



SIXTH FRAMEWORK PROGRAMME

FP-6 STREP 30717 PLATO-N (Aeronautics and Space)

PLATO-N

**A PLAtform for Topology Optimisation incorporating Novel, Large-Scale,
Free-Material Optimisation and Mixed Integer Programming Methods**

Large Scale Methods for Convex FMO-Type Problems

PLATO-N Public Report PU-R-3-2007

September 15, 2007

Authors:

Amir Beck

Aharon Ben-Tal

Luba Tetrushvili

Large Scale Methods for Convex FMO-Type Problems

Amir Beck , Aharon Ben-Tal and Luba Tretushvili *

September 15, 2007

Abstract

In this report we review several first order methods suited for problems arising from large-scale free material optimization (FMO) problems. Our focus is on the Non-Euclidean Restricted Level (NERML) and on the reduced gradient methods. Both algorithms have demonstrated encouraging empirical results in preliminary numerical testing. Finally, we describe the CONERML method, which is a variation of the NERML method capable of dealing with constraints.

Keywords: first-order methods, large-scale convex optimization, bundle methods, free-material problems

*MINERVA Optimization Center, Faculty of Industrial Engineering, Technion - Israel Institute of Technology, Haifa 32000, Israel.
e-mails: becka@ie.technion.ac.il, abental@ie.technion.ac.il, lubate@tx.technion.ac.il

1 Methods for large-scale convex problems

We are interested in solving the following optimization problem:

$$(P) \text{ minimize } f(x) \text{ s.t. } x \in X \subset \mathbb{R}^n.$$

We assume that X is a closed and bounded convex subset of \mathbb{R}^n and that the objective function is convex Lipschitz continuous on X :

$$\|f(x) - f(y)\| \leq L_f \|x - y\|, x, y \in X,$$

where $L_f < \infty$ and $\|\cdot\|$ is the usual Euclidean norm in \mathbb{R}^n . We also assume that a subgradient of f is computable at any point $x \in X$ and will be denoted by $f'(x)$.

We would like to solve the problem (P) within absolute inaccuracy ε , i.e., to find $x \in X$ such that

$$f(x) - \min_{x \in X} f(x) \leq \varepsilon.$$

2 Subgradient Descent Method

One way to solve the problem is to apply the subgradient descent method. The *Subgradient Descent* method generates the sequence of search points $\{x_i\}_{i=1}^{\infty}$ according to the rule

$$x_{i+1} = \Pi_X(x_i - \gamma_i g(x_i)), \quad g(x) = f'(x) / \|f'(x)\|, \quad (2.1)$$

where $x_1 \in X$ is a starting point, $\gamma_i > 0$ are positive stepsizes and

$$\Pi_X(x) = \operatorname{argmin}\{\|x - y\| \mid y \in X\}$$

is the *projector* onto X .

The efficiency estimate of the subgradient algorithm is given by the following proposition.

Proposition 2.1. *Consider the convex problem (P) with X convex compact set and denote by $\operatorname{Diam}(X)$ the diameter of X , i.e., $\operatorname{Diam}(X) := \max_{x,y \in X} \|x - y\|$. Then the efficiency estimate for the subgradient method with stepsizes $\gamma_i = \operatorname{Diam}(X)i^{-1/2}$ is*

$$\min_{1 \leq i \leq N} f(x_i) - \min_{x \in X} f(x) \leq \frac{O(1)L_f \operatorname{Diam}(X)}{\sqrt{N}},$$

where $O(1)$ is a moderate positive constant and L_f is the Lipschitz constant for f .

When X is a "simple" set, e.g., Euclidean ball, box, hyperplane, or the simplex, the method is computationally very cheap - a step costs only $O(n)$ operations (computation of the projection) in addition to those spent by the oracle for the subgradient calculation. But for large problems starting from $n = 1000$ even if the low accuracy solution is sufficient (one or two correct digits after the point) such rate of convergence would result in tens of thousands of steps. And this rate of convergence cannot be improved by more than the absolute constant factor [3], i.e., the worst case complexity is almost "typical" in practice. Therefore, for very large problems the subgradient descent method is far from being the best choice. There are other methods with the same worst case theoretical complexity bound, but with significantly better "typical" performance. From this practical viewpoint bundle methods are preferable.

3 Bundle Methods

The subgradient method does not use previous information on the objective function obtained so far. Bundle methods utilize the knowledge of the entire "prehistory" in the following way. Consider the sequence of the search points $\{x_j\}_{j=1}^i$ computed so far and the *bundle*, i.e., the sequence of the following affine forms

$$f(x_j) + (x - x_j)^T f'(x_j).$$

It follows from the convexity of the objective function $f(x)$ that every form from the bundle underestimates $f(x)$ and coincides with it at the corresponding search point. All these affine forms from the bundle can be assembled into a piecewise linear convex function which is called the *i-th model of the objective function*

$$f_i(x) = \max_{1 \leq j \leq i} \{f(x_j) + (x - x_j)^T f'(x_j)\}.$$

This model underestimates the objective function

$$f_i(x) \leq f(x), x \in X$$

and is exact at the search points $\{x_j\}_{j=1}^i$. The model provides a good approximation of the objective function at least in a neighborhood of the search

points. Therefore, a naive idea is to define the next point as the solution of the piecewise linear approximation f_i :

$$x_{i+1} = \arg \min_{x \in X} f_i(x). \quad (3.1)$$

But f_i might prove to be a poor approximation of f far away from the search points, especially at the beginning of the process. Thus it is reasonable to forbid the next search point to move far away from the best point obtained so far. Let us choose x_{i+1} as the minimizer of the *penalized model*:

$$x_{i+1} = \arg \min_{x \in X} \left\{ f_i(x) + \frac{d_i}{2} \|x - x_i^+\|^2 \right\}, \quad (3.2)$$

where x_i^+ is what is called the *current prox center*, and the positive parameter d_i is the *prox coefficient*. The scheme (3.2) is the generic form of *bundle methods*.

Bundle algorithm

Step 1. Let $\delta > 0$, $m \in (0, 1)$, x_0^+ - the first prox center, $y_0 = x_0$ and $i = 0$.
Compute $f(x_0^+)$ and subgradient $f'(x_0^+)$. Define $f_0(x) = f(x_0^+) + (x - x_0^+)^T f'(x_0^+)$

Step 2. Compute the next iterate

$$y_{i+1} = \arg \min_{x \in X} \left\{ f_i(x) + \frac{d_i}{2} \|x - x_i^+\|^2 \right\}.$$

Step 3. Define $\delta_i := f(x_i^+) - [f_i(y_{i+1}) + \frac{d_i}{2} \|y_{i+1} - x_i^+\|^2] \geq 0$.

Step 4. If $\delta_i < \delta$ Stop.

Step 5. Compute $f(y_{i+1})$ and subgradient $f'(y_{i+1})$.

Step 6. If $f(x_i^+) - f(y_{i+1}) \geq m\delta_i$, **Serious Step (SS)** $x_{i+1}^+ = y_{i+1}$.

Else, **Null Step (NS)** $x_{i+1}^+ = x_i^+$.

Step 7. Update the model

$$f_{i+1}(y) := \max\{f_i(y), f(y_{i+1}) + (y - y_{i+1})^T f'(y_{i+1})\}$$

Step 8. Set $i = i + 1$, and goto **Step 2**.

Note that the sequence $\{f(x_i^+)\}$ is strictly decreasing due to the Serious Step test in Step 6. The SS test requires improvement of the new prox center by at least a fixed fraction m from the improvement δ_k predicted by the model. The role of the prox-coefficient d_i is to control the trade off between minimizing the model of the objective function $f_i(x)$ and staying close to a point x_i^+ which is known to be good. We may offer the following interpretation of the entire process: at each iteration we carry out a search in the neighborhood of the current prox center; if we can reach the improvement (test of the Serious Step checks it) we choose the current iterate point as our new prox center and its neighborhood will be explored at the next iteration. If the improvement was not attained (Null Step) we continue searching the other directions around the same prox center. There are various versions of bundle methods and they differ from each other by explicit or implicit rules for updating the prox centers and the prox coefficients.

4 The Bundle Level method

In this section we review the Bundle Level method from the bundle family due to Lemaréchal, Nemirovsky and Nesterov (1991) [4]. The method possesses the optimal complexity bound $O(1/\varepsilon^2)$.

To describe the method, we must introduce several quantities. Given the i -th model $f_i(\cdot)$, we can find its minimal value

$$f_i^- = \min_{x \in X} f_i(x).$$

Since the model underestimates the objective function, the quantity f_i^- is a lower bound for the actual optimal value f_* and since the models increase with i at every point, their minima also increase, so we have

$$f_1^- \leq f_2^- \leq \dots \leq f_*.$$

Let f_i^+ be the best value of the objective function found so far:

$$f_i^+ = \min_{1 \leq j \leq i} f(x_j).$$

The quantities f_i^+ clearly decrease with i and overestimate the actual optimal value:

$$f_1^+ \geq f_2^+ \geq \dots \geq f_*.$$

It follows that the *gaps*

$$\Delta_i = f_i^+ - f_i^-$$

are nonnegative, nonincreasing and bound from above the inaccuracy of the best approximate solutions found so far:

$$\min_{1 \leq j \leq i} f(x_j) - f_* \leq \Delta_i, \quad \Delta_1 \geq \Delta_2 \geq \dots \geq 0.$$

We can now describe the method. Its i -th step is as follows:

- 1) *find the minimal value f_i^- of i -th model*

$$\text{minimize } f_i(x) \text{ s.t. } x \in X.$$

- 2) *form the level*

$$l_i = (1 - \lambda)f_i^- + \lambda f_i^+ \equiv f_i^- + \lambda \Delta_i,$$

where $\lambda \in (0, 1)$ is a parameter of the method (normally, $\lambda = 1/2$).

3) define the new iterate x_{i+1} as the projection of the previous one x_i onto the level set of the i -th model:

$$x_{i+1} = \operatorname{argmin}\{\|x - x_i\| \mid f_i(x) \leq l_i, x \in X\}.$$

Outside the level set of the i -th model $\{x \in X, f_i(x) \leq l_i\}$ the objective function f is larger than l_i , where l_i is the current level.

Computationally, the method requires solving two auxiliary problems at each iteration. The first is to minimize the model in order to compute f_i^- and the second is to project x_i onto the level set $\{x \in X \mid f_i(x) \leq l_i\}$. If X is a polytope, the first one is a linear programming problem, and the second is a linearly constrained convex quadratic programming.

Now we want to show that the level method actually belongs to the aforementioned bundle family when the prox center is chosen as the last iterate. Consider the iterate point in the bundle scheme

$$x(d) = \operatorname{argmin}_{x \in X} \left\{ f_i(x) + \frac{d}{2} \|x - x_i\|^2 \right\}$$

as a function of the prox-coefficient d . Due to the definition of $x(d)$, for every point in the level set $\{x \in X \mid f_i(x) \leq f_i(x(d))\}$ we have

$$\begin{aligned} f_i(x(d)) + \frac{d}{2} \|x(d) - x_i\|^2 &\leq f_i(x) + \frac{d}{2} \|x - x_i\|^2 \\ \frac{d}{2} \|x(d) - x_i\|^2 &\leq \frac{d}{2} \|x - x_i\|^2 + \underbrace{f_i(x) - f_i(x(d))}_{\leq 0} \end{aligned}$$

$$\frac{d}{2} \|x(d) - x_i\|^2 \leq \frac{d}{2} \|x - x_i\|^2.$$

It is clear now that $x(d)$ is the projection of x_i onto the level set of the i -th model $\{x \in X \mid f_i(x) \leq l_i(d)\}$ with the level $l_i(d) = f_i(x(d))$. As d varies from zero to the infinity, the point $x(d)$ moves along certain path which starts at the minimum of the i -th model and ends at the prox center x_i ; consequently, the level $l_i(d) = f_i(x(d))$ varies from f_i^- to $f_i(x_i) = f(x_i) \geq f_i^+$. Therefore the certain value d_i of the prox coefficient corresponds to the certain level $l_i(d) = l_i$ and, consequently, $x(d) = x_{i+1}$, i.e., the projection onto the level set of i -th model of the objective function is equal to choice the value of the prox coefficient in the usual bundle scheme. Thus we have demonstrated that the level method belongs to the bundle family with certain implicit control of the prox coefficient.

The complexity of the level algorithm is given by the following result.

Proposition 4.1. Consider the convex problem (P) with X convex compact set of diameter $\text{Diam}(X)$. Then the efficiency estimate for the level method is

$$\min_{1 \leq i \leq N} f(x_i) - \min_{x \in X} f(x) \leq c(\lambda) \frac{L_f \text{Diam}(X)}{\sqrt{N}},$$

where L_f is the Lipschitz constant for f and $c(\lambda) = ((1 - \lambda)^2 \lambda (2 - \lambda))^{-1}$ with $\lambda \in (0, 1)$.

5 Non Euclidean Restricted Memory Level method (NERML)

The recent *Non Euclidean Restricted Memory Level* method due to Ben-Tal and Nemirovsky [1] can be applied for solving the convex minimization problem (P). The main characteristic features of the method are the possibility to adjust the scheme to the geometry of the feasible set and flexible handling of the accumulated information. The aforementioned level method can be viewed as a particular case of the NERML. This connection can be seen from the following description of a step:

In the bundle level method the next search point x_{i+1} is given by

$$x_{i+1} = \operatorname{argmin} \left\{ \frac{1}{2} \|x - p_i\|^2 \mid x \in X, A_i x \leq b_i \right\}$$

where $p_i = x_i$ is the current *prox-center*, and the linear inequalities $A_i x \leq b_i$ are such that outside the set $\{x \in X, A_i x \leq b_i\}$ the objective function f is larger than l_i , where l_i is the current *level*. In a NERML method, x_{i+1} is given by

$$x_{i+1} = \operatorname{argmin} \{ \omega(x) - x^T \nabla \omega(p_i) \mid x \in X, A_i x \leq b_i \} \quad (5.1)$$

where $\omega(x)$ is a continuously differentiable strongly convex function on X . The point x_{i+1} defined by (5.1) can be viewed as a noneuclidean projection onto the level set of the i -model. It is immediately seen that level method is a particular case of NERML corresponding to $\omega(x) = \frac{1}{2} x^T x$. The appropriate choice of ω allows to adjust NERML to the geometry of the feasible set X .

Before we describe the NERML method we need to recall the following definition: $\omega(x): X \rightarrow R$ is *strongly convex* on X , with parameter $k > 0$ if

$$\omega(y) \geq \omega(x) + (y - x)^T \nabla \omega(x) + \frac{k}{2} \|y - x\|^2 \quad \forall x, y \in X.$$

We assume that we have access to a first order oracle which for every given input point $x \in X$, returns the value $f(x)$ and a subgradient $f'(x)$ of f at x .

The generic algorithm NERML works as follows.

A. The algorithm generates a sequence of search points belonging to X , (for which the first order oracle is called) and at every step builds the following quantities:

1. the best value of f found so far, along with the corresponding search point; the latter is treated as the current approximate solution built by the method;
2. a lower bound on the optimal value of the problem.

B. The execution is split into *phases*. Phase $s, s = 1, 2, \dots$, is associated with a *prox-center* $c_s \in X$ and a *level* $l_s \in R$ such that

- when starting the phase, we already know $f(c_s), f'(c_s)$;
- the level $l_s = f_s + \lambda(f^s - f_s)$, where
 - f^s is the best value of f known at the time when the phase starts;
 - f_s is the lower bound on the optimal value f_* that we have when the phase starts;
 - $\lambda \in (0, 1)$ is a parameter of the method.

To initialize the first phase we can choose the prox-center $c_1 \in X$ in an arbitrary fashion. We start the entire process by computing $f(c_1)$ and $f'(c_1)$ which results in

$$f^1 = f(c_1)$$

and set

$$f_1 = \min_{x \in X} [f(c_1) + (x - c_1)^T f'(c_1)],$$

which is the initial lower bound on f_* .

C. The description of a particular phase s is as follows. Let

$$\omega_s(x) = \omega(x) - (x - c_s)^T \nabla \omega(c_s);$$

note that strong convexity of $\omega(x)$ implies that

$$\omega_s(y) \geq \omega_s(x) + (y-x)^T \nabla \omega_s(x) + \frac{k}{2} \|y-x\|^2 \quad \forall x, y \in X.$$

Note also that $c_s = \operatorname{argmin}_{x \in X} \omega_s(x)$.

At phase s , the search points $x_t = x_{t,s}$, $t = 1, 2, \dots$ are generated according to the following rules:

1. When generating x_t , we already have in our disposal x_{t-1} , a lower bound $\tilde{f}_t = \tilde{f}_{s,t}$ on f_* and a *localizer* X_{t-1} — a convex compact set $X_{t-1} \subseteq X$ such that

$$\begin{aligned} (a_{t-1}) \quad & x \in X \setminus X_{t-1} \Rightarrow f(x) > l_s; \\ (b_{t-1}) \quad & x_{t-1} \in \operatorname{arg min}_{X_{t-1}} \omega_s. \end{aligned}$$

Here $x_0 = c_s$, $\tilde{f}_0 = f_s$ and $X_0 = X$, which ensures a_0, b_0 .

2. To update (x_{t-1}, X_{t-1}) into (x_t, X_t) , we solve the auxiliary problem

$$\tilde{f} = \min_x \{g_{t-1}(x) \equiv f(x_{t-1}) + (x - x_{t-1})^T f'(x_{t-1}) : x \in X_{t-1}\}. \quad (L_{t-1})$$

Observe that the quantity

$$\hat{f} = \min[\tilde{f}, l_s]$$

is a lower bound on f_* . Indeed, for $x \in X \setminus X_{t-1}$ we have $f(x) > l_s$ by a_{t-1} , while on X_{t-1} we have $f(x) \geq \tilde{f}$ due to the inequality $f(x) \geq g_{t-1}(x)$ given by the convexity of f . Thus, $f(x) \geq \min[l_s, \tilde{f}]$ everywhere on X , so that the quantity

$$\tilde{f}_t = \max[\tilde{f}_{t-1}, \min[l_s, \tilde{f}]]$$

is a new lower bound on f_* .

The future actions depend on the result obtained when solving (L_{t-1}) . Specifically:

- (a) In the case of "significant progress in the lower bound", namely

$$\tilde{f}_t \geq l_s - \theta(l_s - f_s),$$

where $\theta \in (0, 1)$ is a parameter of the method, we terminate phase s , set

$$f^{s+1} = \min[f^s, \min_{0 \leq \tau \leq t-1} f(x_\tau)], \quad f_{s+1} = \tilde{f}_t$$

and pass to phase $s + 1$. The prox-center c_{s+1} for the new phase can be chosen in X in an arbitrary fashion.

- (b) In the case of no significant progress in the lower bound, we solve the optimization problem

$$\min_x \{\omega_s(x) : x \in X_{t-1}, g_{t-1}(x) \leq l_s\}. \quad (P_{t-1})$$

x_t is defined to be the optimal solution of the above problem. We then compute $f(x_t), f'(x_t)$. It is possible that

- (b.1) We get a "significant" progress in the objective function, specifically,

$$f(x_t) - l_s \leq \theta(f^s - l_s).$$

In this case, we terminate the phase, set

$$f^{s+1} = \min[f^s, \min_{0 \leq \tau \leq t} f(x_\tau)], \quad f_{s+1} = \tilde{f}_t$$

and pass to phase $s + 1$. The prox-center c_{s+1} for the new phase can be chosen in X in an arbitrary fashion.

- (b.2) In the case of no significant progress in the objective function, we continue the phase s , choosing as X_t an arbitrary convex compact set such that

$$\begin{aligned} \underline{X}_t &\equiv \{x \in X_{t-1} : g_{t-1}(x) \leq l_s\} \subseteq X_t \subseteq \overline{X}_t \\ &\equiv \{x \in X : (x - x_t)^T \nabla \omega_s(x_t) \geq 0\}. \end{aligned}$$

The summary of the NERML method is as follows:

NERML algorithm

Parameters: $\lambda \in (0, 1), \theta \in (0, 1), \varepsilon > 0.$

Initialization: Choose $c_1 \in X$, compute $f(c_1)$ and $f'(c_1)$ and set

$$f^1 = f(c_1), \quad f_1 = \min_{x \in X} [f(c_1) + (x - c_1)^T f'(c_1)].$$

Phase $s(s=1,2,\dots)$: If $f^s - f_s \leq \varepsilon$ (required tolerance), terminate. Otherwise set $l_s = f_s + \lambda(f^s - f_s)$, start inner iterations:

Initialization: $x_0 = c_s, \tilde{f}_0 = f_s, X_0 = X;$

Inner iteration $t(t=1,2,\dots)$:

Compute

$$\begin{aligned} \tilde{f} &= \min_{x \in X_{t-1}} [f(x_{t-1}) + (x - x_{t-1})^T f'(x_{t-1})], \\ \tilde{f}_t &= \max[\tilde{f}_{t-1}, \min[l_s, \tilde{f}]]. \end{aligned}$$

If $\tilde{f}_t \geq l_s - \theta(l_s - f_s)$,
set

$$f^{s+1} = \min\{f^s, \min_{0 \leq \tau \leq t-1} f(x_\tau)\}, \quad f_{s+1} = \tilde{f}_t,$$

choose $c_{s+1} \in X$ and pass to phase $s + 1$.

else compute

$$x_t = \arg \min_{x \in X_{t-1}} \{\omega(x) - (x - c_s)^T \nabla \omega(c_s) : f(x_{t-1}) + (x - x_{t-1})^T f'(x_{t-1}) \leq l_s\}$$

and $f(x_t), f'(x_t)$.

If $f(x_t) - l_s \leq \vartheta(f^s - l_s)$

choose $c_{s+1} \in X$ and pass to phase $s + 1$,

else

choose X_t as any convex compact set satisfying the following

$$\begin{aligned} \underline{X}_t &\equiv \{x \in X_{t-1} : f(x_{t-1}) + (x - x_{t-1})^T f'(x_{t-1}) \leq l_s\} \subseteq X_t \subseteq \overline{X}_t \\ &\equiv \{x \in X : (x - x_t)^T (\nabla \omega(x_t) - \nabla \omega(c_s)) \geq 0\}. \end{aligned}$$

and pass to step $t + 1$ of phase s .

The similar interpretation given for the bundle method can be applied to NERML. Now the phase of NERML can be viewed as Null Step. Indeed, during the phase we explore the neighborhood of the same prox center and just in case of significant improvement toward the optimal value we start Serious Step, explore the neighborhood of the new prox center.

Let us define s -th gap as the quantity

$$\varepsilon_s = f^s - f_s.$$

By its definition, the gap is nonnegative in s , and is a valid upper bound on the inaccuracy. The complexity results of the NERML method are given by the following result.

Proposition 5.1. (1) *The number N_s of oracle calls at a phase s is bounded from above as follows:*

$$N_s \leq \frac{4\Omega L_f^2}{\theta^2(1-\lambda)^2 k \varepsilon_s^2}$$

where k is strong convexity parameter of function $\omega(\cdot)$ and

$$\Omega = \max_{x,y \in X} [\omega(y) - \omega(x) - (y-x)^T \nabla \omega(x)].$$

(2) *Consequently, for every $\varepsilon > 0$, the total number of oracle calls, before the first phase s for which $\varepsilon_s \leq \varepsilon$ is started (i.e., before an ε -solution to the problem is built) does not exceed*

$$N(\varepsilon) = c(\theta, \lambda) \frac{\Omega L_f^2}{k \varepsilon^2}$$

with an appropriate $c(\theta, \lambda)$ depending solely and continuously on $\theta, \lambda \in (0, 1)$.

5.1 Implementation issues

In this section we describe how to solve efficiently the auxiliary problems $(L_t), (P_t)$. Assume that $X_{t-1} \subseteq X$ given by a finite list of linear inequalities. Then the sets \underline{X}_t and \overline{X}_t are also cut off X by finitely many linear inequalities, so that we may enforce X_t to be cut off X by finitely many linear inequalities as well. Moreover, we can limit the number of the inequalities in the list. Indeed,

- A. Setting all the time $X_t = \overline{X}_t$, we ensure that X_t is cut off X by a single linear inequality;
- B. Setting all the time $X_t = \underline{X}_t$, we ensure that X_t is cut off X by t linear inequalities;
- C. We can choose something in-between the above extremes. We want to limit the number of linear inequalities by m . Until the number of linear inequalities in the description of X_{t-1} reaches the maximum allowed value m , we could use the policy B, so that

$$X_{t-1} = \{x \in X : h_j^{t-1}(x) \leq 0, j = 1, \dots, m\}.$$

At step t , we should choose X_t in-between the two sets $\underline{X}_t, \overline{X}_t$, where

$$\begin{aligned} \underline{X}_t &= \{x \in X : h_1^{t-1}(x) \leq 0, \dots, h_m^{t-1}(x) \leq 0, h_{m+1}^{t-1}(x) \leq 0\}, \\ \overline{X}_t &= \{x \in X : h_m^t(x) \leq 0\}, \end{aligned}$$

$$h_{m+1}^{t-1}(x) \equiv g_{t-1}(x) - l_s,$$

$$(*) h_m^t(x) \equiv (x_t - x)^T (\nabla \omega_s(x_t)).$$

To this end, we can set

$$X_t = \{x \in X : h_j^t(x) \leq 0, j = 1, \dots, m\},$$

where $h_m^t(x)$ is given by (*), and every one of the inequalities $h_j^t(x) \leq 0, j = 1, \dots, m$, is a convex combination of the inequalities $h_1^{t-1}(x) \leq 0, \dots, h_m^{t-1}(x) \leq 0, h_{m+1}^{t-1}(x) \leq 0, h_m^t(x) \leq 0$.

Thus we can always ensure that X_{t-1} is cut off X by at most m linear inequalities, where m is a desirable bound. We may assume now that the *feasible set of* (P_t) *is cut off* X *by* $m + 1$ *linear inequalities* $h_j(x) \leq 0, j = 1, \dots, m + 1$. The crucial point is that with this approach *we can reduce* $(L_t), (P_t)$ *to convex programs with at most* $m + 1$ *decision variables.*

Assume that the problem (P_{t-1}) is strictly feasible, i.e., $\exists \tilde{x} \in \text{rint}X : h_j(\tilde{x}) \leq 0, j = 1, \dots, m + 1$ or $(\underline{X}_t \cap \text{rint}X \neq \emptyset)$. By standard Lagrange Duality, the optimal value of (P_{t-1}) is equal to the value of its dual problem

$$\max_{\lambda \geq 0} L(\lambda), \quad L(\lambda) \equiv \min_{x \in X} [\omega_s(x) + \sum_{j=1}^{m+1} \lambda_j h_j(x)]. \quad (D_{t-1})$$

Note that the dual objective function $L(\lambda)$ is concave function. Computing the value $L(\lambda)$ and a supergradient $L'(\lambda)$ of L at a given λ amounts to finding the optimal solution x_λ of the optimization problem

$$\min_{x \in X} [\omega_s(x) + \sum_{j=1}^{m+1} \lambda_j h_j(x)]; \quad (D[\lambda]).$$

After x_λ is found, we set

$$L(\lambda) = \omega_s(x) + \sum_{j=1}^{m+1} \lambda_j h_j(x_\lambda), \quad L'(\lambda) = (h_1(x_\lambda), \dots, h_{m+1}(x_\lambda))^T.$$

After (D_{t-1}) is solved and we have in our disposal the corresponding maximizer λ_* , we can choose, as x_t , the point x_{λ_*} , since by Lagrange duality theorem the optimal solution x_t of (P_{t-1}) is among the optimal solutions to $(D[\lambda_*])$, and ω_s is strongly convex, the set of the optimal solutions to $(D[\lambda_*])$ is a singleton.

It remains to understand how to solve (L_{t-1}) and how to ensure the strict feasibility of (P_{t-1}) . Here we can again apply the Lagrangian Duality. Assuming that (L_{t-1}) is strictly feasible, i.e., $X_{t-1} \cap \text{rint}X \neq \emptyset$, we have

$$\begin{aligned} & \min_x \{g_{t-1}(x) : x \in X_{t-1} = \{x \in X : h_j(x) \leq 0, j = 1, \dots, m\}\} \\ & = \max_\lambda \{L(\lambda) \equiv \min_{x \in X} [g_{t-1}(x) + \sum_{j=1}^m \lambda_j h_j(x)] : \lambda \geq 0\}. \end{aligned}$$

We can solve the above low dimensional problem by the Ellipsoid or the bundle methods. In this case we want just the optimal value, not an optimal solution, hence the fact that the objective function (L_{t-1}) is not strongly convex does not cause any difficulties. If the optimal value in (L_{t-1}) is at least l_s , we must terminate the phase and hence do not need to solve (P_{t-1}) at all, otherwise the set $\underline{X}_t = \{x \in X_{t-1} : g_{t-1}(x) \leq l_s\}$ clearly intersects $\text{rint}X$ (since X_{t-1} is assumed to possess this property) and thus (P_{t-1}) is strictly feasible and we can solve the problem via duality.

6 The Reduced Gradient method

The convex optimization problem (P) can be solved by the Reduced Gradient algorithm. The attractive feature of this method is its simplicity. We assume that the objective function is convex and also twice continuously differentiable on the convex compact domain X . The Reduced Gradient method (RG) generates the sequence of iterates $x_t \in X$ according to the following scheme:

Initialization:

1. Choose an arbitrary point $x_0 \in X$
2. Find

$$x_1 = \hat{x}_0 \in \operatorname{argmin}_{x \in X} [f(x_0) + (x - x_0)^T f'(x_0)]$$

Step t :

1. Find

$$\hat{x}_t \in \operatorname{argmin}_{x \in X} [f(x_t) + (x - x_t)^T f'(x_t)]$$

2. Set

$$x_{t+1} = x_t + \frac{\gamma}{t+1} (\hat{x}_t - x_t)$$

where $\gamma \in (1, 2]$ is a parameter of the method.

The efficiency estimate of the reduced gradient method is given by the following result.

Proposition 6.1. *Consider the convex minimization problem (P) with X convex compact and denote by $L_X(f)$ the Lipschitz constant of the gradient mapping $f'(x)$. Then the efficiency estimate for the RG method is*

$$f(x_t) - \min_{x \in X} f(x) \leq \frac{L_X(f)\gamma^2}{2(\gamma - 1)} t^{-1}.$$

7 Application to Shape Design

In this section we consider the implementation of the described methods to large-dimensional structural design problem. Structural design is an engineering area dealing with mechanical constructions like trusses and plates. A plate is a construction made up of a material occupying a given domain; the mechanical properties of the material varying continuously from point to point. In engineering, design of plates is called shape design. A typical structural design (SD) problem is: "Given the type of material to be used, an upper bound on the material to be used and a set of loading scenarios - external loads operating on the construction. We need to find an optimal truss or shape, one able to withstand best of all the loads in question." It turns out that structural design problems can be cast as large-dimensional semidefinite problem and then solved by the aforementioned algorithms: Reduced Gradient and NERML. For each of the methods we define the appropriate model of the SD problem and describe in details how the algorithms can be applied to the problem.

7.1 Problem Formulation

Consider a shape specified by a collection $t = \{t_i\}_{i=1}^N$ of positive definite matrices $t_i \in S_{++}^d$; t_i is the rigidity tensor of the shape in finite element cell i multiplied by the area/volume of the cell. The twice compliance of the shape w.r.t. a load f is

$$C_f(t) = f^T(A(t))^{-1}f, \quad A(t) = \sum_{i,s} b_{is}^T t_i b_{is}.$$

The multi-load design problem is

$$\min_{t \in \mathcal{T}} \max_{1 \leq k \leq K} \ln(f_k^T(A(t))^{-1}f_k), \quad (7.1)$$

where \mathcal{T} is a given closed convex set $\{t = \{t_i\}_{i=1}^N : t_i \succeq 0, \sum_i \text{Tr}(t_i) = 1\}$. Consider the following smoothing approximation of this problem:

$$\min_{t \in \mathcal{T}} F_\beta(t), \quad F_\beta(t) = \beta^{-1} \ln \left(\sum_{k=1}^K (f_k^T(A(t))^{-1}f_k)^\beta \right). \quad (7.2)$$

where $F_\beta(t)$ is convex and smooth function, twice continuously differentiable on X . Observe that

$$\max_k \ln(f_k^T(A(t))^{-1}f_k) \leq F_\beta(t) \leq \max_k \ln(f_k^T(A(t))^{-1}f_k) + \beta^{-1}\ln K.$$

It follows that for sufficiently big value of β we obtain a good approximation of the objective function in (7.1) and thus we can substitute the original objective function in (7.1) by the function $F_\beta(t)$. We shall use $F_\beta(t)$ as our objective function in both methods: RG and NERML.

7.2 Solving the SD problem by Reduced Gradient method

Let \bar{t} be a collection of positive definite matrices from S^d and let $\alpha > 1$. Nemirovsky proposed a special heuristic to solve the multi-load shape problem by using RG method which we now describe in details. Consider the convex compact set

$$Y = Y[\bar{t}, \alpha] = \{t = \{t_i\}_{i=1}^N : \alpha^{-1}\bar{t}_i \preceq t_i \preceq \alpha\bar{t}_i, 1 \leq i \leq N\}.$$

The problem to be solved is

$$\min_{t \in X} \beta^{-1} \ln \left(\sum_{k=1}^K (f_k^T(A(t))^{-1}f_k)^\beta \right) \quad (7.3)$$

where

- t is a collection of N $d \times d$ positive definite symmetric matrices ($d = 3$ for planar shapes and $d = 6$ for spatial shapes);
- $X = \mathcal{T} \cap Y = \{t = \{t_i\}_{i=1}^N : \sum_i \text{Tr}(t_i) = 1, \alpha^{-1}\bar{t}_i \preceq t_i \preceq \alpha\bar{t}_i\}$;
- $f_k, k = 1, \dots, K$, are given M -dimensional vectors;
-

$$A(t) = \sum_{i=1}^N \sum_{s=1}^S b_{is}^T t_i b_{is},$$

where b_{is} are given $d \times M$ matrices.

Heuristic. We start with certain initial guess \bar{t} for the design (say, the unit matrix) and fix a moderate α (say, $\alpha = 2$ or $\alpha = 10$). We then solve the problem (7.3) by RG within a moderate accuracy; then we take the resulting approximate solution as \bar{t} ("re-centering") and solve the same problem (7.3) defined on the new domain, and so on.

Implementation details.

At each iteration of the RG algorithm we solve the following problem:

$$\begin{aligned} \min \quad & \sum_{i=1}^N \text{Tr}(u_i t_i) \\ \text{s.t.} \quad & \sum_{i=1}^N \text{Tr}(t_i) = 1, \\ & \alpha^{-1} \bar{t}_i \preceq t_i \preceq \alpha \bar{t}_i, \end{aligned} \tag{7.4}$$

where u_i are given symmetric matrix derived from $F'_\beta(\hat{t})$ (\hat{t} is the last generated iterate point, i.e., collection of positive definite matrices). To solve (7.4), we consider the Cholesky factorization of \bar{t}_i (recall that \bar{t}_i is positive definite), $\bar{t}_i = R_i^T R_i$. Define the following matrices:

$$\begin{aligned} g_i &= R_i u_i R_i^T, \\ h_i &= R_i R_i^T, \end{aligned}$$

and the new matrix variable

$$s_i = (R_i^T)^{-1} t_i R_i^{-1}.$$

Now we can rewrite the problem (7.4) as follows:

$$\begin{aligned} \min \quad & \sum_{i=1}^N \text{Tr}(g_i s_i) \\ \text{s.t.} \quad & \sum_{i=1}^N \text{Tr}(h_i s_i) = 1, \\ & \alpha^{-1} I \preceq s_i \preceq \alpha I, \end{aligned} \tag{7.5}$$

Here we can apply the Lagrangian Duality. Define the dual function $L(\lambda)$:

$$L(\lambda) \equiv \min_{\alpha^{-1}I \preceq s_i \preceq \alpha I} \sum_{i=1}^N \text{Tr}(g_i s_i) + \lambda \left(\sum_{i=1}^N \text{Tr}(h_i s_i) - 1 \right).$$

We need to solve the following max-min problem:

$$\max_{\lambda \in R} L(\lambda) \equiv \max_{\lambda \in R} \min_{\alpha^{-1}I \preceq s_i \preceq \alpha I} \left(\sum_{i=1}^N \text{Tr}((g_i + \lambda h_i) s_i) - \lambda \right) \quad (7.6)$$

The function $L(\lambda)$ is linear piecewise concave function of one variable. We can approximately find the optimal solution λ_* of (7.6) by bisection. Therefore we need to compute derivatives of $L(\lambda)$ at the two extreme points of the intervals $[\lambda_1, \lambda_2]$. To compute the derivative $L'(\lambda)$ for given point λ we solve the following subproblems. For each i we need to find the solution $s_i(\lambda)$ of the problem:

$$\begin{aligned} \min \quad & \text{Tr}((g_i + \lambda h_i) s_i) \\ \text{s.t.} \quad & \alpha^{-1}I \preceq s_i \preceq \alpha I \end{aligned} \quad (7.7)$$

and then we can compute $L'(\lambda) = \sum_{i=1}^N \text{Tr}(h_i s_i(\lambda) - 1)$.

Remark. Note that (7.7) can be easily transformed to the simple linear problem. Since the matrix $g_i + \lambda h_i$ is symmetric, by the eigenvalue decomposition we obtain the diagonal matrix $d(\lambda) = V^T(g_i + \lambda h_i)V$, where V is orthogonal. Define the new variable $x = V^T s_i V$. The problem (7.7) can be rewritten as the following program:

$$\begin{aligned} \min \quad & \text{Tr}(d(\lambda)x) \\ \text{s.t.} \quad & \alpha^{-1}I \preceq x \preceq \alpha I \end{aligned} \quad (7.8)$$

Among optimal solutions of the (7.8) there is clearly exists diagonal matrix. Thus we obtain the linear problem:

$$\begin{aligned} \min \quad & \text{Tr}(d(\lambda)x) \\ \text{s.t.} \quad & \alpha^{-1}I \leq x \leq \alpha I \end{aligned} \quad (7.9)$$

where x is a diagonal matrix.

After we find the optimal solution $s_i(\lambda_*)$ of the problem (7.5), we compute the optimal solution of the original problem (7.4) by $t_i(\lambda_*) = R_i^T s_i(\lambda_*) R_i$.

7.3 Implementation of NERML to SD problems

The problem is:

$$\min_{t \in \mathcal{T}} \beta^{-1} \ln \left(\sum_{k=1}^K (f_k^T (A(t))^{-1} f_k)^\beta \right)$$

where $\mathcal{T} = \{t = \{t_i\}_{i=1}^N : t_i \succeq 0, \sum_i \text{Tr}(t_i) = 1\}$ is a given closed convex set. It follows that implementing NERML to SD problems amounts to solving the following problems:

1.

$$\begin{aligned} \min \quad & \sum_{i=1}^N \text{Tr}(u_i^m t_i) \\ \text{s.t.} \quad & \sum_{i=1}^N \text{Tr}(u_i^j t_i) \leq \tilde{l}_j \quad j = 1, \dots, m-1, \\ & \sum_{i=1}^N \text{Tr}(t_i) = 1, \\ & t_i \succeq 0, \quad i = 1, \dots, N, \end{aligned} \tag{7.10}$$

where m is a given integer value (the size of the current bundle) and u_i^j are given symmetric matrices.

2.

$$\begin{aligned} \min \quad & \sum_{i=1}^N \text{Tr}(t_i^2) - 2 \sum_{i=1}^N \text{Tr}(u_i^0 t_i) \\ \text{s.t.} \quad & \sum_{i=1}^N \text{Tr}(u_i^j t_i) \leq \tilde{l}_j \quad j = 1, \dots, m, \\ & \sum_{i=1}^N \text{Tr}(t_i) = 1, \\ & t_i \succeq 0, \quad i = 1, \dots, N \end{aligned} \tag{7.11}$$

Implementation details.

1. In order to solve the first subproblem, we consider the Lagrangian dual problem:

$$\max_{\lambda \in R_+^{m-1}} L(\lambda), \quad L(\lambda) \equiv \min_{t \in \mathcal{T}} \left(\sum_{i=1}^N \text{Tr}(u_i^m t_i) + \sum_{j=1}^{m-1} \lambda_j \left(\sum_{i=1}^N \text{Tr}(u_i^j t_i) - \tilde{l}_j \right) \right).$$

This is a small problem with $m - 1$ variables. This reduced problem can be solved by the bundle method. To do it, we need to be able to compute the value of the function $L(\lambda)$ and its supergradient $L'(\lambda)$. In what follows we shall describe how it can be done.

The objective function $L(\lambda)$ can be obtained in the following form:

$$L(\lambda) = \min_{t \in \mathcal{T}} \sum_{i=1}^N \text{Tr}(D_i(\lambda) t_i) - \sum_{j=1}^{m-1} \lambda_j \tilde{l}_j$$

where $D_i(\lambda) = u_i^m + \sum_{j=1}^{m-1} \lambda_j u_i^j$ is a symmetric matrix. Using the eigenvalue decomposition $D_i(\lambda) = V_i d_i(\lambda) V_i^T$, we get the diagonal matrix $d_i(\lambda)$. Define new matrix variable $s_i = V_i^T t_i V_i$; to compute the value of $L(\lambda)$ at a given point λ , we should solve the following linear problem:

$$\begin{aligned} \min \quad & \sum_{i=1}^N \text{Tr}(d_i(\lambda) s_i) - \sum_{j=1}^{m-1} \lambda_j \tilde{l}_j \\ \text{s.t.} \quad & \sum_{i=1}^N \text{Tr}(s_i) = 1, \\ & s_i \succeq 0, \quad i = 1, \dots, N, \end{aligned} \tag{7.12}$$

where s_i is a diagonal matrix (the explanation why the inequality $s_i \succeq 0$ can be replaced by $s_i \geq 0$ was given in the previous Remark). Denote by $s_i(\lambda)$ the optimal solution of (7.12) at given point λ . To obtain the supergradient $L'(\lambda)$ we substitute $t_i(\lambda) = (V_i^T)^{-1} s_i(\lambda) V_i^{-1}$ into the following expression:

$$L'(\lambda) = \left(\sum_{i=1}^N \text{Tr}(u_i^1 t_i(\lambda)) - \tilde{l}_1, \dots, \sum_{i=1}^N \text{Tr}(u_i^{m-1} t_i(\lambda)) - \tilde{l}_{m-1} \right).$$

2. We can solve the second problem using the same arguments. The dual problem here is the following:

$$\max_{\lambda \in R_+^{m-1}} L(\lambda), \quad L(\lambda) \equiv \min_{t \in \mathcal{T}} \left(\sum_{i=1}^N \text{Tr}(t_i^2) - 2 \sum_{i=1}^N \text{Tr}(u_i^0 t_i) + \sum_{j=1}^m \lambda_j \left(\sum_{i=1}^N \text{Tr}(u_i^j t_i) - \tilde{l}_j \right) \right).$$

Define the variables s_i as previously. To obtain the value of $L(\lambda)$ at a given point λ , we should solve the following linearly constrained quadratic problem:

$$\begin{aligned} \min \quad & \sum_{i=1}^N \text{Tr}(s_i^2) - \sum_{i=1}^N \text{Tr}(d_i(\lambda)s_i) - \sum_{j=1}^m \lambda_j \tilde{l}_j \\ \text{s.t.} \quad & \sum_{i=1}^N \text{Tr}(s_i) = 1, \\ & s_i \geq 0, \quad i = 1, \dots, N, \end{aligned}$$

where $d_i(\lambda)$ is given diagonal matrix. Having in our disposal the optimal solution $t_i(\lambda)$, we can compute the supergradient via

$$L'(\lambda) = \left(\sum_{i=1}^N \text{Tr}(u_i^1 t_i(\lambda)) - \tilde{l}_1, \dots, \sum_{i=1}^N \text{Tr}(u_i^{m-1} t_i(\lambda)) - \tilde{l}_m \right).$$

8 Problems with functional constraints

In this section we describe a variant of the bundle level algorithm [4] that can solve the constrained optimization problems. The approach is based on the "level" idea, namely, definition of an appropriate nonnegative *gap*, which need to be minimized and is zero when the problem is solved.

We are interested in solving the following constrained optimization problem:

$$\min\{f(x)|g_i(x) \leq 0, i = 1, \dots, m, x \in X\}, \quad (8.1)$$

where $f(x)$ and $g_i(x)$ are convex functions on the compact convex set X . Assume first that we have one inequality constraint, denote it by $G(x)$. Rewrite the problem in the following form:

$$\min_{x \in X} \max[f(x) - f_*, G(x)], \quad (8.2)$$

where f_* denotes the optimal value of the constrained problem (8.1). The optimal value of (8.2) is obviously zero. An equivalent problem to (8.2) is

$$\begin{aligned} \min_{x \in X} \max[f(x) - f_*, G(x)] &= \min_{x \in X} \max_{0 \leq \alpha \leq 1} \{\alpha(f(x) - f_*) + (1 - \alpha)G(x)\} \\ &= \max\{h(\alpha) - \alpha f_* | 0 \leq \alpha \leq 1\}, \end{aligned} \quad (8.3)$$

where

$$h(\alpha) = \min_{x \in X} \{\alpha f(x) + (1 - \alpha)G(x)\}.$$

Clearly, zero is the optimal value and the 'lower bound' for (8.3). To get the gap, we need an upper bound. The function $h(\alpha)$ can be overestimated by the function

$$h_i(\alpha) = \min_{1 \leq j \leq i} \{\alpha f(x_j) + (1 - \alpha)G(x_j)\},$$

where $\{x_j\}_{j=1}^i$ is the sequence of the search points generated so far. We can also underestimate the value f_* by the optimal value f_i^- :

$$f_i^- = \min_{x \in X} \{f_i(x) | G_i(x) \leq 0\}$$

where $f_i(x)$ and $G_i(x)$ are i -th piecewise linear models of $f(x)$ and $G(x)$ (see below (8.4) and (8.5)). Thus we can underestimate the gap by

$$\Delta_i = \max_{0 \leq \alpha \leq 1} \{h_i(\alpha) - \alpha f_i^-\}.$$

The goal is to reduce the gap 0.

Assumptions on the data: $f(x)$ and $g_i(x)$, $i = 1, \dots, m$ are convex Lipschitz continuous functions on the convex compact set X ; L denotes the maximum of the Lipschitz constants of f, g_1, \dots, g_m ; D denotes the diameter of X with respect to the Euclidean norm $\|\cdot\|$ and $G \equiv \max\{g_1, \dots, g_m\}$. There is an oracle that for given $x \in X$ computes $f(x), g_1(x), \dots, g_m(x)$ and $f'(x), g'_1(x), \dots, g'_m(x)$. Accuracy measure at the point $x \in X$ is as follows:

$$\varepsilon(x) = \max\{f(x) - f_*, G(x)\}.$$

Define the following objects.

Model of f .

$$f_i(x) = \max_{1 \leq j \leq i} \{f(x_j) + (x - x_j)^T f'(x_j)\}. \quad (8.4)$$

Model of G .

$$G_i(x) = \max\{g_k(x_j) + (x - x_j)^T g'_k(x_j) | 1 \leq j \leq i, 1 \leq k \leq m\}. \quad (8.5)$$

Clearly,

$$\begin{aligned} f_1(x) &\leq f_2(x) \leq \dots \leq f_i(x) \leq f(x) \quad \forall x \in X, \\ G_1(x) &\leq G_2(x) \leq \dots \leq G_i(x) \leq G(x) \quad \forall x \in X, \\ f_i(x_j) &= f(x_j), \quad G_i(x_j) = G(x_j), \quad 1 \leq j \leq i. \end{aligned}$$

Define the *model's best value*.

$$f_i^- = \min_{x \in X} \{f_i(x) | G_i(x) \leq 0\}.$$

From the above statements it follows that f_i^- is well defined and

$$f_1^- \leq f_2^- \leq \dots \leq f_i^- \leq f_*.$$

Admissible set. $T(i) = \{(f(x_j), G(x_j)) | 1 \leq j \leq i\} \subset R^2$.

Completed admissible set. $C(i) = \text{Conv}T(i) + R_+^2$.

We will use the natural order in R^2 :

$$(u_1, v_1) \leq (u_2, v_2) \quad \text{if } u_1 \leq u_2 \quad \text{and} \quad v_1 \leq v_2.$$

8.1 Constrained Level Method (CLM)

Besides the above objects we need also to define the following.
support function:

$$h_i(\alpha) \equiv \min_{1 \leq j \leq i} \{\alpha(f(x_j) - f_i^-) + (1 - \alpha)G(x_j)\}$$

Gap

$$\Delta_i = \max\{h_i(\alpha) | 0 \leq \alpha \leq 1\}.$$

Best point. Let $(u(i), v(i)) \in \operatorname{argmin}\{\rho(u - f_i^-, v) | (u, v) \in C(i)\}$, where

$$\rho(a, b) = \max\{(a)_+, (b)_+\}.$$

The point $(u(i), v(i))$ belongs to set $C(i)$, so clearly there exists a convex combination $\sum_{j=1}^i r_i(j)(f(x_j), G(x_j))$ of points from $T(i)$, such that $\sum_{j=1}^i r_i(j)(f(x_j), G(x_j)) \leq (u(i), v(i))$. Set

$$x_i^* = \sum_{j=1}^i r_i(j)x_j$$

as the *best point* associated with x_1, \dots, x_i .

Remark 8.1. For the i -th best point $x_i^* \in X$ we have

$$\varepsilon(x_i^*) \leq \min\{\rho(u - f_i^-, v) | (u, v) \in C(i)\} = \Delta_i.$$

The inclusion is evident:

$$\begin{aligned} \Delta_i = \max\{h_i(\alpha) | 0 \leq \alpha \leq 1\} &= \max_{0 \leq \alpha \leq 1} \min_{1 \leq j \leq i} \{\alpha(f(x_j) - f_i^-) + (1 - \alpha)G(x_j)\} \\ &= \min_{1 \leq j \leq i} \max\{f(x_j) - f_i^-, G(x_j)\} \\ &= \min\{\rho(u - f_i^-, v) | (u, v) \in C(i)\}. \end{aligned}$$

The inequality follows from the relations

$$\begin{aligned} (f(x_i^*) - f_i^-, G(x_i^*)) &\leq \sum_{j=1}^i r_i(j)(f(x_j) - f_i^-, G(x_j)) & (8.6) \\ &\leq (u(i) - f_i^-, v(i)) \\ &\leq \min\{\rho(u - f_i^-, v) | (u, v) \in C(i)\}, \end{aligned}$$

where (8.6) follows from the convexity of f and G and the definition of the best point x_i^* . Since $f_* \geq f_i^-$ we obtain the required statement.

Description of CLM

Parameters. $\lambda, \mu \in (0, 1)$.

Initialization. x_1 an arbitrary point of X .

i -th step.

1. Call the oracle for x_i .
2. Compute $f_i^-, h_i(\cdot), \Delta_i, x_i^*$. Stop if at the point x_i^* we reach the required accuracy ε , i.e., if $\Delta_i \leq \varepsilon$.
3. Define $\alpha_{\min}(i)$ as the smallest, and $\alpha_{\max}(i)$ the largest of $\alpha \in [0, 1]$ such that $h_i(\alpha) \geq 0$.

For $i = 1$ set

$$\alpha(1) = \frac{1}{2}(\alpha_{\min}(1) + \alpha_{\max}(1)),$$

and for $i > 1$ set

$$\alpha(i) = \left\{ \begin{array}{ll} \frac{1}{2}(\alpha_{\min}(i) + \alpha_{\max}(i)), & \text{if } \frac{\alpha(i-1) - \alpha_{\min}(i)}{\alpha_{\max}(i) - \alpha_{\min}(i)} \in [\frac{1}{2}\mu, 1 - \frac{1}{2}\mu], \\ \alpha(i-1), & \text{otherwise.} \end{array} \right\}$$

4. Set

$$w(i) = \alpha(i)f_i^-, \quad W(i) = \min_{1 \leq j \leq i} (\alpha(i)f(x_j) + (1 - \alpha(i))G(x_j)),$$

$$l_i = w(i) + \lambda(W(i) - w(i)),$$

$$x_{i+1} = \operatorname{argmin}\{\|x - x_i\| \mid \alpha(i)f_i(x) + (1 - \alpha(i))G_i(x) \leq l_i, x \in X\}.$$

Solving the additional problems in the algorithm

In the CLM we need to solve several problems in each iteration.

1. Computation of

$$f_i^- = \min_{x \in X} \{f_i(x) \mid G_i(x) \leq 0\}.$$

This is the linear programming problem:

$$\begin{array}{ll} \min & t \\ \text{s.t.} & f(x_j) + (x - x_j)^T f'(x_j) \leq t, \quad \forall 1 \leq j \leq i \\ & g_k(x_j) + (x - x_j)^T g'_k(x_j) \leq 0, \quad \forall 1 \leq j \leq i, \forall 1 \leq k \leq m \end{array}$$

2. Computation of $\Delta_i = \max\{h_i(\alpha) | 0 \leq \alpha \leq 1\}$. The function

$$h_i(\alpha) \equiv \min_{1 \leq j \leq i} \{\alpha(f(x_j) - f_i^-) + (1 - \alpha)G(x_j)\}$$

is piecewise linear concave function, so finding its maximal value amounts to solving the linear problem

$$\begin{aligned} \max \quad & t \\ \text{s.t.} \quad & \alpha(f(x_j) - f_i^-) + (1 - \alpha)G(x_j) \geq t, \quad \forall 1 \leq j \leq i \\ & 0 \leq \alpha \leq 1. \end{aligned}$$

3. To obtain the new iterate point we have to solve the quadratic problem with linear constraints:

$$\begin{aligned} \min \quad & \|x - x_i\|^2 \\ \text{s.t.} \quad & \alpha(i)(f(x_j) + (x - x_j)^T f'(x_j)) \leq k, \quad \forall 1 \leq j \leq i \\ & (1 - \alpha(i))(g_k(x_j) + (x - x_j)^T g'_k(x_j)) \leq l_i - k, \quad \forall 1 \leq j \leq i, \forall 1 \leq k \leq m \end{aligned}$$

where $\alpha(i)$ and l_i are given values.

8.2 Constrained NERML (CONERML) method

The approach described in the previous section was adapted and extended by E.Olvovsky and A.Nemirovsky. It was shown in [5] how the NERML method can be applied to problems with functional constraints. The problem to be solved is the following:

$$\min\{f(x) | G(x) \leq 0, x \in X\}, \quad (8.7)$$

where the functions $f(x), G(x)$ are convex and Lipschitz continuous on the convex compact set X . We reformulate the problem by defining the function $q(x)$ as follows:

$$q(x) := \max\{f(x) - f_*, G(x)\}$$

where f_* is the optimal value of the constrained problem. The function $q(x)$ is Lipschitz continuous and convex on X as a maximum of such functions and the set of optimal solutions of

$$\min\{q(x), x \in X\}, \quad (8.8)$$

coincides with the set of optimal solutions of (8.7).

Now replace the unknown optimal value f_* by a parameter r and define the function:

$$q(x, r) := \max\{f(x) - r, G(x)\}.$$

Since, the optimal value of the problem (8.8) is zero, we need to solve the following equation:

$$\varepsilon(r) = \min\{q(x, r) | x \in X\} = 0. \quad (8.9)$$

The solution of (8.9) will be derived in subsequent cycles. For each cycle i we underestimate f_* by r_i using appropriate models $f_i(x), G_i(x)$ of $f(x)$ and $G(x)$ respectively. As a result we obtain the following increasing sequence:

$$r_1 \leq r_2 \leq \dots \leq f_*.$$

For a given r_i we solve

$$\min\{q(x, r_i) | x \in X\}. \quad (8.10)$$

Note that the function $\varepsilon(r)$ is decreasing for $r \leq f_*$, so if $\varepsilon(r) \leq \varepsilon$ for some r ($r \leq f_*$), then $\varepsilon(f_*) \leq \varepsilon$ too. So if $\varepsilon(r_i)$ is sufficiently small we stop and take r_i and the solution of (8.10) as approximations to f_* and the optimal solution to (8.7) respectively. Otherwise, we update r_i to r_{i+1} and so on.

Description of CONERML

Setup for Constrained NERML is similar to the one in the general NERML method. We also need to assume that we have access to a first order oracle computing the values of the functions $f(x)$ and $G(x)$ and their subgradients. Execution of the Constrained NERML algorithm as applied to (8.10) is partitioned into subsequent cycles.

Cycle i . At the beginning of the cycle i ($i = 1, 2, \dots$) we have in our disposal a valid lower bound r_i on the optimal value f_* in (8.7), and the prox-center c_i .

To initialize the very first cycle, we can choose an arbitrary point $c \in X$, compute $f(c), G(c), f'(c), G'(c)$ and set

$$f_1(x) = f(c) + (x - c)^T f'(c);$$

$$G_1(x) = G(c) + (x - c)^T G'(c);$$

$$r_1 = \min\{f_1(x) | G_1(x) \leq 0, x \in X\};$$

$$Q_1(x, r_1) = \max\{f_1(x) - r_1, G_1(x)\}.$$

At each cycle we solve problem (8.10) by the general NERML algorithm (see section 1.4). An initializing of the very first phase of the NERML can be done as follows:

The first phase $s = 1$ of the cycle i . Set

$$c_{i1} = c_i,$$

$$\bar{\delta}_{i1} = \max\{f(c_{i1}) - r_i, G(c_{i1})\},$$

$$\underline{\delta}_{i1} = \min\{Q_{i1}(x, r_i), x \in X\}.$$

At the beginning of the phase s we have in our disposal the following:

- a valid lower bound $\underline{\delta}_{is}$ on the quantity

$$\min\{q(x, r_i) | x \in X\};$$

- the best found so far feasible solution $\bar{\delta}_{is}$ to

$$\min\{q(x, r_i) | x \in X\};$$

- a prox center $c_{is} \in X$.

Let us define s -th level as follows

$$l_s = (1 - \lambda)\underline{\delta}_{is} + \lambda\bar{\delta}_{is}$$

where λ is a parameter of the method.

The step t of the phase s . In order to simplify the notations, we skip the phase and the cycle indexes. At the beginning of step t we have in our disposal

- t -th search point x_t of the phase;
- t -th model $Q_t(x, r_i)$ of the function $q(x, r_i)$, which is Lipschitz continuous piecewise linear convex function satisfying the relation

$$Q_t(x, r_i) \leq q(x, r_i) \quad \forall x \in X;$$

- t -th best found value $\bar{\delta}_{st}$ of the $q(x, r_i)$;
- t -th localizer, the convex compact set $X_t \subseteq X$.

The point x_t and the set X_t satisfy the following:

$$(a_t) \quad x_t = \operatorname{argmin}\{\omega_s(x) | x \in X_t\}$$

$$(b_t) \quad x \in X \setminus X_t \Rightarrow q(x, r_i) > l_s$$

To initialize *the first step* $t = 1$ of the phase s , we set

$$\begin{aligned} x_1 &= c_s, \\ X_1 &= X, \\ f_1(x) &= f(x_1) + (x - x_1)^T f'(x_1), \\ G_1(x) &= G(x_1) + (x - x_1)^T G'(x_1), \\ Q_1(x, r_i) &= \max\{f_1(x) - r_i, G_1(x)\}, \\ \underline{\delta}_{s1} &= \min\{Q_1(x), x \in X_1\}, \\ \bar{\delta}_{s1} &= \bar{\delta}_s. \end{aligned}$$

Our actions at *step* t are as follows:

1. updating the upper bound, enriching the model

Compute $f(x_t), G(x_t), f'(x_t), G'(x_t)$ and

$$q(x_t, r_i) = \max\{f(x_t) - r_i, G(x_t)\}.$$

Set

$$\bar{\delta}_{s,t+1} = \min\{\bar{\delta}_{s,t+1}, q(x_t, r_i)\}.$$

If

$$\bar{\delta}_{s,t+1} \leq \theta \bar{\delta}_s + (1 - \theta) l_s, \tag{8.11}$$

where $\theta \in (0, 1)$ is a parameter of the method. In this case we terminate the phase s and pass to phase $s + 1$, setting

$$\bar{\delta}_{s+1} = \bar{\delta}_{s,t+1}, \quad \underline{\delta}_{s+1} = \underline{\delta}_s.$$

Else we enrich the model of $q(x, r_i)$ by setting

$$\begin{aligned} f_{t+1}(x) &= \max\{f_t(x), f(x_t) + (x - x_t)^T f'(x_t)\}, \\ G_{t+1}(x) &= \max\{G_t(x), G(x_t) + (x - x_t)^T G'(x_t)\}, \\ Q_{t+1}(x, r_i) &= \max\{f_{t+1}(x) - r_i, G_{t+1}(x)\} \end{aligned}$$

and update the lower bound (see 2).

2. updating the lower bound

We should solve the following problem

$$\widehat{\delta}_{st} = \min\{Q_{t+1}(x, r_i) | x \in X_t\}$$

and set

$$\underline{\delta}_{s,t+1} = \max\{\min[l_s, \widehat{\delta}_{st}], \underline{\delta}_{st}\}.$$

If

$$\underline{\delta}_{s,t+1} \geq (1 - \theta)l_s,$$

we terminate the phase s , set

$$\bar{\delta}_{s+1} = \bar{\delta}_s, \quad \underline{\delta}_{s+1} = \underline{\delta}_{s,t+1}.$$

and pass to the phase $s + 1$.

else if

$$\underline{\delta}_{s,t+1} < (1 - \theta)l_s,$$

we update the search point, the localizer and pass to the next step $t + 1$ (see 3).

3. updating the search point and the localizer

To obtain the new search point we solve the following problem

$$x_{t+1} = \operatorname{argmin}\{\omega_s(x) | x \in X_{t+1}^+ \equiv X_t \cap \{x : Q_{t+1}(x, r_i) \leq l_s\}\}.$$

Choose as X_{t+1} any set, cut off X by finitely many linear inequalities, which satisfy the relation

$$X_{t+1}^+ \subset X_{t+1} \subset X_{t+1}^-$$

where $X_{t+1}^- = \{x \in X : (x - x_{t+1})^T \nabla \omega_s(x_{t+1}) \geq 0\}$. By this setting the step t of the phase s is completed and we pass to step $t + 1$.

4. passing to a new problem (new cycle $i+1$)

We also check whether $\underline{\delta}_{s+1} \geq \mu \bar{\delta}_s$, where $\mu \in (\frac{1}{2}, 1)$ is the parameter of the method. If so, we terminate the cycle i and compute r_{i+1} as follows:

$$r_{i+1} = \min\{f_t(x) | G_t(x) \leq 0, x \in X\}.$$

Then we pass to the new cycle, i.e. start to solve the problem

$$\min\{q(x, r_{i+1})|x \in X\}.$$

The approximate solution x^* to the constrained problem (8.7) is the best point among all searching points over all cycles and phases, i.e., which gives $\min_{i,j} q(x_j, r_i)$.

The efficiency estimate of the CONERML method is given by the following proposition.

Proposition 8.1. *Let X be compact set and $\theta > 0, \lambda < 1, \mu \in (\frac{1}{2}, 1)$ be parameters of the Nerml method. Then*

1. *the number of the cycles before we reach an accuracy ε is bounded from above by*

$$N(c) \leq \frac{\log_2 \frac{4\sqrt{2}L_f D\mu}{\varepsilon}}{\log_2 2\mu}.$$

where L_f is Lipschitz constant of the function f and D is a diameter of the set X .

2. *the total number of oracle calls to reach the accuracy ε does not exceed*

$$N(\varepsilon) = c(\theta, \lambda, \mu) \frac{\Omega L_f^2}{k\varepsilon^2} \log_2 \frac{4\sqrt{2}L_f D\mu}{\varepsilon},$$

where Ω and k are defined as previously (see section 1.4) and $c(\theta, \lambda, \mu)$ depends solely and continuously on the parameters of the method.

References

- [1] A.Ben-Tal, A.Nemirovsky. *Non-euclidean restricted memory level method for large-scale convex optimization*. Math.Program., Ser.A 102 (2005).
- [2] A.Ben-Tal, A.Nemirovsky. *Lectures on modern convex optimization*. MPS-SIAM series on optimization.
- [3] A.Nemirovsky, D.Yudin. *Problem complexity and method efficiency in optimization*. A Wiley-Interscience Publication.

- [4] C.Lemaréchal, A.Nemirovsky, Y.Nesterov. *New variants of bundle methods*. Math.Program., Ser.B 69 (1995).
- [5] E.Olvovsky. *Novel Gradient-Type Optimization Algorithms for Extremely Large-Scale Nonsmooth Convex Optimization*. Ph.D Thesis under supervision of A.Nemirovsky.