2.5 Block Optimization Methods

Let $\mathbb{X}_1, \ldots, \mathbb{X}_p$ be particular sets and $f : \mathbb{X}_1 \times \ldots \times \mathbb{X}_p \to \mathbb{R}$ be a continuous function. We consider the following optimization problem:

$$\begin{aligned} \min_{x_1, \ldots, x_p} \quad & f(x_1, \ldots, x_p) \\ \text{subject to} \quad & x_1 \in \mathbb{X}_1, \ldots, x_p \in \mathbb{X}_p. \end{aligned} \tag{2.5.1}$$

In this section, we present two of the most popular numerical methods used to solve the optimization problem (2.5.1): the block coordinate descent method and the maximum block improvement method.

2.5.1 Block Coordinate Descent

The block coordinate descent (BCD) method is a well studied method [27, 28, 157] of solving problem (2.5.1), when $\mathbb{X}_j = \mathbb{R}^n_j$, for all $j = 1, \ldots, p$. The BCD method updates only one variable (or block) x_j while fixing all the other blocks. There are different rules for selecting which block is updated, and while a block is being updated the values of remaining blocks may or may not be up-to-date. The most popular rules are as follows:

1. **Jacobian rule:** at every iteration k, all p variables are updated in parallel; x_j is updated using the values of the remaining $(p-1)$ variables at iteration $j-1$. For example,
$$x_j^{(k)} \in \arg\min_{x_j \in \mathbb{X}_j} f\left(x_1^{(k-1)}, \ldots, x_{j-1}^{(k-1)}, x_j, x_{j+1}^{(k-1)}, \ldots, x_p^{(k-1)}\right), \tag{2.5.2}$$
for $j = 1, \ldots, p$. This can be computed in parallel.

2. **Gauss-Seidel rule:** at every iteration k, variables $x_1, \ldots, x_p$ are cyclically and sequentially selected to update as follows: x_j is updated on the $k-$th iteration using the most recent values of the previous $(p-1)$ variables. For example, on the $k-$th iteration
$$x_j^{(k)} \in \arg\min_{x_j \in \mathbb{X}_j} f\left(x_1^{(k)}, \ldots, x_{j-1}^{(k)}, x_j, x_{j+1}^{(k-1)}, \ldots, x_p^{(k-1)}\right), \tag{2.5.3}$$
for $j = 1, \ldots, p$. The expression in (2.5.3) uses $x_{j+1}^{(k-1)}, \ldots, x_p^{(k-1)}$ since they haven not been updated to $x_{j+1}^{(k)}, \ldots, x_p^{(k)}$ yet.

Algorithm 1 provides the framework of the BCD method, which allows each x_j to be updated by (2.5.2) or (2.5.3).

Algorithm 1 BCD method for solving (2.5.1)

Requires: $x^{(0)} = (x_1^{(0)}, \dots, x_p^{(0)}) \in \mathbb{X}_1 \times \dots \times \mathbb{X}_p$ (Initial point)
Returns: $x^{(k)} = (x_1^{(k)}, \dots, x_p^{(k)}) \in \mathbb{X}_1 \times \dots \times \mathbb{X}_p$
for $k = 1, 2, ...$ **do**
  for $j = 1, ..., p$ **do**
    $x_j^{(k)} \leftarrow$ Using (2.5.2) or (2.5.3)
  end
  if stopping criterion is satisfied **then**
    return $x^{(k)} = (x_1^{(k)}, \dots, x_p^{(k)})$
  end
end

The proof of convergence of the BCD method requires some assumptions (see the examples in [114, 166]). This method can be applied regardless of any convexity assumptions, as long as it is able to optimize over one block or variable while fixing the others [18]. Global convergence of the BCD method is often obtained for functions that are separated into sums of functions of single block variables [27, 173]. In particular, convergence is ensured if f has some type of convexity [157, 166, 172], otherwise, the search routine should be modified [28, 157]. If $x_j^{(k)}$ in Algorithm 1 is updated using the Gauss-Seidel rule and $x_j^{(k)}$ is unique, then convergence is guaranteed [18, 157]. If $p = 2$, then convergence of the BCD method does not require uniqueness of $x_j^{(k)}$ [28]. In summary, under conditions specified in [18, 27, 59, 157, 173], the BCD method may converge to a stationary point, a coordinate-wise minimum point, a local minimum point or a global minimal point.

Although BCD is widely used in problems when $\mathbb{X}_j = \mathbb{R}_j^n$, for each $j = 1, ..., p$, there are several applications of the BCD method in signal processing and machine learning problems when $\mathbb{X}_j = \mathbb{R}^{m_j \times n_j}$, for all $j = 1, \dots, p$ (see, e.g., [22, 68, 73, 79, 130, 134]).

2.5.2 Maximum Block Improvement

A new block optimization method of solving (2.5.1) is developed in [31, 87]. This method is called maximum block improvement (MBI). The MBI method only accepts a block update that achieves the minimum improvement using a greedy approach. Assuming that optimizing over one variable while fixing all others is relatively easy, the MBI method updates the block of variables corresponding to the maximally improving block using the Jacobian rule, which is arguably one of the most natural and simple processes to tackle block-structured problems with great potential for engineering applications [31, 73]. Algorithm 2 provides the framework of the MBI method.

Algorithm 2 MBI method for solving (2.5.1)

Requires: $x^{(0)} = (x_1^{(0)}, \dots, x_p^{(0)}) \in \mathbb{X}_1 \times \dots \times \mathbb{X}_p$ (Initial point)
Returns: $x^{(k)} = (x_1^{(k)}, \dots, x_p^{(k)}) \in \mathbb{X}_1 \times \dots \times \mathbb{X}_p$

for $k = 1, 2, ...$ **do**
  for $j = 1, ..., p$ **do**
    $\widetilde{x}_j^{(k)} =$ Using (2.5.1)
    $e_j^{(k)} = f(x_1^{(k-1)}, \dots, x_{j-1}^{(k-1)}, \widetilde{x}_j^{(k)}, x_{j+1}^{(k-1)}, \dots, x_p^{(k-1)})$
  end
  $i = \arg \min\limits_{j=1,\dots,p} \left\{ e_j^{(k)} \right\}$
  for $j = 1, ..., p$ **do**
    if $j = i$ **then**
      $x_j^{(k)} = \widetilde{x}_j^{(k)}$
    else
      $x_j^{(k)} = x_j^{(k-1)}$
    end
  end
  if stopping criterion is satisfied **then**
    return $x^{(k)} = (x_1^{(k)}, \dots, x_p^{(k)})$
  end
end

The MBI method can be considered as a more expensive version of the BCD method. Improvements of all blocks have to be calculated in each iteration. This problem might be resolved with a straightforward parallel approach. At the same time, the MBI method improves convergence properties of the BCD. If $\mathbb{X}_j$ is a compact set, for all $j = 1, \dots, p$, then the MBI method converges to a coordinate-wise minimum point [31, 172] $x^* = (x_1^*, \dots, x_p^*) \in \mathbb{X}_1 \times \dots \times \mathbb{X}_p$ of problem (2.5.1), i.e.,

$$x_j^* \in \arg \min_{x_j \in \mathbb{X}_j} f\left(x_1^*, \dots, x_{j-1}^*, x_j, x_{j+1}^*, \dots, x_p^*\right).$$