Efficient Continuous Pareto Exploration in Multi-Task Learning

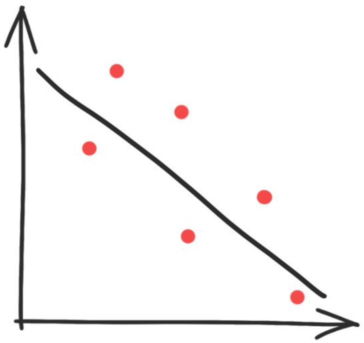
Pingchuan Ma*, Tao Du*, Wojciech Matusik
MIT CSAIL

CSAIL

ICML | 2020

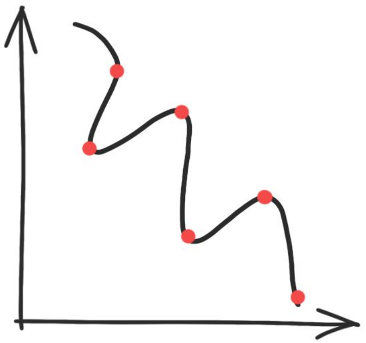
Motivation

Multiple objectives are common in machine learning.



More bias
Less variance

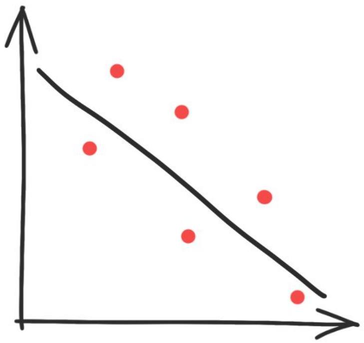
V.S.



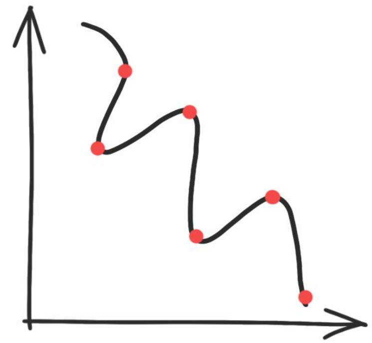
More variance
Less bias

Motivation

Multiple objectives are common in machine learning.



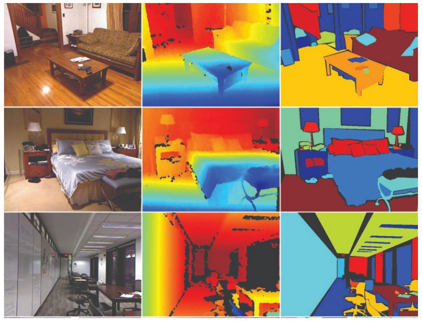
More bias
Less variance



More variance
Less bias

Label(s): Cat, Dog, Duck

Multi-Label

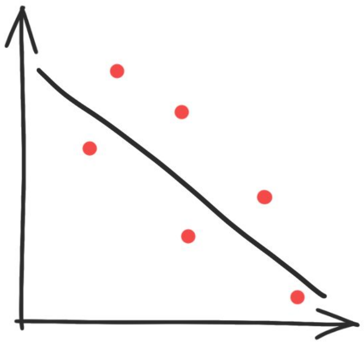


Modal(s): Image, Depth, Segmentation

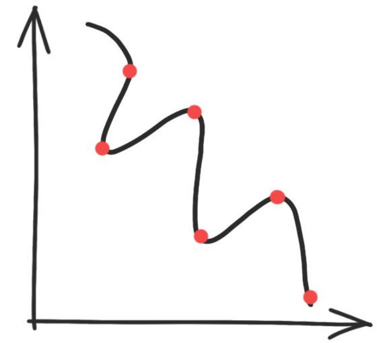
Multimodal

Motivation

...but objectives often conflict with each other!



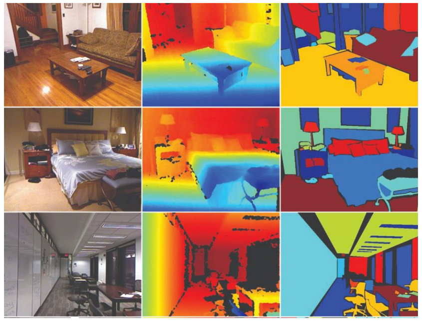
More bias
Less variance



More variance
Less bias

Label(s): Cat, Dog, Duck

Multi-Label

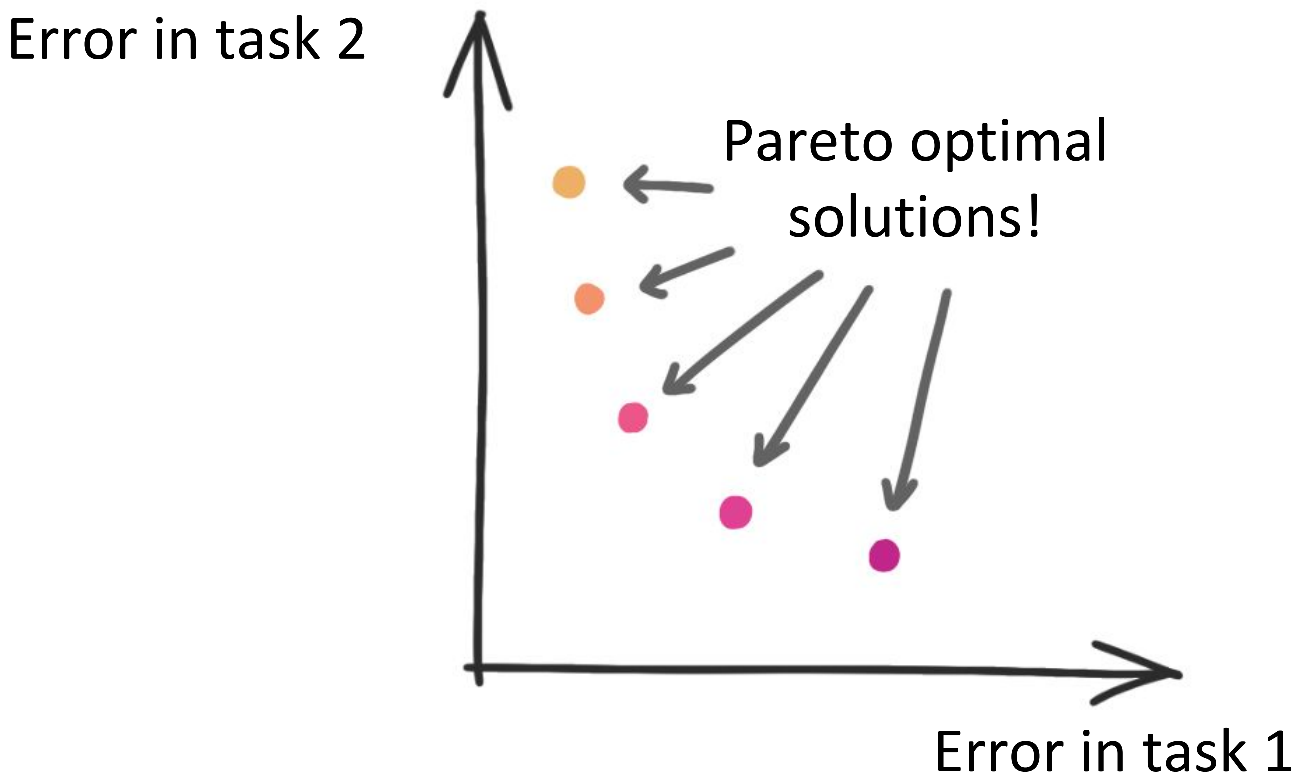


Modal(s): Image, Depth, Segmentation

Multimodal

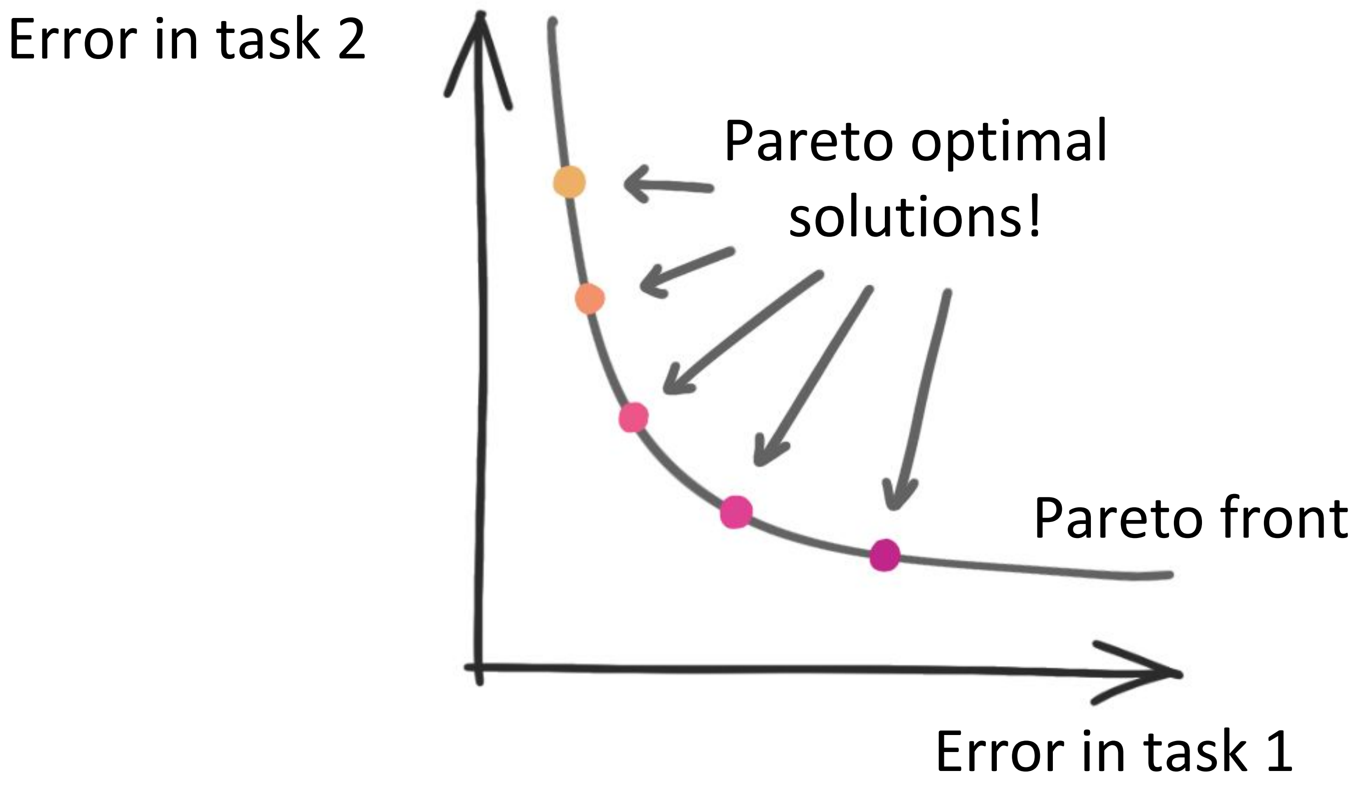
Motivation

This is characterized by the concept of Pareto optimality.



Motivation

This is characterized by the concept of Pareto optimality.



Related work

Multi-objective optimization

ıere exist vectors $\lambda \in \mathbb{R}^m$ and $\alpha \in \mathbb{R}^k$, wit
= 1, such that

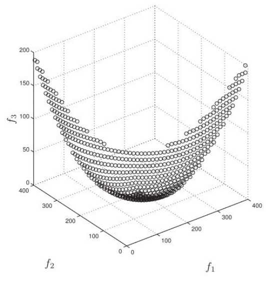
$$\sum_{i=1}^{k} \alpha_i \nabla f_i(x^*) + \sum_{j=1}^{m} \lambda_j \nabla h_j(x^*) = 0,$$

$$h_i(x^*) = 0, \qquad i = 1, \ldots, m.$$

define the scalar-valued function

$$g_\alpha : \mathbb{R}^n \to \mathbb{R}, \qquad x \mapsto g_\alpha(x) := \sum_{i=1}^{k} \alpha_i f_i(x),$$

Hillermeier
2001



Martin & Schutze
2018

	Solution type	Problem size
Hillermeier 01 Martin & Schutze 18	Continuous	Small

Related work

Multi-objective optimization

here exist vectors $\lambda \in \mathbb{R}^m$ and $\alpha \in \mathbb{R}^k$, wit
= 1, such that

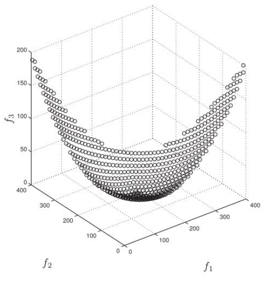
$$\sum_{i=1}^{k} \alpha_i \nabla f_i(x^*) + \sum_{j=1}^{m} \lambda_j \nabla h_j(x^*) = 0,$$

$$h_i(x^*) = 0, \qquad i = 1, \ldots, m.$$

define the scalar-valued function

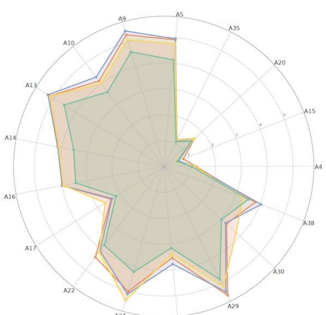
$$g_\alpha : \mathbb{R}^n \to \mathbb{R}, \qquad x \mapsto g_\alpha(x) := \sum_{i=1}^{k} \alpha_i f_i(x),$$

Hillermeier
2001

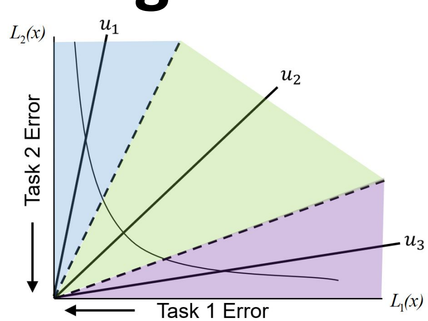


Martin & Schutze
2018

Multi-task learning



Sener & Koltun
2018



Lin et al.
2019

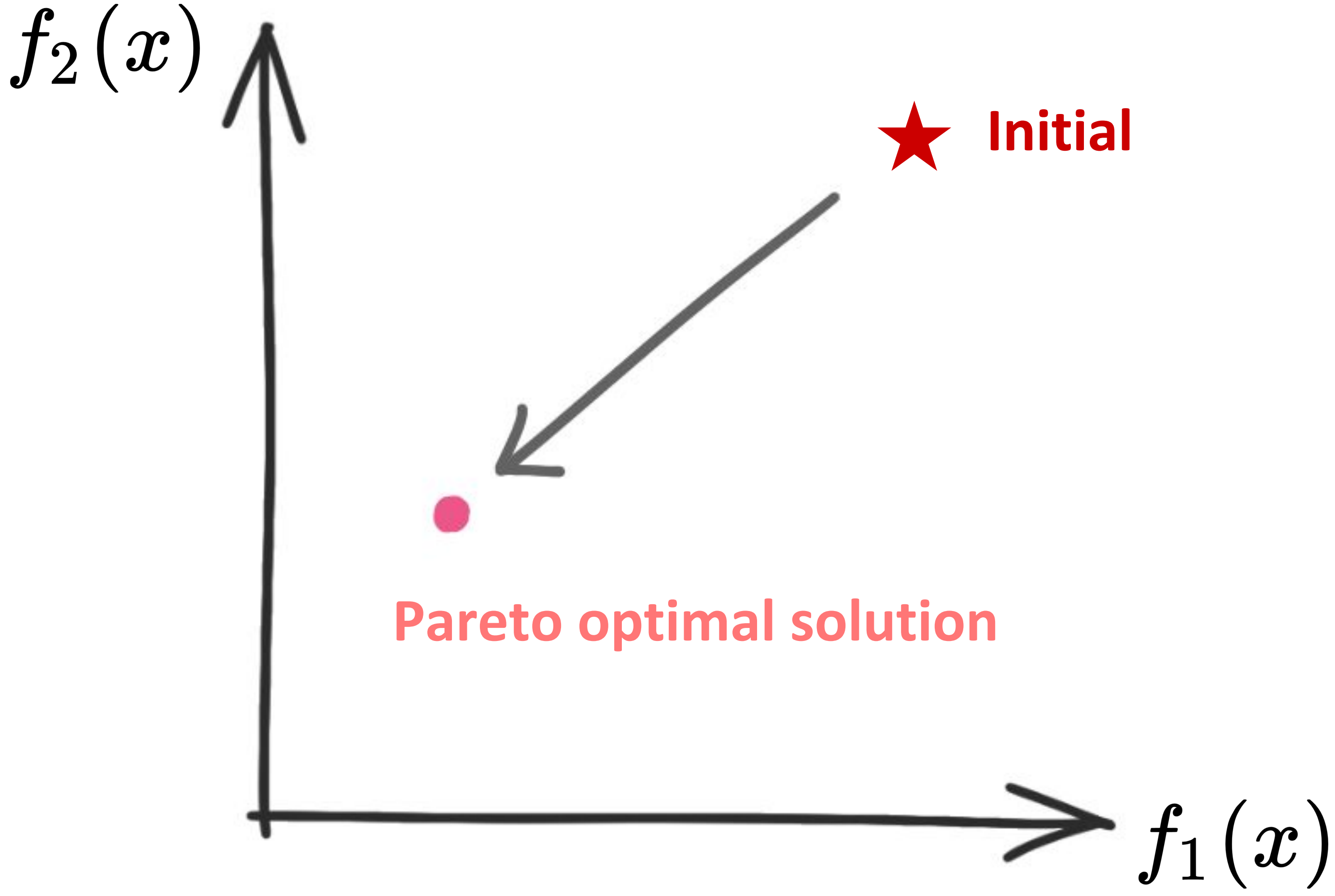
	Solution type	Problem size
Hillermeier 01 Martin & Schutze 18	Continuous	Small
Chen et al. 18 Kendall et al. 18 Sener & Koltun 18	Single discrete	Large
Lin et al. 19	Multiple discrete	Large

Contributions

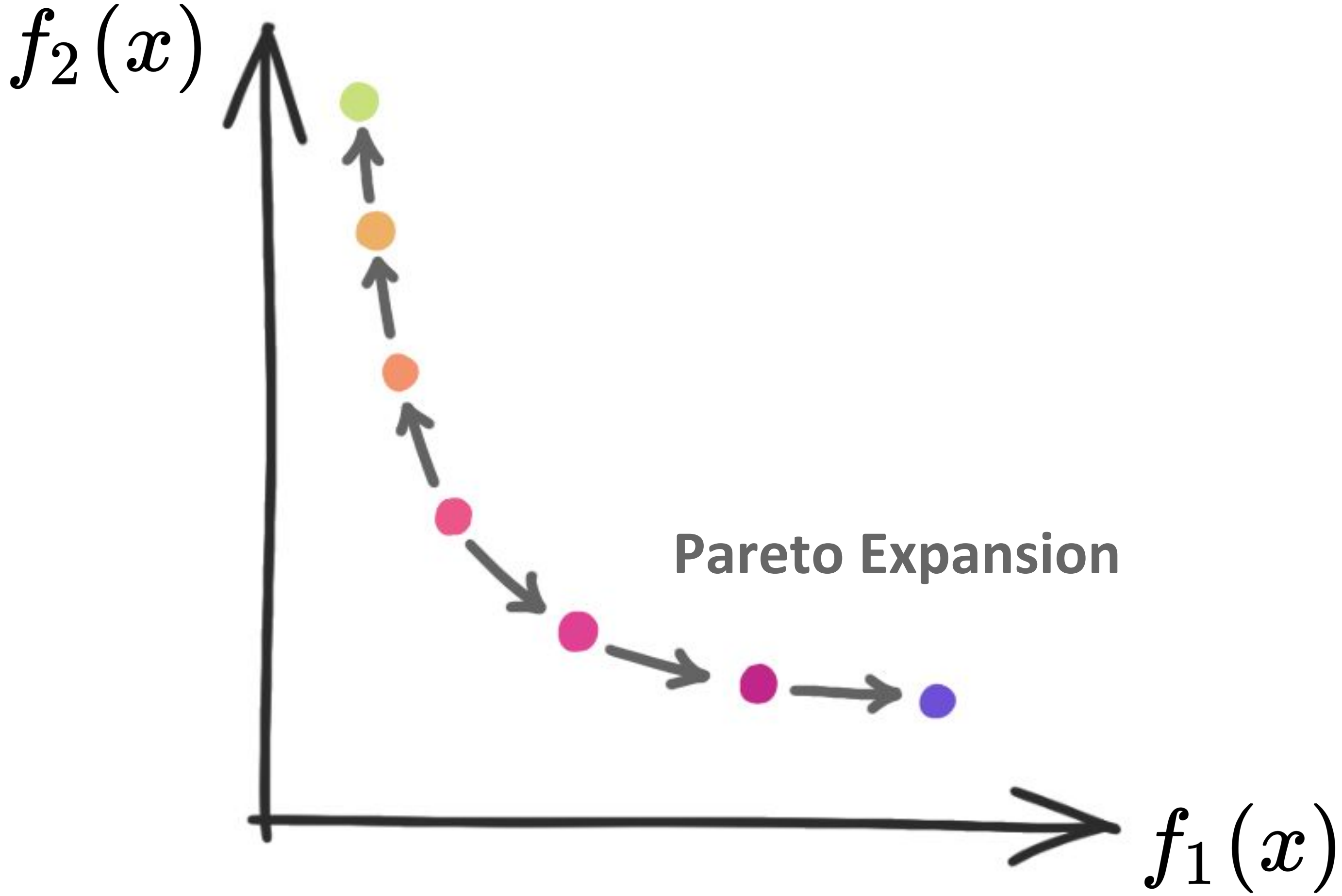
- We presented the first method to discover **continuous approximations** to Pareto fronts for **large deep-learning problems**.

	Solution type	Problem size
Hillermeier 01 Martin & Schutze 18	Continuous	Small
Chen et al. 18 Kendall et al. 18 Sener & Koltun 18	Single discrete	Large
Lin et al. 19	Multiple discrete	Large
Ours	Continuous	Large

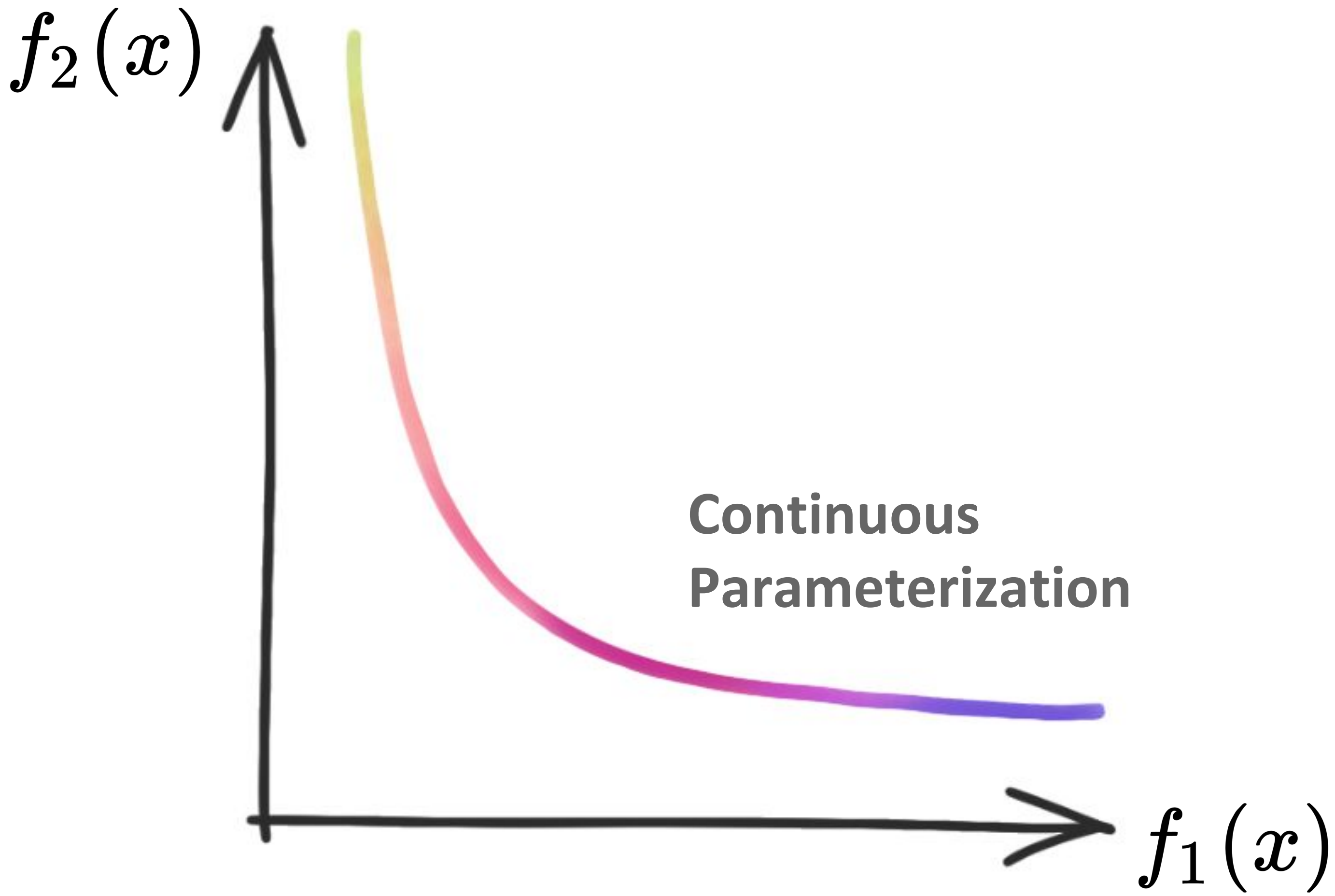
Method overview



Method overview



Method overview



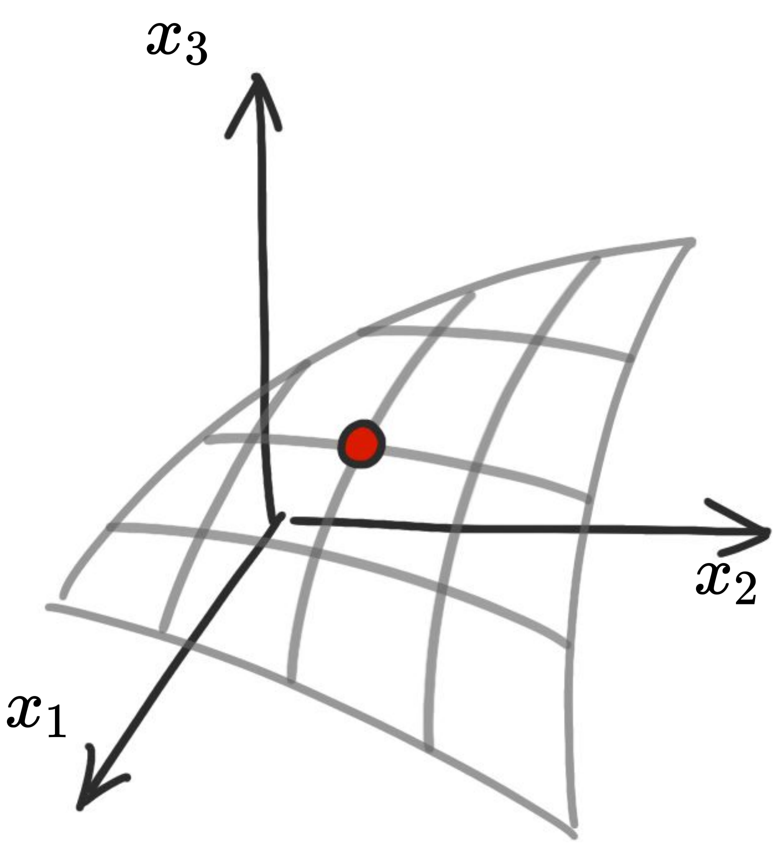
Challenges

- **The most efficient direction for expansion is unknown.**
  - How to recover the Pareto front from one solution?
- **Deep-learning parameter space has large dimensions.**
  - How to scale the method up for large-size problems?
- **Pareto solutions are discrete.**
  - How to build a continuous Pareto front from them?

Challenges

- **The most efficient direction for expansion is unknown.**
  - How to recover the Pareto front from one solution?
- **Deep-learning parameter space has large dimensions.**
  - How to scale the method up for large-size problems?
- **Pareto solutions are discrete.**
  - How to build a continuous Pareto front from them?

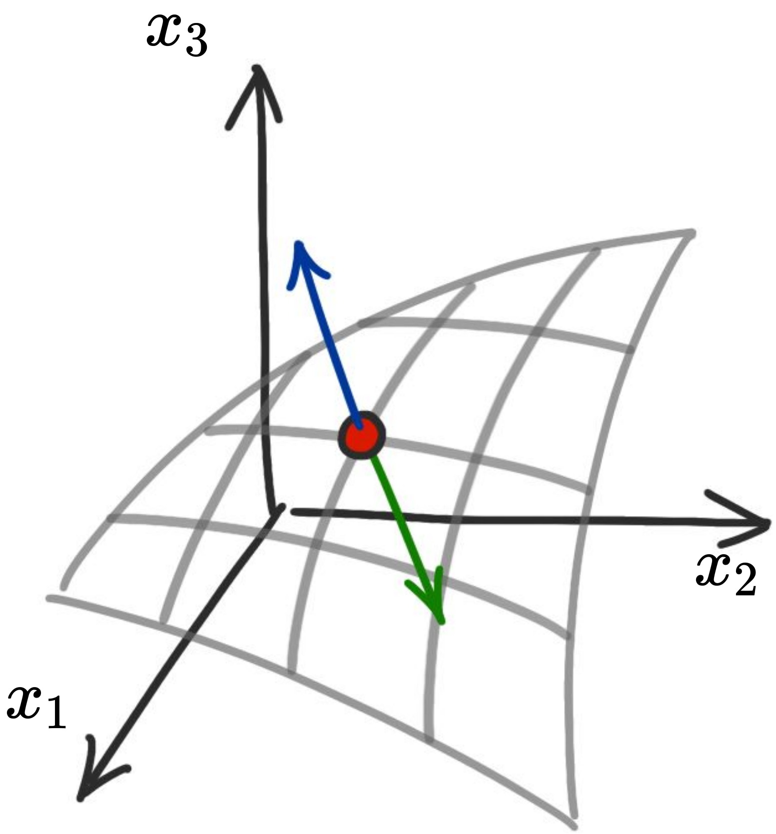
Efficient Pareto-front expansion



- **Necessary conditions**

$$\sum \alpha_i \nabla f_i(\mathbf{x}^*) = 0$$
$$\text{s.t. } \sum \alpha_i = 1,\ \boldsymbol{\alpha} \geq \mathbf{0}$$

Efficient Pareto-front expansion



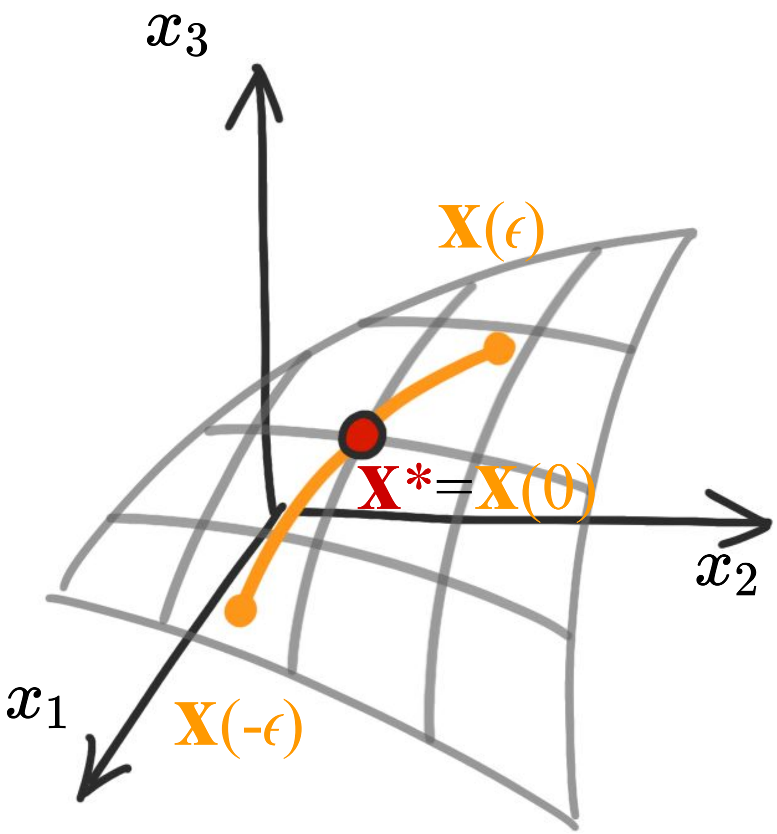
- Necessary conditions

$$\sum \alpha_i \nabla f_i(\mathbf{x}^*) = 0$$
$$\text{s.t. } \sum \alpha_i = 1, \ \boldsymbol{\alpha} \geq \mathbf{0}$$

- Specifically:

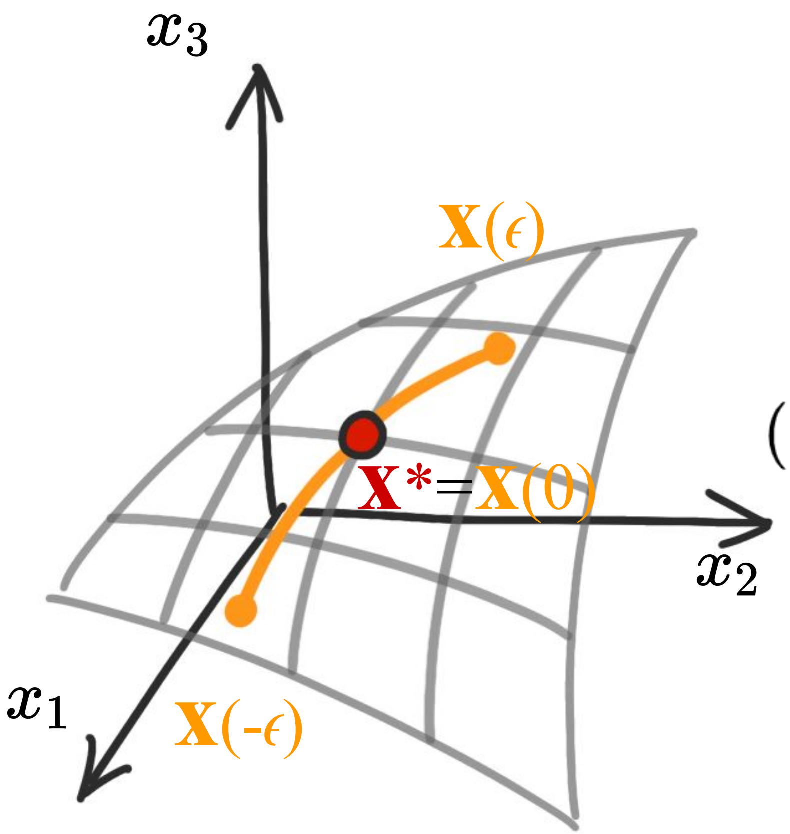
$$\alpha_1 \nabla f_1(\mathbf{x}^*) + \alpha_2 \nabla f_2(\mathbf{x}^*) = 0$$

Efficient Pareto-front expansion



$$\forall -\epsilon \leq t \leq \epsilon$$
$$\sum \alpha_i(t) \nabla f_i(\mathbf{x}(t)) = \mathbf{0}$$

Efficient Pareto-front expansion



$$\forall -\epsilon \leq t \leq \epsilon$$

$$\frac{d}{dt} \sum \alpha_i(t) \nabla f_i(\mathbf{x}(t)) = \mathbf{0}$$

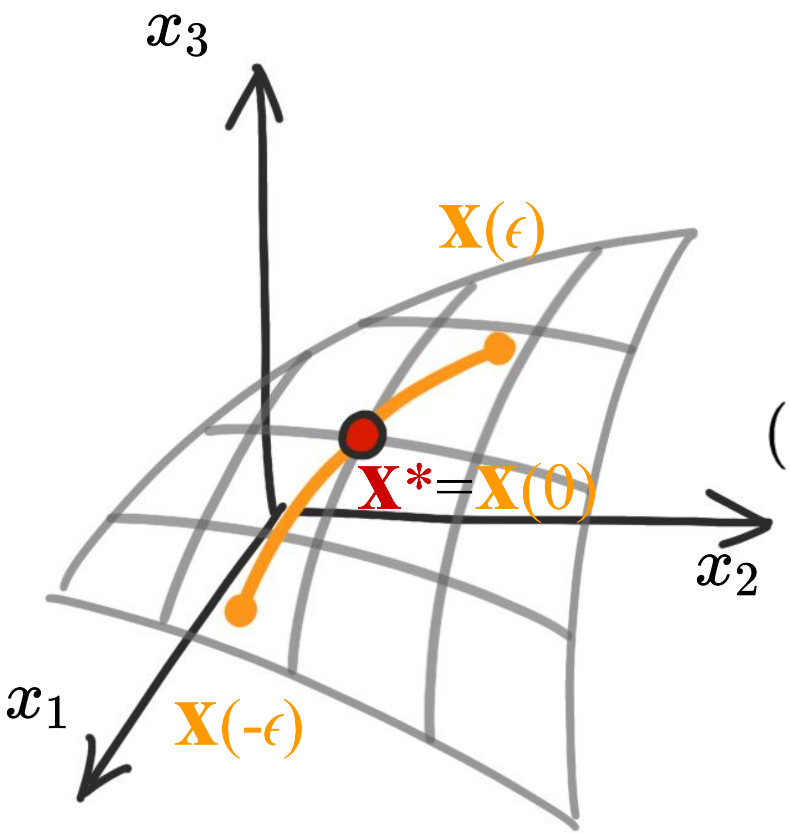
Differentiate ↓ Evaluate at $t = 0$

$$\left(\sum \alpha_i \nabla^2 f_i\right) x'(0) = -\sum \alpha_i' \nabla f_i\left(x(0)\right)$$

||

Our interest

Efficient Pareto-front expansion



$$\forall -\epsilon \leq t \leq \epsilon$$

$$\frac{d}{dt} \sum \alpha_i(t) \nabla f_i(\mathbf{x}(t)) = \mathbf{0}$$

Differentiate ↓ Evaluate at $t = 0$

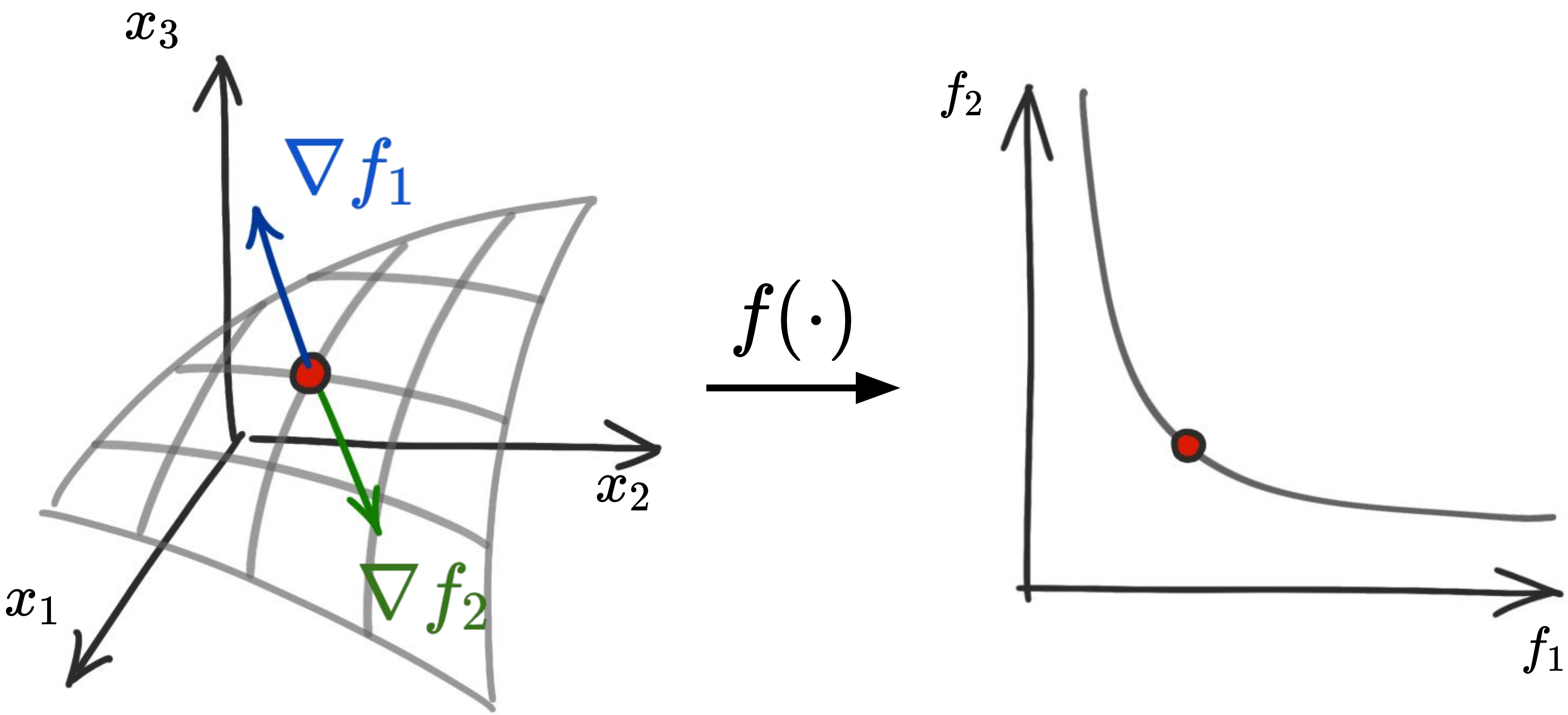
$$\left(\sum \alpha_i \nabla^2 f_i\right) x'(0) = -\sum \alpha_i' \nabla f_i \left(x(0)\right)$$

↓

$$\mathbf{H}v = \nabla \mathbf{f} \beta \in \text{colspan}\{\nabla f_i\}$$

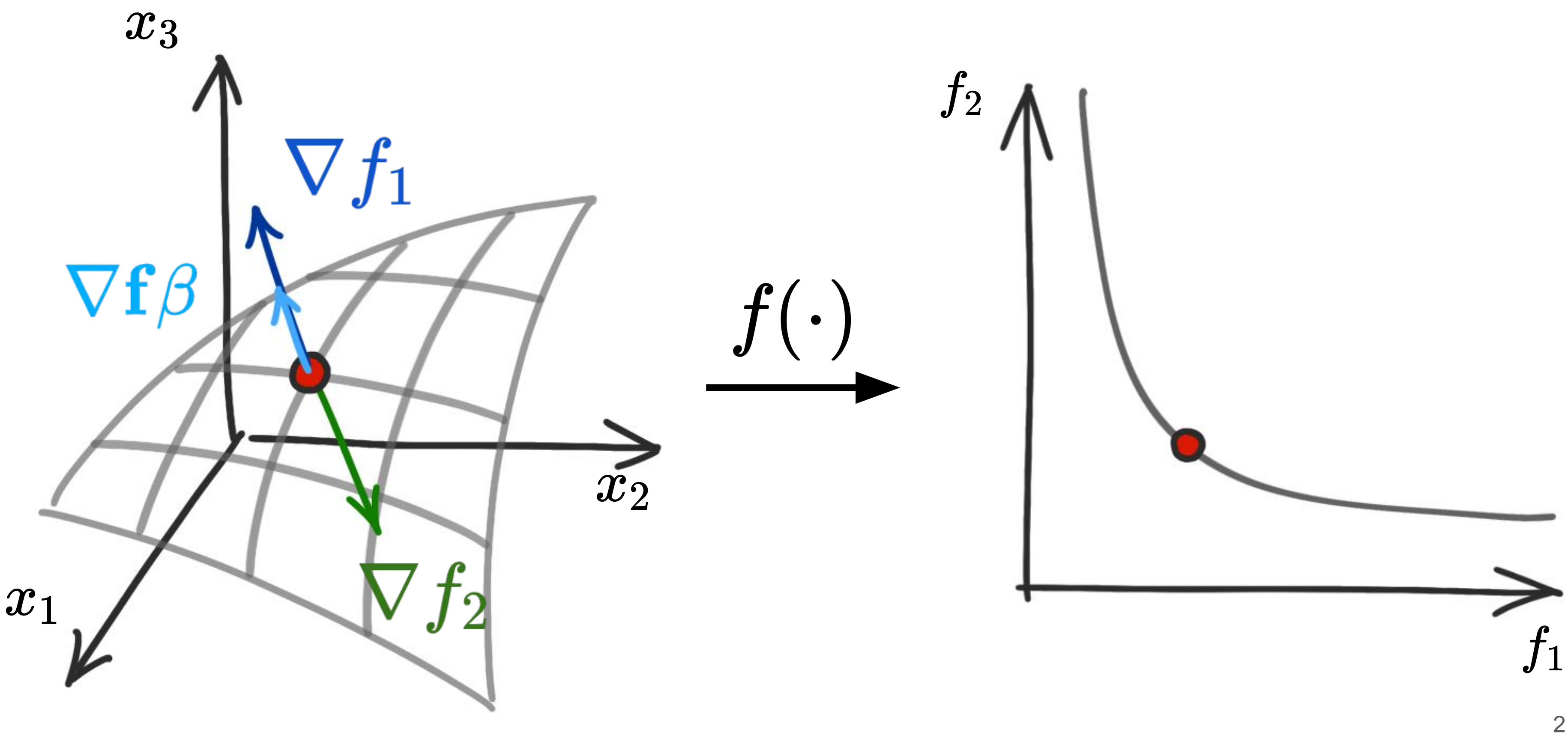
Efficient Pareto-front expansion

$$\mathbf{H}v = \nabla \mathbf{f} \beta \in \text{colspan}\{\nabla f_i\}$$



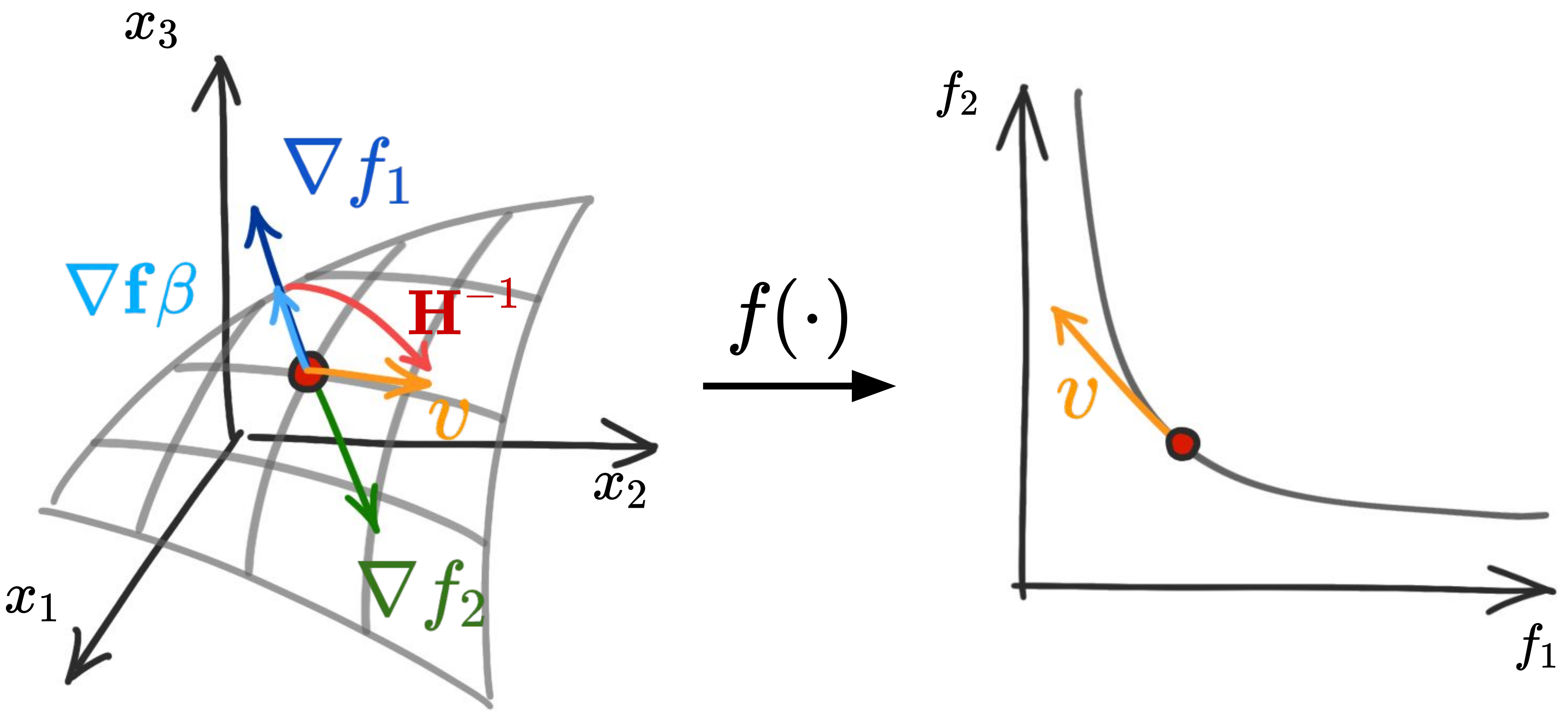
Efficient Pareto-front expansion

$$\mathbf{H}v = \nabla \mathbf{f} \beta \in \mathrm{colspan}\{\nabla f_i\}$$



Efficient Pareto-front expansion

$$\mathbf{H}v = \nabla \mathbf{f}\beta \in \text{colspan}\{\nabla f_i\}$$



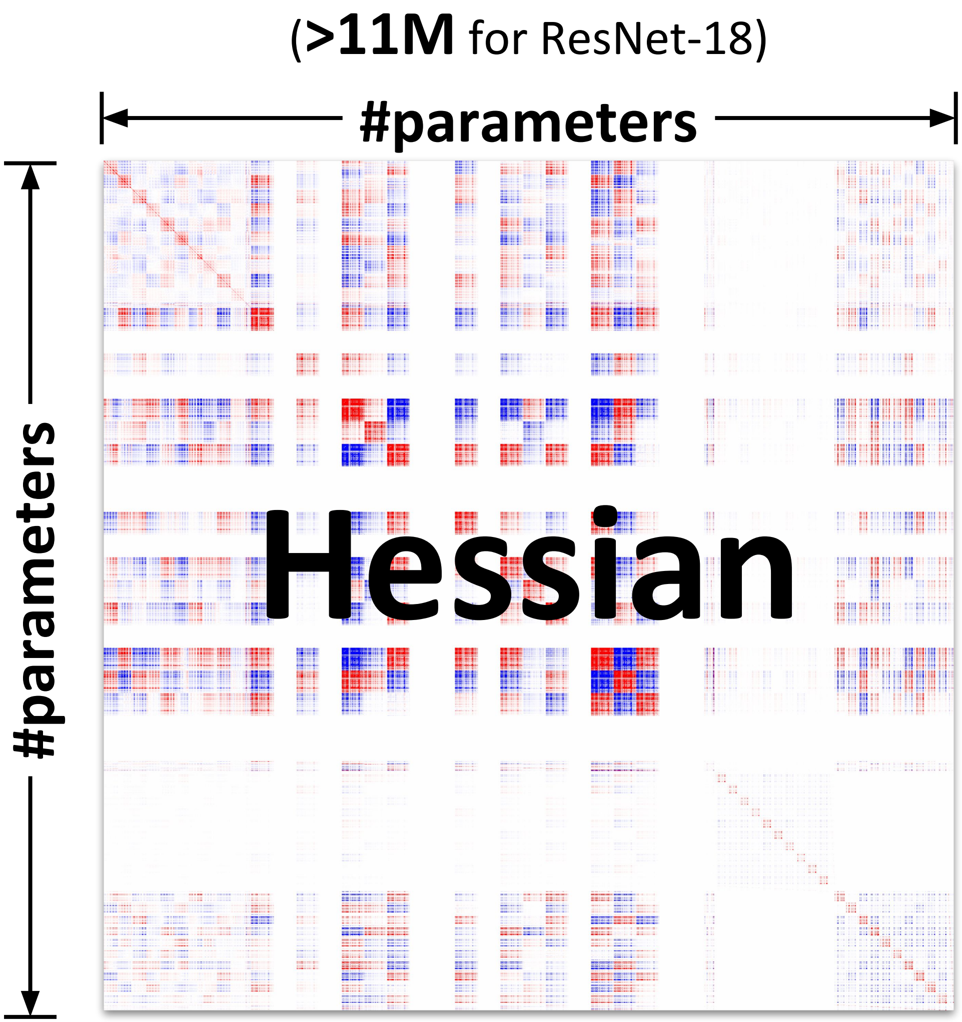
Challenges

- **The most efficient direction for expansion is unknown.**
  - How to recover the Pareto front from one solution?

- **Deep-learning parameter space has large dimensions.**
  - How to scale the method up for large-size problems?

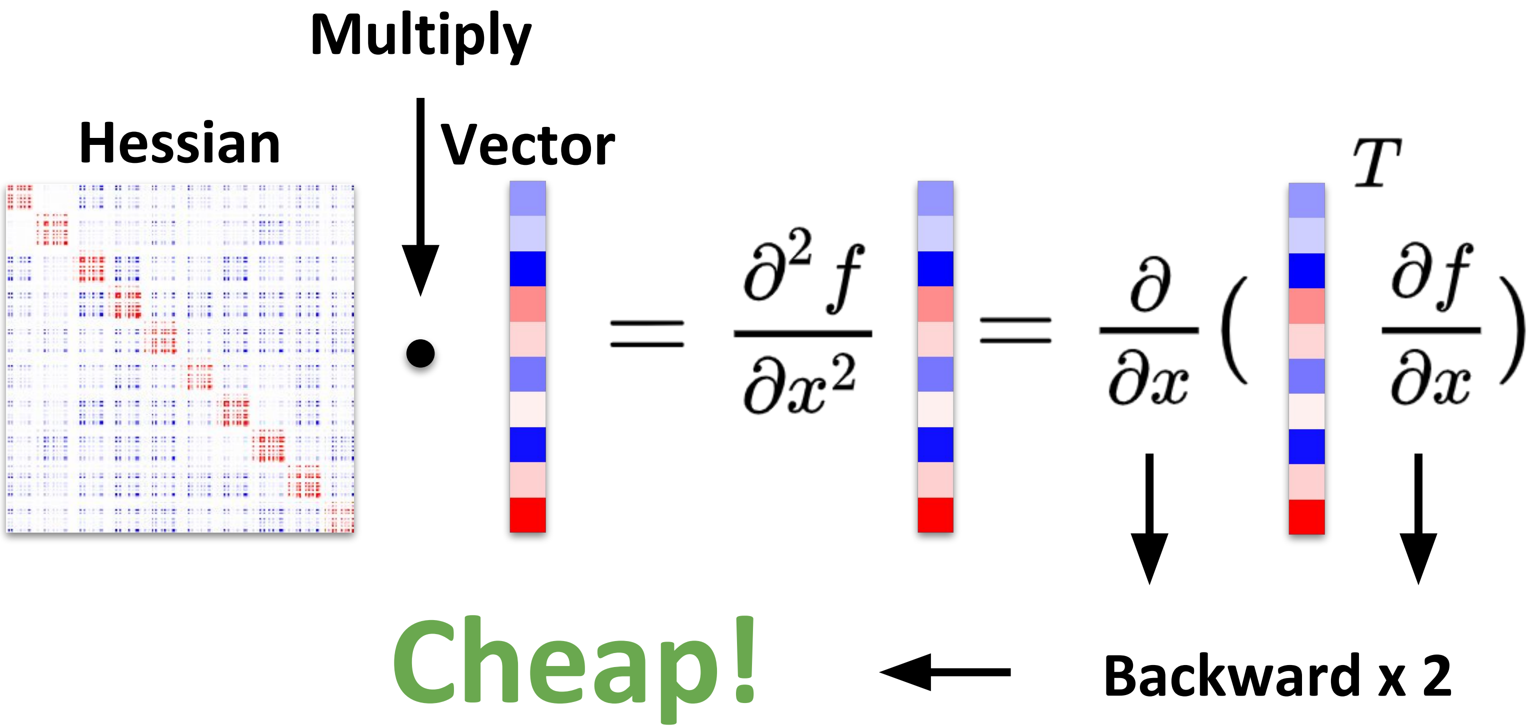
- **Pareto solutions are discrete.**
  - How to build a continuous Pareto front from them?

Scaling it up: the large Hessian issue



- Hessian: **HUGE** (>100 T elements to compute or store)
- Hessian^{-1} : **EXPENSIVE**

Scaling it up: Hessian-vector products (HVP)



$$\frac{\partial^2 f}{\partial x^2}\, v = \frac{\partial}{\partial x}\left(v^T \frac{\partial f}{\partial x} \right)$$

Scaling it up: HVP implementation

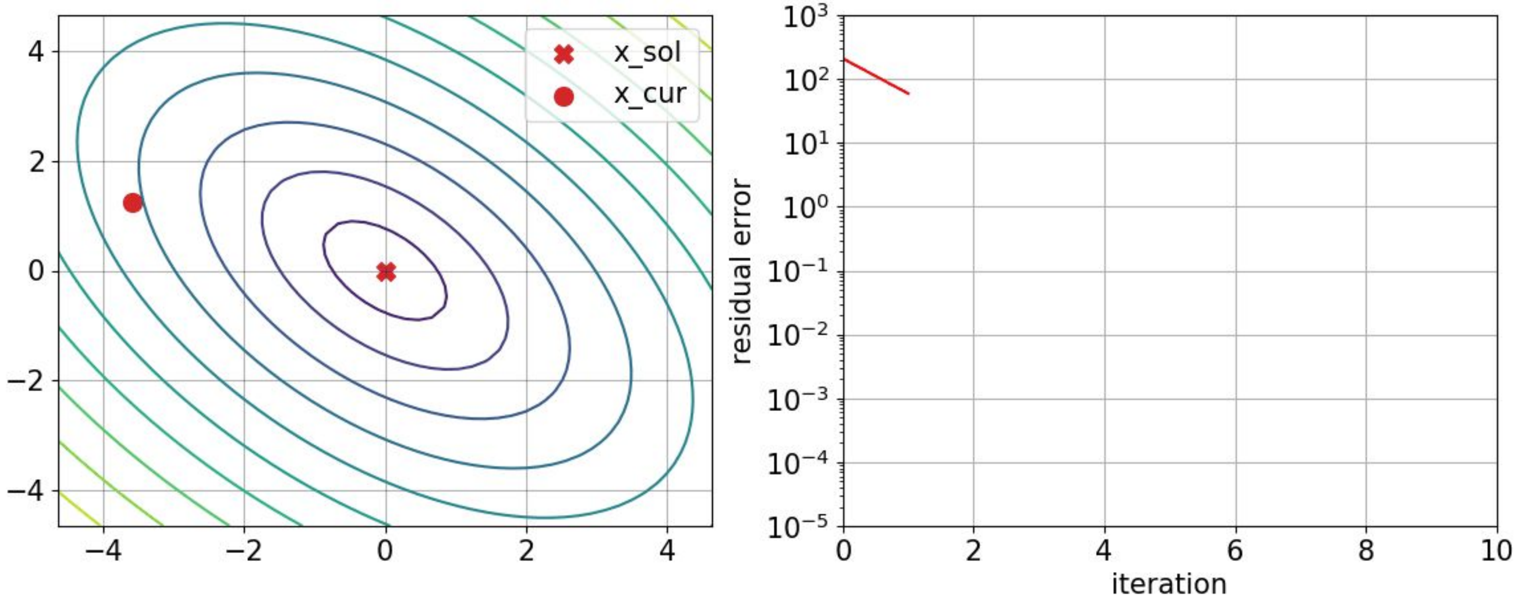
```
def hessian_vector_product(loss, network, v):

    params = network.parameters()

    jacobian = torch.autograd.grad(loss, params, create_graph=True)

    dot = v @ nn.utils.parameters_to_vector(jacobian)

    hvp = torch.autograd.grad(dot, params)

    return nn.utils.parameters_to_vector(hvp)
```

$$\mathbf{H}v = \frac{\partial^2 f}{\partial x^2} v = \frac{\partial}{\partial x}\boxed{\left(v^T \boxed{\frac{\partial f}{\partial x}}\right)}$$

Scaling it up: Krylov subspace method

Key features

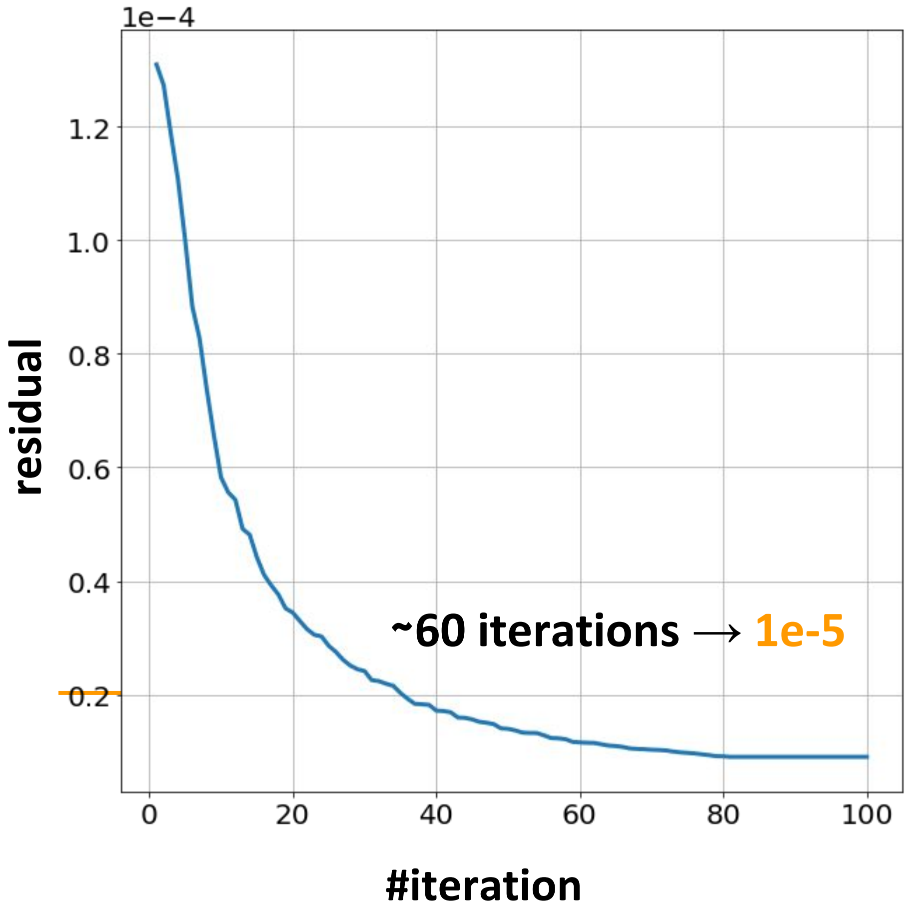
- ❑ More iterations (bounded by the matrix size) = better solutions
- ❑ Each iterations requires matrix-vector products only!



Applying the Conjugate Gradient method to solve a linear system of size 100 x 100.

Scaling it up: Implementation

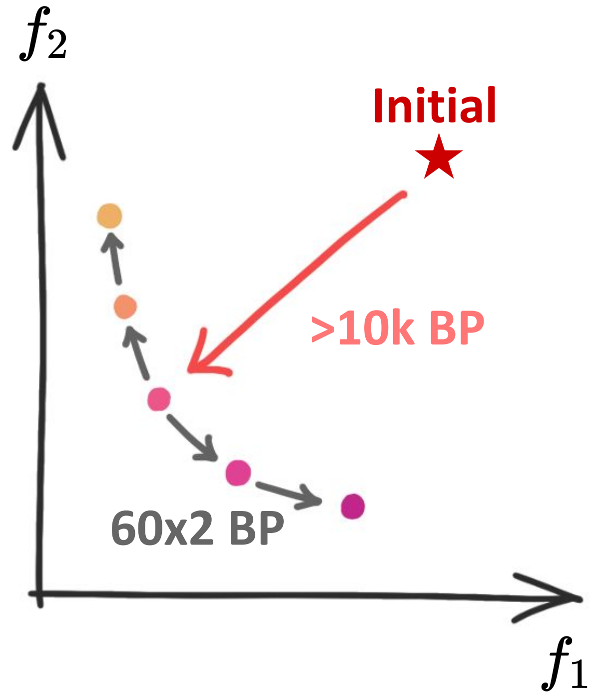
Deep-learning benchmark test (Hessian size = 1500^2)



- Lower residual is better
- Quick convergence compared with the matrix size

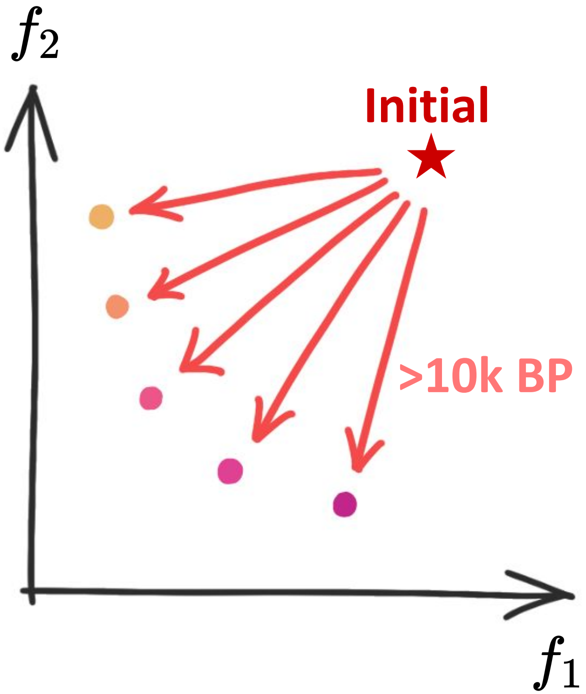
Scaling it up: Implication

Are 60 iterations cheap?



Scaling it up: Implication

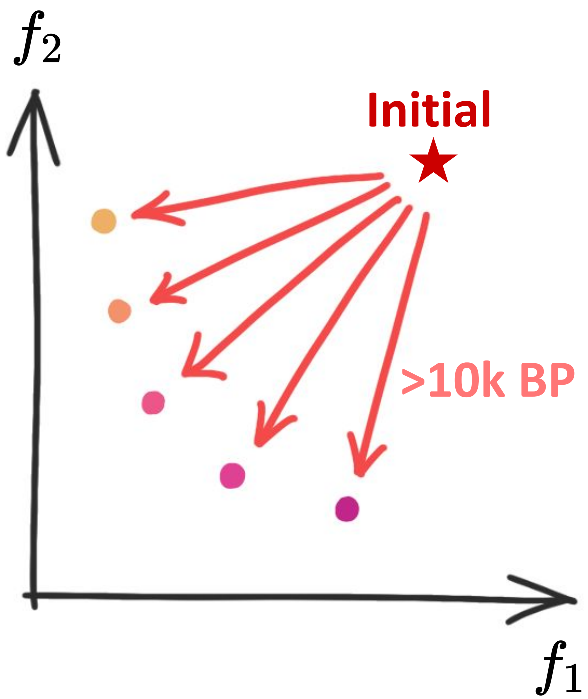
Are 60 iterations cheap?



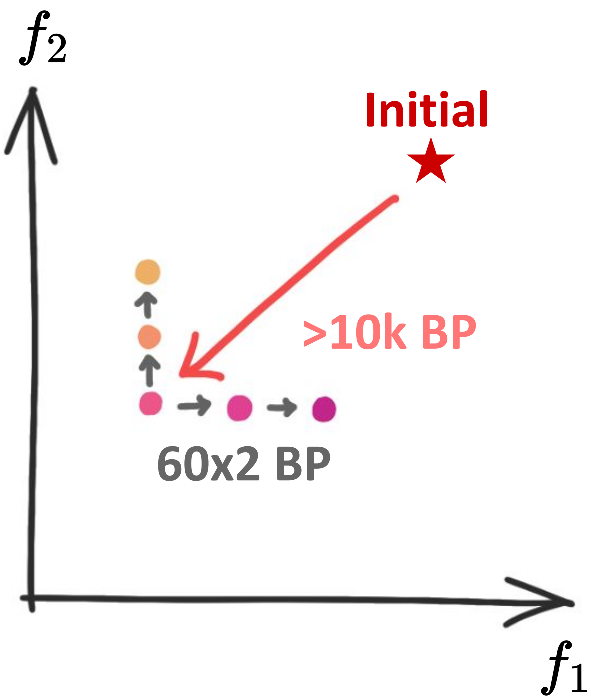
Zero-order method
(re-training)

Scaling it up: Implication

Are 60 iterations cheap?



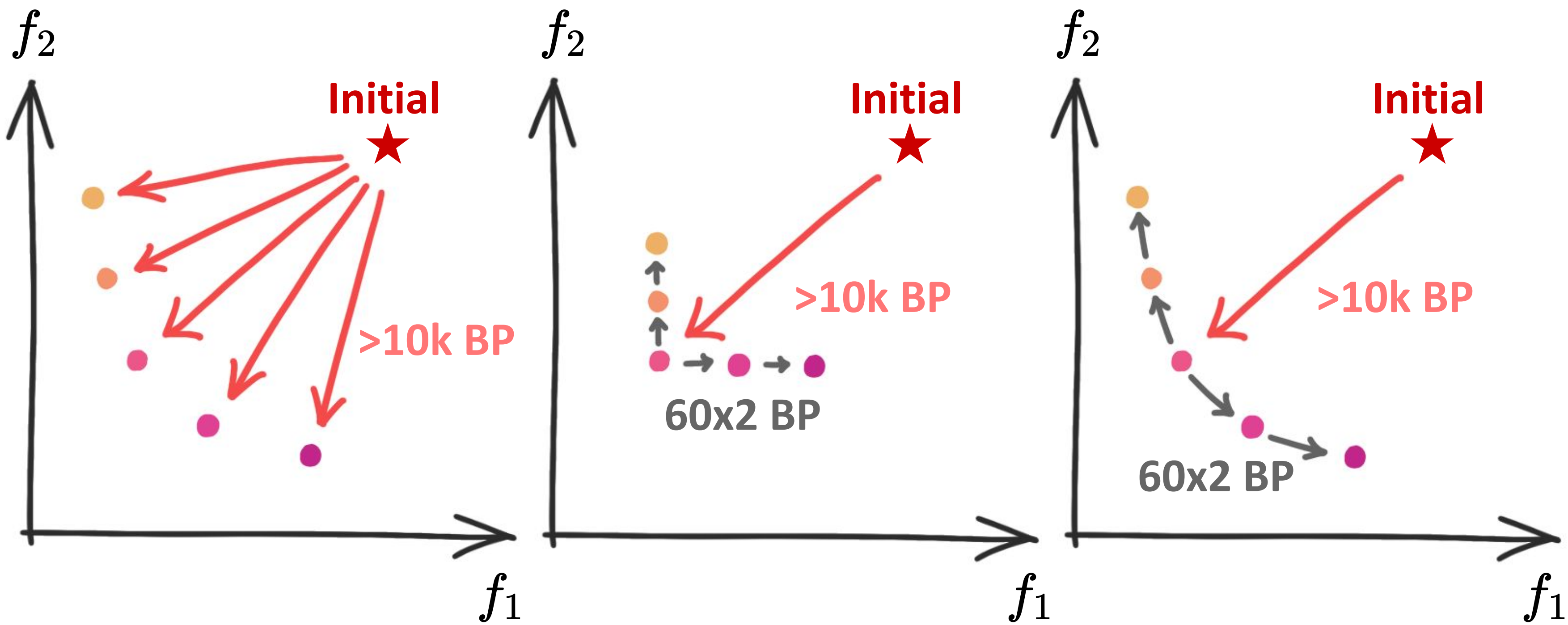
Zero-order method
(re-training)



First-order method
(warm-start SGD)

Scaling it up: Implication

Are 60 iterations cheap? Yes!



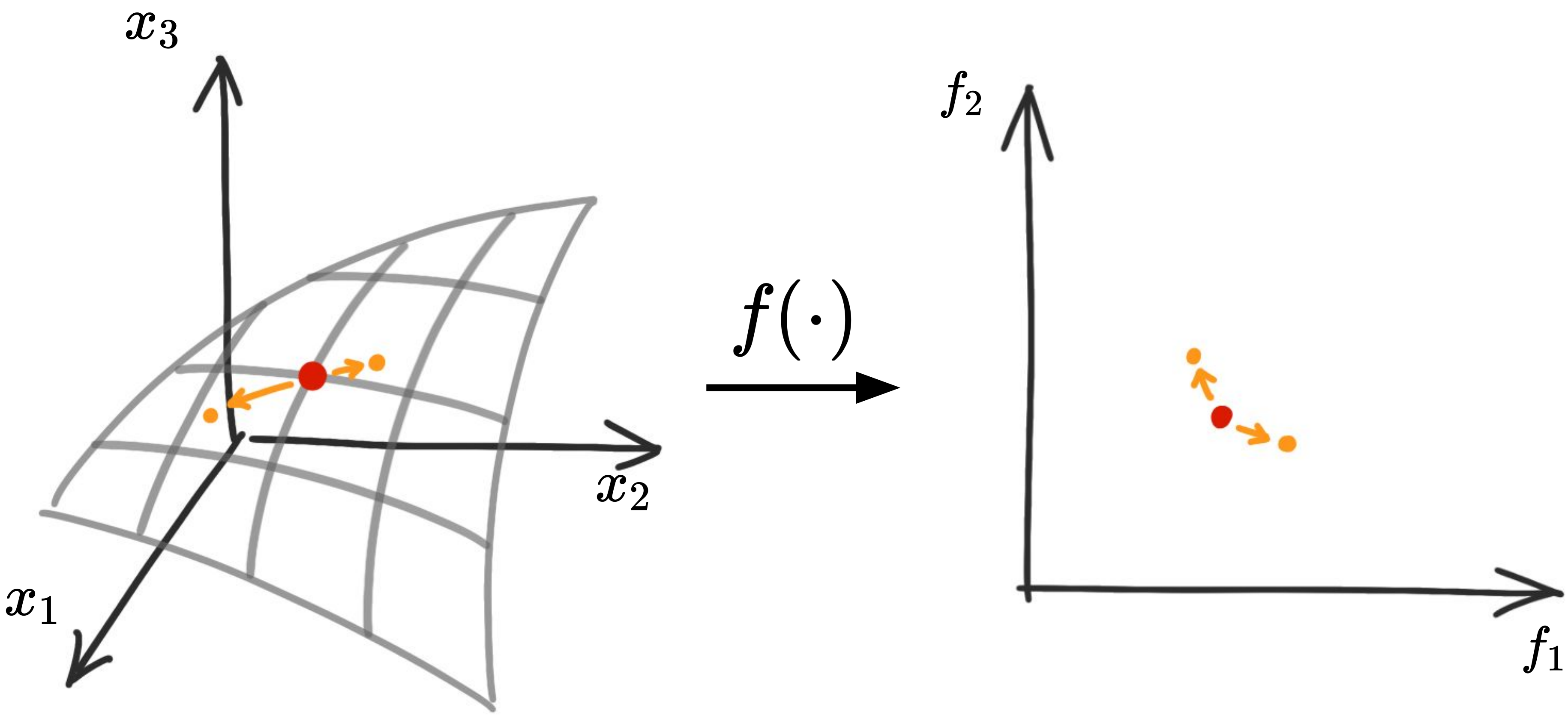
Zero-order method (re-training)	First-order method (warm-start SGD)	**Second-order method (ours)**

Challenges

- **The most efficient direction for expansion is unknown.**
  - How to recover the Pareto front from one solution?
- **Deep-learning parameter space has large dimensions.**
  - How to scale the method up for large-size problems?
- **Pareto solutions are discrete.**
  - How to build a continuous Pareto front from them?

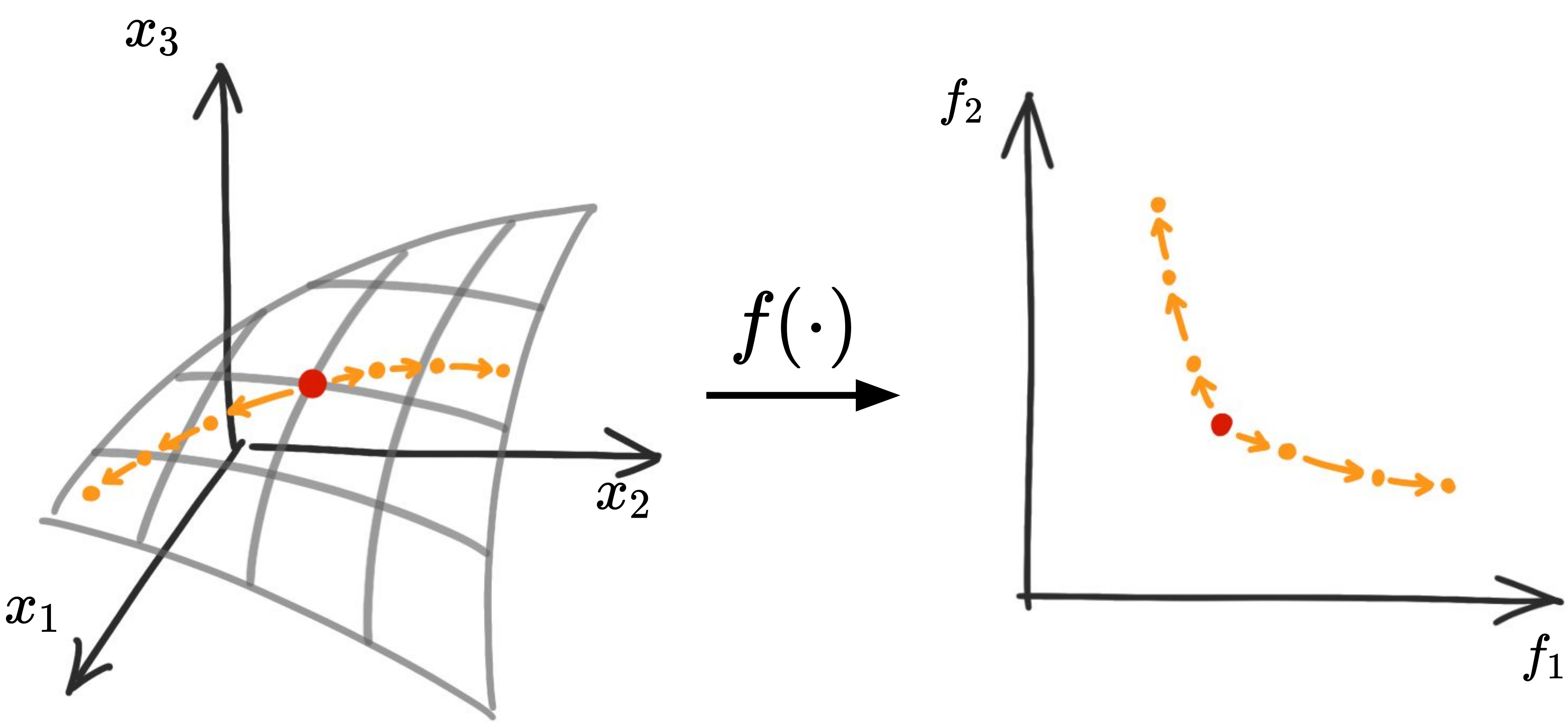
Challenges: Continuous Parameterization

Grow some solutions.



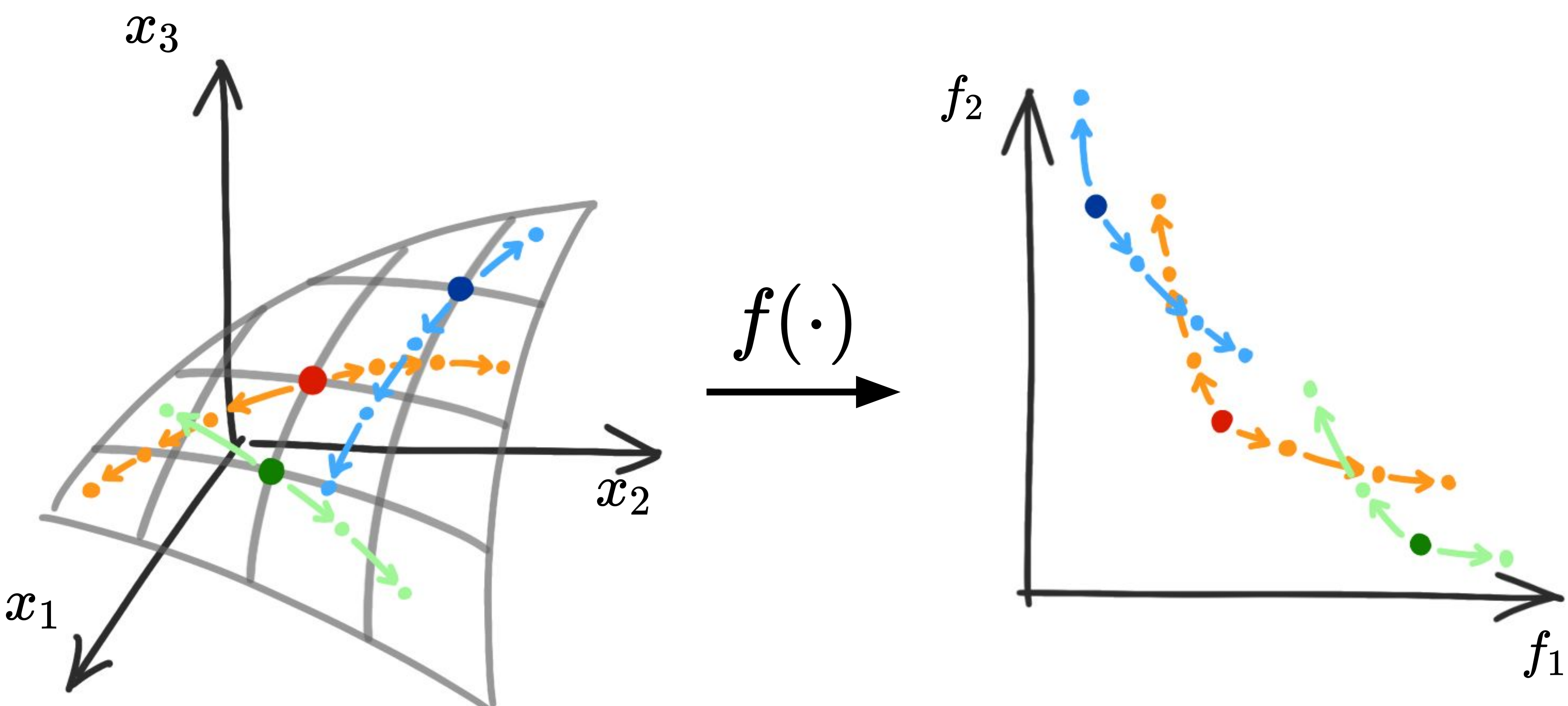
Challenges: Continuous Parameterization

Grow more solutions.



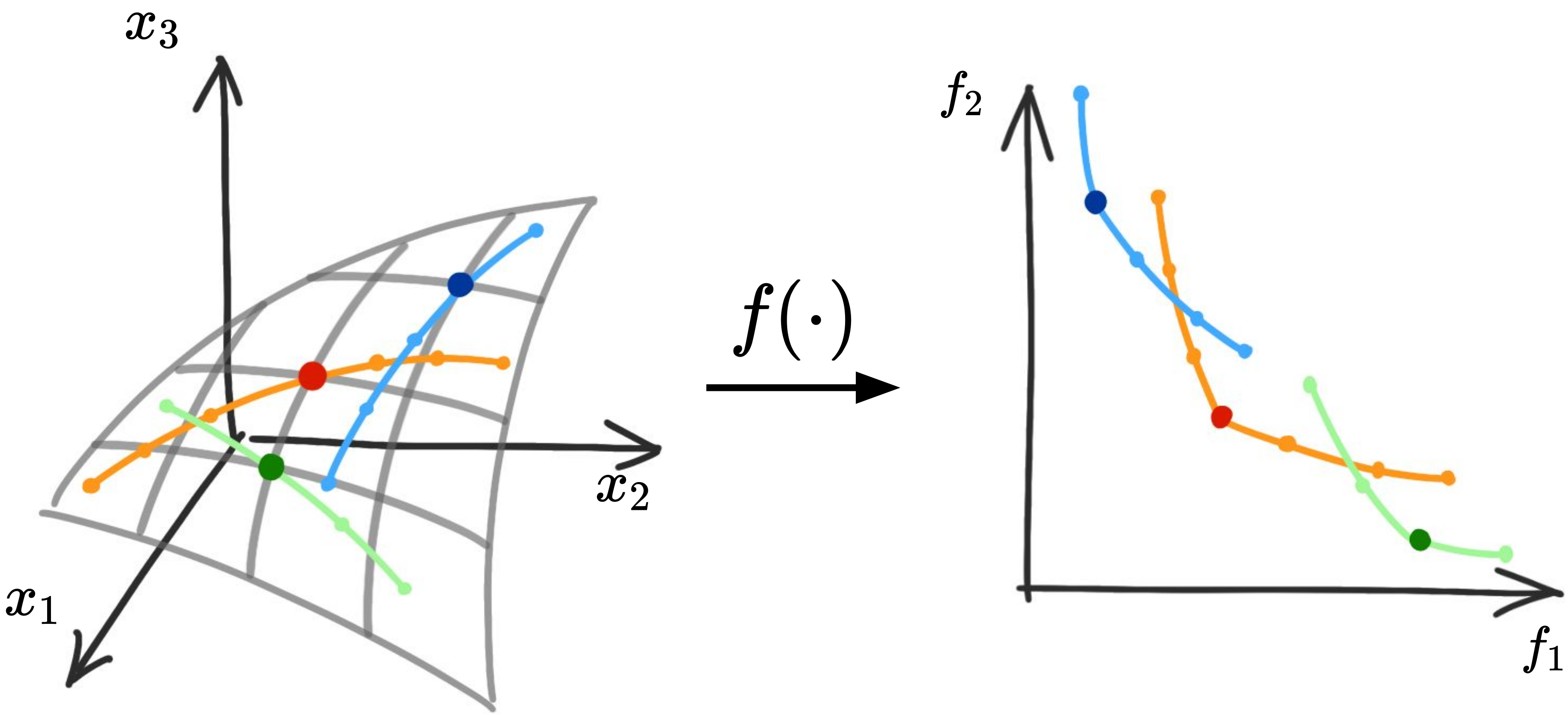
Challenges: Continuous Parameterization

Grow even more solutions from more starting points.



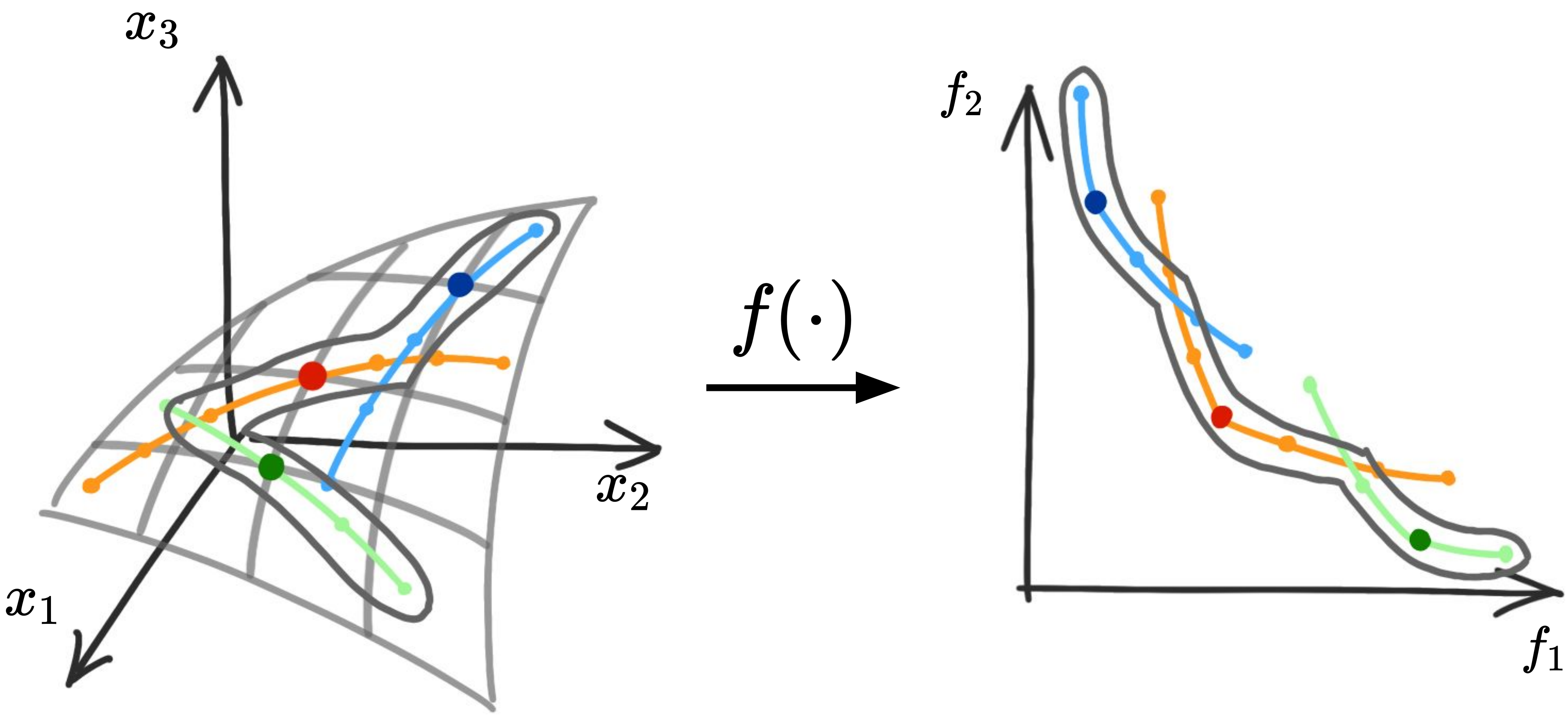
Challenges: Continuous Parameterization

Build continuous parameterization from interpolation.



Challenges: Continuous Parameterization

Filter out others and keep Pareto optimal solutions only.

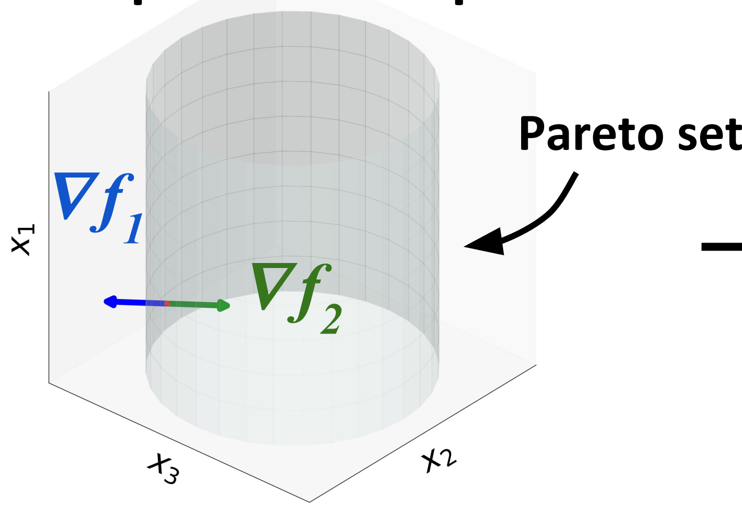


Experiments

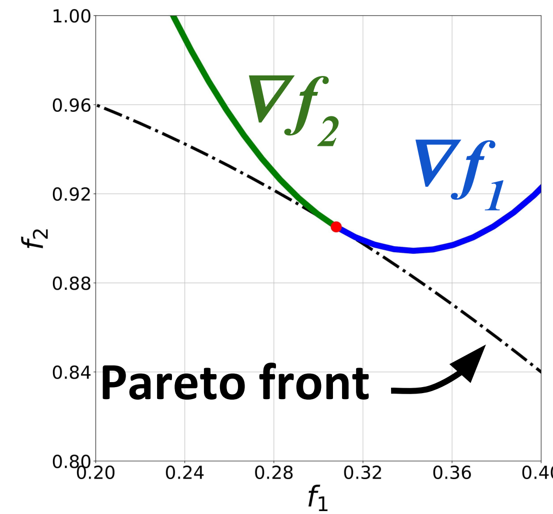
- Synthetic Examples
- Pareto Expansion
- Continuous Parameterization
- Ablation Study

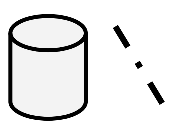
Synthetic Example: ZDT2-Variant

3D parameter space



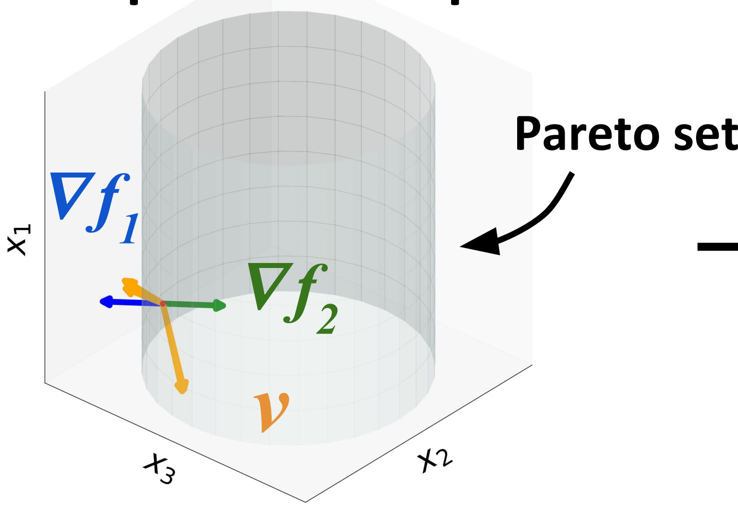
2D performance space



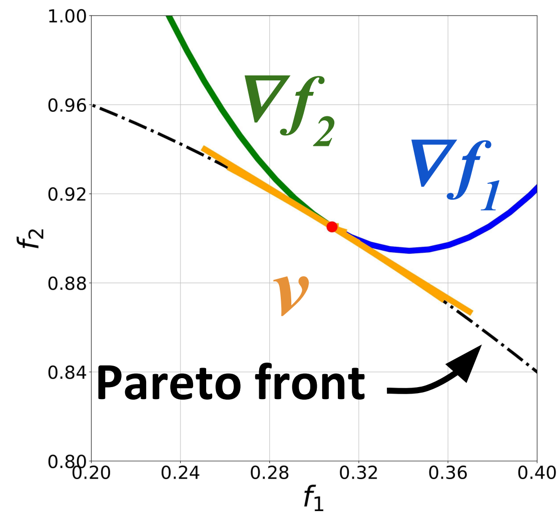
-  = Analytical Pareto set and front
-  = Gradient directions
- Expanding along the gradients deviates from the true Pareto front.

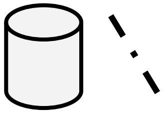
Synthetic Example: ZDT2-Variant

3D parameter space

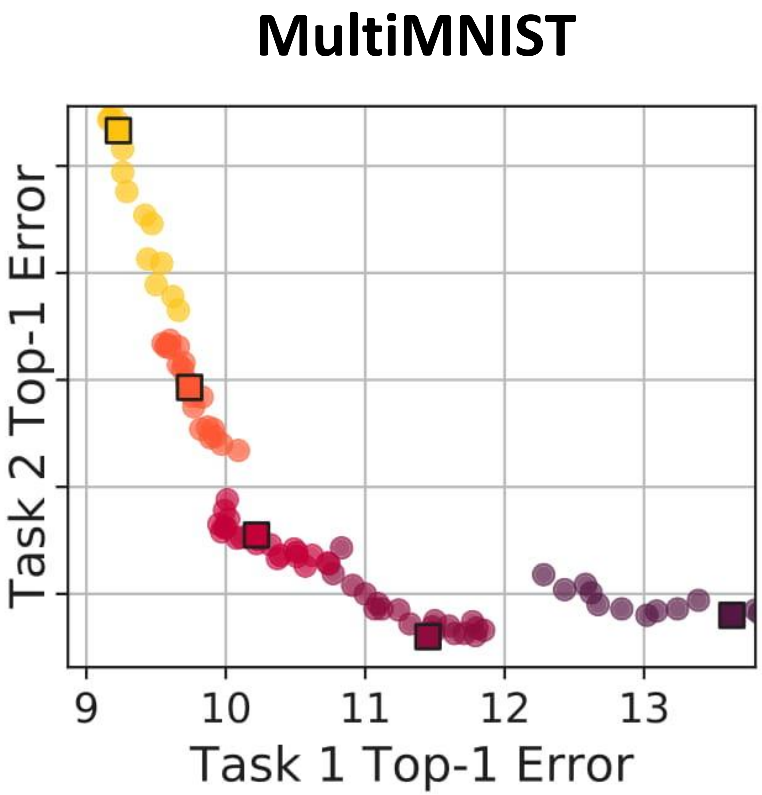


2D performance space



-  = **Analytical Pareto set and front**
- ← = **Our method**
- **Tangent to analytical Pareto set**

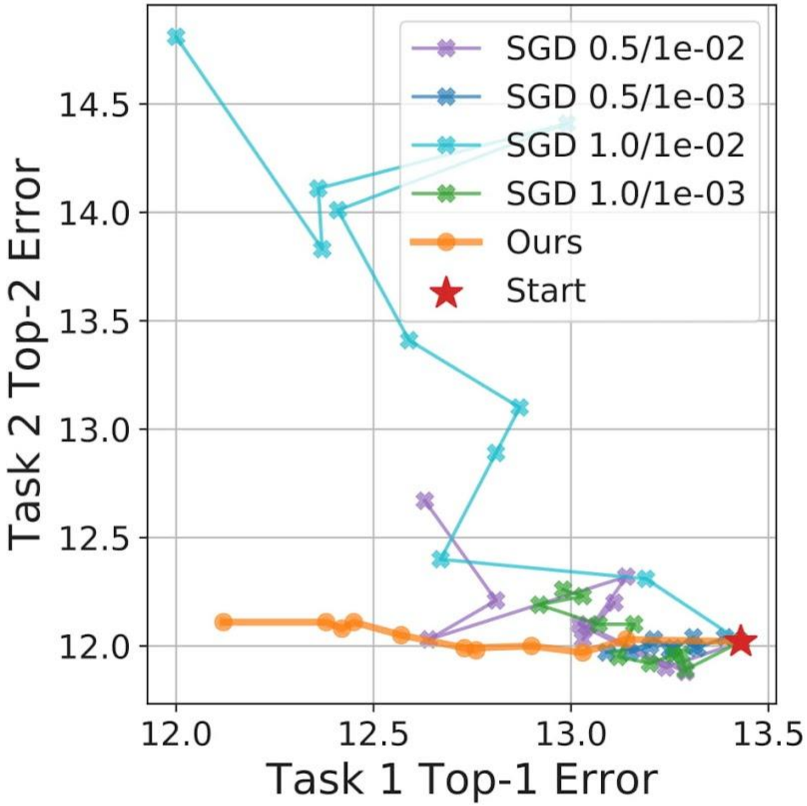
Sufficiency Test



- Color = different start
- Lower left ⇙ = better
- Large coverage

Necessity Test

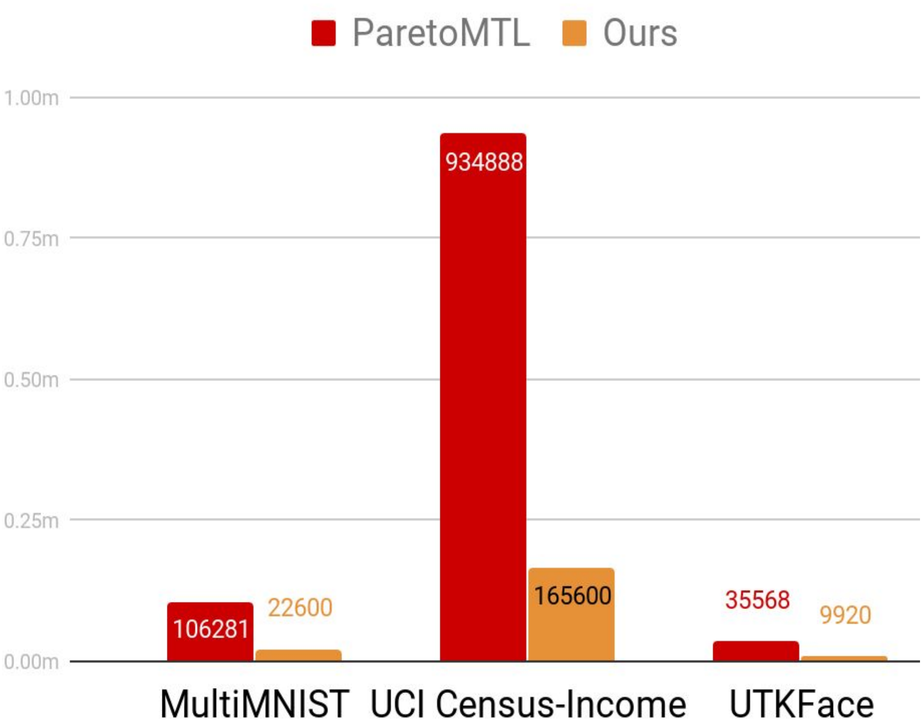
MultiMNIST



- **Expanded from ★**
- **Lower left ⇙ = better**
- **Ours = most effective exploration**

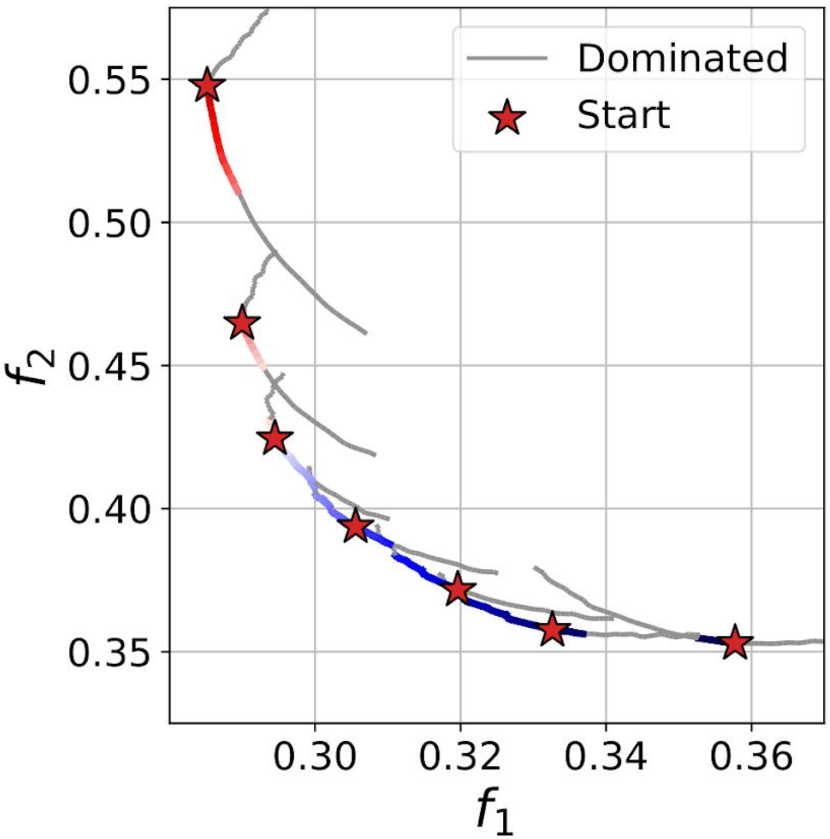
Efficiency Comparison

#Forward/Backward-Propagation



- Our method costs <u>less</u> computation than previous method.

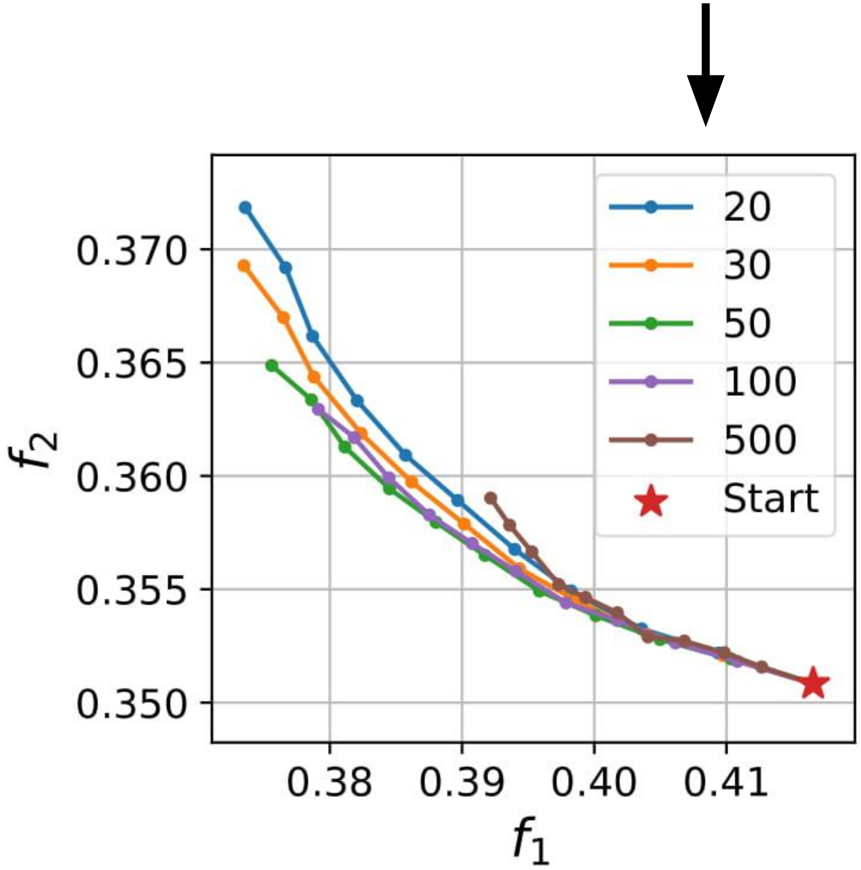
Continuous Parameterization



- Expanded from s
- Red and blue solutions construct the continuous Pareto front
- Gray ones are dominated

Ablation Study: #iteration

#iteration



- Color = #iteration
- Lower left ⇙ = better
- 50 iterations = the best!

Summary

- **Second-order Hessian matrices reveal useful tangent information about Pareto fronts.**

Summary

- **Second-order Hessian matrices reveal useful tangent information about Pareto fronts.**
- **Efficient Hessian-vector product in neural networks unlocks expanding Pareto sets with second-order information.**

Summary

- **Second-order Hessian matrices reveal useful tangent information about Pareto fronts.**
- **Efficient Hessian-vector product in neural networks unlocks expanding Pareto sets with second-order information.**
- **Continuous, first-order accurate Pareto fronts can be obtained by linearly interpolating dense solutions on the tangent plane.**

Thank you!

Acknowledgments

We thank Prof. Tae-Hyun Oh and our reviewers for their feedback.

Code

https://github.com/mit-gfx/ContinuousParetoMTL