Exploring the Learning-based Optimization Algorithms

Zaiwen Wen

Beijing International Center For Mathematical Research
Peking University

Thanks: Zhonglin Xie, Cheng Chen, Ruitao Chen, Tianyou Li, Chenyi Li, Ziyu Wang

XXIII International Conference MOTOR-2024

Outline

A continuous-time viewpoint of acceleration methods: $\min f(x)$

- Gradient descent method corresponds to gradient flow

$$x_{k+1} = x_k - \sqrt{s}\nabla f(x_k) \quad \Leftrightarrow \quad \dot{x}(t) = -\nabla f(x(t))$$

- Nesterov accelerated gradient method corresponds to

$$\begin{cases} x_k = y_{k-1} - s\nabla f\left(y_{k-1}\right) \\ y_k = x_k + \dfrac{k-1}{k+2}\left(x_k - x_{k-1}\right) \end{cases} \Leftrightarrow \begin{aligned} &\ddot{x}(t) + \frac{3}{t}\dot{x}(t) + \sqrt{s}\nabla^2 f(x(t))\dot{x}(t) \\ &\quad + \left(1 + \frac{3\sqrt{s}}{2t}\right)\nabla f(x(t)) = 0 \end{aligned}$$

- Dynamical inertial Newton with asymptotic vanishing damping

$$\ddot{x}(t) + \frac{\alpha}{t}\dot{x}(t) + \beta(t)\nabla^2 f(x(t))\dot{x}(t) + \gamma(t)\nabla f(x(t)) = 0 \qquad \text{(DIN-AVD)}$$

- Let $w(t) = \gamma(t) - \dot{\beta}(t) - \beta(t)/t$. Convergence condition for (DIN-AVD)

$$\gamma(t) > \dot{\beta}(t) + \frac{\beta(t)}{t}, \quad t\dot{w}(t) \leqslant (\alpha - 3)w(t), \quad \text{for all } t \geqslant t_0$$

Convergence rate: $f(x(t)) - f_\star = \mathcal{O}\left(1/(t^2 w(t))\right)$

Two important open questions and solutions

- How to develop acceleration methods using ODE viewpoint?

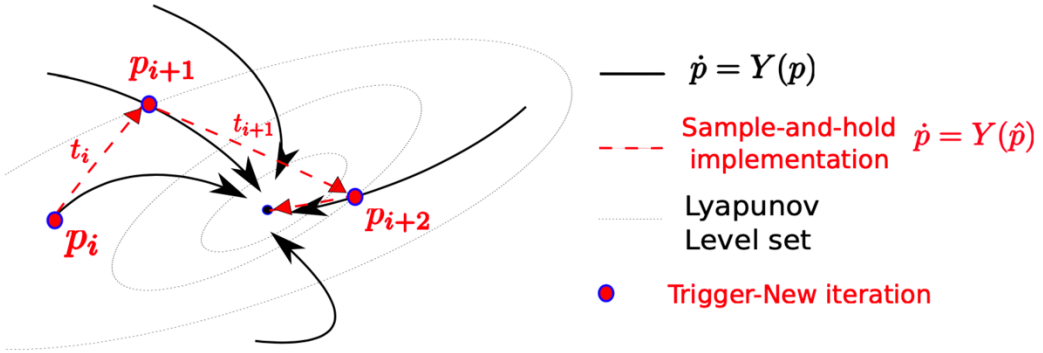


Figure: Rate-matching discretization

Combine error analysis in ODE and complexity analysis in optimization

- How to select the best coefficients for (DIN-AVD)?

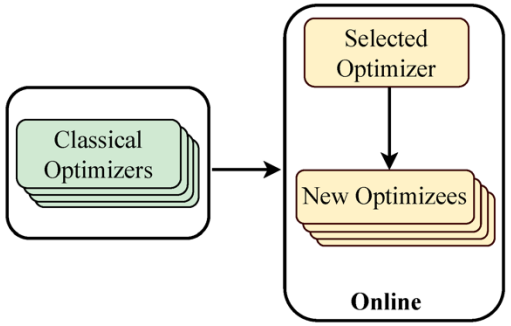


Figure: Classic Optimizer

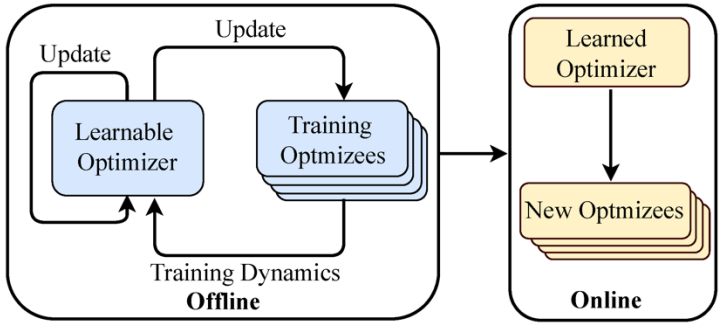
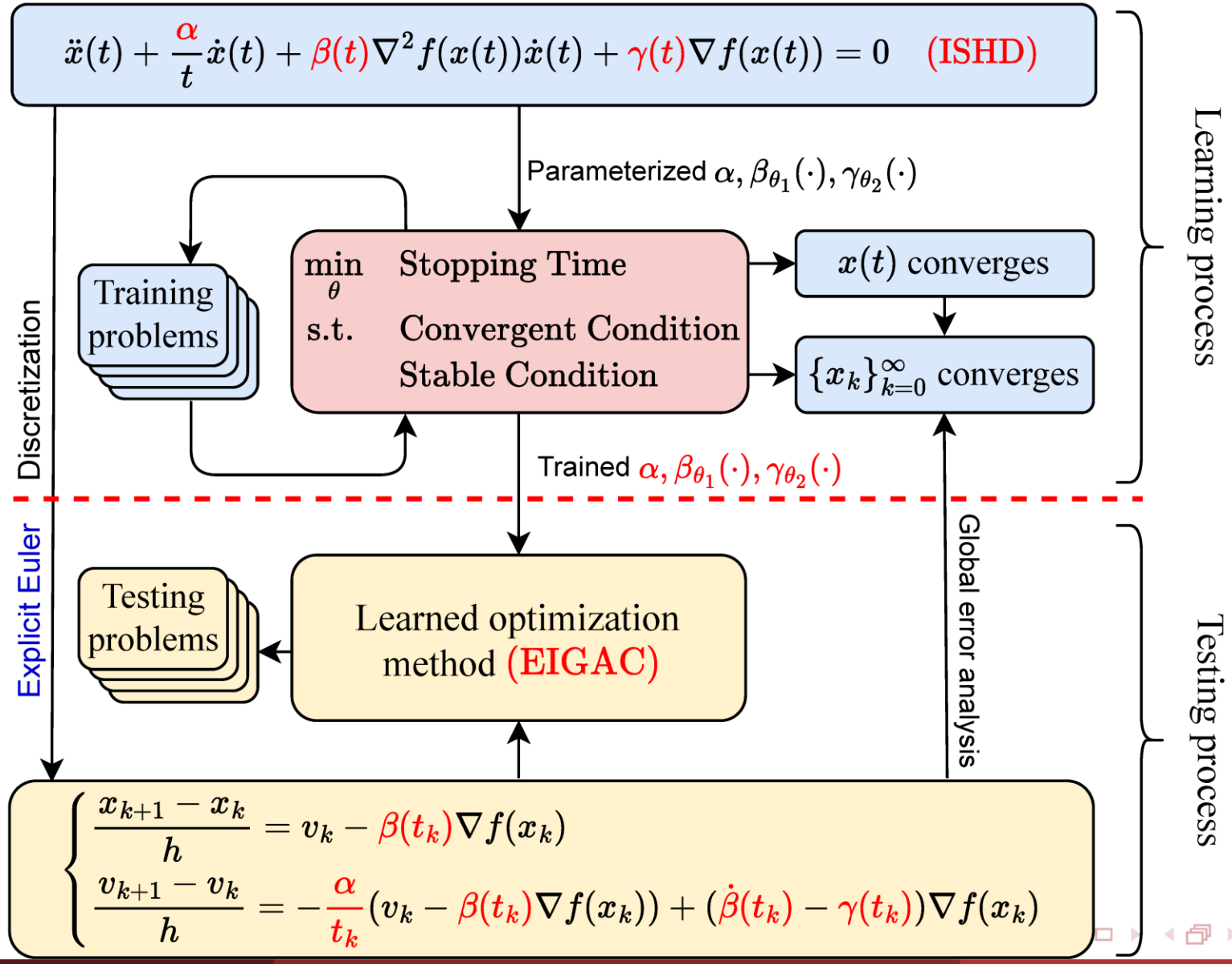


Figure: Learning to Optimize

A learning to optimize framework with theoretical guarantee

Our training and testing framework



A fundamental result: an enhanced convergent condition for DIN-AVD

Theorem

Given $\kappa \in (0,1], \lambda \in (0, \alpha - 1]$ *and* f *is twice differentiable convex*

$$\begin{aligned} &\delta(t) = t^2(\gamma(t) - \kappa\dot{\beta}(t) - \kappa\beta(t)/t) + (\kappa(\alpha - 1 - \lambda) - \lambda(1-\kappa))t\beta(t), \\ &w(t) = \gamma(t) - \dot{\beta}(t) - \beta(t)/t, \quad \delta(t) > 0, \quad \text{and} \quad \dot{\delta}(t) \le tw(t), \end{aligned} \tag{Cvg-cdt}$$

$\alpha \geqslant 3, t_0 > 0, \varepsilon > 0$ *are real numbers,* β *and* γ *are nonnegative continuously differentiable functions defined on* $[t_0, +\infty)$*. Then* $x(t)$ *is bounded and*

$$f(x(t)) - f_\star \le \mathcal{O}\left(\frac{1}{\delta(t)}\right), \ \|\nabla f(x(t))\| \le \mathcal{O}\left(\frac{1}{t\beta(t)}\right), \ \|\dot{x}(t)\| \le \mathcal{O}\left(\frac{1}{t}\right),$$

$$\int_{t_0}^{\infty} (\lambda t w(t) - \dot{\delta}(t))(f(x(t)) - f_\star)\,\mathrm{d}t \le \infty, \quad \int_{t_0}^{\infty} t(\alpha - 1 - \lambda)\|\dot{x}(t)\|^2\,\mathrm{d}t \le \infty,$$

$$\int_{t_0}^{\infty} t^2\beta(t)w(t)\|\nabla f(x)\|^2\,\mathrm{d}t \le \infty, \quad \int_{t_0}^{\infty} t^2\beta(t)\langle \nabla^2 f(x(t))\dot{x}(t), \dot{x}(t)\rangle\,\mathrm{d}t \le \infty$$

Applying forward Euler scheme to (DIN-AVD)

- Let $v(t_0) = x(t_0) + \beta(t_0)\nabla f(x(t_0))$ and

$$\psi_{\Xi}(x(t), v(t), t) = \begin{pmatrix} v(t) - \beta(t)\nabla f(x(t)) \\ -\frac{\alpha}{t}(v(t) - \beta(t)\nabla f(x(t))) + (\dot{\beta}(t) - \gamma(t))\nabla f(x(t)) \end{pmatrix} \tag{1}$$

- The equation (DIN-AVD) can be reformulated as the first-order system

$$\begin{pmatrix} \dot{x}(t) \\ \dot{v}(t) \end{pmatrix} = \psi_{\Xi}(x(t), v(t), t), \text{ notice that } \nabla^2 f(x(t))\dot{x}(t) = \frac{\mathrm{d}}{\mathrm{d}t}\nabla f(x(t))$$

- Let h be the step size, $t_k = t_0 + kh, k \geq 0$. The forward Euler scheme of (DIN-AVD) is

$$\begin{cases} \dfrac{x_{k+1} - x_k}{h} = v_k - \beta(t_k)\nabla f(x_k), \\ \dfrac{v_{k+1} - v_k}{h} = -\dfrac{\alpha}{t}\left(v_k - \beta(t_k)\nabla f(x_k)\right) + (\dot{\beta}(t_k) - \gamma(t_k))\nabla f(x_k) \end{cases} \tag{F-Euler}$$

Conditions for stable discretization

Theorem

Suppose the assumptions in Theorem 1 hold. Given t_0, s_0, and h, the sequence $\{x_k\}_{k=0}^{\infty}$ is

$$\bar{x}(t) = x_k + \frac{x_{k+1} - x_k}{h}(t - t_k), \qquad t \in [t_k, t_{k+1}). \tag{2}$$

Assume three constants $0 \le C_1$, $0 < C_2 \le 1/h - 1/t_0$, and $0 < C_3$ fulfill

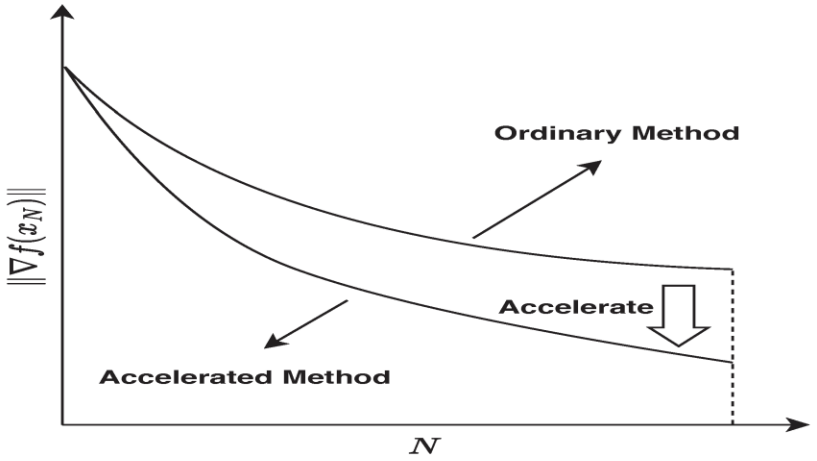
$$|\dot{\beta}(t)| \le C_1\beta(t), \quad |\dot{\gamma}(t) - \ddot{\beta}(t)| \le C_2(\gamma(t) - \dot{\beta}(t)), \quad \beta(t) \le C_3 w(t). \tag{3}$$

Then, it holds $f(x_k) - f_\star \le \mathcal{O}(1/k)$ under the following stability condition:

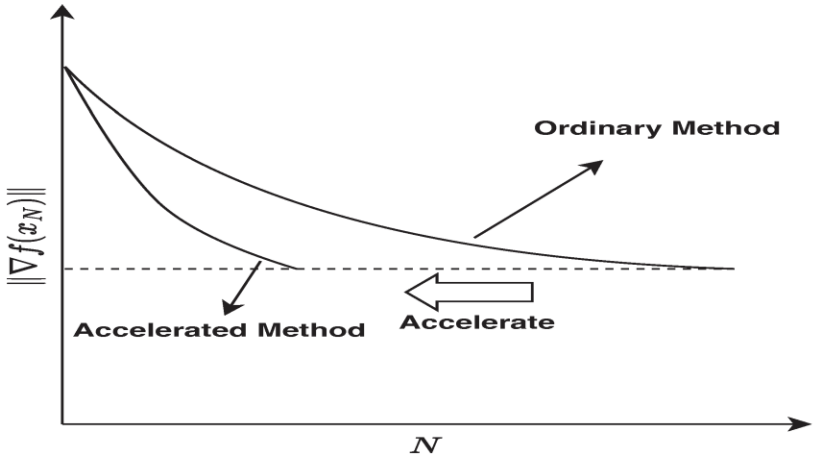
$$\Lambda(x, f) \ge \|\nabla^2 f(x)\|, \quad \alpha\beta(t)/t \le \gamma(t) - \dot{\beta}(t) \le \beta(t)/h, \tag{STB-CDT}$$

$$\sqrt{\int_0^1 \Lambda((1-\tau)X(t, \Xi, f) + \tau\bar{x}(t), f)\,\mathrm{d}\tau} \le \frac{\sqrt{\gamma(t) - \dot{\beta}(t)} + \sqrt{\gamma(t) - \dot{\beta}(t) - \frac{\alpha}{t}\beta(t)}}{\beta(t)}.$$

Stopping time: a differentiable continuous-time complexity



(a) Measure-based



(b) Complexity-based

Definition (Stopping Time)

Given the initial time t_0, the initial value x_0, the initial velocity $\dot{x}(t_0)$, the trajectory $X(\Xi, t, f)$ of the system (DIN-AVD), and a tolerance ε, the stopping time of the criterion $\|\nabla f(x)\| \leq \varepsilon$ is

$$T(\Xi, f) = \inf\{t \mid \|\nabla f(X(\Xi, t, f))\| \leq \varepsilon, t \geq t_0\}. \tag{4}$$

Tackle the point-wise contraints using integration

- With $w(t), \delta(t)$ defined in (Cvg-cdt), we introduce

$$p(x, \bar{x}, \Xi, t, f) = \Bigg[\beta(t)\sqrt{\int_0^1 \Lambda((1-\tau)x + \tau\bar{x}, f)\,\mathrm{d}\tau} - \sqrt{\gamma(t) - \dot{\beta}(t)} - \sqrt{\gamma(t) - \dot{\beta}(t) - \frac{\alpha}{t}\beta(t)}\Bigg]_+,$$

$$q(\Xi, t) = \Big[\gamma(t) - \dot{\beta}(t) - \beta(t)/h\Big]_+ + \Big[\dot{\beta}(t) + \alpha\beta(t)/t - \gamma(t)\Big]_+ + \Big[\dot{\delta}(t) - \lambda t w(t)\Big]_+ + [-\delta(t)]_+ .$$

- Setting $P, Q \leq 0$ ensures (Cvg-cdt) and (STB-CDT) hold for f

$$P(\Xi, f) = \int_{t_0}^{T(\Xi, f)} p(X(t, \Xi, f), \bar{x}(t), \Xi, t, f)\,\mathrm{d}t, \quad Q(\Xi, f) = \int_{t_0}^{T(\Xi, f)} q(\Xi, t)\,\mathrm{d}t$$

A L2O framework for selecting the best coefficients

- Given a random variable $\xi \sim \mathbb{P}$. We say $\mathbb{P}$ is the induced probability of $f(\cdot;\xi)$

$$\mathbb{E}_f[T(\Xi, f)] = \int_{\xi} T(\Xi, f(\cdot;\xi))\,\mathrm{d}\mathbb{P}(\xi) = \mathbb{E}_{\xi}[T(\Xi, f(\cdot;\xi))]$$

- Framework: minimize the expectation of stopping time under conditions of convergence and stable discretization

$$\begin{aligned} \min_{\Xi} \quad & \mathbb{E}_f[T(\Xi, f)], \\ \text{s.t.} \quad & \mathbb{E}_f[P(\Xi, f)] \le 0, \quad \mathbb{E}_f[Q(\Xi, f)] \le 0, \end{aligned}$$

- **Parameterization:** $\beta \to \beta_{\theta_1}, \gamma \to \gamma_{\theta_2}$. Set $\theta = (\alpha, \theta_1, \theta_2)$.
- Given the penalty parameter ρ, the ℓ_1 exact penalty problem writes

$$\begin{aligned} \min_{\theta} \ \Upsilon(\theta) &= \mathbb{E}_f[T(\theta, f)] + \rho\left(\mathbb{E}_f[P(\theta, f)] + \mathbb{E}_f[Q(\theta, f)]\right) \\ &= \mathbb{E}_f\left[T(\theta, f) + \rho\left(P(\theta, f) + Q(\theta, f)\right)\right] \end{aligned}$$

Conservative gradient

- When parameterize α, β, γ using neural networks, they may be nonsmooth
- The output of *auto differentiation* in nonsmooth functions may not be Clarke subdifferentials, but they are certainly conservative gradients
- Conservative gradient generalizes subdifferentials while preserving chain rule
- Ψ is termed the conservative Jacobian (gradient if $m = 1$) of π if and only if

$$\frac{\mathrm{d}}{\mathrm{d}\iota}\pi(r(\iota)) = A\dot{r}(\iota), \quad \text{for all } A \in \Psi(r(\iota)), \text{ for almost all } \iota \in [0,1],$$

  for any absolutely continuous curve $r : [0,1] \to \mathbb{R}^d$
- Consider the example:

$$f(s) = ([-s]_+ + s) - [s]_+ \equiv 0 \qquad \underset{\text{autograd using TensorFlow}}{\Longrightarrow} \qquad g(s) = \begin{cases} 0 & (s \neq 0) \\ 1 & (s = 0) \end{cases}$$

  g is not the Clarke subdifferential of f but a conservative gradient

Evaluate the derivative of stopping time: $\nabla_\Xi T(f,\Xi)$

- Take limit by continuity: $\|\nabla f(X(T(f,\Xi),f,\Xi))\|^2-\varepsilon^2\equiv 0$
- Implicit function theorem (valid in nonsmooth case):

$$\nabla f(X)^\top\nabla^2 f(X)\left(\left.\frac{\partial X}{\partial t}\right|_{t=T}\nabla_\Xi T(f,\Xi)+\frac{\partial X}{\partial \Xi}\right)=0$$

  where $T=T(f,\Xi), X=X(T(f,\Xi),f,\Xi)$
- Invoking the first-order form of (DIN-AVD):

$$\left.\frac{\partial X}{\partial t}\right|_{t=T}=\dot{x}(T)=v(T)-x(T)-\beta(T)\nabla f(x(T))$$

  where $x(t)=X(t,f,\Xi)$
- The derivative:

$$\nabla_\Xi T(f,\Xi)=\Big(\nabla f(X)^\top\nabla^2 f(X)\left(v(T)-X-\beta(T)\nabla f(X)\right)\Big)^{-1}\nabla f(X)^\top\nabla^2 f(X)\frac{\partial X}{\partial \Xi}$$

Conservative gradient of the constraints

- To ease our presentation, we introduce

$$p(x,\Xi,t,f)=\left[\beta_{\Xi_1}(t)\sqrt{\Lambda(x,f)}-\sqrt{\gamma_{\Xi_2}(t)-\dot{\beta}_{\Xi_1}(t)}-\sqrt{\gamma_{\Xi_2}(t)-\dot{\beta}_{\Xi_1}(t)-\frac{\alpha}{t}\beta_{\Xi_1}(t)}\right]_+$$

$$q(\Xi,t)=\left[\gamma_{\Xi_2}(t)-\dot{\beta}_{\Xi_1}(t)-\beta_{\Xi_1}(t)/h\right]_+ + \left[\dot{\delta}_{\Xi}(t)-tw_{\Xi}(t)\right]_+ + \left[-\delta_{\Xi}(t)\right]_+$$

- The constraints P, Q can be represented as

$$P(\Xi,f)=\int_{t_0}^{T(\Xi,f)} p(X(\Xi,t,f),\Xi,t,f)\,\mathrm{d}t, \qquad Q(\Xi,f)=\int_{t_0}^{T(\Xi,f)} q(\Xi,t)\,\mathrm{d}t$$

- Applying the chain rule gives

$$\nabla_{\Xi}P(\Xi,f)=p(X(\Xi,T,f),\Xi,T,f)\nabla_{\Xi}T+\int_{t_0}^{T}\left.\frac{\partial p}{\partial x}\right|_{x=X}\frac{\partial X}{\partial \Xi}+\frac{\partial p}{\partial \Xi}\,\mathrm{d}t$$

$$\nabla_{\Xi}Q(\Xi,f)=q(\Xi,T)\nabla_{\Xi}T+\int_{t_0}^{T}\frac{\partial q}{\partial \Xi}\,\mathrm{d}t$$

SGD converges with (nonsmooth) auto-differentiation: Assumptions

Assumption (Assumptions of the SGD)

1. *The step sizes* $\{\eta_k\}_{k\geq 1}$ *satisfy*

$$\eta_k \geq 0, \quad \sum_{k=1}^{\infty} \eta_k = \infty, \quad \text{and} \quad \sum_{k=1}^{\infty} \eta_k^2 < \infty.$$

2. *Almost surely, the iterates* $\{\Xi_k\}_{k\geq 1}$ *are bounded, i.e.,* $\sup_{k\geq 1} \|\Xi_k\| < \infty$.
3. $\{\xi_k\}_{k\geq 1}$ *is a uniformly bounded difference martingale sequence with respect to the increasing* σ*-fields*

$$\mathscr{F}_k = \sigma(\Xi_j, \varrho_j, \xi_j : j \leq k).$$

In other words, there exists a constant $M_\xi > 0$ *such that*

$$\mathbb{E}[\xi_k \mid \mathscr{F}_k] = 0 \quad \text{and} \quad \mathbb{E}[\|\xi_k\|^2 \mid \mathscr{F}_k] \leq M_\xi \quad \text{for all} \quad k \geq 1.$$

SGD converges with (nonsmooth) auto-differentiation

Assumption

The complementary of $\{\Upsilon(\Xi) \mid 0 \in \mathcal{J}_\Upsilon(\Xi)\}$ *is dense in* $\mathbb{R}$.

Theorem (SGD converges using conservative gradient)

Suppose that Assumptions 1 and 2 hold. Then every limit point of $\{\Xi_k\}_{k\geq 1}$ *is stationary and the function values* $\{\varkappa(\Xi_k)\}_{k\geq 1}$ *converge.*

Theorem (Convergence guarantee)

Suppose Assumptions hold, $\{\Xi_k\}_{k\geq 1}$ *is generated by the Algorithm. Then almost surely, every limit point* $\Xi_\star$ *of* $\{\Xi_k\}_{k\geq 1}$ *satisfies* $\Xi_\star \in S_\mathcal{D}$, $0 \in \mathcal{J}_\Upsilon(\Xi_\star)$ *and the sequence* $\{\Upsilon(\Xi_k)\}_{k\geq 1}$ *converges.*

Setting and datasets

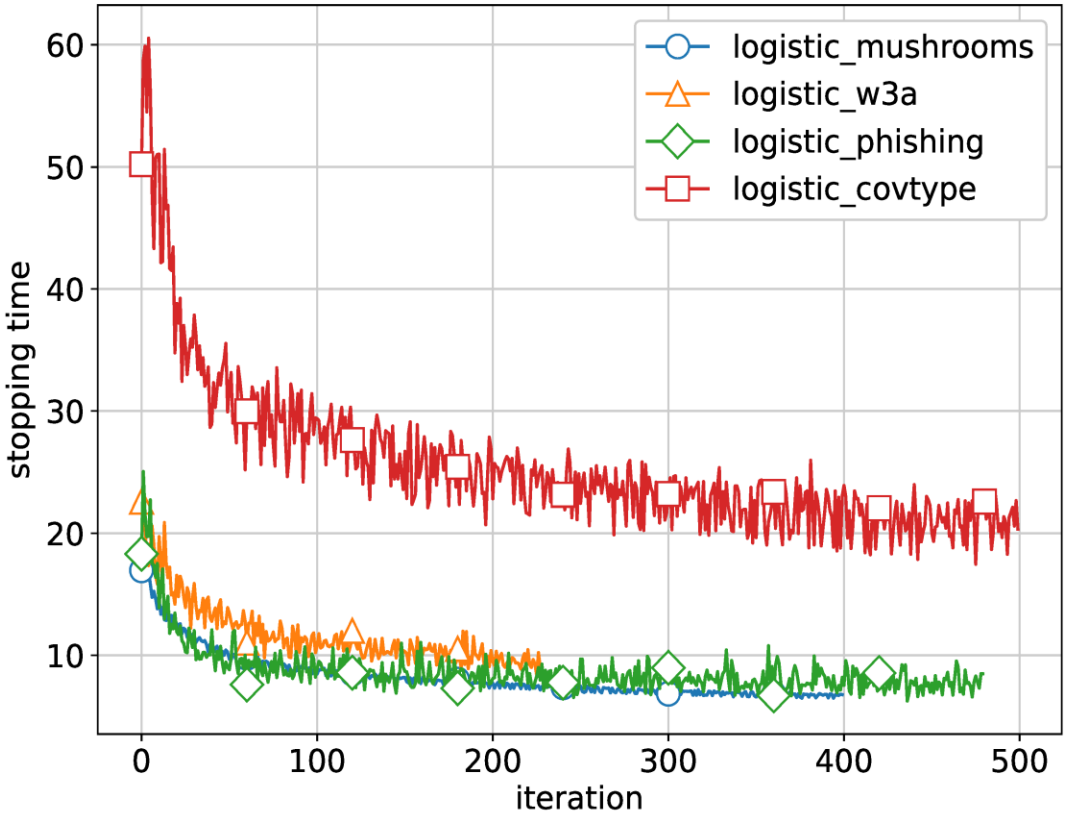
- Consider the logistic regression problem defined by a finite set $\mathscr{D}$, a subset of a given dataset or is sampled from a distribution

$$\min_{x\in\mathbb{R}^n} f_{\mathscr{D}}(x) = \frac{1}{|\mathscr{D}|}\sum_{(a_i,b_i)\in\mathscr{D}} \log(1+\exp(-b_i\langle a_i, x\rangle)),$$

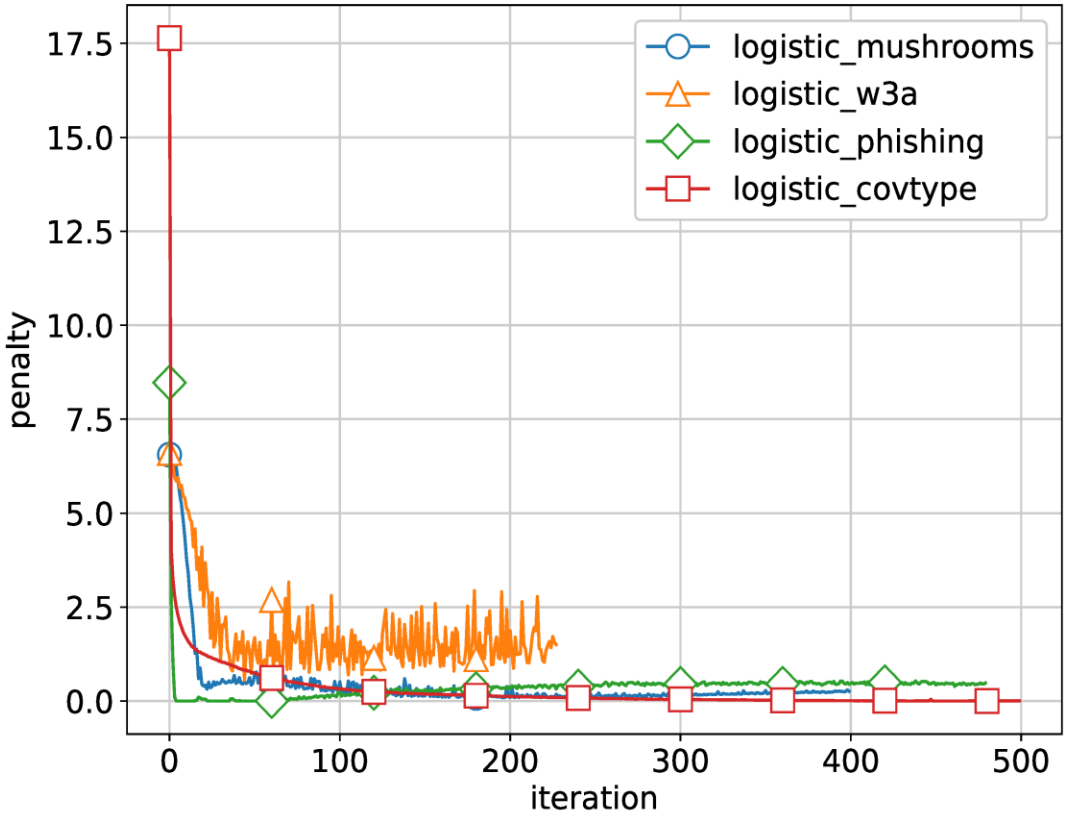
where the data pairs $\{a_i, b_i\} \in \mathbb{R}^n \times \{0,1\}, i \in [|\mathscr{D}|]$

Dataset	n	N_{train}	N_{test}	Separable
a5a	123	6,414	26,147	No
w3a	300	4,912	44,837	No
mushrooms	112	3,200	4,924	Yes
covtype	54	102,400	478,612	No
phishing	68	8,192	2,863	No
separable	101	20,480	20,480	Yes

Training results



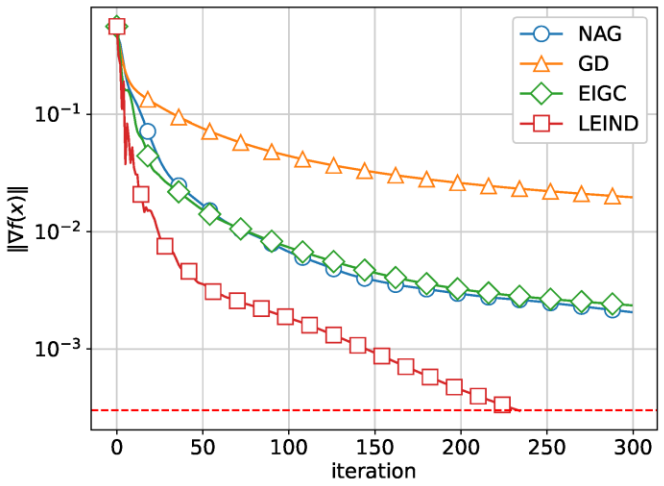
(c) Stopping time on logistic regression



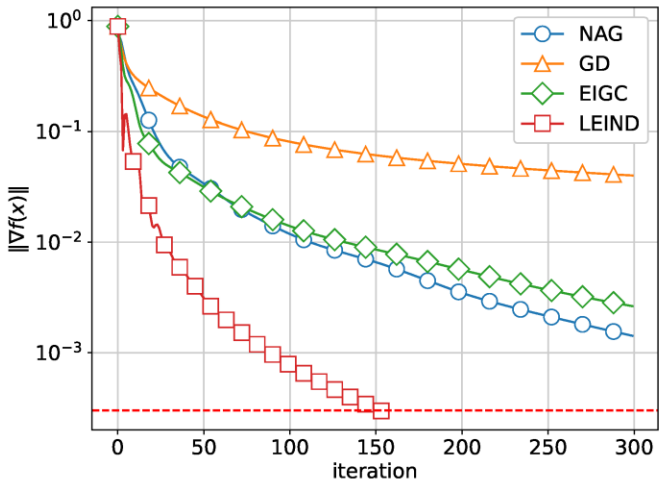
(d) Penalty on logistic regression

Figure: The training process in different tasks.

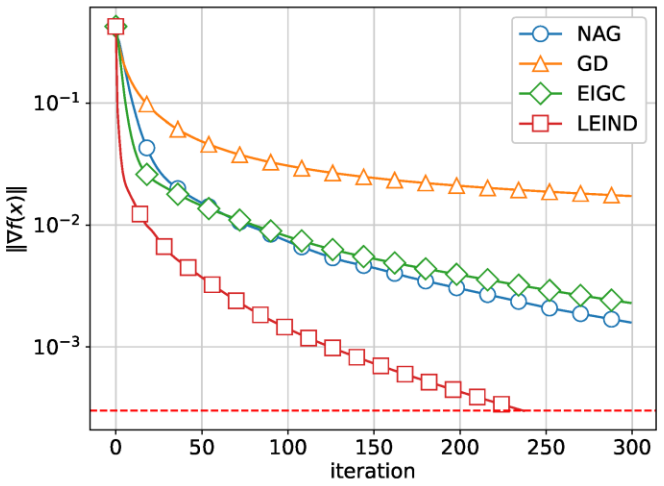
Testing results



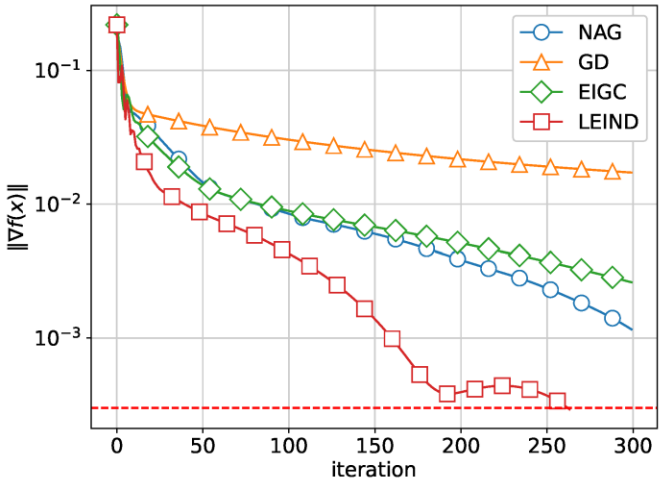
(a) a5a

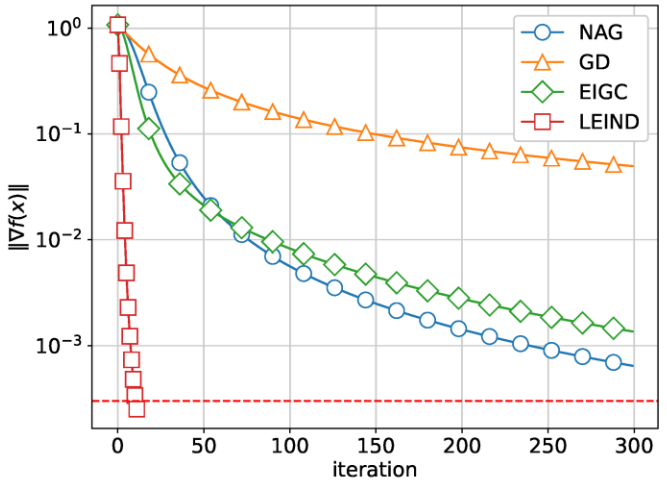


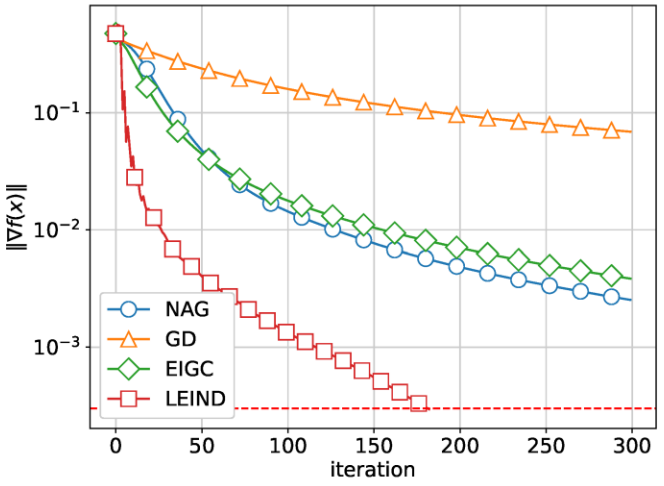
(b) mushrooms



(c) w3a







Outline

Binary Optimization

Let f be arbitrary (even non-smooth) cost function:

$$\min \quad f(x), \quad \text{s.t.} \quad x \in \mathcal{B}_n = \{-1, 1\}^n.$$

- Example: maxcut problem on $G = (V, E)$

$$\max \quad \sum_{(i,j)\in E} w_{ij}(1 - x_i x_j), \quad \text{s.t.} \quad x \in \{-1, 1\}^n.$$

- Example: MaxSAT problem:

$$\begin{aligned} \max_{x\in\{-1,1\}^n} \quad & \sum_{c^i \in C_1} \max\{c_1^i x_1, c_2^i x_2, \cdots, c_n^i x_n, 0\}, \\ \text{s.t.} \quad & \max\{c_1^i x_1, c_2^i x_2, \cdots, c_n^i x_n, 0\} = 1, \quad \text{for } c^i \in C_2 \end{aligned}$$

- Binary optimization is NP-hard due to the combinatorial structure.

Probabilistic Approach

- Let $\mathcal{X}^*$ be the set of optimal solutions and consider the distribution,

$$q^*(x) = \frac{1}{|\mathcal{X}^*|}\mathbf{1}_{\mathcal{X}^*}(x) = \begin{cases} \frac{1}{|\mathcal{X}^*|}, & x \in \mathcal{X}^*, \\ 0, & x \notin \mathcal{X}^*. \end{cases}$$

- To approximate q^*, we introduce Gibbs distributions

$$q_\lambda(x) = \frac{1}{Z_\lambda}\exp\left(-\frac{f(x)}{\lambda}\right) \to \frac{1}{|\mathcal{X}^*|}\mathbf{1}_{\mathcal{X}^*}(x) = q^*, \quad \text{as } \lambda \to 0 \quad x \in \mathcal{B}_n,$$

where $Z_\lambda = \sum_{x\in\mathcal{B}_n}\exp\left(-\frac{f(x)}{\lambda}\right)$ is the normalizer.

Parameterized Probabilistic Model

- KL divergence:

$$\mathrm{KL}\left(p_\theta \,\|\, q_\lambda\right) = \sum_{x \in \mathcal{B}_n} p_\theta(x) \log \frac{p_\theta(x)}{q_\lambda(x)}.$$

- In order to reduce the discrepancy between p_θ and q_λ, the KL divergence is supposed to be minimized:

$$\begin{aligned}\mathrm{KL}\left(p_\theta \,\|\, q_\lambda\right) &= \frac{1}{\lambda} \sum_{x \in \mathcal{B}_n} p_\theta(x) f(x) + \sum_{x \in \mathcal{B}_n} p_\theta(x) \log p_\theta(x) + \log Z_\lambda \\ &= \frac{1}{\lambda} \left(\mathbb{E}_{p_\theta}\left[f(x)\right] + \lambda \mathbb{E}_{p_\theta}\left[\log p_\theta(x)\right]\right) + \log Z_\lambda.\end{aligned}$$

- Loss Function (Z_λ is a constant):

$$\boxed{\min_\theta \quad L_\lambda(\theta) = \mathbb{E}_{p_\theta}\left[f(x)\right] + \lambda \mathbb{E}_{p_\theta}\left[\log p_\theta(x)\right]}$$

Gradient for the Loss Function

Lemma

Suppose for any $x \in \mathcal{B}_n$, $p_\theta(x)$ is differentiable with respect to θ. For any constant $c \in \mathbb{R}$, we denote the advantage function

$$A_\lambda(x;\theta,c) := f(x) + \lambda \log p_\theta(x) - c.$$

Then, the gradient of the loss function is given by

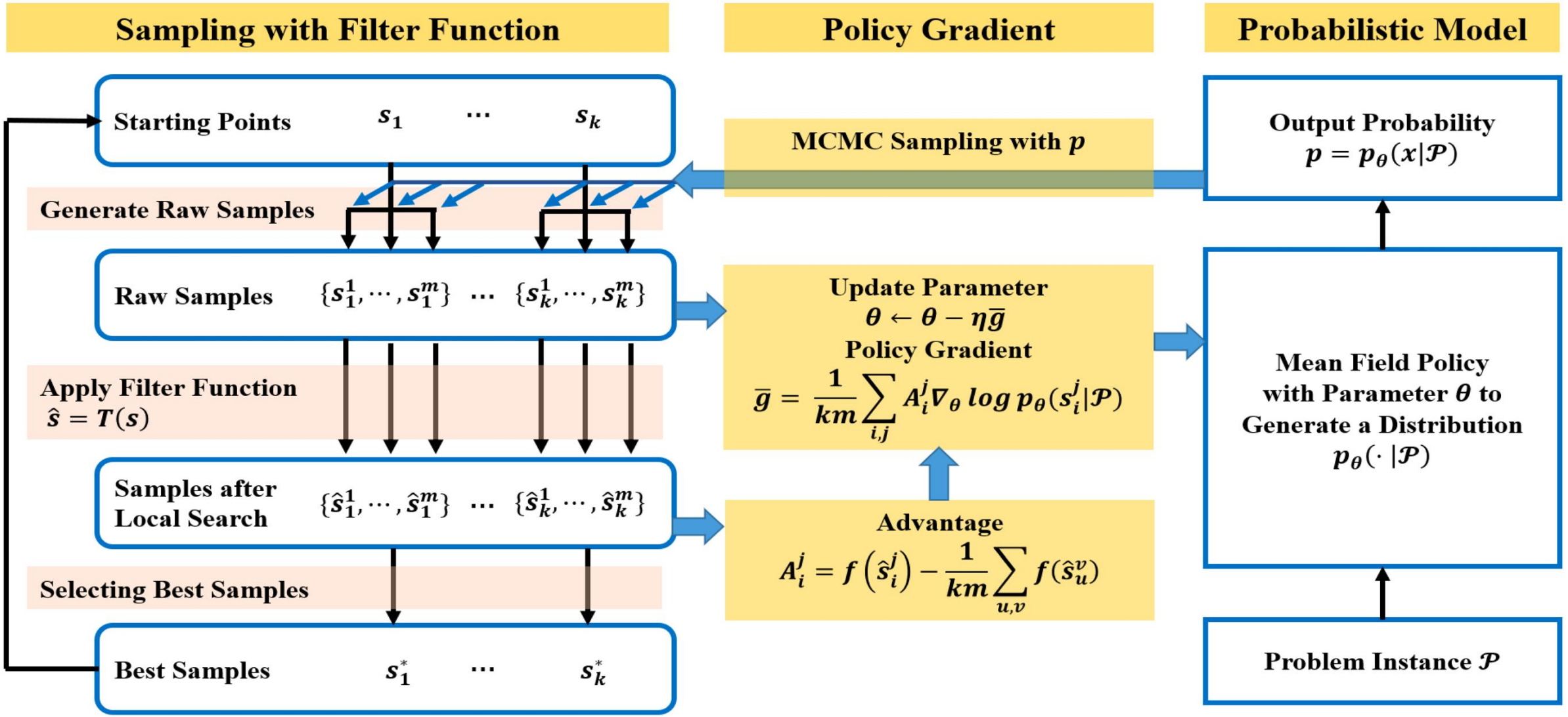
$$\nabla_\theta L_\lambda(\theta) = \mathbb{E}_{p_\theta}\left[A_\lambda(x;\theta,c)\nabla_\theta \log p_\theta(x)\right].$$

One candidate for c is

$$c = \mathbb{E}_{p_\theta}[f(x)].$$

Very similar to the policy gradient in reinforcement learning!

Pipeline of MCPG



Filter Function

Definition (Filter Function)

For each $x \in \mathcal{B}_n$, let $\mathcal{N}(x) \subset \mathcal{B}_n$ be a neighborhood of x such that $x \in \mathcal{N}(x)$, $|\mathcal{N}(x)| \geq 2$ and any point in $\mathcal{N}(x)$ can be reached by applying a series of "simple" operations to x. A filter function $T(x)$ is defined as

$$T(x) \in \arg\min_{\hat{x} \in \mathcal{N}(x)} f(\hat{x}),$$

where $T(x)$ is arbitrarily chosen if there exists multiple solutions.

- Projection to the best solution on the neighborhood:

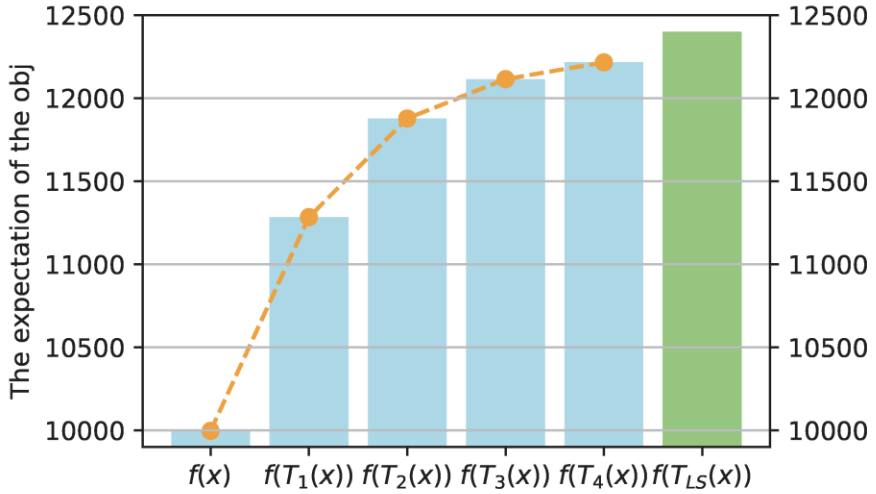
$$T_k(x) = \arg\min_{\|\hat{x}-x\|_1 \leq 2k} f(\hat{x}), \quad \mathcal{N}(x) = \{\hat{x} \mid \|\hat{x} - x\|_1 \leq 2k\}.$$

- Algorithms serves as the filter function: $T_{LS}(x) = \texttt{LocalSearch}_f(x)$.

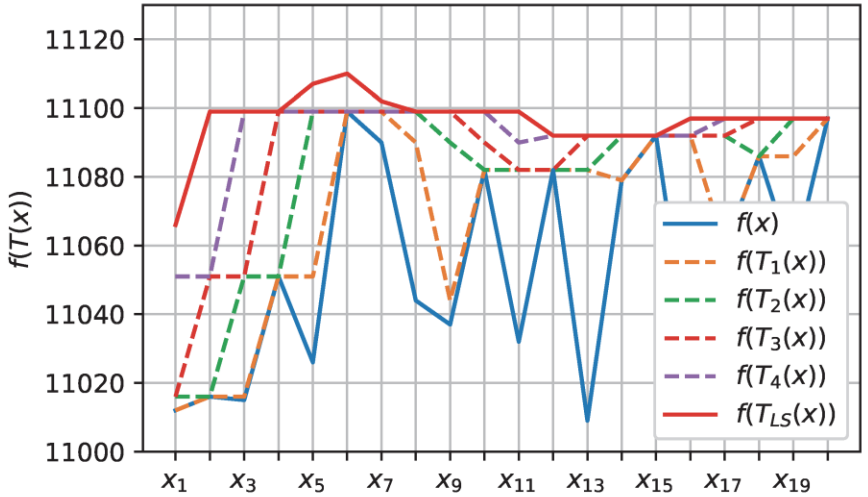
Filter Function

- The filter function T projects x to a better one in the neighborhood.
- Applied with the filter function, $f(T(x))$ has fewer local minima and the same global minimum as the original one.

maxcut, G22:



(g) Expectation of the objective function.



(h) A selected sequence of solutions.

Assumptions

Regularity conditions on policy: Let $\phi(x;\theta) = \log p_\theta(x|\mathcal{P})$. There exists some constants $M_1, M_2, M_3 > 0$ such that, for any $x \in \mathcal{B}_n$,

1. $\sup_{\theta\in\mathbb{R}^n} |\phi(x;\theta)| \le M_1$,
2. $\sup_{\theta\in\mathbb{R}^n} \|\nabla_\theta \phi(x;\theta)\| \le M_2$,
3. $\|\nabla_{\theta_1}\phi(x;\theta) - \nabla_{\theta_2}\phi(x;\theta)\| \le M_3 \|\theta_1 - \theta_2\|, \forall \theta_1, \theta_2, \in \mathbb{R}^n$.

Spectral gap: Let P be the transition matrix of a finite-state time homogenous Markov chain. Then, the spectral gap of P is defined as

$$\gamma(P) := \frac{1 - \max\{|\lambda_2(P)|, |\lambda_{2^n}(P)|\}}{2} \in (0, 1],$$

where $\lambda_i(P)$ is the i-th largest eigenvalue of the matrix P. We assume that there exists a positive lower bound $\gamma(P_\theta) \ge \gamma > 0$ for any $\theta \in \mathbb{R}^n$.

Convergence of MCPG

Theorem

Let the above assumptions hold and $\{\theta^t\}$ be generated by MCPG. If the stepsize satisfies $\eta^t \leq \frac{1}{2L}$, then for any τ, we have

$$\min_{1\leq t\leq \tau} \mathbb{E}\left[\left\|\nabla_\theta L_\lambda(\theta^t)\right\|^2\right] \leq O\left(\frac{1}{\sum_{t=1}^{\tau}\eta^t} + \frac{\sum_{t=1}^{\tau}(\eta^t)^2}{mk\sum_{t=1}^{\tau}\eta^t} + \frac{1}{m^2}\right). \tag{5}$$

In particular, if the stepsize is chosen as $\eta^t = \frac{c\sqrt{mk}}{\sqrt{t}}$ with $c \leq \frac{1}{2l}$, then we have

$$\min_{1\leq t\leq \tau} \mathbb{E}\left[\left\|\nabla_\theta L_\lambda(\theta^t)\right\|^2\right] \leq O\left(\frac{\log\tau}{\sqrt{mk\tau}} + \frac{1}{m^2}\right). \tag{6}$$

- This theorem shows that the probabilistic model of MCPG will finally converge to stationary points.

Parameterization of sampling policy

- Mean field (MF) approximation:

$$p_\theta(x|\mathcal{P}) = \prod_{i=1}^{n} \mu_i^{(1+x_i)/2}(1-\mu_i)^{(1-x_i)/2}, \quad \mu_i = \phi_i(\theta;\mathcal{P})$$

- Parameterization of μ_i:

$$\mu_i = \phi_i(\theta_i) = \frac{1-2\alpha}{1+\exp(-\theta_i)} + \alpha, \quad 1 \le i \le n.$$

  The probability is scaled to the range $(\alpha, 1-\alpha)$, where $0 < \alpha < 0.5$ is given.

- For problems graph structures, combining advanced neural networks such as GNN can also be a good choice.

Maxcut

- The maxcut problem aims to divide a given weighted graph $G = (V, E)$ into two parts, and maximize the total weight of the edges connecting two parts.
- This problem can be expressed as a binary programming problem:

$$\max \quad \sum_{(i,j)\in E} w_{ij}(1 - x_i x_j), \quad \text{s.t.} \quad x \in \{-1, 1\}^n.$$

- We use the results reported by BLS as benchmark. Denoting UB as the results achieved by BLS and obj as the cut size, the gap reported is defined as follows:

$$\mathrm{gap} = \frac{\mathrm{UB} - \mathrm{obj}}{\mathrm{UB}} \times 100\%.$$

Maxcut

- On the Gset instance, MCPG finds all the best-known results.
- For G55 and G70, the results obtained by MCPG is better than all the previous reported results.

Graph			BLS	**MCPG**	DSDP	RUN-CSP	PI-GNN	**MCPG** (limited time)	
name	nodes	edges						gap	time
G14	800	4,694	**3,064**	**3,064**	2,922	2,943	3,026	0.02%	28
G15	800	4,661	**3,050**	**3,050**	2,938	2,928	2,990	0.01%	28
G22	2,000	19,990	**13,359**	**13,359**	12,960	13,028	13,181	0.10%	55
G49	3,000	6,000	**6,000**	**6,000**	**6,000**	**6,000**	5,918	0.00%	107
G50	3,000	6,000	**5,880**	**5,880**	**5,880**	**5,880**	5,820	0.03%	107
G55	5,000	12,468	10,294	**10,296**	9,960	10,116	10,138	0.47%	145
G70	10,000	9,999	9,541	**9595**	9,456	-	9,421	0.86%	274

Table: Computational results on selected Gset instances. The result is sourced from references.

Outline

What kind of reasoning are needed?

A typical real scenario: $\min_x \ f(x)$

Symbolic Computation

- Manipulates mathematical symbols and expressions directly.
- Software:
  1. Mathematica
  2. Maple
  3. SymPy
- Demo: computing the gradient of $f(x)$.
  code: D[f, x]

Numerical Computation

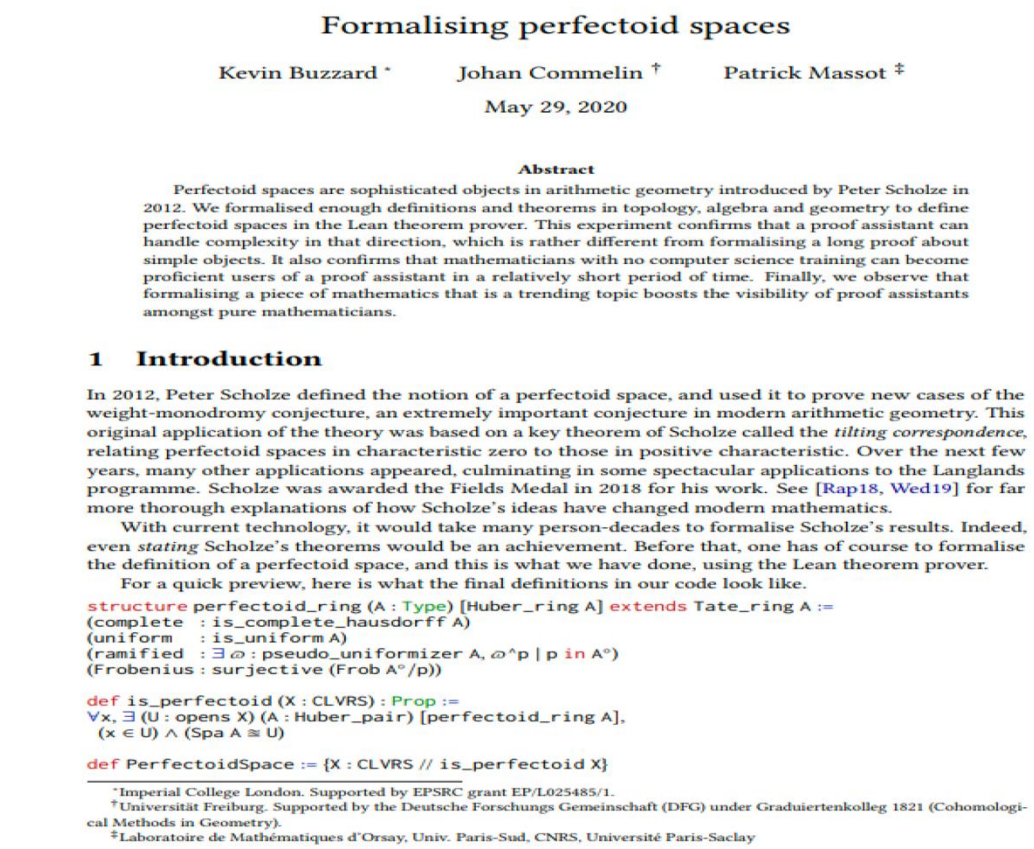
- Approximates solutions to mathematical problems using numerical techniques.
- Software:
  1. MATLAB
  2. NumPy (Python)
- Demo: find a local minimizer using the gradient method.
  code: x=fminunc(fun,x0)

Formalism

- Emphasizes rigorous mathematical proofs and structures.
- Software:
  1. Coq
  2. Isabella
  3. Lean
- Demo: prove the convergence rate of the gradient method

```
theorem gradient_method {alg : Gradient_Descent_fix_stepsize f f' x₀} (hfun:
  ConvexOn ℝ Set.univ f) :
∀ k : ℕ, f (alg.x (k + 1)) - f xm ≤ 1 / (2 * (k + 1) * alg.a) * ‖ x₀ - xm ‖
  ^ 2 := by
```

Exciting Events



Formalising perfectoid spaces

Kevin Buzzard * Johan Commelin † Patrick Massot ‡

May 29, 2020

Abstract

Perfectoid spaces are sophisticated objects in arithmetic geometry introduced by Peter Scholze in 2012. We formalised enough definitions and theorems in topology, algebra and geometry to define perfectoid spaces in the Lean theorem prover. This experiment confirms that a proof assistant can handle complexity in that direction, which is rather different from formalising a long proof about simple objects. It also confirms that mathematicians with no computer science training can become proficient users of a proof assistant in a relatively short period of time. Finally, we observe that formalising a piece of mathematics that is a trending topic boosts the visibility of proof assistants amongst pure mathematicians.

1 Introduction

In 2012, Peter Scholze defined the notion of a perfectoid space, and used it to prove new cases of the weight-monodromy conjecture, an extremely important conjecture in modern arithmetic geometry. This original application of the theory was based on a key theorem of Scholze called the *tilting correspondence*, relating perfectoid spaces in characteristic zero to those in positive characteristic. Over the next few years, many other applications appeared, culminating in some spectacular applications to the Langlands programme. Scholze was awarded the Fields Medal in 2018 for his work. See [Rap18, Wed19] for far more thorough explanations of how Scholze's ideas have changed modern mathematics.

With current technology, it would take many person-decades to formalise Scholze's results. Indeed, even *stating* Scholze's theorems would be an achievement. Before that, one has of course to formalise the definition of a perfectoid space, and this is what we have done, using the Lean theorem prover.

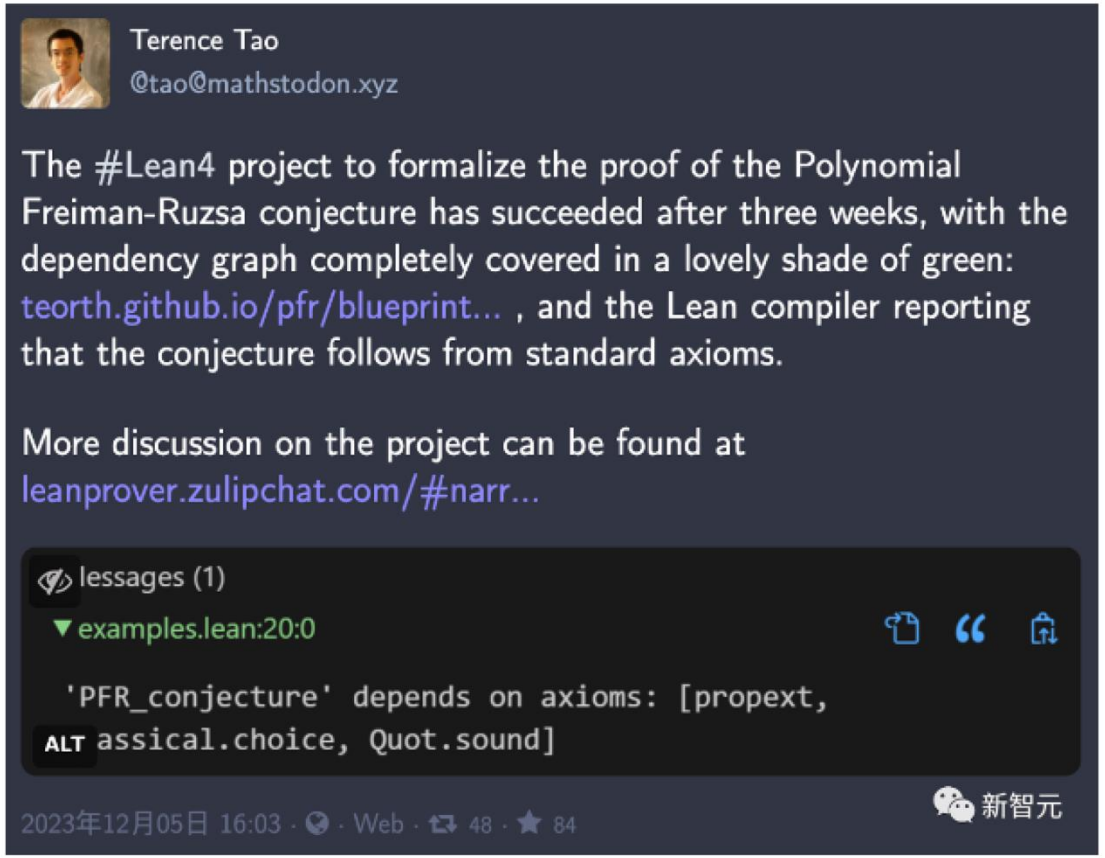
For a quick preview, here is what the final definitions in our code look like.

```
structure perfectoid_ring (A : Type) [Huber_ring A] extends Tate_ring A :=
(complete  : is_complete_hausdorff A)
(uniform   : is_uniform A)
(ramified  : ∃ ϖ : pseudo_uniformizer A, ϖ^p | p in A°)
(Frobenius : surjective (Frob A°/p))

def is_perfectoid (X : CLVRS) : Prop :=
∀x, ∃ (U : opens X) (A : Huber_pair) [perfectoid_ring A],
  (x ∈ U) ∧ (Spa A ≅ U)

def PerfectoidSpace := {X : CLVRS // is_perfectoid X}
```

*Imperial College London.
†Universität Freiburg.
‡Laboratoire de Mathématiques d'Orsay, Univ. Paris-Sud, CNRS, Université Paris-Saclay

*Supported by EPSRC grant EP/L025485/1.
†Supported by the Deutsche Forschungs Gemeinschaft (DFG) under Graduiertenkolleg 1821 (Cohomological Methods in Geometry).

(i) Kevin Buzzard and others successfully formalized Peter Scholze's theory of perfectoid spaces



(j) Terence Tao and others successfully formalized the proof of the Polynomial Freiman-Ruzsa conjecture in about three weeks

Formalizing mathematical optimization: optlib

Goal: a library of formalized theorems of mathematical optimization
Webpage: `https://github.com/optsuite/optlib`
Current progress:

- definition and properties of Gateaux derivative (gradient)
- basic properties of convex functions and L-smooth functions
- the definition of subgradient and proximal operator for nonsmooth convex functions
- optimality conditions for differentiable optimization problems
- convergence analysis of optimization algorithms
  - subgradient method
  - gradient methods
  - proximal gradient method
  - Nesterov's acceleration methods

Formalization of the proximal gradient method

- Consider the composite optimization problem $\min \psi(x) = f(x) + g(x)$.
- the update scheme of proximal gradient descent

$$x_{k+1} = \mathrm{prox}_{\alpha_k g}(x_k - \alpha_k \nabla f(x_k)).$$

```
class proximal_gradient_method (f h: E → ℝ) (f' : E → E) (x₀ : E) :=
  (xm : E) (t : ℝ) (x : ℕ → E) (L : NNReal)
  (fconv : ConvexOn ℝ univ f) (hconv : ConvexOn ℝ univ h)
  (h₁ : ∀ x₁ : E, HasGradientAt f (f' x₁) x₁) (h₂ : LipschitzWith L f')
  (h₃ : ContinuousOn h univ) (minphi : IsMinOn (f + h) Set.univ xm)
  (tpos : 0 < t) (step : t ≤ 1 / L) (ori : x 0 = x₀) (hL : L > (0 : ℝ))
  (update : ∀ k, prox_prop (t • h) (x k - t • f' (x k)) (x (k + 1)))
```

- Convergence rate

$$\psi(x^k) - \psi^* \leq \frac{1}{2kt}\|x^0 - x^*\|^2.$$

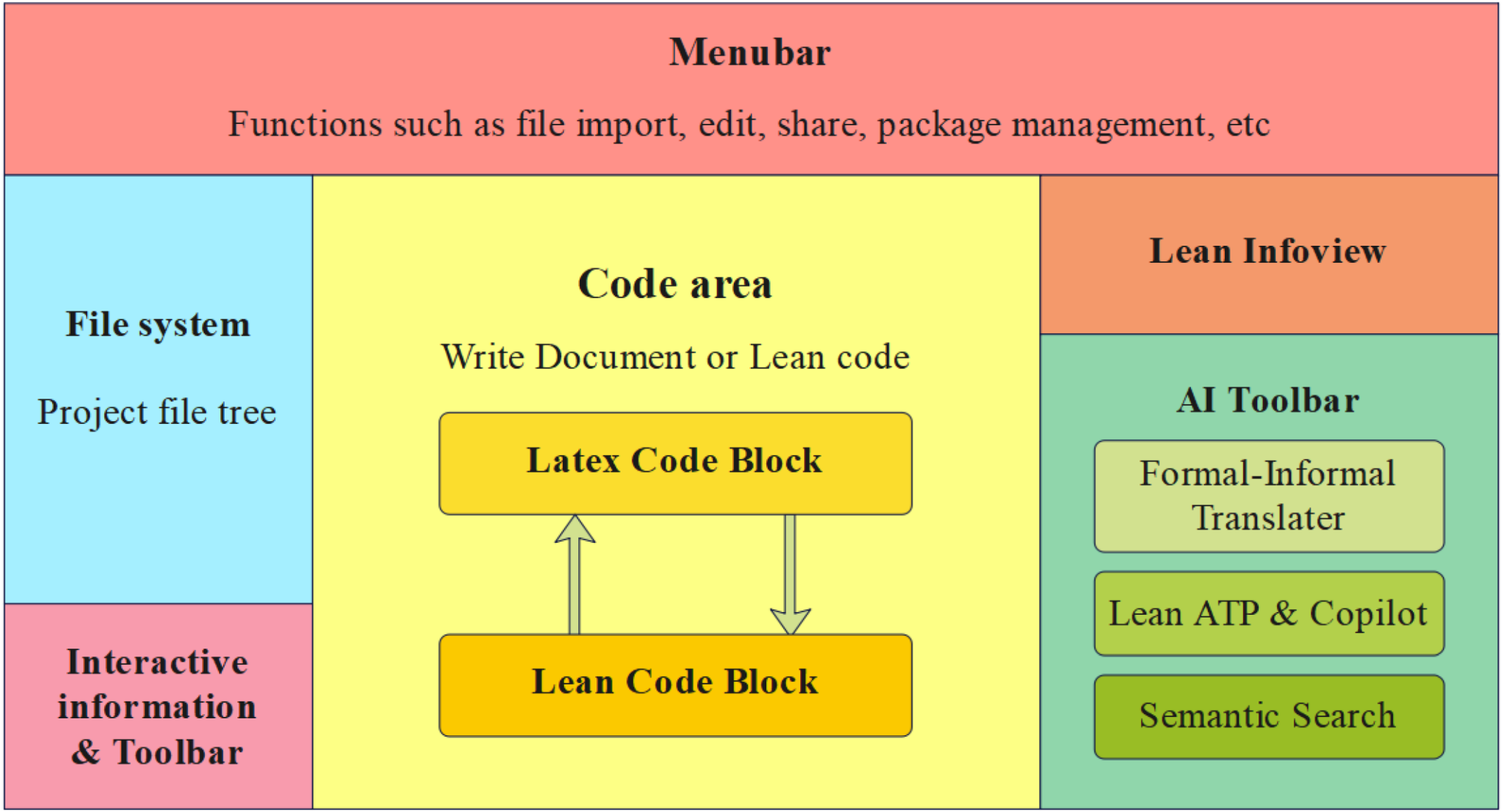
```
theorem proximal_gradient_method_converge {alg : proximal_gradient_method f h f' x₀} :
    ∀ (k : ℕ+), f (alg.x k) + h (alg.x k) - f alg.xm - h alg.xm
    ≤ 1 / (2 * k * alg.t) * ‖x₀ - alg.xm‖ ^ 2 := by
```

Formalize the Core disciplines in applied mathematics

General recipe: **data, modelling, algorithms, analysis, applications**

- scientific computing: numerical analysis, numerical linear algebra, optimization, numerical methods for ODE and PDE, ...
- probability, statistics, stochastic process
- control theory
- combinatorics, graph theory
- operations research, mangagement sciences
- machine Learning and artificial Intelligence
- information theory, signal processing, image processing
- ...

ReasLab: smart tool powered by artificial intelligence beyond formalization



Many Thanks For Your Attention!

- We are hiring: faculty, postdoc, graduate students, engineers
  Competitive salary as U.S and Europe
- Sino-Russian Mathematics Center, Peking University
- Beijing International Center for Mathematical Research, Peking University
- `http://faculty.bicmr.pku.edu.cn/~wenzw`
- E-mail: wenzw@pku.edu.cn
- Office phone: 86-10-62744125