PASCAL Workshop
Thurnau March 16-18, 2005

MODERN CONVEX OPTIMIZATION

Arkadi Nemirovski
Technion - Israel Institute of Technology
nemirovs@ie.technion.ac.il

- Convex Programs
- Efficient solvability of generic convex programs
- Interior-point polynomial time algorithms
  for well-structured convex programs
- Beyond the scope of interior-point algorithms:
  simple methods for extremely large-scale convex programs

♣ **Geometrically, a convex program is**

$$\min_x \left\{c^T x : x \in X\right\} \qquad [X \subset \mathbf{R}^n : \ \textbf{convex set}]$$

♠ **In applications, optimization programs usually arise in the form of**

$$\min_{u \in \mathbf{R}^n} \left\{f_0(u) : f_i(u) \leq 0,\ i = 1, ..., m\right\}. \qquad (*)$$

When all functions $f_0(u), ..., f_m(u)$ are convex, $(*)$ can be rewritten as

$$\min_{x=(t,u) \in X} t, \qquad \underbrace{X = \{(t,u) : f_0(u) \leq t, f_1(u) \leq 0, ..., f_m(u) \leq 0\}}_{\text{convex set!}}$$

and thus $(*)$ is a convex program.

$$\min_{u\in\mathbf{R}^n}\left\{f_0(u) : f_i(u)\le 0,\ i=1,...,m\right\}. \qquad (*)$$

♠ **The simplest case of a convex program is a Linear Programming program where all f_i are linear. Such a problem can be posed in the canonical form**

$$\min_x\left\{c^Tx : Ax-b\in\mathbf{K}\equiv\mathbf{R}^n_+\right\}. \qquad \textbf{(LP)}$$

♢ **Nonlinearity in $(*)$, if any, "sits" in the objective and in the constraints. It is easily seen that a *convex* program $(*)$ can be represented in the *conic* form similar to (LP):**

$$\min_x\left\{c^Tx : Ax-b\in\mathbf{K}\right\} \qquad \textbf{(CP)}$$

where $\mathbf{K}\subset\mathbf{R}^N$ is a cone (closed, pointed, convex and with a nonempty interior), and $x\mapsto Ax:\mathbf{R}^n\to\mathbf{R}^N$ is a linear embedding.

$$\min_x \{c^T x : Ax - b \in \mathbf{K}\} \qquad \textbf{(CP)}$$

♠ **An extremely wide variety of convex programs "is captured" by** *just three* **generic cones K:**

♡ **Nonnegative orthant** $\mathbf{R}^n_+ = \{x \in \mathbf{R}^n : x \geq 0\}$ ⇒**LP**

♡ **Direct products of Lorentz cones** $\mathbf{L}^n = \{(x,t) \in \mathbf{R}^n \times \mathbf{R} : \|x\|_2 \leq t\}$ ⇒*conic quadratic programs*

$$\min_x \{c^T x : \|A_i x - b_i\|_2 \leq c_i^T x - d_i\}$$

♡ **Semidefinite cone** $\mathbf{S}^n_+ = \{X \in \mathbf{S}^n : X \succeq 0\}$ ⇒*semidefinite programs*

$$\min_x \left\{c^T x : \sum_j x_j A_j - B \succeq 0\right\}.$$

♠ **Specific nice structure of the three generic cones in question underlies modern powerful** *primal-dual interior point methods* **for Linear, Conic Quadratic and Semidefinite Programming.**

♠ **There exists a kind of calculus of "conic quadratic and semidefinite representable sets and functions" which offers a systematic way to recognize the possibility to reformulate a convex program as a conic quadratic or semidefinite program.**

minimize $c^T x$ subject to

$$
\begin{array}{ll}
(a) & \left\{ \begin{array}{l} Ax = b \\ x \geq 0 \end{array} \right. \\
(b) & \left\{ \begin{array}{l} \left(\sum_{i=1}^{8} |x_i|^3 \right)^{1/3} \leq x_2^{1/7} x_3^{2/7} x_4^{3/7} + 2 x_1^{1/5} x_5^{2/5} x_6^{1/5} \\ 5 x_2 \geq \frac{1}{x_1^{1/2} x_2^2} + \frac{2}{x_2^{1/3} x_3^3 x_4^{5/8}} \\ \begin{pmatrix} x_2 & x_1 & & \\ x_1 & x_4 & x_3 & \\ & x_3 & x_6 & x_3 \\ & & x_3 & x_8 \end{pmatrix} \preceq 5I \end{array} \right. \\
(c) & \left\{ \begin{array}{l} \begin{pmatrix} x_1 & x_2 - x_1 & x_3 - x_2 & x_4 - x_3 \\ x_2 - x_1 & x_2 & x_3 - x_2 & x_4 - x_3 \\ x_3 - x_2 & x_3 - x_2 & x_3 & x_4 - x_3 \\ x_4 - x_3 & x_4 - x_3 & x_4 - x_3 & x_4 \end{pmatrix} \succeq 0 \\ x_1 + x_2 \sin(\phi) + x_3 \sin(2\phi) + x_4 \sin(4\phi) \geq 0 \quad \forall \phi \in \left[0, \frac{\pi}{2}\right] \end{array} \right.
\end{array}
$$

♠ **The problem can be converted,** *in a systematic way*, **into SDP**

♠ **Removing constraints** (c)**, the resulting problem can be converted,** *in a systematic way*, **into CQP, and can be can be approximated,** *in a polynomial time fashion*, **by LP.**

Why convex programming?

♠ **Convex Programming admits nice and powerful** *Duality Theory* **capable to certify optimality and to quantify reliably the quality of an approximate solution.**
In contrast to this, in a typical nonconvex problems there are no easy ways to certify (global) optimality and to quantify the "level of non-optimality" of an approximate solution.

While typically Convex Duality does not allow to get a solution in a closed analytic form, it still allows for deep understanding of the problem of interest, for building its highly instructive and nontrivial reformulations, etc.

♠ **Generic convex programs, under mild computability and boundedness assumptions, are** *computationally tractable* **- they admit theoretically (and to some extent, also practically) efficient solution methods.**
In contrast to this, nonconvex optimization programs for which a global solution can be efficiently found numerically are rare exceptions (mainly due to "hidden convexity" which we are lucky to recognize).

♣ Convex Duality [conic programming case]

$$\mathrm{Opt} = \min_x \left\{ c^T x : Ax - b \in \mathbf{K} \right\} \qquad \mathbf{(P)}$$

♠ The origin of duality is in the desire to find a systematic way to bound from below the optimal value in (CP). In other words, we are interested in a mechanism for deriving from the constraints of (P) their consequences of the form

$$c^T x \geq a.$$

• The simplest way to build a linear inequality which is a consequence of the constraints of (P) is *linear aggregation*

$$Ax - b \in \mathbf{K} \Rightarrow \lambda^T (Ax - b) \geq 0 \qquad (*)$$

• **Question:** *What should be a "weight vector" λ in order to make the implication (*) valid?*

• **Answer:** *The necessary and sufficient condition is that $\lambda^T \xi \geq 0$ for every $\xi \in \mathbf{K}$:*

$$\lambda \in \mathbf{K}_* \equiv \{\lambda : \lambda^T y \geq 0 \forall y \in \mathbf{K}\}.$$

• **Fact: When $\mathbf{K}$ is a closed, convex and pointed cone with a nonempty interior, so is the** *dual cone* **$\mathbf{K}_*$, and $(\mathbf{K}_*)_* = \mathbf{K}$.**

$$\mathrm{Opt} = \min_x \left\{c^T x : Ax - b \in \mathbf{K}\right\} \qquad \textbf{(P)}$$

$$\mathbf{K}_* = \left\{\lambda : \lambda^T \xi \geq 0 \forall \xi \in \mathbf{K}\right\}$$

$$Ax - b \in \mathbf{K} \Rightarrow \lambda^T (Ax - b) \geq 0$$

• **Conclusion:** *Let $\lambda \in \mathbf{K}_*$ be such that $A^T\lambda = c$. Then* $\mathrm{Opt} \geq b^T\lambda$**.**

Equivalently: Let us associate with (P) the *dual problem*

$$\mathrm{Opt}_* = \max_\lambda \left\{b^T\lambda : A^T\lambda = c, \lambda \in \mathbf{K}_*\right\} \qquad \textbf{(D)}$$

Then Opt_* **is a lower bound on** Opt**.**

Note: $x \mapsto Ax$ **is an embedding, so that (P) can be rewritten equivalently as**

$$\textbf{(Pr):} \qquad \mathrm{Opt} = \min_\xi \left\{e^T\xi : \xi \in (\mathcal{L} - b) \cap \mathbf{K}\right\} \qquad \left[e : A^T e = c; \mathcal{L} = \mathrm{Im}A\right]$$

(D) is of the same geometric structure:

$$\textbf{(Dl):} \qquad \mathrm{Opt}_* = \max_\lambda \left\{b^T\lambda : \lambda \in (\mathcal{M} + e) \cap \mathbf{K}_*\right\} \qquad \left[\mathcal{M} = \mathrm{Ker}\, A^T = \mathcal{L}^\perp\right]$$

♡ *Geometrically, conic problem is to minimize a linear functional over the intersection of affine plane and a cone. Such a problem is called strictly feasible, if the affine plane intersects the interior of the cone.*

$$\begin{array}{lll}
\textbf{(P):} & \mathrm{Opt} = \min\limits_x \{c^T x : Ax - b \in \mathbf{K}\} & \\
 & \Updownarrow & \\
\textbf{(Pr):} & \min\limits_\xi \{e^T \xi : \xi \in (\mathcal{L} - b) \cap \mathbf{K}\} & [e : A^T e = c; \mathcal{L} = \mathrm{Im} A] \\
\textbf{(Dl):} & \max\limits_\lambda \{b^T \lambda : \lambda \in (\mathcal{M} + e) \cap \mathbf{K}_*\} & [\mathcal{M} = \mathrm{Ker}\, A^T = \mathcal{L}^\perp] \\
 & \Updownarrow & \\
\textbf{(D):} & \mathrm{Opt}_* = \max\limits_\lambda \{b^T \lambda : A^T \lambda = c, \lambda \in \mathbf{K}_*\} &
\end{array}$$

♣ **Conic Duality Theorem.** **(i)** *Conic duality is symmetric:* (D) *is a conic problem, and the problem dual to* (D) *is (equivalent to)* (P)**.**
(ii) [Weak Duality]: $\mathrm{Opt}_* \le \mathrm{Opt}$**,** *so that whenever* x *is primal, and* λ *is dual feasible, the duality gap*

$$c^T x - b^T \lambda = [Ax - b]^T \lambda = [c^T x - \mathrm{Opt}] + \underbrace{[\mathrm{Opt} - \mathrm{Opt}_*]}_{\ge 0} + [\mathrm{Opt}_* - b^T \lambda]$$

is nonnegative.
(iii) [Strong Duality] *Let one of the problems* (P), (D) *be strictly feasible and bounded. Then the other problem is solvable, and* $\mathrm{Opt} = \mathrm{Opt}_*$.

In particular, if both (P), (D) *are strictly feasible, both problems are solvable with equal optimal values.*

$$\textbf{(P):} \qquad \mathrm{Opt} = \min_x \left\{c^T x : Ax - b \in \mathbf{K}\right\}$$
$$\textbf{(D):} \qquad \mathrm{Opt}_* = \max_\lambda \left\{b^T \lambda : A^T\lambda = c, \lambda \in \mathbf{K}_*\right\}$$

♣ **Optimality conditions:** *Let both* **(P), (D)** *be strictly feasible. Then a pair* (x, λ) *of primal-dual optimal solutions is comprised of optimal solutions to the respective problems*

- *if and only if*

$$c^T x - b^T \lambda = 0 \qquad \text{[zero duality gap]}$$

and

- *if and only if*

$$[Ax - b]^T \lambda = 0 \qquad \text{[complementary slackness]}$$

♣ **Note: Nonnegative orthant, Lorentz and Semidefinite cones are self-dual. As a result, program dual to an LP/CQP/SDP program is LP/CQP/SDP, respectively.**

♣ **A** *generic convex problem* $\mathcal{P}$ **is a family of convex programs** – *instances*

$$(p): \quad \min_x \left\{ f_{(p)}(x) : x \in X_{(p)} \subset \mathbf{R}^{n(p)} \right\} \qquad [f_{(p)} : \mathbf{R}^{n(p)} \to \mathbf{R}]$$

parameterized by finite-dimensional *data vectors* $\mathrm{Data}(p)$ **and equipped with** *infeasibility measure* $I_{\mathcal{P}}(x, p)$**.**

♡ **Infeasibility measure quantifies the infeasibility of a candidate solution** $x \in \mathbf{R}^{n(p)}$ **to an instance** (p)**. As a function of** $x \in \mathbf{R}^{n(p)}$**, the infeasibility measure** $I_{\mathcal{P}}(x, p)$ **should be**

- **nonnegative everywhere and zero if and only if** x **is feasible for** (p)
- **convex in** x**.**

♡ **The dimension of the data vector of an instance** $(p) \in \mathcal{P}$ **is called the** *size* $\mathrm{Size}(p)$ **of the instance.**

- **Linear Programming is a generic problem with instances**

$$\min_x\{c^Tx : Ax - b \geq 0\},$$

the data vectors being $(\dim\, c, \dim\, b, c, A, b)$**. As an infeasibility measure, one can take**

$$I_{\mathcal{LP}}(x,p) = \min\left\{t \geq 0 : Ax - b + t\mathbf{1} \geq 0\right\},\ \mathbf{1} = (1,...,1)^T;$$

- **Conic Quadratic Programming is a generic problem with instances**

$$\min_x\left\{c^Tx : \|A_ix - b_i\|_2 \leq c_i^Tx - d_i, i = 1,...,m\right\},$$

the data vectors being $(\dim\, x, m, \dim\, b_1, ..., \dim\, b_m, c, \{A_i, b_i, c_i, d_i\}_{i=1}^m)$**. As an infeasibility measure, one can take**

$$I_{\mathcal{CQP}}(x,p) = \min\left\{t \geq 0 : \|A_ix - b_i\|_2 \leq c_i^Tx - d_i + t,\ i = 1,...,m\right\};$$

- **Semidefinite Programming is a generic problem with instances**

$$\min_x\left\{c^Tx : \sum_j x_jA_j - B \succeq 0\right\},$$

data vectors being $(\dim\, x, \dim\, B, c, \{A_j\}_{j=1}^{\dim\, x}, B)$**. As an infeasibility measure, one can take**

$$I_{\mathcal{SDP}}(x,p) = \min\left\{t \geq 0 : \sum_j x_jA_j - B + tI \succeq 0\right\};$$

- **Geometric Programming is a generic problem with instances**

$$\min_x \{f_0(x) : f_i(x) \leq 0,\ i = 1, ..., m\},$$
$$f_\ell(x) = \ln\left(\sum_{j=1}^{k} \exp\{a_{ij} + \alpha_{ij}^T x\}\right)$$

data vectors being $(\dim x, m, k, \{a_{ij}, \alpha_{ij}\}_{i,j})$**. As an infeasibility measure, one can take**

$$I_{\mathcal{GP}}(x, p) = \min\{t \geq 0 : f_i(x) \leq t,\ i = 1, ..., m\}.$$

♣ *A solution algorithm* $\mathcal{A}$ **for a generic problem** $\mathcal{P}$ **is a code for a Real Arithmetic Computer which, given on input**

- **the data vector** $\mathrm{Data}(p)$ **of an instance**

$$(p): \quad \min_x \left\{ f_{(p)}(x) : x \in X_{(p)} \subset \mathbf{R}^{n(p)} \right\}$$

of $\mathcal{P}$**, and**

- **a required accuracy** $\epsilon > 0$**,**

returns on output

- **either an** ϵ*-solution* **to** (p)**, that is, a vector** x_ϵ **which is** ϵ**-optimal and** ϵ**-feasible for** (p)**:**

$$f_{(p)}(x_\epsilon) - \mathrm{Opt}(p) \leq \epsilon \ \& \ I_{\mathcal{P}}(x_\epsilon, p) \leq \epsilon,$$

- **or a correct conclusion "**(p) **is infeasible",**
- **or a correct conclusion "**$\mathrm{Opt}(p) = -\infty$**".**

♣ *The* ϵ*-complexity* **of** $(p) \in \mathcal{P}$ **w.r.t. a solution algorithm** $\mathcal{A}$ **is the number** $\mathcal{T}_{\mathcal{A}}(\epsilon, p)$ **of Real Arithmetic operations required to process the input** $(\mathrm{Data}(p), \epsilon)$

♣ **A solution algorithm $\mathcal{A}$ for a generic problem $\mathcal{P}$ is called** *polynomial time* **("theoretically efficient"), if**

$$\mathcal{T}_{\mathcal{A}}(\epsilon, p) \leq \mathrm{Poly}\left(\mathrm{Size}(p), \mathrm{Digits}_{(p)}(\epsilon)\right),$$

where

$$\mathrm{Digits}_{(p)}(\epsilon) = \ln\left(\frac{\mathrm{Size}(p) + \|\mathrm{Data}(p)\|_\infty + \epsilon^2}{\epsilon}\right)$$

is the *number of accuracy digits* **in an ϵ-solution to (p).**
Interpretation: When a solution algorithm is polynomial time, 10-fold increase in computer power allows

- **to increase by an absolute constant factor the** *sizes* $\mathrm{Size}(p)$ **of instances which can be solved within a given accuracy in a given time, or**
- **to increase by an absolute constant factor the number of accuracy digits which can be obtained in a given time on instances of a given size.**

♣ **A generic problem $\mathcal{P}$ is called** *polynomially solvable* **("computationally tractable"), if it admits a polynomial time solution algorithm.**

• **Claim:** *Under mild computability and boundedness assumptions, a generic convex problem is polynomially solvable.*

In particular, adding to instances of LP/CQP/SDP/GP *box constraints*

$$\|x\|_\infty \leq R$$

and treating R as part of the data vectors, we make the corresponding generic problems polynomially solvable.

♣ **The basic fact underlying polynomial time solvability of generic convex problems is as follows:**

Theorem. *Consider an optimization program*

$$\min_x \{f(x) : x \in B_n \equiv \{x \in \mathbf{R}^n : \|x\|_2 \le 1\}\},$$

with convex and continuous on B_n objective f. There exists an explicit algorithm which, for every $\epsilon > 0$, is capable to find a feasible ϵ-solution to the problem at the cost of computing the values $f(x_i)$ and subgradients $f'(x_i)$ of the objective at

$$N(\epsilon) = O(1)n^2 \ln\left(n + \frac{n\,\mathrm{V}(f)}{\epsilon}\right),$$
$$\mathrm{V}(f) = \max_{B_n} f - \min_{B_n} f$$

recursively generated points $x_i \in \mathrm{int}\ B_n$, with $O(1)n^2$ additional arithmetic operations per every one of these computations.

♠ **"Universal" polynomial time algorithms like the one mentioned in the Theorem use *local* information (values and subgradients) on the objective and the constraints and therefore cannot utilize a priori knowledge of problem's structure. For "well-structured" convex problems, like LP/CQP/SDP, there exist *Interior Point Polynomial Time* algorithms which do utilize problem's structure and, as a result, yield better complexity bounds and exhibit much better practical performance.**

♣ **Consider a** *canonical cone*

$$\begin{aligned}\mathbf{K} &= \mathbf{L}^{n_1} \times ... \times \mathbf{L}^{n_p} \times \mathbf{S}_+^{n_{p+1}} \times ... \times \mathbf{S}_+^{n_{p+q}} \\ &\subset E = \mathbf{R}^{n_1+1} \times ... \times \mathbf{R}^{n_p+1} \times \mathbf{S}^{n_{p+1}} \times ... \times \mathbf{S}^{n_{p+q}}\end{aligned}$$

- E **is equipped with the inner product**

$$\langle \xi, \lambda \rangle = \xi_1^T \lambda_1 + ... \xi_p^T \lambda_p + \mathrm{Tr}(\xi_{p+1}\lambda_{p+1}) + ... + \mathrm{Tr}(\xi_{p+q}\lambda_{p+q})$$

- **K is equipped with** *canonical barrier*

$$K(\xi) = K_1(\xi_1) + ... + K_{p+q}(\xi_{p+q}) : \mathrm{int}\ \mathbf{K} \to \mathbf{R},$$

where

- **for a Lorentz factor** $\mathbf{L}^{n_i} = \{(x,t) \in \mathbf{R}^{n_i} \times \mathbf{R} : \|x\|_2 \le t\}$**, we set**

$$K_i(x,t) = -\ln\left(t^2 - x^T x\right),$$

- **for a semidefinite factor** $\mathbf{S}_+^{n_i} = \{x \in \mathbf{S}^{n_i} : x \succeq 0\}$ **we set**

$$K_i(x) = -\ln \mathrm{Det}\,(x).$$

- **We define the** *parameter* $\vartheta(K)$ **of barrier** K **as twice the number of Lorentz faactors plus the total row size of the semidefinite factors in K.**

$$\mathbf{K} = \mathbf{L}^{n_1} \times ... \times \mathbf{L}^{n_p} \times \mathbf{S}_+^{n_{p+1}} \times ... \times \mathbf{S}_+^{n_{p+q}}$$
$$K(\xi) = K_1(\xi_1) + ... + K_{p+q}(\xi_{p+q}) : \text{int } \mathbf{K} \to \mathbf{R}$$

Important facts:

- **$\mathbf{K}$ is self-dual;**
- **The anti-gradient mapping $\xi \to -\nabla K(\xi)$ is a one-to-one mapping of** int $\mathbf{K}$ **onto itself which is** *self-inverse*:

$$-\nabla K(-\nabla K(\xi)) = \xi \ \ \forall \xi \in \text{int } \mathbf{K}.$$

♣ **Given a primal-dual pair of strictly feasible conic problems**

$$\textbf{(Pr):}\quad \min_{\xi}\{\langle e,\xi\rangle : \xi\in[\mathcal{L}-b]\cap\mathbf{K}\} \qquad \textbf{(Dl):}\quad \max_{\lambda}\left\{\langle b,\lambda\rangle : \lambda\in[\mathcal{L}^\perp+e]\cap\mathbf{K}\right\}$$

we associate with the pair

- *primal central path* $\xi_*(t)=\operatorname{argmin}_{\xi}\{t\langle e,\xi\rangle+K(\xi):\xi\in[\mathcal{L}-b]\cap\mathbf{K}\}\quad[t>0]$
- *dual central path* $\lambda_*(t)=\operatorname{argmin}_{\lambda}\left\{-t\langle b,\lambda\rangle+K(\xi):\lambda\in[\mathcal{L}^\perp+e]\cap\mathbf{K}\right\}\quad[t>0]$

♠ **Important facts:**

- **both paths are well-defined on** $(0,\infty)$ **and belong to the sets of strictly feasible solutions to the respective problems**
- **both paths are images of each other under scaled anti-gradient mapping:**

$$\lambda_*(t)=-t^{-1}\nabla K(\xi_*(t));\qquad \xi_*(t)=-t^{-1}\nabla K(\lambda_*(t))$$

- **Along the** *primal-dual central path* $(\xi_*(t),\lambda_*(t))$**, the duality gap is** $\vartheta(K)/t$**:**

$$\mathrm{DualityGap}(\xi,\lambda)\equiv\langle\xi,\lambda\rangle\equiv[\langle e,\xi\rangle-\mathrm{Opt}(\mathbf{Pr})]+[\mathrm{Opt}(\mathbf{Dl})-\langle b,\lambda\rangle]\big|_{\substack{\xi=\xi_*(t)\\ \lambda=\lambda_*(t)}}=\frac{\vartheta(K)}{t}$$

♠ **We have seen that as $t \to \infty$, the duality gap along the primal-dual central path goes to 0, so that the path approaches the set of primal-dual optimal solutions.** *In primal-dual path-following methods, one traces the path as $t \to \infty$, staying in an appropriately defined neighbourhood of the path and thus approaching primal-dual optimality.*

♠ **"Feasible start" generic primal-dual path-following algorithm:**

• **A pair $(\xi = \xi_*(t), \lambda = \lambda_*(t))$ on the central path is fully characterized by the following conditions:**

$$\begin{array}{ll} \xi \in [\mathcal{L} - b] \cap \mathrm{int}\ \mathbf{K} & \text{[strict primal feasibility]} \\ \lambda \in [\mathcal{L}^\perp + e] \cap \mathrm{int}\ \mathbf{K} & \text{[strict dual feasibility]} \\ \underbrace{t\lambda + \nabla K(\xi)}_{B(\xi,\lambda,t)} = 0 & \left[\begin{array}{c}\text{augmented complementary slackness} \\ \text{in LP: } \lambda_i\xi_i = t^{-1}\ \forall i\end{array}\right] \end{array} \qquad (*)$$

• **In order to trace the path, one iterates the following scheme:**

Given a current triple (ξ, λ, t) with strictly primal-dual feasible (ξ, λ), we

- **choose a target value $t_+ \geq t$ of the penalty;**
- **find a *primal-dual search direction* $(\Delta\xi, \Delta\lambda)$ by linearizing (*):**

$$\Delta\xi \in \mathcal{L};\ \Delta\lambda \in \mathcal{L}^\perp;\ B(\xi,\lambda,t_+) + \frac{\partial B(\xi,\lambda,t_+)}{\partial\xi}\Delta\xi + \frac{\partial B(\xi,\lambda,t_+)}{\partial\lambda}\Delta\lambda = 0$$

- **Update the triple**

$$(\xi, \lambda, t) \leftarrow (\xi_+ = \xi + \Delta\xi, \lambda_+ = \lambda + \Delta\lambda, t_+)$$

Various primal-dual path-following IP's follow the outlined scheme, utilizing two additional options:

- **incorporating line search at the updating step**
- **exploiting various equivalent forms of the augmented complementary slackness condition.**

> **Augmented Complementary Slackness** $t\lambda + \nabla K(\xi) = 0$ **can be rewritten in many equivalent forms (e.g.,** $t\xi + \nabla K(\lambda) = 0$**).** *Outside of the central path, linearizations of various forms of A.C.S. are* <u>*not*</u> *equivalent to each other*, **so that different forms of A.C.S. lead to different path-following methods.**

♠ **Example: "Primal" path-following method. With A.C.S. written as** $t\lambda + \nabla K(\xi) = 0$ **and the primal-dual pair coming from the program**

$$\min_x \left\{ c^T x : Ax - b \in \mathbf{K} \right\},$$

the outlined scheme results in the essentially purely primal recurrence

$$\begin{array}{c} t \mapsto t_+ > t,\ x \mapsto x_+ = x - \left[A^T \nabla^2 K(\underbrace{Ax - b}_{\xi}) A\right]^{-1} \left[t_+ c + A^T \nabla K(\xi)\right] \\ \xi \mapsto \xi_+ = Ax_+ - b,\ \lambda \mapsto \lambda_+ = -t_+^{-1} \left[\nabla K(\xi) + \nabla^2 K(\xi)[\xi_+ - \xi]\right] \end{array} \qquad (*)$$

♡ **Theorem [short-step primal path-following method] Let (*) be initialized at a starting point** $x = x_0$, $t = t_0 > 0$ **"close to the path":**

$$[tc + A^T \nabla K(\xi)]^T [A^T \nabla^2 K(\xi) A]^{-1} [tc + A^T \nabla K(\xi)] \leq 0.1^2 \qquad [\xi = Ax - b]$$

Then, with the penalty updating policy $t_+ = t[1 + 0.1\vartheta^{-1/2}(K)]$**, the algorithm is well-defined, keeps all iterates close to the path and, for every** $\epsilon > 0$**, results in primal-dual strictly feasible solution with duality gap** $\leq \epsilon$ **in no more than**

$$N(\epsilon) = O(1)\sqrt{\vartheta(K)} \ln\left(2 + \frac{\vartheta(K)}{t_0 \epsilon}\right)$$

steps.

♡ **Theorem assumes that the algorithm is started "close to the path". This can be ensured,** *keeping the complexity bound essentially intact*, **by a suitable initialization scheme (based on the same path-following techniques as applied to an appropriate auxiliary problem).**

♡ **The theoretical complexity bounds stated by Theorem are the best known for LP/CQP/SDP. At the same time, the associated worst-case-oriented "short step" penalty updating policy** $t \mapsto t\left(1 + O(1)\vartheta^{-1/2}(K)\right)$ **is too slow from the practical viewpoint.**

"Practical" IPm's do not require "close to the path" (or even feasible) starting point, use more "aggressive" stepsize policies, operate in much wider neighbourhood of the central path and are "symmetric" with respect to primal and dual variables. As a result, practical IPM's exhibit much better behaviour in actual computations than the short-step primal path-following method (although at the price of "spoiling" the theoretical complexity bound to $O(1)\vartheta(K)\ln\left(\frac{\vartheta(K)}{\epsilon}\right)$**).**

♣ After two decades of Interior Point Revolution, the entire Convex Programming is "within the reach" of Interior Point Polynomial Time methods

⇒ theoretical (and to some extent – practical) possibility to solve convex programs to *high* accuracy with *low* iteration count

However: When solving programs with n variables, an IPM iteration requires assembling and solving $n \times n$ Newton system of linear equations. With standard Linear Algebra, this costs $O(n^3)$ operations, unless the matrix of the system is highly sparse with favourable sparsity pattern. As a matter of fact,

- typical LPs of real-world origin do lead to sparse Newton systems

  ⇒ IPMs are capable to solve LPs with $10^4 - 10^5$ and even 10^6 variables

- typical nonlinear convex programs (especially SDPs) lead to dense Newton systems

  ⇒ in reality, IPMs can fail to solve a nonlinear convex program with "just" 10^4 variables – the very first iteration will last forever...

(!) With design dimension $n \sim 10^4$, iteration cost $O(n^3)$ becomes prohibitively large, and with $n \sim 10^5$ and more, a linear in n iteration cost becomes a must...

♣ In the "extremely large-scale case" (n of order of tens and hundreds of thousands), (!) rules out all advanced convex optimization techniques, including all known polynomial time algorithms. As far as nonsmooth and/or constrained convex problems are concerned, at the present level of our knowledge (!) leaves us with the only option: *first order gradient-type methods.*

♣ **A convenient framework for presenting gradient methods is** *Convex Programming in saddle point form*, **where the problem of interest is**

$$\min_{x\in X}\left\{f(x)=\max_{y\in Y}\phi(x,y)\right\} \qquad (S)$$

- X **and** Y **– compact convex sets in Euclidean spaces**
- $\phi(x,y): X\times Y\to \mathbf{R}$ **– Lipschitz continuous function convex in** $x\in X$ **and concave in** $y\in Y$**.**

Note: A convex program $\min_{x\in X} f(x)$ **can be in many ways reduced to** (S)**:**

♢ **there always is a trivial possibility** $\phi(x,y)\equiv f(x)$

♢ **when** f **is "well-structured", other options are possible, e.g.:**

- $[x\in\mathbf{R}^n]\ \ f(x)\equiv \max\limits_{1\le j\le m}[a_i^Tx+b_i]=\max\limits_y\left\{\sum\limits_i y_i[a_i^Tx+b_i]: y\ge 0,\sum\limits_i y_i=1\right\}$
- $[x\in\mathbf{S}^m]\ \ f(x)\equiv \lambda_{\max}(x)=\max\limits_y\{\mathrm{Tr}(yx): y\succeq 0,\mathrm{Tr}(y)=1\}$

  $[\lambda_{\max}(x)\equiv\lambda_1(x)\ge\lambda_2(x)\ge ...\ge\lambda_m(x)$ are eigenvalues of $x\in\mathbf{S}^m]$
- $[x\in\mathbf{S}^m]\ \ f(x)\equiv \lambda_1(x)+...+\lambda_k(x)=\max\limits_y\{\mathrm{Tr}(xy): 0\preceq y\preceq I,\mathrm{Tr}(y)=k\}$
- $[x\in\mathbf{S}^m]\ \ f(x)\equiv \|x\|=\max\limits_{u,v}\{\mathrm{Tr}(x[u-v]): u,v\succeq 0,\mathrm{Tr}(u)=\mathrm{Tr}(v)=1\}$

♠ **Here,** f **is highly nonsmooth and nonlinear, while** ϕ **is just bilinear!**

$$\min_{x\in X}\left\{f(x)=\max_{y\in Y}\phi(x,y)\right\} \qquad (S)$$

♣ **In contrast to IPMs, all known first order methods for solving (S) are unable to utilize detailed a propri knowledge of problem's structure and use *black box representation* of (S). In this representation,**

- **the sets X, Y are known in advance**
- **ϕ is known to belong to a given family $\mathcal{F}$ of convex-concave Lipschitz continuous functions on $Z=X\times Y$**
- ***quantitative* information is obtained during the solution process via subsequent calls to the *First Order Oracle* which, given on input $(x,y)\in X\times Y$, returns $\phi(x,y)$ along with a subgradient $\phi'_x(x,y)$ of ϕ in x and a supergradient $\phi'_y(x,y)$ of ϕ in y.**

$$f_\phi^* = \min_{x\in X}\left\{f_\phi(x) = \max_{y\in Y}\phi(x,y)\right\}, \qquad \phi\in\mathcal{F} \tag{S}$$

♣ **Limits of performance of** *black-box-oriented* **methods for** (S) **are given by Information-Based Complexity Theory.**

♠ **In IBCT, one considers a class** $(X, Y, \mathcal{F})$ **of problems of the form of** (S) **associated with** X, Y **and a given family** $\mathcal{F}$ **of convex-concave functions** ϕ**. A solution method** $\mathcal{B}$ **is a collection of rules for generating**

- **search points** $z_t = (x_t, y_t) \in X = X\times Y$ **where** ϕ, ϕ' **are computed;**
- **approximate solutions** $x^t \in X$**.**

The only restriction on rules is casuality: z_t **and** x^t **should depend solely on the "past information"** $\{\phi(z_\tau), \phi'(z_\tau)\}_{\tau<t}$**.**

♠ **The** ϵ**-complexity of** $(X, Y, \mathcal{F})$ **w.r.t.** $\mathcal{B}$ **is the minimal number of steps** N **in which** $\mathcal{B}$ **solves** *all* **problems from the class within accuracy** ϵ**:**

$$\mathrm{Compl}^{\mathcal{B}}(\epsilon) = \min\left\{N : f_\phi(x^t) - f_\phi^* \le \epsilon\ \forall(\phi\in\mathcal{F}, t\ge N)\right\}.$$

♠ **The** ϵ**-complexity of** $(X, Y, \mathcal{F})$ **is the best of complexities w.r.t.** $\mathcal{B}$**'s:**

$$\mathrm{Compl}(\epsilon) = \min_{\mathcal{B}} \mathrm{Compl}^{\mathcal{B}}(\epsilon).$$

$$f_\phi^* = \min_{x\in X}\{f_\phi(x) = \max_{y\in Y}\phi(x,y)\}; \qquad \phi \mapsto \Phi(x,y) = \left[\begin{array}{c} \phi'_x(x,y) \\ -\phi'_y(x,y) \end{array}\right] \qquad (S)$$

♣ **Information-Based Complexity** $\mathrm{Compl}(\cdot)$ **yields "ultimate" limits on worst-case performance of black-box-oriented methods as applied to a given class of problems** (S)**. These limits are known for all "non-parametric" classes of convex problems, including large-scale ones. Here are "large-scale" results related to the case of "Euclidean geometry":**

Let X, Y **be subsets of the unit Euclidean ball** B_n **in** $\mathbf{R}^n$**. Then**

A. Nonsmooth case: If $\mathcal{F}$ **is the set of all convex-concave functions** ϕ **such that** $\|\Phi(z') - \Phi(z'')\|_2 \le L$ **for all** $z', z'' \in X \times Y$**, then**

$$10\,(L/\epsilon)^2 \ge \ \mathrm{Compl}(\epsilon) \quad \underbrace{\ge \frac{1}{10}\,(L/\epsilon)^2}_{\begin{array}{c}\text{provided } X = Y = B_n \\ \text{and } n \ge (L/\epsilon)^2\end{array}}$$

The lower bound remains valid even in the *minimization* **case, i.e., when** ϕ**'s are further restricted to be independent of** y **and to be as simple as** $\phi(x,y) = \max\limits_{1\le i\le n}[\epsilon_i x_i + a_i]$**,** $\epsilon_i = \pm 1$**.**

B. Smooth saddle point case: If $\mathcal{F}$ is the set of all convex-concave functions ϕ such that $\|\Phi(z') - \Phi(z'')\|_2 \leq L\|z' - z''\|_2$ for all $z', z'' \in X \times Y$, then

$$10\,(L/\epsilon) \geq \ \mathrm{Compl}(\epsilon) \quad \underbrace{\geq \frac{1}{10}\,(L/\epsilon)}_{\substack{\text{provided } X = Y = B_n \\ \text{and } n \geq L/\epsilon}}$$

The lover bound remains valid when ϕ's are further restricted to be bilinear: $\phi(x, y) = a^T x + x^T A y + b^T y$.

C. Smooth minimization case: If, in addition to smoothness from B, ϕ's are restricted to be independent of y, then

$$10\,(L/\epsilon)^{1/2} \geq \ \mathrm{Compl}(\epsilon) \quad \underbrace{\geq \frac{1}{10}\,(L/\epsilon)^{1/2}}_{\substack{\text{provided } X = B_n \text{ and} \\ n \geq (L/\epsilon)^{1/2}}}$$

The lower bound remains valid when ϕ's are further restricted to be convex quadratic forms of x: $\phi(x, y) = x^T A x - 2b^T x$, $A \succeq 0$.

<u>Note:</u> The outlined complexity bounds are dimension-independent!

♣ **The outlined IBCT results and their extensions/modifications provide us with**

Bad news: In large-scale case, gradient type algorithms, same as all black-box-oriented methods, possess slow – sublinear – rate of convergence.

⇒ With gradient type methods, one cannot hope to get high accuracy solutions, provided that n is large.

However,

In numerous applications, all we need are medium-accuracy solutions
⇒ What is of primary importance in large-scale optimization, is whether the rate of convergence does or does not deteriorate as the dimension grows, and not whether this rate is high or slow.

Good news: In the case of problems with "favourable geometry", the complexity is dimension-independent (or nearly so)

More good news: The (nearly) dimension-independent complexity is yielded by computationally cheap gradient-type methods.

⇒ When solving large-scale problems, gradient methods are natural candidates...

♣ **Some History:**

♠ **The very first gradient-type method for solving $\min_X f(x)$ with non-smooth convex f – Subgradient Descent**

$$x_{t+1} = \operatorname*{argmin}_{x \in X} \|[x_t - \gamma_t f'(x_t)] - x\|_2^2$$

originates from N. Shor ('63) and B. Polyak ('65) and over years was intensively modified, primarily via utilizing past information (*bundle methods*).

♠ **SD is intrinsically adjusted to problems with Euclidean geometry. A substantial generalization of SD, the Mirror Descent method [Nem.&Yudin'77, Ben-Tal,Margalit,Nem.'01, Teboulle&Beck'03] allows to adjust the scheme to nonsmooth minimization and saddle point problems on**

- **simplexes**
$$\Delta_n = \{x \in \mathbf{R}^n : x \geq 0, \sum_i x_i \leq 1\}$$
- **spectahedrons**
$$\Sigma_n = \{x \in \mathbf{S}^n : x \succeq 0, \mathrm{Tr}(x) \leq 1\},$$

etc.

♠ **Until recently, common wisdom said that**

• **The only way to exploit problem's structure in nonlinear convex minimization is offered by IPMs (and thus is too computationally demanding in the extremely large-scale case);**

• **Computationally cheap gradient type methods are black-box oriented and thus must obey the IBCT limits of performance. Consequently, when solving a large-scale convex program**

$$\min_{x \in X} f(x), \qquad (*)$$

accuracy after t steps cannot be better than

♢ $O(t^{-1/2})$ **for a nonsmooth Lipschitz continuous** f **(really slow...)**

♢ $O(t^{-2})$ **for** $\mathrm{C}^{1,1}$**-smooth** f **(alas, smoothness is a rare commodity...)**

♠ **A breakthrough in understanding the situation is due to Yu. Nesterov ('03) who observed that if f is of nice analytic structure, it usually can be represented as $f(x) = \max_{y \in Y} \phi(x, y)$ *with* $\mathrm{C}^{1,1}$*-smooth* ϕ, and therefore the resulting saddle point problem can be solved by simple methods at the rate $O(t^{-1})$.**

Note: Accelerating convergence from the "common wisdom" $O(t^{-1/2})$ to $O(t^{-1})$ indeed makes a difference!

$$\min_{x\in X}\left\{f(x)=\max_{y\in Y}\phi(x,y)\right\} \qquad (S)$$

♣ **Nesterov's $O(t^{-1})$-method combines a large-scale optimal algorithm for minimizing *smooth* convex functions with smooth approximation of f, based on saddle point representation and adjusted from step to step.**

We are about to present an alternative $O(t^{-1})$-method – Mirror Prox – which works directly with (S).

♣ **Setup for MP is given by $Z=X\times Y$, a norm $\|\cdot\|$ on the space E where Z lives and a $\mathbf{C}^1$ strongly convex *distance-generating function* $\omega(\cdot):Z\to\mathbf{R}$. We associate with the setup data the parameters**

$$\Theta = \max_{u,v\in Z}\underbrace{[\omega(u)-\omega(v)-\langle\omega'(v),u-v\rangle]}_{\text{local distance }\omega_v(u)}$$

$$\alpha = \max\left\{c:\omega_v(u)\geq\tfrac{c}{2}\|u-v\|^2\ \forall u,v\in Z\right\}$$

and the prox mappings

$$P_v(\xi)=\operatorname*{argmin}_{z\in Z}\left\{\langle\xi,z\rangle+\omega_v(z)\right\}.$$

♠ **"Implementability Assumption": Z and $\omega(\cdot)$ are simple and fit each other, so that prox mappings are easy to compute.**

$$\left.\begin{array}{c}\min\limits_{x\in X}\left\{f(x)=\max\limits_{y\in Y}\phi(x,y)\right\}\\ \Updownarrow\\ \max\limits_{y\in Y}\left\{g(y)=\min\limits_{x\in X}\phi(x,y)\right\}\end{array}\right\}\mapsto\Phi(x,y)=\left[\begin{array}{c}\phi'_x(x,y)\\ -\phi'_y(x,y)\end{array}\right]\qquad(S)$$

♣ **The basic Mirror Prox algorithm is**

$$z_{t-1}\mapsto w_t:=P_{z_{t-1}}(\gamma_t\Phi(z_{t-1}))\mapsto z_t:=P_{z_{t-1}}(\gamma_t\Phi(w_t))$$

$$s^t\equiv(x^t,y^t)=\left(\sum_{\tau=\lfloor t/2\rfloor}^{t}\gamma_\tau\right)\sum_{\tau=\lfloor t/2\rfloor}^{t}\gamma_\tau w_\tau$$

where $\gamma_t>0$ are stepsizes.

<u>Theorem:</u> Let $\Phi(\cdot)$ be Hölder continuous with exponent $\sigma\in[0,1]$ and constant L:

$$\|\Phi(z)-\Phi(z')\|_*\le L\|z-z'\|^\sigma\ \forall z,z'\in Z.$$

Then the stepsize policy $\gamma_t=0.7L^{-1}\left(\frac{\Theta}{t}\right)^{\frac{1-\sigma}{2}}\alpha^{\frac{1+\sigma}{2}}$ ensures that

$$\epsilon(s^t)\equiv\left[f(x^t)-\min_X f\right]+\left[\max_Y g-g(y^t)\right]\le O(1)L\left(\frac{\Theta}{\alpha t}\right)^{\frac{1+\sigma}{2}}.$$

- **Setup:** $(\omega(\cdot), Z = X \times Y, \|\cdot\|) \Rightarrow (\Theta, \alpha)$
- **Problem:** $\min\limits_{x\in X}\max\limits_{y\in Y}\phi(x,y)$
- **Assumption:** $\|\Phi(z) - \Phi(z')\|_* \le L\|z - z'\|^\sigma,\ \Phi = \begin{bmatrix} \phi'_x \\ -\phi'_y \end{bmatrix}$

$$\Downarrow$$

$$\epsilon(s^t) \le O(1)L\left(\frac{\Theta}{\alpha t}\right)^{\frac{1+\sigma}{2}}$$

♠ **Nonsmooth case $\sigma = 0$: ϕ is Lipschitz continuous with constant L w.r.t. $\|\cdot\| \Rightarrow O(t^{-1/2})$-rate of convergence**

♠ **Smooth case $\sigma = 1$: gradient of ϕ is Lipschitz continuous with constant L w.r.t. $\|\cdot\|$: $\Rightarrow O(t^{-1})$-rate of convergence**

♣ **Adjusting $(\omega(\cdot), \|\cdot\|)$, one can optimize the efficiency estimate, thus adjusting the method to the geometry of problem in question.**

♣ **Good setups for MP are known when X and Y are direct products of (simple subsets) of**

- **Euclidean balls** $\{x \in \mathbf{R}^n : \|x\|_2 \leq R\}$,
- **boxes** $\{x \in \mathbf{R}^n : \|x\|_\infty \leq R\}$,
- **"matrix boxes"** $\{x \in \mathbf{S}^n : -RI \preceq x \preceq RI\}$,
- **simplexes** $\{x \in \mathbf{R}^n : 0 \leq x, \sum_i x_i \leq R\}$,
- **spectahedrons** $\{x \in \mathbf{S}^n : 0 \preceq x, \mathrm{Tr}(x) \leq R\}$.

♠ **The resulting efficiency estimate for MP is nearly or fully dimension-independent, provided there are no box-type factors in X, Y**

♣ **Example: SVM**

♠ **Problem:** *Given n points $z^i \in \mathbf{R}^d$ partitioned into two sets B ("blue") and R ("red"), find affine form*

$$\alpha^T z + \beta = 0$$

which separates best of all the blue and the red points.

♠ **Model:**

$$\min_{\alpha,\beta} \left\{ \sum_{i=1}^{n} \left[1 - y_i(\alpha^T z^i + \beta)\right]_+ : \|\alpha\|_1 \le \rho \right\} \qquad (*)$$

- **We have**

$$\begin{aligned} f(\alpha,\beta) &\equiv \sum_i \left[1 - y_i(\alpha^T z^i + \beta)\right]_+ = \sum_i \max_{0\le\lambda_i\le 1} \lambda_i[1 - y_i(\alpha^T z + \beta)] \\ &= \max_{0\le\lambda_i\le 1} \left\{\lambda^T[\mathbf{1} + A\alpha] - \beta\lambda^T y\right\} \quad [A_{ij} = -y_i z^i_j] \end{aligned}$$

$$\Downarrow$$

$$F(\alpha) \equiv \min_\beta f(\alpha,\beta) = \max_{\lambda\in\Lambda} \lambda^T[\mathbf{1} + A\alpha], \ \Lambda = \{\lambda : 0 \le \lambda_i \le 1, \lambda^T y = 0\}$$

- **Thus, (*) reduces to the saddle point problem**

$$\min_{\|\alpha\|_1\le\rho} F(\alpha), \quad F(\alpha) = \max_{\lambda\in\Lambda} \lambda^T[\mathbf{1} + A\alpha].$$

♠ **With MP, an ϵ-solution "costs" $O(1)\frac{\rho\sqrt{n\ln d}\max_j\|A_j\|_2}{\epsilon}$ multiplications of given vectors by A, A^T.**

$$\min_{\alpha,\beta}\left\{\sum_{i=1}^{n}\left[1-y_i(\alpha^T z^i+\beta)\right]_+ : \|\alpha\|_1\le\rho\right\} \qquad (*)$$

$$\Updownarrow$$

$$\min_{\|\alpha\|_1\le\rho} F(\alpha),\ \ F(\alpha)=\max_{\lambda\in\Lambda}\lambda^T\left[\mathbf{1}+A\alpha\right] \qquad (**)$$

$$A=\left[-y_1z^1,...,-y_nz^n\right]^T : n\times d,\ \ \Lambda=\{\lambda\in\mathbf{R}^n : 0\le\lambda\le\mathbf{1}, y^T\lambda=0\}$$

♣ **Numerical illustration:**

♠ **Training set:** $n=15,000$ **randomly generated sparse vectors (7,506 blue and 7,494 red) in** $d=5,000$**-dimensional space**
$\Rightarrow$ **3,142,764 nonzeros in** 15000×5000 **matrix** A **(density 0.04)**
♠ **Generation ensures that the blue and the red vectors admit sparse "near-separator"** $\alpha_*^Tz+\beta_*$ **which misclassifies** $\approx 6\%$ **of vectors.**
♠ **(*) can be posed as an LP program with 25,001 variables and 25,001 inequality constraints and can be solved by IP methods. (**) can be solved by Mirror Prox.**

♠ **With $\rho = 3.0$, the results are as follows:**

CPU time	# of matrix-vector multiplications	relative error in objective	classification error	
			training set	testing set
Mirror Prox solutions:				
$62'$	114	3.4e-3	9.5%	8.9%
$125'$	230	1.3e-3	9.5%	10.1%
$188'$	346	5.0e-4	9.5%	9.7%
$251'$	462	2.7e-4	9.5%	9.7%
$314'$	578	1.8e-4	9.5%	9.4%
$506'$	916	1.8e-4	9.5%	10.0%
$568'$	1028	5.4e-5	9.5%	9.8%
Interior Point solution:				
$1782'$	—	2.e-12	9.5%	10.0%

Note: "True" separator has 71 nonzeros, MP and IP separators have 60 nonzeros.