Robust transmission cost modeling with active learning

Erich Trieschman
Stanford University, Department of Statistics
etriesch@stanford.edu

1. Introduction

Electricity system transmission and generation expansion planning requires accurate forecasts of load, resource availability, and costs. However, current planning methods often rely on a single or few scenarios of load and resource availability forecasts. While deterministic models can be useful for communicating a simple system cost estimate, their results are highly sensitive to model inputs, namely these forecasts.

To address this challenge, the Robust Decision Making (RDM) framework offers a solution to account for stochastic uncertainty. RDM focuses on robustness metrics by developing and stress-testing expansion plans across hundreds of thousands of potential future scenarios. The final output offers a distribution of system costs, which is more useful for describing input uncertainty. However, this approach is more computationally expensive.

In this project, we focus methods to select optimal load, wind, and solar forecasts. Given a distribution of correlated load, wind, and solar scenarios, we use experiment design and active learning techniques to minimize the subset of forecasts required to characterize the distribution of system costs of an expansion decision.

Our primary objective is to evaluate these methods under a single transmission expansion scenario, which we will optimize under the expected value of the forecasts. Time permitting, we would like to evaluate these methods in a general capacity expansion model to understand how expansion results change across nodes of a network from including increased interannual uncertainty.

We intend to use a subset of transmission expansion options slated for Northern California Offshore Wind as a test case for these methods. These expansion options include HVDC Sub-sea cables as well as HVAC 500 kV overhead lines.

2. Methods

The goal of this work is to efficiently generate cost distributions for transmission expansion scenarios using a Security Constrained Unit Commitment and Economic Dispatch model (hereafter ”cost model”). [INSERT: Describe PyPUSA, the transmission plannign model].

To generate our cost distributions we use a sample of model inputs that are generated using a mean-reversion stochastic process method. This method uses a stochastic process based method to generate profiles that represent deviations from an expected base profile. We use the 2032 WECC ADS PCM as the ”base” profile. This work is implemented outside the scope of this project.

To generate our target cost distributions, we first determine the optimal transmission expansion under the expected value of the solar, wind, and load profiles. We then estimate costs under this optimal transmission expansion for each generated profile. We consider two baseline approaches and three optimal approaches to selecting these profiles, which are detailed in Table XX. Given the high-dimensional nature of our data, we also explore several encoding strategies to reduce the dimensionality of the forecasts. This step is necessary for our optimal approaches to selecting forecast profiles.

2.1. Scenario selection

[INSERT: Table summarizing approaches]

To evaluate the performance of our scenario selection methods, we compare them against two baseline approaches. The first baseline approach (Full sample baseline) involves running the cost model under every set of stochastic profiles in our sample. Although this approach provides a gold-standard distribution, it is computationally expensive for many real-world applications.

The second baseline approach (Random sample baseline) involves randomly selecting a subset of scenarios from our sample and running the cost model under those scenarios. We expect the cost distribution generated under this method to have a higher variance; this is the distribution we hope to improve upon with optimal sampling techniques.

To overcome the computational burden of the Full sample baseline, we propose two optimal scenario selection methods.These methods are used to generate surrogate models, which approximate the computationally expensive cost model. Cost distributions can then be generated by the entire sample of forecasts through the inexpensive surrogate models. The first optimal scenario selection approach (One-shot selection) assumes a linear surrogate model and selects model inputs to minimize parameter standard errors. The second optimal scenario selection approach (Bayes optimization selection) is an active learning approach that sequentially selects optimal forecasts based on uncertainty in the surrogate model output. We describe both in more detail below.

2.1.1 One-shot selection

This approach assumes a linear relationship between forecasts and transmission cost outputs. It aims to maximize the covariance matrix of the scenarios, which is an equivalent problem to minimizing the standard error of linear regression parameter estimates.

Under the surrogate cost model, $y_i = x_i^T\beta + \epsilon_i$, with $y_i \in \mathbb{R}$, $x_i, \beta \in \mathbb{R}^n$, and ϵ representing white noise, we can estimate $\hat{\beta}$ using maximum likelihood estimation. If we can optimally choose a fixed set of samples, we can minimize the uncertainty of our estimator $\hat{\beta}$ by solving:

$$\begin{aligned} \underset{m}{\min}.\ (\text{over } S_+) \quad & \left(\sum_{j=1}^{p} m_j v_j v_j^T\right)^{-1} \\ s.t. \quad & m \succeq 0, \quad m^T 1 = M, \quad m \in \{0,1\}^n \end{aligned}$$

Here, p is the number of distinct forecasts, v_j is the jth distinct scenario, m is the vector determining whether a distinct scenario is used, and M is the total number of scenarios to be selected. This approach requires discretizing the dimensions of our input space into scenarios v_j.

We choose $m \in \{0,1\}^n$ instead of the canonical formulation, $m \in \mathbb{Z}^n$, because our cost model is deterministic and we only need to concern ourselves with a single selection for each scenario.

We relax the problem to a convex problem by using L1-norm heuristics and scalarize with the D-optimal design formulation as described in Boyd et. al. [**?**]. Our final formulation becomes

$$\begin{aligned} \underset{\lambda}{\min}. \quad & -\log\det\left(\sum_{j=1}^{p} \lambda_j v_j v_j^T\right) \\ s.t. \quad & 0 \preceq \lambda \preceq 1, \quad \|\lambda\|_1 \leq M \end{aligned}$$

Note this relaxed problem may yield a solution $\lambda_i \in (0,1)$. We propose using heuristics like $\lambda_i > 0.75$ to determine whether a particular scenario is selected to run through the cost model.

Time permitting, we will consider an optimal sample design for Quantile Regression parameters, as described in Wang et al. (2020) [**?**].

2.1.2 Bayes optimization selection

Active learning design is a sequential approach to selecting optimal scenarios for the cost model. In this approach, we iteratively select forecasts to run through the cost model based on the previous forecasts and the cost model outputs.

One popular method for active learning design is Bayesian optimization, which we use here. This approach estimates a probability distribution over cost function outputs, conditional on all of the previous model outputs; it does so through a Gaussian Process (GP) model, which is updated after each evaluation of the function. The selection of the next forecast is determined by an "acquisition function", defined by the objective we seek to optimize [**?**] [**?**].

In our setting, we use Maximum Entropy Search as the acquisition function, which allows us to identify the forecast connected to the point of highest entropy in the surrogate cost model. Entropy is a measure of uncertainty about a given model input, so sequentially selecting maximal entropy points allows us to reduce overall uncertainty across our surrogate model.

We define the Gaussian Process surrogate model as

$$f(x) \sim \mathcal{GP}(m(x), k(x, x'))$$

Where x are the model inputs, and $k(x, x')$ is a pre-defined covariance function, often the squared exponential function, $k(x^{(i)}, x^{(j)}) = \exp\left(-\frac{1}{2}\|x^{(i)} - x^{(j)}\|_2^2\right)$.

A property of the Gaussian Process is that any finite collection of model outputs follows a multivariate Normal distribution. Namely

$$\begin{bmatrix} f(x^{(1)}) \\ \vdots \\ f(x^{(n)}) \end{bmatrix} := \mathbf{f}^{(1:n)} \sim N(0, \mathbf{K})$$

Where $\mathbf{K} \in \mathbb{R}^{n \times n}$ is the covariance matrix of the model inputs with $\mathbf{K}_{ij} = k(x^{(i)}, x^{(j)})$.

With this formulation we may generate the predictive probability of our surrogate cost model, given a model input and the evidence

$$\begin{aligned} f^{(n+1)} \mid \mathbf{x}^{(1:n+1)} \sim \mathcal{N}(\mu_{n+1}, v_{n+1}), \text{ where} & \\ \mu_{n+1} &:= \mathbf{k}(x^{(n+1)})^T \mathbf{K}^{-1} \mathbf{f}^{(1:n)} \\ v_{n+1} &:= k(x^{(n+1)}, x^{(n+1)}) - \mathbf{k}(x^{(n+1)})^T \mathbf{K}^{-1} \mathbf{k}(x^{(n+1)}) \\ \mathbf{k}(x^{(n+1)}) &:= \left[k(x^{(n+1)}, x^{(1)}), \ldots, k(x^{(n+1)}, x^{(n)})\right] \end{aligned}$$

This probability distribution is derived using the Sherman-Morrison-Woodbury formula and properties of conditional multivariate Normal distributions.

For Maximum Entropy Search (MES) as our acquisition function, we first derive an analytical form of the entropy equation, $H(f|\mathbf{x}^{1:n}, x)$:

$$\begin{aligned} H(f \mid \mathbf{x}^{(1:n)}, x) :=& - \int N(f \mid \mu_x, v_x) \log \left[N(f \mid \mu_x, v_x) \right] df \\ =& - \int N(f \mid \mu_x, v_x) \times \\ & \left[-\frac{1}{2} \log(2\pi v_x) - \frac{(f-\mu_x)^2}{2v_x} \right] df \\ =& \frac{1}{2} \log(2\pi v_x) + \frac{1}{2} \end{aligned}$$

Using kernel $k(x^{(i)}, x^{(j)}) = \exp\left(-\frac{1}{2l^2}\|x^{(i)} - x^{(j)}\|_2^2\right)$, the point of maximum entropy in the surrogate model, x^*, is therefore

$$\begin{aligned} & \text{argmax}_x \;\; \frac{1}{2}\log(2\pi v_x) + \frac{1}{2} = \;\text{argmax}_x \;\; v_x \\ =& \;\text{argmax}_x \;\; k(x,x) - \mathbf{k}(x)^T \mathbf{K}^{-1} \mathbf{k}(x) \\ =& \;\text{argmin}_x \;\; \sum_{i=1}^{n} \sum_{j=1}^{n} \mathbf{K}_{ij}^{-1} \mathbf{k}(x)_i \mathbf{k}(x)_j \\ =& \;\text{argmin}_x \;\; \sum_{i=1}^{n} \sum_{j=1}^{n} C_{ij} \exp\left(-\frac{1}{l^2} \|x - \frac{1}{2}(x_i + x_j)\|_2^2 \right) \\ & \text{where } C_{ij} = \mathbf{K}_{ij}^{-1} \exp(-\frac{1}{4l^2}\|x_i - x_j\|_2^2) \end{aligned}$$

We may tune hyperparameter l as described in [**?**].

This problem is clearly not convex, however, we employ several search strategies to approximate x^* with $\hat{x}^*$. First we consider a brute force approach where we calculate entropy for a fixed set of sample points, $x \in \mathcal{D}$. $\hat{x}^*$ is simply the sample point yielding the maximal entropy. We also consider sequential convex programming (SCP) techniques using second order approximations and particle methods; these local methods leverage convex optimization theory.

In SCP we form a convex relaxation to the objective function, $f(x) = \sum_{ij} C_{ij} \exp\left(-\|x - \frac{1}{2}(x_i + x_j)\|_2^2\right)$, at each step of the optimization, iterating until local convergence. The update step in this optimization routine becomes

$$x^{(k+1)} = \text{argmin}_x \hat{f}(x) \;\; \text{s.t} \;\; x \in \mathcal{T}^{(k)}$$

Where $\mathcal{T}^{(k)}$ is a convex trust region for the problem. We consider $\mathcal{T}^{(k)} := \{x \mid |x_i - x_i^{(k)}| \leq \rho_i\}$. **TODO:** Need to understand how to define this trust region.

Using second-order approximation methods, $\hat{f}(x)$ is defined as the convex part of the second order Taylor expansion of our objective function described above, evaluated at point $x^{(k)}$:

$$\hat{f}(x) \approxeq f(x^{(k)}) + \nabla f(x^{(k)})^T (x - x^{(k)}) + (x - x^{(k)})^T P (x - x^{(k)})$$

For

$$\begin{aligned} \nabla f(x) =& \sum_{ij} C_{ij} \exp\left(-\frac{1}{l^2} \|x - \frac{1}{2}(x_i + x_j)\|_2^2 \right) \\ & \times \frac{1}{l^2} \left(-2x + (x_i + x_j) \right) \\ P =& \left(\nabla^2 f(x^{(k)}) \right)_+ \\ \left[\nabla^2 f(x) \right]_{mn} =& \sum_{ij} C_{ij} \exp\left(-\|x - \frac{1}{2}(x_i + x_j)\|_2^2 \right) \\ & \times \frac{1}{l^2} (4 \left[x - \frac{1}{2}(x_i + x_j) \right]_m \left[x - \frac{1}{2}(x_i + x_j) \right]_n \\ & - 2\mathbb{I}\{m = n\}) \end{aligned}$$

Using the particle method, $\hat{f}(x)$ is defined by a convex function fit to a random sample of data, $z_1, \ldots, z_l \in \mathcal{T}^{(k)}$ evaluated with our non-convex objective function, $f(x)$. $\hat{f}(x)$ becomes the solution to

$$\begin{aligned} \min_{P,q,r} \;\; & \sum_{i=1}^{l} \left((z_i - x^{(k)})^T P (x_i - x^{(k)}) + q^T (z_i - x^{(k)}) + r - f(z_i) \right)^2 \\ \text{s.t.} \;\; & P \succeq 0 \end{aligned}$$

We then minimize $\hat{f}(x)$ over all $x \in \mathcal{T}^{(k)}$ to find the next optimal sample point, $x^{(k+1)}$. The particle method approach simplifies computation and allows us to flexibly select where in the trust region, $\mathcal{T}^{(k)}$, to sample.

2.2. Data encoding

In this section, we address the challenge of high-dimensional input space and its impact on optimal scenario selection. Specifically, estimating system costs for just a small subset of nodes over a single year results in a massive number of model inputs (8760 hours in a year × 3 time series × 20 nodes > 500,000 inputs). To overcome the curse of dimensionality, we explore several encoding strategies to drastically reduce the sample space. Our goal is to reduce the number of dimensions to a target of [INSERT: D], while retaining the important features of the original data. To achieve this, we consider three encoding methods: Principal Component Analysis (PCA), Wavelet Transforms, and (time-permitting) a Variational Autoencoder.

2.2.1 Encoding with principal components

PCA is a widely used data reduction technique that identifies linear combinations of input features that capture the most significant variation in the data. This method begins with an eigendecomposition of the covariance matrix of the dataset. The eigenvectors represent the directions along which the data varies the most, while the eigenvalues represent the amount of variation along each of these directions. The eigenvectors with the largest eigenvalues capture the most variation in the data, and are referred to as the principal components once they are normalized. Encoding with PCA involves projecting the original data onto a subset these principal components, which reduces the dimensionality of the dataset while preserving as much of the original variation as possible.

Encoded data, $Y \in \mathbb{R}^{n \times D}$, can be formed as

$$\begin{aligned} X^T X &= Q \Lambda Q^{-1}, \text{ the covariance eigendecomposition} \\ U &\in \mathbb{R}^{n \times n}, \text{ s.t. } u_i = \frac{q_i}{\|q_i\|} \text{ for } Q = [q_1, \ldots, q_n] \\ Y &= X U_D, \text{ for } U_D = [u_1, \ldots, u_D] \end{aligned}$$

2.2.2 Encoding with a Variational Autoencoder

A Variational Autoencoder (VAE) is a type of neural network that can learn a compressed representation of high-dimensional data. The VAE is trained to encode the input data into a lower-dimensional latent space, while still preserving the important features of the original data. This technique allows us to reduce the dimensionality of the input data while capturing the most important information [**?**]

3. Results and discussion

We are picturing the primary output of this project to be a comparison table like the one that follows:

Table 1. Estimated transmission expansion costs, by method

	One-shot		**Active learning**		
	Baseline	D-optimal	Baseline	Taylor	Particle
Min	0.73±0.37	1.49	0.64 ± 0.53	0.46 ± 0.41	0.41 ± 0.39
5th pctl	0.62±0.23	0.85	0.68 ± 0.23	0.24 ± 0.16	0.23 ± 0.16
10th pctl	0.56±0.20	0.53	0.47 ± 0.19	0.14 ± 0.12	0.17 ± 0.14
25th pctl	0.42±0.17	0.26	0.28 ± 0.14	0.11 ± 0.10	0.14 ± 0.11
50th pctl	0.16±0.12	0.11	0.11 ± 0.09	0.09 ± 0.07	0.14 ± 0.10
75th pctl	0.30±0.19	0.34	0.26 ± 0.13	0.11 ± 0.09	0.12 ± 0.09
90th pctl	0.77±0.27	0.66	0.45 ± 0.17	0.14 ± 0.13	0.16 ± 0.12
95th pctl	0.89±0.30	0.75	0.47 ± 0.20	0.18 ± 0.17	0.18 ± 0.15
Max	2.41±0.69	1.35	0.76 ± 0.46	0.40 ± 0.45	0.53 ± 0.37
Mean	0.14±0.10	0.04	0.11 ± 0.09	0.09 ± 0.07	0.11 ± 0.09
Std	0.50±0.13	0.50	0.37 ± 0.09	0.10 ± 0.07	0.10 ± 0.07

4. Conclusion and future work

5. Contributions and acknowledgements

This work is conducted by Erich Trieschman as part of a larger project in collaboration with Kamran Tehranchi. Kamran is responsible for running the transmission planning model and for generating the stochastic profile data for our models.

All of my code, results, and the pretrained models are available in a project repo on my GitHub, available at https://github.com/etrieschman/grid-planner

References

[1] S. Boyd and L. Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[2] E. Brochu, V. M. Cora, and N. de Freitas. A tutorial on bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning, 2010.

[3] S. Odaibo. Tutorial: Deriving the standard variational autoencoder (vae) loss function, 2019.

[4] C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, 11 2005.

[5] H. Wang and Y. Ma. Optimal subsampling for quantile regression in big data, 2020.

[6] J. Wang. An intuitive tutorial to gaussian processes regression, 2022.