ANP-BBO: Attentive Neural Processes and Batch Bayesian Optimization for Scalable Calibration of Physics-Informed Digital Twins

Ankush Chakrabarty [1] Gordon Wichern [1] Christopher R. Laughman [1]

Abstract

Physics-informed dynamical system models form critical components of digital twins of the built environment. These digital twins enable the design of energy-efficient infrastructure, but must be properly calibrated to accurately reflect system behavior for downstream prediction and analysis. Dynamical system models of modern buildings are typically described by a large number of parameters and incur significant computational expenditure during simulations. To handle large-scale calibration of digital twins without exorbitant simulations, we propose ANP-BBO: a scalable and parallelizable batch-wise Bayesian optimization (BBO) methodology that leverages attentive neural processes (ANPs).

1. Motivation

Buildings account for nearly 40% of global electricity use (over 70% in the U.S.) and at least one third of CO_2 emissions, while space cooling specifically plays a prominent role as it represents more than 70% of peak residential electricity demand to cope with extreme weather. Forecasts indicate that the demand for space cooling will continue rapid growth, with the energy consumed by these applications projected to triple between 2016 and 2050 (Birol, 2018). Current efforts to reduce the climate-related impact of this energy consumption are focused on the creation of grid-interactive buildings, which coordinate the dynamic behavior of buildings with electrical grid behavior that is dominated by time-varying distributed energy resources (Satchwell et al., 2021). As the design and control of these buildings represent a significant change in how buildings are operated, new models that accurately predict their experimentally-observed dynamics, the so-called building 'digital twins', are crucial to developing these next-generation systems.

Building and heating, ventilation, and cooling (HVAC) digital twins need to be calibrated to operational data to accurately replicate the observed behavior of the physical system. Physics-informed dynamical models have a number of advantages in digital twin applications, as they have good predictive/extrapolation properties, their parameters are interpretable by domain experts, and they can be built using information that is measured or archived. Unfortunately, these advantages are often accompanied by nonlinear behavior and numerical stiffness that make simulation sluggish, and the models often comprise translucent/opaque components for privacy or proprietary information security. The ensuing calibration problem therefore tends to be black-box and large, because modern digital twins often contain hundreds or thousands of parameters to be calibrated. Machine learning has been identified as a key technology in optimizing building models (Rolnick et al., 2019).

This calibration problem can be abstracted by considering a predictive simulation model

$$y_{0:T} = \mathcal{M}_T(\theta), \tag{1}$$

where $\theta \in \Theta \subset \mathbb{R}^{n_\theta}$ denotes the constant parameters used to parameterize the building and HVAC dynamics. A search domain of parameters Θ is assumed to be available, and we assume Θ is a box in n_θ-space defined by bounded intervals. The output vector $y_{0:T} \in \mathbb{R}^{n_y \times T}$ denotes the outputs that have been measured using the real building sensors over a time-span $[0, T]$. We do not make any assumptions on the underlying mathematical structure of the model $\mathcal{M}_T(\theta)$, except that it has been designed based on building and HVAC physics, implying that the parameters and outputs are interpretable physical quantities. Simulating $\mathcal{M}_T(\theta)$ forward with a set of parameters $\theta \in \Theta$ yields a vector of outputs $y_{0:T} := \begin{bmatrix} y_0 & y_1 & \cdots & y_t & \cdots & y_T \end{bmatrix}$, with $y_t \in \mathbb{R}^{n_y}$.

Example: Building thermal and refrigerant cycle dynamics are often represented by differential algebraic equations (DAEs) of the form $0 = f_{\mathsf{DAE}}(\dot{x}, x, u, \theta_1)$ and $y = h_{\mathsf{DAE}}(x, u, \theta_2)$. One can model this system using (1) by considering $\theta := \{\theta_1\} \cup \{\theta_2\}$ and simulating (i.e., numerically integrating) the system of DAEs forward over $t \in [0, T]$ to generate the sequence of outputs $y_{0:T}$.

[1] Mitsubishi Electric Research Laboratories (MERL), Cambridge, MA, USA.. Correspondence to: A. Chakrabarty <achakrabarty@ieee.org>.

Tackling Climate Change with Machine Learning Workshop at ICML 2021.

The calibration task is to estimate a parameter set $\theta^\star \in \Theta$ that minimizes (in some goodness-of-fit sense) the modeling error $y^\star_{0:T} - \mathcal{M}_T(\theta^\star)$, where $y^\star_{0:T}$ denotes the measured outputs collected from a real system, and $\mathcal{M}_T(\theta^\star)$ denotes the estimated outputs from the model $\mathcal{M}_T(\theta)$ using the estimated parameters $\theta^\star$. To this end, we propose optimizing a calibration cost function $J(y^\star_{0:T}, \mathcal{M}_T(\theta))$ to obtain the optimal parameters

$$\theta^\star = \arg\min_{\theta\in\Theta} J(y^\star_{0:T}, \mathcal{M}_T(\theta)). \tag{2}$$

Since each simulation is expensive and the underlying structure of the calibration model and cost are unknown, we solve the problem (2) using Bayesian optimization (BO), which has shown potential for global optimization of black-box functions in a sample-efficient manner (Snoek et al., 2012). BO requires designing two components: a probabilistic map from the decision variables θ to the cost J, and an acquisition function that guides the selection of the next best optimizer candidate given the available data points. Classically, BO methods leverage Gaussian process (GP) regression for the task of providing a probabilistic map, but it is well known that GPs scale cubically with the number of available data points and the dimension of Θ (Snoek et al., 2015). Since we do not pose restrictions on J, it is possible that solving (2) for large n_θ can require thousands of data points to compute near-optimal solutions. This poses three critical challenges for classical BO methods: (C1) GP regression requires prohibitive training times with thousands of data points in high-dimensional spaces and are therefore not well-suited for calibrating large digital twins of modern buildings, (C2) the GP-approximated cost function is strongly dependent on the kernel selected by the user, and such kernels may induce functional properties like smoothness that are not always seen in practice; and, (C3) evaluating the cost function every time a new candidate parameter is computed is not amenable to parallelization.

In lieu of GPs, we propose using attentive neural processes (ANPs) to approximate the calibration cost. ANPs are deep neural networks that are capable of learning a broad class of stochastic processes, and therefore, can make predictions equipped with uncertainty quantification (Garnelo et al., 2018). ANPs are highly scalable and suitable for training on high-dimensional problems with large datasets, and can perform well without requiring careful kernel selection. Consequently, we posit that replacing GPs with ANPs solves the challenges (C1) and (C2). Another benefit of ANPs is that they incur less computational complexity during inference than GPs. This fact, coupled with the observation that re-training an ANP with single datapoint increments seems wasteful as the inference of a deep neural network is unlikely to change significantly with one additional point, suggests the utility of batch BO (BBO) methods. Unlike BO, BBO acquisition functions generate a batch of candidates that are to be evaluated. Thus, the time-consuming cost function evaluation can be parallelized and the ANP updated with a batch of data points: this provides a way to address (C3). Note that due to multi-scale dynamics and combination of PDEs, DAEs, etc. in digital twins of buildings, simulating the twin can require orders of magnitude more time than retraining ANPs; especially simulations with large T.

2. Relevant Work

State-of-the-art methods for calibration of thermal models are presented in the survey by Wang & Chen (2019): these models often do not consider equipment dynamics and parameters are not easily interpreted. Conversely, the study by Drgoňa et al. (2021) shows the benefits of physics-informed models. However, the associated increase of digital twin model complexity requires scalable and sample-efficient optimization algorithms like BO. Scalable BO methods typically fall into two classes, those based on low dimensional embeddings (Wang et al., 2016; Nayebi et al., 2019; Lu et al., 2018), or those based on alternate probabilistic regressors that scale well with dimensions and number of data points, such as kernel methods (Kandasamy et al., 2015; Oh et al., 2018), or deep Bayesian networks (Snoek et al., 2015; Springenberg et al., 2016). Very recently, a neural process (without attention) has been considered as a surrogate for BO (Shangguan et al., 2021). However, there is clear empirical evidence that ANPs produce 'tight' predictions at data points where the objective is known; this is an essential property needed for BO and the lack of this property is a drawback of the NP. Furthermore, their implementation involves training the NP once with a large amount of initial data, which differs from our approach of batch BO to avoid re-training too often. Recent work on BBO has resulted in powerful algorithms for selecting batches based on the GP posterior (Azimi et al., 2012; Desautels et al., 2014), or by penalizing to find disjoint regions likely to contain extrema (González et al., 2016; Nguyen et al., 2016). Other BBO methods use hallucinations from the GP posterior either by Thompson sampling (De Palma et al., 2019) or multi-scale sampling of GP hyperparameters (Joy et al., 2020). In our proposed ANP-BBO, we combine the benefits of penalization and hallucination during ANP inference.

3. The ANP-BBO Algorithm

Fig. 1 illustrates the ANP-BBO workflow for solving the optimization problem from (2) in an iterative manner. Let $\mathcal{D}^t := \{(\theta_i, J_i)\}_{i=0}^{N_0+tN}$ denote the data (parameter/cost pairs) collected up to the t-th iteration of ANP-BBO, where N_0 is the size of an initial dataset and N is the batch-size, i.e., the number of model simulations/cost evaluations performed at each iteration. In this work, we use the ANP (Kim et al., 2019) to estimate the conditional Gaussian distribu-

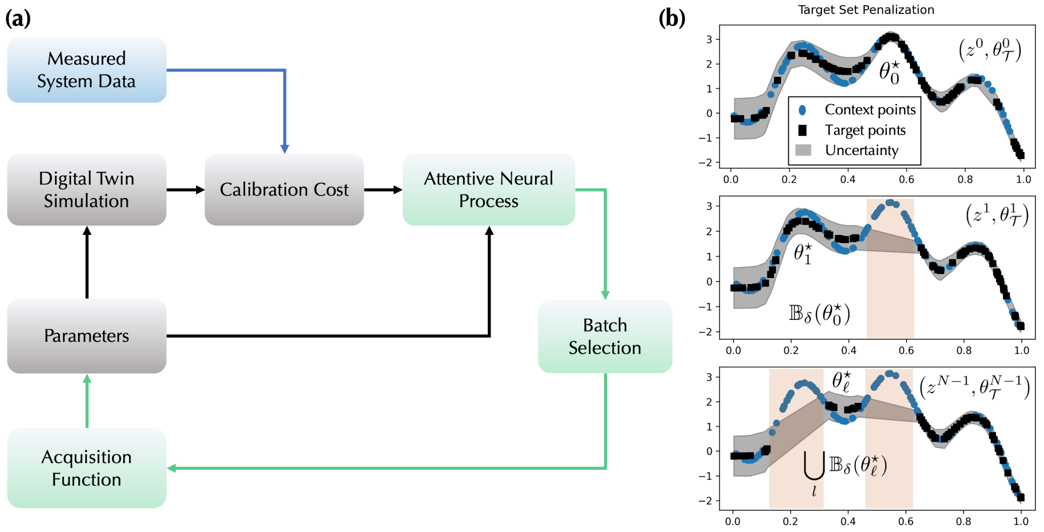

Figure 1. (a) Calibration workflow with ANP-BBO. (b) Illustration of target penalization and latent sampling. More details about subplot (b) are provided in Appendix B.

tion $p(J_\mathcal{T}|\theta_\mathcal{T}, \mathcal{D}^t, z)$, where $J_\mathcal{T}$ is a set of cost function values located at target points $\theta_\mathcal{T} \subset \Theta$, and z is a latent variable that can be sampled to obtain different realizations of the learned stochastic process. A summary of relevant details of the ANP are provided in Appendix A.

Directly substituting a GP with an ANP for BO with $N = 1$ would involve: (i) training the ANP with $\mathcal{D}^t$ at every t, (ii) sampling target points $\theta_\mathcal{T} \subset \Theta$, (iii) obtaining one sample of the latent z, predict mean and variance of $J_\mathcal{T}$ for the target set, and evaluate an acquisition function at those points, (iv) selecting the target that maximizes the acquisition function as the best candidate $\theta^{t,\star}$, and (v) evaluating the cost for $\theta^{t,\star}$ and append this pair to $\mathcal{D}^t$, retrain the ANP, and repeat from (i) for iteration $t + 1$. While not retraining the ANP is an option, this requires N_0 to be extremely large and the ANP trained with N_0 data points equipped with weights that reflect the underlying function closely. In our work, we follow the spirit of classical BO and assume N_0 is small. For this, the ANP needs retraining with a growing dataset. However, we posit that $N > 1$ offers the significant advantages of reducing the number of times the ANP is retrained while enabling parallelized simulations of digital twins during evaluation of the calibration cost.

To this end, we propose the following modifications to the workflow (i)–(v) above; see Fig. 1 for an illustration using a 1-D exemplar function. In particular, we loop through steps (ii)–(iv) N times, and at each iteration $k = 0, \cdots, N-1$, we perform target penalization by selecting a target set $\theta^k_\mathcal{T}$ away from neighborhoods of previous candidates. We explain the target penalization step formally as follows. Let $\mathbb{B}_\delta(\theta)$ denote a ball of radius δ centered at $\theta \in \Theta$. At batch-selection iteration $k = 0$, the target set is constructed by extracting samples from the entirety of Θ. At each subsequent iteration, neighborhoods of Θ are removed to ensure diversity of solutions. Concretely, at the k-th iteration, if $\theta^{t,\star}_{0:k-1}$ denotes the set of candidates selected so far, then the target set $\theta^k_\mathcal{T}$ will be sampled from $\Theta \setminus \left(\bigcup_{\ell=0}^{k-1} \mathbb{B}_\delta\left(\theta^{t,\star}_\ell\right)\right)$. This method is similar to the local penalization approach of (González et al., 2016), but we do not assume knowledge of Lipschitz constants of the cost. Our target penalization method rejects samples from the target set to ensure that candidates in the batch do not cluster around a suspected local minimum. To prevent conglomeration of candidates, we select δ large enough to maintain distance between candidates in the batch, while being small enough to ensure that fewer than N balls cannot cover Θ. Additionally, we utilize the ANPs ability to model families of distributions by sampling the latent variable iteratively during batch-selection. Sampling z fixes a distribution from the family of distributions; thus, resampling z during batch-selection promotes diversity in the statistics of the predicted output. At each k, the target sample that maximizes a given acquisition function is added to the batch. The target penalization and latent sampling can be highly parallelized for efficient cost function evaluation.

4. Results and Discussion

Setup: Details about the physics-informed building digital twin are presented in Appendix C. We obtain the data $y^\star_{0:T}$ by simulating the building dynamics for 5 days and collecting temperature and humidity measurements for 3 rooms (thus, $n_y = 6$) every 15 minutes. The $n_\theta = 12$ true parameters are provided in Table 1. The measurements are corrupted by Gaussian noise of zero mean and 0.5 variance (for temperature) and 4 variance (for humidity); additionally, the sensors are assumed to be quantized at 0.1 resolution. The first 2 days' data is used to train the ANP and perform calibration. The final 3 days are used for testing; that is, the calibrated digital twin predicts the outputs for the final 3 days for comparison with true outputs.

Calibration performance: Details about the ANP-BBO implementation are provided in Appendix D. Fig. 2 illustrates the outputs over both training and testing data (5 days) after 1000 objective function evaluations. Clearly, the continuous lines (which are the digital twin predictions with the best parameters found by ANP-BBO) fit the data (colored circles) well, despite noise in measurements, for both temperature and humidity outputs. In fact, the coefficient-of-variation root-mean-squared error (CVRMSE: $\|\varepsilon_i\|/\sqrt{T}$) of all of our outputs are within 1%, which is far below the ASHRAE guideline of 15% (ASHRAE, 2014). Furthermore, we are encouraged to see that despite considering a search space of significant volume (typically, calibration problems assume search space $\pm 20\%$ of the nominal parameter value), the final set of parameters are close to the true values (see Table 1). Some estimates are better than others due to inherent sensitivities of the parameters, but most parameters are captured to over 90% relative accuracy.

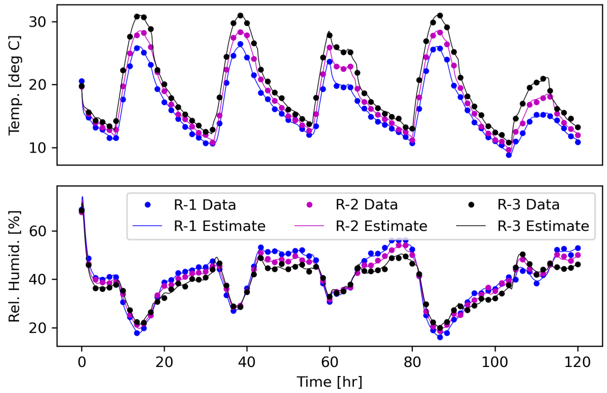

Figure 2. Test data and digital twin estimates for 5 days using the best set of parameters obtained by ANP-BBO. (R-x: Room x.)

θ_i	True	Best	Θ_i	θ_i	True	Best	Θ_i
θ_1	8.00	8.04	[6, 10]	θ_2	5.00	5.09	[3, 7]
θ_3	0.45	0.35	[0, 1]	θ_4	3.00	2.88	[2, 4]
θ_5	1.00	1.31	[0, 2]	θ_6	0.10	0.11	[0, 1]
θ_7	1.00	0.97	[0, 1]	θ_8	0.10	0.07	[0, 1]
θ_9	18.00	18.35	[14, 20]	θ_{10}	10.00	10.32	[8, 11]
θ_{11}	0.48	0.53	[0,2]	θ_{12}	6.00	6.07	[3, 7]

Table 1. Parameter estimates and corresponding search spaces.

Ablation study: We perform additional testing of ANP-BBO with the following modifications: (i) we switch off target set penalization and rely only on latent sampling for batch-selection (ANP-NoTarPen), (ii) we train the ANP once on the initial dataset and do not perform retraining (ANP-NoRetrain) as in Shangguan et al. (2021), (iii) we perform BO with sparse Gaussian processes (Titsias, 2009) with 100 inducing points and 1000 function evaluations (same as ANP-BBO) to prevent prohibitive training times (SGP-VFE-100), and (iv) same as (iii) but with 500 inducing points (SGP-VFE-500). The results of this study are encapsulated in Fig. 3, where we see that ANP-BBO outperforms its competitors, with SGP-VFE-500 showing fast decay but lack of improvement owing to subsequent BO candidates clustering around similar subregions of Θ. The benefit of target penalization is also evident, as we see ANP-BBO's cost decays faster and more consistently than ANP-NoTarPen owing to the exploratory aspect introduced by the diversity amongst predictions induced by target penalization. We observe that ANP-NoRetrain performs poorly, which is expected since lack of retraining implies that the attention weights are not recomputed, and therefore new context points which may contain critical information to the optimization problem is largely underutilized.

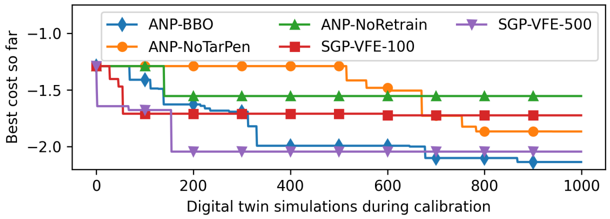

Figure 3. Ablation study results. Comparison of incumbent cost with number of function evaluations (i.e. number of simulations).

To justify that ANP retraining is often faster than N digital twin simulations, we refer the reader to Appendix E, where we have compared training and inference times of ANP and exact GP for a large number of datapoints in 12-D parameter space, as in our building calibration task. We also demonstrated that a week-long simulation of modern buildings that also accurately model stiff HVAC dynamics is comparable to ANP retraining times. Thus, we posit that longer horizon (month/year-long) or larger-scale (cities) digital twin simulations will incur over 10× wall-time than ANP retraining.

5. Conclusions

We proposed an ANP-BBO methodology that harnesses the power of probabilistic deep learning to calibrate industrial digital twins due to the presence of unmodeled dynamics and opacity incorporated to protect privacy, trade secrets, etc. Precisely calibrating digital twins enables monitoring, control, self-optimization, and other key technologies that are strongly coupled with sustainability, air quality control, leakage detection, etc. Thus, *accurate and scalable calibration mechanisms are essential to tackling climate change.*

References

ASHRAE. Guideline 14-2014, measurement of energy, demand, and water savings. *American Society of Heating, Refrigerating, and Air Conditioning Engineers, Atlanta, Georgia*, 2014.

Azimi, J., Jalali, A., and Fern, X. Z. Hybrid batch Bayesian optimization. In *Proceedings of the 29th International Conference on International Conference on Machine Learning*, pp. 315–322, 2012.

Bauer, P., Dueben, P. D., Hoefler, T., Quintino, T., Schulthess, T. C., and Wedi, N. P. The digital revolution of Earth-system science. *Nature Computational Science*, 1(2):104–113, 2021.

Birol, F. The future of cooling. Technical report, International Energy Agency, 2018.

De Palma, A., Mendler-Dünner, C., Parnell, T., Anghel, A., and Pozidis, H. Sampling acquisition functions for batch Bayesian optimization. *arXiv preprint arXiv:1903.09434*, 2019.

Desautels, T., Krause, A., and Burdick, J. W. Parallelizing exploration-exploitation tradeoffs in Gaussian process bandit optimization. *Journal of Machine Learning Research*, 15:3873–3923, 2014.

Drgoňa, J., Tuor, A. R., Chandan, V., and Vrabie, D. L. Physics-constrained deep learning of multi-zone building thermal dynamics. *Energy and Buildings*, 243:110992, 2021.

Gardner, J. R., Pleiss, G., Bindel, D., Weinberger, K. Q., and Wilson, A. G. GPyTorch: Black-box Matrix-Matrix Gaussian Process Inference with GPU Acceleration. In *Proc. NeurIPS*, pp. 7587–7597, 2018.

Garnelo, M., Schwarz, J., Rosenbaum, D., Viola, F., Rezende, D. J., Eslami, S., and Teh, Y. W. Neural processes. *arXiv preprint arXiv:1807.01622*, 2018.

González, J., Dai, Z., Hennig, P., and Lawrence, N. Batch Bayesian optimization via local penalization. In *Artificial intelligence and statistics*, pp. 648–657. PMLR, 2016.

Joy, T. T., Rana, S., Gupta, S., and Venkatesh, S. Batch Bayesian optimization using multi-scale search. *Knowledge-Based Systems*, 187:104818, 2020.

Kandasamy, K., Schneider, J., and Póczos, B. High dimensional Bayesian optimisation and bandits via additive models. In *International conference on machine learning*, pp. 295–304. PMLR, 2015.

Kim, H., Mnih, A., Schwarz, J., Garnelo, M., Eslami, A., Rosenbaum, D., Vinyals, O., and Teh, Y. W. Attentive neural processes. In *Proc. ICLR*, May 2019.

Kingma, D. P., Salimans, T., and Welling, M. Variational dropout and the local reparameterization trick. *arXiv preprint arXiv:1506.02557*, 2015.

Le, T. A., Kim, H., Garnelo, M., Rosenbaum, D., Schwarz, J., and Teh, Y. W. Empirical evaluation of neural process objectives. In *NeurIPS Workshop on Bayesian Deep Learning*, December 2018.

Lu, X., Gonzalez, J., Dai, Z., and Lawrence, N. D. Structured variationally auto-encoded optimization. In *International conference on machine learning*, pp. 3267–3275. PMLR, 2018.

Nayebi, A., Munteanu, A., and Poloczek, M. A framework for Bayesian optimization in embedded subspaces. In *International Conference on Machine Learning*, pp. 4752–4761. PMLR, 2019.

Nguyen, V., Rana, S., Gupta, S. K., Li, C., and Venkatesh, S. Budgeted batch Bayesian optimization. In *2016 IEEE 16th International Conference on Data Mining (ICDM)*, pp. 1107–1112. IEEE, 2016.

Oh, C., Gavves, E., and Welling, M. BOCK: Bayesian optimization with cylindrical kernels. In *International Conference on Machine Learning*, pp. 3868–3877. PMLR, 2018.

Paszke, A. et al. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R. (eds.), *Proc. NeurIPS*, pp. 8024–8035. 2019.

Rolnick, D., Donti, P. L., Kaack, L. H., Kochanski, K., Lacoste, A., Sankaran, K., Ross, A. S., Milojevic-Dupont, N., Jaques, N., Waldman-Brown, A., et al. Tackling climate change with machine learning. *arXiv preprint arXiv:1906.05433*, 2019.

Satchwell, P., Piette, M., Khandekar, A., Granderson, J., Frick, N., Hledik, R., Faruqui, A., Lam, L., Ross, S., Cohen, J., Wang, K., Urigwe, D., Delurey, D., Neukomm, M., and Nemtzow, D. A national roadmap for grid-efficient buildings. Technical report, U.S. Department of Energy, 2021.

Shangguan, Z., Lin, L., Wu, W., and Xu, B. Neural process for black-box model optimization under Bayesian framework. *arXiv preprint arXiv:2104.02487*, 2021.

Snoek, J., Larochelle, H., and Adams, R. P. Practical Bayesian optimization of machine learning algorithms. In *Proc. NeurIPS*, pp. 2951–2959, 2012.

Snoek, J., Rippel, O., Swersky, K., Kiros, R., Satish, N., Sundaram, N., Patwary, M., Prabhat, M., and Adams, R.

Scalable Bayesian optimization using deep neural networks. In *International conference on machine learning*, pp. 2171–2180. PMLR, 2015.

Springenberg, J. T., Klein, A., Falkner, S., and Hutter, F. Bayesian optimization with robust Bayesian neural networks. *Advances in NeurIPS*, 29:4134–4142, 2016.

Titsias, M. Variational learning of inducing variables in sparse Gaussian processes. In *Artificial intelligence and statistics*, pp. 567–574. PMLR, 2009.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. Attention is all you need. In *Proc. NeurIPS*, 2017.

Wang, Z. and Chen, Y. Data-driven modeling of building thermal dynamics: Methodology and state of the art. *Energy and Buildings*, 203:109405, 2019.

Wang, Z., Hutter, F., Zoghi, M., Matheson, D., and de Feitas, N. Bayesian optimization in a billion dimensions via random embeddings. *Journal of Artificial Intelligence Research*, 55:361–387, 2016.

Wetter, M., Zuo, W., Nouidui, T. S., and Pang, X. Modelica Buildings Library. *Journal of Building Performance Simulation*, 7(4):253–270, 2014.

Appendix

A. Architecture and Description of Attentive Neural Processes (ANP)

In the context of Bayesian optimization for digital twin calibration, the ANP (Kim et al., 2019) is a regressor that defines stochastic processes with digital twin parameters serving as inputs $\theta_i \in \mathbb{R}^{n_\theta}$, and function evaluations serving as outputs $J_i \in \mathbb{R}$. Given a dataset $\mathcal{D} = \{(\theta_i, J_i)\}$, we learn an ANP for a set of $n_\mathcal{T}$ target points $\mathcal{D}_\mathcal{T} \subset \mathcal{D}$ conditioned on a set of $n_\mathcal{C}$ observed context points $\mathcal{D}_\mathcal{C} \subset \mathcal{D}$. The ANP is invariant to the ordering of points in $\mathcal{D}_\mathcal{T}$ and $\mathcal{D}_\mathcal{C}$; furthermore, the context and target sets are not necessarily disjoint. The ANP additionally contains a global latent variable z with prior $q(z|\mathcal{D}_\mathcal{C})$ that generates different stochastic process realizations , thereby incorporating uncertainty into the predictions of target function values $J_\mathcal{T}$ despite being provided a fixed context set.

Concretely, given a context set $\mathcal{D}_\mathcal{C}$ and target query points $\theta_\mathcal{T}$, the ANP estimates the conditional distribution of the target values $J_\mathcal{T}$ given by $p(J_\mathcal{T}|\theta_\mathcal{T}, \mathcal{D}_\mathcal{C}) := \int p(J_\mathcal{T}|\theta_\mathcal{T}, r_\mathcal{C}, z)\, q(z|s_\mathcal{C})\, \mathrm{d}z$, where $r_\mathcal{C} := r(\mathcal{D}_\mathcal{C})$ is the output of the transformation induced by the *deterministic* path of the ANP, obtained by aggregating the context set into a finite-dimensional representation that is invariant to the ordering of context set points (*e.g.*, passing through a neural network and taking the mean). The function $s_\mathcal{C} := s(\mathcal{D}_\mathcal{C})$ is a similar permutation-invariant transformation made via a *latent* path of the ANP. Both the transformations r in the deterministic path and s in the latent path are evaluated using self-attention networks (Vaswani et al., 2017) with neural weights $\omega_r \neq \omega_s$ before aggregation. The aggregation operator in the latent path is typically the mean, whereas for the deterministic path, the ANP aggregates using a cross-attention mechanism, where each target query attends to the context points $\theta_\mathcal{C}$ to generate $r_{\mathcal{C}\times\mathcal{T}}(J_\mathcal{T}|\theta_\mathcal{T}, r_\mathcal{C}, z)$. Note that the ANP builds on the variational autoencoder (VAE) architecture, wherein $q(z|s)$, $r_\mathcal{C}$, and $s_\mathcal{C}$ form the encoder arm, and $p(J|\theta, r_{\mathcal{C}\times\mathcal{T}}, z)$ forms the decoder arm. The architecture of ANP with both paths is provided in Appendix-Fig. 4.

For implementation, we make simplifying assumptions: (1) that each point in the target set is derived from conditionally independent Gaussian distributions, and (2) that the latent distribution is a multivariate Gaussian with a diagonal covariance matrix. This enables the use of the reparametrization trick (Kingma et al., 2015) and train the ANP to maximize the evidence-lower bound loss $\mathsf{E}\big[\log p(J_\mathcal{T}|\theta_\mathcal{T}, r_{\mathcal{C}\times\mathcal{T}}, z)\big] - \mathsf{KL}\left[q(z|s_\mathcal{T})||q(z|s_\mathcal{C})\right]$ for randomly selected $\mathcal{D}_\mathcal{C}$ and $\mathcal{D}_\mathcal{T}$ within $\mathcal{D}$. Maximizing the expectation term $\mathsf{E}(\cdot)$ ensured good fitting properties of the ANP to the given data, while minimizing (maximizing the negative of) the KL divergence embeds the intuition that the targets and contexts arise from the same family of stochastic processes. The complexity of ANP with both self-attention and cross-attention is $\mathbf{O}\left(n_\mathcal{C}(n_\mathcal{C} + n_\mathcal{T})\right)$. Empirically, we observed that only using cross-attention does not deteriorate performance while resulting in a reduced complexity of approximately $\mathbf{O}\left(n_\mathcal{C} n_\mathcal{T}\right)$, which is beneficial because $n_\mathcal{T}$ is fixed, but $n_\mathcal{C}$ grows with BO iterations.

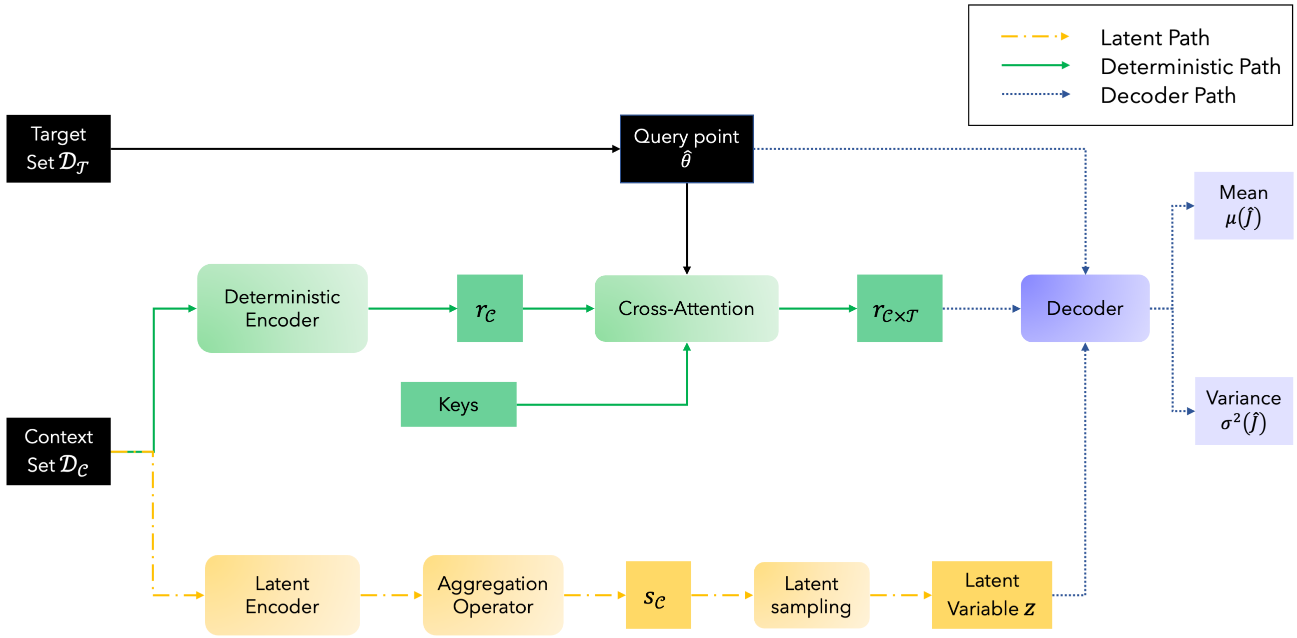

Figure 4. ANP Architecture: Paths for training (latent, deterministic) and decoder.

B. More details about Fig. 1 subplot (b):

The 1-D function used for illustration is given by $J(\theta) := \sin(20\theta) + \left(\frac{10\theta}{3}\right)^2 - 10\theta$ which has its global maximizer at $\theta^\star = 0.5445$ on the search space $\Theta = [0, 1]$. The ANP for this example has been trained using $N_0 = 100$ initial samples randomly extracted from Θ. For each subplot in (b), the blue circles denote context points (these are the same for all subplots), the black squares are target points (these are penalized at subsequent batch-selection iterations after the first), the black shading denotes $\mu \pm 1.96\sigma$ around the ANP predictions at *target* points, and the orange vertical shades depict subsets of Θ that are penalized (this is why there are no target points there). The top-most subplot of (b), with $k = 0$, uses all target points to make a selection $\theta_0^\star = 0.53$ for the batch. Since $\delta = 0.1$, in the next batch-selection iteration $k = 1$, there are no target points in $\mathbb{B}_\delta(\theta_0^\star)$, which is why $\theta_1^\star = 0.21$. The process is repeated. The effect of latent sampling is visible most clearly for $\theta \approx 0.8$, since the uncertainty bands there have larger lobes; empirically we have observed that this has a strong effect on the prediction uncertainties when only a few true data points have been obtained, which is the case in initial BO iterations.

C. Description of Building Digital Twin

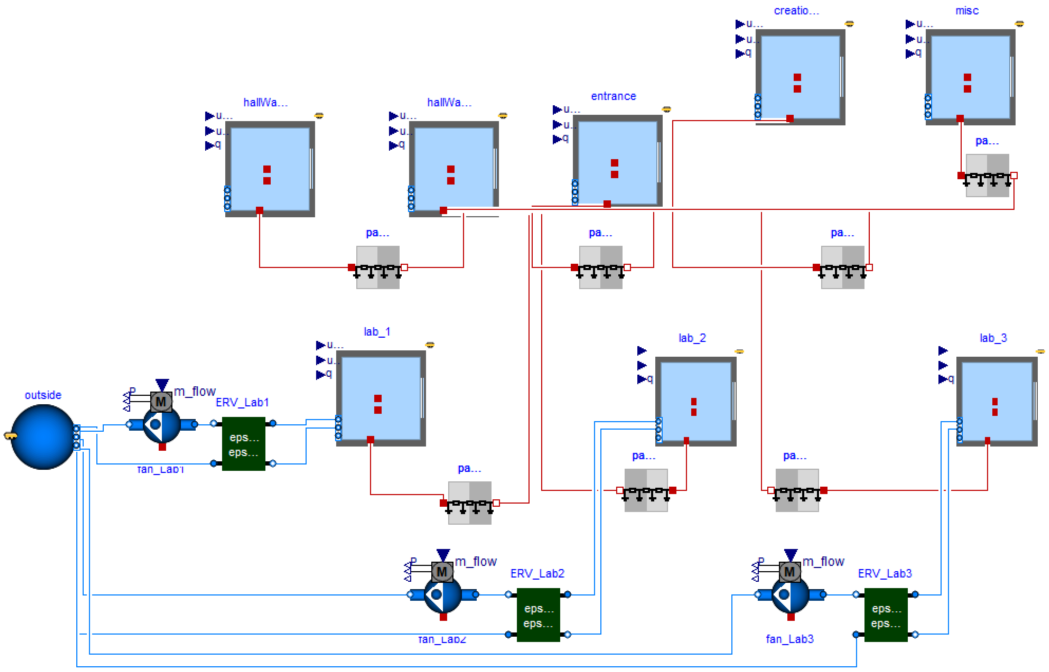

Figure 5. Building Plenum Component of Digital Twin: Modelica Implementation.

A model of one floor of a contemporary four story office building, located in the Tokyo, Japan geographic area, was used for this study. We examine the behavior of three laboratory spaces (rooms) in this building, labeled lab_1, lab_2, and lab_3 in Figure 5, over the course of 5 days of simulated behavior. Each of these laboratory rooms has a floor area of 172.8 m^2 and is 3.1 m tall, with one exterior facade that includes a single-pane 23 m^2 window. Conventional building materials and construction practices were used for the envelope, with an adiabatic lower boundary and a 1.15 m plenum above the laboratories with an adiabatic upper boundary above the plenum. Other adjacent spaces were also included in the model, though the thermal interactions between the laboratories under study and these spaces were limited.

This building model also included a ventilation system to supply fresh outside air for each of the laboratories. These consisted of a fan providing inlet air at a flow rate of 2118 cfm, or approximately 0.5 air changes/hour. This ventilation air was processed with an energy recovery ventilator (ERV) with a constant efficiency of 0.8 and pressure drop of 200 Pa to exchange thermal energy between the supply and exhaust air streams.

Different occupancy schedules and loads were imposed on each of the rooms to explore the effect of different dynamics on the calibration process. Laboratory 1 was occupied between the hours of 5am and 2pm, with a base convective/radiant load of 5 $\mathrm{W/m^2}$ and an occupied load of 14 $\mathrm{W/m^2}$, as well as 3 $\mathrm{W/m^2}$ of latent load during the occupied hours. Laboratory 2 had a much lower overall load and was occupied between the hours of 8:30 am and 6pm, with a base convective/radiant load of 0.1 $\mathrm{W/m^2}$ and an occupied load of 1.1 $\mathrm{W/m^2}$, as well as 0.1 $\mathrm{W/m^2}$ of latent load during the occupied hours. Finally, laboratory 3 had the highest overall load and was occupied between the hours of 3pm and 12am, with a base convective/radiant load of 10 $\mathrm{W/m^2}$ and an occupied load of 28 $\mathrm{W/m^2}$, as well as 6 $\mathrm{W/m^2}$ of latent load during the occupied hours.

This model was constructed using the Modelica Buildings library (Wetter et al., 2014), an open source library developed primarily by the Lawrence Berkeley National Laboratory to characterize a wide variety of components used in today's building systems. These models characterize the convective, radiative, and latent heat transfer observed in occupied spaces, and employ an ideal gas moist air mixture model. The Tokyo-Hyakuri TMY3 model was used to describe the weather conditions and solar heat gains between November 23-28 that were used as the subject of study, which is calibrated on recent climate data.

D. Implementation Details

Codebase

The ANP pipeline is implemented entirely in PyTorch (Paszke et al., 2019).

Comparisons with GP are performed using GPyTorch (Gardner et al., 2018).

ANP Implementation

For the ANP model the architecture we follow the basic version from (Kim et al., 2019), without self attention. We use latent dimension of 128 for both the deterministic and latent paths. The deterministic encoder uses 3 fully-connected layers with Leaky ReLU (slope=0.1) activations and 256 hidden units per layer. The latent encoder uses a similar fully-connected architecture, however after the mean aggregation operation, we pass it through two fully connected layers, the first with linear activation functions, to obtain the latent mean $\mu(z)$, and the second with an activation of $0.1 + 0.9 \cdot \mathrm{sigmoid}(\mathrm{x})$ to obtain the latent standard deviation $\sigma(z)$. For the cross attention block in the deterministic encoder, we first run the query and key coordinate positions through a fully-connected layer of size equal to the latent dimension to obtain learned positional encodings prior to the 8-head multi-head attention operation (Vaswani et al., 2017). Finally the decoder which takes as input the concatenation of the sampled latent z, target position θ_{t_i}, and the deterministic path target encoding r_{t_i} consists of 3 fully-connected layers with Leaky ReLU (slope=0.1) activations and 256 hidden units per layer. Finally, following the best practices in (Le et al., 2018), the output layer in the decoder has two output units, the first with a linear activation function for estimating $\mu(J)$, and the second for estimating $\sigma(J)$ with a regularized softplus activation to avoid the standard deviation collapsing to zero, i.e., $0.1 + 0.9 \cdot \mathrm{softplus}(\mathrm{x})$.

We train the initial ANP for 5000 iterations (this is done offline with the N_0 initial data points) with a learning rate of 10^{-5}, which is decreased by a factor of 2, after 1000 steps, and decreased by an additional factor of 5 after 2500 steps. We train the ANP at each BBO iteration t, using the ADAM optimizer with an initial learning rate of 5×10^{-5}, which is decreased by a factor of 2, after 250 steps, and decreased by an additional factor of 5 after 500 steps. Each step consists of a mini-batch of 32 randomly selected context ($\mathcal{D}_\mathcal{C}$) and target sets ($\mathcal{D}_\mathcal{T}$) within $\mathcal{D}$. For each element of the batch, we select points for $\mathcal{D}_\mathcal{C}$ and $\mathcal{D}_\mathcal{T}$ uniformly at random from all points in $\mathcal{D}$. For all, BBO iterations after the initial one, we warm start the model using the weights from the previous iteration.

ANP-BBO Implementation

We start with an initial set of $N_0 = 1000$ data points, where θ is sampled on Θ (provided in Table 1) via Sobol sequences. The ANP is trained offline on this data, and the weights are stored for warm-starting subsequent retrains. With $\varepsilon(T) := (y^\star_{0:T} - \mathcal{M}_T(\theta))$, we select the cost

$$J(\theta) = \log\left(\sum_{i=1}^{n_y} \varepsilon_i(T)^\top W_i \varepsilon_i(T)\right),$$

where the logarithm helps with numerical conditioning and W is chosen to scale the outputs to similar magnitudes. Note that we transform the minimization problem (2) to a maximization problem, as in classical BO, by reversing the sign of J. We then perform ANP-BBO for $N = 200$ iterations with a batch-size of $K = 5$ per iteration, an upper-confidence bound acquisition function $\mu + 3\sigma$, 5000 target points for sampling to obtain acquisition function maxima. For target penalization, we set $\delta = 0.01$.

E. Comparison of Wall-times

This comparison study was performed on a Windows 10 desktop with 32-GB RAM, Intel(R) Core(TM) i9-9900K CPU @ 3.60GHz. No GPU acceleration was used for training either the ANP or GP methods. All training points are 12-dimensional vectors in accordance with the building digital twin parameters.

While ANP training times can be large, we demonstrate that a digital twin simulation can be significantly larger depending on the time-span of simulation. In Fig. 6, we compare the wall-time incurred on the same computer for 1000 training iterations for ANP with 2000 data points, ANP with 10,000 data points, GP with 2000 data points, GP with 4000 data points (after which GP becomes prohibitively slow). We also compare wall-times for inference of the same number of data points with ANP and GP. Finally, we present simulation times for 1 week using two digital twins: one with simple building dynamics and the cooling/heating system replaced by a lookup table, and another where the HVAC dynamical equations are also incorporated in the twin. Note that these twins could be simulated for month-long (or year-long) time-spans, which would require over $4\times$ (or $52\times$) these wall-times. Digital twins for cities and climate models would require considerably more wall-time to simulate, even with GPU integration; in fact, these Earth-scale simulations are often estimated to require > 5000 GPUs (Bauer et al., 2021) or supercomputing.

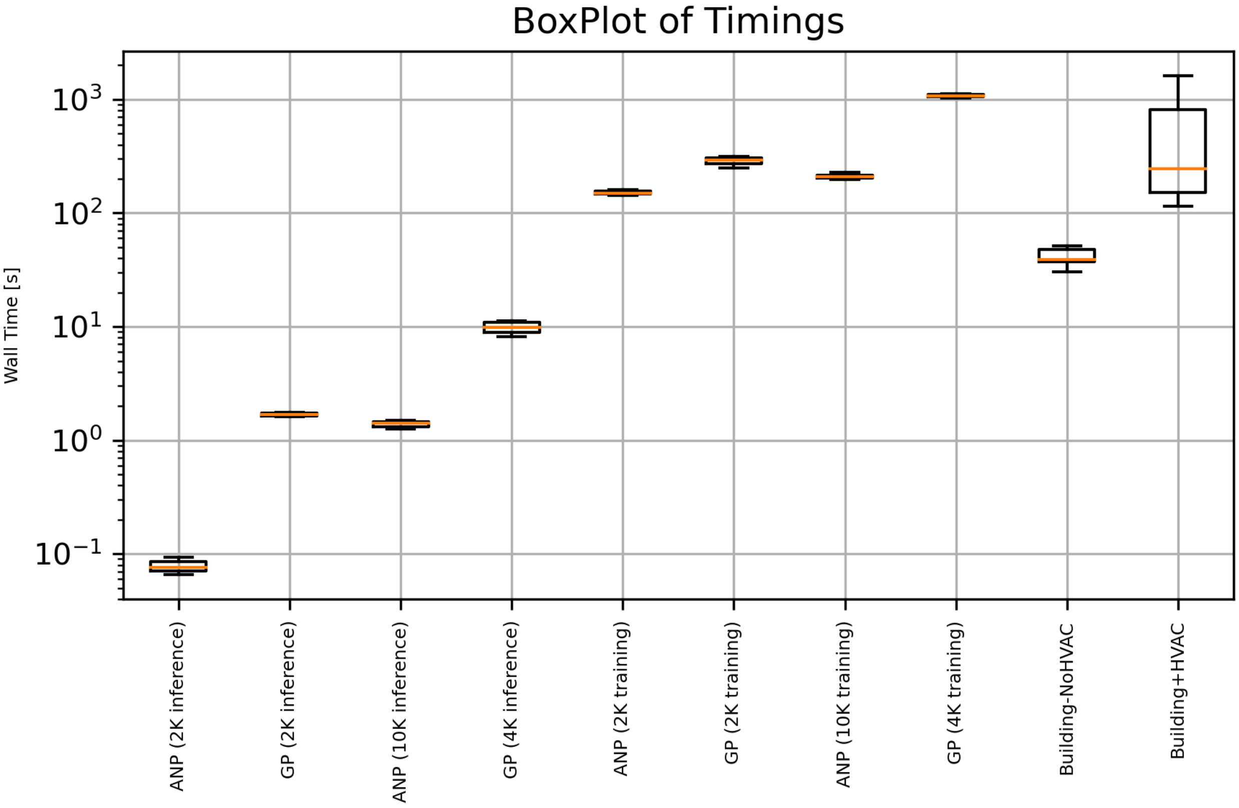

Figure 6. Wall times required for training, inference, and simulation of digital twin with and without HVAC equipment dynamics.