MTL-Designer: An Integrated Flow for Analysis and Synthesis of Microstrip Transmission Line

Qipan Wang[1,2], Ping Liu[4], Liguo Jiang[4], Mingjie Liu[5], Yibo Lin[1,3*], Runsheng Wang[1,3], Ru Huang[1,3]
[1]School of Integrated Circuits & [2]Academy for Advanced Interdisciplinary Studies, Peking University
[3]Institute of Electronic Design Automation, Peking University, Wuxi, China
[4]Xpeedic Technology, Inc. [5]NVIDIA Corp.
{qpwang, yibolin, r.wang, ruhuang}@pku.edu.cn

ABSTRACT

Microstrip transmission line (MTL) appears extensively in microwave integrated circuits (MIC). To sufficiently analyze and synthesize the MTL, we propose MTL-Designer that can design the electrical and geometrical parameters of an MTL given performance specifications. We construct a deep generative model to generate initial solutions, and a surrogate model to predict the characteristics, optimize the solutions, and select from them. We further propose an adaptive sampling algorithm to speedup training. Our flow can generate 1000 feasible solutions within ~0.6 s, realizing > 99.8% accuracy given various design specifications for two common MTL systems, exhibiting its strong potential for MIC design.

1 INTRODUCTION

Microstrip is a popular form of transmission line that conveys microwave-frequency signals. It is extensively used in modern MICs and high-speed digital PCB designs. Fig. 1 sketches the cross-section view of a typical single-ended MTL composed of a thin flat conductor on a dielectric insulating layer, parallel to a ground plane. Given different configurations on the parameters like length, width, and thickness of the conductor, MTLs can achieve different characteristics and work as microwave passive components, lumped microstrip elements, and transmission medium in package [1].

Previous studies on MTL can be roughly categorized into analysis and synthesis, as shown in Fig. 1. The goal of the analysis is to simulate the characteristic impedance, propagation constant, and other important characteristics, given electrical and geometrical parameters or configurations. Rigorous simulators rely on numerical methods [2] to compute accurate solutions even for complicated geometries, but are very time-consuming. To speedup analysis, analytical approximations [3] are proposed for certain structures of MTLs. These methods require careful analysis and are often limited to certain design structures. To balance the accuracy, efficiency, and generality of methodologies, data-driven approaches like machine learning (ML) [4–6] have been proposed, which can be calibrated by rigorous simulations.

Synthesis can be viewed as just the inverse of analysis, aiming at determining the electrical and geometrical parameters given target characteristics (a.k.a design specifications). A common practice is to iteratively optimize design parameters based on the closed-form analytical expressions of MTL characteristics or surrogate models [6–8]. Later, end-to-end deep learning (DL) algorithms [9–11] that directly predict parameters given the design specifications are proposed, avoiding the iterative optimization in the previous approaches, but at the cost of accuracy.

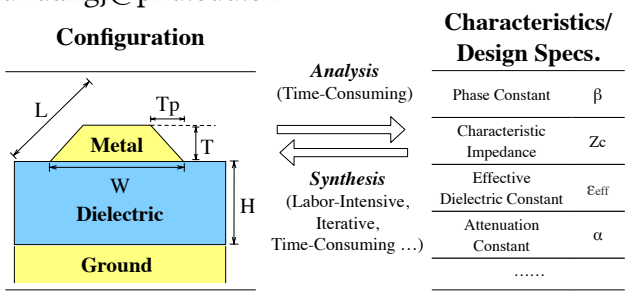


Figure 1: Illustration of the configuration of a single-ended MTL, and the analysis and synthesis process. In the left figure, L represents the MTL length, W and T the width and thickness of the conductor, Tp the trap width, and H the height of the dielectric.

Despite previous efforts, the aforementioned synthesis approaches have several shortcomings. Firstly, real-world designers have very high requirements for the quality of synthesized solutions and pursue extremely close matches to design specifications (< 0.5% relative error). Secondly, designers often want to specify constraints based on prior knowledge, such as limiting the ranges of certain parameters and biasing certain characteristics based on empirical functions. Such knowledge is practical but overlooked before. Additionally, most previous studies only focus on simple structures, like single-ended MTL, but have not been validated under complicated scenarios like waveguide composing of coupled MTLs.

Confronted with these challenges, we propose MTL-Designer (MTLD), an integrated MTL analysis and synthesis flow. The major contributions of this paper are highlighted as follows:

- We propose an integrated MTL analysis and synthesis flow for designing MTL parameters given design specifications and constraints from prior knowledge.
- We customize a surrogate model for accurate and efficient MTL analysis, with an adaptive sampling strategy to speedup training.
- We propose an optimization and selection algorithm with a generative learning model to generate multiple feasible solutions for various design specifications.
- We validate the flow with commercial simulator *TmlExpert*[12]. After training, MTL-Designer can provide 1000 feasible solutions within 0.6 s, with an accuracy > 99.8% for designing both single-ended MTL and waveguide of coupled-MTLs, satisfying various design constraints.

The rest of this paper is organized as follows. Section 2 reviews the basic concepts and formulates the problem discussed. Section 3 provides a through explanation of the proposed flow. Section 4 demonstrates the power of our flow with comprehensive results, followed by the conclusion in Section 5.

*Corresponding author

2 PRELIMINARIES

In this section, we will first introduce some basic knowledge of microstrip line analysis and synthesis in 2.1, and deep generative models in 2.2. The problem discussed in this paper will be formulated in 2.3.

2.1 Basics of MTL

Fig. 1 provides a cross-section view of a single-ended MTL and draws relevant geometrical parameters. The electrical parameters not shown include the dielectric constant ϵ, loss tangent, and the conductor conductivity σ. The trapezoidal cross-section considers the effects of etching in manufacturing [13], which goes beyond the capability of many rectangular-section-based analytical approximations.

Given the configuration of an MTL, characterizing its electrical behavior such as characteristic impedance Z_c and propagation constant requires a full-wave analysis based on discretization and numerical methods [14], which can be computationally intensive. As a result, designing an MTL takes tens of minutes to hours even for experienced designers due to iterative and expensive simulations and manual parameter adjustment. To avoid expensive simulations, a typical synthesis procedure adopts surrogate models to approximate the characteristics over the parameter space [7].

2.2 Conditional Variational Auto-Encoder

MTL synthesis aims at obtaining the configuration parameters given design specifications. It can be viewed as a conditional generation problem, i.e., to generate new data from a similar distribution of the training data given conditions [15].

In this work, we adopt the Conditional Variational Auto-Encoder [16] (CVAE) as the generative model for its stability and interpretability. Vanilla VAE assumes the generated data $\{x_i\}_{i=1}^N$ is governed by some underlying representation z, also known as the latent variable, which typically follows an easy-to-sample prior distribution $p(z)$. The goal is to find a parameterized distribution $p_\theta(x) = \int p_\theta(x,z)dz$, that approximates the genuine data distribution $p(x)$ as close as possible. Conditional VAE augments this process with data incorporating labels, which can be design specifications in MTL synthesis. More details will be given in 3.2.

2.3 Problem Formulation

In this work, we represent the parameters and characteristics with $\boldsymbol{P} \in \mathcal{D}$ and $\boldsymbol{Z} = (f, Z_c)$ respectively, where $\mathcal{D} = [0,1]^d$ is the normalized design space of dimension d. We choose the characteristic impedance at a given frequency as the design specification (target): $\boldsymbol{Z_t} = (f, Z_{c,\,t})$, where t denotes the target. Furthermore, the flow can be generalized to other characteristics easily.

Analysis Accuracy: To evaluate the surrogate analysis model, for test dataset $\{(\boldsymbol{P}_{test}(i), Z^{(i)}_{c,\,test})\}_{i=1}^{N_{test}}$, accuracy of the predicted characteristic impedance $\{Z^{(i)}_{c,\,pred}\}_{i=1}^{N_{test}}$ can be evaluated by the mean absolute percentage error:

$$\mathbf{Acc}_{ana} = 1 - \mathbf{Mean}(\|Z^{(i)}_{c,\,pred}/Z^{(i)}_{c,\,test} - 1\|),$$

where operation $\mathbf{Mean}(\cdot)$ calculates the mean value among test data. With this metric we can define the analysis problem:

Problem I (Analysis): *Given a rigorous MTL simulator, the objective is to put up an analysis model that can predict the characteristic impedance as accurately as possible, that is, to maximize the analysis accuracy.*

Synthesis Accuracy: Given a set of design specifications $\{\boldsymbol{Z}^{(i)}_{\boldsymbol{t}}\}_{i=1}^N$, the accuracy of corresponding synthesis parameters $\{\boldsymbol{P}^{(i)}_g\}$ can also be evaluated by the mean absolute percentage error between the actual characteristics and the design specifications:

$$\mathrm{Acc}_{syn} = 1 - \mathbf{Mean}(\|Z^{(i)}_{c,\,g}/Z^{(i)}_{c,\,t} - 1\|),$$

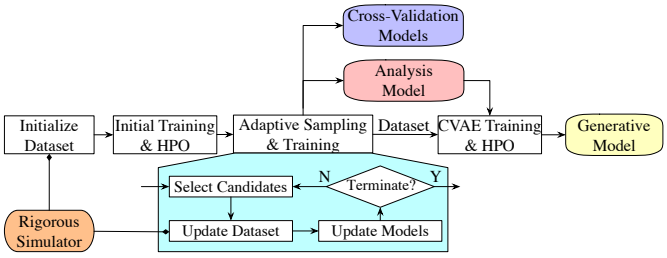
where $Z^{(i)}_{c,\,g}$ is the actual impedance corresponding to $\boldsymbol{P}^{(i)}_g$.

Efficiency: In our flow, a group of feasible solutions will be generated and evaluated by their degrees of confidence. However, more solutions in a group require longer runtime. So we define the synthesis efficiency as: $\frac{\text{Runtime}}{\#\text{ of Solutions}}$, revealing the trade-off between the runtime and the number of solutions generated in synthesis.

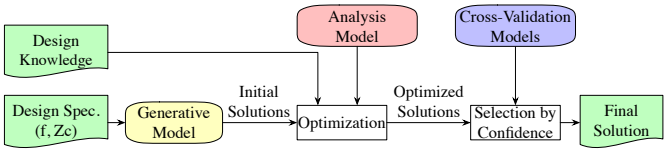
Design-Knowledge-based Constraints: Taking prior knowledge from designers is very useful in practice. In this work, we consider two prevalent types of knowledge and write them as constraints: linear hard constraints and soft constraints. To be concrete, the former contains linear equations like $\langle \boldsymbol{A}, \boldsymbol{P}\rangle = b$, and linear inequalities in the form $\langle \boldsymbol{A}, \boldsymbol{P}\rangle \le b$, where $\boldsymbol{A}$ and b are constructed by experience. For example, the condition $W > 2\,Tp$ in real manufacture can be transformed into $\langle(-1, 2, \overrightarrow{0}), (W, Tp, ...)\rangle < 0$. The latter type considers the soft constraints in the form $\mathbf{min}\{f(\boldsymbol{P})\}$ to indicate additional design intentions. For example, to reduce the attenuation of an MTL, which is approximately proportional to the power of conductor width W, we construct an experience function $f = W^\beta$ ($\beta \in \mathbb{R}$) and try to minimize it during synthesis.

Based on the above considerations, we define the synthesis problem studied in this work as:

Problem II (Synthesis): *Given the design specifications and knowledge-based constraints, the objective is to generate feasible solutions, with the synthesis accuracy and efficiency as better as possible.*



(a) Sampling and training part.



(b) Synthesis part.

Figure 2: Illustration of the proposed flow, composed of two parts: sampling and training part, and synthesis part.

3 WHOLE FRAMEWORK

In this section, we will illuminate the proposed flow, which composes of two parts, as shown in Fig. 2. In the sampling and training part, after determining the design space, initial training set

$\{(P_{ini}, Z_{ini})\}_{i=1}^{N_{ini}}$ and test dataset $\{(P_{test}, Z_{test})\}_{i=1}^{N_{test}}$ are collected using Latin hypercube sampling (LHS) and the rigorous simulator. Then the initial training and hyperparameter optimization (HPO) of the analysis model (refer to Section 3.1) is conducted. Following adaptive sampling (described in Section 3.3) yields the analysis model **SuM**, cross-validation models **CvM** (sharing the same structure with **SuM**), and the full training dataset. Then the eventual analysis model and dataset will be utilized to train the CVAE model (details in Section 3.2) and determine its best combination of hyperparameters, finishing the sampling and training part.

In the design part, the CVAE decoder as the generative model will first generate initial solutions given the design specifications (targets). Then the analysis model is employed to optimize the solutions given the knowledge-based constraints, as stated in Section 3.4. Ultimately, the final solution will be selected by their degree of confidence (Section 3.5), evaluated by the cross-validate models.

3.1 Surrogate Analysis Model

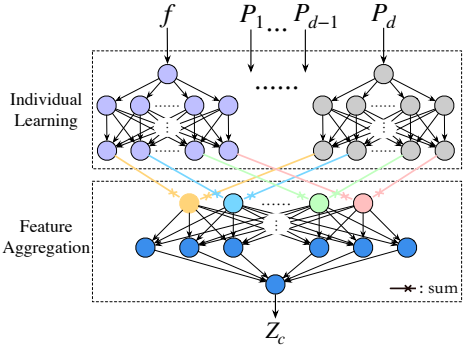


Figure 3: Illustration of the surrogate model, consisting of individual processing part and feature aggregate part.

To predict the characteristic impedance Z_c given the frequency f and system parameters $\boldsymbol{P}$, we customize a surrogate analysis model. It is based on the observation that the impedance changes smoothly with each parameter, following diverse variation rules. For example, Z_c decays non-linearly with f, while increasing quasi-linearly with the trap width Tp. Hence, a reasonable analysis model should treat input components differently, instead of in the same way.

Motivated by this, our analysis model composes of two parts, as shown in Fig. 3. Firstly, the individual learning part handles each input component with an independent multi-layer perception (MLP) network, each exporting a feature vector of the same dimension. The feature vectors are aggregated by summation (rather than concatenation for smaller computational burden), and processed with another MLP to predict the characteristic impedance in the feature aggregation part. Network hyperparameters, including the depth and width of each MLP, are determined by HPO. The sizes of MLPs in the individual part are adjusted according to the importance of corresponding input components.

The predicted impedance will have an error compared with the rigorous value. To evaluate this error, we refer to the cross-validation models $\mathbf{CvM}_k$ in Section 3.3. Specifically, for input $(f, \boldsymbol{P})$, its actual impedance Zc will approximately follow the normal distribution:

$$Zc \sim \mathcal{N}(\mathbf{SuM}(f, \boldsymbol{P}), \sigma_{\boldsymbol{P}}^2), \tag{1}$$

where $\sigma_{\boldsymbol{P}} = \mathbf{Std}(\mathbf{CvM}_k(f, \boldsymbol{P}))$, and $\mathbf{Std}(\cdot)$ calculates the standard deviation of predictions of cross-validation models.

3.2 CVAE Model

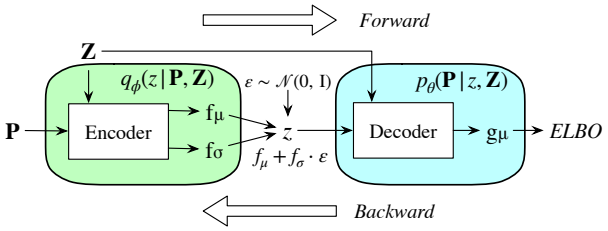


Figure 4: Illustration of the CVAE model during training, composed of an encoder and a decoder, working as $q_\phi(z|\boldsymbol{P}, Z)$ and $p_\theta(\boldsymbol{P}|z, Z)$ respectively. Backward flow is based on the gradient of ELBO.

The goal of introducing the CVAE model is to learn the latent representation of the training dataset $\{(\boldsymbol{P}_{train}, Z_{train})\}_{i=1}^{N_{train}}$, and generate feasible design parameters $\boldsymbol{P}$ given unseen design specifications $Z = (f, Z_c)$. CVAE seeks to find the maximum likelihood estimation in the training dataset: $\theta_{MLE} = \underset{\theta}{argmax}\, \Pi_{i=1}^{N_{train}} p_\theta(\boldsymbol{P}^{(i)}|Z^{(i)})$. We consider the logarithm of the conditional likelihood of each training data and transform it as:

$$\log p_\theta(\boldsymbol{P}|Z) = \log \int_z p_\theta(\boldsymbol{P}, z|Z)dz \geq \mathbb{E}_{q_\phi(z|\boldsymbol{P},Z)}\left(\log \frac{p_\theta(\boldsymbol{P}, z|Z)}{q_\phi(z|\boldsymbol{P}, Z)}\right).$$

Here we introduce $q_\phi(z|\boldsymbol{P}, Z)$ as a parameterized approximation to the intractable posterior distribution $p(z|\boldsymbol{P}, Z)$; and get the lower bound of the likelihood, referred to as evidence lower bound (ELBO). The joint distribution $p_\theta(\boldsymbol{P}, z|Z)$ equals to: $p(z|Z)p_\theta(\boldsymbol{P}|z, Z)$; and we further assume the prior distribution $p(z|Z)$ to be the standard normal distribution, statistically independent of Z. To conclude, now the optimization goal reads:

$$ELBO = \underbrace{\mathbb{E}_{q_\phi(z|\boldsymbol{P},Z)}\left(\log p_\theta(\boldsymbol{P}|z, Z)\right)}_{-\text{ reconstruction loss}} \underbrace{-\mathrm{KL}(q_\phi(z|\boldsymbol{P}, Z)\|p(z))}_{\text{KL of } q(z|\boldsymbol{P},Z) \text{ and prior}}, \tag{2}$$

where $KL(p\|q) = \int p \ln(p/q)dq$ is the KL divergence.

We realize the "recognition" distribution $q_\phi(z|\boldsymbol{P}, Z)$ and the generation distribution $p_\theta(\boldsymbol{P}|z, Z)$ with two Gaussian distributions:

$$q_\phi(z|\boldsymbol{P}, Z) \sim \mathcal{N}(f_\mu(\boldsymbol{P}, Z), f_\sigma^2(\boldsymbol{P}, Z)\,\mathbb{I})$$
$$p_\theta(\boldsymbol{P}|z, Z) \sim \mathcal{N}(g_\mu(z, Z), \sigma_d^2\,\mathbb{I}),$$

where (f_μ, f_σ), g_μ are parameterized by two MLPs respectively as shown in Fig. 4, and σ_d is the variance modulating the reconstruction ability. Now the KL term in (2) can be evaluated analytically.

To handle the negative reconstruction error term, whose gradient information is hard to get, we employ the reparametrization trick [17] that transforms $z = f_\mu + f_\sigma \cdot \varepsilon$, where $\varepsilon \sim \mathcal{N}(0, \mathbb{I})$. Then we can make this term differentiable with a Monte Carlo approximation:

$$\mathbb{E}_{q_\phi}\left(\log p_\theta(\boldsymbol{P}|z, Z)\right) \approx \frac{1}{M}\Sigma_{m=1}^{M} \log p_\theta(\boldsymbol{P}|z^{(m)}, Z)$$
$$= \frac{-1}{M}\Sigma_{m=1}^{M}\left(\log(\sqrt{2\pi}\sigma_d) + \frac{\|g_\mu\left(f_\mu(\boldsymbol{P}, Z) + f_\sigma(\boldsymbol{P}, Z)\varepsilon^{(m)}, Z\right) - \boldsymbol{P}\|^2}{2\sigma_d^2}\right),$$

where M is set to be 1 in experiments. During training, $\boldsymbol{P}, Z$ is selected in a mini-batch manner, and the depth and width of MLPs are also determined by HPO to minimize ELBO (2).

After training of the CVAE model, the CVAE decoder can now generate initial solutions $\{\boldsymbol{P}_{ini}^{(i)}\}_{i=1}^{N_{ini}}$ given the target $\boldsymbol{Z_t}$: $\boldsymbol{P}_{ini}^{(i)} = g_\mu(z^{(i)}, \boldsymbol{Z_t})$, here $z^{(i)} \sim p(z) \sim \mathcal{N}(0, \mathbb{I})$. Multiple solutions can be generated, as much as the variable z sampled.

3.3 Adaptive Sampling

Constructing the aforementioned models with high accuracy requires a huge amount of labeled training data while obtaining labels is expensive due to a long time for rigorous simulations. The goal of adaptive sampling is to actively choose a small number of representative samples that can still achieve high analysis accuracy. Our adaptive sampling algorithm iteratively selects new critical samples and updates the dataset and models. This iterative process mitigates the risk of over- or under-sampling.

In [18], a mixed adaptive sampling strategy is developed featuring a combination of space-filling and adaptive searching. We modify this method to satisfy the goal of training both the analysis model and the generative model. We first generate the candidate parameters $\{\boldsymbol{P}_{cand}^{(i)}\}_{i=1}^{N_{cand}}$ during each iteration step with LHS. Then these candidate parameters are scored according to three criteria: parameter uniformity, value variance, and value uniformity.

Parameter uniformity is evaluated by a space-filling criterion:

$$PU_i = \min_{\boldsymbol{P}_j \in \{\boldsymbol{P}_{train}\}} \sqrt{\|\boldsymbol{P}_i - \boldsymbol{P}_j\|_2}, \tag{3}$$

here $\{\boldsymbol{P}_{train}\}$ is the parameters in existing training dataset, and index i traverses all candidate parameters.

Value variance is an estimate of the variance of the prediction of the analysis model, determined in a K-fold cross-validation manner. Firstly, K cross-validation datasets are generated out of the existing training dataset, each discarding N_{train}/K data exclusively. Next, K cross-validation models $\mathbf{CvM}_k$ ($k = 1 \sim K$) are trained on these datasets correspondingly, sharing the same structure with the analysis model. Value variance can then be derived as:

$$VV_i = \mathbf{Std}(\mathbf{CvM}_k(\boldsymbol{P}_i))/\mathbf{Mean}(\mathbf{CvM}_k(\boldsymbol{P}_i)). \tag{4}$$

$\boldsymbol{P}_{cand}$ with large value variance will be chosen since the uncertainty of the corresponding prediction of the analysis model is high. We set $K = 7$ during experiments, as the variance has basically converged in this case, and will not change greatly with the increase of K.

Value uniformity is originally proposed to enhance the diversity of impedance, since we hope that the impedance in the training dataset for the generative model can cover a large range of values. This metric measures the uniformity of impedance similar to the parameter uniformity:

$$VU_i = \min_{\boldsymbol{Z}^{(j)} \in \{\boldsymbol{Z}_{train}\}} \sqrt{\|Z_c^{(j)}/\mathbf{SuM}(\boldsymbol{P}_i) - 1\|_2}. \tag{5}$$

To conclude, during each iteration step, a weighted score will be attributed to each candidate $\boldsymbol{P}_{cand}^{(i)}$:

$$S_i = w_{PU} \cdot \mathbf{Norm}(PU_i) + w_{VV} \cdot \mathbf{Norm}(VV_i) + w_{VU} \cdot \mathbf{Norm}(VU_i), \tag{6}$$

where $\mathbf{Norm}(X_i) = \frac{X_i}{\mathbf{Std}(\{X\}) + \mathbf{Mean}(\{X\})}$ normalizes each X_i in the set $\{X\}$. We run rigorous simulations on parameters with scores larger than threshold η controlling the number of new data, and append them to the existing training dataset. During experiments we set $w_{PU} = w_{VV} = w_{VU} = 1$, and threshold η empirically.

3.4 Design Optimization

Design optimization is proposed to reduce the deviation between the target impedances and those of solutions, and enforce the solutions to satisfy design constraints. This process is carried out in a gradient-based framework.

3.4.1 Unconstrained Optimization. Firstly, each solution will be optimized to reduce the deviation between its actual impedance and the target by unconstrained optimization. The deviation is measured by the loss between the predicted impedance $\mathbf{SuM}(f, \boldsymbol{P})$ (as an approximation to the actual value) and the target $Z_{c,\,t}$:

$$\mathcal{L} = \|\mathbf{SuM}(f, \boldsymbol{P}) - Z_{c,\,t}\|. \tag{7}$$

Since the surrogate model is differentiable, we can derive the gradient $\frac{\partial \mathcal{L}}{\partial \boldsymbol{P}}$, and update the design parameters by gradient descent:

$$\boldsymbol{P}_{new} = \boldsymbol{P} - \Delta \boldsymbol{P} \times \frac{\partial \mathcal{L}}{\partial \boldsymbol{P}}. \tag{8}$$

Hyperparameter $\Delta \boldsymbol{P}$ is set to be small ($1e-4 \sim 3e-4$ empirically) to avoid overdoing. We conduct the gradient descent iteratively, and typically it only takes several steps (we set it to be 10) for the solutions to converge in experiments.

3.4.2 Constrained Optimization. Then we consider the design-knowledge-based constraints. We categorize them into three classes: linear equations, linear inequalities, and soft constraints. The first two are hard constraints that must be satisfied, and the linear equation $\langle \boldsymbol{A}, \boldsymbol{P} \rangle = b$ can be handled by translating into two linear inequalities: $\langle \boldsymbol{A}, \boldsymbol{P} \rangle \le b$ & $-\langle \boldsymbol{A}, \boldsymbol{P} \rangle \le -b$.

As to the linear inequalities, we utilize the gradient projection method [19] to update the solutions. The update rule resembles (8) apart from a projection operator $\mathbb{P}$: $\boldsymbol{P}_{new} = \mathbb{P}(\boldsymbol{P} - \Delta \boldsymbol{P} \times \frac{\partial \mathcal{L}}{\partial \boldsymbol{P}})$. Here $\mathbb{P}$ takes a simple form to project the parameters to the semi-plane $C = \{\boldsymbol{P} | \langle \boldsymbol{A}, \boldsymbol{P} \rangle \le b\}$:

$$\mathbb{P}_C(\boldsymbol{P}) = \begin{cases} \boldsymbol{P} + \frac{b - \langle \boldsymbol{A}, \boldsymbol{P} \rangle}{\|\boldsymbol{A}\|^2} \boldsymbol{A} & \langle \boldsymbol{A}, \boldsymbol{P} \rangle > b \\ \boldsymbol{P} & \langle \boldsymbol{A}, \boldsymbol{P} \rangle \le b \end{cases}. \tag{9}$$

For a set of linear inequalities $\{\langle \boldsymbol{A^{(i)}}, \boldsymbol{P} \rangle \le b^{(i)}\}_{i=1}^{(N_c)}$, their corresponding projection operator $\mathbb{P}^{(i)}$ can be composed cumulatively: $\mathbb{P}_{tot}(\cdot) = \mathbb{P}^{(1)}(...\mathbb{P}^{(N_c)}(\cdot))$.

The soft constraints $\mathbf{min}\{f(\boldsymbol{P})\}$ are to be satisfied as strictly as possible, but not mandatory. So we propose to add a regularization term $\mathcal{R} = \|f(\boldsymbol{P})\|$ that seeks to minimize the objective function to (7): $\mathcal{L}_{tot} = \mathcal{L} + \lambda \mathcal{R}$, where λ is a weight modulating the degree that the constraints are concerned. Since the gradient descent algorithm seeks to reduce the total loss $\mathcal{L}_{tot}$, we can expect the regularization term also gets smaller during iteration.

3.5 Solution Selection

After generation and optimization, we have gathered multiple feasible solutions. The goal of this selection step is to evaluate and select from these solutions since typically we just need one. To this end, we put forward a metric *DoC* (degree of confidence) based on the discrepancy between the predicted and required impedance:

$$DoC = KL\left(\mathcal{U}(Z_{c,t}, \sigma_t) \| \mathcal{N}(\mathbf{SuM}(f, \boldsymbol{P}), \sigma_{\boldsymbol{P}}^2)\right), \tag{10}$$

where $\mathcal{U}(Z_{c,t}, \sigma_t)$ is a uniform distribution around target $Z_{c,t}$ concerning the acceptable impedance error σ_t (set to be $0.2\% Z_{c,t}$ in experiments), and the other distribution is (1). *DoC* originates from the idea that if the predicted impedance distribution of a solution

is close to the target range, then its actual impedance is also highly probable to fall into this range.

To conclude, our MTL-Designer can run in two modes during synthesis: MTLD-Fast and MTLD-Optim. (optimization). The former selects the most feasible solution (featuring the best *DoC*) from the initial solutions, which is ultra-fast; the latter optimizes the solutions to reduce the impedance deviation and meet certain constraints, at the cost of some efficiency.

4 EXPERIMENTAL RESULTS

In this section, we validate our proposed flow with experiments.

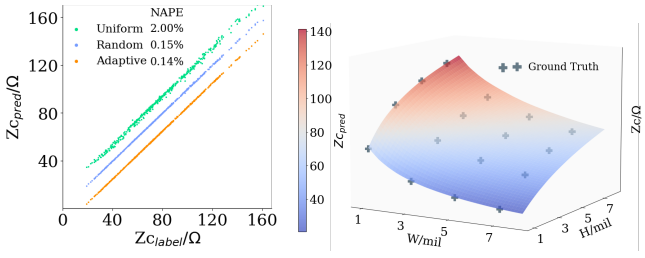
4.1 Experimental Setup

We choose two widely-used MTL systems during experiments: a single-ended MTL with 5 parameters and a differential coplanar waveguide with a finite ground plane, composed of four coupled MTLs with 10 parameters. Note that for the waveguide we choose the differential impedance as the design specification. Commercial tool Xpeedic *TmlExpert* is employed for rigorous simulation.

The sampling and training flow takes about 3 hours for single-ended MTL and 12 hours for the other, with $\sim$ 40% and $\sim$ 70% portions of time consumed by rigorous simulation. HPO and Bayes Optimization (BO) are realized by Optuna [20] and neural network models are constructed with Pytorch. Each model takes 1 $\sim$ 3 minutes to train on an A100 GPU. A total of 1000 solutions will be generated and optimized during synthesis.

4.2 Comparison of Sampling Strategies

The adaptive sampling process collects 605 data for the single-ended MTL and 1015 data for the waveguide. We demonstrate the power of adaptive sampling in the single-ended MTL system in Fig. 5(a). We compare different sampling strategies, including uniform sampling (uniform in each design parameter), pure LHS sampling, and our adaptive sampling. Uniform sampling is worse than the other two methods, for its lower accuracy and larger data size. While LHS sampling achieves high accuracy at the cost of efficiency, and tends to sample too many points at once, since we cannot predetermine a reasonable data size. In contrast, adaptive sampling exhibits its superiority in speeding up training, since it can gradually increase the data size according to the analysis accuracy achieved.



(a) Effects of Sampling Strategies. **(b) Relation between Z_c and (W,H).**

Figure 5: Power of adaptive sampling and surrogate analysis model for the single-ended MTL. (a). Effects of different sampling strategies, showing the relation between data size and the accuracy of corresponding analysis models. The vertical axis has been transformed for clarity. (b). Relation of the predicted impedance and parameters (W, H). The crosses represent the ground truth.

Table 1: Comparison between the accuracy and runtime of different analysis models for two MTL systems.

Algorithm	Single-ended MTL			Waveguide of Coupled-MTLs		
	Ana. Accuracy/%	Runtime/s CPU	GPU	Ana. Accuracy/%	Runtime/s CPU	GPU
Rigorous	100	2.078	——	100	28.538	——
MLP [9]	99.46	3E-4	0.003	99.18	2E-4	3E-4
RBF [5]	98.61	4E-4	4E-4	98.63	4E-4	5E-4
SVR [6]	98.90	2E-4	——	97.55	2E-4	——
MTLD	**99.86**	0.003	0.005	**99.61**	0.002	0.006

4.3 Power of the Analysis Model

Then we compare different analysis models (MTLD, MLP [9], radial basis function (RBF) [5]), and support vector regression (SVR) [6]) for two MTL systems. We randomly sample 200 sets of configuration parameters as test cases for each system and summarize the analysis results in Table 1. Our analysis model can predict the characteristics with an accuracy of 99.86% and 99.61% for the two systems respectively, which is more accurate than others. The runtime of all models is much smaller than that of rigorous simulation, while that on a GPU may be a bit slower than on a CPU for transferring data between CPU and GPU. Fig. 5(b) gives an example of the relation between the characteristic impedance with the metal width W and dielectric height H. Corresponding ground truth points are attached. We can observe a rational fitting between the predicted and actual values, while the impedance varies smoothly and nonlinearly in the whole domain, as stated in Section 3.1.

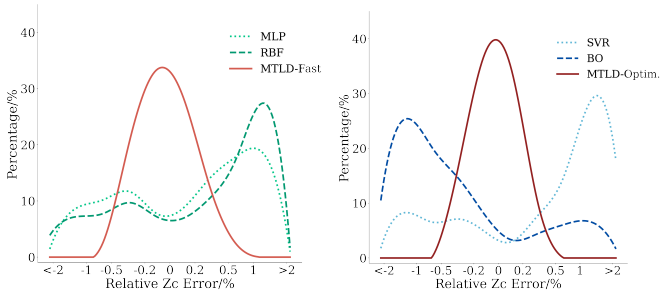
Table 2: Performance comparison between synthesis algorithms for two MTL systems.

Algorithm	Single-ended MTL			Waveguide of Coupled-MTLs		
	Syn. Accuracy/%	Efficiency/(s/#) CPU	GPU	Syn. Accuracy/%	Efficiency/(s/#) CPU	GPU
MLP [9]	99.18	4E-4	0.001	98.20	0.105	0.084
RBF [5]	98.88	2E-4	3E-4	98.27	0.060	0.070
MTLD-Fast	99.82	**3E-6**	**2E-6**	99.83	**1E-5**	**2E-6**
SVR* [6]	98.30	0.223	——	99.38	0.416	——
BO [8]	98.79	5.09	——	98.60	16.238	——
MTLD-Optim.	**99.88**	3E-4	1E-4	**99.88**	6E-4	3E-4

* SVR can only determine one parameter (we choose W), with the others fixed.

4.4 Power of the Synthesis Model

Above all, we test the performance of our synthesis model against other algorithms for unconstrained designs, while the effects of design-knowledge-based constraints will be clarified in the next Section. The synthesis algorithms include end-to-end approaches (MLP [9], RBF [5], and MTLD-Fast) and iteration-based approaches (SVR [6], BO [8], and MTLD-Optim.). Both MTLD-Fast and MTLD-Optim. generate 1000 solutions during synthesis, and select the most feasible one (with the best *DoC*) as the final solution. We randomly generate 100 test specifications as design targets for each system and summarize the synthesis accuracy and efficiency in Table 2. The distributions of relative impedance errors for single-ended MTL are plotted in Fig. 6. In the scenario of end-to-end approaches, MTLD-Fast can achieve the highest accuracy, i.e., 99.82% and 99.83% for the single-ended MTL and waveguide of coupled MTLs, respectively, and displays the highest efficiency. In the scenario of iterative approaches, MTLD-Optim. can further improve the accuracy to over 99.8%, while SVR and BO cannot stably achieve higher than 99% accuracy. Additionally, the runtime of the flow on both CPU and GPU is comparably short, due to the small size of the models and generated solutions.

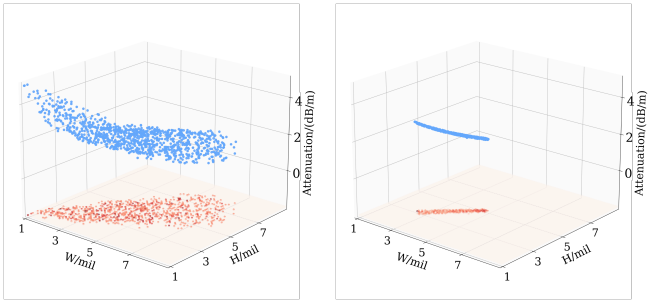


(a) End-to-end Approaches. **(b) Iterative Approaches.**

Figure 6: Count of the relative impedance error of design solutions by different methods for the single-ended MTL synthesis. The horizontal axis is transformed for clarity.

4.5 Effects of Design Optimization

Finally, we demonstrate the effects of design optimization in Fig. 7, illustrating the distribution of W, H, and attenuation of generated solutions before and after the constrained optimization for single-ended MTL. We impose three types of constraints: (1). linear equality: Tp, T and ϵ to be constant; (2). linear inequality : $2 < W, H < 5$; and (3) soft constraint: to minimize the attenuation of the MTL, which is approximately proportional to $W^{-0.23}$ according to prior knowledge. Regularization weight $\lambda = 100$ to balance the magnitude of two losses (impedance deviation and regularization of attenuation). We can see from Fig. 7(a) that the initial solutions spread over the plane, providing abundant exploration of the design space. After the constrained optimization, the solutions gather in the required value range, with W generally larger than the lower bound for smaller attenuation, as indicated by Fig. 7(b).



(a) Initial Solutions. **(b) Optimized Solutions.**

Figure 7: Distributions of the solutions projected onto the (W, H) plane and corresponding attenuation values, (a) before and (b) after the constrained optimization given target (24GHz, 75Ω).

5 CONCLUSION

Determining the configurations of MTLs given design specifications and prior knowledge is one of the key problems in the MIC design. In this work, we propose an integrated analysis and synthesis flow for MTL design, named MTL-Designer. We adopt CVAE to generate multiple initial solutions, in contrast to traditional algorithms that generate only one or a few. To predict the characteristics accurately and efficiently, we customize a surrogate analysis model composed of an individual learning part and a feature aggregation part. The analysis model is utilized to optimize the initial solutions, reducing their deviation of characteristic from targets and accommodating them for various design-knowledge-based constraints. We validate the flow with the commercial simulator *TmlExpert*. After training, MTLD can provide 1000 feasible solutions within 0.6 s, with an accuracy > 99.8% for designing both single-ended MTL and waveguide of coupled-MTLs, satisfying various design constraints. We believe our flow can shed more light on the design of not only MTLs but also other transmission line structures.

ACKNOWLEDGE

This work was supported in part by the National Science Foundations of China (Grant No. 62141404 and No. 62125401) and the 111 project (B18001).

REFERENCES

[1] R. Garg, I. Bahl, and M. Bozzi, *Microstrip lines and slotlines*. Artech house, 2013.
[2] R. Jackson, "Full-wave, finite element analysis of irregular microstrip discontinuities," *IEEE Transactions on Microwave Theory and Techniques*, vol. 37, no. 1, pp. 81–89, 1989.
[3] A. Djordjevic and T. Sarkar, "Closed-form formulas for frequency-dependent resistance and inductance per unit length of microstrip and strip transmission lines," *IEEE Transactions on Microwave Theory and Techniques*, vol. 42, no. 2, pp. 241–248, 1994.
[4] F. Güneş and N. Türker, "Artifial neural networks in their simplest forms for analysis and synthesis of rf/microwave planar transmission lines," *International Journal of RF and Microwave Computer-Aided Engineering*, vol. 15, pp. 587–600, 2005.
[5] J. L. Narayana, K. S. R. Krishna, and L. P. Reddy, "Design of microstrip antennas using artificial neural networks," in *International Conference on Computational Intelligence and Multimedia Applications*, vol. 1. IEEE, 2007, pp. 332–334.
[6] F. Güneş, N. T. Tokan, and F. Gürgen, "Support vector design of the microstrip lines," *International Journal of RF and Microwave Computer-Aided Engineering*, vol. 18, no. 4, pp. 326–336, 2008.
[7] J. Chen and L. He, "Modeling and synthesis of multiport transmission line for multichannel communication," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 25, no. 9, pp. 1664–1676, 2006.
[8] D. De Witte, J. Qing, I. Couckuyt, T. Dhaene, D. Vande Ginste, and D. Spina, "A robust bayesian optimization framework for microwave circuit design under uncertainty," *Electronics*, vol. 11, no. 14, 2022.
[9] K. S. R. Krishna, J. L. Narayana, and L. P. Reddy, "Ann models for microstrip line synthesis and analysis," *Int. J. Elect. Syst. Sci. Eng*, vol. 1, no. 3, pp. 196–200, 2008.
[10] T. Khan and A. De, "Modeling of microstrip antennas using neural networks techniques: a review," *International Journal of RF and Microwave Computer-Aided Engineering*, vol. 25, no. 9, pp. 747–757, 2015.
[11] H. Du, Q. Yang, X. Dai, X. Liao, and A. Zhang, "A parameter extraction method for lc circuit of db-bpf based on fully connected network," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 41, no. 10, pp. 3558–3562, 2022.
[12] Xpeedic, "TmlExpert," http://xpeedic.com/index.php?m=content&c=index&a=show&catid=29&id=143, 2020.
[13] C. J. Railton and J. P. Mcgeehan, "Characterisation of microstrip open-end with rectangular and trapezoidal cross-section," *Electronics Letters*, vol. 26, pp. 685–686, 1990.
[14] M. Alam, K. Hirayama, Y. Hayashi, and M. Koshiba, "Analysis of shielded microstrip lines with arbitrary metallization cross section using a vector finite element method," *IEEE Transactions on Microwave Theory and Techniques*, vol. 42, no. 11, pp. 2112–2117, 1994.
[15] S. Bond-Taylor, A. Leach, Y. Long, and C. G. Willcocks, "Deep generative modelling: A comparative review of vaes, gans, normalizing flows, energy-based and autoregressive models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 11, pp. 7327–7347, 2022.
[16] K. Sohn, X. Yan, and H. Lee, "Learning structured output representation using deep conditional generative models," in *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2*, ser. NIPS'15, 2015, p. 3483–3491.
[17] D. J. Rezende, S. Mohamed, and D. Wierstra, "Stochastic backpropagation and approximate inference in deep generative models," in *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32*, ser. ICML'14, 2014.
[18] J. Eason and S. Cremaschi, "Adaptive sequential sampling for surrogate model generation with artificial neural networks," *Comput. Chem. Eng.*, vol. 68, pp. 220–232, 2014.
[19] D. G. Luenberger and Y. Ye, *Basic Descent Methods*. Springer International Publishing, 2021, pp. 235–300.
[20] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama, "Optuna: A next-generation hyperparameter optimization framework," *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2019.