FOSI: Hybrid First and Second Order Optimization

HADAR SIVAN

MOSHE GABEL

ASSAF SCHUSTER

TECHNION Israel Institute of Technology

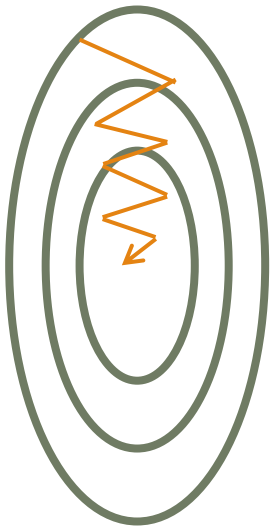
YORK U

TECHNION Israel Institute of Technology

Background

- $\min_{\Theta} f(\Theta)$
- Iterative optimizer:

$$\Theta = \Theta - \alpha \nabla f(\Theta)$$

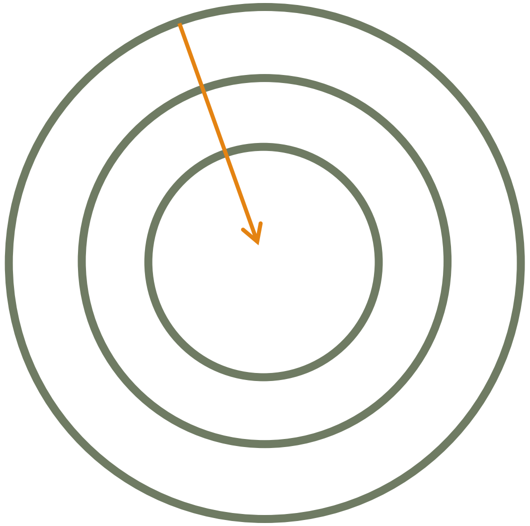


Background

- $\min_{\Theta} f(\Theta)$
- Iterative optimizer:

preconditioner

$$\Theta = \Theta - \alpha \, \boldsymbol{P}^{-1} \nabla f(\Theta)$$



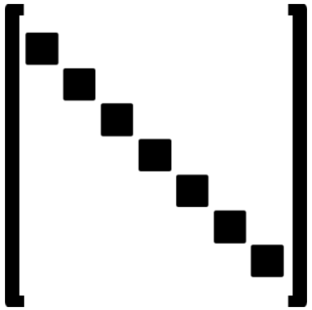
Background

- $\min_{\Theta} f(\Theta)$
- Iterative optimizer:

$$\Theta = \Theta - \alpha\, P^{-1}\, \nabla f(\Theta)$$

- Adam
- AdamW
- RMSProp
- …

$$P^{-1} \sim \mathbf{diag}(H)^{-1}$$



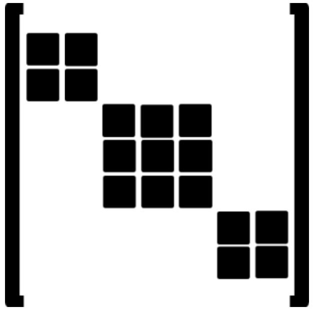
Background

- $\min_{\Theta} f(\Theta)$
- Iterative optimizer:

$$\Theta = \Theta - \alpha\, P^{-1}\, \nabla f(\Theta)$$

  - K-FAC
  - Shampoo
  - K-BFGS
  - …

$$P^{-1} \sim \mathbf{block\_diag}(H)^{-1}$$



FOSI

Improves performance of first-order optimizers by using two key ideas:

1. Split the problem to two parts with different optimizers
2. Estimate P^{-1} directly, avoid inversion of P

✓ Drop-in replacement, no extra tuning, faster wall-time

FOSI: Overview

$f(\theta)$ | θ_0 | Base optimizer

FOSI – repeat until convergence:

1. Gradient
2. Second-order info
3. Update θ_t

Notations

At iteration t:

$$H = \nabla^2 f(\theta_t)$$

1. Gradient
2. Second-order info
3. Update θ_t

eigenvalues

$$\left[\lambda_1 > \dots > \lambda_k > \lambda_{k+1} > \dots > \lambda_{n-\ell} > \lambda_{n-\ell+1} > \dots > \lambda_n\right]$$

eigenvectors

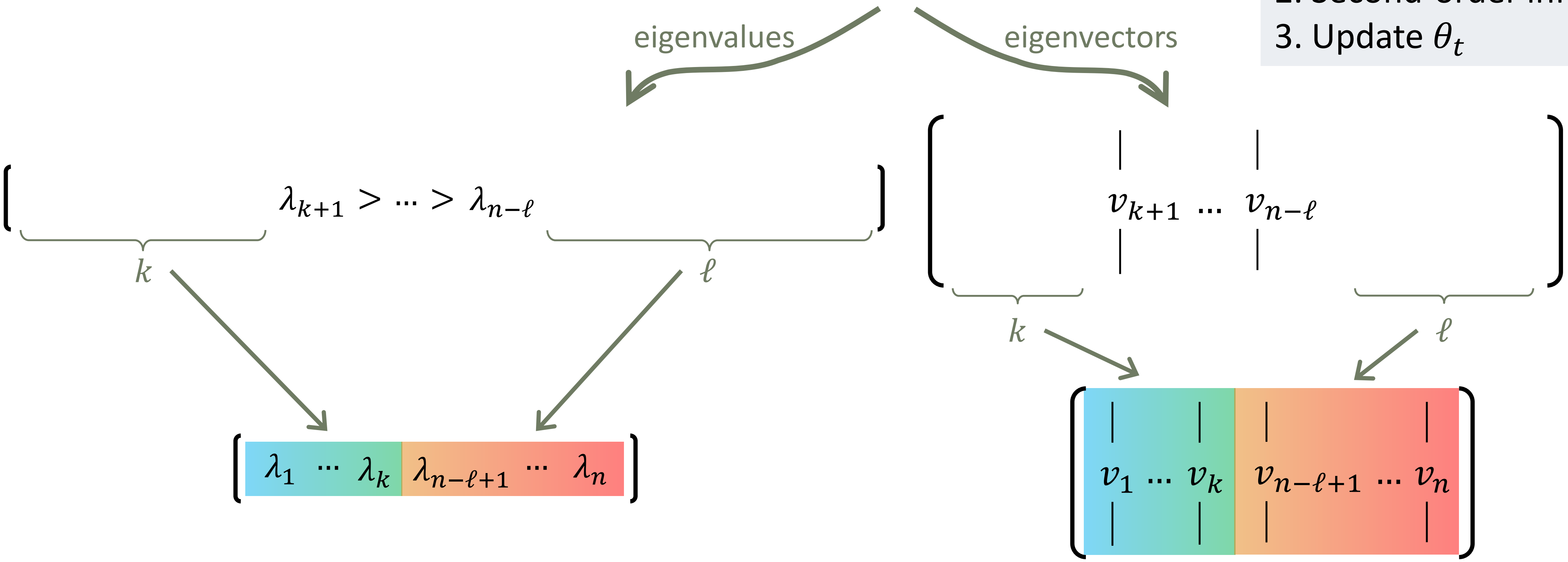
$$\begin{bmatrix} | & & | & | & & | & | & & | \\ v_1 & \dots & v_k & v_{k+1} & \dots & v_{n-\ell} & v_{n-\ell+1} & \dots & v_n \\ | & & | & | & & | & | & & | \end{bmatrix}$$

Notations

At iteration t :

$$H = \nabla^2 f(\theta_t)$$

1. Gradient
2. Second-order info
3. Update θ_t



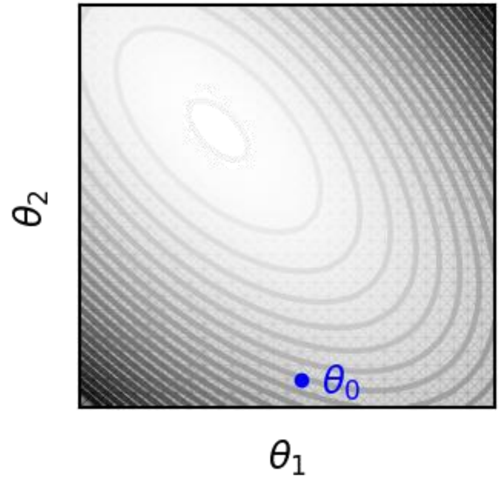
Update Step

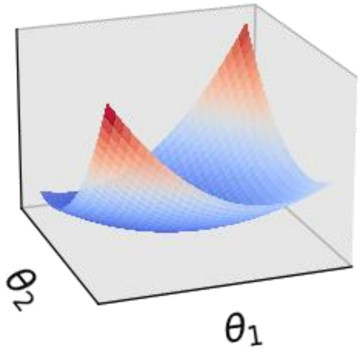
1. Gradient
2. Second-order info
3. Update θ_t

a. Split f to f_1 and f_2
b. Newton's step on f_1
c. Base opt step on f_2

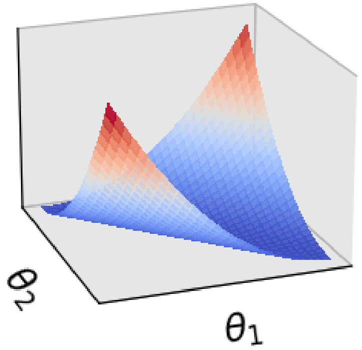
Splitting f $(k = 1, \ell = 0)$

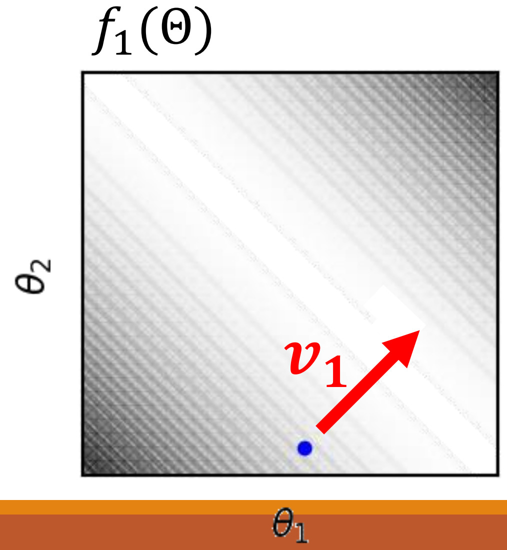
$$f(\theta) = \frac{1}{2}\theta^T \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{-1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{pmatrix} \begin{pmatrix} 4 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{-1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{pmatrix}^T \theta$$

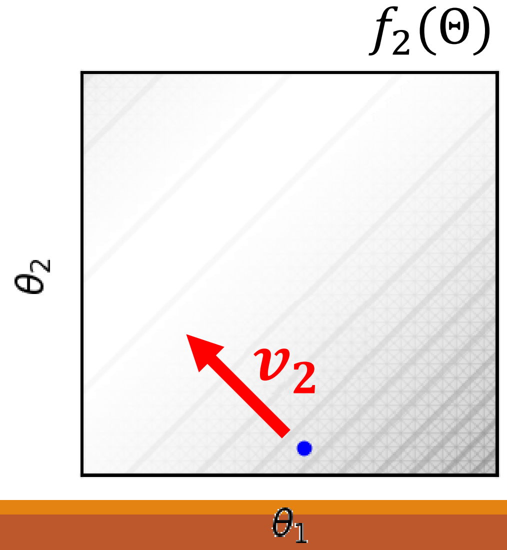


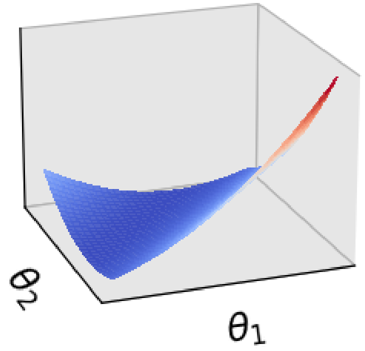


1. Gradient
2. Second-order info
3. Update θ_t
   a. Split f to f_1 and f_2
   b. Newton's step on f_1
   c. Base opt step on f_2



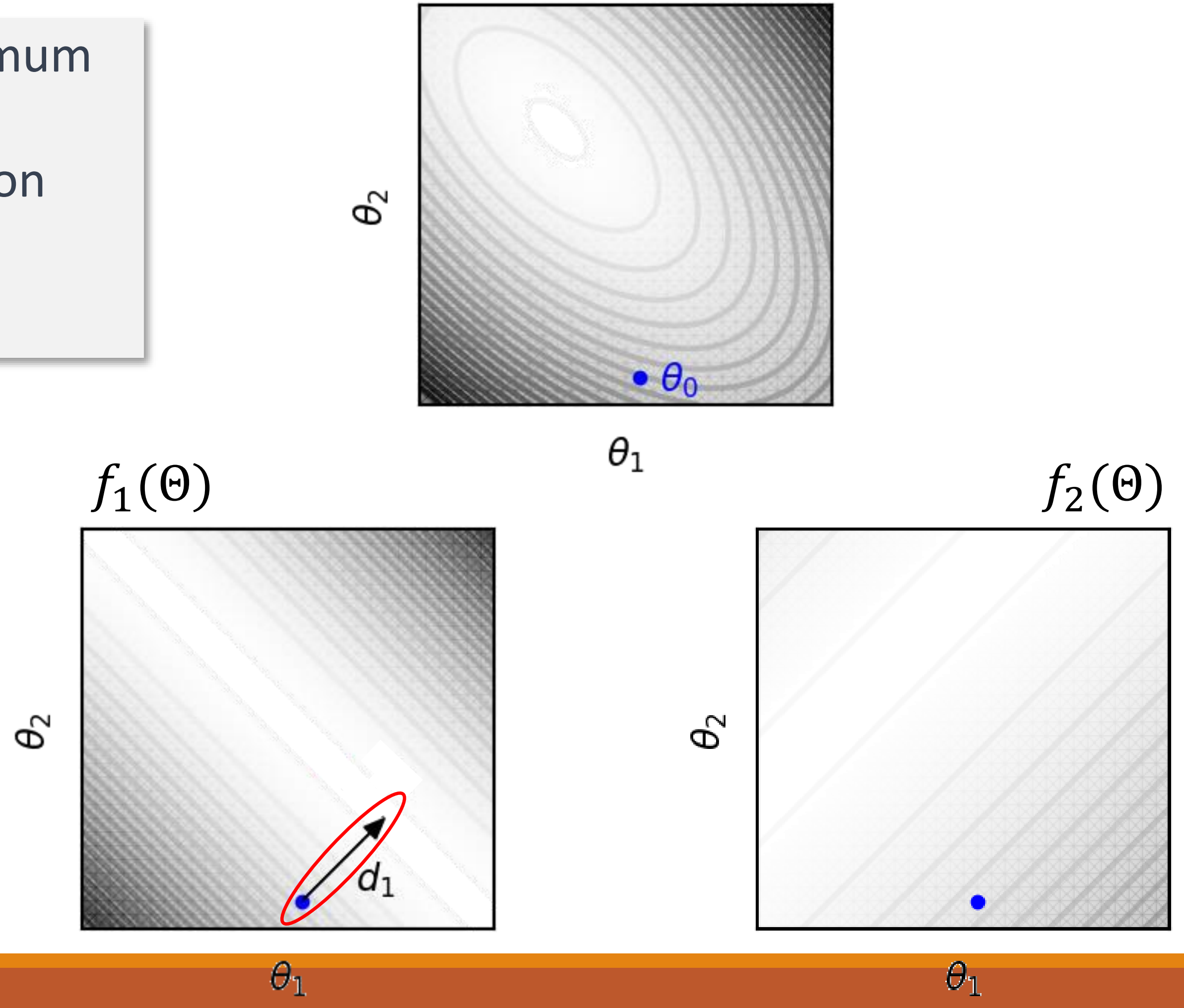






Minimize f_1

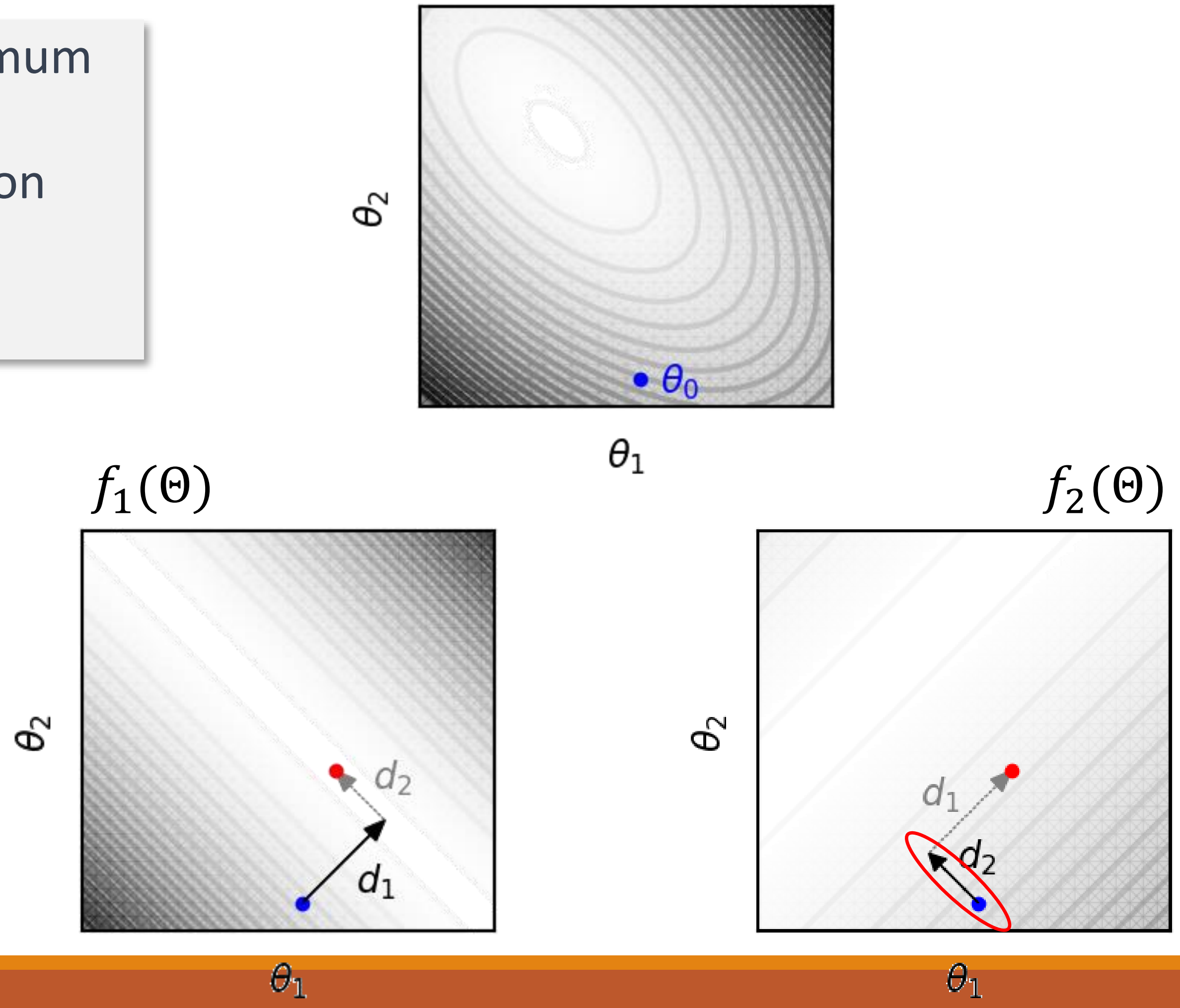
- ✓ d_1 brings f_1 to minimum in a single step
- ✓ Trivial matrix inversion



1. Gradient
2. Second-order info
3. Update θ_t
   a. Split f to f_1 and f_2
   b. Newton's step on f_1
   c. Base opt step on f_2

Minimize f_2

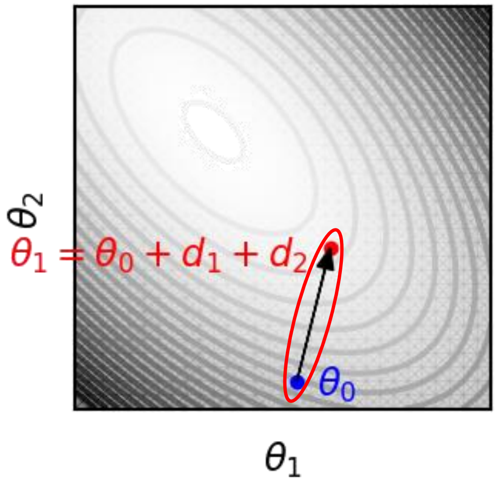
- ✓ d_1 brings f_1 to minimum in a single step
- ✓ Trivial matrix inversion
- ✓ d_1 orthogonal to d_2
- ✓ No mutual impact



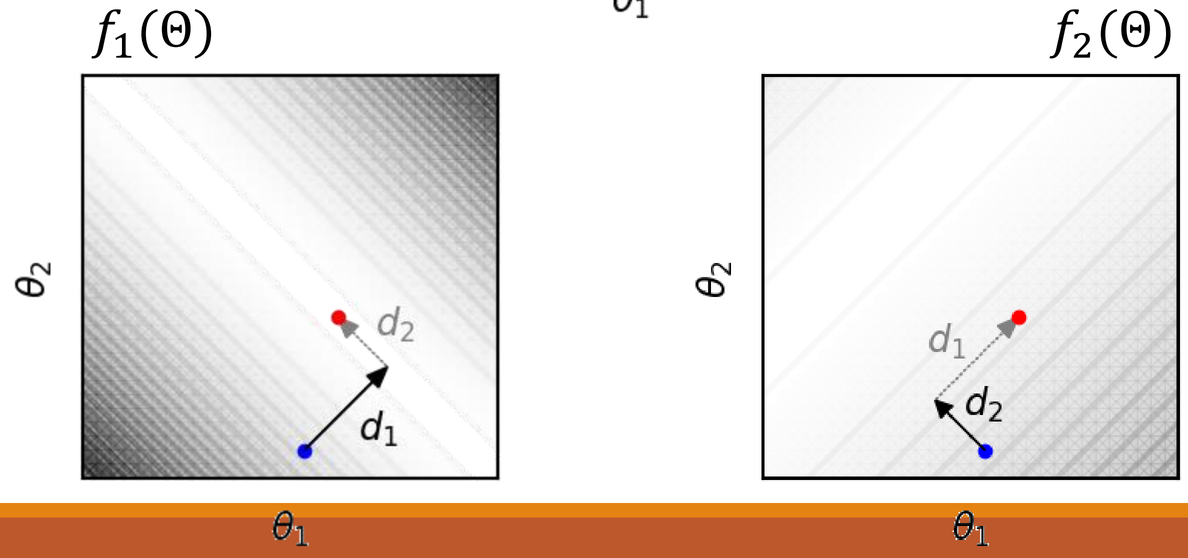
1. Gradient
2. Second-order info
3. Update θ_t
   a. Split f to f_1 and f_2
   b. Newton's step on f_1
   c. Base opt step on f_2

FOSI’s Update Step

- ✓ d_1 brings f_1 to minimum in a single step
- ✓ Trivial matrix inversion
- ✓ d_1 orthogonal to d_2
- ✓ No mutual impact

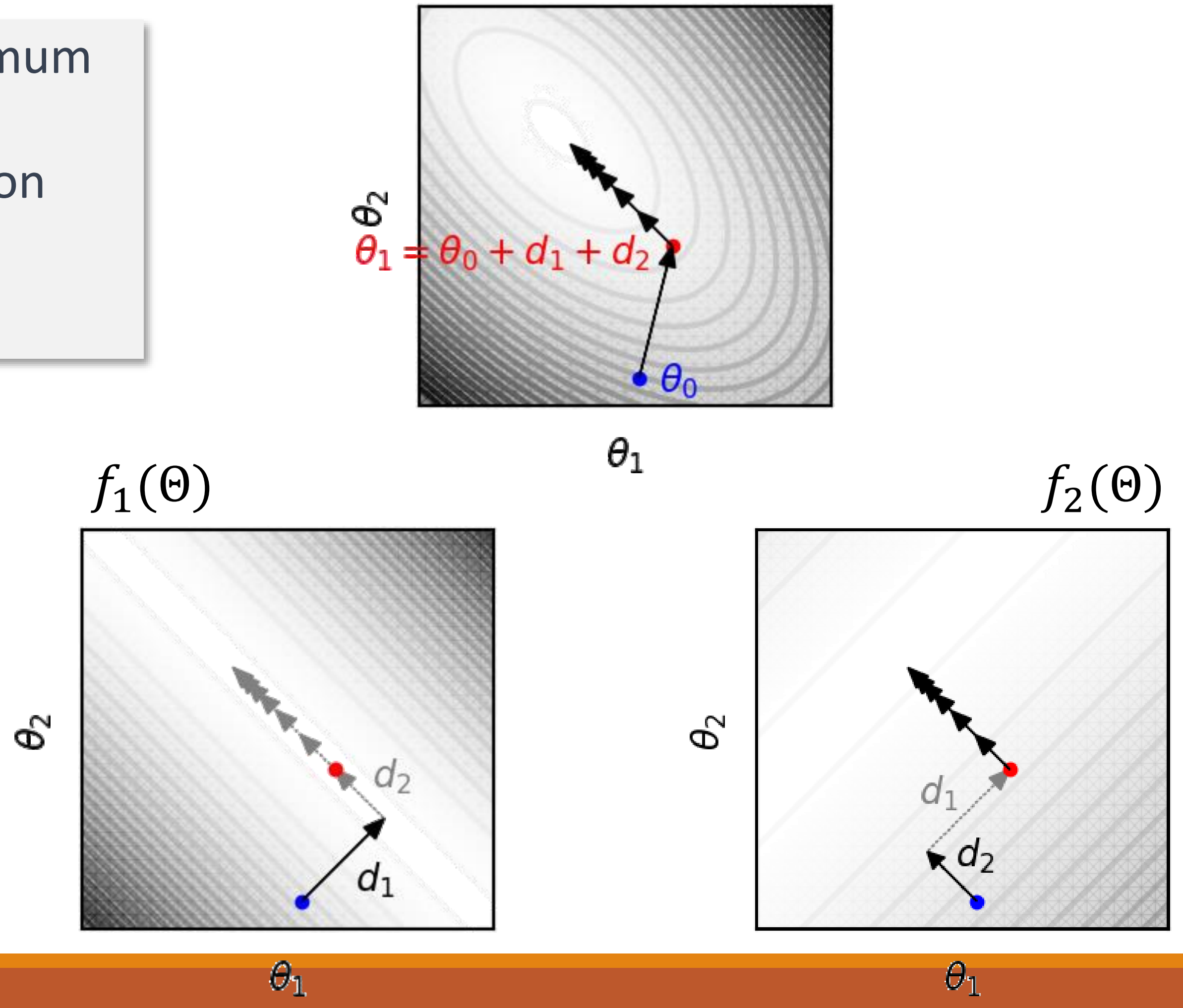


1. Gradient
2. Second-order info
3. Update θ_t
   a. Split f to f_1 and f_2
   b. Newton’s step on f_1
   c. Base opt step on f_2



FOSI's Update Step

- ✓ d_1 brings f_1 to minimum in a single step
- ✓ Trivial matrix inversion
- ✓ d_1 orthogonal to d_2
- ✓ No mutual impact



1. Gradient
2. Second-order info
3. Update θ_t
   a. Split f to f_1 and f_2
   b. Newton's step on f_1
   c. Base opt step on f_2

FOSI Algorithm

Inputs: $f(\theta)$, θ_0, Base optimizer

Repeat until convergence:

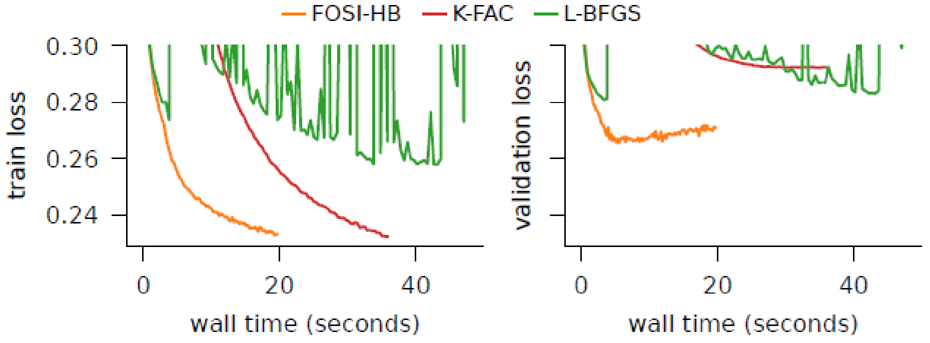
1. Gradient
2. Second-order info
3. Update θ_t
   a. Split f to f_1 and f_2
   b. Newton's step on f_1
   c. Base opt step on f_2

Experiments and Results

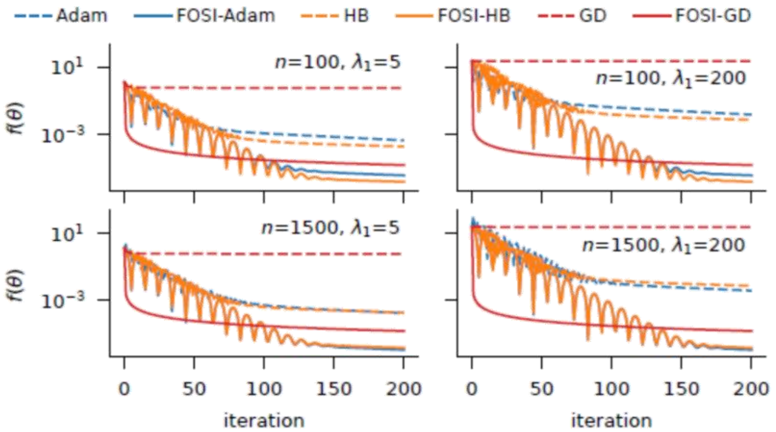
Comparison to base first-order optimizers

Task	HB	FOSI-HB		Adam	FOSI-Adam	
AC	3822	**1850**	(40.4%)	5042	**3911**	(28.9%)
LM	269	**207**	(1.71)	270	**219**	(1.76)
AE	354	**267**	(52.46)	375	**313**	(51.26)
TL	93	**53**	(79.1%)	68	**33**	(79.0%)
LR	16	**8**	(92.8%)	**12**	18	(92.8%)

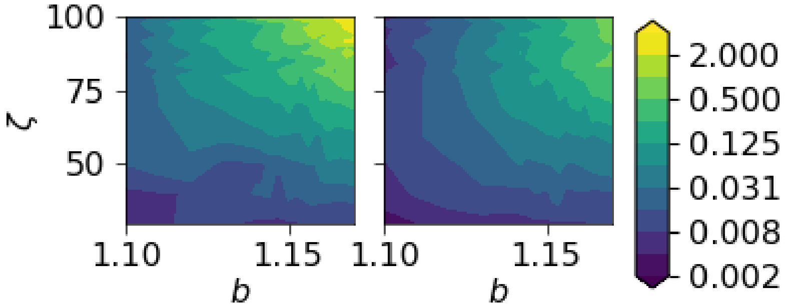
Comparison to second-order methods



Dimensionality and ill-conditioning



Ill-conditioning and diagonally dominance



Summary

- FOSI: hybrid-meta optimizer
  - ✓ **Meta-algorithm**: Applicable to any first order optimizer.
  - ✓ **Mathematical guarantees**: Improves effective condition number.
- Achieves the same loss as the base optimizer in 75% of the wall time.
- Open source: https://github.com/hsivan/fosi
  - ✓ Compatible with **JAX** (Optax) and **PyTorch** (TorchOpt)