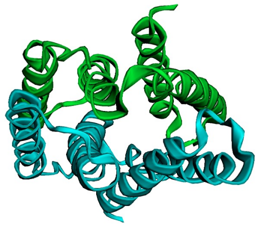
Unifying Likelihood-free Inference with Black-box Optimization and Beyond

ICLR 2022 spotlight

Dinghuai Zhang, Jie Fu, Yoshua Bengio, Aaron Courville

Black-box Optimization (BB-Opt)

- Many real world problems, such as biological drug discovery, are examples of black-box optimization problems.
- Consider $f(\cdot)$ is a (black-box) oracle score function of entity m, e.g. a particular chemical property
- And we want to find a drug that has the optimal property



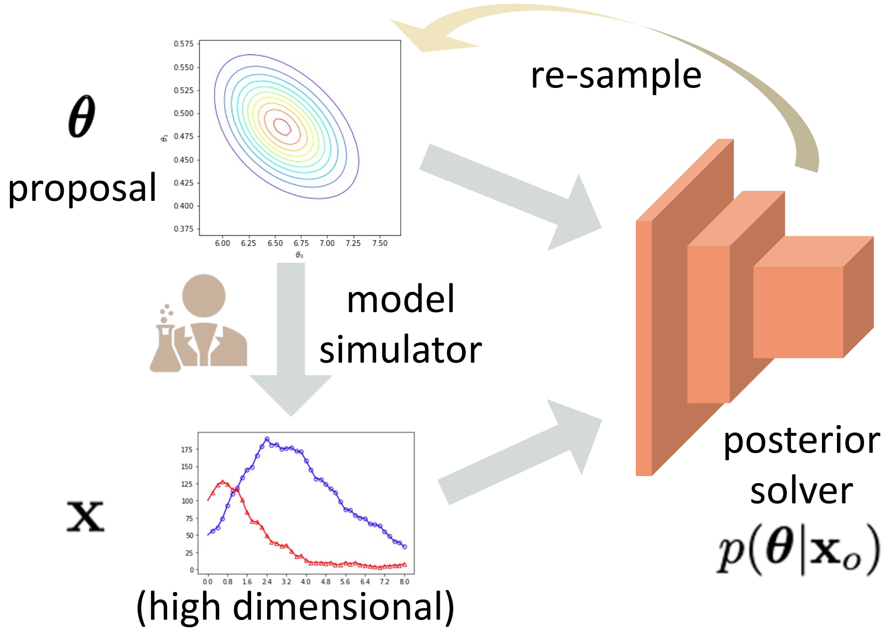
$$\mathbf{m}^* = \arg\max_{\mathbf{m}\in\mathcal{M}} f(\mathbf{m})$$

Likelihood-free Inference (LFI)

- LFI is a special kind of Bayesian inference where the likelihood function is intractable
- Instead, we are allowed to sample from it: $\mathbf{x} \sim p(\mathbf{x}|\boldsymbol{\theta})$
- Want to infer the posterior from prior and likelihood:

$$p(\boldsymbol{\theta}|\mathbf{x}_o) \propto p(\boldsymbol{\theta}) \underbrace{p(\mathbf{x}_o|\boldsymbol{\theta})}_{?}$$

("o" means observation)



Unifying LFI and BB-Opt

- Assume $\mathcal{E}$ denotes a Boolean event:
  - "generated drug $\mathbf{m}$ has good property"
- Then we have a intriguing connection between the two fields!

	Likelihood-free inference	Black-box optimization
Element	$(\boldsymbol{\theta}, \mathbf{x})$	$(\mathbf{m}, s)$
Target	$p(\boldsymbol{\theta}\|\mathbf{x}_o)$	$p(\mathbf{m}\|\mathcal{E})$
Constraint	limited simulation: $\mathbf{x} \sim p(\mathbf{x}\|\boldsymbol{\theta})$ intractable likelihood: $p(\mathbf{x}\|\boldsymbol{\theta})$	limited query: $s \sim f(\mathbf{m})$ black-box oracle: $f(\mathbf{m})$

Table 1: Correspondence between likelihood-free inference and black-box optimization.

This insight can help us understand and design many black-box drug discovery algorithms...

Linking existing methods...

- Some existing sequence-design algorithms, i.e. "*FB-VAE*", already corresponds to classical LFI algorithms "SMC-ABC"

Existing LFI algorithm

Algorithm 1 SMC-ABC

$p_1(\boldsymbol{\theta}) \leftarrow p(\boldsymbol{\theta})$;
for r in 1 to R **do**
  repeat
    sample $\boldsymbol{\theta}_i \sim p_r(\boldsymbol{\theta})$;
    simulate $\mathbf{x}_i \sim p(\mathbf{x}|\boldsymbol{\theta}_i)$;
  until n samples are obtained
  $\mathcal{D} \leftarrow \mathcal{D} \cup \{(\boldsymbol{\theta}_i, \mathbf{x}_i)\}_{i=1}^n$
  sort $\mathcal{D}$ according to $-\|\mathbf{x}_i - \mathbf{x}_o\|$;
  fit $q_\phi(\boldsymbol{\theta})$ with top $\{\boldsymbol{\theta}_i\}_i$ in $\mathcal{D}$;
  $p_{r+1}(\boldsymbol{\theta}) \leftarrow q_\phi(\boldsymbol{\theta})$;
end for
return $\hat{p}(\boldsymbol{\theta}|\mathbf{x}_o) = p_{R+1}(\boldsymbol{\theta})$

Existing BB-Opt algorithm

Algorithm 2 FB-VAE

$p_1(\mathbf{m}) \leftarrow p(\mathbf{m})$;
for r in 1 to R **do**
  repeat
    sample $\mathbf{m}_i \sim p_r(\mathbf{m})$;
    query the oracle: $s_i \leftarrow f(\mathbf{m}_i)$;
  until n samples are obtained
  $\mathcal{D} \leftarrow \mathcal{D} \cup \{(\mathbf{m}_i, s_i)\}_{i=1}^n$;
  sort $\mathcal{D}$ according to s_i
  fit $q_\phi(\mathbf{m})$ with top $\{\mathbf{m}_i\}_i$ in $\mathcal{D}$;
  $p_{r+1}(\mathbf{m}) \leftarrow q_\phi(\mathbf{m})$;
end for
return $\{\mathbf{m} : (\mathbf{m}, s) \in \mathcal{D}\}$

Linking existing methods...

- "Design by adaptive sampling" corresponds to classical LFI algorithms "Sequential Neural Posterior"

Existing LFI algorithm

Algorithm 3 Sequential Neural Posterior

$p_1(\boldsymbol{\theta}) \leftarrow p(\boldsymbol{\theta})$;
for r in 1 to R **do**
  repeat
    sample $\boldsymbol{\theta}_i \sim p_r(\boldsymbol{\theta})$;
    simulate $\mathbf{x}_i \sim p(\mathbf{x}|\boldsymbol{\theta}_i)$;
  until n samples are obtained
  $\mathcal{D} \leftarrow \mathcal{D} \cup \{(\boldsymbol{\theta}_i, \mathbf{x}_i)\}_{i=1}^{n}$;
  $q_\phi \leftarrow \arg\min_q \mathbb{E}_{\mathbf{x}} \left[D_{\mathrm{KL}}(p(\boldsymbol{\theta}|\mathbf{x}) \| q)\right]$;
  $p_{r+1}(\boldsymbol{\theta}) \leftarrow q_\phi(\boldsymbol{\theta}|\mathbf{x}_o)$;
end for
return $\hat{p}(\boldsymbol{\theta}|\mathbf{x}_o) = p_{R+1}(\boldsymbol{\theta})$

Existing BB-Opt algorithm

Algorithm 4 Design by Adaptive Sampling

$p_1(\mathbf{m}) \leftarrow p(\mathbf{m})$;
for r in 1 to R **do**
  repeat
    sample $\mathbf{m}_i \sim p_r(\mathbf{m})$;
    query the oracle: $s_i \leftarrow f(\mathbf{m}_i)$;
  until n samples are obtained
  $\mathcal{D} \leftarrow \mathcal{D} \cup \{(\mathbf{m}_i, s_i)\}_{i=1}^{n}$;
  $q_\phi \leftarrow \arg\min_q D_{\mathrm{KL}}(p(\mathbf{m}|\mathcal{E}) \| q)$;
  $p_{r+1}(\mathbf{m}) \leftarrow q_\phi(\mathbf{m})$;
end for
return $\{\mathbf{m} : (\mathbf{m}, s) \in \mathcal{D}\}$

... and proposing many new ones!

Existing LFI algorithm

Algorithm 5 Sequential Neural Likelihood

$p_1(\boldsymbol{\theta}) \leftarrow p(\boldsymbol{\theta})$;
for r in 1 to R **do**
  repeat
    sample $\boldsymbol{\theta}_i \sim p_r(\boldsymbol{\theta})$;
    simulate $\mathbf{x}_i \sim p(\mathbf{x}|\boldsymbol{\theta}_i)$;
  until n samples are obtained
  $\mathcal{D} \leftarrow \mathcal{D} \cup \{(\boldsymbol{\theta}_i, \mathbf{x}_i)\}_{i=1}^n$
  fit $q_\phi(\mathbf{x}|\boldsymbol{\theta})$ with $\mathcal{D}$;
  $p_{r+1}(\boldsymbol{\theta}) \propto p(\boldsymbol{\theta}) \cdot q_\phi(\mathbf{x}_o|\boldsymbol{\theta})$;
end for
return $\hat{p}(\boldsymbol{\theta}|\mathbf{x}_o) = p_{R+1}(\boldsymbol{\theta})$

Novel BB-Opt algorithm

Algorithm 6 Iterative Scoring

$p_1(\mathbf{m}) \leftarrow p(\mathbf{m})$;
for r in 1 to R **do**
  repeat
    sample $\mathbf{m}_i \sim p_r(\mathbf{m})$;
    query the oracle: $s_i \leftarrow f(\mathbf{m}_i)$;
  until n samples are obtained
  $\mathcal{D} \leftarrow \mathcal{D} \cup \{(\mathbf{m}_i, s_i)\}_{i=1}^n$;
  fit $\hat{f}_\phi(\mathbf{m})$ with $\mathcal{D}$;
  construct $\tilde{q}(\mathbf{m})$ with $\hat{f}_\phi(\cdot)$ and $p(\mathbf{m})$;
  $p_{r+1}(\mathbf{m}) \leftarrow \tilde{q}(\mathbf{m})$;
end for
return $\{\mathbf{m} : (\mathbf{m}, s) \in \mathcal{D}\}$

... and proposing many new ones!

Existing LFI algorithm

Algorithm 7 Sequential Neural Ratio

$p_1(\boldsymbol{\theta}) \leftarrow p(\boldsymbol{\theta})$;
for r in 1 to R **do**
  repeat
    sample $\boldsymbol{\theta}_i, \boldsymbol{\theta}'_i \sim p_r(\boldsymbol{\theta})$;
    simulate $\mathbf{x}_i \sim p(\mathbf{x}|\boldsymbol{\theta}_i)$;
  until n samples are obtained
  $\mathcal{D} \leftarrow \mathcal{D} \cup \{(\boldsymbol{\theta}_i, \mathbf{x}_i)\}_{i=1}^{n}$
  $\mathcal{D}' \leftarrow \mathcal{D}' \cup \{(\boldsymbol{\theta}'_i, \mathbf{x}_i)\}_{i=1}^{n}$
  train $d_\phi(\boldsymbol{\theta}, \mathbf{x})$ classifying between $\mathcal{D}$ and $\mathcal{D}'$ with the loss in Eq. 4;
  $r_\phi(\boldsymbol{\theta}, \mathbf{x}) \leftarrow \frac{d_\phi(\boldsymbol{\theta}, \mathbf{x})}{1 - d_\phi(\boldsymbol{\theta}, \mathbf{x})}$;
  $p_{r+1}(\boldsymbol{\theta}) \propto r_\phi(\boldsymbol{\theta}, \mathbf{x}) \cdot p(\boldsymbol{\theta})$;
end for
return $\hat{p}(\boldsymbol{\theta}|\mathbf{x}_o) = p_{R+1}(\boldsymbol{\theta})$

Novel BB-Opt algorithm

Algorithm 8 Iterative Ratio

$p_1(\mathbf{m}) \leftarrow p(\mathbf{m})$;
for r in 1 to R **do**
  repeat
    sample $\mathbf{m}_i \sim p_r(\mathbf{m})$;
    query the oracle: $s_i \leftarrow f(\mathbf{m}_i)$;
  until n samples are obtained
  $\mathcal{D} \leftarrow \mathcal{D} \cup \{(\mathbf{m}_i, s_i)\}_{i=1}^{n}$;
  construct $\tilde{\mathcal{D}}$ with $\mathbf{m}$ in $\mathcal{D}$ satisfying $\mathcal{E}$;
  construct $\tilde{\mathcal{D}}'$ from $p(\mathbf{m})$;
  train $d_\phi(\mathbf{m})$ classifying between $\tilde{\mathcal{D}}$ and $\tilde{\mathcal{D}}'$;
  $r_\phi(\mathbf{m}) \leftarrow \frac{d_\phi(\mathbf{m})}{1 - d_\phi(\mathbf{m})}$;
  $p_{r+1}(\boldsymbol{\theta}) \propto r_\phi(\mathbf{m}) \cdot p(\mathbf{m})$;
end for
return $\{\mathbf{m} : (\mathbf{m}, s) \in \mathcal{D}\}$

... and proposing many new ones!

We also combine existing methods to propose novel composite black-box opitimization algorithms (see details in paper)

Novel composite BB-Opt algorithms

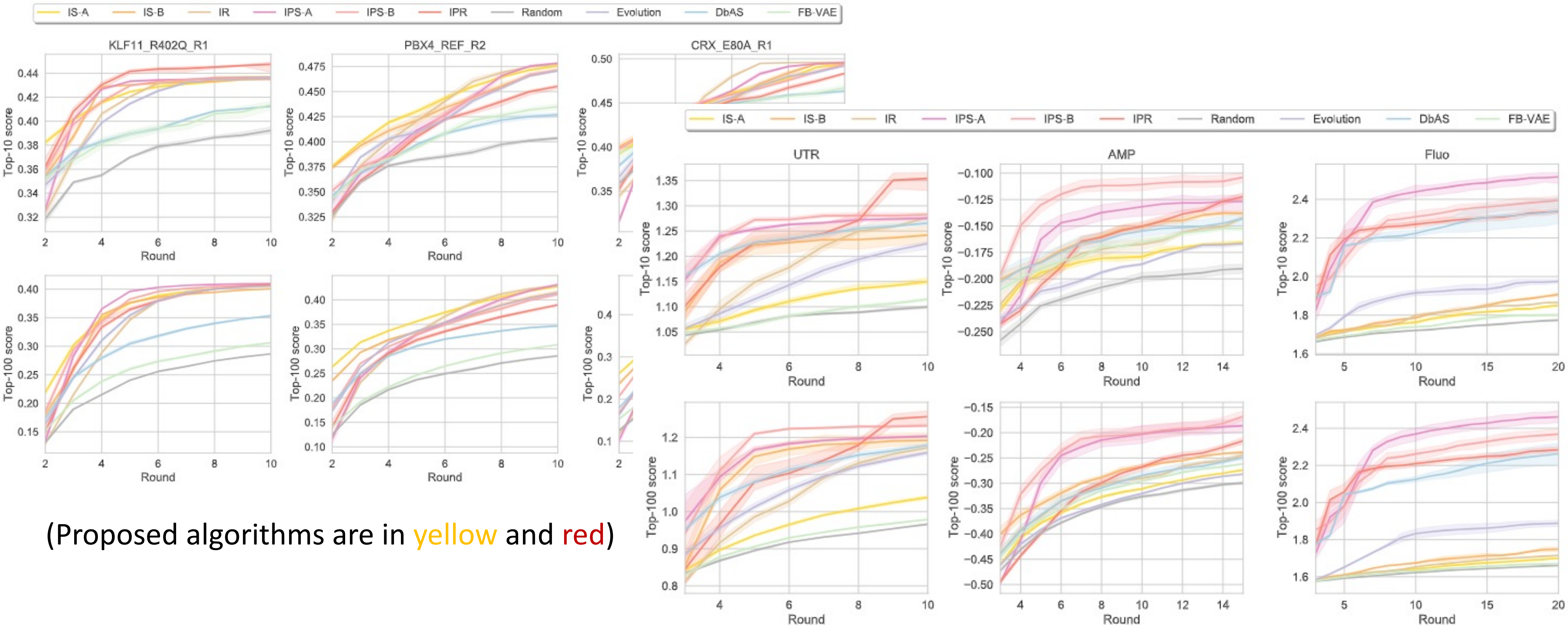
Algorithm 9 Iterative Posterior Scoring

$p_1(\mathbf{m}) \leftarrow p(\mathbf{m})$;
for r in 1 to R **do**
  repeat
    sample $\mathbf{m}_i \sim p_r(\mathbf{m})$;
    query the oracle: $s_i \leftarrow f(\mathbf{m}_i)$;
  until n samples are obtained
  $\mathcal{D} \leftarrow \mathcal{D} \cup \{(\mathbf{m}_i, s_i)\}_{i=1}^{n}$;
  fit $\hat{f}_\phi(\mathbf{m})$ with $\mathcal{D}$;
  construct $\tilde{q}(\mathbf{m})$ with $\hat{f}_\phi(\cdot)$ and $p(\mathbf{m})$;
  $q_\psi \leftarrow \arg\min_q D_{\mathrm{KL}}(\tilde{q}(\mathbf{m}) \| q)$;
  $p_{r+1}(\mathbf{m}) \leftarrow q_\psi(\mathbf{m})$;
end for
return $\{\mathbf{m} : (\mathbf{m}, s) \in \mathcal{D}\}$

Algorithm 10 Iterative Posterior Ratio

$p_1(\mathbf{m}) \leftarrow p(\mathbf{m})$;
for r in 1 to R **do**
  repeat
    sample $\mathbf{m}_i \sim p_r(\mathbf{m})$;
    query the oracle: $s_i \leftarrow f(\mathbf{m}_i)$;
  until n samples are obtained
  $\mathcal{D} \leftarrow \mathcal{D} \cup \{(\mathbf{m}_i, s_i)\}_{i=1}^{n}$;
  construct $\tilde{\mathcal{D}}$ with $\mathbf{m}$ in $\mathcal{D}$ satisfying $\mathcal{E}$;
  construct $\tilde{\mathcal{D}}'$ from $p(\mathbf{m})$;
  train $d_\phi(\mathbf{m})$ classifying between $\tilde{\mathcal{D}}$ and $\tilde{\mathcal{D}}'$;
  $r_\phi(\mathbf{m}) \leftarrow \frac{d_\phi(\mathbf{m})}{1-d_\phi(\mathbf{m})}$;
  construct $\tilde{q}(\mathbf{m})$ with $r_\phi(\mathbf{m})$ and $p(\mathbf{m})$;
  $q_\psi \leftarrow \arg\min_q D_{\mathrm{KL}}(\tilde{q}(\mathbf{m}) \| q)$;
  $p_{r+1}(\mathbf{m}) \leftarrow q_\psi(\mathbf{m})$;
end for
return $\{\mathbf{m} : (\mathbf{m}, s) \in \mathcal{D}\}$

Achieve comparable / better performance...



(Proposed algorithms are in yellow and red)

Thank you for listening!