DAGM 2011 Tutorial on **Convex Optimization for Computer Vision**

Part 1: Convexity and Convex Optimization

Daniel Cremers
Computer Vision Group
Technical University of Munich

Thomas Pock
Institute for Computer Graphics and Vision
Graz University of Technology

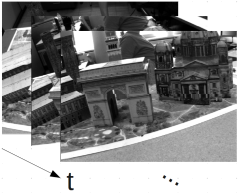
Frankfurt, August 30, 2011

Overview

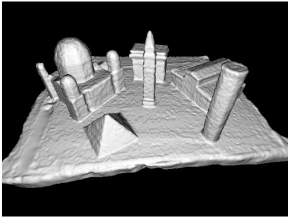
Overview

Computer vision deals with inverse problems

- Projection of the 3D world onto the 2D image plane

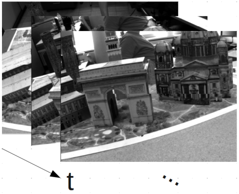


- Determine unknown model parameters based on observed data

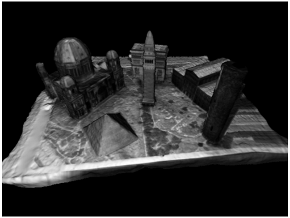


Computer vision deals with inverse problems

- Projection of the 3D world onto the 2D image plane



- Determine unknown model parameters based on observed data



Computer vision is highly ambiguous

What you see ...

Computer vision is highly ambiguous

What you see ... is maybe not what it is!

[Fukuda's Underground Piano Illusion]

Energy minimization methods

- It is in general not possible to solve inverse problems directly
- Add some smoothness assumption to the unknown solution
- Leads to the energy minimization approach

$$\min_u \{E(u) = \mathcal{R}(u) + \mathcal{D}(u, f)\} ,$$

  where f is the input data and u is the unknown solution
- Energy functional is designed such that low-energy states reflect the physical properties of the problem
- Minimizer provides the best (in the sense of the model) solution to the problem

Energy minimization methods

- It is in general not possible to solve inverse problems directly
- Add some smoothness assumption to the unknown solution
- Leads to the energy minimization approach

$$\min_u \{E(u) = \mathcal{R}(u) + \mathcal{D}(u, f)\} ,$$

  where f is the input data and u is the unknown solution
- Energy functional is designed such that low-energy states reflect the physical properties of the problem
- Minimizer provides the best (in the sense of the model) solution to the problem

- Different philosophies:
  - Discrete MRF setting: Images are represented as graphs $\mathcal{G}(\mathcal{V}, \mathcal{E})$, consisting of a node set $\mathcal{V}$, and an edge set $\mathcal{E}$. Each node $v \in \mathcal{V}$ can take a label from a discrete label set $\mathcal{U} \subset \mathbb{Z}$, i.e. $u(v) \in \mathcal{U}$
  - Continuous Variational setting: Images are considered as continuous functions $u : \Omega \to \mathbb{R}$, where $\Omega \subset \mathbb{R}^n$ is the image domain.

Energy minimization methods

- It is in general not possible to solve inverse problems directly
- Add some smoothness assumption to the unknown solution
- Leads to the energy minimization approach

$$\min_u \{E(u) = \mathcal{R}(u) + \mathcal{D}(u, f)\} \ ,$$

  where f is the input data and u is the unknown solution
- Energy functional is designed such that low-energy states reflect the physical properties of the problem
- Minimizer provides the best (in the sense of the model) solution to the problem

- Different philosophies:
  - Discrete MRF setting: Images are represented as graphs $\mathcal{G}(\mathcal{V}, \mathcal{E})$, consisting of a node set $\mathcal{V}$, and an edge set $\mathcal{E}$. Each node $v \in \mathcal{V}$ can take a label from a discrete label set $\mathcal{U} \subset \mathbb{Z}$, i.e. $u(v) \in \mathcal{U}$
  - Continuous Variational setting: Images are considered as continuous functions $u : \Omega \to \mathbb{R}$, where $\Omega \subset \mathbb{R}^n$ is the image domain.

- Link to statistics: In a Bayesian setting, the energy relates to the posterior probability via

$$p(u|f) = \frac{1}{Z} exp(-E(u))$$

- Computing the minimizer of $E(u)$ is equivalent to MAP estimation on $p(u|f)$

Example: Total variation based image restoration

Image model: $f = k * u + n$, blur kernel $k =$ 

Variational model: [Rudin, Osher, Fatemi '92]

$$\min_u \int_\Omega |Du| + \frac{\lambda}{2}\|k * u - f\|_2^2$$

(a) Degraded image f

Example: Total variation based image restoration

Image model: $f = k * u + n$, blur kernel $k =$ 

Variational model: [Rudin, Osher, Fatemi '92]

$$\min_u \int_\Omega |Du| + \frac{\lambda}{2} \|k * u - f\|_2^2$$

(a) Degraded image f

(b) Reconstructed image u

Optimization problems are unsolvable

Consider the following general mathematical optimization problem:

$$\begin{aligned} \min\ & f_0(x) \\ \text{s.t.}\quad & f_i(x) \le 0\ , \quad i = 1 \dots m \\ & x \in S\ , \end{aligned}$$

where $f_0(x) ... f_m(x)$ are real-valued functions, $x = (x_1, ... x_n)^T \in \mathbb{R}^n$ is a n-dimensional real-valued vector, and S is a subset of $\mathbb{R}^n$

How to solve this problem?

Optimization problems are unsolvable

Consider the following general mathematical optimization problem:

$$\begin{aligned} &\min f_0(x) \\ \text{s.t.} \quad &f_i(x) \leq 0 \;, \quad i = 1 \ldots m \\ &x \in S \;, \end{aligned}$$

where $f_0(x) \ldots f_m(x)$ are real-valued functions, $x = (x_1, \ldots x_n)^T \in \mathbb{R}^n$ is a n-dimensional real-valued vector, and S is a subset of $\mathbb{R}^n$

How to solve this problem?

- Naive: "Download a commercial package ..."

Optimization problems are unsolvable

Consider the following general mathematical optimization problem:

$$\begin{aligned} &\min f_0(x) \\ \text{s.t.} \quad & f_i(x) \leq 0\,, \quad i = 1 \ldots m \\ & x \in S\,, \end{aligned}$$

where $f_0(x) ... f_m(x)$ are real-valued functions, $x = (x_1, ... x_n)^T \in \mathbb{R}^n$ is a n-dimensional real-valued vector, and S is a subset of $\mathbb{R}^n$

How to solve this problem?

- Naive: "Download a commercial package ..."
- Reality: "Finding a solution is far from being trivial!"

Optimization problems are unsolvable

Consider the following general mathematical optimization problem:

$$
\begin{aligned}
& \min f_0(x) \\
\text{s.t.} \quad & f_i(x) \leq 0 \,, \quad i = 1 \ldots m \\
& x \in S \,,
\end{aligned}
$$

where $f_0(x) \ldots f_m(x)$ are real-valued functions, $x = (x_1, \ldots x_n)^T \in \mathbb{R}^n$ is a n-dimensional real-valued vector, and S is a subset of $\mathbb{R}^n$

How to solve this problem?

- Naive: "Download a commercial package ..."
- Reality: "Finding a solution is far from being trivial!"

- Example: Minimize a function of 10 variables over the unit box
- Can take more than 30 million years to find an approximate solution!

Optimization problems are unsolvable

Consider the following general mathematical optimization problem:

$$\begin{aligned} &\min f_0(x) \\ \text{s.t.} \quad & f_i(x) \leq 0 \,, \quad i = 1 \ldots m \\ & x \in S \,, \end{aligned}$$

where $f_0(x) \ldots f_m(x)$ are real-valued functions, $x = (x_1, \ldots x_n)^T \in \mathbb{R}^n$ is a n-dimensional real-valued vector, and S is a subset of $\mathbb{R}^n$

How to solve this problem?

- Naive: "Download a commercial package ..."
- Reality: "Finding a solution is far from being trivial!"

- Example: Minimize a function of 10 variables over the unit box
- Can take more than 30 million years to find an approximate solution!

- "Optimization problems are unsolvable"

[Nesterov '04]

Convex versus non-convex

- Non-convex problems
  - Often give more accurate models
  - In general no chance to find the global minimizer
  - Result strongly depends on the initialization
  - Dilemma: Wrong model or wrong algorithm?

Convex versus non-convex

- Non-convex problems
  - Often give more accurate models
  - In general no chance to find the global minimizer
  - Result strongly depends on the initialization
  - Dilemma: Wrong model or wrong algorithm?

- Convex problems
  - Convex models often inferior
  - Any local minimizer is a global minimizer
  - Result is independent of the initialization
  - Note: Convex does not mean easy!

Convex versus non-convex

- Non-convex problems
  - Often give more accurate models
  - In general no chance to find the global minimizer
  - Result strongly depends on the initialization
  - Dilemma: Wrong model or wrong algorithm?

- Convex problems
  - Convex models often inferior
  - Any local minimizer is a global minimizer
  - Result is independent of the initialization
  - Note: Convex does not mean easy!

- Current research: Bridging the gap between convex and non-convex optimization
  - Convex approximations of non-convex models
  - New models
  - Algorithms
  - Bounds

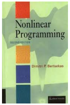
Overview

Literatur on convex analysis and optimization

- Convex optimization, [Boyd,Vandenberghe '04]

- Introductory lectures on convex optimization, [Nesterov '04]

- Nonlinear programming, [Bertsekas '99]

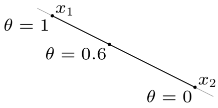


- Variational analysis, [Rockafellar, Wets '88]

Convex sets

- Consider two points x_1, $x_2 \in \mathbb{R}^n$
- The line segment between these two points is given by the points

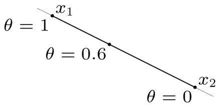
$$x = \theta x_1 + (1 - \theta) x_2, \; \theta \in [0, 1]$$



Convex sets

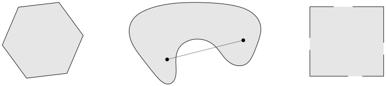
- Consider two points $x_1,\ x_2 \in \mathbb{R}^n$
- The line segment between these two points is given by the points

$$x = \theta x_1 + (1-\theta)x_2,\ \theta \in [0,1]$$



- A set C is said to be convex, if it contains all line segments between any two points in the set

$$x_1,\ x_2 \in C \quad \Rightarrow \quad \theta x_1 + (1-\theta)x_2 \in C,\ \forall\theta \in [0,1]$$

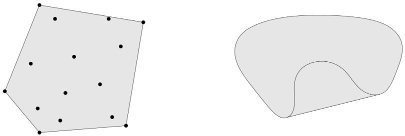


Convex combination and convex hull

- Convex combination of a set of points $\{x_1, ..., x_k\}$, $x_i \in \mathbb{R}^n$ is given by

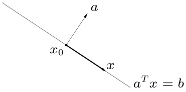
$$x = \sum_{i=1}^{k} \theta_i x_i, \ \theta_i \geq 0, \ \sum_{i=1}^{k} \theta_i = 1$$

- The convex hull of a set of points $\{x_1, ..., x_k\}$ is the set of all convex combinations



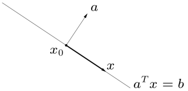
A few important convex sets

- Hyperplane: $\{x \mid a^T x = b\}$

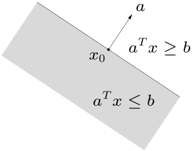


A few important convex sets

- Hyperplane: $\{x \mid a^T x = b\}$

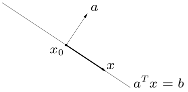


- Halfspace: $\{x \mid a^T x \leq b\}$

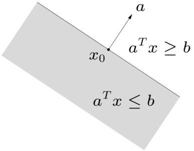


A few important convex sets

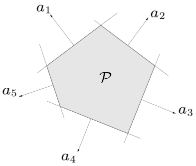
- Hyperplane: $\{x \mid a^T x = b\}$



- Halfspace: $\{x \mid a^T x \leq b\}$



- Polyhedra: Interesection of finitely many hyperplanes and halfspaces

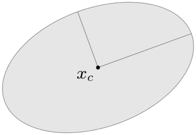


A few important convex sets

- Norm Ball: $\{x \mid \|x - x_c\| \leq r\}$, where $\|\cdot\|$ is any norm

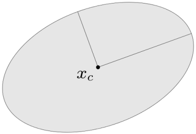
A few important convex sets

- Norm Ball: $\{x \mid \|x - x_c\| \leq r\}$, where $\|\cdot\|$ is any norm
- Ellipsoid: $\{x \mid (x - x_c)P^{-1}(x - x_c) \leq 1\}$, P symmetric and positive definite

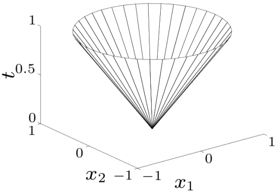


A few important convex sets

- Norm Ball: $\{x \mid \|x - x_c\| \leq r\}$, where $\|\cdot\|$ is any norm
- Ellipsoid: $\{x \mid (x - x_c)P^{-1}(x - x_c) \leq 1\}$, P symmetric and positive definite



- Norm cone: $\{(x, t) \mid \|x\| \leq t\}$



How to show convexity of a set?

- Check definition

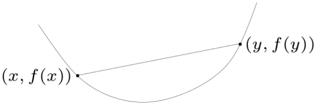
$$x_1,\ x_2 \in C \quad \Rightarrow \quad \theta x_1 + (1-\theta)x_2 \in C,\ \forall \theta \in [0,1]$$

- Show that the set is obtained from simple sets by operations that preserve convexity
  - Intersection
  - Affine transformation (scaling, translation, ...)
  - ...

Convex functions

- Definition: A function $f : \mathbb{R}^n \to \mathbb{R}$ is convex, if $\mathrm{dom}\, f$ is a convex set and

$$f(\theta x + (1-\theta)y) \leq \theta f(x) + (1-\theta)f(y),\ \forall x, y \in \mathrm{dom}\, f,\ \theta \in [0,1]$$



- f is said to be concave, if $-f$ is convex
- f is strictly convex, if

$$f(\theta x + (1-\theta)y) < \theta f(x) + (1-\theta)f(y),\ \forall x, y \in \mathrm{dom}\, f,\ x \neq y,\ \theta \in (0,1)$$

Examples of convex functions

Examples on $\mathbb{R}$

- Linear: $ax + b$, $a, b \in \mathbb{R}$
- Exponential: e^{ax}, $a \in R$
- Powers: x^{α}, for $x \geq 0$, $\alpha \geq 1$ or $\alpha \leq 0$
- Powers of absolute values: $|x|^{\alpha}$, $\alpha \geq 0$
- Negative entropy: $x \log x$, for $x \geq 0$

Examples of convex functions

Examples on $\mathbb{R}$

- Linear: $ax + b$, $a, b \in \mathbb{R}$
- Exponential: e^{ax}, $a \in R$
- Powers: x^α, for $x \geq 0$, $\alpha \geq 1$ or $\alpha \leq 0$
- Powers of absolute values: $|x|^\alpha$, $\alpha \geq 0$
- Negative entropy: $x \log x$, for $x \geq 0$

Examples on $\mathbb{R}^n$

- Affine: $a^T x + b$, $a, b \in \mathbb{R}^n$
- Norms: $\|x\|_p = \left(\sum_{i=1}^n |x_i|^p\right)^{\frac{1}{p}}$, $p \geq 1$

Examples of convex functions

Examples on $\mathbb{R}$

- Linear: $ax + b$, $a, b \in \mathbb{R}$
- Exponential: e^{ax}, $a \in R$
- Powers: x^α, for $x \geq 0$, $\alpha \geq 1$ or $\alpha \leq 0$
- Powers of absolute values: $|x|^\alpha$, $\alpha \geq 0$
- Negative entropy: $x \log x$, for $x \geq 0$

Examples on $\mathbb{R}^n$

- Affine: $a^T x + b$, $a, b \in \mathbb{R}^n$
- Norms: $\|x\|_p = \left(\sum_{i=1}^n |x_i|^p\right)^{\frac{1}{p}}$, $p \geq 1$

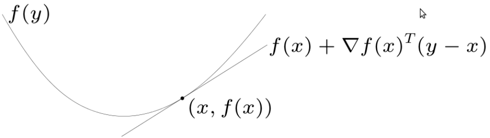
Examples on $\mathbb{R}^{m \times n}$

- Affine: $\mathrm{tr}(A^T X) + b = \sum_{i=1}^m \sum_{j=1}^n A_{ij} X_{ij} + b$, $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}$
- Spectral norm (max non-singular value): $\|X\|_2$

Sufficient conditions of convexity

- **First-order condition:** A differentiable function f with convex domain is convex iff

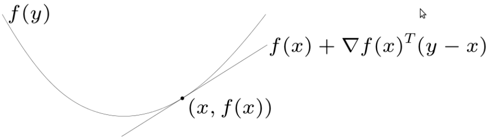
$$f(y) \geq f(x) + \nabla f(x)^T (y - x), \ \forall x, y \in \mathrm{dom}\ f$$



Sufficient conditions of convexity

- **First-order condition:** A differentiable function f with convex domain is convex iff

$$f(y) \geq f(x) + \nabla f(x)^T (y - x), \ \forall x, y \in \operatorname{dom} f$$



- **Second-order condition:** A twice differentiable function f with convex domain is convex iff

$$\nabla^2 f(x) \succeq 0, \ \forall x \in \operatorname{dom} f$$

- It is strictly convex if the assertion becomes $\nabla^2 f(x) \succ 0$

Example

- Quadratic over linear function

$$f(x,y) = \frac{x^2}{y}$$

Example

- Quadratic over linear function

$$f(x,y) = \frac{x^2}{y}$$

- Computing the gradient and Hessian

$$\nabla f(x,y) = \left(\frac{2x}{y}, -\frac{x^2}{y^2}\right), \quad \nabla^2 f(x,y) = \frac{2}{y^3}(y,-x)^T(y,-x)$$

Example

- Quadratic over linear function

$$f(x,y) = \frac{x^2}{y}$$

- Computing the gradient and Hessian

$$\nabla f(x,y) = \left(\frac{2x}{y}, -\frac{x^2}{y^2}\right), \quad \nabla^2 f(x,y) = \frac{2}{y^3}(y,-x)^T(y,-x)$$

- Second-order sufficient condition: $\nabla^2 f(x,y) \succeq 0$ for $y \geq 0$

Example

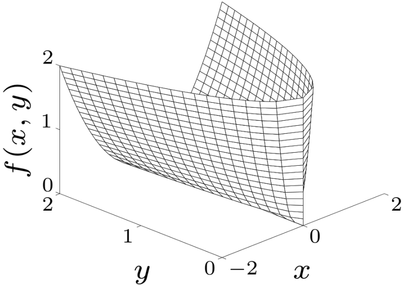
- Quadratic over linear function

$$f(x,y) = \frac{x^2}{y}$$

- Computing the gradient and Hessian

$$\nabla f(x,y) = \left(\frac{2x}{y}, -\frac{x^2}{y^2}\right), \quad \nabla^2 f(x,y) = \frac{2}{y^3}(y,-x)^T(y,-x)$$

- Second-order sufficient condition: $\nabla^2 f(x,y) \succeq 0$ for $y \geq 0$



How to verify convexity?

- Verify the definition of convex functions

$$f(\theta x + (1-\theta)y) \leq \theta f(x) + (1-\theta)f(y), \ \forall x, y \in \mathrm{dom}\ f, \ \theta \in [0,1]$$

- Sometimes simpler to consider the 1D case

How to verify convexity?

- Verify the definition of convex functions

$$f(\theta x + (1-\theta)y) \leq \theta f(x) + (1-\theta) f(y),\ \forall x, y \in \mathrm{dom}\, f,\ \theta \in [0,1]$$

- Sometimes simpler to consider the 1D case

- Check for $\nabla^2 f \succeq 0$

$$\nabla^2 f \succeq 0 \quad \text{iff} \quad v^T (\nabla^2 f) v \geq 0,\ \forall v \neq 0$$

How to verify convexity?

- Verify the definition of convex functions

$$f(\theta x + (1-\theta)y) \leq \theta f(x) + (1-\theta)f(y),\ \forall x, y \in \mathrm{dom}\, f,\ \theta \in [0,1]$$

- Sometimes simpler to consider the 1D case

- Check for $\nabla^2 f \succeq 0$

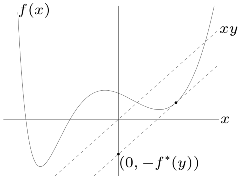
$$\nabla^2 f \succeq 0 \quad \text{iff} \quad v^T(\nabla^2 f)v \geq 0,\ \forall v \neq 0$$

- Show that f is obtained from simple convex functions by operations that preserve convexity
  - Non-negative weihgted sum: $f = \sum_i f_i$, is convex if $\alpha_i > 0$, f_i are convex
  - Composition with an affine function: $f(a^T x + b)$ is convex if f is convex
  - Pointwise maximum: $f(x) = \max\{f_1(x), \ldots, f_n(x)\}$ is convex if $f_1 \ldots f_n$ are convex

The convex conjugate

- The convex conjugate $f^*(y)$ of a function $f(x)$ is defined as

$$f^*(y) = \sup_{x \in \mathrm{dom}\ f} \langle x, y \rangle - f(x)$$



- $f^*(y)$ is a convx function (pointwise supremum over linear functions)
- The biconjugate function $f^{**}(x)$ is the largest convex l.s.c. function below $f(x)$
- If $f(x)$ is a convex, l.s.c. function, $f^{**}(x) = f(x)$

Examples

- $f(x) = |x|$:

$$f^*(y) = \sup_x \langle x, y \rangle - |x| = \begin{cases} 0 & \text{if} \quad |y| \leq 1 \\ \infty & \text{else} \end{cases}$$

- $f(x) = \frac{1}{2} x^T Q x$, Q, posoitive definite

$$f^*(y) = \sup_x \langle x, y \rangle - \frac{1}{2} x^T Q x = \frac{1}{2} y^T Q^{-1} y$$

Duality

- Fenchel's duality theorem

Overview

Convex optimization

- A general convex optimization problem is defined as

$$\begin{array}{ll} \min & f_0(x) \\ \text{s.t.} & f_i(x) \leq 0\,, \quad i = 1 \ldots m \\ & Ax = b \end{array}$$

where $f_0(x) \ldots f_p(x)$ are real-valued convex functions, $x = (x_1, \ldots x_n)^T \in \mathbb{R}^n$ is a m-dimensional real-valued vector, $Ax = b$ are affine equality constraints

Convex optimization

- A general convex optimization problem is defined as

$$\begin{array}{ll} \min & f_0(x) \\ \text{s.t.} & f_i(x) \leq 0\,, \quad i = 1 \ldots m \\ & Ax = b \end{array}$$

where $f_0(x) \ldots f_p(x)$ are real-valued convex functions, $x = (x_1, \ldots x_n)^T \in \mathbb{R}^n$ is a m-dimensional real-valued vector, $Ax = b$ are affine equality constraints

- Convex optimization problems are considered as solvable!

Convex optimization

- A general convex optimization problem is defined as

$$\begin{array}{ll} \min & f_0(x) \\ \text{s.t.} & f_i(x) \le 0\,, \quad i = 1\dots m \\ & Ax = b \end{array}$$

where $f_0(x)...f_p(x)$ are real-valued convex functions, $x = (x_1, ...x_n)^T \in \mathbb{R}^n$ is a m-dimensional real-valued vector, $Ax = b$ are affine equality constraints

- Convex optimization problems are considered as solvable!

- “The great watershed in optimization isn’t between linearity and non-linearity, but convexity and non-convexity” [Rockafellar ’93]

Black box convex optimization

Generic iterative algorithm for convex optimization:

1. Pick any initial vector $x^0 \in \mathbb{R}^n$, set $k = 0$
2. Compute search direction $d^k \in \mathbb{R}^n$
3. Choose step size τ^k such that $f(x^k + \tau^k d^k) < f(x^k)$
4. Set $x^{k+1} = x^k + \tau^k d^k$, $k = k + 1$
5. Stop if converged, else goto 2

Black box convex optimization

Generic iterative algorithm for convex optimization:

1. Pick any initial vector $x^0 \in \mathbb{R}^n$, set $k = 0$
2. Compute search direction $d^k \in \mathbb{R}^n$
3. Choose step size τ^k such that $f(x^k + \tau^k d^k) < f(x^k)$
4. Set $x^{k+1} = x^k + \tau^k d^k$, $k = k + 1$
5. Stop if converged, else goto 2

Different methods to determine the search direction d^k

- steepest descend
- conjugate gradients
- Newton, quasi-Newton
- Working-horse for the lazy: Limited memory BFGS quasi-Newton method [Nocedal '80]

Black box convex optimization

Generic iterative algorithm for convex optimization:

1. Pick any initial vector $x^0 \in \mathbb{R}^n$, set $k = 0$
2. Compute search direction $d^k \in \mathbb{R}^n$
3. Choose step size τ^k such that $f(x^k + \tau^k d^k) < f(x^k)$
4. Set $x^{k+1} = x^k + \tau^k d^k$, $k = k + 1$
5. Stop if converged, else goto 2

Different methods to determine the search direction d^k

- steepest descend
- conjugate gradients
- Newton, quasi-Newton
- Working-horse for the lazy: Limited memory BFGS quasi-Newton method [Nocedal '80]

Black box methods do not exploit the structure of the problem and hence are often less effective

Structured convex optimization

There are a number of efficient solvers available for important types of structured convex optimization problems [Boyd, Vandenberghe '04]:

Structured convex optimization

There are a number of efficient solvers available for important types of structured convex optimization problems [Boyd, Vandenberghe '04]:

- Least squares problems:

$$\min_x \|Ax - b\|_2^2$$

Structured convex optimization

There are a number of efficient solvers available for important types of structured convex optimization problems [Boyd, Vandenberghe '04]:

- Least squares problems:

$$\min_x \|Ax - b\|_2^2$$

- Quadratic program (QP) and linear program (LP) ($Q = 0$):

$$\begin{aligned} \min \ & \tfrac{1}{2}x^T Q x + c^T x, \ Q \succeq 0 \\ \text{s.t. } & Ax = b, \ l \leq x \leq u \end{aligned}$$

Structured convex optimization

There are a number of efficient solvers available for important types of structured convex optimization problems [Boyd, Vandenberghe '04]:

- Least squares problems:

$$\min_x \|Ax - b\|_2^2$$

- Quadratic program (QP) and linear program (LP) ($Q = 0$):

$$\begin{aligned} \min\ & \tfrac{1}{2}x^T Qx + c^T x,\ Q \succeq 0 \\ \text{s.t. } & Ax = b,\ l \leq x \leq u \end{aligned}$$

- Second order cone program (SOCP)

$$\begin{aligned} &\min f^T x \\ \text{s.t. } \|A_i x + b_i\|^2 \leq c_i^T x + d_i,\ & i = 1 \dots m,\ Fx = g \end{aligned}$$

Structured convex optimization

There are a number of efficient solvers available for important types of structured convex optimization problems [Boyd, Vandenberghe '04]:

- Least squares problems:

$$\min_x \|Ax - b\|_2^2$$

- Quadratic program (QP) and linear program (LP) ($Q = 0$):

$$\begin{aligned} \min\ & \tfrac{1}{2} x^T Q x + c^T x,\ Q \succeq 0 \\ \text{s.t. } & Ax = b,\ l \leq x \leq u \end{aligned}$$

- Second order cone program (SOCP)

$$\begin{gathered} \min f^T x \\ \text{s.t. } \|A_i x + b_i\|^2 \leq c_i^T x + d_i,\ i = 1...m,\ Fx = g \end{gathered}$$

- Many problems can be formulated in these frameworks
- Fast methods available (simplex, interior point, ...)
- Sometimes ineffective for large-scale problems

Overview

A class of problems

Let us consider the following class of structured convex optimization problems

$$\min_{x \in X} F(Kx) + G(x) \ ,$$

- $K : X \to Y$ is a linear and continuous operator from a Hilbert space X to a Hilbert space Y.
- $F : Y \to \mathbb{R} \cup \{\infty\}$, $G : X \to \mathbb{R} \cup \{\infty\}$ are "simple" convex, proper, l.s.c. functions, and hence have an easy to compute prox operator:

$$\mathrm{prox}_G(z) = (I + \partial G)^{-1}(z) = \arg\min_x \frac{\|x - z\|^2}{2} + G(x)$$

A class of problems

Let us consider the following class of structured convex optimization problems

$$\min_{x \in X} F(Kx) + G(x) \ ,$$

- $K : X \to Y$ is a linear and continuous operator from a Hilbert space X to a Hilbert space Y.
- $F : Y \to \mathbb{R} \cup \{\infty\}$, $G : X \to \mathbb{R} \cup \{\infty\}$ are "simple" convex, proper, l.s.c. functions, and hence have an easy to compute prox operator:

$$\mathrm{prox}_G(z) = (I + \partial G)^{-1}(z) = \arg\min_x \frac{\|x - z\|^2}{2} + G(x)$$

- It turns out that many standard problems can be cast in this framework.
- There exists a vast literature of numerical algorithms to solve this class of problems

Examples

- Image restoration: The ROF model

$$\min_u \|\nabla u\|_1 + \frac{\lambda}{2}\|k * u - f\|_2^2 \ ,$$

- Compressed sensing: Basis pursuit problem (LASSO)

$$\min_x \|x\|_1 + \frac{\lambda}{2}\|Ax - b\|_2^2$$

Examples

- Image restoration: The ROF model

$$\min_u \|\nabla u\|_1 + \frac{\lambda}{2}\|k * u - f\|_2^2 \, ,$$

- Compressed sensing: Basis pursuit problem (LASSO)

$$\min_x \|x\|_1 + \frac{\lambda}{2}\|Ax - b\|_2^2$$

- Machine learning: Linear support vector machine

$$\min_{w,b} \frac{\lambda}{2}\|w\|_2^2 + \sum_{i=1}^{n} \max\left(0, 1 - y_i\left(\langle w, x_i\rangle + b\right)\right)$$

Examples

- Image restoration: The ROF model

$$\min_{u} \|\nabla u\|_1 + \frac{\lambda}{2}\|k * u - f\|_2^2 \ ,$$

- Compressed sensing: Basis pursuit problem (LASSO)

$$\min_{x} \|x\|_1 + \frac{\lambda}{2}\|Ax - b\|_2^2$$

- Machine learning: Linear support vector machine

$$\min_{w,b} \frac{\lambda}{2}\|w\|_2^2 + \sum_{i=1}^{n} \max\left(0, 1 - y_i\left(\langle w, x_i\rangle + b\right)\right)$$

- General linear programming problems

$$\min_{x} \langle c, x\rangle \ , \ \text{s.t.} \left\{ \begin{array}{ccc} Ax & = & b \\ x & \geq & 0 \end{array} \right.$$

Primal, dual, primal-dual

The real power of convex optimization comes through duality

Recall the convex conjugate:

$$F^*(y) = \sup_{x \in X} \langle x, y \rangle - F(x) \ , \quad F^{**}(x) = \sup_{y \in Y} \langle x, y \rangle - F^*(y)$$

If $F(x)$ convex, l.s.c. than $F^{**}(x) = F(x)$

Primal, dual, primal-dual

The real power of convex optimization comes through duality

Recall the convex conjugate:

$$F^*(y) = \sup_{x \in X} \langle x, y \rangle - F(x) \,, \quad F^{**}(x) = \sup_{y \in Y} \langle x, y \rangle - F^*(y)$$

If $F(x)$ convex, l.s.c. than $F^{**}(x) = F(x)$

$$\min_{x \in X} F(Kx) + G(x) \quad \text{(Primal)}$$

Primal, dual, primal-dual

The real power of convex optimization comes through duality

Recall the convex conjugate:

$$F^*(y) = \sup_{x \in X} \langle x, y \rangle - F(x) \,, \quad F^{**}(x) = \sup_{y \in Y} \langle x, y \rangle - F^*(y)$$

If $F(x)$ convex, l.s.c. than $F^{**}(x) = F(x)$

$$\min_{x \in X} F(Kx) + G(x) \quad \text{(Primal)}$$

$$\min_{x \in X} \max_{y \in Y} \langle Kx, y \rangle + G(x) - F^*(y) \quad \text{(Primal-Dual)}$$

TU Graz
Graz University of Technology

Primal, dual, primal-dual

The real power of convex optimization comes through duality

Recall the convex conjugate:

$$F^*(y) = \sup_{x \in X} \langle x, y \rangle - F(x) \,, \quad F^{**}(x) = \sup_{y \in Y} \langle x, y \rangle - F^*(y)$$

If $F(x)$ convex, l.s.c. than $F^{**}(x) = F(x)$

$$\min_{x \in X} F(Kx) + G(x) \quad \text{(Primal)}$$

$$\min_{x \in X} \max_{y \in Y} \langle Kx, y \rangle + G(x) - F^*(y) \quad \text{(Primal-Dual)}$$

$$\max_{y \in Y} -(F^*(y) + G^*(-K^*y)) \quad \text{(Dual)}$$

Allows to compute the duality gap, which is a measure of optimality

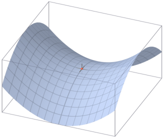
Optimality conditions

We focus on the primal-dual formulation:

$$\min_{x \in X} \max_{y \in Y} \langle Kx, y \rangle + G(x) - F^*(y)$$

We assume, there exists a saddle-point $(\hat{x}, \hat{y}) \in X \times Y$ which satisfies the Euler-Lagrange equations

$$\begin{cases} K\hat{x} - \partial F^*(\hat{y}) \ni 0 \\ K^*\hat{y} + \partial G(\hat{x}) \ni 0 \end{cases}$$



Standard first order approaches

1. Classical Arrow-Hurwicz method [Arrow, Hurwicz, Uzawa '58]

$$\begin{cases} y^{n+1} = (I + \tau\partial F^*)^{-1}(y^n + \tau K x^n) \\ x^{n+1} = (I + \tau\partial G)^{-1}(x^n - \tau K^* y^{n+1}) \end{cases}$$

- Alternating forward-backward step in the dual variable and the primal variable
- Convergence under quite restrictive assumptions on τ
- For some problems very fast, e.g. ROF problem using adaptive time steps [Zhu, Chan, '08]

2. Proximal point method [Martinet '70], [Rockafellar '76] for the search of zeros of an operator, i.e. $0 \in T(x)$

$$x^{n+1} = (I + \tau^n T)^{-1}(x^n)$$

- Very simple iteration
- In our framework $T(x) = K^*\partial F(Kx) + \partial G(x)$
- Unfortunately, in most interesting cases, $(I + \tau^n T)^{-1}$ is hard to evaluate
- Hence, the practical interest is limited

Standard first order approaches

3. Douglas-Rachford splitting (DRS) [Mercier, Lions '79]
   - Special case of the proximal point method if the operator T is the sum of two operators, i.e. $T = A + B$

$$\begin{cases} w^{n+1} = (I + \tau A)^{-1}(2x^n - w^n) + w^n - x^n \\ x^{n+1} = (I + \tau B)^{-1}(w^{n+1}) \end{cases}$$

   - Only needs to evaluate the resolvent operator with respect to A and B
   - Let $A = K^* \partial F(K)$ and $B = \partial G$, the DRS algorithm becomes

$$\begin{cases} w^{n+1} = \arg\min_v F(Kv) + \frac{1}{2\tau} \|v - (2x^n - w^n)\|^2 + w^n - x^n \\ x^{n+1} = \arg\min_x G(x) + \frac{1}{2\tau} \left\|x - w^{n+1}\right\|^2 \end{cases}$$

   - Equivalent to the alternating direction method of multipliers (ADMM) [Eckstein, Bertsekas '89]
   - Equivalent to the split-Bregman iteration [Goldstein, Osher '09]
   - Equivalent to an alternating minimization on the augmented Lagrangian formulation

A simple primal-dual algorithm

Proposed in a series of papers: [Pock, Cremers, Bischof, Chambolle, '09], [Chambolle, Pock, '10], [Pock, Chambolle, '11]

- Initialization: Choose s.p.d. T, Σ, $\theta \in [0,1]$, $(x^0, y^0) \in X \times Y$.
- Iterations ($n \geq 0$): Update x^n, y^n as follows:

$$\begin{cases} x^{n+1} &= (I + \mathrm{T}\partial G)^{-1}(x^n - \mathrm{T}K^* y^n) \\ y^{n+1} &= (I + \Sigma\partial F^*)^{-1}(y^n + \Sigma K(x^{n+1} + \theta(x^{n+1} - x^n))) \end{cases}$$

- Alternates gradient descend in x and gradient ascend in y
- Linear extrapolation of iterates of x in the y step
- T, Σ can be seen as preconditioning matrices
- Can be derived from a pre-conditioned DRS splitting algorithm
- Can be seen as a relaxed Arrow-Hurwicz scheme

- The iterations of PD can be written as the variational inequality [He,Yuan '10]

$$\left\langle \begin{pmatrix} x - x^{n+1} \\ y - y^{n+1} \end{pmatrix}, \begin{pmatrix} \partial G(x^{n+1}) + K^* y^{n+1} \\ \partial F^*(y^{n+1}) - K x^{n+1} \end{pmatrix} + M \begin{pmatrix} x^{n+1} - x^n \\ y^{n+1} - y^n \end{pmatrix} \right\rangle \geq 0 \,,$$

$$M = \begin{bmatrix} \mathrm{T}^{-1} & -K^* \\ -\theta K & \Sigma^{-1} \end{bmatrix}$$

Relations to the proximal-point algorithm

- The iterations of PD can be written as the variational inequality [He,Yuan '10]

$$\left\langle \begin{pmatrix} x - x^{n+1} \\ y - y^{n+1} \end{pmatrix}, \begin{pmatrix} \partial G(x^{n+1}) + K^* y^{n+1} \\ \partial F^*(y^{n+1}) - K x^{n+1} \end{pmatrix} + M \begin{pmatrix} x^{n+1} - x^n \\ y^{n+1} - y^n \end{pmatrix} \right\rangle \geq 0 \,,$$

$$M = \begin{bmatrix} \mathrm{T}^{-1} & -K^* \\ -\theta K & \Sigma^{-1} \end{bmatrix}$$

- This is exactly the proximal-point algorithm, but with a norm in M.

Relations to the proximal-point algorithm

- The iterations of PD can be written as the variational inequality [He,Yuan '10]

$$\left\langle \begin{pmatrix} x - x^{n+1} \\ y - y^{n+1} \end{pmatrix}, \begin{pmatrix} \partial G(x^{n+1}) + K^* y^{n+1} \\ \partial F^*(y^{n+1}) - K x^{n+1} \end{pmatrix} + M \begin{pmatrix} x^{n+1} - x^n \\ y^{n+1} - y^n \end{pmatrix} \right\rangle \geq 0 \,,$$

$$M = \begin{bmatrix} \mathrm{T}^{-1} & -K^* \\ -\theta K & \Sigma^{-1} \end{bmatrix}$$

- This is exactly the proximal-point algorithm, but with a norm in M.
- Convergence is ensured if M is symmetric and positive definite

Relations to the proximal-point algorithm

- The iterations of PD can be written as the variational inequality [He,Yuan '10]

$$\left\langle \begin{pmatrix} x - x^{n+1} \\ y - y^{n+1} \end{pmatrix}, \begin{pmatrix} \partial G(x^{n+1}) + K^* y^{n+1} \\ \partial F^*(y^{n+1}) - K x^{n+1} \end{pmatrix} + M \begin{pmatrix} x^{n+1} - x^n \\ y^{n+1} - y^n \end{pmatrix} \right\rangle \geq 0 \,,$$

$$M = \begin{bmatrix} \mathrm{T}^{-1} & -K^* \\ -\theta K & \Sigma^{-1} \end{bmatrix}$$

- This is exactly the proximal-point algorithm, but with a norm in M.
- Convergence is ensured if M is symmetric and positive definite
- This is the case if $\theta = 1$ and the assertion $\|\Sigma^{\frac{1}{2}} K \mathrm{T}^{\frac{1}{2}}\|^2 < 1$ is fulfilled.

A family of diagonal preconditioners

- It is important to choose T, Σ such that the prox-operators are still easy to compute
- Restrict the preconditioning matrices to diagonal matrices
- It turns out that: Let $K \in \mathbb{R}^{m\times n}$, $\mathrm{T} = \text{diag}(\boldsymbol{\tau})$ and $\Sigma = \text{diag}(\boldsymbol{\sigma})$ such that

$$\tau_j = \frac{1}{\sum_{i=1}^{m} |K_{i,j}|^{2-\alpha}}, \quad \sigma_i = \frac{1}{\sum_{j=1}^{n} |K_{i,j}|^{\alpha}}$$

then for any $\alpha \in [0, 2]$

$$\|\Sigma^{\frac{1}{2}} K \mathrm{T}^{\frac{1}{2}}\|^2 \leq 1 \,.$$

- Gives an automatic problem-dependent choice of the primal and dual steps
- Allows to apply the algorithm in a plug-and-play fashion
- For $\alpha = 0$, equivalent to the alternating step method [Eckstein, Bertsekas, '90]

Convergence rates

The algorithm gives different convergence rates on different problem classes [Chambolle, Pock, '10]

- F^* and G nonsmooth: $O(1/N)$

Convergence rates

The algorithm gives different convergence rates on different problem classes [Chambolle, Pock, '10]

- F^* and G nonsmooth: $O(1/N)$
- F^* or G uniformly convex: $O(1/N^2)$

Convergence rates

The algorithm gives different convergence rates on different problem classes [Chambolle, Pock, '10]

- F^* and G nonsmooth: $O(1/N)$
- F^* or G uniformly convex: $O(1/N^2)$
- F^* and G uniformly convex: $O(\omega^N)$, $\omega < 1$

Convergence rates

The algorithm gives different convergence rates on different problem classes [Chambolle, Pock, '10]

- F^* and G nonsmooth: $O(1/N)$
- F^* or G uniformly convex: $O(1/N^2)$
- F^* and G uniformly convex: $O(\omega^N)$, $\omega < 1$

- Coincides with so far best known rates of first-order methods

Parallel computing?

The algorithm basically computes matrix-vector products
The matrices are usually very sparse

Parallel computing?

The algorithm basically computes matrix-vector products
The matrices are usually very sparse

- One simple processor for each unknown
- Each processor has a small amount of local memory
- According to the structure of K, each processor can inter-change data with its neighboring processors

Parallel computing?

The algorithm basically computes matrix-vector products
The matrices are usually very sparse

- One simple processor for each unknown
- Each processor has a small amount of local memory
- According to the structure of K, each processor can inter-change data with its neighboring processors

Recent GPUs already go into this direction