données

Nous allons réaliser une sélection de variables via la méthode Stepwise sur l'équation de base du jeu de donnée, nous obtenons ce modèle :

```
Call:
lm(formula = Reynolds ~ Amplitude + Temps + Premiere_cesure +
    volume + surface + ratio_sv + long_onde + poly4 + poly3 +
    poly2 + poly1 + poly0 + Premiere_cesure...Amplitude + Amplitude.2 +
    Temps.2 + Premiere_cesure.2 + nb_cesures.2 + volume.2 + surface.2 +
    ratio_sv.2 + long_onde.2 + Premiere_cesure...Amplitude.2 +
    Amplitude.3 + Premiere_cesure.3 + nb_cesures.3 + volume.3 +
    surface.3 + ratio_sv.3 + long_onde.3 + Premiere_cesure...Amplitude.3,
    data = Train)

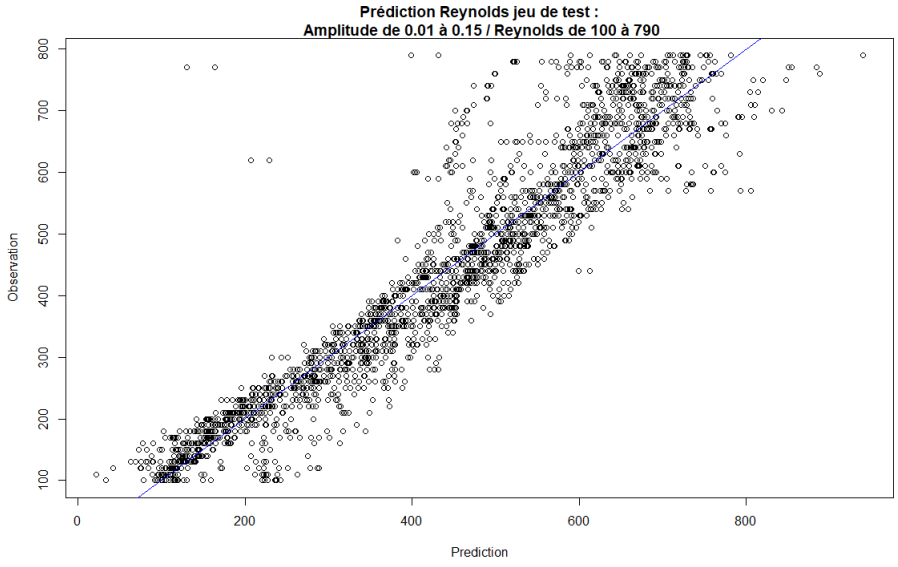
Residuals:
    Min      1Q  Median      3Q     Max
-253.30  -37.04   -1.83   27.84  650.05

Coefficients:
                                  Estimate Std. Error t value Pr(>|t|)
(Intercept)                     -8.755e+06  9.654e+05  -9.069  < 2e-16 ***
Amplitude                       -3.720e+04  1.733e+03 -21.468  < 2e-16 ***
Temps                            1.400e+03  9.366e+02   1.495   0.1350
Premiere_cesure                 -5.668e+01  2.378e+00 -23.839  < 2e-16 ***
volume                           2.103e+04  2.419e+03   8.694  < 2e-16 ***
surface                         -1.421e+04  1.490e+03  -9.537  < 2e-16 ***
ratio_sv                         9.008e+06  9.517e+05   9.465  < 2e-16 ***
long_onde                        6.217e+01  3.638e+01   1.709   0.0875 .
poly4                            1.341e+11  9.941e+09  13.492  < 2e-16 ***
poly3                            4.139e+08  5.811e+07   7.123 1.14e-12 ***
poly2                           -1.948e+06  3.661e+05  -5.320 1.06e-07 ***
poly1                           -7.273e+04  2.770e+03 -26.253  < 2e-16 ***
poly0                           -7.822e+02  3.901e+01 -20.055  < 2e-16 ***
Premiere_cesure...Amplitude      3.056e+02  1.282e+01  23.846  < 2e-16 ***
Amplitude.2                      1.727e+05  1.223e+04  14.121  < 2e-16 ***
Temps.2                         -3.155e+00  2.090e+00  -1.509   0.1313
Premiere_cesure.2                2.901e-01  1.643e-02  17.660  < 2e-16 ***
nb_cesures.2                    -6.534e-01  5.067e-02 -12.895  < 2e-16 ***
volume.2                        -1.459e+01  1.830e+00  -7.974 1.72e-15 ***
surface.2                        6.675e+00  6.916e-01   9.652  < 2e-16 ***
ratio_sv.2                      -2.688e+06  2.822e+05  -9.524  < 2e-16 ***
long_onde.2                     -8.161e+00  4.451e+00  -1.833   0.0668 .
Premiere_cesure...Amplitude.2   -8.777e+00  6.009e-01 -14.607  < 2e-16 ***
Amplitude.3                     -5.179e+05  3.596e+04 -14.402  < 2e-16 ***
Premiere_cesure.3               -6.704e-04  4.108e-05 -16.319  < 2e-16 ***
nb_cesures.3                     2.234e-02  1.454e-03  15.369  < 2e-16 ***
volume.3                         4.464e-03  6.155e-04   7.252 4.43e-13 ***
surface.3                       -1.388e-03  1.424e-04  -9.744  < 2e-16 ***
ratio_sv.3                       3.556e+05  3.713e+04   9.577  < 2e-16 ***
long_onde.3                      2.932e-01  1.667e-01   1.759   0.0786 .
Premiere_cesure...Amplitude.3    1.185e-01  1.033e-02  11.475  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 69.92 on 8974 degrees of freedom
Multiple R-squared:  0.8801,    Adjusted R-squared:  0.8797
F-statistic:  2196 on 30 and 8974 DF,  p-value: < 2.2e-16
```

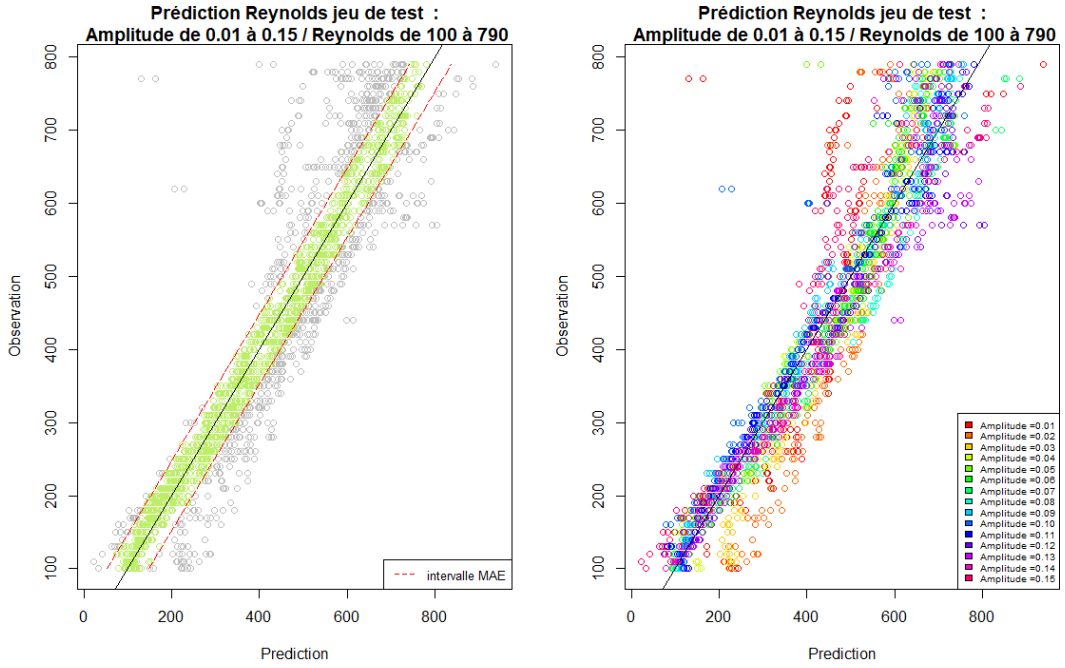
FIGURE 12 – Régression modèle 3

Les variables satellite et nb_cesures ont été retiré du modèle lors de la sélection de variables. Le modèle est globalement significatif avec une p-value associée inférieure à 1%.
Les coefficients ne sont pas significatifs pour les variables Temps, $Temps^2$.
Long_onde t *long_onde*^{2} et *long_onde*^{3} sont significativement différents de zéro avec une p-value associée inférieure à 10%. Les autres des coefficients sont aussi significativement différents de zéro, avec une p-value associée inférieure à 1%.
On obtient un R^2 ajusté de 0.88, il y a 88% de la variabilité des données qui est expliquée par le modèle.



On calcul une erreur absolue moyenne sur le jeu de test de 47.71, soit une précision à 48 Reynolds

près.



On observe que pour la prédiction des Reynolds compris entre 100 et 400, ce sont les observations avec une très faible amplitude qui sont surestimées, tandis que pour la prédiction des Reynolds supérieure à 500, elles sont sous-estimées. Les prédictions pour les Reynolds à moyennes et fortes amplitudes sont bien prédites.

En testant sur 20 germes différents pour la répartition du jeu d'entraînement et de test on obtient une erreur absolue moyenne sur le jeu de test de 47.93, soit une précision à 48 Reynolds près.

3.2.4 4eme Jeu de données

Nous allons réaliser une sélection de variables via la méthode Stepwise sur l'équation de base du jeu de donnée, nous obtenons ce modèle :

```
Call:
lm(formula = Reynolds ~ Amplitude + Temps + Premiere_cesure +
    nb_cesures + volume + surface + ratio_vs + satellite + long_onde +
    poly4 + poly3 + poly2 + poly1 + poly0 + tableau.Premiere_cesure...tableau.Amplitude +
    Amplitude.2 + Temps.2 + Premiere_cesure.2 + nb_cesures.2 +
    volume.2 + surface.2 + ratio_vs.2 + long_onde.2 + tableau.Premiere_cesure...tableau.Amplitude.2 +
    Amplitude.3 + Temps.3 + Premiere_cesure.3 + nb_cesures.3 +
    volume.3 + surface.3 + ratio_vs.3 + long_onde.3 + tableau.Premiere_cesure...tableau.Amplitude.3,
    data = Train)

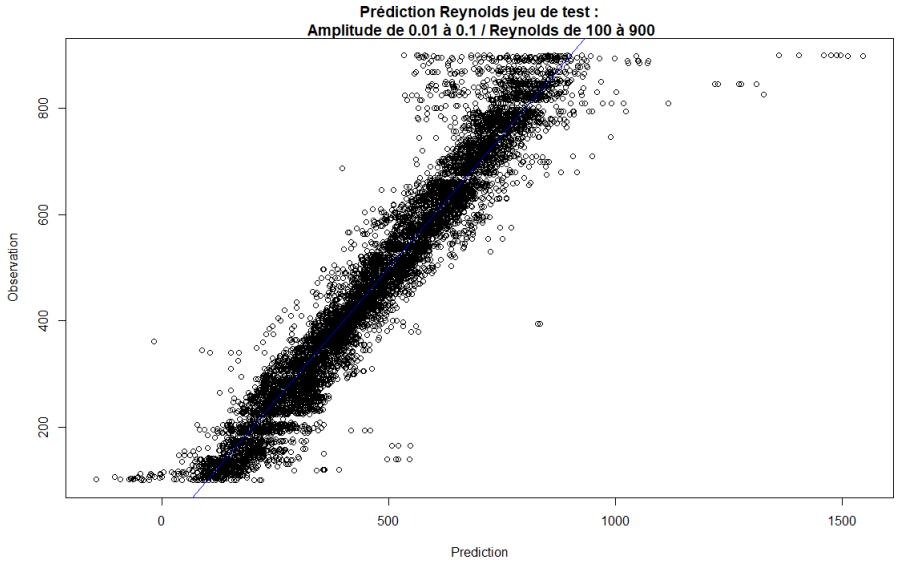
Residuals:
    Min      1Q  Median      3Q     Max
-649.99  -31.23   -2.86   28.93  459.26

Coefficients:
                                                 Estimate Std. Error  t value Pr(>|t|)
(Intercept)                                    -6.029e+07  2.932e+06  -20.566  < 2e-16 ***
Amplitude                                       5.048e+04  1.379e+03   36.604  < 2e-16 ***
Temps                                           5.231e+05  2.535e+04   20.635  < 2e-16 ***
Premiere_cesure                                 2.338e+01  9.796e-01   23.871  < 2e-16 ***
nb_cesures                                     -9.244e+01  1.253e+00  -73.765  < 2e-16 ***
volume                                         -7.315e+02  2.627e+01  -27.840  < 2e-16 ***
surface                                        -6.487e+01  2.254e+01   -2.878 0.004006 **
ratio_vs                                        1.464e+05  2.747e+04    5.332 9.77e-08 ***
satellite                                       3.718e+00  1.363e+00    2.727 0.006396 **
long_onde                                       2.692e+03  1.412e+02   19.067  < 2e-16 ***
poly4                                          -7.093e+10  2.013e+10   -3.524 0.000426 ***
poly3                                          -1.716e+09  1.039e+08  -16.523  < 2e-16 ***
poly2                                          -2.162e+07  5.700e+05  -37.922  < 2e-16 ***
poly1                                          -2.627e+05  3.459e+03  -75.944  < 2e-16 ***
poly0                                          -3.360e+03  2.997e+01 -112.118  < 2e-16 ***
tableau.Premiere_cesure...tableau.Amplitude    -4.347e+02  9.428e+00  -46.108  < 2e-16 ***
Amplitude.2                                    -3.862e+05  1.457e+04  -26.497  < 2e-16 ***
Temps.2                                        -1.509e+03  7.306e+01  -20.653  < 2e-16 ***
Premiere_cesure.2                              -1.758e-01  5.996e-03  -29.322  < 2e-16 ***
nb_cesures.2                                    3.285e+00  4.557e-02   72.097  < 2e-16 ***
volume.2                                        8.459e-01  1.944e-02   43.514  < 2e-16 ***
surface.2                                      -7.887e-02  1.179e-02   -6.690 2.27e-11 ***
ratio_vs.2                                     -1.411e+04  9.007e+03   -1.567 0.117193
long_onde.2                                    -4.376e+02  2.361e+01  -18.534  < 2e-16 ***
tableau.Premiere_cesure...tableau.Amplitude.2  4.348e+01  5.769e-01   75.360  < 2e-16 ***
Amplitude.3                                     3.534e+05  6.705e+04    5.271 1.37e-07 ***
Temps.3                                         1.451e+00  7.018e-02   20.671  < 2e-16 ***
Premiere_cesure.3                               1.817e-04  1.367e-05   13.292  < 2e-16 ***
nb_cesures.3                                   -3.666e-02  5.594e-04  -65.534  < 2e-16 ***
volume.3                                       -2.865e-04  6.951e-06  -41.216  < 2e-16 ***
surface.3                                       3.932e-05  3.281e-06   11.984  < 2e-16 ***
ratio_vs.3                                     -6.578e+03  1.259e+03   -5.226 1.74e-07 ***
long_onde.3                                     2.331e+01  1.309e+00   17.804  < 2e-16 ***
tableau.Premiere_cesure...tableau.Amplitude.3 -1.145e+00  1.403e-02  -81.611  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

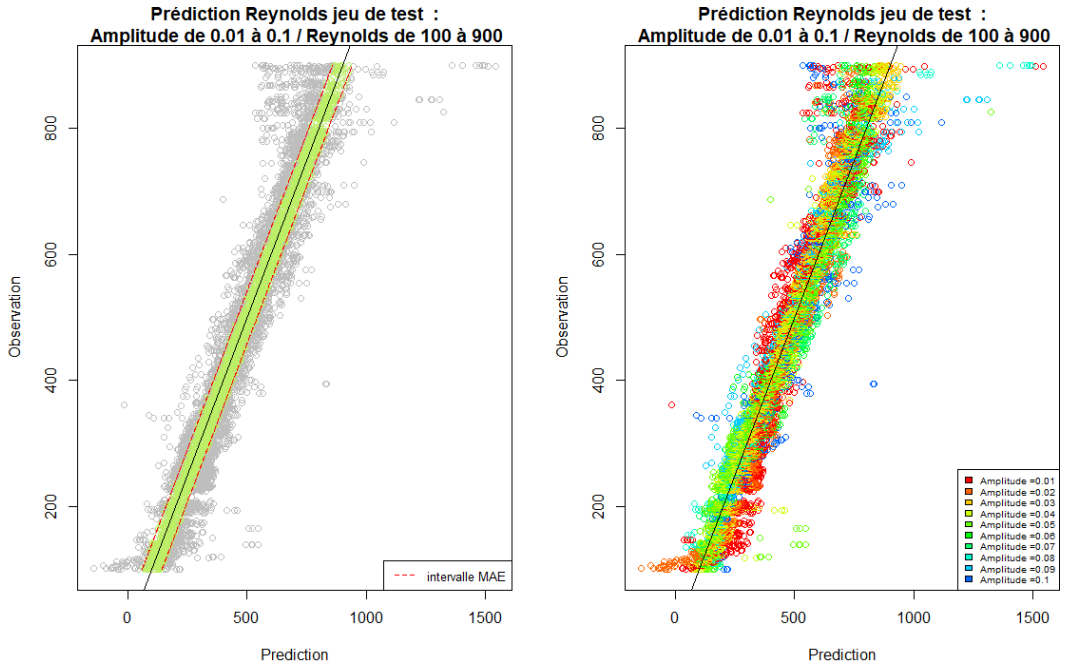
Residual standard error: 60.17 on 33860 degrees of freedom
Multiple R-squared:  0.9226,    Adjusted R-squared:  0.9225
F-statistic: 1.222e+04 on 33 and 33860 DF,  p-value: < 2.2e-16
```

FIGURE 13 – Régression modèle actuel

Aucune variable n'a été exclu lors de la sélection de variables.
Le modèle est globalement significatif avec une p-value associée inférieure à 1%.
Le coefficient n'est pas significatif pour $ratio_v s^2$.
Les autres coefficients sont bien significativement différents de zéro, avec une p-value associée inférieure à 1%.
On obtient un R^2 ajusté de 0.92, il y a 92% de la variabilité des données qui est expliquée par le modèle.



On calcul une erreur quadratique moyenne sur le jeu de test de 41.56, soit une précision à 42 Reynolds près.



On observe que pour la prédiction des Reynolds, ce sont les observations avec des amplitudes extrêmes qui sont le moins bien estimées, tandis que pour les prédictions des Reynolds à moyennes amplitudes sont bien prédites.

En testant sur 20 germes différents pour la répartition du jeu d'entraînement et de test on obtient une erreur quadratique moyenne sur le jeu de test de 41.02, soit une précision à 41 Reynolds près.

3.3 Réseau de neurones

Pour chaque jeu de donnée différent nous allons appliquer la démarche mis en place pour réaliser le réseau de neurone et analyser la mae des prédictions.

3.3.1 1er Jeu de données

- Meilleur modèle :

Le graphique ci-dessous montre l'évolution de la MAE par itération dans le modèle (epoch). Les points de validation ont été calculés avec le jeu d'entraînement. On préférera un modèle avec une validation stable, de manière à ce que le modèle puisse s'adapter plus facilement à de nouvelles données.

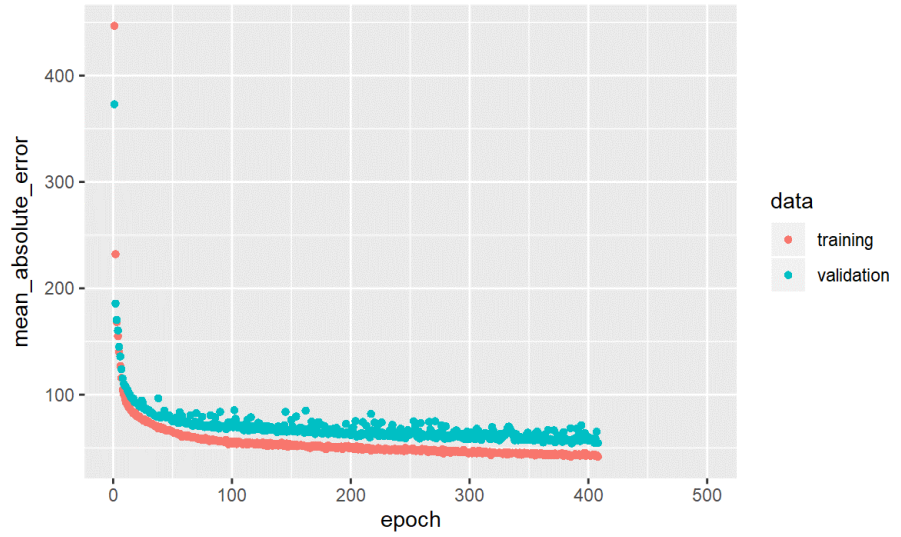


FIGURE 14 – Évolution de la MAE au fil des itérations (1er dataset meilleur modèle)

Ce modèle a duré 400 époques, ce qui est plus élevé que la moyenne des modèles créé avec ce tableau. En effet le modèle s'arrête lorsque les résultats de la validation se stabilisent. En effet il s'agit d'éviter le sur-apprentissage. Si on fait trop tourner le modèle, celui-ci ne sera plus capable de s'adapter à d'autres jeux de données. Ainsi il sera inutile. On peut voir avec l'évolution des MAE du jeu de test que le modèle est plutôt stable, il n'y a pas trop de variations.

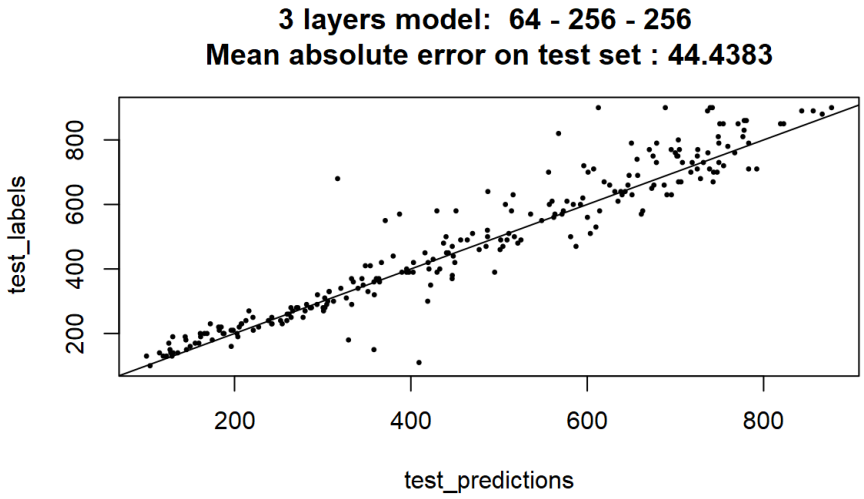


FIGURE 15 – Nuage de points : prédictions x valeurs réelles (1er dataset meilleur modèle)

On a une MAE de 44.44. On ne peut pas encore affirmer que le modèle est précis. Cependant à faible Reynolds les prédictions sont bonnes, même si elles sous-estiment légèrement les Reynolds. Lorsque le nombre de Reynolds est supérieur à 400 les résultats sont bien moins précis. On voit notamment que le modèle peut fortement sous-estimer les Reynolds.

- Modèle moyen :

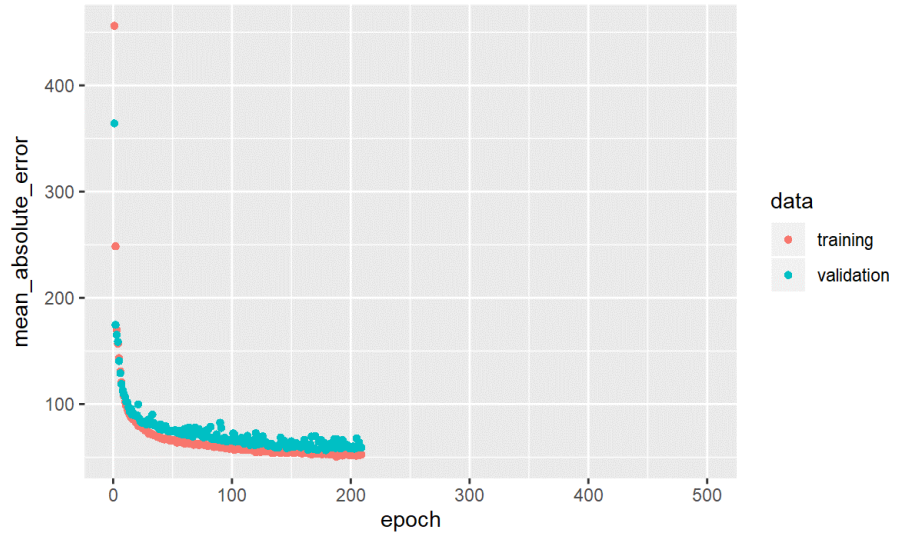


FIGURE 16 – Évolution de la MAE au fil des itérations (1er dataset modèle moyen)

On peut voir que ce modèle a été effectué en légèrement plus de 200 époques, ce qui est bien moins élevé que le premier modèle. Les résultats de validation sont relativement proches des résultats d'entraînement lors de l'évolution du modèle. Le modèle est plutôt stable.

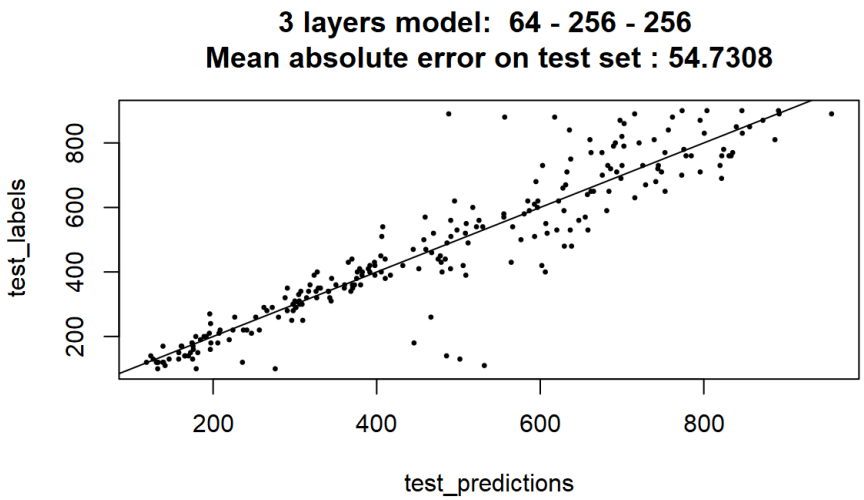


FIGURE 17 – Nuage de points : prédictions x valeurs réelles (1er dataset modèle moyen)

Ce modèle a une MAE de 54.73. Ce qui est loin d'être assez précis. On peut le voir sur le graphique, il y a plus de points isolés que sur le modèle précédent. La variation est également plus grande. Sur 100 modèles, on a en moyenne une MAE de 55.8732. Voici la répartition des MAE sur tous les modèles.

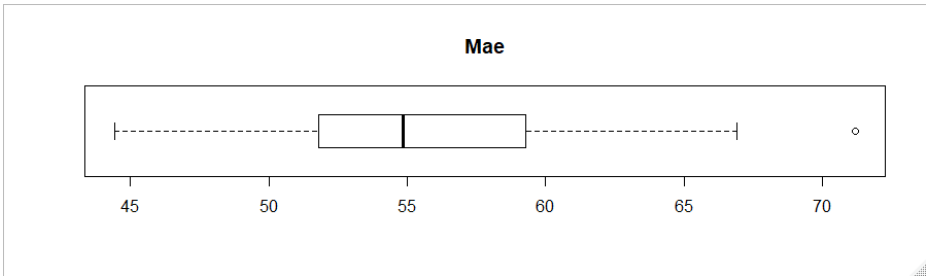


FIGURE 18 – Boxplot : MAE des 100 modèles créés (1er dataset)

3.3.2 2eme Jeu de données

- Meilleur modèle :

Ce modèle a obtenu une mae de 22.5, ce qui est un très gros progrès par rapport au dernier jeu de données.

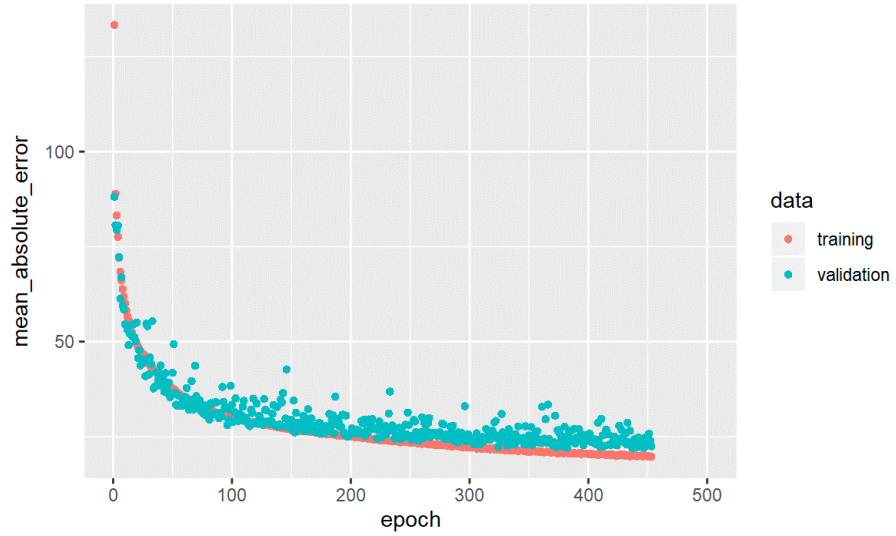


FIGURE 19 – Évolution de la MAE au fil des itérations (2eme dataset meilleur modèle)

On peut voir que ce modèle a itéré plus de 450 fois. C'est à cause du fait qu'il n'arrivait pas à se stabiliser. Les calculs ont duré 9 minutes avec l'importante quantité de données. Le fait qu'il ait autant itéré lui donne un avantage au niveau du score.

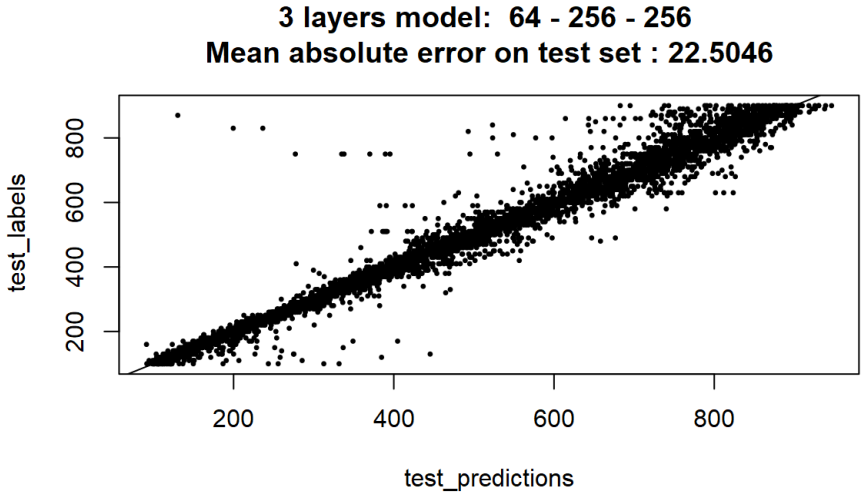


FIGURE 20 – Nuage de points : prédictions x valeurs réelles (2eme dataset meilleur modèle)

Les points sont très concentrés autours de la diagonale. On remarque toujours que plus les Reynolds sont haut, plus les points sont dispersés autours de cette droite. On trouve également un bon nombre de points isolés.

- Modèle moyen :

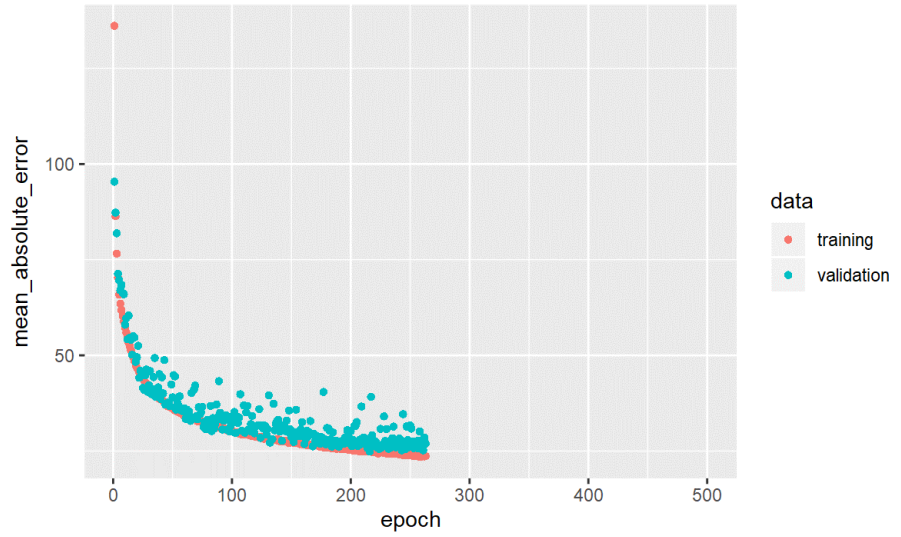


FIGURE 21 – Évolution de la MAE au fil des itérations (2eme dataset modèle moyen)

On constate une nouvelle fois que le modèle moyen itère bien moins que le "meilleur modèle". On peut voir que celui-ci s'est bien stabilisé à partir de 250 itérations. Il obtient au final une MAE de 27.1.

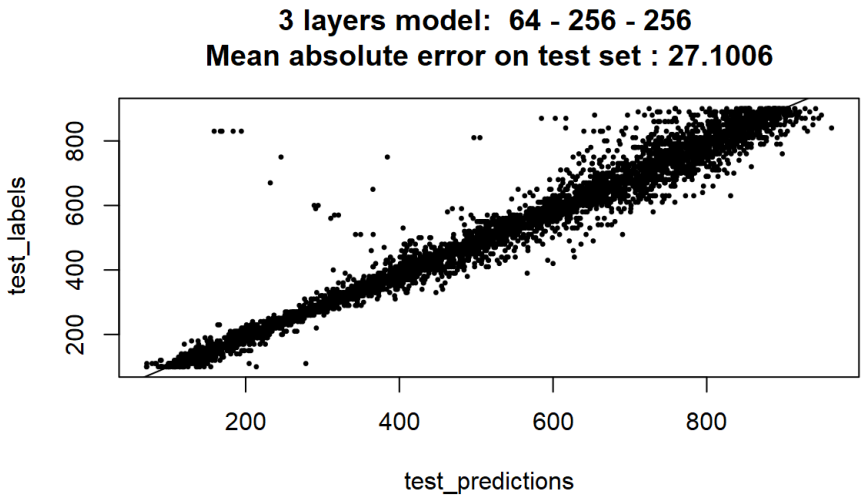


FIGURE 22 – Nuage de points : prédictions x valeurs réelles (2eme dataset modèle moyen)

On a un graphique très similaire avec le précédent. On remarque peu de différences. On peut tout de même noter qu'il y a moins de points extrêmes. La variance autour de la droite doit cependant être plus élevée que le précédent modèle.

Sur 100 modèles on a obtenu une MAE moyenne de 27.71. Il y a bien eu une nette amélioration avec ce jeu de données.

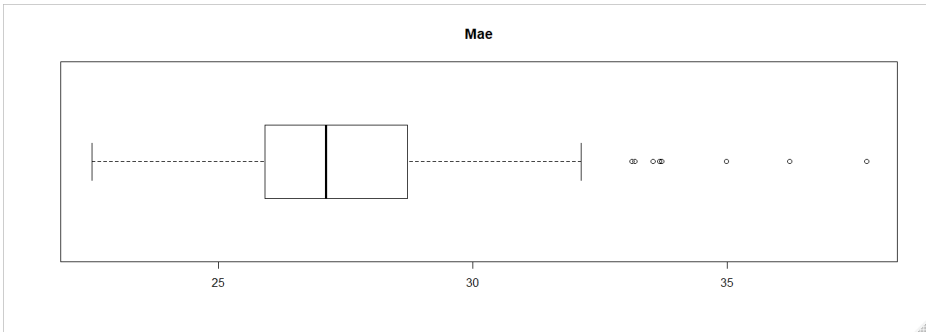


FIGURE 23 – Boxplot : MAE des 100 modèles créés (2eme dataset)

3.3.3 3eme Jeu de données

- Meilleur modèle :

Ce modèle a obtenu le meilleur score sur les 400 modèles. Il a donc obtenu un meilleur score que le 4eme jeu de données. Il a obtenu une MAE de 6.95. Il y a encore une fois un gros gap avec les données précédentes.

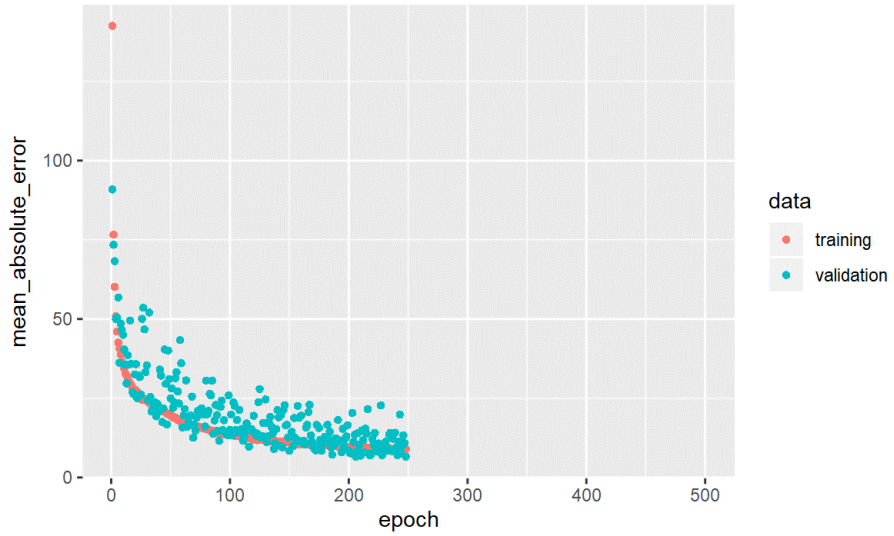


FIGURE 24 – Évolution de la MAE au fil des itérations (3eme dataset meilleur modèle)

Ce modèle, contrairement aux précédents meilleurs modèles, n'a itéré que 250 fois. Ce qui est dans la moyenne. On peut remarquer que la MAE pour le jeu de test est parfois sous la courbe de celle pour le jeu d'entraînement. Ce modèle est bizarrement meilleur sur le jeu d'entraînement. C'est la raison pour laquelle sa MAE est si basse. Cependant le modèle n'est pas très stable et aura sans doute du mal avec de nouveaux jeux de données.

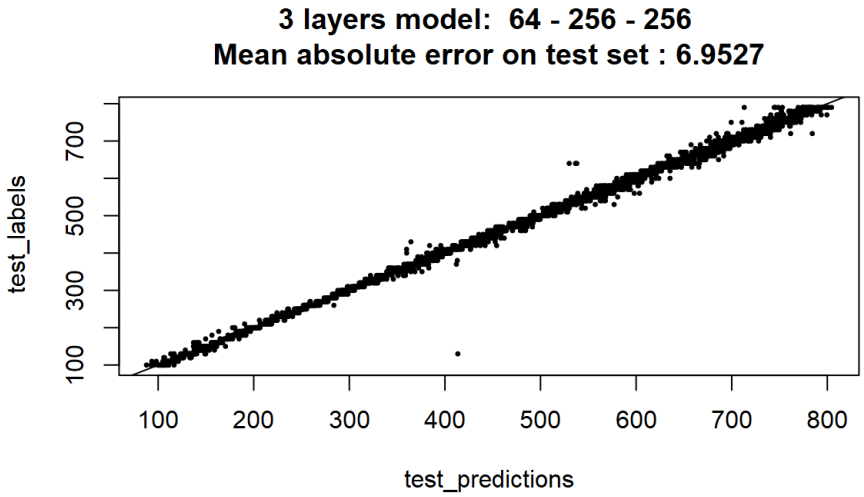


FIGURE 25 – Nuage de points : prédictions x valeurs réelles (3eme dataset meilleur modèle)

Mis à part les quelques points excentrés, les prédictions sont très précises. Les points sont très proches de la diagonale. Que ce soit à haut ou à bas Reynolds.

- Modèle moyen :

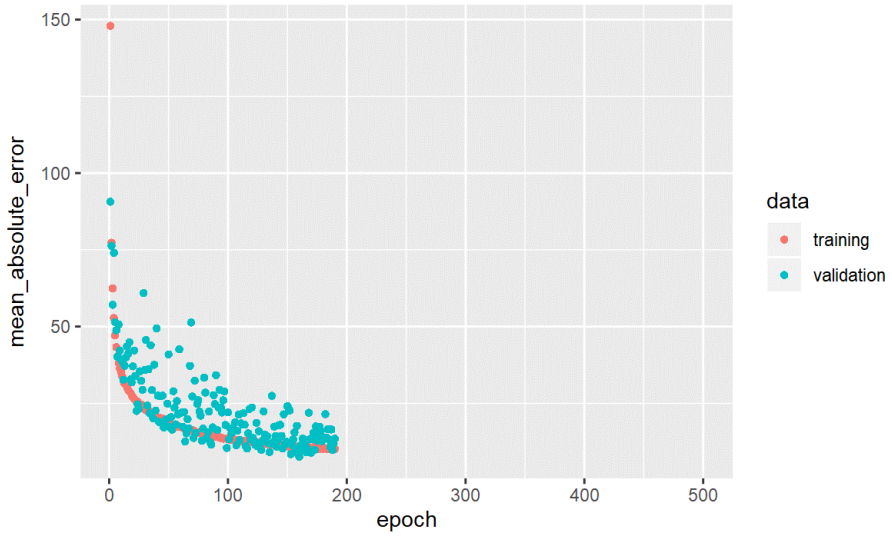


FIGURE 26 – Évolution de la MAE au fil des itérations (3eme dataset modèle moyen)

Le modèle moyen a effectué moins de 200 itérations. C'est assez peu. Il s'est stabilisé assez vite. Il a obtenu une MAE de 13.182, ce qui est encore une fois bien mieux que ce qu'on pouvait avoir avec les anciens jeux de données. On remarque que comme pour le modèle précédent, certains points de validations se retrouvent sous la courbe de la MAE obtenue avec le jeu d'entraînement. Cependant le modèle ne s'est pas arrêt au moment ou le modèle était meilleur avec le jeu de test.

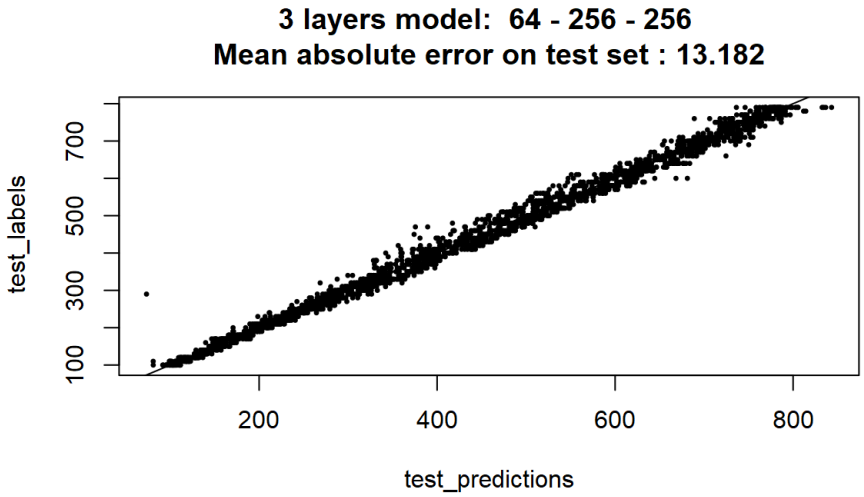


FIGURE 27 – Nuage de points : prédictions x valeurs réelles (3eme dataset modèle moyen)

On peut observer un très bon modèle, très bien centré et régulier. On note seulement un point isolé. Le reste est bien réparti autour de la diagonale. On peut évidemment remarquer que la variance autour de la droite est supérieure au modèle précédent. Néanmoins ce modèle reste très bon.

Sur les 100 modèles calculés avec le 3eme jeu de données, on obtient une MAE moyenne de 15.3257.

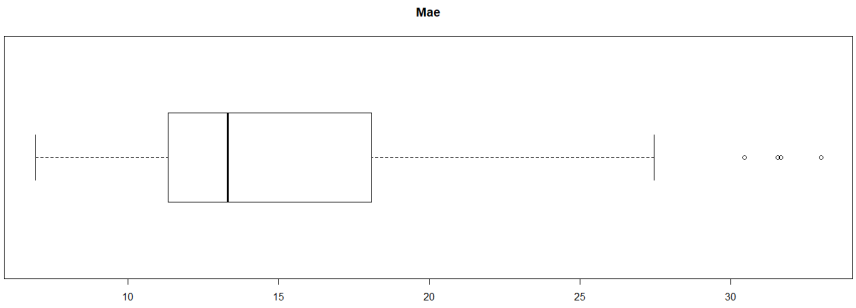


FIGURE 28 – Boxplot : MAE des 100 modèles créés (3eme dataset)

On remarque que la MAE n'est pas uniformément répartie.

3.3.4 4eme Jeu de données

- Meilleur modèle :

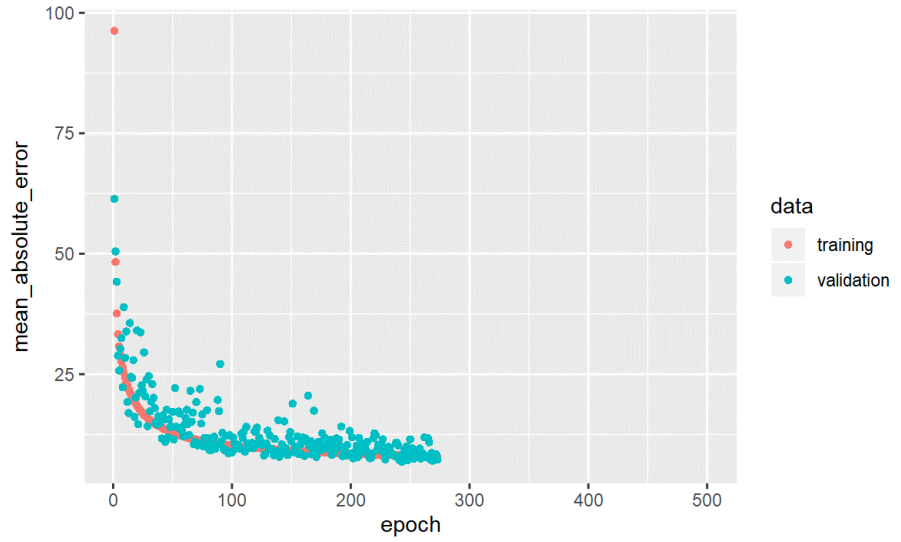


FIGURE 29 – Évolution de la MAE au fil des itérations (4eme dataset meilleur modèle)

On a encore une fois un modèle qui a de meilleurs résultats sur le jeu de test. La MAE n'est pas aussi faible que le meilleur modèle du set de données précédent. Elle est de 7.26. Le modèle est plus stable cependant.

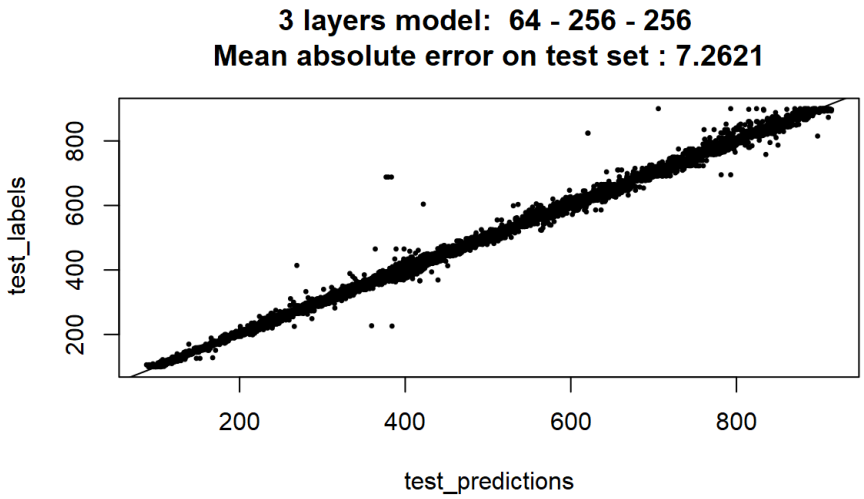


FIGURE 30 – Nuage de points : prédictions x valeurs réelles (4eme dataset meilleur modèle)

Mis à part les quelques points isolés, les prédictions sont très proches du vrai nombre de Reynolds, on peut voir la forte concentration autour de la droite.

- Modèle moyen :

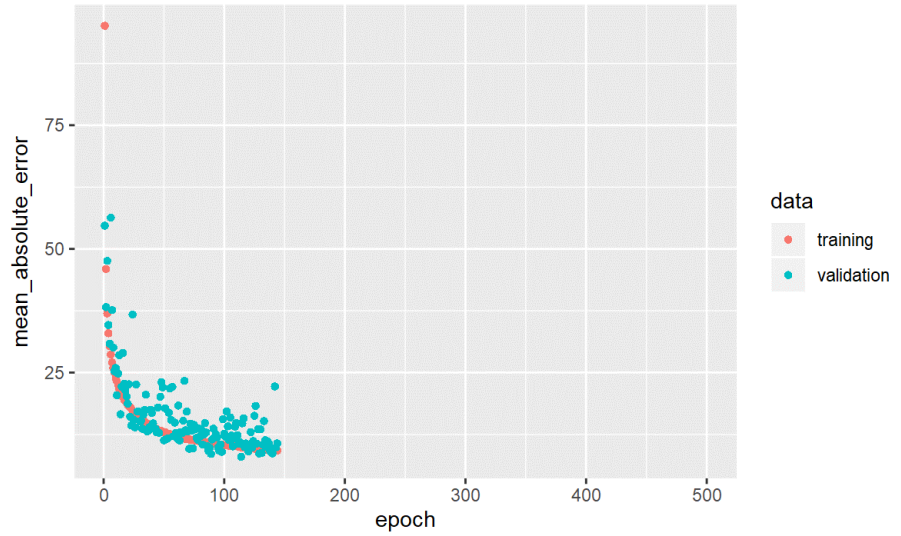


FIGURE 31 – Évolution de la MAE au fil des itérations (4eme dataset modèle moyen)

On a encore une fois quelques itérations pour lesquelles la MAE est plus faible pour le jeu de test. Ce modèle est cependant peu stable, on remarque notamment juste avant la fin un point de validation bien plus haut que les autres.

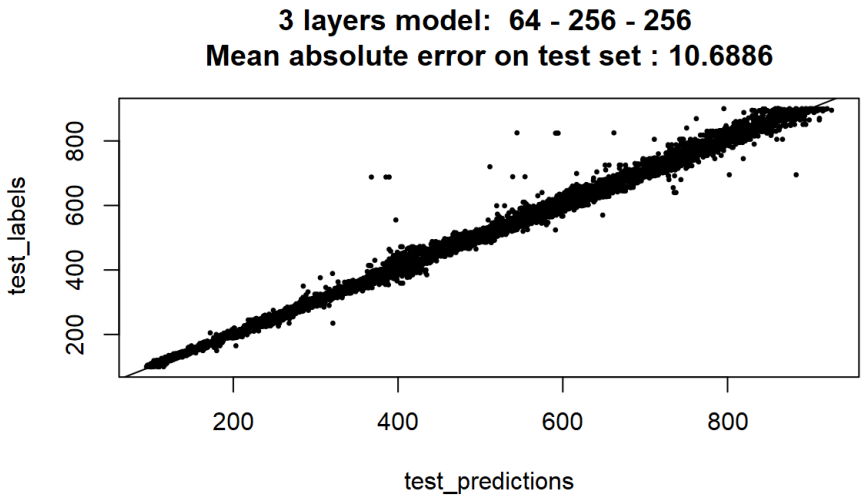


FIGURE 32 – Nuage de points : prédictions x valeurs réelles (4eme dataset modèle moyen)

La variance autour de la droite est légèrement supérieure à celle du modèle précédent. Les points isolés sont souvent des points sous-estimés.

Sur les 100 modèles calculés sur le 4eme jeu de données, on a une MAE moyenne de 11.10785. Ce jeu de donnée n'a pas pu produire le réseau de neurones avec le meilleur score, mais on a tout de même de meilleurs résultats en moyenne.

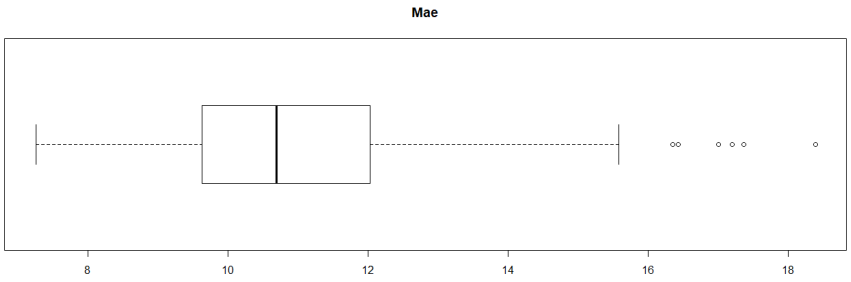


FIGURE 33 – Boxplot : MAE des 100 modèles créés (4eme dataset)

Conclusion

Grâce aux calculs de nos différents indicateurs représentatifs de la morphologie d'un jet, issu d'une simulation numérique. Nous avons pu créer un modèle linéaire optimal qui n'a pas nécessité de suppression de variable. Aussi nous avons pu évaluer la mae afin de se rendre compte de la qualité de notre modèle, nous avons ainsi obtenu pour la prédiction du nombre de Reynolds une mae moyenne de 41. On a pu constater que les observations à hautes amplitudes étaient les moins bien prédites. Par ailleurs grâce à l'utilisation de notre réseau de neurone nous sommes arrivé à prédire de manière beaucoup plus précise le nombre de Reynolds d'un fluide, à 11 près en moyenne pour un modèle moyen et à 7 près en moyenne pour le meilleur modèle. On a pu donc faire un meilleur usage des mesures disponibles qu'avec la méthode de régression linéaire multiple.

Il serait intéressant pour la suite du projet d'analyser plus en détails le modèle afin de se rendre compte de sa performance selon les données fournit en entrée pour une prédiction :
- L'influence du nombre de photos/interfaces sur l'estimation du nombre de Reynolds en sachant que, les photos sont identiques à une période près(comme la stimulation est sinusoïdale).
- L'influence du nombre d'amplitudes différentes pour un même jet en entrée sur la prédiction.

En vue d'une amélioration du modèle et pour le rendre plus polyvalent il faudrait créer un nouveau modèle prenant en compte le type de buse utilisé lors de la simulation du jet.

Annexes

3.4 Fonctions indicateurs

```
gouttes_info <- function(interface){
  # On met dans l'ordre les coordonnées de l'interface
  interface <- interface[order(interface[,1]),]
  # Nombre de points qui séparent deux gouttes
  n_points <- 150
  # Coordonnées du haut des gouttes
  coord <- list("x"=vector(), "y"=vector())
  # Nombre de gouttes
  n_gouttes <- 0
  # Coordonnées des cesures
  cesures <- vector()
  # Y a t'il un satellite ? Oui : 1 - Non : 0
  satellite <- 0
  # Longueur d'onde : distance entre 2 gouttes avant que le jet ne se brise
  long_onde <- NA

  # Polynome : 5 coeff d'un polynome de degré 4
  polynome <- rep(NA, 5)

  # Volume : volume de tout le jet
  volume <- 0

  # Surface : surface de tout le jet
  surface <- 0


  # Variable = TRUE si on a déjà détecté une cesure entre les 2 gouttes
  deja <- FALSE
  if(min(interface[,2])>=0 && max(interface[,2])<=3.5){
    for(i in 1:nrow(interface)){

      debut <- max(1, i-n_points)
      fin <- min(nrow(interface), i+n_points)


      # Vérifie si on est entre 2 gouttes et non pas au début de l'interface
      # Normalement cette condition ne sert à rien mais on ne sait jamais
      if(n_gouttes>0){
        if((interface[min(nrow(interface),i+1),1]-interface[i,1])>0.3){
          if(deja == TRUE){
            satellite <- 1
          }
          cesures <- c(cesures, interface[i,1])
          if(length(cesures)==1){
            long_onde <- coord$x[length(coord$x)]-coord$x[length(coord$x)-1]
          }
          deja <- TRUE
        }
      }

      # Vérifie que le point étudié est bien le sommet d'une goutte
      if(sum(interface[i,2]<interface[debut:i,2])==0 &&
      sum(interface[i,2]<interface[(i):fin,2])==0){
        if(length(coord$x)==0){
          coord$x <- c(coord$x, interface[i,1])
          coord$y <- c(coord$y, interface[i,2])
```

```
          n_gouttes <-  n_gouttes+1
          deja <- FALSE
        }else{
          if(coord$x[length(coord$x)]!=interface[i,1] &&
          coord$y[length(coord$y)]!=interface[i,2]){
            coord$x <- c(coord$x, interface[i,1])
            coord$y <- c(coord$y, interface[i,2])
            n_gouttes <-  n_gouttes+1
            deja <- FALSE
          }
        }
      }


      # Calcule le volume et la surface du jet

      if(i>1){
        x <- interface[(i - 1):i, 1]
        y <- interface[(i - 1):i, 2]
        if(abs(y[1]-y[2])<0.1){
          # volume + (rectangle + triangle) * rayon^2*pi
          volume <- volume + ((x[2]-x[1])*min(y) +
          ((x[2]-x[1])*(max(y)-min(y)))/2)*mean(c(min(y), max(y)))^2*pi
          # surface + (moyenne de la surface pour le y le plus haut et le
          plus faible)*2pi
          surface <- surface + mean((x[2]-x[1])*min(y),
          (x[2]-x[1])*max(y))*2*pi
        }
      }
    }


    polynome <- lm(coord$y~poly(coord$x, degree =4,
    raw=TRUE))$coefficients
    if(is.na(long_onde))
      long_onde <- coord$x[length(coord$x)]-coord$x[max(2,length(coord$x))
      -1]
  }

  return(
    list(
      "coord" = coord,
      "polynome" = polynome,
      "cesures" = cesures,
      "satellite" = satellite,
      "long_onde" = long_onde,
      "volume" = volume,
      "surface" = surface
    )
  )
}




dist_indiv <- function(d, i, n, n_voisins = 0){
  if(n_voisins==0)
    n_voisins <- n
  j_vect <- c(0,cumsum((n-1):1))
  jb_vect <- c(0,cumsum(1:(n-1)))
```

```
  res <- c(0)
  if(i<n){
    j <- (j_vect[i]+1):j_vect[min(i+1,length(j_vect))]
    res <- c(res,d[j[1:min(n_voisins, length(j))]])
  }
  if(i>1){
    jb <- (j_vect[1:(i-1)])
    jb <- jb + (i-1):1
    res <- c(d[jb[max(1,i-n_voisins):length(jb)]], res)
  }
  res
}



filtre_gouttes_info <- function(interface){
  interface <- interface[order(interface[,1]),]
  d <- dist(interface)
  j_vect <- which(interface[,2]>quantile(interface[,2], 0.90))
  suppr <- vector()
  for(j in j_vect){
    d_i <- dist_indiv(d, j, nrow(interface), 20)
    if(mean(d_i)>0.60){
      suppr <- c(suppr, j)
    }
  }
  if(length(suppr)>0)
    interface <- interface[-suppr,]


  gouttes <- gouttes_info(interface)
  gouttes$suppr <- suppr
  return(gouttes)
}
```

3.5 Création du jeu de données

```
rep <- "../flat_240_350_gnu"

files <- list.files(rep)




init <- rep(NA, length(files))
tableau <- data.frame("Reynolds" = as.integer(substr(files, 1,3)),
                      "Amplitude" = as.numeric(substr(files, 5,8)),
                      "Temps" = as.numeric(substr(files, 10,14)),
                      "Premiere_cesure" = init,
                      "nb_cesures" = init,
                      "volume" = init,
                      "surface" = init,
                      "ratio_vs" = init,
                      "satellite" = init,
                      "long_onde" = init,
                      "poly4" = init,
                      "poly3" = init,
                      "poly2" = init,
                      "poly1" = init,
                      "poly0" = init)
```

```
i <- 1L
temps_cum <- 0
i_prec <- i
for(i in i:length(files)){
  temps_avant <- Sys.time()
  interface <- read.table(paste0(rep,"/",files[i]))
  interface <- interface[order(interface[,1]),1:2]
  gouttes <- filtre_gouttes_info(interface)
  tableau$Premiere_cesure[i] <- if(is.na(gouttes$cesures[1])) interface[nrow(interface),1]
  else gouttes$cesures[1]
  tableau$nb_cesures[i] <- length(gouttes$cesures)
  tableau$volume[i] <- gouttes$volume
  tableau$surface[i] <- gouttes$surface
  tableau$ratio_vs[i] <- gouttes$surface/gouttes$volume
  tableau$satellite[i] <- gouttes$satellite
  tableau$long_onde[i] <- gouttes$long_onde
  tableau$poly0[i] <- as.numeric(gouttes$polynome[1])
  tableau$poly1[i] <- as.numeric(gouttes$polynome[2])
  tableau$poly2[i] <- as.numeric(gouttes$polynome[3])
  tableau$poly3[i] <- as.numeric(gouttes$polynome[4])
  tableau$poly4[i] <- as.numeric(gouttes$polynome[5])
  # plot(interface, cex = 0.2, asp = 3, main = files[i], ylim = c(-3.5,3.5))
  # points(interface[,1], -interface[,2], cex=0.2)
  if(loading<trunc((i*100)/length(files))){
    write.csv2(tableau, "tableau_final.csv", row.names = FALSE)
    print("Sauvé")
  }
  temps_cum <- difftime(Sys.time(), temps_avant, units = "hours") + temps_cum
  loading <- (i*100)/length(files)
  reste <- temps_cum/(((i-i_prec+1)*100)/length(files))*(100-loading)
  cat(paste0("\r",sprintf('%.3f',loading),"%   -   ",
             trunc(reste), "h"), round((reste-trunc(reste))*60), "min -   ",
             i-nrow(na.omit(tableau)), "NA's")
}
```

3.6 Modèle linéaire

1er Modèle :

```
```{r}
dataR<-read.csv2("C:/Users/Lisa/Documents/M1_SSD/Projet_tut/Rheology_Project/REG1/
tableau1.csv", header = TRUE)
dataR <- na.omit(dataR)
summary(dataR)

set.seed(22071997)
idTrain <-sample(1:nrow(dataR),round(nrow(dataR)*0.8),replace=F)

Train <-dataR[idTrain,]

Test <-dataR[-idTrain,]
```
```{r fig.cap="Modèle linéaire", echo=FALSE}
R.lm <- lm(Reynolds.,Train)
R.lm <- step(R.lm,direction="both")
summary(R.lm)
```
```{r echo=FALSE, fig.width=12}
Ypred <- predict(R.lm,Test[,-1])
Ytest <- Test$Reynolds
```
```

```
plot(Ypred,Ytest,xlab="Prediction",ylab="Observation")
title(main="Prédiction Reynolds jeu de test :\n Amplitude de 0.01 à 0.15 /
Reynolds de 100 à 900", line = 0.19)
abline(0,1,col="blue")
```

```{r echo=FALSE}
(MAE <-mean(abs(Ypred-Ytest)))
```

```{r echo=FALSE, fig.width=12}
dataOut <- data.frame(rep(NA,length(Ytest)),(rep(NA,length(Ytest))),rep(NA,
length(Ytest)))
colnames(dataOut) <- c("Index","Ytest","Ypred")
for (i in 1:length(Ytest)) {
  if (Ytest[i] > 0 && Ypred[i] <800) {
    if (Ytest[i]>(Ypred[i]+MAE)){
      dataOut[i,1] <- names(Ypred)[i]
      dataOut[i,2] <- Ytest[i]
      dataOut[i,3] <- Ypred[i]
    }
  }

    if (Ytest[i]<(Ypred[i]-MAE)){
      dataOut[i,1] <- names(Ypred)[i]
      dataOut[i,2] <- Ytest[i]
      dataOut[i,3] <- Ypred[i]

  }
}
dataOut <- na.omit(dataOut)

par(mfrow=c(1,2))
couleur <- c(rep("darkolivegreen2",length(Ytest)))
couleur[as.numeric(row.names(dataOut))] <- "grey"
plot(Ypred,Ytest,col=couleur,xlab="Prediction",ylab="Observation")
title(main="Prédiction Reynolds jeu de test  :\n Amplitude de 0.01 à 0.15 /
Reynolds de 100 à 900", line=0.19)
lines((Ytest-MAE),Ytest, lty = 5, col = "red",type="l")
lines((Ytest+MAE),Ytest, lty = 5, col = "red",type="l")
legend("bottomright", legend="intervalle MAE",col="red", lty=2, cex=0.8)
abline(0,1,col="black")



couleur <- Test$Amplitude*100

plot(Ypred,Ytest,xlab="Prediction",ylab="Observation",col=rainbow(15)
[Test$Amplitude*100])
title(main="Prédiction Reynolds jeu de test  :\n Amplitude de 0.01 à 0.15
/ Reynolds de 100 à 900", line=0.19)
legend("bottomright",c(paste0("Amplitude=0.0",1:9),
paste0("Amplitude=0.",10:15)),fill=rainbow(15),cex = 0.6)
abline(0,1,col="black")
```
```{r echo=FALSE}
Ypred <- predict(R.lm,Test[,-1])
Ytest <- Test$Reynolds

g <- 22071997
```

```
E1 <- rep(0,20)

for (i in 1:20) {
  set.seed(g)
  g <- g+10

idTrain <-sample(1:nrow(dataR),round(nrow(dataR)*0.8),replace=F)

Train <-dataR[idTrain,]

Test <-dataR[-idTrain,]

R.lm <- lm(Reynolds.,Train)
R.lm <- step(R.lm,direction="both")


Ypred <- predict(R.lm,Test[,-1])
Ytest <- Test$Reynolds

E1[i] <- mean(abs(Ypred-Ytest))

}
(MAE <- (mean(E1)))
```

- 2ème Modèle :

```
```{r echo=FALSE}
dataR<-read.csv2("C:/Users/Lisa/Documents/M1_SSD/Projet_tut/Rheology_Project/
REG2/tableau2.csv", header = TRUE)
dataR <- na.omit(dataR)
summary(dataR)

set.seed(22071997)
idTrain <-sample(1:nrow(dataR),round(nrow(dataR)*0.8),replace=F)

Train <-dataR[idTrain,]

Test <-dataR[-idTrain,]
```
```{r fig.cap="Modèle linéaire", echo=FALSE}
R.lm <- lm(Reynolds.,Train)
R.lm <- step(R.lm,direction="both")
summary(R.lm)
```

```{r echo=FALSE, fig.width=12}
Ypred <- predict(R.lm,Test[,-1])
Ytest <- Test$Reynolds
plot(Ypred,Ytest,xlab="Prediction",ylab="Observation")
title(main="Prédiction Reynolds jeu de test :\n Amplitude de 0.01 à 0.15 /
Reynolds de 100 à 900", line = 0.19)
abline(0,1,col="blue")
```
```{r}
(MAE <-mean(abs(Ypred-Ytest)))
```
```{r echo=FALSE, fig.width=12}
dataOut <- data.frame(rep(NA,length(Ytest)),(rep(NA,length(Ytest))),
rep(NA,length(Ytest)))
colnames(dataOut) <- c("Index","Ytest","Ypred")
```
```

```
for (i in 1:length(Ytest)) {
  if (Ytest[i] > 0 && Ypred[i] <800) {
    if (Ytest[i]>(Ypred[i]+MAE)){
      dataOut[i,1] <- names(Ypred)[i]
      dataOut[i,2] <- Ytest[i]
      dataOut[i,3] <- Ypred[i]
    }
  }

    if (Ytest[i]<(Ypred[i]-MAE)){
      dataOut[i,1] <- names(Ypred)[i]
      dataOut[i,2] <- Ytest[i]
      dataOut[i,3] <- Ypred[i]

  }
}
dataOut <- na.omit(dataOut)

par(mfrow=c(1,2))
couleur <- c(rep("darkolivegreen2",length(Ytest)))
couleur[as.numeric(row.names(dataOut))] <- "grey"
plot(Ypred,Ytest,col=couleur,xlab="Prediction",ylab="Observation")
title(main="Prédiction Reynolds jeu de test  :\n Amplitude de 0.01 à 0.15 /
Reynolds de 100 à 900", line=0.19)
lines((Ytest-MAE),Ytest, lty = 5, col = "red",type="l")
lines((Ytest+MAE),Ytest, lty = 5, col = "red",type="l")
legend("bottomright", legend="intervalle MAE",col="red", lty=2, cex=0.8)
abline(0,1,col="black")

couleur <- Test$Amplitude*100

plot(Ypred,Ytest,xlab="Prediction",ylab="Observation",col=rainbow(15)
[Test$Amplitude*100])
title(main="Prédiction Reynolds jeu de test  :\n Amplitude de 0.01 à 0.15
/ Reynolds de 100 à 900", line=0.19)
legend("bottomright",c(paste0("Amplitude=0.0",1:9),
paste0("Amplitude=0.",10:15)),fill=rainbow(15),cex = 0.6)
abline(0,1,col="black")
```
```{r echo=FALSE}
Ypred <- predict(R.lm,Test[,-1])
Ytest <- Test$Reynolds

g <- 22071997
E1 <- rep(0,20)

for (i in 1:20) {
  set.seed(g)
  g <- g+10

idTrain <-sample(1:nrow(dataR),round(nrow(dataR)*0.8),replace=F)

Train <-dataR[idTrain,]

Test <-dataR[-idTrain,]

R.lm <- lm(Reynolds.,Train)
R.lm <- step(R.lm,direction="both")
```

```
Ypred <- predict(R.lm,Test[,-1])
Ytest <- Test$Reynolds

E1[i] <- mean(abs(Ypred-Ytest))

}
(MAE <- (mean(E1)))
```
\end{min}

- 3ème Modèle :\newline
\begin{minted}{r}
```{r echo=FALSE}
dataR<-read.csv2("C:/Users/Lisa/Documents/M1_SSD/Projet_tut/Rheology_Project/
REG3/tableau3.csv", header = TRUE)
dataR <- na.omit(dataR)


set.seed(22071997)
idTrain <-sample(1:nrow(dataR),round(nrow(dataR)*0.8),replace=F)

Train <-dataR[idTrain,]

Test <-dataR[-idTrain,]
```
```{r fig.cap="Modèle linéaire", echo=FALSE}
R.lm <- lm(Reynolds.,Train)
R.lm <- step(R.lm,direction="both")
summary(R.lm)
```
```{r fig.width=12}
Ypred <- predict(R.lm,Test[,-1])
Ytest <- Test$Reynolds
plot(Ypred,Ytest,xlab="Prediction",ylab="Observation")
title(main="Prédiction Reynolds jeu de test :\n Amplitude de 0.01 à 0.15 /
Reynolds de 100 à 790", line = 0.19)
abline(0,1,col="blue")
```
```{r}
(MAE <-mean(abs(Ypred-Ytest)))
```

```{r echo=FALSE, fig.width=12}
dataOut <- data.frame(rep(NA,length(Ytest)),(rep(NA,length(Ytest))),
rep(NA,length(Ytest)))
colnames(dataOut) <- c("Index","Ytest","Ypred")
for (i in 1:length(Ytest)) {
  if (Ytest[i] > 0 && Ypred[i] <800) {
    if (Ytest[i]>(Ypred[i]+MAE)){
      dataOut[i,1] <- names(Ypred)[i]
      dataOut[i,2] <- Ytest[i]
      dataOut[i,3] <- Ypred[i]
    }
  }

    if (Ytest[i]<(Ypred[i]-MAE)){
      dataOut[i,1] <- names(Ypred)[i]
      dataOut[i,2] <- Ytest[i]
      dataOut[i,3] <- Ypred[i]


```

```
  }
}
dataOut <- na.omit(dataOut)

par(mfrow=c(1,2))
couleur <- c(rep("darkolivegreen2",length(Ytest)))
couleur[as.numeric(row.names(dataOut))] <- "grey"
plot(Ypred,Ytest,col=couleur,xlab="Prediction",ylab="Observation")
title(main="Prédiction Reynolds jeu de test  :\n Amplitude de 0.01 à 0.15
/ Reynolds de 100 à 790", line=0.19)
lines((Ytest-MAE),Ytest, lty = 5, col = "red",type="l")
lines((Ytest+MAE),Ytest, lty = 5, col = "red",type="l")
legend("bottomright", legend="intervalle MAE",col="red", lty=2, cex=0.8)
abline(0,1,col="black")

couleur <- Test$Amplitude*100

plot(Ypred,Ytest,xlab="Prediction",ylab="Observation",col=rainbow(15)
[Test$Amplitude*100])
title(main="Prédiction Reynolds jeu de test  :\n Amplitude de 0.01 à 0.15
/ Reynolds de 100 à 790", line=0.19)
legend("bottomright",c(paste0("Amplitude =0.0",1:9),
paste0("Amplitude =0.",10:15)),fill=rainbow(15),cex = 0.6)
abline(0,1,col="black")
```
```{r echo=FALSE}
Ypred <- predict(R.lm,Test[,-1])
Ytest <- Test$Reynolds

g <- 22071997
E1 <- rep(0,20)

for (i in 1:20) {
  set.seed(g)
  g <- g+10

idTrain <-sample(1:nrow(dataR),round(nrow(dataR)*0.8),replace=F)

Train <-dataR[idTrain,]

Test <-dataR[-idTrain,]

R.lm <- lm(Reynolds.,Train)
R.lm <- step(R.lm,direction="both")

Ypred <- predict(R.lm,Test[,-1])
Ytest <- Test$Reynolds

E1[i] <- mean(abs(Ypred-Ytest))

}
(MAE <- (mean(E1)))
```

- 4ème Modèle :

```{r echo=FALSE}
dataR<-read.csv2("C:/Users/Lisa/Documents/M1_SSD/Projet_tut/Rheology_Project
/REG4/tableau4.csv", header = TRUE)
dataR <- na.omit(dataR)
```

```
set.seed(22071997)
idTrain <-sample(1:nrow(dataR),round(nrow(dataR)*0.8),replace=F)

Train <-dataR[idTrain,]

Test <-dataR[-idTrain,]
```
```{r fig.cap="Résumé des données", echo=FALSE}
summary(dataR)
```
```{r fig.cap="Modèle linéaire", echo=FALSE}
R.lm <- lm(Reynolds.,Train)
R.lm <- step(R.lm,direction="both")
summary(R.lm)
```

```{r fig.width=12}
Ypred <- predict(R.lm,Test[,-1])
Ytest <- Test$Reynolds
plot(Ypred,Ytest,xlab="Prediction",ylab="Observation")
title(main="Prédiction Reynolds jeu de test :\n Amplitude de 0.01 à 0.1
/ Reynolds de 100 à 900", line = 0.19)
abline(0,1,col="blue")
```
```{r}
(MAE<-mean(abs(Ypred-Ytest)))
```
```{r fig.width=12}
dataOut <- data.frame(rep(NA,length(Ytest)),(rep(NA,length(Ytest))),
rep(NA,length(Ytest)))
colnames(dataOut) <- c("Index","Ytest","Ypred")
for (i in 1:length(Ytest)) {
  if (Ytest[i] > 0 ) {
    if (Ytest[i]>(Ypred[i]+MAE)){
      dataOut[i,1] <- names(Ypred)[i]
      dataOut[i,2] <- Ytest[i]
      dataOut[i,3] <- Ypred[i]
    }
  }

    if (Ytest[i]<(Ypred[i]-MAE)){
      dataOut[i,1] <- names(Ypred)[i]
      dataOut[i,2] <- Ytest[i]
      dataOut[i,3] <- Ypred[i]

  }
}
dataOut <- na.omit(dataOut)

par(mfrow=c(1,2))
couleur <- c(rep("darkolivegreen2",length(Ytest)))
couleur[as.numeric(row.names(dataOut))] <- "grey"
plot(Ypred,Ytest,col=couleur,xlab="Prediction",ylab="Observation")
title(main="Prédiction Reynolds jeu de test  :\n Amplitude de 0.01 à 0.1 /
Reynolds de 100 à 900", line=0.19)
lines((Ytest-MAE),Ytest, lty = 5, col = "red",type="l")
lines((Ytest+MAE),Ytest, lty = 5, col = "red",type="l")
legend("bottomright", legend="intervalle MAE",col="red", lty=2, cex=0.8)
abline(0,1,col="black")
```

```
couleur <- Test$Amplitude*100

plot(Ypred,Ytest,xlab="Prediction",ylab="Observation",col=rainbow(15)
[Test$Amplitude*100])
title(main="Prédiction Reynolds jeu de test  :\n Amplitude de 0.01 à 0.1 /
Reynolds de 100 à 900", line=0.19)
legend("bottomright",c(paste0("Amplitude =0.0",1:9),
paste0("Amplitude =0.",1)),fill=rainbow(15),cex = 0.6)
abline(0,1,col="black")
```
```{r echo=FALSE}
Ypred <- predict(R.lm,Test[,-1])
Ytest <- Test$Reynolds

g <- 22071997
E1 <- rep(0,20)

for (i in 1:20) {
  set.seed(g)
  g <- g+10

idTrain <-sample(1:nrow(dataR),round(nrow(dataR)*0.8),replace=F)

Train <-dataR[idTrain,]

Test <-dataR[-idTrain,]

R.lm <- lm(Reynolds.,Train)
R.lm <- step(R.lm,direction="both")

Ypred <- predict(R.lm,Test[,-1])
Ytest <- Test$Reynolds

E1[i] <- mean(abs(Ypred-Ytest))

}
(MAE <- (mean(E1)))
```

3.7 Réseau de neurones

```
library(keras)
# library(tfruns)
# install_keras()


# Hyperparameter flags ---------------------------------------------------

# Parametres modifiables

# Il y a le nombre de neurones par couche ainsi que la seed

FLAGS <- flags(
  flag_string("data", "tableau_final.csv"),
  flag_numeric("dense1", 64),
  flag_numeric("dense2", 256),
  flag_numeric("dense3", 256),
  flag_numeric("seed", 22071997)
)
```

```r
# Data Preparation -----------------------------------------------------

set.seed(FLAGS$seed)

data <- read.csv2(FLAGS$data)[,1:15]
# data <- read.csv2("tableau_final.csv")
# data <- read.csv2("tableau_pred_reynolds.csv")
# data <- read.csv2("tableau_pred_amplitude.csv")

# On supprime les valeurs manquantes dans le tableau


data <- na.omit(data)

# The data, shuffled and split between train and test sets

labels <- data[,1]
data <- data[,-1]
train_i <- sample(nrow(data), nrow(data)*80/100)

row.names(data) <- 1:nrow(data)

test_i <- as.numeric(row.names(data[-train_i,]))

train_data <- data[train_i,]
train_labels <- labels[train_i]
test_data <- data[test_i,]
test_labels <- labels[test_i]


row.names(train_data) <- 1:length(train_data[,1])
row.names(test_data) <- 1:length(test_data[,1])




# Test data is *not* used when calculating the mean and std.

# Normalize training data
train_data <- scale(train_data)

# Use means and standard deviations from training set to normalize test set
col_means_train <- attr(train_data, "scaled:center")
col_stddevs_train <- attr(train_data, "scaled:scale")
test_data <- scale(test_data, center = col_means_train, scale = col_stddevs_train)



# Display training progress by printing a single dot for each completed epoch.
print_dot_callback <- callback_lambda(
  on_epoch_end = function(epoch, logs) {
    cat("\repoch : ", epoch, "/", epochs, " (", sprintf("%.2f",epoch*100/epochs), "%)    ")
  }
)



# Define Model ---------------------------------------------------------------
```

```
epochs <- 500


build_model <- function() {
  model <- keras_model_sequential() %>%
    layer_dense(units = FLAGS$dense1, activation = "relu",
                input_shape = dim(train_data)[2]) %>%
    layer_dense(units = FLAGS$dense2, activation = "relu") %>%
    layer_dense(units = FLAGS$dense3, activation = "relu") %>%
    # layer_dense(units = 256, activation = "relu") %>%
    layer_dense(units = 1)

  model %>% compile(
    loss = "mse",
    optimizer = optimizer_rmsprop(),
    metrics = list("mean_absolute_error")
  )

  model
}


# The patience parameter is the amount of epochs to check for improvement.

early_stop <- callback_early_stopping(monitor = "val_loss", patience = 30)



# Building the model

model <- build_model()

# Training & Evaluation ------------------------------------------------------

history <- model %>% fit(
  train_data,
  train_labels,
  epochs = epochs,
  # view_metrics =TRUE,
  validation_split = 0.2,
  verbose = 0,
  callbacks = list(print_dot_callback, early_stop)
)

plot(history, metrics = "mean_absolute_error", smooth = FALSE)

score <- model %>% evaluate(
  test_data, test_labels,
  verbose = 0
)

cat('Test loss:', score$loss, '\n')
cat('Test accuracy:', score$mean_absolute_error, '\n')


test_predictions <- model %>% predict(test_data)
plot(test_predictions,test_labels,
main = paste("3 layers model: ",FLAGS$dense1,"-",
             FLAGS$dense2,"-",FLAGS$dense3,"\nMean absolute error on test set :",
```

```
            round(score$mean_absolute_error,4)), pch=20, cex=0.5)
abline(0,1)
```