39 Zipf's law applied to a source code.

The first thing we need to consider if we want to evaluate a source code is "which tokens we will take into account". For example, we can take just the keywords, but it will be better to take into account tokens like "new line", "end of line", or the indentations.

Even more, is important to decide if we want to count the words inside the comments. If we do, it the complexity will increase, because we can find source-code chunks inside the comments.

```
/*
  public  functionX() {}
*/

public void functionX() {
}
```

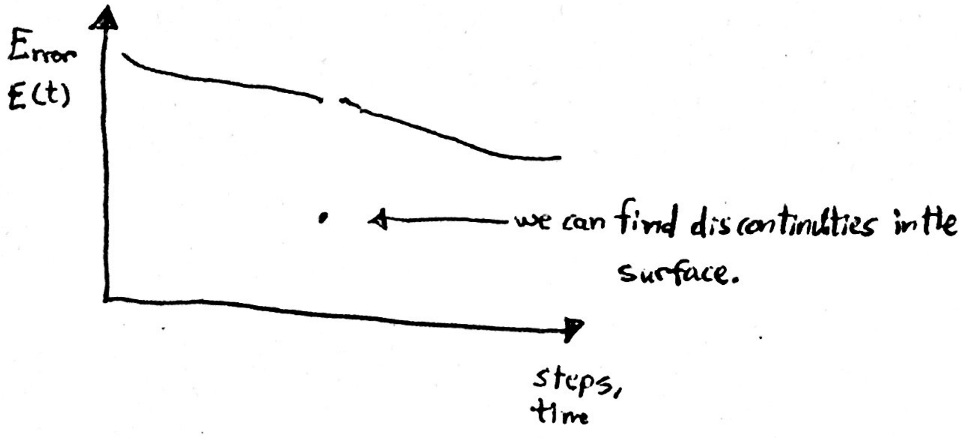
this pieces of code must be different. for the metric we will use.

The complexity of this task is firstly to decide which metric we will use, the more detailed is our metric the more complicated will be a parser to count each token.

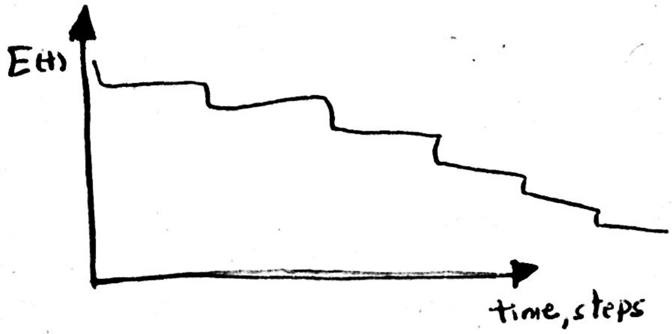
A good data to evaluate could be any open source code available online; e.g. "Linux source code".

40. Gradient descent drawbacks:

1) Isolated points:

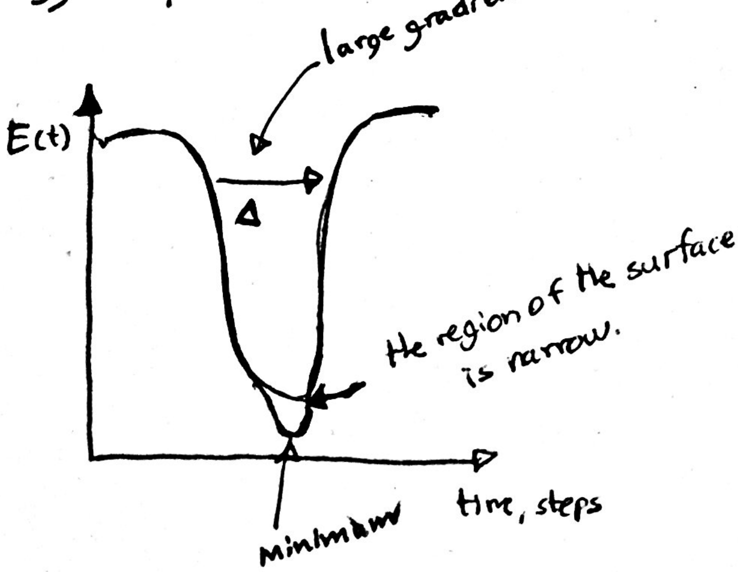


2) Flat plateaus



A flat plateaus means that the slope in some regions of the surface is zero or close to zero, which result in slow progess,

3) steep valleys

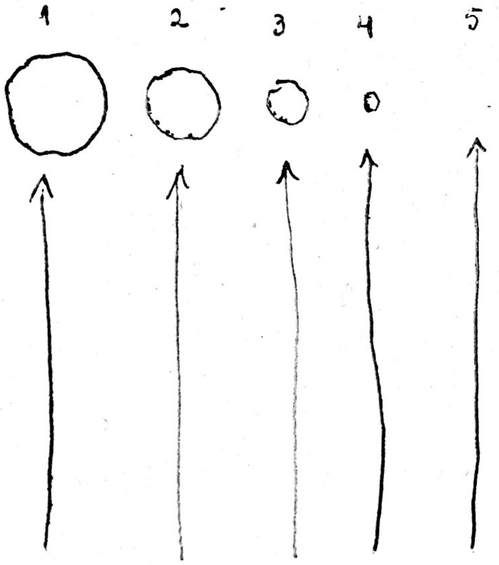


We can find some valleys, (narrow regions), and a large gradient, then we will skip the minimum.

44

After the discovery of a source of food ~~and~~ the concentration of pheromones will increase in the path used for that discovery.

The evaporation of that ~~source~~ path is necessary after the source of food is exhausted, or when a better path has been discovered.



[illegible]

In the fifth step the concentration of pheromones is needed to decrease, because there is no more source of food.