Rapport algorithmes de pré-traitement

Arthur Fezard

12/06/2017

1 Introduction

Ce petit rapport se concentre principalement sur les fichiers python qui servent à effectuer le pré-traitement des fichiers. Ces algorithmes vont permettre de préparer les fichiers du CCD et de normaliser leur nom afin d'avoir un suivi plus facile des différentes nuits.

2 Search_bias.py

Ce premier bout de code se contente de récupérer tout les bias d'une même nuit et de les sauvegarder dans un nouveau dossier fits. Il est composé de 2 fonctions.

2.1 find_bias

Cette fonction permet de stocker dans un tableau tout les fichiers bias (de terminaison "b.fits") qui se trouve dans un dossier. Elle prend en argument le chemin absolu du dossier en question. Dans le cadre de Néo-Narval, ce dossier est celui dans lequel vont arriver toutes les données d'une même nuit.

2.2 combine_bias

Cette fonction permet de récupérer tout les bias d'une même nuit et d'en faire la moyenne et de stocker cette moyenne dans un nouveau fichier fits. Elle prend en argument le chemin absolu du dossier ou chercher tout les bias. Tout d'abord, cette fonction fait appel à la fonction précédente afin de stocker tout les bias dans un même tableau. L'idée est de travailler uniquement sur ce tableau. Le premier élément de ce tableau est notre référence pour la nuit. On stocke le header entier de cette référence et l'on garde également la donnée "DATE" et l'image de ce fichier. Une fois ceci fait, on supprime du tableau de départ ce fichier. La donnée "DATE" étant présente dans le nom du fichier que l'on va créer, on la convertit grâce au module datetime de python (au format "YMD_HMS" mais l'on y reviendra plus tard).

Ensuite, pour chacun des fichiers restants dans le tableau, on récupère la donnée "DATE" des fichiers. Si la différence entre le "DATE" actuel et le "DATE" de référence est inférieur à 1h : alors on considère que ces 2 fichiers font parti de la même nuit. Dans ce cas, on somme leur images. Si le fichier a été utilisé, il est alors supprimé du tableau de départ.

Ainsi, lorsque tout les fichiers ont été ouverts au moins une fois, on crée le nouveau fichier fits contenant pour image la somme de tous les fichier bias utilisés que l'on divise par le nombre de ces fichiers. On copie également les header du fichier de référence dans le header du nouveau fichier. Seule la donnée "TIMEEND" est remplacée par la donnée "TIMEEND" du dernier fichier utilisé. Le nouveau fichier est nommé "Narval_YMD_HMS_10b.fts", ou "YMD_HMS" correspond à l'année, mois, jour,heure, minute et seconde du fichier de référence pour la nuit.

Si le tableau de départ est non vide, le nouveau premier fichier sert de référence pour une nouvelle nuit et l'on recommence l'algorithme jusqu'à ce que le tableau soit vide.

3 Search_flat.py

Cette partie est consacrée a l'algorithme qui permet de faire la moyenne d'une série de flats. Il crée un nouveau fichier avec le format "Narval_YMD_HMS_10f.fts". Il a la même structure que l'algorithme précédent avec cependant quelques sous-fonctions supplémentaires.

3.1 find_flats & find_10bias

Ces 2 fonctions distinctes permettent de stocker dans un tableau les flats (fichiers se terminant par "f.fits") et dans un autre tableau les 10bias (fichier se terminant par "10b.fts"). Elles prennent en argument le chemin absolu du dossier en question. Dans le cadre de Neo-Narval, ce dossier est celui dans lequel vont arriver toutes les données d'une même nuit pour la fonction find_flats. Pour la fonction find_10bias, ce dossier est celui où sont rangés les fichiers 10b.

3.2 find_10bias_date

Cette fonction permet de trouver le fichier 10b de la même nuit que le fichier de référence. Elle prend en argument le tableau contenant les fichiers 10b et le nom du fichier de référence (qui est dans ce cas "YMD_HMS"). L'algorithme considère que le fichier de référence et le bias sont de la même nuit s'ils sont espacés de moins de 4h.

3.3 combine_flats

Cette fonction permet de récupérer tout les flats d'une même nuit et d'en faire la moyenne et de stocker cette moyenne dans un nouveau fichier fits, et également de retirer le bias. Elle prend en argument le chemin absolue du dossier où chercher tout les flats. le fonctionnement est le même que pour combine_bias. Cependant, il y a au sein de ce code une sécurité supplémentaire. En effet, grâce à l'algorithme de chelli, on vérifie que les ordres ne se sont pas trop décalés sur le CCD d'un flat à l'autre. Si le décalage est inférieur à 0.5 pixels, on peut sommer les différents flats. La sécurité est volontairement grande par rapport à la précision, cela nous permet de détecter et d'éliminer les très grosses erreurs sans forcément éliminer les erreurs faibles. Enfin, cette fonction enlève automatiquement le fichier 10b de la nuit au fichier 10f que l'on crée.

4 Le traitement des autres fichiers

Pour finir, il nous faut également traiter les autres fichiers, à savoir le thorium-argon, le fabry-perrot et les images stellaires. Il faut seulement enlever le fichier 10b pour chacun de ces fichiers. Ainsi, les 3 algorithmes Search_XXXX (ou XXXX est soit thar, fp ou star) sont structurés de la même façon.

Chacun de ces algorithmes possède sa propre fonction find_XXXX (qui fonctionne exactement de la même façon que find_flats et find_bias) et également les fonctions find_10bias et find_10bias_date qui ont déjà été détaillées précédemment. Le seul détail qui change est pour la fonction find_10bias_date pour l'algorithme Search_star : l'intervalle de temps correspondant est passé de 4h à 12h.

la nouvelle fonction est la fonction XXXXs_unbiased qui permet de soustraire le 10b de la bonne nuit. En utilisant les fonctions précédentes, l'algorithme trouve le fichier 10b de la même nuit, l'enlève à l'actuelle image et créer un nouveau fichier sous le format "Narval_YMD_HMS_xxx.fts" ou xxx est :

- th1 pour le Thorium-Argon
- fp1 pour le fabry-perrot
- st1 pour les données stellaires

5 Fonction find_path, la fonction commune

Cette fonction permet de chercher tout les chemins qu'il nous faut lorsqu'ils sont stockés dans un fichier ".txt".

Dans un fichier nommé "path.txt", on stocke les chemins des différents dossiers utiles dans lequel on va stocker les différents fichiers que l'on crée. Les chemins sont stockés de la forme "Name : path", avec un saut de ligne entre chaque.

La fonction find_path prend donc en argument le chemin du fichier path.txt et le nom du dossier dont on veux récupérer le chemin. Elle renvoie donc en sortie le chemin en question.

6 Fonction cosmic_auto

Cette fonction permet de traiter les cosmiques en les retirant de nos divers fichiers fits. Elle prend en argument le fichier donné le fichier fits que l'on veux traiter, sa reproduction crée par la matrice A et le fichier contenant toutes les enveloppes. Elle crée en retour un fichier fits avec le bon nom et la bonne terminaison.

Cette méthode repose sur le fait que la matrice A va atténuer les cosmiques lorsqu'elle recrée une image. Nous partons du principe que les cosmiques qui vont nous déranger sont ceux qui ont une intensité nettement supérieure à l'ordre que l'on étudie. Donc pour chacune des lignes de la CCD, on stocke le pixels qui a le maximum d'intensité pour l'ordre considéré. Nous effectuons ce travail avec la vraie image et l'image reconstituée. En normalisant ces deux tableaux (en les divisant chacun par le 10e maximum et non pas le premier qui est certainement un cosmique), nous obtenons deux images qui ont la même allure avec une forte différence au niveau du cosmique.

Enfin, en calculant la déviation standard de cette différence, nous avons un seuil au delà duquel nous considérons que notre pixel est un cosmique. Pour avoir une certaine sécurité, ce seuil est égal à 5 fois la déviation standard. Lorsque que notre cosmique est repéré, nous prenons la valeur moyenne entre le pixels juste au dessus et juste en dessous. Nous répétons ensuite tout ce travail pour chaque ordre du fichier fits considéré.

Enfin, l'image traitée est stockée dans un nouveau fichier fits qui reprend le nom du fichier traité mais en changeant la terminaison.

Cet algorithme est un peu à part dans le pré traitement car il a besoin d'une matrice A.

7 Architecture de l'algorithme global

Si l'on en vient à parler des chemins, il nous faut forcément parler de l'architecture de l'algorithme à laquelle nous avons pensé pour l'instant pour que ce soit NETTEMENT plus clair.

A l'heure actuelle, le dossier DRS contient 5 sous dossiers :

- le dossier Brut. Ce dossier contient toutes les données de la nuit (les fichiers Thorium-Argon,Bias,Flat,Stellaire et Fabry-Perrot).
- le dossier DATA. Ce dossier contient toutes les données extérieures qui sont constantes quelque soit la nuit (le fichier "path.txt", l'atlas du Thorium-Argon...)
- le dossier FILES. Ce dossier contient tous les dossier fits que l'on crée.
- le dossier SRC. Ce dossier contient tous les fichiers python exécutables.
- le dossier TEMP. Ce dossier contient tous les fichiers pickle.

Mais tout décrire avec des mots serait insupportable. Du coup, voici en illustration le schéma de notre architecture réseaux. Les carrés avec une bulle orange sont les dossiers, les bulles vertes les algorithmes et le reste sont les noms des fichiers (pickles pour la plupart) qui sont créés.

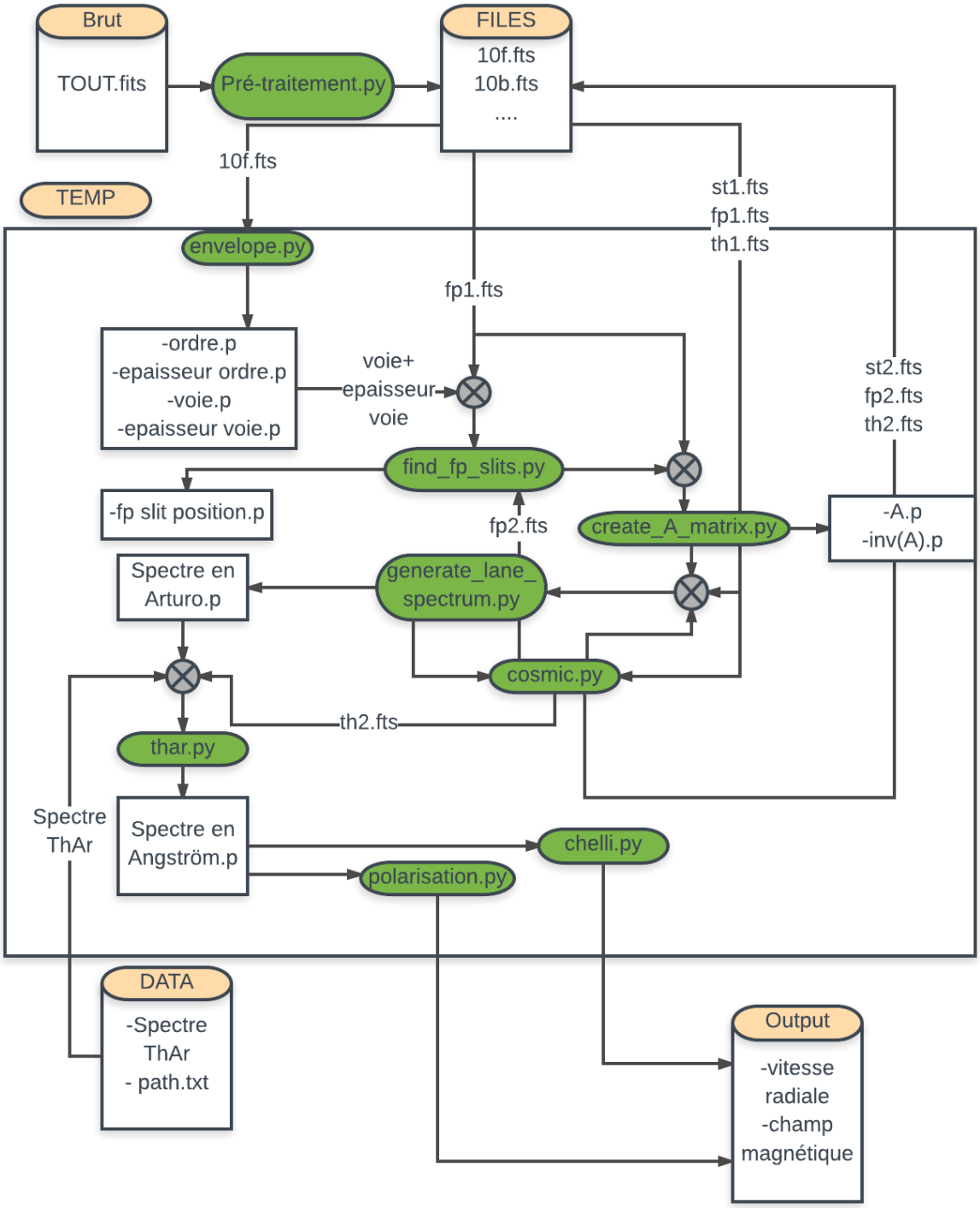

FIGURE 1 – architecture réseaux

Mais essayons de décrire quand même un peu ce que l'on voit. Tout d'abord, les fichiers de la nuit sont traités par les algorithmes de pré-traitement et crée les fichiers "10b.fts", "10f.fts", "st1.fts", "th1.fts" et "fp1.fts" qui sont stockés dans le dossier "FILES".

Ensuite, le fichier 10f est traité par l'algorithme "envelope" afin de localiser l'enveloppe des ordres et des voies. Une fois cette étape réalisée, l'algorithme "find_fp_slits" utilise l'enveloppe des voies, leur épaisseur et le fichier "fp1.fts" afin de localiser les raies du Fabry-Perrot sur les ordres. En utilisant ce résultat avec le fichier "fp1.fts", nous pouvons créer la matrice A (ou son inverse).

Cette matrice nous permet de créer "artificiellement" les spectres du Thorium-Argon, du Fabry-Perrot et de l'étoile via l'algorithme "generate_lane_spectrum". Ainsi, avec ces différentes images et en les comparant avec le fichier "th1.fts", "fp1.fts" et "st1.fts" respectivement, l'algorithme "cosmic" va traiter les cosmiques des différentes images. Cet algorithme crée les fichier "th2.fts", "fp2.fts" et "st2.fts" qui sont stockés dans le dossier "FILES". Le fichier "fp2.fts" est de nouveaux utilisé afin d'affiner la localisation des raies du Fabry-Perrot dans le but de créer une nouvelle matrice A débarrassée des cosmiques. Ces nouveaux fichiers fits et cette nouvelle matrice A sont à nouveau traités par "generate_lane_spectrum" pour créer des nouveaux spectres. Une fois que ces spectre sont jugés suffisamment "propres", ils sont traités avec le fichier "th2.fts" et la référence "Spectre ThAr" (Qui est un fichier fits du Thorium-Argon calibré par la lune) pour convertir nos Spectres des Arturo aux Ångströms.

Unes fois cette dernière étape réalisée, il est possible de traiter les données stellaires afin de récolter les informations souhaitées.