Unfolded neural networks for image reconstruction

August 2024

Student: Arthur Barjot, first year of master degree at the ENS of Lyon
Supervision team: Audrey Repetti & Yves Wiaux, Heriot-Watt University
Duration: 3 months

Contents

1 Introduction

I did my three months internship at the Heriot-Watt University of Edinburgh. I enjoyed this internship that allowed me to discover new topics around neural networks, oriented toward mathematical aspect. I worked on image reconstruction using neural networks, and in particular on the robustness.

- First I discovered the topics of proximal method, dual-forward algorithm, firmly non-expansive and set-valued functions.
- Then I worked on a full proof of the theorem of convergence of the Condat-Vũ algorithm, you can find it in section A.3 of the appendix.
- After that I worked on unfolded neural networks 3.2, the heart of the internship was to compute a bound on the Lipschitz constant of this network, so I read about the classical way to find this Lipschitz constant. As my hypotheses of studies was more general than those of the known case, I had to make new calculation to develop new formulas, you can find them in section A.4 of the appendix.
- Finally I implemented some unrolled neural network and spent time on their tuning, in order to test if my Lipschitz constant was too loose , and I tried to understand why.

The report does not follow the same order, I had preferred to start with the experimental part to better introduce the topics of unrolled neural network and then I will give my results about the Lipschitz constant.

2 Mathematical foundation

If C is closed, non-empty and convex, we denote Π_C the projection onto C. And ι_C the indicator of C : $\iota_C(x) = 0$ if $x \in C$, $\iota_C(x) = +\infty$ otherwise. H will denote a real Hilbert space (in a lot of cases we just have $H = \mathbb{R}^N$). $\langle \mid \rangle$ is the scalar product of H and we denote $\|.\| = \|.\|_2$ the associate norm. We also denote $\|.\|$ the operator norm on $\mathcal{B}(H)$ associate to the norm $\|.\|_2$.

Let $f : H \to]-\infty, +\infty]$, we denote $\operatorname{dom}(f) := \{x \in H | f(x) < +\infty\}$, it is said to be proper if $\operatorname{dom}(f) \neq \emptyset$. f is lower-semi-continuous if its epigraph is closed. f is convex if its epigraph is a convex set. $\Gamma_0(H)$ is the space of functions $f : H \to]-\infty, +\infty]$ that are convex, proper and lower-semi-continuous. Let $f \in \Gamma_0(H)$, we define :

- its subdifferential : $\forall \bar{x} \in \operatorname{dom}(f), \partial f(\bar{x}) = \{p \in H | \forall x \in H, \langle x - \bar{x} \mid p\rangle + f(\bar{x}) \leqslant f(x)\}$,
- its proximal operator : $\forall \bar{x} \in H, \operatorname{prox}_f(\bar{x}) = \underset{x \in H}{\operatorname{argmin}} \ f(x) + \frac{1}{2}\|x - \bar{x}\|^2$,

- its conjugate : $\forall u \in H, f^*(u) = \sup_{x \in H} \langle x \mid u \rangle - f(x) \in [-\infty, +\infty]$.

Example 2.1 If $\lambda \in (0, +\infty)$, we have : $(\lambda \| \cdot \|_1)^* = \iota_{[-\lambda,\lambda]^S}$. If $C \subset \mathbb{R}^N$ is a non-empty, convex, closed set, then $\mathrm{prox}_{\tau \iota_C} = \Pi_C$.

Let $\phi : H \to H$, we denote $\mathrm{Fix}\,(\phi) := \{x \in H \mid \phi(x) = x\}$. We say that

(i) ϕ is non expansive if it is 1-Lipschitz.

(ii) ϕ is δ-averaged for $\delta \in (0,1)$ if $\phi = \delta g + (1-\delta)I$, where g is non expansive.

(iii) $\frac{1}{2}$-averaged functions are called firmly non expansive functions.

I proved the following result that enables to show asymptotic convergence of sequences based on averaging properties, and which seems to be new.

Proposition 2.2 *Let H be a finite dimensional Hilbert space, and $\phi : H \to H$ be a δ-averaged function for $\delta \in (0,1)$. Consider $(z_k)_{k \in \mathbb{N}} \in H^{\mathbb{N}}$ defined by :*

$$z_{k+1} = \phi(z_k)$$

If $\mathrm{Fix}\,(\phi) \neq \emptyset$*, then $(z_k)_k$ converge to a fixed point of ϕ.*

Proof. This result can be obtained combining techniques from [4] and the ideas on the Quasi-Fejér sequence of Type II in the section 3 of [3] (in particular Theorem 3.11). A full proof is available in Appendix A.2. □

Let H be a Hilbert space, a set-valued function M is defined by its graph, which can be every subset of $H \times H$. The domain of M is $\mathrm{dom}\,(M) = \{z \in H \mid \exists w \in H : (z,w) \in M\}$. Indeed it can be seen as a function which map one pre-image ($z \in \mathrm{dom}\,(M)$) to multiple images $w \in H$: $w \in M(z) \iff (z,w) \in \mathrm{graph}\,(M)$.

A set-valued function M is said to be monotone if :

$$\forall \big((z_1,w_1),(z_2,w_2)\big) \in (\mathrm{graph}\,M)^2, \quad \langle z_1 - z_2 \mid w_1 - w_2 \rangle \geqslant 0$$

A set-valued function M is said to be maximally monotone if for all monotone set-valued function M', $\mathrm{graph}\,(M) \subset \mathrm{graph}\,(M')$ implies $M = M'$. We also define the inverse of M using its graph : $(z,w) \in \mathrm{graph}\,M^{-1} \iff (w,z) \in \mathrm{graph}\,M$.

We recall two properties you can find, with their proof in [2].

Proposition 2.3 *[2]*

- *If $f \in \Gamma_0(H)$, then we have the formula :* $\mathrm{prox}_f + \mathrm{prox}_{f^*} = Id$
- *If $f \in \Gamma_0(H)$, ∂f is a maximally monotone set-valued operator, and we have : $(\partial f)^{-1} = \partial f^*$*

3 Problem description

In the context of inverse imaging problems, the objective is to find an estimate $\widehat{x} \in \mathbb{R}^N$ of an original image $\overline{x} \in \mathbb{R}^N$ from degraded observations $y \in \mathbb{R}^M$. The project will focus on linear measurements obtained as

$$y = H\overline{x} + w \tag{3.1}$$

where $w \in \mathbb{R}^M$ is a realization of an additive white Gaussian noise with standard deviation $s > 0$. $H \colon \mathbb{R}^N \to \mathbb{R}^M$ corresponds to a measurement operator, you can see some examples which will lead our study later :

(i) The problem of pure denoising correspond to $H = Id : \mathbb{R}^N \to \mathbb{R}^N$. (3.1) becomes $y = \bar{x} + w$, so we want to cancel the noise w, to obtain $\bar{x}$ from y.

(ii) The inpainting problem uses $H : \mathbb{R}^N \to \mathbb{R}^M$ an inpainting operator, which is just a linear projection onto a vector space of the form Span $(e_{i_1}, \ldots, e_{i_M})$, where $(e_i)_{i \in [\![1,N]\!]}$ is the canonical base of $\mathbb{R}^N$. Indeed it remains to cancel some pixels of our input image $\bar{x}$. The sub-sampling ratio will be the portion of preserved pixels $r = M/N$.

(iii) H can also be an undersampled Fourier operator, but this case is very similar to the previous one, indeed it remains to use an inpainting operator on the Fourier transform of our image. As it is very similar to the previous example, we would not use it.

(iv) The deblurring problem uses a convolution operator H, with a kernel $h \in \mathbb{R}^{m \times m}$:

$$H : \begin{pmatrix} \mathbb{R}^N \to \mathbb{R}^N \\ x \mapsto h * x \end{pmatrix}$$

The general problem 3.1 can be solved adopting a variational perspective, where the estimate $\widehat{x}$ corresponds to a minimiser of a regularised objective function:

$$\text{find } \widehat{x} \in \underset{x \in \mathbb{R}^N}{\operatorname{Argmin}} \ \frac{1}{2}\|Hx - y\|^2 + r(x). \tag{3.2}$$

In (3.2), $r \in \Gamma_0(\mathbb{R}^N)$ corresponds to a regularisation function, incorporating *a priori* information we have on the target image.

3.1 Proximal methods

Since early 2000's, iterative optimisation algorithms (in particular proximal methods [5]) have been state-of-the-art approaches for solving inverse problems. Most optimisation methods hold asymptotic convergence guarantees toward a minimiser of the objective function, making them highly reliable for decision-making processes.

In the following, we focus on the case where

$$(\forall x \in \mathbb{R}^N) \quad r(x) = \lambda \|Lx\|_1 + \iota_C(x), \tag{3.3}$$

where $\lambda > 0$ is a regularisation parameter balancing the least squares data-fidelity function and the ℓ_1 regularisation norm, $L\colon \mathbb{R}^N \to \mathbb{R}^S$ is a linear sparsifying dictionary (Total Variation operator, wavelet, etc), and $C \subset \mathbb{R}^N$ is a closed, convex, non-empty set.

Definition 3.1 The problem we want to solve is called the primal problem, it remains to find $\widehat{x}$ such that :

$$\widehat{x} \in \underset{x \in \mathbb{R}^N}{\operatorname{Argmin}} \ \frac{1}{2}\|Hx - y\|^2 + \iota_C(x) + \lambda \|Lx\|_1$$

Its dual problem remains to find $\widehat{u}$ such that :

$$\widehat{u} \in \underset{u \in \mathbb{R}^S}{\operatorname{Argmin}} \ \left(\frac{1}{2}\|H\cdot - y\|^2 + \iota_C\right)^* (L^* u) + (\lambda\|\cdot\|_1)^* (u)$$

The important hypothesis are that $\frac{1}{2}\|H\cdot - y\|^2$ is differentiable, $\iota_C \in \Gamma_0(\mathbb{R}^N)$ and $\lambda\|\cdot\|_1 \in \Gamma_0(\mathbb{R}^S)$.

An efficient algorithm to solve the resulting minimisation problem is the primal-dual Condat-Vũ algorithm [8]. We will not use relaxation parameter, and we can simplify the formulas in our case using the examples 2.1, hence the algorithm can be written :

$$\begin{array}{l} x_0 \in \mathbb{R}^N, u_0 \in \mathbb{R}^S, \tau > 0, \sigma > 0, \\ \text{for } k = 0, 1, \ldots \\ \left\lfloor \begin{array}{l} x_{k+1} := \Pi_C(x_k - \tau H^*(Hx_k - y) - \tau L^* u_k) \\ u_{k+1} := \Pi_{[-\lambda,\lambda]^S}(u_k + \sigma L(2x_{k+1} - x_k)) \end{array} \right. \end{array} \tag{3.4}$$

We denote by β the Lipschitz differentiable constant of the function $x \mapsto \frac{1}{2}\|Hx - y\|^2$, indeed $\beta = \|H^* H\|$.

Theorem 3.2 *Let $(x_k)_{k\in\mathbb{N}}$ and $(u_k)_{k\in\mathbb{N}}$ be generated by algorithm (3.4). Assume that $\tau > 0$, $\sigma > 0$, $\beta > 0$ are such that :*

$$\frac{1}{\tau} - \sigma\|L\|^2 > \frac{\beta}{2} \tag{3.5}$$

Then there exists $(\hat{x}, \hat{u})$ solution to the primal and dual problem respectively, such that $(x_k)_k$ and $(u_k)_k$ converge to $\hat{x}$ and $\hat{u}$ respectively.

The above theorem is a particular case of the theorem 3.1 in [8]. We propose a simplified version of the proof given in [8] for our particular case. I used all the result presented in the section 2, but also many other, you can find them in the Appendix A.1 and A.2. The full proof of this Theorem is provided in the Appendix A.3.

3.2 Unfolded proximal networks

Recently, optimisation algorithms have been outperformed by neural networks in a wide range of applications, from medicine to security and astronomy. However, despite providing high quality reconstructions, pure deep learning methods often lack theoretical guarantees (e.g., robustness against model approximation, generalisation, characterisation of the solution). To overcome this issue, neural network architectures grounded on optimisation theory have been proposed, leading to "unfolded neural networks". They consist in unrolling a finite number of iterations of an optimisation algorithm, while learning some of their linear operators [10, 12, 11]. Unfolded networks are usually much lighter than traditional networks (i.e., less learnable parameters), they have shown to be more robust in practice, and they reach at least as good reconstruction quality. Nevertheless, although they inherit some of the theoretical results of the associated optimisation algorithms (e.g., robustness), they loose the asymptotic convergence guarantees of the formers.

In [10], the authors proposed to unroll algorithm 3.4 for solving a similar problem as in 3.2-3.3. Indeed each step of the algorithm 3.4 will be transform in a layer of a neural network. One important idea is that in the algorithm 3.4, the parameters τ and σ and the matrix L are fixed during all the iterations, but it is surely better to adapt it at the different steps, that is why our neural network will learn for all steps these parameters : we will use τ_k, σ_k and L_k.

First we have to fix a number of iteration $K \in \mathbb{N}$, it will be the depth of the neural network. Then we can describe how to construct a layer : classically we write $a_{k+1} = \eta_k(D_k a_k + b_k)$ where a_k is the input of the layer k, D_k and b_k are the learnable parameters (the weights and the bias respectively) and η_k is the activation function of the layer k.

Here we want "$a_k = \begin{pmatrix} x_k \\ u_k \end{pmatrix}$", where (x_k, u_k) are like in the algorithm (3.4). In [10] it is possible because the two steps of their algorithm are not as entangled as the ones of (3.4). In our study, we need a bit more complex architecture : $\forall a_k \in \mathbb{R}^{N+S}, \forall x \in \mathbb{R}^N$,

$$\begin{cases} \mathcal{L}_k^x(a_k) = \eta^x(D_k^x a_k + b_k^x) \\ \mathcal{L}_k^u \begin{pmatrix} x \\ a_k \end{pmatrix} = \eta_k^u \left(D_k^u \begin{pmatrix} x \\ a_k \end{pmatrix} \right) \end{cases} \tag{3.6}$$

Where:

$$\begin{cases} \eta^x = \Pi_C & D_k^x = \begin{pmatrix} I_N - \tau_k H^* H & -\tau_k L_k^* \end{pmatrix} & b_k^x = \tau_k H^* y \\ \eta_k^u = \Pi_{[-\lambda_k, \lambda_k]^S} & D_k^u = \begin{pmatrix} 2\sigma_k L_k & -\sigma_k L_k & I_S \end{pmatrix} & \end{cases} \tag{3.7}$$

And then we define

$$x_{k+1} = \mathcal{L}_k^x(a_k),\, u_{k+1} = \mathcal{L}_k^u \begin{pmatrix} \mathcal{L}_k^x(a_k) \\ a_k \end{pmatrix},\, a_{k+1} = \begin{pmatrix} \mathcal{L}_k^x(a_k) & \mathcal{L}_k^u \begin{pmatrix} \mathcal{L}_k^x(a_k) \\ a_k \end{pmatrix} \end{pmatrix}^T =: T_k(a_k) \tag{3.8}$$

Remark 3.3 We can notice that if we take $C = \mathbb{R}^N$ we obtain the easier case develop in [10].

In the next we define a neural network which uses the transition rule (3.8) for the deep layers. For the input layer, if we denote like before $y \in \mathbb{R}^M$ the noisy data, a classical input [10] for the algorithm (3.4) is $H^*(y) \in \mathbb{R}^N$ for the primal variable, and $L_0(H^*(y)) \in \mathbb{R}^S$ for the dual one. So we will take for input of our neural network $H^*(y)$, and the first linear transition will be $In : z \mapsto (z, L_0(z))$. For the last transition, we want in output the primal variable, so we choose a transition $Out : (x, u) \mapsto x$. You can visualise the network on the figures 1 and 2.

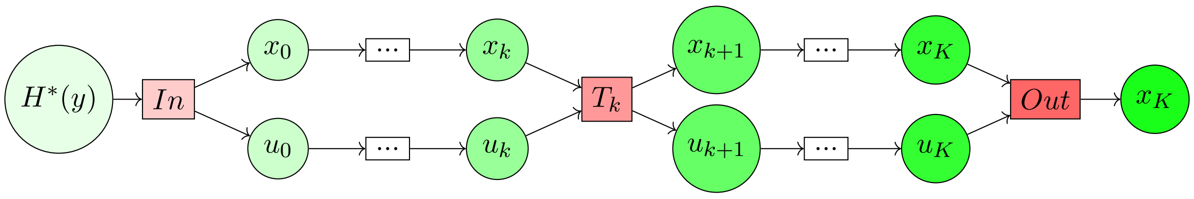


Figure 1: Global architecture of our unrolled neural network.

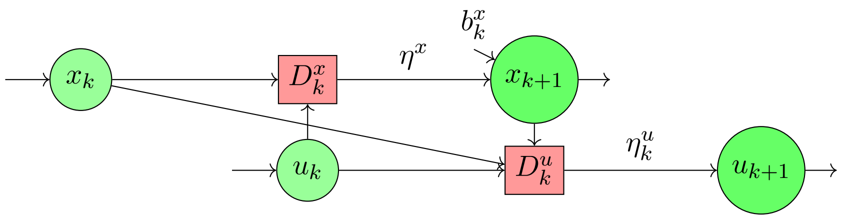


Figure 2: A deep layer T_k of our unrolled neural network.

The next result explain the links between the forward backward algorithm and the deep learning approach.

Proposition 3.4 *Let $y \in \mathbb{R}^M$ be the input of our neural network. If we denote a_k the output of the k-th deep layer, and we suppose $\forall k, \tau_k = \tau, \sigma_k = \sigma, L_k = L$. We have $\forall k \in \mathbb{N}, a_k = (x_k, u_k)$, where (x_k, u_k) are obtained with the algorithm (3.4) for the input $(H^*(y), L(H^*(y))$.*

4 Implementation

In this section we will explain the implementation of an unrolled neural network, and compare this to standard variational methods with TV regularisation [13]. First we simplify a bit the problem, we fix the convex set $C = [0,1]^N$ in (3.3), it is the set of all gray-scale images with pixel values ranging in $[0,1]$. The noise w in (3.1) will be an additive centered white Gaussian noise with variable standard deviation $s > 0$. For the operator H, we consider two settings: first a denoising setting with $H = Id$, and second an inpainting setting where H corresponds to a masking operator.

The regularisation parameter λ will depend on this noise level, i.e., we fix $\lambda = s^2$. Hence we suppose known the standard deviation s of the noise before entering the neural network, but we can imagine models which can learn this parameter.

We will investigate the behavior of five methods: two variational approaches where L is the linear TV gradient operator [13]; and three unfolded methods based on Algorithm (3.4).

(i) **'TV 1' model:** The first variational model will apply $K \in \{5, 20, 50\}$ iterations of algorithm 3.4, where we choose L to be a total variation operator.

(ii) **'TV CV' model:** The second variational model will be the same as the above one, with asymptotic convergence: we stop the algorithm either after $K = 1000$ iterations, or when it satisfies the stopping condition $\|x_{k+1} - x_k\|/\|x_k\| < 10^{-6}$.

(iii) **'FC' (for Fully connected) model:** The third model is a neural network unrolling algorithm 3.4 with $K \in \{5, 20, 50\}$ layers. It will learn the matrix L as fully connected layers. The input dimension of L is $N = 28^2$ (number of pixels of the image), and we will use different output dimensions, denoted by $S \in \{50, 100, 200\}$.

(iv) **'Conv 1' model:** The forth model will be the same than the third but we will use a convolution layer L with 64 features, and 3×3 kernel size.

(v) **'Conv 2' model:** The last one will be the same than the forth one, but it will learn K different convolution layers L_k. For this model only, we use a weight decay, fixed to the value 2×10^{-4}.

For all the models presented above, we use the theoretical condition (3.5) for the stepsizes, that we adapt as follows

$$\sigma_k > 0 \quad \text{and} \quad \tau_k = 0.99 \times \frac{1}{\beta/2 + \sigma_k \|L\|^2}. \tag{4.1}$$

In the above equation, for the network models, the σ_k are learned, while for the variational models **TV 1** and **TV CV** we fix $\sigma = 1/\|L\|^2$. Note that for model **Conv 2**, since L_k is different at each layer, the above condition needs to be adapted, and we use $\tau_k = 0.99 \times \frac{1}{\beta/2+\sigma_k\|L_k\|^2}$.

Training setting: For each evaluation of the neural network during the training, we choose the noise standard deviation s uniformly randomly in $[0, 0.15]$ to create the observed images.

The networks will be trained using the dataset MNIST, this dataset contain 60000 gray-scale pictures of written numbers, this pictures has 28×28 pixels. We use a batch size of 100. We use the Adam optimiser with a starting learning rate of 10^{-3} for the convolution model and 10^{-2} for the linear one, and we divide the learning rate by ten each ten epochs. We will always denote by K the number of iterations of the algorithm, hence with the architecture presented in the figure 1, we will use $K + 1$ layers (we add the input layer, the Out one has not to be learn). Notice that the choice of the hyper-parameters is the result of a lot of different test, indeed this was the hardest part of the internship for me. In the next we will use few different models to compare their accuracy.

4.1 Pure denoising

In this section we consider problem 3.1 with $H = Id$, and we focus on denoising. For the image reconstruction problem we generally use the snr to compare the estimate of images, which quantify the amount of noise removed by the model, a bigger snr is a better reconstruction :

$$\text{snr}(x, y) = 20 \log_{10}(\frac{\|x\|}{\|x - y\|})$$

There are two interesting values : the $\text{snr}_{in} = \text{snr}(\bar{x}, y)$ where $\bar{x}$ is the true image and y is the noised one (the input of the neural network), and the $\text{snr}_{out} = \text{snr}(\bar{x}, x_{out})$, where x_{out} is the output of the neural network. We expect the snr_{out} to be greater than the snr_{in}, and we will compare to model using the fact that for a same snr_{in}, the model which have the bigger snr_{out} is the best.

First we try to understand the influence of the parameter S on the fully connected model, so you can see in the table 1 the snr_{out} of the fully connected model with different values of S and N. We understand that the influence of S grows with the number of layers. Hence we choose every time $S = 200$ in the next.

	$S = 50$	$S = 100$	$S = 200$
5 layers	18.80	18.81	18.80
20 layers	18.81	18.83	18.85
50 layers	18.89	18.93	18.95

Table 1: snr_{out} for different number of layers and values of S of the fully connected model.

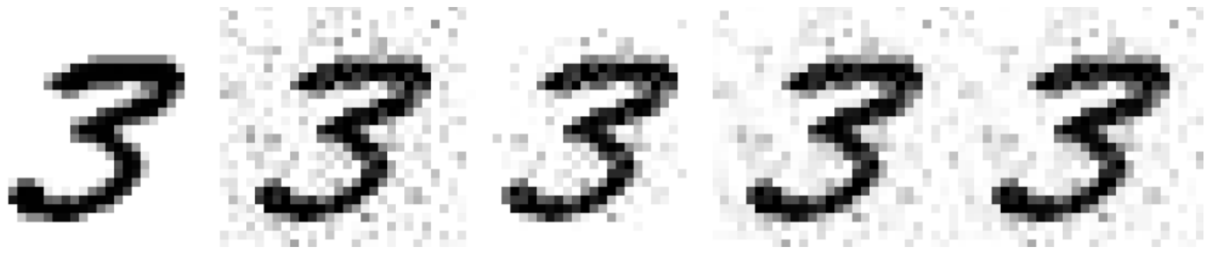


Figure 3: An image, its noised version and the denoised image using **FC**, **TV 1** and finally **TV CV**.

The table 2 and 3 shows respectively the values of the l_1-norm and the snr_{out} of the different models. Notice that the model **TV CV** stops after 312 iterations.

	FC	**Conv 1**	**Conv 2**	**TV 1**	**TV CV**
5 layers	1.77	1.80	1.21	2.87	
20 layers	1.76	1.55	1.06	2.80	2.80
50 layers	1.74	1.53	1.07	2.80	

Table 2: Average of the l_1-norm ($\times 10^{-2}$) on the validation set for different number of layers.

	snr_{in}	**FC**	**Conv 1**	**Conv 2**	**TV 1**	**TV CV**
5 layers		18.80	20.56	21.57	18.84	
20 layers	15.45	18.85	21.43	22.73	19.13	19.13
50 layers		18.95	21.46	22.70	19.13	

Table 3: Average of the snr_{out} (dB) on the validation set for different number of layers.

We can see that the model **Conv 2** which uses different matrix L_k is clearly the best and it is understandable. It is more interesting to compare the fully connected and the **Conv 1** models because they have a more similar structure. For pure denoising, we understand that the convolution is the best, it has fewer weights to learn ($64 \times 9 + K$), and better results.

Notice that to compute the value of τ_k, we need the operator norm of L_k, and the computation of this norm is a complex operation. That why in reality the model **Conv 2** has fewer parameters than the fully connected model, but it takes much more time (it has to compute $\|L_0\|$, ..., $\|L_{K-1}\|$) to train (the fully connected one only has to compute $\|L\|$). You can see on the figure 4 an example of a denoised image with the model **Conv 2**.

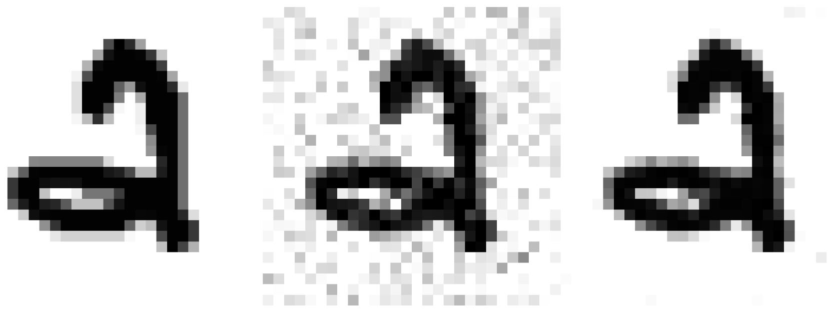


Figure 4: An image, its noised version and the denoised image using **Conv 2**.

4.2 Inpainting

In this section we use H an inpainting operator with a sub-sampling ratio of 0.5, hence $M = N/2$. H is fixed at the beginning of the experiment : we choose uniformly the pixels which will be canceled by H, and then we never change this choice. For the fully connected model we will always take $S = 200$. You can see on the table 4 that for a small number of layer, the linear model is the best, but it does not improve to much when the number of layers grows, whereas the two convolution model **Conv 1** and **Conv 2** follow the opposite rule. The model **TV CV** used 1000 iterations for the inpainting, hence it needs more iterations to converge. You can see on the figure 5 an example of a denoised image using the **Conv 1** model.

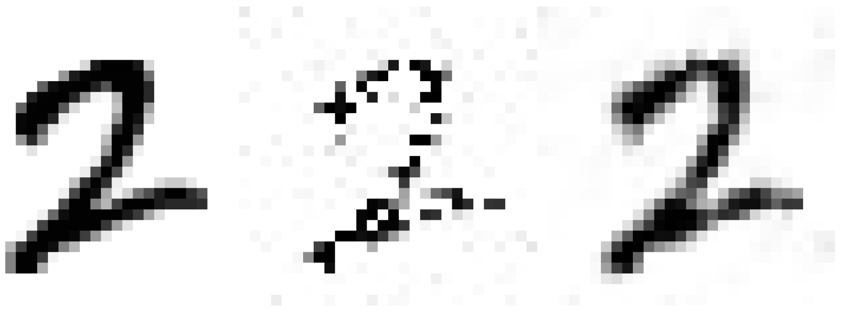


Figure 5: An example of an image, its noised version and the denoised image using **Conv 1**.

	snr$_{in}$	FC	Conv 1	Conv 2	TV 1	TV CV
5 layers	2.62	9.46	3.03	6.40	3.11	6.47
20 layers		9.97	10.59	10.77	3.94	
50 layers		9.91	10.65	11.73	4.88	

Table 4: Average of the snr$_{out}$ (dB) on the validation set for different number of layers.

5 Lipschitz constant

An important characteristic of a model in deep learning is the robustness. A model is said to be robust if its output on two similar input are similar. In particular in image reconstruction it is very important because we consider noised images, and we want to minimise the influence of the noise, adding a little noise to an image should not affect to much the output of the model. That why we are interested in the Lipschitz constant of our neural network, because a small Lipschitz constant means a very robust model. We will see three different ways to compute the Lipschitz constant of our neural network defined in section 3.2.

5.1 Jacobian method

The first idea to compute a Lipschitz constant of our neural network is to think about the mean value inequality, and we understand that a Lipschitz constant (indeed it is the best Lipschitz constant) of a differentiable function $f : \mathbb{R}^N \to \mathbb{R}^M$ can be express using its Jacobian like that : $\sup_{x\in\mathbb{R}^N} \|J_f(x)\|$. As we can not compute the norm $\|J_f(x)\|$ in a lot of points x (it would be to long, the computation of an operator norm has a big complexity), we just compute it in a few points, generally a hundred, and then we take the maximum. We understand that in reality this method does not give a Lipschitz constant but a lower bound to the best Lipschitz constant. The issue is the accuracy of the method, we can not know if this lower bound is good or not. We denote α_{jac} this lower bound Lipschitz constant.

5.2 Layer decomposition formula

This subsection is inspired by the works found in [6]. Indeed we know that the projections are firmly non expansive (see the appendix). Then if we have to rewrite the transition rule 3.8 of our neural network : $a_{k+1} = T_k(a_k) = P_k^{(1)} \circ V_k^{(1)} \circ P_k^{(0)} \circ V_k^{(0)}(a_k)$ in order to specify the linear parts, and the firmly non-expansive ones, the mathematical forms of $P_k^{(1)}$, $V_k^{(1)}$, $P_k^{(0)}$ and $V_k^{(0)}$ are in the appendix, but they are represented on the figure 6, and 7.

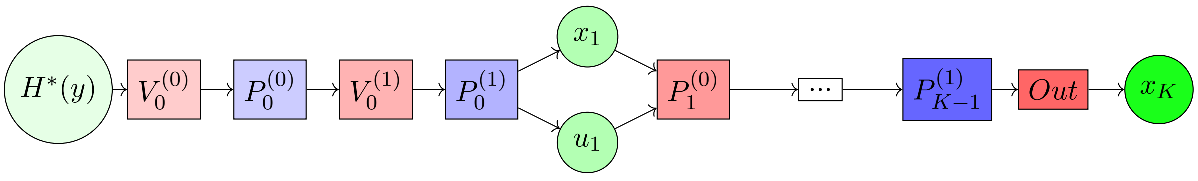


Figure 6: Global architecture of our unrolled neural network.

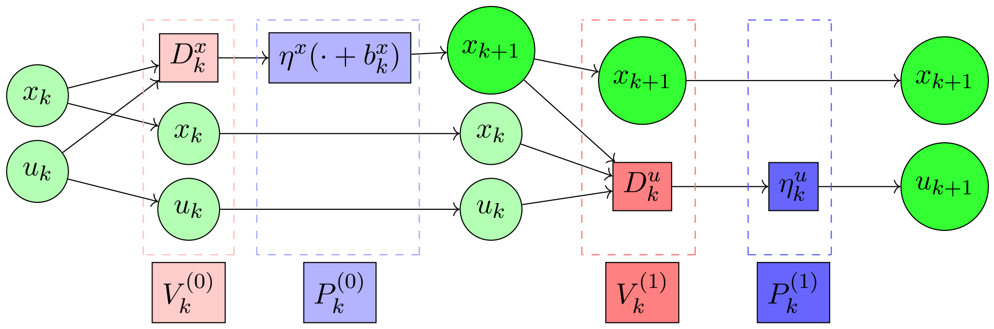


Figure 7: Decomposition into linear operators and firmly non expansive functions of our unrolled neural network.

The above figures are just reformulation of figures 1 and 2, we have not modify the neural network. Then using a technical induction (which you can find in [7, Thm.4.2, Prop.4.3]), we can show the following theorem.

Theorem 5.1 *[7, Thm.4.2, Prop.4.3] Let $K \in \mathbb{N}^*$ be the fixed depth of our unfolded neural network. We define by induction the sequence :*

$$\theta_{-1} = 1 \qquad \forall n \in [\![0, 2K]\!], \;\; \theta_n = \sum_{i=0}^{n} \theta_{i-1} \| W_n \circ ... \circ W_i \|$$

Where :

$$(W_{2K}, ..., W_0) = (Out, V_{K-1}^{(1)}, V_{K-1}^{(0)}, ..., V_0^{(1)}, V_0^{(0)}), \qquad \forall k \in [\![0, 2K-1]\!], \;\; W_k = V_{k//2}^{(k\%2)}$$

Then our neural network is α-Lipschitz, for :

$$\alpha = \frac{\theta_{2K}}{2^{2K}}$$

We will denote α_{lay} this Lipschitz constant.

5.3 Ortho-diagonalisation formula

The final method followed the previous one, indeed we will use the theorem 5.1. The problem with this theorem is that it gives a too complex algorithm to compute a Lipschitz constant, its complexity is at least $O(K^2 \times N^3) \approx O(N^3)$ because in order to compute the operator norms, we have to multiply matrices of size bigger than $N \times N$. K the depth of the network is generally small compared to N the number of pixel of the image. For our database MNIST, $N^3 \approx 5 \times 10^8$, it is computable, but if we take images of better quality, we can think about the ImageNet dataset, with an average of 469×387 pixels, we have $N^3 \approx 6 \times 10^{15}$. Hence we need to develop a more efficient algorithm. Using some inspiration in [1], we develop a new theorem :

Theorem 5.2 *Let $K \in \mathbb{N}^*$ be the fixed depth of the unfolded neural network. Under the hypothesis :*

(i) *L does not depends on the layer, $L_k := L, \forall k \in [\![0, K-1]\!]$.*

(ii) *H^*H and L^*L commute.*

Our neural network is α-Lipschitz, where α can be computed following the sketched algorithm below.

(i) *Diagonalise H^*H and L^*L in the same orthonormal basis $(v_p)_{p \in [\![1,N]\!]}$.*

(ii) *Define $M_k = V_k^{(1)} V_k^{(0)}$ and $M_{n,i} = M_n M_{n-1} ... M_i$, then notice that :*

$$M_{n,i} = \begin{pmatrix} A_{n,i} & B_{n,i} L^* \\ L C_{n,i} & I_S + L D_{n,i} L^* \end{pmatrix}$$

where $A_{n,i}$, $B_{n,i}$, $C_{n,i}$ and $D_{n,i}$ are defined by induction, and they are multinomial in H^*H and L^*L, hence we can diagonalise them in the eigenbasis (v_p).

(iii) *we have two variable, the primal $x \in \mathbb{R}^N$, and the dual $u \in \mathbb{R}^S$, but we have to introduce $x' = L^*(u)$, doing that we can explicit the value of $\|M_{n,i} \begin{pmatrix} x \\ u \end{pmatrix}\|$ as a function of x and x'. With the approximation $\|x'\| \leqslant \|L\| \|u\|$, taking the supremum in x and u of norm less than 1 we find an estimate of $\|M_{n,i}\|$.*

(iv) *We have obtain an approximation of $\|M_{n,i}\|$ as a function of the eigenvalues of L^*L, $A_{n,i}$, $B_{n,i}$, $C_{n,i}$ and $D_{n,i}$, and we have to do something similar to obtain approximation of $\|V_{n+1}^{(0)} M_{n,i}\|$, $\|M_{n,i} V_{i-1}^{(1)}\|$ and $\|V_{n+1}^{(0)} M_{n,i} V_{i-1}^{(1)}\|$, then we just use the theorem 5.1.*

If we suppose known the eigenvalues of H^*H and L^*L in a simultaneous orthonormal diagonalisation basis, the computation complexity of the above algorithm is of the order $O(N \times K^2) \approx O(N)$.

Proof. The full proof is available in Appendix A.4. ☐

We will denote α_{approx} this Lipschitz constant. As H is fixed, we can diagonalise H^*H before the training, hence it is understandable to not count the complexity of this operation. Then these hypotheses implies a particular mathematical form for L, this is explain in the section B of the Appendix. This form depends on H, we can generally compute easily the diagonalisation of L^*L. Notice that in the worst case, if $H = Id$, the hypothesis (ii) is verified for all matrix $L \in \mathbb{R}^{S\times N}$. Hence we have to diagonalise L^*L, which is equivalent to compute a singular value decomposition of L, we need $O(N \times S^2)$ operations, which is generally much better than $O(N^3)$.

To compute this Lipschitz constant we make some approximation, hence we obtain a Lipschitz constant which is fast to compute, but of worst quality than the previous one. We have : $\alpha_{jac} \leqslant \alpha \leqslant \alpha_{lay} \leqslant \alpha_{approx}$, if we denote α the best (the smallest) Lipschitz constant of our neural network.

Sometimes the hypothesis (ii) is too strong. For example, if H is an inpainting operator, we can show (see the section B.2 of the appendix) that L has to act independently on the canceled pixels, and on the preserved ones, explicitly if we denote F_c the subspace of $\mathbb{R}^N$ of the cancel pixels, and F_p the one of the preserved pixels, we have :

$$L(F_c) \cap L(F_p) = \{0\}$$

This is not desirable because to estimate the masked pixels, we need to use their neighbors, we want to use the preserved pixels to reconstruct the cancel ones, but it is not possible with the hypothesis (ii). The intuition is good here because experimentally we obtain a very bad model if we force it to verify this hypothesis with an inpainting operator H. Hence the model can cancel the noise, but it does not reconstruct the pixels, like you can see on the figure 8.

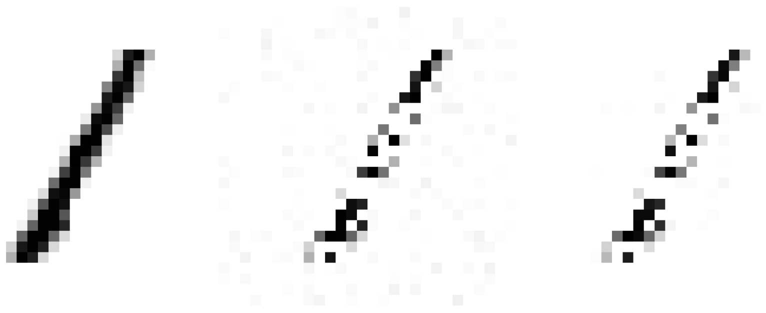


Figure 8: An example of image reconstruction verifying hypothesis (ii)

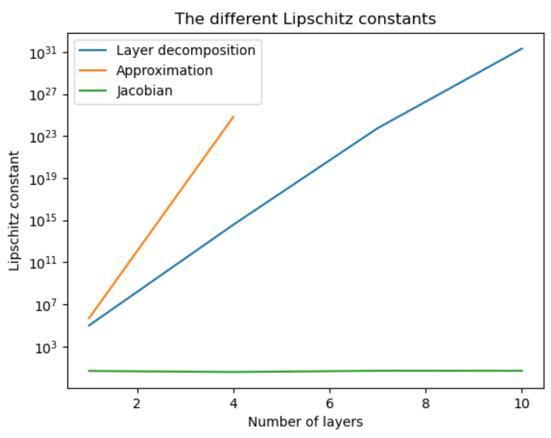
But for the pure denoising problem $H = Id$, the hypothesis (ii) is always verified, so we can test the value of the different Lipschitz constants of the fully connected model (ii) (this model is easier because it gives explicitly the matrix L).

5.4 Comparison of the different Lipschitz constant

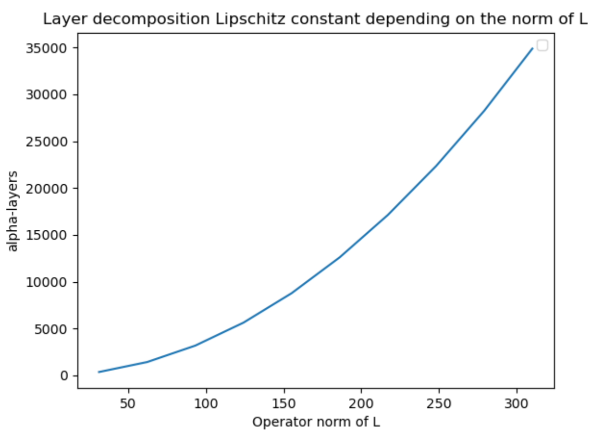
We implement the three presented way to compute the Lipschitz constant, to experiment their accuracy, and to compare them. In the next we will use the fully connected model with different numbers of layers but always $S = 200$. We simplify a bit the problem : the noise will be fixed, $s = 0.15$ would not change anymore.

A first result is that the layer decomposition Lipschitz constant α_{lay} is huge, even for small number of layers. But if you look at the demonstration of its formula in [7], you can see that we use as little approximation as possible. For example in the article [1], you can see they use the same Lipschitz constant α_{lay} obtain with Theorem 5.1, and in their experimentation in section 4, they obtain excellent result. With more time we should have try, like in [1], to take an Abel integral operators H.

We can understand that the problem we are trying to solve is maybe to hard, or the fully connected model is not suitable. Hence we can not say if the Lispchitz constant α that is too loose, or if the fully connected neural network is truly very not robust.



(a) The three Lipschitz constants in log-scale, for different number of layers.



(b) The layer decomposition Lipschitz constants with respect to the operator norm of L.

Figure 9: Study of the variations of the Lipschitz constants for the fully connected model

Then we can see that α_{lay} is huge behind α_{jac} which is always near from 1, but as α_{jac} is a lower-bound of the Lipschitz constant, we do not know if the best approximation of α is α_{lay} or α_{jac}. With more time, we should have study this question. We also obtain that generally α_{approx} is huge compared to α_{lay}. So we tried to understand why those two Lipschitz constant was so loose. So first, we can see that the deeper the network is, the looser the approximation becomes, it is well shown on the figure 9a. We can see that the grow of α_{lay} and α_{approx} are exponential in the number of layers as the curves are in log-scale.

I also find that the Lispchitz constant grow with the operator norm of L, it is hard to show, you can see it on the figure 9b. To produce this figure a took a trained neural network, and I plot its α_{lay} Lipschitz constant but artificially changing L by $\frac{k}{10} \times L$ for $k \in [\![1, 10]\!]$. With more time we should have try to explain better this correlation.

I also try to take a convolution layer L. The problem is that we can not represent it by a matrix, indeed it would be a to huge matrix. Hence it is not easy to apply the formula of Theorem 5.1 to compute α_{lay}, the terms $W_n \circ W_{n-1} \circ \ldots W_i$ are composition of functions and not matrix product now, so it is very hard to implement, and I did not have enough time at the end of my internship.

For the Lipschitz constant α_{approx}, we can think it is easier to compute, but we have to diagonalise L^*L, I start to study this in section B.4 of the Appendix, but I did not have enough time to finalise and to implement it.

6 Conclusion

Those three months were very interesting for me and I really like the subject of the internship. I worked on several different topics, some was similar to what I can do at the ENS, not on the contents which was always completely new, but on the fact that they uses very theoretical mathematics. But I also work on more applied math, and computer science with the neural networks, thankfully I was able to ask my coworkers and my internship supervisor some help, because this was quite new for me. This internship helps me to think about what I want to do for my second year of master degree, in order to prepare the best my thesis. I would have liked to have more time because the experimentation was very complex, and after that I did not have enough time to study, and to improve my results about the Lipschitz constant. But I also think that this internship presents a relatively complete project, and the different steps of the scientific approach.

References

[1] Emilie Chouzenoux, Cecile Della Valle, and Jean-Christophe Pesquet. Inversion of integral models: a neural network approach. *arXiv preprint arXiv:2105.15044*, 2021.

[2] Heinz H. Bauschke Patrick L. Combettes. Convex analysis and monotone operator theory in hilbert spaces. 2016.

[3] Patrick L. Combettes. Quasi-fejérian analysis of some optimization algorithms. In Dan Butnariu, Yair Censor, and Simeon Reich, editors, *Inherently Parallel Algorithms in Feasibility and Optimization and their Applications*, volume 8 of *Studies in Computational Mathematics*, pages 115–152. Elsevier, 2001.

[4] Patrick L Combettes*. Solving monotone inclusions via compositions of nonexpansive averaged operators. *Optimization*, 53(5-6):475–504, 2004.

[5] Patrick L Combettes and Jean-Christophe Pesquet. Proximal splitting methods in signal processing. *Fixed-point algorithms for inverse problems in science and engineering*, pages 185–212, 2011.

[6] Patrick L Combettes and Jean-Christophe Pesquet. Lipschitz certificates for layered network structures driven by averaged activation operators. *SIAM Journal on Mathematics of Data Science*, 2(2):529–557, 2020.

[7] Patrick L Combettes and Jean-Christophe Pesquet. Lipschitz certificates for layered network structures driven by averaged activation operators. *SIAM Journal on Mathematics of Data Science*, 2(2):529–557, 2020.

[8] Laurent Condat. A primal–dual splitting method for convex optimization involving lipschitzian, proximable and linear composite terms. *Journal of optimization theory and applications*, 158(2):460–479, 2013.

[9] J ECKSTEIN and D. P BERTSEKAS. On the douglas-rachford splitting method and the proximal point algorithm for maximal monotone operators. *Mathematical programming*, 55(3):293–318, 1992.

[10] Mingyuan Jiu and Nelly Pustelnik. A deep primal-dual proximal network for image restoration. *IEEE Journal of Selected Topics in Signal Processing*, 15(2):190–203, 2021.

[11] Hoang Trieu Vy Le, Audrey Repetti, and Nelly Pustelnik. PNN: From proximal algorithms to robust unfolded image denoising networks and plug-and-play methods. *arXiv preprint arXiv:2308.03139*, 2023.

[12] Audrey Repetti, Matthieu Terris, Yves Wiaux, and Jean-Christophe Pesquet. Dual forward-backward unfolded network for flexible plug-and-play. In *2022 30th European Signal Processing Conference (EUSIPCO)*, pages 957–961. IEEE, 2022.

[13] Leonid I Rudin, Stanley Osher, and Emad Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D: nonlinear phenomena*, 60(1-4):259–268, 1992.

Appendix

A Proof of the theorems

A.1 Basic properties

Proposition A.1 *[2] Let $f \in \Gamma_0(H)$, for all $\bar{p} \in \operatorname{dom}(f)$, for all $\bar{x} \in H$,*

$$\bar{p} = \operatorname{prox}_f(\bar{x}) \text{ if and only if } \bar{x} - \bar{p} \in \partial f(\bar{p})$$

Lemma A.2 *[2] If $f \in \Gamma_0(H)$, then ∂f is a maximally monotone operator.*

Lemma A.3 *[2] Let H be an Hilbert space,*

(i) *$f : H \to H$ is firmly non expansive if and only if :*

$$\forall (z, z') \in H^2, \quad \langle f(z) - f(z') \mid z - z' \rangle \geqslant \|f(z) - f(z')\|^2.$$

(ii) *$f : H \to H$ is δ-averaged for $\delta \in (0,1)$, if and only if :*

$$\forall (z, z') \in H^2, \quad \|z - z'\|^2 - \frac{1-\delta}{\delta}\|(I-f)(z) - (I-f)(z')\|^2 \geqslant \|f(z) - f(z')\|^2.$$

Lemma A.4 *If $f \in \Gamma_0(H)$ such that f is differentiable, then*

$$\nabla f \text{ is non expansive} \iff \nabla f \text{ is firmly non expansive}$$

Theorem A.5 *[9, 2] Let H be a Hilbert space and M a set-valued function. Then M is maximally monotone if and only if $(I+M)^{-1}$ is firmly non expansive. In particular, if M is maximally monotone, $(I+M)^{-1}$ is a single-valued operator (it is a classical operator), well define everywhere (*$\operatorname{dom}((I+M)^{-1}) = H$*).*

Lemma A.6 *Let $L : \mathbb{R}^N \to \mathbb{R}^S$ a matrix. We have :*

$$\|L\|^2 = \|L^*\|^2 = \|L^*L\| = \|LL^*\|$$

Proof. Let $x \in \mathbb{R}^N$, using Cauchy-Schwarz and the properties of operator norm :

$$\|Lx\|^2 = \langle Lx \mid Lx \rangle = \langle L^*Lx \mid x \rangle \leqslant \|LL^*\|\|x\|^2 \leqslant \|L\|\|L^*\|\|x\|^2$$

As $L \mapsto L^*$ is an involution, we can exchange L and L^* is the above formula. Hence we have :

$$\begin{cases} \|L\|^2 & \leqslant \|L^*L\| \leqslant \|L^*\|\|L\| \\ \|L^*\|^2 & \leqslant \|LL^*\| \leqslant \|L^*\|\|L\| \end{cases} \tag{A.1}$$

So in particular, $\|L\| \leqslant \|L^*\|$ and $\|L^*\| \leqslant \|L\|$, $\|L\| = \|L^*\|$. Knowing that and using (A.1), $\|L^*\|^2 = \|L\|^2 = \|L^*L\| = \|LL^*\|$. □

Lemma A.7 • *Let $k \in \mathbb{N}, k \geqslant 2$. For all $p \in [\![1, N]\!]$, we define Γ^p a $k \times k$ symmetric matrix. And we define Γ is the $kN \times kN$ symmetric block matrix :*

$$\Gamma := \begin{pmatrix} \Gamma^1 & & (0) \\ & \ddots & \\ (0) & & \Gamma^N \end{pmatrix}$$

Then we have :

$$\|\Gamma\| = \max_{p \in [\![1,N]\!]} \|\Gamma^p\|$$

Notice that there exists efficient algorithm to compute $\|\Gamma^p\|$, and the smaller k is, the faster the calculation is.

- *More specifically, if $k = 2$, we write :*

$$\Gamma^p = \begin{pmatrix} \alpha_p & \beta_p \\ \beta_p & \gamma_p \end{pmatrix}$$

And we have :

$$\|\Gamma\| = \frac{1}{2} \max_{p \in [\![1,N]\!]} \left(|\alpha_p + \gamma_p| + \sqrt{(\alpha_p - \gamma_p)^2 + 4\beta_p^2} \right)$$

Proof. As Γ is symmetric and block diagonal, we know that :

$$\|\Gamma\| = \max_{\lambda \in \mathrm{Sp}\,(\Gamma)} |\lambda| = \max_{p \in [\![1,N]\!]} \max_{\lambda \in \mathrm{Sp}\,(\Gamma^p)} |\lambda|$$

Now we compute for all $p \in [\![1, N]\!]$, $\max_{\lambda \in \mathrm{Sp}\,(\Gamma^p)} |\lambda|$, to do that we define the characteristic polynomial :

$$\begin{aligned} \det(XI_2 - \Gamma^p) &= \begin{vmatrix} X - \alpha_p & -\beta_p \\ -\beta_p & X - \gamma_p \end{vmatrix} \\ &= X^2 - X(\alpha_p + \gamma_p) + \alpha_p\gamma_p - \beta_p^2 \end{aligned}$$

Then we compute the discriminant Δ of this polynomial :

$$\Delta = (\alpha_p + \gamma_p)^2 + 4\beta_p^2 - 4\alpha_p\gamma_p = (\alpha_p - \gamma_p)^2 + 4\beta_p^2 \geqslant 0$$

Hence we only have real roots. Then we compute the maximum of the absolute values of these roots in two cases : either $\alpha_p + \gamma_p \leqslant 0$ or $\alpha_p + \gamma_p \geqslant 0$ and we obtain :

$$\max \mathrm{Sp}\,(\Gamma^p) = \frac{1}{2}\left(|\alpha_p + \gamma_p| + \sqrt{(\alpha_p - \gamma_p)^2 + 4\beta_p^2} \right)$$

□

A.2 Proof of Proposition 2.2

On the one hand, according to Lemma A.3,

$$\forall (z,z') \in H^2, \quad \|z-z'\|^2 - \frac{1-\delta}{\delta}\|(I-f)(z)-(I-f)(z')\|^2 \geqslant \|f(z)-f(z')\|^2 \tag{A.2}$$

Since Fix (f) is non empty, let $\tilde{z} \in$ Fix (f) such that $f(\tilde{z}) = \tilde{z}$. Let $k \in \mathbb{N}$. Using (A.2) we have :

$$\begin{aligned}\|f(z_k)-f(\tilde{z})\|^2 = \|z_{k+1}-\tilde{z}\|^2 &\leqslant \|z_k-\tilde{z}\|^2 - \frac{1-\delta}{\delta}\|(I-f)(z_k)-(I-f)(\tilde{z})\|^2 \\ &\leqslant \|z_k-\tilde{z}\|^2 - \frac{1-\delta}{\delta}\|z_k-z_{k+1}\|^2.\end{aligned}$$

Summing the last inequality over $k \in \{l,\dots,K\}$, for $K > l > 0$, we obtain

$$\begin{aligned}\sum_{k=l}^{K}\|z_{k+1}-z_k\|^2 &\leqslant \frac{\delta}{1-\delta}\sum_{k=l}^{K}\|z_k-\tilde{z}\|^2 - \|z_{k+1}-\tilde{z}\|^2 \\ &\leqslant \frac{\delta}{1-\delta}\left(\|z_l-\tilde{z}\|^2 - \|z_{K+1}-\tilde{z}\|^2\right) \\ &\leqslant \frac{\delta}{1-\delta}\|z_l-\tilde{z}\|^2.\end{aligned} \tag{A.3}$$

Hence

$$\sum_{k=0}^{+\infty}\|z_{k+1}-z_k\|^2 < +\infty. \tag{A.4}$$

On the other hand, since f is non expansive, we have $\|f(z_k)-f(\tilde{z})\|^2 = \|z_{k+1}-\tilde{z}\|^2 \leqslant \|z_k-\tilde{z}\|^2$. Hence $(z_k)_{k\in\mathbb{N}}$ is a bounded sequence, and dist $(z_{k+1}, \text{Fix}\,(f)) \leqslant$ dist $(z_k, \text{Fix}\,(f))$, then we deduce that $\left(\text{dist}\,(z_k, \text{Fix}\,(f))\right)_{k\in\mathbb{N}}$ is a non increasing positive sequence, so it converges.

As $(z_k)_{k\in\mathbb{N}}$ is bounded, let $\phi : \mathbb{N} \to \mathbb{N}$ an increasing sequence such that $z_{\phi(k)} \to \bar{z}$ when $k \to +\infty$. By (A.4), we have $\|f(z_{\phi(k)}) - z_{\phi(k)}\| = \|z_{\phi(k)+1} - z_{\phi(k)}\| \to 0$ when $k \to +\infty$. So by continuity of f, we deduce $f(\bar{z}) = \bar{z}$. Since $\bar{z} \in$ Fix (f), $\liminf$ dist $(z_k, \text{Fix}\,(f)) = 0$. Then, since the sequence $\left(\text{dist}\,(z_k, \text{Fix}\,(f))\right)_{k\in\mathbb{N}}$ converges, we have

$$\lim_{k\to+\infty} \text{dist}\,(z_k, \text{Fix}\,(f)) = 0. \tag{A.5}$$

Finally, let $l \in \mathbb{N}^*$. Using Jensen's inequality, we have

$$\|z_k - z_{k+l}\|^2 \leqslant 2(\|z_k-\bar{z}\|^2 + \|z_{k+l}-\bar{z}\|^2)$$

Then using (A.3) with $K = k+l-1$ and $l = k$:

$$\begin{aligned}\|z_k - z_{k+l}\|^2 &\leqslant 4\|z_k-\bar{z}\|^2 - \frac{2(1-\delta)}{\delta}\sum_{i=k}^{k+l-1}\|z_{k+1}-z_k\|^2 \\ &\leqslant 4\text{dist}\,(z_k, \text{Fix}\,(f))^2 - \frac{2(1-\delta)}{\delta}\sum_{i=k}^{k+l-1}\|z_{k+1}-z_k\|^2.\end{aligned}$$

We can deduce from (A.4) and (A.5), that both terms in the last inequality tend to 0 when $k \to +\infty$. Hence $(z_k)_{k\in\mathbb{N}}$ is a Cauchy sequence, that converges in an Hilbert space, i.e. $z_k \to \bar{z}$. By continuity of f, $z_{k+1} = f(z_k) \to f(\bar{z})$ leads to $f(\bar{z}) = \bar{z} \in \text{Fix}\,(f)$.

A.3 Proof of Theorem 3.2

We will see a proof of the theorem 3.2, as it is long, I decided to define some propositions and lemmas to structure the ideas. Recall $\beta = \|H^*H\| > 0$, $\tau, \sigma > 0$ and there is just one hypothesis, (3.5) : $\frac{1}{\tau} - \sigma\|L\|^2 > \frac{\beta}{2}$. Let $(x_k)_k, (u_k)_k$ be the sequences generated by the algorithm 3.4 for fixed $x_0 \in \mathbb{R}^N, u_0 \in \mathbb{R}^S$. We need some preliminary results.

We will consider the matrix P :

$$P = \begin{pmatrix} \frac{1}{\tau}I_N & -L^* \\ -L & \frac{1}{\sigma}I_S \end{pmatrix}$$

the function B from $\mathbb{R}^N \times \mathbb{R}^S$ to $\mathbb{R}^N \times \mathbb{R}^S$:

$$B = \begin{pmatrix} H^*(H\cdot - y) & 0 \\ 0 & 0 \end{pmatrix}$$

and the set valued function A whose graph is in $(\mathbb{R}^N \times \mathbb{R}^S) \times (\mathbb{R}^N \times \mathbb{R}^S)$:

$$A = \begin{pmatrix} \partial\iota_C & L^* \\ -L & \partial\iota_{[-\lambda,\lambda]^S} \end{pmatrix}$$

The first result is inspired by [2] :

Proposition A.8 *Searching $(\widehat{x}, \widehat{u})$ as in 3.1 is equivalent to search*

$$(\tilde{x}, \tilde{u}) \in \mathbb{R}^N \times \mathbb{R}^S \textit{ such that } \begin{pmatrix} 0 \\ 0 \end{pmatrix} \in \begin{pmatrix} \partial\iota_C(\tilde{x}) + L^*\tilde{u} + \nabla\left(\frac{\|H\cdot - y\|^2}{2}\right)(\tilde{x}) \\ -L\tilde{x} + \partial\iota_{[-\lambda,\lambda]^S}(\tilde{u}) \end{pmatrix} = (A+B)(\tilde{x}, \tilde{u})$$

We also have that $(\widehat{x}, \widehat{u})$ exists, in our case of study, 3.1 has solutions.

Proof. We use the fact that $0 \in \text{sri}\,(L(\text{dom}\,\iota_C) - \text{dom}\,(\lambda\|\cdot\|_1)) = \text{sri}\,(L(C) - \mathbb{R}^S) = \text{sri}\,(\mathbb{R}^S)$. Where $\text{sri}\,(D)$ is the strong relative interior of D :

$$\text{sri}\,(D) := \{x \in D \mid R_{++}(D - x) = \{\nu y \mid \nu > 0, y \in D - x\} = \overline{span}(D - x)\}$$

Clearly, $\text{sri}\,(\mathbb{R}^S) = \mathbb{R}^S$. Then we have to use the Theorem 15.23, and Theorem 19.1 of [2]. □

All the next results A.9, A.10, A.11 and A.12 are inspired by the proof of theorem 3.2 given in [8], but the presented proofs are much more detailed.

Proposition A.9

$$-\underbrace{\begin{pmatrix} H^*(Hx_k - y) \\ 0 \end{pmatrix}}_{B(x_k,u_k)} \in \underbrace{\begin{pmatrix} \partial\iota_C(x_{k+1}) + L^*u_{k+1} \\ -Lx_{k+1} + \partial\iota_{[-\lambda,\lambda]^S}(u_{k+1}) \end{pmatrix}}_{A(x_{k+1},u_{k+1})} + P\begin{pmatrix} x_{k+1} - x_k \\ u_{k+1} - u_k \end{pmatrix}$$

Proof.

(i) $-H^*(Hx_k - y) \in \partial\iota_C(x_{k+1}) + L^*u_{k+1} + \frac{1}{\tau}(x_{k+1} - x_k) - L^*(u_{k+1} - u_k)$:

$$\begin{aligned}
&\iff -H^*(Hx_k - y) \in \partial\iota_C(x_{k+1}) + \frac{1}{\tau}(x_{k+1} - x_k) + L^*(u_k) \\
&\iff (-\tau H^*(Hx_k - y) - \tau L^*(u_k) + x_k) - \mathbf{x_{k+1}} \in \partial(\tau\iota_C)(\mathbf{x_{k+1}}) \\
&\iff x_{k+1} = \text{prox}_{\tau\iota_C}\left(x_k - \tau H^*(Hx_k - y) - \tau L^*(u_k)\right) && \text{by the proposition A.1} \\
&\iff x_{k+1} = \Pi_C\left(x_k - \tau H^*(Hx_k - y) - \tau L^*(u_k)\right) && \text{using the example 2.1}
\end{aligned}$$

(ii) $0 \in -Lx_{k+1} + \partial\iota_{[-\lambda,\lambda]^S}(u_{k+1}) - L(x_{k+1} - x_k) + \frac{1}{\sigma}(u_{k+1} - u_k)$

$$\begin{aligned}
&\iff (\sigma L(2x_{k+1} - x_k) + u_k) - \mathbf{u_{k+1}} \in \partial(\sigma\iota_{[-\lambda,\lambda]^S})(\mathbf{u_{k+1}}) \\
&\iff u_{k+1} = \text{prox}_{\sigma\iota_{[-\lambda,\lambda]^S}}\left(u_k + \sigma L(2x_{k+1} - x_k)\right) && \text{by the proposition A.1} \\
&\iff u_{k+1} = \Pi_{\iota_{[-\lambda,\lambda]^S}}\left(u_k + \sigma L(2x_{k+1} - x_k)\right) && \text{using the example 2.1}
\end{aligned}$$

So we have find exactly the two steps of the algorithm 3.4, the proposition is proved □

Lemma A.10 *P is a symmetric positive definite matrix. In particular P is reversible and we can define the P-scalar product :* $\langle x, y\rangle_P = \langle x, Py\rangle$.

Proof. P is clearly symmetric. Let $z \in \mathbb{R}^N \times \mathbb{R}^S$, $z = (x, y)$, we compute $z^T Pz$:

$$z^T Pz = \frac{1}{\tau}\|x\|^2 - 2\langle Lx, y\rangle + \frac{1}{\sigma}\|y\|^2$$

So by Cauchy-Schwarz, and operator norm,

$$z^T Pz \geqslant \frac{1}{\tau}\|x\|^2 - 2\|L\|\|x\|\|y\| + \frac{1}{\sigma}\|y\|^2 = Q(\|x\|)$$

Where Q is a polynomial, by computing its discriminant Δ :

$$\Delta = \frac{4\|y\|^2}{\sigma}\left(\sigma\|L\|^2 - \frac{1}{\tau}\right)$$

and using the hypothesis (3.5) : $\frac{1}{\tau} - \sigma\|L\|^2 > \frac{\beta}{2} > 0$, we can conclude :

$$z^T Pz > 0 \text{ if and only if } z \neq 0$$

□

Lemma A.11 *[8, 2] $P^{-1}A$ is maximally monotone in $\mathbb{R}^N \times \mathbb{R}^S$ equipped with the P-scalar product.*

Proof. We will show that $A_1 = \begin{pmatrix} 0 & L^* \\ -L & 0 \end{pmatrix}$ and $A_2 = \begin{pmatrix} \partial\iota_C & 0 \\ 0 & \partial\iota_{[-\lambda,\lambda]^S} \end{pmatrix}$ are maximally monotone operator for the classical scalar product, and A_1 has full domain. So by using the Corollary 25.5 of [2], $A = A_1 + A_2$ is a maximally monotone operator for the classical scalar product. Finally $P^{-1}A$ is maximally monotone for the P-scalar product : if $(z_1, w_1), (z_2, w_2) \in P^{-1}A$ then $(z_1, Pw_1), (z_2, Pw_2) \in A$, as A is maximally monotone,

$$\langle z_1 - z_2 \mid Pw_1 - Pw_2 \rangle \geqslant 0$$

And it is exactly saying that $P^{-1}A$ is monotone in $\mathbb{R}^N \times \mathbb{R}^S$ equipped with the P-scalar product. If M is a monotone operator such that $P^{-1}A \subset M$, we have $A \subset PM$, by maximality of A, $A = PM$ so $P^{-1}A = M$. We conclude the result.

- A_1 is maximally monotone : A_1 is an antisymmetric matrix, so $\forall z \in \mathbb{R}^N \times \mathbb{R}^S$, $\langle A_1 z \mid z \rangle = -\langle z \mid A_1 z \rangle = 0 \geqslant 0$, A_1 is monotone.

  To show that A_1 is maximally monotone we can try to add (z, w) to A_1 in order to define a new monotone operator, and see that necessary $(z, w) \in A_1$, here $w = A_1 z$. Suppose that

  $$\forall z' \in \mathbb{R}^N \times \mathbb{R}^S, \langle z - z' \mid w - A_1 z' \rangle \geqslant 0$$

  Then by computation, $\forall z' \in \mathbb{R}^N \times \mathbb{R}^S, \langle A_1 z - w \mid z' \rangle + \langle z \mid w \rangle \geqslant 0$, but it is not possible except if $A_1 z - w = 0$. We have the result.

- A_2 is maximally monotone : as ι_C and $\iota_{[-\lambda,\lambda]^S}$ are in $\Gamma_0(\mathbb{R}^N)$ and $\Gamma_0(\mathbb{R}^S)$ respectively, by lemma A.2 $\partial\iota_C$ and $\partial\iota_{[-\lambda,\lambda]^S}$ are maximally monotone in $\mathbb{R}^N$ and $\mathbb{R}^S$ respectively, with their classical scalar product. Then it is not hard to prove that $(x, u) \to (\partial\iota_C(x), \partial\iota_{[-\lambda,\lambda]^S}(u)) = A_2(x, u)$ is also a maximally monotone operator.

□

Lemma A.12 *[8] Let $\kappa = \frac{1}{\beta}(\frac{1}{\tau} - \sigma\|L\|^2)$, then $\kappa P^{-1}B$ is firmly non expansive in $\mathbb{R}^N \times \mathbb{R}^S$ equipped with the P-scalar product.*

Proof. First we show that $\kappa P^{-1}B$ is non expansive in $\mathbb{R}^N \times \mathbb{R}^S$ for the P-scalar product :

$$\begin{aligned}\|P^{-1}B(z) - P^{-1}B(z')\|_P^2 &= \langle P^{-1}B(z) - P^{-1}B(z') \mid B(z) - B(z') \rangle \\ &= \left\langle (\frac{1}{\tau}I - \sigma L^* L)^{-1}(H^* H(x - x')) \,\middle|\, H^* H(x - x') \right\rangle\end{aligned}$$

Here we have compute the first line of P^{-1} (we resolve a little system). We recall that $\frac{1}{\tau}I - \sigma L^* L$ is well reversible because $\|\tau\sigma L^* L\| = \tau\sigma\|L\|^2$ by lemma A.6 and $\|\tau\sigma L^* L\| < 1 - \frac{\tau\beta}{2} < 1$ by hypothesis (3.5), hence we have the formula :

$$(\frac{1}{\tau}I - \sigma L^*L)^{-1} = \tau(I - \tau\sigma L^*L)^{-1} = \tau \sum_{n\in\mathbb{N}} (\tau\sigma L^*L)^n$$
$$\Rightarrow \|(\frac{1}{\tau}I - \sigma L^*L)^{-1}\| \leqslant \tau \sum_{n\in\mathbb{N}} (\tau\sigma\|L\|^2)^n = \tau(1 - \tau\sigma\|L\|^2)^{-1} = (\frac{1}{\tau} - \sigma\|L\|^2)^{-1} = \frac{1}{\kappa\beta}$$

As $\beta = \|H^*H\|$:

$$\left\langle (\frac{1}{\tau}I - \sigma L^*L)^{-1}(H^*H(x-x')) \;\middle|\; H^*H(x-x') \right\rangle \leqslant \|(\frac{1}{\tau}I - \sigma L^*L)^{-1}\| \times \beta^2\|x-x'\|^2 \leqslant \frac{\beta}{\kappa}\|x-x'\|^2$$

And we have $\kappa\beta\|x-x'\|^2 = \left\langle z - z' \;\middle|\; \kappa\beta \begin{pmatrix} I_N & 0 \\ 0 & 0 \end{pmatrix} (z-z') \right\rangle \leqslant \langle z - z' \mid P(z-z')\rangle = \|z-z'\|_P^2$ because $P - \kappa\beta \begin{pmatrix} I_N & 0 \\ 0 & 0 \end{pmatrix}$ is non negative (the calculations are similar than in lemma A.10) :

Let $z \in \mathbb{R}^N \times \mathbb{R}^S, z = (x,y),\;\; z^T \left(P - \kappa\beta \begin{pmatrix} I_N & 0 \\ 0 & 0 \end{pmatrix}\right) z = \sigma\|L\|^2\|x\|^2 - 2\langle Lx, y\rangle + \frac{1}{\sigma}\|y\|^2$

Hence by the Cauchy-Schwarz inequality :

$$z^T \left(P - \kappa\beta \begin{pmatrix} I_N & 0 \\ 0 & 0 \end{pmatrix}\right) z \geqslant \sigma\|L\|^2\|x\|^2 - 2\|L\|\|x\|\|y\| + \frac{1}{\sigma}\|y\|^2 = Q(\|x\|)$$

We compute the discriminant of Q, we find : $\Delta = 0$ so we can conclude : $z^TPz \geqslant 0$.

We conclude :

$$\|\kappa P^{-1}B(z) - \kappa P^{-1}B(z')\|_P^2 \leqslant \kappa\beta\|x-x'\|^2 \leqslant \|z-z'\|_P^2$$

Now we define $J : (x,u) \to \kappa\frac{\|x-y\|^2}{2}$, we have $\nabla J(x,u) = \kappa \begin{pmatrix} H^*(Hx-y) \\ 0 \end{pmatrix} = \kappa B(x,u)$, and :

$$\left\langle \kappa P^{-1}B(z) \mid Pz' \right\rangle = \left\langle \kappa B(z) \mid z' \right\rangle = \left\langle \nabla J(z) \mid z' \right\rangle$$

So $\kappa P^{-1}B = \nabla_P J$ is non expansive, where ∇_P is the gradient in the space $\mathbb{R}^N \times \mathbb{R}^S$ equipped with the P-scalar product. By using the lemma A.4, $\kappa P^{-1}B$ is firmly non expansive. □

Now we have all we needed to prove the theorem 3.2 :

Proof. Let $(x_k)_{k\in\mathbb{N}}$ be generated by algorithm (3.4). A.9 says exactly that $-B(z_k) \in A(z_{k+1}) + P(z_{k+1} - z_k)$ where $\forall k \in \mathbb{N}, z_k = (x_k, u_k)$. So with A.10, we know that P is revertible and we find :

$$(I - P^{-1}B)(z_k) \in (I + P^{-1}A)(z_{k+1})$$

In the next, all the non expansive properties will by with respect to the $P-$norm, so I would not notify it anymore.

Using A.11 and the theorem A.5 we know that $(I+P^{-1}A)^{-1}$ is a firmly non expansive function. In particular we have the well defined formula :

$$z_{k+1} = (I+P^{-1}A)^{-1} \circ (I-P^{-1}B)(z_k) =: T(z_k)$$

Where $T : \mathbb{R}^N \times \mathbb{R}^S \to \mathbb{R}^N \times \mathbb{R}^S$ is a classical function. Let g_A a non expansive function such that $(I+P^{-1}A)^{-1} = \frac{1}{2}g_A + \frac{1}{2}I$.

Let $\kappa = \frac{1}{\beta}(\frac{1}{\tau} - \sigma\|L\|^2)$, we know that $\kappa > \frac{1}{2}$ by hypothesis (3.5). Hence we can define

$$\delta = \frac{2\kappa}{4\kappa - 1}$$

and $\delta \in (0,1)$.

By the lemma A.12 we also know that $\kappa P^{-1}B$ is firmly non expansive, so let g_B a non expansive function such that $\kappa P^{-1}B = \frac{1}{2}g_B + \frac{1}{2}I$, we have $I - P^{-1}B = \left(1 - \frac{1}{2\kappa}\right) I - \frac{1}{2\kappa}g_B$. We define a function g such that :

$$T = (I+P^{-1}A)^{-1} \circ (I-P^{-1}B) = \frac{2\kappa}{4\kappa-1}g + \frac{2\kappa-1}{4\kappa-1}I = \delta g + (1-\delta)I$$

By calculation,

$$g = \frac{1}{8\kappa^2}\left[(1-2\kappa)I - (4\kappa-1)g_B + 2\kappa(4\kappa-1)g_A \circ (I-P^{-1}B)\right]$$

where $1-2\kappa, 4\kappa-1, 2\kappa(4\kappa-1) \geqslant 0$ and $I, g_B, g_A \circ (I-P^{-1}B)$ are all 1-Lipschitz functions. So using the triangle inequality :

$$\begin{aligned}\forall z, z' \in (\mathbb{R}^N \times \mathbb{R}^S), \|g(z) - g(z')\|_P &\leqslant \frac{1}{8\kappa^2}\left((1-2\kappa) + (4\kappa-1) + 2\kappa(4\kappa-1)\right)\|z-z'\|_P \\ &\leqslant \frac{1}{8\kappa^2} \times 8\kappa^2\|z-z'\|_P = \|z-z'\|_P\end{aligned}$$

Hence we have that T is δ-averaged in $\mathbb{R}^N \times \mathbb{R}^S$ equipped with the P-scalar for $\delta \in (0,1)$.

Notice that :

$$\begin{aligned}\bar{z} \in \text{Fix}\,(T) &\iff \bar{z} = T(\bar{z}) = (I+P^{-1}A)^{-1} \circ (I-P^{-1}B)(\bar{z}) \\ &\iff (I+P^{-1}A)(\bar{z}) = (I-P^{-1}B)(\bar{z}) \\ &\iff A(\bar{z}) + B(\bar{z}) = 0\end{aligned}$$

So by the proposition A.8, Fix $(T) \neq \emptyset$ and to prove the theorem 3.2, it is enough to show that z_k converge to a fixed point of T. Indeed T is δ-averaged in $\mathbb{R}^N \times \mathbb{R}^S$ equipped with the P-scalar product, so we can conclude by proposition 2.2. □

A.4 Proof of Theorem 5.2

In our case, we want a Lipschitz constant for $T = Out \circ T_{K-1} \circ ... \circ T_0 = Out \circ (P^{(1)}_{K-1} \circ V^{(1)}_{K-1} \circ P^{(0)}_{K-1} \circ V^{(0)}_{K-1}) \circ ... \circ (P^{(1)}_0 \circ V^{(1)}_0 \circ P^{(0)}_0 \circ V^{(0)}_0)$. Hence we define $(W_{2K}, ..., W_0) = (Out, V^{(1)}_{K-1}, V^{(0)}_{K-1}, ..., V^{(1)}_0, V^{(0)}_0)$ and with the theorem 5.1, we understand that it suffice to find an estimate of $\|W_n \circ ... \circ W_i\|$, for all $i \leqslant n$ in $\{0, ..., 2K\}$. And then by defining $\theta_{-1} = 1$, and

$$\theta_k = \sum_{i=0}^{k} \theta_{i-1} \|W_k \circ ... \circ W_i\|$$

we find that $\frac{\theta_{2K}}{2^{2K-1}}$ is a Lipschitz constant of our neural network.

For all $n \geqslant i$ in $[\![0, K-1]\!]$, $k \in [\![0, K-1]\!]$ we define :

$$M_k = V^{(1)}_k V^{(0)}_k, \qquad M_{n,i} = M_n M_{n-1} ... M_i, \tag{A.6}$$

And for all $i \in [\![0, K]\!]$, $M_{i-1,i} = I_{N+S}$. We have a first result :

Lemma A.13 *For all $i \in [\![1, K-1]\!]$, $n \in [\![i, K-1]\!]$, we can write :*

$$M_{n,i} = \begin{pmatrix} A_{n,i} & B_{n,i}L^* \\ LC_{n,i} & I_S + LD_{n,i}L^* \end{pmatrix} \tag{A.7}$$

Where $A_{n,i}$, $B_{n,i}$, $C_{n,i}$ and $D_{n,i}$ are defined by the induction formulas :

$$\begin{cases} A_{i,i} = & I_N - \tau_i H^* H, \\ B_{i,i} = & -\tau_i I_N, \\ C_{i,i} = & \sigma_i (I_N - 2\tau_i H^* H), \\ D_{i,i} = & 2\tau_i \sigma_i I_N \end{cases} \tag{A.8}$$

$$\begin{cases} \begin{pmatrix} A_{n+1,i} \\ C_{n+1,i} \end{pmatrix} = \begin{pmatrix} I_N - \tau_{n+1} H^* H & -\tau_{n+1} L^* L \\ \sigma_{n+1}(I_N - 2\tau_{n+1} H^* H) & I_N - 2\tau_{n+1}\sigma_{n+1} L^* L \end{pmatrix} \begin{pmatrix} A_{n,i} \\ C_{n,i} \end{pmatrix} \\ \begin{pmatrix} B_{n+1,i} \\ D_{n+1,i} \end{pmatrix} = \begin{pmatrix} I_N - \tau_{n+1} H^* H & -\tau_{n+1} L^* L \\ \sigma_{n+1}(I_N - 2\tau_{n+1} H^* H) & I_N - 2\tau_{n+1}\sigma_{n+1} L^* L \end{pmatrix} \begin{pmatrix} B_{n,i} \\ D_{n,i} \end{pmatrix} - \tau_{n+1} \begin{pmatrix} I_N \\ 2\sigma_{n+1} I_N \end{pmatrix} \end{cases} \tag{A.9}$$

Remark A.14 We expend the lemma to the $M_{i-1,i} = I_{N+S}$, for all $i \in [\![0, K]\!]$, by defining :

$$A_{i-1,i} = I_N, \qquad B_{i-1,i} = C_{i-1,i} = D_{i-1,i} = 0$$

It is like a convention to simplify the formulas after.

Proof. For the cases where $n = i - 1$ are just the definition. Hence we suppose now $n \geqslant i$. We prove this result by induction. We start with the base case : as $M_{i,i} = M_i$, then with the definition (A.6), we compute :

$$M_i = V_i^{(1)} V_i^{(0)} = \begin{pmatrix} I_N - \tau_i H^*H & -\tau_i L^* \\ \sigma_i L(I_N - 2\tau_i H^*H) & I_S - 2\tau_i\sigma_i LL^* \end{pmatrix}$$

so we obtain (A.8). Then for the inductive step we suppose

$$M_{n,i} = \begin{pmatrix} A_{n,i} & B_{n,i}L^* \\ LC_{n,i} & I_S + LD_{n,i}L^* \end{pmatrix}$$

and we compute $M_{n+1,i} = M_{n+1}M_{n,i}$. We find :

$$\begin{aligned} M_{n+1,i} &= \begin{pmatrix} I_N - \tau_{n+1} H^*H & -\tau_{n+1} L^* \\ \sigma_{n+1} L(I_N - 2\tau_{n+1} H^*H) & I_S - 2\tau_{n+1}\sigma_{n+1} LL^* \end{pmatrix} \begin{pmatrix} A_{n,i} & B_{n,i}L^* \\ LC_{n,i} & I_S + LD_{n,i}L^* \end{pmatrix} \\ &=: \begin{pmatrix} A_{n+1,i} & B_{n+1,i}L^* \\ LC_{n+1,i} & I_S + LD_{n+1,i}L^* \end{pmatrix} \end{aligned}$$

Where :

$$\begin{aligned} A_{n+1,i} &= (I_N - \tau_{n+1}H^*H)A_{n,i} - \tau_{n+1}L^*LC_{n,i} \\ B_{n+1,i} &= (I_N - \tau_{n+1}H^*H)B_{n,i} - \tau_{n+1}(I_N + L^*LD_{n,i}) \\ C_{n+1,i} &= \sigma_{n+1}(I_N - 2\tau_{n+1}H^*H)A_{n,i} + (I_N - 2\tau_{n+1}\sigma_{n+1}L^*L)C_{n,i} \\ D_{n+1,i} &= \sigma_{n+1}(I_N - 2\tau_{n+1}H^*H)B_{n,i} - 2\tau_{n+1}\sigma_{n+1}I_N + (I_N - 2\tau_{n+1}\sigma_{n+1}L^*L)D_{n,i} \end{aligned}$$

Hence we can rewrite the definition of $A_{n+1,i}$, $B_{n+1,i}$, $C_{n+1,i}$ and $D_{n+1,i}$ as in (A.9) :

$$\begin{cases} \begin{pmatrix} A_{n+1,i} \\ C_{n+1,i} \end{pmatrix} = \begin{pmatrix} I_N - \tau_{n+1}H^*H & -\tau_{n+1}L^*L \\ \sigma_{n+1}(I_N - 2\tau_{n+1}H^*H) & I_N - 2\tau_{n+1}\sigma_{n+1}L^*L \end{pmatrix} \begin{pmatrix} A_{n,i} \\ C_{n,i} \end{pmatrix} \\ \begin{pmatrix} B_{n+1,i} \\ D_{n+1,i} \end{pmatrix} = \begin{pmatrix} I_N - \tau_{n+1}H^*H & -\tau_{n+1}L^*L \\ \sigma_{n+1}(I_N - 2\tau_{n+1}H^*H) & I_N - 2\tau_{n+1}\sigma_{n+1}L^*L \end{pmatrix} \begin{pmatrix} B_{n,i} \\ D_{n,i} \end{pmatrix} - \tau_{n+1} \begin{pmatrix} I_N \\ 2\sigma_{n+1}I_N \end{pmatrix} \end{cases}$$

□

Corollary A.15 *We can now deduce a formula for $M_{n,0}$, for all $n \in [\![0, K-1]\!]$:*

$$M_{n,0} = \begin{pmatrix} A_{n,0} \\ LC_{n,0} \end{pmatrix} \tag{A.10}$$

Where :

$$A_{n,0} = A_{n,1}\big(I_N - \tau_0(H^*H + L^*L)\big) + B_{n,1}L^*L\big((\sigma_0 + 1)I_N - 2\sigma_0\tau_0(H^*H + L^*L)\big) \tag{A.11}$$

$$C_{n,0} = C_{n,1}\big(I_N - \tau_0(H^*H + L^*L)\big) + (I_N + D_{n,1}L^*L)\big((\sigma_0 + 1)I_N - 2\sigma_0\tau_0(H^*H + L^*L)\big) \tag{A.12}$$

Proof. First we compute M_0 : $M_0 = \begin{pmatrix} I_N - \tau_0(H^*H + L^*L) \\ L\big((\sigma_0+1)I_N - 2\sigma_0\tau_0(H^*H+L^*L)\big) \end{pmatrix}$, then have just to apply the lemma A.13, and compute $M_{n,0} = M_{n,1} \times M_0$. □

As $H^*H \in \mathcal{M}_N(\mathbb{R})$, $L^*L \in \mathcal{M}_S(\mathbb{R})$ are symmetric matrices, and by the hypothesis (ii), we can diagonalise these matrices in the same orthogonal basis : let $(h_p)_{p\in[\![1,N]\!]}$ and $(l_p)_{p\in[\![1,N]\!]}$ the eigenvalues of H^*H and L^*L respectively. We define $(v_p)_{p\in[\![1,N]\!]}$ an orthogonal basis of $\mathbb{R}^N$ composed of eigenvectors of both H^*H and L^*L :

$$H^*Hv_p = h_pv_p \qquad\qquad L^*Lv_p = l_pv_p$$

Then using the lemma A.13 and its corollary A.15, we can define for all $i \in [\![1, K-1]\!]$, $n \in [\![i, K-1]\!]$ (and also for the $n = i-1$, where $i \in [\![0, K]\!]$) : $A_{n,i}$, $B_{n,i}$, $C_{n,i}$ and $D_{n,i}$, and for all $n \in [\![0, K-1]\!]$, $A_{n,0}$, $C_{n,0}$. We have seen that the matrices $A_{n,i}$, $B_{n,i}$, $C_{n,i}$ and $D_{n,i}$ are multinomials of two variables (in $\mathbb{R}[X,Y]$), evaluated in H^*H and L^*L. Hence they are symmetric (H^*H and L^*L commute), and diagonalisable in the basis $(v_p)_{p\in[\![1,N]\!]}$.

For all $i \in [\![1, K-1]\!]$, $n \in [\![i, K-1]\!]$ (or $n = i-1$ for $i \in [\![0, K]\!]$), for all $p \in [\![1, N]\!]$ we denote $a_{n,i,p}$, $b_{n,i,p}$, $c_{n,i,p}$ and $d_{n,i,p}$ the eigenvalues of $A_{n,i}$, $B_{n,i}$, $C_{n,i}$ and $D_{n,i}$, and for all $n \in [\![0, K-1]\!]$, we denote $a_{n,0,p}$ and $c_{n,0,p}$ the eigenvalues of $A_{n,0}$ and $C_{n,0}$ associate to the eigenvectors v_p. The next result gives an inductive rule to compute these eigenvalues.

Lemma A.16 *For all $p \in [\![1, N]\!]$, for $i \in [\![0, K]\!]$,*

$$a_{i-1,i,p} = 1 \qquad\qquad b_{i-1,i,p} = c_{i-1,i,p} = d_{i-1,i,p} = 0$$

For $i \in [\![1, K-1]\!]$:

$$\begin{cases} a_{i,i,p} = & 1 - \tau_i h_p, \\ b_{i,i,p} = & -\tau_i, \\ c_{i,i,p} = & \sigma_i(1 - 2\tau_i h_p), \\ d_{i,i,p} = & 2\tau_i\sigma_i \end{cases}$$

If $n \in [\![i, K-2]\!]$ we can compute $a_{n,i,p}$, $b_{n,i,p}$, $c_{n,i,p}$ and $d_{n,i,p}$ using this inductive rule :

$$\begin{cases} \begin{pmatrix} a_{n+1,i,p} \\ c_{n+1,i,p} \end{pmatrix} = \begin{pmatrix} 1 - \tau_{n+1}h_p & -\tau_{n+1}l_p \\ \sigma_{n+1}(1 - 2\tau_{n+1}h_p) & 1 - 2\tau_{n+1}\sigma_{n+1}l_p \end{pmatrix} \begin{pmatrix} a_{n,i,p} \\ c_{n,i,p} \end{pmatrix} \\ \begin{pmatrix} b_{n+1,i,p} \\ d_{n+1,i,p} \end{pmatrix} = \begin{pmatrix} 1 - \tau_{n+1}h_p & -\tau_{n+1}l_p \\ \sigma_{n+1}(1 - 2\tau_{n+1}h_p) & 1 - 2\tau_{n+1}\sigma_{n+1}l_p \end{pmatrix} \begin{pmatrix} b_{n,i} \\ d_{n,i} \end{pmatrix} - \tau_{n+1} \begin{pmatrix} 1 \\ 2\sigma_{n+1} \end{pmatrix} \end{cases}$$

for all $n \in [\![0, K-1]\!]$,

$$\begin{cases} a_{n,0,p} = & a_{n,1,p}\big(1 - \tau_0(h_p + l_p)\big) + b_{n,1,p}l_p\big((\sigma_0+1) - 2\sigma_0\tau_0(h_p+l_p)\big) \\ c_{n,0,p} = & c_{n,1,p}\big(1 - \tau_0(h_p + l_p)\big) + (1 + d_{n,1,p}l_p)\big((\sigma_0+1) - 2\sigma_0\tau_0(h_p+l_p)\big) \end{cases} \tag{A.13}$$

Proof. It is a clear consequence of the lemma A.13 and corollary A.15. □

Now we define some matrices we will use after. For all $i \in [\![1, K-1]\!]$, $n \in [\![i, K-1]\!]$ (or $n = i-1$, for $i \in [\![0, K]\!]$), for all $p \in [\![1, N]\!]$, we define the 2×2 symmetric matrices :

$$\mathcal{A}^p_{n,i} = \begin{pmatrix} a^2_{n,i,p} + c^2_{n,i,p} l_p & a_{n,i,1} b_{n,i,p} + c_{n,i,p}(1 + d_{n,i,p} l_p) \\ a_{n,i,1} b_{n,i,p} + c_{n,i,p}(1 + d_{n,i,p} l_p) & d_{n,i,p}(d_{n,i,p} l_p + 2) + b^2_{n,i,p} \end{pmatrix} \tag{A.14}$$

$$\mathcal{B}^p_{n,i} = \begin{pmatrix} \mathcal{B}^{(1,1)}_{n,i,p} & \mathcal{B}^{(1,2)}_{n,i,p} \\ \mathcal{B}^{(1,2)}_{n,i,p} & \mathcal{B}^{(2,2)}_{n,i,p} \end{pmatrix} \qquad \mathcal{C}^p_{n,i} = \begin{pmatrix} \mathcal{C}^{(1,1)}_{n,i,p} & \mathcal{C}^{(1,2)}_{n,i,p} \\ \mathcal{C}^{(1,2)}_{n,i,p} & \mathcal{C}^{(2,2)}_{n,i,p} \end{pmatrix} \tag{A.15}$$

And the 3×3 symmetric matrices:

$$\mathcal{D}^p_{n,i} = \begin{pmatrix} \mathcal{D}^{(1,1)}_{n,i,p} & \mathcal{D}^{(1,2)}_{n,i,p} & \mathcal{D}^{(1,3)}_{n,i,p} \\ \mathcal{D}^{(1,2)}_{n,i,p} & \mathcal{D}^{(2,2)}_{n,i,p} & \mathcal{D}^{(2,3)}_{n,i,p} \\ \mathcal{D}^{(1,3)}_{n,i,p} & \mathcal{D}^{(2,3)}_{n,i,p} & \mathcal{D}^{(3,3)}_{n,i,p} \end{pmatrix} \tag{A.16}$$

Where :

$$\begin{aligned}
\mathcal{B}^{(1,1)}_{n,i,p} &= (a_{n,i,p} + 2\sigma_{i-1} b_{n,i,p} l_p)^2 + (\sigma_{i-1} b_{n,i,p} l_p)^2 + b^2_{n,i,p} l_p \\
\mathcal{B}^{(1,2)}_{n,i,p} &= (a_{n,i,p} + 2\sigma_{i-1} b_{n,i,p} l_p)(c_{n,i,p} + 2\sigma_{i-1}(1 + d_{n,i,p} l_p)) + \sigma^2_{i-1} b_{n,i,p} l_p (1 + d_{n,i,p} l_p) \\
&\quad + b_{n,i,p}(1 + l_p d_{n,i,p}) \\
\mathcal{B}^{(2,2)}_{n,i,p} &= (c_{n,i,p} + 2\sigma_{i-1}(1 + d_{n,i,p} l_p))^2 + \sigma^2_{i-1}(1 + d_{n,i,p} l_p)^2 + 2d_{n,i,p} + d^2_{n,i,p} l_p
\end{aligned}$$

$$\begin{aligned}
\mathcal{C}^{(1,1)}_{n,i,p} &= ((1 - \tau_{n+1} h_p) a_{n,i,p} - \tau_{n+1} l_p c_{n,i,p})^2 + a^2_{n,i,p} \\
\mathcal{C}^{(1,2)}_{n,i,p} &= ((1 - \tau_{n+1} h_p) a_{n,i,p} - \tau_{n+1} l_p c_{n,i,p})((1 - \tau_{n+1} h_p) b_{n,i,p} - \tau_{n+1}(1 + l_p d_{n,i,p})) \\
\mathcal{C}^{(2,2)}_{n,i,p} &= ((1 - \tau_{n+1} h_p) b_{n,i,p} - \tau_{n+1}(1 + l_p d_{n,i,p}))^2 + 2d_{n,i,p} + d^2_{n,i,p} l_p
\end{aligned}$$

$$
\begin{aligned}
\mathcal{D}^{(1,1)}_{n,i,p} &= \Big((1-\tau_{n+1}h_p)a_{n,i,p} - \tau_{n+1}l_p c_{n,i,p} + 2\sigma_{i-1}\big((1-\tau_{n+1}h_p)b_{n,i,p} - \tau_{n+1}(1+l_p d_{n,i,p})\big)l_p\Big)^2 \\
&\quad + a^2_{n,i,p} + 4\sigma^2_{i-1}l_p(1+d_{n,i,p}l_p)^2 \\
\mathcal{D}^{(1,2)}_{n,i,p} &= -\sigma_{i-1}\Big((1-\tau_{n+1}h_p)a_{n,i,p} - \tau_{n+1}l_p c_{n,i,p} + 2\sigma_{i-1}\big((1-\tau_{n+1}h_p)b_{n,i,p} - \tau_{n+1}(1+l_p d_{n,i,p})\big)l_p\Big) \\
&\quad \times \Big((1-\tau_{n+1}h_p)b_{n,i,p} - \tau_{n+1}(1+l_p d_{n,i,p})\Big)l_p - 2\sigma^2_{i-1}(1+d_{n,i,p}l_p)^2 l_p \\
\mathcal{D}^{(1,3)}_{n,i,p} &= \Big((1-\tau_{n+1}h_p)a_{n,i,p} - \tau_{n+1}l_p c_{n,i,p} + 2\sigma_{i-1}\big((1-\tau_{n+1}h_p)b_{n,i,p} - \tau_{n+1}(1+l_p d_{n,i,p})\big)l_p\Big) \\
&\quad \times \Big((1-\tau_{n+1}h_p)b_{n,i,p} - \tau_{n+1}(1+l_p d_{n,i,p})\Big) + 2\sigma_{i-1}(1+l_p d_{n,i,p})^2 \\
\mathcal{D}^{(2,2)}_{n,i,p} &= \sigma^2_{i-1}\Big((1-\tau_{n+1}h_p)b_{n,i,p} - \tau_{n+1}(1+l_p d_{n,i,p})\Big)^2 l_p^2 + \sigma^2_{i-1}l_p(1+d_{n,i,p}l_p)^2 \\
\mathcal{D}^{(2,3)}_{n,i,p} &= -\sigma_{i-1}l_p\Big((1-\tau_{n+1}h_p)b_{n,i,p} - \tau_{n+1}(1+l_p d_{n,i,p})\Big)\Big((1-\tau_{n+1}h_p)b_{n,i,p} - \tau_{n+1}(1+l_p d_{n,i,p})\Big) \\
&\quad - \sigma_{i-1}(1+l_p d_{n,i,p})^2 \\
\mathcal{D}^{(3,3)}_{n,i,p} &= \Big((1-\tau_{n+1}h_p)b_{n,i,p} - \tau_{n+1}(1+l_p d_{n,i,p})\Big)^2 + d_{n,i,p}(2+l_p d_{n,i,p})
\end{aligned}
$$

Then we define the $2N \times 2N$ symmetric block matrices :

$$
\mathcal{A}_{n,i} = \begin{pmatrix} \mathcal{A}^1_{n,i} & & (0) \\ & \ddots & \\ (0) & & \mathcal{A}^N_{n,i} \end{pmatrix} \quad \mathcal{B}_{n,i} = \begin{pmatrix} \mathcal{B}^1_{n,i} & & (0) \\ & \ddots & \\ (0) & & \mathcal{B}^N_{n,i} \end{pmatrix} \quad \mathcal{C}_{n,i} = \begin{pmatrix} \mathcal{C}^1_{n,i} & & (0) \\ & \ddots & \\ (0) & & \mathcal{C}^N_{n,i} \end{pmatrix} \tag{A.17}
$$

and the $3N \times 3N$ symmetric block matrix :

$$
\mathcal{D}_{n,i} = \begin{pmatrix} \mathcal{D}^1_{n,i} & & (0) \\ & \ddots & \\ (0) & & \mathcal{D}^N_{n,i} \end{pmatrix} \tag{A.18}
$$

We can easily compute their norms using the lemma A.7.

As said before we have four different cases for the value of $\|M_{n,i}\|$ and the next proposition give an approximation of this value in each case.

The next result is inspired by [1] for some trick of calculation but our case of study is more complex so in the proof of this proposition we develop some new ideas. In particular, we can apply an existing result, we necessary have to make all the calculations.

Proposition A.17 *There are several cases for the value of $\|W_n \dots W_i\|$, we will give an approximation of all of them.*

(i) *Let* $i \in [\![1, K-1]\!]$, $n \in [\![i, K-1]\!]$, *we have :*

$$\|W_{2n+1} \dots W_{2i}\| = \|M_{n,i}\| \leqslant \max\{\sqrt{1 + \|L^*L\|\|\mathcal{A}_{n,i}\|}, \sqrt{\|\mathcal{A}_{n,i}\|}\}$$

Let $n \in [\![0, K-1]\!]$, *we have :*

$$\|W_{2n+1} \dots W_0\| = \|M_{n,0}\| = \max_{p \in [\![1,N]\!]} \sqrt{a_{n,0,p}^2 + c_{n,0,p}^2 l_p}$$

Let $i \in [\![1, K]\!]$, *we have :*

$$\|Out \times W_{2K-1} \dots W_{2i}\| = \|Out M_{K-1,i}\| = \max_{p \in [\![1,N]\!]} \sqrt{a_{K-1,i,p}^2 + b_{K-1,i,p}^2 l_p}$$

$$\|Out \times W_{2K-1} \dots W_0\| = \|Out M_{K-1,0}\| = \max_{p \in [\![1,N]\!]} |a_{K-1,0,p}|$$

(ii) *Let* $i \in [\![1, K-1]\!]$, $n \in [\![i-1, K-1]\!]$, *we have :*

$$\|W_{2n+1} \dots W_{2i-1}\| = \|M_{n,i} V_{i-1}^{(1)}\| \leqslant \max\{\sqrt{1 + \|L^*L\|\|\mathcal{B}_{n,i}\|}, \sqrt{\|\mathcal{B}_{n,i}\|}\}$$

For $i \in [\![1, K]\!]$,

$$\begin{aligned}\|Out \times W_{2K-1} \dots W_{2i-1}\| &= \|Out \times M_{K-1,i} V_{i-1}^{(1)}\| \\ &= \max_{p \in [\![1,N]\!]} \sqrt{(a_{K-1,i,p} + 2\sigma_{i-1} b_{K-1,i,p} l_p)^2 + \sigma_{i-1}^2 b_{K-1,i,p}^2 l_p^2 + b_{K-1,i,p}^2 l_p}\end{aligned}$$

(iii) *Let* $i \in [\![1, K-1]\!]$, $n \in [\![i-1, K-2]\!]$, *we have :*

$$\|W_{2(n+1)} \dots W_{2i}\|^2 = \|V_{n+1}^{(0)} M_{n,i}\|^2 \leqslant \max\{1 + \|L^*L\|\|\mathcal{C}_{n,i}\|, \|\mathcal{C}_{n,i}\|\}$$

for $n \in [\![0, K-2]\!]$

$$\begin{aligned}\|W_{2(n+1)} \dots W_0\| &= \|V_{n+1}^{(0)} M_{n,0}\| \\ &= \max_{p \in [\![1,N]\!]} \sqrt{\big((1 - \tau_{n+1} h_p) a_{n,0,p} - \tau_{n+1} l_p c_{n,0,p}\big)^2 + a_{n,0,p}^2 + c_{n,0,p}^2 l_p} \\ \|W_0\| = \|V_0^{(0)}\| &= \sqrt{\max_{p \in [\![1,N]\!]} \big(1 - \tau_0 (h_p + l_p)\big)^2 + 1 + l_p}\end{aligned}$$

(iv) *Let* $i \in [\![1, K-1]\!]$, $n \in [\![i-1, K-2]\!]$, *we have :*

$$\|W_{2(n+1)} \dots W_{2i-1}\|^2 = \|V_{n+1}^{(0)} M_{n,i} V_{i-1}^{(1)}\|^2 \leqslant \max\{1 + \|L^*L\|\|\mathcal{D}_{n,i}\|, \|\mathcal{D}_{n,i}\|\}$$

As the proof is very long, I choose to organise it in subsections, on per cases. But before the proof of this proposition, we will deduce the theorem 5.2 :

To compute the values of $a_{n,i,p}, b_{n,i,p}, c_{n,i,p}, d_{n,i,p}$, we need $O(N \times K^2)$ by lemma A.16. Then to compute the norms of $\mathcal{A}_{n,i}$, $\mathcal{B}_{n,i}$, $\mathcal{C}_{n,i}$ and $\mathcal{D}_{n,i}$ by the lemma A.7, we need $O(N)$ operations for each. And for the $max_{p \in [\![1,N]\!]}$, we need also $O(N)$ operations. Then a total of $O(N \times K^2)$ to compute all of the upper bounds given in proposition A.17. Hence finally, by replacing the $\|W_n \dots W_i\|$ in the theorem 5.1, by those upper bounds, we can compute θ_{2K} in $O(K^2)$. We can conclude that we only need $O(N \times K^2)$ operations for the calculation of the Lipschitz constant.

A.4.1 Proof of (i)

Using the lemma A.13, we write :

$$M_{n,i} = \begin{pmatrix} A_{n,i} & B_{n,i}L^* \\ LC_{n,i} & I_S + LD_{n,i}L^* \end{pmatrix}$$

Let $x \in \mathbb{R}^N$, $u \in \mathbb{R}^S$,

$$\begin{aligned}
\|M_{n,i}\begin{pmatrix} x \\ u \end{pmatrix}\|^2 &= \|\begin{pmatrix} A_{n,i}x + B_{n,i}L^*u \\ LC_{n,i}x + (I_S + LD_{n,i}L^*)u \end{pmatrix}\|^2 \\
&= \|A_{n,i}x + B_{n,i}L^*u\|^2 + \|LC_{n,i}x + (I_S + LD_{n,i}L^*)u\|^2 \\
&= \|A_{n,i}x\|^2 + \|B_{n,i}L^*u\|^2 + \|LC_{n,i}x\|^2 + \|(I_S + LD_{n,i}L^*)u\|^2 \\
&\quad + 2\langle A_{n,i}x \mid B_{n,i}L^*u\rangle + 2\langle LC_{n,i}x \mid (I_S + LD_{n,i}L^*)u\rangle \\
&= \|A_{n,i}x\|^2 + \|B_{n,i}L^*u\|^2 + \|LC_{n,i}x\|^2 + \|u\|^2 + \|LD_{n,i}L^*u\|^2 \\
&\quad + 2\langle u \mid LD_{n,i}L^*u\rangle + 2\langle A_{n,i}x \mid B_{n,i}L^*u\rangle + 2\langle LC_{n,i}x \mid u + LD_{n,i}L^*u\rangle \\
&= \|A_{n,i}x\|^2 + \|B_{n,i}L^*(u)\|^2 + \langle x \mid C_{n,i}L^*LC_{n,i}x\rangle + \|u\|^2 + \langle L^*(u) \mid D_{n,i}L^*LD_{n,i}L^*(u)\rangle \qquad \text{(A.19)} \\
&\quad + 2\langle L^*(u) \mid D_{n,i}L^*(u)\rangle + 2\langle A_{n,i}x \mid B_{n,i}L^*u\rangle + 2\langle C_{n,i}x \mid L^*(u) + L^*LD_{n,i}L^*(u)\rangle
\end{aligned}$$

In the next, we will write : $x = \sum_{p=1}^{N} x_p v_p$, $L^*(u) = \sum_{p=1}^{N} x'_p v_p$. As in A.16, we denote $a_{n,i,p}$, $b_{n,i,p}$, $c_{n,i,p}$, $d_{n,i,p}$ and l_p the eigenvalues of $A_{n,i}$, $B_{n,i}$, $C_{n,i}$, $D_{n,i}$ and L^*L associate to the eigenvectors v_p.

Hence by (A.19) :

$$\begin{aligned}
\|M_{n,i}\begin{pmatrix}x\\u\end{pmatrix}\|^2 &= \sum_{p=1}^N x_p^2 a_{n,i,p}^2 + \sum_{p=1}^N (x_p')^2 b_{n,i,p}^2 + \sum_{p=1}^N x_p^2 c_{n,i,p}^2 l_p + \|u\|^2 + \sum_{p=1}^N (x_p')^2 d_{n,i,p}^2 l_p \\
&\quad + 2\sum_{p=1}^N (x_p')^2 d_{n,i,p} + 2\sum_{p=1}^N x_p x_p' a_{n,i,p} b_{n,i,p} + 2\sum_{p=1}^N x_p' x_p c_{n,i,p}(1 + d_{n,i,p} l_p) \\
&= \|u\|^2 + \sum_{p=1}^N \Big[x_p^2 (a_{n,i,p}^2 + c_{n,i,p}^2 l_p) + (x_p')^2 (b_{n,i,p}^2 + d_{n,i,p}(d_{n,i,p} l_p + 2)) \\
&\quad + 2 x_p x_p' (a_{n,i,p} b_{n,i,p} + c_{n,i,p}(1 + d_{n,i,p} l_p)) \Big] \\
&= \|u\|^2 + \sum_{p=1}^N \left\langle \mathcal{A}_{n,i}^p \begin{pmatrix}x_p\\x_p'\end{pmatrix} \middle| \begin{pmatrix}x_p\\x_p'\end{pmatrix} \right\rangle
\end{aligned}$$

Where $\mathcal{A}_{n,i}^p$ is as in (A.14) :

$$\mathcal{A}_{n,i}^p = \begin{pmatrix} a_{n,i,p}^2 + c_{n,i,p}^2 l_p & a_{n,i,1} b_{n,i,p} + c_{n,i,p}(1 + d_{n,i,p} l_p) \\ a_{n,i,1} b_{n,i,p} + c_{n,i,p}(1 + d_{n,i,p} l_p) & d_{n,i,p}(d_{n,i,p} l_p + 2) + b_{n,i,p}^2 \end{pmatrix}$$

We define the new variable $X = \begin{pmatrix} x_1 \\ x_1' \\ \vdots \\ x_N \\ x_N' \end{pmatrix} \in \mathbb{R}^{N+N}$. and we have :

$$\|X\|^2 = \|x\|^2 + \|L^*(u)\|^2 \leqslant \|x\|^2 + \|L^*\|^2 \|u\|^2 \tag{A.20}$$

So using Cauchy-Schwarz, the definition of the operator norm and (A.20) we obtain :

$$\begin{aligned}
\|M_{n,i}\|^2 = \sup_{\|x\|^2+\|u\|^2=1} \|M_{n,i}\begin{pmatrix}x\\u\end{pmatrix}\|^2 &= \sup_{\|x\|^2+\|u\|^2=1} \|u\|^2 + \sum_{p=1}^N \left\langle \mathcal{A}_{n,i}^p \begin{pmatrix}x_p\\x_p'\end{pmatrix} \middle| \begin{pmatrix}x_p\\x_p'\end{pmatrix} \right\rangle \\
&= \sup_{\|x\|^2+\|u\|^2=1} \|u\|^2 + \left\langle \begin{pmatrix} \mathcal{A}_{n,i}^1 & & (0) \\ & \ddots & \\ (0) & & \mathcal{A}_{n,i}^N \end{pmatrix} X \middle| X \right\rangle \\
&\leqslant \sup_{\|x\|^2+\|u\|^2=1} \|u\|^2 + \|X\|^2 \max \operatorname{Sp}(\mathcal{A}_{n,i}) \\
&\leqslant \sup_{\|x\|^2+\|u\|^2=1} \|u\|^2 + (\|x\|^2 + \|L^*\|^2 \|u\|^2) \|\mathcal{A}_{n,i}\| \\
&\leqslant \sup_{\|u\|^2 \in [0,1]} \|u\|^2 + ((1 - \|u\|^2) + \|L^*\|^2 \|u\|^2) \|\mathcal{A}_{n,i}\| \\
&= \sup_{\|u\|^2 \in [0,1]} \|u\|^2 (1 + \|L^*\|^2 \|\mathcal{A}_{n,i}\| - \|\mathcal{A}_{n,i}\|) + \|\mathcal{A}_{n,i}\|
\end{aligned}$$

Finally there are two cases :

- If $1 + \|L^*\|^2 \|\mathcal{A}_{n,i}\| \geqslant \|\mathcal{A}_{n,i}\|$:
$$\|M_{n,i}\|^2 \leqslant 1 + \|L^*\|^2 \|\mathcal{A}_{n,i}\|$$
- If $1 + \|L^*\|^2 \|\mathcal{A}_{n,i}\| \leqslant \|\mathcal{A}_{n,i}\|$:
$$\|M_{n,i}\|^2 \leqslant \|\mathcal{A}_{n,i}\|$$

We can conclude :
$$\|M_{n,i}\|^2 \leqslant \max\{1 + \|L^*\|^2 \|\mathcal{A}_{n,i}\|, \|\mathcal{A}_{n,i}\|\}$$
Notice that $\|L^*\|^2 = \|L^* L\|$ by the lemma A.6.

Then for $n \in [\![0, K-1]\!]$, we can compute the same way, without approximation here the norm $\|M_{n,0}\|$: let $x \in \mathbb{R}^N$,
$$\begin{aligned}\|M_{n,0}(x)\|^2 &= \sum_{p=1}^{N} a_{n,0,p}^2 x_p^2 + \sum_{p=1}^{N} c_{n,0,p}^2 l_p x_p^2 \\ &= \left\| \begin{pmatrix} \sqrt{a_{n,0,1}^2 + c_{n,0,1}^2 l_1} & & (0) \\ & \ddots & \\ (0) & & \sqrt{a_{n,0,N}^2 + c_{n,0,N}^2 l_N} \end{pmatrix} (x) \right\|^2\end{aligned}$$
Then we can conclude taking the supremum that $\|M_{n,0}(x)\| = max_{p \in [\![1,N]\!]} \sqrt{a_{n,0,p}^2 + c_{n,0,p}^2 l_p}$.

Finally with the Out layer : let $i \in [\![1, K]\!]$,
$$\|(Out M_{K-1,i})^*(x)\|^2 = \left\| \begin{pmatrix} A_{K-1,i} \\ L B_{K-1,i} \end{pmatrix} (x) \right\|^2$$
We have already made a similar compute, we find :
$$\|Out M_{K-1,i}\| = max_{p \in [\![1,N]\!]} \sqrt{a_{K-1,i,p}^2 + b_{K-1,i,p}^2 l_p}$$

The last particular case is $\|Out M_{K-1,0}\|$: $Out M_{K-1,0} = A_{K-1,0}$, hence $\|Out M_{K-1,0}\| = max_{p \in [\![1,N]\!]} |a_{K-1,0,p}|$.

A.4.2 Proof of (ii)

The proof will be less detailed than the proof of the first case because it uses the same ideas. Using lemma A.13, we write :
$$\begin{aligned} M_{n,i} V_{i-1}^{(1)} &= \begin{pmatrix} A_{n,i} & B_{n,i} L^* \\ L C_{n,i} & I_S + L D_{n,i} L^* \end{pmatrix} \begin{pmatrix} I_N & 0 & 0 \\ 2\sigma_{i-1} L & -\sigma_{i-1} L & I_S \end{pmatrix} \\ &= \begin{pmatrix} A_{n,i} + 2\sigma_{i-1} B_{n,i} L^* L & -\sigma_{i-1} B_{n,i} L^* L & B_{n,i} L^* \\ L(C_{n,i} + 2\sigma_{i-1}(I_N + D_{n,i} L^* L)) & -\sigma_{i-1} L(I_N + D_{n,i} L^* L) & I_S + L D_{n,i} L^* \end{pmatrix}\end{aligned}$$

By lemma A.6, we know that the norm of $M_{n,i}V_{i-1}^{(1)}$ is the same than the norm of its transpose.

Let $x, u \in \mathbb{R}^N \times \mathbb{R}^S$,

$$\|(M_{n,i}V_{i-1}^{(1)})^* \begin{pmatrix} x \\ u \end{pmatrix} \|^2 = \| \begin{pmatrix} (A_{n,i} + 2\sigma_{i-1}B_{n,i}L^*L)x + (C_{n,i} + 2\sigma_{i-1}(I_N + D_{n,i}L^*L))L^*u \\ -\sigma_{i-1}B_{n,i}L^*Lx - \sigma_{i-1}(I_N + D_{n,i}L^*L)L^*u \\ LB_{n,i}x + (I_S + LD_{n,i}L^*)u \end{pmatrix} \|^2$$

So using the notation $x = \sum_{p=1}^N x_p v_p$, $L^*(u) = \sum_{p=1}^N x'_p v_p$, we obtain :

$$\begin{aligned}
\|(M_{n,i}V_{i-1}^{(2)})^* \begin{pmatrix} x \\ u \end{pmatrix} \|^2 &= \sum_{p=1}^N (a_{n,i,p} + 2\sigma_{i-1}b_{n,i,p}l_p)^2 x_p^2 + \sum_{p=1}^N (c_{n,i,p} + 2\sigma_{i-1}(1 + d_{n,i,p}l_p))^2 (x'_p)^2 \\
&\quad + 2\sum_{p=1}^N (a_{n,i,p} + 2\sigma_{i-1}b_{n,i,p}l_p)(c_{n,i,p} + 2\sigma_{i-1}(1 + d_{n,i,p}l_p)) x_p x'_p \\
&\quad + \sum_{p=1}^N (\sigma_{i-1}b_{n,i,p}l_p)^2 x_p^2 + \sum_{p=1}^N \sigma_{i-1}^2 (1 + d_{n,i,p}l_p)^2 (x'_p)^2 \\
&\quad + 2\sum_{p=1}^N \sigma_{i-1}^2 b_{n,i,p}l_p(1 + d_{n,i,p}l_p) x_p x'_p \\
&\quad + \sum_{p=1}^N b_{n,i,p}^2 l_p x_p^2 + \|u\|^2 + 2\sum_{p=1}^N d_{n,i,p}(x'_p)^2 + \sum_{p=1}^N d_{n,i,p}^2 l_p (x'_p)^2 \\
&\quad + 2\sum_{p=1}^N b_{n,i,p}(1 + l_p d_{n,i,p}) x_p x'_p \\
&= \|u\|^2 + \sum_{p=1}^N \left\langle \mathcal{B}_{n,i}^p \begin{pmatrix} x_p \\ x'_p \end{pmatrix} \mid \begin{pmatrix} x_p \\ x'_p \end{pmatrix} \right\rangle
\end{aligned}$$

Where $\mathcal{B}_{n,i}^p$ is as in (A.15) : $\begin{pmatrix} \mathcal{B}_{n,i,p}^{(1,1)} & \mathcal{B}_{n,i,p}^{(1,2)} \\ \mathcal{B}_{n,i,p}^{(1,2)} & \mathcal{B}_{n,i,p}^{(2,2)} \end{pmatrix}$ where $\mathcal{B}_{n,i,p}^{(1,1)}$, $\mathcal{B}_{n,i,p}^{(1,2)}$, $\mathcal{B}_{n,i,p}^{(1,2)}$, $\mathcal{B}_{n,i,p}^{(2,2)}$ are :

$$\begin{aligned}
\mathcal{B}_{n,i,p}^{(1,1)} &= (a_{n,i,p} + 2\sigma_{i-1}b_{n,i,p}l_p)^2 + (\sigma_{i-1}b_{n,i,p}l_p)^2 + b_{n,i,p}^2 l_p \\
\mathcal{B}_{n,i,p}^{(1,2)} &= (a_{n,i,p} + 2\sigma_{i-1}b_{n,i,p}l_p)(c_{n,i,p} + 2\sigma_{i-1}(1 + d_{n,i,p}l_p)) + \sigma_{i-1}^2 b_{n,i,p}l_p(1 + d_{n,i,p}l_p) \\
&\quad + b_{n,i,p}(1 + l_p d_{n,i,p}) \\
\mathcal{B}_{n,i,p}^{(2,2)} &= (c_{n,i,p} + 2\sigma_{i-1}(1 + d_{n,i,p}l_p))^2 + \sigma_{i-1}^2(1 + d_{n,i,p}l_p)^2 + 2d_{n,i,p} + d_{n,i,p}^2 l_p
\end{aligned}$$

Now the end of the proof is exactly the same than in the first case, see the section A.4.1.

For the *Out* layer : let $i \in [\![1, K]\!]$,

$$\|(OutM_{K-1,i}V_{i-1}^{(1)})^*(x)\|^2 = \left\| \begin{pmatrix} A_{K-1,i} + 2\sigma_{i-1}B_{K-1,i}L^*L \\ -\sigma_{i-1}B_{K-1,i}L^*L \\ LB_{K-1,i} \end{pmatrix} (x) \right\|^2$$

We have already made a similar compute, we find :

$$\|OutM_{K-1,i}V_{i-1}^{(1)}\| = max_{p\in[\![1,N]\!]}\sqrt{(a_{K-1,i,p}+2\sigma_{i-1}b_{K-1,i,p}l_p)^2+\sigma_{i-1}^2 b_{K-1,i,p}^2 l_p^2 + b_{K-1,i,p}^2 l_p}$$

A.4.3 Proof of (iii)

The proof will be even less detailed than the proof of the second case because it uses again the same ideas. Using lemma A.13, we write :

$$V_{n+1}^{(0)}M_{n,i} = \begin{pmatrix} I_N-\tau_{n+1}H^*H & -\tau_{n+1}L^* \\ I_N & (0) \\ (0) & I_S \end{pmatrix}\begin{pmatrix} A_{n,i} & B_{n,i}L^* \\ LC_{n,i} & I_S+LD_{n,i}L^* \end{pmatrix}$$

$$= \begin{pmatrix} (I_N-\tau_{n+1}H^*H)A_{n,i}-\tau_{n+1}L^*LC_{n,i} & \Big((I_N-\tau_{n+1}H^*H)B_{n,i}-\tau_{n+1}(I_N+L^*LD_{n,i})\Big)L^* \\ A_{n,i} & (0) \\ (0) & I_S+LD_{n,i}L^* \end{pmatrix}$$

Let $x,u\in\mathbb{R}^N\times\mathbb{R}^S$, we write $x=\sum_{p=1}^N x_pv_p$, $L^*(u)=\sum_{p=1}^N x'_pv_p$, then we obtain :

$$\begin{aligned}\|V_{n+1}^{(0)}M_{n,i}\begin{pmatrix}x\\u\end{pmatrix}\|^2 &= \sum_{p=1}^N((1-\tau_{n+1}h_p)a_{n,i,p}-\tau_{n+1}l_pc_{n,i,p})^2x_p^2 \\ &+\sum_{p=1}^N((1-\tau_{n+1}h_p)b_{n,i,p}-\tau_{n+1}(1+l_pd_{n,i,p}))^2(x'_p)^2 \\ &+2\sum_{p=1}^N((1-\tau_{n+1}h_p)a_{n,i,p}-\tau_{n+1}l_pc_{n,i,p})((1-\tau_{n+1}h_p)b_{n,i,p}-\tau_{n+1}(1+l_pd_{n,i,p}))x_px'_p \\ &+\sum_{p=1}^N a_{n,i,p}^2x_p^2+\|u\|^2+2\sum_{p=1}^N d_{n,i,p}(x'_p)^2+\sum_{p=1}^N d_{n,i,p}^2l_p(x'_p)^2 \\ &=\|u\|^2+\sum_{p=1}^N\left\langle \mathcal{C}_{n,i}\begin{pmatrix}x_p\\x'_p\end{pmatrix}\mid\begin{pmatrix}x_p\\x'_p\end{pmatrix}\right\rangle\end{aligned}$$

Where $\mathcal{C}_{n,i}^p$ is as in (A.15) : $\begin{pmatrix}\mathcal{C}_{n,i,p}^{(1,1)} & \mathcal{C}_{n,i,p}^{(1,2)} \\ \mathcal{C}_{n,i,p}^{(1,2)} & \mathcal{C}_{n,i,p}^{(2,2)}\end{pmatrix}$ where $\mathcal{C}_{n,i,p}^{(1,1)}, \mathcal{C}_{n,i,p}^{(1,2)}, \mathcal{C}_{n,i,p}^{(1,2)}, \mathcal{C}_{n,i,p}^{(2,2)}$ are :

$$\begin{aligned}\mathcal{C}_{n,i,p}^{(1,1)} &= ((1-\tau_{n+1}h_p)a_{n,i,p}-\tau_{n+1}l_pc_{n,i,p})^2+a_{n,i,p}^2 \\ \mathcal{C}_{n,i,p}^{(1,2)} &= ((1-\tau_{n+1}h_p)a_{n,i,p}-\tau_{n+1}l_pc_{n,i,p})((1-\tau_{n+1}h_p)b_{n,i,p}-\tau_{n+1}(1+l_pd_{n,i,p})) \\ \mathcal{C}_{n,i,p}^{(2,2)} &= ((1-\tau_{n+1}h_p)b_{n,i,p}-\tau_{n+1}(1+l_pd_{n,i,p}))^2+2d_{n,i,p}+d_{n,i,p}^2l_p\end{aligned}$$

Now the end of the proof is exactly the same than in the first case, see the section A.4.1.

Then for $n \in [\![0, K-2]\!]$, we can compute the same way, without approximation here the norm $\|V_{n+1}^{(0)} M_{n,0}\|$: let $x \in \mathbb{R}^N$,

$$\begin{aligned}
\|V_{n+1}^{(0)} M_{n,0}(x)\|^2 &= \left\| \begin{pmatrix} (I_N - \tau_{n+1} H^* H) A_{n,0} - \tau_{n+1} L^* L C_{n,0} \\ A_{n,0} \\ L C_{n,0} \end{pmatrix} (x) \right\|^2 \\
&= \sum_{p=1}^{N} \Big((1 - \tau_{n+1} h_p) a_{n,0,p} - \tau_{n+1} l_p c_{n,0,p} \Big)^2 x_p^2 + \sum_{p=1}^{N} a_{n,0,p}^2 x_p^2 + \sum_{p=1}^{N} c_{n,0,p}^2 l_p x_p^2 \\
&= \left\| Diag\left(\sqrt{((1 - \tau_{n+1} h_p) a_{n,0,p} - \tau_{n+1} l_p c_{n,0,p})^2 + a_{n,0,p}^2 + c_{n,0,p}^2 l_p} \right) (x) \right\|^2
\end{aligned}$$

Then we can conclude taking the supremum :

$$\|V_{n+1}^{(0)} M_{n,0}\| = max_{p \in [\![1,N]\!]} \sqrt{\left((1 - \tau_{n+1} h_p) a_{n,0,p} - \tau_{n+1} l_p c_{n,0,p}\right)^2 + a_{n,0,p}^2 + c_{n,0,p}^2 l_p}$$

Finally we compute $(V_0^{(0)})^* V_0^{(0)}$,

we obtain $(I_N - \tau_0(H^*H + L^*L))^2 + I_N + L^*L$. Hence we have :

$$\|W_0\| = \|V_0^{(0)}\| = \sqrt{\max_{p \in [\![1,N]\!]} \left| \left(1 - \tau_0(h_p + l_p)\right)^2 + 1 + l_p \right|}$$

A.4.4 Proof of (iv)

The proof uses again the same ideas as A.4.1 so look at this proof for more details. Using lemma A.13, we write :

$$\begin{aligned}
V_{n+1}^{(0)} M_{n,i} V_{i-1}^{(1)} &= \begin{pmatrix} I_N - \tau_{n+1} H^* H & -\tau_{n+1} L^* \\ I_N & (0) \\ (0) & I_S \end{pmatrix} \begin{pmatrix} A_{n,i} & B_{n,i} L^* \\ L C_{n,i} & I_S + L D_{n,i} L^* \end{pmatrix} \begin{pmatrix} I_N & 0 & 0 \\ 2\sigma_{i-1} L & -\sigma_{i-1} L & I_S \end{pmatrix} \\
&= \begin{pmatrix} M^{1,1} & M^{1,2} & M^{1,3} \\ M^{2,1} & M^{2,2} & M^{2,3} \\ M^{3,1} & M^{3,2} & M^{3,3} \end{pmatrix}
\end{aligned}$$

Where :

$$\begin{aligned}
M^{1,1} &= (I_N - \tau_{n+1} H^* H) A_{n,i} - \tau_{n+1} L^* L C_{n,i} \\
&\quad + 2\sigma_{i-1} \Big((I_N - \tau_{n+1} H^* H) B_{n,i} - \tau_{n+1} (I_N + L^* L D_{n,i}) \Big) L^* L, \\
M^{1,2} &= -\sigma_{i-1} \Big((I_N - \tau_{n+1} H^* H) B_{n,i} - \tau_{n+1} (I_N + L^* L D_{n,i}) \Big) L^* L, \\
M^{1,3} &= \Big((I_N - \tau_{n+1} H^* H) B_{n,i} - \tau_{n+1} (I_N + L^* L D_{n,i}) \Big) L^*,
\end{aligned}$$

$$M^{2,1} = A_{n,i}, \qquad M^{2,2} = (0), \qquad M^{2,3} = (0),$$
$$M^{3,1} = 2\sigma_{i-1} L(I_N + D_{n,i}L^*L), \quad M^{3,2} = -\sigma_{i-1}L(I_N + D_{n,i}L^*L), \quad M^{3,3} = I_S + LD_{n,i}L^*.$$

Let $x, x', u \in \mathbb{R}^N \times \mathbb{R}^N \times \mathbb{R}^S$, we write $x = \sum_{p=1}^N x_p v_p$, $x' = \sum_{p=1}^N x'_p v_p$, $L^*(u) = \sum_{p=1}^N x''_p v_p$, then we obtain :

$$\begin{aligned}
&\|V_{n+1}^{(0)} M_{n,i} V_{i-1}^{(1)} \begin{pmatrix} x \\ x' \\ u \end{pmatrix} \|^2 = \\
&\sum_{p=1}^N \Big((1-\tau_{n+1}h_p)a_{n,i,p} - \tau_{n+1}l_p c_{n,i,p} + 2\sigma_{i-1}\big((1-\tau_{n+1}h_p)b_{n,i,p} - \tau_{n+1}(1+l_p d_{n,i,p})\big)l_p\Big)^2 x_p^2 \\
&+ \sum_{p=1}^N \sigma_{i-1}^2 \Big((1-\tau_{n+1}h_p)b_{n,i,p} - \tau_{n+1}(1+l_p d_{n,i,p})\Big)^2 l_p^2 (x'_p)^2 \\
&+ \sum_{p=1}^N \Big((1-\tau_{n+1}h_p)b_{n,i,p} - \tau_{n+1}(1+l_p d_{n,i,p})\Big)^2 (x''_p)^2 \\
&- 2\sum_{p=1}^N \sigma_{i-1}\Big((1-\tau_{n+1}h_p)a_{n,i,p} - \tau_{n+1}l_p c_{n,i,p} + 2\sigma_{i-1}\big((1-\tau_{n+1}h_p)b_{n,i,p} - \tau_{n+1}(1+l_p d_{n,i,p})\big)l_p\Big) \\
&\times \Big((1-\tau_{n+1}h_p)b_{n,i,p} - \tau_{n+1}(1+l_p d_{n,i,p})\Big) l_p x_p x'_p \\
&+ 2\sum_{p=1}^N \Big((1-\tau_{n+1}h_p)a_{n,i,p} - \tau_{n+1}l_p c_{n,i,p} + 2\sigma_{i-1}\big((1-\tau_{n+1}h_p)b_{n,i,p} - \tau_{n+1}(1+l_p d_{n,i,p})\big)l_p\Big) \\
&\times \Big((1-\tau_{n+1}h_p)b_{n,i,p} - \tau_{n+1}(1+l_p d_{n,i,p})\Big) x_p x''_p \\
&- 2\sum_{p=1}^N \sigma_{i-1} l_p \Big((1-\tau_{n+1}h_p)b_{n,i,p} - \tau_{n+1}(1+l_p d_{n,i,p})\Big)\Big((1-\tau_{n+1}h_p)b_{n,i,p} - \tau_{n+1}(1+l_p d_{n,i,p})\Big) x'_p x''_p \\
&+ \sum_{p=1}^N a_{n,i,p}^2 x_p^2 \\
&+ \sum_{p=1}^N 4\sigma_{i-1}^2 l_p (1+d_{n,i,p}l_p)^2 x_p^2 + \sum_{p=1}^N \sigma_{i-1}^2 l_p (1+d_{n,i,p}l_p)^2 (x'_p)^2 \\
&+ \|u\|^2 + \sum_{p=1}^N d_{n,i,p}(2+l_p d_{n,i,p})(x''_p)^2 - 2\sum_{p=1}^N 2\sigma_{i-1}^2 (1+d_{n,i,p}l_p)^2 l_p x_p x'_p \\
&+ 2\sum_{p=1}^N 2\sigma_{i-1}(1+d_{n,i,p}l_p)^2 x_p x''_p - 2\sum_{p=1}^N \sigma_{i-1}(1+d_{n,i,p}l_p)^2 x'_p x''_p
\end{aligned}$$

Hence we have :

$$\|V_{n+1}^{(0)}M_{n,i}V_{i-1}^{(1)}\begin{pmatrix}x\\x'\\u\end{pmatrix}\|^2=\|u\|^2+\sum_{p=1}^{N}\left\langle\mathcal{D}_{n,i}\begin{pmatrix}x_p\\x'_p\\x''_p\end{pmatrix}\mid\begin{pmatrix}x_p\\x'_p\\x''_p\end{pmatrix}\right\rangle$$

Where $\mathcal{D}_{n,i}^{p}$ is as in (A.16) : $\mathcal{D}_{n,i}^{p}=\begin{pmatrix}\mathcal{D}_{n,i,p}^{(1,1)}&\mathcal{D}_{n,i,p}^{(1,2)}&\mathcal{D}_{n,i,p}^{(1,3)}\\\mathcal{D}_{n,i,p}^{(1,2)}&\mathcal{D}_{n,i,p}^{(2,2)}&\mathcal{D}_{n,i,p}^{(2,3)}\\\mathcal{D}_{n,i,p}^{(1,3)}&\mathcal{D}_{n,i,p}^{(2,3)}&\mathcal{D}_{n,i,p}^{(3,3)}\end{pmatrix}$ where the coefficients are :

$$\begin{aligned}
\mathcal{D}_{n,i,p}^{(1,1)}&=\Big((1-\tau_{n+1}h_p)a_{n,i,p}-\tau_{n+1}l_pc_{n,i,p}+2\sigma_{i-1}\big((1-\tau_{n+1}h_p)b_{n,i,p}-\tau_{n+1}(1+l_pd_{n,i,p})\big)l_p\Big)^2\\
&\quad+a_{n,i,p}^2+4\sigma_{i-1}^2l_p(1+d_{n,i,p}l_p)^2\\
\mathcal{D}_{n,i,p}^{(1,2)}&=-\sigma_{i-1}\Big((1-\tau_{n+1}h_p)a_{n,i,p}-\tau_{n+1}l_pc_{n,i,p}+2\sigma_{i-1}\big((1-\tau_{n+1}h_p)b_{n,i,p}-\tau_{n+1}(1+l_pd_{n,i,p})\big)l_p\Big)\\
&\quad\times\Big((1-\tau_{n+1}h_p)b_{n,i,p}-\tau_{n+1}(1+l_pd_{n,i,p})\Big)l_p-2\sigma_{i-1}^2(1+d_{n,i,p}l_p)^2l_p\\
\mathcal{D}_{n,i,p}^{(1,3)}&=\Big((1-\tau_{n+1}h_p)a_{n,i,p}-\tau_{n+1}l_pc_{n,i,p}+2\sigma_{i-1}\big((1-\tau_{n+1}h_p)b_{n,i,p}-\tau_{n+1}(1+l_pd_{n,i,p})\big)l_p\Big)\\
&\quad\times\Big((1-\tau_{n+1}h_p)b_{n,i,p}-\tau_{n+1}(1+l_pd_{n,i,p})\Big)+2\sigma_{i-1}(1+l_pd_{n,i,p})^2\\
\mathcal{D}_{n,i,p}^{(2,2)}&=\sigma_{i-1}^2\Big((1-\tau_{n+1}h_p)b_{n,i,p}-\tau_{n+1}(1+l_pd_{n,i,p})\Big)^2l_p^2+\sigma_{i-1}^2l_p(1+d_{n,i,p}l_p)^2\\
\mathcal{D}_{n,i,p}^{(2,3)}&=-\sigma_{i-1}l_p\Big((1-\tau_{n+1}h_p)b_{n,i,p}-\tau_{n+1}(1+l_pd_{n,i,p})\Big)\Big((1-\tau_{n+1}h_p)b_{n,i,p}-\tau_{n+1}(1+l_pd_{n,i,p})\Big)\\
&\quad-\sigma_{i-1}(1+l_pd_{n,i,p})^2\\
\mathcal{D}_{n,i,p}^{(3,3)}&=\Big((1-\tau_{n+1}h_p)b_{n,i,p}-\tau_{n+1}(1+l_pd_{n,i,p})\Big)^2+d_{n,i,p}(2+l_pd_{n,i,p})
\end{aligned}$$

We define the new variable $X=\begin{pmatrix}x_1\\x'_1\\x''_1\\\vdots\\x_N\\x'_N\\x''_N\end{pmatrix}\in\mathbb{R}^{N+N+N}$. and we have :

$$\|X\|^2=\|x\|^2+\|x'\|^2+\|L^*(u)\|^2\leqslant\|x\|^2+\|x'\|^2+\|L^*\|^2\|u\|^2\tag{A.21}$$

So using Cauchy-Schwarz, the definition of the operator norm and (A.21) we obtain :

$$\begin{aligned}
\|M_{n,i}\|^2 &= \sup_{\|x\|^2+\|x'\|^2+\|u\|^2=1} \|M_{n,i}\begin{pmatrix} x \\ u \end{pmatrix}\|^2 \\
&= \sup_{\|x\|^2+\|x'\|^2+\|u\|^2=1} \|u\|^2 + \sum_{p=1}^{N} \left\langle \mathcal{D}_{n,i}^{p}\begin{pmatrix} x_p \\ x_p' \end{pmatrix} \middle| \begin{pmatrix} x_p \\ x_p' \end{pmatrix} \right\rangle \\
&\leqslant \sup_{\|x\|^2+\|x'\|^2+\|u\|^2=1} \|u\|^2 + \|X\|^2\|\mathcal{D}_{n,i}\| \\
&\leqslant \sup_{\|x\|^2+\|x'\|^2+\|u\|^2=1} \|u\|^2 + (\|x\|^2 + \|x'\|^2 + \|L^*\|^2\|u\|^2)\|\mathcal{D}_{n,i}\|
\end{aligned}$$

Where we denote $\mathcal{D}_{n,i} = \begin{pmatrix} \mathcal{D}_{n,i}^{1} & & (0) \\ & \ddots & \\ (0) & & \mathcal{D}_{n,i}^{N} \end{pmatrix}$ as in (A.18). Hence we have :

$$\begin{aligned}
\sup_{\|x\|^2+\|x'\|^2+\|u\|^2=1} & \|u\|^2 + (\|x\|^2 + \|x'\|^2 + \|L^*\|^2\|u\|^2)\|\mathcal{D}_{n,i}\| \\
&= \sup_{\|u\|^2\in[0,1]} \|u\|^2 + ((1-\|u\|^2) + \|L^*\|^2\|u\|^2)\|\mathcal{D}_{n,i}\| \\
&= \sup_{\|u\|^2\in[0,1]} \|u\|^2(1 + \|L^*\|^2\|\mathcal{D}_{n,i}\| - \|\mathcal{D}_{n,i}\|) + \|\mathcal{D}_{n,i}\|
\end{aligned}$$

Now the end of the proof is exactly the same than in the first case, see the section A.4.1.

B Discussion on the hypothesis (ii) of the Theorem 5.2

First we will characterize all the matrix L verifying the hypothesis (ii) : H^*H and L^*L commute. Then we will give some examples of classical form for L which verify this hypothesis.

B.1 Characterization of the hypothesis

Let us suppose that H^*H and L^*L commute. As H^*H is symmetric, we can define $E_1 = \text{Span}\,(b_1, .., b_{n_1})$, ..., $E_r = \text{Span}\,(b_{n_{r-1}}, .., b_N)$ the eigenspaces of H^*H, they are orthogonal pairwise, and their direct sum is $\mathbb{R}^N$.

$$n_0 = 0 \qquad\qquad n_r = N \qquad\qquad \forall k \in [\![1, r]\!], n_k - n_{k-1} = \dim(E_k)$$

$(b_k)_{k\in[\![1,N]\!]}$ an ortho-diagonalisation basis for H^*H. We define $P = \begin{pmatrix} b_1 & \dots & b_N \end{pmatrix}$ which verify that P^*H^*HP is diagonal. Notice that H^*H and L^*L commutes if and only if the eigenspaces E_k

are fixed by L^*L, so if and only if we can write :

$$P^*L^*LP = \begin{pmatrix} B'_1 & & (0) \\ & \ddots & \\ (0) & & B'_r \end{pmatrix} \tag{B.1}$$

where $B'_k \in \mathcal{M}_{n_k - n_{k-1}}(\mathbb{R})$.

Proposition B.1 *Choose L verifying (ii) is exactly the same as :*

- *choose an orthogonal matrix $U \in \mathcal{O}_S(\mathbb{R})$*
- *choose a sequence $m_1, ..., m_r \in \mathbb{N}$ such that :*

$$\sum_{k=1}^{r} m_k = S$$

- *choose a block matrix :*

$$\begin{pmatrix} B_1 & & (0) \\ & \ddots & \\ (0) & & B_r \end{pmatrix}$$

  such that $B_k \in \mathcal{M}_{m_k, n_k - n_{k-1}}$.

An then define $L = U \begin{pmatrix} B_1 & & (0) \\ & \ddots & \\ (0) & & B_r \end{pmatrix} P^*$.

Proof. We denote $LP =: \begin{pmatrix} L_1 & \dots & L_r \end{pmatrix}$ where for all $k \in [\![1, r]\!]$, $L_k \in \mathcal{M}_{S, n_k - n_{k-1}}(\mathbb{R})$. Then we compute :

$$P^*L^*LP = \begin{pmatrix} L_1^* \\ \vdots \\ L_r^* \end{pmatrix} \begin{pmatrix} L_1 & ... & L_r \end{pmatrix} = \begin{pmatrix} L_1^*L_1 & ... & L_1^*L_r \\ \vdots & \ddots & \vdots \\ L_r^*L_1 & ... & L_r^*L_r \end{pmatrix}$$

By (B.1) we have necessary :

$$\forall k \neq j \in [\![1, r]\!], L_k^*L_j = 0 \tag{B.2}$$

Which can be written : $\forall k \neq j \in [\![1, r]\!]$, the column of L_k and the ones of L_j are in orthogonal subspaces of $\mathbb{R}^S$. So we explicit the column of L_k : $\mathcal{C}_{n_{k-1}+1}, ..., \mathcal{C}_{n_k}$, and we denote $\mathcal{F}_k := \text{Span}\left(\mathcal{C}_{n_{k-1}+1}, ..., \mathcal{C}_{n_k}\right)$. Let m_k be the dimension of $\mathcal{F}_k$. We have for all $k \in [\![1, r]\!]$ an orthonormal basis $u_1^{(k)}, ..., u_{m_k}^{(k)}$ $(m_k \in \mathbb{N})$ of $\mathcal{F}_k$, then if we concatenate them : $u_1^{(1)}, ..., u_{m_r}^{(r)}$ and we recall it $w_1, ..., w_{\sum_{k=1}^r m_k}$, it is an orthonormal family of $\mathbb{R}^S$. We can complete this family to obtain an orthonormal basis of $\mathbb{R}^S$: $w_1, ..., w_S$. We put those vectors into a matrix W :

$$W = \begin{pmatrix} w_1 & \dots & w_S \end{pmatrix}$$

We have : $W^* \begin{pmatrix} L_1 & \dots & L_r \end{pmatrix} = \begin{pmatrix} B_1 & & (0) \\ & \ddots & \\ (0) & & B_r \end{pmatrix}$, where $B_k \in \mathcal{M}_{m_k, n_k - n_{k-1}}(\mathbb{R})$.

Hence we have well :

$$L = W \begin{pmatrix} B_1 & & (0) \\ & \ddots & \\ (0) & & B_r \end{pmatrix} B^*$$

where $W \in \mathcal{O}_S(\mathbb{R})$.

Conversely, if $L = U \begin{pmatrix} B_1 & & (0) \\ & \ddots & \\ (0) & & B_r \end{pmatrix} P^*$ where $U \in \mathcal{O}_S(\mathbb{R})$. Then we compute :

$$\begin{aligned} P^* L^* L P &= \begin{pmatrix} B_1^* & & (0) \\ & \ddots & \\ (0) & & B_r^* \end{pmatrix} U^* U \begin{pmatrix} B_1 & & (0) \\ & \ddots & \\ (0) & & B_r \end{pmatrix} \\ &= \begin{pmatrix} B_1^* & & (0) \\ & \ddots & \\ (0) & & B_r^* \end{pmatrix} \begin{pmatrix} B_1 & & (0) \\ & \ddots & \\ (0) & & B_r \end{pmatrix} = \begin{pmatrix} B_1' & & (0) \\ & \ddots & \\ (0) & & B_r' \end{pmatrix} \end{aligned}$$

where $B_k' \in \mathcal{M}_{n_k - n_{k-1}}(\mathbb{R})$, so by (B.1), $L^* L$ and $H^* H$ commute. □

B.2 Case where H is an inpainting operator

In this case, $H : \mathbb{R}^N \to \mathbb{R}^N$ is of the form :

$$H = \begin{pmatrix} \epsilon_1 & & (0) \\ & \ddots & \\ (0) & & \epsilon_N \end{pmatrix}$$

where $\epsilon_k \in \{0, 1\}$, we denote by $r \in \mathbb{N}$ the number of 1 among the ϵ_k. Hence H is an orthogonal projector, $H^* H = H^2 = H$. Let P the permutation matrix such that

$$PHP^* = \begin{pmatrix} I_r & (0) \\ (0) & (0) \end{pmatrix}$$

Now we apply the proposition B.1, and we obtain that the matrices L which verify the hypothesis (ii) are of the form :

$$U \begin{pmatrix} B_1 & (0) \\ (0) & B_2 \end{pmatrix} P^*$$

with $U \in \mathcal{O}_S(\mathbb{R})$, and $B_1 \in \mathcal{M}_{m,r}(\mathbb{R})$, $B_2 \in \mathcal{M}_{S-m,N-r}(\mathbb{R})$, $m \in [\![0, S]\!]$.

B.3 Case where H is an undersampled Fourier operator

If we denote $x = (x_1, ..., x_N) \in \mathbb{R}^N$ our image. Then the discrete Fourier transform of x is defined by :

$$\forall k \in [\![1, N]\!], [\mathcal{F}(x)]_k := \sum_{l=1}^{N} x_l \exp(-2i\pi \frac{kl}{N})$$

Hence we have :

$$\mathcal{F} = \begin{pmatrix} \exp(\frac{-2i\pi}{N}) & \exp(\frac{-4i\pi}{N}) & \dots & 1 \\ \exp(\frac{-4i\pi}{N}) & \exp(\frac{-8i\pi}{N}) & \dots & 1 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & \dots & 1 \end{pmatrix} = \mathcal{F}^T$$

Then to undersampled this Fourier transform, we fix a subset $\Omega \subset [\![1, N]\!]$ and we define :

$$H : \begin{pmatrix} \mathbb{R}^N & \rightarrow & \mathbb{R}^S \\ x & \mapsto & (X_\omega)_{\omega \in \Omega} \end{pmatrix}$$

We denote $\Omega = \{\omega_1 < ... < \omega_r\}$, $M = (\delta_{j=\omega_i})_{i,j} \in \mathcal{M}_{S,N}(\mathbb{R})$ and $(e_k)_k$ the canonical basis of $\mathbb{R}^N$. So we have : $Me_k = e_l$ if $k = \omega_l$, and $Me_k = 0$ otherwise. Let $k \in [\![1, N]\!]$

We see that :

$$H = M\mathcal{F}$$

Notice that M is an inpainting operator and we have :

$$M^*M = \begin{pmatrix} \epsilon_1 & & (0) \\ & \ddots & \\ (0) & & \epsilon_N \end{pmatrix}$$

where $\epsilon_k = 1$ if $k \in \Omega$, $\epsilon_k = 0$ otherwise.

Then we compute :

$$\begin{aligned}[\mathcal{F}^*\mathcal{F}]_{k,l} &= \sum_{p=1}^{N} \exp(-\frac{2i\pi}{N}kp)\exp(\frac{2i\pi}{N}lp) \\ &= \sum_{p=1}^{N} \exp(\frac{2i\pi}{N}p(l-k)) \quad \text{where } l-k \in [\![-(N-1),(N-1)]\!] \\ &= \begin{cases} 0 & \text{if } l \neq k \\ N & \text{otherwise} \end{cases}\end{aligned}$$

Hence we have :

$$\frac{1}{\sqrt{N}}\mathcal{F} \in \mathcal{O}_N$$

Then we can conclude on the spectral decomposition of H^*H :

$$H^*H = P^* \begin{pmatrix} I_r & (0) \\ (0) & (0) \end{pmatrix} P$$

where $r = |\Omega|$ and $P = \frac{1}{\sqrt{N}}P'\mathcal{F}$, with P' the permutation matrix such that $(P')^* \begin{pmatrix} I_r & (0) \\ (0) & (0) \end{pmatrix} P' = M^*M = \begin{pmatrix} \epsilon_1 & & (0) \\ & \ddots & \\ (0) & & \epsilon_N \end{pmatrix}$.

Hence like in the case of H an inpainting operator : by the proposition B.1, and we obtain the matrices L which verify the hypothesis (ii) are of the form :

$$U \begin{pmatrix} B_1 & (0) \\ (0) & B_2 \end{pmatrix} P^*$$

with $U \in \mathcal{O}_S(\mathbb{R})$, and $B_1 \in \mathcal{M}_{m,r}(\mathbb{R})$, $B_2 \in \mathcal{M}_{S-m,N-r}(\mathbb{R})$, $m \in [\![0, S]\!]$.

B.4 Convolution operator

If we denote $x = (x_{u,v})_{u,v\in[\![1,n]\!]} \in \mathbb{R}^{n\times n}$ our image. And $h = (h_{u,v})_{u,v\in[\![-(m-1)/2,(m-1)/2]\!]} \in \mathbb{R}^{m\times m}$ a matrix called the kernel, we assume that m is odd. Then in the next we will see x as an element of $\mathbb{Z}^2$, indeed we expend by periodicity : $x_{u,v} = x_{[u],[v]}$ where $[u]$ is the rest in the euclidean division of u by n. We can define the convolution of an image x, with a kernel h by :

$$\forall \bar{u}, \bar{v} \in [\![1, n]\!], (h * x)_{\bar{u},\bar{v}} = \sum_{u=-(m-1)/2}^{(m-1)/2} \sum_{u=-(m-1)/2}^{(m-1)/2} h_{u,v} x_{\bar{u}-u,\bar{v}-v}$$

We also define the 2−dimensional Fourier transform for the $n \times n$ images :

$$\forall \bar{u}, \bar{v} \in [\![1, n]\!], \mathcal{F}(x)_{\bar{u},\bar{v}} = \widehat{x}_{\bar{u},\bar{v}} = \sum_{u=1}^{n}\sum_{v=1}^{n} x_{u,v} \exp\left(-2i\pi\left(\frac{u\bar{u}}{n} + \frac{v\bar{v}}{n}\right)\right)$$

It is quite the same definition for $\widehat{h}$, but we implicitly extend it by zero to a $n \times n$ image (or if $m > n$ it extracts a $n \times n$ matrix), hence we define :

$$\forall \bar{u}, \bar{v} \in [\![1, n]\!], \mathcal{F}(h)_{\bar{u},\bar{v}} = \widehat{h}_{\bar{u},\bar{v}} = \sum_{u=-(m-1)/2}^{(m-1)/2}\ \sum_{v=-(m-1)/2}^{(m-1)/2} h_{u,v} \exp\left(-2i\pi\left(\frac{u\bar{u}}{n} + \frac{v\bar{v}}{n}\right)\right)$$

it is not an isomorphism if $m > n$ but no matter here. Then we will denote $\otimes$ the matrix multiplication term by term.

Lemma B.2 *Let x periodic and h a kernel like before. We have the formula : $\mathcal{F}(h * x) = \widehat{h} \otimes \widehat{x}$.*

Proof. Let $\bar{u}, \bar{v} \in [\![1, n]\!]$, we have :

$$\begin{aligned}
\mathcal{F}(h * x)_{\bar{u},\bar{v}} &= \sum_{u=1}^{n}\sum_{v=1}^{n} (h * x)_{u,v} \exp\left(-2i\pi\left(\frac{u\bar{u}}{n} + \frac{v\bar{v}}{n}\right)\right) \\
&= \sum_{u=1}^{n}\sum_{v=1}^{n} \sum_{\tilde{u}=-(m-1)/2}^{(m-1)/2} \sum_{\tilde{v}=-(m-1)/2}^{(m-1)/2} h_{\tilde{u},\tilde{v}} x_{u-\tilde{u},v-\tilde{v}} \exp\left(-2i\pi\left(\frac{u\bar{u}}{n} + \frac{v\bar{v}}{n}\right)\right) \\
&= \sum_{\tilde{u}=-(m-1)/2}^{(m-1)/2}\ \sum_{\tilde{v}=-(m-1)/2}^{(m-1)/2} h_{\tilde{u},\tilde{v}} \exp\left(-2i\pi\left(\frac{\tilde{u}\bar{u}}{n} + \frac{\tilde{u}\bar{v}}{n}\right)\right) \\
&\quad \sum_{u=1}^{n}\sum_{v=1}^{n} x_{u-\tilde{u},v-\tilde{v}} \exp\left(-2i\pi\left(\frac{(u-\tilde{u})\bar{u}}{n} + \frac{(v-\tilde{u})\bar{v}}{n}\right)\right) \\
&= \widehat{h}_{\bar{u},\bar{v}} \times \widehat{x}_{\bar{u},\bar{v}} = (\widehat{h} \otimes \widehat{x})_{\bar{u},\bar{v}}
\end{aligned}$$

□

Here we consider an operator $H : \mathbb{R}^N \to \mathbb{R}^N$ of the form : $H(x) = h * x$, for a fixed vector $h \in \mathbb{R}^{m \times m}$.

Lemma B.3 *We have : $\forall x \in \mathbb{R}^{n \times n}, H^*(x) = \tilde{h} * x$, where $\forall u, v \in [\![-(m-1)/2, (m-1)/2]\!], \tilde{h}_{u,v} = h_{-u,-v}$*

Proof. Let $x, y \in \mathbb{R}^{n\times n}$. We compute $\langle H(x) \mid y\rangle$:

$$\begin{aligned}
\langle H(x) \mid y\rangle &= \sum_{\bar u=1}^{n}\sum_{\bar v=1}^{n}\sum_{u=-(m-1)/2}^{(m-1)/2}\sum_{v=-(m-1)/2}^{(m-1)/2} h_{u,v}x_{\bar u-u,\bar v-v}y_{\bar u,\bar v}\\
&= \sum_{u=-(m-1)/2}^{(m-1)/2}\sum_{v=-(m-1)/2}^{(m-1)/2}\sum_{\bar u=1}^{n}\sum_{\bar v=1}^{n} h_{u,v}x_{\bar u-u,\bar v-v}y_{\bar u,\bar v}\\
&= \sum_{u=-(m-1)/2}^{(m-1)/2}\sum_{v=-(m-1)/2}^{(m-1)/2}\sum_{\bar u=1}^{n}\sum_{\bar v=1}^{n} h_{u,v}x_{\bar u,\bar v}y_{\bar u+u,\bar v+v} \quad \text{by periodicity.}\\
&= \sum_{\bar u=1}^{n}\sum_{\bar v=1}^{n} x_{\bar u,\bar v}\sum_{u=-(m-1)/2}^{(m-1)/2}\sum_{v=-(m-1)/2}^{(m-1)/2} \tilde h_{-u,-v}y_{\bar u+u,\bar v+v}\\
&= \left\langle x \mid \tilde h * y\right\rangle
\end{aligned}$$

□

Now we can compute H^*H, and find its eigenspaces : Let $x \in \mathbb{R}^{n\times n}$, let $\bar u, \bar v \in [\![1, n]\!]$, we have :

$$\begin{aligned}
H^*H(x)_{\bar u,\bar v} &= (\tilde h * (h * x))_{\bar u,\bar v}\\
&= \sum_{\tilde u=-(m-1)/2}^{(m-1)/2}\sum_{\tilde v=-(m-1)/2}^{(m-1)/2} \tilde h_{\tilde u,\tilde v}\sum_{u=-(m-1)/2}^{(m-1)/2}\sum_{v=-(m-1)/2}^{(m-1)/2} h_{u,v}x_{(\bar u-\tilde u)-u,(\bar v-\tilde v)-v}\\
&= \sum_{s_1=-(m-1)}^{(m-1)}\sum_{s_2=-(m-1)}^{(m-1)}\sum_{t_1\in U_{s_1}}\sum_{t_2\in U_{s_2}} \tilde h_{t_1,t_2}h_{s_1-t_1,s_2-t_2}x_{\bar u-s_1,\bar v-s_2}
\end{aligned}$$

where in this formula, $U_{s_i} = [\![-(m-1)/2, (m-1)/2]\!]\cap[\![s_i-(m-1)/2, s_i+(m-1)/2]\!]$. Hence we have $H^*H\cdot = \bar h * \cdot$, where $\bar h \in \mathbb{R}^{(2m-1)\times(2m-1)}$ is exactly what we expected knowing the associativity of the classical convolution product (different from the one we are interesting in here). We also have :

$$\begin{aligned}
\forall s_1, s_2 \in [\![-(m-1), (m-1)]\!],\ \bar h_{s_1,s_2} &= \sum_{t_1\in U_{s_1}}\sum_{t_2\in U_{s_2}} \tilde h_{t_1,t_2}h_{s_1-t_1,s_2-t_2}\\
&= \sum_{t_1\in U_{s_1}-s_1}\sum_{t_2\in U_{s_2}-s_2} h_{-s_1-t_1,-s_2-t_2}h_{-t_1,-t_2}\\
&= \sum_{t_1\in U_{-s_1}}\sum_{t_2\in U_{-s_2}} h_{-s_1-t_1,-s_2-t_2}h_{-t_1,-t_2}\\
&= \bar h_{-s_1,-s_2} \qquad \text{(B.3)}
\end{aligned}$$

Hence to diagonalise H^*H is the same as diagonalise H, then replacing h by $\bar{h}$. Let $x \in \mathbb{R}^{n\times n}$ an eigenvector of H and λ the associate eigenvalue. Using the lemma B.2, we have :

$$h * x = \lambda x \Rightarrow \forall u, v \in [\![1, n]\!], \lambda \widehat{x}_{u,v} = \widehat{h}_{u,v} \times \widehat{x}_{u,v}$$

Hence as $x \neq 0$, necessary $\lambda \in \{\widehat{h}_{u,v} | u, v \in [\![1, n]\!]\}$. Conversely if $\lambda = \widehat{h}_{\tilde{u},\tilde{v}}$, we define $(x_\lambda)_{u,v} = \exp(2i\pi \left(\frac{u\tilde{u}}{n} + \frac{v\tilde{v}}{n}\right))$, we have :

$$\begin{aligned}
(h * x_\lambda)_{\bar{u},\bar{v}} &= \sum_{u=-\infty}^{+\infty} \sum_{v=-\infty}^{+\infty} h_{u,v} x_{\bar{u}-u,\bar{v}-v} \\
&= \sum_{u=-\infty}^{+\infty} \sum_{v=-\infty}^{+\infty} h_{u,v} \exp(2i\pi \left(\frac{(\bar{u}-u)\tilde{u}}{n} + \frac{(\bar{v}-v)\tilde{v}}{n}\right)) \\
&= \widehat{h}_{\tilde{u},\tilde{v}} \exp(2i\pi \left(\frac{\bar{u}\tilde{u}}{n} + \frac{\bar{v}\tilde{v}}{n}\right)) = \widehat{h}_{\tilde{u},\tilde{v}} (x_\lambda)_{\bar{u},\bar{v}}
\end{aligned}$$

Hence we know $E[\widehat{h}_{\tilde{u},\tilde{v}}]$, the eigenspace associate to the eigenvalue $\widehat{h}_{\tilde{u},\tilde{v}}$:

$$E[\widehat{h}_{\tilde{u},\tilde{v}}] = \text{Span} \left((\exp(2i\pi \left(\frac{u\bar{u}}{n} + \frac{v\bar{v}}{n}\right)))_{u,v} \mid \widehat{h}_{\bar{u},\bar{v}} = \widehat{h}_{\tilde{u},\tilde{v}} \right)$$

In particular we have show that :

Lemma B.4 *Let $h \in \mathbb{R}^{m\times m}$ and $H : x \to h * x$ the associate convolution operator, we have :*

$$\|H\| = \|\widehat{h}\|_\infty$$

Now we compute $\widehat{\bar{h}}$, using B.3:

$$\begin{aligned}
\widehat{\bar{h}}_{\bar{u},\bar{v}} &= \sum_{u=-(m-1)}^{m-1} \sum_{v=-(m-1)}^{m-1} (\bar{h})_{u,v} \exp\left(-2i\pi\left(\frac{u\bar{u}}{n} + \frac{v\bar{v}}{n}\right)\right) \\
&= \sum_{v=-(m-1)}^{m-1} (\bar{h})_{0,v} \exp\left(-2i\pi\frac{v\bar{v}}{n}\right) + \sum_{u=-(m-1)}^{-1} \sum_{v=-(m-1)}^{m-1} (\bar{h})_{u,v} \exp\left(-2i\pi\left(\frac{u\bar{u}}{n} + \frac{v\bar{v}}{n}\right)\right) \\
&\quad + \sum_{u=1}^{m-1} \sum_{v=-(m-1)}^{m-1} (\bar{h})_{u,v} \exp\left(-2i\pi\left(\frac{u\bar{u}}{n} + \frac{v\bar{v}}{n}\right)\right) \\
&= \sum_{v=-(m-1)}^{m-1} (\bar{h})_{0,v} \exp\left(-2i\pi\frac{v\bar{v}}{n}\right) + \sum_{u=1}^{m-1} \sum_{v=-(m-1)}^{m-1} (\bar{h})_{u,v} \exp\left(2i\pi\left(\frac{u\bar{u}}{n} + \frac{v\bar{v}}{n}\right)\right) \\
&\quad + \sum_{u=1}^{m-1} \sum_{v=-(m-1)}^{m-1} (\bar{h})_{u,v} \exp\left(-2i\pi\left(\frac{u\bar{u}}{n} + \frac{v\bar{v}}{n}\right)\right) \\
&= \sum_{v=-(m-1)}^{m-1} (\bar{h})_{0,v} \exp\left(-2i\pi\frac{v\bar{v}}{n}\right) + \sum_{u=1}^{m-1} \sum_{v=-(m-1)}^{m-1} (\bar{h})_{u,v} \times 2\cos\left(2\pi\left(\frac{u\bar{u}}{n} + \frac{v\bar{v}}{n}\right)\right) \\
&= \dots \\
&= (\bar{h})_{0,0} + \sum_{v=1}^{m-1} (\bar{h})_{0,v} \times 2\cos\left(2\pi\frac{v\bar{v}}{n}\right) + \sum_{u=1}^{m-1} \sum_{v=-(m-1)}^{m-1} (\bar{h})_{u,v} \times 2\cos\left(2\pi\left(\frac{u\bar{u}}{n} + \frac{v\bar{v}}{n}\right)\right)
\end{aligned}$$