Advanced Optimization

Master AIC - Paris Saclay University

Exercices - Stochastic Continuous Optimization

Anne Auger and Dimo Brockhoff
anne.auger@inria.fr

I Pure Random Search (PRS)

We consider the following optimization algorithm.

[Objective: minimize $f : [-1, 1]^n \to \mathbb{R}$
X_t is the estimate of the optimum at iteration t
`Input` $(U_t)_{t\geq 0}$ independent identically distributed each $U_t \sim \mathcal{U}_{[-1,1]^n}$ (unif. distributed in $[-1, 1]^n$)]
`1. Initialize` $t = 0$, $X_0 = U_0$
`2. while not terminate`
`3.` $t = t + 1$
`4.` If $f(U_t) \leq f(X_{t-1})$
`5.` $X_t = U_t$
`6.` Else
`8.` $X_t = X_{t-1}$

1. Show that for all $t \geq 0$
$$f(X_t) = \min\{f(U_0), \ldots, f(U_t)\}$$

2. We consider the simple case where $f(x) = \|x\|_\infty$ (we remind that $\|x\|_\infty := \max(|x_1|, \ldots, |x_n|)$). Show the convergence in probability of the PRS algorithm towards the optimum of f, that is prove that for all $\epsilon > 0$
$$\lim_{t\to\infty} \Pr\left(\|X_t\|_\infty \geq \epsilon\right) = 0$$
Hint: Prove and use the equality
$$\{\|X_t\|_\infty \geq \epsilon\} = \cap_{k=0}^{t} \{\|U_k\|_\infty \geq \epsilon\}$$

3. Let $T_\epsilon = \inf\{t | X_t \in [-\epsilon, \epsilon]^n\}$ (with $\epsilon > 0$) be the first hitting time of $[-\epsilon, \epsilon]^n$.
Show that T_ϵ follows a geometric distribution with a parameter p that we will determine.
Deduce the expected value of T_ϵ, that is the expected hitting time of the PRS algorithm.

4. When we implement a DFO optimization algorithm, the cost of the algorithm is the number of calls to the objective function. Write a pseudo-code of the PRS algorithm where at each iteration the objective function f is called only once.

II Adaptive step-size algorithms

Below is an exercice for students who never experienced stochastic optimization algorithms before.
They are asked to run some experiments in Matlab, make some observations and then understand what they have observed.
Given that you all already experienced stochastic algorithms before, this exercice will be used to refresh your mind ... and in particular you are asked to do the exercice WITHOUT the computer, i.e. you have to read the question and think about the answer without the help of the computer ...

We are going to test the convergence of several algorithms on some test functions, in particular on the so-called sphere function

$$f_{\text{sphere}}(\mathbf{x}) = \sum_{i=1}^{n} \mathbf{x}_i^2$$

and the ellipsoid function

$$f_{\text{elli}}(\mathbf{x}) = \sum_{i=1}^{n} (100^{\frac{i-1}{n-1}} \mathbf{x}_i)^2 \quad .$$

1. What is the condition number associated to the Hessian matrix of the functions above? Are the functions ill-conditioned?

2. Use Matlab to implement the functions. We can create two functions `fsphere.m` and `felli.m` that take as input a vector $\mathbf{x}$ and returns $f(\mathbf{x})$.

(1+1)-ES with constant step-size

The $(1+1)$-ES algorithm is on of the simplest stochastic search method for numerical optimization. We will start by implementing a $(1+1)$-ES with constant step-size. The pseudo-code of the algorithm is given by

```
Initialize x ∈ R^n and σ > 0
while not terminate
    x' = x + σ N(0, I)
    if f(x') ≤ f(x)
        x = x'
```

where $\mathcal{N}(\mathbf{0}, \boldsymbol{I})$ denotes a Gaussian vector with mean $\mathbf{0}$ and covariance matrix equal to the identity.

1. Implement the algorithm in Matlab. You can write a function that takes as input an initial vector $\mathbf{x}$, an initial step-size σ and a maximum number of function evaluations and returns a vector where you have recorded at each iteration the best objective function value.

2. Use the algorithm to minimize the sphere function in dimension $n = 5$. We will take as initial search point $\mathbf{x}^0 = (1, \ldots, 1)$ [`x=ones(1,5)`] and initial step-size $\sigma = 10^{-3}$ [`sigma=1e-3`] and stopping criterion a maximum number of function evaluations equal to 2×10^4.

3. Plot the evolution of the function value of the best solution versus the number of iterations (or function evaluations). We will use a log scale for the y-axis (`semilogy`).

4. Explain the three phases observed on the figure.

(1+1)-ES with one-fifth success rule

To accelerate the convergence, we will implement a step-size adaptive algorithm, i.e. σ is not fixed once for all. The method to adapt the step-size is called one-fifth success rule. The pseudo-code of the $(1+1)$-ES with one-fifth success rule is given by:

```
Initialize x ∈ R^n and σ > 0
while not terminate
    x' = x + σ N(0, I)
    if f(x') ≤ f(x)
        x = x'
        σ = 1.5 σ
    else
        σ = (1.5)^{-1/4} σ
```

5. Implement the (1+1)-ES with one-fifth success rule and test the algorithm on the sphere function $f_{\text{sphere}}(x)$ in dimension 5 ($n = 5$) using $\mathbf{x}^0 = (1, \ldots, 1)$, $\sigma_0 = 10^{-3}$ and as stopping criterion a maximum number of function evaluations equal to 6×10^2. Plot the evolution of the square root of the best function value at each iteration versus the number of iterations. Use a logarithmic scale for the y-axis. Compare to the plot obtained on Question 3. Plot also on the same graph the evolution of the step-size.

6. Use the algorithm to minimize the function f_{elli} in dimension $n = 5$. Plot the evolution of the objective function value of the best solution versus the number of iterations. Why is the $(1+1)$-ES with one-fifth success much slower on f_{elli} than on f_{sphere} ?

7. Same question with the function

$$f_{\text{Rosenbrock}}(x) = \sum_{i=1}^{n-1} (100(x_i^2 - x_{i+1})^2 + (x_i - 1)^2) \ .$$

8. We now consider the functions, $g(f_{\text{sphere}})$ and $g(f_{\text{elli}})$ where $g : \mathbb{R} \to \mathbb{R}, y \mapsto y^{1/4}$. Modify your implementation in Questions 5 and 6 so as to save at each iteration the distance between $\mathbf{x}$ and the optimum. Plot the evolution of the distance to the optimum versus the number of function evaluations on the functions f_{sphere} and $g(f_{\text{sphere}})$ as well as on the functions f_{elli} and $g(f_{\text{elli}})$. What do you observe? Explain.

II On Linear Convergence

For a deterministic sequence x_t the linear convergence towards a point x^* is defined as:

The sequence $(x_t)_t$ convergences linearly towards x^* if there exists $\mu \in (0, 1)$ such that

$$\lim_{t \to \infty} \frac{\|x_{t+1} - x^*\|}{\|x_t - x^*\|} = \mu \tag{1}$$

The constant μ is then the convergence rate.

We consider a sequence $(x_t)_t$ that converges linearly towards x^*.

1. Prove that (1) is equivalent to

$$\lim_{t \to \infty} \ln \frac{\|x_{t+1} - x^*\|}{\|x_t - x^*\|} = \ln \mu \tag{2}$$

2. Prove that (2) implies

$$\lim_{t \to \infty} \frac{1}{t} \sum_{k=0}^{t-1} \ln \frac{\|x_{k+1} - x^*\|}{\|x_k - x^*\|} = \ln \mu \tag{3}$$

HINT: Use Cesàro means

3. Prove that (3) is equivalent

$$\lim_{t\to\infty} \frac{1}{t} \ln \frac{\|x_t - x^*\|}{\|x_0 - x^*\|} = \ln \mu \tag{4}$$

For a sequence of random variables $(x_t)_t$. We define linear convergence by either considering the expected log progress, that is the sequence converges linearly if

$$\lim_{t\to\infty} E\left[\ln \frac{\|x_{t+1} - x^*\|}{\|x_t - x^*\|}\right] = \ln \mu \ ,$$

or, in order to consider the almost sure behavior (related to a single realization of an algorithm) we introduce the definition of linear convergence of a sequence of random variable as

$$\lim_{t\to\infty} \frac{1}{t} \ln \frac{\|x_t - x^*\|}{\|x_0 - x^*\|} = \ln \mu \text{ a.s.} \tag{5}$$

This will be illustrated as the log-distance to the optimum decreases to minus infinity as $\ln \mu \times t$, that is you observe asymptotically a line if you plot the convergence using a log-scale for the y-axis.