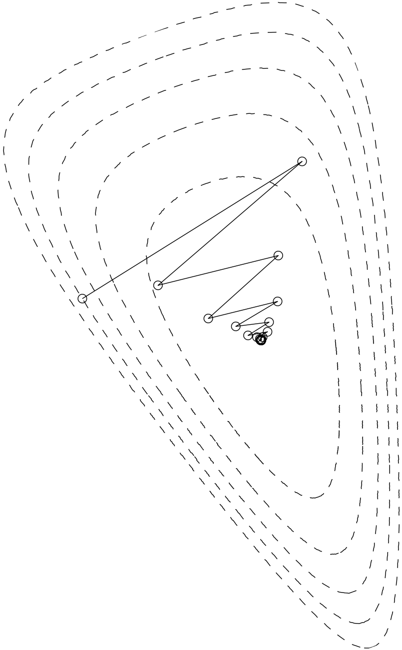


Optimization Algorithms

Implicit Functions & Differentiable Optimization

Marc Toussaint
Technical University of Berlin
Winter 2024/25

Outline

- Implicit Functions
  - Definition
  - Implicit Function Theorem and differentiation
- Differentiable Optimization

Implicit Functions

What is an Implicit Function?

- A function $F : \mathbb{R}^d \to Y$ can be defined **implicitly**, e.g. via

$$F(x) = \operatorname*{argmin}_y f(x, y) \qquad \text{optimality formulation}$$

  or alternatively via

$$F(x) = y \ \text{ s.t. } \ f(x, y) = 0 \qquad \text{standard (root) formulation}$$

- F is called *implicit function*, f is sometimes called **discriminative function**, as it discriminates "correct" outputs y from others.

What is an Implicit Function?

- A function $F : \mathbb{R}^d \to Y$ can be defined **implicitly**, e.g. via

$$F(x) = \operatorname*{argmin}_y f(x, y) \qquad \text{optimality formulation}$$

  or alternatively via

$$F(x) = y \ \text{ s.t. } \ f(x, y) = 0 \qquad \text{standard (root) formulation}$$

- F is called *implicit function*, f is sometimes called **discriminative function**, as it discriminates "correct" outputs y from others. Examples:
  - **ML classification**: A classifier $F : \mathbb{R}^d \to \{A, B, C\}$ is represented via a discriminative function $f(x, y)$ that assignes different neg-likelihoods to the three possible outputs $y \in \{A, B, C\}$ (cf. logistic regression, multi-class classification, conditional random fields).
  - **Implicit Surface Functions**: A 3D surface is implicitly defined as the *set* of points $y \in \mathbb{R}^3$ for which $f(y) = 0$ (often no parameter x here) (cf. recent work in CV and robotics to use neural implicit functions (NIF) to represent objects and scenes).
  - **Control & Robot Motion**: Optimal control and robot are described via optimality principles, e.g., motion such that various constraints $h(\text{environment, motion}) = 0$ are fulfilled.

Implicit Function Theorem

$$F : x \mapsto y \text{ s.t. } f(x,y) = 0$$

where $f : \mathbb{R}^d \times \mathbb{R}^n \to \mathbb{R}^n$ has n-dimensional output

- Is F really well-defined? E.g., what if no y solves $f(x,y) = 0$? What if multiple y solve $f(x,y) = 0$?

Implicit Function Theorem

- **Theorem:** Let $f(x,y)$, $x \in \mathbb{R}^d, y \in \mathbb{R}^n$ be a continuously differentiable $\mathbb{R}^n$-valued function (in C^1). Assume we have a point $(x^*, y^*) \in \mathbb{R}^{d+n}$ where

$$f(x^*, y^*) = 0 \quad \text{and} \quad \det \tfrac{\partial}{\partial y} f(x^*, y^*) \neq 0 \ .$$

a) Then there exists a radius r such that for each x, $|x - x^*| < r$, there exists a **unique** $y = F(x)$ such that $f(x,y) = 0$.
b) The implicit function F is continuously differentiable, and

$$f(x, F(x)) = 0 \quad \Rightarrow \quad \tfrac{\partial}{\partial x} f(x,y) + \tfrac{\partial}{\partial y} f(x,y) \tfrac{\partial}{\partial x} F(x) = 0 \quad \text{at } y = F(x),$$

and since $\frac{\partial}{\partial y} f$ is invertible, we have

$$\tfrac{\partial}{\partial x} F(x) = -[\tfrac{\partial}{\partial y} f(x,y)]^{-1} \tfrac{\partial}{\partial x} f(x,y) \ .$$

Implicit Function Theorem

- **Theorem:** Let $f(x,y)$, $x \in \mathbb{R}^d, y \in \mathbb{R}^n$ be a continuously differentiable $\mathbb{R}^n$-valued function (in C^1). Assume we have a point $(x^*, y^*) \in \mathbb{R}^{d+n}$ where

$$f(x^*, y^*) = 0 \quad \text{and} \quad \det \tfrac{\partial}{\partial y} f(x^*, y^*) \neq 0 \ .$$

a) Then there exists a radius r such that for each x, $|x - x^*| < r$, there exists a **unique** $y = F(x)$ such that $f(x,y) = 0$.
b) The implicit function F is continuously differentiable, and

$$f(x, F(x)) = 0 \quad \Rightarrow \quad \tfrac{\partial}{\partial x} f(x,y) + \tfrac{\partial}{\partial y} f(x,y) \tfrac{\partial}{\partial x} F(x) = 0 \quad \text{at } y = F(x),$$

and since $\frac{\partial}{\partial y} f$ is invertible, we have

$$\tfrac{\partial}{\partial x} F(x) = -[\tfrac{\partial}{\partial y} f(x,y)]^{\text{-}1} \tfrac{\partial}{\partial x} f(x,y) \ .$$

- $\det \frac{\partial}{\partial y} f(x^*, y^*) \neq 0 \ \Leftrightarrow$ Jacobian w.r.t. y has full rank $\ \Leftrightarrow \ f(x,y) = 0$ has non-zero gradient in all y-directions

Interpretation in view of Newton step*

(Same statement, just derived as Newton step for root finding)

- Assume you already found y^* to solve $f(x^*, y^*) = 0$ for a given x^*. But now the parameter/input x varies slightly. How does the solution y vary?
- Consider the 1st order Taylor approximation of f:

$$f(x,y) = \underbrace{f(x^*,y^*)}_{=0} + \tfrac{\partial}{\partial x} f(x^*,y^*)\ (x - x^*) + \tfrac{\partial}{\partial y} f(x^*,y^*)\ (y - y^*)$$

If we also want $f(x,y) = 0$, then we need

$$(y - y^*) = -[\tfrac{\partial}{\partial y} f]^{\text{-1}}\ \tfrac{\partial}{\partial x} f\ (x - x^*)\ ,$$

which is the Newton step for root finding, and coincides with the Implicit Function Theorem.

Differentiable Optimization

The KKT Implicit Function

- Consider a **parameterized** problem

$$x^*(\theta) = \underset{x}{\operatorname{argmin}}\, f(\theta, x) \ \text{ s.t. } \ g(\theta, x) \le 0,\ h(\theta, x) = 0$$

The KKT Implicit Function

- Consider a **parameterized** problem

$$x^*(\theta) = \underset{x}{\operatorname{argmin}}\, f(\theta, x) \;\; \textbf{s.t.} \;\; g(\theta, x) \le 0, \; h(\theta, x) = 0$$

- We define the **implicit function** $F : \theta \mapsto (x^*, \kappa^*, \lambda^*)$ **s.t.** $r(\theta, x, \kappa, \lambda) = 0$ for the KKT residual

$$r(\theta, x, \kappa, \lambda) = \begin{pmatrix} \nabla \left[f(\theta, x) + \lambda^\top g(\theta, x) + \kappa^\top h(\theta, x) \right] \\ h(\theta, x) \\ \operatorname{diag}(\lambda) g(\theta, x) \end{pmatrix}$$

(i.e., for any θ, F outputs the primal and dual solution to the KKT conditions.)

The KKT Implicit Function

- Consider a **parameterized** problem

$$x^*(\theta) = \operatorname*{argmin}_x f(\theta, x) \ \text{ s.t. } \ g(\theta, x) \le 0,\ h(\theta, x) = 0$$

- We define the **implicit function** $F : \theta \mapsto (x^*, \kappa^*, \lambda^*)$ s.t. $r(\theta, x, \kappa, \lambda) = 0$ for the KKT residual

$$r(\theta, x, \kappa, \lambda) = \begin{pmatrix} \nabla\left[f(\theta, x) + \lambda^\top g(\theta, x) + \kappa^\top h(\theta, x)\right] \\ h(\theta, x) \\ \mathrm{diag}(\lambda) g(\theta, x) \end{pmatrix}$$

(i.e., for any θ, F outputs the primal and dual solution to the KKT conditions.)

- In particular, at $(x, \kappa, \lambda) = F(\theta)$ we have

$$\tfrac{\partial}{\partial\theta} F = -\left[\tfrac{\partial}{\partial_{x\kappa\lambda}} r\right]^{-1} \tfrac{\partial}{\partial\theta} r\ .$$

The KKT Implicit Function

$$\frac{\partial}{\partial\theta} F = -[\frac{\partial}{\partial_{x\kappa\lambda}} r]^{\text{-1}} \ \frac{\partial}{\partial\theta} r \ .$$

– The matrix $\frac{\partial}{\partial x\kappa\lambda} r \in \mathbb{R}^{(n+l+m)\times(n+l+m)}$ is the **KKT Jacobian** (cf. Primal-Dual Newton!)

$$\frac{\partial}{\partial_{x\kappa\lambda}} r = \begin{pmatrix} \nabla^2[f + \lambda^\top g + \kappa^\top h] & \partial_x h^\top & \partial_x g^\top \\ \partial_x h & 0 & 0 \\ \text{diag}(\lambda)\partial_x g & 0 & \text{diag}(g) \end{pmatrix}$$

– The vector $\frac{\partial}{\partial\theta} r \in \mathbb{R}^{n+l+m}$ describes how the KKT residual depends on θ:

$$\frac{\partial}{\partial\theta} r = \begin{pmatrix} \partial_\theta \nabla[f + \lambda^\top g + \kappa^\top h] \\ \partial_\theta h \\ \text{diag}(\lambda)\partial_\theta g \end{pmatrix}$$

- E.g., for a small variation $(\theta - \theta^*)$, the new optimum is (in linear approx.) at

$$(x, \kappa, \lambda) = (x^*, \kappa^*, \lambda^*) - [\frac{\partial}{\partial_{x\kappa\lambda}} r]^{\text{-1}} \ \frac{\partial}{\partial\theta} r \ (\theta - \theta^*)$$

Example

- Assume $\phi(x;\theta)$ is a NN with parameters $\theta \in \mathbb{R}^d$, inputs $x \in \mathbb{R}^n$, outputs $\phi(x;\theta) \in \mathbb{R}^o$
- For given θ, a Newton method converges to $x^* = \mathrm{argmin}_x \, \phi(x;\theta)^2$
  (We assume a least squares form $f(\theta, x) = \phi(x;\theta)^2$, it could be $o = 1$)
- What is $\frac{dx^*}{d\theta} = \frac{\partial}{\partial\theta} F$?
- Since we have no κ, λ here, we have

$$\frac{\partial}{\partial\theta} F = -[\tfrac{\partial}{\partial x} r]^{\text{-}1} \, \frac{\partial}{\partial\theta} r$$
$$\frac{\partial}{\partial x} r = \nabla^2 f \ , \qquad \frac{\partial}{\partial\theta} r = \partial_\theta \nabla f$$
$$\frac{\partial}{\partial\theta} F = -[\nabla^2 f]^{\text{-}1} \, \partial_\theta \nabla f$$

where we could approximate $\nabla^2 f(x) \approx 2J^\top J$, with the NN's Jacobian $J = \partial_x \phi(x;\theta)$.

Switching Constraints Example

- For $x \in \mathbb{R}$, Consider the problem

$$\min_x (x-\theta)^2 \text{ s.t. } x \geq 0 \ .$$

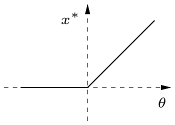
What is the implicit function $F(\theta) = x^*$?

Switching Constraints Example

- For $x \in \mathbb{R}$, Consider the problem

$$\min_x (x-\theta)^2 \text{ s.t. } x \geq 0 \; .$$

What is the implicit function $F(\theta) = x^*$?



$$F(\theta) = x^* = \max\{0, \theta\}$$

which is non-differentiable at $\theta = 0$.

Limitation – Constraint Activity Switching

- Note that the KKT residual $r(\theta, x, \kappa, \lambda) = 0$ neglects the conditions $g(\theta, x) \leq 0, \lambda \geq 0$
- The Implicit Function Theorem assumes $r \in C^1$ and $\det \partial_{x\kappa\lambda} r \neq 0$, but when constraint activity switches, r changes in a non-differentiable manner.

Limitation – Constraint Activity Switching

- Note that the KKT residual $r(\theta, x, \kappa, \lambda) = 0$ neglects the conditions $g(\theta, x) \leq 0, \lambda \geq 0$
- The Implicit Function Theorem assumes $r \in C^1$ and $\det \partial_{x\kappa\lambda} r \neq 0$, but when constraint activity switches, r changes in a non-differentiable manner.

$\rightarrow$ In a **vicinity** of a solution x^*, κ^*, λ^*, we may assume that constraint activity is stable, the inequalities $g(x) \leq 0, \lambda \geq 0$ remain fulfilled, and that the Jacobian of active constraints have full rank (aka. *constraint qualification assumption*).
THEN, **locally**, the implicit function theorem holds and we have the correct gradient.

- However, in general, constraint activity switches somewhere – then we have a discontinuity in the active constraint Jacobians, and in the implicit function gradient.

Limitation – Constraint Activity Switching

- Note that the KKT residual $r(\theta, x, \kappa, \lambda) = 0$ neglects the conditions $g(\theta, x) \leq 0, \lambda \geq 0$
- The Implicit Function Theorem assumes $r \in C^1$ and $\det \partial_{x\kappa\lambda} r \neq 0$, but when constraint activity switches, r changes in a non-differentiable manner.

$\rightarrow$ In a **vicinity** of a solution x^*, κ^*, λ^*, we may assume that constraint activity is stable, the inequalities $g(x) \leq 0, \lambda \geq 0$ remain fulfilled, and that the Jacobian of active constraints have full rank (aka. *constraint qualification assumption*).
THEN, **locally**, the implicit function theorem holds and we have the correct gradient.

- However, in general, constraint activity switches somewhere – then we have a discontinuity in the active constraint Jacobians, and in the implicit function gradient.

$\Rightarrow$ **NLPs with inequalities are *piece-wise* differentiable!**

Classical Literature: “Sensitivity Analysis”

- Lot’s of classical literature on differentiation through NLP solutions:
  - Ralph & Dempe. **Directional derivatives of the solution of a parametric nonlinear program. 1994**. Research Report.
  - Fiacco & Kyparisis. **Sensitivity analysis in nonlinear programming** under second order assumptions. Lecture Notes in Control and Information Sciences, 74-97, **1985**.
  - Kyparisis. Sensitivity analysis for nonlinear programs and variational inequalities with nonunique multipliers. Mathematics of Operations Research, 15:286–298, 1990.
  - Levy & Rockafellar. Sensitivity of solutions in nonlinear programs with nonunique multiplier. Recent Adv. in Nonsmooth Optimzation: 215-223, 1995

  (More recent publications at NeurIPS (keyword “Differentiable Optimization”) ignore this classical literature.)

Classical Literature: "Sensitivity Analysis"

- The implicit function $F(\theta)$ is also called *quasi-solution mapping:* Assume a parameterized NLP $\mathcal{P}(\theta)$

$$F : \theta \mapsto \{x : \text{KKT hold for } \mathcal{P}(\theta)\}$$

  *"We show **under a standard constraint qualification**, not requiring uniqueness of the multipliers, that the quasi-solution mapping is differentiable in a generalized sense, and we present a formula for its derivative."*

- Constant rank constraint qualification (CRCQ): For each subset of the gradients of the active inequality constraints and the gradients of the equality constraints the rank at a vicinity of x^* is constant.

Conclusions

- We can analyze how changes in the optimization problem translate to changes of the optimium x^*
- Using the KKT Jacobian, we can provide the gradient of x^* w.r.t. problem parameters θ
- We can embed optimization algos in auto-differentiation computation graphs (torch, tensorflow)
- Important implications for Differentiable Physics
- **But:** Gradients can be discontinuous across constraint activations