Automatic Pipeline for Microarray Image Analysis Based on Image projections

Mohamed Maoui

Program of Biological & Biomedical Engineering

McGill University, Montreal
November 2021

A thesis submitted to McGill University in partial fulfillment of the requirements of the degree of Master of Engineering

Contents

List of Figures

List of Tables

Abstract

Microarrays are a powerful tool for multiplexed analysis of DNA, RNA, proteins and EVs. They involve microarray designing, manufacturing, incubation with sample and fluorescent detection reagents, scanning, image processing, and data analyzing. Once the microarray images related to a particular experiment are produced, we follow a process of image analysis that is organized into three steps: gridding, segmentation, and intensity calculation. Ideally, the analysis would be much easier without noise sources that is mostly due to inhomogeneous spots intensities, dust, slides inhomogeneity and bias, as well as other challenges such as spots with noncircular shape or similar size, which require additional processing steps in order to accurately analyze the microarray data. Different software packages are available to analyze microarray data and overcome some of the traditional image analysis issues, such as ScanAlyze, and GenePix Pro6, or Array pro Analyzer. Unfortunately, these software require manual intervention to achieve accurate gridding, signal and background separation, as well as precise expression levels estimation, and they are “black boxes” which produce the final microarray image quantification results without showing most preprocessing or analysis steps such as channels splitting, or the denoising phase which is unsatisfactory and limits the ability to diagnose and correct image analysis errors. In this work, we developed an automated image analysis pipeline for microarray images, with emphasis on exosomes and EV analysis. The process of automation includes (1) gridding based on projection techniques (2) a thresholding image segmentation, and (3) an algorithm for intensity calculation. Different gridding parameters have been benchmarked based on the processing time, and accuracy of correctly gridding the spots, and its noise removal ability. We also explored the use of machine learning for microarray research by implementing k-means clustering algorithm on similar microarray images. We compared our data processing pipeline to Array Pro by using real microarray data from EV analysis experiments. Our automated pipeline, which is fast, effective with a gridding accuracy that gets to 97.5% and a segmentation similarity of 98%, can improve our understanding of the image analysis steps of many software used to analyze microarray data, and offer a universal way for more accurate analysis of microarray data and EVs, by assessing different quality metrics of any type of microarray, and therefore help in answering different biological questions.

Résumé

La technologie des micropuces est un outil essentiel utilisé pour l'analyse multiplexée de l'ADN, d'ADN, l'ARN, l'analyse des protéines et les biopuces à anticorps. Ils impliquent la conception, la fabrication, l'incubation avec des échantillons et des réactifs de détection fluorescents, la numérisation, le traitement d'images et l'analyse des données. Une fois que les images de micropuces liées à une expérience particulière sont produites, Nous utilisons un pipeline d'analyse d'images organisé en trois étapes: le maillage, la segmentation et calcul d'intensité. Idéalement, le processus d'analyse d'image serait beaucoup plus facile, si l'arrière-plan était dépourvu de sources de bruit sont principalement due à des intensités de points non homogènes, à la poussière, à l'inhomogénéité et au biais, et si tous les points avaient des formes circulaires, une taille similaire qui nécessitent des étapes de traitement supplémentaires afin d'analyser avec précision les données des micropuces. Différents logiciels sont disponibles pour analyser les données des micropuces et surmonter certains des problèmes d'analyse d'image traditionnels, tels que ScanAlyze et GenePix Pro6 ou Array Pro Analyzer. Malheureusement, ces logiciels nécessitent une intervention manuelle pour obtenir un maillage précis, une séparation du signal et de l'arrière-plan, ainsi qu'une estimation précise des niveaux d'expression, et ce sont des 'boîtes noires' qui produisent les résultats finaux de quantification d'images de micropuces sans montrer la plupart des étapes de prétraitement ou d'analyse telles que la séparation des canaux, ou la phase de débruitage qui n'est pas satisfaisante et limite la capacité de diagnostic et de correction des erreurs d'analyse d'images. Dans ce travail, nous avons développé un pipeline d'analyse d'image automatisé en mettant l'accent sur les exosomes et l'analyse des EV qui permettra de lier les résultats du diagnostic à des paramètres de traitement d'image spécifiques. Le processus d'automatisation comprend (1) un maillage basé sur des techniques de projection (2) une segmentation d'image de seuillage, et (3) un algorithme de calcul d'intensité. Différents paramètres de cette approche ont été évalués tels que le temps de traitement, la précision de quadrillage et sa capacité d'élimination du bruit de fond. Nous avons également exploré l'impact de l'apprentissage automatique sur la recherche liée aux micropuces en implémentant l'algorithme k-means sur des images de micropuces similaires. Nous avons comparé notre pipeline de traitement de données à Array Pro en utilisant des données micropuces réelles issues d'expériences d'analyse EV. Notre pipeline, qui est rapide, efficace et avec une précision de maillage atteignant 97,5 % et une similarité de segmentation de 98 %, peut améliorer notre compréhension des nombreux logiciels utilisés pour analyser les données des micropuces et offrir un moyen universel pour une analyse plus précise des données des micropuces et des EV, en évaluant différentes métriques de qualité de tout type de micropuces, et donc aider à répondre à différentes questions biologiques.

Acknowledgment

I would like to start by thanking my supervisor Pr. David Juncker for his continuous support and patience, this thesis wouldn't have been possible without him. I would like to also thank Rosalie Martel for providing most of the microarray images I used for this project, and for the helpful discussions we had about data analysis. I am also thankful to Edward Zhang for offering academic guidance as my mentor in the Juncker lab, and to Alia Alameri for the thoughtful discussions we had and the emotional support she given me during my first year at McGill, and to Ilias Hurley for peer reviewing my research proposals, and for the empathetic discussions we had, without I forget everyone at the Juncker lab for their generosity, and consistent feedback.

I am honestly beyond grateful to everyone at the OSD, especially Gift, Claire, Isabella, and my mentors Victoria Madge and Amanda keller who supported me in so many ways and positively impacted both my academic and personal life.

Furthermore, I am thankful to the BBME program at McGill for their genuine support and accommodations. I also would like to acknowledge the financial support I got from AlGhurair Foundation for Education for offering me a full ride scholarship to pursue my graduate degree, as well as the office of graduate and postdoctoral studies at McGill and the scholarship and students aid office for offering different awards. Graduate school was a rewarding experience, and being at McGill has helped me grow in so many levels, and shaped the person I am today.

Finally, I am grateful for my mother and my family for their moral support and all the sacrifices they made for me. A warm thought to those friends who helped me in every step of my degree, especially Yuri Sosero, Jeremy Drelon, Yasmine Bouguerche, Redouane Belmokhtar and my friends at MORSL of McGill especially Ffion huges and Carlene gardner.

Chapter 1: Introduction

Array based technology represents a powerful tool to analyze data generated from experiments on DNA, RNA, protein analysis, and the expression levels of several molecules of interest [1]. The technology is a multiplex lab-on-a-chip, a two-dimensional array consisting of a solid surface, onto which molecules of interests such as antibodies or DNA molecules are chemically bound. The platform assays large amounts of biological material using high-throughput screening miniaturized, multiplexed, and parallel processing and detection methods[2]. Any reaction between the probe and the immobilised molecule of interest emits a fluorescent signal that is read by a laser scanner [3]. The result is a set of microarray images related to a particular experience that require a pipeline of image analysis. The process is organized in three steps: gridding, segmentation and intensity calculation [4]. Ideally, the image analysis procedure would be much easier, if the background was noise and artifact-free and all the spots had circular shapes, similar size, and homogeneous intensity. However, this is not the case, and the quality of microarrays we usually encounter makes the analysis more challenging since there are various noise sources, including slide inhomogeniety and bias that require a lot of flexibility and adjustments in order to accurately analyze the microarray data [4]. We first start the image analysis by locating the spots from the scanned images using a gridding technique. Next, we segment each spot individually into regions of two classes: foreground and background, and finally, the expression levels will be computed using the segmentation results.

During the preparation of microarrays gridding, a lot of parameters need to be taken into consideration, such as the total count of spots and their dimension, the total count of associated-grids and their position, and while trying to evaluate these parameters, we often have to deal with obstacles such as noise, or improper alignment, that can make it hard to locate the correct location of the spots, which affects the gridding results[5] The gridding step is an essential part of the image analysis process, and when it is done manually, it could cause variations of the expression results[6]. After the gridding step is done, we move to the segmentation phase. The purpose of this step is to prepare the image for future feature extractions analysis. Segmentation can be seen as the classification of each image pixel to be assigned to one of the image compositions[7]. It was also shown that the method chosen for the segmentation significantly affects the precision of the microarray data, so it is essential that the step of segmentation is carefully done for an accurate quantification[8].The last step of the image analysis process is to extract intensity values and expression levels based on the results of the previous processing stages (gridding and segmentation) [9]. The numerical results will have to undergo a data analysis procedure.

Extracellular vesicles (EVs) are small phospholipid membrane-enclosed entities secreted by cells, they come in three different subtypes, which are microvesicles (MVs), exosomes, and apoptotic bodies, and are distinguished by their biogenesis, release pathways, size, content and function [10, 11] Exosomes are the smallest (30–150 nm) of these EVs secreted and are present in all body fluids, they form a mean of intercellular communication and of transmission of sensitive molecules such as nucleic acids, lipids and proteins between cells [12] Proteins embedded in the

membrane of exosomes are also involved in the interaction with recipient cells, and can identify the physiological and pathological states of their parental cells or indicate their preferential target cells or tissues [13] For instance, alterations in protein profiles of exosomes in plasma correlate with pathological processes of many diseases including cancer[14]. In neurodegenerative diseases, exosomes have been implicated in the trafficking of prion proteins, such as prion protein, β-amyloid which is the main reason behind Alzheimer diseases (AD) and tau [15] Researchers have come up with different biomedical techniques to analyze these exosomal proteins. Once the vesicles are purified, it becomes easier to detect and identity their proteins and use them to answer different biological questions.

In recent years, researchers have been using different methods and software packages that are available to analyze microarray data and deal with some of the issues that can be encountered during the image analysis process such as ScanAlyze (Buckly 2000), and GenePix Pro6 (Axon Instruments Inc. 2004), but the problem is that these software and others tend to require a certain amount of human intervention to achieve a good result that enables the biologist to come up with a more accurate and logical analysis of the data [7]

The Micro ana Nano bioengineering laboratory at McGill University has been using Array Pro Analyzer as a software to analyse different types of microarray images in order to characterize exosomes. Array-Pro Analyzer is an indispensable tool for the nascent microarray and high-throughput screening research that is fueling the genomics, proteomics, drug discovery, and bioinformatics revolution. The software is well designed for microarray gene expression, and sequencing analyses, and it has the ability to detest low-level expression using extensive background and signal options to optimize results. It can characterize data more precisely through grouping by cells, grids, and images[16]. Despite its efficiency, the software also has some downsides, for example it requires manual input to identify the spots from the background signal, making the process of gridding, which is the basis of the whole image analysis manual, and this may cause a lot of inaccuracies. Besides, this can be time-consuming, especially if we are dealing with a lot of data to process. Moreover, the software is only able to provide numerical values without any interpretation, and the user will have to look at the raw data and intellectually make sense of it. Finally, Array Pro and other microarray data analysis software are ‘black boxes’ that produce the final microarray image quantification results without showing any preprocessing or analysis steps, which limits the ability to diagnose and correct image analysis errors when they occur.

Therefore, we noticed a need to come up with a new pipeline to analyze microarray images from the Micro and Nano bioengineering lab in a more accurate and convenient way, while optimizing the whole image analysis process. Array-Pro analyzer is going to be used as one of the ways to test the efficiency of the developed pipeline Besides, given the importance of exosomes in medical settings and their relevance to the type of EVs research and analysis conducted at the lab, those molecules are selected as a main model to test the pipeline developed in this work, by using images of antibody microarrays, which is the most polyvalent approach within the multiplexed immunoassay technologies, as they allow the detection and study of protein function, pathways and other characteristics[17] The analysis method developed in this work, will also be validated on different types of microarray data to demonstrate that the model used is universal.

Furthermore, it is known that microarray experiments usually provide huge amounts of data, that requires the right methodology and the scientific tools to understand and explore any hidden knowledge and uncover biological patterns The need to manage and handle large data sets, leads us to think about adapting data mining and machine learning approaches on the microarray data we will be using for this project and assess whether those approaches add any scientific value to our analysis[18, 19].

The pipeline consists of developing several algorithms for automatic gridding, segmentation methods based on machine learning and thresholding, and intensity calculation. In summary the objectives of this project are the following:

- Develop an automatic gridding method, and measure its parameters such as time of processing, and accuracy.
- Develop an automatic segmentation based on thresholding and assess its scores.
- Explore the effect of machine learning on the segmentation results, and on research on microarray data in general.
- Understand the mechanism of functioning of microarray image analyzing softwares.
- Calculate the spots intensities and expression levels.
- Measure the statistical parameters to determine microarray experiment quality.
- Demonstrate the universality of the developed pipeline by validating used pipeline on different microarray data.
- Understand the different noise sources in microarray images.
- Test and evaluate the performance of the proposed pipeline by comparing quantification results with ArrayPro, and other data.

Having a better approach for analyzing microarray images can improve our understanding of the many software used to analyze microarray data and offer a universal way for more accurate analysis of microarray data and EVs, and therefore a better understanding of several illnesses including tumor invasion.

Chapter 2: Literature review

2. Introduction

Array technology is an essential tool to analyze data generated for multiplexed analysis of DNA, RNA, proteins and EVs. In this chapter, we define the microarray technology and go through the image analysis pipeline along with different approaches to improve the image analysis process. We also provide an overview to EV analysis since the pipeline being developed in this work uses exosomes a model to validate the different conclusions[20]. The importance of extracellular vesicles (EVs) in cell-cell communication has long been recognized due to their ability to transfer important cellular cargoes such as DNA, mRNA, miRNAs, and proteins to target cells. Among those EVs exosomes constitute potential biomarkers for cancers due to their abundance in biological fluids, ease of uptake, and cellular content. Therefore, the use of exosomes in diagnosis and prognosis can revolutionize patient care[21] Finally, we discuss machine learning techniques and how they could be applied to microarray data management.

2.1 Microarray data processing

A microarray is a powerful tool and is widely used in many research areas. For biologists, genetic research, understanding, and diagnosis of cancer and many other dangerous diseases. Once the experimental design of the microarray is set up, the next step will be to produce an image, where the microarray is scanned by a laser. The expression process consists of three steps: grinding, segmentation, and intensity calculation that we are going to explore in this paper.[22]. Figure 1 shows the general workflow of a DNA microarray experiment.

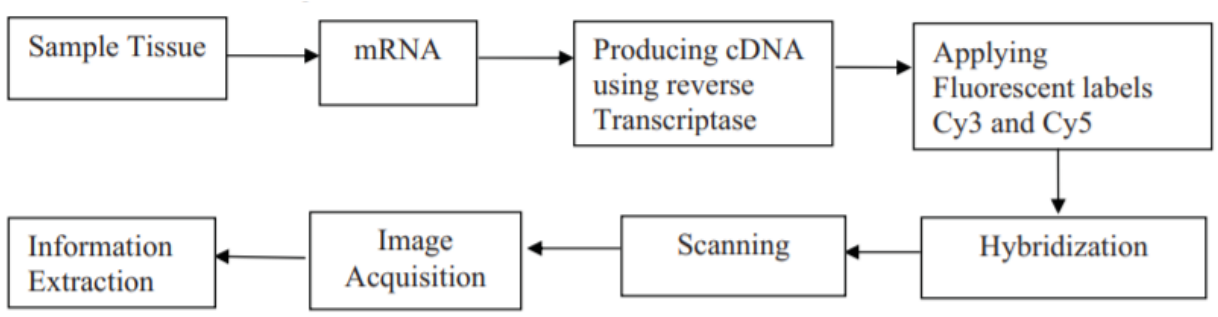


Figure 1 *Work flow of a typical DNA microarray experiment [24]*

2.1.1 Gene expression using microarray technology

In this section, we try to understand one important genomic application of microarray technology, along with its different steps. Microarray gene expression data have usually information regarding expression levels of the genes in certain tissue and cell, and it is useful to compare those expression levels to identity diseased genes which can lead to accurate production of a therapeutic drug for that particular disease.. This type of data can be leveraged in different medical studies and can

eventually help in the field of tumor and cancerous gene detection[23]. The microarray technology has the advantage of measuring the expression level of thousands of genes in a sample and in a rapid and efficient way[22]. A DNA microarray or DNA chip consists of a solid surface or glass slides, onto which thousands of DNA sequences have been chemically bounded. Each slide contains several sub-grids that are two-dimensional arrays of DNA spots. The technology enables the detection of the presence and abundance of labeled nucleic acids in a specific biological sample that is going to hybridize to the DNA on the array, and which can be detected via the label. Figure 2 details the steps of a DNA experiment.

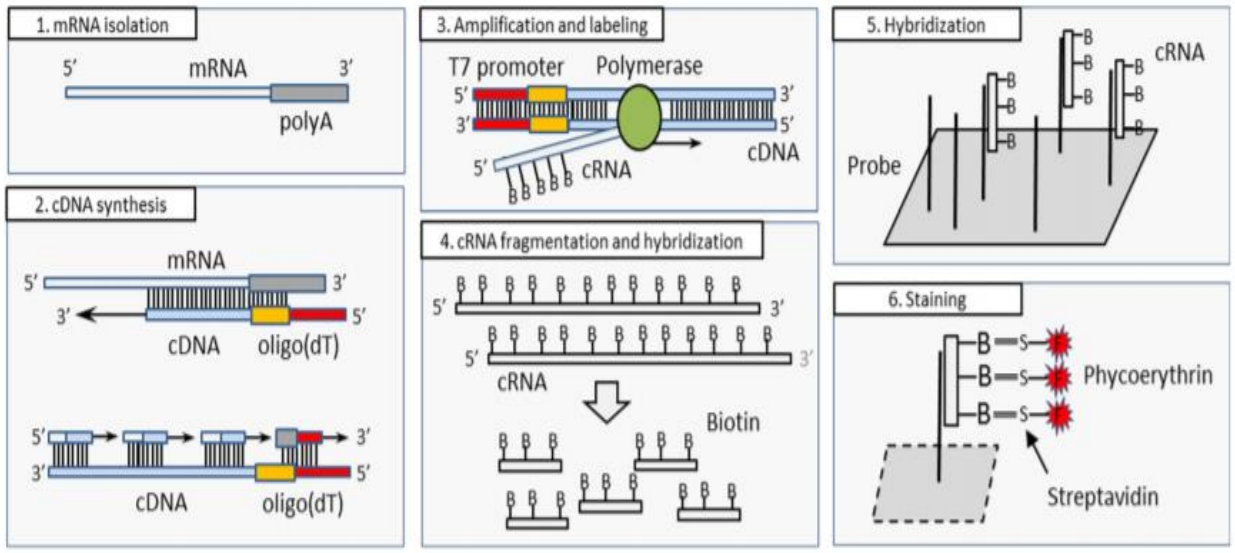


Figure 2 *Individual steps of a DNA microarray experiment. After isolation of sample mRNA (1) synthesis of cDNA chains by addition of oligo(dT) primers (2) cDNA amplification (3) fragmentation (4) Hybridization of CRNA with microarray probes (5) The final staining press (6) [24]*

In the microarray experiment ribonucleic acid (RNA) is first isolated from both control and experimental samples of the cells of interest. The extracted RNAs are then converted into cDNA. Usually, the labeled nucleic acids are derived from the mRNA of a sample of tissue. Typically, control and test RNA samples are processed on the same array using two different dye-tagged probes (e.g., the red fluorescent dye Cy5 and green fluorescent dye Cy3)[22, 25] Those two populations are usually labeled with fluorescent dyes such as Cy3 (green) and Cy5 (red). The mixture of these samples is then hybridized to a glass slide that will be later scanned with red and green laser. The relative difference of gene expression between the two sources is shown by the observed difference in fluorescence between the two-color channels, which will give a scanned array image that is going to be used to derive data for analysis. The images that are obtained from microarray experiments are usually represented as an RGB image with blue being zero. The images have several blocks that are called sub-grids that have several spots with varying intensities. Those spots are placed in rows and columns and are also spaced such as shown by figure 3[25].

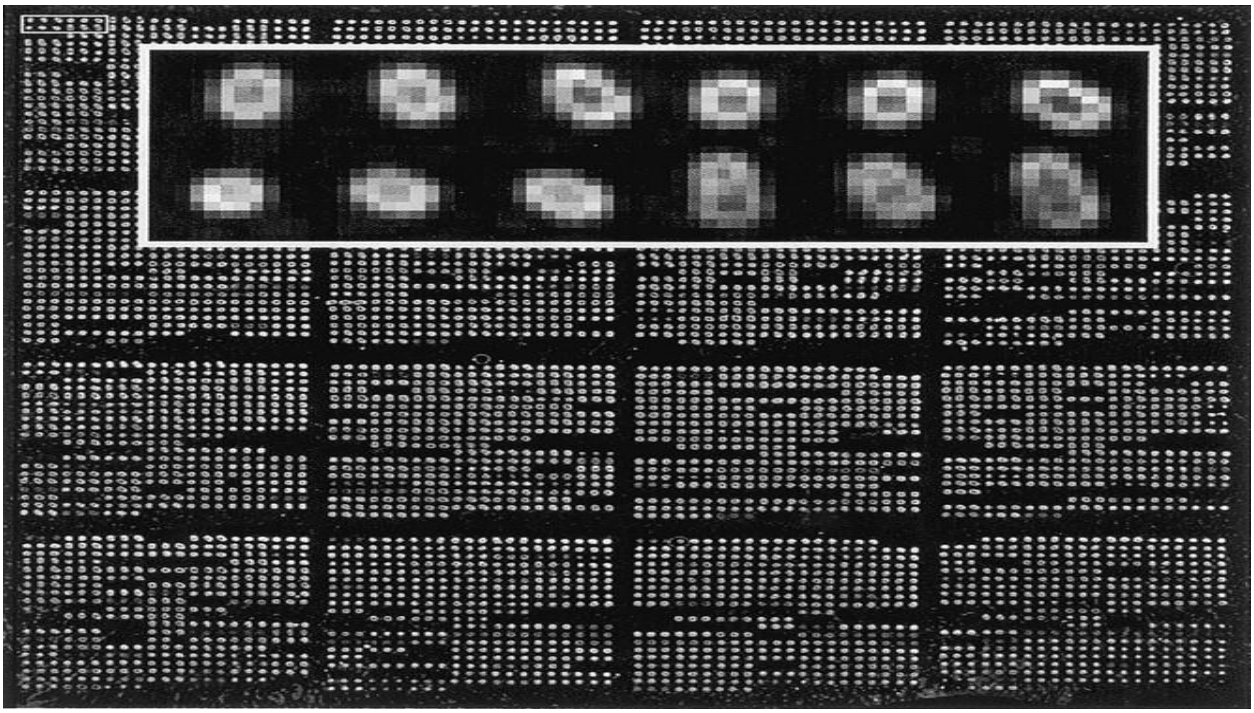


Figure 3 *A typical microarray image, containing 6000 spots, arrayed in a set of 4 × 4 subarrays each containing 21 × 18 spots. The spots are printed with ~130 μm center-to-center distances.[28]*

2.1.2 Scanning

Once the hybridized arrays are ready, they need to be passed by a scanner to measure the fluorescence intensities of the different spots related to each sample. The original images that are scanned are stored as a pair of 16-bit TIFF files, ranging from 2.5 to 20MB in size. Usually, one file corresponds to the testing sample and the other to the reference sample. The spots are separated based on their individual size, so that the spots can be hybridized, washed, and scanned. These principles of microarray production, lead to the fact that spots are not of the same size and some spots have high or low intensity[26].

2.1.3 Pre-analysis issues of microarrays

Before analyzing microarray data, it is common to encounter several pre-analysis issues that need to be addressed mostly in the preprocessing steps. These challenges may influence the stability and reproducibility of the microarray platform and affect the whole process of the image analysis[27]. The following table summarizes the most common pre-analysis issues encountered while working with microarray platform:

Table 1 *Summary of most common pre-analysis issues for microarray images along with their description*

The pre-analysis issue	Description
Image reading	Usually, the microarray image is stored as J-PEG file, and that's why it is usually bigger than the screen. Scaling down the size of the image is necessary for the analysis, and this can be done using different functions in Matlab for example such as imshow( ) function [28]
Cropping	It is essential to use a programming function such as imcrop( ) to crop a particular section of the microarray image and focus on it [29]
RGB separation	The microarray image is stored in a RGB format, to study the red and green plans, since each plan is represented with a color [30]. A simple indexing step is needed. The shape of the spots is always the same.
Greyscale conversion	Once the microarray image is cropped, the RGB image may go through a conversion to the grayscale to be prepared for spot finding and an easier gridding [31]
Global thresholding	Distinctive distribution of brightness of objects and background pixels require a global thesholding to the entire image so that all spots are detected equally[32]
Local thresholding	Applying global thresholding for the brightness issue may not be enough, since some weak spots can still be missed, that's why a local thresholding is also required, and the bounding boxes can be used to determine local threshold values for each spot [33]
Transformations	Before the analysis and comparison between the expression data extracted from the spots, a number of transformations must be conducted to remove low-quality measurements and adjust the measured intensities to improve the study[34].

2.2. Gridding

Microarray image gridding is the first step in the image analysis process and it is essential to determine 2D image coordinates of the spots by putting each spot into an individual comportment.[35] Being able to accurately identity the exact location of a spot in a specific sub

grid is an important step for a more smooth image analysis process. An error in this task can create other issues for analysis and may alter the truthfulness and exactness of the study [36] Usually a gridding technique should be able to grid images that include spots of various shapes, sizes and intensities while being robust to noise and artefacts introduced by the experiment [37] There are in generally three types of gridding depending on the level of human intervention required.

- Manual gridding
- Semi-automatic gridding
- Automatic gridding

Figure 4 describes a microarray image before and after the gridding process. The next section explores the different gridding methods that are common.

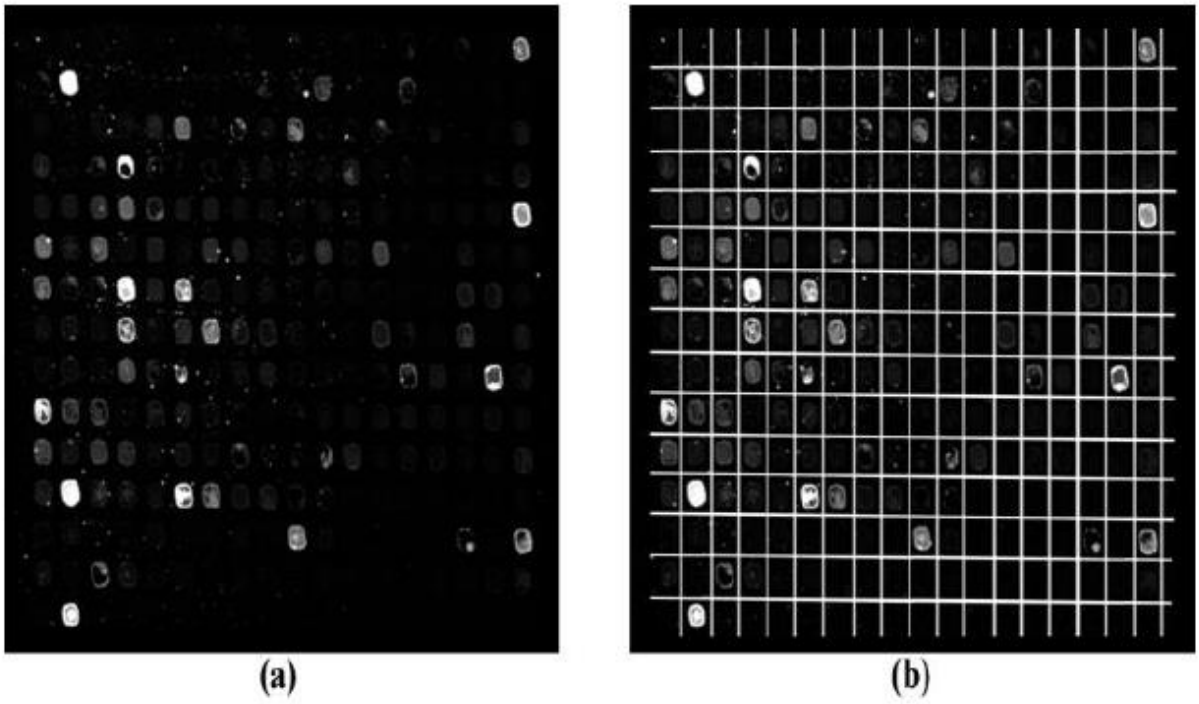


Figure 4 *An associated-grid of a typical microarray image: (a) before gridding and (b) after gridding [41]*

2.2.1 Subgridding

This method is based on the identification of the sub-grids by identifying first the probable rotations of the image, and applying the radon transform, and smoothing the sums of the horizontal or vertical pixel intensities. One of the downsides of this technique is that it is not able to detect the accurate count of associated grids automatically. Another technique for associated-gridding is to carry out a sequence of phases with rotation to identify the rotated spots and evaluates the

horizontal or vertical sums of the uppermost and lowermost parts of a particular microarray image, and this helps in detecting the rotated direction with respect to either of the axes[38]

2.2.2 The bayesien model

This method is based on the use of a the Bayesian probabilistic model which is becoming ever more popular in applied and fundamental research, including image processing to locate the exact location of spots[39]. It uses the posterior probability model to correctly adjust the grid spots. The technique can be done in two steps: first, it is essential to use an orientation matching and radon transform to generate a preliminary grid. Once this is done, the grid nodes are adjusted based on local spot deformations. In this technique, there is a still a possibility of misalignments between the actual sub-grid and the matched grid, and this issue should be resolved before measuring the intensity level of each spot [38].

2.2.3 Genetic algorithms

A genetic algorithm (GA) is a powerful tool that suitable for optimization problems. A typical GA searches for the optimal solution is required. First, it creates a finite number of potential solutions encoded as alpha-numerical sequences called chromosomes. Chromosomes represent a possible solution for the gridding process. These chromosomes constitute an initial population Pop. Another population of chromosomes is created. This evolutionary cycle from one population to the next continues until a specific termination criterion is satisfied, such as meeting the minimum required gridding parameters. Each potential solution is evaluated using a fitness function. This function measures the quality of a gridding solution that those chromosomes represent. Therefore, the essential elements of the GA are chromosome representation, chromosome evaluation, the evolutionary cycle, and the termination criteria.[40] This kind of algorithm can differentiate between the different grids and identify the location of the spots. The method consists of finding the lines that separate between the spots. The population of chromosomes are encoded as triplets that have three real values: the first two values are the y coordinates, and the third value is the space between neighboring spots. A separating line should be contained in an empty area between adjacent spots. [38] The method seems to be efficient for gridding, even if the image seems to be contaminated by noise or artifact[41] and even though genetic algorithms are widely used in the estimation of expression profiles from microarrays data. This kind of approaches are unable to produce stable and robust solutions suitable to use in clinical and biomedical studies[42]

2.2.4 Optimal multilevel thresholding gridding

This method showed success in microarray image gridding, and particularly in DNA images with different configurations. It used thresholding without input parameters. It locates the spots in a recursive way for each possible threshold, and then uses an objective function to assess to set the best number of thresholds. This method showed success in microarray image gridding, and particularly in DNA images with different configurations. It used thresholding without input parameters. It locates the spots in a recursive way for each possible threshold, and then uses an objective function to assess to set the best number of thresholds. It is essential to find thresholds in typical DNA microarray image. to do so, this method relies on the use of radon transform as a preface stage. To find the amount of spots in each sub-grid the same procedure is followed along with using an index of validity.[38]

2.2.5 Grid alignment algorithm

The method starts by locating a 2 D array of rows and columns of the image, along with its rotational and translational offsets. The spot center is represented by the intersection between two rows and columns. This method however performs badly when it comes to noise removal. Figure 5 shows the diagram of the different steps of the algorithm[5]

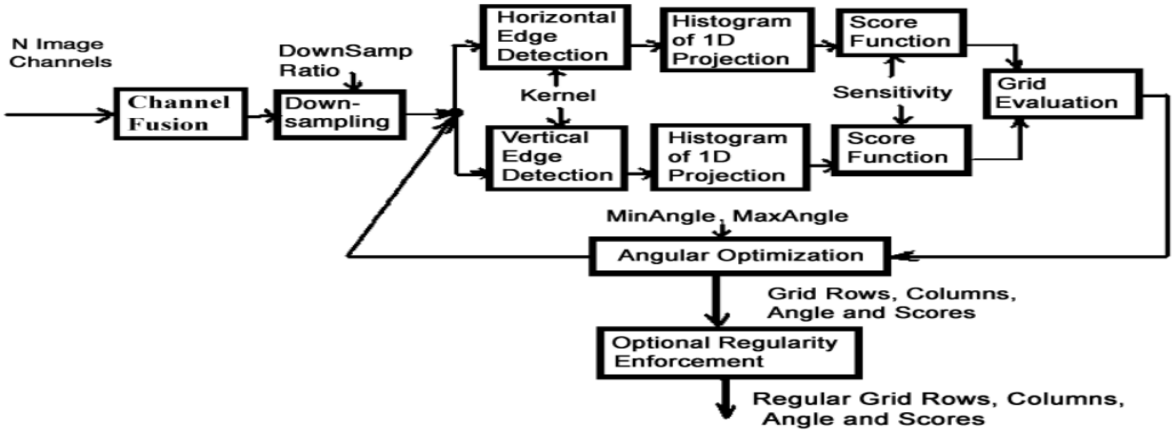


Figure 5 *Grid alignment algorithm diagram detailing the different steps [47]*

2.2.6 Support victor machine

This type of grinding is based on the use of a common machine learning algorithm called the support victor machine (SVM). The technique relies on the use of a soft-margin SVMs to predict the lines of the microarray grid by finding the maximum between the lines and the spots. In order to maximize the margin between the spots and the lines, it is essential to automatically set the separation line between consecutive rows and columns. Once the separation step is done, a detection step is followed by selecting the place of the microarray grid that has specific properties before filtering out any anomalies and artifacts. Finally, the rest of the spots are automatically split into rows and columns by calculating the distance between two consecutive rows and columns of spots, as well as the image rotation angle [5]

2.2.7 Otsu method

This method consists of converting a greyscale microarray image into a binary image. Once the conversion is done, the pixels are grouped into two separate categories: background and foreground image. It is essential to carefully use a threshold value to get the best separation possible. The method takes into consideration the removal of unwanted parts using an edge processing step, before the grid can be calculated, and in case of dealing with a colour microarray image, converting it to the greyscale should be the first step to be done[5]

As we have seen in this section, there are several gridding methods that require manual to semi automatic human intervention, and this can delay the image analysis process, and cause some

accuracies issues. In this project, we will be presenting an automatic gridding method, and we will assess its effectiveness by measuring the processing time that the algorithm requires to perform the gridding operation, as well as the accuracy of correctly gridding and locating the spots. This will be described in more details in chapter 3. The following part of this chapter addresses the segmentation phase of the microarray images.

2.3 Segmentation

Medical applications are essential to the healthcare industry, and therefore, it requires correct and fast segmentation associated with medical images for correct diagnosis, but the process of using the right segmentation method remains challenging[43] Once the microarray image gridding step is done, the next step in the microarray image pipeline is the segmentation of the spots, which is still a problem because of the variations of spots qualities such as spots shapes and sizes[44] Segmentation is defined as the segregating of the microarray image into multiple fundamental fragments, So that the image pixels are separated in two classes, whether they belong to background or spots.[45, 46]. Researchers have come up with several methods to segment microarray images, among these techniques, we have the followings:

2.3.1 Fixed circle segmentation

This method fits a circle to all the spots in the image. The method is easy to implement and gives good results when applied to a microarray image that has spots of similar size and circular shape. In this technique, and in the case where the background affects the foreground value, and the background value can be estimated then, it is possible to use a very large fixed diameter to cover the entire spot in this situation [47]

2.3.2 Adaptive circle segmentation

This method looks closely at each spot. It considers the shape of each spot as a circle, where the center and the diameter of the circle can be estimated. The first step consists of estimating the center of each spot, and then the diameter of the circle will be adjusted [48]

2.3.3 Thresholding segmentation

In this method, the first thing to do is to apply a single threshold level to the entire image in order to detect all the spots equally. One of the challenges that can be encountered while applying this method is the large differences in the spot brightness, Therefore, the intensity values of the spots need to be transformed to logarithmic space or go through other type of image processing transformations, in order to boost the effectiveness of the technique. Using different ways of applying the thresholding process using combinations may boost the segmentation process[48].

2.3.4 Seeded region growing

Seeded region growing (SRG), is robust, rapid, that does not require any tuning of particular parameters. The algorithm could be applied to various images. It can be particularly relevant for semantic image segmentation that requires high knowledge of image components in the seed selection procedure. One of the algorithm downsides, is the problems of automatic seed generation and pixel sorting orders for labeling. However, there are still several ways available to improve the algorithm performance[49].

2.3.5 Mann-Whitney

This method can be used to compute a specific segmentation threshold iteratively with Mann–Whitney test, which is a non-parametric test used to assess for large differences in a scale or ordinal dependent variable using a single dichotomous independent variable, and it has several similarities to the statistical non-parametric t-test[4, 50] The method consists of providing a hypothesis test and confidence interval to quantify the significance of observed differences in expression ratios. So basically, the probability density of the ratio and the maximum-likelihood estimator for the distribution is derived, and an iterative procedure for signal calibration is developed[51].

2.3.6 Model-based segmentation (MSB)

The method usually reports better stability than a fixed-circle segmentation method or the seeded region growing method. The first step of this method is to apply model-based clustering to the distribution of pixel intensities, using the Bayesian Information Criterion (BIC) to choose the number of groups up to a maximum of three, then next find the large spatially connected components in each cluster of pixels, and therefore this method can be considered as clustering model of pixels and extraction of connected components to perform segmentation[52].

2.3.7 Matarray (MA)

This method is usually presented as an integrated image analysis package for cDNA microarray technology. It relies on an iterative algorithm that use both intensity characteristics and spatial information of the grid spots to achieve the segmentation. The technique defines five quality scores for each spot to record irregularities in spot intensity, size, and background noise levels. These are the important parameters of the segmentation procedure [53]

2.4 Intensity calculation

Microarray technology has given a way to quantify the simultaneous expression of a large number of genes, and the approach is dependent on reproducible, accurate quantitation of spot intensities, which is the last step of the image processing pipeline.[54, 55] We usually, crop a particular area of the microarray image, we measure the intensities of the spots in that particular cropped region, and then we compute all the intensities, several transformations could be applied to better interpret the meaning of the immense amount of biological information formatted in numerical matrices by the microarray[28]. The following table summarizes the main transformations along with their advantages and shortcomings:

Table 2 *Type of intensity transformations along with their advantages and inconveniences*

Transformation	Strength	Weakness
Logarithmic	Dynamic range compression	The compressed value needs to be brought back to its initial value in times of display

Gamma	Decent results if the value of the parameter gamma doesn't exceed 1	Limited if terms of improvements: unless the value of gamma is superior to 1, the results remain the same.
Contrast stretching	Increasing the value of E, improves the results	Limited by thresholding techniques for binary images

2.5 Extracellular Vesicles (EVs): an overview

Extracellular vesicles (EVs) are lipid bound vesicles secreted into extracellular space and can be categorized into three main subgroups: exosomes, microvesicles (MVs), and apoptotic bodies. These three categories of EVs differ in size, origin, molecular composition, and function. Exosomes are the smallest EVs, ranging from 30-150 nm in diameter, while MVs range from 100-1000 nm and apoptotic bodies range between 50-5000 nm in diameter[56]. Exosomes are derived from early endosomes and are enriched in proteins from the endosomal sorting complexes required for transport (ESCRT) pathway, while MVs and apoptotic bodies are formed at the plasma membrane[11]. In this section we present the different types of EVs by describing their size, origin, composition, biological purpose as well as their applications and use clinical settings. Figure 6 summarizes some important information about EVs that we mentioned before,

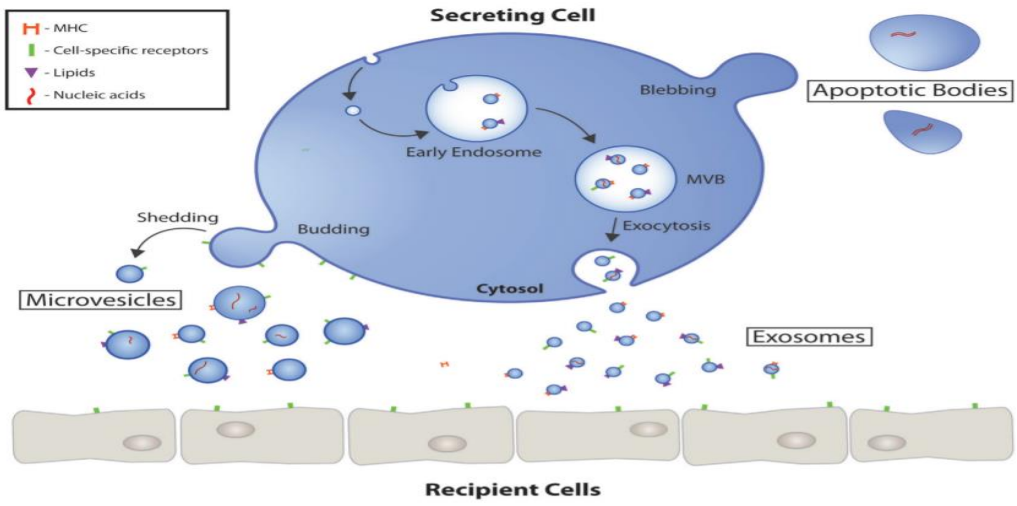


Figure 6 *Overview of the different types of EVs and their origin [66]*

2.5.1 Apoptotic Bodies

Apoptotic bodies are released from dying cells into extracellular space, and their size range from 50 nm up to 5000 nm. Once the cell contracts, there is an increase in the hydrostatic pressure which lead to the separation of the cell's plasma membrane from the cytoskeleton and therefore apoptotic bodies are formed. These vesicles are made of intact organelles, chromatin and small amounts of glycosylated proteins, and their proteomic profiles[11]

2.5.2 Microvesicles

Mvs form by direct outward budding, or pinching of the cell's plasma membrane, their size range from 100 nm to 1um in diameter. As for their composition, it is thought to require cytoskeleton components such as actin and microtubules, along with molecular motors and fusion machinery. Since MVs form by an outward budding of the cell's plasma membrane, it is understood that MVs contain mainly cytosolic and plasma. Moreover, there are several proteins that were identified in MVs that include cytoskeletal proteins, heat shock proteins, integrins, and proteins containing post-translational modifications, such as glycosylation and phosphorylation[11] As for the biological purpose of MVs, it was concluded that these EVs are involved in cell–cell communication between local and distant cells, and their ability to alter the recipient cell was well demonstrated. The, particularly of MVs along with other EVs, is their ability to package active cargo (proteins, nucleic acids, and lipids) and deliver it to another cell, neighboring or distant, and alter the recipient cell's functions with its delivery. The fact that cancerous cells also package their active machinery in EVs and transport it to other cells, just like other healthy cells function, open the door for a better understanding of cancer development and progression and provide better therapies option[11]

2.5.3 Exosomes

Exosomes are a subtype of EVs that secreted by all cell types and have been found in plasma, urine, semen, saliva, bronchial fluid, cerebral spinal fluid (CSF), breast milk, serum, and other body fluids. They are formed by an endosomal route and are typically 30–150 nm in diameter. Exosomes are formed by inward budding of the limiting membrane of early endosomes, which mature into multivesicular bodies (MVBs) during the process. MVBs are usually either sent to the lysosome to be degraded along with all of its components or fused with the cell's plasma membrane to release its content, including exosomes, into the extracellular space. The regulation of MVB and exosome formation and release is through the endosomal sorting complexes required for transport (ESCRT) pathway, and it is understood that the formation of MVBs can be stimulated by growth factors and the cell adjusts its exosome production according to its needs[11]

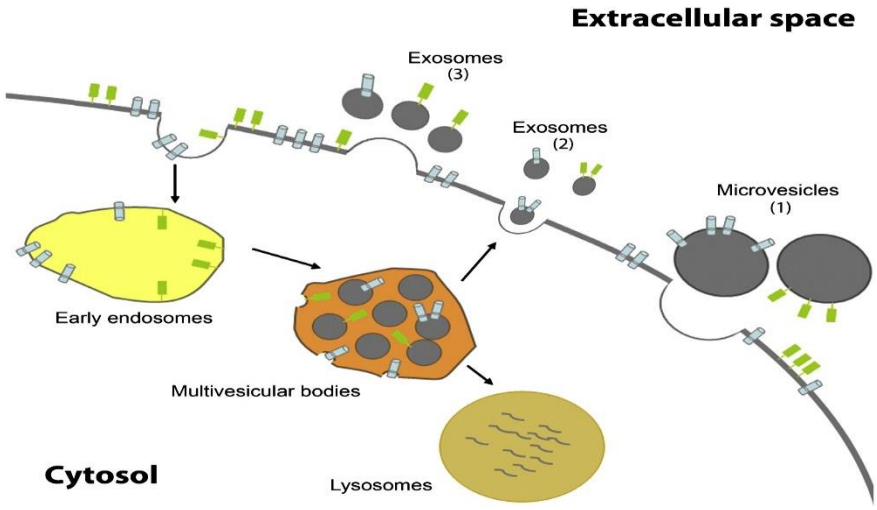


Figure 7 *Process of formation of Extracellular Vesicles EV[67]*

Since exosomal formation and MVB transportation are regulated by ESCRT proteins, these proteins and its accessory proteins (Alix, TSG101, HSC70, and HSP90) which are called " exosomal marker proteins" are expected to be found in exosomes regardless of the type of cell. [11] The main biological function of exosomes is to participate in cell–cell communication, cell maintenance, and tumor progression. In the nervous system, exosomes help in tissue repair and regeneration by promoting neurite growth and neuronal survival. They also contain pathogenic proteins, such as beta-amyloid peptide that is the main reason behind Alzheimer's disease (AD). Another common interest in exosomal research is in studying their ability to act as carriers of biomarkers for different type of diseases[11].

2.6 Exosome array-based capture and detection techniques

This section explores the different ways of the use of array technology to capture and detect exosomes for different medical applications.

2.6.1 Surface Plasmon Resonance Imaging

In a study by Yang et al. presented a method called iPM to detect, visualize and analyze exosomes using interferometric plasmonic imaging. The technique is based on the Kreschemenn configuration, where a later light is collimated and illuminated a AU chip at a highly inclined incident angle, stimulating the surface plasmons (SPs) on the AU surface, the reflective light along with the light scatted by the object will be then collected and imaged onto a CMOS camera. The capability of iPM in sizing and tracking single exosomes enables the quantitative study of membrane fusion between exosomes and liposomes, as well as the exosome–antibody interaction[57]. Figure 8 shows the main principle of surface plasmon resonance imaging.

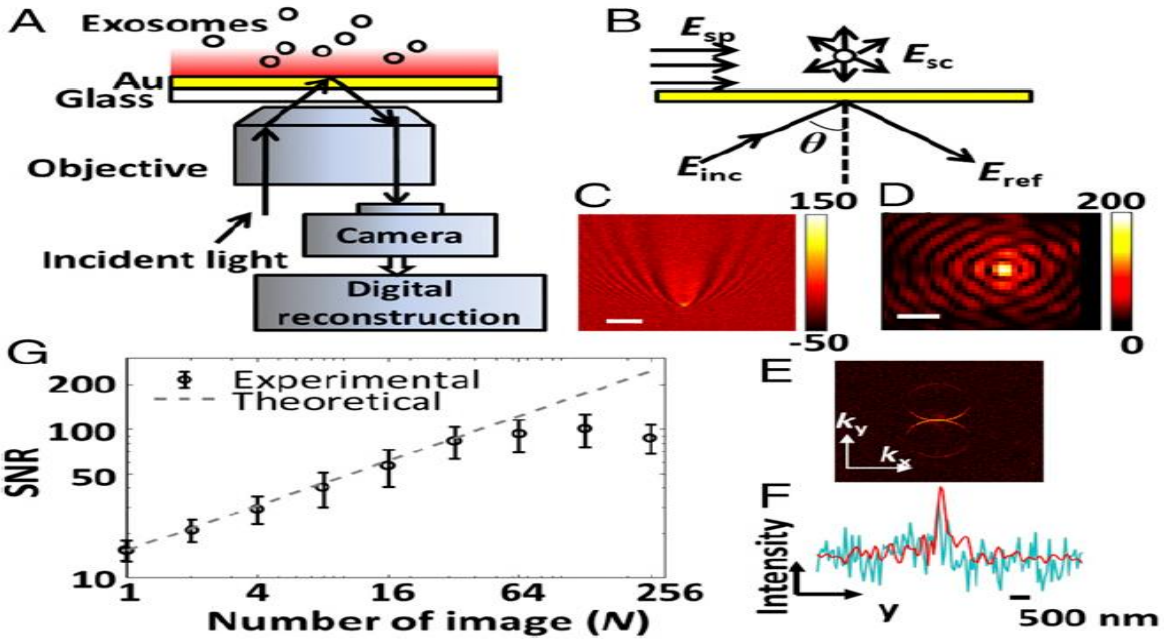


Figure 8 *The process of capturing and detecting exosomes using surface resonance imaging. (A) Schematic of iPM. (B) Interferometric scattering model of iPM. (C) Images of a 100-nm silica nanoparticle without and (D) image-reconstruction process. (E) k-space image by taking 2D-FFT of C. (F) Longitudinal intensity profile across the particle in C, and D (red). (G) SNR in iPM detection of 100-nm silica nanoparticles. [57]*

2.6.2 Protein microarray technology

A highly sensitive extracellular vesicle EV array that has the ability to detect and phenotype exosomes along with other extracellular vesicules in a high-throughput manner was presented in literature[58]. In the research conducted, the team has used a cocktail of antibodies against the tetraspanins CD9, CD63 and CD81, in order to solely detect the captured exosomes, which helped in excluding other types of microvesicles. In this research, two kinds of microarrays have been used. The first one was made of similar spots containing several exosome markers such as anti-CD9, CD63, and CD81. While the other array was made of 21 capture antibodies on different spots. For the second microarray, there was a panel of antibodies against 21 different cellular surface antigens and cancer antigens. For each donor, there was considerable heterogeneity in the expression levels of individual markers. Besides, the protein profiles of the exosomes (defined as positive for CD9, CD63 and CD81) revealed that the expression level of CD9 and CD81 was equal in the 7 donors, which made the team question the use of CD63 as a standard exosomal marker, because its expression level was very low.

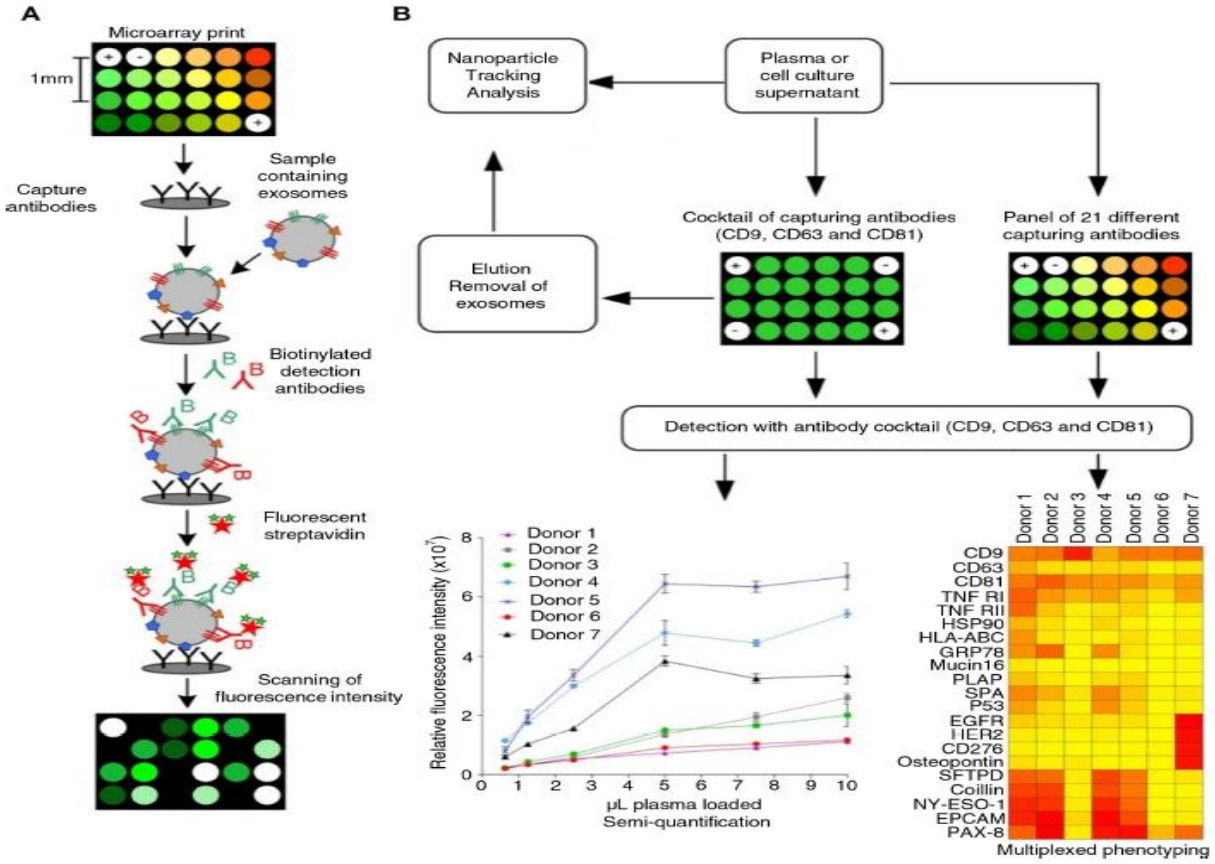


Figure 9 *Protein microarray for exosome characterization. (A) Multiplexed exosome protein analysis using the EV array. (B) EV Array workflow. Two distinct microarrays are used, one for semi-quantification and one for protein analysis; exosomes capture for semiquantification.*

2.6.3 Detection by interferometric imaging

A microarray-based solid phase chip was used along with a single particle interferometric reflectance imaging sensor (SP-IRIS) in order to achieve multiplexed phenotyping and digital quantification of various populations of individual exosomes (>50 nm) that were captured and characterized directly from a very small human cerebrospinal fluid volume (hCSF). The method enables the capture of other nanoparticles similar in size to exosomes, using antibodies that are directed against tetraspanins, which will help in improving the diagnosis of different disorders [2].

2.7 Introduction to machine learning (ML) and data mining

Machine learning has become a pivotal tool for many projects in computational biology, bioinformatics, and health informatics [59] In this section we provide an overview of the most common machine learning algorithms and some of the technique to handle data using machine learning such as feature extractions and so on. Figure 10 shows the general model of ML.

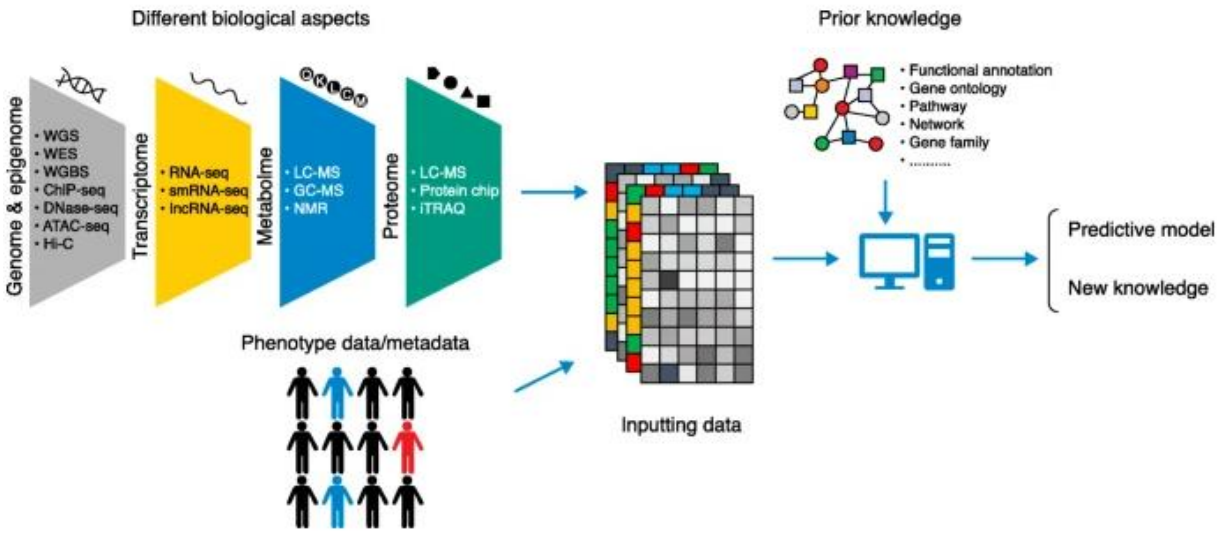


Figure 10 *Machine learning model applied to biological applications [60]*

2.7.1 Common machine learning algorithms

Machine learning algorithms are separated into supervised methods and clustering or unsupervised methods. The first one consists of designing a predictor that uses samples that have known class labels as training data to predict the class of a new testing sample, and we call this process classification when the output label is discreet or categorical, while we call it regression when it is continuous. [61] In the clustering algorithms, there are no outputs to predict. Instead, we look to find occurring patterns or groupings within the data[62] In more details these are the class of machine learning algorithms:

1. ***Classification:*** High-throughput technologies such as DNA, RNA, protein, antibody and peptide microarrays are usually used to distinguish between patterns in diseases, drug treatment and other applications, and this requires the training of a classification system or a classification algorithm to achieve this task, by gathering large amount of data that can be used to feed the classification system [63]

2. ***Regression:*** Regression is often used in cancer biology, to predict of the time for a tumour to recur after surgery. The data used for that represents the expression levels of several genes on around a hundred or so tumours, and the time of the recurrence of the cancer for each particular patient. The regression algorithm would aim to identify a small number of genes whose expression values can enable a correct and reliable prediction of the recurrence time [64]

3. ***Clustering:*** Clustering is a popular exploratory technique, and it deals mostly with high dimensionality data such as microarray data. The main purpose behind this unsupervised approach is to divide objects into different clusters or groups using some similarity parameters such as one minus correlation or Euclidean distance[65].

The following chart (figure 11) summarizes the most useful machine learning algorithms along with their learning method:

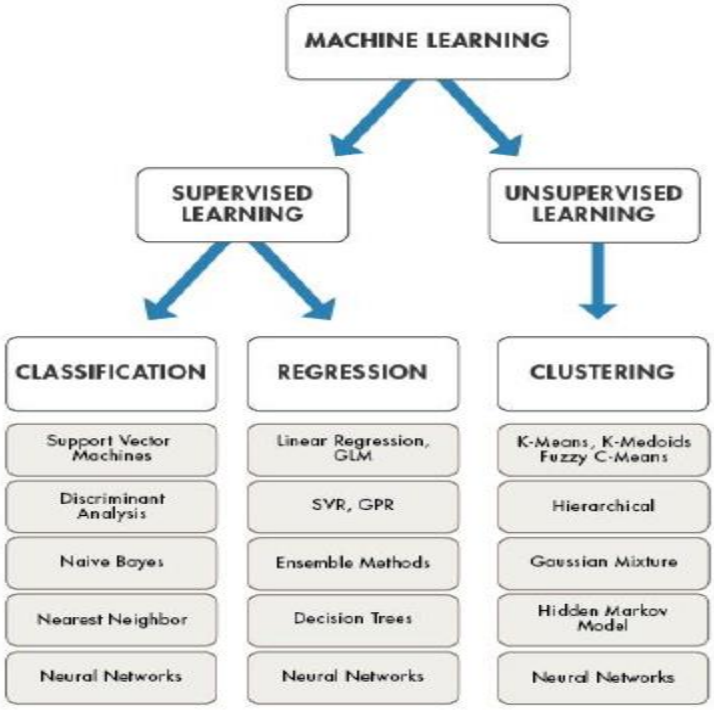


Figure 11 *Most common machine learning algorithms [66]*

Since we will be exploring machine learning as a way to segment our microarray images, we will be using k-mean clustering method, that we briefly describe. k-mean clustering is considered one of the simplest approaches when It comes to solving clustering problems [67]. The technique begins by initiating the centroid of the spots and the background, and then each pixel is assigned to the closest cluster. Once all the pixels are clustered, a new centroid is calculated and the process goes on, just like the figure below shows.[5] The mechanism of the algorithm is illustrated in figure 12.

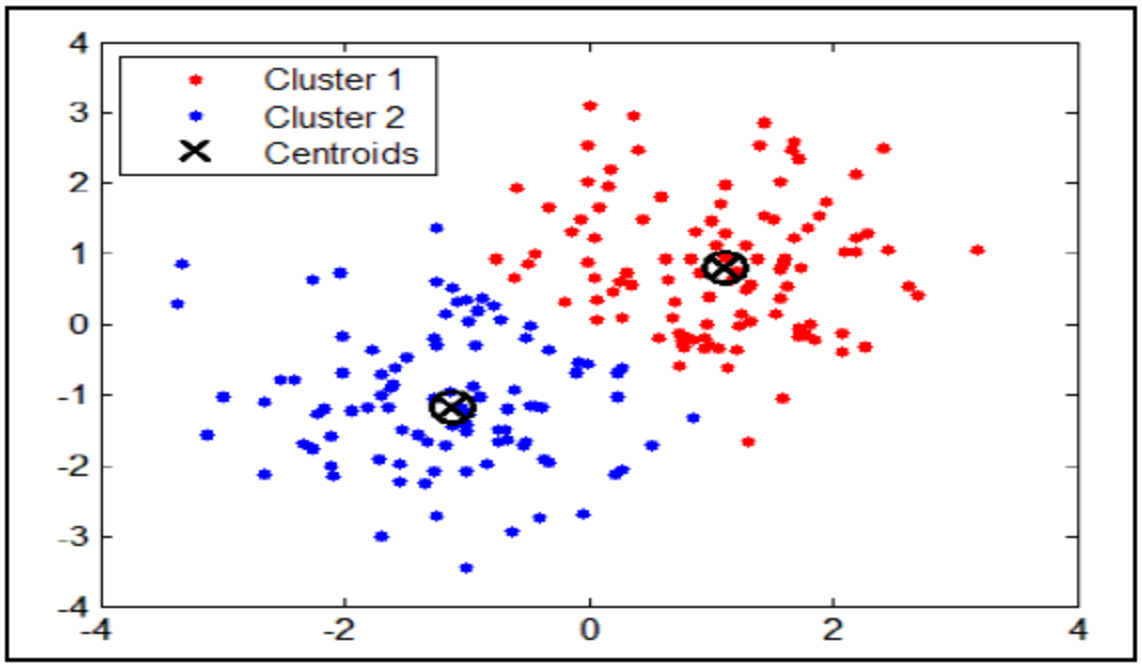


Figure 12 *K-means clustering algorithm mechanism [49]*

Machine learning algorithms require features to properly function. In general, we define a feature as an individual measurable property of the process being observed. Machine learning algorithms are able to solve classification problems using a combination of different sets of features[68] It is important to understand the notion of feature extraction and feature selection while dealing with large data.

2.7.2 Feature extraction applied on microarray data

When dealing with DNA chips, the gene expression data is often large and hard to processed, which requires its transformation onto a reduced representation of those genes, which is the main goal of feature extraction[69] There are many options for the implementation of feature extraction and gene clustering programs to help reduce the size of the data [70]

2.7.3 Feature selection applied on microarray data

Feature selection is currently a good choice for dimensionality reduction for microarray data, and they can help in making the data easier to analyze and translate into useful information. [68, 71] The main reason behind the feature selection algorithms is to remove any redundant or irrelevant feature from the inputs while maintaining or even improving performance[72, 73] Usually feature selection methods contain two essential aspects: the first one is to evaluate the

goodness of the feature subset and the second aspect is to search through the feature space[74].There are three types of feature selection techniques

Filter methods: this kind of methods perform the feature selection independently of construction of any classification model that requires those features to achieve the classification or clustering task [75] there are basically two types of filters that can be used: the first one is called the univariate feature filters, and they usually rank a single feature, while the other type of filters is called multivariate filter, and this one evaluates an entire feature subset, and feature subset generation for multivariate filters depends on the search strategy [76]. The subset is basically generated using specific type of searching algorithms that are categorized into three categories: exponential algorithms, which is an exhaustive optimized search that guarantees the best solution or the best subset[77], sequential algorithms, that constitute a family of greedy search algorithms that reduce an initial d-dimensional feature space to a k dimensional feature subspace where $k < d$ [78] and finally random algorithms, and this one use its randomness to avoid the algorithm to stay on a local minimum and to allow temporarily moving to other states with worse solutions and can provide several optimal subsets as solution. [79, 80]. The table below summarizes the most common search algorithms used to achieve feature selection:

Wrapper method: this kind of methods estimate the merit of a given subset of features or a set of attributes.[81] a wrapper approach suffers from high computational complexity in high dimension, but it has ability to increase the accuracy of a classification algorithm[82]

Embedded method: this kind of methods inject the selection process into the learning of the classifier [83] the goal of this method is to search for an optimal subset of features while building the classifier, and its main advantage the fact that it is far less computationally intensive compared to wrapper methods[84].

2.7.3 Machine learning segmentation methods

Image segmentation is an essential area and computer vision with applications such as medical image analysis, and image compression, among many others[85]. The following chart summarizes the most used algorithms when it comes to microarray segmentation. In this project we will be exploring the segmentation using K-means and we will be describing this algorithm and its details in the next chapters.

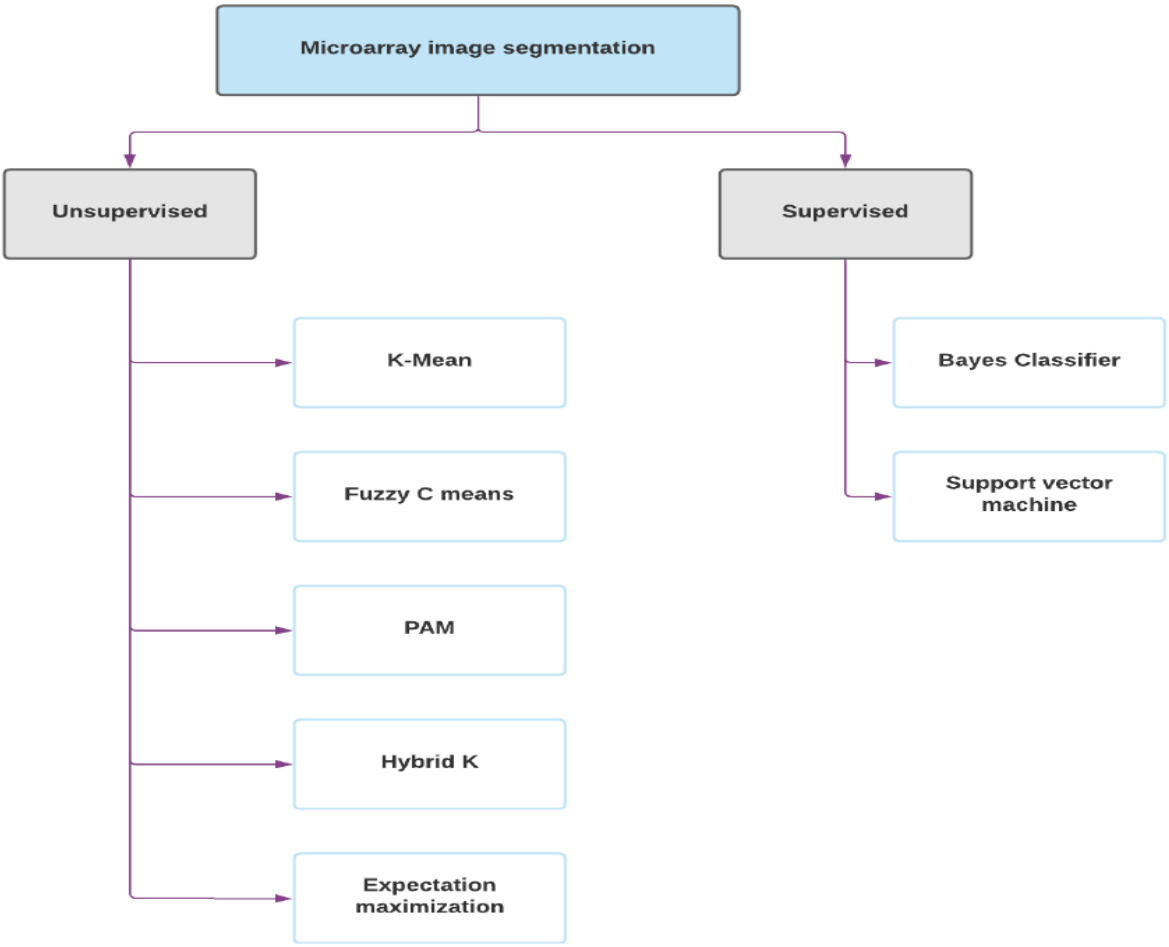


Figure 13 *Chart showing different types of machine learning based segmentation methods [86]*

Chapter 3 : Materials and Methods

3.1 Production of the microarray images

To test the efficiency of our pipeline, we choose exosomes and EV analysis as a model to validate our findings. Different real microarray data have been obtained from the Juncker lab. The microarray image used in this project were generated following an experiment that was conducted with the goal to simultaneously detect intra and extravesicular proteins using antibody microarray, Previous experiences from the lab have focused solely on surface and internal proteins. Assessing detection markers alone separately to study the impact of a combined detection, and if it is effective to use those multiple detections in analyzing co-expression values was also explored. In order to make sure two different proteins can still be detected in the same well, with minimal cross-reactivity, the following materials have been used:

- Mouse or rat antibodies for capture.
- Biotinylated antibody 1 + streptavidin-AF488.
- Rabbit antibody 2 + fluorescent anti-rabbit secondary-AF647.
- Separate wells for CFSE stain.

So basically, an assay of antibodies was inkjet-printed and on which EVs were incubated overnight, then fixed and heat treated. Once this preparation step is done, primary and secondary antibodies/streptavidin were flown in succession. The primary detection is used by a specific antibody for the target, while the secondary antibody/streptavidin is the labeled antibody that detects the first one. The antibody system used is very similar to ELISA system, except that the capture antibody and the detection antibody don't necessarily recognize the same target since we are dealing with EVs, for example we can capture based on CD63, yet detect based on CD81. In order to obtain microarray images from the experiment, we use 3 different slides, each one has the goal to target specific exosomal protein:

- Slide 1: Alix
- Slide 2: Individual targets
- Slide 3: Hsp70

For each slide 16 conditions were studied. Each condition represents a different pair of primary and secondary detection antibodies.

3.2 Features of the microarray data

Those microarray images were stored as TIFF files. The image we obtained have various kinds of noise and artifacts. In this project, we are going to rely on Matlab for data analysis and other computational tasks, given its high performance and power[87] Each TIFF file has two greyscale images per slide. Each of those images represent a channel, and that's how usually the scanner saves the images. The first channel is red, while the second one is blue for the purpose of our study. The images we obtained have been already previously analyzed using Array Pro analyzer by the lab. The results will be used as a benchmark to compare against the results we obtain using the proposed pipeline. They image we obtained have different scanning resolutions, and different noise types, which is a good opportunity in order to study the flexibility of the proposed pipeline

to analyze spots with different sizes and features. Figure 14 shows a sample of the type of microarray images we had from the previous experience that was conducted, while trying to target Alix protein:

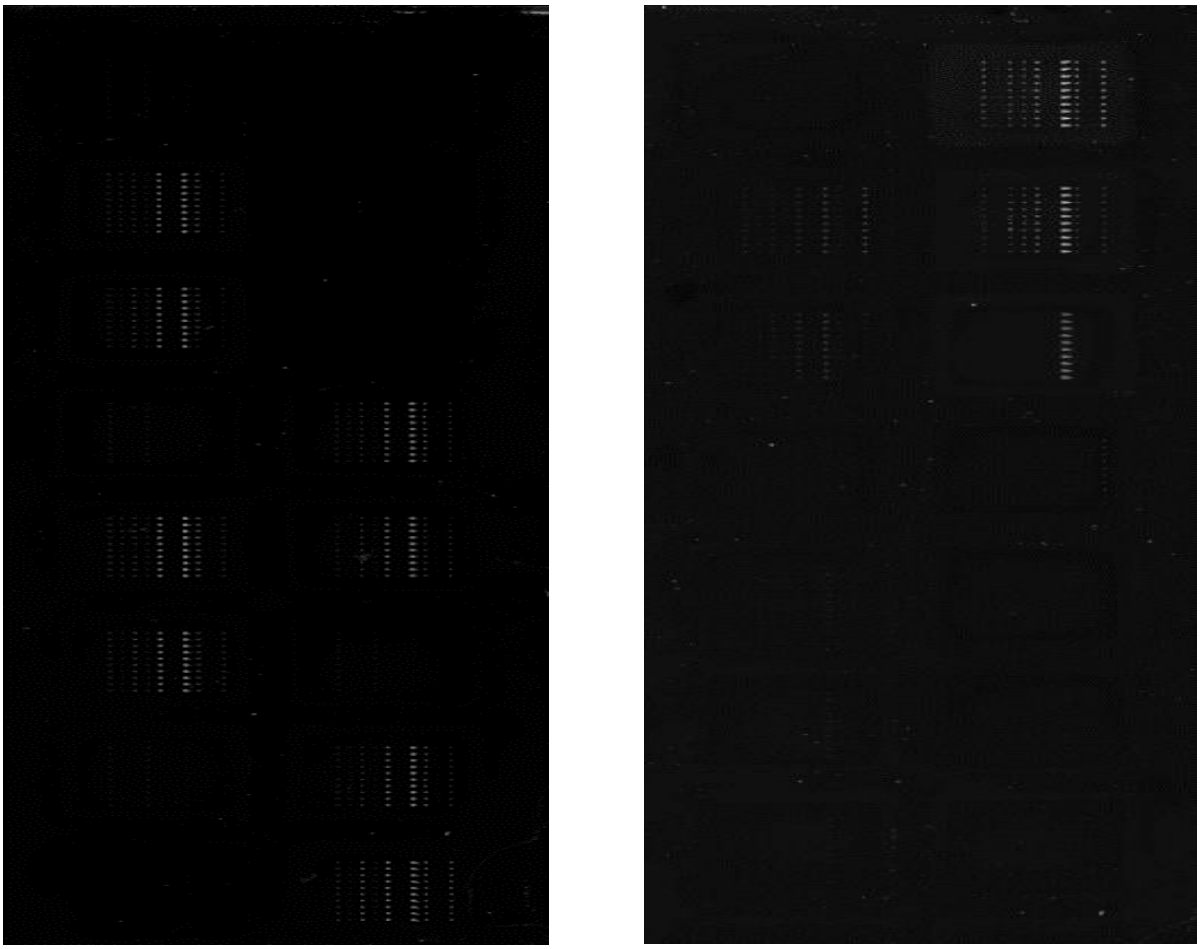


Figure 14 *Red channel (left) and blue channel(right) microarray image extracted from slide 1 of the antibody microarray experiment*

The microarray image in hands has 16 sub grids spaced, in each sub grid we have vertical lines of spots or columns. Each column represents a different capture antibody, while each subgrid represents one detection cocktail (one specific condition as described above). The pixel size is 5 µm. Figure 15 shows a closer look at slide 1 of the antibody experiment slide.

Figure 15 *Cropped view of a microarray image generated from slide 1 of the antibody microarray experiment*

A difference in brightness between the different lines in each particular sub grids, and across the entire microarray image was observed, and that's mainly due to having a different capture antibody from one column to the other among the subgrid. We also notice some weak brightness in some areas of the grid. Some subgrids look identical since they represent the same experimental condition, or they were conducted using the same antibody cocktail.

3.3 Preprocessing

Preprocessing microarray image is required before applying the gridding step. Preprocessing includes reading the image, cropping it, changing its size so that it can fit the screen and noise removal operations. In order to read the TIFF file, we will be using the image processing toolbox command of Matlab that treats an image as a matrix. MATLAB version 9.8 (R2020a) on a windows 7 platform was used for this project. The first step is to split the TIFF file after we read it, into two separate grey channels, before feeding them to the algorithm. In order to split the TIFF files into two separate pages, we use the open source software ImageJ, which is a great tool for data visualization and advanced image processing [88]. Splitting the TIFF file into two different channels (two different colors) will make it easier to process separately for the final stages of the image process such as intensity calculation and expression levels estimations, and It provides an intuitive and powerful ways to conduct analyses and make conclusions based on comparisons that might otherwise wouldn't be possible if we are dealing with the entire image at once[89].

To remove noise, we design a morphological filter based on the mathematical notion of structing elements. The denoising technique will be required for a smooth gridding, as it is going to be used to remove the background noise from the intensity profile of the image projection that we will be using to perform the gridding step. A top hat morphological filter need to be used, because this kind of filters can be a classical way to remove imperfections and provide a good information on the form and structure of the image being denoised, it mostly helps in removing

small objects from an image, while being faithful to the shape of larger objects in the image. A top hat transformation is a denoising process that computes the difference between the microarray image f(x,y) and its opening.[90, 91] An opening for an image f(x,y) by a structuring element is an erosion followed by a dilation[92]. The opening operator therefore requires two inputs: an image to be opened, and a structuring element[93]. Opening removes small objects from the foreground of an image, placing them in the background[94] In other words. Image opening manipulates the process of erosion and dilation to improve the quality of the image, which is the goal of the denoising step we are conducting here.[95] So basically, the opening operation starts with eroding the microarray image, then dilates it using the same structuring element for both operations. Mathematically, a white top-hat filter of an image f is given by the following definition:

F: $E \rightarrow R$ be a grayscale image, mapping points from a Euclidean space or discrete grid *E*, into the real line, let b(x) be a structuring element of grayscale. Then, the white top-hat transform of *f* is given by:

$$T_w(f) = f - fob$$

o denotes the opening operation.

To apply morphological operations, we usually require gray scale or binary images. Besides, one characteristic of mathematical morphology is that it requires exact specification of the structuring element [96] Usually structing elements are smaller than the image being processed, and they focus on specific parts of the image [97] We typically choose a structuring element with the same size and shape as the objects we want to process in the input image[98]. Given the nature of the microarray image we are processing in this project, we choose a one-line 1D structure.

The chart below explains the denoising process using the morphological filter we described above. Another preprocessing step that should be taken into consideration is to convert any image that is stored in RGB format to the grey scale. The RGB image has different color plans that we are interested in. To extract the first plan, we index the first later, otherwise we can index other layers, using some mathematical functions of the image processing tool of Matlab. It is essential to observe that the spot shapes isn't necessary the same in both colors. Since we are already dealing with a greyscale images, we won't need to do this conversion step, and the analysis process should be the same.

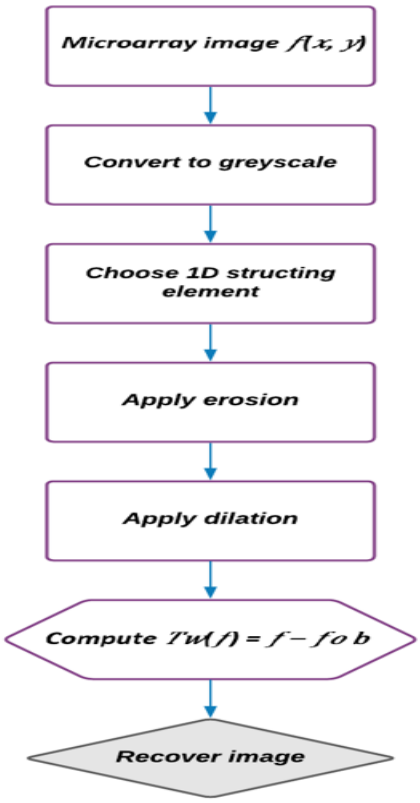


Figure 16 *Charts describing the design of a top-hat filter used in the preprocessing step of the pipeline*

3.4 Automatic gridding algorithm

A gridding method using projection technique is proposed in this project. This method is entirely based on computing image projections. One of the benefits of the horizontal and vertical profiles of a particular image is that they help with removing the noise, and can offer an easy way to get the distance between the spots. Mathematically speaking, the horizontal and the vertical projections of a microarray image f(x, y) can be defined as:

$$P_h = \frac{1}{X} \sum_{X=0}^{X-1} f(x, y)$$

$$P_v = \frac{1}{Y} \sum_{Y=0}^{Y-1} f(x, y)$$

Where P_h is the horizontal profile of the microarray image $f(x, y)$, and P_v is the vertical projection. X, Y represents the number of spots along the x-axis (rows) and vertical directions or y-axis (column) respectively. The actual image we are dealing with has noise and other artifacts, so it would be necessary to apply a denoising step to refine the projection profile we first obtained. This will ensure that there won't be any missing or redundant grid lines and ensures a smooth gridding image. The denoising is going to be done using the same morphological filter used before. Once the denoising steps and the horizontal profile is computer, we follow the following steps that summarizes the algorithm:

1. Compute the mean values of the horizontal projection of the image MP_h, give its equation

$$MP_h = \frac{1}{X} * \sum_{X=0}^{X-1} f(x,y)$$

Where MP_h represents the mean horizontal profile of the image, $f(x,y)$, and X is the horizontal dimension, following the x-axis and the couple (x,y) represents a specific pixel

2. Get the maximum peak indices from the MP_h
1. Assess the periodicity of the spots T_{Spots}, by calculating the period between two adjacent spots centers, this helps locating the spot spacing using the statistical properties of the autocorrelation function.
2. Segment peaks and locate centers.
3. Draw horizontal bounding box.
4. Calculate the mean values of the vertical projection of the image MP_v, defined by

$$MP_v = \frac{1}{Y} * \sum_{Y=0}^{Y-1} f(x,y)$$

Where MP_h represents the mean horizontal profile of the image, $f(x,y)$, and X is the horizontal dimension, following the x-axis and the couple (x,y) represents a specific pixel

7. Display the gridding results

The following charts summarizes the main steps of the algorithm

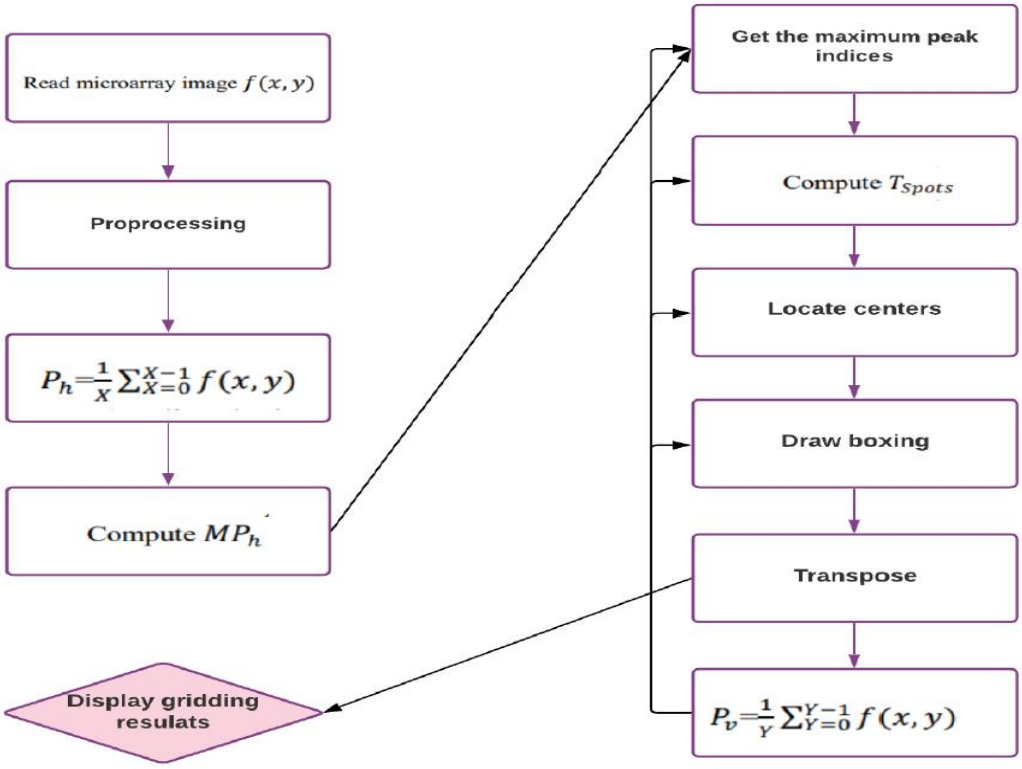


Figure 17 *Chart describing the gridding steps used for the proposed pipeline*

3.5 Evaluating the efficiency of the gridding method

In order to assess the performance of the suggested gridding method, we rely on three important parameters: the accuracy, the processing time and the noise removal ability of the modality.The accuracy (Ac) of the proposed gridding method can be defined as :

$$\text{Ac}= (\frac{N_{Correct\ Spots}}{N_{Total\ Spots}}) * 100\ \%$$

$N_{Correct\ Spots}$: The number of the spots correctly gridded

$N_{Total\ Spots}$: The total number of spots in a particular microarray image

While the speed of the gridding is the processing time the gridding technique takes to be fully executed by the software. Finally, having a noise resistant gridding approach is essential for the rest of the image analysis process, and therefore, it is important to address and comment on the algorithm behavior in regards to the different types of noise[99].

3.6 Automatic Segmentation using thresholding approach

The second step in our pipeline is to apply image segmentation. We need to differentiate between areas that are considered the spot signal and areas that are considered the background signal to carry on the analysis. There are several methods that can be used to automatize the segmentation process, and in this project we use thresholding approach given the advantages of this approach that we discussed in the previous chapter. We start by applying a threshold level to the entire microarray image in order to detect all spots equally. We need to carefully observe the image in hands, as we may encounter a pre-analysis issue that was previously described, which is the difference in the spot brightness across the grid. If we encounter this issue then we try to apply logarithmic transformation to equalize large variations in magnitude is by transforming intensity values to logarithmic space, and then apply the global transformation again. This will help detecting all spots. However, some weak spots may still be missed, and that's why we try to determine local thresholding values as a first variant to segment those weak spots.

To understand more the thresholding segmentation, we use another variant of the thresholding methods which is the multilevel thresholding technique, and it is pretty much a method that is based on segmentation, where two levels of thresholds can be chosen. The process segments a gray level image into several distinct regions. This technique determines segments the image into certain brightness regions using the two threshold values, which correspond to one background and several signals[100] We assess the performance of each method separately using the results we obtain, and the segmentation score Jaccard, which indicated the similarity between the ground truth and the segmented image. The score is calculated by dividing the number of observations in both sets by the number of observations in either set.[101].

.

3.7 Exploring the effect of machine learning on segmentation

We will be using an unsupervised machine learning method, we choose the K-mean clustering algorithm that we described in the previous chapter, because we are mostly dealing with pixel-wise segmentation.

The steps that we are going to follow to perform the K-Means algorithm are:

1. We start by choosing the number of clusters K.
2. Automatically calculate the centroids
3. Form k clusters, by assigning each data point to the closest centroid.
4. Computer and place the new centroid of each cluster.
5. Reassign each data point to the new closest centroid.
6. Assess the readiness of the algorithm.

3.8 Intensity extraction and expression level

This step seeks to calculate the intensity value of each spot. We measure the intensity levels of all spots for a given greyscale image or channel separately. In order to calculate the spot intensities, it is essential to understand that we need to define a mask around each spot, this masks is chosen based on the different regions of interests that we define using a microarray image. Once the mask is defined, we compute either the median or the mean value of the pixels that make that particular mask. The value that we get is considered the spot intensity for that spot in the particular channel. Once those intensities are calculated, we compute the expression levels for each specific spot and the entire image using the following equation:

$$\text{Expression level} = \frac{\text{Log (Intensity (First channel)}}{\text{Intensity (Second channel)}} = \frac{\text{Log (Intensity (Red))}}{\text{Log(Intensity (Blue))}}$$

3.9 Data analysis

Calculating the spots intensities for both channels for the previous model, as well as the expression levels using the previous equation is the final step in the pipeline. It would be essential to assess the performance of the pipeline. First of all, comparing the quantification results generated using ArrayPro Analyzer versus the one calculated using the pipeline is helpful. Calculating other statistical parameters such as the mean, median and standards deviations of some ROI would be a plus as well. Evaluating the effect of removing the morphological filters on the spot intensities would be also addressed. Once this is done, we feed the pipeline two different models of microarray images that are going to be used for validating that the pipeline is applied for any type of microarray data. The second model is simply different proportions of 3 different colours of fluorescent streptavidin printed on a slide, which is a multiple channels model that could help with the testing and colocalization. The last model is a two-channel microarray image with different dyes for each channel that was taken from a random biology experiment will be used for validation

as well. Finally, to get a more objective evaluation of the performance of the pipeline, we rely on simulation using ImageJ. Using the third model, we compare the true value of the intensities given by the software versus the ones obtained using the pipeline, and we make conclusions. The last part of the data analysis would be to test the robustness of the pipeline to different levels of Gaussian noise, and the impact of logarithmic transformation on the segmentation results and the reconstruction of weak spots.

Chapter 4: Results and Discussion

4.1 Automatic gridding of microarray images

4.1.1 Addressing pre-analysis issues

The first step is feeding the microarray images we have to MATLAB. The images used for this project are stored as TIFF files, which means each image is composed of two pages or two channels, the first one is the red channel and the second one is the blue channel in the case of the main data used. In order to better analyse the microarray images. For each slide, we split each channel separately and process it as a unique image. To do so we use ImageJ, which is an open source software, pretty effective when it comes to processing microarray images[102]. After loading each TIFF file that represents a different slide to the program, the channels could be splitted using the following command: Image->Stacks >Images to Stack. Figure 18 shows the interface of ImageJ.

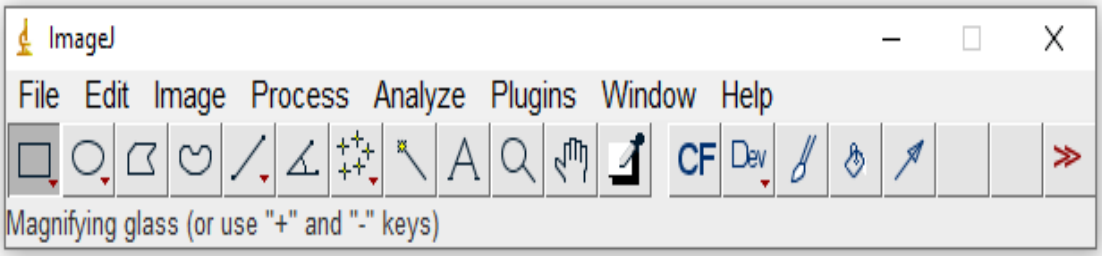


Figure 18 *Display of the interface of ImageJ software used to split the TIFF files of the microarray images used in this project*

It is important to note that it is still possible to use the entire TIFF image for each slide, instead of using the channels separately, since the main focus of the pipeline is to locate the spots, and this shouldn't affect the image analysis steps. However, it would be challenging to measure the spot intensities or the expression levels in the last part of the pipeline without separating the channels. The results of the splitting for each slide are shown in figures 19, 20 and 21.

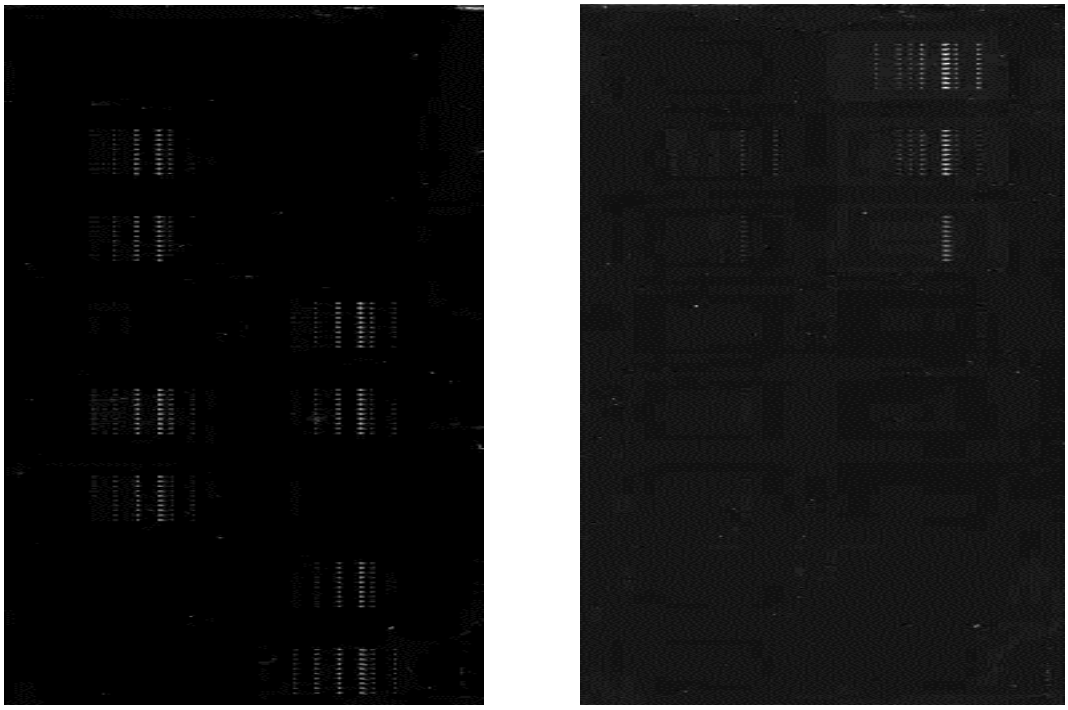


Figure 19 *Channels splitting for slide 1 of the antibody microarray experiment using ImageJ. Red channel (left) and blue channel(right) of the microarray image.*

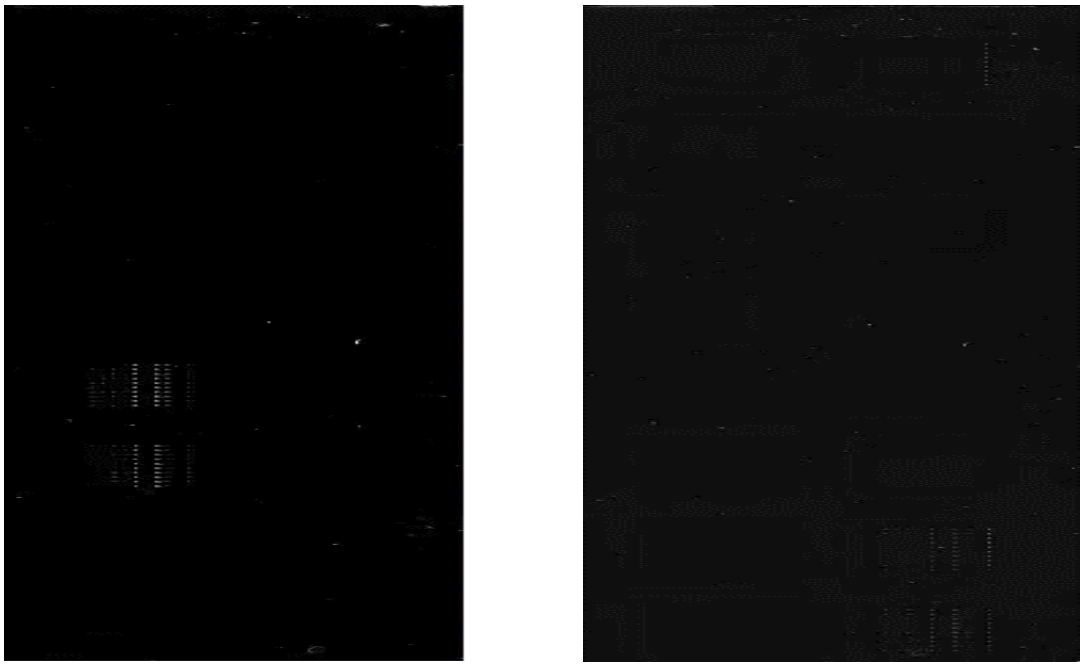


Figure 20 *Channels splitting for slide 2 of the antibody microarray experiment using ImageJ. Red channel (left) and blue channel(right) of the microarray image.*

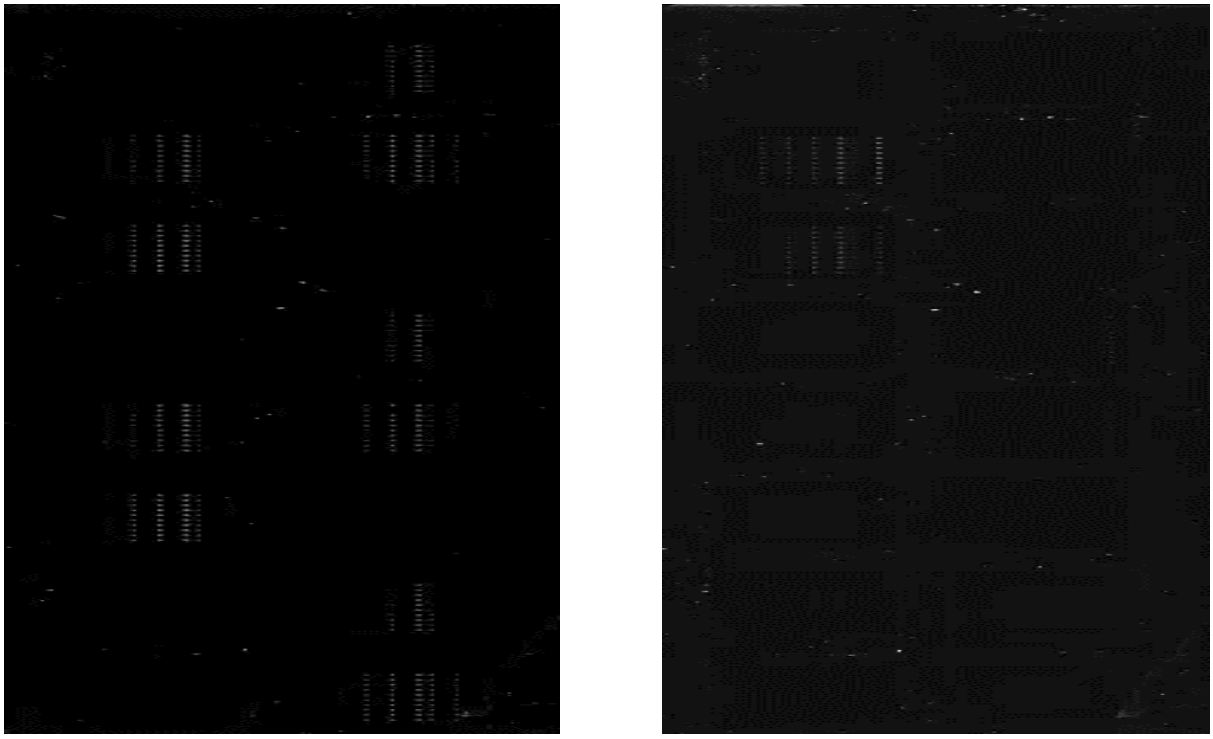


Figure 21 *Channels splitting for slide 3 of the antibody microarray experiment using ImageJ. Red channel (left) and blue channel (right) of the microarray image.*

Once the TIFF files are processed and the channels are successfully splitted. The next step would be to input each image representing a particular channel of each slide separately into MATLAB. This requires the use of the function imread, and since the size of the images is too big for the screen, the function imshow was used to reduce the size. Visualizing a particular area of the image and cropping sections of it to further observe how the subgrids are aligned, along with getting an idea of the variation in the spots intensities is helpful to get a more concrete understanding of our data. Figure 22 shows a cropped section of the microarray image from the first channel of slide 1 showing how the subgrid organizes in that microarray.

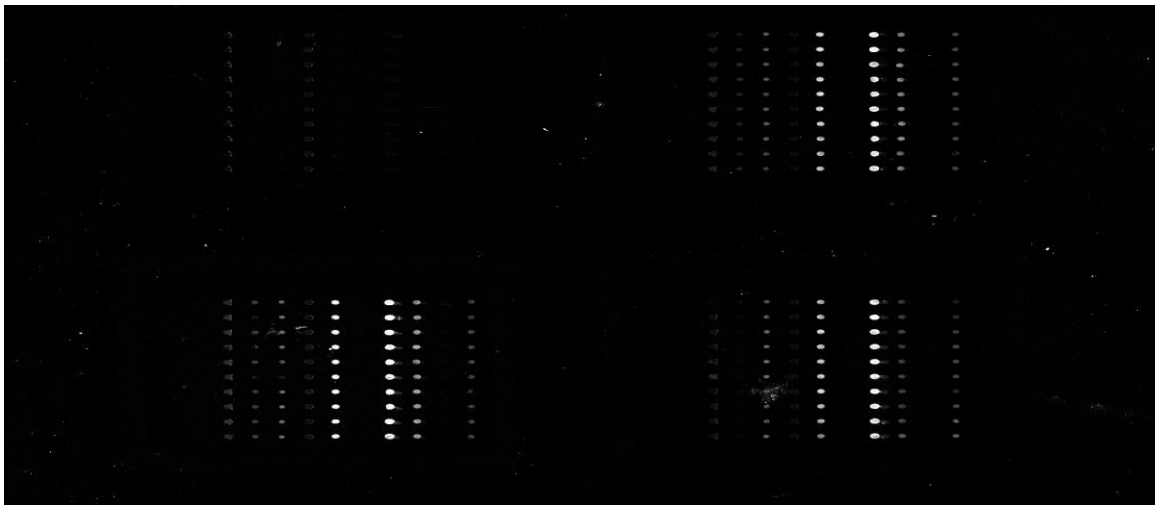


Figure 22 *Cropped view of a microarray image taken from slide 1 of the antibody microarray experiment used for a better understanding of the data.*

Before running the gridding algorithm, evaluating the mean intensity of the microarray image or the horizontal profile for each microarray image is needed. This is done by averaging the intensities of every column. Assessing the horizontal profile can help in locating the centers of the spots, as well as the spacing between those spots within the same sub grid and throughout the entire microarray image. This information will be used to build separations between the spots, and the entire gridding process relies on an accurate and smooth horizontal and vertical profile of the image. The figure below shows the horizontal projects for each slide. The Horizontal profiles on figure 23 seem to be uneven across slides, this is mainly due to the presence of noise, also in practice the spots tend to have different sizes and intensities, that could be presented in the form of irregularities in the horizontal/vertical profiles.

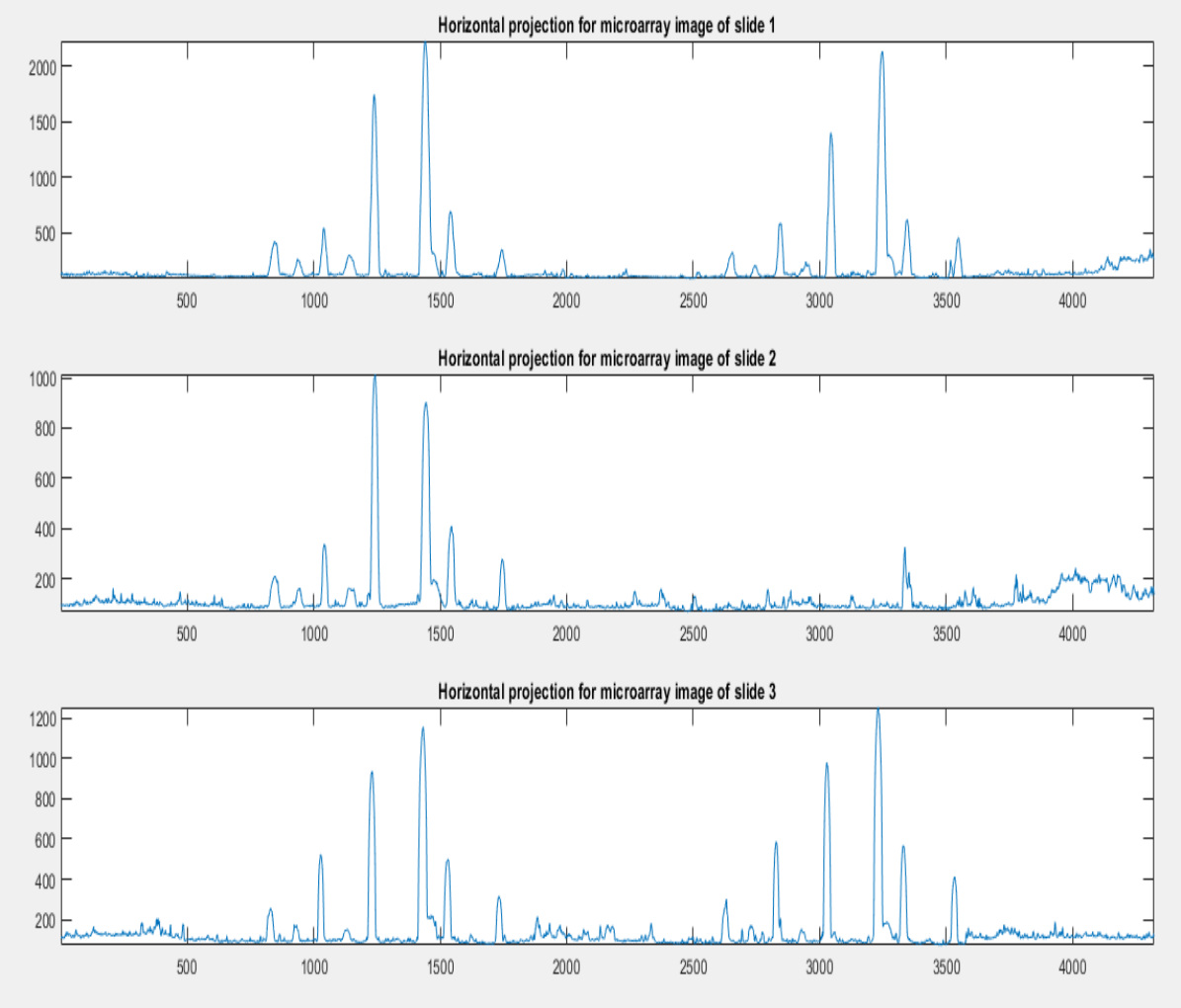


Figure 23 *Horizontal profile of the microarray images for slides 1, 2 and 3 of the antibody microarray experiment*

The next step is to apply the autocorrelation of the projection profile (figure 25), this allows the estimation of the period between different spots in one sub grid, which is basically the spacing value between spots. The value of the period provides useful information that can be used in correctly drawing the grids later on and achieving a successful gridding process. A possible limitation which could make the autocorrelation function prone to errors is the case where the microarray image is rotated such that the grid is not aligned with the x and y axes. One of the possible ways to solve challenge would be to manually point with the mouse two spots from the same column to calculate the rotation angle. For example, it could be possible to choose the center of the uppermost left spot and the center of the lowest left spot. Figure 24 shows an example of how the two points can be selected from any subgrid. The arrows indicate the points, as well as the order at which they have been selected.

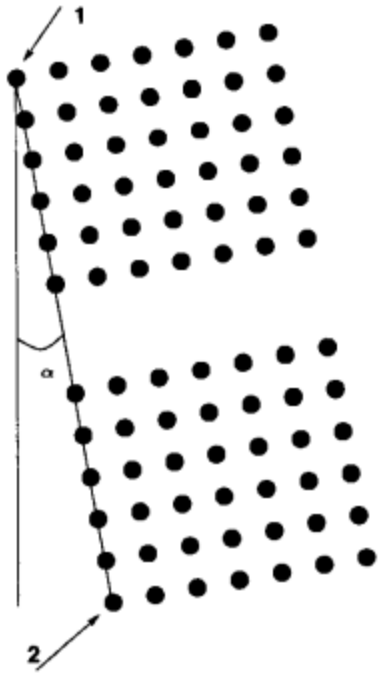


Figure 24 *Rotation correction for a random microarray image [103]*

After doing so, we calculate the image projections, estimate spot periodicity, and continue with the gridding steps of the proposed algorithm. This solution could be time consuming, especially when dealing with large data, and it wouldn't be the best as the entire goal of the pipeline is to automatize the image analysis process [103, 104] The second possible solution would be to first distinguish between the conventional x and y axis, from the x' and y' axis of the rotated image. We identify the main axes (x',y)' of the image using an iterative algorithm. Once those axes are identified, we compute the mean of the difference of the correlation values with respect to the maximum. Next, by using trigonometric operations, it is possible to find the angle relative to the conventional axes (x, y). The usual rotation methods (linear, bilinear..etc) can not be used to correct the rotation because they change the value of the pixels which could alter the quantification step of the pipeline[105, 106] Once this is done, we proceed to computing the spacing between the spots and follow the gridding algorithm steps.

It is essential to note that the images we are dealing with don't present such angles, and thus, we won't be addressing such scenario in this work, otherwise, the rotation would translate to having an autocorrelation function that is blurred, which require more adjustments for a successful gridding.

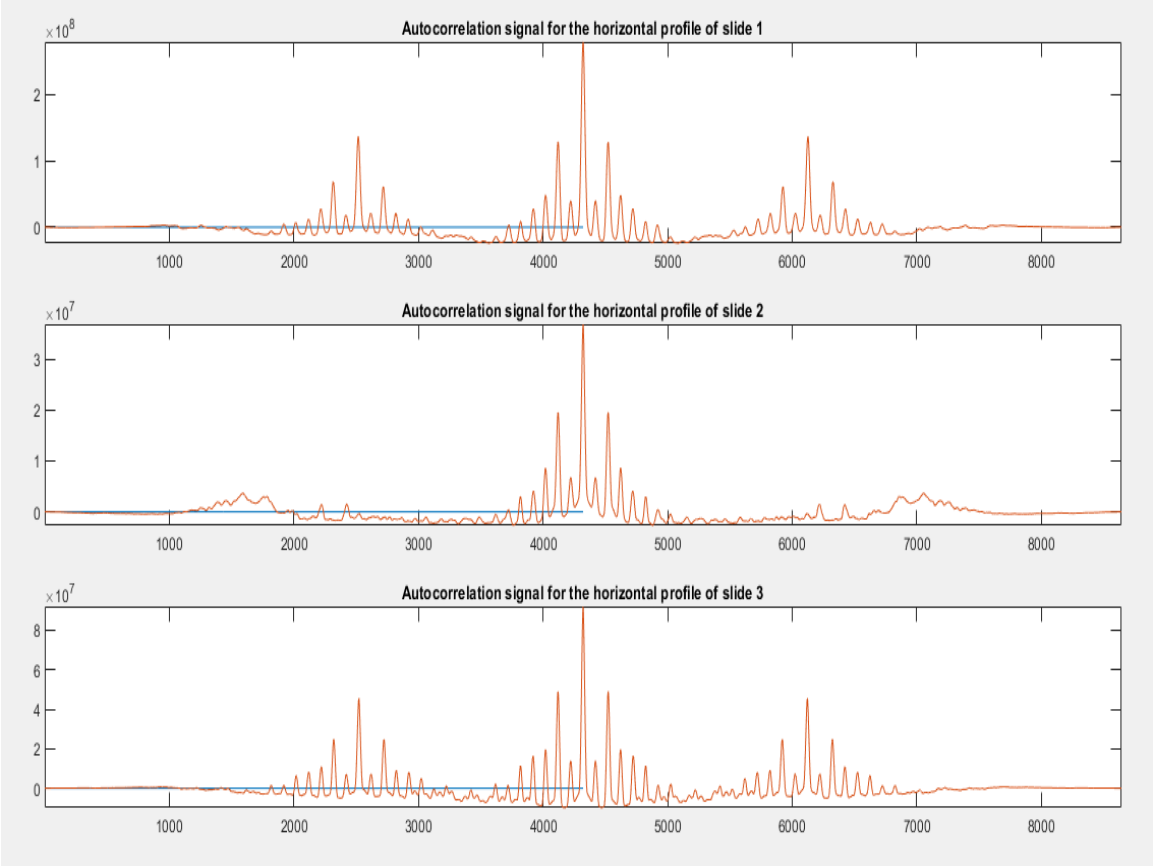


Figure 25 *Autocorrelation function of the microarray images for slides 1, 2 and 3 of the antibody microarray experiment used to improve the self similarity of the horizontal projection*

Table 3 presents the estimated value of the spacing between spots for each channel and for each slide. The values of the spacings are pretty much the same between two different channels for each slide, because basically the physical feature of the spots is conserved from one channel to the other.

Table 3 *Summary of estimated spacing between spots or periodicity for both channels and for slide 1, 2 and 3 of the antibody microarray experiment*

Number of slides	Tspot x axis (µm)	Tspot y axis (µm)
Slide 1	63	40
Slide 2	25	15
Slide 3	22	15

The next step is to improve the horizontal profile and make it more regular by removing background noise using morphological filters. The first filter would be a 1D line shaped structure given the form of the projection signal. Besides, cropping and observing the images for the three slides has demonstrated that the area near some spots, which belongs to the background part had

some intensity values, and this could create a bias in the gridding results and the rest of the pipeline analysis. Therefore, instead of only filtering the background intensities from the projections profiles, another preprocessing step was added to remove those surrounding background intensities, by using another morphological with a 2D structing element to subtract the neighbouring background intensities [107]. After applying both morphological filters, a more regular and smooth profile compared to the initial calculated horizontal profile is obtained, presented in figure 26.

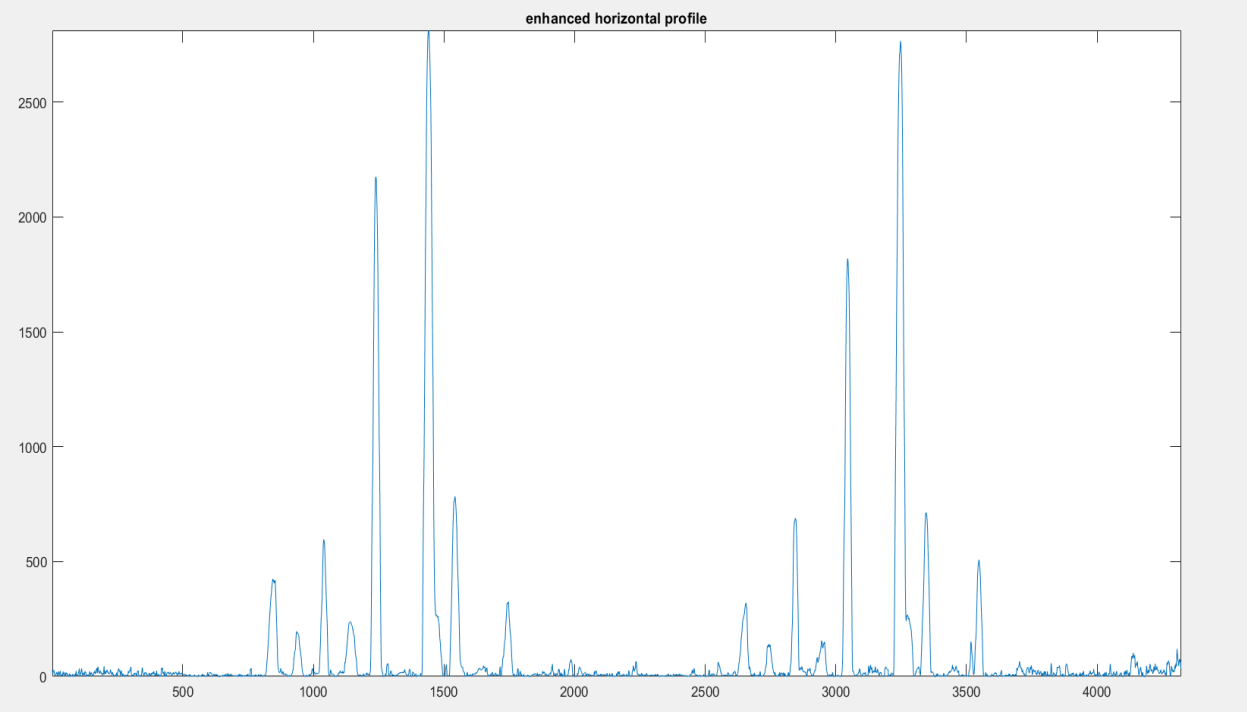


Figure 26 *Enhanced horizontal projection for slide 1 of the antibody microarray experiment, after applying two morphological filters to the horizontal profile signal. The signal shows more regularity in after filtering.*

4.1.2 Building horizontal and vertical projections of the microarray image

Once the spacing between spots is obtained using the autocorrelation function, the locations of the spots regions (SR), or the peaks that indicate the presence of a spot were determined. Figure 27 provides a visualization on where the peaks are located on the autocorrelation graph. Having a separation between spots is required, which lead to segmenting the previous peaks to clearly differentiate between an SR and a background. The peaks were segmented using a thresholding value of 118 for slide 1, 91.50 for slide 2 and finally 110.50 for slide 3. The thresholding values for each slide were calculated using the statistical properties of the data, and it was a global threshold obtained using the function graythresh that calculate the value of the threshold for those peaks. The threshold was applied for all the peaks of a particular slide. Once the peaks are segmented, it became easier to obtain the center of each spot using feature extraction. The horizontal projection and the corresponding autocorrelation function were used to get the horizontal centers (HC), while using the vertical projection gave the values of the centers of the spots that were organized on the columns, or the vertical centers (VC).

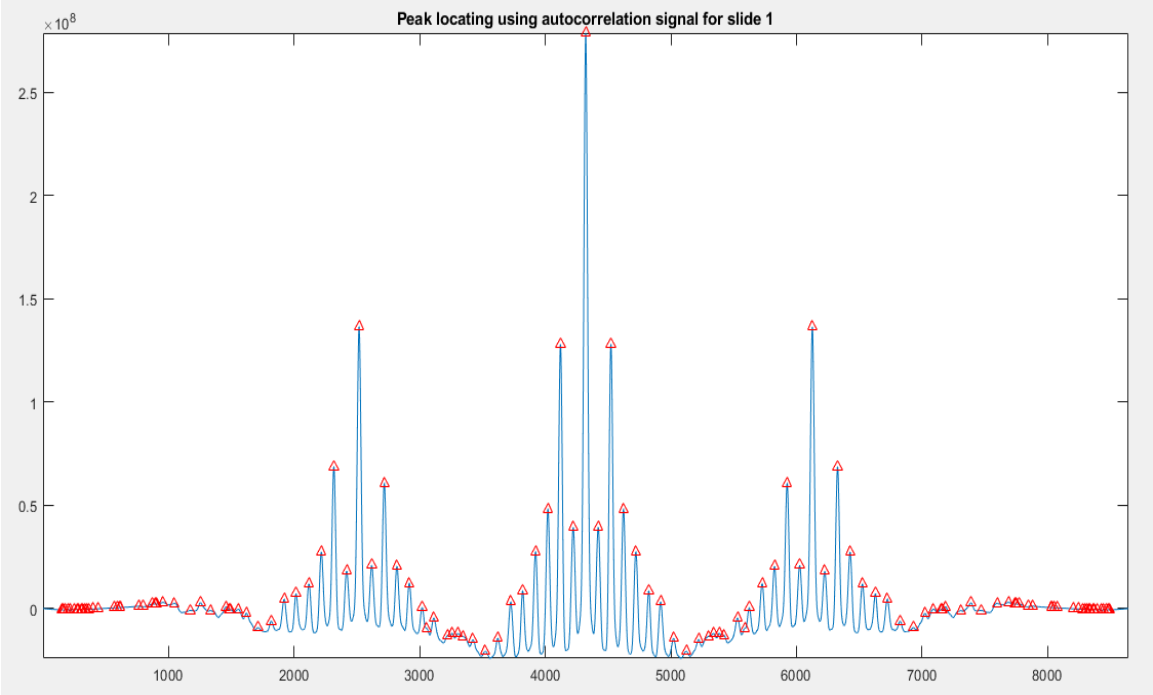


Figure 27 *Peaks regions finding using autocorrelation signal for slide 1 of the antibody microarray experiment.*

After the peaks are segmented, we obtain clear peak regions that we label, and it became easier to obtain the center of each spot using feature extraction (figure 28). The horizontal projection and the corresponding autocorrelation function were used to get the horizontal centers (HC), while using the vertical projection gave the values of the centers of the spots that were organized on the columns, or the vertical centers (VC).

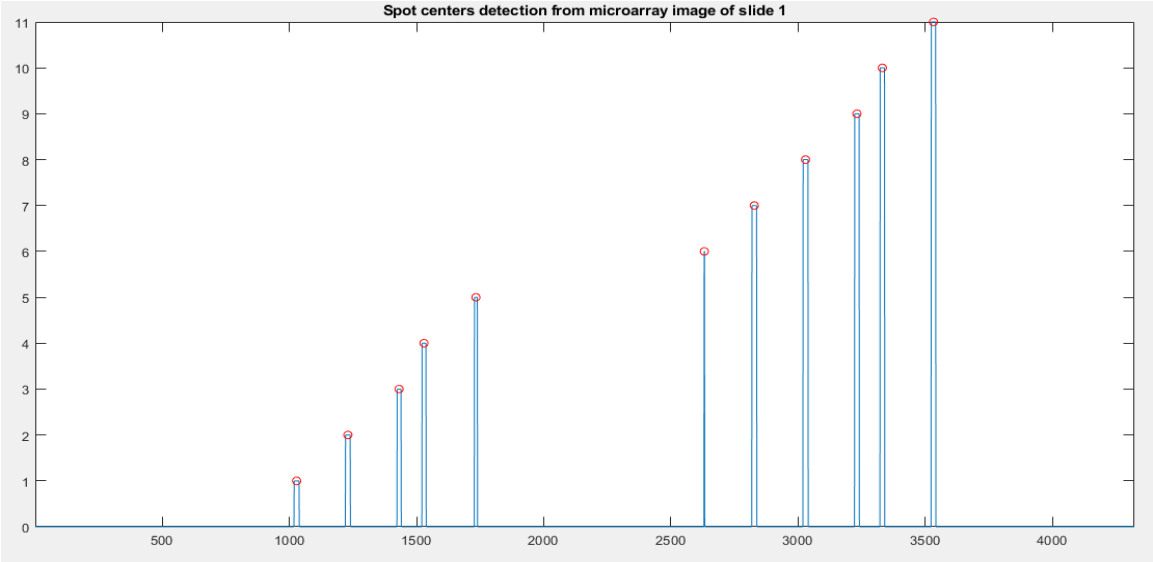


Figure 28 *Determination of the centers of the horizontal spots after segmenting the peaks into labeled regions. The analysis consists of applying a feature extraction task that extracts the centroids of the peaks or spot centers, from slide 1 of the antibody microarray experiment.*

Once the information about the spots centers is obtained, the next step is to build vertical separations as a first step of building the actual gridding separators. The centroids coordinates calculated before, are used to calculate gap between each spot, that can be given by the equation below, Where Δ_x is the difference between two adjacent centers following the x-axis.

$$\text{Gap}= \frac{\Delta_x}{2}$$

Once the gab values are obtained, we draw the vertical separations between the different lines of the microarray image, which is basically the division between the spots are determined, this is necessary to know how to draw the vertical lines or the boxing step for the gridding.

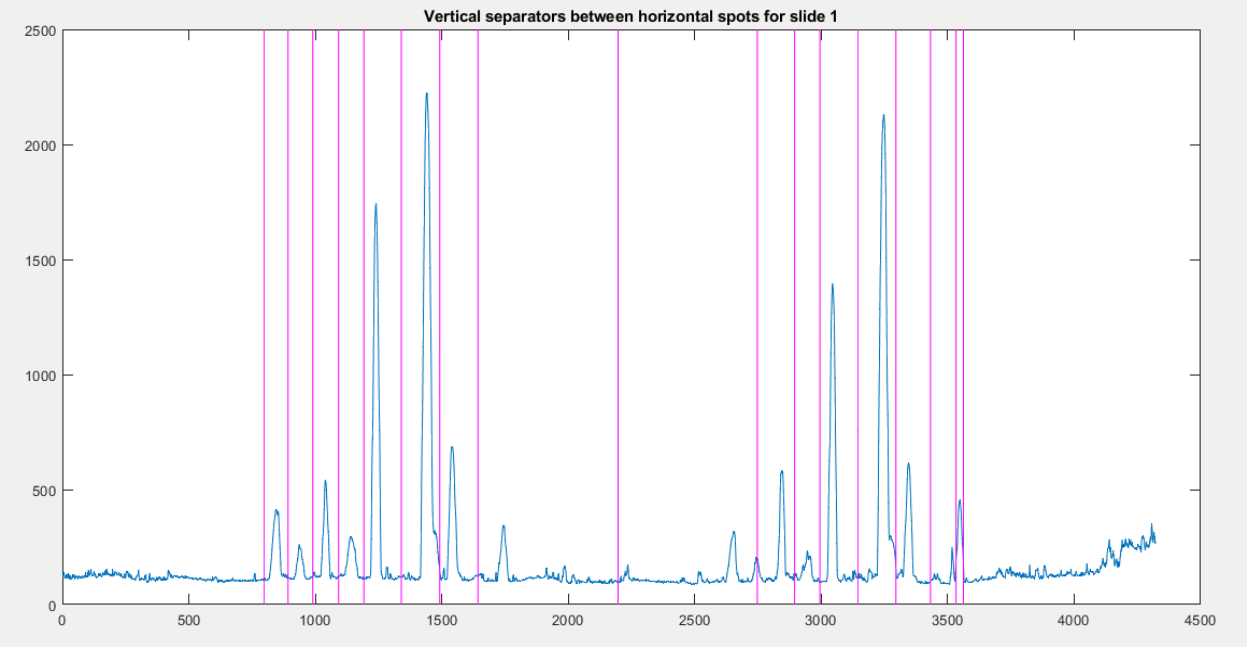
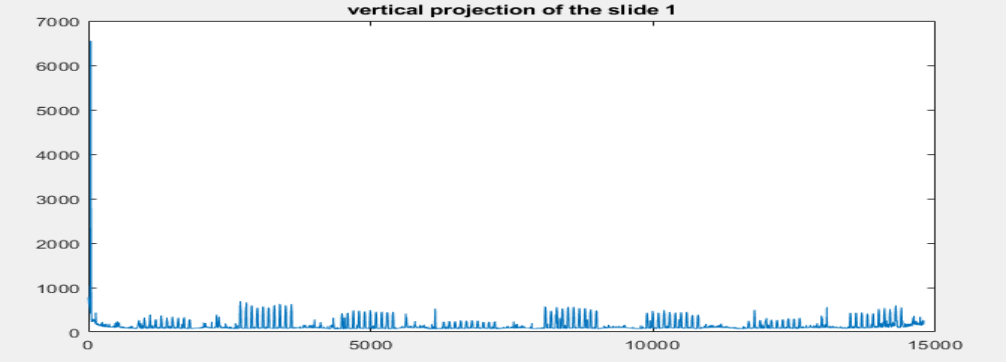


Figure 29 *Vertical divisions between peaks regions for slide 1 of the antibody microarray. The midpoints between adjacent peaks provides grid point locations that is the basis of building the divisions.*

Now that the vertical separations are determined and drawn on the grids of the microarray image, we repeated the same steps to complete the grid. Starting by transposing the input signal, and computing the vertical projection this time, along with its autocorrelation function as presented by figure 30. More points are shown as there are more spots that are aligned on the columns compared to the lines of the image.



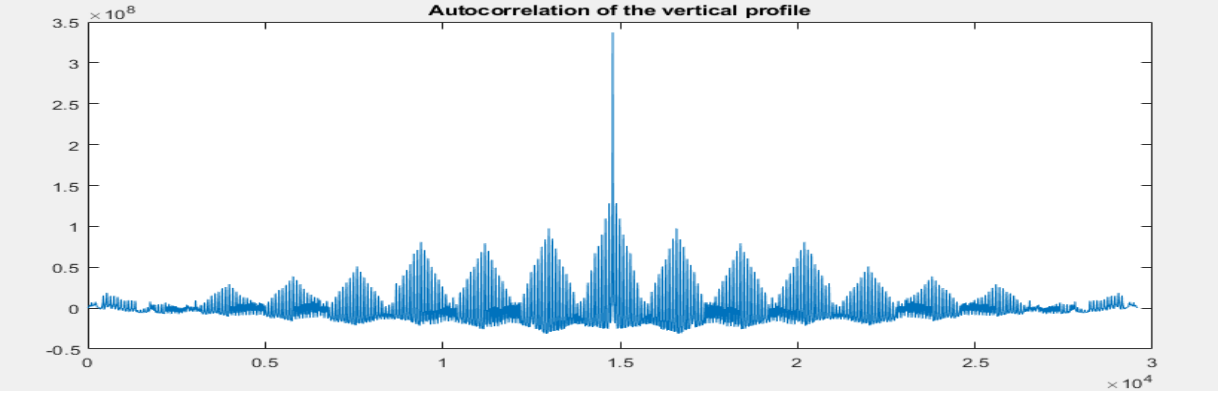


Figure 30 *vertical profile of the microarray images along with the corresponding autocorrelation signal for slides 1 of the antibody microarray experiment*

Similarly, we enhance the enhanced profile, we locate the peaks, segment those peaks and label them to determine the center of the spots, to calculate the gaps between the spots following the y-axis using the previous formula:

$$\text{Gap}=\frac{\Delta_y}{2}$$

Figure 31 shows the results of the horizontal divisions between spots from the first slide.

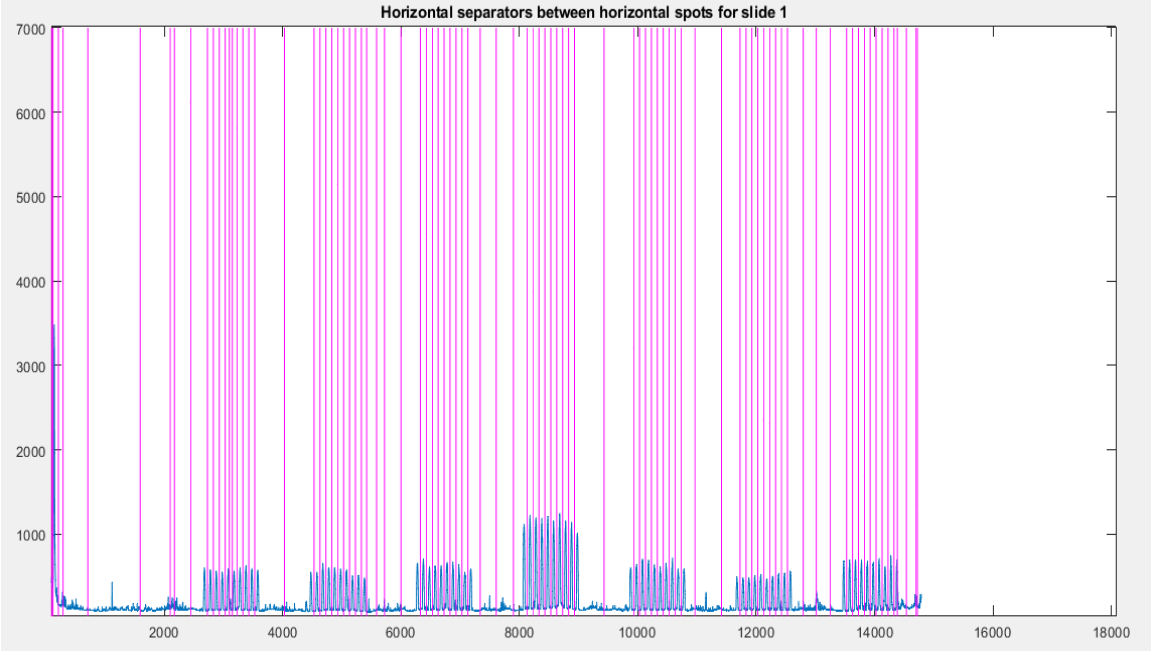


Figure 31 *Horizontal divisions between peak regions from slide 1 of the antibody microarray experiment. The midpoints between adjacent peaks provides grid point locations that is the basis of building this divisions*

The process of the gridding is finalized, by drawing the horizontal separations on the grid and putting bounding boxes around each spot to have it confined. The final result of the gridding for the first slide are shown on figure 32 below:

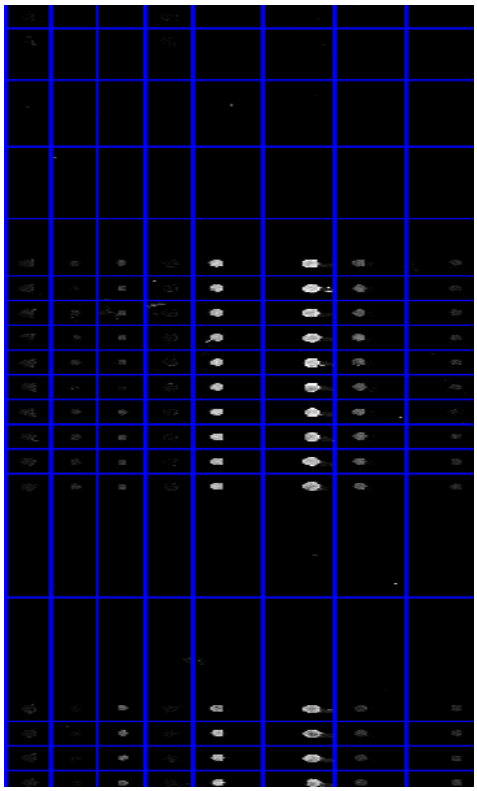


Figure 32 *Results of the automatic gridding for slide 1 of the antibody microarray experiment*

The following table represents the gridding parameters that were implemented in order to assess the efficiency of this method. Some spots seemed to have centers that are connected to other adjacent centers, which made obtaining a correct estimation of their centers a bit challenging, these spots were mostly inaccurately gridded, which explains a lower gridding accuracy for slide 2. The execution time of the algorithm is short, and the main timing difference between each slide is due to the fact that some slides have more spots compared to the others, and also because of the presence of different background levels. The difference in the accuracy is also related to the quality of the image. For instance, the image quality for slide 1 is better than the one of slide 3, since for slide 1 the variations in the spot intensities is not as large as the one from slide 3, so there is less probability to mistake the spots from slide 1 with the background, and therefore, the gridding accuracy is expected to be higher on slide 1, as demonstrated by the table 4.

Table 4 *The gridding accuracy and the processing time for the automatic gridding of all the three slides of the antibody microarray experiment*

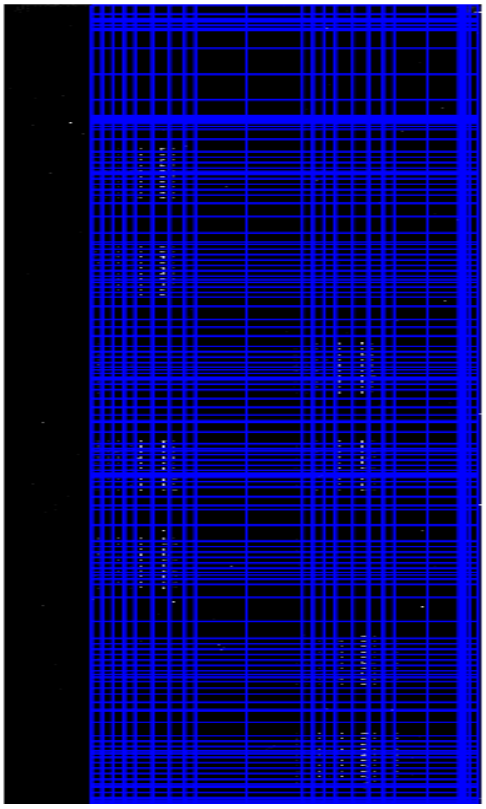
Slide number	Accuracy	Processing time
1	97.5%	4s
2	66.25%	3s
3	92.35	6s

Exploring the effect of the preprocessing and the denoising on the gridding results and the gridding parameters is an interesting aspect to look at. To do so, we compared the gridding results with denoising versus the case where no background noise filters were implemented. The accuracy and speed of the gridding were reported in table 5.

Table 5 *Comparison between the gridding parameters for all the three slides of the antibody microarray experiment in the case of background removal using morphological filters, versus the case of keeping the background noise.*

Slide number	Accuracy		Processing time	
	Without denoising	With denoising	Without denoising	With denoising
1	85.94%	97.5%	4.50s	4s
2	88.00%	66.25%	4.48s	3s
3	83.91%	92.35%	6.26s	6s

We observe a displacement of the entire grid towards the right as figure 33 and 34 show, this means the spots that were situated in sub grids where there was a noticeable shift in the coordinates of the grids were more exposed to gridding errors.



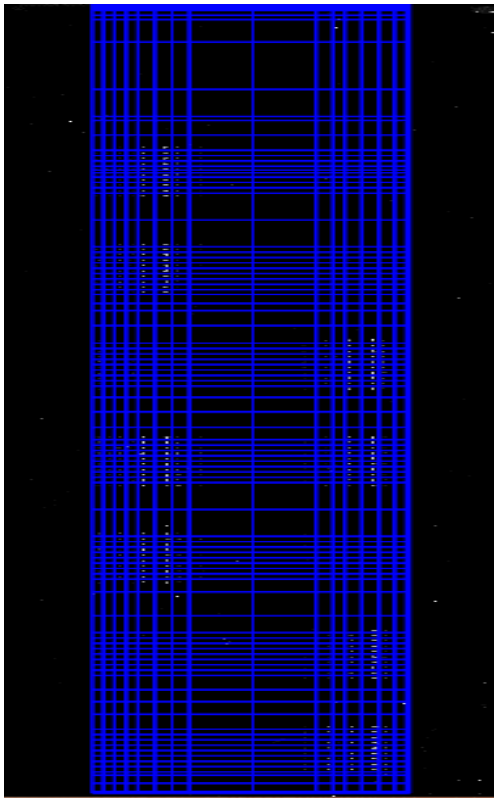


Figure 33 *Displacement of the grid of slide 1 of the antibody microarray experiment due to removing the presence of background noise as a result of removing the morphological filters*

Besides, the accuracy results have dropped for both slide 1 and slide 3, the gridding still effective despite the percentage of error, and this is mostly due to the fact that half of the sub grids for slide 1 and 3 were situated in the right corner of the platform, so more spots would be incorrectly gridded as a result of the grid displacement towards the right, and therefore the accuracy drops. For slide 2 however, we notice that the accuracy improved, that's mostly due to the fact that one of the denoising steps may haven't been necessary. Although refining and regulating the horizontal and vertical profile would have been useful for estimating the spacing, trying to remove the intensities around the spots using the second morphological filter may have missed with the spacing values between some adjacent spots, and therefore the entire organization of the grid.

The processing time is slightly higher in both slide 1 and 2 as it would be more challenging to analyze the horizontal profile when it is not regular, and thus more time is required to estimate the spacing, complete the gridding steps and build both the horizontal and vertical grids. The time of the gridding is still low and the process is still fast despite removing both layers of the morphological filters.

There are different numbers of grids between channel 1 and channel 2 for each slide because the spacing between the spots is not exactly the same in general, even when the denoising is applied, since the spots tend to have different shape from one channel to the other, and therefore we could expect a slight different in the number of grids.

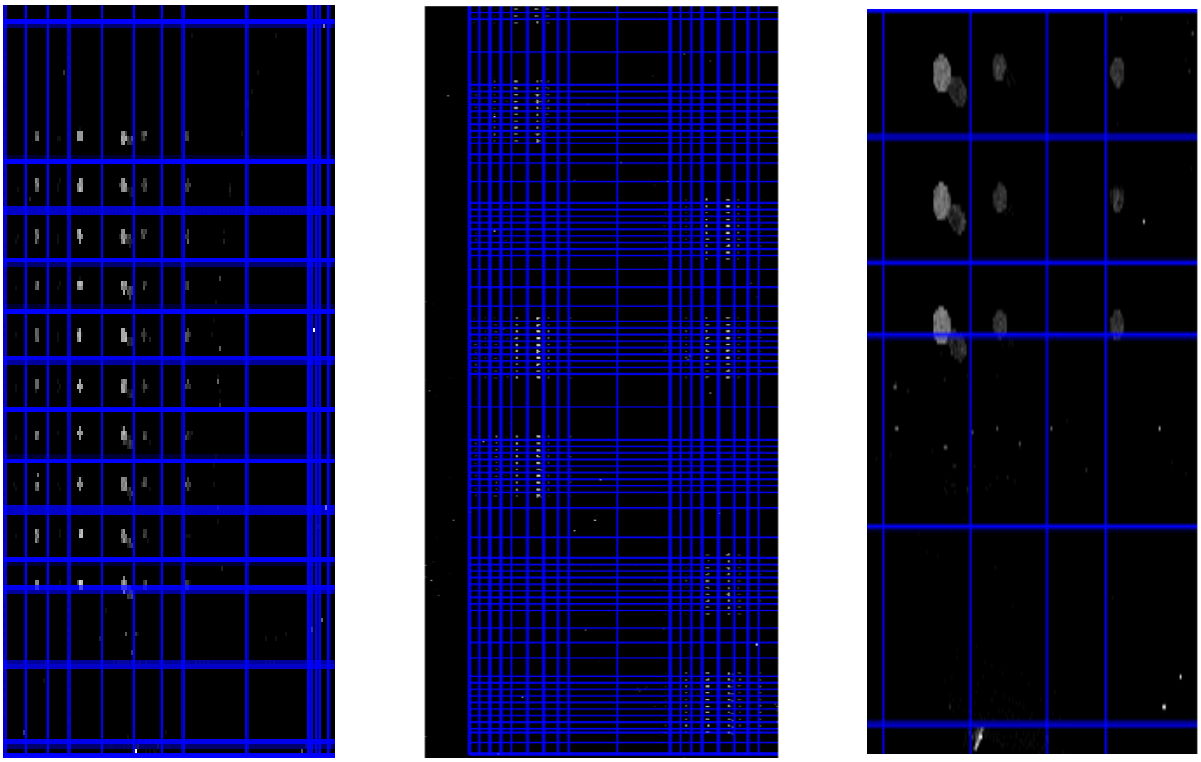


Figure 34 *Results of the gridding method applied on slide 1 of the antibody microarray experiment without background filtering, showing a displacement of the grid creating a misalignment issue and more spots to be inaccurately gridded.*

Comparing the previous results showed the importance of the preprocessing in obtaining a better image projection. Without background noise subtraction, it is still possible to get decent gridding results, as even without morphological filters being added, most of the spots were still correctly gridded. However, the grid was still displaced, and this has been observed in all three slides, causing gridding errors for the spots that are situated far in both the right side and left side of the microarray image. Another issue is that some of the spots with low level of brightness were more easily missed, as they were mistaken as background intensities. Basically, the estimation of the spacing between spots is biased, which leads to more errors in detecting the peak values and areas around spots, and therefore it becomes more challenging to correctly build the grids and the boxes that contain the spots. The execution times increased since we removed the denoising layers from the gridding algorithm.

The proposed gridding method relies mostly on the autocorrelation function to define the grid first, then segments the spots in the grid. Alternatively, one could identify and segment the spots using a Lorentzian of Gaussian filter, and then use the spot centroids to define the grid or rely on other techniques or algorithms that could enable the grid definition, using a different approach, or a mechanism that serves a similar goal while performing the gridding steps in a relatively different order.

The gridding technique is expected to be more robust. First, the method doesn't require a lot of processing steps or computational power, as most operations as already available in the form of simple functions under the image processing toolbox of MATLAB. Besides the gridding method, relies on feature extraction, which helps in improving the accuracy of calculating the centroids, as it was used in the automatic gridding algorithm of this pipeline. The information about the centroid is essential since it is the basis of the calculation of the gap between the spots across the grid and therefore the horizontal and vertical divisions that build the boxes around the spots. Determining the value of the centroids using feature extraction reduces the size of the dataset, speeds up the algorithm [108], but also shows that there is always a room for improvement since research is always trying to come up with ways to improve feature extraction techniques, which means the first step of our pipeline could always be optimized for a more regular grid, with a better control of any misalignment that could occur[109]. The gridding steps makes no assumptions about the size of the spots, rows, and columns in the grid, which gives it a more universal usage compared to other algorithms that may require to input parameters or some level of human intervention. Finally, applying a two steps denoising at the early stage of the preprocessing phase helps with having an accurate definition of the grid, since removing the morphological filters created misalignment and displacement of the grid as seen before. Defining the grid before segmenting the peak regions could help with the segmentation score, as it adds an additional layer of background separation from the spot regions, which therefore improve the quantification results that follows the segmentation. In other words, basing the entire the pipeline on the autocorrelation could build a solid foundation for the remaining image processing steps.

4.2 Image segmentation based on thresholding methods and machine learning approach

4.2.1. Segmentation using thresholding techniques

After successfully gridding the previous microarray images, we proceed into the second step of the pipeline, which is the segmentation. The method consists of applying a global threshold to the entire image as explained before in the previous chapter, the value of the threshold is computationally chosen using MATLAB in a way that all the spots are detected equally. The threshold values obtained were the following: 0.17 for slide 1, a value of 0.23 for slide 2, and 0.086 for slide 3. Figure 35 displays the results for the first slide.

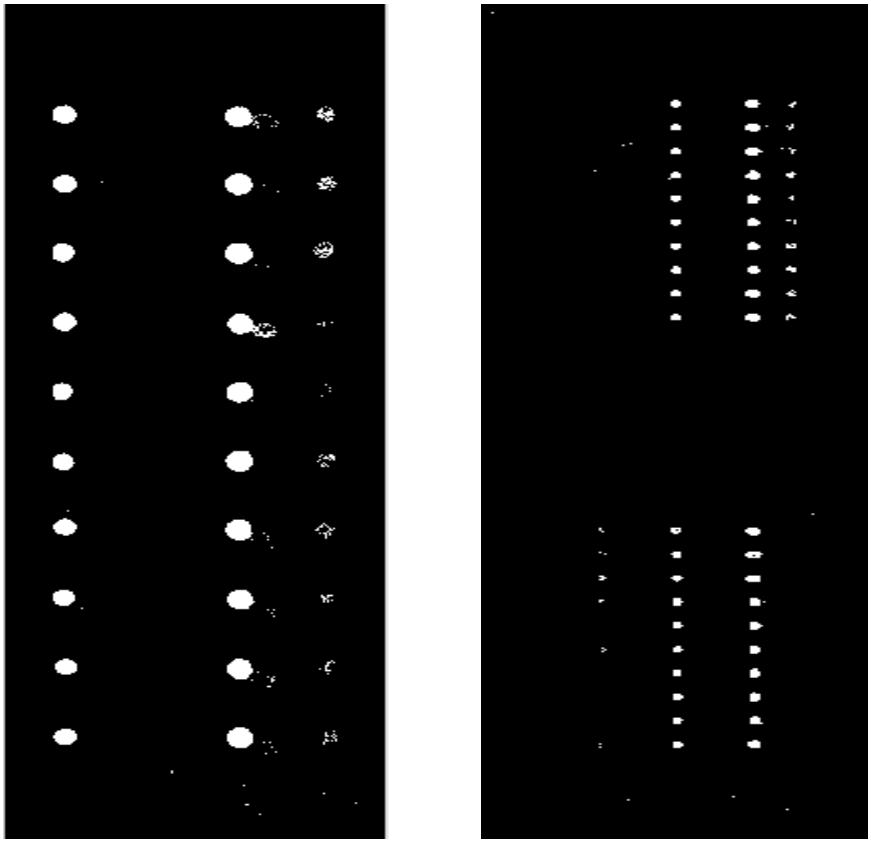


Figure 35 *Results of the global thresholding segmentation with a threshold value of 0.17 for slide 1 of the antibody microarray experiment*

For slide 1 some spots are missed, this is due to the variation in the level of brightness between spots, especially between those in two different columns within a specific subgrid. The main reason behind the brightness variation is the fact that the capture antibody that was used in the experience that led to generating these slides differs from one column to the other, and as mentioned in the previous chapter, each column represented a different capture antibody. Therefore, the brightness varies within the same subgrid, and across the entire image, making some weak spots miss the threshold value. Table 6 summarizes the values of the segmentation score Jaccard, in comparison to the ground truth for all of the three slides [110].

Table 6 *Segmentation scores for global thresholding segmentation method for all the three slides of the antibody microarray experiment.*

Slide number	Jaccard
1	0.27
2	0.51
3	0.38

There are different ways to perform a segmentation based on thresholding. The following section investigates the effect of some variations of the thresholding technique and reports the segmentation scores for each one of them. First, for adaptive segmentation, a locally adaptive threshold for 2-D grayscale image is used, with a value that is based on the local mean intensity or the first-order statistics in the neighborhood of each pixel. The threshold is then used to binarize the image. Using MATLAB, and the function adaptthresh, the values of the first-order statistics are shown in figure 36 below, while the result of the image reconstruction are illustrated in figure 37.

	1	2	3	4	5	6	7	8	9	10
1	0.0108	0.0108	0.0109	0.0109	0.0110	0.0111	0.0111	0.0112	0.0113	0.0113
2	0.0107	0.0108	0.0109	0.0109	0.0110	0.0111	0.0111	0.0112	0.0113	0.0113
3	0.0107	0.0108	0.0109	0.0109	0.0110	0.0110	0.0111	0.0112	0.0113	0.0113
4	0.0107	0.0108	0.0108	0.0109	0.0110	0.0110	0.0111	0.0112	0.0112	0.0113
5	0.0107	0.0108	0.0108	0.0109	0.0110	0.0110	0.0111	0.0112	0.0112	0.0113
6	0.0107	0.0108	0.0108	0.0109	0.0110	0.0110	0.0111	0.0112	0.0112	0.0113
7	0.0107	0.0107	0.0108	0.0109	0.0109	0.0110	0.0111	0.0111	0.0112	0.0113
8	0.0107	0.0107	0.0108	0.0109	0.0109	0.0110	0.0111	0.0111	0.0112	0.0113
9	0.0107	0.0107	0.0108	0.0109	0.0109	0.0110	0.0111	0.0111	0.0112	0.0113
10	0.0107	0.0107	0.0108	0.0109	0.0109	0.0110	0.0110	0.0111	0.0112	0.0113
11	0.0107	0.0107	0.0108	0.0109	0.0109	0.0110	0.0110	0.0111	0.0112	0.0112
12	0.0107	0.0107	0.0108	0.0108	0.0109	0.0110	0.0110	0.0111	0.0112	0.0112

Figure 36 *Values of first-order statistics in the neighborhood of each pixel for slide 1 of the antibody microarray experiment calculated for the adaptive thresholding method.*

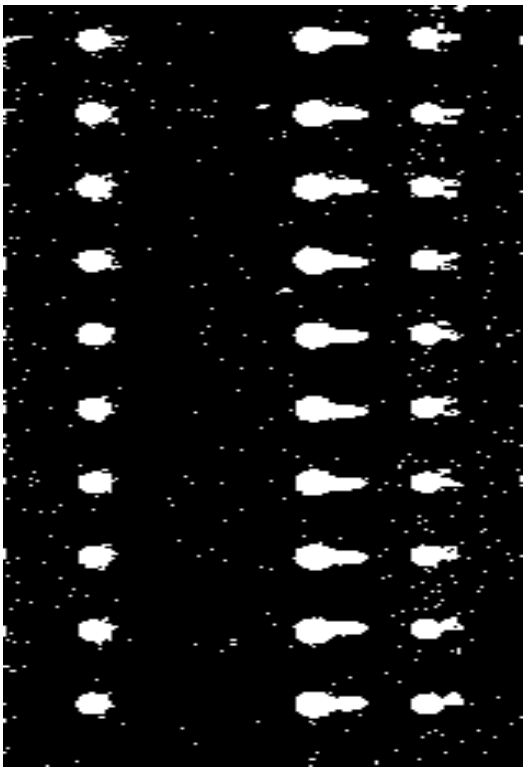


Figure 37 *Results of the adaptive thresholding segmentation for slide 1 of the antibody microarray experiment. Spots are indicated with the big white pixels.*

Unlike the global thresholding technique, the local adaptive thresholding calculates different values of thresholds for every pixel in the microarray image based on the analysis of its neighboring pixels. The technique is mostly useful when dealing with images with different levels of contrasts, and usually as an alternative whenever the global segmentation isn't giving interesting results[111] In this work, and despite having a low segmentation score using the global thresholding method, it stills better than the adaptive local thresholding given the type of microarray image used given the segmentation score (Table 7) and the quality of the images obtained by each segmentation modality comparing to the ground truth. More white pixels were observed as and the contours seem to be affected as well, this is because local properties of different regions of the microarray image are usually not homogeneous at the same scale, which makes it difficult to estimate the local parameters with satisfactory accuracy without incorrectly affecting other spots[112].

Table 7 *Segmentation scores of the adaptive thresholding method for all slides of the antibody microarray experiment*

Slide number	Jaccard
1	0.2606
2	0.3288
3	0.2794

We also use a different variant of the global thresholding method which is the multilevel thresholding based on the otsu method that was described above. The results are shown in figure 38. Basically, several weak spots were correctly shown, providing high segmentation scores for all the slides as mentioned in table 8. The method seems to be faithful to uniformity after reconstruction of the image, as well as to shape measures of the different stops [100]

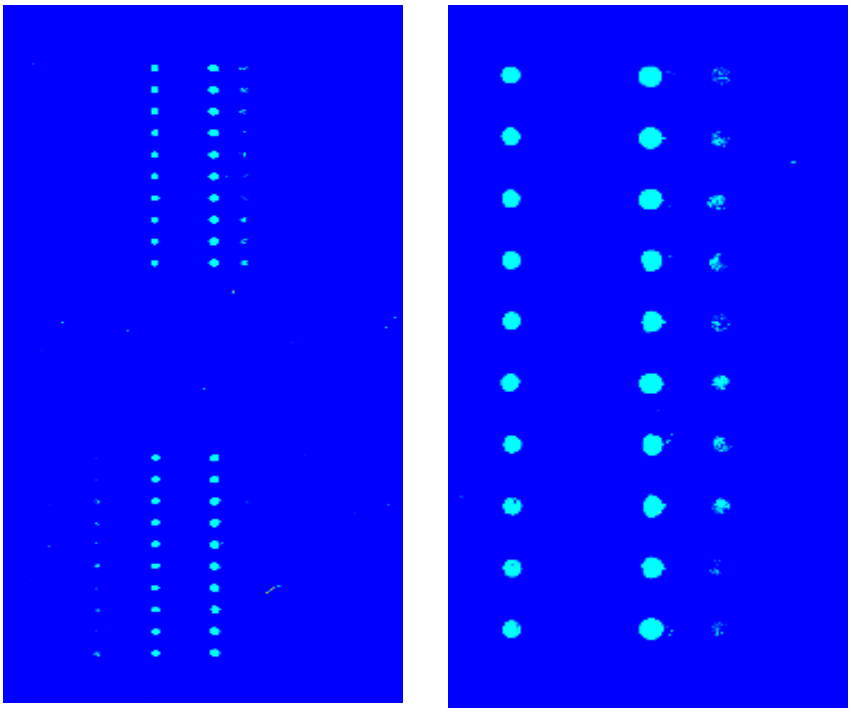


Figure 38 *Results of the multilevel segmentation with two levels for slide 1 of the antibody microarray experiment*

Table 8 *Segmentation scores for the multilevel thresholding for all the slides of the antibody microarray experiment*

Slide number	Jaccard
1	0.9131
2	0.8700
3	0.8656

4.2.2 Segmentation based on k-means algorithm offers better quality scores

In traditional clustering algorithms, the number of clusters and initial centroids are randomly selected by the user. The most important input for any clustering algorithm is the number of clusters K. It is a difficult task to estimate the value of clusters that could easily generate accurate segmentation results, and this applies for any data[113]. The k-means algorithm was first executed using 2 clusters (k=2) This number was chosen, while the initial centroids values were

automatically calculated[114] Setting the number of clusters and evaluating the segmentation scores would be an alternative in choosing an optimal value of clusters for this segmentation task. The results of the segmentation are shown in figure 39.

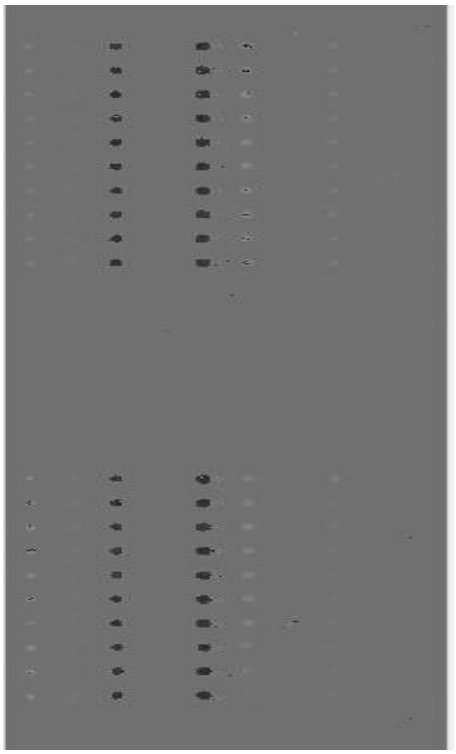


Figure 39 *Results of machine learning based segmentation with the number of cluster equal to 2 (k=2) for slide 1 of the antibody microarray experiment. The cluster number was chosen randomly.*

Table 9 *Segmentation scores for k-means method with the number of clusters equal to 2 for all the three slides of the first microarray image model*

Slide number	Jaccard
1	0.9131
2	0.9999
3	0.8700

The Jaccard similarity is pretty high for most of the slides (table 9), which shows that the segmentation was faithful to the ground truth. Trying to expand the number of clusters to K=4 instead of two clusters, showed a difficulty in segmenting the spots, and more spots were missed (figure 40). Therefore, for the rest of this work, the algorithm will be run with two clusters only.

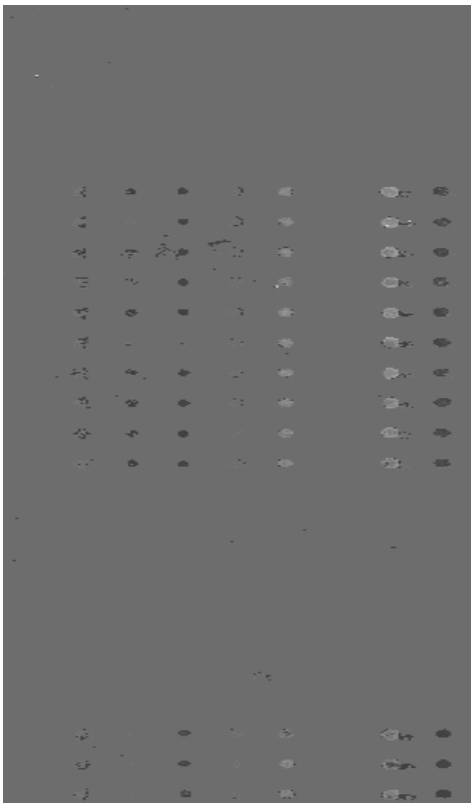


Figure 40 *Results of machine learning based segmentation with K=4 of slide 1 of the antibody microarray experiment showing a decrease in the quality of the reconstruction of some weak spots*

Table 10 *Segmentation scores for k-means method with the number of clusters equal to 4 for all the slides*

Slide number	Jaccard
1	0.9131
2	0.8700
3	0.8655

Table 11 summarizes the segmentation scores for all the methods that have been used. The table showed that using a multilevel thresholding with two levels is superior to the other variants of the thresholding method. The results also demonstrated the power of machine learning in providing an effective segmentation. It is also essential to note that the application of the K-Means algorithm is restricted by the fact that the number of clusters should be known beforehand, and that for other microarray images and in other settings, more values of clusters should be tested before choosing the optimal one that could offer an accurate segmentation.[115]

Table 11 *Segmentation scores for all thresholding methods of the slides from the antibody microarray experiment.*

Slide number	Global thresholding	Adaptive thresholding	Multilevel thresholding (2 levels)	k-Mean clustering (2 clusters)
1	0.27	0.2606	0.9131	0.9131
2	0.51	0.3288	0.8700	0.9999
3	0.38	0.2794	0.8656	0.8700

4.3 Microarray image quantification

4.3.1 Intensity extraction and expression levels estimation

Calculating the intensity of the spots for the previous three slides, requires focusing on the intensities of each channel individually. The algorithm applies a mask to capture each spot separately, each mask captures a spot that is represented with different coordinates on the slide, forming a region of interest (ROI). The mean value of the pixels within each mask represents the intensity of the spot assigned to that ROI. The expression levels of the spots were calculated using the method described in the previous chapter by taking the division of the logarithmic of intensities of the first channel on the intensities of the second channel, as indicated with this equation:

$$\text{Expression level} = \frac{\text{Log (Intensity (First channel))}}{\text{Log(Intensity (Second channel))}} = \frac{\text{Log (Intensity (Red))}}{\text{Log(Intensity (Blue))}}$$

The tables below summarize the different intensities for different regions of interests for both channels for the three slides, along with the expression levels at that particular region of interest. The ROI were selected randomly, but the method can compute ROI across all the images.

Table 12 *Intensity of spots and expression level calculation from 20 random ROI for slide 1 of the antibody microarray experiment.*

ROI	Intensity of first channel	Intensity of second channel	Expression level
1	2408.3	7311.3	-0.482
2	266.839	5922.2	-1.346
3	322.302	1530.9	-0.676
4	306.881	1388.5	-0.655
5	248.204	1.663.2	-0.826
6	496.253	1399.2	-0.450
7	395.930	1512.7	-0.582
8	190.556	1406.4	-0.868
9	80.166	1367.0	-1.231
10	1198.4	1455.6	-0.084
11	213.415	1508.4	-0.849

12	71.415	1468.3	-1.313
13	126.304	1569.7	-1.094
14	253.654	1560.0	-0.788
15	101.155	1627.1	-1.206
16	1034.6	1495.9	-0.160
17	90.866	1518.7	-1.223
18	98.355	1774.2	-1.256
19	78.539	2092.4	-1.425
20	168.778	1643.4	-0.988

Table 13 *Intensity of spots and expression level calculation from 20 random ROI for slide2 of the antibody microarray experiment.*

ROI	Intensity of first channel	Intensity of second channel	Expression level
1	1950.7	101.305	1.284
2	1596.6	7047.2	-0.644
3	68.272	1531.7	-1.350
4	77.217	1454.8	-1.275
5	78.746	1542.8	-1.292
6	75.041	1547.7	-1.314
7	126.194	1594.9	-1.101
8	192.751	1427.7	-0.869
9	84.276	1391	-1.217
10	1127.0	1458.8	-0.112
11	262.442	7047.2	-1.428
12	65.201	1545	-1.374
13	1096	1468.5	-0.127
14	134.393	1315.8	-0.990
15	1356.4	1424.1	-0.021
16	86.731	1573.6	-1.258
17	80.072	1310	-1.213
18	106.215	1347.7	-1.103
19	100.087	1669.9	-1.222
20	102.944	1888.7	-1.263

Table 14 *Intensity of spots and expression level calculation from 20 random ROI for slide 3 of the antibody microarray experiment.*

ROI	Intensity of first channel	Intensity of second channel	Expression level
1	123	108.712	0.053
2	1350.8	5069	-0.574
3	165.534	1595	-0.983
4	84.010	1610.1	-1.282

5	336.374	1551.8	-0.664
6	87.779	1557.9	-1.249
7	80.549	1566.1	-1.288
8	114.273	1478.7	-1.111
9	422.68	1472.6	-0.542
10	82.001	1495.2	-1.260
11	1350.8	5097	-0.576
12	97.215	1323	-1.133
13	94.401	1465.6	-1.191
14	878.54	1449.2	-0.217
15	145.415	1443.8	-0.996
16	76.453	1467.1	-1.283
17	78.175	1392.9	-1.250
18	79.413	1381.5	-1.240
19	169.457	1419	-0.922
20	86.834	1524.1	-1.244

Testing the effect of the background removal on the intensity level is also important. Five different regions of interests were selected and their intensities were calculated with denoising and without it. The background removal was conducted using both morphological filters described before. Figure 41 demonstrates that the values of the intensities were reduced for both channels where the noise wasn't removed, since the background values usually negatively add to the signal intensity.

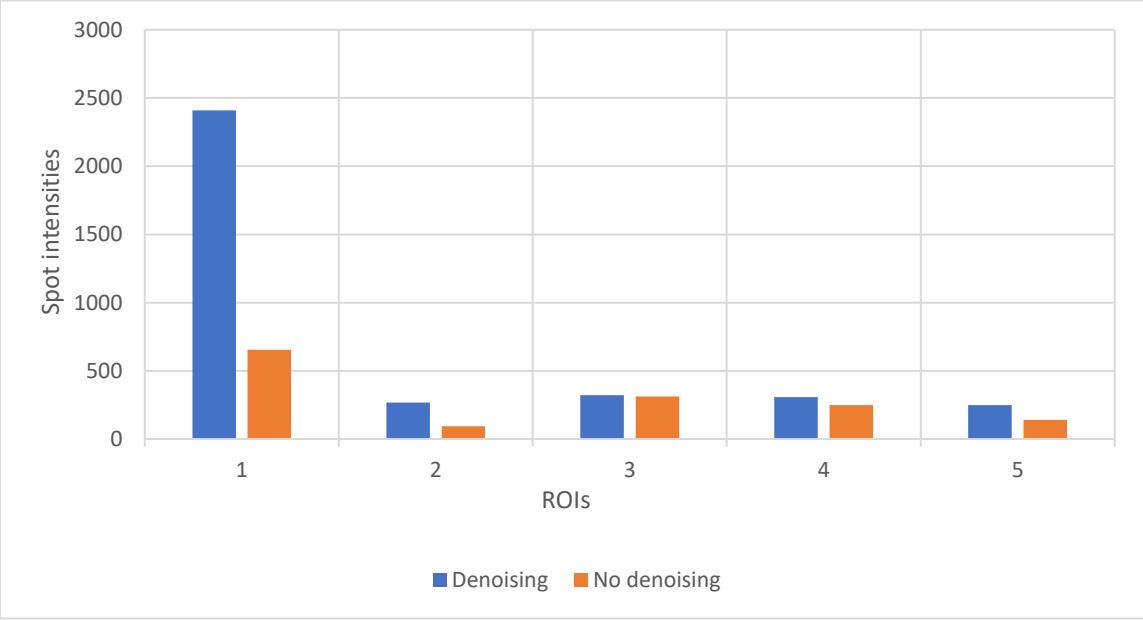


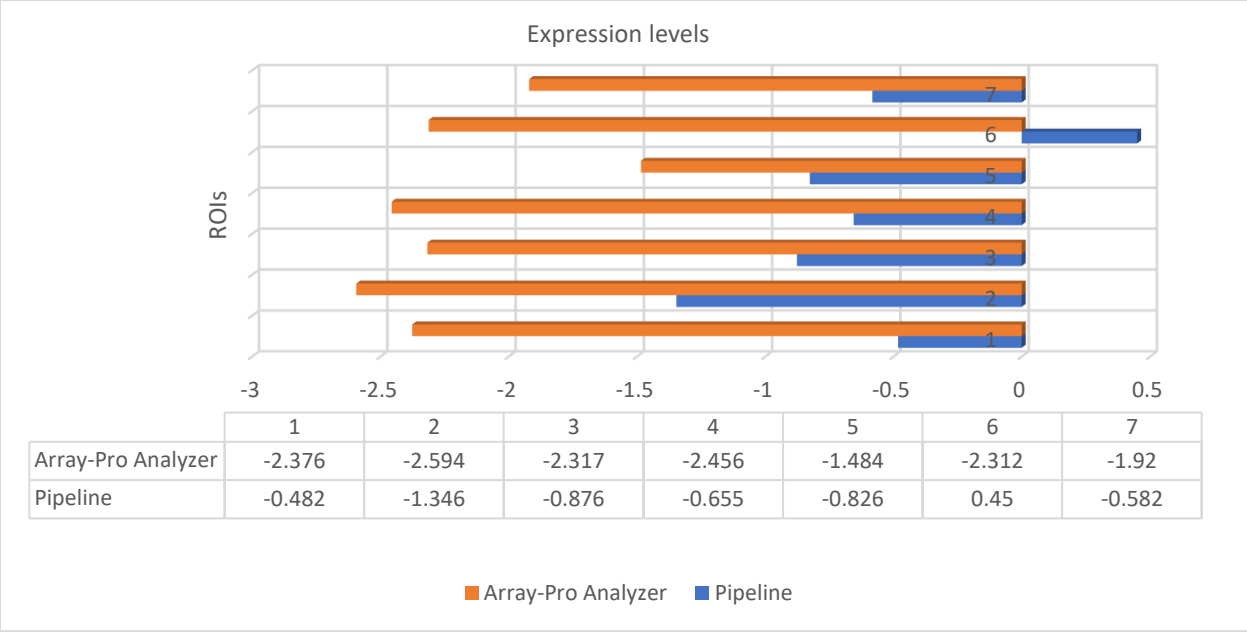
Figure 41 *Background removal effect on intensity levels of 5 different ROIs taken from slide 1 of the antibody microarray experiment*

Another way to compute the spot intensities is to take the median value of the pixels within a spot mask such as shown by table 15. In a normal distribution, the mean and the median are equal [116]. We notice that the values of the intensities computed using the mean are different from those calculated using the median for the first channel, while the intensities for some ROI such as number 3 is a bit close for the second channel. This is due to the fact that the intensities distribution is considered to be skewed for both channels.

Table 15 *Mean versus Median for the spots quantification using 3 ROIs taken from slide 1 of the antibody microarray experiment*

ROI	Intensity calculated using the mean for first channel	Intensity calculated using the mean for second channel	Intensity calculated using the median for first channel	Intensity calculated using the median for second channel
1	2408.3	7311.3	235	10688
2	266.839	5922.2	68	2469
3	322.302	1530.9	85	1486

Given that the spot intensities for both channels were already estimated using the mean values with ArrayPro Analyzer, all the intensities calculations in this work will be based on the mean calculation. This could be helpful in drawing conclusions by comparing the quantification results generated by ArrayPro Analyzer versus the ones obtained using the method proposed in this work. The expression levels for seven ROI that were selected randomly are illustrated in figure 42. A large difference in the expression results for those ROI, which could be difficult to objectively evaluate without knowing the true value of the spot intensities. It would be also helpful to look at the expression levels across the entire image to get a better idea on how other ROI performed. The difference in expression levels could be due to several things, such as the fact that the gridding using ArrayPro was conducted manually, and it is not clear how the segmentation was also conducted using the software. Other processing steps such as intensity transformations using the software may have been applied to the intensities. The mechanism of background subtraction using the software is an essential variable that is unknown as well. In the next section of this chapter, an alternative way to objectively assess the proposed pipeline will be presented.



	1	2	3	4	5	6	7
Array-Pro Analyzer	-2.376	-2.594	-2.317	-2.456	-1.484	-2.312	-1.92
Pipeline	-0.482	-1.346	-0.876	-0.655	-0.826	0.45	-0.582

Figure 42 *Comparing expression levels generated by ArrayPro Analyzer and the ones obtained using the proposed pipeline for 7 different ROIs from slide 1 of the antibody microarray experiment*

In order to further compare the proposed pipeline with Arraypro Analyzer, we calculate more statistical parameters, such as the mean of the intensities, the standards deviations using the intensities for the same random regions of interests. Some parameters seem to be close for some channels when computed with the results of the pipeline or ArrayPro, but it still not enough to determine how good the quantification of the pipeline was, as the most accurate statistical behaviors of the intensities isn't known, as the true values for the experiment weren't given.

Table 16 *Comparison between three different statistical parameters: mean, standard deviation and the median calculated using intensities generated from ArrayPro Analyzer versus the one obtained using the proposed pipeline from different ROIs of slide 1 of the antibody microarray experiment*

Statistical parameter	First channel using the pipeline	First channel using ArrayPro	Second channel using the pipeline	Second channel using ArrayPro
Mean of the intensities	407.545	585.037	2060.8	12670
Standard deviation of the intensities	559.83	1160	1582.6	15290
Median of regions of interests	230.81	147.74	1524.8	3079.2

4.3.2 Pipeline validation using other microarray data models and simulation

The previous results have showed that the proposed pipeline works well with the slides we have been analyzing for the antibody microarray experiment. Applying the pipeline in one type of

microarray images, and three different slides is a good start, but it would be more interesting to show that the pipeline is universal by validating it on different models of microarray image. The first model represents microarray images taken from three different colours of fluorescent streptavidin printed on a slide that was previously analyzer at the Micro and Nano Bioengineering lab. Figure 43 shows the results of the enhanced horizontal projection as well as the peak location using the autocorrelation function that was used before. The estimated values of the spacing are 49 µm following the x-axis and 45 µm following the y-axis.

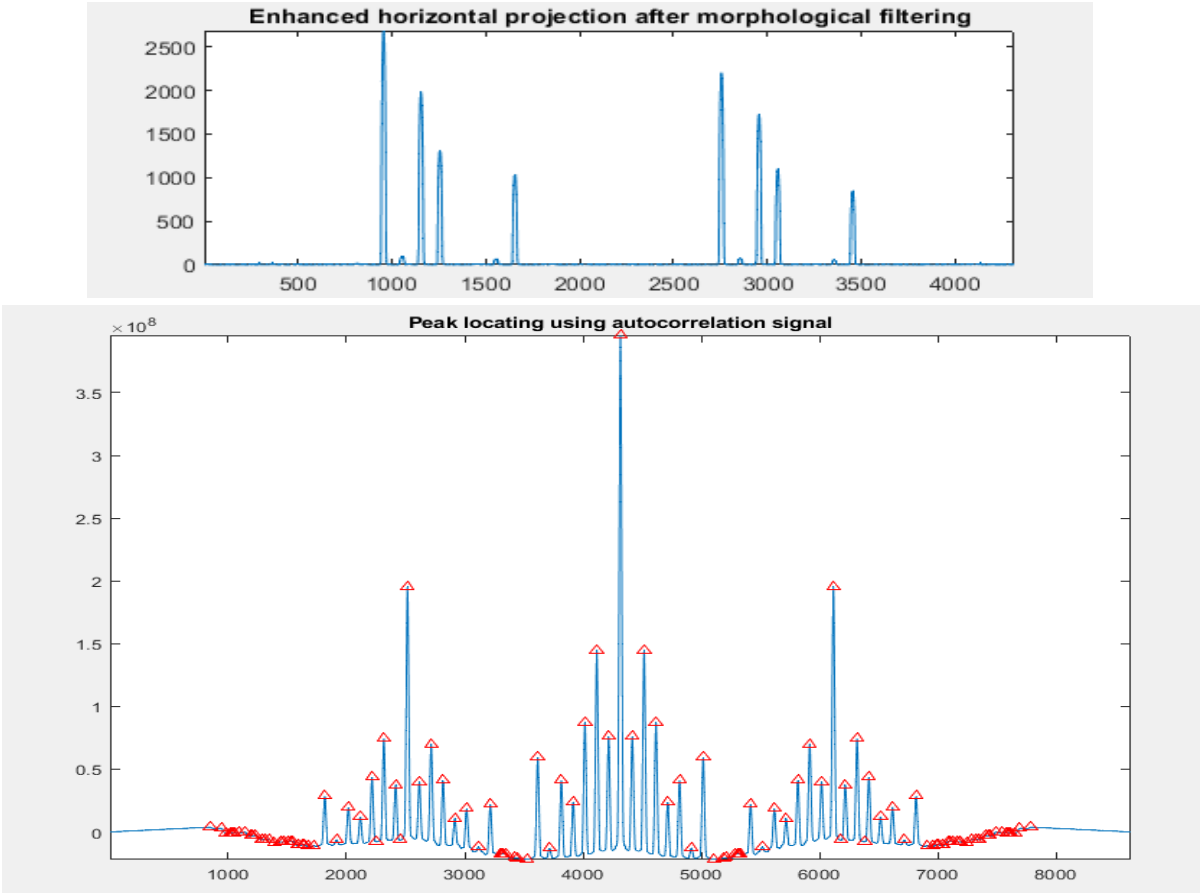


Figure 43 *Horizontal profile along with the corresponding autocorrelation signal of a multiple channels model representing three different colors of fluorescent streptavidin printed on a slide*

It is expected to have a more regular grid compared to the antibody microarray model, since the gap between the horizontal and vertical spacings is smaller. The gridding procedure is the same as before, and after creating the vertical separations, we transpose the signal and repeat the same steps, and finally draw the boxes around the spots. The gridding accuracy was excellent for this model 100%, so all the spots were correctly gridded, with a processing time of 6.51 seconds. The gridding results are shown in figure 44

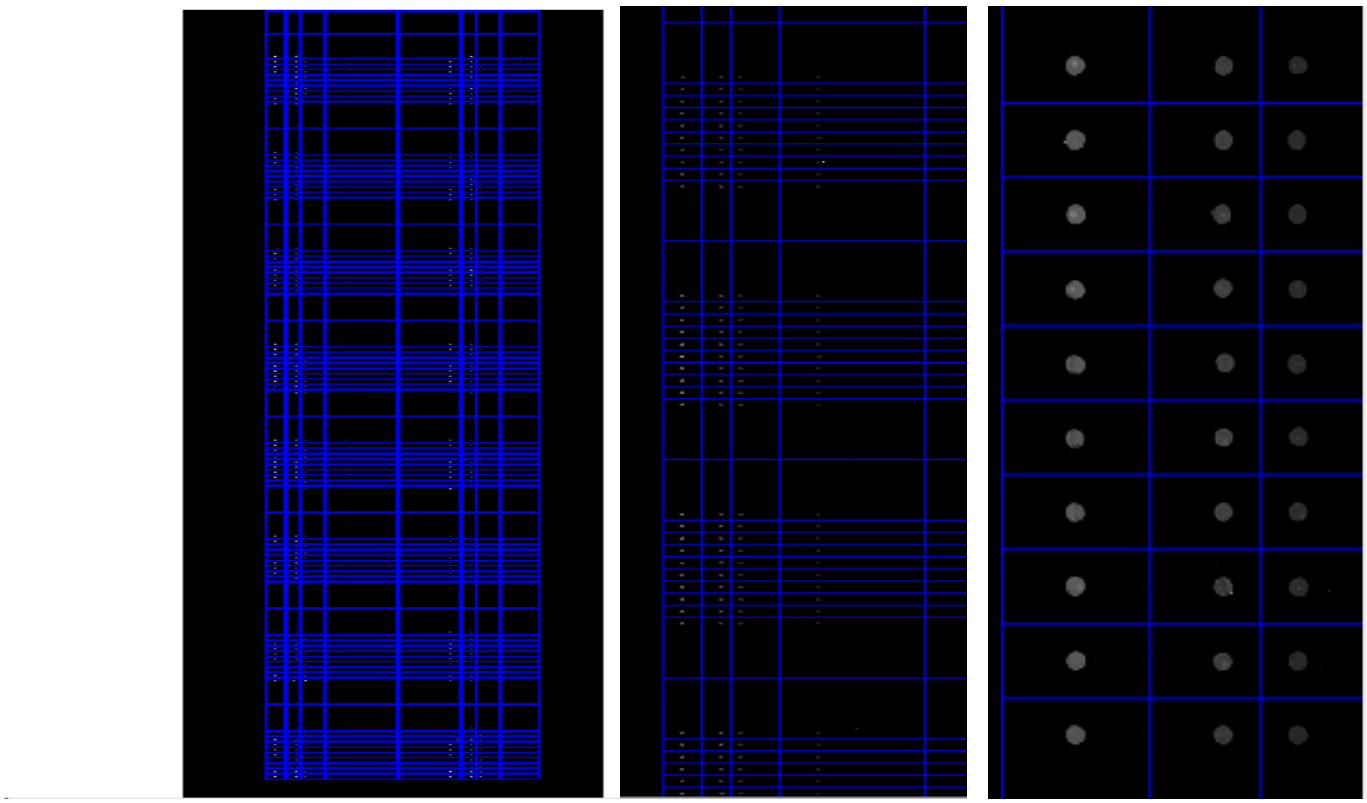


Figure 44 *Gridding results for a microarray image, with an accuracy of 100% and a processing time of 6.51s, of a multiple channels model representing three different colors of fluorescent streptavidin printed on a slide*

Now let's perform the segmentation using a global threshold with a value of 0.902. Figure 45 presents the results:

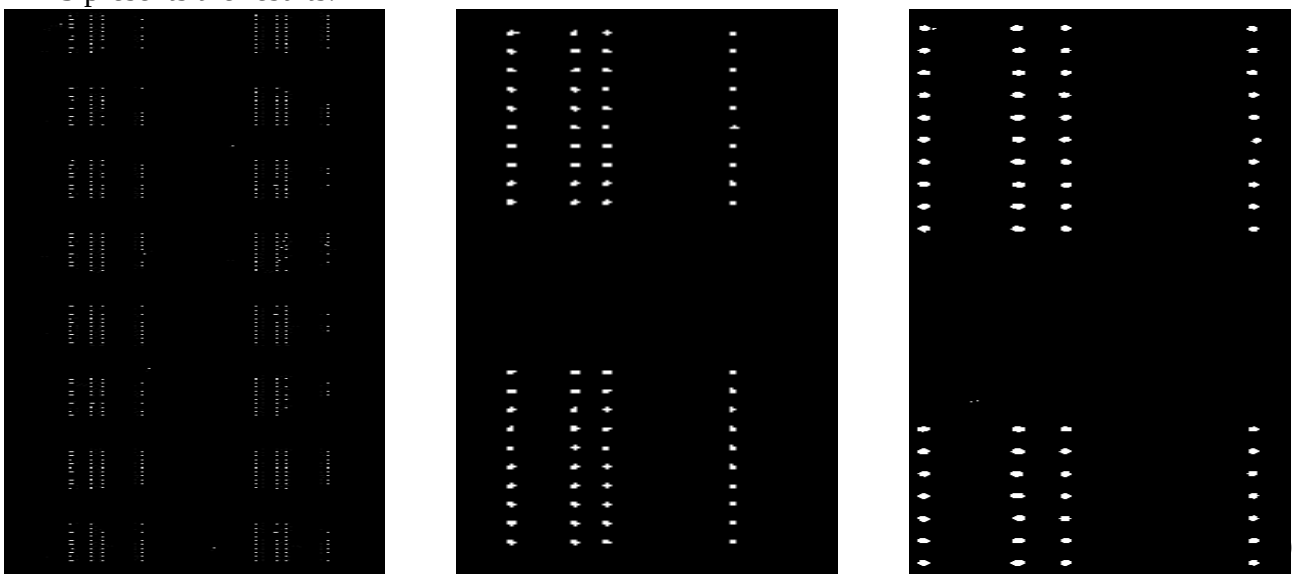


Figure 45 *Results of the global thresholding segmentation of a multiple channels model representing three different colors of fluorescent streptavidin printed on a slide*

Table 17 *Segmentation scores for multiple channels model representing three different colors of fluorescent streptavidin printed on a slide*

Segmentation score parameter	Global thresholding	Adaptive thresholding	Multilevel thresholding (2 levels)	k-Mean clustering (2 clusters)
Jaccard	0.17	0.1680	0.8351	0.8351

It would be interesting to calculate the spots intensity for each channel (Table 18). In this model, it wouldn't be possible to apply the previous method to calculate the expression levels, as we are dealing with three different channels.

Table 18 *Spots quantification of multiple channels model representing three different colors of fluorescent streptavidin printed on a slide*

ROI	Intensity of first channel	Intensity of second channel	Intensity of third channel
1	207	1042	444
2	10	69	400
3	10	50	335
4	589.577	1039.1	1161.4
5	585.955	938.655	1131.9
6	538.230	976.679	1009
7	434.754	972.653	435.228
8	674.111	1044.6	1029.9
9	875.470	2552	1714.3
10	435.289	1430.2	1322
11	16.019	72.841	412.574
12	192.126	148.749	1107.7
13	188.287	159.483	877.406
14	62.277	1065.3	588.737
15	31.233	42.321	309.212
16	449.988	270.140	1140.6
17	110.459	1186	477.200
18	91.653	1227.5	915.302
19	98.208	1441.7	418.879
20	103.332	1341	876.141

The third model that will be used for pipeline validation, is a two-channels microarray taken from a random experiment where the Cy3-green/Cy5-red. Figure 46 shows the enhanced profile along with the autocorrelation function. After following the remaining steps of the gridding, the results were an accuracy of 97.98% and a processing time of: 02:08 seconds. In this case Tx=Ty= 11 µm. As for the global segmentation, the result was similar to previous models (figures 47 and 48), with Jaccard score of 0.1881, this makes sense as this slide seemed to have more weak spots, so it would be more challenging to segment and properly reconstruct all of them. The local adaptive threshold offered a slightly better results with a score of 0.2619. Finally, the multilevel thresholding with level equal 2 and k-mean algorithm offered the best similarity values of 0.9400 and 0.9792.

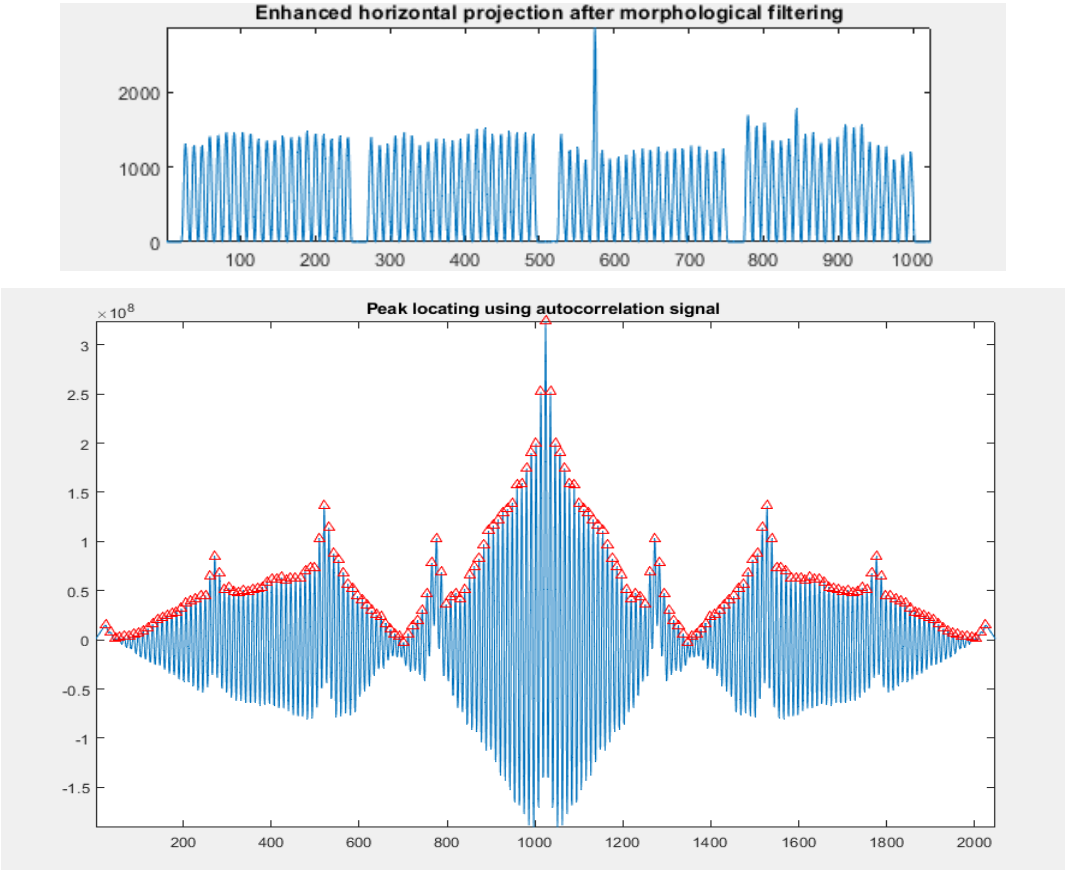


Figure 46 *Horizontal profile of the microarray images along with the corresponding autocorrelation signal of a two-channel microarray image with different dyes for each channel taken from a random biology experiment*

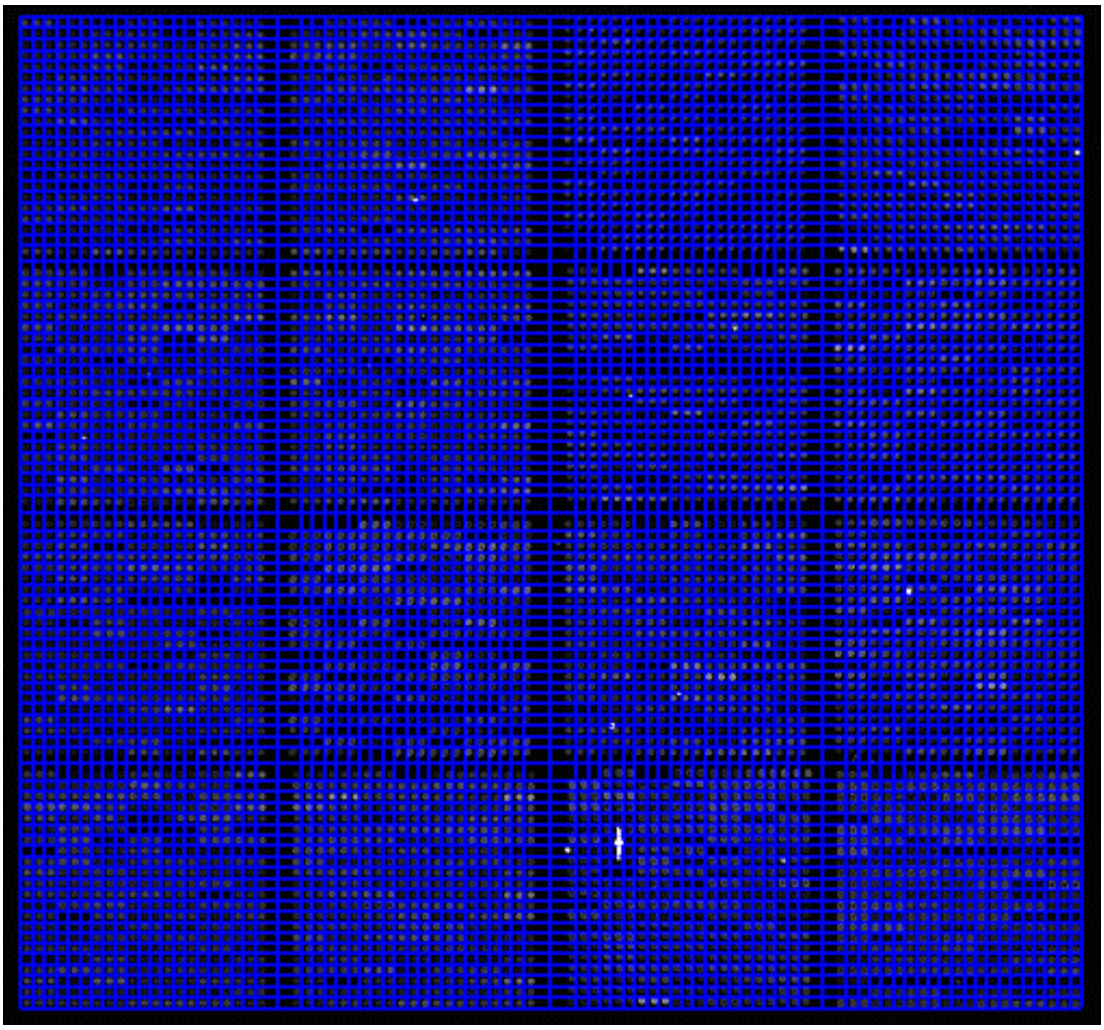
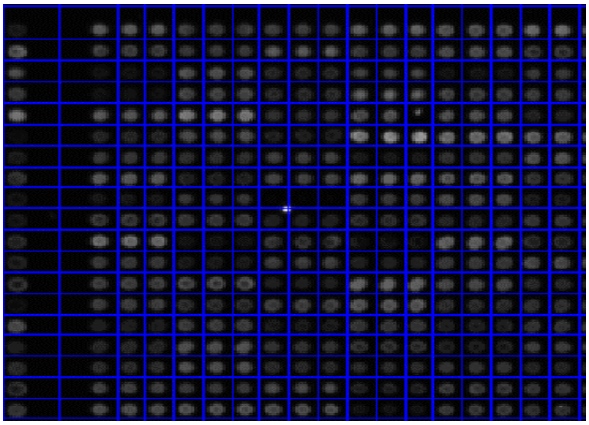


Figure 47 *Automatic gridding results for a slide representing a two-channel microarray image with different dyes for each channel taken from a random biology experiment*



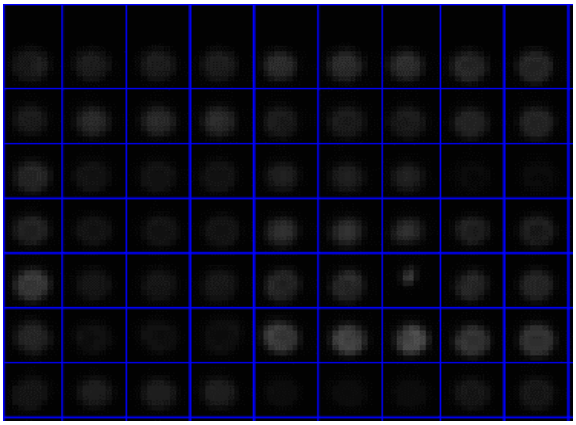


Figure 48 *Closer look at the gridding results showing a high gridding accuracy of 97.98% and a processing time of 02:08 seconds of a two-channel microarray image with different dyes for each channel taken from a random biology experiment*

The expression levels are given by the following equation:

$$Expression\ level = \frac{Log\ (Intensity\ (First\ channel))}{Log(Intensity\ (Second\ channel))} = \frac{Log\ (Intensity\ (Green))}{Log(Intensity\ (Red))}$$

Table 19 *Intensity of spots and expression level calculation from 20 random ROI taken from of a two-channel microarray image with different dyes for each channel taken from a random biology experiment*

ROI	Intensity of first channel	Intensity of second channel	Expression levels
1	882	550	0.205
2	861.500	610.500	0.149
3	922	685.500	0.128
4	1536.1	1322.2	0.065
5	1025.9	882.180	0.066
6	685.902	680.583	0.003
7	1205.3	944.430	0.105
8	1106.1	1075.9	0.034
9	767.298	766.131	0.001
10	817.347	781.263	0.019
11	963.833	739.583	0.115
12	897.294	814.173	0.042
13	1251.6	973.138	0.109
14	797.222	813.854	-0.008
15	1367.8	1186.5	0.061
16	1005.6	912.777	0.420
17	1808	1375.8	0.118
18	1298.2	1169.4	0.045
19	805.840	759.576	0.025
20	1767.4	1438.8	0.089

As seen before, trying to compare the quantification results for the first model by referring to the data generated by ArrayPro Analyzer wasn't informative for the reasons that were previously mentioned. Therefore, a simulation experiment using ImageJ where the true values of the ratios between channels or the spots intensities are known, could help one to evaluate the quantification method of the pipeline in a more objective way. The first step is to open the file and load it to the software as presented by figure 46.

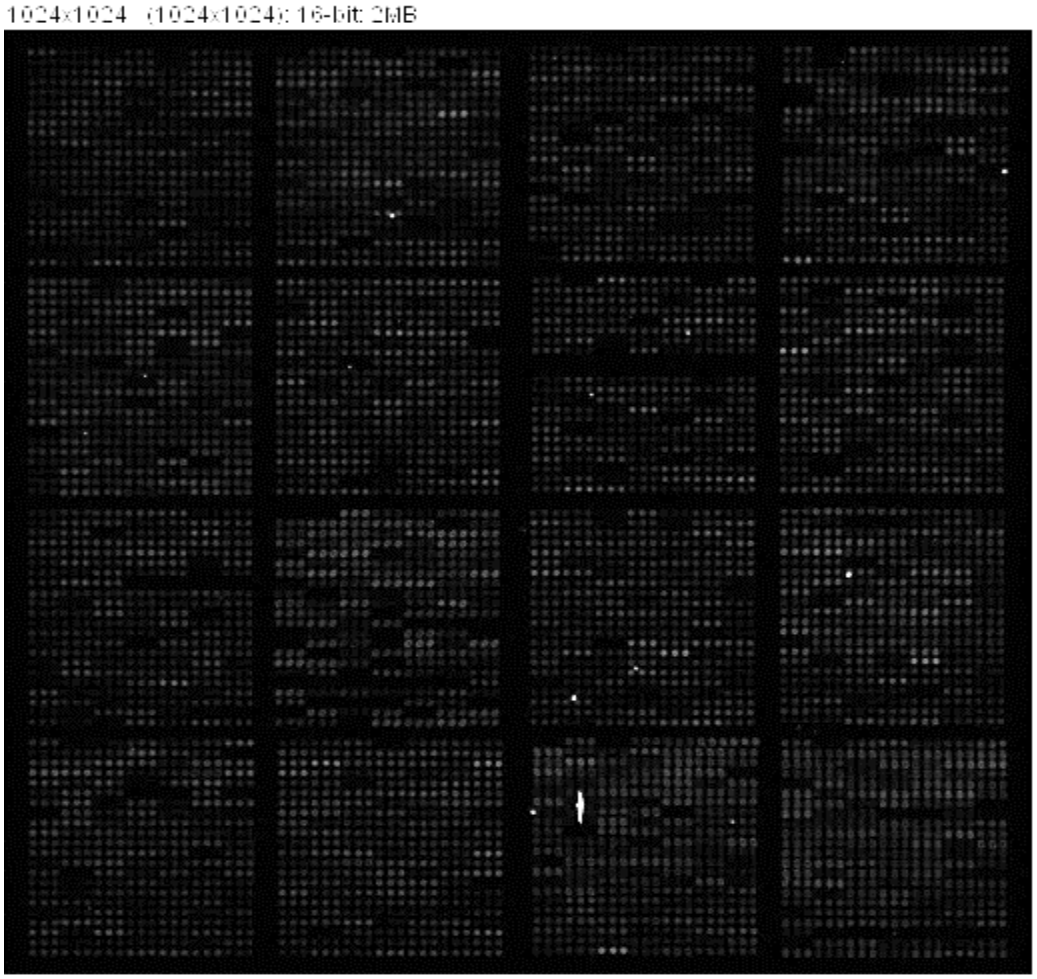


Figure 49 *Loading a microarray image of a two-channel microarray image with different dyes for each channel taken from a random biology experiment using ImageJ software*

We then measure the spot intensities. To measure the intensity of a specific spot, it should be first outlined, then we use the *Analyze/Measure* command. Before estimating this parameter, it would be helpful to perform a background correction, by subtracting the value of the background using the command *Process/Subtract Background*. We set the rolling ball radius to 25 pixels.
It would be helpful to visualize the plot profile to make sure the background is indeed subtracted. To do so, we select a random subgrid as shown in figure 50, and then use the *Analyze/Plot Profile* command on ImageJ to display the profile of that particular grid before and after the background subtraction as presented by figure 51, to make sure the command was correctly executed.

Figure 50 *Subgrid selection using ImageJ for profile plot calculation of a two-channel microarray image with different dyes for each channel taken from a random biology experiment*

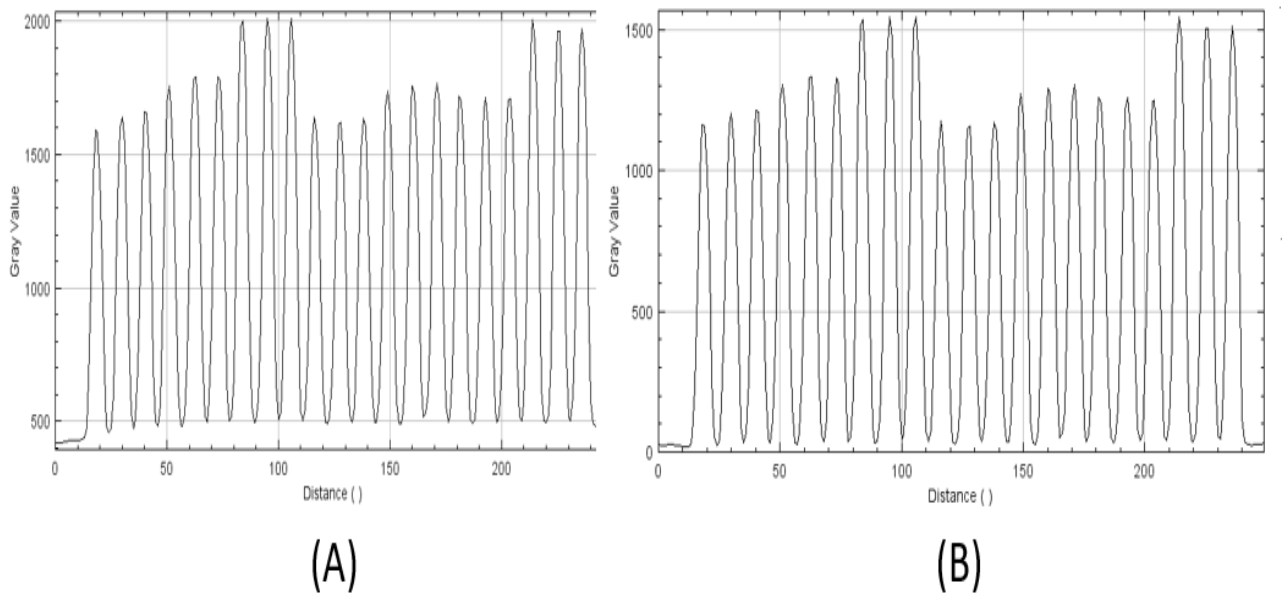


Figure 51 *Profile plot using ImageJ for the slide representing a two-channel microarray image with different dyes for each channel taken from a random biology experiment (A) Before the background subtraction. (B) After background subtraction*

Now that the background is subtracted for a correction calculation, we enable the "Mean" in *Analyze/Set Measurements*, to quantify the spots and to measure the values of this parameter we create a circular selection on different spots with known coordinates or regions of interests. We make sure to choose a similar region of interests that we calculated before using our pipeline to have a more objective comparison. Table 22 shows the true value for the different ROIs, along with the values of the intensities for the same ROIs that were obtained using the pipeline described in this work. The percentage error was also calculated using the following formula, where V_A is the actual observed value, calculated using the pipeline, and V_T is the true value.

$$\text{Percentage error} = \left|\frac{V_A - V_T}{VT}\right|. 100\%$$

Table 20 *Calculation of the percentage error for the spot quantifications of 20 random ROIs of a two-channel microarray image with different dyes for each channel taken from a random biology experiment. The true values of the spot's intensities were obtained using ImageJ*

ROI	Intensity of first channel using pipeline	True value	Percentage error for first channel	Intensity of second channel using pipeline	True value	Percentage error for second channel
1	882	972.722	**9.3%**	550	518.042	**6.16 %**
2	861.500	848,6	**1.52%**	610.500	585.03	**4.35%**
3	922	971.551	**5.10%**	685.500	708.51	**3.24%**
4	1536.1	1555,040	**1.22%**	1322.2	1258.690	**5.04%**
5	1025.9	914.809	**12.14%**	882.180	787.58	**12.01%**
6	685.902	518.439	**32.29%**	680.583	498.17	**36.61%**
7	1205.3	1015.699	**18.66%**	944.430	836.250	**12.93%**
8	1106.1	1024.527	**7.96%**	1075.9	953.850	**12.79%**
9	767.298	604.708	**26.88%**	766.131	549.300	**39.47%**
10	817.347	650.75	**25.60%**	781.263	583.960	**33.78%**
11	963.833	692.037	**39.27%**	739.583	537.200	**37.67%**
12	897.294	741.192	**21.06%**	814.173	698.692	**16.52%**
13	1251.6	1175.576	**6.46%**	973.138	890.12	**9.32%**
14	797.222	645.618	**23.48%**	813.854	663.42	**22.67%**
15	1367.8	1262.902	**8.30%**	1186.5	1098.090	**8.05%**
16	1005.6	910.124	**10.49%**	912.777	754.733	**20.94%**
17	1808	1742.570	**3.75%**	1375.8	1255.230	**9.60%**
18	1298.2	1092.818	**18.79%**	1169.4	988.925	**18.24%**
19	805.840	677.852	**18.88%**	759.576	675.670	**12.41%**
20	1767.4	1351.925	**30.73%**	1438.8	1097.812	**31.06%**

The spots with quantification error below 15% are considered to be correctly quantified. Some other spots weren't correctly quantified as the error value was high. The difference in the

error percentage between channels is mostly due to the difference in the value of the estimated spot spacing, that could slightly change how the spots are placed in regard to the grid.

Another way to test the method robustness to noise is to add noise with different levels and compute the gridding and segmentation parameters. Gaussian noise was added with standards deviations of 25, 150 and 350. Adding noise doesn't seem to affect the accuracy of gridding that much, as the results are still pretty good. The processing time is still short in general, which is a promising, the results are summarized in table 21.

Table 21 *Effect of different levels of Gaussian noise on the gridding accuracy and the processing time of a two-channel microarray image with different dyes for each channel taken from a random biology experiment*

Noise standard deviation value	Accuracy of gridding	Processing time (s)
25	94.38%	01.95
150	93.73%	02.03
350	92%	02.12

Having a large variation in the spots intensities could make it challenging to correctly segment some weak spots, as seen in the previous segmentation results from the different models. One of the ways to compensate for that variation is by applying a logarithmic transformation to the base 2 on the data to equalize and minimize large variations in magnitude between the different spots. The transformation was applied on the third model, and the new data is shown in figure 52 [117] The global segmentation threshold value moved from 0.03 before applying the logarithmic transformation to 0.63, and the Jaccard parameter moved from 0.18 to 0.40. A lot of weak spots that were previously missed during the global segmentation, were reconstructed after applying the logarithmic transformation (figure 53). Even though there were other weak spots that were still missed, there was still an improvement.

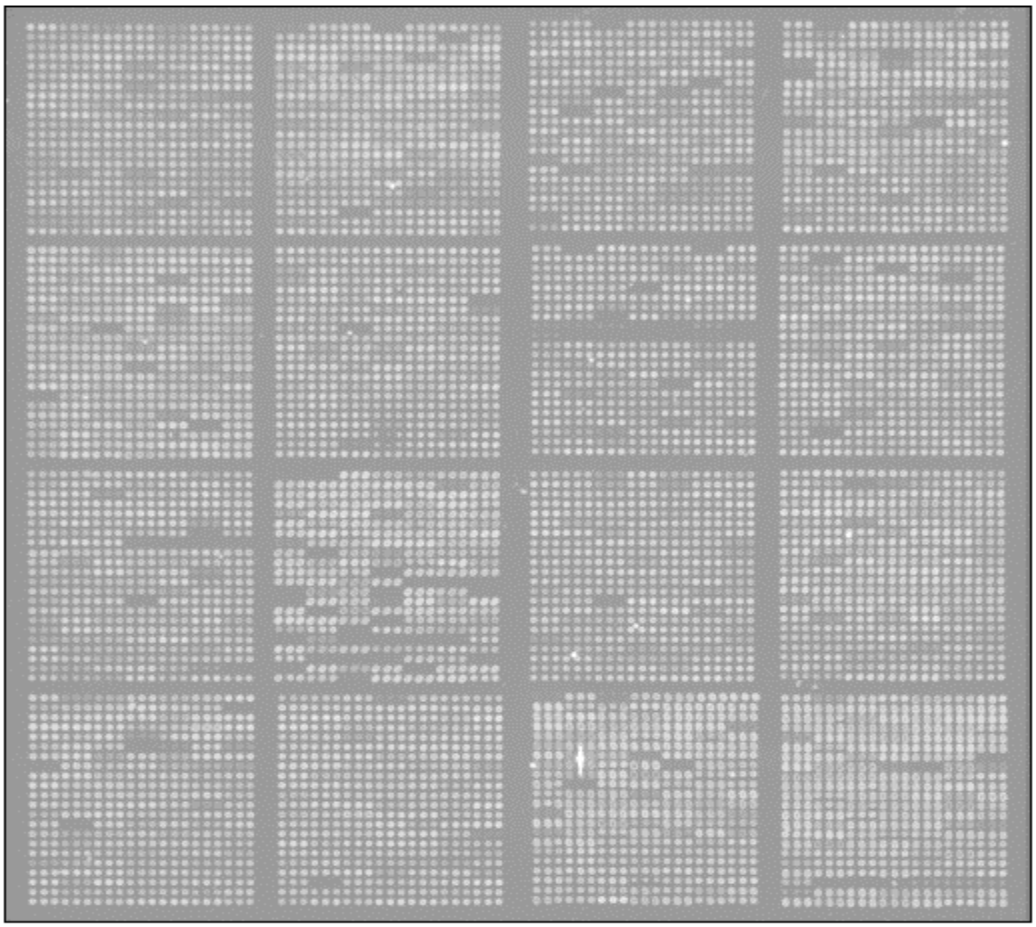


Figure 52 *Display of a two-channel microarray image with different dyes for each channel taken from a random biology experiment after applying logarithmic transformation to improve the segmentation results and restore some missed weak spots*

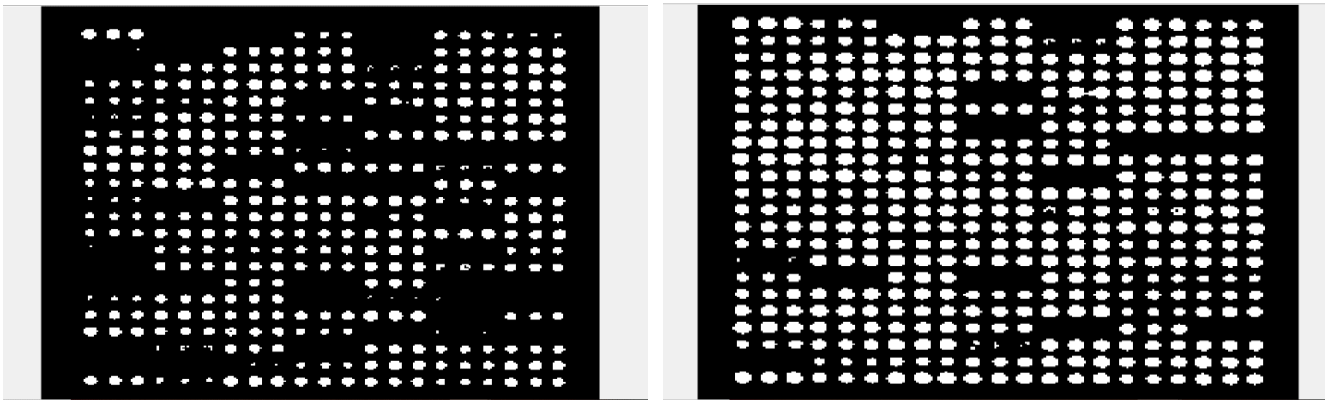


Figure 53 *Results of the global segmentation after applying the logarithmic transformation. on a two-channel microarray image with different dyes for each channel taken from a random biology experiment (left) global segmentation without logarithmic transformation. (right) global segmentation without logarithmic transformation*

Chapter 5: Conclusion

5.1 Summary

An automated pipeline to study microarray data was presented in this project. The pipeline was first developed to analyze microarray images generated from an experiment based on antibody microarray to capture and investigate different types of exosomal proteins. The data were obtained from three different slides, and ImageJ software was used in the early stages of the image preprocessing, and as a tool to adjust some relevant parameters such as noise level and intensity values.

We first developed a gridding algorithm based on image projection techniques, and we measured different gridding parameters such as accuracy and processing time. The method showed a high gridding accuracy, as well as a short processing time in all of the three slides. The effect that the preprocessing step or the background removal had on the gridding results was also investigated, by running the algorithm without using any morphological filter. A decline in the accuracy results was observed when removing the filters, as more spots were missed, or inaccurately gridded, which showed the importance of the denoising step while analyzing microarray images.

After gridding the microarray images, for each slide, the background needed to be separated from the intensity signals by exploring different thresholding-based segmentation methods by implementing global thresholding, adaptive thresholding, and multilevel thresholding with two levels. In order to assess the segmentation performance, the similarity parameter Jaccard was used. When using the global segmentation, a lot of spots with weak intensities were missed, and even though the thresholding techniques that were used had a similar working mechanism in general, the segmentation results showed that their impact on signal and background separation was different, and a comparison between the segmentation scores for these methods showed better results when using multilevel thresholding modality. The previous issues that were encountered using segmentation methods have led to exploring the use of the k-mean algorithm, which is a clustering machine learning method. We performed the segmentation using a different number of clusters while observing the effect of this parameter on the segmentation scores. Using a low number of clusters was more effective for the segmentation of the microarray images for all the microarray images of the slides from the antibody microarray experiment. After a set of comparisons with the previous segmentation methods that were presented, it seemed that using machine learning for segmentation had dramatically improved the segmentation scores, which proved the utility of machine learning in microarray data research.

Different regions of interests (ROI) were selected to calculate the intensities of random spots with known coordinates, as well as their expression levels under two scenarios: the case where the background noise was removed using morphological filters, and the case where those filters weren’t implemented. The results showed that not subtracting the background contributed to having lower values of the spot intensities. Besides, the statistical behavior of some ROI was also studied, using the intensities calculated from the pipeline versus the one generated by ArrayPro Analyzer, to get a better idea of the quality of the experiment. While some values were close for ROI of the first channel from the first slide of the antibody microarray experiment, it was

challenging to determine which quantification was better without knowing the true value of the intensities. This has led us to explore more ways to objectively assess the effectiveness of the proposed pipeline through simulation using ImageJ.

To validate that the proposed pipeline was universal and to assess its performance, other data generated from the Juncker lab were used. The first model was a multiple channel slide consisting of different proportions of three different colours of fluorescent streptavidin printed on a slide. We reported an accuracy of 100%- and 6.51-seconds processing time, which was very good. Another model for validation, was a two-channel microarray image (Cy3-green/Cy5-red) images from a random experiment. We used the second model to test the application of the logarithmic transformation to compensate for the variation in the spot intensities, this has given us better segmentation results as many spots that were missed before were properly reconstructed after applying the transformation on the data. As for the quantification evaluation, the previous two channel model was used in combination with ImageJ to locate the true values and compare them with the ones obtained using the proposed pipeline. For some ROIs, we obtained promising quantification values with error percentage as low as 1.52%. We also demonstrated the robustness of the proposed methods to noisiness by adding Gaussian noise with different standard deviations values. The gridding results were promising despite observing a slight decline in the gridding accuracy when using higher standards deviations of noise.

5.2 Future work

Despite the high values of the gridding accuracy that were obtained across the several models that were used to validate the pipeline, and the positive result of the pipeline analysis, it is still essential to describe some limitations of the pipeline before fully adopting it. For instance, some user assistance may be required in the case of processing rotated images, which could reduce the level of automatization of the pipeline and delay the processing time. In the case of relying on the K-means clustering algorithm for the segmentation step, choosing the right number of clusters could be tricky sometimes, since the pipeline relies on a random selection of the number of clusters, which is a process of trailer and error that could be computationally time consuming, especially when dealing with images with large data points such as microarray images. and the application of the algorithm itself is limited by knowing an optimal number of clusters. Therefore, exploring ways to automate the selection of the number of clusters K would help with saving more time[109]. Besides, the ability of the gridding method to remove background noise using the morphological filters wasn’t extensively explored. For instance, removing those filters created some grids displacements as seen before, which proved the efficiency of the denoising step, and the image projections were visually enhanced compared to the initial profiles that were computed, but the pipeline doesn’t rely on a specific metric to evaluate and quantify the noise removal ability of the gridding, which could be another limitation of the method.

Important next steps would be to test the pipeline on other simulated data to assess its robustness while dealing with other type of contaminations such as non-specific hybridizations and dust, by computing the previous gridding parameters and segmentation score. Measuring the performance of microarray segmentation algorithms is a challenging issue and lacks research. In general, the most natural way for evaluating the segmentation algorithms is by measuring the segmentation error on a pixel-level, which can't be achievable in several microarray experiments[4] This could be done by evaluating the pixel accuracy by finding the values of the following: the true positive, true negative, false positive, and false negative values. This helps with reporting the percent of pixels in the image which were correctly classified [118] Therefore, it would be also helpful to use more similar metrics to evaluate the performance of the segmentation methods used in this work.

The analysis could also include microarray images that are rotated in different directions following different angles and study the behavior of the autocorrelation function, and what the gridding results would look like in that case. The angle could be considered as a different feature that could be fed to the algorithm. As far as features extraction is concerned, it is considered to be a very time-consuming in some cases, and this problem has been successfully resolved by implementing various deep neural networks[119], which could enable the extraction of new features that have never been discovered before while analyzing the data, such as spot centers, spots diameter, shape and other features that can ameliorate the segmentation phase, and since deep learning doesn't require any manual feature extraction, there won't be concerned when it comes to the processing time, and so basically exploring the use of deep learning while using the proposed pipeline and data from this project could be interesting [120].

Furthermore, there are several variations of K-means clustering algorithms in literature such as Fuzzy C-Means. The two algorithms have similarities in their working mechanism when it comes to image segmentation, so it could be possible to explore how they would perform when presented with similar microarray images that were used in this work. It would be interesting to combine the k-means algorithm with other stages of pre-processing and post-processing used to provide maximum segmentation results[121]

Other adjustments may be needed, such as using different type of transformations such as the one listed on the pre analysis issues section of the second chapter and compare it with the logarithmic transformation that was used in this work, to better assess which transformation is better when it comes to dealing with missed spots and similar data. The work has also presented both the global threshold and local thresholding methods separately, and each method has failed to reconstruct all the images that had weak intensities. Besides, the global properties of the image are characterized by the mean values of different pixel classes and the boundary of the different regions, while the local properties are characterized by the interactions of neighboring pixels and the image edge, as shown by the adaptive local thresholding method used before[112]. It could be possible to try a logical combination of both segmentations to get the best of both of both thresholding variations and assess the results.

Microarray data comes in different qualities, in this project the quality of the images were almost the same, so using low, medium and high-quality images to compare would strengthen the proposed pipeline. Most of the available microarray gridding approaches require human

intervention; to specify landmarks, or to precisely locate individual spots [122]. Comparing the proposed gridding method with those methods, as well as with other automated methods from the literature, to make more objective assessment of its efficiency could also be added to the results section of the project.

6. Abbreviations

Ac	Accuracy
AD	Alzheimer disease
BIC	Bayesian Information Criterion
CFSE	Carboxyfluorescein succinimidyl ester
CSF	Cerebral spinal fluid
CMOS	Complementary metal-oxide-semiconductor
ESCRT	Endosomal sorting complexes required for transport
ELISA	Enzyme-linked immunosorbent assay
EVs	Extracellular vesicles
GA	Genetic algorithm
hsc70	Heat-shock cognate protein 70
Hsp70:	Heat Shock Protein 70
Hsp90	Heat shock protein 90
HC	Horizontal centers
hCSF	Human cerebrospinal fluid volume
ML	Machine learning
MA	Matarray
MSB	Model-based segmentation
MVs	Microvesicles
MVBs	Multivesicular bodies
ROI	Region of interest
SRG	Seeded region growing
SP-IRIS	Single particle interferometric reflectance imaging sensor
SPs	Stimulating the surface plasmons
SR	Spots regions
SVM	Support victor machine
TSG101	Tumor Susceptibility 101
VC	Vertical centers

7. Bibliography

1. Hall, D.A., J. Ptacek, and M. Snyder, *Protein microarray technology.* Mech Ageing Dev, 2007. **128**(1): p. 161-7.
2. Daaboul, G.G., et al., *Digital Detection of Exosomes by Interferometric Imaging.* Sci Rep, 2016. **6**: p. 37246.
3. Lysov, Y., et al., *Microarray analyzer based on wide field fluorescent microscopy with laser illumination and a device for speckle suppression.* Biomed Opt Express, 2017. **8**(11): p. 4798-4810.
4. Lehmussola, A., P. Ruusuvuori, and O. Yli-Harja, *Evaluating the performance of microarray segmentation algorithms.* Bioinformatics, 2006. **22**(23): p. 2910-7.
5. Ahmad, M.M., A.B. Jambek, and M.Y. bin Mashor, *A study on microarray image gridding techniques for DNA analysis*, in *2014 2nd International Conference on Electronic Design (ICED)*. 2014. p. 171-175.
6. Zacharia, E. and D. Maroulis, *An original genetic approach to the fully automatic gridding of microarray images.* IEEE Trans Med Imaging, 2008. **27**(6): p. 805-13.
7. Fouad, I., *Automatic and Accurate Segmentation of Gridded cDNA Microarray Images Using Different Methods.* Advances in computing, 2014. **4**: p. 41-54.
8. Ahmed, A.A., et al., *Microarray segmentation methods significantly influence data precision.* Nucleic Acids Res, 2004. **32**(5): p. e50.
9. Shao, G., et al., *Automatic microarray image segmentation with clustering-based algorithms.* PLoS One, 2019. **14**(1): p. e0210075.
10. Nolte-'t Hoen, E., et al., *Extracellular vesicles and viruses: Are they close relatives?* Proc Natl Acad Sci U S A, 2016. **113**(33): p. 9155-61.
11. Doyle, L.M. and M.Z. Wang, *Overview of Extracellular Vesicles, Their Origin, Composition, Purpose, and Methods for Exosome Isolation and Analysis.* Cells, 2019. **8**(7).
12. Jella, K.K., et al., *Exosomes, Their Biogenesis and Role in Inter-Cellular Communication, Tumor Microenvironment and Cancer Immunotherapy.* Vaccines (Basel), 2018. **6**(4).
13. Hu, Q., et al., *Clinical applications of exosome membrane proteins.* Precis Clin Med, 2020. **3**(1): p. 54-66.
14. Li, W., et al., *Role of exosomal proteins in cancer diagnosis.* Mol Cancer, 2017. **16**(1): p. 145.
15. Hartmann, A., et al., *Exosomes and the Prion Protein: More than One Truth.* Front Neurosci, 2017. **11**: p. 194.
16. Mohamed, M. and A. Sheikh, *Magnetic Resonance Spectroscopy in Major Depressive Disorder.* International journal of emergency mental health, 2015. **17**: p. 167-187.
17. Chen, Z., et al., *Current applications of antibody microarrays.* Clin Proteomics, 2018. **15**: p. 7.
18. Kuo, W.P., et al., *A primer on gene expression and microarrays for machine learning researchers.* J Biomed Inform, 2004. **37**(4): p. 293-303.
19. Dubitzky, W., M. Granzow, and D. Berrar, *Data Mining and Machine Learning Methods for Microarray Analysis*, in *Methods of Microarray Data Analysis*. 2002. p. 5-22.
20. Eldh, M., et al., *Importance of RNA isolation methods for analysis of exosomal RNA: evaluation of different methods.* (1872-9142 (Electronic)).
21. Javeed, N. and D. Mukhopadhyay, *Exosomes and their role in the micro-/macro-environment: a comprehensive review.* J Biomed Res, 2017. **31**(5): p. 386-394.
22. Mohammed, F., et al., *An Efficient Fully Automated Method for Gridding Microarray Images.* American Journal of Biomedical Engineering, 2012. **2**: p. 115-119.

23. Ghosh, M., et al., *Genetic algorithm based cancerous gene identification from microarray data using ensemble of filter methods.* Medical & Biological Engineering & Computing, 2019. **57**(1): p. 159-176.
24. Jaksik, R., et al., *Microarray experiments and factors which affect their reliability.* Biol Direct, 2015. **10**: p. 46.
25. Helmy, A.K. and G.S. El-taweel, *Regular gridding and segmentation for microarray images.* Computers & Electrical Engineering, 2013. **39**(7): p. 2173-2182.
26. Qin, L., et al., *Spot Detection and Image Segmentation in DNA Microarray Data.* Applied Bioinformatics, 2005. **4**(1): p. 1-11.
27. Wang, B. and Y. Xi, *Challenges for MicroRNA Microarray Data Analysis.* (2076-3905 (Print)).
28. Anandhavalli, M., C. Mishra, and M. Ghose, *Analysis of Microarray Image Spots Intensity: A Comparative Study.* International Journal of Computer Theory and Engineering, 2009. **1**: p. 1793-8201.
29. Cho, J., et al., *LISA: a MATLAB package for Longitudinal Image Sequence Analysis.* 2019. **abs/1902.06131**.
30. Nykter, M., et al., *Simulation of microarray data with realistic characteristics.* BMC Bioinformatics, 2006. **7**: p. 349.
31. Mukhtar, M., A. Jambek, and M. Mashor, *Image gridding algorithm for DNA microarray analyser*. 2016. 452-457.
32. Sivakumar, V. and V. Murugesh. *A brief study of image segmentation using Thresholding Technique on a Noisy Image*. in *International Conference on Information Communication and Embedded Systems (ICICES2014)*. 2014.
33. Rehna, V.J. and G. Raju. *Signal extraction from microarray images for gene array data analysis*. in *2010 The 2nd International Conference on Computer and Automation Engineering (ICCAE)*. 2010.
34. Quackenbush, J., *Microarray data normalization and transformation.* Nat Genet, 2002. **32 Suppl**: p. 496-501.
35. Yanjun, F., S. Kai, and L. Jun. *A Microarray Image Gridding Method Based on Projection Transformation and Power Spectral Analysis*. in *2015 International Symposium on Computers & Informatics*. 2015. Atlantis Press.
36. Bajcsy, P., *An Overview of DNA Microarray Grid Alignment and Foreground Separation Approaches.* EURASIP Journal on Advances in Signal Processing, 2006. **2006**(1).
37. Bariamis, D., D.K. Iakovidis, and D. Maroulis *M3G: maximum margin microarray gridding*. BMC bioinformatics, 2010. **11**, 49 DOI: 10.1186/1471-2105-11-49.
38. Karthik, S.A., et al. *A Review on Gridding Techniques of Microarray Images*. in *2019 1st International Conference on Advanced Technologies in Intelligent Control, Environment, Computing & Communication Engineering (ICATIECE)*. 2019.
39. Schoot, R., et al., *A Gentle Introduction to Bayesian Analysis: Applications to Developmental Research.* Child development, 2013. **85**.
40. Zacharia, E. and D. Maroulis, *An Original Genetic Approach to the Fully Automatic Gridding of Microarray Images.* IEEE Transactions on Medical Imaging, 2008. **27**(6): p. 805-813.
41. Zacharia, E. and D. Maroulis. *Microarray image gridding via an evolutionary algorithm*. in *2008 15th IEEE International Conference on Image Processing*. 2008.
42. Luque-Baena, R.M., et al., *Robust gene signatures from microarray data using genetic algorithms enriched with biological pathway keywords.* Journal of Biomedical Informatics, 2014. **49**: p. 32-44.
43. Kashyap, R. and P. Gautam, *Fast Medical Image Segmentation Using Energy-Based Method*. 2016. p. 1017-1042.

44. Farouk, R.M. and M.A. SayedElahl, *Microarray spot segmentation algorithm based on integro-differential operator.* Egyptian Informatics Journal, 2019. **20**(3): p. 173-178.
45. Thumu, S., *Analysis and CDNA Microarray Image Segmentation Based on Hough Circle Transform*. 2019.
46. Barra, V., *Robust segmentation and analysis of DNA microarray spots using an adaptative split and merge algorithm.* Computer Methods and Programs in Biomedicine, 2006. **81**(2): p. 174-180.
47. Yang, Y.H., M.J. Buckley, and T.P. Speed, *Analysis of cDNA microarray images.* Briefings in Bioinformatics, 2001. **2**(4): p. 341-349.
48. Fouad, I., M. Mabrouk, and A. Sharawi, *Automatic Segmentation of cDNA Microarray Images Using Different Methods.* Journal of Biomedical Engineering and Medical Imaging, 2014. **1**.
49. Fan, J., et al., *Seeded region growing: an extensive and comparative study.* Pattern Recognition Letters, 2005. **26**(8): p. 1139-1156.
50. Delmo, J. and C. Refugio, *Empirical Research on Mann-Whitney U-test*. 2018.
51. Chen Y Fau - Dougherty, E.R., M.L. Dougherty Er Fau - Bittner, and M.L. Bittner, *Ratio-based decisions and the quantitative analysis of cDNA microarray images.* (1083-3668 (Print)).
52. Li, Q., et al., *Donuts, scratches and blanks: robust model-based segmentation of microarray images.* Bioinformatics, 2005. **21**(12): p. 2875-2882.
53. Wang, X., S.W. Ghosh S Fau - Guo, and S.W. Guo, *Quantitative quality control in microarray image processing and data acquisition.* (1362-4962 (Electronic)).
54. Held, G.A., G. Grinstein, and Y. Tu, *Modeling of DNA microarray data by using physical properties of hybridization.* Proceedings of the National Academy of Sciences, 2003. **100**(13): p. 7575.
55. Nagarajan, R. and C. Peterson, *Identifying spots in microarray images.* IEEE transactions on nanobioscience, 2002. **1**: p. 78-84.
56. Abels, E.R. and X.O. Breakefield, *Introduction to Extracellular Vesicles: Biogenesis, RNA Cargo Selection, Content, Release, and Uptake.* Cell Mol Neurobiol, 2016. **36**(3): p. 301-12.
57. Yang, Y., et al., *Interferometric plasmonic imaging and detection of single exosomes.* Proc Natl Acad Sci U S A, 2018. **115**(41): p. 10275-10280.
58. Jorgensen, M., et al., *Extracellular Vesicle (EV) Array: microarray capturing of exosomes and other extracellular vesicles for multiplexed phenotyping.* J Extracell Vesicles, 2013. **2**.
59. Chicco, D., *Ten quick tips for machine learning in computational biology.* BioData Mining, 2017. **10**(1): p. 35.
60. Xu, C. and S.A. Jackson, *Machine learning and complex biological data.* Genome Biology, 2019. **20**(1): p. 76.
61. Noorbakhsh, J., et al., *Machine Learning in Biology and Medicine.* Advances in Molecular Pathology, 2019. **2**(1): p. 143-152.
62. Deo, R.C., *Machine Learning in Medicine.* Circulation, 2015. **132**(20): p. 1920-1930.
63. Kukreja, M., S.A. Johnston, and P. Stafford, *Comparative study of classification algorithms for immunosignaturing data.* BMC Bioinformatics, 2012. **13**(1): p. 139.
64. Vidyasagar, M., *Machine learning methods in the computational biology of cancer.* Proceedings. Mathematical, physical, and engineering sciences, 2014. **470**(2167): p. 20140081-20140081.
65. Tarca, A.L., et al., *Machine Learning and Its Applications to Biology.* PLOS Computational Biology, 2007. **3**(6): p. e116.
66. Sharma, D. and N. Kumar, *A Review on Machine Learning Algorithms, Tasks and Applications.* 2017. **6**: p. 2278-1323.
67. Jose, J., et al., *Case study on enhanced K-means algorithm for bioinformatics data clustering.* International Journal of Applied Engineering Research, 2017. **12**: p. 15147-15151.

68. Tadist, K., et al., *Feature selection methods and genomic big data: a systematic review.* Journal of Big Data, 2019. **6**(1): p. 79.
69. Tan, C.S., et al., *A Review of Feature Extraction Software for Microarray Gene Expression Data.* BioMed Research International, 2014. **2014**: p. 213656.
70. Sampas, N., et al., *Feature extraction and clustering tools for analysing gene expression data from DNA microarrays.* Nature Genetics, 1999. **23**(3): p. 71-72.
71. Li, Z., W. Xie, and T. Liu, *Efficient feature selection and classification for microarray data.* PLOS ONE, 2018. **13**(8): p. e0202167.
72. Khaire, U.M. and R. Dhanalakshmi, *Stability of feature selection algorithm: A review.* Journal of King Saud University - Computer and Information Sciences, 2019.
73. Sánchez-Maroño, N., A. Alonso-Betanzos, and R.M. Calvo-Estévez. *A Wrapper Method for Feature Selection in Multiple Classes Datasets*. in *Bio-Inspired Systems: Computational and Ambient Intelligence*. 2009. Berlin, Heidelberg: Springer Berlin Heidelberg.
74. Dash, M. and H. Liu, *Consistency-based search in feature selection.* Artificial Intelligence, 2003. **151**(1): p. 155-176.
75. Suppers, A., A. van Gool, and H. Wessels, *Integrated Chemometrics and Statistics to Drive Successful Proteomics Biomarker Discovery.* Proteomes, 2018. **6**: p. 20.
76. Bommert, A., et al., *Benchmark for filter methods for feature selection in high-dimensional classification data.* Computational Statistics & Data Analysis, 2020. **143**: p. 106839.
77. Danasingh, A.A., S. Balamurugan, and J.L. Epiphany, *Literature Review on Feature Selection Methods for High-Dimensional Data.* International Journal of Computer Applications, 2016. **136**.
78. Rückstieß, T., C. Osendorfer, and P. van der Smagt, *Sequential Feature Selection for Classification*. 2011.
79. Sun, S., Q. Peng, and A. Shakoor, *A kernel-based multivariate feature selection method for microarray data classification.* PloS one, 2014. **9**(7): p. e102541-e102541.
80. Molina, L., L. Belanche, and À. Nebot, *Feature Selection Algorithms: A Survey and Experimental Evaluation*. Vol. 4. 2002. 306-313.
81. Nnamoko, N., et al., *Evaluation of Filter and Wrapper Methods for Feature Selection in Supervised Machine Learning*. 2014.
82. Bahojb Imani, M., M. Keyvanpour, and R. Azmi, *A novel embedded feature selection method: A comparative study in the application of text categorization.* Applied Artificial Intelligence, 2013. **27**: p. 408-427.
83. Roffo, G., *Feature Selection Library (MATLAB Toolbox)*. 2016.
84. Saeys, Y., I. Inza, and P. Larrañaga, *A review of feature selection techniques in bioinformatics.* Bioinformatics, 2007. **23**(19): p. 2507-2517.
85. Minaee, S., et al., *Image Segmentation Using Deep Learning: A Survey*. 2020.
86. A Sukanya, R.R., *Machine Learning Techniques for Microarray Image Segmentation: A Review.* INTERNATIONAL JOURNAL OF ENGINEERING RESEARCH & TECHNOLOGY (IJERT) NCICCT, 2014. **Volume 2** (Issue 05).
87. Schmidt, H., *SBaddon: high performance simulation for the Systems Biology Toolbox for MATLAB.* (1367-4811 (Electronic)).
88. Schindelin, J., et al., *The ImageJ ecosystem: An open platform for biomedical image analysis.* Mol Reprod Dev, 2015. **82**(7-8): p. 518-29.
89. Smyth, G.K. and N.S. Altman, *Separate-channel analysis of two-channel microarrays: recovering inter-spot information.* BMC Bioinformatics, 2013. **14**(1): p. 165.
90. Wang, G., et al., *Morphological background detection and illumination normalization of text image with poor lighting.* PLoS One, 2014. **9**(11): p. e110991.

91. Wang, G., et al., *Morphological Background Detection and Illumination Normalization of Text Image with Poor Lighting.* PloS one, 2014. **9**: p. e110991.
92. Srisha, R. and A. Khan, *Morphological Operations for Image Processing : Understanding and its Applications*. 2013.
93. Haralick, R.M., S.R. Sternberg, and X. Zhuang, *Image Analysis Using Mathematical Morphology.* IEEE Transactions on Pattern Analysis and Machine Intelligence, 1987. **PAMI-9**(4): p. 532-550.
94. Maghsoumi, H. and H. Aghababa, *An Impressive Quantum Prescription of Image Processing Algorithms*. 2020.
95. Mat Said, K.A., A. Jambek, and N. Sulaiman, *A study of image processing using morphological opening and closing processes.* International Journal of Control Theory and Applications, 2016. **9**: p. 15-21.
96. Song, J. and E.J. Delp, *The analysis of morphological filters with multiple structuring elements.* Computer Vision, Graphics, and Image Processing, 1990. **50**(3): p. 308-328.
97. A, M. and B. Ramasamy, *Efficient 2-D Structuring Element for Noise Removal of Grayscale Images using Morphological Operations.* International Journal of Computer Applications, 2016. **143**: p. 24-28.
98. Ves, E., et al., *Selecting the structuring element for morphological texture classification.* Pattern Analysis and Applications, 2006. **9**: p. 48-57.
99. Li, T., et al., *Contrast enhancement for cDNA microarray image based on fourth-order moment.* Signal, Image and Video Processing, 2018. **12**(6): p. 1069-1077.
100. Arora, S., et al., *Multilevel thresholding for image segmentation through a fast statistical recursive algorithm.* Pattern Recognition Letters, 2008. **29**(2): p. 119-125.
101. Bertels, J., et al., *Optimizing the Dice Score and Jaccard Index for Medical Image Segmentation: Theory & Practice*. 2019.
102. Schindelin, J., et al., *Fiji: an open-source platform for biological-image analysis.* Nature Methods, 2012. **9**(7): p. 676-682.
103. Hirata, R., et al., *Microarray gridding by mathematical morphology.* Proceedings XIV Brazilian Symposium on Computer Graphics and Image Processing, 2001: p. 112-119.
104. Hirata Jr, R., et al., *Segmentation of Microarray Images by Mathematical Morphology.* Real-Time Imaging, 2002. **8**: p. 491-505.
105. Le Brese, C. and J.J. Zou, *Automatic Gridding of Rotated Microarray Images.* 2006 International Conference on Biomedical and Pharmaceutical Engineering, 2006: p. 34-38.
106. Dibella, E., et al., *A Comparison of rotation-based methods for iterative reconstruction algorithms.* Nuclear Science, IEEE Transactions on, 1997. **43**: p. 3370-3376.
107. Bengtsson, A. and H. Bengtsson, *Microarray image analysis: background estimation using quantile and morphological filters.* BMC bioinformatics, 2006. **7**: p. 96-96.
108. Lakshmanan, M., H. Karnan, and S. Natarajan, *Chapter 14 - Smart diagnosis of cardiac arrhythmias using optimal feature rank score algorithm for solar based energy storage ECG acquisition system*, in *Smart Healthcare for Disease Diagnosis and Prevention*, S. Paul and D. Bhatia, Editors. 2020, Academic Press. p. 125-139.
109. Li, L., *Research on target feature extraction and location positioning with machine learning algorithm.* Journal of Intelligent Systems, 2020. **30**(1): p. 429-437.
110. Balafar, M.A. *Segmentation evaluation*. 2021; Available from: https://www.mathworks.com/matlabcentral/fileexchange/29737-segmentation-evaluatation.
111. *Local Adaptive Thresholding*, in *Encyclopedia of Biometrics*, S.Z. Li and A. Jain, Editors. 2009, Springer US: Boston, MA. p. 939-939.
112. Wang, Z., *Image segmentation by combining the global and local properties.* Expert Systems with Applications, 2017. **87**: p. 30-40.

113. Sivalakshmi, B. and N.N. Rao. *Microarray Image Analysis Using Adaptive Data Clustering Algorithms*. 2019.
114. Rama, D., et al., *Various Versions of K-means Clustering Algorithm for Segmentation of Microarray Image.* 2020.
115. Nanjundan, S., et al., *Identifying the number of clusters for K-Means: A hypersphere density based approach.* ArXiv, 2019. **abs/1912.00643**.
116. Manikandan, S., *Measures of central tendency: Median and mode.* Journal of pharmacology & pharmacotherapeutics, 2011. **2**(3): p. 214-215.
117. Wilson, C.H., et al., *Experimental Design and Analysis of Microarray Data.* Applied Mycology and Biotechnology, 2006. **6**: p. 1-36.
118. Azimi, S., et al., *Advanced Steel Microstructural Classification by Deep Learning Methods.* Scientific Reports, 2018. **8**.
119. Fedorenko, Y., V. Chernenkiy, and Y. Gapanyuk, *The Neural Network for Online Learning Task Without Manual Feature Extraction*. 2019. p. 67-76.
120. Wicht, B., *Deep Learning feature Extraction for Image Processing*. 2018.
121. Mapayi, T., S. Viriri, and J.-R. Tapamo, *Retinal Vessel Segmentation Based on Difference Image and K-Means Clustering*. 2014.
122. Ceccarelli, M. and G. Antoniol, *A deformable grid-matching approach for microarray images.* IEEE Trans Image Process, 2006. **15**(10): p. 3178-88.