Predict-then-Optimize v/s Probabilistic Approximations: Tackling Uncertainties and Error Propagation

Priya Shanmugasundaram [1] **Saurabh Jha** [1] **Kumar Muthuraman** [2]

Abstract

Proactive planning is a key necessity for businesses to function efficiently under uncertain and unforeseen circumstances. Planning for the future involves solving optimization problems, which are often naturally convex or are modeled as convex approximations to facilitate computation. The primary source of uncertainties in the real world that business are dealing with (eg. demand) cannot be reasonably approximated by deterministic values. Hence deterministic convex optimization approximation do not not yield reasonable solutions. Classically, one relies on assumptions on the data generating process (like for eg. that demand is log normal) to formulate as a stochastic optimization problem. However, in today's world, such major uncertainties are often best predicted by machine learning methods. In this paper, we propose a novel method to integrate predictions from machine learning systems and optimization steps for a specific context of a resource utilisation problem that faces non-stationary incoming workload. The proposed solution is robust and shows improved performance against using the traditional point-predictions directly in the optimization. The proposed solution can be easily extended to different kind of machine learning methods and objective functions.

1. Introduction

Data-driven decision making is a key competitive advantage to businesses of today's world. Statistical and machine learning based methods are widely adopted to analyze data, identify patterns and make predictions. The results and inferences from these methods are then used in decision-making processes, which can be thought of a decision pipeline which begins with raw data as input and the output as impactful real-world business decisions. This decision pipeline that can be decomposed into Prediction and Decision-making modules. The prediction module generates approximations of unknown parameters from data. The decision-making module might use the inferences from the prediction module to help the business make a decision or use the results from the prediction module as input to other sub-routines that help them generate optimal decisions. To generate optimal decisions, these predicted parameters could be used in the objective function of an optimization problem which are then solved using optimization algorithms. Data-driven decision making under uncertainty can be impactful across domains like resource utilization, headcount planning, transport planning and inventory management.

In this paper, we address a resource utilisation problem which faces non-stationary incoming workload. The resource could be anything ranging from time or money to manpower or equipment. Resource availability is influenced by various parameters like the incoming workload, resource outages and carry-over workload from the past. The incoming workload is uncertain and predicted using machine learning models. The goal of the optimization problem is to minimize the expected costs related to utilisation when dealing with non-stationary incoming workload while ensuring that resource health is maintained.

Despite incredible improvements in the area of machine learning, predictions will still be different from future observations cause of the natural uncertainties in the real world. Predictions are hence best interpreted as the expectation of a random variable that will be revealed in the future. Hence ignoring the uncertainty, as in a predict first and then optimize using the prediction results in sub optimal future performance of optimization solutions.

The proposed solution solves these key challenges and develops an enhanced decision pipeline that connects the prediction module and the decision-making module. The solution models the probability distributions for predictions instead of a single point prediction so that the difference in the true values can be safely estimated to lie within the 95% confidence interval of our predicted distributions. In

[1]CFS AI Research, Dell Inc., USA [2]McCombs School of Business, University of Texas, Austin, USA. Correspondence to: Priya Shanmugasundaram <priyashan943@gmail.com>.

The Many Facets of Preference Learning Workshop at the International Conference on Machine Learning (ICML), Honolulu, Hawaii, USA, 2023.

the decision-making module, the proposed solution casts an optimization problem with the predicted parameters as random variables. The proposed solution is robust to uncertainty as the optimization objective is designed to account for the error-propagation from the prediction module. The proposed solution does not require extensive mathematical modelling for different kind of optimization objectives and can be easily extended to different kinds of machine learning methods and optimization problems. The solution is unique compared to existing solutions as it models the predicted parameters as random variables to circumvent the problem of developing differentiable approximations for optimization regret to induce it into the prediction error for back-propagation.

2. Related Work

Planning with parameters which have uncertainty (forecast-based parameters) is usually modelled as a stochastic optimization problem. These uncertainties are tackled using methods like two-stage stochastic programming (Shapiro et al., 2021) or by using approximations of the objective function (Elmachtoub & Grigas, 2022) to be used during back-propagation so that the prediction module can learn to make predictions that are sensitive to the optimization problem as well.

Existing Optimization under Uncertainty Methods: The randomness in uncertain parameters in optimization problems are modelled using probability distributions in stochastic programming (SP) (Birge & Louveaux, 2011) methods. The two-stage SP methods have recourse variables which can be corrected after uncertainty is revealed and re-optimized, however these two-stage SP problems are computationally expensive as the computational time grows with the number of scenarios. The second class of methods are chance-constrained optimization problems (Li et al., 2008) where the goal is to optimize an objective while ensuring that the constraints need to be satisfied with a specific probability in an uncertain environment. The probability of constraint satisfaction involves a multivariate integral and this can be computationally expensive. The third class of algorithms, robust optimization (Lappas & Gounaris, 2016), hedges against the worst case within the uncertainty set which gives rise to the largest constraint violation. However, these methods can lead to very conservative solutions as they are optimizing against the worst case.

Existing Predict + Optimization Methods: In (Amos & Kolter, 2017), the authors propose a quadratic program solver that offers exact gradients for backprop by differentiating the Karush Kuhn Tucker (KKT) (Karush, 2013);(Kuhn & Tucker, 2013) optimality conditions and combining it with the prediction error. In (Elmachtoub & Grigas, 2022), the differentiable surrogate loss is proposed for a constrained linear optimization problem. In (Kong et al., 2022), they propose a method for stochastic optimization using energy models for a differentiable optimization layer. However, these methods require extensive mathematical modelling to derive differentiable approximations to loss functions and have not been studied extensively as research in this area is quite nascent with most literature on LPs and QPs.

3. Proposed Solution

The novel solution proposed in this paper is an enhanced decision pipeline that connects an upstream forecasting module with a downstream optimization module using probabilistic approximations for the predicted parameters to generate collective reduction in the prediction error and the optimization regret. The decision pipeline consists of novelties and improvements in the following main components, 1) The Prediction module, and, 2) The Optimization module. The prediction module generates a probability distribution of the predicted features learned based on historical data. This probability distribution of the predicted features is then used in the Optimization module to solve an optimization problem. The prediction module uses the predictions and the ground truth realizations, to construct distributions of the predicted parameters. The optimization module tries to minimize an objective function defined on the predicted parameters by using the probability distribution generated in the prediction module.

3.1. Prediction Module

Predictive models are widely being used to learn patterns in underlying data such that new predictions can be made from unseen data. In real-world systems, these models operate autonomously under larger processes where their predictions are leveraged to optimize decision-making. Supervised Learning methods are commonly used for such predictive models which can be trained and validated using historical data to assess their accuracy and performance. The optimization problem is dependent on features $y = y_o \cup y_u$ that can be decomposed into observed features y_o and unobserved features y_u. The unobserved features are predicted by the prediction module using the input parameters x_i that are correlated to the unobserved feature y_{u_i}.

The training dataset $D : \{(x_1, y_{u_1}), ..., (x_n, y_{u_n})\}$ is used to train a predictive model f parameterised by θ such that $\hat{y_u} = f(x, \theta)$. The samples $(x_{u_i}, y_{u_i}) \sim P$ where the true distribution P is unknown. The proposed solution models the distribution $p(y_u)$ to minimize the cost associated with this policy. The residuals $e(\hat{y_u}, y_u)$ are defined as the difference between the predicted unobserved features $\hat{y_u}$ and the realized unobserved features y_u, as shown below,

$$e(\hat{y_u}, y_u) = \hat{y_u} - y_u \tag{1}$$

The probability distribution of the residuals $p(e)$ has zero mean (μ) and σ as the empirical standard deviation between the predicted and true values, $\hat{y_u}$ and y_u. The distribution for $p(y_u)$ is derived from the residual distribution where the mean $\mu = \hat{y_u}$. The objective function for the optimization module is given by minimizing an expected cost over the unobserved features y_u. To simplify the setting, we assume that the $\hat{y}_u$ are discrete and take values like $y_{u_1}, ..., y_{u_k}$ with the probabilities given as $p(y_{u_i})$. These discrete probabilities are then used succinctly in the optimization objective as follows,

$$o_{y,v} = \mathbb{E}_{\sim p(y_u)}[g_c(y_u, y_o, v)], \tag{2}$$

where an expectation over $p(y_u)$ for the cost function $g_c(y_u, y_o, v)$ is calculated. The parameters are modelled as random variables as opposed to single-point values as they can capture the temporal correlations and underlying uncertainty better. The use of probability distributions in the downstream optimization problem instead of a single point value can help increase the robustness of the solution to uncertainties like in situations when the true values are different from the predicted values.

3.2. Optimization Module

Combinatorial Optimization (Wolsey & Nemhauser, 1999) methods work under the assumption that all the parameters are fixed and known beforehand. However, in the current setup some of the parameters are estimates generated by the prediction module and are not known with certainty. A combinatorial optimization problem, can be defined as follows (Boyd & Vandenberghe, 2004), where y are the parameters of the optimization problem, v are the decision variables,

$$\begin{aligned} \min_{y} \quad & g_c(y) \\ \text{s.t.} \quad & C(y, v) \end{aligned} \tag{3}$$

$g_c(y)$ is the objective to be minimized and $C(y, v)$ are the set of constraints to be obeyed while obtaining the optimal solution. The constraint set $C(y, v)$ determines the feasible region $S \subseteq \mathbb{R}^d$, where d is the dimension of the decision space. The optimal objective $w^*(g_c) \in W^*(g_c)$, the set of optimal decisions corresponding to g_c lies in the feasible region S. The parameters y of the problem contains y_u which are predicted by the upstream prediction module and used in the cost function $g_c(y, v)$. However, the true values of the parameters are not known during the time of solving the problem and thus the predicted values $\hat{y}_u$ are used instead. The predictive model f produces as output $\hat{y}_u = f(x)$ for the input parameters x. The cost function $g_c(\hat{y_u}, y_o, v)$ is used to produce the optimal decision $w^*(g_c)$. The proposed solution is different from the traditional predict-then-optimize based solutions. The optimization objective in our solution is defined as follows, where $p(\hat{y}_u)$ refers to the probability of the predicted parameters obtained from the residual probability distribution $p(e)$ in the prediction module.

$$\begin{aligned} \min_{v} \quad & \sum_{y_u} p(y_u) g_c(y_u, y_o, v) \\ \text{s.t.} \quad & C(y_u, y_o, v) \end{aligned} \tag{4}$$

In our solution, we formulate a resource utilization problem as an optimization problem where a certain number of available resources need to be utilised optimally when faced with non-stationary incoming workload by adjusting decision variables to increase productivity. Over-utilisation of the available resources affects the health of the resource and also introduces additional cost, while under-utilisation of the available resources or outages causes subdued performance. Carry-over workloads can be used to reduce over-utilisation but might also encourage under-utilisation, if not properly constrained. The incoming workload to be handled by the resources, (y_u) is the unobserved parameter predicted by the supervised learning based prediction module, while the observed parameters include the hourly penalties and bonuses $(c_z, c_o, c_i, c_p, c_u)$ for the under/over-utilization of the resources. The probability values $p(y_u)$ estimated from the empirical PDF in the prediction module are used. The optimisation problem is defined as follows,

$$\begin{aligned} z = \min \quad & \sum_{y_u} p(y_u)[(y_u - (k + v_o - v_i - v_p))_+ * c_z \\ & - v_i * c_i + v_o * c_o - v_p * c_p)] \\ \text{s.t.} \quad & (y_u - (k + v_o - v_i - v_p)) \geq 0 \\ & -r_a * I \leq v_i \leq r_a * I \\ & 0 \leq v_o \leq r_a * O \\ & 0 \leq v_p \leq r_a * P \end{aligned} \tag{5}$$

The objective function is a linear combination of the terms and costs associated with extra utilisation (c_o, c_z), outages (c_p) and carried-over workload in hours from previous weeks (c_i) which need to be corrected in the upcoming weeks. The magnitude of the cost coefficients are designed in such a way that over-utilisation is penalised more compared to under-utilisation as the latter is expensive. The corresponding decision variables (v_o, v_i, v_p), prescribe how the working hours of the resources need to be adjusted optimally to minimize the total expected cost. The positive side of the difference $(y_u - (k + v_o - v_i - v_p))_+$ is used in

the objective function to penalise cases where the resources are not used to capacity resulting in subdued performance. The constraints reinforce the allowed ranges for the decision variables based on their corresponding limits.

Usually, the optimization problem is either modelled as a two-stage stochastic program (Shapiro et al., 2021), with recourse variables or in the predict-and-optimize setup, (Elmachtoub & Grigas, 2022), differentiable approximations of the optimization regret is added to the prediction error to enhance decision making. However, our method provides a simple and elegant solution to account for uncertainty in a way that the solution does not become computationally expensive like in stochastic programs or mathematically intense like in predict-and-optimize settings. The proposed solution is robust to uncertainty and can be easily extended to different kind of optimization problems as well as prediction methods.

Algorithm 1 Decision Pipeline for Optimization

Input: dataset $D(x, y_u) : \{(x_1, y_{u_1}), ..., (x_n, y_{u_n})\}$,
Prediction Module:
Obtain predictions $\hat{y}_u = f(x)$; f is the predictive model,
Find residuals $e(\hat{y}_u, y) = \hat{y}_u - y$,
Derive distributions $p(e)$ and $p(\hat{y}_u)$,
Optimization Module:
Sample from distribution $p(y_u)$,
Solve optimization problem $z = \min \sum_{y_u} p(y_u)[(y_u - (k + v_o - v_i - v_p))_+ * c_z - v_i * c_i + v_o * c_o - v_p * c_p)]$,
s.t. $-r_a * I \leq v_i \leq r_a * I$,
s.t. $0 \leq v_o \leq r_a * O$
s.t. $0 \leq v_i \leq r_a * P$,
s.t. $(y_u - (k + v_o - v_i - v_p)_+ = max(0, (y_u - (k + v_o - v_i - v_p))$
Results: Obtain optimal cost z^* and decision variables $v^* = [v_o^*, v_i^*, v_p^*]$

4. Results

In this section, we present the results and discussions for an implementation of our proposed approach for a real-world resource planning scenario. The performance of our algorithm is studied and compared with the predict-then-optimize methods. The optimization cost and decision variables are examined to evaluate the robustness of the residual distribution and its dependence on the predictions.

We have used the open-source library PySP (Watson et al., 2012), which supports formulating, solving, and analyzing optimization models. Coin-OR CBC (Forrest et al., 2023), has been used as the back-end solver to simulate and solve the optimization problems under different problem settings. We considered the resource allocation problem described in section 3.2 as the combinatorial optimization problem to be solved. PyEPO (Tang & Khalil, 2022), a PyTorch-based end-to-end predict-then-optimize library in Python has been used to study the methods which use surrogate loss to solve the optimization problem.

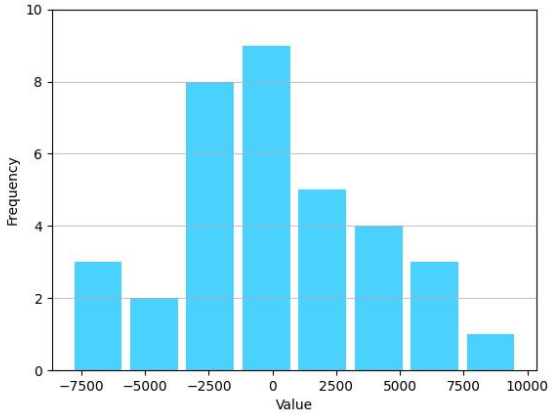

Figure 1. Residual Distribution $p(e)$.

We can see from figure 1, that the residual distribution, after our machine learning task is completed, is centered around a mean μ = 0 with a standard deviation σ = 5000. The predictions come from an existing neural network that has been continuously trained, updated and maintained for providing point wise predictions. We use the models past predictions along with observed values to find empirical distributions.

The residual distribution is derived for the unobserved parameter y_u from the residual errors between the predicted $\hat{y}_u$ and the realised true value y_u. The predicted parameter $\hat{y}_u$ represents the expected incoming workload (in hours) that need to be managed by the available resources r_a. The distribution $p(y_u)$ for the predicted parameter is obtained from the residual distribution by centering it around a mean $\mu = \hat{y}_u$. The distribution $p(y_u)$ is shown in figure 2.

The optimization problem is initially solved directly using the point-prediction $\hat{y}_u$, like in traditional predict-then-optimize problem settings, where the problem definition looks as below,

$$\begin{aligned} z = \min \quad & [(y_u - (k + v_o - v_i - v_p))_+ * c_z \\ & - v_i * c_i + v_o * c_o - v_p * c_p)] \\ \text{s.t.} \quad & (y_u - (k + v_o - v_i - v_p))_+ \geq 0 \\ & - r_a * I \leq v_i \leq r_a * I \\ & 0 \leq v_o \leq r_a * O \\ & 0 \leq v_p \leq r_a * P \end{aligned} \tag{6}$$

Secondly, we used the estimates from the empirical dis-

tribution $p(\hat{y}_u)$ in the optimization objective function as given by our solution and defined in equation 5. The optimal decision variables $v^* = [v_o^*, v_i^*, v_p^*]$ and the optimal cost z^* are compared between the two problem settings namely, the predict-then-optimize method and our proposed solution. The results in Table 1, show that the cost is reduced by around 10% for our solution compared to the cost obtained from the predict-then-optimize method based solution. This shows that our method can provide a more optimal solution and can effectively minimize costs despite uncertainty. The proposed solution is robust to uncertainty as the optimization objective uses a distribution rather than a point-prediction and is designed to account for the error-propagation from the prediction module.

Table 1. Comparison of cost and decision variable values for traditional Predict-then-Optimize (PnO) v/s Proposed Solution

METRIC	PNO	PROPOSED SOLUTION
COST z^*	186,540	170,195
UTILISATION v_o^*	730	690
CARRYOVER v_i^*	-730	-600
OUTAGE v_p^*	20	60

The dependence of $\hat{y}_u$ on the residual distribution needs to be studied, as a close relationship between the two can cause the residual distribution to be biased.To establish that the $\hat{y}_u$ is not strongly related to the residual distribution $p(e)$ we look at the residual behavior for values of $\hat{y}_u \geq \hat{y}_{u_{median}}$ and $\hat{y}_u < \hat{y}_{u_{median}}$. The residual distributions $p_{g_{med}}(e)$ and $p_{l_{med}}(e)$ shown in figure 4 and figure 4. are generated for the different values of $\hat{y}_u$ which are less than the median and greater than the median respectively. We solve the optimization problem defined in equation 5 for each of these subsets of $\hat{y}_u$ by deriving the corresponding distributions for $\hat{y}_u$ from the residual distributions $p_{g_{med}}(e)$ and $p_{l_{med}}(e)$ respectively.

We examined the optimal cost and the decision variable values obtained while solving the optimization problem for each of the subsets of $\hat{y}_u$ and observed that the results were similar in both the settings. As shown in Table 2, the optimal cost and the decision variables are almost same at $\hat{y}_u \geq \hat{y}_{u_{median}}$ and $\hat{y}_u < \hat{y}_{u_{median}}$. This shows that the residual distribution is not sensitive to $\hat{y}_u$ and that the distribution can be used across different values for $\hat{y}_u$ that might be generated by the prediction module. Thus, we can infer that the proposed solution is highly stable.

One of the key-drawbacks in the predict-then-optimize methods is that the predictions are often insensitive to the impact they create on the optimization problem. This is caused due to the propagation of the prediction error from the prediction module to the optimization module where it is compounded

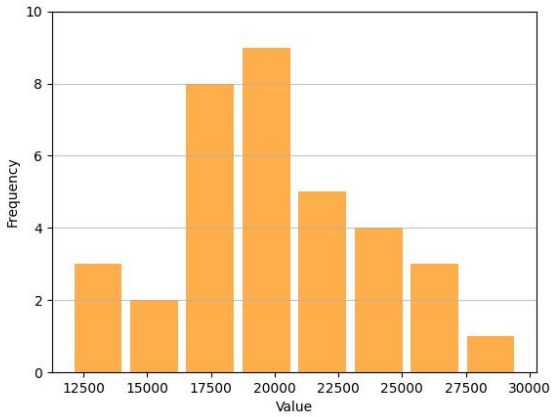

Figure 2. Distribution $p(\hat{y}_u)$.

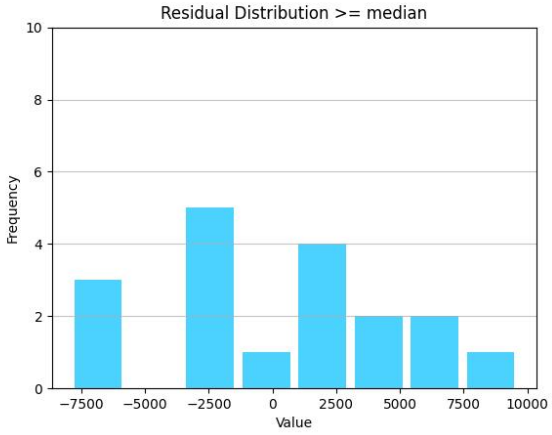

Figure 3. Residual Distribution for $\hat{y}_u \geq \hat{y}_{u_{median}}$.

and results in unfavorable outcomes. Obtaining differentiable approximations of the optimization regret to to incorporate it in back-propagation along with the prediction error is one of the research areas in improving the predict-then-optimize method. However, these methods are tightly coupled to the optimization problem and require extensive mathematical modelling to derive surrogate losses for the optimization regret. Our method offers a simple and elegant solution to circumvent the extensive mathematical effort, by making the optimization objective robust to uncertainties that might arise due to the prediction error.

5. Conclusion

The usual mechanism of first using a complex machine learning model to make predictions and using the predictions as (almost) a god-given input in an optimization problem is essentially the equivalent of finding the function of the expected value of a random variable. The function in our

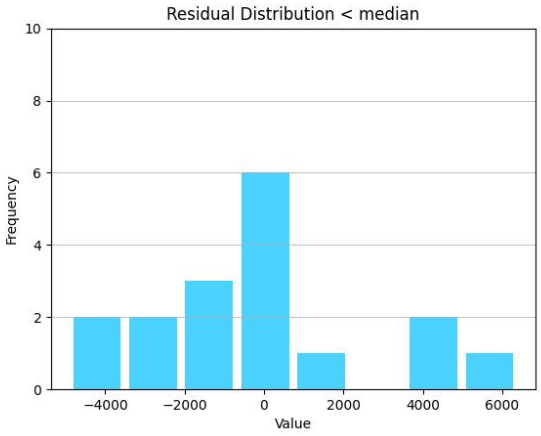

Figure 4. Residual Distribution for $\hat{y}_u < \hat{y}_{u_{median}}$.

Table 2. Comparison of cost and decision variable values for residual distribution for $\hat{y}_u \geq \hat{y}_{u_{median}}$ v/s $\hat{y}_u < \hat{y}_{u_{median}}$

METRIC	$\hat{y}_u \geq \hat{y}_{u_{median}}$	$\hat{y}_u < \hat{y}_{u_{median}}$
COST z^*	203,597	206,892
UTILISATION v_o^*	1,040	1,040
CARRYOVER v_i^*	-980	-980
OUTAGE v_p^*	0	0

case is the entire optimization objective including the minimization subject to the constraints. However, what one truly seeks is the expectation of the function and Jensen's inequality (McShane, 1937) assures us that these are not the same (except in special linear cases). Hence, simply put, the predict then optimize mechanism is bound to result in sub-optimal policies even when the prediction models are extremely good. In this paper, we proposed a novel method to elegantly incorporate uncertainty into the inputs of an optimization problem from the predictions and residuals obtained from any machine learning model. Our method is simple, flexible and quite general in being able to work on different kinds of optimization problems.

References

Amos, B. and Kolter, J. Z. Optnet: Differentiable optimization as a layer in neural networks. In *International Conference on Machine Learning*, pp. 136–145. PMLR, 2017.

Birge, J. R. and Louveaux, F. *Introduction to stochastic programming*. Springer Science & Business Media, 2011.

Boyd, S. P. and Vandenberghe, L. *Convex optimization*. Cambridge university press, 2004.

Elmachtoub, A. N. and Grigas, P. Smart "predict, then optimize". *Management Science*, 68(1):9–26, 2022.

Forrest, J., Ralphs, T., Santos, H. G., Vigerske, S., Forrest, J., Hafer, L., Kristjansson, B., jpfasano, EdwinStraver, Lubin, M., Jan-Willem, rlougee, jpgoncal1, Brito, S., h-i gassmann, Cristina, Saltzman, M., tosttost, Pitrus, B., MATSUSHIMA, F., and to st. coin-or/cbc: Release releases/2.10.10, April 2023. URL https://doi.org/10.5281/zenodo.7843975.

Karush, W. Minima of functions of several variables with inequalities as side conditions. In *Traces and Emergence of Nonlinear Programming*, pp. 217–245. Springer, 2013.

Kong, L., Cui, J., Zhuang, Y., Feng, R., Prakash, B. A., and Zhang, C. End-to-end stochastic optimization with energy-based model, 2022.

Kuhn, H. W. and Tucker, A. W. Nonlinear programming. In *Traces and emergence of nonlinear programming*, pp. 247–258. Springer, 2013.

Lappas, N. H. and Gounaris, C. E. Multi-stage adjustable robust optimization for process scheduling under uncertainty. *AIChE Journal*, 62(5):1646–1667, 2016.

Li, P., Arellano-Garcia, H., and Wozny, G. Chance constrained programming approach to process optimization under uncertainty. *Computers & chemical engineering*, 32(1-2):25–45, 2008.

McShane, E. J. Jensen's inequality. 1937.

Shapiro, A., Dentcheva, D., and Ruszczynski, A. *Lectures on stochastic programming: modeling and theory*. SIAM, 2021.

Tang, B. and Khalil, E. B. Pyepo: A pytorch-based end-to-end predict-then-optimize library for linear and integer programming. *arXiv preprint arXiv:2206.14234*, 2022.

Watson, J.-P., Woodruff, D. L., and Hart, W. E. Pysp: modeling and solving stochastic programs in python. *Mathematical Programming Computation*, 4(2):109–149, 2012.

Wolsey, L. A. and Nemhauser, G. L. *Integer and combinatorial optimization*, volume 55. John Wiley & Sons, 1999.