Deep Adaptive Design: Amortizing Sequential Bayesian Experimental Design Discussion

Presented by

James Johnson

Authors

- Adam Foster
  - Senior Researcher at Microsoft Research AI4Science
- Desi R. Ivanova
  - StatML CDT programme at the University of Oxford
- Ilyas Malik
  - AI Research Engineer|ex Amazon, IBM Research, Oxford
- Tom Rainforth
  - Senior Researcher in Machine learning and leader of the RainML Research Lab at the Dept of Statistics in the University of Oxford.

Contents

Introduction

- We want to design experiments so that outcomes are very informative.
- Experiments may have multiple iterations and decisions so an update to the experimental design may be necessary.
- Example Marco Polo Game where player 2 doesn’t move.
  - Player 1 closes there eyes and yells Marco.
  - Player 2 responds Polo.
  - Player 1 moves in direction of Player 2.
  - Player calls out Marco again from the new location and continues until player 2 is found.

Bayesian Optimal Experimental Design (BOED)

- Goal of the BOED framework is to aid in the design of experiments so that the outcomes will be as informative as possible.
- BOED framework modeled in a Bayesian manner
  - y is the experiment outcome.
  - ξ is our controllable design
  - θ is the set of parameters we wish to learn about.
  - $p(\theta)$ is the prior. What we know about the parameters before data.
  - $p(y|\theta, \xi)$ is the likelihood.
- Key idea is to optimize ξ to maximize the expected amount of information that will be gained about variables of interest θ upon observing outcome y.

Entropy and the conventional iterative approach.

- Entropy of a discrete random variable X with dist p over K states.

$$\mathbb{H}(X) \triangleq -\sum_{k=1}^{K} p(X=k) \log \; p(X=k) = -\mathbb{E}_X [\log p(X)]$$

- Optimize ξ to maximize the expected information gained about θ

$$\mathrm{IG}(\xi, y) = \mathrm{H}[p(\theta)] - \mathrm{H}[p(\theta|\xi, y)]$$

- BOED is truly realized when it's used to design a sequence of experiments $\xi_1, \ldots, \xi_T$
  - To construct *adapative* strategies which utilize information from past data to tailor each successive design ξ_t during the progress of the experiment.
- The conventional iterative approach for selecting each ξ_t is to fit the posterior .

$$p(\theta|\xi_{1:t-1}, y_{1:t-1})$$

BOED Framework

- After running a hypothetical experiment with design ξ and observing y, our updated beliefs are the posterior $p(\theta|\xi, y)$.

$$\mathrm{IG}(\xi, y) = \mathrm{H}\left[p(\theta)\right] - \mathrm{H}\left[p(\theta|\xi, y)\right]$$

- Expected Information Gain
  - Formed by taking the expectation over possible outcomes y, using the model itself to simulate these.

BOED Framework Cont.

- Marginal Likelihood

$$y \sim p(y|\xi) = \mathbb{E}_{p(\theta)}[p(y|\theta, \xi)]$$

- Expected Information Gain (EIG)

$$\begin{aligned} I(\xi) &:= \mathbb{E}_{p(y|\xi)}\left[\mathrm{IG}(\xi, y)\right] \\ &= \mathbb{E}_{p(\theta)p(y|\theta,\xi)}\left[\log p(\theta|\xi, y) - \log p(\theta)\right] \\ &= \mathbb{E}_{p(\theta)p(y|\theta,\xi)}\left[\log p(y|\theta, \xi) - \log p(y|\xi)\right] \end{aligned}$$

BOED Framework Cont.

- The optimal design is defined as $\xi^* = \text{argmax}_{\xi \Xi} I(\xi)$ the space of feasible designs.
- Ξ: The space of feasible designs.
- The power of the BOED framework can thus be significantly increased by using an adaptive design strategy that chooses each ξ_t dependent upon $\xi_{1:t-1}, y_{1:t-1}$.
  - Enables us to use what we learned previous experiments to design the next one optimally.

Iterative Approach is computational expensive.

- Significant Computational time between each step of the experiment in order to update the posterior and compute the next optimal design.
- $I(\xi)$ is doubly intractable and its optimization constitutes a significant computational bottleneck.

Conventional Approach

- The conventional approach is to fit the posterior distribution at each step and then optimize the EIG objective that uses this posterior in place of the prior.

$$I(\xi_t) = \mathbb{E}_{p(\theta|\xi_{1:t-1}, y_{1:t-1})p(y_t|\theta,\xi_t)} \left[\log \frac{p(y_t|\theta, \xi_t)}{p(y_t|\xi_t)} \right] \quad (3)$$

where $p(y_t|\xi_t) = \mathbb{E}_{p(\theta|\xi_{1:t-1}, y_{1:t-1})}[p(y_t|\theta, \xi_t)]$.

Theorem 1

Theorem 1. *The total expected information gain for policy π over a sequence of T experiments is*

$$\mathcal{I}_T(\pi) := \mathbb{E}_{p(\theta)p(h_T|\theta,\pi)}\left[\sum_{t=1}^{T} I_{h_{t-1}}(\xi_t)\right] \tag{7}$$

$$= \mathbb{E}_{p(\theta)p(h_T|\theta,\pi)}\left[\log p(h_T|\theta,\pi) - \log p(h_T|\pi)\right] \tag{8}$$

where $p(h_T|\pi) = \mathbb{E}_{p(\theta)}[p(h_T|\theta,\pi)]$.

Theorem 1

- $I_T(\pi)$ is the expected reduction in entropy from the prior $p(\theta)$ to the final posterior $p(\theta|h_T)$ **without** considering the intermediate posteriors at all.
- Paper generalizes conventional BOED frameworks to having a fixed T.

Theorem 1. *The total expected information gain for policy π over a sequence of T experiments is*

$$\mathcal{I}_T(\pi) := \mathbb{E}_{p(\theta)p(h_T|\theta,\pi)}\left[\sum_{t=1}^{T} I_{h_{t-1}}(\xi_t)\right] \tag{7}$$

$$= \mathbb{E}_{p(\theta)p(h_T|\theta,\pi)}\left[\log p(h_T|\theta,\pi) - \log p(h_T|\pi)\right] \tag{8}$$

where $p(h_T|\pi) = \mathbb{E}_{p(\theta)}[p(h_T|\theta,\pi)]$.

Thinking in terms of Policy

- $I_T(\pi)$ is a function of the policy, not the designs themselves.
- Conventional adaptive BOED approximates π_s.
- Once π is learned, it can just be directly evalutated during the experiment itself.

Theorem 1. *The total expected information gain for policy π over a sequence of T experiments is*

$$\mathcal{I}_T(\pi) := \mathbb{E}_{p(\theta)p(h_T|\theta,\pi)}\left[\sum_{t=1}^{T} I_{h_{t-1}}(\xi_t)\right] \quad (7)$$

$$= \mathbb{E}_{p(\theta)p(h_T|\theta,\pi)}\left[\log p(h_T|\theta,\pi) - \log p(h_T|\pi)\right] \quad (8)$$

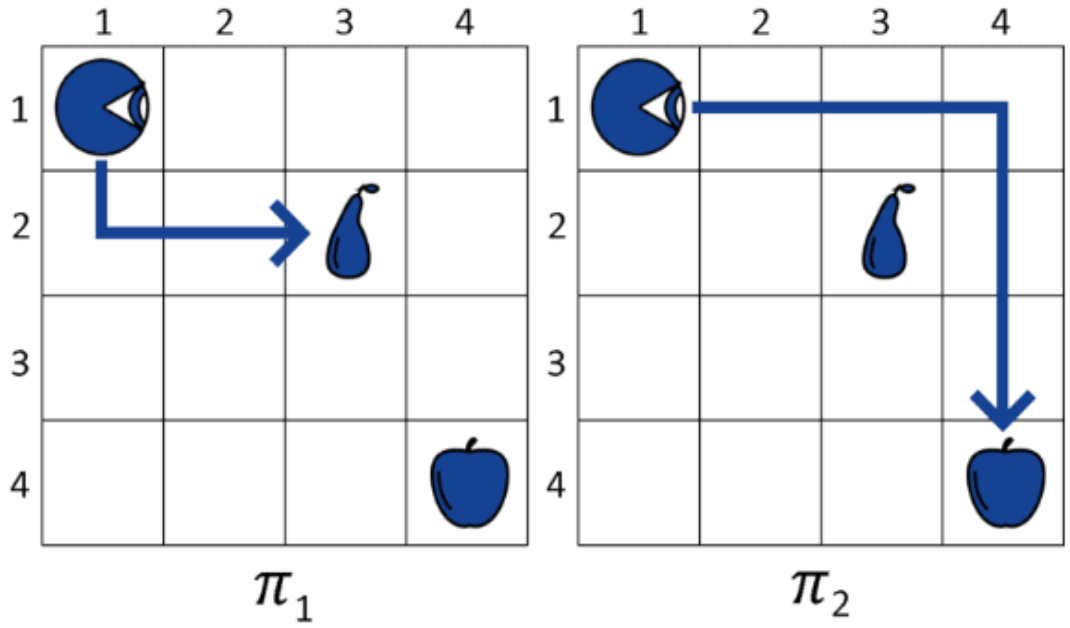
where $p(h_T|\pi) = \mathbb{E}_{p(\theta)}[p(h_T|\theta,\pi)]$.

Optimal Design in terms of ξ: $\xi^* = \text{argmax}_{\xi\Xi} I(\xi)$

Optimal Design in terms of π: $\pi^* = \arg\max_{\pi} I_T(\pi)$

Policy

- List of actions a system or agent learns.
- $a = \pi(x)$
- Specifies which action to take in response to each possible input.
- Ex. of 2 policies
  - $\pi_1 = down, right, right \rightarrow Pear$
  - $\pi_2 = right\ right\ right\ down\ down\ down \rightarrow Apple$



Deep Adaptive Design (DAD)

- A method for amortizing the cost of adaptive Bayesian experimental design
- DAD looks to approximate π^* in the policy optimal design eq.
- Constructs a single design network π_ϕ learned by simulating hypothetical experimental trajectories with trainable params ϕ.
  - Input:
    - Designs and Observations from previous stages
  - Output:
    - The design to use for the next experiment.

Deep Adaptive Design (DAD)

- Learns a design policy which makes decisions as a function of the past data, and we optimize the parameters of this policy rather than an individual design.
- **Can be learned without any direct posterior or marginal likelihood estimation.**

Deep Adaptive Design Algorithm

Algorithm 1 Deep Adaptive Design (DAD)

Input: Prior $p(\theta)$, likelihood $p(y|\theta, \xi)$, number of steps T
Output: Design network π_ϕ
while training compute budget not exceeded **do**
  Sample $\theta_0 \sim p(\theta)$ and set $h_0 = \varnothing$
  for $t = 1, ..., T$ **do**
    Compute $\xi_t = \pi_\phi(h_{t-1})$
    Sample $y_t \sim p(y|\theta_0, \xi_t)$
    Set $h_t = \{(\xi_1, y_1), ..., (\xi_t, y_t)\}$
  end
  Compute estimate for $d\mathcal{L}_T/d\phi$ as per § 4.2
  Update ϕ using stochastic gradient ascent scheme
end
At deployment, π_ϕ is fixed, we take $\xi_t = \pi_\phi(h_{t-1})$, and each y_t is obtained by running an experiment with ξ_t.

Contrastive bounds for sequential experiments

- To train π_ϕ with stochastic gradient methods
  - Optimize the lower bounds on $I_T(\pi_\phi)$
  - Equations we looked at before:

$$= \mathbb{E}_{p(\theta)p(h_T|\theta,\pi)} \left[\log p(h_T|\theta,\pi) - \log p(h_T|\pi)\right] \quad (8)$$

where $p(h_T|\pi) = \mathbb{E}_{p(\theta)}[p(h_T|\theta,\pi)]$.

$$p(\theta)p(h_T|\theta,\pi) = p(\theta)\prod_{t=1}^{T} p(y_t|\theta,\xi_t). \quad (5)$$

Contrastive bounds for sequential experiments

- Given a sample $\theta_0, h_T \sim p(\theta, h_T|\pi)$
  - Estimation can be made by introducing L independent contrastive samples $\theta_{1:L} \sim p(\theta)$.
- Equation 9 and 10 computes the log ratio in eq 8 in 2 ways.
  - g_L cannot exceed log(L+1)
  - f_L is potentially unbounded.

$$= \mathbb{E}_{p(\theta)p(h_T|\theta,\pi)} \left[\log p(h_T|\theta,\pi) - \log p(h_T|\pi)\right] \quad (8)$$

where $p(h_T|\pi) = \mathbb{E}_{p(\theta)}[p(h_T|\theta,\pi)]$.

$$p(\theta)p(h_T|\theta,\pi) = p(\theta)\prod_{t=1}^{T} p(y_t|\theta,\xi_t). \quad (5)$$

$$g_L(\theta_{0:L}, h_T) = \log \frac{p(h_T|\theta_0,\pi)}{\frac{1}{L+1}\sum_{\ell=0}^{L} p(h_T|\theta_\ell,\pi)} \quad (9)$$

$$f_L(\theta_{0:L}, h_T) = \log \frac{p(h_T|\theta_0,\pi)}{\frac{1}{L}\sum_{\ell=1}^{L} p(h_T|\theta_\ell,\pi)}. \quad (10)$$

Theorem 2

Theorem 2 (Sequential PCE). *For a design function π and a number of contrastive samples $L \geq 0$, let*

$$\mathcal{L}_T(\pi, L) = \mathbb{E}_{p(\theta_0, h_T|\pi)p(\theta_{1:L})}\left[g_L(\theta_{0:L}, h_T)\right] \tag{11}$$

where $g_L(\theta_{0:L}, h_T)$ is as per (9), and $\theta_0, h_T \sim p(\theta, h_T|\pi)$, and $\theta_{1:L} \sim p(\theta)$ independently. Given minor technical assumptions discussed in the proof, we have[2]

$$\mathcal{L}_T(\pi, L) \uparrow \mathcal{I}_T(\pi) \text{ as } L \to \infty \tag{12}$$

at a rate $\mathcal{O}\left(L^{-1}\right)$.

Contrastive Bounds

- Upper bound: Sequential Nested Monte Carlo (sNMC)

$$\mathcal{U}_T(\pi, L) = \mathbb{E}_{p(\theta_0, h_T|\pi)p(\theta_{1:L})}\left[f_L(\theta_{0:L}, h_T)\right]. \qquad (13)$$

- Theorem 4 in Appendix A shows that U_T satisfies complementary properties of L_T
- The bounds are:

$$\mathcal{L}_T(\pi, L) \leq \mathcal{I}_T(\pi) \leq \mathcal{U}_T(\pi, L)$$

Gradient Estimation

- Design network parameters ϕ can be optimized using a stochastic optimization scheme
- Using the reparameterization trick and the law of unconscious statician we can write:

$$\frac{d\mathcal{L}_T}{d\phi} = \mathbb{E}_{p(\theta_{0:L})p(\epsilon_{1:T})}\left[\frac{d}{d\phi} g_L(\theta_{0:L}, h_T)\right]. \qquad (14)$$

-

Gradient Estimation

- One approach to computing the gradient is to sum over all possible histories h_T integrating out the variables $y_{1:T}$ and take gradients with respect to ϕ:

$$\frac{d\mathcal{L}_T}{d\phi} = \mathbb{E}\left[\sum_{h_T} \frac{d}{d\phi}\Big(p(h_T|\theta_0)g_L(\theta_{0:L}, h_T)\Big)\right], \qquad (15)$$

Unbiased gradient estimates can be computed using samples from the prior.

Gradient Estimation

- This gradient estimator has a computational cost $O(|Y|^T)$.
- Number of experiments T and # of outcomes |Y| need to be relatively small.

Score Function Gradient Estimator

$$\frac{d\mathcal{L}_T}{d\phi} = \mathbb{E}\left[\left(\log \frac{p(h_T|\theta_0,\pi_\phi)}{\sum_{\ell=0}^{L} p(h_T|\theta_\ell,\pi_\phi)}\right)\frac{d}{d\phi}\log p(h_T|\theta_0,\pi_\phi) - \frac{d}{d\phi}\log\sum_{\ell=0}^{L} p(h_T|\theta_\ell,\pi_\phi)\right] \tag{16}$$

- Gradient is amenable to existing variance reduction methods.
- Unbiased estimates may again be obtained using samples.
- A complete derivation of the gradients estimators are in Appendix C.

Theorem 3 (Permutation invariance)

Theorem 3 (Permutation invariance). *Consider a permutation* $\sigma \in S_k$ *acting on a history* h_k^1, *yielding* $h_k^2 = (\xi_{\sigma(1)}, y_{\sigma(1)}), \ldots, (\xi_{\sigma(k)}, y_{\sigma(k)})$. *For all such* σ, *we have*

$$\mathbb{E}\left[\sum_{t=1}^{T} I_{h_{t-1}}(\xi_t) \middle| h_k = h_k^1\right] = \mathbb{E}\left[\sum_{t=1}^{T} I_{h_{t-1}}(\xi_t) \middle| h_k = h_k^2\right]$$

such that the EIG is unchanged under permutation. Further, the optimal policies starting in h_k^1 *and* h_k^2 *are the same.*

Deep Learning Architecture

- Permutation invariance
  - Summing or combining multiple inputs into a single representation that is invariant to their order.

$$R(h_t) := \sum_{k=1}^{t} E_{\phi_1}(\xi_k, y_k), \qquad (17)$$

- $E_{\phi 1}$ is a neural network encoder with parameters ϕ_1 to be learned.

Theorem 3 (Permutation invariance). *Consider a permutation* $\sigma \in S_k$ *acting on a history* h_k^1, *yielding* $h_k^2 = (\xi_{\sigma(1)}, y_{\sigma(1)}), \ldots, (\xi_{\sigma(k)}, y_{\sigma(k)})$. *For all such* σ, *we have*

$$\mathbb{E}\left[\sum_{t=1}^{T} I_{h_{t-1}}(\xi_t) \middle| h_k = h_k^1\right] = \mathbb{E}\left[\sum_{t=1}^{T} I_{h_{t-1}}(\xi_t) \middle| h_k = h_k^2\right]$$

such that the EIG is unchanged under permutation. Further, the optimal policies starting in h_k^1 *and* h_k^2 *are the same.*

Deep Learning Architecture

$$R(h_t) := \sum_{k=1}^{t} E_{\phi_1}(\xi_k, y_k), \qquad (17)$$

- $R(h_t)$ is a pooled representation.
- $\pi_{\phi(h_t)} = F_{\phi 2}(R(h_t))$ where $F_{\phi 2}$ is a learned emitter network.
- The trainable parameters are $\phi = \{\phi_1, \phi_2\}$.
- Combining simple networks in way that is sensitive to the permutation invariance of the problem
  - $E_{\phi 1}$ is re-used for each input pair and for each time step t.
  - Improved performance compared to networks that are forced to learn the relevant symmetries of the problem.

Experiments (Location Finding)

- This experiment there are K hidden objects or sources in $\mathbb{R}^d, d\epsilon\{1,2,3\}$
- Aims to learn their locations, $\theta = \{\theta_k\}_{k=1}^K$.
- K sources is assumed to be known
- Source is located at θ_k
- We measure at ξ

Method	Lower bound, $\mathcal{L}_{30}$	Upper bound, $\mathcal{U}_{30}$
Random	8.303 ± 0.043	8.322 ± 0.045
Fixed	8.838 ± 0.039	8.914 ± 0.038
DAD	$\mathbf{10.926 \pm 0.036}$	$\mathbf{12.382 \pm 0.095}$
Variational	8.776 ± 0.143	9.064 ± 0.187

Table 1. Upper and lower bounds on the total EIG, $\mathcal{I}_{30}(\pi)$, for the location finding experiment. Errors indicate ± 1 s.e. estimated over 256 (variational) or 2048 (others) rollouts.

DAD vs Fixed baseline

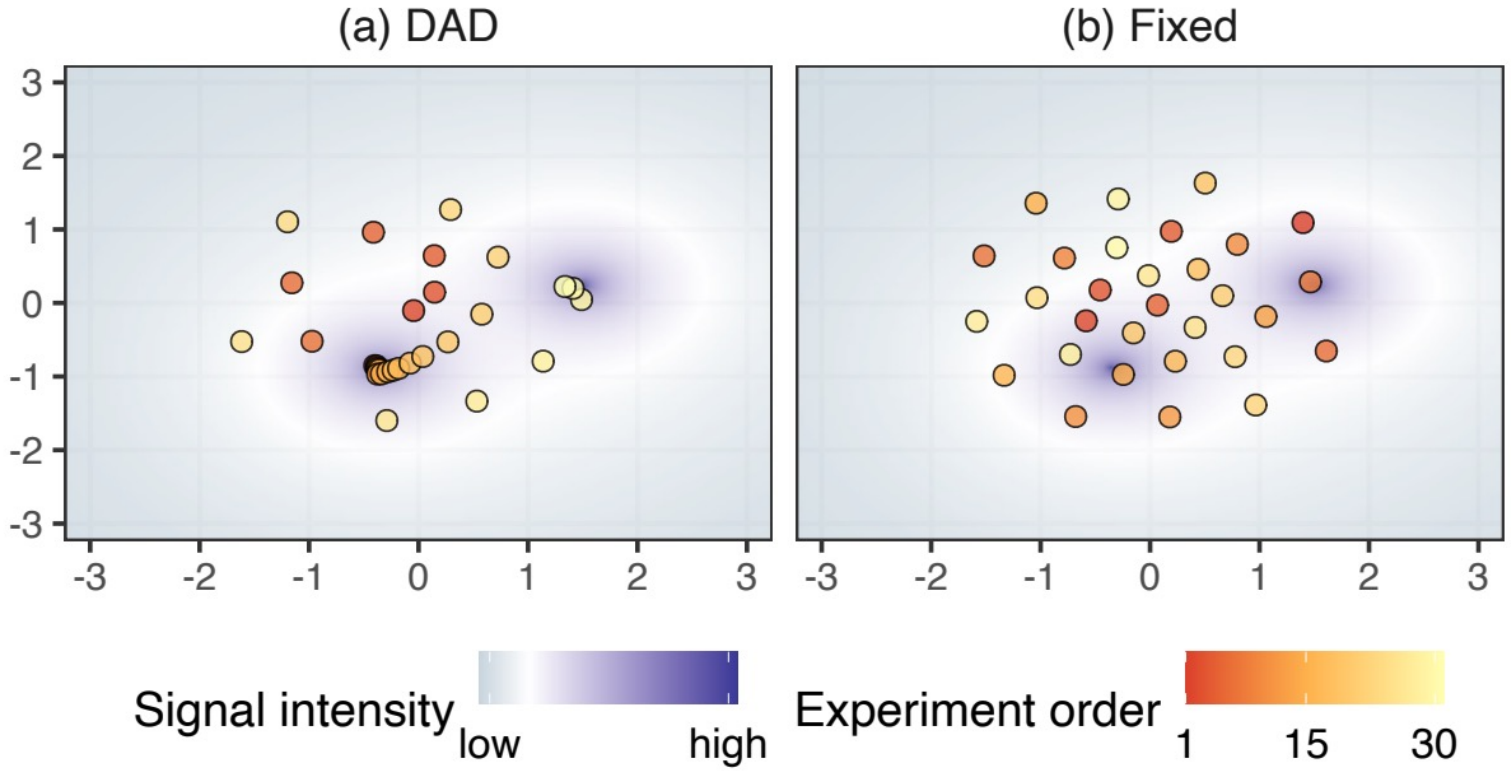


Figure 1. An example of the designs learnt by (a) the DAD network and (b) the fixed baseline for a given θ sampled from the prior.

Experiments (Location Finding)

- Random baseline
  - Select designs uniformly at random.
- Fixed baseline
  - Ignores opportunity for adaptation.
  - Uses static design to learn a fixed $\xi_1, \ldots, \xi_T$ before the experiment.
  - SG-BOED approach with the PCE bound to optimize $\xi_{1:T}$

Method	Lower bound, $\mathcal{L}_{30}$	Upper bound, $\mathcal{U}_{30}$
Random	8.303 ± 0.043	8.322 ± 0.045
Fixed	8.838 ± 0.039	8.914 ± 0.038
DAD	$\mathbf{10.926 \pm 0.036}$	$\mathbf{12.382 \pm 0.095}$
Variational	8.776 ± 0.143	9.064 ± 0.187

Table 1. Upper and lower bounds on the total EIG, $\mathcal{I}_{30}(\pi)$, for the location finding experiment. Errors indicate ± 1 s.e. estimated over 256 (variational) or 2048 (others) rollouts.

Experiments (Location Finding)

- Variational
  - SG-BOED approach
  - Traditional iterative approach to approximate π_s
  - Requires significant runtime computation.
- Reports upper and lower bounds for each strategy

Method	Lower bound, $\mathcal{L}_{30}$	Upper bound, $\mathcal{U}_{30}$
Random	8.303 ± 0.043	8.322 ± 0.045
Fixed	8.838 ± 0.039	8.914 ± 0.038
DAD	$\mathbf{10.926 \pm 0.036}$	$\mathbf{12.382 \pm 0.095}$
Variational	8.776 ± 0.143	9.064 ± 0.187

Table 1. Upper and lower bounds on the total EIG, $\mathcal{I}_{30}(\pi)$, for the location finding experiment. Errors indicate ± 1 s.e. estimated over 256 (variational) or 2048 (others) rollouts.

Deployment times for Hyperbolic Temporal Discounting methods

- Total design time for T = 20 experiments.
- Binary Question:
  - Would you prefer R pounds today or 100 pounds in D days.
  - $With\ design\ \xi = (R, D)$
- Time required to deploy each method is in the table.

Method	Deployment time (s)
Frye et al. (2016)	0.0902 ± 0.0003
Kirby (2009)	N/A
Fixed	N/A
DAD	0.0901 ± 0.0007
Badapted	25.2679 ± 0.1854

Deployment times for Hyperbolic Temporal Discounting methods

- Kirby: A human constructed fixed set of designs
- Frye: A problem specific adaptive strategy
- DAD: Using a fixed designed policy
- Badapted: Partially customized sequential BOED method that use Pop. Monte Carlo

Method	Deployment time (s)
Frye et al. (2016)	0.0902 ± 0.0003
Kirby (2009)	N/A
Fixed	N/A
DAD	0.0901 ± 0.0007
Badapted	25.2679 ± 0.1854

Deployment times for Hyperbolic Temporal Discounting methods

- Total design time for T = 20 experiments.
- Binary Question:
  - Would you prefer R pounds today or 100 pounds in D days.
  - $With\ design\ \xi = (R, D)$
- Time required to deploy each method is in the table.

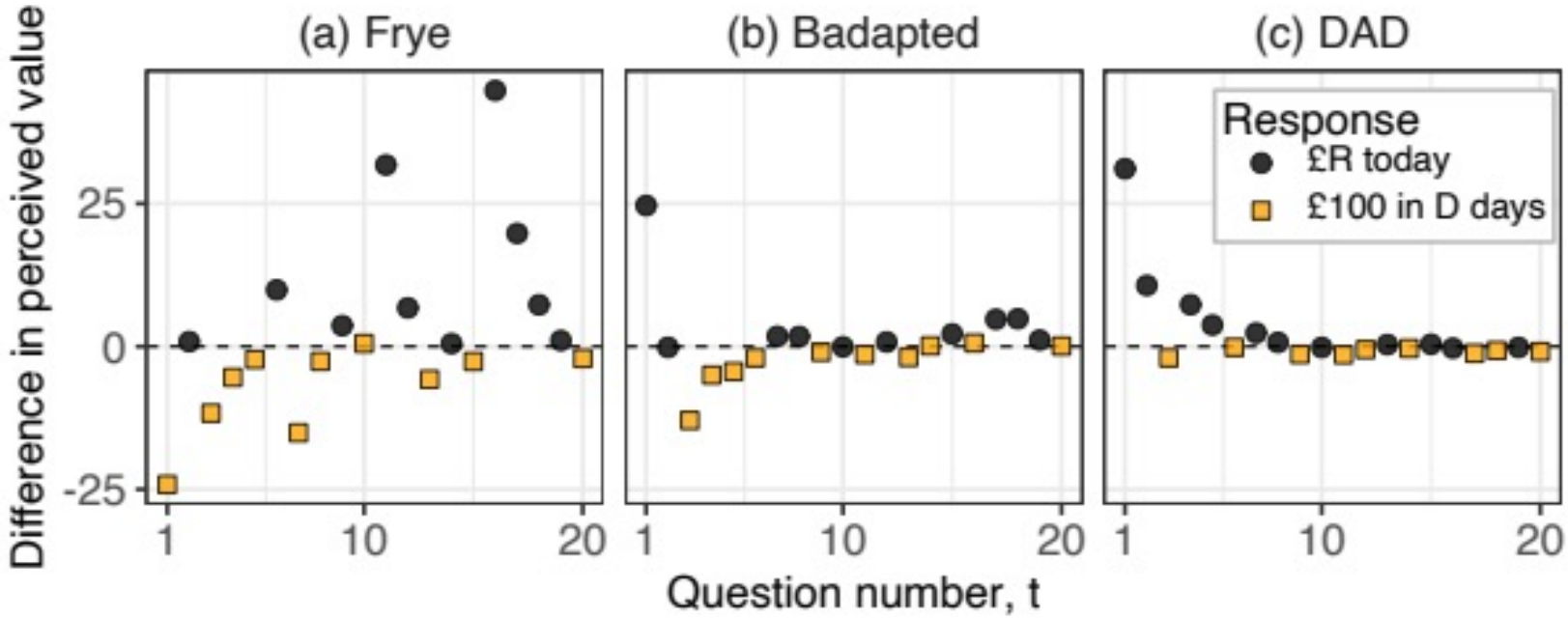
Method	Deployment time (s)
Frye et al. (2016)	0.0902 ± 0.0003
Kirby (2009)	N/A
Fixed	N/A
DAD	0.0901 ± 0.0007
Badapted	25.2679 ± 0.1854

Hyperbolic Temporal Lower and Upper bounds on the total Information

- This table shows the performance of each method.
- The bounds are finite sample estimates of $L_T(\pi, L)$ and $U_T(\pi, L)$ with $L = 5000$.
- The errors indicate ±1 s.e. over the sampled histories.

Method	Lower bound	Upper bound
Frye et al. (2016)	3.500 ± 0.029	3.513 ± 0.029
Kirby (2009)	1.861 ± 0.008	1.864 ± 0.009
Fixed	2.518 ± 0.007	2.524 ± 0.007
DAD	$\mathbf{5.021 \pm 0.013}$	$\mathbf{5.123 \pm 0.015}$
Badapted	4.454 ± 0.016	4.536 ± 0.018

Designs learnt by Frye, Badapted, and DAD



- Observe that DAD and Badapted are similar.
- This demonstrates the accuracy of DAD

Death Process

- The design problem
  - Choose observation times $\xi > 0$ at which to observe the number of infected individuals
  - T = 4

Method	Deployment time (s)	$\mathcal{I}_T(\pi)$
Fixed	N/A	2.023 ± 0.007
DAD	$0.0051 \pm 12\%$	$\mathbf{2.113 \pm 0.008}$
Variational	$1935.0 \pm 2\%$	2.076 ± 0.034
SeqBED*	25911.0	1.590

Discussion

- DAD outperforms non-amortized approaches despite using a tiny fraction of the resources at deployment time.
  - Conventional methods must approximate posterior $p(\theta|h_t)$ at each stage.
  - The policy learned by DAD has the potential to be non-myopic.
    - It does not choose a design that is optimal for the current experiment in isolation, but takes into account that there are more experiments to perform later.

DAD vs Optimal myopic strategies.

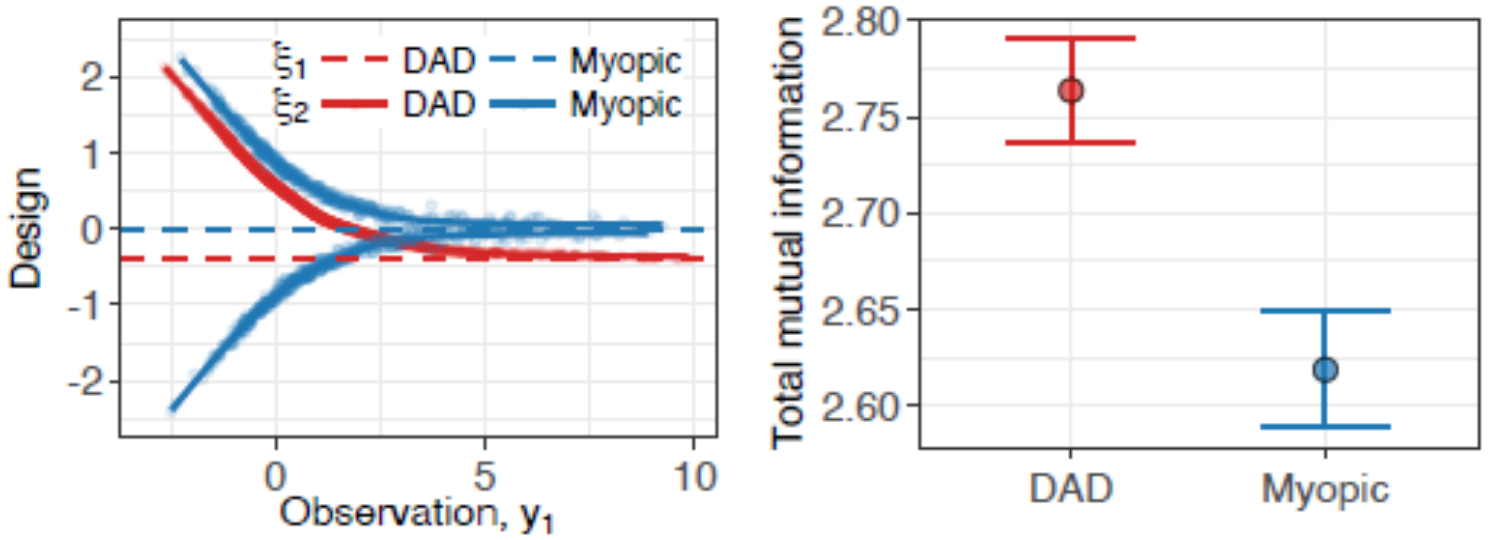


Figure 4. 1D location finding with 1 source, $T = 2$. [Left] the design function, dashed lines correspond to the first design ξ_1, which is independent of y_1. [Right] $\mathcal{I}_2(\pi)$, the total EIG ± 1 s.e.

Conclusion

- Focuses on learning a design policy and can deploying it during the live experiment to **quickly** make adaptive design decisions.
- Eliminates the need to estimate intermediate posterior distributions or optimize over designs
- An approach to allow adaptive BOED to be run in real-teim for general problems.

References

- Deep Adaptive Design: Amortizing Sequential Bayesian Experimental Design, Adam Foster, Desi R. Ivanova, Ilyas Malik Tom Rainforth
- Probabilistic Machine Learning. Murphy, Kevin P.
- https://www.baeldung.com/cs/ml-policy-reinforcement-learning