Analyse de l'impact de l'assimilation de données par filtre d'ensemble de Kalman sur la performance des prévisions hydrologiques à court et moyen terme

par

Jade LEBEL

MÉMOIRE PRÉSENTÉ À L'ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
COMME EXIGENCE PARTIELLE À L'OBTENTION DE
LA MAÎTRISE AVEC MÉMOIRE EN GÉNIE DE LA CONSTRUCTION
M. Sc. A.

MONTRÉAL, LE 21 JUILLET 2025

ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
UNIVERSITÉ DU QUÉBEC

PRÉSENTATION DU JURY

CE MÉMOIRE A ÉTÉ ÉVALUÉ

PAR UN JURY COMPOSÉ DE :

M. Richard Arsenault, directeur de mémoire
Département du génie de la construction à l'École de technologie supérieure

M. François Brissette, président du jury
Département du génie de la construction à l'École de technologie supérieure

M. Jean-Luc Martel, membre du jury
Département du génie de la construction à l'École de technologie supérieure

IL A FAIT L'OBJET D'UNE SOUTENANCE DEVANT JURY ET PUBLIC

LE 14 JUILLET 2025

À L'ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

REMERCIEMENTS

Je tiens tout d'abord à remercier mon directeur de maîtrise Richard Arsenault, pour son support, sa patience et ses encouragements tout au long de ce projet. Son enthousiasme contagieux pour le domaine, que j'ai découvert grâce au cours CTN336 où il était mon professeur, a suscité en moi un intérêt pour l'hydrologie. Ce cours a été un moment tournant pour la suite de mon parcours académique. Son expertise, ses conseils et sa disponibilité ont été essentiels dans la réalisation de ce mémoire.

J'aimerais également remercier ma famille, mon copain et mes amis, qui ont été là pour moi à chaque étape de mon parcours académique. Leur patience, leur soutien indéfectible et leur compréhension, même dans les moments de stress où je pouvais parfois être marabout, ont été d'une aide inestimable. Leurs encouragements dans les moments de doutes ou de réussite m'ont permis de garder le cap et de persévérer tout au long de ce projet. Merci à mes parents de m'avoir offert un environnement rempli de soutien, de bonheur et de bienveillance. Cet équilibre m'a permis d'avancer avec confiance et de m'épanouir pleinement.

Un merci tout particulier à mon amie Orianne, sans qui mon parcours aurait probablement été différent. C'est grâce à ses conseils et son soutien que j'ai osé contacter Richard pour explorer les possibilités de projets à la maîtrise. Sa confiance en mes capacités, sa présence et ses encouragements dès le début ont joué un rôle crucial dans la poursuite de mon parcours académique, et je lui en serai toujours reconnaissante.

Analyse de l'impact de l'assimilation de données sur la performance des prévisions hydrologiques à court et moyen terme

Jade LEBEL

RÉSUMÉ

Cette étude présente une analyse de l'impact de l'assimilation de données par le filtre d'ensemble de Kalman sur les performances des prévisions hydrologiques d'ensemble, avec pour objectif principal d'analyser et de quantifier l'impact de l'assimilation de données sur la caractérisation de l'incertitude dans les prévisions hydrologiques. Plus spécifiquement, il s'agit d'évaluer l'impact de l'assimilation de données sur la performance des prévisions hydrologiques à court et moyen terme, à l'aide de métriques telles que le CRPS et les diagrammes de Talagrand.

L'étude a été menée sur le bassin versant du Lac-Saint-Jean à l'aide de trois modèles conceptuels globaux : MOHYSE, Blended et GR4J-CN. La calibration des modèles a été réalisée à l'aide de l'algorithme d'optimisation *Dynamically Dimensioned Search,* tandis que l'assimilation de données a été réalisée à l'aide du filtre d'ensemble de Kalman avec 25 membres. Les données météorologiques utilisées proviennent des réanalyses ERA5 pour la période 1960-2023 et les prévisions météorologiques proviennent de la base de données du Centre Européen pour les Prévisions Météorologiques à Moyen Terme, comprenant 50 membres de 14 jours, couvrant la période 2016-2023. Les prévisions hydrologiques ont été générées sur un horizon de 14 jours pour les trois modèles.

Afin d'évaluer la performance des prévisions hydrologiques, les diagrammes de Talagrand ont été utilisés pour évaluer la fiabilité tandis que le Continuous Ranked Probability Score (CRPS) a été utilisé pour évaluer leur précision. Les résultats obtenus montrent que l'assimilation de données améliore la qualité des prévisions, en particulier pour le modèle GR4J-CN. Cette amélioration se traduit par une réduction des erreurs, une meilleure stabilité des prévisions dans le temps et une fiabilité renforcée. Cependant, l'impact de l'assimilation de données varie selon le modèle. En effet, pour le modèle MOHYSE, les gains sont plus limités et une légère dégradation des performances est parfois observée. Les modèles assimilés parviennent à mieux représenter l'incertitude en couvrant plus efficacement les observations dans leurs intervalles de prévision. Cette tendance est observable sur l'ensemble des saisons, quoiqu'elle soit plus prononcée en hiver et en automne.

Mots-clés : prévisions hydrologiques d'ensemble, modélisation hydrologique, assimilation de données, incertitude, sous-dispersion

Analysis of the impact of data assimilation on the performance of short- and medium-term hydrological forecasts

Jade LEBEL

ABSTRACT

This study presents an analysis of the impact of data assimilation using the Ensemble Kalman Filter on ensemble hydrological forecast performance. The main objective is to analyze and quantify the impact of data assimilation on the characterization of uncertainty in hydrological forecasts. More specifically, it aims to assess the impact of data assimilation on the performance of short- to medium-term hydrological forecasts, using metrics such as CRPS and Talagrand diagrams.

The study was conducted on the Lac-Saint-Jean watershed using three lumped conceptual hydrological models: MOHYSE, Blended and GR4J-CN. The models were calibrated using the Dynamically Dimensioned Search optimization algorithm. Data assimilation was implemented using the Ensemble Kalman Filter with 25 members. Meteorological data were drawn from the ERA5 reanalysis dataset for the period 1960-2023, while meteorological forecasts were obtained from the European Centre for Medium-Range Weather Forecasts ensemble, consisting of 50 ensemble members over 14 days, covering the period 2016-2023. Hydrological forecasts were generated over a 14 days lead time for all three models.

To assess the performance of hydrological forecasts, Talagrand diagrams were used to evaluate forecast reliability, while the Continuous Ranked Probability Score was used to assess forecast accuracy. The results show that data assimilation improves forecast quality, with the most impact observed for the GR4J-CN model. This improvement is reflected in reduced forecast errors, increased temporal stability and enhanced reliability. However, the impact of data assimilation varies depending on the model. For MOHYSE, the gains are more limited and a slight degradation in performance is sometimes observed. The assimilated models better capture forecast uncertainty by mode effectively encompassing the observations within their prediction intervals. This trend is observed across all season, although it is more pronounced in winter and autumn.

Keywords: Ensemble hydrological forecasting, hydrological modelling, data assimilation, uncertainty, under-dispersion

TABLE DES MATIÈRES

Page

LISTE DES FIGURES

LISTE DES ABRÉVIATIONS, SIGLES ET ACRONYMES

AM	Saison de la fonte de neige – Avril à Mai
CRPS	Continuous ranked probability score (Score de probabilité classé en continu)
DJFM	Saison hivernale – Décembre à Mars
ECCC	Environnement et Changement climatique Canada
ECMWF	European Centre for Medium-Range Weather Forecasts (Centre Européen pour les prévisions météorologiques à moyen terme)
EnFK	Ensemble Kalman Filter (Filtre d'ensemble de Kalman)
ERA5	ECMWF 5th generation ReAnalysis (Réanalyse de 5^{e} génération)
GR4J-CN	Modèle hydrologique Génie rural à 4 paramètres – Journalier, développé par le Centre national du machinisme agricole du génie rural, des eaux et des forêts en France
JJA	Saison d'été – Juin à Août
KGE	Kling-Gupta Efficiency (Efficacité de Kling-Gupta)
LSJ	Lac-Saint-Jean
MOHYSE	Modèle hydrologique simplifié à l'extrême développé par l'Institut national de la recherche scientifique (INRS) de l'Université du Québec
NSE	Nash-Sutcliffe Efficiency (Efficacité de Nash-Sutcliffe)
SON	Saison d'automne – Septembre à Novembre

LISTE DES SYMBOLES ET UNITÉS DE MESURE

Aire	
km^2	kilomètre carré
Débit	
m^3/s	mètre cube par seconde

INTRODUCTION

L'hydrologie est une science qui étudie le cycle de l'eau, depuis les précipitations jusqu'à leur écoulement dans les cours d'eau et leur infiltration dans les sols. Elle étudie la répartition des précipitations sur un bassin versant et leur contribution aux différents processus hydrologiques. Ainsi, la compréhension de l'hydrologie est essentielle pour assurer une gestion efficace des ressources en eau, tant pour répondre aux besoins humains et environnementaux que pour soutenir les activités industrielles.

Cependant, le cycle de l'eau est de plus en plus perturbé par les changements climatiques, qui affectent la disponibilité des ressources en eau. Ainsi, les défis posés par les changements climatiques rendent la gestion efficace des ressources en eau cruciale, notamment pour anticiper et gérer les évènements extrêmes tels que les inondations et les sécheresses. Comme ces phénomènes seront de plus en plus fréquents et sévères, il est essentiel d'avoir des prévisions hydrologiques précises et fiables (Arnell et Gosling, 2013).

La modélisation hydrologique constitue l'un des principaux outils mis à la disposition des hydrologues pour simuler les processus hydrologiques et pour fournir une estimation des débits futurs (Goodarzi, Niknam & Sabaghzadeh, 2022). En effet, les modèles hydrologiques sont calibrés à partir de données hydrométéorologiques historiques, puis ils sont utilisés avec des prévisions météorologiques pour simuler le débit à un endroit dans le bassin versant. Ainsi, ces prévisions hydrologiques permettent d'orienter les décisions opérationnelles telles que la gestion des réservoirs, la prévision des crues ou la planification des ressources en eau.

Cependant, la performance des modèles hydrologiques reste limitée par les nombreuses incertitudes, tant dans les données d'entrées que dans les processus simulés. En conséquence, les prévisions hydrologiques issues de ces modèles comportent elles aussi des incertitudes. Ainsi, pour mieux gérer ces incertitudes, il est nécessaire de les quantifier et, dans la mesure du possible, de les réduire à un niveau optimal.

L’assimilation de données est une technique utilisée pour améliorer les prévisions hydrologiques et minimiser leurs biais liés aux conditions initiales. Cette technique consiste à intégrer des observations dans les modèles hydrologiques afin de corriger leurs états initiaux et d’améliorer la précision des prévisions. C’est dans ce contexte que s’inscrit cette étude, qui vise à explorer comment l’assimilation de données peut influencer la fiabilité et la précision des prévisions hydrologiques d’ensemble. Cette étude est réalisée sur le bassin versant du Lac-Saint-Jean. Pour ce faire, trois modèles hydrologiques sont utilisés afin de comparer leurs performances et de mieux comprendre l’impact de l’assimilation de données par le filtre d’ensemble de Kalman sur les prévisions.

Ce mémoire est structuré en six chapitres afin de présenter les différentes étapes de l’étude. Le premier chapitre présente une revue de la littérature portant sur les différentes prévisions, l’assimilation de données et l’incertitude liée aux prévisions hydrométéorologiques. Le deuxième chapitre détaille la méthodologie utilisée, incluant le choix des modèles hydrologiques, leur calibration, l’assimilation de données et l’évaluation de la performance des prévisions. Les résultats obtenus sont quant à eux présentés dans le chapitre trois. Le quatrième chapitre est défié à une discussion, où les résultats sont analysés et interprétés. Le cinquième chapitre présente une conclusion qui synthétise les principaux constats, tandis que le sixième chapitre est consacré aux recommandations.

CHAPITRE 1

REVUE DE LA LITTÉRATURE

Ce chapitre a pour objectif de présenter la littérature sur laquelle cette s'étude s'appuie et de présenter comment elle s'intègre dans la littérature existante. Il aborde les prévisions météorologiques, le processus de modélisation hydrologique, les prévisions hydrologiques, l'assimilation de données et les sources d'incertitudes liées aux prévisions hydrologiques. Puis, les principaux objectifs de l'étude y sont également présentés.

1.1 Prévisions météorologiques

Les prévisions météorologiques sont essentielles pour alimenter les modèles hydrologiques puisqu'elles fournissent les principaux forçages nécessaires aux simulations. Dans le cas des modèles hydrologiques simples, elles se limitent de manière générale aux prévisions de précipitations et de températures. Quant aux modèles plus complexes, ils intègrent également des variables telles que le vent, l'humidité de l'air ou encore le rayonnement solaire.

La précision et la fiabilité des prévisions météorologiques influencent celles des prévisions hydrologiques, car les biais présents dans les prévisions météorologiques se répercutent sur les prévisions hydrologiques. En effet, les incertitudes liées aux prévisions météorologiques s'additionnent à celles des modèles hydrologiques, ce qui amplifie les écarts entre les prévisions hydrologiques et les observations (Baran, Hemri & El Ayari, 2019). De plus, ces prévisions peuvent être déterministes ou probabilistes. *L'European Centre for Medium-Range Weather Forecasts* (ECMWF) et Environnement et Changement climatique Canada (ECCC) sont parmi les principaux centres météorologiques qui produisent quotidiennement des prévisions sur différentes échelles spatiales et temporelles. Ils jouent un rôle important en fournissant des prévisions météorologiques les plus précises et fiables possibles. En effet, ils simulent et émettent régulièrement de nouvelles prévisions météorologiques sur différents horizons, allant du court terme, c'est-à-dire 1 à 48 heures, au moyen terme, c'est-à-dire 3 à 15 jours. Ces prévisions sont obtenues à partir de modèles météorologiques qui simulent l'état de

l'atmosphère terrestre en fonction de conditions initiales telles que la pression atmosphérique, l'intensité du rayonnement solaire ou l'humidité spécifique et relative.

Cependant, des incertitudes persistent dans ces prévisions météorologiques, et il reste difficile de déterminer l'origine de tous ces biais et de les caractériser (Leutbecher et al., 2017). Certaines incertitudes peuvent provenir des imperfections des modèles météorologiques eux-mêmes (Leutbecher et al., 2017), des erreurs dans les conditions initiales (Roundy, Duan & Schaake, 2019) ou des limitations dans la représentation des processus physiques (Meng et al., 2024). Pour atténuer ces incertitudes, les techniques de post-traitement jouent un rôle important (Meng et al., 2024). Parmi ces techniques, on retrouve notamment la réduction d'échelle et la correction des biais, la cartographie des quantiles et la cartographie de la distribution, qui sont grandement utilisées pour améliorer la précision des prévisions météorologiques (Meng et al., 2024).

1.1.1 Prévisions déterministes

Les prévisions déterministes utilisent les données météorologiques telles que les précipitations et la température, issues d'un seul modèle météorologique exécuté à la résolution de grille la plus élevée réalisable en termes de capacité de calcul (Pappenberger, Thielen & Del Medico, 2011). Pour ce faire, la meilleure estimation des conditions initiales est utilisée (Pappenberger et al., 2011). Cependant, ces prévisions présentent des limites, car elles ne prennent pas en compte l'incertitude et donc ne présentent pas les différents scénarios possibles qui pourraient survenir (Schaake, Hamill, Buizza & Clark, 2007). Par ailleurs, la nature chaotique de l'atmosphère entraine une amplification des erreurs dans les prévisions météorologiques déterministes (Hamill, 2001). De plus, étant donné que la précision et fiabilité des prévisions déterministes dépendent de la qualité des conditions initiales, leur utilisation devient moins adaptée lorsque ces dernières sont incertaines. (Schwanenberg et al., 2015). En revanche, ces dernières ont l'avantage d'exiger moins de temps de calcul et d'être disponibles en meilleure résolution que les prévisions probabilistes (Demeritt et al., 2007).

1.1.2 Prévisions d'ensemble

Les prévisions d'ensembles météorologiques offrent une alternative aux prévisions déterministes en tenant compte de l'incertitude. Contrairement aux prévisions déterministes, qui produisent un seul scénario, les prévisions d'ensembles s'appuient sur un ensemble de scénarios météorologiques équiprobables, obtenus en générant plusieurs simulations où les conditions initiales du modèle météorologique sont légèrement perturbées selon l'incertitude liée aux observations pour obtenir les autres membres de l'ensemble. Cette approche permet d'améliorer l'estimation des probabilités des évènements météorologiques, notamment les phénomènes extrêmes, tels que les fortes précipitations ou les vagues de chaleur (Wetterhall, Pappenberger, Alfieri, Cloke & Thielen, 2014). De plus, générer un ensemble de prévision plutôt qu'une seule prévision déterministe permet de quantifier l'incertitude associée aux prévisions et ainsi évaluer plus précisément les risques liés aux conditions météorologiques futures. (Demeritt et al., 2007). Une étude de McCollor et Stull (2008) qui présente une évaluation d'ensemble des précipitations à court terme (soit 24 heures) montre que l'utilisation de l'ensemble complet des membres ou la moyenne de l'ensemble tend à se rapprocher davantage des observations qu'avec l'utilisation d'un seul modèle déterministe.

1.2 Modélisation hydrologique

Un modèle hydrologique est un outil essentiel pour simuler et prévoir les débits dans un bassin versant. Un modèle hydrologique appliqué en mode simulation vise à reconstituer les débits passés d'un cours d'eau à partir de données hydrométéorologiques historiques, tandis que le modèle appliqué en mode prévision cherche à estimer les débits futurs à partir de prévisions météorologiques.

Les principaux intrants d'un modèle hydrologique simple sont les précipitations et la température de l'air, en plus de certains descripteurs physiques comme la superficie du bassin versant ou son emplacement géographique pour estimer le rayonnement solaire. Cependant, certains modèles plus complexes peuvent prendre en compte d'autres variables telles que l'évapotranspiration, l'humidité du sol et la couverture du sol. Les modèles hydrologiques

représentent les processus hydrologiques de différentes manières, bien qu'ils correspondent toujours à une simplification d'un système complexe. En effet, ils peuvent être classés en trois grandes catégories : modèles à base physique, modèles empiriques et modèles conceptuels. Un modèle hydrologique à base physique tente de reproduire les processus hydrologiques en s'appuyant sur des équations établies par des lois de la physique telles que la loi de Darcy, l'équation de Richards et l'équation de Saint-Venant. Cependant, ces modèles nécessitent une grande quantité de données et sont plus complexes que les modèles conceptuels. De plus, leur temps de calcul est plus long puisque l'augmentation du nombre de processus modélisés entraine une augmentation des résolutions spatiales et temporelles, ce qui accroît le temps de calcul (Fatichi et al., 2016). Les modèles empiriques reposent sur des relations mathématiques et sont basés sur les relations statistiques entre les variables hydrologiques. Quant aux modèles hydrologiques conceptuels, ils sont basés sur des réservoirs et des équations simplifiées pour représenter les processus hydrologiques. Ces modèles utilisent des réservoirs théoriques et des paramètres ajustables pour reproduire les observations, bien que ces paramètres n'aient pas nécessairement de signification physique. Leur principal avantage réside dans leur simplicité. En effet, ils nécessitent peu de données et ont un temps de calcul réduit tout en offrant une bonne capacité des prévisions à grande échelle et en nécessitant un seuil plus bas de connaissances des processus physiques (Fatichi et al., 2016).

Qui plus est, un modèle peut être global, semi-distribué ou distribué. Dans un modèle global, le bassin versant est considéré comme une entité homogène, où les caractéristiques, telles que la pente du bassin versant, sont uniformisées, souvent en utilisant la valeur moyenne. Ainsi, les processus hydrologiques sont considérés comme uniformes sur tout le bassin versant. D'un autre côté, un modèle distribué ou semi-distribué divise le bassin versant en plusieurs sous bassins versants ou sous forme de grille afin de simuler les processus à plus petite échelle. Ces modèles permettent d'observer des effets localisés, comme les variations de ruissellement dues aux différences de sol ou de végétation. Toutefois, ils requièrent une plus grande quantité de données, car il faut des données pour chaque grille ou sous-bassin.

1.3 Prévisions hydrologiques

Les prévisions hydrologiques jouent un rôle essentiel dans la gestion des ressources en eau et la prévention des évènements extrêmes (Schaake et al., 2007). Des modèles hydrologiques sont utilisés pour effectuer les prévisions hydrologiques. Elles consistent à estimer les débits futurs en se basant sur des données météorologiques. Tout comme les prévisions météorologiques, les prévisions hydrologiques peuvent être déterministes ou ensemblistes.

Les prévisions déterministes génèrent une seule valeur de débit basé sur la meilleure estimation des conditions initiales et sur une seule simulation du modèle hydrologique. Les prévisions déterministes ont un temps de calcul moindre que les prévisions d'ensembles (Demeritt et al., 2007). Cependant, les prévisions déterministes créent une illusion de certitude, car elles ne fournissent qu'une seule valeur de prévision, ce qui peut induire les utilisateurs en erreur en leur donnant l'impression que les résultats sont précis et fiables (Krzysztofowicz, 2001). Elles ne tiennent pas compte de tous les débits possibles, ainsi l'incertitude associée à la valeur prédite n'est pas reflétée (Demargne et al., 2014).

Les prévisions d'ensembles quant à elles reposent sur un ensemble de scénarios possibles. Un ensemble de prévisions comprend un certain nombre de membres, chacun correspondant à une simulation distincte. Cette approche permet de prendre en compte l'incertitude et aide à la prise décisionnelle. En effet, elles permettent d'attribuer un certain niveau de confiance aux prévisions et d'évaluer les risques économiques. L'étude de Montanari et Grossi (2008) a démontré qu'un système de prévision ensembliste offre une valeur économique supérieure à celle d'un modèle de prévision unique ou d'une prévision basée uniquement sur la moyenne de l'ensemble. De plus, les prévisions d'ensemble permettent de mieux représenter et quantifier les évènements extrêmes (Wetterhall et al., 2014). De manière générale, la moyenne des membres d'une prévision d'ensemble peut offrir une performance supérieure à celle d'une prévision déterministe (McCollor & Stull, 2008; Wetterhall et al., 2013). Toutefois, les prévisions hydrologiques d'ensembles comportent certaines limites telles que la sous-dispersion et les biais (Hemri, Fundel & Zappa, 2013). En effet, la sous-estimation de la

variabilité du climat et des états de l'atmosphère (reflétés dans les prévisions météorologiques d'ensemble), de la structure des modèles hydrologiques et de l'incertitude liée à leurs conditions initiales engendre fréquemment des prévisions sous-dispersées. La sous-dispersion survient lorsque la variabilité des membres de l'ensemble est trop faible, ce qui entraine une sous-estimation de l'incertitude et une tendance des observations à se situer en dehors des limites prévues par l'ensemble, ce qui réduit la fiabilité de l'intervalle de confiance. Les biais indiquent que les prévisions hydrologiques d'ensembles ne représentent pas adéquatement la magnitude des observations et que ces dernières sont soit plus élevées (biais négatif) ou plus basses (biais positifs) que les membres des prévisions probabilistes.

1.4 Fonctions-objectif pour le calage des modèles

Le calage des modèles est une étape essentielle permettant d'ajuster les paramètres du modèle afin de produire des estimations de débits fidèles aux observations. Pour ce faire, un calage manuel peut être effectué ou un algorithme d'optimisation peut être utilisé. Cependant, lorsque les modèles sont complexes et comportent de nombreux paramètres, le calage manuel devient difficile et risque de retourner un minimum local de moins bonne qualité (Arsenault, Poulin, Côté et Brissette, 2013). Ainsi, l'utilisation d'un algorithme d'optimisation est souvent privilégiée. Des méthodes telles que le *Dynamically Dimensioned Search* (DDS), *le Genetic Algorithm* (GA), *le Covariance Matrix adaptation Evolution Strategy (CMAES)* et le *Pattern Search* (PS) effectuent plusieurs itérations afin de générer différents jeux de paramètres et de comparer les débits simulés aux débits observés. L'objectif de ces algorithmes est d'optimiser une fonction-objectif, qui permet de quantifier l'écart entre les débits simulés et observés afin d'obtenir la meilleure adéquation possible entre le modèle et les données réelles.

Le critère de Nash-Sutcliffe (NSE) est l'une des fonctions-objectif les plus répandues pour évaluer la performance des modèles hydrologiques (Gupta, Kling, Yilmaz & Martinez, 2009). L'équation (1.1) présente le calcul du score de NSE, où n représente le nombre de pas de temps, t le pas de temps, $x_{s,t}$ le débit simulé au pas de temps t, $x_{0,t}$ le débit observé au temps t et μ_0 la moyenne des débits observés.

$$\text{NSE} = 1 - \frac{\sum_{t=1}^{n}(x_{0,t} - x_{s,t})^2}{\sum_{t=1}^{n}(x_{0,t} - \mu_0)^2} \quad (1.1)$$

Un NSE de 1 indique une performance parfaite du modèle tandis qu'une valeur négative signifie que le modèle performe moins bien que si la moyenne des observations avait été utilisée comme prédicteur.

Une autre fonction-objectif couramment utilisée en hydrologie est la racine de l'erreur quadratique moyenne (RMSE). L'équation (1.2) présente le calcul du RMSE, où n représente le nombre de pas de temps, t le pas de temps, $x_{s,t}$ le débit simulé au pas de temps t et $x_{0,t}$ le débit observé au temps t.

$$\text{RMSE} = \frac{\sum_{t=1}^{n}(x_{s,t} - x_{0,t})^2}{n} \quad (1.2)$$

L'une des principales limites du NSE est son utilisation de la moyenne observée comme référence, ce qui peut entrainer une surestimation des performances du modèle, en particulier pour des variables influencées par une forte saisonnalité comme le ruissellement dans les bassins due à la fonte des neiges (Gupta et al., 2009). Ainsi, pour pallier cette limitation, Kling et Gupta (2009) ont proposé une alternative, soit le coefficient de Kling-Gupta (KGE), qui prend en compte le biais moyen et la variabilité relative des débits. L'équation (1.3) montre le calcul du KGE, où r représente la corrélation entre les débits observés et simulés, α le ratio des écarts-types et β le biais. L'équation (1.4) présente la formule pour alpha, où σ_s représente l'écart-type des débits simulés et σ_o l'écart-type des débits observés. L'équation (1.5) présente l'équation pour beta, où μ_s représente la moyenne de la série temporelle simulée et μ_o représente la moyenne de la série temporelle observée.

$$\text{KGE} = 1 - \sqrt{(r-1)^2 + (\alpha - 1)^2 + (\beta - 1)^2} \quad (1.3)$$

$$\alpha = \frac{\sigma_s}{\sigma_o} \quad (1.4)$$

$$\beta = \frac{\mu_s}{\mu_o} \quad (1.5)$$

Toutefois, il n'y a pas de fonction-objectif unique qui permet de calibrer un modèle hydrologique de manière optimale pour représenter simultanément toutes les caractéristiques de la forme de l'hydrogramme (Jie, Chen, Xu, Zeng, & Tao, 2016). Il est donc essentiel d'adapter le choix de la fonction-objectif en fonction des objectifs de la modélisation et de l'étude.

1.5 Incertitudes liées aux prévisions hydrologiques

Les prévisions hydrologiques comprennent plusieurs sources d'incertitudes qui influencent leur fiabilité et leur précision. Ces incertitudes peuvent provenir soit des données d'entrées, des conditions initiales, de la structure du modèle hydrologique ou de l'estimation des paramètres (Mazrooei et Sankarasubramanian, 2019). L'une des principales sources d'incertitude des prévisions hydrologiques se trouve dans les prévisions météorologiques. Effectivement, comme les modèles météorologiques sont eux-mêmes soumis à des incertitudes, une mauvaise estimation des précipitations ou des températures peut entrainer des écarts entre les débits simulés et observés. De cette manière, l'incertitude des prévisions météorologiques se propage dans le modèle hydrologique lors des prévisions hydrologiques, ce qui amplifie l'incertitude totale (Baran et al., 2019).

Les incertitudes liées à la structure même du modèle proviennent du fait que les modèles hydrologiques sont des représentations simplifiées des processus physiques, et chaque modèle repose sur des hypothèses spécifiques propres à chacun. Par exemple, dans le cas du ruissellement, le comportement réel des sols est souvent plus complexe que ce que les modèles peuvent représenter. De plus, le ruissellement dépend de paramètres physiques empiriques pour lesquels on ne dispose pas toujours de mesures précises. Ainsi, le processus est simplifié, ce qui entraine une incertitude. D'un autre côté, certains modèles qui ne prennent pas en

compte la fonte de neige vont nécessairement sous-estimer les débits au printemps dans un bassin versant nordique. Dans ce cas, il ne s'agit pas d'une incertitude structurelle, mais plutôt d'un choix de modèle inadapté au contexte hydrologique du bassin, puisqu'un processus hydrologique important est complètement omis. De ce fait, ce n'est pas parce qu'un modèle est bon à un endroit qu'il est bon sur tous les bassins versants. Par exemple, certains modèles peuvent être plus performants dans les régions nordiques que dans les régions chaudes et sèches (Kavetski, Kuczera & Franks, 2006).

L'incertitude paramétrique peut provenir de l'interaction entre les paramètres puisque dans la plupart des modèles, les paramètres n'agissent pas indépendamment les uns des autres, mais interagissent de manière complexe. Ainsi, cela complique le processus de calage, car la modification d'un paramètre peut affecter la performance des autres paramètres, ce qui rend plus difficile la recherche d'un ensemble optimal de paramètres. De plus, les paramètres obtenus peuvent varier d'un calage à l'autre, un phénomène connu sous le nom d'équifinalité (Beven & Binley, 1992). La sensibilité des modèles aux paramètres constitue également une source d'incertitude puisque certains modèles sont très sensibles aux variations de paramètres, de sorte qu'un léger changement dans une valeur paramétrique peut entrainer des écarts significatifs dans les prévisions hydrologiques (Ratto et al., 2007).

1.6 Assimilation de données

Comme les prévisions hydrologiques contiennent des incertitudes, il est essentiel de les circonscrire afin d'améliorer la fiabilité des simulations. Pour ce faire, des techniques d'assimilation de données peuvent être utilisées pour réduire l'incertitude et améliorer les prévisions hydrologiques (Mazrooei & Sankarasubramanian, 2019). L'assimilation de données consiste à diminuer l'écart présent entre les états initiaux du modèle et le débit observé en modifiant les conditions initiales à chaque début de prévisions hydrologiques. En améliorant les conditions initiales, cela améliore les performances des prévisions de débits, surtout sur les prévisions à court terme (Piazzi, Thirel, & Perrin, 2021). De cette manière, en raison de l'assimilation de données, les modèles simulent des débits qui sont plus proches de la réalité

(Mai, Arsenault, Tolson, Latraverse, & Demeester, 2020). En effet, cela permet de débuter la prévision hydrologique de manière plus représentative de l'état actuel du bassin versant. Différentes techniques d'assimilation de données existent et peuvent être utilisées selon le cas d'application. L'assimilation manuelle permet à l'hydrologue d'ajuster manuellement les états initiaux du modèle hydrologique en fonction de son expertise (Pagano et al., 2016). Toutefois, cette méthode présente des limites importantes, notamment le fait qu'elle ne soit pas reproductible et que certains bassins versants et modèles nécessitent un trop grand nombre d'assimilations manuelles, ce qui peut dénaturer les bilans d'eau sur le bassin. Ainsi, d'autres méthodes d'assimilation de données automatiques ont été développées, telles que les filtres particulaires (DeChant & Moradkhani, 2011), l'approche bayésienne (Reggiani & Weerts, 2008), le filtre de Kalman (Kalman, 1960) et le filtre d'ensemble de Kalman (EnKF; Evensen, 1994). Ce dernier est l'une des techniques les plus couramment utilisées en hydrologie en raison de sa robustesse et sa facilité d'implémentation (Liu et al., 2012), bien que certains défis persistent (Thiboult & Anctil, 2015).

L'EnKF est une méthode qui prend en compte les incertitudes inhérentes aux observations et aux intrants utilisés telles que les précipitations et la température. Étant donné que ces données contiennent inévitablement des erreurs liées aux instruments de mesure, l'EnKF estime et échantillonne cette incertitude et y ajoute une variabilité correspondant aux incertitudes estimées. L'échantillonnage des séries météorologiques parmi la distribution proposée est effectué de telle sorte à créer entre 25 et 50 membres, ce qui retourne un vecteur de 25 à 50 membres pour chaque variable (précipitation, température, etc.). En alimentant ces séries météorologiques dans le modèle hydrologique, ce dernier génère ainsi un ensemble d'états hydrologiques perturbés. L'algorithme EnKF utilise ces états et débits simulés pour estimer de nouveaux états initiaux plus représentatifs de la distribution des débits observés. Le filtre est appliqué séquentiellement dans le temps, permettant d'ajuster progressivement les simulations pour mieux correspondre aux observations.

Toutefois, l'EnKF est assujetti à des limitations. Une première contrainte est de caractériser l'incertitude de la station hydrométrique. En effet, cette incertitude varie dans le temps selon

l'amplitude des débits, un phénomène connu sous le nom d'hétéroscédasticité. Une autre contrainte est que l'EnKF peut compromettre la fiabilité des prévisions en ayant une trop grande confiance dans l'exactitude des conditions initiales (DeChant & Morakhani, 2012). Enfin, l'ajout de membres requiert un temps de calcul plus long et une puissance de calcul plus importante. Malgré ces limitations, l'EnKF reste une méthode largement utilisée et reconnue pour son efficacité dans la réduction des biais entre les simulations et les observations dans la plupart des études effectuées (Bergeron, Trudel & Leconte, 2016; Maxwell, Jackson & McGregor, 2018; Thiboult & Anctil, 2015).

En somme, bien que plusieurs études aient exploré l'assimilation de données, très peu l'ont appliquée à l'intérieur d'une chaine complète de prévision incluant à la fois les prévisions météorologiques, l'assimilation de données et plusieurs modèles hydrologiques de complexité différente. De plus, les comparaisons directes entre divers modèles avec et sans assimilation de données restent rares. Ainsi, ce projet vise à pallier ce manque en évaluant concrètement l'impact de l'assimilation de données sur la performance des prévisions hydrologiques dans une chaine de prévision complète.

1.7 Objectifs de recherche

L'objectif principal de cette étude est d'analyser et de quantifier l'impact de l'assimilation de données par le filtre d'ensemble de Kalman sur la caractérisation de l'incertitude dans les prévisions hydrologiques. Plus spécifiquement, l'objectif est d'évaluer dans quelle mesure l'assimilation de données par le filtre d'ensemble de Kalman permet d'améliorer la performance des prévisions hydrologiques. En considérant les différentes sources d'incertitudes qui peuvent affecter la modélisation hydrologique, cette étude cherche à mesurer l'incertitude liée aux conditions initiales et à déterminer son influence sur la qualité des prévisions. En comparant les prévisions réalisées avec et sans assimilation de données, il sera possible de quantifier l'impact de la correction des états initiaux sur la précision et la fiabilité des prévisions hydrologiques. Pour ce faire, différentes métriques de performance seront utilisées pour analyser les résultats. Un autre objectif secondaire de ce projet consiste à tester

les capacités de la plateforme PAVICS-Hydro, en réalisant l'ensemble du projet, soit de l'acquisition des données à l'analyse des résultats, sur la plateforme. Ainsi, le but est de favoriser la reproductibilité du processus, en veillant à ce que le code soit bien documenté et que l'approche puisse être facilement partagée ou réutilisée par d'autres utilisateurs.

CHAPITRE 2

MÉTHODOLOGIE

Ce chapitre présente la méthodologie employée pour mener cette étude sur le bassin versant du Lac-Saint-Jean. La région à l'étude et ses principales caractéristiques sont présentées. Ensuite, les différents modèles hydrologiques utilisés et leur processus de calibration sont détaillés. Enfin, les méthodes d'assimilation ainsi que les mesures de performance des prévisions sont présentées.

2.1 Secteur à l'étude

Cette étude se concentre sur le bassin versant du Lac-Saint-Jean (LSJ), situé dans la province de Québec, au Canada. Ce bassin couvre une superficie d'environ 45 000 km^2 et est d'une grande importance pour l'entreprise Rio Tinto, qui y exploite les ressources hydriques pour la production d'aluminium par le biais de l'hydroélectricité. La figure 2.1 présente la carte du bassin versant à l'étude. Le climat de cette région est caractérisé par de grandes variations de température entre l'été et l'hiver. Ainsi, on retrouvera une saisonnalité marquée. Les hivers sont rigoureux et caractérisés par une accumulation significative de neige, ce qui engendre, au printemps, une fonte des neiges avec une crue printanière importante. Ces crues ont un impact sur la gestion des réservoirs en raison des grands volumes d'eau qui y affluent pendant cette période. Elles compliquent la gestion des réservoirs, car les hydrologues doivent trouver un équilibre entre maintenir des niveaux d'eau suffisants pour assurer une production optimale dans les centrales hydroélectriques et éviter les débordements. Ainsi, la gestion efficace des réservoirs repose sur des prévisions hydrologiques précises et fiables.

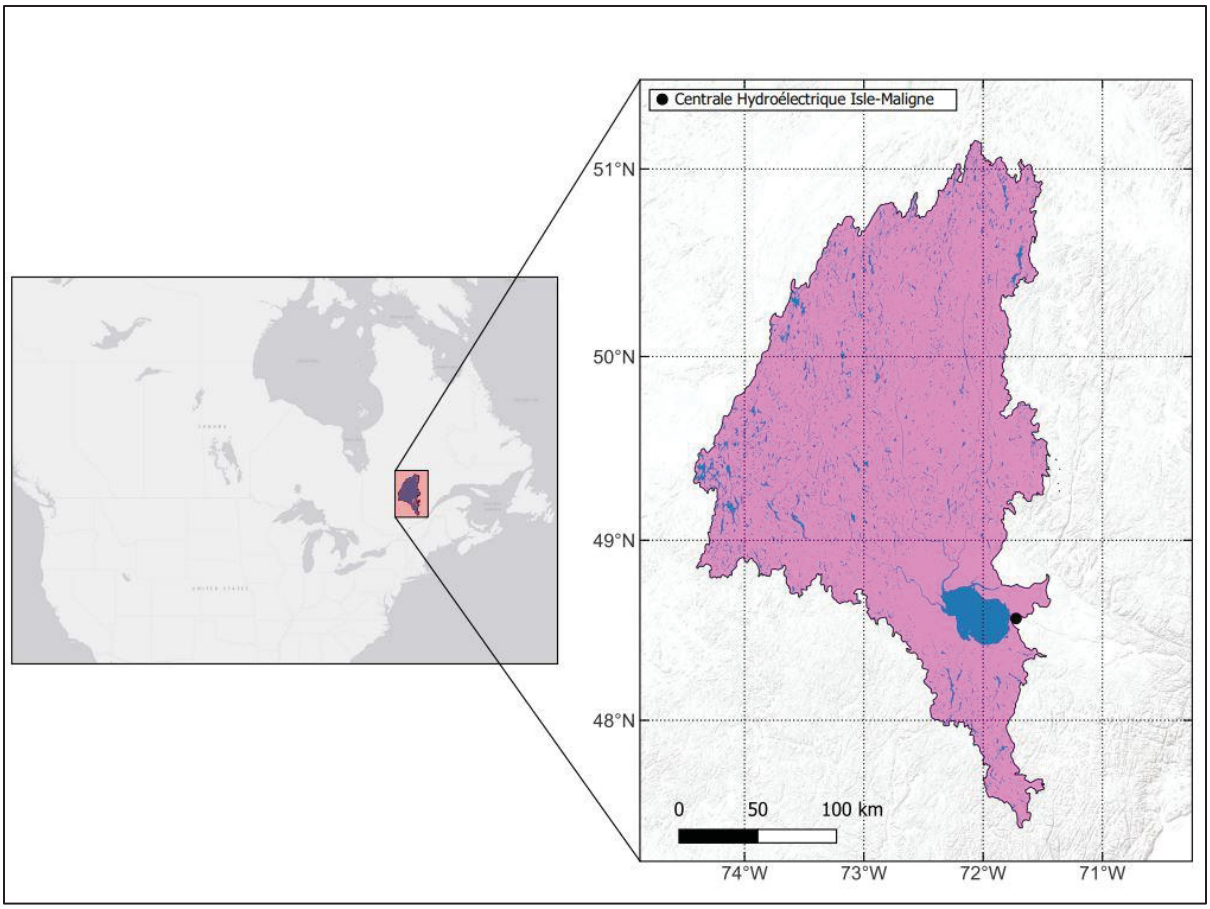

Figure 2.1 Localisation du bassin versant à l'étude

2.2 Jeux de données

Les données météorologiques proviennent des réanalyses ERA5 (Fifth generation of the European ReAnalysis) (Hersbach et al., 2018) et couvrent la période de 1960 à 2023. La réanalyse ERA5 a été utilisée, car elle assure des séries temporelles complètes sur une résolution spatiale de 0.25° x 0.25°, sans données manquantes, pour une panoplie de variables météorologiques. De plus, il a été démontré que ERA5 performe au même niveau que les observations pour le secteur à l'étude (Tarek et al. 2020). Les variables météorologiques requises pour les trois modèles hydrologiques de cette étude sont la température minimale, la température maximale et les précipitations totales. Ces données ont été utilisées pour préparer l'assimilation de données à l'aide du bruitage, mais aussi pour le calage du modèle

hydrologique et pour alimenter la période de chauffe des modèles avant de procéder à la prévision hydrologique.

Les séries temporelles des débits observés proviennent du partenaire industriel Rio Tinto. Ces données, reconstruites par bilan de masse au niveau du réservoir, excluent les débits de la rivière Péribonka, lesquels sont entièrement contrôlés et qui peuvent donc être isolés et ignorés. Les données d'observation sont disponibles de janvier 1950 à mars 2024.

Les données de prévision météorologique utilisées dans cette étude ont été extraites de la base de données du Centre Européen pour les Prévisions Météorologiques à Moyen Terme (ECMWF). Les prévisions se composent de 50 membres avec un horizon de prévision de 14 jours et une résolution de 18 km. Elles couvrent la période allant de mars 2016 à juin 2023. Les variables extraites incluent les précipitations ainsi que les températures minimales et maximales journalières. Il est à noter que pour alimenter les modèles conceptuels globaux de cette étude, la moyenne des points de grille de prévision sur le bassin versant, initialement à une résolution de 0.2° x 0.2°, a été utilisée.

2.3 Description des modèles hydrologiques

Trois modèles hydrologiques ont été retenus pour cette étude. Les détails de chaque modèle sont présentés dans cette section. Il est important de souligner que ces modèles ont été sélectionnés parmi les huit modèles proposés dans la plateforme PAVICS-Hydro (Arsenault et al. 2023), qui a été utilisée pour générer l'ensemble des simulations et analyses dans cette étude. PAVICS-Hydro est un environnement de modélisation et d'analyse hydrométéorologique permettant l'accès à des données climatiques et des données hydroclimatiques ainsi que l'exécution de simulations et de prévisions hydrologiques. Il s'agit de l'une des premières études réalisées entièrement sur la plateforme PAVICS-Hydro.
Les scripts, sous format de Jupyter Notebooks, sont présentés à l'annexe I.

2.3.1 GR4J-CN

GR4J est un modèle hydrologique conceptuel global développé pour simuler les débits journaliers à l'exutoire d'un bassin versant. Il est basé sur deux réservoirs, soit un réservoir de stockage et un réservoir de routage, ainsi que quatre paramètres. Une version améliorée du modèle, développé par Valéry (2010), intègre le module Cema-Neige (CN) afin de prendre en compte l'accumulation et la fonte de la neige dans les bassins versants qui sont soumis à des conditions hivernales. En effet, ce modèle permet de représenter le manteau neigeux en simulant l'accumulation, la couverture neigeuse et la fonte de neige. Cette composante ajoute un stock de neige, qui permet d'accumuler les précipitations solides et de les libérer sous forme liquide lors de la fonte. La fonte est contrôlée par un facteur degré-jour et tandis qu'un coefficient d'inertie thermique régule la capacité thermique du manteau neigeux. Ainsi, l'eau de fonte rejoint les précipitations liquides pour alimenter le réservoir de production. Le réservoir de production capte et distribue les précipitations et l'eau de fonte vers des flux d'évapotranspiration, de pluie nette et de ruissellement, en fonction de son état de saturation, défini par le paramètre X1. Les flux de pluie nette et de ruissellement sont ensuite transformés par un hydrogramme unitaire, dont la réponse dépend du paramètre X4, avant d'être divisés en deux flux et dirigés vers le réservoir de routage et le cours d'eau. Le réservoir de routage, quant à lui, module les écoulements et simule les échanges souterrains en fonction de la capacité du réservoir, défini par le paramètre X3. Des échanges souterrains, gouvernés par le paramètre X2, influencent la redistribution des flux et modulent le débit du bassin. Pour ce qui est de l'évapotranspiration potentielle, elle a été calculée à l'aide de la formule d'Oudin (Oudin et al., 2005). Enfin, le débit total simulé est obtenu en faisant la somme des écoulements issus du réservoir de routage et les flux souterrains. La version de GR4J-CN employée dans cette étude diffère légèrement de celle couramment utilisée puisqu'elle a été adaptée pour fonctionner sur la plateforme de modélisation de PAVICS-Hydro.

2.3.2 MOHYSE

MOHYSE (MOdèle HYdrologique Simplifié à l'Extrême) est un modèle hydrologique conceptuel global développé par Fortin et Turcotte (2007) à des fins pédagogiques. Nonobstant

cela, il a été utilisé à maintes reprises dans des projets de recherche (Dion, Martel & Arsenault, 2021; Arsenault et al., 2013; Armstrong et al., 2024). Il utilise trois intrants météorologiques, soit la température moyenne, les précipitations liquides et les précipitations solides, pour simuler les débits à l'exutoire en tenant compte de la fonte, des infiltrations, de l'évapotranspiration et des écoulements. Le modèle repose sur une représentation simplifiée des processus hydrologiques en différents compartiments. Ces compartiments comprennent le stock de neige, la zone vadose, l'aquifère et un réservoir de transformation basé sur un hydrographe unitaire. Pour ce qui est du stock de neige, il accumule les précipitations solides et libère de l'eau par fonte lorsque la température dépasse un seuil critique, défini par un modèle degré-jour. Cette eau de fonte s'ajoute aux précipitations liquides et peut ensuite s'évaporer, s'infiltrer ou ruisseler. Les précipitations liquides, quant à elles, atteignent directement le sol. L'eau qui s'infiltre se retrouve dans le compartiment de la zone vadose, où elle est par la suite soit évapotranspirée, soit transférée dans l'aquifère, soit drainée vers le cours d'eau par écoulement hypodermique. L'eau qui atteint l'aquifère s'écoule ensuite gravitairement vers le cours d'eau. L'évapotranspiration potentielle a été déterminée selon la formule propre à MOHYSE qui est basée sur le rayonnement solaire. Ainsi, la production correspond à la somme de ces trois composantes. Enfin, le réservoir de transformation applique un hydrogramme unitaire basé sur une distribution gamma afin de convertir cette production en débit à l'exutoire du bassin versant, en tenant compte du temps de transfert de l'eau à travers le bassin versant. Ce modèle contient 10 paramètres qui doivent être calés, dont le coefficient d'ajustement de l'évapotranspiration potentielle, le coefficient d'ajustement de la transpiration, le taux de fonte, la température seuil pour la fonte, trois paramètres pour la vidange des différents compartiments, ainsi que deux paramètres qui déterminent la forme et l'échelle de l'hydrogramme unitaire. Comme il a été spécifiquement pensé pour des climats nordiques, son application au projet est adéquate.

2.3.3 BLENDED

Blended est un modèle hydrologique conceptuel développé par Mai et al. (2020) qui repose sur une approche permettant d'intégrer plusieurs choix d'algorithmes pour représenter des

processus hydrologiques au sein d'un même modèle. Contrairement aux modèles plus traditionnels qui utilisent une seule formule pour chaque processus, Blended permet de combiner et de pondérer différentes formules pour un même processus. La configuration du modèle Blended comprend cinq processus, soit l'infiltration, l'écoulement rapide, l'écoulement de base, l'évapotranspiration et le bilan de neige. Les pondérations peuvent être ajustées avec le calage ou d'autres méthodes sans changer sa structure, ce qui en fait un modèle flexible. Les pondérations peuvent être fournies sous deux formes : directement sous forme de fractions qui totalisent un ou sous forme d'un système de génération de poids aléatoires basé sur la méthode de partage de tarte (pie share; Mai, Craig & Tolson, 2024), qui permet de générer des distributions indépendantes. L'évapotranspiration potentielle est gérée par la structure interne du modèle, qui pondère les sorties de plusieurs formulations selon les paramètres établis lors du calage. Le modèle contient 42 paramètres qui peuvent être des paramètres associés aux algorithmes des processus ou des paramètres générateurs de poids.

2.4 Calage des modèles hydrologiques

Le calage des modèles hydrologiques a été réalisé afin d'optimiser leurs paramètres. L'algorithme *Dynamically Dimensioned Search* (DDS) a été choisi pour calibrer les modèles en raison de son efficacité démontrée dans l'étude de Arsenault et al. (2013). De plus, il est particulièrement adapté en contexte opérationnel et ne nécessite aucun réglage des paramètres. Le coefficient de Nash-Sutcliffe (NSE) a été retenu comme métrique de performance pour le processus de calage. Le calage a été réalisé sur une période allant de 1960 à 1990. Pour assurer une exploration adéquate de l'espace des paramètres, 5 000 évaluations du modèle ont été réalisées avec l'algorithme DDS. Ce nombre d'itérations a été jugé suffisant pour obtenir des paramètres optimaux tout en maintenant un coût computationnel raisonnable.

2.5 Assimilation de données

L'assimilation de données a été effectuée à l'aide du filtre d'ensemble de Kalman (EnKF). Dans un premier temps, les données météorologiques observées (précipitations, températures maximales et minimales) ont été perturbées afin de représenter l'incertitude associée aux

observations. Pour les précipitations, une perturbation multiplicative issue d'une distribution uniforme variant entre 0,8 et 1,2 a été utilisée. Pour les températures maximales et minimales, une erreur additive suivant une distribution normale de moyenne 0 et d'écart-type de 2 a été appliquée. Ces valeurs de perturbation ont été définies par essais-erreurs en vérifiant la performance des modèles sur les prévisions obtenues. Finalement, 25 scénarios ont été échantillonnés à partir de ces distributions. Avant d'appliquer l'assimilation de données, une période de chauffe (*spin-up)* de trois ans a été réalisée pour permettre au modèle hydrologique d'atteindre un état initial réaliste. Cette étape permet d'éviter des conditions initiales arbitraires qui pourraient biaiser les résultats. Une fois l'état initial établi, une assimilation en boucle fermée a été réalisée en avançant dans le temps par incrément de trois jours. Tous les trois jours, les nouvelles observations étaient assimilées, ce qui permet de réajuster les états du modèle. Une fréquence d'assimilation aux trois jours a été choisie puisque l'assimilation quotidienne avait tendance à trop contraindre le modèle, ce qui limitait la capacité du modèle à propager l'incertitude entre deux mises à jour. Plus précisément, l'assimilation de données ciblait les états liés à la teneur en eau des deux premières couches du sol, la surface étant divisée en trois zones possédant des capacités de stockage différentes. Cette procédure a été répétée de manière itérative sur l'ensemble de la période d'étude. Les états des modèles ont été enregistrés pour chacun de ces cas en vue d'effectuer des prévisions hydrologiques dans un second temps. Enfin, l'ensemble des 25 membres générés par l'assimilation de données a été combiné aux 50 membres des prévisions hydrologiques d'ECMWF, produisant ainsi un total de 1 250 membres pour chaque prévision hydrologique.

2.6 Méthodes de prévision

Afin d'évaluer l'impact de l'assimilation de données sur les prévisions hydrologiques, deux scénarios distincts sont utilisés : un scénario *open-loop* (sans assimilation) et un scénario *closed-loop* (avec assimilation). Le scénario *closed-loop* intègre l'assimilation de données avec le filtre d'ensemble de Kalman. À chaque trois jours, les états initiaux des modèles hydrologiques sont mis à jour en assimilant les observations de débit. Les 25 états issus de l'assimilation de données sont alors utilisés comme conditions initiales pour produire les

prévisions hydrologiques, combinées aux 50 membres des prévisions météorologiques ECMWF. Ainsi, chaque prévision hydrologique comporte un total de 1 250 membres (25 états assimilés x 50 membres ECMWF) ce qui permet de mieux représenter l'incertitude associée aux prévisions. Avec le scénario *open-loop*, les modèles hydrologiques utilisent uniquement les données météorologiques observées et les prévisions météorologiques ECMWF pour générer les prévisions hydrologiques. Aucune mise à jour des états du modèle par assimilation n'est effectuée, donc chaque prévision hydrologique comporte un total de 50 membres. Les prévisions hydrologiques sont réalisées quotidiennement pour un horizon de 14 jours sur l'ensemble de la période d'étude allant de mars 2016 à juin 2023.

2.7 Mesures de performance des prévisions

Cette section présente les différentes mesures utilisées pour évaluer les prévisions avec et sans assimilation, soit le *Continuous Ranked Probability Score* (CRPS) et les diagrammes de Talagrand.

2.7.1 Continuous Ranked Probability Score (CRPS)

Le CRPS est une mesure couramment utilisée pour évaluer la qualité des prévisions hydrologiques d'ensemble (Pagano, Shrestha, Wang, Robertson & Hapuarachchi, 2013). Le CRPS peut être interprété comme une généralisation de l'erreur moyenne absolue, adaptée aux prévisions d'ensemble. Il permet de mesurer la précision des prévisions en quantifiant à quel point celles-ci sont proches des évènements observés, tout en considérant la dispersion de l'ensemble. Contrairement à d'autres métriques qui évaluent uniquement une estimation centrale (comme la moyenne ou la médiane), le CRPS considère l'ensemble de la distribution des prévisions. Ainsi, cela signifie qu'il tient compte à la fois de la dispersion et de la concentration des prévisions autour de l'observation. Le CRPS peut avoir des valeurs comprises entre 0 et l'infini. Une valeur de 0 correspond à une prévision parfaitement précise, où tous les membres prédisent ce qui a été observé. À l'inverse, une valeur élevée indique une divergence importante entre la distribution des prévisions et l'observation. Le CRPS permet de quantifier l'impact de l'assimilation de données sur les prévisions hydrologiques. En

comparant les scores obtenus des prévisions avec et sans assimilation, il est possible d'évaluer dans quelle mesure l'intégration de l'assimilation de données améliore la précision et la fiabilité des prévisions hydrologiques. Le CRPS est défini selon l'équation figure 2.2, où $F_k'(x)$ est la fonction de distribution cumulative prédictive entre une prévision et une observation, x est la variable prédite, et F_k^o est la fonction de distribution cumulative observée correspondante.

$$\text{CRPS (k)} = \int_{-\infty}^{\infty} [F_k'(x) - F_k^o]^2 \, dx \tag{2.1}$$

La figure 2.2 illustre graphiquement le calcul du CRPS. La courbe rouge représente la fonction de distribution cumulative (CDF) de l'ensemble des prévisions, tandis que la ligne bleue montre la fonction de distribution cumulative de l'observation réelle. Le CRPS correspond à l'aire en rouge entre les deux courbes, c'est-à-dire l'intégrale de la différence au carré entre les deux CDF sur l'ensemble des valeurs possibles. Plus cette aire est petite, plus la prévision d'ensemble est proche de l'observation et a une faible variance.

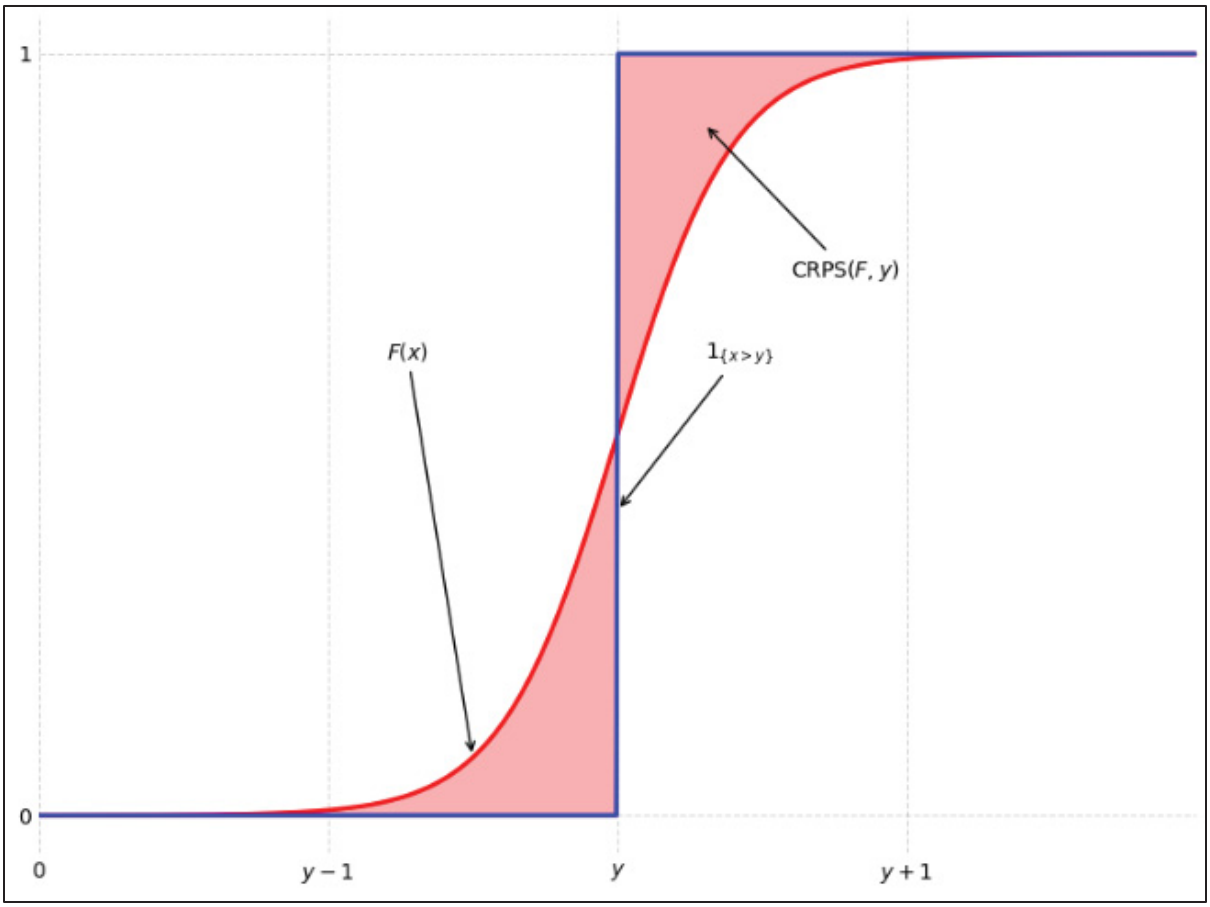

Figure 2.2 Représentation graphique du calcul du CRPS

2.7.2 Diagramme de Talagrand

Les diagrammes de Talagrand, également appelés histogrammes de rang, sont couramment utilisés pour évaluer la fiabilité des prévisions d'ensemble en hydrologie (Talagrand & Vautard, 1997). Ils permettent d'illustrer et d'analyser la dispersion des prévisions probabilistes par rapport aux observations. Ces diagrammes sont construits en classant l'observation réelle parmi les différentes prévisions d'ensemble, puis en comptant la fréquence à laquelle elle se retrouve dans chaque intervalle défini par les membres de l'ensemble. En répétant cette opération sur l'ensemble des prévisions, on obtient un histogramme indiquant la fréquence à laquelle les observations tombent dans chaque intervalle. Un histogramme uniforme indique une bonne dispersion des prévisions, où les observations ont une probabilité égale d'appartenir à chaque décile. À l'inverse, une distribution en forme de U suggère une

sous-dispersion, ce qui signifie que les observations se situent souvent en dehors des bornes de prévisions. Une forme de dôme centré indique une surdispersion, où les observations sont trop souvent au centre des prévisions. Enfin, un diagramme asymétrique révèle un biais dans les prévisions, avec une tendance à surestimer ou sous-estimer les observations. La figure 2.3 présente les formes typiques d'un diagramme de Talagrand. Les diagrammes de Talagrand sont particulièrement utiles pour évaluer la fiabilité des prévisions en vérifiant si les prévisions d'ensemble englobent correctement les résultats possibles.

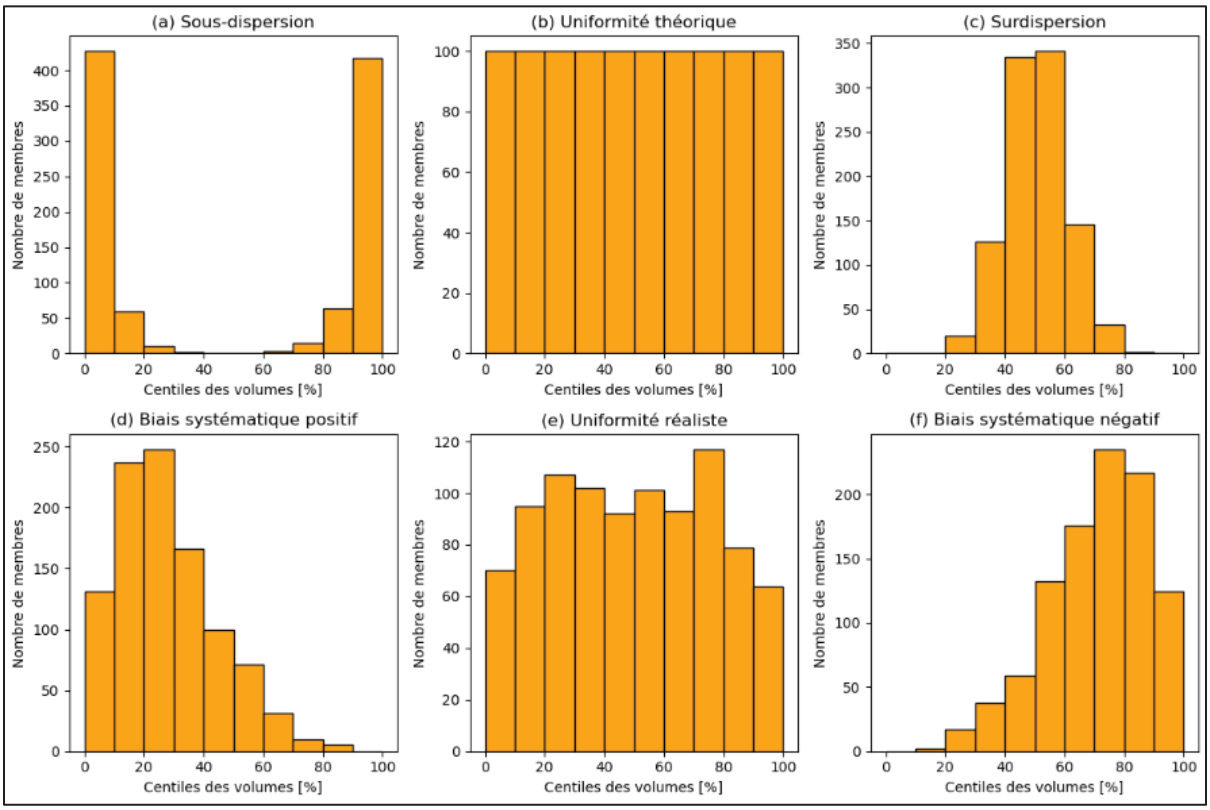

Figure 2.3 Exemples de formes typiques de diagrammes de Talagrand

2.8 Sommaire de la méthodologie

Ce projet met en œuvre une méthodologie qui utilise les données météorologiques issues des réanalyses ERA5 et des prévisions météorologiques ECMWF. Les trois modèles hydrologiques conceptuels ont été calibrés sur la période 1960-1990 en utilisant l'algorithme DDS avec le NSE comme critère d'évaluation. Une assimilation de données est effectuée avec EnKF afin d'améliorer les états initiaux des modèles hydrologiques. Enfin, la performance des

prévisions hydrologiques, avec et sans assimilation de données, a été évaluée en utilisant des métriques telles que le CRPS et les diagrammes de Talagrand. La figure 2.4 présente un sommaire méthodologique qui illustre les différentes étapes de la méthodologie.

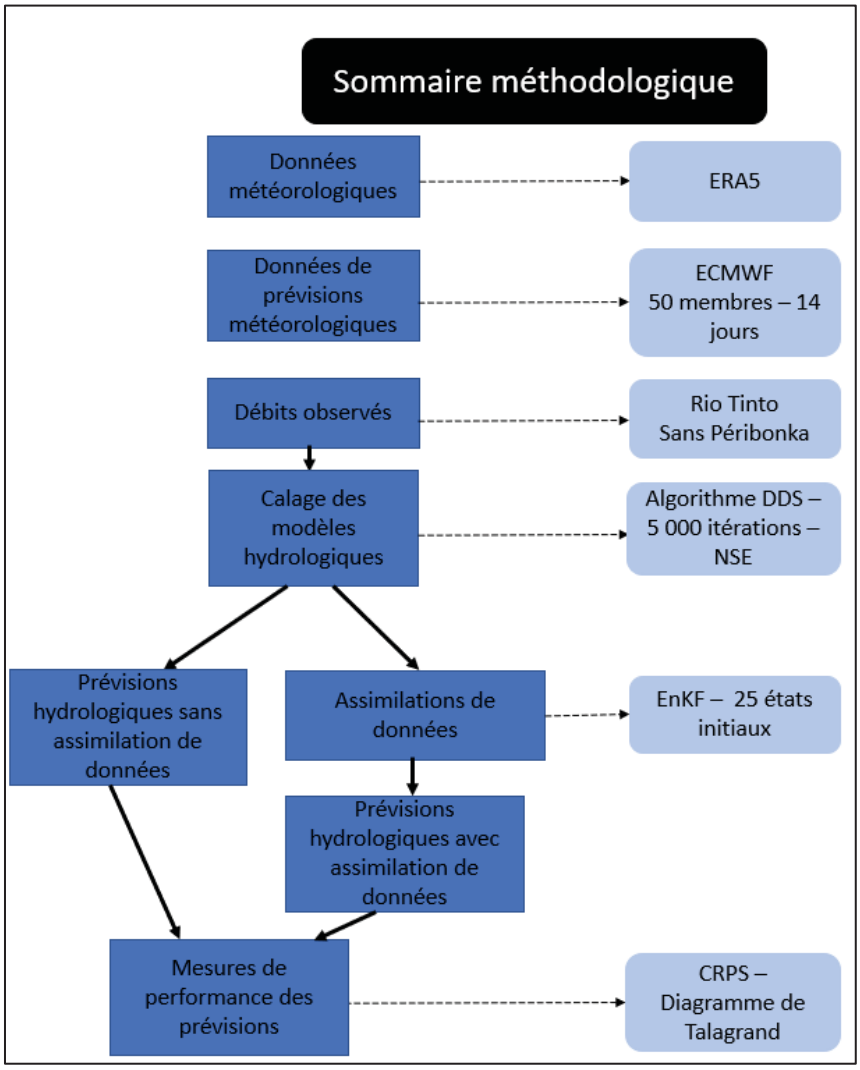

Figure 2.4 Sommaire de la méthodologie utilisée dans cette étude

CHAPITRE 3

RÉSULTATS

Ce chapitre présente les résultats obtenus à la suite des différentes manipulations de la méthodologie proposée. Plusieurs aspects sont analysés, tels que le calage des modèles hydrologiques, la performance des modèles, l'évaluation de l'impact de l'assimilation de données et l'évaluation de l'impact des saisons.

3.1 Calage des modèles hydrologiques

Les valeurs de NSE obtenues après le calage des modèles hydrologiques sur le bassin versant du LSJ sont présentées au tableau 3.1. Le modèle Blended se démarque par sa meilleure performance ayant le NSE le plus élevé, tandis que le modèle GR4J-CN présente le score le plus faible. Malgré ces différences, les trois modèles montrent des performances satisfaisantes.

Tableau 3.1 NSE obtenus au calage pour les trois modèles hydrologiques

Modèles hydrologiques	NSE
GR4J-CN	0,6902
MOHYSE	0,7839
Blended	0,8613

3.2 Performance des prévisions hydrologiques

Cette section présente la performance des prévisions hydrologiques, analysée sous deux angles complémentaires. D'une part, la précision des prévisions est évaluée à partir des scores de CRPS, en tenant compte de différents horizons de prévision. D'autre part, la fiabilité des prévisions est évaluée à l'aide des diagrammes de Talagrand. Pour des raisons de lisibilité, l'axe en ordonnée a été tronqué sur certains graphiques de CRPS afin de mieux voir les boîtes à moustaches. Cette modification n'affecte pas l'interprétation des résultats.

3.2.1 CRPS des prévisions hydrologiques

La figure 3.1 présente les résultats du CRPS pour les prévisions hydrologiques obtenues sans assimilation de données, pour les trois modèles à l'étude, sur un horizon de prévision allant jusqu'à 14 jours. Le modèle GR4J-CN présente les valeurs de CRPS les plus faibles sur l'ensemble des horizons de prévision. Les modèles Blended et MOHYSE affichent des CRPS plus élevés et relativement semblables, sauf à partir du jour 8, où une plus grande dispersion des résultats est observée. En effet, une augmentation plus importante des scores pour Blended est observée. De plus, le CRPS augmente avec les jours de prévisions. Ainsi, plus l'horizon est long, plus les prévisions sont incertaines, ce qui montre une diminution de la précision des prévisions avec le temps. Cette dégradation de la précision est observable autant sur la médiane que sur l'étendue des valeurs, notamment pour les horizons plus longs (7 à 14 jours).

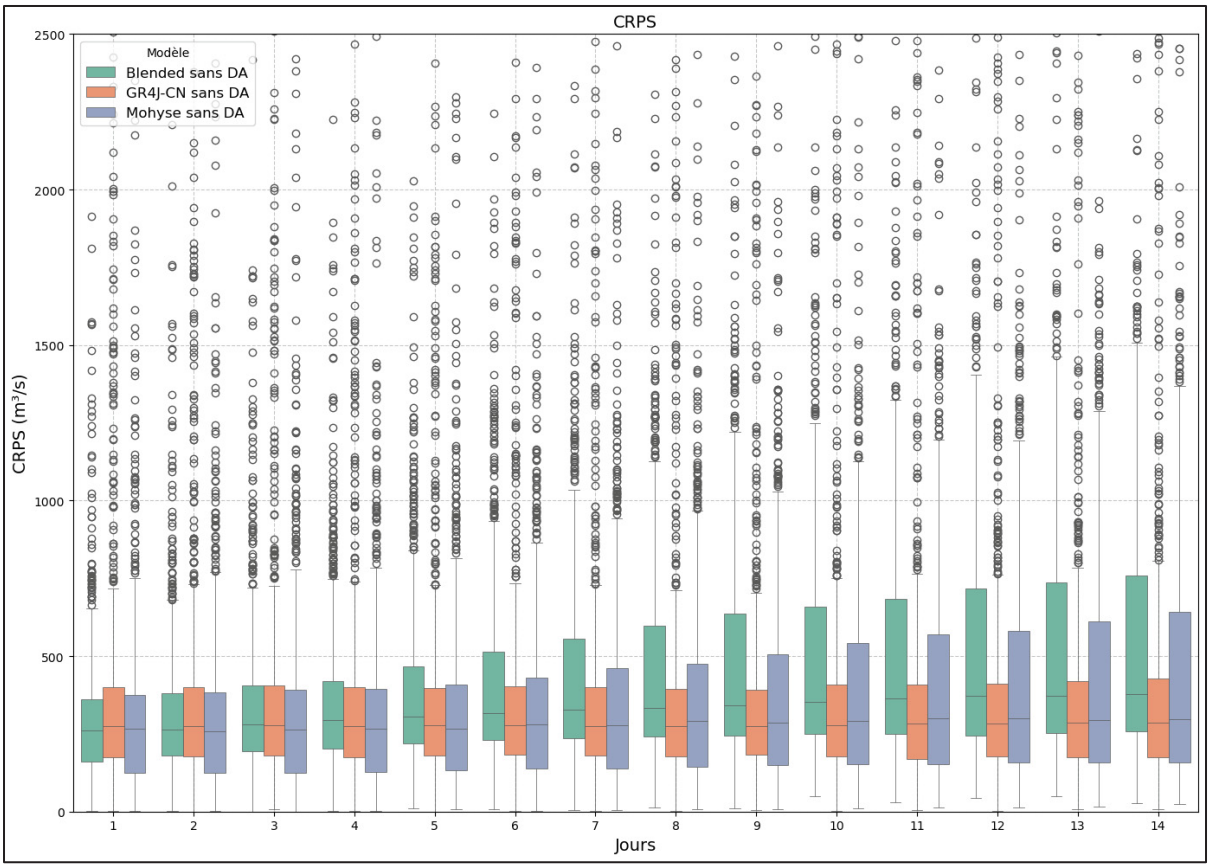

Figure 3.1 Scores CRPS pour les prévisions hydrologiques obtenues sans assimilation de données pour les trois modèles avec un horizon de prévision de 14 jours

La figure 3.2 présente les résultats des CRPS pour les prévisions hydrologiques obtenues avec assimilation de données, pour les trois modèles, sur les mêmes horizons prédictifs. Il est possible de constater l'effet de l'assimilation de données sur la qualité des prévisions. En effet, pour l'ensemble des modèles, l'assimilation de données permet une réduction des valeurs de CRPS sur tous les horizons de prévisions. Cette amélioration est particulièrement marquée pour le modèle GR4J-CN, qui conserve les meilleures performances tout en affichant une variabilité réduite. Le modèle Blended affiche une amélioration particulièrement pour les horizons courts, mais sa performance demeure inférieure à celle de GR4J-CN. MOHYSE, quant à lui, présente une amélioration moins prononcée. Par rapport aux résultats obtenus sans assimilation, les boîtes à moustaches sont plus resserrées et les valeurs médianes plus faibles. Enfin, plus l'horizon de prévision est élevé, plus la variabilité des résultats augmente, et ce, pour les trois modèles.

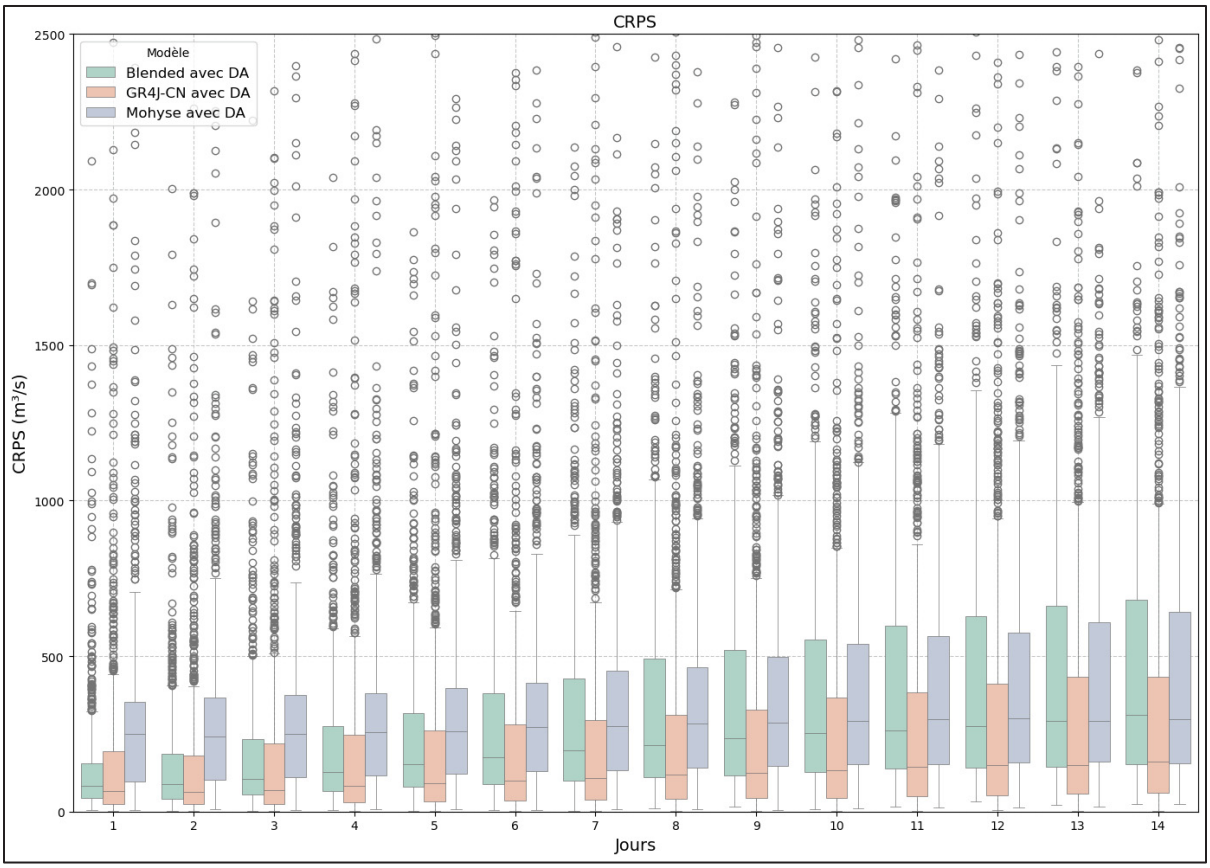

Figure 3.2 Scores CRPS pour les prévisions hydrologiques obtenues avec assimilation de données pour les trois modèles avec un horizon de prévision de 14 jours

La figure 3.3 présente les résultats du CRPS pour les trois modèles sans assimilation de données, répartis par saison hydrologique. Les résultats indiquent des performances globalement plus stables et des valeurs de CRPS plus faibles durant la saison hivernale (DJFM), tous modèles confondus. À l'inverse, le printemps (AM) se caractérise par les CRPS les plus élevés et une variabilité accrue, en particulier à partir du jour 5. La saison estivale demeure celle où les CRPS sont les plus élevés de manière globale, en raison des valeurs médianes plus élevées et d'une plus grande dispersion. Le modèle GR4J-CN montre les performances les plus constantes sur l'ensemble des saisons, avec des CRPS plus faibles que ceux des autres modèles, en particulier en été et au printemps. En été (JJA), les erreurs augmentent, avec une forte dispersion des résultats et la présence de nombreux horsains. L'automne (SON), présente une dispersion modérée, mais avec de nombreux horsains. De manière générale, plus l'horizon prédictif est long, plus le CRPS tend à augmenter, mais cette

tendance est amplifiée en été et au printemps. Ces résultats montrent l'impact de la saisonnalité sur les prévisions hydrologiques.

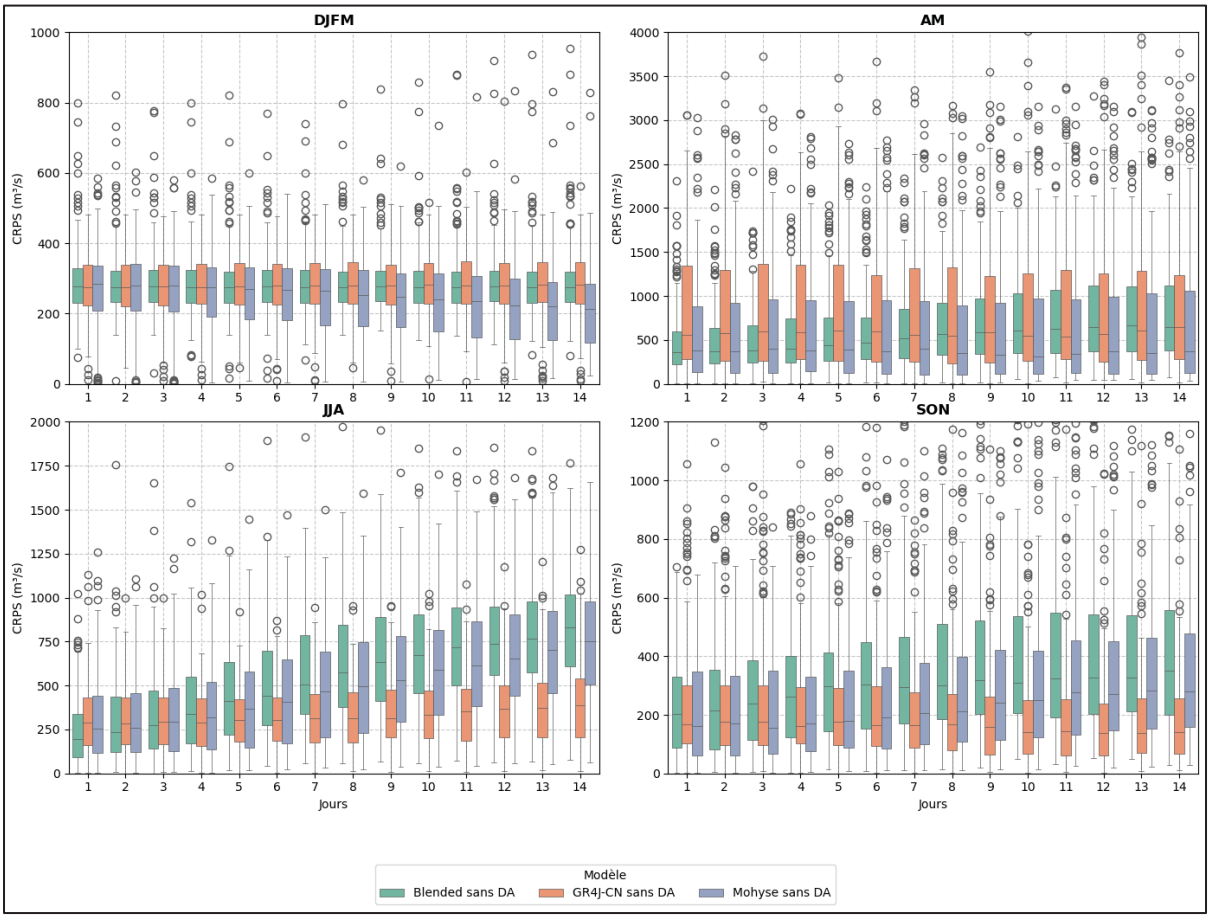

Figure 3.3 CRPS obtenus pour les trois modèles sans assimilation de données, répartis par saison hydrologique

La figure 3.4 présente les résultats du CRPS pour les trois modèles avec assimilation de données, séparés par saison hydrologique. L'impact de l'assimilation de données est visible dans toutes les saisons, avec une diminution des valeurs de CRPS par rapport aux résultats obtenus sans assimilation de données. Cette amélioration est particulièrement observable durant la saison hivernale, où les CRPS sont plus faibles et les distributions moins dispersées. En été, malgré le fait que les valeurs de CRPS restent plus élevées qu'en hiver, l'assimilation de données permet de réduire la médiane et la dispersion des résultats. Au printemps, l'effet de l'assimilation est également observable, mais atténué par les nombreuses valeurs aberrantes

qui persistent. L'automne présente encore des résultats intermédiaires comparativement aux autres saisons, mais avec des distributions moins étendues. Dans l'ensemble, le modèle GR4J-CN est celui qui bénéficie le plus de l'assimilation de données dans toutes les saisons. En effet, il affiche des médianes plus faibles et une variabilité plus faible que les deux autres modèles, sauf au printemps où Blended est celui qui offre une meilleure performance.

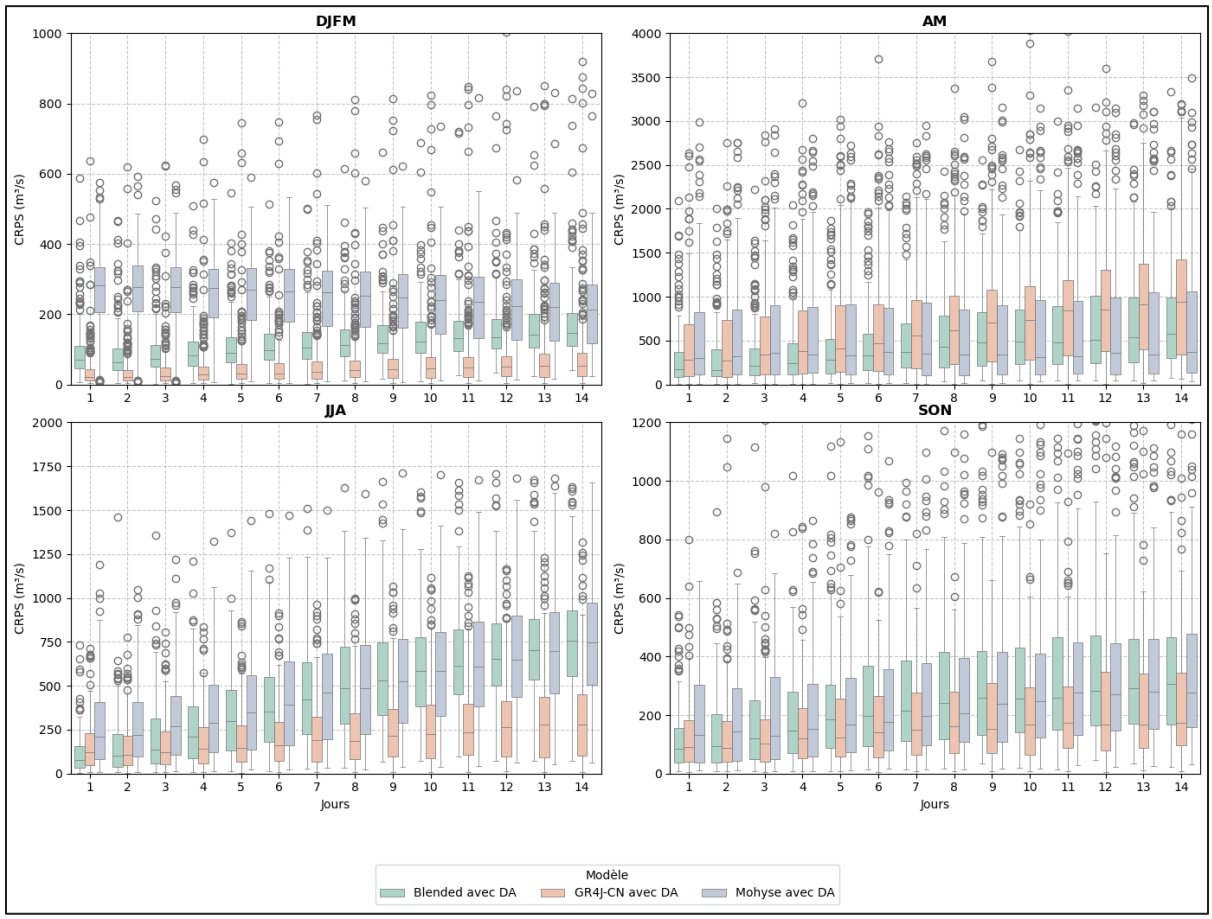

Figure 3.4 CRPS obtenus pour les trois modèles avec assimilation de données, répartis par saison hydrologique

La figure 3.5 illustre l'effet de l'assimilation de données sur les performances du modèle Blended, en termes de CRPS sur un horizon de 14 jours. Les résultats montrent l'effet bénéfique de l'assimilation de données. En effet, pour l'ensemble des horizons prédictifs, les valeurs de CRPS sont plus faibles avec assimilation de données. Dès le premier jour de prévision, les médianes sont plus basses avec assimilation, et cette différence se maintient tout

au long de l'horizon prédictif, ce qui indique une meilleure précision. De plus, les boîtes à moustaches associées au scénario avec assimilation de données sont moins étendues, particulièrement jusqu'au jour 6, indiquant une réduction de la variabilité. Cependant, à partir du jour 7, la dispersion augmente progressivement, mais les performances restent supérieures à celles obtenues sans assimilation de données.

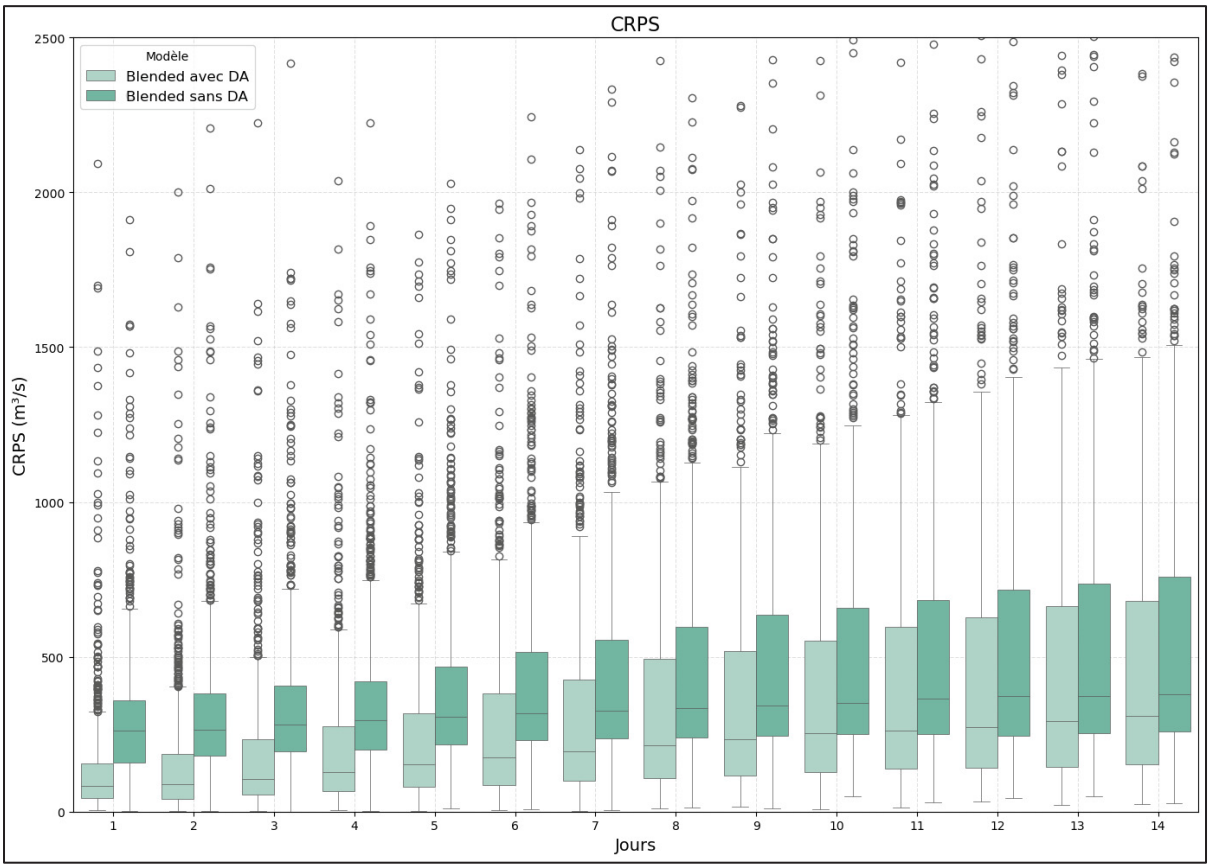

Figure 3.5 CRPS obtenus avec et sans assimilation de données pour le modèle Blended

La figure 3.6 présente les résultats de CRPS du modèle GR4J-CN avec et sans assimilation de données, sur un horizon prédictif de 14 jours. On constate que ce modèle bénéficie aussi de l'assimilation de données puisque les valeurs de CRPS sont plus faibles avec assimilation. Cette différence est particulièrement observable pour tous les horizons, où la médiane diminue visiblement. Cependant, à partir du jour sept, malgré les médianes plus faibles, on observe que

les boîtes à moustaches associées aux prévisions avec assimilation sont parfois plus étendues, ce qui indique une variabilité légèrement accrue.

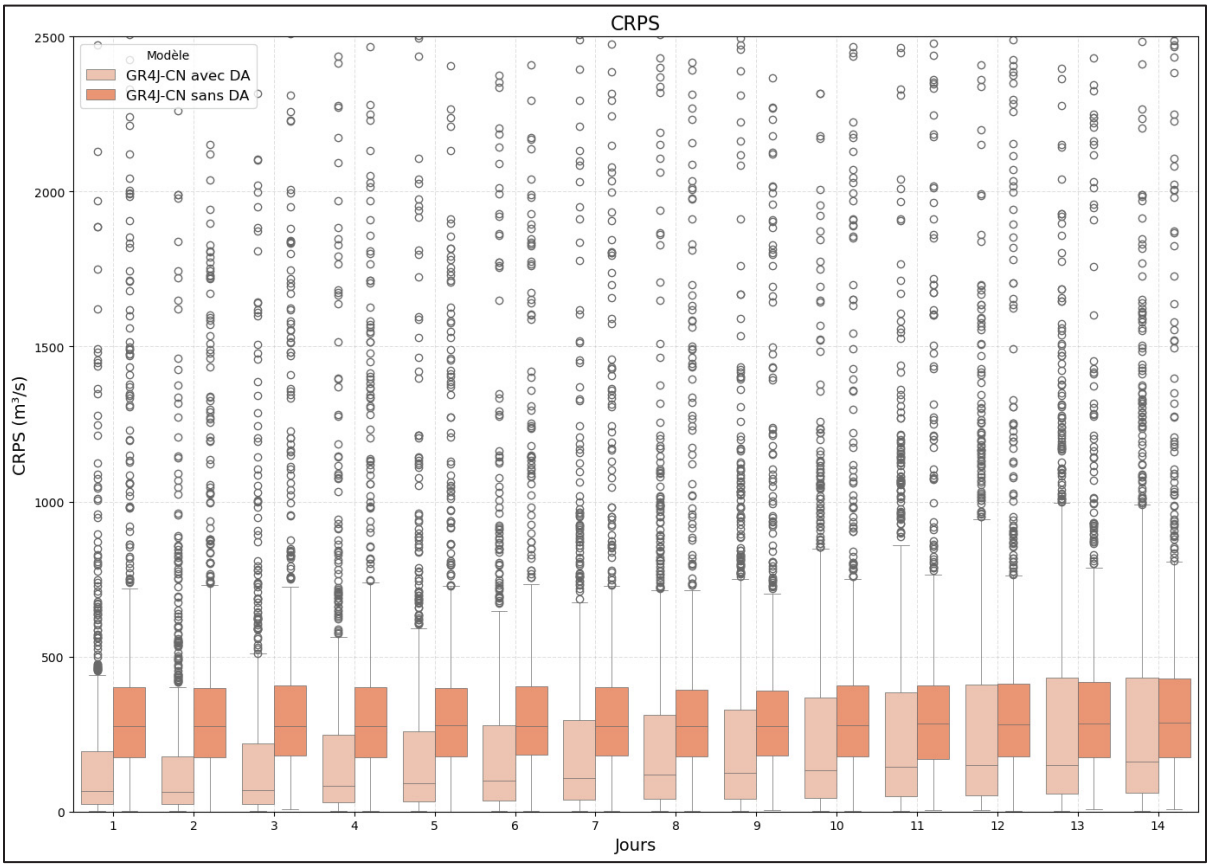

Figure 3.6 CRPS obtenus avec et sans assimilation de données pour le modèle GR4J-CN

La figure 3.7 compare les CRPS obtenus avec et sans assimilation de données pour le modèle MOHYSE, sur une période de prévision allant de 1 à 14 jours. Contrairement aux autres modèles, l'impact de l'assimilation de données est ici plus modéré. Une légère amélioration est observable pour les jours 1 à 5, notamment par une diminution de la médiane et une réduction de la dispersion. Cependant, cet effet tend à s'estomper au-delà du jour 6. Pour les horizons plus longs, la différence entre les deux scénarios devient peu remarquable et les distributions se superposent. Ainsi, cela suggère que, pour ce modèle, l'assimilation de données ne permet pas de corriger la propagation de l'incertitude à moyen terme. Néanmoins,

l'assimilation de données contribue tout de même à améliorer légèrement les prévisions à court terme.

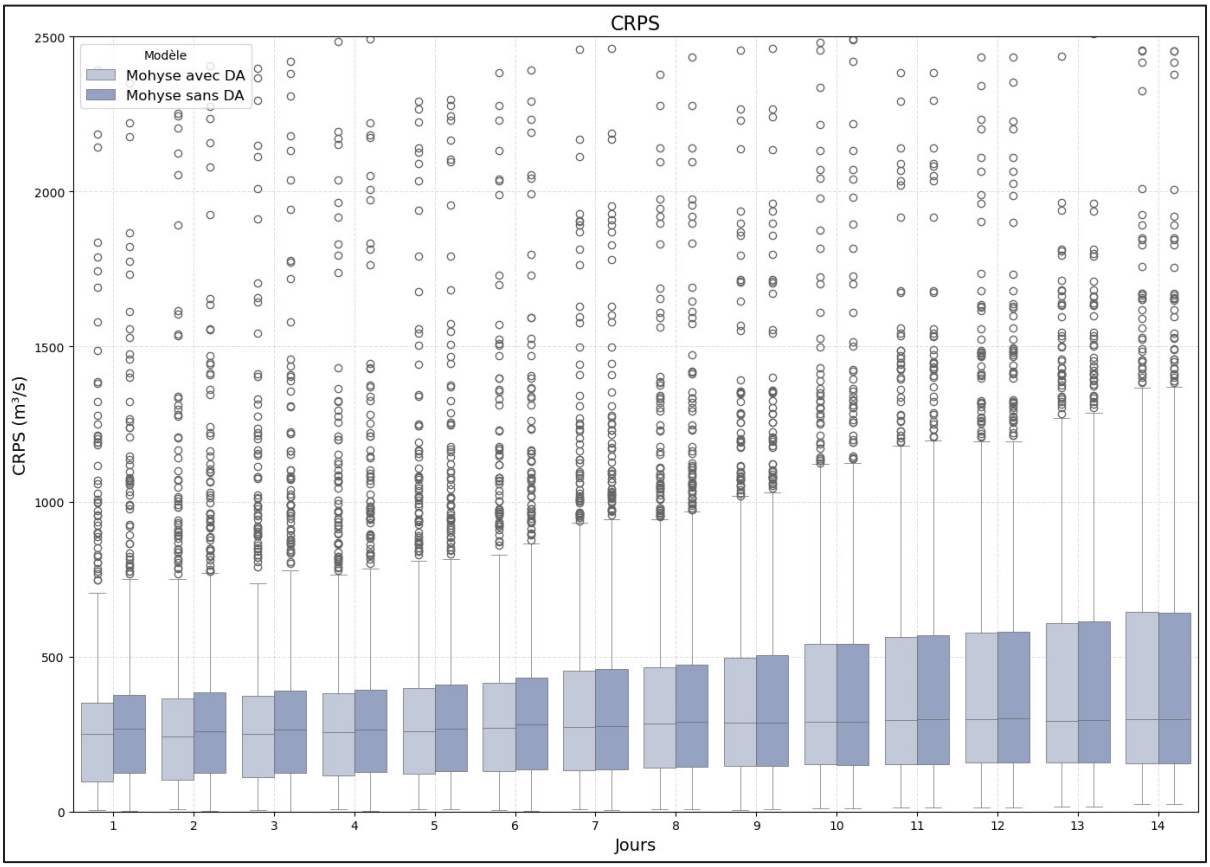

Figure 3.7 CRPS obtenus avec et sans assimilation de données pour le modèle MOHYSE

3.2.2 Diagramme de Talagrand

La figure 3.8 présente les diagrammes de Talagrand des prévisions hydrologiques réalisées sans assimilation de données pour les trois modèles hydrologiques et quatre horizons prédictifs : 1, 3, 7 et 14 jours. Il est possible de constater qu'il y a une forte concentration des observations dans les intervalles extrêmes, ce qui indique une sous-dispersion dans les prévisions d'ensemble. Par ailleurs, à mesure que l'horizon prédictif augmente, la répartition des observations devient légèrement plus visible, particulièrement pour MOHYSE, tandis qu'elle reste relativement similaire pour les deux autres modèles. Le modèle GR4J-CN

présente un diagramme relativement constant sur tous les horizons de prévisions, ce qui montre une sous-dispersion sur les différentes échéances temporelles. Dans l'ensemble, aucun des trois modèles ne présente une distribution uniforme dans les intervalles centraux, ce qui indique un manque de fiabilité des prévisions d'ensemble.

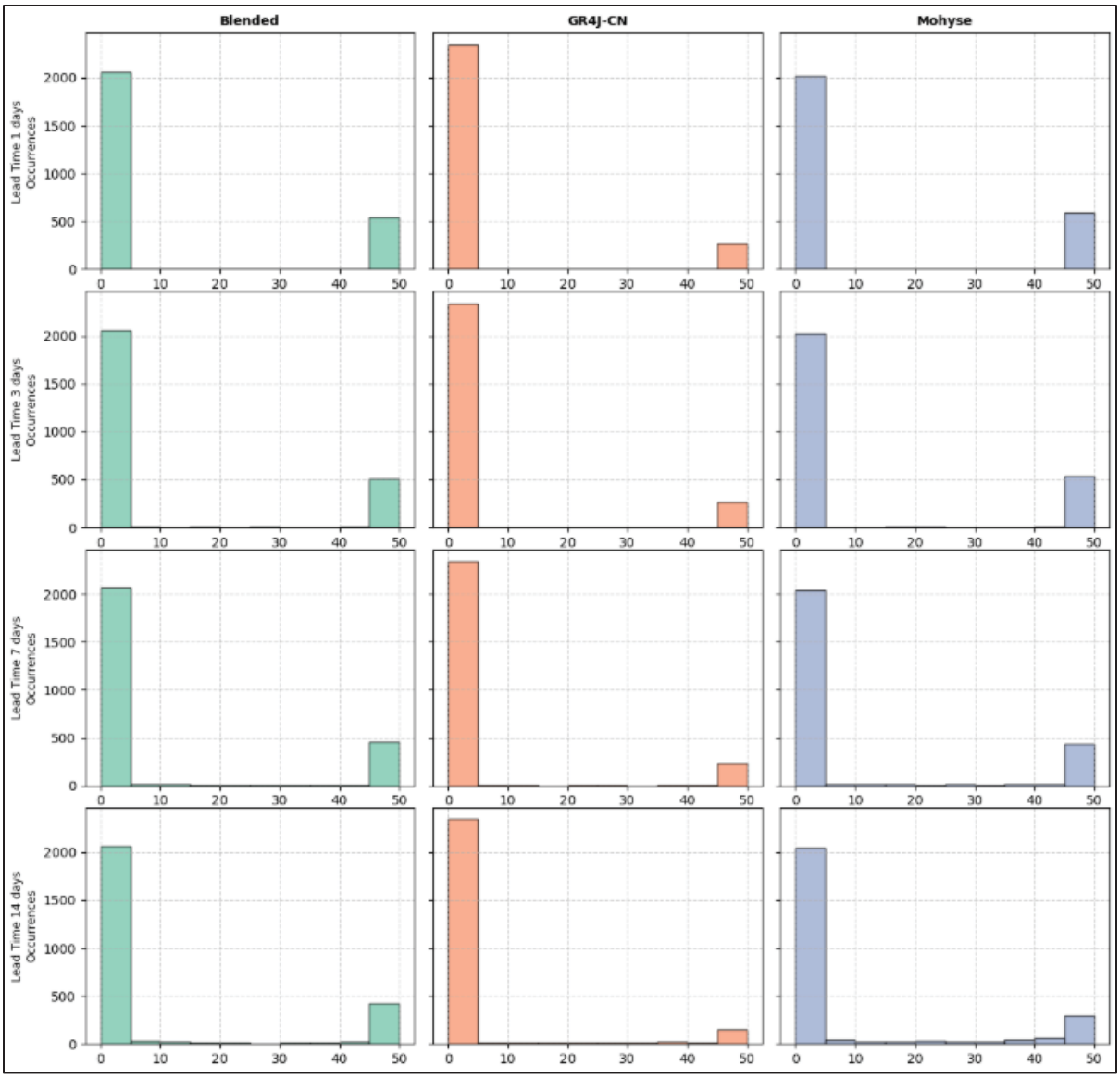

Figure 3.8 Diagramme de Talagrand des prévisions hydrologiques réalisées sans assimilation de données pour les trois modèles hydrologiques et quatre horizons prédictifs

La figure 3.9 présente quant à elle les diagrammes de Talagrand des prévisions hydrologiques réalisées avec l'assimilation de données pour les trois modèles hydrologiques, et ce pour les quatre mêmes horizons prédictifs. Le modèle Blended continue d'afficher une concentration importante aux extrémités, mais la distribution des prévisions apparaît légèrement mieux répartie dans les intervalles centraux, particulièrement pour les horizons courts (1 et 3 jours). Pour le modèle GR4J-CN, une amélioration subtile, mais visible est observée, avec une répartition un peu plus dispersée des observations sur l'ensemble des intervalles, ce qui suggère une amélioration modérée de la dispersion des prévisions. En revanche, le modèle MOHYSE présente une détérioration des résultats après l'application de l'assimilation de données. La distribution des rangs est moins bien répartie, avec une diminution des occurrences dans les intervalles centraux. La sous-dispersion reste prononcée pour les trois modèles. De plus, l'amélioration apportée par l'assimilation de données diminue avec l'augmentation de l'horizon de prévision. En effet, les diagrammes avec des horizons plus longs (7 et 14 jours) demeurent plus biaisés, ce qui démontre une difficulté persistante à estimer correctement l'incertitude des prévisions à mesure que l'échéance temporelle augmente.

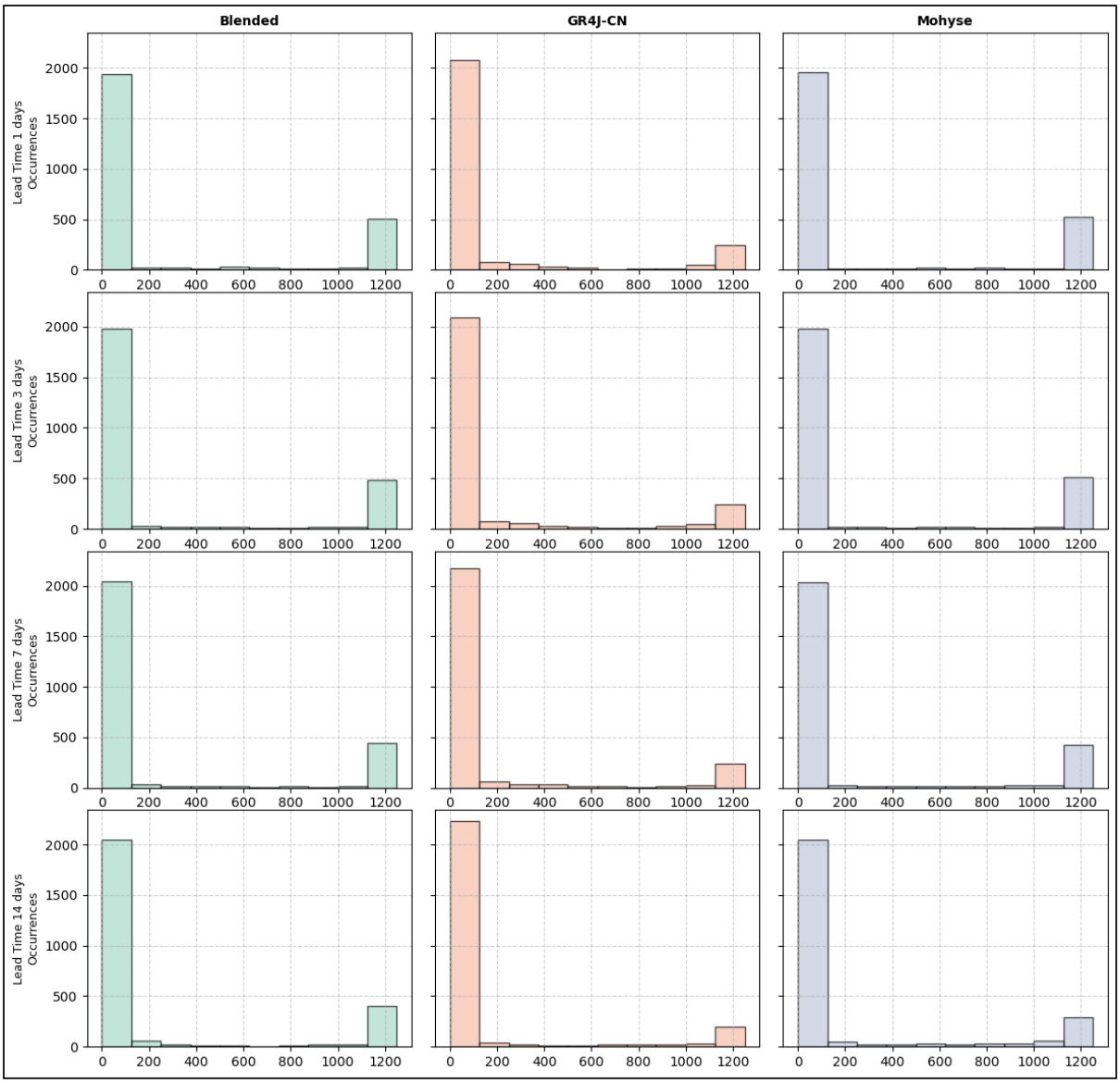

Figure 3.9 Diagramme de Talagrand des prévisions hydrologiques réalisées avec assimilation de données pour les trois modèles hydrologiques et quatre horizons prédictifs

La figure 3.10 présente les diagrammes de Talagrand pour les quatre saisons hydrologiques, générés à partir des prévisions hydrologiques obtenues sans assimilation de données, pour les trois modèles étudiés et pour quatre horizons prédictifs. On constate que les histogrammes sont soit fortement asymétriques, ce qui indique un biais systématique positif des ensembles de prévisions, soit qu'ils présentent une concentration importante aux extrémités, ce qui indique une sous-dispersion. Cela signifie que les observations se situent fréquemment en dehors des

prévisions. En hiver, la sous-dispersion est plus prononcée pour le modèle GR4J-CN, tandis que pour MOHYSE et Blended montrent plutôt un biais systématique. Au printemps, un biais est toujours observable, ce qui limite la capacité des modèles à représenter la variabilité des débits liée à la fonte des neiges. Une légère augmentation de la dispersion est toutefois observable à l'horizon de 14 jours. D'un autre côté, en automne, une légère amélioration est observée pour les modèles Blended et MOHYSE aux horizons plus longs. La concentration plus marquée dans les premiers intervalles indique une tendance des modèles à surestimer les débits. En été, les trois modèles présentent des biais systématiques positifs. Qui plus est, l'analyse des horizons prédictifs montre que la sous-dispersion et le biais systématique persistent jusqu'au 14^{e} jour de prévision. Malgré une dispersion occasionnellement plus équilibrée du modèle MOHYSE, aucun des trois modèles ne parvient à fournir une couverture fiable, quel que soit la saison ou l'horizon prédictif. Ainsi, comme aucun des modèles ne parvient à générer des histogrammes uniformes, cela démontre une fiabilité limitée par rapport aux prévisions générées par les trois modèles.

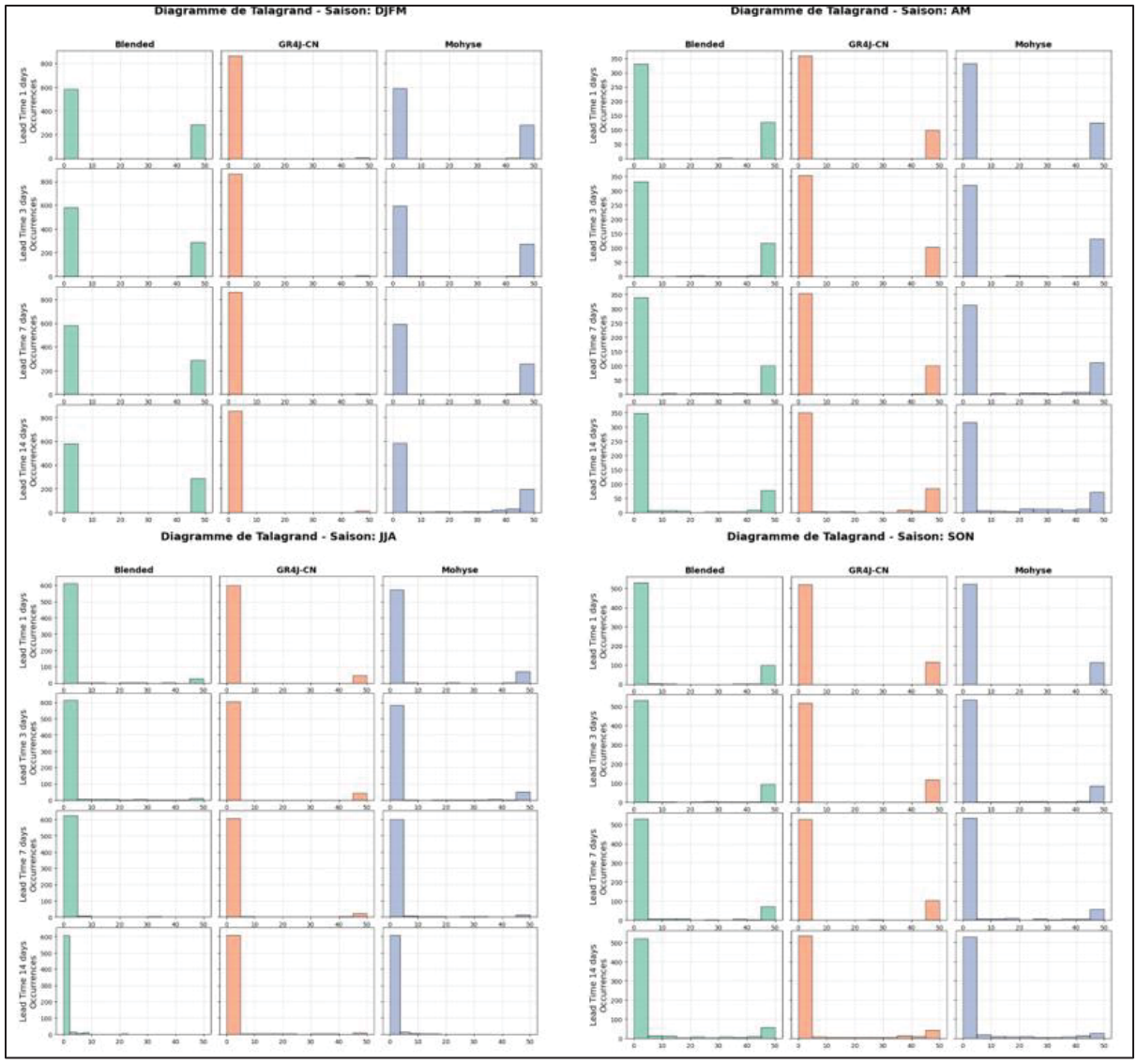

Figure 3.10 Diagrammes de Talagrand pour quatre saisons hydrologiques, générés à partir des prévisions hydrologiques obtenues sans assimilation de données, pour les trois modèles étudiés et pour quatre horizons prédictifs

La figure 3.11 présente les diagrammes de Talagrand obtenus avec assimilation de données pour les modèles, en fonction des saisons et de l'horizon de prévision. Les résultats montrent une amélioration de la fiabilité par rapport aux configurations sans assimilation. En effet, malgré le fait que les histogrammes demeurent généralement asymétriques et avec une concentration dans les valeurs extrêmes, la sous-dispersion des ensembles de prévisions est réduite. En effet, les observations se trouvent plus fréquemment à l'intérieur de l'ensemble de

prévisions, et la concentration dans les classes extrêmes diminue. Les rangs de prévisions sont un peu plus équilibrés, bien qu'une dispersion parfaitement uniforme ne soit pas encore atteinte. En hiver et au printemps, l'assimilation de données améliore la capacité des modèles à inclure les observations dans les intervalles de prévisions, même si les histogrammes présentent encore des biais, plus particulièrement pour les horizons prédictifs plus longs. En hiver, GR4J-CN est le modèle qui bénéficie le plus de l'assimilation, avec une répartition des rangs légèrement améliorée. Au printemps, ce sont les modèles Blended et GR4J-CN qui montrent une plus grande amélioration. En été, où les biais étaient les plus prononcés pour les trois modèles en absence d'assimilation de données, les histogrammes montrent une légère amélioration, avec une répartition un peu plus équilibrée aux horizons courts. Toutefois, l'été est la saison où l'effet de l'assimilation de données semble le moins bénéfique. En automne, une atténuation du biais de surestimation est également observée pour les trois modèles. De plus, en analysant l'effet des horizons de prévisions, on constate que l'assimilation de données contribue à limiter l'augmentation de l'incertitude au fil du temps, ce qui améliore les prévisions sur des horizons à plus long terme.

Ces résultats montrent que même si l'assimilation de données améliore partiellement la qualité des prévisions hydrologiques, certaines lacunes persistent quant à la fiabilité des prévisions hydrologiques, en particulier pour les horizons prédictifs à long terme (7 et 14 jours).

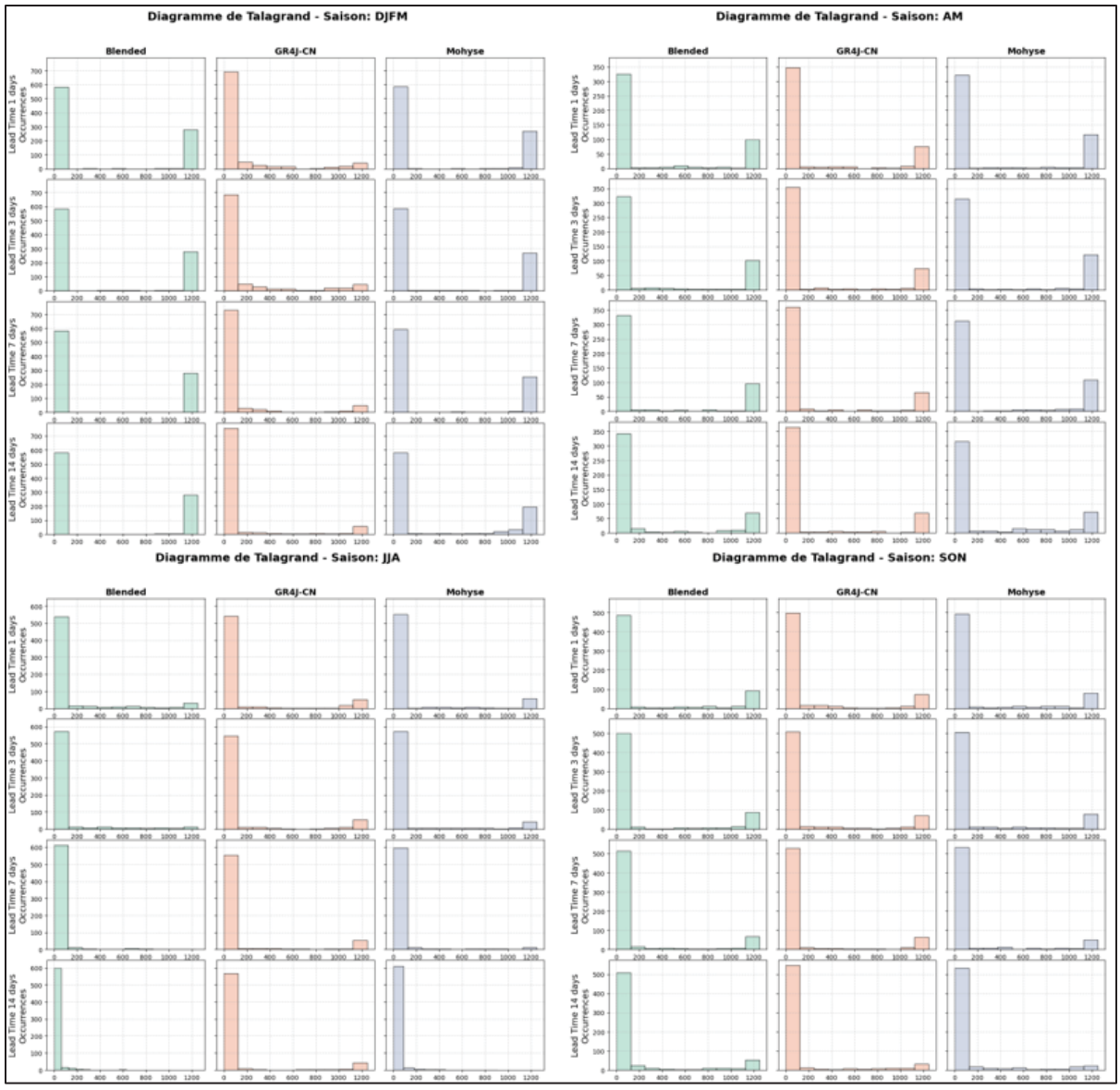

Figure 3.11 Diagrammes de Talagrand pour quatre saisons hydrologiques, générés à partir des prévisions hydrologiques obtenues avec assimilation de données, pour les trois modèles étudiés et pour quatre horizons prédictifs

3.3 Comparaison de la dispersion des ensembles

Les figures 3.12 et 3.13 présentent les ensembles de prévisions hydrologiques sur un horizon de 14 jours pour le cas sans assimilation de données et avec assimilation de données, respectivement, pour le jour 150 des prévisions, pour le modèle Blended. Ceux obtenus pour les deux autres modèles hydrologiques sont présentés à l'annexe II. La différence de dispersion

entre les deux cas est facilement observable. En effet, sans assimilation de données, les prévisions montrent une dispersion beaucoup plus faible et tendent à surestimer les débits par rapport aux observations. L'enveloppe formée par les membres de l'ensemble ne parvient pas à englober les débits observés. À l'inverse, le cas avec assimilation de données, qui combinent les 25 membres assimilés avec les 50 membres de prévisions météorologiques, produit un ensemble de 1 250 membres qui génère un éventail de scénarios beaucoup plus large. Ainsi, l'augmentation de la dispersion montre une meilleure représentation de l'incertitude et offre une enveloppe plus réaliste autour des observations. On remarque d'ailleurs que l'ensemble généré avec assimilation de données parvient à couvrir plus fréquemment les observations. Toutefois, l'impact de l'assimilation de données semble être moins dominant que celui induit par la variabilité des membres météorologiques.

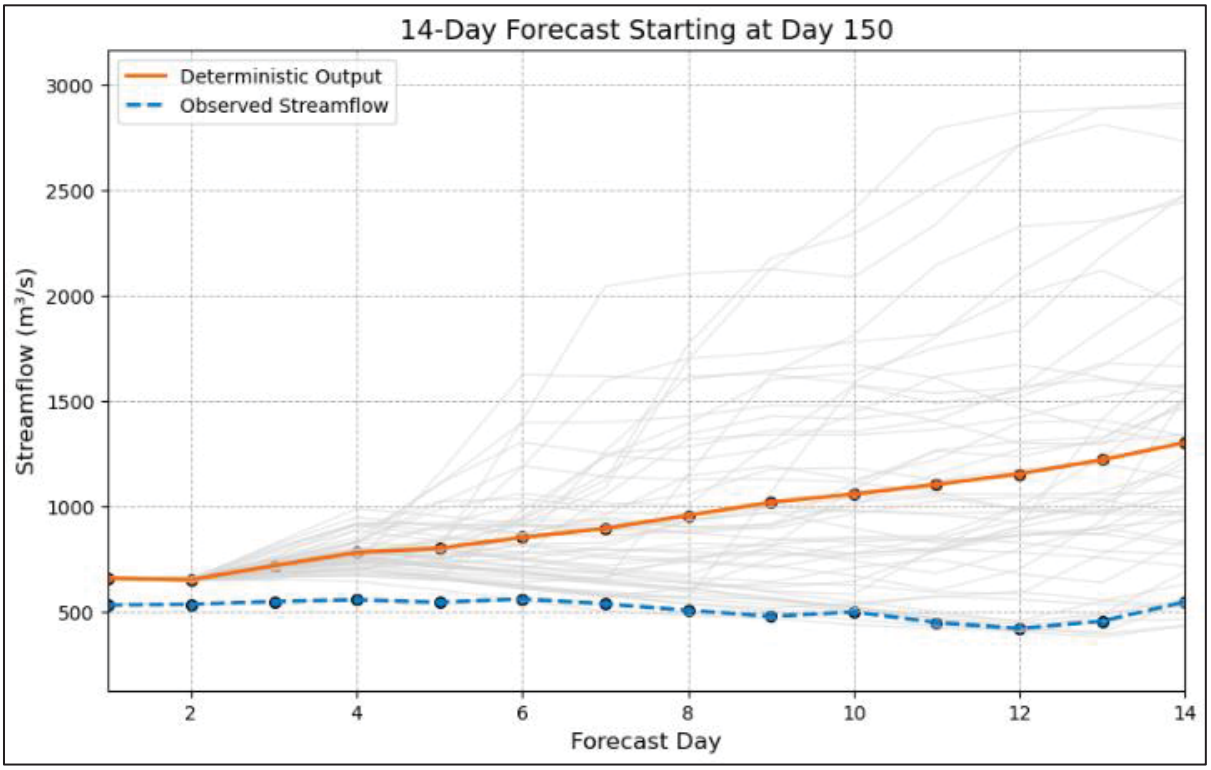

Figure 3.12 Dispersion de l'ensemble sans assimilation de données sur un horizon de 14 jours commençant le jour 150 pour le modèle Blended

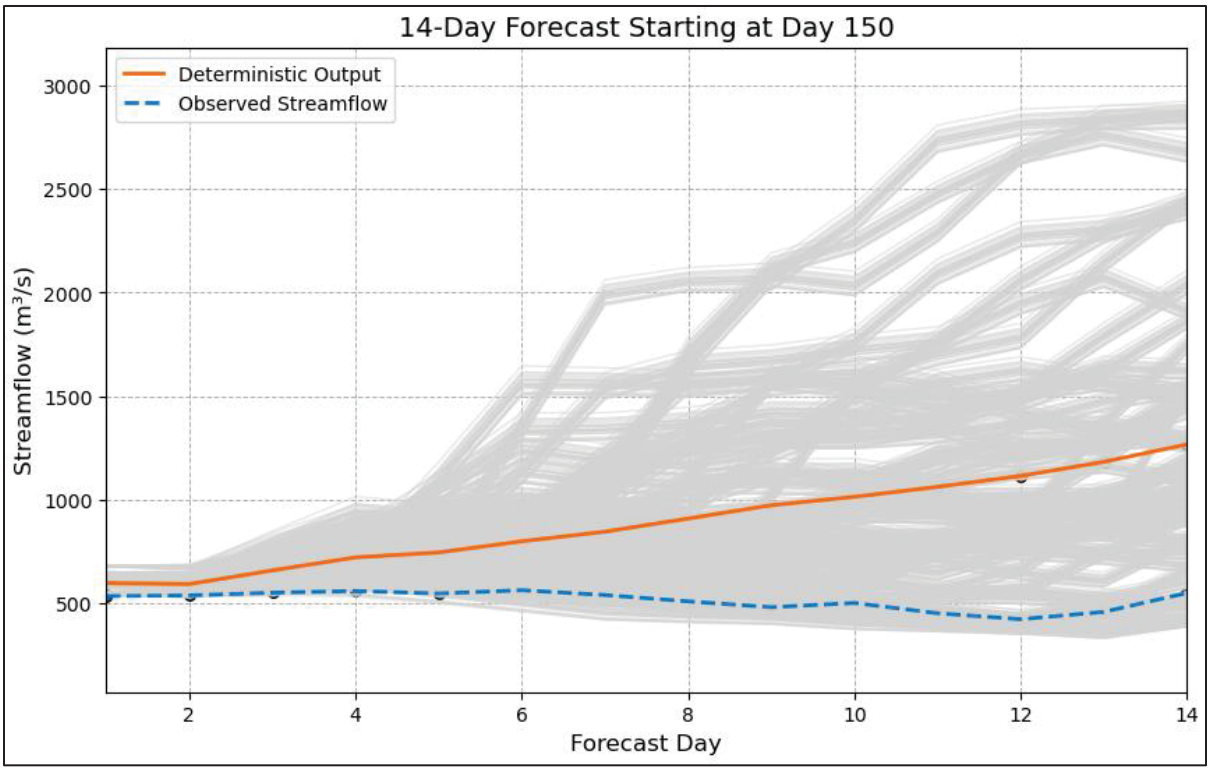

Figure 3.13 Dispersion de l'ensemble avec assimilation de données sur un horizon de 14 jours commençant le jour 150 pour le modèle Blended

3.4 Hydrogrammes

Cette section présente des hydrogrammes prévisionnels pour chacun des modèles, et les figures montrent la moyenne de tous les membres de l'ensemble pour chaque date et chaque horizon de prévision.

Les figures 3.14 à 3.17 présentent la comparaison entre les débits observés et les prévisions déterministes obtenues à partir de la moyenne des ensembles pour le modèle GR4J-CN, avec et sans assimilation de données, aux horizons de prévision, 1, 3, 7 et 14 jours. Dans le cas sans assimilation de données, on constate que les prévisions tendent à sous-estimer les débits lors des pics de crue. Cette sous-estimation est présente dès le premier jour de prévision et s'accentue à mesure que l'horizon augmente, comme le démontre la baisse progressive du KGE, passant de 0,60 à 0,46 entre le jour 1 et 14. Cependant, la saisonnalité est bien reproduite, le modèle parvient à représenter correctement les tendances générales du régime hydrologique.

Le cas avec assimilation de données montre une meilleure concordance avec les débits observés. En effet, les pics de crue sont mieux captés, bien qu'ils soient parfois surestimés et les écarts sont réduits. Cette amélioration se reflète dans l'augmentation des valeurs de KGE, qui atteint 0,84 au jour 1, 0,82 au jour 3, 0,76 au jour 7, avant de diminuer à 0,72 au jour 14. La qualité des prévisions diminue avec l'augmentation de l'horizon de prévision, cependant, elle demeure supérieure à celle obtenue sans assimilation de données.

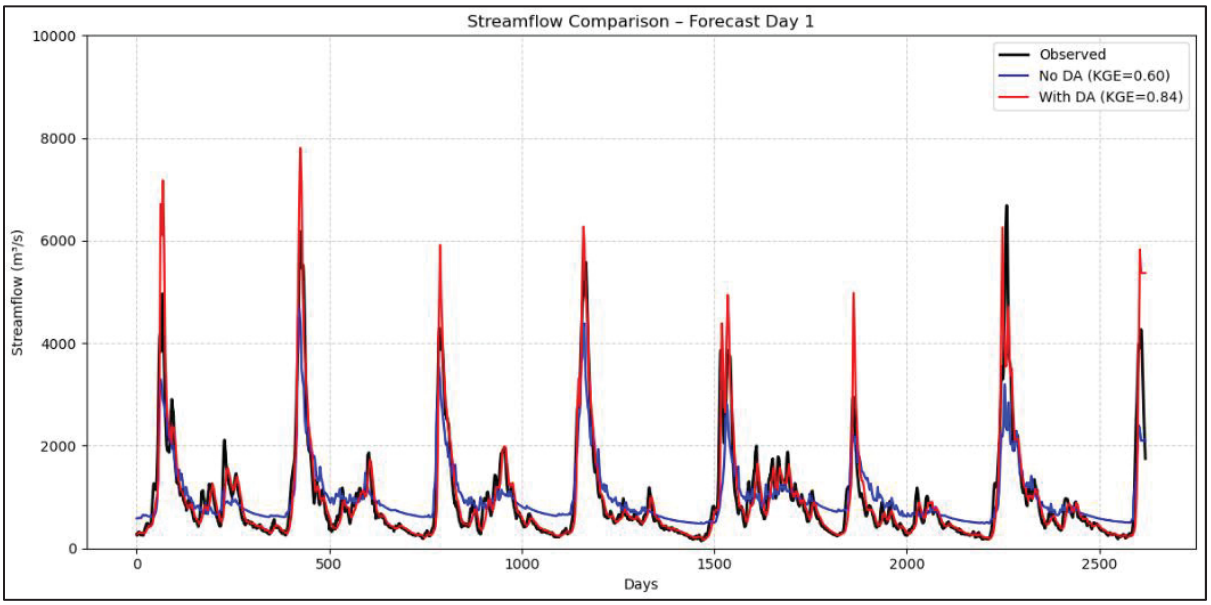

Figure 3.14 Comparaison entre les débits observés et les prévisions déterministes obtenues à partir de la moyenne des ensembles pour le modèle GR4J-CN pour l'horizon de prévision 1 jour

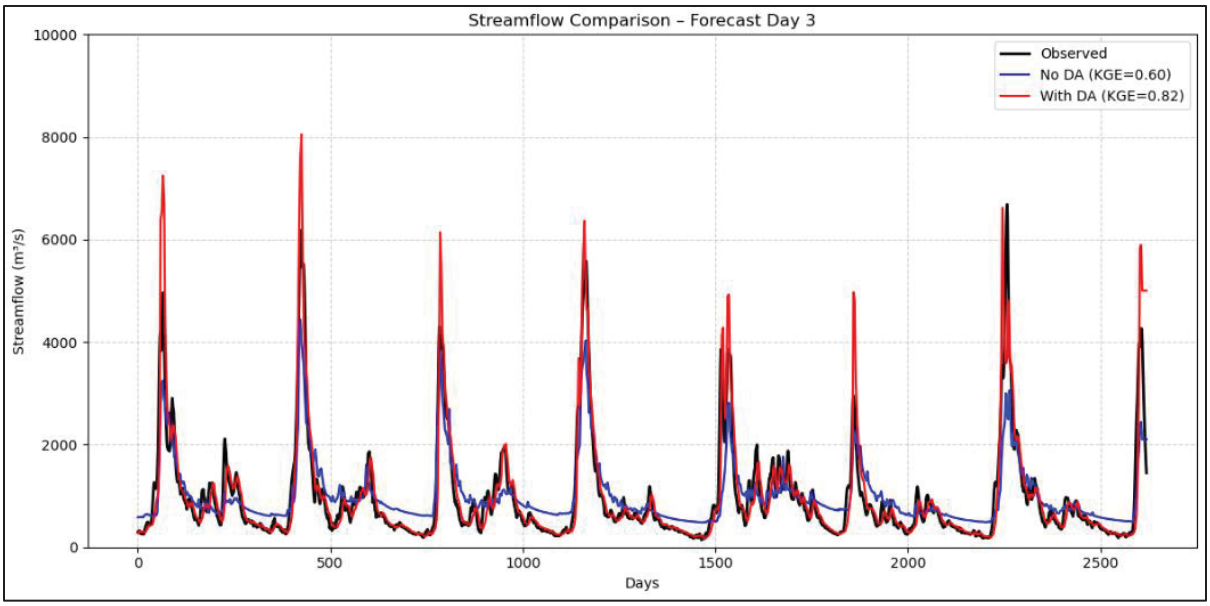

Figure 3.15 Comparaison entre les débits observés et les prévisions déterministes obtenues à partir de la moyenne des ensembles pour le modèle GR4J-CN pour l'horizon de prévision 3 jours

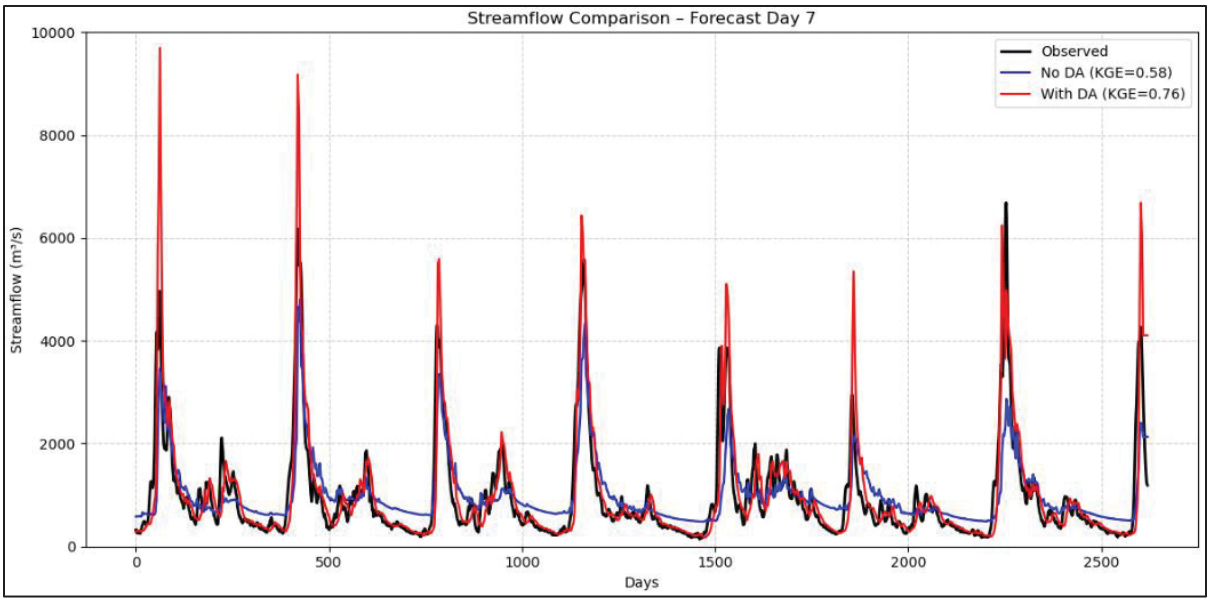

Figure 3.16 Comparaison entre les débits observés et les prévisions déterministes obtenues à partir de la moyenne des ensembles pour le modèle GR4J-CN pour l'horizon de prévision 7 jours

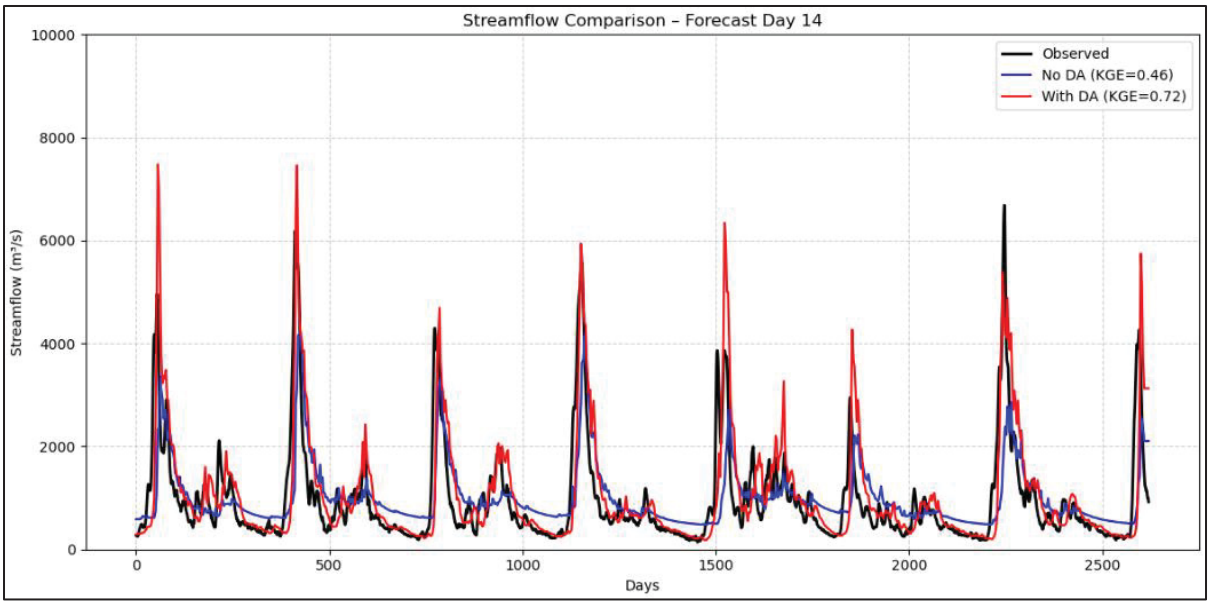

Figure 3.17 Comparaison entre les débits observés et les prévisions déterministes obtenues à partir de la moyenne des ensembles pour le modèle GR4J-CN pour l'horizon de prévision 14 jours

Les figures 3.18 à 3.21 présentent la comparaison entre les débits observés et les prévisions déterministes obtenues à partir de la moyenne des ensembles pour le modèle Blended, avec et sans assimilation de données, aux horizons de prévision, 1, 3, 7 et 14 jours. Le modèle reproduit bien la dynamique saisonnière, mais a de la difficulté à reproduire les faibles débits entre les périodes de crues. Ce phénomène est de plus en plus prononcé à mesure que l'horizon de prévision avance. De plus, dans le cas sans assimilation de données, on observe la présence récurrente de prévisions de débit nul entre les pics de crues, alors que les observations montrent des débits positifs. Ainsi, le modèle a de la difficulté à correctement prédire les débits durant les périodes d'étiage. Par ailleurs, les pics de crue sont assez bien représentés, mais sont progressivement sous-estimés avec l'augmentation de l'horizon. Dans le cas avec assimilation de données, on remarque que les pics de crue sont mieux captés ainsi qu'une amélioration dans la représentation des faibles débits. En effet, les débits simulés ne tombent plus à zéro pendant les périodes de faibles débits. Qui plus est, le KGE est amélioré aux horizons courts, il diminue progressivement plus l'horizon prédictif s'allonge. Au jour 14, le KGE obtenu avec assimilation de données devient équivalent à celui sans assimilation, ce qui montre que l'effet bénéfique de l'assimilation de données s'atténue avec le temps.

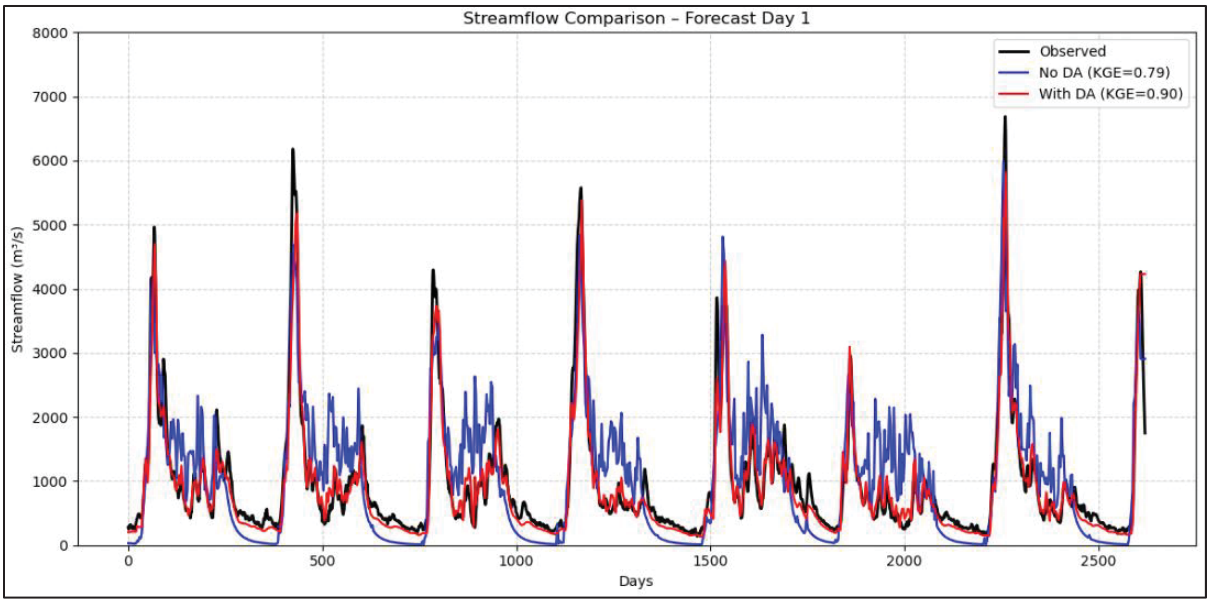

Figure 3.18 Comparaison entre les débits observés et les prévisions déterministes obtenues à partir de la moyenne des ensembles pour le modèle Blended pour l'horizon de prévision 1 jour

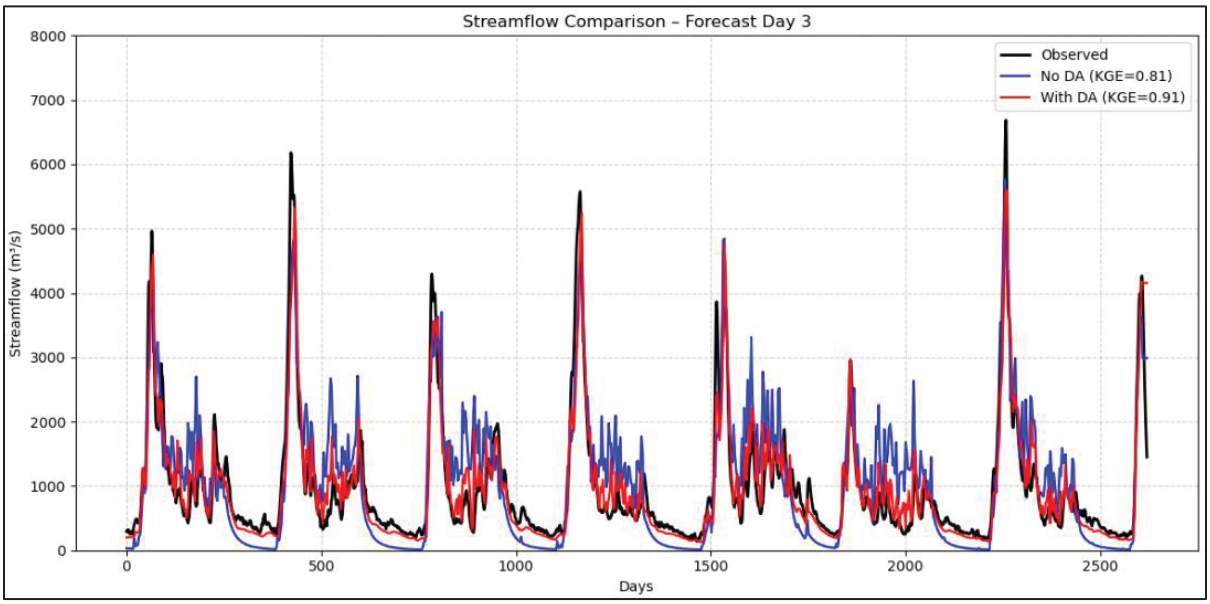

Figure 3.19 Comparaison entre les débits observés et les prévisions déterministes obtenues à partir de la moyenne des ensembles pour le modèle Blended pour l'horizon de prévision 3 jours

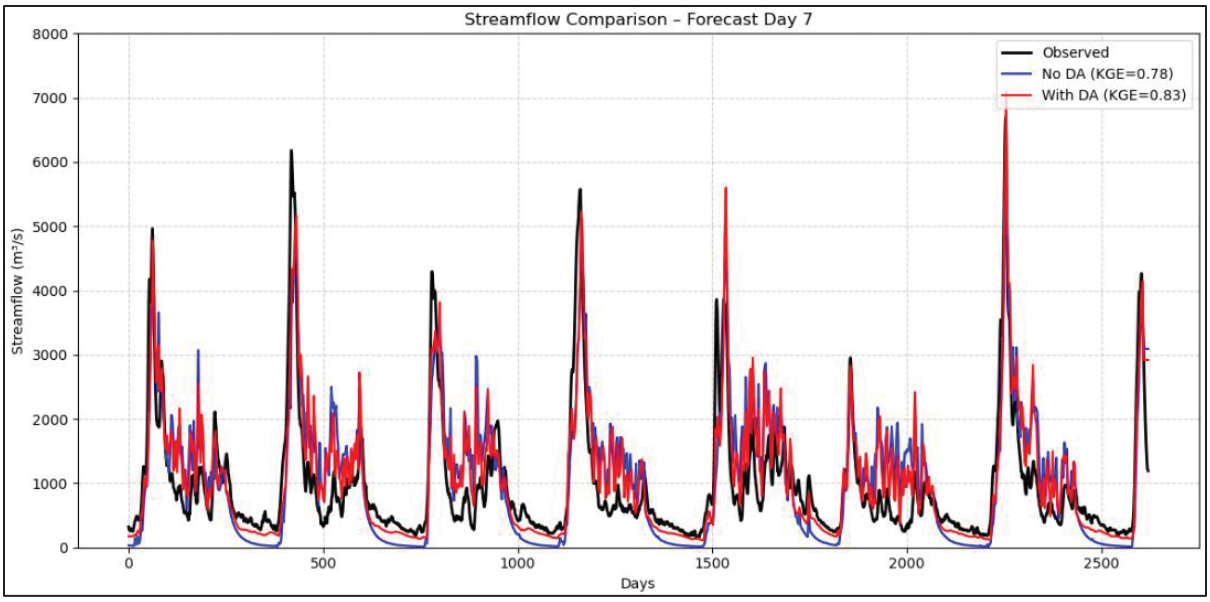

Figure 3.20 Comparaison entre les débits observés et les prévisions déterministes obtenues à partir de la moyenne des ensembles pour le modèle Blended pour l'horizon de prévision 7 jours

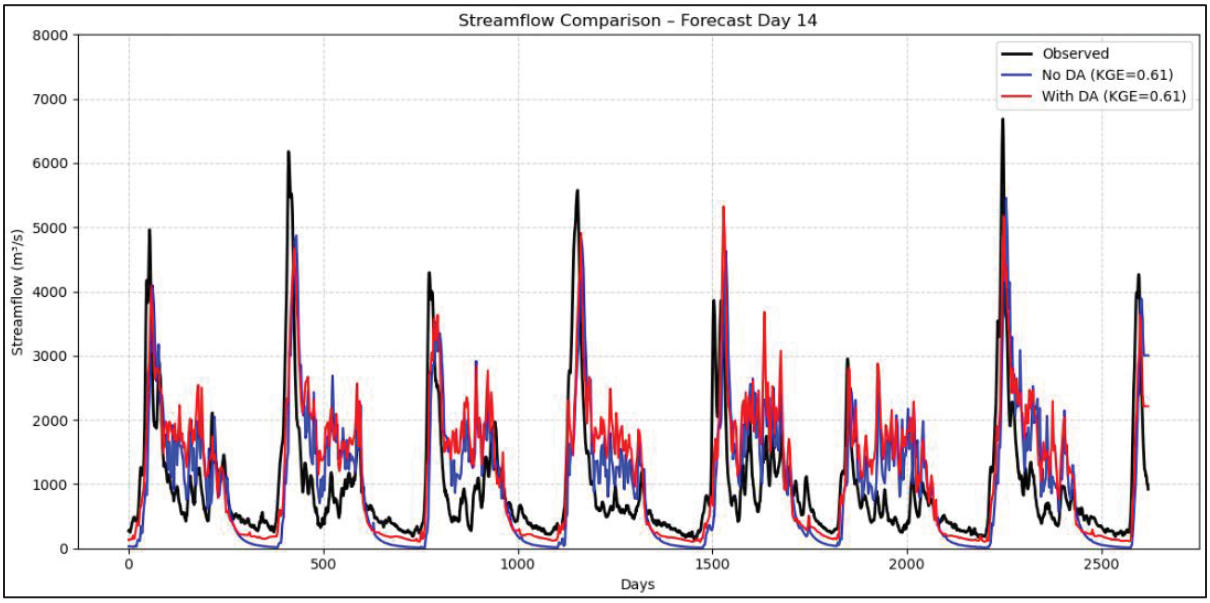

Figure 3.21 Comparaison entre les débits observés et les prévisions déterministes obtenues à partir de la moyenne des ensembles pour le modèle Blended pour l'horizon de prévision 14 jours

Les figures 3.22 à 3.25 présentent la comparaison entre les débits observés et les prévisions déterministes obtenues à partir de la moyenne des ensembles pour le modèle MOHYSE, avec et sans assimilation de données, aux horizons de prévision, 1, 3, 7 et 14 jours. Dans le cas sans assimilation de données, les pics de crues sont bien observables, mais leur ampleur est sous-estimée. En période de faibles débits, le modèle prévoit fréquemment des débits nuls. De plus, à mesure que l'horizon de prévision s'allonge, les débits ont tendance à être de plus en plus surestimés entre les périodes de crues. Cette dégradation progressive de la performance se reflète dans la diminution des valeurs du KGE qui baisse progressivement avec l'augmentation de l'horizon de prévision. L'assimilation de données semble avoir peu d'impact sur la performance du modèle MOHYSE. En effet, les courbes des débits avec et sans assimilation sont presque visuellement superposées à certains moments. Le KGE diminue progressivement plus l'horizon prédictif s'allonge et il reste comparable à celui sans assimilation voire même identique à certains moments. De plus, les difficultés à représenter correctement les faibles débits persistent, entre autres avec la présence continue de plusieurs débits nuls. Par ailleurs, les pics de crues sont encore sous-estimés.

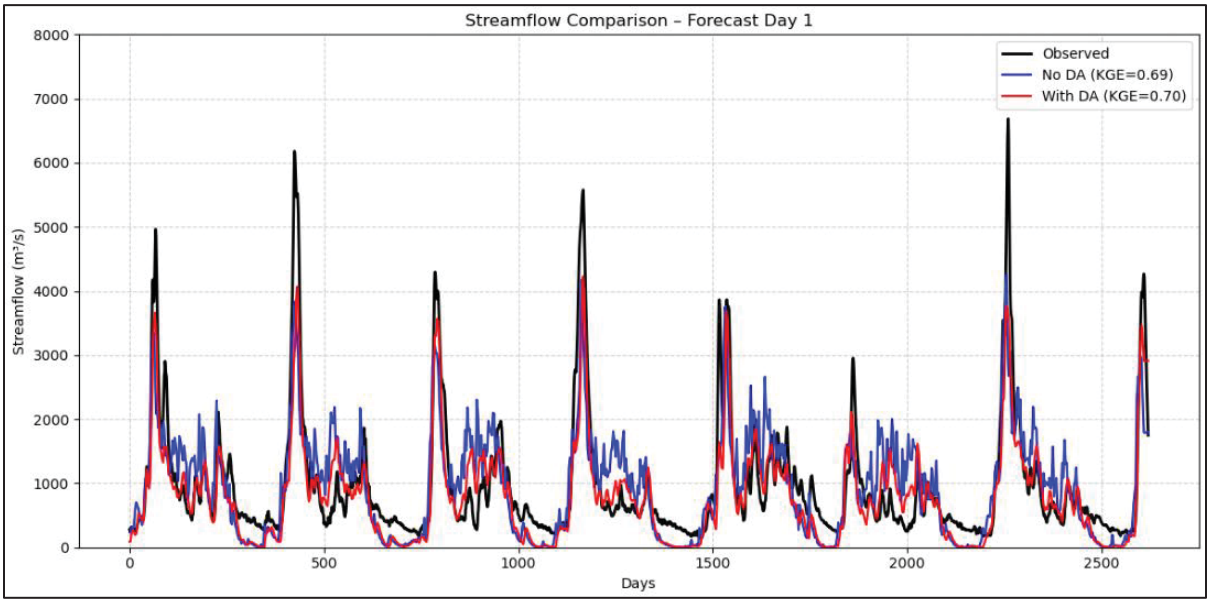

Figure 3.22 Comparaison entre les débits observés et les prévisions déterministes obtenues à partir de la moyenne des ensembles pour le modèle MOHYSE pour l'horizon de prévision 1 jour

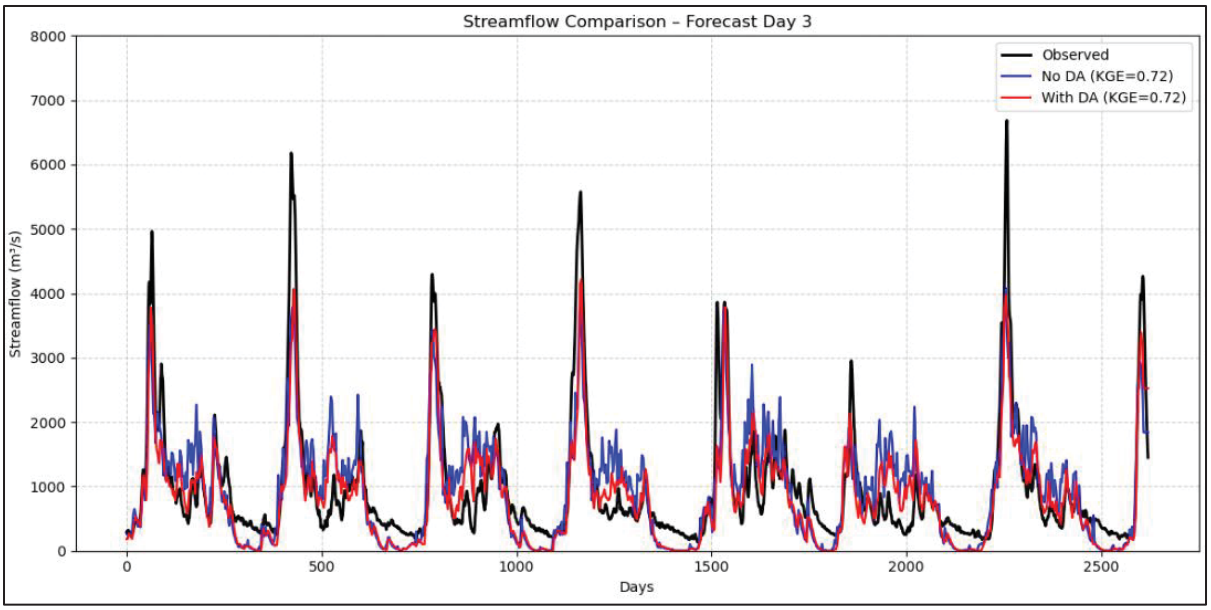

Figure 3.23 Comparaison entre les débits observés et les prévisions déterministes obtenues à partir de la moyenne des ensembles pour le modèle MOHYSE pour l'horizon de prévision 3 jours

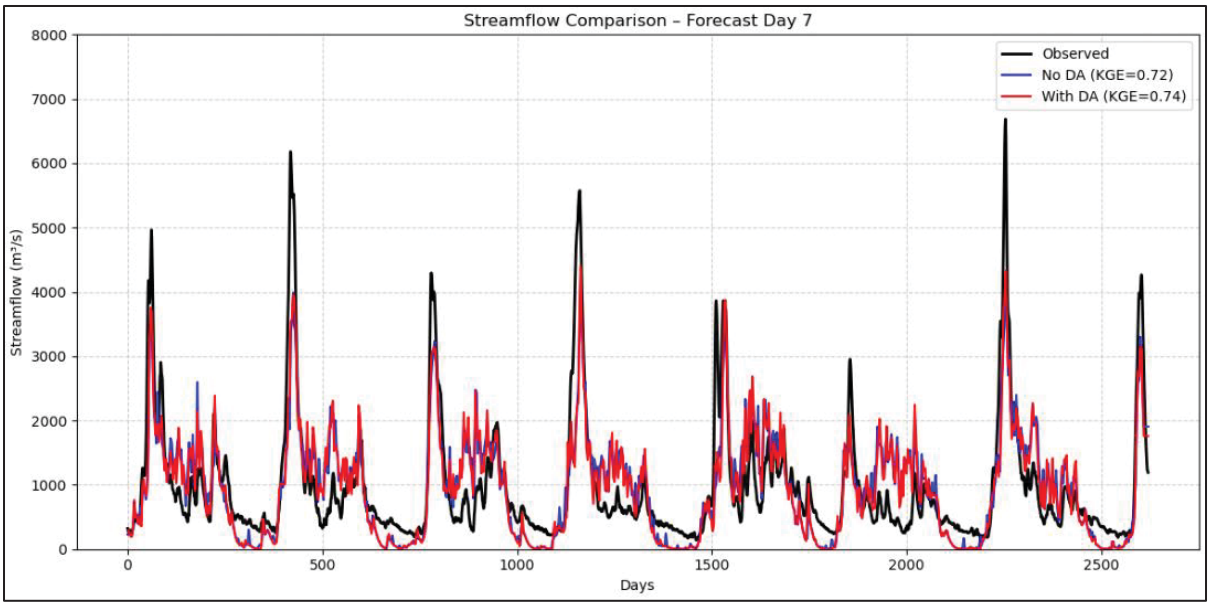

Figure 3.24 Comparaison entre les débits observés et les prévisions déterministes obtenues à partir de la moyenne des ensembles pour le modèle MOHYSE pour l'horizon de prévision 7 jours

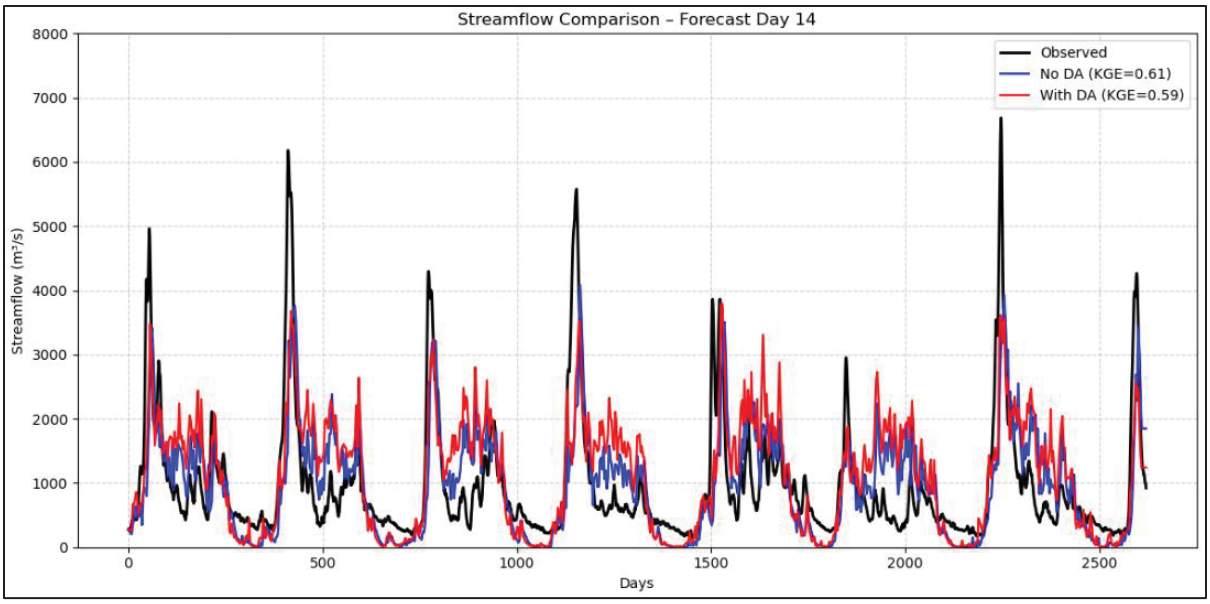

Figure 3.25 Comparaison entre les débits observés et les prévisions déterministes obtenues à partir de la moyenne des ensembles pour le modèle MOHYSE pour l'horizon de prévision 14 jours

CHAPITRE 4

DISCUSSION

Cette section présente une analyse plus approfondie des résultats présentés au chapitre précédent. Elle vise à mieux comprendre l'effet de l'assimilation de données sur la performance des prévisions hydrologiques, selon les modèles, les saisons, les horizons de prévision et les résultats de calage. Les sous-sections permettront de présenter ces différents aspects, en comparant également les résultats obtenus à ceux d'autres études. Enfin, un retour sur les objectifs est effectué, suivi d'une discussion sur les principales limites de l'étude.

4.1 Analyse entre les modèles

Les résultats mettent en évidence des différences entre les trois modèles, tant sur leurs performances que sur l'impact de l'assimilation de données sur chacun d'entre eux. Blended a obtenu le meilleur NSE en calage, mais c'est le modèle GR4J-CN qui se démarque en mode prévision avec assimilation de données en affichant de meilleurs résultats en termes de précision. Les meilleurs résultats de GR4J-CN pourraient s'expliquer par sa structure simple qui semble permettre de mieux gérer la correction des états initiaux. Cette efficacité se reflète dans les scores de CRPS globalement plus faible que ceux obtenus avec les deux autres modèles. De plus, GR4J-CN offre des prévisions très resserrées, mais présente des biais systématiques dans la majorité des cas. D'un autre côté, le modèle MOHYSE réagit très peu à l'assimilation, malgré de bonne performance en calage. En effet, ses résultats restent stables et il reproduit mal les faibles débits et tend à sous-estimer l'intensité des crues. Sa structure interne semble moins adaptée à une mise à jour efficace des états internes, ce qui limite sa performance en prévision. En effet, MOHYSE affiche les scores de CRPS les plus élevés pour plusieurs jours de prévision. De son côté, le modèle Blended, malgré une structure plus complexe que les deux autres modèles, offre une bonne performance en calage, mais des résultats variables en prévision. Sa structure issue de la combinaison de plusieurs sous-modèles avec différentes pondérations peut complexifier la gestion des états internes, ce qui pourrait expliquer les performances observées. Ses scores de CRPS indiquent une performance

intermédiaire. Il offre une meilleure performance que MOHYSE, mais souvent moins bonne que GR4J-CN, surtout pour les plus longs horizons de prévision. En somme, GR4J-CN est le modèle qui se distingue par sa précision, tandis que MOHYSE présente des prévisions plus dispersées et moins précises et que Blended occupe une position intermédiaire.

4.2 Impact de l'assimilation de données

L'impact de l'assimilation de données par le filtre d'ensemble de Kalman diffère selon le modèle. Le modèle GR4J-CN bénéficie clairement de l'assimilation de données, avec une amélioration des performances en prévision, comme le montrent les KGE plus élevés obtenus après l'assimilation de données. Cette tendance est corroborée par les CRPS, où GR4J-CN affiche les scores les plus faibles et la meilleure amélioration parmi les trois modèles. Cette amélioration confirme la capacité du modèle à intégrer efficacement les corrections des états initiaux par la méthode EnKF. De plus, les diagrammes de Talagrand montrent une légère amélioration, avec une répartition plus équilibrée des observations sur l'ensemble des intervalles, et ce pour les horizons courts. De l'autre côté, le modèle MOHYSE réagit peu à l'assimilation de données, avec des résultats très similaires, voire inférieurs à ceux obtenus avec le scénario sans assimilation de données. Cette difficulté à tirer parti de l'assimilation de données est observable également sur les diagrammes de Talagrand, où l'on observe que l'assimilation n'améliore pratiquement pas la répartition des observations dans l'ensemble. L'assimilation de données améliore légèrement les résultats aux échéances courtes, mais les observations restent souvent concentrées aux extrémités. Cette faible amélioration est également observée dans les résultats du CRPS, qui évoluent peu et restent relativement élevés même après l'assimilation de données. Une hypothèse est que la structure interne de MOHYSE ne favorise pas la convergence des états avec l'assimilation de données avec la technique EnKF, comme le suggèrent certains travaux (Thiboult & Anctil, 2015). En effet, cette étude démontre que les modèles hydrologiques ne bénéficient pas tous de manière équivalente de la mise à jour des états par l'EnKF, certains présentent une faible convergence ou une instabilité des états corrigés. Cela peut être dû au choix d'hyperparamètres employé par rapport à la structure du modèle. Le modèle Blended, de son côté, présente une légère amélioration quand

l'assimilation de données est effectuée. Celle-ci améliore la simulation des faibles débits. Les diagrammes de Talagrand obtenus avec assimilation de données, révèlent une légère augmentation des fréquences dans les intervalles intermédiaires, mais qu'une concentration persiste aux extrémités. Les résultats de CRPS de ce modèle montrent une amélioration plus prononcée aux horizons courts. Comme il s'agit d'un modèle ayant une structure complexe, issue de la combinaison de plusieurs sous-modèles avec des pondérations et mises à jour différentes selon le modèle utilisé, cela peut limiter la capacité du modèle à bien intégrer les mises à jour des états. De plus, le choix des variables d'état mises à jour peut également jouer un rôle dans la réussite de l'assimilation. Ainsi, il se pourrait que les états mis à jour par l'assimilation dans cette étude aient peu d'impact sur les modèles. De plus, les diagrammes de Talagrand des trois modèles montrent que l'assimilation de données contribue peu à renforcer la fiabilité des prévisions pour l'ensemble des modèles. Les gains apportés par l'assimilation de données sont surtout visibles aux courts horizons. Par conséquent, l'assimilation de données par le filtre d'ensemble de Kalman ne constitue pas une solution universelle puisqu'elle peut améliorer la qualité des prévisions lorsqu'elle fonctionne bien, mais elle peut également détériorer les prévisions comme il a été observé avec les hydrogrammes du modèle MOHYSE.

4.3 Analyse des performances selon les saisons

L'analyse des performances des modèles hydrologiques selon les saisons est essentielle pour mieux comprendre les processus hydrologiques en contexte nordique. Puisque chaque saison a ses conditions climatiques propres, cela influence le régime hydrologique. L'automne se distingue par des précipitations plus régulières et une évapotranspiration réduite, menant à une augmentation des débits. En hiver, les basses températures favorisent l'accumulation de neige et limitent le ruissellement, tandis qu'au printemps, la fonte rapide des neiges, sur un sol encore gelé, génère des hausses de débits difficiles à anticiper. Enfin, l'été est marqué par des périodes sèches ou des évènements pluvieux intenses, ce qui augmente la variabilité des débits. La performance des modèles hydrologiques fluctue selon les saisons, en grande partie en raison de cette variabilité. En effet, les résultats obtenus dans cette étude mettent en évidence que la performance des prévisions hydrologiques varie selon les saisons. En hiver, les CRPS sont plus

faibles et les ensembles sont moins dispersés, tandis que les diagrammes de Talagrand montrent une meilleure couverture des observations particulièrement pour GR4J-CN. Cette performance s'explique en partie par les conditions météorologiques plus stables, dominées par l'accumulation de neige, ce qui entraine une variabilité plus faible des débits à court terme. Elle est aussi liée à la simplicité des processus hydrologiques actifs en hiver, avec peu d'évènements de pluie, peu ou pas de fonte de neige et une prédominance des débits de base. Le printemps correspond à la période où les prévisions hydrologiques présentent les plus faibles performances, tous modèles confondus, avec des CRPS les plus élevés et une variabilité accrue. L'assimilation de données contribue à une légère amélioration des résultats plus précisément pour Blended et GR4J-CN. Cependant, son impact reste limité comparativement à d'autres saisons comme l'hiver. Cette période de transition hydrologique engendre une complexité que les trois modèles ont du mal à capter. L'été est également caractérisé par une moins bonne performance des prévisions. Les valeurs de CRPS sont élevées, la dispersion des ensembles est importante et les observations se trouvent fréquemment en dehors des intervalles de prévision. Cependant, l'assimilation de données permet une amélioration des prévisions pour le modèle GR4J-CN. En revanche, les modèles Blended et MOHYSE ne semblent pas bénéficier de la même amélioration pour cette saison. Enfin, en automne, les modèles Blended et GR4J-CN affichent des performances relativement stables et sont les plus sensibles à l'assimilation de données. Il est donc clair que la saisonnalité influence fortement la performance des prévisions. Les meilleures performances sont observées en hiver, tandis que l'été et le printemps représentent les périodes les plus complexes à modéliser. Cette performance hivernale peut s'expliquer par la stabilité des conditions météorologiques et le fait qu'en hiver, les processus hydrologiques sont moins nombreux et plus simples. En effet, les précipitations sont généralement solides, les températures sont plus stables et l'essentiel des écoulements est lié aux débits de base. Ainsi, ces conditions réduisent l'incertitude des prévisions. À l'inverse, les saisons plus chaudes sont caractérisées par la fonte des neiges, des précipitations intenses et des interactions complexes entre ruissellement, infiltration et évapotranspiration, ce qui engendre une dynamique hydrologique plus difficile à représenter de manière fiable dans les modèles. Le modèle GR4J-CN est celui qui bénéficie le plus de l'assimilation de données, toutes saisons confondues.

4.4 Performance de calage vs performance en prévision

Un constat intéressant ressort de la comparaison entre les performances obtenues lors du calage des modèles et celles issues des prévisions hydrologiques. En effet, bien que le modèle GR4J-CN ait obtenu le plus faible NSE lors du calage, il est celui qui présente les meilleures performances en prévision lorsque l'assimilation de données est effectuée. Les hydrogrammes montrent que le modèle GRJ4-CN est celui qui bénéficie le plus de l'assimilation de données. En effet, GR4J-CN parvient à bien reproduire les débits observés et présente la plus grande amélioration du KGE. MOHYSE, pourtant mieux calibré, tend à davantage sous-estimer ou surestimer certains débits comparativement à GR4J-CN. Cette tendance s'observe sur l'ensemble des horizons de prévisions lorsque l'assimilation de données est effectuée. À certains horizons de prévision, GR4J-CN performe mieux que le modèle Blended, lequel avait pourtant obtenu le meilleur NSE lors du calage. Ainsi, cela met en évidence que le modèle GR4J-CN est celui qui bénéficie le plus de la correction des états initiaux, ce qui permet d'améliorer la qualité de ses prévisions, et ce en dépit d'un calage moins performant. Ce résultat met en lumière le fait qu'un bon calage sur les données historiques ne garantit pas nécessairement de bonnes performances en prévision, point également soulevé par Davidson-Chaput et al. (2025). Par conséquent, ce constat souligne le fait que le calage, bien qu'essentiel, ne devrait pas être le seul critère d'évaluation d'un modèle hydrologique pour un usage en prévision hydrologique.

4.5 Effet de l'horizon de prévision

Les résultats obtenus montrent une tendance claire par rapport à la longueur de l'horizon de prévision, où plus l'horizon de prévision est lointain, plus la performance diminue. En effet, à mesure que l'horizon de prévision augmente, les erreurs s'accumulent, la dispersion des ensembles s'élargit et la capacité de modèles à inclure les observations dans les intervalles de prévision augmente. Toutefois, une trop grande dispersion n'est pas nécessairement signe de bonne performance. En réalité, une trop grande dispersion peut nuire à la qualité des prévisions, car les ensembles deviennent si larges qu'ils capturent certes l'observation, mais sans valeur réelle utile. Cette couverture trop vaste réduit la précision des prévisions, les membres

s'éloignent davantage des observations et se traduit par des scores CRPS de plus en plus élevés à mesure que l'horizon de prévision augmente. On peut expliquer cette tendance par l'augmentation intrinsèque de l'incertitude liée aux prévisions météorologiques qui deviennent de moins en moins fiables à mesure que l'horizon de prévision s'allonge. Cette dégradation de la performance affecte à la fois la précision des prévisions et leur fiabilité. Cette tendance est observée dans toutes les saisons, mais elle est plus prononcée en été et au printemps. Malgré l'assimilation de données, cette dégradation avec le temps reste visible. En effet, l'assimilation de données améliore les conditions initiales et réduit l'erreur, mais son effet tend à s'atténuer au fil du temps, car les incertitudes météorologiques tendent à dominer.

4.6 Comparaison avec d'autres études

Les résultats de l'étude de Dion et al (2021), qui avait pour objectif de développer une méthodologie combinant l'assimilation de données, le post-traitement et l'approche multi-modèle, ont été comparés à ceux de la présente étude. Les résultats obtenus dans la présente étude sont cohérents avec ceux présentés dans l'étude de Dion et al. (2021). En effet, comme observé ici, les résultats obtenus par Dion et al. (2021) ont montré que l'assimilation de données réduit les biais systématiques et améliore la dispersion des prévisions. De plus, l'étude montre également que l'assimilation de données est particulièrement efficace aux horizons de prévisions courts (1 à 3 jours), mais que son impact diminue avec l'augmentation de l'horizon de prévision. Concernant les modèles, Dion et al. (2021) ont identifié que GR4J était le modèle le plus performant après assimilation en termes de scores KGE, ce qui est cohérent avec les résultats obtenus ici, où GR4J-CN est le modèle qui bénéficie le plus de la correction des états initiaux. En outre, l'étude de Dion et al. (2021) montre que l'assimilation de données améliore de manière générale la performance des prévisions d'ensembles, mais que de biais subsistent pour certaines saisons telles que DJFM et JJA, un constat également observé dans la présente étude. Il est important de noter que, contrairement aux résultats de Dion et al. (2021), où l'assimilation de données semble améliorer les prévisions dans tous les cas, l'étude présente montre que l'impact de l'assimilation peut varier selon le modèle utilisé, avec des cas où elle entraine peu d'améliore voire une légère dégradation.

Une autre étude récente, également menée dans le bassin versant du LSJ, est celle de Armstrong et al. (2024), qui propose une approche hybride multi-modèle combinant huit modèles hydrologiques, un modèle semi-distribué et un modèle d'apprentissage profond. L'objectif de l'étude était de déterminer si l'apprentissage profond combiné à des modèles et de l'assimilation de données permet d'améliorer les prévisions. Les résultats montrent que pour le modèle semi-distribué CEQUEAU7, l'intégration de l'assimilation de données permet d'améliorer la précision des prévisions, et ce particulièrement en hiver. En effet, les scores CRPS obtenus sont réduits en moyenne de 43%, ce qui rejoint également les observations de la présente étude, où l'assimilation de données est également plus efficace aux saisons froides. De plus, l'étude de Armstrong et al. (2024) montre que l'intégration du modèle LSTM et du modèle GRJ4 améliore la qualité des prévisions hydrologiques pour toutes les saisons et tous les horizons de prévision. Qui plus est, l'intégration du modèle LSTM au modèle CEQUEAU7 avec assimilation de données permet d'obtenir l'amélioration la plus significative, avec une réduction moyenne du CRPS de 50%, atteignant jusqu'à 70% en été par rapport au modèle CEQUEAU7 sans assimilation de données.

Ainsi, malgré le fait que les approches et les contextes d'étude diffèrent et que dans certains cas l'impact spécifique de l'assimilation de données soit plus difficile à isoler, l'ensemble des résultats montre que lorsque l'assimilation de données est effectuée, les prévisions hydrologiques sont plus performantes.

4.7 Retour sur les objectifs

L'objectif principal était d'analyser et de quantifier l'impact de l'assimilation de données par le filtre d'ensemble de Kalman sur la caractérisation de l'incertitude dans les prévisions hydrologiques. Plus précisément, il s'agissait d'évaluer dans quelle mesure l'assimilation de données permet de corriger les états initiaux des modèles et par conséquent, d'améliorer la fiabilité et la précision des prévisions.

Les résultats obtenus répondent en grande partie à cet objectif. En comparant les configurations avec et sans assimilation de données, il a été possible d'observer une réduction des erreurs, une meilleure couverture des observations et dans quelques cas, une diminution de la sous-dispersion des ensembles. Ces effets sont particulièrement observables pour le modèle GR4J-CN qui a montré des gains de performance dans toutes les saisons, tandis que d'autres modèles, comme MOHYSE, se sont révélés moins sensibles à l'assimilation de données. L'étude a également permis de mieux cerner l'influence des conditions initiales sur la qualité des prévisions, surtout en hiver. À l'inverse, au printemps et en été, la forte variabilité des débits et la complexité des processus hydrologiques ont limité l'impact de la correction des états initiaux.

Ainsi, les analyses menées à l'aide des indicateurs de performance ont permis de répondre aux objectifs, tout en relevant les conditions dans lesquelles l'assimilation de données est la plus bénéfique pour améliorer la performance des modèles hydrologiques.

4.8 Limitations

Cette section présente les limitations de l'étude ainsi que l'impact potentiel qu'aurait l'utilisation de données ou de configurations différentes. Les principales limites abordées sont les modèles hydrologiques utilisés, les données météorologiques utilisées, la zone d'étude restreinte à un seul bassin versant, ainsi que certains choix méthodologiques liés à l'assimilation de données.

4.8.1 Modèles hydrologiques

L'étude a été menée sur un nombre restreint de modèles hydrologiques, se limitant à trois modèles conceptuels globaux. Ce choix limite la généralisation des conclusions à d'autres types de modèles. En effet, d'autres types de modèles tels que les modèles semi-distribués ou à base physique qui possèdent des structures et des représentations des processus hydrologiques différentes pourraient réagir différemment à l'assimilation de données, particulièrement dans la correction des états internes. Cependant, l'application de méthodes

d'assimilation de données à des modèles distribués est beaucoup plus complexe et n'aurait pas été possible dans le contexte de l'utilisation de la plateforme PAVICS-Hydro. Néanmoins, étant donné que chaque modèle hydrologique est unique et conçu selon une structure et des hypothèses spécifiques, l'utilisation d'autres types de modèles pourrait influencer les résultats obtenus.

4.8.2 Zone d'étude

La méthodologie proposée a été vérifiée seulement sur le bassin versant du LSJ dont les caractéristiques hydrologiques et climatiques lui sont spécifiques. Ce bassin est soumis à un régime hydrologique contrôlé par la fonte de neige au printemps. Ainsi, les conclusions tirées dans cette étude s'appliquent principalement à des bassins versants similaires à celui-ci. En effet, la performance de la méthodologie pourrait varier sur des bassins présentant une climatologie, une superficie ou un régime hydrologique différent. Par conséquent, il serait pertinent d'étendre l'étude à des bassins différents afin d'évaluer si l'impact de l'assimilation de données est similaire dans des conditions différentes.

4.8.3 Prévisions météorologiques

Les prévisions météorologiques d'ensemble ECMWF utilisées dans cette étude couvrent seulement la période allant de juin 2016 à mars 2023, avec un horizon de prévision de 14 jours. Cette période de 7 ans a permis d'évaluer la performance de la méthodologie dans des conditions météorologiques variées. Cependant, une période plus longue aurait permis d'obtenir des résultats plus robustes qui intègrent davantage d'évènements météorologiques extrêmes tels que des crues historiques, des épisodes de précipitations intenses ou des périodes de sécheresse prolongées. En outre, puisque l'étude est réalisée avec un horizon de prévision de 14 jours, cela restreint la portée des conclusions à cette période de prévision. Ainsi, un horizon plus long aurait pu permettre d'analyser l'impact de l'assimilation de données sur des horizons à long terme. Par ailleurs, les données météorologiques utilisées pourraient être combinées à d'autres sources que celles de ECMWF comme celles fournies par ECCC et par le *National Oceanic and Atmospheric Administration* afin de diversifier les intrants

météorologiques. Comparer plusieurs sources de données météorologiques permettrait non seulement de mieux comprendre la sensibilité des prévisions hydrologiques aux prévisions météorologiques, mais aussi de réduire les biais potentiels.

4.8.4 Choix méthodologiques de l'assimilation de données

L'assimilation de données a été réalisée uniquement à l'aide du EnKF en utilisant un nombre fixe de membres d'ensemble. Ainsi, bien que l'EnKF soit largement utilisé et reconnu dans la littérature, explorer d'autres méthodes d'assimilation, tel que les filtres particulaires ou l'assimilation variationnelle pourraient conduire à des résultats différents. De plus, plusieurs choix méthodologiques auraient pu être explorés afin d'améliorer potentiellement les résultats obtenus. Par exemple, augmenter le nombre de membres d'ensemble permettrait d'évaluer si cela améliore la représentation de l'incertitude et la fiabilité des prévisions. La sélection des variables internes mises à jour lors de l'assimilation pourrait également être examinée plus en détail afin d'évaluer leur impact sur les prévisions. Qui plus est, l'ajustement des hyperparamètres associés aux perturbations des précipitations et des températures, tel que l'écart-type ou la nature des distributions utilisées constitue un choix méthodologique qui influence les résultats. Ainsi, tous ces choix méthodologiques sont susceptibles d'affecter l'efficacité globale de l'assimilation de données.

CONCLUSION

Ce mémoire présente l'effet de l'assimilation de données par le filtre d'ensemble de Kalman sur la performance des prévisions hydrologiques. L'objectif principal était d'évaluer dans quelle mesure l'assimilation de données permet de corriger les états initiaux d'un modèle hydrologique et ainsi améliorer la fiabilité et la précision des prévisions hydrologiques. Pour ce faire, trois modèles hydrologiques conceptuels globaux ont été utilisés sur le bassin versant du LSJ, en y intégrant un EnKF avec 25 membres combinés aux 50 membres de prévisions météorologiques d'ensemble de ECMWF, sur un horizon de 14 jours.

Les performances ont été évaluées selon différents critères, soit le CRPS et les diagrammes de Talagrand, en tenant compte de la saisonnalité, des horizons de prévisions et des différents modèles. L'étude montre que l'assimilation de données améliore la qualité des prévisions, en réduisant les erreurs et en augmentant la capacité des ensembles à inclure les observations, surtout pour GRJ4-CN. Cependant, les bénéfices de l'assimilation de données varient selon le modèle hydrologique. En effet, l'étude montre qu'elle peut améliorer les prévisions, mais que son efficacité dépend de différents facteurs. Par exemple. GR4J-CN a largement bénéficié de l'assimilation de données, tandis que MOHYSE a montré peu d'amélioration. Ainsi, intégrer une méthode d'assimilation doit être fait en tenant compte des modèles utilisés, des états assimilés et de la technique d'assimilation choisie. Ces résultats montrent l'importance du choix des états assimilés et de la structure du modèle. De plus, l'amélioration est plus prononcée aux horizons courts et tend à diminuer avec le temps, en raison de l'incertitude croissante des prévisions météorologiques.

L'analyse saisonnière a révélé que l'hiver et l'automne sont les saisons où les prévisions sont les plus fiables, tandis qu'elles tendent à être moins fiables en été et au printemps, en raison de la variabilité des précipitations et de la fonte des neiges. De plus, l'étude a permis de montrer que la performance en calage ne garantit pas nécessairement une bonne performance en prévision. Le modèle GR4J-CN, qui avait le plus faible NSE en calage, s'est révélé performant en prévision une fois l'assimilation de données appliquée. Ainsi, le choix du modèle ne doit

pas uniquement se baser sur la performance au calage, mais aussi sur sa capacité à intégrer l'incertitude et sa capacité à corriger ses états. Malgré le fait qu'il soit le plus précis, GR4J-CN n'est pas le plus fiable. En effet, il offre des prévisions très resserrées, mais présente des biais systématiques dans la majorité des cas. Les prévisions des différents membres sont regroupées et proches les unes des autres, mais elles sont également situées en dessous ou au-dessus de la valeur des observations, ce qui fait que la couverture des observations est insuffisante.

Enfin, bien que l'assimilation de données par l'EnKF se soit montrée bénéfique dans plusieurs cas, son efficacité dépend de nombreux facteurs : le modèle utilisé, les états corrigés, la saison et l'horizon de prévision.

Les prochaines études pourraient inclure l'évaluation de cette méthodologie sur d'autres types de modèles, l'analyse d'autres bassins versants aux caractéristiques différentes ou encore tester d'autres techniques d'assimilation de données. Ces perspectives pourraient permettre de mieux cerner les conditions dans lesquelles l'assimilation de données offre le plus de valeur ajoutée en contexte opérationnel pour améliorer la gestion des ressources en eau.

RECOMMANDATIONS

Face aux limitations identifiées dans l'étude et aux perspectives identifiées, les recommandations suivantes visent à améliorer la qualité des recherches futures sur la prévision hydrologique :

1) Inclure un plus grand éventail de modèles tels que des modèles distribués ou semi-distribués afin d'évaluer si une représentation spatiale plus fine influence la performance des prévisions. L'intégration de modèles à base physique pourrait également enrichir l'analyse. Ces ajouts pourraient contribuer à une meilleure généralisation et reproductibilité des conclusions obtenues.
2) Élargir les recherches à plusieurs bassins versants ayant des caractéristiques climatiques, météorologiques et physiographiques différentes. Cela permettrait de mieux comprendre l'influence du contexte géographique sur la performance des modèles et d'évaluer si l'impact de l'assimilation de données est similaire dans des conditions environnementales différentes.
3) Examiner les incertitudes météorologiques en amont. Une étape de post-traitement des prévisions météorologiques pour corriger les biais et à ajuster leur fiabilité pourrait permettre d'évaluer dans quelle mesure la qualité des prévisions météorologiques influence la performance des prévisions hydrologiques améliorer les prévisions météorologiques permettrait de voir si cela peut contribuer à améliorer la fiabilité des prévisions hydrologiques.
4) Explorer d'autres méthodes d'assimilation pourrait permettre d'obtenir des résultats différents. Par ailleurs, ajuster le nombre de membres de l'ensemble pourrait influencer les performances des modèles en captant mieux la variabilité et l'incertitude. Une autre option pourrait être de modifier la sélection des états du modèle qui sont assimilés pour voir si certains états sont plus sensibles que d'autres.
5) Tester des horizons de prévisions plus longs permettrait d'examiner l'effet de l'assimilation de données sur des horizons à plus long terme puisque les incertitudes sont plus grandes et donc la performance des modèles pourrait évoluer différemment.

ANNEXE I

CODES PYTHON UTILISÉS POUR LE PROJET

```
import copy
import shutil
import numpy as np
import pandas as pd
from ravenpy.config import commands as rc
from ravenpy.config import options as o
from ravenpy import EnsembleReader, Emulator
import matplotlib.pyplot as plt
import datetime as dt
import spotpy
from ravenpy.utilities.calibration import SpotSetup
from ravenpy.config.emulators import GR4JCN #À changer selon le modèle désiré
import xarray as xr
from pathlib import Path
import xskillscore as xs

def define_land_hru(area, longitude, latitude, elevation, slope):

    hru = dict(
        area=area,
        elevation=elevation,
        latitude=latitude,
        longitude=longitude,
        slope=slope,
        hru_type="land",
    )
    return hru

def get_ERA5_data_tags(elevation, latitude, longitude):

    data_kwds = {"ALL": {"elevation": elevation,
                         'latitude': latitude,
                         'longitude': longitude}}

    # Here, we need to give the name of your different dataset in order to match with Raven
models.
    alt_names = {
        "PRECIP": "pr",
        "TEMP_MIN": "tmin",
```

```
        "TEMP_MAX": "tmax",
    }
    data_type = ["PRECIP", "TEMP_MIN", "TEMP_MAX"]

    return data_kwds, alt_names, data_type

def setup_ERA5_gauge(ERA5_data_path, data_type, alt_names, data_kwds):

    gauge = [
        rc.Gauge.from_nc(
            ERA5_data_path,
            data_type=data_type,
            alt_names=alt_names,
            data_kwds=data_kwds,
        )
    ]

    return gauge

def calibrate_GR4J_model(calibration_start_date, calibration_end_date, qobs_data_path,
gauge, hru):
    eval_metrics = ("NASH_SUTCLIFFE",)

    # We want to skip the first year for model evaluation
    evaluation_start_date = calibration_start_date + dt.timedelta(days=365)
    evaluation_period = rc.EvaluationPeriod(name="CALIBRATION",
start=evaluation_start_date, end=calibration_end_date)

    # We need to create the desired model with its parameters
    model_config = GR4JCN(
        ObservationData=[rc.ObservationData.from_nc(qobs_data_path, alt_names={"qobs":
"qobs"})],
        RainSnowFraction='RAINSNOW_DINGMAN',
        Evaporation='PET_OUDIN',
        Gauge=gauge,
        HRUs=[hru],
        StartDate=calibration_start_date,
        EndDate=calibration_end_date,
        RunName="test",
        EvaluationMetrics=eval_metrics,
        SuppressOutput=True,
        EvaluationPeriod=[evaluation_period],

    )
```

```
    # In order to calibrate your model, you need to give the lower and higher bounds of the model.
    low_params = (0.01, -15.0, 10.0, 0.0, 1.0, 0.0)
    high_params = (2.5, 10.0, 700.0, 7.0, 30.0, 1.0)

    spot_setup = SpotSetup(
        config=model_config,
        low=low_params,
        high=high_params,
    )

    # Number of total model evaluations in the calibration. This value should be over 500 for real optimisation,
    # and upwards of 10000 evaluations for models with many parameters.
    model_evaluations = 5000

    # Set up the spotpy sampler with the method, the setup configuration, a run name and other options. Please refer to
    sampler = spotpy.algorithms.dds(spot_setup, dbname="RAVEN_model_run", dbformat="ram", save_sim=False)

    # Launch the actual optimization. Multiple trials can be launched, where the entire process is repeated and
    # the best overall value from all trials is returned.
    sampler.sample(model_evaluations, trials=1)

    # Get all the values of each iteration
    results = sampler.getdata()
    print("The best Nash-Sutcliffe value is : ")

    bestindex, bestobjfun = spotpy.analyser.get_maxlikeindex(results)  # Want to get the MAX NSE
    best_model_run = list(results[bestindex][0])  # Get the parameter set returning the best NSE
    # Remove the NSE value (position 0) and the ID at the last position to get the actual parameter set.
    optimized_parameters_GR4J = best_model_run[1:-1]

    # Display the parameter set ready to use in a future run:
    print("\nThe best parameters are : ")
    print(optimized_parameters_GR4J)

    return optimized_parameters_GR4J, bestobjfun

def get_default_GR4J_config(hru, qobs_data_path, optimized_parameters_GR4J):
```

```
    default_emulator_config = dict(
        HRUs=[hru],
        ObservationData=[rc.ObservationData.from_nc(qobs_data_path, alt_names="qobs")],
        RainSnowFraction='RAINSNOW_DINGMAN',
        Evaporation='PET_OUDIN',
        params=optimized_parameters_GR4J,
        # GR4JCN needs an extra constant parameter for the catchment, corresponding to the G50
parameter in CEMANEIGE
        # description. Here is how we provide it.
        GlobalParameter={"AVG_ANNUAL_SNOW": 408.480},
    )

    return default_emulator_config


def get_basic_assim_config(EnKF_members,
                           precip_p1,
                           precip_p2,
                           temperature_p2,
                           qobs_p2,
                           default_emulator_config):

    basic_assim_config = dict(
        EnsembleMode=rc.EnsembleMode(n=EnKF_members),
        EvaluationMetrics=("NASH_SUTCLIFFE",),
        # The folder where the ensemble runs will be generated. By default, the runs are called
ens_1... ens_N.
        OutputDirectoryFormat="./ens_*",
        # We need to tell Raven which inputs to perturb. the perturbation is applied following a
distribution
        # that should realistically represent the uncertainty of the observations of these variables.
        ForcingPerturbation=[
            rc.ForcingPerturbation(
                forcing="PRECIP",
                dist="DIST_UNIFORM",
                p1=precip_p1,
                p2=precip_p2,
                adj="MULTIPLICATIVE",
            ),
            rc.ForcingPerturbation(
                forcing="TEMP_MAX",
                dist="DIST_NORMAL",
                p1=0.0,
                p2=temperature_p2,
                adj="ADDITIVE",           ),
```

```
            rc.ForcingPerturbation(
                forcing="TEMP_MIN",
                dist="DIST_NORMAL",
                p1=0.0,
                p2=temperature_p2,
                adj="ADDITIVE",
            ),
        ],
        # Define the HRU Groups the assimilation will be applied on. Here we apply to all HRUs
(single HRU)
        DefineHRUGroups=["All"],
        HRUGroup=[{"name": "All", "groups": ["1"]}],
        # Define which variables we want to assimilate.
        # Here we only adjust the water content of the 2 first layers of soil (SOIL[0] and SOIL[1])
        AssimilatedState=[
            rc.AssimilatedState(state="SOIL[0]", group="All"),
            rc.AssimilatedState(state="SOIL[1]", group="All"),
        ],
        # Define which subbasin id the streamflow is associated with
        AssimilateStreamflow=[rc.AssimilateStreamflow(sb_id=1)],
        # Define the error model for the observed streamflow.
        ObservationErrorModel=[
            rc.ObservationErrorModel(
                state="STREAMFLOW",
                dist="DIST_NORMAL",
                p1=1,
                p2=qobs_p2,
                adj="MULTIPLICATIVE",
            )
        ],
        # Set to true for more details (verbosity)
        DebugMode=False,
        NoisyMode=False,
        **default_emulator_config,
    )

    return copy.deepcopy(basic_assim_config)

def perform_spinup_simulation(hru,
                              qobs_data_path,
                              optimized_parameters_gr4j,
                              EnKF_members,
                              precip_p1,
                              precip_p2,
                              temperature_p2,
```

```
                    qobs_p2,
                    gauge_ERA5,
                    spinup_start_date,
                    spinup_end_date,
                    path_assi):

    default_emulator_config = get_default_GR4J_config(hru, qobs_data_path,
optimized_parameters_gr4j)
    basic_assim_config = get_basic_assim_config(EnKF_members,
                                precip_p1,
                                precip_p2,
                                temperature_p2,
                                qobs_p2,
                                default_emulator_config,
                                )
    # Configure the spinup run now
    conf_spinup = GR4JCN(
        Gauge=gauge_ERA5,
        StartDate=spinup_start_date,
        EndDate=spinup_end_date,
        EnKFMode=o.EnKFMode.SPINUP,
        RunName="spinup",
        **basic_assim_config,
    )

    # Now that the configuration is completed, we can actually launch Raven to do the
assimilation
    _ = Emulator(config=conf_spinup, workdir=path_assi,
overwrite=True).run(overwrite=True)
    paths_spinup = list(path_assi.glob("ens_*"))
    ens_spinup = EnsembleReader(run_name=conf_spinup.run_name, paths=paths_spinup)

    return ens_spinup, default_emulator_config, copy.deepcopy(conf_spinup)

def plot_spinup_results(ens_spinup):
    # We can now plot the results
    ens_spinup.hydrograph.q_sim[:, :, 0].plot.line("b", x="time", add_legend=False, lw=0.5)
    ens_spinup.hydrograph.q_sim[1, :, 0].plot.line("b", x="time", label="Forecasts", lw=0.5)
    ens_spinup.hydrograph.q_obs[1, :, 0].plot.line("r", x="time", label="Observation", lw=1.0)
    plt.legend(loc="upper left")
    plt.ylabel("Streamflow (m³/s)")
    plt.title("Spinup period")
    plt.show(block=False)
```

```
def perform_postspinup_assim(spinup_end_date,
                             conf_spinup,
                             path_assi,
                             ):
    # Set the start date equal to the assimilated date of the prior run, as we want to start from the
assimilated
    # states. The end date is set 3 days later, after which assimilation will be automatically
performed.
    postspinup_start_date = spinup_end_date
    postspinup_end_date = spinup_end_date + dt.timedelta(days=3)

    # Closed-Loop assimilation. From the previous configuration, we can make a copy and only
change the required
    # parameters, such as the run name, start and end dates, and the type of EnKF.
    conf_loop = conf_spinup.duplicate(
        EnKFMode=o.EnKFMode.CLOSED_LOOP,
        RunName="loop",
        SolutionRunName="spinup",
        UniformInitialConditions=None,
        StartDate=postspinup_start_date,
        EndDate=postspinup_end_date,
    )

    _ = Emulator(config=conf_loop, workdir=path_assi, overwrite=True).run(overwrite=True)
    paths_loop = list(path_assi.glob("ens_*"))
    ens_loop = EnsembleReader(run_name=conf_loop.run_name, paths=paths_loop)
    total_hydrograph = ens_loop.hydrograph.isel(time=1)

    # Here is where the assimilation loop is performed. We will apply the assimilation 30
successive times, advancing
    # in time by 3 days each iteration.
    end_date = postspinup_end_date
    for i in range(0, 798, 3):
        print(i)
        # Set the new start_date and end_dates
        start_date = end_date
        end_date = end_date + dt.timedelta(days=3)

        # Again, copy the configuration object and change some elements
        conf_loop = conf_loop.duplicate(
            RunName="loop",
            SolutionRunName="loop",
            StartDate=start_date,
            EndDate=end_date,
        )
```

```
        # Perform the actual simulation and assimilation
        _                    =                    Emulator(config=conf_loop,                    workdir=path_assi,
overwrite=True).run(overwrite=True)

        # Extract the results for this day hydrograph and store it into our "total_hydrograph" which
keeps track of the flows for each day

        ens_loop = EnsembleReader(run_name=conf_loop.run_name, paths=paths_loop)  # lire
les résultats
        total_hydrograph = xr.concat([total_hydrograph, ens_loop.hydrograph.isel(time=slice(1,
100))], dim="time")

    return total_hydrograph, conf_loop.duplicate()

def plot_postspinup_assim(total_hydrograph):

    _, _ = plt.subplots()
    # Once the loop is complete, plot the results:
    total_hydrograph = total_hydrograph.isel(nbasins=0)
    total_hydrograph['q_sim'] = total_hydrograph["q_sim"].transpose("time", "member")
    total_hydrograph['q_obs'] = total_hydrograph["q_obs"].transpose("time", "member")
    plt.plot(total_hydrograph.time, total_hydrograph.q_sim[:, :], label="_", lw=0.5, color='b')
    plt.plot(total_hydrograph.time,  total_hydrograph.q_sim[:,  1],  label="Forecasts",  lw=0.5,
color="b")
    plt.plot(total_hydrograph.time, total_hydrograph.q_obs[:, 1], label="Observations", lw=0.5,
color="r")
    plt.legend(loc="upper left")
    plt.ylabel("Streamflow (m³/s)")
    plt.title("All closed-loop periods")
    plt.show(block=False)

def make_forecast_dataset(ds_tmp, t):
    step_data_tp = ds_tmp.tp.values
    step_data_mn2t6 = ds_tmp.mn2t6.values
    step_data_mx2t6 = ds_tmp.mx2t6.values
    dates = pd.date_range(t, t + dt.timedelta(days=13))
    ds3 = xr.Dataset(
        data_vars=dict(
            tp=(["time"], step_data_tp, {"units": "mm"}),
            mn2t6=(["time"], step_data_mn2t6, {"units": "degC"}),
            mx2t6=(["time"], step_data_mx2t6, {"units": "degC"}),

        ),
        coords=dict(
```

```
            time=dates,
        ),
    )
    ds3.to_netcdf("tmp_gr4j.nc")

def make_gauge_for_fcst(data_type, data_kwds):
    gauge_fcst = [
        rc.Gauge.from_nc(
            "tmp_gr4j.nc",
            data_type=data_type,
            alt_names={
                "PRECIP": "tp",
                "TEMP_MIN": "mn2t6",
                "TEMP_MAX": "mx2t6",
            },
            data_kwds=data_kwds,
        ),
    ]
    return gauge_fcst

def prepare_forecast_dataset(forecast_data_path):

    fcst_ds = xr.open_dataset(forecast_data_path)
    fcst_ds = fcst_ds.drop_vars(["sd", "u10", "v10", "sp", "crs", 'd2m'])
    fcst_ds['tp'].values = fcst_ds['tp'].values * 1000
    fcst_ds['tp'].values[fcst_ds['tp'].values < 0] = 0
    fcst_ds['mn2t6'].values = fcst_ds['mn2t6'].values - 273.15
    fcst_ds['mx2t6'].values = fcst_ds['mx2t6'].values - 273.15

    return fcst_ds
def perform_forecast_with_assimilation(forecast_data_path,
                                       forecast_start_date,
                                       forecast_end_date,
                                       data_type,
                                       data_kwds,
                                       path_assi,
                                       gauge_ERA5,
                                       conf_loop,
                                       forecast_da_path):

    # Read and correct raw data
    fcst_ds = prepare_forecast_dataset(forecast_data_path)

    # Path where to read files.
    paths_loop = list(path_assi.glob("ens_*"))
```

```
    # Prepare timesteps
    timesteps = pd.date_range(forecast_start_date, forecast_end_date)
    nb_timesteps = len(timesteps)

    # Preallocate matrices to store results
    mega_matrice = np.empty((25, 14, 50, nb_timesteps))
    mega_matrice[:, :, :, :] = np.nan
    qobs_matrice = np.empty((14, nb_timesteps))
    qobs_matrice[:, :] = np.nan

    # Pour chaque pas de temps
    for idx_t in range(0, nb_timesteps, 3):

        t = timesteps[idx_t]
        print(t)

        # Ajouter boucle pour chacun des membres ECMWF (50) et rouler ceci pour chacun
        for i in range(1, 51):

            ds_tmp = fcst_ds.sel(number=i).sel(time=t).drop_vars({'number', 'time'})
            make_forecast_dataset(ds_tmp, t)  # Builds the dataset and writes the netcdf to file.

            # Prepare gauge data for forecast
            gauge_fcst = make_gauge_for_fcst(data_type, data_kwds)

            conf_forecast = conf_loop.duplicate(
                EnKFMode=o.EnKFMode.FORECAST,
                RunName="forecast",
                SolutionRunName="loop",  # Utiliser les états du dernier run d'assimilation
                UniformInitialConditions=None,
                StartDate=t,  # Start date égale à la fin de la dernière assimilation
                EndDate=t + dt.timedelta(days=14),
                gauge=gauge_fcst,
            )

            _ = Emulator(config=conf_forecast, workdir=path_assi,
overwrite=True).run(overwrite=True)
            ens_loop_fcst = EnsembleReader(run_name='forecast', paths=paths_loop)

            donnees_forecast_25 = ens_loop_fcst.hydrograph.q_sim.isel(nbasins=0)[:, 1:]
            mega_matrice[:, :, i - 1, idx_t] = donnees_forecast_25
        # Add the Qobs for the same period after the 25 members have run.
        qobs_timestep = ens_loop_fcst.hydrograph.q_obs.isel(nbasins=0)[0, 1:]
        qobs_matrice[:, idx_t] = qobs_timestep
```

```
        # Faire l'assimilation pour une période de plus.
        conf_loop = conf_loop.duplicate(
            EnKFMode=o.EnKFMode.CLOSED_LOOP,
            RunName="loop",
            SolutionRunName="loop",
            UniformInitialConditions=None,
            StartDate=t,
            EndDate=t + dt.timedelta(days=3),
            gauge=gauge_ERA5,
        )

        _ = Emulator(config=conf_loop, workdir=path_assi, overwrite=True).run(overwrite=True)

    # When all is done, save to a large dataset and then to file.
    ds_matrice = xr.Dataset(
        data_vars=dict(
            q_sim=(["enkf_member", "step", "fcst_member", "time"], mega_matrice, {"units": "m3/s"}),
            q_obs=(["step", "time"], qobs_matrice, {"units": "m3/s"})
        ),
        coords=dict(
            time=timesteps,
            enkf_member=list(range(1, 26)),
            step=list(range(1, 15)),
            fcst_member=list(range(1, 51)),
        ),
    )

    ds_matrice.to_netcdf(forecast_da_path)

def perform_forecast_without_assimilation(forecast_data_path,
                                          forecast_start_date,
                                          forecast_end_date,
                                          data_type,
                                          data_kwds,
                                          gauge_ERA5,
                                          default_emulator_config,
                                          sim_start_date,
                                          workdir,
                                          forecast_no_da_path):
    # Read and correct raw data
    fcst_ds = prepare_forecast_dataset(forecast_data_path)
```

```
    # Prepare timesteps
    timesteps = pd.date_range(forecast_start_date, forecast_end_date)
    nb_timesteps = len(timesteps)

    mega_matrice = np.empty((14, 50, nb_timesteps))
    mega_matrice[:, :, :] = np.nan

    qobs_matrice = np.empty((14, nb_timesteps))
    qobs_matrice[:, :] = np.nan

    for idx_t in range(0, nb_timesteps, 3):
        t = timesteps[idx_t]
        print(t)

        # Start with a model simulation from day 1 to day t. Will take more computing time but
will also be more robust,
        # and return the solution file to use as a hotstart file
        hotstart = perform_simulation_from_start(gauge_ERA5,
                                                 default_emulator_config,
                                                 sim_start_date,
                                                 t,
                                                 workdir)
        shutil.copyfile(hotstart, "./sim_hotstart_gr4j.rvc")
        hotstart = Path("./sim_hotstart_gr4j.rvc")

        # Ajouter boucle pour chacun des membres ECMWF (50) et rouler ceci pour chacun
        for i in range(1, 51):

            ds_tmp = fcst_ds.sel(number=i).sel(time=t).drop_vars({'number', 'time'})
            make_forecast_dataset(ds_tmp, t)  # Builds the dataset and writes the netcdf to file.

            # Prepare gauge data for forecast
            gauge_fcst = make_gauge_for_fcst(data_type, data_kwds)

            config = GR4JCN(
                Gauge=gauge_fcst,
                UniformInitialConditions=None,
                StartDate=t,  # Start date égale à la fin de la dernière assimilation
                EndDate=t + dt.timedelta(days=14),
                **default_emulator_config
            )

            config = config.set_solution(Path(hotstart))
            config.run_name = "fcst_wo_da"
```

```
            # Must keep this in a function to kill the netcdf-handle of the OutputReader object after
every loop.
            hydro_forecast, qobs_timestep_wo_da = run_single_model_evaluation(config,
workdir)
            mega_matrice[:, i - 1, idx_t] = hydro_forecast

        qobs_matrice[:, idx_t] = qobs_timestep_wo_da

    ds_matrice_wo_da = xr.Dataset(
        data_vars=dict(
            q_sim=(["step", "fcst_member", "time"], mega_matrice, {"units": "m3/s"}),
            q_obs=(["step", "time"], qobs_matrice, {"units": "m3/s"})
        ),
        coords=dict(
            time=timesteps,
            step=list(range(1, 15)),
            fcst_member=list(range(1, 51)),
        ),
    )

    ds_matrice_wo_da.to_netcdf(forecast_no_da_path)


def perform_simulation_from_start(gauge_ERA5,
                                  default_emulator_config,
                                  sim_start_date,
                                  sim_end_date,
                                  workdir):

    m = GR4JCN(
        StartDate=sim_start_date,
        EndDate=sim_end_date,
        RunName='gr4j_sim',
        Gauge=gauge_ERA5,
        **default_emulator_config,
    )

    m.write_rv(workdir=workdir, overwrite=True)
    e = Emulator(config=m, workdir="./simulations/gr4j", overwrite=True)
    outputs = e.run(overwrite=True)

    return outputs.files["solution"]

def run_single_model_evaluation(config, workdir):
```

```
    fcst_wo_da = Emulator(config=config, workdir=workdir, overwrite=True).run(overwrite=True)

    hydro_forecast = fcst_wo_da.hydrograph.q_sim[1:, 0]  # Exclure les conditions initiales
    qobs_timestep_wo_da = fcst_wo_da.hydrograph.q_obs.values[1:, 0]

    return hydro_forecast, qobs_timestep_wo_da

def calculate_CRPS_with_assimilation(forecast_da_path, crps_da_path, EnKF_members):

    # Reshape dataset
    ds_da = xr.open_dataset(forecast_da_path)
    ds_da_reshape = ds_da.q_sim.transpose("enkf_member", "fcst_member", "step", "time").values.reshape(50 * EnKF_members, 14, -1)
    ds_da.close()

    q_sim_reshaped = xr.DataArray(
        ds_da_reshape,
        dims=['member', 'step', 'time'],
        coords={
            'member': range(50 * EnKF_members),
            'step': ds_da.coords['step'],
            'time': ds_da.coords['time'],
        }
    )

    crps_results = []

    for t in range(len(ds_da.coords["time"])):
        crps = xs.crps_ensemble(ds_da.q_obs.isel(time=slice(t, t + 1)), q_sim_reshaped.isel(time=slice(t, t + 1)),
                                dim='time')

        crps_results.append(crps)

    crps_combined = xr.concat(crps_results, dim="time")
    crps_combined = crps_combined.assign_coords(time=ds_da.coords["time"])
    crps_dataset = crps_combined.to_dataset(name="CRPS")
    crps_dataset.to_netcdf(crps_da_path)

def calculate_CRPS_without_assimilation(forecast_no_da_path, crps_no_da_path):
    ds_wo_da = xr.open_dataset(forecast_no_da_path)
    ds_wo_da_transpose = ds_wo_da.transpose('fcst_member', 'step', 'time')
    ds_wo_da.close()
```

```
    ds_wo_da_renamed = ds_wo_da_transpose.rename(
        {"fcst_member": "member"})

    crps_results_wo_da = []

    for t in range(len(ds_wo_da.coords["time"])):
        crps_wo_da = xs.crps_ensemble(ds_wo_da.q_obs.isel(time=slice(t, t + 1)),
                                      ds_wo_da_renamed.q_sim.isel(time=slice(t, t + 1)), dim='time')

        crps_results_wo_da.append(crps_wo_da)

    crps_combined_wo_da = xr.concat(crps_results_wo_da, dim="time")
    crps_combined_wo_da                                                                    =
crps_combined_wo_da.assign_coords(time=ds_wo_da.coords["time"])
    crps_dataset_wo_da = crps_combined_wo_da.to_dataset(name="CRPS")

    crps_dataset_wo_da.to_netcdf(crps_no_da_path)

import os
import datetime as dt
from pathlib import Path
import warnings
from numba.core.errors import NumbaDeprecationWarning
warnings.simplefilter("ignore", category=NumbaDeprecationWarning)

# Controls for modules activation
do_calibration = True
do_spinups = True
do_forecast_with_assimilation = True
do_forecast_without_assimilation = True
do_CRPS_calculation = True

# Define constants for catchment
area = 44505.998549328215
longitude = -72.70001976857071
latitude = 49.507906818124745
elevation = 406.856848375195
slope = 3.4303454465601417

# Define the filenames for ERA5, Qobs and forecast data
ERA5_data_path = "./data/ERA5_weather_data.nc"
qobs_data_path = './Debit_LSJ_mars2024_filtre_5_jours2.nc'
forecast_data_path = "./FC_LSJ_all_vars_units.nc"
forecast_da_path = "mega_matrice_fcst_da_gr4j.nc"  # Résultats prévisions avec DA
forecast_no_da_path = "mega_matrice_fcst_wo_da_gr4j.nc"  # Résultats précisions sans DA
```

```
crps_da_path = "crps_matrice_fcst_da_gr4j.nc"  # Résultats CRPS avec DA
crps_no_da_path = "crps_matrice_fcst_wo_da_gr4j.nc"  # Résultats CRPS sans DA

# Working directories for results:
workdir = Path("./simulation_outputs_gr4j")
os.makedirs(workdir, exist_ok=True)
path_assi = Path("./assimilation_outputs_gr4j")
os.makedirs(path_assi, exist_ok=True)

# Define calibration start and end dates. Will only happen if we let the calibration run
calibration_start_date = dt.date(1961, 12, 31)
calibration_end_date = dt.date(1990, 1, 1)

# Define spinup period dates (at least 1 year)
spinup_start_date = dt.datetime(2010, 12, 31)
spinup_end_date = dt.datetime(2013, 12, 31)

# Define forecast period dates.

forecast_start_date = spinup_end_date + dt.timedelta(days=801)
forecast_end_date = dt.datetime(2023, 5, 28)

# specify which timeframe we want to extract.
reference_start_day = dt.datetime(1960, 12, 31)
reference_stop_day = dt.datetime(2023, 5, 31)

# EnKF assimilation parameters.TODO THESE COULD/SHOULD BE ADJUSTED/TESTED.
EnKF_members = 25
temperature_p2 = 2.0
precip_p1 = 0.8
precip_p2 = 1.2
qobs_p2 = 0.07

# Get some tags and ravenpy-specific formats for data.
data_kwds, alt_names, data_type = get_ERA5_data_tags(elevation, latitude, longitude)
hru = define_land_hru(area, longitude, latitude, elevation, slope)
gauge_ERA5 = setup_ERA5_gauge(ERA5_data_path, data_type, alt_names, data_kwds)

# Calibration of the model, if required
if do_calibration:
    optimized_parameters_gr4j, NSE = calibrate_GR4J_model(calibration_start_date, calibration_end_date,
                                                          qobs_data_path, gauge_ERA5, hru)
else:
```

```
    optimized_parameters_gr4j = [0.31379529382416127, 9.98, 179.93596741220426,
6.436312461599519, 27.013488212477228,
                                 0.867064439354094]
# If we want to do the spinup and postspinup simulation. Note that it is necessary to do these
spinups if we also want
# to do forecasting. We can set to False in the flags if we want to only compute CRPS and to
avoid doing this for nothing
if do_spinups:
    # Spinup of the model from the long-term historical data to a point near where we want to
do our forecasts
    ens_spinup, default_emulator_config, conf_spinup = perform_spinup_simulation(hru,
                                                        qobs_data_path,
                                                        optimized_parameters_gr4j,
                                                        EnKF_members,
                                                        precip_p1,
                                                        precip_p2,
                                                        temperature_p2,
                                                        qobs_p2,
                                                        gauge_ERA5,
                                                        spinup_start_date,
                                                        spinup_end_date,
                                                        path_assi)
    plot_spinup_results(ens_spinup)
    print("DONE SPINUPS. STARTING POSTSPINUP ASSIMILATION.")

    # Start postspinup assimilation (continuous assimilation over a few years to converge states).
    total_hydrograph, conf_loop = perform_postspinup_assim(spinup_end_date, conf_spinup,
path_assi)
    plot_postspinup_assim(total_hydrograph)

print("DONE POSTSPINUP ASSIMILATION. STARTING FORECAST WITH
ASSIMILATION.")

# Do forecast with assimilation if required
if do_forecast_with_assimilation:
    if not do_spinups:
        raise Exception("Spinup must be done to do a forecast")

    perform_forecast_with_assimilation(forecast_data_path,
                                       forecast_start_date,
                                       forecast_end_date,
                                       data_type,
                                       data_kwds,
                                       path_assi,
                                       gauge_ERA5,
```

```
                        conf_loop,
                        forecast_da_path)


print("DONE FORECAST WITH ASSIMILATION. STARTING FORECAST WITHOUT
ASSIMILATION.")

# Prévision sans assimilation
if do_forecast_without_assimilation:
    if not do_spinups:
        raise Exception("Spinup must be done to do a forecast")

    perform_forecast_without_assimilation(forecast_data_path,
                                          forecast_start_date,
                                          forecast_end_date,
                                          data_type,
                                          data_kwds,
                                          gauge_ERA5,
                                          default_emulator_config,
                                          spinup_start_date,
                                          workdir,
                                          forecast_no_da_path)


# If we want to compute the CRPS values. Does both files (with and without DA) by default
if do_CRPS_calculation:
    # Calculate CRPS for the run with DA
    calculate_CRPS_with_assimilation(forecast_da_path, crps_da_path, EnKF_members)
    calculate_CRPS_without_assimilation(forecast_no_da_path, crps_no_da_path)


#Code pour faire les graphiques
import os
import netCDF4 as nc
import numpy as np
import xarray as xr
import seaborn as sns
import copy
import matplotlib.pyplot as plt

# Define path
path_data = "../writable-workspace"

# Load Observed Streamflow
obs_file = './Debit_LSJ_mars2024_filtre_5_jours2.nc'
```

```
obs_dataset = nc.Dataset(os.path.join(path_data, obs_file))
obs_streamflow = np.array(obs_dataset.variables["qobs"][:])
obs_time = obs_dataset.variables["time"][:]

# Convert time to NumPy datetime64
obs_time_units = obs_dataset.variables["time"].units
obs_dates_cftime = nc.num2date(obs_time, units=obs_time_units, only_use_cftime_datetimes=True)
obs_dates = np.array([np.datetime64(f"{t.year}-{t.month:02d}-{t.day:02d}T{t.hour:02d}:{t.minute:02d}:{t.second:02d}") for t in obs_dates_cftime])

# Define the simulated time range
sim_start_date = np.datetime64("2016-03-11T00:00:00")
sim_end_date = np.datetime64("2023-05-28T00:00:00")

# Filter observed data within the simulated range
mask = (obs_dates >= sim_start_date) & (obs_dates <= sim_end_date)
obs_streamflow_filtered = obs_streamflow[mask]
obs_dates_filtered = obs_dates[mask]

# Load Simulated Streamflows
sim_files = [
    "mega_matrice_fcst_da_blended.nc",
    "mega_matrice_fcst_da_gr4j.nc",
    "mega_matrice_fcst_da_mohyse.nc",
    "mega_matrice_fcst_wo_da_blended.nc",
    "mega_matrice_fcst_wo_da_gr4j.nc",
    "mega_matrice_fcst_wo_da_mohyse.nc",

]

sim_data = {}
for sim_file in sim_files:
    file_path = os.path.join(path_data, sim_file)
    dataset = nc.Dataset(file_path)

    q_sim = np.array(dataset.variables["q_sim"][:])
    sim_time = dataset.variables["time"][:]
    sim_time_units = dataset.variables["time"].units
    sim_dates_cftime = nc.num2date(sim_time, units=sim_time_units, only_use_cftime_datetimes=True)
    sim_dates = np.array([np.datetime64(f"{t.year}-{t.month:02d}-{t.day:02d}T{t.hour:02d}:{t.minute:02d}:{t.second:02d}") for t in sim_dates_cftime])

    sim_data[sim_file] = {"q_sim": q_sim, "time": sim_dates}
```

```
print(f"Observed data filtered: {obs_streamflow_filtered.shape[0]} time points")
print("Data loading complete.")
#transformation de la matrice
sim = sim_data["mega_matrice_fcst_wo_da_mohyse.nc"]["q_sim"]
if sim.shape[0] == 25 and sim.shape[2] == 50:
    sim = sim.transpose(2, 0, 1, 3)
    sim = sim.reshape( 25 * 50, 14, 2635)
print(sim.shape)
def create_sequences(data, seq_len):
    """Create overlapping sequences of a given length from a time series array."""
    if len(data) < seq_len:
        raise ValueError(f"Data length ({len(data)}) must be greater than sequence length
({seq_len}).")
    return np.array([data[i : i + seq_len] for i in range(len(data) - seq_len + 1)])

# sim: (members, lead_times, days)
# Calculer le déterministe : (lead_times, days)
sim_det = np.mean(sim, axis=0)   # (lead_times, days)
sim_det = sim_det.T              # (days, lead_times)  pour correspondre à obs

# Create overlapping sequences for observed streamflow
obs = create_sequences(obs_streamflow_filtered, 14)  # Shape: (n_seq, 14), ici n_seq = len -
13

# Pour la cohérence, il faut que sim et sim_det aient aussi (days, lead_times)
# Donc on transpose sim (qui est (members, lead_times, days)) pour (days, lead_times,
members)
sim_days_lead_members = sim.transpose(2, 1, 0)   # (days, lead_times, members)

# Garde la même taille pour tout le monde
seq_len = 14
n_seq = obs.shape[0]
sim_days_lead_members = sim_days_lead_members[:n_seq, :, :]  # (n_seq, 14, members)
sim_det = sim_det[:n_seq, :]                                 # (n_seq, 14)
obs = obs[:n_seq, :]                                         # (n_seq, 14)

# Garde tout sauf les 15 derniers éléments si besoin
n_final = n_seq - 15
sim_days_lead_members = sim_days_lead_members[:n_final, :, :]
sim_det = sim_det[:n_final, :]
obs = obs[:n_final, :]

# Print shapes to verify alignment
```

```
print(f"Simulated Streamflow Shape: {sim_days_lead_members.shape}")  # (n_final, 14, members)
print(f"Deterministic Simulated Shape: {sim_det.shape}")          # (n_final, 14)
print(f"Observed Streamflow Shape: {obs.shape}")                  # (n_final, 14)

#Code pour dispersion sans DA

def plot_forecast(sim_streamflow, det_streamflow, obs_streamflow, selected_day, forecast_days, ensemble_size=50):
    """
    Plots the 14-day forecast for a specific starting day with ensemble predictions, deterministic output, and observed streamflow.

    Args:
        sim_streamflow (numpy.ndarray): Probabilistic streamflow (shape: [members, forecast_days, total_days]).
        det_streamflow (numpy.ndarray): Deterministic streamflow (shape: [total_days, forecast_days]).
        obs_streamflow (numpy.ndarray): Observed streamflow (shape: [total_days, forecast_days]).
        selected_day (int): The day to start the forecast.
        forecast_days (int): Number of forecast days.
        ensemble_size (int): Number of ensemble members.
    """
    if selected_day >= sim_streamflow.shape[2]:
        raise ValueError(f"Selected day {selected_day} exceeds available data range (max: {sim_streamflow.shape[2]-1}).")

    # Extract forecasted data for the selected day
    # New sim shape: (members, lead_times, days)
    # Get all members, all lead_times for the selected day
    pred_segment = sim_streamflow[:, :, selected_day]
    det_segment = det_streamflow[selected_day, :]
    obs_segment = obs_streamflow[selected_day, :]

    # Set up the figure
    fig, ax = plt.subplots(figsize=(10, 6))

    # Plot each ensemble member as a light grey line
    for i in range(min(ensemble_size, pred_segment.shape[0])):
        ax.plot(range(1, forecast_days + 1), pred_segment[i, :], color='lightgrey', alpha=0.5, linewidth=1)

    # Plot deterministic forecast
```

```
    ax.plot(range(1, forecast_days + 1), det_segment, color='#ff6600', linestyle='-', linewidth=2, label='Deterministic Output')

    # Plot observed streamflow
    ax.plot(range(1, forecast_days + 1), obs_segment, color='#007acc', linestyle='--', linewidth=2, label='Observed Streamflow')

    # Scatter points for observed and deterministic values
    ax.scatter(range(1, forecast_days + 1), det_segment, color='#ff6600', s=30, edgecolors='black', label='_nolegend_')
    ax.scatter(range(1, forecast_days + 1), obs_segment, color='#007acc', s=30, edgecolors='black', label='_nolegend_')

    # Dynamically adjust the y-axis limits
    combined_min = min(pred_segment.min(), obs_segment.min(), det_segment.min())
    combined_max = max(pred_segment.max(), obs_segment.max(), det_segment.max())
    padding = 0.1 * (combined_max - combined_min)
    ax.set_ylim(combined_min - padding, combined_max + padding)

    # Set labels, title, and grid
    ax.set_xlim(1, forecast_days)
    ax.set_title(f'{forecast_days}-Day Forecast Starting at Day {selected_day}', fontsize=14)
    ax.set_xlabel('Forecast Day', fontsize=12)
    ax.set_ylabel('Streamflow (m³/s)', fontsize=12)

    # Customize legend and grid
    ax.legend(fontsize=10)
    ax.grid(True, linestyle='--', linewidth=0.7, color='gray', alpha=0.6)

    plt.show()

#Transpose avant le plot
sim_to_plot = np.transpose(sim_days_lead_members, (1, 2, 0))  # (50, 14, 2607)
sim_det = np.mean(sim_to_plot, axis=0).T  # (2607, 14)

#Vérifier les shapes
print("sim_to_plot:", sim_to_plot.shape)
print("sim_det:", sim_det.shape)
print("obs:", obs.shape)

plot_forecast(sim_to_plot, sim_det, obs,
              selected_day=150,
              forecast_days=14,
              ensemble_size=sim_to_plot.shape[0])
```

```
#Code pour dispersion avec DA

def plot_forecast(sim_streamflow, det_streamflow, obs_streamflow, selected_day, forecast_days, ensemble_size=50):
    """
    Plots the 14-day forecast for a specific starting day with ensemble predictions, deterministic output, and observed streamflow.

    Args:
        sim_streamflow (numpy.ndarray): Probabilistic streamflow (shape: [members, forecast_days, total_days]).
        det_streamflow (numpy.ndarray): Deterministic streamflow (shape: [total_days, forecast_days]).
        obs_streamflow (numpy.ndarray): Observed streamflow (shape: [total_days, forecast_days]).
        selected_day (int): The day to start the forecast.
        forecast_days (int): Number of forecast days.
        ensemble_size (int): Number of ensemble members.
    """
    if selected_day >= sim_streamflow.shape[2]:
        raise ValueError(f"Selected day {selected_day} exceeds available data range (max: {sim_streamflow.shape[2]-1}).")

    # Extract forecasted data for the selected day
    # New sim shape: (members, lead_times, days)
    # Get all members, all lead_times for the selected day
    pred_segment = sim_streamflow[:, :, selected_day]  # Shape: (members, forecast_days)
    det_segment = det_streamflow[selected_day, :]      # Shape: (forecast_days,)
    obs_segment = obs_streamflow[selected_day, :]      # Shape: (forecast_days,)

    # Set up the figure
    fig, ax = plt.subplots(figsize=(10, 6))

    # Plot each ensemble member as a light grey line
    for i in range(min(ensemble_size, pred_segment.shape[0])):
        ax.plot(range(1, forecast_days + 1), pred_segment[i, :], color='lightgrey', alpha=0.5, linewidth=1)

    # Plot deterministic forecast
    ax.plot(range(1, forecast_days + 1), det_segment, color='#ff6600', linestyle='-', linewidth=2, label='Deterministic Output')

    # Plot observed streamflow
    ax.plot(range(1, forecast_days + 1), obs_segment, color='#007acc', linestyle='--', linewidth=2, label='Observed Streamflow')
```

```
    # Scatter points for observed and deterministic values
    ax.scatter(range(1, forecast_days + 1), det_segment, color='#ff6600', s=30,
edgecolors='black', label='_nolegend_')
    ax.scatter(range(1, forecast_days + 1), obs_segment, color='#007acc', s=30,
edgecolors='black', label='_nolegend_')

    # Dynamically adjust the y-axis limits
    combined_min = min(pred_segment.min(), obs_segment.min(), det_segment.min())
    combined_max = max(pred_segment.max(), obs_segment.max(), det_segment.max())
    padding = 0.1 * (combined_max - combined_min)
    ax.set_ylim(combined_min - padding, combined_max + padding)

    # Set labels, title, and grid
    ax.set_xlim(1, forecast_days)
    ax.set_title(f'{forecast_days}-Day Forecast Starting at Day {selected_day}', fontsize=14)
    ax.set_xlabel('Forecast Day', fontsize=12)
    ax.set_ylabel('Streamflow (m³/s)', fontsize=12)

    # Customize legend and grid
    ax.legend(fontsize=10)
    ax.grid(True, linestyle='--', linewidth=0.7, color='gray', alpha=0.6)

    plt.show()

plot_forecast(sim, sim_det, obs,
              selected_day=150,
              forecast_days=14,
              ensemble_size=sim.shape[0])

#Code pour les hydrogrammes

def interp_nan(arr):
    """Interpole linéairement les NaN dans une série."""
    arr = np.asarray(arr)
    nans = np.isnan(arr)
    if np.any(nans):
        arr[nans] = np.interp(np.flatnonzero(nans), np.flatnonzero(~nans), arr[~nans])
    return arr

def compute_kge(sim, obs):
    """Calcule le Kling-Gupta Efficiency (KGE) entre 2 séries."""
    sim = np.asarray(sim).flatten()
    obs = np.asarray(obs).flatten()
    mask = (~np.isnan(sim)) & (~np.isnan(obs))
```

```
    if mask.sum() == 0:
        return np.nan
    sim = sim[mask]
    obs = obs[mask]
    r = np.corrcoef(obs, sim)[0,1]
    alpha = np.std(sim) / np.std(obs)
    beta = np.mean(sim) / np.mean(obs)
    kge = 1 - np.sqrt((r-1)**2 + (alpha-1)**2 + (beta-1)**2)
    return kge

def create_sequences(data, seq_len):
    """Découpe une série temporelle en séquences chevauchantes de longueur seq_len."""
    data = np.asarray(data)
    if len(data) < seq_len:
        raise ValueError(f"Data length ({len(data)}) must be greater than sequence length
({seq_len}).")
    return np.array([data[i : i + seq_len] for i in range(len(data) - seq_len + 1)])

def plot_forecast_comparison(prob_sim, det_sim, det_sim_no_da, obs_data,
                             start_idx, num_days, lead_day, num_members=50):
    x = np.arange(num_days)
    end = start_idx + num_days

    prob_series    = prob_sim[start_idx:end, lead_day, :]      # (num_days, members)
    det_series     = det_sim[start_idx:end, lead_day]          # (num_days,)
    det_no_da_ser  = det_sim_no_da[start_idx:end, lead_day]    # (num_days,)
    obs_series     = obs_data[start_idx:end, lead_day]         # (num_days,)

    # Interpoler les éventuels NaN pour l'affichage/KGE
    det_series    = interp_nan(det_series)
    det_no_da_ser = interp_nan(det_no_da_ser)

    # Calculer KGE
    kge_da    = compute_kge(det_series,    obs_series)
    kge_no_da = compute_kge(det_no_da_ser, obs_series)

    plt.figure(figsize=(12, 6))
    # Membres
    for i in range(num_members):
        plt.plot(x, prob_series[:, i], color='lightgrey', alpha=0.5, linewidth=1)
    # Observé
    plt.plot(x, obs_series,      'k-', label='Observed', linewidth=2)
    # No DA
    plt.plot(x, det_no_da_ser,   'b-', label=f'No DA (KGE={kge_no_da:.2f})', linewidth=1.5)
    # With DA
```

```
    plt.plot(x, det_series,      'r-', label=f'With DA (KGE={kge_da:.2f})', linewidth=1.5)

    plt.title(f'Streamflow Comparison – Forecast Day {lead_day+1}')
    plt.xlabel('Days')
    plt.ylabel('Streamflow (m³/s)')
    plt.ylim(0, 8000)
    plt.legend(loc='upper right')
    plt.grid(linestyle='--', alpha=0.6)
    plt.tight_layout()
    plt.show()

#  WithDA
sim_da = sim_data["mega_matrice_fcst_da_mohyse.nc"]["q_sim"]
print("raw sim_da shape:", sim_da.shape)
if sim_da.shape[0] == 25 and sim_da.shape[2] == 50:
    sim_da = sim_da.transpose(2, 0, 1, 3)              # (50, 25, 14, 2635)
    sim_da = sim_da.reshape(25 * 50, 14, 2635)         # (1250, 14, 2635)
print("reshaped sim_da shape:", sim_da.shape)
# NoDA
sim_no_da = sim_data["mega_matrice_fcst_wo_da_mohyse.nc"]["q_sim"]
print("raw sim_no_da shape:", sim_no_da.shape)
if sim_no_da.shape[0] == 25 and sim_no_da.shape[2] == 50:
    sim_no_da = sim_no_da.transpose(2, 0, 1, 3)
    sim_no_da = sim_no_da.reshape(25 * 50, 14, 2635)
print("reshaped sim_no_da shape:", sim_no_da.shape)

#  Mettre sous (jours, lead, membres)
sim_da    = sim_da.transpose(2, 1, 0)      # (2635, 14, 1250)
sim_no_da = sim_no_da.transpose(2, 1, 0)   # (2635, 14, 1250)

# Déterministes
sim_da_det    = np.mean(sim_da, axis=2)    # (2635, 14)
sim_no_da_det = np.mean(sim_no_da, axis=2) # (2635, 14)

# Séquences observations
obs = create_sequences(obs_streamflow_filtered, 14)   # (n, 14)
n = obs.shape[0]

sim_da         = sim_da[:n, :, :]        # (n, 14, 1250)
sim_no_da      = sim_no_da[:n, :, :]     # (n, 14, 1250)
sim_da_det     = sim_da_det[:n, :]       # (n, 14)
sim_no_da_det  = sim_no_da_det[:n, :]    # (n, 14)
obs            = obs[:n, :]              # (n, 14)

for day in range(14):
```

```
    plot_forecast_comparison(
        prob_sim=sim_da,                  # (n, 14, 1250)
        det_sim=sim_da_det,               # (n, 14)
        det_sim_no_da=sim_no_da_det,      # (n, 14)
        obs_data=obs,                     # (n, 14)
        start_idx=1,
        num_days=obs.shape[0]-1,
        lead_day=day,
        num_members=sim_da.shape[2]
    )

#Code pour les CRPS sans saison pour comparer les 3 modèles ensemble

model_files = {
   "Blended avec DA": "crps_matrice_fcst_da_blended.nc",
    "GR4J-CN avec DA": "crps_matrice_fcst_da_gr4j.nc",
    "Mohyse avec DA":  "crps_matrice_fcst_da_mohyse.nc",
    #"Blended sans DA": "crps_matrice_fcst_wo_da_blended.nc",
    #"GR4J-CN sans DA": "crps_matrice_fcst_wo_da_gr4j.nc",
    #"Mohyse sans DA":  "crps_matrice_fcst_wo_da_mohyse.nc"
}

# Chargement des CRPS pour chaque modèle
sim_data_crps = {}
for model_name, file_path in model_files.items():
    ds = xr.open_dataset(file_path)
    sim_data_crps[model_name] = ds["CRPS"].values  # shape [time, step]

def plot_precomputed_crps_global(sim_data_crps, forecast_days=14):
    """
    Trace un unique boxplot global du CRPS pour tous les modèles.

    sim_data_crps : dict { model_name: np.ndarray de forme (time, step) }
    """
    colors = sns.color_palette("Set2", len(sim_data_crps))
    # Variante claire
    colors_light = [sns.light_palette(c, n_colors=3)[1] for c in colors]

    fig, ax = plt.subplots(figsize=(14, 10))

    model_crps_values = []
    model_names = []
    lead_times = []

    # Parcours de tous les modèles et accumulation des valeurs CRPS sur tous les indices
```

```
    for (color, (model_name, crps_values)) in zip(colors, sim_data_crps.items()):
        max_time = crps_values.shape[0]
        indices = np.arange(max_time)
        for day in range(forecast_days):

            valid_indices = indices
            if len(valid_indices) == 0:
                continue
            day_crps = crps_values[valid_indices, day]
            model_crps_values.extend(day_crps)
            lead_times.extend([str(day + 1)] * len(day_crps))
            model_names.extend([model_name] * len(day_crps))

    sns.boxplot(
        x=lead_times,
        y=model_crps_values,
        hue=model_names,
        palette=colors_light,#Changer selon la teinte souhaitée
        ax=ax,
        width=0.8,
        linewidth=0.5,
        showfliers=True,
        whis=1.5
    )

    ax.set_title("CRPS", fontsize=14)
    ax.set_xlabel("Jours", fontsize=14)
    ax.set_ylabel("CRPS (m³/s)", fontsize=14)
    ax.grid(True, linestyle="--", alpha=0.7)
    ax.set_ylim(0, 2500)

    # Légende
    handles, labels = ax.get_legend_handles_labels()
    ax.legend(handles, labels, title="Modèle", fontsize=12)

    plt.tight_layout()
    plt.savefig("CRPS_DA_.png", bbox_inches='tight')
    plt.show()

plot_precomputed_crps_global(sim_data_crps, forecast_days=14)

#Comparaison CRPS même modèle
model_files = {
    #"Blended avec DA": "crps_matrice_fcst_da_blended.nc",
    #"Blended sans DA": "crps_matrice_fcst_wo_da_blended.nc",
```

```
    #"GR4J-CN avec DA": "crps_matrice_fcst_da_gr4j.nc",
    #"GR4J-CN sans DA": "crps_matrice_fcst_wo_da_gr4j.nc",
    "Mohyse avec DA":  "crps_matrice_fcst_da_mohyse.nc",
    "Mohyse sans DA":  "crps_matrice_fcst_wo_da_mohyse.nc"
}

sim_data_crps = {}
for name, path in model_files.items():
    ds = xr.open_dataset(path)
    sim_data_crps[name] = ds["CRPS"].values  # shape [time, lead_time]


base  = sns.color_palette("Set2", 3)[2]  #changer selon la couleur souhaitée

light = sns.light_palette(base, n_colors=3)[1]  # version claire

palette = {
    "Mohyse avec DA":   light, #changer selon le modèle
    "Mohyse sans DA":   base
}

values, lead_times, names = [], [], []
forecast_days = 14

for model_name, arr in sim_data_crps.items():
    ntime, nstep = arr.shape
    for day in range(min(forecast_days, nstep)):
        vals = arr[:, day]
        values.extend(vals)
        lead_times.extend([str(day+1)] * len(vals))
        names.extend([model_name] * len(vals))

fig, ax = plt.subplots(figsize=(14, 10))
sns.boxplot(
    x=lead_times,
    y=values,
    hue=names,
    palette=palette,
    ax=ax,
    width=0.8,
    linewidth=0.5,
    showfliers=True,
    whis=1.5
)
```

```
ax.set_title("CRPS", fontsize=16)
ax.set_xlabel("Jours", fontsize=14)
ax.set_ylabel("CRPS (m³/s)", fontsize=14)
ax.grid(True, linestyle="--", alpha=0.4)
ax.set_ylim(0, 2500)
handles, labels = ax.get_legend_handles_labels()
ax.legend(handles, labels, title="Modèle", fontsize=12)

plt.tight_layout()
plt.savefig("CRPS_Mohyse.png", bbox_inches='tight')
plt.show()

#Code pour les CRPS séparé par saison
#avec les CRPS déjà calculés

model_files = {
   "Blended avec DA": "crps_matrice_fcst_da_blended.nc",
   "GR4J-CN avec DA": "crps_matrice_fcst_da_gr4j.nc",
   "Mohyse avec DA":  "crps_matrice_fcst_da_mohyse.nc",
   #"Blended sans DA": "crps_matrice_fcst_wo_da_blended.nc",
   #"GR4J-CN sans DA": "crps_matrice_fcst_wo_da_gr4j.nc",
   #"Mohyse sans DA":  "crps_matrice_fcst_wo_da_mohyse.nc"
}

# Define seasonal indices with specific Y-limits
seasons = {
   "DJFM": {"indices": list(range(0, 21)) + list(range(265, 386)) + list(range(630, 751)) +
list(range(995, 1116)) +
          list(range(1360, 1482)) + list(range(1726, 1847)) + list(range(2091, 2212)) +
list(range(2456, 2577)), "ylim": 1000}, #HIVER
   "AM": {"indices": list(range(21, 82)) + list(range(386, 447)) + list(range(751, 812)) +
list(range(1116, 1177)) +
             list(range(1482, 1543)) + list(range(1847, 1908)) + list(range(2212, 2273)) +
list(range(2577, 2635)),"ylim": 4000}, #PRINTEMPS
   "JJA": {"indices": list(range(82, 174)) + list(range(447, 539)) + list(range(812, 904)) +
list(range(1177, 1269)) +
         list(range(1543, 1635)) + list(range(1908, 2000)) + list(range(2273, 2365)), "ylim":
2000}, #ÉTÉ
   "SON": {"indices":list(range(174, 265)) + list(range(539, 630)) + list(range(904, 995)) +
list(range(1269, 1360)) +
           list(range(1635, 1726)) + list(range(2000, 2091)) + list(range(2365, 2456)),"ylim":
1200} #Automne
}

# Load each model's CRPS into a dictionary
```

```
sim_data_crps = {}
for model_name, file_path in model_files.items():
    ds = xr.open_dataset(file_path)
    sim_data_crps[model_name] = ds["CRPS"].values  # shape [time, step]

def plot_precomputed_crps_multiple(sim_data_crps, forecast_days=14):
    """
    Plot CRPS for multiple models.
    sim_data_crps: dict { model_name: np.ndarray of shape (time, step) }
    """
    colors = sns.color_palette("Set2", len(sim_data_crps))
    # Variante claire
    colors_light = [sns.light_palette(c, n_colors=3)[1] for c in colors]
    fig, axes = plt.subplots(2, 2, figsize=(14, 10), sharex=True)

    for (saison, data), ax in zip(seasons.items(), axes.flatten()):
        saison_indices = np.array(data["indices"])

        # We'll gather all models' CRPS for this season
        model_crps_values = []
        model_names = []
        lead_times = []

        for (color, (model_name, crps_values)) in zip(colors, sim_data_crps.items()):
            max_time = crps_values.shape[0]
            saison_indices_clipped = saison_indices[saison_indices < max_time]

            for day in range(forecast_days):
                valid_indices = saison_indices_clipped
                if len(valid_indices) == 0:
                    continue

                day_crps = crps_values[valid_indices, day]
                model_crps_values.extend(day_crps)
                lead_times.extend([str(day + 1)] * len(day_crps))
                model_names.extend([model_name] * len(day_crps))

        sns.boxplot(
            x=lead_times,
            y=model_crps_values,
            hue=model_names,
            palette=colors_light,
            ax=ax,
            width=0.8,
            linewidth=0.5,
```

```
            showfliers=True,
            whis=1.5
        )

        ax.set_title(saison, fontsize=12, fontweight="bold")
        ax.set_xlabel("Jours")
        ax.set_ylabel("CRPS (m³/s)")
        ax.set_ylim(0, data["ylim"])
        ax.grid(True, linestyle="--", alpha=0.7)
        ax.legend_.remove()

    handles, labels = ax.get_legend_handles_labels()
    fig.legend(handles, labels, title="Modèle", loc="lower center", ncol=len(sim_data_crps),
               fontsize=10, bbox_to_anchor=(0.5, -0.05))
    plt.tight_layout(rect=[0, 0.05, 1, 1])

    for ax in axes.flat:
        ax.tick_params(labelbottom=True)

    plt.savefig("SaisonDA.png", bbox_inches='tight')
    plt.show()
plot_precomputed_crps_multiple(sim_data_crps, forecast_days=14)

#Pour les Talagrand sans saison

# Créer une copie pour éviter de modifier l'original
sim_data_transformed = copy.deepcopy(sim_data)

def process_simulation_data(sim):
    """
    Applique les transformations à la simulation pour obtenir la forme (members, lead_time,
time).
    """
    print(f"Original shape: {sim.shape}")

    # Reshape
    if sim.shape[0] == 25 and sim.shape[2] == 50:
        #transpose pour avoir les axes dans l'ordre (0=ensemble, 1=lead_time, 2=time,
3=members)
        sim = sim.transpose(2, 0, 1, 3)  # devient (50, 25, 14, 2635)
        # On reshape pour aplatir les ensembles en un seul axe: (members=25*50, lead_time=14,
time=2635)
        sim = sim.reshape(25 * 50, 14, 2635)

    print(f"Transformed shape: {sim.shape}")
```

```
    return sim

# Appliquer la transformation à tous les modèles
for model in sim_files:
    print(f"Processing {model}...")
    sim_transformed = process_simulation_data(sim_data_transformed[model]["q_sim"])

    sim_transformed = sim_transformed[:, :, :obs.shape[0]+15]
    sim_transformed = sim_transformed[:, :, :-15]

    # Stocker dans le dataset transformé
    sim_data_transformed[model]["q_sim"] = sim_transformed

    print(f"Final shape: {sim_transformed.shape}")
    print("-" * 50)

print("✅ All models transformed and stored in `sim_data_transformed`")

#Comparaison talagrand entre les 3 modèles
sim_files = [
    "mega_matrice_fcst_da_blended.nc",
    "mega_matrice_fcst_da_gr4j.nc",
    "mega_matrice_fcst_da_mohyse.nc",
    #"mega_matrice_fcst_wo_da_blended.nc",
    #"mega_matrice_fcst_wo_da_gr4j.nc",
    #"mega_matrice_fcst_wo_da_mohyse.nc",
]
sim_labels = [
    "Blended",
    "GR4J-CN",
    "Mohyse"
]

palette = sns.color_palette("Set2", n_colors=len(sim_files))
# Variante claire
colors_light = [sns.light_palette(c, n_colors=3)[1] for c in palette]


def compute_rank_histogram_all(obs_unscaled, sim_data_transformed, forecast_days=14,
num_bins=10):
    """
    Génère des diagrammes de Talagrand pour plusieurs lead times,
    en coloriant chaque modèle avec sa propre couleur.
    """
    selected_lead_times = [1, 3, 7, 14]
```

```
all_sim_models = [sim_data_transformed[f]["q_sim"] for f in sim_files]

fig, axes = plt.subplots(
    nrows=len(selected_lead_times),
    ncols=len(all_sim_models),
    figsize=(len(all_sim_models)*4, len(selected_lead_times)*3),
    sharex=True, sharey=True
)

axes = np.atleast_2d(axes)
all_indices = np.arange(obs_unscaled.shape[0])

for model_idx, prob_unscaled in enumerate(all_sim_models):
    color = colors_light[model_idx]
    short_label = sim_labels[model_idx]

    for lt_idx, lead_time in enumerate(selected_lead_times):
        ax = axes[lt_idx, model_idx]
        valid_indices = all_indices[all_indices < obs_unscaled.shape[0]]
        obs_data = obs_unscaled[valid_indices, lead_time - 1]

        # extraction des membres pour chaque date
        n_dates = len(valid_indices)
        n_members = prob_unscaled.shape[0]
        ensemble_data = np.zeros((n_dates, n_members))
        for i, t in enumerate(valid_indices):
            ensemble_data[i, :] = prob_unscaled[:, lead_time - 1, t]

        rank_data = np.sum(ensemble_data < obs_data[:, None], axis=1)
        ax.hist(
            rank_data,
            bins=num_bins,
            color=color,
            edgecolor="black",
            alpha=0.7
        )

        if lt_idx == 0:
            ax.set_title(short_label, fontsize=10, fontweight="bold", wrap=True)
        if model_idx == 0:
            ax.set_ylabel(f"Lead Time {lead_time} days\nOccurrences", fontsize=9)
        ax.grid(True, linestyle="--", alpha=0.6)

plt.tight_layout(rect=[0, 0, 1, 0.96])
for ax in axes.flat:
```

```
        ax.tick_params(labelbottom=True)
    plt.savefig("talagrand_all_da_colored.png", bbox_inches='tight')
    plt.show()

# Appel de la fonction
compute_rank_histogram_all(obs, sim_data_transformed, forecast_days=14)

#Pour obtenir les Talagrand par saison
sim_files = [
    "mega_matrice_fcst_da_blended.nc",
    "mega_matrice_fcst_da_gr4j.nc",
    "mega_matrice_fcst_da_mohyse.nc",
    #"mega_matrice_fcst_wo_da_blended.nc",
    #"mega_matrice_fcst_wo_da_gr4j.nc",
    #"mega_matrice_fcst_wo_da_mohyse.nc",
]
sim_labels = [
    "Blended",
    "GR4J-CN",
    "Mohyse"
]

# Saisons
seasons = {
    "DJFM": {"indices": list(range(0, 21)) + list(range(265, 386)) + list(range(630, 751)) +
list(range(995, 1116)) +
            list(range(1360, 1482)) + list(range(1726, 1847)) + list(range(2091, 2212)) +
list(range(2456, 2577)), "ylim": 1000},
    "AM": {"indices": list(range(21, 82)) + list(range(386, 447)) + list(range(751, 812)) +
list(range(1116, 1177)) +
              list(range(1482, 1543)) + list(range(1847, 1908)) + list(range(2212, 2273)) +
list(range(2577, 2635)), "ylim": 4000},
    "JJA": {"indices": list(range(82, 174)) + list(range(447, 539)) + list(range(812, 904)) +
list(range(1177, 1269)) +
           list(range(1543, 1635)) + list(range(1908, 2000)) + list(range(2273, 2365)), "ylim":
2000},
    "SON": {"indices": list(range(174, 265)) + list(range(539, 630)) + list(range(904, 995)) +
list(range(1269, 1360)) +
             list(range(1635, 1726)) + list(range(2000, 2091)) + list(range(2365, 2456)), "ylim":
1600}
}

colors = sns.color_palette("Set2", n_colors=len(sim_files))
# Variante claire
colors_light = [sns.light_palette(c, n_colors=3)[1] for c in colors]
```

```
def process_simulation_data(sim):
    print(f"Original shape: {sim.shape}")
    # AVEC assimilation (25, 14, 50, N)
    if sim.ndim == 4 and sim.shape[0] == 25 and sim.shape[2] == 50:
        sim = sim.transpose(2, 0, 1, 3)         # (50, 25, 14, N)
        sim = sim.reshape(25 * 50, 14, sim.shape[3])    # (1250, 14, N)
    # SANS assimilation (50, 14, N)
    elif sim.ndim == 3 and sim.shape[0] == 50 and sim.shape[1] == 14:
        pass  # déjà bon
    # SANS assimilation (14, 50, N)
    elif sim.ndim == 3 and sim.shape[0] == 14 and sim.shape[1] == 50:
        sim = sim.transpose(1, 0, 2)  # devient (50, 14, N)
    else:
        raise ValueError(f"Format inattendu pour la matrice de simulation: {sim.shape}")
    print(f"Transformed shape: {sim.shape}")
    return sim


# Appliquer la transformation à tous les modèles
sim_data_transformed = copy.deepcopy(sim_data)
for model in sim_files:
    print(f"Processing {model}...")
    sim_transformed = process_simulation_data(sim_data_transformed[model]["q_sim"])
    sim_transformed = sim_transformed[:, :, :obs.shape[0]+15]
    sim_transformed = sim_transformed[:, :, :-15]
    sim_data_transformed[model]["q_sim"] = sim_transformed
    print(f"Final shape: {sim_transformed.shape}")
    print("-" * 50)
print("✅ All models transformed and stored in `sim_data_transformed`")


def compute_rank_histogram_per_season(obs_unscaled, sim_data_transformed,
forecast_days=14, num_bins=10):
    selected_lead_times = [1, 3, 7, 14]
    all_sim_models = [sim_data_transformed[file]["q_sim"] for file in sim_files]

    for season_name, season_data in seasons.items():
        print(f"🔷 Processing Season: {season_name}")

        fig, axes = plt.subplots(len(selected_lead_times), len(all_sim_models),
                                 figsize=(len(all_sim_models) * 4, len(selected_lead_times) * 3),
                                 sharex=True, sharey=True)
        axes = np.atleast_2d(axes)
```

```
    saison_indices = np.array(season_data["indices"])
    saison_indices = saison_indices[saison_indices < obs_unscaled.shape[0]]

    for model_idx, prob_unscaled in enumerate(all_sim_models):
        short_label = sim_labels[model_idx]

        for lt_idx, lead_time in enumerate(selected_lead_times):
            ax = axes[lt_idx, model_idx]
            valid_indices = saison_indices[saison_indices < prob_unscaled.shape[2]]

            obs_season = obs_unscaled[valid_indices, lead_time - 1]

            n_dates = len(valid_indices)
            n_members = prob_unscaled.shape[0]
            ensemble_data = np.zeros((n_dates, n_members))
            for i, t in enumerate(valid_indices):
                ensemble_data[i, :] = prob_unscaled[:, lead_time - 1, t]

            rank_season = np.sum(ensemble_data < obs_season[:, None], axis=1)

            #color = colors_light[model_idx]
            ax.hist(rank_season,
                    bins=num_bins,
                    color=colors_light[model_idx],
                    edgecolor="black",
                    alpha=0.7)
            if lt_idx == 0:
                ax.set_title(short_label, fontsize=14, fontweight="bold", wrap=True)
            if model_idx == 0:
                ax.set_ylabel(f"Lead Time {lead_time} days\nOccurrences", fontsize=14)
            ax.grid(True, linestyle="--", alpha=0.6)

    plt.suptitle(f"Diagramme de Talagrand - Saison: {season_name}", fontsize=18,
fontweight="bold")
    plt.tight_layout(rect=[0, 0, 1, 0.96])
    for ax in axes.flat:
        ax.tick_params(labelbottom=True, labelsize=10)
    output_filename = f"talagrand_DA_{season_name}.png"
    #plt.savefig(output_filename, bbox_inches='tight')
    plt.show()

compute_rank_histogram_per_season(obs, sim_data_transformed, forecast_days=14)
```

ANNEXE II

DISPERSION DES ENSEMBLES POUR MOHYSE ET GR4J-CN

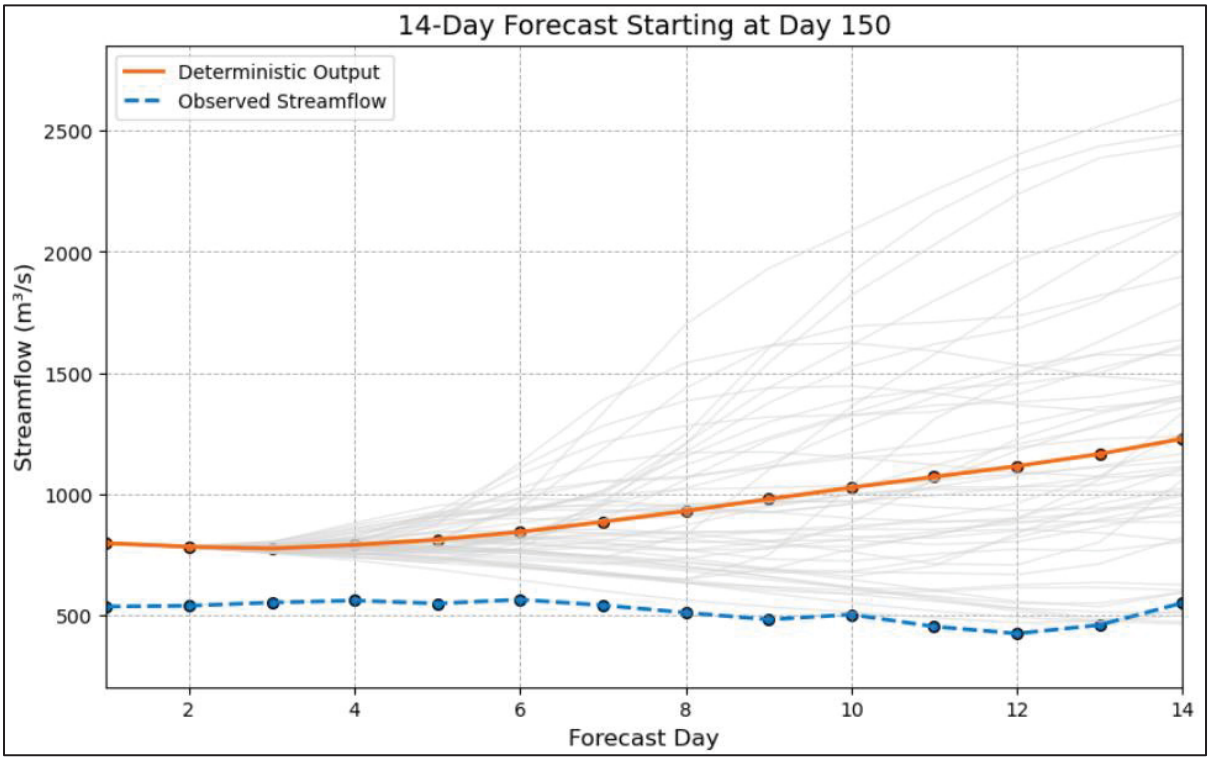

Figure-A II-1 Dispersion de l'ensemble sans assimilation de données sur un horizon de 14 jours commençant le jour 150 pour le modèle MOHYSE

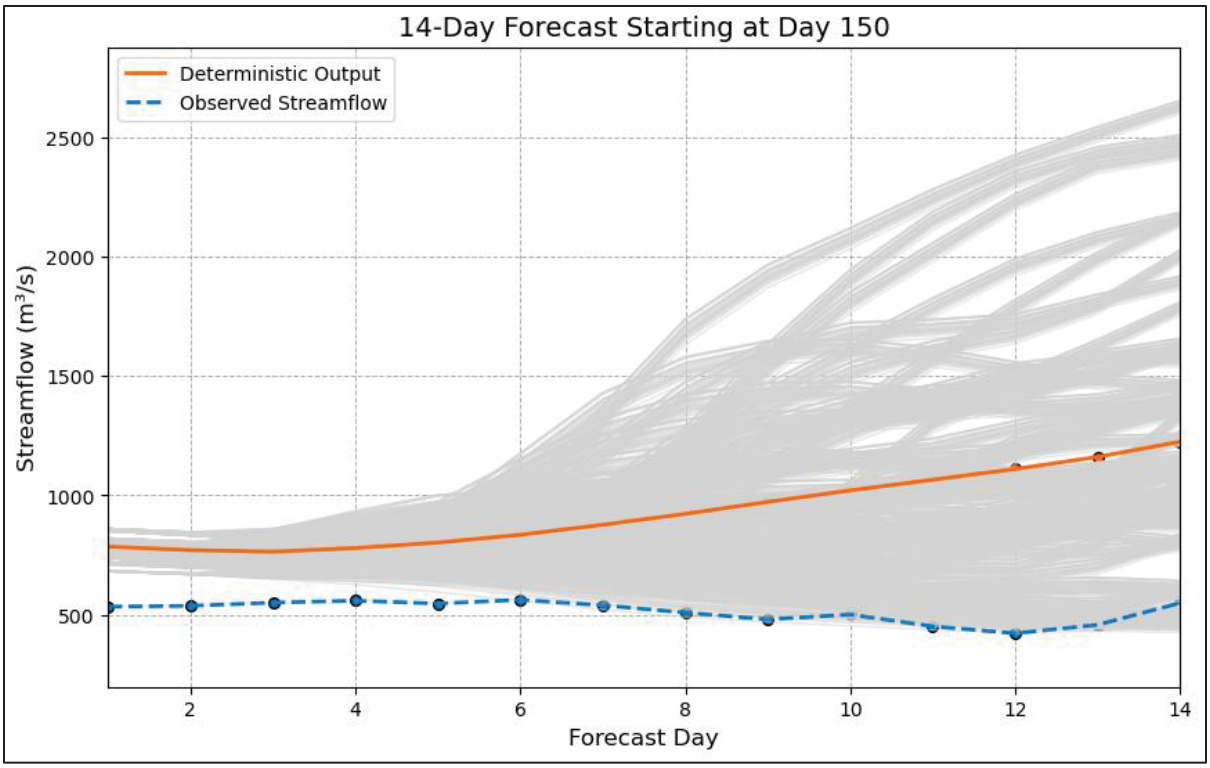

Figure-A II-2 Dispersion de l'ensemble avec assimilation de données sur un horizon de 14 jours commençant le jour 150 pour le modèle MOHYSE

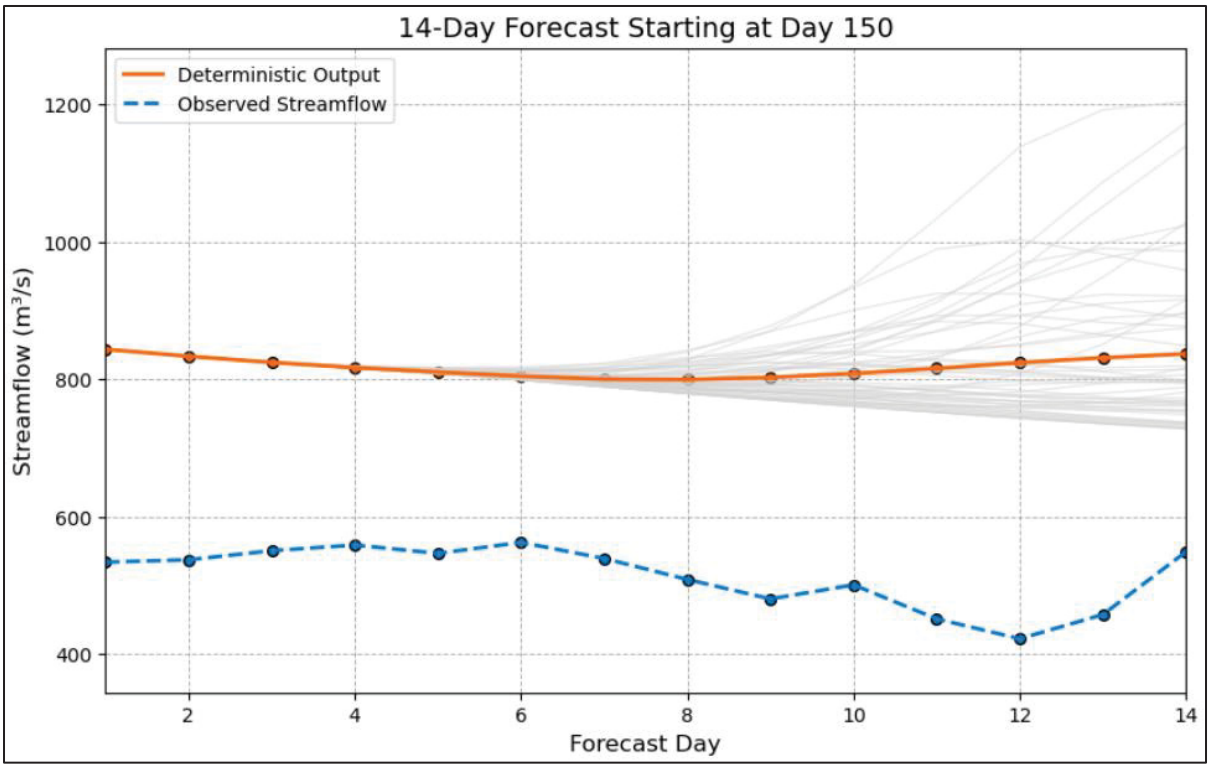

Figure-A II-3 Dispersion de l'ensemble sans assimilation de données sur un horizon de 14 jours commençant le jour 150 pour le modèle GR4J-CN

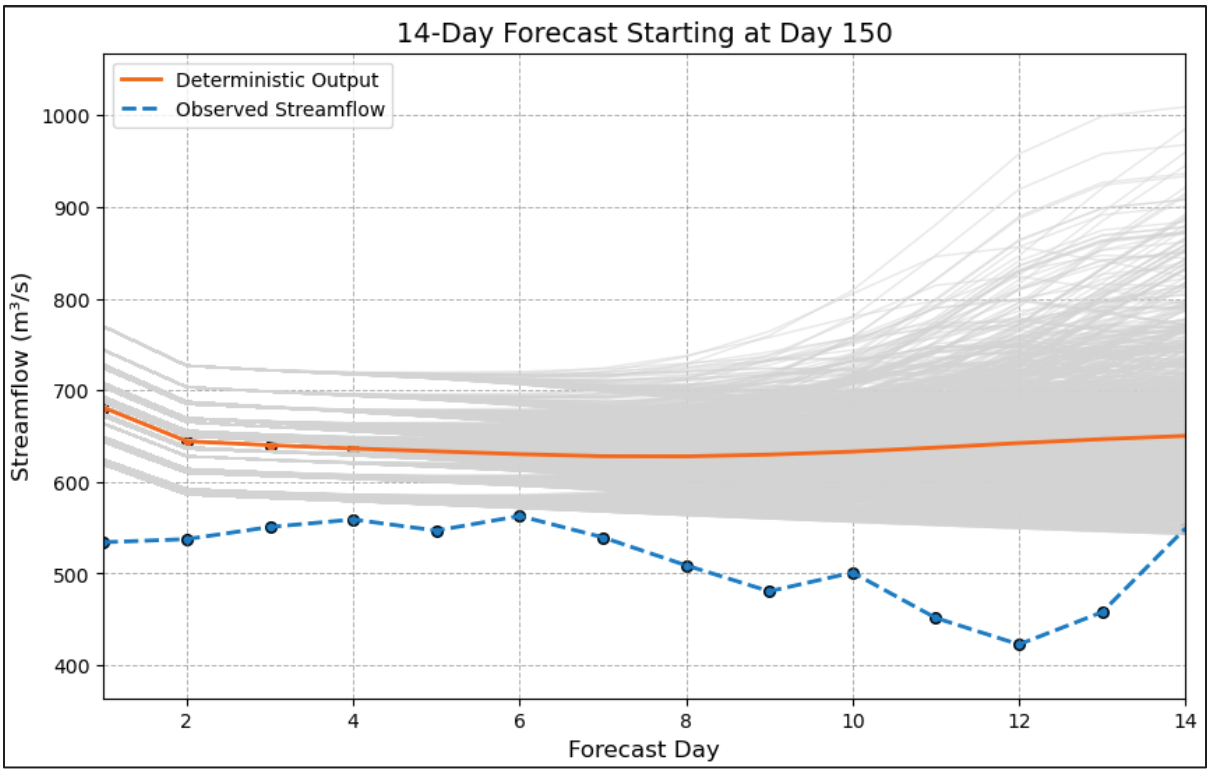

Figure-A II-4 Dispersion de l'ensemble avec assimilation de données sur un horizon de 14 jours commençant le jour 150 pour le modèle GR4J-CN

LISTE DE RÉFÉRENCES BIBLIOGRAPHIQUES

Arnell, N. W., & Gosling, S. N. (2013). The impacts of climate change on river flow regimes at the global scale. *Journal of Hydrology, 486*, 351-364. doi: https://doi.org/10.1016/j.jhydrol.2013.02.010

Armstrong, W., Arsenault, R., Martel, J-L., Troin, M., Dion, P., Sabzipour, B., Brissette, F., & Mai, J. (2024). A hybrid multi-model approach for hydrological ensemble forecasting: The added value of deep learning models. doi: http://dx.doi.org/10.2139/ssrn.4737841

Arsenault, R., Poulin, A., Côté, P., & Brissette, F. (2013). Comparison of Stochastic Optimization Algorithms in Hydrological Model Calibration. *Journal of Hydrologic Engineering, 19*(7), 1374-1384. Repéré à https://ascelibrary.org/doi/10.1061/%28ASCE%29HE.1943-5584.0000938#abstract

Baran, S., Hemri, S., & El Ayari, M. (2019). Statistical Postprocessing of Water Level Forecasts Using Bayesian Model Averaging with Doubly Truncated Normal Components. *Water Resources Research, 55*(5), 3397-4013. doi: https://doi.org/10.1029/2018WR024028

Bergeron, J. M., Trudel, M., & Leconte, R. (2016). Combined assimilation of streamflow and snow water equivalent for mid-term ensemble streamflow forecasts in snow-dominated regions. *Hydrol. Earth Syst. Sci., 20*(10), 4375-4389. doi: 10.5194/hess-20-4375-2016

Beven, K., & Binley, A. (1992). The future of distributed models: Model calibration and uncertainty prediction. *Hydrological Processes, 6*(3), 279-298. doi: 10.1002/hyp.3360060305

Chlumsky, R., Mai, J., Craig, J., & Tolson, B. (2024). Advancement of a Blended Hydrological Model for Robust Model Performance. *Journal of Hydrologic Engineering, 29*(5), 4024-4033. Repéré à https://ascelibrary.org/doi/full/10.1061/JHYEFF.HEENG-6246

Cloke, H. L., & Pappenberger, F. (2009). Ensemble flood forecasting: A review. *Journal of Hydrology, 375*(3-4), 613-626. doi: https://doi.org/10.1016/j.jhydrol.2009.06.005

Coulibaly, P., Haché, M., Fortin, V., & Bobée, B. (2005). Improving Daily Reservoir Inflow Forecasts with Model Combination. *Journal of Hydrologic Engineering, 10*(2), 91-99. doi: 10.1061/(ASCE)1084-0699(2005)10:2(91)

Davidson-Chaput, J., Arsenault, R., Martel, J.L. and Troin, M., 2025. Value of various elements of the hydrological forecasting chain: Is there a successful pathway for improving the overall performance?. *Journal of Hydrology*, p.133186.

DeChant, C. M., & Moradkhani, H. (2011). Improving the characterization of initial condition for ensemble streamflow prediction using data assimilation. *Hydrol. Earth Syst. Sci., 15*(11), 3399-3410. doi: 10.5194/hess-15-3399-2011

DeChant, C. M., & Moradkhani, H. (2012). Examining the effectiveness and robustness of sequential data assimilation methods for quantification of uncertainty in hydrologic forecasting. *Water Resources Research, 48*(4). doi: 10.1029/2011wr011011

Demargne, J., Wu, L., Regonda, S. K., Brown, J. D., Lee, H., He, M., ... & Zhu, Y. (2014). The science of NOAA's operational hydrologic ensemble forecast service. *Bulletin of the American Meteorological Society, 95*(1), 79-98. doi: https://doi.org/10.1175/BAMS-D-12-00081.1

Demeritt, D., Cloke, H., Pappenberger, F., Thielen, J., Bartholmes, J., & Ramos, M.-H. (2007). Ensemble predictions and perceptions of risk, uncertainty, and error in flood forecasting. *Environmental Hazards, 7*(2), 115-127. doi: 10.1016/j.envhaz.2007.05.001

Dion, P., Martel, J-L., Arsenault, R. (2021). Hydrological ensemble forecasting using a multi-model framework. *Journal of Hydrology, 600*. doi: https://doi.org/10.1016/j.jhydrol.2021.126537

Evensen, G. (1994). Sequential data assimilation with a nonlinear quasi-geostrophic model using Monte Carlo methods to forecast error statistics. *Journal of Geophysical Research: Oceans, 99*(C5), 10143-10162.doi: https://doi.org/10.1029/94JC00572

Evensen, G. (2003). The Ensemble Kalman Filter: theoretical formulation and practical implementation. *Ocean Dynamics, 53*(4), 343-367. doi: https://doi.org/10.1007/s10236-003-0036-9

Fatichi, S., Vivoni, E. R., Ogden, F. L., Ivanov, V. Y., Mirus, B., Gochis, D., ... & Tarboton, D. (2016). An overview of current applications, challenges, and future trends in distributed process-based models in hydrology. *Journal of Hydrology, 537*, 45-60. doi: https://doi.org/10.1016/j.jhydrol.2016.03.026

Goodarzi, M., Niknam, A., & Sabaghzadeh, M. (2022). Chapter 11 - Rainfall-runoff modeling using GIS: A case study of Gorganrood Watershed, Iran. *Water Resource Modeling and Computational Technologies, 7*(11), 165-181. doi: https://doi.org/10.1016/B978-0-323-91910-4.00011-X

Gupta, H., Kling, H., Yilmaz, K., & Martinez, G. (2009). Decomposition of the mean squared error and NSE performance criteria: Implications for improving hydrological modelling. *Journal of Hydrology, 377*(1), 80-91. doi: https://doi.org/10.1016/j.jhydrol.2009.08.003

Hamill, T. M. (2001). Interpretation of Rank Histograms for Verifying Ensemble Forecasts. *Monthly Weather Review, 129*(3), 550-560. doi: 10.1175/1520-0493(2001)129

Hemri, S., Fundel, F., & Zappa, M. (2013). Simultaneous calibration of ensemble river flow predictions over an entire range of lead times. *Water Resources Research, 49*(10), 6744-6755. doi: 10.1002/wrcr.20542

Her, Y., Yoo, S. H., Cho, J., Hwang, S., Jeong, J., & Seong, C. (2019). Uncertainty in hydrological analysis of climate change: multi-parameter vs. multi-GCM ensemble predictions. *Scientific Reports, 9*(1), 4974. doi: https://doi.org/10.1038/s41598-019-41334-7

Jie, M.-X., Chen, H., Xu, C.-Y., Zeng, Q., & Tao, X.-e. (2016). A comparative study of different objective functions to improve the flood forecasting accuracy. *Hydrology Research, 47*(4), 718-735. doi: https://doi.org/10.2166/nh.2015.078

Kalman, R. E. (1960). A new approach to linear filtering and prediction problems. *Journal of Basic Engineering, 82*(1). doi: https://doi.org/10.1115/1.3662552

Kavetski, D., Kuczera, G., & Franks, S. W. (2006). Bayesian analysis of input uncertainty in hydrological modeling: 1. Theory. *Water Resources Research, 42*(3).doi: https://doi.org/10.1029/2005WR004368

Krzysztofowicz, R. (2001). The case for probabilistic forecasting in hydrology. *Journal of Hydrology, 249*(1), 2-9. doi: https://doi.org/10.1016/S0022-1694(01)00420-6

Leutbecher, M., Lock, S. J., Ollinaho, P., Lang, S. T., Balsamo, G., Bechtold, P., ... & Weisheimer, A. (2017). Stochastic representations of model uncertainties at ECMWF: State of the art and future vision. *Quarterly Journal of the Royal Meteorological Society, 143*(707), 2315-2339. doi: https://doi.org/10.1002/qj.3094

Liu, Y., Weerts, A. H., Clark, M., Hendricks Franssen, H. J., Kumar, S., Moradkhani, H., ... & Restrepo, P. (2012). Advancing data assimilation in operational hydrologic forecasting: progresses, challenges, and emerging opportunities. *Hydrol. Earth Syst. Sci., 16*(10), 3863-3887. doi: 10.5194/hess-16-3863-2012

Mai, J., Arsenault, R., Tolson, B., Latraverse, M., & Demeester, K. (2020). Application of Parameter Screening to Derive Optimal Initial State Adjustments for Streamflow Forecasting. *Water Resources Research, 56*(9). doi: https://doi.org/10.1029/2020WR027960

Martel, J.-L., Demeester, K., Brissette, F., Poulin, A., & Arsenault, R. (2017). HMETS-A simple and efficient hydrology model for teaching hydrological modelling, flow forecasting and climate change impacts. *International Journal of Engineering Education, 33*(4), 1307-1316.

Mazrooei, A., & Sankarasubramanian, A. (2019). Improving monthly streamflow forecasts through assimilation of observed streamflow for rainfall-dominated basins across the CONUS. *Journal of Hydrology, 575*, 704-715. doi: https://doi.org/10.1016/j.jhydrol.2019.05.071

Maxwell, D. H., Jackson, B. M., & McGregor, J. (2018). Constraining the ensemble Kalman filter for improved streamflow forecasting. *Journal of Hydrology, 560*, 127-140. doi: https://doi.org/10.1016/j.jhydrol.2018.03.015

McCollor, D., & Stull, R. (2008). Hydrometeorological Short-Range Ensemble Forecasts in Complex Terrain. Part II: Economic Evaluation. *Weather and Forecasting, 23*(4), 557-574. doi: 10.1175/2007waf2007064.1

Meng, X., Zhao, H., Shu, T., Zhao, J., & Wan, Q. (2024). Machine learning-based spatial downscaling and bias-correction framework for high-resolution temperature forecasting. *Appl Intell, 54*, 8399-8414. doi: https://doi.org/10.1007/s10489-024-05504-z

Montanari, A., & Grossi, G. (2008). Estimating the uncertainty of hydrological forecasts: A statistical approach. *Water Resources Research, 44*(12). doi: 10.1029/2008wr006897

Oudin, L., Hervieu, F., Michel, C., Perrin, C., Andréassian, V., Anctil, F., & Loumagne, C. (2005). Which potential evapotranspiration input for a lumped rainfall–runoff model?: Part 2—towards a simple and efficient potential evapotranspiration model for rainfall–runoff modelling. *Journal of Hydrology, 303*(1-4), 290-306. doi: https://doi.org/10.1016/j.jhydrol.2004.08.026

Pagano, T. C., Pappenberger, F., Wood, A. W., Ramos, M. H., Persson, A., & Anderson, B. (2016). Automation and human expertise in operational river forecasting. *Wiley Interdisciplinary Reviews: Water, 3*(5), 692-705. doi: 10.1002/wat2.1163

Pagano, T. C., Shrestha, D. L., Wang, Q. J., Robertson, D., & Hapuarachchi, P. (2013). Ensemble dressing for hydrological applications. *Hydrological Processes, 27*(1), 106-116. doi: 10.1002/hyp.9313

Pappenberger, F., Thielen, J., & Del Medico, M. (2011). The impact of weather forecast improvements on large-scale hydrology: analysing a decade of forecasts of the European Flood Alert System. *Hydrological Processes, 25*(7), 1091-1113. doi: 10.1002/hyp.7772

Ratto, M., Young, P. C., Romanowicz, R., Pappenberger, F., Saltelli, A., & Pagano, A. (2007). Uncertainty, sensitivity analysis and the role of data-based mechanistic modeling in hydrology. *Hydrol. Earth Syst. Sci., 11*(4), 1249-1266. doi: https://doi.org/10.5194/hess-11-1249-2007

Reggiani, P., & Weerts, A. H. (2008). A Bayesian approach to decision-making under uncertainty: An application to real-time forecasting in the river Rhine. *Journal of Hydrology, 356*(1), 56-69. doi: https://doi.org/10.1016/j.jhydrol.2008.03.027

Roundy, K. R., Duan, Q., & Schaake, J. C. (2019). Hydrological Predictability, Scales, and Uncertainty Issues. Dans Q. Duan, F. Pappenberger, A. Wood, H. L. Cloke, & J. C. Schaake (Éds), Handbook of Hydrometeorological Ensemble Forecasting. Berlin, Heidelberg: Springer Berlin Heidelberg. doi: 10.1007/978-3-642-39925-1

Schaake, J. C., Demargne, J., Hartman, R., Mullusky, M., Welles, E., Wu, L., ... & Seo, D. J. (2007). Precipitation and temperature ensemble forecasts from single-value forecasts. *Hydrol. Earth Syst. Sci. Discuss., 2007*, 655-717. doi: 10.5194/hessd-4-655-2007

Schaake, J. C., Hamill, T. M., Buizza, R., & Clark, M. (2007). HEPEX: The Hydrological Ensemble Prediction Experiment. *Bulletin of the American Meteorological Society, 88*(10), 1541-1548. doi: 10.1175/bams-88-10-1541

Schwanenberg, D., Fan, F. M., Naumann, S., Kuwajima, J. I., Montero, R. A., & Assis dos Reis, A. (2015). Short-Term Reservoir Optimization for Flood Mitigation under Meteorological and Hydrological Forecast Uncertainty. *Water Resources Management, 29*(5), 1635-1651. doi: 10.1007/s11269-014-0899-1

Talagrand, O., Vautard, R. (1997). Evaluation of probabilistic prediction systems. Communication présentée au Workshop on Predictability, Paris. Repéré à : https://www.ecmwf.int/en/elibrary/76596-evaluation-probabilistic-prediction-systems

Tarek, M., Brissette, F., & Arsenault, R. (2020). Evaluation of the ERA5 reanalysis as a potential reference dataset for hydrological modelling over North America. *Hydrol. Earth Syst. Sci., 24*(5), 2527-2544. doi: 10.5194/hess-24-2527-2020

Thiboult, A., & Anctil, F. (2015). On the difficulty to optimally implement the Ensemble Kalman filter: An experiment based on many hydrological models and catchments. *Journal of Hydrology, 529*(3), 1147-1160. doi: https://doi.org/10.1016/j.jhydrol.2015.09.036

Velázquez, J. A., Anctil, F., Ramos, M. H., & Perrin, C. (2011). Can a multi-model approach improve hydrological ensemble forecasting? A study on 29 French catchments using 16 hydrological model structures. *Adv. Geosci., 29*, 33-42. doi: https://doi.org/10.5194/adgeo-29-33-2011, 2011.

Wetterhall, F., Pappenberger, F., Alfieri, L., Cloke, H., & Thielen, J. (2014). Forecaster priorities for improving probabilistic flood forecasts présentée à *EGU General Assembly Conference Abstracts*. Repéré à https://ui.adsabs.harvard.edu/abs/2014EGUGA..16.8159W

Wetterhall, F., Pappenberger, F., Alfieri, L., Cloke, H. L., Thielen-del Pozo, J., Balabanova, S., ... & Holubecka, M. (2013). HESS Opinions "Forecaster priorities for improving probabilistic flood forecasts". *Hydrol. Earth Syst. Sci., 17*(11), 4389-4399. doi: 10.5194/hess-17-4389-2013