On consensum

Giulio Galvan

26 settembre 2015

Algorithm 1 shows a Gauss-Seidel like decomposition method which optimizes a function $f(\cdot)$ w.r.t a block of variables i in a iteratively fashion. It can be shown that the algorithm converges if $\mathcal{L}_0$ is compact.

Algorithm 1: Gauss-Seidel like decomposition method

Data:
$x^0 \in \mathbb{R}^m$: candidate solution
$\lambda_i > 0, \quad i = 1, \dots, m$
1 $k \leftarrow 0$
2 **while** *stop criterion* **do**
3   **for** $i = 1, ..., m$ **do**
4         $d_i^k \leftarrow -\nabla_i f(x^k)$
5   compute α_i^k with line search
6         $x_i^{k+1} = x_i^k + \alpha_i^k d_i^k$
7   **end**
8     $k \leftarrow k + 1$
9 **end**

In the case fo RNNs we would, ideally, in a consesum inspired way, solve

$$\begin{aligned} \min_{W, W^i} \quad & f(W, W^i, i = 1 \dots, T) \\ \text{subject to} \quad & W_i = W, \quad i = 1, \dots, T \end{aligned},$$

considering only W^{rec} for easiness. The Lagrangian can then be written as:

$$\min_{\tilde{W}, W^i} g(\cdot) = f(\tilde{W}, W^i, i = 1 \dots, T) + \lambda(W^i - \tilde{W}) \tag{1}$$

The blocks of variables are, hence, $W^i, i = 1, ..., T$ where T is the length of the sequence, and $\tilde{W}$. However we cannot have T of such matrix in memory so we have to devise a modification of the algorithm like I tried to do in Algorithm 2

The idea is not to store all the matrices, computing gradients for each block separately but using an iteratively update rule, like is done at line 7.

Algorithm 2: Gauss-Seidel (proximal point)

```
Data:
W^0 ∈ ℝ^m: candidate solution
λ_i > 0,    i = 1, ..., m
k ← 0
while stop criterion do
    for i = 1, ..., m do
        d_i^k ← −∇_i g(x^k)
        compute α_i^k with line search
        W̃ = W_i^k + α_i^k d_i^k
        W_j^{k+1} = W̃,    j = 1, ..., i
    end
    k ← k + 1
end
```