ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

SUPER-RESOLUTION OFF THE GRID

SEMESTER PROJECT IN COMPUTER SCIENCE

Sébastien Ollquist

29th April 2022

ABSTRACT

Super-Resolution is the tool that allows us to upgrade the quality of images. The goal of this project is to study a recent paper on super-resolution and discuss the main algorithm in order to eventually find an improvement using information theoretical bounds.

> *In theory, theory and practice are the same. In practice, they are not.*
>
> Albert Einstein

CHAPTER 1

INTRODUCTION

With the increasing amount of data nowadays, it has become extremely important to develop tools that allow us to treat it better. In the case of images, unfortunately a lot of them are still taken with old technology leading to a weak resolution. Recently, a new theory called super-resolution has emerged. Its aim is to reconstruct an image of better quality from a poor quality image.

The paper we will study here is called "Super-Resolution Off the Grid" and has been published by Qingqing Huang and Sham M. Kakade in September 2015. From a theoretical point of view, super-resolution is the problem of recovering a superposition of point sources using bandlimited measurements, which may be corrupted with noise.

The goal of this project is to get familiar with the theory of super-resolution along with all important mathematical preliminaries and study the main result presented in the above mentioned paper. The idea is then to compare this work with other related ones and discuss the efficiency of the main algorithm. Ultimately, it would be a nice addition to start working on an improvement of the main algorithm in order to reduce its complexity and thus make it run faster.

CHAPTER 2

MATHEMATICAL REFRESHER

2.1 A MATHEMATICAL THEORY OF SUPER-RESOLUTION

MATHEMATICAL ABSTRACTION We consider k point sources in d dimensions, where the points are separated by a distance at least Δ (in Euclidean distance). The d-dimensional signal $x(t)$ can be modeled as a weighted sum of k Dirac measures in $\mathbb{R}^d$ as

$$x(t) = \sum_{j=1}^{k} w_j \delta_{\mu^{(j)}},$$

where the $\mu^{(j)}$'s are the point sources in $\mathbb{R}^d$ and $w_j \in \mathbb{C}$ the weights such that $|w_j| < C$ for every $j \in [k]$ and some absolute constant $C > 0$. Δ is by definition the minimal possible distance between any two pair of points of the d-dimensional plane. Formally, we regard Δ in terms of Euclidean distance as

$$\Delta = \min_{j \neq j'} ||\mu^{(j)} - \mu^{(j')}||_2.$$

MEASUREMENT FUNCTION There are two main factors that represent our signal: the weights w_j and the complex low pass point spread functions $e^{i\pi\langle s,t\rangle}$ for some measurement s. The former is simply a weighting factor that is not too relevant to us here. The latter however represents the response of an imaging system to a point source. That is, it determines the overall performance level of our system and will be of crucial interest to our study.

We define the measurement function $f(s) : \mathbb{R}^d \to \mathbb{C}$ as being the convolution of the point source $x(t)$ with the point spread function $e^{i\pi\langle s,t\rangle}$. Formally,

$$f(s) = \sum_{j \in [k]} w_j e^{i\pi\langle \mu^{(j)}, s\rangle}.$$

The measurements s in the noisy setting are corrupted with a uniformly bounded perturbation z so that the noisy recovery problem becomes

$$\tilde{f}(s) = f(s) + z(s),$$

in which $|z(s)| \leq \epsilon_z$ for every s and a constant $\epsilon_z \in (0, 1/2)$. The idea is that given access to the signal $x(t)$, we wish to generate a set of random bandlimited Fourier measurements and evaluate the noisy function $\tilde{f}$ on every measurement s in order to recover the parameters $\{w_j, \mu^{(j)} : j \in [k]\}$ of the point source signal as best as we can. The problem statement will be described in more detail in Chapter 4.

2.2 MAIN TOOLS

The main mathematical tools that are needed to understand the paper essentially lie around the subject of linear algebra. They include operations related to vectors, matrices and tensors. We also require some probabilistic analysis tools since the algorithm is partly random. In this chapter, we introduce those tools, prove the more important results and closely relate them to the paper.

2.2.1 GENERALISED EIGENVALUE PROBLEM

Before introducing the generalised eigenvalue problem, it is important to recall what a condition number is for a particular matrix. Suppose for instance we have a matrix $X \in \mathbb{R}^{m\times n}$. We let $\lambda_1, \ldots, \lambda_n$ be the eigenvalues of $X^T X$ (with repetitions) and arrange them so that $\lambda_1 \geq \ldots \geq \lambda_n \geq 0$. Then, the $\sigma_1 \geq \ldots \geq \sigma_n \geq 0$ such that $\sigma_i = \sqrt{\lambda_i}$ are called the *singular values* of X. Define $\sigma_{max}(X) = \sigma_1$ and $\sigma_{min}(X) = \sigma_n$. We then define the condition number of a matrix to be the ratio between the largest and the smallest singular value of X. That is

$$\text{cond}_2(X) = \sigma_1/\sigma_n = \frac{\sigma_{max}(X)}{\sigma_{min}(X)}. \tag{2.1}$$

The above factor governs the noise tolerance of the generalised eigenvalue problem, i.e. it is the measure of sensitiveness of a matrix to arbitrary perturbations. Taking its limiting value will allow us to state whether or not the main algorithm achieves stable recovery of the point sources.

The goal of an eigenvalue problem is to simply find the eigenvalues of a particular matrix A. To do so, we generally solve the equation $AU = UV$, where U is an orthogonal matrix. Multiplying each side by U^T yields

$$AUU^T = UVU^T \Rightarrow A = UVU^T = UVU^{-1},$$

where $\mathbb{R}^{d\times d} \ni U := (u_1, \ldots, u_d)$ are the eigenvectors and $\mathbb{R}^{d\times d} \ni V := Diag\left[(\lambda_1, \ldots, \lambda_d)^T\right]$ the eigenvalues. However, in the generalised version of this problem, we add another random matrix B such that the problem becomes

$$\begin{aligned} AU = BUV &\Rightarrow AUU^T = BUVU^T \\ &\Rightarrow A = BUVU^T = BUVU^{-1}, \end{aligned}$$

where U again are the eigenvectors and V the eigenvalues. Note that A and B are both symmetric and that B is generated randomly with the only condition being that it is positive definite. We form the pair (A, B) called the *pencil* and the pair (U, V) called the *eigenpair*. Therefore, we introduce a first version of what is called the matrix pencil method, in which given a pair of matrices (A, B), we wish to find the generalised eigenvalues λ for which there is a vector $\mathbf{x}$ such that $A\mathbf{x} = \lambda B\mathbf{x}$. Notice that the eigenvalues of A are the solution to the generalised eigenvalue problem where $B = I$.

2.2.2 COMMON NORMS

VECTORS Norms are a powerful linear algebra tool that allows us to retreive important information regarding a particular matrix or vector. The most common vector norm is the L^p-norm defined for a vector X of size n as

$$||X||_p = \left(\sum_{i\in[n]} x_i^p\right)^{1/p}. \tag{2.2}$$

Note that p can take any value in $\mathbb{N}^*$ but the most common ones are 1, 2 or ∞. In essence, the L^1-norm is the sum of absolute values of our vector, the L^2-norm represents the Euclidean distance of the vector from the origin and the L^∞ norm is by definition the largest element in the vector, in absolute value.

MATRICES The same way we define a norm operator for vectors, we can define a similar operator for matrices, that will however have a slightly different meaning. Essentially, a matrix norm is a vector norm in a vector space whose elements are matrices. Let A be a matrix of size $m \times n$. Equivalently, we define the L^p-norm of A as

$$||A||_p = \left(\sum_{i \in [m]} \sum_{j \in [n]} [A]_{i,j}^p \right)^{1/p}. \tag{2.3}$$

The most important matrix norm in our case is an equivalent version of the L^2-norm called the *Frobenius* norm. It basically represents the size of the matrix A, and we define it as

$$||A||_F = \sqrt{\sum_{i \in [m]} \sum_{j \in [n]} [A]_{i,j}^2}. \tag{2.4}$$

2.3 TENSOR DECOMPOSITION

2.3.1 GENTLE INTRODUCTION

A tensor is a generalisation of a matrix to more than two dimensions. We can think of a tensor as a point in $\mathbb{C}^{m_1 \times \ldots \times m_k}$ where k is the order of the tensor. Most of the time here, $k = 3$ since three dimensions suffice for our analysis. Note that if T is an order three tensor of dimensions $m \times n \times p$, we can view it as a collection of p matrices of size $m \times n$ stacked on top of each other. For example, the entry $A_{i,j,k}$ of a 3-tensor A will simply be the (i, j)'th entry of the k'th matrix [1].

We define the *rank* of a tensor V as the minimum r such that we can write V as the sum of rank one tensors. A rank one tensor will be decomposed in the form of a tensor product of three matrices A, B and C as $V = A \otimes B \otimes C$, where each matrix has the same size than that defined above [2]. The decomposition is element-wise defined as $V_{i_1,i_2,i_3} = \sum_{j=1}^{r} A_{i_1,j} B_{i_2,j} C_{i_3,j}$, for every $i_1 \in [m], i_2 \in [n]$ and $i_3 \in [p]$.

An alternative definition is given using the notion of a multi-linear mapping. Namely, for given dimensions m_A, m_B, m_C, the mapping $V(\cdot, \cdot, \cdot) : \mathbb{C}^{m \times m_A} \times \mathbb{C}^{m \times m_B} \times \mathbb{C}^{m \times m_C} \to \mathbb{C}^{m_A \times m_B \times m_C}$ is defined as:

$$[V(X_A, X_B, X_C)]_{i_1,i_2,i_3} = \sum_{j_1,j_2,j_3 \in [m]} V_{j_1,j_2,j_3} [X_A]_{j_1,i_1} [X_B]_{j_2,i_2} [X_C]_{j_3,i_3}.$$

We can verify that for a particular vector $a \in \mathbb{C}^m$, the projection $V(I, I, a)$ of V along the 3rd dimension is $V(I, I, a) = ADiag(C^T a)B^T$ as long as V admits a tensor decomposition $V = A \otimes B \otimes C$. Indeed,

$$\begin{aligned}
[V(I, I, a)]_{i_1,i_2} &= \sum_{j_1,j_2,j_3 \in [m]} V_{j_1,j_2,j_3} [I]_{j_1,i_1} [I]_{j_2,i_2} [a]_{j_3} \\
&= \sum_{j_1,j_2,j_3 \in [m]} \sum_{n \in [k]} A_{j_1,n} B_{j_2,n} C_{j_3,n} [I]_{j_1,i_1} [I]_{j_2,i_2} [a]_{j_3} \\
&= \sum_{j_3 \in [m]} \sum_{n \in [k]} A_{i_1,n} B_{i_2,n} C_{j_3,n} [a]_{j_3} \\
&= \underbrace{\sum_{n \in [k]} A_{i_1,n} B_{i_2,n}}_{=AB} \underbrace{\sum_{j_3 \in [m]} C_{j_3,n} [a]_{j_3}}_{=Diag(C^T a)} \\
&= \left[ADiag(C^T a)B^T \right]_{i_1,i_2}.
\end{aligned}$$

Since the above is true for every i_1, i_2, and we do not consider the i_3 coordinate since a is a vector, we get the desired equality.

2.3.2 TENSOR FROBENIUS NORM

We can also extend the matrix norm definition to that of a tensor. However in our work, we only will need the equivalent Frobenius norm which will be useful in the main algorithm. For the purpose, let V be a rank k 3-tensor of dimensions $m \times n \times 3$. Then, the Frobenius norm of V is simply defined as

$$||V||_F = \sqrt{\sum_{i\in[m]} \sum_{j\in[n]} \sum_{l\in[3]} [V]^2_{i,j,l}}. \tag{2.5}$$

2.3.3 JENNRICH'S ALGORITHM

As previously stated, the goal of tensor decomposition is to, given a tensor T of rank k, decompose it as a sum of rank 1 tensors of appropriate dimensions. Jennrich's algorithm is commonly used for tensor decomposition. We denote the following theorem:

Theorem 1 *Let $T = \sum_{j=1}^{k} u_i \otimes v_i \otimes w_i$ be a tensor in which each set of vectors $\{u_i\}_i$ and $\{v_i\}_i$ are linearly independent. Moreover, each pair of vectors in $\{w_i\}_i$ are also linearly independent. Then, the above decomposition is unique up to rescaling, and there is an efficient algorithm to find it.*

The aforementioned algorithm is described in Algorithm 1. Observe that

Algorithm 1 Jennrich's algorithm for tensor decomposition

Input: a tensor $\tilde{F} \in \mathbb{C}^{m\times m\times 3}$ of rank k.
Choose random unit vectors $a, b \in \mathbb{R}^m$.
Compute $\tilde{F}(I, I, a) = UD_aV^T$, where $D_a = Diag(\langle w^{(i)}, a\rangle)$.
Compute $\tilde{F}(I, I, b) = UD_bV^T$, where $D_b = Diag(\langle w^{(i)}, b\rangle)$.
Compute the diagonalisations $\tilde{F}(I, I, a)\tilde{F}(I, I, b)^{-1}$ and $\tilde{F}(I, I, b)\tilde{F}(I, I, a)^{-1}$.
Solve the linear system to recover the w_j's.
Return U, V, W.

$$\tilde{F}(I, I, a)\tilde{F}(I, I, b)^{-1} = UD_aV^T(V^T)^{-1}D_b^{-1}U^{-1} = UD_aD_b^{-1}U^{-1},$$

and similarly

$$\tilde{F}(I, I, a)^{-1}\tilde{F}(I, I, b) = (V^T)^{-1}D_a^{-1}U^{-1}UD_bV^T = (V^T)^{-1}D_a^{-1}D_bV^T.$$

The correctness of the algorithm follows from the uniqueness of eigendecomposition of a matrix when the eigenvalues are distinct. For a random choice of a and b (in our case we choose the basis vectors e_1 and e_2), with high probability the eigendecompositions are unique so we can recover the u_i's and the v_i's easily by simply recovering the columns of U and V, respectively.

CHAPTER 3

INITIAL WORK

3.1 PRONY'S METHOD

In 1795, the french mathematician Gaspard de Prony came up with a very useful trick to solve a recovery problem, which aims to reconstruct functions from their values at given points. Essentially, we can view those functions as an n'th degree polynomial from which we want to recover the roots. This is today called *Prony's method* and can easily be applied to the theory of super-resolution [3].

GENERAL FORM In a general sense, consider functions of the form $z = f(x) = \sum_{j\in[n]} \mu_j \rho_j^x$, where $\mu_j \in \mathbb{R}$ and $\rho_j \in \mathbb{R}_+$ for every $1 \leq i, j \leq n$. We want to recover those parameters from the samples $z_k := f(x_k)$ of f. We are faced with what is called Prony's condition, which states that given an undetermined number of measurements or sampling points $z_0, z_1, \ldots$, we impose that

$$(z * p)_k = \sum_{j=0}^{n} z_{k+1} p_j = 0,$$

for $k = 0, 1, 2, \ldots$ and we wish to solve the above equations for the p_j's. We see that the solutions are given by $p(x) = (x - \widehat{\rho}_1) \cdots (x - \widehat{\rho}_n)$. In a more algorithmic sense, we seek to

1. solve the above linear system for p and
2. find the zeroes of $p(x)$ to obtain the ρ_j's.

LINK WITH SUPER-RESOLUTION Prony's method can in fact be applied to many domains. For example in microscopy, images often contain brightness spikes on the plane but the optical system that contains all the hardware to create the picture essentially acts as a low pass filter: it will keep the low frequencies of a signal and attenuate the frequencies that are higher than a pre-determined cutoff frequency. By doing so, those spikes are turned into what are called *point spread functions*, that determine the performance of an imaging system. Similarly to our problem, the goal of Prony's method is to reconstruct the image by recovering estimates of the original spikes.

We can view the plane of an image as part of $\mathbb{R}^2$. Let its spikes be at positions $X \subset \mathbb{R}^2$ and let them have amplitude a_x. Note that X is sparse. Taking the Fourier transform yields

$$z_\alpha := z_{\alpha_1,\alpha_2} = \sum_{x\in X} a_x e^{\alpha_1 x_1 + \alpha_2 x_2},$$

where $\alpha = (\alpha_1, \alpha_2) \in \mathbb{Z}^2$. Through the low pass filter, we obtain a finite amount of measurements

$$z_\alpha = \sum_{x \in X} a_x \rho_x^\alpha,$$

for $|\alpha_1|, |\alpha_2| \leq n$ where $\rho_x^\alpha = \rho_{x,1}^{\alpha_1} \rho_{x,2}^{\alpha_2}$ and $\rho_{x,j} = e^{ix_j}$ for $j = 1, 2$. Note that the recovery of the z_α's is exactly the 2D version of Prony's problem: the a's are our weights and the ρ_x^α's are our point sources.

3.2 UNIVARIATE CASE: MATRIX-PENCIL METHOD

As an extension of Prony's method which is not stable, we consider the main algorithm in the univariate case, the setting of which is as follows. We construct two $m \times m$ complex valued Hankel matrices H_0 and H_1, that is, matrices such that their skew-diagonals[1] are constants. We have $D_w \in \mathbb{C}^{k \times k}$ where $[D_w]_{j,j} = w_j$ and $D_\mu \in \mathbb{C}^{k \times k}$ where $[D_\mu]_{j,j} = e^{i\pi\mu^{(j)}}$. We furthermore have a Vandermonde matrix $V_m \in \mathbb{C}^{m \times k}$ defined as

$$V_m = \begin{pmatrix} 1 & \dots & 1 \\ \left(e^{i\pi\mu^{(1)}}\right)^1 & \dots & \left(e^{i\pi\mu^{(k)}}\right)^1 \\ \vdots & \dots & \vdots \\ \left(e^{i\pi\mu^{(1)}}\right)^{m-1} & \dots & \left(e^{i\pi\mu^{(k)}}\right)^{m-1} \end{pmatrix}$$

We construct a 3rd order tensor $F \in \mathbb{C}^{m \times m \times 2}$ in which $F_{i,i',j} = [H_{j-1}]_{i,i'}$ for $j = 1, 2$ and $i, i' \in [m]$. The following computations will help us better understand tensor decomposition and the main algorithm. We first denote the following fact:

Fact 2 *In the setting above, F admits the unique rank k tensor decomposition $F = V_m \otimes V_m \otimes (V_2 D_w)$.*

Proof We start by computing the product $V_2 D_w$. By definition of V_m, we have that $[V_2]_{r,c} = \left(e^{i\pi\mu^{(c)}}\right)^{r-1}$ for $r \in [2]$ and $c \in [k]$. Multiplying with the diagonal matrix D_w yields

$$[V_2 D_w]_{r,c} = w_c \left(e^{i\pi\mu^{(c)}}\right)^{r-1}, \quad \forall r \in [2], c \in [k].$$

We aim to show that

$$\begin{aligned} F_{i_1,i_2,i_3} &= \sum_{n=1}^{k} [V_m]_{i_1,n} [V_m]_{i_2,n} [V_2 D_w]_{i_3,n} \\ &= \sum_{n=1}^{k} \left(e^{i\pi\mu^{(n)}}\right)^{i_1-1} \left(e^{i\pi\mu^{(n)}}\right)^{i_2-1} w_n \left(e^{i\pi\mu^{(n)}}\right)^{i_3-1} \\ &= \sum_{n=1}^{k} w_n \left(e^{i\pi\mu^{(n)}}\right)^{i_1+i_2+i_3-3}, \end{aligned}$$

for $i_1, i_2 \in [m]$ and $i_3 \in [2]$. It can be verified that the measurements that form both Hankel matrices are given by

$$f(s) = \sum_{j \in [k]} w_j \left(e^{i\pi\mu^{(j)}}\right)^s,$$

with $s = 0, \dots, 2m-1$ for H_0 and $s = 1, \dots, 2m$ for H_1. For the details of the proof, see Appendix A. Then, since $F_{i_1,i_2,i_3} = [H_{i_3-1}]_{i_1,i_2}$, for a fixed i_3 we have either $F_{i_1,i_2,1} = [H_0]_{i_1,i_2}$ or $F_{i_1,i_2,2} = [H_1]_{i_1,i_2}$. Note that since we are dealing with Hankel matrices, $[H_0]_{i_1,i_2} = f(i_1 + i_2 - 2)$. Hence,

$$F_{i_1,i_2,i_3} = \sum_{n \in [k]} w_n \left(e^{i\pi\mu^{(n)}}\right)^{i_1+i_2+i_3-3} = f(i_1 + i_2 + i_3 - 3),$$

[1] A skew-diagonal is the diagonal in the North-East direction.

as required. ■

Recall that Jennrich's algorithm computes the simultaneous diagonalisations of $F(I, I, v_1)$ and $F(I, I, v_2)$, in which v_1 and v_2 are two random projections of $\mathbb{R}^m$. Note that setting $v_1 = e_1$ and $v_2 = e_2$ yields H_0 and H_1 exactly. Hence, we conclude that the matrix-pencil method studied here is just a special case of Jennrich's algorithm where both projections are done on the two basis vectors e_1 and e_2.

3.3 MULTIVARIATE TOY CASE

This section describes a preview of the main algorithm when the measurements are not random and where we assume for simplicity that the point sources are uniformly distributed in $[-1, +1]$. We follow the definitions given in the paper. We have $D_w = Diag(w_j) \in \mathbb{C}^{k\times k}$ and $[V_d]_{r,c} = e^{i\pi\mu_r^{(c)}} \in \mathbb{C}^{d\times k}$ the latter in which $\mu_n^{(i)}$ are i.i.d. $\sim Unif([-1, +1])$. Furthermore, we have $F_{n_1,n_2,n_3} = f(s)\big|_{s=e_{n_1}+e_{n_2}+e_{n_3}}$, for all $n_1, n_2, n_3 \in [d]$. We have the following fact:

Fact 3 *In the setting above, F admits the tensor decomposition $F = V_d \otimes V_d \otimes (V_d D_w)$.*

Proof We wish to show that $f(s)\big|_{s=s^{(n_1)}+s^{(n_2)}+s^{(n_1)}} = \sum_{j=1}^{k} w_j e^{i\pi(\mu_{n_1}^{(j)}+\mu_{n_2}^{(j)}+\mu_{n_3}^{(j)})}$. To do so, we first compute the matrix product $V_d D_w$. Since $[V_d]_{r,c} = e^{i\pi\mu_r^{(c)}}$ for $r \in [d], c \in [k]$ and $D_w = Diag(w_j)$, we directly have that

$$[V_d D_w]_{r,c} = w_c e^{i\pi\mu_r^{(c)}}, \quad r \in [d], c \in [k],$$

so that by definition of tensor decomposition, we get

$$\begin{aligned}
F_{n_1,n_2,n_3} &= \sum_{j=1}^{k} [V_d]_{n_1,j}[V_d]_{n_2,j}[V_d D_w]_{n_3,j} \\
&= \sum_{j=1}^{k} e^{i\pi\mu_{n_1}^{(j)}} e^{i\pi\mu_{n_2}^{(j)}} w_j e^{i\pi\mu_{n_3}^{(j)}} \\
&= \sum_{j=1}^{k} w_j e^{i\pi(\mu_{n_1}^{(j)}+\mu_{n_2}^{(j)}+\mu_{n_3}^{(j)})} \\
&= f(s)\big|_{s=s^{(n_1)}+s^{(n_2)}+s^{(n_1)}} = f(e_{n_1} + e_{n_2} + e_{n_3}),
\end{aligned}$$

as required. ■

The key idea here is to notice that the entries $e^{i\pi\mu_n^{(j)}}$ are i.i.d. and uniformly distributed over the unit circle of the complex plane. This is due to the fact that the $\mu_n^{(j)}$ factors are uniform. This implies that the factor V_d almost surely has full column rank and therefore that the tensor decomposition exists and is unique (since each vector contained in the tensor's matrices are linearly independent). Again by uniformity assumption, the condition number of V_d concentrates around 1 so the above algorithm achieves stable recovery. We will see in more details in the next chapter.

CHAPTER 4

MAIN ALGORITHM

Let us recall what the goal of the problem is. Broadly speaking, super-resolution aims to recover a superposition of point sources using bandlimited measurements that may be corrupted with noise. In this setting, the created algorithm works in the Fourier domain where each of the k points of the d-dimensional plane are separated by a distance at least Δ. Hence, the frequencies of the Fourier measurements are bounded by $O(1/\Delta)$. The idea of the procedure is to take random bandlimited measurements with cutoff frequency bounded by $\Omega(\sqrt{d}/\Delta)$ and perform Tensor decomposition to recover the estimates of the point sources.

This chapter discusses the main algorithm along with its analysis. We will go through all the important results and describe in detail their implication on the main procedure to effectively recover the point sources.

4.1 ALGORITHM DESCRIPTION

4.1.1 INTRODUCTION

The basic idea of the algorithm is to efficiently solve the problem of recovering the superposition of our point sources using *coarse* Fourier measurements. Once again, we are given a signal $x(t) = \sum_{j\in[k]} w_j \delta_{\mu^{(j)}}$ from which we want to recover the w_j and $\mu^{(j)}$ coefficients. We have access to a noisy measurement function $\tilde{f}(\cdot)$. The idea is to generate random samples s and evaluate $\tilde{f}$ for each of them in order to find the best fit, i.e. the best fitting $\mu^{(j)}$'s. More formally, for all s we compute

$$\tilde{f}(s) = \sum_{j\in[k]} w_j e^{i\pi\langle\mu^{(j)},s\rangle} + z(s),$$

where $z(s)$ is the noise upper bounded by ϵ_z for every s and store those results in a 3-way tensor. Compared to previous work, the algorithm guarantees a certain stability of recovery and runs in quadratic time. Namely, the measurements are bounded in frequency by $O(1/\Delta)$ and the algorithm runs in time $O((k\log k+d)^2)$. The key recovery guarantee relies on the condition number of the random Vandermonde matrix. This is however further discussed in the next section.

4.1.2 MAIN RESULT

INPUT AND OUTPUT The algorithm takes as input a cutoff frequency R, the defined number of measurements m and a noisy measurement function $\tilde{f}(\cdot)$ and outputs the set of estimates

$$\{\widehat{w}_j, \widehat{\mu}^{(j)} : j \in [k]\},$$

where the $\widehat{w}_j$'s are the complex weight coefficients and the $\widehat{\mu}^{(j)}$'s are the estimates of the point sources. Note that in case the noise is non-existant (i.e. when $\epsilon_z = 0$), the parameters are recovered exactly. Otherwise, stable recovery implies that the estimates over all permutations π on $[k]$ are such that

$$\min_{\pi} \max \left\{ ||\widehat{\mu}^{(j)} - \mu^{(\pi(j))}||_2 : j \in [k] \right\} \leq poly(d,k)\epsilon_z.$$

In words, the estimates of the point sources differ from their real values by at most the noise ϵ_z scaled by a polynomial function that depends on the number of dimensions d and the number of point sources k.

MEASUREMENTS We generate a set of measurements $\mathcal{S} = \{s^{(1)}, \ldots, s^{(m+d+1)}\}$ randomly in which

1. the $s^{(1)}, \ldots, s^{(m)}$ are m i.i.d. samples from the Gaussian distribution $\mathcal{N}(0, R^2 I_{d\times d})$,
2. $s^{(m+n)} = e_n$ for all $n \in [d]$ and
3. $s^{(m+d+1)} = 0$.

We also let $m' = m + d + 1$ for simplicity. Note that each measurement is a d-dimensional vector, i.e. a point in the d-dimensional plane. We then take a sample v from the unit sphere and set $v^{(1)} = v$ and $v^{(2)} = 2v$. Since we are in dimension d, the sample v is such that $v_1^2 + \ldots + v_d^2 = 1$.

TENSOR DECOMPOSITION We now construct a tensor

$$\tilde{F} \in \mathbb{C}^{m'\times m'\times 3} : \tilde{F}_{n_1,n_2,n_3} = \tilde{f}(s)|_{s=s^{(n_1)}+s^{(n_2)}+v^{(n_3)}}$$

containing all the $2(m')^2$ possible combinations of the above measurements. We apply Jennrich's algorithm on $\tilde{F}$ to obtain the estimates $\widehat{V}_{S'}$ and $\widehat{D}_w$. Note that the estimate $\widehat{V}_{S'}$ is normalised so that its last line is all ones, in order to match the definition of $V_{S'}$ (see next section).

READ OF ESTIMATES We finish by recovering the real part of the estimates of the point sources by setting $\widehat{\mu}^{(j)} = Real(\log([\widehat{V}_{S'}]_{[m+1:m+d,j]})/(i\pi))$ for every $j \in [k]$ and find the best possible matching coefficients by setting $\widehat{W} = \arg\min_{W\in\mathbb{C}^k} ||\widehat{F} - \widehat{V}_{S'} \otimes \widehat{V}_{S'} \otimes \widehat{V}_d D_w||_F$, where $||\cdot||_F$ is the Frobenius norm of $(\cdot)$. Here, observe that by definition $[\widehat{V}_{S'}]_{[m+1:m+d,j]} = [\widehat{V}_d]_{[1:d,j]}$, where $\log(\widehat{V}_d)$ is performed element-wise in $\widehat{V}_d$ to recover the list of all estimates.

4.2 STABILITY GUARANTEE

The main guarantee on the stability of the recovery of the point sources is given by

$$\min_{\pi} \max \left\{ ||\widehat{\mu}^{(j)} - \mu^{(\pi(j))}||_2 : j \in [k] \right\} \leq C \frac{\sqrt{d}k^5}{\Delta\delta_v} \frac{w_{max}}{w_{min}^2} \left(\frac{1+2\epsilon_x}{1-2\epsilon_x} \right)^{5/2} \epsilon_z,$$

where C is a universal constant such that the bound holds with probability at least $(1-\delta_s)$ over the random sampling of $\mathcal{S}$ and with probability at least $(1-\delta_v)$ over the random projections in Jennrich's algorithm. Note that **the spatial resolution of an imaging device may be measured by how closely lines can be resolved**.

4.2.1 STABILITY OF TENSOR DECOMPOSITION

A main factor of stability resides in the tensor decomposition procedure. Since the measurements are linearly independent, the tensor decomposition exists and is unique up to column permutation and

rescaling. Furthermore, suppose that F admits the decomposition $F = V \otimes V \otimes (V_2 D_w) \in \mathbb{C}^{m\times m\times 3}$ and that $\tilde{F}$ is element-wise close to F, that is, each of their elements differ by at most a constant ϵ_z. If $\tilde{F}$ is passed as input to Jennrich's algorithm, it happens that $\widehat{V}$ is close enough to V with high probability. To show this, denote $D_1 = diag([V_2]_{1,:}D_w)$ and $D_2 = diag([V_2]_{2,:}D_w)$ where $[V_w]_{i,:}$ is the i'th row vector of V_2. Let $F_1 = F(I, I, e_1)$. By previous computations, we know that $F_1 = VD_1V^T$. Consider the SVD $F_1 = P\Lambda P^T$ and let $U = P^TV$. Define $E = F(P, P, I)$.

Fact 4 *In the above setting,* $E = F(P, P, I) = U \otimes U \otimes (V_2 D_w) \in \mathbb{C}^{k\times k\times 3}$.

Proof By direct computation, we have

$$\begin{aligned}
E_{n_1,n_2,n_3} = F(P,P,I)_{n_1,n_2,n_3} &= \sum_{j_1,j_2\in[m']}\sum_{j_3\in[3]} F_{j_1,j_2,j_3}[P]_{j_1,n_1}[P]_{j_2,n_2}[I]_{j_3,n_3} \\
&= \sum_{j_1,j_2\in[m']}\sum_{j_3\in[3]}\sum_{n\in[k]} [V]_{j_1,n}[V]_{j_2,n}[V_2D_w]_{j_3,n}[P]_{j_1,n_1}[P]_{j_2,n_2}[I]_{j_3,n_3} \\
&= \sum_{j_1,j_2\in[m']}\sum_{n\in[k]} [V]_{j_1,n}[P]_{j_1,n_1}[V]_{j_2,n}[P]_{j_2,n_2}[V_2D_w]_{n_3,n} \\
&= \sum_{j_1,j_2\in[m']}\sum_{n\in[k]} [P^T]_{n_1,j_1}[V]_{j_1,n}[P^T]_{n_2,j_2}[V]_{j_2,n}[V_2D_w]_{n_3,n} \\
&= \sum_{n\in[k]} [P^TV]_{n_1,n}[P^TV]_{n_2,n}[V_2D_w]_{n_3,n},
\end{aligned}$$

which is equivalent to writing that $E = (P^TV) \otimes (P^TV) \otimes (V_2D_w)$ when summing over all possible n_1, n_2 and n_3's. This concludes the proof. ■

Now slice the tensor E in two such that we define $E_1 = E(I, I, e_1)$ and $E_2 = E(I, I, e_2)$. Note that $E_1 = UD_1U^T$ and $E_2 = UD_2U^T$. Thus, computing $E_1E_2^{-1}$ yields

$$\begin{aligned}
E_1E_2^{-1} &= UD_1U^T(U^TD_2U)^{-1} = UD_1U^T(U^T)^{-1}D_2^{-1}U^{-1} \\
&= UD_1D_2^{-1}U^{-1} = UDU^{-1} = M,
\end{aligned}$$

when letting $D = D_1D_2^{-1}$.

4.3 ANALYSIS

Let us first recall what variables are used in the main algorithm.

- $[V_S]_{r,c} = e^{i\pi\langle\mu^{(c)},s^{(r)}\rangle}$ is the *characteristic matrix*, that is, the matrix that contains all the point spread functions generated by the random Gaussian measurements.
- $[V_d]_{r,c} = e^{i\pi\mu_r^{(c)}}$ is the matrix that contains all the complex valued coefficients that lie on the unit circle of the complex plane. Note that this is due to the uniformity of the μ's.
- $V_{S'} = \left[V_S,\, V_d,\, [1]^k\right]^T$ is the matrix, the coefficients of which we shall try to recover during tensor decomposition.
- Finally, define

$$V_2 = \begin{pmatrix} e^{i\pi\langle\mu^{(1)},v^{(1)}\rangle} & \dots & e^{i\pi\langle\mu^{(k)},v^{(1)}\rangle} \\ e^{i\pi\langle\mu^{(1)},v^{(2)}\rangle} & \dots & e^{i\pi\langle\mu^{(k)},v^{(2)}\rangle} \\ 1 & \dots & 1 \end{pmatrix},$$

  where the last row of all ones is useful for normalisation purposes. We can view it as containing exponentials, the factor of which is a dot product with 0.

Fact 5 *In the exact case (i.e. when $\epsilon_z = 0$), the constructed tensor F admits a rank-k decomposition*

$$F = V_{S'} \otimes V_{S'} \otimes (V_2 D_w).$$

Proof As usual, we start by computing the product $V_2 D_w$. Since D_w is a diagonal matrix, we have

$$V_2 D_w = \begin{pmatrix} w_1 e^{i\pi\langle \mu^{(1)}, v^{(1)}\rangle} & \dots & w_k e^{i\pi\langle \mu^{(k)}, v^{(1)}\rangle} \\ w_1 e^{i\pi\langle \mu^{(1)}, v^{(2)}\rangle} & \dots & w_k e^{i\pi\langle \mu^{(k)}, v^{(2)}\rangle} \\ w_1 & \dots & w_k \end{pmatrix}.$$

Notice that once again, we can think of multiplying the last row of w's with an exponential whose exponent is 0. Then, write $[V_d]_{r,c} = e^{i\pi\mu_r^{(c)}} = e^{i\pi\langle \mu^{(c)}, e_r = s(m+r)\rangle}$. Using this, we can express the r'th row and the c'th column of $V_{S'}$ as

$$[V_{S'}]_{r,c} = e^{i\pi\langle \mu^{(c)}, s^{(r)}\rangle}, \quad r \in [m'], c \in [k].$$

From there for $n_1, n_2 \in [m']$ and $n_3 \in [3]$, it follows easily that

$$\begin{aligned} F_{n_1,n_2,n_3} &= \sum_{n\in[k]} [V_{S'}]_{n_1,n}[V_{S'}]_{n_2,n}[V_2 D_w]_{n_3,n} \\ &= \sum_{n\in[k]} e^{i\pi\langle \mu^{(n)}, s^{(n_1)}\rangle} e^{i\pi\langle \mu^{(n)}, s^{(n_2)}\rangle} w_n e^{i\pi\langle \mu^{(n)}, v^{(n_3)}\rangle} \\ &= \sum_{n\in[k]} w_n e^{i\pi\left(\langle \mu^{(n)}, s^{(n_1)}\rangle + \langle \mu^{(n)}, s^{(n_2)}\rangle + \langle \mu^{(n)}, v^{(n_3)}\rangle\right)} \\ &= \sum_{n\in[k]} w_n e^{i\pi\langle \mu^{(n)}, s^{(n_1)} + s^{(n_2)} + v^{(n_3)}\rangle} \\ &= \tilde{f}(s)\big|_{s = s^{(n_1)} + s^{(n_2)} + v^{(n_3)}}, \end{aligned}$$

as required. ∎

4.3.1 EXACT CASE RECOVERY

CHAPTER 5

DISCUSSION

CHAPTER 6

IMPROVEMENT

CHAPTER 7

CONCLUSION

BIBLIOGRAPHY

[1] Ankur Moitra. *Lecture notes in Algorithmic aspects of Machine Learning*. 2015.

[2] Tim Roughgarden and Gregory Valiant. *Lecture notes in The modern algorithmic toolbox*. 2015.

[3] Tomas Sauer. 'Prony's method: an old trick for new problems'. In: *Snapshots of modern mathematics from Oberwolfach* (2018).

APPENDIX A

CONSTRUCTION OF HANKEL MATRICES

In this section, we prove that the Hankel matrices H_0 and H_1 defined earlier admit the respective diagonalisations $V_m D_w V_m^T$ and $V_m D_w D_\mu V_m^T$. To do so, we first compute the product $D_w V_m^T$. We have

$$
\begin{aligned}
D_w V_m^T &= \begin{pmatrix} w_1 & 0 & \dots & 0 \\ 0 & w_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & w_k \end{pmatrix} \begin{pmatrix} 1 & e^{i\pi\mu^{(1)}} & \dots & \left(e^{i\pi\mu^{(1)}}\right)^{m-1} \\ 1 & e^{i\pi\mu^{(2)}} & \dots & \left(e^{i\pi\mu^{(2)}}\right)^{m-1} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & e^{i\pi\mu^{(k)}} & \dots & \left(e^{i\pi\mu^{(k)}}\right)^{m-1} \end{pmatrix} \\
&= \begin{pmatrix} w_1 & w_1 e^{i\pi\mu^{(1)}} & \dots & w_1\left(e^{i\pi\mu^{(1)}}\right)^{m-1} \\ w_2 & w_2 e^{i\pi\mu^{(2)}} & \dots & w_2\left(e^{i\pi\mu^{(2)}}\right)^{m-1} \\ \vdots & \vdots & \ddots & \vdots \\ w_k & w_k e^{i\pi\mu^{(k)}} & \dots & w_k\left(e^{i\pi\mu^{(k)}}\right)^{m-1} \end{pmatrix},
\end{aligned}
$$

so that

$$
\begin{aligned}
H_0 = V_m D_w V_m^T &= \begin{pmatrix} 1 & \dots & 1 \\ e^{i\pi\mu^{(1)}} & \dots & e^{i\pi\mu^{(k)}} \\ \vdots & \dots & \vdots \\ \left(e^{i\pi\mu^{(1)}}\right)^{m-1} & \dots & \left(e^{i\pi\mu^{(k)}}\right)^{m-1} \end{pmatrix} \begin{pmatrix} w_1 & w_1 e^{i\pi\mu^{(1)}} & \dots & w_1\left(e^{i\pi\mu^{(1)}}\right)^{m-1} \\ w_2 & w_2 e^{i\pi\mu^{(2)}} & \dots & w_2\left(e^{i\pi\mu^{(2)}}\right)^{m-1} \\ \vdots & \vdots & \ddots & \vdots \\ w_k & w_k e^{i\pi\mu^{(k)}} & \dots & w_k\left(e^{i\pi\mu^{(k)}}\right)^{m-1} \end{pmatrix} \\
&= \begin{pmatrix} \sum_{j\in[k]} w_j & \sum_{j\in[k]} w_j e^{i\pi\mu^{(1)}} & \dots & \sum_{j\in[k]} w_j\left(e^{i\pi\mu^{(1)}}\right)^{m-1} \\ \sum_{j\in[k]} w_j e^{i\pi\mu^{(1)}} & \sum_{j\in[k]} w_j\left(e^{i\pi\mu^{(1)}}\right)^{2} & \dots & \sum_{j\in[k]} w_j\left(e^{i\pi\mu^{(1)}}\right)^{m} \\ \vdots & \vdots & \ddots & \vdots \\ \sum_{j\in[k]} w_j\left(e^{i\pi\mu^{(1)}}\right)^{m-1} & \sum_{j\in[k]} w_j\left(e^{i\pi\mu^{(1)}}\right)^{m} & \dots & \sum_{j\in[k]} w_j\left(e^{i\pi\mu^{(1)}}\right)^{2m-1} \end{pmatrix} \\
&= \begin{pmatrix} f(0) & f(1) & \dots & f(m-1) \\ f(1) & f(2) & \dots & f(m) \\ \vdots & \vdots & \ddots & \vdots \\ f(m-1) & f(m) & \dots & f(2m-1) \end{pmatrix},
\end{aligned}
$$

where $f(s) = \sum_{j\in[k]} w_j \left(e^{i\pi\mu^{(j)}}\right)^s$ for $s = 0, \dots, 2m-1$ which indeed corresponds to the defined Hankel matrix. The proof is similar for H_1 except that s varies from 1 to $2m$ since element-wise multiplication of H_0 with the diagonal matrix D_μ yields a simple scaling by a factor $e^{i\pi\mu^{(j)}}$.