Splitting methods
for fourth-order image inpainting

Luca Calatroni
Supervisor: Dr. Carola Schönlieb

November 6, 2012

1 Image inpainting

One of the most important applications of image processing is the process of filling the missing parts of damaged images using the information obtained from the surrounding areas. We can think of this process as, essentially, a type of interpolation called inpainting. Such a technique can be used for restoring images within parts damaged due to, for example, wear and tear (like the restoration of ancient frescoes or the scratch removal in old photographs) or for needs of reducing artifacts in MRI-imaging reconstructions.

From a mathematical point of view, image inpainting can be described as follows: given an image $f$ on an image domain $\Omega$, the task is to reconstruct the original image $u$ in the damaged domain $D \subset \Omega$, called the inpainting domain (see Figure 1).

![Figure 1: The inpainting problem: the image is missing on $D$ which can have many connected components with arbitrarily shapes and sizes.](image)

Various approaches to solve this task exist in the literature. A basic division of inpainting methodologies yields two groups, global methods (interpolation is performed with all the available image contents) and local methods.
(interpolation is based only on the surrounding parts of the image). In what follows, we will only consider methods from the latter class phrased as variational problems or partial differential equations.

More precisely, let \( \Omega \subset \mathbb{R}^2 \) be an open and bounded domain with Lipschitz boundary and let \( B_1, B_2 \) two Banach spaces with, typically, \( B_2 \subseteq B_1 \). Let \( f \in B_1 \) denote the given image and \( D \subset \Omega \) the inpainting domain. A general variational approach in image inpainting can be generally formulated as a minimization problem for a functional \( E : B_2 \to \mathbb{R} \), that is:

\[
\min_{u \in B_2} \{ E(u) := R(u) + \| \lambda(f - u) \|^2_{B_1} \} \tag{1}
\]

where \( R : B_2 \to \mathbb{R} \) denotes the regularizing term and

\[
\lambda(x) := \begin{cases} 
\lambda_0 & \text{if } x \in \Omega \setminus D \\
0 & \text{if } x \in D 
\end{cases}
\]

is the characteristic function of the set \( \Omega \setminus D \) multiplied by a typically large constant \( \lambda_0 \). The norm \( \| \lambda(f - u) \|^2_{B_1} \) is the so called fidelity term of the inpainting approach. The inclusion \( B_2 \subseteq B_1 \) in general represents the smoothing effect of \( R \) on the minimizer \( u \). We note that such an approach poses the inpainting problem in the whole domain \( \Omega \), not in the inpainting domain \( D \). This allows us to deal even with inpainting problems in which \( D \) is not connected and has arbitrary regularity features.

Depending on the choice of the regularizing term \( R \) and the spaces \( B_1, B_2 \), various inpainting approaches have been developed. Under some regularity assumptions on the minimizer \( u \) in (1), we can consider the corresponding Euler-Lagrange equation, just taking the Fréchet derivative of the functional \( E \). The highest order of the derivatives of such an equation determines the so called order of the inpainting method used. In the case \( B_1 = B_2 = L^2(\Omega) \) with the choice \( R(u) = \| \nabla u \|^2 \), for instance, the underlying second order PDE related to the minimization process is the following:

\[
u_t = -\Delta u + \lambda(f - u) \tag{2}
\]

where the dependence on time arises by embedding the process into a steepest-descent. The equation (2) is then an example of second order method for inpainting, typically called linear inpainting.

Second order variational inpainting methods (like, for instance, (2) and the TV model) have been largely used, but they present drawbacks in connecting edges over large distances (thus failing the so-called connectivity principle) and in the smooth propagation of level lines into the damaged domain. The reason is due to the penalization of the length within the minimizing
process with a second order regularizer (see [13, Section 1.2] for further details).

In order to solve such a problem, some higher order methods have been developed, at first using third-order methods (see, e.g., Bertalmio et al. [2]), then using fourth-order methods (as the Eulers elastica models presented in [5], the Cahn-Hilliard [3] and TV-$H^{-1}$ [4] ones). For the well-posedness of these problems another boundary condition is required, but on the other hand they are able to fulfill the connectivity principle.

From a numerical point of view, discretizing explicitly a fourth-order evolution equation may restrict the time-step $\Delta t$ up to order $(\Delta x)^4$ where $\Delta x$ denotes the step size of the spatial grid: this kind of methods requires a huge amount of iterations and then these methods are computationally prohibitive. Efficient numerical schemes for inpainting methods of higher-order is an active area of research. In this work we present some methods developed to solve this problem based on a general idea presented in the next section.

2 The splitting idea

The general idea behind any so called splitting method is breaking down a complicated problem into smaller (and, typically, easier to approach) parts, such that these smaller problems can be solved efficiently. Different splitting methods have been presented both to solve ODEs and PDEs. The main idea is the splitting of the operator defining the problem into different components. A particular case of operator splitting is the dimensional splitting where the decompositions is performed such that all the computations become actually one-dimensional. Typically, the original problem is decomposed into more manageable substeps (called fractional steps) which yield approximations of the solution of the full problem, but which, generally, are not consistent. In the following subsection we present some general examples of operator splitting methods. Namely, we describe Strang splitting and convexity splitting methods and, as example of dimensional splitting, the alternating direction implicit (ADI) methods. In the next section we will apply them to the problem of inpainting.

2.1 A second-order symmetrical splitting method: the Strang splitting

Let us consider a linear, homogeneous ODE system with an initial condition:

$$u'(t) = Au(t), \quad t > 0, \quad u(0) = u_0$$  \hspace{1cm} (3)
and assume that the operator $A$ can be splitted as $A = A_1 + A_2$. Considering a uniform time-discretized grid with nodes $t_i$, $i \geq 1$ and setting $\Delta t := t_{n+1} - t_n$ for every $n$, we know that the solution of (3) is given by:

$$u(t_{n+1}) = e^{\Delta t A} u(t_n).$$

(4)

Indicating with $u_n$ the approximation of the exact solution at the point $t_n$, that is $u_n \approx u(t_n)$, we can introduce for every time step $n - n + 1$ a middle step $n + 1/2$ and solving auxiliary problems starting from $u_n$ to obtain $u_{n+1}$. Solving at first:

$$\begin{cases}
\frac{d}{dt} u^{\ast}(t) = A_1 u^{\ast}(t) & \text{for } t_n < t \leq t_{n+1}/2 \quad \text{with } u^{\ast}(t_n) = u_n \\
\frac{d}{dt} u^{**}(t) = A_2 u^{**}(t) & \text{for } t_n < t \leq t_{n+1}/2 \quad \text{with } u^{**}(t_n) = u^{\ast}(t_{n+1}/2)
\end{cases}$$

(5)

one after another starting from $u_n$, we can take $u_{n+1}/2 = u^{**}(t_{n+1}/2)$, thus completing the half step. We can now redo the same, interchanging the order of application of $A_1$ and $A_2$, starting now from $u_{n+1}/2$ and solving

$$\begin{cases}
\frac{d}{dt} u^{\ast}(t) = A_2 u^{\ast}(t) & \text{for } t_{n+1}/2 < t \leq t_{n+1} \quad \text{with } u^{\ast}(t_{n+1}/2) = u_{n+1}/2 \\
\frac{d}{dt} u^{**}(t) = A_1 u^{**}(t) & \text{for } t_{n+1}/2 < t \leq t_{n+1} \quad \text{with } u^{**}(t_{n+1}/2) = u^{\ast}(t_{n+1})
\end{cases}$$

(6)

one after another, taking $u_{n+1} = u^{**}(u_{n+1})$ to complete the splitting integration step. This gives:

$$u_{n+1} = e^{\frac{\Delta t}{2} A_1} e^{\frac{\Delta t}{2} A_2} u_{n+1}/2 = \left( e^{\frac{\Delta t}{2} A_1} e^{\frac{\Delta t}{2} A_2} \right) \left( e^{\frac{\Delta t}{2} A_2} e^{\frac{\Delta t}{2} A_1} \right) u_n$$

$$= e^{\frac{\Delta t}{2} A_1} e^{\Delta t A_2} e^{\frac{\Delta t}{2} A_1} u_n$$

(7)

which leads symmetry and accuracy to the problem. Of course, replacing (4) by (7) introduces an error, the splitting error. Inserting the exact solution $u$ to the original problem into (7), we have:

$$u(t_{n+1}) = e^{\frac{\Delta t}{2} A_1} e^{\Delta t A_2} e^{\frac{\Delta t}{2} A_1} u(t_n) + \Delta t \rho_n$$

where $\rho_n$ is the local truncation error. By series expansion it is possible to show that $\rho_n$ is of order two, thus revealing a formal consistency of such an order. Generalization of such method exists both in the case the step size $\Delta t$ is not constant and when the operator splitting has more than two components. For further details see [11].
2.2 Convexity splitting

The idea of \textit{convexity splitting} was originally introduced to solve energy minimizing equations (see [7]): let $V \in C^2(\mathbb{R}^n, \mathbb{R})$ a functional. Let $\Omega$ be the spatial domain of the data. We consider the following problem:

\[
\begin{aligned}
  &\text{find } u \in \mathbb{R}^N \text{ such that } \\
  &\quad \begin{cases}
    u_t = -\nabla V(u) & \text{in } \Omega \\
    u(\cdot, 0) = u_0 & \text{in } \Omega.
  \end{cases}
\end{aligned}
\]

(8)

The main idea of convexity splitting is to split the functional $V$ into a convex and a concave part, that is decomposing $V$ as:

\[
V(u) = V_c(u) - V_e(u)
\]

where both $V_c$ and $V_e$ are strictly convex for all $u \in \mathbb{R}^n$. We can write down then a semi-implicit discretization of (8) given by:

\[
U_{k+1} - U_k = -\Delta t(\nabla V_c(U_{k+1}) - \nabla V_e(U_k)).
\]

where by $U_k$ we indicate the numerical approximation of $u(k\Delta t)$. As we are going to see in the following, the convexity splitting idea can be applied also to more general evolution equations, in particular to those which do not follow a variational principle \footnote{We recall that saying that the solutions of Euler-Lagrange equations follow a \textit{variational principle} means that they cause the vanish of the Fréchet derivative.}. Such methods have a long tradition in several parts of numerical analysis. They have been applied in finite element and finite difference approximations (see [7], [14] and [13])). Using a semi-implicit discretization scheme, the application of such a method consists in treating implicitly in time the convex part and explicitly the concave one, thus having a balance between the nice properties of convex functions and the computational effort to solve implicit methods.

2.3 The ADI method

Consider the initial value problem for a system of ODEs:

\[
\begin{aligned}
  &\text{suppose that it is possible to decompose the function } F \text{ into a number of component functions such that each of these components acts just on one single direction of the space, that is:} \\
  &\quad F(t, u) = F_0(t, u) + F_1(t, u) + \cdots + F_s(t, u) \quad \text{for some } s \geq 1
\end{aligned}
\]

(9)
where the component $F_j, 1 \leq j \leq s$ is acting in the $j$-th direction of $\mathbb{R}^s$. We define now a method which uses decomposition [9] by treating in each stage of the calculation at most one of the components $F_i$ implicitly. We assume instead that the $F_0$ component can be treated explicitly (or, equivalently, it is not stiff) and contains contributes coming from mixed directions, whereas the $F_j, j \geq 1$ are stiff and so they need an implicit treatment. We consider the following scheme, the so-called ADI Douglas method, depending on a parameter $\theta > 0$:

$$
\begin{align*}
\begin{cases}
v_0 &= u_n + \Delta t F(t_n, u_n), \\
v_j &= v_{j-1} + \theta \Delta t (F_j(t_{n+1}, v_j) - F_j(t_n, u_n)) \quad \text{for } j = 1, 2, \ldots, s, \\
u_{n+1} &= v_s
\end{cases}
\end{align*}
$$

where the $v_j$ are the internal vectors for the step from $t_n$ to $t_{n+1}$ and we have used the same notation as in Section 2.1. Heuristically, the calculation of vectors $v_j, j = 1, \ldots, s$ serves to stabilize the forward Euler method used to calculate $v_0$. It can be verified that the order of the scheme (10) is equal to 2 whenever $F_0 = 0$ and $\theta = 1/2$ and it is of order 1 otherwise (see [11]).

Some extension of the Douglas method has been suggested in order to increase the order of the method independently of $F_0$ (see [9], [10]): this is basically done introducing further additional stages which, however, do not badly influenced the property of unconditional stability of such methods, which holds in any case. These methods have been largely applied in the study of convection-diffusion equations such as in financial option pricing. In this work we will use a suitable generalization of (10), applying it to our problem of inpainting.

### 3 Inpainting models based on splitting ideas

We present now some meaningful applications of the presented splitting methods to the problem of inpainting presented in Section 1. We point out that the models (5)-(6) and (10) refer to the solution of ODEs system while the methods we are going to deal with are based on a PDE approach. However, as it is well-known, discretizing the spatial derivatives of an evolution PDE with finite differences we get exactly very large systems of ODEs. We can then apply the splitting technique in these cases.

#### 3.1 TV-$H^{-1}$ inpainting

We present now a recently proposed approach to inpainting for grayvalues images, called TV-$H^1$ inpainting (see [13] for further details). It can be seen
as a generalization of the Cahn-Hilliard inpainting approach for grayvalues image. In TV-$H^1$ method the inpainted image $u$ of the given image $f \in L^2(\Omega)$ evolves via:

$$
\begin{cases}
    u_t = -\Delta \xi + \lambda(f-u) \\
    \xi \in \partial TV(u)
\end{cases}
$$

where the total variation $TV(u)$ is defined as:

$$
TV(u) := \begin{cases}
    |Du|_{\Omega} & \text{if } \|u\|_\infty \leq 1 \text{ a.e. in } \Omega \\
    +\infty & \text{otherwise.}
\end{cases}
$$

For the definition of the quantities involved see Appendices A-B. The $L^\infty$-bound of the functional $TV(u)$ is justified remembering that we are only considering images $u$ which take values just in the interval $[-1, 1]$ which corresponds to the grayvalues taken by $u$. In our approach we approximate the element $\xi$ by the square root regularization of $\nabla \cdot (\nabla u / |\nabla u|_\varepsilon)$, i.e. we consider for $0 < \varepsilon \ll 1$ the smoothed approximation given by:

$$
u_t = -\Delta \nabla \cdot \left( \frac{\nabla u}{|\nabla u|_\varepsilon} \right) + \lambda(f-u) \quad (11)
$$

where, for notational simplicity we write $|\nabla u|_\varepsilon$ to indicate the regularizing term $\sqrt{|\nabla u|^2 + \varepsilon}$.

### 3.1.1 Convexity splitting for TV-$H^{-1}$ inpainting

We present an application of a convexity splitting method to solve (11). Equation (11) is not a gradient system (see (8)), nonetheless we can consider two different gradient flows and apply the splitting idea separately. The regularizing term can be modeled by a gradient flow in $H^{-1}$ (i.e. taking the Fréchet derivative with respect to the $H^{-1}$ norm) of the energy:

$$E_1(u) := \int_\Omega |\nabla u(x)|_\varepsilon dx.$$

Now, we split $E_1$ as $E_{1,e} - E_{1,e}$, with

$$E_{1,e}(u) := \int_\Omega \frac{C_1}{2} |\nabla u(x)|_\varepsilon^2 dx$$

and

$$E_{1,e}(u) := \int_\Omega \left( -|\nabla u(x)|_\varepsilon + \frac{C_1}{2} |\nabla u(x)|_\varepsilon^2 \right) dx.$$
where the constant $C_1$ has to be chosen such that $E_{1,c}$ and $E_{1,e}$ are strictly convex. As $E_{1,c}$ is strictly convex for every choice of $C_1$, the convexity condition for the second derivative for $E_{1,e}$ gives easily that $C_1 > \frac{1}{\sqrt{\varepsilon}}$. The second term of (11) can be derived instead from a gradient flow in $L^2$ for the energy:

$$E_2(u) := \int_{\Omega} (\lambda(x)(f(x) - u(x)))^2 \, dx.$$ 

Now we split $E_2$ as $E_{2,c} - E_{2,e}$ with

$$E_{2,c}(u) := \frac{1}{2} \int_{\Omega} C_2 |u(x)|^2 \, dx$$

and

$$E_{2,e}(u) := \frac{1}{2} \int_{\Omega} \left( -(\lambda(x)(f(x) - u(x)))^2 + C_2 |u(x)|^2 \right) \, dx$$

where now the constant $C_2$ has to be chosen such that $C_2 > \lambda_0$. Denoting by $U_k$ the solution of the time-discrete equation at time $k\Delta t$, we have that the resulting semi-implicit discrete time-stepping scheme for an initial condition $U_0 = u_0$ is:

$$\frac{U_{k+1} - U_k}{\Delta t} = -\delta_{H^{-1}}(E_{1,c}(U_{k+1}) - E_{1,c}(U_k)) - \delta_{L^2}(E_{2,c}(U_{k+1} - E_{2,e}(U_k))$$

where $\delta_{H^{-1}}$ and $\delta_{L^2}$ represent the Fréchet derivatives with respect to the norm in $H^{-1}$ and $L^2$, respectively.

Here the quantities are assumed to be discretized in space as follows: the discrete gradient $\nabla$ is computed using forward differences, the divergence $\nabla \cdot$ using backward differences and then the Laplacian is defined by $\nabla \cdot \nabla$. Hence, the resulting scheme is given by:

$$\frac{U_{k+1} - U_k}{\Delta t} + C_1 \Delta^2 U_{k+1} + C_2 U_{k+1} = C_1 \Delta^2 U_k - \Delta \left( \nabla \cdot \left( \frac{\nabla U_k}{|\nabla U_k|} \right) \right) + C_2 U_k + \lambda(f - U_k).$$

Assuming zero Neumann boundary conditions on $\partial \Omega$, that is $\nabla U_{k+1} \cdot \nu = \nabla \Delta U_{k+1} \cdot \nu = 0$ on $\partial \Omega$, we can solve (12) using the discrete cosine transform. Now we state the main results concerning such scheme, summarizing them in the following theorem ([13, Theorem 4.1]):

**Theorem 3.1** (Consistency, Stability, Convergence). Let $u$ be the exact solution of (11) and $u_k = u(k\Delta t)$ the exact solution evaluated at time $k\Delta t$, $k \in \mathbb{N}$. Let $U_k$ the $k^{th}$ iterate of (12) with constant $C_1 > 1/\sqrt{\varepsilon}$ and $C_2 > \lambda_0$. Then, for any $T > 0$ fixed and every $k\Delta t \leq T$:
1. If \( \|u_{tt}\|_{H^{-1}} \) and \( \|\nabla \Delta u_{tt}\|_{L^2} \) are bounded, then the numerical scheme (12) is consistent with the continuous equation (11) with local truncation error \( \|\Delta t_k\|_{H^{-1}} = O(\Delta t) \).

2. For every \( \Delta t > 0 \):
\[
\|\nabla U_k\|_{L^2}^2 + \Delta t K_1 \|\nabla \Delta U_k\|_{L^2}^2 \leq e^{K_2 T}(\|\nabla U_0\|_{L^2}^2 + \Delta t K_1 \|\nabla \Delta U_0\|_{L^2}^2 + \Delta t TC')
\]
for suitable constants \( K_1, K_2 \) and a constant \( C = C(\Omega, D, \lambda_0, f) \). In other words, the sequence \( U_k \) is bounded on a finite interval \([0, T]\) for all \( \Delta t > 0 \). We say that the method is unconditionally stable.

3. Let \( e_k := u_k - U_k \). We have:
\[
\|\nabla e_k\|_{L^2}^2 + \Delta t M_1 \|\nabla \Delta e_k\|_{L^2}^2 \leq \frac{T}{M_2} e^{M_3 T}(\Delta t)^2
\]
for suitable constants \( M_1, M_2 \) and \( M_3 \). In other words, the error \( e_k \) converges to zero as \( \Delta t \to 0 \).

We will not present here the proof of this Theorem, as it constitutes the main part of Section 4 of [13].

### 3.1.2 Strang splitting for TV-\(H^{-1}\) inpainting

We present now an application of the Strang splitting method presented in 2.1 to solve in a different way our inpainting problem.

In order to solve the fourth-order equation (11), we consider the system of two second-order equations given by:
\[
\begin{align*}
    u_t &= \Delta v + \lambda(f - u), \\
    v &= -\nabla \cdot \left( \frac{\nabla u}{|\nabla u|_\varepsilon} \right).
\end{align*}
\]
Now we apply backward Euler temporal discretization an the Strang method, thus obtaining the following scheme:
\[
\begin{align*}
    U_{n+1/2} &= \frac{U_n + \frac{\Delta t}{2}(\Delta V_n + \lambda f)}{1 + \frac{\lambda}{2}}, \\
    V_{n+1/2} &= -\nabla \cdot \left( \frac{\nabla U_{n+1/2}}{|\nabla U_{n+1/2}|_\varepsilon} \right), \\
    U_{n+1} &= \frac{U_{n+1/2} + \frac{\Delta t}{2}(\Delta V_{n+1/2} + \lambda f)}{1 + \frac{\lambda}{2}}.
\end{align*}
\]

\(^2\)As we have seen, the TV-\(H^{-1}\) inpainting is not given by a single gradient flow, then the meaning of unconditional stability has to be refined in this way.
The spatial operators appearing in (13) are discretized in the same way as done for the convexity splitting. Results as Theorem 3.1 about convergence, stability and consistency do not exist, so they represent an interesting area of future research. We just point out that as the reader may note by himself, the scheme (13) is essentially an explicit method, then stability issues are expected.

3.1.3 Numerical results

Numerical results for both convexity approach (12) and Strang approach to the TV-$H^{-1}$ inpainting problem are presented in Figures 2-3. We can see that these two new methods can continue the gradient of the image function into the inpainting domain. A rigorous proof of this observation is a matter of future research. In both examples the total variation $TV(u)$ is approximated by $\sqrt{|\nabla u|^2 + \varepsilon}$ with $\varepsilon = 0.01$, the time step $\Delta t$ is chosen to be equal to one in the TV case and equal to 0.1 in the Strang splitting one and the parameter $\lambda_0$ is chosen to be equal to $10^2$. As pointed out in the previous subsection, the restriction on the tim-step size is due to the explicit character of Strang splitting.

Figure 2: The global result of TV-$H^{-1}$ inpainting solved with Strang and convexity splitting, respectively.
4 Two new methods based on an ADI approach

In the following we will present two different ADI schemes to solve the inpainting problem. We will present both of them in the total variation framework, using once again the total variation square root regularization, i.e. we will consider the PDE \((11)\) as governing the problem. Numerical solution of such an equation poses several problems:

- As pointed out in Section 1, the stiffness of fourth-order parabolic equations like \((11)\) lays down some constraints on the time-step size for explicit methods typically of the type \(\Delta t = O(\Delta x^4)\) which make such methods prohibitive. Hence, implicit methods are necessary;
- The presence of \(|\nabla u|_\varepsilon\) in \((11)\) makes the equation strongly nonlinear, hence convergence and accuracy are important considerations;
- The spatial operator necessarily includes mixed derivative terms for which splitting schemes appear to be more complicated.

Sensible methods for finding the numerical solution of such a problem were found by using a simpler equation, not directly related to problems arising in image processing, but with the common feature of being of higher order.

4.1 ADI-splitting schemes for the biharmonic equation

We exploit an auxiliary fourth order parabolic equation, called the \textit{biharmonic} equation. The behaviour of the solution of such an equation is very well-known and it is then easy comparing the numerical results with what we expect.
4.1.1 The scheme applied to the whole equation

We are dealing with the following partial differential equation, defined on \( \Omega \times \mathbb{R}^+ \):

\[
    u_t = -\Delta^2 u = -u_{xxxx} - u_{yyyy} - 2u_{xxyy} = F(u). \tag{14}
\]

We would like to semi-discretize the equation (14) and perform an ADI-splitting scheme similar to (10) for such an equation. We need then to decompose the function \( F \) into the sum of three parts \( F_0, F_1, F_2 \) such that \( F_0 \) could be handled explicitly, whereas \( F_1 \) and \( F_2 \) could be used implicitly, according to the rules of ADI scheme. We therefore decompose \( F \) in the following way:

\[
    F_0(u) = -2u_{xxyy} \\
    F_1(u) = -u_{xxxx} \\
    F_2(u) = -u_{yyyy} \\
    F(u) = F_0(u) + F_1(u) + F_2(u).
\]

With the choice above, we can write down a modification of the ADI splitting-scheme (10) introduced by Hunsdorfer and Verwer in [11] in order to increase the order of the scheme independently of \( F_0 \) (see Section 2.3). As usual, we use the notation \( U_n \) to indicate the approximation of the value \( u(n\Delta t) \), \( n \geq 1 \). For each time step \( \Delta t \) and real parameters \( \sigma, \theta > 0 \), the scheme is the following:

\[
\begin{aligned}
    Y_0 &= U_n + \Delta t F(U_n) \\
    Y_1 &= Y_0 + \theta \Delta t (F_1(Y_1) - F_1(U_n)) \\
    Y_2 &= Y_1 + \theta \Delta t (F_2(Y_2) - F_2(U_n)) \\
    \tilde{Y}_0 &= Y_0 + \sigma \Delta t (F(Y_2) - F(U_n)) \\
    \tilde{Y}_1 &= \tilde{Y}_0 + \theta \Delta t (F_1(\tilde{Y}_1) - F_1(Y_2)) \\
    \tilde{Y}_2 &= \tilde{Y}_1 + \theta \Delta t (F_2(\tilde{Y}_2) - F_2(Y_2)) \\
    U_{n+1} &= \tilde{Y}_2
\end{aligned}
\tag{15}
\]

The weights \( \theta \) and \( \sigma \) are the corrective parameters of the scheme usually taken to be \( \theta = \sigma = 1/2 \). With such a choice, in fact, it is possible to prove that the scheme (15) is of order 2 independently of the choice of \( F_0 \) which instead appears to be crucial with the original scheme without corrections (typically \( F_0 \) must be even taken equal to 0 to get a method of order 2!).

In the scheme above the spatial quantities are discretized using the finite difference schemes written below. For each pixel \( i, j \) we discretize as:
\[
\begin{align*}
\delta_{xxx}(u_{i,j}) &= \delta_{xx}(\delta_{xx}(u_{i,j})) = \frac{u_{i+2,j} - 4u_{i+1,j} + 6u_{i,j} - 4u_{i-1,j} + u_{i-2,j}}{h^4} \\
\delta_{yyy}(u_{i,j}) &= \delta_{yy}(\delta_{yy}(u_{i,j})) = \frac{u_{i,j+2} - 4u_{i,j+1} + 6u_{i,j} - 4u_{i,j-1} + u_{i,j-2}}{h^4}
\end{align*}
\]

where for the mixed derivative term we have used the 25-point stencil given by the approximation of the second order mixed derivatives with weight \( \beta \) used, for instance, in [9]. We observe that we are using the same stencil for the approximation of the second order mixed derivatives with weight \( \beta \).

Instead of applying the ADI scheme directly to the equation (14), it is also possible simplifying the problem by splitting the whole fourth order equation into a system of two different equations of order 2. The following ideas are similar in spirit to the ones presented in [6] where the authors apply the method to the imaging problem of denoising. Such ideas will help us in the following when we will describe our application to the inpainting problem.

### 4.1.2 The biharmonic equation reduced to two second order equations

Instead of applying the ADI scheme directly to the equation (14), it is also possible simplifying the problem by splitting the whole fourth order equation into a system of two different equations of order 2. The following ideas are similar in spirit to the ones presented in [6] where the authors apply the method to the imaging problem of denoising. Such ideas will help us in the following when we will describe our application to the inpainting problem. We consider then the system:

\[
\begin{align*}
\begin{cases}
  u_t &= \Delta v = v_{xx} + v_{yy} = F^1(u,v) \\
  v_t &= -\Delta u = -u_{xx} - u_{yy} = F^2(u,v)
\end{cases}
\end{align*}
\]  

(16)
where we have used the notations $F^1$ and $F^2$ to indicate the first and the second component of the column vector $F$, i.e.:

$$F(u,v) = \begin{pmatrix} F^1(u,v) \\ F^2(u,v) \end{pmatrix}.$$ 

In the following, we are going to use the same notation to indicate the components of the vectors $F_1$ and $F_2$ as well. We observe that with such a simplification no mixed derivatives appear in (16): the ADI method we are going to apply will use just pure derivatives schemes. When reducing the problem in this way, the main concern is not-changing the structure given by the numerical scheme (15), thus making sure that we are really solving the same problem, thus finding the same solutions. Practically, we observe that we don’t have a ”$F_0$ term” as we don’t have anything we prefer to deal with explicitly (as the matrix of the mixed derivatives we had before). So the difference between the explicit and the implicit steps is just the explicit application of the whole $F$ instead of the implicit one of the single components and the explicit steps are somehow a new ”initialization” of the numerical scheme which gives consistency to the following steps. The ADI scheme we are going to present has the feature of preserving the structure of the related scheme (15) presented before. We will comment more on this feature below, presenting our numerical results. Starting from the approximations $(U_n, V_n)$ we have then the following coupled ADI scheme depending again on the positive parameters $\theta, \sigma$:

$$\begin{cases} 
Y_0^2 &= \frac{F^2(U_n, V_n)}{U_n + \Delta t F^1(U_n, Y_0^1)} , \\
Y_0^1 &= \frac{Y_0^1}{V_n} + \left( \theta \Delta t (F^1_1(Y_1^1, Y_1^2) - F^1_1(U_n, V_n)) \right) , \\
Y_1^2 &= \frac{Y_1^2}{V_n} + \left( \theta \Delta t (F^2_1(Y_1^1, Y_1^2) - F^2_1(U_n, V_n)) \right) , \\
Y_2^1 &= \frac{Y_2^1}{V_n} + \left( \theta \Delta t (F^1_2(Y_2^1, Y_2^2) - F^1_2(U_n, V_n)) \right) , \\
Y_2^2 &= \frac{Y_2^2}{V_n} + \left( \theta \Delta t (F^2_2(Y_2^1, Y_2^2) - F^2_2(U_n, V_n)) \right) , \\
\hat{Y}_0^2 &= \frac{F^2(Y_2^1, Y_2^2)}{Y_0^1 + \sigma \Delta t (F^1(Y_2^1, \hat{Y}_0^2) - F^1(U_n, V_n))} , \\
\hat{Y}_0^1 &= \frac{\hat{Y}_0^1}{V_n} + \left( \theta \Delta t (F^1_2(\hat{Y}_1^1, \hat{Y}_1^2) - F^1_2(U_n, V_n)) \right) , \\
\hat{Y}_1^2 &= \frac{\hat{Y}_1^2}{V_n} + \left( \theta \Delta t (F^2_2(\hat{Y}_1^1, \hat{Y}_1^2) - F^2_2(U_n, V_n)) \right) , \\
\hat{Y}_2^1 &= \frac{\hat{Y}_2^1}{V_n} + \left( \theta \Delta t (F_2^1(\hat{Y}_2^1, \hat{Y}_2^2) - F_2^1(U_n, V_n)) \right) , \\
\hat{Y}_2^2 &= \frac{\hat{Y}_2^2}{V_n} + \left( \theta \Delta t (F_2^2(\hat{Y}_2^1, \hat{Y}_2^2) - F_2^2(U_n, V_n)) \right) , \\
(U_{n+1}^1, V_{n+1}^1) &= \hat{Y}_1^1 + \left( \theta \Delta t (F_2^1(\hat{Y}_1^1, \hat{Y}_2^2) - F_2^1(U_n, V_n)) \right) , \\
(U_{n+1}^2, V_{n+1}^2) &= \hat{Y}_2^2 + \left( \theta \Delta t (F_2^2(\hat{Y}_2^1, \hat{Y}_2^2) - F_2^2(U_n, V_n)) \right) .
\end{cases}$$

(17)
where the functions $F, F_1$ and $F_2$ are given by:

$$F_1(U, V) = \begin{pmatrix} A_1 & B_1 \\ C_1 & D_1 \end{pmatrix} \cdot \begin{pmatrix} U \\ V \end{pmatrix} = \begin{pmatrix} 0 & \delta_{xx} \\ -\delta_{xx} & 0 \end{pmatrix} \cdot \begin{pmatrix} U \\ V \end{pmatrix},$$

$$F_2(U, V) = \begin{pmatrix} A_2 & B_2 \\ C_2 & D_2 \end{pmatrix} \cdot \begin{pmatrix} U \\ V \end{pmatrix} = \begin{pmatrix} 0 & \delta_{yy} \\ -\delta_{yy} & 0 \end{pmatrix} \cdot \begin{pmatrix} U \\ V \end{pmatrix},$$

$$F(U, V) = F_1(U, V) + F_2(U, V)$$

(18)

where we discretize the spatial operators using the standard finite differences stencil for second derivatives.

We point out now some remarks explaining why and how we get such a scheme:

- As the reader may note, in both the explicit steps of the scheme above we swapped the order of application of the method for consistency issues. Namely, we first found consistent approximations for $V_{n+1}$ using them to get consistent approximations of $U_{n+1}$.

- As an example we point out how we got an expression for the approximation of $V$ in the implicit steps of the method. As the ADI scheme is usually performed for evolution equations (as the one for $u$) we have some sort of ”freedom” for the equation in $v$ even though we do not want to change the problem solved directly by (15) in the previous section. We consider as example the first implicit step giving the approximations $(Y_{11}^1, Y_{12}^1)$. For the approximation $Y_{11}^1$ of $U_{n+1}$ we have explicitly:

$$Y_{11}^1 = Y_{10}^1 + \theta \Delta t((F_1(Y_{11}^1, Y_{12}^2)-F_1(U_n, V_n)) = Y_{10}^1 + \theta \Delta t((Y_{21}^2)_{xx}-(V_n)_{xx}) = \cdots$$

Using now the expression of $Y_{12}^1$ given by the implicit step regarding the approximation for $v$ we have:

$$\cdots = Y_{10}^1 + \theta \Delta t((V_n - (Y_{11}^1)_{xx} + (U_n)_{xx} - (V_n)_{xx})$$

$$= Y_{10}^1 + \theta \Delta t((-Y_{12}^1)_{xxxx} + (U_n)_{xxxx})$$

which, compared to the relative step performed directly in (15) seems to give exactly the same result. We performed the same technique for the other implicit steps.

- The remarkable fact in the application of both these methods is that the numerical experiments (see the following section) show good and sensible results even for time-steps larger than $\Delta t \sim (\Delta x)^4$. We get
stable and sensible results both for \( \Delta t = O(\Delta x)^3 \) and also for \( \Delta t = O(\Delta x)^2 \). Considering \( \Delta t = O(\Delta x) \) we still get stable (bounded) results, but the numerical accuracy suffers and hence does not give sensible solutions.

• As we pointed out above, in this particular case we do not have any ”\( F_0 \) term” appearing in the coupled system for \( u \) and \( v \) (as we do not have any mixed derivatives), then, according to the classical theory regarding ADI methods, we could use a simpler ADI method, avoiding the use of the corrective parameter \( \sigma \) and the ”tilde” steps. Nevertheless, we decided to use the adaptive scheme anyway in order to have a direction to follow for the original problem of inpainting we are interested into where mixed derivatives appear even in the coupled system, as we are going to see in the following section.

4.1.3 Numerical results for the biharmonic equation

We present now some numerical results achieved with both the methods (15) and (17) applied to the biharmonic equation (14). In the implementation of both methods we had to face the problem of inverting the operators arising in the implicit steps of the ADI schemes. For the scheme (15) we simply used the standard LU factorization for these steps, but we could not do the same for the coupled system (16) because of the presence of bad conditioned matrices which prescribed long invertibility conditions. In order to avoid such a problem we use for the coupled ADI scheme the Schur complement method for the inversion of the matrices which seems to give good results as well as being computationally quick.

In the following numerical examples we considered the spatial domain \( \Omega \) being the unit square with 100 \( \times \) 100 gridpoints. We analyse the example of the evolution of a two-dimensional density having as initial condition \( U_0 \) the Gaussian density \( U_{ij}^0 = \exp(((x_i - 1/2)^2 + (y_j + 1/2)^2)/\gamma^2) \) where the variance \( \gamma^2 \) is equal to \( \gamma^2 = 100 \) (see Figure 4). In both cases we observe that, even increasing the time-step size up to \( \Delta t = O(\Delta x)^2 \), we get sensible and stable results preserving the self-similarity feature of the solutions (see Figures 5 and 6). Other numerical experiments showed that changing the initial condition \( U_0 \) and taking, for instance, some very oscillatory function does not effect the performance of the method. In the following numerical results the parameters are chosen as follows: \( \theta = \sigma = 1/2 \) and \( \beta = 0 \).
Motivated by these good results, we tried to write down an ADI method.
solving our TV-$H^{-1}$ inpainting equation (11). Unfortunately, our first attempt of expanding directly the differential operators appearing in the equation in order to split the mixed derivative terms from the pure ones, showed the impracticability of applying the ADI scheme to the whole equation. Such an expansion, in fact, generates a lot of nonlinear terms with different derivation orders. We performed then an ADI scheme similar to (17), reducing the original fourth order equation to a system of two second order equations.

4.2 ADI splitting method for TV-$H^{-1}$ inpainting equation

We propose now our ADI splitting method solving (11). As before, we rewrite the fourth-order equation (11) as a system of two second-order equations, then we linearize and we apply an ADI-splitting for equations with mixed derivatives following the ideas given by [9] and [10]. We start rewriting the equation as the following system of equations:

\[
\begin{cases}
    u_t = \Delta v + \lambda(f - u), \\
    v = -\nabla \cdot \left( \frac{\nabla u}{|\nabla u|_\varepsilon} \right) = -\frac{\Delta u}{|\nabla u|_\varepsilon} + \frac{u_x^2 u_{xx} + 2u_x u_y u_{xy} + u_y^2 u_{yy}}{|\nabla u|_\varepsilon^3}. 
\end{cases}
\]  

(19)

Denoting by $U$ and $V$ the semi-discretized approximations to a solution of (19) and with $\tilde{U}$ and $\tilde{V}$ the solution taken explicitly in some step of the ADI scheme, we have:

\[
\begin{cases}
    U_t = \delta \Delta V + \lambda(f - U) \\
    V = -\frac{\varepsilon + (\delta_y \tilde{U})^2}{|\nabla \tilde{U}|^3_\varepsilon} \delta_{xx} U - \frac{\varepsilon + (\delta_y \tilde{U})^2}{|\nabla \tilde{U}|^3_\varepsilon} \delta_{yy} U + 2\frac{\delta_x \tilde{U} \delta_y \tilde{U}}{|\nabla \tilde{U}|^3_\varepsilon} \delta_{xy} U 
\end{cases}
\]  

(20)

where we discretize in space all the first and second order pure derivatives of $u$ and $v$ using central and finite differences with periodic boundary conditions, whereas for the mixed derivatives we use the following scheme, averaging on a 9 points-grid and depending on a real parameter $\beta \in [-1, 1]$ (see [9]):

\[
\delta_{xy}(u_{i,j}) = \frac{(1 + \beta)(u_{i+1,j+1} + u_{i-1,j-1}) - (1 - \beta)(u_{i-1,j+1} + u_{i+1,j-1})}{4\Delta x \Delta y} + \frac{4\beta u_{i,j} - 2\beta(u_{i+1,j} + u_{i,j+1} + u_{i-1,j} + u_{i,j-1})}{4\Delta x \Delta y}.
\]
4.2.1 The splitting of the operator

We write now the system (20) in the following matricial form:

\[
\begin{pmatrix}
U_t \\
V
\end{pmatrix} = F(U, V) = \begin{pmatrix}
A & B \\
C & D
\end{pmatrix} \cdot \begin{pmatrix}
U \\
V
\end{pmatrix} + \begin{pmatrix}
S \\
T
\end{pmatrix}
\]

for suitable matrices \(A, B, C, D, S\) and \(T\) in \(\mathbb{R}^{NM \times NM}\). We split now \(F\) into the sum of three different terms: \(F_0\) containing the mixed derivative term and \(F_1\) and \(F_2\) which contain the derivatives with respect to \(x\) and to \(y\) only, respectively. This produces the splitting:

\[
F(U, V) = F_0(U, V) + F_1(U, V) + F_2(U, V) \quad (21)
\]

with:

\[
F_0(U, V) = \begin{pmatrix}
A_0 & B_0 \\
C_0 & D_0
\end{pmatrix} \cdot \begin{pmatrix}
U \\
V
\end{pmatrix} + \begin{pmatrix}
S \\
T
\end{pmatrix} = \begin{pmatrix}
-\Lambda/3 & 0 \\
2\delta_{xy}U\delta_{xy} & 0
\end{pmatrix} \cdot \begin{pmatrix}
U \\
V
\end{pmatrix} + \begin{pmatrix}
\Lambda_f \\
0
\end{pmatrix},
\]

\[
F_1(U, V) = \begin{pmatrix}
A_1 & B_1 \\
C_1 & D_1
\end{pmatrix} \cdot \begin{pmatrix}
U \\
V
\end{pmatrix} = \begin{pmatrix}
-\Lambda/3 & \delta_{xx} \\
\varepsilon + (\delta_{xy}U)^2 \delta_{xx} & 0
\end{pmatrix} \cdot \begin{pmatrix}
U \\
V
\end{pmatrix},
\]

\[
F_2(U, V) = \begin{pmatrix}
A_2 & B_2 \\
C_2 & D_2
\end{pmatrix} \cdot \begin{pmatrix}
U \\
V
\end{pmatrix} = \begin{pmatrix}
-\Lambda/3 & \delta_{yy} \\
\varepsilon + (\delta_{yy}U)^2 \delta_{yy} & 0
\end{pmatrix} \cdot \begin{pmatrix}
U \\
V
\end{pmatrix},
\]

where \(\Lambda\) and \(\Lambda_f\) are suitable matrices in \(\mathbb{R}^{NM \times NM}\) related to the fidelity term, then containing the information on the region to inpaint. The idea of splitting \(\Lambda\) in three parts, one explicit and two implicit, seems to get good numerical results. An alternative could be splitting it between \(A_1\) and \(A_2\). For stability reason, it does not seem good putting the whole \(\Lambda\) in the explicit matrix \(A_0\).

Finally, regarding the notations, for every \(F_j, j = 0, 1, 2\), we will indicate the rows of the respective coefficient matrix as follows:

\[
F_j(U, V) = \begin{pmatrix}
F_j^1(U, V) \\
F_j^2(U, V)
\end{pmatrix}
\]

and we do the same for every vector \(Y = \begin{pmatrix} Y^1 \\ Y^2 \end{pmatrix}\).
4.2.2 The ADI scheme

Given an initial condition \((U_0, V_0)\) our problem consists in finding an approximation \((U_{n+1}, V_{n+1})\) of the solution \((u(t_{n+1}), v(t_{n+1}))\) where \(t_n = n\Delta t, n \geq 1\) of (19). Now, setting \((\tilde{U}, \tilde{V}) = (U_n, V_n)\), we present an ADI scheme to compute these approximations:

\[
\begin{align*}
(Y_0^2) &= \left( \frac{F^2(U_n, V_n)}{U_n + \Delta t F^1(U_n, Y_0^2)} \right), \\
(Y_1^2) &= \left( \frac{F^2(Y_0^1, V_n)}{V_n} + \left( \theta \Delta t (F^1(Y_1^1, Y_1^2) - F^1(U_n, V_n)) \right) \right), \\
(Y_2^2) &= \left( \frac{F^2(Y_1^1, Y_2^2)}{F^1(Y_2^1, Y_2^2) - F^1(U_n, V_n)} \right),
\end{align*}
\]

(22)

We briefly comment this elaborated scheme noting that, as in (17), there are only two computations in the previous algorithm which are explicit timesteps: the first one and the fourth one where we take care of the mixed derivative terms of the problem. Once again in them the order of application of the method is reversed for consistency reasons, as pointed out previously for the biharmonic equation. The other computations are implicit timesteps and, recalling the splitting given by (21), they break down the original problem acting once again just along one direction. Namely, the second and the fifth computation of the scheme take care of the pure \(x\)-derivatives, the third and the sixth one take care of the pure \(y\)-derivatives. Moreover we remark that, for consistency reasons, in the explicit steps we could not use the approximations of \(U_{n+1}\) and \(V_{n+1}\) given by the explicit steps. Namely, in the second and the third steps, for the approximations of \(Y_1^2\) and \(Y_2^2\) we had to use just the value \(V_n\) (not the values \(Y_1^2\) and \(Y_2^2\), respectively) and similarly for \(Y_1^2\) and \(V_{n+1}\).

From a numerical point of view we just emphasize again that the inversion of the operators in the implicit steps was performed again by using the Schur complement technique.
4.2.3 The choice of the linearization

In this subsection we want to explain why the choice of the linearization from which our ADI scheme arises turns out to be important to get good results. Heuristically, such a choice is important from two different points of view, intrinsically related to each other. The former is the accuracy of the scheme we are considering: rough linearizations (i.e. linearizations which consider most of the nonlinear terms explicitly evaluating them in one or more given approximation) are likely to present poor accuracy as well as stability issues arising from the explicit evaluation of the nonlinear and stiff terms. This is general consideration in the numerical solution of each partial differential equation and, of course, it must be taken into account and balanced with the choice of linearizations which might be more accurate and precise, but which could present, on the other hand, difficulties in their implementation and application. The latter point of view is, in some sense, peculiar to our choice of performing a dimensional splitting scheme. As pointed out more than once above, in fact, our purpose is splitting our partial differential operator into the sum of components which are considered both explicitly (see $F_0$ above) and implicitly (see $F_1$ and $F_2$). The choice of the linearization affects the splitting from two points of view: the operators we want to put in the explicit component of $F$ depend, in fact, from the linearization itself as well as the quantities multiplying the differential operators considered implicitly, which play the role of coefficients (as they are given). Stability considerations are, even from this point of view, very important and are strictly related then to the choice of the linearization.

In the following, we will make more precise our considerations and choices, analyzing in detail the linearization performed in (20) and showing how things change when performing a different choice.

The first choice

We give now some motivations to the linearization performed in (20), pointing out some details. In the expression for $v$ found in (19), obtained by expanding fully the total variation term, the first choice we tried was a rather accurate linearization where we decided to consider explicitly the nonlinear term $|\nabla u|\varepsilon$ in the denominator and the derivatives of order one (and their powers) in the numerator, thus finding the approximation given in (20) which, after discretizing also in time, can be more precisely written as:
\[
\begin{aligned}
&\frac{U^{n+1} - U^n}{\Delta t} = \Delta V^{n+1} + \lambda (f - U^{n+1}), \\
&V^{n+1} = -\nabla \cdot \left( \frac{\nabla U^{n+1}}{\nabla U^n} \right) - \frac{1}{\nabla U^n} \delta_{xx} U^{n+1} + \frac{U^n U_x^n}{\nabla U^n} \delta_{xx} U^{n+1} + \frac{U^n U_y^n}{\nabla U^n} \delta_{yy} U^{n+1}
\end{aligned}
\]  
(23)

for every \( n \geq 0 \). The disadvantage of performing such an accurate choice, is the appearing of the mixed derivative term with all the related problems we have already discussed above. Nonetheless, it is hoped the accuracy of this linearization can have a better performance than more brutal ones.

The second choice

Another possibility is linearizing the system \([19]\) in an apparently more inaccurate way, though with some advantages from the point of view of our ADI numerical scheme. The alternative is the following:

\[
\begin{aligned}
&\frac{U^{n+1} - U^n}{\Delta t} = \Delta V^{n+1} + \lambda (f - U^{n+1}), \\
&V^{n+1} = -\nabla \cdot \left( \frac{\nabla U^{n+1}}{\nabla U^n} \right) - \frac{1}{\nabla U^n} \delta_{xx} U^{n+1} + \frac{U^n U_x^n}{\nabla U^n} \delta_{xx} U^{n+1} + \frac{U^n U_y^n}{\nabla U^n} \delta_{yy} U^{n+1}
\end{aligned}
\]  
(24)

We observe that with such a choice we do not have anymore the mixed derivative operator acting on \( U^{n+1} \) because the mixed terms are encoded and considered in the previous time step. On the other hand, we get first derivative operators and not just second order ones as in \([23]\). Namely, we are still considering the splitting the operator given by:

\[ F(U, V) = F_0(U, V) + F_1(U, V) + F_2(U, V) \]

but this time the choice is:

\[
\begin{aligned}
F_0(U, V) &= \left( \begin{array}{cc} -\Lambda/3 & 0 \\ 0 & 0 \end{array} \right) \cdot \left( \begin{array}{c} U \\ V \end{array} \right) + \left( \begin{array}{c} \Lambda_f \\ 0 \end{array} \right), \\
F_1(U, V) &= \left( \begin{array}{cc} -\Lambda/3 & -U^n U_x^n + U^n U_y^n \\ -U^n U_x^n + U^n U_y^n & \delta_{xx} \end{array} \right) \cdot \left( \begin{array}{c} U \\ V \end{array} \right), \\
F_2(U, V) &= \left( \begin{array}{cc} -\Lambda/3 & -U^n U_x^n + U^n U_y^n \\ -U^n U_x^n + U^n U_y^n & \delta_{yy} \end{array} \right) \cdot \left( \begin{array}{c} U \\ V \end{array} \right)
\end{aligned}
\]
where we have used the same notation as above. The resulting scheme has
the same structure as in \cite{22}, but with the different choice of the functions
pointed out above. The balance between the predicted improvements in the
stability behaviour giving by the absence of the explicit mixed derivative
operator and the issues related to too rough-linearization seems to indicate a
generical bad instability tendency with blowing up the numerical solutions.

The third choice

Another possible alternative to the linearization is exploiting the so-called
primal-dual formulation of the problem \cite{11} and adding a penalty or bareer
term as suggested in \cite{1} and \cite{12}...

4.2.4 Numerical results

We present now some numerical results obtained applying the scheme \cite{22}
to our inpainting equation \cite{11}. In order to study the properties of the
numerical scheme we first applied it to the equation taking $\lambda = 0$, i.e. we just
studied the evolution of the nonlinear process without the fidelity inpainting
term for a given initial datum. What we expect for such an equation is a
better edge-preserving behaviour than the one showed by the biharmonic
equation as the term $\nabla \cdot (\frac{\nabla u}{|\nabla u|})$ is the subgradient of to the total variation
and as such reduces diffusion in areas of large image gradient (i.e. of edges).
The approximation parameter $\varepsilon$ becomes then a "measure" of how close the
nonlinear process is to the biharmonic one. Big values of $\varepsilon$ should in fact show
behaviours similar to the smoothing effect of the biharmonic equation (as
$\varepsilon$ increases the denominator $|\nabla u|_\varepsilon$ becomes infinitely large), whereas small
values of $\varepsilon$ should show edge-preserving features, typical of total-variation
methods. The results we got seem instead to show a better (and stable)
behaviour for big values of $\varepsilon$, according also to the size of the time-step size
considered. We tested also the stability of the scheme and the results given
for different $\varepsilon$ when considering different, typically more oscillatory, initial
conditions, thus finding that the stability properties of such a scheme seem
to depend also on the variations of the gradient of the initial condition: very
oscillatory initial conditions with really steep gradients will present harder
stability constraints on the time step and require then bigger values of $\varepsilon$.
The analytical and precise reasons for that has still to be explored and it is
a matter of future research.

In the following we present first the results we get by the application of
the scheme \cite{22} to the nonlinear equation \cite{11} with the choice $\lambda = 0$. The
different choices of the regularizing parameter $\varepsilon$ are written gradually. We
consider at first the same initial condition as before, changing it with the
more oscillatory function $u_0(x, y) = \sin(15x) + \cos(15y)$. We include here the results obtained taking time-step sizes equal to $\Delta t = O(\Delta x)^3$ and $\Delta t = O(\Delta x)^2$ as they appear more interesting from a computational point of view because, as we have said before, they overcome the prohibitive constraint $\Delta t = O(\Delta x)^4$ to solve our original inpainting problem.

![Initial conditions](image1)

(a) Gaussian initial condition.  
(b) Oscillatory initial condition.

Figure 7: Initial conditions

![Initial conditions](image2)

(a) Gaussian datum.  
(b) Oscillatory datum.

Figure 8: Result $\varepsilon = 100$ and $\Delta t = O(\Delta x)^3$ for the two different initial conditions after 200 iterations.
Figure 9: Result with $\varepsilon = 1000$ and $\Delta t = O(\Delta x)^3$ for the two different initial conditions after 200 iterations.

Figure 10: Unstable results with $\varepsilon = 1$ and $\Delta t = O(\Delta x)^3$ for the two different initial conditions after few iterations.

Figure 11: Results with $\varepsilon = 100$ and $\Delta t = O(\Delta x)^2$ for the two different initial conditions after 100 and 3 iterations.
4.2.5 Numerical results applied to TV-$H^{-1}$ inpainting

We conclude this report presenting in the following figures some numerical results obtained applying the ADI method (22) presented above. For the sake of simplicity in the example below we have considered the square image ($N = M$) with dimension 100 × 100 pixels. Figures 14 and 15 show the best results we can get (i.e. for sensible values of $\varepsilon$) in a reasonable amount of time for different time-step sizes. Figure 16 shows instead the behaviour of the method for different values of the tuning parameter $\lambda$. The stability problems mentioned above still exist for the equation (11) and they do not seem to be overcome by changing the size of the fidelity parameter $\lambda$. 

Figure 12: Results with $\varepsilon = 1000$ and $\Delta t = O(\Delta x)^2$ for the two different initial conditions after 100 and 13 iterations.

Figure 13: Results with $\varepsilon = 10^4$ and $\Delta t = O(\Delta x)^2$ for the two different initial conditions after 100 iterations.
Figure 14: On the left: initial condition. On the right: result at time $t_n = 9 \times 10^{-5}$ for $\Delta t = O(\Delta x)^4$, $\lambda = 10^9$ and $\varepsilon = 0.1$.

Figure 15: On the left: result at time $t_n = 5.5 \times 10^{-5}$ for $\Delta t = O(\Delta x)^3$, $\lambda = 10^9$ and $\varepsilon = 10$. On the right: result at time $t_n = 0.0142$ for $\Delta t = O(\Delta x)^2$, $\lambda = 10^9$ and $\varepsilon = 100$.

Figure 16: On the left: result at time $t_n = 1 \times 10^{-5}$ for $\Delta t = O(\Delta x)^4$, $\lambda = 10^5$ and $\varepsilon = 0.01$. On the right: result at time $t_n = 1.22 \times 10^{-5}$ for $\Delta t = O(\Delta x)^3$, $\lambda = 10^5$ and $\varepsilon = 10$. We observe that we get poor and rather unstable results.
4.3 The ADI-Newton scheme

The second method we propose follows the ideas presented in [15] where, however, the authors deal with problems related to the study of lubrication flows of thin liquid films and are able to approximate the mean curvature (i.e. the divergence term in the inpainting equation) by the Laplace operator in the small gradient limit. We cannot perform such approximation anymore, since we are interested in looking at the variations of the image when edges appear, that is when the values of the gradient are significantly large. Accurate numerical solutions of nonlinear problems generally necessitate the use of iterative schemes like Newton’s method to calculate the approximated solution at the next time-step \( U_{n+1} \) given the solution at the previous time-steps \( U_n, U_{n-1}, \ldots \). Solving nonlinear problems like (11) using a backward Euler scheme involves actually a combination of iterative processes:

1. The application of Newton’s method at each time step to find the solution of the discretized nonlinear problems (i.e. for every \( n \) we want to find \( U_{n+1} \) such that for some \( F \) describing the problem we have \( F(U_{n+1}) = 0 \));

2. At each step of Newton’s method, some other iterative method must be used to solve large sparse linear algebra problem produced by the two-dimensional linearized operator, namely the Jacobian matrix which appears in Newton’s method (i.e. we iterate the process according to an index \( k \) in order to get approximations \( U^{(k)}_{n+1} \) of \( U_{n+1} \)).

We want to use some ADI scheme for the second process in order to reduce it to a direct solution of an approximately factored problem. Then, the iteration of 1. guarantees convergence to the solution\(^3\). We focus now on the use of Newton’s method. Consider solving (11) directly, using a backward Euler method. At each time-step, the discretized problem requires the solution of the system of nonlinear equations given by

\[
F(U_{n+1}) := U_{n+1} - U_n + \Delta t \Delta \nabla \cdot \left( \frac{\nabla U_{n+1}}{|\nabla U_{n+1}|} \right) - \Delta t \lambda (f - U_{n+1}) = 0
\]

where the performed space discretizations will be specified later on. The application of the Newton’s method to solve nonlinear system yields the following iterative method:

\[
J(U^{(k)}_{n+1})v = -F(U^{(k)}_{n+1}), \quad v := U^{(k+1)}_{n+1} - U^{(k)}_{n+1}, \quad k = 0, 1, 2, \ldots, \quad (25)
\]

\(^3\)Some authors refer to this kind of method calling them iterative factorized methods.
where, by iterating the process, we find some approximation to the solution at $t_{n+1}$. At each time step we initially estimate $U_{n+1}^{(0)}$ by the trivial explicit first-order approximation $U_{n+1}^{(0)} = U_n$ and then we keep iterating on $k$ thus finding better and better approximations of $U_{n+1}$. According to the theory regarding Newton’s methods we know that if $U_{n+1}^{(0)}$ is chosen to be sufficiently close to $U_{n+1}$, the sequence $\{U_{n+1}^{(k)}\}_k$ converges quadratically to $U_{n+1}$.

Using now the shorthand $\hat{U}$ to indicate the quantity $U_{n+1}^{(k)}$ (“explicit” as regarding the previous iteration on $k$), we write down the Jacobian (namely, the Fréchet derivative) appearing in (25) by:

\[
J(\hat{U})v \equiv \frac{\delta F}{\delta U} v = \frac{d}{d\epsilon} F(\hat{U} + \epsilon v)|_{\epsilon=0} = \Delta t \Delta [ -\hat{U} \cdot \nabla v \\
- \frac{1}{|\nabla \hat{U}|^2} (\nabla \hat{U} \cdot H_v \nabla \hat{U} + 2 \nabla \hat{U} \cdot H_{\hat{U}} \nabla v) + \frac{3}{|\nabla \hat{U}|^5} (\nabla \hat{U} \cdot H_{\hat{U}} \nabla \hat{U}) (\nabla \hat{U} \cdot \nabla v) \\
+ \frac{\Delta v}{|\nabla \hat{U}|} + (1 + \lambda) v
\]

(26)

where we have indicated by $H_U$ and $H_v$ the matrices:

\[
H_U = \begin{pmatrix} (\hat{U})_{xx} & (\hat{U})_{xy} \\ (\hat{U})_{xy} & (\hat{U})_{yy} \end{pmatrix} \quad H_v = \begin{pmatrix} v_{xx} & v_{xy} \\ v_{xy} & v_{yy} \end{pmatrix}
\]

Such approach can be computationally expensive and may become limiting in the speed and of numerical simulations for these problems. In order to develop a more efficient approach, we consider an approximate Newton’s method, where we approximate the Jacobian using the factorization $J \approx \hat{L}_x \hat{L}_y$ with the following choice of the operators:

\[
\hat{L}_x = I + \Delta t \hat{D}_x \quad \hat{L}_y = I + \Delta t \hat{D}_y.
\]

The differential operators $\hat{D}_x$ and $\hat{D}_y$ appearing in such a decomposition are obtained linearizing the quantity in the square brackets of (26) so that no mixed derivatives of $v$ could appear. The idea exploited to do that is to consider for each operator just the terms involving effectively the ’corresponding’ derivatives (for example, for $\hat{D}_x$ just the terms involving derivatives with respect to $x$) and to substitute the other quantities (for example, the derivatives with respect to $y$) with the corresponding explicit term related to $\hat{U}$. Acting in such a way, we add some further terms, but on the other hand we are able to exploit the advantage of considering the ’one-dimensional’ operators given
by:

\[
\begin{align*}
\tilde{D}_x(\phi) &= \partial_{xx} \left[ -\Delta \tilde{U} \phi_x + \tilde{U}_x^2 \right] - \frac{1}{|\nabla \tilde{U}|_e^3} (\tilde{U}_x^2 \phi_x + \phi_x (\tilde{U}_x \tilde{U}_x + \tilde{U}_{xx} \tilde{U}_y) + 3 \tilde{U}_x \tilde{U}_y \tilde{U}_{xy} + 2 \tilde{U}_y^2 \tilde{U}_{yy} + 3 \tilde{U}_x \tilde{U}_y \tilde{U}_{xy} + 2 \tilde{U}_y^2 \tilde{U}_{yy}) \\
+ & \frac{3}{|\nabla \tilde{U}|_e^3} (\nabla \tilde{U} \cdot \tilde{H} \nabla \tilde{U}) (\tilde{U}_x \phi_x + \tilde{U}_x^2) + \frac{\phi_{xx} + \tilde{U}_{yy}}{|\nabla \tilde{U}|_e}, \\
\tilde{D}_y(\phi) &= \partial_{yy} \left[ -\Delta \tilde{U} \phi_y + \tilde{U}_y^2 \right] - \frac{1}{|\nabla \tilde{U}|_e^3} (2 \tilde{U}_x \tilde{U}_{xx} + \phi_y (\tilde{U}_x \tilde{U}_{xy} + \tilde{U}_{xy} \tilde{U}_y) + \tilde{U}_{xx} + \phi_{yy}) \\
+ & \frac{3}{|\nabla \tilde{U}|_e^3} (\nabla \tilde{U} \cdot \tilde{H} \nabla \tilde{U}) (\tilde{U}_y^2 + \tilde{U}_y \phi_y) + \frac{\tilde{U}_{xx} + \phi_{yy}}{|\nabla \tilde{U}|_e} \\
- & \frac{1}{|\nabla \tilde{U}|_e^3} (\tilde{U}_x^2 \tilde{U}_{xx} + \tilde{U}_x \tilde{U}_{xy} \tilde{U}_y + \phi_y (\tilde{U}_{xy} \tilde{U}_x + \tilde{U}_{yy} \tilde{U}_y))
\end{align*}
\]

With this choice of operators, we have the following ADI scheme, used in the Newton method (25) to solve the method approximatively:

\[
\begin{align*}
\begin{cases}
\dot{L}_x w &= -F(U_{n+1}^{(k)}), \\
\dot{L}_y v &= w \\
U_{n+1}^{(k+1)} &= U_{n+1}^{(k)} + v.
\end{cases}
\end{align*}
\tag{27}
\]

In the implementation of such a method we had to write down all the previous (horrible) operators. In order to do that we used the following decomposition of the operators \(\tilde{D}_x\) and \(\tilde{D}_y\):

\[
\begin{align*}
\tilde{D}_x(\phi) &= D_x(\phi) + g_1(\tilde{U}) \quad \text{and} \quad \tilde{D}_y(\phi) = D_y(\phi) + g_2(\tilde{U}) \tag{28}
\end{align*}
\]

such that both the operators \(D_x\) and \(D_y\) act directly on \(\phi\) via some differential operators (namely, taking \(x\) and \(y\) derivatives of \(\phi\) multiplied for some coefficients depending just on the value \(\tilde{U} = U_{n+1}^{(k)}\) which is given in the previous iteration) while the functions \(g_1\) and \(g_2\) contain all the given quantities depending just on \(\tilde{U}\) and then they are not acting on the unknown \(\phi\). Using the same notation used in the code, the operators \(D_x\) and \(D_y\) and
the functions $g_1$ and $g_2$ turn out to be:

$$\mathbf{D}_x(\phi) = \partial_{xx} \left[ -\frac{\Delta \tilde{U}}{\left| \nabla \tilde{U} \right|^3} \tilde{U} x \phi_x \right] + \partial_{xx} \left[ -\frac{(\tilde{U}_x)^2}{\left| \nabla \tilde{U} \right|^3} \phi_{xx} \right] + \partial_{xx} \left[ -\frac{2(\tilde{U}_x \tilde{U}_{xx} + \tilde{U}_y \tilde{U}_{xy})}{\left| \nabla \tilde{U} \right|^3} \phi_x \right]$$

$$+ \partial_{xx} \left[ \frac{3(\nabla \tilde{U} \cdot H \nabla \tilde{U}) \tilde{U}_x}{\left| \nabla \tilde{U} \right|^5} \phi_x \right] + \partial_{xx} \left[ \frac{1}{\left| \nabla \tilde{U} \right|^3} \phi_{xx} \right]$$

$$= a_{1xx} \phi_x + 2a_{1x} \phi_{xx} + a_{1} \phi_{xxx} + a_{2xx} \phi_{xx} + 2a_{2x} \phi_{xxx} + a_{2} \phi_{xxxx} + a_{3xx} \phi_x + 2a_{3x} \phi_{xx}$$

$$+ a_3 \phi_{xxx} + a_4 \phi_{xx} + 2a_4 \phi_{xxx} + a_5 \phi_{xx} + a_5 \phi_{xxx} + a_5 \phi_{xxxx}$$

$$= (a_{1xx} + a_{3xx} + a_{4xx}) \phi_x + (2a_{1xx} + 2a_{2xx} + 2a_{3xx} + a_{5xx}) \phi_{xx}$$

$$+ (a_1 + 2a_2 + a_3 + a_4 + 2a_5) \phi_{xxx} + (a_2 + a_5) \phi_{xxxx}$$

$$\mathbf{D}_y(\phi) = \partial_{yy} \left[ -\frac{\Delta \tilde{U}}{\left| \nabla \tilde{U} \right|^3} \tilde{U} y \phi_y \right] + \partial_{yy} \left[ -\frac{(\tilde{U}_y)^2}{\left| \nabla \tilde{U} \right|^3} \phi_{yy} \right] + \partial_{yy} \left[ -\frac{2(\tilde{U}_y \tilde{U}_{yy} + \tilde{U}_x \tilde{U}_{xy})}{\left| \nabla \tilde{U} \right|^3} \phi_y \right]$$

$$+ \partial_{yy} \left[ \frac{3(\nabla \tilde{U} \cdot H \nabla \tilde{U}) \tilde{U}_y}{\left| \nabla \tilde{U} \right|^5} \phi_y \right] + \partial_{yy} \left[ \frac{1}{\left| \nabla \tilde{U} \right|^3} \phi_{yy} \right]$$

$$= b_{1yy} \phi_y + 2b_{1y} \phi_{yy} + b_{1} \phi_{yyy} + b_{2yy} \phi_{yy} + 2b_{2y} \phi_{yyy} + b_{2} \phi_{yyyy} + b_{3yy} \phi_y + 2b_{3y} \phi_{yy} + 2b_{3} \phi_{yyyy}$$

$$+ b_{3} \phi_{yyy} + b_{4yy} \phi_y + 2b_{4y} \phi_{yy} + b_{4} \phi_{yyyy} + b_{5yy} \phi_y + 2b_{5y} \phi_{yy} + b_{5} \phi_{yyyy}$$

$$= (b_{1yy} + b_{3yy} + b_{4yy}) \phi_y + (2b_{1y} + 2b_{2y} + 2b_{3y} + 2b_{4y} + 2b_{5y}) \phi_{yy}$$

$$+ (b_1 + 2b_2 + b_3 + b_4 + 2b_5) \phi_{yyy} + (b_2 + b_5) \phi_{yyyy}$$

$$g_1(\tilde{U}) = \partial_{xx} \left( -\frac{\Delta \tilde{U}}{\left| \nabla \tilde{U} \right|^3} (\tilde{U}_y)^2 \right) + \partial_{xx} \left( -\frac{1}{\left| \nabla \tilde{U} \right|^3} (3\tilde{U}_x \tilde{U}_y \tilde{U}_{xy} + 2(\tilde{U}_y)^2 \tilde{U}_{yy}) \right)$$

$$+ \partial_{xx} \left( \frac{3}{\left| \nabla \tilde{U} \right|^5} (\nabla \tilde{U} \cdot H \nabla \tilde{U})(\tilde{U}_y)^2 \right) + \partial_{xx} \left( \frac{\tilde{U}_{yy}}{\left| \nabla \tilde{U} \right|^3} \right)$$

$$+ \partial_{xx} \left( -\frac{1}{\left| \nabla \tilde{U} \right|^3} (\tilde{U}_x \tilde{U}_y \tilde{U}_{xy} + \tilde{U}_{yy} \tilde{U}_y)^2 \right) = \partial_{xx} t_1 + \partial_{xx} t_2 + \partial_{xx} t_3 + \partial_{xx} t_4 + \partial_{xx} t_5$$

$$g_2(\tilde{U}) = \partial_{yy} \left( -\frac{\Delta \tilde{U}}{\left| \nabla \tilde{U} \right|^3} (\tilde{U}_x)^2 \right) + \partial_{yy} \left( -\frac{1}{\left| \nabla \tilde{U} \right|^3} (3\tilde{U}_x \tilde{U}_y \tilde{U}_{xy} + 2(\tilde{U}_x)^2 \tilde{U}_{xx}) \right)$$

$$+ \partial_{yy} \left( \frac{3}{\left| \nabla \tilde{U} \right|^5} (\nabla \tilde{U} \cdot H \nabla \tilde{U})(\tilde{U}_x)^2 \right) + \partial_{yy} \left( \frac{\tilde{U}_{xx}}{\left| \nabla \tilde{U} \right|^3} \right)$$

$$+ \partial_{yy} \left( -\frac{1}{\left| \nabla \tilde{U} \right|^3} (\tilde{U}_x \tilde{U}_y \tilde{U}_{xy} + \tilde{U}_{xx} \tilde{U}_x)^2 \right) = \partial_{yy} s_1 + \partial_{yy} s_2 + \partial_{yy} s_3 + \partial_{yy} s_4 + \partial_{yy} s_5.$$
on $\phi$. The expressions above have been found by simply applying the Leibniz formula for products of functions. We can observe then that each differential operator $D_x$ and $D_y$ acts just on the corresponding derivatives of the function to which it is applied: no mixed derivatives appear and the nonlinearities appearing in (11) are overcome by considering all the terms deriving from them explicitly, by the linearization described above.

As far as the spatial discretizations are concerned, we point out that we used for each operator different ”stencils” of grid points on which we averaged to find the approximation of the derivatives. Namely we used 9-point grid stencil for the first and the second derivatives, and the 25-point one for the third and the fourth derivatives, then assembling everything together. Namely, we discretized the $x$-derivatives of $u_{i,j}$, where the couple $i,j$ represents a pixel in the image as:

\[
\delta^c_x(u_{i,j}) = \frac{u_{i+1,j} - u_{i-1,j}}{2\Delta x}
\]
\[
\delta_{xx}(u_{i,j}) = \frac{u_{i+1,j} - u_{i-1,j} - 2u_{i,j}}{\Delta x^2}
\]
\[
\delta_{xxx}(u_{i,j}) = \frac{u_{i+1,j} - 2u_{i+1,j} + 2u_{i-1,j} - u_{i-2,j}}{2\Delta x^3}
\]
\[
\delta_{xxxx}(u_{i,j}) = \frac{u_{i+2,j} - 4u_{i+1,j} + 6u_{i,j} - 4u_{i-1,j} + u_{i-2,j}}{\Delta x^4}
\]

and similarly for the $y$-derivatives.

By construction we see that this method turns out to be a semi-implicit method, i.e. some parts of the equation are taken explicitly (introduced by the Newton approximation and the dimensional splitting), some implicit in time. Hence, the stability of such a strategy becomes an important consideration and has still to be explored. However, we point out that differently from the second method we are about to present in the next section, in this ADI-Newton method there are no solutions of explicit in time-steps involved.

Even though from a theoretical point of view the ADI-Newton method presented above seems to be very promising, its implementation gave us some problems we were not able to fix. The main problem we experienced was the bad conditioning behaviour of the large and sparse matrices representing the operators $\tilde{L}_x$ and $\tilde{L}_y$ which caused issues in the inversion of such operators in the steps of (27). A first attempt could be using some preconditioning method to overcome this problem or finding some other approximation of the Jacobian $J$ different from the ”factorized” one presented above.
5 Appendices

We recall now some mathematical concepts which have been useful in our analysis. In all this section we will indicate with $\Omega$ an open and bounded domain in $\mathbb{R}^2$ with Lipschitz boundary $\Gamma := \partial \Omega$.

A The space $H^{-1}$

We define the non-standard Hilbert space $H^{-1}(\Omega)$ as:

$$H^{-1}(\Omega) := \{ F \in H^1(\Omega)^* : \langle F, 1 \rangle_{H^1(\Omega)} = 0 \}.$$

The norm of a function $F \in H^{-1}(\Omega)$ is defined as

$$\| F\|_{-1} := \| \nabla \Delta^{-1} F \|_{L^2(\Omega)}^2 = \int_{\Omega} (\nabla \Delta^{-1} F(x))^2 dx$$

where the operator $\Delta^{-1}$ denotes the inverse of the Laplacian with Neumann boundary conditions, i.e. $u := \Delta^{-1} F$ is the unique weak solution in the space $H^1_N(\Omega) := \{ \psi \in H^1(\Omega) : \int_{\Omega} \psi(x) dx = 0 \}$ of the problem:

$$\begin{cases}
\Delta u = F & \text{in } \Omega \\
\nabla u \cdot \nu = 0 & \text{on } \Gamma
\end{cases}$$

where $\nu$ denote the outward normal vector on $\Gamma$. For further characterizations of this space we refer to [4, Appendix A].

B Functions of bounded variation

We define the space of functions of bounded variation $BV(\Omega)$ in bidimensional domain as follows:

\textbf{Definition B.1} ($BV(\Omega)$. Let $u \in L^1(\Omega)$. We say that $u$ is a \textbf{function of bounded variation in} $\Omega$ if the distributional derivative of $u$ is representable by a finite Radon measure in $\Omega$, i.e. if:

$$\int_{\Omega} u(x) \frac{\partial \phi}{\partial x_i} dx = -\int_{\Omega} \phi(x) dD_i u(x) = \langle u', \phi \rangle_{L^2(\Omega)} \quad \forall \phi \in C^\infty_c(\Omega), \quad i = 1, 2$$

for some $\mathbb{R}^2$-valued measure $D u = (D_1 u, D_2 u)$ in $\Omega$. We denote the vector space of all such functions by $BV(\Omega)$.}
It is possible to characterize the space $BV(\Omega)$ by the total variation of $Du$. Before doing it we define the variation of a function in $L^1_{\text{loc}}(\Omega)$ and the total variation of a measure.

**Definition B.2 (Variation).** Let $u \in L^1_{\text{loc}}(\Omega)$. The variation $V(u, \Omega)$ of $u$ in $\Omega$ is defined by:

$$
\int_{\Omega} |Du| := \sup \left\{ \int_{\Omega} u(x) \nabla \cdot \phi(x) dx : \phi \in (C^1_c(\Omega))^2, \|\phi\|_{\infty} \leq 1 \right\}.
$$

We point out that just integrating by parts we see that $\int_{\Omega} |Du| = \int_{\Omega} |\nabla u(x)| dx$ if $u \in C^1(\Omega)$ and, using density argument, the same holds for functions in $W^{1,1}(\Omega)$.

**Definition B.3 (Total variation).** Let $(X, \mathcal{E})$ be a measure space. If $\mu$ is a measure, we define for every $E \in \mathcal{E}$ its total variation $|\mu|$ as follows:

$$
|\mu|(E) := \sup \left\{ \sum_{n=0}^{\infty} |\mu(E_n)| : E_n \in \mathcal{E}, E_i \cap E_j = \emptyset \forall i \neq j, E = \bigcup_{n=0}^{\infty} E_n \right\}.
$$

With the previous definitions we can characterize the space $BV(\Omega)$ with the following theorem.

**Theorem B.4 (Characterization of $BV(\Omega)$).** Let $u \in L^1(\Omega)$. Then, $u \in BV(\Omega)$ if and only if $\int_{\Omega} |Du| < \infty$. Moreover, $\int_{\Omega} |Du|$ coincides with $|Du|(\Omega)$ for any $u \in BV(\Omega)$ and the map:

$$
\mathcal{T} : u \mapsto |Du|(\Omega)
$$

is lower semicontinuous in $BV(\Omega)$ with respect to the $L^1_{\text{loc}}(\Omega)$ topology.

We also recall that $BV(\Omega)$ is a Banach space endowed with the norm:

$$
\|u\|_{BV(\Omega)} := \|u\|_{L^1(\Omega)} + \int_{\Omega} |Du|.
$$

Moreover, we point out that $BV(\Omega)$ is continuously embedded in $L^2(\Omega)$ if, as in our case, $\Omega \subset \mathbb{R}^2$.

## C Fréchet derivatives and subdifferentials

**Definition C.1 (Fréchet differentiability).** Let $H$ be a Hilbert space with norm $\|\cdot\|$ and inner product $(\cdot, \cdot)$ and let $E : H \to \mathbb{R}$. Given $u \in H$ we say that $E$ is Fréchet differentiable at $u$ if there exists $\xi \in H$ such that:

$$
\lim_{\|v\| \to 0} \frac{E(u + v) - E(u) - \langle \xi, v \rangle}{\|v\|} = 0.
$$
The element $\xi$ is necessarily unique and it is called the Fréchet derivative of $E$ at $u$ or the first variation of $E$ at $u$. We will indicate such $\xi$ with $\delta E$.

When the limit does not exist we can define a suitable generalization introducing the notion of subdifferential of a convex function.

**Definition C.2** (Subdifferential). Let $X$ be a locally convex space, let $X^*$ be its dual and $\langle \cdot, \cdot \rangle$ the duality pairing $X^* \times X$. Given a convex map $F$ of $X$ into $\mathbb{R}$, we define the subdifferential of $F$ at $u \in X$ as:

$$\partial F(u) := \{ \xi \in X^* : \langle \xi, v - u \rangle \leq F(v) - F(u) \ \forall v \in X \}.$$

Since every normed vector space is locally convex, we can therefore apply the theory of subdifferentials in the framework of the functions of bounded variation introduced above. For further details see [8].

**Acknowledgements**

I am deeply and sincerely grateful to Carola Schönlieb for all the discussions we had and for all the time I took her up to realize this project. Thanks to her, I was introduced to the exciting area of image processing which I really would like to deepen in the future.

**References**


