Differentiating Solution Mappings of Non-smooth Optimization Problems

Peter Ochs
Mathematical Optimization for Data Science
Saarland University

— 04.09.2023 —

joint work: Sheheryar Mehmood
Variational Problems with Regularization

\[
\min_x E_\lambda(x), \quad E_\lambda(x) = \underbrace{D(x)}_{\text{data term}} + \lambda \underbrace{R(x)}_{\text{regularization term}}.
\]

\( \lambda > 0 \) is a regularization parameter.

Regularization weight has a huge impact on the result!
Variational Problems with Regularization

What about learning the whole regularization term?

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

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

(This regularizer reflects a 1-hidden-layer neural network.)
Variational Problems with Regularization

What about learning the whole regularization term?

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

- \( K_l \) 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)
Applications in Machine Learning

Similar Problems:

◆ (Hyper)Parameter Learning [Domke '12], [Kunisch, Pock '13], [O. et al '16], ...

◆ Implicit Models [Amos, Kolter '17], [Agrawal et al. '19], [Bai, Kolter, Koltun '19], [Chen et al. '21], ...

◆ Meta Learning [Hospedales et al. '21], ...

◆ Learning to Optimize [Chen et al. '21], ...

◆ Sensitivity Analysis.
Bilevel optimization / parameter learning

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

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

- \( \theta \in \mathbb{R}^P \): optimization variable parameter (vector).
- \( \mathcal{L} : \mathbb{R}^N \times \mathbb{R}^P \to \mathbb{R} \): smooth loss function.
- \( E : \mathbb{R}^N \times \mathbb{R}^P \to \mathbb{R} \): 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

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

subject to \(x^*(\theta) \in \arg \min_{x \in \mathbb{R}^N} E(x, \theta)\) \(\text{(lower level)}\)

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

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

i.e., in particular, the sensitivity of the solution mapping \(\frac{\partial x^*}{\partial \theta}\).
Bilevel optimization / parameter learning

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

\[
\theta \in \mathbb{R}^P : \text{optimization variable parameter (vector)}.\]

\[
\mathcal{L} : \mathbb{R}^N \times \mathbb{R}^P \to \mathbb{R} : \text{smooth loss function}.
\]

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

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

Gradient based optimization requires \( \nabla_\theta \mathcal{L}(x^*(\theta^{(k)}), \theta^{(k)}) \), i.e., in particular, the sensitivity of the solution mapping \( \frac{\partial x^*}{\partial \theta} \).

<table>
<thead>
<tr>
<th>Outline:</th>
<th>( E )</th>
<th>type of bilevel algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td>(I) smooth setting</td>
<td>smooth</td>
<td>gradient</td>
</tr>
<tr>
<td>(II) partly smooth setting</td>
<td>structured non-smooth</td>
<td>gradient</td>
</tr>
<tr>
<td>(III) non-smooth setting</td>
<td>non-smooth</td>
<td>generalized derivative</td>
</tr>
</tbody>
</table>
(I) Smooth Setting: 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

\[ \frac{\partial x^*}{\partial \theta} (\theta) \]
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. \( \text{(use automatic differentiation)} \)
- Unrolling of a fixed point equation.

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.
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

$$\frac{\partial x^*}{\partial \theta}(\theta) = -\left( \frac{\partial^2 E}{\partial x^2}(x^*, \theta) \right)^{-1} \frac{\partial^2 E}{\partial \theta \partial x}(x^*, \theta).$$
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

$$\frac{\partial x^*}{\partial \theta}(\theta) = -\left(\frac{\partial^2 E}{\partial x^2}(x^*, \theta)\right)^{-1} \frac{\partial^2 E}{\partial \theta \partial x}(x^*, \theta).$$

**Disadvantages:**

- Requires twice differentiability of $E$.
- Requires several approximations: $x^*$ and $\left(\frac{\partial^2 E}{\partial x^2}\right)^{-1}$ (solve linear system).
- Unstable for badly conditioned $\frac{\partial^2 E}{\partial x^2}$.
- Requires estimation (and storing) $\frac{\partial^2 E}{\partial x^2}$. 
Approximate by a fixed \( n \in \mathbb{N} \): (where \( x^{(0)} \) is some fixed initialization)
\[
x^*(\theta) \approx x^{(n+1)} := A^{(n+1)}(x^{(0)}, \theta) = A \circ \ldots \circ A(x^{(0)}, \theta)
\]

Evaluate the chain rule by reverse mode AD (backpropagation):
\[
\frac{\partial A^{(n+1)}}{\partial \theta}(x^{(0)}, \theta) = \frac{\partial A}{\partial x}(x^{(n)}, \theta) \frac{\partial A^{(n)}}{\partial \theta}(x^{(0)}, \theta) + \frac{\partial A}{\partial \theta}(x^{(n)}, \theta) = \ldots \text{unroll} \ldots
\]
Approximate by a fixed $n \in \mathbb{N}$: (where $x^{(0)}$ is some fixed initialization)

$$x^*(\theta) \approx x^{(n+1)} := A^{(n+1)}(x^{(0)}, \theta) = A \circ \ldots \circ A(x^{(0)}, \theta)$$

Algorithmic formulation: Set $J^{(1)}_{\theta} := \frac{\partial A}{\partial \theta}(x^{(0)}, \theta)$ and for $n = 1, 2, \ldots$:

$$J^{(n+1)}_{\theta} = \frac{\partial A}{\partial x}(x^{(n)}, \theta) J^{(n)}_{\theta} + \frac{\partial A}{\partial \theta}(x^{(n)}, \theta)$$
Unrolling / Automatic Differentiation (AD)

Approximate by a fixed \( n \in \mathbb{N} \): (where \( x^{(0)} \) is some fixed initialization)

\[
x^*(\theta) \approx x^{(n+1)} := A^{(n+1)}(x^{(0)}, \theta) = A \circ \ldots \circ A(x^{(0)}, \theta)
\]

**Algorithmic formulation:** Set \( J^{(1)}_\theta := \frac{\partial A}{\partial \theta}(x^{(0)}, \theta) \) and for \( n = 1, 2, \ldots \):

\[
J^{(n+1)}_\theta = \frac{\partial A}{\partial x}(x^{(n)}, \theta)J^{(n)}_\theta + \frac{\partial A}{\partial \theta}(x^{(n)}, \theta)
\]

**Advantages:**
- Output is **unambiguous** (unique), even in case \( \arg\min E \) is multi-valued.
- After the algorithm \( A \) and \( n \in \mathbb{N} \) are fixed, the approach is exact.
- All iterations depend on the same parameter \( \Rightarrow \) possibly truncate backprop.
- Easy implementation using standard AD packages.
- For \( E \) strongly convex, we have convergence rates for

\[
J^{(n+1)}_\theta \xrightarrow{n \to \infty} \frac{\partial x^*}{\partial \theta}(\theta) \quad \text{Accelerated convergence for accelerated algorithms!} \quad [\text{Mehmood, O. '20}]
\]

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

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

Imitate $n \in \mathbb{N}$ iterations of $A$ starting at $x^{(0)} := x^*$:

$$A^{(n+1)}(x^*, \theta) = A \circ \ldots \circ A(x^*, \theta)$$
Replace $\arg\min E(x, \theta)$ by a fixed point equation:

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

Imitate $n \in \mathbb{N}$ iterations of $A$ starting at $x^{(0)} := x^*$:

$$A^{(n+1)}(x^*, \theta) = A \circ \ldots \circ A(x^*, \theta)$$

Evaluate the chain rule by reverse mode AD (backpropagation): For $n = 1, 2, \ldots$:

$$\hat{J}_{\theta}^{(n+1)} = \frac{\partial A}{\partial x}(x^*, \theta) \hat{J}_{\theta}^{(n)} + \frac{\partial A}{\partial \theta}(x^*, \theta)$$
Replace $\arg \min E(x, \theta)$ by a fixed point equation:

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

Imitate $n \in \mathbb{N}$ iterations of $A$ starting at $x^{(0)} := x^*$:

$$A^{(n+1)}(x^*, \theta) = A \circ \ldots \circ A(x^*, \theta)$$

Evaluate the chain rule by reverse mode AD (backpropagation): For $n = 1, 2, \ldots$:

$$\hat{j}_{\theta}^{(n+1)} = \frac{\partial A}{\partial x}(x^*, \theta) \hat{j}_{\theta}^{(n)} + \frac{\partial A}{\partial \theta}(x^*, \theta) \quad n \to \infty \quad \frac{\partial x^*}{\partial \theta}(\theta)$$

$\leadsto$ faster convergence rate of derivative sequence [Mehmood, O. '20].

(known in AD community [Gilbert '92, Christianson '94];
connection to ID via Von Neumann series.)
Replace $\arg\min E(x, \theta)$ by a fixed point equation:

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

Imitate $n \in \mathbb{N}$ iterations of $A$ starting at $x^{(0)} := x^*$:

$$A^{(n+1)}(x^*, \theta) = A \circ \ldots \circ A(x^*, \theta)$$

Evaluate the chain rule by reverse mode AD (backpropagation): For $n = 1, 2, \ldots$:

$$\hat{J}_\theta^{(n+1)} = \frac{\partial A}{\partial x}(x^*, \theta) \hat{J}_\theta^{(n)} + \frac{\partial A}{\partial \theta}(x^*, \theta) \quad n \to \infty \quad \frac{\partial x^*}{\partial \theta}(\theta)$$

$\leadsto$ faster convergence rate of derivative sequence [Mehmood, O. ’20].

(known in AD community [Gilbert ’92, Christianson ’94]; connection to ID via Von Neumann series.)

Equivalent to AD with intermediate variables replaced by the optimum: [O. et al. ’16]

$$J_\theta^{(n+1)} = \frac{\partial A}{\partial x}(x^{(n)}(\theta), \theta) J_\theta^{(n)} + \frac{\partial A}{\partial \theta}(x^{(n)}, \theta)$$
Replace $\arg \min E(x, \theta)$ by a fixed point equation:

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

Imitate $n \in \mathbb{N}$ iterations of $A$ starting at $x^{(0)} := x^*$:

$$A^{(n+1)}(x^*, \theta) = A \circ \ldots \circ A(x^*, \theta)$$

Evaluate the chain rule by reverse mode AD (backpropagation): For $n = 1, 2, \ldots$:

$$\hat{J}^{(n+1)}(\theta) = \frac{\partial A}{\partial x}(x^*, \theta) \hat{J}^{(n)}(\theta) + \frac{\partial A}{\partial \theta}(x^*, \theta)$$

$n \to \infty \quad \frac{\partial x^*}{\partial \theta}(\theta)$

$\rightsquigarrow$ faster convergence rate of derivative sequence [Mehmood, O. ’20].

(known in AD community [Gilbert ’92, Christianson ’94]; connection to ID via Von Neumann series.)

Equivalent to AD with intermediate variables replaced by the optimum: [O. et al. ’16]

$$J^{(n+1)}(\theta) = \frac{\partial A}{\partial x}(x^{(n)}, \theta) J^{(n)}(\theta) + \frac{\partial A}{\partial \theta}(x^{(n)}, \theta)$$

Key advantages:

- Requires to store only $x^*$.
- Easy implementation using standard AD packages.
Structured Non-Smooth Setting: Strategies for differentiation

In the following:

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

Goal: Compute

\[ \frac{\partial x^*}{\partial \theta} (\theta) \quad !? \]
In the following: \[ x^*(\theta) \in \arg\min_{x \in \mathbb{R}^N} E(x, \theta) \quad (E \text{ structured non-smooth}) \]

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

**Strategies that compute (classic) gradients:**

- Strategies above after smoothing \( E \) (ID, AD, FPAD):
  - Often instable or requires significant smoothing
  - No approximation bounds
Structured Non-Smooth Setting: Strategies for differentiation

In the following:

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

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

Strategies that compute (classic) gradients:

- Strategies above after smoothing \( E \) (ID, AD, FPAD):
  - often instable or requires significant smoothing
  - no approximation bounds

- Unrolling a “smooth algorithm” that solves the non-smooth problem. \([O. \text{ et al. '16}\]

  - \( A^{(n+1)}(x^{(0)}, \theta) \rightarrow x^*(\theta) \) for \( n \rightarrow \infty \), where \( A \) is a smooth mapping.
  - Idea: Assert that iterates lie in interior of constraint set.
    (e.g. Bregman Proximal Gradient Method)

- Smoothing is controlled by the number of iterations \( n \in \mathbb{N} \).
- Convergence of derivative sequence was not studied.
  Limit requires generalized notion of derivatives.
In the following: \[ x^*(\theta) \in \arg \min_{x \in \mathbb{R}^N} E(x, \theta) \quad (E \text{ structured non-smooth}) \]

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

Strategies that compute (classic) gradients:

- Strategies above after smoothing \( E \) (ID, AD, FPAD):
  - often instable or requires significant smoothing
  - no approximation bounds

- **Unrolling a “smooth algorithm”** that solves the non-smooth problem. [O. et al. '16]
  - \( A^{(n+1)}(x^{(0)}, \theta) \rightarrow x^*(\theta) \) for \( n \rightarrow \infty \), where \( A \) is a smooth mapping.
  - **Idea:** Assert that iterates lie in interior of constraint set.
    (e.g. Bregman Proximal Gradient Method)
  - Smoothing is controlled by the number of iterations \( n \in \mathbb{N} \).
  - Convergence of derivative sequence was **not** studied.
    Limit requires generalized notion of derivatives.

- FPAD variant requires \( x^* \) to lie in the interior of the constraint set.
Definition of partial smoothness for convex $E$ from [Liang, Fadili, Peyré '14] (original [Lewis '02]):

$E$ with $\partial E(x) \neq \emptyset$ is partly smooth at $x$ relative to $M \ni x$, if

- (Smoothness) $M$ is a $C^2$-Manifold around $x$ and $E|_M \in C^2$,
- (Sharpness) the normal space $N_M$ is $\text{par}(\partial E(x))$, and
- (Continuity) $\partial E$ 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]
Consider: \[ x^*(\theta) \in \arg \min_{x \in \mathbb{R}^N} E(x, \theta) \]

**Theorem:** [ID under Partial Smoothness [Lewis '02], [Vaiter et al. '17]]

**Assumptions:**
- \( E \) is partly smooth relative to \( \mathcal{M} \times \Omega \) where \( \Omega \subset \mathbb{R}^P \) is open, and
- for some \((x^*, \theta^*)\), the following **restricted positive definiteness** holds

\[
\forall v \in T_{x^*} \mathcal{M} : \quad \langle v, \nabla^2_{\mathcal{M}} E(x^*, \theta^*) v \rangle > 0
\]

and **non-degeneracy** holds

\[
0 \in \text{ri} (\partial_x E(x^*, \theta^*)).
\]

Then the **manifold version of the Implicit Differentiation (ID) holds:**
- There exists a neighborhood \( \Theta \) of \( \theta^* \) and a \( C^1 \) mapping \( \psi : \Theta \to \mathcal{M} \) such that
- \( \Theta \ni \theta \mapsto \psi(\theta) = \arg \min_x E(x, \theta) \), and
- \( \Theta \ni \theta \mapsto \frac{\partial \psi}{\partial \theta}(\theta) = -\nabla^2_{\mathcal{M}} E(\psi(\theta), \theta) \uparrow \frac{\partial}{\partial \theta} \nabla_{\mathcal{M}} E(\psi(\theta), \theta). \)
Idea: 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 \( A \) becomes smooth eventually.

Examples: Forward–backward Splitting (Proximal Gradient Descent), FISTA, ...
Unrolling / Automatic Differentiation (AD) under Partial Smoothness

**Idea:** Many algorithms have the **finite identification property** (see papers by [J. Liang]):

\[ \exists n_0 \in \mathbb{N}: \quad x^{(n)} \in \mathcal{M} \text{ for all } n \geq n_0, \]

hence the update mapping \( A \) becomes smooth eventually.

**Examples:** Forward–backward Splitting (Proximal Gradient Descent), FISTA, ...

---

**Theorem:** [AD for Forward–backward Splitting] [Liang et al. ’14], [Mehmood, O. ’22]

**Assumptions:**

- Partial smoothness, restricted positive definiteness, the non-degeneracy assumption, and convergence of the iterates.

Then,

- the rate of convergence for \( (x^{(n)})_{n \in \mathbb{N}} \) is actually **locally linear**.
- If, additionally, \( x^{(0)} \) is close enough to \( x^* \), then

\[
\frac{\partial A^{(n+1)}}{\partial \theta}(x^{(0)}, \theta) =: J^{(n)}_{\theta} \xrightarrow{n \to \infty} \frac{\partial \psi}{\partial \theta}(\theta).
\]

**Crucial issue:** Derivatives are not defined for \( x^{(n)} \notin \mathcal{M} \).

- Linear convergence, if all involved derivatives are locally Lipschitz.
Remedy of Differentiation Issue:

- Since $x^* \in \mathcal{M}$, by definition, all iteration mappings in FPAD are well defined.
Remedy of Differentiation Issue:

- Since $x^* \in \mathcal{M}$, by definition, all iteration mappings in FPAD are well defined.

**Theorem:** [Convergence of FPAD for Forward–backward Splitting [Mehmood, O. ’22]]

**Assumptions:**

- Partial smoothness, restricted positive definiteness and the non-degeneracy.

Then,

- the derivative sequence converges linearly

$$
\frac{\partial A^{(n+1)}}{\partial \theta} (x^*, \theta) =: \hat{J}_\theta^{(n)} \xrightarrow{n \to \infty} \frac{\partial \psi}{\partial \theta} (\theta).
$$
In the following:

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

Goal: Compute

\[ \frac{\partial x^*}{\partial \theta}(\theta) \]
Non-Smooth Setting: Strategies for Generalized Differentiation

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

Goal: Compute \( \frac{\partial x^*}{\partial \theta}(\theta) \) !?

Strategies:

- Weak differentiation of iterative algorithms. [Deledalle et al. '14]
  
  (using Rademacher Theorem; show update map is Lipschitz)

  \( \rightarrow \) yields only weak derivatives (not defined pointwise)

  \( \rightarrow \) convergence of derivative sequence unknown

- Using generalized non-smooth derivatives. [Bolte et al. '21]

  \( \rightarrow \) Some details on next slides.
Definition: [Bolte, Pauwels '19]

A locally Lipschitz function \( F: \mathbb{R}^N \to \mathbb{R} \) is **path differentiable**, if

- there exists a nowhere empty-valued mapping \( D_F: \mathbb{R}^N \Rightarrow \mathbb{R}^N \)
- that has a closed graph and is locally bounded, such that
- for any absolutely continuous curve \( \gamma: [0, 1] \to \mathbb{R}^N \)

\[
\frac{d}{dt} F \circ \gamma(t) = A\dot{\gamma}(t), \quad \forall A \in D_F(\gamma(t)), \quad \text{a.e. for } t \in [0, 1].
\]

Call \( D_F \) **conservative Jacobian** of \( F \).

Construction idea:

\( \Rightarrow \) By definition, **path differentiable functions admit a chain rule**.

\( \Rightarrow \) Formalization of backpropagation used in AD packages for neural networks.

\( \Rightarrow \) Straightforward generalization to locally Lipschitz mappings.
**Theorem:** [Implicit Path Differentiation (ID)] [Bolte et al. '21], [Bolte, Pauwels, Silveti–Falls '23]

**Assumptions:**
- $F : \mathbb{R}^N \times \mathbb{R}^P \rightarrow \mathbb{R}^N$ is path differentiable.
- $(x^*, \theta^*)$ satisfies the optimality condition
  \[ F(x^*, \theta^*) = 0. \]
- $DF(x^*, \theta^*)$ is convex and $\forall [A, B] \in DF(x^*, \theta^*)$, we have that $A$ is invertible.

Then the **path differentiable version of the Implicit Differentiation (ID) holds**:
- $\exists \Theta$ neighborhood of $\theta^*$ and path differentiable $\Psi : \Theta \rightarrow \mathbb{R}^N$ such that
  \[ \forall \theta \in \Theta : \quad F(\Psi(\theta), \theta) = 0, \]
- and $\Theta \ni \theta \mapsto D_{\Psi}(\theta) = \{-A^{-1}B \mid [A, B] \in DF(\Psi(\theta), \theta)\}$.

**Key Disadvantage:**
- Invertibility condition for all $A$ above; Sufficient condition by strong monotonicity of $F$. 
(III) Unrolling / Automatic Differentiation (AD)

Idea:
- Compositions of path differentiable functions are path differentiable.
- Path differentiable functions admit a chain rule.

Unrolling:
- Evaluate the chain rule. Set $\mathcal{J}_\theta^{(1)} := \{ J_\theta^{(1)} \mid [A_x^{(1)}, J_\theta^{(1)}] = D_A(x^{(0)}, \theta) \}$ and for $n = 1, 2, \ldots$:
  \[
  \mathcal{J}_\theta^{(n+1)} = \{ A_x^{(n)} J_\theta^{(n)} + A_\theta^{(n)} \mid [A_x^{(n)}, A_\theta^{(n)}] \in D_A(x^{(n)}, \theta), J_\theta^{(n)} \in \mathcal{J}_\theta^{(n)} \}.
  \]

Theorem: [Convergence of AD under path differentiability] [Bolte, Pauwels, Vaiter '22]

Assumption:
- $A$ is locally Lipschitz with unique fixed point $x^*(\theta)$, path differentiable in $(x, \theta)$,
- and $\exists \rho \in [0, 1): \forall (x, \theta)$ and all $[A, B] \in D_A(x, \theta)$, we have $\|A\| \leq \rho$.

Then:
- The gap between $\mathcal{J}_\theta^{(n+1)}$ and $\text{fix}(D_A(x^*(\theta), \theta))$ vanishes as $n \to \infty$.
- Under additional assumptions on a Lipschitz gradient selection structure, the convergence is linear.
Bi-level optimization / parameter learning

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

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

**Goal:** Compute \( \frac{\partial x^*}{\partial \theta}(\theta) \) !?

**Outline:**

<table>
<thead>
<tr>
<th>Type of bilevel algorithm</th>
<th>Smooth Setting</th>
<th>Partly Smooth Setting</th>
<th>Non-Smooth Setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>( E )</td>
<td>smooth</td>
<td>structured non-smooth</td>
<td>non-smooth</td>
</tr>
<tr>
<td>type of bilevel algorithm</td>
<td>gradient</td>
<td>gradient</td>
<td>generalized derivative</td>
</tr>
</tbody>
</table>

For each part:

- Implicit differentiation
- Automatic Differentiation / Backpropagation
- Fixed Point Automatic Differentiation
- Generalized derivatives of \( x^*(\theta) \) requires algorithms for non-smooth bi-level opt.