Projeto 1: Manual para o código pipeline

Matheus Mendes Silva Barboza

Maio, 2018

1 Introdução

Antes de entrar em detalhes sobre o código, é importante entender o que ele faz exatamente.

O objetivo é usar o código como um pipeline, possibilitando a redução de imagens de ciência.

O pipeline é uma estrutura de processos, são sequências de passos pelos quais os dados observados passam para serem corrigidos de vários efeitos (ruídos).

Para a redução de dados, geralmente é suficiente usar o IRAF e pacotes de programação, pois qualquer processo de redução pode ser realizado por linguagens de programação de alta performance (neste caso, o Python está sendo utilizado).

Porém, alguns instrumentos possuem características próprias que devem ser corrigidas (correções óticas, por exemplo) e dependendo da ciência envolvida, podem existir passos adicionais (ao trabalhar com uma fonte não-pontual, por exemplo).

Então, cada telescópio/instrumento costuma possuir o seu próprio pipeline, o que facilita bastante o trabalho de redução.

Quando não há pipeline disponível, é necessário seguir um procedimento padrão de redução em alguma linguagem de programação de alto nível. Para isso é necessário identificar inicialmente quais ruídos devemos corrigir.

2 O bias e o flat

O procedimento padrão consiste em subtrair uma constante positiva que leva em conta a voltagem que existe no CCD, essa informação está salva nos bias (ruído devido à voltagem inicial).

Então, o bias é a imagem que representa a corrente de fundo. Ele deve ser subtraído de todas as outras imagens, tanto as de calibração, quanto as de ciência.

A resposta do CCD não é espacialmente uniforme (distribuições não são descritas igualmente), então uma fonte de luz é usada para iluminar o CCD, permitindo estimar a eficiência espacial. Essas informações estão contidas nos

flats. As imagens de ciência precisam ser divididas pelos flats (que funcionam como imagens de calibração) para que a correção possa ser realizada.

Então, flat representa a discrepância de eficiência pixel-a-pixel. Antes de aplicar a correção por flat às imagens de ciência, é necessário corrigí-lo do bias. Após corrigir o flat, dividimos a imagem de ciência pelo flat, o que significa corrigir o efeito de diferença de eficiência espacial do CCD. Porém, antes da divisão, o flat precisa ser normalizado para que as contagens finais de cada distribuição não sejam modificadas.

Depois que as imagens de ciência passam pelo pipeline, o resultado é a obtenção de imagens de ciência corrigidas de efeitos indesejáveis (ruídos).

3 O repositório

O código pode ser obtido a partir de um repositório no GitHub (plataforma de hospedagem), a partir deste link: https://github.com/MatheusMendesSB/TdDProjeto1

Além do código, também está disponível um exemplo, para que o código possa ser testado, este manual, a licença, o README, os dois arquivos com o material utilizado como referência para o desenvolvimento do projeto e uma pasta vazia.

Para entender o motivo de existir uma pasta vazia no repositório (ImagensTeste), leia a seção "O exemplo" deste manual.

4 O código

O arquivo contendo o código principal está nomeado como "projeto1.py".

O código deve ser usado como um pipeline para o processo de redução, ele corrige as imagens de ciência do bias e do flat.

Ele realiza todo o processo automaticamente, mas o usuário precisa passar duas informações: o caminho do diretório no qual as imagens (bias, flat e de ciência) encontram-se e o nome utilizado para identificar as imagens de ciência. Quando o usuário tenta rodar o código, ele já começa pedindo essas duas informações.

Quando o código recebe o caminho para as imagens, ele cria três listas contendo as imagens de bias, flat e ciência (uma lista para cada tipo de imagem).

Essas listas são utilizadas dentro das diversas funções presentes no código.

O pipeline começa com a criação do Master Bias, que corresponde à combinação de todas as imagens de bias (combinação pela mediana). O próximo passo é usar o Master Bias para corrigir cada imagem de flat individualmente. Após corrigir as imagens de flat, elas são normalizadas (cada imagem de flat é individualmente dividida pela média entre todas as imagens de flat), depois disso elas são combinadas de forma análoga às imagens de bias (pela mediana) e com isso, o código cria o Master Flat.

Após a criação das "imagens master", o código finalmente corrige as imagens de ciência: elas são subtraídas do Master Bias e divididas pelo Master Flat.

A lista de matrizes resultante de todo o processo de correção pode ser impressa na tela, mas isso não é feito por "default", é necessário remover o sinal de comentário da linha: print(sci-flatbias).

O código salva uma imagem FITS para cada imagem corrigida, por "default".

Um simples teste de estatística também é realizado por default, o código imprime na tela se o Master Bias e o Master Flat passaram (ou não) no teste. Como a média do bias fica em torno de 20 e a média do flat fica em torno de 1, o critério utilizado para o Master Bias e o Master Flat passarem no teste foi verificar se suas médias estão dentro do intervalo 0-30 e 0-2, respectivamente.

Depois disso o código é finalizado, mas algumas questões importantes precisam ser levadas em conta:

- As imagens de bias, precisam ser nomeadas como bias*.fits, o asterisco significa concatenação, ou seja, após a palavra "bias" qualquer coisa pode ser escrita.
- As imagens de flat, precisam ser nomeadas como flat*.fits, novamente o asterisco indica concatenação.

5 O exemplo

O arquivo contendo o código de exemplo está nomeado como "exemplo.py".

Junto ao código principal, um exemplo também é disponibilizado no repositório.

O exemplo é útil para que o usuário possa testar o código antes de tentar corrigir suas próprias imagens.

No exemplo, um caminho preestabelecido aponta para um diretório já existente no repositório do GitHub. O nome das imagens também já é conhecido pelo código.

Então o usuário não precisa fornecer nenhuma informação, basta rodar o código.

Porém, as imagens não estão presentes no repositório, por se tratar de um volume grande de dados, elas precisam ser obtidas separadamente.

O código de teste também salva as imagens FITS por default e faz os testes estatísticos para o Master Bias e Master Flat, mas para que ele imprima o resultado na tela, também é necessário apagar o comentário da linha: print(sci-flatbias).

Basta baixar as imagens e introduzí-las no diretório "ImagensTeste" antes de rodar o código.

Link para o download das imagens de teste:
https://www.dropbox.com/s/4u5qmtv1b154tzz/xo2b.zip?dl=0

6 Referências

As referências utilizadas na construção do código foram os materiais disponibilizados em sala de aula.

Eles também estão disponíveis no repositório do GitHub.

O PDF "OLV474-0004" contém toda a discussão sobre pipelines e o arquivo HTML "OVL474 – Astropy and FITS" contem informações quanto à manipulação de arquivos FITS dentro do Python.