1 - Répartition du travail

Julien s'est plutôt penché sur la partie c tandis que Jules s'est plutôt penché sur le shell et la partie gnuplot. Nous avons quand même globalement travaillé main dans la main et nous faisions des points d'étapes fréquents afin de vérifier que le shell appelait correctement le c et que le c recevait correctement les informations envoyées par le shell (il s'est d'ailleurs avéré que cette stratégie ne s'est pas avérée rentable en termes de temps comparé aux autres groupes qui ont décidés de tout mettre en commun à la fin). Nous n'avons néanmoins pas travaillé 100% chacun sur chaque partie.

2 –Planning

La première semaine du projet, Jules a développé un bout de programme en shell permettant de récupérer les arguments rentrés par l'utilisateur (gros switch case au début du programme). Tandis que Julien à récupérer les fonctions faites en TD pour gérer les avl.

Pendant les vacances Jules a timidement avancé sur le shell en complétant le bout de programme qui récupère les arguments rentrés par l'user, il a aussi fait la partie permettant d'appeler le C avec les arguments correspondant

Au retour des vacances, Julien a travaillé sur la récupération des données par le fichier c. Nous avons revu cette partie ensemble pour être sûrs que les arguments étaient bons des deux côtés (le fameux point d'étape) (ils ne l'étaient pas)

La semaine d'avant les partiels notre professeur de TD nous as signalés que le format des paramètres à appeler en C était bien plus strict que nous le pensions donc nous avons dû refaire la partie commune.

La semaine d'après les partiels, la deadline approchant, nous avons clairement enclenchés la seconde :

Le lundi et mardi, Julien avançait sur la partie permettant de traiter les données dans le cas du mode t1 (et p1 par définition) pour l'avl. Jules quant à lui travaillait sur la manière de stocker et de filtrer les dates rentrés par l'utilisateur.

Le mercredi Jules à attaqué la partie gnuplot et a finit en pouvant tracer les graphiques pour t1, p1 et, a un beug près w, dans la nuit du mercredi à jeudi il a d'abord écrit la première version du ReadME puis a repris une partie du programme c pour préparer le terrain pour t2 et p2 (ce fut infructueux), frustré d'avoir sacrifié son temps de sommeil inutilement, il finit par implémenter le –a et –g (restrictions de longitude et de latitude). Julien de sons côté a commencé la partie sur les abr et a codé les parties permettant de traiter le cas de –h et –m

Le jeudi Jules a avancé sur la partie gnuplot, en terminant la partie permettant d'afficher les graphes pour –w puis a fait les parties –h –m, il a aussi fait une belle refonte du programme qui récupère les entrées utilisateur de manière à être le plus strict possible. Dans la nuit du Jeudi au vendredi (donc en ce moment même au moment de l'écriture de ce pdf) il écrit le pdf, il a aussi avancé sur la partie permettant de gérer t2 et p2 (quasiment fonctionnelle), il a réécrit une partie du programme de détection des paramètres dans le shell de manière à harmoniser le "skip" des arguments après -d –f –a et -g. Julien quant à lui a complètement finit les parties de gestion de t1 t2 p1 p2 m h s w.

Le vendredi de la date de rendu, Nous avons commencés par gérer un bug concernant l'envoi de dates au C et la réception des dates par le shell (dans les deux sens). Jules à finit de gérer l'affichage pour les cas –p2 –p3 –t2 –t3, il a enfin réorganisé le fichier du rendu en ajoutant des rm à la fin du programme pour rester sur un dossier clean (qui furent déplacés dans le makefile par la suite), et il a mis les différents éléments de gnuplot dans un fichier à part. Julien à finit de gérer le tri pour les modes t2 t3 p2 et p3, puis il a implémenté le tri par tab pour tous les types de donnés en se basant sur l'ébauche qu'a proposé Jules pour le tri tableau pour le mode t1, il finit par faire le makefile et à corrigé un beug du skript qui bloquait l'utilisation du –d et du –f.

3- Ce qui marche

Le programme est capable de lire n'importe quel argument rentré par l'utilisateur (la seule condition est que l'argument suivant –f soit nécessairement un fichier, les 2 arguments suivants –d soient des dates au format YYYY-MM-DD, les 2 arguments après -a et –g soient des flotants positifs)

Le programme vérifie chaque paramètre et sort une erreur dans les cas suivants :

Doublon d'un argument géographique

Doublon d'un argument de type de donnée (par exemple, -t1 –t1, en revanche –t1 –p1 ne retourne pas de code d'erreur, le programme continue normalement avec les deux valeurs)

Mauvais format des dates

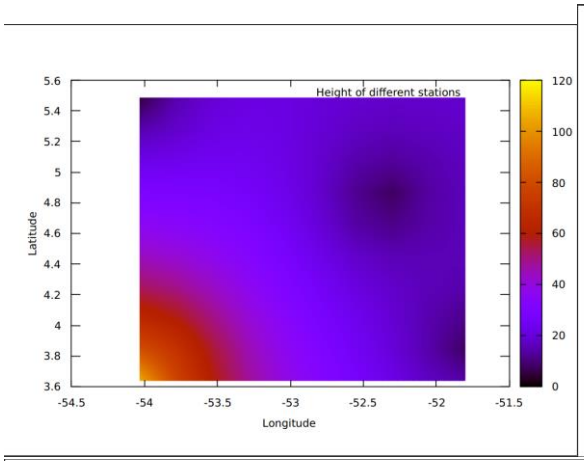
Dates non valides (Donc soit valeurs non comprises entre 2010 et 2022 soit les mois non compris entre 0 et 12 soit les jours non valides (oui les 29 février des années bissextiles fonctionnent) soit si la première date est supérieure à la deuxième)

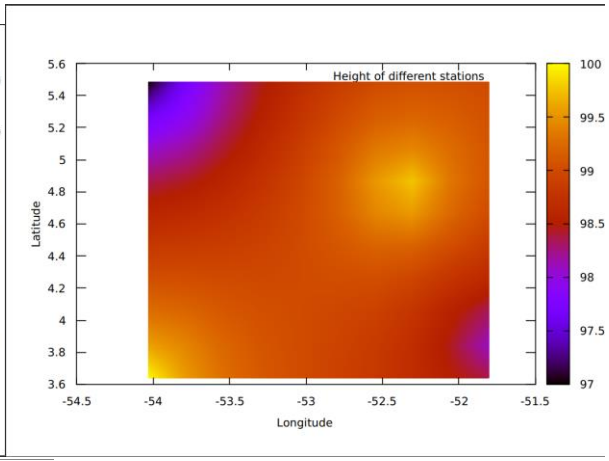
Si –f n'est pas précisé

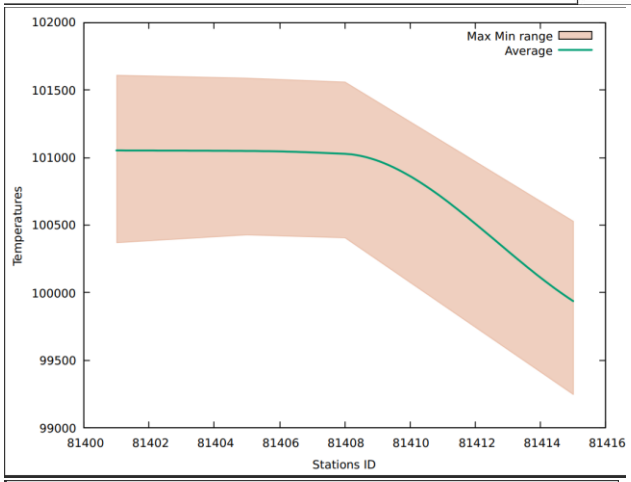
Si le fichier donné par –f n'existe pas il cherchera automatiquement meteo_filtered_data_v1.csv mais s'il n'existe pas le programme quittera

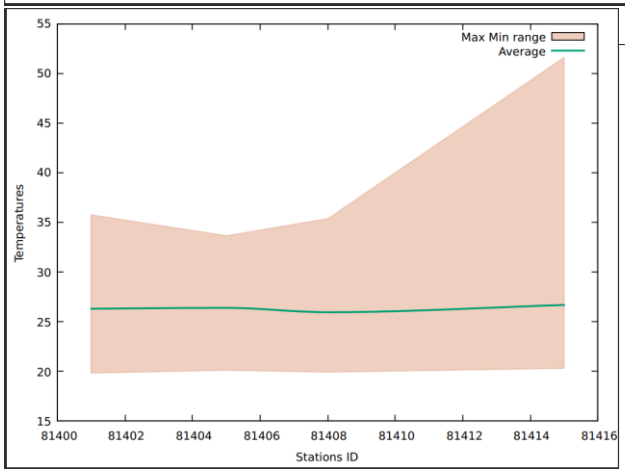
Ensuite le programme effectuera dans l'ordre : les restrictions temporelles puis géographiques puis appelleras le c n fois en fonction du nombre de type de donnée demandé il afficheras à chaque fois le graph demandé puis quitteras

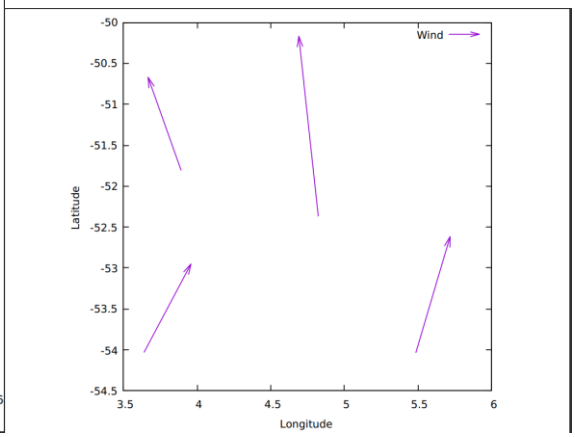
Les arguments qui marchent correctement sont –t1 –p1 –w –h –w –t2 –p2 –t3 -p3 par exemple avec les restrictions –G et –d 2010-09-30 2011-09-30, donc en rentrant, ./skript.sh -G –f meteo_filtered_data_v1.csv -t1 –p1 –w –h -m –d 2010-09-30 2011-09-30 on obtient :











4- Ce qui ne marche pas ou mal

Le mode de tri par tableau ne fonctionne pas pour t3

Pour le tri des modes p2 et t2 le fichier c ne tri que par date et pas par heure

Le fichier en entrée doit être nécessairement situé dans le même répertoire que le skript.sh