End-of-study internship final report :

Development of a rapid process parameter optimization methodology

Maxime ACHARD

2020

Acknowledgement

Firstly, I would like to thank all people that made this internship possible including Emilien HANS, my friend and colleague who also did his internship in AmPro Innovations and at MCAM.

I think about our supervisors Professor Aijun HUANG and Doctor Louis CHIU. They both warmly welcomed us. They also took their time to help us when we needed or made sure we were alright during the virus pandemic. It has been a real pleasure to work with them.

Also, I would like to thank Matthieu BONNERIC who helped us a lot during this internship.

Finally, I want to thank Professor Nicolas SAINTIER. Without him, I would not meet AmPro Innovations and MCAM. He also did regular meetings with us to follow the project progress.

Contents

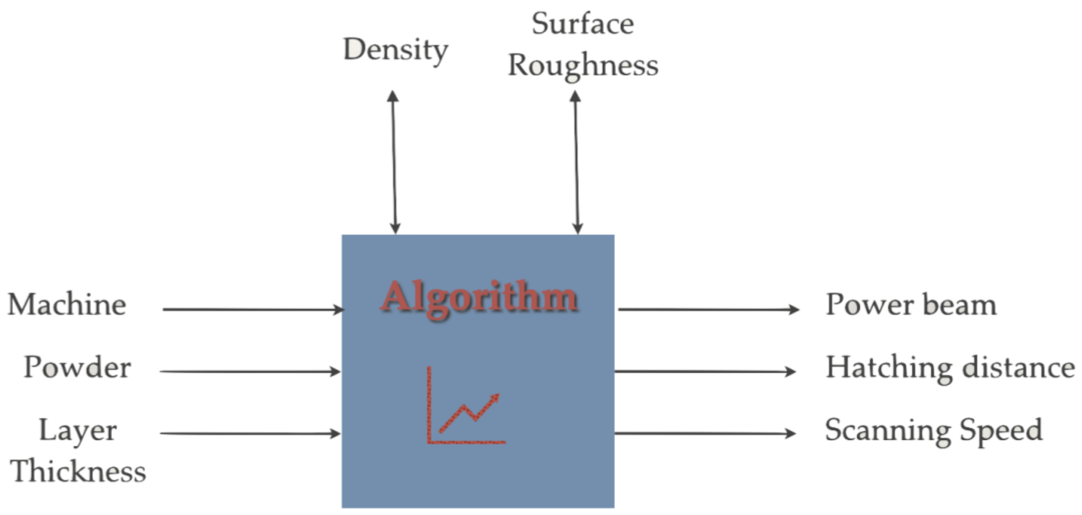
Context :

There are high expectations for additive manufacturing in many fields (such as aeronautics or medical) and for the moment the mastery of the process is not sufficient to exploit it in certain applications where the criticality of the part is high. Consequently, research activities are particularly intense including the Selective Laser Melting (SLM) process. SLM is a very flexible process that can be used to produce components in a wide variety of different materials. The SLM process involves the use of a laser heat source to melt metallic powder on a powder bed to form the desired geometry of the component. The mechanical performance of the resulting part is very sensitive to the processing parameters. These parameters primarily include the laser power, scanning speed or hatch distance. Currently, the development of optimal processing parameters for a given material is particularly laborious and sometimes approximations are made based on parameters for similar materials due to time constraints.
The use a Bayesian optimization algorithm can have the potential to greatly reduce the laborious nature of this task. This project aims to develop a method that can optimize processing parameters for a material for its build density, surface roughness and mechanical performance with minimal printing involved.

I. Introduction :

1) Goal :

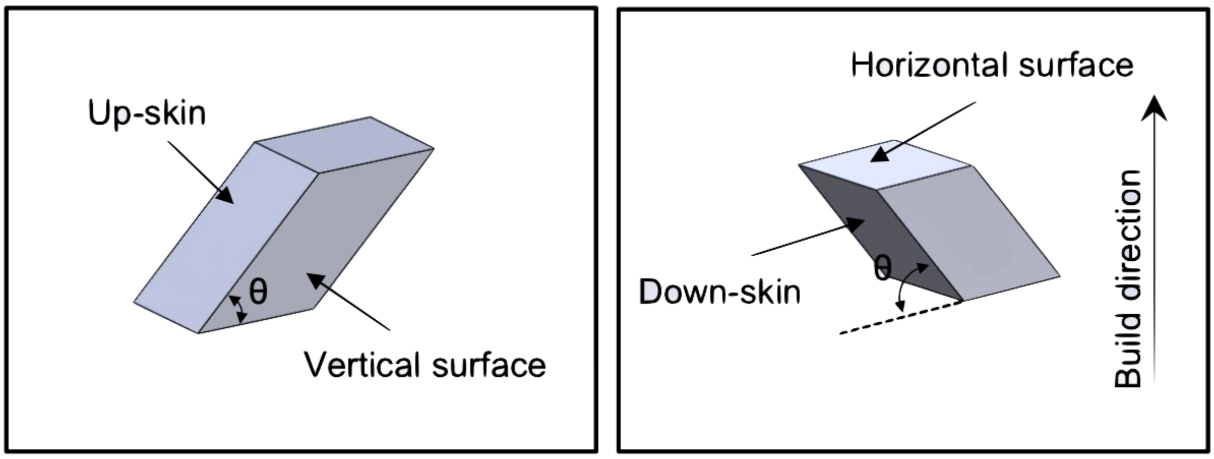
The goal of this project is to design an algorithm to find an optimized set of parameters (Power Beam; Scanning Speed and Hatching distance). All others parameters are fixed. The two main criteria are : the density of the piece and its surface roughness (up-skin and down-skin). The density must be higher than 99.5%.



As the machine allows us to set different sets of parameters for the core and the contouring of the piece, each set of parameters is optimized separately.

Samples are designed and printed such as we can measure density and surface roughness for up-skin and down-skin. It is a squared parallelepiped of 20mm height and a 10mm by side. The parallelepiped is tilted to a theta angle of 45° from the platform.

The algorithm is run with Ti-6Al-4V (or Ti64) alloy samples printed with an EOS M290 machine but obviously it can be applied to any alloy and to any machine. The layer thickness is fixed to 30µm.



2) The traditional way to optimize such a problem :

A traditional way to optimize such a problem is to implement a design of experiments (DoE) based algorithm.
The results obtained with a DoE is consistent to get a good set of parameters and it is possible to design an optimization algorithm to enhance it. More specifically, The principle is to carry out a series of DoE for a fixed machine/powder/layer thickness configuration, whose input parameters are :

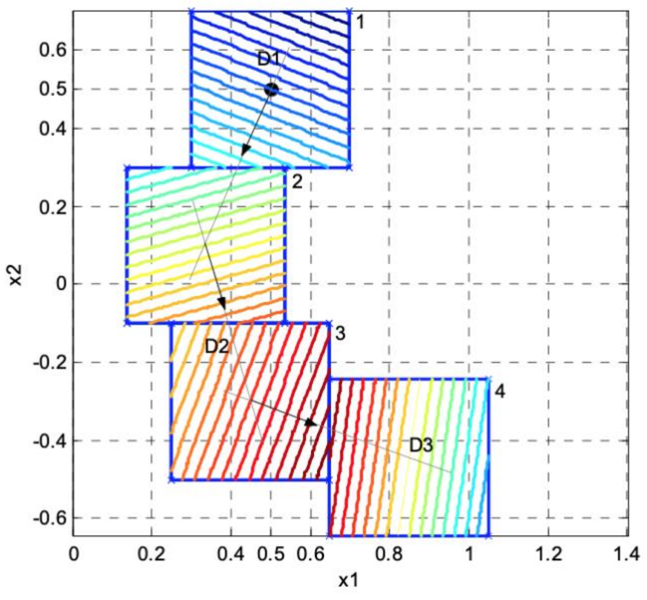
- Laser power beam
- Scanning speed
- Hatching distance

From one experimental design to another, the field of study is modified to approach the optimal parameters. Strategies [1] for changing the field of study can be of two types :

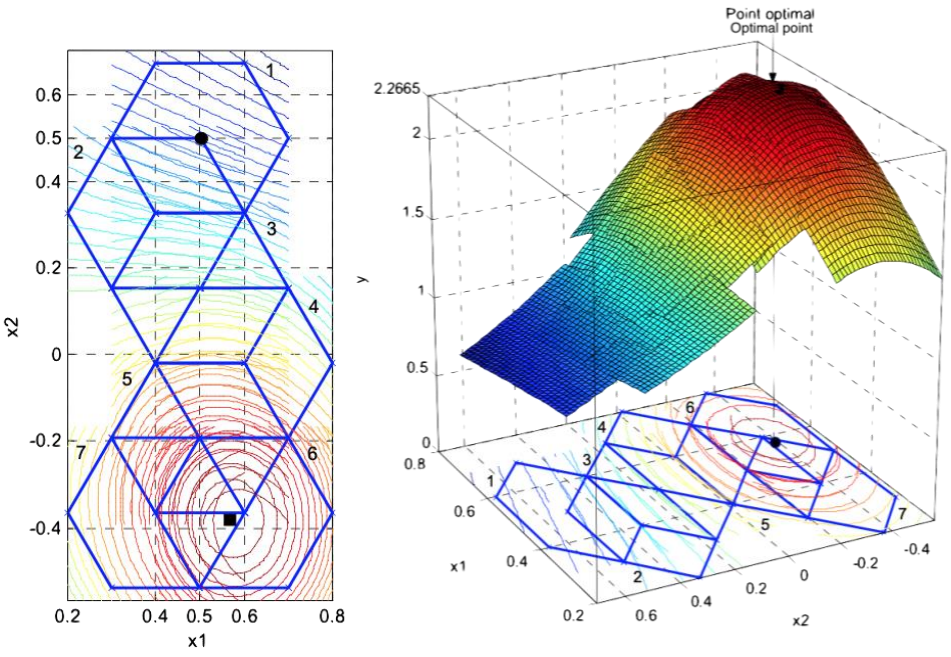
- Sliding the field of study
- Zooming on the field of study

Sliding strategy :

The sliding strategy consists in making a first plan of experiment and see in which direction it is necessary to move on the field of study in order to optimize the set of parameters. A second DoE is then carried out after having shifted the field of study to the previously identified set of parameters and these operations are repeated as many times as necessary to reach a suitable set of parameters. There are different ways of applying this algorithm. One can, after each iteration, calculate the model (1st or 2nd order) and use a gradient descent algorithm to move the domain of study or simply reuse points already calculated at the edge of the domain of study to reduce the number of experiments.



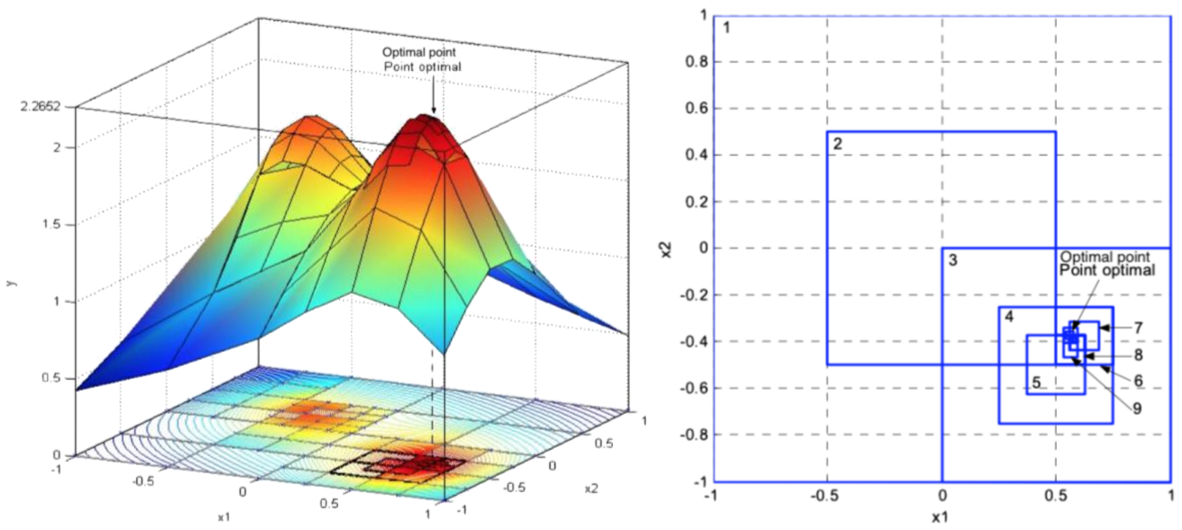
Sliding strategy and gradient descent algorithm



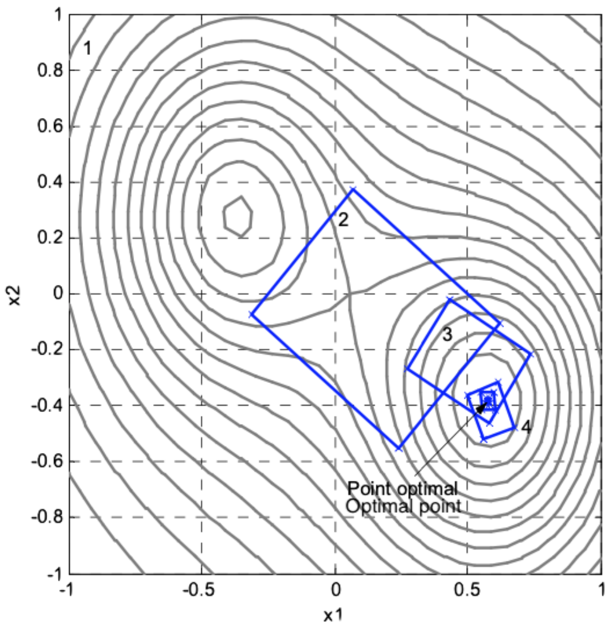
Sliding strategy with Doehlert and reuse of points

Zooming strategy :

The zooming strategy consists of zooming in on the field of study between each iteration. A first DoE is made, with a very large field of study. In the second iteration, the field of study is specified around a promising set of parameters and so on until it converges to a suitable set of parameters for our problem. Variants exist : simple zoom, zoom with rotation and translation, with or without model calculation between each iteration...



Simple zoom algorithm



Zoom algorithm with rotation and translation

Conclusion of DoE algorithm :

These algorithms are quite expensive in term of time and does not ensure the optimized set of parameter because hypothesis and approximations are made regarding the number of parameters involved and their interaction between each other. For example, to reduce the number of experiments, it is common to select a limited number of parameters and to assume that a three order relation between parameters is insignificant in regards to a second order relation.

As the relation between all the parameters is unknown because of the complexity of the process and all the physic phenomenon involved, a smart way to find an optimized set of parameters is to consider a “black box” objective function. As a consequence, the real system behavior is considered, not only its mathematical modeling. Moreover, the cost of a trial is expensive (in term of time and money). Consequently it is essential to minimize the number of test to reach an optimized set of parameters.

Therefore, Bayesian optimization method seems to be a relevant method to optimize the set of parameter for SLM process.

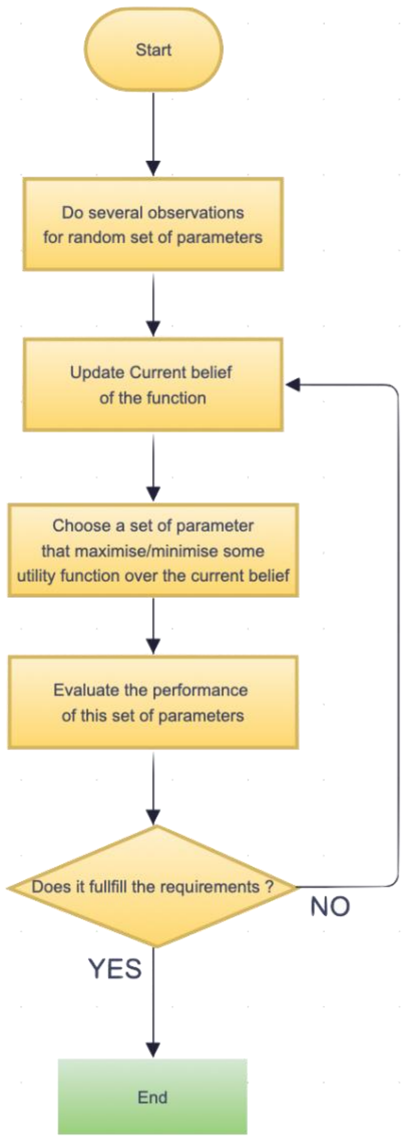
II. Bayesian Optimization :

1) What is the principle ?

This optimization algorithm is based on a probabilistic approach (Bayesian inference theory). This algorithm is widely used in the field of artificial intelligence (machine learning, neural networks...). It consists in measuring values (density, surface roughness...) for a given configuration where the probability of reaching the optimum is the highest.
From a series of initial observations, the algorithm returns parameters to be tested that seem the most promising. These parameters are calculated by extrapolating the observations using a Gaussian (or Kriegeage) process.

Flowchart :

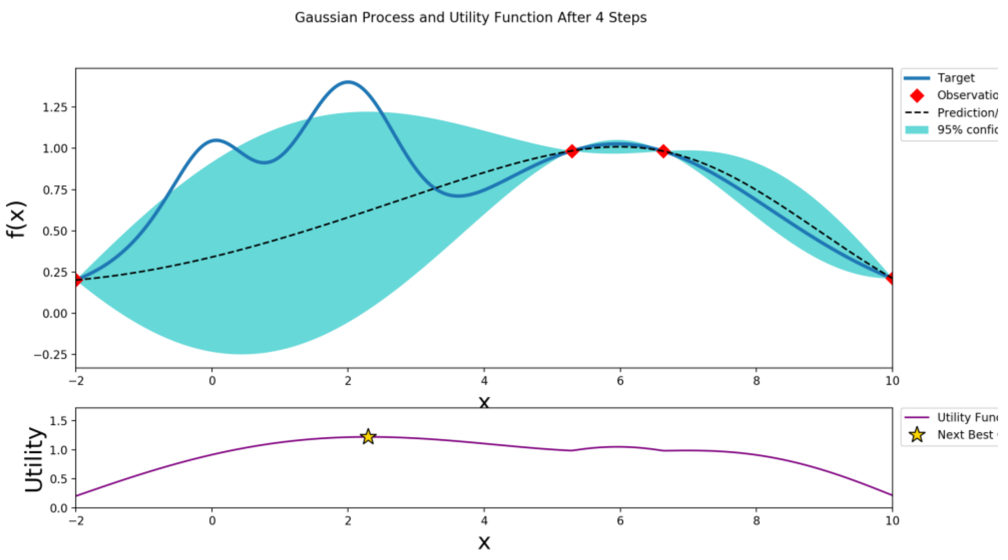
Here is the flowchart of the Bayesian optimization algorithm :

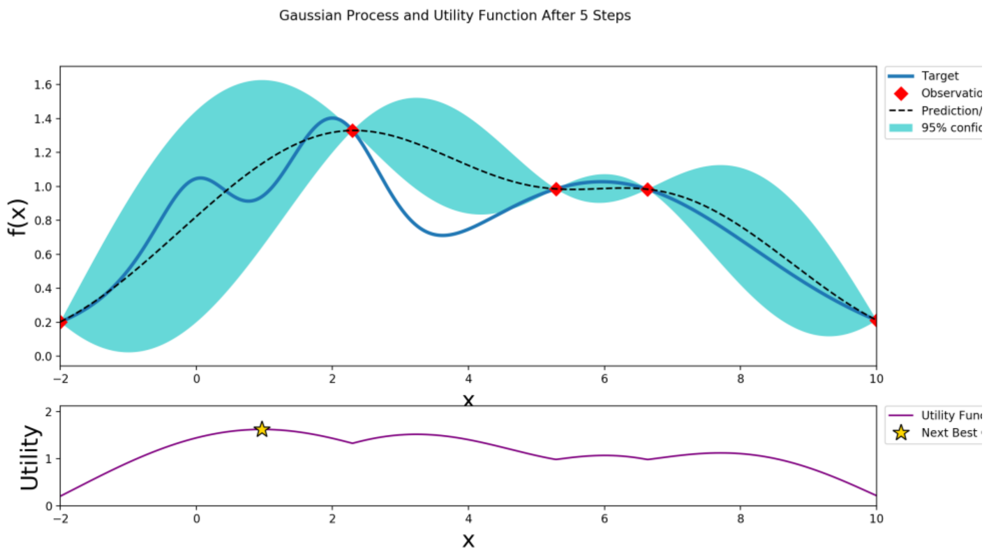

Start
Do several observations for random set of parameters
Update Current belief of the function
Choose a set of parameter that maximise/minimise some utility function over the current belief
Evaluate the performance of this set of parameters
Does it fullfill the requirements ?
NO
YES
End

Flowchart of the Bayesian algorithm

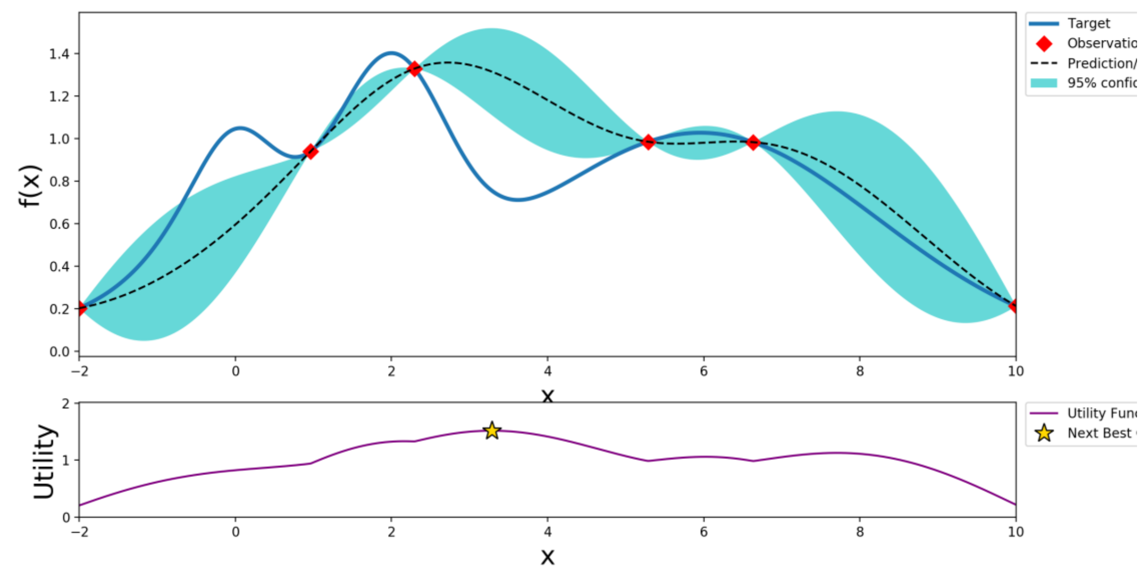
Example for a single parameter problem :

The example below shows some iterations of the Bayesian algorithm in the search for the maximum of the target function (objective function). The Gaussian method allows to evaluate which abscissa value has the highest probability to reach the maximum. This abscissa is called "Next Best Guess" on the representation of the utility function below and it is the abscissa where the 95% confidence envelope reaches the maximum. The 95% confidence envelope in the result of the Gaussian process. This value is then tested and the knowledge of the target function is updated (represented by the dotted curve "Prediction/GP Mean"). In this case, the maximum is found in about ten iterations with only two initial observations. This can vary depending on the function, the field of study, the number of initial observations, the observation points initially chosen, etc...

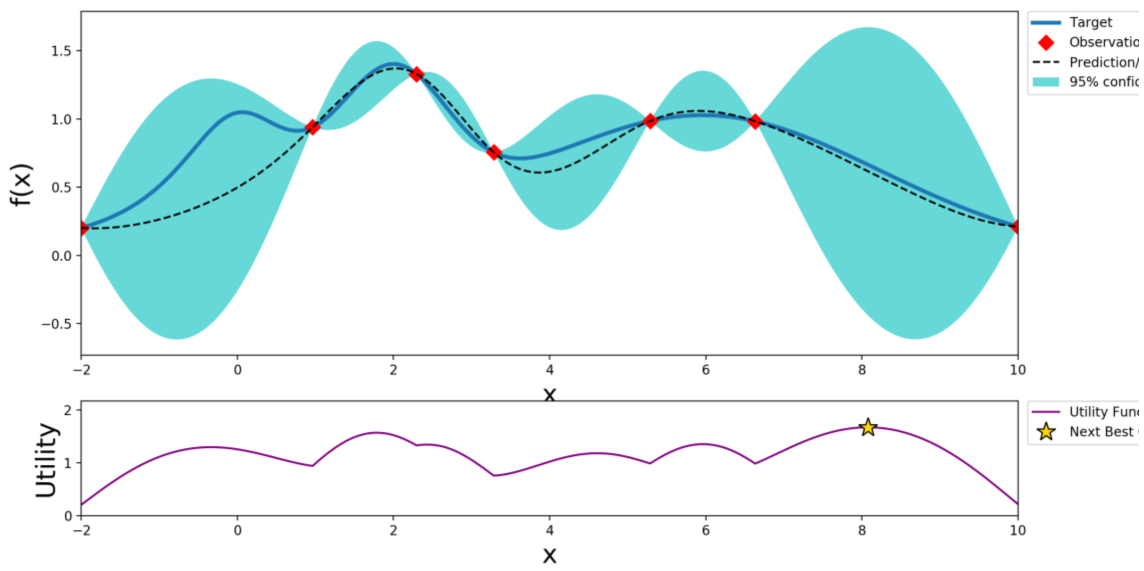




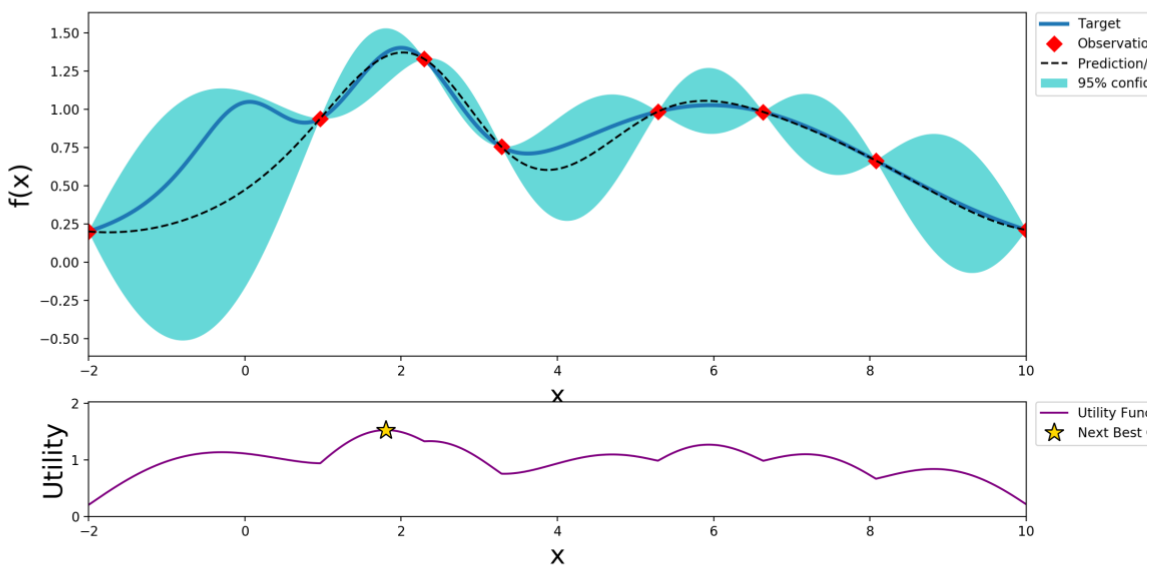
Gaussian Process and Utility Function After 6 Steps

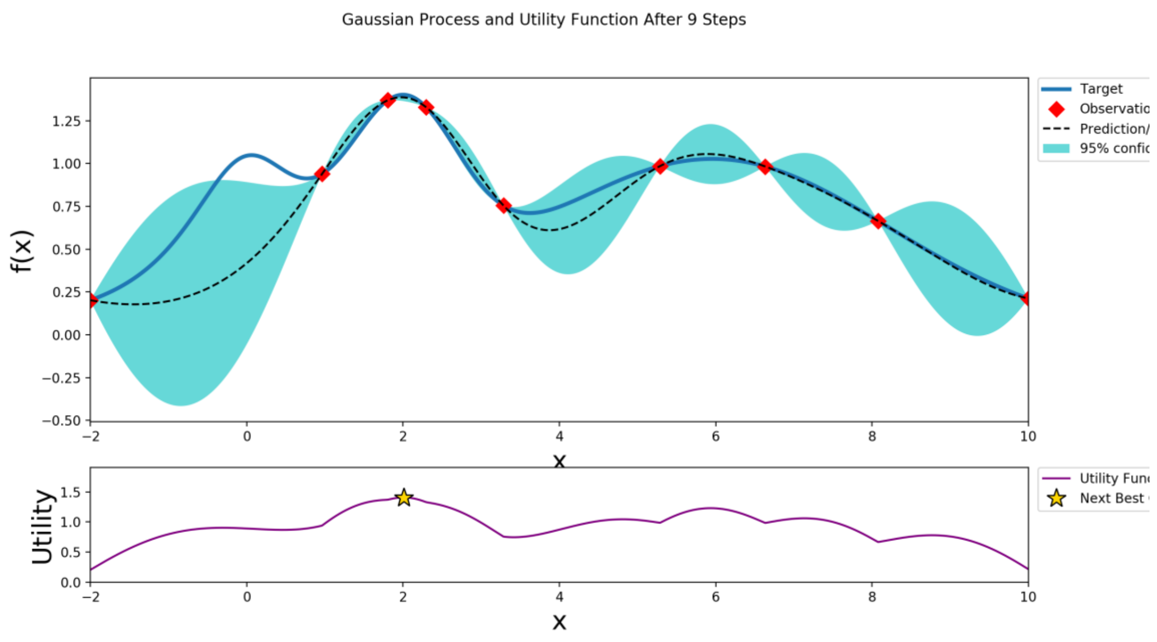


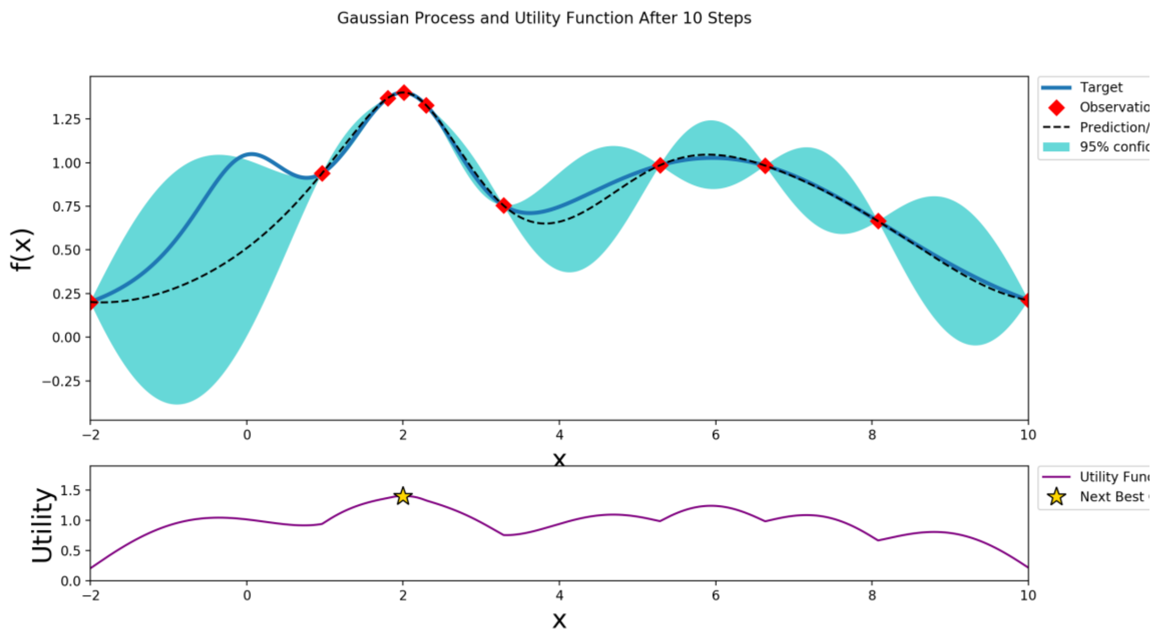
Gaussian Process and Utility Function After 7 Steps



Gaussian Process and Utility Function After 8 Steps







Example step by step : How to find the maximum of the utility function ?

2) Python Library :

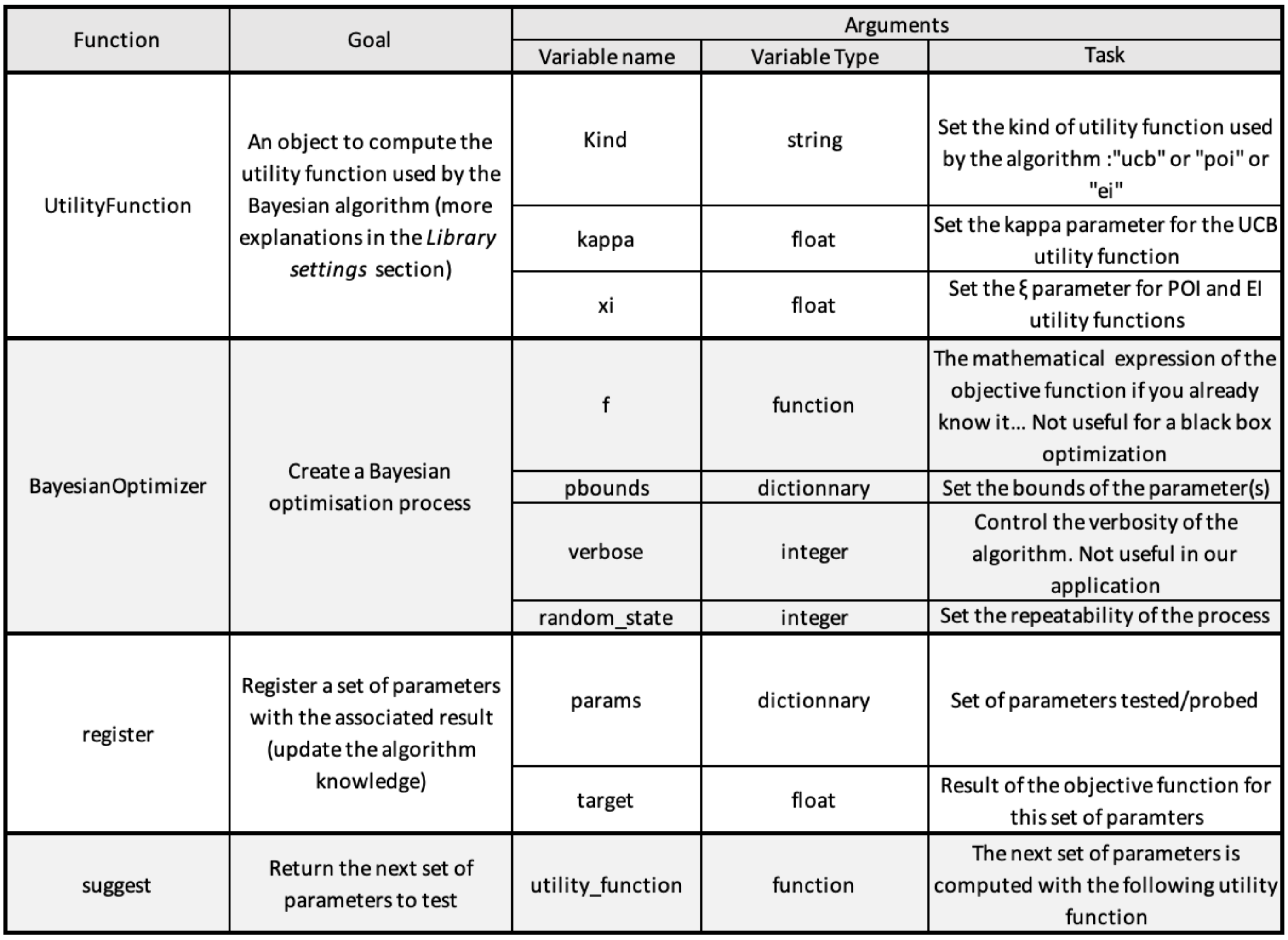
In the following work, a Bayesian algorithm is coded using a python library developed by Fernando M. F. Nogueira licensed under the MIT License.

The library [2] can be find on github : github.com/fmfn/BayesianOptimization

This library is operational both for single or multiple parameters.

Library functions :

Here are the functions used for our application and the main settings needed to reach our goal. We can notice some functions or settings in library are useless for our specific application.



Function	Goal	Arguments		
		Variable name	Variable Type	Task
UtilityFunction	An object to compute the utility function used by the Bayesian algorithm (more explanations in the *Library settings* section)	Kind	string	Set the kind of utility function used by the algorithm :"ucb" or "poi" or "ei"
		kappa	float	Set the kappa parameter for the UCB utility function
		xi	float	Set the ξ parameter for POI and EI utility functions
BayesianOptimizer	Create a Bayesian optimisation process	f	function	The mathematical expression of the objective function if you already know it... Not useful for a black box optimization
		pbounds	dictionnary	Set the bounds of the parameter(s)
		verbose	integer	Control the verbosity of the algorithm. Not useful in our application
		random_state	integer	Set the repeatability of the process
register	Register a set of parameters with the associated result (update the algorithm knowledge)	params	dictionnary	Set of parameters tested/probed
		target	float	Result of the objective function for this set of paramters
suggest	Return the next set of parameters to test	utility_function	function	The next set of parameters is computed with the following utility function

Library settings :

This library has to be configurated according to your needs. Here are the settings we need to focus on.

Utility functions :

There are three different utility functions used in this library. Each function is used in the algorithm at each iteration for different reasons (See the strategy paragraph). In the next paragraphs you can find a brief explanation of the mathematical concepts. However, to go deeper in mathematical explanations you can read the paper wrote by Eric Brochu, Vlad M. Cora and Nando de Freitas in 2010 [3] (from page 11 to 15).
The main parameter to keep in mind for choosing a utility function is the balance between exploration of the search domain and the exploitation of the results (ie : the points already probed).

- Exploration : Sets of parameters where the variance is high
  - The algorithm tends to probe very different set of parameters
- Exploitation : Sets of parameters where the surrogate mean is high
  - The algorithm tends to trust sets of parameters already probes

POI : Probability Of Improvement :

This utility function is well-suited if you are already very close to the optimized set of parameters. There is no exploration of the domain without introducing a trade-off parameter called ξ (or xi in the python script) to balance exploration and exploitation. ξ ≥0, should be high at the beginning of the optimization to make sure the algorithm explore but then it needs to be decreased. However, it is not always the case and in practice, we can choose ξ empirically. This parameters enables the algorithm to balance the ratio between exploration and exploitation.

$$POI(X) = \Phi\left(\frac{\mu(X) - Ymax - \xi}{\sigma(X)}\right)$$

Φ : normal Cumulative Distribution Function (CDF)

- For a continuous random variable X, the CDF is the integral of its probability density function ϕ (PDF) :

$$\Phi_X(\mathrm{y}) = \int_{-\infty}^{y} \phi_X(t)\, dt\,.$$

With :

ϕ : Probability Density Function

μ : Gaussian mean
σ : Standard deviation of the Gaussian
ξ : Trade-off parameter

EI : Expected Improvement :

It would be relevant to take into account not only the probability of improvement but also the magnitude of this improvement (ie : minimize the expected deviation from the maximum of the objective function when choosing a new set of parameters).

$$Z(X) = \frac{\mu(X) - Ymax - \xi}{\sigma(X)}$$

$$EI(X) = Z(X).\phi \circ Z(X) + \sigma(X).\phi \circ Z(X)\,;\; if\; \sigma(X) > 0$$

$$EI(X) = 0\,;\; if\;\; \sigma(X) = 0$$

Φ : Normal Cumulative Distribution Function (CDF)
ϕ : Probability Density Function (PDF)
μ : Gaussian mean
σ : Standard deviation of the Gaussian
ξ : Trade-off parameter

The trade-off parameter is a key parameter for the efficiency of the process. Paradoxically, it is not a good idea to choose a high ξ at the beginning and decreased it in the end according to Lizotte's work [4,5]. Consequently, it is chosen empirically in practice.

UCB : Upper Confidence Bound :

UCB is used in the Sequential Design for Optimization algorithm (or SDO algorithm) to maximize a problem. (If you want to minimize a problem, you use the LCB : Lower Confidence Bound).
UCB utility function returns the maximum value across the weighted sum of the expected performance on the search domain. The expected performance is given by :

$$UCB(X) = \mu(X) + \kappa.\sigma(X)$$

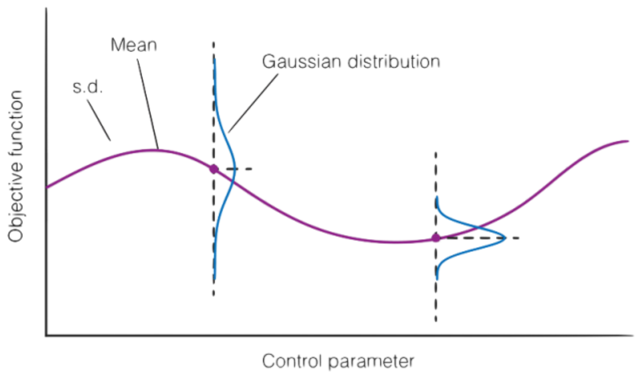
With :

- μ : Gaussian mean
- σ : Standard deviation of the Gaussian
- κ : Control the balance between exploitation and exploration over the search domain (it is also a trade-off parameter). It is like a “weight factor”.

So to choose the next parameter to probe :

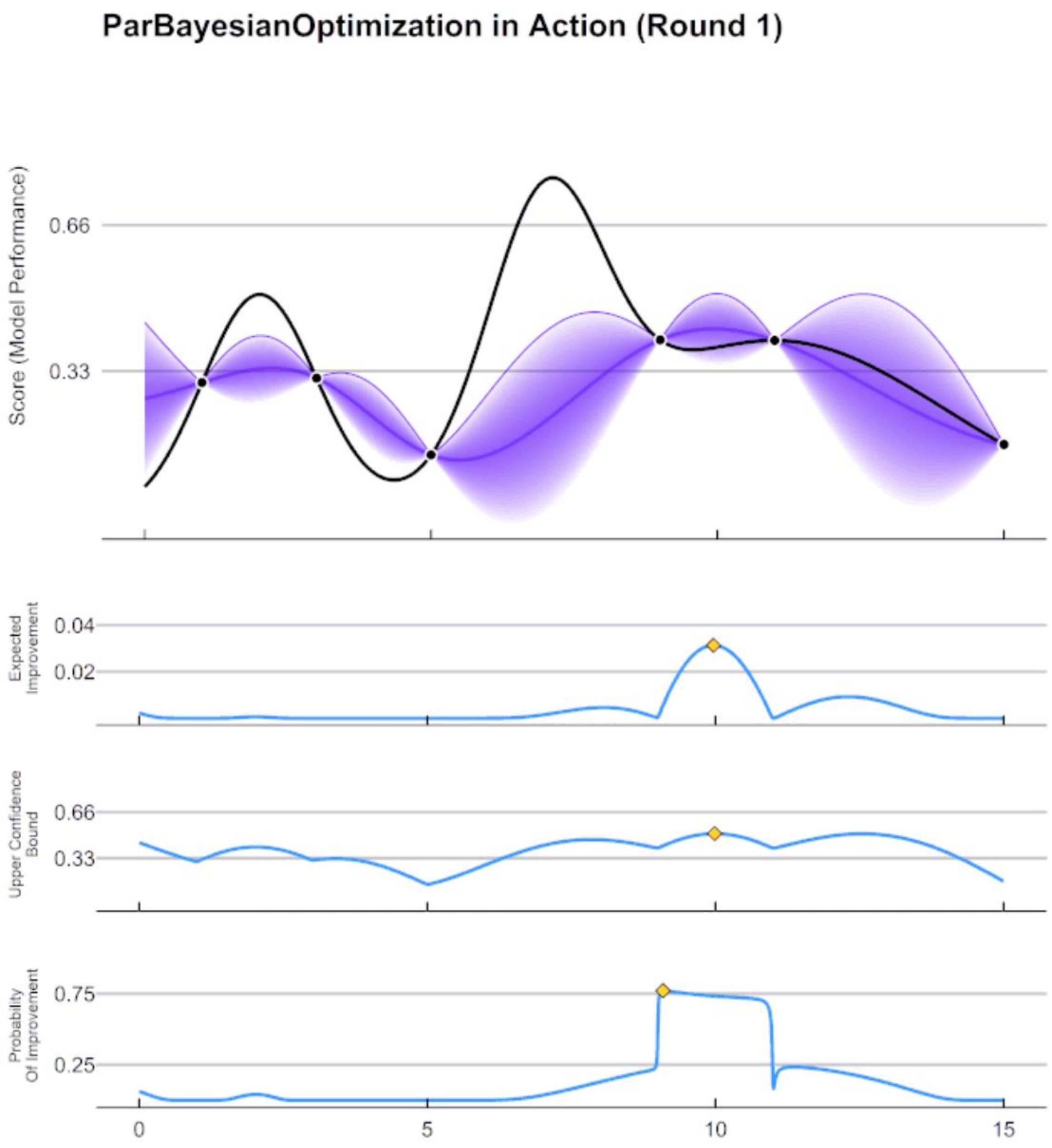
$$X_{n+1} = Arg\,max_X\,(\mu_n(X) + \kappa.\sigma_n(X))$$

If κ is high (>2.5), the algorithm will focus on the exploration of the domain but if κ is low(<1.5), it will focus on the exploitation of the results, where a high performance parameter is expected.

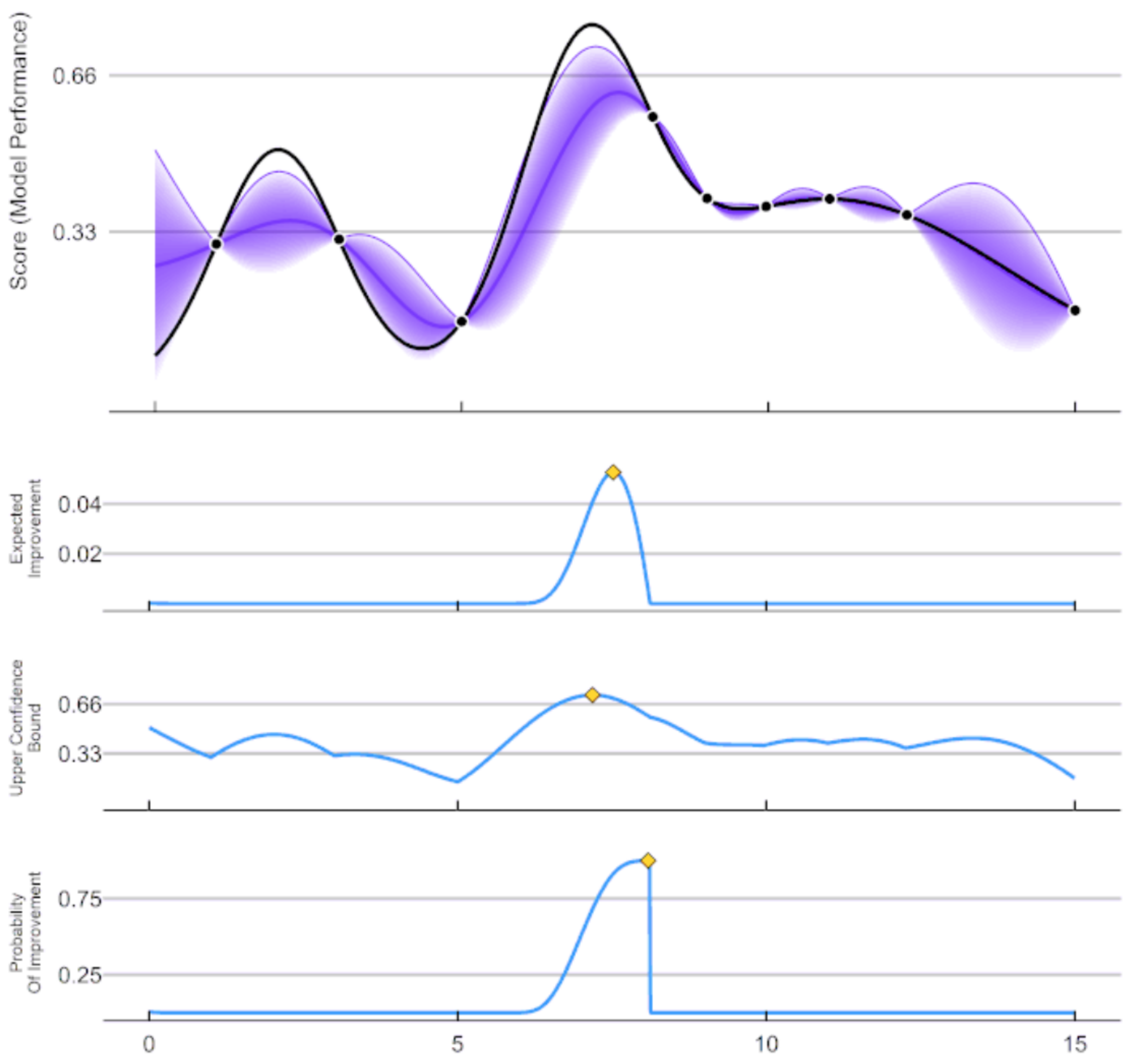


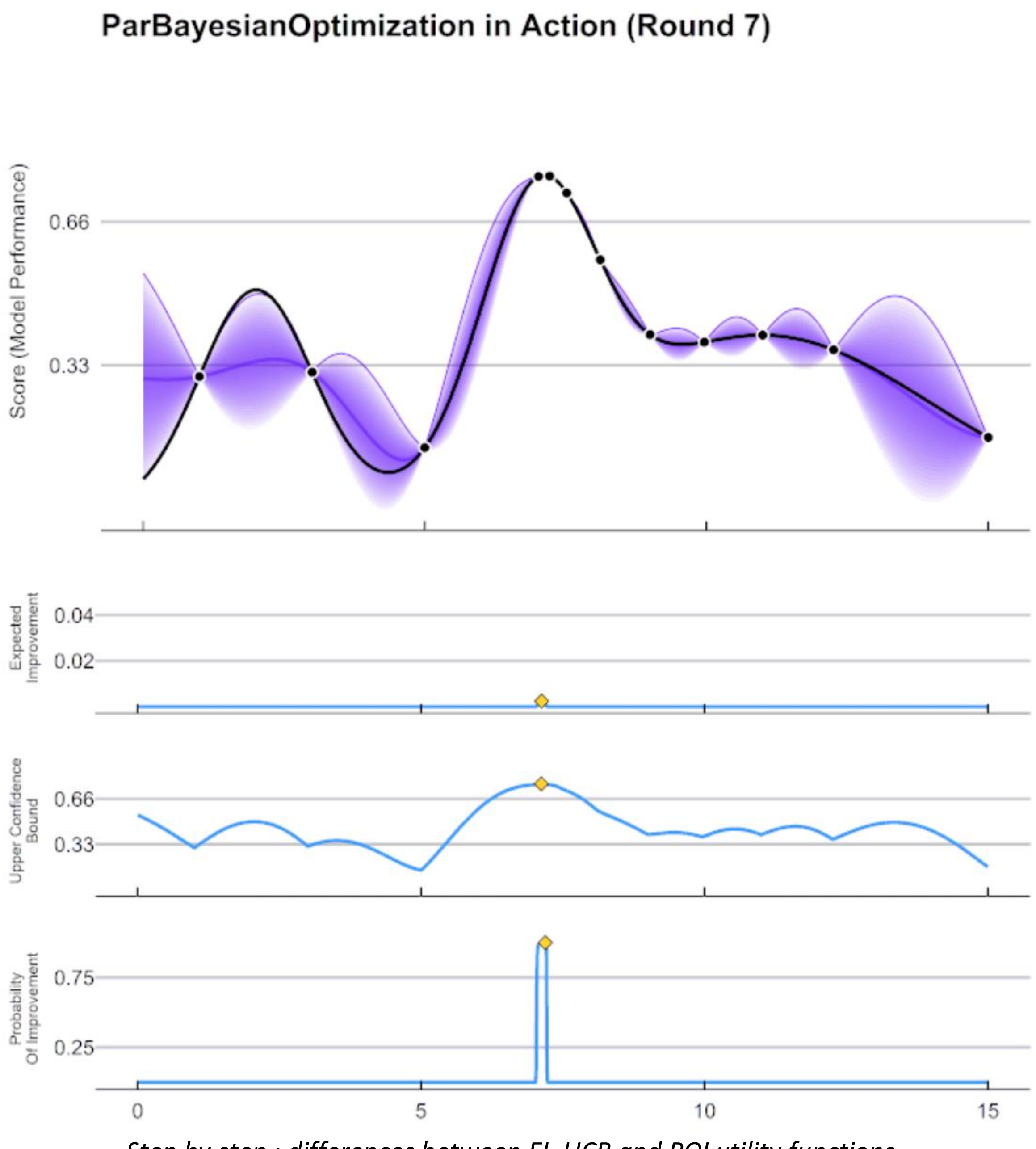
Representation of the different utility functions :

The next illustrations are from Bayesian optimization Wikipedia page and represent the difference between the three kind of utility functions. On top, there is a plot of an objective function, with prediction and the confidence envelop and below, the plots of the EI, UCB and POI utility functions. We can notice at the beginning there are some differences between the next abscissa to probe depending on the utility function. However, the results converge toward the same abscissa.
The UCB function seems less confident regarding the next abscissa to test than the POI and EI utility functions. Indeed, the UCB function is more spread all over the search domain.


ParBayesianOptimization in Action (Round 1)
Score (Model Performance)
0.66
0.33
Expected Improvement
0.04
0.02
Upper Confidence Bound
0.66
0.33
Probability Of Improvement
0.75
0.25
0
5
10
15

ParBayesianOptimization in Action (Round 4)





Step by step : differences between EI, UCB and POI utility functions

Convergence test :

A convergence test have been realized in order to have an idea of the influence of the kappa parameter and the trade-off parameter. It is realized for a linear function and for a quadratic one so we have two different kind of behavior. However, as we do not know the real kind of function that linked the different parameters (power beam, scanning speed and hatching distance) we cannot conclude if it is better to choose a particular value or another one.

POI	Iterations	
xi	abs(150-p)	(150-p)^2
0.0	55	30
0.001	54	30
0.01	54	30
0.1	67	14
1	27	14
1.5	21	14
1.75	19	14
1.8	18	14
1.85	NOCONV	14
1.9	18	14
1.95	19	14
2	NOCONV	14

EI	Iterations	
xi	abs(150-p)	(150-p)^2
0.0	32	30
0.001	32	30
0.01	41	30
0.1	39	15
1	NOCONV	15
1.5	21	15
1.75	19	15
1.8	NOCONV	15
1.85	NOCONV	15
1.9	18	15
1.95	18	15
2	NOCONV	15

We can notice, it does not always converge (NOCONV). It seems trade-off parameter has a significative influence over the number of iteration that are needed to converge and also over the kind of function. Literature suggests that this value should be more or less around 0.01 for artificial intelligence applications. However, we can conclude that depending on the objective function, a higher value could lead to a better result particularly for a quadratic form of objective function. In practice, it is chosen empirically.

UCB	Iterations	
kappa	abs(150-p)	(150-p)^2
1.96	36	25
2.5	34	25
5	29	25
10	22	25
20	12	25
50	9	30

Regarding the kappa parameter, it seems more relevant to have a "small" kappa parameter for a quadratic form of objective function but it is the opposite for a linear objective function.

Repeatability of the process :

As the process will be run over several days, we want the process to be repeatable to ensure the data from the previous iterations to be as consistent as possible. Indeed, sometimes is it as relevant to choose a specific abscissa as another one slightly different but the results are very different. It leads to a different series of set of parameters to probe to reach the same optimized set of parameters. Consequently, we need to set properly the "random_state" setting. It is important set an integer as an argument. The value of the number is not important, it just needs to be different than "None" and it needs to be always the same at each iteration. Indeed, it ensures that the process deals with the previous data the same way iterations after iterations. This way, we can run the algorithm different times with exactly the same series of next sets of parameters to probe in order to reach the optimized set of parameters.

III. Strategy :

1) Minimize the number of iteration :

Bayesian optimization allows to obtain an optimized set of parameters in a limited number of trials. However, this criterion alone is not sufficient in the case of SLM process optimization. Indeed, it is tedious to produce a specimen because a complete cycle must be carried out :

1) Prepare the printout on the machine software with the appropriate set of parameters
2) Prepare the machine
3) Start printing and wait for it to finish
4) Unpack
5) Clean the machine
6) Separate the specimen from the platform
7) Carry out density and roughness measurements
8) Provide data to the algorithm and run the next iteration

In order to optimize this cycle, we chose to print several specimens per platform. This leads us to choose different utility functions for each specimen.

2) How to choose the proper set of utility functions ?

Each specimen is made with a specific set of parameters returned by the Bayesian algorithm. Therefore, at each iteration of the algorithm, ten different sets of parameters are returned by the algorithm.
One of the challenge is to find the best combination between different utility functions.
It is also important to put the proper settings in each function. The settings are chosen according to the convergence test of each utility functions.

A matrix test have been done to get an idea of the different combination possible to reach an efficient algorithm.

For this test matrix, we used the following *-unreal-* relation between the Power Beam (p), the Scanning Speed (s) and the Hatching distance (h) :

$$f(p,s,h) = p^2 + 2.s.p + 2.s^2 + h^2 ; \ (p,s,h) \in [-300 ; 300]^3$$

The goal is to minimize this function.

This relation is more complex than a linear relation between all parameters because s and p parameters are linked together. In the real relation between all parameters, we can supposed parameters are linked together. There are eleven initial observations widespread randomly over $[-300 ; 300]$. The algorithm has been run five times for each combination and the number of iteration needed to reach the minimum of f is given in the corresponding cell. Settings implemented in the utility functions are recorded in the second tab (κ and ξ).

	UCB	POI	EI	#1	#2	#3	#4	#5
A	10	0	0	8	+10	10	9	9
B	9	1	0	+10	9	+10	10	+10
C	9	0	1	6	+10	9	+10	+10
D	0	10	0	+10	+10	+10	+10	+10
E	0	0	10	+10	+10	+10	+10	+10
F	2	4	4	10	+10	5	+10	+10
G	3	4	3	9	8	+10	10	+10
H	4	3	3	8	10	10	9	8
I	4	3	3	8	9	8	10	9
J	5	3	2	9	8	9	8	10

	kappa parameter		POI xi parameter		EI xi parameter					
A	1	1.96	2.5	5	10	20	30	40	50	60
B	1	1.96	2.5	5	10	20	30	40	50	0.1
C	1	1.96	2.5	5	10	20	30	40	50	0.1
D	0.0	0.001	0.01	0.1	1.0	1.5	1.75	1.8	1.85	1.95
E	0.0	0.001	0.01	0.1	1.0	1.5	1.75	1.8	1.85	1.95
F	2.5	10	0.01	0.1	1.5	1.95	0.01	0.1	1.5	1.95
G	2.5	5	10	0.1	1.5	1.95	0.01	0.1	1.5	1.95
H	1.96	2.5	5	10	0.01	0.1	1.5	0.01	0.1	1.5
I	2.5	5	10	20	0.01	0.1	1.5	0.01	0.1	1.5
J	1.96	2.5	5	10	20	0.1	1.5	0.01	0.1	1.5

Eventually, it does not seem very relevant to try ten sets of promising parameters returned by the same kind of utility function.
The only utility function that converge toward an optimized set of parameters is the UCB one. However, the convergence is quite slow and exploration of the domain is limited even with different kappa parameters (values end up being very close to each other). This could stick the algorithm in a local maximum.
EI and POI utility functions converge quickly toward an optimized area but do not reach the optimized set of parameters (even if the ξ parameter is 0.0). When a promising area is found,

algorithm starts to explore more the domain. The algorithm cannot be stuck in a local optimum and it improves the knowledge of the algorithm about the objective function.
We can take advantage of these results to design an "optimized" set of utility functions in order to enhance the performances of the optimization process. Consequently, according the test results we can choose to return four results provided by UCB utility function with different values of kappa and three EI and POI utility function with various trade-off parameter values.

It seems relevant to choose :

UCB :

1. $\kappa = 1.96$
2. $\kappa = 2.5$
3. $\kappa = 5$
4. $\kappa = 10$

POI :

1. $\xi = 0.01$
2. $\xi = 0.1$
3. $\xi = 1.5$

EI :

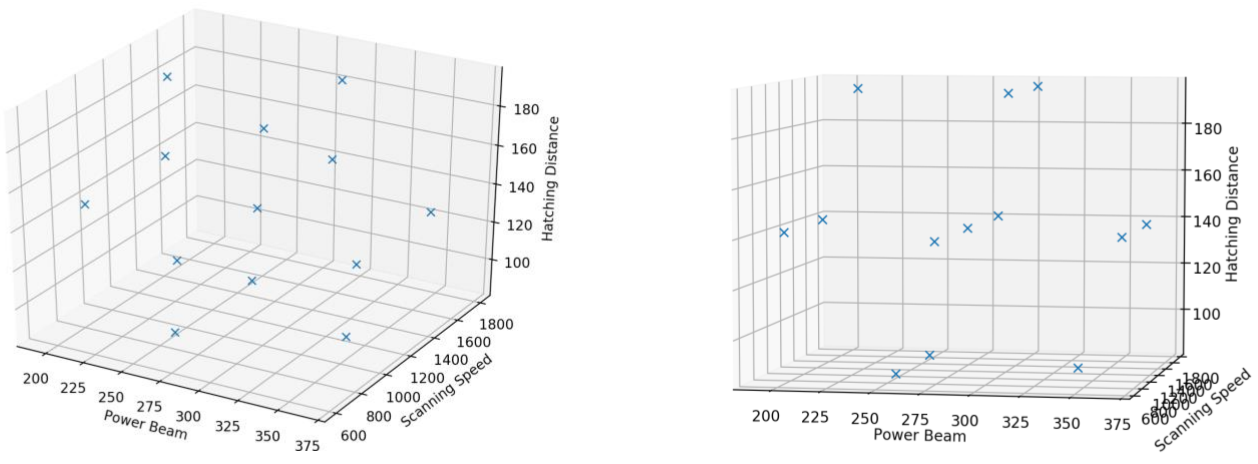
1. $\xi = 0.01$
2. $\xi = 0.1$
3. $\xi = 1.5$

What is the influence of the initial observations ?

We can easily understand the more initial observations you have the better it is (it takes less iterations to find the optimized set of parameters). Nevertheless, how initial observations are distributed over the search domain is important. Here is an overview of results for the same objective function seen previously.

			Initial Observation Domain				
UCB	POI	EI	[-300,300]	[-200,200]	[-100,100]	[-50,50]	[100,300]
kappa	xi	xi	10	7	5	3	+10
1.96	0.01	0.01	6	4	3	2	+10
2.5	0.1	0.1	10	5	5	5	+10
5	1.5	1.5	10	8	2	3	+10
10			7	5	5	2	+10
AVG			8,6	5,8	4	3	+10

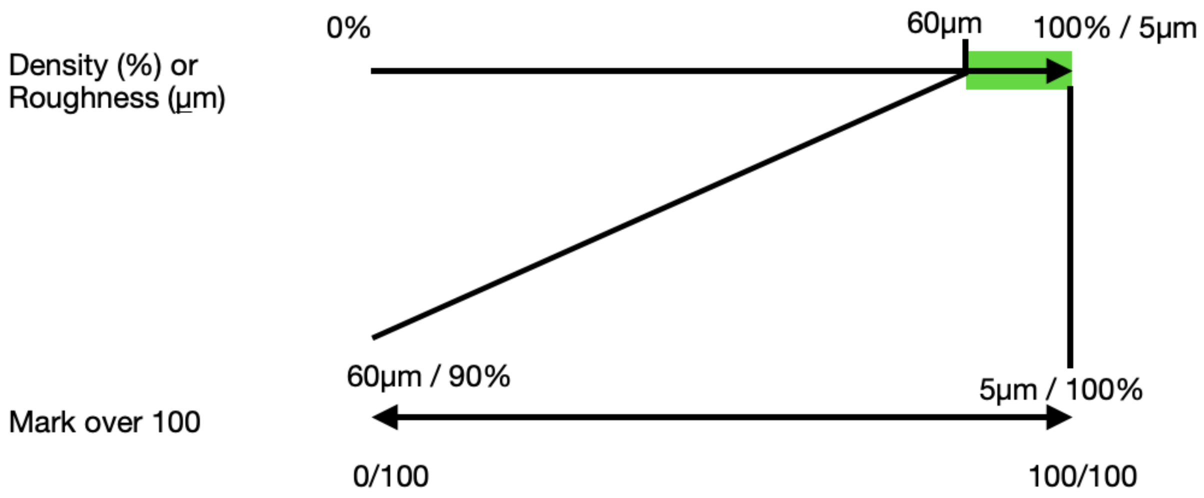
We can notice the number of iteration required to reach the optimized set of parameters highly depends on the initial spread of observations. If observations are well distributed over the search domain, the number of iteration to reach the goal is limited, even if the search domain is large. However, if the initial observations are concentrated in a specific area, the algorithm needs way more iterations.
As we do not know the true interaction between each parameters we cannot give a specific number of iteration for our application and we cannot be sure that our combination is the best. However as the repartition of the different initial observations are spread according a Doehlert design of experiments, the repartition is optimized to explore the whole search domain.



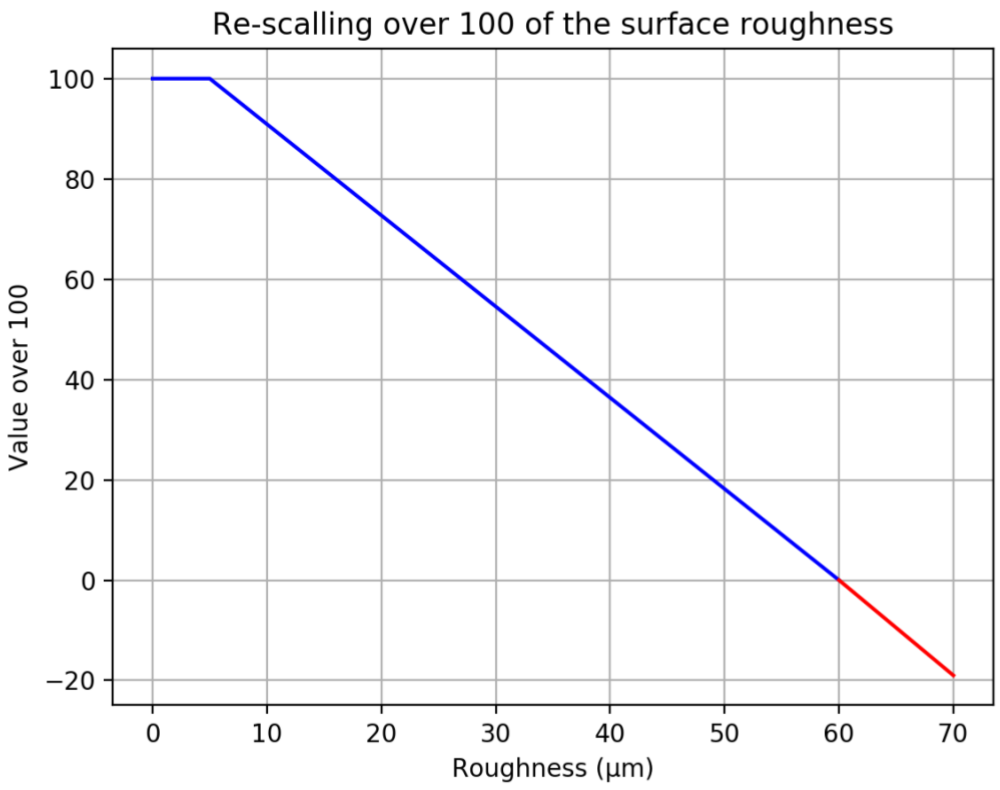
3D representation of initial observations for density optimization

3) Objective functions :

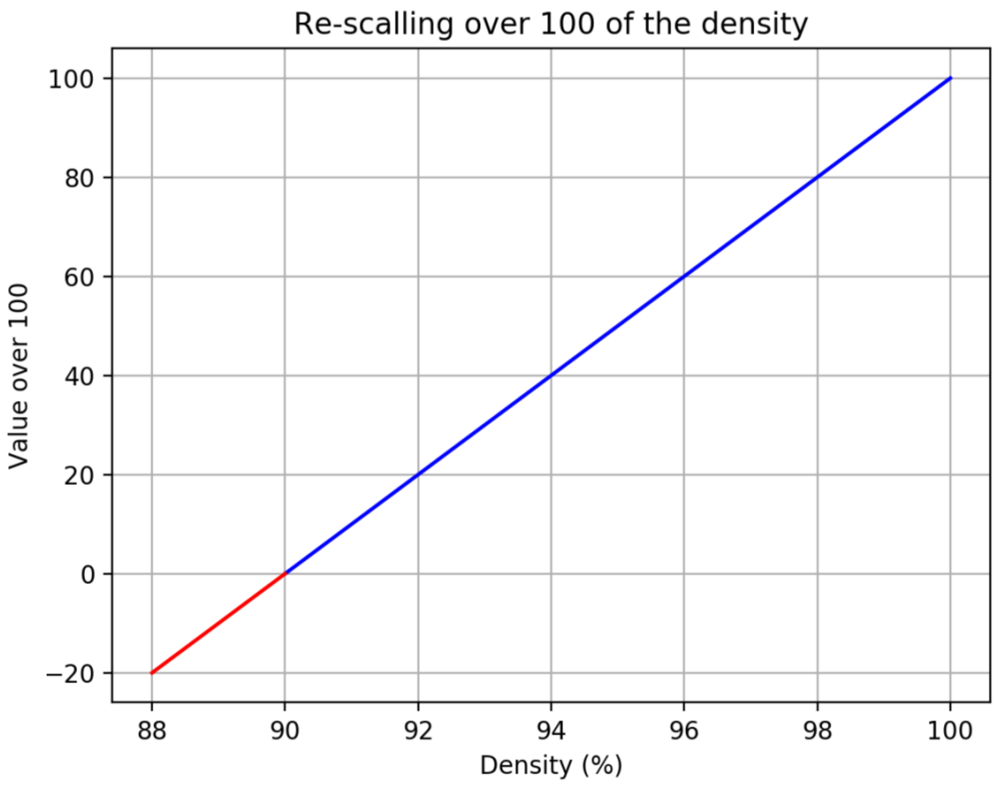
An objective function needs to be defined as we work on a optimization problem. The goal of this optimization is to maximize the objective function (either the function which is linked with the density or the functions which is linked with the opposite of the surface roughness, up-skin or down-skin).
A rating scale has been created : It is a question of anticipating what will be the interval where the majority of measurements will take place (interval in green on the diagram below) and rescaling it out of 100. It can possibly be negative.



Here is an example for roughness (upper or lower skin). We consider that most roughness measurements take place between 5µm and 60µm (based on Patrick's thesis [6]). If the roughness is less than 5µm (very unlikely given the process) we still give a score of 100/100 and if the roughness is greater than 60µm we give a negative score.



The same principle is used for density :



The rescaling for density is based on the Wen Hao’s work for MCAM in 2020 [7].

IV. Results :

As a reminder, the algorithm is run with Ti-6Al-4V alloy samples printed with an EOS M290 machine with a layer thickness of 30 µm.
The following results are obtained with the parameters highlighted in the chapter :
“ Strategy : How to choose the proper set of utility function ? ”.
10 different utility functions are used :

UCB		POI		EI	
κ	1,96	ξ	0,01	ξ	0,01
	2,5		0,1		0,1
	5		1,5		1,5
	10				

To get the next sets of parameters to test, three Bayesian algorithms are run. One to optimize the density, one to optimize the up-skin surface roughness and eventually, one for the down-skin surface roughness. Then samples are printed with the corresponding core and contour parameters.

1) Initialization Step :

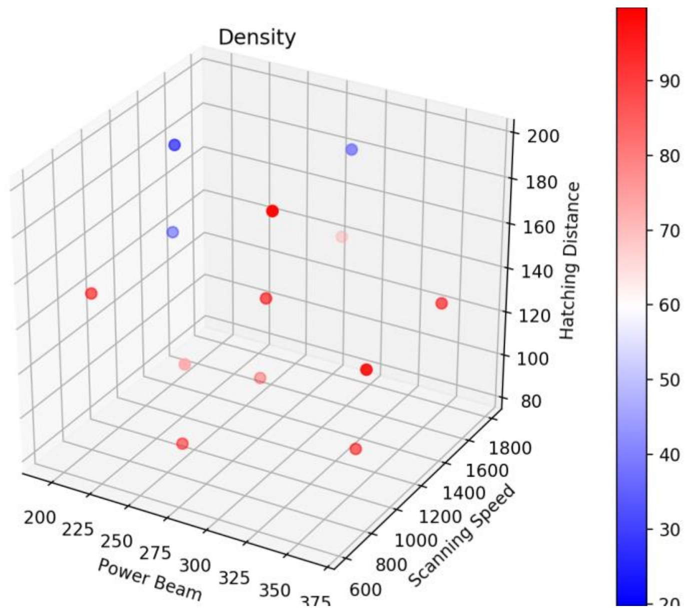
During the initialization step, we need to fill the algorithm with knowledge (more specifically, we fill the three algorithms).
We use the data provided by the Patrick’s thesis [6] for the surface roughness (contour parameters). Regarding the density, we use data from Wen Hao’s work [7].

Here is the paramaters and the corresponding results :

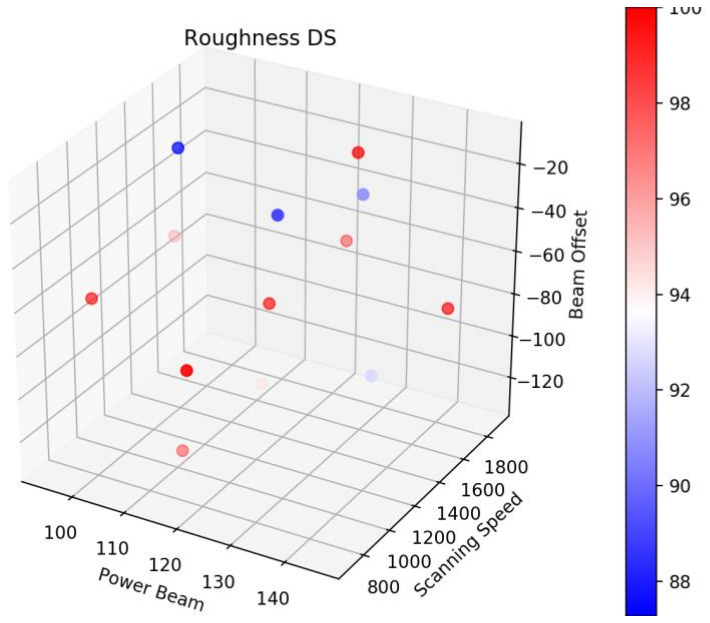
Initialization											
Core				Up_Skin				Down_Skin			
Power beam W	Scan Speed mm.s-1	Hatch Dist um	Dens %	Power beam W	Scan Speed mm.s-1	Beam Offset um	Ra um	Power beam W	Scan Speed mm.s-1	Beam Offset um	Ra um
280	1200	140	99.836	230	1200	-70	16.5	120	1300	-70	17,69
280	1800	140	97.406	230	1900	-70	24.5	120	1900	-70	34
366,6	1500	140	99.698	273	1550	-70	21	146	1600	-70	28,5
308,9	1500	197,12	93.63	244	1550	-10	25	129	1600	-10	25,5
280	600	140	97.318	230	500	-70	6.75	120	700	-70	24,5
193,4	900	140	99.836	187	850	-70	14.75	94	1000	-70	28
251,1	900	82,88	99.698	216	850	-130	19	111	1000	-130	23
193,4	1500	140	91.976	187	1550	-70	18.5	94	1600	-70	33
251,1	1500	82,88	99.982	216	1550	-130	19.25	111	1600	-130	20
366,6	900	140	99.792	273	850	-70	12	146	1000	-70	20,5
337,7	1200	82,88	99.956	259	1200	-10	19.25	137	1300	-10	21,5
308,9	900	197,12	99.852	244	850	-10	17	129	1000	-10	25
222,3	1200	197,12	92.47	201	1200	-10	17	103	1300	-10	26
280	1200	140	99.89								
280	1200	140	99.922								

We can notice in the “core” column (density) that the first set of parameters and the two last ones are the same. As we cannot give different results for a specific set of parameters, an average is done and provided to the algorithm.

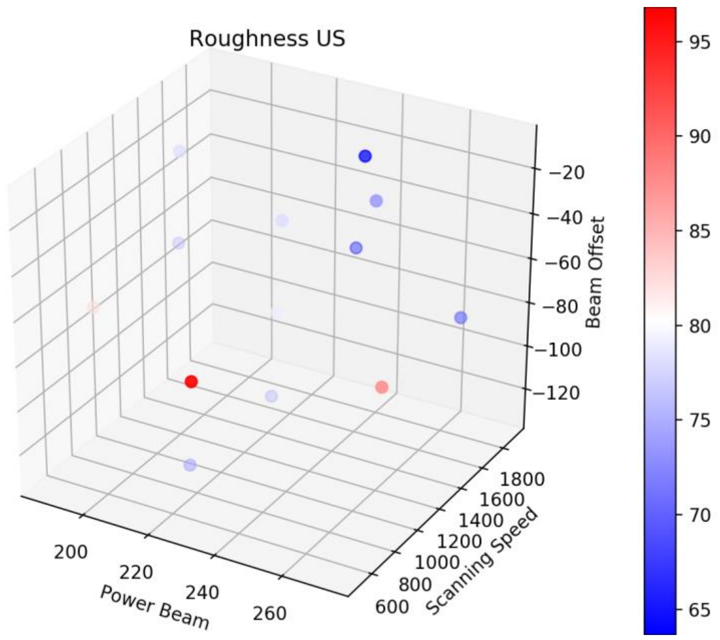
These data are rescaled as mentioned in the "Strategy : objective function" chapter and represented in the following 4D representations :



4D representation of the core sets of parameters



4D representation of the down-skin sets of parameters



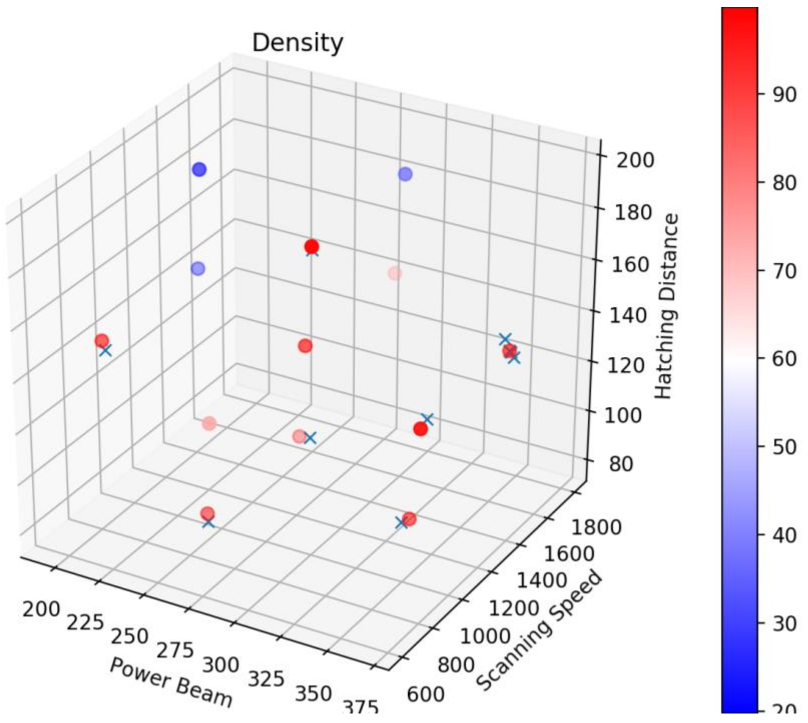
4D representation of the up-skin sets of parameters

2) First Step :

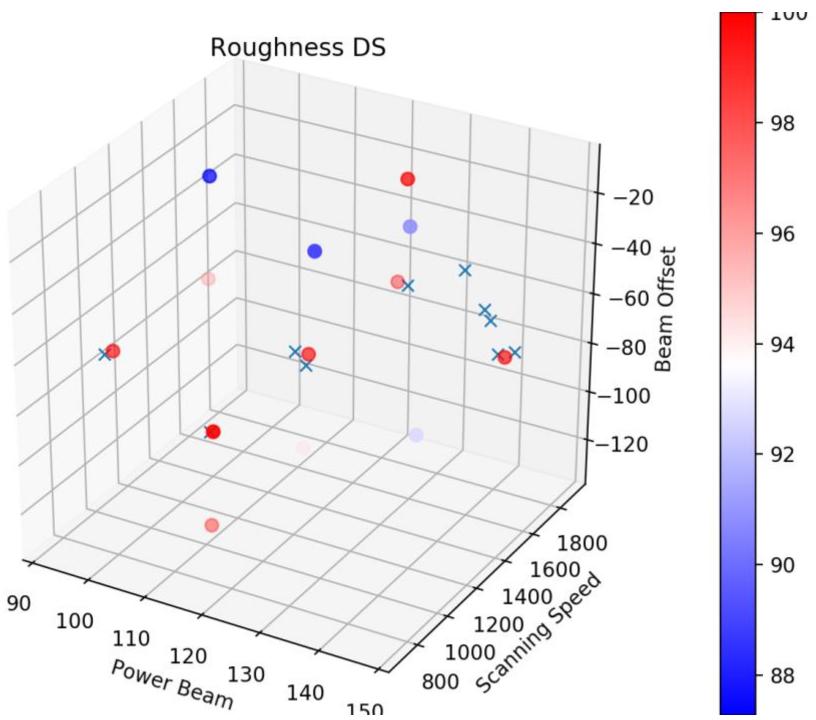
Algorithms are run for the next sets of parameters to test, here are the sets :

	Core			Up_Skin			Down_Skin		
sample	Power beam W	Scan Speed mm.s-1	Hatch Dist um	Power beam W	Scan Speed mm.s-1	Beam Offset um	Power beam W	Scan Speed mm.s-1	Beam Offset um
C1	366	1503,6	139,9	228.2	500	-73,1	142.4	1600	-53,2
C2	195,6	896,3	137,1	225.9	1202.1	-70,4	117.6	1301.8	-70,4
C3	251,4	899,5	79,7	229.0	507.1	-70	138.9	1600	-39,8
C4	364,2	1496,2	144,3	234.7	501.5	-72,4	143.4	1600	-56,8
C5	369,5	1494,5	138,5	273.3	849.5	-68,2	144.8	1602.3	-69,7
C6	333	1204,3	80,4	228.9	503.5	-70,8	119.4	703.0	-70,8
C7	367,1	1502,4	139,5	248.3	854.0	-12,2	147.7	1598.7	-66,8
C8	257,1	1501,3	83,7	271.7	854.5	-71,4	92.3	1002.3	-72,5
C9	369,8	897,6	144,5	199.7	1201.6	-8,1	121.9	1897.5	-70,2
C10	309	899,7	196,1	231.5	500.8	-68,9	119.6	1296.5	-74,5

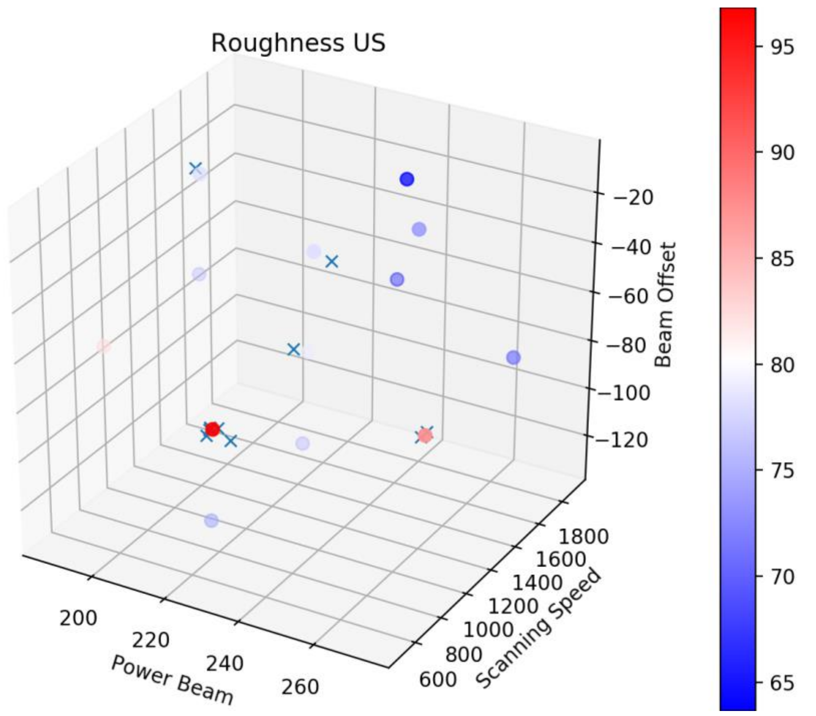
These sets are quite close to the initial observations and it seems weird. However, most of the time it is the case with multiple parameters optimization and at the second step, it becomes more widespread. 3D representations (with blue crosses) are on top of the previous 4D representations. The crosses represent the sets of parameters obtained after the first iteration.



Core parameters

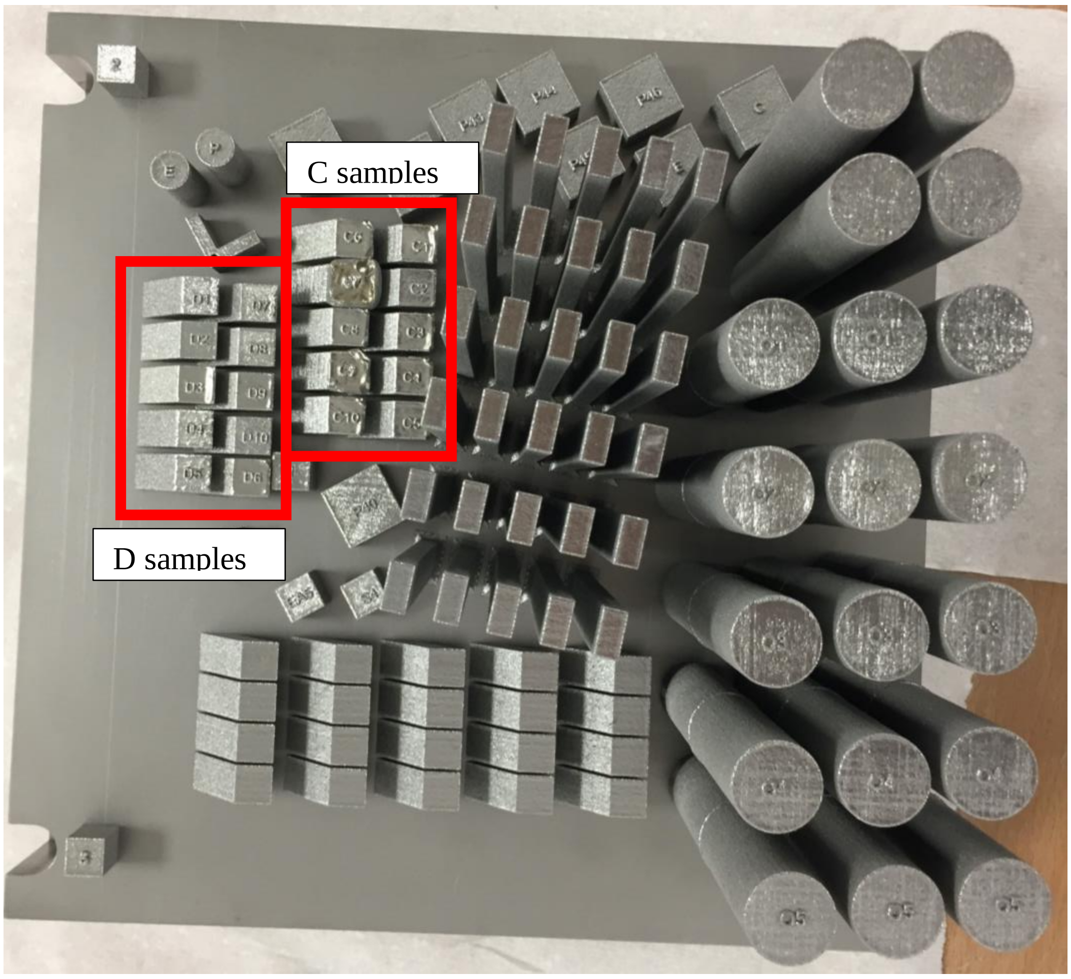


Down-skin contour parameters



Up-skin contour parameters

Instead of printing only ten samples (C1 to C10), we printed ten more (D1 to D10) with the core parameters of the corresponding C sample and the same contour parameters than the core ones. We do not use these D samples in this project but it will be linked to this project in a future study about the influence of the core parameters on the surface roughness.



Samples of the 1st step

Unfortunately, I did not make all measurements because of the Covid19 pandemic. Indeed, Melbourne is hard-hit by a second wave and to face this crisis a 3 and then a 4 stage lockdown is set from beginning of July to beginning of September. Consequently, from beginning of August I cannot access to the lab anymore. So this project is in stand-by until the end of lockdown.

3) Next Step to pursue the optimization process :

Bayesian optimization seems to be a great hope to reach an optimized set of parameters. Consequently, this project needs to be continued after the lockdown. Two strategies can be follow :

- Continue to print ten samples per batch and per platform and make a campaign of measurements. This strategy is the most efficient regarding the number of printing needed to reach the optimized set of parameters. However it requires a lot of time between two printing to make all these measurements. Indeed, it needs to make twenty surface roughness measurements and then the density measurements. The density measurements are quite time consuming even with the automatic polishing machine.
- Make a batch of only one or two samples at each printing. All the following steps are the same but obviously it is easier to conduct measurements during another project.

Depending of the chosen strategy, an optimized set of parameters should be reached sooner or later. However I cannot predict how many steps it requires as I did not run the algorithm further than the first iteration.

V. Conclusion :

To conclude this final report, set an SLM process is a complex task to perform. If this manufacturing process can be very useful in many fields including aeronautics, the process set of parameters is one of the hot button. Design of Experiences based algorithms are very useful to reach a virtual optimized set of parameters but do not take into account all of the interactions between each parameters and physical phenomena. This leads to consider an alternative way to optimize with a black box function approach. The use of Bayesian algorithms are not only innovative for the SLM domain but could improve the traditional way to optimize SLM parameters by returning consistent results within a limited number of trials. The Covid19 pandemic slowed down this study and unfortunately I was not able to run the algorithm as I expected. However, I am confident that this process is relevant regarding the optimization of density and surface roughness.

References :

DoE based algorithms :

[1] Stéphane Vivier, Stratégies d'optimisation par la méthode des Plans d'Expériences, et Application aux dispositifs électrotechniques modélisés par Éléments Finis. Modélisation et simulation. Université des Sciences et Technologie de Lille - Lille I, 2002. Français.

Python Library :

[2] github.com/fmfn/BayesianOptimization

Utility functions :

[3] Eric Brochu, Vlad M. Cora and Nando de Freitas, A Tutorial on Bayesian Optimization of Expensive Cost Functions, with Application to Active User Modeling and Hierarchical Reinforcement Learning, 2010

[4] D. Lizotte. Practical Bayesian Optimization. PhD thesis, University of Alberta, Edmonton, Alberta, Canada, 2008.

[5] Lizotte et al., D. Lizotte, T. Wang, M. Bowling, and D. Schuurmans. Automatic gait optimization with Gaussian process regression. In IJCAI, 2007.

Surface Roughness data :

[6] "Patrick's Thesis" : [Zhuoer Chen, 2018] Surface roughness of Selective Laser Melted Ti-6Al-4V alloy, PhD thesis, Monash University, 2018

Density data :

[7] Wen Hao, Process Optimization Milestone Report, MCAM, Monash University, 2020