École Polytechnique Fédérale de Lausanne

SGM
ME-314: Projet d'Ingénierie Simultanée

Embraille
BERNINA Design Challenge

Aunosua Dey
Juliet Kern
Maria Pociello

EPFL

Spring 2022

Contents

1 Introduction

1.1 Context

A study published by the *Community Eye Health Journal* stated there are approximately 253 million people with visual impairment worldwide, 36 million are blind and 217 million have moderate to severe visual impairment, who purchase and wear clothing everyday [1]. Often people with a visual disability are unable to visually identify clothes that they want to buy or wear, and lack information that sighted people have. We decided to focus our design problem on a more accessible solution for the visually impaired relating to sorting through or purchasing clothing.

Solutions vary from person to person, but visually impaired people have created unique systems that work for them. For example, feeling for the texture of the fabric or the shape of the garment. However, a theme discovered during our research is that identifying "sizing, color, pattern comes last with a pair of sighted human eyes" [2]. So, for many visually impaired people, it is necessary to bring a sighted shopping companion, or to have a team at a shop to help them out. It is a similar story for picking out clothes from their closet. Often, a visually impaired person will have a good organizational system and memorization to help them locate and identify pieces of their wardrobe. Moreover, additional information may be printed on a label on the clothing that is completely inaccessible for the visually impaired.

From this, we identified two main areas of improvement. First, blind people have trouble identifying size, color, pattern or additional information printed on a label. Second, they often depend on a sighted person to help with their shopping. In this problem space, we saw an opportunity for clothing manufacturers to make clothing accessible to the blind, and for households with a blind family member to make items more easily identifiable. Therefore, this project focuses on leveraging the BERNINA software system to create solutions for the daily challenges faced by the visually impaired, through the implementation of a braille module that can be easily integrated into the machines workflow. This report also focuses on validating the feasibility of these braille stitches in reference to their readability and relative cost.

In the design, we want to target and improve 4 main areas: accessibility, ease of use, independence of the visually impaired, and comfort. Solutions currently exist to aid the blind with shopping or selecting clothing, but they fall short in one or more of these categories. Additionally, *Embraille* will be accessible for the user since the cost and speed of the production would be optimal, and for BERNINA, as it will be a simple software upgrade that will come with almost no cost of implementation. It will be easy to use and comfortable (component most of the already existing solutions lack). All of this will allow independence in the organizing and shopping skills of visually impaired people. Finally, the solution will be comfortable, since it is fabric, it blends into clothing, and is washable.

1.2 State-of-the-art

- Braille clothing tags [3]: Metal tags with braille inscription on them. It is attached with clothing pins. But they are uncomfortable and not easy to use (Figure 1). Companies: 'The Braille Store', 'Yeuell'.

- Penfriend audio tag [3]: A sticker put in the clothes, and with a penfriend that reads the audio label. But they are not easy to use, durable or comfortable (Figure

2). Companies: ‘RNIB’, ‘Maxiaids’.

– Camera-based colour and pattern recognition [3]: Mobile app that identifies the colours and patterns. But it is not in production.

– Tactile solutions [3]: Sew buttons to provide tactile feedback.Companies: ‘Carroll’, ‘Maxiaids’.

Figure 1: Braille tags.

Figure 2: Penfriend audio label.

1.3 Design plan

Our design, *Embraille* is a new module on the BERNINA machine that will allow the users to write braille on any fabric or material. Braille can be written with the module directly onto clothing, or onto a fabric label and sewn on, based on users preference. The design will address the 4 areas of accessibility, ease of use, independence of the visually impaired, and comfort. We will identify the ideal method of stitching braille, through an iterative prototyping process and design testing. Then we will evaluate the cost effectiveness of the solution to ensure it is accessible.

1.4 Requirements and constraints

The requirement of the project can be defined as: Users must be able to read the braille. Through analysis of documentation assigning suggested braille specifications, it can be assumed that the closer the braille dot is to a hemisphere the increased readability, since in order to be readable, braille dots should have a domed or rounded shape – not pointy or flat [4]. They must be rounded in order to be ADA compliant as well [5].

In addition, the greater the height of the dot, the greater the readability. Therefore, the perceived congruence with a three dimensional curve and the height of the dot are included in the design requirements. To further validate the readability requirement of the stitch design, usability testing to gather the percent accuracy of each stitch is conducted on each design. The readability can be tested by evaluating various measurements of the stitches using a computational decision matrix, which is discussed in the Methods section. A summary of this requirement and its respective testing protocols are outlined in Table 1.

Description	Measured parameter	Testing/evaluation protocol	Metric
1.0 Users must be able to read the braille.	Congruence with a three dimensional curve and height of braille dot	1.0.0 Profilometer data of the stitched dots and matlab data - variety of measurements to be defined and weighed in methods section	Ranking (/100)
	Accuracy of reading (%)	1.0.1 Record the number of accurate letters read - number of incorrect letters read / total number of letters x 100%	Percent accuracy (%)

Table 1: Requirements and testing/evaluation protocol.

Our project must also consider several constraints. These are due to constraints from the BERNINA sewing machine, or constraints from the standard Braille sizing. Although there is no global-ISO type norm for braille writing, there are several recommendations from the National Library Service for the Blind and Physically Handicapped [6]. Dots should be a minimum of 1.5mm in diameter, and the minimum space should be 2.3mm between dots, with a minimum height of 0.48mm. As the braille is being embroidered on fabric rather than paper, it is assumed that we should maximize the size of the dots, within reason, therefore those values will be the minimum constraints. The constraints from the BERNINA machine, which limits the space to 9mm x 9mm, will serve as the maximum constraints. Taking the ratio of 5.3:9, this leads to maximum constraints of 2.55mm for the diameter of the dot and 3.9mm for the space of the dot. In addition, the presser foot of the BERNINA sewing machine should be presser foot 20C. Due to the geometry of this foot, tactile braille dots can be sewn onto the fabric without being compressed by the metal parts of the foot. This leads to a maximum constraint of 5mm for the height of the dot.

Our solution also aims to be cost effective. Thus, based on existing solutions, we decided that stitches should take no more than 10s to complete, based on a standard speed of 720 stitches/minute. In addition, based on the cost of thread as 0.0343CHF/m, the cost of one stitch should be no more than 0.05CHF. These constraints are summarized in Table 2.

Constraints	
Factor	**Values**
Diameter of dot	1.5 mm to 2.55mm
Space between dots	2 mm to 3.9mm
Height of dot	0.48mm to 5mm
BERNINA Presser Foot	20C
Time to create one stitch	Less than 10s
Cost of one stitch	Less than 0.05CHF

Table 2: Constraints for the dimensions of the braille dot and presser foot used.

2 Methods and Results

The design process can be divided into five main sections: determining initial design direction, stitch design process and creating prototypes, analyzing the stitches with profilometer data and MATLAB, user testing and measuring cost-effectiveness. The first part of the design process is dedicated to determining the design direction and approach. This includes preliminary testing and research to resolve our standardized thread to use for the stitches and method of stitching. The stitch design process consists of using braille specifications and the BERNINA software to design and realize the stitches onto the fabric. The analysis of the stitches is done through acquiring the three dimensional data using a profilometer, and analyzing the collected data using MATLAB, testing protocol 1.0.0 (Table 1). Next, testing is achieved to validate our design and inform decisions for future recommendations, testing protocol 1.0.1 (Table 1). Finally, cost effectiveness of the product will be computed.

2.1 Stitch Design

The BERNINA machine provides two main ways of sewing: stitching and embroidery. Since the embroidery module takes a certain allotted time and materials to set up, we decided to create a braille alphabet that can be uploaded to the machine in order to provide additional ease of use and reduce set-up or sewing time. To begin the stitch design process, a standardized list of materials and dimensions were selected to provide consistency between analysis of each dot design. Six designs were chosen to prototype for further analysis, and to determine the ideal stitch design.

2.1.1 Materials and Instrumentation

Materials used are as listed below. The thread colour must provide enough contrast between the fabric for the profilometer laser displacement sensor to detect the surface topography. Hence we primarily use a bright red colour for our samples against a white fabric. Gabardine fabric was chosen out of the available fabrics as it was a tougher material to provide stability and resistance against tear during prototype creation. The samples are adhered to cardboard to provide a flat surface and allow for the profilometer to accurately detect the stitch topography with minimal distortion. Instruments and Software used include the BERNINA Embroidery Software v.8.0, the BERNINA 590 Machine, a profilometer, MATLAB v.R2022a and Minitab.

List of materials

- No.100 SERALON® polyester thread col.0501;
- White Gabardine fabric;
- Double sided adhesive tape;
- CANSON® Gray Cardboard ;
- Stabilizer paper.

List of instruments and software

- BERNINA software v.8.0;
- BERNINA 590 machine;
- BERNINA 20C presser foot;
- MATLAB v.R2022a;
- Profilometer;
- Minitab.

2.1.2 Stitch Dimensions

The limitations of the BERNINA machine restrict the stitches to have to fit within a 9.0mm x 9.0mm space. Therefore, the diameter of each dot is resolved to be 1.8mm in length to comfortable fit the character to scale within the stitch constraint, and the space between the midpoints of each dot is resolved to be 3.3mm, as defined below in Figure 3, the scaled braille diagram. The letter "S" is used to demonstrate both diagonal and side-by-side spacing between braille dots. Each stitch sample is made within the same metrics, to allow for consistent comparison during the braille analysis process.

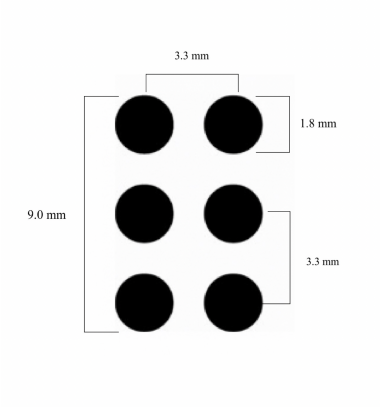


Figure 3: Scaled braille to a 9.0mm x 9.0mm space based on BERNINA machine constraints recommended specifications [6].

2.1.3 Stitch Designs

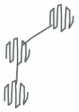
#	Stitch Design	Dot Design	Stitch Implementation	Image
1.0				
2.0				
3.0				
4.0				
5.0				
6.0				

Table 3: Stitch Designs.

2.2 Prototype Creation

2.2.1 Prototype Creation Steps

The BERNINA embroidery software is used to realize the stitches onto the fabric. Keeping with the specified dimensions the following steps are used to implement the stitch design and stitch them onto the fabric:

1. Define a 9.0mm x 9.0mm space scale on the software canvas.
2. Upload a braille image to be used as a guide.
3. Implement the character with stitch 'Open Object' drawing tool.
4. Select auto-start and auto-end at either side of the stitch.

5. Upload the stitch to the machine and save it to the machine files. The braille character can now be used in combination mode, as seen in Figure 4.
6. Adjust tension from 2.5 to 6.0.
7. Prepare fabric by adhering stabilizer to opposite side.
8. Stitch the sample onto the prepared fabric.
9. Cut, label and adhere samples to cardboard.
10. Repeat steps 1-9 for each stitch in each of the 6 stitches.

Figure 4: Braille character stitches uploaded to the machine.

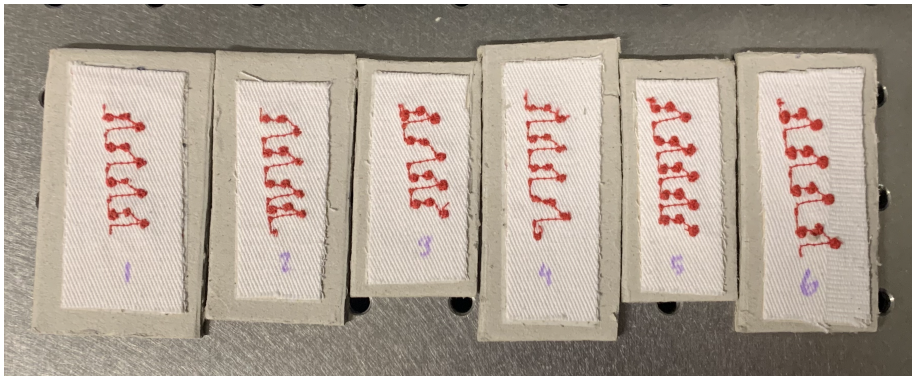


Figure 5: Samples created for analysis.

2.2.2 Prototype Challenges

When sewing several characters together on the machine, the connecting stitch between characters would overlap, creating a more raised stitch. This is problematic, as one could interpret the raised connecting stitch as another braille dot, leading to a misreading of a character. The reason for this raised stitch is unknown but is something we plan on

exploring later. In addition, the nature of the stitch design caused the bottom side to have a larger raised character than the desired side. By adjusting the tension on the machine from 2.5 to 6.0, it allowed for more tactility on the desired side of the fabric.

Another challenge that we faced was the distortion of the braille characters when sewing with the machine onto the chosen fabric. To rectify this, we attached a stabilizing fabric to the fabric as another layer. This minimized the distortion of the braille in the prototypes.

2.3 Analysis using Profilometer and MATLAB

To identify the ideal stitch design, the 6 different stitches must be evaluated. The 'best' stitch in this section is defined as the stitch with the highest 'readability' ranking, which fulfills requirement 1.0 of the design - users must be able to read the braille. To evaluate the performance of the stitches in terms of their readability, quantitative metrics will be defined. Data will be collected using a profilometer, and a numerical analysis will then be conducted using MATLAB. A variety of measurements will be collected, then a computational decision matrix will be used. By using the measurements from MATLAB and the relative weights from the decision matrix, the 'best' stitch design in terms of readability can be determined. This is testing protocol 1.0.0 outlined in table 1.

2.3.1 Data Collection: Profilometer

Profilometry is a technique used to extract topographical data from a surface - a full three dimensional scan in the case of our design [3]. The purpose of profilometry is to get surface morphology, step heights and surface roughness. Step heights were of interest for this project, as we could then use the matrix of heights to calculate our parameters of interest, to be defined later in this section. The method for the profilometer is as follows:

1. Set up the profilometer.
2. Place the sample on the base of the profilometer, and use the position arrows to change the position of the sample.
3. Using the 'Focus Guide,' adjust the two dotted white lines to the dot of interest.
4. In 'Expert Mode,' adjust the brightness so the areas of interest are not red or yellow.
5. In the 'Process Image' tab, set the reference plane as the fabric.
6. In the 'File' tab, export the 'height data' as a CSV.
7. Repeat steps 2-6 for each dot in each of the 6 stitches.

These steps result in a CSV file for each dot analyzed in the profilometer. The file contains a matrix of the heights of each dot. The next step is to use MATLAB to process the data. The character S in each sample is made up of 3 dots, so an average was taken of the 3 dots, seen in Figure 6.

An important remark about the resulting matrix of heights is the fact that, as you select the area you want to study, plus setting up a precision of the results, it gives a matrix where value of row 1 column 1 is the first value, and the following ones corresponds in the (x,y) directions with respect to the row and column. This can be seen in the plot

below. Moreover, for some technical reasons, although having the same diameter for all of the dots, some dots got more data than others, that is why sample 3 in the index of column is much larger than sample 1, because it got much more data, even having the same diameter.

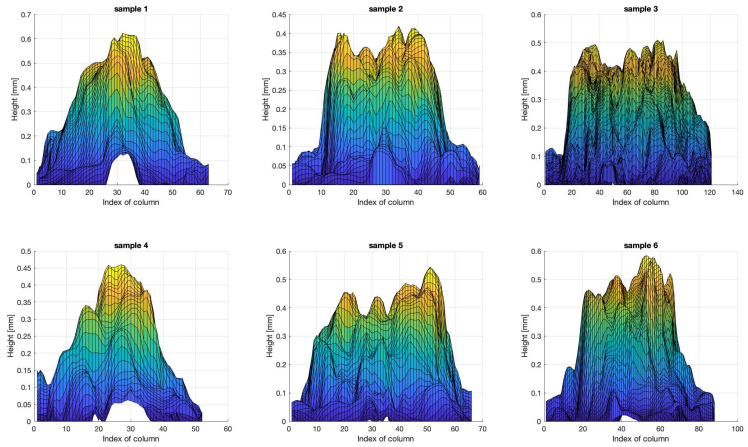


Figure 6: Original dots of each sample in x-z plane.

The standard deviation of each sample is represented in the Figure 7.

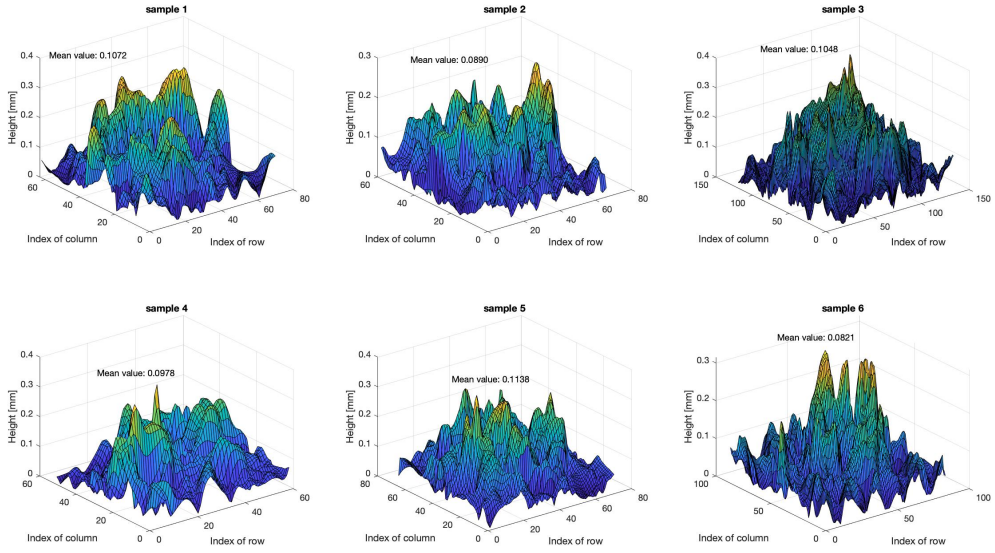


Figure 7: Std of each sample with its mean value.

Then, to check how smooth the actual data is,the polynomial function of degree 4 of best fit is computed. This approximation is done with a robust bi-squared fitting, and the visualization can be seen in Figure 8. It follows the following general form:

$$p(x,y) = p_{00} + p_{10}x + p_{01}y + p_{20}x^2 + p_{11}xy + p_{02}y^2 + p_{30}x^3 + p_{21}x^2y + p_{12}xy^2 + p_{03}y^3 + p_{40}x^4 + p_{31}x^3y + p_{22}x^2y^2 + p_{13}xy^3 + p_{04}y^4$$

A robust bi-squared fitting was applied since adding these features will result into a fit less sensible to outliers and that follows better the trend (in this way the resulting errors computed below were smaller). Moreover, we went up until polynomial of degree 4 because up to this degree, errors were too significant, and at degree 4 it started to diminish much more.

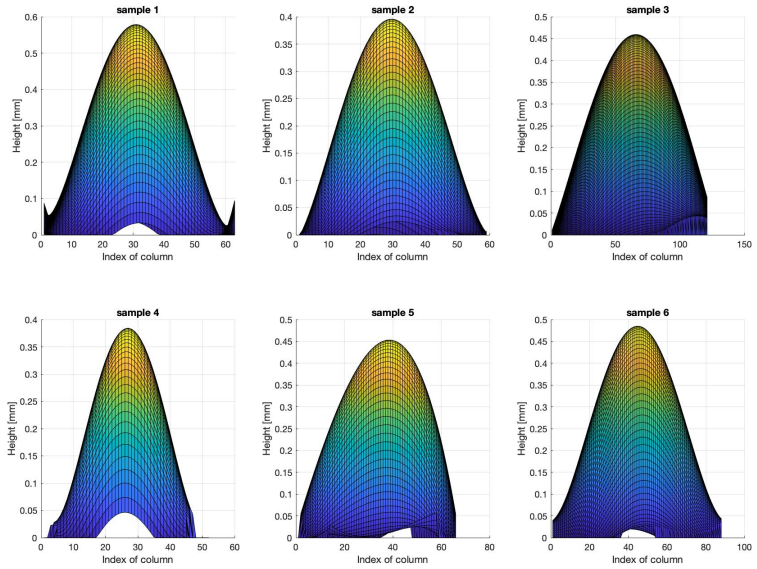


Figure 8: Polynomial approximation of each sample.

For evaluating the geometrical characteristics of the different samples, the following parameters are computed:

- Abs.Error: Difference of height (z component) between each (x,y) of the polynomial fit and the actual data.
- MSE: Mean-squared error.
- R: squared root of R^2. Proportion of the variation in the dependent variable that is predictable from the independent variables.
- Rel.Vol.Error: Relative error between exact volume and approximated (polynomial) volume.
- $\bar{H}$: As the data plots a lot of points around the dot, meaning it exists a lot of points with zero height. Instead of computing the mean height, the 'mean' height will be computed as the volume divided by the total area.

	Abs.Error	MSE	R	Rel.Vol.Error	Max. height	$\bar{H}$
Sample 1	0.0428	0.0033	0.9516	7.3025	0.6223	0.3117
Sample 2	0.0378	0.0026	0.9174	2.3534	0.4193	0.2109
Sample 3	0.0468	0.0040	0.9084	9.0904	0.5097	0.2871
Sample 4	0.0365	0.0023	0.9232	1.2176	0.4597	0.2204
Sample 5	0.0432	0.0031	0.9320	9.0713	0.5433	0.2556
Sample 6	0.0359	0.0027	0.9468	17.28	0.5840	0.2603

Table 4: Summary of significant variables.

2.3.2 Computational Decision Matrix

The parameters described above are the metrics of interest for evaluating the different types of stitches. Abs. Error, MSE, R and Rel. Vol Error evaluate the congruence of the braille dot with a three dimensional curve. Mean height and maximum height evaluate the tactility of the stitches. Each parameter will be given a different weighting, justified below.

In order for braille to be tactile and readable, the surface of the dot is ideally uniform and smooth. Data of each dot will be interpolated to calculate the polynomial of best fit. This is representative of the smooth surface that is closest to the stitch. In order to be readable, braille dots should have a domed or rounded shape – not pointy or flat [2]. They must be rounded in order to be ADA compliant as well [3]. Thus, it is important for the Abs. Error, MSE, R and Rel. Vol Error to be minimized, as this will indicate how close to a smooth surface the stitch is. Each metric will have a relative weighting of 0.1.

The 'mean' height and maximum height will help evaluate the tactility of the stitches. The standards of braille embossed on paper is 0.48mm in height [4]. However, fabric is less uniform than paper, and the braille dots must stand out from the base fabric in order to be readable. Therefore, it is assumed that the greater the height of the dot, the more tactile and therefore readable the stitch will be. Maximum height of the dot will have a relative weighting of 0.1 as it only takes into consideration the height at one x-y coordinate of the stitch. 'Mean' height will have a weighting of 0.5 as it is a better representation of the braille dot. These parameters should be maximized.

In the computational decision matrix, each stitch will be ranked in terms of each parameter. For parameters that should be minimized, the smallest value will be ranked a 6 and the greatest a 1, 6 being the most ideal. Similarly, for parameters that should be maximized, the greatest value will be ranked a 6 and the smallest a 1. The rank is then multiplied by the weight of the metric, to get the weighted rank, as seen in Figure 9. The maximum weighted rank is 6, so that is used to calculate the weighted score out of 100.

Metrics		Weight	Stitch 3			Stitch 5		
			Value	**Rank**	**W Rank**	**Value**	**Rank**	**W Rank**
Abs Error	Min	0.1	0.0468	1	0.1	0.0432	2	0.2
MSE	Min	0.1	0.004	1	0.1	0.0031	3	0.3
R	Max	0.1	0.9084	1	0.1	0.932	4	0.4
Rel. Vol Error	Min	0.1	9.0904	2	0.2	9.0713	3	0.3
Max Height	Max	0.1	0.5097	3	0.3	0.5433	4	0.4
Mean Height	Max	0.5	0.2871	5	2.5	0.2556	3	1.5
TOTAL		1			3.3			3.1
(/100)					55			51.66666667

Table 5: Computational Decision Matrix (Full Matrix in Appendix).

Stitch	Weighted Result (/100)
1	35
2	26.67
3	55
4	36.67
5	51.67
6	36.67

Table 6: Summary of Weighted Ranks from Decision Matrix.

The weighted ranks are summarized in Figure 10. Based on the decision matrix, stitches 3 and 5 are the most readable stitches based on testing protocol 1.0.0, as they are the only ones that score above 50.

2.4 User Testing

To evaluate requirement 1.0.1, the accuracy of reading was measured through user testing. We set up a meeting with the *Fédération Suisse des Aveugles et Malvoyants* (SBV-FSA), and spoke to 3 visually impaired individuals with varying levels of blindness. User 1 has almost complete blindness, user 2 has low vision, user 3 has complete blindness. Based on our results from section 2.3 and requirement 1.0.0, samples of words were prepared with stitches 3 and 5. If the same word was created with both stitches, testers would likely be biased towards a higher reading accuracy for the second stitch, since they would already know the correct word. Therefore, we randomized the characters and words that were created with each stitch.

Stitch	Number associated	Word
3	3.1	CHAT
	3.2	TABLE
	3.3	SABLE
	3.4	OURS
5	5.0	TBORD
	5.1	PORTE
	5.2	CORDE
	5.3	SPORT

Table 7: Test prepared for the visually impaired.

Each word will be read and each letter will have an outcome of 1 (if it is readable) or 0 (if it is not). Therefore, readability will take a binary (discrete) number: 0 or 1. Minitab will be used as the analytical tool for analyzing the data. Minitab is a statistic software.

Table 8 is a summary of the resulting readability of the different stitches with respect to each user. It represents the total count of the readability of the letters. Followed by some plots that will allow us to understand the data better.

User	Stitch	Readability			
		0	1	All	% readable
1	3	11	7	18	38.8%
	5	6	14	20	70%
2	3	5	13	18	72.22%
	5	3	17	20	85%
3	3	18	0	18	0 %
	5	15	5	20	25%

Table 8: Stitch percent readability of each user.

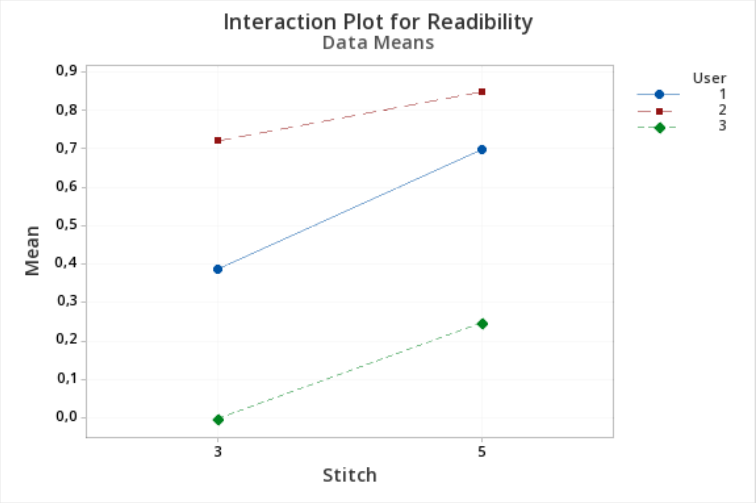


Figure 9: Interaction plot with respect to users and stratified by the type of stitch.

Figure 10: Mean of readability with respect to the different variables: Letter, Stitch, User.

The data shows us that stitch 5 is more readable than stitch 3, with an average readability of 60.0% and 37.0% respectively. There is a notable difference between the readability of the users. As all the users know how to read braille, this is likely due to their differing levels of blindness. In addition, it is important to bear in mind when analyzing the data the fact that, although each dot was analyzed during our project, we did not analyze the total composition of multiple dots related to its readability, just one dot. This is remarkable because they were issues related to the readability of the words and of the letters. As mentioned during the prototyping stage, the connecting stitch between characters would overlap, creating a more raised stitch. The users specifically said that 3 was less tactile than 5, however, as we predicted, they found the raised connecting stitch confusing. The users sometimes interpreted it as another braille dot, and were confused about which dots were of the braille character, leading to a misreading of a character. On the other hand, they confirmed that stitch 5 felt better (more tactile) than 3 but it was much more difficult to read it since the spacing between the dots was smaller. Therefore,

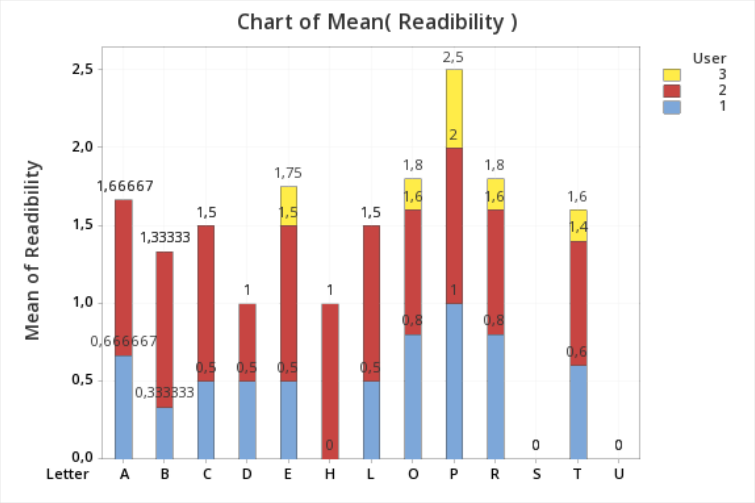


Figure 11: Mean of readability added up with respect to its letter and stratified by user.

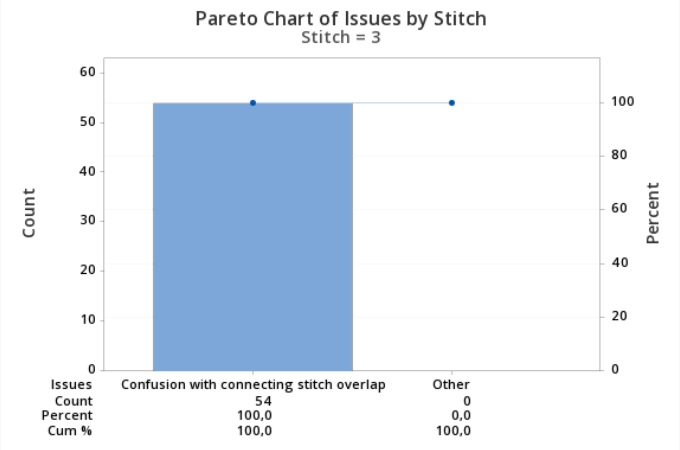


Figure 12: Pareto Chart of the Issues during the test for Stitch A.

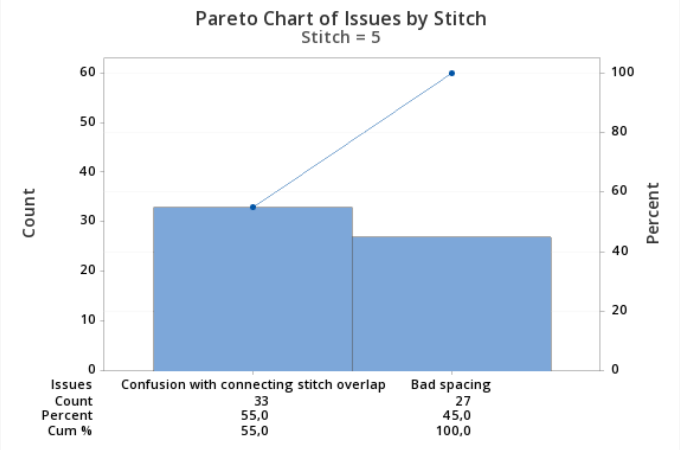


Figure 13: Pareto Chart of the Issues during the test for Stitch B.

as we are analyzing the dot itself and not the total composition of it, we can say that stitch 5 is much better. Lastly, the readability of each letter is demonstrated in Figure 11 in order to take into consideration in next steps which letters need improvement.

Stitches	**Time (s)***				**Cost**	
Trial Numbers	1	2	3	**Avg.**	**Length (m)**	**Calculated cost ($)****
1.0	8.34	8.09	8.13	**8.19**	0.383	**0.0131**
2.0	6.71	6.66	6.40	**6.59**	0.318	**0.0109**
3.0	6.46	6.51	6.20	**6.39**	0.316	**0.0108**
4.0	5.82	5.53	5.60	**5.65**	0.264	**0.0091**
5.0	7.90	7.70	7.80	**7.80**	0.366	**0.0126**
6.0	9.55	9.63	9.50	**9.56**	0.465	**0.0159**

Table 9: Time of Stitch Sample to Complete one Stitch at a Standard Speed of 720 stitches / minute and Calculated Cost of Stitch.
*Time of stitch sample to complete one stitch of a standard speed of 720 stitches/minute
**Length of thread used (m) x cost of thread (0.0343 CHF/m)

2.5 Measuring time-effectiveness and cost-effectiveness

Finally, time-effectiveness and cost-effectiveness were measured to ensure that our solution is accessible and low-cost. The method for collecting this data was as follows:

1. Open the stitch file on the BERNINA software.
2. Using the 'stitch playback' option, set the playback speed to the standard speed of 720 stitches / minute.
3. Measure how long it takes to start and end the stitch.
4. Repeat step 1-3 three times, for a total of three trials.
5. Calculate the average time taken.
6. Using the BERNINA software, measure the total length of the stitch.
7. Based on the cost of thread, 0.0343CHF/m, calculate the cost of the stitch.
8. Repeat steps 1-7 for each stitch.

Figure 13. Cost of Stitch Sample to produce one Stitch (CHF)

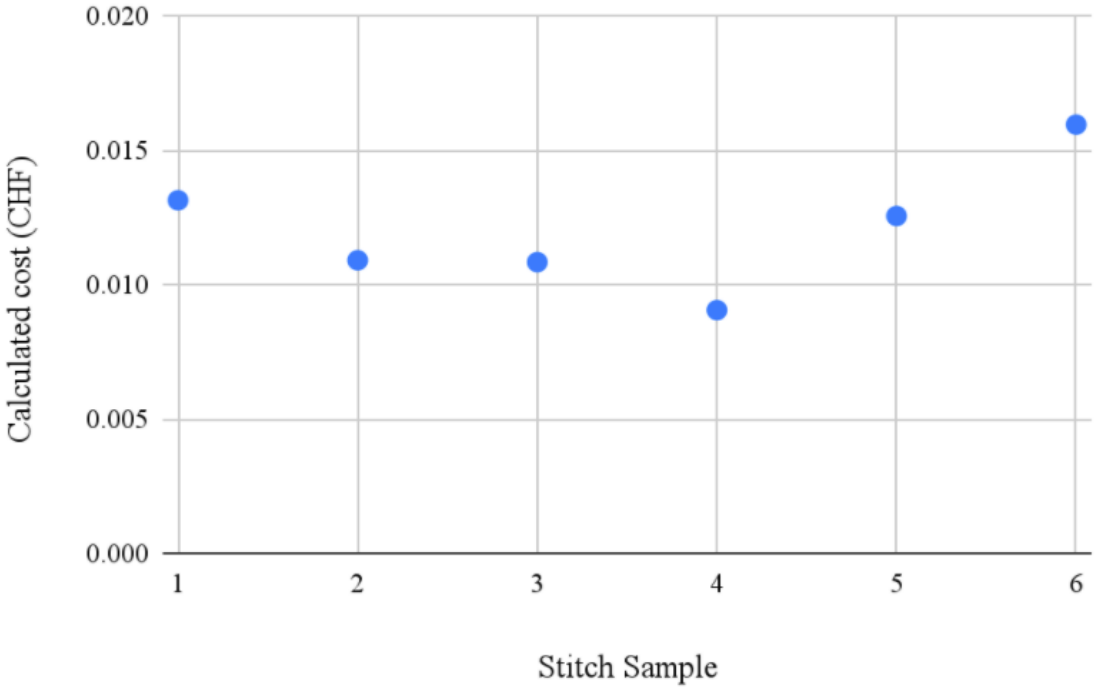


Figure 14: Time of Stitch Sample to Complete one Stitch at a Standard Speed of 720 stitches / minute.

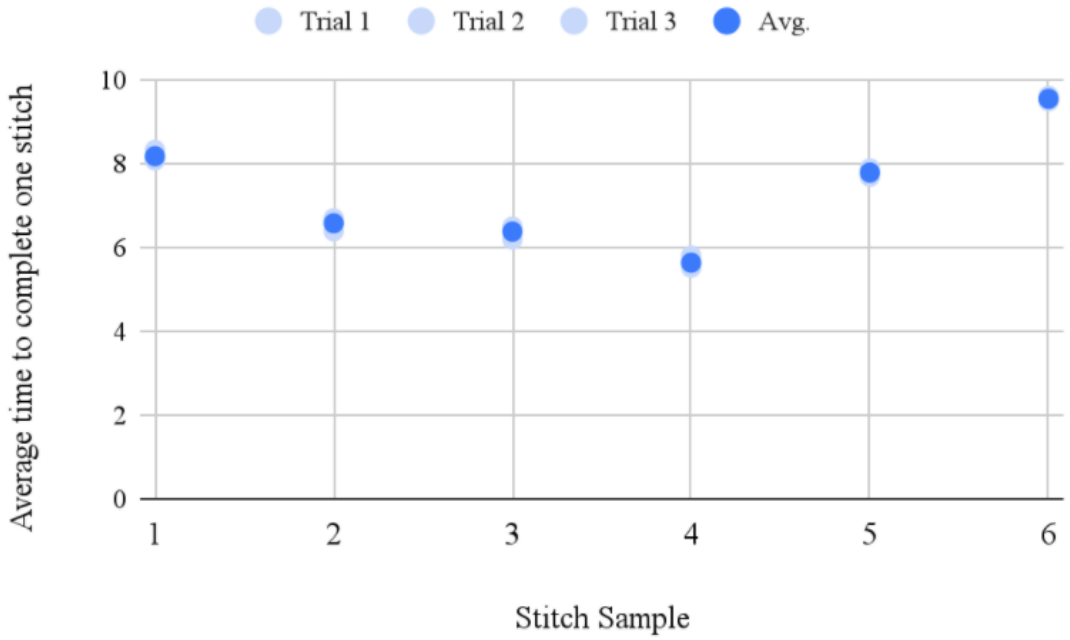


Figure 15: Cost of Stitch Sample to produce one Stitch (CHF).

From the results, Stitch 4 performed the best in terms of taking the least amount of time and the lowest cost, taking 5.65s and 0.0091 CHF. However, all 6 stitch designs fall within the constraints defined in Table 2, taking less than 10s and costing less than 0.05CHF.

3 Discussion and Conclusions/Reflections

3.1 Profilometer and MATLAB: Discussion

From the profilometer and MATLAB data, we deduced that stitches 3 and 5 were the best, most readable stitches according to testing protocol 1.0.0. The weighted results of the stitches were 55 and 51.67 respectively, in terms of their congruence with a three dimensional curve and their height (Figure 9). These were the only stitches with a score above 50.

Stitches 3 and 5 were created in a similar way, with peaks crossing over each other as seen in Table 3. Stitch 3 had 3 peaks going vertically and 4 horizontally, and Stitch 5 had 4 peaks going in each direction, which is likely why both stitches performed similarly well. As the 2 stitches that scored above a 50 were created with the crossing peaks method, it can be determined that this is the best method of creating a stitch for the braille dots, in terms of readability. However, Stitch 6 was also created using the peak design, with 5 peaks crossing over in both directions. It did not perform as well in the data analysis, with a weighted rank of only 36.67 (Figure 9), less than 50. This is of interest as we originally thought that a greater number of peaks would be more readable, since the maximum height would be greater. Something we could explore in the future is the ideal number of peaks to create a stitch. Currently, we hypothesize that it is due to the fact that since it is a larger stitch, the machine has less control as it goes over the previous stitches.

Although Stitch 3 performed better than Stitch 5, we noticed that Stitch 3 performed poorly in terms of the congruence of the braille dot with a three dimensional curve. The rank of the stitch in terms of Absolute Error, MSE, R value and the Relative Volume Error was relatively low. Stitch 5, however, performed consistently well in terms of its congruence with a three dimensional curve as well as its height (Figure 8). Though Stitch 5 had a lower weighted rank, perhaps this consistency is why it performed better with the user testers. In the future, even more metrics could be calculated to determine which are the deciding metrics and factors for readability.

In terms of limitations, the method of sewing braille dots limits the weighted score of the stitches. Through sewing, the braille dots can never be a perfect hemisphere, since stitches are crossing over at different areas, creating bumps and inconsistencies. Due to this, the congruence with a 3-dimensional curve can never be perfect.

Another limitation in this process was that each stitch had a different reference plane when collecting data with the profilometer. With the profilometer, the reference plane for each stitch was chosen as the fabric closest to the dot. Due to each stitch having a slightly different geometry, although they were the same diameter, this resulted in different reference planes and therefore different sizes of matrices for analyzing the data. This could result in some inconsistencies when calculating the metrics of interest.

3.2 User Testing: Discussion

Through analysis for the data collected during user testing, the braille stitch 5 outperformed stitch 3 in terms of readability. As stated in figure 6, stitch 5 had a mean readability value of 70.0%, 85.0% and 25.0%, whereas stitch 3 had a mean readability value of 38.8%, 72.22%, and 0% for users 1, 2 and 3 respectively. The sources of error for stitch 3 resulted from overlap that occurred between the joining stitches of the braille characters that was not visually discernible, however had a notable tactile influence. All of the issues identified with stitch 3 were caused by the overlap of the joining stitches (Figure 14). Whereas for stitch 5, 55.0% of the issues identified were caused by the overlap issue, and the remaining 45.0% was caused by spacing errors (Figure 15). Although stitch 5 out performed stitch 3 in terms of readability, it should be noted that the stitch 3 samples had better spacing between the braille characters, which in return influenced the readability.

As seen in figure 12, the majority of characters had a mean readability value of 1.0 - 1.8, except for letters S and U which had a mean readability value of 0.0, and P which had a mean readability value of 2.5.

The three users that tested the braille prototypes had varying levels of vision. As stated previously, user 1 had almost complete blindness but could detect changes in lighting, user 2 had low vision in her peripheral, and user 3 had completely blindness since birth. Visual context of the stitches provided additional insight into the testing that should be taken into consideration when evaluating stitch design and quality. This can be reflected in the data we collected, as shown in figure 11 the mean readability was inversely related to the level of blindness. Both users 1 and 2 were above the standard mean of readability at a value of 0.55 and 0.79 respectively, whereas user 3 was below at a value of 0.13. As user 3 has been completely blind since birth, she had heavy reliance on braille accuracy, and therefore the fabric distortion greatly influenced her ability to read the braille. Whereas for Users 1 and 2 they had some previous visual context to understand where the stitches braille design may be negatively influencing the intended braille design, and therefore inadvertently helped them during the testing process. Through analysis of the collected data, it can be determined that reading stitched braille is possible, however the level of blindness and visual context has influence on the ability to read the braille.

3.3 Time and Cost-effectiveness: Discussion

Through data collection of the time and cost needed to create one stitch, we can conclude that all 6 stitch designs are indeed time-efficient and cost-effective, based on the constraints defined in Table 2. This validates that our solution is accessible in terms of cost. By gathering information on the time and cost of the stitches, it provides insight into roughly how long it would take to stitch onto the clothing compared to the existing solutions. Product competitors such as braille tags can vary in price ranging from 2.50CHF/tag to 0.57CHF/tag [7], and the penfriend has a general price of 0.08CHF/label [8]. Therefore, it was important for us to create a solution that would be in the scope of existing solutions with respect to the time and cost required to create the braille identifier. Compared to the solutions cost, it can be concluded that it would be in general less expensive to use the braille stitches at a cost of 0.01CHF/stitch for stitch 3 and 5 (table 9). With respect to the time, it is more challenging to gather exact information on competitor solutions. However, it can be assumed for both the braille tags and penfriend that the label will have to be removed before wear and re-added after wear as they are either uncomfortable (metal tag), or non-washable (adhesive sticker). Therefore, the *Em-*

braille module would only require initial braille identification rather than re-identifying the clothing after each wear. Which in return would reduce time required, and make situations such as laundry a lot easier and quicker.

3.4 How successfully does it achieve the intended goal?

Based on the requirement defined, that users must be able to read braille (1.0), we can determine that Stitch 5 is the ideal stitch. Stitches 3 and 5 scored above 50 in the geometric analysis. Although stitch 4 performed the best in terms of cost-effectiveness, stitches 3 and 5 are also within the constraints defined in Table 2 for time and cost. However, Stitch 5 outperformed Stitch 3 in the user testing, leading to our final conclusion.

Our initial idea was to create braille such that it would fulfill the 4 main areas: accessibility, ease of use, independence of the visually impaired, and comfort. We developed an idea that allows braille to be created with generic polyester thread and a piece of fabric, making it a low-cost and accessible solution, as evidenced by testing the cost-effectiveness. In terms of the ease of use, our module is easy to implement on the BERNINA machine and can be used in combination mode just like the pre-existing stitches on the machine. In regards independence, our solution achieved this since readability of the braille dots was somewhat achieved. With further improvement in the space between dots and the overlapping issue, independence of the visually impaired could be further increased. Regarding the comfort, we did not test this in the scope of our project, but due to the braille being stitched with fabric thread in a non-invasive manner, we can consider this area achieved as well. Overall, our solution achieves its goals.

3.5 Further Improvements

Our design, *Embraille*, could be further improved in many different directions. In this report, we explored the ideal stitch to create braille with the BERNINA sewing machine. However, there are many more factor to consider for readability – for example the spacing of characters and much more. In the future, our product can be improved both in terms of the singular stitch for a braille cell as well as for the readability of sewn braille in general.

First, in terms of improving the single stitch, we could explore the ideal number of peaks for creating stitches. This was explained briefly in section 3.1, where we discussed that stitches 3, 5 and 6 were all created with peaks crossing over each other (seen in Table 3), with 3, 4 and 5 cross-overs respectively. However, according to the profilometer and MATLAB analysis, the readability did not improve proportionally to the number of crossovers. In the future, the ideal number of crossovers could be explored and determined.

Our design can also be improved in regards to a larger scope than just the stitch. During user testing based on qualitative feedback from the users, there are many other factors to consider for readability – for example the spacing of characters and the spacing of the cells, which were held constant in our experiments. To further develop our project, we could change the independent variable in our experiment, and have spacing of the characters and cells be of interest rather than the type of stitch.

During user testing and prototyping, we also noted that the connecting stitches between each braille character had an overlapping stitch, which caused readability issues for the users. This was due to the fact that the bump created was interpreted as a braille

cell, leading them to read the characters as inaccurate letters. As a next step, we should explore this issue and determine why the software and machine is creating this overlap between characters. This will likely lead to a large improvement in the readability.

User feedback also gave us insight into what information the visually impaired would like to have written on their clothing. This ranges from size to color to the instructions for the washing machine. In the future, we could conduct user surveys to determine what information exactly is necessary, so that braille contractions and short forms can be implemented into the module to make sewing of the information even faster.

In terms of implementing our product onto the BERNINA machines, in this report we implemented a variety of letters created with different stitches onto the machines. Currently, the letters are accessible through the 'Saved Files' tab on the BERNINA machine. However, due to the software of the machine, the different letters cannot be labelled, and only the stitch can be seen. In the future, the files can be imported as their own alphabet, with each stitch labelled with the letter.

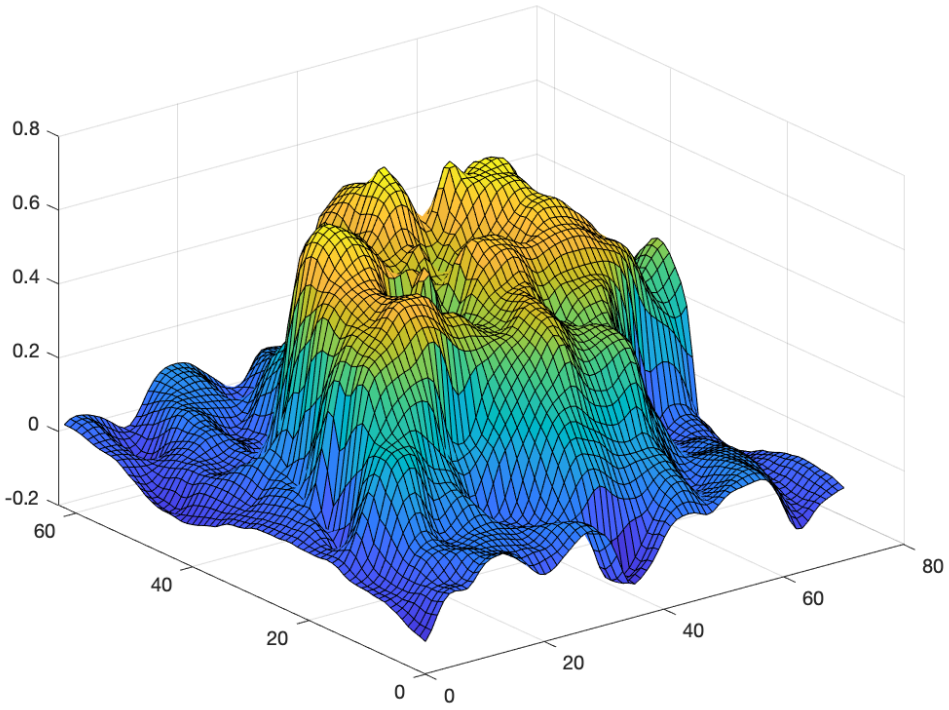
4 Bibliography

[1] Ackland, P., Resnikoff, S., Bourne, R. (2017). World blindness and visual impairment: despite many successes, the problem is growing. Community eye health, 30(100), 71–73.
[2] YouTube. (2018). How I Shop Without Sight! (come shop with me). YouTube. Retrieved May 23, 2022, from https://www.youtube.com/watch?v=8G19hywI9RM
[3]Willings, C. (2020, April 7). Teach Dressing and Clothing Management Skills to Blind and Visually Impaired. Teaching Students with Visual Impairments. Retrieved June 3, 2022, from https://www.teachingvisuallyimpaired.com/dressing--clothing-management.html#:%7E:text=Clothing%20Identification&text=Braille%20readers%20can%20use%20braille,color%20words%20and%20pattern%20words.
[4] Braille Literacy Canada. (2016). Accessible Signage Guidelines. E Formats Committee of Braille Literacy Canada. https://www.brailleliteracycanada.ca/storage/standards/AccessibleSignageGuidelines2016.pdf
[5] Spence, J. S. (2017). Creating Braille for ADA Signs. Engravers Journal. Retrieved May 23, 2022, from https://www.engraversjournal.com/articles/online/creating-braille-for-ada-signsAs. Printed in July 2017, Volume 43, No. 1 of The Engravers Journal
[6] Library of Congress. Braille Book and Pamphlets; Specification 800:2014. National Library Service for the Blind and Physically Handicapped 2014.
[7] T. B. S. Future Aids, "Braille clothing tags, colors," Braille Clothing Tags for the Blind, Colors — Braille Superstore. [Online]. Available: http://www.braillebookstore.com/Braille-Clothing-Tags,-Colors.1. [Accessed: 07-Jun-2022].
[8] From: "Penfriend 3 voice labeling system," MaxiAids.com. [Online]. Available: https://www.maxiaids.com/penfriend2-voice-labeling-system. [Accessed: 07-Jun-2022].

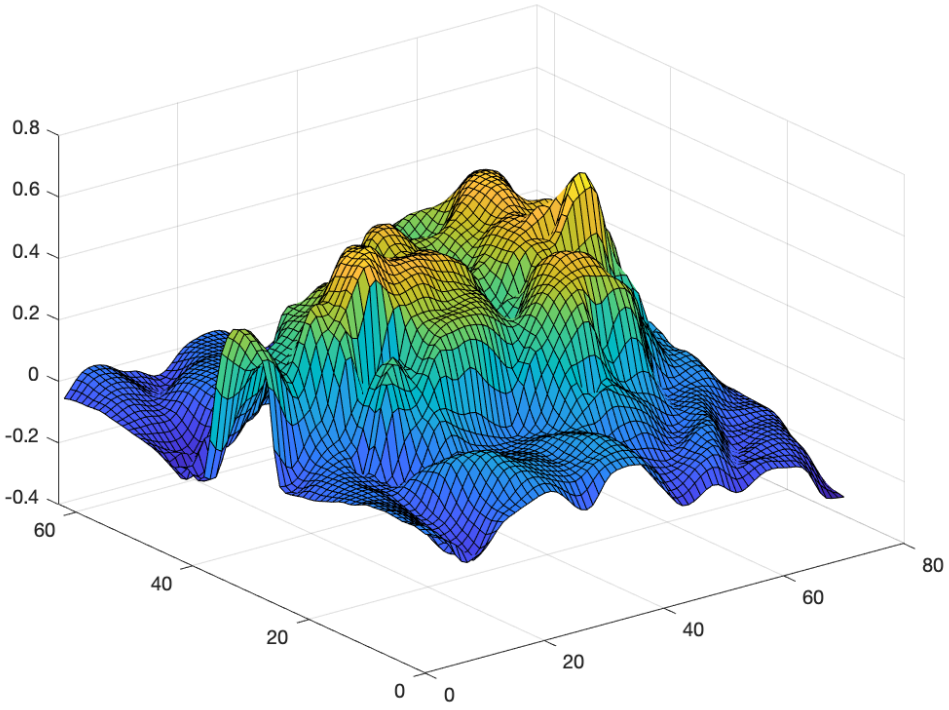
5 Appendix

MATLAB files of each sample:

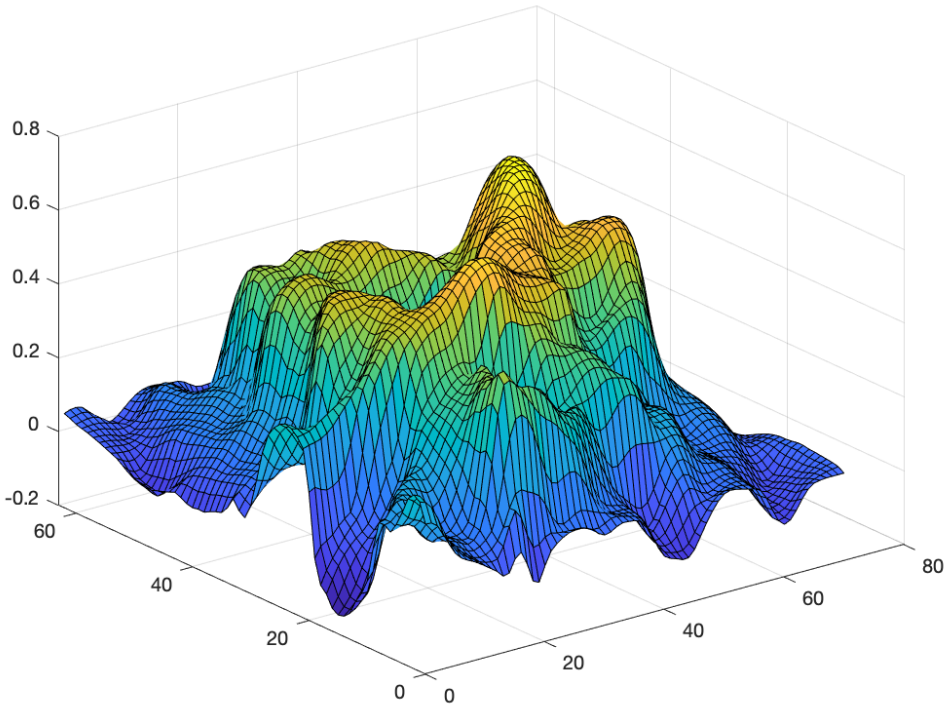
```
clear all
load('samples.mat')
x1=63;
y1=71;
SAMPLE1_1=SAMPLE1_1(4:end-1,:);
SAMPLE1_3=SAMPLE1_3(12:end-1,12:82);
S1={SAMPLE1_1,SAMPLE1_2,SAMPLE1_3};
A=zeros(63,71,3);
for k=1:3
    s = S1{k};
    s(isnan(s))=0;
    for i=1:63
        for j=1:71
            A(i,j,k)=s(i,j); %matrix that contains the 3 samples
        end
    end
end
surf(SAMPLE1_1)
```




```
surf(SAMPLE1_2)
```

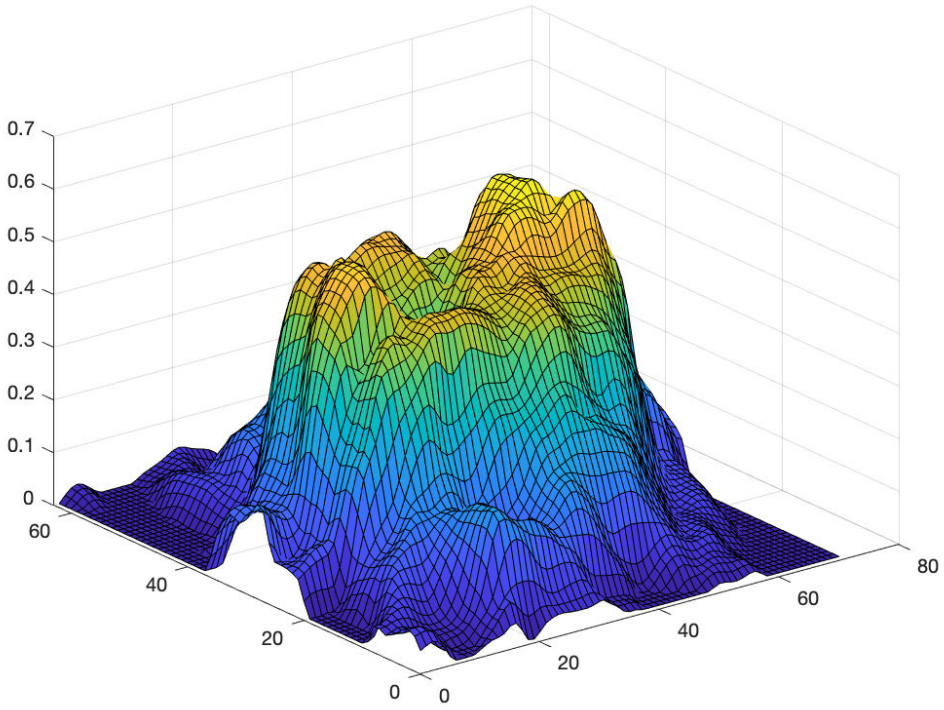



```
surf(SAMPLE1_3)
```



Mean value of the sample and variance for each point (wrt the 3 samples)

```
avg=mean(A,3); %mean value of z for each (x,y) point
avg(avg<0)=0; %deleat negative values
surf(avg)
```



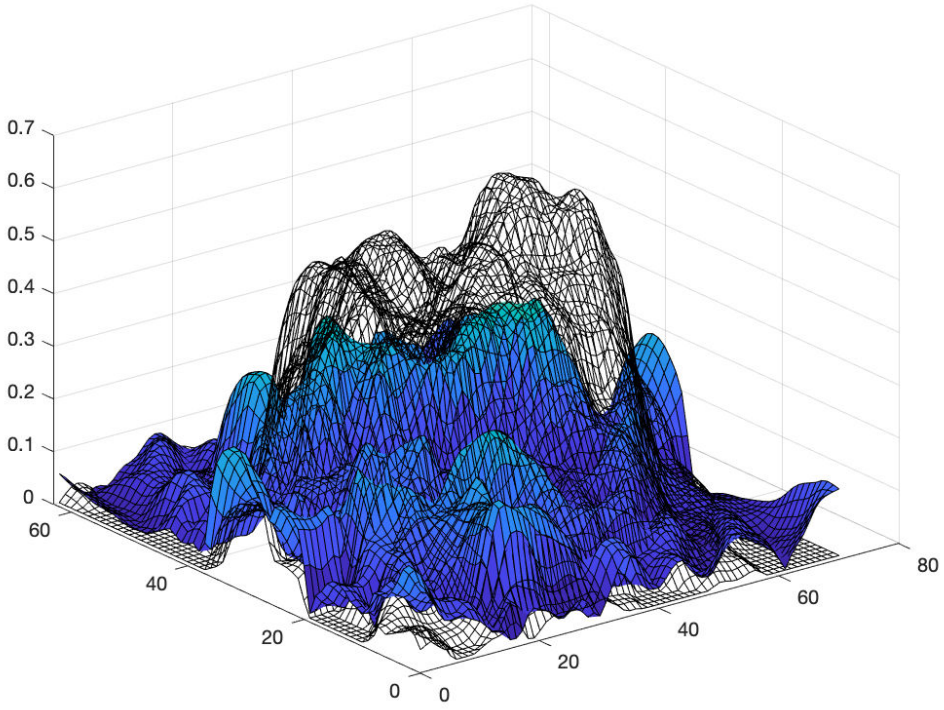

```
deviation = std(A,0,3);
mean(deviation(:))
```

```
ans = 0.1072
```

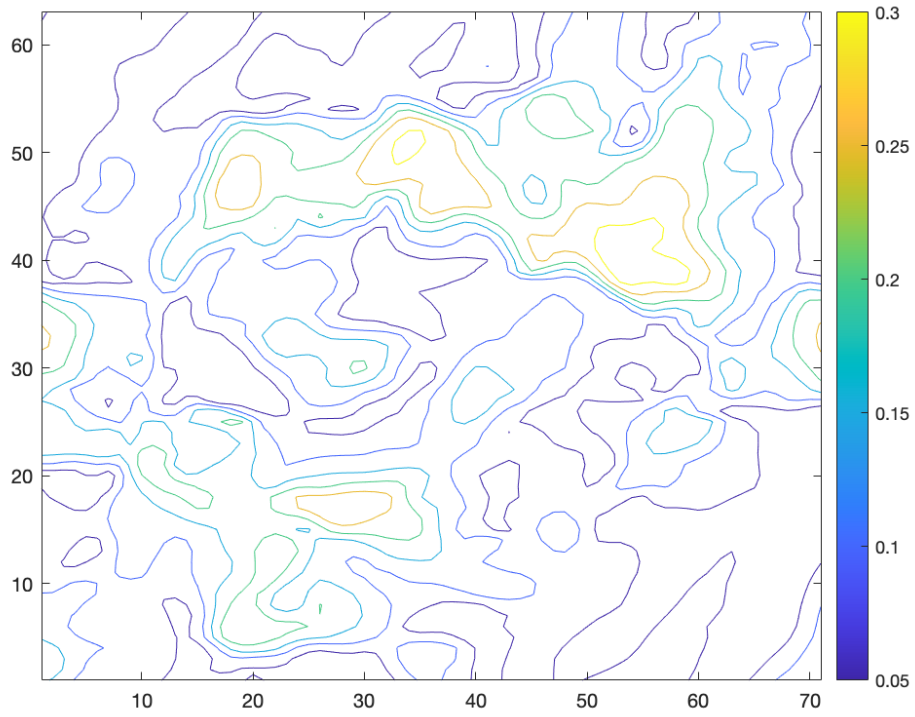
```
variance=var(A,0,3);
mean(variance(:))
```

```
ans = 0.0163
```

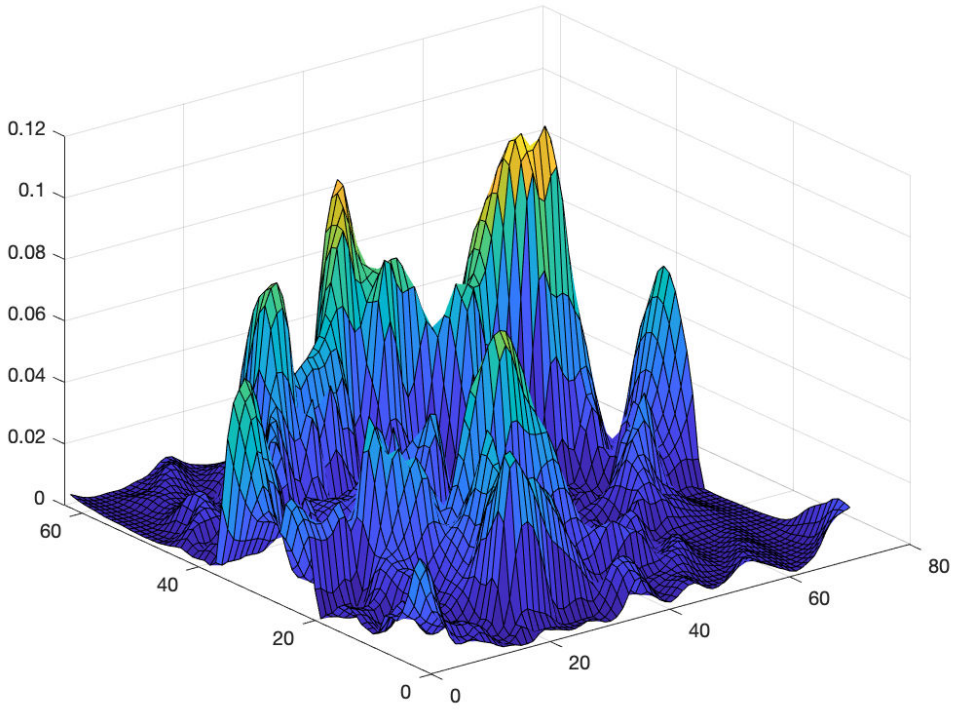
```
figure(1)
surf(avg,"FaceColor","none")
hold on
surf(deviation)
hold off
```




```
figure()
contour(deviation)
colorbar
```




```
surf(variance)
```



Approximating to a polynomial function

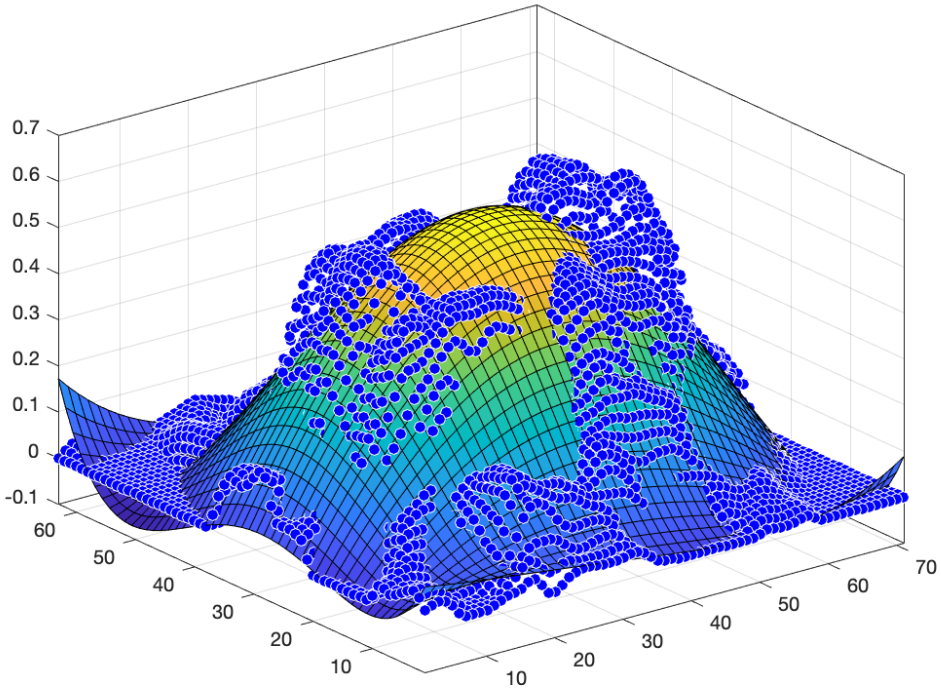
```
[X,Y] = meshgrid(1:71,1:63);
%figure(2)
%surf(X,Y,avg) --> used to ensure that the meshgrid was correctly done
%figure(3)
%contour(avg,15)
%colorbar
x2=[];
y2=[];
z2=[];
for i=1:size(X,1)
    for j=1:size(X,2)
        x2=[x2;X(i,j)];
        y2=[y2;Y(i,j)];
        z2=[z2;avg(i,j)];
    end
end

sf = fit([x2, y2],z2,'poly44','Robust','Bisquare' )
```

```
     Linear model Poly44:
     sf(x,y) = p00 + p10*x + p01*y + p20*x^2 + p11*x*y + p02*y^2 + p30*x^3 +
                   p21*x^2*y + p12*x*y^2 + p03*y^3 + p40*x^4 + p31*x^3*y
                   + p22*x^2*y^2 + p13*x*y^3 + p04*y^4
     Coefficients (with 95% confidence bounds):
       p00 =       0.124  (0.1008, 0.1472)
       p10 =   -0.005113  (-0.007584, -0.002641)
       p01 =    -0.03612  (-0.03891, -0.03332)
       p20 =   0.0001113  (6.494e-07, 0.000222)
       p11 =    0.002212  (0.002114, 0.00231)
       p02 =     0.00255  (0.002409, 0.002691)
       p30 =   -2.68e-06  (-4.783e-06, -5.776e-07)
       p21 =  -2.819e-05  (-3.001e-05, -2.638e-05)
       p12 =  -3.429e-05  (-3.634e-05, -3.224e-05)
       p03 =  -6.555e-05  (-6.857e-05, -6.254e-05)
       p40 =    3.01e-08  (1.593e-08, 4.426e-08)
       p31 =   -5.84e-08  (-7.232e-08, -4.449e-08)
       p22 =   5.536e-07  (5.382e-07, 5.689e-07)
       p13 =  -6.394e-08  (-8.162e-08, -4.627e-08)
       p04 =   5.466e-07  (5.238e-07, 5.695e-07)
```

```
options = fitoptions(sf)
```

```
options =

    Normalize: 'off'
      Exclude: []
      Weights: []
       Method: 'LinearLeastSquares'
       Robust: 'Bisquare'
        Lower: [1×0 double]
        Upper: [1×0 double]
```

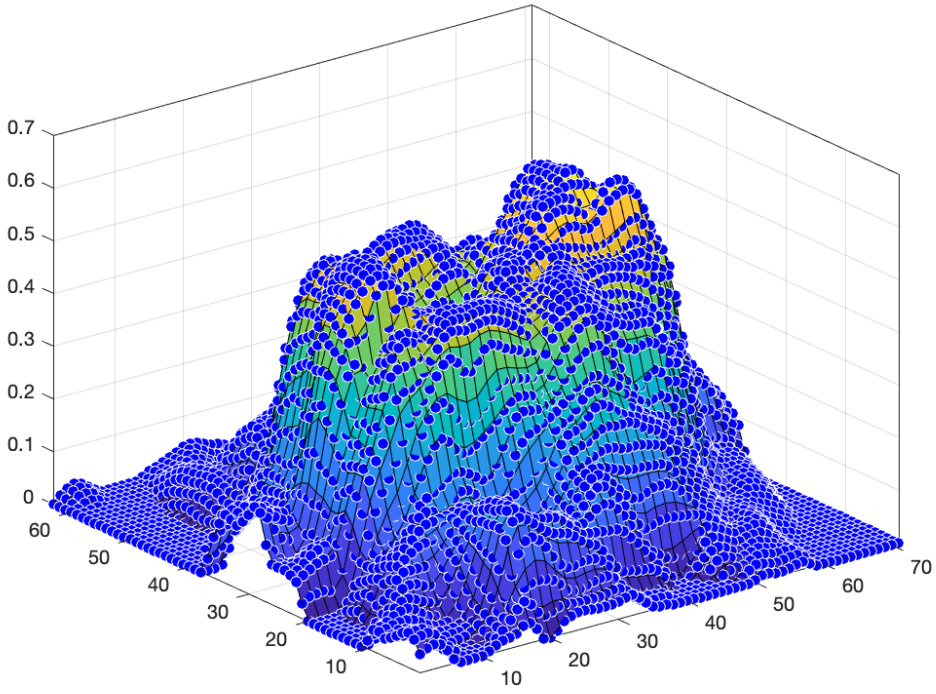
```
figure()
plot(sf,[x2,y2],z2)
```




```
sf1 = fit([x2, y2],z2,'linearinterp')
```

```
Linear interpolant:
  sf1(x,y) = piecewise linear surface computed from p
Coefficients:
  p = coefficient structure
```

```
figure()
plot(sf1,[x2,y2],z2)
```




```
%z3=zeros(63,71);
adj=zeros(63,71);
adj1=adj; %a copy
```

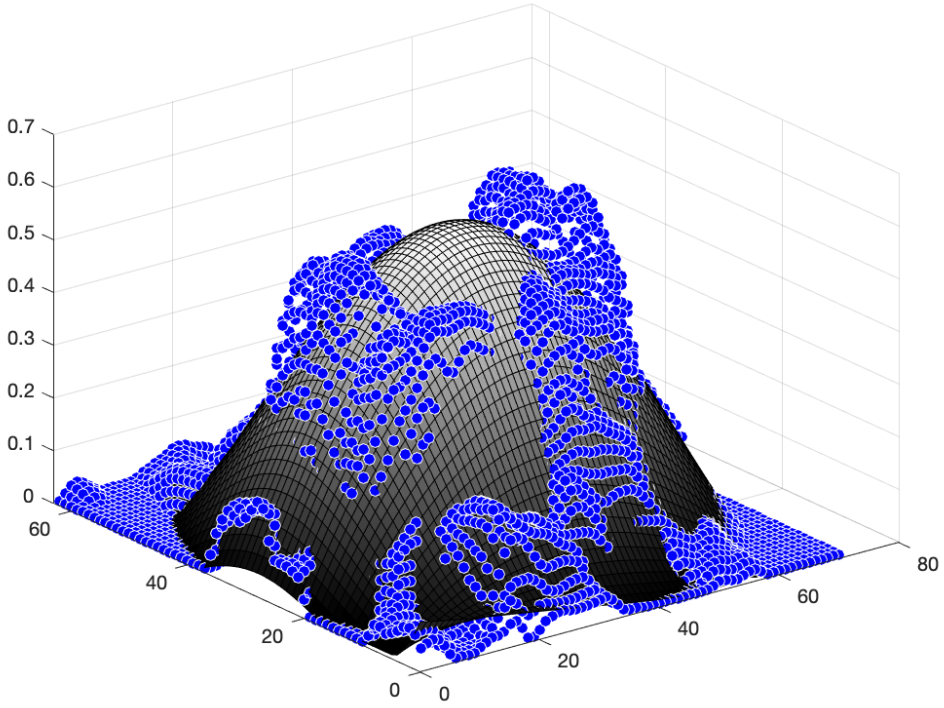
We correct the negative values and those on the extreme that go again up

```
for i=1:length(x2)
    %z3(y2(i),x2(i))= z2(i);
    if sf(x2(i),y2(i)) > 0
        adj(y2(i),x2(i))= sf(x2(i),y2(i));
    else
        adj(y2(i),x2(i))= 0;
    end
    if y2(i) > 821/15+4/15*x2(i)
        adj(y2(i),x2(i))= 0;
    %elseif y2(i) > 259/3-10/21*x2(i)
        %adj(y2(i),x2(i))= 0;
    elseif y2(i) < -39+5/7*x2(i)
        adj(y2(i),x2(i))= 0;

    end
end
%(z3==avg) it is the same
```

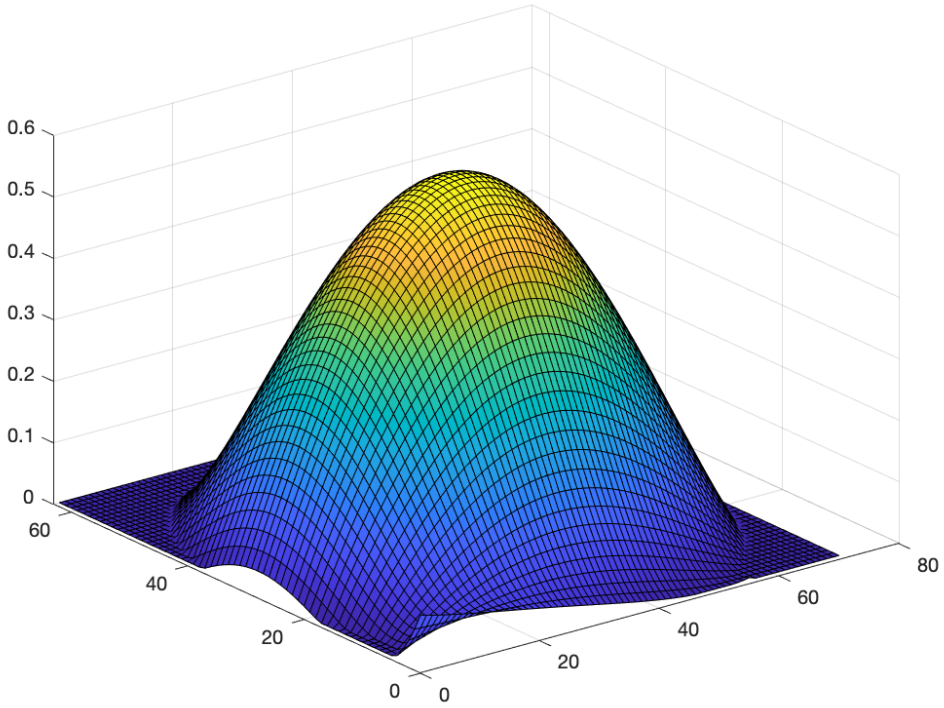
Corrected values: adj and z2

```
surf(adj)
```

```
colormap("gray")
hold on
plot3(x2,y2,z2,'ow','MarkerSize',6,'MarkerFaceColor','blue')
hold off
```




```
surf (adj)
colormap("default")
```



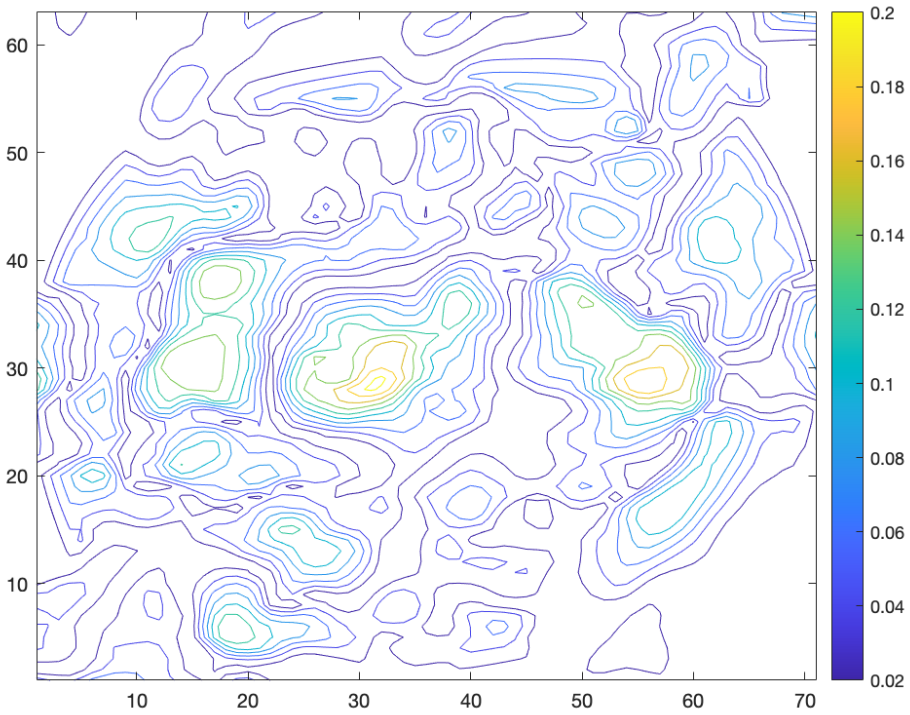
Errors

Absolute and relative errors

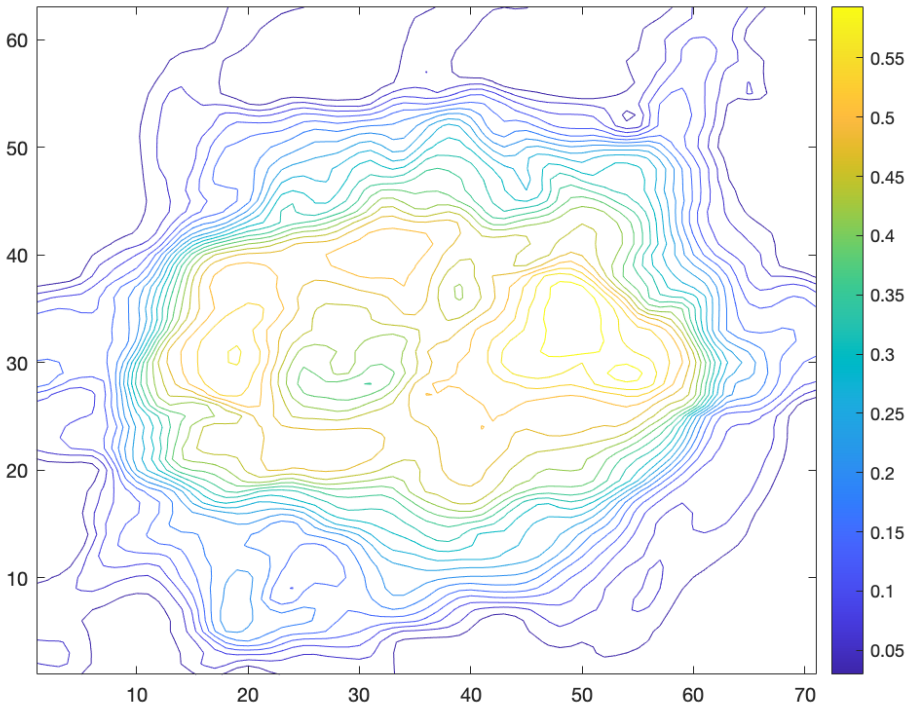
```
AbsErr=abs(adj-avg);
RelErr=AbsErr./avg*100;
mean(AbsErr(:))
```

```
ans = 0.0428
```

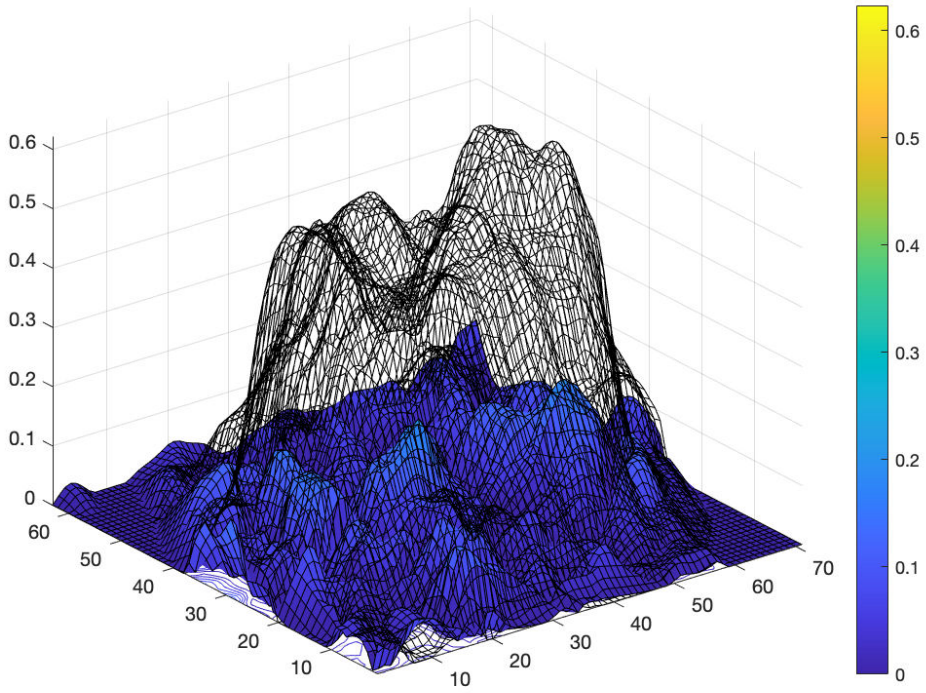
```
figure()
contour(AbsErr)
colorbar
```



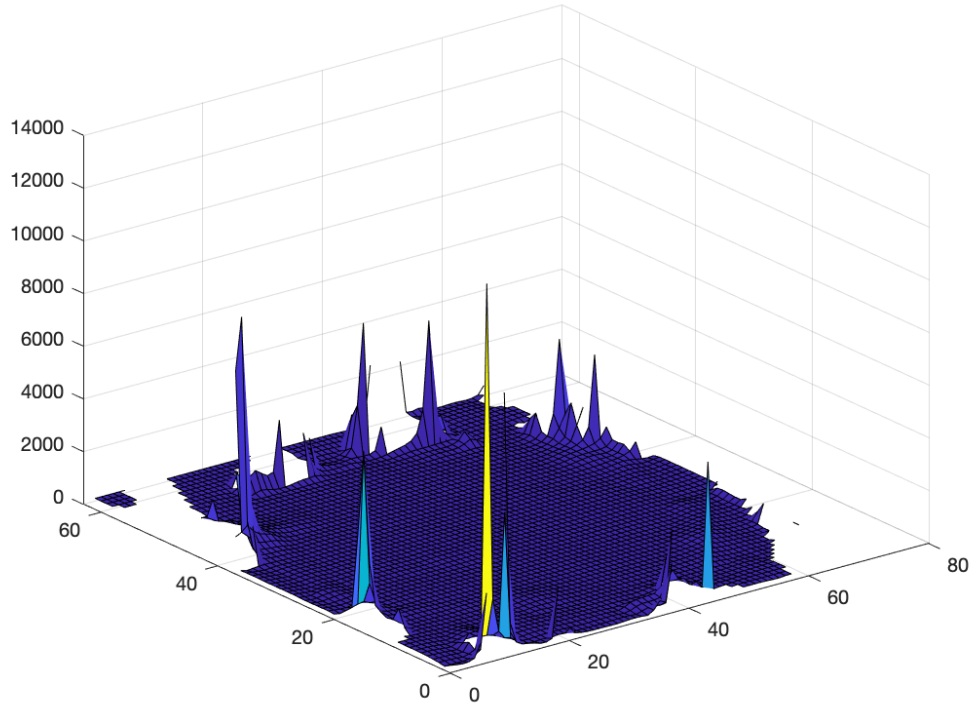

```
contour(avg,20)
colorbar
```



```
figure()
surfc(AbsErr)
hold on
surf(avg,'FaceColor',"none")
colorbar
hold off
```




```
figure()
surf(RelErr)
```




```
ans = NaN
```

Computation of RMSE

```
s=[]; %predicted data corrected
for i=1:size(X,1)
    for j=1:size(X,2)
        s=[s;adj(i,j)];
    end
end
s(s<0)=0;

z2=z2; %exact data
n=length(s);
MSE=1/n*sum((z2-s).^2) %mean squared error
```

```
MSE = 0.0033
```

```
Rsquared=1-MSE/var(z2)
```

```
Rsquared = 0.9056
```

```
R=sqrt(Rsquared)
```

```
R = 0.9516
```

Maximum height

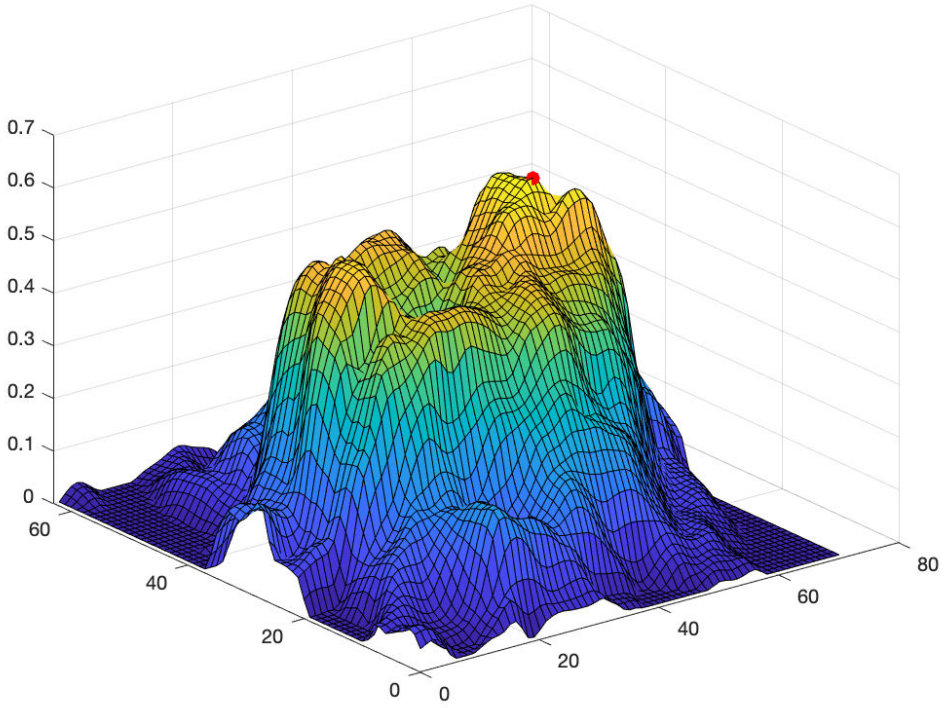
```
maxheight=max(avg,[],'all')
```

```
maxheight = 0.6223
```

```
[rowsOfMaxes colsOfMaxes] = find(avg == maxheight)
```

```
rowsOfMaxes = 32
colsOfMaxes = 50
```

```
surf(avg)
hold on
plot3(colsOfMaxes,rowsOfMaxes,avg(rowsOfMaxes,colsOfMaxes),'.r',"MarkerSize",25)
hold off
```



Volume of the surface

```
[TriIdx, Vol] =convhull(X,Y,avg); %exact data

Vol %real volume
```

```
Vol = 1.3943e+03
```

```
s=sf(x2,y2);
s(s<0)=0;
[TriIdx, Vol1] =convhull(x2,y2,s);
Vol1
```

```
Vol1 = 1.2925e+03
```

```
AbsVol=abs(Vol1-Vol)
```

```
AbsVol = 101.8201
```

```
RelVol=AbsVol/Vol*100 %9.6190 (without robust)
```

```
RelVol = 7.3025
```

```
H_bar=Vol/(x1*y1)
```

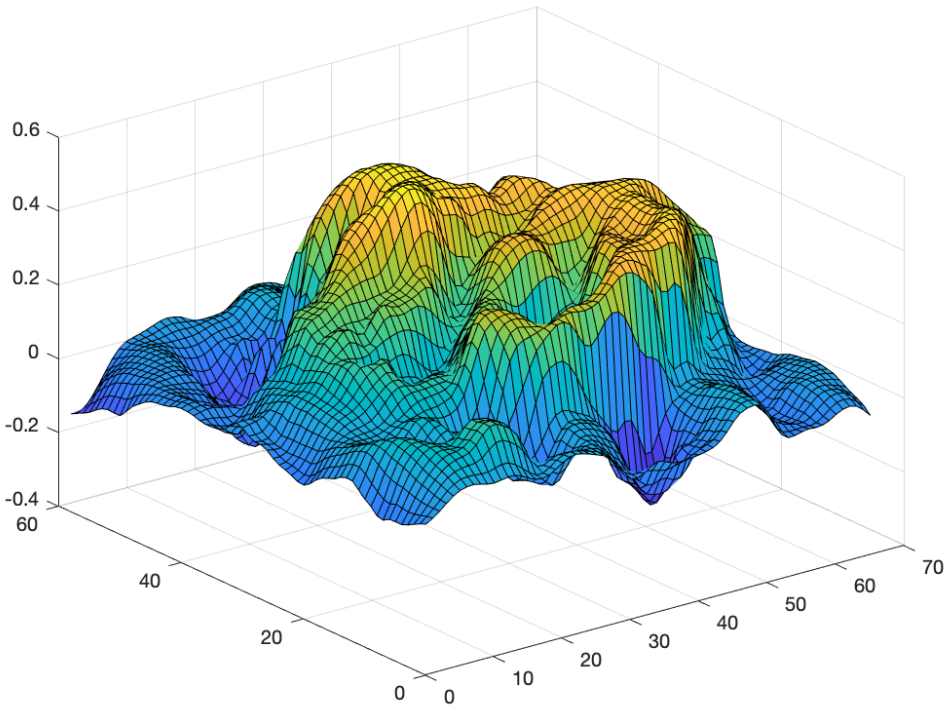
```
H_bar = 0.3117
```

```

```

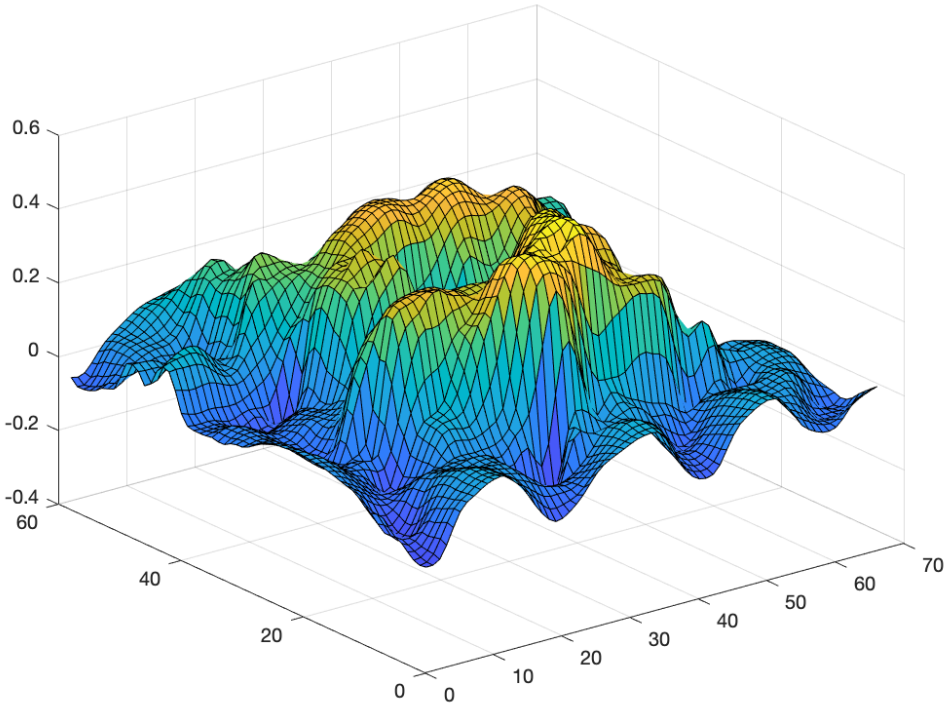
```
ans = 0.1986
diff = 740.3950
```

```
%save('adj_s1.mat','adj')
%save('avg_s1.mat','avg')
%save('var_s1.mat','deviation')
```

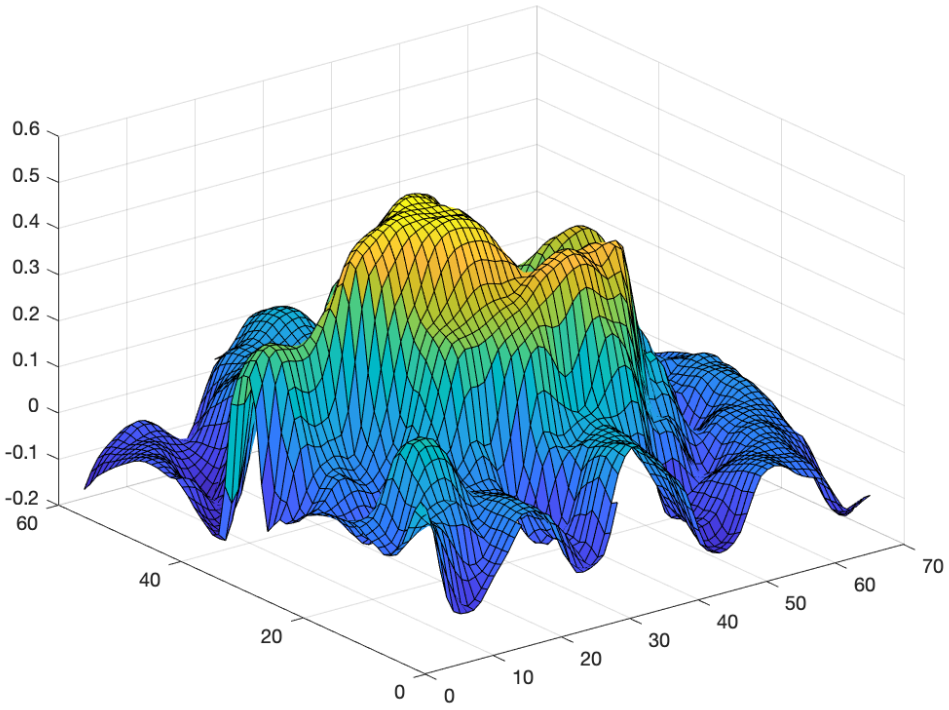
```
clear all
load('samples.mat')
x1=59;
y1=67;
SAMPLE2_1=SAMPLE2_1(5:end-4,:);
SAMPLE2_2=SAMPLE2_2(6:end-6,2:end-2);
S1={SAMPLE2_1,SAMPLE2_2,SAMPLE2_3};
A=zeros(x1,y1,3);
for k=1:3
    s = S1{k};
    s(isnan(s))=0;
    for i=1:x1
        for j=1:y1
            A(i,j,k)=s(i,j); %matrix that contains the 3 samples
        end
    end
end

surf(SAMPLE2_1)
```




```
surf(SAMPLE2_2)
```

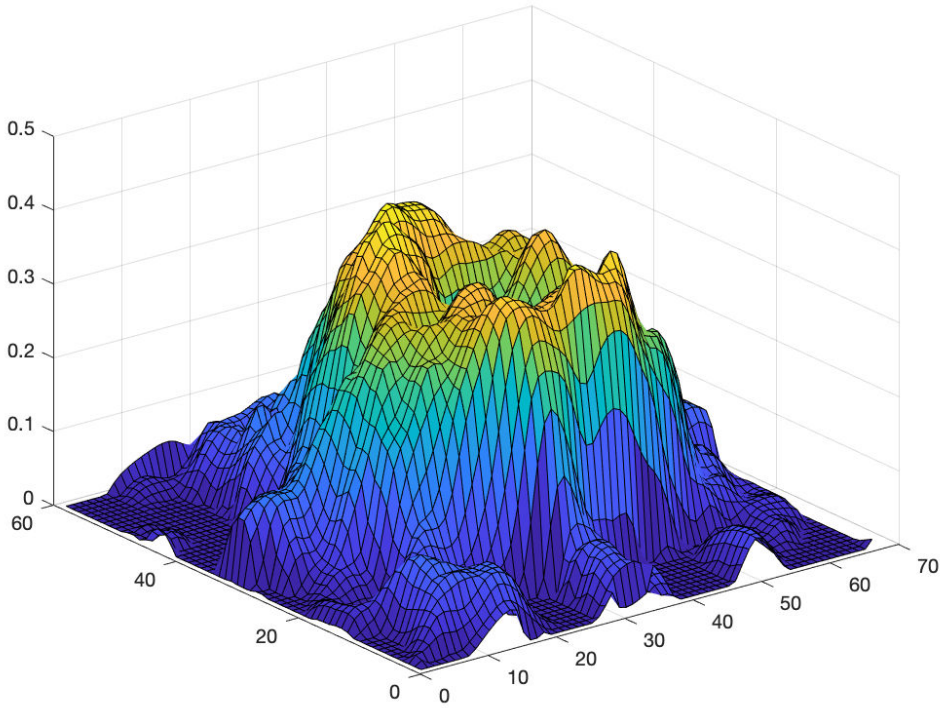



```
surf(SAMPLE2_3)
```



Mean value of the sample and variance for each point (wrt the 3 samples)

```
avg=mean(A,3); %mean value of z for each (x,y) point
avg(avg<0)=0; %deleat negative values
surf(avg)
```



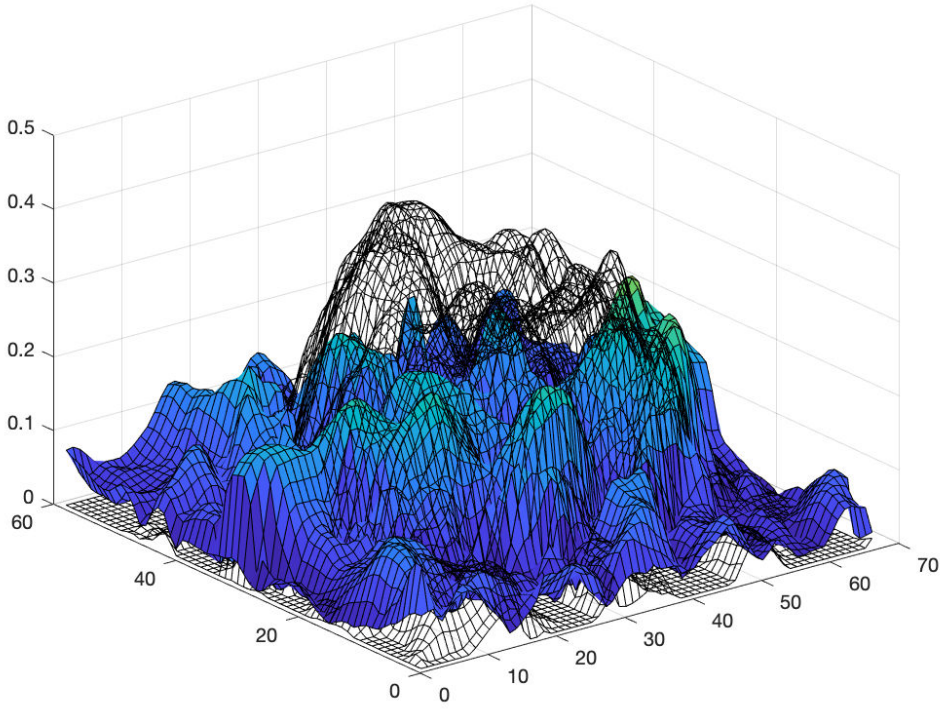

```
deviation = std(A,0,3);
mean(deviation(:))
```

```
ans = 0.0890
```

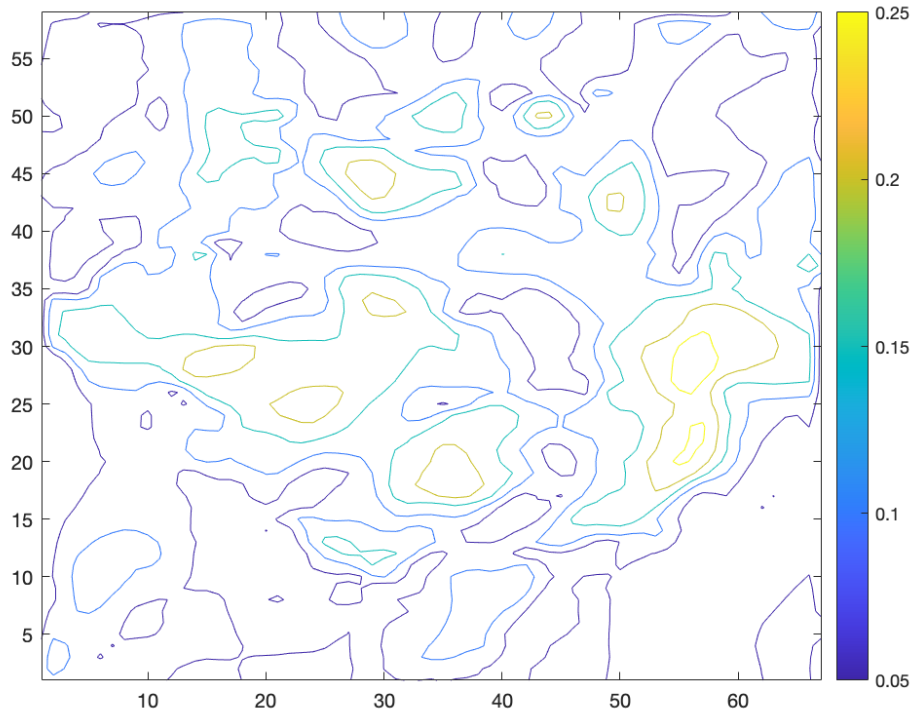
```
variance=var(A,0,3);
mean(variance(:))
```

```
ans = 0.0109
```

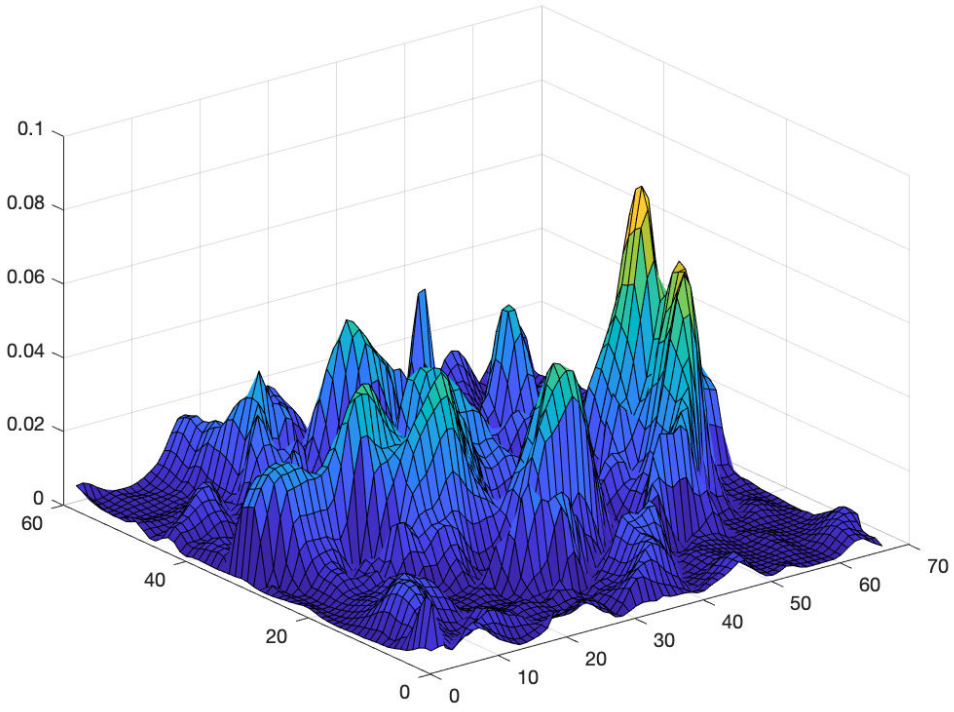
```
figure(1)
surf(avg,"FaceColor","none")
hold on
surf(deviation)
hold off
```




```
figure()
contour(deviation)
colorbar
```




```
surf(variance)
```



Approximating to a function

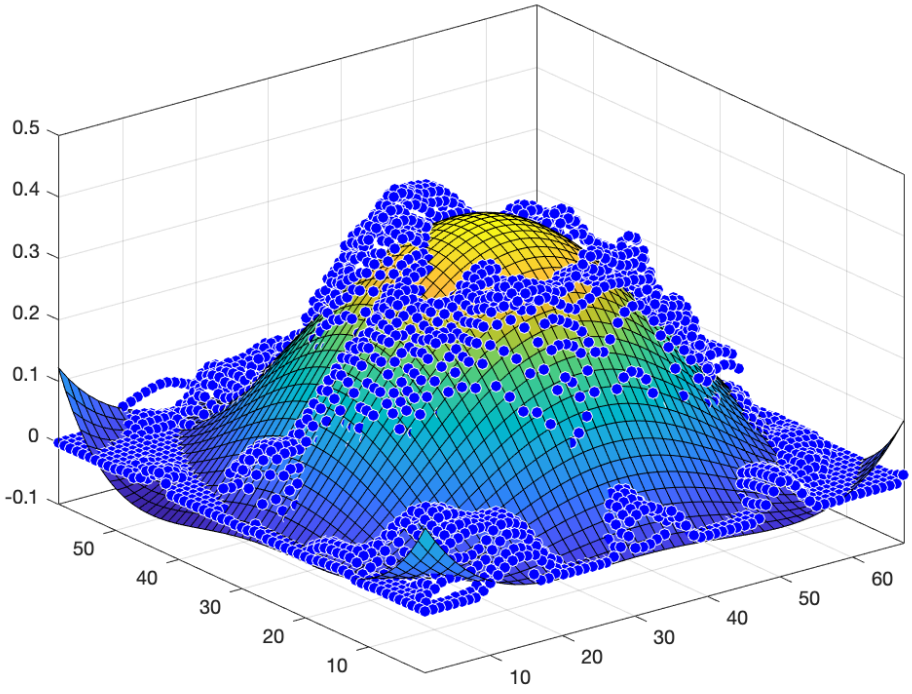
```
[X,Y] = meshgrid(1:y1,1:x1);
%figure(2)
%surf(X,Y,avg) --> used to ensure that the meshgrid was correctly done
%figure(3)
%contour(avg,15)
%colorbar
x2=[];
y2=[];
z2=[];
for i=1:size(X,1)
    for j=1:size(X,2)
        x2=[x2;X(i,j)];
        y2=[y2;Y(i,j)];
        z2=[z2;avg(i,j)];
    end
end

sf = fit([x2, y2],z2,'poly44','Robust','Bisquare' )
```

```
     Linear model Poly44:
     sf(x,y) = p00 + p10*x + p01*y + p20*x^2 + p11*x*y + p02*y^2 + p30*x^3 +
                   p21*x^2*y + p12*x*y^2 + p03*y^3 + p40*x^4 + p31*x^3*y
                   + p22*x^2*y^2 + p13*x*y^3 + p04*y^4
     Coefficients (with 95% confidence bounds):
       p00 =      0.1862  (0.1642, 0.2082)
       p10 =    -0.02509  (-0.02757, -0.0226)
       p01 =    -0.02993  (-0.03276, -0.0271)
       p20 =    0.001166  (0.001048, 0.001283)
       p11 =    0.002349  (0.002245, 0.002454)
       p02 =    0.001657  (0.001505, 0.00181)
       p30 =  -2.356e-05  (-2.592e-05, -2.119e-05)
       p21 =   -3.56e-05  (-3.766e-05, -3.355e-05)
       p12 =  -3.631e-05  (-3.865e-05, -3.398e-05)
       p03 =   -4.06e-05  (-4.408e-05, -3.713e-05)
       p40 =   1.723e-07  (1.554e-07, 1.892e-07)
       p31 =   1.867e-08  (1.967e-09, 3.536e-08)
       p22 =   5.614e-07  (5.428e-07, 5.8e-07)
       p13 =   -3.27e-08  (-5.424e-08, -1.116e-08)
       p04 =   3.541e-07  (3.26e-07, 3.822e-07)
```

```
options = fitoptions(sf)
```

```
options =

    Normalize: 'off'
      Exclude: []
      Weights: []
       Method: 'LinearLeastSquares'
       Robust: 'Bisquare'
        Lower: [1×0 double]
        Upper: [1×0 double]
```

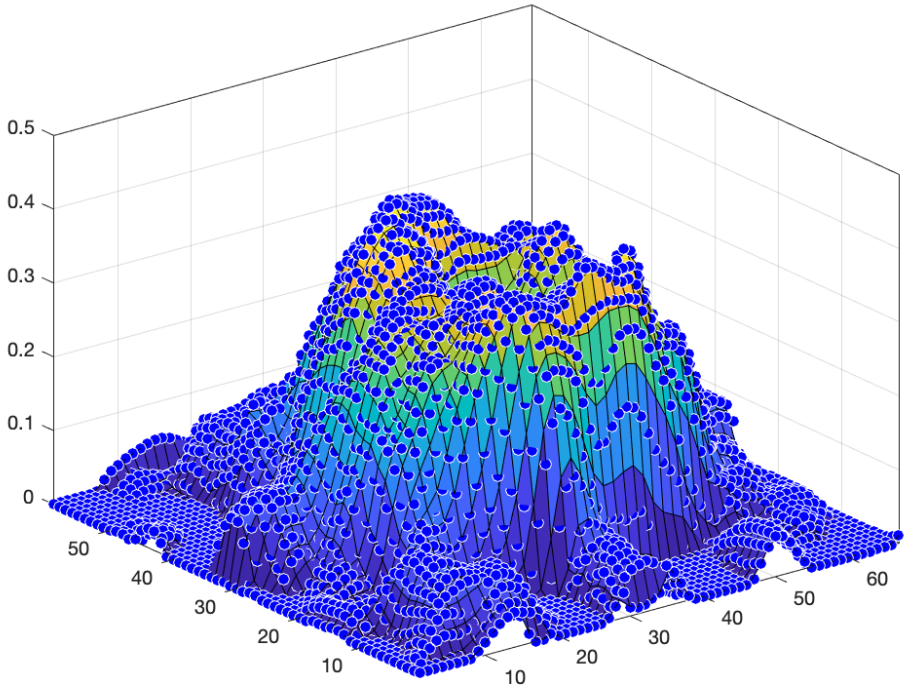
```
figure()
plot(sf,[x2,y2],z2)
```




```
sf1 = fit([x2, y2],z2,'linearinterp')
```

```
Linear interpolant:
  sf1(x,y) = piecewise linear surface computed from p
Coefficients:
  p = coefficient structure
```

```
figure()
plot(sf1,[x2,y2],z2)
```




```
%z3=zeros(63,71);
adj=zeros(x1,y1);
adj1=adj; %a copy
```

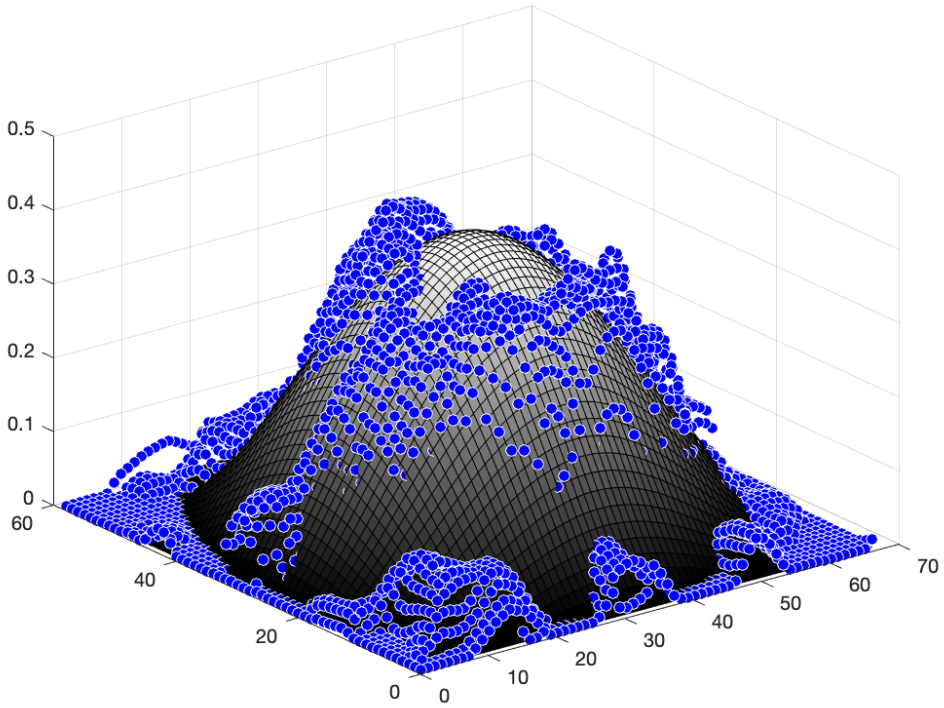
We correct the negative values and those on the extreme that go again up

```
for i=1:length(x2)
    %z3(y2(i),x2(i))= z2(i);
    if sf(x2(i),y2(i)) > 0
        adj(y2(i),x2(i))= sf(x2(i),y2(i));
    else
        adj(y2(i),x2(i))= 0;
    end
    if (x2(i)-35)^2/1200+(y2(i)-30)^2/1200 > 1
        adj(y2(i),x2(i))= 0;
    end

end

%(z3==avg) it is the same
```

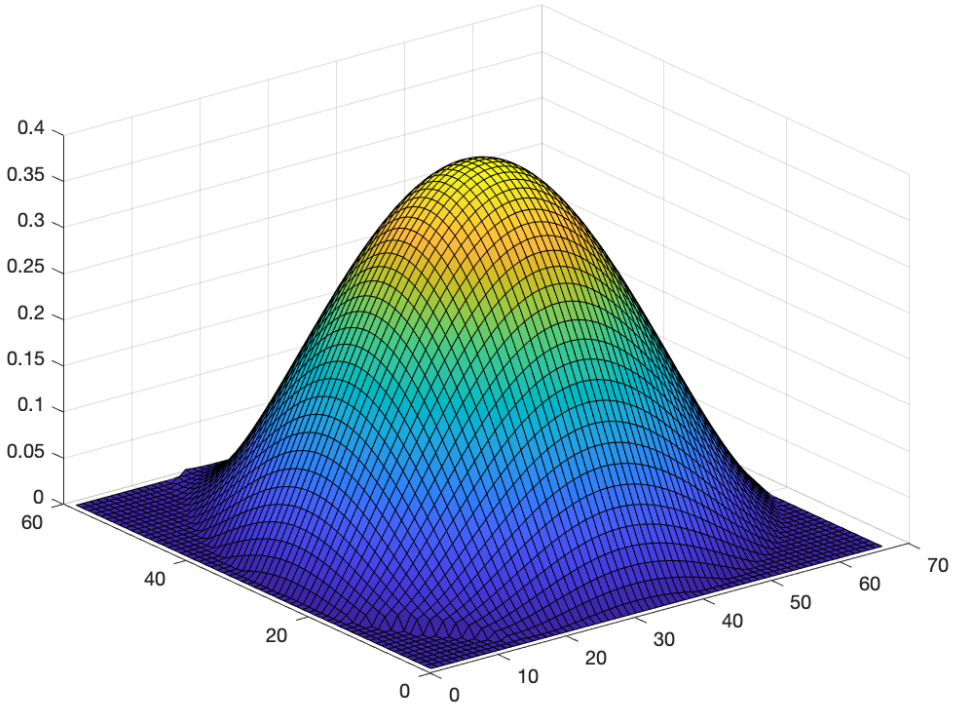
Corrected values: adj and z2

```
surf(adj)
colormap("gray")
hold on
plot3(x2,y2,z2,'ow','MarkerSize',6,'MarkerFaceColor','blue')
```

```
hold off
```




```
surf (adj)
colormap("default")
```



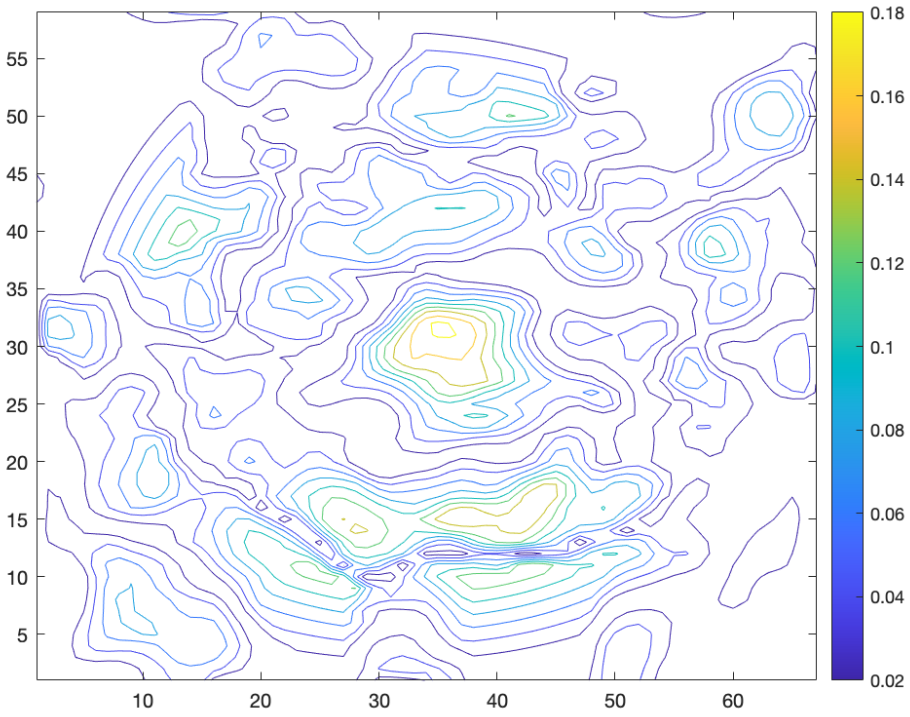
Errors

Absolute and relative errors

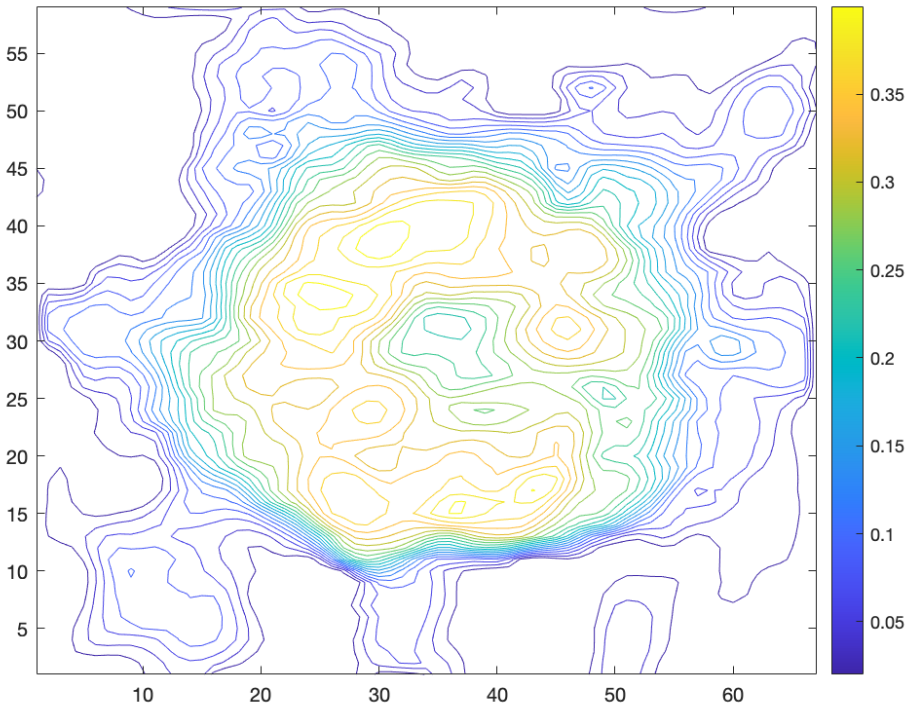
```
AbsErr=abs(adj-avg);
RelErr=AbsErr./avg*100;
mean(AbsErr(:))
```

```
ans = 0.0378
```

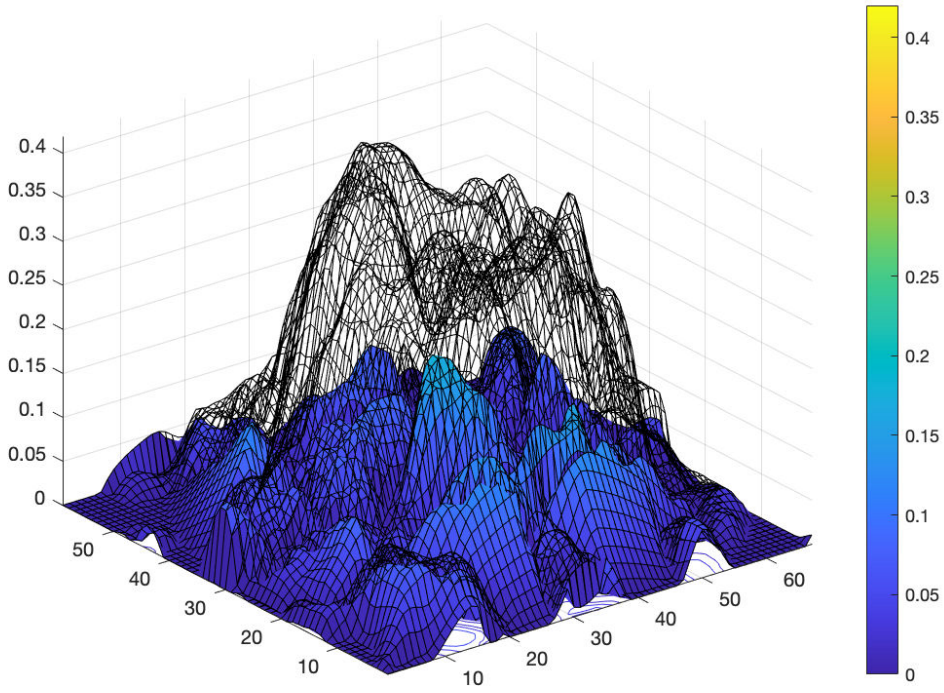
```
figure()
contour(AbsErr)
colorbar
```



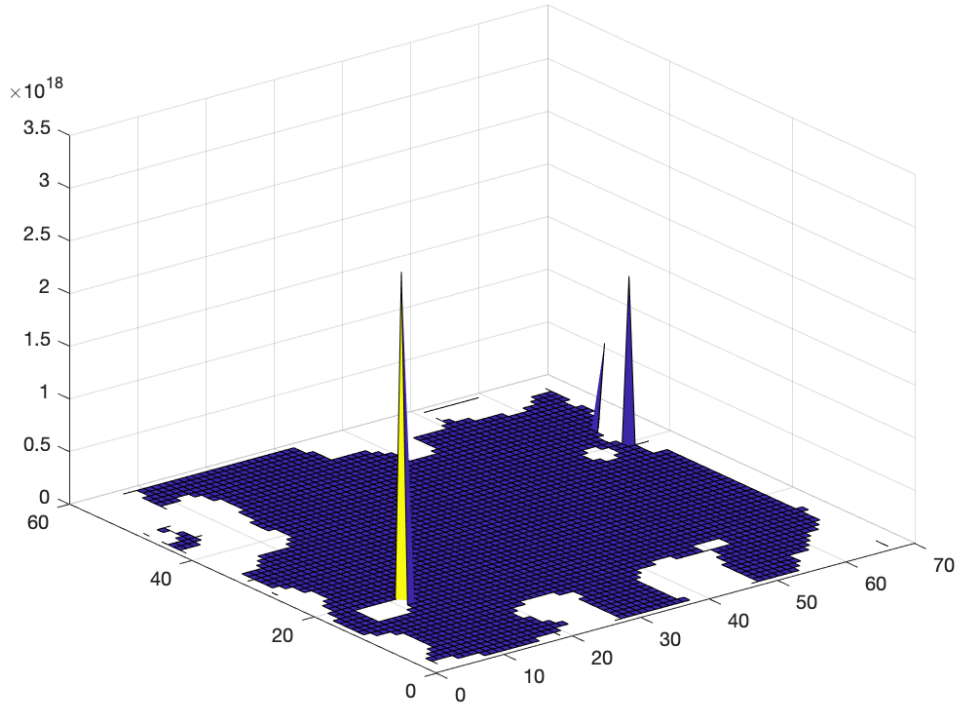

```
contour(avg,20)
colorbar
```



```
figure()
surfc(AbsErr)
hold on
surf(avg,'FaceColor',"none")
colorbar
hold off
```




```
figure()
surf(RelErr)
```




```
ans = NaN
```

Computation of RMSE

```
s=[]; %predicted data corrected
for i=1:size(X,1)
    for j=1:size(X,2)
        s=[s;adj(i,j)];
    end
end
s(s<0)=0;

z2=z2; %exact data
n=length(s);
MSE=1/n*sum((z2-s).^2) %mean squared error
```

```
MSE = 0.0026
```

```
Rsquared=1-MSE/var(z2)
```

```
Rsquared = 0.8416
```

```
R=sqrt(Rsquared)
```

```
R = 0.9174
```

Maximum height

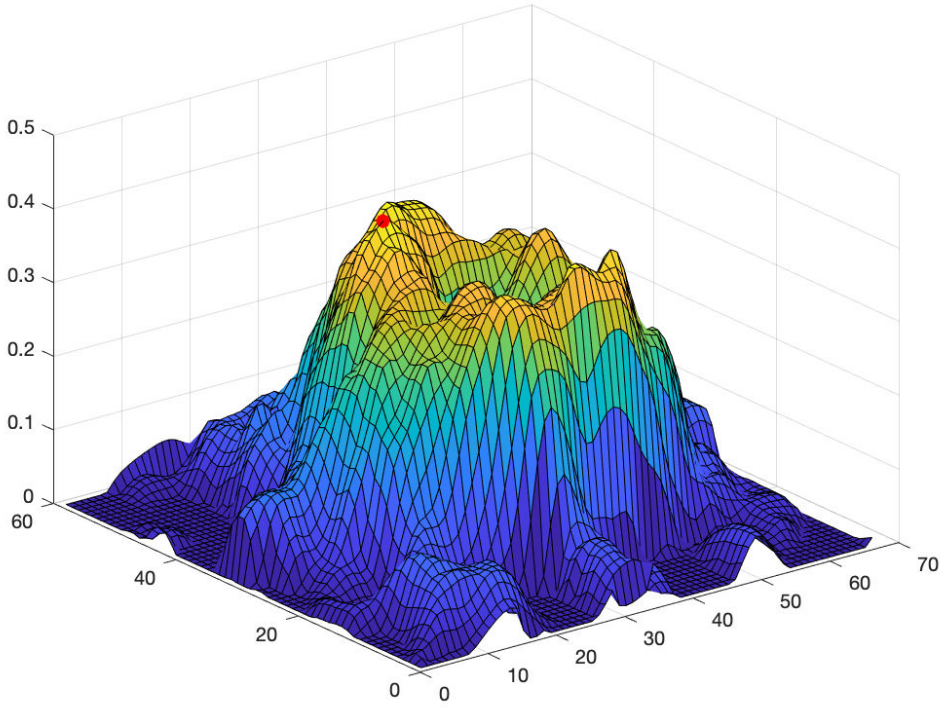
```
maxheight=max(avg,[],'all')
```

```
maxheight = 0.4193
```

```
[rowsOfMaxes colsOfMaxes] = find(avg == maxheight)
```

```
rowsOfMaxes = 34
colsOfMaxes = 25
```

```
surf(avg)
hold on
plot3(colsOfMaxes,rowsOfMaxes,avg(rowsOfMaxes,colsOfMaxes),'.r',"MarkerSize",25)
hold off
```



Volume of the surface

```
[TriIdx, Vol] =convhull(X,Y,avg); %exact data

Vol %real volume
```

```
Vol = 833.5484
```

```
s=sf(x2,y2);
s(s<0)=0;
[TriIdx, Vol1] =convhull(x2,y2,s);
Vol1
```

```
Vol1 = 853.1651
```

```
AbsVol=abs(Vol1-Vol)
```

```
AbsVol = 19.6166
```

```
RelVol=AbsVol/Vol*100 %9.6190 (without robust)
```

```
RelVol = 2.3534
```

```
H_bar=Vol/(x1*y1)
```

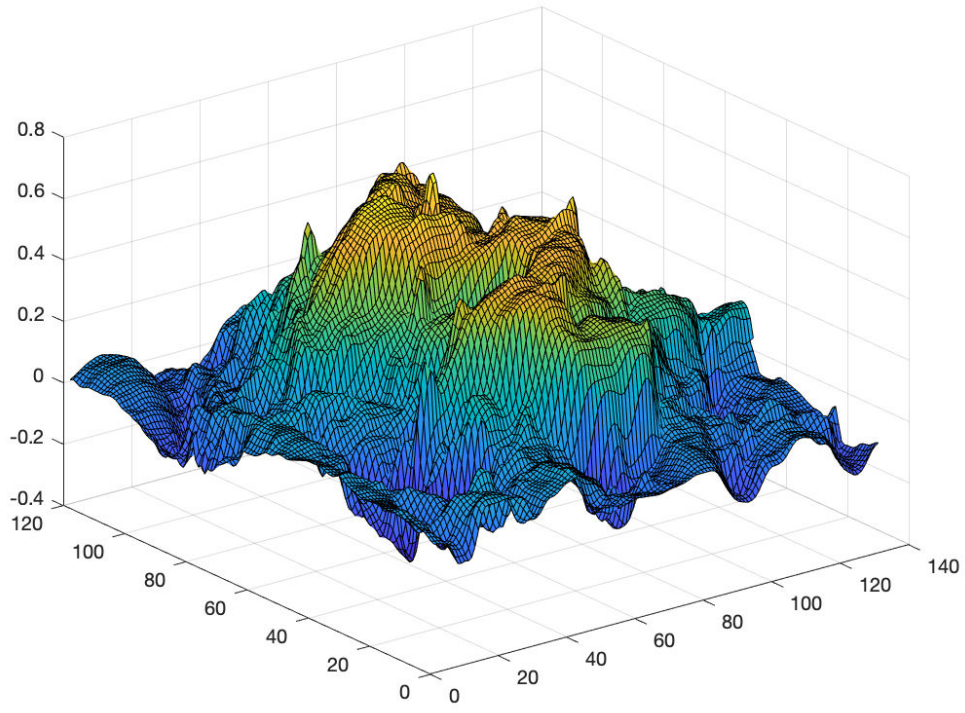
```
H_bar = 0.2109
```

```
z_lim=mean([H_bar,maxheight])
```

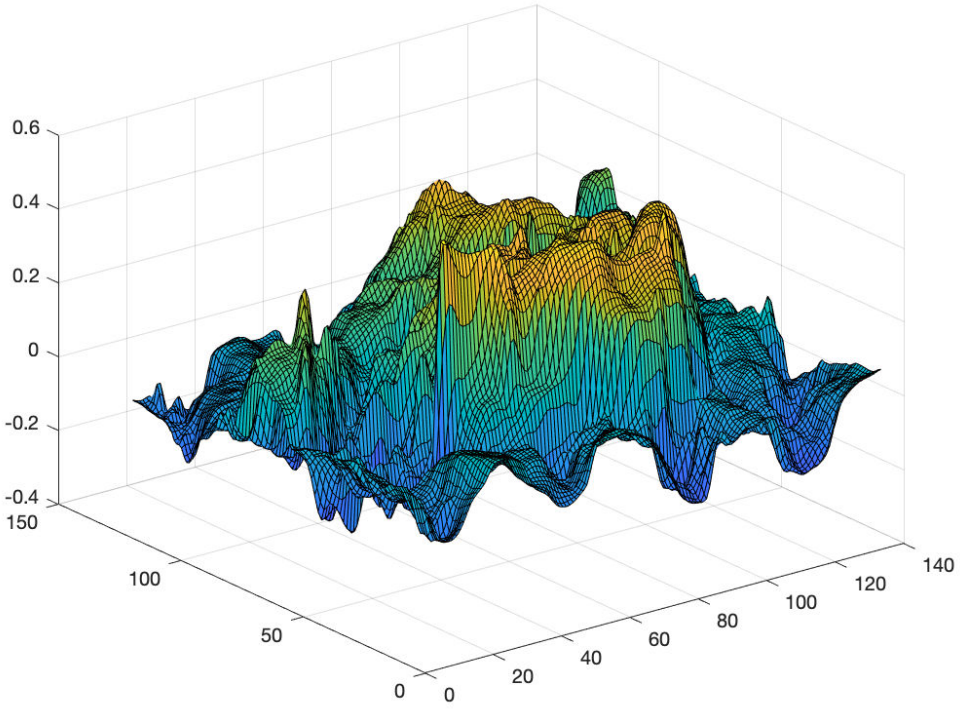
```
z_lim = 0.3151
UD = 0.7378
```

```
%UD=sum(z2>z_lim)/length(z2)
%save('adj_s2.mat',"adj")
%save('avg_s2.mat','avg')
%save('var_s2.mat','deviation')
```

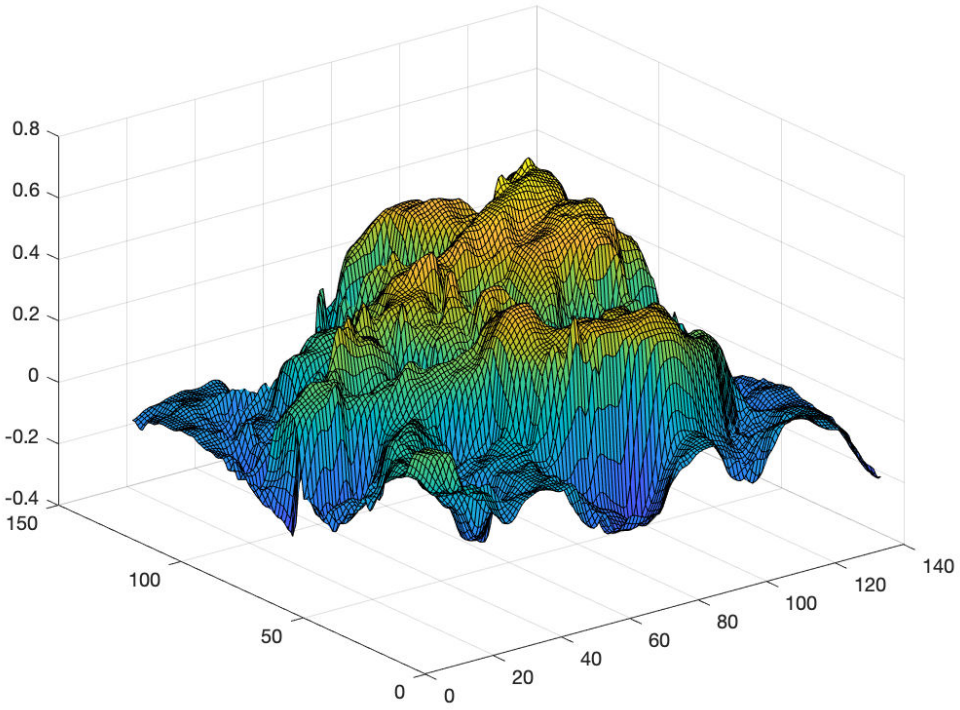
```
clear all
load('samples.mat')
x1=121;
y1=134;
SAMPLE3_2=SAMPLE3_2(9:end-9,7:end-6);
SAMPLE3_3=SAMPLE3_3(9:end-10,4:end-4);

S1={SAMPLE3_1,SAMPLE3_2,SAMPLE3_3};
A=zeros(x1,y1,3);
for k=1:3
    s = S1{k};
    s(isnan(s))=0;
    for i=1:x1
        for j=1:y1
            A(i,j,k)=s(i,j); %matrix that contains the 3 samples
        end
    end
end

surf(S1{1})
```




```
surf(S1{2})
```

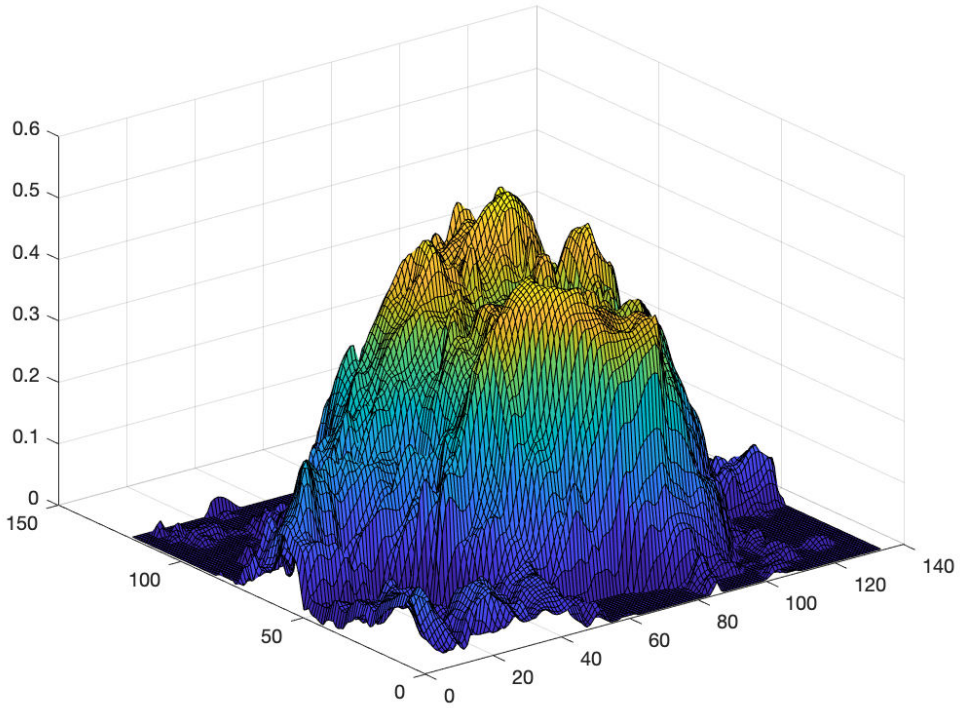

0.6
0.4
0.2
0
-0.2
-0.4
150
100
50
0
0
20
40
60
80
100
120
140

```
surf(S1{3})
```


0.8
0.6
0.4
0.2
0
-0.2
-0.4
150
100
50
0
0
20
40
60
80
100
120
140

Mean value of the sample and variance for each point (wrt the 3 samples)

```
avg=mean(A,3); %mean value of z for each (x,y) point
avg(avg<0)=0; %deleat negative values
surf(avg) %already corrected
```



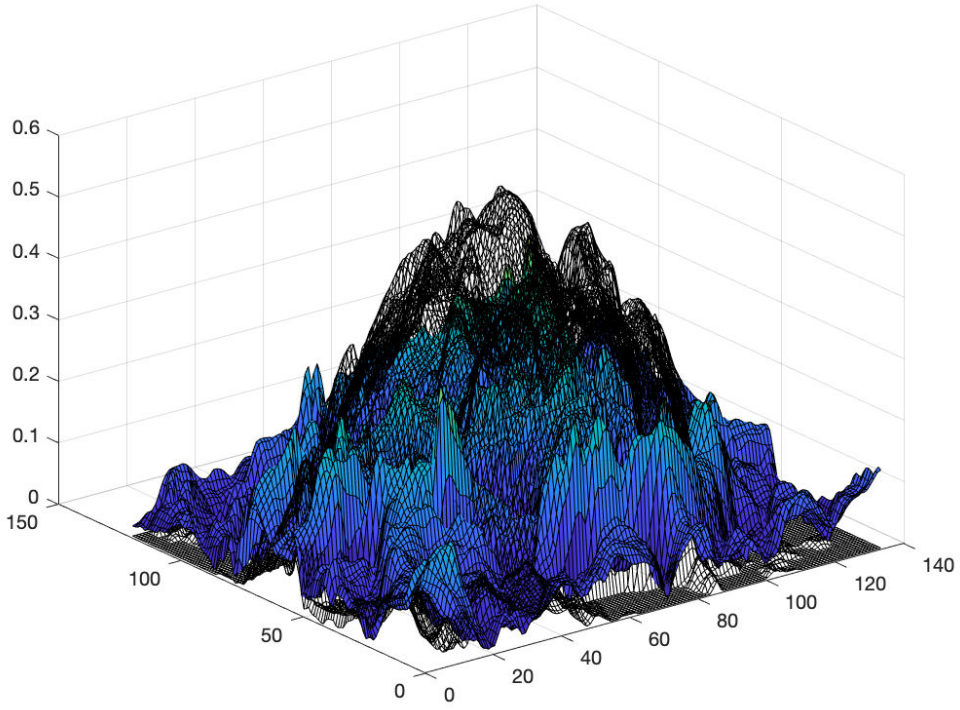

```
deviation = std(A,0,3);
mean(deviation(:))
```

```
ans = 0.1048
```

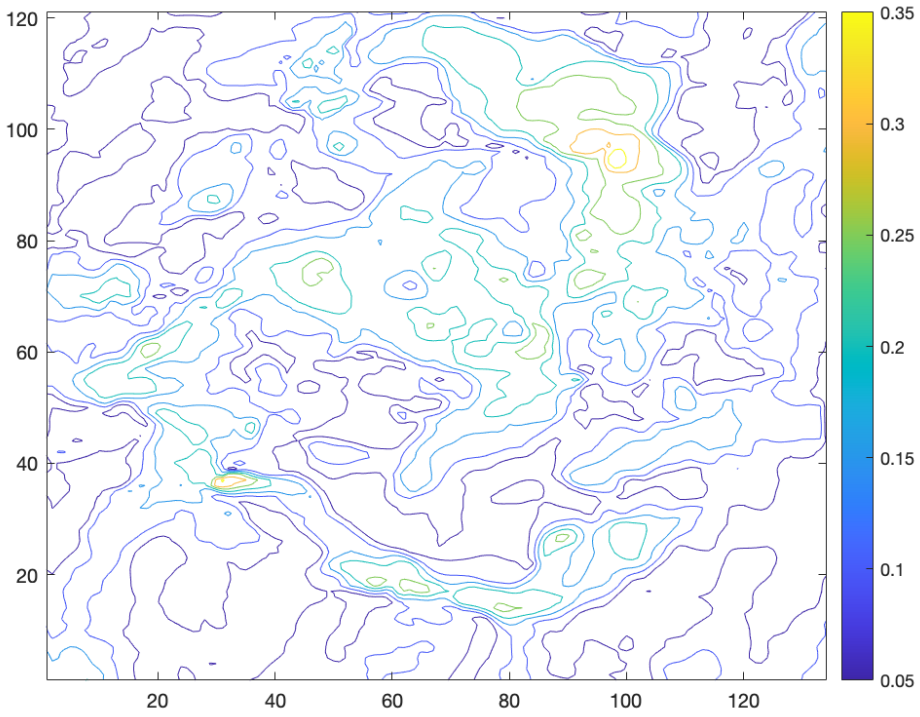
```
variance=var(A,0,3);
mean(variance(:))
```

```
ans = 0.0153
```

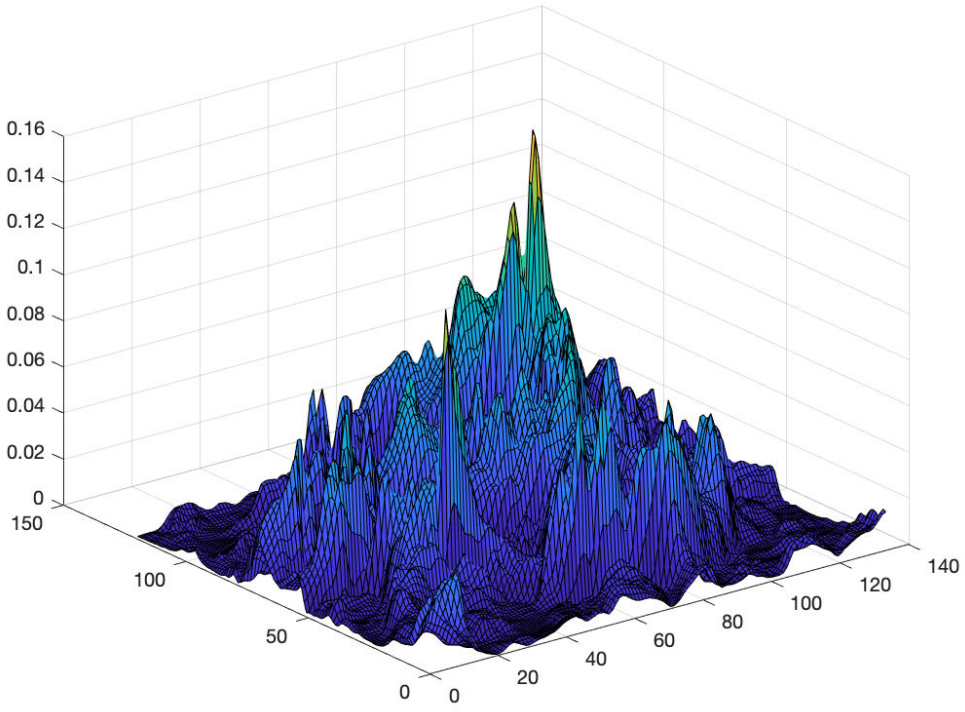
```
figure(1)
surf(avg,"FaceColor","none")
hold on
surf(deviation)
hold off
```




```
figure()
contour(deviation)
colorbar
```




```
surf(variance)
```



Approximating to a function

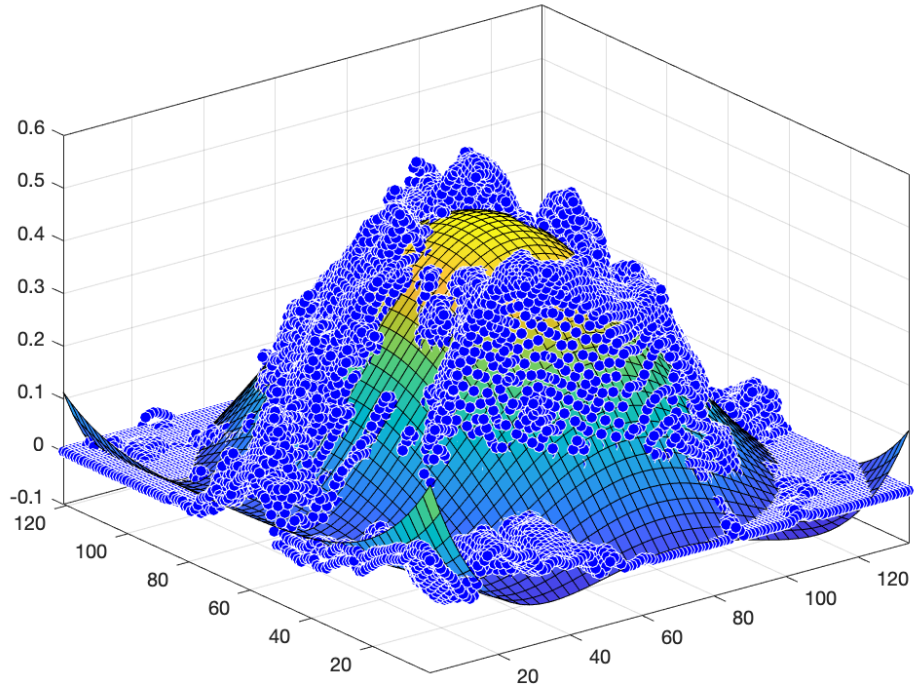
```
[X,Y] = meshgrid(1:y1,1:x1);
%figure(2)
%surf(X,Y,avg) --> used to ensure that the meshgrid was correctly done
%figure(3)
%contour(avg,15)
%colorbar
x2=[];
y2=[];
z2=[];
for i=1:size(X,1)
    for j=1:size(X,2)
        x2=[x2;X(i,j)];
        y2=[y2;Y(i,j)];
        z2=[z2;avg(i,j)];
    end
end

sf = fit([x2, y2],z2,'poly44','Robust','Bisquare' )
```

```
     Linear model Poly44:
     sf(x,y) = p00 + p10*x + p01*y + p20*x^2 + p11*x*y + p02*y^2 + p30*x^3 +
                   p21*x^2*y + p12*x*y^2 + p03*y^3 + p40*x^4 + p31*x^3*y
                   + p22*x^2*y^2 + p13*x*y^3 + p04*y^4
     Coefficients (with 95% confidence bounds):
       p00 =      0.2927  (0.2807, 0.3046)
       p10 =    -0.02409  (-0.02478, -0.02341)
       p01 =    -0.01308  (-0.01384, -0.01232)
       p20 =   0.0006216  (0.0006052, 0.0006381)
       p11 =   0.0005997  (0.0005855, 0.000614)
       p02 =   0.0002669  (0.0002467, 0.0002871)
       p30 =  -6.388e-06  (-6.554e-06, -6.221e-06)
       p21 =  -4.079e-06  (-4.22e-06, -3.937e-06)
       p12 =  -4.737e-06  (-4.894e-06, -4.58e-06)
       p03 =  -2.518e-06  (-2.744e-06, -2.291e-06)
       p40 =   2.254e-08  (2.194e-08, 2.314e-08)
       p31 =  -1.828e-09  (-2.407e-09, -1.249e-09)
       p22 =   3.441e-08  (3.378e-08, 3.504e-08)
       p13 =   2.874e-10  (-4.223e-10, 9.972e-10)
       p04 =   9.211e-09  (8.31e-09, 1.011e-08)
```

```
options = fitoptions(sf)
```

```
options =

    Normalize: 'off'
      Exclude: []
      Weights: []
       Method: 'LinearLeastSquares'
       Robust: 'Bisquare'
        Lower: [1×0 double]
        Upper: [1×0 double]
```

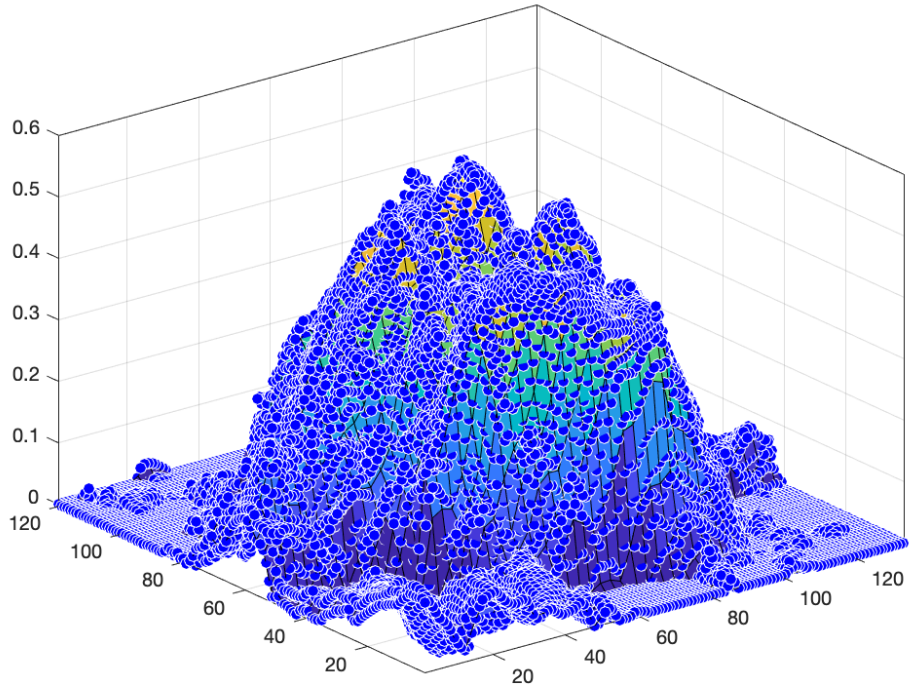
```
figure()
plot(sf,[x2,y2],z2)
```




```
sf1 = fit([x2, y2],z2,'linearinterp')
```

```
Linear interpolant:
  sf1(x,y) = piecewise linear surface computed from p
Coefficients:
  p = coefficient structure
```

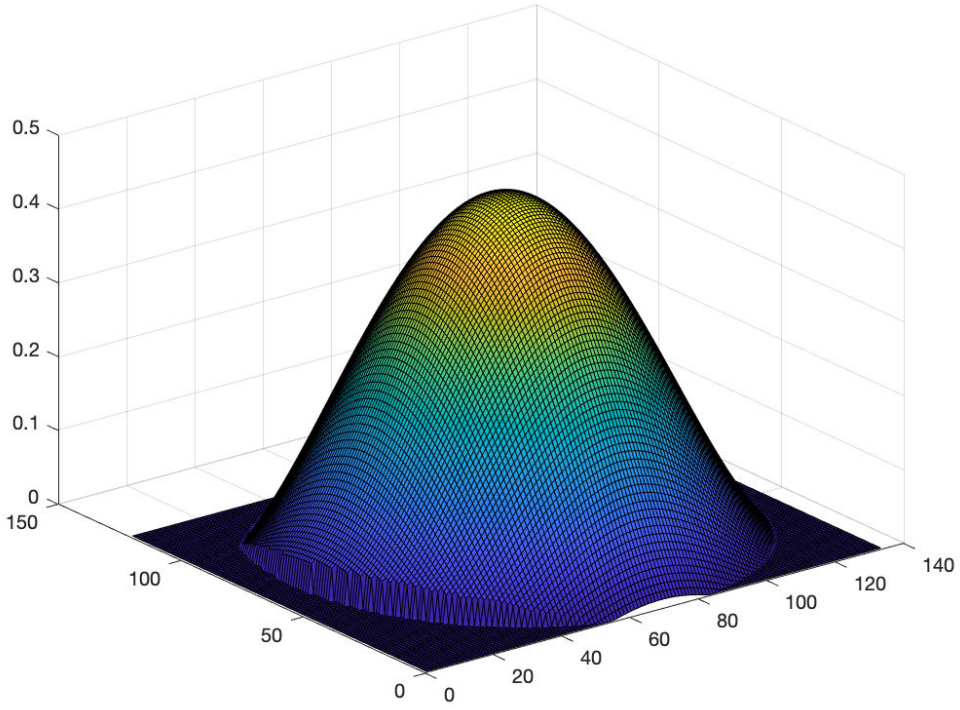
```
figure()
plot(sf1,[x2,y2],z2)
```




```
%z3=zeros(63,71);
adj=zeros(x1,y1);
adj1=adj; %a copy
```

We correct the negative values and those on the extreme that go again up

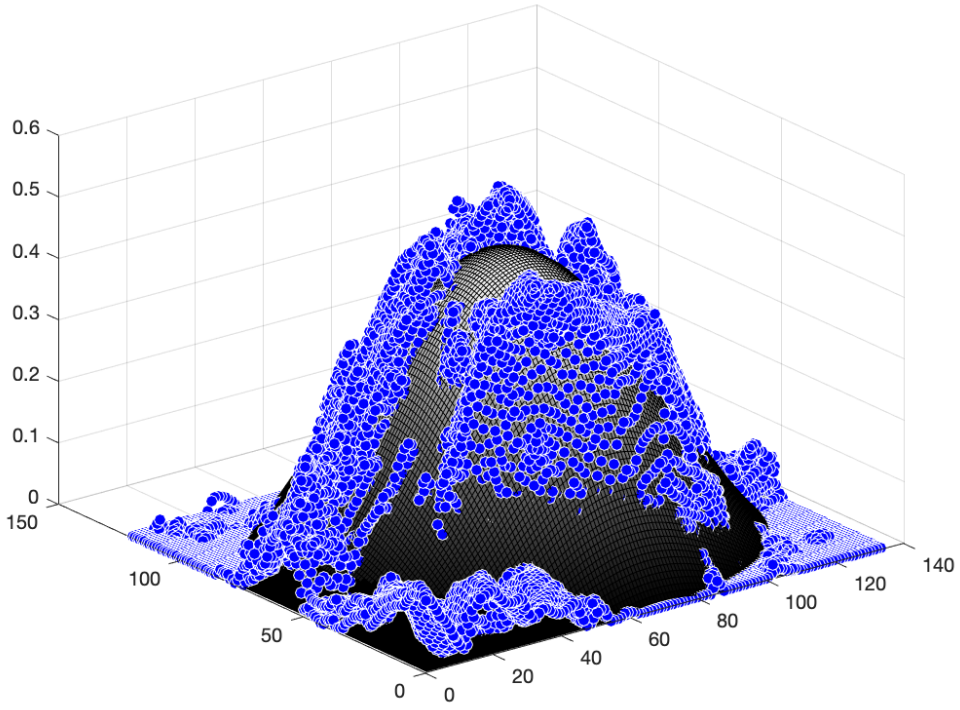
```
for i=1:length(x2)
    %z3(y2(i),x2(i))= z2(i);
    if sf(x2(i),y2(i)) > 0
        adj(y2(i),x2(i))= sf(x2(i),y2(i));
    else
        adj(y2(i),x2(i))= 0;
    end
    if (x2(i)-74)^2/4000+(y2(i)-68)^2/5000 > 1
        adj(y2(i),x2(i))= 0;
    end
    %if y2(i) < 60-3*x2(i)
        %adj(y2(i),x2(i))= 0.05;
    %elseif y2(i) > 259/3-10/21*x2(i)
        %adj(y2(i),x2(i))= 0;
    %elseif y2(i) < -39+5/7*x2(i)
        %adj(y2(i),x2(i))= 0;

    %end
end
surf (adj)
```



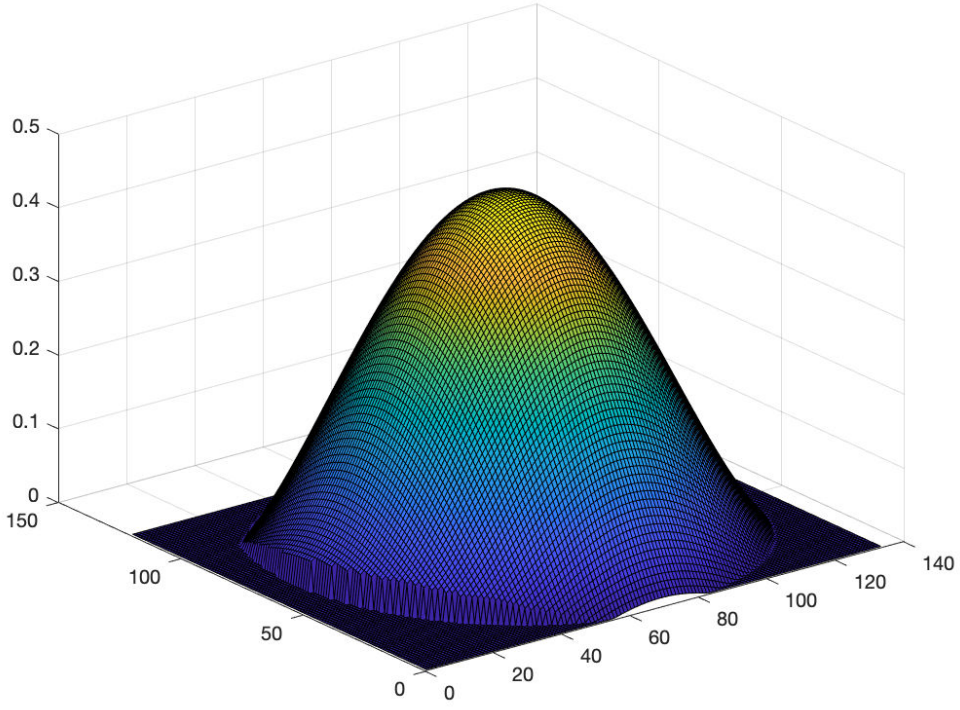

```
%(z3==avg) it is the same
```

Corrected values: adj and z2

```
surf(adj)
colormap("gray")
hold on
plot3(x2,y2,z2,'ow','MarkerSize',6,'MarkerFaceColor','blue')
hold off
```




```
surf (adj)
colormap("default")
```



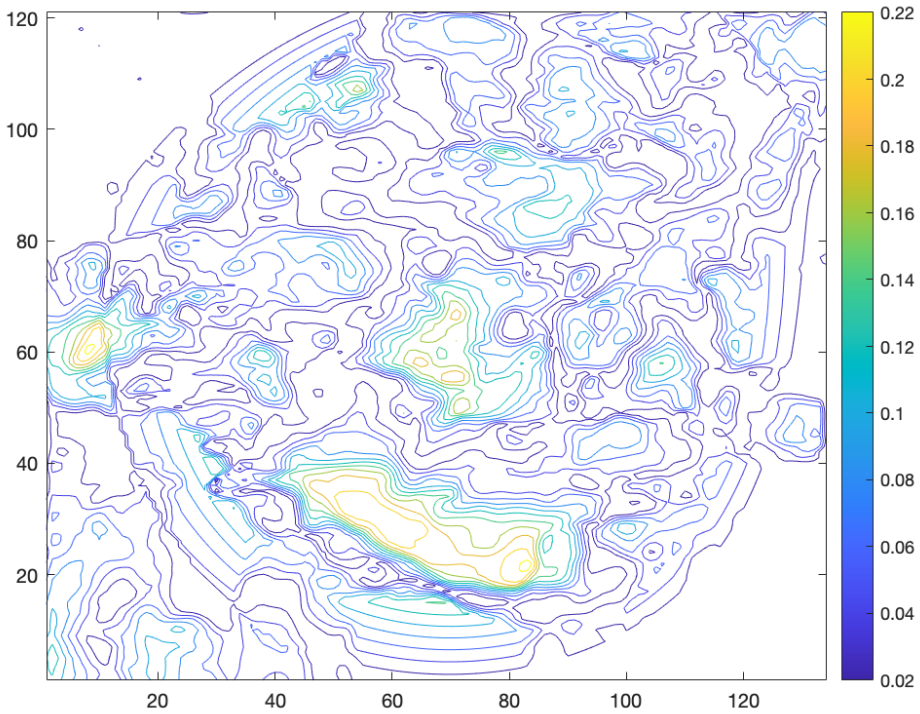
Errors

Absolute and relative errors

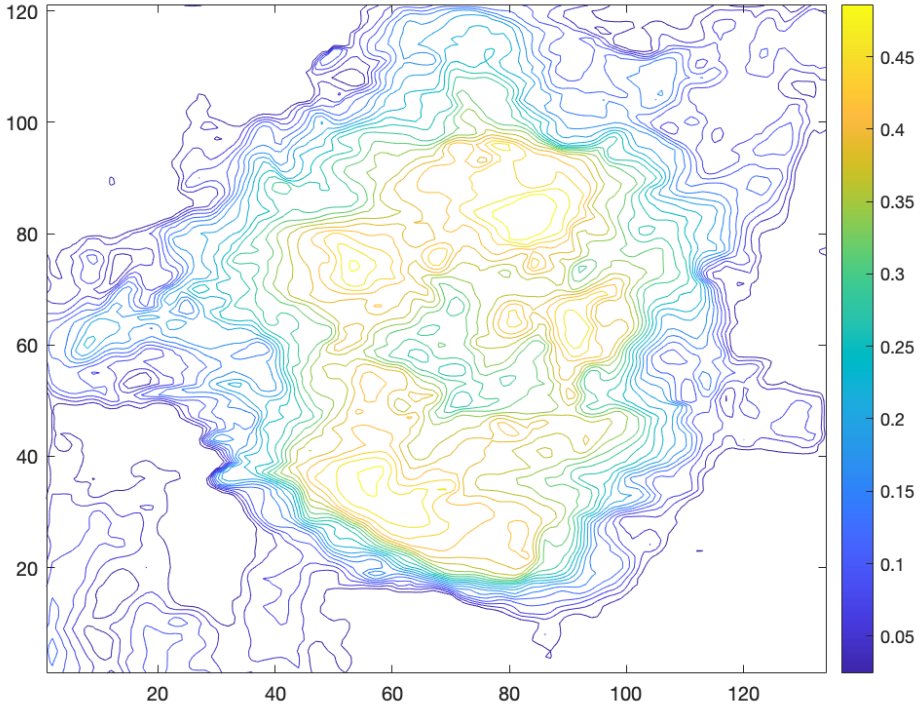
```
AbsErr=abs(adj-avg);
RelErr=AbsErr./avg*100;
mean(AbsErr(:))
```

```
ans = 0.0468
```

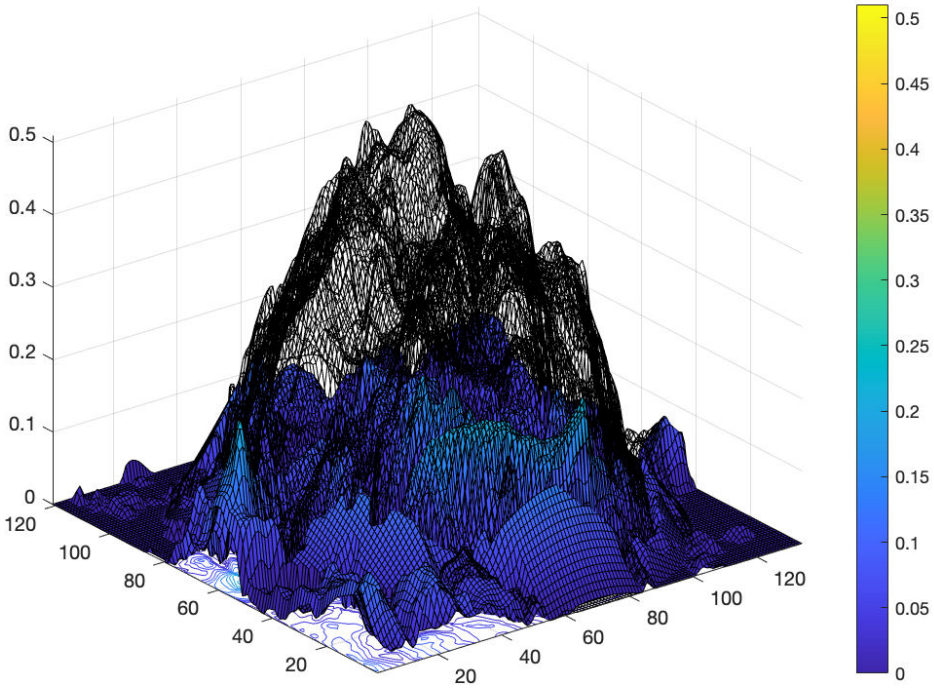
```
figure()
contour(AbsErr)
colorbar
```



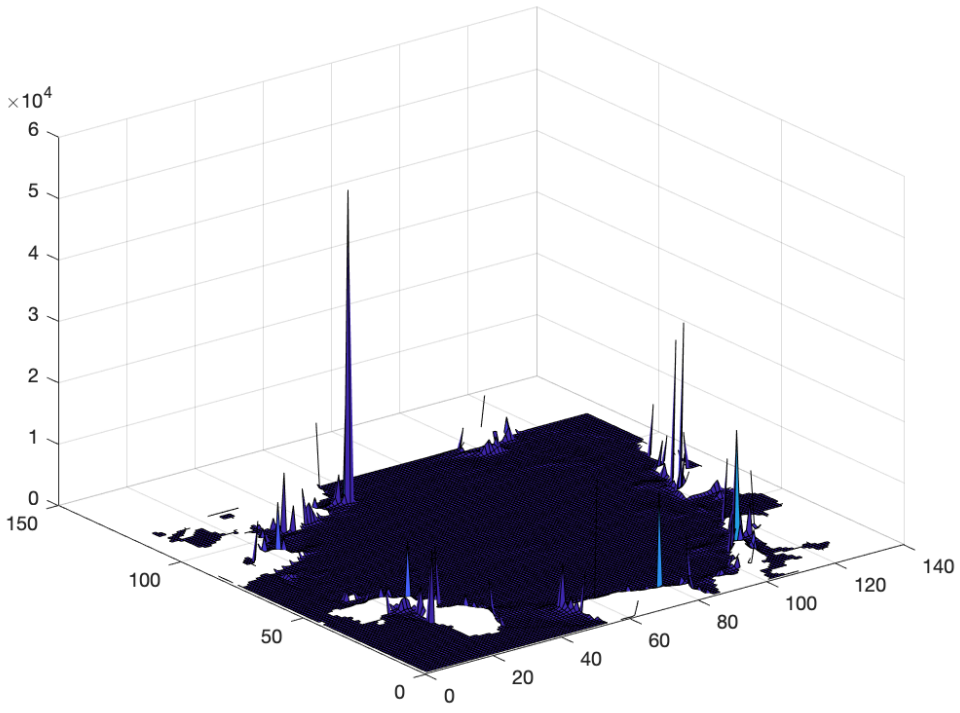

```
contour(avg,20)
colorbar
```




```
figure()
surfc(AbsErr)
hold on
surf(avg,'FaceColor',"none")
colorbar
hold off
```




```
figure()
surf(RelErr)
```



```
mean(RelErr(:))
```

```
ans = NaN
```

Computation of RMSE

```
s=[]; %predicted data corrected
for i=1:size(X,1)
    for j=1:size(X,2)
        s=[s;adj(i,j)];
    end
end
s(s<0)=0;

z2=z2; %exact data
n=length(s);
MSE=1/n*sum((z2-s).^2) %mean squared error
```

```
MSE = 0.0040
```

```
Rsquared=1-MSE/var(z2)
```

```
Rsquared = 0.8252
```

```
R=sqrt(Rsquared)
```

```
R = 0.9084
```

Maximum height

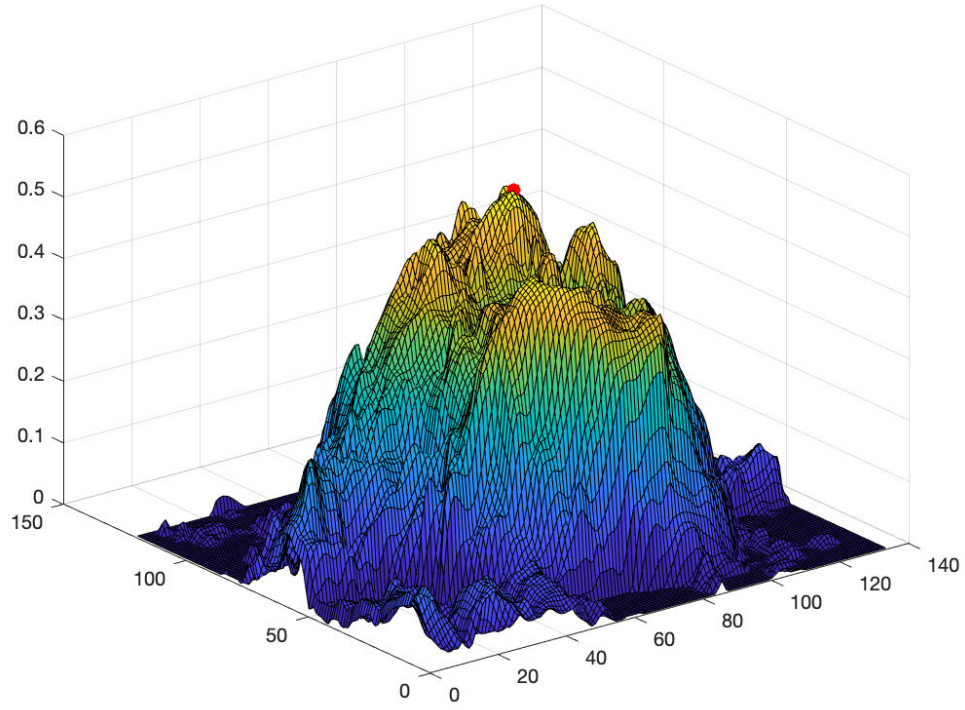
```
maxheight=max(avg,[],'all')
```

```
maxheight = 0.5097
```

```
[rowsOfMaxes colsOfMaxes] = find(avg == maxheight)
```

```
rowsOfMaxes = 82
colsOfMaxes = 83
```

```
surf(avg)
hold on
plot3(colsOfMaxes,rowsOfMaxes,avg(rowsOfMaxes,colsOfMaxes),'.r',"MarkerSize",25)
hold off
```



Volume of the surface

```
[TriIdx, Vol] =convhull(X,Y,avg); %exact data

Vol %real volume
```

```
Vol = 4.6555e+03
```

```
s=sf(x2,y2);
s(s<0)=0;
[TriIdx, Vol1] =convhull(x2,y2,s);
Vol1
```

```
Vol1 = 4.2323e+03
```

```
AbsVol=abs(Vol1-Vol)
```

```
AbsVol = 423.2043
```

```
RelVol=AbsVol/Vol*100 %9.6190 (without robust)
```

```
RelVol = 9.0904
```

```


H_bar=Vol/(x1*y1)
```

```
H_bar = 0.2871
```

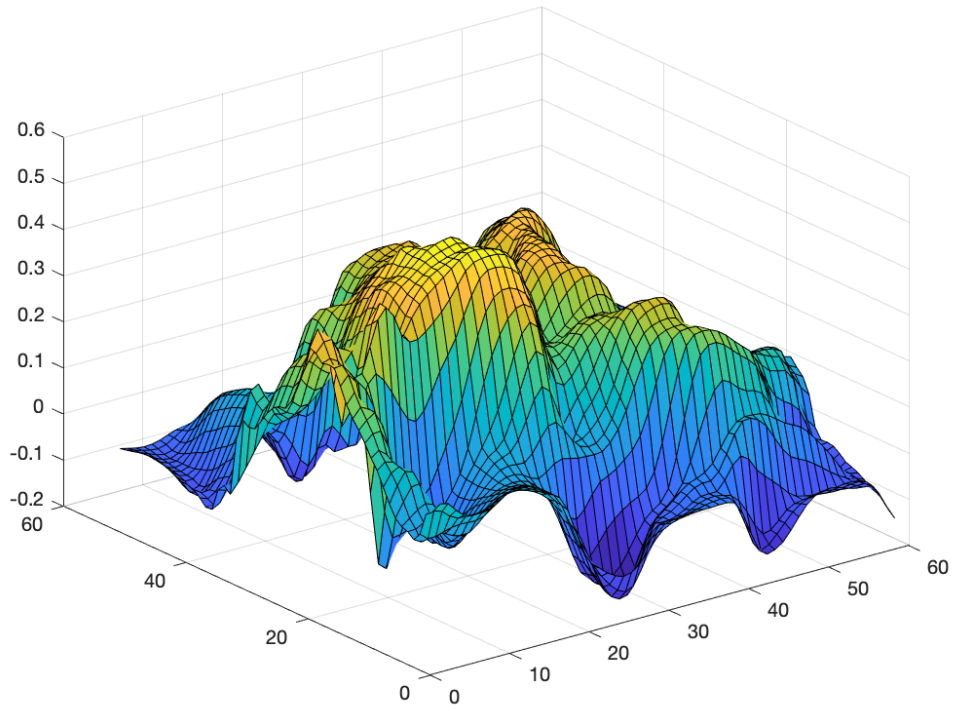
```

```

```
z_lim = 0.2967
UD = 0.5917
```

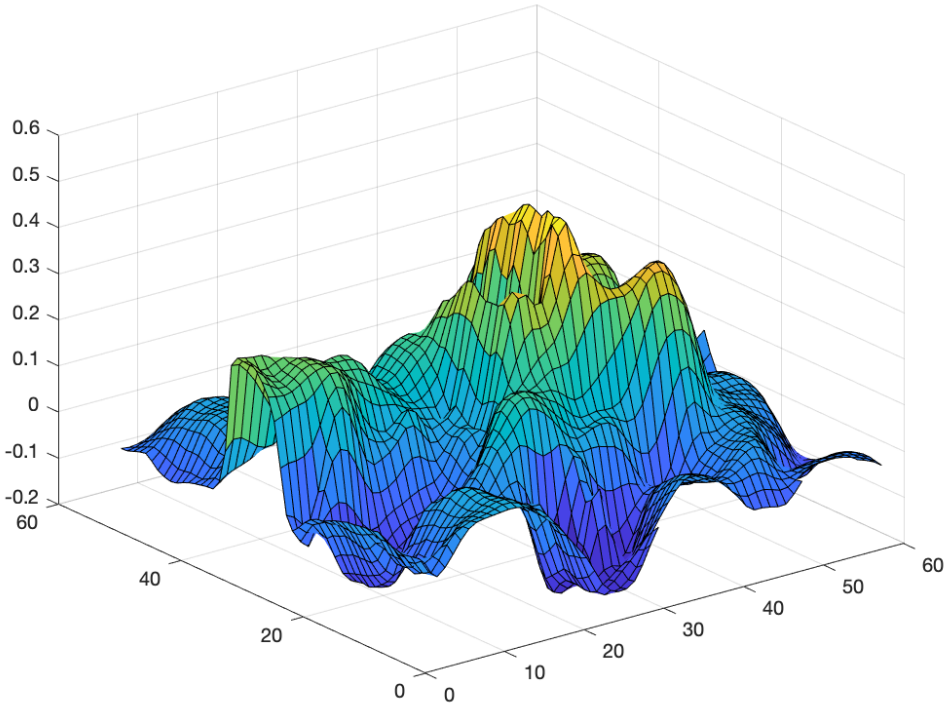
```
%save('adj_s3.mat',"adj")
%save('avg_s3.mat','avg')
%save('var_s3.mat','deviation')
```

```
ans = 0.1565
diff = 2.1787e+03
```

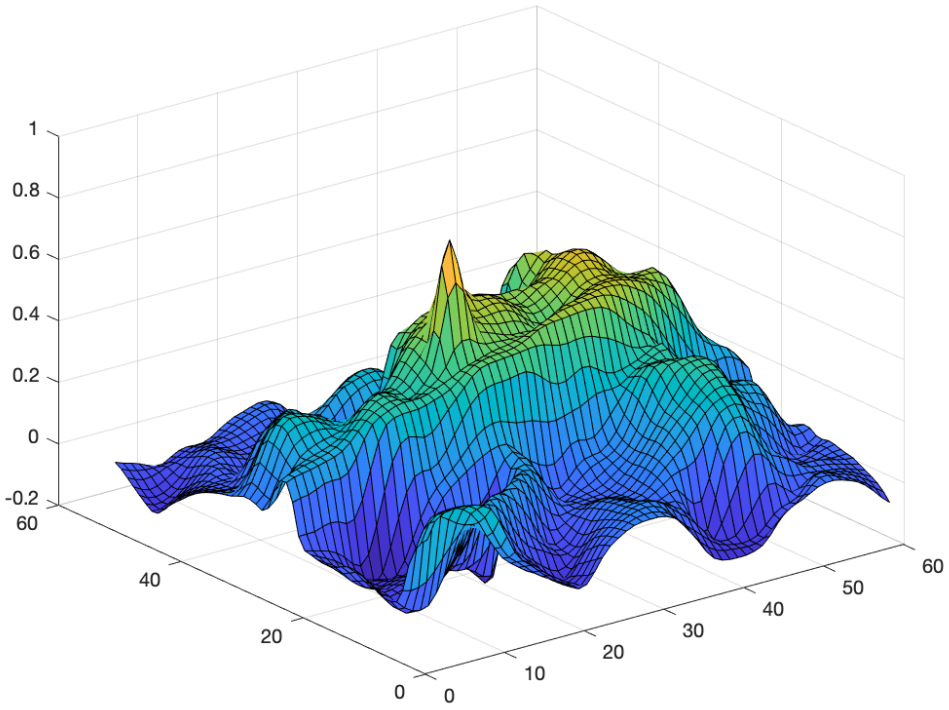
```
clear all
load('samples.mat')
x1=52;
y1=59;
SAMPLE4_1=SAMPLE4_1(4:end-4,4:end-5);
SAMPLE4_3=SAMPLE4_3(6:end-5,8:end-8);
S1={SAMPLE4_1,SAMPLE4_2,SAMPLE4_3};
A=zeros(x1,y1,3);
for k=1:3
    s = S1{k};
    s(isnan(s))=0;
    for i=1:x1
        for j=1:y1
            A(i,j,k)=s(i,j); %matrix that contains the 3 samples
        end
    end
end

surf(S1{1})
```




```
surf(S1{2})
```

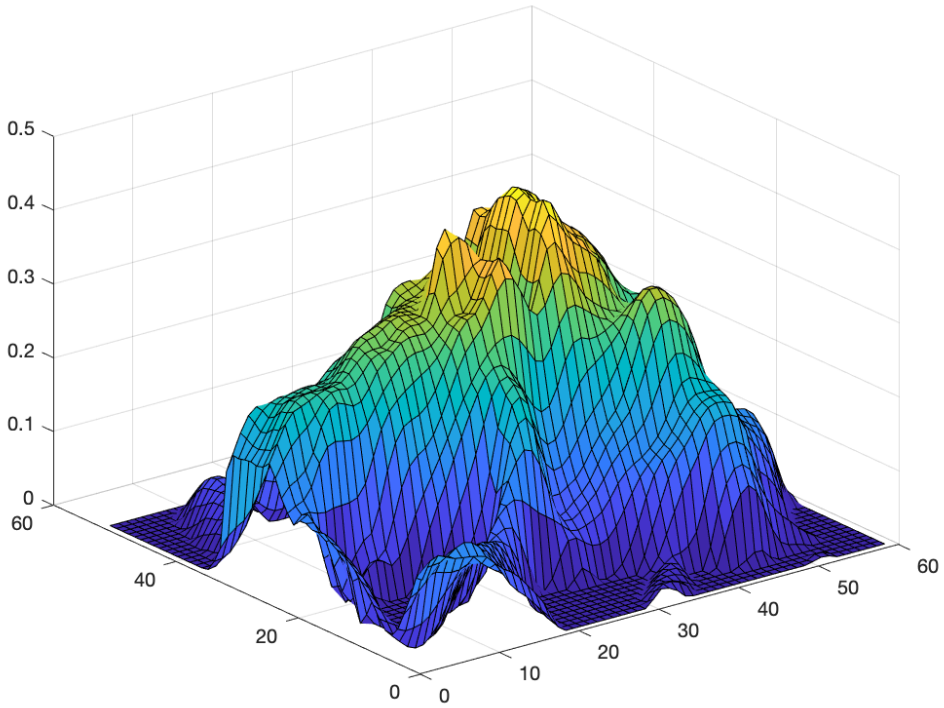

0.6
0.5
0.4
0.3
0.2
0.1
0
-0.1
-0.2
60
40
20
0
0
10
20
30
40
50
60

```
surf(S1{3})
```


1
0.8
0.6
0.4
0.2
0
-0.2
60
40
20
0
0
10
20
30
40
50
60

Mean value of the sample and variance for each point (wrt the 3 samples)

```
avg=mean(A,3); %mean value of z for each (x,y) point
avg(avg<0)=0; %deleat negative values
surf(avg) %already corrected
```



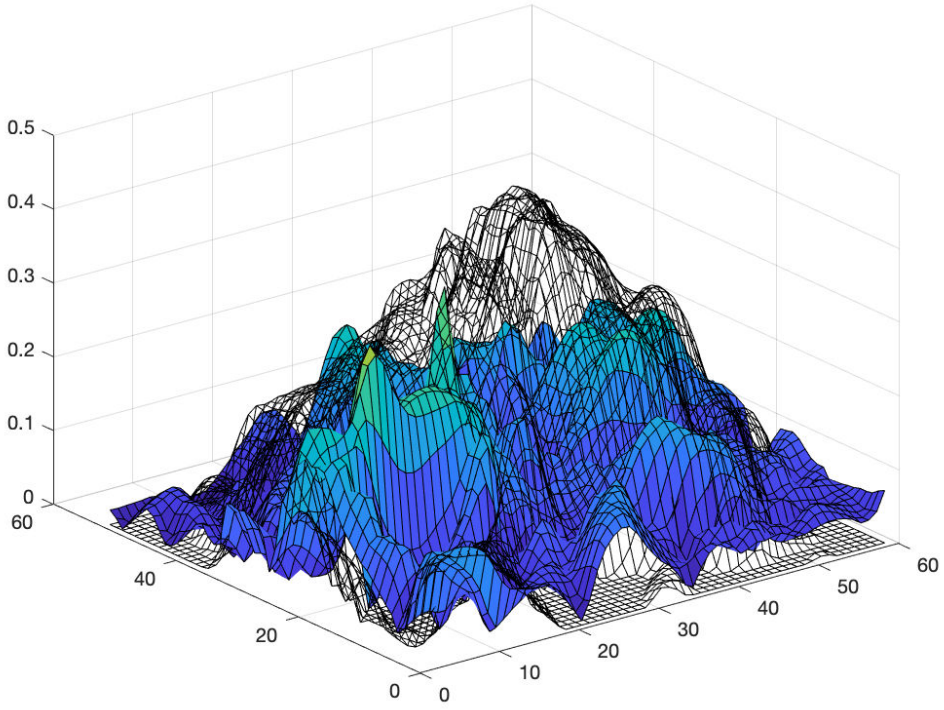

```
deviation = std(A,0,3);
mean(deviation(:))
```

```
ans = 0.0978
```

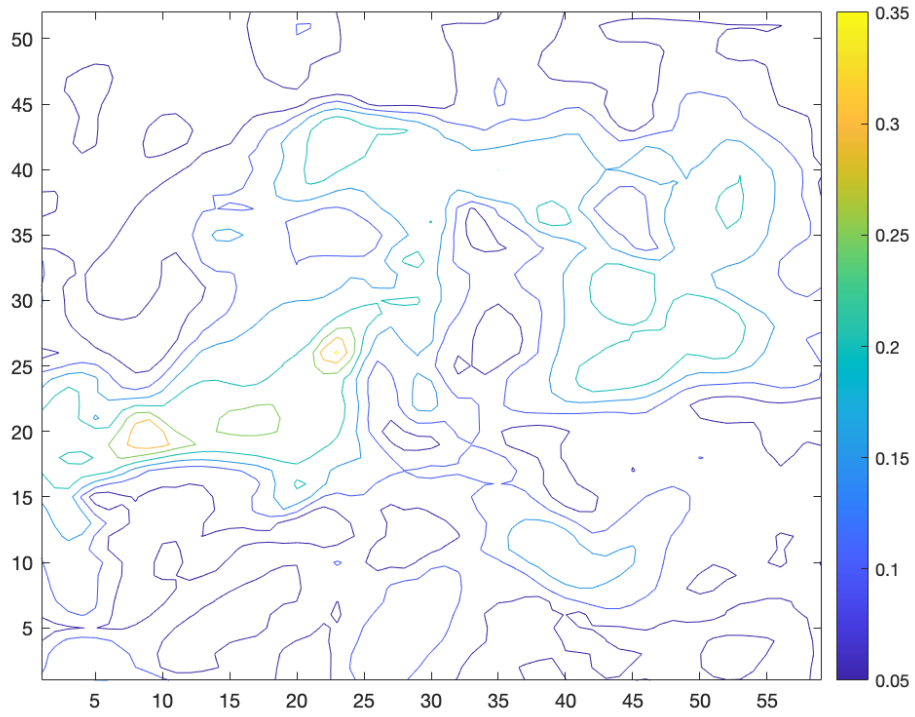
```
variance=var(A,0,3);
mean(variance(:))
```

```
ans = 0.0135
```

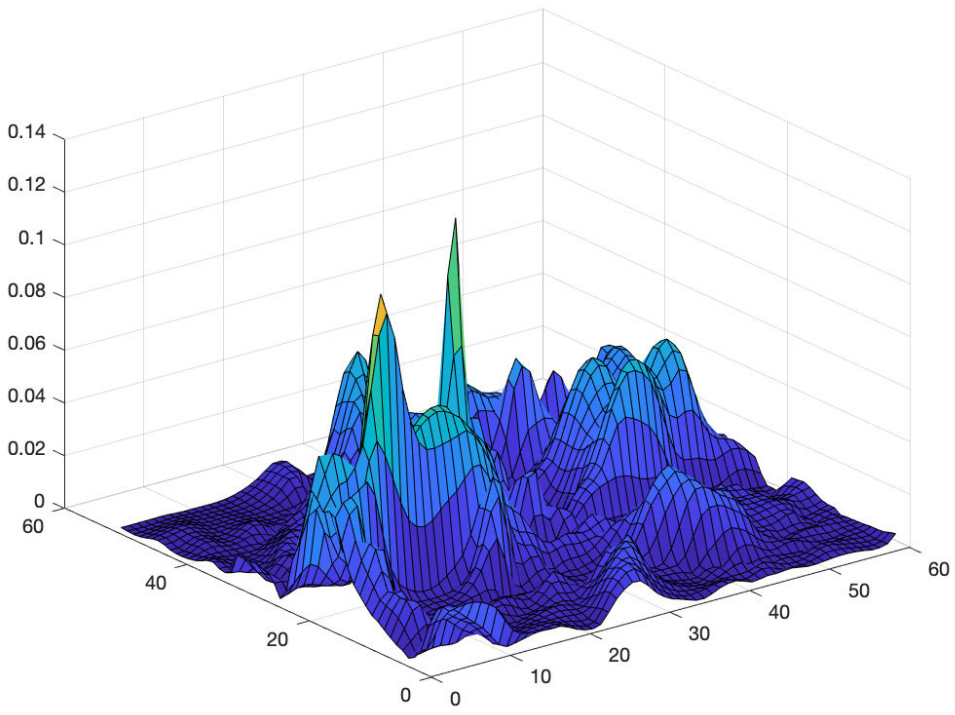
```
figure(1)
surf(avg,"FaceColor","none")
hold on
surf(deviation)
hold off
```




```
figure()
contour(deviation)
colorbar
```




```
surf(variance)
```



Approximating to a function

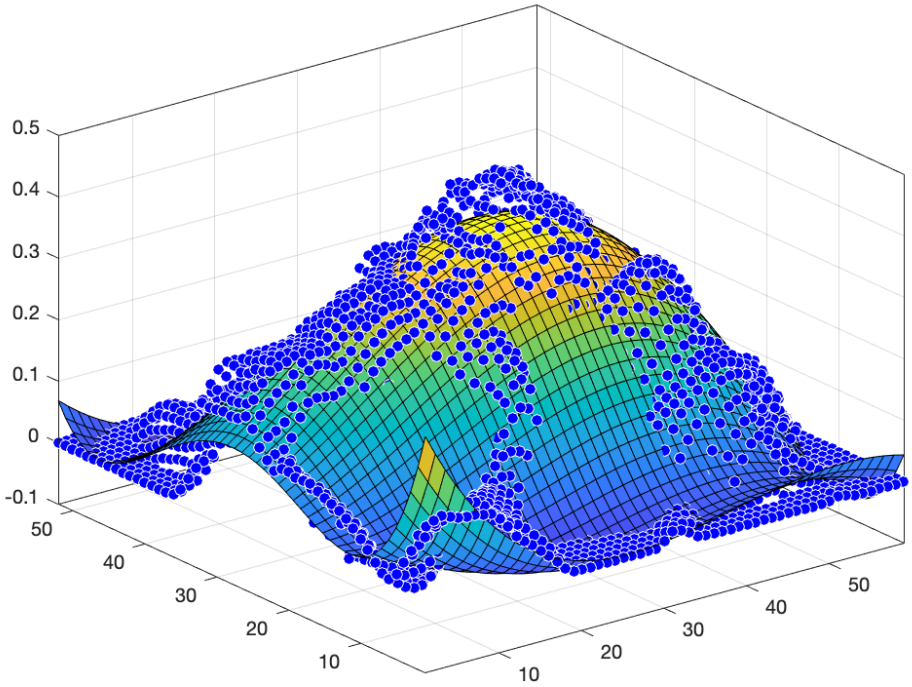
```
[X,Y] = meshgrid(1:y1,1:x1);
%figure(2)
%surf(X,Y,avg) --> used to ensure that the meshgrid was correctly done
%figure(3)
%contour(avg,15)
%colorbar
x2=[];
y2=[];
z2=[];
for i=1:size(X,1)
    for j=1:size(X,2)
        x2=[x2;X(i,j)];
        y2=[y2;Y(i,j)];
        z2=[z2;avg(i,j)];
    end
end

sf = fit([x2, y2],z2,'poly44','Robust','Bisquare' )
```

```
     Linear model Poly44:
     sf(x,y) = p00 + p10*x + p01*y + p20*x^2 + p11*x*y + p02*y^2 + p30*x^3 +
                   p21*x^2*y + p12*x*y^2 + p03*y^3 + p40*x^4 + p31*x^3*y
                   + p22*x^2*y^2 + p13*x*y^3 + p04*y^4
     Coefficients (with 95% confidence bounds):
       p00 =        0.38  (0.3559, 0.4041)
       p10 =    -0.03333  (-0.0364, -0.03026)
       p01 =    -0.07056  (-0.07406, -0.06705)
       p20 =    0.001014  (0.0008491, 0.001179)
       p11 =    0.003175  (0.003029, 0.003321)
       p02 =    0.004919  (0.004705, 0.005132)
       p30 =  -1.093e-05  (-1.468e-05, -7.168e-06)
       p21 =   -5.08e-05  (-5.406e-05, -4.754e-05)
       p12 =  -6.527e-05  (-6.898e-05, -6.157e-05)
       p03 =  -0.0001296  (-0.0001351, -0.0001241)
       p40 =   3.156e-08  (1.193e-09, 6.193e-08)
       p31 =   9.041e-08  (6.039e-08, 1.204e-07)
       p22 =   7.851e-07  (7.517e-07, 8.185e-07)
       p13 =   2.254e-07  (1.867e-07, 2.64e-07)
       p04 =   1.134e-06  (1.083e-06, 1.184e-06)
```

```
options = fitoptions(sf)
```

```
options =

    Normalize: 'off'
      Exclude: []
      Weights: []
       Method: 'LinearLeastSquares'
       Robust: 'Bisquare'
        Lower: [1×0 double]
        Upper: [1×0 double]
```

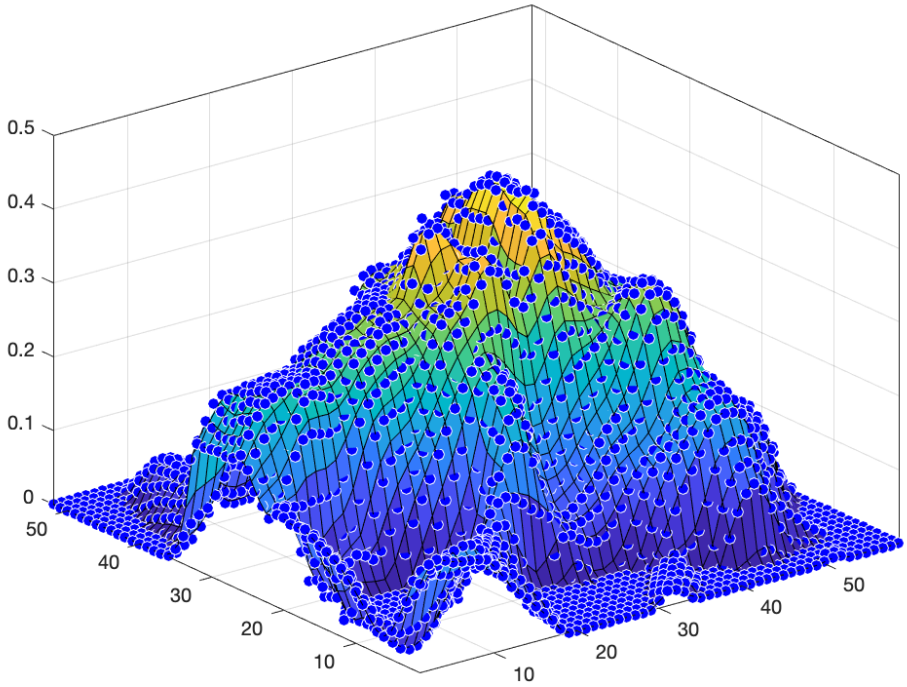
```
figure()
plot(sf,[x2,y2],z2)
```




```
sf1 = fit([x2, y2],z2,'linearinterp')
```

```
Linear interpolant:
  sf1(x,y) = piecewise linear surface computed from p
Coefficients:
  p = coefficient structure
```

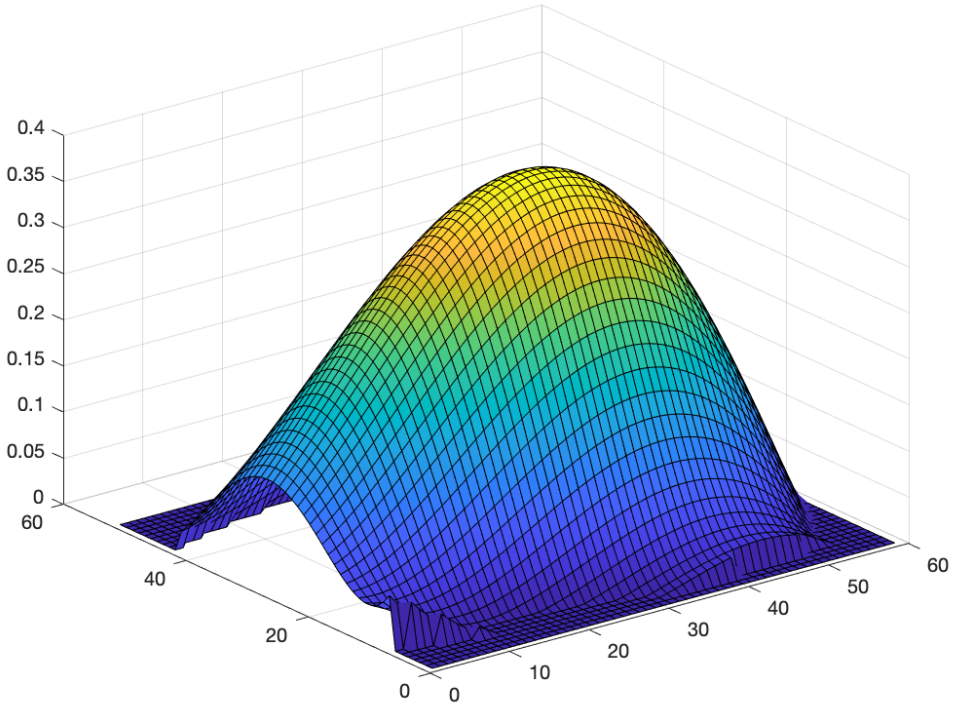
```
figure()
plot(sf1,[x2,y2],z2)
```




```
%z3=zeros(63,71);
adj=zeros(x1,y1);
adj1=adj; %a copy
```

We correct the negative values and those on the extreme that go again up

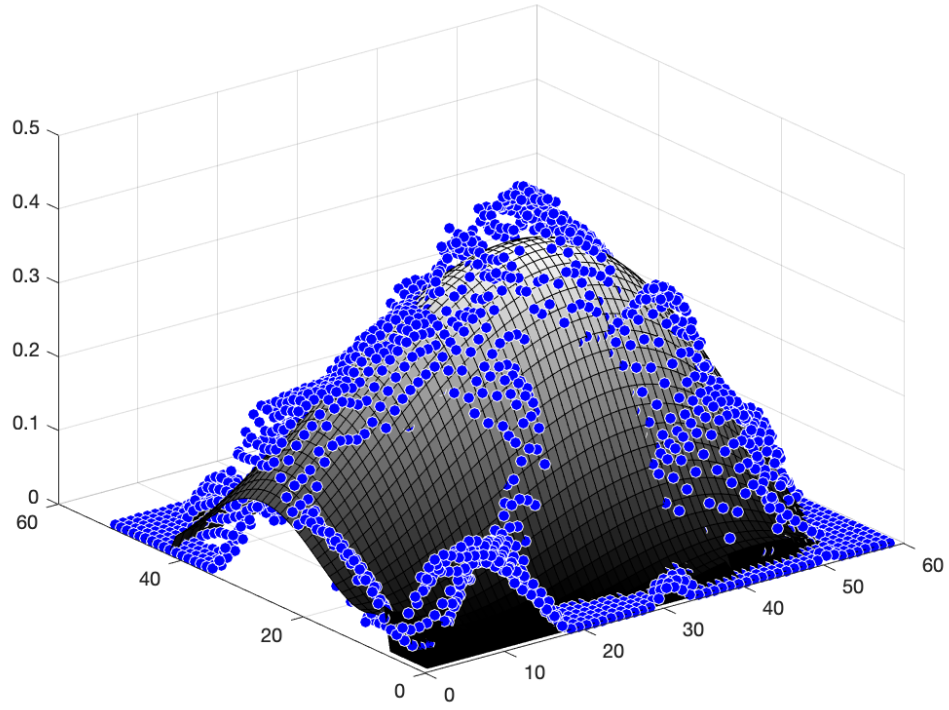
```
for i=1:length(x2)
    if sf(x2(i),y2(i)) > 0
        adj(y2(i),x2(i))= sf(x2(i),y2(i));
    else
        adj(y2(i),x2(i))= 0;
    end
    if (x2(i)-32)^2/2500+(y2(i)-25)^2/500 > 1
        adj(y2(i),x2(i))= 0;
    end
end
surf (adj)
```



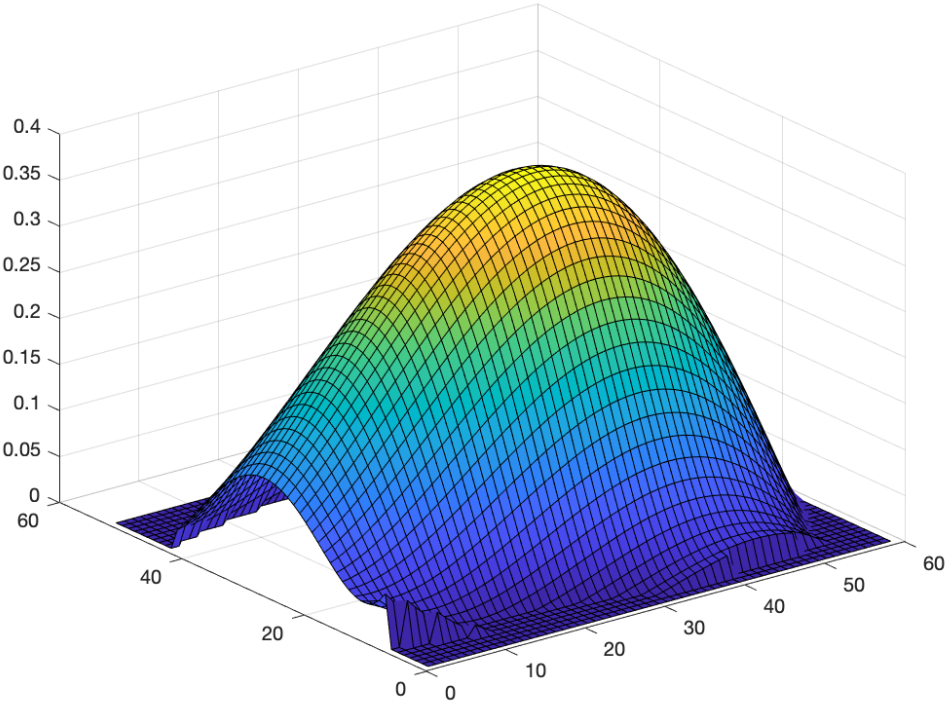

```
%(z3==avg) it is the same
```

Corrected values: adj and z2

```
surf(adj)
colormap("gray")
hold on
plot3(x2,y2,z2,'ow','MarkerSize',6,'MarkerFaceColor','blue')
hold off
```




```
surf (adj)
colormap("default")
```



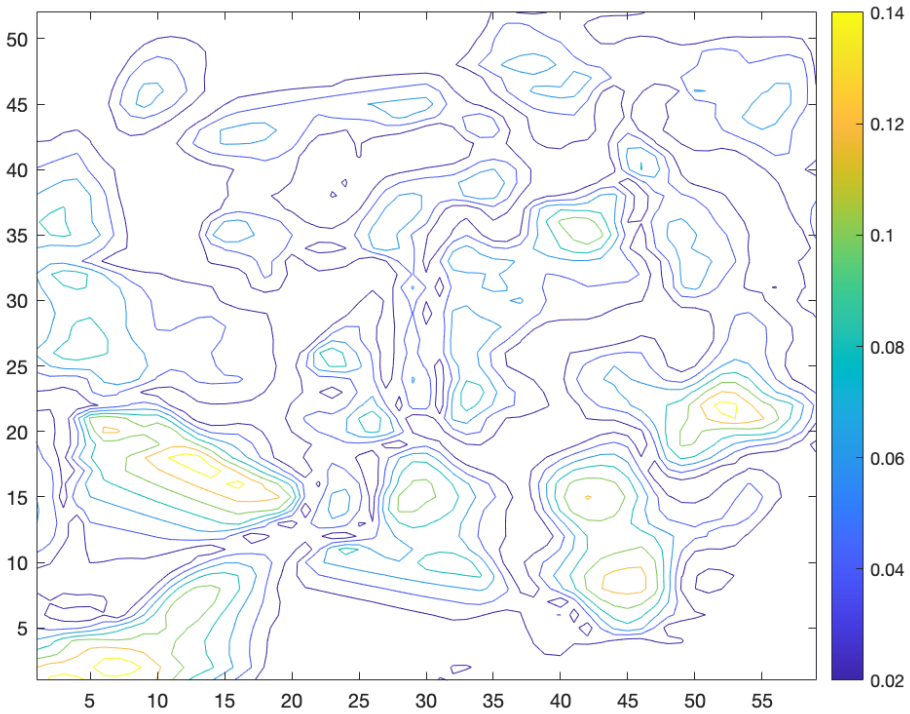
Errors

Absolute and relative errors

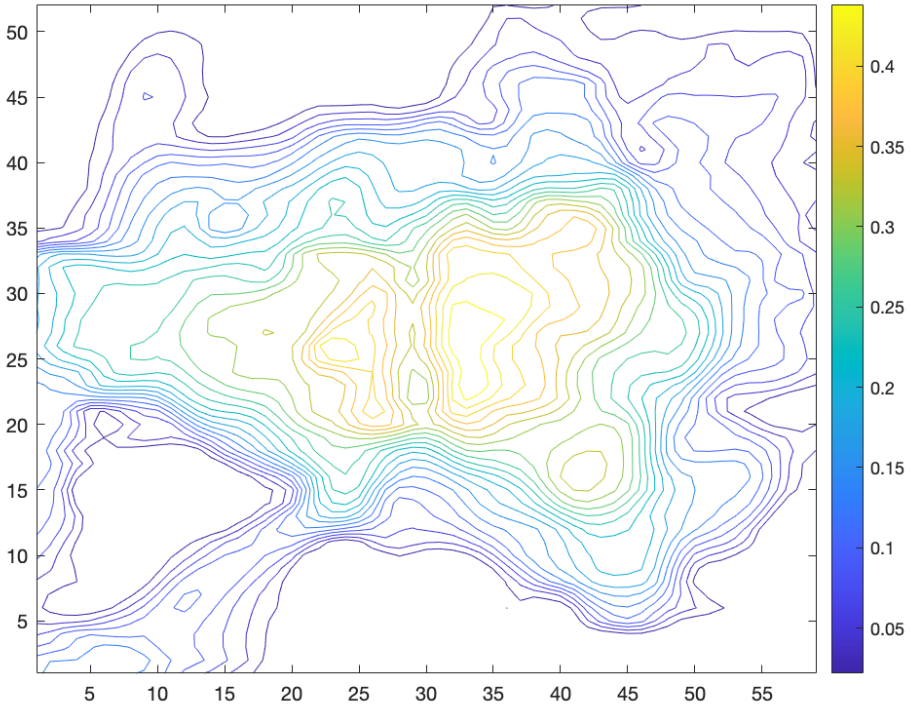
```
AbsErr=abs(adj-avg);
RelErr=AbsErr./avg*100;
mean(AbsErr(:))
```

```
ans = 0.0365
```

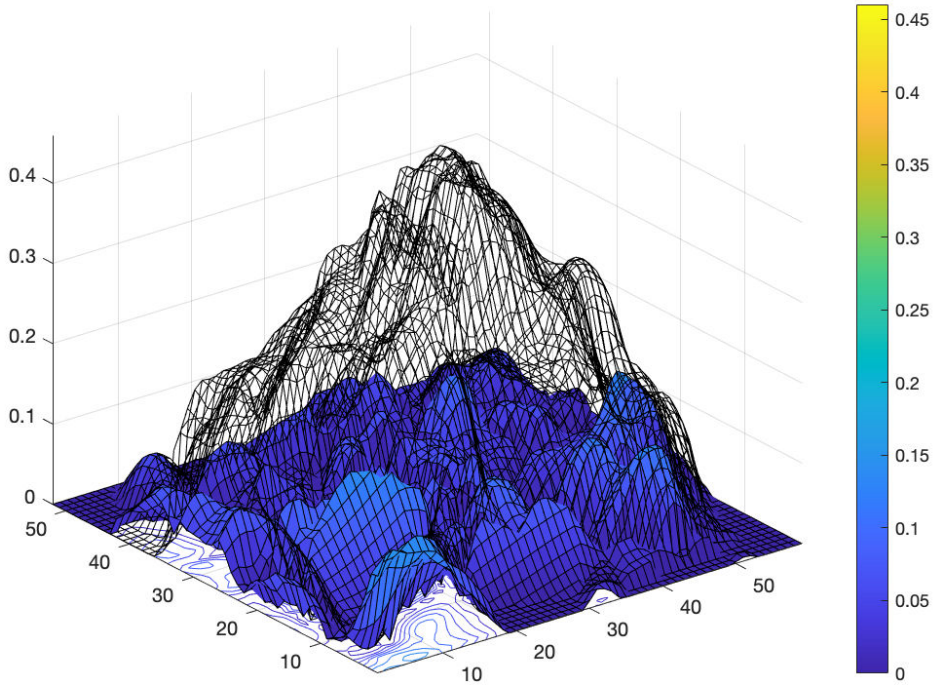
```
figure()
contour(AbsErr)
colorbar
```



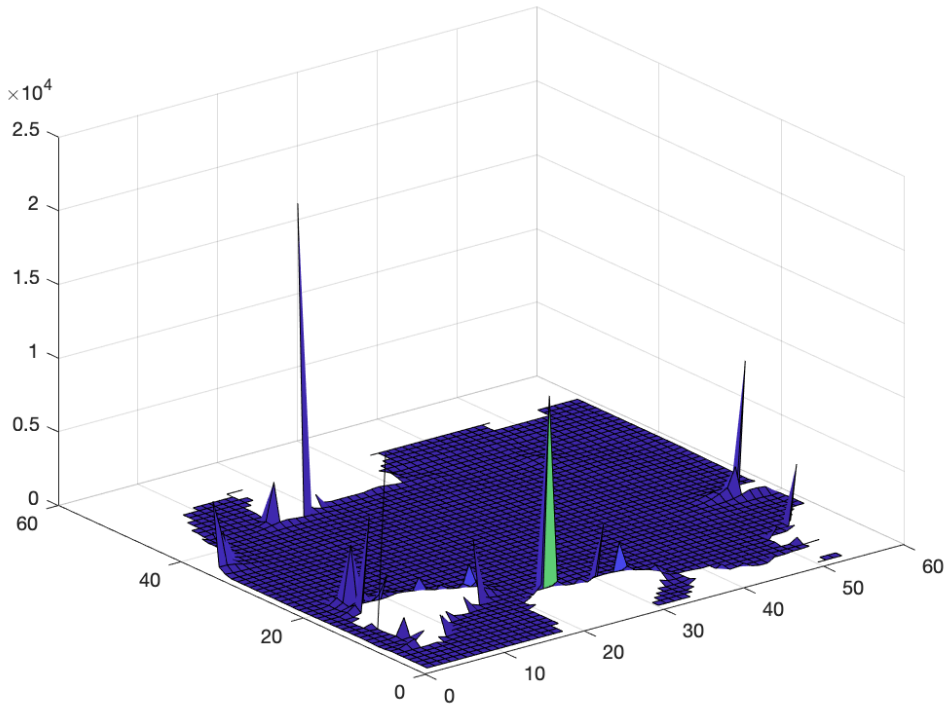

```
contour(avg,20)
colorbar
```




```
figure()
surfc(AbsErr)
hold on
surf(avg,'FaceColor',"none")
colorbar
hold off
```




```
figure()
surf(RelErr)
```



```
ans = NaN
```

Computation of RMSE

```
s=[]; %predicted data corrected
for i=1:size(X,1)
    for j=1:size(X,2)
        s=[s;adj(i,j)];
    end
end
s(s<0)=0;

z2=z2; %exact data
n=length(s);
MSE=1/n*sum((z2-s).^2) %mean squared error
```

```
MSE = 0.0023
```

```
Rsquared=1-MSE/var(z2)
```

```
Rsquared = 0.8523
```

```
R=sqrt(Rsquared)
```

```
R = 0.9232
```

Maximum height

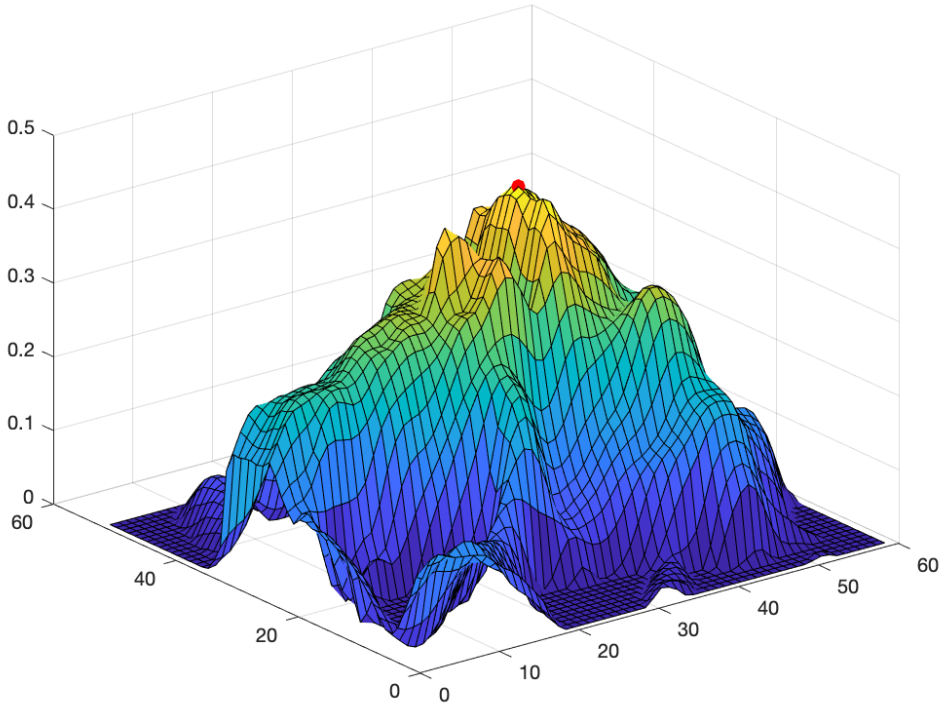
```
maxheight=max(avg,[],'all')
```

```
maxheight = 0.4597
```

```
[rowsOfMaxes colsOfMaxes] = find(avg == maxheight)
```

```
rowsOfMaxes = 27
colsOfMaxes = 33
```

```
surf(avg)
hold on
plot3(colsOfMaxes,rowsOfMaxes,avg(rowsOfMaxes,colsOfMaxes),'.r',"MarkerSize",25)
hold off
```



Volume of the surface

```
[TriIdx, Vol] =convhull(X,Y,avg); %exact data

Vol %real volume
```

```
Vol = 676.0391
```

```
s=sf(x2,y2);
s(s<0)=0;
[TriIdx, Vol1] =convhull(x2,y2,s);
Vol1
```

```
Vol1 = 684.2708
```

```
AbsVol=abs(Vol1-Vol)
```

```
AbsVol = 8.2317
```

```
RelVol=AbsVol/Vol*100 %9.6190 (without robust)
```

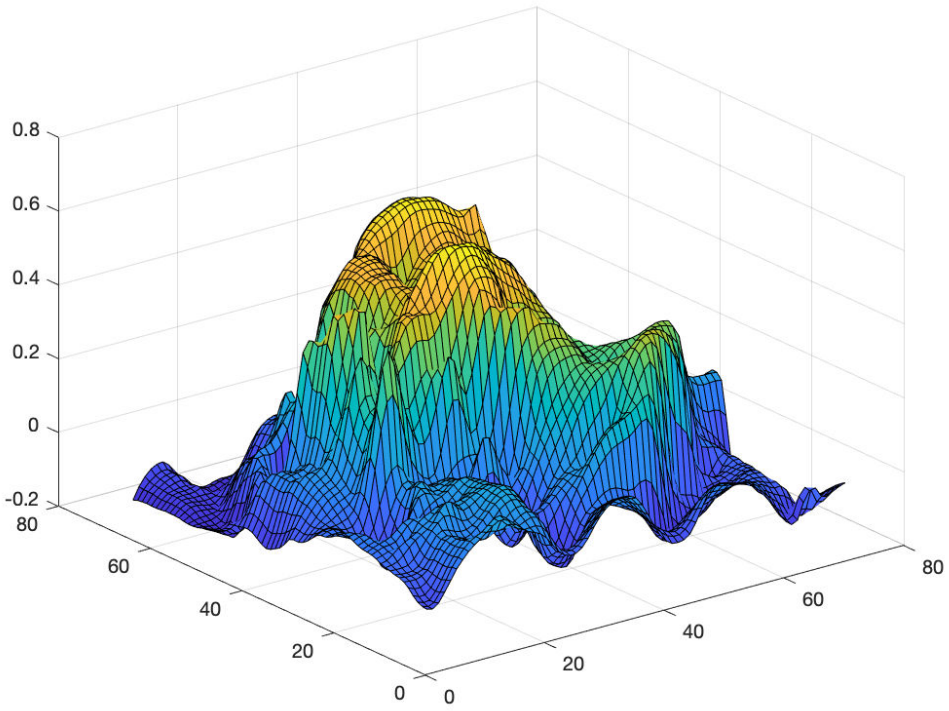
```
RelVol = 1.2176
```

```


H_bar=Vol/(x1*y1)
```

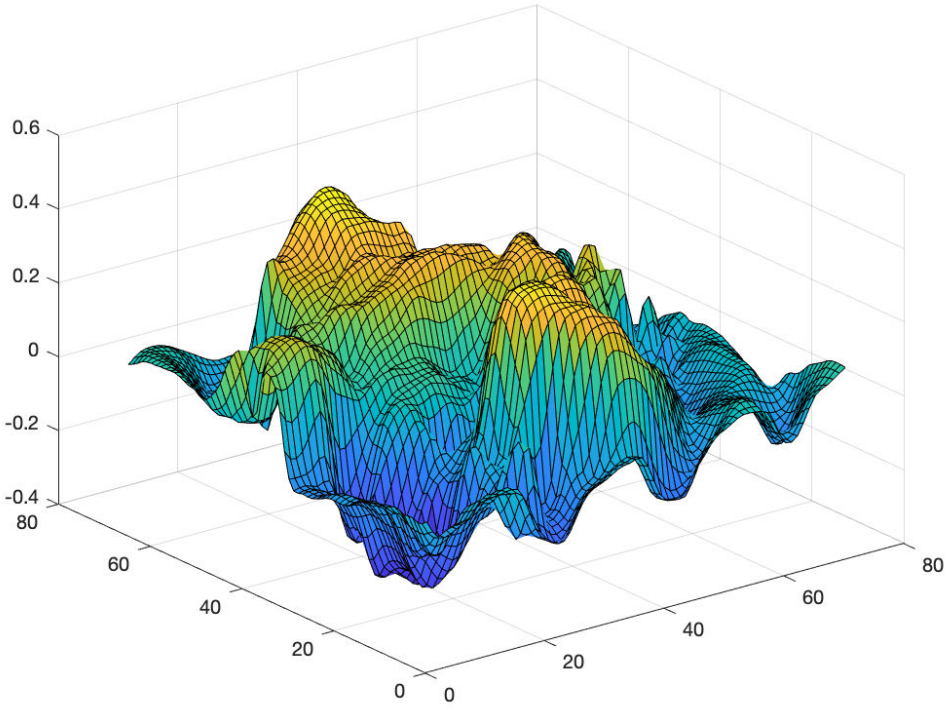
```
H_bar = 0.2204
z_lim = 0.3400
UD = 0.0815
```

```
%save('adj_s4.mat',"adj")
%save('avg_s4.mat','avg')
%save('var_s4.mat','deviation')
```

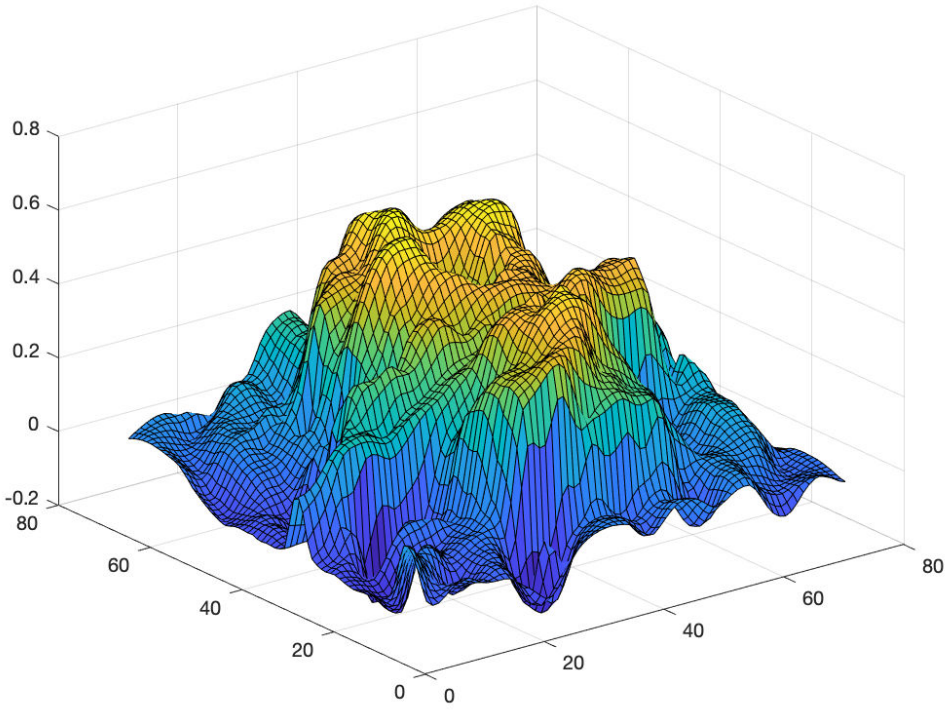
```
clear all
load('samples.mat')
x1=66;
y1=71;
SAMPLE5_2=SAMPLE5_2(12:end-11,9:end-10);
SAMPLE5_3=SAMPLE5_3(10:end-11,9:end-9);
S1={SAMPLE5_1,SAMPLE5_2,SAMPLE5_3};
A=zeros(x1,y1,3);
for k=1:3
    s = S1{k};
    s(isnan(s))=0;
    for i=1:x1
        for j=1:y1
            A(i,j,k)=s(i,j); %matrix that contains the 3 samples
        end
    end
end

surf(S1{1})
```




```
surf(S1{2})
```

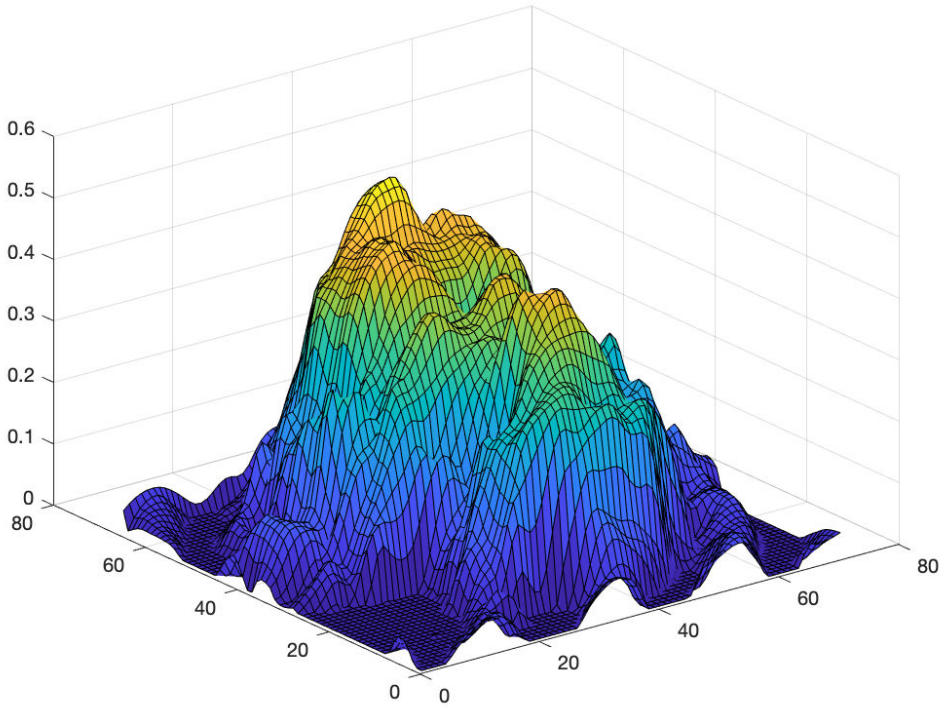

0.6
0.4
0.2
0
-0.2
-0.4
80
60
40
20
0
0
20
40
60
80

```
surf(S1{3})
```


0.8
0.6
0.4
0.2
0
-0.2
80
60
40
20
0
0
20
40
60
80

Mean value of the sample and variance for each point (wrt the 3 samples)

```
avg=mean(A,3); %mean value of z for each (x,y) point
avg(avg<0)=0; %deleat negative values
surf(avg) %already corrected
```



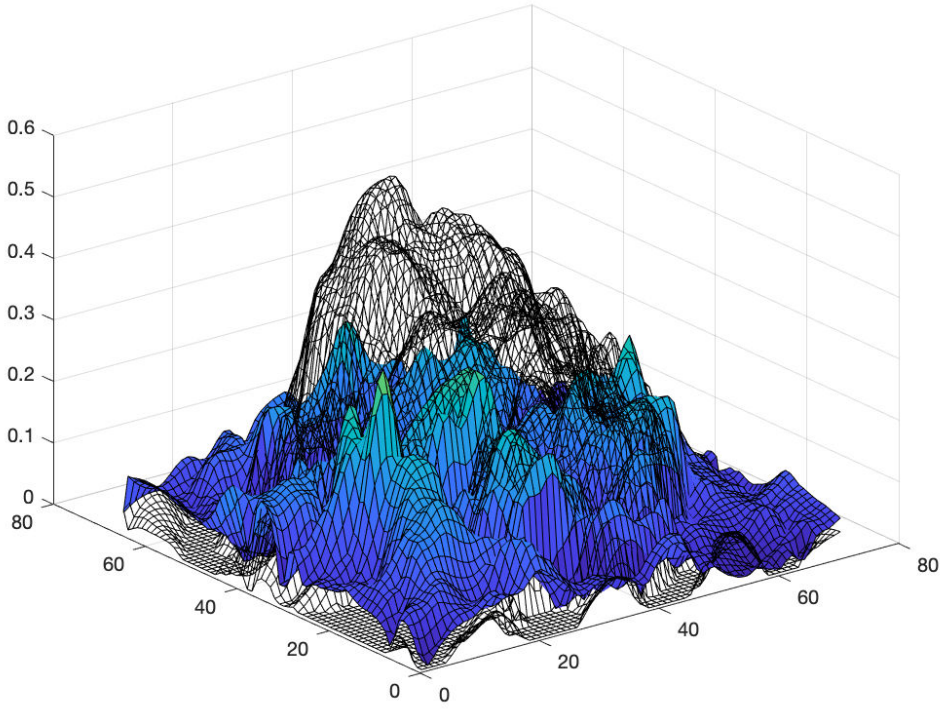

```
deviation = std(A,0,3);
mean(deviation(:))
```

```
ans = 0.1138
```

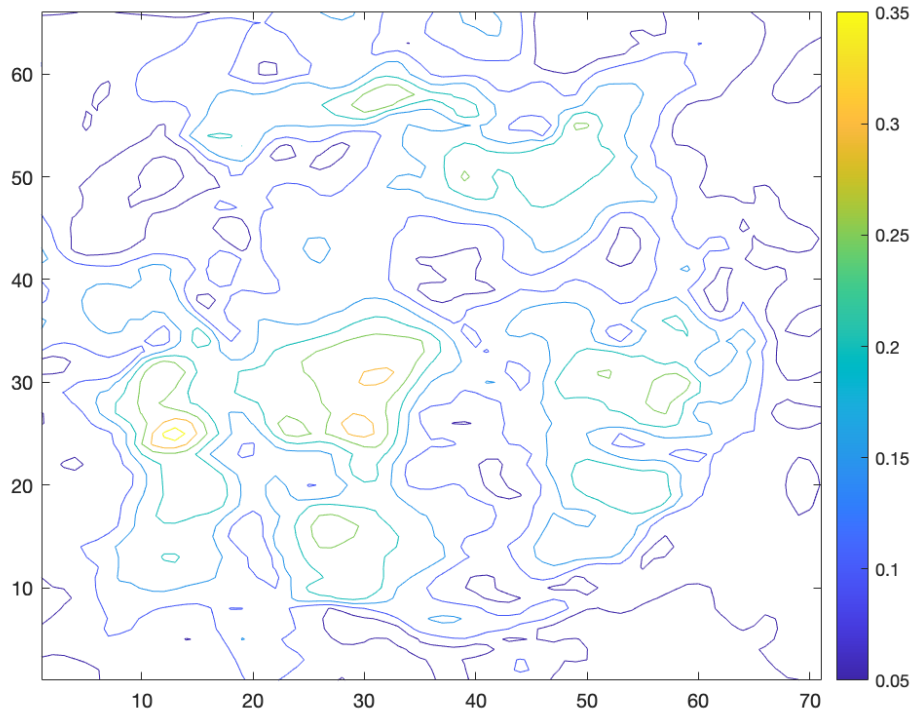
```
variance=var(A,0,3);
mean(variance(:))
```

```
ans = 0.0171
```

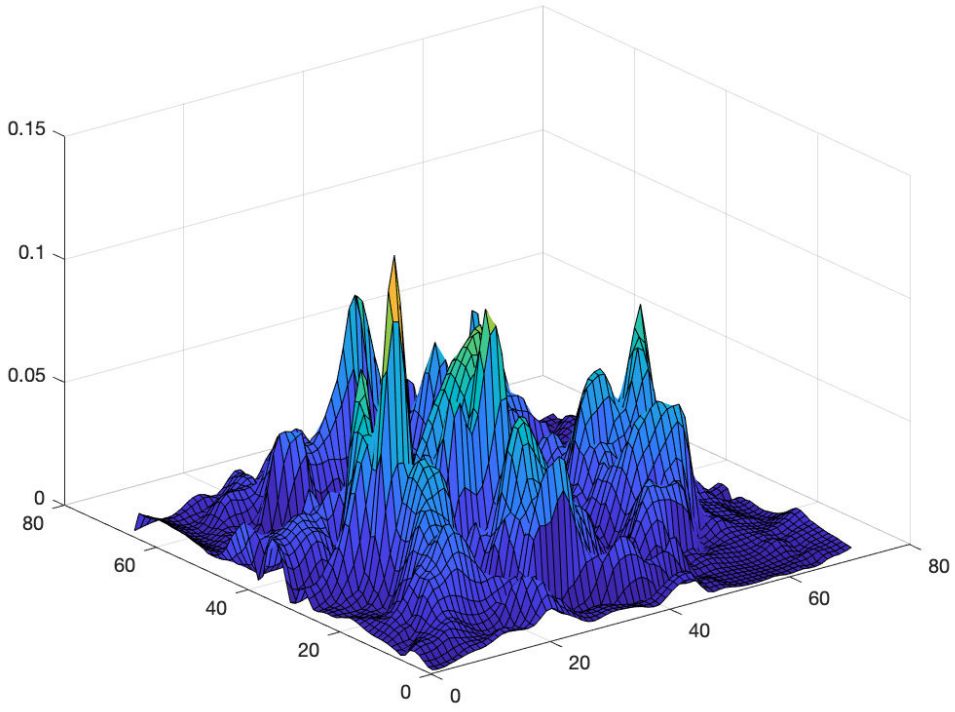
```
figure(1)
surf(avg,"FaceColor","none")
hold on
surf(deviation)
hold off
```




```
figure()
contour(deviation)
colorbar
```




```
surf(variance)
```



Approximating to a function

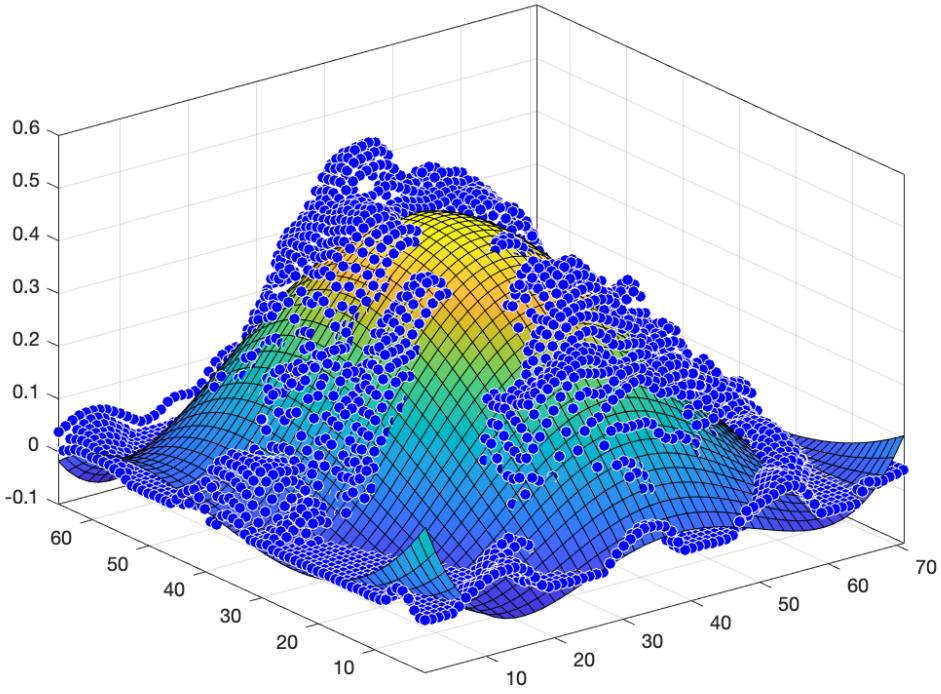
```
[X,Y] = meshgrid(1:y1,1:x1);
%figure(2)
%surf(X,Y,avg) --> used to ensure that the meshgrid was correctly done
%figure(3)
%contour(avg,15)
%colorbar
x2=[];
y2=[];
z2=[];
for i=1:size(X,1)
    for j=1:size(X,2)
        x2=[x2;X(i,j)];
        y2=[y2;Y(i,j)];
        z2=[z2;avg(i,j)];
    end
end

sf = fit([x2, y2],z2,'poly44','Robust','Bisquare' )
```

```
     Linear model Poly44:
     sf(x,y) = p00 + p10*x + p01*y + p20*x^2 + p11*x*y + p02*y^2 + p30*x^3 +
                   p21*x^2*y + p12*x*y^2 + p03*y^3 + p40*x^4 + p31*x^3*y
                   + p22*x^2*y^2 + p13*x*y^3 + p04*y^4
     Coefficients (with 95% confidence bounds):
       p00 =      0.2394  (0.2159, 0.2628)
       p10 =     -0.0448  (-0.0473, -0.0423)
       p01 =    -0.02262  (-0.02531, -0.01992)
       p20 =    0.002348  (0.002236, 0.00246)
       p11 =    0.002243  (0.002149, 0.002337)
       p02 =   0.0006166  (0.0004867, 0.0007464)
       p30 =   -4.66e-05  (-4.873e-05, -4.447e-05)
       p21 =  -3.293e-05  (-3.468e-05, -3.118e-05)
       p12 =   -3.27e-05  (-3.458e-05, -3.082e-05)
       p03 =  -4.335e-06  (-6.986e-06, -1.684e-06)
       p40 =   3.106e-07  (2.963e-07, 3.25e-07)
       p31 =   5.903e-08  (4.56e-08, 7.246e-08)
       p22 =   3.713e-07  (3.571e-07, 3.855e-07)
       p13 =   5.909e-08  (4.354e-08, 7.464e-08)
       p04 =  -9.701e-09  (-2.889e-08, 9.491e-09)
```

```
options = fitoptions(sf)
```

```
options =

    Normalize: 'off'
      Exclude: []
      Weights: []
       Method: 'LinearLeastSquares'
       Robust: 'Bisquare'
        Lower: [1×0 double]
        Upper: [1×0 double]
```

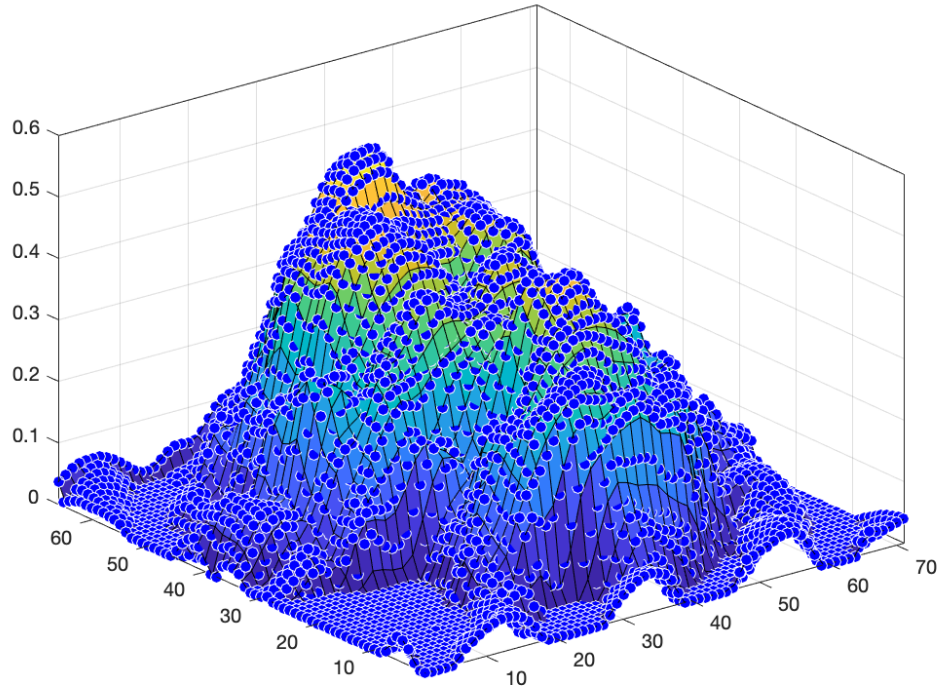
```
figure()
plot(sf,[x2,y2],z2)
```




```
sf1 = fit([x2, y2],z2,'linearinterp')
```

```
Linear interpolant:
  sf1(x,y) = piecewise linear surface computed from p
Coefficients:
  p = coefficient structure
```

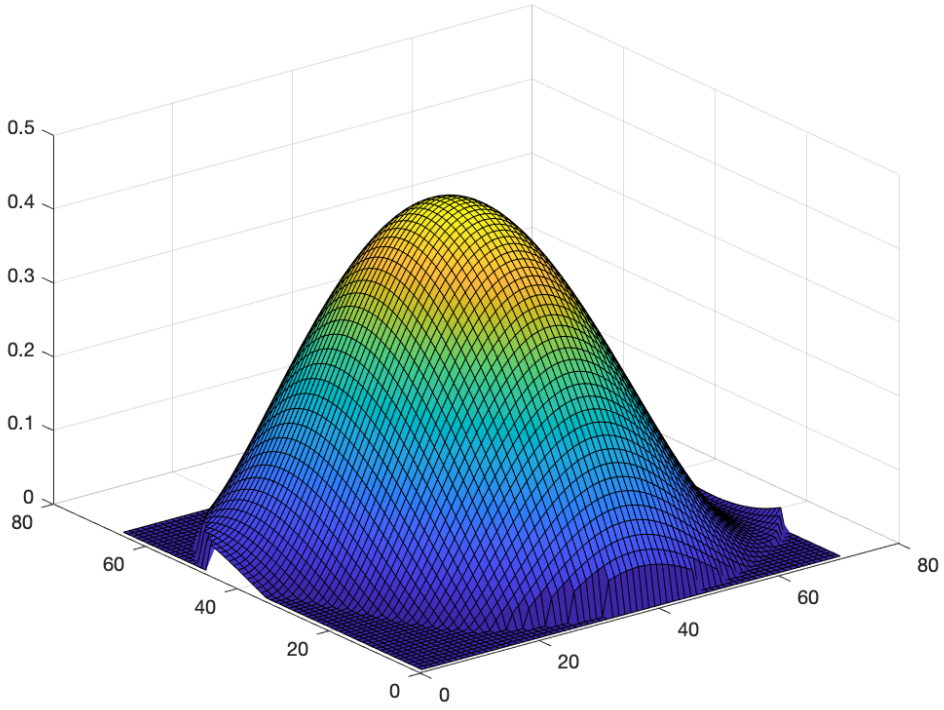
```
figure()
plot(sf1,[x2,y2],z2)
```




```
%z3=zeros(63,71);
adj=zeros(x1,y1);
adj1=adj; %a copy
```

We correct the negative values and those on the extreme that go again up

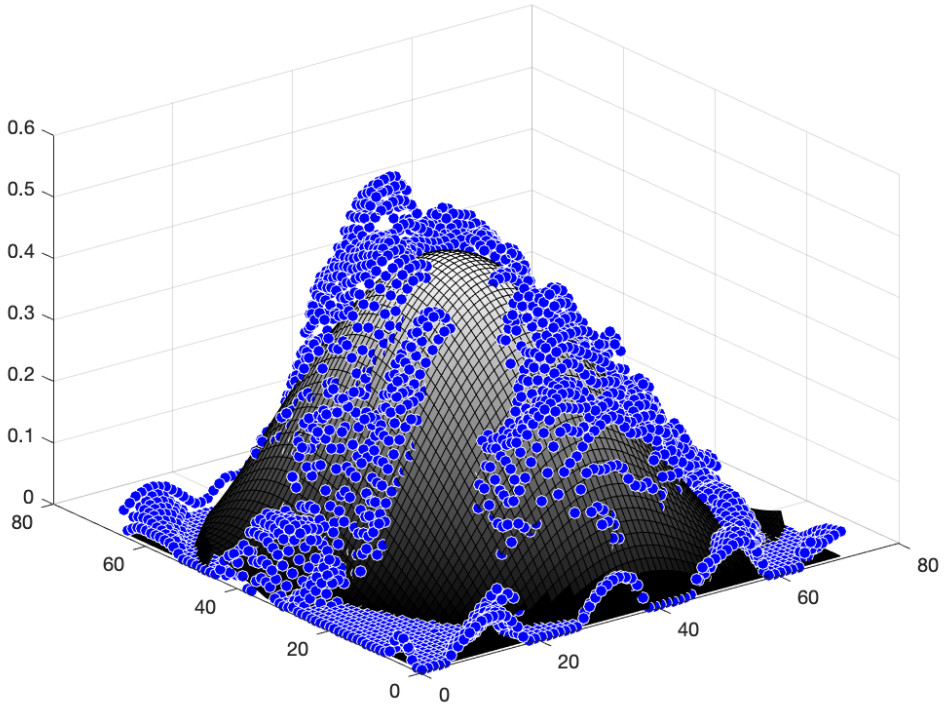
```
for i=1:length(x2)
    %z3(y2(i),x2(i))= z2(i);
    if sf(x2(i),y2(i)) > 0
        adj(y2(i),x2(i))= sf(x2(i),y2(i));
    else
        adj(y2(i),x2(i))= 0;
    end
    if (x2(i)-40)^2/1700+(y2(i)-36)^2/1200 > 1
        adj(y2(i),x2(i))= 0;
    end
    %if y2(i) < 60-3*x2(i)
        %adj(y2(i),x2(i))= 0.05;
    %elseif y2(i) > 259/3-10/21*x2(i)
        %adj(y2(i),x2(i))= 0;
    %elseif y2(i) < -39+5/7*x2(i)
        %adj(y2(i),x2(i))= 0;

    %end
end
surf (adj)
```



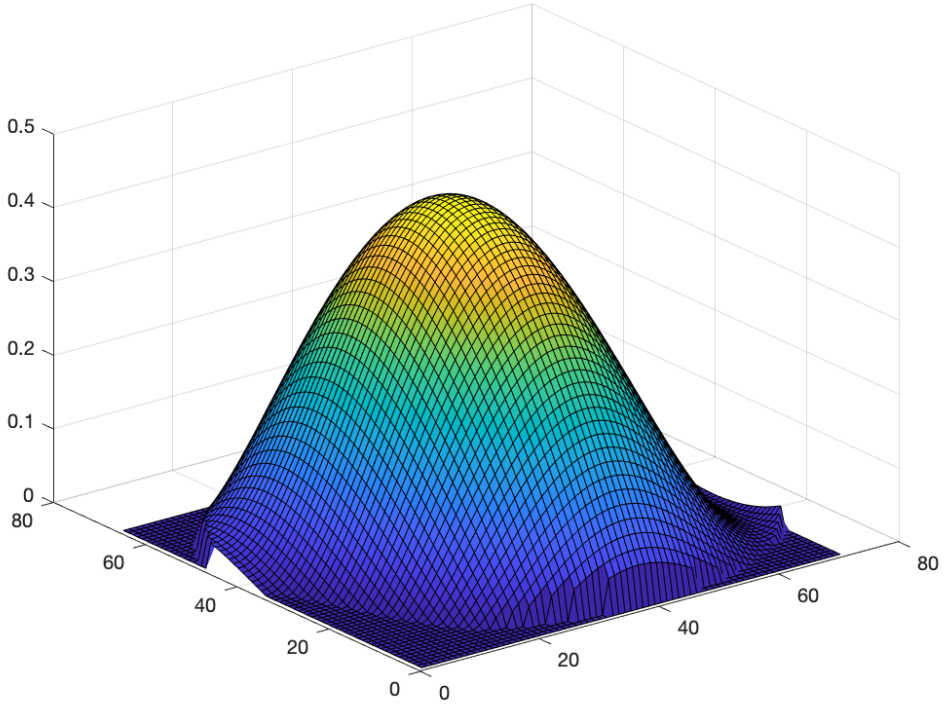

```
%(z3==avg) it is the same
```

Corrected values: adj and z2

```
surf(adj)
colormap("gray")
hold on
plot3(x2,y2,z2,'ow','MarkerSize',6,'MarkerFaceColor','blue')
hold off
```




```
surf (adj)
colormap("default")
```



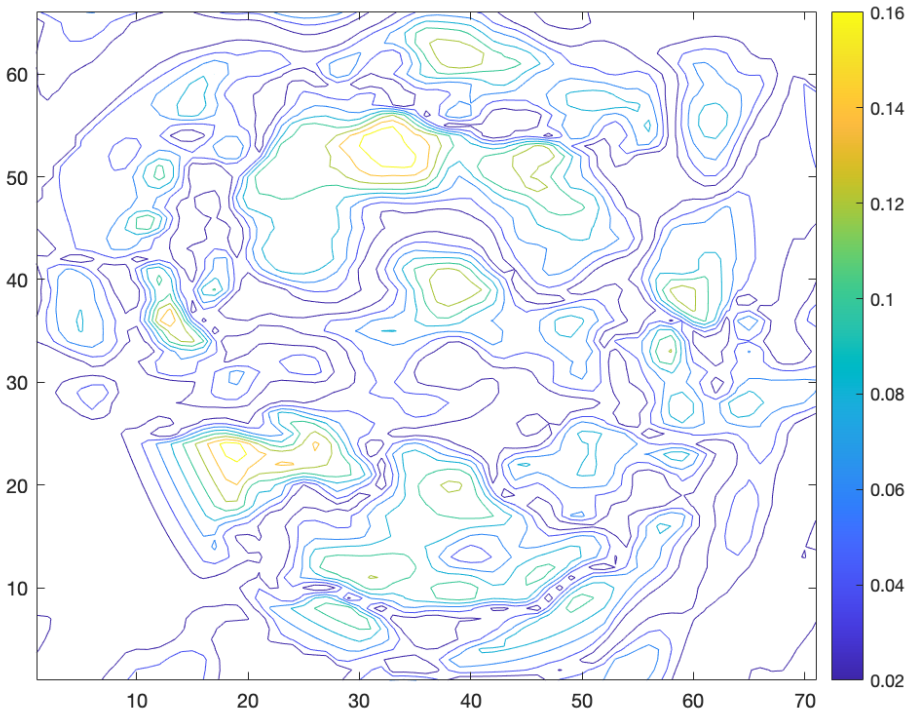
Errors

Absolute and relative errors

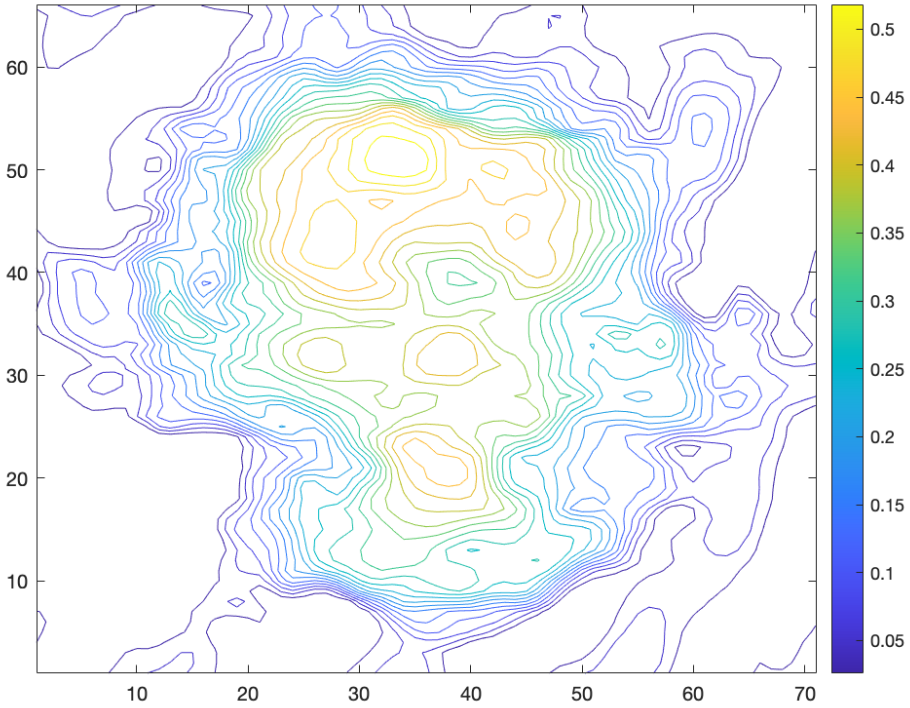
```
AbsErr=abs(adj-avg);
RelErr=AbsErr./avg*100;
mean(AbsErr(:))
```

```
ans = 0.0432
```

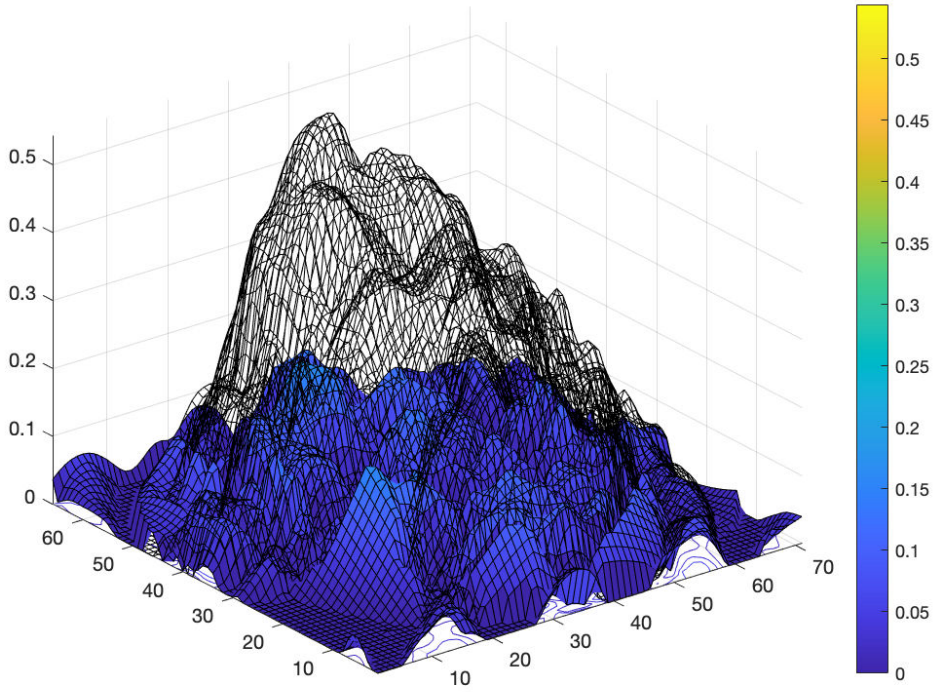
```
figure()
contour(AbsErr)
colorbar
```



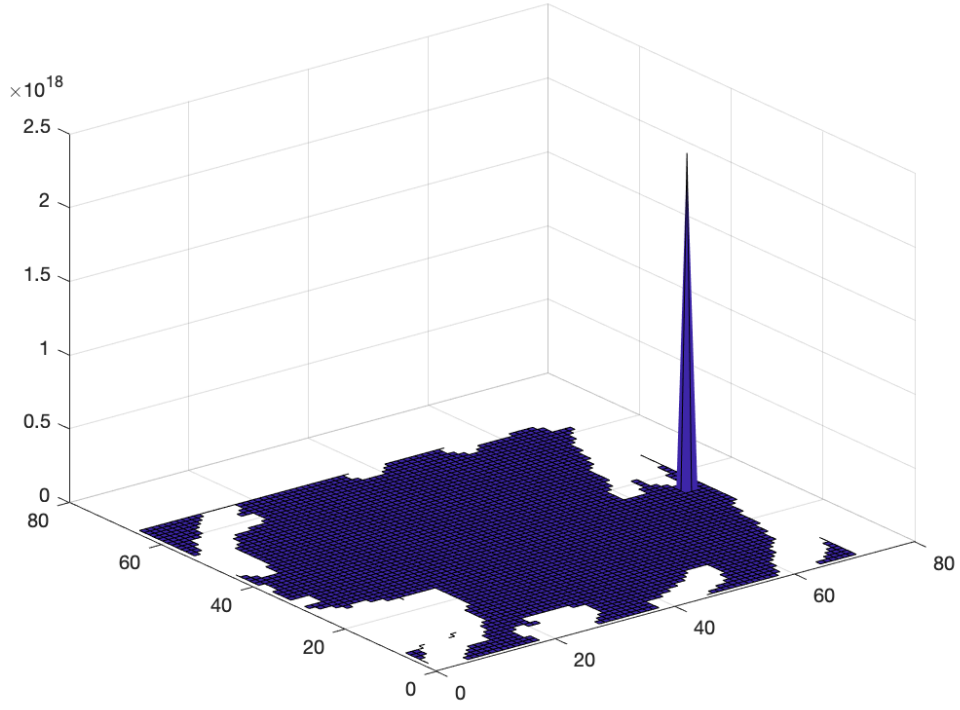

```
contour(avg,20)
colorbar
```




```
figure()
surfc(AbsErr)
hold on
surf(avg,'FaceColor',"none")
colorbar
hold off
```




```
figure()
surf(RelErr)
```



```
ans = NaN
```

Computation of RMSE

```
s=[]; %predicted data corrected
for i=1:size(X,1)
    for j=1:size(X,2)
        s=[s;adj(i,j)];
    end
end
s(s<0)=0;

z2=z2; %exact data
n=length(s);
MSE=1/n*sum((z2-s).^2) %mean squared error
```

```
MSE = 0.0031
```

```
Rsquared=1-MSE/var(z2)
```

```
Rsquared = 0.8686
```

```
R=sqrt(Rsquared)
```

```
R = 0.9320
```

Maximum height

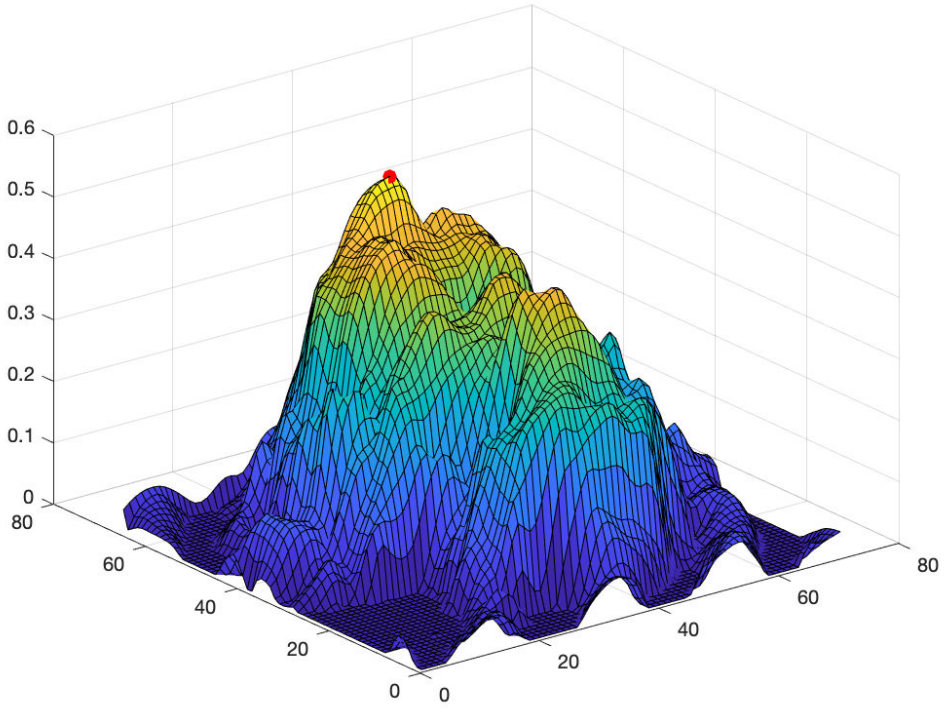
```
maxheight=max(avg,[],'all')
```

```
maxheight = 0.5433
```

```
[rowsOfMaxes colsOfMaxes] = find(avg == maxheight)
```

```
rowsOfMaxes = 51
colsOfMaxes = 34
```

```
surf(avg)
hold on
plot3(colsOfMaxes,rowsOfMaxes,avg(rowsOfMaxes,colsOfMaxes),'.r',"MarkerSize",25)
hold off
```



Volume of the surface

```
[TriIdx, Vol] =convhull(X,Y,avg); %exact data

Vol %real volume
```

```
Vol = 1.1979e+03
```

```
s=sf(x2,y2);
s(s<0)=0;
[TriIdx, Vol1] =convhull(x2,y2,s);
Vol1
```

```
Vol1 = 1.0892e+03
```

```
AbsVol=abs(Vol1-Vol)
```

```
AbsVol = 108.6615
```

```
RelVol=AbsVol/Vol*100 %9.6190 (without robust)
```

```
RelVol = 9.0713
```

```


H_bar=Vol/(x1*y1)
```

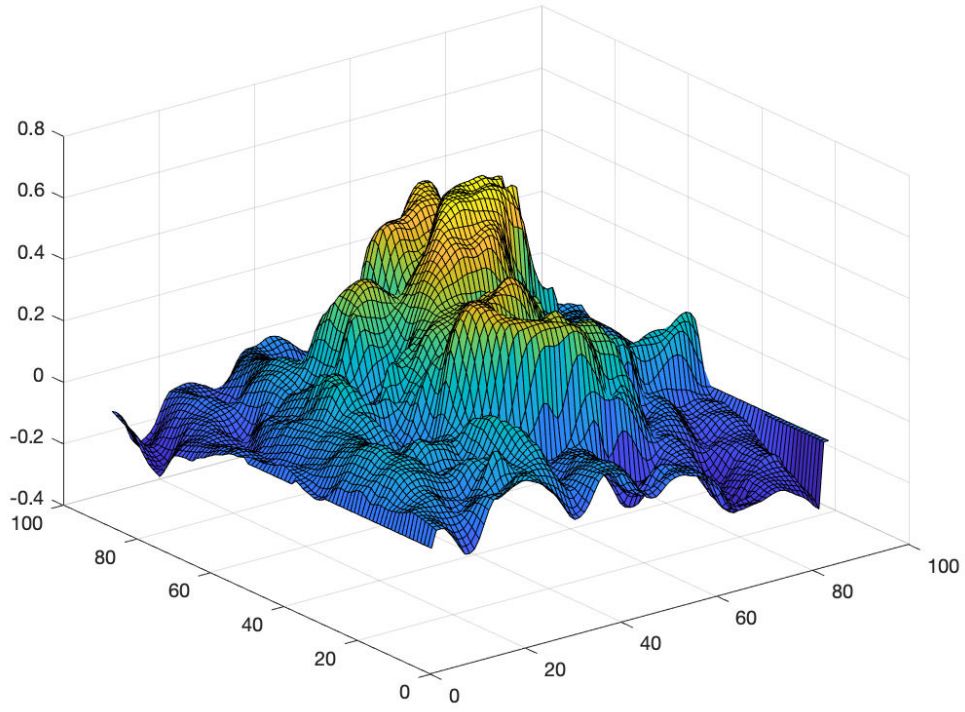
```
H_bar = 0.2556
```

```

```

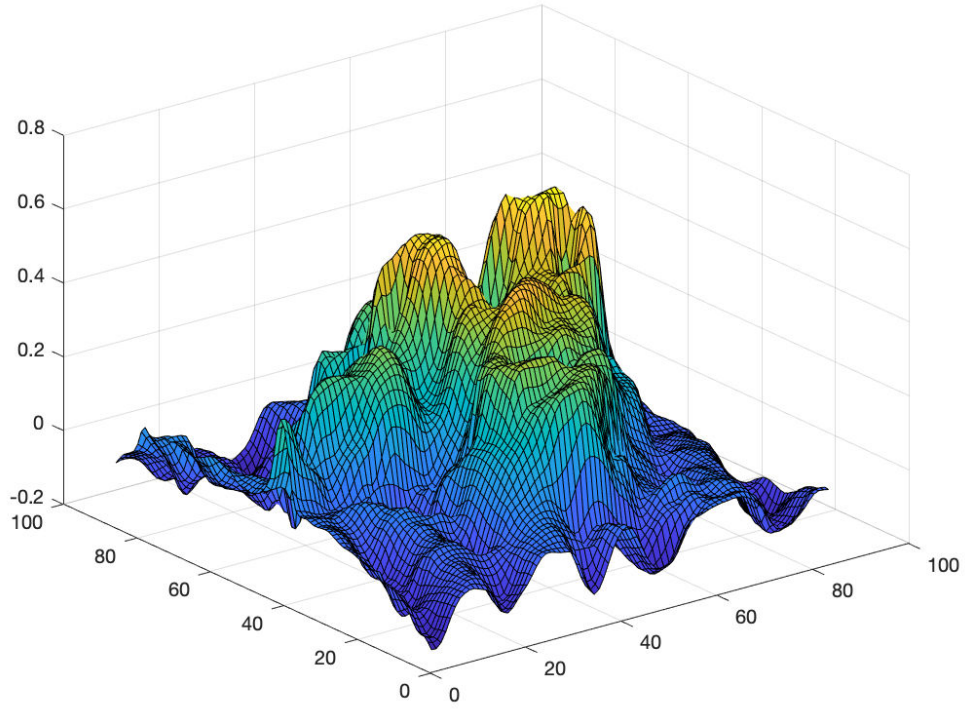
```
z_lim = 0.3995
UD = 0.0962
```

```
%save('adj_s5.mat',"adj")
%save('avg_s5.mat','avg')
%save('var_s5.mat','deviation')
```

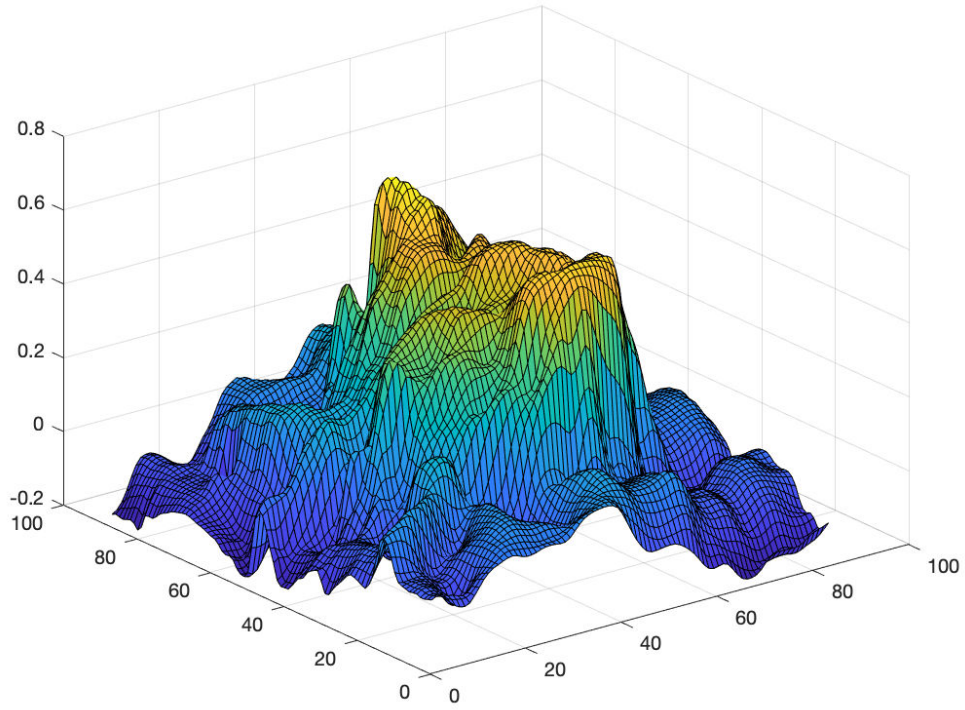
```
clear all
load('samples.mat')
x1=88;
y1=84;
SAMPLE6_1=SAMPLE6_1(2:end-2,:);
SAMPLE6_2=SAMPLE6_2(:,4:end-5);
SAMPLE6_3=SAMPLE6_3(2:end-2,8:end-7);

S1={SAMPLE6_1,SAMPLE6_2,SAMPLE6_3};
A=zeros(x1,y1,3);
for k=1:3
    s = S1{k};
    s(isnan(s))=0;
    for i=1:x1
        for j=1:y1
            A(i,j,k)=s(i,j); %matrix that contains the 3 samples
        end
    end
end

surf(S1{1})
```




```
surf(S1{2})
```

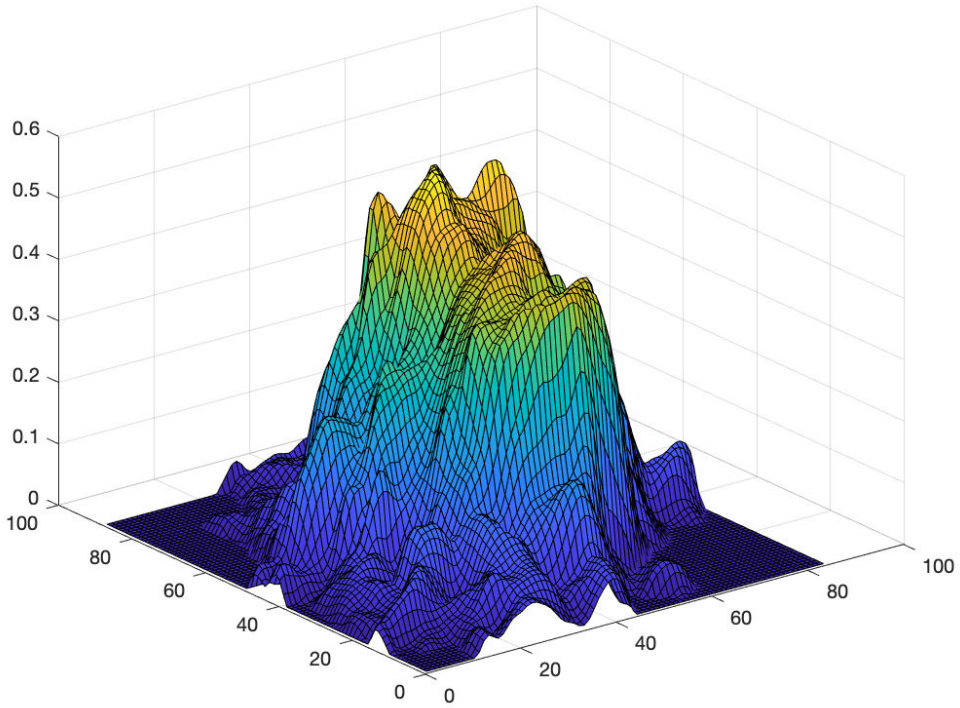

0.8
0.6
0.4
0.2
0
-0.2
100
80
60
40
20
0
0
20
40
60
80
100

```
surf(S1{3})
```


0.8
0.6
0.4
0.2
0
-0.2
100
80
60
40
20
0
0
20
40
60
80
100

Mean value of the sample and variance for each point (wrt the 3 samples)

```
avg=mean(A,3); %mean value of z for each (x,y) point
avg(avg<0)=0; %deleat negative values
surf(avg) %already corrected
```



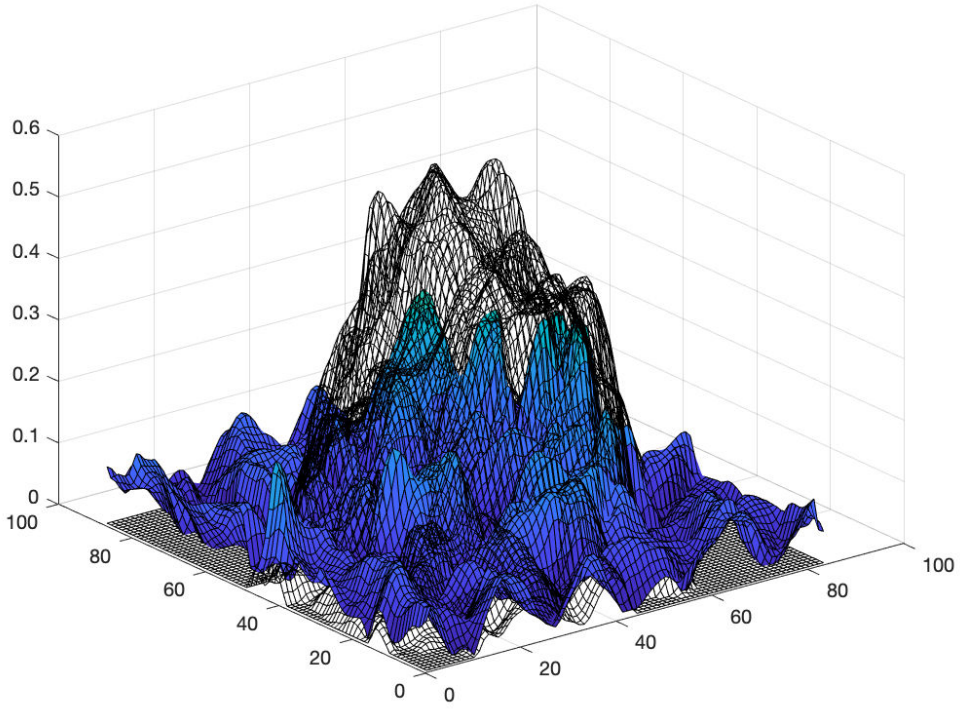

```
deviation = std(A,0,3);
mean(deviation(:))
```

```
ans = 0.0821
```

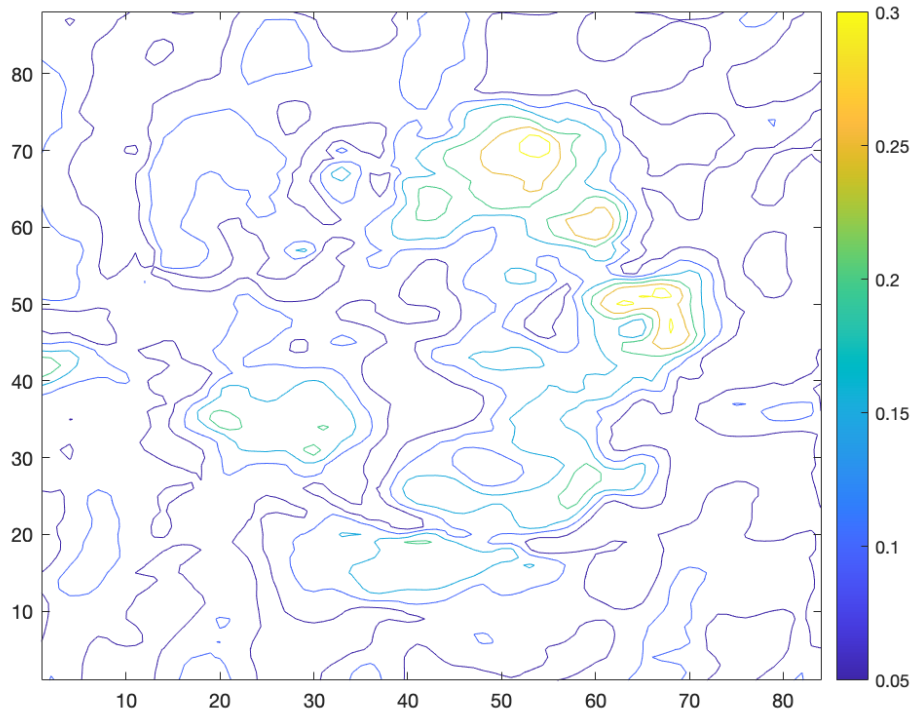
```
variance=var(A,0,3);
mean(variance(:))
```

```
ans = 0.0096
```

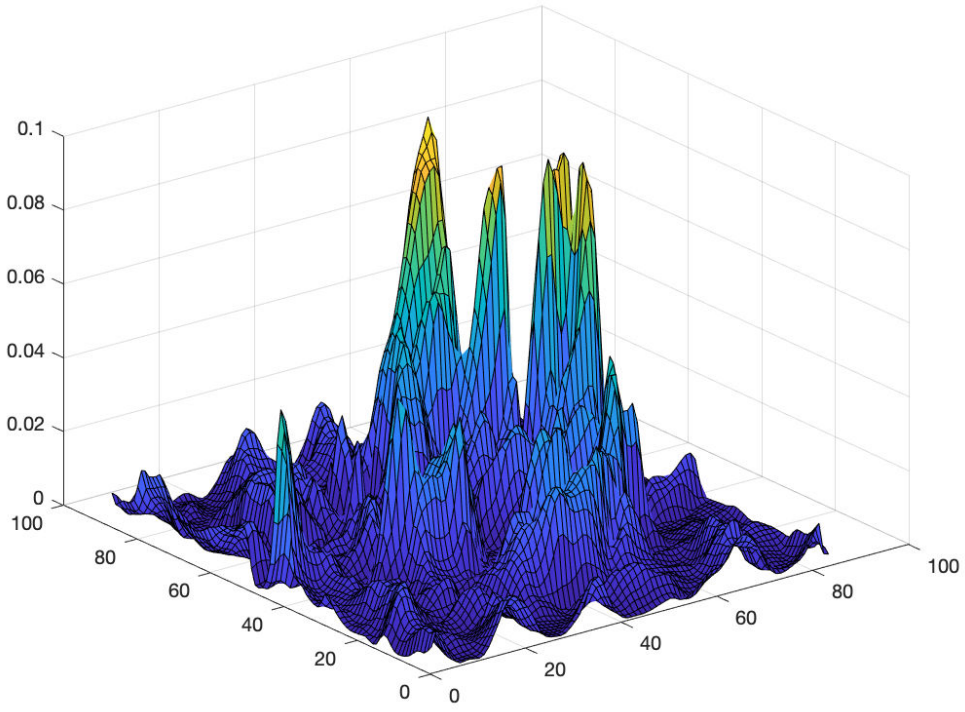
```
figure(1)
surf(avg,"FaceColor","none")
hold on
surf(deviation)
hold off
```




```
figure()
contour(deviation)
colorbar
```




```
surf(variance)
```



Approximating to a function

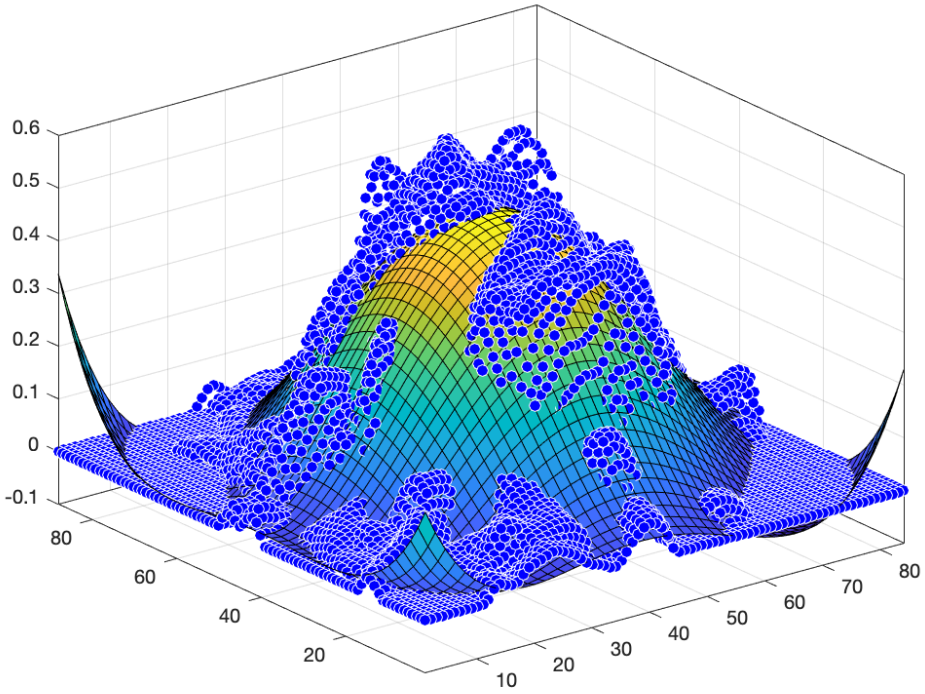
```
[X,Y] = meshgrid(1:y1,1:x1);
%figure(2)
%surf(X,Y,avg) --> used to ensure that the meshgrid was correctly done
%figure(3)
%contour(avg,15)
%colorbar
x2=[];
y2=[];
z2=[];
for i=1:size(X,1)
    for j=1:size(X,2)
        x2=[x2;X(i,j)];
        y2=[y2;Y(i,j)];
        z2=[z2;avg(i,j)];
    end
end

sf = fit([x2, y2],z2,'poly44','Robust','Bisquare' )
```

```
     Linear model Poly44:
     sf(x,y) = p00 + p10*x + p01*y + p20*x^2 + p11*x*y + p02*y^2 + p30*x^3 +
                   p21*x^2*y + p12*x*y^2 + p03*y^3 + p40*x^4 + p31*x^3*y
                   + p22*x^2*y^2 + p13*x*y^3 + p04*y^4
     Coefficients (with 95% confidence bounds):
       p00 =      0.2593  (0.2434, 0.2753)
       p10 =    -0.03165  (-0.03311, -0.0302)
       p01 =    -0.02559  (-0.02697, -0.0242)
       p20 =     0.00143  (0.001375, 0.001485)
       p11 =    0.001243  (0.001202, 0.001284)
       p02 =    0.001005  (0.0009543, 0.001055)
       p30 =  -2.573e-05  (-2.662e-05, -2.484e-05)
       p21 =  -1.204e-05  (-1.269e-05, -1.139e-05)
       p12 =  -1.371e-05  (-1.433e-05, -1.309e-05)
       p03 =  -1.679e-05  (-1.756e-05, -1.602e-05)
       p40 =    1.57e-07  (1.52e-07, 1.621e-07)
       p31 =  -4.344e-08  (-4.766e-08, -3.921e-08)
       p22 =   1.982e-07  (1.943e-07, 2.022e-07)
       p13 =  -2.844e-08  (-3.229e-08, -2.46e-08)
       p04 =   1.007e-07  (9.645e-08, 1.049e-07)
```

```
options = fitoptions(sf)
```

```
options =

    Normalize: 'off'
      Exclude: []
      Weights: []
       Method: 'LinearLeastSquares'
       Robust: 'Bisquare'
        Lower: [1×0 double]
        Upper: [1×0 double]
```

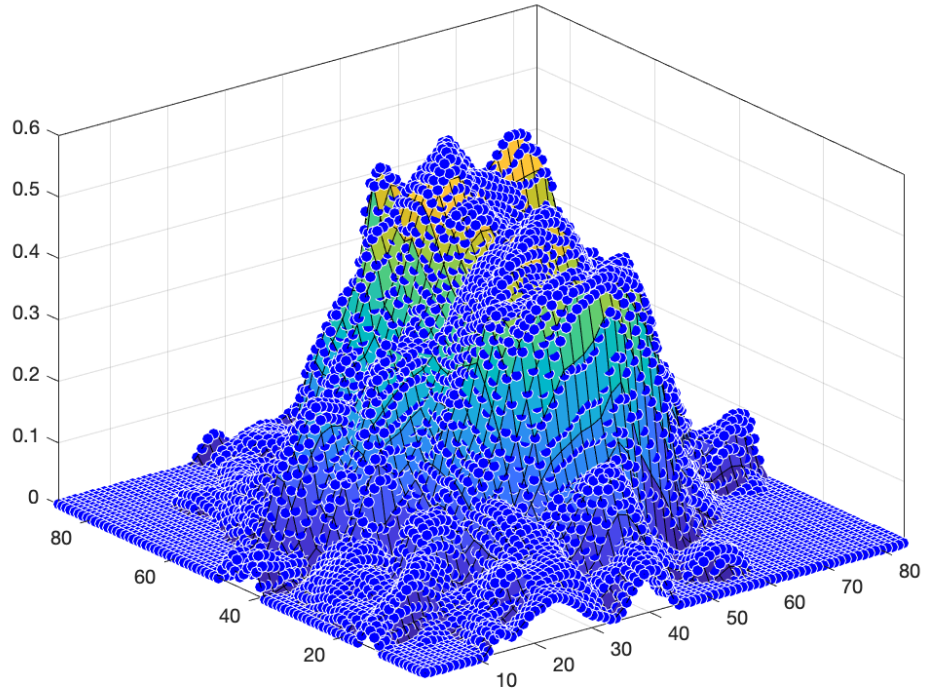
```
figure()
plot(sf,[x2,y2],z2)
```




```
sf1 = fit([x2, y2],z2,'linearinterp')
```

```
Linear interpolant:
  sf1(x,y) = piecewise linear surface computed from p
Coefficients:
  p = coefficient structure
```

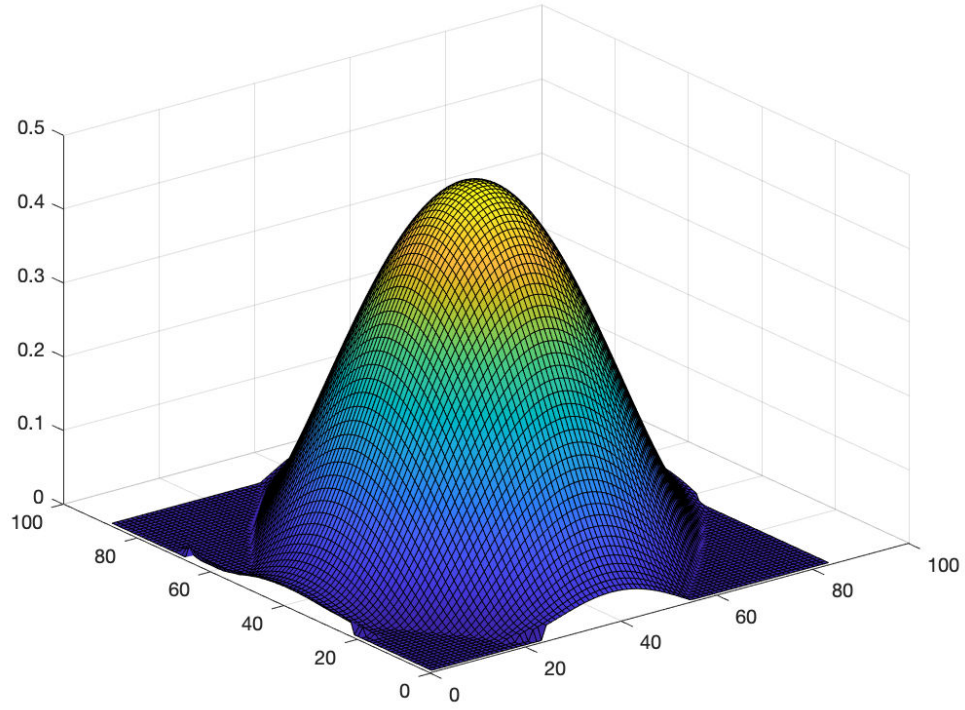
```
figure()
plot(sf1,[x2,y2],z2)
```




```
%z3=zeros(63,71);
adj=zeros(x1,y1);
adj1=adj; %a copy
```

We correct the negative values and those on the extreme that go again up

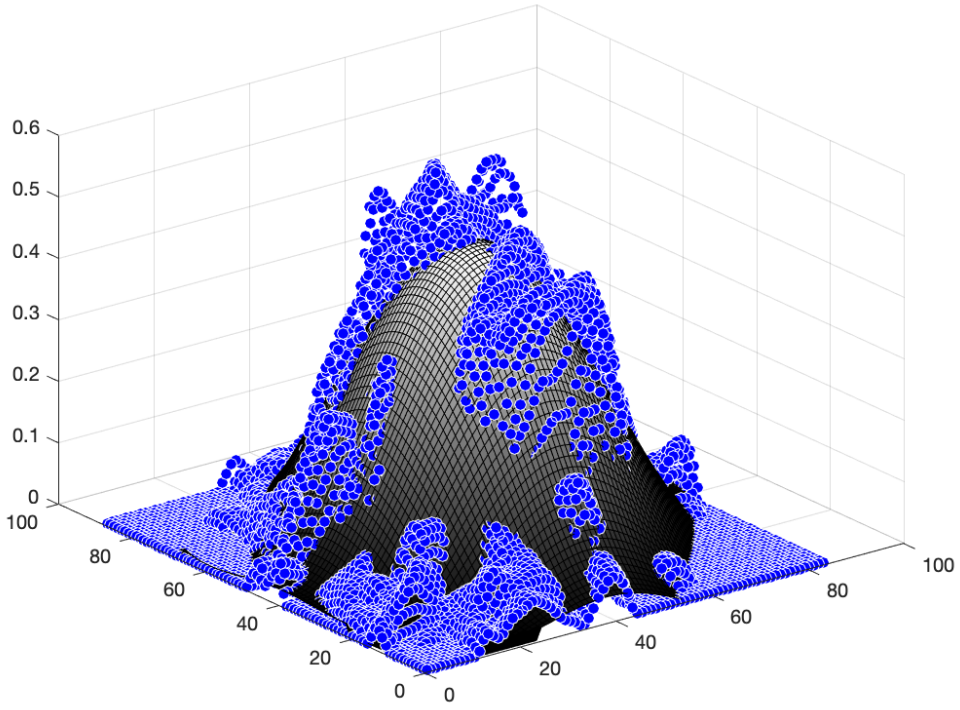
```
for i=1:length(x2)
    %z3(y2(i),x2(i))= z2(i);
    if sf(x2(i),y2(i)) > 0
        adj(y2(i),x2(i))= sf(x2(i),y2(i));
    else
        adj(y2(i),x2(i))= 0;
    end
    if (x2(i)-40)^2/2000+(y2(i)-45)^2/2200 > 1
        adj(y2(i),x2(i))= 0;
    end
    %if y2(i) < 60-3*x2(i)
        %adj(y2(i),x2(i))= 0.05;
    %elseif y2(i) > 259/3-10/21*x2(i)
        %adj(y2(i),x2(i))= 0;
    %elseif y2(i) < -39+5/7*x2(i)
        %adj(y2(i),x2(i))= 0;

    %end
end
surf (adj)
```



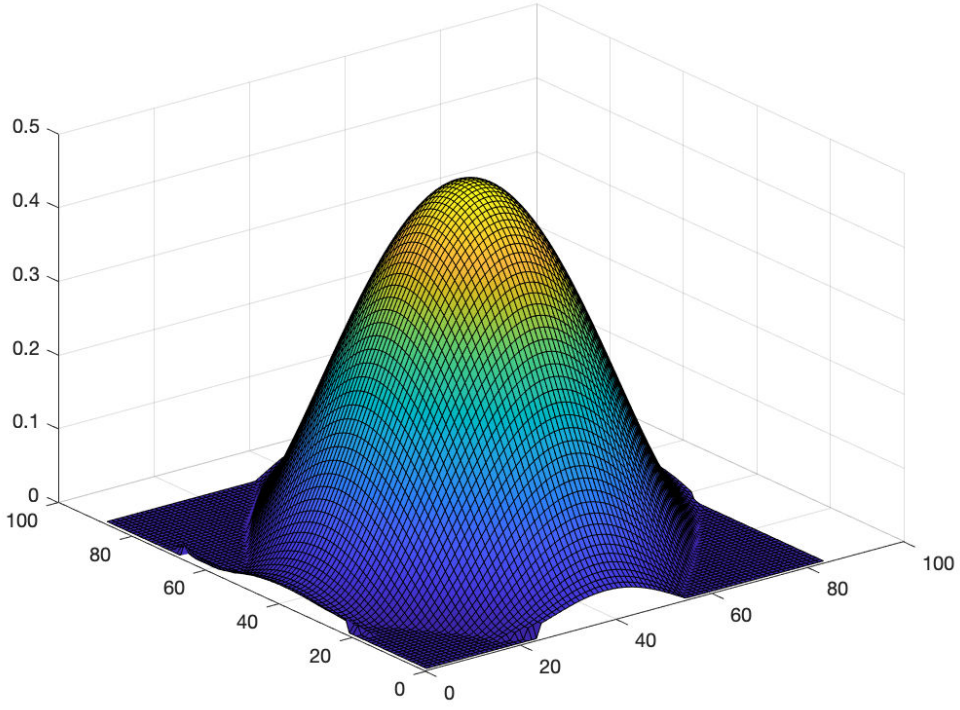

```
%(z3==avg) it is the same
```

Corrected values: adj and z2

```
surf(adj)
colormap("gray")
hold on
plot3(x2,y2,z2,'ow','MarkerSize',6,'MarkerFaceColor','blue')
hold off
```




```
surf (adj)
colormap("default")
```



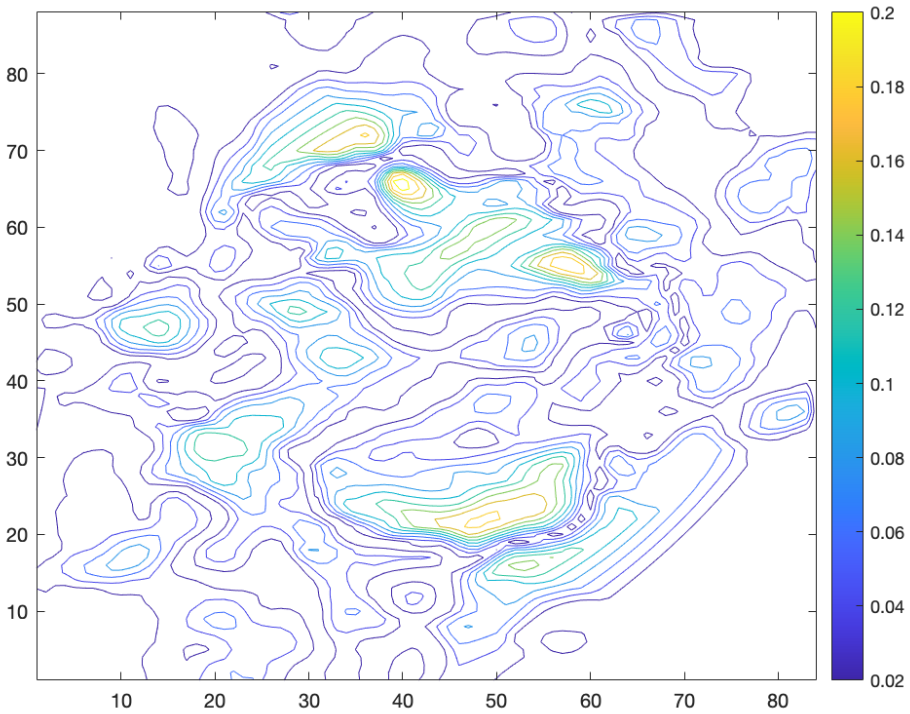
Errors

Absolute and relative errors

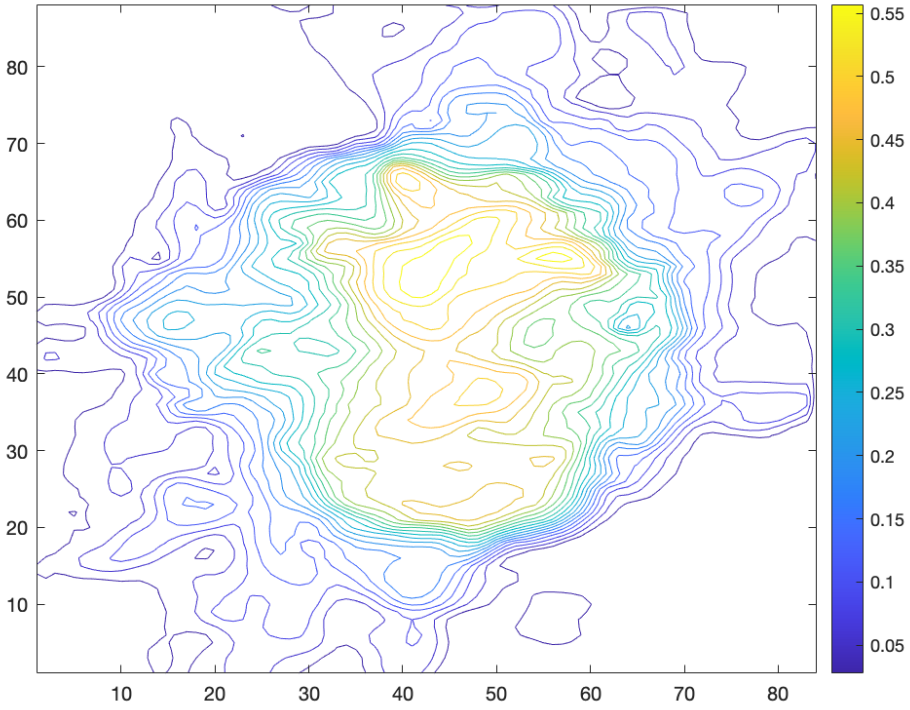
```
AbsErr=abs(adj-avg);
RelErr=AbsErr./avg*100;
mean(AbsErr(:))
```

```
ans = 0.0359
```

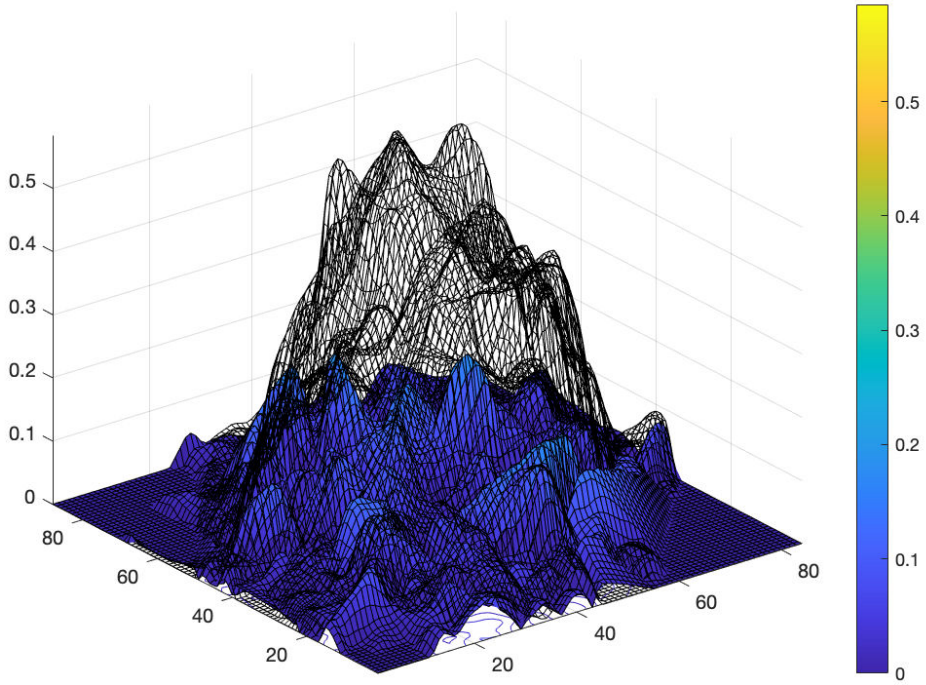
```
figure()
contour(AbsErr)
colorbar
```



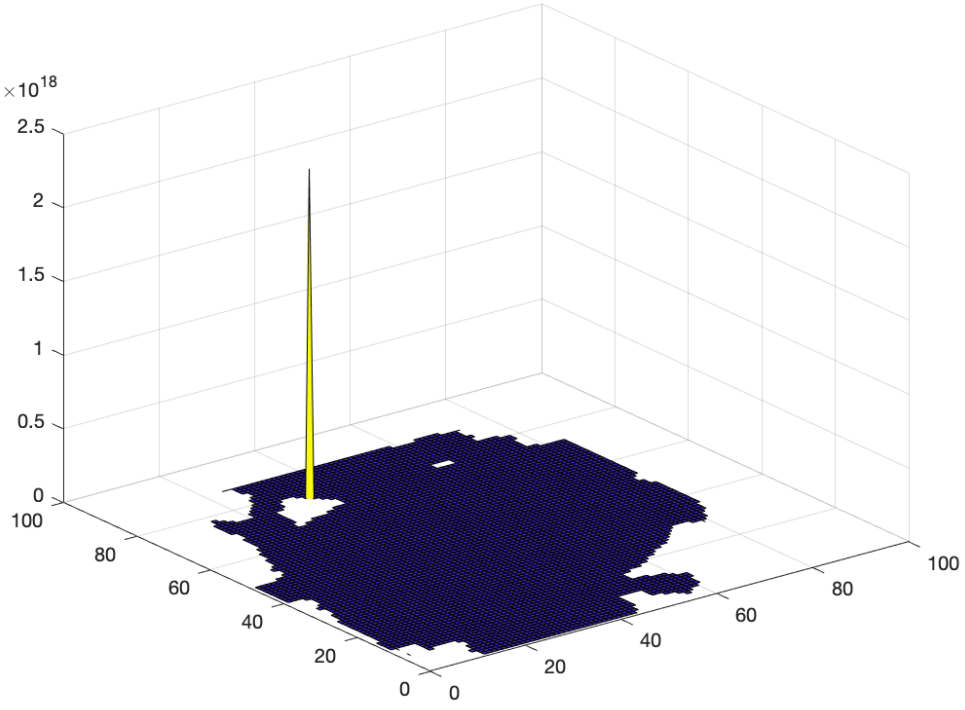

```
contour(avg,20)
colorbar
```




```
figure()
surfc(AbsErr)
hold on
surf(avg,'FaceColor',"none")
colorbar
hold off
```




```
figure()
surf(RelErr)
```



```
ans = NaN
```

Computation of RMSE

```
s=[]; %predicted data corrected
for i=1:size(X,1)
    for j=1:size(X,2)
        s=[s;adj(i,j)];
    end
end
s(s<0)=0;

z2=z2; %exact data
n=length(s);
MSE=1/n*sum((z2-s).^2) %mean squared error
```

```
MSE = 0.0027
```

```
Rsquared=1-MSE/var(z2)
```

```
Rsquared = 0.8965
```

```
R=sqrt(Rsquared)
```

```
R = 0.9468
```

Maximum height

```
maxheight=max(avg,[],'all')
```

```
maxheight = 0.5840
```

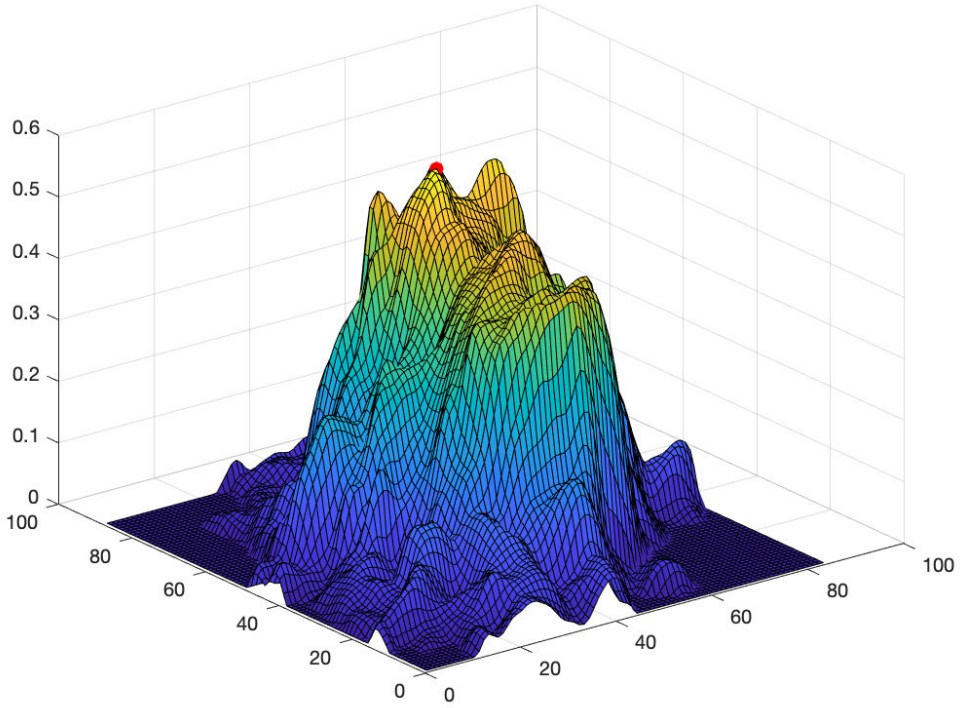
```
[rowsOfMaxes colsOfMaxes] = find(avg == maxheight)
```

```
rowsOfMaxes = 53
colsOfMaxes = 43
```

```
rough_slope_x= (maxheight)/colsOfMaxes
```

```
rough_slope_x = 0.0136
```

```
surf(avg)
hold on
plot3(colsOfMaxes,rowsOfMaxes,avg(rowsOfMaxes,colsOfMaxes),'.r',"MarkerSize",25)
hold off
```



Volume of the surface

```
[TriIdx, Vol] =convhull(X,Y,avg); %exact data

Vol %real volume
```

```
Vol = 1.9465e+03
```

```
s=sf(x2,y2);
s(s<0)=0;
[TriIdx, Vol1] =convhull(x2,y2,s);
Vol1
```

```
Vol1 = 2.2829e+03
```

```
AbsVol=abs(Vol1-Vol)
```

```
AbsVol = 336.4200
```

```
RelVol=AbsVol/Vol*100 %9.6190 (without robust)
```

```
RelVol = 17.2834
```

```
H_bar=Vol/(x1*y1)
```

```
H_bar = 0.2633
```

```
z_lim = 0.4237
UD = 0.1011
```

```
%save('adj_s6.mat',"adj")
%save('avg_s6.mat','avg')
%save('var_s6.mat','deviation')
```

MATLAB files of the pictures used in section 2.4.1:

```
clear all
cd '/Users/Maria/Desktop/BERNINA/Matlab-scripts'
add path '/Users/Maria/Desktop/BERNINA/Matlab-scripts/POLYNOMIAL
    FITS OF SAMPLES'

load('adj_s1.mat')
adj_s1=adj;
load('adj_s2.mat')
adj_s2=adj;
load('adj_s3.mat')
adj_s3=adj;
load('adj_s4.mat')
adj_s4=adj;
load('adj_s5.mat')
adj_s5=adj;
load('adj_s6.mat')
adj_s6=adj;
clear('adj');

figure()
subplot(2,3,1)
surf(adj_s1)
title('sample 1')
view([90 0])

subplot(2,3,2)
surf(adj_s2)
title('sample 2')
view([90 0])

subplot(2,3,3)
surf(adj_s3)
title('sample 3')
view([90 0])

subplot(2,3,4)
surf(adj_s4)
title('sample 4')
view([90 0])

subplot(2,3,5)
surf(adj_s5)
title('sample 5')
view([90 0])

subplot(2,3,6)
surf(adj_s6)
```

```
47 title('sample 6')
48 view([90 0])
49
50
51
52
53 load('avg_s1.mat')
54 adj_s1=avg;
55 load('avg_s2.mat')
56 adj_s2=avg;
57 load('avg_s3.mat')
58 adj_s3=avg;
59 load('avg_s4.mat')
60 adj_s4=avg;
61 load('avg_s5.mat')
62 adj_s5=avg;
63 load('avg_s6.mat')
64 adj_s6=avg;
65 clear('avg');
66 figure()
67 subplot(2,3,1)
68 surf(adj_s1)
69 title('sample 1')
70 view([90 0])
71
72 subplot(2,3,2)
73 surf(adj_s2)
74 title('sample 2')
75 view([90 0])
76
77 subplot(2,3,3)
78 surf(adj_s3)
79 title('sample 3')
80 view([90 0])
81
82 subplot(2,3,4)
83 surf(adj_s4)
84 title('sample 4')
85 view([90 0])
86
87 subplot(2,3,5)
88 surf(adj_s5)
89 title('sample 5')
90 view([90 0])
91
92 subplot(2,3,6)
93 surf(adj_s6)
94 title('sample 6')
```

```
95 view([90  0])
96
97
98 %% variance
99 clear all
100 cd '/Users/Maria/Desktop/BERNINA/Matlab-scripts'
101 addpath '/Users/Maria/Desktop/BERNINA/Matlab-scripts/POLYNOMIAL
      FITS OF SAMPLES'
102
103 load('var_s1.mat')
104 m1=mean(deviation(:));
105 adj_s1=deviation;
106 load('var_s2.mat')
107 m2=mean(deviation(:));
108 adj_s2=deviation;
109 load('var_s3.mat')
110 m3=mean(deviation(:));
111 adj_s3=deviation;
112 load('var_s4.mat')
113 m4=mean(deviation(:));
114 adj_s4=deviation;
115 load('var_s5.mat')
116 m5=mean(deviation(:));
117 adj_s5=deviation;
118 load('var_s6.mat')
119 m6=mean(deviation(:));
120 adj_s6=deviation;
121 clear('deviation');
122
123 figure()
124 subplot(2,3,1)
125 surf(adj_s1)
126 title('sample 1')
127 txt = {'Mean value: 0.1072'};
128 text(90,150,txt)
129
130 subplot(2,3,2)
131 surf(adj_s2)
132 title('sample 2')
133 txt = {'Mean value: 0.0890'};
134 text(95,135,txt)
135
136
137 subplot(2,3,3)
138 surf(adj_s3)
139 title('sample 3')
140 txt = {'Mean value: 0.1048'};
141 text(200,350,txt)
```

```
142
143
144 subplot(2,3,4)
145 surf(adj_s4)
146 title('sample 4')
147 txt = {'Mean value: 0.0978'};
148 text(70,120,txt)
149
150
151 subplot(2,3,5)
152 surf(adj_s5)
153 title('sample 5')
154 txt = {'Mean value: 0.1138'};
155 text(95,150,txt)
156
157
158 subplot(2,3,6)
159 surf(adj_s6)
160 title('sample 6')
161 txt = {'Mean value: 0.0821'};
162 text(150,240,txt)
```

User Testing datasheet:

User	Sample	Letter	Readibility	Stitch	Issues
1	2	C	1	A	Confusion with little dot
1	2	H	0	A	Confusion with little dot
1	2	A	1	A	Confusion with little dot
1	2	T	0	A	Confusion with little dot
1	3	T	1	A	Confusion with little dot
1	3	A	1	A	Confusion with little dot
1	3	B	1	A	Confusion with little dot
1	3	L	1	A	Confusion with little dot
1	3	E	1	A	Confusion with little dot
1	4	S	0	A	Confusion with little dot
1	4	A	0	A	Confusion with little dot
1	4	B	0	A	Confusion with little dot
1	4	L	0	A	Confusion with little dot
1	4	E	0	A	Confusion with little dot
1	5	O	0	A	Confusion with little dot
1	5	U	0	A	Confusion with little dot
1	5	R	0	A	Confusion with little dot
1	5	S	0	A	Confusion with little dot
1	6	T	0	B	Confusion with little dot
1	6	B	0	B	Confusion with little dot
1	6	O	1	B	Confusion with little dot
1	6	R	1	B	Confusion with little dot
1	6	D	0	B	Confusion with little dot
1	7	P	1	B	Confusion with little dot
1	7	O	1	B	Confusion with little dot
1	7	R	1	B	Confusion with little dot
1	7	T	1	B	Confusion with little dot
1	7	E	1	B	Confusion with little dot
1	8	C	0	B	Confusion with little dot
1	8	O	1	B	Bad spacing
1	8	R	1	B	Bad spacing
1	8	D	1	B	Bad spacing
1	8	E	0	B	Bad spacing
1	9	S	0	B	Bad spacing
1	9	P	1	B	Bad spacing
1	9	O	1	B	Bad spacing
1	9	R	1	B	Bad spacing
1	9	T	1	B	Bad spacing
2	2	C	1	A	Confusion with little dot
2	2	H	1	A	Confusion with little dot
2	2	A	1	A	Confusion with little dot
2	2	T	1	A	Confusion with little dot
2	3	T	1	A	Confusion with little dot
2	3	A	1	A	Confusion with little dot
2	3	B	1	A	Confusion with little dot

2	3	L	1	A	Confusion with little dot
2	3	E	1	A	Confusion with little dot
2	4	S	0	A	Confusion with little dot
2	4	A	1	A	Confusion with little dot
2	4	B	1	A	Confusion with little dot
2	4	L	1	A	Confusion with little dot
2	4	E	1	A	Confusion with little dot
2	5	O	0	A	Confusion with little dot
2	5	U	0	A	Confusion with little dot
2	5	R	0	A	Confusion with little dot
2	5	S	0	A	Confusion with little dot
2	6	T	0	B	Confusion with little dot
2	6	B	1	B	Confusion with little dot
2	6	O	1	B	Confusion with little dot
2	6	R	1	B	Confusion with little dot
2	6	D	0	B	Confusion with little dot
2	7	P	1	B	Confusion with little dot
2	7	O	1	B	Confusion with little dot
2	7	R	1	B	Confusion with little dot
2	7	T	1	B	Confusion with little dot
2	7	E	1	B	Confusion with little dot
2	8	C	1	B	Confusion with little dot
2	8	O	1	B	Bad spacing
2	8	R	1	B	Bad spacing
2	8	D	1	B	Bad spacing
2	8	E	1	B	Bad spacing
2	9	S	0	B	Bad spacing
2	9	P	1	B	Bad spacing
2	9	O	1	B	Bad spacing
2	9	R	1	B	Bad spacing
2	9	T	1	B	Bad spacing
3	2	C	0	A	Confusion with little dot
3	2	H	0	A	Confusion with little dot
3	2	A	0	A	Confusion with little dot
3	2	T	0	A	Confusion with little dot
3	3	T	0	A	Confusion with little dot
3	3	A	0	A	Confusion with little dot
3	3	B	0	A	Confusion with little dot
3	3	L	0	A	Confusion with little dot
3	3	E	0	A	Confusion with little dot
3	4	S	0	A	Confusion with little dot
3	4	A	0	A	Confusion with little dot
3	4	B	0	A	Confusion with little dot
3	4	L	0	A	Confusion with little dot
3	4	E	0	A	Confusion with little dot
3	5	O	0	A	Confusion with little dot
3	5	U	0	A	Confusion with little dot

3	5	R	0	A	Confusion with little dot
3	5	S	0	A	Confusion with little dot
3	6	T	0	B	Confusion with little dot
3	6	B	0	B	Confusion with little dot
3	6	O	0	B	Confusion with little dot
3	6	R	0	B	Confusion with little dot
3	6	D	0	B	Confusion with little dot
3	7	P	1	B	Confusion with little dot
3	7	O	1	B	Confusion with little dot
3	7	R	1	B	Confusion with little dot
3	7	T	1	B	Confusion with little dot
3	7	E	1	B	Confusion with little dot
3	8	C	0	B	Confusion with little dot
3	8	O	0	B	Bad spacing
3	8	R	0	B	Bad spacing
3	8	D	0	B	Bad spacing
3	8	E	0	B	Bad spacing
3	9	S	0	B	Bad spacing
3	9	P	0	B	Bad spacing
3	9	O	0	B	Bad spacing
3	9	R	0	B	Bad spacing
3	9	T	0	B	Bad spacing

			Stitch 1			Stitch 2		
Metrics		**Weight**	**Value**	**Rank**	**W Rank**	**Value**	**Rank**	**W Rank**
Abs Error	Min	0.1	0.0428	3	0.3	0.0378	4	0.4
MSE	Min	0.1	0.0033	2	0.2	0.0026	4	0.4
R	Max	0.1	0.9516	6	0.6	0.9174	2	0.2
Rel. Vol Error	Min	0.1	7.3025	4	0.4	2.3534	5	0.5
Max Height	Max	0.1	0.6223	6	0.6	0.4193	1	0.1
Mean Height	Max	0.5	0.3117	6	3	0.2109	1	0.5
TOTAL		1			2.1			1.6
(/100)					35			26.66666667
			Stitch 3					
			Value	**Rank**	**W Rank**	**Value**	**Rank**	**W Rank**
			0.0468	1	0.1	0.0365	5	0.5
			0.004	1	0.1	0.0023	6	0.6
			0.9084	1	0.1	0.9232	3	0.3
			9.0904	2	0.2	1.2176	6	0.6
			0.5097	3	0.3	0.4597	2	0.2
			0.2871	5	2.5	0.2204	2	1
					3.3			2.2
					55			36.66666667
			Stitch 5					Stitch 6
			Value	**Rank**	**W Rank**	**Value**	**Rank**	**W Rank**
			0.0432	2	0.2	0.0359	6	0.6
			0.0031	3	0.3	0.0027	5	0.5
			0.932	4	0.4	0.9468	5	0.5
			9.0713	3	0.3	17.28	1	0.1
			0.5433	4	0.4	0.584	5	0.5
			0.2556	3	1.5	0.2603	4	2
					3.1			2.2
					51.66666667			36.66666667

Complete Computation Decision Matrix