A Tightrope Walk Between Convexity and Non-convexity in Computer Vision

Thomas Pock

Institute for Computer Graphics and Vision,
Graz University of Technology,
8010 Graz, Austria

Qualcomm Augmented Reality Lecture, 29.11.2013

Joint work with A. Chambolle, D. Cremers, H. Bischof, P. Ochs, Y. Chen, T. Brox, R. Ranftl. M. Unger, M. Werlberger

Optimization methods in computer vision

- Typical energies in computer vision consist of a regularization term and a data term

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

where f is the input data and u is the unknown solution

Optimization methods in computer vision

- Typical energies in computer vision consist of a regularization term and a data term

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

where f is the input data and u is the unknown solution

- $\mathcal{R}(u)$: regularizer, prior, complexity term
- $\mathcal{D}(u, f)$: data model, fidelity, term, loss function

Optimization methods in computer vision

- Typical energies in computer vision consist of a regularization term and a data term

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

where f is the input data and u is the unknown solution

- $\mathcal{R}(u)$: regularizer, prior, complexity term
- $\mathcal{D}(u, f)$: data model, fidelity, term, loss function

- 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

Optimization in nature

- Intelligent systems perform optimization all the time

Optimization in nature

- Intelligent systems perform optimization all the time
- Many laws of nature are nothing but optimality conditions

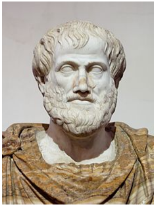
Optimization in nature

- Intelligent systems perform optimization all the time
- Many laws of nature are nothing but optimality conditions
- Examples of optimization in nature:

Minimal surfaces

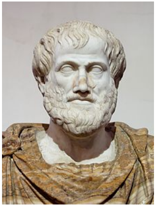
Heliostat field optimization

Regularization?



Aristotle: “Natura non facit saltus”

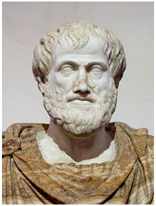
Regularization?



Aristotle: “Natura non facit saltus”

- Handcrafted models
  - piecewise constant/smooth functions
    [Rudin, Osher, Fatemi, '92], [Mumford, Shah, '89]
  - sparsity in some linear transform
    [Starck, Candes, Donoho, '02], [Candes, Romberg, Tao, '06]

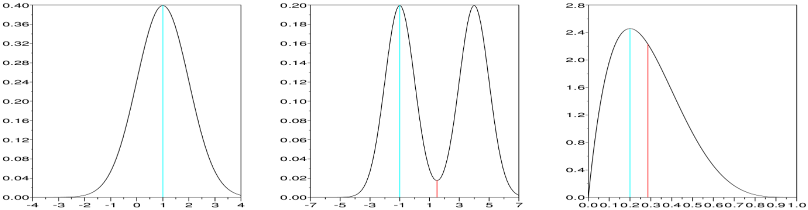
Regularization?



Aristotle: “Natura non facit saltus”

- Handcrafted models
  - piecewise constant/smooth functions
    [Rudin, Osher, Fatemi, ’92], [Mumford, Shah, ’89]
  - sparsity in some linear transform
    [Starck, Candes, Donoho, ’02], [Candes, Romberg, Tao, ’06]
- Learned models
  - Synthesis, Analysis based sparsity priors
    [Aharon, Elad, Bruckstein ’06], [Rubinstein, Faktor, Elad ’12]
  - MRF models
    [Roth, Black ’09], [Samuel, Tappen ’09]

Link to statistical approaches



- In a Bayesian setting, the energy relates to the posterior probability via a Gibbs distribution

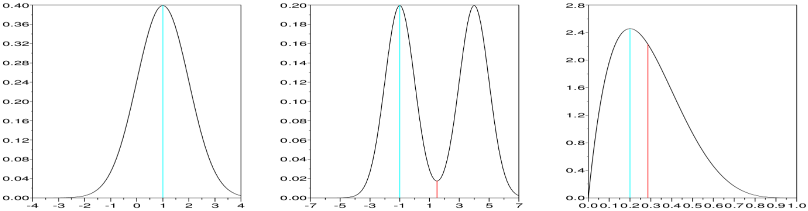
$$p(u|f) = \frac{1}{Z}\exp(-E(u))$$

- Expectation: Compute sample, that minimizes the squared distance to the distribution

$$\bar{u} = \frac{1}{Z}\int_u u\, p(u|f)\ \mathrm{d}u$$

- Needs subtle algorithms to approximate the integral (MCMC)

Link to statistical approaches



- In a Bayesian setting, the energy relates to the posterior probability via a Gibbs distribution

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

- MAP: Computing that sample that maximizes the probability

$$u^* = \text{arg}\ \max_u p(u|f) = \text{arg}\ \min_u E(u)$$

- Leads to well-defined optimization algorithms

Continous vs. discrete energy minimization methods

- Continuous variational approach:
  - Images are defined on continuous domains, rectangle, volume, manifold, e.g. $\Omega \subset \mathbb{R}^n$, the image is considered to be integer, or real-valued, e.g. $u : \Omega \to \mathbb{R}$

$$\min_{u:\Omega\to\mathbb{R}} \int_\Omega \mathrm{d}|\nabla u|_2 + \frac{\lambda}{2}\int_\Omega (u-f)^2 \, \mathrm{d}x$$

Continous vs. discrete energy minimization methods

- Continuous variational approach:
  - Images are defined on continuous domains, rectangle, volume, manifold, e.g. $\Omega \subset \mathbb{R}^n$, the image is considered to be integer, or real-valued, e.g. $u : \Omega \to \mathbb{R}$

$$\min_{u:\Omega\to\mathbb{R}} \int_\Omega \mathrm{d}|\nabla u|_2 + \frac{\lambda}{2}\int_\Omega (u-f)^2 \, \mathrm{d}x$$

- Discretized variational approach
  - Discretization of spatially continuous functions, images are elements of some finite dimensional vector space, e.g. $u \in \mathbb{R}^N$

$$\min_{u\in\mathbb{R}^N} \|\nabla u\|_{2,1} + \frac{\lambda}{2}\|u-f\|_2^2 \, , \quad \|\nabla u\|_{2,1} = \sum_{i=1}^N \sqrt{(\nabla_i^1 u)^2 + (\nabla_i^2 u)^2}$$

Continous vs. discrete energy minimization methods

- Continuous variational approach:
  - Images are defined on continuous domains, rectangle, volume, manifold, e.g. $\Omega \subset \mathbb{R}^n$, the image is considered to be integer, or real-valued, e.g. $u : \Omega \to \mathbb{R}$

$$\min_{u:\Omega\to\mathbb{R}} \int_\Omega \mathrm{d}|\nabla u|_2 + \frac{\lambda}{2}\int_\Omega (u-f)^2 \, \mathrm{d}x$$

- Discretized variational approach
  - Discretization of spatially continuous functions, images are elements of some finite dimensional vector space, e.g. $u \in \mathbb{R}^N$

$$\min_{u\in\mathbb{R}^N} \|\nabla u\|_{2,1} + \frac{\lambda}{2}\|u-f\|_2^2 \, , \quad \|\nabla u\|_{2,1} = \sum_{i=1}^{N} \sqrt{(\nabla_i^1 u)^2 + (\nabla_i^2 u)^2}$$

- 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 $i \in \mathcal{V}$ can take a label from a discrete label set $\mathcal{L} \subset \mathbb{Z}$, i.e. $u(i) \in \mathcal{L}$

$$\min_{u_i\in\{0,1,\ldots,255\}} \sum_{(i,j)\in\mathcal{E}} \theta_{ij}|u_i-u_j| + \frac{\lambda}{2}\sum_{i\in\mathcal{V}}(f_i-u_i)2$$

Discrete vs. continuous

(a) Clean image

(b) $\lambda = 1$

(c) MRF-8

(d) VM-simple

Convex versus non-convex

- From optimization, it is well known that the great watershed in optimization is between convexity and non-convexity [Rockafellar]

Convex versus non-convex

- From optimization, it is well known that the great watershed in optimization is between convexity and non-convexity [Rockafellar]
- In recent years, machine learning and computer vision has suffered from a convexivitis epidemic [LeCun]

Convex versus non-convex

- From optimization, it is well known that the great watershed in optimization is between convexity and non-convexity [Rockafellar]
- In recent years, machine learning and computer vision has suffered from a convexivitis epidemic [LeCun]
- Nowadays, convexity is considered to be a virtue for new models in machine learning and vision

Convex versus non-convex

- From optimization, it is well known that the great watershed in optimization is between convexity and non-convexity [Rockafellar]
- In recent years, machine learning and computer vision has suffered from a convexivitis epidemic [LeCun]
- Nowadays, convexity is considered to be a virtue for new models in machine learning and vision
- Sure, convexity is a very useful property, but it can also be a limitation

Convex versus non-convex

- From optimization, it is well known that the great watershed in optimization is between convexity and non-convexity [Rockafellar]
- In recent years, machine learning and computer vision has suffered from a convexivitis epidemic [LeCun]
- Nowadays, convexity is considered to be a virtue for new models in machine learning and vision
- Sure, convexity is a very useful property, but it can also be a limitation
- Most interesting problems are non-convex (optical flow, stereo, image restoration, segmentation, classification, ...)

Convex versus non-convex

- From optimization, it is well known that the great watershed in optimization is between convexity and non-convexity [Rockafellar]
- In recent years, machine learning and computer vision has suffered from a convexivitis epidemic [LeCun]
- Nowadays, convexity is considered to be a virtue for new models in machine learning and vision
- Sure, convexity is a very useful property, but it can also be a limitation
- Most interesting problems are non-convex (optical flow, stereo, image restoration, segmentation, classification, ...)
- To solve complicated tasks, we will not be able to avoid non-convexity

Convex versus non-convex

- From optimization, it is well known that the great watershed in optimization is between convexity and non-convexity [Rockafellar]
- In recent years, machine learning and computer vision has suffered from a convexivitis epidemic [LeCun]
- Nowadays, convexity is considered to be a virtue for new models in machine learning and vision
- Sure, convexity is a very useful property, but it can also be a limitation
- Most interesting problems are non-convex (optical flow, stereo, image restoration, segmentation, classification, ...)
- To solve complicated tasks, we will not be able to avoid non-convexity

Strategies for bridging the gap between convex and non-convex approaches

Strategies to solve non-convex problems

1. Work directly with the non-convex problem
    - Sometimes works well
    - Sometimes, does not work at all
    - Consider functions with a small degree of non-convexity

Strategies to solve non-convex problems

1. Work directly with the non-convex problem
   - Sometimes works well
   - Sometimes, does not work at all
   - Consider functions with a small degree of non-convexity

2. Local convexification of the problem
   - Majorization- and minimization of the problem
   - Linearization of the source of non-convexity
   - We can solve a sequence of convex problems
   - Can work very well, but often no guarantees

Strategies to solve non-convex problems

1. Work directly with the non-convex problem
   - Sometimes works well
   - Sometimes, does not work at all
   - Consider functions with a small degree of non-convexity

2. Local convexification of the problem
   - Majorization- and minimization of the problem
   - Linearization of the source of non-convexity
   - We can solve a sequence of convex problems
   - Can work very well, but often no guarantees

3. Minimize the (approximated) convex envelope
   - Compute the convex envelope of the problem
   - We can solve a single convex optimization problem
   - Often allows to give a-priori approximation guarantees
   - Restricted to relatively simple models

Overview

1 Introduction

2 Non-convex Optimization

3 Convex Optimization

4 Local Convexification

5 Convex Envelopes

6 Conclusion

Non-convex optimization problems

- Efficiently finding solutions to the whole class of Lipschitz continuous problems is a hopeless case [Nesterov '04]
- Can take several million years for small problems with only 10 unknowns

Non-convex optimization problems

- Efficiently finding solutions to the whole class of Lipschitz continuous problems is a hopeless case [Nesterov '04]
- Can take several million years for small problems with only 10 unknowns

- Smooth non-convex problems can be solved via generic nonlinear numerical optimization algorithms (SD, CG, BFGS, ...)
- Often hard to generalize to constraints, or non-differentiable functions
- Line-search procedure can be time intensive

Non-convex optimization problems

- Efficiently finding solutions to the whole class of Lipschitz continuous problems is a hopeless case [Nesterov '04]
- Can take several million years for small problems with only 10 unknowns

- Smooth non-convex problems can be solved via generic nonlinear numerical optimization algorithms (SD, CG, BFGS, ...)
- Often hard to generalize to constraints, or non-differentiable functions
- Line-search procedure can be time intensive

- A reasonable idea is to develop algorithms for special classes of structured non-convex problems
- A promising class of problems that has a moderate degree of non-convexity is given by the sum of a smooth non-convex function and a non-smooth convex function [Sra '12], [Chouzenoux, Pesquet, Repetti '13]

Smooth plus convex problems

- We consider the problem of minimizing a function $h \colon X \to \mathbb{R} \cup \{+\infty\}$

$$\min_{x \in X} \; h(x) = f(x) + g(x) \,,$$

  where X is a finite dimensional real vector space.
- We assume that h is coercive, i.e. $\|x\|_2 \to +\infty \quad \Rightarrow \quad h(x) \to +\infty$ and bounded from below by some value $\underline{h} > -\infty$

Smooth plus convex problems

- We consider the problem of minimizing a function $h\colon X \to \mathbb{R} \cup \{+\infty\}$

$$\min_{x \in X} \; h(x) = f(x) + g(x)\,,$$

  where X is a finite dimensional real vector space.
- We assume that h is coercive, i.e. $\|x\|_2 \to +\infty \quad \Rightarrow \quad h(x) \to +\infty$ and bounded from below by some value $\underline{h} > -\infty$
- The function $f \in C_L^{1,1}$ is possibly non-convex but has a Lipschitz continuous gradient, i.e.

$$\|\nabla f(x) - \nabla f(y)\|_2 \leq L\|x - y\|_2\,, \quad \forall x, y \in \operatorname{dom} f\,.$$

Smooth plus convex problems

- We consider the problem of minimizing a function $h\colon X \to \mathbb{R} \cup \{+\infty\}$

$$\min_{x \in X} \; h(x) = f(x) + g(x)\,,$$

where X is a finite dimensional real vector space.
- We assume that h is coercive, i.e. $\|x\|_2 \to +\infty \quad \Rightarrow \quad h(x) \to +\infty$ and bounded from below by some value $\underline{h} > -\infty$
- The function $f \in C_L^{1,1}$ is possibly non-convex but has a Lipschitz continuous gradient, i.e.

$$\|\nabla f(x) - \nabla f(y)\|_2 \leq L\|x - y\|_2\,, \quad \forall x, y \in \operatorname{dom} f\,.$$

- The function g is a proper lower semi-continuous convex function with an efficient to compute proximal map

$$(I + \alpha\partial g)^{-1}(\hat{x}) := \text{arg}\min_{x \in X} \frac{\|x - \hat{x}\|_2^2}{2} + \alpha g(x)\,,$$

where $\alpha > 0$.

Forward-backward splitting

- We aim at seeking a critical point x^*, i.e. a point satisfying $0 \in \partial h(x^*)$ which in our case becomes

$$-\nabla f(x^*) \in \partial g(x^*) \,.$$

- A critical point can also be characterized via the *proximal residual*

$$r(x) := x - (I + \partial g)^{-1}(x - \nabla f(x)) \,,$$

where I is the identity map.
- Clearly $r(x^*) = 0$ implies that x^* is a critical point.
- The norm of the proximal residual can be used as a (bad) measure of optimality

Forward-backward splitting

- We aim at seeking a critical point x^*, i.e. a point satisfying $0 \in \partial h(x^*)$ which in our case becomes

$$-\nabla f(x^*) \in \partial g(x^*) \,.$$

- A critical point can also be characterized via the *proximal residual*

$$r(x) := x - (I + \partial g)^{-1}(x - \nabla f(x)) \,,$$

  where I is the identity map.
- Clearly $r(x^*) = 0$ implies that x^* is a critical point.
- The norm of the proximal residual can be used as a (bad) measure of optimality
- The proximal residual already suggests an iterative method of the form

$$x^{n+1} = (I + \partial g)^{-1}(x^n - \nabla f(x^n))$$

- For f convex, this algorithm is well studied [Lions, Mercier '79], [Tseng '91], [Daubechie et al. '04], [Combettes, Wajs '05], [Raguet, Fadili, Peyré '13]

Inertial methods

- Introduced by Polyak in [Polyak '64] as a special case of multi-step algorithms for minimizing a function $f \in \mathcal{S}^{1,1}_{\mu,L}$

$$x^{n+1} = x^n - \alpha \nabla f(x^n) + \beta(x^n - x^{n-1})$$

- Optimal convergence rate on strongly convex problems
- Close relations to the conjugate gradient method
- Can be seen as a discrete variant of the heavy-ball with friction dynamic system
- Hence, the inertial term acts as an acceleration term
- Can help to avoid suprious critical points
- We propose a generalization to minimize the sum of a smooth and a convex function

iPiano (inertial Proximal algoriothm for non-convex optimization)

For minimizing the sum of a smooth and a convex function, we propose the following algorithm:

- Initialization: Choose $c_1, c_2 > 0$, $x^0 \in \operatorname{dom} h$ and set $x^{-1} = x^0$.
- Iterations $(n \geq 0)$: Update

$$x^{n+1} = (I + \alpha_n \partial g)^{-1}(x^n - \alpha_n \nabla f(x^n) + \beta_n (x^n - x^{n-1})) ,$$

where $L_n > 0$ is the local Lipschitz constant satisfying

$$f(x^{n+1}) \leq f(x^n) + \left\langle \nabla f(x^n), x^{n+1} - x^n \right\rangle + \frac{L_n}{2} \|x^{n+1} - x^n\|_2^2 ,$$

and $\alpha_n \geq c_1$, $\beta_n \geq 0$ are chosen such that $\delta_n \geq \gamma_n \geq c_2$ defined by

$$\delta_n := \frac{1}{\alpha_n} - \frac{L_n}{2} - \frac{\beta_n}{2\alpha_n} \quad \text{and} \quad \gamma_n := \frac{1}{\alpha_n} - \frac{L_n}{2} - \frac{\beta_n}{\alpha_n} .$$

and $(\delta_n)_{n=0}^{\infty}$ is monotonically decreasing.

Convergence Analysis

We can give the following convergence result:

Theorem

(a) *The sequence* $(h(x^n))_{n=0}^{\infty}$ *converges.*

(b) *There exists a converging subsequence* $(x^{n_k})_{k=0}^{\infty}$.

(c) *Any limit point* $x^* := \lim_{k\to\infty} x^{n_k}$ *is a critical point of* h.

- Convergence of the whole sequence can be obtained by assuming that the so-called Kurdyka-Łojasiewicz property holds, wich is true for most reasonable functions

Convergence rate in the non-convex case

- Absence of convexity makes live hard

Convergence rate in the non-convex case

- Absence of convexity makes live hard
- We can merely establish the following very weak convergence rate

Convergence rate in the non-convex case

- Absence of convexity makes live hard
- We can merely establish the following very weak convergence rate

Theorem

The iPiano algorithm guarantees that for all $N \geq 0$

$$\min_{0 \leq n \leq N} \|r(x^n)\|_2 \leq \frac{2}{c_1 c_2} \sqrt{\frac{h(x^0) - \underline{h}}{N+1}}$$

i.e. the smallest proximal residual converges with rate $\mathcal{O}(1/\sqrt{N})$.

- Similar bound for $\beta = 0$ is shown in [Nesterov '12]

Application to image compression based on linear diffusion

- A new image compression methodology introduced in [Galic, Weickert, Welk, Bruhn, Belyaev, Seidel '08]
- The idea is to select a subset of image pixels such that the reconstruction of the whole image via linear diffusion yields the best reconstruction [Hoeltgen, Setzer, Weickert '13]
- Is written as the following bilevel optimization problem

$$\min_{u,c} \frac{1}{2}\|u - u^0\|_2^2 + \lambda\|c\|_1$$
$$\text{s.t. } C(u - u^0) - (I - C)Lu = 0\,,$$

  where $C = \mathrm{diag}(c) \in \mathbb{R}^{N \times N}$ and L is the Laplace operator
- We can transform the problem into an non-convex single-level problem of the form

$$\min_{c} \frac{1}{2}\|A^{-1}Cu^0 - u^0\|_2^2 + \lambda\|c\|_1\,, \quad A = C + (C - I)L$$

- Perfectly fits to the framework of iPiano
- We choose $f = \frac{1}{2}\|A^{-1}Cu^0 - u^0\|_2^2$ and $g = \lambda\|c\|_1$
- The gradient of f is given by

$$\nabla f(c) = \mathrm{diag}(-(I+L)u + u^0)(A^\top)^{-1}(u - u^0)\,, \quad u = A^{-1}Cu^0$$

- Lipschitz, if at least one entry of c is non-zero
- One evaluation of the gradient requires to solve two linear systems
- Proximal map with respect to g is standard

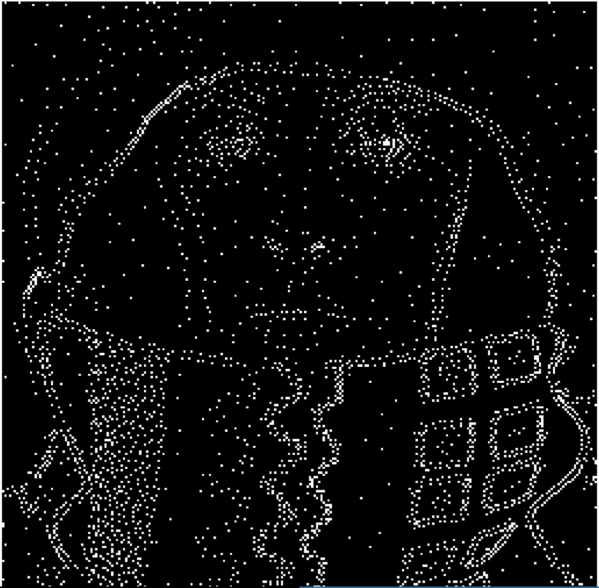
Results

Comparison with the successive primal-dual (SPD) algorithm proposed in [Hoeltgen, Setzer, Weickert '13]

Test image	Algorithm	Iterations	Energy	Density	MSE
Trui	iPiano	1000	21.574011	4.98%	17.31
	SPD	200/4000	21.630280	5.08%	17.06
Peppers	iPiano	1000	20.631985	4.84%	19.50
	SPD	200/4000	20.758777	4.93%	19.48
Walter	iPiano	1000	10.246041	4.82%	8.29
	SPD	200/4000	10.278874	4.93%	8.01

Results for Trui

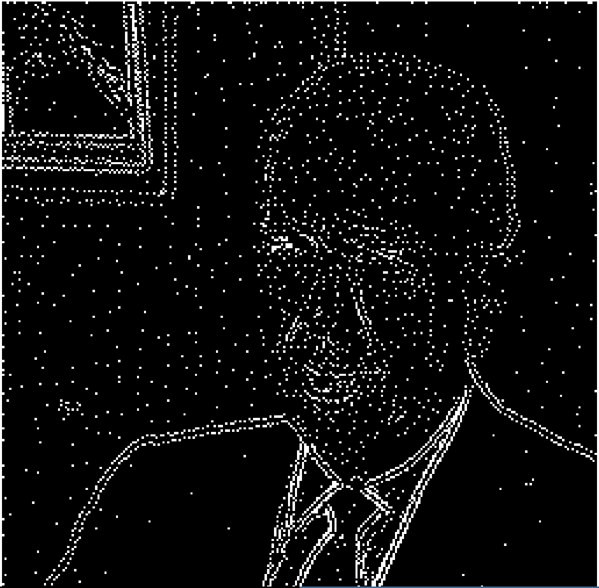
Results for Trui



Results for Trui

Results for Walter

Results for Walter



Results for Walter

Overview

1. Introduction
2. Non-convex Optimization
3. Convex Optimization
4. Local Convexification
5. Convex Envelopes
6. Conclusion

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 and F, G are convex, (non-smooth) proper, l.s.c. functions.

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 and F, G are convex, (non-smooth) proper, l.s.c. functions.
- **Main assumption:** F, G are "simple" in the sense that they have easy to compute resolvent operators:

$$(I + \partial F)^{-1}(\hat{p}) = \text{arg} \min_{p} \frac{\|p - \hat{p}\|^2}{2\lambda} + F(p)$$

$$(I + \partial G)^{-1}(\hat{x}) = \text{arg} \min_{x} \frac{\|x - \hat{x}\|^2}{2\lambda} + G(x)$$

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 and F, G are convex, (non-smooth) proper, l.s.c. functions.
- **Main assumption:** F, G are "simple" in the sense that they have easy to compute resolvent operators:

$$(I + \partial F)^{-1}(\hat{p}) = \text{arg} \min_p \frac{\|p - \hat{p}\|^2}{2\lambda} + F(p)$$

$$(I + \partial G)^{-1}(\hat{x}) = \text{arg} \min_x \frac{\|x - \hat{x}\|^2}{2\lambda} + G(x)$$

- It turns out that many standard problems can be cast in this framework.

Some examples

- The ROF model

$$\min_u \|\nabla u\|_{2,1} + \frac{\lambda}{2}\|u - f\|_2^2 \ ,$$

Some examples

- The ROF model

$$\min_u \|\nabla u\|_{2,1} + \frac{\lambda}{2}\|u - f\|_2^2 \ ,$$

- Basis pursuit problem (LASSO)

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

Some examples

- The ROF model

$$\min_u \|\nabla u\|_{2,1} + \frac{\lambda}{2}\|u - f\|_2^2 \ ,$$

- Basis pursuit problem (LASSO)

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

- Linear support vector machine

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

Some examples

- The ROF model

$$\min_u \|\nabla u\|_{2,1} + \frac{\lambda}{2}\|u - f\|_2^2 \ ,$$

- Basis pursuit problem (LASSO)

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

- 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

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

Primal, dual, primal-dual

The real power of convex optimization comes through duality

Recall the convex conjugate:

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

we can transform our initial problem

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

Primal, dual, primal-dual

The real power of convex optimization comes through duality

Recall the convex conjugate:

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

we can transform our initial problem

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

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

Primal, dual, primal-dual

The real power of convex optimization comes through duality

Recall the convex conjugate:

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

we can transform our initial problem

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

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

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

Primal, dual, primal-dual

The real power of convex optimization comes through duality

Recall the convex conjugate:

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

we can transform our initial problem

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

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

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

There is a primal-dual gap:

$$\mathcal{G}(x, y) = F(Kx) + G(x) + \left(F^*(y) + G^*(-K^*y)\right)$$

that vanishes if and only if (x, y) is optimal

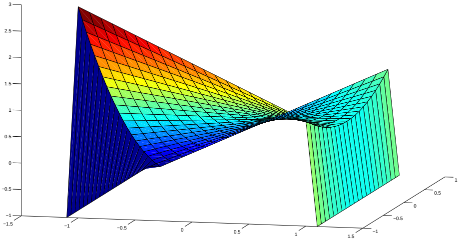
Optimality conditions

We focus on the primal-dual saddle-point formulation:

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

The optimal solution is a saddle-point $(\hat{x}, \hat{y}) \in X \times Y$ which satisfies the Euler-Lagrange equations

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



$|x| + |x - f|^2/2$

How can we find a saddle-point $(\hat{x}, \hat{y})$?

A first-order primal-dual algorithm

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

- Initialization: Choose $\mathrm{T}, \Sigma \in S_{++}$, $\theta \in [0,1]$, $(x^0, y^0) \in X \times Y$.

A first-order primal-dual algorithm

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

- Initialization: Choose $\mathrm{T}, \Sigma \in S_{++}$, $\theta \in [0,1]$, $(x^0, y^0) \in X \times Y$.
- Iterations $(n \geq 0)$: Update x^n, y^n as follows:

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

A first-order primal-dual algorithm

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

- Initialization: Choose $\mathrm{T}, \Sigma \in S_{++}$, $\theta \in [0,1]$, $(x^0, y^0) \in X \times Y$.
- Iterations $(n \geq 0)$: Update x^n, y^n as follows:

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

- T, Σ are preconditioning matrices

A first-order primal-dual algorithm

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

- Initialization: Choose $\mathrm{T}, \Sigma \in S_{++}$, $\theta \in [0,1]$, $(x^0, y^0) \in X \times Y$.
- Iterations $(n \geq 0)$: Update x^n, y^n as follows:

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

- T, Σ are preconditioning matrices
- Alternates gradient descend in x and gradient ascend in y

A first-order primal-dual algorithm

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

- Initialization: Choose $\mathrm{T}, \Sigma \in S_{++}$, $\theta \in [0,1]$, $(x^0, y^0) \in X \times Y$.
- Iterations $(n \geq 0)$: Update x^n, y^n as follows:

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

- T, Σ are preconditioning matrices
- Alternates gradient descend in x and gradient ascend in y
- Linear extrapolation of iterates of x in the y step

Convergence

Theorem

Let $\theta = 1$, T *and* Σ *symmetric positive definite maps satisfying*

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

then the primal-dual algorithm converges to a saddle-point.

Convergence

Theorem

Let $\theta = 1$, T and Σ symmetric positive definite maps satisfying

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

then the primal-dual algorithm converges to a saddle-point.

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

- F^* and G non-smooth: $O(1/n)$

Convergence

Theorem

Let $\theta = 1$, T and Σ symmetric positive definite maps satisfying

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

then the primal-dual algorithm converges to a saddle-point.

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

- F^* and G non-smooth: $O(1/n)$
- F^* or G uniformly convex: $O(1/n^2)$

Convergence

Theorem

Let $\theta = 1$, T *and* Σ *symmetric positive definite maps satisfying*

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

then the primal-dual algorithm converges to a saddle-point.

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

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

Convergence

Theorem

Let $\theta = 1$, T and Σ symmetric positive definite maps satisfying

$$\|\Sigma^{\frac{1}{2}} K \mathrm{T}^{\frac{1}{2}}\|^2 < 1 \; ,$$

then the primal-dual algorithm converges to a saddle-point.

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

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

- Coincide with lower complexity bounds for first-order methods [Nesterov, '04]

α-preconditioning

- It is important to choose the preconditioner such that the prox-operators are still easy to compute
- Restrict the preconditioning matrices to diagonal matrices

α-preconditioning

- It is important to choose the preconditioner such that the prox-operators are still easy to compute
- Restrict the preconditioning matrices to diagonal matrices

Lemma

Let $\mathrm{T} = \mathrm{diag}(\tau_1, \ldots \tau_n)$ *and* $\Sigma = \mathrm{diag}(\sigma_1, \ldots, \sigma_m)$.

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

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

$$\|\Sigma^{\frac{1}{2}} K \mathrm{T}^{\frac{1}{2}}\|^2 = \sup_{x \in X,\, x \neq 0} \frac{\|\Sigma^{\frac{1}{2}} K \mathrm{T}^{\frac{1}{2}} x\|^2}{\|x\|^2} \leq 1 \,.$$

[P., Chambolle, '11]

α-preconditioning

- It is important to choose the preconditioner such that the prox-operators are still easy to compute
- Restrict the preconditioning matrices to diagonal matrices

Lemma

Let $\mathrm{T} = \operatorname{diag}(\tau_1, ...\tau_n)$ *and* $\Sigma = \operatorname{diag}(\sigma_1, ..., \sigma_m)$.

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

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

$$\|\Sigma^{\frac{1}{2}} K \mathrm{T}^{\frac{1}{2}}\|^2 = \sup_{x \in X,\, x \neq 0} \frac{\|\Sigma^{\frac{1}{2}} K \mathrm{T}^{\frac{1}{2}} x\|^2}{\|x\|^2} \leq 1 \ .$$

[P., Chambolle, '11]

- The parameter α can be used to vary between pure primal ($\alpha = 0$) and pure dual ($\alpha = 2$) preconditioning

Parallel computing?

- The algorithm basically computes matrix-vector products

Parallel computing?

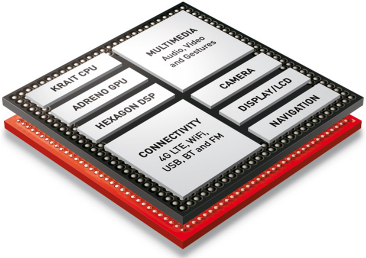
- The algorithm basically computes matrix-vector products
- The matrices are usually very sparse

Parallel computing?

- The algorithm basically computes matrix-vector products
- The matrices are usually very sparse
- Well suited for highly parallel architectures

Parallel computing?

- The algorithm basically computes matrix-vector products
- The matrices are usually very sparse
- Well suited for highly parallel architectures
- Gives high speedup factors (∼30-50)



Overview

1 Introduction

2 Non-convex Optimization

3 Convex Optimization

4 Local Convexification

5 Convex Envelopes

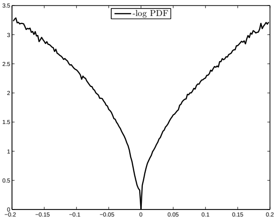
6 Conclusion

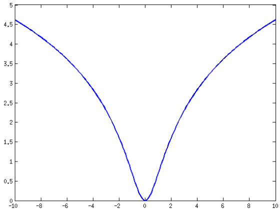
Local Convexification

- The local convexification uses the structure of the problem
- Identify the source of non-convexity
- Locally approximate the non-convex function by a convex one
- Solve the resulting non-convex problem and repeat the convexification

Non-convex potential functions

- The choice of the potential function in image restoration is motivated by the statistics of natural images
- Let us record a histogram of the filter-response of a DTC5 filter on natural images [Huang and Mumford '99]





- A good fit is obtanied for the family of non-convex functions $\log(1 + x^2)$

Application to non-convex image denoising

- Approximately minimize a non-convex energy based on Student-t potential functions

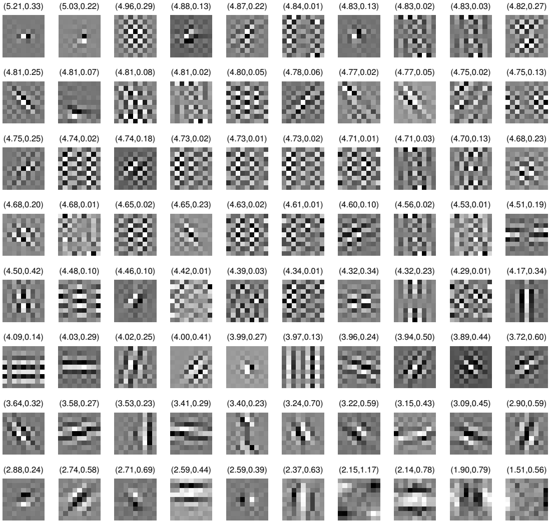
$$\min_x \sum_i \alpha_i \sum_p \log(1 + |(K_i x)_p|^2) + \frac{1}{2}\|x - f\|_2^2 ,$$

- The application of the linear operators K_i are realized via convolution with filters k_i

$$K_i x \quad \Leftrightarrow \quad k_i * x$$

- Parameters α_i and filters k_i are learned using bilevel optimization [Chen et al. '13]

The filters

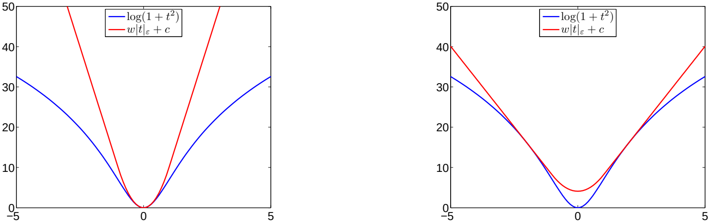

(5.21,0.33) (5.03,0.22) (4.96,0.29) (4.88,0.13) (4.87,0.22) (4.84,0.01) (4.83,0.13) (4.83,0.02) (4.83,0.03) (4.82,0.27)
(4.81,0.25) (4.81,0.07) (4.81,0.08) (4.81,0.02) (4.80,0.05) (4.78,0.06) (4.77,0.02) (4.77,0.05) (4.75,0.02) (4.75,0.13)
(4.75,0.25) (4.74,0.02) (4.74,0.18) (4.73,0.02) (4.73,0.01) (4.73,0.02) (4.71,0.01) (4.71,0.03) (4.70,0.13) (4.68,0.23)
(4.68,0.20) (4.68,0.01) (4.65,0.02) (4.65,0.23) (4.63,0.02) (4.61,0.01) (4.60,0.10) (4.56,0.02) (4.53,0.01) (4.51,0.19)
(4.50,0.42) (4.48,0.10) (4.46,0.10) (4.42,0.01) (4.39,0.03) (4.34,0.01) (4.32,0.34) (4.32,0.23) (4.29,0.01) (4.17,0.34)
(4.09,0.14) (4.03,0.29) (4.02,0.25) (4.00,0.41) (3.99,0.27) (3.97,0.13) (3.96,0.24) (3.94,0.50) (3.89,0.44) (3.72,0.60)
(3.64,0.32) (3.58,0.27) (3.53,0.23) (3.41,0.29) (3.40,0.23) (3.24,0.70) (3.22,0.59) (3.15,0.43) (3.09,0.45) (2.90,0.59)
(2.88,0.24) (2.74,0.58) (2.71,0.69) (2.59,0.44) (2.59,0.39) (2.37,0.63) (2.15,1.17) (2.14,0.78) (1.90,0.79) (1.51,0.56)

Iterated Huber for Student-t

- Majorize-Minimize strategy:
- Minimize a sequence a of convex weighted Huber-ℓ_1 problems

$$x^{n+1} = \text{arg}\min_x \sum_i \alpha_i \sum_p w_i(x^n)_p |(K_i x)_p|_\varepsilon + \frac{1}{2}\|x - f\|_2^2$$

where $w_i(x^n) = 2\frac{\max\{\varepsilon, |K_i x^n|\}}{1+|K_i x^n|^2}$ and $|\cdot|_\varepsilon$ denotes the Huber function

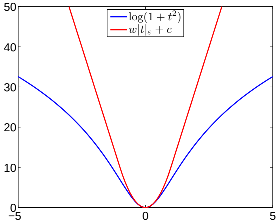


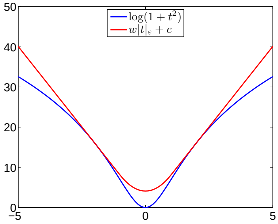
Iterated Huber for Student-t

- Majorize-Minimize strategy:
- Minimize a sequence a of convex weighted Huber-ℓ_1 problems

$$x^{n+1} = \text{arg}\min_x \sum_i \alpha_i \sum_p w_i(x^n)_p |(K_i x)_p|_\varepsilon + \frac{1}{2}\|x - f\|_2^2$$

where $w_i(x^n) = 2\frac{\max\{\varepsilon,|K_i x^n|\}}{1+|K_i x^n|^2}$ and $|\cdot|_\varepsilon$ denotes the Huber function





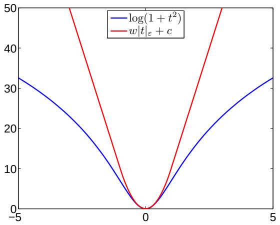
- Best fit for $\varepsilon = 1$

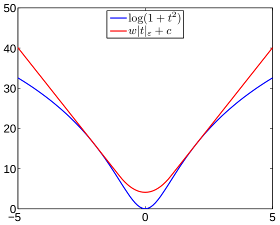
Iterated Huber for Student-t

- Majorize-Minimize strategy:
- Minimize a sequence a of convex weighted Huber-ℓ_1 problems

$$x^{n+1} = \text{arg}\min_x \sum_i \alpha_i \sum_p w_i(x^n)_p |(K_i x)_p|_\varepsilon + \frac{1}{2}\|x-f\|_2^2$$

where $w_i(x^n) = 2\frac{\max\{\varepsilon, |K_i x^n|\}}{1+|K_i x^n|^2}$ and $|\cdot|_\varepsilon$ denotes the Huber function





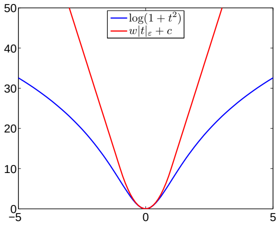
- Best fit for $\varepsilon = 1$
- The primal-dual algorithm has a linear convergence rates on the convex sub-problems

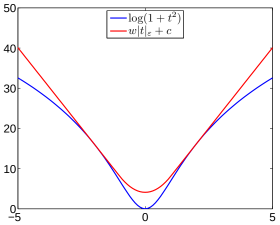
Iterated Huber for Student-t

- Majorize-Minimize strategy:
- Minimize a sequence a of convex weighted Huber-ℓ_1 problems

$$x^{n+1} = \text{arg}\min_x \sum_i \alpha_i \sum_p w_i(x^n)_p |(K_i x)_p|_\varepsilon + \frac{1}{2}\|x - f\|_2^2$$

where $w_i(x^n) = 2\frac{\max\{\varepsilon, |K_i x^n|\}}{1+|K_i x^n|^2}$ and $|\cdot|_\varepsilon$ denotes the Huber function





- Best fit for $\varepsilon = 1$
- The primal-dual algorithm has a linear convergence rates on the convex sub-problems

Example

Example

Example

Evaluation

- Comparison with five state-of-the-art approaches: K-SVD [Elad and Aharon '06], FoE [Q. Gao and Roth '12], BM3D [Dabov et al. '07], GMM [D. Zoran et al. '12], LSSC [Mairal et al. '09]
- We report the average PSNR on 68 images of the Berkeley image data base

σ	KSVD	FoE	BM3D	GMM	LSSC	ours
15	30.87	30.99	31.08	31.19	**31.27**	31.22
25	28.28	28.40	28.56	28.68	**28.70**	**28.70**
50	25.17	25.35	25.62	25.67	25.72	**25.76**

- Performs as well as state-of-the-art
- A GPU implementation is significantly faster
- Can be used as a prior for general inverse problems

Optical flow

- Optical Flow is a central topic in computer vision [Horn, Schunck, 1981], [Shulman, Hervé '89], [Bruhn, Weickert, Schnörr '02], [Brox, Bruhn, Papenberg, Weickert '04], [Zach, P., Bischof, DAGM'07] ...
- Computes a vector field, describing the aparent motion of pixel intensities
- Numerous applications

- TV-L^1 optical flow

$$\min_u \|\nabla u\|_{2,1} + \lambda \|I_2(x+u) - I_1(x)\|_1$$

- The source of non-convexity lies in the expression $I_2(x+u)$

Optical flow

- Optical Flow is a central topic in computer vision [Horn, Schunck, 1981], [Shulman, Hervé '89], [Bruhn, Weickert, Schnörr '02], [Brox, Bruhn, Papenberg, Weickert '04], [Zach, P., Bischof, DAGM'07] ...
- Computes a vector field, describing the apparent motion of pixel intensities
- Numerous applications

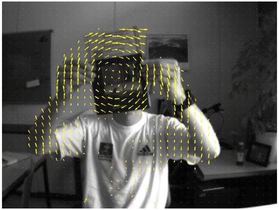
- TV-L^1 optical flow

$$\min_u \|\nabla u\|_{2,1} + \lambda \|I_2(x+u) - I_1(x)\|_1$$

- The source of non-convexity lies in the expression $I_2(x+u)$

Optical flow

- Optical Flow is a central topic in computer vision [Horn, Schunck, 1981], [Shulman, Hervé '89], [Bruhn, Weickert, Schnörr '02], [Brox, Bruhn, Papenberg, Weickert '04], [Zach, P., Bischof, DAGM'07] ...
- Computes a vector field, describing the aparent motion of pixel intensities
- Numerous applications



- TV-L^1 optical flow

$$\min_u \|\nabla u\|_{2,1} + \lambda \|I_2(x+u) - I_1(x)\|_1$$

- The source of non-convexity lies in the expression $I_2(x+u)$

Optical flow

- Optical Flow is a central topic in computer vision [Horn, Schunck, 1981], [Shulman, Hervé '89], [Bruhn, Weickert, Schnörr '02], [Brox, Bruhn, Papenberg, Weickert '04], [Zach, P., Bischof, DAGM'07] ...
- Computes a vector field, describing the aparent motion of pixel intensities
- Numerous applications

- TV-L^1 optical flow

$$\min_u \|\nabla u\|_{2,1} + \lambda \|I_2(x+u) - I_1(x)\|_1$$

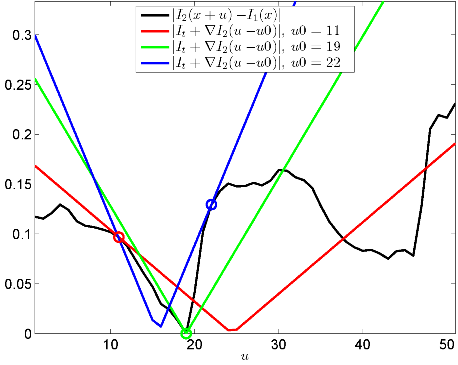
- The source of non-convexity lies in the expression $I_2(x+u)$

Optical Flow

- Convexification via linearization:

$$\|I_2(x+u) - I_1(x)\|_1 \approx \|I_t + \nabla I_2(u - u_0)\|_1$$

- Only valid in a small neighborhood around u_0



- Minimized via the primal-dual algorithm

Real-time implementation

- Due to the strong non-convexity, the algorithm has to be integrated into a coarse-to-fine / warping framework
- Works well in case of small displacements but can fail in case of large displacements [Brox, Bregler, Malik '09]
- GPU-implementation yields real-time performance (> 20 fps) for 854×480 images using a recent Nvidia graphics card [Zach, P., Bischof, '07] [Werlberger, P., Bischof, '10]
- GLSL shader implementation on a mobile GPU (Adreno 330 in Nexus 5) implementation currently yields 10 fps on 320×240 images (implemented by Christoph Bauernhofer).
- The performance is expected to increase in near future.

Resolution: 854×480

Generalization to features

- The two images $I_{1,2}$ can be easily replaced by their corresponding feature transforms, e.g. SIFT descriptors
- Optical flow algorithm can be used for wide-baseline matching

Generalization to features

Generalization to features

Generalization to features

Overview

1 Introduction

2 Non-convex Optimization

3 Convex Optimization

4 Local Convexification

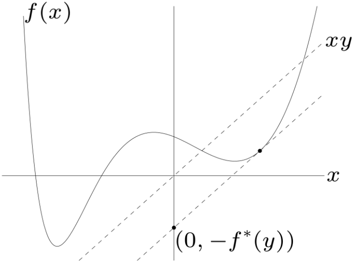
5 Convex Envelopes

6 Conclusion

The convex conjugate

- The convex conjugate $f^*(y)$ of a function $f(x)$ is defined through the Legendre-Fenchel transform

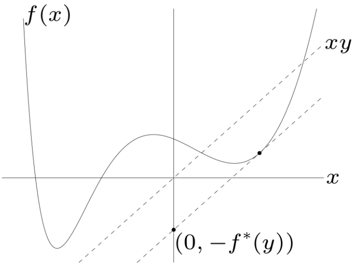
$$f^*(y) = \sup_{x \in \mathrm{dom}\ f} \langle x, y \rangle - f(x)$$



The convex conjugate

- The convex conjugate $f^*(y)$ of a function $f(x)$ is defined through the Legendre-Fenchel transform

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



- $f^*(y)$ is a convex function (pointwise supremum over linear functions)

The convex envelope

- The biconjugate function is defined by twice application of the Legendere-Fenchel transform

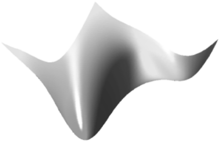
$$f^{**}(x) = \sup_{y \in \mathrm{dom}\ f^*} \langle x, y \rangle - f^*(y)$$

The convex envelope

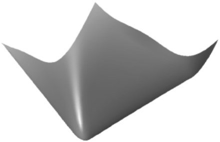
- The biconjugate function is defined by twice application of the Legendere-Fenchel transform

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

- $f^{**}(x)$ is the largest convex l.s.c. function below $f(x)$ minimum
- If $f(x)$ is a convex, l.s.c. function, $f^{**}(x) = f(x)$



$f(x)$



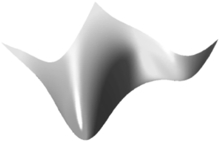
$f^{**}(x)$

The convex envelope

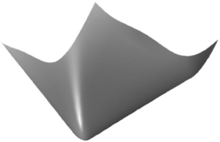
- The biconjugate function is defined by twice application of the Legendere-Fenchel transform

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

- $f^{**}(x)$ is the largest convex l.s.c. function below $f(x)$ minimum
- If $f(x)$ is a convex, l.s.c. function, $f^{**}(x) = f(x)$



$f(x)$



$f^{**}(x)$

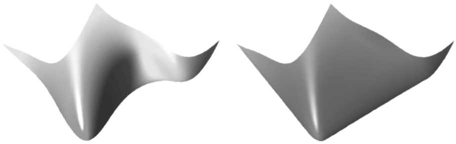
- Allows to "convexify" a non-convex problem
- Unfortunately, computing $f^{**}(x)$ is not tractable for most problems

The convex envelope

- The biconjugate function is defined by twice application of the Legendere-Fenchel transform

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

- $f^{**}(x)$ is the largest convex l.s.c. function below $f(x)$ minimum
- If $f(x)$ is a convex, l.s.c. function, $f^{**}(x) = f(x)$



$f(x)$ $f^{**}(x)$

- Allows to "convexify" a non-convex problem
- Unfortunately, computing $f^{**}(x)$ is not tractable for most problems
- **The key: Looking for tractable approximations of the convex envelope**

Global solutions of non-convex variational models

- Consider the following non-convex energy-functional

$$\min_u \int_\Omega f(x, u(x), \nabla u(x)) \, \mathrm{d}x$$

- We assume that $f(x, t, p)$ is convex in p but non-convex in t

Global solutions of non-convex variational models

- Consider the following non-convex energy-functional

$$\min_u \int_\Omega f(x, u(x), \nabla u(x)) \, \mathrm{d}x$$

- We assume that $f(x, t, p)$ is convex in p but non-convex in t
- Example: TV-ℓ_1 stereo

$$f(x, u(x), \nabla u(x)) = \alpha|\nabla u| + |I_1(x) - I_2(x + u(x))|$$

- How can we confexify this problem?

Global solutions of non-convex variational models

- Consider the following non-convex energy-functional

$$\min_u \int_\Omega f(x, u(x), \nabla u(x)) \, \mathrm{d}x$$

- We assume that $f(x, t, p)$ is convex in p but non-convex in t
- Example: TV-ℓ_1 stereo

$$f(x, u(x), \nabla u(x)) = \alpha |\nabla u| + |I_1(x) - I_2(x + u(x))|$$

- How can we confexify this problem?
- In a discrete MRF setting, a solution has been proposed by [Ishikawa, '03] by a graph cut on a higher-dimensional graph

Global solutions of non-convex variational models

- Consider the following non-convex energy-functional

$$\min_u \int_\Omega f(x, u(x), \nabla u(x)) \, \mathrm{d}x$$

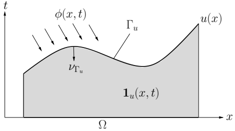
- We assume that $f(x, t, p)$ is convex in p but non-convex in t
- Example: TV-ℓ_1 stereo

$$f(x, u(x), \nabla u(x)) = \alpha |\nabla u| + |I_1(x) - I_2(x + u(x))|$$

- How can we confexify this problem?
- In a discrete MRF setting, a solution has been proposed by [Ishikawa, '03] by a graph cut on a higher-dimensional graph
- What about the continuous setting?
- [P., Cremers, Bischof, Chambolle, SIIMS'10]

The approach of Alberti, Bouchitte and Dal Maso

- The calibration method of [Alberti, Bouchitte, Dal Maso, '03]
- The basic idea is to consider the graph Γ_u of u instead of the function u
- Rewrite $E(u)$ by means of the flux of vector field ϕ through the graph Γ_u



- The characteristic function $\mathbf{1}_u$ of the subgraph of a function $u \in \mathcal{BV}(\Omega \times \mathbb{R}, [0,1])$ is defined as

$$\mathbf{1}_u(x,t) = \begin{cases} 1, & \text{if } t < u(x), \\ 0, & \text{else.} \end{cases}$$

- The normal ν_{Γ_u} of the interface Γ_u is given by

$$\nu_{\Gamma_u} = \frac{(\nabla u, -1)}{\sqrt{|\nabla u|^2 + 1}}$$

A lower bound

- Suppose, the maximum flux of a vector field $\phi = (\phi^x, \phi^t)$ through the graph provides a lower bound to $E(u)$

$$E(u) \geq \sup_{\phi \in \mathcal{K}} \int_{\Gamma_u} \phi \cdot \nu_{\Gamma_u} \, \mathrm{d}\mathcal{H}^2.$$

A lower bound

- Suppose, the maximum flux of a vector field $\phi = (\phi^x, \phi^t)$ through the graph provides a lower bound to $E(u)$

$$E(u) \geq \sup_{\phi \in \mathcal{K}} \int_{\Gamma_u} \phi \cdot \nu_{\Gamma_u} \, \mathrm{d}\mathcal{H}^2.$$

- It turns out that equality holds for

$$\mathcal{K} = \left\{ \phi = (\phi^x, \phi^t) \;\middle|\; \phi^t(x,t) \geq f^*(x, t, \phi^x(x,t)) \right\}$$

A lower bound

- Suppose, the maximum flux of a vector field $\phi = (\phi^x, \phi^t)$ through the graph provides a lower bound to $E(u)$

$$E(u) \geq \sup_{\phi \in \mathcal{K}} \int_{\Gamma_u} \phi \cdot \nu_{\Gamma_u} \, \mathrm{d}\mathcal{H}^2.$$

- It turns out that equality holds for

$$\mathcal{K} = \left\{ \phi = (\phi^x, \phi^t) \;\middle|\; \phi^t(x,t) \geq f^*(x,t,\phi^x(x,t)) \right\}$$

- The integral can be extended to $\Omega \times \mathbb{R}$

$$E(u) = \sup_{\phi \in K} \int_{\Omega \times \mathbb{R}} \phi \cdot D\mathbf{1}_u,$$

- Relaxation of the binary constraint and solution via the primal-dual algorithm

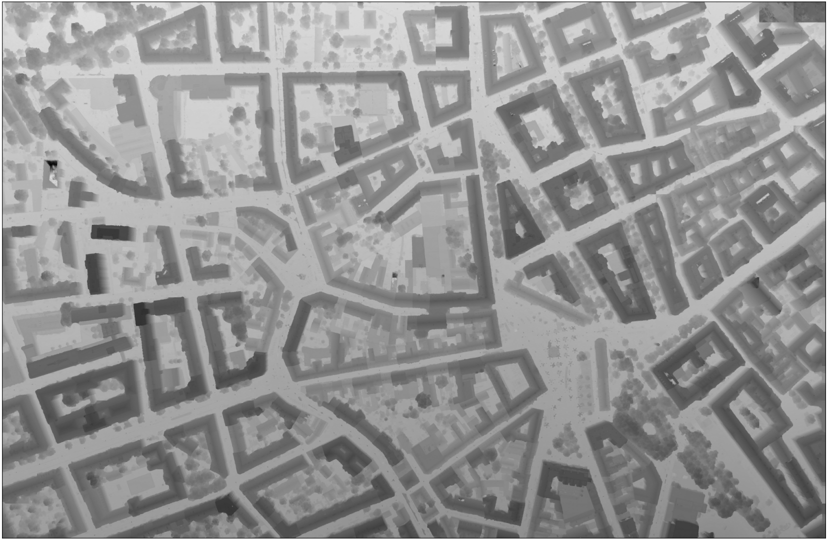
Digital surface model of Graz

Input

Digital surface model of Graz

Data term only

Digital surface model of Graz



Convex variational approach

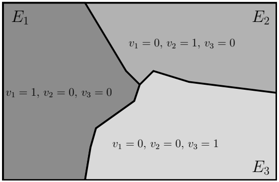
Digital surface model of Graz

Minimal partitions

- The "continuous" Potts model
- Minimizes the total interface length (area) of the partitioning subject to some given external fields f_i
- NP-hard for $k > 2$
- We propose the following convex relaxation

$$\min_{\mathbf{v}} \mathcal{J}(\mathbf{v}) + \sum_{i=1}^{k} \int_{\Omega} v_i f_i dx, \quad \text{s.t. } v_i(x) \geq 0, \sum_{i=1}^{k} v_i(x) = 1, \; \forall x \in \Omega$$

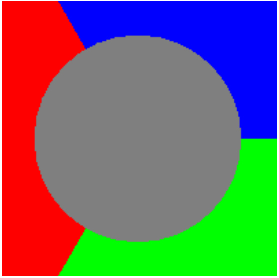
- Minimization using the proposed primal-dual algorithm



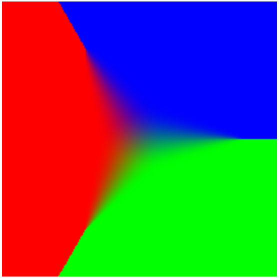
[P., Cremers, Bischof, Chambolle, '09], [Chambolle, Cremers, P. '11]

The triple-junction problem in 2D

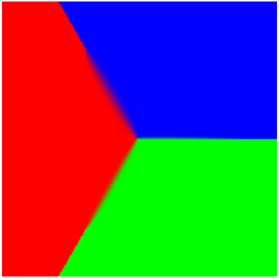
A comparison using the "triple-junction" problem



(a) Input



(b) Zach et al.



(c) Ours

Our relaxation is provably the largest in a certain class of local convex envelopes

Image segmentation

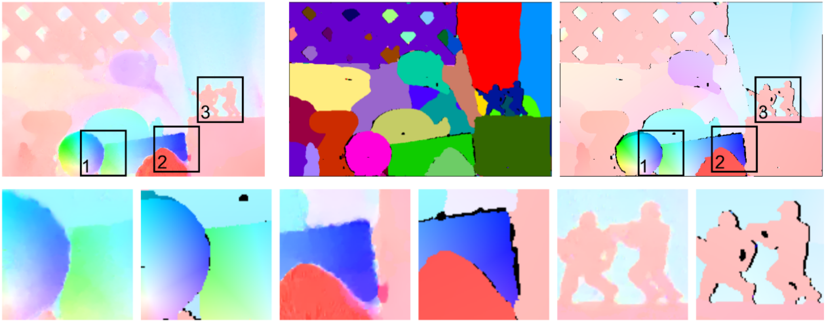
Piecewise constant Mumford-Shah segmentation with $k = 16$ labels
Data term: $f_i = (I - \mu_i)^2$

(a) Input

(b) Segmentation

Minimal surfaces in 3D

Motion segmentation



Joint motion estimation and segmentation [Unger, Werlberger, P., Bischof '12]

Overview

1 Introduction

2 Non-convex Optimization

3 Convex Optimization

4 Local Convexification

5 Convex Envelopes

6 Conclusion

Summary

- Energy minimization methods

Summary

- Energy minimization methods
- Convex versus non-convex models

Summary

- Energy minimization methods
- Convex versus non-convex models
- Efficient algorithm for minimizing the sum of a smooth and a convex function

Summary

- Energy minimization methods
- Convex versus non-convex models
- Efficient algorithm for minimizing the sum of a smooth and a convex function
- Efficient primal-dual algorithm for minimizing convex-concave saddle-point problems

Summary

- Energy minimization methods
- Convex versus non-convex models
- Efficient algorithm for minimizing the sum of a smooth and a convex function
- Efficient primal-dual algorithm for minimizing convex-concave saddle-point problems
- Example of non-convex optimization

Summary

- Energy minimization methods
- Convex versus non-convex models
- Efficient algorithm for minimizing the sum of a smooth and a convex function
- Efficient primal-dual algorithm for minimizing convex-concave saddle-point problems
- Example of non-convex optimization
- Local convexification

Summary

- Energy minimization methods
- Convex versus non-convex models
- Efficient algorithm for minimizing the sum of a smooth and a convex function
- Efficient primal-dual algorithm for minimizing convex-concave saddle-point problems
- Example of non-convex optimization
- Local convexification
- Convex envelopes

Summary

- Energy minimization methods
- Convex versus non-convex models
- Efficient algorithm for minimizing the sum of a smooth and a convex function
- Efficient primal-dual algorithm for minimizing convex-concave saddle-point problems
- Example of non-convex optimization
- Local convexification
- Convex envelopes

- **Tried to bridge the gap between convex and non-convex approaches**

Summary

- Energy minimization methods
- Convex versus non-convex models
- Efficient algorithm for minimizing the sum of a smooth and a convex function
- Efficient primal-dual algorithm for minimizing convex-concave saddle-point problems
- Example of non-convex optimization
- Local convexification
- Convex envelopes

- **Tried to bridge the gap between convex and non-convex approaches**

- In future, we will have to consider considerably more complex models

Summary

- Energy minimization methods
- Convex versus non-convex models
- Efficient algorithm for minimizing the sum of a smooth and a convex function
- Efficient primal-dual algorithm for minimizing convex-concave saddle-point problems
- Example of non-convex optimization
- Local convexification
- Convex envelopes

- **Tried to bridge the gap between convex and non-convex approaches**

- In future, we will have to consider considerably more complex models
- It is very likely that we will not be able to avoid non-convexity!

Thank you for your attention!