Chapter 2

Background

In this chapter, the mathematical background of the proximal gradient algorithms, interpretation of the proximal gradient methods as the fixed-point iterations and generalization of the proximal gradient methods as forward-backward splitting is discussed. A brief discussion on accelerated gradient algorithms that converges faster than the traditional proximal gradient algorithms is also covered. Along with the general assumptions of our approaches, the two concepts of extragradient-based and viscosity approximation-based fixed point methods, which we utilized in our work, are also presented. We start this chapter with the basic introduction of the operator splitting techniques.

2.1 Operator Splitting Techniques

We consider the following minimization problem,

$$\min_{x \in \mathbb{R}^d} F(x), \tag{2.1}$$

where F is a maximal monotone operator (see definition in appendix). Another way to interpret this problem is to find the zeros of operator F, i.e.,

$$0 \in F(x). \tag{2.2}$$

The main idea is to split F as sum of two maximal monotone operators A and B and find the solution of (2.2), i.e.,

$$0 \in (A + B)(x). \tag{2.3}$$

The core assumption behind this idea is that the computation of zeros of $(A+B)$ is simpler than to find zeros of F. In the literature, there exists a variety of operator splitting methods. However, three techniques are considered to be the standard, namely, Forward-backward Operator Splitting, Peaceman-Rachford Operator Splitting and the Douglas-Rachford Operator Splitting [63].

Consider two proper closed and convex functions f and g. In the field of learning theory, the above framework (2.3) is applicable for solving the regularized convex loss minimization problem, as follows,

$$\min_{x \in \mathbb{R}^d} F(x) = f(x) + g(x). \tag{2.4}$$

With respect to (2.3), the problem we need to solve is the following,

$$0 \in (\partial f + \partial g), \tag{2.5}$$

where ∂f and ∂g are the subgradients of functions f and g, and are equivalent to A and B, respectively.

2.2 Forward-backward Splitting Techniques

In this subsection, a class of operator splitting method, the forward-backward splitting methods, and their properties are discussed. It is known that the Hilbert spaces $\mathscr{H}$, which generalize the notion of Euclidean spaces and have distance function induced by the inner product. Let $T : \mathscr{H} \to \mathscr{H}$ be an operator. T is called an $L-$Lipschtiz operator if there exists $L \in [0, \infty)$ such that

$$\|Tx - Ty\| \leq L\|x - y\|, \quad x, y \in \mathscr{H}.$$

An $L-$Lipschitz operator is called a non-expansive operator if $L = 1$ and contraction if $L < 1$.

The main assumption for the forward-backward splitting methods is the differentiability of the function $f(\cdot)$. Under this assumption, (2.5) can be written as,

$$0 \in \nabla f(x^*) + \partial g(x^*),$$

where ∇f and ∂g refer to the gradient and sub-gradient of functions f and g respectively, and x^* is the solution of (2.4). Now, for any $\lambda > 0$ and the identity matrix Id, optimality condition holds if,

$$\begin{aligned} 0 &\in \lambda \nabla f(x^*) + \lambda \partial g(x^*) \\ 0 &\in \lambda \nabla f(x^*) - x^* + x^* + \lambda \partial g(x^*) \\ (Id + \lambda \partial g)(x^*) &\ni (Id - \lambda \nabla f)(x^*) \\ x^* &= (Id + \lambda \partial g)^{-1} (Id - \lambda \nabla f)(x^*) \qquad (2.6) \\ &= prox_{\lambda g}(x^* - \lambda \nabla f(x^*)), \qquad (2.7) \end{aligned}$$

From last two equations, it is clear that the solution x^* minimizes the sum of function f and g, if it is a fixed-point of the forward-backward operator $(Id + \lambda \partial g)^{-1}(Id - \lambda \nabla f)$. It also demonstrates the relation between the proximity operator and the forward-backward operator. It is well-known that for any $\lambda > 0$, the resolvent of a maximal monotone operator T is a non-expansive operator [167]. This very relation is used in finding the solution x^* iteratively, using the proximal gradient algorithms, which we discuss next.

2.3 Proximal Gradient Methods

Let $f(\cdot)$ be a convex smooth loss function with $L-$ Lipschitz gradient, $g(\cdot)$ be a non-smooth convex function and $\rho > 0$ be a regularization parameter, we solve the following regularized convex loss minimization problem,

$$\min_{x \in \mathbb{R}^d} F(x) = f(x) + \rho \cdot g(x). \qquad (2.8)$$

The traditional approach to solve this problem is to apply the basic subgradient descent method as proposed in [30], which uses a black-box type technique to solve (2.8) with the

subgradients of non-smooth functions. The subdifferential of the non-smooth function g at x (denoted as $\partial g(x)$) is defined by,

$$\partial g(x) = \{y \mid g(z) \geq g(x) + y^T(z-x) \quad \forall z \in dom \ \ g\}. \tag{2.9}$$

Any point $y \in \partial g(x)$ is called a subgradient of g at x. The main drawback of this algorithm is that it is not able to learn the sparsity structure and is slow to converge [13, 12]. Proximal gradient methods are used as a popular alternative to the subgradient descent methods.

In proximal algorithms, in place of computing ∂g, we compute the resolvent of ∂g, which is called the proximal operator (denoted as $\text{prox}_{\lambda g}$) with respect to a $\lambda > 0$, defined as $(Id + \lambda \partial g)^{-1}$. For any non-differentiable function $g(\cdot)$, we consider computing the proximity operator $\text{prox}_{\gamma g}$, defined as follows,

$$\text{prox}_{\gamma g}(z) := \underset{x}{\text{argmin}} \ \{\gamma g + \frac{1}{2}\|x-z\|^2\}. \tag{2.10}$$

Most of the problems we consider in this work come under the nonsmooth lasso framework, which assumes $g(\cdot) = \|\cdot\|_1$, thus providing a sparse solution. To compute the exact value of the proximity operator corresponding to different definitions of function $g(\cdot)$, various formulations are available in the literature. For example, in the case of the l_1 norm, the proximity operator is defined as a soft-thresholding operator. The exact value of proximity operator is defined as a soft thresholding operator as follows,

$$prox_{\lambda\|\cdot\|_1}(v) = (v-\lambda)_+ - (-v-\lambda)_+ = \begin{cases} v_i - \lambda, & v_i \geq \lambda \\ 0, & |v_i| \leq \lambda \\ v_i + \lambda, & v_i \leq -\lambda \end{cases} \tag{2.11}$$

For more complex structured penalty functions, the corresponding proximity operators are not directly solvable, and hence the inexact computation of such proximity operators are considered. Objective functions with such complexed structured penalties are optimized in a nested fashion, the outer optimization problem for the loss minimization and the inner problem being the computation of proximity operator. In this work, we consider two complex structures, namely overlapping group penalty and fused penalty in Chapter

3. Evaluating the proximity operator of a function is considered as an essential step of the Proximal Gradient Algorithm.

The Proximal Gradient Algorithm (PGA) solves problem (2.4) with the following iterative scheme, with $\lambda > 0$ and $x_1 \in \mathbb{R}^d$,

$$x_{n+1} \leftarrow prox_{\lambda g}(Id - \lambda \nabla f)(x_n), \tag{2.12}$$

where x_n denotes the n^{th} iteration of the algorithm. We assumed that the parameter λ is updated at each iteration, i.e. we use parameter λ_n.

For simplicity, consider the forward-backward operator with respect to λ, i.e., $prox_{\lambda g}(Id - \lambda \nabla f)$ as T and with respect to λ_n, i.e., $prox_{\lambda_n g}(Id - \lambda_n \nabla f)$ as T_n. Thus, the forward-backward algorithm corresponding to (2.12) becomes the iterative procedure,

$$x_{n+1} \leftarrow T_n(x_n). \tag{2.13}$$

The above scheme follows the Picard fixed-point iterative scheme. To guarantee the convergence of this algorithm, the value of λ should belong to $(0, 1/L_f]$, where L_f is the Lipschitz constant of ∇f. This range assures that the forward-backward operator is averaged, and thus the iterations will converge to a fixed point. We have considered that the value of the parameter λ is not known in advance. The Lipschitz constant L_f plays an important role as a step size. So, fixing $\lambda = L_f$ is not a good idea for the practical implementations [102]. Thus, following [23], we adopt the backtracking line search at each iteration in our algorithms.

2.3.1 Backtracking Line Search

Let us consider the quadratic approximation of our objective function $F(x) = f(x) + g(x)$ at any point p is given by:

$$Q_\lambda(x, p) = f(p) + < x - p, \nabla f(p) > + \frac{\lambda}{2}\|x - p\|^2 + g(x). \tag{2.14}$$

Algorithm 1: Backtracking Line Search

Result: $\lambda_n := \lambda$; $x_{n+1} := z$.
begin
  $\lambda = \lambda_{n-1}$;
  repeat
    $z \leftarrow T_n(x_n)$;
    break if $f(z) \leq Q_{\lambda_n}(z, x_n)$;
    Update $\lambda \leftarrow \beta \cdot \lambda$;
  until;

With the backtracking line search, the iterative scheme (2.13) will be as follows. At the n^{th} step, given x_n, λ_{n-1} and a parameter $\beta \in (0,1)$. Then the backtracking line search is defined as shown in algorithm 1.

2.4 Accelerated Gradient Methods

The main drawback of proximal gradient methods is their slow convergence, which can be removed using an inertial extrapolate. The main concept was given by Nesterov in [135], which solves the following problem of minimizing a smooth convex function f as follows,

$$\min_{x \in \mathbb{R}^d} f(x) \tag{2.15}$$

In the heavy ball method by Polyak [158], at each iteration gradient is calculated at point x_n, whereas in Nesterov's method gradient, which is a modification of the heavy-ball method, is to be calculated at an extrapolated point y_n, as follows,

$$\begin{cases} y_n & \leftarrow x_n + \alpha_n(x_n - x_{n-1}) \\ x_{n+1} & \leftarrow y_n - \lambda_n \nabla f(y_n). \end{cases} \tag{2.16}$$

The sequence $\{\alpha_n\}$ has to be computed in a specific manner such that it allows the optimal convergence rates. In [23], this idea is extended for solving the non-smooth convex minimization problem, with the same convergence rates as provided by the Nesterov's

method in [136], as follows,

$$\begin{cases} y_n & \leftarrow x_n + \alpha_n(x_n - x_{n-1}) \\ x_{n+1} & \leftarrow prox_{\lambda_n g}(y_n - \lambda_n \nabla f(y_n)). \end{cases} \tag{2.17}$$

Here $\alpha_n = \left(\frac{t_{n-1}-1}{t_n}\right)$ with $t_{n+1} \leftarrow \frac{1+\sqrt{1+4t_n^2}}{2}$ and $g(\cdot)$ is a $\|\cdot\|_1$ function. Other definitions for α_n are also available, for example [47, 138]. The basic behaviour of the term α_n is shown in figure 2.1. The above algorithm is known as the Fast Iterative Shrinkage-Thresholding Algorithm (FISTA). With all other definitions of the sequence $\{\alpha_n\}$, the general class of such algorithms is call the Accelerated Gradient Descent (AGD) algorithm. In the general settings under the Hilbert space $\mathscr{H}$, AGD is called an inertial forward-backward splitting algorithm, given as follows,

$$\begin{cases} y_n & \leftarrow x_n + \alpha_n(x_n - x_{n-1}) \\ x_{n+1} & \leftarrow T_n(y_n). \end{cases} \tag{2.18}$$

With an extrapolated point y_n, the iterative scheme (2.18) is again a Picard's fixed point iterative scheme. In this thesis, we aim to propose new iterative schemes of accelerated gradient algorithms. The general assumptions for our schemes are given in the next section.

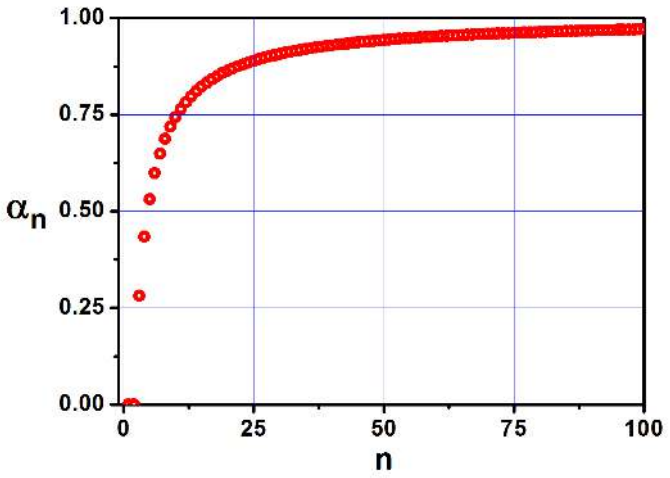

FIGURE 2.1: Illustration of the behavior of sequence $\{\alpha_n\}$ vs. iterations.

2.5 General Assumptions

Our proposed algorithms are based on the following assumptions:

1. In the first two contributions (Chapter 3 and Chapter 4), the loss function $f(\cdot)$ is convex and continuously differentiable with Lipschitz continuous gradient L_f, and there exists a constant $L > 0$, with $L \geq L_f$ such that:

$$\|\nabla f(x) - \nabla f(y)\| \leq L\|x - y\| \quad \forall\, x, y \in \mathbb{R}^d,$$

2. The regularization function $g(\cdot)$ is a continuous convex function which may be non-smooth.

3. The problem (2.1) is solvable, i.e., $x^* = argmin\, F(x) \neq \phi$.

4. The sequence $\{\alpha_n\} \in (0,1)$, for $n = 1, 2, \cdots \mathbb{N}$ is non-decreasing.

The assumption of differentiability is relaxed in the third contribution (Chapter 5). We have utilized two new concepts of fixed-point theory in this thesis which are, the concept of extragradient of fixed-point theory and the viscosity-approximation based fixed-point iterations. A brief discussion on each one of these concepts is presented next.

2.6 Viscosity-Approximation Fixed-point Scheme

In the field of mathematical analysis, viscosity methods are very popular for providing efficient solution to various problems of different branches such as mathematical programming (for example, Tikhonov regularization and exponential penalty methods for linear programming), variational problems (for example, minimal hyper-surfaces, plasticity theory, and phase transition), partial differential equations (for example, entropy solutions of first-order non-linear hyperbolic systems, viscosity solutions of Hamilton-Jacobi equations), and ill-posed problems. A major feature of these methods is to provide, as a limit of the solutions of the approximate problems, a particular (possibly relaxed or generalized) solution of the original problem, called a viscosity solution, which has remarkable properties. Let C be a close convex subset of a real Hilbert space $\mathscr{H}$. As we know

that a strongly nonexpansive mapping (or contraction mapping) $\mathscr{T}$ satisfies the following property for $x, y \in C$ (see appendix):

$$\|\mathscr{T}x - \mathscr{T}y\| \leq \theta\|x-y\|, \qquad \text{with } 0 \leq \theta < 1. \tag{2.19}$$

In the field of fixed point theory, it is well known that a fixed point x^* exists for $\mathscr{T}$, and it is unique. Moreover, the convergence is stable with respect to perturbation in $\mathscr{T}$. As θ reaches to 1, the problem becomes unstable, and the convergence can be only weak. Thus, it is necessary to apply some regularizing procedures.

In [114], authors combined the proximal point method to the Tikhonov regularization to introduce the prox-Tikhonov method, which generates the sequence $\{x_n\}$ as follows,

$$x_{n+1} \leftarrow J^{T_n}_{\lambda_n} x_n, \quad n \geq 0,$$

where $T_n = \mu_n I + T,\ \mu_n \geq 0$ is viewed as a Tikhonov regularization of T and is a strongly monotone operator. In [208], author proposed another prox-Tikhonov proximal point method as follows,

$$x_{n+1} \leftarrow J^{A}_{c_n}(\alpha_n u + (1-\alpha_n)x_n + e_n),$$

where $\alpha_n \in [0,1]$, A is a monotone operator, $J^A_{\lambda_n}$ is the resolvent of A and e_n is computational error. These two methods are very basic viscosity-approximation fixed-point schemes. We have applied a recent viscosity-approximation-based forward-backward scheme to solve the machine learning problems, and proposed an inertial viscosity-approximation-based forward-backward scheme in this thesis.

2.7 Extra-gradient Methods

Initially proposed by Korpelevich [110], extragradient method is a classical method for solving variational inequality problems, which is defined as the problem of finding $x^* \in \mathscr{S}$ such that,

$$\langle H(x), x - x^* \rangle \geq 0, \quad \forall x \in S,$$

where S is a nonempty, closed and convex subset of $\mathbb{R}^d$, $H : \mathbb{R}^d \to \mathbb{R}^d$ is a monotone mapping, and $\langle \cdot, \cdot \rangle$ denotes the Euclidean inner product in $\mathbb{R}^d$. The extragradient method

in [110] is defined as the following iterative scheme,

$$y_n \leftarrow P_S(x_n - \alpha H(x_n)),$$
$$x_{n+1} \leftarrow P_S(x_n - \alpha H(y_n)).$$

Many modified versions of the extragradient method are proposed thereafter [48, 161, 49]. For instance, in [190], authors introduced the approach for solving the problem of structural prediction, which is formulated as a convex-concave saddle point problem. In this work, we will exploit the practical performance of this method for solving machine learning problems.

All the traditional fixed-point schemes that are utilized in the field of machine learning are very old. Various new fixed-point schemes and concepts are proposed thereafter. In this thesis, we applied few of these new concepts for solving the problem of machine learning as well as analyzed the practical performance of these concepts on real-world problems. We first applied the concept of extragradient fixed-point iterative schemes to the machine learning in next chapter (Chapter 3) and shown its performances with some real-world applications.