Enhancing Hypergradients Estimation: A Study of Preconditioning and Reparameterization

Zhenzhang Ye[1,4] **Gabriel Peyré**[2] **Daniel Cremers**[1,4] **Pierre Ablin**[3]
[1]Technical University of Munich [2]CNRS, ENS - PSL University [3]Apple [4]Munich Center for Machine Learning
[1]{zhenzhang.ye, cremers}@tum.de, [2]gabriel.peyre@ens.fr, [3]pierre.ablin@apple.com

Abstract

Bilevel optimization aims to optimize an outer objective function that depends on the solution to an inner optimization problem. It is routinely used in Machine Learning, notably for hyperparameter tuning. The conventional method to compute the so-called hypergradient of the outer problem is to use the Implicit Function Theorem (IFT). As a function of the error of the inner problem resolution, we study the error of the IFT method. We analyze two strategies to reduce this error: preconditioning the IFT formula and reparameterizing the inner problem. We give a detailed account of the impact of these two modifications on the error, highlighting the role played by higher-order derivatives of the functionals at stake. Our theoretical findings explain when super efficiency, namely reaching an error on the hypergradient that depends quadratically on the error on the inner problem, is achievable and compare the two approaches when this is impossible. Numerical evaluations on hyperparameter tuning for regression problems substantiate our theoretical findings.

1 INTRODUCTION

Bilevel optimization, the problem of minimizing an outer function that depends on the solution to an inner problem, has become a standard tool in many areas of machine learning. Typical applications include hyperparameter optimization (Franceschi et al., 2018; Pedregosa, 2016; Bertrand et al., 2020), meta-learning (Finn et al., 2017; Rajeswaran et al., 2019) or neural architecture search (Liu et al., 2018). It is also used to train implicit deep learning models like deep equilibrium models (Bai et al., 2019) or networks with optimization layers (Amos and Kolter, 2017; Blondel et al., 2022). Large-scale bi-level problems are usually solved using the implicit function theorem (IFT) to compute the gradient of the outer problem relying only on an estimated solution of the inner problem. In this paper, we challenge this de facto standard by studying variations around this idea, obtained either by preconditioning the IFT formula or by reparameterization, i.e., doing a change of variables. The fundamental question we tackle is to understand the impact of these new IFT-type formulas on the outer gradient approximation error.

Bilevel Optimization. We study the bilevel program

$$\min_{y\in\mathbb{R}^{d_y}} \; h(y) = g(x^\star(y), y) \quad \text{s.t.} \quad F(x^\star(y), y) = 0 \quad (1)$$

where $g : \mathbb{R}^{d_x} \times \mathbb{R}^{d_y} \to \mathbb{R}$ and $F : \mathbb{R}^{d_x} \times \mathbb{R}^{d_y} \to \mathbb{R}^{d_x}$ are smooth functions. The energy function g is called *outer function*, and the root-finding problem $F(x^\star(y), y) = 0$ in the constraint is called *inner problem*. When $F(x, y) = \nabla_1 f(x, y)$ is the gradient of a convex scalar *inner function* $f(x, y)$ w.r.t x, the inner problem corresponds to the optimization of $f(\cdot, y)$. In the following, we assume for simplicity that $x^\star(y)$ is uniquely defined for each y. If this condition does not hold, we assume a consistent selection strategy for a solution $x^\star(y)$ (Arbel and Mairal, 2022).

IFT Formula. Optimizing over y in Eq. (1) typically requires the gradient of the function h w.r.t. y, called *hypergradient* $\nabla h(y)$. Assuming the Jacobian $\nabla_1 F(x^\star(y), y)$ is invertible, the hypergradient can be computed by using the chain rule and Implicit Function Theorem (IFT) (Krantz and Parks, 2002):

$$\nabla h(y) = \Omega(x^\star(y), y) \quad (2)$$

$$\text{where } \Omega(x, y) \coloneqq \nabla_2 g(x, y) + \Psi(x, y)\nabla_1 g(x, y). \quad (3)$$

Proceedings of the 27th International Conference on Artificial Intelligence and Statistics (AISTATS) 2024, Valencia, Spain. PMLR: Volume 238.

$$\text{and } \Psi(x,y) \coloneqq -[\nabla_2 F(x,y)]^\top [\nabla_1 F(x,y)]^{-1}. \quad (4)$$

Here, $\Psi(x^\star(y), y) = \partial x^\star(y)$ relies on the IFT to compute the Jacobian of the map $y \mapsto x^\star(y)$.

Approximate Inner Resolution. When the exact root $x^\star(y)$ is available, the IFT formula (2) computes exactly $\nabla h(y)$. In practice, one only has access to an approximate root $\hat{x}$, for instance, $\hat{x} = x^k(y)$ can be obtained by running k steps of an iterative resolution method. The fundamental question studied in this paper is to analyze the error $\Omega(\hat{x}, y) - \nabla h(y)$ as a function of the inner problem error $\hat{x} - x^\star(y)$. Several strategies like warm starting (Bai et al., 2022; Thornton and Cuturi, 2023) and amortized learning (Amos et al., 2023) have been proposed to reduce this error $\hat{x} - x^\star(y)$. Finally, Ramzi et al. (2021) uses the Hessian approximation learned by a Quasi-Newton method during the inner problem resolution to have a better estimation of the Jacobian Ψ. Still, as long as this error is non-zero, directly using $\Omega(\hat{x}, y)$ as a proxy for $\nabla h(y)$ leads to an inaccurate estimation of the hypergradient, which could cause an accumulated error when optimizing the function h (Devolder et al., 2014). Even with some simple convex functions, the error $\hat{x} - x^\star(y)$ can be amplified on the hypergradient estimation (Mehmood and Ochs, 2021). We question the direct use of Ω and propose alternate formulas $\tilde{\Omega}$ based on preconditioning or reparameterization which might lower the error $\tilde{\Omega}(\hat{x}, y) - \nabla h(y)$.

Preconditioning and Reparameterization. Many methods accelerate the convergence of $x^k(y)$ toward $x^\star(y)$ by preconditioning each step with a linear mapping (Golub and Van Loan, 2013; Spielman and Teng, 2004). When $F = \nabla_1 f$, the intuition is that this preconditioning should capture the curvature of f, hence it should be close to the inverse of the Hessian of f, which corresponds to Newton's method. Finding an efficient preconditioner is a trade-off between the approximation of the Hessian and the ease of inversion. Another widely used strategy is reparemeterization, i.e., to perform a change of variable $z = \phi(x, y)$ over the inner problem, and perform the inner optimization over the z variable (Salimans and Kingma, 2016; Kingma and Welling, 2013; Moins et al., 2023). From an optimization perspective, reparameterization is closely related to preconditioning, where the preconditioner depends on the Jacobian of $\phi(\cdot, y)$.

Contributions and Paper Organization. In this paper, we propose a unified study of the IFT-type formula to estimate ∇h. We study in particular formulas derived by preconditioning and reparameterization:

- In Sec. 2, we characterize the error of the hypergradient estimation when using $(x^k(y), y)$. The Jacobian of Ω (Eq. (2)) w.r.t. x determines the error decay of estimation. We introduce the concept of super efficiency where the Jacobian is 0, leading to a hypergradient estimation that decays quadratically with the error $\hat{x} - x^\star$.
- In Sec. 3, we analyze the impact of two strategies, preconditioning and reparameterization, on the hypergradient estimation. We describe cases where each strategy achieves super efficiency.
- In Sec. 4, we compare these two strategies in different settings. Our results hint at the superiority of preconditioning, while reparameterization could be a better choice in certain corner cases.
- Sec. 5 presents numerical experiments illustrating this paper's theory.

Related Work. Problem (1) is usually solved with an iterative algorithm, like gradient-based algorithms (Beck, 2017), Newton's method (Boyd and Vandenberghe, 2004), and second order Quasi-Newton methods (Shanno, 1970). Two main approaches can be used to compute the gradient of h: automatic and implicit differentiation. Assume that we have access to an iterative strategy that builds the sequence $x^i(y)$ for $i = 0 \dots k-1$ that converges to $x^\star(y)$, like the power method, and that we use the last iterate x^k as an approximation to $x^\star$. Automatic differentiation (Griewank and Walther, 2008) computes an approximation of $\nabla h(y)$ as $\partial_y h(x^k(y))$, where the differentiation is done through the iterates of the algorithm. It does so by leveraging the chain rule repeatedly to the elementary operations and functions in the reverse mode (Christianson, 1994). Gilbert (1992) analyzes its behavior in the context of the iterative procedure. It has become popular in several bilevel applications (Domke, 2012; Franceschi et al., 2017; Mehmood and Ochs, 2020; Bolte et al., 2022). This approach requires storing each iterate $x^i(y)$ in memory, which makes it impractical for iterative procedures with thousands of iterations. Implicit differentiation (Bengio, 2000) overcomes this drawback, using only the last iterate x^k. It is the approach of choice for problems such as deep equilibrium network (Bai et al., 2019), non-smooth problems (Bolte et al., 2021), and hyperparameter tuning (Franceschi et al., 2018).

The preconditioning strategy is common for optimizing a function (Becker and Fadili, 2012; Pock and Chambolle, 2011). Some typical preconditioners include diagonal preconditioner, incomplete Cholesky factorization (Golub and Van Loan, 2013), and Laplacian preconditioning (Spielman and Teng, 2004). Although various works analyze the convergence of the

technique (Benzi, 2002), the impact of preconditioning on the hypergradient estimation is unclear.

Many methods can be viewed as reparameterization. Salimans and Kingma (2016) introduce a reparameterization of the weight vectors in a neural network to accelerate the training process. Kingma and Welling (2013) apply the reparameterization trick on variational autoencoders to allow the backpropagation on a random node. When optimizing a constrained problem, reparameterization is a common strategy to deal with simple constraints (Jorge and Stephen, 2006). In this work, we study whether reparameterization can improve hypergradient estimation.

Variable	Definition
$\|\cdot\|, \|\cdot\|_{\text{op}}$	Euclidean, operator norm
I_d	$d \times d$ identity matrix
$F_1(\cdot,\cdot)$	Jacobian of F w.r.t the first variable
$F_{11}(\cdot,\cdot)$	Jacobian of F_1 w.r.t. the first variable

Table 1: Notations

Notation. To simplify notations, we use the subscript to denote differentiation w.r.t. that variable, i.e. we use for short $F_1(x,y) = \nabla_1 F(x,y)$, $F_2(x,y) = \nabla_2 F(x,y)$. The second-order derivative of $g(x,y)$ is denoted by $g_{12}(x,y) = \nabla^2_{12} g(x,y) \in \mathbb{R}^{d_x \times d_y}$, similarly for $g_{11}(x,y) \in \mathbb{R}^{d_x \times d_x}$. A detailed table of notations is shown in Table 1.

2 ERROR ANALYSIS AND SUPER EFFICIENCY

In this section, we study the structure of the hypergradient estimation problem. We consider a generic formula $\tilde{\Omega}(x,y)$, where $\tilde{\Omega} : \mathbb{R}^{d_x} \times \mathbb{R}^{d_y} \to \mathbb{R}^{d_y}$ is a function that approximates the hypergradient $\nabla h(y)$. The prototypical example is the IFT formula (3). The most basic requirement on the formula $\tilde{\Omega}$ is that it is *consistent*, which means that it correctly recovers the hypergradient if we set $x = x^\star(y)$.

Definition 1 (Consistency)**.** *A formula $\tilde{\Omega}$ is said to be consistent if it satisfies*

$$\forall y, \quad \tilde{\Omega}(x^\star(y), y) = \nabla h(y).$$

By definition, the IFT formula Ω is consistent. Assuming that $\tilde{\Omega}$ is a smooth C^1 map, one can control the impact of an approximate computation $\hat{x}$ of $x^\star(y)$ by doing a Taylor expansion, as stated in the following proposition.

Proposition 1 (Hypergradient approximation)**.** *If $\tilde{\Omega}$ is C^1 and consistent, then for all $\hat{x}$ and y*

$$\|\tilde{\Omega}(\hat{x}, y) - \nabla h(y)\| \leq C_y \|x^\star(y) - \hat{x}\| + \mathcal{O}(\|x^\star(y) - \hat{x}\|^2),$$
$$\text{where} \quad C_y \coloneqq C_y(\tilde{\Omega}) \coloneqq \|\tilde{\Omega}_1(x^\star(y), y)\|_{\text{op}}.$$

This simple result exposes the fact that, at first order, controlling the estimation error on the hyper-gradient requires the control of $C_y(\tilde{\Omega})$, which is the norm of the Jacobian w.r.t. x of the formula $\tilde{\Omega}$. It is the fundamental quantity that needs to be analyzed to understand the efficiency of a formula. We see that the hypegradient estimation error diminishes with $C_y(\tilde{\Omega})$: a good hypergradient estimator should therefore strive to make this constant as small as possible. Of particular interest is when this term is 0:

Definition 2 (Super efficiency (Ablin et al., 2020))**.** *If $C_y(\tilde{\Omega}) = 0$, the formula $\tilde{\Omega}$ is said to be* super efficient*. Equivalently, according to Prop. 1, the estimation error on the hypergradient has a quadratic decay with respect to the inner problem resolution error.*

The following proposition computes $\tilde{\Omega}_1$ in the case of the IFT formula Ω (Eq. (2)).

Proposition 2 (Jacobian of estimation)**.** *Assuming g and F are smooth, one has*

$$\Omega_1(x,y) = g_{21}(x,y) + \Psi_1(x,y) g_1(x,y) + \Psi(x,y) g_{11}(x,y),$$

$$\begin{aligned} &\text{with } \Psi_1(x,y) = -[F_{12}(x,y)]^\top [F_1(x,y)]^{-1} \\ &+ [[F_1(x,y)]^{-1} F_{11}(x,y) [F_1(x,y)]^{-1} F_2(x,y)]^\top. \end{aligned} \tag{5}$$

Efficiency on the Inner Problem. While we phrase all our results directly in terms of the estimation error of the hypergradient ∇h, the core of our analysis aims at controlling the error on the Jacobian $\partial x^\star(y)$ of the inner variable. The estimation of this Jacobian of interest in itself beyond just bilevel programming. The following proposition shows the relation between the error of the estimator Ω of the hypergradient and the estimator Ψ of the inner problem.

Proposition 3 (IFT efficiency)**.** *One has*

$$C_y(\Omega) \leq \|g_{21} + \partial x^\star(y) g_{11}\|_\infty + \|g_1\|_\infty C_y(\Psi)$$

where $\|H\|_\infty \coloneqq \sup_{x,y} \|H(x,y)\|_{\text{op}}$. Hence, if g is of the form $g(x,y) = ax + m(y)$, then Ω is super-efficient if Ψ is super-efficient. If, furthermore, F is of the form $Ax + M(y)$, then Ω is super-efficient.

Prop. 3 describes how the non-linearity of the outer problem impacts the efficiency of Ω. In general, $C_y(\Omega)$ is not 0; the following section aims to design alternate formulas $\tilde{\Omega}$ so that $C_y(\tilde{\Omega})$ as small as possible.

3 PROPOSED STRATEGIES

We detail here two classes of formula $\tilde{\Omega}$ which are consistent by design, and which hopefully improve the efficiency constant $C_y(\tilde{\Omega}) = \|\tilde{\Omega}_1(x^\star(y), y)\|_{\text{op}}$ over the vanilla IFT formula Ω. These two strategies operate either by directly preconditioning the IFT formula or by applying the IFT to a reparameterized problem.

3.1 Preconditioning

We consider an invertible matrix $P(x, y) \in \mathbb{R}^{d_x \times d_x}$ and perform an update on a given x:

$$\tilde{x} \coloneqq x - P(x, y)^{-1} F(x, y). \tag{6}$$

With a proper choice of $P(x, y)$, $\tilde{x}$ becomes closer to $x^\star(y)$ than x. For instance, if F is the gradient of a convex function, a simple choice for P is a large enough symmetric positive matrix. Therefore, rather than using x, we estimate the hypergradient with $\tilde{x}$:

$$\Omega^P(x, y) \coloneqq \Omega(\tilde{x}, y) = \Omega(x - P(x, y)^{-1} F(x, y), y). \tag{7}$$

Preconditionning Efficiency. We now turn to the analysis of the error of this new estimator, by relating it to the Jacobian Ω_1 of the IFT formula Ω.

Proposition 4 (Preconditioned estimation)**.** *Ω^P is consistent and*

$$\Omega_1^P(x, y) = \Omega_1(\tilde{x}, y) E^P(x, y) \tag{8}$$

where $E^P(x, y) \coloneqq I_{d_x} - [P(x, y)]^{-1} F_1(x, y) + [P(x, y)]^{-1} P_1(x, y) [P(x, y)]^{-1} F(x, y)$.

Proof. We give a sketch proof and the full one is available in the appendix. The root of F means that $F(x^\star(y), y) = 0$. Plugging it into Eq. (6) leads to $x^\star - [P(x^\star(y), y)]^{-1} F(x^\star(y), y) = x^\star$, for any P. With Eq. (7), we have $\Omega^P(x^\star(y), y) = \Omega(x^\star(y), y)$. Deriving Ω^P w.r.t x using the chain rule and using $F(x^\star(y), y) = 0$, we have $\Omega_1^P(x^\star(y), y)$. □

This proposition shows that the improvement brought by preconditioning with respect to the IFT formula is precisely captured by E^P. Importantly, this term nicely simplifies at $x^\star(y)$, giving $E^P(x^\star(y), y) = I_{d_x} - P(x^\star(y), y)^{-1} F_1(x^\star(y), y)$. As long as $\|E^P(x^\star(y), y)\|_{\text{op}} < 1$, preconditioning improves the hypergradient estimation compared to the vanilla IFT. In the trivial case where $P(x, y) \equiv 0$, $E^P(x^\star(y), y) = I_{d_x}$, and $\Omega_1^P(x^\star(y), y) = \Omega_1(x^\star(y), y)$. More generally, reducing Ω_1^P amounts to finding a good approximation to the (inverse) Hessian, which is studied next.

Newton Preconditioners. To achieve super-efficiency, the goal is to design P so that $C_y(\Omega^P) = 0$. This can be achieved by using a Newton-type strategy.

Proposition 5 (Newton-like preconditioner)**.** *For $P(x, y) = F_1(x, y)$, Ω^P is super-efficient.*

Proof. In this case, we have $E^p(x^\star(y), y) = 0$, which implies $\Omega_1^P(x^\star(y), y) = 0$. □

Note that Eq. (6) with the "ideal" preconditioner $P(x, y)$ corresponds to the iterates of Newton's method. A chief advantage of such a strategy is that it is super-efficient independent of the choice of the inner function f and the outer function g. This however comes at a price, because applying this formula requires computing a Hessian and solving a linear system. These operations are computationally expensive and even intractable in large-scale problems. In practice, one usually leverages a cheap approximation of the inverse of F_1. When F_1 is diagonally dominant, an efficient solution, the Jacobi preconditioner, only retains the diagonal of F_1.

3.2 Reparameterization

Another classical way to accelerate optimization algorithms is by *reparameterization*, with a well-chosen change of variable. We denote $x = \phi(z, y)$ a *surjective* change of variable, so that the initial problem (1) is equivalent (in the case where $F = \nabla_x f$ for concreteness) to

$$\min_y h(y) = \tilde{g}(z(y), y) \text{ s.t. } z(y) \coloneqq \arg\min_z \tilde{f}(z, y), \tag{9}$$

where $\tilde{g}(z(y), y) \coloneqq g(\phi(z(y), y), y)$ and $\tilde{f}(z, y) \coloneqq f(\phi(z, y), y)$. Note that we reformulate the bilevel optimization in Eq. (1) with the constraint of a minimization problem for the sake of concreteness. For the general case of a constraint $F(x, y) = 0$, $F(x, y)$ should be changed into $\tilde{F}(z, y) \coloneqq \phi_1(z, y)^\top F(\phi(z, y), y) = 0$. We emphasize that the change of variable should be applied to the inner function f. Although the following discussion only requires ϕ to be surjective, we assume ϕ to be bijective for the sake of simplicity. Its inverse on x is denoted as ϕ^{-1}. A crucial point is that, even though the bilevel program is invariant under this change of variable, the IFT formula $\Omega(x, y)$ is not. In the following, we denote Ω^ϕ the formula obtained by replacing (g, F) by $(\tilde{g}, \tilde{F})$.

Proposition 6. *Ω^ϕ is consistent and*

$$\Omega^\phi(x, y) \coloneqq g_2(x, y) + \Psi^\phi(x, y) g_1(x, y), \tag{10}$$

$$\Psi^\phi(x, y) \coloneqq \phi_2(z, y)^\top - U^\phi(x, y)^\top [V^\phi(x, y)]^{-1}, \tag{11}$$

$$U^\phi(x, y) \coloneqq F_2 + F_1 \phi_2 + \phi_1^{-1}(\phi_{21} F), \tag{12}$$

$$V^\phi(x, y) \coloneqq \phi_1^{-\top}(\phi_{11} F) \phi_1^{-1} + F_1. \tag{13}$$

with $z = \phi^{-1}(x, y)$. *Additionally, denoting* $x^\star := x^\star(y)$:

$$\Omega_1^\phi(x^\star, y) = D(y) + \Psi_1^\phi(x^\star, y) g_1(x^\star, y), \tag{14}$$

$$D(y) := g_{21}(x^\star, y) + [\partial x^\star]^\top g_{11}(x^\star, y), \tag{15}$$

$$\Psi_1^\phi(x^\star, y) = \Psi_1(x^\star, y) + C^\phi(y), \tag{16}$$

$$C^\phi(y) = W^\phi(y) + S^\phi(y) + T^\phi(y), \tag{17}$$

$$\begin{aligned} W^\phi(y) &:= -F_1^{-1} \phi_1^{-\top} \phi_{12}^\top F_1, \\ S^\phi(y) &:= [\phi_1^{-\top}(\phi_{11} F_1)\phi_1^{-1} F_1^{-1} \phi_2]^\top, \\ T^\phi(y) &:= [F_1^{-1}(\phi_1^{-\top}(\phi_{11} F_1)\phi_1^{-1}) F_1^{-1} F_2]^\top. \end{aligned} \tag{18}$$

Proof. We omitted (x, y), (z, y) in the right side of Eq. (12)- (13) and $(x^\star, y)$, $(z^\star, y)$ in the right side of Eq. (18) for simplicity. A more detailed derivation can be found in the Supplementary. $g(\phi(z(y), y), y)$ contains three ys. The derivative w.r.t. the latter two can be computed by using the chain rule directly. The derivative w.r.t. the first y requires the IFT. The gradient of $\tilde{f}(z, y)$ w.r.t. z corresponds to a new root-finding problem. Viewing it as a function on $(z^\star(y), y)$, we can apply IFT to get the derivative of the map $y \mapsto z^\star(y)$. Using this derivative and the chain rule, we have $\Psi^\phi(x, y)$. Optimality of $x^\star(y)$ reads $F(x^\star, y) = 0$. Plugging it into Eq. (11), we have $U = F_2 + F_1\phi_2$ and $V = F_1$. Computing $\Psi^\phi(x^\star, y)$ by these two equations yields $\Psi^\phi(x^\star, y) = \Psi(x^\star, y)$. Therefore, we have Ω^ϕ is consistent. The Jacobian of $\Omega^\phi(x, y)$ w.r.t. x follows the chain rule directly. □

Remark 1 (Identity map)**.** *Despite the complexity of these expressions, a sanity check is to verify that if* $\phi(z, y) = z$, *then the above formulae simplify to the IFT ones, since* $\Omega^\phi = \Omega$. *Indeed, in that case,* $\phi_2(z, y) = \phi_{21}(z, y) = \phi_{11}(z, y) = 0$, *so that in Eq.* (11), *one has* $\Psi^\phi(x, y) = \Psi(x, y)$ *for any* (x, y).

Super-Efficient Reparameterization in 1-D. Using Prop. 6, one can seek for a change of variable ϕ in order to minimize $C_y(\Omega^\phi)$. In particular, it turns out that achieving super-efficiency is equivalent to solving a high-dimensional second-order partial-differential equation in ϕ, which, to the best of our knowledge, has no explicit solution. The following proposition shows this second-order differential equation in a simple scalar case.

Proposition 7. *Assume* $x, y \in \mathbb{R}$ *and that* $g(x, y)$ *and* $F(x, y)$ *are linear w.r.t.* x *but arbitrary on* y. *Then* Ω^ϕ *is super-efficient if and only if for all* y,

$$\frac{\phi_{12}}{\phi_1} - \frac{\phi_2 \phi_{11}}{[\phi_1]^2} - \frac{F_2 \phi_{11}}{F_1 [\phi_1]^2} = \frac{g_{12}}{g_1} - \frac{F_{12}}{F_1} \tag{19}$$

where $\phi_{12} = \phi_{12}(z^\star(y), y)$ *(and similarly for other terms).*

In the (admittedly singular) case where the outer function is affine, the above formulation can be leveraged to construct super-efficient reparameterizations $\phi(z)$ depending only on the variable z.

Proposition 8. *If* g *is affine on* x *(see Prop. 3), then super-efficient reparameterizations* $\phi(z, y) = \phi_0(z)$ *exist and define locally a 2-parameters family of maps. If furthermore* F *is linear of* x, *i.e.* $F(x, y) = a(y)x + b$, *where* $a : \mathbb{R} \to \mathbb{R}$ *and* $b \in \mathbb{R}$, *these super-efficient maps are of the form* $\phi_0(z) = \alpha e^{\beta z}$ *for* $(\alpha, \beta) \in \mathbb{R}^2$.

Proof. Eq. (19) boils down to a second-order ODE of a scalar variable on $\phi_0(z)$. Under the smoothness hypothesis on (g, F), one can apply the Cauchy-Lipschitz theorem to ensure the local existence of a super-efficient change of variable. In the simple case where F is affine, the equation is simple enough to be solved, giving the advertised formula for ϕ. □

If one drops the constraint that $\phi(z, y)$ only depends on z, then another super-efficient reparameterization is $\phi(z, y) = z/a(y)$, which coincides with the preconditioning step.

3.3 Separable Localized Reparameterizations

Localized Reparametrizations. A difficulty with the computation of a reparameterized formula Ω^ϕ is the necessity to be able to compute the inverse map $\phi^{-1}(\cdot, y)$. To allow for easily inversible maps, we introduce "localized" changes of variable of the form $\phi(z, y) = \psi_{x,\bar{y}}(z, y)$ which depend on extra fixed parameters $(x, \bar{y})$. The resulting localized formula is then

$$\Omega^\psi_{\text{loc}}(x, y) := \Omega^{\psi_{x,y}}(x, y).$$

Note that while it leverages at each (x, y) a change of variable formula, it is not globally an estimator of the form Ω^ϕ for some fixed ϕ. The following section highlights its versatility in the context of the computation of separable change of variables. Despite being more general, the following proposition shows that the computation of its efficiency constant C_y still boils down to the one of a classical change of variable.

Proposition 9. *The estimator* Ω^ψ_{loc} *is consistent and one has*

$$C_y(\Omega^\psi_{\text{loc}}) = C_y(\Omega^{\psi_{x^\star(y),y}}).$$

Separable Localized Reparameterization. Even with a localized change of variable, designing efficient change of variable remains difficult, even in the 1D case as shown in Eq. (19). To ease this task, inspired by the separation of variables used while solving PDE, we relax this problem by assuming a separable form for $\psi_{x,\bar{y}}(z, y)$:

$$\psi_{x,\bar{y}}(z, y) = R(x, y) Q(z, \bar{y}) + x, \tag{20}$$

where $R(x,y) \in \mathbb{R}^{d_x \times d_x}$ and $Q(z,\bar{y}) \in \mathbb{R}^{d_x}$. To ensure that $\psi_{x,\bar{y}}(z,y)$ is bijective, we impose that $R(x,y)$ is invertible and $Q(\cdot,\bar{y})$ is bijective. The following proposition explicitly gives the quantities involved in Prop. 6.

Proposition 10. *Let $\phi = \psi_{x,\bar{y}}$ as in Eq. (20), one has:*

$$W^\phi(y) = -F_1^{-1}R^{-\top}Q_1^{-\top}Q_1R_2^\top F_1,$$
$$S^\phi(y) = [R^{-\top}Q_1^{-\top}(Q_{11}RF_1)Q_1^{-1}R^{-1}F_1^{-1}(QR_2)]^\top,$$
$$T^\phi(y) = [F_1^{-1}(R^{-\top}Q_1^{-\top}(Q_{11}RF_1)Q_1^{-1}R^{-1})F_1^{-1}F_2]^\top,$$

for any $(x,\bar{y})$, where $R = R(x,y)$, $Q = Q(z^\star(y),\bar{y})$ with $\psi_{x,\bar{y}}(z^\star(y),y) = x^\star(y)$ (and similarly for other terms).

This shows that the search for super-efficient separable parameterization leads to a simpler differential equation on (R,Q) than the original equation on ϕ. However, it is still non-linear and there exists no analytical solution in general. We now show that this efficiency is however always improving as one approaches a Newton-type class of changes of variables and the outer problem is close to being affine.

Efficiency of Ω^ψ_{loc}. Mimicking the preconditioner studied in Prop. 5, we now show that a local change of variable based on a Newton step defines a super-efficient reparameterization. Note however that, in contrast to Prop. 5, in this case, super-efficiency is only possible in the case of an affine outer problem (so it only improves the efficiency of the inner problem estimation).

Proposition 11 (Newton-like reparameterization)**.** *We assume g is of the form $g(x,y) = ax + m(y)$ (see Prop. 3 for a discussion). Let $\psi_{x,\bar{y}}$ as in Eq. (20) and let $F(x,y)$ be bijective on x for all y. For $R(x,y) = [F_1(x,y)]^{-1}$, $Q(z,\bar{y}) = -F(z,\bar{y})$, Ω^ψ_{loc} is super efficiency.*

Proof. The super efficiency requires to examine $C_y(\Omega^\psi_{\text{loc}})$. Because $(x,\bar{y})$ are extra parameters not variables, Ω^ψ_{loc} has the same formula as in Prop. 6. While computing its Jacobian, using the fact of $F(x^\star(y),y) = 0$, we can still get the same results as in Prop. 10. Plugging the special choice of R and Q, we get $\Psi_1^{\psi_{x^\star(y),y}}(x^\star(y),y) = 0$. $C_y(\Omega^\psi_{\text{loc}})$ is thus 0 because g is affine. □

Remark 2. *The above proposition reveals that the reparameterization is never equivalent to the preconditioning strategy. In the case of a quadratic inner problem, the Newton-like preconditioner in Prop. 5 achieves super-efficiency with any g, while the Newton-like reparameterization requires an assumption on g.*

This proposition should be understood as providing heuristic guidance to design R and Q, which we exploit in the numerical simulations in Sec. 5. The following more general proposition states that for generic localized separable changes of variable, the efficiency constant is upper-bounded by error terms measuring how far the change of variable is different from the above-mentioned Newton-type reparameterization.

Proposition 12. *Let $x^\star := x^\star(y)$ and $\psi_{x^\star,y}$ be defined as in Eq. (20) and $z^\star$ satisfy $\psi_{x^\star,y}(z^\star,y) = x^\star$. Denoting $E^Q(y) := Q(z^\star,y) + F(x^\star,y)$, $E^{Q_1}(y) := Q_1(z^\star,y) + F_1(x^\star,y)$, $E^{Q_{11}}(y) := Q_{11}(z^\star,y) + F_{11}(x^\star,y)$, $E^R(y) := R(x^\star,y) - [F_1(x^\star,y)]^{-1}$, $E^{R_2}(y) := R_2(x^\star,y) + [F_1(x^\star,y)]^{-1}F_{21}(x^\star,y)[F_1(x^\star,y)]^{-1}$. We have that $C_y(\Psi^\psi_{\text{loc}}) = \mathcal{O}(\|E^Q\|_{\text{op}}, \|E^{Q_1}\|_{\text{op}}, \|E^{Q_{11}}\|_{\text{op}}, \|E^R\|_{\text{op}}, \|E^{R_2}\|_{\text{op}})$.*

Note that this proposition shows (R,Q) should be close to Newton-type functionals, but their higher-order derivatives should also be close, which highlights the difficulty of designing efficient changes of variables.

4 COMPARISON BETWEEN METHODS

As it should be clear from the above analysis, super-efficiency is out of reach for cases of practical interest. We thus focus on comparing the efficiency constant C_y of the different strategies. We first focus our analysis on the preconditioning Ω^P with $P(x,y)$ and the reparameterization Ω^ϕ with an arbitrary smooth and bijective functional $\phi(z,y)$. The following proposition, which is leveraged in special cases below, gives a general formula to compare the efficiency constants.

Proposition 13 (Comparison of the two methods)**.** *Let ϕ be smooth and bijective, one has*

$$[C_y(\Omega^\phi)]^2 - [C_y(\Omega^P)]^2 \geq \langle U_+ v_P, U_- v_P \rangle \tag{21}$$
$$[C_y(\Omega^P)]^2 - [C_y(\Omega^\phi)]^2 \geq \langle V_+ v_\phi, V_- v_\phi \rangle \tag{22}$$

where, for E^P as in Eq. (8), Ψ_1^ϕ and D as in Eq. (15),

$$U_\pm := D \pm DE^P + \Psi_1^\phi g_1 \pm \Psi_1 g_1 E^P,$$
$$V_\pm := DE^P \pm D + \Psi_1 g_1 E^P \pm \Psi_1^\phi g_1,$$
$$v_\omega := \operatorname*{arg\,max}_{\|u\|=1} \|\Omega_1^\omega(x^\star(y),y)u\| \text{ for } \omega \in \{P,\phi\},$$

The dependencies on $(x^\star(y),y)$ on the right side are omitted.

Asymptotic Analysis of Preconditioning Superiority. As a consequence, the following proposition shows that as the preconditioning quality δ is small

enough, then the preconditioning formula Ω^P is necessarily better than the reparameterization one Ω^ϕ. This should not be surprising since when $\delta = 0$, the preconditioning is super-efficient, while the reparameterization is not necessarily super-efficient.

Proposition 14. *For* $\delta := \|P(x^\star(y), y) - F_1(x^\star(y), y)^{-1}\|_\infty$, *we have*

$$[C_y(\Omega^\phi)]^2 - [C_y(\Omega^P)]^2 \geq \|(D + \Psi_1^\phi g_1)v_P\|^2 + o(\delta),$$

all the terms being evaluated at $(x^\star(y), y)$ *and* v_P *as defined in Prop. 13.*

Note that the term $(D + \Psi_1^\phi g_1)v_P$ does not cancel in general, which shows that the preconditioning strategy should be favored when a suitable preconditioner (that makes δ small) is available. A suitable preconditioner can also be leveraged in localized reparameterization by $\psi_x(z, y) = [P(x, y)]^{-1}z$. However, the improvement from the reparameterization is less than the preconditioning because of $(D + \Psi_1^\phi g_1)v_P$.

Asymptotic Analysis of Separable Localized Reparameterization Superiority. On the other hand, if P does not approximate very well F_1^{-1}, it might be possible that the reparameterization Ω^ϕ is better than Ω^P if ϕ is well designed. Extreme cases were already given for 1-D problems in Prop. 8, where super-efficient changes of variables are detailed. Additionally, in the case of the separable localized reparameterization, the following proposition shows that if g is close to being affine on x (as already analyzed in Prop. 3), then the reparameterization Ω^ψ_{loc} could be a better choice than the preconditioning Ω^P.

Proposition 15. *For* $\sigma := \|g_1(x^\star(y), y)\|_\infty C_y(\Psi^\psi_{\text{loc}})$,

$$\begin{aligned}&[C_y(\Omega^P)]^2 - [C_y(\Psi^\psi_{\text{loc}})]^2\\ \geq&\|(D + \Psi_1 g_1)E^P v_\phi\|^2 - \|Dv_\phi\|^2 + o(\sigma),\end{aligned}$$

where v_ϕ *is defined in Prop. 13 with* $\phi = \Psi^\psi_{\text{loc}}$.

In conclusion, the preconditioning strategy outperforms in general as long as a valuable preconditioner is available. Otherwise, a well-designed reparameterization could be a better choice when approximating F_1^{-1} is difficult.

5 NUMERICAL EXPERIMENTS

We illustrate our findings on regression and classification supervised learning problems. Bilevel programming is used to compute hyper-parameters y controlling the regularization function. The inner and outer problems are of the form:

$$\begin{aligned}g(x) &= L(A_{\text{val}}x, b_{\text{val}}),\\ f(x, y) &= L(A_{\text{tr}}x, b_{\text{tr}}) + \mathcal{R}(x, y),\end{aligned} \tag{23}$$

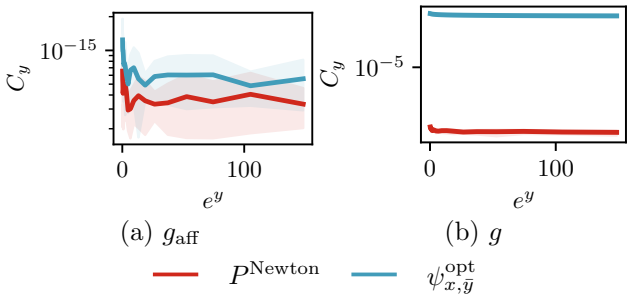


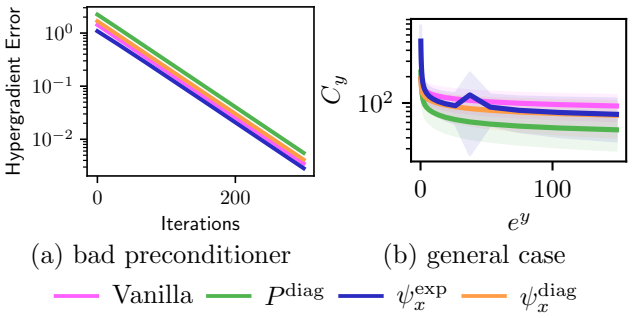
Figure 1: Compare P^{Newton} from Prop. 5 and $\psi^{\text{opt}}_{x,\bar{y}}$ from Prop. 11 on ridge regression but with different outer problems. We show the efficiency constant C_y in log space under different y. (a) When the outer problem is affine, both strategies can achieve a small efficiency constant C_y around machine accuracy. (b) When the outer problem is quadratic, the Newton preconditioner achieves the super efficiency while the $\psi^{\text{opt}}_{x,\bar{y}}$ has a large constant C_y.

with $L(z, b) = \sum_i \ell(z_i, b_i)$, and the loss ℓ depends on the task (regression or classification). Here $A_{\text{tr}} \in \mathbb{R}^{M\times d_x}$, $A_{\text{val}} \in \mathbb{R}^{N\times d_x}$ are the train and test design matrix respectively, $b_{\text{tr}} \in \mathbb{R}^M$ and $b_{\text{val}} \in \mathbb{R}^N$ are the train and test labels (restricted to $\{-1, 1\}$ for classification). The coefficients $x \in \mathbb{R}^{d_x}$ are the weight parameters of the predictor. The value of y determines the structure of $F_1(x, y)$, for instance large y lead to $F_1(x, y)$ being diagonally dominant. The regularization functional is a ridge penalty $\mathcal{R}(x, y) := \frac{1}{2}\sum_i^{d_x} e^{y_i}x_i^2$, so that y introduces a feature-dependent penalization (Pedregosa, 2016).

Inspired by Prop. 5, the first two strategies we consider are Newton preconditioner $P^{\text{Newton}}(x, y) := F_1(x, y)$ and the diagonal preconditioner $P^{\text{diag}}(x, y) := \text{diag}(F_1(x, y))$. Similarly to Prop. 8, we also consider an exponential reparameterization defined as $\psi^{\text{exp}}_x(z) := \text{sign}(x)\exp(z)$. The last strategy is a separable diagonal reparameterization $\psi^{\text{diag}}_x(z, y) := [\text{diag}(F_1(x, y))]^{-1}z$. We run a gradient descent $x_k = x_{k-1} - \tau_k\nabla_x f(x_{k-1}, y)$ with a proper step size $\tau_k > 0$ to attain an approximate root x^k after k steps. All the experiments are run with Jax (Bradbury et al., 2018) and can be found in `https://github.com/zhenzhang-ye/enhance_hypergradient`.

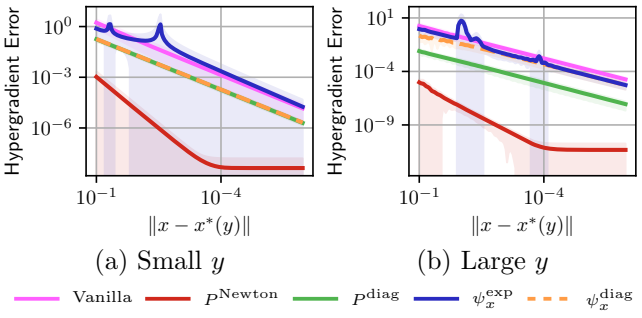
5.1 Ridge Regression

The first problem we consider is a regularized ridge regression, obtained using $\ell(z, b) = (z - b)^2$ in Eq. (23). The design matrix A_{tr} and labels b_{tr} are from the dataset **mpg** in LIBSVM (Chang and Lin, 2011), which consists of $M = 392$ data and $d_x = 7$ features. The $A_{\text{val}} \in \mathbb{R}^{M\times d_x}$ and $b_{\text{val}} \in \mathbb{R}^M$ in the outer problem



(a) bad preconditioner (b) general case

Figure 2: Compare P^{diag}, ψ_x^{exp}, ψ_x^{diag} on ridge regression with the outer problem g. (a) We show the hypergradient errors of different strategies in log space over the number of iterations when having a bad preconditioner. It turns out that ψ_x^{exp} could be a better choice in this setting. (b) We show the efficiency constant C_y of each strategy in log space under different y. Although reparameterization could perform better in some cases, P^{diag} in general is the best choice.



(a) Small y (b) Large y

Figure 3: Comparison of different strategies on hypergradient error in log space over approximated root for logistic regression. P^{Newton} always achieve the super efficiency. (a) With a small y, the performances of P^{diag} and ψ_x^{diag} are nearly the same. ψ_x^{exp} performs worse than the vanilla one in some situations. (b) The performance of P^{diag} improves thanks to the large y which leads to a diagonally dominated F_1. The performances of two reparameterizations are the similar, both better than the vanilla one.

is randomly generated. The optimal $x^\star(y)$ is computed using the linear solver from Jax. To directly assess the estimation error on $\partial x^\star(y)$ (see Prop. 3), we also consider a special case where the outer problem is affine, denoted by g_{aff}.

Since F is bijective in this case, the reparameterization $\psi_{x,\bar{y}}^{\text{opt}}(z, y) \coloneqq -[F_1(x, y)]^{-1}F(z, \bar{y}) + x$ from Prop. 11 is readily applicable. We first compare it to P^{Newton} under different values of y. We ran the experiments 10 times for each y to remove the effect of randomness. Fig. 1(a) shows that when the outer problem is g_{aff}, the two strategies both achieve very small efficiency constant C_y. This agrees with our finding that both of the algorithms can achieve super efficiency if D is 0. On the other hand, the advantage of Newton's preconditioner shows up when having the general g. As shown in Fig. 1(b), the efficiency constant C_y of P^{Newton} is independent of the outer problem, while $\psi_{x,\bar{y}}^{\text{opt}}$ fails to attain a small constant C_y.

To cover cases where F_1^{-1} is unavailable or expensive to apply, we compare P^{diag}, ψ_x^{exp} and ψ_x^{diag}. In Fig. 2 (a), we show a special case where ψ_x^{exp} works the best even with an arbitrary (non-affine) outer problem g. The reason is that the F_1 is not diagonally dominated, and the diagonal preconditioner is not efficient anymore. To this end, we compare the three strategies with different y and show the results in Fig. 2(b). All three strategies improve the hypergradient estimator compared to the vanilla one when the outer problem is quadratic g. The average performance of P^{diag} is the best, but there can be cases where the two reparameterizations are the best choices.

5.2 Logistic Regression

We now consider classification using logistic regression, using $\ell(z, b) = \log(1 + \exp(-zb))$. The design matrices and labels are from **liver-disorder** in LIBSVM with 145 training and 200 testing data. There are 5 features and 2 classes of each data. We ran all experiments 10 times with different random seeds. The root $x^\star(y)$ is attained by the `minimize` function from Jax with a tolerance of 10^{-15}.

To assess the impact of the scale of y on F_1 (and hence on the estimator efficiency), we compare the strategies with y drawn from two different uniform distributions. The first notable result in Fig. 3 is that P^{Newton} achieves super efficiency in both cases. This comes without a surprise because our finding shows that the performance of Newton's preconditioner is independent of F_1 and the problems. Additionally when y is generated over $[-1, 1)$ in Fig. 3(a), the F_1 is away from diagonally dominated. It makes the performances of P^{diag} and ψ_x^{diag} nearly the same. The exponential reparameterization ψ_x^{exp} performs even worse than the vanilla one. However, when y generated over $[3, 6)$ becomes larger, all the strategies outperform the vanilla one. The difference between P^{diag} and ψ_x^{diag} is expected because the diagonal preconditioner leads to a small δ in Prop. 14 in this case. For larger values of y, the performance of ψ_x^{exp} is improved as well. A plausible explanation is that larger y makes F_1 close to a diagonal matrix, in which case the exponential reparameterization achieves super efficiency.

Conclusion

In this paper, we have developed an in-depth local analysis of implicit differentiation for bilevel programming. Of particular interest are detailed expressions of the efficiency constant C_y of several variations around the vanilla IFT formula. This highlights the key challenge in designing an efficient hypergradient formula. On the theoretical side, the question of the existence (let alone the computation) of super-efficient reparameterization is still mostly open. It corresponds to highly non-linear partial differential equations. Better exploiting the connection between reparametrization and preconditioning could also be fruitful, for instance, in designing parametric formulas that could be adapted to specific machine-learning problems.

Acknowledgement

ZY and DC were supported by the ERC Advanced Grant SIMULACRON. The work of GP was supported by the European Research Council (ERC project NORIA) and the French government under management of Agence Nationale de la Recherche as part of the "Investissements d'avenir" program, reference ANR-19-P3IA-0001 (PRAIRIE 3IA Institute). The work was done during ZY's internship at ENS - PSL University. We thank Zaccharie Ramzi for the fruitful discussions.

References

P. Ablin, G. Peyré, and T. Moreau. Super-efficiency of automatic differentiation for functions defined as a minimum. In *International Conference on Machine Learning*, pages 32–41. PMLR, 2020.

B. Amos and J. Z. Kolter. Optnet: Differentiable optimization as a layer in neural networks. In *International Conference on Machine Learning*, pages 136–145. PMLR, 2017.

B. Amos et al. Tutorial on amortized optimization. *Foundations and Trends® in Machine Learning*, 16 (5):592–732, 2023.

M. Arbel and J. Mairal. Non-convex bilevel games with critical point selection maps. *Advances in Neural Information Processing Systems*, 35:8013–8026, 2022.

S. Bai, J. Z. Kolter, and V. Koltun. Deep equilibrium models. *Advances in Neural Information Processing Systems*, 32, 2019.

S. Bai, Z. Geng, Y. Savani, and J. Z. Kolter. Deep equilibrium optical flow estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 620–630, 2022.

A. Beck. *First-order methods in optimization*. SIAM, 2017.

S. Becker and J. Fadili. A quasi-newton proximal splitting method. *Advances in neural information processing systems*, 25, 2012.

Y. Bengio. Gradient-based optimization of hyperparameters. *Neural computation*, 12(8):1889–1900, 2000.

M. Benzi. Preconditioning techniques for large linear systems: a survey. *Journal of computational Physics*, 182(2):418–477, 2002.

Q. Bertrand, Q. Klopfenstein, M. Blondel, S. Vaiter, A. Gramfort, and J. Salmon. Implicit differentiation of lasso-type models for hyperparameter optimization. In *International Conference on Machine Learning*, pages 810–821. PMLR, 2020.

M. Blondel, Q. Berthet, M. Cuturi, R. Frostig, S. Hoyer, F. Llinares-López, F. Pedregosa, and J.-P. Vert. Efficient and modular implicit differentiation. *Advances in neural information processing systems*, 35:5230–5242, 2022.

J. Bolte, T. Le, E. Pauwels, and T. Silveti-Falls. Nonsmooth implicit differentiation for machine-learning and optimization. *Advances in neural information processing systems*, 34:13537–13549, 2021.

J. Bolte, E. Pauwels, and S. Vaiter. Automatic differentiation of nonsmooth iterative algorithms. *Advances in Neural Information Processing Systems*, 35:26404–26417, 2022.

S. P. Boyd and L. Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

J. Bradbury, R. Frostig, P. Hawkins, M. J. Johnson, C. Leary, D. Maclaurin, G. Necula, A. Paszke, J. VanderPlas, S. Wanderman-Milne, and Q. Zhang. JAX: composable transformations of Python+NumPy programs, 2018. URL `http://github.com/google/jax`.

C.-C. Chang and C.-J. Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

B. Christianson. Reverse accumulation and attractive fixed points. *Optimization Methods and Software*, 3 (4):311–326, 1994.

O. Devolder, F. Glineur, and Y. Nesterov. First-order methods of smooth convex optimization with inexact oracle. *Mathematical Programming*, 146:37–75, 2014.

J. Domke. Generic methods for optimization-based modeling. In *Artificial Intelligence and Statistics*, pages 318–326. PMLR, 2012.

C. Finn, P. Abbeel, and S. Levine. Model-agnostic meta-learning for fast adaptation of deep networks.

In *International conference on machine learning*, pages 1126–1135. PMLR, 2017.

L. Franceschi, M. Donini, P. Frasconi, and M. Pontil. Forward and reverse gradient-based hyperparameter optimization. In *International Conference on Machine Learning*, pages 1165–1173. PMLR, 2017.

L. Franceschi, P. Frasconi, S. Salzo, R. Grazzi, and M. Pontil. Bilevel programming for hyperparameter optimization and meta-learning. In *International conference on machine learning*, pages 1568–1577. PMLR, 2018.

J. C. Gilbert. Automatic differentiation and iterative processes. *Optimization methods and software*, 1(1): 13–21, 1992.

G. H. Golub and C. F. Van Loan. *Matrix computations*. JHU press, 2013.

A. Griewank and A. Walther. *Evaluating derivatives: principles and techniques of algorithmic differentiation*. SIAM, 2008.

N. Jorge and J. W. Stephen. *Numerical optimization*. Spinger, 2006.

D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

S. G. Krantz and H. R. Parks. *The implicit function theorem: history, theory, and applications*. Springer Science & Business Media, 2002.

H. Liu, K. Simonyan, and Y. Yang. Darts: Differentiable architecture search. *arXiv preprint arXiv:1806.09055*, 2018.

S. Mehmood and P. Ochs. Automatic differentiation of some first-order methods in parametric optimization. In *International Conference on Artificial Intelligence and Statistics*, pages 1584–1594. PMLR, 2020.

S. Mehmood and P. Ochs. Differentiating the value function by using convex duality. In *International Conference on Artificial Intelligence and Statistics*, pages 3871–3879. PMLR, 2021.

T. Moins, J. Arbel, S. Girard, and A. Dutfoy. Reparameterization of extreme value framework for improved bayesian workflow. *Computational Statistics & Data Analysis*, page 107807, 2023.

F. Pedregosa. Hyperparameter optimization with approximate gradient. In *International conference on machine learning*, pages 737–746. PMLR, 2016.

T. Pock and A. Chambolle. Diagonal preconditioning for first order primal-dual algorithms in convex optimization. In *2011 International Conference on Computer Vision*, pages 1762–1769. IEEE, 2011.

A. Rajeswaran, C. Finn, S. M. Kakade, and S. Levine. Meta-learning with implicit gradients. *Advances in neural information processing systems*, 32, 2019.

Z. Ramzi, F. Mannel, S. Bai, J.-L. Starck, P. Ciuciu, and T. Moreau. Shine: Sharing the inverse estimate from the forward pass for bi-level optimization and implicit models. *arXiv preprint arXiv:2106.00553*, 2021.

T. Salimans and D. P. Kingma. Weight normalization: A simple reparameterization to accelerate training of deep neural networks. *Advances in neural information processing systems*, 29, 2016.

D. F. Shanno. Conditioning of quasi-newton methods for function minimization. *Mathematics of computation*, 24(111):647–656, 1970.

D. A. Spielman and S.-H. Teng. Nearly-linear time algorithms for graph partitioning, graph sparsification, and solving linear systems. In *Proceedings of the thirty-sixth annual ACM symposium on Theory of computing*, pages 81–90, 2004.

J. Thornton and M. Cuturi. Rethinking initialization of the sinkhorn algorithm. In *International Conference on Artificial Intelligence and Statistics*, pages 8682–8698. PMLR, 2023.

Checklist

1. For all models and algorithms presented, check if you include:
   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. [Yes] See Sec. 1 and Sec. 3
   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. [Yes] See Sec. 3
   (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. [Yes], we plan to publish code.

2. For any theoretical claim, check if you include:
   (a) Statements of the full set of assumptions of all theoretical results. [Yes]
   (b) Complete proofs of all theoretical results. [Yes] All proofs are provided in the supplementary.
   (c) Clear explanations of any assumptions. [Yes]

3. For all figures and tables that present empirical results, check if you include:
   (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). [Yes]

(b) All the training details (e.g., data splits, hyperparameters, how they were chosen). [N/A]

(c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). [Yes]

(d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). [N/A]

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:

   (a) Citations of the creator If your work uses existing assets. [Yes]

   (b) The license information of the assets, if applicable. [Yes]

   (c) New assets either in the supplemental material or as a URL, if applicable. [N/A]

   (d) Information about consent from data providers/curators. [N/A]

   (e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. [N/A]

5. If you used crowdsourcing or conducted research with human subjects, check if you include:

   (a) The full text of instructions given to participants and screenshots. [N/A]

   (b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. [N/A]

   (c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. [N/A]

Enhancing Hypergradients Estimation: A Study of Preconditioning and Reparameterization Supplementary Materials

Proof of Propositions

Proposition 1 (Hypergradient approximation)**.** *If $\tilde{\Omega}$ is C^1 and consistent, then for all $\hat{x}$ and y*

$$\|\tilde{\Omega}(\hat{x}, y) - \nabla h(y)\| \leq C_y \|x^\star(y) - \hat{x}\| + \mathcal{O}(\|x^\star(y) - \hat{x}\|^2),$$
$$\text{where} \quad C_y \coloneqq C_y(\tilde{\Omega}) \coloneqq \|\tilde{\Omega}_1(x^\star(y), y)\|_{\text{op}}.$$

Proof. The Taylor expansion of $\tilde{\Omega}(\hat{x}, y)$ at $(x^\star(y), y)$ yields:

$$\tilde{\Omega}(\hat{x}, y) = \tilde{\Omega}(x^\star(y), y) + \tilde{\Omega}_1(x^\star(y), y)(x^\star(y) - \hat{x}) + \mathcal{O}(\|x^\star(y) - \hat{x}\|^2).$$

Because $\tilde{\Omega}$ is consistent, we have

$$\begin{aligned}\|\tilde{\Omega}(\hat{x}, y) - \nabla h(y)\| &= \|\tilde{\Omega}_1(x^\star(y), y)(\hat{x} - x^\star(y)) + \mathcal{O}(\|x^\star(y) - \hat{x}\|^2)\| \\ &\leq \|\tilde{\Omega}_1(x^\star(y), y)\|_{\text{op}} \|(x^\star(y) - \hat{x})\| + \mathcal{O}(\|x^\star(y) - \hat{x}\|^2).\end{aligned}$$

□

Proposition 2 (Jacobian of estimation)**.** *Assuming g and F are smooth, one has*

$$\Omega_1(x, y) = g_{21}(x, y) + \Psi_1(x, y) g_1(x, y) + \Psi(x, y) g_{11}(x, y),$$
$$\text{with } \Psi_1(x, y) = -[F_{12}(x, y)]^\top [F_1(x, y)]^{-1} + \left([F_1(x, y)]^{-1} F_{11}(x, y) [F_1(x, y)]^{-1} F_2(x, y)\right)^\top.$$

Proof. Recall that $\Omega(x, y) = g_2(x, y) + \Psi(x, y) g_1(x, y)$ with $\Psi(x, y) = -[F_2(x, y)]^\top [F_1(x, y)]^{-1}$. Using the chain rule to compute the Jacobian w.r.t. x, we have:

$$\Omega_1 = g_{21}(x, y) + \Psi_1(x, y) g_1(x, y) + \Psi(x, y) g_{11}(x, y).$$

Using the fact that the Jacobian of $[F(x)]^{-1}$ is $-[F(x)]^{-1} F_1(x) [F(x)]^{-1}$, we can get $\Psi_1(x, y)$ as defined. □

Proposition 3 (IFT efficiency)**.** *One has*

$$C_y(\Omega) \leq \|g_{21}(x^\star(y), y) + [\partial x^\star(y)]^\top g_{11}\|_\infty + \|g_1(x^\star(y), y)\|_\infty C_y(\Psi)$$

where $\|H\|_\infty \coloneqq \sup_{x,y} \|H(x, y)\|_{\text{op}}$. Hence, if g is of the form $g(x, y) = ax + m(y)$, then Ω is super-efficient if Ψ is super-efficient. If, furthermore, F is of the form $Ax + M(y)$, then Ω is super-efficient.

Proof. Recall the definition of $C_y(\Omega)$ in Prop. 1, we have:

$$\begin{aligned}C_y(\Omega) &= \|\Omega_1(x^\star(y), y)\|_{\text{op}} \\ &= \|g_{21}(x^*(y), y) + \Psi_1(x^*(y), y) g_1(x^*(y), y) + \partial x^\star(y) g_{11}(x^*(y), y)\|_{\text{op}} \\ &\leq \|g_{21}(x^*(y), y) + [\partial x^\star(y)]^\top g_{11}(x^*(y), y)\|_{\text{op}} + \|\Psi_1(x^*(y), y) g_1(x^*(y), y)\|_{\text{op}} \\ &\leq \|g_{21}(x^*(y), y) + [\partial x^\star(y)]^\top g_{11}(x^*(y), y)\|_\infty + \|\Psi_1(x^*(y), y)\|_{\text{op}} \|g_1(x^*(y), y)\|_\infty\end{aligned}$$

If g is affine, we have $g_{21}(x, y) = g_{11}(x, y) \equiv 0$ and the first term becomes 0. If furthermore, $F(x, y) = Ax + M(y)$, $C_y(\Psi) \equiv 0$, which implies $C_y(\Omega) \equiv 0$. □

Proposition 4 (Preconditioned estimation)**.** Ω^P *is consistent and*

$$\Omega_1^P(x,y) = \Omega_1(\tilde{x},y)E^P(x,y),$$

where $E^P(x,y) \coloneqq I_{d_x} - [P(x,y)]^{-1}F_1(x,y) + [P(x,y)]^{-1}P_1(x,y)[P(x,y)]^{-1}F(x,y)$.

Proof. The root of F reads $F(x^\star(y),y) = 0$. Recall the preconditioning step $\tilde{x} \coloneqq x - [P(x,y)]^{-1}F(x,y)$. Plugging $x = x^\star(y)$, we have $\tilde{x} = x^\star(y)$. We thus have $\Omega^P(x^\star(y),y) = \Omega(x^\star(y),y)$.

Because $\Omega^P = \Omega(x - [P(x,y)]^{-1}F(x,y),y)$, using the chain rule and the fact that the Jacobian of $[P(x)]^{-1}$ is $-[P(x)]^{-1}P_1(x)[P(x)]^{-1}$, we have:

$$\Omega_1^P(x,y) = \Omega_1(x - [P(x,y)]^{-1}F(x,y),y)(I_{d_x} - [P(x,y)]^{-1}F_1(x,y) + [P(x,y)]^{-1}P_1(x,y)[P(x,y)]^{-1}F(x,y)).$$

□

Proposition 5 (Newton-like preconditioner)**.** *For* $P(x,y) = F_1(x,y)$, Ω^P *is super-efficient.*

Proof. Note that $F(x^\star(y),y) = 0$ in this case. Thus, we have $E^P(x^*(y),y) = I_{d_x} - [P(x^\star(y),y)]^{-1}F_1(x^\star(y),y) = 0$, which implies $\Omega_1^P(x^\star(y),y) = 0$. □

Proposition 6. Ω^ϕ *is consistent and*

$$\begin{aligned}
\Omega^\phi(x,y) &\coloneqq g_2(x,y) + \Psi^\phi(x,y)g_1(x,y),\\
\Psi^\phi(x,y) &\coloneqq \phi_2(z,y)^\top - U^\phi(x,y)^\top[V^\phi(x,y)]^{-1},\\
U^\phi(x,y) &\coloneqq F_2(x,y) + F_1(x,y)\phi_2(z,y) + [\phi_1(z,y)]^{-1}\phi_{21}(z,y)F(x,y),\\
V^\phi(x,y) &\coloneqq [\phi_1(z,y)]^{-\top}\phi_{11}(z,y)F(x,y)[\phi_1(z,y)]^{-1} + F_1(x,y),
\end{aligned}$$

with $z = \phi^{-1}(x,y)$*. Additionally, denoting* $x^\star \coloneqq x^\star(y)$ *and* $z^\star \coloneqq \phi^{-1}(x^\star,y)$*:*

$$\begin{aligned}
\Omega_1^\phi(x^\star,y) &= D(y) + \Psi_1^\phi(x^\star,y)g_1(x^\star,y),\\
D(y) &\coloneqq g_{21}(x^\star,y) + [\partial x^\star(y)]^\top g_{11}(x^\star,y),\\
\Psi_1^\phi(x^\star,y) &= \Psi_1(x^\star,y) + C^\phi(y),\\
C^\phi(y) &\coloneqq W^\phi(y) + S^\phi(y) + T^\phi(y).
\end{aligned}$$

$$\begin{aligned}
W^\phi(y) &\coloneqq -[F_1(x^\star,y)]^{-1}[\phi_1(z^\star,y)]^{-\top}[\phi_{12}(z^\star,y)]^\top F_1(x^\star,y),\\
S^\phi(y) &\coloneqq \big([\phi_1(z^\star,y)]^{-\top}\phi_{11}(z^\star,y)F_1(x^\star,y)[\phi_1(z^\star,y)]^{-1}[F_1(x^\star,y)]^{-1}\phi_2(z^\star,y)\big)^\top,\\
T^\phi(y) &\coloneqq \big([F_1(x^\star,y)]^{-1}[\phi_1(z^\star,y)]^{-\top}\phi_{11}(z^\star,y)F_1(x^\star,y)[\phi_1(z^\star,y)]^{-1}[F_1(x^\star,y)]^{-1}F_2(x^\star,y)\big)^\top.
\end{aligned}$$

Proof. After the change of variable, we have the inner problem $f(\phi(z,y),y)$ and the outer problem $g(\phi(z(y),y),y)$. Using the chain rule, the hypergradient is then:

$$\begin{aligned}
\nabla h(y) &= \nabla g(\phi(z(y),y),y)\\
&= g_2(\phi(z(y),y),y) + [\partial z(y)]^\top[\phi_1(z(y),y)]^\top g_1(\phi(z(y),y),y) + [\phi_2(z(y),y)]^\top g_1(\phi(z(y),y)).
\end{aligned} \tag{24}$$

Now, we turn to compute $\partial z(y)$. Computing the gradient of the inner problem w.r.t. z and denoting the fixed point as $(z^\star,y)$ with $z^\star \coloneqq z^\star(y)$, we have the new equation:

$$\tilde{F}(z^\star,y) \coloneqq \phi_1(z^\star,y)^\top F(\phi(z^\star,y),y) = 0.$$

Viewing it as a fixed-point equation on $(z^\star,y)$, we can apply IFT to get $\partial z^\star(y)$:

$$\begin{aligned}
\partial z^\star(y) &= -[\tilde{F}_1(z^\star,y)]^{-1}\tilde{F}_2(z^\star,y),\\
\text{where } \tilde{F}_1(z^\star,y) &= \phi_{11}(z^\star,y)F(\phi(z^\star,y),y) + [\phi_1(z^\star,y)]^\top F_1(\phi(z^\star,y),y)\phi_1(z^\star,y),\\
\tilde{F}_2(z^\star,y) &= [\phi_{21}(z^\star,y)]^\top F(\phi(z^\star,y),y) + [\phi_1(z^\star,y)]^\top F_1(\phi(z^\star,y),y)\phi_2(z^\star,y) + [\phi_1(z^\star,y)]^\top F_2(\phi(z^\star,y),y).
\end{aligned}$$

Plugging it back to Eq. (24), we can get $\Omega^\phi(\phi(z^\star(y),y),y)$. Because $\phi(z^\star(y),y)=x^\star(y)$ and ϕ is bijective, we recall the notation that $\phi(x^\star(y),y) \coloneqq \phi(\phi^{-1}(x^\star(y),y),y)$. $\Omega^\phi(x,y)$ is achieved.

Now we turn to compute the Jacobian of $\Omega^\phi(x,y)$ w.r.t. x. We introduce a new notation $\mathbf{D}_1 F(x,y)$ meaning the derivative of F w.r.t. the first variable.

$$
\begin{aligned}
\Omega_1^\phi(x,y) &= g_{21}(x,y) + \Psi_1^\phi(x,y) g_1(x,y) + \Psi^\phi(x,y) g_{11}(x,y),\\
\text{where } \Psi_1^\phi(x,y) &= \mathbf{D}_1\phi_2^\top(z,y) - [U_1^\phi(x,y)]^\top [V^\phi(x,y)]^{-1} + [U^\phi(x,y)]^\top [V^\phi(x,y)]^{-1} V_1^\phi(x,y) [V^\phi(x,y)]^{-1},\\
U_1^\phi(x,y) &= F_{12}(x,y) + F_{11}(x,y)\phi_2(z,y) + F_1(x,y)\mathbf{D}_1\phi_2(z,y) + [\phi_1(z,y)]^{-1}\big([\phi_{12}(z,y)]^\top F_1(x,y)\big)^\top\\
&\quad + \mathbf{D}_1\phi_1^{-1}(x,y)\phi_{21}(z,y)F(x,y) + \phi^{-1}(x,y)\mathbf{D}_1\phi_{21}(z,y)F(x,y),\\
V_1^\phi(x,y) &= F_{11}(x,y) + [\phi_1(z,y)]^{-\top}\phi_{11}(z,y)F_1(x,y)[\phi_1(z,y)]^{-1}\\
&\quad + \mathbf{D}_1\phi_1^{-\top}(x,y)\phi_{11}(z,y)F(x,y)[\phi_1(z,y)]^{-1} + [\phi_1(z,y)]^{-\top}\mathbf{D}_1\phi_{11}(z,y)F(x,y)[\phi_1(z,y)]^{-1}\\
&\quad + [\phi_1(z,y)]^{-\top}\phi_{11}(z,y)F(x,y)\mathbf{D}_1\phi_1^{-1}(z,y).
\end{aligned}
\tag{25}
$$

Note that this is true for any (x,y). However, the terms $\mathbf{D}_1\phi_2$, $\mathbf{D}_1\phi_1^{-1}$, $\mathbf{D}_1\phi_{21}$, $\mathbf{D}_1\phi_{11}$ are complicated. Because we have a inversion $\phi(x,y)=\phi(\phi^{-1}(x,y),y)$ depending on x. Fortunately, when we evaluate it at the root $(x^\star,y)$ where $x^\star \coloneqq x^\star(y)=\phi(z^\star,y)$, $F(x^\star,y)=0$. Thus, we have simplified U_1^ϕ and V_1^ϕ:

$$
\begin{aligned}
U_1^\phi(x^\star,y) &= F_{12}(x^\star,y) + F_{11}(x^\star,y)\phi_2(z^\star,y) + F_1(x^\star,y)\mathbf{D}_1\phi_2(z^\star,y) + [\phi_1(z^\star,y)]^{-1}\big[[\phi_{12}(z^\star,y)]^\top F_1(x^\star,y)\big]^\top,\\
V_1^\phi(x^\star,y) &= F_{11}(x^\star,y) + [\phi_1(z^\star,y)]^{-\top}\phi_{11}(z^\star,y)F_1(x^\star,y)[\phi_1(z^\star,y)]^{-1}.
\end{aligned}
\tag{26}
$$

Plugging this back to $\Psi_1^\phi(x^\star,y)$, we have:

$$
\begin{aligned}
\Psi_1^\phi(x^\star,y) =& -\big(F_{12}(x^\star,y) + F_{11}(x^\star,y)\phi_2(z^\star,y) + [\phi_1(z^\star,y)]^{-1}[\phi_{12}(z^\star,y)]^\top F_1(x^\star,y)\big)^\top [F_1(x^\star,y)]^{-1}\\
&+\big(F_2(x^\star,y) + F_1(x^\star,y)\phi_2(z^\star,y)\big)^\top [F_1(x^\star,y)]^{-1} F_{11}(x^\star,y)[F_1(x^\star,y)]^{-1}\\
&+\big(F_2(x^\star,y) + F_1(x^\star,y)\phi_2(z^\star,y)\big)^\top [\phi_1(z^\star,y)]^{-\top}\phi_{11}(z^\star,y)F_1(x^\star,y)[\phi_1(z^\star,y)]^{-1}[F_1(x^\star,y)]^{-1}\\
=& \underbrace{-[F_{12}(x^\star,y)]^\top [F_1(x^\star,y)]^{-1} + \big([F_1(x^\star,y)]^{-1}F_{11}(x^\star,y)[F_1(x^\star,y)]^{-1}F_2(x^\star,y)\big)^\top}_{\Psi_1(x^\star,y)}\\
&+\underbrace{-[F_1(x^\star,y)]^{-1}[\phi_1(z^\star,y)]^{-\top}[\phi_{12}(z^\star,y)]^\top F_1(x^\star,y)}_{W^\phi(y)}\\
&+\underbrace{\big([\phi_1(z^\star,y)]^{-\top}\phi_{11}(z^\star,y)F_1(x^\star,y)[\phi_1(z^\star,y)]^{-1}[F_1(x^\star,y)]^{-1}\phi_2(z^\star,y)\big)^\top}_{S^\phi(y)}\\
&+\underbrace{\big([F_1(x^\star,y)]^{-1}[\phi_1(z^\star,y)]^{-\top}\phi_{11}(z^\star,y)F_1(x^\star,y)[\phi_1(z^\star,y)]^{-1}[F_1(x^\star,y)]^{-1}F_2(x^\star,y)\big)^\top}_{T^\phi(y)}.
\end{aligned}
\tag{27}
$$

The $F_1(x^\star,y)\mathbf{D}_1\phi_2(z^\star,y)$ is canceled out in the first equation. The $F_{11}(x^\star,y)\phi_2(z^\star,y)$ is canceled out in the second equation. We want to emphasize that the above equation is only true at $(x^\star,y)$. □

Proposition 7. *Assume $x,y\in\mathbb{R}$ and that $g(x,y)$ and $F(x,y)$ are linear w.r.t. x but arbitrary on y. Then Ω^ϕ is super-efficient if and only if for all y,*

$$
\frac{\phi_{12}(z^\star(y),y)}{\phi_1(z^\star(y),y)} - \frac{\phi_2(z^\star(y),y)\phi_{11}(z^\star(y),y)}{[\phi_1(z^\star(y),y)]^2} - \frac{F_2(x^\star(y),y)\phi_{11}(z^\star(y),y)}{F_1(x^\star(y),y)[\phi_1(z^\star(y),y)]^2} = \frac{g_{12}(x^\star(y),y)}{g_1(x^\star(y),y)} - \frac{F_{12}(x^\star(y),y)}{F_1(x^\star(y),y)}
$$

Proof. Because g and F are linear on x, $g_{11}(x,y)$ and $F_{11}(x,y)$ are 0. The inversion in 1D case equals division

and the transpose can be ignored. Using these two facts and Prop. 6, we have:

$$\begin{aligned}
D(y) &= g_{12}(x^\star(y), y)\\
\Psi_1(x^\star(y), y) &= -\frac{F_{12}(x^\star(y), y)}{F_1(x^\star(y), y)}\\
W^\phi(y) &= -\frac{\phi_{12}(z^\star(y), y)}{\phi_1(z^\star(y), y)}\\
S^\phi(y) &= \frac{\phi_{11}(z^\star(y), y)\phi_2(z^\star(y), y)}{[\phi_1(z^\star(y), y)]^2}\\
T^\phi(y) &= \frac{\phi_{11}(z^\star(y), y)F_2(x^\star(y), y)}{F_1(x^\star(y), y)[\phi_1(z^\star(y), y)]^2}
\end{aligned}$$

Plugging everything into $\Psi_1^\phi(x^\star(y), y)$, we have:

$$\begin{aligned}
\Psi_1^\phi(x^\star(y), y) &= D(y) + \big(\Psi_1(x^\star(y), y) + W^\phi(y) + S^\phi(y) + T^\phi(y)\big) g_1(x^\star(y), y)\\
&= g_{12}(x^\star(y), y) - \frac{F_{12}(x^\star(y), y)}{F_1(x^\star(y), y)} g_1(x^\star(y), y) - \frac{\phi_{12}(z^\star(y), y)}{\phi_1(z^\star(y), y)} g_1(x^\star(y), y)\\
&+ \frac{\phi_{11}(z^\star(y), y)\phi_2(z^\star(y), y)}{[\phi_1(z^\star(y), y)]^2} g_1(x^\star(y), y) + \frac{\phi_{11}(z^\star(y), y)F_2(x^\star(y), y)}{F_1(x^\star(y), y)[\phi_1(z^\star(y), y]^2} g_1(x^\star(y), y)
\end{aligned}$$

Setting it to 0 and dividing both side with $g_1(x^\star(y), y)$, we get what we want.

□

Proposition 8. *If g is affine on x (see Prop.3), then super-efficient reparameterization $\phi(z, y) = \phi_0(z)$ exists and defines locally a 2-parameters family of maps. If furthermore F is linear of x, i.e. $F(x, y) = a(y)x + b$, where $a : \mathbb{R} \to \mathbb{R}$ and $b \in \mathbb{R}$, these super-efficient maps are of the form $\phi_0(z) = \alpha e^{\beta z}$ for $(\alpha, \beta) \in \mathbb{R}$.*

Proof. Because g is affine on x, we have $g_{12}(x, y) = 0$. $\phi(z, y) = \phi_0(z)$ reads that $\phi_{12}(z, y) = \phi_2(z, y) = 0$. We first show the existence of $\phi_0(z)$. Denoting that $phi_0(z^\star(y)) = x^\star(y)$ and from Prop. 7, we have:

$$\frac{F_2(x^\star(y), y)\phi_{0,11}(z^\star(y))}{F_1(x^\star(y), y)[\phi_{0,1}(z^\star(y), y)]^2} = \frac{g_{12}(x^\star(y), y)}{g_1(x^\star(y), y)} - \frac{F_{12}(x^\star(y), y)}{F_1(x^\star(y), y)}$$

This equation can be reformulated to:

$$\begin{aligned}
\phi_{0,11} &= \mathcal{F}(\phi_0, \phi_{0,1})\\
\text{where } \mathcal{F}(\phi_0, \phi_{0,1}) &= \left(\frac{g_{12}(\phi_0(z^\star(y)), y)}{g_1(\phi_0(z^\star(y)), y)} - \frac{F_{12}(\phi_0(z^\star(y)), y)}{F_1(\phi_0(z^\star(y)), y)}\right) \frac{F_1(\phi_0(z^\star(y)), y)[\phi_{0,1}(z^\star(y), y)]^2}{F_2(\phi_0(z^\star(y)), y)}
\end{aligned}$$

Denoting $\phi_{0,1} = \mathcal{G}(\phi)$ yields:

$$\begin{aligned}
\phi_{0,11} &= \mathcal{G}(\phi)\mathcal{G}'(\phi)\\
\Rightarrow &\mathcal{G}(\phi)\mathcal{G}'(\phi) = \mathcal{F}(\phi, \mathcal{G})\\
\Rightarrow &\mathcal{G}'(\phi) = \tilde{\mathcal{F}}(\phi, \mathcal{G}(\phi)).
\end{aligned}$$

with $\tilde{\mathcal{F}}(\phi, \mathcal{G}(\phi)) = \mathcal{F}(\phi, \mathcal{G})/\mathcal{G}(\phi)$. This becomes a first-order differential equation about $\mathcal{G}$. The Cauchy-Lipschtz theorem shows the existence of $\mathcal{G}$ with one parameter. Performing another integration we can get ϕ_0 with one additional parameter.

If $F(x, y) = a(y)x + b$, we have $F_1(x, y) = a(y)$, $F_2(x, y) = a'(y)x$ and $F_{12}(x, y) = a'(y)$. Plugging everything into Prop. 7, we have:

$$\begin{aligned}
\frac{a'(y)\phi(z, y)\phi_{11}(z, y)}{a(y)[\phi_1(z, y)]^2} &= \frac{a'(y)}{a(y)}\\
\Rightarrow \phi(z, y)\phi_{11}(z, y) &= [\phi_1(z, y)]^2
\end{aligned}$$

The solution of this second-order ODE should be $\phi(z, y) = \alpha e^{\beta z}$ for $(\alpha, \beta) \in \mathbb{R}^2$. □

Proposition 9. *The estimator $\Omega^{\psi}_{\text{loc}}$ is consistent and one has*

$$C_y(\Omega^{\psi}_{\text{loc}}) = C_y(\Omega^{\psi_{x^\star(y),y}}).$$

Proof. As Prop. 6 is true for any bijective ϕ, so $\Omega^{\psi}_{\text{loc}}$ is consistent.

We then show $C_y(\Omega^{\psi}_{\text{loc}}) = C_y(\Omega^{\psi_{x^\star(y),y}})$. We use the same notation $\mathbf{D}_1 F(x,y)$ as in the proof of Prop. 6. The derivations in Eq. (25) still holds when computing $\mathbf{D}_1\Omega^{\psi}_{\text{loc}}(x,y)$ and $\mathbf{D}_1\Omega^{\psi_{x^\star(y),y}}(x,y)$. Note that there are differences on $\mathbf{D}_1\phi_2$, $\mathbf{D}_1\phi_1^{-1}$, $\mathbf{D}_1\phi_{21}$, $\mathbf{D}_1\phi_{11}$ and $\mathbf{D}_1\phi_2$. In the case $\Omega^{\psi}_{\text{loc}}$, the derivative considers the dependency on x from the parameter of ψ, the change of variable (x,y) and the inversion of ψ^{-1}, while $\Omega^{\psi_{x^\star(y),y}}$ only considers the latter two. However, they agree when evaluating at $(x^\star(y),y)$ thanks to $F(x^\star(y),y)=0$.

□

Proposition 10. *Let $\phi(z,y) = \psi_{x,\bar{y}}(z,y) = R(x,y)Q(z,\bar{y}) + x$, denoting $x^\star \coloneqq x^\star(y)$ and $\psi_{x,\bar{y}}(z^\star,y) = x^\star$. one has:*

$$\begin{aligned}
W^{\phi}(y) =& -[F_1(x^\star,y)]^{-1}[R(x,y)]^{-\top}[Q_1(z^\star,\bar{y})]^{-\top}Q_1(z^\star,\bar{y})[R_2(x^\star,y)]^{\top}F_1(x^\star,y),\\
S^{\phi}(y) =& \big([R(x,y)]^{-\top}[Q_1(z^\star,\bar{y})]^{-\top}Q_{11}(z^\star,\bar{y})R(x,y)F_1(x^\star,y)\\
&[Q_1(z^\star,\bar{y})]^{-1}[R(x,y)]^{-1}[F_1(x^\star,y)]^{-1}Q(z^\star,\bar{y})R_2(x,y)\big)^{\top},\\
T^{\phi}(y) =& \big([F_1(x^\star,y)]^{-1}[R(x,y)]^{-\top}[Q_1(z^\star,\bar{y})]^{-\top}[Q_{11}(z^\star,\bar{y})]\\
&R(x,y)F_1(x^\star,y)[Q_1(z^\star,\bar{y})]^{-1}[R(x,y)]^{-1}[F_1(x^\star,y)]^{-1}F_2(x^\star,y)\big)^{\top},
\end{aligned}$$

for any $(x,\bar{y})$.

Proof. Because $\psi_{x,\bar{y}}$ is a special case of ϕ, we can apply the results from Prop. 6 and 9. Computing $W^{\phi}(y)$, $S^{\phi}(y)$ and $T^{\phi}(y)$ require ϕ_1, ϕ_{12}, ϕ_2 and ϕ_{11}. Viewing $x,\bar{y}$ as constant, we have:

$$\begin{aligned}
\phi_1(z,y) &= \psi_{1,x,\bar{y}}(z,y) = R(x,y)Q_1(z,\bar{y})\\
\phi_{12}(z,y) &= \psi_{12,x,\bar{y}}(z,y) = Q_1(z,\bar{y})R_2(x,y)\\
\phi_2(z,y) &= \psi_{2,x,\bar{y}}(z,y) = Q(z,\bar{y})R_2(x,y)\\
\phi_{11}(z,y) &= \psi_{11,x,\bar{y}}(z,y) = Q_{11}(z,\bar{y})R(x,y).
\end{aligned}$$

Plugging them into Prop. 6 to compute $W^{\phi}(y)$, $S^{\phi}(y)$ and $T^{\phi}(y)$, we get the desired results. □

Proposition 11 (Newton-like reparameterization)**.** *We assume g is of the form $g(x,y) = ax + m(y)$ (see Prop. 3 for a discussion). Let $\psi_{x,\bar{y}}(z,y) = R(x,y)Q(z,\bar{y}) + x$ and let $F(x,y)$ be bijective on x for all y. For $R(x,y) = [F_1(x,y)]^{-1}$, $Q(z,\bar{y}) = -F(z,\bar{y})$, $\Omega^{\psi}_{\text{loc}}$ is super efficiency.*

Proof. We need to examine $C_y(\Omega^{\psi}_{\text{loc}})$, which is the same as $\Omega_1^{\psi_{x^\star(y),y}}(x^\star(y),y)$ from Prop. 9. Therefore, we compute Q_1, Q_{11}, R_2 with $Q(z,y) = -F(z,y)$ and $R(x^\star(y),y) = [F_1(x^\star(y),y)]^{-1}$, we have:

$$\begin{aligned}
Q_1(z,y) &= -F_1(z,y)\\
Q_{11}(z,y) &= -F_{11}(z,y)\\
R_2(x,y) &= -\big([F_1(x^\star(y),y)]^{-1}[F_{12}(x^\star(y),y)]^{\top}[F_1(x^\star(y),y)]^{-1}\big)^{\top}
\end{aligned}$$

Because F is bijective in the assumption, given $x^\star(y)$ we can compute $z^\star(y)$:

$$z^\star(y) = \psi^{-1}_{x^\star(y),y}(x^\star(y),y) = Q^{-1}(-[R(x^\star(y),y)]^{-1}(x^\star(y)-x^\star(y)),y) = F^{-1}(0,y) = x^\star(y).$$

We thus have $Q(z^\star(y),y)=0$. Using that$F_1(x^\star(y),y)$ is symmetric, and denoting $x^\star = x^\star(y)$ and $z^\star = z^\star(y)$,

we have from Prop 10:

$$\begin{aligned}
W^{\phi}(y) =& -[F_1(x^\star,y)]^{-1}[F_1(x^\star,y)]^\top[-F_1(z^\star,y)]^{-1}\\
&[-F_1(z^\star,y)][-F_1(x^\star,y)]^{-1}[F_{12}(x^\star,y)]^\top[F_1(x^\star,y)]^{-1}F_1(x^\star,y)\\
=&[F_{12}(x^\star,y)]^\top[F_1(x,y)]^{-1}\\
S^{\phi}(y) =&0\\
T^{\phi}(y) =&\big([F_1(x^\star,y)]^{-1}[F_1(x^\star,y)][-F_1(z^\star,y)]^{-1}[-F_{11}(z^\star,y)]\\
&[F_1(x^\star,y)]^{-1}F_1(x^\star,y)[-F_1(x^\star,y)]^{-1}F_1(x^\star,y)[F_1(x^\star,y)]^{-1}F_2(x^\star,y)\big)^\top\\
=&-\big([F_1(z^\star,y)]^{-1}F_{11}(z^\star,y)[F_1(z^\star,y)]^{-1}F_2(x^\star,y)\big)^\top
\end{aligned}$$

Because $z^\star = x^\star$, we have $\Psi_1^{\phi}(x^\star,y) = \Psi_1(x^\star,y) + W^{\phi}(y) + T^{\phi}(y) = 0$ with $\phi = \psi_{x^\star(y),y}$. $g(x,y) = ax + m(y)$ reads that $D(y) = 0$. In total, we have $\Omega_1^{\psi_{x^\star(y),y}}(x^\star(y),y) = 0$. □

Proposition 12. *Let $x^\star \coloneqq x^\star(y)$, $\psi_{x^\star,y} \coloneqq R(x^\star,y)Q(z,y) + x^\star$, and $z^\star$ satisfy $\psi_{x^\star,y}(z^\star,y) = x^\star$. Denoting $E^Q(y) \coloneqq Q(z^\star,y) + F(x^\star,y)$, $E^{Q_1}(y) \coloneqq Q_1(z^\star,y) + F_1(x^\star,y)$, $E^{Q_{11}}(y) \coloneqq Q_{11}(z^\star,y) + F_{11}(x^\star,y)$, $E^R(y) \coloneqq R(x^\star,y) - [F_1(x^\star,y)]^{-1}$, $E^{R_2}(y) \coloneqq R_2(x^\star,y) + [F_1(x^\star,y)]^{-1}F_{21}(x^\star,y)[F_1(x^\star,y)]^{-1}$. We have that $C_y(\Psi^{\psi}_{\text{loc}}) = \mathcal{O}(\|E^Q\|_{\text{op}}, \|E^{Q_1}\|_{\text{op}}, \|E^{Q_{11}}\|_{\text{op}}, \|E^R\|_{\text{op}}, \|E^{R_2}\|_{\text{op}})$.*

Proof. We need to compute $C_y(\Psi^{\psi}_{\text{loc}})$, which is the same as $C_y(\Psi^{\psi_{x^\star,y}})$. Because $\Psi_1^{\psi_{x^\star(y),y}}(x^\star,y) = \Psi_1(x^\star,y) + W^{\psi_{x^\star(y),y}}(y) + S^{\psi_{x^\star(y),y}}(y) + T^{\psi_{x^\star(y),y}}(y)$ from Prop. 6 and $W^{\psi_{x^\star(y),y}}(y) = [F_{12}(x^\star,y)]^\top[F_1(x,y)]^{-1}$ when $E^R(y) = E^{Q_1}(y) = E^{R_2}(y) = 0$ from the proof of Prop. 11. We first analyze the error on $W^{\psi_{x^\star(y),y}}$:

$$\begin{aligned}
W^{\psi_{x^\star(y),y)}}(y) =& -[F_1(x^\star,y)]^{-1}[R(x,y)]^{-\top}[Q_1(z^\star,y)]^{-\top}Q_1(z^\star,y)[R_2(x^\star,y)]^\top F_1(x^\star,y)\\
=&[F_1(x^\star,y)]^{-1}(E^R(y) + [F_1(x^\star,y)]^{-1})^{-\top}(E^{Q_1}(y) - F_1(x^\star,y))^{-\top}(E^{Q_1}(y) - F_1(x^\star,y))\\
&(E^{R_2}(y) - [F_1(x^\star,y)]^{-1}F_{21}(x^\star,y)[F_1(x^\star,y)]^{-1})^\top F_1(x^\star,y)\\
=&[F_{12}(x^\star,y)]^\top[F_1(x^\star,y)]^{-1}\\
&+ (E^{Q_1}(y) - F_1(x^\star,y))^{-\top}(E^{Q_1}(y) - F_1(x^\star,y))[E^{R_2}(y)]^\top F_1(x^\star,y)\\
&+ \big([E^{Q_1}(y)]^{-\top}[E^{Q_1}(y)] + [-F_1(x^\star,y)]^{-\top}[E^{Q_1}(y)]\\
&+ [E^{Q_1}(y)]^{-\top}[-F_1(x^\star,y)]\big)\big(-F_{21}(x^\star,y)[F_1(x^\star,y)]^{-1}\big)^\top\\
&+ [F_1(x^\star,y)]^{-1}[E^R(y)]^{-\top}(E^{Q_1}(y) - F_1(x^\star,y))^{-\top}(E^{Q_1}(y) - F_1(x^\star,y))(-F_{21}(x^\star,y)[F_1(x^\star,y)]^{-1})^\top\\
&+ [F_1(x^\star,y)]^{-1}[E^R(y)]^{-\top}(E^{Q_1}(y) - F_1(x^\star,y))^{-\top}(E^{Q_1}(y) - F_1(x^\star,y))[-E^{R_2}(y)]^\top F_1(x^\star,y).
\end{aligned}$$

Though it is complicated, the last four lines in the last equation are polynomial on E^{Q_1}, E^{R_2} and E^R. Therefore, we have $\|W^{\psi_{x^\star(y),y}}(y)\|_{\text{op}} = \|[F_{12}(x^\star,y)]^\top[F_1(x^\star,y)]^{-1}\|_{\text{op}} + \mathcal{O}(\|E^{Q_1}\|_{\text{op}}, \|E^{R_2}\|_{\text{op}}, \|E^R\|_{\text{op}})$. The same logic can be applied to $S^{\psi_{x^\star(y),y}}$ and $T^{\psi_{x^\star(y),y}}$. Putting them all together, we finish the proof. □

Proposition 13 (Comparison of the two methods)**.** *Let ϕ be smooth and bijective, one has*

$$[C_y(\Omega^{\phi})]^2 - [C_y(\Omega^P)]^2 \geq \langle U_+(y)v_P(y), U_-(y)v_P(y)\rangle \tag{28}$$

$$[C_y(\Omega^P)]^2 - [C_y(\Omega^{\phi})]^2 \geq \langle V_+(y)v_{\phi}(y), V_-(y)v_{\phi}(y)\rangle \tag{29}$$

where, for E^P as in Prop. 4, Ψ_1^{ϕ} and D as in Prop. 6,

$$\begin{aligned}
U_{\pm}(y) &\coloneqq D(y) \pm D(y)E^P(x^\star(y),y) + \Psi_1^{\phi}(x^\star(y),y)g_1(x^\star(y),y) \pm \Psi_1(x^\star(y),y)g_1(x^\star(y),y)E^P(x^\star(y),y),\\
V_{\pm}(y) &\coloneqq D(y)E^P(x^\star(y),y) \pm D(y) + \Psi_1(x^\star(y),y)g_1(x^\star(y),y)E^P(x^\star(y),y) \pm \Psi_1^{\phi}(x^\star(y),y)g_1(x^\star(y),y),\\
v_{\omega}(y) &\coloneqq \underset{\|u\|=1}{\arg\max} \|\Omega_1^{\omega}(x^\star(y),y)u\| \text{ for } \omega \in \{P,\phi\},
\end{aligned}$$

Proof. We derive $[C_y(\Omega^\phi)]^2 - [C_y(\Omega^P)]^2$ and the similar idea can be applied to derive $[C_y(\Omega^P)]^2 - [C_y(\Omega^\phi)]^2$.

$$\begin{aligned}
[C_y(\Omega^\phi)]^2 - [C_y(\Omega^P)]^2 =& \|\Omega_1^\phi(x^\star(y),y)\|_{\text{op}}^2 - \|\Omega^P(x^\star(y),y)\|_{\text{op}}^2 \\
=& \|D(y) + \Psi_1^\phi(x^\star(y),y)g_1(x^\star(y),y)\|_{\text{op}}^2 - \|\Omega_1(x^\star(y),y)E^P(x^\star(y),y)\|_{\text{op}}^2 \\
=& \|D(y) + \Psi_1^\phi(x^\star(y),y)g_1(x^\star(y),y)\|_{\text{op}}^2 - \|(D(y) + \Psi_1(x^\star(y),y)g_1(x^\star(y),y))E^P(x^\star(y),y)\|_{\text{op}}^2 \\
\geq& \langle \big(D(y) + \Psi_1^\phi(x^\star(y),y)g_1(x^\star(y),y) - (D(y) + \Psi_1(x^\star(y),y)g_1(x^\star(y),y))E^P(x^\star(y),y)\big)v(y), \\
& \big(D(y) + \Psi_1^\phi(x^\star(y),y)g_1(x^\star(y),y) + (D(y) + \Psi_1(x^\star(y),y)g_1(x^\star(y),y))E^P(x^\star(y),y)\big)v(y)\rangle
\end{aligned}$$

by choosing $v(y) = \arg\max_{\|u\|=1} \|\Omega^P(x^\star(y),y)u\|$. □

Proposition 14. *For $\delta \coloneqq \|P(x^\star(y),y) - F_1(x^\star(y),y)\|_\infty$, we have*

$$[C_y(\Omega^\phi)]^2 - [C_y(\Omega^P)]^2 \geq \|(D(y) + \Psi_1^\phi(x^\star(y),y)g_1(x^\star(y),y))v_P\|^2 + o(\delta),$$

with v_P as defined in Prop. 13

Proof. Recall that $E^P(x^\star(y),y) = I_{d_x} - [P(x^\star(y),y)]^{-1}F_1(x^\star(y),y) = [P(x^\star(y),y)]^{-1}(P(x^\star(y),y) - F_1(x^\star(y),y))$. Continuing deriving from Prop. 13, we have:

$$\begin{aligned}
[C_y(\Omega^\phi)]^2 - [C_y(\Omega^P)]^2 \geq& \|(D(y) + \Psi_1^\phi(x^\star(y),y)g_1(x^\star(y),y))v_P\|^2 \\
& - \|(D(y) + \Psi_1(x^\star(y),y)g_1(x^\star(y),y))E^P(x^\star(y),y)v_P\|^2 \\
\geq& \|(D(y) + \Psi_1^\phi(x^\star(y),y)g_1(x^\star(y),y))v_P\|^2 \\
& - \|D(y) + \Psi_1(x^\star(y),y)g_1(x^\star(y),y)\|_{\text{op}}^2 \|E^P(x^\star(y),y)\|_{\text{op}}^2 \|v_P\|^2 \\
\geq& \|(D(y) + \Psi_1^\phi(x^\star(y),y)g_1(x^\star(y),y))v_P\|^2 \\
& - \|D(y) + \Psi_1(x^\star(y),y)g_1(x^\star(y),y)\|_{\text{op}}^2 \|P(x^\star(y),y)\|_{\text{op}}^2 \|P(x^\star(y),y) - F_1(x^\star(y),y)\|_\infty^2 \|v_P\|^2 \\
=& \|(D(y) + \Psi_1^\phi(x^\star(y),y)g_1(x^\star(y),y))v_P\|^2 + o(\delta).
\end{aligned}$$

□

Proposition 15. *For $\sigma \coloneqq \|g_1(x^\star(y),y)\|_\infty C_y(\Psi_{\text{loc}}^\psi)$*

$$[C_y(\Omega^P)]^2 - [C_y(\Omega_{\text{loc}}^\psi)]^2 \geq \|(D(y) + \Psi_1(x^\star(y),y)g_1(x^\star(y),y))E^P(x^\star(y),y)v_\phi\|^2 - \|D(y)v_\phi\|^2 + o(\sigma),$$

with v_ϕ as defined in Prop. 13 with $\phi = \Psi_{\text{loc}}^\psi$.

Proof. Continue from Prop. 13, we have:

$$\begin{aligned}
[C_y(\Omega^P)]^2 - [C_y(\Omega_{\text{loc}}^\psi)]^2 \geq& \|(D(y) + \Psi_1(x^\star(y),y)g_1(x^\star(y),y))E^P(x^\star(y),y)v_\phi\|^2 \\
& - \|(D(y) + \Psi_{1,\text{loc}}^\psi(x^\star(y),y)g_1(x^\star(y),y))v_\phi\|^2 \\
\geq& \|(D(y) + \Psi_1(x^\star(y),y)g_1(x^\star(y),y))E^P(x^\star(y),y)v_\phi\|^2 - \|D(y)v_\phi\|^2 \\
& - \|\Psi_{1,\text{loc}}^\psi(x^\star(y),y)g_1(x^\star(y),y))v_\phi\|^2 \\
\geq& \|(D(y) + \Psi_1(x^\star(y),y)g_1(x^\star(y),y))E^P(x^\star(y),y)v_\phi\|^2 - \|D(y)v_\phi\|^2 \\
& - \|g_1(x^\star(y),y)\|_\infty^2 [C_y(\Psi_{\text{loc}}^\psi)]^2 \|v_\phi\|^2 \\
=& \|(D(y) + \Psi_1(x^\star(y),y)g_1(x^\star(y),y))E^P(x^\star(y),y)v_\phi\|^2 - \|D(y)v_\phi\|^2 + o(\sigma).
\end{aligned}$$

□