Théo Belliere

Maëlys Picault

Pré-ING 2 MI Groupe 5

Compte-rendu de projet

CY-Météo

Sommaire

1.Le projet

a. Objectifs du projet

Ce projet a pour principal objectif de créer une application qui va traiter un fichier météorologique pour ensuite afficher des graphiques.

Le programme est composé de deux parties, une partie en Shell et une en C.

b. Fonctionnement de la partie Shell

La première partie, en Shell a pour objectif de filtrer les données. Le programme récupère les différentes options entrées par l'utilisateur, à l'aide de *getopts*. Puis il vérifie l'unicité des options de lieu, de fichier et de date, mais également la présence d'au moins une option de données. Si les conditions ne sont pas respectées le programme va afficher un message d'erreur approprié. Le programme Shell prend ensuite le fichier csv à étudier et en fonction des différentes options entrées par l'utilisateur, il va ensuite créer un ou plusieurs fichiers csv temporaires avec uniquement les informations voulus, un par option de donnée. Ces fichiers csv sont ensuite envoyés à l'exécutable de tri, qui sort autant de fichiers triés qu'il y a de fichiers filtrés.

Une fois que le C a terminé le tri des fichiers, le script Shell va dans un premier temps traiter le code de retour du programme de tri, 0 si tout va bien, 1 pour une erreur sur les options activées, 2 pour une erreur avec le fichier de données d'entrée, 3 pour une erreur avec le fichier de données de sortie et 4 pour toute erreur d'exécution interne. Le script Shell utilisera ensuite *gnuplot* avec les fichiers triés, puis effacera les fichiers csv temporaires.

Le filtrage sans restriction géographique utilise la commande « cut » qui sélectionne les colonnes nécessaires pour la suite, et enlève la première ligne qui contient les titres de colonnes.

Le filtrage par restriction géographique utilise « grep » qui cherche les lignes où le code communes correspond à l'emplacement géographique souhaité. Malheureusement, ce type de filtrage n'est pas exhaustif et entraîne un manque de données. Par manque de temps, le filtrage par coordonnées GPS n'a pas pu être implémenté.

Pour l'option d, le script récupère l'argument avec la date min et la date max (de la forme « AAAA-MM-DD,AAAA-MM-DD ») et récupère dans deux variables la date minimum et la date maximum. Malheureusement, par manque de temps, le filtrage par date n'a pas pu être implémenté.

Les graphiques sont générés par *gnuplot*. Les images sont exportées automatiquement dans le dossier data.

c. Fonctionnement de la partie C

La seconde partie, en C a pour objectif de trier les données du fichier csv filtré qu'il récupère en fonction du mode de tri voulu. Il y a trois modes de tri, avl, abr et tab. Le programme C triera en fonction du mode voulu. En revanche si l'utilisateur n'en demande pas un en particulier, il triera en AVL qui est le plus rapide. Il va donc ensuite créer un autre fichier csv trié. Si le programme C arrive à effectuer son traitement correctement jusqu'au bout, il doit retourner la valeur 0, sinon il retourne une valeur strictement positive.

L'option h exécute un tri par altitude. En cas de doublons, les stations sont ajoutées en fils du milieu, et si elle existe déjà dans l'arbre, elle est ignorée.

L'option m exécute un tri par station. En cas de doublons, on regarde la valeur d'humidité du nouvel élément, et si elle est plus grande, alors elle est le nouveau maximum. Un deuxième tri est exécuté, cette fois-ci par humidité, de la même manière que pour l'altitude.

L'option w exécute un tri par station. En cas de doublons, les valeurs de vitesse de vents et leur direction sont ajoutées aux propriétés du nœud. Puis lorsque le tri est terminé, un parcours calcule la moyenne de ces valeurs pour chaque station.

L'option p1/t1 exécute un tri par station. En cas de doublons, la valeur de température est ajoutée à la propriété du nœud et on cherche un minimum et un maximum. Puis lorsque le tri est terminé, un parcours calcule la moyenne de ces valeurs pour chaque station.

L'option p2/t2 exécute un tri par date. En cas de doublons, la valeur de température est ajoutée à la propriété du nœud. Puis lorsque le tri est terminé, un parcours calcule la moyenne de ces valeurs pour chaque station.

L'option p3/t3 exécute un tri par date. En cas de doublons, la nouvelle valeur est ignorée. Puis lorsque le tri est terminé, un parcours fait un autre tri, cette fois ci par station. Cette option n'est pas optimisée et est très longue à exécuter.

Le fichier C doit être exécuté avec l'option f pour indiquer le fichier d'entrée, l'option o pour indiquer le fichier de sortie, l'option t avec en argument le mode de tri, et l'option k avec le mode de donnée.

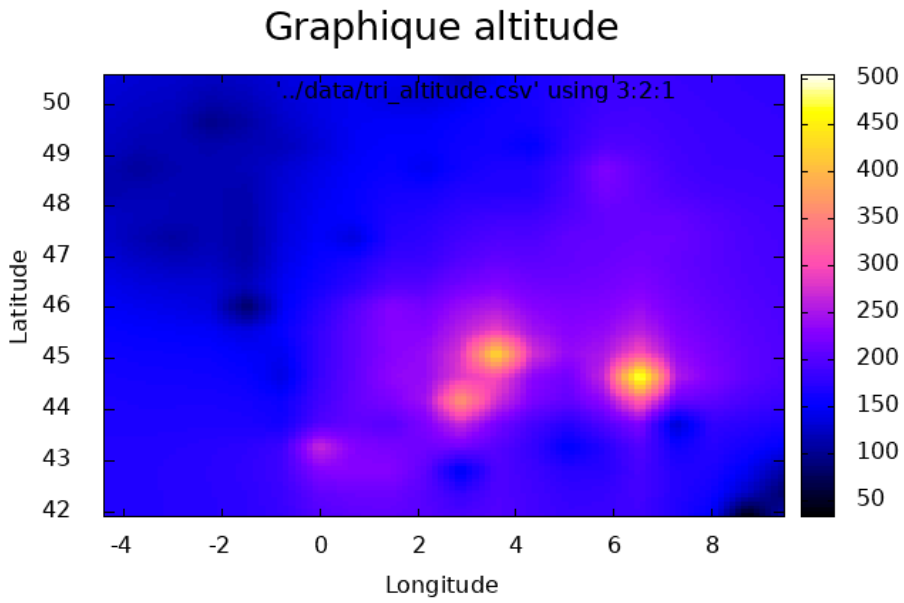
d. Exemples d'exécution

Exemple de fichier de filtrage et de tri pour l'altitude :

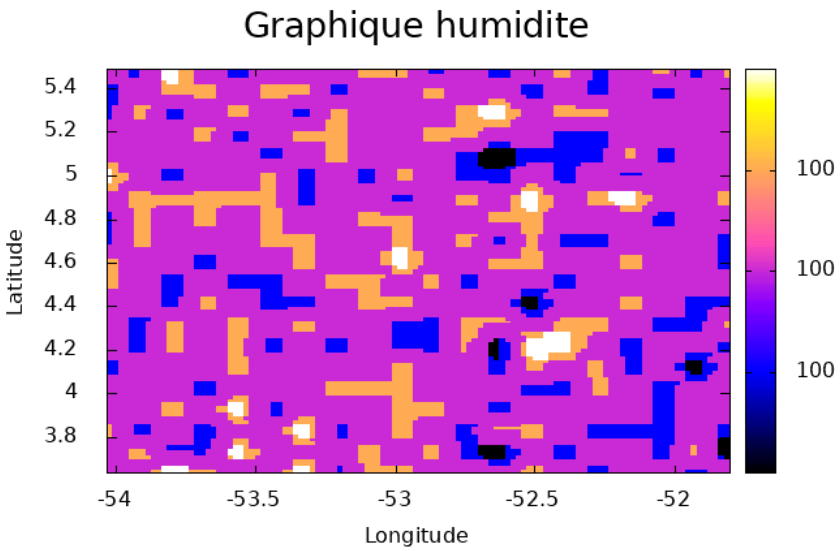
Le fichier filtre_altitude.csv est filtré pour avoir uniquement les données nécessaires au tri, et au graphique, à savoir le numéro de station, les coordonnées GPS et l'altitude. Il contient seulement toutes les stations situées aux Antilles. Le séparateur de champ est « ; ». Les champs sont dans l'ordre où ils sont dans le fichier de référence, à cause du fonctionnement de la commande « cut ». Le champ contenant les coordonnées géographiques est considéré comme un seul champ, traité comme chaîne de caractères.

Le fichier trié a pour délimitateur « , » car il est facile d'intégrer les coordonnées géographiques comme deux champs. Il y a donc trois champs, le premier étant l'altitude, le deuxième la latitude et le troisième la longitude. Les données inutiles comme le numéro de station ne sont pas ajoutées au fichier.

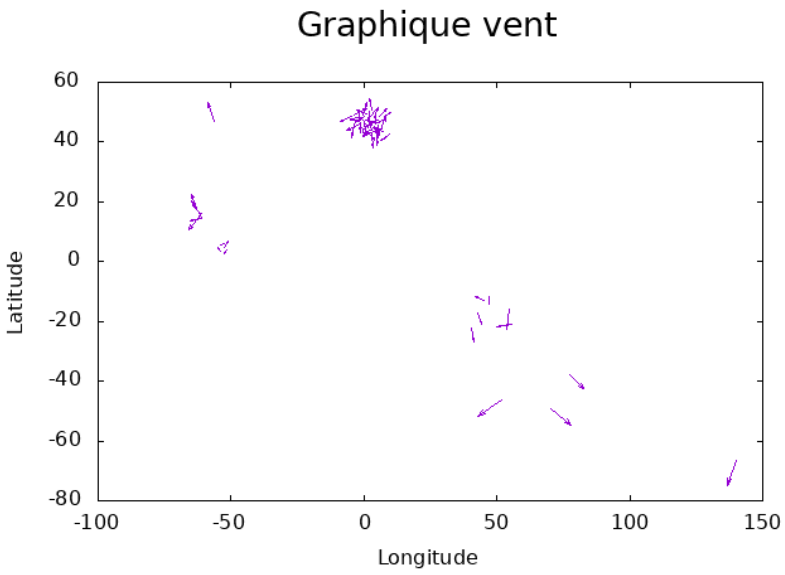
Exemples de graphiques générés par *gnuplot* :



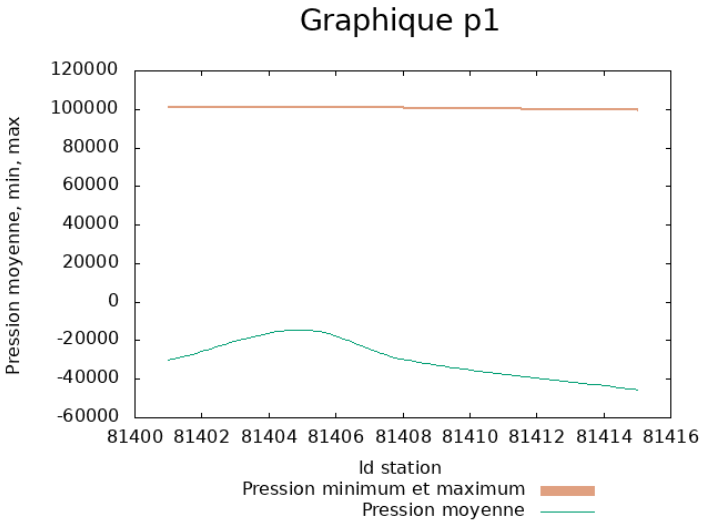
Altitude en France



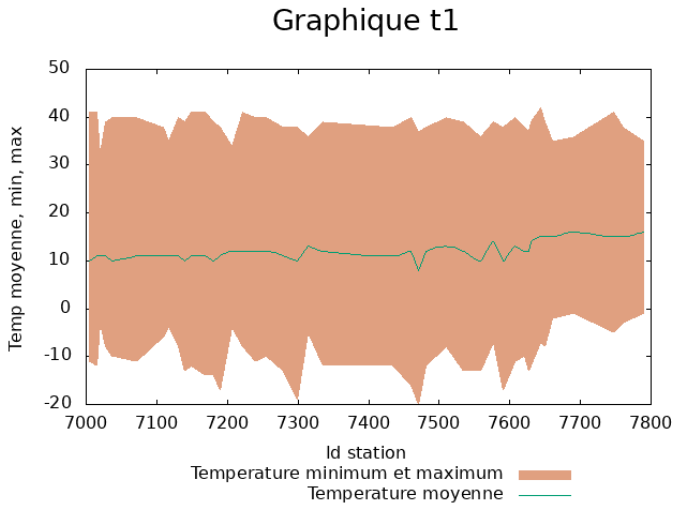
Humidité en Guyane



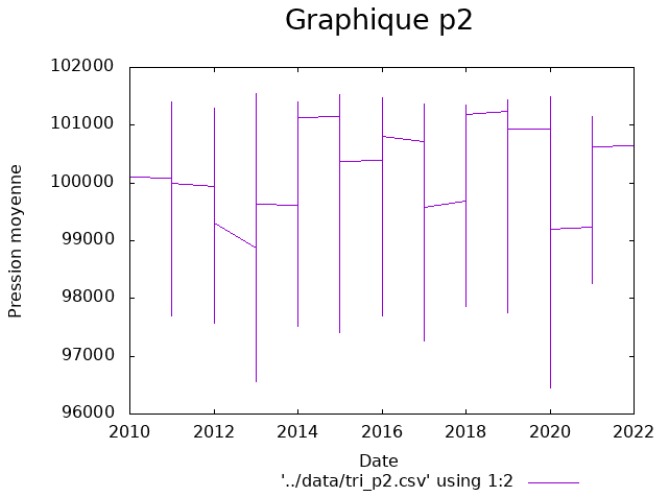
Vent dans le monde



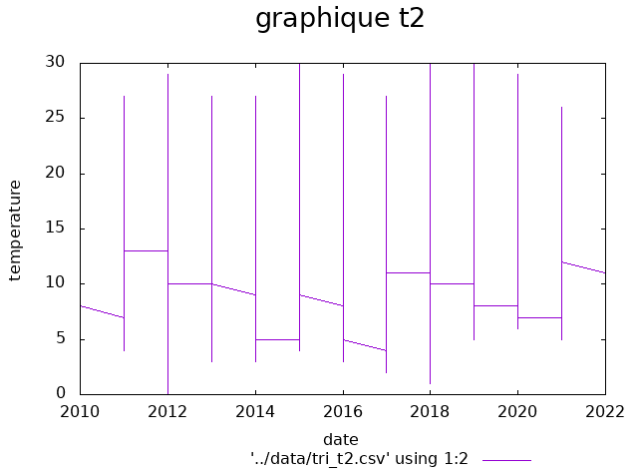
Mode p1 en Guyane



Mode t1 en France



Mode p2 dans le monde



Mode t2 dans le monde

Les graphiques selon le mode p3 et t3 n’ont pas été implémentés.

e. Problèmes techniques

Toutes les options du script Shell sont prises en compte, mais ne sont pas toutes opérationnelles. Les modes m, h, w, p2, t1, t2 fonctionnent correctement, cependant les modes p2 et t2 sont peu optimisés, et les modes p3 et t3 sont très longs à exécuter. Le mode p1 donne des moyennes de pression négatives, ce qui fausse le graphique généré par le suite.

Les données à valeur flottante comme la température sont tronquées avant la virgule, ce qui altère légèrement les valeurs de températures minimales, maximales et moyennes. Les données de vitesse de vent, quant à elles, sont correctement prises en compte.

L'option d est considérée par le script Shell mais ne fait aucun filtrage par date.

Les graphiques selon le mode p3 et t3 n'ont pas été implémentés.

Les données géographiques sont filtrées via le code commune des stations, et non par position géographique. Cependant, certaines lignes du fichier ne contiennent pas ces codes, ce qui entraîne une perte de données partielle, voire presque totale pour certaines régions.

Malheureusement, la majorité du script Shell et du programme C a été codée spécifiquement pour chaque option de donnée, ce qui entraîne beaucoup de duplication de code. Cependant, le code pour chaque option est bien adapté celle-ci. En conséquence, l'option r pour le programme de tri est totalement inutile.

Le tri en C est opérationnel pour les AVL et ABR. Le tri par liste chaînée a été ajouté, mais ne fonctionne pas.

2.Notre organisation

a. Comment on s'est organisé

Tout d'abord lors de l'arrivée du sujet, nous en avons pris connaissance chacun de notre côté. Nous avons ensuite dans la semaine pu discuter de ce que nous avions compris ou pas et commencé à réfléchir à comment procéder. Puis, nous avons travaillé le plan de notre projet et chaque étape que nous allions devoir travailler. Les vacances sont vite arrivées et on avait commencé à se répartir des taches, Théo était chargé de commencer le Shell et Maëlys le programme en C. Au retour des vacances chacun avait avancé à son rythme, Théo a débuté le Shell comme prévu et Maëlys à regrouper toutes les fonctions de base pour les ABR, AVL et les listes. Lors de la première semaine au retour des vacances Theo a fait le getopts en laissant la gestion de la date à Maëlys. Ensuite la seconde semaine après le retour des vacances Maëlys s'est chargé de faire le Makefile et Théo a fait le filtrage et il a commencé à faire la lecture des fichiers en C. Ensuite l'arrivée des partiels nous a freiné dans ce projet et nous n'avons avancé. La semaine suivante, nous avons fait tous les tris en ABR et en AVL. Maëlys a fait le README, le compte rendu et l'option –help du script Shell.

b. Comment on aurait pu s’améliorer

La différence de niveau entre Théo et Maëlys a posé un problème dans la répartition des tâches. Ajouté au manque de temps, la qualité du code a diminué et le cahier des charges n’a pas pu être complètement respecté.