Towards Differentiation of Solution Mappings of Non-smooth Optimization Problems

Peter Ochs
Mathematical Optimization Group
University of Tübingen

— 06.09.2021 —
Inverse Problem:

The degradation process is modeled as follows

\[ f \approx A(h) + N \]

\[ N: \text{additive pixel-wise noise} \]

Goal: Reconstruct \( h \)
Image Deblurring: of an image $f$ (using Total Variation)

$$\min_u \frac{1}{2} \|Au - f\|^2 + \lambda \|Du\|_{2,1}$$

▷ A convolution / blurr operator; $D$ discrete derivative operator

clean image  blurry image  reconstruction
Regularization weight has a huge impact on the result!

- **Ground truth**
- **Noisy input**
- **Reconstruction; small $\lambda$**
- **Reconstruction; large $\lambda$**
Variational Problems with Regularization

$$\min_u E_\lambda(u), \quad E_\lambda(u) = \underbrace{D(u)}_{\text{data term}} + \lambda \underbrace{R(u)}_{\text{regularization term}}.$$  

- $\lambda > 0$ is a regularization parameter.

**How to find the best regularization weight $\lambda$?**
- hand-tuning
- grid search
What about learning the whole regularization term?

\[ R(u, \nu, \vartheta) := \sum_{i,j} \left( \sum_{k=1}^{m} \nu_k \rho \left( \sum_{l=1}^{L} \vartheta_{kl} (K_l u)_{ij} \right) \right) \]

- \( K_l \) are predefined basis filters (e.g. DCT filter)
- \( \rho \) is a potential function (convex)

(This regularizer reflects a 1-hidden-layer neural network.)
Deep Neural Network with Inference Layer/Variational Model:

\[ R(\mathcal{N}^R_\theta(u)) \]

\[ D(\mathcal{N}^D_\vartheta(u)) \]

⇝ Resurge of variational models as layers in deep learning.

[Ranftl, Pock 14]
What about learning the whole regularization term?

\[ R(u, \nu, \vartheta) := \sum_{i,j} \left( \sum_{k=1}^{m} \nu_k \rho \left( \sum_{l=1}^{L} \vartheta_{kl}(K_l u)_{ij} \right) \right) \]

- \( K_l \) are predefined basis filters (e.g. DCT filter)
- \( \rho \) is a potential function (convex)

(This regularizer reflects a 1-hidden-layer neural network.)

How to find the best weights \( \nu, \vartheta \)?

- hand-tuning and grid search are not feasible
- sampling and regression of loss function using Gaussian processes or Random Fields (up to \( \approx 200 \) parameters)
- gradient based bi-level optimization (several 100 000 parameters)
Bilevel optimization / parameter Learning

\[
\min_{\theta \in \mathbb{R}^P, x^*} \mathcal{L}(x^*(\theta), \theta) \quad \text{(upper level)}
\]

\[
s.t. \quad x^*(\theta) \in \arg \min_{x \in \mathbb{R}^N} E(x, \theta) \quad \text{(lower level)}
\]

- \( \theta \in \mathbb{R}^P \): optimization variable parameter (vector).
- \( \mathcal{L} : \mathbb{R}^N \times \mathbb{R}^P \to \mathbb{R} \): smooth loss function.
- \( E : \mathbb{R}^N \times \mathbb{R}^P \to \overline{\mathbb{R}} \): parametric (energy) minimization problem. (convex for each \( \theta \in \mathbb{R}^P \))
- \( x^* : \mathbb{R}^P \to \mathbb{R}^N \) is a selection of the solution mapping of \( E \).
Bilevel optimization / parameter Learning

\[
\min_{\theta \in \mathbb{R}^P, x^*} \mathcal{L}(x^*(\theta), \theta) \quad \text{(upper level)}
\]
\[
\text{s.t. } x^*(\theta) \in \arg \min_{x \in \mathbb{R}^N} E(x, \theta) \quad \text{(lower level)}
\]

▶ $\theta \in \mathbb{R}^P$: optimization variable parameter (vector).

▶ $\mathcal{L}: \mathbb{R}^N \times \mathbb{R}^P \to \mathbb{R}$: smooth loss function.

▶ $E: \mathbb{R}^N \times \mathbb{R}^P \to \overline{\mathbb{R}}$: parametric (energy) minimization problem. (convex for each $\theta \in \mathbb{R}^P$)

▶ $x^*: \mathbb{R}^P \to \mathbb{R}^N$ is a selection of the solution mapping of $E$.

Gradient based optimization requires $\nabla_\theta \mathcal{L}(x^*(\theta^{(k)}), \theta^{(k)})$. 
Computation of $\nabla_{\theta} \mathcal{L}(x^*(\theta^{(k)}), \theta^{(k)})$.

- If loss function $\mathcal{L}$ and solution mapping $x^*(\theta)$ are differentiable:

$$\nabla_{\theta} \mathcal{L}(x^*(\theta^{(k)}), \theta^{(k)}) = \left( \frac{\partial x^*}{\partial \theta} (\theta^{(k)}) \right)^\top \nabla_x \mathcal{L}(x^*, \theta^{(k)}) + \nabla_{\theta} \mathcal{L}(x^*, \theta^{(k)})$$

- Actually, we are interested in the directional derivative of $x^*(\theta)$ at $\theta^{(k)}$ in direction $v^{(k)} := \nabla_x \mathcal{L}(x^*, \theta^{(k)})$:

$$\left( \frac{\partial x^*}{\partial \theta} (\theta^{(k)}) \right)^\top v^{(k)}$$
Computation of $\nabla_{\theta} \mathcal{L}(x^*(\theta^{(k)}), \theta^{(k)})$.

- If loss function $\mathcal{L}$ and solution mapping $x^*(\theta)$ are differentiable:

$$\nabla_{\theta} \mathcal{L}(x^*(\theta^{(k)}), \theta^{(k)}) = \left(\frac{\partial x^*}{\partial \theta}(\theta^{(k)})\right)^\top \nabla_x \mathcal{L}(x^*, \theta^{(k)}) + \nabla_{\theta} \mathcal{L}(x^*, \theta^{(k)})$$

- Actually, we are interested in the directional derivative of $x^*(\theta)$ at $\theta^{(k)}$ in direction $v^{(k)} := \nabla_x \mathcal{L}(x^*, \theta^{(k)})$:

$$\left(\frac{\partial x^*}{\partial \theta}(\theta^{(k)})\right)^\top v^{(k)}$$

Study $\frac{\partial x^*}{\partial \theta}$ or, more generally, sensitivity of the solution mapping.
Strategies for differentiation:

In the following: \[ x^*(\theta) \in \arg \min_{x \in \mathbb{R}^N} E(x, \theta) \quad (E \text{ smooth}) \]

Goal: Compute \[ \frac{\partial x^*}{\partial \theta} (\theta) \]

Strategies:

▶ Implicit differentiation. (requires \( E \) strictly convex)
▶ Unrolling an algorithm. \( \text{(use automatic differentiation)} \)
▶ Implicit differentiation or unrolling of a fixed point equation. [O. et al. ’16]

Warning:

▶ Even for smooth \( E \), the solution mapping \( x^*(\theta) \) is often non-smooth.
▶ Solution mapping multivalued, when \( \arg \min_{x \in \mathbb{R}^N} E(x, \theta) \) is not unique.
**Implicit Differentiation**: (widely used approach; constraints via KKT)

- The optimality condition is $\nabla_x E(x, \theta) = 0$.
- This implicitly defines $x^*(\theta)$ (**implicit function theorem**).
- Let $(x^*, \theta)$ be such that $\nabla_x E(x^*, \theta) = 0$, then, if [...] we have
  $$\frac{\partial x^*}{\partial \theta}(\theta) = -\left(H_E(x^*, \theta)\right)^{-1} \frac{\partial^2 E}{\partial \theta \partial x}(x^*, \theta).$$

**Disadvantages**: (reasons why we avoid this approach)

- Requires twice differentiability of $E$.
- Requires several approximations: $x^*$ and $H_E^{-1}$ (solve linear system).
- Unstable for badly conditioned $H_E$.
- Requires estimation (and storing) $H_E$. 
Unrolling an algorithm: $A^{(n+1)}(x^{(0)}, \theta) \rightarrow x^*(\theta)$ for $n \rightarrow \infty$

- Approximate by a fixed $n \in \mathbb{N}$: (where $x^{(0)}$ is some initialization.)
  
  $$x^*(\theta) \approx A^{(n+1)}(x^{(0)}, \theta)$$

$\Rightarrow$ derivative by reverse mode AD (backpropagation)

- **Advantages:**
  - Output is **unambiguous** (unique), also if $\text{argmin } E$ is multi-valued.
  - After the algorithm $A$ is fixed, the approach is exact.
  - All iterations depend on the same parameter $\Rightarrow$ truncate backprop.
  - For $E$ strongly convex, we have convergence rates for

  $$\frac{\partial A^{(n+1)}}{\partial \theta} \xrightarrow{n \rightarrow \infty} \frac{\partial x^*}{\partial \theta}$$

  Accelerated convergence for accelerated algorithms! [Mehmood, O. '20]

- **Disadvantages:** Store all intermediate iterates (in reverse mode).
Replace $\arg\min E(x, \theta)$ by a fixed point equation:

$$x^*(\theta) = A(x^*(\theta), \theta)$$

▶ Implicit function theorem:

$$\frac{\partial x^*}{\partial \theta} = \left(I - \frac{\partial A}{\partial x^*}\right)^{-1} \frac{\partial A}{\partial \theta}$$

▶ Using

$$\left(I - \frac{\partial A}{\partial x^*}\right)^{-1} \|\frac{\partial A}{\partial x^*}\| < 1 \sum_{n=0}^{\infty} \left(\frac{\partial A}{\partial x^*}\right)^n \frac{\partial A}{\partial \theta} \approx \sum_{n=0}^{n_0} \left(\frac{\partial A}{\partial x^*}\right)^n \frac{\partial A}{\partial \theta}$$

we obtain AD with intermediate variables replaced by the optimum. [O. et al. ’16]

↝ accelerates convergence rate [Mehmood, O. ’20].

▶ Requires to store only $x^*$. 
Strategies for differentiation if $E$ is non-smooth:

In the following: 

$$x^*(\theta) \in \arg \min_{x \in \mathbb{R}^N} E(x, \theta) \quad (E \text{ non-smooth})$$

**Goal:** Compute 

$$\frac{\partial x^*}{\partial \theta}(\theta) \quad ??$$

**Strategies:**

▶ Strategies above after smoothing $E$.

⇝ often instable or requires significant smoothing
⇝ no approximation bounds

▶ Weak differentiation of iterative algorithms. [Deledalle et al. ’14]

(using Rademacher Theorem; show update map is Lipschitz)

⇝ yields only weak derivatives
⇝ requires “Subgradient Descent” methods for bilevel problem
⇝ convergence of derivative sequence unknown
Strategies for differentiation if $E$ is non-smooth: (continued)

▶ Unrolling a “smooth algorithm” that solves the non-smooth problem. [O. et al. ’16] (see next slides)

$\Rightarrow E$ is non-smooth,

$\Rightarrow \mathcal{A}^{(n+1)}(x^{(0)}, \theta) \rightarrow x^*(\theta)$ for $n \rightarrow \infty$,

$\Rightarrow$ update mapping $\mathcal{A} : \mathbb{R}^N \times \mathbb{R}^P \rightarrow \mathbb{R}^N$ is smooth.

▶ Implicit Differentiation of fixed point equations. [O. et al. ’16]
(“unrolling a smooth algorithm”-strategy but for fixed-point equation)

▶ Implicit Differentiation for partly smooth functions. [Vaiter et al. 16, Riis ’19]

▶ Unrolling in the partial smoothness framework. [Riis, PhD Thesis ’19]
Unrolling a “smooth algorithm”:

- Key are Bregman distances $D_\psi$.
- Update mapping $B_\psi$ of Bregman Proximal Gradient Method solves

$$B_\psi(c(\theta)) := \arg\min_x g(x) + \langle c(\theta), x \rangle + \psi(x)$$

- $c$ constant vector (depends on current iterate and $\theta$); $g$ is possibly non-smooth (e.g. indicator function)
- Bregman Proximal Gradient or Bregman Primal Dual Algorithm.
- Idea similar to barrier approach $\rightsquigarrow$ all iterates lie in interior
Example 1:  (non-negativity constraint)

- $X = [x \geq 0]$ and $g = \delta_X$

- entropy function $\psi(x) = x \log(x)$

- **Bregman Proximal mapping:**

  $$\mathcal{B}_\psi(c(\theta)) = \exp(-c(\theta) - 1)$$

**Application:** non-negative least squares problem.
Example 2: (simplex constraint)

- Minimize $f(x)$ over the unit simplex in $\mathbb{R}^N$

  $$\left\{ x \in \mathbb{R}^N \mid \sum_{i=1}^{N} x_i = 1 \text{ and } x_i \geq 0 \right\}.$$ 

- Use entropy $\psi(x) = \sum_{i=1}^{N} x_i \log(x_i)$ to drop the non-negativity constraint.

- **Bregman proximal update mapping**:

  $$\left( \mathcal{B}_\psi (c) \right)_i = \frac{\exp(-c_i - 1)}{\sum_{j=1}^{N} \exp(-c_j - 1)} \quad \text{for } i = 1, \ldots, N.$$ 

**Application**: Multi-label segmentation problem or Matrix games.
Example 3: (box constraint)

- Use \( \psi(x) = \frac{1}{2}((x + 1) \log(x + 1) + (1 - x) \log(1 - x)) \).

- Bregman proximal update mapping:
  \[
  B_\psi(c) = \frac{\exp(-2c) - 1}{\exp(-2c) + 1}.
  \]

These linear functions are important thanks to convex duality.

- For instance the TV-norm can be represented as
  \[
  \|Dx\|_1 = \max_y \langle Dx, y \rangle + \delta_{[-1 \leq y \leq 1]}(y),
  \]
  which can be employed in Bregman Primal–Dual Algorithms.
Differentiation Strategies for partly smooth functions [Lewis ’03] (Definition here in convex setting taken from [Liang, Fadili, Peyré ’14])

\[ J \in \Gamma_0(\mathbb{R}^N) \text{ and } \partial J(x) \neq \emptyset. \] We call \( J \) partly smooth at \( x \) relative to \( M \ni x \), if the following conditions are satisfied:

- **(Smoothness)** \( M \) is a \( C^2 \)-Manifold around \( x \) and \( J|_M \in C^2 \)
- **(Sharpness)** The tangent space \( T_M \) is \( \text{par}(\partial J(x)) \perp \).
- **(Continuity)** \( \partial J \) is continuous at \( x \) relative to \( M \).

Examples: \( \ell_1 \)-norm, \( \ell_{2,1} \)-norm, \( \ell_\infty \)-norm, nuclear norm, TV-norm, ...

from [G. Peyré, talk “Low Complexity Regularization of Inverse Problems”, 2014]
Implicit differentiation under partial smoothness
(from [Vaiter et al. '16])

\[ E(x, \theta) = F(x, \theta) + J(x) \]

**Assumptions:**

- \( F \) block-wise \( C^2 \)
- \( \bar{\theta} \not\in \mathcal{H} \) (certain transition space of measure 0; non-degeneracy ass.)
- \( J \in \Gamma_0 \) is partly smooth at solution \( x^*(\bar{\theta}) \) relative to \( \mathcal{M} \)
- restricted positive definiteness holds at \( x^*(\bar{\theta}) \).

**Then:**

- there exists open \( \mathcal{V} \ni \bar{\theta} \) and a mapping \( \hat{x} : \mathcal{V} \rightarrow \mathcal{M} \) such that
  1. For all \( \theta \in \mathcal{V} \), \( \hat{x}(\theta) \) is a solution that coincides with \( x^* \) at \( \bar{\theta} \)
  2. \( \hat{x} \in C^1(\mathcal{V}) \) and for all \( \theta \in \mathcal{V} \):

\[
\frac{\partial \hat{x}}{\partial \theta}(\theta) = - \left( \nabla^2_{\mathcal{M}} F(\hat{x}(\theta), \theta) + \nabla^2_{\mathcal{M}} J(\hat{x}(\theta)) \right)^\dagger P_{T_{\hat{x}(\theta)} \frac{\partial (\nabla F)}{\partial \theta}}(\hat{x}(\theta), \theta)
\]
Unrolling algorithms under partial smoothness (idea):

- $E$ is non-smooth (but partly smooth)
- Solution lies on a smooth manifold $\mathcal{M}$ and is stable
- $\mathcal{A}^{(n+1)}(x^{(0)}, \theta) \rightarrow x^*(\theta)$ for $n \rightarrow \infty$
- requires non-degeneracy assumption for $x^*(\theta)$
- Many algorithms have the **finite identification property** (see papers by [J. Liang]):
  
  $$\exists n_0 \in \mathbb{N}: x^{(n)} \in \mathcal{M} \text{ for all } n \geq n_0,$$

  hence the update mapping $\mathcal{A}$ becomes smooth eventually.
Differentiating the Value Function by using Convex Duality, AISTATS 2021 (different situation) [Mehmood, O. ’21]

\[ p(u) = \min_x f(x, u) \]

Compute the derivative of the value function:

\[ \nabla p(u) \quad \text{(} p \text{ can be smooth with } f \text{ being smooth)} \]

- Requires \( f \) to be convex in \( x \) and \( u \).
- Derivative is given by (convex duality; \( f^* \): convex conjugate of \( f \))

\[ \partial p(u) = \arg \max_y \langle u, y \rangle - f^*(0, y) \]

\( \rightsquigarrow \) derivative by solving an optimization problem (dual problem).

\( \rightsquigarrow \) convergence rates of the derivative sequence in situations where \( f \) is not strongly convex.
Conclusion

- Bilevel optimization as framework for parameter learning.
- Solve bilevel problem by gradient based algorithms.
- Strategies for computing the derivative of the solution mapping in smooth and non-smooth setup:
  - Implicit differentiation.
  - Weak differentiation of iterative algorithms.
  - Unrolling a “smooth algorithm” for a non-smooth problem.
  - Implicit Differentiation or unrolling of fixed point equations.
  - Differentiation under partial smoothness assumption.
Derivation:
\[ \partial_\theta x^* \approx \partial_\theta x^{(n+1)} \]
\[ = \partial_x x^{(n+1)} \partial_\theta x^{(n)} + \partial_\theta x^{(n+1)} \]
\[ = \partial_x x^{(n+1)} (\partial_x x^{(n)} \partial_\theta x^{(n-1)} + \partial_\theta x^{(n)}) + \partial_\theta x^{(n+1)} \]
\[ = \ldots \]
\[ = \sum_{k=0}^{n+1} \left( \prod_{j=k}^{n} \partial_x x^{(j+1)} \right) \partial_\theta x^{(k)} \]
\[ = \sum_{k=0}^{n+1} \left( \prod_{j=k}^{n} \partial_x A(x^{(j)}, \theta) \right) \partial_\theta A(x^{(k-1)}, \theta) \]
\[ \approx \sum_{k=0}^{n+1} \left( \prod_{j=k}^{n} \partial_x A(x^*, \theta) \right) \partial_\theta A(x^*, \theta) \]
\[ = \sum_{k=0}^{n+1} \left( \partial_x A(x^*, \theta) \right)^k \partial_\theta A(x^*, \theta) \approx \left( I - \partial_x A(x^*, \theta) \right)^{-1} \partial_\theta A(x^*, \theta) \]