End-to-End Differentiability and Tensor Processing Unit Computing to Accelerate Materials' Inverse Design

Han Liu, Yuhan Liu, Zhangji Zhao, Mathieu Bauchy*
Physics of AmoRphous and Inorganic Solids Laboratory (PARISlab),
Department of Civil and Environmental Engineering,
University of California, Los Angeles, California, USA
{happylife, yliu1021, zzhao13, bauchy}@ucla.edu

Samuel S. Schoenholz, Ekin D Cubuk
Google Research, Brain Team, Mountain View, California, USA
{schsam, cubuk}@google.com

Abstract

Numerical simulations have revolutionized material design. However, although simulations excel at mapping an input material to its output property, their direct application to inverse design (i.e., mapping an input property to an optimal output material) has traditionally been limited by their high computing cost and lack of differentiability—so that simulations are often replaced by surrogate machine learning models in inverse design problems. Here, taking the example of the inverse design of a porous matrix featuring targeted sorption isotherm, we introduce a computational inverse design framework that addresses these challenges. We reformulate a lattice density functional theory of sorption as a differentiable simulation programmed on TensorFlow platform that leverages automated end-to-end differentiation. Thanks to its differentiability, the simulation is used to directly train a deep generative model, which outputs an optimal porous matrix based on an arbitrary input sorption isotherm curve. Importantly, this inverse design pipeline leverages for the first time the power of tensor processing units (TPU)—an emerging family of dedicated chips, which, although they are specialized in deep learning, are flexible enough for intensive scientific simulations. This approach holds promise to accelerate inverse materials design.

1 Introduction

Numerical simulations have transformed the way we design materials (1). For instance, density functional theory and molecular dynamics excel at predicting the properties of materials based on the knowledge of their composition and atomic structure (2; 3). This makes it possible to replace costly trial-and-error experiments by simulations so as to screen in silico promising materials (4). However, numerical simulations are of limited help to tackle "inverse design" problems (i.e., identifying an optimal material featuring optimal properties within a given design space) (5; 6; 7). Indeed, although numerical simulations are typically faster and cheaper than experiments, their computational burden usually prevents a thorough exploration of the design space (e.g., the systematic exploration of all possible materials' compositions) (8). In addition, traditional numerical simulations are usually not differentiable, which prevents their seamless integration with gradient-based optimization methods

*corresponding author: bauchy@ucla.edu, http://www.lab-paris.com

Workshop on machine learning for engineering modeling, simulation and design @ NeurIPS 2020

(9; 10). These limitations—which are reminiscent of the state of machine learning before automatic differentiation became popular (11)—have limited the use of numerical simulations in inverse design pipelines (12).

To address this issue, it is common to replace simulations by a differentiable surrogate predictor machine learning model, which aims to approximately interpolate the mapping between design space parameters (e.g., the material's structure) and the target property of interest (7; 12; 13). Following this approach, Generative Networks (GNs) (5) have been used for inverse design application using, for instance, autoencoders (14), generative adversarial networks (15), or generative inverse design networks (12). The generator can then be combined with the differentiable surrogate predictor in the same pipeline so as to be trained by gradient backpropagation (12; 16; 17). However, this approach can result in difficulties associated with the fact that the generator and predictor must both be trained, either simultaneously or sequentially. In addition, the ability of the generator to discover new unknown, potentially non-intuitive material designs (i.e., which are very different from those in the training set) is often limited by the accuracy and generalizability of the surrogate predictor (5; 6; 7).

Here, to address these challenges, we introduce a deep generative pipeline that combines an end-to-end differentiable simulator with a generator model. We illustrate the power of this approach by taking the example of the inverse design of a porous matrix featuring targeted sorption isotherm—wherein the sorption isotherm corresponds here to the amount of adsorbed liquid water in the porous structure as a function of relative humidity. This is enabled by the implementation of an end-to-end differentiable lattice-based density functional theory code in TensorFlow (18; 19). We show that the trained generative model is able to successfully generate porous structures with arbitrary sorption curves. Moreover, this generator-simulator pipeline leverages for the first time the power of tensor processing units (TPU)—an emerging family of dedicated chips (20), which, although they are specialized in deep learning, are flexible enough for intensive scientific simulations. This approach holds promise to accelerate the inverse design of materials with tailored properties and functionalities.

2 Related work

TPU in scientific simulations. TPU is a family of dedicated chips that assemble different computing units for machine learning applications (21). In contrast to general purposes processors (i.e., CPUs and GPUs), TPUs are specifically designed as matrix processors thanks to their matrix unit (MXU) (22; 23). Although TPUs have been extensively used for deep learning, their application to numerical simulations has thus far remained limited an Ising model (24). However, TPUs exhibit enough flexibility to have the potential to accelerate a broader range of computations.

End-to-end differentiable simulations. With the recent expansion of automatic differentiation technologies (25; 26), differentiable programming platforms—such as TensorFlow (18), JAX (19), and TaiChi (27)—are rapidly developing and getting attention for differentiable simulation applications (28; 29), including molecular dynamics [[10,11]] and robotic dynamics (30). However, it has never been applied to lattice-based density functional theory simulations.

Generative Networks for materials' inverse design. Many recent works have been using a generator combined with a surrogate predictor trained based on a simulator for inverse design applications (6; 7; 12; 13; 16; 17). For example, Gu *et al.* recently developed generative inverse design networks to discover composite structures featuring optimal target mechanical properties (12). However, the potential of directly training the generator based on a differentiable predictor has received less attention.

3 End-to-end differentiable simulator

We first focus on the end-to-end differentiable implementation of the sorption simulator used herein. We consider as a toy model a square N-by-N lattice, wherein each pixel i of the grid can be filled with solid or be a pore (see Figure 1a). Initially empty pixel can then be filled with water upon increasing relative humidity (RH). In a given configuration, the state of each pixel i is given by the knowledge of (η_i, ρ_i), where $\eta_i = 0$ and 1 indicate that the pixel if filled with solid or is a pore, respectively, and ρ_i is the density of water in the pore ($\rho_i = 0$ and 1 denote that the pore is fully empty or saturated with water, respectively). The equilibrium fraction of water in each pore at given temperature T and

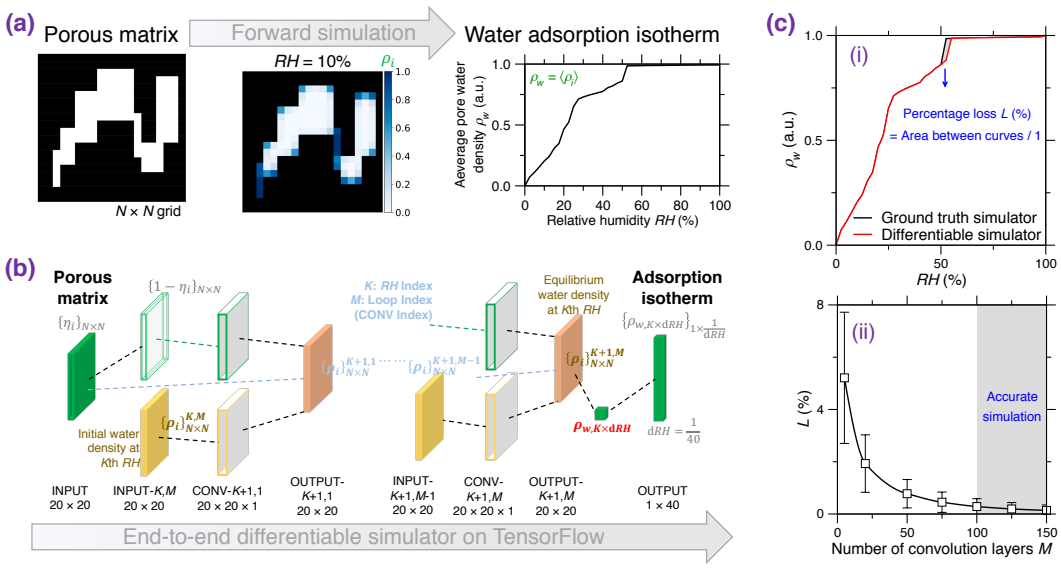

Figure 1: End-to-end differentiable simulation of water adsorption in porous materials. **(a)** Illustration of the numerical water sorption simulation for a target porous matrix. **(b)** End-to-end differentiable reformation of the sorption simulation as a series of convolutional layers in TensorFlow. **(c i)** Comparison between the sorption curve ground-truth (undifferentiable) sorption simulator and its reformulated differentiable counterpart for the porous matrix shown in panel (a), which defines the percentage loss. **(c ii)** Average percentage loss as a function of the number of convolution layers.

RH is then solved by lattice density functional theory (LDFT) (31; 32). Based on this formalism, the water density ρ_i at a given pixel i is given by:

$$\rho_i = \eta_i / (1 + e^{-\{\mu + \Sigma_{j/i}[w_{\rm ff}\rho_j + w_{\rm mf}(1-\eta_j)]\}/kT}) \tag{1}$$

where μ is the chemical potential (which depends on RH), k is the Boltzmann constant, $w_{\rm ff}$ is interaction energy between two neighboring pixels that are filled with water, $w_{\rm mf}$ is the interaction energy between a pixel filled with water and a substrate (i.e., a neighboring pixel filled with solid), and j/i are the pixel IDs of the 4 neighbors of pixel i (note that, to avoid any surface effect, periodic boundary conditions are applied) (32). As such, the water density at a given pixel i depends on the state of its 4 neighbors, which is essentially a convolution operation. At fixed RH, the equilibrium fraction of water is then determined by iteratively applying Eq. (1) on each pixel until a convergence in the ρ_i values is obtained. The sorption of water in the porous matrix is then iteratively simulated by computing the equilibrium values of ρ_i for RH = 0-to-100% with an increment dRH. At each increment K, the equilibrium water density values $\{\rho_i\}^{K\rm th}$ at RH $= K \times$ dRH serve as starting configuration to calculate $\{\rho_i\}^{K+1}$ at the subsequent step $K+1$. More details of the numerical simulations can be found in Ref. (32).

Such (LDFT) simulations are traditionally not differentiable. Here, to address this limitation, we decompose Eq. (1) into a series of mathematical operations that can be implemented as differentiable computation layers in TensorFlow (Figure 1b). For instance, the CONV layer represents the convolution operation in Eq. (1)—i.e., one of the operations that can be efficiently performed by TPUs. This block is then repeated into M convolutional layers, which is equivalent to iteratively solving Eq. (1) until a convergence in the water density is achieved. Figure 1c shows the accuracy of the TensorFlow-based simulator as a function of M. We find that M = 100 offers satisfactory accuracy. Overall, by reformulating the LDFT simulation into a succession of convolutional layers, this approach enables end-to-end differentiability and TPU acceleration.

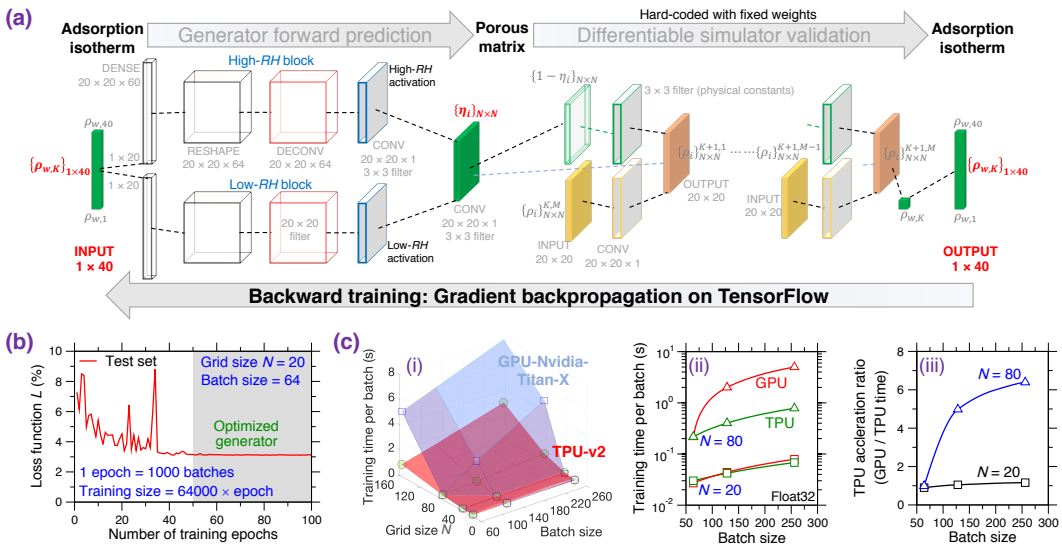

Figure 2: Training of the generative model by differentiable simulation and tensor processing unit (TPU) computing. **(a)** General architecture of the generator-simulator training pipeline. **(b)** Evolution of the test set loss function of a function of the number of training epochs. **(c)** Comparison of the training time per batch as a function of the grid size and batch size offered by Google's TPU-v2 and an NVIDIA TITAN X GPU.

4 Training of the generative model

4.1 Architecture of the generator-simulator pipeline

In this section, we first present the architecture of our generative model and its integration with the differentiable simulator presented in Sec. 3. Figure 2a shows the architecture of the generator-simulator pipeline. In detail, the training pipeline takes as inputs the sorption isotherms curves of the training set, which are transformed into porous matrices by the generator. The generated grids are then fed to the differentiable simulator to compute the "real" sorption curve of the generated porous matrices. In detail, the generator is designed as a dual, parallel deconvolution-block structure, where each block is fed with half of the input curve to decrease the generator complexity. These two blocks aim to specifically generate small and large pores, which are saturated with water at low and large RH, respectively. Since each layer of the pipeline is differentiable, the generator can then be optimized by gradient backpropagation in TensorFlow so as to minimize the difference between the input and output sorption curves. Note that, here, the convolutional layers of the simulator are hard-coded with fixed weights and, hence, are not optimized. This is key advantage of our approach since it avoids difficulties arising from the simultaneous optimization of the generator and predictor in traditional implementations of generative pipelines.

4.2 Training acceleration by Tensor Processing Unit computing

During the training process, a grid size of 20×20 yields about 7 million parameters to be optimized for the generator, while the simulator comprises about 4000 convolution layers to compute. Here, the generator is trained based on a training set of 6,400,000 sorption isotherm curves and then subsequently evaluated based on a test set of 8,769 curves. Figure 2b shows the evolution of the loss function L (i.e., the difference between input and output sorption curve, see Fig. 1c) as a function of the number of training epochs, wherein the batch size is set as 64 and each epoch contain 1000 batches. We find that the accuracy of the generator plateaus after 50 epochs (which corresponds to a training size of 3,200,000). Considering the large depth of the simulator and the number of parameters to be optimized in the generator, the training process comes with a significant computational cost. To mitigate this issue, as a pioneering experiment, the training is conducted on TPUs (20). Figure 2c show the training time per batch as a function of both grid size and batch size on a TPU-v2 chip with 8 cores and 64 GB memory (20). The computational performance is compared with the training

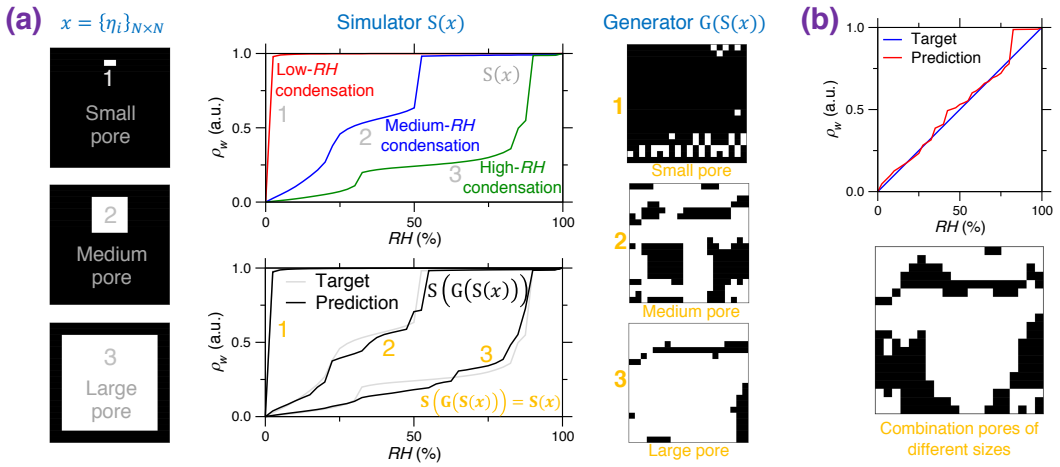

Figure 3: Accuracy of the generative model. **(a)** Illustration of three porous matrices that are generated so as to present three archetypical sorption isotherms associated with small, medium, and large pores. **(b)** Porous matrix generated for a target sorption curve $y = x$.

time yielded by a NVIDIA TITAN X GPU. All benchmarks are conducted on Google Colab using the same TensorFlow code and single precision (float32). We find that, especially for large grid size and batch size, the deliciated TPU hardware results in a training time that is several times faster than that offered by the GPU hardware considered herein (more than 6× faster). These results highlight the exciting, largely untapped potential of TPU computing in accelerating computationally-intensive scientific simulations (i.e., besides traditional deep learning applications).

4.3 Accuracy of the generator

Finally, we evaluate the accuracy of the trained generator on the test set (which comprises more than 8000 target sorption isotherms). After training, we find that the generator exhibits an average prediction loss of 3% (see Figure 2b), which is here considered very good. Figure 3a offers an illustration of three porous matrices that are generated so as to present three archetypical sorption isotherms wherein: (i) full water saturation occurs at very low RH (which arises in the presence of very small pores), (ii) water saturation is delayed and occurs at very large RH (which is a consequence of large pores), and (iii) an intermediate case (with medium-size pores). Overall, we find that the generator model is able to predict realistic porous matrices, with expected length scales for the pores. Importantly, the simulated sorption curves of the generated porous structures exhibit all the features (in terms of trend, convexity, and value) as the target sorption curves.

As a final test of the generator, we assess the ability of the generator to predict a porous structure featuring a target identity sorption curve $y = x$. This is an especially challenging test set case since (i) the sorption curve is not included in the training set, (ii) such a smooth sorption curve (with no sudden jump in water sorption) requires a complex, continuous pore size distribution, and (iii) this case corresponds to maximum degeneracy—unlike the cases of a 1-pixel or $(N-1)\times(N-1)$ pores, which present a limited number of possible solutions. Once again, we find that the generator yields a very realistic generated porous matrix, which, as expected, exhibits a combination of small, medium, and large pores (see Fig. 3b). Notably, the real sorption curve (computed by the simulator) of the generated porous matrix indeed exhibit a very close match with the $y = x$ target. This confirms that the generative model has learned the basic physical rules governing water sorption in porous media (e.g., small and large pores get saturated as low and high RH, etc.) and can successfully predict new unknown porous structures featuring tailored arbitrary sorption curves. In that regard, the fact that the generator is directly trained based on the simulator (rather than on surrogate model that approximates reality by learning from finite training set examples) is key to ensure that the generator is not limited by the accuracy of the predictor, or its ability to extrapolate predictions to grids it has never been exposed to during its training.

5 Conclusion

Overall, this work establishes a robust pipeline to enable the inverse design of materials by leveraging an end-to-end differentiable simulation as predictor. The fact that the generator is directly trained based on a simulator rather than on a surrogate machine learning model is key to ensure that the generator is not limited by the accuracy or extrapolation ability of the predictor. As a key enabler of this approach, we adopt TPUs to accelerate the training of the generator by gradient backpropagation in TensorFlow. This illustrates the exciting possibilities of TPU computing to accelerate scientific numerical simulations.

Broader Impact

This research has several scientific and societal implications. First, this work illustrates the benefits of integrating differentiable simulations in machine learning pipelines—which is key to accelerate the discovery of new materials. Second, our results establish TPU computing as a promising route to accelerate scientific simulations, which are ubiquitous in various applications (drug discovery by molecular dynamics, architectural design by finite element method, weather forecast predictions, etc.) (1; 2; 3). Finally, the ability to design new porous structures with tailored sorption isotherms could leapfrog several important applications, including for CO_2 capture (33; 34) and gas separation (35; 36). In addition, designing new porous structures featuring a smooth, continuous sorption isotherm (i.e., as close as possible to the $y = x$ target used herein) is important for drug delivery applications, to ensure that drugs are continuously released at a constant rate in a given environment (37; 38).

Acknowledgments and Disclosure of Funding

This research was supported by the National Science Foundation (DMREF-1922167). TPU computing time was provided by a grant allocation from Google's TensorFlow Research Cloud (TFRC) program.

References

[1] E. V. Levchenko, Y. J. Dappe, and G. Ori, Eds., *Theory and Simulation in Physics for Materials Applications: Cutting-Edge Techniques in Theoretical and Computational Materials Science*, ser. Springer Series in Materials Science. Springer International Publishing, 2020. [Online]. Available: http://link.springer.com/10.1007/978-3-030-37790-8

[2] A. Agrawal and A. Choudhary, "Perspective: Materials informatics and big data: Realization of the "fourth paradigm" of science in materials science," *APL Materials*, vol. 4, no. 5, p. 053208, 2016. [Online]. Available: https://aip.scitation.org/doi/10.1063/1.4946894

[3] J. C. Mauro, "Decoding the glass genome," *Current Opinion in Solid State and Materials Science*, vol. 22, no. 2, pp. 58–64, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1359028617301249

[4] E. O. Pyzer-Knapp, C. Suh, R. Gómez-Bombarelli, J. Aguilera-Iparraguirre, and A. Aspuru-Guzik, "What is high-throughput virtual screening? a perspective from organic materials discovery," *Annual Review of Materials Research*, vol. 45, no. 1, pp. 195–216, 2015. [Online]. Available: https://doi.org/10.1146/annurev-matsci-070214-020823

[5] B. Sanchez-Lengeling and A. Aspuru-Guzik, "Inverse molecular design using machine learning: Generative models for matter engineering," *Science*, vol. 361, no. 6400, pp. 360–365, 2018. [Online]. Available: https://www.sciencemag.org/lookup/doi/10.1126/science.aat2663

[6] T. W. Liao and G. Li, "Metaheuristic-based inverse design of materials – a survey," *Journal of Materiomics*, vol. 6, no. 2, pp. 414–430, 2020. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2352847819302084

[7] J. Noh, G. H. Gu, S. Kim, and Y. Jung, "Machine-enabled inverse design of inorganic solid materials: promises and challenges," *Chemical Science*, vol. 11, no. 19, pp. 4871–4881, 2020. [Online]. Available: http://xlink.rsc.org/?DOI=D0SC00594K

[8] H. Liu, Z. Fu, K. Yang, X. Xu, and M. Bauchy, "Machine learning for glass science and engineering: A review," *Journal of Non-Crystalline Solids: X*, vol. 4, p. 100036, 2019. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2590159119300494

[9] S. Plimpton, "Fast parallel algorithms for short-range molecular dynamics," *Journal of Computational Physics*, vol. 117, no. 1, pp. 1–19, 1995. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S002199918571039X

[10] W. Wang, S. Axelrod, and R. Gómez-Bombarelli, "Differentiable molecular simulations for control and learning," *arXiv:2003.00868 [physics, stat]*, 2020. [Online]. Available: http://arxiv.org/abs/2003.00868

[11] S. S. Schoenholz and E. D. Cubuk, "JAX, m.d.: End-to-end differentiable, hardware accelerated, molecular dynamics in pure python," *arXiv:1912.04232 [cond-mat, physics:physics, stat]*, 2019. [Online]. Available: http://arxiv.org/abs/1912.04232

[12] C.-T. Chen and G. X. Gu, "Generative deep neural networks for inverse materials design using backpropagation and active learning," *Advanced Science*, vol. 7, no. 5, p. 1902607, 2020. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/advs.201902607

[13] Y. Dan, Y. Zhao, X. Li, S. Li, M. Hu, and J. Hu, "Generative adversarial networks (GAN) based efficient sampling of chemical composition space for inverse design of inorganic materials," *npj Computational Materials*, vol. 6, no. 1, pp. 1–7, 2020. [Online]. Available: https://www.nature.com/articles/s41524-020-00352-0

[14] D. P. Kingma and M. Welling, "An introduction to variational autoencoders," *Foundations and Trends® in Machine Learning*, vol. 12, no. 4, pp. 307–392, 2019. [Online]. Available: http://arxiv.org/abs/1906.02691

[15] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in Neural Information Processing Systems 27*, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2014, pp. 2672–2680. [Online]. Available: http://papers.nips.cc/paper/5423-generative-adversarial-nets.pdf

[16] B. Kim, S. Lee, and J. Kim, "Inverse design of porous materials using artificial neural networks," *Science Advances*, vol. 6, no. 1, p. eaax9324, 2020. [Online]. Available: https://advances.sciencemag.org/content/6/1/eaax9324

[17] Z. Liu, D. Zhu, S. P. Rodrigues, K.-T. Lee, and W. Cai, "Generative model for the inverse design of metasurfaces," *Nano Letters*, vol. 18, no. 10, pp. 6570–6576, 2018. [Online]. Available: https://pubs.acs.org/doi/10.1021/acs.nanolett.8b03171

[18] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mane, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viegas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous distributed systems," *arXiv:1603.04467 [cs]*, 2016. [Online]. Available: http://arxiv.org/abs/1603.04467

[19] R. Frostig, M. J. Johnson, and C. Leary, "Compiling machine learning programs via high-level tracing." [Online]. Available: https://www-cs.stanford.edu/~rfrostig/pubs/jax-mlsys2018.pdf

[20] Cloud TPU | google cloud. [Online]. Available: https://cloud.google.com/tpu

[21] Y. E. Wang, G.-Y. Wei, and D. Brooks, "Benchmarking TPU, GPU, and CPU platforms for deep learning," *arXiv:1907.10701 [cs.LG]*, 2019. [Online]. Available: https://arxiv.org/abs/1907.10701v4

[22] T. Lu, Y.-F. Chen, B. Hechtman, T. Wang, and J. Anderson, "Large-scale discrete fourier transform on TPUs," *arXiv:2002.03260 [cs]*, 2020. [Online]. Available: http://arxiv.org/abs/2002.03260

[23] F. Huot, Y.-F. Chen, R. Clapp, C. Boneti, and J. Anderson, "High-resolution imaging on TPUs," *arXiv:1912.08063 [physics]*, 2019. [Online]. Available: http://arxiv.org/abs/1912.08063

[24] K. Yang, Y.-F. Chen, G. Roumpos, C. Colby, and J. Anderson, "High performance monte carlo simulation of ising model on TPU clusters," in *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*. ACM, 2019, pp. 1–15. [Online]. Available: https://dl.acm.org/doi/10.1145/3295500.3356149

[25] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in PyTorch," *31st Conference on Neural Information Processing Systems*, 2017. [Online]. Available: https://openreview.net/pdf?id=BJJsrmfCZ

[26] A. Griewank and A. Walther, *Evaluating Derivatives*. Society for Industrial and Applied Mathematics, 2008. [Online]. Available: https://epubs.siam.org/doi/book/10.1137/1.9780898717761

[27] Y. Hu, L. Anderson, T.-M. Li, Q. Sun, N. Carr, J. Ragan-Kelley, and F. Durand, "DiffTaichi: Differentiable programming for physical simulation," *arXiv:1910.00935 [physics, stat]*, 2020. [Online]. Available: http://arxiv.org/abs/1910.00935

[28] A. Hernández and J. M. Amigó, "Differentiable programming and its applications to dynamical systems," *arXiv:1912.08168 [cs, math]*, 2020. [Online]. Available: http://arxiv.org/abs/1912.08168

[29] F. de Avila Belbute-Peres, K. Smith, K. Allen, J. Tenenbaum, and J. Z. Kolter, "End-to-end differentiable physics for learning and control," in *Advances in Neural Information Processing Systems 31*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Eds. Curran Associates, Inc., 2018, pp. 7178–7189. [Online]. Available: http://papers.nips.cc/paper/7948-end-to-end-differentiable-physics-for-learning-and-control.pdf

[30] Y. Hu, J. Liu, A. Spielberg, J. B. Tenenbaum, W. T. Freeman, J. Wu, D. Rus, and W. Matusik, "ChainQueen: A real-time differentiable physical simulator for soft robotics," in *2019 International Conference on Robotics and Automation (ICRA)*, 2019, pp. 6265–6271.

[31] E. Kierlik, P. A. Monson, M. L. Rosinberg, and G. Tarjus, "Adsorption hysteresis and capillary condensation in disordered porous solids: a density functional study," *Journal of Physics: Condensed Matter*, vol. 14, no. 40, pp. 9295–9315, 2002. [Online]. Available: https://doi.org/10.1088%2F0953-8984%2F14%2F40%2F319

[32] E. Kierlik, M. L. Rosinberg, G. Tarjus, and P. Viot, "Equilibrium and out-of-equilibrium (hysteretic) behavior of fluids in disordered porous materials: Theoretical predictions," *Physical Chemistry Chemical Physics*, vol. 3, no. 7, pp. 1201–1206, 2001. [Online]. Available: https://pubs.rsc.org/en/content/articlelanding/2001/cp/b008636n

[33] G. Sneddon, A. Greenaway, and H. H. P. Yiu, "The potential applications of nanoporous materials for the adsorption, separation, and catalytic conversion of carbon dioxide," *Advanced Energy Materials*, vol. 4, no. 10, p. 1301873, 2014. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/aenm.201301873

[34] K. Sumida, D. L. Rogow, J. A. Mason, T. M. McDonald, E. D. Bloch, Z. R. Herm, T.-H. Bae, and J. R. Long, "Carbon dioxide capture in metal–organic frameworks," *Chemical Reviews*, vol. 112, no. 2, pp. 724–781, 2012. [Online]. Available: http://pubs.acs.org/doi/10.1021/cr2003272

[35] P. G. Boyd, Y. Lee, and B. Smit, "Computational development of the nanoporous materials genome," *Nature Reviews Materials*, vol. 2, no. 8, p. 17037, 2017. [Online]. Available: https://www.nature.com/articles/natrevmats201737

[36] P. Nugent, Y. Belmabkhout, S. D. Burd, A. J. Cairns, R. Luebke, K. Forrest, T. Pham, S. Ma, B. Space, L. Wojtas, M. Eddaoudi, and M. J. Zaworotko, "Porous materials with optimal adsorption thermodynamics and kinetics for CO_2 separation," *Nature*, vol. 495, no. 7439, pp. 80–84, 2013. [Online]. Available: https://www.nature.com/articles/nature11893

[37] E. J. Anglin, L. Cheng, W. R. Freeman, and M. J. Sailor, "Porous silicon in drug delivery devices and materials," *Advanced Drug Delivery Reviews*, vol. 60, no. 11, pp. 1266–1277, 2008. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0169409X08001002

[38] P. Horcajada, T. Chalati, C. Serre, B. Gillet, C. Sebrie, T. Baati, J. F. Eubank, D. Heurtaux, P. Clayette, C. Kreuz, J.-S. Chang, Y. K. Hwang, V. Marsaud, P.-N. Bories, L. Cynober, S. Gil, G. Férey, P. Couvreur, and R. Gref, "Porous metal–organic-framework nanoscale carriers as a potential platform for drug delivery and imaging," *Nature Materials*, vol. 9, no. 2, pp. 172–178, 2010. [Online]. Available: https://www.nature.com/articles/nmat2608