A Comprehensive Overview of Bayesian Optimization Motivation, Algorithms, and Recent Advances

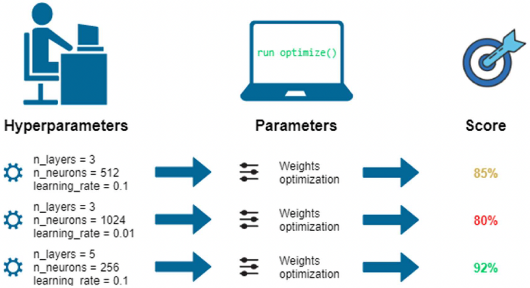
Mohammed Jamal

April 17, 2025

Outline

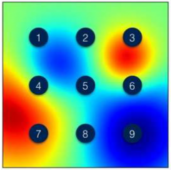
Hyperparameters Optimization

- ML algorithm's performances depend on hyper-parameters.
- Finding the best hyperparameters for the highest performance.

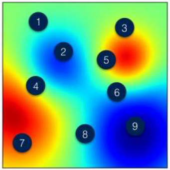


Traditional Hyperparameters Tuning

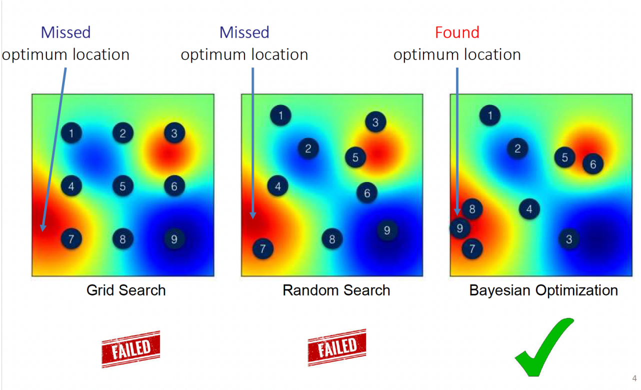
- Grid Search:
  - Create a list of values for each parameter.
  - Consider all possible combinations of these values.
  - Exhaustively evaluate the model and choose the best parameter.



- Random Search:
  - Randomly select a parameter to evaluate.
  - Select the best parameter.

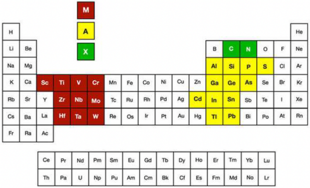


Grid vs Random vs BO



Alloy Development

- Alloy composition: $X = [\%\,Al, \%\,Co, \%Fe, \%\mathrm{Cu}, \%\mathrm{C}\ \ldots]$
- Strength: $\mathbf{y}$
- Goal: find the best composition X for the highest strength y.



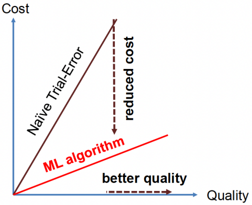
5

Trial error

Trial error approach is used for alloy development using expert knowledge.

Time and cost

- 1 Alloy Testing = 1 day and 100 dollars.
- 100 experiments = 3months and 10 000 dollars.
- Even with 100 experiments, trial-error still can not get the optimum solution



Practical Applications of Bayesian Optimization

Hyperparameter Tuning:

- Optimize learning rate, dropout, architecture parameters.
- Systems such as Google Vizier and Hyperopt are based on BO.

Experimental Design:

- Alloy design, chemical synthesis, or biological experiments.
- Reduces time and cost by selecting experiments wisely.

Robotics and Control:

- Tuning control parameters for bipedal robot design.
- Learning feedback policies in uncertain environments.

Other Examples:

- Neural architecture search.
- Deep reinforcement learning hyperparameter tuning.

Towards Black-Box Optimization

Problem:

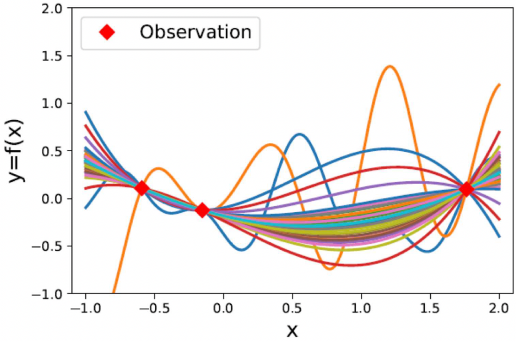
$$x^* = \arg\max_{x \in \mathcal{X}} f(x)$$

where $f(x)$ is unknown and expensive to evaluate.

Black box : only known through evaluation/simulation results: query an evaluation at x_i, observe the result
Question : Where should we evaluate next ?

Surrogate models in BO

1. **Surrogate Modeling:** Define a prior over f (usually a GP).
2. A surrogate model mimics the behaviour of the true function f as closely as possible.
3. surrogate model should be cheap to evaluate.



Gaussian Process Surrogate Model

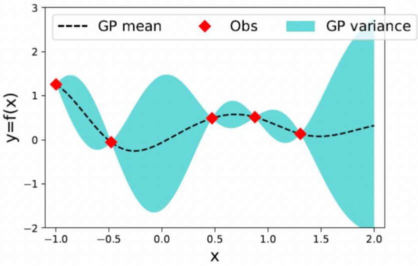
GP Prior:

$$f(x) \sim \mathcal{GP}\big(m(x),\ k(x, x')\big),$$

with:

- Mean function: $m(x)$.
- Covariance function (e.g., RBF kernel):

$$k(x, x') = \sigma_f^2 \exp\left(-\frac{\|x - x'\|^2}{2l^2}\right) .$$



Rasmussen, C. E. Gaussian processes for machine learning, 2006.

Posterior Mean and Variance with Noisy Evaluations

Noisy Evaluations:

$$y_i = f(x_i) + \varepsilon_i, \quad \varepsilon_i \sim \mathcal{N}(0, \sigma_n^2)$$

Posterior Prediction: Given the data $\mathcal{D}_n = \{(x_i, y_i)\}_{i=1}^n$, the GP posterior at a new point x is a normal distribution:

$$f(x) \,|\, \mathcal{D}_n \sim \mathcal{N}\Big(\mu_n(x),\, \sigma_n^2(x)\Big),$$

with

$$\mu_n(x) = k(x, X)[K+\sigma_n^2 I]^{-1} y, \quad \sigma_n^2(x) = k(x,x) - k(x, X)[K+\sigma_n^2 I]^{-1} k(X, x).$$

The $\mu_n(x)$ represents our best estimate of $f(x)$ given the observed (noisy) data, while $\sigma_n^2(x)$ quantifies the uncertainty in our prediction at x.

Bayesian Optimization Algorithm

1. **Input:**
   - Domain $\mathcal{X}$
   - Initial dataset $\mathcal{D}_0 = \{(x_i, y_i)\}_{i=1}^{n_0}$
2. **For** $t = n_0 + 1, n_0 + 2, \ldots, T$ **do:**
   1. Fit a Gaussian Process (GP) model to the dataset $\mathcal{D}_{t-1}$.
   2. Define an acquisition function $a(x)$
   3. Optimize the acquisition function to select
      $$x_t = \arg\max_{x \in \mathcal{X}} a(x).$$
   4. Evaluate
      $$y_t = f(x_t) + \varepsilon_t.$$
   5. Update the dataset with the new observation:
      $$\mathcal{D}_t \leftarrow \mathcal{D}_{t-1} \cup \{(x_t, y_t)\}.$$
3. **Output:**
   $$x^* = \arg\max_{(x,y) \in \mathcal{D}_T} y.$$

Why acquisition function ?

- Based on a GP surrogate above, BO defines an acquisition function $\alpha(x)$ to select a point for evaluation.

instead of $x_t = argmax_{x \in X} f(x)$ $\Rightarrow$ $x_t = argmax_{x \in X} \alpha(x)$

unsolvable! solvable!

- Optimizing the acquisition function α is <u>cheaper</u> without using black-box evaluation.

Explore + Exploit

Expected Improvement (EI) : Mokus, 1972

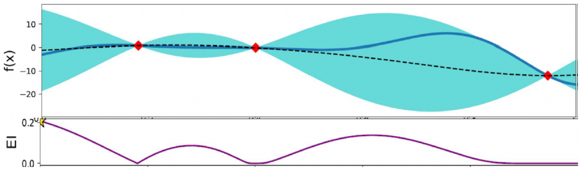
Goal: Maximize expected improvement over current best observed value.

Improvement Function:

$$I(x) = \max\left(f(x) - f(x^{+}) - \xi,\, 0\right)$$

Expected Improvement:

$$\mathrm{EI}(x) = \mathbb{E}[I(x)] == (\mu_n(x) - f(x^{+}) - \xi)\,\Phi(Z) + \sigma_n(x)\,\phi(Z)$$



Intuition: Chooses points with a high chance of improving over the current best.

Probability of Improvement (PI) :Krushner, 1997

Goal: Maximize probability of improving over current best observed value.

Closed form :

$$a_{\mathrm{PI}}(x) = \Phi\left(\frac{\mu_n(x) - f(x^+) - \xi}{\sigma_n(x)}\right)$$

Where:

- Φ: CDF of the standard normal distribution
- $f(x^+) = \max_{i \leq n} y_i$
- $\xi > 0$: optional exploration parameter

- Easy to compute and interpret.
- Often overly greedy — tends to ignore uncertainty.
- Rarely used in practice compared to EI or UCB.

Upper Confidence Bound (UCB) : Srinivas, 2010

Goal: Select points with high mean and/or high uncertainty.

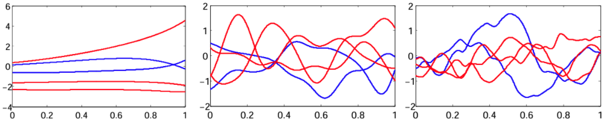
$$a_{\mathrm{UCB}}(x) = \mu_n(x) + \sqrt{\beta_t}\,\sigma_n(x)$$

Theorem 1 *Let* $\delta \in (0,1)$ *and* $\beta_t = 2\log(|D|t^2\pi^2/6\delta)$. *Running GP-UCB with* β_t *for a sample* f *of a GP with mean function zero and covariance function* $k(\boldsymbol{x}, \boldsymbol{x}')$, *we obtain a regret bound of* $\mathcal{O}^*(\sqrt{T\gamma_T \log|D|})$ *with high probability. Precisely,*

$$\Pr\left\{R_T \leq \sqrt{C_1 T \beta_T \gamma_T} \quad \forall T \geq 1\right\} \geq 1 - \delta.$$

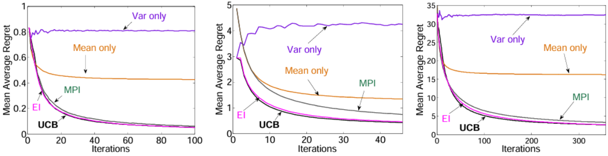
where $C_1 = 8/\log(1+\sigma^{-2})$.

Regret



(a) *Bayesian Linear Regression* (b) *Squared Exponential* (c) *Matérn*

Figure 4. Sample functions drawn from a GP with linear, squared exponential and Matérn kernels ($\nu = 2.5$.)



(a) *Squared exponential* (b) *Temperature data* (c) *Traffic data*

Figure 5. Comparison of performance: GP-UCB and various heuristics on synthetic (a), and sensor network data (b, c).

Acquisition Strategy: Thompson Sampling (TS)

Goal: Sample functions from the posterior and optimize them directly.

Algorithm:

1. Sample $f_t(x) \sim \mathcal{GP}(\mu_n(x), \sigma_n^2(x))$
2. Select:
$$x_t = \arg\max_{x \in \mathcal{X}} f_t(x)$$

Intuition: Naturally balances exploration and exploitation by randomizing the acquisition.

Advantages:

- Simple and effective.
- Competitive theoretical regret bounds.
- Scales well in batch BO (via multiple independent samples).

Bayesian regret

$$\mathrm{BCRT}_T = \mathbb{E}\left[\sum_{t=1}^{T} \left(f(x^*) - f(x_t)\right)\right] \qquad \mathrm{BSRT}_T = \mathbb{E}\left[f(x^*) - \max_{t \le T} f(x_t)\right]$$

Algorithm 2 PIMS

Require: Input space $\mathcal{X}$, GP prior $\mu = 0$ and k, and initial dataset $\mathcal{D}_0$

for $t = 1, \dots$ **do**

  Fit GP to $\mathcal{D}_{t-1}$

  Generate a sample path $g_t \sim p(f \mid \mathcal{D}_{t-1})$

  $g_t^* \leftarrow \max_{\boldsymbol{x} \in \mathcal{X}} g_t$

  $\boldsymbol{x}_t \leftarrow \arg\min_{\boldsymbol{x} \in \mathcal{X}} \frac{g_t^* - \mu_{t-1}(\boldsymbol{x})}{\sigma_{t-1}(\boldsymbol{x})}$

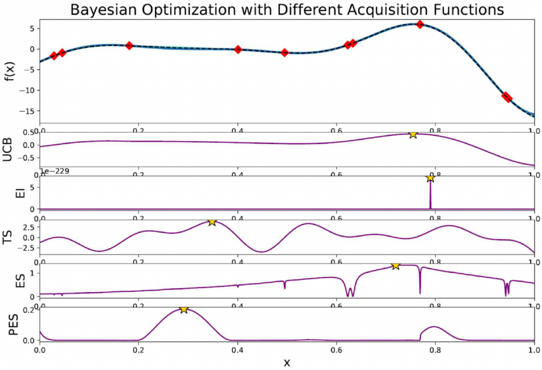
  Observe $y_t = f(\boldsymbol{x}_t) + \epsilon_t$ and $\mathcal{D}_t \leftarrow \mathcal{D}_{t-1} \cup (\boldsymbol{x}_t, y_t)$

end for

	GP-UCB	IRGP-UCB	TS	PIMS
BCR for $\lvert\mathcal{X}\rvert < \infty$	$O(\sqrt{T\gamma_T \log(\lvert\mathcal{X}\rvert T)})$	$O(\sqrt{T\gamma_T \log \lvert\mathcal{X}\rvert})$	* $O(\sqrt{T\gamma_T \log \lvert\mathcal{X}\rvert})$	* $O(\sqrt{T\gamma_T \log \lvert\mathcal{X}\rvert})$
BCR for $\mathcal{X} \subset [0, r]^d$	$O(\sqrt{T\gamma_T \log T})$	$O(\sqrt{T\gamma_T \log T})$	* $O(\sqrt{T\gamma_T \log T})$	* $O(\sqrt{T\gamma_T \log T})$

Table 1: Summary of BCR bounds. The first and second rows show the BCR bounds for the finite and infinite input domains, respectively, where γ_T is the maximum information gain[Srinivas et al., 2010], $\mathcal{X}$ is the input domain, $d > 0$ is the input dimension, and $r > 0$ is a constant. The BCR bounds of GP-UCB and IRGP-UCB are shown in Theorem B.1 and Theorems 4.2 and 4.3 in Takeno et al. [2023], respectively. Stars mean our results.

Diffferent acquistions Agree / Disagree ?



Knowledge Gradient (KG): Definition

Setup: Assume

$$f(x) \mid \mathcal{D}_n \sim \mathcal{N}\big(\mu_n(x),\, \sigma_n^2(x)\big),$$

where $\mathcal{D}_n = \{(x_1, y_1), \dots, (x_n, y_n)\}$ is the data so far. Define the incumbent solution as the point with the largest posterior mean:

$$\mu_n^* = \max_{x \in \mathcal{A}} \mu_n(x).$$

Improvement Function: If we were to take one more sample at x and update the posterior, the new maximum is

$$\mu_{n+1}^* = \max_{x' \in \mathcal{A}} \mu_{n+1}(x').$$

The improvement due to sampling at x is then

$$I(x) = \max\big(\mu_{n+1}^* - \mu_n^*,\, 0\big).$$

Knowledge Gradient: The KG acquisition function is defined as the expected improvement in the maximal posterior mean,

$$\mathrm{KG}_n(x) \;:=\; \mathbb{E}_n\Big[\, \mu_{n+1}^* - \mu_n^* \;\Big|\; x_{n+1} = x \Big].$$

Simulation-Based Estimation of KG

To estimate $\mathrm{KG}_n(x)$ via simulation, proceed as follows:

1. For a candidate x, repeat for $j = 1, \ldots, J$:
   1. Simulate an outcome $y_{n+1}^{(j)}$ from the predictive distribution at x:
   $$y_{n+1}^{(j)} \sim \mathcal{N}\left(\mu_n(x),\, \sigma_n^2(x)\right).$$
   2. Update the GP posterior by "hallucinating" the observation $(x, y_{n+1}^{(j)})$ to compute
   $$\mu_{n+1}^{(j)}(x') \quad \text{for all } x' \in \mathcal{A}.$$
   3. Let
   $$\mu_{n+1}^{*(j)} = \max_{x' \in \mathcal{A}} \mu_{n+1}^{(j)}(x').$$
   4. Compute the simulated improvement:
   $$\Delta^{(j)}(x) = \mu_{n+1}^{*(j)} - \mu_n^*.$$
2. Estimate the KG at x by averaging:
$$\mathrm{KG}_n(x) \approx \frac{1}{J} \sum_{j=1}^{J} \Delta^{(j)}(x).$$

Gradient !

$$\nabla \mathrm{KG}_n(x) = \mathbb{E}_n \left[\nabla \left(\mu^*_{n+1} - \mu^*_n \right) \right].$$

Algorithm 4 Simulation of unbiased stochastic gradients G with $E[G] = \nabla \mathrm{KG}_n(x)$. This stochastic gradient can then be used within stochastic gradient ascent to optimize the KG acquisition function.

for $j = 1$ to J **do**
  Generate $Z \sim \mathrm{Normal}(0, 1)$
  $y_{n+1} = \mu_n(x) + \sigma_n(x)Z$.
  Let $\mu_{n+1}(x'; x, y_{n+1}) = \mu_{n+1}(x'; x, \mu_n(x) + \sigma_n(x)Z)$ be the posterior mean at x' computed via (3) with (x, y_{n+1}) as the last observation.
  Solve $\max_{x'} \mu_{n+1}(x'; x, y_{n+1})$, e.g., using L-BFGS. Let $\widehat{x^*}$ be the maximizing x'.
  Let $G^{(j)}$ be the gradient of $\mu_{n+1}(\widehat{x^*}; x, \mu_n(x) + \sigma_n(x)Z)$ with respect to x, holding $\widehat{x^*}$ fixed.
end for
Estimate $\nabla \mathrm{KG}_n(x)$ by $G = \frac{1}{J} \sum_{j=1}^{J} G^{(j)}$.

Optimizing KG via Multistart Stochastic Gradient Ascent

Procedure:

1. Select R starting points $x_0^{(r)}$ uniformly from the feasible set $\mathcal{A}$.
2. For each starting point $r = 1, \dots, R$ and iterate $t = 0, 1, \dots, T-1$:

$$x_{t+1}^{(r)} = x_t^{(r)} + \alpha_t \, G\big(x_t^{(r)}\big),$$

   where:
   - $G(x_t^{(r)})$ is an unbiased stochastic gradient estimate of $\nabla \, \mathrm{KG}_n\big(x_t^{(r)}\big)$, obtained via infinitesimal perturbation analysis.
   - α_t is a stepsize (e.g., $\alpha_t = \frac{a}{a+t}$ for some parameter $a > 0$).
3. For each run r, estimate $\mathrm{KG}_n\big(x_T^{(r)}\big)$ using the simulation-based method above.
4. Return the best point among all runs:

$$x^* = \arg \max_{r=1,\dots,R} \mathrm{KG}_n\big(x_T^{(r)}\big).$$

Entropy Search (ES) and Predictive Entropy Search (PES)

Entropy Search (ES):

- ES quantifies uncertainty about the location of the global maximum x^* using differential entropy.
- It seeks the point x that produces the largest expected reduction in the entropy of the posterior over x^*.

$$\mathrm{ES}_n(x) = H\big(P_n(x^*)\big) - \mathbb{E}_{f(x)}\Big[H\big(P_n(x^* \mid f(x))\big)\Big].$$

Predictive Entropy Search (PES):

- PES reformulates the objective using mutual information:

$$\mathrm{PES}_n(x) = H\big(P_n(f(x))\big) - \mathbb{E}_{x^*}\Big[H\big(P_n(f(x) \mid x^*)\big)\Big].$$

- PES is generally more computationally tractable.

Takeaway: Both ES and PES aim to reduce uncertainty about x^* rather than simply improve the best expected value, and they can be particularly useful in *exotic* Bayesian optimization settings.

Expensive Constrained Optimization Problems (ECOPs)

- **ECOPs**: Optimization with computationally or financially expensive objectives and constraints.
- **Formulation**:

$$\begin{aligned} &\min_{\mathbf{x}} \; f(\mathbf{x}) = (f_1(\mathbf{x}), \ldots, f_m(\mathbf{x})) \\ &\text{s.t. } c_j(\mathbf{x}) \geq a_j, \; j = 1, \ldots, q, \\ &\mathbf{x} \in \mathcal{X}, \end{aligned}$$

  where $\mathbf{x} = (x_1, \ldots, x_d)$, $\mathcal{X}$ is the decision space, m objectives, q constraints.
- **Challenges**: Expensive evaluations, feasible solutions constrained.
- **Applications**: PID controller tuning, engineering design.

Constrained Bayesian Optimization (CBO)

- **Augmented Lagrangian (AL)**:

$$L_A(\mathbf{x}; \boldsymbol{\lambda}, \rho) = f(\mathbf{x}) + \boldsymbol{\lambda}^\top \mathbf{c}(\mathbf{x}) + \frac{1}{2\rho} \sum_{j=1}^{q} \max(0, c_j(\mathbf{x}))^2$$

  Converts constrained to unconstrained problems for Bayesian Optimization (BO).

- **CBO Approaches**:
  1. **Probability of Feasibility**: Constrained Expected Improvement (cEI):

$$\text{cEI}(\mathbf{x}) = \text{EI}(\mathbf{x}) \prod_{j=1}^{q} \Pr[c_j(\mathbf{x}) \leq a_j]$$

  2. **Expected Volume Reduction**: Uncertainty reduction via entropy or variance.
  3. **Multi-step Look-ahead**: Non-myopic, e.g., 2-OPT-C for long-term reward.

Surrogate-Assisted Methods and Advances

- **Surrogate-Assisted Constraint Handling**:
  - Combines BO with evolutionary algorithms.
  - Gaussian Processes (GPs) model objectives and constraints separately.
- **Recent Advances**:
  - AL with slack variables for mixed constraints.
  - ADMM-based BO for unknown constraints.
  - Predictive Entropy Search (PES) for decoupled constraints.
- **Challenges**:
  - Nonstationary modeling in AL.
  - Brittleness of cEI in highly constrained problems.
  - Computational burden in multi-step methods.

Multi-Fidelity Bayesian Optimization: Motivation

- **Engineering Design Challenge**: Optimize expensive high-fidelity (HF) functions $f_H(\mathbf{x})$, e.g., crash simulations (36-160h) or structural analysis (23 days) [1].
- **Limitations**: HF evaluations are costly, limiting optimization iterations under resource constraints.
- **Solution**: Multi-Fidelity Bayesian Optimization (MF BO) leverages cheap low-fidelity (LF) models to reduce HF evaluations while maintaining accuracy.
- **Advantages**:
  - Incorporates physical/mathematical insights.
  - Balances exploration-exploitation trade-off.
  - Handles uncertainty and supports parallel computing [1].
- **Applications**: Aerodynamic design, hyperparameter tuning, materials design.

Problem Formulation

- **Objective**: Solve

$$\min_{\mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^d} f_H(\mathbf{x}),$$

  where $f_H(\mathbf{x})$ is the HF objective, costly to evaluate.
- **Multi-Fidelity Setup**: Access to T models $f_1(\mathbf{x}), \ldots, f_T(\mathbf{x})$, with f_1 cheapest (LF) and $f_T = f_H$.
- **MF BO Approach**:
  - Use GP-based MF surrogates to model relationships between fidelities.
  - Guide optimization with acquisition functions to select next evaluation points and fidelities.
- **Goal**: Minimize HF evaluations by exploiting LF models' correlations [1].

Kennedy-O'Hagan (KOH) Auto-Regressive Model

- **Model**: For two fidelities, LF $f_1(\mathbf{x})$ and HF $f_2(\mathbf{x}) = f_H(\mathbf{x})$:

$$
\begin{aligned}
f_1(\mathbf{x}) &= \delta_1(\mathbf{x}), \\
f_2(\mathbf{x}) &= \rho_1 f_1(\mathbf{x}) + \delta_2(\mathbf{x}),
\end{aligned}
$$

  where $\delta_1, \delta_2 \sim \mathrm{GP}$, ρ_1 is a constant scaling factor [1].

- **General Form** (T fidelities):

$$
f_t(\mathbf{x}) = \rho_{t-1} f_{t-1}(\mathbf{x}) + \delta_t(\mathbf{x}), \quad t = 2, \ldots, T.
$$

- **Advantage**: Captures linear correlations between fidelities.
- **Limitation**: Assumes constant scaling, may not model complex relationships.

Hierarchical and Recursive Models

- **Hierarchical Kriging**:

$$f_1(\mathbf{x}) = a + z_1(\mathbf{x}),$$
$$f_t(\mathbf{x}) = \rho_{t-1}\mu_{f,t-1}(\mathbf{x}) + z_t(\mathbf{x}), \quad t = 2, \ldots, T,$$

  where $\mu_{f,t-1}$ is the Kriging predictor, $z_t \sim$ GP.

- **Recursive Model**:

$$f_t(\mathbf{x}) = \rho_{t-1}(\mathbf{x})\hat{f}_{t-1}(\mathbf{x}) + \delta_t(\mathbf{x}),$$

  with $\rho_{t-1}(\mathbf{x})$ a spatially varying adjustment, $\hat{f}_{t-1}$ the GP posterior [1].

- **Advantage**: Recursive model reduces training cost to $O(T \times \max\{N_t^3\})$.
- **Use Case**: Efficient for multiple fidelities with non-linear relationships.

Graphical Multi-Fidelity Gaussian Process (GMGP)

- **Model**: Represents fidelities as a directed acyclic graph (DAG):

$$f_t(\mathbf{x}) = \sum_{t' \in \mathrm{Pa}(t)} \rho_{t,t'} \hat{f}_{t'}(\mathbf{x}) + \delta_t(\mathbf{x}),$$

  where $\mathrm{Pa}(t)$ are parent nodes, $\hat{f}_{t'}$ is the GP posterior [1].
- **Covariance**: Structured via a lower triangular matrix $\mathbf{R}$.
- **Advantage**: Handles non-hierarchical fidelity relationships, e.g., multiple LF models informing HF.
- **Training Cost**: Recursive GMGP: $O(T \times \max\{N_t^3\})$.

- **Bayesian Hierarchical Model**:

$$f_2(\mathbf{x}) = \rho(\mathbf{x}) f_1(\mathbf{x}) + \delta_2(\mathbf{x}) + \varepsilon_2(\mathbf{x}),$$

  with $\rho(\mathbf{x}) \sim \mathrm{GP}$, $\varepsilon_2 \sim \mathcal{N}(0, \sigma^2_{\varepsilon,2})$.
- **Deep Gaussian Process (DGP)**:

$$f(\mathbf{x}) = f_{L-1}(\ldots f_1(f_0(\mathbf{x}))),$$

  where each $f_l \sim \mathrm{GP}$.
- **MF DGP**: Fidelities as layers, marginal likelihood computed via integration [1].
- **Challenge**: High computational cost for training and inference.

Input-Augmentation Multi-Fidelity GPs

- **Model**: Treat fidelity as an input variable in $g(\mathbf{t}, \mathbf{x})$, where $f_H(\mathbf{x}) = g(\mathbf{t}_T, \mathbf{x})$.
- **Continuous Fidelity**:

$$g(\cdot) \sim \mathrm{GP}(0, \kappa_g((\mathbf{t}, \mathbf{x}), (\mathbf{t}', \mathbf{x}') | \phi_g)),$$

  with $\kappa_g = \kappa_t(\mathbf{t}, \mathbf{t}')\kappa_x(\mathbf{x}, \mathbf{x}')$.
- **Categorical Fidelity**: Use non-continuous covariance functions, e.g., hypersphere decomposition [1].
- **Advantage**: Flexible for continuous or discrete fidelity levels, widely used in BO.

Acquisition Functions in MF BO

- **Types for BO** [1]:
  - *Improvement-based*: Expected Improvement (EI), balances exploration-exploitation.
  - *Optimistic*: Upper Confidence Bound (UCB), favors uncertainty.
  - *Information-based*: Entropy Search, maximizes information gain.
  - *Multi-step Look-ahead*: Considers future evaluations.
- **MF Considerations**:
  - *No-fidelity*: Treat all data as HF, inefficient.
  - *Heuristic*: Weight fidelities by cost-accuracy trade-off.
  - *Sequential Selection*: Choose fidelity and point iteratively.
- **Portfolio Approach**: Combine multiple acquisition functions for robustness.

Multi-step Look-ahead Acquisition Functions: Motivation

- **Problem**: Single-step acquisition functions (e.g., EI, UCB) are myopic, optimizing only for the immediate next evaluation [1].
- **Limitation**: May lead to suboptimal long-term decisions, especially in MF BO with varying fidelity costs and accuracies.
- **Solution**: Multi-step look-ahead acquisition functions consider future evaluations, planning a sequence of points to maximize cumulative improvement.
- **Advantages**:
  - Improves efficiency by anticipating future information gain.
  - Balances short-term gains with long-term optimization goals.
  - Critical for MF BO to optimize fidelity selection over multiple steps [1].
- **Applications**: Resource-constrained settings, e.g., aerodynamic optimization with limited HF budget.

Multi-step Look-ahead: Mathematical Formulation

- **Objective**: Maximize expected utility over a sequence of K future evaluations:

$$\alpha_{\text{MS}}(\mathbf{x}_1, \ldots, \mathbf{x}_K) = \mathbb{E}\left[U(f_H(\mathbf{x}^*)|\mathcal{D} \cup \{(\mathbf{x}_k, f_{t_k}(\mathbf{x}_k))\}_{k=1}^K)\right],$$

  where U is a utility function (e.g., improvement), $\mathcal{D}$ is current data, t_k is the fidelity at step k, and $\mathbf{x}^*$ is the optimal point [1].

- **Formulation**: For a two-step look-ahead:

$$\alpha_{\text{2-step}}(\mathbf{x}_1, t_1) = \mathbb{E}\left[\max_{\mathbf{x}_2, t_2} \mathbb{E}\left[U(f_H(\mathbf{x}^*)|\mathcal{D} \cup \{(\mathbf{x}_1, f_{t_1}(\mathbf{x}_1)), (\mathbf{x}_2, f_{t_2}(\mathbf{x}_2))\})\right]\right].$$

- **MF Extension**: Include fidelity selection t_k, weighting by cost c_{t_k}:

$$\alpha_{\text{MF-MS}}(\mathbf{x}_1, t_1) = \mathbb{E}\left[\max_{\mathbf{x}_2, t_2} \frac{\mathbb{E}[U|\mathcal{D} \cup \{(\mathbf{x}_1, f_{t_1}(\mathbf{x}_1)), (\mathbf{x}_2, f_{t_2}(\mathbf{x}_2))\}]}{c_{t_1} + c_{t_2}}\right].$$

- **Challenge**: High computational cost due to nested expectations.

Multi-step Look-ahead: Implementation and Techniques

- **Approximation Methods**:
  - *Monte Carlo Sampling*: Approximate expectations by sampling possible future outcomes [1].
  - *Dynamic Programming*: Use Bellman's principle to break down multi-step problem [1].
  - *One-shot Multi-step Trees*: Precompute decision trees for efficiency [2].
- **MF Considerations**:
  - Optimize both $\mathbf{x}_k$ and fidelity t_k at each step.
  - Incorporate cost-accuracy trade-offs in utility function.
- **Advantages**: Reduces HF evaluations by planning LF-heavy sequences early, reserving HF for final steps.
- **Limitations**: Computationally intensive; requires efficient sampling or approximation [1].

R. Bellman, "On the theory of dynamic programming," Proc. Natl. Acad. Sci., vol. 38, pp. 716–719, 1952.

S. Jiang et al., "Efficient nonmyopic Bayesian optimization via one-shot multi-step trees," Adv. Neural Inf. Process. Syst., vol. 33, pp. 18039–18049, 2020.

Applications and Challenges

- **Applications** [1]:
  - *Airfoil Design*: Optimize lift/drag using LF (XFOIL) and HF (CFD) models.
  - *Materials Design*: Ternary alloys via multi-fidelity simulations.
  - *Hyperparameter Tuning*: Use subset training as LF, full dataset as HF.
- **Future Research Topics**:
  - Constrained optimization: Handle complex constraints.
  - High-dimensional optimization: Subspace or additive structure approaches.
  - Optimization under uncertainty: Robust and reliability-based methods.
  - Multi-objective optimization: Pareto front exploration [1].

Key notes

- **Summary**: MF BO accelerates optimization of expensive HF functions by leveraging LF models, using GP-based surrogates and acquisition functions.
- **Key Models**: KOH, hierarchical/recursive, GMGP, Bayesian hierarchical, DGP, input-augmentation.
- **Impact**: Reduces computational cost, enables real-world applications in engineering and beyond.
- **Future**: Address high-dimensional, constrained, and multi-objective problems to broaden MF BO's applicability [1].

B. Do and R. Zhang, "Multi-Fidelity Bayesian Optimization: A Review," arXiv:2311.13050v2, 2023.

High-Dimensional BO: Challenges & Motivation

Challenges:

- **Exponential Sample Complexity:** Sample requirements grow exponentially with the dimension D.
- **Sparsity of Data:** Standard Gaussian Process surrogates lose accuracy when data is sparse.
- **Acquisition Function Landscape:** Often very flat with a few narrow peaks.

Motivation: Develop scalable surrogate models that exploit low-dimensional structure,

Random Embedding Methods (REMBO)

Approach: Assume that $f(x)$ varies mainly in a d-dimensional subspace.

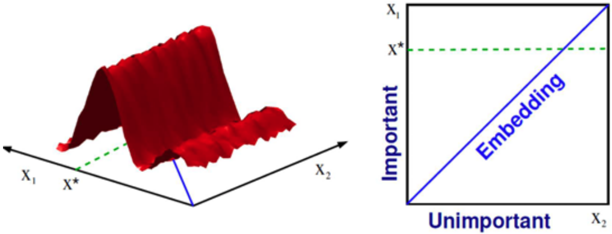
Method:

- Define a random projection $A : \mathbb{R}^d \to \mathbb{R}^D$, and represent x as

$$x = Az, \quad z \in \mathcal{Z} \subset \mathbb{R}^d.$$

- Optimize in the low-dimensional space:

$$z^* = \arg\max_{z \in \mathcal{Z}} f(Az).$$



Additive Gaussian Process Models

Approach: If f decomposes over variable groups,

$$f(x) = \sum_{i=1}^{M} f_i(x_{S_i}),$$

Surrogate Construction:

$$\mu(x) = \sum_{i=1}^{M} \mu_i(x_{S_i}), \quad \sigma^2(x) = \sum_{i=1}^{M} \sigma_i^2(x_{S_i}),$$

with μ_i, σ_i^2 defined via independent GP posteriors for f_i.

Trust Region Bayesian Optimization (TuRBO)

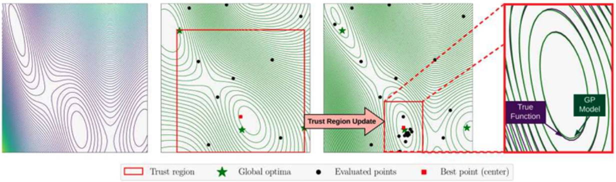
Approach: Restrict the search to a local region around the current best.

Method:

- Define the trust region at iteration t as

$$\mathcal{T}_t = \{x \in \mathcal{X} : \|x - x^+\| \leq \delta_t\},$$

  where x^+ is the current best and δ_t is the radius.
- Optimize the acquisition function (e.g., GP-UCB or EI) over $\mathcal{T}_t$.
- Adapt δ_t based on observed improvements.



Bayesian Neural Network (BNN) Surrogates

Motivation: GP surrogates may become computationally expensive in high dimensions. BNNs scale better and capture complex structure.

BNN Model: For a deep neural network with weights w and input x:

$$f(x; w).$$

Prior and Posterior:

- Place a prior $p(w)$ on the weights.
- Given data $\mathcal{D}$, the posterior is

$$p(w \mid \mathcal{D}) \propto p(\mathcal{D} \mid w)\, p(w).$$

- Approximate via variational inference with $q(w; \lambda) \approx p(w \mid \mathcal{D})$.

Predictive Distribution:

$$p(y \mid x, \mathcal{D}) \approx \int p(y \mid x, w)\, q(w; \lambda)\, dw,$$

often approximated with Monte Carlo sampling.

Advantage: Scales to high dimensions and integrates modern deep learning methods.

Other Approaches for High-Dimensional BO

Additional Strategies:

- **Dropout Methods:** Apply dropout at test time to create an implicit ensemble, reducing effective dimensions.
- **Deep Ensembles:** Train several independent models and aggregate their predictions:

$$\mu_{\text{ens}}(x) = \frac{1}{M} \sum_{m=1}^{M} f(x; w_m), \quad \sigma^2_{\text{ens}}(x) = \frac{1}{M} \sum_{m=1}^{M} \left(f(x; w_m) - \mu_{\text{ens}}(x)\right)^2 .$$

- **Random Embedding with Local Search:** Combine REMBO with local optimization to refine the search in the embedded space.

Take-Away: The aim is to reduce the effective dimensionality (or exploit low-dimensional structure) while retaining accurate uncertainty estimates.

References

- Srinivas, N., Krause, A., Kakade, S., & Seeger, M. (2010). Gaussian Process Optimization in the Bandit Setting: No Regret and Experimental Design. In ICML.
- Abbasi-Yadkori, Y., Pál, D., & Szepesvári, C. (2011). Improved Algorithms for Linear Stochastic Bandits.
- Rasmussen, C. E. (2006). Gaussian Processes for Machine Learning.
- Hernandez-Lobato, J. M., et al. (2014). Predictive Entropy Search for Efficient Global Optimization of Black-Box Functions. NeurIPS.
- Snoek, J., Larochelle, H., & Adams, R. P. (2012). Practical Bayesian Optimization of Machine Learning Algorithms. NeurIPS.
- Recent advances: Posterior Sampling-Based Bayesian Optimization with Tighter Bayesian Regret Bounds; Batch Bayesian Optimization methods (e.g., GP-BUCB, Local Penalization); High-dimensional BO via Random Embedding and Additive GPs.
- Additional references from Vu Nguyen's BO tutorials.

Thank You

Thank you for your attention!

Questions?