Gradient Descent

Hoàng Anh Quân

Ngày 18 tháng 4 năm 2019

Hanoi University of Science - VNU group

Table of contents

Bài toán mở đầu

Bài toán mở đầu

Cho hàm số $f : D \to \mathbb{R}$ với $D \subset \mathbb{R}^n$. Tìm giá trị nhỏ nhất[1] của hàm số f trên tập D (ta thường xét $D = \mathbb{R}^n$).

[1] trường hợp tìm giá trị lớn nhất được giải quyết một cách tương tự

Cách tiếp cận đầu tiên

Tại một điểm $x = x_k$, ta tìm một điểm $x = x_{k+1}$, là một trong $2n$ **điểm δ-lân cận** của x_k, thỏa mãn $f(x_{k+1}) < f(x_k)$ $(\star)$.
Nếu không có điểm x_{k+1} thỏa mãn, xét $\delta' = \delta/2$ và quay trở lại bước $(\star)$.

Với $x = (x_1, x_2, \ldots, x_n)$, ta định nghĩa $2n$ điểm δ-lân cận của x là tập hợp các điểm

$$S = \{(x_1 \pm \delta, x_2, \ldots, x_n), \ldots, (x_1, x_2, \ldots, x_n \pm \delta)\}$$

Thuật toán Gradient Descent

Định nghĩa gradient vector

Cho hàm số $f : \mathbb{R}^n \to \mathbb{R}$ khả vi theo mọi biến. Khi đó gradient vector của hàm số f tại x_0 có giá trị bằng

$$\nabla f(x_0) = \left(\frac{\partial f}{\partial x_1} \cdots \frac{\partial f}{\partial x_n} \right)^\top$$

Tính chất quan trọng

Cho hàm số $f : D \to \mathbb{R}$ khả vi theo mọi biến.

- Theo công thức khai triển Taylor, ta có:

$$f(x + \alpha p) = f(x) + \alpha p^\top \nabla f(x) + O(\alpha^2)$$

với $x, p \in \mathbb{R}^n$.
Trong trường hợp $\nabla f(x) \neq 0$ ta chọn $p = -\nabla f(x)$ thì $p^\top \nabla f(x) < 0$. Khi đó, **tồn tại** $\alpha_0 > 0$ sao cho

$$f(x + \alpha p) < f(x) \ \forall \alpha \in (0, \alpha_0)$$

- Nếu x là một điểm cực trị địa phương của hàm số f thì $\nabla f(x) = 0$.

Thuật toán Gradient Descent

1. Chọn một điểm khởi đầu $x_0 \in D$, gán giá trị $k = 0$, ϵ đủ nhỏ.
2. Xét giá trị $\nabla_k = \|\nabla f(x_k)\|$,
   - nếu $\nabla_k > \epsilon$, chọn α [2] sao cho

     $$f(x_{k+1}) < f(x_k) \text{ với } x_{k+1} = x_k + \alpha(-\nabla f(x_k))$$

     gán $k = k + 1$, rồi quay trở lại bước 2.
   - nếu $\nabla_k \leq \epsilon$, thì thuật toán kết thúc. Giá trị tối ưu đủ tốt tìm được là $f(x_k)$.

[2] sự tồn tại của α được chỉ ra ở slide trước

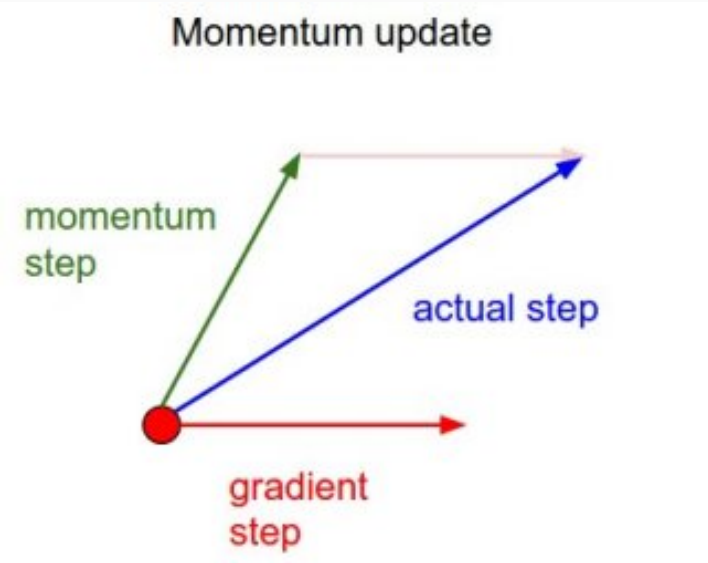
Các cách chọn bước nhảy α

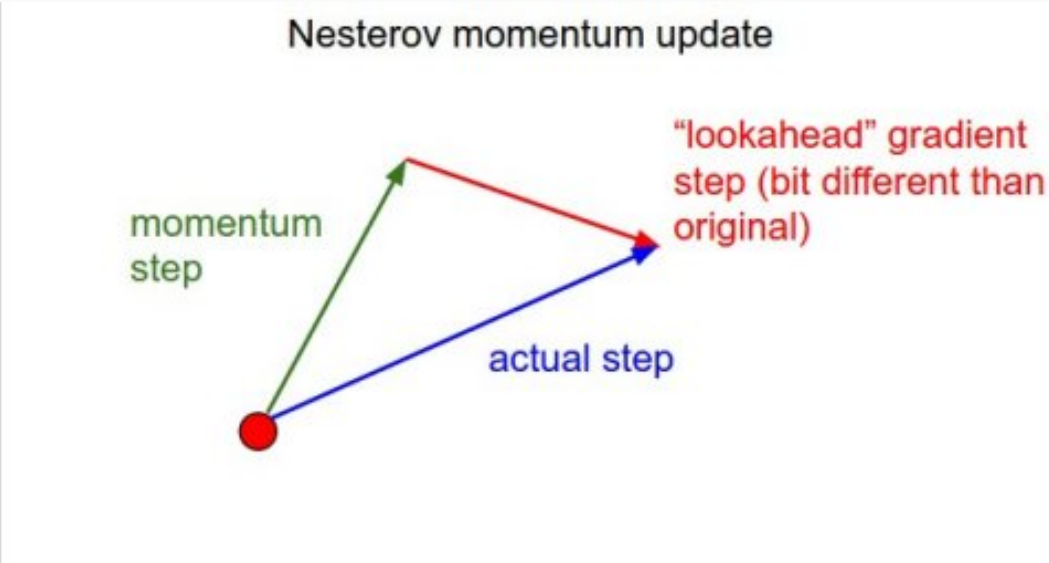
1. Chọn cố định một giá trị α_0 đủ nhỏ,
2. Giảm dần giá trị α (theo hàm số mũ) theo thời gian,
3. Ở mỗi bước, chọn α tối thiểu hóa hàm mục tiêu[3] $f(x_k + \alpha(-\nabla f(x_k))$.

[3] cách này không thực tế, vì trạng thái ở mỗi bước trong thuật toán là tạm thời

Cải tiến thuật toán

Momentum Gradient Descent





Stochastic Gradient Descent

Stochastic gradient descent

Consider minimizing an average of functions

$$\min_x \frac{1}{m} \sum_{i=1}^m f_i(x).$$

As $\nabla \sum_{i=1}^m f_i(x) = \sum_{i=1}^m \nabla f_i(x)$, gradient descent would repeat

$$x^{(k)} = x^{(k-1)} - t_k \cdot \frac{1}{m} \sum_{i=1}^m \nabla f_i(x^{(k-1)}), \quad k = 1, 2, 3, \ldots$$

In comparison, stochastic gradient descent or SGD (or incremental gradient descent) repeats:

$$x^{(k)} = x^{(k-1)} - t_k \cdot \nabla f_{i_k}(x^{(k-1)}), \quad k = 1, 2, 3, \ldots$$

where $i_k \in \{1, \ldots, m\}$ is some chosen index at iteration k.

Stochastic gradient descent

Two rules for choosing index i_k at iteration k:

- Randomized rule: choose $i_k \in \{1, \ldots m\}$ uniformly at random.
- Cyclic rule: choose $i_k = 1, 2, \ldots, m, 1, 2, \ldots, m, \ldots$

Randomized rule is more common in practice. For randomized rule, note that

$$\mathbb{E}[\nabla f_{i_k}(x)] = \nabla f(x),$$

so we can view SGD as using an unbiased estimate of the gradient at each step.

Main appeal of SGD:

- Iteration cost is independent of m (number of functions).
- Can also be a big savings in terms of memory usage.

Mini-batches Gradient Descent

Mini-batches

Also common is mini-batch stochastic gradient descent, where we choose a random subset $I_k \subseteq \{1, \ldots, m\}$ of size $|I_k| = b \ll m$ and repeat

$$x^{(k)} = x^{(k-1)} - t_k \cdot \frac{1}{b} \sum_{i \in I_k} \nabla f_i(x^{(k-1)}), \quad k = 1, 2, 3, \ldots$$

Again, we are approximating full gradient by an unbiased estimate

$$\mathbb{E}\left[\frac{1}{b} \sum_{i \in I_k} \nabla f_i(x)\right] = \nabla f(x).$$

Using mini-batches reduces the variance of our gradient estimate by a factor $1/b$, but is also b times more expensive.

Sử dụng Python để viết thuật toán

Một số câu lệnh quan trọng

```
def safe(f, case = 'min'):
  """
  return a new function that's the same as f,
  except that it outputs infinity
  whenever f produces an error
  """
  def safe_f(*args, **kwargs):
    try:
      return f(*args, **kwargs)
    except:
      if case == 'min':
        return np.inf
      elif case == 'max':
        return -np.inf
  return safe_f
```

Định nghĩa một hàm giống với f nhưng “không gặp lỗi" khi giá trị đầu vào ngoài miền xác định.

Một số câu lệnh quan trọng

```
class function:
  def __init__(self, var_range, func, min_val, min_state):
    self.range = var_range
    self.var_number = len(var_range)
    self.func = safe(func)
    self.min_val = min_val
    self.min_state = min_state

var_range = ( (-5,5), (-10,10) )
func = lambda x,y: x**2 - x*y + y**2
min_val = 0
min_state = (0,0)
func1 = function(var_range, func, min_val, min_state)
```

Một số câu lệnh quan trọng

```
def derivative(f, coor, i_th, h = 10**-6):
  (x,y) = coor
  if i_th < function.var_number:
    coor_start = (x,y)
    coor_end = [x,y]
    coor_end[i_th] += h
    return 1/h * (f(coor_end) - f(coor_start))
  else:
    print('Wrong i_th input')
    return
```

Chú ý: kết quả đầu ra của hàm này là **giá trị xấp xỉ** của đạo hàm tại một điểm. Điều đó làm khẳng định về sự tồn tại của α ở slide 6 **(Tính chất quan trọng)** không còn đúng.

Gradient Descent's scripts

Toàn bộ code của thuật toán[4] có thể tìm thấy ở đây.

[4] được viết bởi các thành viên tham dự Seminar Optima

Tham khảo thêm

Tham khảo thêm

1. Overview
2. Course
3. Visualizing
4. Interactive mode
5. Library: scikit-learn, keras, caffe, lasagne