TRATAMENTO DE DADOS

Construção de Uma Pipeline Pardrão IRAF

Douglas Brambila dos Santos

Prof: Walter Martins

2º Semestre 2017

1- Introdução

Para se realizar uma análise em astronomia é necessário fazer uma observação do objeto astronômica do interesee. Essa observação é produzida via telescópio, seja ele nas múltiplas faixas de observação disponíveis. Ao se produzir uma imagem óptica é necessário tomar algumas precauções antes de se realizar fotometria e espectroscopia devem ser tomadas.

Todas observações sofrem efeitos do ambiente que o telescópio, dos corpos celestes próximos ao objeto ou campo de desejo, do próprio brilho de fundo do céu e até mesmo dos instrumentos utilizados para fazer a observação. Os efeitos mais comuns e os que iremos tratar nesse projeto são o Bias, o Flat Field e o Background.

Bias é uma imagem com tempo de exposição zero e com telescópio fechado. Durante o processo de leitura dos fótons captados, alguns ruídos são adicionados durante a conversão para dados digitais. Essa exposição fechada nos dá exatamente esse ruído, que posteriormente deve ser retirado das imagens pois acrescenta sinais inexistentes no objeto.

Em um CCD ideal todos o pixeis que o compõe tem uma função de resposta a exposição luminosa da mesma maneira. No entanto, em um CCD real, temos que tal idealização não acontece, de modo que é necessário realizar uma observação à uma fonte de luz conhecida de modo a calcular a resposta de cada píxel frente a essa exposição luminosa. Essa exposição geralmente é feita observando-se um alvo iluminado uniformemente a um uma distância bem próxima, ou observando-se o céu muito perto do anoitecer/amanhecer. Essa resposta do CCD frente a esta exposição é o chamado Flat Field.

No ponto mais escuro do céu, mesmo onde não se encontrar nenhuma estrela o CDD realizará contagens. Isso porque mesmo nessas regiões escuras existe emissão do céu de fundo e de luz difusa ao longo do mesmo. Desse modo é preciso remover essa contribuição ao longo da imagem para que quando se for realizar a fotometria e/ou espectroscopia se esteja computando apenas a luz do objeto. Essa correção recebe o nome de Sky ou Background.

A maneira mais propagada de se realizar essas correções é chamada de Procedimento IRAF. Recebe esse nome pois são realizadas na mesma ordem que o programa de tratamento de imagem IRAF realiza. O procedimento IRAF consiste em seguir os seguintes passos:

1º: Corrigir o Bias de todas as imagens, tanto as de ciência quanto as imagens do Flat Field;

2º: Corrigir o Flat Field (já corrigido para Bias) das imagens de ciência (já corrigidas para Bias);

3º: Remover o background das imagens de ciência.

2- Bias

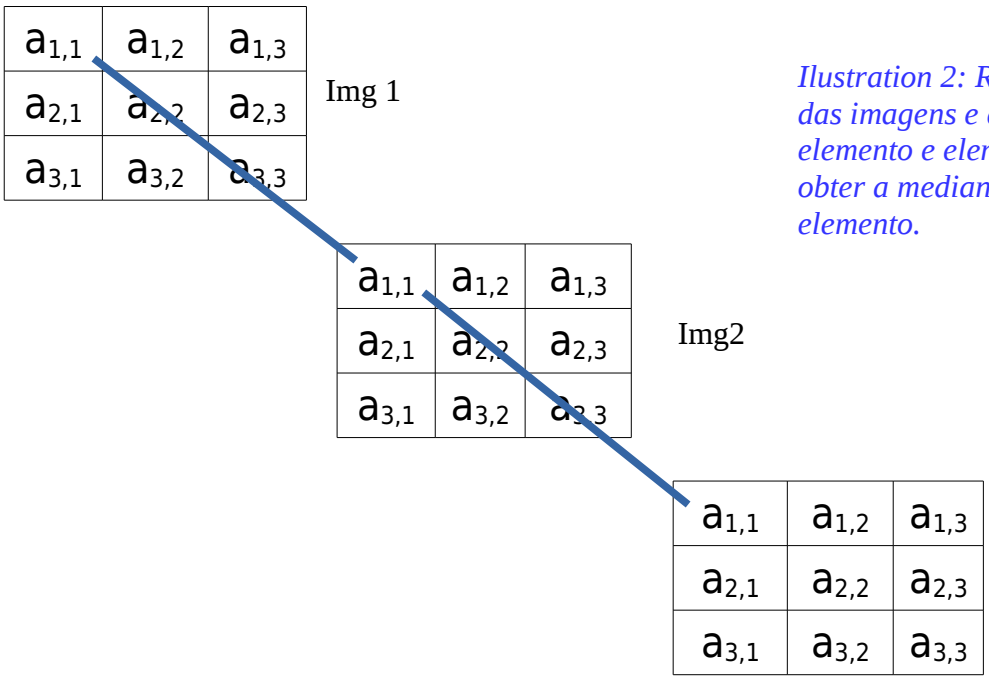
Os ruídos de bias não são constantes, podendo aumentar ou diminuir de observação para observação. De modo que, para se ter um valor confiável do quanto esse ruído esta influenciando em nosso sinal fazemos uma mediana dos valores de cada píxel. Usamos uma mediana pois essa estatística é menos influenciada por valores extremos, fornecendo assim um valor mais confiável.

Em nosso caso de exemplo temos disponíveis 10 imagens bias. As imagens bias devem ser produzidas filtro a filtro, e é necessário que se tenha bastantes imagens para que seja possível realizar uma estatística confiável.

```
3  def bias_median(path, numbers, filtro):
4      bias_images = []
5      bias_header = []
6      for i in range (numbers):
7          a=str(i+1)
8          if i+1 >= 10:
9              bias_path = path + 'bias.%s.00%s.fits' %(filtro, str(a))
10         else:
11             bias_path = path + 'bias.%s.000%s.fits' %(filtro, str(a))
12         img, hdr = fits.getdata(bias_path, header=True)
13         img = np.array(img, dtype='Float64')
14         bias_images.append(img)
15         bias_header.append(hdr)
16     median = np.median(bias_images, axis=0)
17     return median
```

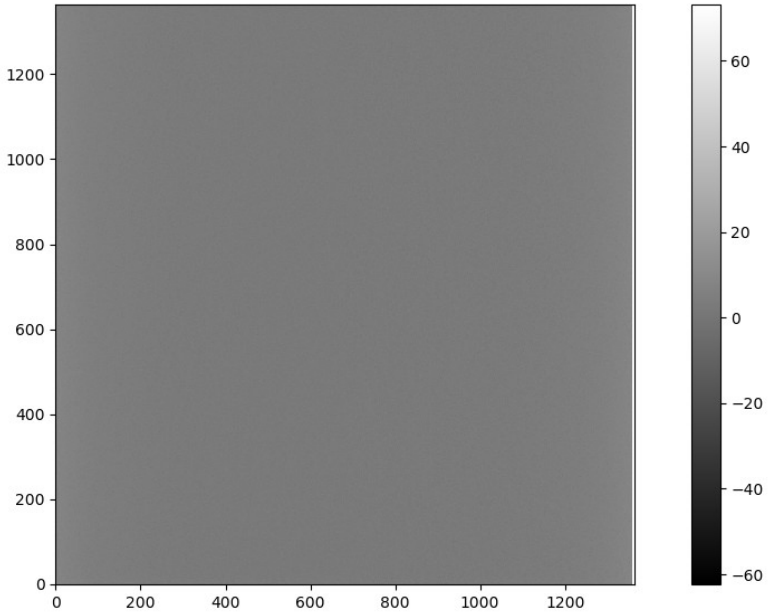
Illustration 1: Código responsável por abrir as imagens bias e calcular a imagem mediana que será utilizada para correção do bias. Para um código mais detalhado veja o arquivo python referente a correção bias (bias.py).

Mas como essa imagem mediana é tomada? O comando *np.median* da linha 16 na imagem acima é responsável por produzir essa imagem. Para entender como isso é feito primeiro é necessário entender como uma imagem digital é representada. Quando se abre uma imagem FITS ela sera representada por uma matriz, onde cada elemento representa um píxel e sua contagem. O comando *np.median(bias_imagen, axis=0)* irá fazer uma comparação elemento por elemento nas múltiplas matrizes de imagem bias, isto é, ele ira compara os elementos a_{ij} da imagem 1 com os elementos a_{ij} das demais imagens e criará uma nova matriz com as mesmas dimensões da imagem, mas agora contendo apenas os valores medianos a_{ij}.



Ilustration 2: Representação das imagens e a combinação elemento e elemento para se obter a mediana de cada elemento.

Feito isso, temos nossa matriz de bias:



Ilustration 3: Ao lado o plot da imagem de resultante do bias e abaixo a matriz referente a essa imagem.

```
array([[ 286.88000488,  285.87915039,  288.06213379, ...,  315.57910156,
         316.58032227,  316.63049316],
       [ 285.38061523,  285.38134766,  286.38110352, ...,  311.60235596,
         314.52593994,  314.47576904],
       [ 285.87982178,  286.05993652,  287.06103516, ...,  316.52642822,
         314.57836914,  314.62750244],
       ...,
       [ 285.05438232,  287.05535889,  287.05462646, ...,  312.50134277,
         315.63708496,  315.71685791],
       [ 288.60974121,  285.61022949,  288.05279541, ...,  313.49945068,
         314.63500977,  314.71636963],
       [ 284.61010742,  284.05200195,  284.60986328, ...,  313.63592529,
         313.63568115,  313.71594238]])
```

O próximo passo é utilizar essa matriz para corrigir o bias de todas as imagens.

```
20  def correcao_flat_bias(bcaminho, fcaminho, bnumero, fnumero, filtro):
21      images_flat = abrindo_flat(fcaminho, fnumero, filtro)
22      bflat = []
23      for i in range(len(images_flat[0])):
24          correcao = images_flat[0][i] - bias_median(bcaminho, bnumero, filtro)
25          bflat.append(correcao)
26      return bflat
```

Illustration 4: Código responsável por remover o bias das imagens de flat. Para mais detalhes do código ver o arquivo python referente a correção das imagens flat (flat.py).

```
31  def bimagens(bcaminho, icaminho, campo, bnumero, filtro):
32      bciencia = []
33      imagen_ciencia = sky.abrindo_imagens(icaminho,campo)
34      imagen_bias = bias.bias_median(bcaminho,bnumero, filtro)
35      for i in range(len(imagen_ciencia)):
36          img = imagen_ciencia[i] - imagen_bias
37          bciencia.append(img)
38      return bciencia
```

Illustration 5: Código responsável por remover o bias das imagens de ciência. Para mais detalhes do código ver o arquivo python referente a correção das imagens de ciência (correcoes.py).

2- Flat Field

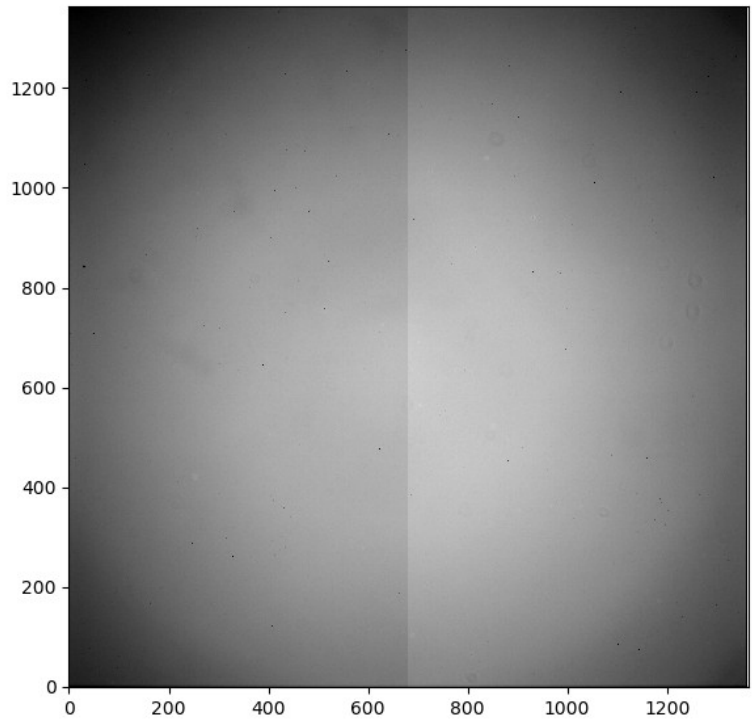
Como dito anteriormente, o segundo passo a se tomar e corrigir o Flat Field das imagens de ciência. A correção flat é feita em três etapas:

1º: Subtrai-se a matriz mediana de bias de todas as imagens de flat field (esse passo já foi feito).

2º: Normaliza-se todas as imagens flats que se tem. Isso é feito dividindo-se as imagens imagens de flat (já corrigidas pelo bias) e dividindo cada imagem pela média dos elementos de suas matriz.

3º: Toma-se a mediana das matrizes elemento por elemento (mesmo procedimento explicado par ao bias).

Tem-se em mão a imagem final do flat, normalizada e com valores medianos.



Ilustration 6: Ao lado o plot da imagem de resultante de flat e abaixo a matriz referente a essa imagem.

```
array([[ 1.09279282,  0.31396411,  0.31967505, ...,  0.33771913,
         1.26309926,  1.26349917],
       [ 0.91083355,  0.22461917,  0.22878924, ...,  0.22547107,
         1.11600917,  1.11635242],
       [ 0.92580824,  0.25487242,  0.25564955, ...,  0.25848462,
         1.20281123,  1.2031652 ],
       ...,
       [ 0.98344795,  0.26192626,  0.26262143, ...,  0.26890419,
         1.1826589 ,  1.182935  ],
       [ 0.96007433,  0.26243451,  0.26279355, ...,  0.27348994,
         1.16273197,  1.16299888],
```

```
 [ 0.91005165,  0.26442122,  0.26288552, ...,  0.27018231,
   1.07904405,  1.07929528]])
```

Tendo essa informação é possível corrigir o Flat Field das imagens de ciência. O Flat é corrigido como uma espécie de normalização da imagem de ciência pelas imagem de flat (mediana das imagens flat normalizadas), ou seja, divide-se cada uma das imagens de ciência pela imagem resultante flat. Isso é feito pelo seguinte código:

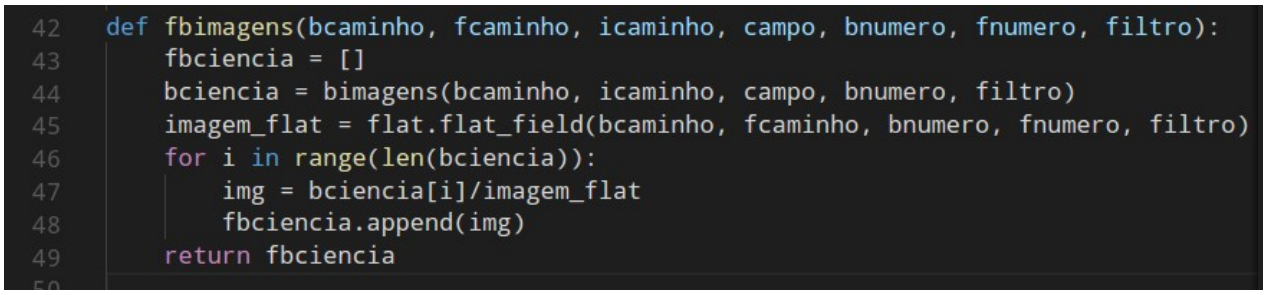


Illustration 6: Código responsável por corrigir o flat field das imagens de ciência. Para mais detalhes do código ver o arquivo python referente a correção das imagens de ciência (correcoes.py).

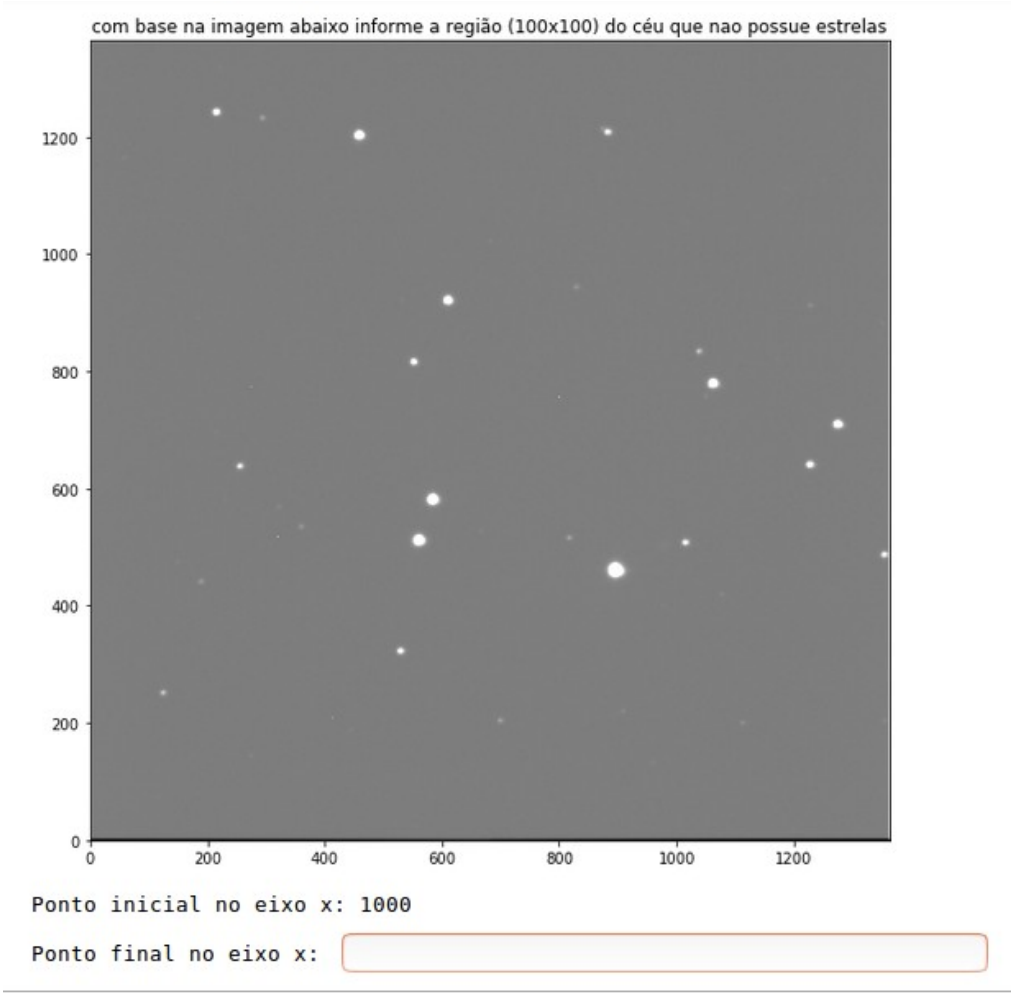
3- Sky ou Background

A correção de Backgroud é a última correção a ser feita antes da imagem estar pronta para ser utilizada para se fazer ciência. O primeiro passo a ser tomado é determinar uma região de 100x100 do céu onde não se tem estrelas para se calcular o brilho de fundo. Isso é feito pelo seguinte código:

```
51  def imagem_sky(bcaminho,fcaminho,icaminho,campo,bnumero,fnumero,filtro):
52      imagem = abrindo_imagens(icaminho, campo)[0]
53      plt.figure()
54      fig, eixos = plt.subplots(nrows=1, ncols=1, figsize=(15,10))
55      grafico = plt.imshow(imagem,vmin=np.mean(imagem)-2.5*np.std(imagem),
56          vmax=np.mean(imagem)+2.5*np.std(imagem),cmap=plt.cm.gray,origin='lower')
57      plt.title("com base na imagem abaixo informe a região (100x100) do céu que nao possue estrelas")
58      plt.show()
59      xi = int(input("Ponto inicial no eixo x: "))
60      xf = int(input("Ponto final no eixo x: "))
61      yi = int(input("Ponto inicial no eixo y: "))
62      yf = int(input("Ponto final no eixo y: "))
63      regiao = imagem[xi:xf, yi:yf]
64      sky_imagem = np.random.poisson(np.mean(regiao), imagem.shape)
65      return sky_imagem
66
```

Illustration 7: Código responsável por definir a região de céu.Para mais detalhes do código ver o arquivo python referente a abertura e produção da imagem sky (sky.py).

A região no céu é definida pelo usuário, onde ele fornece os valores iniciais e finais de cada um dos eixos (operação realizada pelo código da ilustração 7):



Com os limites estabelecidos pegasse da imagem apenas a região indicada pelos limites do usuário. Dessa região tirasse a médias dos elementos. Assumindo-se que a background tenha uma distribuição poissonica com média igual a médias dos elementos dentro da região indicada pelo usuário. Essa nova matriz aleatória e poissonica devem ter dimensões iguais a imagens de ciência.

Tendo essa matriz, basta subtraí-las das imagens que já foram corrigidas para bias e flat. Operação feita pelo código abaixo:

```
68  def sfbimagens(bcaminho, fcaminho, icaminho, campo, bnumero, fnumero, filtro):
69      sfbciencia = []
70      fbciencia = fbimagens(bcaminho, fcaminho, icaminho, campo, bnumero, fnumero, filtro)
71      imagem_sky = sky.imagem_sky(bcaminho,fcaminho,icaminho,campo,bnumero,fnumero,filtro)
72      for i in range(len(fbciencia)):
73          img = fbciencia[i] - imagem_sky
74          sfbciencia.append(img)
75      return sfbciencia
```

Illustration 8: Código responsável por subtrair o background da imagem de ciência (já corrigida para bias e flat).Para mais detalhes do código ver o arquivo python referente a correções (correcoes.py).

Temos então a imagem final pronta para ser utilizada para fotometria e/ou espectroscopia.

Para mostrar o resultado das correções, temos abaixo uma sequencia de imagem que mostra a imagem original, a imagem após todas as correções e por fim a subtração entre a imagem inicial e final e suas respectivas matrizes.

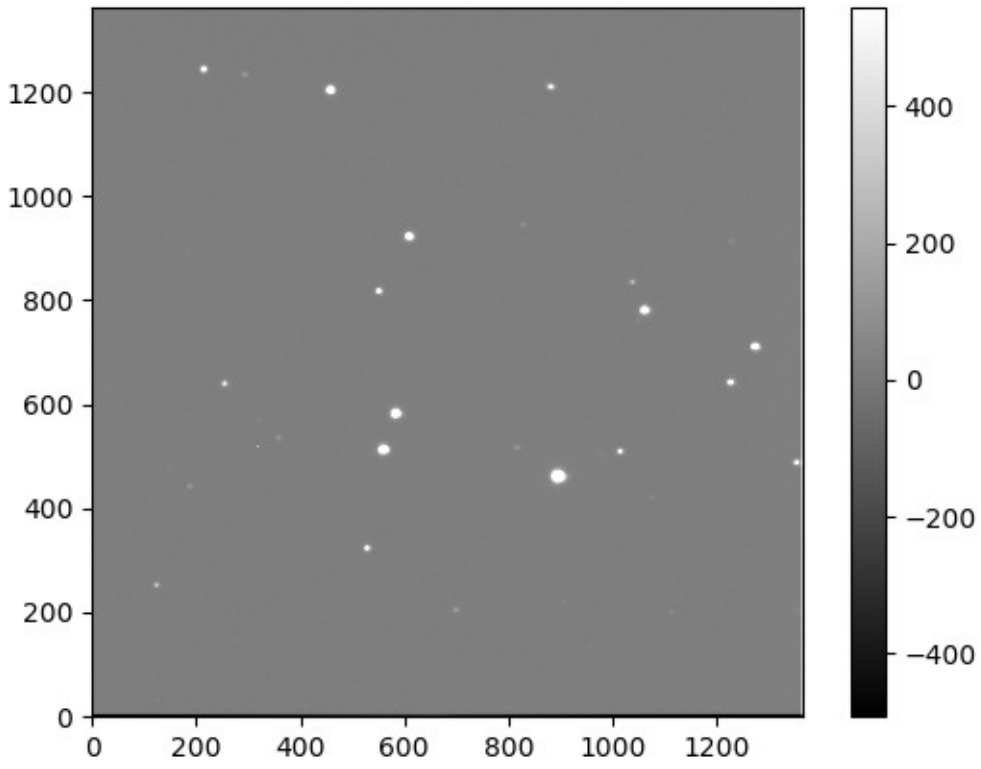


Illustration 9: Imagem inicial

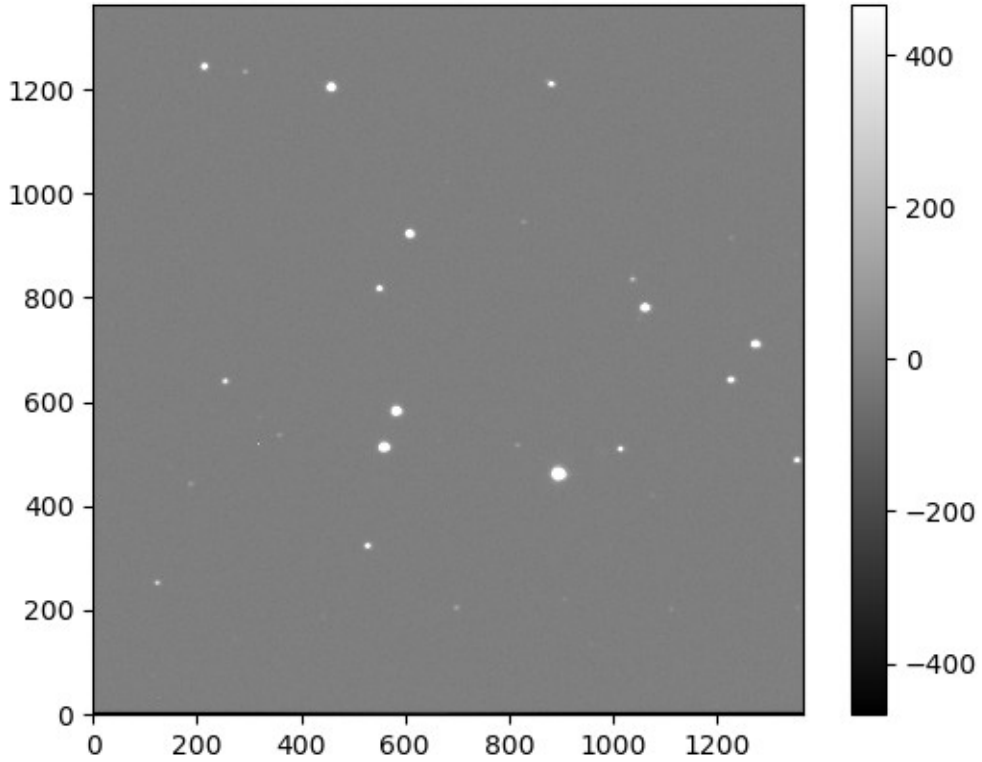


Illustration 10: Imagem final

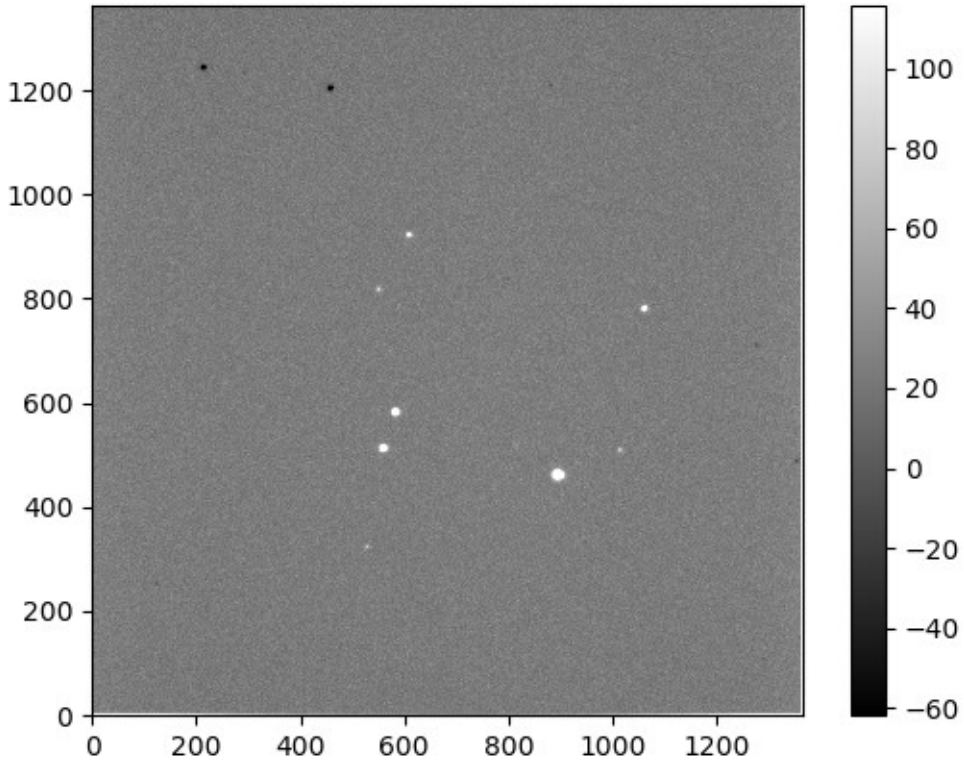


Illustration 11: Resto da subtração da imagem inicial e final

Matriz da imagem inicial:

```
array([[-765.82775879, -776.82568359, -772.82727051, ..., -913.81256104,
        -905.81542969, -904.95697021],
       [-222.00598145, -234.00158691, -229.00134277, ..., -288.96972656,
        -271.97283936, -270.98150635],
       [  43.90002441,   25.90625   ,   28.90625   , ...,   13.94671631,
          37.9420166 ,   38.99414062],
       ...,
       [ 301.59155273,  291.59460449,  288.59521484, ...,  318.31567383,
         330.31311035,  330.41772461],
       [ 302.59320068,  286.59820557,  292.59735107, ...,  317.32049561,
         336.31634521,  336.42315674],
       [ 308.59716797,  291.60168457,  289.60058594, ...,  316.32281494,
         333.31842041,  333.42858887]])
```

Matriz da imagem final:

```
array([[ -9.82516063e+02,  -3.41892536e+03,  -3.33975616e+03, ...,
         -3.65793149e+03,  -9.84456266e+02,  -9.89532773e+02],
       [ -5.83707417e+02,  -2.34528824e+03,  -2.27314191e+03, ...,
         -2.69218410e+03,  -5.58114376e+02,  -5.46158691e+02],
       [ -2.82492281e+02,  -1.04305133e+03,  -1.02788754e+03, ...,
         -1.19197555e+03,  -2.51883436e+02,  -2.50961885e+02],
       ...,
       [  2.39167722e+00,  -5.20086136e-01,  -1.73580298e+01, ...,
          6.88643515e-01,  -1.12836081e+01,  -1.32183657e+01],
       [ -7.44272112e+00,  -2.40169250e+01,   5.05566004e+00, ...,
         -1.52747619e+01,  -2.02087700e-01,  -7.12723458e+00],
       [  8.16212607e+00,  -9.85841185e+00,   7.44436291e-02, ...,
         -1.71797070e+01,  -9.68058467e+00,  -1.06629073e+01]])
```

Matris do resto da subtração entre as duas imagens:

```
array([[  216.6883042 ,  2642.09967486,  2566.92889437, ...,
         2744.11892426,    78.64083666,    84.57580251],
       [  361.70143587,  2111.28664952,  2044.14056627, ...,
         2403.21437743,   286.1415362 ,   275.17718436],
       [  326.39230507,  1068.95758262,  1056.7937896 , ...,
         1205.92226821,   289.82545302,   289.95602587],
       ...,
       [  299.19987551,   292.11469063,   305.95324463, ...,
          317.62703031,   341.59671842,   343.63609034],
       [  310.03592181,   310.61513059,   287.54169103, ...,
          332.59525752,   336.51843292,   343.55039132],
       [  300.4350419 ,   301.46009642,   289.52614231, ...,
          333.50252191,   342.99900508,   344.09149612]])
```

4- Bibliográfia

http://www.qsimaging.com/ccd_noise_measure.html

http://www.cfht.hawaii.edu/~baril/Pyxis/Help/flatdarkfield.html

https://waltersmartinsf.github.io/TDAclass/