Laboratoire des Sciences du Climat et de l'Environnement

Liste des tests post-soutenance : stage GraphCut

Mats GARVIK

Direction de recherche : Dr. Mathieu VRAC, Dr. Soulivanh THAO, Pr. Grégoire MARIÉTHOZ

31 août 2019

1 Tests post-soutenance

À la suite de la soutenance, et comme indiqué dans les perspectives de la suite du stage, nous nous sommes concentrés sur l'approche multivariée du graph cut. Cependant, dans un premier temps, afin d'essayer de réduire les temps de calculs, Grégoire a suggéré de combiner le graph cut à une autre méthode : l'érosion. Cette dernière appartient au domaine de la morphologie mathématique et permet d'effectuer une sorte de "gommage" en appliquant un kernel (représenté ici par une matrice). L'idée est de transformer la matrice du graph cut en une matrice binaire où les points attribué à un terminal sont représentés par un 1 et les autres par un 0. Ensuite, le kernel (ici une matrice en diamant) est appliqué à chaque point attribué. Si l'ensemble des points autour du point attribué est égal à 1, le point de grille central restera 1, sinon il devient 0. Cela permet de gommer les points isolés et de conserver les grandes zones de biais faibles (figure 1).

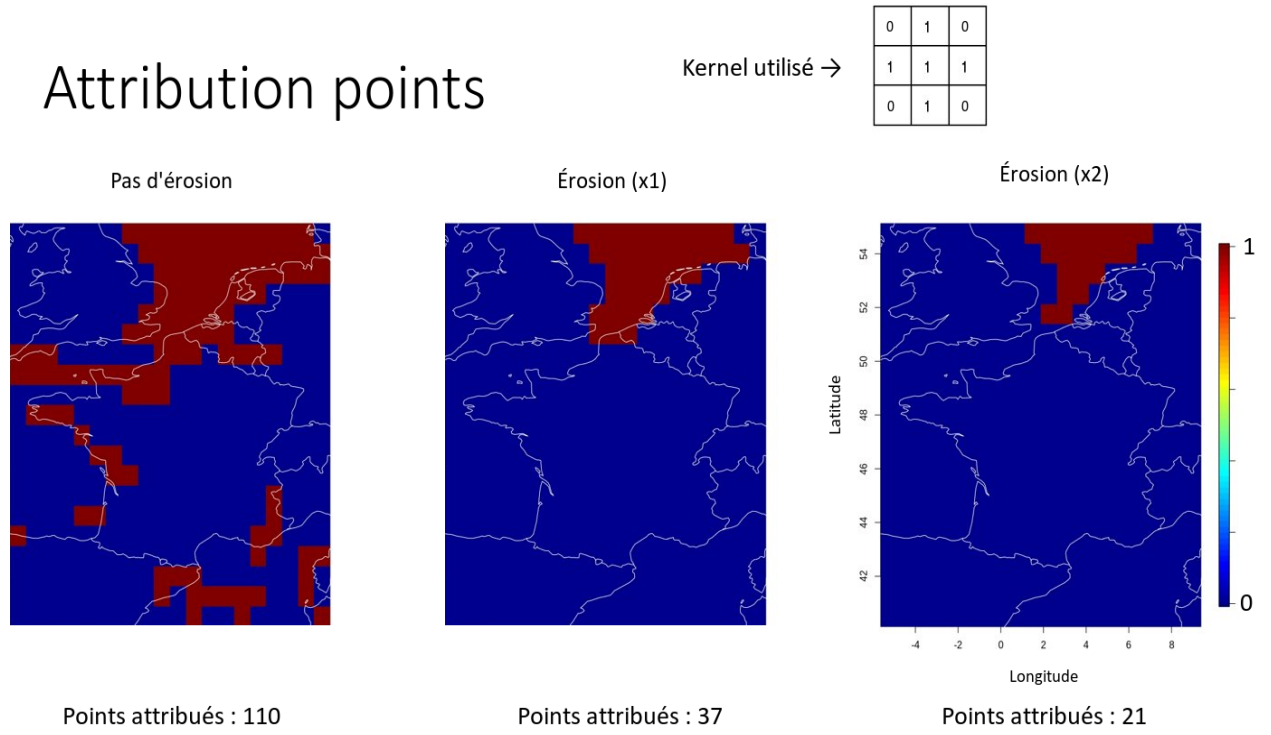

FIGURE 1 – Exemple d'érosion sur les points de grille représentant les biais faibles des modèles CNRM et IPSL en France

Dans un premier temps, le kernel utilisé était une matrice 3x3, cependant, l'érosion, même appliquée une seule fois, était beaucoup trop importante. De ce fait, une matrice en diamant semblait plus judicieux. Un des autres objectifs de cette érosion était de réduire le temps de calcul, mais malheureusement, cela n'a pas fonctionné. De ce fait, le temps de calcul de l'algorithme utilisé pour le graph cut n'est pas affecté par le nombre de points pris en compte, et ne prend que en compte la taille de l'overlap. C'est pourquoi le moyen le plus efficace de réduire drastiquement le temps de calcul est de faire un code itératif prenant en compte de petits overlaps.
Par la suite, une autre des perspectives a été réalisée : les tests multimodèles. En effet, durant la première partie du stage, le graph cut n'a été effectué que sur 2 modèles : CNRM et IPSL. Cepen-

dant, le but du test multimodèle est de voir comment se comporte l'algorithme lorsque l'on combine plus de modèles. Deux approches étaient donc possibles : soit effectuer un graph cut avec les 3 modèles directement, soit avec 2 modèles, puis refaire un graph cut avec le résultat et le 3e modèle. La première approche nécéssitant de réécrire l'algorithme, c'est la seconde approche qui a été choisie. Le 3e modèle utilisé dans ces tests est le modèle allemand MPI (Max-Planck-Institut für Meteorologie). Ainsi, afin de voir le meilleur des résultats, les 3 différentes combinaisons ont été calculées :

— CNRM/IPSL + MPI
— IPSL/MPI + CNRM
— MPI/CNRM + IPSL

Afin de voir comment fonctionne ce GraphCut en deux temps, une carte d'attribution des points de grilles aux modèle à été réalisée (figure 2) :

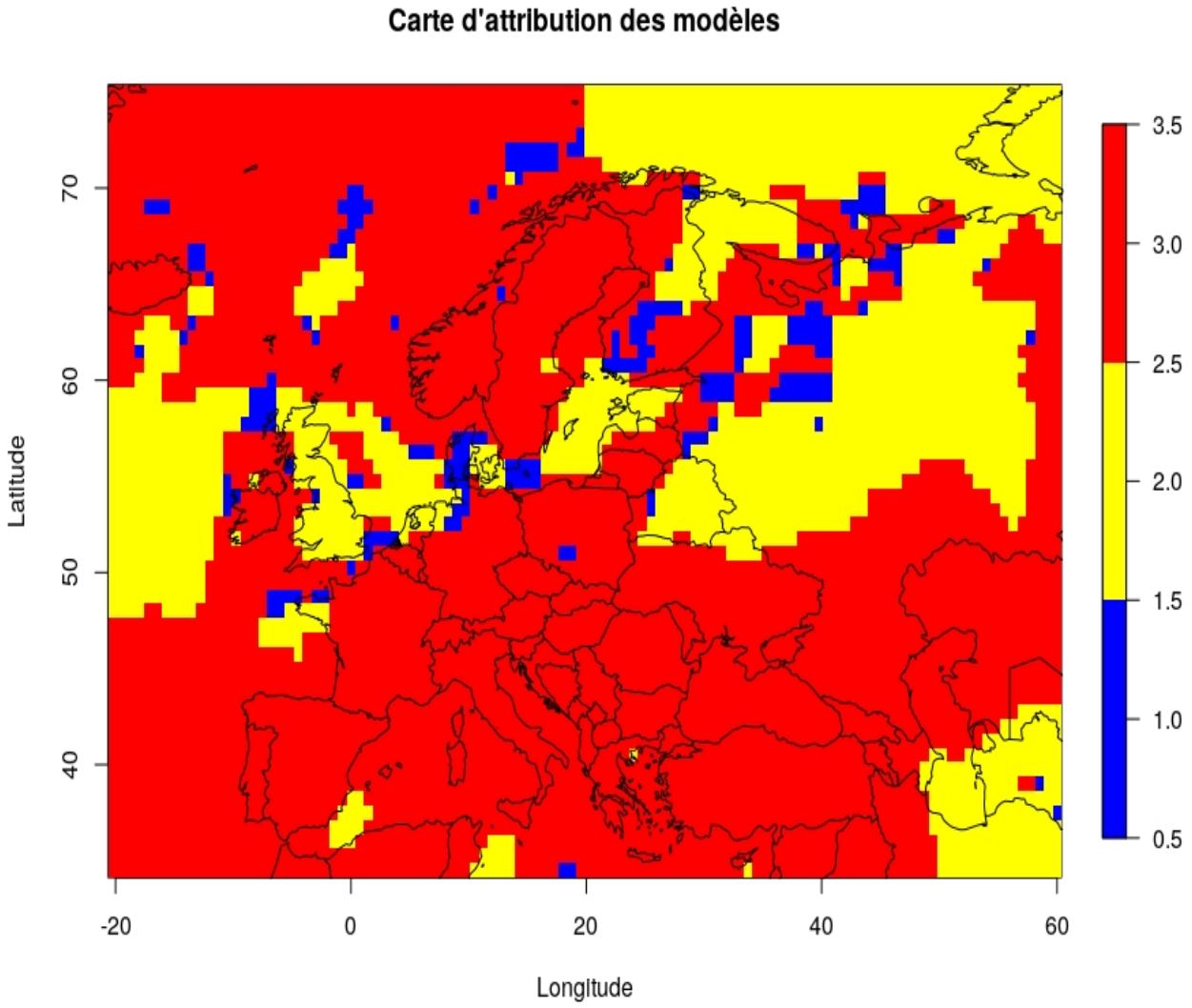

FIGURE 2 – Carte d'attribution des points de grille aux modèles avec en bleu le modèle 1, en jaune le modèle 2 et en rouge le modèle 3

Par la suite, au lieu d'effectuer des tests multivariés au graph cut, nous avons essayé d'appliquer des données multivariées. Pour ce faire, deux choix étaient possibles :

— Différence multivariée des biais

$$\delta_{1d} = [|a_1(BT_{CM1}) - b_1(BT_{CM2})|] + [|a_2(BPR_{CM1}) - b_2(BPR_{CM2})|] \quad (1)$$

— Différence des biais multivariés

$$\delta_{2d} = [a(|BT_{CM1} - BPR_{CM1}|)] + [b(|BT_{CM2} - BPR_{CM2}|)] \quad (2)$$

Afin d'effectuer le moins de modifications possible sur les données et refléter au mieux les différences avec les tests sur des données univariées, le deuxième choix a été utilisé pour les tests. Par la suite, d'autres tests ont été effectués, notamment en comparant les cuts de différentes variables. En effet, l'idée était de voir s'il y avait une forte dégradation entre le cut des valeurs de précipitations ou de températures sur des valeurs de précipitations par exemple. Bien que les résultats utilisant un cut issu de variable différente est forcément moins bon que le cut issu de la variable, ces derniers restent corrects (figure 3).

La dernière partie de ce stage consistait à rendre le code itératif et facile d'utilisation. En effet, il est possible d'obtenir l'ensemble des résultats de ce stage en changeant les paramètres en amont de la boucle (6) :

```
seasons = list(winter="winter") # A choisir entre saisons
("winter","summer","spring" et "fall") ou "annual"
quant_node = seq(0.30,0.40,by=0.10) # % de biais faibles attribues
aux terminaux entre 0 et 1 (attention, pour le domaine "fr",
les petits % ne marchent pas)
domaine = list(fr="fr") # Domaine (fr ou eur)
comb=1 # Combinaison des modeles pour un graph cut a 3 modeles
(voir la matrice combinaison ci-dessous)
models = 1:3 # Nombre de modeles en entree
method = list(st="st_smallest") # Methode d'attribution des points
aux terminaux ("no_st" ou "st_smallest")
```

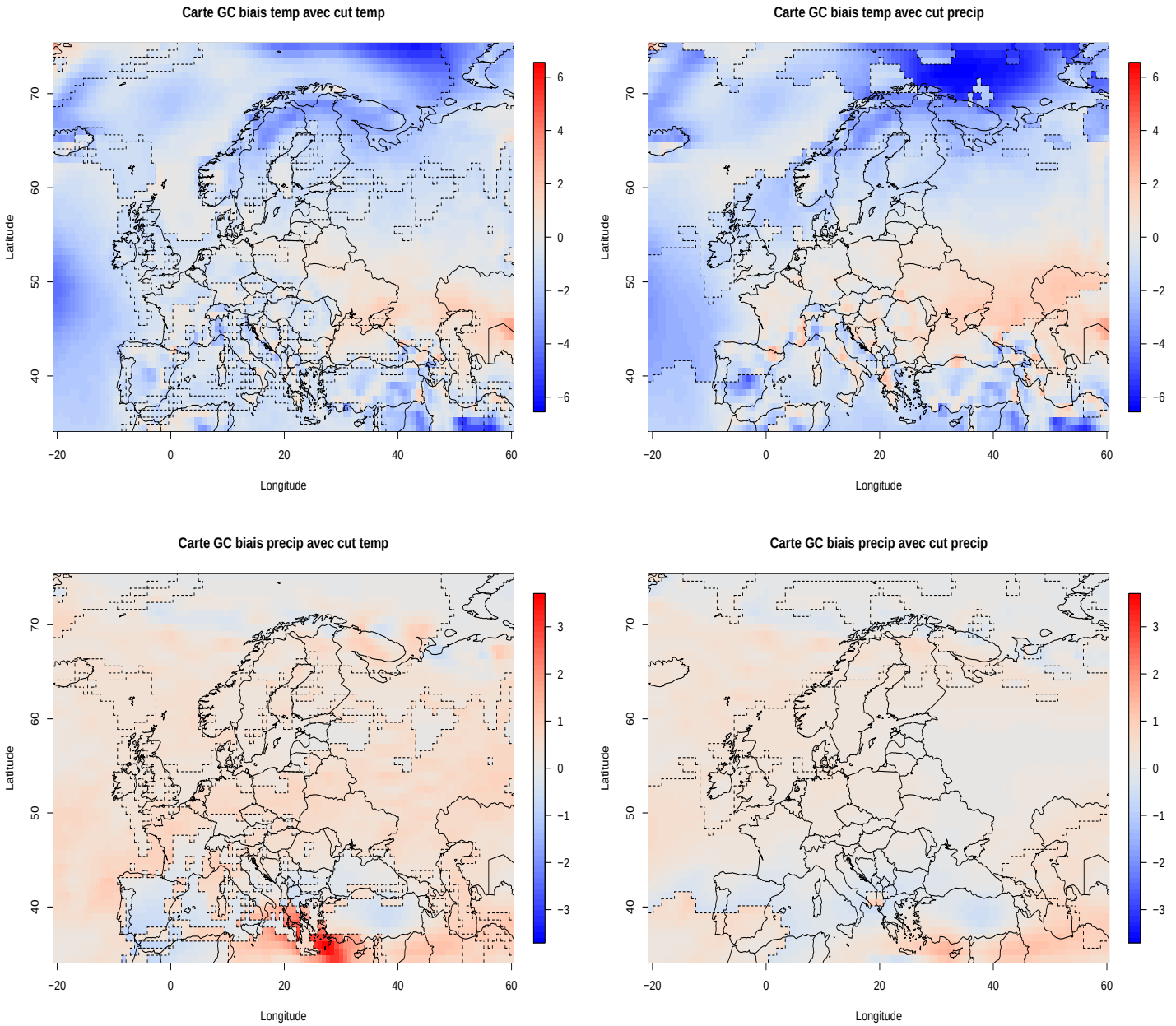

FIGURE 3 – Cartes des projections de température (haut) et précipitations (bas) avec les cuts de issu du graph cut sur des variables de température (gauche) et de précipitations (droite)

À la suite des deux GraphCut, un plot et le RData associé à la matrice du GraphCut sont sauvegardé sous la forme : **Q40_multi_winter_CNRM_IPSL_MPI_fr_st_smallest.pdf** avec :

- Q40 : % de biais faibles attribués aux terminaux
- multi : type de variables (ici multivariées)
- winter : saison
- CNRM_IPSL_MPI : ordre de combinaison des modèles (d'abord 1er et 2e puis leur résultat avec 3e)
- fr : domaine d'étude
- st_smallest : méthode utilisée.

Avec ce code, deux autres fichiers sont présents : un fichier comprenant l'ensemble des données utilisées pendant ce stage, et un autre fichier comprenant les fonctions créés afin de créer une matrice utilisable par l'algorithme du GraphCut. De ce fait, si des tests veulent être effectués pour d'autres variables, il faudra modifier le data file directement, dans la section "Données utilisées pour le code".