1 28th of November 2018 — A. Frangioni

1.1 Subgradient methods

We are still in the hypothesis of convex objective functions that are not differentiable in the whole domain.

Do you recall?

?? of last lecture: for f convex function, $\forall g \in \partial f(x), \forall\ y \in \mathbb{R}^n g(y) \equiv f(y) \geq f(x) + g(y - x)$. This is a characterization of the function with respect to the model.

Let us assume we know the minimum point x_*. We observe that the scalar product between the subgradient and the direction we should choose is negative. Formally, $f(x_*) \geq f(x) + < g, x_* - x >$, hence $< g, x_* - x > \leq f(x_*) - f(x) \leq 0$.

We want to bound the distance between the point at the "next step" and the optimum:

$$\begin{aligned} \left\|x^{i+1} - x_*\right\|^2 &\overset{(1)}{=} \left\|x^i - \alpha^i d^i - x_*\right\|^2 \\ &= \left\|x^i - x_*\right\|^2 + 2\alpha^i g^i \frac{(x_* - x^i)}{\|g^i\|} + (\alpha^i)^2 \\ &\overset{(2)}{\leq} \left\|x^i - x_*\right\|^2 - 2\alpha^i \frac{(f(x^i) - f(x_*))}{\|g^i\|} + (\alpha^i)^2 \end{aligned} \tag{1.1}$$

Where $\overset{(1)}{=}$ follows from the definition of a normalized step: $x^{i+1} = \frac{x^i - \alpha^i g^i}{\|g^i\|}$ and $\overset{(2)}{\leq}$ follows from the inequality we stated above.

Observation 1.1. *The distance from the optimum is bounded by a square function in α_i (the step size), where the linear part is negative and the quadratic part is positive.*

Hence, if the steps are short enough we get close to the optimum fast enough, because the linear part dominates the quadratic one.

Formally, $-2\frac{(f(x^i) - f(x_*))}{\|g^i\|} < 0 + \alpha^i \searrow$, hence $(\alpha^i)^2 \searrow 0_+$.

An attentive reader may notice that from Equation (1.1) follows $\|x^{i+1} - x_*\|^2 < \|x^i - x_*\|^2 + (\alpha_i)^2$ and $\|x^{i+1} - x_*\|^2 \leq \|x^1 - x_*\|^2 + \sum_{k=1}^{i} (\alpha^k)^2$.

The point here is that we cannot choose a too small α, recall Armijo conditions.

If the series of the squares of step sizes does not diverge, then the sequence does not diverge as well, hence it converges somewhere, say $\bar{x}$.

The convergence of the series of the squares may be obtained using the following

Definition 1.1 (Diminishing-Square Summable)**.** *We term a series that diverges, while the series of the squares of the terms diverges, as* ***diminisching-square summable****.*

Formally,

(DSS) $\sum_{i=1}^{\infty} \alpha^i = \infty \ \wedge \ \sum_{i=1}^{\infty} (\alpha^i)^2 < \infty$.

Assumptions: function convex, definite in all its domain, hence the norm of the gradient will never become very large.

Fact 1.1. *We claim that the sequence of the stepsizes is a diminishing-square sequence.*

Proof by contraddiction. Let us assume $f(x^i) - f_* \geq \varepsilon > 0,\ \forall i$. Then,

$$\left\|x^{i+1} - x_*\right\|^2 \leq \left\|x^1 - x_*\right\|^2 - \delta \cdot \sum_{k=1}^{i} \alpha^k + \sum_{k=1}^{i} \left(\alpha^k\right)^2$$

The contraddiction is due to the fact that as $i \to \infty$ the right-hand side goes to $-\infty$. □

This family of algorithms is clearly incredibly robust in theory, but in practice it does not work very well.

Let us make an experiment: let us suppose we know the minimum of the function f.

Definition 1.2 (Plyak stepsize)**.** *Let f be our convex objective function and let x_* be the optimum for f. We term* ***Polyak stepsize****:*

$$\textit{(PSS)}\ \alpha^i = \beta^i \frac{(f(x^i) - f(x_*))}{\|g^i\|}$$

where $\beta \in (0, 2)$.

If we pick a Polyak stepsize, then $\|x^{i+1} - x_*\|^2 < \|x^i - x_*\|^2$, so $\{x^i\} \to x_*$. The best value for β_i is 1. In this case, we obtain what follows by sybstituting $\beta_i = 1$ in Equation (1.1)

$$\frac{\left(f(x^i) - f(x_*)\right)^2}{\|g^i\|^2} \leq \left\|x^i - x_*\right\|^2 - \left\|x^{i+1} - x_*\right\|^2$$

The problem is that we don't know the optimum.

Let us assume that we know it and compute the efficiency:

Since we know that the sequence is bounded, we know that the objective function is globally Lipshitz (the norm of the gradient is bounded above).

The point is that the sequence $\{x_1\}$ is not necessarily monotone, so we pick the so-called record value of best vaue ($\underline{f}^i = \min\{f(x^h)\ :\ h = 1, \ldots, i\}$)

$$\frac{\left(\underline{f}^i - f(x_*)\right)^2}{L^2} \leq \frac{\left(f^i - f(x_*)\right)^2}{\|g_i\|^2} \leq \left\|x^i - x_*\right\|^2 - \left\|x^{i+1} - x_*\right\|^2$$

Summing for $i = 1, \ldots, k$ we obtain a telescopic series:

$$\left\|x^1 - x_*\right\| - \cancel{\left\|x^2 - x_*\right\|} + \cancel{\left\|x^2 - x_*\right\|} - \cancel{\left\|x^3 - x_*\right\|} + \cdots + \cancel{\left\|x^k - x_*\right\|} - \left\|x^{k+1} - x_*\right\|$$

Hence resulting in

$$k \cdot \frac{\left(\underline{f}^k - f(x_*)\right)^2}{L^2} \leq \left\|x^1 - x_*\right\|^2 - \left\|x^{k+1} - x_*\right\|^2 \leq \left\|x^1 - x_*\right\|^2 = R$$

which is equivalent to $\left(\underline{f}^k - f(x^*)\right)^2 \leq \frac{R^2L^2}{k}$, which is again equivalent to $\underline{f}^k - f(x^*) \leq \sqrt{\frac{RL}{k}}$, where L is the Lipshitz constant.

The issue here is that the convergence is sublinear: $k \geq \frac{1}{\varepsilon^2}$.

Theorem 1.2. *Take an algorithm that uses only the subgradient. It's possible to construct a function that makes the algorithm converge with sublinear speed. Hence, we cannot do better.*

It comes without saying that although this algorithm is not very good it is the "less bad" it can be.

There are some lucky cases in which we do know the optimal value, but this is not the case usually.

1.1.1 Target level stepsize

Let us assume $f(x_*)$ is unknown. The only information available is that this value is below any value in any iteration.

The rationale behind this algorithm is to assume to know the optimal value and as soon as we realize it is not correct we change it.

Let us first give an informal description of the algorithm:

- δ is the displacement: how much below the function is with respect to the best value obtained so far;
- reference value $f_{rec} = \underline{f}$. At the beginning is the value at the first iterate and then we define the target value as the difference between the reference value and some δ (at the beginning δ_0).

ALGORITHM 1.1 Pseudocode for target level stepsize.

```
1: procedure SGPTL(f, g, x, i_max, β, δ_0, R, ρ)
2:     r ← 0;
3:     δ ← δ_0;
4:     f_ref ← f_rec ← f(x);
5:     i ← 1;
6:     while (i < i_max) do
7:         g = g(x);
8:         α = β(f(x) − (f_ref − δ))/‖g‖^2;
9:         x ← x − αg;
10:        if (f(x) ≤ f_ref − δ/2) then
11:            f_ref ← f_rec;
12:            r ← 0;
13:        else
14:            if ( r > R ) then
15:                δ ← δρ;
16:                r ← 0;
17:            else
18:                r ← r + α‖ g ‖;
19:            end if
20:        end if
21:    end while
22:    f_rec ← min{ f_rec , f(x) };
23:    i ← i + 1;
24: end procedure
```

At this point we defined the algorithm and we are ready to implement it, except for the fact that we need to choose of a lot of parameters. A way to choose them is to use the ML approach: try many possible values.

Two big issues of this algorithm are that it does not provide a good stopping criterion and it is very sensitive to many parameters.

1.2 Deflected subgradient

The idea is to use the same trick of ball-step (also called primal-dual).

Let us assume that our function was differentiable. The subgradient method collapses to the gradient method and we know that the gradient method does not provide a good convergence. Yet, deflection is possible: $d^i = \gamma^i g^i + (1 - \gamma^i) d^{i-1}$, $x^{i+1} = x^i - \alpha^i d^i$. We can prove that d^i approximates the subgradient. We can also prove that the algorithm converges in the end. The prameters of this algorithm are two: β (stepsize) and γ (deflection). In order to choose them we have two different approaches:

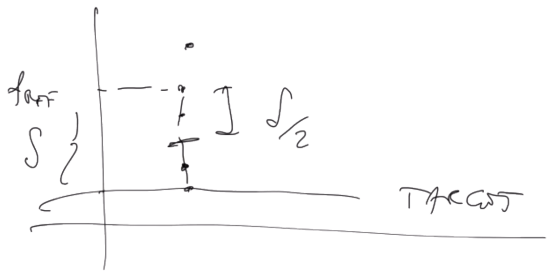

FIGURE 1.1: There are two cases: either the function value is significanlty below the reference (for example $\frac{\delta}{2}$ below the reference) or it's not . If we are in the "happy case" (we just move the reference to the best value). For what conserns the "unhappy case", if we are unhapy for 1 iteration, no problem. Two iterations? no prolem. Many iterations? Problem: after some iterations in which we are not improving it means that we have to decreas the reference values. How? r is updated at each bad step and reset when a good step occurs. When r gets too large we decrease δ and reset everything.

STEPSIZE-RESCRICTED $\equiv$ deflection-first. We first choose β and when choosing α we need to take into account β: $\alpha^i = \frac{\beta^i(f(x^i)-f_*)}{\|d^i\|} \ \wedge \ \beta^i \leq \gamma^i$ "as deflection $\nearrow$, stepsize has to $\searrow$";

DEFLECTION-RESTRICTED $\equiv$ stepsize-first. We first choose γ, then we pick a step size that depends on γ:

$$\text{(DSS)} \ \wedge \ \frac{\alpha^{i-1}\|d^{i-1}\|}{(f(x^i)-f_*)+\alpha^{i-1}}\left\|d^{i-1}\right\| \leq \gamma^i$$

"as $f(x^i) \to f_*$, deflection $\searrow$"

This algorithm gets the optimal $O(1/\varepsilon^2)$ on average, sadly not worst case.

1.3 Smoothed gradient methods

Let us assume that the target function is a **Lagrangian function**.

Definition 1.3 (Lagrangian function)**.** *Let $f : \mathbb{R}^n \to \mathbb{R}$ be a function of the following shape:*

$$f(x) = \max\{x^T Az \ : \ z \in Z\}$$

where Z is convex and bounded.

Let us assume that Z is also compact.

A graphical example in the case of $f(x) = |x| = \max\{x, -x\} = \max\{zx : z \in [-1, 1]\}$ is shown in Figure 1.2.

In the case of the absolute value, the nasty trick is "to make it have only one optimal solution" in the point 0.

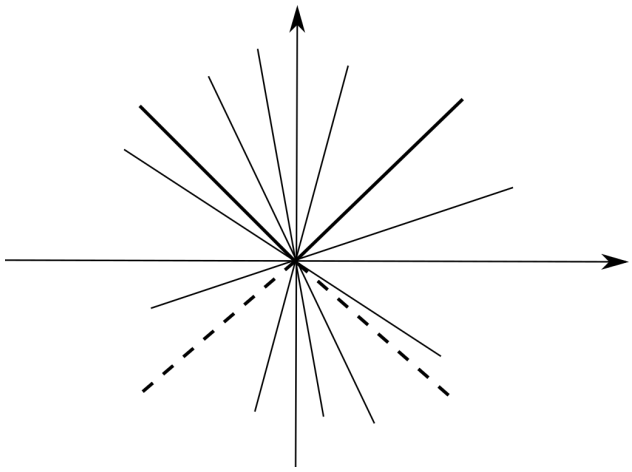

FIGURE 1.2: Let us take the absolute value function $f(x) = |x|$. This function would be differentiable, if it wasn't for some nasty points (in this case 0, were the optimization problem has many optimal solutions). In 0 there are many subgradients, so it has many optima.

This result is obtained adding a small quadratic term: $f(x) = max\{x^T Az - \mu\|z\|^2 \ : \ z \ inG\}$ that is shown in Figure 1.3. At this point the new function f_μ is not the original function anymore, but it is very close to it whenever the μ is small.

Notice that this new function is smooth (differentiable).

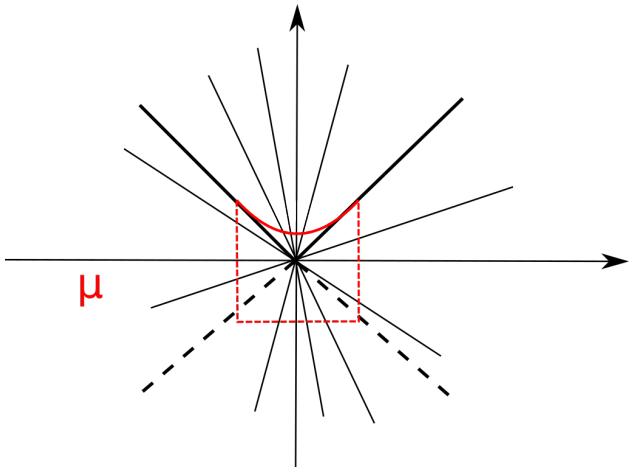

FIGURE 1.3: Geometric intuition of the usage of variable μ.

It might be not ery easy to compute once chosen the value for μ.

We are solving a problem which is different from the original one and would be exactly the same if μ where 0.

On the other hand for $\mu = 0$ we have the problem which is not differentiable once again, so we need to keep close to 0 but not too close.

At this point we are in the situation of $f_\mu(x) \le f(x) \le f_\mu(x) + \mu R$, such that as $\mu \searrow 0$, "argmin $\{ f_\mu(x) \} \to x_*$".

The new function is not only convex, but it is also Lipshitz continuous: ∇f_μ Lipschitz with $L = \|A\|^2/\mu$, but it is "less and less Lipschitz" as $\mu \searrow 0$.

Fact 1.3. *If $f_* > -\infty$ and picking a very special value of μ ($\mu = \varepsilon/(2R)$), then an appropriate ACCG obtains $f(x^i) - f_* \le \varepsilon$ for $i \ge 4 \cdot \|A\| \cdot \|x_*\| \cdot \sqrt{R}/\varepsilon$.*

We observe that the convergence is much better, because it depends on $O(1/\varepsilon)$ instead of $O(1/\varepsilon^2)$.

FIGURE 1.4: At the beginning we eill make a lot of bad steps (the upper gray line). We can improve (pick the black line) changing the ε value and we obtain something that looks more stable. The more precision we want, the smaller the step we make. At the ending it pays, but at the beginning it is not so.

1.4 Cutting-plane algorithm

We cheated to get first order information, we want to do more. We want to cheat and have also the second order information.

We want to use the same idea of limited memory quasi Newton methods. Using some limited memory of the Hessian, in order to understand th curvature.

The point is that the directional derivatives are defined and (if computed massively) give me a hint of the curvature of the function (cfr. Figure 1.5). Notice that it is not possible to build a matrix, because it is not defined.

Let us say we have performed i iterates, then we have collected i subgradients ($\mathscr{B} = \{(x^i,\ f^i = f(x^i),\ g^i \in \partial f(x^i))\} \equiv$ bundle of first-order information) and function values.

We can now define a piece wise linear function defined as the maximum of the first order model:

$$f_{\mathscr{B}}(x) = \max\{f^i + g^i(x - x^i)\ :\ (x^i, f^i, g^i) \in \mathscr{B}\}$$

At this point we can apply Newton method: first we minimize the model and the use the minimum as next point. We collect information in that specific point and then we repeat.

Notice that the model is always below the objective function ($f_{\mathscr{B}}(x) \le f(x)\ \forall x$), hence $\min\{f_{\mathscr{B}}(x)\} \le f_*$, so $x^*_{\mathscr{B}} \in \operatorname{argmin}\{f_{\mathscr{B}}(x)\} \approx\ x_*$.

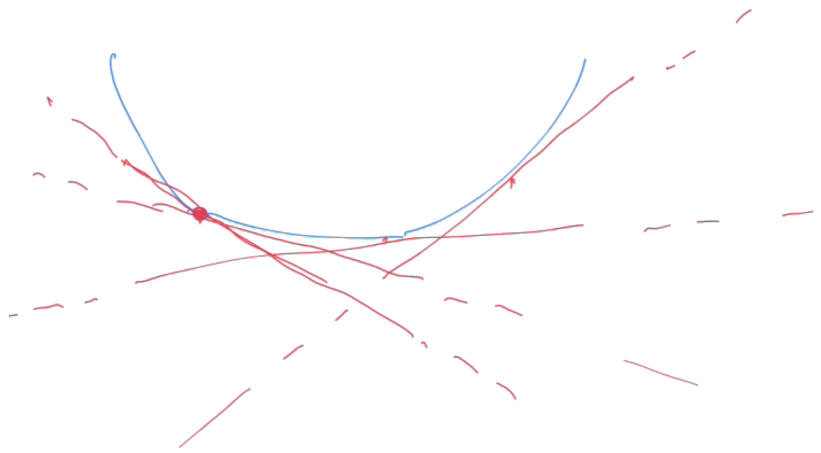

FIGURE 1.5: The idea is that we do not compute a subgradient and discard it. We store it, thus resulting in a model

This function has many kinky points, so how to minimize it? Dirty trick from Ricerca Operativa:

$$\min\{\, f_{\mathscr{B}}(x)\,\} = \min\{\, v \,:\, v \geq f^i + g^i(x - x^i),\ \text{where } (x^i, f^i, g^i) \in \mathscr{B}\}$$

And on this problem, we can use the simplex method.

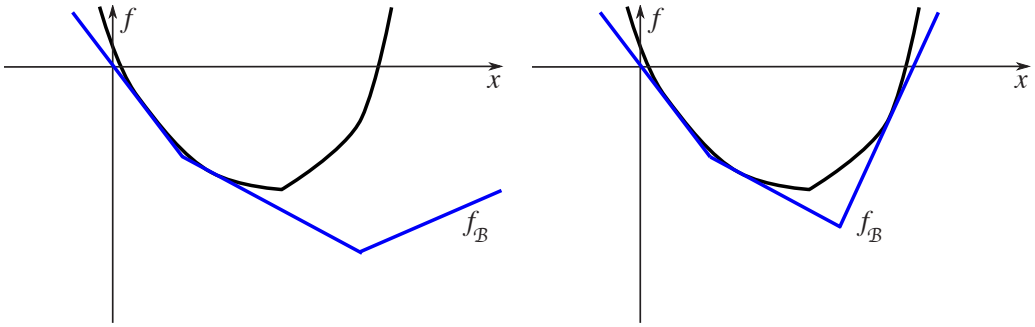

FIGURE 1.6: A geometric representation of how the model (blue line) changes after one iteration.

At this point we have obtained both an upperbound and a lower bound, which go one towards the other.

We may decide to stop iterating when they are "close enough".

The problem is that at each step we need to solve a minimization problem and the convergence is very slow.

It's also possibile that the blue function (the model) does not have a minimum (unbounded below).

How to overcome this problem? Use so-called bundle methods.

1.5 Bundle methods

The intuition behind this is that we want to keep quite close to the point where the model corresponds to the function, because the further we get the more the difference from the function.

A way to express this is to add a quadratic quantity to the function that grows when we move outside the current point.

Definition 1.4 (Stabilized master problem)**.** *We term* ***stabilized master problem*** *the following*

$$\min\{f_{\mathscr{B}}(x) + \mu\|x - \bar{x}\|^2/2\}$$

This improved model cannot be undounded below (quadratic function), but we need to choose μ and $\bar{x}$ wisely.

A possible way is to move the **stability center** whenever the current value is better the the best encountered so far.

ALGORITHM 1.2 Pseudocode for boundle method.

```
1: procedure PBM(f, g, x̄, m_1, ε)
2:     choose μ;
3:     B ← {(x̄, f(x̄), g(x̄))};
4:     while ( true ) do
5:         x* ← argmin {f_B(x) + μ‖x − x̄‖²/2};
6:         if (μ‖ x* − x̄ ‖_2 ≤ ε) then
7:             break;
8:         end if
9:         if (f(x*) ≤ f(x̄) + m_1(f_B(x*) − f(x̄))) then
10:            x̄ ← x*;
11:            possibly decrease μ;
12:        else
13:            possibly increase μ;
14:        end if
15:        B ← B ∪ ( x* , f(x*) , g(x*) );
16:    end while
17: end procedure
```

This algorithm may never move (without cycling, luckily), but at least we gained some information.

The bundle method converges in few steps, although each step is quite costly.

We reached a point where to solve an unconstrained problem we need to solve a contrained one, so from next lecture we will start dealing with constrained optimization problems.