Abstract

Vivamus vehicula leo a justo. Quisque nec augue. Morbi mauris wisi, aliquet vitae, dignissim eget, sollicitudin molestie, ligula. In dictum enim sit amet risus. Curabitur vitae velit eu diam rhoncus hendrerit. Vivamus ut elit. Praesent mattis ipsum quis turpis. Curabitur rhoncus neque eu dui. Etiam vitae magna. Nam ullamcorper. Praesent interdum bibendum magna. Quisque auctor aliquam dolor. Morbi eu lorem et est porttitor fermentum. Nunc egestas arcu at tortor varius viverra. Fusce eu nulla ut nulla interdum consectetuer. Vestibulum gravida. Morbi mattis libero sed est.

Contents

Chapter 1

Introduction

This thesis

This note presents a conjecture stemming from our investigations in the generation of sigmoid tensor categories of Picard numbers of tori in Banach algebras. Nulla ac nisl. Nullam urna nulla, ullamcorper in, interdum sit amet, gravida ut, risus. Aenean ac enim. In luctus. Phasellus eu quam vitae turpis viverra pellentesque. Duis feugiat felis ut enim. Phasellus pharetra, sem id porttitor sodales, magna nunc aliquet nibh, nec blandit nisl mauris at pede. Suspendisse risus risus, lobortis eget, semper at, imperdiet sit amet, quam. Quisque scelerisque dapibus nibh. Nam enim. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Nunc ut metus. Ut metus justo, auctor at, ultrices eu, sagittis ut, purus. Aliquam aliquam.

1.1 Motivation for this work

In the works of Petri (**Petri**) we find the following statement

Theorem 1.1.1 (**Petri**, see also **BlackScholes**)**.** *The Gramm matrix for*

E_8 is:

$$\begin{pmatrix} 2 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 2 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 2 & -1 & 0 & 0 & 0 & -1 \\ 0 & 0 & -1 & 2 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 2 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 2 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 2 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 & 0 & 2 \end{pmatrix}.$$

1.1.1 The problem of exponential extensions

Donec et nisl id sapien blandit mattis. Aenean dictum odio sit amet risus. Morbi purus. Nulla a est sit amet purus venenatis iaculis. Vivamus viverra purus vel magna. Donec in justo sed odio malesuada dapibus. Nunc ultrices aliquam nunc. Vivamus facilisis pellentesque velit. Nulla nunc velit, vulputate dapibus, vulputate id, mattis ac, justo. Nam mattis elit dapibus purus. Quisque enim risus, congue non, elementum ut, mattis quis, sem. Quisque elit.

1.1.2 The approach of Junderstein

Donec et nisl id sapien blandit mattis. Aenean dictum odio sit amet risus. Morbi purus. Nulla a est sit amet purus venenatis iaculis. Vivamus viverra purus vel magna. Donec in justo sed odio malesuada dapibus. Nunc ultrices aliquam nunc. Vivamus facilisis pellentesque velit. Nulla nunc velit, vulputate dapibus, vulputate id, mattis ac, justo. Nam mattis elit dapibus purus. Quisque enim risus, congue non, elementum ut, mattis quis, sem. Quisque elit. Chambolle and Pock, 2011

Chapter 2

Eulerian topological string motives

Donec et nisl id sapien blandit mattis. Aenean dictum odio sit amet risus. Morbi purus. Nulla a est sit amet purus venenatis iaculis. Vivamus viverra purus vel magna. Donec in justo sed odio malesuada dapibus. Nunc ultrices aliquam nunc. Vivamus facilisis pellentesque velit. Nulla nunc velit, vulputate dapibus, vulputate id, mattis ac, justo. Nam mattis elit dapibus purus. Quisque enim risus, congue non, elementum ut, mattis quis, sem. Quisque elit.

2.1 Definitions

Donec et nisl id sapien blandit mattis. Aenean dictum odio sit amet risus. Morbi purus. Nulla a est sit amet purus venenatis iaculis. Vivamus viverra purus vel magna. Donec in justo sed odio malesuada dapibus. Nunc ultrices aliquam nunc. Vivamus facilisis pellentesque velit. Nulla nunc velit, vulputate dapibus, vulputate id, mattis ac, justo. Nam mattis elit dapibus purus. Quisque enim risus, congue non, elementum ut, mattis quis, sem. Quisque elit.

2.1.1 Tate's theorem

Preliminary considerations Donec et nisl id sapien blandit mattis. Aenean dictum odio sit amet risus. Morbi purus. Nulla a est sit amet purus venenatis iaculis. Vivamus viverra purus vel magna. Donec in justo sed odio malesuada dapibus. Nunc ultrices aliquam nunc. Vivamus facilisis pellentesque velit. Nulla nunc velit, vulputate dapibus, vulputate id, mattis ac, justo. Nam mattis elit dapibus purus. Quisque enim risus, congue non, elementum ut, mattis quis, sem. Quisque elit.

Motivic financial algebroids Donec et nisl id sapien blandit mattis. Aenean dictum odio sit amet risus. Morbi purus. Nulla a est sit amet purus venenatis iaculis. Vivamus viverra purus vel magna. Donec in justo sed odio malesuada dapibus. Nunc ultrices aliquam nunc. Vivamus facilisis pellentesque velit. Nulla nunc velit, vulputate dapibus, vulputate id, mattis ac, justo. Nam mattis elit dapibus purus. Quisque enim risus, congue non, elementum ut, mattis quis, sem. Quisque elit.

2.1.2 Grothendieck topologies

Donec et nisl id sapien blandit mattis. Aenean dictum odio sit amet risus. Morbi purus. Nulla a est sit amet purus venenatis iaculis. Vivamus viverra purus vel magna. Donec in justo sed odio malesuada dapibus. Nunc ultrices aliquam nunc. Vivamus facilisis pellentesque velit. Nulla nunc velit, vulputate dapibus, vulputate id, mattis ac, justo. Nam mattis elit dapibus purus. Quisque enim risus, congue non, elementum ut, mattis quis, sem. Quisque elit.

2.2 Calculation of the invariant cycles

Donec et nisl id sapien blandit mattis. Aenean dictum odio sit amet risus. Morbi purus. Nulla a est sit amet purus venenatis iaculis. Vivamus viverra purus vel magna. Donec in justo sed odio malesuada dapibus. Nunc ultrices aliquam nunc. Vivamus facilisis pellentesque velit. Nulla nunc velit, vulputate dapibus, vulputate id, mattis ac, justo. Nam mattis elit dapibus purus. Quisque enim risus, congue non, elementum ut, mattis quis, sem. Quisque elit.

2.2.1 Fontaine's theorem

Donec et nisl id sapien blandit mattis. Aenean dictum odio sit amet risus. Morbi purus. Nulla a est sit amet purus venenatis iaculis. Vivamus viverra purus vel magna. Donec in justo sed odio malesuada dapibus. Nunc ultrices aliquam nunc. Vivamus facilisis pellentesque velit. Nulla nunc velit, vulputate dapibus, vulputate id, mattis ac, justo. Nam mattis elit dapibus purus. Quisque enim risus, congue non, elementum ut, mattis quis, sem. Quisque elit.

Chapter 3

Conclusions

Vivamus vehicula leo a justo. Quisque nec augue. Morbi mauris wisi, aliquet vitae, dignissim eget, sollicitudin molestie, ligula. In dictum enim sit amet risus. Curabitur vitae velit eu diam rhoncus hendrerit. Vivamus ut elit. Praesent mattis ipsum quis turpis. Curabitur rhoncus neque eu dui. Etiam vitae magna. Nam ullamcorper. Praesent interdum bibendum magna. Quisque auctor aliquam dolor. Morbi eu lorem et est porttitor fermentum. Nunc egestas arcu at tortor varius viverra. Fusce eu nulla ut nulla interdum consectetuer. Vestibulum gravida. Morbi mattis libero sed est.

3.0.1 Discrete TGV minimisation problem

Let

$$\Omega_h = \{(i,j) \mid i,j \in \mathbb{N},\ 1 \le i \le N_1,\ 1 \le j \le N_2\}. \tag{3.1}$$

be ...

An image is a function $\Omega_h \to \mathbb{R}$.

Let $u \in \mathbb{R}^{N_1 \times N_2}$ be an image, $f \in \mathbb{R}^{N_1 \times N_2}$ the observed image.

The spaces of scalar, vector, and symmetric matrix valued functions are defined as (page 13)

$$\begin{aligned} U &= \{u : \Omega_h \to \mathbb{R}\} \\ V &= \{u : \Omega_h \to \mathbb{R}^2\} \\ W &= \{u : \Omega_h \to \mathrm{Sym}^2(\mathbb{R}^2)\} \end{aligned} \tag{3.2}$$

where $\mathrm{Sym}^2(\mathbb{R}^2)$ is the space of symmetric 2×2 matrices.

We use the notations v and w where

$$\begin{aligned} \mathrm{v} &\in V \text{ has components } (v)^{(1)} \text{ and } (v)^{(2)} \\ \mathrm{w} &\in W \text{ has components } (w)^{(1,1)},\ (w)^{(1,2)},\ (w)^{(2,1)},\ (w)^{(2,2)} \end{aligned} \tag{3.3}$$

with $(w)^{(1,2)} = (w)^{(2,1)}$.

Use the scalar product notation

$$a, b : \Omega_h \to \mathbb{R} : \quad \langle a, b \rangle = \sum_{i=1}^{N_1} \sum_{j=1}^{N_2} a_{i,j} b_{i,j} \tag{3.4}$$

the scalar products in the spaces V and W are

$$\begin{aligned} v, p \in V : \langle v, p \rangle_V &= \left\langle (v)^{(1)}, (p)^{(1)} \right\rangle + \left\langle (v)^{(2)}, (p)^{(2)} \right\rangle \\ w, q \in W : \langle w, q \rangle_W &= \left\langle (w)^{(1,1)}, (q)^{(1,1)} \right\rangle + \left\langle (w)^{(2,2)}, (q)^{(2,2)} \right\rangle + 2 \left\langle (w)^{(1,2)}, (q)^{(1,2)} \right\rangle \end{aligned} \tag{3.5}$$

are the scalar products in U, V, W.

The x and y forward finite difference operators are (page 13)

$$(\partial_x^+ u)_{i,j} = \begin{cases} u_{i+1,j} - u_{i,j} & \text{for } 1 \leq i < N_1 \\ 0 & \text{for } i = N_1 \end{cases} \tag{3.6}$$

$$(\partial_y^+ u)_{i,j} = \begin{cases} u_{i,j+1} - u_{i,j} & \text{for } 1 \leq j < N_2 \\ 0 & \text{for } j = N_2 \end{cases} \tag{3.7}$$

and the backward finite difference operators are (page 13)

$$(\partial_x^- u)_{i,j} = \begin{cases} u_{1,j} & \text{if } i = 1 \\ u_{i,j} - u_{i-1,j} & \text{for } 1 < i < N_1 \\ -u_{N_1-1,j} & \text{for } i = N_1 \end{cases} \tag{3.8}$$

$$(\partial_y^- u)_{i,j} = \begin{cases} u_{i,1} & \text{if } j = 1 \\ u_{i,j} - u_{i,j-1} & \text{for } 1 < j < N_2 \\ -u_{i,N_2-1} & \text{for } j = N_2 \end{cases} \tag{3.9}$$

The gradient operator ∇_h and the symmetrised gradient operator $\mathcal{E}_h$ are defined as (pages 13, 14)

$$\nabla_h : U \to V, \quad \nabla_h u = \begin{pmatrix} \partial_x^+ u \\ \partial_y^+ u \end{pmatrix}$$
$$\mathcal{E}_h : V \to W, \quad \mathcal{E}_h(v) = \begin{pmatrix} \partial_x^-(v)^{(1)} & \frac{1}{2}\left(\partial_y^-(v)^{(1)} + \partial_x^-(v)^{(2)}\right) \\ \frac{1}{2}\left(\partial_y^-(v)^{(1)} + \partial_x^-(v)^{(2)}\right) & \partial_y^-(v)^{(2)} \end{pmatrix} \tag{3.10}$$

and the divergence operators div_h are

$$\begin{aligned} \mathrm{div}_h : V \to U, \quad & \mathrm{div}_h v = \partial_x^-(v)^{(1)} + \partial_y^-(v)^{(2)} \\ \mathrm{div}_h : W \to V, \quad & \mathrm{div}_h w = \begin{pmatrix} \partial_x^+(w)^{(11)} + \partial_y^+(w)^{(12)} \\ \partial_x^+(w)^{(12)} + \partial_y^+(w)^{(22)} \end{pmatrix} \end{aligned} \tag{3.11}$$

The discrete ∞-norms are

$$\begin{aligned} v \in V: \quad & \|v\|_\infty = \max_{(i,j)\in\Omega_h} \left(\left((v)_{i,j}^{(1)}\right)^2 + \left((v)_{i,j}^{(2)}\right)^2 \right)^{1/2} \\ w \in W: \quad & \|w\|_\infty = \max_{(i,j)\in\Omega_h} \left(\left((w)_{i,j}^{(1,1)}\right)^2 + \left((w)_{i,j}^{(2,2)}\right)^2 + 2\left((w)_{i,j}^{(1,2)}\right)^2 \right)^{1/2} \end{aligned} \tag{3.12}$$

Finally, the TGV minimisation problem is defined as (page ...)

$$\min_{u\in\mathbb{R}^{N_1\times N_2}} F(u) + \mathrm{TGV}_\alpha^2(u) \tag{3.13}$$

where the data fidelity/discrepancy function is (page ...)

$$F(u) = \frac{1}{2}||u - f||_2^2 = \frac{1}{2}\sum_{i=1}^{N_1}\sum_{j=1}^{N_2}(u_{i,j} - f_{i,j})^2 \tag{3.14}$$

and the regularisation term is (page 14)

$$\mathrm{TGV}_\alpha^2(u) = \max\left\{ \langle u, \mathrm{div}_h v\rangle_U \;\middle|\; \begin{array}{l} (v,w) \in V \times W,\ \mathrm{div}_h w = v, \\ \|w\|_\infty \le \alpha_0,\ \|v\|_\infty \le \alpha_1 \end{array} \right\} \tag{3.15}$$

In [14, 34] it is demonstrated that the TGV functional can be equivalently written as

$$\mathrm{TGV}_{\alpha}^{2}(u) = \min_{w \in BD(\Omega)} \alpha_1 |Du - w|(\Omega) + \alpha_0 |\mathcal{E}w|(\Omega), \tag{3.16}$$

3.0.2 Numerical algorithm for TGV minimisation problem

Algorithm 1 PDHG algorithm for image denoising with fixed regularisation parameter-map $\mathbf{\Lambda}$ (adapted from Kofler et al., 2023 using the implementation by Shote, 2024)

Input: $L = \|[\mathbf{I}, \nabla]^{\mathrm{T}}\|_2$, $\tau = \mathrm{sigmoid}(10)/L$, $\sigma = \mathrm{sigmoid}(10)/L$, $\theta = \mathrm{sigmoid}(10)$, noisy image $\mathbf{x}_0$
Output: reconstructed image $\hat{\mathbf{x}}$
$\bar{\mathbf{x}}_0 = \mathbf{x}_0$
$\mathbf{p}_0 = \mathbf{x}_0$
$\mathbf{q}_0 = \mathbf{0}$
for $k < T$ **do**
    $\mathbf{p}_{k+1} = (\mathbf{p}_k + \sigma(\bar{\mathbf{x}}_k - \mathbf{x}_0))\,/(1+\sigma)$
    $\mathbf{q}_{k+1} = \mathrm{clip}_{\mathbf{\Lambda}}\,(\mathbf{q}_k + \sigma \nabla \bar{\mathbf{x}}_k)$
    $\mathbf{x}_{k+1} = \mathbf{x}_k - \tau \mathbf{p}_{k+1} - \tau \nabla^{\mathrm{T}} \mathbf{q}_{k+1}$
    $\bar{\mathbf{x}}_{k+1} = \mathbf{x}_{k+1} + \theta(\mathbf{x}_{k+1} - \mathbf{x}_k)$
end for
$\hat{\mathbf{x}} = \mathbf{x}_T$

Algorithm 2 Solve $\min_{u\in U} F_h(u) + \mathrm{TGV}^2_\alpha(u)$

Input: $f \in U$ is the input (noisy) image, and $\alpha_0, \alpha_1 > 0$ are the regularisation parameters.
Choose $\sigma > 0$, $\tau > 0$ such that $\sigma\tau\frac{1}{2}(17 + \sqrt{33}) \leq 1$. Here we choose $\sigma = \tau = 0.29$
$u^{[0]} = f$
$p^{[0]} = \mathbf{0}_V$
$v^{[0]} = \mathbf{0}_V$
$w^{[0]} = \mathbf{0}_W$
$\bar{u}^{[0]} = u^{[0]}$
$\bar{p}^{[0]} = p^{[0]}$
for $n = 0, 1, 2, \ldots$ **do**
    $v^{[n+1]} = \mathcal{P}_{\alpha_1}\left(v^{[n]} + \sigma(\nabla_h \bar{u}^{[n]} - \bar{p}^{[n]})\right)$
    $w^{[n+1]} = \mathcal{P}_{\alpha_0}\left(w^{[n]} + \sigma\mathcal{E}_h(\bar{p}^{[n]})\right)$
    $u^{[n+1]} = \left(u^{[n]} + \tau(\mathrm{div}_h v^{[n+1]} + f)\right)/(1+\tau)$
    $p^{[n+1]} = p^{[n]} + \tau(v^{[n+1]} + \mathrm{div}_h w^{[n+1]})$
    $\bar{u}^{[n+1]} = 2u^{[n+1]} - u^{[n]}$
    $\bar{p}^{[n+1]} = 2p^{[n+1]} - p^{[n]}$
end for
Return $u^{[N]}$ for some large N.

where $\mathbf{0}_V \in V$ and $\mathbf{0}_W \in W$ are the zeros matrices in V and W respectively, and $\mathcal{P}_{\alpha_1}$ and $\mathcal{P}_{\alpha_0}$ are the projection operators defined as

$$\begin{aligned} \mathcal{P}_{\alpha_1}(v) &= \frac{v}{\max\left(\mathbf{1}, \frac{1}{\alpha_1}|v|\right)}, \quad v \in V \\ \mathcal{P}_{\alpha_0}(w) &= \frac{w}{\max\left(\mathbf{1}, \frac{1}{\alpha_0}|w|\right)}, \quad w \in W \end{aligned} \tag{3.17}$$

where $|v|$, $|w| \in U$ are matrices with

$$
\begin{aligned}
|v|_{i,j} &= \left(\left((v)_{i,j}^{(1)} \right)^2 + \left((v)_{i,j}^{(2)} \right)^2 \right)^{1/2} \\
|w|_{i,j} &= \left(\left((w)_{i,j}^{(11)} \right)^2 + \left((w)_{i,j}^{(22)} \right)^2 + 2 \left((w)_{i,j}^{(12)} \right)^2 \right)^{1/2}
\end{aligned}
\tag{3.18}
$$

Here **1** is a matrix of ones. Note that the division is element-wise. (see below)

Denote the matrix underneath as a. The matrix under the division is 2D whereas the matrix above the division is 3D and 4D. Therefore each element $v_{i,j}^{(k)}$ and $w_{i,j}^{(k,h)}$ of the matrices v and w is divided by the corresponding element $a_{i,j}$ of the matrix under the division.

This algorithm is used to solve the minimisation problem (3.43). It is taken from the paper **recovering˙piecewise˙smooth˙multichannel˙images**, page ....

3.0.3 Appendix: Calculate projection P_α

Given matrix $a \in V$, we need to find matrix $u \in V$ such that

$$
u = \arg \min_{||v||_\infty \leq 1} ||v - a||_V \tag{3.19}
$$

where

$$
||v||_V = \sum_{i,j} \left(\left(v_{i,j}^{(1)} \right)^2 + \left(v_{i,j}^{(2)} \right)^2 \right)^{1/2} \tag{3.20}
$$

is the $L_{1,2}$-norm. (Note that here $u \in V$ is not the same as $u \in U$ in the previous section.)

Constraints:

$$
||v||_\infty = \max_{(i,j) \in \Omega_h} \left(\left(v_{i,j}^{(1)} \right)^2 + \left(v_{i,j}^{(2)} \right)^2 \right)^{1/2} \leq 1 \qquad \forall i, j \tag{3.21}
$$

Let $t_{i,j} = \sqrt{(v_{i,j}^{(1)})^2 + (v_{i,j}^{(2)})^2}$. Then the constraints can be written as

$$t_{i,j} \leq 1 \qquad \forall i, j \tag{3.22}$$

We have n^2 constraints. Each constraint is applied to a specific element $v_{i,j}$ and is independent of the others. Therefore, the solution can be found by solving n^2 independent problems:

$$u_{i,j} = \arg\min_{t_{i,j} \leq 1} \|v_{i,j} - a_{i,j}\| \quad \forall i, j \tag{3.23}$$

Denote $v_{i,j} = x$, $a_{i,j} = y$, $u_{i,j} = z$:

$$z = \arg\min_{\|x\|_2 \leq 1} \|x - y\| \qquad x, y, z \in \mathbb{R}^2 \tag{3.24}$$

The equation (3.54) is a projection of y onto the unit circle in $\mathbb{R}^2$. Therefore, it has the solution

$$\begin{aligned} z &= \begin{cases} \frac{y}{\|y\|_2} & \text{if } \|y\|_2 > 1 \\ y & \text{if } \|y\|_2 < 1 \end{cases} \\ &= \frac{y}{\max(\mathbf{1}, \|y\|_2)} \end{aligned} \tag{3.25}$$

Applying to u and a, we have

$$u_{i,j} = \frac{a_{i,j}}{\max(1, \|a_{i,j}\|_2)} \tag{3.26}$$

or in matrix notation,

$$u = \frac{a}{\max(\mathbf{1}, \|a\|_\infty)} \tag{3.27}$$

where $\mathbf{1}$ is a matrix of ones.

TODO: $||a||_\infty$ is a scalar, but we need a matriasdasdx?

3.0.4 Parameter Maps

In the scalar version, we are treating every region equally.

In the parameter map version, we will treat different regions differently. We want to assign appropriate values of α_0 and α_1 to different regions.

The parameter maps are matrices $\mathbf{\Lambda}_0$ and $\mathbf{\Lambda}_1$ of the same size as the image.

The minimisation problem is now

$$\min_{u \in U} F_h(u) + \mathrm{TGV}^2_{\mathbf{\Lambda}}(u) \tag{3.28}$$

where $F_h(u)$ is the data fidelity term defined in (3.44) and

$$\mathrm{TGV}^2_{\mathbf{\Lambda}}(u) = \max \left\{ \langle u, \mathrm{div}_h v \rangle_U \;\middle|\; \begin{array}{l} (v, w) \in V \times W,\ \mathrm{div}_h w = v, \\ |w| \leq \mathbf{\Lambda}_0,\ |v| \leq \mathbf{\Lambda}_1 \end{array} \right\} \tag{3.29}$$

is the regularisation term. The comparison $\leq$ is element-wise, and $|v|$, $|w| \in U$ are defined in (3.48).

3.0.5 PDHG Algorithm for TGV with Parameter Maps

Algorithm 3 Solve $\min_{u\in U} F_h(u) + \mathrm{TGV}^2_\alpha(u)$

Input: $f \in U$ is the input (noisy) image, and $\alpha_0, \alpha_1 > 0$ are the regularisation parameters.
Choose $\sigma > 0$, $\tau > 0$ such that $\sigma\tau\frac{1}{2}(17 + \sqrt{33}) \leq 1$. Here we choose $\sigma = \tau = 0.29$
$u^{[0]} = f$
$p^{[0]} = \mathbf{0}_V$
$v^{[0]} = \mathbf{0}_V$
$w^{[0]} = \mathbf{0}_W$
$\bar{u}^{[0]} = u^{[0]}$
$\bar{p}^{[0]} = p^{[0]}$
for $n = 0, 1, 2, \ldots$ **do**
    $v^{[n+1]} = \mathcal{P}_{\mathbf{\Lambda}_1}\left(v^{[n]} + \sigma(\nabla_h \bar{u}^{[n]} - \bar{p}^{[n]})\right)$
    $w^{[n+1]} = \mathcal{P}_{\mathbf{\Lambda}_0}\left(w^{[n]} + \sigma\mathcal{E}_h(\bar{p}^{[n]})\right)$
    $u^{[n+1]} = \left(u^{[n]} + \tau(\mathrm{div}_h v^{[n+1]} + f)\right)/(1+\tau)$
    $p^{[n+1]} = p^{[n]} + \tau(v^{[n+1]} + \mathrm{div}_h w^{[n+1]})$
    $\bar{u}^{[n+1]} = 2u^{[n+1]} - u^{[n]}$
    $\bar{p}^{[n+1]} = 2p^{[n+1]} - p^{[n]}$
end for
Return $u^{[N]}$ for some large N.

where

$$\begin{aligned} \mathcal{P}_{\mathbf{\Lambda}_1}(v) &= \frac{v}{\max\left(\mathbf{1}, \frac{|v|}{\mathbf{\Lambda}_1}\right)}, \quad v \in V \\ \mathcal{P}_{\mathbf{\Lambda}_0}(w) &= \frac{w}{\max\left(\mathbf{1}, \frac{|w|}{\mathbf{\Lambda}_0}\right)}, \quad w \in W \end{aligned} \tag{3.30}$$

$\frac{|v|}{\mathbf{\Lambda}_1}$ and $\frac{|w|}{\mathbf{\Lambda}_0}$ are element-wise divisions between matrices of the same size.

3.0.6 Discrete TGV minimisation problem

Let

$$\Omega_h = \{(i,j) \mid i,j \in \mathbb{N},\ 1 \le i \le N_1,\ 1 \le j \le N_2\}. \tag{3.31}$$

be ...

An image is a function $\Omega_h \to \mathbb{R}$.

Let $u \in \mathbb{R}^{N_1 \times N_2}$ be an image, $f \in \mathbb{R}^{N_1 \times N_2}$ the observed image.

The spaces of scalar, vector, and symmetric matrix valued functions are defined as (page 13)

$$\begin{aligned} U &= \{u : \Omega_h \to \mathbb{R}\} \\ V &= \{u : \Omega_h \to \mathbb{R}^2\} \\ W &= \{u : \Omega_h \to \mathrm{Sym}^2(\mathbb{R}^2)\} \end{aligned} \tag{3.32}$$

where $\mathrm{Sym}^2(\mathbb{R}^2)$ is the space of symmetric 2×2 matrices.

We use the notations v and w where

$$\begin{aligned} \mathrm{v} &\in V \text{ has components } (v)^{(1)} \text{ and } (v)^{(2)} \\ \mathrm{w} &\in W \text{ has components } (w)^{(1,1)},\ (w)^{(1,2)},\ (w)^{(2,1)},\ (w)^{(2,2)} \end{aligned} \tag{3.33}$$

with $(w)^{(1,2)} = (w)^{(2,1)}$.

Use the scalar product notation

$$a, b : \Omega_h \to \mathbb{R} : \quad \langle a, b \rangle = \sum_{i=1}^{N_1} \sum_{j=1}^{N_2} a_{i,j} b_{i,j} \tag{3.34}$$

the scalar products in the spaces V and W are

$$\begin{aligned} v, p \in V : \langle v, p \rangle_V &= \left\langle (v)^{(1)}, (p)^{(1)} \right\rangle + \left\langle (v)^{(2)}, (p)^{(2)} \right\rangle \\ w, q \in W : \langle w, q \rangle_W &= \left\langle (w)^{(1,1)}, (q)^{(1,1)} \right\rangle + \left\langle (w)^{(2,2)}, (q)^{(2,2)} \right\rangle + 2 \left\langle (w)^{(1,2)}, (q)^{(1,2)} \right\rangle \end{aligned} \tag{3.35}$$

are the scalar products in U, V, W.

The x and y forward finite difference operators are (page 13)

$$(\partial_x^+ u)_{i,j} = \begin{cases} u_{i+1,j} - u_{i,j} & \text{for } 1 \leq i < N_1 \\ 0 & \text{for } i = N_1 \end{cases} \tag{3.36}$$

$$(\partial_y^+ u)_{i,j} = \begin{cases} u_{i,j+1} - u_{i,j} & \text{for } 1 \leq j < N_2 \\ 0 & \text{for } j = N_2 \end{cases} \tag{3.37}$$

and the backward finite difference operators are (page 13)

$$(\partial_x^- u)_{i,j} = \begin{cases} u_{1,j} & \text{if } i = 1 \\ u_{i,j} - u_{i-1,j} & \text{for } 1 < i < N_1 \\ -u_{N_1-1,j} & \text{for } i = N_1 \end{cases} \tag{3.38}$$

$$(\partial_y^- u)_{i,j} = \begin{cases} u_{i,1} & \text{if } j = 1 \\ u_{i,j} - u_{i,j-1} & \text{for } 1 < j < N_2 \\ -u_{i,N_2-1} & \text{for } j = N_2 \end{cases} \tag{3.39}$$

The gradient operator ∇_h and the symmetrised gradient operator $\mathcal{E}_h$ are defined as (pages 13, 14)

$$\nabla_h : U \to V, \quad \nabla_h u = \begin{pmatrix} \partial_x^+ u \\ \partial_y^+ u \end{pmatrix}$$
$$\mathcal{E}_h : V \to W, \quad \mathcal{E}_h(v) = \begin{pmatrix} \partial_x^-(v)^{(1)} & \frac{1}{2}\left(\partial_y^-(v)^{(1)} + \partial_x^-(v)^{(2)}\right) \\ \frac{1}{2}\left(\partial_y^-(v)^{(1)} + \partial_x^-(v)^{(2)}\right) & \partial_y^-(v)^{(2)} \end{pmatrix} \tag{3.40}$$

and the divergence operators div_h are

$$\begin{aligned} \mathrm{div}_h : V \to U, \quad & \mathrm{div}_h v = \partial_x^-(v)^{(1)} + \partial_y^-(v)^{(2)} \\ \mathrm{div}_h : W \to V, \quad & \mathrm{div}_h w = \begin{pmatrix} \partial_x^+(w)^{(11)} + \partial_y^+(w)^{(12)} \\ \partial_x^+(w)^{(12)} + \partial_y^+(w)^{(22)} \end{pmatrix} \end{aligned} \tag{3.41}$$

The discrete ∞-norms are

$$\begin{aligned} v \in V: \quad & \|v\|_\infty = \max_{(i,j)\in\Omega_h} \left(\left((v)_{i,j}^{(1)}\right)^2 + \left((v)_{i,j}^{(2)}\right)^2 \right)^{1/2} \\ w \in W: \quad & \|w\|_\infty = \max_{(i,j)\in\Omega_h} \left(\left((w)_{i,j}^{(1,1)}\right)^2 + \left((w)_{i,j}^{(2,2)}\right)^2 + 2\left((w)_{i,j}^{(1,2)}\right)^2 \right)^{1/2} \end{aligned} \tag{3.42}$$

Finally, the TGV minimisation problem is defined as (page ...)

$$\min_{u\in\mathbb{R}^{N_1\times N_2}} F(u) + \mathrm{TGV}_\alpha^2(u) \tag{3.43}$$

where the data fidelity/discrepancy function is (page ...)

$$F(u) = \frac{1}{2}||u - f||_2^2 = \frac{1}{2}\sum_{i=1}^{N_1}\sum_{j=1}^{N_2}(u_{i,j} - f_{i,j})^2 \tag{3.44}$$

and the regularisation term is (page 14)

$$\mathrm{TGV}_\alpha^2(u) = \max\left\{ \langle u, \mathrm{div}_h v\rangle_U \;\middle|\; \begin{array}{l} (v,w) \in V \times W,\ \mathrm{div}_h w = v, \\ \|w\|_\infty \le \alpha_0,\ \|v\|_\infty \le \alpha_1 \end{array} \right\} \tag{3.45}$$

In [14, 34] it is demonstrated that the TGV functional can be equivalently written as

$$\mathrm{TGV}_{\alpha}^{2}(u) = \min_{w \in BD(\Omega)} \alpha_1 |Du - w|(\Omega) + \alpha_0 |\mathcal{E}w|(\Omega), \tag{3.46}$$

3.0.7 Numerical algorithm for TGV minimisation problem

Algorithm 4 PDHG algorithm for image denoising with fixed regularisation parameter-map $\mathbf{\Lambda}$ (adapted from Kofler et al., 2023 using the implementation by Shote, 2024)

Input: $L = \|[\mathbf{I}, \nabla]^{\mathrm{T}}\|_2$, $\tau = \mathrm{sigmoid}(10)/L$, $\sigma = \mathrm{sigmoid}(10)/L$, $\theta = \mathrm{sigmoid}(10)$, noisy image $\mathbf{x}_0$
Output: reconstructed image $\hat{\mathbf{x}}$
$\bar{\mathbf{x}}_0 = \mathbf{x}_0$
$\mathbf{p}_0 = \mathbf{x}_0$
$\mathbf{q}_0 = \mathbf{0}$
for $k < T$ **do**
    $\mathbf{p}_{k+1} = (\mathbf{p}_k + \sigma(\bar{\mathbf{x}}_k - \mathbf{x}_0))\,/(1+\sigma)$
    $\mathbf{q}_{k+1} = \mathrm{clip}_{\mathbf{\Lambda}}\,(\mathbf{q}_k + \sigma \nabla \bar{\mathbf{x}}_k)$
    $\mathbf{x}_{k+1} = \mathbf{x}_k - \tau \mathbf{p}_{k+1} - \tau \nabla^{\mathrm{T}} \mathbf{q}_{k+1}$
    $\bar{\mathbf{x}}_{k+1} = \mathbf{x}_{k+1} + \theta(\mathbf{x}_{k+1} - \mathbf{x}_k)$
end for
$\hat{\mathbf{x}} = \mathbf{x}_T$

Algorithm 5 Solve $\min_{u\in U} F_h(u) + \mathrm{TGV}^2_\alpha(u)$

Input: $f \in U$ is the input (noisy) image, and $\alpha_0, \alpha_1 > 0$ are the regularisation parameters.
Choose $\sigma > 0$, $\tau > 0$ such that $\sigma\tau\frac{1}{2}(17 + \sqrt{33}) \leq 1$. Here we choose $\sigma = \tau = 0.29$
$u^{[0]} = f$
$p^{[0]} = \mathbf{0}_V$
$v^{[0]} = \mathbf{0}_V$
$w^{[0]} = \mathbf{0}_W$
$\bar{u}^{[0]} = u^{[0]}$
$\bar{p}^{[0]} = p^{[0]}$
for $n = 0, 1, 2, \ldots$ **do**
    $v^{[n+1]} = \mathcal{P}_{\alpha_1}\left(v^{[n]} + \sigma(\nabla_h \bar{u}^{[n]} - \bar{p}^{[n]})\right)$
    $w^{[n+1]} = \mathcal{P}_{\alpha_0}\left(w^{[n]} + \sigma\mathcal{E}_h(\bar{p}^{[n]})\right)$
    $u^{[n+1]} = \left(u^{[n]} + \tau(\mathrm{div}_h v^{[n+1]} + f)\right)/(1+\tau)$
    $p^{[n+1]} = p^{[n]} + \tau(v^{[n+1]} + \mathrm{div}_h w^{[n+1]})$
    $\bar{u}^{[n+1]} = 2u^{[n+1]} - u^{[n]}$
    $\bar{p}^{[n+1]} = 2p^{[n+1]} - p^{[n]}$
end for
Return $u^{[N]}$ for some large N.

where $\mathbf{0}_V \in V$ and $\mathbf{0}_W \in W$ are the zeros matrices in V and W respectively, and $\mathcal{P}_{\alpha_1}$ and $\mathcal{P}_{\alpha_0}$ are the projection operators defined as

$$\begin{aligned} \mathcal{P}_{\alpha_1}(v) &= \frac{v}{\max\left(\mathbf{1}, \frac{1}{\alpha_1}|v|\right)}, \quad v \in V \\ \mathcal{P}_{\alpha_0}(w) &= \frac{w}{\max\left(\mathbf{1}, \frac{1}{\alpha_0}|w|\right)}, \quad w \in W \end{aligned} \tag{3.47}$$

where $|v|$, $|w| \in U$ are matrices with

$$
\begin{aligned}
|v|_{i,j} &= \left(\left((v)_{i,j}^{(1)} \right)^2 + \left((v)_{i,j}^{(2)} \right)^2 \right)^{1/2} \\
|w|_{i,j} &= \left(\left((w)_{i,j}^{(11)} \right)^2 + \left((w)_{i,j}^{(22)} \right)^2 + 2 \left((w)_{i,j}^{(12)} \right)^2 \right)^{1/2}
\end{aligned}
\tag{3.48}
$$

Here **1** is a matrix of ones. Note that the division is element-wise. (see below)

Denote the matrix underneath as a. The matrix under the division is 2D whereas the matrix above the division is 3D and 4D. Therefore each element $v_{i,j}^{(k)}$ and $w_{i,j}^{(k,h)}$ of the matrices v and w is divided by the corresponding element $a_{i,j}$ of the matrix under the division.

This algorithm is used to solve the minimisation problem (3.43). It is taken from the paper **recovering˙piecewise˙smooth˙multichannel˙images**, page ....

3.0.8 Appendix: Calculate projection P_α

Given matrix $a \in V$, we need to find matrix $u \in V$ such that

$$
u = \arg \min_{||v||_\infty \leq 1} ||v - a||_V \tag{3.49}
$$

where

$$
||v||_V = \sum_{i,j} \left(\left(v_{i,j}^{(1)} \right)^2 + \left(v_{i,j}^{(2)} \right)^2 \right)^{1/2} \tag{3.50}
$$

is the $L_{1,2}$-norm. (Note that here $u \in V$ is not the same as $u \in U$ in the previous section.)

Constraints:

$$
||v||_\infty = \max_{(i,j) \in \Omega_h} \left(\left(v_{i,j}^{(1)} \right)^2 + \left(v_{i,j}^{(2)} \right)^2 \right)^{1/2} \leq 1 \qquad \forall i, j \tag{3.51}
$$

Let $t_{i,j} = \sqrt{(v_{i,j}^{(1)})^2 + (v_{i,j}^{(2)})^2}$. Then the constraints can be written as

$$t_{i,j} \leq 1 \qquad \forall i, j \tag{3.52}$$

We have n^2 constraints. Each constraint is applied to a specific element $v_{i,j}$ and is independent of the others. Therefore, the solution can be found by solving n^2 independent problems:

$$u_{i,j} = \arg\min_{t_{i,j} \leq 1} \|v_{i,j} - a_{i,j}\| \quad \forall i, j \tag{3.53}$$

Denote $v_{i,j} = x$, $a_{i,j} = y$, $u_{i,j} = z$:

$$z = \arg\min_{\|x\|_2 \leq 1} \|x - y\| \qquad x, y, z \in \mathbb{R}^2 \tag{3.54}$$

The equation (3.54) is a projection of y onto the unit circle in $\mathbb{R}^2$. Therefore, it has the solution

$$\begin{aligned} z &= \begin{cases} \frac{y}{\|y\|_2} & \text{if } \|y\|_2 > 1 \\ y & \text{if } \|y\|_2 < 1 \end{cases} \\ &= \frac{y}{\max(\mathbf{1}, \|y\|_2)} \end{aligned} \tag{3.55}$$

Applying to u and a, we have

$$u_{i,j} = \frac{a_{i,j}}{\max(1, \|a_{i,j}\|_2)} \tag{3.56}$$

or in matrix notation,

$$u = \frac{a}{\max(\mathbf{1}, \|a\|_\infty)} \tag{3.57}$$

where $\mathbf{1}$ is a matrix of ones.

TODO: $||a||_\infty$ is a scalar, but we need a matriasdasdx?

3.0.9 Parameter Maps

In the scalar version, we are treating every region equally.

In the parameter map version, we will treat different regions differently. We want to assign appropriate values of α_0 and α_1 to different regions.

The parameter maps are matrices $\mathbf{\Lambda}_0$ and $\mathbf{\Lambda}_1$ of the same size as the image.

The minimisation problem is now

$$\min_{u \in U} F_h(u) + \mathrm{TGV}^2_{\mathbf{\Lambda}}(u) \tag{3.58}$$

where $F_h(u)$ is the data fidelity term defined in (3.44) and

$$\mathrm{TGV}^2_{\mathbf{\Lambda}}(u) = \max \left\{ \langle u, \mathrm{div}_h v \rangle_U \;\middle|\; \begin{array}{l} (v, w) \in V \times W,\ \mathrm{div}_h w = v, \\ |w| \leq \mathbf{\Lambda}_0,\ |v| \leq \mathbf{\Lambda}_1 \end{array} \right\} \tag{3.59}$$

is the regularisation term. The comparison $\leq$ is element-wise, and $|v|$, $|w| \in U$ are defined in (3.48).

3.0.10 PDHG Algorithm for TGV with Parameter Maps

Algorithm 6 Solve $\min_{u\in U} F_h(u) + \mathrm{TGV}^2_\alpha(u)$

Input: $f \in U$ is the input (noisy) image, and $\alpha_0, \alpha_1 > 0$ are the regularisation parameters.
Choose $\sigma > 0$, $\tau > 0$ such that $\sigma\tau\frac{1}{2}(17 + \sqrt{33}) \leq 1$. Here we choose $\sigma = \tau = 0.29$
$u^{[0]} = f$
$p^{[0]} = \mathbf{0}_V$
$v^{[0]} = \mathbf{0}_V$
$w^{[0]} = \mathbf{0}_W$
$\bar{u}^{[0]} = u^{[0]}$
$\bar{p}^{[0]} = p^{[0]}$
for $n = 0, 1, 2, \ldots$ **do**
    $v^{[n+1]} = \mathcal{P}_{\mathbf{\Lambda}_1}\left(v^{[n]} + \sigma(\nabla_h \bar{u}^{[n]} - \bar{p}^{[n]})\right)$
    $w^{[n+1]} = \mathcal{P}_{\mathbf{\Lambda}_0}\left(w^{[n]} + \sigma\mathcal{E}_h(\bar{p}^{[n]})\right)$
    $u^{[n+1]} = \left(u^{[n]} + \tau(\mathrm{div}_h v^{[n+1]} + f)\right)/(1+\tau)$
    $p^{[n+1]} = p^{[n]} + \tau(v^{[n+1]} + \mathrm{div}_h w^{[n+1]})$
    $\bar{u}^{[n+1]} = 2u^{[n+1]} - u^{[n]}$
    $\bar{p}^{[n+1]} = 2p^{[n+1]} - p^{[n]}$
end for
Return $u^{[N]}$ for some large N.

where

$$\begin{aligned} \mathcal{P}_{\mathbf{\Lambda}_1}(v) &= \frac{v}{\max\left(\mathbf{1}, \frac{|v|}{\mathbf{\Lambda}_1}\right)}, \quad v \in V \\ \mathcal{P}_{\mathbf{\Lambda}_0}(w) &= \frac{w}{\max\left(\mathbf{1}, \frac{|w|}{\mathbf{\Lambda}_0}\right)}, \quad w \in W \end{aligned} \tag{3.60}$$

$\frac{|v|}{\mathbf{\Lambda}_1}$ and $\frac{|w|}{\mathbf{\Lambda}_0}$ are element-wise divisions between matrices of the same size.

3.1 Neural Network Architecture

3.1.1 Convolutional Neural Network (CNN)

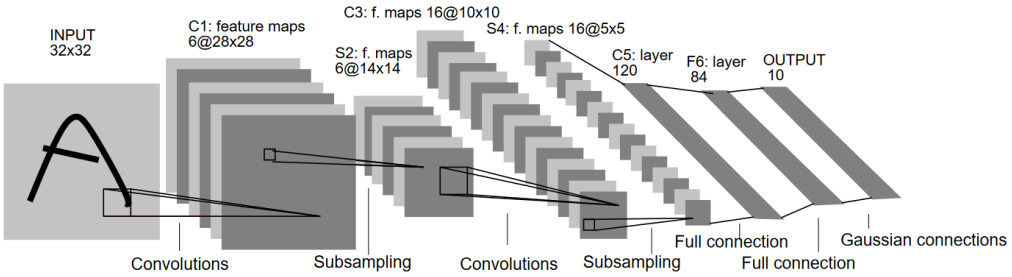

Figure 3.1: Architecture of LeNet-5, one of the earliest Convolutional Neural Networks (Lecun et al., 1998)

Convolutional Neural Networks (CNNs) are a specific type of artificial neural network particularly well-suited for analysing image data and other grid-like patterns. Here we assume a 2D CNN architecture, where each layer is represented as one or more 2-dimensional matrices, rather than a vector as in a normal "vanilla" network.

Inspiration and Function

CNNs are inspired by the visual cortex in the human brain. The visual cortex processes visual information, extracting features one-by-one, from simple edges and shapes to complex objects. CNNs try to mimic this through convolutional layers and pooling layers.

Convolutional Layers

Similar to a normal "vanilla" neural network layer, each element in a convolutional layer's output comes from a linear combination followed by an activation function. The difference is the use of filters (also called kernels)

instead of a single weight matrix. Each filter slides across the input, computing the dot product between its weights and the corresponding elements, then passes the output through an activation function. This is a convolution operation. For each input matrix, one filter produces one output matrix. We can have multiple filters in one layer. Each output matrix is called a feature map, since one filter can be thought of as a feature-extractor that is designed to highlight a specific feature. The number of feature maps is also called the number of channels. Implementation-wise, using filters instead of weight matrices reduces the ratio between the number of weights and the number of output values, hence cutting down on the number of trainable parameters for a image task.

Pooling Layers

An additional type of layer is a pooling layer. Usually we use a max-pooling layer which outputs the maximum instead. This project also utilises max-pooling. The goal is to keep only the most significant values.

These convolutional and pooling layers will form the building block for the U-Net architecture.

3.1.2 U-Net Architecture

The U-Net is a type of neural network that is commonly used for image segmentation (Ronneberger et al., 2015). The U-Net is an encoder-decoder network that is designed to take an image as input and produce a segmentation mask as output. The U-Net is made up of a series of convolutional layers that downsample the image and a series of transposed convolutional layers that upsample the image. The U-Net is able to learn to segment images by training on a large dataset of images and their corresponding segmentation masks.

In our case, we will optimise a U-Net model to find the regularisation

parameter map Λ that produces the best denoised image for any particular noisy image.

The U-Net architecture is divided into two principal components: an Encoder and a Decoder.

Encoder

The Encoder functions similarly to a standard CNN, utilising a combination of convolutional layers and max-pooling layers organised into distinct "blocks." In this project, each Encoder block comprises a pair of convolutional layers followed by a max-pooling layer, designed to successively double the number of channels (or feature maps) while reducing the feature map size due to the pooling.

Decoder

The Decoder is also a CNN with a series of blocks. Opposite to an encoding block which doubles the number of channels, each successive decoding block cuts the number of channels (feature maps) in half while increasing the size of the output feature map (hence the "unrolling"). Instead of a max-pooling layer, each decoding block ends with a so-called up-convolutional layer and a skip connection. The up-convolutional layer is just a normal convolutional layer whose output is concatenated with the output of another convolutional layer in an Encoder block.

3.1.3 Full Architecture

For this project, our model comprises two primary components:

1. A U-Net, denoted as NET_Θ, which is responsible for learning the regularisation parameters $\mathbf{\Lambda}_\Theta$ from an input image $\mathbf{x}_{\text{noisy}}$
2. A Primal Dual Hybrid Gradient (PDHG) algorithm solver

We refer to the set of trainable parameters in the U-Net as Θ, and the output of the U-Net as $\mathbf{\Lambda}_\Theta$.

The general architecture is depicted in shown in Figure 1,

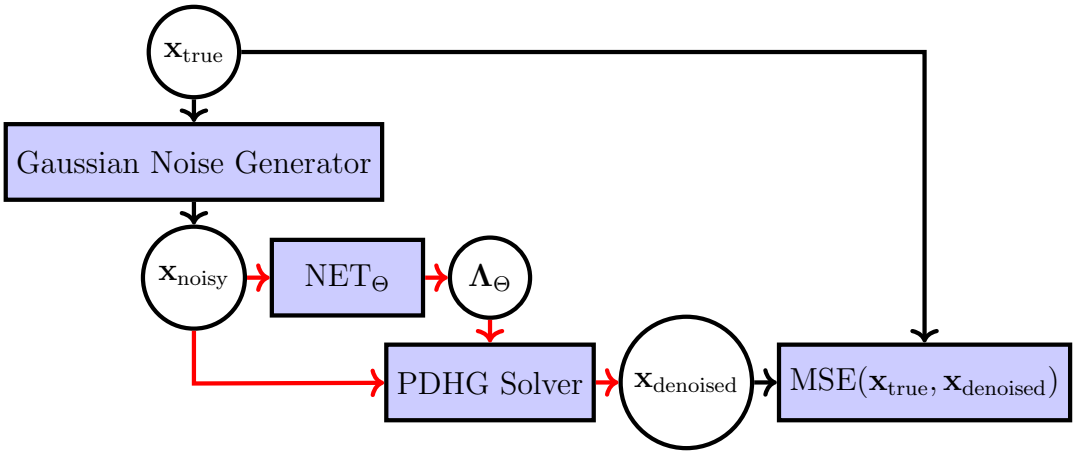

Figure 3.2: The general architecture

where

- $\mathbf{x}_{\text{true}}$ represents the clean image, serving as the ground truth.
- $\mathbf{x}_{\text{noisy}}$ denotes the noisy image inputted to NET_Θ.
- $\mathbf{\Lambda}_\Theta$ is the regularisation parameters map which is the final output from the U-Net.

We can treat the PDHG solver as the final hidden layer in our network. The result of the loss function $\text{MSE}(\mathbf{x}_{\text{true}}, \mathbf{x}_{\text{denoised}})$ is then used for backpropagation to train the parameters Θ.

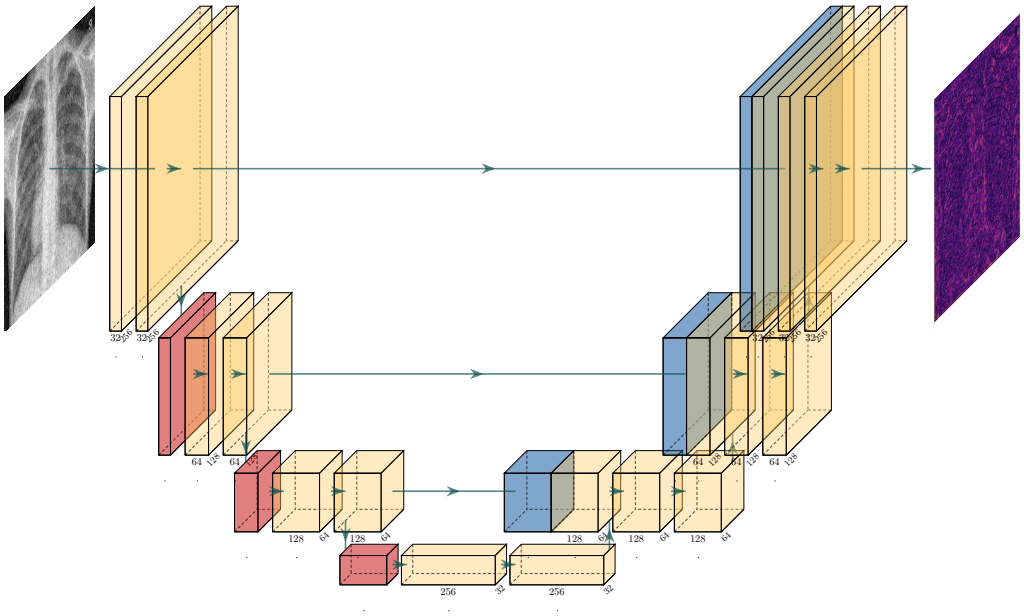

Figure 3.3: Our NET_Θ Architecture

3.1.4 Results

Appendix A

Implementation of the `BarrierOptionCVA` class

Vivamus vehicula leo a justo. Quisque nec augue. Morbi mauris wisi, aliquet vitae, dignissim eget, sollicitudin molestie, ligula. In dictum enim sit amet risus. Curabitur vitae velit eu diam rhoncus hendrerit. Vivamus ut elit. Praesent mattis ipsum quis turpis. Curabitur rhoncus neque eu dui. Etiam vitae magna. Nam ullamcorper. Praesent interdum bibendum magna. Quisque auctor aliquam dolor. Morbi eu lorem et est porttitor fermentum. Nunc egestas arcu at tortor varius viverra. Fusce eu nulla ut nulla interdum consectetuer. Vestibulum gravida. Morbi mattis libero sed est.

Appendix B

Additional details on the Gundermanian determinant

Vivamus vehicula leo a justo. Quisque nec augue. Morbi mauris wisi, aliquet vitae, dignissim eget, sollicitudin molestie, ligula. In dictum enim sit amet risus. Curabitur vitae velit eu diam rhoncus hendrerit. Vivamus ut elit. Praesent mattis ipsum quis turpis. Curabitur rhoncus neque eu dui. Etiam vitae magna. Nam ullamcorper. Praesent interdum bibendum magna. Quisque auctor aliquam dolor. Morbi eu lorem et est porttitor fermentum. Nunc egestas arcu at tortor varius viverra. Fusce eu nulla ut nulla interdum consectetuer. Vestibulum gravida. Morbi mattis libero sed est.

Bibliography

Chambolle, A., & Pock, T. (2011). A first-order primal-dual algorithm for convex problems with applications to imaging. *J. Math. Imaging Vis.*, *40*(1), 120–145. https://doi.org/10.1007/s10851-010-0251-1

Kofler, A., Altekrüger, F., Ba, F. A., Kolbitsch, C., Papoutsellis, E., Schote, D., Sirotenko, C., Zimmermann, F. F., & Papafitsoros, K. (2023). Learning regularization parameter-maps for variational image reconstruction using deep neural networks and algorithm unrolling. https://github.com/koflera/LearningRegularizationParameterMaps

Lecun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*(11), 2278–2324. https://doi.org/10.1109/5.726791

Ronneberger, O., Fischer, P., & Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation.

Shote, D. (2024). Pdhg algorithm implementation for dynamic image denoising. https://github.com/koflera/LearningRegularizationParameterMaps/blob/7537667e81adf3bdebbab8ebdc98fd631b586c03/networks/dyn_img_primal_dual_nn.py