

Implémentation d'approches optimales pour la création d'un ensemble d'apprentissage pour la calibration de modèles et application à la mécanique des fluides.

Nithushan Karunakaran^{a,*}

^a*Université Paris 8, Saint-Denis, 93200, France*

Keywords: Mot-clés: Calibration de modèles de mécanique de fluides, machine learning, simulation numérique, optimisation multiparamètres.

1. Introduction

A l'heure actuelle, les constructeurs de câbles aériens rencontrent des problèmes de vieillissement des câbles. Parmi les approches permettant de mieux gérer ces phénomènes et d'y remédier se trouve notamment le machine learning. Cependant, cette approche se heurte à un manque de données terrain. C'est cela qui a amené le développement d'un modèle dit modèle de référence qui a l'avantage de bien modéliser les câbles en question, mais qui a le gros désavantage d'être lourd dans les manipulations logicielles. Pour répondre à ce problème nous avons un modèle plus simplifié reprenant les caractéristiques du modèle de référence, mais plus simplifiés. Cependant, cette approche nécessite de nombreux ajustements manuels lors de l'utilisation du modèle simplifié.

La problématique de nos travaux a donc été de trouver une approche permettant d'utiliser le modèle simplifié sans passer par des ajustements manuels.

*Corresponding author

Email address: nithushankarunakaran@gmail.com (Nithushan Karunakaran)

¹⁴ Nous avons donc cherché à répondre à cette problématique dans le cadre de
¹⁵ nos travaux et de nos résultats.

¹⁶ **2. Méthodologie**

¹⁷ *2.1. Organisation du projet*

¹⁸ Dans le cadre de ce projet, nous disposons d'un simulateur (appelé Modèle
¹⁹ simplifié) permettant de générer des données de représentation des mouvements
²⁰ de conducteurs de lignes aériennes. Nous disposons d'un fichier YAML contenant
²¹ la liste des paramètres utilisées par le simulateur. Il y a également les séries tem-
²² porelles et leurs paramètres respectifs, séparées en 2 catégories, pour les modèles
²³ de références à travers 2 fichiers textes (List1.txt et List2.txt). Enfin, nous dis-
²⁴ posons de fichiers CSV contenant les données des séries temporelles de référence
²⁵ respectivement pour la Liste 1 et pour la Liste 2.

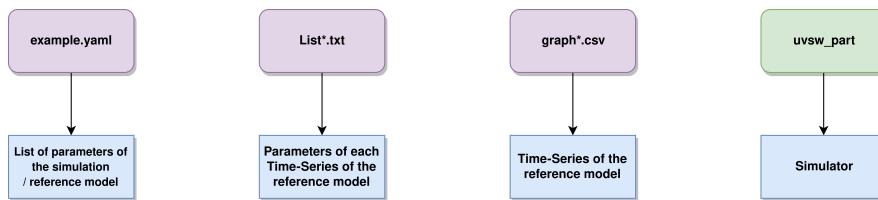


Figure 1: Main files

²⁶ Le projet est organisé en plusieurs dossiers. Le dossier data est le dossier
²⁷ contenant les données utilisées dans le projet. Il se compose d'un dossier config
²⁸ contenant le fichier YAML des paramètres utilisés par le simulateur. Le dossier in-
²⁹ termédiaire contient les données intermédiaires obtenues via des simulations pour
³⁰ ne pas avoir à reproduire l'exécution du simulateur pour récupérer des résultats
³¹ à chaque itération. Le dossier "params" contient quant à lui les données liées au

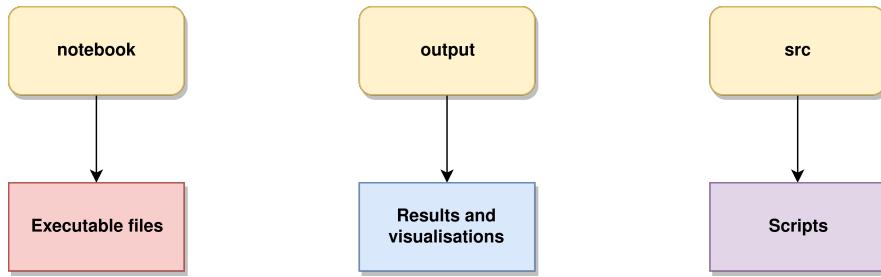


Figure 2: Folder organization

32 modèle de référence à savoir les fichiers List1.txt et List2.txt. Le dossier ref quant
 33 à lui contient les données de s séries temporelles de référence.
 34 Le dossier notebook est le dossier contenant les codes exécutables. Le dossier
 35 output contient les résultantes du projet notamment les visualisations. Enfin le
 36 dossier src contient l'ensemble des scripts utilisés dans le projet.

37 *2.2. Organisation du projet dans Github*

38 Le code du projet est disponible via Github : Code
 39 Le dépôt se compose de 5 branches :

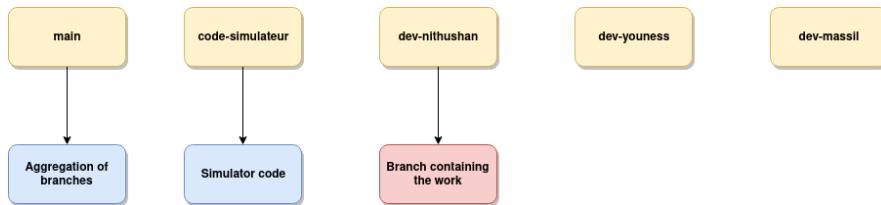


Figure 3: Branches

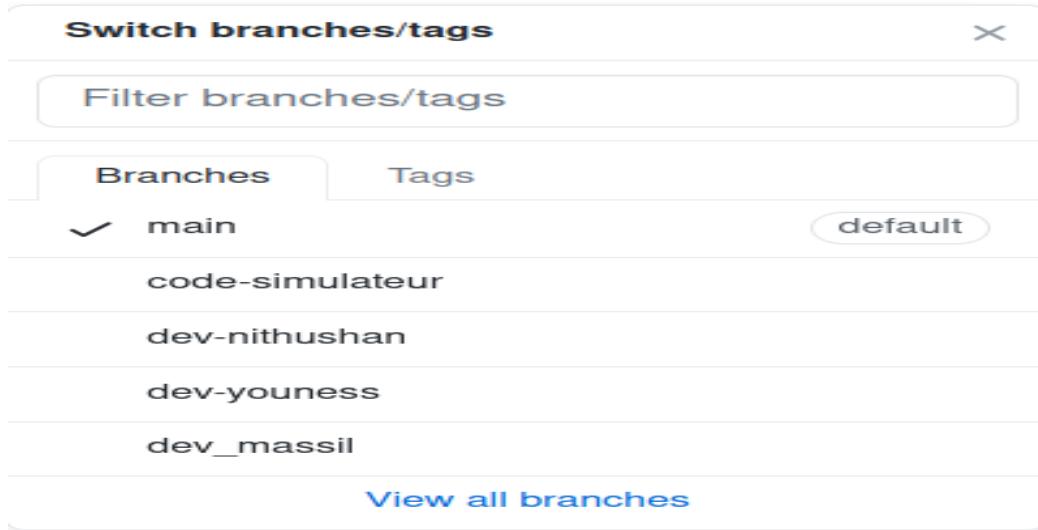


Figure 4: Project's branches

⁴⁰ *2.3. Simulateur*

⁴¹ Tout d'abord, un simulateur fourni par RTE permettant de générer un modèle
⁴² simplifié de modélisation des mouvements de conducteurs de lignes aériennes. Le
⁴³ simulateur prend en entrée une configuration de paramètres, ce qui lui permet de
⁴⁴ générer le modèle simplifié que nous exploitons.

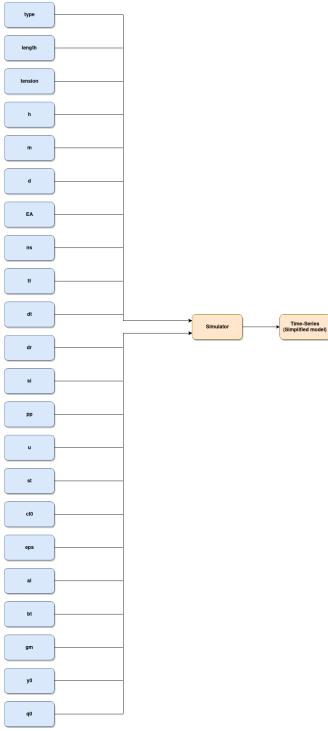


Figure 5: Parameters of the simulator

45 Le simulateur prend en entrée une configuration de paramètres et génère en
 46 sortie une série temporelle représentant le modèle simplifié. Des fichiers con-
 47 tenant des modèles de référence au format CSV ainsi qu'un fichier de configura-
 48 tion YAML contenant les paramètres utilisés ont également été fournis. Une fois
 49 ces fichiers mis à disposition, nous nous sommes mis à étudier le moyen de créer
 50 un ensemble d'apprentissage avec le simulateur.

51 Nous avons essayé de manière itérative des combinaisons de paramètres avec
 52 le simulateur afin de se rapprocher du modèle de référence. Cependant, cette ap-
 53 proche s'est révélée infructueuse dû au coût computationnel de chaque simulation.
 54 Cela nous a amené à nous orienter vers d'autres approches afin d'optimiser



Figure 6: Parameters and reference model

Table 1: Execution time for one simulator run

List index	Time-Series index	Execution time (seconds)
1	1	50
1	24	146
2	1	290

55 la recherche des combinaisons de paramètres optimales. Nous avons décidé dans
 56 un premier temps de nous intéresser à des méthodes d'optimisations, notamment
 57 l'approche par descente de gradient.

58 *2.4. Optimisation par descente de gradient*

59 Étant donné le nombre de paramètres, nous avons décidé d'appliquer une
 60 méthode d'optimisation par descente de gradient afin de faire converger les résultats
 61 du modèle de simulation avec le modèle de référence. La méthode par descente
 62 de gradient est une méthode permettant de trouver le minimum d'une fonction
 63 différentiable. La méthode de descente de gradient est une méthode comme son
 64 nom l'indique qui se base sur le gradient. Le gradient d'une fonction de plusieurs
 65 variables en un certain point est un vecteur qui caractérise la variabilité de cette
 66 fonction au voisinage de ce point. Le gradient se calcule à partir de la dérivée de
 67 la fonction. Si le gradient est élevé, la fonction est loin du minimum, si le gradient
 68 est faible ou nul, alors on est proche du minimum. La fonction de coût doit par la

69 suite être minimisée via l'algorithme de descente de gradient afin de d'obtenir un
70 modèle de simulation proche du modèle de référence.

71 Dans notre cas, la fonction à minimiser est la fonction permettant de générer
72 les résultats du simulateur. À partir des fichiers contenant le simulateur, nous nous
73 sommes rendu compte qu'il est actuellement impossible de calculer la fonction
74 mathématique du simulateur, et par conséquent, sa dérivée est inaccessible. Cela
75 empêche les approches basées sur les fonctions différentiables.

76 *2.5. Difficultés pour établir la fonction de coût et comparer les séries temporelles*

77 Pour la fonction de coût, nous nous sommes tournés vers les fonctions vers
78 la Racine de l'erreur quadratique moyenne (RMSE). La fonction RMSE étant
79 parfaitement définie dans le cadre de différences entre les valeurs entre 2 séries de
80 points. Cette méthode évalue la dispersion des résidus. Nous nous sommes rendu
81 compte que la série temporelle de simulation et la série temporelle de référence
82 n'étaient pas composé du même nombre de points. Les séries temporelles de
83 référence et de simulation disposait d'un temps final (tf) identique, mais pas du
84 même pas de temps, par conséquence le nombre de points entre le modèle de
85 référence et le modèle de simulation n'était pas identique. La différence en terme
86 de nombre de points rendaient impossible l'application de méthode comme la
87 RMSE pour comparer les 2 modèles.

88 *2.6. Correction de la longueur des Time-Series*

89 Nous avons besoin d'une métrique adaptée pour comparer 2 séries temporelles.
90 L'une des métriques abondamment utilisée est la méthode RMSE. Cette méthode
91 nécessite d'avoir 2 séries de temporelles de même longueur, c'est -à-dire avec
92 le même nombre de points. Afin d'obtenir des séries temporelles avec le même

93 nombre, on applique une transformation sur le modèle de référence, en basant la
94 définition du timestep (dt) et de l'output step (dr) sur le temps final du modèle de
95 simulation et sur le nombre de points totaux.

$$tf(\text{tempsfinal}) = Dt/Dr \quad (1)$$

96 Cette formule permet d'obtenir une série temporelle de référence ayant le
97 même nombre de points que le modèle de simulation. Cela permet par la suite
98 l'utilisation de méthodes permettant de comparer 2 séries temporelles notamment
99 la RMSE.

100 *2.7. Modification manuelle des paramètres U et h*

101 L'équipe RTE a indiqué le paramètre le plus prometteur afin de faire con-
102 verger le modèle de simulation vers le modèle de référence. Ce paramètre est
103 le paramètre u (Wind Speed), et les modifications apportées ont été testées pour
104 un intervalle de valeur allant de 0.1 à 7.0 . La modification de ce paramètre ac-
105 centue la courbure générale de la série temporelle, les points sont plus rapprochés,
106 L'amplitude maximale prise par les points est plus importante. Par la suite, nous
107 avons évalué l'impact de la modification du paramètre de Tension sur la série tem-
108 porelle. L'intervalle de valeur du paramètre de tension est de 1 à 40 000. La plage
109 de valeur étant grande, il a été établi que la tension a un impact important et que
110 le pas pour explorer l'intervalle doit être assez faible pour ne pas rater des séries
111 temporelles intéressantes à exploiter.

112 Au final, une exploration des plages de valeurs avec un pas de 100 donne des
113 résultats qui sont plus facilement exploitables et transposables à d'autres séries
114 temporelles. Par ailleurs, une définition du paramètre de tension = 100 provoque
115 une augmentation significative de l'amplitude maximale.

116 2.8. *Modification manuelle des paramètres cl0 et eps*

117 A la suite de l'étude de l'impact des paramètres u et h sur la série temporelle
118 de simulation, nous avons étendus l'étude aux paramètres cl0 et eps défini dans
119 la configuration du simulateur. L'augmentation de la valeur du paramètre cl0
120 provoque une augmentation linéaire de l'amplitude maximale de la série tem-
121 porelle. La modification de la valeur de eps n'a pas d'impact sur la plupart des
122 séries temporelles, certaines séries temporelles montrent une augmentation de
123 l'amplitude pour eps = 0.85.

124 2.9. *Résultats de l'approche manuelle*

125 L'approche manuelle a permis d'obtenir des résultats intéressants pour 2 Time-
126 Series : la Time-Series n°1 et n°2 de la Liste 2. Par itérations et en utilisant une
127 méthode dichotomique, nous sommes parvenus à des résultats de simulation qui
128 se rapprochent visuellement du modèle de référence.

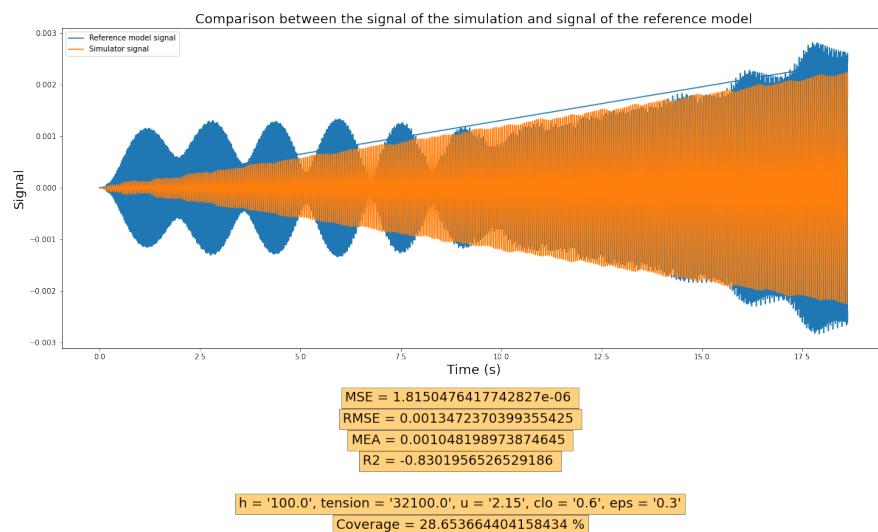


Figure 7: Time-Series 1, List 2

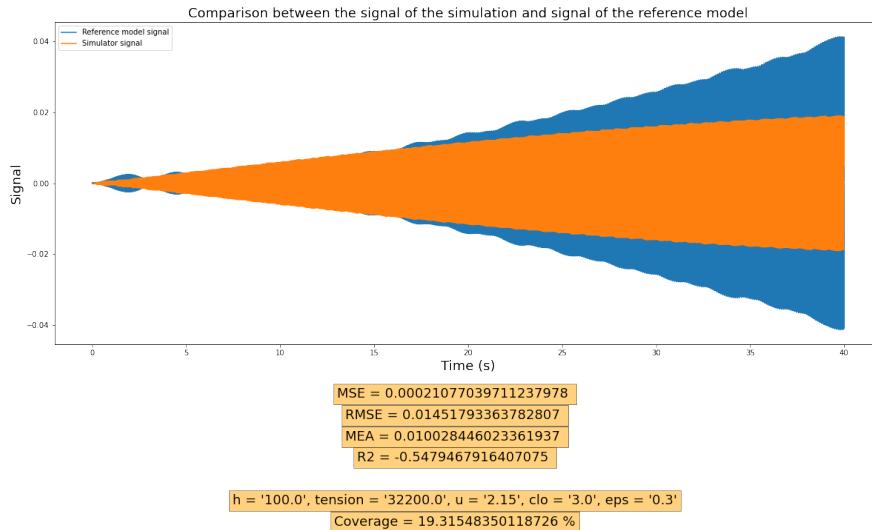


Figure 8: Time-Series 2, List 2

129 Néanmoins, l'approche manuelle n'a pas donné de résultats concluants sur les
 130 autres Time-Series. Cela est dû principalement au temps d'exécution du simula-
 131 teur pour la Time-Series correspondante. Les 2 Time-Series sur lesquelles nous
 132 sommes parvenus à obtenir des résultats prometteurs étaient des Time-Series qui
 133 en terme de longueur (temps final) étaient les plus courtes, cela induit un plus
 134 faible nombre de points et par conséquent étaient celles qui demandent le moindre
 135 temps d'exécution. Cela a permis de réduire les plages de recherches rapidement.
 136 Cependant pour les autres Time-Series, cette approche se révèle n'être pas assez
 137 efficace.

138 2.10. Modélisation

139 Nous avons obtenu des résultats avec 2 séries temporelles qui se révèlent
 140 prometteurs en terme d'amplitude de valeur. Nous avons donc décidé de con-
 141 stituer un ensemble d'apprentissage primitif avec des données afin d'essayer de

142 construire un modèle primitif et de le calibrer.

143 *2.11. Ensemble d'apprentissage*

144 Nous nous sommes rendu compte que pour chaque pas de temps, le simulateur
145 produit une valeur, alors que le modèle de référence quant à lui, produit pour
146 chaque pas de temps 2 valeurs : une valeur positive et une valeur négative. Afin
147 de pouvoir créer une modélisation, nous nous sommes concentrés dans un premier
148 temps sur les valeurs positives du modèle de référence.

149 Nous récupérons les données du simulateur pour entraîner le modèle et les
150 données du modèle de référence en tant que données de test. Nous appliquons
151 cette procédure en concaténant dans un premier temps les séries temporelles puis
152 en séparant les données de simulation et de référence avec 80% des données pour
153 l'entraînement et 20% pour la validation.

154 *2.12. Traitement des données*

155 Par moments, nous obtenons des valeurs non-exploitables dans le modèle
156 de référence comme les valeurs manquantes (NA). Ces valeurs sont expurgées
157 de l'ensemble d'apprentissage. Enfin, nous veillons à ce que la longueur du
158 vecteur contenant les données de référence soit de la même longueur que celle
159 pour l'entraînement.

160 *2.13. Sélection du modèle d'apprentissage*

161 Nous avons pour cela évaluer les différents types de modèle d'apprentissage
162 existants de manière empirique afin de déterminer quel type d'algorithme produit
163 un résultat exploitable. Nous sommes parvenus à la conclusion que le modèle
164 d'apprentissage Random Forest produit les résultats les plus prometteurs. Nous
165 avons donc testé avec ce modèle d'apprentissage avec les paramètres par défaut.

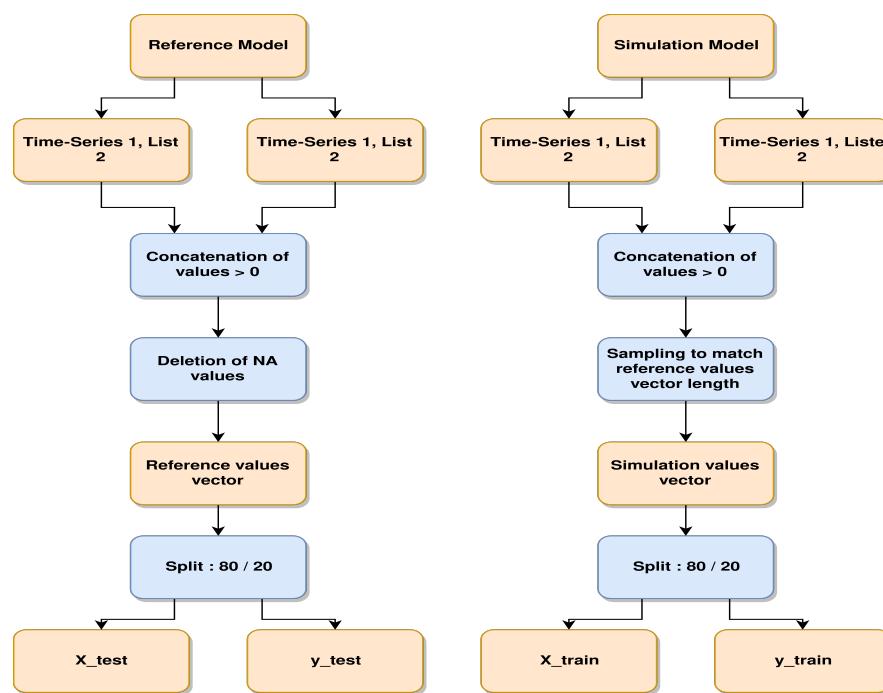


Figure 9: Preprocessing and modelization

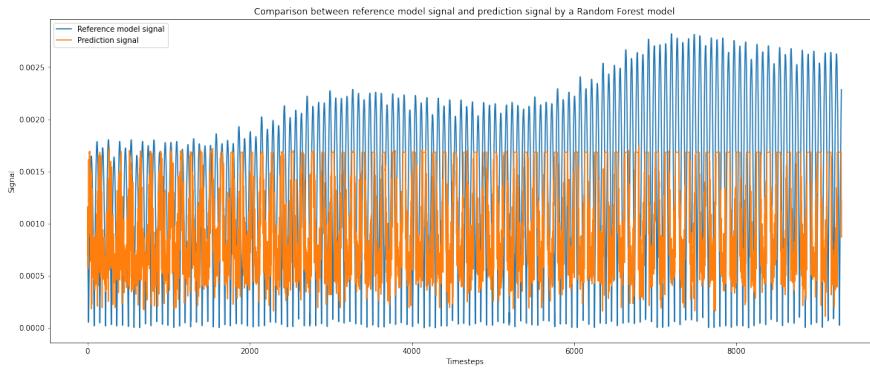


Figure 10: Reference model signal and the Random Forest model prediction signal

166 L’observation visuelle empirique et les valeurs obtenues par la métrique RMSE
 167 poussent à croire que bien que le résultat soit prometteur,l’ensemble d’apprentissage
 168 actuel n’est pas suffisant pour créer un modèle calibré de mécanique des flu-
 169 ides. Nous avons donc décidé de produire un ensemble d’apprentissage plus
 170 conséquent. Étant donné que les modifications manuelles de la configuration du
 171 simulateur n’a pas donné de conclusions transposables à toutes les séries tem-
 172 porelles, nous avons décidé de changer d’approche. Nous avons décidé d’utiliser
 173 un algorithme d’optimisation non basé sur le gradient, afin de trouver la combinai-
 174 son de paramètres optimale. Le but de cette approche est qu’une approche guidée
 175 par optimisation serait plus rapide qu’une approche dichotomique pour trouver
 176 les plages de valeurs intéressantes. Pour ce faire, nous avons étudié les méthodes
 177 d’optimisation afin de déterminer lesquelles sont utilisables dans le cadre de notre
 178 étude.

179 *2.14. Etudes des méthodes d’optimisation*

180 Nous avons donc réalisé une étude des différentes méthodes d’optimisation
 181 afin de déterminer lesquelles sont adaptées à notre cas d’usage.

Table 2: Optimization methods study 1

	Objective	Principles	Hyper-parameters	Advantages	Disadvantages	Field of use	Algorithm	Applicability in our use case
Simulated Annealing	Reduce energy level of the system	if negative variation, change accepted and state changes. if positive variation, change can be accepted based on a probability law. Probability decreases when temperature decreases	Temperature, Temperature decrease rate, Cost function, Stop criterion	Accuracy, Flexibility, Applicability	Convergence rate, Parameters initial values	Telecom, Electronic circuit	Hyperopt	Low memory usage, Accuracy with enough iterations
Genetic Algorithm	Select the best solutions by elimination and reinforcement	Population initialization, Objective function calculation, Selection, Cross-over, Mutation	Population size, Iterations, Cost function, Mutation rate, Cross-over type	Applicability : Useful when there is a lot of parameters, Accuracy : avoid local minima	Accuracy : no guarantee to find optimal solution, Applicability : not applicable in all circumstances, Convergence rate : depends heavily on data used and space search, Flexibility : Cost function need to be well defined	Telecom, Robotics		Not effective with big search space, Need a lot of chromosomes, No convergence guarantee, Cost function is heavy to compute
Adam	Minimize the cost function	Gradient calculation, First Moment update, Second Moment update, Parameters update	Learning rate, First moment decrease rate, Second moment decrease rate, Epsilon	Convergence rate : fast, Applicability : used in a lot of domains, Applicability : Low memory usage, Flexibility : no need to changes parameters a lot to fine tune	Accuracy : no convergence for convex functions	Deep Learning, Machine Learning	Neural Networks	Need a gradient, not accessible
Adagrad	Minimize the cost function	Calculation of the gradient, Add square gradient to the accumulator, Calculate gradient / accumulator	Learning rate, Accumulator, Epsilon	Convergence rate : fast, Applicability : Used a lot in neural network and machine learning, Flexibility : Applicable to all derivable functions	Accuracy : Tendency to have a vanishing gradient	Deep Learning, Machine Learning	Neural Networks	Need for a gradient, not accessible
PSO	Particle swarm moving in a search space to find global minimum	Particles initialization, Update of the cognitive component, Update of the social component, Calculation of the velocity of the particles, Update of particles position	Inertia, Cognitive component, Social component, Iterations	Convergence rate : fast, Flexibility : no gradient, low number of parameters	Applicability: Initial parameters are hard to optimize, Accuracy : can get stuck in a local minimum	Diesels Motors, Transport network		Space search is big, hard to use, Lot of local minima, Memory problem with a lot of particles

Table 3: Optimization methods study 2

	Objective	Principles	Hyper-parameters	Advantages	Disadvantages	Field of use	Algorithm	Applicability in our use case
Differential evolution	Selection of the best solutions	Population initialization, Recombination, Selection	Population, Scaling factor, Recombination, Iterations, Type of recombination	Applicability : only 3 parameters and no need to implement again the problem to use the method, Speed : Faster than the genetic algorithm	Accuracy : No guarantee to find the global minimum, Convergence rate : depends heavily on initial parameters and population	Neural networks training	Tensorflow	Space search is big, makes the method hard to use, Need to optimize hyper-parameters, Cost function is heavy to compute
Nelder-Mead	Search with a simplex, Geometrical interpretation	Order, Reflection, Expansion, Contraction, Shrink	Vertex, Iterations	Applicability : low number of parameters, no gradient	Accuracy : degrades heavily with a lot of parameters, Applicability : arbitrary definition of the initial simplex, Accuracy : no guarantee to converge to the global minimum	Chemistry	Tensorflow	High number of simulator parameters makes it hard to use, No guarantee of convergence, Cost function is heavy to compute

182 L'étude des méthodes d'optimisations a permis d'établir que ces méthodes
 183 sont soient inutilisables dans notre cas d'usage ou bien ont une moins bonne per-
 184 formance que Hyperopt. Nous avons donc étudié la méthode d'optimisation Hy-
 185 peropt dans les détails pour améliorer nos résultats.

186 *2.15. Hyperopt : principes*

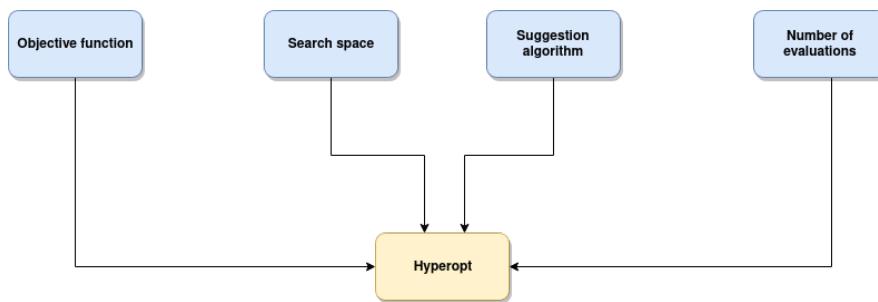


Figure 11: Hyperopt Arguments

187 L'algorithme Hyperopt fonctionne avec 4 arguments. Une fonction objectif à
188 minimiser, le but étant de trouver la configuration donnant en sortie de la fonction
189 objectif le plus petit possible. L'espace de recherche dans notre cas est la liste des
190 paramètres utilisés dans le simulateur avec leur plage de valeur respective. Un des
191 avantages fournis par Hyperopt est la possibilité de définir les plages de valeur
192 selon des distributions particulières (selon une distribution uniforme ou lognor-
193 male par exemple), de définir à la fois des distributions de valeurs discrètes pour
194 certains paramètres et des valeurs flottantes pour d'autres. L'espace de recherche
195 accepte également 1 espace de recherche imbriqué dans un autre. L'algorithme
196 de suggestion est le cœur de l'algorithme Hyperopt. Il permet d'évaluer quelle
197 partie de l'espace de recherche est prometteur en suggérant des combinaisons de
198 paramètres prometteurs à passer à la fonction objectif. L'algorithme de sugges-
199 tion principalement utilisé est le Tree-Parzen Estimator (TPE). Enfin, le nom-
200 bre d'évaluations permet de définir le nombre de fois où la fonction objectif sera
201 évaluée. Plus le nombre d'évaluations est élevé, plus l'algorithme peut explorer
202 les plages de valeurs pour trouver la combinaison de paramètres optimale.

203

204 Le fonctionnement général se base sur l'algorithme de suggestion qui évalue
205 l'espace de recherche afin d'optimiser la recherche vers des sous-plages de paramètres
206 qui peuvent être intéressantes à explorer. Ces paramètres sont ensuite fournis en
207 entrée à la fonction objectif dont la sortie doit être minimisée. L'algorithme garde
208 en mémoire le meilleur résultat obtenu.

209

210 L'algorithme TPE se base sur un historique de recherche que l'algorithme con-
211 struit au fur et à mesure des itérations d'Hyperopt. Cet historique est composé de

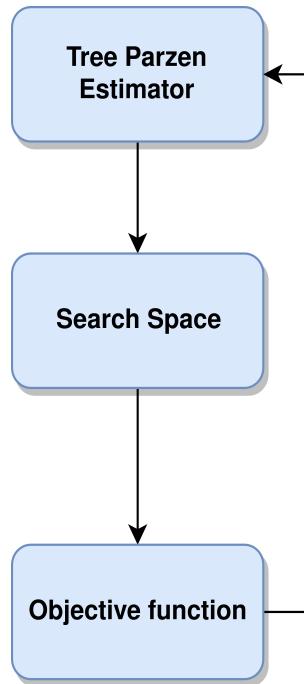


Figure 12: Hyperopt algorithm

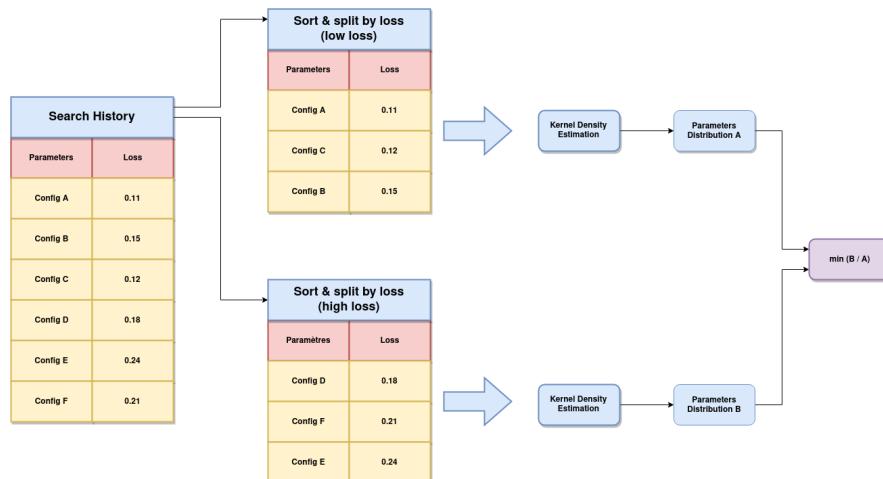


Figure 13: TPE Algorithm

212 tuples composé d'un côté des paramètres et de l'autre la sortie obtenue par la fonc-
213 tion objectif. Dans un premier temps, TPE trie les tuples selon la valeur obtenue
214 par la fonction objectif puis les sépare en 2 ensembles. Par la suite, TPE applique
215 une estimation par noyau permettant d'obtenir une probabilité dite de densité des
216 paramètres dans les 2 ensembles. Enfin, on récupère le minimum du rapport entre
217 les 2 ensembles c'est-à-dire avec le rapport du moins bon ensemble sur le meilleur
218 ensemble. La combinaison de paramètres obtenus est stockée dans l'historique de
219 recherche et est passée à la fonction objectif.

220 *2.16. Hyperopt : difficultés d'implémentations et applications*

221 Nous avons donc créer un espace de recherche que l'algorithme Hyperopt
222 peut explorer. Néanmoins, nous avons rencontré des difficultés liées à une con-
223 sommation de RAM qui augmentait de façon exponentielle. Cela était dû au fait
224 que les simulations générées s'accumulaient dans la RAM ainsi que le paramètre
225 pp qui était définie sur "True" faisait que l'affichage des résultats sur la console
226 d'exécution faisait crasher l'outil de programmation. La suppression systématique
227 de la simulation après son usage pour les calculs ainsi que la définition du paramètre
228 pp à "False" ont permis de régler ce problème.

229 Nous avons par la suite d'appliquer l'algorithme Hyperopt à nos Time-Series.
230 Nous avons pour cela utilisé des itérations allant de 10 à 550, des variations dans
231 la plage de recherche des paramètres, ainsi que l'usage des algorithmes de sugges-
232 tion Tree-Parzen et Recuit simulé. Avec cette configuration de paramètres, nous
233 ne sommes pas parvenus à obtenir des résultats qui soient en suffisant pour être
234 intégrés dans l'ensemble d'apprentissage.

235 De plus, le temps d'exécution de l'algorithme Hyperopt ne permettait pas
236 d'effectuer plus d'itérations pour affiner le résultat. Parmi les méthodes d'optimisation

237 exploitable que nous avons listées, se trouve la méthode basée sur le Recuit simulé.
 238 Nous avons donc essayé d'utiliser cet algorithme pour optimiser les paramètres de
 239 notre simulateur.

240 *2.17. Recuit Simulé*

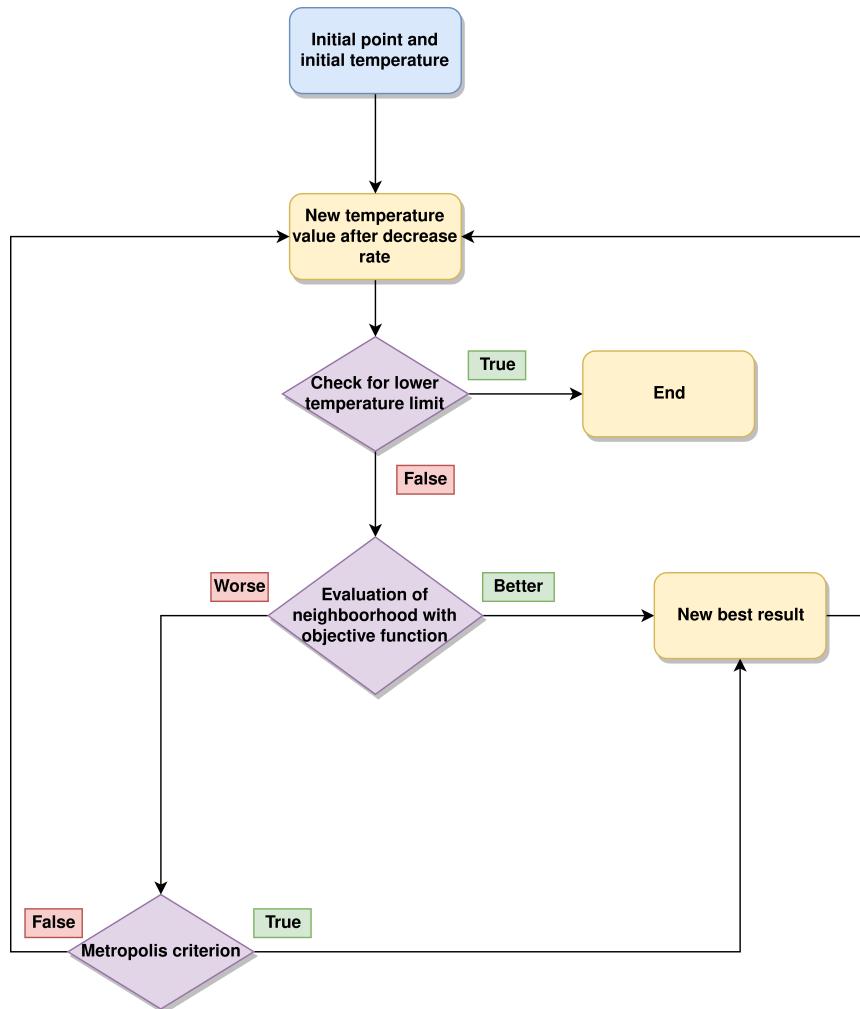


Figure 14: Simulated Annealing algorithm

241 Le recuit simulé est un algorithme inspiré du refroidissement des métaux.

242 Le principe consiste à définir une valeur de température de départ, un taux de
243 décroissance de température, ainsi que la limite minimale que la température
244 peut atteindre. L'algorithme commence par sélectionner un point initial et évalue
245 les points voisins à l'aide d'une fonction coût. Si le point voisin est meilleur
246 que le point actuel, l'algorithme le garde en mémoire comme étant le meilleur
247 résultat. Le cœur de l'algorithme repose sur le cas où le point voisin est moins
248 bon que le point actuel. Dans le cas suivant, il y a toujours une chance que
249 le résultat soit accepté selon une probabilité d'acceptation qui décroît avec la
250 valeur de température. A chaque itération, la température décroît selon le taux
251 de décroissance. Ce système, appelé critère de Métropolis, permet dans un pre-
252 mier temps d'explorer les résultats, car pour une température élevée, la probabilité
253 d'acceptation est élevée. Par la suite, à mesure que la température diminue, la
254 probabilité d'acceptation diminue ce qui permet d'exploiter les résultats dans la
255 région la plus prometteuse.

256 *2.18. Recuit Simulé : application*

257 L'application du recuit simulé au simulateur n'a pas permis d'obtenir de résultats
258 intéressants. Les paramètres trouvés ont été sujets à de grandes fluctuations et ne
259 montraient pas de signes de convergence. Néanmoins une variante plus robuste du
260 recuit simulé, appelée Dual Annealing semblait offrir une robustesse accrue par
261 rapport au recuit simulé. Cette approche se base sur la combinaison de 2 versions
262 du recuit simulé en combinant la version classique du recuit simulé avec une ver-
263 sion "rapide" qui permet la sélection des paramètres à explorer avec un change-
264 ment dans la méthode de probabilité d'acceptation de résultats et une méthode
265 de décroissement de température modifiée. Nous avons donc essayé d'appliquer
266 l'algorithme Dual Annealing pour optimiser les paramètres de notre simulateur.

267 *2.19. Dual Annealing : résultats*

268 Les résultats obtenus sont nettement plus intéressants que ceux obtenus via
269 recuit simulé simple, mais ne sont pas pour autant meilleur que les résultats
270 obtenus via Hyperopt. De plus, le temps d'exécution de cette méthode rend dif-
271 fice la réalisation de nombreuses itérations nécessaires pour évaluer la justesse
272 de l'algorithme dans notre cas d'usage et rend l'usage d'Hyperopt sur ces plages
273 d'itérations plus intéressantes en termes de résultats.

274 *2.20. Migration vers une architecture cloud : AWS Sagemaker*

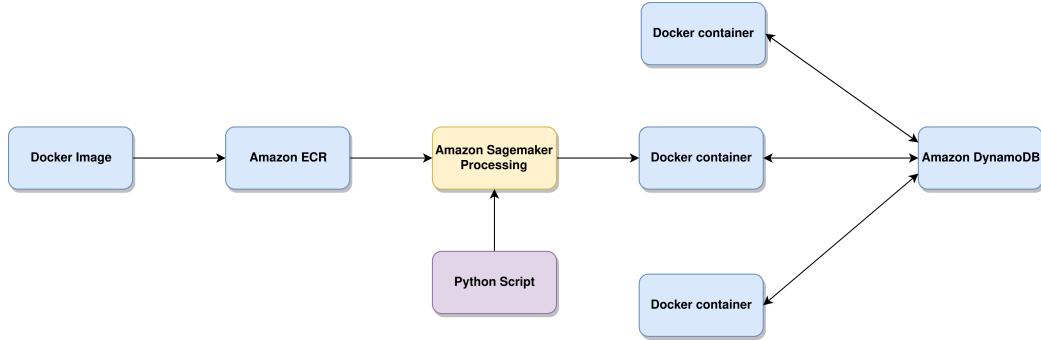


Figure 15: AWS Sagemaker : Principles

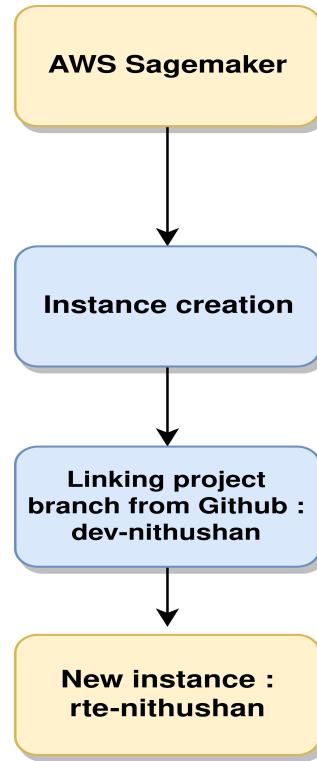


Figure 16: AWS Sagemaker : Instance creation process

275 Étant donné que les exécutions d'Hyperopt ne fournissaient pas de résultats
 276 exploitables pour un nombre d'itérations limitées, nous avons décidé de migrer

277 les exécutions vers une instance distribuée. Cette instance disposant de ressources
278 computationnelles plus importantes, permet de réaliser des calculs plus rapidement et ainsi d'utiliser l'algorithme Hyperopt pour un nombre d'itérations supérieur
279 à une exécution sur des machines en local. L'architecture choisie dans le cadre de
280 ce projet est AWS Sagemaker avec une instance distribuée ml.c4.2xlarge. Le code
281 utilisé dans cette instance est chargé depuis le Github.
282

283 *2.21. Modélisation avec amplitude maximale du modèle de simulation inférieure
284 au modèle de référence*

285 Nous avons essayé une nouvelle approche basée sur la génération de simulation dont l'amplitude maximale se rapproche au maximum de l'amplitude maximale du modèle de référence. Néanmoins, cette approche n'a pas fourni de meilleures modélisations que l'approche classique basée la ressemblance visuelle.

289 *2.22. Passage de la fonction coût RMSE à R2 Score*

290 Jusqu'à présent, l'algorithme Hyperopt utilisait une fonction coût RMSE. Nous
291 avons tenté d'utiliser une fonction coût basé sur le R2 score. Le R2 Score est une
292 métrique évaluation la dispersion des résultats entre 2 jeux de données. Un résultat
293 négatif indiquant une régression moins bonne qu'une ligne droite, un résultat nul indiquant une différence nette et un résultat proche
294 de 1 indiquant une ressemblance entre les 2 courbes. Étant donné que Hyperopt
295 ne prend que des fonctions à minimiser et non maximiser, nous avons choisis de
296 définir une fonction coût définit par la formule :

$$F(x) = 1.0 - R2Score \quad (2)$$

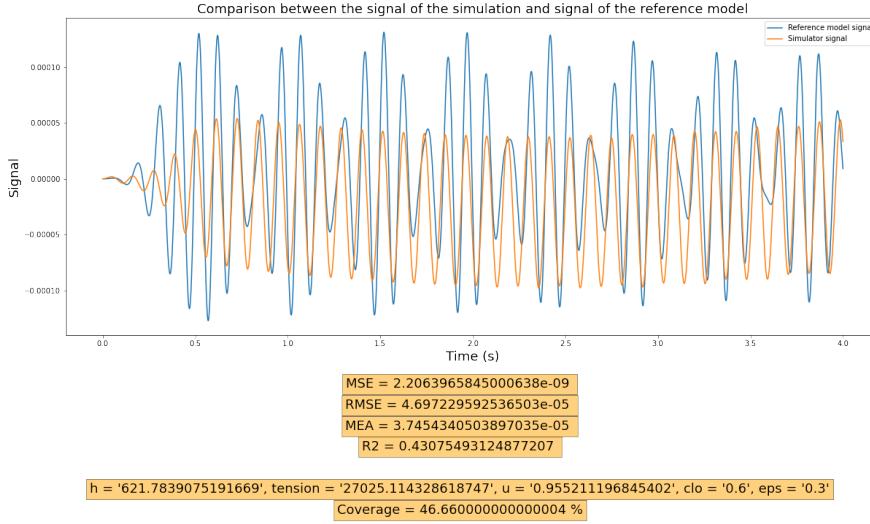


Figure 17: Hyperopt resultats with 1 - R2 Score cost function

298 L’application de cette fonction coût à Hyperopt a permis d’améliorer signifi-
 299 cativement la qualité des résultats obtenus. Les résultats obtenus étant large-
 300 ment plus intéressants visuellement que les résultats obtenus avec la fonction coût
 301 RMSE. Elle permet également d’établir une métrique plus robuste mathématiquement
 302 d’évaluation de la ressemblance entre les Time-Series qui jusqu’à présent était
 303 surtout évalué sur leurs tendances et leur aspect visuels.

304 Néanmoins bien que l’algorithme Hyperopt avec cette nouvelle fonction coût
 305 basé sur le R2 score est prometteuse, l’algorithme Hyperopt ne converge pas de
 306 manière optimale vers le meilleur résultat. Pour pallier à cette difficulté, nous
 307 avons élaboré une nouvelle approche pour faire converger l’algorithme de manière
 308 optimale.

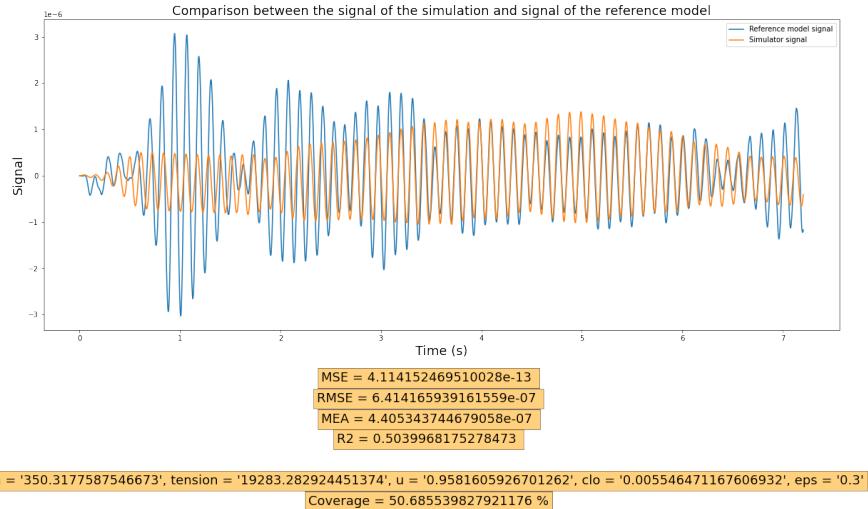


Figure 18: Time-Series 1, List 1

309 **2.23. Nouvelle approche semi-automatique**

310 Cette approche se base sur la récupération de résultats obtenus pour une plage
 311 de paramètres donnée, une liste des plages plus restreintes à explorer par l'algorithme.
 312 Cette approche permet de cibler les plages à explorer ce qui permet à d'autres
 313 exécutions d'Hyperopt de rechercher les meilleurs résultats dans ces plages de
 314 valeurs et éviter les minima locaux. Cette approche a produit des résultats ex-
 315 ploitables et tangibles qui ont permis d'obtenir des simulations se rapprochant du
 316 modèle de référence pour plusieurs Time-Series dont notamment la Time-Series
 317 1 et 6 de la Liste 1.

318 **2.24. Modélisation avec le nouvel ensemble d'apprentissage**

319 Étant donné que nous avons obtenu des données supplémentaires, nous avons
 320 tenté de créer un nouvel ensemble d'apprentissage incorporant ces données (Time-
 321 Series 1 et 6, Liste 1). Nous disposons donc d'un ensemble de simulations avec 4

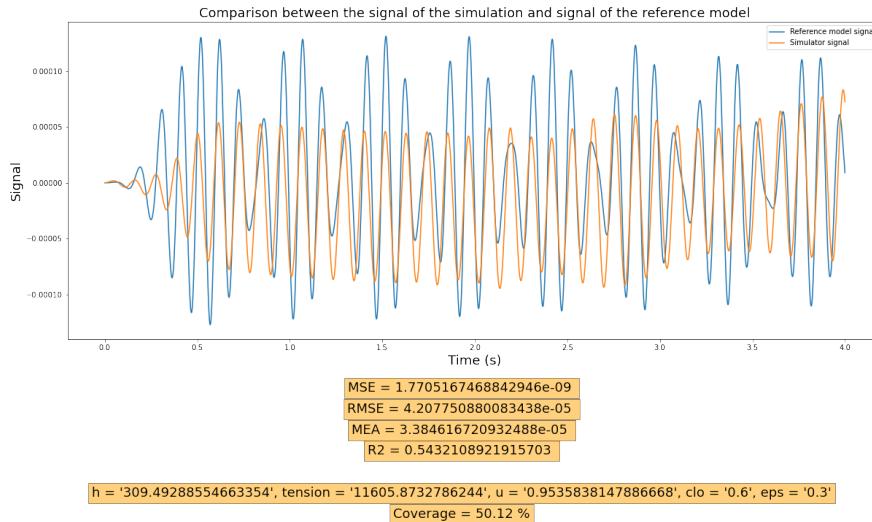


Figure 19: Time-Series 6, List 1

322 Time-Series dont 2 obtenus manuellement et 2 avec l'approche semi-automatique
 323 de Hyperopt :

- 324 • Time-Series 1, Liste 2 (Manuelle)
- 325 • Time-Series 2, Liste 2 (Manuelle)
- 326 • Time-Series 1, Liste 1 (Hyperopt Semi-Automatique)
- 327 • Time-Series 6, Liste 1 (Hyperopt Semi-automatique)

328 Cette modélisation est basée sur la récupération des valeurs positives du modèle
 329 de référence et de la concaténation des valeurs issus des simulations d'un côté et
 330 celle issus du modèle de référence. Cela permet un ensemble d'apprentissage et
 331 un ensemble de validation avec 80 % des données destinées à l'apprentissage et
 332 20 % pour le test.

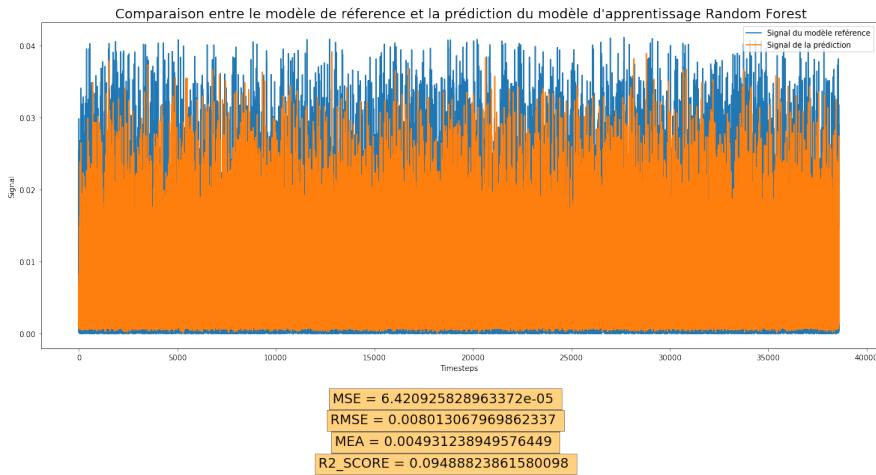


Figure 20: Modelization and prediction with a set of 4 Time-Series with Random Forest

333 Cette nouvelle modélisation basée sur un algorithme d'apprentissage Random
 334 Forest produit un résultat plus intéressant en terme de R2 score (0.09 vs 0.0009)
 335 et un rendu visuel qui se rapproche mieux du modèle de référence.

336 2.25. *Modélisation : Approche basée sur la récupération de toutes les valeurs*

337 Par la suite, nous avons décidé de créer des ensembles d'apprentissage con-
 338 tenant non seulement les valeurs positives, mais de prendre l'intégralité des valeurs
 339 positives ou négatives.

340 2.26. *Comparaison des ensembles d'apprentissages*

341 Nous avons choisi de comparer le résultat d'un modèle calibré selon l'ensemble
 342 d'apprentissage choisi. Dans un premier cas, nous testons un ensemble d'apprentissage
 343 composé des résultats obtenus à la main et ceux fournis par Hyperopt. Les résultats
 344 obtenus à la main ont un visuel intéressant, mais un R2 score bas.

- 345 • Time-Series 1, Liste 2 (Approche manuelle)

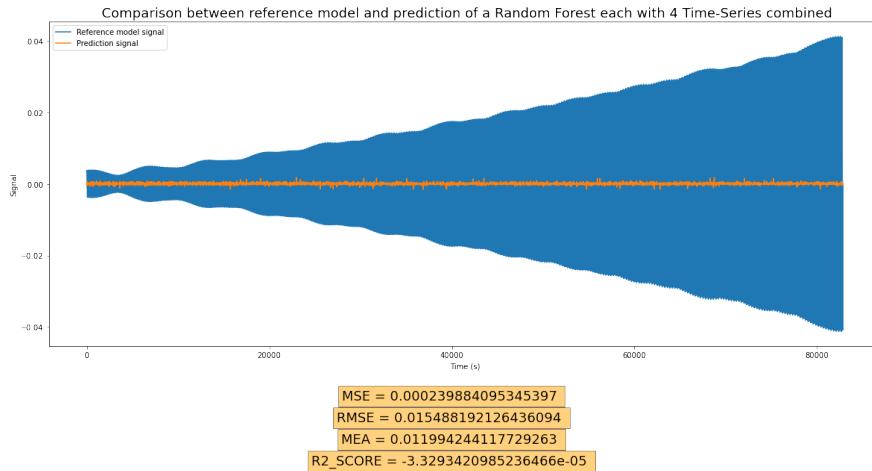


Figure 21: Modelization and prediction with a set of 4 Time-Series with Random Forest

346 • Time-Series 2, Liste 2 (Approche manuelle)

347 • Time-Series 1, Liste 1 (Hyperopt Semi-Automatique)

348 • Time-Series 6, Liste 1 (Hyperopt Semi-automatique)

349 Dans un second temps, nous testons uniquement avec un ensemble d'apprentissage
 350 obtenu via l'approche Hyperopt semi-automatique avec un R2 score élevé. Le
 351 modèle d'apprentissage utilisé est Random Forest.

352 • Time-Series 1, Liste 1 (Hyperopt Semi-Automatique)

353 • Time-Series 6, Liste 1 (Hyperopt Semi-automatique)

354 Les résultats obtenus montrent qu'un ensemble d'apprentissage basé sur des
 355 séries temporelles ayant un R2 score élevé donne un meilleur modèle calibré (0.59
 356 vs -3.0). Cela nous permet de baser nos nouveaux ensembles d'apprentissage
 357 sur le postulat que les Time-Series faisant partie de cet ensemble d'apprentissage

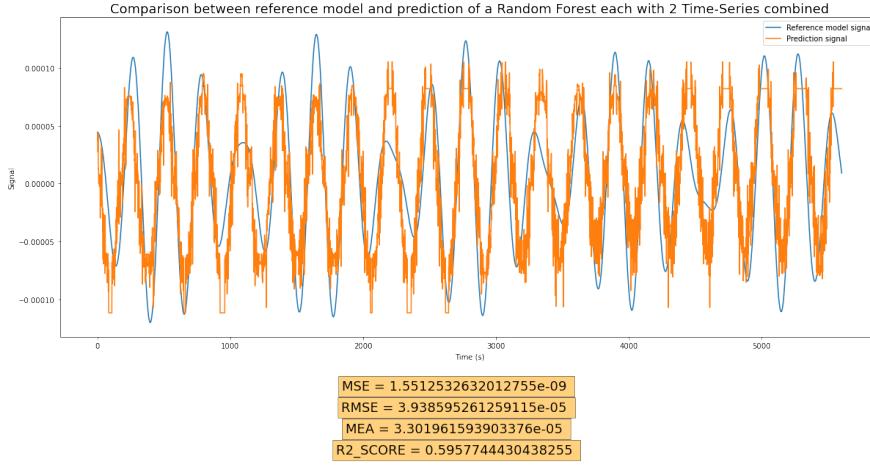


Figure 22: Modelization and prediction with a set of 2 Time-Series with Random Forest

358 doivent avoir un R2 score au-dessus de 0.30. De même, pour évaluer la qualité du
 359 modèle calibré, nous partons du postulat qu'un modèle calibré est représentatif du
 360 modèle de référence à partir du moment où son R2 score est supérieur ou égal à
 361 0.70.

362 *2.27. Modélisation avec un modèle d'apprentissage basé sur des Réseaux de neu-
 363 rones*

364 Nous reprenons le même ensemble d'apprentissage et nous réalisons un modèle
 365 d'apprentissage basé sur des Réseaux de neurones. Le réseau de neurones est
 366 composé d'une couche de 8 neurones et utilise pour fonction d'optimisation SGD
 367 avec un taux d'apprentissage = 0.001.

368 Le résultat obtenu est moins intéressant que celui obtenu via Random Forest
 369 (0.36 vs 0.59), mais reste exploitable. Néanmoins, du fait de la nature stochastique
 370 des réseaux de neurones, il est possible que les fluctuations n'ont pas permis en-
 371 core de trouver la combinaison de paramètres optimaux. Afin de palier au manque

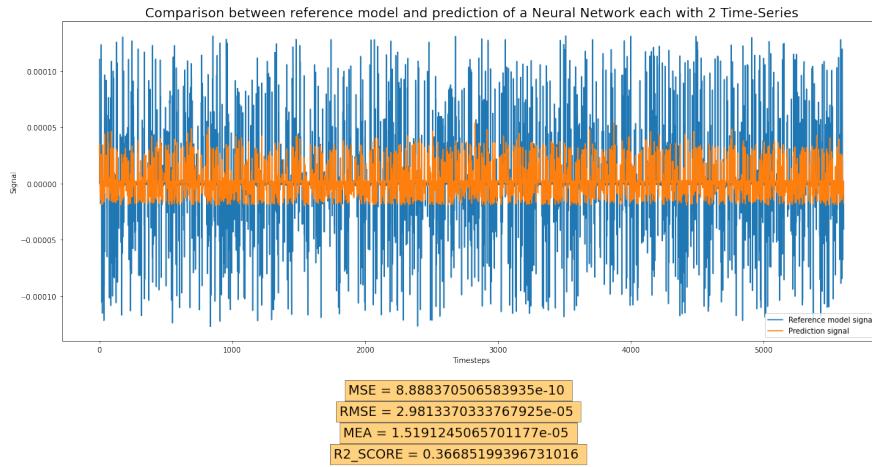


Figure 23: Modelization and prediction with a set of 2 Time-Series with neural network

372 de données dont nous disposons, nous avons eu l'idée de récupérer plusieurs
 373 résultats obtenus pour une série temporelle avec différents jeux de paramètres
 374 et de la concaténer. Cette approche vise à augmenter les données via des séries
 375 temporelles très similaires.

376 2.28. *Modélisation par augmentation de données*

377 Nous avons également exploré l'approche basée sur l'augmentation de données,
 378 c'est-à-dire en récupérant plusieurs résultats proches
 379 mais différents d'une même Time-Series afin d'avoir un ensemble d'apprentissage
 380 plus volumineux. Pour cela, nous avons concaténé avec 3 (Random Forest) et 6
 381 (Réseaux de neurones) résultats proches obtenus pour la Time-Series n°6 de la
 382 Liste 1. Nous avons fait un entraînement avec 80 % des valeurs et consacré 20 %
 383 pour la validation. Nous avons testé ce nouvel ensemble d'apprentissage avec un
 384 modèle Random Forest et un modèle basé sur des réseaux de neurones.

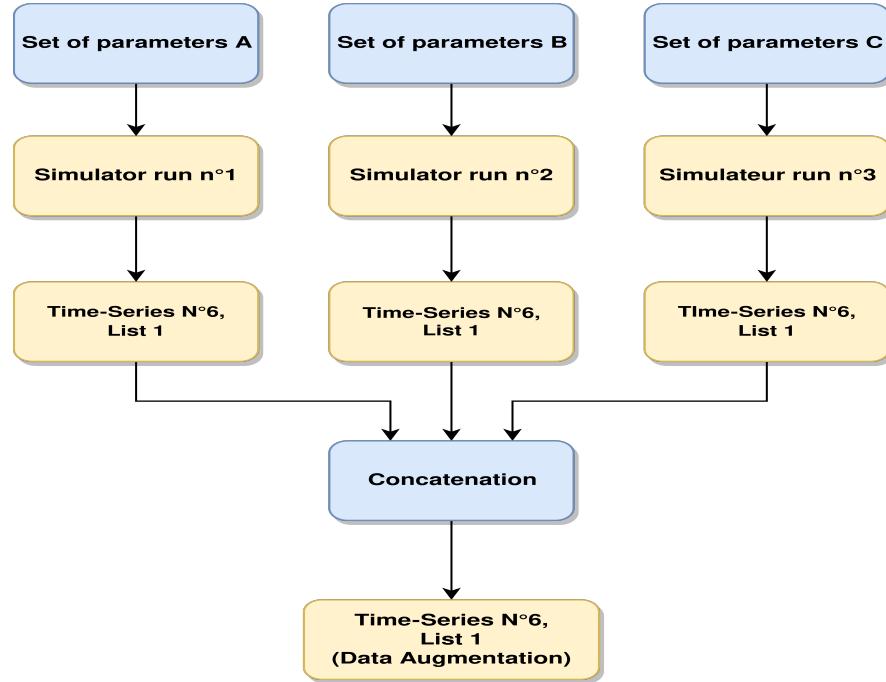


Figure 24: Data Augmentation

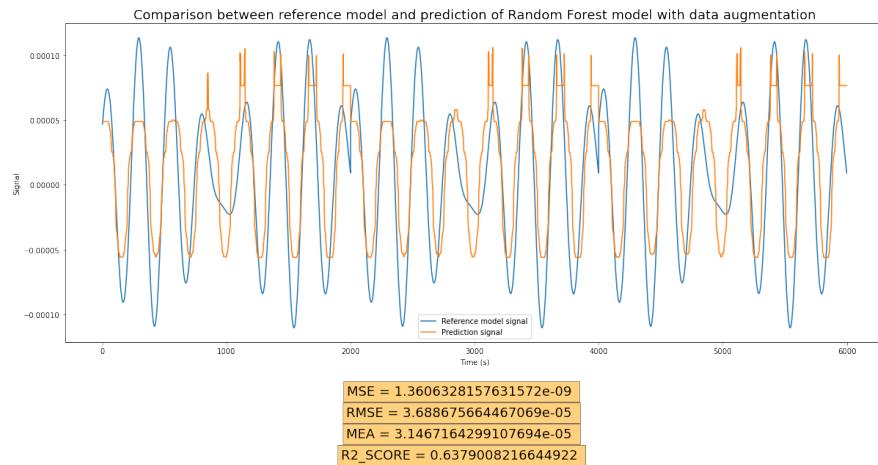


Figure 25: Modelization and prediction with a set of 2 Time-Series with Random Forest and data augmentation

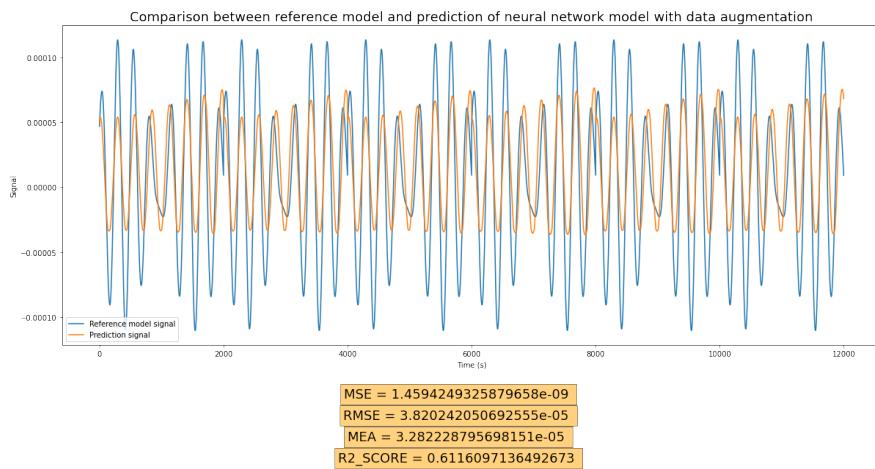


Figure 26: Modelization and prediction with a set of 2 Time-Series with neural network and data augmentation

385 Les résultats obtenus mettent en évidence un R2 score proche de 0.65 sur la
386 même Time-Series alors que les simulations ont un R2 score proche de 0.54, ce
387 qui peut signifier que cette approche peut être intéressante à exploiter si le volume
388 des données disponibles le permet.

389 **3. Conclusions**

390 Nos travaux ont donc permis d'explorer les différentes approches possibles
391 afin d'automatiser complètement le modèle simplifié. Nos résultats montrent que
392 l'usage de la méthode d'optimisation Hyperopt permet de faire de l'exploration
393 semi-automatique avec la désignation des espaces de recherche prometteurs. L'usage
394 d'une fonction objectif 1 - R2 Score
395 permet notamment des gains de convergence significatifs par rapports à d'autres
396 métriques tel que la RMSE. Ces approches ont donc permis d'établir qu'une ap-
397 proche semi-automatique permet d'obtenir à la fois des résultats plus intéressants
398 et avec un délai plus court dû à une exploration ciblée.

399 Dans le cadre de la création d'un modèle calibré, nous avons établi que la con-
400 stitution d'un ensemble d'apprentissage avec critère de R2 Score supérieur à 0.30
401 et de notre métrique personnelle basé sur le taux de couverture permet de créer des
402 modèles de meilleure qualité. Parmi les modèles de machine learning existant, les
403 modèles Random Forest et réseaux de neurones s'avèrent prometteurs. L'usage
404 de l'augmentation des données peut être une voie intéressante qui mérite d'être
405 explorée davantage.

406 *3.1. Perspectives*

407 Dans le cadre de nos travaux, nous avons exploré différentes méthodes
408 d'optimisation utilisables afin d'automatiser le modèle simplifié. Néanmoins, bien

409 que nous ayons réussi à réduire la part de recherche manuelle, il n'est pas possible
410 à l'aide d'Hyperopt de complètement automatiser la recherche des combinaisons
411 de paramètres optimales. Des approches basés sur la recherche de paramètres op-
412 timaux sur des sous-sections des séries temporelles peut être une piste intéressante
413 à explorer.

414 La création d'un modèle calibré n'a pas vu le jour dû au manque de données
415 disponibles dans l'ensemble d'apprentissage. Il est possible que la constitution
416 d'un ensemble d'apprentissage basé sur d'autres métriques d'évaluation de series
417 temporelles puisse produire des résultats plus intéressants. Les critères de choix de
418 ces métriques d'évaluation de séries temporelles pourront notamment faire l'objet
419 d'une étude plus approfondie.

420 **Remerciements**

421 Je voudrais remercier chaleureusement dans un premier temps Hamdi AM-
422 ROUN qui par sa supervision amicale et flexible, sa rigueur et la confiance qu'il
423 m'a accordé pour mener ses travaux m'ont permis d'acquérir des compétences
424 que je n'avais pas et de me dépasser. Je souhaite témoigner de ma reconnaissance
425 et de ma gratitude pour ses conseils précieux.

426 Je voudrais également adresser mes vifs remerciements à
427 Youness EL MARHRAOUI pour sa sympathie, sa disponibilité et la pédagogie
428 dont il a fait preuve à mon égard qui m'a permis de prendre mes marques très
429 rapidement dans mes travaux et a été pour moi une aide personnelle et profession-
430 nelle cruciale.

431 Enfin, je voudrais remercier Massil KSOURI qui dès le premier jour de notre
432 collaboration et tout le long de celle-ci à preuve d'une grande sympathie et disponi-

⁴³³ bilité me permettant de prendre part aux travaux avec une base solide.

⁴³⁴ **References**