HPC ESIR PROJECT 2024

List of definitions to be explained:

- optimitation problem
  Given a function $f : x \in X \longrightarrow f(x) \in \mathbb{R}$, find the point $\hat{x}$ such that $f(\hat{x}) < f(x)$ for each $x \in X$.
  f = objective function
- constraints on definition domain X
  For many real-life optimitation problems, the definition domain X is subject to contraints. In many situations, constraints may be included in the objective function.
- gradient descent for differentiable functions
  If the objective function f is differentiable, and we have a point $\bar{x} \in X$, the opposite direction of the gradient $f'(\bar{x})$ shows us the way towards points of X where $f(x)$ is smaller.
- differently from the method above, heuristics are not based on math proofs; they are rather inspired by nature or animal behavior.
- Simulated Annealing is a heuristic proposed in 1983 which is inspired by the physical annealing process occurring when the slow decrease of temperature transforms a liquid into a solid, by forming a typical crystal structure (corresponding to low energy).
- One of the main drawbacks of Simulated Annealing: at low temperature, the majority of the generated points are rejected, so the search may get stuck.
- Possible approach to overcome this drawback: run the simulation on a GPU! Instead of generating one point per time, we can generate hundreds of thousands points per time, increasing the chances that at least one is accepted.
- Even better: we could use several GPUs in one shot!

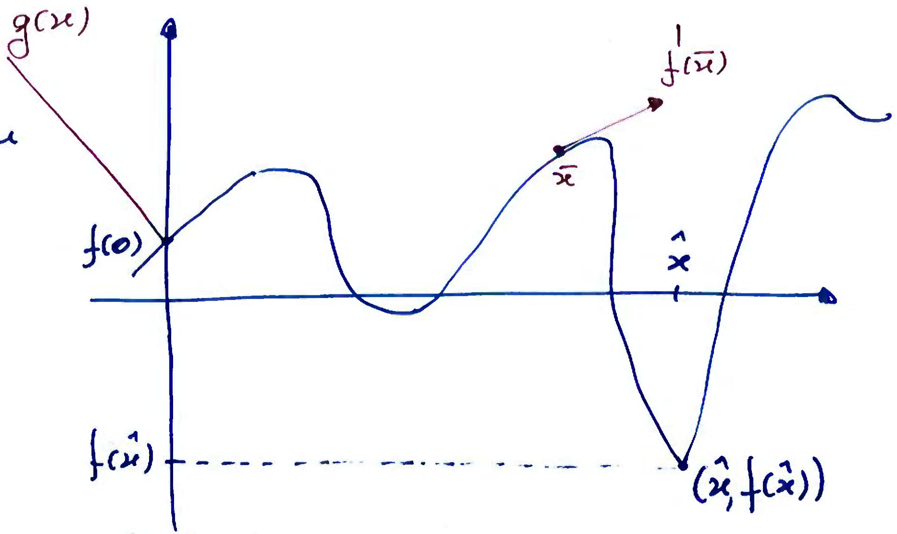
Explainations

Optimitation problem

In this example,

$X = \mathbb{R}$.

But suppose we have the constraint $x > 0$.



We can replace the original objective function with:

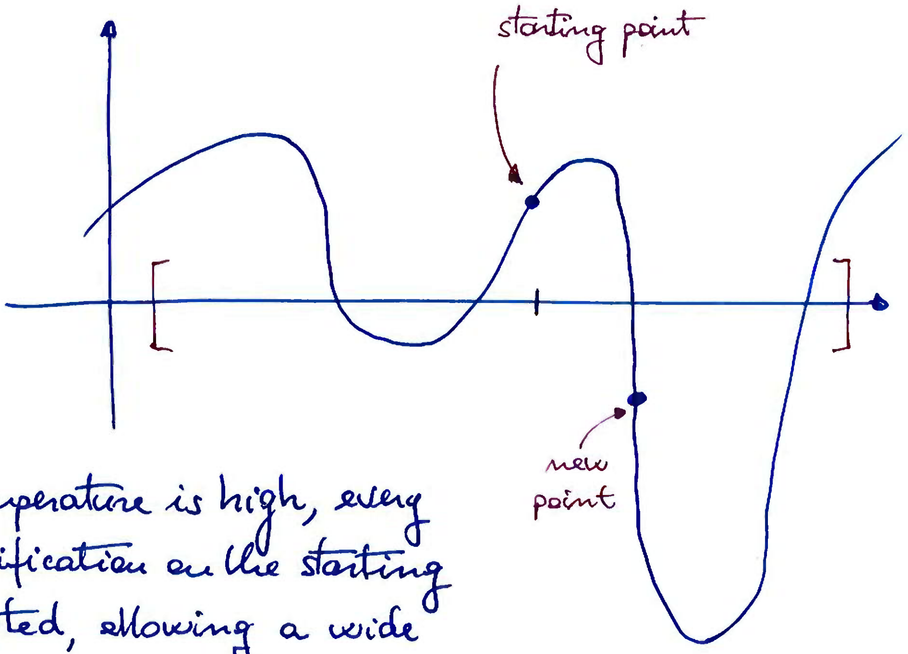
$$g(x) = \begin{cases} f(x) & \text{if } x \geq 0 \\ f(0) - x & \text{if } x < 0 \end{cases}$$

What is the graphic of g?

In our example, the gradient $f'(x)$ is the tangent on the graphic of f in x. There are two main problems in implementing a gradient descent:

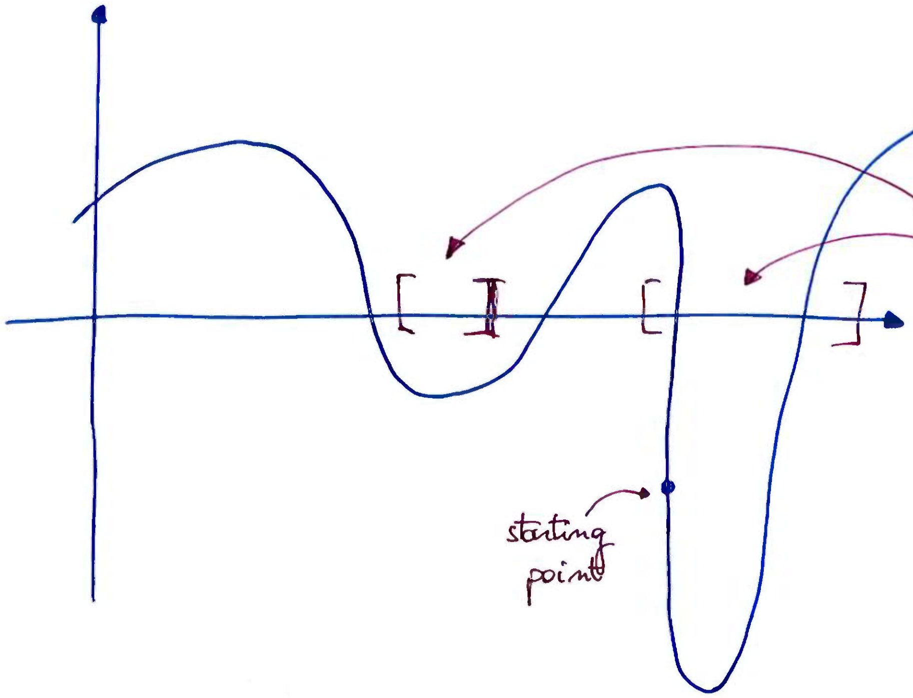
1) we do not know how far from $\bar{x}$ we need to go to catch the minimum;
2) when we do catch the minimum, we don't know if it's the global one.

How to run the Simulated Annealing on an small example?



When the temperature is high, every random modification on the starting point is accepted, allowing a wide exploration.

When the temperature is decreased, the search focuses in the areas where there are more chances to find the global optimum:



At lower temperatures, points that worsen too much the objectiv function are not anymor eccepted.

There are no guarantees however that we will find the global optimum. When the function is so regular, we can consider to use a local search (such as the gradient descent method) to improve (some of) the generated points.

So, what is our project about?

First of all, what is not about:

- we will not develop heuristics
- we will not implement a gradient descent
- we will not work in a continous space X

We will:

- work with a C type defining a bit string of any length.

```
struct bits
{
  size_t nbytes;
  short ignore;
  char *byte;
}
typedef struct bits
                    bits_t;
```

- develop functions for the manipulation of these strings of bits (typical manipulations performed by heuristics).
- each of you will have a different function to develop, randomly assigned.
- each of you will have to write it in C and test it at first, and only after to start a CUDA implementation.
- your functions will have to be written in a lowlevel fashion, we prefer speed to modulation.
- your functions will have to verify that the input arguments are correct, as long as this is possible, through the use of "assert".