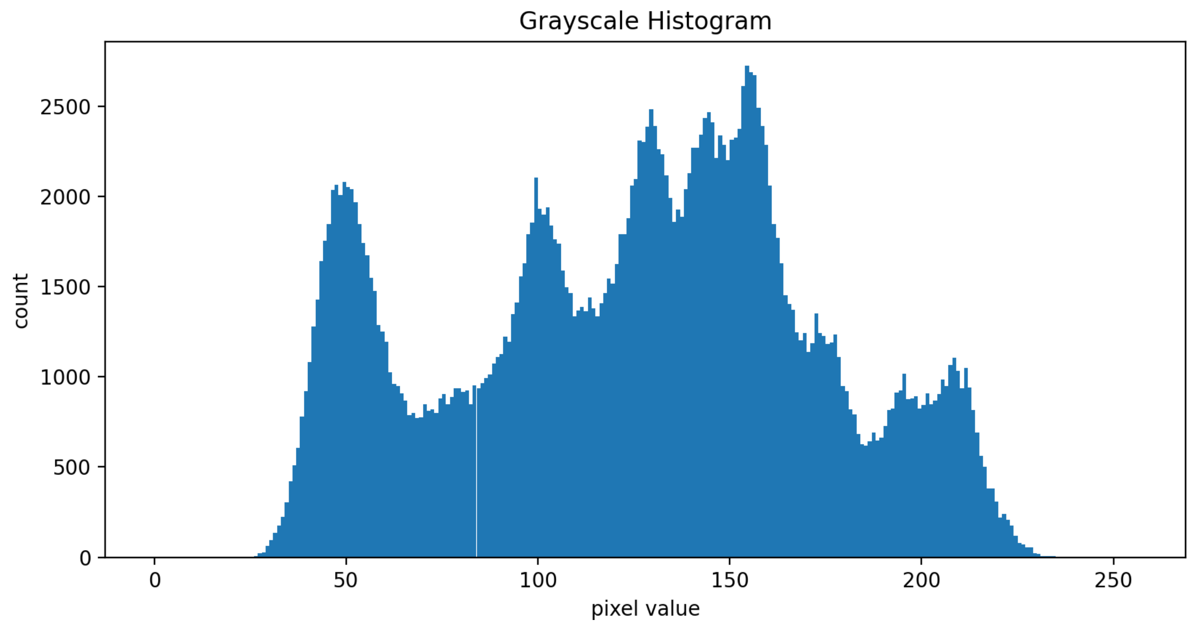
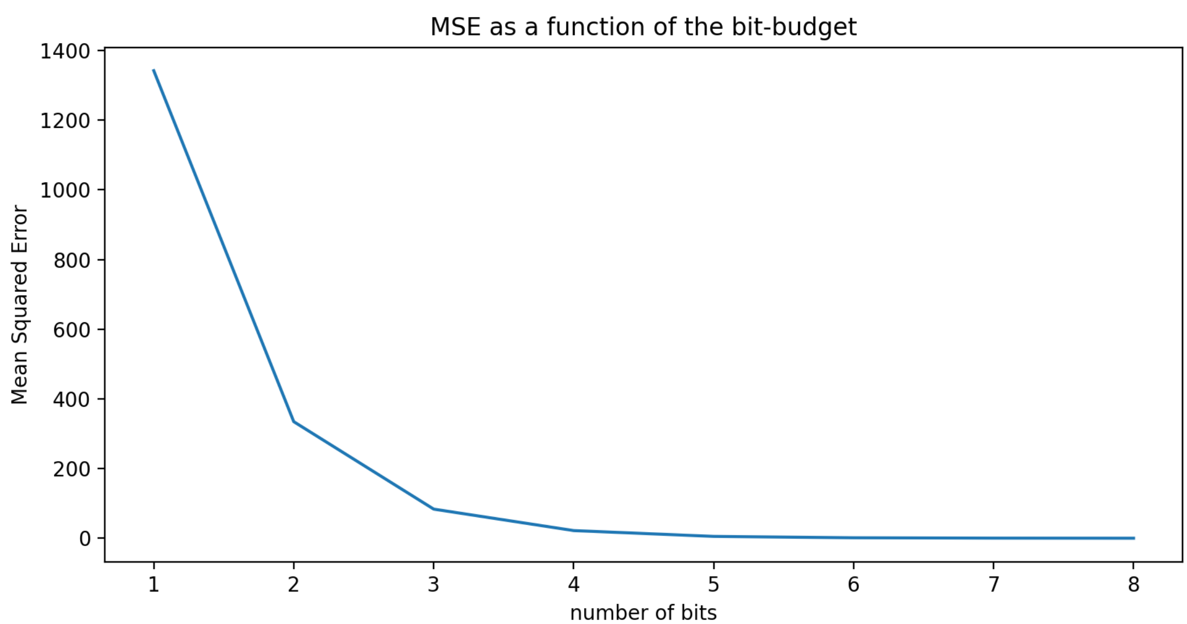
1. **Quantization**

1.1. As can be seen in the histogram, the distribution of the gray levels is not uniform.

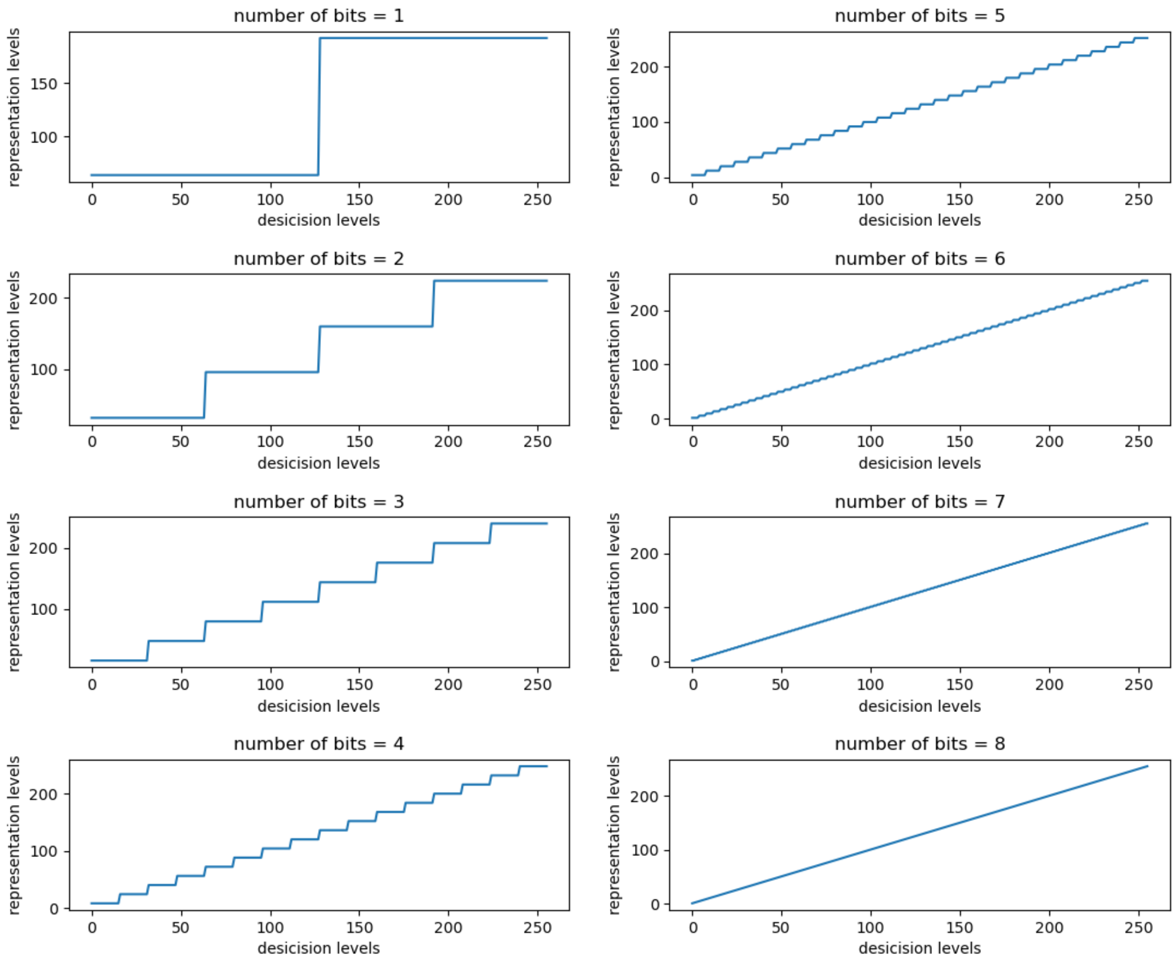


1.2.

a)



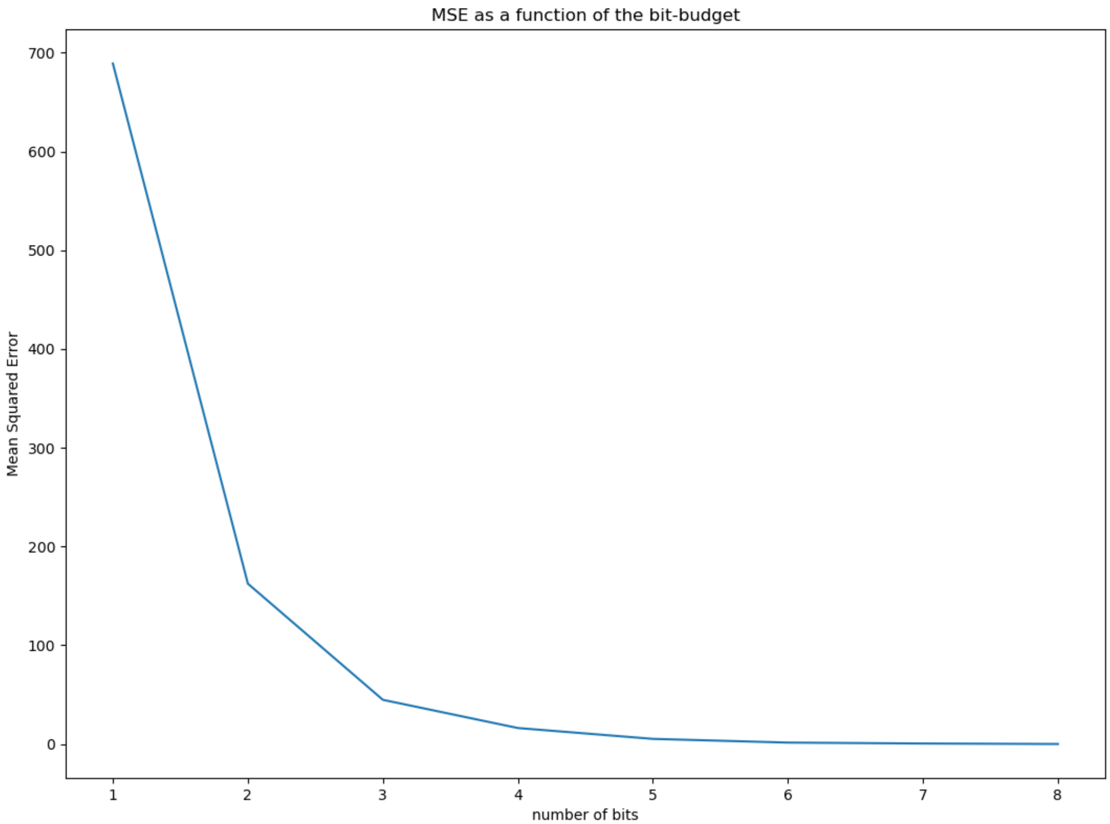
b)


number of bits = 1
number of bits = 2
number of bits = 3
number of bits = 4
number of bits = 5
number of bits = 6
number of bits = 7
number of bits = 8
representation levels
desicision levels
decision levels

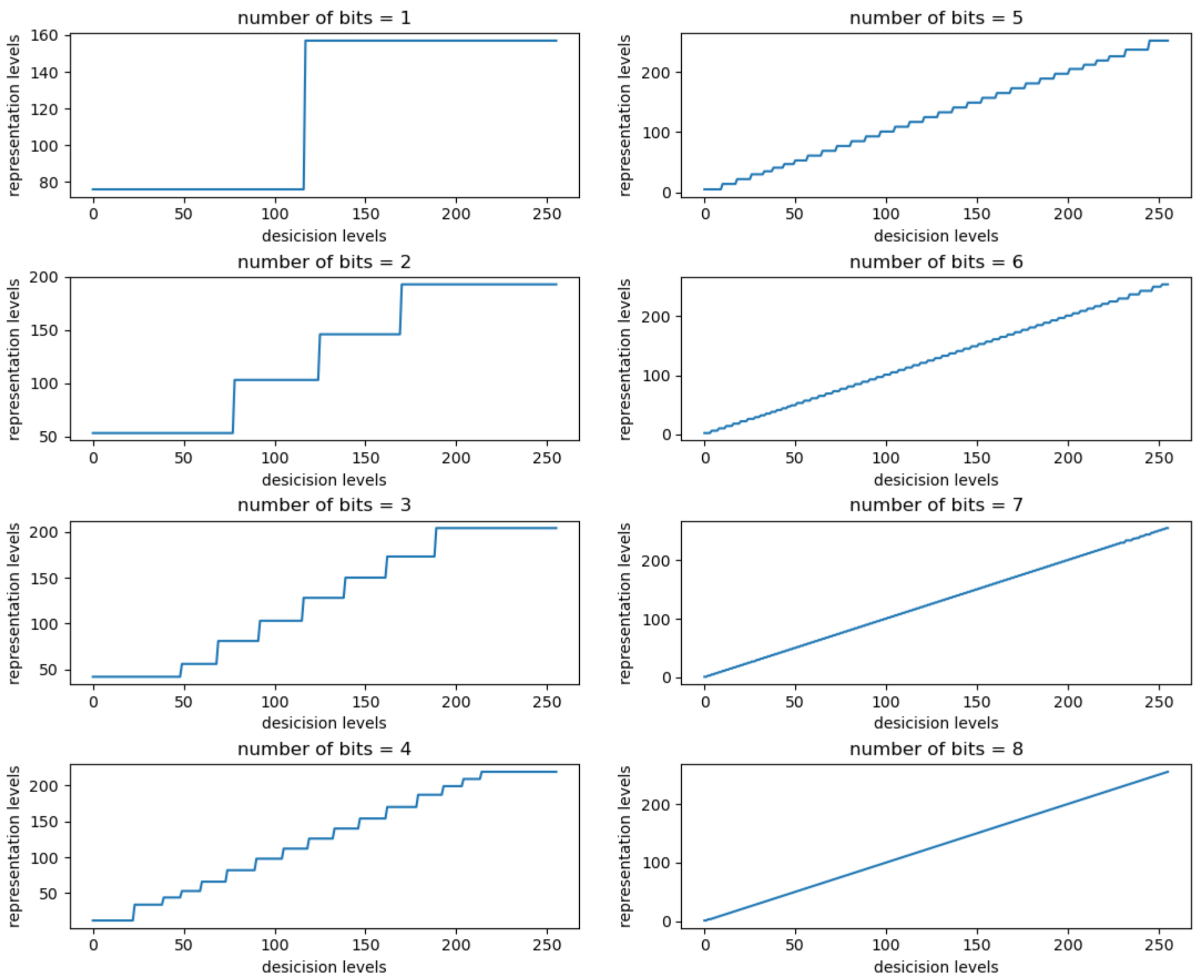
1.3. Implemented in code

1.4.

a)



b)

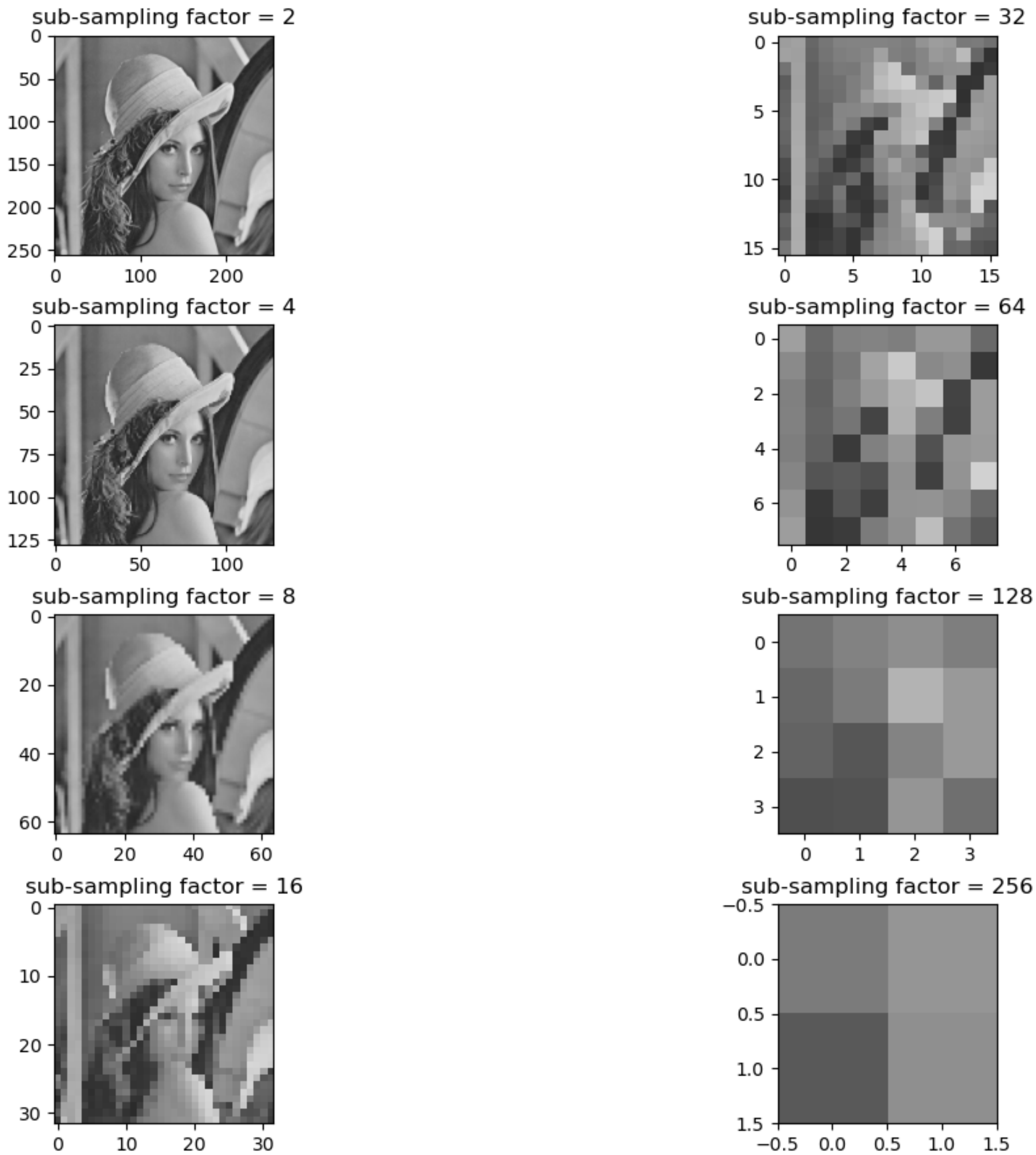


c) As Expected, The Intervals that determined by the decision levels are not uniformly distributed. They are more concentrated where the pdf is denser. Therefore, the Max-Lloyd algorithm achieves a smaller MSE especially when the number of bits for representation is small. We can also see that the Max-Lloyd algorithm MSE is always lower since its decision levels are initialized uniformly and at each step the error can not get bigger.

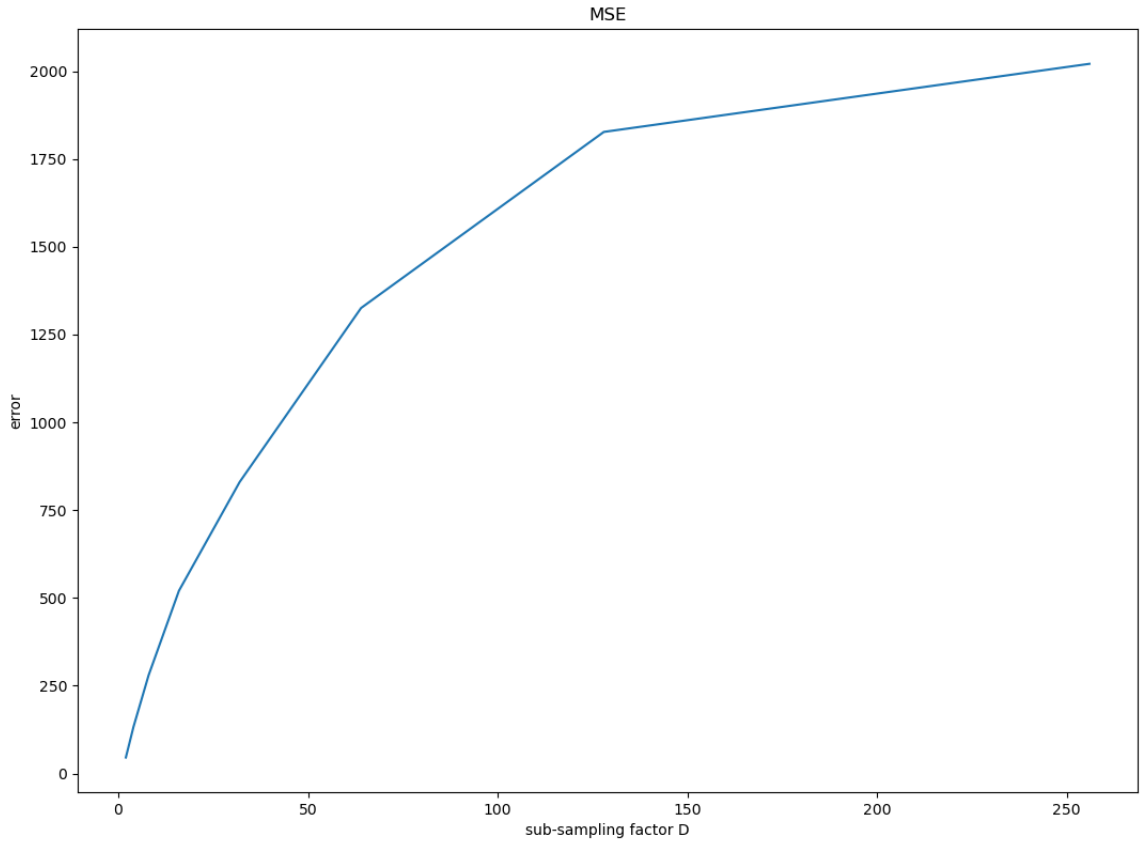
2. Subsampling and Reconstruction

2.1.

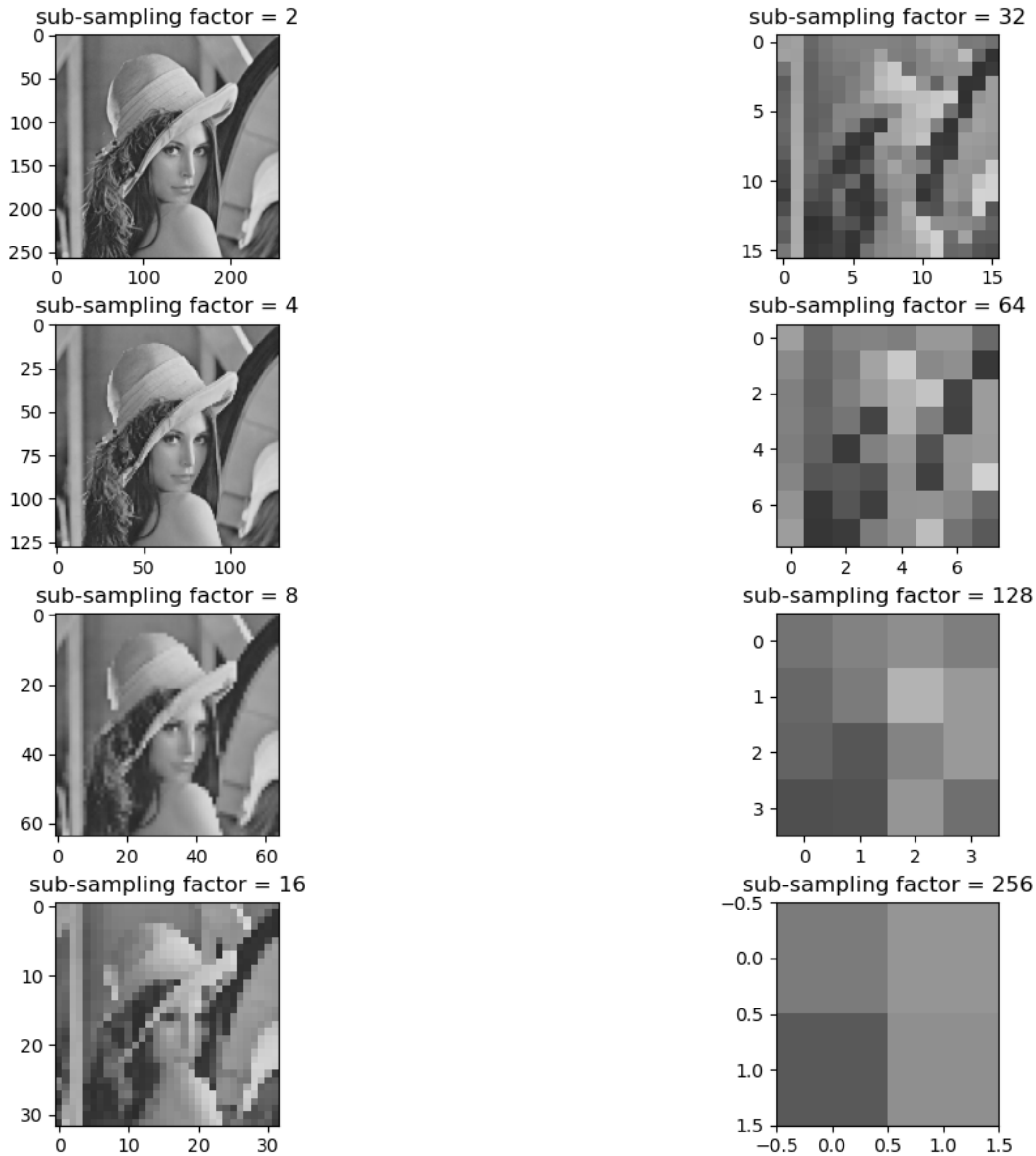
a) sub-sampled image in the MSE sense, for all different sub-sampling factor:



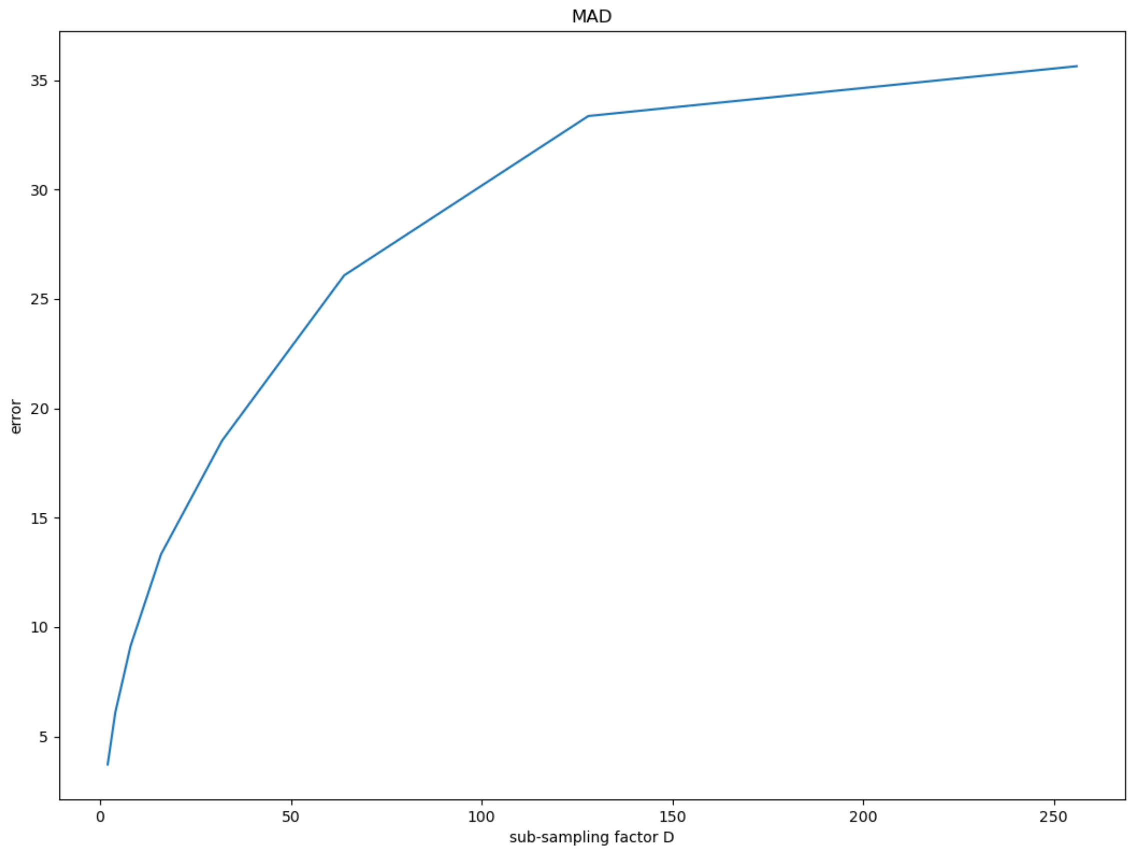
The MSE as a function of the integer sub-sampling factor:



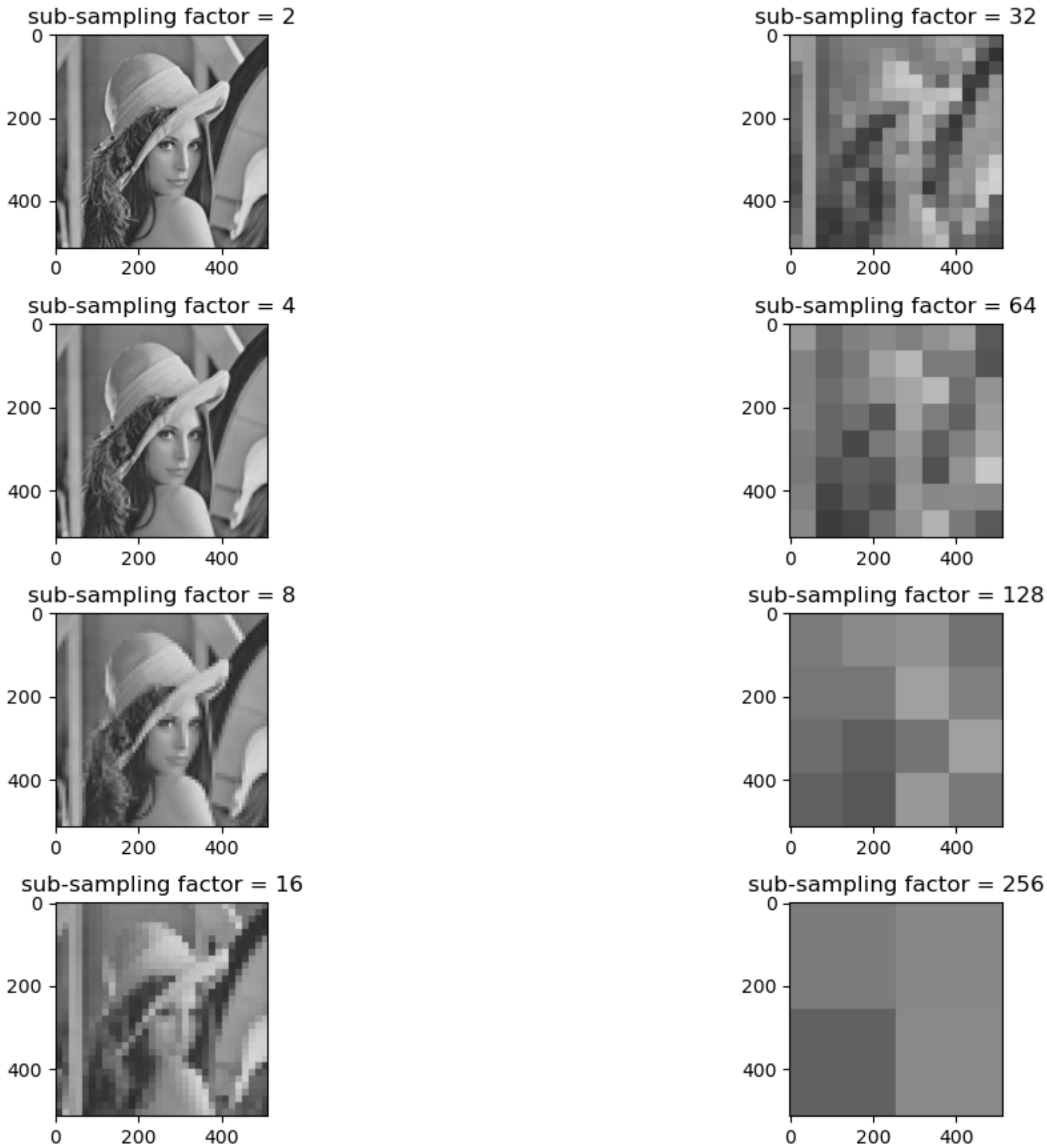
b) sub-sampled image in the MAD sense, for all different sub-sampling factor:



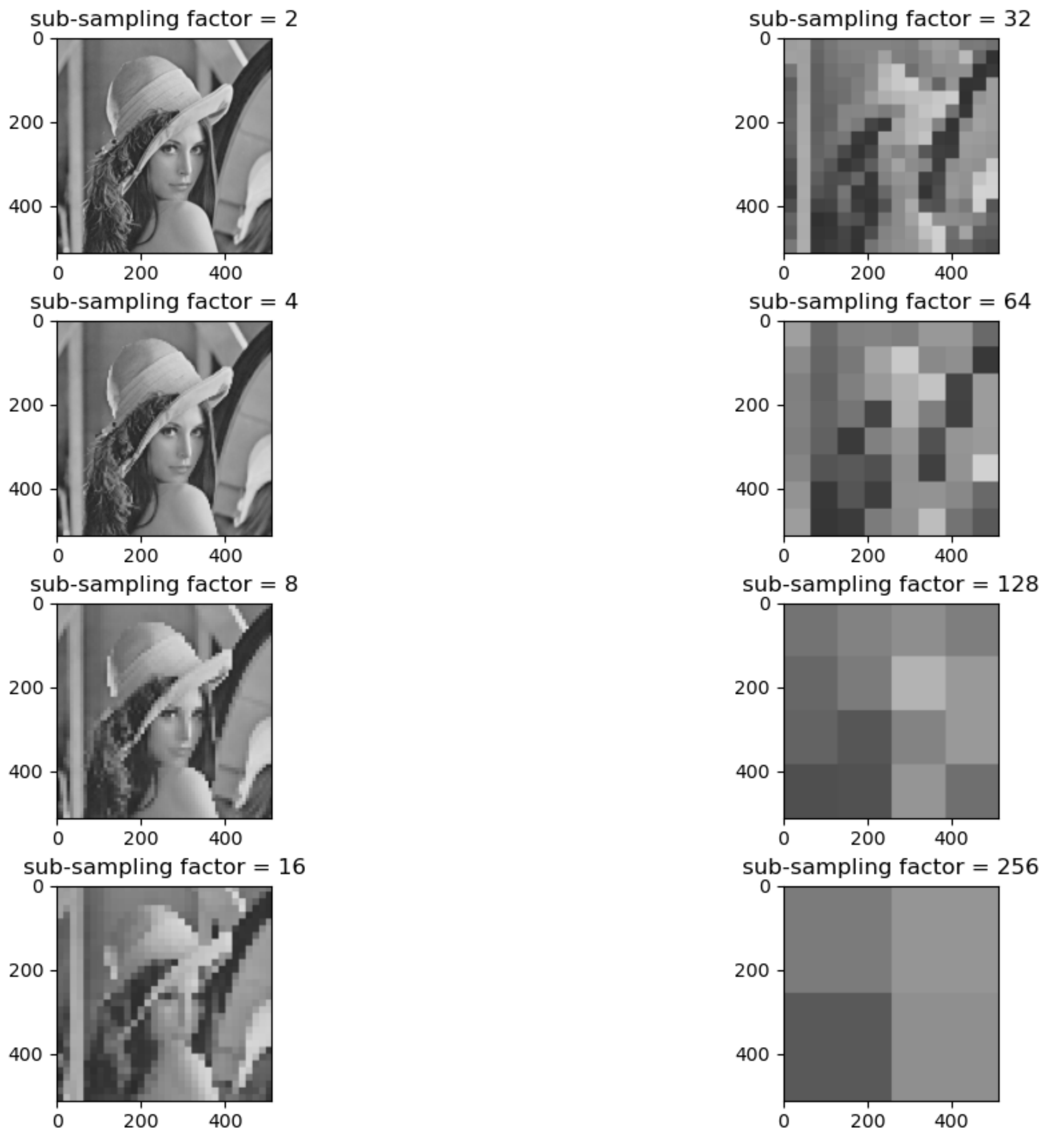
The MAD as a function of the integer sub-sampling factor:



2.2. reconstructed MSE:



Reconstructed MAD:



2.3. As expected, the bigger the sub-sampling integer D is, the more information the picture loses and thus the picture gets blurrier both in the MSE and MAD sense. For $D \geq 2^5$ the picture is not recognizable.

3. Solving the L p problem using the L 2 solution

3.1. Pseudo-code:

a) Input:

3.1.a.1. $signal\ f, N, \epsilon, p$

b) Output:

3.1.b.1. $\widehat{f},\ an\ approximation\ of\ L^p\ solution\ with\ N \times N\ samples$

3.1.b.2. $w, a\ weight\ function\ that\ when\ applied\ to\ the\ W - MSE\ problem\ approximats\ the\ L^p\ solution$

c) Initialization:

3.1.c.1. $w \leftarrow positive\ values\ (usualy\ 1s)$

d) While stopping condition is not met:

3.1.d.1. For each sample i in the domain:

3.1.d.1.1. $\hat{f}^i{}_{next} \leftarrow \frac{1}{\int_{I_|} w\{x\} dx} \int_{I_i} f(x)\, w(x) dx,\ \ \forall x \in I_i$

3.1.d.1.2. $w_{next}^i \leftarrow \min\{\left|f(x) - \hat{f}_{next}^i(x)\right|^{p-2}, \frac{1}{\epsilon}\},\ \ \forall x \in I_i$

3.1.d.2. $\hat{f} \leftarrow \hat{f}^i{}_{next}\ in\ interval\ I_i\ for\ each\ i\ in\ the$ domain

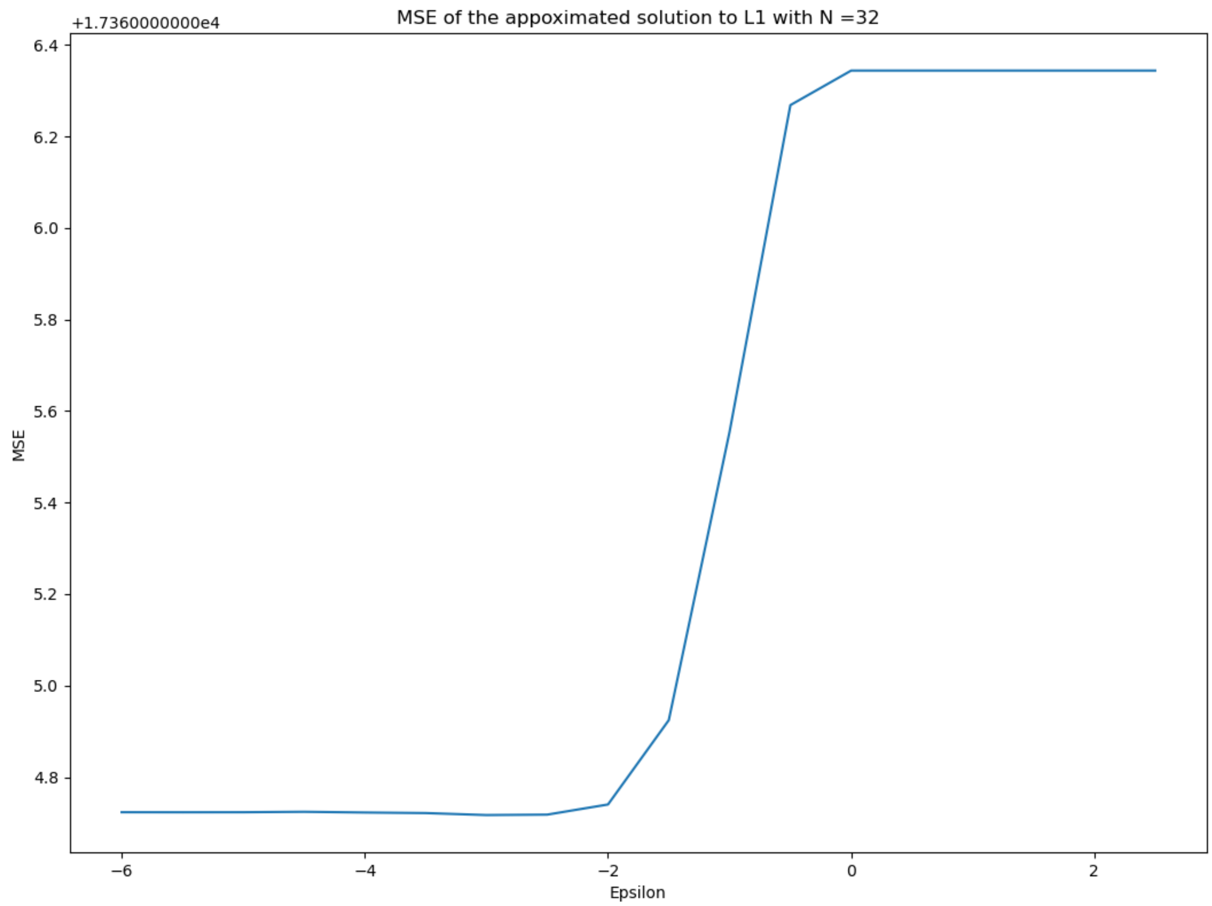
3.1.d.3. $w \leftarrow w^i{}_{next}\ in\ interval\ I_i\ for\ each\ i\ in\ the$ domain

e) Return $\hat{f}$

3.2. Implemented in code

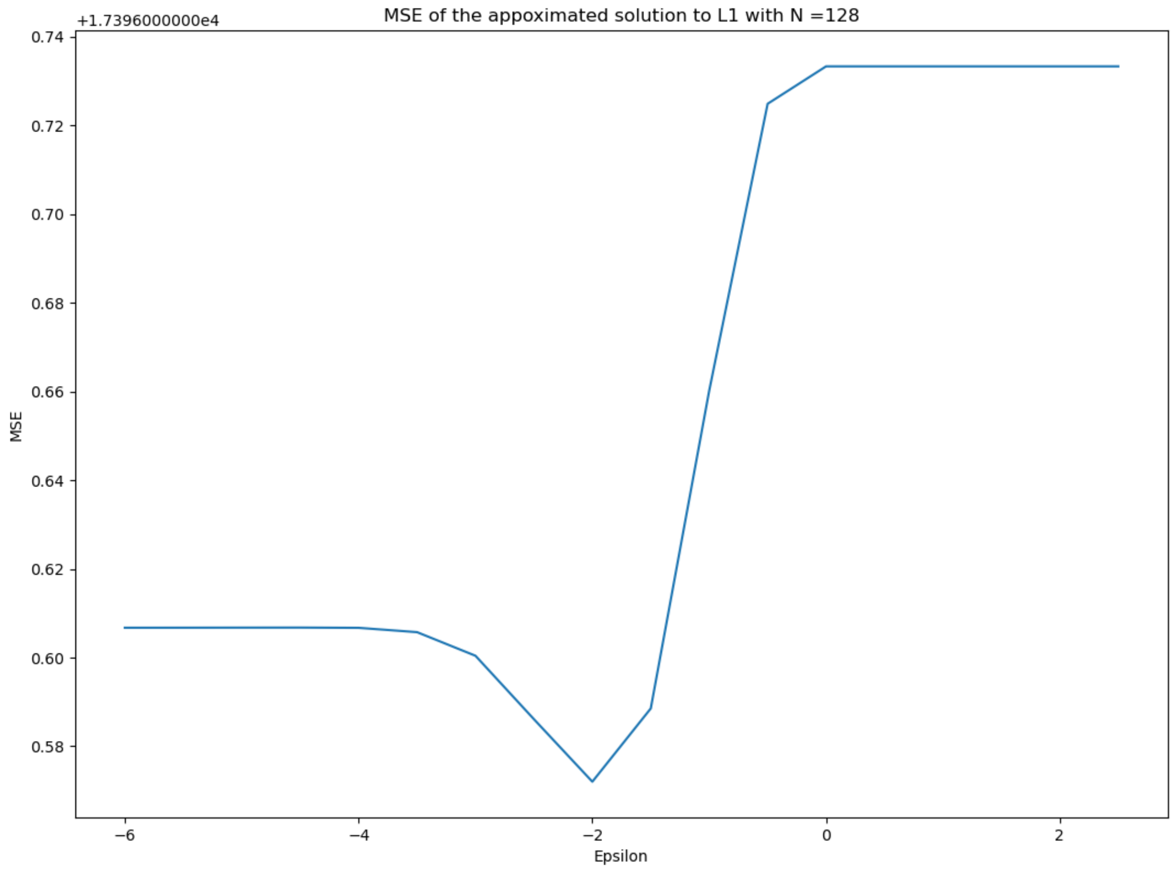
3.3. Implemented in code

3.4. Graph 1:



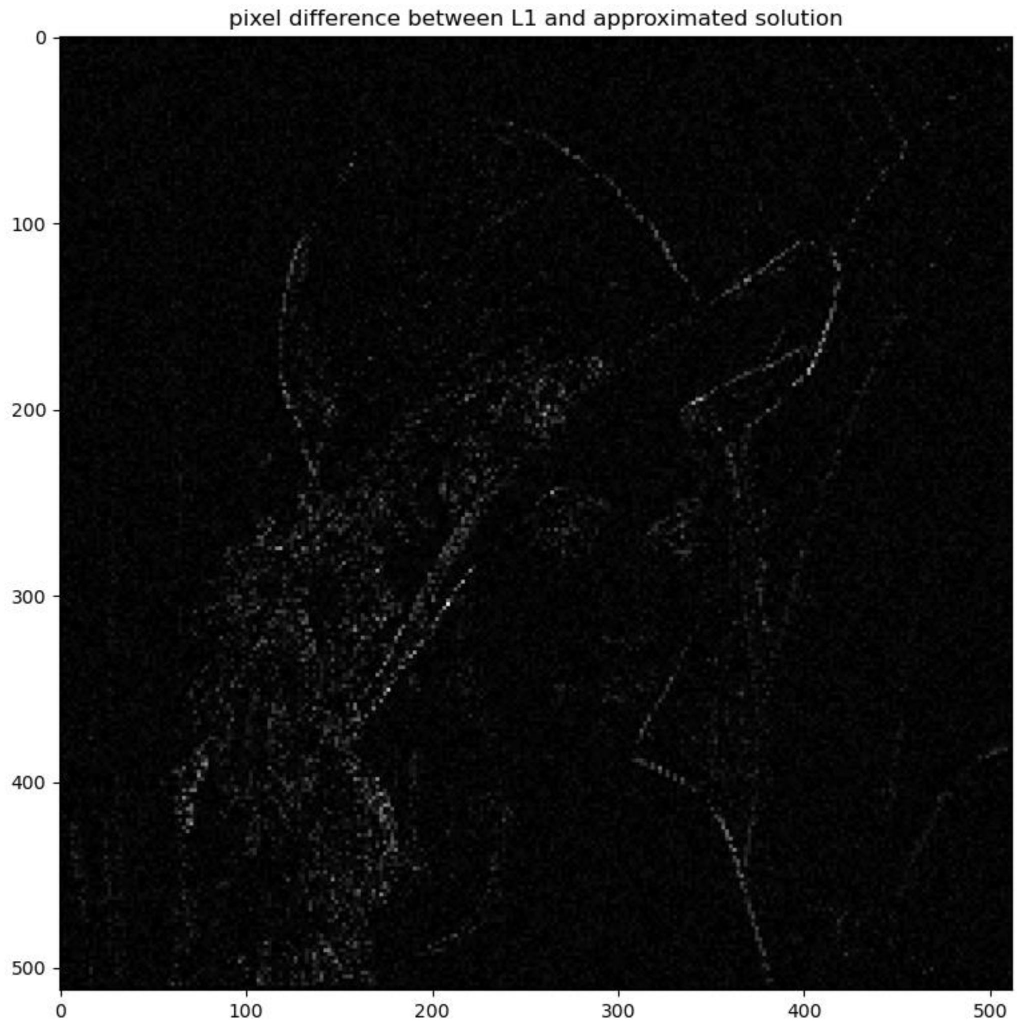
Our explanation- it seems that for N=32 there are not many small outlier values in our W and therefore there is no need to use an epsilon.
We can see in the graph that when epsilon is smaller then 0.01 it doesn't significantly effect the convergence, and as the epsilon grows more values in W are chosen to be epsilon and the error grows, until epsilon is large enough that constant W=1/epsilon and IRLS converges to the L2 solution.

Graph 2:



In the case of N =128 we can see there are some small outliers in W that effect our convergence, and thus choosing the right epsilon normalizes them and the error get's smaller.
And again when epsilon is too big our error get's larger until W is constant and we converge to the L2 solution.

Graph 3:



We examined the difference between the approximated and the L1 solution in order to see how the difference is distributed across the domain.
It is fascinating to see that the error of the approximation is larger around edges in the picture.

3.5. We ran IRLS for p = 1.5 and p = 4 and found that:

a) For p=1.5 the algorithm converges for all the different epsilons tested.
b) For p=4 the algorithm diverges for epsilons that are too small, the algorithm oscillates between two solutions because the step is too large. As we enlarged epsilon the step got smaller and the algorithm converged to a solution.
c) We read about the IRLS convergence problem (1) and it aligns with our results. It seems that for $1.5 \leq p < 3$ the algorithm should converge even without the use of epsilon. However, for $p > 3$ such as $p = 4$, the basic algorithm diverges and the various methods discussed in this paper must be used.

(1) - https://cnx.org/contents/krkDdys0@12/Iterative-Reweighted-Least-Squares