Robot Physicist

Manoj Kumar, Phil Yeres, Michele Ceru

Abstract—We integrate the necessary components and demonstrate a proof-of-concept for a ”robot physicist” that is able to plan the next experiment needed to most efficiently measure a fundamental constant of nature. We then apply the algorithm to simulated data.

1 INTRODUCTION

PARTICLE physics experiments generally seek to estimate fundamental constants of nature. Depending on the choice of experimental settings, these constants can be measured with varying levels of uncertainty. Our project is designed to demonstrate a proof-of-concept for automating the identification of the experimental settings which minimize uncertainty in the measurement of the quantity of interest in the experiment. This proof of concept can be extended to real physics simulations.

This ”robot physicist” system can be understood using the simple analogy of two experimenters, Alice and Bob. Alice runs experiments which generate data, and she uses the data to update her prior on the distribution of the natural constant of interest. Bob uses the posterior from Alice's experiment to run a set of simulated experiments with various possible experimental settings, and he identifies the experimental settings that maximize the expected information gain. Bob then hands the best experimental settings back to Alice, and Alice repeats the process.

1.1 Notation

The experimental model has three main components:

1) ϕ : Experimental configurations such as the energy required to run the particular experiment.
2) θ: Quantities that we would like to infer from the experiment
3) X: Data generated from the experiment

2 BAYESIAN FRAMING

The data generated by the experiment, x, depends on θ, the constant of nature we seek to measure, as well as Φ, the settings of the experiment. $P(x|\theta,\Phi)$ is the likelihood (the probability of the data given the experimental settings and the constant of nature). Then, using Bayes theorem, we can compute the posterior (the probability distribution for the constant of interest given the observed data and the experimental settings) as follows:

$$P(\theta|x,\Phi) = \frac{P(x|\theta,\Phi)P(\theta|\Phi)}{P(x|\Phi)} \tag{1}$$

As the experimenter varies Φ, this distribution changes. The experimenter's goal is to find the value of ϕ that estimates θ with a minimum of uncertainty.

To find the best ϕ we define the information gain as a function of ϕ and than we find the value of ϕ that maximizes it. The information gain is defined as:

$$EIG(\Phi) = \int P(x|\Phi)\Big[H[P(\theta)] - H[P(\theta|x,\Phi)]\Big]dx \tag{2}$$

where H denotes entropy. We approximate the entropy by discretizing the distribution on θ, as:

$$H[P] \simeq -\sum_{k\geq 1} p_k \log p_k \tag{3}$$

To compute that we need to sample from $P(x|\Phi)$ and know $P(\theta|x,\Phi)$.

- We don't have $P(x|\Phi)$ because the data of the experiment is conditioned on θ as well, consequently we can only sample from the distribution $P(x|\theta,\Phi)$. But we can calculate it using the following:

$$P(x|\Phi) = \int P(x,\theta|\Phi)d\theta = \int P(x|\theta,\Phi)P(\theta|\Phi)d\theta \tag{4}$$

Where the last equality is the expected value of $P(x|\theta,\Phi)$ under the distribution $P(\theta|\Phi)$. Discretising the integral:

$$P(x|\Phi) \simeq \sum_{i=1}^{n} P(x|\theta_i,\Phi)P(\theta_i|\Phi) = \frac{1}{n}\sum_{i=1}^{n} P(x|\theta_i,\Phi) \tag{5}$$

Where in the last equality we are assuming to have an uniform distribution $P(\theta_i|\Phi) = 1/n$. We use the black box to generate the distributions $P(x|\theta_i,\Phi)$ for n values of θ:

$$\begin{aligned} &\theta_1 \rightarrow P(x|\theta_1,\Phi) \\ &\theta_2 \rightarrow P(x|\theta_2,\Phi) \\ &\dots \\ &\theta_n \rightarrow P(x|\theta_n,\Phi) \end{aligned} \tag{6}$$

each of these distribution is obtained from a histogram of the sampled data. Using these we can compute $P(x|\Phi)$.

Since $P(x|\Phi)$ is now discretized because of the histogram, we sample from a multinomial with the parameters, x_i's given by the bin centers and the probability computed from the normalized histogram. This is done from lines 92 to 107 in the code.

- To calculate the posterior $P(\theta|x_j, \Phi)$ where x_j is sampled using the method described above, we can use Bayes theorem:

$$P(\theta|x_j, \Phi) = \frac{P(x_j|\theta, \Phi)P(\theta|\Phi)}{P(x_j|\Phi)} \tag{7}$$

- After doing this, we average out $-H[P(\theta|x_j, \Phi)]$ across all x_j's to calculate the Expected Information Gain.

9.

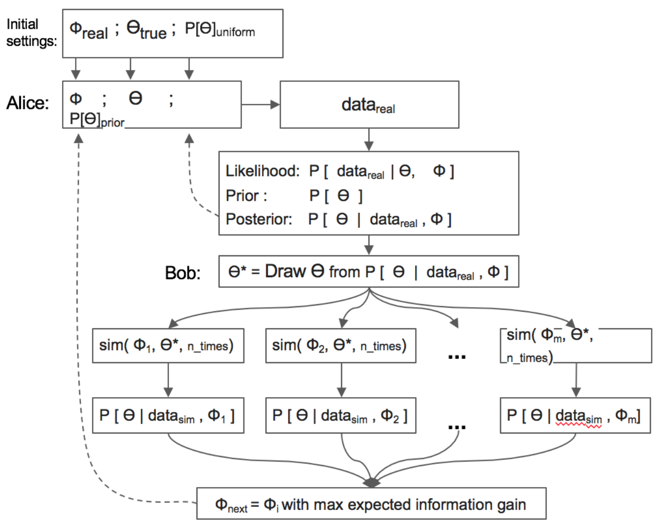

Fig. 1. Flowchart of the program

3 Robot Physicist Proof-of-Concept

3.1 Toy Black-box Simulator

To demonstrate the Robot Physicist algorithm we define a black-box simulator which generates data, X, such that X follows a univariate Gaussian distribution with mean θ and variance $2 + \cos(\phi)$. This proof-of-concept would hold for any black box, but this configuration was chosen for convenience, as ϕ is unconstrained, and variance is always non-negative.

3.2 Robot Physicist Algorithm

3.2.1 Generate a posterior distribution on theta

This step contains two notable processes:

1) Computing the likelihood of X given θ and ϕ:
We generate an empirical pdf using a normalized histogram with samples generated from the black-box using θ and ϕ.
2) Finding an appropriate step size for θ:
Given an lower- and upper-bound for θ (which should be available for most physical constants of interest), and, for convenience, using a univariate Gaussian distribution, $\sigma_{theta_{posterior}}$, which scales as $\frac{\sigma_{theta_{prior}}}{\sqrt{n}}$, with n = the number of samples used to compute the likelihood.

After calculating these components, the computation of the posterior is trivial.

Algorithm 1: Main algorithm

Result: estimate θ_{true} and ϕ_{real}
initialize θ_{true}, ϕ_{real} and $P(\theta)_{prior}$;
while *iter < num iters* **do**
  $P(\theta_{pos})$ = `gen_theta_posterior(`θ_{true}`,` ϕ_{real}`,` $P(\theta)_{prior}$`);`
  $P(\theta_{prior}) = P(\theta_{pos})$;
  ϕ_{real} = `optimize_phi(`θ_{prior}`,` $H(\theta)$`);`
end
function `gen_theta_posterior` (θ, ϕ, $P(\theta)_{prior}$);
X = `black_box(`θ`,` ϕ`);`
$P(X|\theta, \phi)$ = `likelihood(X,` θ`,` ϕ`);`
return `normalize(`$P(\theta) \times P(X|\theta, \phi)$`);`

function `optimize_phi`($P(\theta)$, $H(\theta)$;
return $\arg\max_\phi$ =
$\int P(X|\Phi)(H[P(\theta)] - H[P(\theta|X, \Phi)])\ dX$

3.2.2 Optimize phi

We use the following simplifications to optimize over ϕ in our toy model:

1) We use a greedy procedure in which we compute the information gain for a set of plausible ϕ 's (experimental settings) and we return the ϕ that maximizes the expected information gain.
2) For a fixed value of phi, we approximate $\int P(X|\Phi)(H[P(\theta)] - H[P(\theta|X, \Phi)])\ dX$ with $H[P(\theta)] - \frac{\sum_{i=1}^{N} H[P(\theta|X_i, \Phi)])}{N}$ where N is the number of times the experiment is carried out with the same experimental settings.
3) To estimate the integrated expected information gain, we need to integrate $H[P(\theta|X, \Phi)]$ over θ. We in-turn replace this integration with a point-estimate using the MAP estimate of P(θ)

4 Extension to a Real Physics Simulator

We connected our data pipeline to a real physics experiment simulation provided by Lukas Heinrich (https://github.com/lukasheinrich/higgs-mc-studies). The experiment takes has ϕ = (sqrtshalf, polbeam1, polbeam2), and it generates experimental data (events) as json.

There is substantial overhead associated with running a simulation on the real physics simulator. Generating 100 events takes 15 seconds, and generating 100K events take 100 seconds. To optimize the process, we run simulations in parallel and cache the results:

1) Caching: To avoid starting up the simulator to generate small samples, we cache results for 2,000 combinations of ϕ and θ. We ran the simulator for 10 values of ϕ and 200 values of θ.
2) Parallelization: To generate and cache the data we used five 8-core AWS compute-optimized instances. All 2,000 simulations were completed in 12 hours of wall time (i.e., 60 hours of computing time).

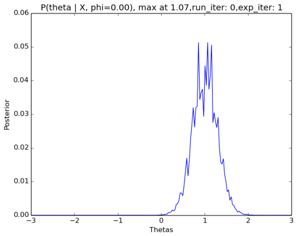

Fig. 2. Toy model: $\phi = 0$

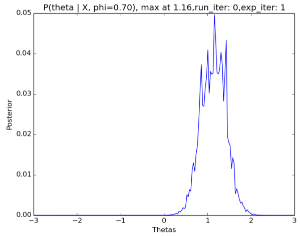

Fig. 3. Toy model: $\phi = 0.70$

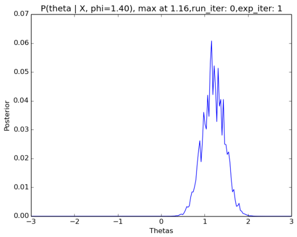

Fig. 4. Toy model: $\phi = 1.40$

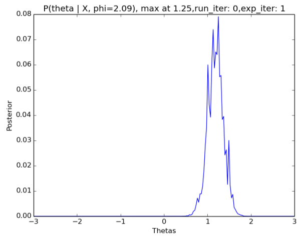

Fig. 5. Toy model: $\phi = 2.09$

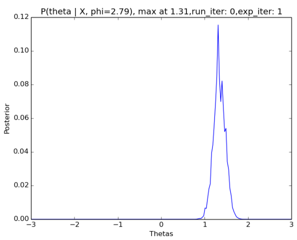

Fig. 6. Toy model: $\phi = 2.79$

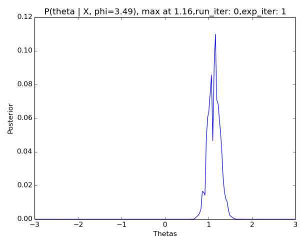

Fig. 7. Toy model: $\phi = 3.49$

5 RESULTS

5.1 Toy Model Results

We ran our toy model with $\theta_{true} = 1.0$ and we scanned θ in the range from -3 to 3 with 200 steps. Then we loop over the possible values of ϕ in the range $\phi = 2 + \cos(w)$ with w in $(0 - 2\pi)$.

In Figures 1 to 6 we plot the posterior of θ for different value of ϕ . As ϕ, the experimental settings, approaches the optimal value, ~ 3, the variance in our measure of the constant of interest, θ, decreases.

The information gain is plotted in Figure 7. We notice that the posterior gets better (with less variance and with mean value closer to θ_{true}) when ϕ approaches the value that correspond to the max information gain.

5.2 Physics Experiment Results

[INSERT PHYSICS EXPERIMENT RESULTS] [INSERT PHYSICS EXPERIMENT RESULTS] [INSERT PHYSICS EXPERIMENT RESULTS] [INSERT PHYSICS EXPERIMENT RESULTS] [INSERT PHYSICS EXPERIMENT RESULTS] [INSERT PHYSICS EXPERIMENT RESULTS]

6 POTENTIAL IMPLEMENTATION IMPROVEMENTS

6.1 Likelihood-free inference

Our toy model demonstrates the case where X is a univariate gaussian. For multi-dimensional X , we can replace the usage of histograms with likelihood-free inference

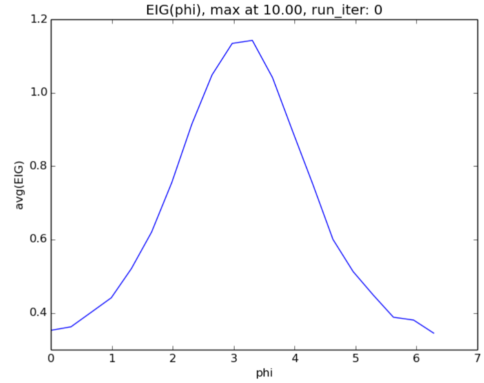

Fig. 8. Toy model: Expected Information Gain

techniques to compute $P(\theta|X, \phi)$ For the actual simulated experiments, the ground truth is not known, and this proof of concept can be extended to make use of a likelihood-free inference toolbox (https://github.com/diana-hep/carl) to estimate $P(X|\theta, \phi)$.

6.2 Bayesian optimization

Currently we optimize over the experimental settings, ϕ, by choosing a range of values with a fixed step size. The optimization over ϕ is a technique that very naturally fits into a bayesian optimization framework since the function that computes the expected information gain is a very expensive function that includes many simulator calls.

6.3 Bob's choice of θ

Computing the expected information gain using the MAP estimate of θ could be replaced by computing the mean information gain by sampling from the posterior.

7 CONCLUSION

We have designed a proof-of-concept for a robot physicist system which is capable of searching for the experimental settings which most effectively measure a quantity of interest. We demonstrated a working example of the system using toy data, we applied the system to real physics simulations, and we outlined next steps to improve the process.

ACKNOWLEDGMENTS

The authors would like to thank Professor Kyle Cranmer for his guidance on this project; Lukas Heinrich for making a physics simulator available to us; Brenden Lake for helping us with Bayesian analysis; and Professor Claudio Silva and our classmates for providing helpful feedback during our project update presentations.

REFERENCES

[1] A. Gelman, *Bayesian Data Analysis*, 2nd ed. Chapman & Hall/CRC Texts in Statistical Science, 2003.
[2] L. Heinrich, *weinberg-exp*, https://github.com/lukasheinrich/weinberg-exp, 2016.

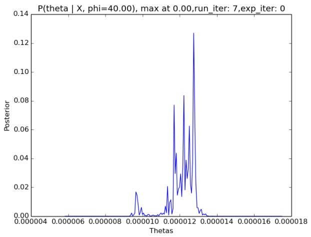

Fig. 9. Physics simulation: $\phi = 40$

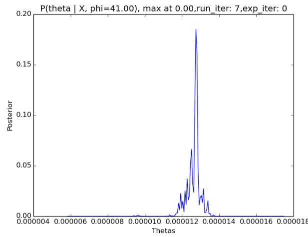

Fig. 10. Physics simulation: $\phi = 41$

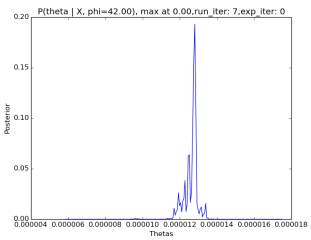

Fig. 11. Physics simulation: $\phi = 42$

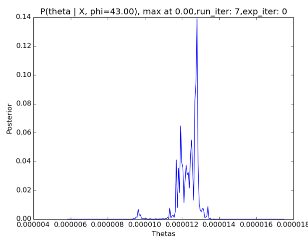

Fig. 12. Physics simulation: $\phi = 43$

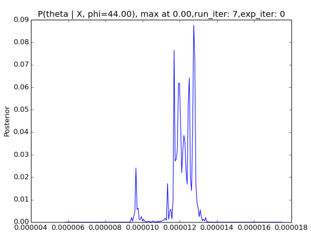

Fig. 13. Physics simulation: $\phi = 44$

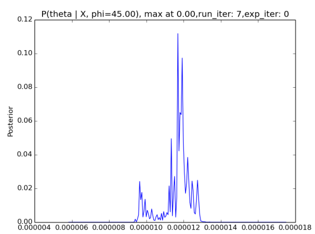

Fig. 14. Physics simulation: $\phi = 45$

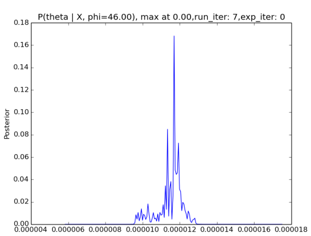

Fig. 15. Physics simulation: $\phi = 46$

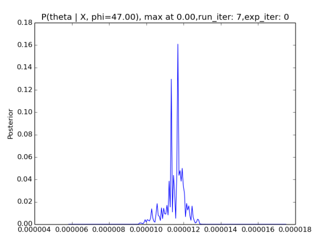

Fig. 16. Physics simulation: $\phi = 47$

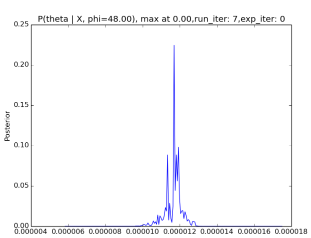

Fig. 17. Physics simulation: $\phi = 48$

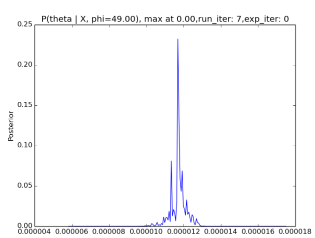

Fig. 18. Physics simulation: $\phi = 49$

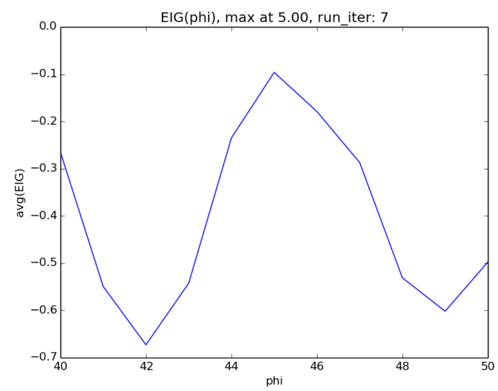

Fig. 19. Physics simulation: Expected Information Gain