A decisional step for Variational Monte Carlo

Optimizing Neural Quantum States with Decision Geometry

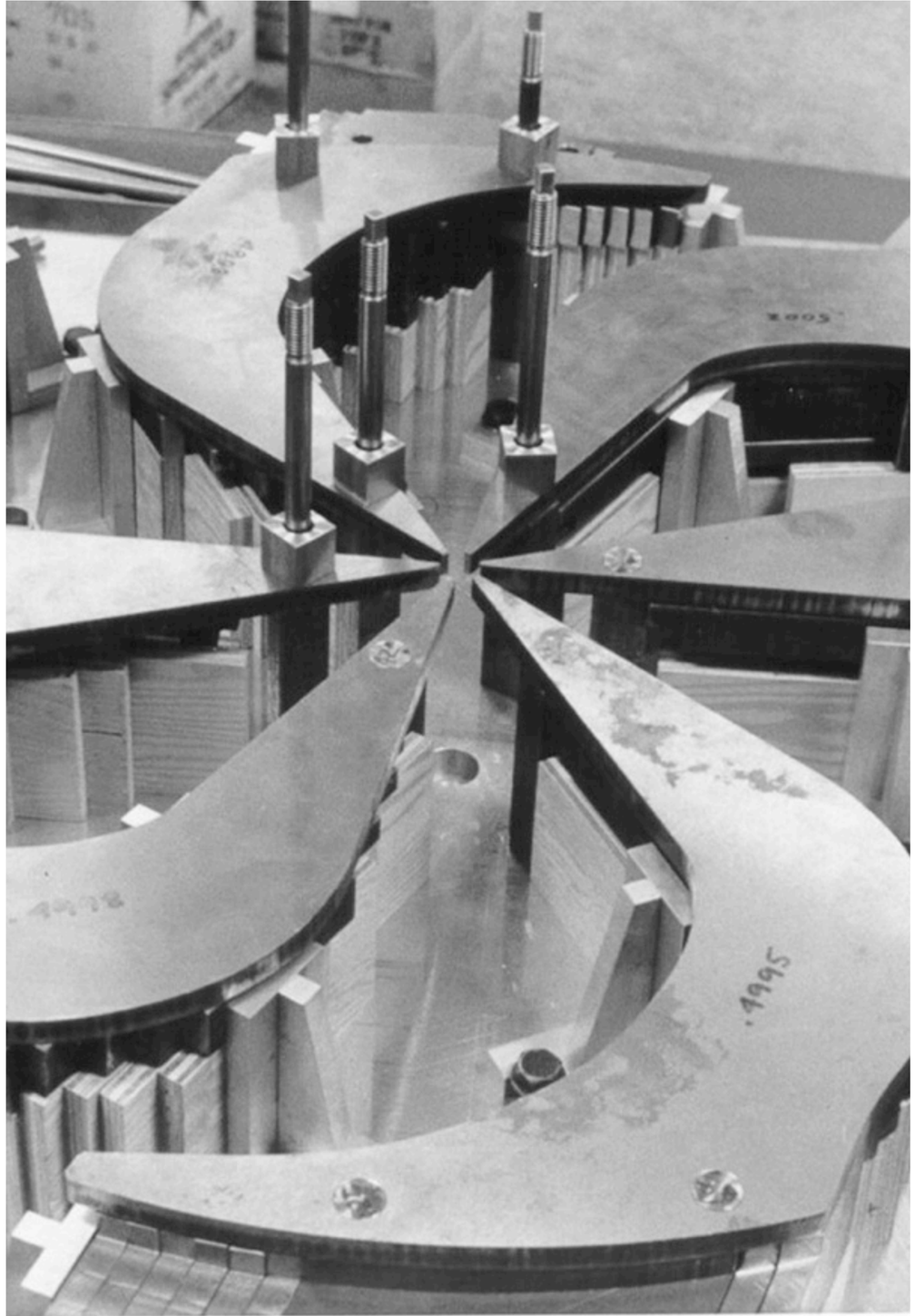
Collaborators: Arnau Rios, James Keeble, Javier Rozalén Sarmiento

Publications: Physical Review A, **108** 063320 (2023)
Arxiv: 2401.17550 [nucl-th]

Mehdi Drissi
TRIUMF - Theory department

Progress in Ab Initio for Nuclear Theory

TRIUMF - Vancouver
1st of March 2024



Discovery, accelerated

Outline

- **Variational Monte Carlo with Neural Quantum States**
  - Overview of VMC with NQS
  - The Kronecker-Factored Approximate Curvature (KFAC)
- **Augmented KFAC for VMC problems**
  - Scaling improvement from a Quasi-Newton approach
  - Direction improvement from MINRES
- **Decision geometry for VMC**
  - Game theory reformulation of VMC
  - Testing decisional gradient descent

Variational Monte-Carlo in a nutshell

General Many-body problem

- Many-body system of interacting particles
  - Input Hamiltonian: H
- Here focus on:
  - Many-body system of A fermions
  - Canonical ensemble at $T = 0$
- Goal:
  - Finding $\{E_{gs}\ , |\Psi_{gs}\rangle\ \}$ s.t. $H\,|\Psi_{gs}\rangle = E_{gs}\,|\Psi_{gs}\rangle$

Variational approach

- Rayleigh-Ritz variational principle

$$\forall\,|\Psi\rangle \in \mathscr{H}_{\mathrm{A}},\ \frac{\langle\Psi|H|\Psi\rangle}{\langle\Psi|\Psi\rangle} \geq \frac{\langle\Psi_{gs}|H|\Psi_{gs}\rangle}{\langle\Psi_{gs}|\Psi_{gs}\rangle}$$

Variational reformulation

$$\boxed{\begin{aligned} E_{gs} &= \min_{|\Psi\rangle} \frac{\langle\Psi|H|\Psi\rangle}{\langle\Psi|\Psi\rangle} \\ \Psi_{gs}\rangle &= \mathrm{argmin}_{|\Psi\rangle} \frac{\langle\Psi|H|\Psi\rangle}{\langle\Psi|\Psi\rangle} \end{aligned}}$$

Variational Monte-Carlo in a nutshell

General Many-body problem

- Many-body system of interacting particles
  - Input Hamiltonian: H
- Here focus on:
  - Many-body system of A fermions
  - Canonical ensemble at $T = 0$
- Goal:
  - Finding $\{E_{gs}\ , |\Psi_{gs}\rangle\ \}$ s.t. $H\,|\Psi_{gs}\rangle = E_{gs}\,|\Psi_{gs}\rangle$

Technical challenges and solutions of VMC

	Tradeoffs
Infinite dimensional variational space	
Ansatz based wave-functions →	Biased estimation
High-dimension integrals [Metropolis et al. (1953)]	
Markov Chain Monte-Carlo sampling →	Statistical noise
Non-linear global optimization problem	
Iterative linear/quadratic local optimization →	Local minima

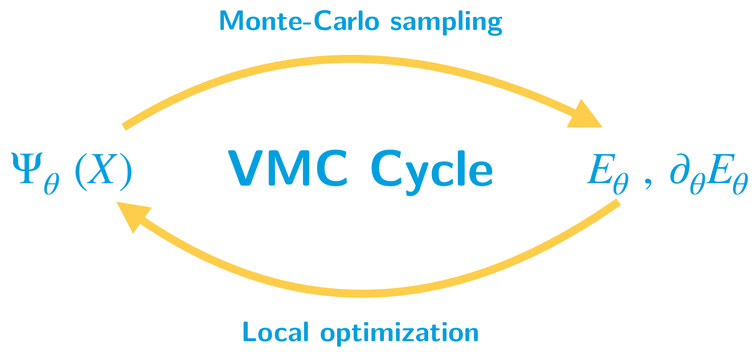
Variational approach

- Rayleigh-Ritz variational principle

$$\forall\,|\Psi\rangle \in \mathscr{H}_{\mathrm{A}},\ \frac{\langle\Psi|H|\Psi\rangle}{\langle\Psi|\Psi\rangle} \geq \frac{\langle\Psi_{gs}|H|\Psi_{gs}\rangle}{\langle\Psi_{gs}|\Psi_{gs}\rangle}$$

Variational reformulation

$$\boxed{\begin{aligned} E_{gs} &= \min_{|\Psi\rangle} \frac{\langle\Psi|H|\Psi\rangle}{\langle\Psi|\Psi\rangle} \\ \Psi_{gs}\rangle &= \mathrm{argmin}_{|\Psi\rangle} \frac{\langle\Psi|H|\Psi\rangle}{\langle\Psi|\Psi\rangle} \end{aligned}}$$



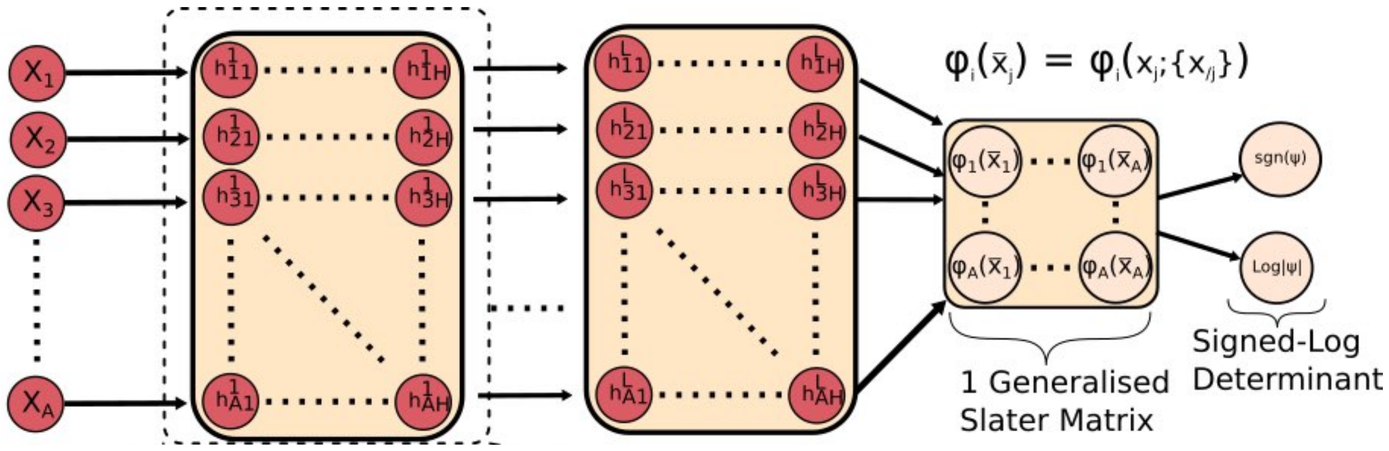
A simple yet insightful many-body problem

Many-body system

- Hamiltonian in 1D
  - $H = -\sum_i \frac{1}{2}\partial_{x_i}^2 + \sum_i \frac{1}{2}x_i^2 + \sum_{i<j} \frac{V_0}{\sqrt{2\pi}\sigma_0} \exp\left(-\frac{(x_i - x_j)^2}{2\sigma_0^2}\right)$

    Harmonic trap

    Gaussian interaction
- Constraints:
  - Fixed particle number A
  - Fixed temperature $T = 0$



A simple yet insightful many-body problem

Many-body system

- Hamiltonian in 1D
  - $$H = -\sum_i \frac{1}{2}\partial_{x_i}^2 + \sum_i \frac{1}{2}x_i^2 + \sum_{i<j} \frac{V_0}{\sqrt{2\pi}\sigma_0} \exp\left(-\frac{(x_i - x_j)^2}{2\sigma_0^2}\right)$$

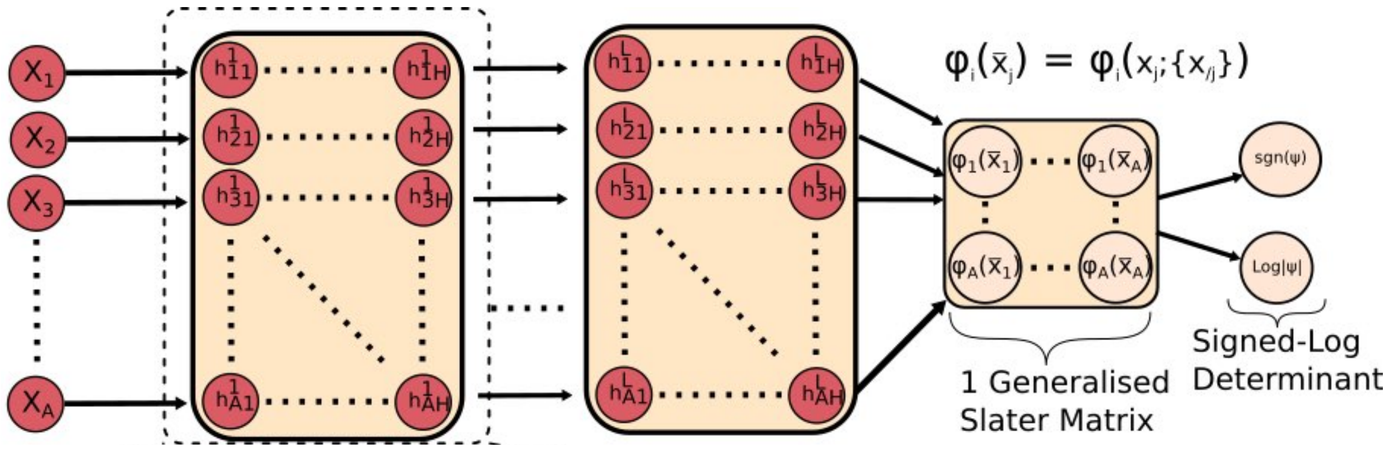
Harmonic trap

Gaussian interaction

- Constraints:
  - Fixed particle number A
  - Fixed temperature $T = 0$

NQS architecture

- Default architectural hyperparameters
  - Number of layers: $L = 2$
  - Width of each layer: $H = 64$
  - Number of determinants: $D = 1$
  - Total number of parameters $\sim 10\ 000$



A simple yet insightful many-body problem

Many-body system

- Hamiltonian in 1D
  - $H = -\sum_i \frac{1}{2}\partial^2_{x_i} + \sum_i \frac{1}{2}x_i^2 + \sum_{i<j} \frac{V_0}{\sqrt{2\pi}\sigma_0} \exp\left(-\frac{(x_i - x_j)^2}{2\sigma_0^2}\right)$

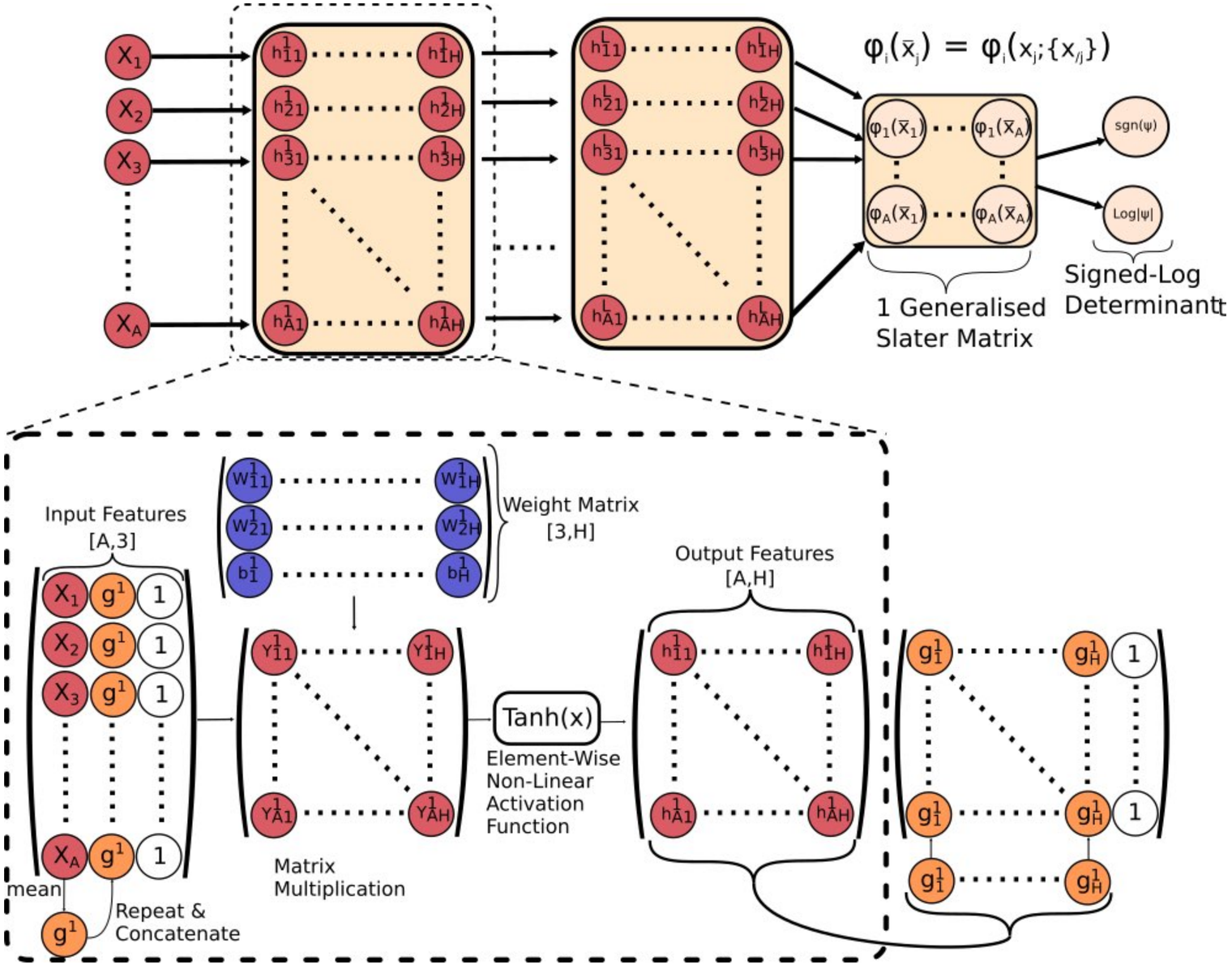
Harmonic trap

Gaussian interaction

- Constraints:
  - Fixed particle number A
  - Fixed temperature $T = 0$

NQS architecture

- Default architectural hyperparameters
  - Number of layers: $L = 2$
  - Width of each layer: $H = 64$
  - Number of determinants: $D = 1$
  - Total number of parameters $\sim 10\,000$
- Permutation equivariant layers
  - Permutation of input rows
    Permutation of output rows
  - Propagates all the way to the orbitals
  - Final layer with determinant:
    equivariance $\Rightarrow$ antisymmetry



Outline

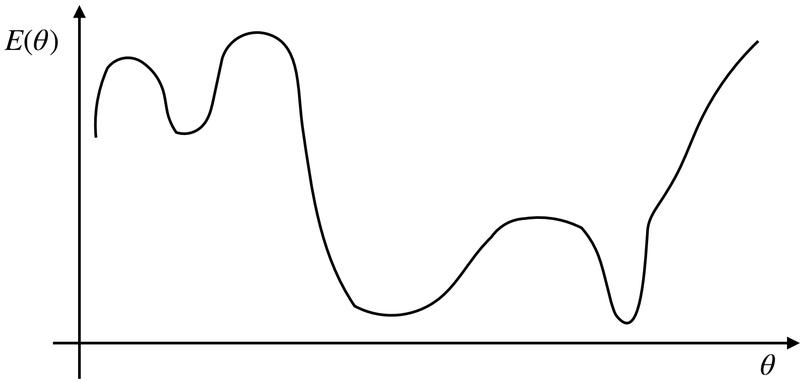
- **Variational Monte Carlo with Neural Quantum States**
  - Overview of VMC with NQS
  - The Kronecker-Factored Approximate Curvature (KFAC)
- **Augmented KFAC for VMC problems**
  - Scaling improvement from a Quasi-Newton approach
  - Direction improvement from MINRES
- **Decision geometry for VMC**
  - Game theory reformulation of VMC
  - Testing decisional gradient descent

General strategy for non-linear optimizers

Definition of the problem

- Let $E(\theta)$ be our cost function
- <u>Goal</u>
  - $E^* = \min_{\theta \in \mathbb{R}^D} E(\theta)$
  - $\theta^* = \text{argmin}_{\theta \in \mathbb{R}^D} E(\theta)$
- <u>Problem</u>
  - $D > 10\ 000$
  - $E(\theta)$ highly non-linear

1D example

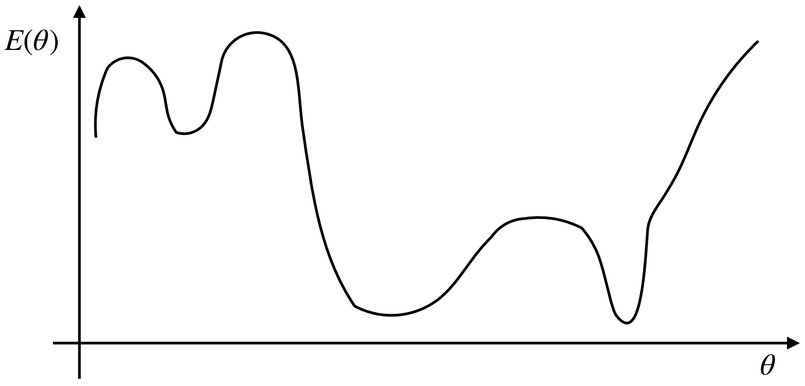


General strategy for non-linear optimizers

Definition of the problem

- Let $E(\theta)$ be our cost function
- Goal
  - $E^* = \min_{\theta \in \mathbb{R}^D} E(\theta)$
  - $\theta^* = \text{argmin}_{\theta \in \mathbb{R}^D} E(\theta)$
- Problem
  - $D > 10\ 000$
  - $E(\theta)$ highly non-linear

1D example



General strategy

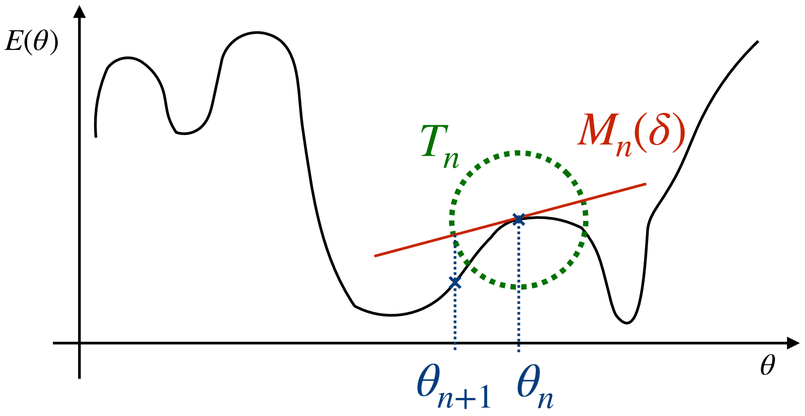
- Complicated problem
  Many trivial problems
- Sequence of linear/quadratic optimizations

General strategy for non-linear optimizers

Definition of the problem

- Let $E(\theta)$ be our cost function
- Goal
  - $E^* = \min_{\theta \in \mathbb{R}^D} E(\theta)$
  - $\theta^* = \text{argmin}_{\theta \in \mathbb{R}^D} E(\theta)$
- Problem
  - $D > 10\ 000$
  - $E(\theta)$ highly non-linear

1D example



General strategy

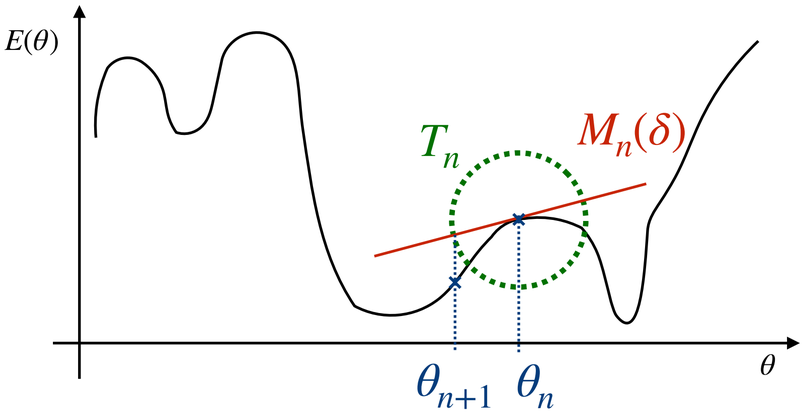
- Complicated problem
  Many trivial problems
- Sequence of linear/quadratic optimizations
- Iterative algorithm
  - $\theta_{n+1} = \theta_n + \text{argmin}_{\delta \in T_n} M_n(\delta)$
  - $E_{n+1} = E(\theta_{n+1})$
  - where, $M_n(\delta) = \frac{1}{2}\delta^T Q \delta + L^T \delta + C$
    and T_n = region where $M_n(\delta)$ is trusted
  - Update $M_n(\delta)$ and T_n
- In practice: T_n is replaced by a **regulator**
  - $M_n(\delta) \leftarrow M_n(\delta) + \frac{1}{2}\lambda_n\, \delta^T R_n \delta$, with $R_n \succeq 0$
  - Tikhonov regularization

General strategy for non-linear optimizers

Definition of the problem

- Let $E(\theta)$ be our cost function
- Goal
  - $E^* = \min_{\theta \in \mathbb{R}^D} E(\theta)$
  - $\theta^* = \text{argmin}_{\theta \in \mathbb{R}^D} E(\theta)$
- Problem
  - $D > 10\,000$
  - $E(\theta)$ highly non-linear

1D example



General strategy

- Complicated problem
  Many trivial problems
- Sequence of linear/quadratic optimizations
- Iterative algorithm
  - $\theta_{n+1} = \theta_n + \text{argmin}_{\delta \in T_n} M_n(\delta)$
  - $E_{n+1} = E(\theta_{n+1})$
  - where, $M_n(\delta) = \frac{1}{2}\delta^T Q \delta + L^T \delta + C$
    and T_n = region where $M_n(\delta)$ is trusted
  - Update $M_n(\delta)$ and T_n
- In practice: T_n is replaced by a **regulator**
  - $M_n(\delta) \leftarrow M_n(\delta) + \frac{1}{2}\lambda_n \, \delta^T R_n \delta$, with $R_n \succeq 0$
  - Tikhonov regularization

Optimizers discussed here

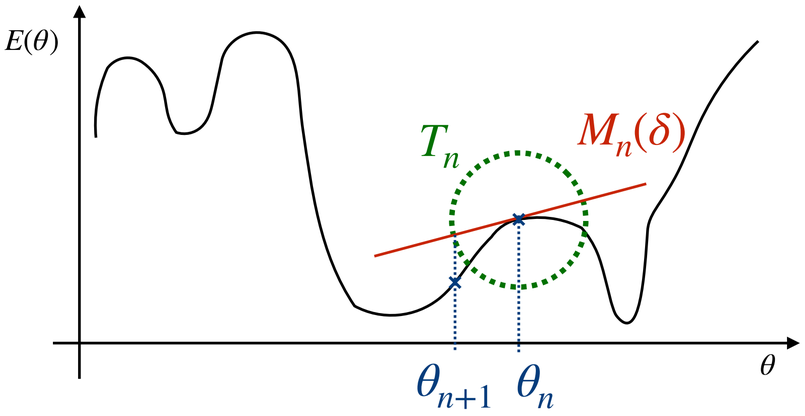
- Gradient descent ($\sim$ Adam)
  - $M_n(\delta) \equiv \nabla E(\theta_n)^T \delta + E(\theta_n)$
  - $T_n \equiv \{\delta : \; ||\delta||_2 \leq \alpha ||\nabla E(\theta_n)||_2\}$
  - $\alpha \equiv$ **learning rate**
  - ➡ $\delta_n = -\alpha \, \nabla E(\theta_n)$

General strategy for non-linear optimizers

Definition of the problem

- Let $E(\theta)$ be our cost function
- Goal
  - $E^* = \min_{\theta \in \mathbb{R}^D} E(\theta)$
  - $\theta^* = \text{argmin}_{\theta \in \mathbb{R}^D} E(\theta)$
- Problem
  - $D > 10\,000$
  - $E(\theta)$ highly non-linear

1D example



General strategy

- Complicated problem
  Many trivial problems
- Sequence of linear/quadratic optimizations
- Iterative algorithm
  - $\theta_{n+1} = \theta_n + \text{argmin}_{\delta \in T_n} M_n(\delta)$
  - $E_{n+1} = E(\theta_{n+1})$
  - where, $M_n(\delta) = \frac{1}{2}\delta^T Q \delta + L^T \delta + C$
    and T_n = region where $M_n(\delta)$ is trusted
  - Update $M_n(\delta)$ and T_n
- In practice: T_n is replaced by a **regulator**
  - $M_n(\delta) \leftarrow M_n(\delta) + \frac{1}{2}\lambda_n \, \delta^T R_n \delta$, with $R_n \succeq 0$
  - Tikhonov regularization

Optimizers discussed here

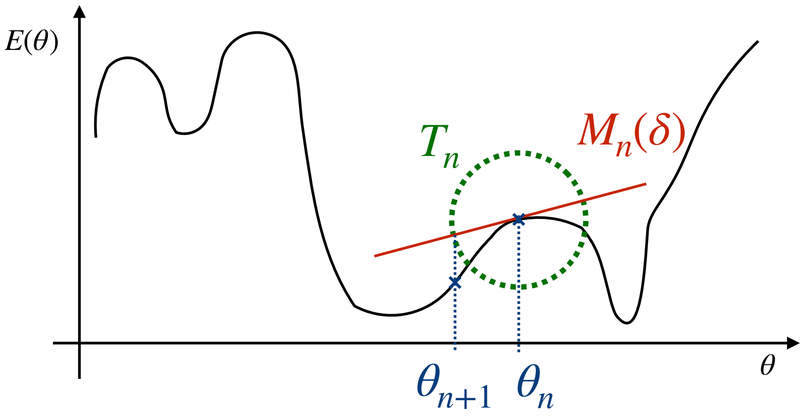
- Gradient descent ($\sim$ Adam)
  - $M_n(\delta) \equiv \nabla E(\theta_n)^T \delta + E(\theta_n)$
  - $T_n \equiv \{\delta : \; ||\delta||_2 \leq \alpha ||\nabla E(\theta_n)||_2\}$
  - $\alpha \equiv$ **learning rate**
  - ➡ $\delta_n = -\alpha \, \nabla E(\theta_n)$
- Natural gradient descent
  - Fisher information metric
    $F_{ij}(p) \equiv \mathbb{E}_{X \sim p}\left[\partial_{\theta_i} \ln p(X) \; \partial_{\theta_j} \ln p(X)\right]$
  - $T_n(r) = \{\delta : \; \delta^T F(\theta_n) \delta \leq r^2\}$
  - $\Leftrightarrow M_n(\delta) = \frac{1}{2}\delta^T F \delta + \nabla E(\theta_n)^T \delta + E(\theta_n)$
  - ➡ $\delta_n = -F^{-1}(\theta_n) \, \nabla E(\theta_n)$

General strategy for non-linear optimizers

Definition of the problem

- Let $E(\theta)$ be our cost function
- Goal
  - $E^* = \min_{\theta \in \mathbb{R}^D} E(\theta)$
  - $\theta^* = \text{argmin}_{\theta \in \mathbb{R}^D} E(\theta)$
- Problem
  - $D > 10\ 000$
  - $E(\theta)$ highly non-linear

1D example



General strategy

- Complicated problem
  Many trivial problems
- Sequence of linear/quadratic optimizations
- Iterative algorithm
  - $\theta_{n+1} = \theta_n + \text{argmin}_{\delta \in T_n} M_n(\delta)$
  - $E_{n+1} = E(\theta_{n+1})$
  - where, $M_n(\delta) = \frac{1}{2}\delta^T Q \delta + L^T \delta + C$
    and T_n = region where $M_n(\delta)$ is trusted
  - Update $M_n(\delta)$ and T_n
- In practice: T_n is replaced by a **regulator**
  - $M_n(\delta) \leftarrow M_n(\delta) + \frac{1}{2}\lambda_n\, \delta^T R_n \delta$, with $R_n \succeq 0$
  - Tikhonov regularization

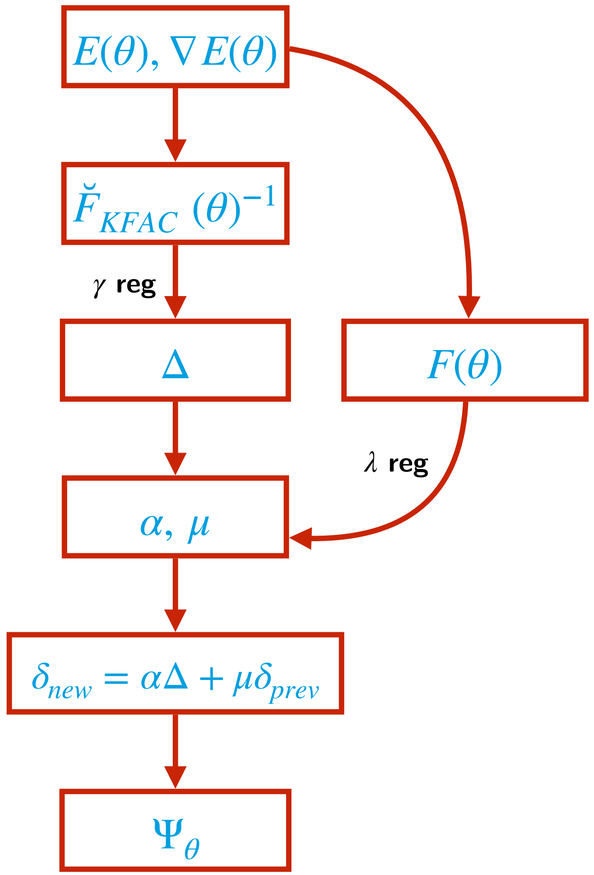
Optimizers discussed here

- Gradient descent ($\sim$ Adam)
  - $M_n(\delta) \equiv \nabla E(\theta_n)^T \delta + E(\theta_n)$
  - $T_n \equiv \{\delta : \ ||\delta||_2 \leq \alpha ||\nabla E(\theta_n)||_2\}$
  - $\alpha \equiv$ **learning rate**
  - ➡ $\delta_n = -\alpha\, \nabla E(\theta_n)$
- Natural gradient descent
  - Fisher information metric
    $F_{ij}(p) \equiv \mathbb{E}_{X\sim p}\left[\partial_{\theta_i} \ln p(X)\ \partial_{\theta_j} \ln p(X)\right]$
  - $T_n(r) = \{\delta : \ \delta^T F(\theta_n)\, \delta \leq r^2\}$
  - $\Leftrightarrow M_n(\delta) = \frac{1}{2}\delta^T F \delta + \nabla E(\theta_n)^T \delta + E(\theta_n)$
  - ➡ $\delta_n = -F^{-1}(\theta_n)\, \nabla E(\theta_n)$
- KFAC (Kronecker-Factored Approximate Curvature)
  - ANNs $\Rightarrow D > 10\ 000$
  - $F^{-1}(\theta_n)\, \nabla E(\theta_n) \Rightarrow O(D^2)$
  - KFAC $\sim$ crude approx of the Fisher metric
  - Direction update using KFAC Fisher
  - Scaling update using *exact* Fisher

Direct application of KFAC

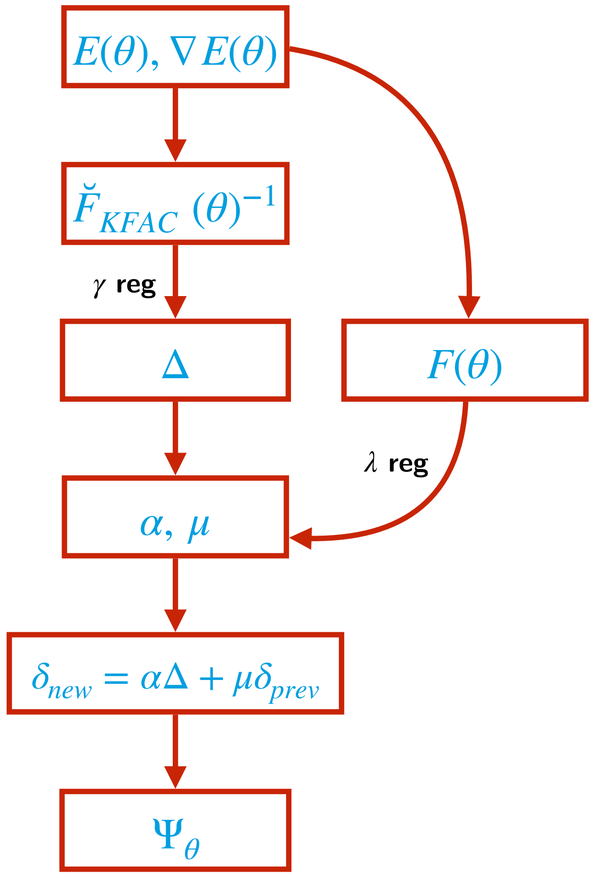
KFAC optimizer

[Martens, Grosse (2015)]

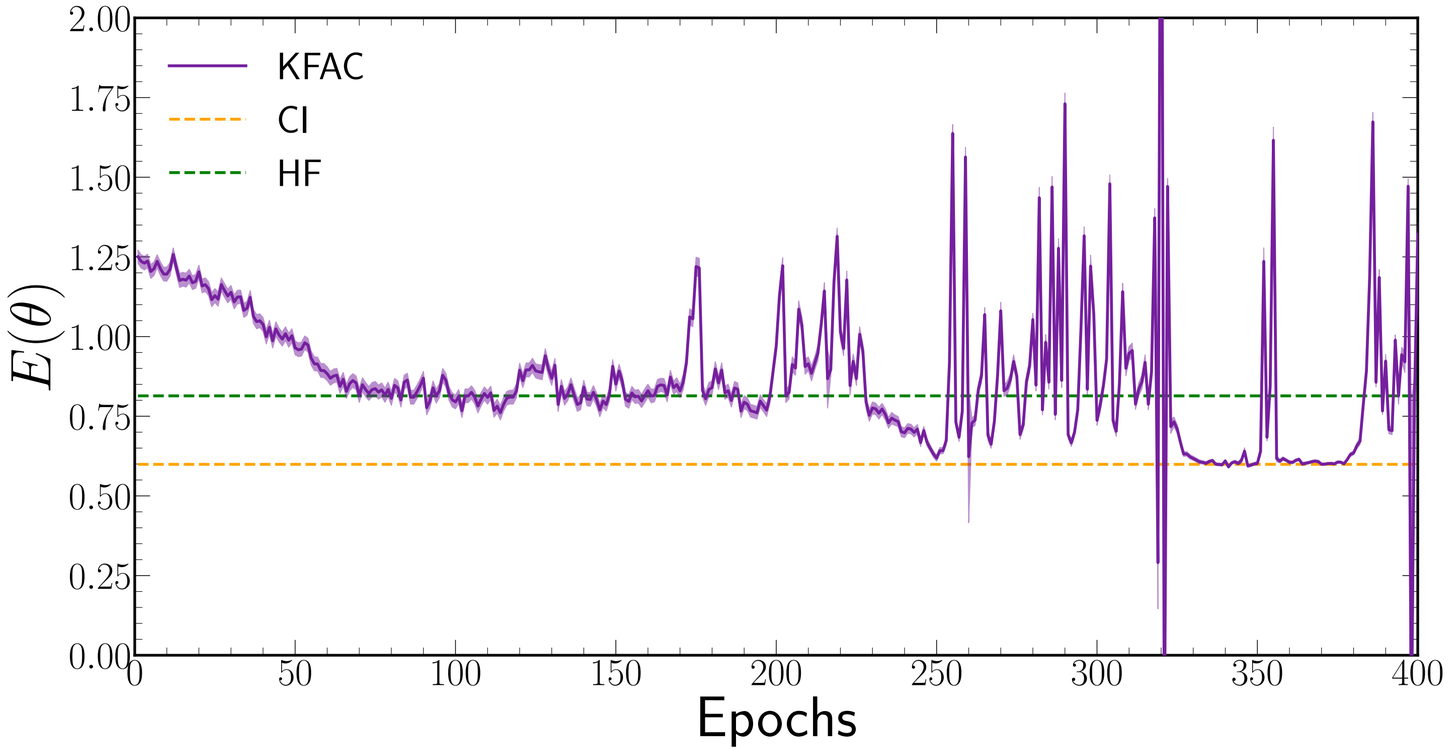


Direct application of KFAC

KFAC optimizer
[Martens, Grosse (2015)]

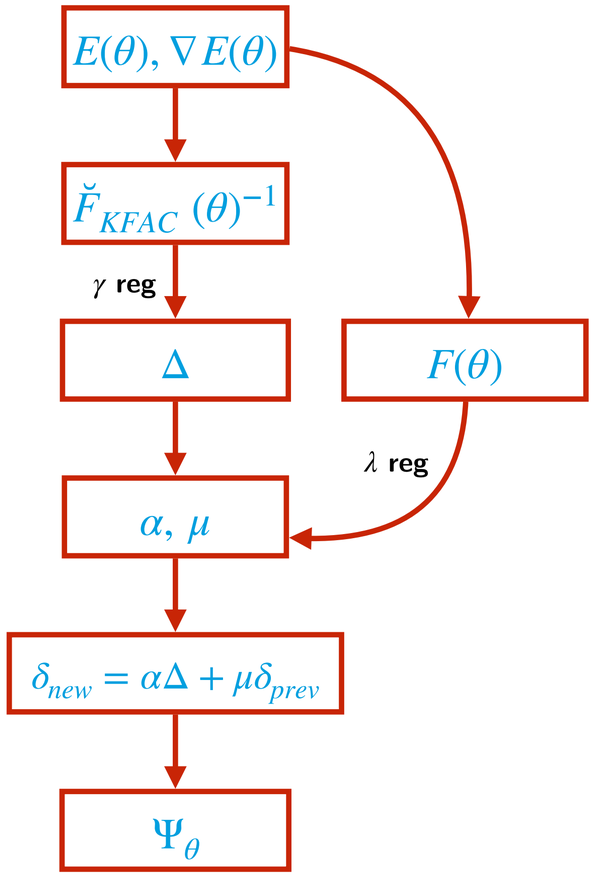


KFAC: $A = 2$, $V_0 = -10$

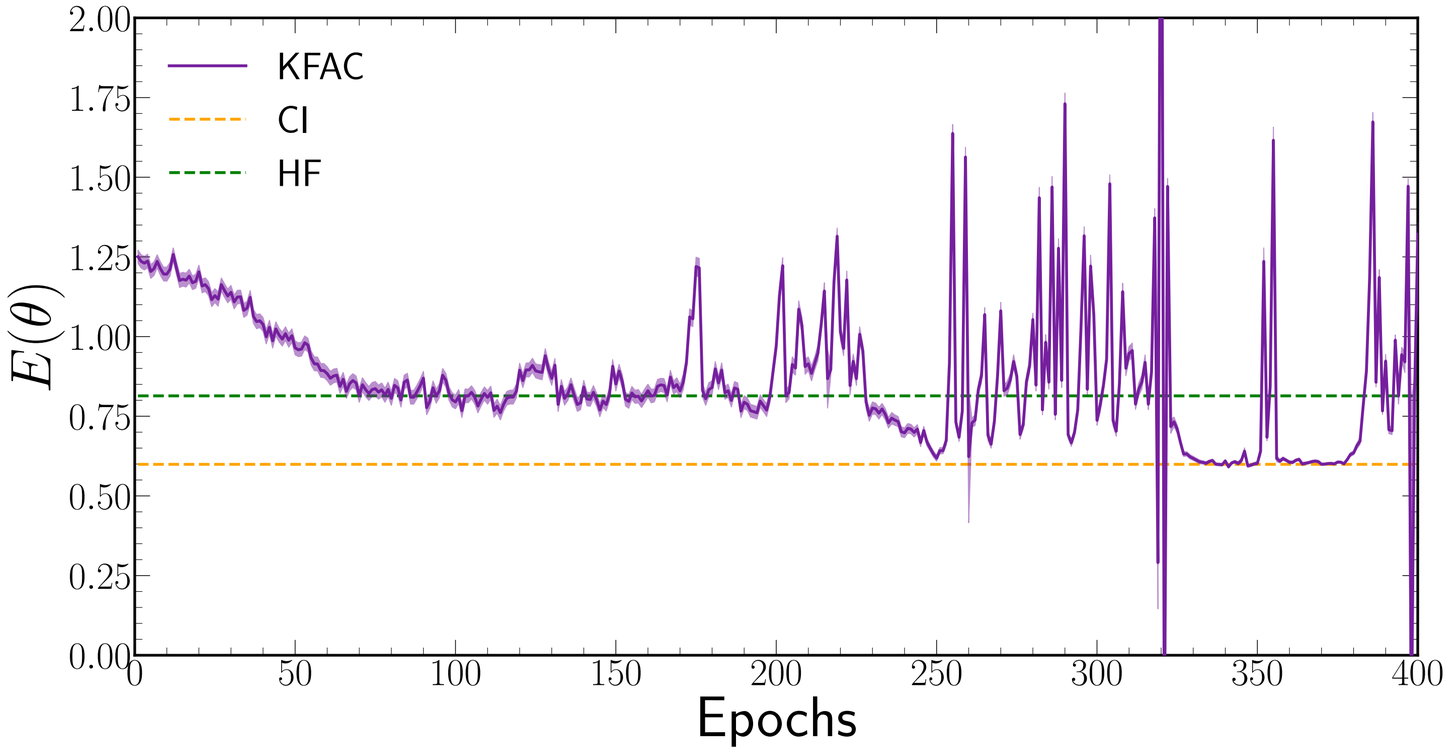


Direct application of KFAC

KFAC optimizer
[Martens, Grosse (2015)]



KFAC: $A = 2$, $V_0 = -10$



Extensive testing

- Sometimes works nicely, sometimes unstable, sometimes fake convergence
- ➡ Difficult to predict performance
- ➡ **Not reliable optimization ⇒ How to improve it ?**

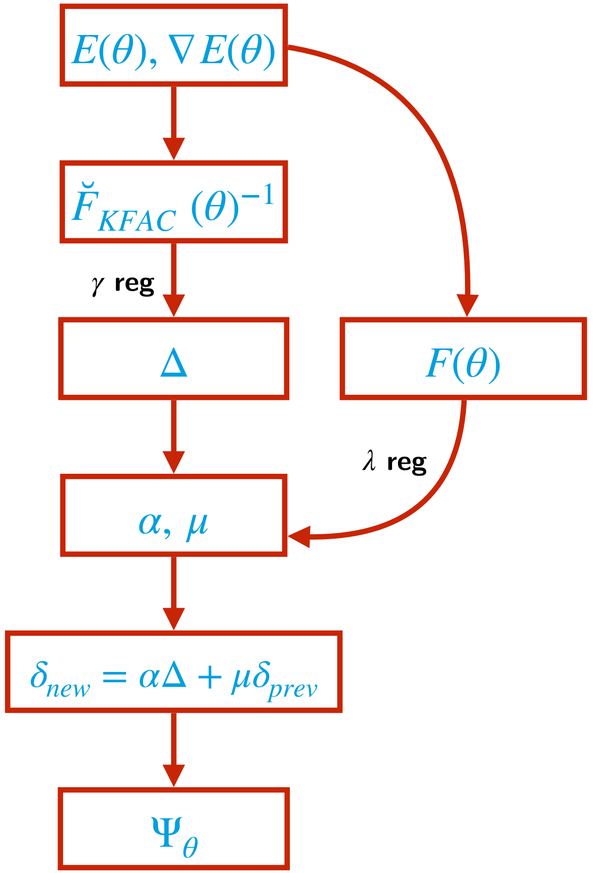
Outline

- **Variational Monte Carlo with Neural Quantum States**
  - Overview of VMC with NQS
  - The Kronecker-Factored Approximate Curvature (KFAC)
- **Augmented KFAC for VMC problems**
  - Scaling improvement from a Quasi-Newton approach
  - Direction improvement from MINRES
- **Decision geometry for VMC**
  - Game theory reformulation of VMC
  - Testing decisional gradient descent

Alternative approach to KFAC shortcomings

Recap: KFAC optimizer

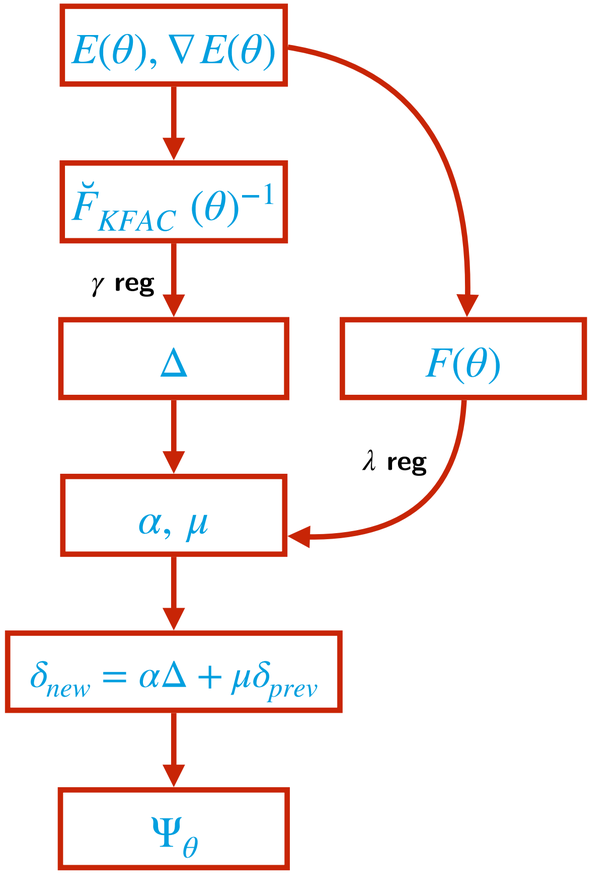
[Martens, Grosse (2015)]



Alternative approach to KFAC shortcomings

Recap: KFAC optimizer

[Martens, Grosse (2015)]



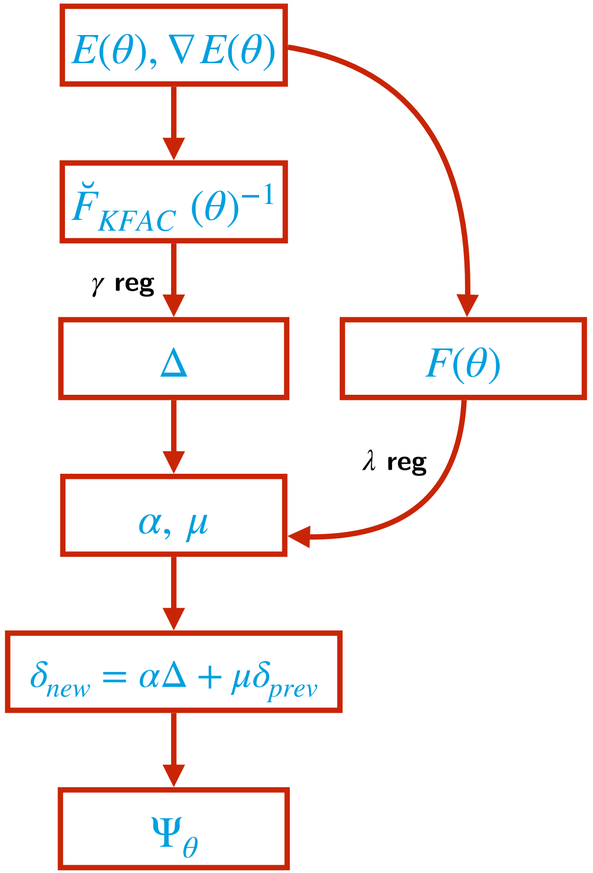
Improving scaling of the update

- Analysis
  - Original argument for KFAC: $F \sim$ Hessian
  - Only valid for supervised learning problems
  - **VMC $\neq$ supervised learning**
- Proposed solution
  - Just use a better quadratic model !

Alternative approach to KFAC shortcomings

Recap: KFAC optimizer

[Martens, Grosse (2015)]



Improving scaling of the update

- Analysis
  - Original argument for KFAC: $F \sim$ Hessian
  - Only valid for supervised learning problems
  - **VMC $\neq$ supervised learning**
- Proposed solution
  - Just use a better quadratic model !

Quasi-Newton KFAC

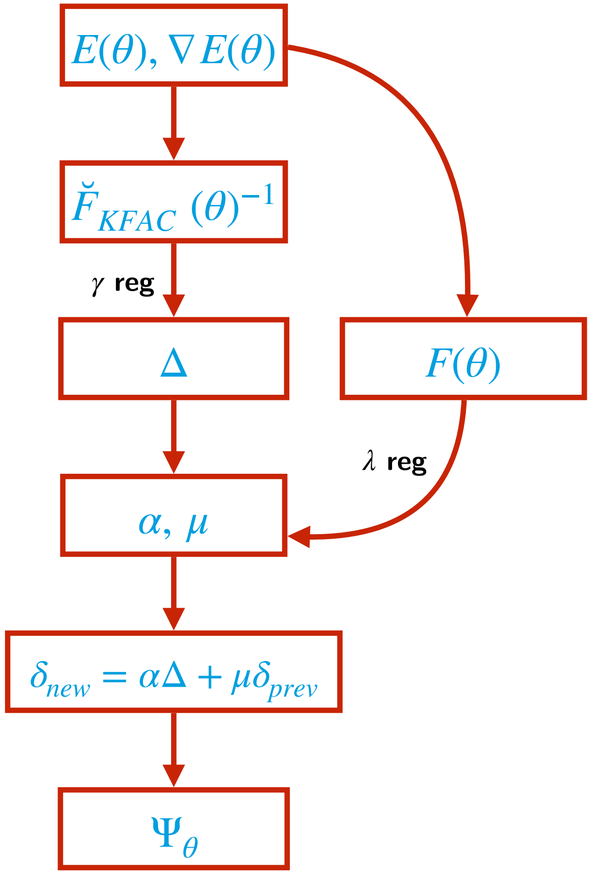
- Supervised learning: [Martens (2020), Amari (2016)]
  - $F(\theta) \sim$ Cost function's Hessian $+ \partial_{\theta_1}\partial_{\theta_2} \ln|\Psi_\theta(X)| = 0$
- In our case: cost function $= E(\theta)$
- Hessian:

$$\begin{aligned}\partial_{\theta_1}\partial_{\theta_2}E(\theta) = {} & 2\mathbb{E}\left[(E_{L,\theta} - E(\theta))\,\partial_{\theta_1}\partial_{\theta_2}\ln|\Psi_\theta(X)|\right] \\ & + 4\mathbb{E}\left[(E_{L,\theta} - E(\theta))\,\partial_{\theta_1}\ln|\Psi_\theta(X)|\partial_{\theta_2}\ln|\Psi_\theta(X)|\right] \\ & + 2\mathbb{E}\left[\partial_{\theta_1}E_{L,\theta}(X)\partial_{\theta_2}\ln|\Psi_\theta(X)|\right]\end{aligned}$$

Alternative approach to KFAC shortcomings

Recap: KFAC optimizer

[Martens, Grosse (2015)]



Improving scaling of the update

- Analysis
  - Original argument for KFAC: $F \sim$ Hessian
  - Only valid for supervised learning problems
  - **VMC $\neq$ supervised learning**
- Proposed solution
  - Just use a better quadratic model !

Quasi-Newton KFAC

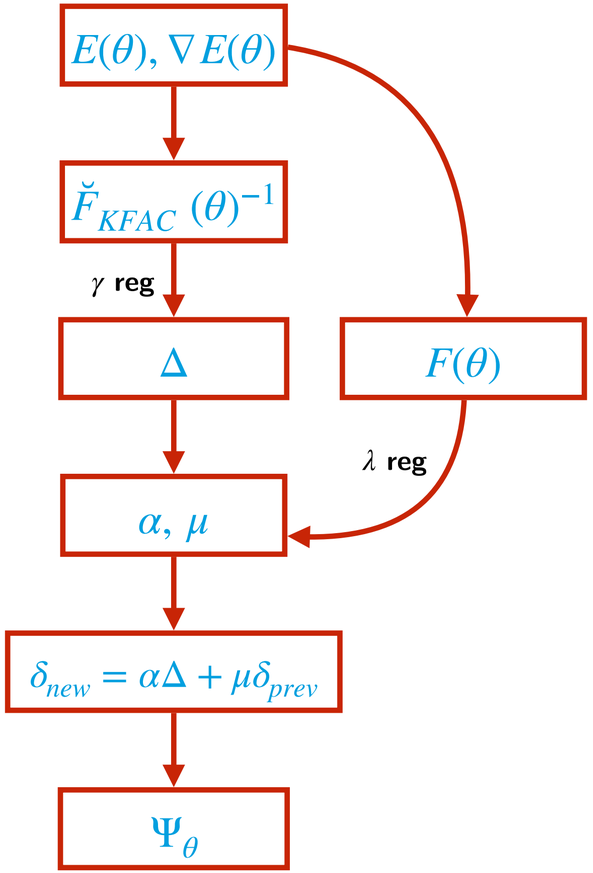
- Supervised learning: [Martens (2020), Amari (2016)]
  - $F(\theta) \sim$ Cost function's Hessian $+ \partial_{\theta_1}\partial_{\theta_2} \ln|\Psi_\theta(X)| = 0$
- In our case: cost function $= E(\theta)$
- Hessian:

$$\begin{aligned}\partial_{\theta_1}\partial_{\theta_2}E(\theta) = & \cancel{2\mathbb{E}\left[(E_{L,\theta} - E(\theta))\,\partial_{\theta_1}\partial_{\theta_2}\ln|\Psi_\theta(X)|\right]} \\ & + 4\mathbb{E}\left[(E_{L,\theta} - E(\theta))\,\partial_{\theta_1}\ln|\Psi_\theta(X)|\partial_{\theta_2}\ln|\Psi_\theta(X)|\right] \\ & + 2\mathbb{E}\left[\partial_{\theta_1}E_{L,\theta}(X)\partial_{\theta_2}\ln|\Psi_\theta(X)|\right]\end{aligned}$$

Alternative approach to KFAC shortcomings

Recap: KFAC optimizer

[Martens, Grosse (2015)]



Improving scaling of the update

- Analysis
  - Original argument for KFAC: $F \sim$ Hessian
  - Only valid for supervised learning problems
  - **VMC $\neq$ supervised learning**
- Proposed solution
  - Just use a better quadratic model !

Quasi-Newton KFAC

- Supervised learning: [Martens (2020), Amari (2016)]
  - $F(\theta) \sim$ Cost function's Hessian $+ \partial_{\theta_1}\partial_{\theta_2} \ln|\Psi_\theta(X)| = 0$
- In our case: cost function $= E(\theta)$
- Hessian:

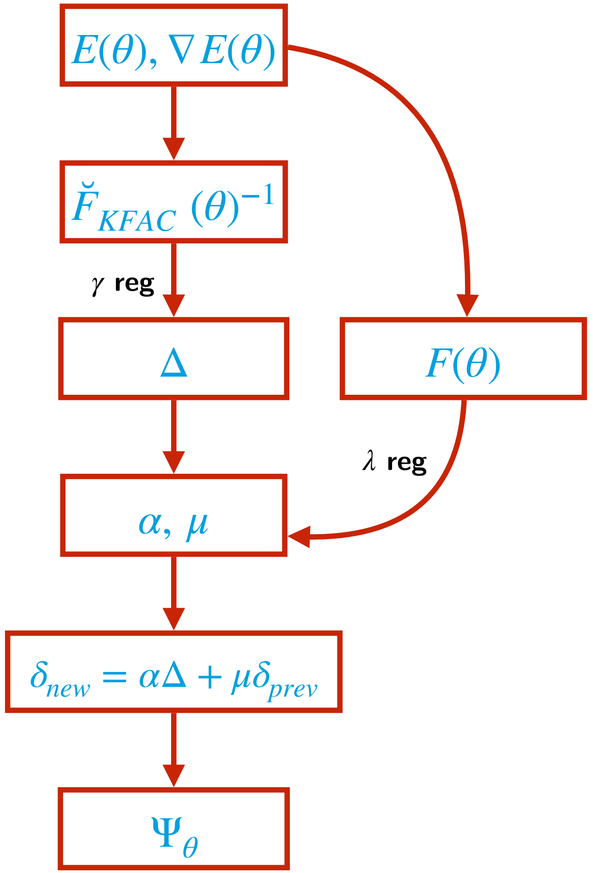
$$\partial_{\theta_1}\partial_{\theta_2} E(\theta) = \cancel{2\mathbb{E}\left[(E_{L,\theta} - E(\theta))\,\partial_{\theta_1}\partial_{\theta_2} \ln|\Psi_\theta(X)|\right]}$$
$$\begin{cases} + 4\mathbb{E}\left[(E_{L,\theta} - E(\theta))\,\partial_{\theta_1} \ln|\Psi_\theta(X)|\partial_{\theta_2} \ln|\Psi_\theta(X)|\right] \\ + 2\mathbb{E}\left[\partial_{\theta_1} E_{L,\theta}(X)\partial_{\theta_2} \ln|\Psi_\theta(X)|\right] \end{cases}$$

Defines our quasi-Hessian $H_Q(\theta)$

Alternative approach to KFAC shortcomings

Recap: KFAC optimizer

[Martens, Grosse (2015)]



Improving scaling of the update

- Analysis
  - Original argument for KFAC: $F \sim$ Hessian
  - Only valid for supervised learning problems
  - **VMC $\neq$ supervised learning**
- Proposed solution
  - Just use a better quadratic model !

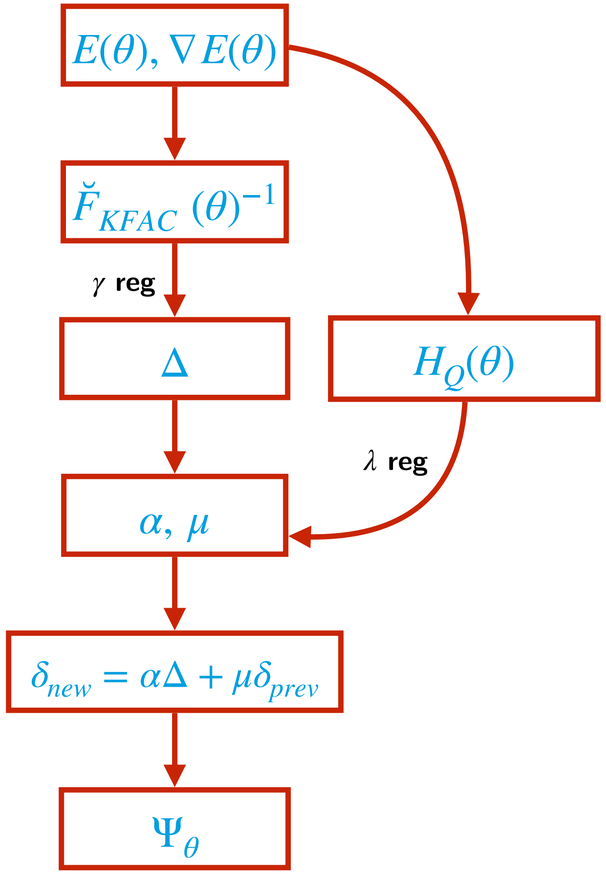
Quasi-Newton KFAC

- Supervised learning: [Martens (2020), Amari (2016)]
  - $F(\theta) \sim$ Cost function's Hessian $+ \partial_{\theta_1}\partial_{\theta_2} \ln|\Psi_\theta(X)| = 0$
- In our case: cost function $= E(\theta)$
- Hessian:

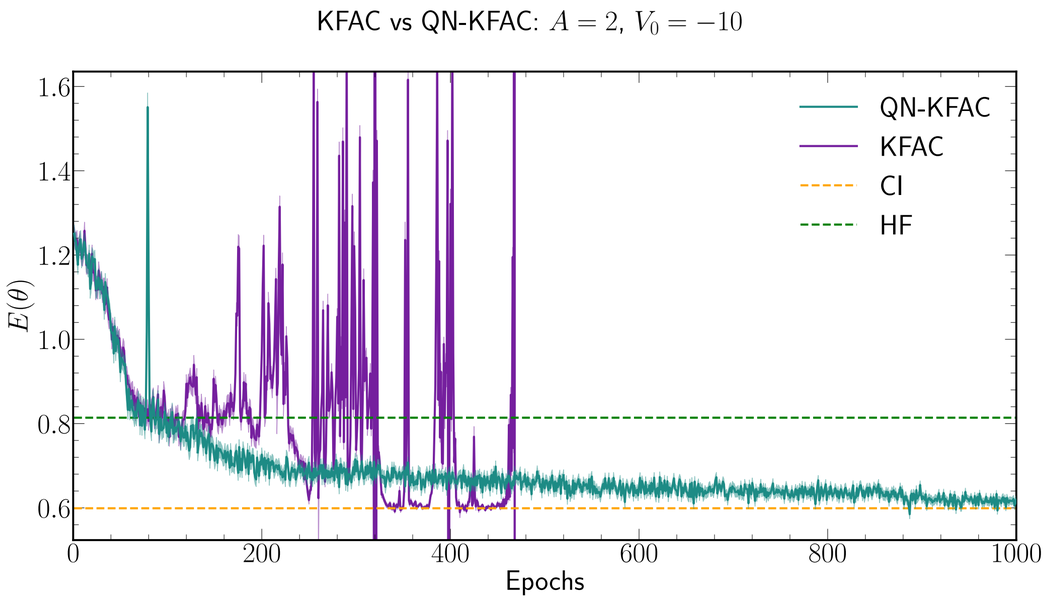
$$\partial_{\theta_1}\partial_{\theta_2} E(\theta) = \cancel{2\mathbb{E}\left[(E_{L,\theta} - E(\theta))\,\partial_{\theta_1}\partial_{\theta_2} \ln|\Psi_\theta(X)|\right]}$$
$$\begin{cases} + 4\mathbb{E}\left[(E_{L,\theta} - E(\theta))\,\partial_{\theta_1} \ln|\Psi_\theta(X)|\partial_{\theta_2} \ln|\Psi_\theta(X)|\right] \\ + 2\mathbb{E}\left[\partial_{\theta_1} E_{L,\theta}(X)\partial_{\theta_2} \ln|\Psi_\theta(X)|\right] \end{cases}$$

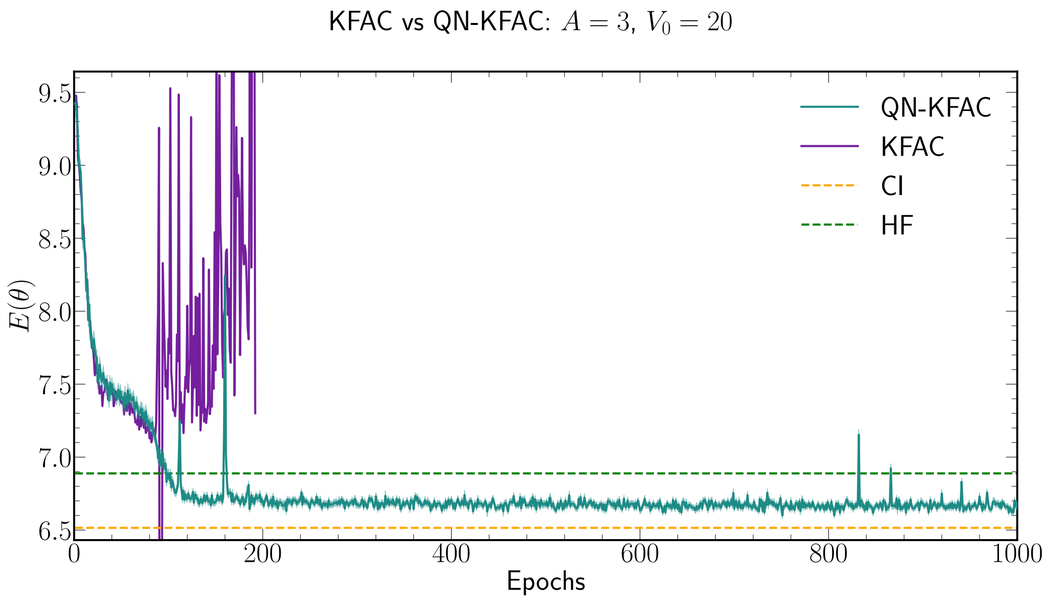
Defines our quasi-Hessian $H_Q(\theta)$

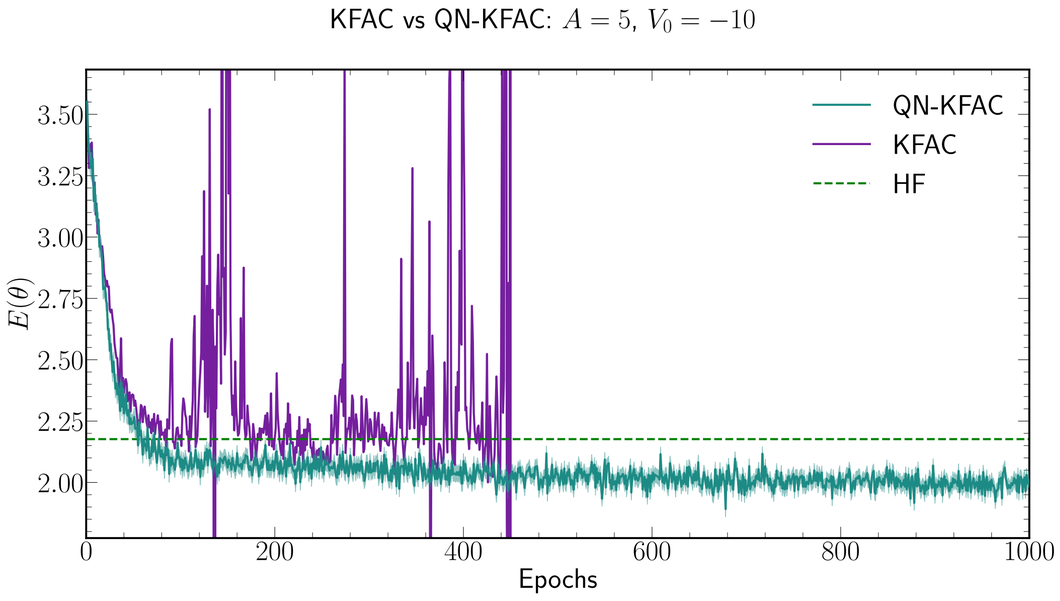
QN-KFAC optimizer



Impact of new re-scaling on convergence







QN-KFAC vs KFAC

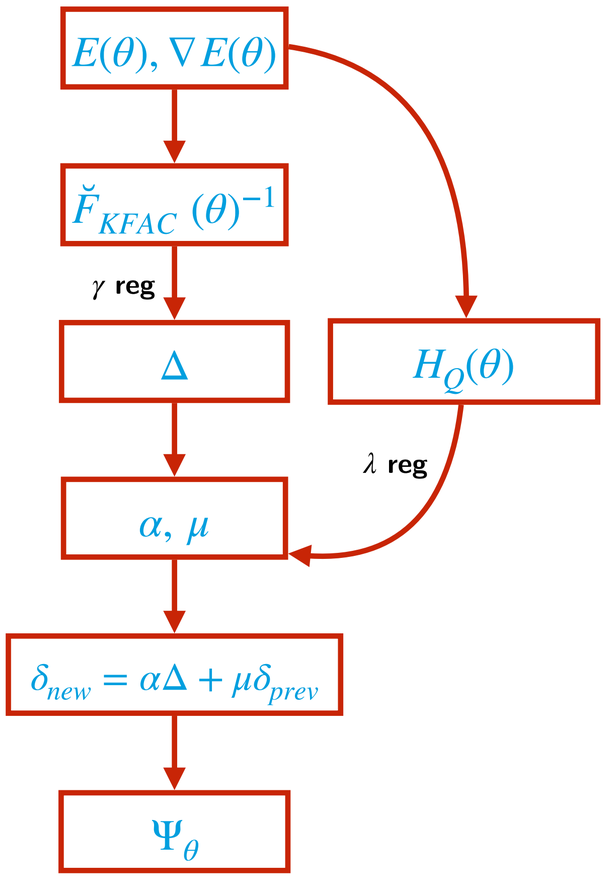
- Overall Improvements
  - Energy fluctuations much reduced
  - Reduction of cases where it get stuck in local minima
- But not perfect
  - Still some instabilities (not shown here because large λ_{init})
  - Can take time to get out of local minima
  - Slow final convergence

Outline

- **Variational Monte Carlo with Neural Quantum States**
  - Overview of VMC with NQS
  - The Kronecker-Factored Approximate Curvature (KFAC)
- **Augmented KFAC for VMC problems**
  - Scaling improvement from a Quasi-Newton approach
  - Direction improvement from MINRES
- **Decision geometry for VMC**
  - Game theory reformulation of VMC
  - Testing decisional gradient descent

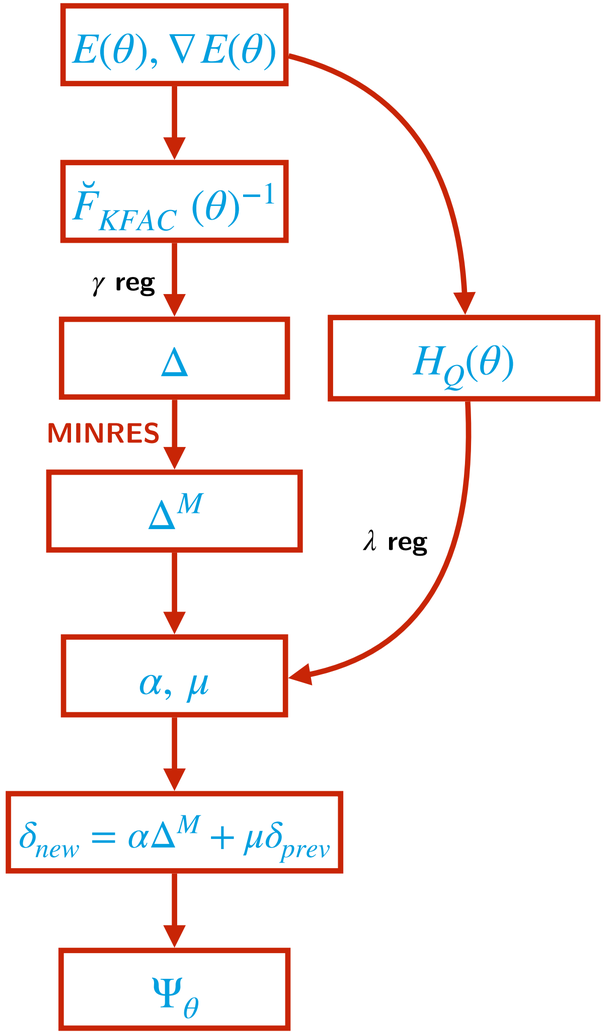
Testing direction improvement with MINRES

QN-KFAC optimizer



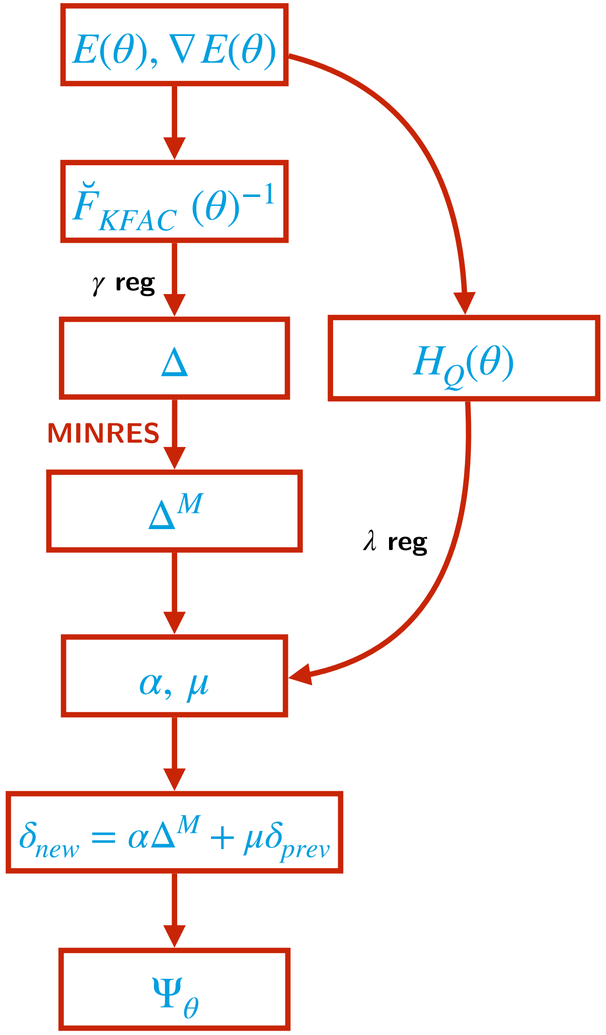
Testing direction improvement with MINRES

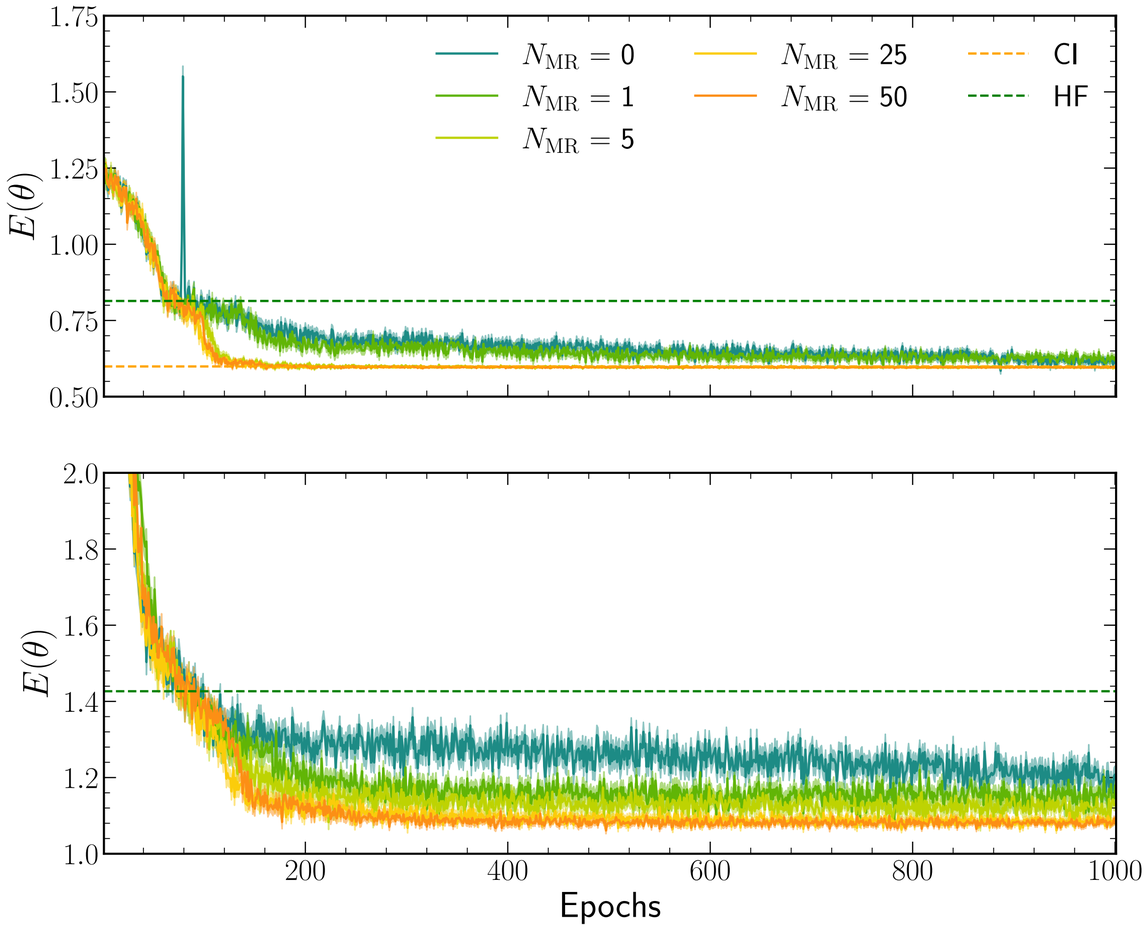
QN-MR-KFAC optimizer



Testing direction improvement with MINRES

QN-MR-KFAC optimizer





$A = 2, \quad V_0 = -10$

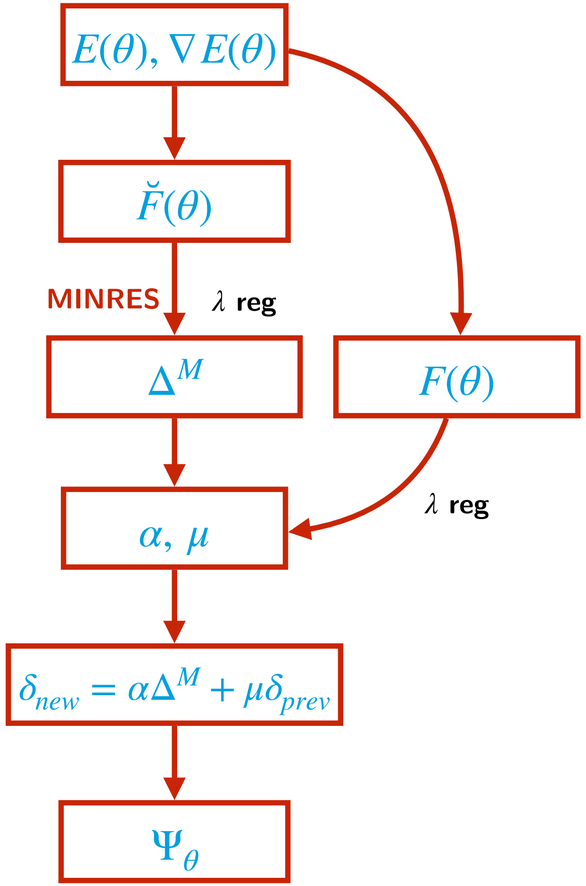
$A = 4, \quad V_0 = -10$

Improving KFAC estimation of direction update

- Test: take Δ as initial guess for MINRES on $\breve{F}(\theta).x = \nabla E(\theta)$ $\Rightarrow$ **Improved direction** Δ^M
- Observation: **MINRES $\Rightarrow$ Better accuracy!** (and in general more stable)

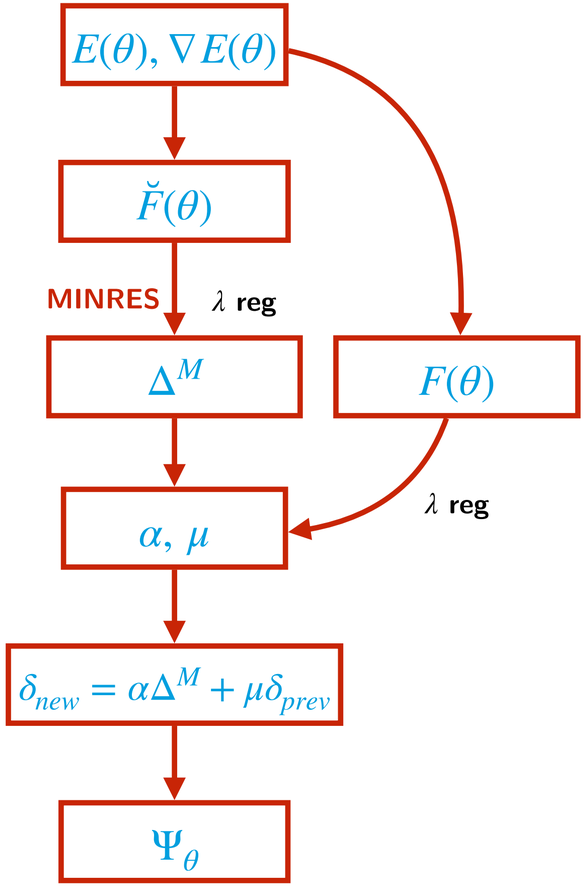
Testing pure NGD with MINRES

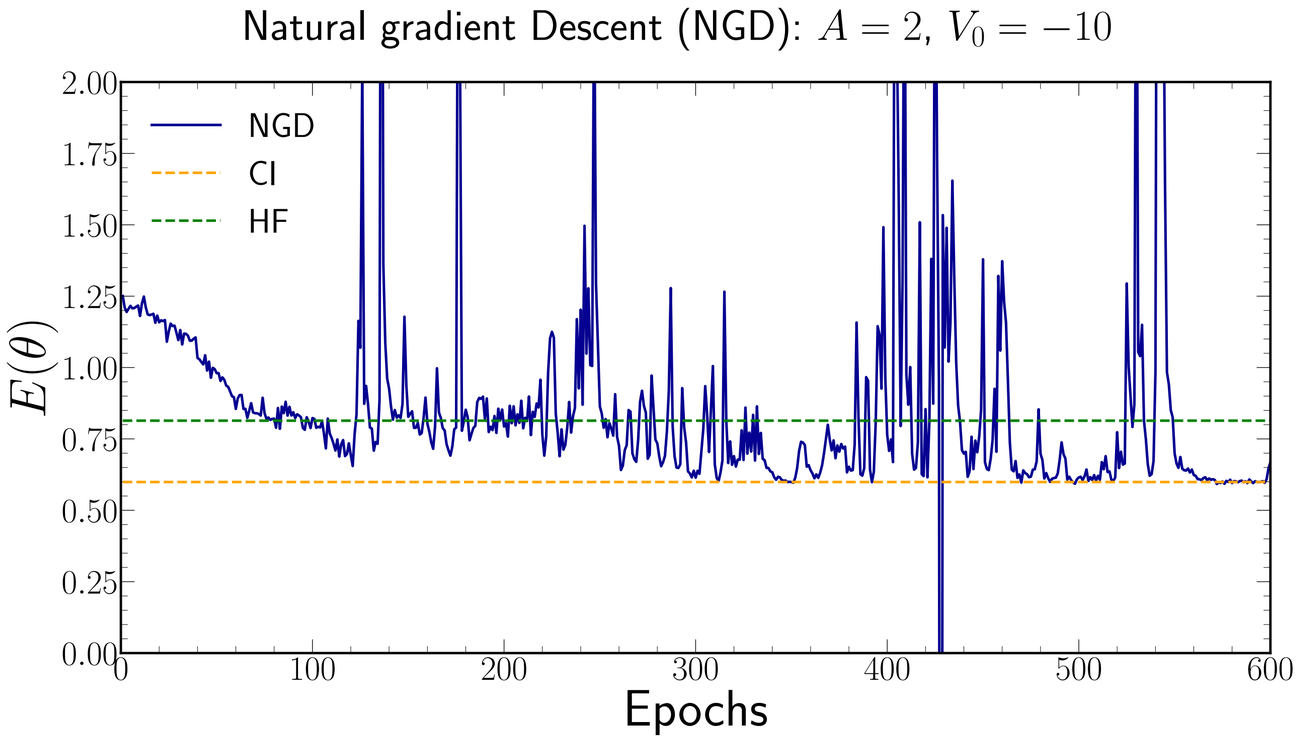
Natural Gradient Descent (NGD)



Testing pure NGD with MINRES

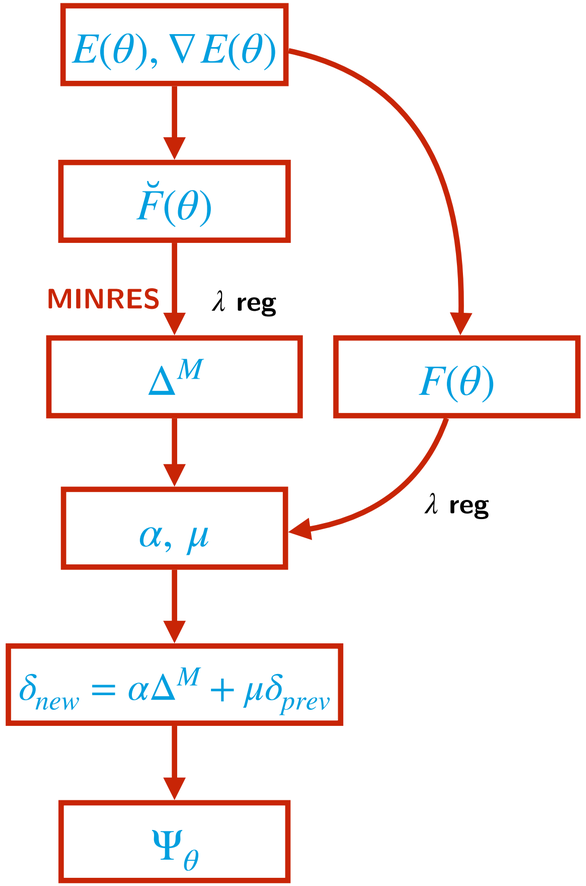
Natural Gradient Descent (NGD)



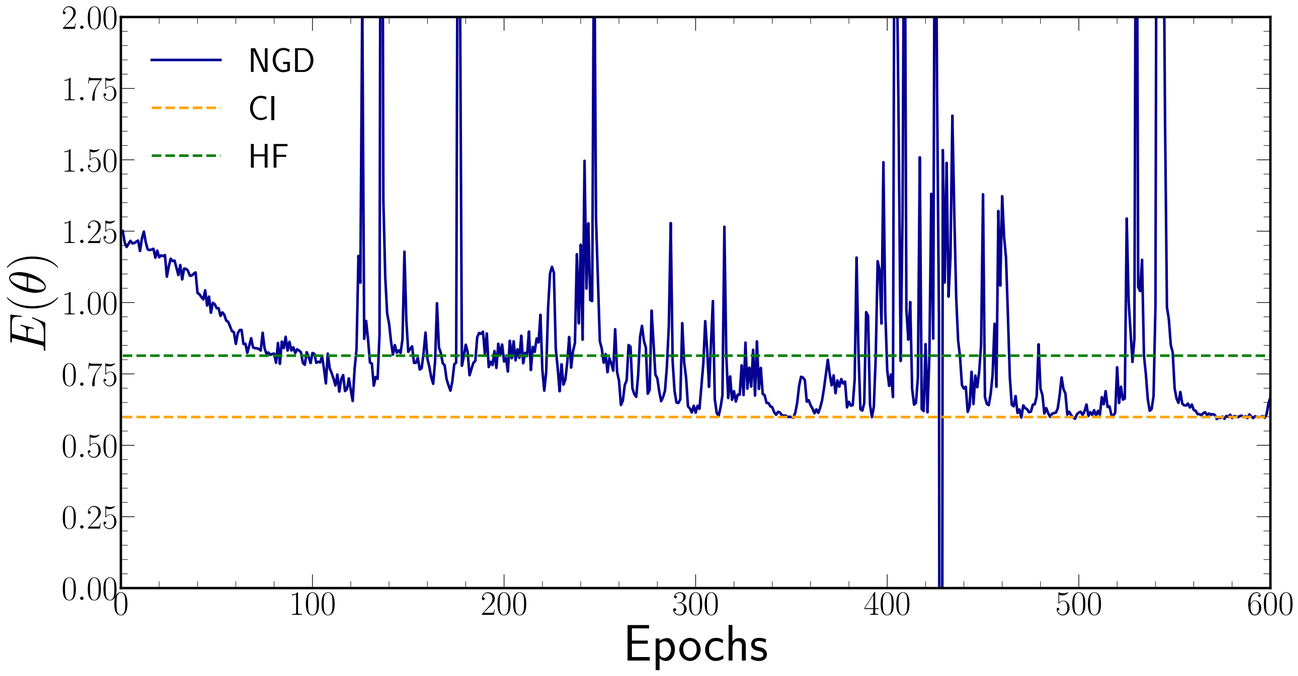


Testing pure NGD with MINRES

Natural Gradient Descent (NGD)



Natural gradient Descent (NGD): $A = 2$, $V_0 = -10$



Failure of Natural Gradient Descent (NGD)

- Testing information geometry with MINRES:
  - Observation: even when using exact Fisher $F(\theta)$ → **huge instabilities**
  - Confirms relevance of H_Q and suggests that information geometry is sub-optimal for VMC
- Can we find better than the Fisher metric?
  - Quasi-Hessian $H_Q \neq$ PSD ⇒ lead to **instabilities** as well

➡ **Better geometry for VMC?**

Outline

- **Variational Monte Carlo with Neural Quantum States**
  - Overview of VMC with NQS
  - The Kronecker-Factored Approximate Curvature (KFAC)
- **Augmented KFAC for VMC problems**
  - Scaling improvement from a Quasi-Newton approach
  - Direction improvement from MINRES
- **Decision geometry for VMC**
  - Game theory reformulation of VMC
  - Testing decisional gradient descent

From information to decision geometry

Supervised learning problem

- Minimize $L(\theta) = \mathbb{E}_{X\sim q}\left[-\ln p_\theta(X)\right]$ (cross-entropy loss)
  - $q \equiv$ target distribution, $p_\theta \equiv$ model to optimize
  - Equivalent to "fitting data points" problems

From information to decision geometry

Supervised learning problem

- Minimize $L(\theta) = \mathbb{E}_{X\sim q}\left[-\ln p_\theta(X)\right]$ (cross-entropy loss)
  - $q \equiv$ target distribution, $p_\theta \equiv$ model to optimize
  - Equivalent to "fitting data points" problems

Natural gradient descent [Amari (1997)]

- Local problem: solve for δ such that $||\delta||_F = \text{cst}$
- Kullback-Leibler divergence and the Fisher matrix
  - $D_{\text{KL}}(p_1, p_2) \equiv \mathbb{E}_{X\sim p_1}\left[-\ln p_2(X) - (-\ln p_1(X))\right]$
  - $D_{\text{KL}}(p_\theta, p_{\theta+\delta}) = \frac{1}{2}\delta^T F(\theta)\delta + O(\delta^3)$ **Information geometry**

➡ Fisher metric: $F(\theta)_{\theta_1\theta_2} \equiv \mathbb{E}_{X\sim p_\theta}\left[\partial_{\theta_1}\ln p_\theta(X)\ \partial_{\theta_2}\ln p_\theta(X)\right]$

➡ $\delta_{NGD} = -F^{-1}(\theta)\ \nabla L(\theta)$

From information to decision geometry

Supervised learning problem

- Minimize $L(\theta) = \mathbb{E}_{X\sim q}\left[-\ln p_\theta(X)\right]$ (cross-entropy loss)
  - $q \equiv$ target distribution, $p_\theta \equiv$ model to optimize
  - Equivalent to "fitting data points" problems

Natural gradient descent [Amari (1997)]

- Local problem: solve for δ such that $||\delta||_F = \text{cst}$
- Kullback-Leibler divergence and the Fisher matrix
  - $D_{\text{KL}}(p_1, p_2) \equiv \mathbb{E}_{X\sim p_1}\left[-\ln p_2(X) - (-\ln p_1(X))\right]$
  - $D_{\text{KL}}(p_\theta, p_{\theta+\delta}) = \frac{1}{2}\delta^T F(\theta)\delta + O(\delta^3)$ **Information geometry**

➡ Fisher metric: $F(\theta)_{\theta_1\theta_2} \equiv \mathbb{E}_{X\sim p_\theta}\left[\partial_{\theta_1}\ln p_\theta(X)\ \partial_{\theta_2}\ln p_\theta(X)\right]$

➡ $\delta_{NGD} = -F^{-1}(\theta)\ \nabla L(\theta)$

Efficient implementation

[Martens, Grosse (2015)]

- KFAC (Kronecker-Factored Approximate Curvature)
  - KFAC $\sim$ crude approximation of the Fisher metric
  - Direction update using KFAC Fisher
  - Scaling update using *exact* Fisher

➡ **Fast and reliable convergence**

From information to decision geometry

Supervised learning problem

- Minimize $L(\theta) = \mathbb{E}_{X\sim q}\left[-\ln p_\theta(X)\right]$ (cross-entropy loss)
  - $q \equiv$ target distribution, $p_\theta \equiv$ model to optimize
  - Equivalent to "fitting data points" problems

Non supervised learning problem

- Minimize $h(\theta) = -\,\mathbb{E}_{X\sim p_\theta}[S(X, p_\theta)] \;\equiv -\,S(p_\theta, p_\theta)$
  - $S \equiv$ scoring rule, $p_\theta \equiv$ model to optimize
  - Very general problem

Natural gradient descent [Amari (1997)]

- Local problem: solve for δ such that $||\delta||_F = \mathrm{cst}$
- Kullback-Leibler divergence and the Fisher matrix
  - $D_{\mathrm{KL}}(p_1, p_2) \equiv \mathbb{E}_{X\sim p_1}\left[-\ln p_2(X) - (-\ln p_1(X))\right]$
  - $D_{\mathrm{KL}}(p_\theta, p_{\theta+\delta}) = \frac{1}{2}\delta^T F(\theta)\delta + O(\delta^3)$ **Information geometry**

➡ Fisher metric: $F(\theta)_{\theta_1\theta_2} \equiv \mathbb{E}_{X\sim p_\theta}\left[\partial_{\theta_1}\ln p_\theta(X)\;\partial_{\theta_2}\ln p_\theta(X)\right]$

➡ $\delta_{NGD} = -\,F^{-1}(\theta)\;\nabla L(\theta)$

Efficient implementation

[Martens, Grosse (2015)]

- KFAC (Kronecker-Factored Approximate Curvature)
  - KFAC $\sim$ crude approximation of the Fisher metric
  - Direction update using KFAC Fisher
  - Scaling update using *exact* Fisher

➡ **Fast and reliable convergence**

From information to decision geometry

Supervised learning problem

- Minimize $L(\theta) = \mathbb{E}_{X\sim q}\left[-\ln p_\theta(X)\right]$ (cross-entropy loss)
  - $q \equiv$ target distribution, $p_\theta \equiv$ model to optimize
  - Equivalent to "fitting data points" problems

Non supervised learning problem

- Minimize $h(\theta) = -\,\mathbb{E}_{X\sim p_\theta}[S(X, p_\theta)] \;\equiv -\,S(p_\theta, p_\theta)$
  - $S \equiv$ scoring rule, $p_\theta \equiv$ model to optimize
  - Very general problem

Natural gradient descent [Amari (1997)]

- Local problem: solve for δ such that $||\delta||_F = \text{cst}$
- Kullback-Leibler divergence and the Fisher matrix
  - $D_{\text{KL}}(p_1, p_2) \equiv \mathbb{E}_{X\sim p_1}\left[-\ln p_2(X) - (-\ln p_1(X))\right]$
  - $D_{\text{KL}}(p_\theta, p_{\theta+\delta}) = \frac{1}{2}\delta^T F(\theta)\delta + O(\delta^3)$ **Information geometry**

➡ Fisher metric: $F(\theta)_{\theta_1\theta_2} \equiv \mathbb{E}_{X\sim p_\theta}\left[\partial_{\theta_1} \ln p_\theta(X)\; \partial_{\theta_2} \ln p_\theta(X)\right]$

➡ $\delta_{NGD} = -\,F^{-1}(\theta)\;\nabla L(\theta)$

Decision geometry [Dawid (2006)]

- Necessary condition
  - $\forall p, q, \quad S(p,p) \leq S(p,q) \;\Rightarrow$ *proper* scoring rule [Gneiting, Raftery (2007)]
- Game-theory generalizations
  - Entropy: $H(p) \equiv S(p,p)$
  - Cross-entropy: $H(p,q) \equiv S(p,q)$
  - Divergence: $D_S(p,q) \equiv S(p,q) - S(p,p)$
  - $D_S(p_\theta, p_{\theta+\delta}) = \frac{1}{2}\delta^T G_S(\theta)\delta + O(\delta^3)$ **Decision geometry**

➡ $\delta_{DGD} = -\,G_S^{-1}(\theta)\;\nabla h(\theta)$

- Recovers information geometry: $S(p,x) = -\ln p(x)$

Efficient implementation

[Martens, Grosse (2015)]

- KFAC (Kronecker-Factored Approximate Curvature)
  - KFAC $\sim$ crude approximation of the Fisher metric
  - Direction update using KFAC Fisher
  - Scaling update using *exact* Fisher

➡ **Fast and reliable convergence**

From information to decision geometry

Supervised learning problem

- Minimize $L(\theta) = \mathbb{E}_{X\sim q}\left[-\ln p_\theta(X)\right]$ (cross-entropy loss)
  - $q \equiv$ target distribution, $p_\theta \equiv$ model to optimize
  - Equivalent to "fitting data points" problems

Non supervised learning problem

- Minimize $h(\theta) = -\,\mathbb{E}_{X\sim p_\theta}[S(X, p_\theta)] \equiv -\,S(p_\theta, p_\theta)$
  - $S \equiv$ scoring rule, $p_\theta \equiv$ model to optimize
  - Very general problem

Natural gradient descent [Amari (1997)]

- Local problem: solve for δ such that $||\delta||_F = \text{cst}$
- Kullback-Leibler divergence and the Fisher matrix
  - $D_{\text{KL}}(p_1, p_2) \equiv \mathbb{E}_{X\sim p_1}\left[-\ln p_2(X) - (-\ln p_1(X))\right]$
  - $D_{\text{KL}}(p_\theta, p_{\theta+\delta}) = \frac{1}{2}\delta^T F(\theta)\delta + O(\delta^3)$ **Information geometry**

➡ Fisher metric: $F(\theta)_{\theta_1\theta_2} \equiv \mathbb{E}_{X\sim p_\theta}\left[\partial_{\theta_1}\ln p_\theta(X)\ \partial_{\theta_2}\ln p_\theta(X)\right]$

➡ $\delta_{NGD} = -\,F^{-1}(\theta)\ \nabla L(\theta)$

Decision geometry [Dawid (2006)]

- Necessary condition
  - $\forall p, q, \quad S(p,p) \leq S(p,q) \ \Rightarrow$ *proper* scoring rule [Gneiting, Raftery (2007)]
- Game-theory generalizations
  - Entropy: $H(p) \equiv S(p,p)$
  - Cross-entropy: $H(p,q) \equiv S(p,q)$
  - Divergence: $D_S(p,q) \equiv S(p,q) - S(p,p)$
  - $D_S(p_\theta, p_{\theta+\delta}) = \frac{1}{2}\delta^T G_S(\theta)\delta + O(\delta^3)$ **Decision geometry**

➡ $\delta_{DGD} = -\,G_S^{-1}(\theta)\ \nabla h(\theta)$

- Recovers information geometry: $S(p,x) = -\ln p(x)$

Efficient implementation

[Martens, Grosse (2015)]

- KFAC (Kronecker-Factored Approximate Curvature)
  - KFAC $\sim$ crude approximation of the Fisher metric
  - Direction update using KFAC Fisher
  - Scaling update using *exact* Fisher

➡ **Fast and reliable convergence**

Open questions

- When is a scoring rule leading to efficient DGD?
- Efficient implementation?
- ...

From information to decision geometry

Supervised learning problem

- Minimize $L(\theta) = \mathbb{E}_{X\sim q}\left[-\ln p_\theta(X)\right]$ (cross-entropy loss)
  - $q \equiv$ target distribution, $p_\theta \equiv$ model to optimize
  - Equivalent to "fitting data points" problems

Non supervised learning problem

- Minimize $h(\theta) = -\,\mathbb{E}_{X\sim p_\theta}[S(X, p_\theta)] \;\equiv -\,S(p_\theta, p_\theta)$
  - $S \equiv$ scoring rule, $p_\theta \equiv$ model to optimize
  - Very general problem

Variational Monte Carlo problem

- Minimize $E(\theta) = \mathbb{E}_{X\sim|\Psi_\theta|^2}[E_{L,\theta}(X)]$
  - $E_{L,\theta}(X) \equiv -\frac{1}{2}\sum_i \left[\partial^2_{x_i} \ln|\Psi_\theta(X)| + \left(\partial_{x_i} \ln|\Psi_\theta(X)|\right)^2\right] + V(X)$
  - Quantum many-body problem

Natural gradient descent [Amari (1997)]

- Local problem: solve for δ such that $||\delta||_F = \text{cst}$
- Kullback-Leibler divergence and the Fisher matrix
  - $D_{\text{KL}}(p_1, p_2) \equiv \mathbb{E}_{X\sim p_1}\left[-\ln p_2(X) - (-\ln p_1(X))\right]$
  - $D_{\text{KL}}(p_\theta, p_{\theta+\delta}) = \frac{1}{2}\delta^T F(\theta)\delta + O(\delta^3)$ **Information geometry**

➡ Fisher metric: $F(\theta)_{\theta_1\theta_2} \equiv \mathbb{E}_{X\sim p_\theta}\left[\partial_{\theta_1} \ln p_\theta(X)\; \partial_{\theta_2} \ln p_\theta(X)\right]$

➡ $\delta_{NGD} = -\,F^{-1}(\theta)\;\nabla L(\theta)$

Decision geometry [Dawid (2006)]

- Necessary condition
  - $\forall p, q, \quad S(p,p) \leq S(p,q) \;\Rightarrow$ *proper* scoring rule [Gneiting, Raftery (2007)]
- Game-theory generalizations
  - Entropy: $H(p) \equiv S(p,p)$
  - Cross-entropy: $H(p,q) \equiv S(p,q)$
  - Divergence: $D_S(p,q) \equiv S(p,q) - S(p,p)$
  - $D_S(p_\theta, p_{\theta+\delta}) = \frac{1}{2}\delta^T G_S(\theta)\delta + O(\delta^3)$ **Decision geometry**

➡ $\delta_{DGD} = -\,G_S^{-1}(\theta)\;\nabla h(\theta)$

- Recovers information geometry: $S(p,x) = -\ln p(x)$

Efficient implementation

[Martens, Grosse (2015)]

- KFAC (Kronecker-Factored Approximate Curvature)
  - KFAC $\sim$ crude approximation of the Fisher metric
  - Direction update using KFAC Fisher
  - Scaling update using *exact* Fisher

➡ **Fast and reliable convergence**

Open questions

- When is a scoring rule leading to efficient DGD?
- Efficient implementation?
- ...

From information to decision geometry

Supervised learning problem

- Minimize $L(\theta) = \mathbb{E}_{X\sim q}\left[-\ln p_\theta(X)\right]$ (cross-entropy loss)
  - $q \equiv$ target distribution, $p_\theta \equiv$ model to optimize
  - Equivalent to "fitting data points" problems

Non supervised learning problem

- Minimize $h(\theta) = -\,\mathbb{E}_{X\sim p_\theta}[S(X, p_\theta)] \;\equiv -\,S(p_\theta, p_\theta)$
  - $S \equiv$ scoring rule, $p_\theta \equiv$ model to optimize
  - Very general problem

Variational Monte Carlo problem

- Minimize $E(\theta) = \mathbb{E}_{X\sim|\Psi_\theta|^2}[E_{L,\theta}(X)]$
  - $E_{L,\theta}(X) \equiv -\frac{1}{2}\sum_i \left[\partial^2_{x_i}\ln|\Psi_\theta(X)| + \left(\partial_{x_i}\ln|\Psi_\theta(X)|\right)^2\right] + V(X)$
  - Quantum many-body problem

Natural gradient descent [Amari (1997)]

- Local problem: solve for δ such that $||\delta||_F = \text{cst}$
- Kullback-Leibler divergence and the Fisher matrix
  - $D_{\text{KL}}(p_1, p_2) \equiv \mathbb{E}_{X\sim p_1}\left[-\ln p_2(X) - (-\ln p_1(X))\right]$
  - $D_{\text{KL}}(p_\theta, p_{\theta+\delta}) = \frac{1}{2}\delta^T F(\theta)\delta + O(\delta^3)$ **Information geometry**

➡ Fisher metric: $F(\theta)_{\theta_1\theta_2} \equiv \mathbb{E}_{X\sim p_\theta}\left[\partial_{\theta_1}\ln p_\theta(X)\;\partial_{\theta_2}\ln p_\theta(X)\right]$

➡ $\delta_{NGD} = -\,F^{-1}(\theta)\;\nabla L(\theta)$

Efficient implementation

[Martens, Grosse (2015)]

- KFAC (Kronecker-Factored Approximate Curvature)
  - KFAC $\sim$ crude approximation of the Fisher metric
  - Direction update using KFAC Fisher
  - Scaling update using *exact* Fisher

➡ **Fast and reliable convergence**

Decision geometry [Dawid (2006)]

- Necessary condition
  - $\forall p, q, \quad S(p,p) \leq S(p,q) \;\Rightarrow$ *proper* scoring rule [Gneiting, Raftery (2007)]
- Game-theory generalizations
  - Entropy: $H(p) \equiv S(p,p)$
  - Cross-entropy: $H(p,q) \equiv S(p,q)$
  - Divergence: $D_S(p,q) \equiv S(p,q) - S(p,p)$
  - $D_S(p_\theta, p_{\theta+\delta}) = \frac{1}{2}\delta^T G_S(\theta)\delta + O(\delta^3)$ **Decision geometry**

➡ $\delta_{DGD} = -\,G_S^{-1}(\theta)\;\nabla h(\theta)$

- Recovers information geometry: $S(p,x) = -\ln p(x)$

Open questions

- When is a scoring rule leading to efficient DGD?
- Efficient implementation?
- …

Game-theory reformulation of VMC

- Natural scoring rule
  - $\forall p_\theta, x,\; S_{VMC}(x, p_\theta) \equiv -\,E_{L,\theta}(x) \rightarrow$ **Proper scoring rule**
- Induced geometry (Divergence)
  - $D_{VMC}(p_\theta, p_{\theta'}) = \frac{1}{8}\sum_i \mathbb{E}\left[\left(\partial_{x_i}\ln p_\theta(X) - \partial_{x_i}\ln p_{\theta'}(X)\right)^2\right]$
  - $G_{VMC}(\theta)_{\theta_1\theta_2} = \frac{1}{4}\sum_i \mathbb{E}\left[\left(\partial_{\theta_1}\partial_{x_i}\ln p_\theta(X)\right)\left(\partial_{\theta_2}\partial_{x_i}\ln p_\theta(X)\right)\right]$

➡ $\delta_{VMC} = -\,G_{VMC}^{-1}(\theta)\;\nabla E(\theta)$

- Physically motivated geometry

$$D_{VMC}(p_\theta, p_{\theta'}) = \mathbb{E}_{X\sim p_\theta}\left[E_{L,\theta}(X) - E_{L,\theta'}(X)\right]$$
$$\simeq E(\theta) - E(\theta') \text{ (up to re-weighting)}$$

From information to decision geometry

Supervised learning problem

- Minimize $L(\theta) = \mathbb{E}_{X\sim q}\left[-\ln p_\theta(X)\right]$ (cross-entropy loss)
  - $q \equiv$ target distribution, $p_\theta \equiv$ model to optimize
  - Equivalent to "fitting data points" problems

Non supervised learning problem

- Minimize $h(\theta) = -\,\mathbb{E}_{X\sim p_\theta}[S(X, p_\theta)] \;\equiv -\,S(p_\theta, p_\theta)$
  - $S \equiv$ scoring rule, $p_\theta \equiv$ model to optimize
  - Very general problem

Variational Monte Carlo problem

- Minimize $E(\theta) = \mathbb{E}_{X\sim|\Psi_\theta|^2}[E_{L,\theta}(X)]$
  - $E_{L,\theta}(X) \equiv -\frac{1}{2}\sum_i \left[\partial^2_{x_i}\ln|\Psi_\theta(X)| + \left(\partial_{x_i}\ln|\Psi_\theta(X)|\right)^2\right] + V(X)$
  - Quantum many-body problem

Natural gradient descent [Amari (1997)]

- Local problem: solve for δ such that $||\delta||_F = \text{cst}$
- Kullback-Leibler divergence and the Fisher matrix
  - $D_{\text{KL}}(p_1, p_2) \equiv \mathbb{E}_{X\sim p_1}\left[-\ln p_2(X) - (-\ln p_1(X))\right]$
  - $D_{\text{KL}}(p_\theta, p_{\theta+\delta}) = \frac{1}{2}\delta^T F(\theta)\delta + O(\delta^3)$ **Information geometry**
  - ➡ Fisher metric: $F(\theta)_{\theta_1\theta_2} \equiv \mathbb{E}_{X\sim p_\theta}\left[\partial_{\theta_1}\ln p_\theta(X)\;\partial_{\theta_2}\ln p_\theta(X)\right]$
  - ➡ $\delta_{NGD} = -\,F^{-1}(\theta)\;\nabla L(\theta)$

Decision geometry [Dawid (2006)]

- Necessary condition
  - $\forall p, q, \quad S(p,p) \leq S(p,q) \;\Rightarrow$ *proper* scoring rule [Gneiting, Raftery (2007)]
- Game-theory generalizations
  - Entropy: $H(p) \equiv S(p,p)$
  - Cross-entropy: $H(p,q) \equiv S(p,q)$
  - Divergence: $D_S(p,q) \equiv S(p,q) - S(p,p)$
  - $D_S(p_\theta, p_{\theta+\delta}) = \frac{1}{2}\delta^T G_S(\theta)\delta + O(\delta^3)$ **Decision geometry**
  - ➡ $\delta_{DGD} = -\,G_S^{-1}(\theta)\;\nabla h(\theta)$
- Recovers information geometry: $S(p,x) = -\ln p(x)$

Game-theory reformulation of VMC

- Natural scoring rule
  - $\forall p_\theta, x,\; S_{VMC}(x, p_\theta) \equiv -\,E_{L,\theta}(x) \rightarrow$ **Proper scoring rule**
- Induced geometry (Divergence)
  - $D_{VMC}(p_\theta, p_{\theta'}) = \frac{1}{8}\sum_i \mathbb{E}\left[\left(\partial_{x_i}\ln p_\theta(X) - \partial_{x_i}\ln p_{\theta'}(X)\right)^2\right]$
  - $G_{VMC}(\theta)_{\theta_1\theta_2} = \frac{1}{4}\sum_i \mathbb{E}\left[\left(\partial_{\theta_1}\partial_{x_i}\ln p_\theta(X)\right)\left(\partial_{\theta_2}\partial_{x_i}\ln p_\theta(X)\right)\right]$
  - ➡ $\delta_{VMC} = -\,G_{VMC}^{-1}(\theta)\;\nabla E(\theta)$
- Physically motivated geometry

$$D_{VMC}(p_\theta, p_{\theta'}) = \mathbb{E}_{X\sim p_\theta}\left[E_{L,\theta}(X) - E_{L,\theta'}(X)\right]$$
$$\simeq E(\theta) - E(\theta') \text{ (up to re-weighting)}$$

Efficient implementation

[Martens, Grosse (2015)]

- KFAC (Kronecker-Factored Approximate Curvature)
  - KFAC $\sim$ crude approximation of the Fisher metric
  - Direction update using KFAC Fisher
  - Scaling update using *exact* Fisher
- ➡ **Fast and reliable convergence**

Open questions

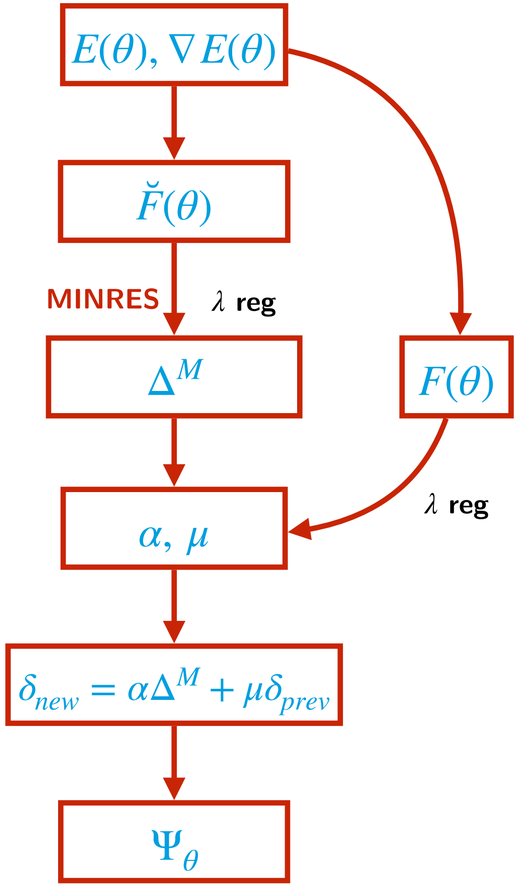
- When is a scoring rule leading to efficient DGD?
- Efficient implementation?
- ...

Practicable optimizer?

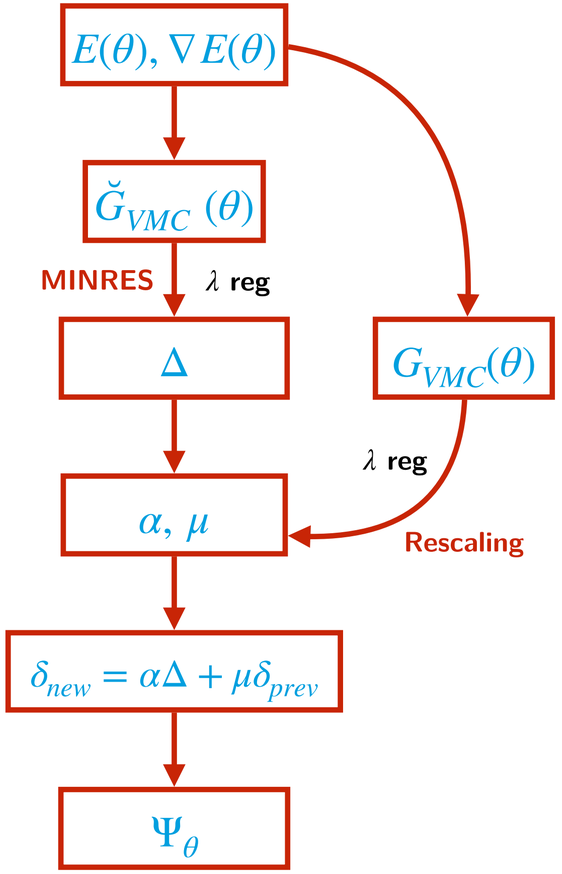
- How good is it strategically? (Convergence in epochs)
- Can it be performant? (Overall wall-time and stability)

Decision vs information geometry

Natural Gradient Descent

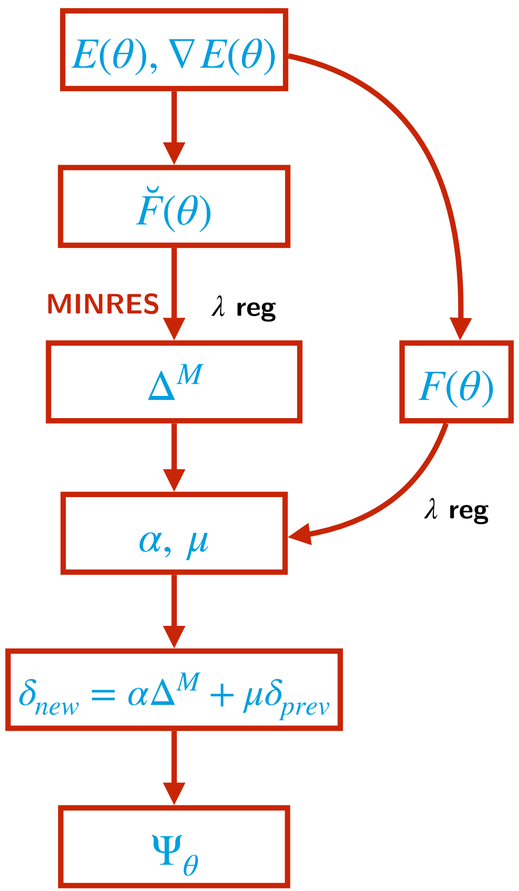


Decisional Gradient Descent

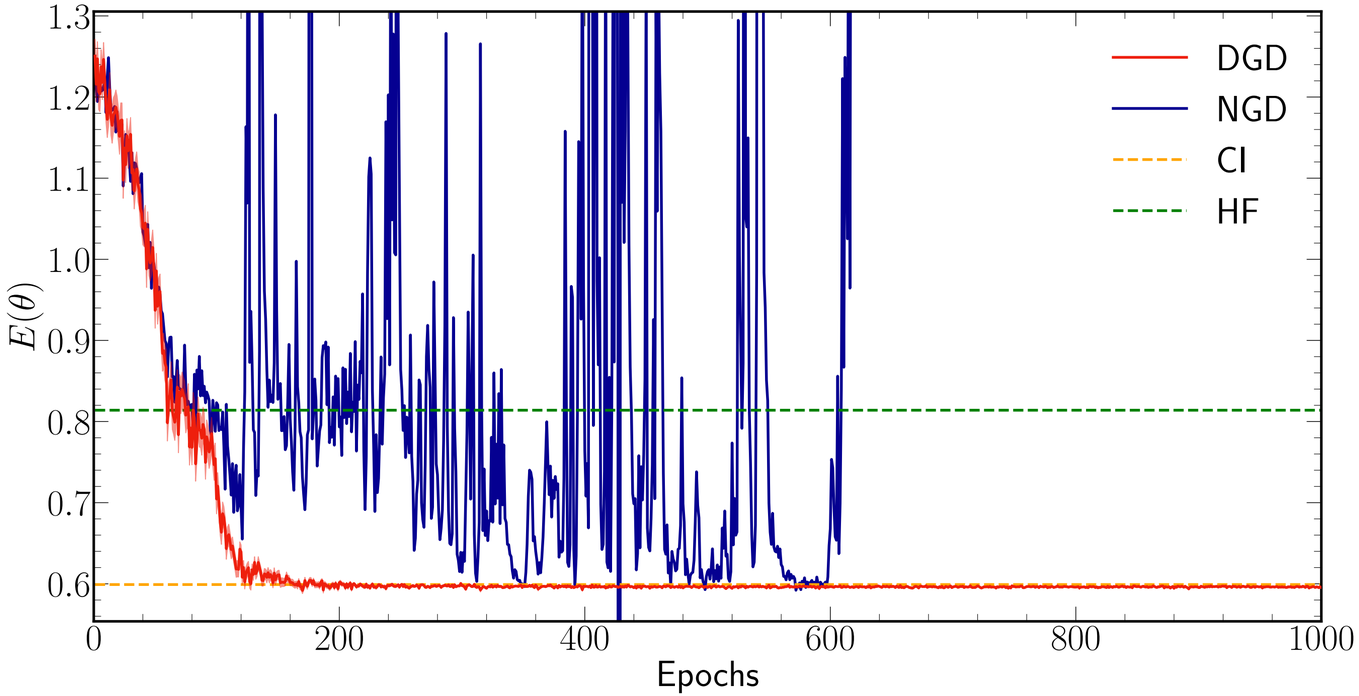


Decision vs information geometry

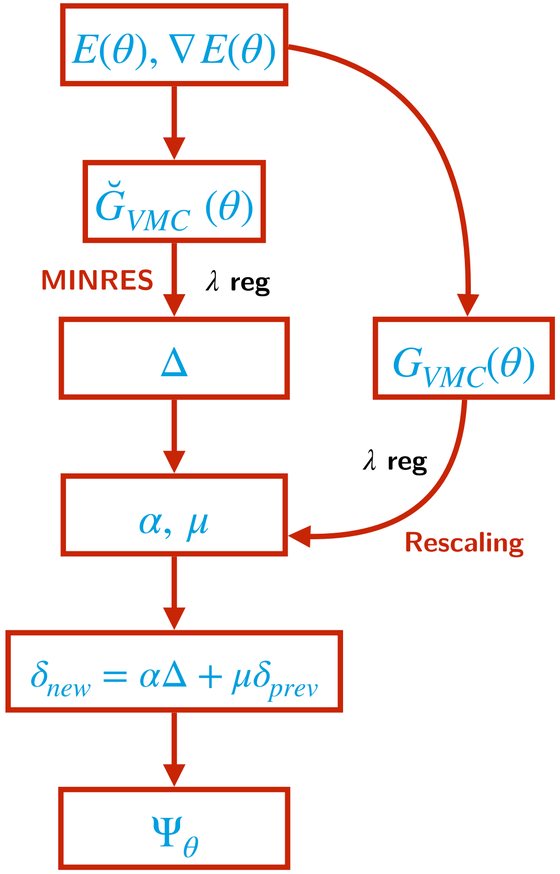
Natural Gradient Descent



NGD vs DGD: $A = 2$, $V_0 = -10$



Decisional Gradient Descent



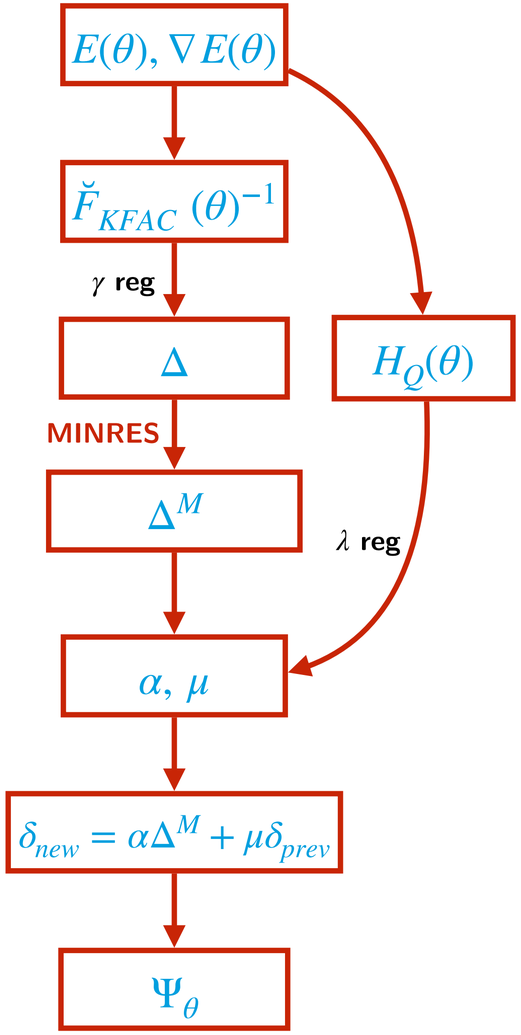
Results

- Stability: huge improvement from decision geometry in all cases

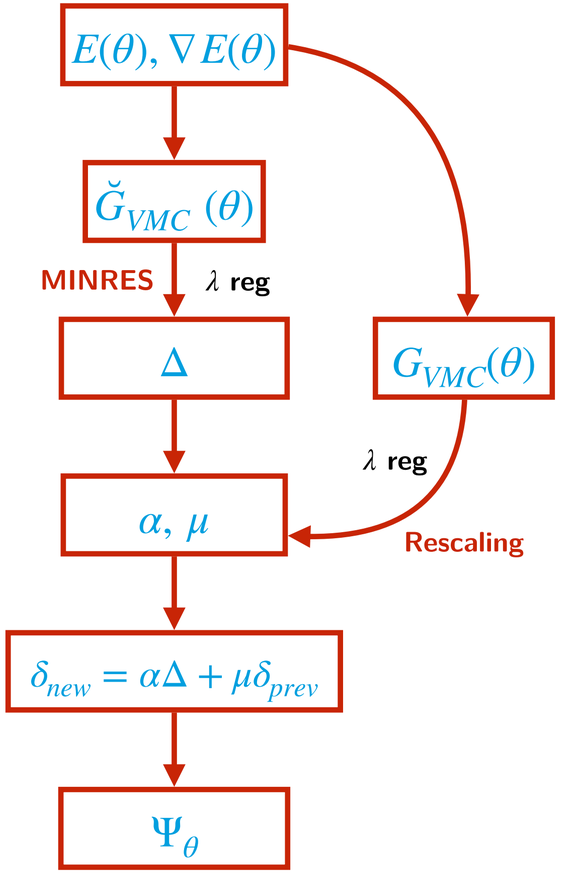
➡ **Much better starting point for designing optimizers for VMC**

Comparing with our previous best optimizer

QN-MR-KFAC

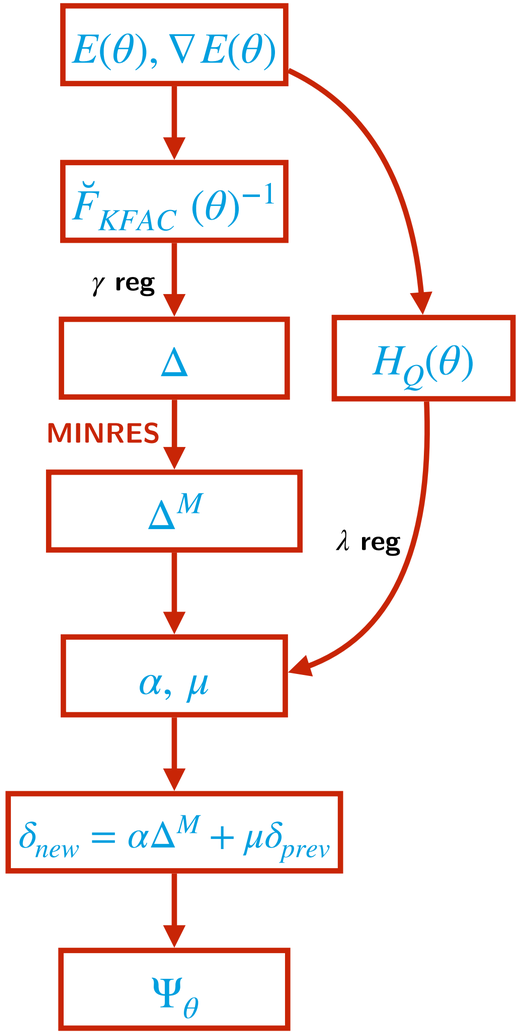


Decisional Gradient Descent

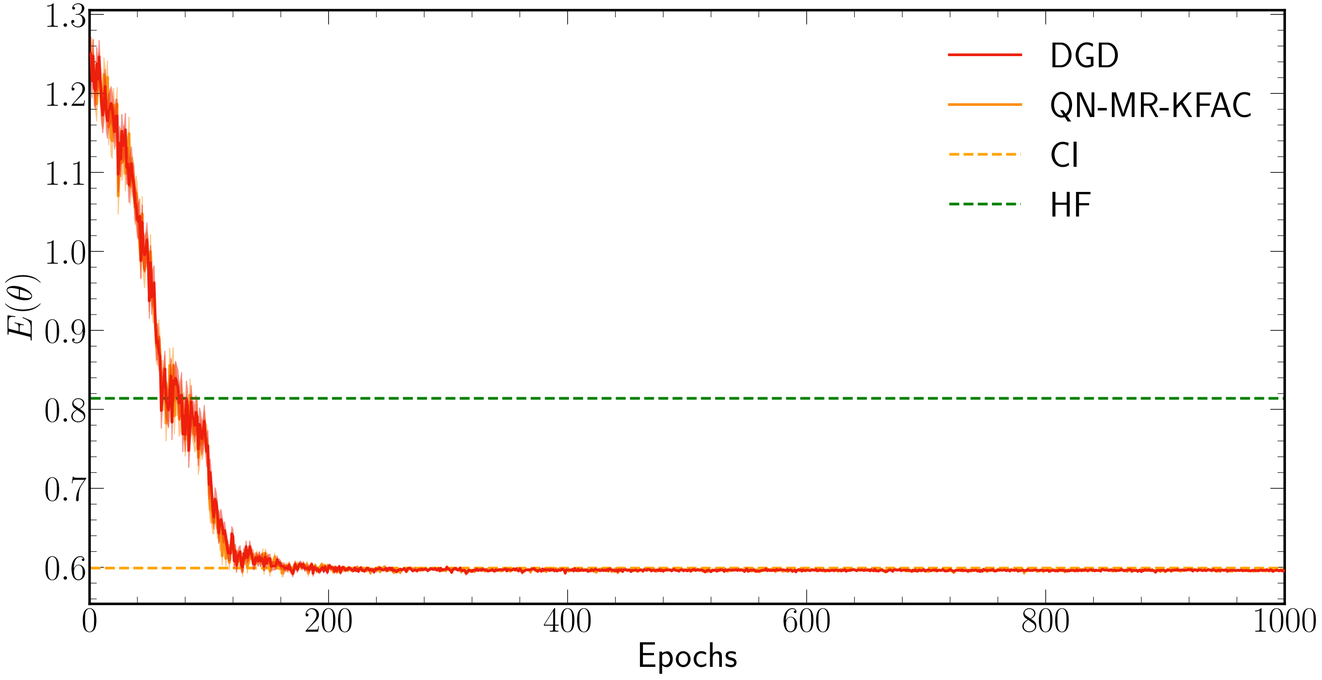


Comparing with our previous best optimizer

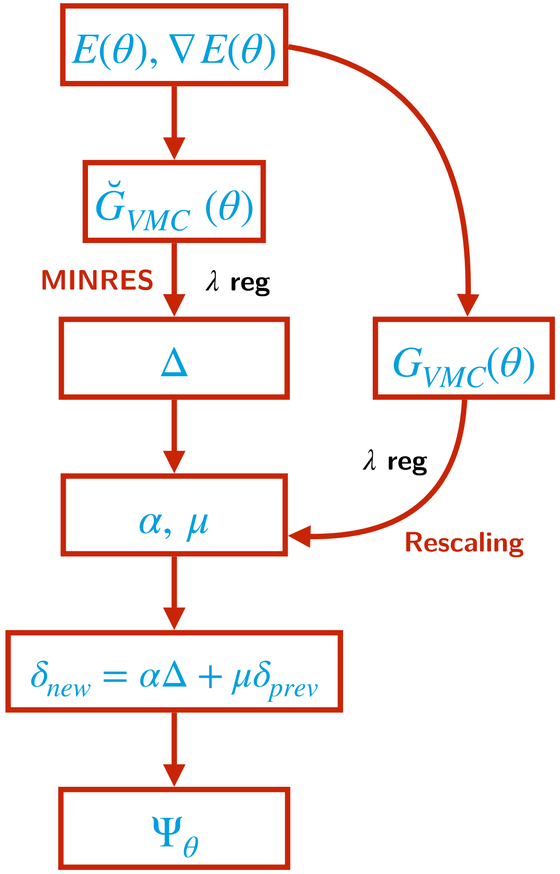
QN-MR-KFAC



QN-MR-KFAC vs DGD: $A = 2$, $V_0 = -10$



Decisional Gradient Descent



Results

- <u>Stability:</u> DGD more stable than any other refinement of KFAC (not shown here)
- <u>Accuracy and speed:</u> DGD on par with QN-MR-KFAC

Comparing with Adam

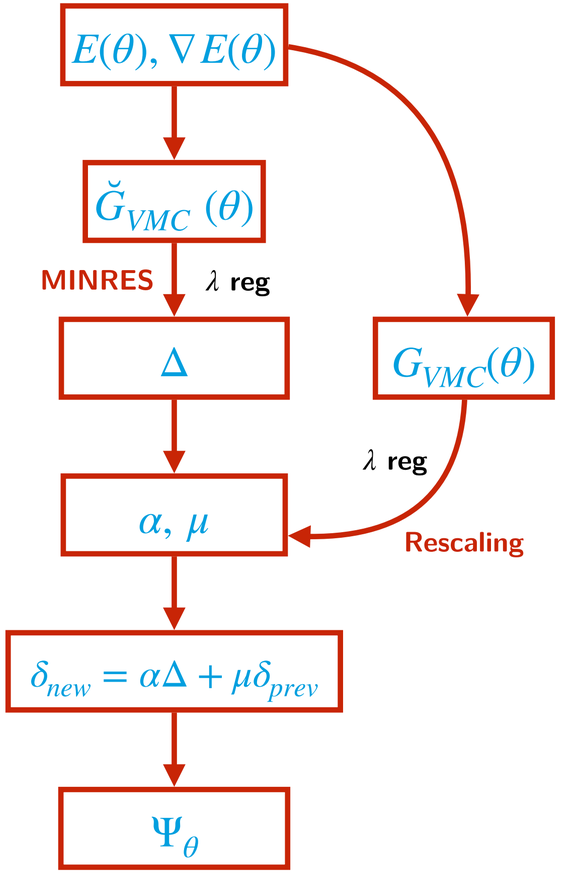
Adam

$$m_n = \frac{\beta_1 \; m_{n-1} + (1-\beta_1) \; \nabla E(\theta_n)}{(1-\beta_1^n)}$$

$$s_n = \frac{\beta_2 \; s_{n-1} + (1-\beta_2) \; (\nabla E(\theta_n))^2}{(1-\beta_2^n)}$$

$$\theta_{n+1} = \theta_n - \alpha \frac{m_n}{\sqrt{s_n} + \epsilon}$$

Decisional Gradient Descent



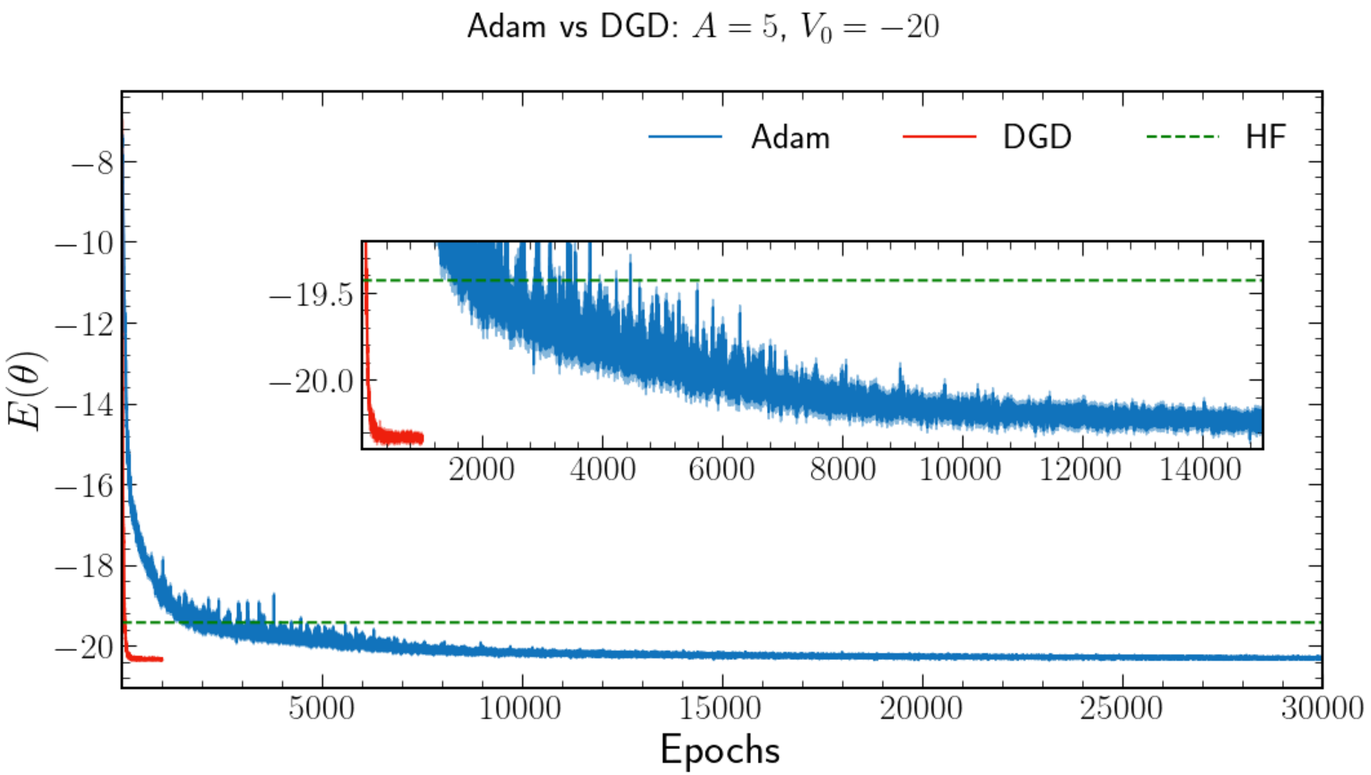
Comparing with Adam

Adam

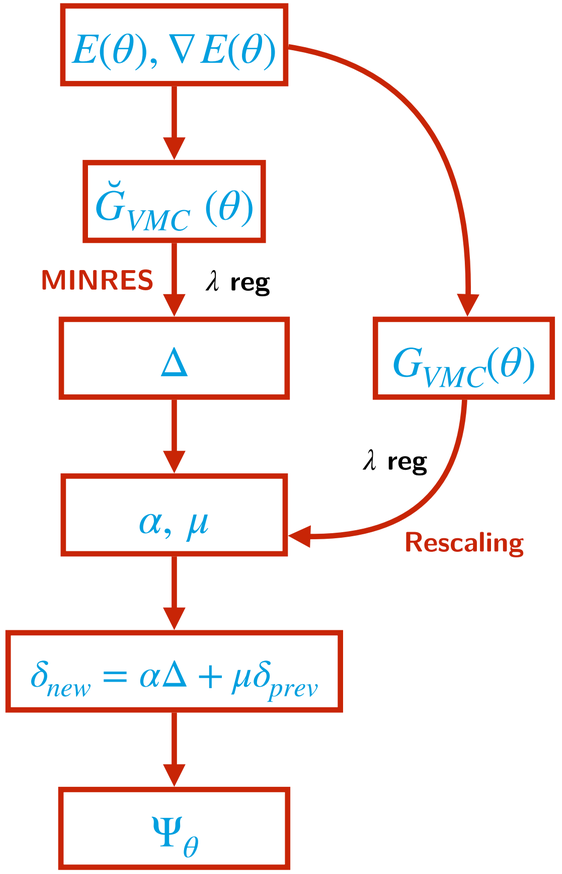
$$m_n = \frac{\beta_1\ m_{n-1} + (1-\beta_1)\ \nabla E(\theta_n)}{(1-\beta_1^n)}$$

$$s_n = \frac{\beta_2\ s_{n-1} + (1-\beta_2)\ (\nabla E(\theta_n))^2}{(1-\beta_2^n)}$$

$$\theta_{n+1} = \theta_n - \alpha \frac{m_n}{\sqrt{s_n} + \epsilon}$$



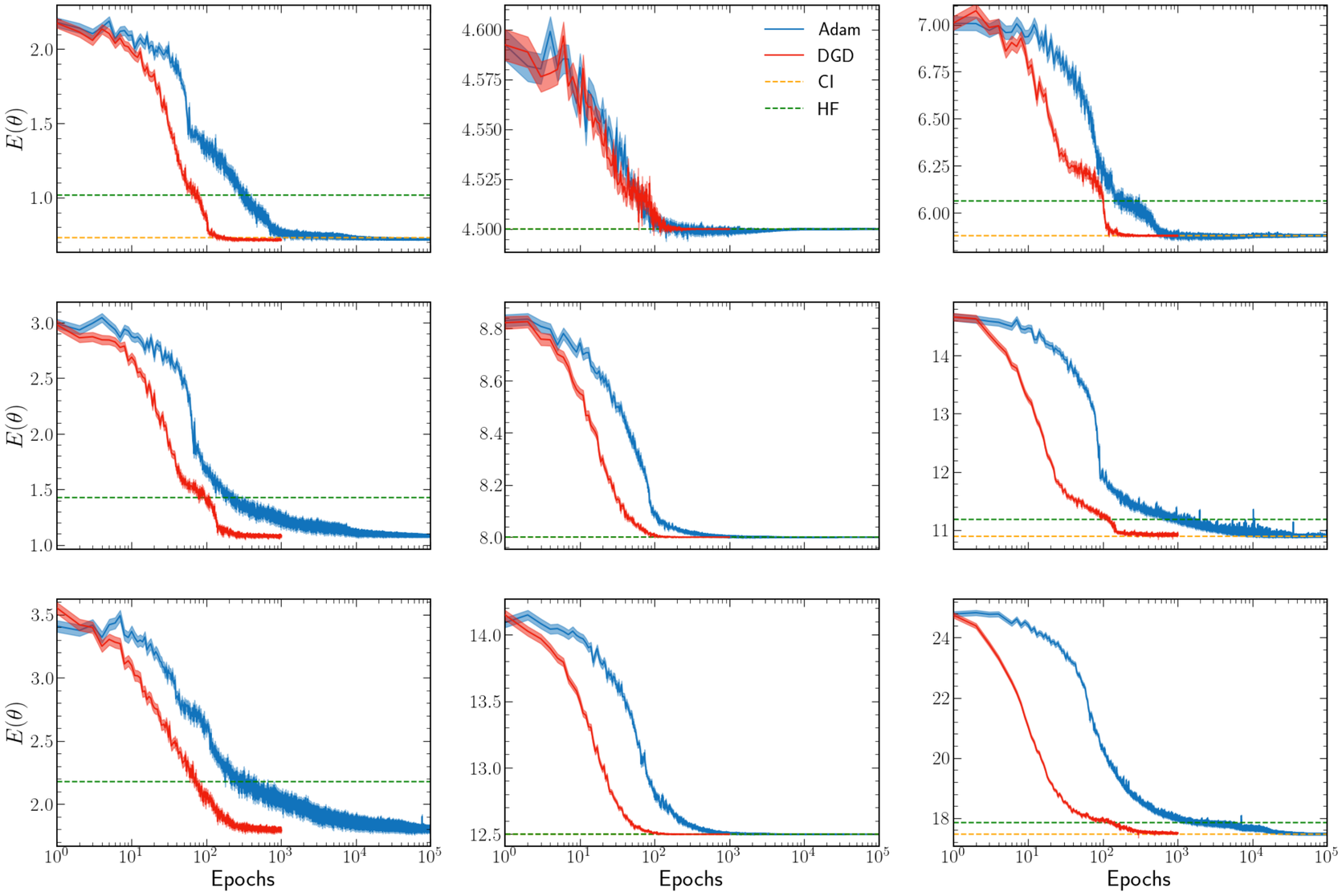
Decisional Gradient Descent



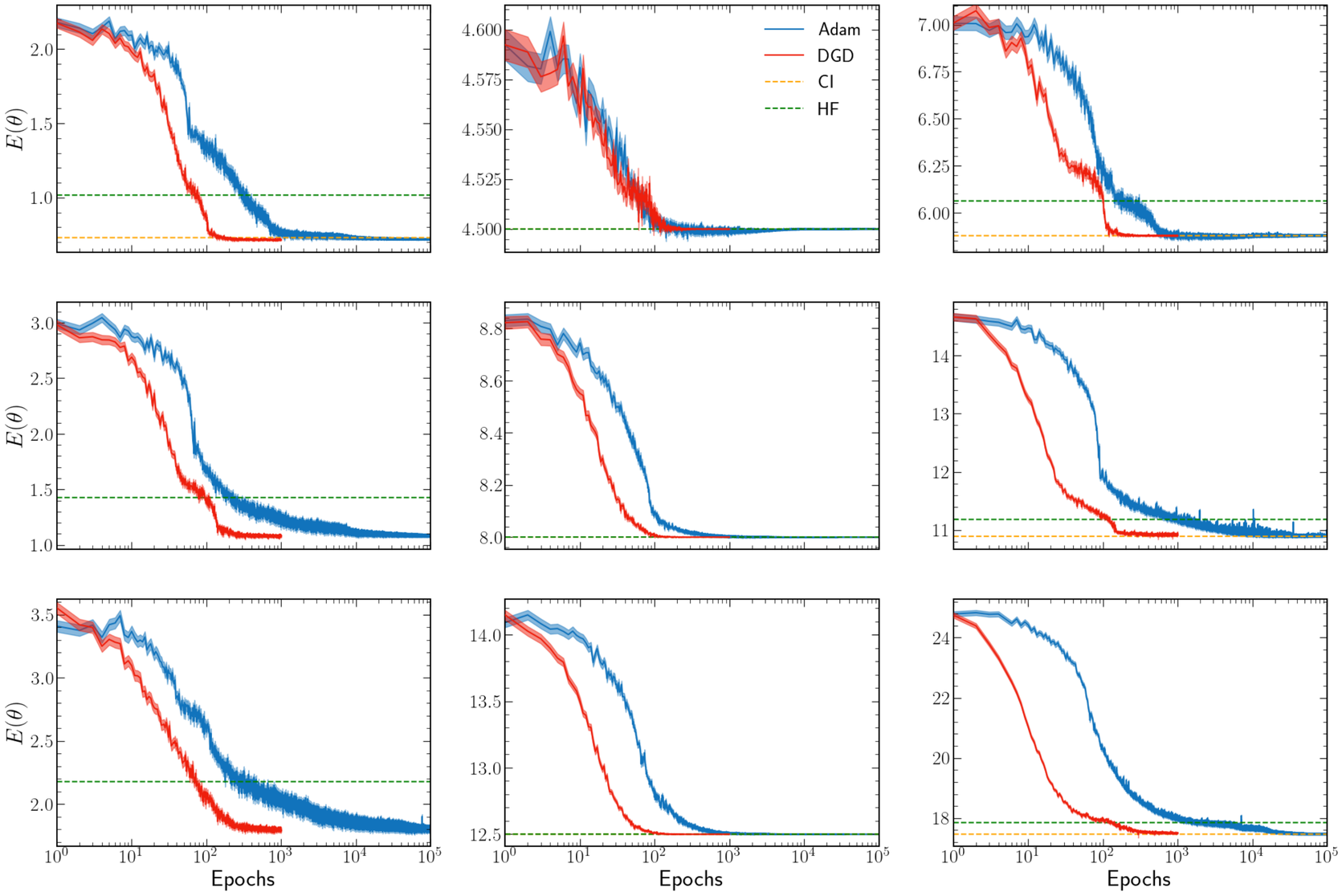
Results

- ✓ Accuracy: DGD systematically over perform Adam
- ✓ Speed: DGD converges 10-100x faster than Adam in #epochs
- ✗ Wall-time: Very naïve implementation of DGD ⇒ too early to quantify

Testing across phenomenologies

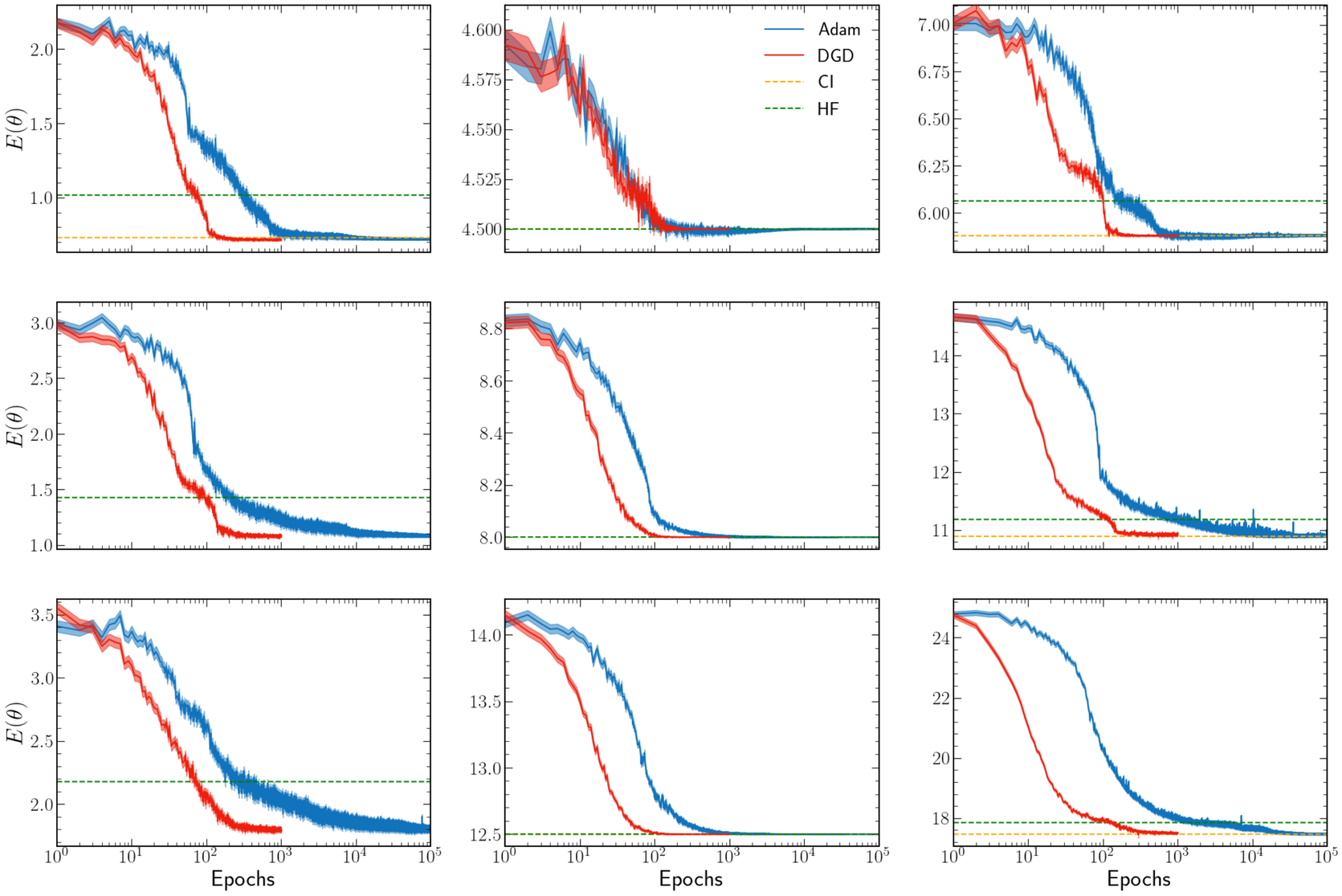


Testing across phenomenologies



Convergence of DGD: 22 out of the 25 cases

Testing across phenomenologies



Convergence of DGD: 22 out of the 25 cases

Confirms the great potential of DGD for future optimizers!

Conclusions

Conclusions

VMC with neural networks

- Rapidly evolving field!
- Competitive with CCSD(T) in quantum chemistry

- **Realistic nuclear systems now being investigated**
  - On-going work to reach $A \sim 100$ nuclei
  - **See Alessandro's talk!**

- **More systematic studies to be performed**
  - Numerical implementation to be optimized
  - Optimal architecture for nuclear systems?
  - Numerical complexity (time/memory)

Conclusions

VMC with neural networks

- Rapidly evolving field!
- Competitive with CCSD(T) in quantum chemistry
- **Realistic nuclear systems now being investigated**
  - On-going work to reach $A \sim 100$ nuclei
  - **See Alessandro's talk!**
- **More systematic studies to be performed**
  - Numerical implementation to be optimized
  - Optimal architecture for nuclear systems?
  - Numerical complexity (time/memory)

The optimizer: a critical part

- Simple many-body systems ⇒ easy to test new ideas
- **A promising novel optimizer based on decision geometry!**
  - Motivated by deficiencies of KFAC for VMC
  - Game theory re-formulation of VMC ⇒ **Decisional gradient descent**
  - Accurate, stable and fast
  - Simplest implementation ⇒ solid foundation for future improvements
- **With many potential refinements!**
  - Hessian-free-like ⇒ Inspiration for many potential algo improvements
  - KFAC-like approximation on decision metric?
  - Adapting the geometry for different many-body problems?
  - Other ML problems? **Can it be made as versatile as Adam?**

TRIUMF

Thank you
Merci

www.triumf.ca

Follow us **@TRIUMFLab**

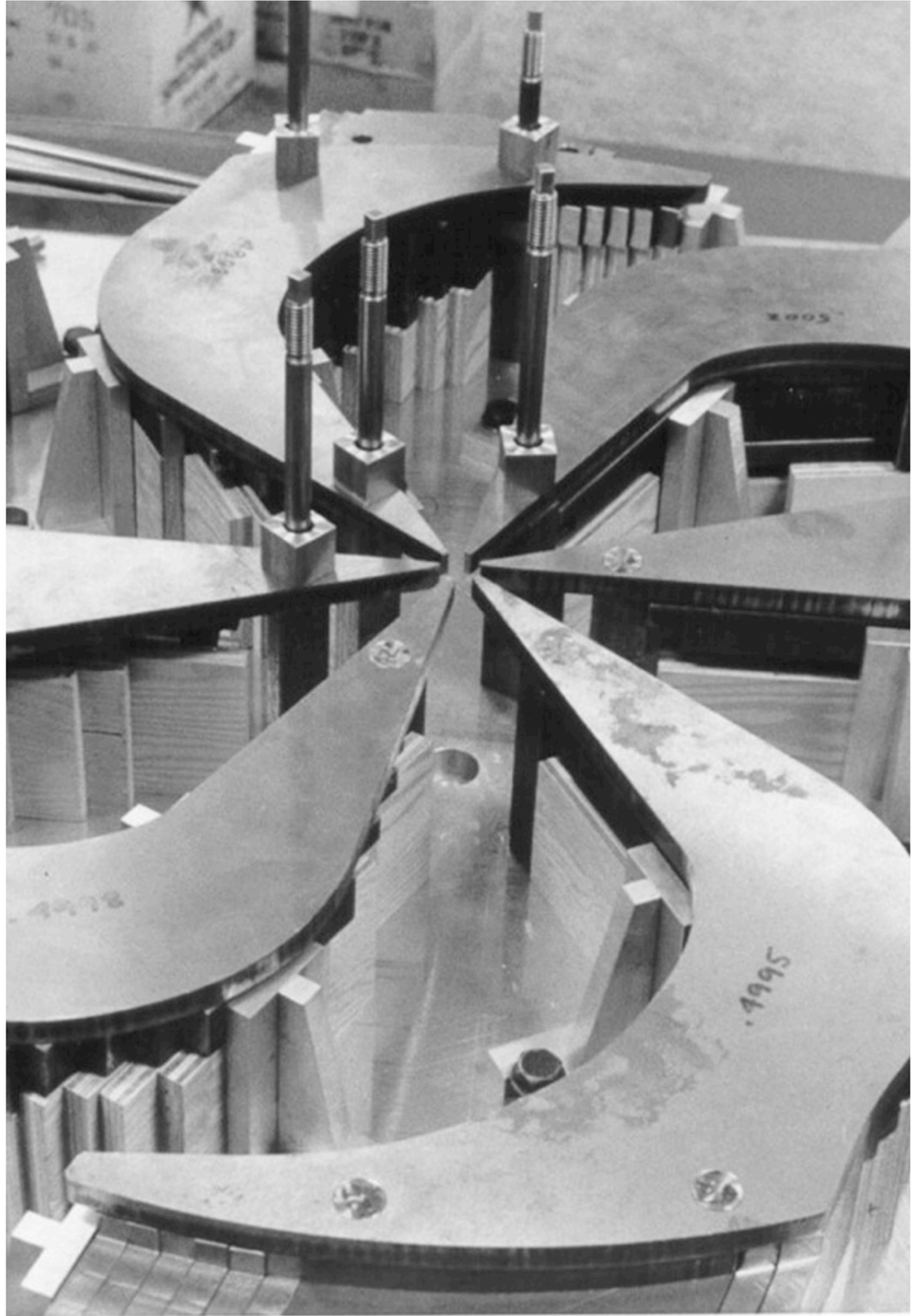
UNIVERSITY OF SURREY

UNIVERSITAT DE BARCELONA

UKRI Science and Technology Facilities Council

GOBIERNO DE ESPAÑA MINISTERIO DE CIENCIA E INNOVACIÓN

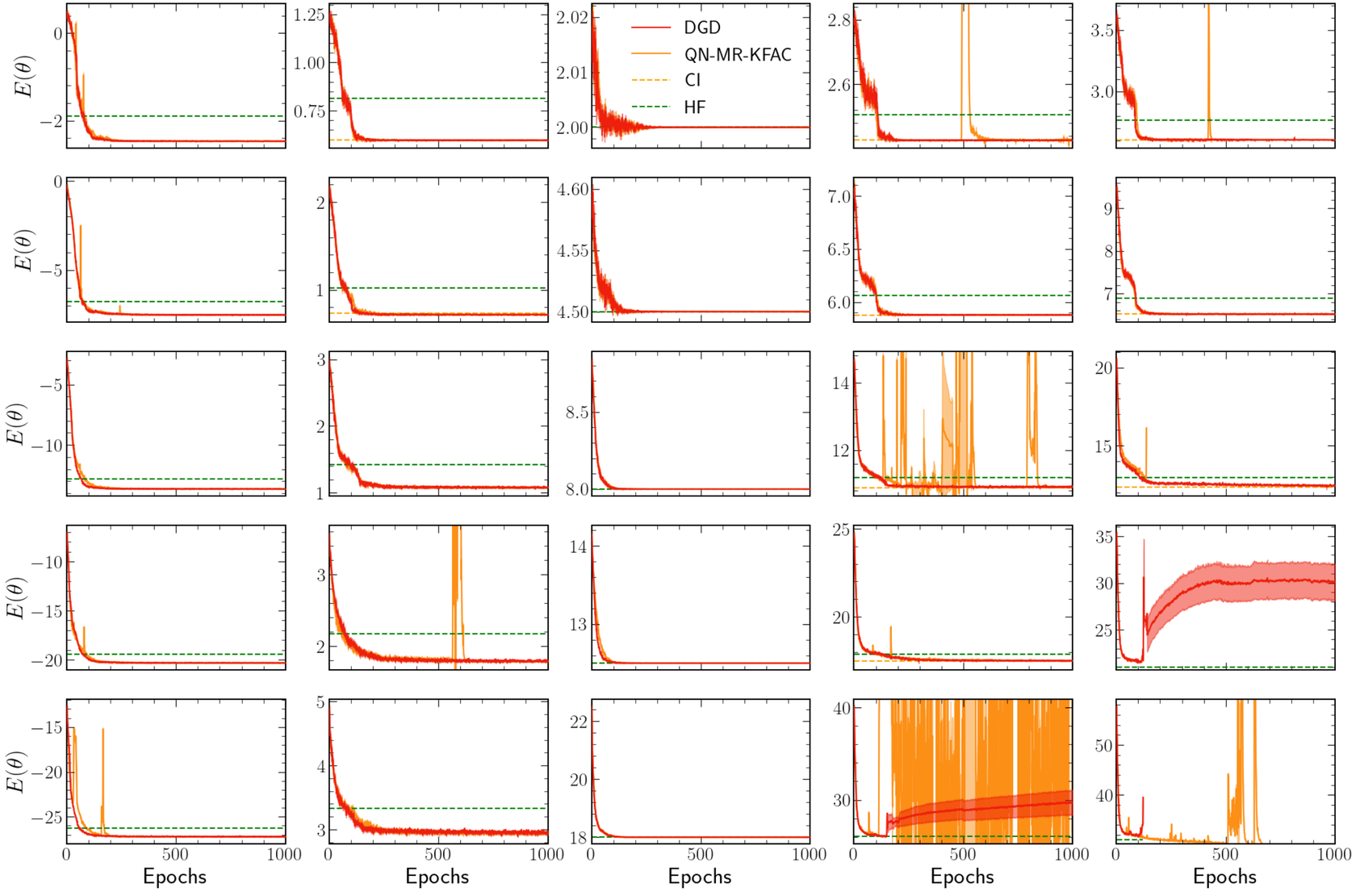
The author(s) gratefully acknowledges the computer resources at Artemisa, funded by the European Union ERDF and Comunitat Valenciana as well as the technical support provided by the Instituto de Física Corpuscular, IFIC (CSIC-UV).



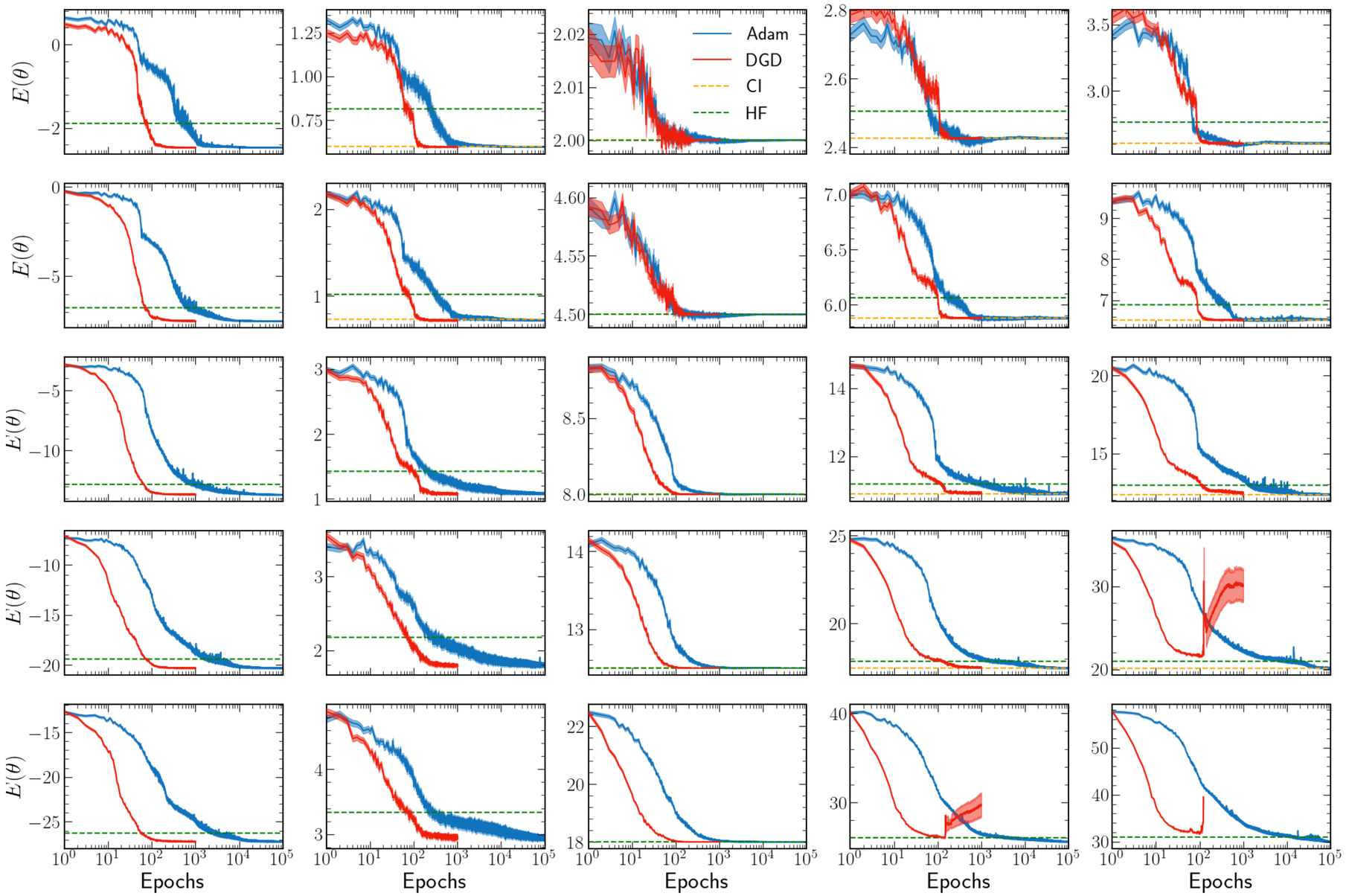
Discovery, accelerated

Back-up slides

Systematic QN-MR-KFAC vs DGD



Systematic Adam vs DGD



Keeping biases under controlled

Testbed for our FNN

- Two different phenomenologies
  - $V_0 \ll -1 \Rightarrow$ Boson-Fermion duality [Girardeau and Olshanii, arXiv:cond-mat/0309396] [Valiente, PRA 103, L021302 (2021)]
  - $V_0 \gg 1 \Rightarrow$ Wigner crystallization
- Benchmarking against other calculations
  - Only for $A = 2$: semi-analytical calculation (= Space) [Busch, Englert, Rzazewski and Wilkens, Found. Phys. 28 549 (1998)]
  - For $A \geq 2$: full CI in HO (= Diag) [Rojo-Francàs, Polls and Juliá-Diaz, Mathematics 8 1196 (2020)]
  - For $A \geq 2$: Hartree-Fock (= HF)

Test result [See James Keeble's talk for more details]

➡ Agreement with exact results

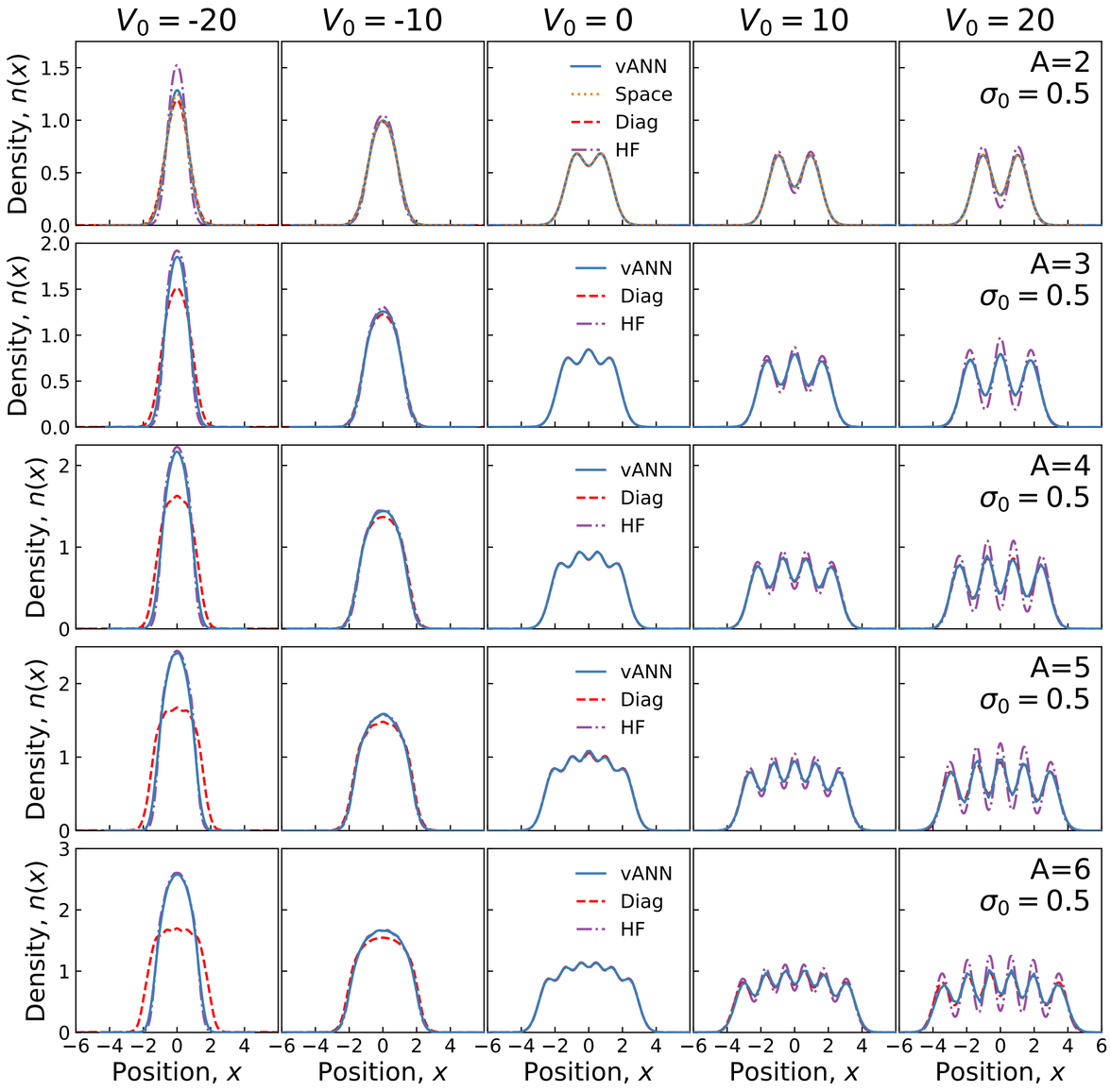
✓ Strong and weak regimes

✓ Repulsive and attractive regimes

➡ Access to different many-body observables

✓ One and two-body densities (not shown here)

➡ **Ansatz biased is under controlled**



Natural gradient descent

Arbitrariness of gradient descent

- Trust regions depends on choice of a norm
- Direction depends on choice of a parametrization
- Is there a "best" choice ?

➡ Natural gradient

Natural gradient descent

Arbitrariness of gradient descent

- Trust regions depends on choice of a norm
- Direction depends on choice of a parametrization
- Is there a "best" choice ?

➡ Natural gradient

Information geometry

[For a review: Amari (2016)]

- Consider the manifold of probability measures (for a fixed σ-algebra of events)
- Chentsov's theorem (1972)
  - *There is a unique Riemannian metric that is invariant under sufficient statistics* ⟵ ~ **lossless re-parametrizations**
  - This is the Fisher information metric
  - $F_{ij}(p) \equiv \mathbb{E}_{X\sim p}\left[\partial_{\theta_i} \ln p(X)\ \partial_{\theta_j} \ln p(X)\right]$
- $F(p) \equiv$ quadratic approximation of Kullback-Leibler divergence

Natural gradient descent

Arbitrariness of gradient descent

- Trust regions depends on choice of a norm
- Direction depends on choice of a parametrization
- Is there a "best" choice ?

➡ Natural gradient

Information geometry

[For a review: Amari (2016)]

- Consider the manifold of probability measures (for a fixed σ-algebra of events)
- Chentsov's theorem (1972)
  - *There is a unique Riemannian metric that is invariant under sufficient statistics* ⟵ ~ **lossless re-parametrizations**
  - This is the Fisher information metric
  - $F_{ij}(p) \equiv \mathbb{E}_{X\sim p}\left[\partial_{\theta_i} \ln p(X)\ \partial_{\theta_j} \ln p(X)\right]$
- $F(p) \equiv$ quadratic approximation of Kullback-Leibler divergence

Natural gradient descent

- Local problem
  - $M_n(\delta) = \nabla E(\theta_n)^T \delta + E(\theta_n)$
  - $T_n(r) = \left\{\delta :\ \delta^T F(\theta_n)\, \delta \leq r^2\right\}$

➡ $\delta_n = -\,\alpha\ F^{-1}(\theta_n)\ \nabla E(\theta_n)$

Natural gradient descent

Arbitrariness of gradient descent

- Trust regions depends on choice of a norm
- Direction depends on choice of a parametrization
- Is there a "best" choice ?

➡ Natural gradient

Information geometry

[For a review: Amari (2016)]

- Consider the manifold of probability measures (for a fixed σ-algebra of events)
- Chentsov's theorem (1972)
  - *There is a unique Riemannian metric that is invariant under sufficient statistics* ← ~ lossless re-parametrizations
  - This is the Fisher information metric
  - $F_{ij}(p) \equiv \mathbb{E}_{X\sim p}\left[\partial_{\theta_i} \ln p(X)\ \partial_{\theta_j} \ln p(X)\right]$
- $F(p) \equiv$ quadratic approximation of Kullback-Leibler divergence

Natural gradient descent

- Local problem
  - $M_n(\delta) = \nabla E(\theta_n)^T \delta + E(\theta_n)$
  - $T_n(r) = \left\{\delta :\ \delta^T F(\theta_n)\, \delta \leq r^2\right\}$

➡ $\delta_n = -\,\alpha\ F^{-1}(\theta_n)\ \nabla E(\theta_n)$ ← Natural gradient definition

Natural gradient descent

Arbitrariness of gradient descent

- Trust regions depends on choice of a norm
- Direction depends on choice of a parametrization
- Is there a "best" choice ?

➡ Natural gradient

Information geometry

[For a review: Amari (2016)]

- Consider the manifold of probability measures (for a fixed σ-algebra of events)
- Chentsov's theorem (1972)
  - *There is a unique Riemannian metric that is invariant under sufficient statistics* ← ~ lossless re-parametrizations
  - This is the Fisher information metric
  - $F_{ij}(p) \equiv \mathbb{E}_{X\sim p}\left[\partial_{\theta_i} \ln p(X)\ \partial_{\theta_j} \ln p(X)\right]$
- $F(p) \equiv$ quadratic approximation of Kullback-Leibler divergence

Natural gradient descent

- Local problem
  - $M_n(\delta) = \nabla E(\theta_n)^T \delta + E(\theta_n)$
  - $T_n(r) = \left\{\delta :\ \delta^T F(\theta_n)\, \delta \leq r^2\right\}$

➡ $\delta_n = -\,\alpha\ F^{-1}(\theta_n)\ \nabla E(\theta_n)$ ← Natural gradient definition

- Still not clear if the "best" but
  - Independent from parametrization
  - $F(p) \succeq 0 \ \Rightarrow$ Bounded/Stable updates
  - Argued to be close to a 2nd order optimizer [Martens, 2020]

➡ Motivates the use of $\alpha = 1$

Natural gradient descent

Arbitrariness of gradient descent

- Trust regions depends on choice of a norm
- Direction depends on choice of a parametrization
- Is there a "best" choice ?

➡ Natural gradient

Information geometry

[For a review: Amari (2016)]

- Consider the manifold of probability measures (for a fixed σ-algebra of events)
- Chentsov's theorem (1972)
  - *There is a unique Riemannian metric that is invariant under sufficient statistics* ← ~ lossless re-parametrizations
  - This is the Fisher information metric
  - $F_{ij}(p) \equiv \mathbb{E}_{X\sim p}\left[\partial_{\theta_i} \ln p(X)\ \partial_{\theta_j} \ln p(X)\right]$
- $F(p) \equiv$ quadratic approximation of Kullback-Leibler divergence

Natural gradient descent

- Local problem
  - $M_n(\delta) = \nabla E(\theta_n)^T \delta + E(\theta_n)$
  - $T_n(r) = \left\{\delta :\ \delta^T F(\theta_n)\, \delta \leq r^2\right\}$

Natural gradient definition

➡ $\delta_n = -\,\alpha\ F^{-1}(\theta_n)\ \nabla E(\theta_n)$

- Still not clear if the "best" but
  - Independent from parametrization
  - $F(p) \succeq 0 \ \Rightarrow$ Bounded/Stable updates
  - Argued to be close to a 2nd order optimizer [Martens, 2020]

Only for supervised learning

➡ Motivates the use of $\alpha = 1$

Natural gradient descent

Arbitrariness of gradient descent

- Trust regions depends on choice of a norm
- Direction depends on choice of a parametrization
- Is there a "best" choice ?

➡ Natural gradient

Information geometry

[For a review: Amari (2016)]

- Consider the manifold of probability measures (for a fixed σ-algebra of events)
- Chentsov's theorem (1972)
  - *There is a unique Riemannian metric that is invariant under sufficient statistics* ← ~ lossless re-parametrizations
  - This is the Fisher information metric
  - $F_{ij}(p) \equiv \mathbb{E}_{X\sim p}\left[\partial_{\theta_i} \ln p(X)\ \partial_{\theta_j} \ln p(X)\right]$
- $F(p) \equiv$ quadratic approximation of Kullback-Leibler divergence

Natural gradient descent

- Local problem
  - $M_n(\delta) = \nabla E(\theta_n)^T \delta + E(\theta_n)$
  - $T_n(r) = \left\{\delta :\ \delta^T F(\theta_n)\, \delta \leq r^2\right\}$

Natural gradient definition

➡ $\delta_n = -\,\alpha\ F^{-1}(\theta_n)\ \nabla E(\theta_n)$

- Still not clear if the "best" but
  - Independent from parametrization

Only for supervised learning

  - $F(p) \succeq 0 \ \Rightarrow$ Bounded/Stable updates
  - Argued to be close to a 2nd order optimizer [Martens, 2020]

➡ Motivates the use of $\alpha = 1$

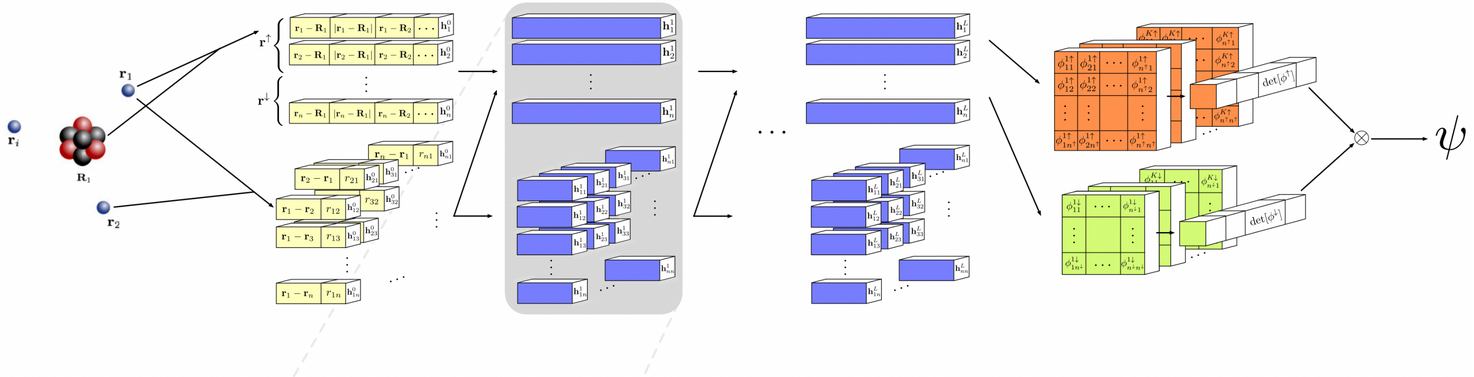
- What about VMC problems ? ($\neq$ supervised learning)
  - Stochastic reconfiguration method empirically efficient [Park et al. (2020)]

➡ Equivalent to natural gradient descent with $p = |\Psi_\theta|^2$

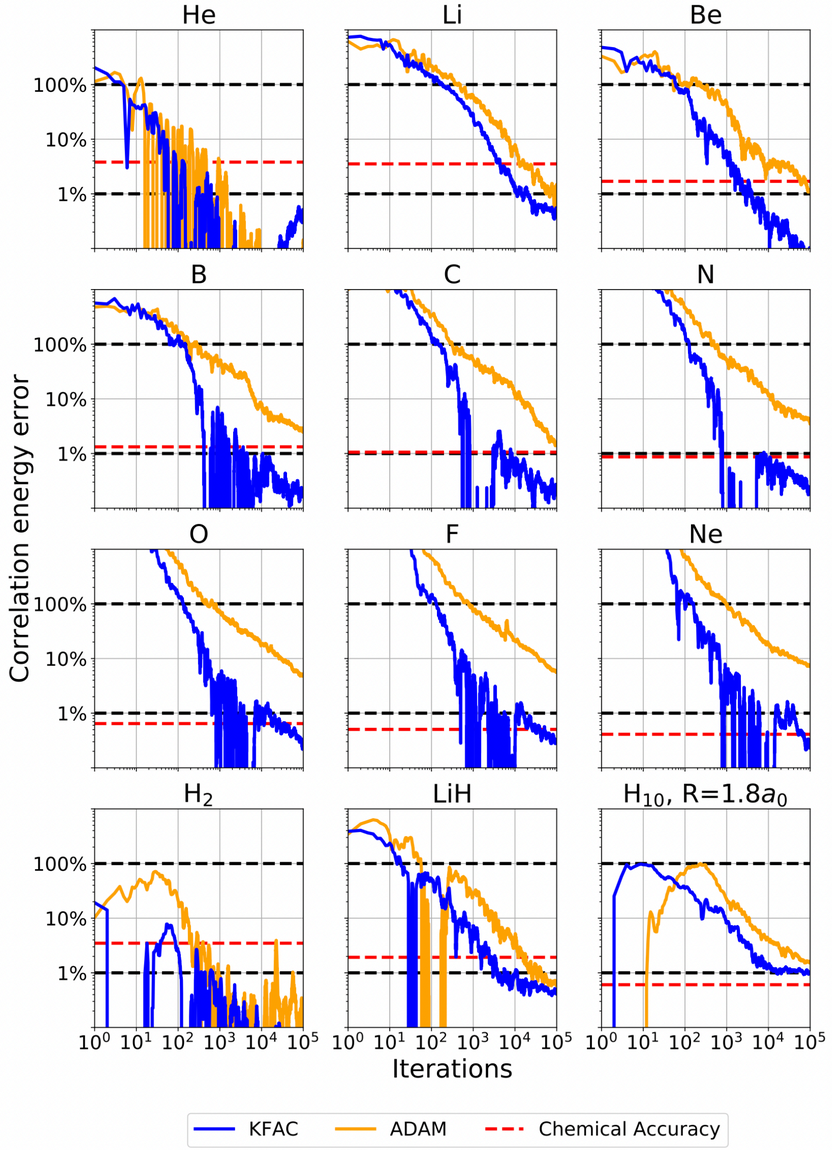
KFAC: a promising new optimizer

State-of-the-art

- Two main optimizers for Neural Quantum States:
  - Adam → Stable but slow ($\sim 10\,000$ epochs to get a good accuracy)
  - KFAC → Fast and more accurate on paper!
- Known problems with KFAC: [Pfau, Spencer, Matthews and Foulkes, Phys Rev Res **2**, 033429 (2020)]
  - Fine-tuning hyperparameters is critical
  - Oscillates between slow and unstable convergence
- From our own practice:
  - KFAC comes with **several adjustments** to compensate its instabilities
  - Case by case fine-tuning is required to get decent results **with no guarantee**



[Pfau, Spencer, Matthews and Foulkes, Phys Rev Res **2**, 033429 (2020)]



Kronecker-Factored approximation

Optimizing neural networks

- Number of parameters $N \sim 10\ 000$
- Numerical complexity per epoch critical
- Exact natural gradient descent $\Rightarrow O(N^3)$
- KFAC goal [Martens and Grosse (2015)]
  - Lower complexity as much as possible
  - While keeping number of epochs to converge $\sim$ constant

Kronecker-Factored approximation

Optimizing neural networks

- Number of parameters $N \sim 10\ 000$
- Numerical complexity per epoch critical
- Exact natural gradient descent $\Rightarrow O(N^3)$
- KFAC goal [Martens and Grosse (2015)]
  - Lower complexity as much as possible
  - While keeping number of epochs to converge $\sim$ constant

Kronecker factorization rationale

[Martens and Grosse (2015)]

- $F_{ij}(\theta) = 4 \times \mathbb{E}_{X \sim |\Psi_\theta|^2} \left[\partial_{\theta_i} \ln |\Psi_\theta(X)| \ \partial_{\theta_j} \ln |\Psi_\theta(X)| \right]$
- Can be reformulated
  - Chain rule $\Rightarrow \dfrac{\partial \ln |\Psi_\theta(X)|}{\partial W^l_{ij}} = (a_{l-1} \otimes e_l)_{ij} \quad \Rightarrow \quad F = 4 \times \mathbb{E}\left[(a \otimes e)\ (a \otimes e)^T\right]$
  - So $F = 4 \times \mathbb{E}\left[(aa^T) \otimes (ee^T)\right]$

➡ $F_{KFAC} \simeq 4 \times \mathbb{E}\left[aa^T\right] \otimes \mathbb{E}\left[ee^T\right]$

- Further approximation: block-diagonal between layers

➡ $\breve{F}^l_{KFAC} \simeq 4 \times \mathbb{E}\left[a_{l-1}a_{l-1}^T\right] \otimes \mathbb{E}\left[e_l e_l^T\right]$ **Much easier to inverse !**

Kronecker-Factored approximation

Optimizing neural networks

- Number of parameters $N \sim 10\ 000$
- Numerical complexity per epoch critical
- Exact natural gradient descent $\Rightarrow O(N^3)$
- KFAC goal [Martens and Grosse (2015)]
  - Lower complexity as much as possible
  - While keeping number of epochs to converge $\sim$ constant

Kronecker factorization rationale

[Martens and Grosse (2015)]

- $F_{ij}(\theta) = 4 \times \mathbb{E}_{X \sim |\Psi_\theta|^2} \left[\partial_{\theta_i} \ln |\Psi_\theta(X)| \ \partial_{\theta_j} \ln |\Psi_\theta(X)| \right]$
- Can be reformulated
  - Chain rule $\Rightarrow \dfrac{\partial \ln |\Psi_\theta(X)|}{\partial W^l_{ij}} = (a_{l-1} \otimes e_l)_{ij} \quad \Rightarrow \quad F = 4 \times \mathbb{E}\left[(a \otimes e)\ (a \otimes e)^T\right]$

    Backward sensitivities

    Activities
  - So $F = 4 \times \mathbb{E}\left[(aa^T) \otimes (ee^T)\right]$

➡ $F_{KFAC} \simeq 4 \times \mathbb{E}\left[aa^T\right] \otimes \mathbb{E}\left[ee^T\right]$

- Further approximation: block-diagonal between layers

➡ $\breve{F}^l_{KFAC} \simeq 4 \times \mathbb{E}\left[a_{l-1}a_{l-1}^T\right] \otimes \mathbb{E}\left[e_l e_l^T\right]$ **Much easier to inverse !**

Kronecker-Factored approximation

Optimizing neural networks

- Number of parameters $N \sim 10\ 000$
- Numerical complexity per epoch critical
- Exact natural gradient descent $\Rightarrow O(N^3)$
- KFAC goal [Martens and Grosse (2015)]
  - Lower complexity as much as possible
  - While keeping number of epochs to converge $\sim$ constant

Kronecker factorization rationale

[Martens and Grosse (2015)]

- $F_{ij}(\theta) = 4 \times \mathbb{E}_{X \sim |\Psi_\theta|^2} \left[\partial_{\theta_i} \ln |\Psi_\theta(X)| \ \partial_{\theta_j} \ln |\Psi_\theta(X)| \right]$
- Can be reformulated
  - Chain rule $\Rightarrow \dfrac{\partial \ln |\Psi_\theta(X)|}{\partial W^l_{ij}} = (a_{l-1} \otimes e_l)_{ij} \Rightarrow F = 4 \times \mathbb{E}\left[(a \otimes e)\ (a \otimes e)^T\right]$

    Backward sensitivities

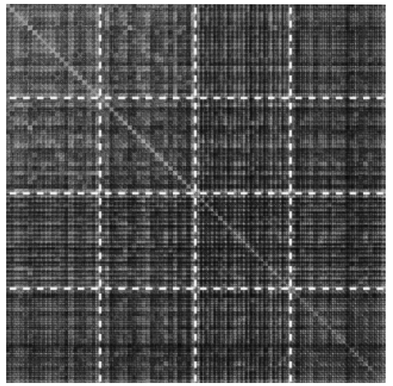
    Activities
  - So $F = 4 \times \mathbb{E}\left[(aa^T) \otimes (ee^T)\right]$

➡ $F_{KFAC} \simeq 4 \times \mathbb{E}\left[aa^T\right] \otimes \mathbb{E}\left[ee^T\right]$

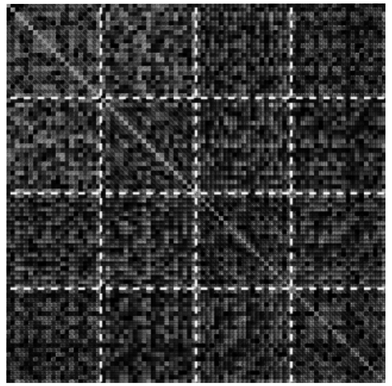
- Further approximation: block-diagonal between layers

➡ $\breve{F}^l_{KFAC} \simeq 4 \times \mathbb{E}\left[a_{l-1}a_{l-1}^T\right] \otimes \mathbb{E}\left[e_l e_l^T\right]$ **Much easier to inverse !**

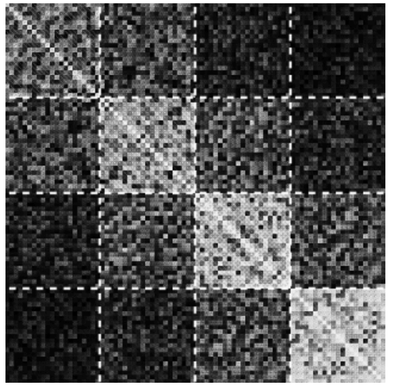
Example on MNIST database [Martens, Grosse (2015)]



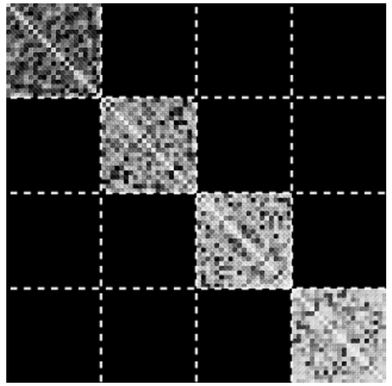
Exact Fisher: F



KFAC Fisher: F_{KFAC}



Inverse KFAC Fisher: $(F_{KFAC})^{-1}$



Inverse block KFAC Fisher: $(\breve{F}_{KFAC})^{-1}$

Kronecker-Factored approximation

Optimizing neural networks

- Number of parameters $N \sim 10\ 000$
- Numerical complexity per epoch critical
- Exact natural gradient descent $\Rightarrow O(N^3)$
- KFAC goal [Martens and Grosse (2015)]
  - Lower complexity as much as possible
  - While keeping number of epochs to converge $\sim$ constant

Kronecker factorization rationale

[Martens and Grosse (2015)]

- $F_{ij}(\theta) = 4 \times \mathbb{E}_{X \sim |\Psi_\theta|^2} \left[\partial_{\theta_i} \ln |\Psi_\theta(X)| \ \partial_{\theta_j} \ln |\Psi_\theta(X)| \right]$
- Can be reformulated
  - Chain rule $\Rightarrow \dfrac{\partial \ln |\Psi_\theta(X)|}{\partial W^l_{ij}} = (a_{l-1} \otimes e_l)_{ij} \Rightarrow F = 4 \times \mathbb{E}\left[(a \otimes e)\ (a \otimes e)^T\right]$
  - So $F = 4 \times \mathbb{E}\left[(aa^T) \otimes (ee^T)\right]$

Backward sensitivities

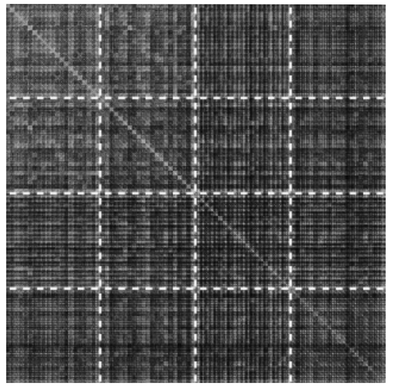
Activities

➡ $F_{KFAC} \simeq 4 \times \mathbb{E}\left[aa^T\right] \otimes \mathbb{E}\left[ee^T\right]$

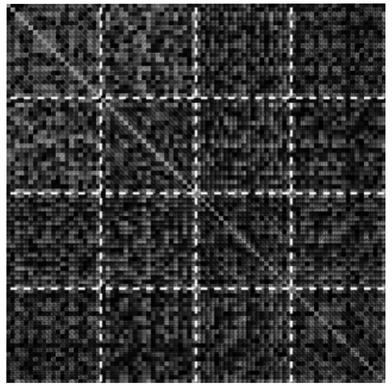
- Further approximation: block-diagonal between layers

➡ $\check{F}^l_{KFAC} \simeq 4 \times \mathbb{E}\left[a_{l-1}a^T_{l-1}\right] \otimes \mathbb{E}\left[e_l e^T_l\right]$ **Much easier to inverse !**

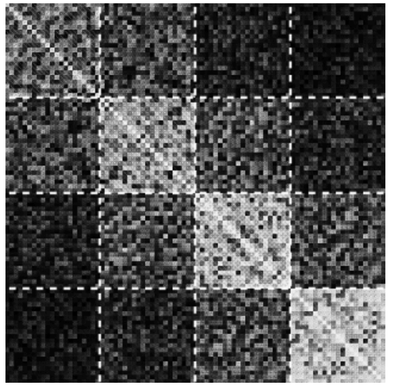
Example on MNIST database [Martens, Grosse (2015)]



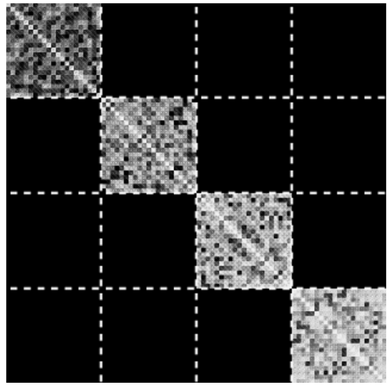
Exact Fisher: F



KFAC Fisher: F_{KFAC}



Inverse KFAC Fisher: $(F_{KFAC})^{-1}$



Inverse block KFAC Fisher: $(\check{F}_{KFAC})^{-1}$

Crude but necessary approximation !

Mitigating the crudeness of KFAC

General strategy

- <u>KFAC update:</u> $\delta_n = \alpha \times \Delta_n$ with $\Delta_n \equiv -(\breve{F}_{KFAC}(\theta_n))^{-1} \nabla E(\theta_n)$
  - When Δ_n is a good direction: take α large
  - When Δ_n is a bad direction: take α small
- How to quantify it ?

Mitigating the crudeness of KFAC

General strategy

- KFAC update: $\delta_n = \alpha \times \Delta_n$ with $\Delta_n \equiv -(\check{F}_{KFAC}(\theta_n))^{-1} \nabla E(\theta_n)$
  - When Δ_n is a good direction: take α large
  - When Δ_n is a bad direction: take α small
- How to quantify it ?

Update evaluation

- Define a local quadratic model: use the exact Fisher $F(\theta_n)$
  - $M_n(\delta) \equiv \frac{1}{2}\delta^T F(\theta_n)\delta + \nabla E(\theta_n)^T \delta + E(\theta_n)$
  - Minimize $M_n(\alpha\ \Delta_n)$ over α
- Momentum extension: find minimum for $\delta = \alpha\ \Delta_n + \mu\ \delta_{n-1}$
  - Analytical solution

$$\begin{pmatrix} \alpha \\ \mu \end{pmatrix} = -\begin{pmatrix} \Delta_n^T F(\theta_n)\Delta_n & \Delta_n^T F(\theta_n)\delta_{n-1} \\ \Delta_n^T F(\theta_n)\delta_{n-1} & \delta_{n-1}^T F(\theta_n)\delta_{n-1} \end{pmatrix}^{-1} \begin{pmatrix} \nabla E(\theta_n)^T\ \Delta_n \\ \nabla E(\theta_n)^T\ \delta_{n-1} \end{pmatrix}$$

Mitigating the crudeness of KFAC

General strategy

- KFAC update: $\delta_n = \alpha \times \Delta_n$ with $\Delta_n \equiv -(\check{F}_{KFAC}(\theta_n))^{-1} \nabla E(\theta_n)$
  - When Δ_n is a good direction: take α large
  - When Δ_n is a bad direction: take α small
- How to quantify it ?

Update evaluation

- Define a local quadratic model: use the exact Fisher $F(\theta_n)$
  - $M_n(\delta) \equiv \frac{1}{2}\delta^T F(\theta_n)\delta + \nabla E(\theta_n)^T \delta + E(\theta_n)$
  - Minimize $M_n(\alpha\ \Delta_n)$ over α
- Momentum extension: find minimum for $\delta = \alpha\ \Delta_n + \mu\ \delta_{n-1}$
  - Analytical solution

$$\begin{pmatrix} \alpha \\ \mu \end{pmatrix} = -\begin{pmatrix} \Delta_n^T F(\theta_n)\Delta_n & \Delta_n^T F(\theta_n)\delta_{n-1} \\ \Delta_n^T F(\theta_n)\delta_{n-1} & \delta_{n-1}^T F(\theta_n)\delta_{n-1} \end{pmatrix}^{-1} \begin{pmatrix} \nabla E(\theta_n)^T\ \Delta_n \\ \nabla E(\theta_n)^T\ \delta_{n-1} \end{pmatrix}$$

Regularization

- Trust regions are all the more important !
- Direction/scale split $\Rightarrow$ use different trust regions for each

Mitigating the crudeness of KFAC

General strategy

- KFAC update: $\delta_n = \alpha \times \Delta_n$ with $\Delta_n \equiv -(\check{F}_{KFAC}(\theta_n))^{-1} \nabla E(\theta_n)$
  - When Δ_n is a good direction: take α large
  - When Δ_n is a bad direction: take α small
- How to quantify it ?

Update evaluation

- Define a local quadratic model: use the exact Fisher $F(\theta_n)$
  - $M_n(\delta) \equiv \frac{1}{2}\delta^T F(\theta_n)\delta + \nabla E(\theta_n)^T \delta + E(\theta_n)$
  - Minimize $M_n(\alpha\ \Delta_n)$ over α
- Momentum extension: find minimum for $\delta = \alpha\ \Delta_n + \mu\ \delta_{n-1}$
  - Analytical solution

$$\begin{pmatrix} \alpha \\ \mu \end{pmatrix} = -\begin{pmatrix} \Delta_n^T F(\theta_n)\Delta_n & \Delta_n^T F(\theta_n)\delta_{n-1} \\ \Delta_n^T F(\theta_n)\delta_{n-1} & \delta_{n-1}^T F(\theta_n)\delta_{n-1} \end{pmatrix}^{-1} \begin{pmatrix} \nabla E(\theta_n)^T\ \Delta_n \\ \nabla E(\theta_n)^T\ \delta_{n-1} \end{pmatrix}$$

Regularization

- Trust regions are all the more important !
- Direction/scale split $\Rightarrow$ use different trust regions for each
- Directional trust region
  - Tikhonov regularization: $\check{F}^{l\ Reg}_{KFAC} \simeq 4 \times \left(\mathbb{E}\left[a_{l-1}a_{l-1}^T\right] + \gamma\pi_l\ \mathrm{Id}\right) \otimes \left(\mathbb{E}\left[e_l e_l^T\right] + \frac{\gamma}{\pi_l}\ \mathrm{Id}\right)$
  - Regularizes calculation of inverses as well
  - π_l automatically chosen to minimize cross-term
  - γ adapted with greedy algorithm: take best of $\left(\frac{\gamma}{\omega_2}, \gamma, \omega_2\ \gamma\right)$

Mitigating the crudeness of KFAC

General strategy

- KFAC update: $\delta_n = \alpha \times \Delta_n$ with $\Delta_n \equiv -(\check{F}_{KFAC}(\theta_n))^{-1} \nabla E(\theta_n)$
  - When Δ_n is a good direction: take α large
  - When Δ_n is a bad direction: take α small
- How to quantify it ?

Update evaluation

- Define a local quadratic model: use the exact Fisher $F(\theta_n)$
  - $M_n(\delta) \equiv \frac{1}{2}\delta^T F(\theta_n)\delta + \nabla E(\theta_n)^T \delta + E(\theta_n)$
  - Minimize $M_n(\alpha\ \Delta_n)$ over α
- Momentum extension: find minimum for $\delta = \alpha\ \Delta_n + \mu\ \delta_{n-1}$
  - Analytical solution

$$\begin{pmatrix} \alpha \\ \mu \end{pmatrix} = -\begin{pmatrix} \Delta_n^T F(\theta_n)\Delta_n & \Delta_n^T F(\theta_n)\delta_{n-1} \\ \Delta_n^T F(\theta_n)\delta_{n-1} & \delta_{n-1}^T F(\theta_n)\delta_{n-1} \end{pmatrix}^{-1} \begin{pmatrix} \nabla E(\theta_n)^T\ \Delta_n \\ \nabla E(\theta_n)^T\ \delta_{n-1} \end{pmatrix}$$

Regularization

- Trust regions are all the more important !
- Direction/scale split $\Rightarrow$ use different trust regions for each
- Directional trust region
  - Tikhonov regularization: $\check{F}^{l\ Reg}_{KFAC} \simeq 4 \times \left(\mathbb{E}\left[a_{l-1}a_{l-1}^T\right] + \gamma\pi_l\ \mathrm{Id}\right) \otimes \left(\mathbb{E}\left[e_l e_l^T\right] + \frac{\gamma}{\pi_l}\ \mathrm{Id}\right)$
  - Regularizes calculation of inverses as well
  - π_l automatically chosen to minimize cross-term
  - γ adapted with greedy algorithm: take best of $\left(\frac{\gamma}{\omega_2}, \gamma, \omega_2\ \gamma\right)$
- Scaling trust region
  - Spherical regularization: $F^{reg} = F + \lambda\ \mathrm{Id}$
  - λ adapted with a Levenberg-Marquardt algorithm [Moré (1978)]
  - Measure quality of local model with $\rho \equiv \frac{E(\theta_n + \delta_n) - E(\theta_n)}{M_n(\delta_n) - M_n(0)}$
    - If $\rho < 0.25$: $\lambda \leftarrow \frac{\lambda}{\omega_1}$
    - If $\rho > 0.75$: $\lambda \leftarrow \omega_1\ \lambda$
    - If $0.25 \leq \rho \leq 0.75$: $\lambda \leftarrow \lambda$

Mitigating the crudeness of KFAC

General strategy

- KFAC update: $\delta_n = \alpha \times \Delta_n$ with $\Delta_n \equiv -(\check{F}_{KFAC}(\theta_n))^{-1} \nabla E(\theta_n)$
  - When Δ_n is a good direction: take α large
  - When Δ_n is a bad direction: take α small
- How to quantify it ?

Update evaluation

- Define a local quadratic model: use the exact Fisher $F(\theta_n)$
  - $M_n(\delta) \equiv \frac{1}{2}\delta^T F(\theta_n)\delta + \nabla E(\theta_n)^T \delta + E(\theta_n)$
  - Minimize $M_n(\alpha\ \Delta_n)$ over α
- Momentum extension: find minimum for $\delta = \alpha\ \Delta_n + \mu\ \delta_{n-1}$
  - Analytical solution

$$\begin{pmatrix} \alpha \\ \mu \end{pmatrix} = -\begin{pmatrix} \Delta_n^T F(\theta_n)\Delta_n & \Delta_n^T F(\theta_n)\delta_{n-1} \\ \Delta_n^T F(\theta_n)\delta_{n-1} & \delta_{n-1}^T F(\theta_n)\delta_{n-1} \end{pmatrix}^{-1} \begin{pmatrix} \nabla E(\theta_n)^T\ \Delta_n \\ \nabla E(\theta_n)^T\ \delta_{n-1} \end{pmatrix}$$

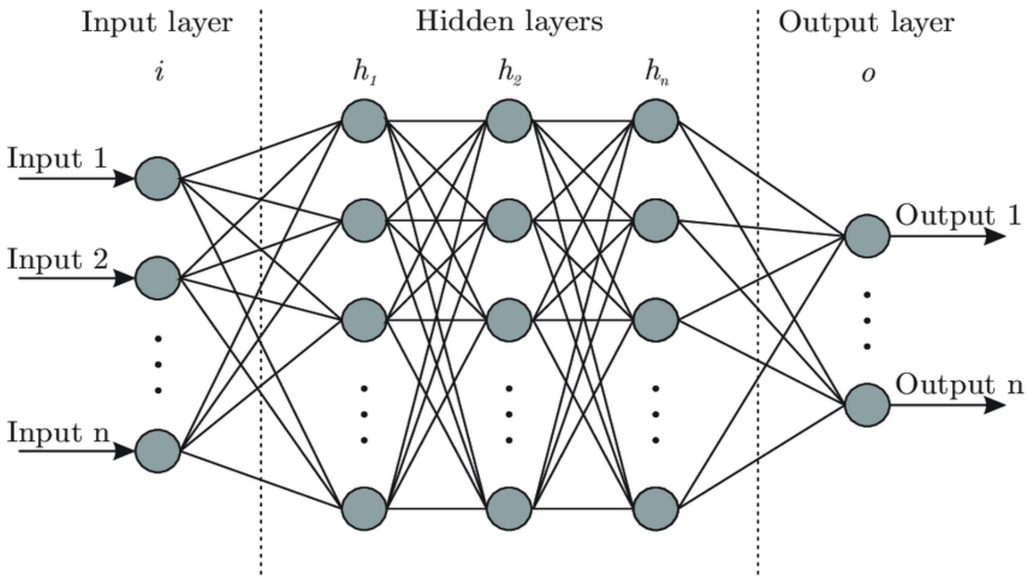
Regularization

- Trust regions are all the more important !
- Direction/scale split $\Rightarrow$ use different trust regions for each
- Directional trust region
  - Tikhonov regularization: $\check{F}^{l\ Reg}_{KFAC} \simeq 4 \times \left(\mathbb{E}\left[a_{l-1}a_{l-1}^T\right] + \gamma\pi_l\ \mathrm{Id}\right) \otimes \left(\mathbb{E}\left[e_l e_l^T\right] + \frac{\gamma}{\pi_l}\ \mathrm{Id}\right)$
  - Regularizes calculation of inverses as well
  - π_l automatically chosen to minimize cross-term
  - γ adapted with greedy algorithm: take best of $\left(\frac{\gamma}{\omega_2}, \gamma, \omega_2\ \gamma\right)$
- Scaling trust region
  - Spherical regularization: $F^{reg} = F + \lambda\ \mathrm{Id}$
  - λ adapted with a Levenberg-Marquardt algorithm [Moré (1978)]
  - Measure quality of local model with $\rho \equiv \frac{E(\theta_n + \delta_n) - E(\theta_n)}{M_n(\delta_n) - M_n(0)}$
    - If $\rho < 0.25$: $\lambda \leftarrow \frac{\lambda}{\omega_1}$
    - If $\rho > 0.75$: $\lambda \leftarrow \omega_1\ \lambda$
    - If $0.25 \leq \rho \leq 0.75$: $\lambda \leftarrow \lambda$

Two hyperparameters

$\omega_1, \omega_2\ \in\ ]0, 1]$

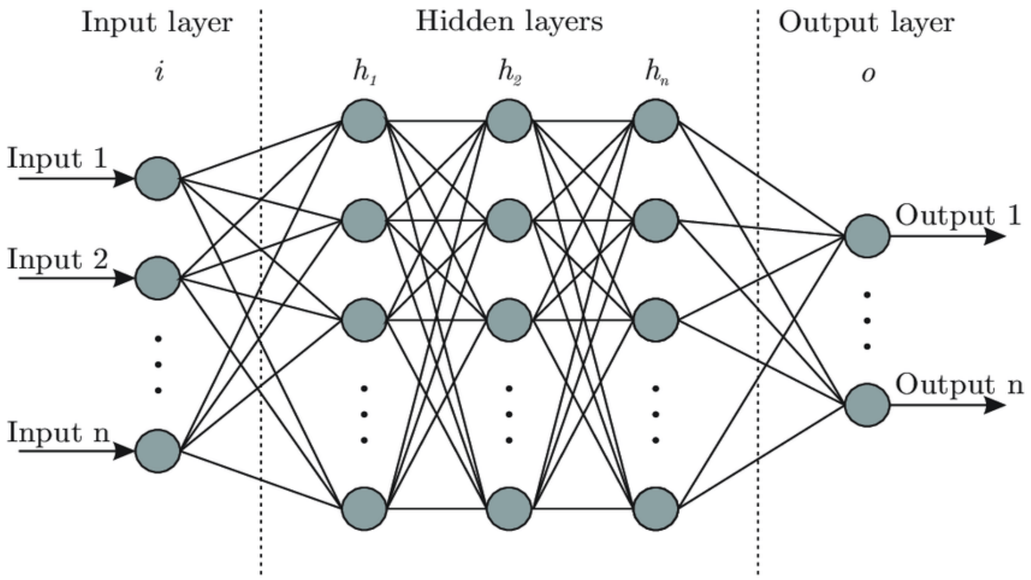
Breaking of Kronecker factorization



Kronecker-Factorization in slow-motion

- Assume your regular feed-forward neural network
  - Fisher matrix: $F_{ij}(\theta) = 4 \times \mathbb{E}_{X \sim |\Psi_\theta|^2} \left[\partial_{\theta_i} \ln |\Psi_\theta(X)| \; \partial_{\theta_j} \ln |\Psi_\theta(X)| \right]$
  - Chain rule: $\dfrac{\partial \ln |\Psi_\theta(X)|}{\partial W_{ij}^l} = \dfrac{\partial h_j^l}{\partial W_{ij}^l} \dfrac{\partial \ln |\Psi_\theta(X)|}{\partial h_j^l} = a_i^{l-1} e_j^l$
  - For any **vectors** $\in \mathbb{R}^H$: $\text{vec}\left(ea^T\right) = a \otimes e$
- Back to Fisher
  - $F_{ij}(\theta) = 4\mathbb{E}\left[\text{vec}\left(ea^T\right) \text{vec}\left(ea^T\right)^T\right] = 4\mathbb{E}\left[(a \otimes e)(a^T \otimes e^T)\right] = 4\mathbb{E}\left[(aa^T) \otimes (ee^T)\right]$

Breaking of Kronecker factorization



Kronecker-Factorization in slow-motion

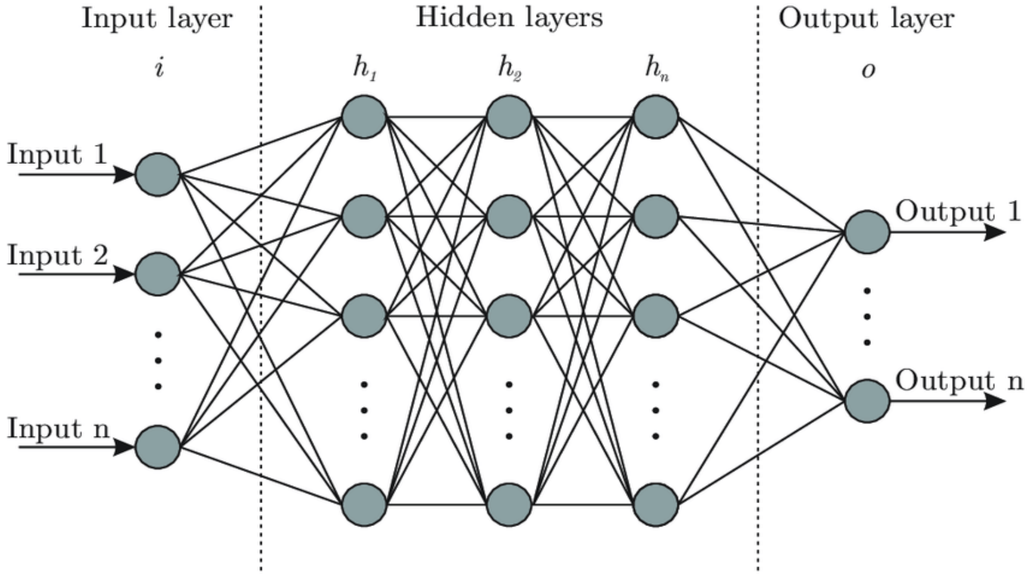
- Assume your regular feed-forward neural network
  - Fisher matrix: $F_{ij}(\theta) = 4 \times \mathbb{E}_{X \sim |\Psi_\theta|^2} \left[\partial_{\theta_i} \ln |\Psi_\theta(X)| \; \partial_{\theta_j} \ln |\Psi_\theta(X)| \right]$
  - Chain rule: $\dfrac{\partial \ln |\Psi_\theta(X)|}{\partial W_{ij}^l} = \dfrac{\partial h_j^l}{\partial W_{ij}^l} \dfrac{\partial \ln |\Psi_\theta(X)|}{\partial h_j^l} = a_i^{l-1} e_j^l$
  - For any **vectors** $\in \mathbb{R}^H$: $\text{vec}\left(ea^T\right) = a \otimes e$

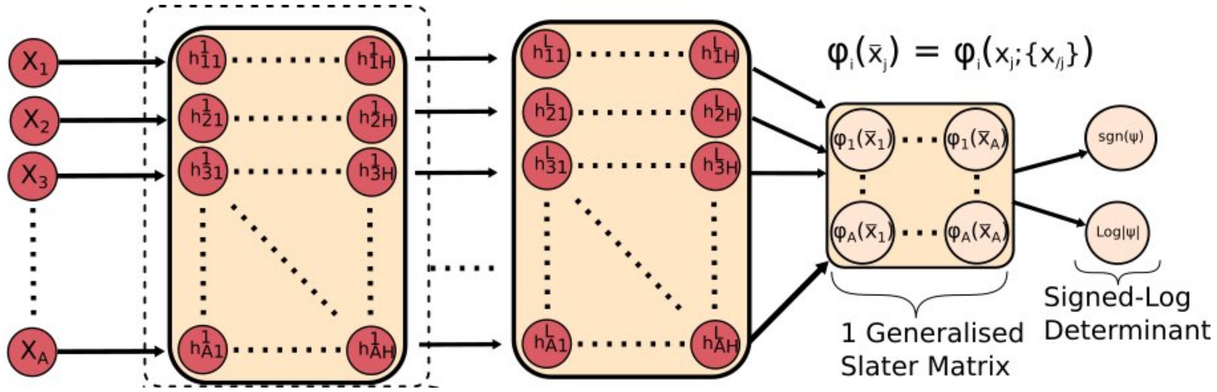
$$\boxed{F_{KFAC} \simeq 4 \times \mathbb{E}\left[aa^T\right] \otimes \mathbb{E}\left[ee^T\right]}$$

KFAC

- Back to Fisher
  - $F_{ij}(\theta) = 4\mathbb{E}\left[\text{vec}\left(ea^T\right) \text{vec}\left(ea^T\right)^T\right] = 4\mathbb{E}\left[(a \otimes e)(a^T \otimes e^T)\right] = 4\mathbb{E}\left[(aa^T) \otimes (ee^T)\right]$

Breaking of Kronecker factorization





Kronecker-Factorization in slow-motion

- Assume your regular feed-forward neural network
  - Fisher matrix: $F_{ij}(\theta) = 4 \times \mathbb{E}_{X\sim|\Psi_\theta|^2}\left[\partial_{\theta_i} \ln|\Psi_\theta(X)|\ \partial_{\theta_j} \ln|\Psi_\theta(X)|\right]$
  - Chain rule: $\dfrac{\partial \ln|\Psi_\theta(X)|}{\partial W^l_{ij}} = \dfrac{\partial h^l_j}{\partial W^l_{ij}} \dfrac{\partial \ln|\Psi_\theta(X)|}{\partial h^l_j} = a^{l-1}_i e^l_j$
  - For any **vectors** $\in \mathbb{R}^H$: $\mathrm{vec}\left(ea^T\right) = a \otimes e$

$$\boxed{F_{KFAC} \simeq 4 \times \mathbb{E}\left[aa^T\right] \otimes \mathbb{E}\left[ee^T\right]}$$

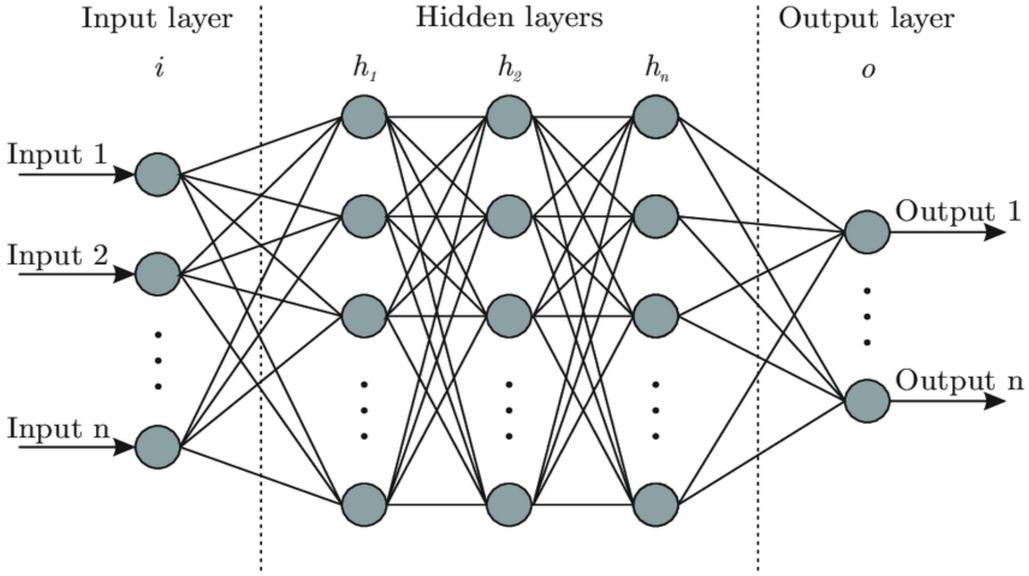
KFAC

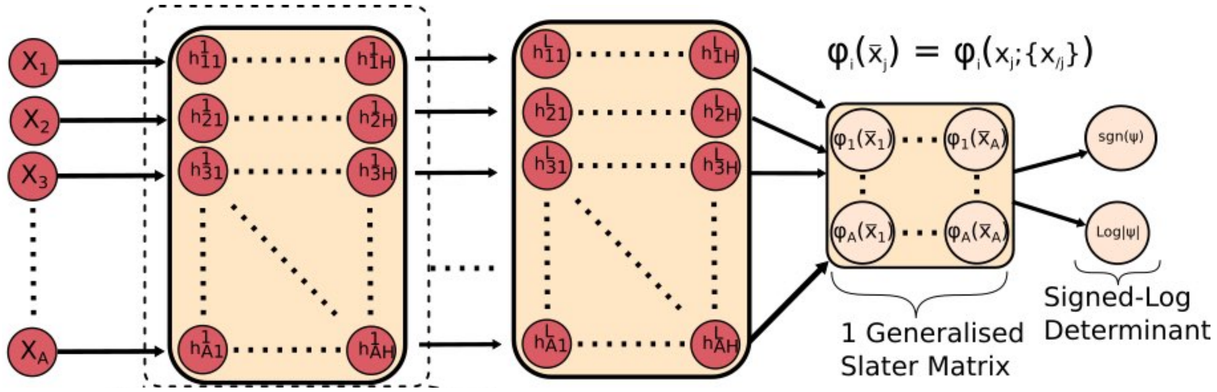
- Back to Fisher
  - $F_{ij}(\theta) = 4\mathbb{E}\left[\mathrm{vec}\left(ea^T\right)\mathrm{vec}\left(ea^T\right)^T\right] = 4\mathbb{E}\left[(a\otimes e)(a^T \otimes e^T)\right] = 4\mathbb{E}\left[(aa^T)\otimes(ee^T)\right]$

KFAC for fermionic neural networks?

- FermiNet-like uses weight-sharing
  - The *same* weights are used for each rows
  - Chain rule: $\dfrac{\partial \ln|\Psi_\theta(X)|}{\partial W^l_{ij}} = \displaystyle\sum_{m=1}^{A} \dfrac{\partial h^l_{jm}}{\partial W^l_{ij}} \dfrac{\partial \ln|\Psi_\theta(X)|}{\partial h^l_{jm}} = a^{l-1}_i e^l_j$
  - But now a and e are $H \times A$ **matrices**: $\mathrm{vec}\left(ea^T\right) = (a \otimes e)\,.\,\mathrm{vec}(I_A)$
- Back to Fisher
  - $F_{ij}(\theta) = 4\mathbb{E}\left[(a\otimes e)\ \left(\mathrm{vec}(I_A)\ \mathrm{vec}(I_A)^T\right)\ (a^T \otimes e^T)\right]$

Breaking of Kronecker factorization





Kronecker-Factorization in slow-motion

- Assume your regular feed-forward neural network
  - Fisher matrix: $F_{ij}(\theta) = 4 \times \mathbb{E}_{X\sim|\Psi_\theta|^2}\left[\partial_{\theta_i} \ln|\Psi_\theta(X)|\ \partial_{\theta_j} \ln|\Psi_\theta(X)|\right]$
  - Chain rule: $\dfrac{\partial \ln|\Psi_\theta(X)|}{\partial W^l_{ij}} = \dfrac{\partial h^l_j}{\partial W^l_{ij}} \dfrac{\partial \ln|\Psi_\theta(X)|}{\partial h^l_j} = a^{l-1}_i e^l_j$
  - For any **vectors** $\in \mathbb{R}^H$: $\mathrm{vec}\left(ea^T\right) = a \otimes e$

$$\boxed{F_{KFAC} \simeq 4 \times \mathbb{E}\left[aa^T\right] \otimes \mathbb{E}\left[ee^T\right]}$$

KFAC

- Back to Fisher
  - $F_{ij}(\theta) = 4\mathbb{E}\left[\mathrm{vec}\left(ea^T\right)\mathrm{vec}\left(ea^T\right)^T\right] = 4\mathbb{E}\left[(a\otimes e)(a^T \otimes e^T)\right] = 4\mathbb{E}\left[(aa^T)\otimes(ee^T)\right]$

KFAC for fermionic neural networks?

- FermiNet-like uses weight-sharing
  - The *same* weights are used for each rows
  - Chain rule: $\dfrac{\partial \ln|\Psi_\theta(X)|}{\partial W^l_{ij}} = \displaystyle\sum_{m=1}^{A} \dfrac{\partial h^l_{jm}}{\partial W^l_{ij}} \dfrac{\partial \ln|\Psi_\theta(X)|}{\partial h^l_{jm}} = a^{l-1}_i e^l_j$
  - But now a and e are $H \times A$ **matrices**: $\mathrm{vec}\left(ea^T\right) = (a \otimes e)\,.\,\mathrm{vec}(I_A)$
- Back to Fisher
  - $F_{ij}(\theta) = 4\mathbb{E}\left[(a\otimes e)\ \left(\mathrm{vec}(I_A)\ \mathrm{vec}(I_A)^T\right)\ (a^T \otimes e^T)\right]$ ✗ KFAC

A primer on MINRES

The linear problem

- Definition
  - Solve: $Ax = b$, where $A^T = A$ and where Ax can be calculated efficiently
- Reformulations
  - Change of variable: $x = P_k u$, where $P_k^T P_k = I$
  - Variational: minimize $f(u) = (AP_k u - b)^T (AP_k u - b)$
- Stationary solution: $\mathrm{d}f = 0$
  - $P_k^T A^2 P_k \; u = P_k^T A \; b \rightarrow$ solution $u_k \equiv$ *minimum residual*

➡ What P_k to choose ?

A primer on MINRES

The linear problem

- Definition
  - Solve: $Ax = b$, where $A^T = A$ and where Ax can be calculated efficiently
- Reformulations
  - Change of variable: $x = P_k u$, where $P_k^T P_k = I$
  - Variational: minimize $f(u) = (AP_k u - b)^T (AP_k u - b)$
- Stationary solution: $\mathrm{d}f = 0$
  - $P_k^T A^2 P_k \; u = P_k^T A \; b \rightarrow$ solution $u_k \equiv$ *minimum residual*

➡ What P_k to choose ?

Lanczos tridiagonalization algorithm [Lanczos (1950)]

- Initialization: $v_1 = b/\beta_1$ and $\beta_1 = ||b||$
- Recursion: α_i, β_i and v_i
  - $\beta_{i+1} v_{i+1} = A v_i - \alpha_i v_i - \beta_i v_{i-1}$
  - $\alpha_i = v_i^T A v_i$ and β_{i+1} st $||v_{i+1}|| = 1$
- Output at step k: T_k and $P_k = [v_1 | \ldots | v_k]$ s.t.
  - $AP_k \; = \; P_k T_k \; + \; \beta_{k+1} \; v_{k+1} e_k^T$
  - $P_k^T P_k = I$ and $P_k^T v_{k+1} = 0$

$$\underbrace{\begin{pmatrix} \alpha_1 & \beta_2 & & & & \\ \beta_2 & \alpha_2 & \beta_3 & & & \\ & \beta_3 & \alpha_3 & \ddots & & \\ & & \ddots & \ddots & \beta_{k-1} & \\ & & & \beta_{k-1} & \alpha_{k-1} & \beta_k \\ & & & & \beta_k & \alpha_k \end{pmatrix}}_{P_k^T A P_k = T_k}$$

A primer on MINRES

The linear problem

- Definition
  - Solve: $Ax = b$, where $A^T = A$ and where Ax can be calculated efficiently
- Reformulations
  - Change of variable: $x = P_k u$, where $P_k^T P_k = I$
  - Variational: minimize $f(u) = (AP_k u - b)^T (AP_k u - b)$
- Stationary solution: $\mathrm{d}f = 0$
  - $P_k^T A^2 P_k \; u = P_k^T A \; b \rightarrow$ solution $u_k \equiv$ *minimum residual*

➡ What P_k to choose ?

Lanczos tridiagonalization algorithm [Lanczos (1950)]

- Initialization: $v_1 = b/\beta_1$ and $\beta_1 = ||b||$
- Recursion: α_i, β_i and v_i
  - $\beta_{i+1} v_{i+1} = A v_i - \alpha_i v_i - \beta_i v_{i-1}$
  - $\alpha_i = v_i^T A v_i$ and β_{i+1} st $||v_{i+1}|| = 1$
- Output at step k: T_k and $P_k = [v_1 | \ldots | v_k]$ s.t.
  - $AP_k \; = \; P_k T_k \; + \; \beta_{k+1} \; v_{k+1} e_k^T$
  - $P_k^T P_k = I$ and $P_k^T v_{k+1} = 0$

$$\underbrace{\begin{pmatrix} \alpha_1 & \beta_2 & & & & \\ \beta_2 & \alpha_2 & \beta_3 & & & \\ & \beta_3 & \alpha_3 & \ddots & & \\ & & \ddots & \ddots & \beta_{k-1} & \\ & & & \beta_{k-1} & \alpha_{k-1} & \beta_k \\ & & & & \beta_k & \alpha_k \end{pmatrix}}_{P_k^T A P_k = T_k}$$

MINRES solver

[Paige, Saunders (1975)]

- Two-step algorithm
  - Lanczos: $\left(T_k^2 + \beta_{k+1}^2 e_k e_k^T\right) u = \beta_1 T_k e_1$
  - LQ factorization: $T_k = L_k Q_k$

$\boxed{\text{MINRES}}$

- Nice properties
  - Cumulative $\rightarrow$ keep only last few steps in memory
  - Only requires $Ax \rightarrow$ practical for large sparse matrices
  - Iterative $\rightarrow$ can improve initial guess *just a little bit*
  - Guaranteed to decrease norm of the residue

A primer on MINRES

The linear problem

- Definition
  - Solve: $Ax = b$, where $A^T = A$ and where Ax can be calculated efficiently
- Reformulations
  - Change of variable: $x = P_k u$, where $P_k^T P_k = I$
  - Variational: minimize $f(u) = (AP_k u - b)^T (AP_k u - b)$
- Stationary solution: $\mathrm{d}f = 0$
  - $P_k^T A^2 P_k \; u = P_k^T A \; b \rightarrow$ solution $u_k \equiv$ *minimum residual*

➡ What P_k to choose ?

MINRES solver

[Paige, Saunders (1975)]

- Two-step algorithm
  - Lanczos: $\left(T_k^2 + \beta_{k+1}^2 e_k e_k^T\right) u = \beta_1 T_k e_1$
  - LQ factorization: $T_k = L_k Q_k$

$\boxed{\text{MINRES}}$

- Nice properties
  - Cumulative → keep only last few steps in memory
  - Only requires Ax → practical for large sparse matrices
  - Iterative → can improve initial guess *just a little bit*
  - Guaranteed to decrease norm of the residue

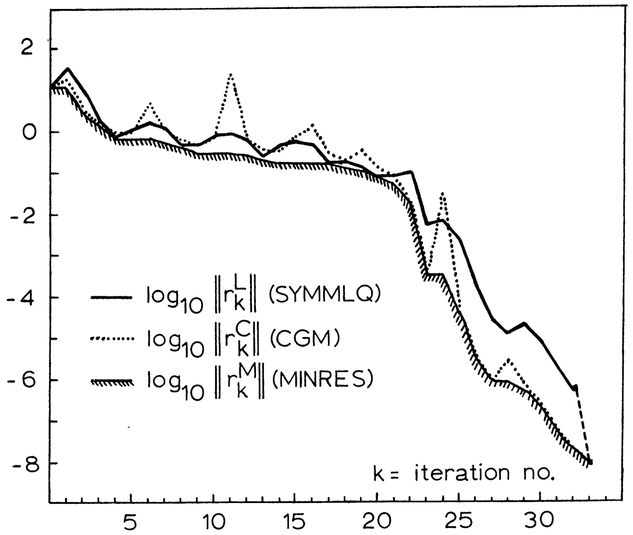
Lanczos tridiagonalization algorithm

[Lanczos (1950)]

- Initialization: $v_1 = b/\beta_1$ and $\beta_1 = ||b||$
- Recursion: α_i, β_i and v_i
  - $\beta_{i+1} v_{i+1} = A v_i - \alpha_i v_i - \beta_i v_{i-1}$
  - $\alpha_i = v_i^T A v_i$ and β_{i+1} st $||v_{i+1}|| = 1$
- Output at step k: T_k and $P_k = [v_1 | \ldots | v_k]$ s.t.
  - $AP_k \; = \; P_k T_k \; + \; \beta_{k+1} \; v_{k+1} e_k^T$
  - $P_k^T P_k = I$ and $P_k^T v_{k+1} = 0$

➡

$$\underbrace{\begin{pmatrix} \alpha_1 & \beta_2 & & & & \\ \beta_2 & \alpha_2 & \beta_3 & & & \\ & \beta_3 & \alpha_3 & \ddots & & \\ & & \ddots & \ddots & \beta_{k-1} & \\ & & & \beta_{k-1} & \alpha_{k-1} & \beta_k \\ & & & & \beta_k & \alpha_k \end{pmatrix}}_{P_k^T A P_k = T_k}$$



Original numerical test [Paige, Saunders (1975)]

Can be used to gradually refine KFAC approximation!

Implementing decisional gradient descent

Reminder

- Local problem of decisional gradient descent
  - $M_n(\delta) = \nabla E(\theta_n)^T \delta + E(\theta_n)$
  - $T_n(r) = \left\{\delta : \ \delta^T G_{VMC}(\theta_n)\, \delta \leq r^2\right\}$

➡ $\delta_n \propto G_{VMC}^{-1}(\theta_n)\ \nabla E(\theta_n)$

Implementing decisional gradient descent

Reminder

- Local problem of decisional gradient descent
  - $M_n(\delta) = \nabla E(\theta_n)^T \delta + E(\theta_n)$
  - $T_n(r) = \left\{\delta : \ \delta^T G_{VMC}(\theta_n)\, \delta \leq r^2\right\}$

➡ $\delta_n \propto G_{VMC}^{-1}(\theta_n)\ \nabla E(\theta_n)$

Evaluation of the update direction

- Approximation on the metric:
  - Block diagonal between layers $G_{VMC}(\theta_n) \to \breve{G}_{VMC}(\theta_n)$
- MINRES solver:
  - Linear system: $\breve{G}_{VMC}(\theta_n)\,.\,x = \nabla E(\theta_n)$
  - Starting point: $\zeta\ \delta_{n-1}$ with $\zeta = 0.95$
  - Preconditioner: $\left(\mathrm{diag}(\breve{G}_{VMC}(\theta_n)) + \kappa I\right)^{\xi}$
    - $\kappa = 10^{-2}$ and $\xi = 0.75$
- Inspired by Hessian-Free type of optimizer

[For a nice review: Martens, Sutskever (2012)]

Implementing decisional gradient descent

Reminder

- Local problem of decisional gradient descent
  - $M_n(\delta) = \nabla E(\theta_n)^T \delta + E(\theta_n)$
  - $T_n(r) = \left\{\delta : \ \delta^T G_{VMC}(\theta_n)\, \delta \leq r^2\right\}$

➡ $\delta_n \propto G_{VMC}^{-1}(\theta_n)\ \nabla E(\theta_n)$

Evaluation of the scaling

- Scaling factors: use the exact metric $G_{VMC}(\theta_n)$
  - $M_n^T(\delta) \equiv \frac{1}{2}\delta^T G_{VMC}(\theta_n)\delta + \nabla E(\theta_n)^T \delta + E(\theta_n)$
  - Minimize $M_n^T(\alpha\ \Delta_n + \mu\ \delta_{n-1})$ over α and μ

➡ Minimum obtained defines our δ_n

Evaluation of the update direction

- Approximation on the metric:
  - Block diagonal between layers $G_{VMC}(\theta_n) \to \breve{G}_{VMC}(\theta_n)$
- MINRES solver:
  - Linear system: $\breve{G}_{VMC}(\theta_n)\,.\,x = \nabla E(\theta_n)$
  - Starting point: $\zeta\ \delta_{n-1}$ with $\zeta = 0.95$
  - Preconditioner: $\left(\text{diag}(\breve{G}_{VMC}(\theta_n)) + \kappa I\right)^{\xi}$
    - $\kappa = 10^{-2}$ and $\xi = 0.75$
- Inspired by Hessian-Free type of optimizer

[For a nice review: Martens, Sutskever (2012)]

Implementing decisional gradient descent

Reminder

- Local problem of decisional gradient descent
  - $M_n(\delta) = \nabla E(\theta_n)^T \delta + E(\theta_n)$
  - $T_n(r) = \left\{\delta : \ \delta^T G_{VMC}(\theta_n)\, \delta \leq r^2\right\}$

➡ $\delta_n \propto G_{VMC}^{-1}(\theta_n)\ \nabla E(\theta_n)$

Evaluation of the scaling

- Scaling factors: use the exact metric $G_{VMC}(\theta_n)$
  - $M_n^T(\delta) \equiv \frac{1}{2}\delta^T G_{VMC}(\theta_n)\delta + \nabla E(\theta_n)^T \delta + E(\theta_n)$
  - Minimize $M_n^T(\alpha\ \Delta_n + \mu\ \delta_{n-1})$ over α and μ

➡ Minimum obtained defines our δ_n

Evaluation of the update direction

- Approximation on the metric:
  - Block diagonal between layers $G_{VMC}(\theta_n) \to \breve{G}_{VMC}(\theta_n)$
- MINRES solver:
  - Linear system: $\breve{G}_{VMC}(\theta_n)\,.\,x = \nabla E(\theta_n)$
  - Starting point: $\zeta\ \delta_{n-1}$ with $\zeta = 0.95$
  - Preconditioner: $\left(\mathrm{diag}(\breve{G}_{VMC}(\theta_n)) + \kappa I\right)^{\xi}$
    - $\kappa = 10^{-2}$ and $\xi = 0.75$
- Inspired by Hessian-Free type of optimizer

[For a nice review: Martens, Sutskever (2012)]

Regularizations

- Only one regulator for now: λ
  - Used both in MINRES and Re-scaling steps
  - No KFAC approximation → 2 regulators less justified
  - Levenberg-Marquardt rule: dynamical λ
- Hessian-Free type of optimizer [Martens, Sutskever (2012)]
  - Suggests smarter regulators → more stable
  - Left for future improvements

Implementing decisional gradient descent

Reminder

- Local problem of decisional gradient descent
  - $M_n(\delta) = \nabla E(\theta_n)^T \delta + E(\theta_n)$
  - $T_n(r) = \left\{\delta : \ \delta^T G_{VMC}(\theta_n)\, \delta \leq r^2\right\}$

➡ $\delta_n \propto G_{VMC}^{-1}(\theta_n)\ \nabla E(\theta_n)$

Evaluation of the scaling

- Scaling factors: use the exact metric $G_{VMC}(\theta_n)$
  - $M_n^T(\delta) \equiv \frac{1}{2}\delta^T G_{VMC}(\theta_n)\delta + \nabla E(\theta_n)^T \delta + E(\theta_n)$
  - Minimize $M_n^T(\alpha\ \Delta_n + \mu\ \delta_{n-1})$ over α and μ

➡ Minimum obtained defines our δ_n

Evaluation of the update direction

- Approximation on the metric:
  - Block diagonal between layers $G_{VMC}(\theta_n) \to \breve{G}_{VMC}(\theta_n)$
- MINRES solver:
  - Linear system: $\breve{G}_{VMC}(\theta_n)\,.\,x = \nabla E(\theta_n)$
  - Starting point: $\zeta\ \delta_{n-1}$ with $\zeta = 0.95$
  - Preconditioner: $\left(\mathrm{diag}(\breve{G}_{VMC}(\theta_n)) + \kappa I\right)^{\xi}$
    - $\kappa = 10^{-2}$ and $\xi = 0.75$
- Inspired by Hessian-Free type of optimizer

[For a nice review: Martens, Sutskever (2012)]

Regularizations

- Only one regulator for now: λ
  - Used both in MINRES and Re-scaling steps
  - No KFAC approximation → 2 regulators less justified
  - Levenberg-Marquardt rule: dynamical λ
- Hessian-Free type of optimizer [Martens, Sutskever (2012)]
  - Suggests smarter regulators → more stable
  - Left for future improvements

Decisional Gradient Descent (DGD)