

On Functional Gradient Descent Methods

Daniel Csillag
FGV EMAp

Inria Paris – Oct 14, 2026

Joint works with Rodrigo Schuller², Pedro Dall’Antonia¹, Caio Peixoto¹, Leonidas Guibas³,
Luiz Velho², Tiago Novello^{2,3}, Bernardo Costa¹ and Yuri Saporito¹

¹ FGV EMAp ² IMPA ³ Stanford University

Functional Optimization

$$\operatorname{argmin}_{f : \mathcal{X} \rightarrow \mathcal{Y}} L(f)$$

Functional Optimization

$$\operatorname{argmin}_{f : \mathcal{X} \rightarrow \mathcal{Y}} L(f)$$

$$\operatorname{argmin}_{\theta \in \mathbb{R}^d} F(\theta), \quad F(\theta) = L(f_\theta)$$

Functional Optimization

$$\operatorname{argmin}_{f : \mathcal{X} \rightarrow \mathcal{Y}} L(f)$$

$$\operatorname{argmin}_{\theta \in \mathbb{R}^d} F(\theta), \quad F(\theta) = L(f_\theta)$$

Loss & gradient operator:

$$F : \mathbb{R}^d \rightarrow \mathbb{R}, \quad \nabla F : \mathbb{R}^d \rightarrow \mathbb{R}^d$$

Functional Optimization

$$\operatorname{argmin}_{f : \mathcal{X} \rightarrow \mathcal{Y}} L(f)$$

$$\operatorname{argmin}_{\theta \in \mathbb{R}^d} F(\theta), \quad F(\theta) = L(f_\theta)$$

Loss & gradient operator:

$$F : \mathbb{R}^d \rightarrow \mathbb{R}, \quad \nabla F : \mathbb{R}^d \rightarrow \mathbb{R}^d$$

Optimize with gradient descent:

$$\theta_{t+1} = \theta_t - \eta \nabla F(\theta_t)$$

(or some variant thereof, e.g. Adam)

Functional Optimization

$$\operatorname{argmin}_{f: \mathcal{X} \rightarrow \mathcal{Y}} L(f)$$

$$\operatorname{argmin}_{\theta \in \mathbb{R}^d} F(\theta), \quad F(\theta) = L(f_\theta)$$

Loss & gradient operator:

$$F: \mathbb{R}^d \rightarrow \mathbb{R}, \quad \nabla F: \mathbb{R}^d \rightarrow \mathbb{R}^d$$

Optimize with gradient descent:

$$\theta_{t+1} = \theta_t - \eta \nabla F(\theta_t)$$

(or some variant thereof, e.g. Adam)

$$\operatorname{argmin}_{f \in \mathcal{H}} L(f), \quad \mathcal{H} \text{ Hilbert}$$

Functional Optimization

$$\operatorname{argmin}_{f: \mathcal{X} \rightarrow \mathcal{Y}} L(f)$$

$$\operatorname{argmin}_{\theta \in \mathbb{R}^d} F(\theta), \quad F(\theta) = L(f_\theta)$$

Loss & gradient operator:

$$F: \mathbb{R}^d \rightarrow \mathbb{R}, \quad \nabla F: \mathbb{R}^d \rightarrow \mathbb{R}^d$$

Optimize with gradient descent:

$$\theta_{t+1} = \theta_t - \eta \nabla F(\theta_t)$$

(or some variant thereof, e.g. Adam)

$$\operatorname{argmin}_{f \in \mathcal{H}} L(f), \quad \mathcal{H} \text{ Hilbert}$$

Loss & gradient operator:

$$L: \mathcal{H} \rightarrow \mathbb{R}, \quad \nabla L: \mathcal{H} \rightarrow \mathcal{H}$$

Functional Optimization

$$\operatorname{argmin}_{f: \mathcal{X} \rightarrow \mathcal{Y}} L(f)$$

$$\operatorname{argmin}_{\theta \in \mathbb{R}^d} F(\theta), \quad F(\theta) = L(f_\theta)$$

Loss & gradient operator:

$$F: \mathbb{R}^d \rightarrow \mathbb{R}, \quad \nabla F: \mathbb{R}^d \rightarrow \mathbb{R}^d$$

Optimize with gradient descent:

$$\theta_{t+1} = \theta_t - \eta \nabla F(\theta_t)$$

(or some variant thereof, e.g. Adam)

$$\operatorname{argmin}_{f \in \mathcal{H}} L(f), \quad \mathcal{H} \text{ Hilbert}$$

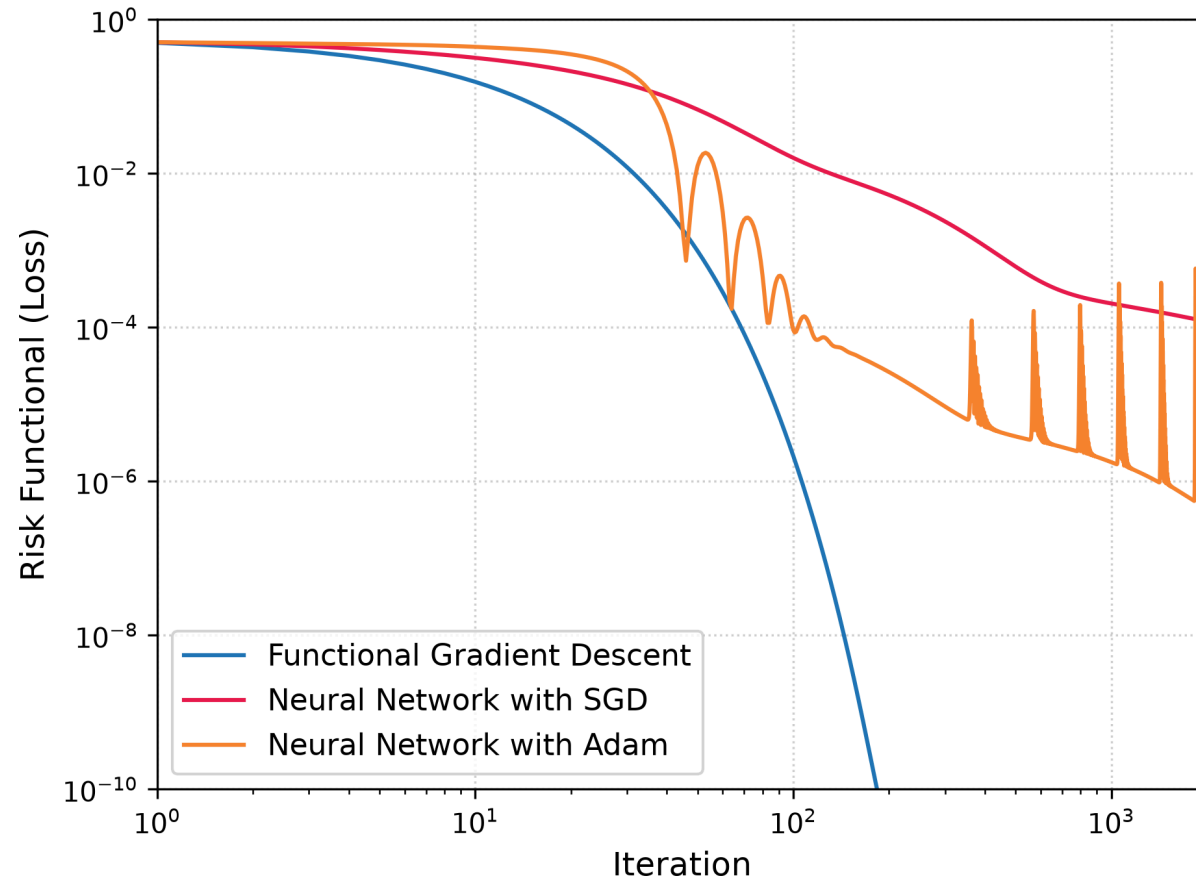
Loss & gradient operator:

$$L: \mathcal{H} \rightarrow \mathbb{R}, \quad \nabla L: \mathcal{H} \rightarrow \mathcal{H}$$

Optimize with *functional* gradient descent:

$$f_{t+1} = f_t - \eta \nabla L(f_t)$$

Functional Optimization



Functional Gradient Descent is Hard

Functional Gradient Descent is Hard

1. FGD is infinite-dimensional – how do we implement that?

Functional Gradient Descent with Adaptive Representations (NeurIPS 2026)

D. Csillag, R. Schuller, P. Dall'Antonia, L. Guibas, L. Velho, T. Novello

(Part 1 of this talk)

2. Functional gradients are often *extremely* hard to compute

Random Gradient-Free Optimization in Infinite Dimensional Spaces (under review)

C. Peixoto, D. Csillag, B. Costa, Y. Saporito

(Part 2 of this talk)

Part 1: How to Discretize Functional GD

Ideal functional gradient descent update:

$$f_{t+1}^{\text{ideal}} = f_t^{\text{ideal}} - \eta \nabla L(f_t^{\text{ideal}})$$

Part 1: How to Discretize Functional GD

Ideal functional gradient descent update:

$$f_{t+1}^{\text{ideal}} = f_t^{\text{ideal}} - \eta \nabla L(f_t^{\text{ideal}})$$

Problem: $\nabla L(f_t^{\text{ideal}})$ is infinite-dimensional, and computers have finite memory.

Part 1: How to Discretize Functional GD

Ideal functional gradient descent update:

$$f_{t+1}^{\text{ideal}} = f_t^{\text{ideal}} - \eta \nabla L(f_t^{\text{ideal}})$$

Problem: $\nabla L(f_t^{\text{ideal}})$ is infinite-dimensional, and computers have finite memory.

What we *can* do:

$$f_{t+1} = f_t - \eta g_t, \quad \text{with} \quad g_t \approx \nabla L(f_t).$$

Part 1: How to Discretize Functional GD

Ideal functional gradient descent update:

$$f_{t+1}^{\text{ideal}} = f_t^{\text{ideal}} - \eta \nabla L(f_t^{\text{ideal}})$$

Problem: $\nabla L(f_t^{\text{ideal}})$ is infinite-dimensional, and computers have finite memory.

What we *can* do:

$$f_{t+1} = f_t - \eta g_t, \quad \text{with} \quad g_t \approx \nabla L(f_t).$$

→ Can we guarantee convergence of this approximate process?

Formal Setup

We have a Hilbert space $\mathcal{H}$, loss $L : \mathcal{H} \rightarrow \mathbb{R}$ and gradient operator $\nabla L : \mathcal{H} \rightarrow \mathcal{H}$.

Formal Setup

We have a Hilbert space $\mathcal{H}$, loss $L : \mathcal{H} \rightarrow \mathbb{R}$ and gradient operator $\nabla L : \mathcal{H} \rightarrow \mathcal{H}$.

Our updates:

$$f_{t+1} = f_t - \eta g_t,$$

for some direction $g_t \in \mathcal{H}$.

Formal Setup

We have a Hilbert space $\mathcal{H}$, loss $L : \mathcal{H} \rightarrow \mathbb{R}$ and gradient operator $\nabla L : \mathcal{H} \rightarrow \mathcal{H}$.

Our updates:

$$f_{t+1} = f_t - \eta g_t,$$

for some direction $g_t \in \mathcal{H}$.

Formal Setup

We have a Hilbert space $\mathcal{H}$, loss $L : \mathcal{H} \rightarrow \mathbb{R}$ and gradient operator $\nabla L : \mathcal{H} \rightarrow \mathcal{H}$.

Our updates:

$$f_{t+1} = f_t - \eta g_t,$$

~~for some direction $g_t \in \mathcal{H}$.~~

for some direction $g_t \in \mathcal{B}$ in a larger *Banach* space $\mathcal{B}$ (i.e., $\mathcal{H} \hookrightarrow \mathcal{B}$).

Formal Setup

We have a Hilbert space $\mathcal{H}$, loss $L : \mathcal{H} \rightarrow \mathbb{R}$ and gradient operator $\nabla L : \mathcal{H} \rightarrow \mathcal{H}$.

Our updates:

$$f_{t+1} = f_t - \eta g_t,$$

~~for some direction $g_t \in \mathcal{H}$.~~

for some direction $g_t \in \mathcal{B}$ in a larger *Banach* space $\mathcal{B}$ (i.e., $\mathcal{H} \hookrightarrow \mathcal{B}$).

Assumption (Extension). The loss and its gradient admit extensions

$$L : \mathcal{B} \rightarrow \mathbb{R} \quad \text{and} \quad \nabla L : \mathcal{B} \rightarrow \mathcal{B}.$$

A Descent Lemma

Lemma (Sufficient descent). If the loss L is K -smooth and satisfies bridging conditions between $\mathcal{H}$ and $\mathcal{B}$ with constants $\alpha, \beta > 0$, then

$$L(f_{t+1}) \leq L(f_t) - \eta c \|\nabla L(f_t)\|_{\mathcal{B}}^2,$$

where

$$c = \alpha - \frac{K\eta}{2} - \left(\beta + \frac{3}{2}K\eta\right) \frac{\text{RelErr}_t}{1 - \text{RelErr}_t}, \quad \text{RelErr}_t := \frac{\|g_t - \nabla L(f_t)\|_{\mathcal{B}}}{\|g_t\|_{\mathcal{B}}} \leq \frac{1}{2}.$$

Convergence

Theorem (Convergence to a stationary point). If the loss L is lower bounded and

$$\text{RelErr}_t \leq \varepsilon, \quad \text{for all } t = 0, \dots, T - 1,$$

for some $\varepsilon < \min\{1/2, \alpha/(\alpha + \beta)\}$, then, for a sufficiently small learning rate η ,

$$\min_{t=0, \dots, T-1} \|\nabla L(f_t)\|_{\mathcal{B}}^2 \leq \frac{L(f_0) - L^*}{T\eta r},$$

for some $r > 0$.

Convergence

Theorem (Convergence to a stationary point). If the loss L is lower bounded and

$$\text{RelErr}_t \leq \varepsilon, \quad \text{for all } t = 0, \dots, T - 1,$$

for some $\varepsilon < \min\{1/2, \alpha/(\alpha + \beta)\}$, then, for a sufficiently small learning rate η ,

$$\min_{t=0, \dots, T-1} \|\nabla L(f_t)\|_{\mathcal{B}}^2 \leq \frac{L(f_0) - L^*}{T\eta r},$$

for some $r > 0$.

Theorem (Convergence to a global minimizer). If furthermore L is μ -PL in $\mathcal{B}$, then

$$L(f_T) - L^* \leq [1 - 2\eta\beta^{-2}\mu r]^T (L(f_0) - L^*).$$

Ensuring the relative error bound

$$\text{RelErr}_t = \frac{\|g_t - \nabla L(f_t)\|_{\mathcal{B}}}{\|g_t\|_{\mathcal{B}}} \leq \varepsilon, \quad \text{for all } t = 0, \dots, T - 1,$$

Ensuring the relative error bound

$$\text{RelErr}_t = \frac{\|g_t - \nabla L(f_t)\|_{\mathcal{B}}}{\|g_t\|_{\mathcal{B}}} \leq \varepsilon, \quad \text{for all } t = 0, \dots, T - 1,$$

Note: we can compute $\|g_t\|_{\mathcal{B}}$ and form an upper bound $U_t(g) \geq \|g - \nabla L(f_t)\|_{\mathcal{B}}, \forall g$

Ensuring the relative error bound

$$\text{RelErr}_t = \frac{\|g_t - \nabla L(f_t)\|_{\mathcal{B}}}{\|g_t\|_{\mathcal{B}}} \leq \varepsilon, \quad \text{for all } t = 0, \dots, T - 1,$$

Note: we can compute $\|g_t\|_{\mathcal{B}}$ and form an upper bound $U_t(g) \geq \|g - \nabla L(f_t)\|_{\mathcal{B}}, \forall g$

Therefore, a simple idea is to use “**adaptive representations**”:

- Take a sequence of approximations $(g_t^0, g_t^1, g_t^2, \dots)$ such that $U_t(g_t^k) \rightarrow 0$
- For $k = 0, 1, 2, \dots$:
 - If $U_t(g_t^k) / \|g_t^k\|_{\mathcal{B}} \leq \varepsilon$, use $g_t = g_t^k$
 - Otherwise, continue

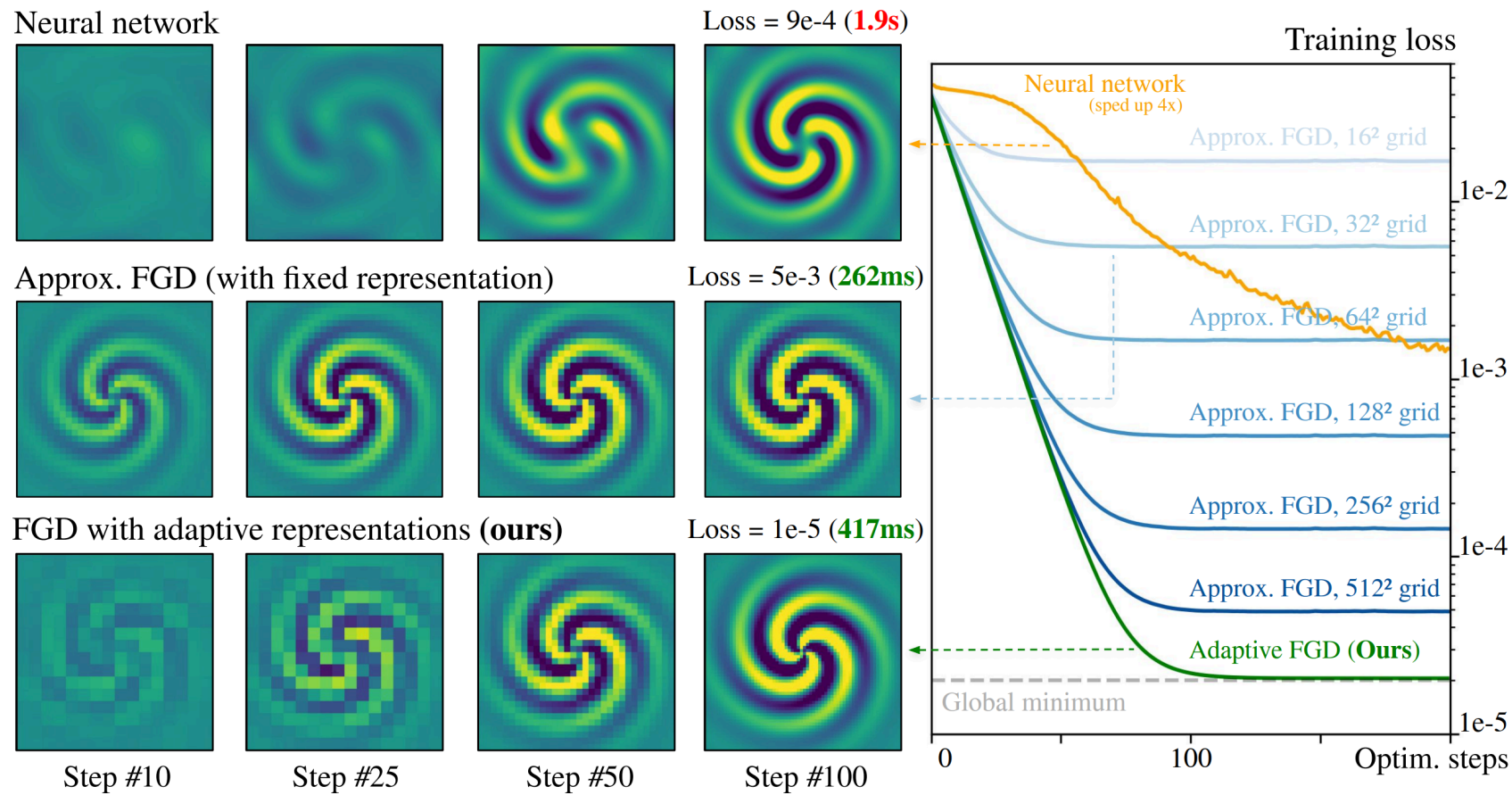
Example 1: A toy fitting problem

Consider the toy function approximation problem

$$\operatorname{argmin}_{f \in L^2} \frac{1}{2} \|f - f^\star\|^2 + \text{const.},$$

for a target function $f^\star : [0, 1]^2 \rightarrow \mathbb{R}$ in L^2 .

Example 1: A toy fitting problem



Example 2: Solving PDEs

A wave equation:

$$\partial_{tt}f = c^2 \Delta f \quad \text{with} \quad \begin{aligned} f(0, x) &= h(x) \\ \partial_t f(0, x) &= 0 \end{aligned}$$

No general closed form solution for dimension ≥ 2 .

Example 2: Solving PDEs

A wave equation:

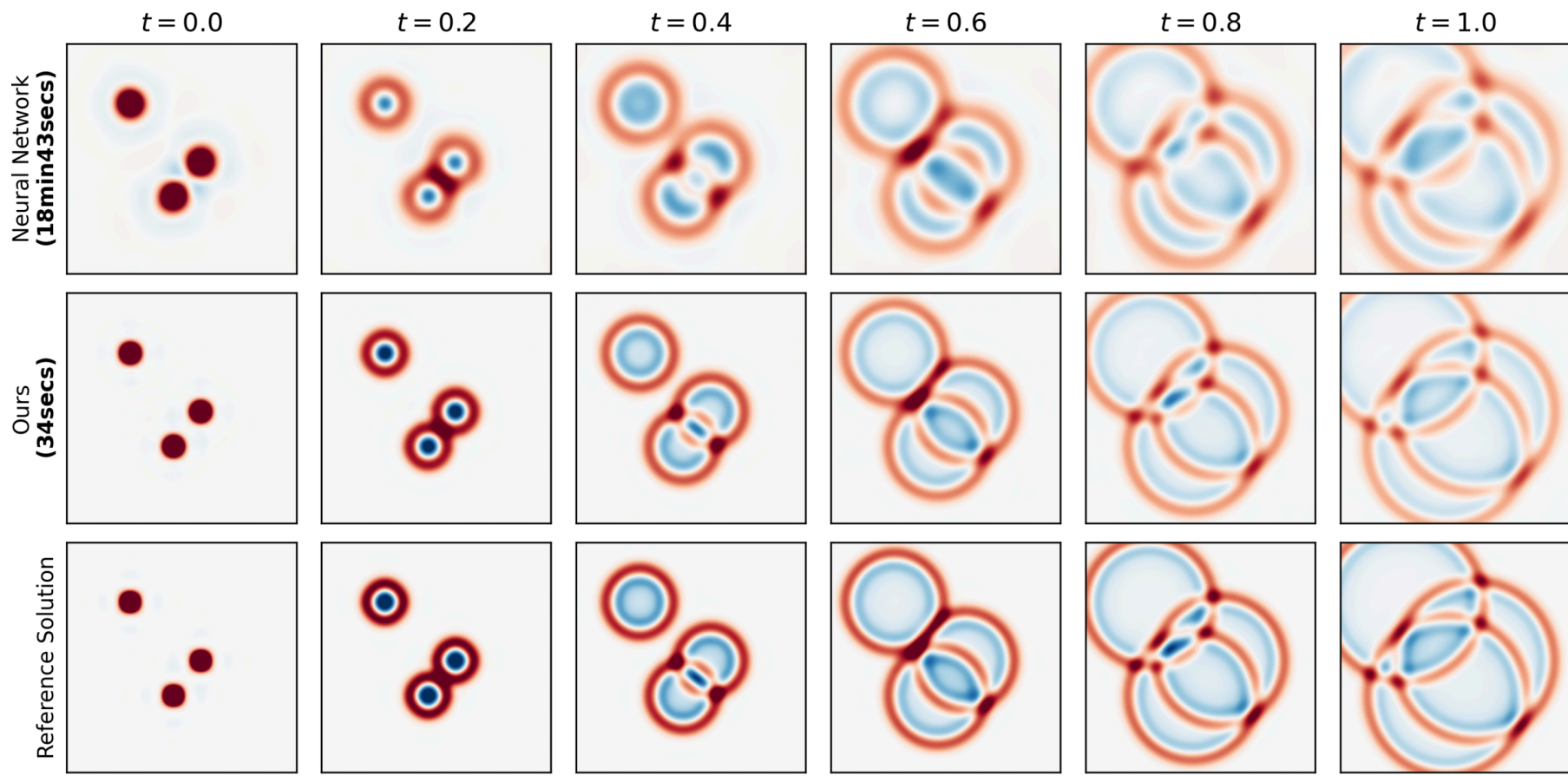
$$\partial_{tt}f = c^2 \Delta f \quad \text{with} \quad \begin{aligned} f(0, x) &= h(x) \\ \partial_t f(0, x) &= 0 \end{aligned}$$

No general closed form solution for dimension ≥ 2 .

As an optimization problem:

$$\operatorname{argmin}_{f(\cdot)} \underbrace{\frac{1}{2} \|\partial_{tt}f - c^2 \Delta f\|_{L^2}^2}_{\text{diff. eq.}} + \underbrace{\frac{1}{2} \|f(0, \cdot) - h\|_{L^2}^2 + \frac{1}{2} \|\partial_t f(0, \cdot)\|_{L^2}^2}_{\text{init conds.}}$$

Example 2: Solving PDEs



Example 3: 3D reconstruction



Example 3: 3D reconstruction



density: $\sigma : \mathbb{R}^3 \rightarrow \mathbb{R}$

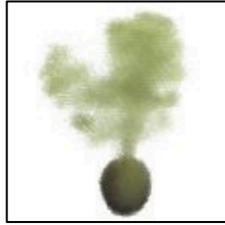
color: $c : \mathbb{R}^3 \times S^2 \rightarrow \mathbb{R}^3$

As an optimization problem:

$$\operatorname{argmin}_{\sigma(\cdot), c(\cdot)} \frac{1}{n} \sum_{i=1}^n \frac{1}{2} \left\| \operatorname{render}_{\sigma, c}(C_i) - Y_i \right\|^2$$

Example 3: 3D reconstruction

Neural network



Loss = $5.3\text{e-}3$ (**7m 21s**)

Approx. FGD (with fixed representation)



Loss = $5.0\text{e-}3$ (**52s**)

FGD with adaptive representations (**ours**)



Loss = $3.4\text{e-}3$ (**4m 36s**)

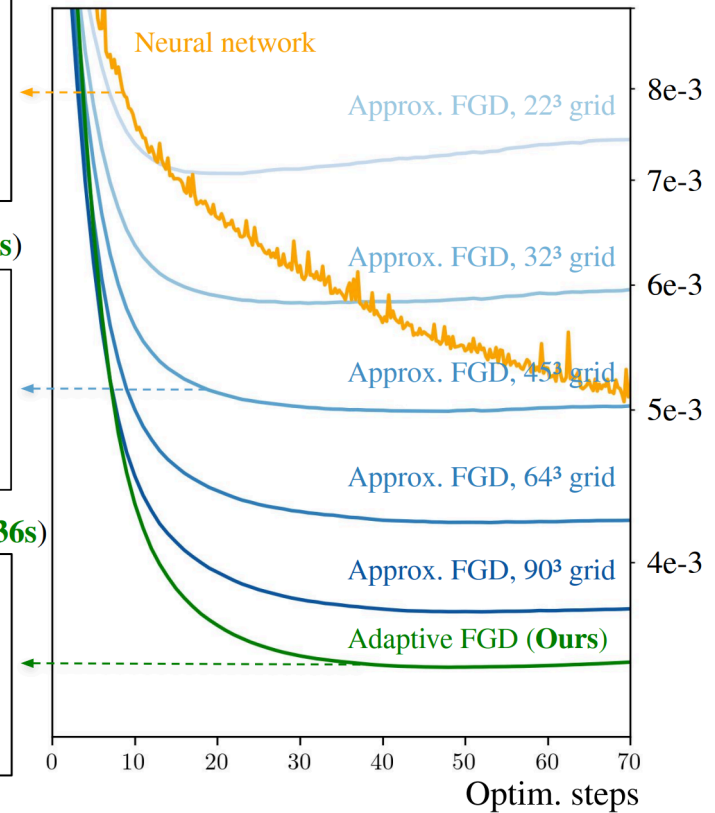
Step #8

Step #16

Step #32

Step #64

Test loss



Part 2: Functional Gradients are Hard

Problem: it's really hard to compute functional gradients.

Part 2: Functional Gradients are Hard

Problem: it's really hard to compute functional gradients.

Example: Solving a PDE

$$\begin{cases} R[f](x) = 0 & x \in \Omega \\ f(x) = h(x) & x \in \partial\Omega \end{cases}$$

where R is a differential operator; consider the loss functional

$$L : \mathbb{H}^s(\Omega) \rightarrow \mathbb{R}$$

$$L(f) = \frac{1}{2} \|R[f]\|_{L^2(\Omega)}^2 + \frac{1}{2} \|f - h\|_{L^2(\partial\Omega)}^2$$

Part 2: Functional Gradients are Hard

Full gradients are ***hard***: $\nabla L(f)$ is the unique $g \in \mathbb{H}^s(\Omega)$ such that

$$\sum_{j=0}^s \int_{\Omega} D^{(j)} g(x) \cdot D^{(j)} v(x) \, dx = \int_{\Omega} R[f](x) R[v](x) \, dx + \int_{\partial\Omega} (f(x) - h(x)) v(x) \, \sigma(dx)$$

for all $v \in \mathbb{H}^s(\Omega)$.

Part 2: Functional Gradients are Hard

Full gradients are ***hard***: $\nabla L(f)$ is the unique $g \in \mathbb{H}^s(\Omega)$ such that

$$\sum_{j=0}^s \int_{\Omega} D^{(j)} g(x) \cdot D^{(j)} v(x) \, dx = \int_{\Omega} R[f](x) R[v](x) \, dx + \int_{\partial\Omega} (f(x) - h(x)) v(x) \, \sigma(dx)$$

for all $v \in \mathbb{H}^s(\Omega)$.

But directional derivatives are easy:

$$\begin{aligned} DL(f; v) &= \lim_{\delta \rightarrow 0} \frac{L(f + \delta v) - L(f)}{\delta} \\ &= \left[\frac{d}{d\delta} L(f + \delta v) \right]_{\delta=0} \end{aligned}$$

Grad. Descent with Directional Derivatives

Q: Can we use just directional derivatives for optimization?

Grad. Descent with Directional Derivatives

Q: Can we use just directional derivatives for optimization?

A: In finite dimensions, yes: Random 'Gradient-Free' Optimization

Grad. Descent with Directional Derivatives

Q: Can we use just directional derivatives for optimization?

A: In finite dimensions, yes: Random ‘Gradient-Free’ Optimization

To estimate $\nabla L(f)$, sample $v_1, \dots, v_M \sim \mathbb{P}_v$, and compute:

$$f, v_1, \dots, v_M \in \mathbb{R}^d$$

$$\hat{g} = \frac{1}{M} \sum_{m=1}^M \text{DL}(f; v_m) v_m$$

Grad. Descent with Directional Derivatives

Q: Can we use just directional derivatives for optimization?

A: In finite dimensions, yes: Random ‘Gradient-Free’ Optimization

To estimate $\nabla L(f)$, sample $v_1, \dots, v_M \sim \mathbb{P}_v$, and compute:

$$f, v_1, \dots, v_M \in \mathbb{R}^d$$

$$\hat{g} = \frac{1}{M} \sum_{m=1}^M \text{DL}(f; v_m) v_m$$

$$\bullet \quad \mathbb{E}[v] = 0 \text{ and } \text{Cov}[v] = I \quad \implies \quad \mathbb{E}[\hat{g}] = \nabla L(f).$$

$$(\text{e.g. } \mathbb{P}_v = \mathcal{N}(0, I))$$

Grad. Descent with Directional Derivatives

Q: Can we use just directional derivatives for optimization?

A: In finite dimensions, yes: Random ‘Gradient-Free’ Optimization

To estimate $\nabla L(f)$, sample $v_1, \dots, v_M \sim \mathbb{P}_v$, and compute:

$$f, v_1, \dots, v_M \in \mathbb{R}^d$$

$$\hat{g} = \frac{1}{M} \sum_{m=1}^M \mathrm{D}L(f; v_m) v_m$$

$$\bullet \quad \mathbb{E}[v] = 0 \text{ and } \mathrm{Cov}[v] = I \implies \mathbb{E}[\hat{g}] = \nabla L(f). \quad (\text{e.g. } \mathbb{P}_v = \mathcal{N}(0, I))$$

However, this breaks in infinite dimensions:

- Variance blows up as $d \rightarrow +\infty$
- There is no isotropic Gaussian $\mathcal{N}(0, I)$, nor analogous distributions
- Even if we change the distribution, $\mathbb{E}[\|v\|^2] < +\infty \implies \mathrm{Cov}[v] \neq I$.

Infinite Dim. Grad.-Free Descent

Ingredients:

- A prebasis for the Hilbert space: $\mathcal{P} = \{b_1, b_2, \dots\}$ such that
$$\mathcal{P} \text{ is linearly independent} \quad \text{and} \quad \text{span}(\mathcal{P}) \text{ is dense in } \mathcal{H}$$
- A distribution $\mathbb{P}_K$ over the positive naturals;

Infinite Dim. Grad.-Free Descent

Ingredients:

- A prebasis for the Hilbert space: $\mathcal{P} = \{b_1, b_2, \dots\}$ such that
 $\mathcal{P}$ is linearly independent and $\text{span}(\mathcal{P})$ is dense in $\mathcal{H}$
- A distribution $\mathbb{P}_K$ over the positive naturals;

The procedure: To estimate $\nabla L(f)$:

- Sample a dimension $K \sim \mathbb{P}_K$, and choose a batch size $M_K \in \mathbb{N}$ (e.g. $\lceil K/2 \rceil$)

Infinite Dim. Grad.-Free Descent

Ingredients:

- A prebasis for the Hilbert space: $\mathcal{P} = \{b_1, b_2, \dots\}$ such that
 $\mathcal{P}$ is linearly independent and $\text{span}(\mathcal{P})$ is dense in $\mathcal{H}$
- A distribution $\mathbb{P}_K$ over the positive naturals;

The procedure: To estimate $\nabla L(f)$:

- Sample a dimension $K \sim \mathbb{P}_K$, and choose a batch size $M_K \in \mathbb{N}$ (e.g. $\lceil K/2 \rceil$)
- Sample weights $\mathbf{w}^{(1)}, \dots, \mathbf{w}^{(M_K)} \sim \mathcal{N}(0, \Sigma_K)$ in $\mathbb{R}^K$, with a tailored Σ_K

Infinite Dim. Grad.-Free Descent

Ingredients:

- A prebasis for the Hilbert space: $\mathcal{P} = \{b_1, b_2, \dots\}$ such that
 $\mathcal{P}$ is linearly independent and $\text{span}(\mathcal{P})$ is dense in $\mathcal{H}$
- A distribution $\mathbb{P}_K$ over the positive naturals;

The procedure: To estimate $\nabla L(f)$:

- Sample a dimension $K \sim \mathbb{P}_K$, and choose a batch size $M_K \in \mathbb{N}$ (e.g. $\lceil K/2 \rceil$)
- Sample weights $\mathbf{w}^{(1)}, \dots, \mathbf{w}^{(M_K)} \sim \mathcal{N}(0, \Sigma_K)$ in $\mathbb{R}^K$, with a tailored Σ_K
- Compute

$$\hat{g} = \frac{\lambda_K}{M_K} \sum_{m=1}^{M_K} \text{DL}(f; v_m) v_m, \quad \text{where} \quad v_i := \sum_{j=1}^K w_j^{(i)} b_j$$

Infinite Dim. Grad.-Free Descent

Ingredients:

- A prebasis for the Hilbert space: $\mathcal{P} = \{b_1, b_2, \dots\}$ such that
 $\mathcal{P}$ is linearly independent and $\text{span}(\mathcal{P})$ is dense in $\mathcal{H}$
- A distribution $\mathbb{P}_K$ over the positive naturals;

The procedure: To estimate $\nabla L(f)$:

- Sample a dimension $K \sim \mathbb{P}_K$, and choose a batch size $M_K \in \mathbb{N}$ (e.g. $\lceil K/2 \rceil$)
- Sample weights $\mathbf{w}^{(1)}, \dots, \mathbf{w}^{(M_K)} \sim \mathcal{N}(0, \Sigma_K)$ in $\mathbb{R}^K$, with a tailored Σ_K
- Compute

$$\hat{g} = \frac{\lambda_K}{M_K} \sum_{m=1}^{M_K} \text{DL}(f; v_m) v_m, \quad \text{where} \quad v_i := \sum_{j=1}^K \mathbf{w}_j^{(i)} b_j$$

Infinite Dim. Grad.-Free Descent

Proposition. $\hat{g}$ is an unbiased estimator of a preconditioned gradient:

$$\mathbb{E}[\hat{g}] = C \nabla L(f),$$

where $C : \mathcal{H} \rightarrow \mathcal{H}$ is a positive-definite diagonal operator determined by $(\lambda_1, \lambda_2, \dots)$.

Infinite Dim. Grad.-Free Descent

Proposition. $\hat{g}$ is an unbiased estimator of a preconditioned gradient:

$$\mathbb{E}[\hat{g}] = C \nabla L(f),$$

where $C : \mathcal{H} \rightarrow \mathcal{H}$ is a positive-definite diagonal operator determined by $(\lambda_1, \lambda_2, \dots)$.

If we take $\lambda_k \equiv 1$, then we get $C = I$. But the **variance explodes**:

Proposition. If $\lambda_k \equiv 1$, then

$$\mathbb{E}[\|\hat{g}\|^2] \geq \sum_{i=1}^{\infty} \frac{\langle \nabla L(f), e_i \rangle^2}{\mathbb{P}[K \geq i]}.$$

Infinite Dim. Grad.-Free Descent

Proposition. $\hat{g}$ is an unbiased estimator of a preconditioned gradient:

$$\mathbb{E}[\hat{g}] = C \nabla L(f),$$

where $C : \mathcal{H} \rightarrow \mathcal{H}$ is a positive-definite diagonal operator determined by $(\lambda_1, \lambda_2, \dots)$.

So we take $\lambda_k = \mathbb{P}[K \geq k]$:

Proposition. Take $\lambda_k = \mathbb{P}[K \geq k]$ and $M_k = \lceil k/c \rceil$, for some $c > 0$. We then have

$$\mathbb{E}[\|\hat{g}\|_C^2] \leq 2(1 + 2c) \|\nabla L(f)\|^2,$$

where $\|\hat{g}\|_C := \|C^{-\frac{1}{2}} \hat{g}\|$.

Convergence

Theorem. If the loss is convex, K -smooth and the minimizer $f^\star$ is reasonably regular, then for a sufficiently small learning rate α ,

$$\mathbb{E}[L(\hat{f})] - L(f^\star) \leq \|f^\star - f_1\|_C^2 \cdot \left(\frac{1}{2T\alpha} + \alpha(1 + 2c)K^2 \right),$$

where $\hat{f}$ is the averaged iterate $\frac{1}{T}(f_1 + \dots + f_T)$.

Convergence

Theorem. If the loss is convex, K -smooth and the minimizer $f^\star$ is reasonably regular, then for a sufficiently small learning rate α ,

$$\mathbb{E}[L(\hat{f})] - L(f^\star) \leq \|f^\star - f_1\|_C^2 \cdot \left(\frac{1}{2T\alpha} + \alpha(1 + 2c)K^2 \right),$$

where $\hat{f}$ is the averaged iterate $\frac{1}{T}(f_1 + \dots + f_T)$.

Corollary. Choosing $\alpha \propto 1/\sqrt{T}$, we get

$$\mathbb{E}[L(\hat{f})] - L(f^\star) \leq O\left(\frac{1}{\sqrt{T}}\right).$$

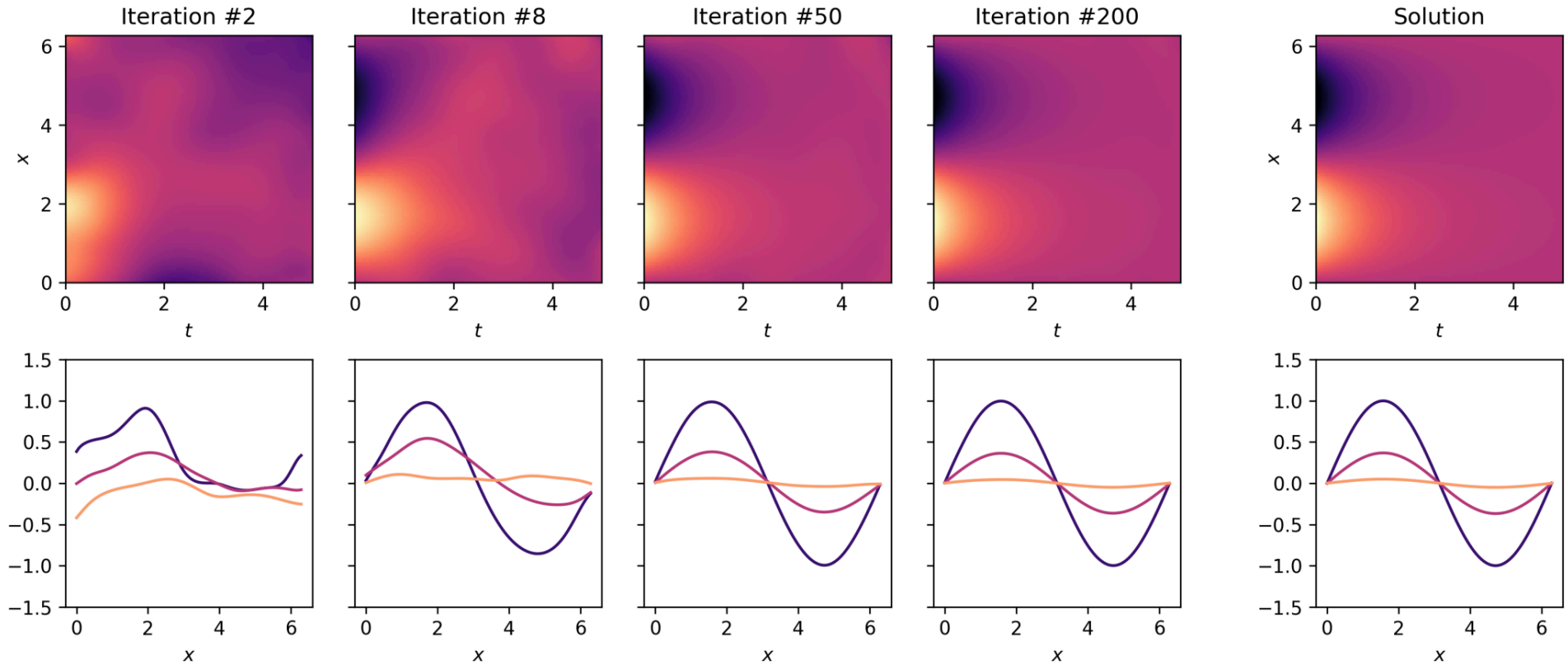
Example 1: Heat Equation

A simple heat equation:

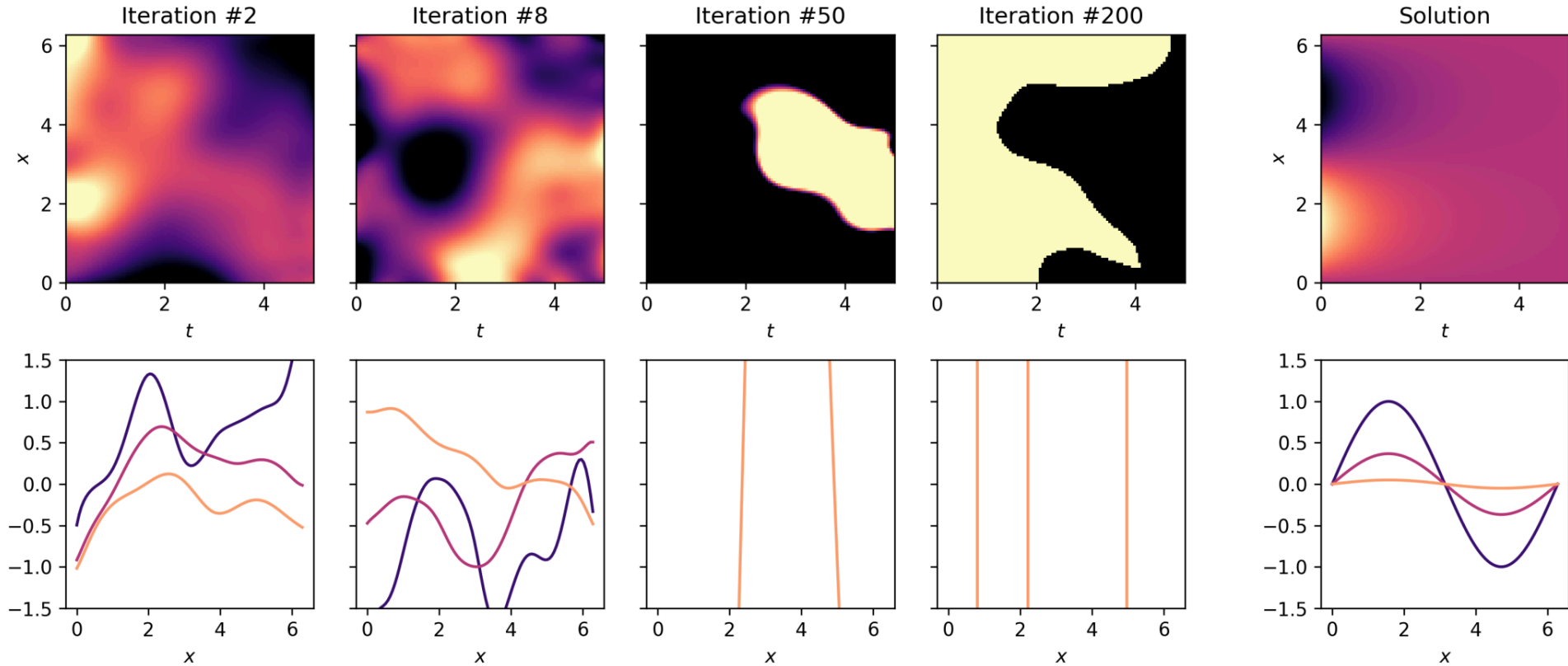
$$\begin{cases} \partial_t f(t, x) = \Delta f(t, x) & t \in (0, 5), x \in (0, 2\pi) \\ f(0, x) = \sin x & x \in (0, 2\pi) \\ f(t, 0) = f(t, 2\pi) = 0 & t \in (0, 5) \end{cases}$$

Unique solution given by $f(t, x) = e^{-t} \sin x$

Example 1: Heat Equation

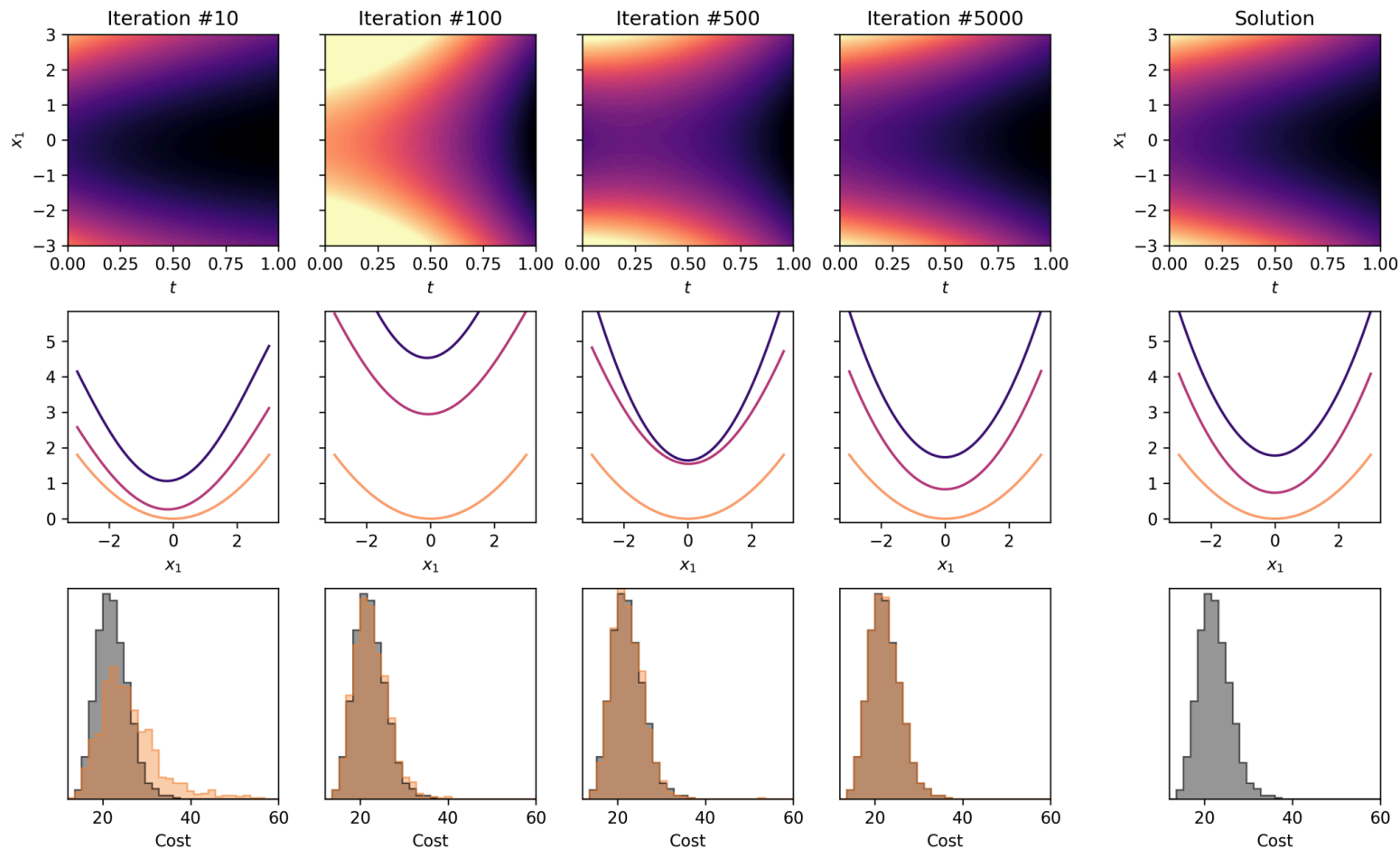


Example 1: Heat Equation



without preconditioning ($\lambda_k \equiv 1$)

Example 2: HJB Equation in 5+1 dims.



Conclusion

- Functional gradient descent is effective, but nontrivial to implement
 - To approximate: consider relative error conditions (and if possible consider adapting the representation over the course of optimization)
 - To compute the gradients: if the gradient is intractable, you can use random “gradient-free” estimators for the gradient with autodiff
- **Also in the papers:** more general statements, some impossibility results, gradients and function space properties, more experiments, *etc.*

Caio Peixoto, Daniel Csillag, Bernardo Costa and Yuri Saporito. *Random Gradient-Free Optimization in Infinite Dimensional Spaces*

Daniel Csillag, Rodrigo Schuller, Pedro Dall’Antonia, Leonidas Guibas, Luiz Velho, Tiago Novello. *Functional Gradient Descent with Adaptive Representations*

supplementary

Analysis of Banach Approx. Grad. Descent

For analysis, we assume smoothness:

Assumption (Smoothness in $\mathcal{B}$). There exists some $K > 0$ such that

$$L(f') \leq L(f) + \mathrm{D}L(f; f' - f) + \frac{K}{2} \|f' - f\|_{\mathcal{B}}^2, \quad \text{for all } f, f' \in \mathcal{B}.$$

We will also need two novel “gradient bridging” assumptions:

Assumption ($\mathcal{H}$ descends in $\mathcal{B}$). There exists some $\alpha > 0$ such that

$$\mathrm{D}L(f; \nabla L(f)) \geq \alpha \|\nabla L(f)\|_{\mathcal{B}}^2, \quad \text{for all } f \in \mathcal{B}.$$

Assumption (Gradient compatibility). There exists some $\beta > 0$ such that

$$\|\mathrm{D}L(f; \cdot)\|_{\mathcal{B}^*} \leq \beta \|\nabla L(f)\|_{\mathcal{B}}, \quad \text{for all } f \in \mathcal{B}.$$

Infinite Dim. Grad.-Free Descent

Key ingredients:

1. A prebasis for the Hilbert space: $\mathcal{P} = \{b_1, b_2, \dots\}$ such that
 $\mathcal{P}$ is linearly independent and $\text{span}(\mathcal{P})$ is dense in $\mathcal{H}$.
2. Tail probabilities $t_i := \mathbb{P}[K \geq i]$.

To sample $v|K$:

1. Compute the Gram matrix $[A_K]_{i,j} := \langle b_i, b_j \rangle$
2. Find its Cholesky decomposition $\mathbf{R}_K^\top \mathbf{R}_K = A_K$.
(Equivalent to finding the generalized QR decomposition $Q_K \mathbf{R}_K$ of $[b_1 \ \dots \ b_K]$.)
3. Sample weights $\mathbf{w} \sim \mathcal{N}(0, \mathbf{R}_K^{-1} \mathbf{T}_K^{-1} \mathbf{R}_K^{-\top})$, where $\mathbf{T}_K := \text{diag}(t_1, \dots, t_K)$, and take

$$v = \sum_{i=1}^K w_i b_i.$$