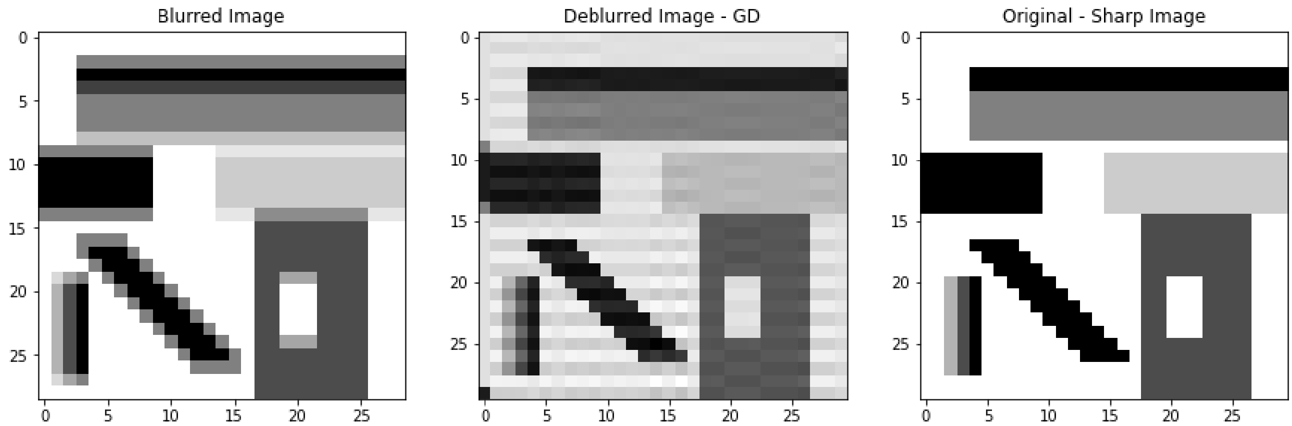
1 Gradient Descent

1.1 λ = 0

- No more blurry areas, the blurring process was reverted perfectly, but...
- Many artifacts in originally homogenous areas, as we do not use regularization.
- As with kernel k1, the derivative of the data term for y = 0 is 0, the first column of the image is pretty much the same as the first of the blurred image. This leads to there being no regularization for y = 0. Furthermore, we also have an arbitrary value in the bottom left corner.
- As there is no regularization, the loss data term is very small: 0.00151 in our case.



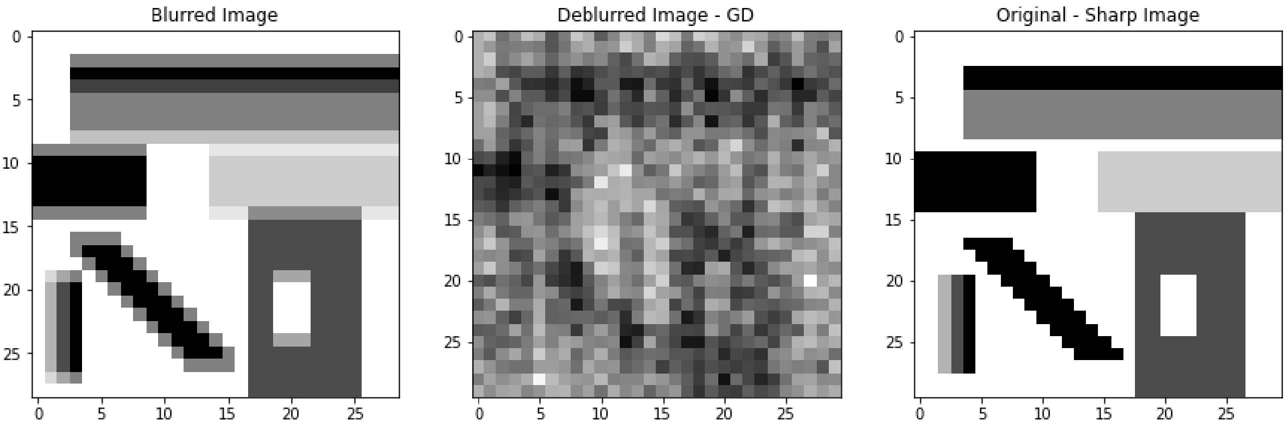
1.2 Comparing three regularization approaches

The best result is achieved using Anisotropic Total Variation (see below

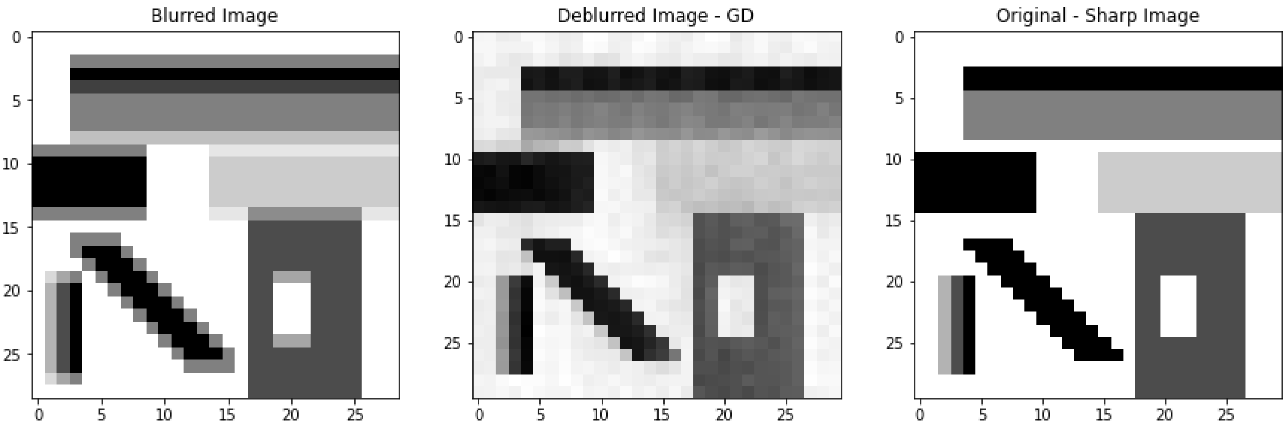
2 Different λ for Gradient Descent with Anisotropic Total Variation

In the middle you can see the deblurred image with the fitting λ, with the blurred image on the left and the original sharp image on the right

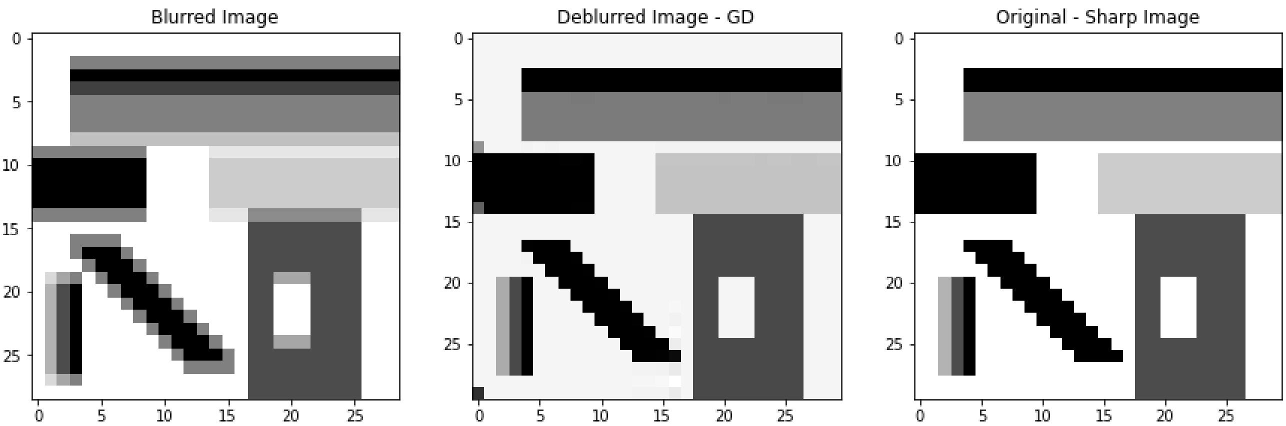
Very high (λ = 1, out of exercise defined bounds, but good for explanation)



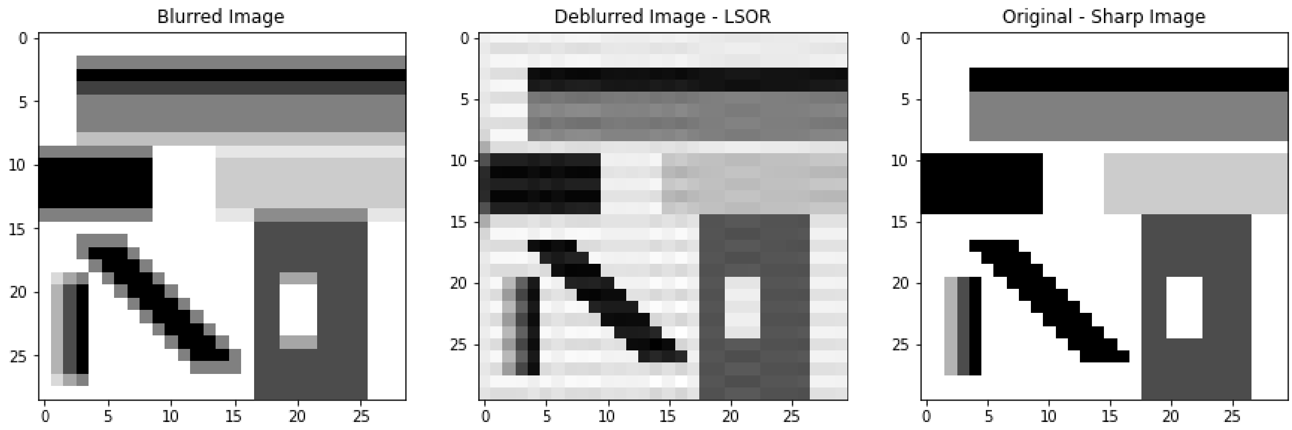
High (λ = 0.1)



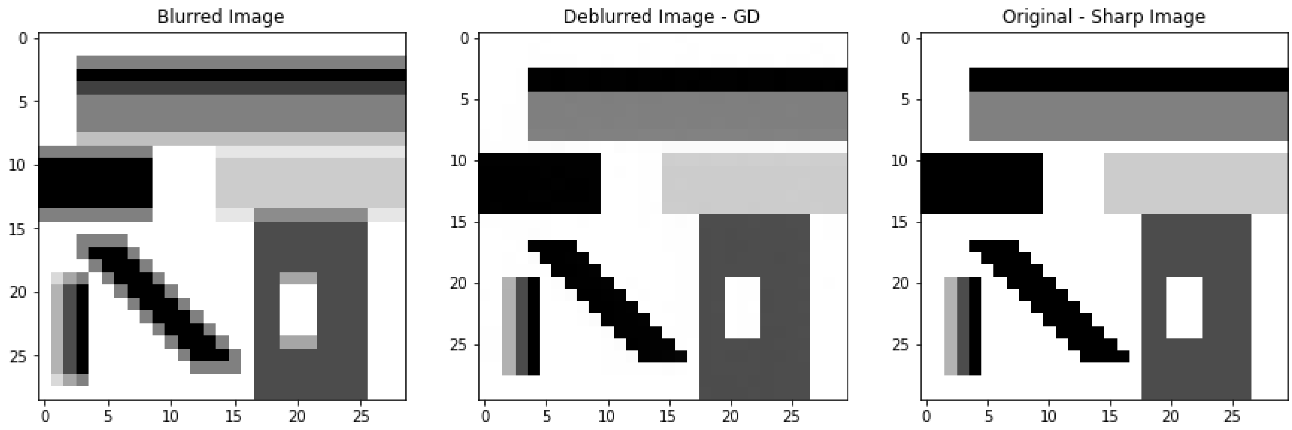
Low (λ = 0.001)



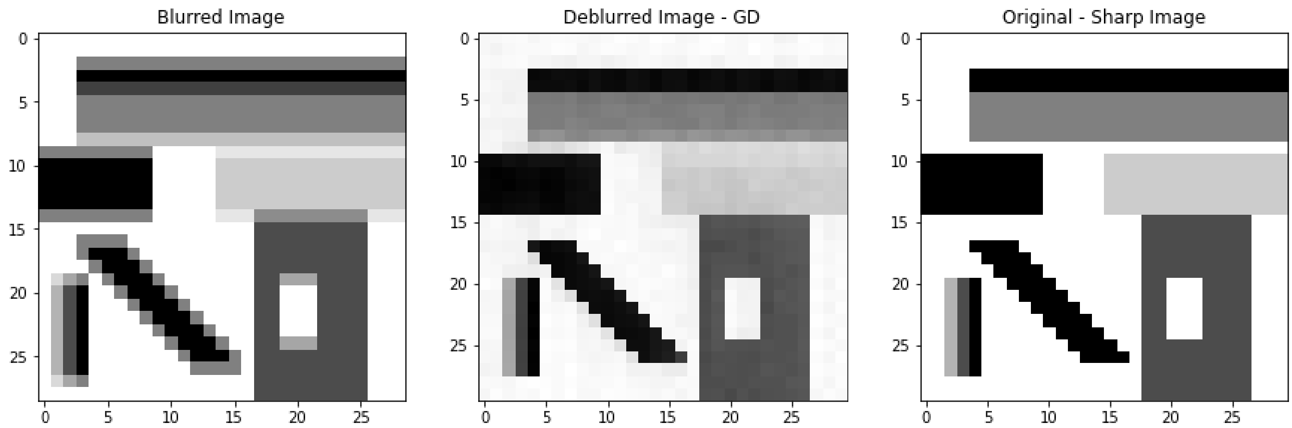
Very low (λ = 0.0001, out of exercise defined bounds, but good for explanation)



Reasonable (λ = 0.005)



Fitting (λ = 0.05)



2.1 λ's effects

λ can be understood as the weight of the regularization parameter. It can be understood as how much noise is expected to be present in the image. The higher its value, the less the algorithm regards the data term. If we choose a low value, the algorithm tries to ensure that it finds an image, which when convolved with the kernel, gives an image that is as close as it can to the blurred image. Looking at Anisotropic Total Variation, we receive a checker pattern, as it tries to get as close to 0 as possible. If the λ is properly chosen, we actually get a very good result, as seen for λ = 0.005.

3. Sum of Square Distances for Gradient descent with Anisotropic Total Variation → See Jupyter Notebook