Solving Inverse Optimization Problems via Bayesian Optimization

Yen-An Lu,a Vikram Kumar,a Wei-Shou Hu,a Joel Paulson,\*b Qi Zhang\*a

a Department of Chemical Engineering and Materials Science, University of Minnesota, Minneapolis, MN 55455, USA

b Department of Chemical and Biomolecular Engineering, The Ohio State University, Columbus, OH 43210, USA

\* paulson.82@osu.edu, qizh@umn.edu

Abstract

Data-driven inverse optimization refers to the learning of unknown optimization models from optimal or near-optimal solutions of that optimization problem. As such, it can be used to uncover hidden decision-making processes, assuming that they can be described as optimization problems. However, the inverse optimization problem is commonly formulated as a large-scale bilevel program, which is often very difficult to solve, especially when the lower-level problems are nonconvex. In this work, we propose a black-box optimization approach based on Bayesian optimization to tackle general inverse optimization problems. Here, the objective function is approximated using a probabilistic surrogate model and can, for fixed parameter values, be evaluated by directly solving the given lower-level optimization problems. In a computational case study, we apply the proposed method to estimate the missing parameters in a standard pooling problem that is nonlinear and nonconvex. The results demonstrate the ability of the proposed framework to accurately estimate the model parameters with small numbers of data points and Bayesian optimization iterations.

**Keywords**: Inverse optimization, Bayesian optimization, decision making.

* 1. Introduction

Data-driven inverse optimization (IO) is an emerging approach to learning the explicit rules behind complex decision-making processes (Chan et al., 2021). It leverages mathematical optimization as a model for decision making, where observed decisions are assumed to be the optimal or near-optimal solutions to an underlying optimization problem. The primary objective of IO is to decipher the optimization model that best mirrors an agent's decision-making patterns based on their past decisions. A key strength of IO is its ability to directly account for constraints, thereby harnessing the full modeling versatility of mathematical programming and providing a natural mechanism to incorporate domain knowledge, ultimately yielding interpretable decision models.

Inverse optimization problems (IOPs) are typically formulated as bilevel programs, which involve as many lower-level problems as there are training data points. Each lower-level problem represents an instance of the forward optimization problem (FOP) with the corresponding model inputs and observed decisions, whereas the objective of the upper-level problem is to estimate model parameters that yield FOP solutions that best align with the observed decisions. Common approaches to solving IOPs apply single-level reformulations and cutting-plane methods (Chan et al., 2019; Keshavarz et al., 2011). Recent works have also proposed decomposition methods to handle IOPs with high-dimensional FOPs and large datasets (Gupta and Zhang, 2022, 2023). However, these existing exact solution methods often become intractable when the FOP is nonconvex.

In this work, we propose to solve general IOPs using Bayesian optimization (BO) where we treat the loss function of the IOP as a black box. Here, the key advantage of BO is that, at each iteration, it only requires the evaluation of the loss function with the current parameter estimates; this can be achieved by directly solving the FOP for every data point, which circumvents the need of a single-level reformulation or a cutting-plane algorithm. Consequently, BO remains applicable even when the FOP is nonlinear and nonconvex. In a computational study, we demonstrate that the proposed BO framework can efficiently identify robust estimates of model parameters in a nonlinear and nonconvex pooling problem with relatively small numbers of iterations and observations, particularly when the number of unknown model parameters is limited.

* 1. Mathematical formulation

We assume that the decision-making process of interest can be formulated as an optimization problem of the following general form, which we refer to as the FOP:

|  |  |
| --- | --- |
|  | (1) |
|  |

where objective function and constraint functions are parameterized by , denotes the -dimensional vector of decision variables, and denote the contextual inputs that describe the system conditions. Here, we assume that the model parameters are unknown; however, we can measure decisions under varying input conditions for a given set of observations . We further assume that the observations are noisy due to, for example, measurement errors, suboptimal decisions, or model mismatches. The goal of IO is to estimate the unknown such that the model predictions best fit the observed decisions, which gives rise to a data-driven IOP that can be formulated as the following bilevel optimization problem (Gupta and Zhang, 2023):

|  |  |
| --- | --- |
|   | (2) |
|  |

where the objective of the upper-level problem is to obtain estimates of the model parameters, denoted by , that minimize the decision loss defined as the weighted sum of squared residuals of predicted and observed decisions. Here, denotes the diagonal matrix of weighting factors. The bilevel optimization problem (2) is commonly solved using a single-level reformulation where the lower-level problems are replaced by their optimality conditions; however, this is only applicable when the FOP is convex. To address nonconvex FOPs, we propose a BO-based approach as described in the following.

* 1. Solution approach

In this work, we treat the upper-level objective of problem (2) as a black-box function and directly optimize it through BO (Frazier, 2018). Problem (2) is thus converted to the following black-box optimization problem:

|  |  |
| --- | --- |
|   | (3) |

* + 1. Gaussian process regression

We approximate the loss function with a Gaussian process (GP) surrogate , where the prior GP distribution is specified by the choice of the mean function and covariance (kernel) function (Rasmussen and Williams, 2005). By maximizing the log marginal likelihood, the hyperparameters of the kernel are calibrated to a set of past evaluations , where denotes the evaluation of the decision loss obtained by solving FOPs at .

We consider a zero mean, which can be achieved by normalizing the output data, and focus on the stationary covariance functions from the Mateŕn class. Conditioned on the evaluations, the predicted posterior distribution of the function at a future input  remains Gaussian with the following posterior mean and covariance:

|  |  |
| --- | --- |
|  | (4) |
|  |

where , , and is a matrix whose entry is .

* + 1. Bayesian optimization

The BO framework aims to minimize the function over the input domain by sequentially querying particular values based on the posterior of the GP surrogate that provides uncertainty quantification of function over the input domain. In each BO iteration the next sample point is queried by optimizing an acquisition function that is defined in terms of the posterior information and the past evaluations :

|  |  |
| --- | --- |
| . | (5) |

We use the expected improvement (EI) acquisition function (Jones et al., 1998), where the GP surrogate conditioned on the past evaluations is denoted by , and denotes the minimum decision loss across the previous evaluations (also known as the incumbent). Based on the new query point , the true loss function value is evaluated by solving the solving FOPs. The dataset is then appended with the new evaluation to create a concatenated dataset . The new dataset is later used to update the GP model, and the same process is repeated until the termination criterion is met. A pseudocode for the BO algorithm is shown in Algorithm 1.

|  |
| --- |
| **Algorithm 1** Bayesian optimization for inverse optimization |
| **Input**: GP prior and , input domain , initial data , total number of iterations , and experiments with pairs of and corresponding FOPs |
| 1: | **for**  **do** |
| 2: | Update GP model using  |
| 3: |   |
| 4: | Evaluate by solving FOPs at current  |
| 5: |   |
| 6: | **end for** |
| **Output:** Optimal solution  |

* 1. Computational case study

We consider a standard pooling problem where an operator blends a set of feedstocks in a pooling network to create various final products that meet desired qualities and demands while minimizing the total cost. Provided below is the formulation of a standard pooling problem (Misener and Floudas, 2009):

|  |  |
| --- | --- |
|   | (6a) |
|   | (6b) |
|  |   | (6c) |
|  |   | (6d) |
|  |   | (6e) |
|  |   | (6f) |
|  |  | (6g) |
|  |   | (6h) |
|  |   | (6i) |
|  | ,  | (6j) |

whereis the set of input feedstocks, is the set of mixing pools,and is the set of output products. As incoming feedstocks can connect to a pool ordirectly to an output, sets , , and denote the existing streams from input to pool , pool to output , and input to output, respectively. The cost per unit of feedstock is denoted by . The revenue per unit flow from poolto output and input to output are denoted byand *,* respectively. We use variables ,, and to denote the flow from input to pool , pool to product , and input to output , respectively, whereas the quality level in pool is denoted by .

In problem (6), we assume that each feedstock has a limited availabilityas indicated in constraints (6b); the material and quality balance at pool are maintained through constraints (6c) and (6d), respectively; the upper acceptable product quality of each product is set in (6e). We consider a scenario in which the demand for each product ( in constraints (6f)) is known; however, due to the limited input availability, the operator has varying unknown preferences on the minimum demand they would like to satisfy for each product , where denotes the minimum fraction of demand that needs to be satisfied. Notably, constraints (6e) contain bilinear terms that render the overall optimization problem nonconvex. The goal is to apply IO to estimate the values by observing a part of the decisions, namely and , based on varying input conditions in a set of experiments .

We test the proposed BO framework on two benchmark pooling networks, Haverly1 and Foulds2 (Adhya et al., 1999), where their network specifications () are (3,1,2) and (6,2,4), respectively. For each random instance of the IOP, we first generate a set of ground truth and create a training dataset of experiments with randomized demand , feedstock availability , upper acceptable quality , and revenue and . Lastly, we solve these experiments of problem (6) to collect the true optimal solutions ), and a Gaussian noise is added to generate noisy observations of decisions , where A separate set of testing data is generated based on the same randomization procedure with 25 experiments. The computational statistics are obtained from the results of 10 random instances of each IOP.



Figure 1: Effect of network size and dimensionality of () on the accuracy of the estimated FOPs. Convergence analysis of (a) training loss, (b) prediction loss, and (c) loss on haverly1 and foulds2 networks with varying . Training and prediction loss describe the normalized decision loss on the training and testing datasets, respectively, whereas loss denotes the difference between the ground truth and estimate values. Here the solid lines and shaded areas respectively denote the medians and confidence intervals of the corresponding loss across the 10 random instances.

We first implement the proposed method to solve the IOPs for the two networks with under varying dimensionality of ), i.e., number of is selected to be unknown. *Figure* 1 shows the convergence of three different types of losses over the BO iterations, including decision loss on the training and testing data as well as loss compared to the ground truth values. A similar trend in loss convergence is observed in all problems, where a rapid and consistent decrease is observed in the first 20-30 iterations followed by small and discrete improvements. The estimated models in all cases show good generalizability in predicting the decisions under unseen conditions, as a similar convergence between training and prediction losses is observed. A comparable rate of convergence is obtained in the two benchmark networks with , indicating that the algorithm is robust with respect to the network size, which directly affects the number of decision variables in the FOP. While the algorithm shows a slower convergence for increasing in the foulds2 problem, it converges to good estimates of that show accurate predictions on decisions within 100 iterations.

Next, we examine the impact of the size of training datasets on the prediction accuracy of the estimated FOPs at 100 BO iterations, as shown in *Figure* 2a. A monotonic decrease in prediction loss is observed with increasing up to 150 data points; however, 50-100 data points seem to be enough to estimate models that provide accurate predictions under the given 10 random instances. Lastly, we test the algorithm performance under varying noise levels in terms of the observed decisions (*Figure* 2b), where the algorithm provides robust estimates even under relatively high measurement noises (). These results demonstrate that the IO approach is data-efficient and robust in learning unknown optimization model parameters, which can be especially advantageous when experiments for collecting decision data are expensive and subject to high levels of noise.



Figure 2: Effect of (a) training data size and (b) noise level on the prediction and error of the estimated models based on the foulds2 network with The boxplots show the interquartile ranges of the corresponding losses of the estimated models obtained at 100 BO iterations with 10 different random instances of training datasets.

* 1. Conclusions

In this work, we proposed a solution algorithm for solving general inverse optimization problems, where the goal is to learn unknown parameters of optimization models. This was achieved by treating the objective of an IOP as a black box and iteratively optimizing it using Bayesian optimization. To assess the performance of the algorithm, we apply it to a nonlinear and nonconvex pooling problem with unknown model parameters. The results demonstrate the effectiveness and robustness of the proposed method for estimating model parameters based on a small set of noisy data within a limited number of BO iterations.

References

Adhya, N., Tawarmalani, M., and Sahinidis, N.V., 1999. A Lagrangian approach to the pooling problem. *Industrial and Engineering Chemistry Research*, *38*(5), 1956–1972.

Chan, T. C. Y., Lee, T., and Terekhov, D., 2019. Inverse optimization: Closed-form solutions, geometry, and goodness of fit. *Management Science*, *65*(3), 1115–1135.

Chan, T. C. Y., Mahmood, R., and Zhu, I. Y., 2021. Inverse optimization: Theory and applications. *ArXiv Preprint,* ArXiv:2109.03920.

Frazier, P. I., 2018. *A Tutorial on Bayesian Optimization*. *ArXiv Preprint,* ArXiv:1807.02811.

Gupta, R. and Zhang, Q., 2022. Decomposition and Adaptive Sampling for Data-Driven Inverse Linear Optimization. *INFORMS Journal on Computing*.

Gupta, R. and Zhang, Q., 2023. Efficient learning of decision-making models: A penalty block coordinate descent algorithm for data-driven inverse optimization. *Computers and Chemical Engineering*, *170*, 108123.

Jones, D. R., Schonlau, M., and Welch, W. J., 1998. Efficient Global Optimization of Expensive Black-Box Functions. *Journal of Global Optimization*, *13*(4), 455–492.

Keshavarz, A., Wang, Y., and Boyd, S., 2011. Imputing a convex objective function. *IEEE International Symposium on Intelligent Control - Proceedings*, 613–619.

Misener, R., and Floudas, C. A., 2009. Advances for the pooling problem: Modeling, global optimization, and computational studies Survey. *Applied and Computational Mathematics*, *8*(1), 3–22.

Rasmussen, C. E., and Williams, C. K. I., 2005. Gaussian Processes for Machine Learning. *The* *MIT Press*.