2024 International Conference on Simulation of Semiconductor Processes and Devices (SISPAD) | 979-8-3315-1635-2/24/$31.00 ©2024 IEEE | DOI: 10.1109/SISPAD62626.2024.10733127

Neural Drift-Diffusion Model Based on Operator Learning in Fourier Space

Kyeyeop Kim, Sanghoon Myung*, Yunji Choi, Gijae Kang, Kyungmi Yeom,
Songyi Han, Jaehoon Jeong, and Dae Sin Kim
Computational Science and Engineering Team, Innovation Center, Samsung Electronics
Email: *shoon.myung@samsung.com

***Abstract*—We present a novel neural drift-diffusion model that aims to emulate the process of solving the drift-diffusion equation. By utilizing Fourier neural operators, the model learns the underlying physics of the drift-diffusion equation and accurately predicts device characteristics. Our approach stands out from previous studies by representing the learned principle as a generalized Green's function in Fourier space, which maps between the solutions of the initial and the next biases. This capability allows our model to provide precise solutions even with unseen inputs in terms of doping profiles or biases.**

***Index Terms*—Drift-diffusion, Partial differential equations, Operator learning, Neural operator, Fourier space**

I. Introduction

As the complexity of semiconductor fabrication processes has increased due to miniaturization, the cost of process development has risen. To minimize the cost, many industries have thus employed Technology Computer-Aided Design (TCAD) simulation that provides a cost-effective means of pre-designing devices and predicting their electrical characteristics. Nevertheless, the growing complexity of three-dimensional (3-D) devices has led to escalated computational expenses and reduced convergence rate in TCAD simulations [1], [2]. To address these issues, numerous studies have attempted to introduce deep learning techniques through various approaches [3]–[12]. However, these methods usually tend to memorize patterns in the data rather than fulfilling the fundamental role of the solver, leading to poor accuracy in particular extrapolation case. To mitigate this issue, we present a neural drift-diffusion (NDD) model aimed at solving the drift-diffusion (DD) equation precisely. The rest of this paper is divided into the following sections. Section II describes a generalized Green‘s function and introduces NDD model that can learn the generalized Green‘s function. Section III shows experimental results that NDD model can successfully solve drift-diffusion equation and be compatible with TCAD. Section IV summarizes our contribution.

II. Methodologies

A. Preliminaries

In the realm of semiconductor device modeling, TCAD device simulators play a crucial role in predicting the electrical characteristics of various devices by solving nonlinear partial differential equations (PDEs), commonly known as the DD equation. Due to the inherent non-linearity, it is not feasible to solve the DD equation directly. Therefore simulators must rely on numerical methods such as the Newton-Raphson method to approximate the solutions. The Newton-Raphson method is an iterative technique that seeks to find the solutions of a given function by linearizing the function around an initial guess and iteratively improving the guess until a desired level of accuracy is achieved. This iterative solving process can be described as follows:

$$\begin{aligned} u(r) &= u(\phi, n, p; N)(r), \quad r \in \Omega \\ \mathcal{L}(u_{i+1})(r) &= u_i(r), \quad r \in \Omega \\ \mathcal{B}(u)(r) &= h(r; V), \quad r \in \delta\Omega, V \in \mathbb{R}^n \end{aligned} \tag{1}$$

where u represents a coupled solution of the DD equation that includes the electrostatic potential ϕ, electron density n, and hole density p, all dependent on the given doping profile N. The DD operator $\mathcal{L}$ maps the initial solution u_i to the next solution u_{i+1} within the domain Ω. The boundary operator $\mathcal{B}$ maps the solution u to the boundary function h at the boundary $\delta\Omega$ of the domain. In this case, we set h as the Dirichlet boundary condition, as TCAD simulators typically solve the DD model according to the constant voltages V applied to n contacts such as the drain, source, and gate.

B. Neural drift-diffusion model

As evident from Eq. (1), the solution is heavily dependent on the initial solution. In other words, it is crucial to obtain an appropriate initial solution. However, it is challenging to guess the initial solution except for the equilibrium state. Previous works [6], [7] have therefore employed deep learning models to suggest good initial solutions to TCAD simulators. However, these approaches may over-fit the pattern of the data rather than performing the fundamental role of the solver, leading to errors beyond the training range. To address this problem, we aim to enable the model to learn the solver's role through operator learning. The solution u in Eq. (1) can be obtained by utilizing the generalized Green's function as follows:

$$\begin{aligned} u_{i+1}(r) = &\int_\Omega u_i(\xi)\mathcal{G}(r,\xi)d\xi \\ &+ \int_\Omega \nabla \cdot (h_{i+1}(\xi)\nabla\mathcal{G}(r,\xi))d\xi, \end{aligned} \tag{2}$$

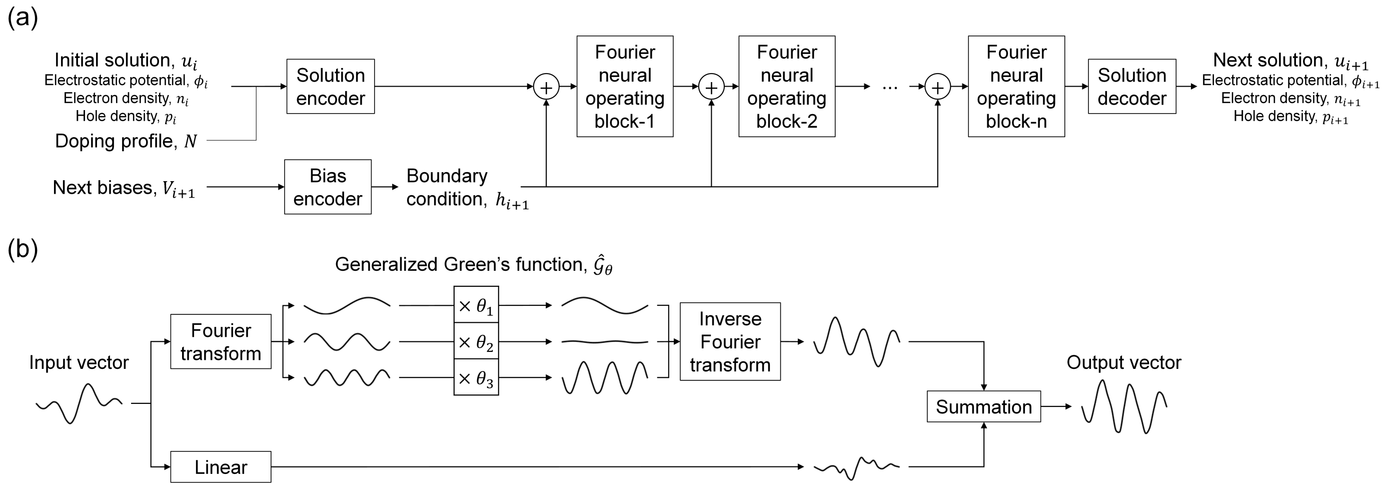

Fig. 1. Illustration of the architecture of (a) NDD and (b) Fourier neural operator block that is induced to learn the knowledge in Fourier space.

where $\mathcal{G}$ is so-called generalized Green's function. The generalized Green's function ($\mathcal{G}$), if accessible, would enable an easy mapping of the inputs (u_i, N, and h_{i+1}) to the next solution (u_{i+1}). However, acquiring $\mathcal{G}$ poses a significant challenge. We hence employ a deep learning model that can learn this $\mathcal{G}$ from data. In Eq. (2), if $\mathcal{G}$ satisfies the property $\mathcal{G}(r,\xi) = \mathcal{G}(r-\xi)$, the kernel integral operation can be transformed into a linear operation in the Fourier space.

$$\begin{aligned} u_{i+1}(r) &\simeq \mathcal{F}^{-1}(\mathcal{F}(\mathcal{G}) \cdot \mathcal{F}(u_i \oplus h_{i+1}))(r) \\ &\simeq \mathcal{F}^{-1}(\mathcal{F}(\hat{\mathcal{G}}_\theta) \cdot \mathcal{F}(u_i \oplus h_{i+1}))(r) \end{aligned} \quad (3)$$

Capitalizing on this property, we design the NDD model that can approximate the $\mathcal{G}$ with $\hat{\mathcal{G}}_\theta$ in the Fourier space. Fig. 1 (a) illustrates an overall architecture of NDD model that utilizes the Fourier Neural Operator (FNO) blocks [13] in iterative manners, including three modules: i) solution encoder, ii) bias encoder, and iii) solution decoder.

Fourier neural operator. Fig. 1 (b) shows an architectural detail of the FNO block. Each FNO block transforms the input vectors into Fourier space using Fast Fourier Transform (FFT, $\mathcal{F}$) [14], computes the parameters (θ), and subsequently returns the output vector after performing the inverse FFT ($\mathcal{F}^{-1}$). The stacked FNO blocks approximate the generalized Green's function ($\hat{\mathcal{G}}_\theta$). In this work, we convert unstructured meshes into structured ones featuring uniform intervals to facilitate the training process [7]. Thus, the solutions are limited in the regular grids, so that Fourier and inverse Fourier transforms can be conducted with low costs by FFT algorithm [13].

Solution encoder. The initial solution (u_i) and the doping profile (N) are encoded by the solution encoder. This encoder serves to elevate an input vector to the high-dimensional representation. In our implementation, we use few fully-connected layers to fulfill this function.

Bias encoder. The next biases (V_{i+1}) are encoded by the bias encoder. The bias encoder is responsible for lifting V_{i+1} to the high-dimensional representation and concatenate it to an output of solution encoder similar to the solution encoder. This high-dimensional representation can function as the boundary condition (h_{i+1}). It is important to note that the bias differences (ΔV) between neighboring steps is limited to a constant during the training phase. We thus eliminate bias of previous step (V_i) to avoid redundancy.

Solution decoder. This module aims to decode the latent of u_i. In this work, we employ a couple of fully-connected layers to aggregate the output vector of the last FNO block to u_{i+1}.

III. Experiments

In this section, we evaluate the effectiveness of the NDD model. In particular, we emphasize the importance of extrapolation in evaluating the model's performance as a partial differential equation (PDE) solver, as previous works are inherently limited to interpolation tasks [6], [7], [12]. The interpolation accuracy of the trained NDD model was also above 0.99.

We first describe the fabricated process and the data used for training in Section III-A. Next, we assess whether NDD model can handle input conditions that has never encountered during the training. In Sec. III-B and III-C, we conduct an extrapolation test with respect to doping and voltage conditions, respectively. Finally, Sec. III-D test whether the solutions obtained by the NDD model converge well in the TCAD simulator.

A. Experimental Environment

In this study, we conducted experiments on n-type metal-oxide-semiconductor field-effect transistors (MOSFETs) fabricated in a 28-nanometer (nm) process (Fig. 2(a)). To generate the training data, we utilized an in-house TCAD simulator (Polaris), varying the doping profile and applied biases, as shown in Table I.

For the training dataset, we modified the doping profile solely by adjusting the conditions of the lightly doped drain (LDD) and channel implantation during the process simulation (Fig. 2 (b)). In contrast, for the test dataset, we varied the doping profile based on the conditions of halo implantation

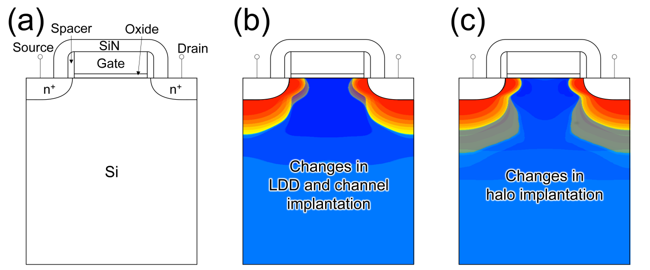
(a)
Spacer
Oxide
Source
SiN
Gate
Drain
n+
n+
Si
(b)
Changes in LDD and channel implantation
(c)
Changes in halo implantation

Fig. 2. (a) Description on 28-nm n-type MOSFET. Illustration how to vary the ion-implantation for (b) in-training and (c) out-of-training data, respectively.

TABLE I
DESCRIPTION OF EXPERIMENTAL SETUP. THE EXPERIMENTAL SETUP INVOLVES VARYING DOPING PROFILE, GATE BIAS, AND DRAIN BIAS. THE TEST DATA ARE POSITIONED IN AN UNSEEN DOMAIN REGION.

	Train dataset Variation	#	Test dataset Variation	#
Doping profile (N)	LDD Channel	1000	Halo	10
Gate bias (V_{GS})	0.05∼1.0 [V]	20	-0.45∼0.0 [V]	10
Drain bias (V_{DS})	0.05∼1.0 [V]	20	-0.45∼0.0 [V]	10

(Fig. 2 (c)). The numbers of doping profiles in the training and test datasets were 1,000 and 10, respectively. In addition to the doping profiles, we also varied the drain bias (V_{DS}) and gate bias (V_{GS}). For the training dataset, the biases were set at intervals of 0.05 volts [V], ranging from 0.05 [V] to 1 [V], respectively. The test dataset varied from -0.45 [V] to 0.0 [V]. As a result, the total number of training and test data points amounted to 400,000 and 1,000, respectively.

With the training dataset, we trained the NDD model to ramp up a regular bias step of 0.05 [V]. The parameters (θ) of the NDD model were optimized to align the predicted solution ($\hat{u}_{i+1}$) with the results of TCAD simulator (u_{i+1}) at the next bias.

B. Extrapolation on Doping Profile

As aforementioned above, NDD model was exclusively trained to recognize changes in LDD and channel implantation. Consequently, we subjected the model to the unseen doping profiles that modify the halo implantation. The test experiments were conducted on ten such unseen doping profiles. The target solutions for V_{DS} ranging from 0.05 [V] to 1.0 [V] were predicted from the previous solutions for V_{DS} ranging from 0.0 [V] to 0.95 [V], with a fixed V_{GS} spanning from 0.0 [V] to 1.0 [V]. Similarly, the target solutions for V_{DS} ranging from 0.05 [V] to 1.0 [V] were predicted. The accuracies of the extrapolated solutions for the target V_{DS} and V_{GS} were found to be 0.98 and 0.99, respectively. Fig. 3 depicts a comparison between the TCAD and the NDD solutions for an unseen doping profile at 0.05 [V] of V_{DS} and 0.0 [V] of V_{GS}. The figure demonstrates that the NDD model can predict solutions nearly identical to the TCAD results for the previously unseen doping profile.

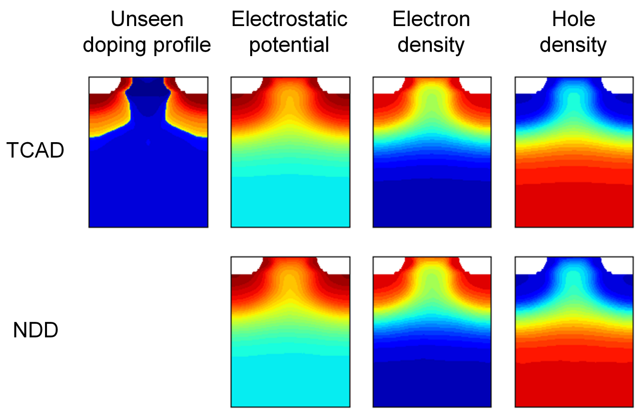

Fig. 3. Comparison of solutions between TCAD and the NDD model when presented with an unseen doping profile as input. Both solutions for electrostatic potential, electron, and hole density are indistinguishable.

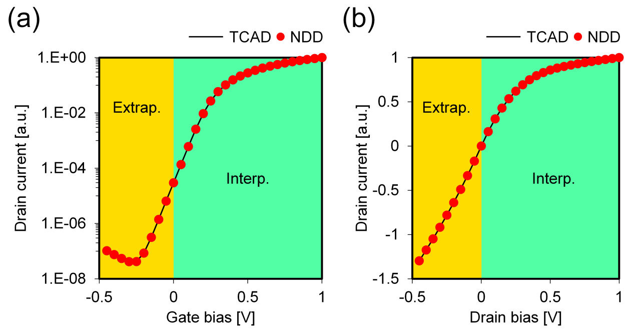

Fig. 4. (a) Comparison of (a) I_D-V_G and (b) I_D-V_D curves. Green and red indicate the regions of interpolation and extrapolation for NDD, respectively.

C. Extrapolation on Bias

Because the NDD model was trained from 0.05 [V] to 1 [V], we assessed whether our method can present accurate solution out-of-training range or not. We therefore applied the reverse biases to the V_{GS} and V_{DS} respectively, then incrementally increased the each bias. The solutions for V_{GS} ranging from -0.45 [V] to 0.0 [V] were predicted from the initial solutions for V_{GS} varied from -0.5 [V] to -0.05 [V] and 1.0 [V] of V_{DS}. Similarly, the solutions for V_{DS} from -0.45 [V] to 0.0 [V] were predicted. The accuracy of the extrapolated solutions for the target V_{GS} and V_{DS} is 0.94 and 0.99, respectively. To validate the accuracy of our model, we solved the DD model using the predicted solutions as the initial guess with our in-house TCAD simulator (Polaris) and extracted the drain current. Fig. 4 demonstrates that the NDD model can accurately calculate the solutions for the untrained biases of (a) V_{GS} and (b) V_{DS}, respectively. Especially, Fig. 4(a) shows that our model can accurately predict the Gate-Induced Drain Leakage (GIDL) phenomenon under the unseen negative gate bias.

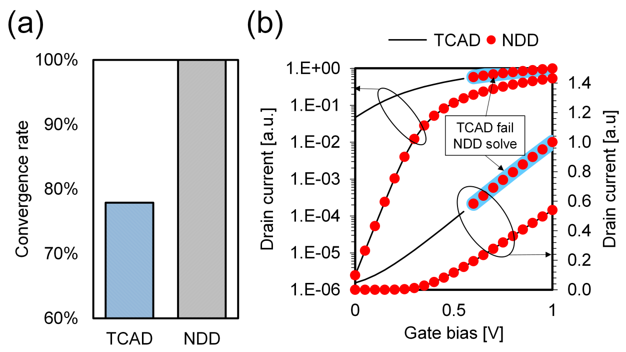

Fig. 5. (a) Convergence rate of TCAD and NDD. (b) An example of I_D-V_G curve showing NDD can calculate I_D-V_G curve even if TCAD cannot.

D. Replacement of TCAD operation with NDD

TCAD simulators often solve the drift-diffusion equation starting from zero bias (i.e., thermal equilibrium) and then gradually increase the applied bias to the target. However, TCAD simulator often fails to solve DD equation due to the mesh quality. In fact, during the generation of the TCAD dataset, only 78 % of the calculations achieved convergence (Fig. 5 (a)). Leveraging the capability of NDD model that can directly solve DD equation, we were able to subject the model to solutions that TCAD simulators were unable to find. Indeed, with the aid of the NDD model, TCAD simulators can now solve the DD equation at the desired bias conditions that were previously not converged. Fig. 5 (b) shows that NDD model could succeed even for previously failed bias points, and the convergence rate of NDD became 100 % as shown in Fig. 5 (a).

IV. Discussion and Conclusion

We have shown that the NDD model can accurately extrapolate to the unseen doping profiles and applied biases. The accuracy of extrapolation has been significantly enhanced by introducing operator learning through Fourier space and carefully considering boundary conditions. Our finding has established the potential for deep learning models as alternative of TCAD simulator burdened by high computational costs. However, some challenges persist in fully substituting it with a deep learning model. First, it suffers inefficiency due to the limitation of inputs confined to regular grids. Integration of graph-based deep learning models [12] with the NDD model is expected to alleviate inefficiencies by directly utilizing mesh information as input. Secondly, the NDD model is limited to predicting steady-state solutions and cannot infer transient-state solutions. To address this issue, we propose expanding the NDD model to learn time-dependent partial differential equations. Finally, the model necessitate retraining when processes change or new processes are introduced. We believe that retraining issue are addressed by combining NDD model with transfer learning [15], [16] or meta learning [17]. These studies will be left for future research.

References

[1] A. Toselli and O. Widlund, *Domain Decomposition Methods - Algorithms and Theory*, ser. Springer Series in Computational Mathematics. Springer Berlin Heidelberg, 2006. [Online]. Available: https://books.google.co.kr/books?id=h7EVoI2g1nkC

[2] H. Koshimoto, H. Ishimabushi, J. Yoo, Y. Kayama, S. Yamada, U. Kwon, and D. S. Kim, "Gummel-cycle algebraic multigrid preconditioning for large-scale device simulations," in *2020 International Conference on Simulation of Semiconductor Processes and Devices (SISPAD)*, 2020, pp. 51–54, doi: https://doi.org/10.23919/SISPAD49475.2020.9241643.

[3] C. Jeong, S. Myung, I. Huh, B. Choi, J. Kim, H. Jang, H. Lee, D. Park, K. Lee, W. Jang *et al.*, "Bridging tcad and ai: Its application to semiconductor design," *IEEE Transactions on Electron Devices*, vol. 68, no. 11, pp. 5364–5371, 2021, doi: https://doi.org/10.1109/TED.2021.3093844.

[4] K. Mehta and H.-Y. Wong, "Prediction of finfet current-voltage and capacitance-voltage curves using machine learning with autoencoder," *IEEE Electron Device Letters*, vol. 42, no. 2, pp. 136–139, 2021, doi: https://doi.org/10.1109/LED.2020.3045064.

[5] H. Dhillon, K. Mehta, M. Xiao, B. Wang, Y. Zhang, and H. Y. Wong, "Tcad-augmented machine learning with and without domain expertise," *IEEE Transactions on Electron Devices*, vol. 68, no. 11, pp. 5498–5503, 2021, doi: https://doi.org/10.1109/TED.2021.3073378.

[6] S.-C. Han, J. Choi, and S.-M. Hong, "Acceleration of semiconductor device simulation with approximate solutions predicted by trained neural networks," *IEEE Transactions on Electron Devices*, vol. 68, no. 11, pp. 5483–5489, 2021, doi: https://doi.org/10.1109/TED.2021.3075192.

[7] S. Myung, W. Jang, S. Jin, J. M. Choe, C. Jeong, and D. S. Kim, "Restructuring tcad system: Teaching traditional tcad new tricks," in *2021 IEEE International Electron Devices Meeting (IEDM)*, 2021, pp. 18.2.1–18.2.4, doi: https://doi.org/10.1109/IEDM19574.2021.9720616.

[8] S. Myung, B. Choi, W. Jang, J. Kim, I. Huh, J. M. Choe, Y.-G. Kim, and D. S. Kim, "Comprehensive studies on deep learning applicable to tcad," *Japanese Journal of Applied Physics*, vol. 62, no. SC, p. SC0808, 2023, doi: https://doi.org/10.35848/1347-4065/acbaa6.

[9] W.-J. Lee, W.-T. Hsieh, B.-H. Fang, K.-H. Kao, and N.-Y. Chen, "Device simulations with a u-net model predicting physical quantities in two-dimensional landscapes," *Scientific Reports*, vol. 13, no. 731, pp. 1–9, 2023, doi: https://doi.org/10.1038/s41598-023-27599-z.

[10] L. Ruthotto and E. Haber, "Deep neural networks motivated by partial differential equations," *Journal of Mathematical Imaging and Vision*, vol. 62, pp. 352–364, 2020, doi: https://doi.org/10.1007/s10851-019-00903-1.

[11] B. Kim and M. Shin, "A novel neural-network device modeling based on physics-informed machine learning," *IEEE Transactions on Electron Devices*, vol. 70, no. 11, pp. 6021–6025, 2023, doi: https://doi.org/10.1109/TED.2023.3316635.

[12] W. Jang, S. Myung, J. M. Choe, Y.-G. Kim, and D. S. Kim, "Tcad device simulation with graph neural network," *IEEE Electron Device Letters*, vol. 44, no. 8, pp. 1368–1371, 2023, doi: https://doi.org/10.1109/LED.2023.3290930.

[13] Z. Li, N. B. Kovachki, K. Azizzadenesheli, B. liu, K. Bhattacharya, A. Stuart, and A. Anandkumar, "Fourier neural operator for parametric partial differential equations," in *International Conference on Learning Representations*, 2021. [Online]. Available: https://openreview.net/forum?id=c8P9NQVtmnO

[14] E. O. Brigham, *The fast Fourier transform and its applications*. Prentice-Hall, Inc., 1988.

[15] Y. Choi, S. Myung, K. Kim, G. Kang, B. Jeong, Y. Jeon, S. Han, J. Jeong, and D. S. Kim, "Real time tcad calibration via transfer learning," in *2020 International Conference on Simulation of Semiconductor Processes and Devices (SISPAD)*, 2024.

[16] S. Myung, I. Huh, W. Jang, J. M. Choe, J. Ryu, D. Kim, K.-E. Kim, and C. Jeong, "Pac-net: A model pruning approach to inductive transfer learning," in *International Conference on Machine Learning*. PMLR, 2022, pp. 16 240–16 252. [Online]. Available: https://proceedings.mlr.press/v162/myung22a.html

[17] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *International conference on machine learning*. PMLR, 2017, pp. 1126–1135. [Online]. Available: https://proceedings.mlr.press/v70/finn17a.html