Tasks

1. Constrained quadratic optimization to include additional hard constraints
As in 2. in Exercise 1, define a spatial grids x by $x_i = (i-1)/(N-1), i = 1, \dots, N$ using `linspace` and $N = 20$. Construct the matrix A by $A_{i,j} = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(x_i - x_j)^2}{2\sigma^2}\right)$ with $\sigma = 0.1$ and normalize all rows with `A(i,:) = A(i,:)/norm(A(i,:),1)`. The vector f we want to recover is given as

$$f_i = \begin{cases} 0 & \text{if} \quad x_i < 0.3 \\ \sqrt{x_i - 0.3} & \text{if} \quad x_i \geq 0.3 \end{cases}$$

The data is given as $g = Af + n$ with n begin Gaussian with a standard deviation of 0.1. While we can compute the zero-order Tikhonov regularisation

$$f_\alpha = \underset{f}{\operatorname{argmin}} \|Af - g\|_2^2 + \alpha \|f\|_2^2 \tag{1}$$

as `fAlpha = (A'*A + alpha * eye(n))\(A'*g);`, it is difficult to add a-priori knowledge on f that we have as additional hard constraints.
1.1 Reformulate 1 as a quadratic optimization problem in its standard form

$$(1) \Longleftrightarrow \underset{f}{\operatorname{argmin}} \tfrac{1}{2} f^T H f + q^T f, \tag{2}$$

use `quadprog.m` to solve it and compare to `fAlpha`.
1.2 Read the documentation of `quadprog.m` and add positivity constraints $f \geq 0$ to (2). *1.3* Assume that we know f_1 and f_n (*end point constrains*) and incorporate them as equality constraints into (2).
1.4 Assume that we know that f is monotonically increasing $f_1 \leq f_2 \leq \ldots \leq f_n$ and add this knowledge as inequality constraints to (2).
1.5 Compute the solutions for all possible combinations of the three types of constraints above and compare them.

2. Wavelet deconvolution with iterative soft thresholding (IST)
Load the house image, compute a convolution with a Gaussian kernel and add noise to the result as in task 1. of CW1 and demonstrated in the lectures.
We will first use IST to solve the wavelet denoising problem given by

$$c_\lambda = \underset{c}{\operatorname{argmin}} \left\{ \tfrac{1}{2} \|g - AW^{-1}c\|_2^2 + \lambda |c|_1 \right\}$$

Here, W^{-1} denotes the inverse of a wavelet transform, i.e., a linear operator that reconstructs an image f from its wavelet coefficients c. The denoised image f_λ can be recovered from c_λ as $f_\lambda = W^{-1} c_\lambda$. Implementing IST consists of two steps:
2.1 For a given c, implement a function that computes the gradient $p(c) = (W^{-1})^* A^* (AW^{-1}c - g)$, where $(W^{-1})^* = W$ as the wavelet transform is unitary. For a given image f and a coefficient vector c, you can compute $y = W(f)$ by the function `haardec2.m` and $x = W^{-1}(c)$ by the function `haarrec2.m` which can be found on the website

`http://www.cs.ucl.ac.uk/staff/S.Arridge/teaching/optimisation/CW2`

2.2 Implement the "Soft-Thresholding Operator"

$$S_\alpha(x) := \underset{y}{\operatorname{argmin}} \left\{ \frac{1}{2} \|y - x\|_2^2 + \alpha |y|_1 \right\} = \max(|x| - \alpha, 0) \operatorname{sgn}(x)$$

and plot it as a function of x for different values of α. *2.3* With these preparations, the IST iteration is given as

$$c_\lambda^k = S_{\lambda\tau}\left(c^{k-1} - \tau p(c^{k-1})\right)$$

Write a function that implements IST for our scenario and returns c_λ^K, the result of K iterations with a step size of τ. Plot the value of $J(c^k) := \frac{1}{2}\|g - AW^{-1}c^k\|_2^2 + \lambda|c^k|_1$ against k for different choices of τ and choose a reasonable value for τ. Consult the documentation of `imagesc.m` to learn how to visualize different images with the same color scaling. Use this to compare f_α for different α.

3. TV deconvolution with ADMM

Several discrete versions of the total variation (TV) regularization functional exist. Here, we consider the isotropic version based on finite forward differences and Neumann boundary conditions. If we use (i, j) to index the pixel in the i^{th} row and j^{th} column of the discrete, $N_x \times N_y$ sized image f, we can define $TV(f)$ as:

$$TV(f) = \sum_{(i,j)} \sqrt{\left(f_{(i+1,j)} - f_{(i,j)}\right)^2 + \left(f_{(i,j+1)} - f_{(i,j)}\right)^2}, \quad f_{(N_x+1,j)} := f_{(N_x,j)},\ f_{(i,N_y+1)} := f_{(i,N_y)}$$

3.1 Write a function that computes $TV(f)$ by using implementations of the finite difference operators $D_x(f)$ and $D_y(f)$ applied to an image f as

```
Dxf   = [diff(f,1,2),zeros(size(f,1),1)];
Dyf   = [diff(f,1,1);zeros(1,size(f,2))];
```

Verify that the corresponding adjoint operations $D_x^T(f)$ and $D_y^T(f)$ are given as:

```
DxTf = [-f(:,1),-diff(f(:,1:end-1),1,2),f(:,end-1)];
DyTf = [-f(1,:);-diff(f(1:end-1,:),1,1);f(end-1,:)];
```

3.2 Write a function that computes the "Vectorial Soft-Thresholding Operator"

$$S_\alpha(u) := \begin{cases} \max(s-\alpha, 0)\ u/s & \text{if} \quad s > 0, \quad \text{where} \quad s = \sqrt{u_x^2 + u_y^2} \\ 0 \quad \text{if} \quad s = 0 \end{cases}$$

for a vector field given as two images `ux`, `uy` storing the x and y components at every pixel.
3.3 Load, convolute and noise the house image as in 3 and write a function that implements the ADMM iteration for our scenario to solve

$$f_\lambda = \underset{f}{\operatorname{argmin}}\left\{\tfrac{1}{2}\|g - Af\|_2^2 + \lambda TV(f)\right\}$$

The quadratic optimization problem for the update of f, can be solved as in previous work, only the right hand side has to be modified and changed in every iteration. Consult the documentation of the linear solver you use to restrict the number of iterations to about 20 and start with setting $\rho = 1$. Compare f_α for different values of α, compute their residuum and ℓ_2 error to the real solution.

Report

You should write a short report, probably no more than 6-8 pages, in which you describe the steps you took and the design choices you made. Indicate any problems encountered and how you solved them. Compare and contrast the different solutions you obtained, and draw conclusions regarding what you consider to be an optimal approach. Code, if you consider it relevant, can be attached as an appendix (i.e. not included in the 6-8 page report).