Mathematics of data: homework 3

Marco Pietro Abrate 292996

December 13, 2018

1 Learning-base compressed subsempling

Here we study the problem of recovering a vector $\mathbf{x}^\natural \in \mathbb{C}^p$ from a set of linear measurements of the form

$$\mathbf{b} = \mathbf{A}\mathbf{x}^\natural \tag{1}$$

where $\mathbf{A} \in \mathbb{C}^{n \times p}$ is a known measurement matrix. The problem is ill-posed unless $n \geq p$, but even if $p > n$ the reconstruction is possible if the matrix $\mathbf{x}^\natural$ is sparse or compressible. We define $\Psi \in \mathbb{C}^{p \times p}$ be a unitary matrix. We then design $\mathbf{A}$ in (1) by taking subsamples

$$\mathbf{b} = \mathbf{P}_\Omega \Psi \mathbf{x}^\natural \tag{2}$$

for some subsampling matrix $\mathbf{P}_\Omega \in \{0,1\}^{n \times p}$ whose rows are canonical basis vectors containing 1 in one entry and 0 in all other entries. The set $\Omega \subseteq \{1, ..., p\}$ indexes the entries that are subsampled. We do not know $\mathbf{x}^\natural$, but we have access to some number m of fully sampled training signals $\mathbf{x}_1, ..., \mathbf{x}_m$. We assume that $||\mathbf{x}_j||_2^2 = 1$ for all j. We then select the indices that capture the most energy on average

$$\hat{\Omega} = \underset{\Omega:|\Omega|=n}{\arg\max} \frac{1}{m} \sum_{j=1}^{m} ||\mathbf{P}_\Omega \Psi \mathbf{x}_j||_2^2 \tag{3}$$

The least-squares estimator is given by

$$\hat{\mathbf{x}} = \mathbf{A}^\dagger \mathbf{b} = \mathbf{A}^*(\mathbf{A}\mathbf{A}^*)^{-1}\mathbf{b} \tag{4}$$

And the squared-error is given by

$$\varepsilon(\mathbf{x}, \hat{\mathbf{x}}) = ||\mathbf{x} - \hat{\mathbf{x}}||_2^2 \tag{5}$$

1.1 Exercise 1.2

The optimization problem in (3) can be solved by sorting. Expressing the function in terms of $\langle \Psi_i, \mathbf{x}_j \rangle$, where Ψ_i^T is the *i-th* row of Ψ, we obtain

$$\Psi \mathbf{x}_j = \begin{pmatrix} \langle \Psi_1, \mathbf{x}_j \rangle \\ ... \\ \langle \Psi_p, \mathbf{x}_j \rangle \end{pmatrix} \tag{6}$$

At this point we compute the terms $\langle \Psi_i, \mathbf{x}_j \rangle$ for $i = 1, ..., p$ and $j = 1, ..., m$. We end up having $p \cdot m$ terms. We sum the terms with the same i and we get p terms of the form $\sum_{j=1}^{m} \langle \Psi_i, \mathbf{x}_j \rangle$. Sorting these last terms, taking the $n(< p)$ biggest and looking at the indices of Ψ, we have a solution for problem (3).

1.2 Exercise 1.3

Considering now the non-linear estimator

$$\hat{\mathbf{x}} = \Delta(\mathbf{P}_\Omega \Psi \mathbf{x}^\natural) \tag{7}$$

where Δ is any non-linear decoder, we define the non-linear optimization problem as

$$\hat{\Omega} = \arg\max_{\Omega : |\Omega| = n} \eta(\Omega) \tag{8}$$

$$\eta(\Omega) := -\frac{1}{m} \sum_{j=1}^{m} \varepsilon(\mathbf{x}_j, \hat{\mathbf{x}}_j) = -\frac{1}{m} \sum_{j=1}^{m} ||\mathbf{x}_j - \Delta(\mathbf{P}_\Omega \Psi \mathbf{x}_j)||_2^2 \tag{9}$$

The problem becomes combinatorial. We consider a set $\mathscr{S}$ of indices of $\{1, ..., p\}$ that we choose to sample, thus the mask takes the form

$$\Omega = \bigcup_{j=1}^{n} S_j, \quad S_j \in \mathscr{S} \tag{10}$$

We define a greedy algorithm to solve the problem.

Greedy mask optimization

Input: Training data $x_1, ..., x_m$, reconstruction rule Δ, sampling subset $\mathscr{S}$, maximum cost n
Output: Sampling pattern Ω

```
1. while $|\Omega| \leq n$ do
2.     for $S \in \mathscr{S}$ not yet in $\Omega$ do
3.         $\Omega' = \Omega \cup S$
4.         for each $j$, set $\mathbf{b}_j = \mathbf{P}_{\Omega'} \Psi \mathbf{x}_j$, $\hat{\mathbf{x}}_j = \Delta(\mathbf{P}_{\Omega'} \Psi \mathbf{x}_j)$
5.         $\eta(\Omega') = -\frac{1}{m} \sum_{j=1}^{m} \varepsilon(\mathbf{x}_j, \hat{\mathbf{x}}_j)$
6.     $\Omega = \Omega \cup S^*$, where $S^* = \arg\max_{S : |\Omega \cup S| \leq n} \{\eta(\Omega \cup S) - \eta(\Omega)\}$
7. return $\Omega$
```

We now study the complexity of the former algorithm. The reconstruction Δ will be assumed to have complexity C_Δ. As it is the most expensive part, we will consider the other operations to be negligible before it. The variables are the number of elements that we will sample n, the number of training elements m and the space dimension p.

From now on, we will number the loops in the algorithm with the number associated to them in the former definition, e.g. the *while* corresponds to 1, the first *for* to 2 and the second *for* to 4. Considering that 1 loops for n iterations, 2 for $p-k$ (where k is the number of elements in Ω, that varies each iteration of 1) and 4 for m iterations, we have a final complexity of

$$complexity = \sum_{k=0}^{n-1}(p-k)mC_{\Delta} \tag{11}$$

Now, adding l unique elements instead of one means that $|S| = l$. The for loop in 2 will sample from the l-combinations from the set $\mathscr{S} \setminus \Omega$. The number of iterations in 2 is now $C_{l,p-lk}$. So the complexity becomes

$$complexity = \sum_{k=0}^{n-1} C_{l,p-lk} mC_{\Delta} = \sum_{k=0}^{n-1} \binom{l}{p-lk} mC_{\Delta} \tag{12}$$

Moreover, the complexity of the greedy algorithm can be decreased using parallelization. For each element in $\mathscr{S} \setminus \Omega$, in 2, can be run a new process. The complexity would now be $n \cdot m \cdot C_{\Omega}$. Iteration 4 can also be parallelized, running a new process for each j, in order to compute $\mathbf{b}_j$ and $\hat{\mathbf{x}}_j$. Reaching a complexity of $n \cdot C_{\Omega}$. Finally, using a batch $\mathscr{S}_t$ instead of the whole set $\mathscr{S}$ can be effective because we would reduce the number of iterations in 2 from $p-k$ to $t-k$. Talking about parallelization, having a batch would also reduce the number of processes running for iteration 2 and consequently for iteration 4.

2 Proximal operators and image denoising

In this part, by an image we mean a grayscale digital image expressed as a matrix, each entry of which represents the intensity of a pixel. A widely used multi-scale localized representation in signal and image processing is the wavelet transform. The wavelet functions form a basis $\mathbf{W}$ which is orthonormal. In this section we want to find an approximation of some noisy images, to do so we can solve the following optimization problems

$$\min_{\alpha \in \mathbb{R}^p} \frac{1}{2}||\mathbf{y} - \mathbf{W}^T\alpha||_F^2 + \lambda_1||\alpha||_1 \tag{13}$$

$$\min_{\mathbf{x} \in \mathbb{R}^p} \frac{1}{2}||\mathbf{y} - \mathbf{x}||_F^2 + \lambda_{TV}||\mathbf{x}||_{TV} \tag{14}$$

where $\mathbf{y} \in \mathbb{R}^p$ is the vectorized image, of length $p = m \times m$, to be denoised, $\mathbf{W}^T$ is the 2D inverse Wavelet transform and λ_1 and $\lambda_{TV} > 0$ are regularization parameters.

2.1 Exercise 2.1

Given a convex function $g : \mathbb{R}^p \to \mathbb{R} \cup \{+\infty\}$, we recall the proximal operator of g as the solution of the following convex problem

$$prox_g(\mathbf{x}) := \arg\min_{\mathbf{y} \in \mathbb{R}^p}\{g(\mathbf{y}) + \frac{1}{2}||\mathbf{y} - \mathbf{x}||_2^2\} \tag{15}$$

Computing the proximal operator of the function $g(\mathbf{x}) := \lambda_1||\mathbf{x}||_1$ and changing name to the independent variables, we get

$$prox_g(\mathbf{y}) = \underset{\alpha\in\mathbb{R}^p}{\arg\min}\{g(\alpha) + \frac{1}{2}||\mathbf{y} - \alpha||_2^2\} = \underset{\alpha\in\mathbb{R}^p}{\arg\min}\{\lambda_1||\alpha||_1 + \frac{1}{2}||\mathbf{y} - \alpha||_2^2\} \tag{16}$$

Computing it in $\mathbf{Wy}$ is the same as solving problem (13) since $\mathbf{W}$ is orthonormal, which implies $\mathbf{WW}^T = \mathbf{I}$ and $||\mathbf{WA}|| = ||\mathbf{A}||$ for any matrix $\mathbf{A}$

$$prox_g(\mathbf{Wy}) = \underset{\alpha\in\mathbb{R}^p}{\arg\min}\{\lambda_1||\alpha||_1 + \frac{1}{2}||\mathbf{Wy} - \alpha||_2^2\} = \tag{17a}$$

$$= \min_{\alpha\in\mathbb{R}^p}\{\lambda_1||\alpha||_1 + \frac{1}{2}||\mathbf{Wy} - \mathbf{WW}^T\alpha||_2^2\} = \tag{17b}$$

$$= \min_{\alpha\in\mathbb{R}^p}\{\lambda_1||\alpha||_1 + \frac{1}{2}||\mathbf{W}(\mathbf{y} - \mathbf{W}^T\alpha)||_2^2\} = \tag{17c}$$

$$= \min_{\alpha\in\mathbb{R}^p}\{\lambda_1||\alpha||_1 + \frac{1}{2}||\mathbf{y} - \mathbf{W}^T\alpha||_2^2\} \tag{17d}$$

2.2 Exercise 2.2

Given $g(\mathbf{x}) := ||\mathbf{x}||_1$, the proximal function of λg is

$$prox_{\lambda g}(\mathbf{z}) = \underset{\alpha\in\mathbb{R}^p}{\arg\min}\{\lambda||\alpha||_1 + \frac{1}{2}||\alpha - \mathbf{z}||_2^2\} \tag{18}$$

which can be solved computing the gradient of the argument. Considering $\alpha \neq 0$

$$\bigtriangledown_\alpha \{\lambda||\alpha||_1 + \frac{1}{2}||\alpha - \mathbf{z}||_2^2\} = \lambda\mathbf{1}sign(\alpha) + (\mathbf{1}\alpha - \mathbf{z}) \tag{19}$$

setting it to zero, considering each entry and solving for α

$$\alpha > 0 \quad \rightarrow \quad \alpha = z - \lambda \quad \rightarrow \quad z > \lambda \tag{20}$$

$$\alpha < 0 \quad \rightarrow \quad \alpha = z + \lambda \quad \rightarrow \quad z < -\lambda \tag{21}$$

from the two conditions on z, we can imply that the variable must be bounded as follow

$$z > |\alpha| \tag{22}$$

This helds for each entry of $\mathbf{z}$. Considering now the case in which $\alpha = 0$, we need to take the subgradient into account, which has values from -1 to +1 and leads to the following constraint on z

$$\lambda[-1;1] - z = 0 \quad \rightarrow \quad z \in \lambda[-1;1] \quad \rightarrow \quad |z| \leq \lambda \tag{23}$$

Putting the three results together and broadcasting the result to the entire vector, we get a solution for (18):

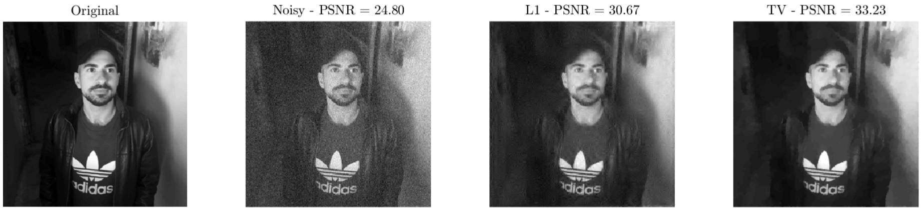
$$prox_{\lambda g}(\mathbf{z}) = \max(|\mathbf{z}| - \lambda, 0) \otimes sign(\mathbf{z}) \tag{24}$$

2.3 Exercise 2.3

The aim of this exercise is to reconstruct a noisy image (created with a standard deviation of 15) solving problem (13) exactly and problem (14) up to a given accuracy. In Figure 1, one can see, from left to right, the original image, the noisy image, the reconstructed one solving problem (13) and the reconstructed one solving problem (14) with 100 iterations and a tolerance of 10^{-5}. Above the images are also shown their corresponding Peak Signal-to-Noise Ratio (PSNR), calculated as follow

$$PSNR(I, \hat{I}) := 20 \log_{10} \left(\frac{\max(I)}{\sqrt{\frac{1}{N}||I - \hat{I}||_F^2}} \right) \tag{25}$$

where N is the dimension of the image and I and $\hat{I}$ are the original and noisy images, respectively.



Original

Noisy - PSNR = 24.80

L1 - PSNR = 30.67

TV - PSNR = 33.23

Figure 1: Original image, noisy image, reconstructed image solving problem (13) and reconstructed image solving problem (14), from left to right.

2.4 Exercise 2.4

Using the same original and noisy images as in section 2.3, the purpose of this exercise is to find the best regularization parameters λ_1 and λ_{TV}, as defined in equations (13) and (14), that results in the highest PSNR for the two reconstructed images. The best value for the L1 regularization parameter lies around 20, while for the TV regularization parameter it is around 14, as it can be seen in Figure 2.

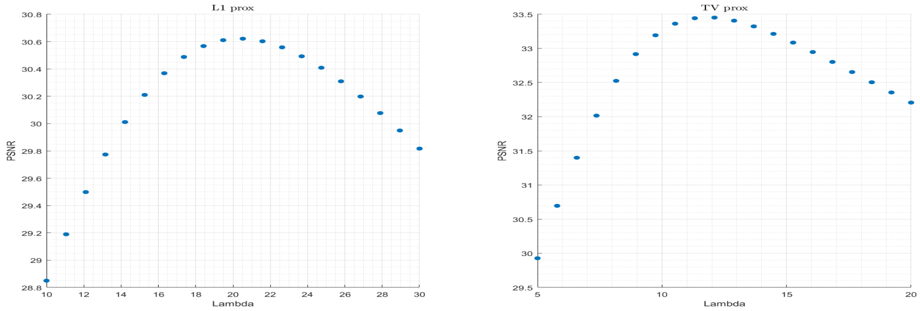


Figure 2: PSNR of the reconstructed images solving problems (13), on the left, and (14), on the right, using a range of different regularization parameters.

3 Non-smooth composite minimization and compressive MRI

In Compressive Magnetic Resonance Imaging (MRI), we are interested in recovering a structured signal $\mathbf{x}^\natural \in \mathbb{C}^p$ from measurements $\mathbf{b} = \mathbf{A}\mathbf{x}^\natural$, where $\mathbf{A} = \mathbf{P}_\Omega \mathbf{F} \in \mathbb{C}^{n\times p}$ is the dimensionality reducing Fourier measurement operator, with $n << p$ (in MRI imaging, the measurements are taken in Fourier domain). Exploiting the structures commonly found in natural images, it is possible to reconstruct MRI images using less number of its Fourier coefficients. We will reconstruct the original image by solving the following regularization problem

$$\min_{\alpha \in \mathbb{C}^p} \frac{1}{2}||\mathbf{b} - \mathbf{P}_\Omega \mathbf{F}\mathbf{W}^T \alpha||_2^2 + \lambda_1 ||\alpha||_1 \tag{26}$$

where $\mathbf{W}^T$ is the inverse 2D Wavelet transform, $\mathbf{F}$ is the 2D Fourier transform and $\mathbf{P}_\Omega \in \mathbb{R}^{n\times p}$ is an operator that selects only few ($n << p$) pixels from the vectorized image $\mathbf{x} \in \mathbb{R}^p$. From now on, we call $f(\alpha)$ and $g(\alpha)$ the first and the second term of the sum in (26), respectively.

3.1 Exercise 3.1

Since α and $\mathbf{b}$ are complex vectors, they can be represented with two different terms each: α_R and α_I, $\mathbf{b}_R$ and $\mathbf{b}_I$. Where $\alpha = \alpha_R + i\alpha_I$ and $\mathbf{b} = \mathbf{b}_R + i\mathbf{b}_I$. Then, putting $\mathbf{A} = \mathbf{A}_R + i\mathbf{A}_I := \mathbf{P}_\Omega \mathbf{F}\mathbf{W}^T$, f can be written as

$$\bar{f}\left(\begin{bmatrix}\alpha_R \\ \alpha_I\end{bmatrix}\right) = \frac{1}{2}\left\|\mathbf{b}_R - \begin{bmatrix}\mathbf{A}_R & -\mathbf{A}_I\end{bmatrix}\begin{bmatrix}\alpha_R \\ \alpha_I\end{bmatrix}\right\|_2^2 + \frac{1}{2}\left\|\mathbf{b}_I - \begin{bmatrix}\mathbf{A}_I & \mathbf{A}_R\end{bmatrix}\begin{bmatrix}\alpha_R \\ \alpha_I\end{bmatrix}\right\|_2^2 \tag{27}$$

Calling $\bar{f}_1$ and $\bar{f}_2$ the left and the right term in equation (27), respectively, the gradient of $\bar{f}$ can be computed as

$$\nabla \bar{f} = \nabla \bar{f}_1 + \nabla \bar{f}_2 \tag{28}$$

$$\nabla \bar{f}_1 = \begin{bmatrix} -\mathbf{A}_R^T \\ \mathbf{A}_I^T \end{bmatrix} \left(\mathbf{b}_R - \begin{bmatrix} \mathbf{A}_R & -\mathbf{A}_I \end{bmatrix} \begin{bmatrix} \alpha_R \\ \alpha_I \end{bmatrix} \right) \tag{29}$$

$$\nabla \bar{f}_2 = \begin{bmatrix} -\mathbf{A}_I^T \\ -\mathbf{A}_R^T \end{bmatrix} \left(\mathbf{b}_I - \begin{bmatrix} \mathbf{A}_I & \mathbf{A}_R \end{bmatrix} \begin{bmatrix} \alpha_R \\ \alpha_I \end{bmatrix} \right) \tag{30}$$

Then, the Lipschitz constant $L_{\bar{f}}$ of the gradient of $\bar{f}$ can be found as follow

$$\begin{align}
&\left\|\nabla \bar{f}(\alpha_R, \alpha_I) - \nabla \bar{f}(\gamma_R, \gamma_I)\right\| = \tag{31a}\\
&= \left\|\nabla \bar{f}(\alpha_R, \alpha_I) - \nabla \bar{f}(\alpha_R, \gamma_I) + \nabla \bar{f}(\alpha_R, \gamma_I) - \nabla \bar{f}(\gamma_R, \gamma_I)\right\| \leq \tag{31b}\\
&\leq \left\|\nabla \bar{f}(\alpha_R, \alpha_I) - \nabla \bar{f}(\alpha_R, \gamma_I)\right\| + \left\|\nabla \bar{f}(\alpha_R, \gamma_I) - \nabla \bar{f}(\gamma_R, \gamma_I)\right\| \leq \tag{31c}\\
&\leq L' \left(\|\alpha_R - \gamma_R\| + \|\alpha_I - \gamma_I\|\right) \leq \tag{31d}\\
&\leq \sqrt{2} L' \left\|(\alpha_R, \alpha_I) - (\gamma_R, \gamma_I)\right\| \tag{31e}\\
&\qquad \rightarrow L_{\bar{f}} = \sqrt{2} L' \tag{31f}
\end{align}$$

where $L' = \max(L_R, L_I)$. Setting $\mathbf{A}_1 := \begin{bmatrix} -\mathbf{A}_R^T \\ \mathbf{A}_I^T \end{bmatrix}$ and $\mathbf{A}_2 := \begin{bmatrix} -\mathbf{A}_I^T \\ -\mathbf{A}_R^T \end{bmatrix}$, L_R and L_I are defined as follow

$$\begin{align}
&\left\|\nabla \bar{f}(\alpha_R, \alpha_I) - \nabla \bar{f}(\alpha_R, \gamma_I)\right\| = \tag{32a}\\
&= \left\|\left(\mathbf{A}_1 \mathbf{A}_I - \mathbf{A}_2 \mathbf{A}_R\right)(\alpha_I - \gamma_I)\right\| \leq \tag{32b}\\
&\leq \left\|\mathbf{A}_1 \mathbf{A}_I - \mathbf{A}_2 \mathbf{A}_R\right\| \|\alpha_I - \gamma_I\| := \tag{32c}\\
&:= L_I \|\alpha_I - \gamma_I\| \tag{32d}
\end{align}$$

$$\begin{align}
&\left\|\nabla \bar{f}(\alpha_R, \gamma_I) - \nabla \bar{f}(\gamma_R, \gamma_I)\right\| \leq \tag{33a}\\
&\leq \left\|\mathbf{A}_2 \mathbf{A}_I - \mathbf{A}_1 \mathbf{A}_R\right\| \|\alpha_R - \gamma_R\| := \tag{33b}\\
&:= L_R \|\alpha_R - \gamma_R\| \tag{33c}
\end{align}$$

Since $L_R = L_I = 1$, $L' = 1$ and $L_{\bar{f}} = \sqrt{2}$.

3.2 Exercise 3.2 and 3.3

In this exercise we will implement the FISTA algorithm with exact non-monotonicity test as restart criterion for solving

$$\min_{\alpha_R, \alpha_I} \bar{f}(\alpha_R, \alpha_I) + g(\alpha_R, \alpha_I) \tag{34}$$

where $\bar{f}$ and g are convex functions. The algorithm is then used to solve problem (26), in order to do the reconstruction of a 256 × 256 brain image.
In Figure 3, it can be seen that the best regularization parameter for reconstructing the image is around $3 \cdot 10^{-4}$. In Figure 4, the original and the reconstructed brain images are shown. The FISTA algorithm was run for 100 iterations with a tolerance of 10^{-5}.

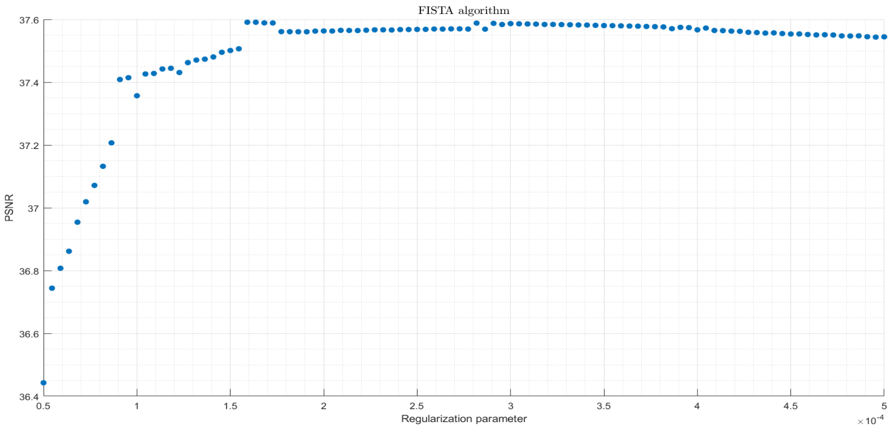


Figure 3: PSNR of the reconstructed brain image solving problem (26), running FISTA on a range of different regularization parameters.

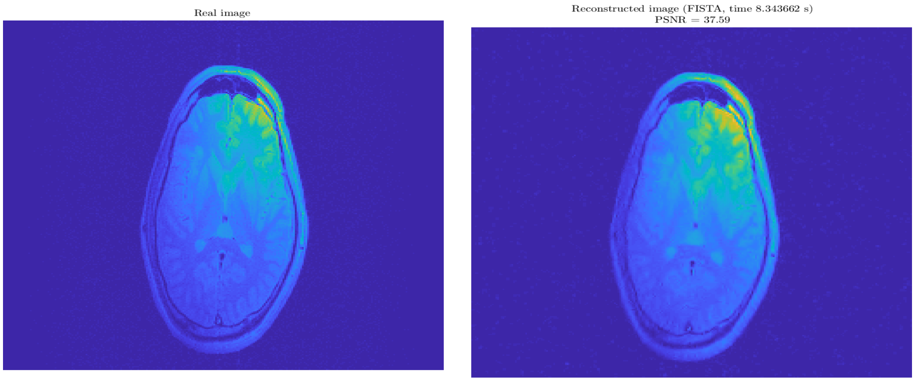


Figure 4: The original and the reconstructed brain images, using a regularization parameter of $3 \cdot 10^{-4}$, 100 iterations and a tolerance of 10^{-5}.

3.3 Exercise 3.4

We now turn our attention to the linear estimator

$$\hat{\mathbf{x}} = \Psi^* \mathbf{P}_\Omega^T \mathbf{b} \tag{35}$$

where Ψ is the Fourier transform and $\mathbf{b} = \mathbf{P}_\Omega \Psi \mathbf{x}^\natural$.
We consider three different sets Ω. The first is made of 20% of the pixel locations picked at random, the second is the set given in the previous exercise and the third is the best set for the brain image. In Figure 5, 6 and 7, the PSNR of the reconstructed images are shown, comparing FISTA (run again for 100 iterations) and the linear estimator (35), along with the time to solve the problems and the masks used (in black the taken pixels). It can be seen that solving the

linear problem is in general much faster than using the FISTA algorithm. For the given mask, FISTA performs better since the PSNR of the reconstructed image is higher than the one of the image reconstructed by the linear estimator. Finally, for the last mask, both FISTA and the linear solver estimate the image very well.

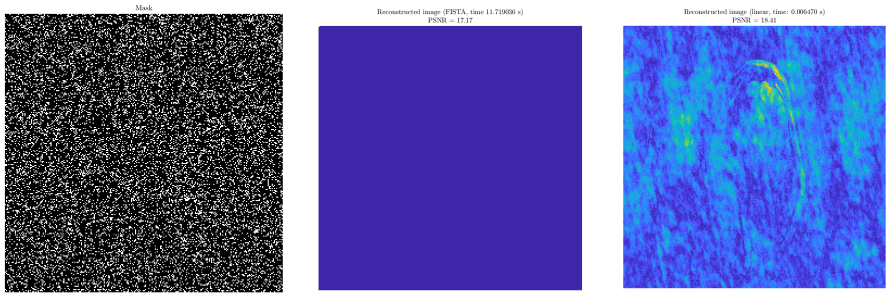

Mask
Reconstructed image (FISTA, time 11.719036 s)
PSNR = 17.17
Reconstructed image (linear, time: 0.006470 s)
PSNR = 18.41

Figure 5: The random mask, the reconstructed brain image using FISTA and the reconstructed brain image using the linear estimator, from left to right.

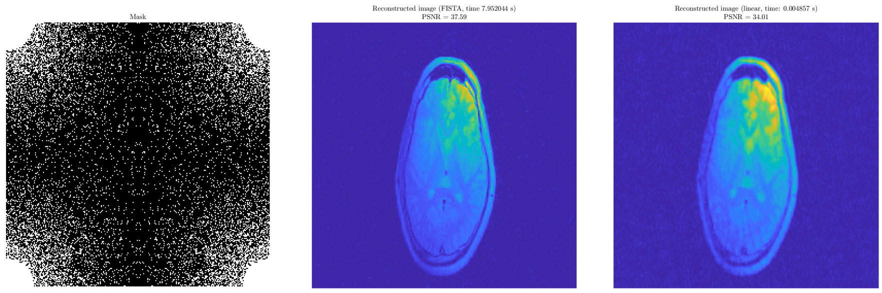

Mask
Reconstructed image (FISTA, time 7.952044 s)
PSNR = 37.59
Reconstructed image (linear, time: 0.004857 s)
PSNR = 34.01

Figure 6: The given mask, the reconstructed brain image using FISTA and the reconstructed brain image using the linear estimator, from left to right.

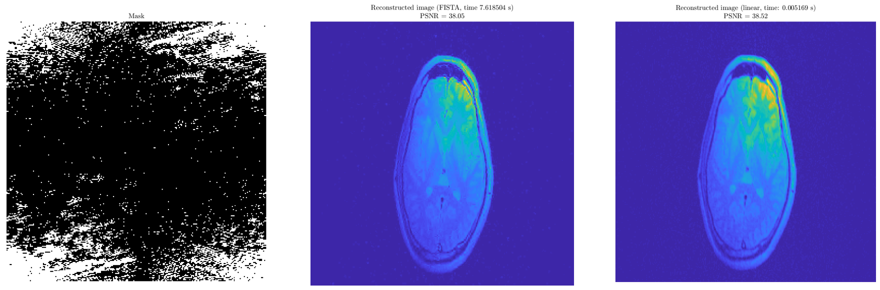

Mask
Reconstructed image (FISTA, time 7.618504 s)
PSNR = 38.05
Reconstructed image (linear, time: 0.005169 s)
PSNR = 38.52

Figure 7: The best mask, the reconstructed brain image using FISTA and the reconstructed brain image using the linear estimator, from left to right.

4 Image in-painting

Image in-painting consists in reconstructing the missing parts of an image. In this part, we are going to study the convergence of different methods related to FISTA, as well as different restart strategies. We consider a subsampled image $\mathbf{b} = \mathbf{P}_\Omega \mathbf{x}$, where $\mathbf{P}_\Omega \in \mathbb{R}^{n \times p}$ is again an operator that selects only few pixels from the vectorized image $\mathbf{x} \in \mathbb{R}^p$. We can reconstruct the original image by solving

$$\min_{\alpha \in \mathbb{R}^p} \frac{1}{2} ||\mathbf{b} - \mathbf{P}_\Omega \mathbf{W}^T \alpha||_2^2 + \lambda_1 ||\alpha||_1 \tag{36}$$

$$\min_{\mathbf{x} \in \mathbb{R}^p} \frac{1}{2} ||\mathbf{b} - \mathbf{P}_\Omega \mathbf{x}||_2^2 + \lambda_{TV} ||\alpha||_{TV} \tag{37}$$

4.1 Exercise 4.1

We start with a 1024×1024 image and randomly and uniformly subsample 40% of its pixels. We perform a parameter sweep over the regularization parameters λ_1 and λ_{TV} used in FISTA to solve problems (36) and (37), with 100 iterations for FISTA and 20 to solve the TV proximal operator, as shown in Figure 8. Thus, picking $\lambda_1 = 5$ and $\lambda_{TV} = 1.3$, we reconstruct the image with 100 iterations for FISTA, 50 iterations to solve the TV proximal operator and a tolerance of 10^{-5}, as shown in Figure 9 and 10.

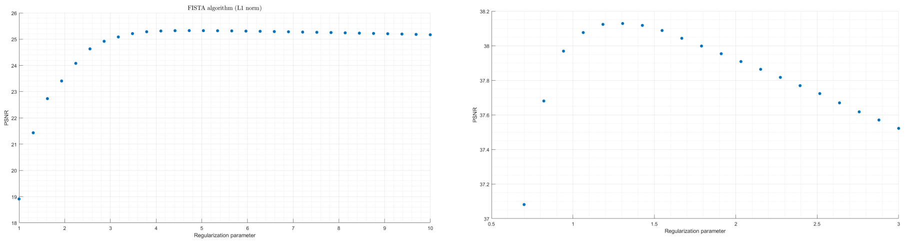


Figure 8: PSNR of the reconstructed image solving problem (36) on the left and (37) on the right, running FISTA on a range of different regularization parameters.

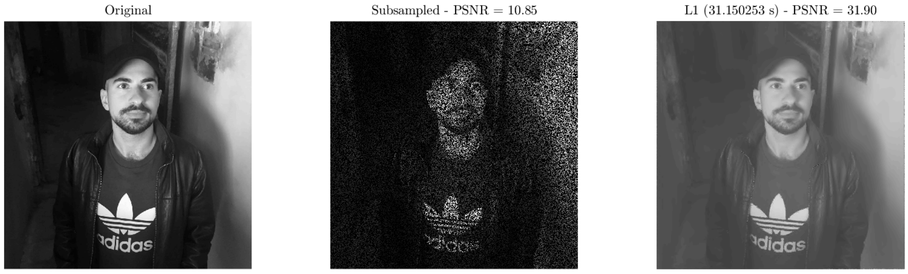


Figure 9: Original image, noisy image and reconstructed image solving problem (36), from left to right.

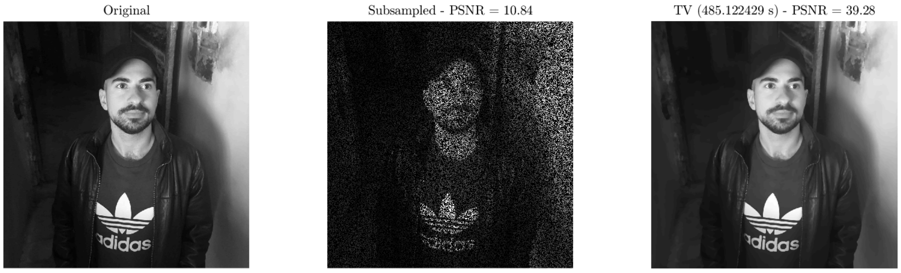


Figure 10: Original image, noisy image and reconstructed image solving problem (37), from left to right.

4.2 Exercise 4.2

We now perform convergence analysis and study the convergence to the optimal solution $\mathbf{x}^*$ of problem (36) as well as to the ground truth. We again uniformly subsample 40% of the pixels of the image and we set the regularization parameter to be $\lambda_1 = 10$. To find $\mathbf{x}^*$ we run FISTA with the non-monotonicity test as restart criterion for 5000 iterations with tolerance 10^{-15}. In Figure 11, 12, 13 and 14, the convergence to $\mathbf{x}^*$ as $\log\left(|F(\mathbf{x}_k) - F^*|/F^*\right)$ are shown, for the following variations of FISTA (2000 iterations, stopping criterion $\log\left(|F(\mathbf{x}_k) - F^*|/F^*\right) < 10^{-15}$):

- Iterative Soft Thresholding Algorithm (ISTA);
- FISTA without restart;
- FISTA with fixed iteration restart (every 25, 50, 100 and 200 iterations);
- FISTA with gradient scheme restart.

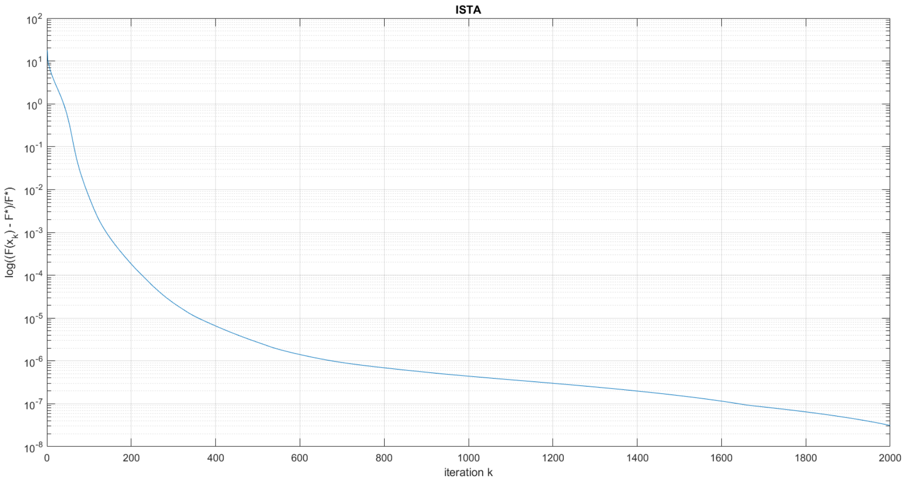


Figure 11: Convergence to the optimal solution of ISTA.

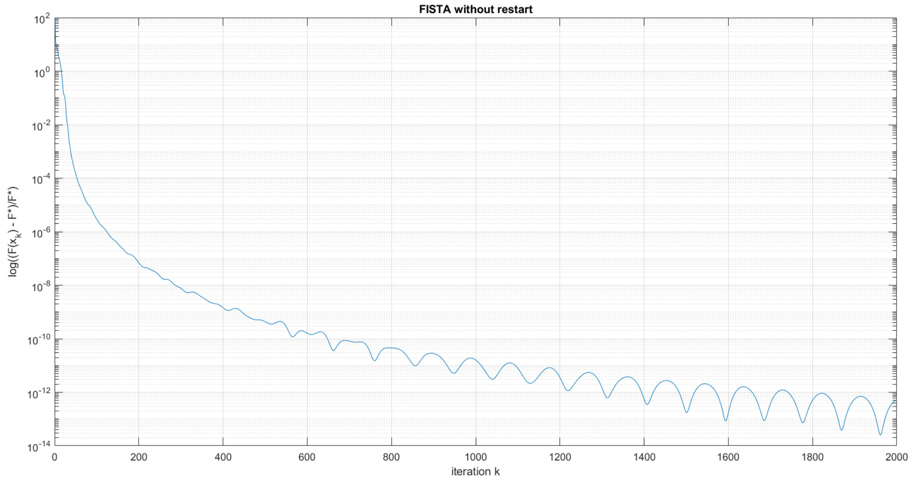
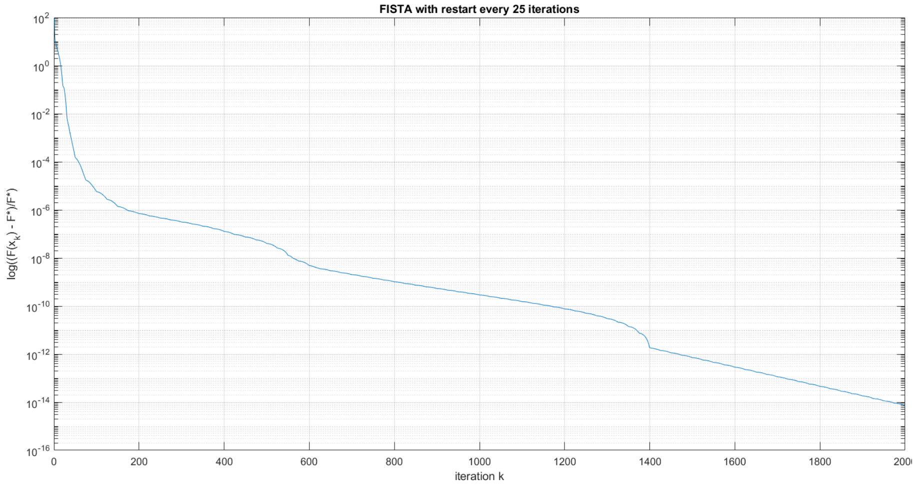
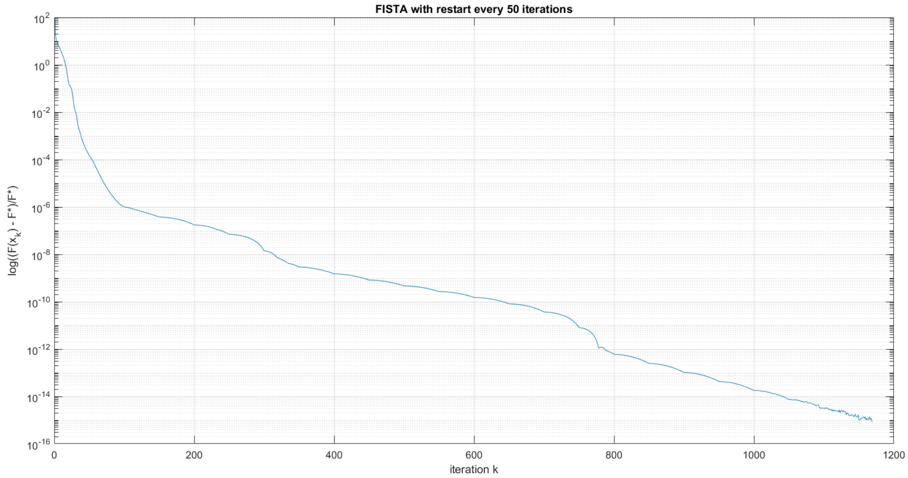
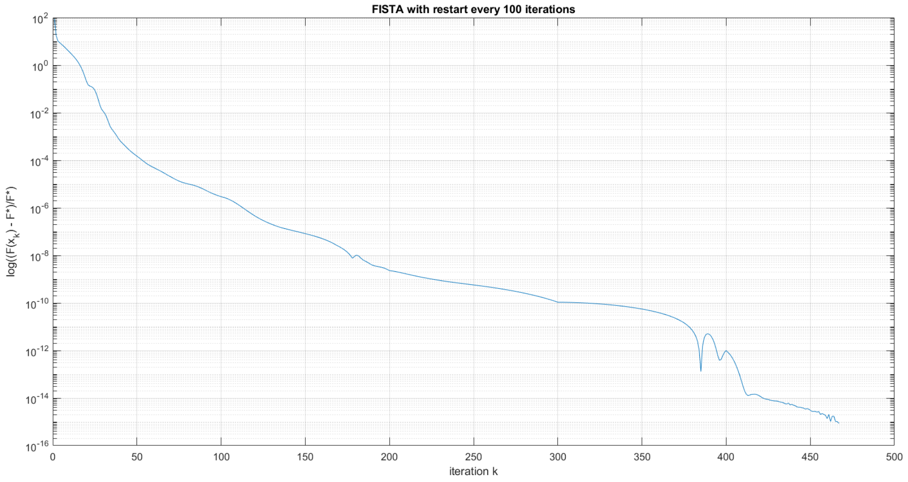


Figure 12: Convergence to the optimal solution of FISTA without restart.


FISTA with restart every 25 iterations
log((F(x_k) - F*)/F*)
10^2
10^0
10^-2
10^-4
10^-6
10^-8
10^-10
10^-12
10^-14
10^-16
0
200
400
600
800
1000
1200
1400
1600
1800
200
iteration k


FISTA with restart every 50 iterations
log((F(x_k) - F*)/F*)
10^2
10^0
10^-2
10^-4
10^-6
10^-8
10^-10
10^-12
10^-14
10^-16
0
200
400
600
800
1000
1200
iteration k



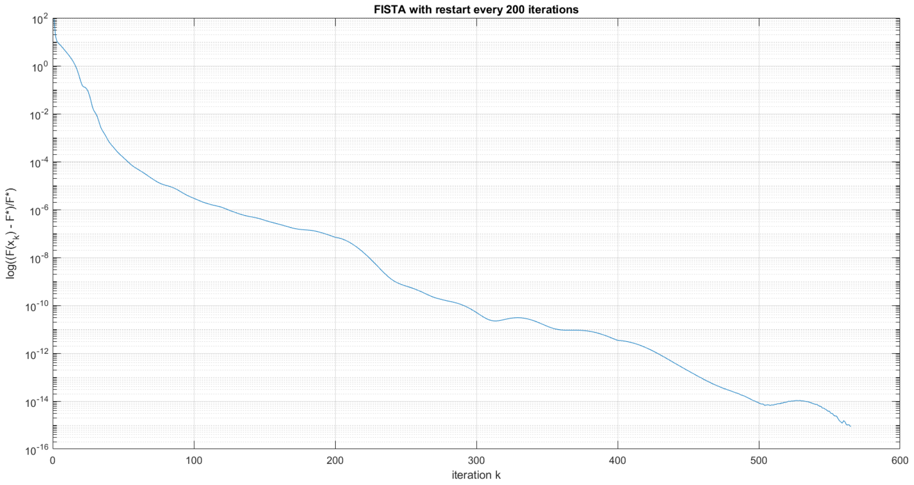


Figure 13: Convergence to the optimal solution of FISTA with fixed restart (every 25, 50, 100 and 200 iterations.

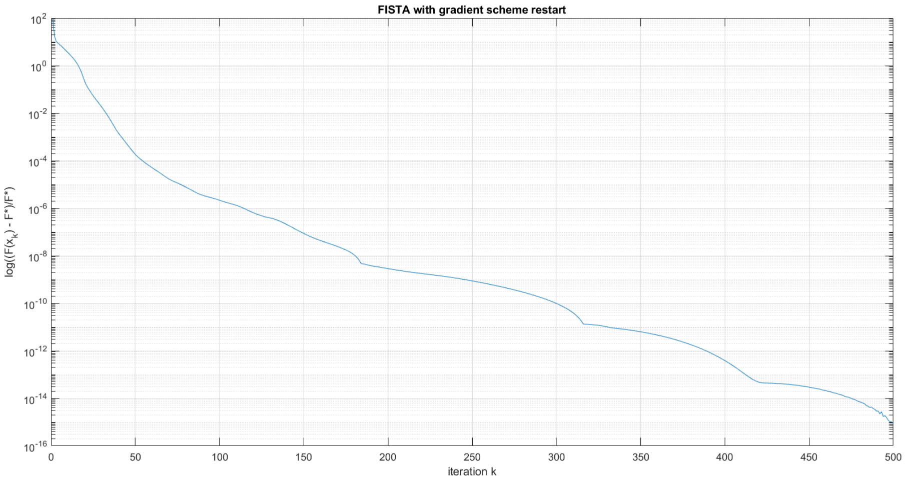


Figure 14: Convergence to the optimal solution of FISTA with gradient scheme restart.

It can be seen that FISTA with fixed restart every 50, 100 and 200 iterations and FISTA with gradient scheme restart reach the stopping criterion. FISTA with restart every 25 iterations has a precision of 10^{-14} after 2000 iterations, FISTA without restart oscillates around a precision of 10^{-13} and ISTA ends with a precision of around 10^{-7}.
Finally, in Figure 15, 16, 17 and 18, the convergences to the ground truth $\mathbf{x}^\natural$ as $\log\left(|F(\mathbf{x}_k) - F^\natural|/F^\natural\right)$ of the same algorithms as above (1000 iterations) are shown. In this case, it is clear that none of the algorithms manage to reach the ground truth (they end with a distance of around 0.44). This can be a consequence of being stuck in a local minimum.

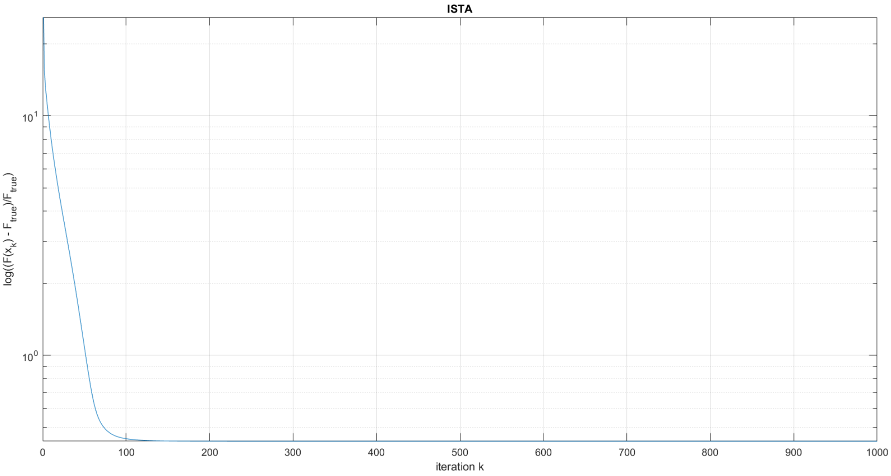


Figure 15: Convergence to the ground truth of ISTA.

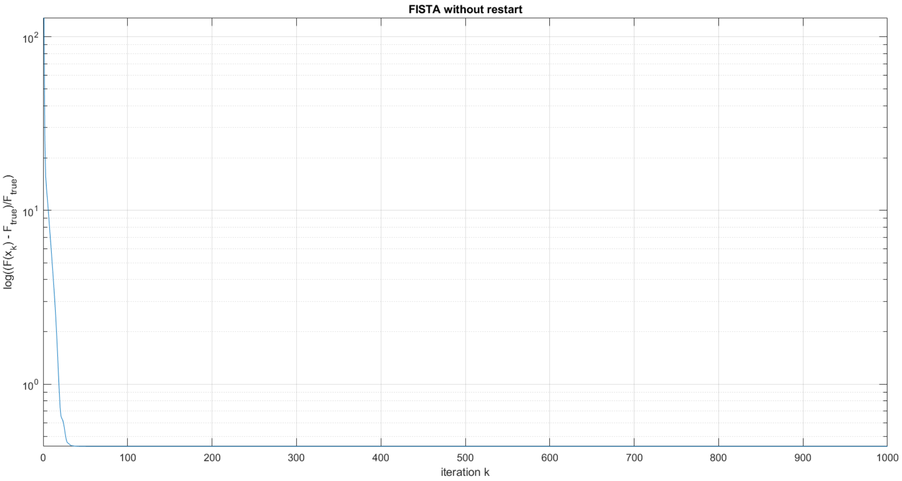
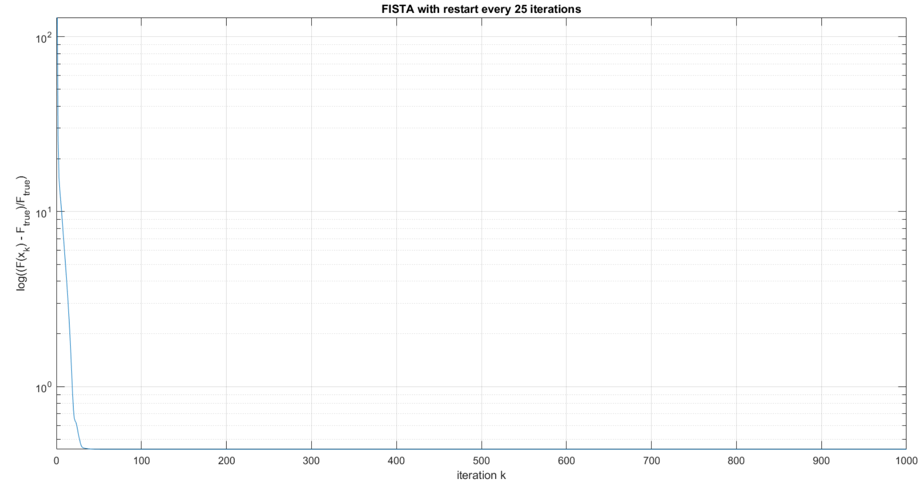
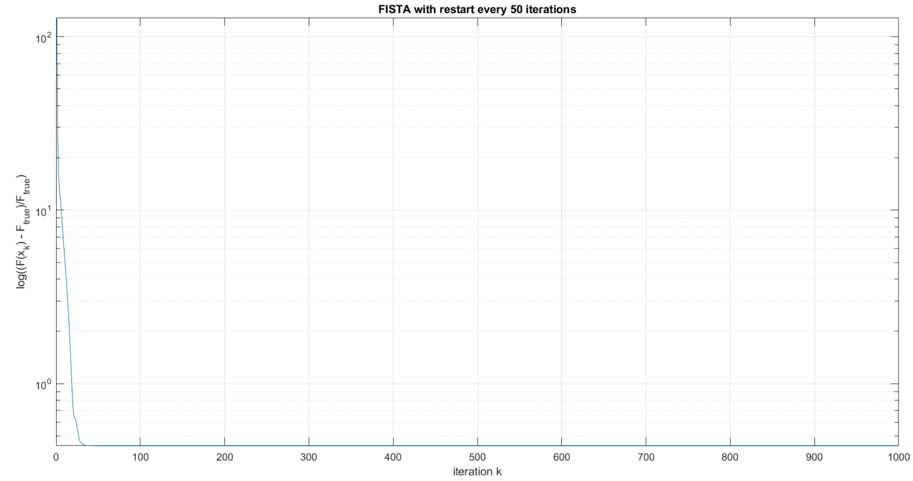
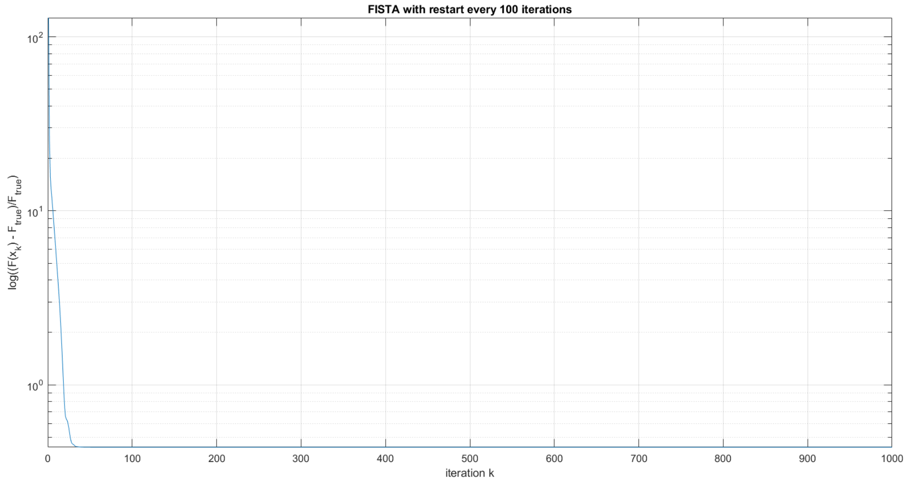


Figure 16: Convergence to the optimal solution of FISTA without restart.


FISTA with restart every 25 iterations
log((F(x_k) - F_true)/F_true)
10^2
10^1
10^0
0
100
200
300
400
500
600
700
800
900
1000
iteration k


FISTA with restart every 50 iterations
log((F(x_k) - F_true)/F_true)
10^2
10^1
10^0
0
100
200
300
400
500
600
700
800
900
1000
iteration k



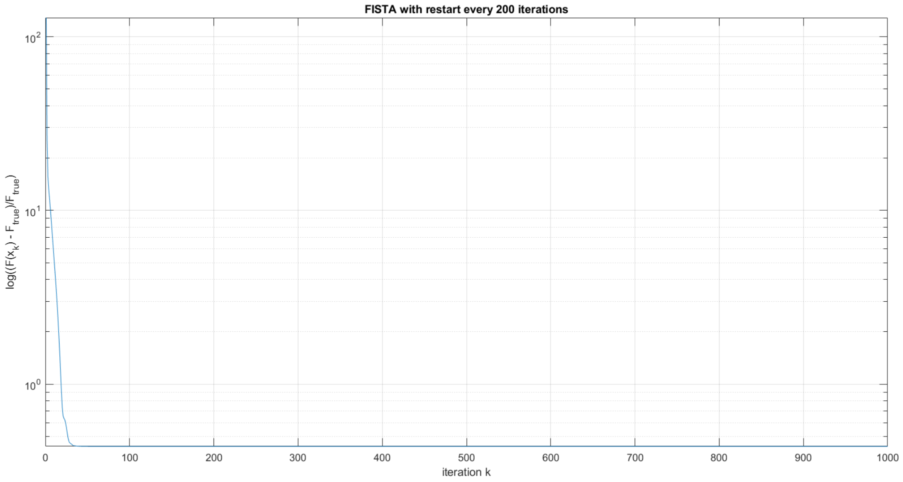


Figure 17: Convergence to the optimal solution of FISTA with fixed restart (every 25, 50, 100 and 200 iterations.

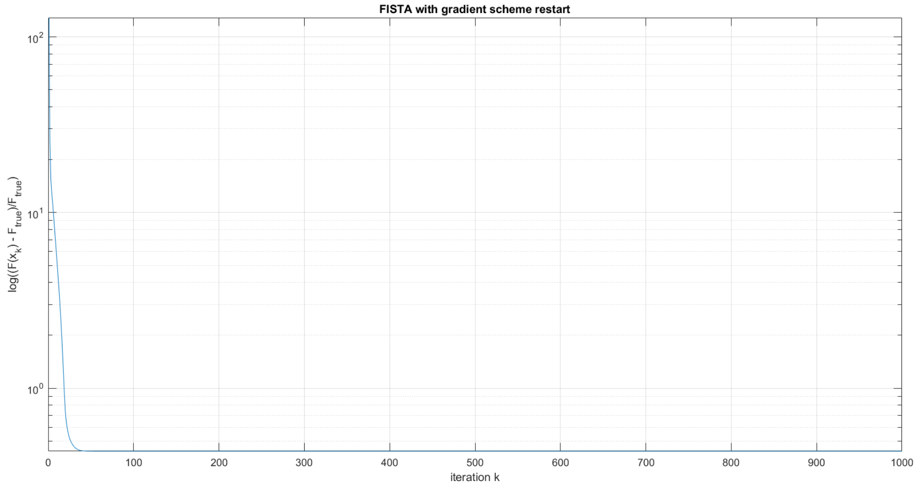


Figure 18: Convergence to the optimal solution of FISTA with gradient scheme restart.