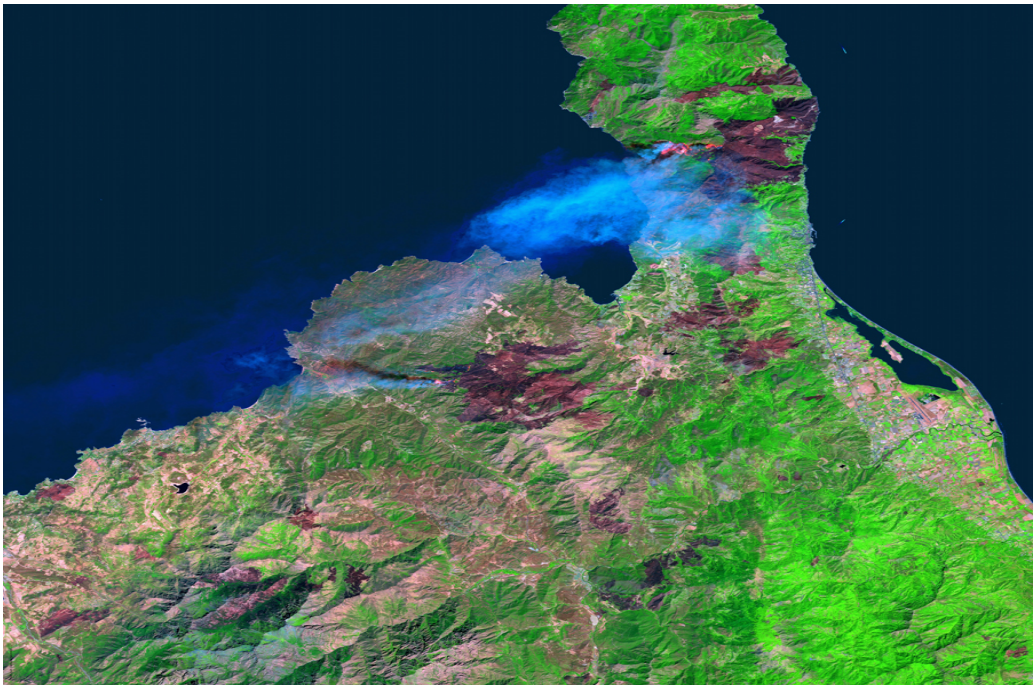
École Nationale de la Météorologie
Centre Européen de Recherche et de Formation Avancée en Calcul Scientifique
Encadrantes : M. Rochoux[1,2] et S. Ricci[1]

[1] *CERFACS/CNRS-URA1875,* [2] *École Centrale Paris*

CERFACS Technical Report

Caroline Allouache, Mickaël Durand et Maxime Taillardat
14 Février 2013

Assimilation de données pour la propagation des feux de forêt



Observation des feux de forêt en Corse en septembre 2003 par le satellite SPOT 5. (*CNES*©)

Projet de modélisation de deuxième année de l'École Nationale de la Météorologie
Année scolaire 2012-2013

Remerciements

Comment remercier suffisamment Sophie Ricci et Mélanie Rochoux sans lesquelles ce projet n'aurait pas vu le jour et qui nous ont offert l'opportunité d'y participer ? Nous n'oublierons pas leur disponibilité, leur écoute ainsi que le temps et l'énergie qu'elles nous ont accordés ces cinq dernières semaines. Nous retiendrons grâce à elles qu'il n'est pas impossible de s'atteler à des tâches difficiles tout en gardant sa bonne humeur. Une mention spéciale doit être faite à Thierry Morel pour son aide et son savoir-faire en matière de débogage, qui nous ont fait gagner un temps précieux.

Sommaire

Introduction

Présentation du CERFACS

Le **CERFACS** (Centre Européen de Recherche et de Formation Avancée en Calcul Scientifique) est un organisme de recherche travaillant sur les simulations numériques et les solutions algorithmiques dans un grand nombre de domaines scientifiques et techniques. Pour mener à bien ses activités, le CERFACS utilise des moyens de calcul à haute performance (HPC).

Il est dirigé par un Conseil de Gérance qui représente ses actionnaires, et bénéficie des recommandations du Conseil Scientifique. Les sept actionnaires du CERFACS sont : le **CNES** (Centre National d'Etudes Spatiales) ; **EADS** (European Aeronautic and Defense Space Company) ; **EDF** (Electricité de France) ; **Météo-France** ; l'**ONERA** (Office National d'Etudes et de Recherches Aérospatiales) ; **SAFRAN**, équipementier international de hautes technologies, et **TOTAL**.

Le CERFACS emploie environ 115 personnes, dont plus de 95 chercheurs et ingénieurs, au travers d'équipes interdisciplinaires comprenant physiciens, chercheurs en mathématiques appliquées, analystes numériques, et ingénieurs programmeurs.

Les principaux domaines de recherche du CERFACS sont : les algorithmes parallèles, le couplage de codes numériques, l'aérodynamique, les turbines à gaz, la combustion, le climat, l'impact environnemental, l'assimilation de données, les champs électromagnétiques et acoustiques, l'analyse numérique et l'optimisation (Fig. 1).

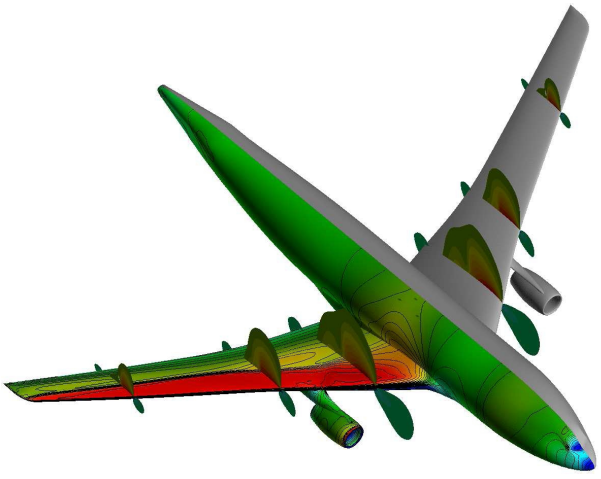


FIGURE 1 – Optimisation de la traînée d'un avion (*cerfacs.fr*)

Le CERFACS est associé au **CNRS** (Centre National de Recherche Scientifique) au travers de l'URA1875 qui englobe les activités des équipes **GLOBC** (GLOBal Change & climate modeling) et **PAE** (Aviation Environment). Une collaboration avec l'**INRIA** (Institut National de Recherche en Informatique et Automatique) est formalisée par le groupe **HIEPACS** qui regroupe des activités de l'équipe **ALGO** (ALGOrithme et calcul parallèle).

Le CERFACS participe également aux programmes de **TVE** (Terre Vivante et Espace) et d'**AESE** (Aéronautique, Espace et Systèmes Embarqués), et est membre du pôle **RTRA/STAE** (Réseau Thématique de Recherche Avancée Sciences et Technologies pour l'Aéronautique et l'Espace).

Présentation du projet de modélisation

La prévision de la vitesse et de la direction d'un feu de forêt représente un enjeu évident mais demeure un défi de taille dans la mesure où de nombreux paramètres des modèles de propagation relatifs à la description de la végétation (type, humidité, répartition) et de l'environnement (pente, conditions météorologiques) sont mal connus. La combinaison de la simulation numérique et de l'observation avec des méthodes d'assimilation de données permet d'améliorer la modélisation de la vitesse de propagation du front de flamme et par conséquence la prévision de la propagation du front de flamme. Les travaux décrits dans [MR12] et [MR13] décrivent l'implémentation d'un Filtre de Kalman d'Ensemble (EnKF) sur un simulateur eulérien de propagation de front de type level-set (FireFly) ; dans ce cadre, la grandeur physique d'interêt est la vitesse de propagation du front, aussi appelée *Rate Of Spread* (*ROS*). Elle est décrite en fonction des propriétés locales de la végétation et des conditions de vent. Les paramètres d'humidité, de la vitesse et de la direction du vent et du rapport surface/volume des particules de végétation sont corrigés par l'assimilation d'observations des positions du front de flamme pour un feu de prairie contrôlé de petite dimension (*cf.* Fig. 2). Ici, l'assimilation de données est faite dans FireFly pour un feu de petite dimension ($4m \times 4m$). Il y a à droite les observations du temps d'arrivée du front (correspondant à l'isocontour de température $600K$ sur l'imagerie moyen infrarouge) au temps de l'assimilation ($t = 78s$, en trait plein) et au temps de la prévision ($t = 106s$, ligne en tireté). On a à gauche les résultats de la prévision à $t = 106s$ avec FireFly : les observations à $t = 106s$ sont représentés par les points noirs le front simulé (après assimilation à $t = 78s$) est plus proche des observations et possède un écart-type réduit par rapport à la simulation libre (sans assimilation, en bleu).

Pour poursuivre ce travail et répondre au défi de la mise en place opérationnelle, le simulateur de propagation au sein de la maquette d'assimilation doit être amélioré afin de permettre la prise en compte de la topographie, des variations spatiales des composantes du vent et de réaliser des simulations de feux naturels à échelle régionale, ce qui n'est pas immédiat avec le simulateur FireFly ; aussi, dans le cadre de ce projet de modélisation, on propose d'interfacer le modèle lagrangien de propagation ForeFire développé par le SPE (Laboratoire des Sciences pour l'Environnement de l'Université de Corte), au sein de la maquette d'assimilation de données existante développée sous le coupleur de codes dynamique OpenPALM (mis au point au CERFACS en collaboration avec l'ONERA). Le but est ainsi de procéder à une étude de sensibilité des paramètres à corriger et à terme, d'implémenter l'algorithme d'assimilation de données sur un cas test dans le contexte OSSE (Observing System Simulation Experiment, ou Expériences jumelles), c'est-à-dire pour un feu idéalisé muni d'observations synthétiques. Soulignons le fait que ForeFire a déjà été couplé avec le modèle atmosphérique Méso-NH [JF12] dans le cadre du projet national de recherche ANR-IDEA (Incendies, de la Dynamique aux Émissions Atmosphériques), ce qui permet d'envisager à long terme l'assimilation de données sur le modèle couplé ForeFire/Méso-NH. Ici on s'intéresse à interfacer uniquement le code ForeFire non couplé comme une étape préliminaire.

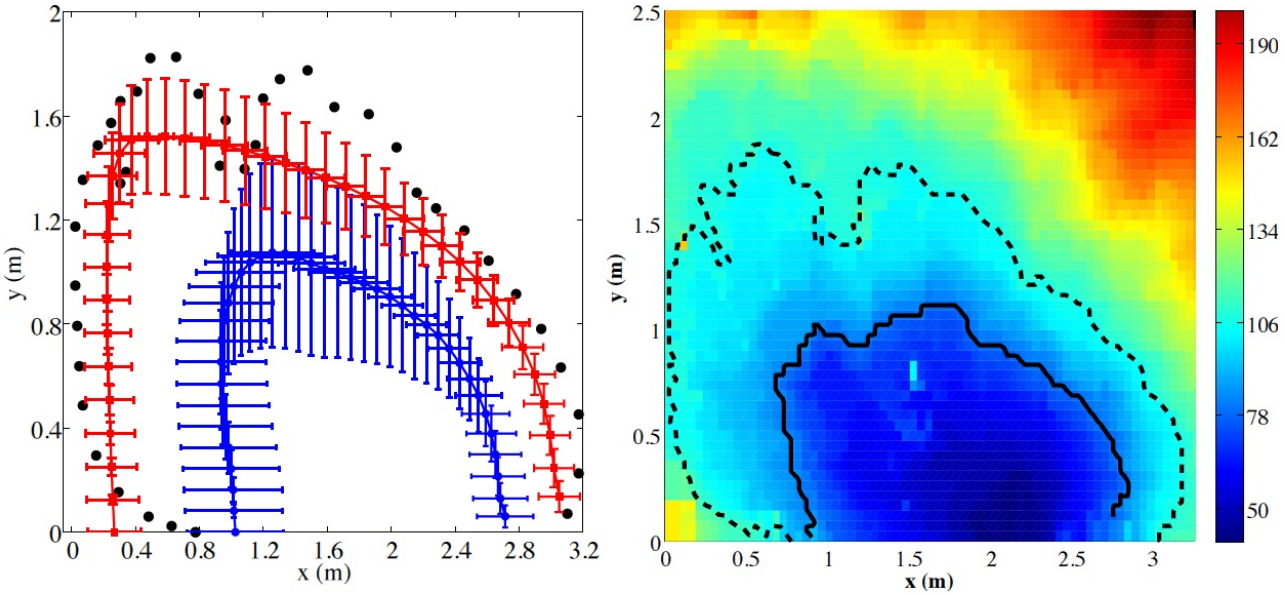


FIGURE 2 – FireFly appliqué à un feu de petite dimension (*M. Rochoux*)

Ce rapport comporte 3 parties. La première situe le problème dans son contexte en présentant tour à tour la physique du problème et le simulateur de propagation ForeFire. La seconde partie montre la méthodologie utilisée pour pratiquer l'étude de sensibilité indispensable à l'identification des paramètres à contrôler *via* l'assimilation, ainsi que les modifications techniques apportées à ForeFire pour le rendre compatible avec le coupleur de codes OpenPALM. La troisième partie porte sur la description de l'algorithme EnKF et la méthodologie utilisée pour appliquer l'assimilation de données à ForeFire *via* le coupleur OpenPALM ainsi que la présentation des premiers résultats de l'assimilation.

Partie 1

Position du problème

1.1 Prérequis sur la physique et la modélisation de la propagation d'un feu

Les feux de forêt sont responsables de plusieurs types de dégâts dévastateurs : une perte de biodiversité, la diminution des surfaces végétales, la dégradation des sols, l'augmentation de l'effet de serre, etc. Pour combattre ce risque naturel et aider à la décision dans le cadre de la lutte anti-incendies, la communauté scientifique tente de développer des méthodes pour anticiper le scénario le plus probable de propagation du feu avec une plus grande précision. Plusieurs stratégies pour la modelisation des feux de forêt ont été développées. Ces modèles doivent permettre le dimensionnement des zones coupe-feu, la gestion de la couverture végétale et la définition de zones d'intervention pour les secours. Pour aider à la prise de décision dans la cadre d'un incendie et permettre d'ajuster les opérations de lutte sur le terrain, le coût en temps de calcul de ces modèles et l'utilisation de mémoire doit être plus court que la durée réelle de la propagation du feu.Autrement dit, le simulateur de propagation doit être capablee de simuler un feu d'échelle régionale en respectant ces contraintes opérationnelles. De plus, les responsables forestiers et les pompiers veulent connaître la cinématique du front de feu mais aussi les caractéristiques liées telles que la hauteur de flamme, l'angle d'inclinaison de flamme etc.

Le type de modèle qui combine à la fois vraisemblance et coût informatique réduit est le modèle semi-empirique, alliant une prise en compte partielle de la représentation des mécanismes physiques intervenant dans un feu et les statistiques de feux expérimentaux (de laboratoire ou à l'air libre). À grande échelle, on peut décrire le feu comme la propagation d'une interface entre la zone brûlée de la végétation et la zone non brûlée. Les modèles de propagation d'un feu de forêt reposent donc sur le calcul d'une vitesse-modèle du front (*Rate Of Spread ROS*) qui est fonction de plusieurs paramètres du terrain et de la végétation dont les principaux (ceux auxquels nous nous intéressons ici) sont l'humidité (la teneur en eau de la végétation), la topographie, le vent et le rapport surface/volume des particules de combustible constituant la couche de végétation. La végétation peut en effet être assimilée à un milieu poreux, constitué d'air et de particules solides qui ont un certain nombre de caractéristiques. Ces propriétés varient d'une espèce végétale à une autre ; elles déterminent le délai de temps nécessaire à la végétation pour s'enflammer et donc la vitesse à laquelle le front peut se propager dans la couche de végétation. Contrairement à un modèle purement physique qui peut résoudre toute la physique d'un feu et cela pour toutes les échelles spatiales, un modèle semi-empirique se concentre uniquement sur le transport de l'interface entre la flamme et la zone non brûlée de la végétation par l'intermédiaire du *ROS*, dont la définition repose sur le principe de conservation de l'énergie. Nous pouvons observer sur la Fig. 3 les différents phénomènes rencontrés dans la

physique d'un feu :

- La combustion dégage de l'énergie sous forme thermique (chaleur). Sous l'effet de cette chaleur, le combustible se décompose et libère des gaz inflammables (CO, CH_4, CO_2) entretenant la combustion : c'est la pyrolyse de la végétation.
- La dynamique de la combustion est influencée par les échanges thermiques entre la flamme et la végétation mais aussi par le vent et la pente. En effet, dans la direction du vent et les directions de forte pente, la flamme est plus inclinée vers la végétation non brûlée, ce qui accélère la pyrolyse. De plus, le feu se propage dans l'axe du vent d'autant plus vite que le vent est fort. Grâce à cet "effet de pente", le feu se propage plus rapidement dans les directions où la pente augmente. *A contrario*, cette convection ralentit la propagation d'un feu descendant une pente. Le vent peut même charrier des éléments incandescents en avant du front et ainsi "sauter" des zones incombustibles comme une route par exemple.
- La dynamique atmosphérique locale s'en trouve modifiée. En effet, l'eau contenue dans la végétation se transforme en vapeur et la température au sol extrême favorise l'échauffement de l'air et les mouvements convectifs ascendants. Nous pourrions dire qu'un feu de grande ampleur a des conséquences météorologiques à l'échelle locale. Tout d'abord le feu forme un panache puis les changements/fluctuations du vent au voisinage du feu modifient le comportement du feu et inversement : il existe donc des rétroactions entre l'atmosphère et le feu.

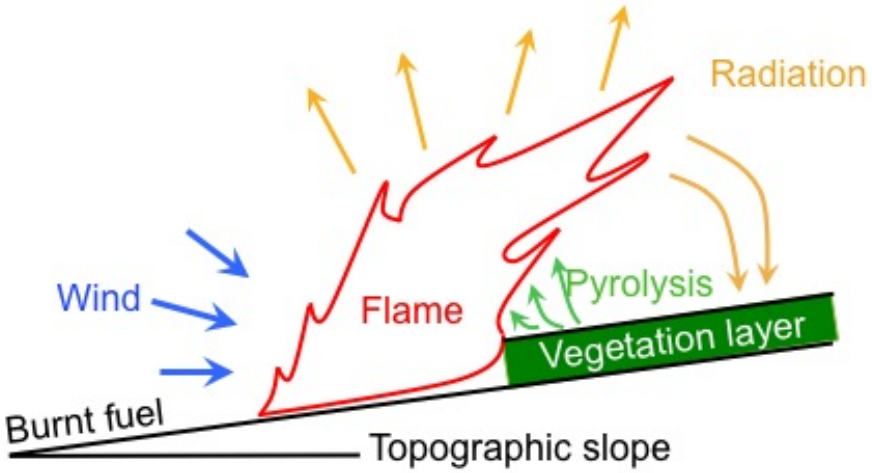


FIGURE 3 – Phénomènes physiques rencontrés au voisinage d'un front de flammes (*M. Rochoux*)

Le feu est le résultat des interactions multi-physiques multi-échelles entre pyrolyse, transferts thermiques, dynamique de l'air, combustion etc.. La simulation complète de la physique du feu étant trop coûteuse pour de grandes configurations (feux supérieurs à plusieurs centaines de mètres) et étant limitée à des cas expérimentaux, nous utilisons un modèle semi-empirique pour résoudre des feux à l'échelle régionale. Les phénomènes de petite échelle sont alors pris en compte de manière globale dans la formulation du *ROS*. Dans la suite, nous nous intéressons en détail à deux modèles semi-empiriques : ceux de Rothermel [Rot72] et de Balbi [JB09].

Les modèles semi-empiriques expriment une approximation de la vitesse de propagation : le *ROS* noté R pour la suite. Cette vitesse est fonction de paramètres tels que la vitesse et la direction du vent U, la pente α, l'humidité de la végétation M_d et d'autres paramètres du combustible comme le rapport surface/volume des particules de végétation réunis sous l'appellation V_f :

$$R = R(U, \alpha, M_d, V_f) \tag{1.1}$$

Le modèle de Rothermel calcule la vitesse par le biais de l'équation de conservation de l'energie seulement, la fermeture du modèle étant gérée par des corrélations expérimentales sur des feux de laboratoire. Des paramètres intermédiaires nécessaires à la modélisation, n'apparaissant ni à l'entrée, ni à la sortie du modèle, sont ainsi entièrement déterminés par ces expériences. La conservation de l'énergie se fait par le calcul de flux dans la zone non brûlée à proximité de la flamme, qui seront directement utilisés dans la formulation de la vitesse (*cf.* Fig. 4) La vitesse R $(m.s^{-1})$, décrite dans l'équation 1.1, est calculée comme le rapport du flux propagé vers la zone non brûlée I_p $(J.m^{-2}.s^{-1})$ sur l'énergie nécessaire pour brûler ce volume de végétation $\rho_f h_{ig}$ $(J.m^{-3})$, en effet ρ_f est la densité du combustible (kg/m^{-3}) et h_{ig} l'enthalpie massique $(J.kg^{-1})$ nécessaire à la combustion.

$$R = \frac{I_p}{\rho_f h_{ig}} \tag{1.2}$$

Le flux propagé I_p a différentes contributions (au sein de la couche de végétation et de la flamme) et peut être défini comme suit :

$$I_p = I_0(1 + \Phi_s + \Phi_w) \tag{1.3}$$

où Φ_s et Φ_w sont des coefficients prenant respectivement en compte la pente et le vent et I_0 est le flux propagé sans vent ni pente. Les effets du vent et de la pente dans ce modèle sont donc considérés comme proportionnels à I_0. Le point clé du modèle de Rothermel a été de faire le lien entre le flux d'énergie reçu par la végétation sans vent ni pente I_0 et le flux d'énergie émis par la combustion des gaz noté I_R. Ce dernier terme se calcule *via* :

$$I_0 = \xi I_R \tag{1.4}$$

où ξ un coefficient dépendant du combustible (en termes de compacité et de rapport surface/-volume des particules de végétation).

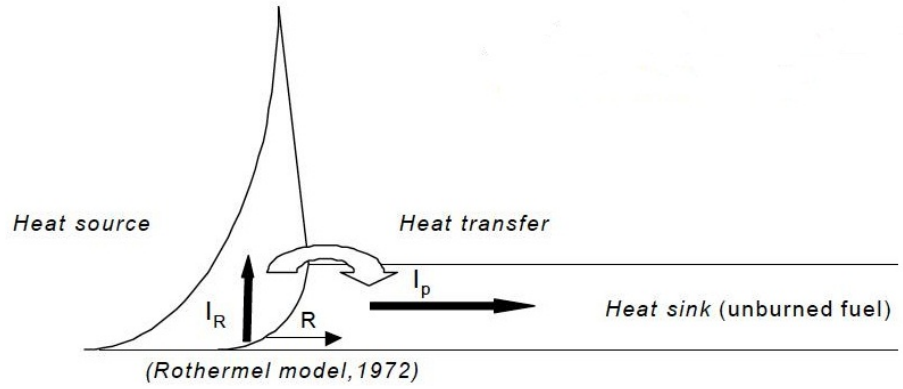


FIGURE 4 – Calcul du *ROS* R dans le modèle de Rothermel ([Dup97])

Bien que le modèle de Balbi ait des paramètres d'entrée caractérisant la végétation communs avec celui de Rothermel comme le vent, la topographie, la teneur en eau du sol, l'enthalpie de combustion et le rapport surface/volume (D. Gorham, 2012), il se base sur toutes les équations de la dynamique (conservation de la masse, de la quantité de mouvement et de l'énergie) et prend en compte le rayonnement. La fermeture du modèle se fait alors par des hypothèses simplificatrices (forme de flamme triangulaire, hypothèse de Mach faible, rayonnement prépondérant dans la zone de pré-chauffage etc.). Nous pouvons dire de fait que c'est un modèle physique simplifié. Il possède néanmoins la même rapidité de calcul qu'un modèle semi-emprique classique.

Comme le montre la Fig. 5, le modèle de *ROS* (défini à partir des conditions locales) est intégré dans un simulateur de propagation de front afin d'obtenir l'évolution des positions du front de feu à échelle régionale (ou échelle macroscopique).

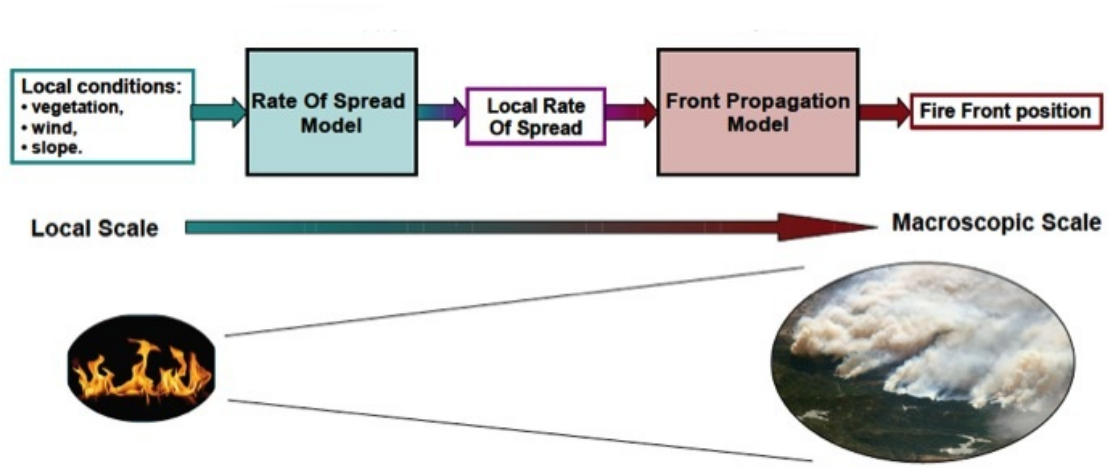


FIGURE 5 – Architecture entre les modèles de vitesse de propagation et les simulateurs de propagation. ([Del11])

Pour réaliser la trace du front à une échelle macroscopique, les simulateurs de propagation utilisent ce que l'on appelle des *marqueurs*. Dans un simulateur lagrangien, ces marqueurs ne sont ni plus ni moins que des points situés sur le front et suivis au cours du temps, des marqueurs pouvant être ajoutés ou enlevés suivant leur distance les uns par rapport aux autres (ce qui permet de garantir une résolution minimale de la simulation).

Quant aux simulateurs level-set tels que FireFly, ils ne caractérisent pas la trace d'un front sous forme de points mais sous la forme d'une isoligne d'un champ 2D. Le *ROS* étant une fonction dépendant de l'espace et du temps, on définit la variable de progrès c comme marqueur de flamme : c vaut 0 dans la zone non brûlée et 1 dans la zone brulée. La trace du front est ainsi prise sur l'iso-contour $c_f = 0.5$. c est alors définie comme la solution de l'équation de propagation dite *équation level-set* :

$$\frac{\partial c}{\partial t} = R\,|\nabla c| \tag{1.5}$$

où R est le *ROS* calculé localement par le modèle semi-empirique dans direction normale au front.

L'état de développement actuel des simulateurs est décrit pour FireFly dans [MR12], [MR13] et pour ForeFire sur le site texttthttp ://forefire.univ-corse.fr. Notons que le simulateur ForeFire a déjà été testé sur des feux réels tirés de la base de données `Prométhée` qui recense les incendies de forêt dans le bassin méditerranéen depuis 1973. On peut retrouver le résultat des comparaisons entre les feux observés avec les feux simulés par ForeFire à l'adresse

`http://forefire.univ-corse.fr/FireCases/`. Enfin, une version "ludique" de ForeFire utilisable en ligne *via* `Google Earth` est disponible sur `http://forefire.univ-corse.fr/websim2/`. En guise d'exemple, voici sur la Fig. 6 une simulation ForeFire : Les zones délimitées par les traits bleus correspondent à différents types de végétation mis en données dans ForeFire, le pictogramme de feu indique la position observée du départ de feu, l'aire rouge correspond à la zone brûlée observée et la courbe rouge trace la position de front simulée par ForeFire.

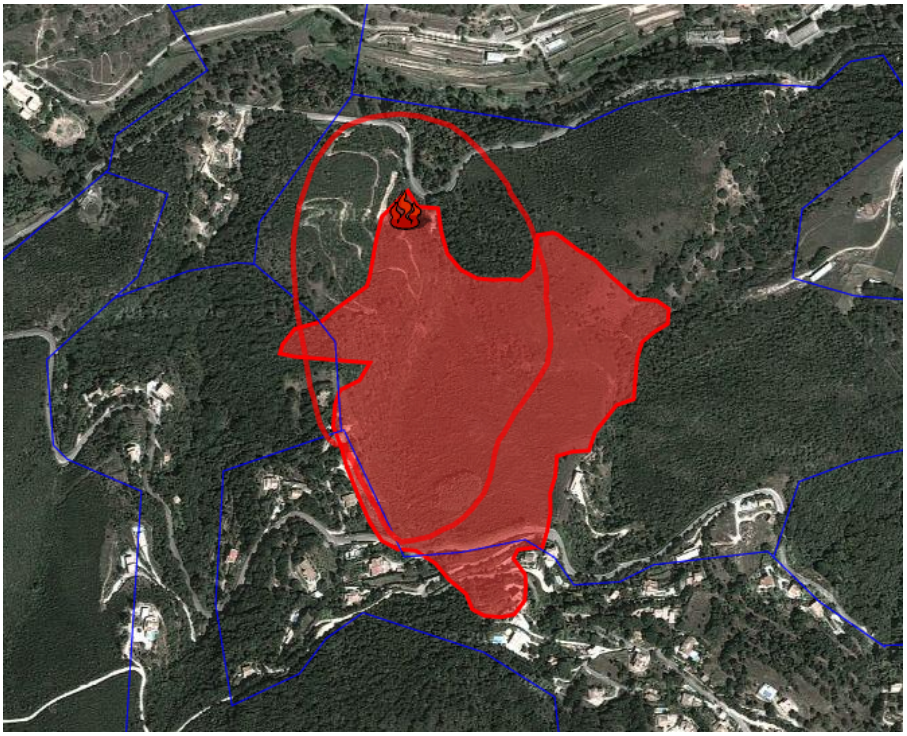


FIGURE 6 – Vue aérienne tirée de Prométhée avec la simulation ForeFire

1.2 Le simulateur de propagation de fronts ForeFire

Développé par le SPE (Systèmes Physiques de l'Environnement) de l'Université de Corte, ForeFire est à la fois une interface de programmation (*API* : Application Programming Interface), une librairie et un interpréteur dont les codes sources sont écrits en `C++`. Le but de ce code est de propager une interface (des fronts actifs de feux, de laves) *via* l'estimation de la vitesse de propagation par un modèle (semi-)empirique ou physique. ForeFire a été conçu de sorte à ce qu'il puisse être utilisé simplement par lignes de commandes (interpréteur en synthaxe `Python`), couplé à d'autres codes (Méso-NH) et enrichi par des modèles de propagation et d'émission (simulation de panaches) définis par l'utilisateur. La Fig. 7 présente le fonctionnement et la mise en données de ForeFire.

1.2.1 Configuration d'incendie

Le fichier de configuration nommé `FireSimulation.ff` permet la mise en données de la simulation, on y retrouve le modèle d'estimation de *ROS* utilisé, les résolutions spatiale et temporelle de la simulation, la position du front initial ainsi que l'emplacement des fichiers de sortie. Il comporte aussi l'emplacement du fichier de données `NetCDFfile` qui permet à

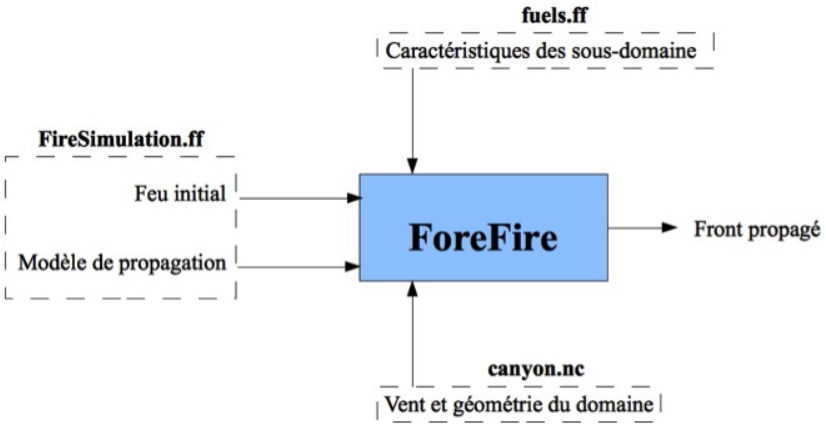


FIGURE 7 – Schéma du fonctionnement de ForeFire

l'utilisateur de spécifier le nom du fichier d'entrée de mise en donnée `NetCDF` (de vent et de topographie) et l'emplacement du fichier `fluxNetCDFfile` qui met en donnée les modèles de flux de chaleur et de vapeur utilisés. La variable `fuelsTableFile` donne l'emplacement d'un fichier décrivant les types de végétation et leurs caractéristiques.

Voici un exemple de ce fichier de configuration pour un feu se propageant selon le modèle de Balbi. Le fichier `FireSimulation.ff` contient les lignes suivantes et fait appel à un certain nombre de routines `C++` (`setParameters`,`FireNode`,etc.) :

```
setParameter[caseDirectory=/home/globc/taillardat/projetmod/forefire.0.beta1/examples/
    03_Canyon/01_forefire/]
setParameters[fireOutputDirectory=Outputs/;outputsUpdate=15]
setParameters[NetCDFfile=canyon.nc;fuelsTableFile=fuels.ff;fluxNetCDFfile=canyon.nc]
setParameters[propagationModel=BalbiUnsteady;spatialIncrement=0.15;perimeterResolution=0.5]
FireDomain[sw=(0.,0.,0.);ne=(640.,320.,0.);t=0.]
    FireFront[t=15]
        FireNode[loc=(296,130,0);vel=(-0.5,-0.5,0);t=15]
        FireNode[loc=(296,190,0);vel=(-0.5,0.5,0);t=15]
        FireNode[loc=(304,190,0);vel=(0.5,0.5,0);t=15]
        FireNode[loc=(304,130,0);vel=(0.5,-0.5,0);t=15]
step[dt=150s]
print[]
save[]
```

- Les paramètres de simulation `caseDirectory` et `fireOutputDirectory` permettent de spécifier, respectivement, le dossier de travail dans lequel est stocké le fichier `FireSimulation.ff`, et le dossier dans lequel sont stockés les fichiers de sortie de simulation. (Il est préférable de noter le chemin absolu du dossier de travail dans le paramètre `caseDirectory`. Pour les autres dossiers, il n'est plus nécessaire d'indiquer leur chemin absolu, mais plutôt leur chemin relatif à partir du dossier de travail.)
- Le paramètre `outputsUpdate` permet à l'utilisateur de spécifier la fréquence d'écriture de fichiers de sortie `Python`, qui contiennent la position du front.
- Le paramètre `propagationModel` permet à l'utilisateur de choisir le modèle de vitesse

de propagation de front, suivant le cas que l'utilisateur désire simuler (coulée de lave, feu de forêt, etc). Dans cet exemple, le modèle de propagation de Balbi Unsteady est utilisé. Cependant, le paramètre `"nomdumodèle".speed`, s'il est présent, permet de forcer la valeur de la vitesse du front actif.

- La valeur du paramètre `perimeterResolution` fixe la distance maximale (en mètres) séparant deux nœuds constituant les points du front actif. Durant la simulation, chaque nœud évolue dans le domaine et, si la distance entre deux nœuds dépasse cette distance maximale, un nœud est créé pour ainsi réduire cette distance et garder une bonne résolution du front. Le paramètre `spatialIncrement` permet quant à lui de fixer la distance minimale (en mètres) que doit parcourir le front en un pas de temps pour être considéré comme actif. D'autres paramètres, tels que la puissance rayonnée par le front (en $W.m^{-2}$), permettent de caractériser l'aspect actif ou non du front. Le rapport entre le paramètre `perimeterResolution` et le paramètre `spatialIncrement` constitue la condition de stabilité du modèle de propagation ForeFire.
- L'accesseur `FireDomain[]` permet de créer un domaine de simulation rectangulaire dans un repère cartésien. L'utilisateur doit préciser sa taille *via* les coordonnées (en mètres) du coin situé au Sud-Ouest du domaine et les coordonnées du coin situé au Nord-Est du domaine (coordonnées dans un espace à trois dimensions). L'utilisateur peut également préciser à quel instant correspond la création de ce domaine de simulation (ici, le domaine est créé à l'instant $t = 0s$).
- L'accesseur `FireFront[]` permet ensuite à l'utilisateur d'initialiser la position d'un front actif à un instant donné (ici à 15 secondes). L'accesseur associé `FireNode[]` permet à l'utilisateur d'indiquer la position et la vitesse de chaque nœud constituant ce front initial.
- La commande `step[]` permet à l'utilisateur d'avancer la simulation au temps souhaité (ici, 150 secondes à partir de $t = 0$).
- Le fichier de sortie `NetCDF` généré par la commande `save[]` contient le temps d'arrivée du front pour chacune des mailles de la grille de discrétisation, identique à la grille de discrétisation donnée au fichier `NetCDF` de mise en donnée du modèle.

1.2.2 Mise en donnée de la topographie, du vent et de la végétation

Le fichier de configuration `FireSimulation.ff` fait appel au fichier `NetCDF "canyon.nc"` dont le nom est repéré par la paramètre `fluxNetCDF` pour la mise en donnée de la topographie, du vent, des modèles de végétation, des modèles de flux de chaleur et de vapeur utilisés. Ce fichier permet d'affecter ces valeurs de données sur une grille de discrétisation du domaine de simulation en $n_x \times n_y \times n_z$ cellules. Cette grille de discrétisation peut être choisie de sorte à correspondre à la grille de discrétisation d'un autre modèle couplé à ForeFire (modèle de vent comme Méso-NH, de pollution atmosphérique, etc). Cette mise en donnée n'est donc pas en lien avec la résolution spatiale du modèle ForeFire, mais sert à forcer le comportement de ForeFire en chacune des mailles de la grille de discrétisation. Pour affecter la valeur de tous les paramètres de fuel en chacune des mailles de la grille de discrétisation, un fichier `Python` appelé ici `"fuels.ff"`, permet de découper notre domaine en plusieurs sous-domaines correspondant à des types de végétation différents (végétation dense, routes, etc.) ; chaque sous-domaine est caractérisé par un jeu de paramètres et est repéré par un indice (3 domaines différents repérés par les indices 1, 2 et 3 sont illustrés en Fig. **??**). La valeur de cet indice est ensuite affectée aux cellules de discrétisation selon les types de fuels que comporte le domaine de simulation. Le paramètre `fuelsTableFile` dans le fichier de configuration `FireSimulation.ff` permet à l'utilisateur de spécifier le nom du fichier `Python` associant les indices aux valeurs des paramètres de fuels. Le fichier `"fuels.ff"` est comme suit pour l'exemple traité dans ce paragraphe :

```
Index;Rhod;Rhol;Md;Ml;sd;sl;e;Sigmad;Sigmal;stoch;RhoA;Ta;Tau0;Deltah;DeltaH;Cp;Cpa;Ti;X0;
    r00;Blai
1;650.0;120.0;0.05;1.0;7500.0;4766.0;2.4;0.892;1.793;8.3;1.0;310.0;70000.0;2300000.0;
    1.964E7;1912.0;1000.0;500.0;0.3;2.5E-5;2.0
2;720.0;120.0;0.01;1.0;5544.0;4766.0;2.4;0.892;1.793;8.3;1.0;300.0;70000.0;2300000.0;
    1.964E7;1912.0;1000.0;600.0;0.3;2.5E-5;2.0
3;0.0;0.0;0.0;0.0;0.0;0.0;0.0;0.0;0.0;0.0;0.0;0.0;0.0;0.0;0.0;0.0;0.0;0.0;0.0;0.0;0.0
```

En guise d'exemple, nous pouvons nous focaliser pour l'indice 2 sur l'humidité de la végétation (Md) et son rapport surface/volume (sd) : ici l'humidité très faible (1%) va contribuer à une propagation du front très rapide. La grandeur du rapport surface/volume (allant de quelques centaines de m^{-1} pour des chênes centenaires à $10000m^{-1}$ pour un combustible de type "aiguilles de pin") va être aussi un moteur de la propagation du front.

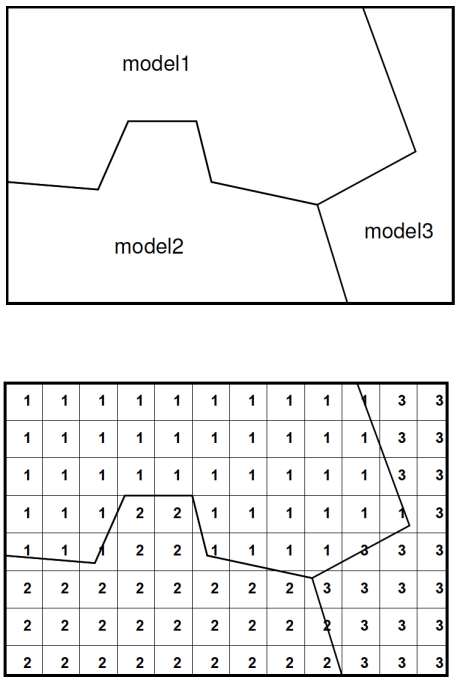


FIGURE 8 – Indexation des zones de végétation suivant le domaine

Partie 2

Étude de sensibilité dans ForeFire

2.1 Aspects techniques pour les simulations d'ensemble avec le coupleur OpenPALM et l'utilitaire PALM_PARASOL

Afin de réaliser l'EnKF sur ForeFire on utilise le coupleur de code OpenPALM qui avait servi à relier FireFly aux fonctions qui réalisaient les différentes étapes de calcul de l'EnKF. Pour faire l'étude de sensibilité et pour implémenter l'EnKF il on a besoin de lancer plusieurs instances du simulateur ForeFire, pour cela on utilise PALM_PARASOL qui est un utilitaire d'OpenPALM.

2.1.1 Le coupleur de codes OpenPALM

Le coupleur OpenPALM est développé au CERFACS depuis 1998. C'est un logiciel libre qui depuis janvier 2011 est co-développé par le CERFACS et l'ONERA. Il permet de définir des algorithmes complexes autour de codes à coupler et faire du parallélisme de tâches. Il supporte divers langages tels que le `Fortran90`, le `C` ou encore le `C++`.

La prise en main du logiciel est grandement facilitée par l'interface graphique PrePALM (Un exemple est illustré sur la figure 9) associée au coupleur servant à modéliser les différentes routines des codes à coupler sous forme de "boîtes" que nous appelons des *unités OpenPALM*. Ces unités sont insérées dans des branches et elles sont exécutées en descendant le long de ces branches.

Après avoir indiqué dans le code de l'unité OpenPALM les variables en entrées et en sorties, des plots (en vert sur la Fig. 9) sont affichées en haut de ces unités pour les entées et en bas pour les sorties. Il suffit ensuite de relier ces plots sous PrePALM pour effectuer les communications. OpenPALM s'appuie sur la technologie MPI (Multiple Protocol Interface) pour le lancement de processus et l'échange de données entre les unités OpenPALM. Dans l'exemple ci-dessous, unit_3 est appelée après unit_1. Unit_2 a besoin d'une sortie de unit_1, elle attend si besoin que unit_1 ait envoyé la variable.

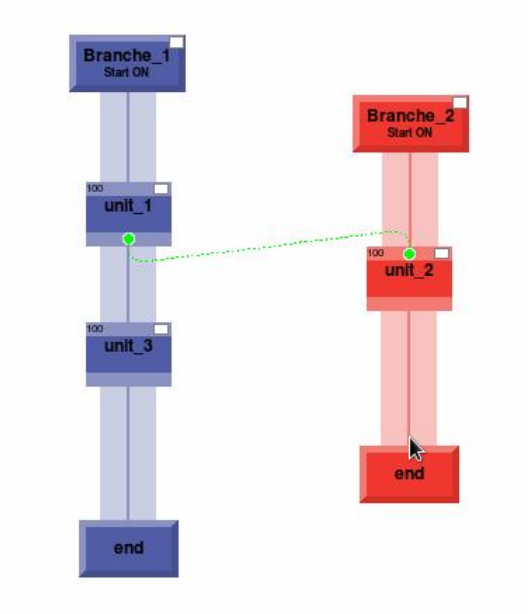


FIGURE 9 – Une unité OpenPALM *unit_1* qui envoie une variable à *unit_2*

L'adaptation de ForeFire pour OpenPALM a été réalisée par Clément Doche ([Doc12]). Initialement ForeFire fonctionnait avec un interpréteur `Python`. Clément Doche a intégré la fonction *main* de ForeFire, renommée *forefire_palm*, dans une unité OpenPALM `unitForeFire.cpp`. À cette étape on a donc le canevas suivant :

FIGURE 10 – Canevas de ForeFire sous Open_PALM

La fonction *forefire_palm* récupère à travers le fichier `main_forefire.args` le chemin du fichier de configuration `FireSimulation.ff` qui contient les commandes `Python` que l'on souhaite effectuer.

2.1.2 Lancement ensembliste avec PALM_PARASOL

PALM_PARASOL est un utilitaire d'OpenPALM (écrit en langage tcl) permettant de lancer automatiquement en parallèle un nombre n quelconque d'instances d'un même code de calcul. Son but n'est pas de paralléliser automatiquement un code de calcul via une décomposition de domaine, son intérêt est plutôt d'exécuter n instances d'un même code avec des entrées (et fatalement des sorties) différentes. PALM_PARASOL est une fonctionnalité d'OpenPALM dans la mesure où il lui permet d'encapsuler du code source dans des unités OpenPALM nécessaires au lancement de la simulation d'ensemble. Le but de PALM_PARASOL consiste donc à faire

travailler différentes instances de code avec différents arguments et de collecter les résultats dans le bon ordre. Pour équilibrer la répartition des calculs, OpenPALM nous permet de définir le nombre de processeurs alloués à la simulation suivant les capacités de notre machine. Notons que le code de calcul à "PARASOLiser" ne voit jamais la taille n du problème, c'est un paramètre de l'application couplée mais non du code : PALM_PARASOL est donc très peu intrusif dans le code, ce qui est très commode.

PALM_PARASOL crée une subroutine *slave* et le `makefile` correspondant ; c'est cette subroutine qui appelle la fonction à PARASOLiser. PALM_PARASOL crée une unité OpenPALM appelée *master*_nomDeFonction qui appelle n instances de forfor : L'unité *master* récupère l'ensemble des paramètres d'entrée (*cf.* Fig. 11). Ensuite, pour chaque élément de l'ensemble d'entrée, l'unité *master* appelle une instance de la subroutine *slave* en utilisant les processeurs esclaves qu'on lui alloue. La subroutine *slave* fait un appel à la fonction à "PARASOLiser", puis retourne les sorties vers le *master* qui se charge de formater l'ensemble des sorties pour une utilisation ultérieure.

Dans le cadre du projet, l'unité à "PARASOLiser" est la fonction *main* de ForeFire qui est codée en `C++` (elle avait été renommée *forefire_palm* dans le fichier `unitForeFire.cpp`). Un problème rencontré avec PALM_PARASOL a été le fait qu'il soit développé pour communiquer avec des subroutines `Fortran`. Pour parer à cela, il est nécessaire d'inclure ForeFire dans une subroutine `Fortran` (que l'on appelle ici *forfor* pour **For**eFire **for**tran). On fait alors appel à ForeFire depuis *forfor* comme ceci :

forfor.f90

```
1  !Appel du main de forefire (code dans unitForeFire.cpp)
2  call forefire_palm(i,md(1))
```

Ainsi on fait communiquer un langage avec l'autre. Mais la gestion des noms de fonctions des langages `C++` et `Fortran` étant différente, on doit utiliser la fonction OpenPALM `f2c_name` au niveau de la déclaration de la fonction, pour réaliser la traduction :

unitForeFire.cpp

```
1 extern "C" int f2c_name(forefire_palm)(int *id_i, double *md)
```

Les arguments de `forefire_palm` ne contiennent non plus le chemin du fichier de configuration mais l'indice de l'instance PALM_PARASOL (noté i) et la valeur d'un paramètre (ici l'humidité Md mais on peut facilement choisir un autre paramètre) à faire varier. Pour chaque instance nous créons alors un fichier `FireSimulation$i.ff` et un fichier `fuels$i.ff` *via* un *ScriptShell*. Ce *ScriptShell* crée aussi un dossier de sortie pour chaque instance. Le fichier FireSimulation$i.ff qu'on a produit contient le nouveau chemin du dossier de sortie et le chemin du fichier `fuels$i.ff`. Ce fichier contient la valeur de Md pour l'instance i.

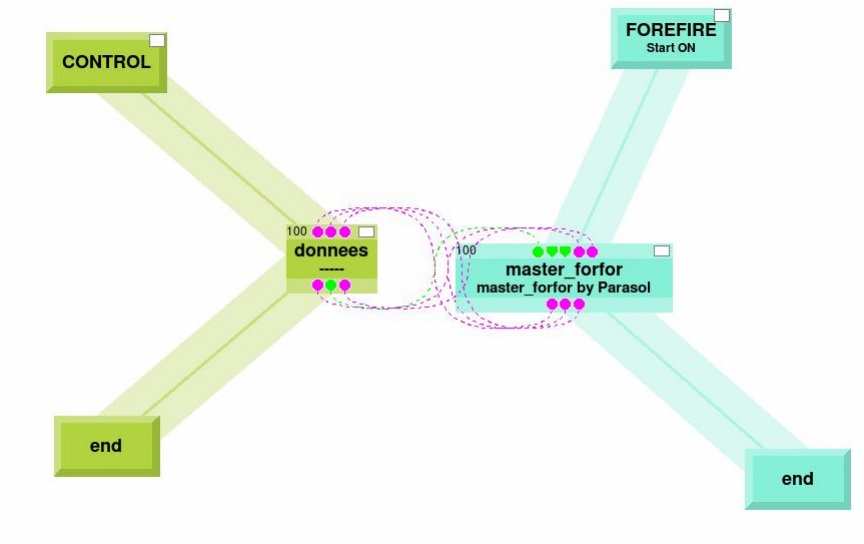


FIGURE 11 – L'unité *donnees* envoie un ensemble de paramètres à l'unité *master_forfor* créée par PALM_PARASOL. (Les retours de *master_forfor* vers *donnees* sont dûs à la façon dont est conçu PALM_PARASOL).

La Fig. 12 illustre le fonctionnement de l'unité *Master_forfor* créée par le script `parasol.tcl` : Celle-ci reçoit l'ensemble des vecteurs de contrôle et elle retourne l'ensemble des sorties des simulations qui ont chacune tourné avec un des vecteurs de contrôle. Sur la Fig. 12, ces vecteurs n'ont qu'un seul élément Md et les vecteurs de contrôle sont composés des positions x, y et z d'un ensemble de p points situés le long du front de flamme.

Finalement, ForeFire est lancé par *forefire_palm* qui est une fonction codée dans *unit_ForeFire.cpp*. Elle sert à ne pas utiliser le *shell* en mode interactif et à envoyer les bons paramètres en entrée. Ensuite, *forfor* appelle *forefire_palm* pour passer du `C++` au `Fortran`. Puis grâce à PALM_PARASOL, on rajoute les couches *slave_forfor* et *master_forfor* pour pouvoir lancer n simulations. Enfin, cette dernière unité est intégrée dans l'algorithme d'assimilation existant grâce à OpenPALM.

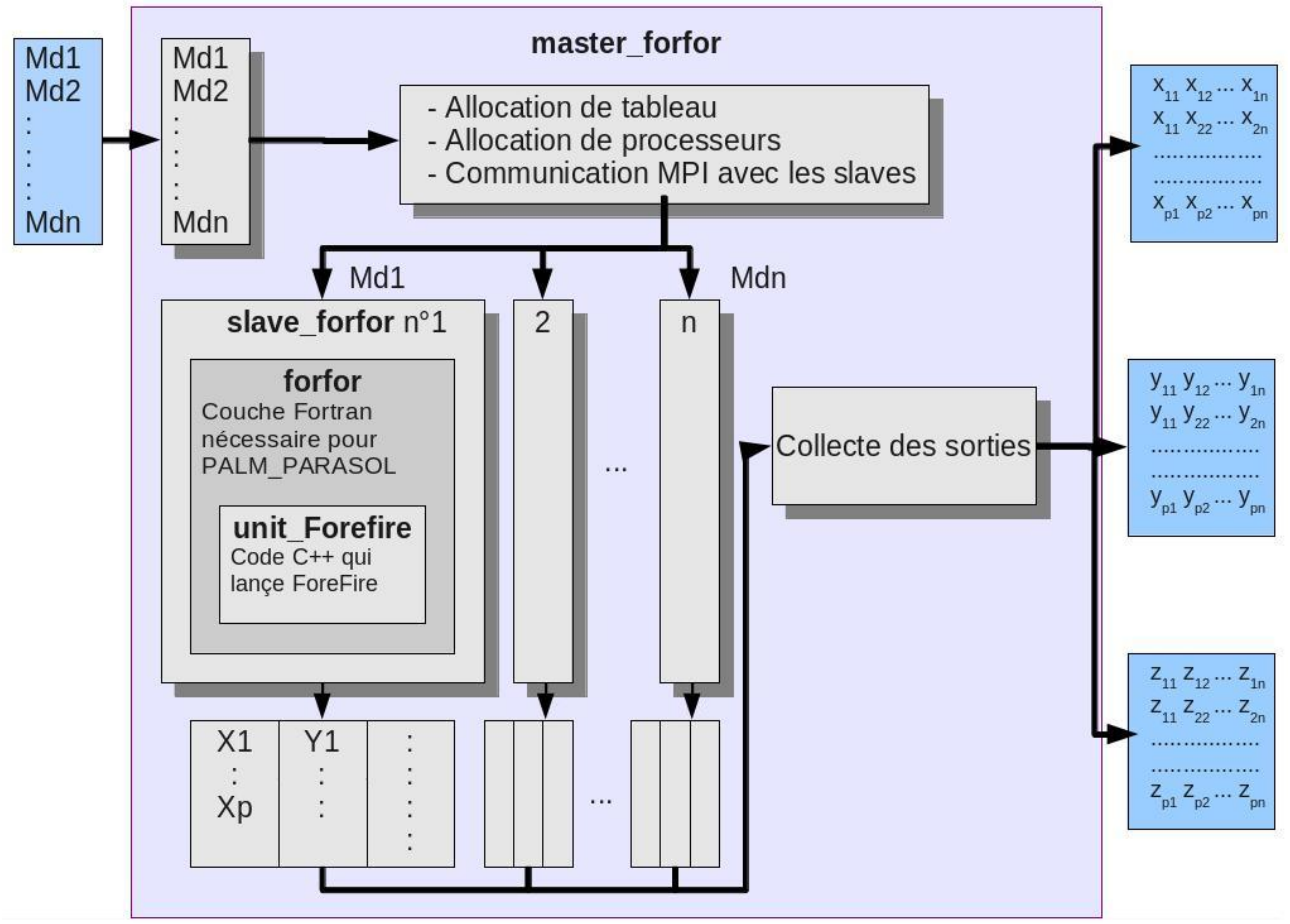


FIGURE 12 – Fonctionnement de l'unité OpenPALM *master*

2.2 Étude qualitative de la sensibilité des paramètres

Une étude de sensibilité sur le modèle ForeFire consiste à identifier les paramètres pour lesquels la position du front de feu est sensible. Ces paramètres seront ensuite importants à assimiler pour corriger cette position de front vers les observations. C'est une étape indispensable sans quoi nous pourrions corriger des paramètres qui n'auraient qu'une importance minime sur la propagation du front. Disposant déjà des conclusions de l'étude de sensibilité réalisée pour FireFly avec le modèle de Rothermel ([Roc10]), il s'agit de confirmer ici que les paramètres sensibles pour Rothermel (Md,sd) sont aussi sensibles pour le modèle de Balbi et aussi que ForeFire se comporte comme attendu. La méthode est donc simplement de faire varier les paramètres jugés pertinents un à un et de visualiser au dernier pas de temps de la simulation les traces du front de feu associées. Plusieurs simulations ont été réalisées avec des valeurs de paramètres générées par une loi uniforme, chaque simulation prenant une valeur différente est appelée ici un *membre*. Informatiquement, cette étude est commandée par l'utilitaire PALM_PARASOL. Les détails de l'implémentation sont données en section 2.1.2.

Notons que toutes les simulations ont été faites en prenant le modèle de vitesse de Balbi sur un domaine totalement plat de $640m \times 320m$ avec un vent uniforme et venant du Sud.

La première étude fait varier l'humidité de la végétation Md de 5 à 50% avec 10 membres pour un vent de secteur Sud de $1m.s^{-1}$ et un rapport surface/volume des particules de végétation $sd = 5544m^{-1}$. La Fig. 13 montre que le front progresse plus rapidement si la végétation est plus sèche. On remarque que la propagation a tendance à s'effectuer plus fortement dans l'axe du vent. Il n'y a pas de propagation contre le vent.

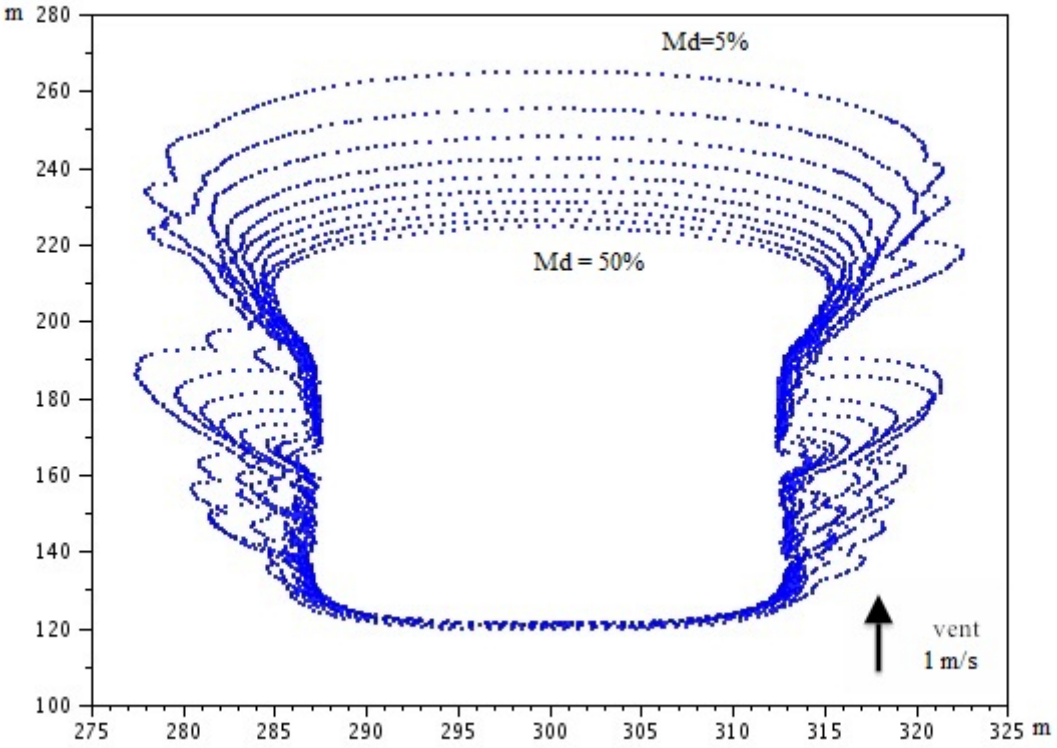


FIGURE 13 – Position des fronts simulés à $t = 1800s$ avec ForeFire pour 10 valeurs d'humidité différentes.

La deuxième étude consiste à étudier l'influence du rapport surface/volume sd variant de 1000 à $10000m^{-1}$ avec 10 membres toujours avec le même vent et $Md = 15\%$. La Fig. 14 montre que la propagation est plus importante si le rapport surface/volume est plus grand. De façon réelle, cela signifie que le front de feu se déplace surtout par la combustion des fines particules de végétation telles que des brindilles : le délai d'inflammation est plus court. Remarquons aussi que le rapport surface/volume est une composante beaucoup plus forte que la paramètre d'humidité. En effet, dans ce cas, le front a tendance à aussi se propager dans des directions normales et même (dans une moindre mesure) opposées à la direction du vent.

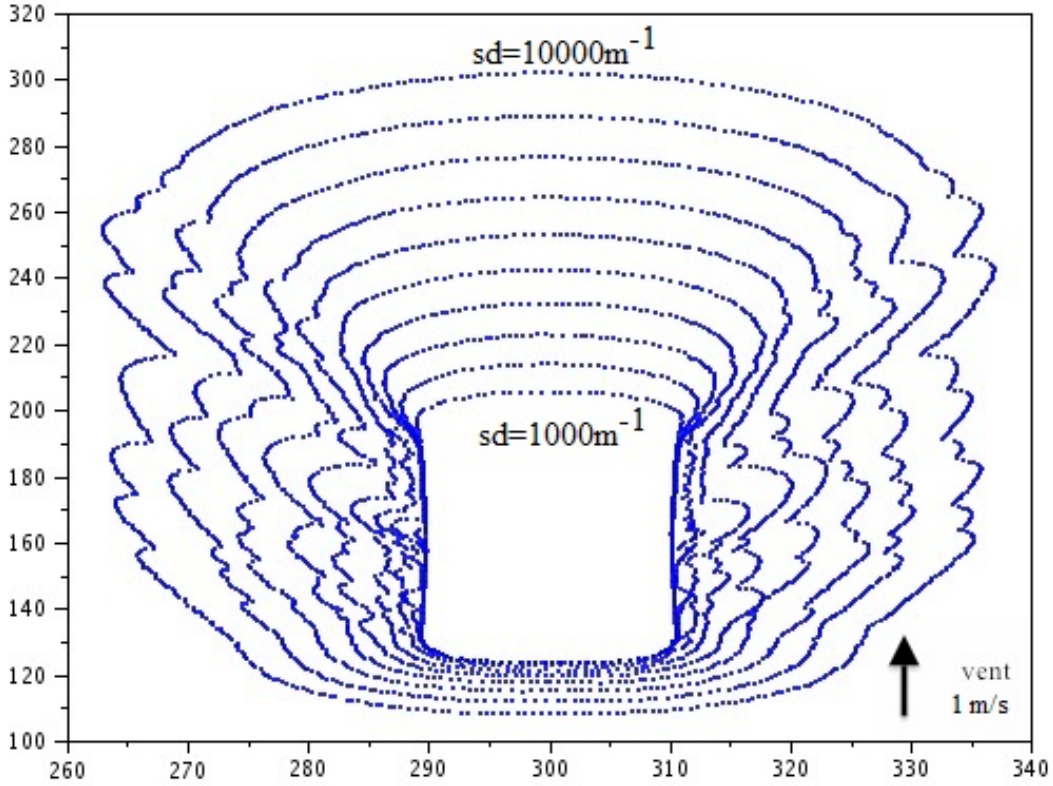


FIGURE 14 – Position des fronts simulés à $t = 1800s$ avec ForeFire pour 10 valeurs de rapport surface/volume différentes.

La troisième étude fait varier la vitesse du vent de 0.5 à $2m.s^{-1}$ avec 4 membres, celui-ci reste de secteur Sud avec $Md = 15\%$ et $sd = 5544m^{-1}$. La Fig. 15 démontre bien le rôle prépondérant du vent dans la propagation d'un front. De plus, nous pouvons remarquer que la variation du vent semble proportionnelle avec l'écart entre les différents fronts : ceci corrobore la Fig. 16 tirée de [Gor12] montrant la dépendance linéaire du vent sur les modèles d'estimation de *ROS*.

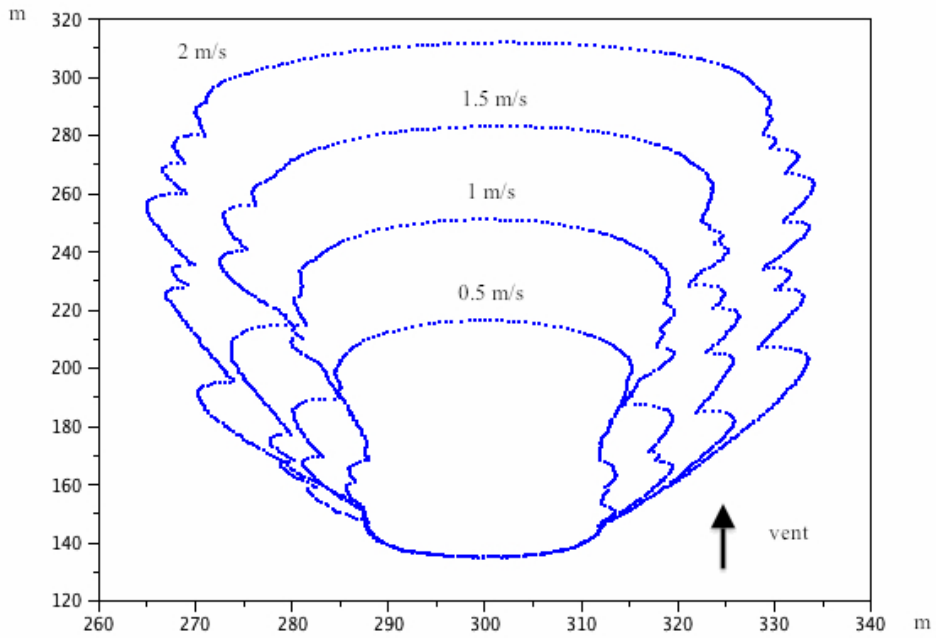


FIGURE 15 – Position des fronts simulés à $t = 1650s$ avec ForeFire pour 4 vitesses de vent différentes.

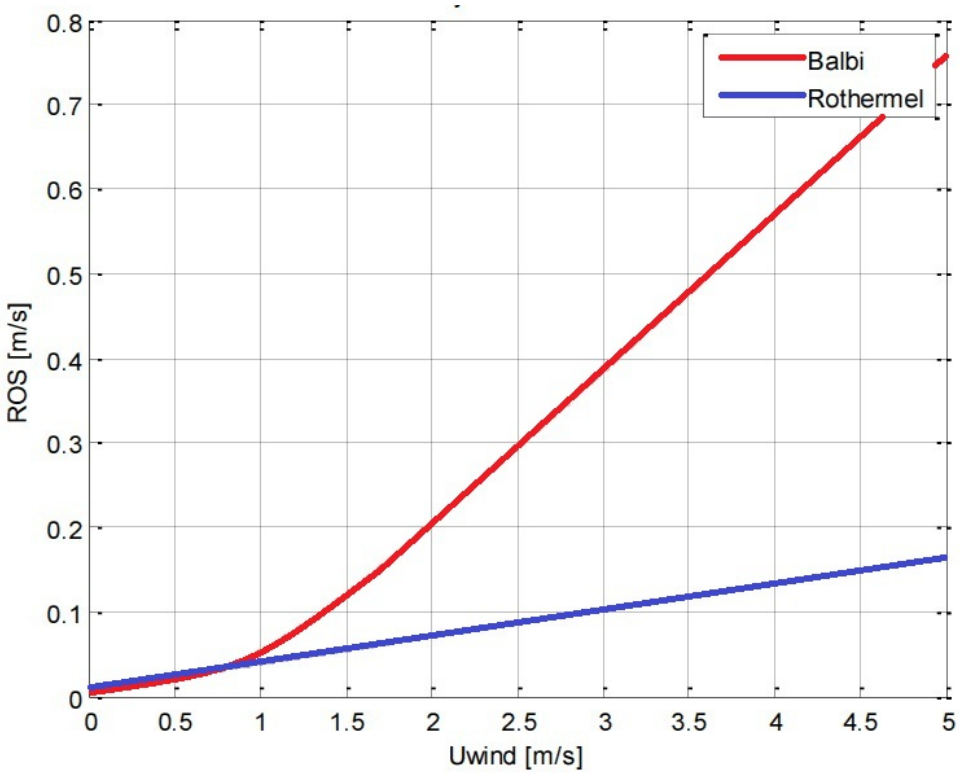


FIGURE 16 – Comparaison Rothermel/Balbi et comportement linéaire du *ROS* avec le vent ([Gor12]).

Partie 3

L'assimilation de données dans ForeFire

3.1 Prérequis sur l'assimilation de données

3.1.1 Principe et applications

La modélisation des feux de forêt est un exemple typique de problème physique pour lequel la méconnaissance de l'état vrai de l'environnement limite la qualité des prévisions. La prévisibilité d'une situation est ainsi dépendante des incertitudes liées au modèle décrivant les processus mis en jeu, rarement parfaits, et aux observations permettant de décrire l'état initial. Partant d'une ébauche (estimation *a priori* de l'état du système dynamique étudié) fournie par un modèle, les techniques d'assimilation de données permettent de tenir compte des informations supplémentaires déduites d'observations pour aboutir à un état analysé, ou analyse, plus proche de l'état vrai que ne l'est l'ébauche. Cette analyse peut ainsi servir d'état initial pour une prévision du système à l'aide du modèle, ou de référence afin de vérifier la qualité des observations.

L'assimilation de données est mise en oeuvre dès que l'on dispose d'observations fiables. Les applications sont donc nombreuses. Parmi celles-ci, l'hydrologie, la prévision de la trajectoire des ouragans, la détermination d'orbites satellitales, les systèmes de guidage, le traitement d'images, les sciences de la terre ou l'économétrie. En ce qui concerne la surveillance des feux de forêt, l'objectif de cette technique est de corriger les paramètres d'entrée du modèle utilisé, tels que l'humidité de la végétation, le rapport surface/volume des particules de combustible dans le but d'améliorer la prévision de l'évolution du front de flammes.

3.1.2 Variables d'assimilation

Grâce au développement des simulations numériques, il est possible de tester la validité et la précision d'un modèle physique en s'appuyant sur les informations fournies par des observations. Les variables auxquelles le modèle est sensible et sur lesquelles les incertitudes sont les plus élevées sont les plus légitimes à bénéficier de la correction apportée par les méthodes d'assimilation de données. Avant d'aborder la stratégie visant à minimiser les incertitudes sur ces variables d'intérêt, il convient de définir un certain nombre d'objets mathématiques parmi lesquels le vecteur de contrôle, le vecteur d'observations et l'opérateur d'observation.

On appelle vecteur de contrôle $\mathbf{x}$, le vecteur formé par un nombre n de variables à corriger (qui peuvent être scalaires ou vectoriels). Dans le cas de l'application à la surveillance des feux de forêt, il sera constitué des paramètres d'entrée du modèle de propagation (de Rothermel ou Balbi), nommés plus loin paramètres de contrôle, dans notre cas la teneur en eau de la

végétation (Md) et le rapport surface/volume des particules de végétation (Sd). Pour aboutir aux valeurs optimales des paramètres de contrôle, on dispose de valeurs a priori contenues dans le vecteur d'ébauche $\mathbf{x}_b$ (background) de taille n, et du vecteur d'observations $\mathbf{y}_0$, de taille p. Le vecteur des paramètres corrigés, noté $\mathbf{x}_a$ (analysis), correspond à l'analyse. L'état vrai du système dynamique est représenté par $\mathbf{x}_t$, qui contient les vraies valeurs des paramètres que l'on cherche à approcher et non parfaitement connues. Précisons que le vecteur de contrôle et le vecteur d'observations ne sont pas nécessairement définis dans le même espace, en particulier ici dans le cas du problème de propagation d'incendie. En effet, le choix des variables observées dépend des outils fournissant les mesures, tandis que les paramètres de contrôle sont choisis parmi les paramètres les plus sensibles du modèle de vitesse de propagation. De plus, dans certains cas, les observations peuvent être disposées très irrégulièrement. Il faut donc définir une fonction permettant de passer de l'espace des paramètres à celui des observations. Cette fonction, nommée opérateur d'observation , permet d'obtenir un équivalent du vecteur de contrôle dans l'espace des observations. Dans le cadre de ce projet, cet opérateur inclut le modèle de propagation utilisé (Rothermel ou Balbi) dans le simulateur ForeFire et permet ainsi de convertir le vecteur de contrôle en une position du front de flammes. Mais il peut aussi être construit sur la base d'opérateurs d'interpolation et d'autres opérateurs permettant de transformer les variables du modèle en paramètres observés. Par exemple, en météorologie, il peut transformer les températures des différents niveaux de pression en une radiance mesurée par les satellites. Pour la simulation de données appliquée au simulateur eulérien FireFly, l'opérateur H a été construit par composition de deux opérateurs, l'intégration du modèle M (qui permet de passer des variables d'entrée du modèle à un isocontour de la variable de progrès c) et la sélection des positions observées le long de cet isocontour S. Cette sélection est nécessaire pour pouvoir aisément comparer les fronts simulés au front observé au temps d'assimilation, autrement dit pour calculer la distance front à front, ce qu'on appelle communément le vecteur d'innovation :

$$\mathcal{H} = \mathcal{S}o\mathcal{M} \tag{3.1}$$

Dans le cas de ForeFire, rappelons que la trace du front de flammes est repérée par des noeuds (ou points) disposés suivant un maillage dynamique. C'est-à-dire que des points peuvent être enlevés ou ajoutés selon la distance les séparant deux à deux. Ce maillage est implémenté dans le code source FireNode.cpp qui définit la classe noeud (node en anglais). Afin de pouvoir suivre l'évolution de la position des points présents depuis l'instant initial, nous avons dû désactiver l'option merge définie dans ce code source qui supprime des points trop proches les uns des autres. Un identifiant étant affecté à chaque noeud du début à la fin de la simulation, nous nous sommes appuyés sur celui-ci pour pouvoir sélectionner les points présents lors de l'instant initial et ainsi suivre à chaque temps d'observation l'évolution du même jeu de points.

3.1.3 Modélisation des erreurs : fiabilité du modèle et des observations

Pour prendre en compte les incertitudes dans l'ébauche, les observations et l'analyse, il faut faire des hypothèses sur la modélisation des erreurs entre ces vecteurs et leurs équivalents vrais. L'utilisation des fonctions de densité de probabilité, nommées *PDF*, est une approche adaptée pour construire des modèles d'erreur. Les vecteurs d'erreurs sont supposés se comporter comme des variables aléatoires.

Il est possible de définir les erreurs d'ébauche, d'observation et d'analyse telles que présentées dans le tableau suivant :

Nom	Définition	Covariances
Erreur d'ébauche	$\epsilon^b = \mathbf{x}^b - \mathbf{x}^t$	$\mathbf{B} = \mathbb{E}((\epsilon^b - \bar{\epsilon}^b)(\epsilon^b - \bar{\epsilon}^b)^T)$
Erreur d'observation	$\epsilon^o = \mathbf{Y}^o - \mathbf{H}\mathbf{x}^t$	$\mathbf{R} = \mathbb{E}((\epsilon^o - \bar{\epsilon}^o)(\epsilon^o - \bar{\epsilon}^o)^T)$
Erreur d'analyse	$\epsilon^a = \mathbf{x}^a - \mathbf{x}^t$	$\mathbf{A} = \mathbb{E}((\epsilon^a - \bar{\epsilon}^a)(\epsilon^a - \bar{\epsilon}^a)^T)$

On note généralement **B** la matrice de covariance de l'ébauche, **A** celle de l'analyse et **R** celle des observations.

La qualité de la méthode d'assimilation est d'autant plus grande que l'erreur d'analyse est faible. Si c'est le cas, en accumulant les informations issues des observations et de l'estimation fournie par le modèle, l'assimilation de données aura permis d'aboutir à un meilleur diagnostique de l'état du système que les observations et le modèle numérique considérés séparément.

La moyenne de ces différentes erreurs sera appelée le biais et représente un problème systématique dans le système d'assimilation qui peut être une dérive du modèle, un biais dans les observations ou encore une erreur systématique dans la manière d'utiliser les observations. On suppose par la suite qu'ébauche et observations sont non biaisées.

On choisit de faire l'hypothèse forte que la loi suivie par les erreurs est de type gaussien. Celle-ci possède la propriété intéressante d'être entièrement décrite par seulement deux moments (moyenne et variance). Elle est de plus cohérente avec celle qui considère les erreurs comme des variables aléatoires.

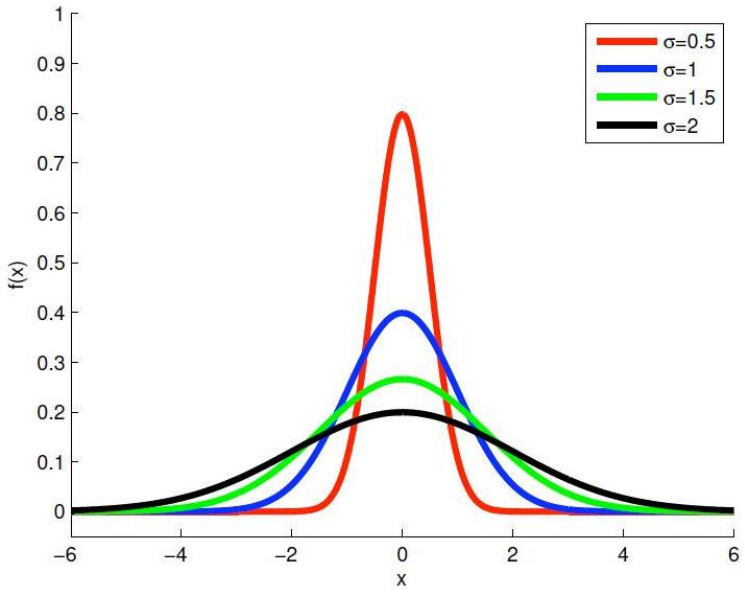


FIGURE 17 – Diverses lois gaussiennes centrées. (*J.M. Bart*)

Dans le cas de variables aléatoires non corrélées entre elles (ni spatialement ni temporellement) suivant une loi gaussienne, les matrices de variances/covariances sont diagonales, et chacun des éléments représente la variance d'erreurs des variables de contrôle. On note généralement B la matrice de covariance de l'ébauche, A celle de l'analyse et R celle des observations.

Il existe deux stratégies principales qui permettent de calculer les moyennes et variances d'erreurs pour ainsi construire les matrices citées plus haut qui interviennent dans les algorithmes d'assimilation :

- La méthode BLUE (Best Linear Unbiaised Estimator)
- La méthode du Filtre de Kalman d'Ensemble

3.2 Méthode du Filtre de Kalman : itérations temporelles de l'assimilation de données

Le but de notre travail étant d'implémenter une méthode d'assimilation pour améliorer la prévision de la propagation d'un feu de forêt, il convient d'exposer le principe d'une méthode qui permet d'apporter les corrections sur les paramètres de contrôle dans le temps. C'est ce qui est rendu possible par l'algorithme du Filtre de Kalman.

Supposons que l'on dispose d'un jeu de données pour plusieurs temps d'observations :

$$\{\mathbf{Y}_1^0, \mathbf{Y}_2^0, ..., \mathbf{Y}_{N_{obs}}^0\} \tag{3.2}$$

Sur le schéma suivant, est proposée une description des deux premiers cycles d'assimilation. Après prise en compte des observations disponibles au premier instant $\mathbf{Y}_1^0$, une analyse est produite $\mathbf{X}_1^a$. Celle-ci est ensuite intégrée par le modèle jusqu'au deuxième temps d'observation pour former $\mathbf{X}_2^f$. Ce vecteur est ensuite corriger par les observations $\mathbf{Y}_2^0$ pour produire $\mathbf{X}_2^a$ et ainsi de suite.

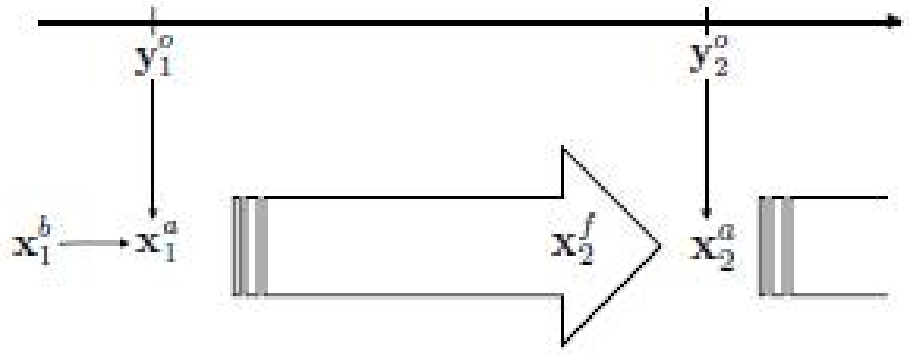


FIGURE 18 – Représentation schématique des deux premières itérations de l'algorithme du Filtre de Kalman (*J.M. Bart*)

Il faut introduire l'opérateur traduisant la propagation par le modèle du vecteur de contrôle à tous les instants d'observations, noté :

$$\mathbf{X}_{i+1}^f = \mathcal{M}_{i \to i+1}(\mathbf{X}_i^a) \tag{3.3}$$

Avant de présenter l'algorithme généralisé à tous les temps d'observations, rappelons un certain nombre d'hypothèses sur lesquelles repose la méthode.

1.L'erreur du modèle est non biaisée

2.Les erreurs d'analyse et du modèle sont non corrélées

3.L'opérateur du modèle de prévision doit être linéaire

On notera $\mathbf{M}_{i-1 \to i}$ le linéaire tangent de l'opération d'intégration du modèle.

$$\mathbf{X}_i^f = \mathcal{M}_{i-1 \to i}(\mathbf{X}_{i-1}^a) \approx \mathbf{M}_{i-1 \to i}(\mathbf{X}_{i-1}^a) \tag{3.4}$$

La matrice de covariance de prévision vérifie l'équation ci dessous :

$$\mathbf{P}_i^f = \mathbf{M}_{i-1 \to i} \mathbf{P}_{i-1}^a \mathbf{M}_{i-1 \to i}^T + \mathbf{Q}_{i-1} \tag{3.5}$$

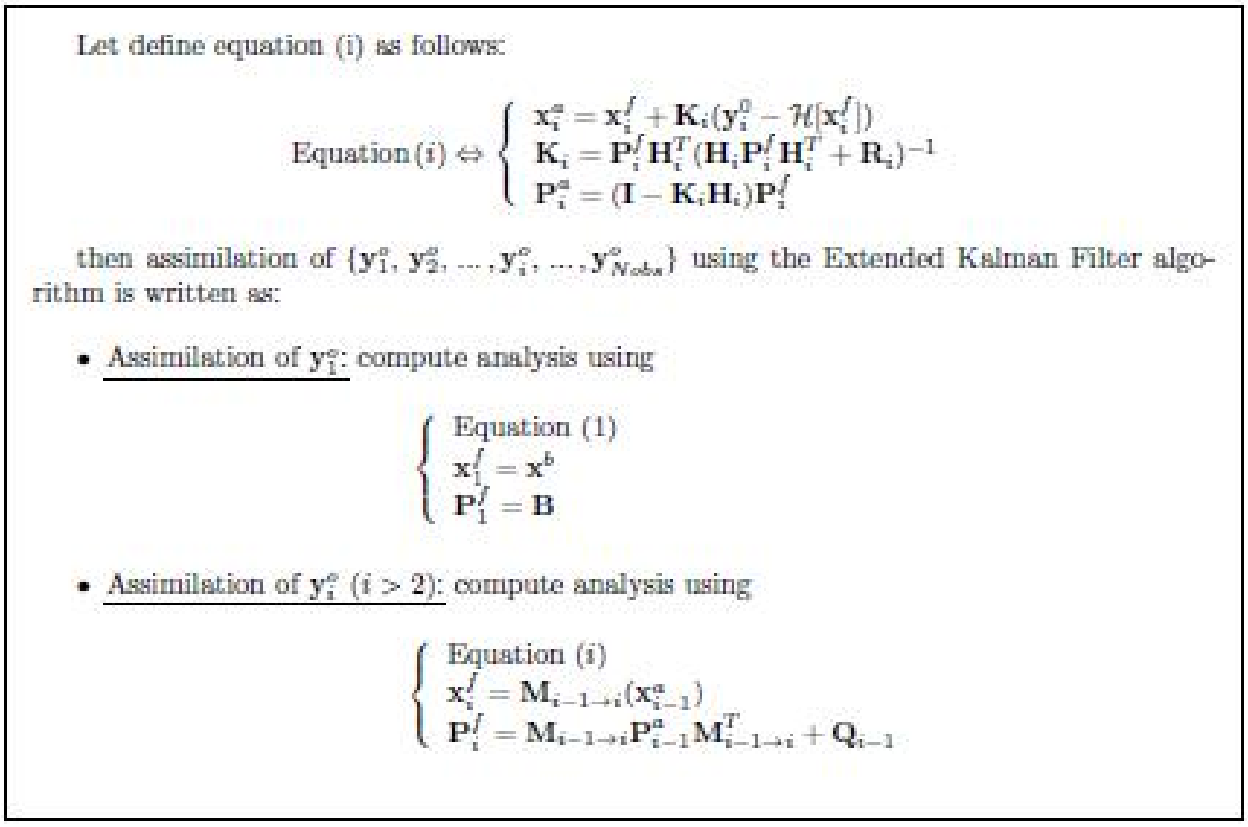


FIGURE 19 – Description de l'algorithme tenant compte des hypothèses citées et des approximations décrites ci dessous (*J.M. Bart*)

avec

$$\mathbf{Q}_{i-1} = (\mathcal{M}_{i-1\to i}(\mathbf{x}_{i-1}^t) - \mathbf{x}_i^t)(\mathcal{M}_{i-1\to i}(\mathbf{x}_{i-1}^t) - \mathbf{x}_i^t)^T \tag{3.6}$$

en utilisant :

$$\mathcal{M}_{i-1\to i}(\mathbf{x}_{i-1}^a) - \mathcal{M}_{i-1\to i}(\mathbf{x}_{i-1}^t) = \mathbf{M}_{i-1\to i}(\mathbf{x}_{i-1}^a - \mathbf{x}_{i-1}^t) \tag{3.7}$$

il vient :

$$\mathbf{x}_i^f = \mathbf{M}_{i-1\to i}(\mathbf{x}_{i-1}^a - \mathbf{x}_{i-1}^t) + \mathcal{M}_{i-1\to i}(\mathbf{x}_{i-1}^t) \tag{3.8}$$

puis :

$$\epsilon_i^f = \mathbf{M}_{i-1\to i}(\epsilon_{i-1}^a) + \mathcal{M}_{i-1\to i}(\mathbf{x}_{i-1}^t) - \mathbf{x}_i^t \tag{3.9}$$

où ϵ_i^f et ϵ_{i-1}^a sont les erreurs de prévision et d'analyse au temps de l'observation de $\mathbf{Y}_{N_{obs}}^0$ Par définition de la matrice de covariance d'erreurs on obtient finalement :

$$\mathbf{P}_i^f = \mathbf{M}_{i-1\to i}\mathbf{P}_{i-1}^a \mathbf{M}_{i-1\to i}^T + (\mathcal{M}_{i-1\to i}(\mathbf{x}_{i-1}^t) - \mathbf{x}_i^t)(\mathcal{M}_{i-1\to i}(\mathbf{x}_{i-1}^t) - \mathbf{x}_i^t)^T \tag{3.10}$$

La dernière des trois hypothèses n'étant pas respectée pour notre modèle de propagation des fronts de flammes, nous aurons recours à une autre méthode : le filtre de Kalman d'ensemble.

3.3 Méthode du Filtre de Kalman d'Ensemble

Comme dit précédemment, il est capital pour une méthode d'assimilation de données de pouvoir représenter les *PDF* des erreurs d'ébauche et d'observation. La méthode du Filtre de Kalman d'Ensemble (*EnKF*) réalise une discrétisation de ces *PDF* avant de les représenter statistiquement. L'avantage de cette méthode est qu'elle permet de décrire les *PDF* non pas à partir d'une seule erreur par paramètre contrôlé mais à partir d'une population d'erreurs, qui vont évoluer dans le temps. Son inconvénient principal en revanche repose sur son coût de calcul élevé car elle implique autant d'intégrations du modèle que de membres dans la population. Cependant, cet inconvénient peut être contrebalancé par le fait que les matrices de covariances n'ont plus à être stockées. C'est Evensen (1994) qui a été le premier à proposer cette stratégie.

3.3.1 Les équations de l'EnKF

On appelle $\mathbf{X}_e$ la matrice d'ensemble, contenant une population de N vecteurs de contrôle.

$$\mathbf{X}_e = (\mathbf{x}_{[1]}, \mathbf{x}_{[2]}, ..., \mathbf{x}_{[N]}) \tag{3.11}$$

avec

$$\mathbf{x}_{[i]} = \mathbf{x}^t + \epsilon_{[i]} \tag{3.12}$$

Les matrices d'ensemble de l'analyse et prévue seront natrellement notées $\mathbf{X}_e^a$ et $\mathbf{X}_e^f$. Les erreurs d'analyse, de prévision et d'observation seront notées $\epsilon_{[i]}^a$,$\epsilon_{[i]}^b$,$\epsilon_{[i]}^o$, avec i allant de 1 à N. Chaque vecteur de contrôle de la matrice d'ensemble (appelé membre) correspond à une série de valeurs des paramètres de contrôle (tirées du modèle pour $\mathbf{X}_e^f$ et corrigées par les observations pour $\mathbf{X}_e^a$), approchant le vecteur de contrôle "vrai" qui correspond le plus à la réalité.

Le Filtre de Kalman d'Ensemble est très populaire car il est conceptuellement très simple et sa mise en œuvre est aisée. En effet, il ne nécessite ni dérivation des opérateurs tangent-linéaires et des équations adjointes, ni intégration rétrograde du modèle d'évolution.Il repose les mêmes équations que le Filtre de Kalman appliqué aux vecteurs d'ensemble :

$$\mathbf{K}_e = \mathbf{P}_e^f \mathbf{H}^T (\mathbf{H}\mathbf{P}_e^f \mathbf{H}^T + \mathbf{R}_e)^{-1} \tag{3.13}$$

Avec $\mathbf{Y}_e^o$ la matrice d'observations d'ensemble :

$$\mathbf{Y}_e^0 = (\mathbf{Y}^0 + \epsilon_{[1]}^0, \mathbf{Y}^0 + \epsilon_{[2]}^0, ..., \mathbf{Y}^0 + \epsilon_{[N]}^0) \tag{3.14}$$

Tandis que $\mathbf{X}_e^f$ est la matrice d'ensemble prévue. Les erreurs de prévision $\epsilon_{[i]}^b$ et d'observation $\epsilon_{[i]}^o$ sont générées par la méthode de Monte Carlo et sont également supposées sans biais. On peut ainsi substituer le vecteur de contrôle "vrai" $\mathbf{x}^t$ par la moyenne des membres de $\mathbf{X}_e^f$ et définir respectivement les matrices de covariance d'erreur d'observation et de prévision :

$$\begin{aligned} \mathbf{P}_e^f &= \overline{(\mathbf{X}_e^f - \overline{\mathbf{X}_e^f})(\mathbf{X}_e^f - \overline{\mathbf{X}_e^f})^T} \\ \mathbf{R}_e &= \mathbb{E}((\epsilon^o - \bar{\epsilon}^o)(\epsilon^o - \bar{\epsilon}^o)^T) \\ \mathbf{P}_e^f &= \mathbb{E}((\mathbf{X}_e^f - \overline{\mathbf{X}_e^f})(\mathbf{X}_e^f - \overline{\mathbf{X}_e^f})^T) \end{aligned} \tag{3.15}$$

De plus, $\mathcal{H}(\mathbf{X}_e^f)$ est considéré comme une bonne approximation de $\mathcal{H}(\mathbf{x}^t)$. Pour cette raison, il n'est plus nécessaire de linéariser l'opérateur d'observation $\mathcal{H}$.

Il est donc possible d'évaluer avec une bonne précision la matrice de gain :

$$\mathbf{K}_e = \mathbf{P}_e^f \mathbf{H}^T (\mathbf{H}\mathbf{P}_e^f \mathbf{H}^T + \mathbf{R}_e)^{-1} \tag{3.16}$$

Avec :

$$\mathbf{P}_e^f\mathbf{H}^T = \frac{(\mathbf{X}_e^f - \overline{\mathbf{X}_e^f})(\mathcal{H}(\mathbf{X}_e^f) - \overline{\mathcal{H}(\mathbf{X}_e^f)})^T}{N-1} \tag{3.17}$$

et

$$\mathbf{HP}_e^f\mathbf{H}^T = \frac{(\mathcal{H}(\mathbf{X}_e^f) - \overline{\mathcal{H}(\mathbf{X}_e^f)})((\mathcal{H}(\mathbf{X}_e^f) - \overline{\mathcal{H}(\mathbf{X}_e^f)})^T}{N-1} \tag{3.18}$$

De même $\mathbf{P}_e^f\mathbf{H}^T$ et $\mathbf{HP}_e^f\mathbf{H}^T$ constituent de bonnes approximations de $\mathbf{P}^f\mathbf{H}^T$ et $\mathbf{HP}^f\mathbf{H}^T$ (opérateurs linéarisés) du Filtre de Kalman.

3.4 Adaptation de l' algorithme d'assimilation de données au modèle ForeFire

Dans cette partie, il sera expliqué comment nous avons fait communiquer le modèle de propagation ForeFire avec l'algorithme EnKF.

3.4.1 Vecteur de contrôle

Dans le modèle utilisé, la vitesse de propagation du front est calculée d'après une formule semi empirique (modèle de Balbi), mettant en évidence une dépendance aux conditions météorologiques et aux propriétés de la végétation. A l'issue de l'étude de sensibilité, des paramètres de contrôle pertinents semblent être l'humidité de la végétation Md (dry Moisture), le rapport surface/volume des végétaux Sd et le vent. Dans le cadre de ce projet, nous nous sommes particulièrement intéressés au paramètre Md. En effet, par soucis de faciliter l'implémentation de l'algorithme d'assimilation, nous avons choisi de valider la méthode sur un seul paramètre avant de pouvoir la généraliser à plusieurs. Dans ce cas, notre vecteur de contrôle est donc un scalaire : $\mathbf{x} = (\mathrm{Md})$.

3.4.2 Vecteur d'observation

On peut imaginer que les observations disponibles sont sous forme de coordonnées de points situés sur le front de flammes discrétisé. De telles observations pourront être fournies par des images satellites par exemple. C'est donc sous cette forme que sera artificiellement construit notre vecteur d'observation.

$$\mathbf{Y}^0 = \begin{pmatrix} x_1^{obs} \\ x_2^{obs} \\ \vdots \\ x_p^{obs} \\ y_1^{obs} \\ y_2^{obs} \\ \vdots \\ y_p^{obs} \end{pmatrix}$$

La dimension p de ce vecteur est $2N_p$ où N_p est le nombre de points sur le front.

Pour produire ce vecteur, nous effectuons dans un premier temps une simulation avec la valeur vraie $\mathbf{x}^t$ du paramètre de contrôle Md. La position du front ainsi obtenue est ensuite

perturbée grâce à des erreurs générées par une loi gaussienne, de moyenne nulle et de variance choisie (les erreurs sont ainsi totalement contrôlées). Il s'agit du procédé *OSSE : Observing System Simulation Experiments.*

3.4.3 Principe des expériences jumelles

L'objectif des expériences jumelles est de vérifier la qualité de la correction des ébauches permettant de produire les analyses. Pour ce faire, on dispose de deux jeux de paramètres : une référence, ou valeurs vraies, et une série perturbée (ébauche). On considère que la valeur vraie est parfaitement connue. C'est en estimant l'écart entre la valeur vraie et l'analyse que l'on évalue la qualité de la correction.

L'ébauche est obtenue en perturbant la valeur vraie. Les observations sont produites d'après le procédé OSSE décrit précédemment. Les erreurs ajoutées pour fournir l'ébauche et les observations doivent être cohérentes avec les hypothèses suivies par l'algorithme d'assimilation. Elles doivent donc être de moyenne nulle et de variances conformes à celles prescrites par les matrices de variances/covariances

Les expériences jumelles sont donc un outil privilégié pour valider un schéma d'assimilation et tester la sensibilité de l'algorithme à une perturbation donnée. La figure suivante en illustre le principe. Dans notre cas, nous nous sommes intéressés à la correction apportée par la méthode

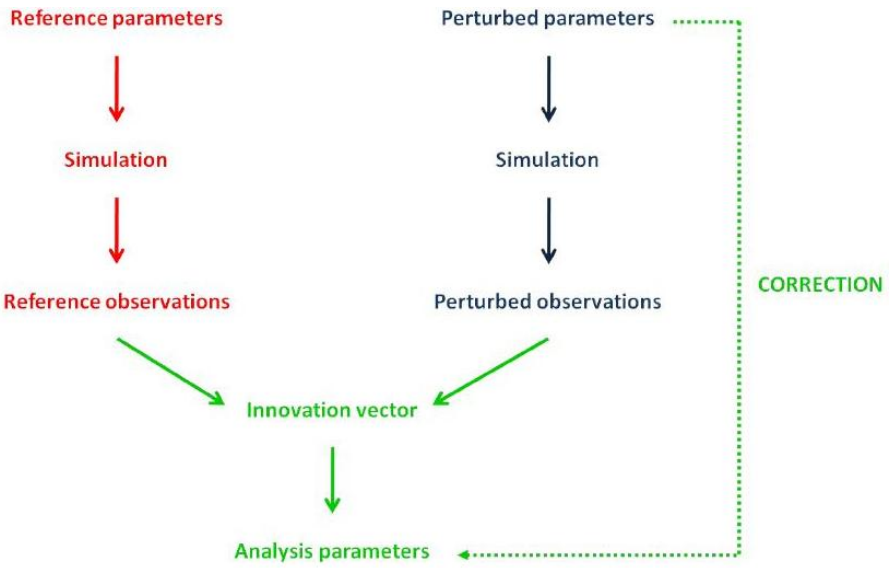


FIGURE 20 – Principe des expériences jumelles (*M. Rochoux*)

du Filtre de Kalman d'Ensemble sur le paramètre d'humidité Md. Nous nous sommes limité à un seul cycle d'assimilation sans aller jusqu'à un temps de prévision.

Voici enfin l'interface PrePALM où est indiquée les différentes composantes de l'algorithme. Dans notre cas, nous nous sommes intéressés à la correction apportée par la méthode du Filtre de Kalman d'Ensemble sur le paramètre d'humidité Md.

3.5 Validation de l'algorithme d'assimilation

3.5.1 Stratégie de débogage

Il est nécessaire de rester critique envers les futurs résultats de l'algorithme d'assmilitation. Pour ce faire, il est essentiel d'avoir une stratégie de contrôle de toutes les étapes de l'algorithme,

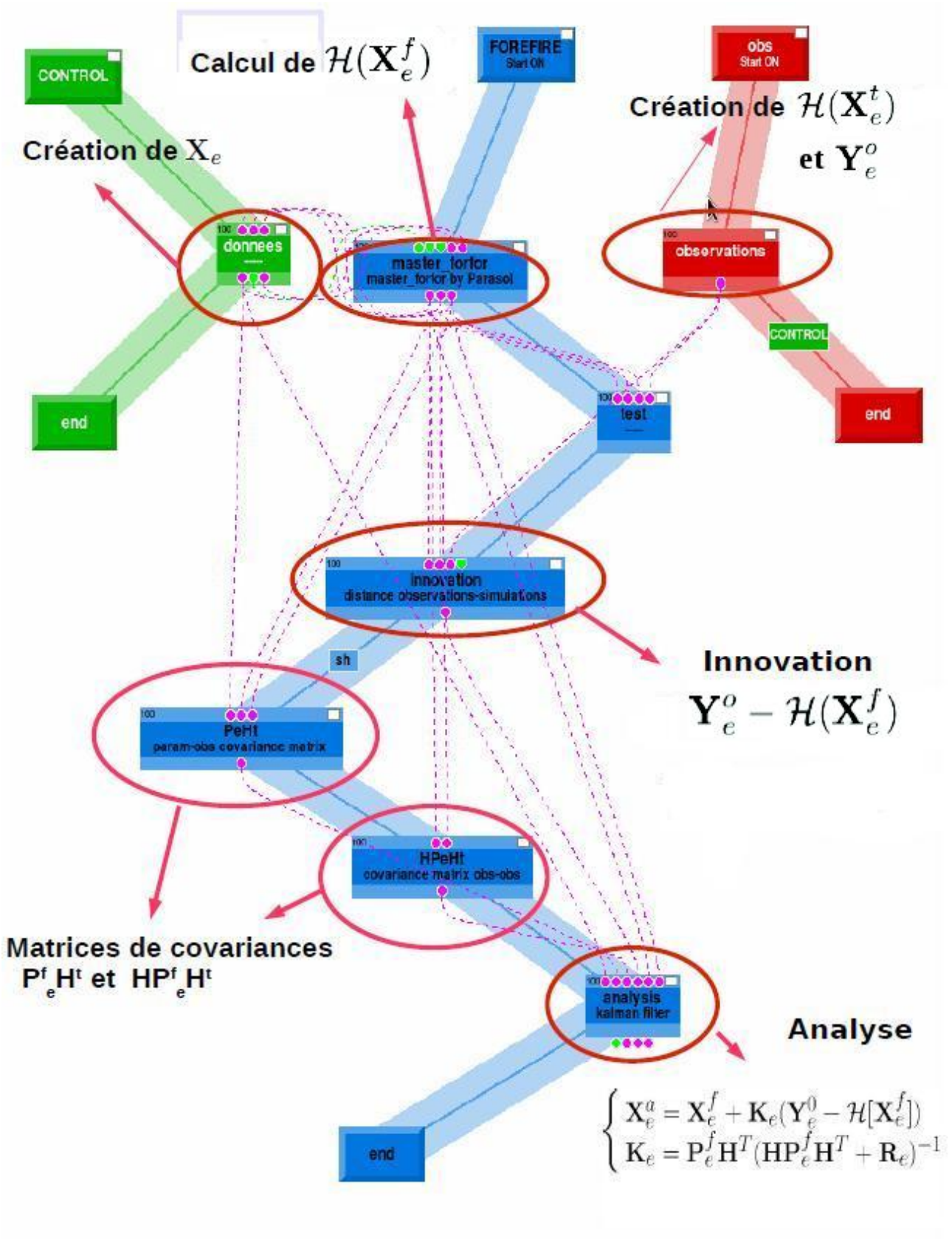


FIGURE 21 – Interface PrePALM de ForeFire avec l'emplacement des calculs réalisés pour l'assimilation de données

pour éventuellement le corriger si besoin est.

En premier lieu, on se propose de vérifier que chacun des termes de l'équation 3.19 est correctement implémenté dans Open_PALM.

$$\mathbf{X}_e^a = \mathbf{X}_e^f + \mathbf{K}_e(\mathbf{Y}_e^o - \mathcal{H}(\mathbf{X}_e^f)) \tag{3.19}$$

avec la matrice $\mathbf{K}_e$ définie comme suit :

$$\mathbf{K}_e = \mathbf{P}_e\mathbf{H}^T(\mathbf{H}\mathbf{P}_e\mathbf{H}^T + \mathbf{R}_e)^{-1} \tag{3.20}$$

Pour être sûr qu'un problème peut venir du terme que l'on étudie on doit respecter un certain ordre :

1. Premier niveau de test

   On commence par regarder la forme des fronts d'ébauche $\mathcal{H}(\mathbf{X}_e^f)$ et du front observé $\mathbf{Y}^o$ (*cf.* figure 22) puis celle de l'ensemble des fronts observé perturbé $\mathbf{Y}_e^o$.

   On s'intéresse ensuite au terme $\mathbf{K}_e(\mathbf{Y}_e^o - \mathbf{H}\mathbf{X}_e^f)$:

2. Contrôle du vecteur d'innovation

   (a) Front observé (*cf.* figure 22)

   On utilise le modéle BalbiUnsteady sans vent ni relief et le paramètre assimilé est le rapport surface/volume des particules de végétation *sd*. On choisit $5544m^{-1}$ pour la valeur réelle de *sd* et pour la valeur moyenne d'ébauche de *sd* (c'est-à-dire $\overline{\mathbf{X}_e^f} = \mathbf{X}^t$). Ce cas test est issu d'une version très modifiée de l'exemple canyon

   (b) Innovation non aberrante (*cf.* figure 23)

   On peut supposer que le problème vient de la forme du front dans le cas canyon : en effet sur les 2 bords où l'innovation est plus importante, le front "oscille" car pour pouvoir les suivre afin d'associer les points deux à deux d'un front à l'autre, on désactive la fonction *merge* qui supprime des points. Cela aurait po rajouter du bruit dans la résolution du front et donc dans les statistiques calculées sur la position des fronts. L'impact est d'autant plus important qu'il y a beaucoup de points placés dans ces zones à $t = 0$.

3. $\mathbf{K}_e = 0 \Rightarrow \mathbf{X}_e^a = \mathbf{X}_e^f$. La matrice de covariance des erreurs d'observation pour les expériences jumelles vaut $\mathbf{R}_e = (N-1)\sigma_o^2\,\mathbf{Id}_n$ où σ_o est l'erreur standard d'observation. En augmentant artificiellement σ_o au moment de la création de $\mathbf{R}_e$, on obtient : $\mathbf{K}_e = 0$ donc on doit vérifier que $\mathbf{X}_e^a = \mathbf{X}_e^f$. En exécutant cet version du code on obtient bien le même ensemble de vecteurs de contrôle, à l'ébauche ($\mathbf{X}_e^f$) et à l'analyse ($\mathbf{X}_e^f$).

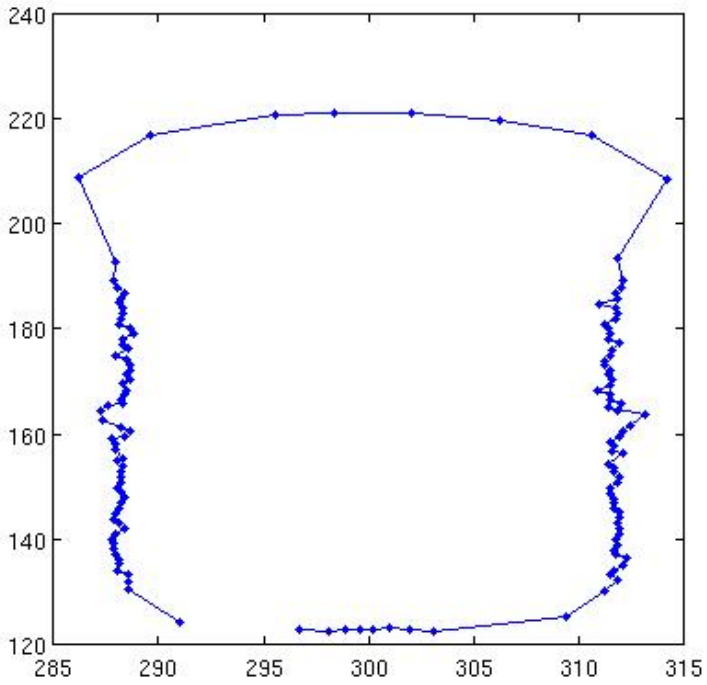


FIGURE 22 – Front observé pour un cas test

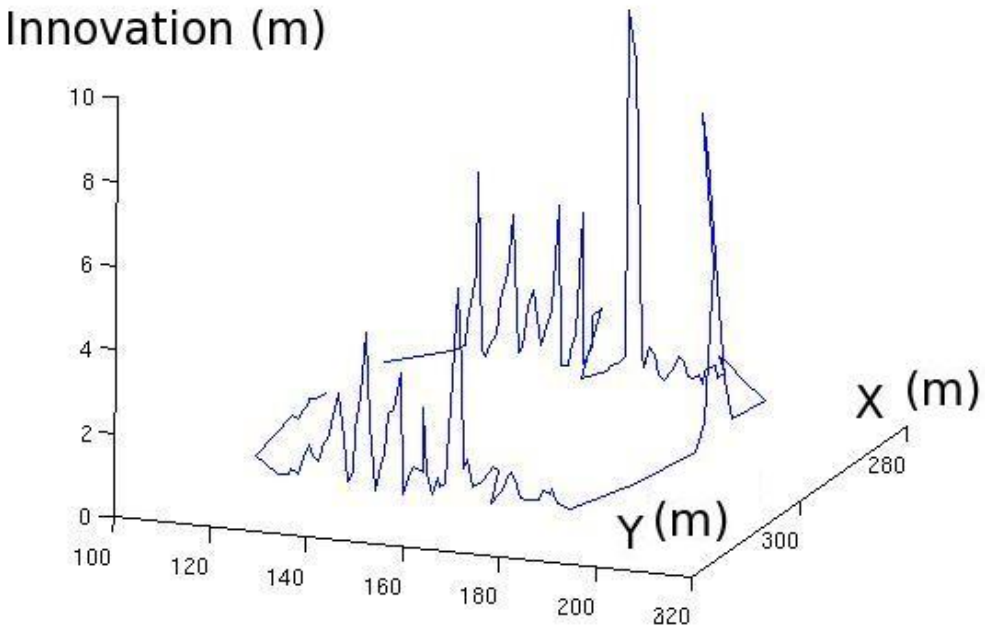


FIGURE 23 – Réprésentation de l'innovation

Pour résoudre ces problèmes il faut essayer de sélectionner autrement les points sur les fronts, ce qui permettrait de garder *merge* : Nous pourrions imaginer un algorithme proche de ce qui est fait pour FireFly ; c'est-à-dire sélectionner les points régulièrement le long d'un front qui ne correspondrait pas forcément aux points simulés puis les projeter orthogonalement sur

les autres fronts pour sélectionner les points sur tous les fronts. Cependant, cela semble assez complexe à réaliser notamment parce que ForeFire est un modèle lagrangien. Il n'y a pas de champ 2D accessible qui donnerait le contour de manière fine. Ici, il faudrait réussir à partir d'un ensemble de points espacés spécialement pour et à calculer la normale au front sur ces points.

On peut adapter le code au cas "Isotropic" et prenant comme valeur de contrôle la vitesse de propagation. On simule donc la propagation d'un front circulaire. Ce cas étant très simple (puisqu'il admet des solutions analytiques) il est beaucoup moins sujet aux bruits, et les résultats sont bien plus faciles à interpréter. Un autre intérêt est que l'on n'a plus besoin d'utiliser *merge* mais surtout l'avantage principal est que l'on a une solution analytique des matrices de covariances $\mathbf{P}_e\mathbf{H}^T$ et $\mathbf{H}\mathbf{P}_e\mathbf{H}^T$

Cela nous a permis de constater que le calcul de $\mathbf{P}_e\mathbf{H}^T$ peut être faux : sur un cas test disposant d'un relief on peut constatere la moitié des valeurs de $\mathbf{P}_e\mathbf{H}^T$ correspondantes à y étaient nulles. On a peu alors remonter dans le code de $\mathbf{P}_e\mathbf{H}^T$, et l'on constate que l'erreur vient de la communication Open_PALM : On avait relié la sortie z de l'unité *master_forfor* avec l'entrée y de l'unité *PeHt*.

3.5.2 Cas Circulaire

On se propose d'étudier ici le cas d'un feu circulaire. On considère une répartition uniforme de fuel ainsi qu'une vitesse de propagation isotrope. Cette vitesse est le paramètre de contrôle pour l'assimilation. Sa valeur vraie est fixée à $\mathbf{x}^t = 0.7m/s$. Le front vrai est représenté par 16 points ainsi que la condition initale sur la Fig. 24.

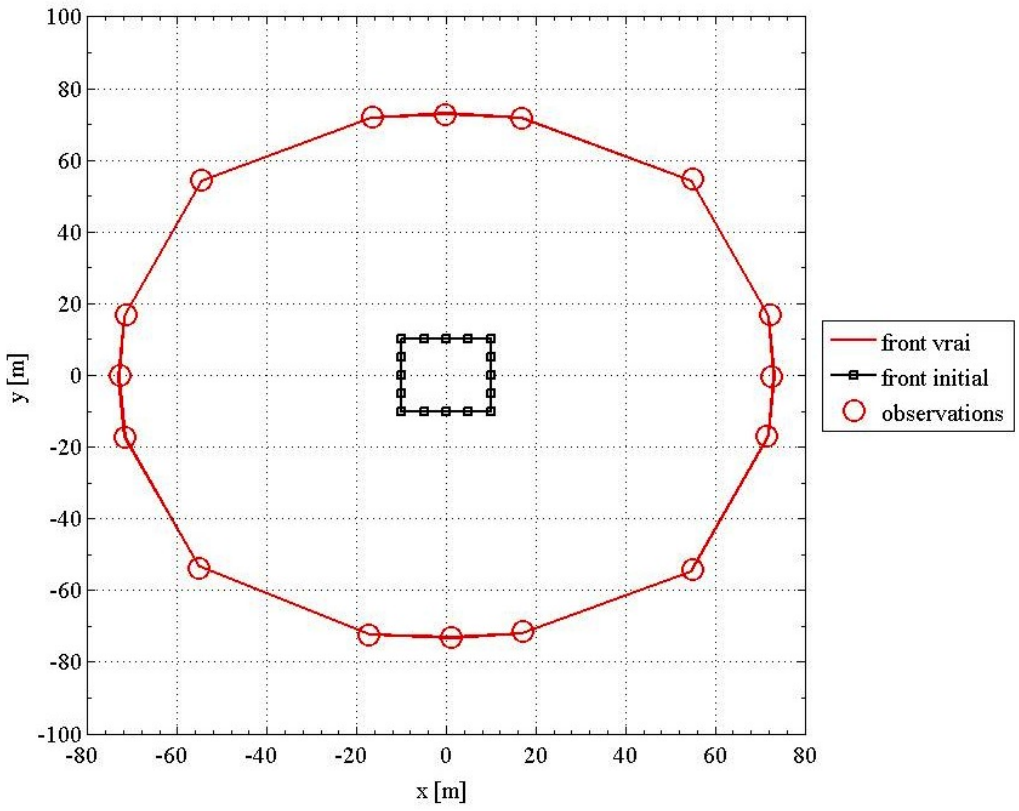


FIGURE 24 – Condition initiale en noir, position du front vrai en trait plein rouge et les 16 points d'observation (petits cercles rouges).

On choisit de partir d'un ensemble d'ébauche de 10 membres de moyenne égale à $1m/s$ et d'écart-type de $0.3m/s$. On génère de plus des observations qui ont une très faible erreur et une

moyenne égale à la valeur vraie pour qu'elle se superpose suffisament au front vrai. Comme on peut le voir sur la Fig. 25.

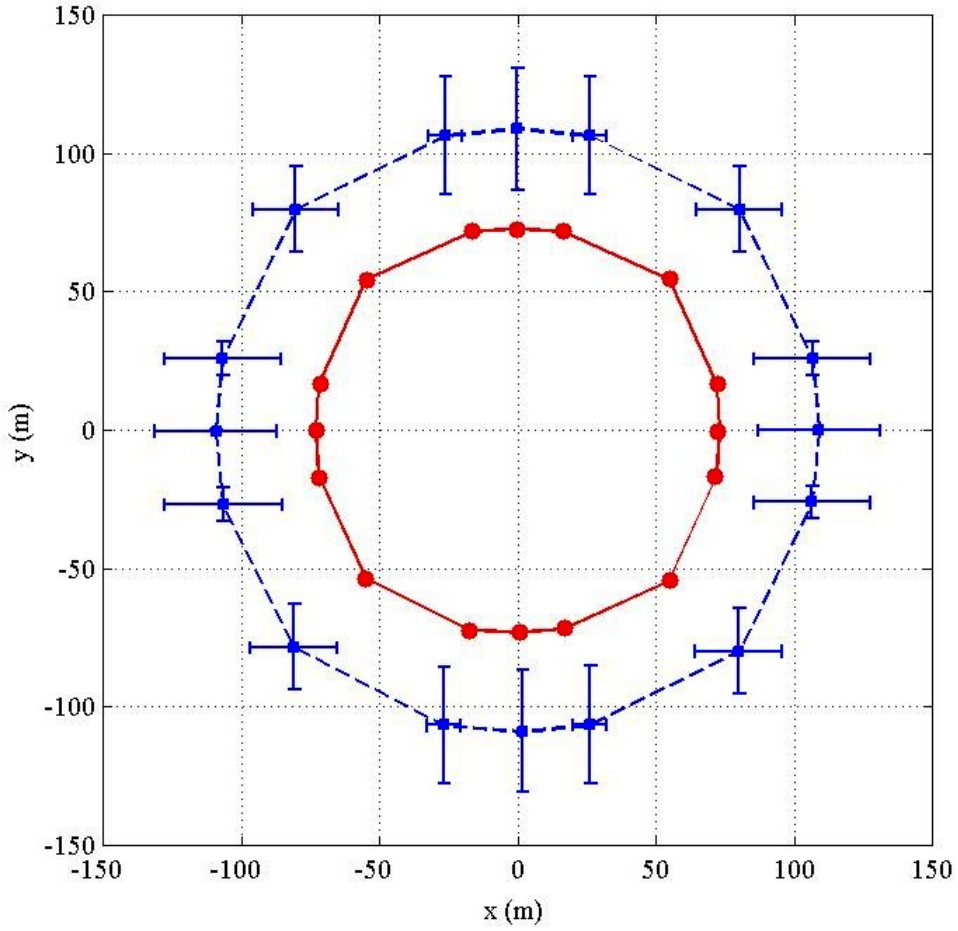


FIGURE 25 – Comparaison du front observé (en rouge) et de l'ensemble des fronts simulés avec les valeurs d'ébauche de la vitesse (en bleu). Le trait plein représente la position moyenne et les barres d'erreurs traduisent les écarts types

On obtient une analyse (de la vitesse) de moyenne $0.70m/s \pm 6.7 \times 10^{-4}$. Les fronts simulés avec les paramètres corrigés sont représentés sur la Fig. 26. Le front moyen analysé est superposé aux observations et l'écart-type des valeurs analysées est très faible. On peut donc conclure que tous les membres se sont rapprochés de la valeur vraie de façon importante.

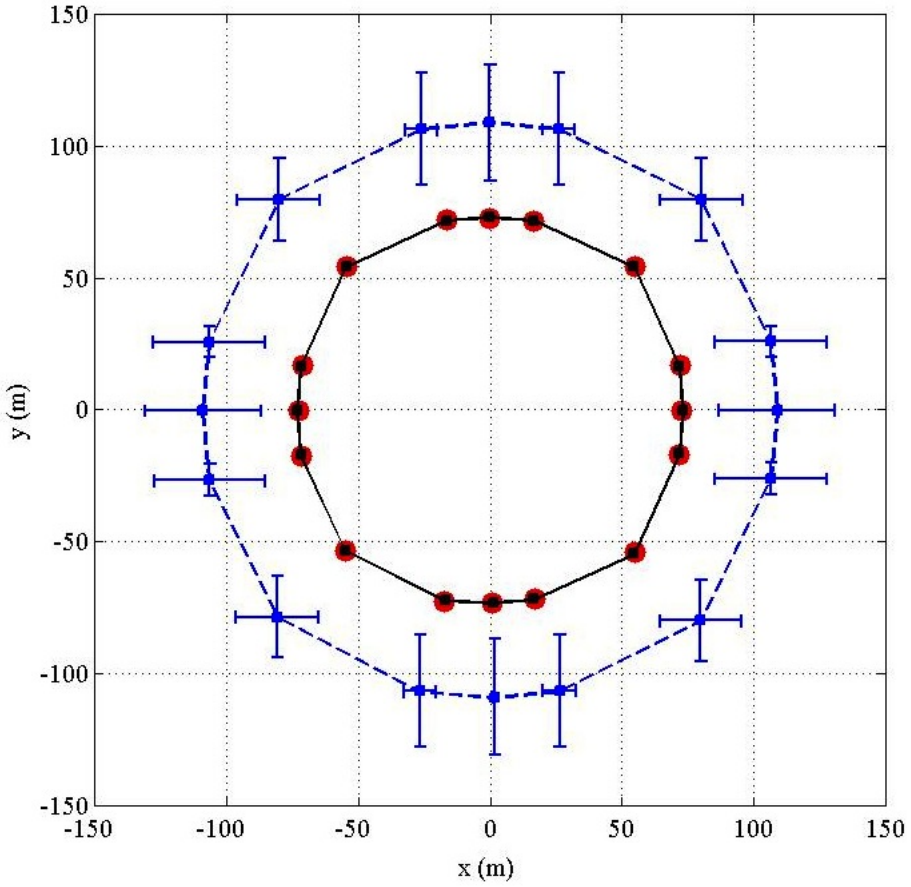


FIGURE 26 – Comparaison du front observé (en rouge), de l'ensemble des fronts simulés avec les valeurs d'ébauche de la vitesse (en bleu) et avec les valeurs d'analyse de la vitesse (en noir). Le trait plein représente la position moyenne et les barres d'erreurs traduisent les écarts types

En augmentant l'écart-type des observations, on cherche à vérifier la dégradation de la correction apportée à l'ébauche lorsque les observations sont de mauvaises qualité (c'est-à-dire bruitées). On a ainsi produit des analyses issues d'observations d'écart-type $20m$ puis $50m$. Les valeurs analysées du paramètre de contrôle (la vitesse) correspondantes sont respectivement ont une moyenne de $0.74m/s$ et $0.82m/s$.

3.5.3 Cas Canyon

Dans cette section on s'intéresse au cas d'un incendie dans un canyon, dont la topographie a été aplatie et le vent réduit à $0m.s^{-1}$. Le paramètre de contrôle est la teneur en eau dela végétation. Sa valeur vraie est fixée à $\mathbf{x}^t = 0.4$.
Le front observé est représenté par 140 points. Ces observations ont un faible écart-type de $0.1m$.On choisit de partir d'un ensemble d'ébauche de 10 membres de moyenne égale à 0.3 et d'écart-type de 0.1.

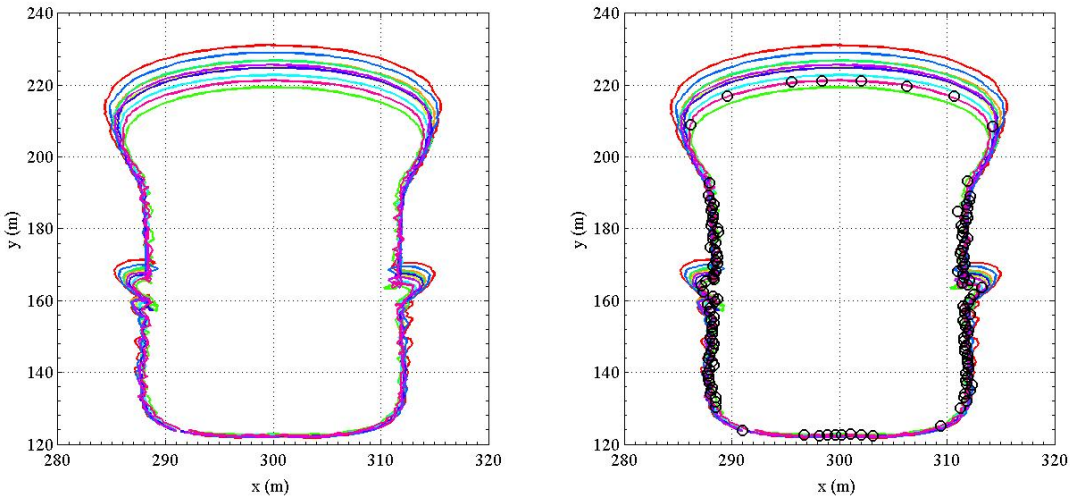


FIGURE 27 – Trace du front à 1800s simulé pour 10 valeurs de Md, les points observés sont représentés par des cercles.

Les analyses produites ont une moyenne de $0.399 \pm 8.3 \times 10^{-4}$. Le front moyen analysé est superposé aux observations et l'écart-type des valeurs analysées est très faible. On peut donc conclure que tous les membres se sont rapprochés de la valeur vraie de façon importante comme on peut le voir sur la Fig. 28

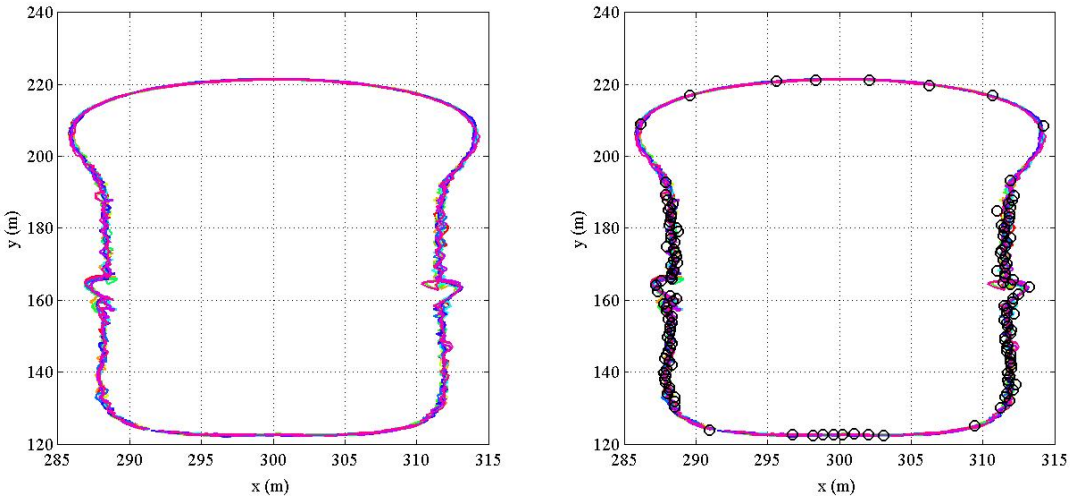


FIGURE 28 – Trace du front à 1800s simulé pour 10 valeurs de Md, les points observés sont représentés par des cercles, et le front analysé moyen en rose.

Conclusion

Au terme de ce projet, grâce à la prise en main du coupleur de code OpenPALM et de son utilitaire de parallélisation de tâches PARASOL, nous avons pu mener à bien une étude de sensibilité portant sur les paramètres d'entrée du modèle de vitesse de propagation exploité dans le simulateur lagrangien d'incendie ForeFire. Nous avons en particulier mis en évidence une sensibilité du modèle à l'humidité (Md) et au rapport surface/volume de la végétation (Sd) ainsi qu'à la vitesse du vent environnant. La suite de notre étude a consisté à implémenter un algorithme d'assimilation de données, plus précisément celui du Filtre de Kalman d'Ensemble, pour corriger un paramètre auquel le modèle s'est montré sensible : l'humidité de la végétation Md. Les corrections appliquées se sont révélées satisfaisantes puisqu'à l'issue d'un cycle d'assimilation (on ne dispose que d'un seul temps d'observation) les valeurs analysées du paramètre s'avèrent très proches de leurs valeurs vraies (qui en pratique ne sont jamais parfaitement connues, mais que nous avons fixé ici). Si le temps nous l'avait permis, nous aurions pu envisager de tester la qualité de la méthode d'assimilation de données sur des cas de plus en plus réalistes, comme ceux par exemple disponibles dans la base de données Prométhée. Nous aurions également pu envisager de prendre en compte plus de paramètres influents tels que la topographie (nous avons artificiellement aplati notre domaine), le rapport surface/volume de particules de végétation. Les améliorations futures envisageables portent sur une évolution du simulateur ForeFire. Après quelques échanges avec l'un de ses développeurs , J.B. Fillipi, il se pourrait que soient mis en place des accesseurs des paramètres contrôlés, pour éviter d'avoir à dupliquer les fichiers de configuration des simulations de façon à optimiser les exécutions multiples du code en terme de coût informatique.

Table des figures

Bibliographie

[Bar12] J.M. Bart. *Data Assimilation for Simulation of Wildfire Spread using an Ensemble Kalman Filter Technique.* CERFACS Global Change Research Team, University of Maryland - Department of Fire Protection Engineering, 2012.

[Del11] B. Delmotte. *Parameter Calibration Using Data Assimilation for Simulation of Forest Fire Spread.* CERFACS Global Change Research Team, University of Maryland - Department of Fire Protection Engineering, 2011.

[Doc12] C. Doche. *Vers l'assimilation de données pour la simulation des feux de forêts à l'échelle régionale.* Météo-France, École Nationale de la Météorologie - Toulouse INP, CERFACS Global Change Research Team, 2012.

[Dup97] J.L. Dupuy. *An analysis of semi-empirical and physical models for fire spread in wildland fuels.* National Institute for Agronomic Research – Mediterranean Forest Researches Unit - Forest Fire Prevention Research Team, 1997.

[Gor12] D. Gorham. *Validation of simplify fully physical (Balbi) wildfire rate of spread model and comparison to a semiphysical model (Rothermel).* University of Maryland, College Park, 2012.

[JB09] X. Silvani J.B. Filippi F. Rinieri J.H. Balbi, F. Morandini. A physical model for wildland fires. *Elsevier*, 2009.

[JF12] C.B. Clements J.B. Filippi, X. Pialat. Assessment of forefire/mesonh for wildland fire/atmosphere coupled simulation of the fireflux experiment. *Elsevier*, 2012.

[MR12] D. Lucor B. Cuenot A. Trouvé J.M. Bart M. Rochoux, S. Ricci. *Towards predictive simulation of wildfire spread using a reduced-cost Ensemble Kalman Filter based on Polynomial Chaos approximation.* Center for Turbulence Research, NASA AMES, Stanford University, USA, 2012.

[MR13] B. Cuenot S. Ricci A. Trouvé M. Rochoux, B. Delmotte. Regional-scale simulations of wildland fire spread informed by real-time flame front observations. *Elsevier*, Janvier 2013.

[Roc10] M. Rochoux. *Forest Fire Propagation using Data Assimilation.* CERFACS Global Change Research Team, University of Maryland - Department of Fire Protection Engineering, 2010.

[Rot72] R. Rothermel. *A mathematical model for predicting fire spread in wildland fuels.* US Department of Agriculture Forest Service, Intermountain Forest and Range Experiment, 1972.