Reimagining the noise reduction algorithm

Amaro Jr., Rafael A. N. and Igor Abritta

28/10/2020

Universidade Federal de Juiz de Fora (UFJF)
Juiz de Fora, MG

Summary

- Motivation;
- Thought solution;
- The new algorithm;
- Results;
- Conclusion

Motivation

- As commented last meeting, pre-processing could be a more cumbersome process than clusterization itself;
- We’ve found that the noise reduction algorithm was the bottleneck for full resolution images (45~60s in google colab);
    - This part is important because it reduces the number of elements to be sent to the clustering algorithm;
- Then we’ve searched for faster methods applied to noise reduction.

Thought solution

- The original process is done through a recursive removal of noise pixels, being necessary to sweep every pixel while removing noise.
- A similar process could be done with a convolution of the whole image with a unit kernel of size 3.

The new algorithm

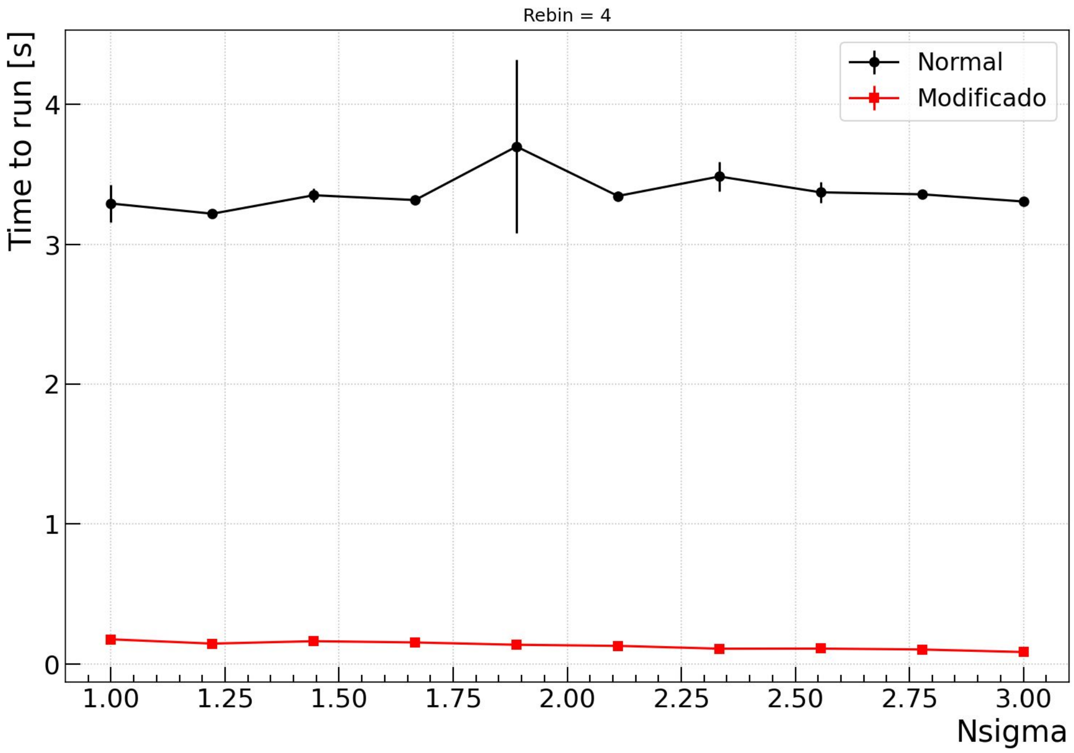
UNIVERSIDADE FEDERAL DE JUIZ DE FORA

- As the Noise_reductor is a recursive process and the convolution is done for the whole image, to get a similar result we should filter more than once;
- We used nsigma as a parameter to set the number of recursions;
- All tests were done for the rebined image, as both filters were calibrated for it.

The new algorithm

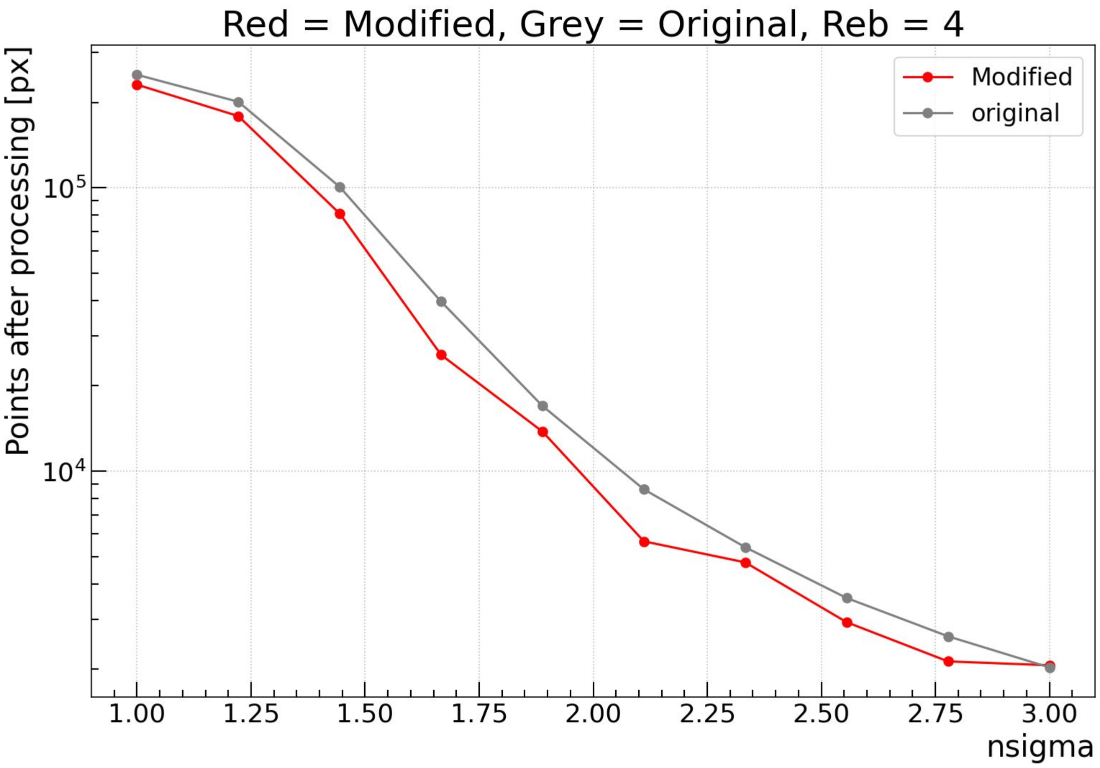
```python
def noisereductor_mod(edges,min_energy=0.35,sigma=3):
    tpx = 10
    for x in range(int(10 - 1.8*sigma)):

        mpx = uniform_filter(edges,size=3,mode='constant')
        mpx = (mpx*9.-edges)/8.
        edges[np.abs(edges - mpx) > tpx] = mpx[np.abs(edges - mpx) > tpx]
        mpx = uniform_filter(edges,size=3,mode='constant')
        mpx = (mpx*9.-edges)/8.
        neighbours = np.copy(edges)
        neighbours[neighbours>0] = 1
        neighbours = uniform_filter(neighbours,size=3,mode='constant')*9

        edges[neighbours < x] = 0
        edges[mpx < (min_energy + x/70 )] = 0

    return edges
```

Results



As time was our biggest concern, our first look will be on runtime of each process.

Results



Then we verified the number of points remaining for each process.

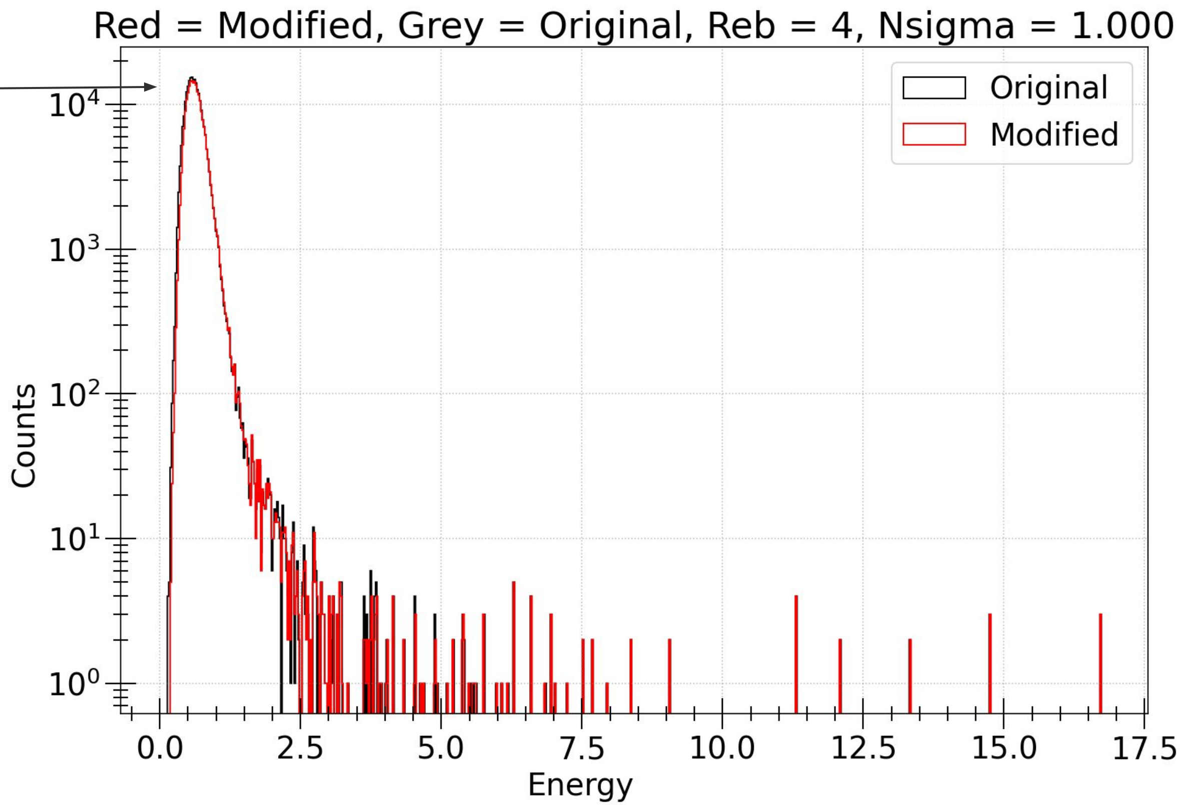
Results

UNIVERSIDADE FEDERAL DE JUIZ DE FORA

- Then we’ve looked at the energy histogram of the remaining points to compare both processes in a more measurable way.

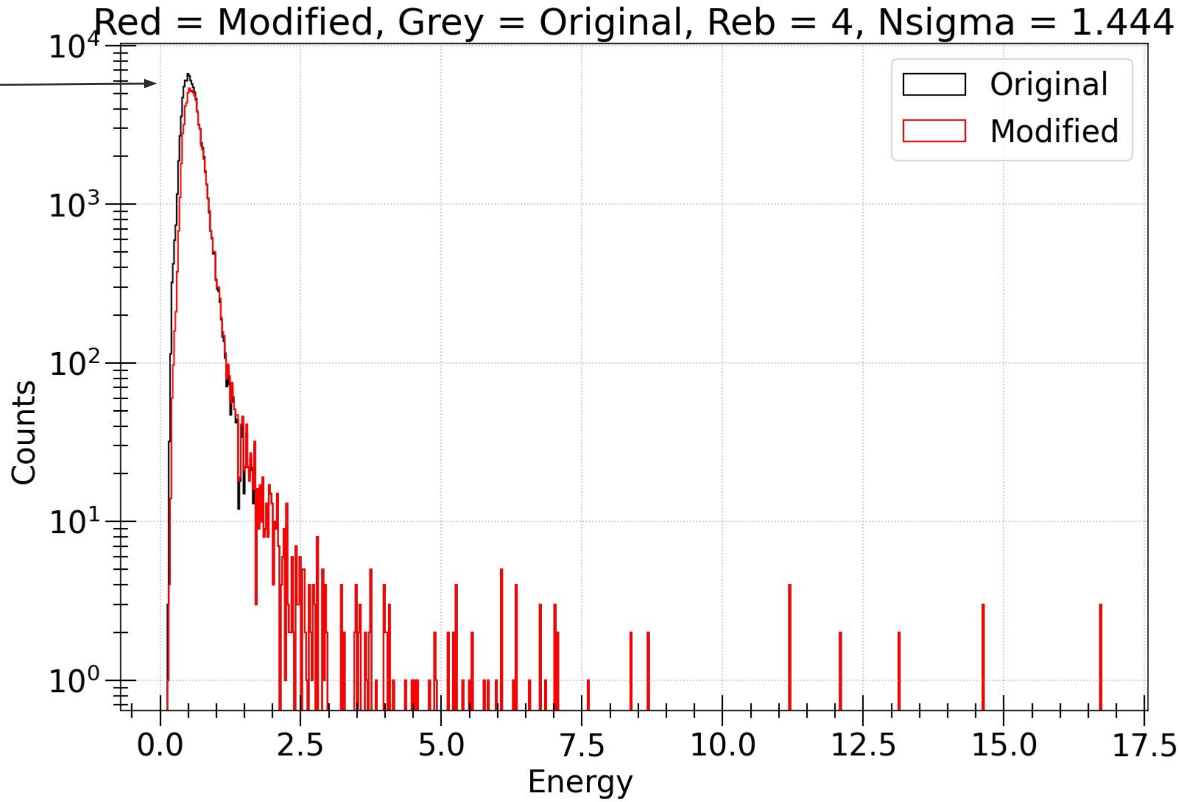
Results - Energy distribution (*nsigma* = 1)

The biggest difference seems to be at low energies



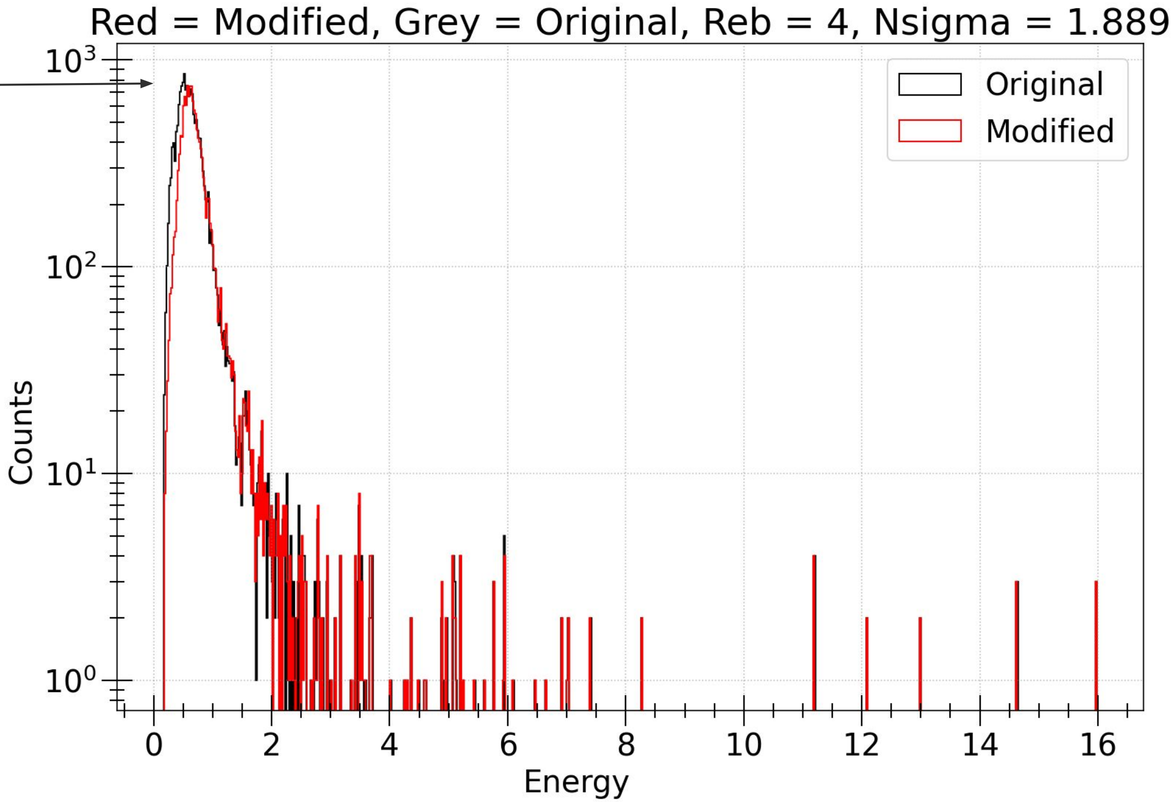
Results - Energy distribution (*nsigma* = 1.44)

The biggest difference seems to be at low energies



Results - Energy distribution ($nsigma$ = 1.89)

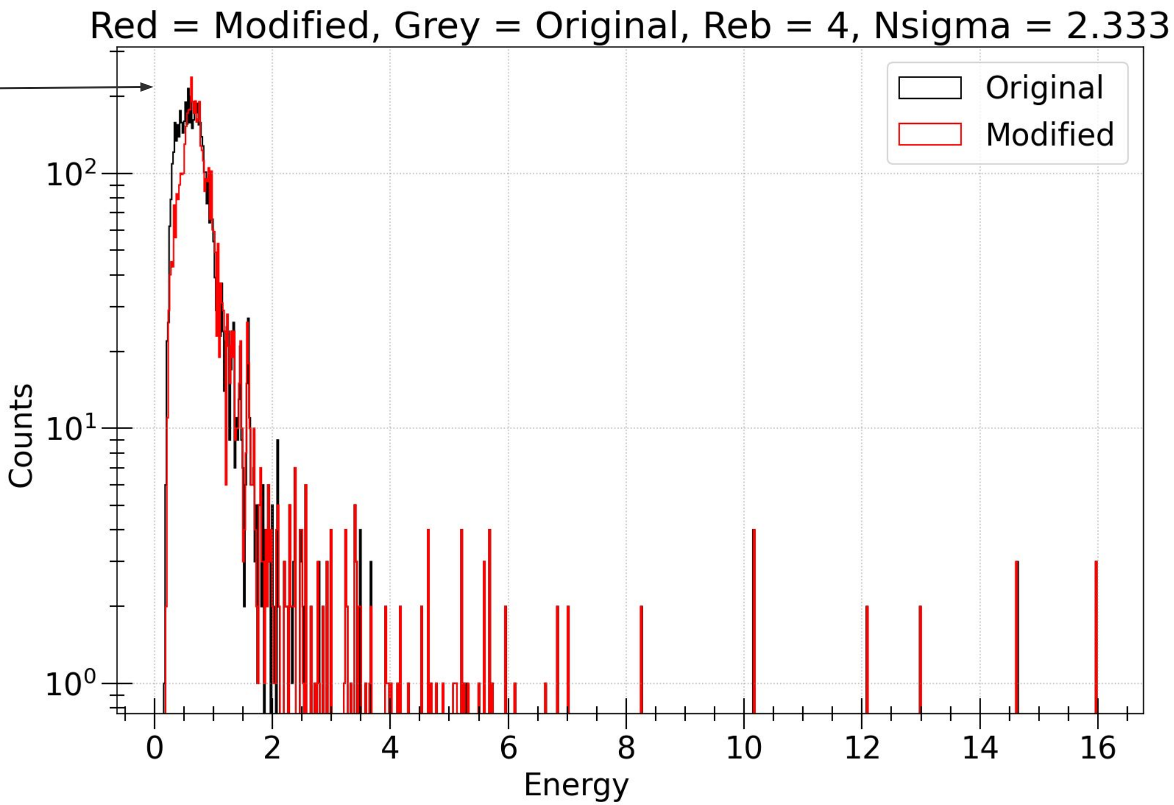
The biggest difference seems to be at low energies



Results - Energy distribution ($nsigma = 2.33$)

UNIVERSIDADE FEDERAL DE JUIZ DE FORA

The biggest difference seems to be at low energies

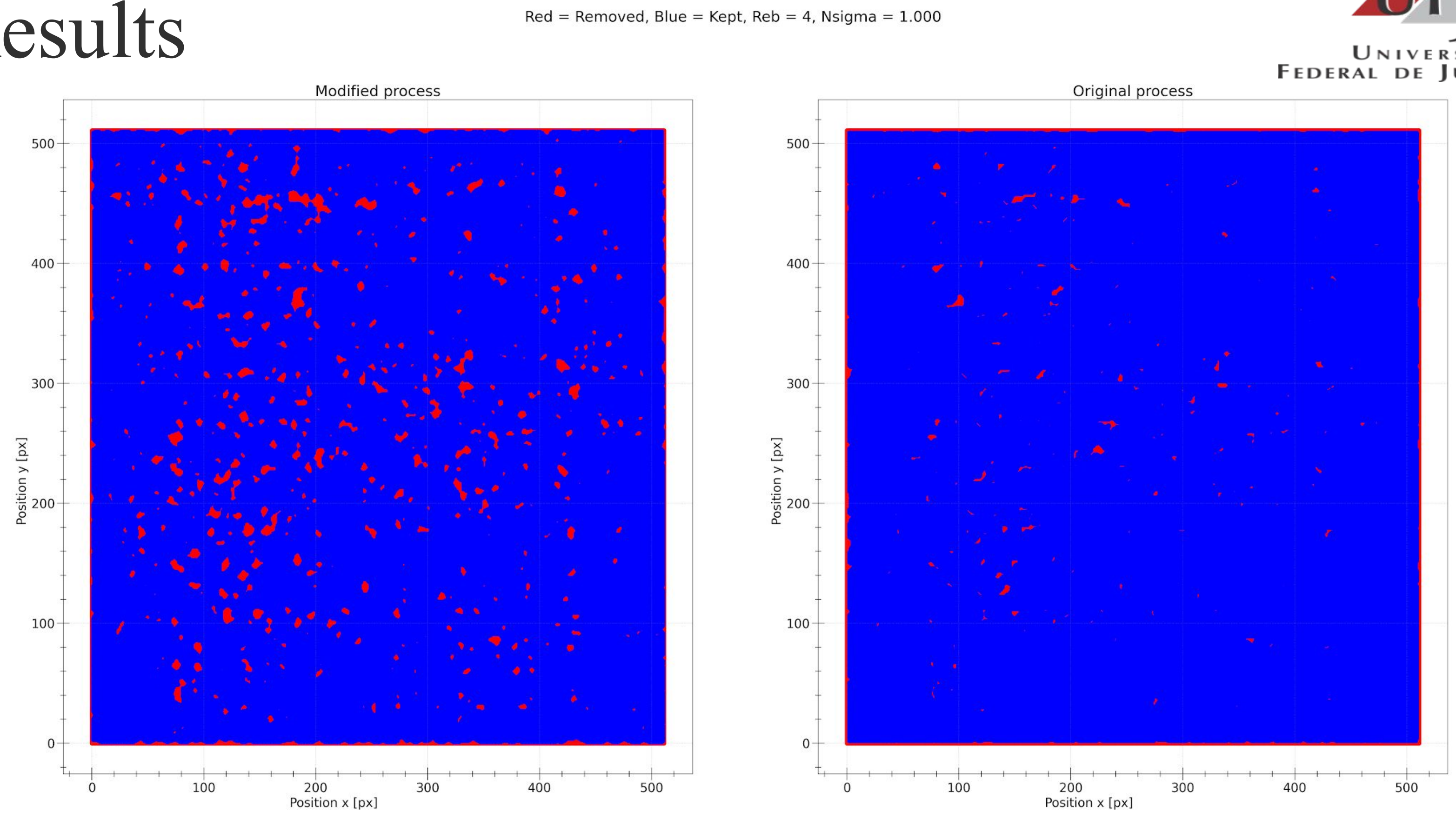


Results

UNIVERSIDADE FEDERAL DE JUIZ DE FORA

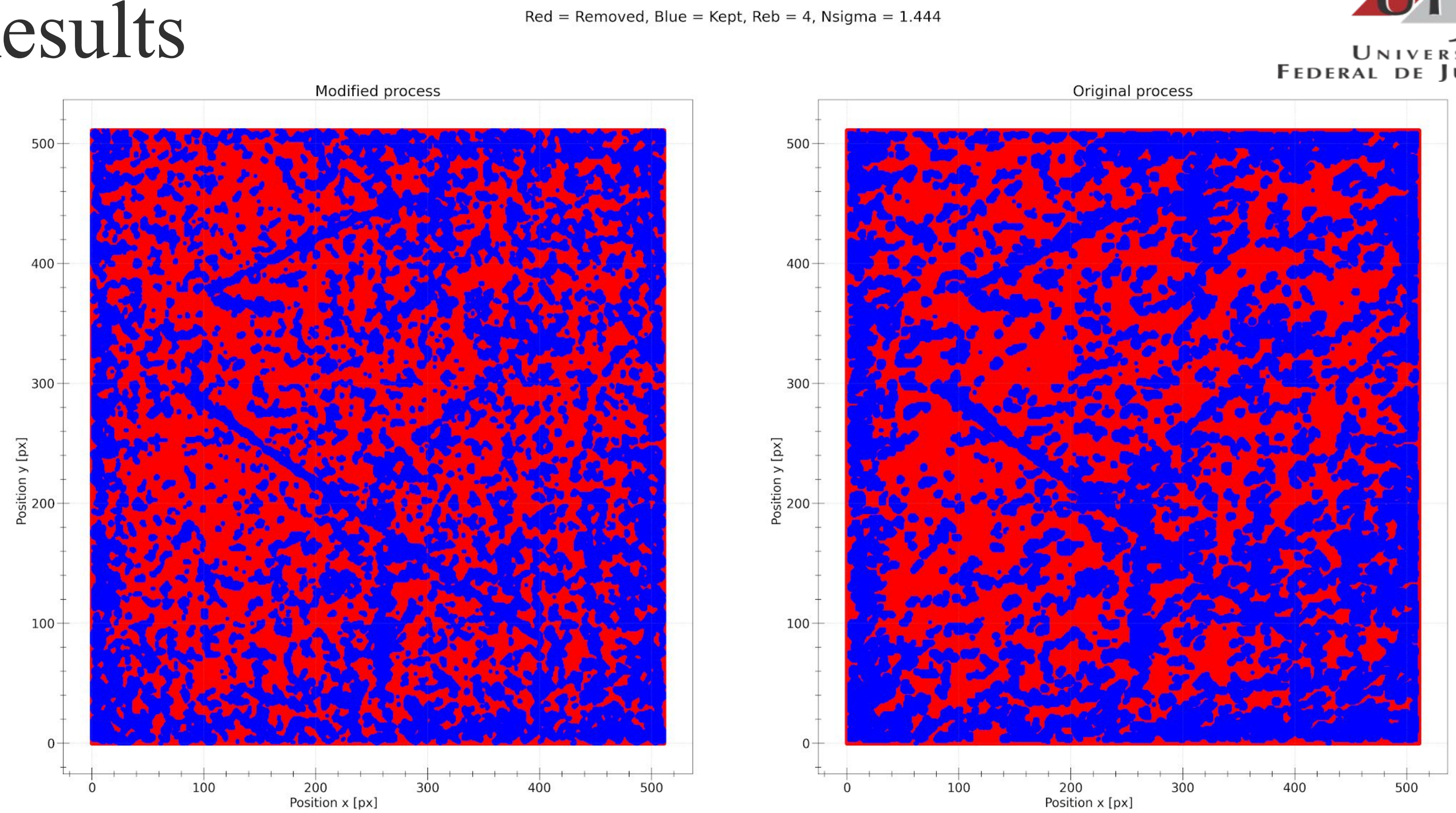
- Then we verified which points were being removed by each process;
- The next slides will show a scatter plot for removed points during the noise reduction algorithm;
- Red represents removed points while blue shows which points were kept.

Results



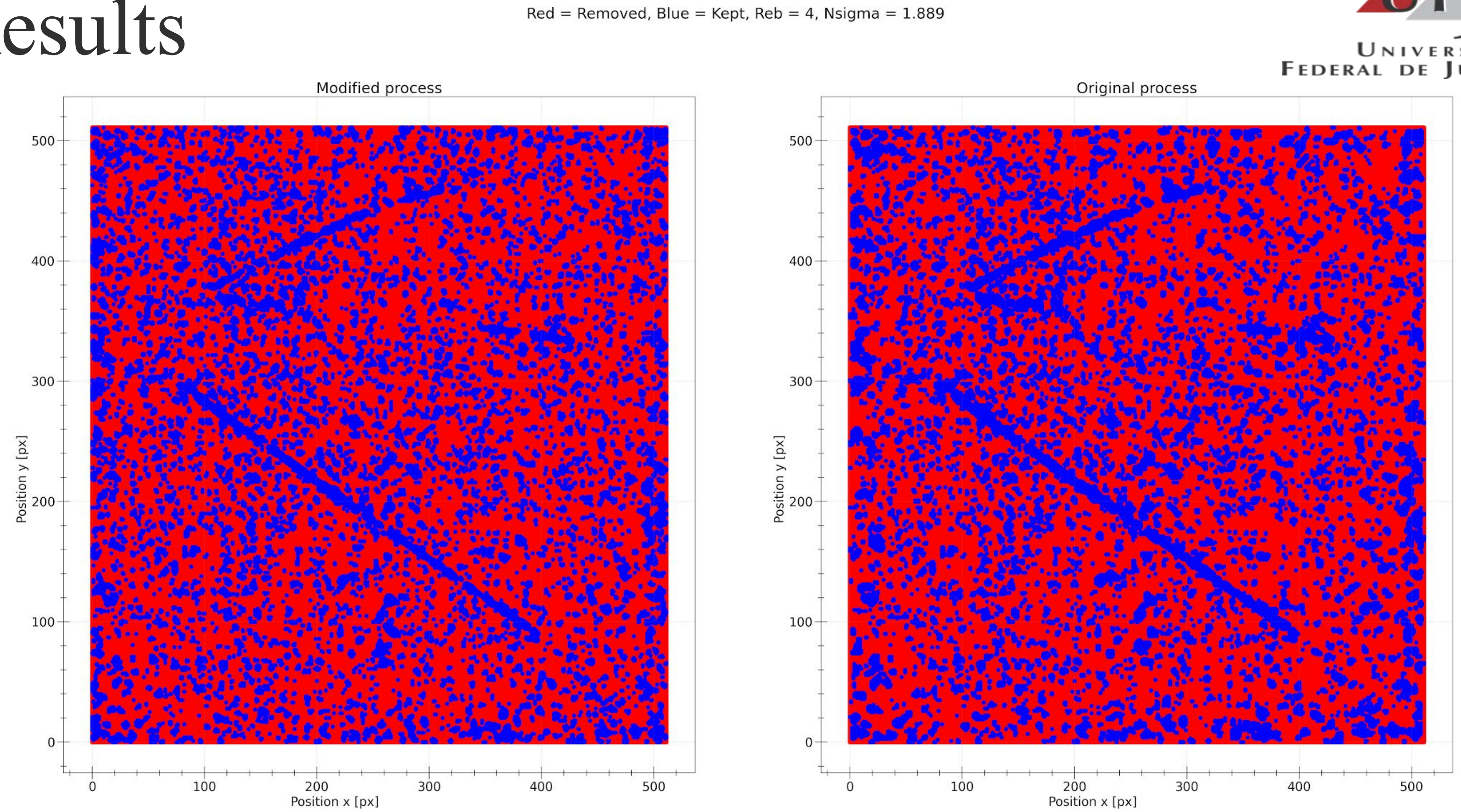
Red = Removed, Blue = Kept, Reb = 4, Nsigma = 1.000
Modified process
Original process
Position y [px]
Position x [px]
0
100
200
300
400
500

Results



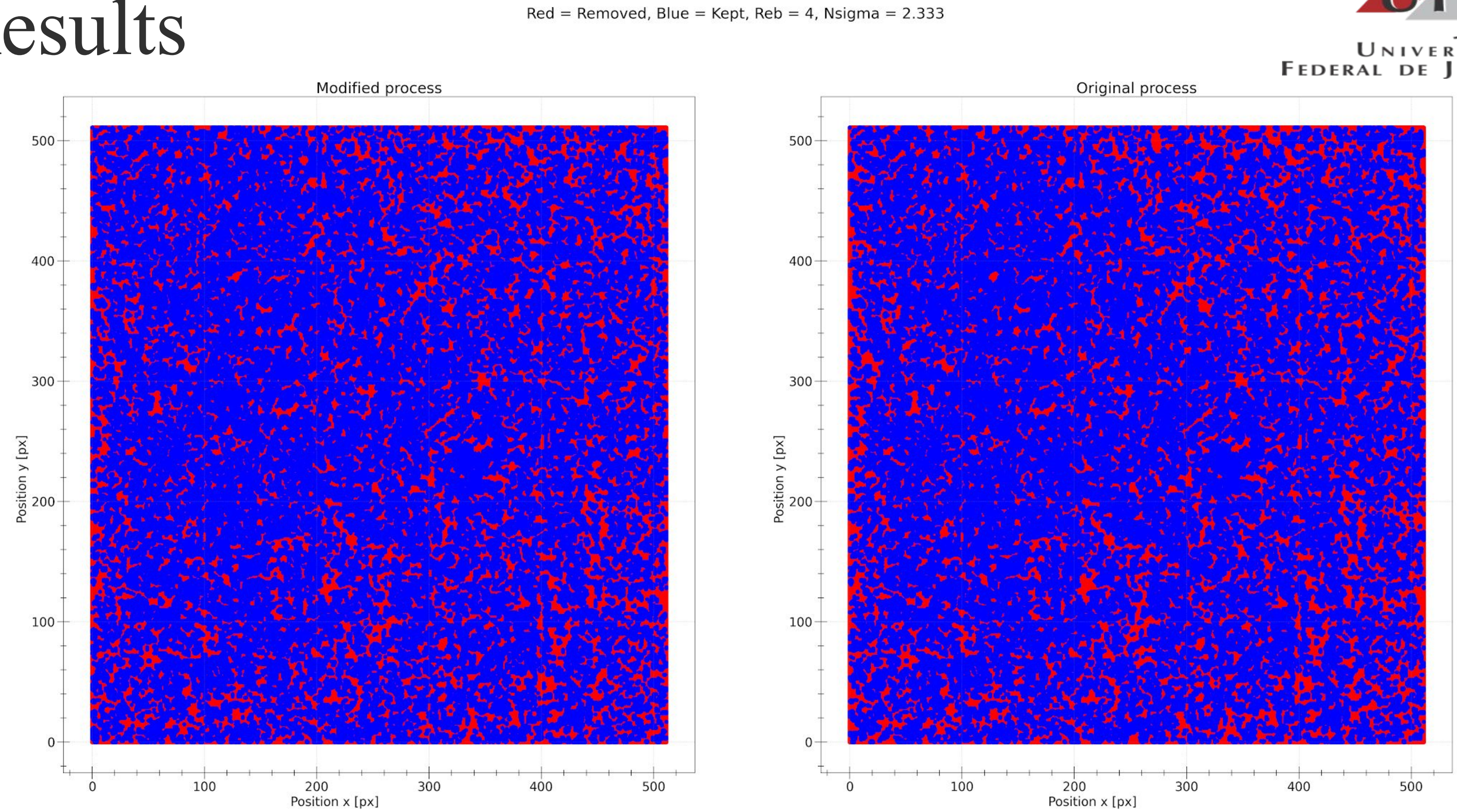
Red = Removed, Blue = Kept, Reb = 4, Nsigma = 1.444
Modified process
Original process
Position y [px]
Position x [px]
0
100
200
300
400
500

Results



Red = Removed, Blue = Kept, Reb = 4, Nsigma = 1.889
Modified process
Original process
Position y [px]
Position x [px]
0
100
200
300
400
500

Results

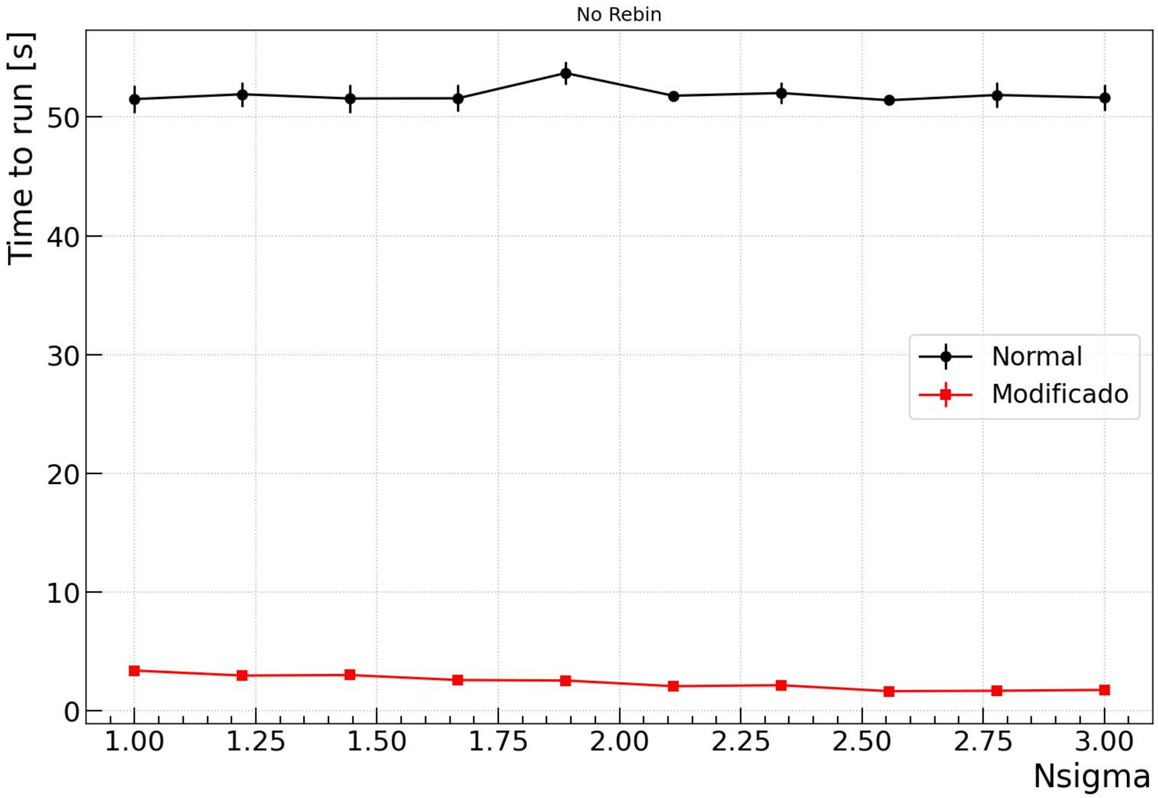


Red = Removed, Blue = Kept, Reb = 4, Nsigma = 2.333
Modified process
Original process
Position y [px]
Position x [px]
0
100
200
300
400
500

Results

- This modification was made with the full resolution images in mind, so we had to test how long does it take to complete the process for it.

Results - time to run at full resolution



Conclusion

UNIVERSIDADE FEDERAL DE JUIZ DE FORA

- Given how crucial the noise reduction algorithm is for the pre-processing, we've realized that it's one of the most time consuming steps, as it's recursive;
- If the collaboration intends to process images in full resolution, the noise reductor should to be redesigned as a faster filter;
- In order to help, a kernel-based filter was designed, with similar response and smaller processing time.
- Adiciona aqui que ainda esta em fase de testes...

Conclusion

- Any thoughts?