Extrapolating Quantum Observables with Machine Learning: Inferring Multiple Phase Transitions from Properties of a Single Phase

Rodrigo A. Vargas-Hernández,[1] John Sous,[1,2,3] Mona Berciu,[2,3] and Roman V. Krems[1]
[1]Department of Chemistry, University of British Columbia, Vancouver, British Columbia, Canada V6T 1Z1
[2]Department of Physics and Astronomy, University of British Columbia, Vancouver, British Columbia, Canada V6T 1Z1
[3]Stewart Blusson Quantum Matter Institute, University of British Columbia, Vancouver, British Columbia, Canada V6T 1Z4

(Received 21 March 2018; revised manuscript received 24 June 2018; published 17 December 2018)

We present a machine-learning method for predicting sharp transitions in a Hamiltonian phase diagram by extrapolating the properties of quantum systems. The method is based on Gaussian process regression with a combination of kernels chosen through an iterative procedure maximizing the predicting power of the kernels. The method is capable of extrapolating across the transition lines. The calculations within a given phase can be used to predict not only the closest sharp transition but also a transition removed from the available data by a separate phase. This makes the present method particularly valuable for searching phase transitions in the parts of the parameter space that cannot be probed experimentally or theoretically.

DOI: 10.1103/PhysRevLett.121.255702

It is very common in quantum physics to encounter a problem with the Hamiltonian

$$H = H_0 + \alpha H_1 + \beta H_2 \tag{1}$$

whose eigenspectrum can be readily computed or measured in certain limits of α and β, e.g., at $\alpha = 0$ or at $\alpha \gg \beta$, but not at arbitrary values of α and β. For such problems, it is necessary to interpolate the properties of the quantum system between the known limits or extrapolate from a known limit. Both the interpolation and the extrapolation become exceedingly complex if the system properties undergo sharp transitions at some values of α and/or β. Such sharp transitions separate the phases of the Hamiltonian (1). Because the wave functions of the quantum system are drastically different in the different phases [1], an extrapolation of quantum properties across phase transition lines is generally considered unfeasible.

Here, we challenge this premise. We note that, while certain properties of quantum systems undergo a sharp change at a phase transition, other properties evolve smoothly through the transition. Using the example of three different lattice models, we show that the evolution of such properties within a given phase contains information about the transitions and the same properties beyond the transitions. We present a machine-learning method that can be trained by *the evolution* of such properties in a given phase to predict the sharp transitions and the properties of the quantum system in other phases by extrapolation. The importance of this result is clear. Characterizing quantum phase transitions embodied in model Hamiltonians is one of the foremost goals of quantum condensed-matter physics. Our work illustrates the possibility of predicting transitions at Hamiltonian parameters, where obtaining the solutions of the Schrödinger equation may not be feasible.

The application of machine-learning (ML) tools for solving problems in condensed-matter physics has recently become popular [2–34]. In all of these applications, ML is used as an efficient method to solve one of three problems: interpolation, classification, or clustering. Interpolation amounts to fitting multidimensional functions or functionals, whereas classification and clustering are used to separate physical data by properties. For example, ML can be used to identify quantum phases of lattice spin Hamiltonians [5,6,12,16,19,23,24]. However, in order to identify a quantum phase transition by interpolation and/or classification, the aforementioned ML models must be trained (fed on input) by the data describing both phases on both sides of the transition. The distinct feature of the present work is a ML method that requires information from only one phase and *extrapolates* the properties of lattice models to and across the transitions. To illustrate the method, we consider four different problems: lattice polaron models with zero, one, and two sharp transitions and the mean-field Heisenberg model with a critical temperature. In all cases, we show that the phase transitions (or lack thereof) can be accurately identified.

We first consider a generalized lattice polaron model describing an electron in a one-dimensional lattice with $N \to \infty$ sites coupled to a phonon field:

$$\mathcal{H} = \sum_k \epsilon_k c_k^\dagger c_k + \sum_q \omega_q b_q^\dagger b_q + V_{e\text{-ph}}, \tag{2}$$

where c_k and b_q are the annihilation operators for the electron with momentum k and phonons with momentum q, $\epsilon_k = 2t\cos(k)$ and $\omega_q = \omega = \text{const}$ are the electron and

phonon dispersions, respectively, and $V_{e\text{-ph}}$ is the electron-phonon coupling. We choose $V_{e\text{-ph}}$ to be a combination of two qualitatively different terms $V_{e\text{-ph}} = \alpha H_1 + \beta H_2$, where

$$H_1 = \sum_{k,q} \frac{2i}{\sqrt{N}} [\sin(k+q) - \sin(k)] c^\dagger_{k+q} c_k (b^\dagger_{-q} + b_q) \tag{3}$$

describes the Su-Schrieffer-Heeger (SSH) [35] electron-phonon coupling and

$$H_2 = \sum_{k,q} \frac{2i}{\sqrt{N}} \sin(q) c^\dagger_{k+q} c_k (b^\dagger_{-q} + b_q) \tag{4}$$

is the breathing-mode model [36]. The lowest energy eigenstate of the model (2) represents polarons known to exhibit two sharp transitions as the ratio α/β increases from zero to large values [37]. At $\alpha = 0$, the model (2) describes breathing-mode polarons, which have no sharp transitions [38]. At $\beta = 0$, the model (2) describes SSH polarons, which exhibit one sharp transition in the polaron phase diagram [35]. At these transitions, the ground state momentum and the effective mass of the polaron change sharply.

Method.—We use Gaussian process (GP) regression as the prediction method [39], described in detail in Supplemental Material [40]. The goal of the prediction is to infer an unknown function $f(\cdot)$ given n inputs $\mathbf{x}_i$ and outputs y_i. The assumption is that $y_i = f(\mathbf{x}_i)$. The function f is generally multidimensional, so $\mathbf{x}_i$ is a vector.

GPs do not infer a single function $f(\cdot)$ but rather a distribution over functions, $p(f|\mathbf{X}, \mathbf{y})$, where $\mathbf{X}$ is a vector of all known $\mathbf{x}_i$ and $\mathbf{y}$ is a vector of the corresponding values y_i. This distribution is assumed to be normal. The joint Gaussian distribution of random variables $f(\mathbf{x}_i)$ is characterized by a mean $\mu(\mathbf{x})$ and a covariance matrix $K(\cdot,\cdot)$. The matrix elements of the covariance $K_{i,j}$ are specified by a kernel function $k(\mathbf{x}_i, \mathbf{x}_j)$ that quantifies the similarity relation between the properties of the system at two points $\mathbf{x}_i$ and $\mathbf{x}_j$ in the multidimensional space.

Prediction at $\mathbf{x}_*$ is done by computing the conditional distribution of $f(\mathbf{x}_*)$ given $\mathbf{y}$ and $\mathbf{X}$. The mean of the conditional distribution is [39]

$$\boldsymbol{\mu}(\mathbf{x}_*) = \sum_i^n d(\mathbf{x}_*, \mathbf{x}_i) y_i = \sum_i^n \alpha_i k(\mathbf{x}_*, \mathbf{x}_i), \tag{5}$$

where $\alpha = K^{-1}\mathbf{y}$ and $\mathbf{d} = K(\mathbf{x}_*, \mathbf{X})^\top K(\mathbf{X}, \mathbf{X})^{-1}$. The predicted mean $\mu(\mathbf{x}_*)$ can be viewed as a linear combination of the training data y_i or as a linear combination of the kernels connecting all training points $\mathbf{x}_i$ and the point $\mathbf{x}_*$, where the prediction is made. In order to train a GP model, one must choose an analytical representation for the kernel function.

To solve the *interpolation* problem, one typically uses a simple form for the kernel. In the limit of large n, any simple kernel function produces accurate interpolation results [39]. For example, k can be approximated by any of the following functions:

$$k_{\text{LIN}}(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^\top \mathbf{x}_j + \alpha, \tag{6}$$

$$k_{\text{RBF}}(\mathbf{x}_i, \mathbf{x}_j) = \exp\left[-\frac{1}{2} r^2(\mathbf{x}_i, \mathbf{x}_j)\right], \tag{7}$$

$$k_{\text{MAT}}(\mathbf{x}_i, \mathbf{x}_j) = \left[1 + \sqrt{5} r(\mathbf{x}_i, \mathbf{x}_j) + \frac{5}{3} r^2(\mathbf{x}_i, \mathbf{x}_j)\right] \times \exp\left[-\sqrt{5} r(\mathbf{x}_i, \mathbf{x}_j)\right], \tag{8}$$

$$k_{\text{RQ}}(\mathbf{x}_i, \mathbf{x}_j) = \left(1 + \frac{|\mathbf{x}_i - \mathbf{x}_j|^2}{2\alpha \ell^2}\right)^{-\alpha}, \tag{9}$$

where $r^2(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i - \mathbf{x}_j)^\top \times M \times (\mathbf{x}_i - \mathbf{x}_j)$ and M is a diagonal matrix with different length scales ℓ_d for each dimension of $\mathbf{x}_i$. The length-scale parameters ℓ_d, ℓ, and α are the free parameters. We describe them collectively by θ. A GP is trained by finding the estimate of θ (denoted by $\hat{\theta}$) that maximizes the logarithm of the *marginal likelihood* function:

$$\log p(\mathbf{y}|\mathbf{X}, \theta, \mathcal{M}_i) = -\frac{1}{2}\mathbf{y}^\top K^{-1}\mathbf{y} - \frac{1}{2}\log|K| - \frac{n}{2}\log 2\pi. \tag{10}$$

For the *extrapolation* problem, the prediction produced by Eq. (5) is clearly sensitive to the particular choice of the kernel function. While different interpolation problems can be solved with the same mathematical form of the kernel function, different extrapolation problems generally require different kernels. The key for successful extrapolation is thus to find the appropriate kernel function. Because we aim to solve a variety of different problems with varying underlying physics, the procedure for constructing the kernel must be fully automated and independent of the particular problem under consideration.

Here, we follow Refs. [47,48] to build a prediction method based on a *combination* of products of different kernels (6)–(9). To select the best combination, we use the Bayesian information criterion (BIC) [49],

$$\text{BIC}(\mathcal{M}_i) = \log p(\mathbf{y}|\mathbf{X}, \hat{\theta}, \mathcal{M}_i) - \frac{1}{2}|\mathcal{M}_i| \log n, \tag{11}$$

where $|\mathcal{M}_i|$ is the number of kernel parameters of kernel $\mathcal{M}_i$. Here, $p(\mathbf{y}|\mathbf{X}, \hat{\theta}, \mathcal{M}_i)$ is the marginal likelihood for an optimized kernel $\hat{\theta}$. It is impossible to train and try models with all possible combinations of kernels. We use an iterative procedure schematically depicted in Fig. 1.

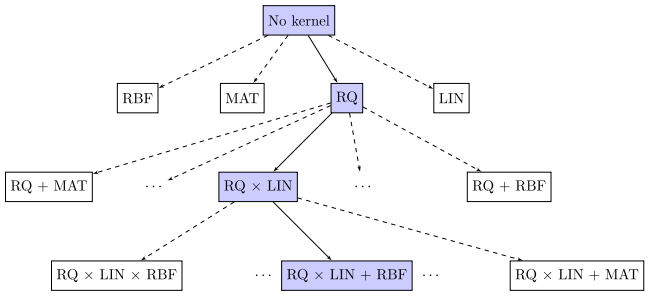

FIG. 1. Schematic diagram of the kernel construction method employed to develop a Gaussian process model with extrapolation power. At each iteration, the kernel with the highest Bayesian information criterion (11) is selected. The labels in the boxes correspond to the kernel functions defined in (6)–(9).

We begin by training a GP model with *each* of the kernels (6)–(9). These kernels have one (LIN), d (RBF and MAT), and two (RQ) free parameters [50]. The algorithm then selects the kernel—denoted k_0—that leads to the model with the highest BIC and combines $k_0(\cdot,\cdot)$ with each of the original kernels k_i defined by Eqs. (6)–(9). The kernels are combined as products $k_0(\cdot,\cdot) \times k_i(\cdot,\cdot)$ and additions $k_0(\cdot,\cdot) + k_i(\cdot,\cdot)$. Each kernel in the combination is scaled by a constant factor, which introduces another free parameter for the product or two parameters for the sum. For each of the possible combinations, a new GP model is constructed and a BIC is computed. The kernel yielding the highest BIC is then used as a new base kernel k_0, and the procedure is iterated. This fully automated algorithm is applied here to four different problems, yielding physical extrapolation results, thus showing that Eq. (11) can be used as a criterion for building prediction models capable of physical extrapolation.

Results.—All GP models are trained by the dispersions $E(K)$, where E is the polaron energy and K is the polaron momentum. These dispersions are calculated for infinite lattices using the momentum average (MA) approach from previous work [37,51–55]. The models are trained to predict the polaron energy as a function of K and the Hamiltonian parameters α, β, and ω. The vectors $\mathbf{x}_i$ are thus $\mathbf{x}_i \Rightarrow \{K, \omega, \alpha, \beta\}$, while $f(\cdot)$ is the polaron energy. Once the models are trained, we numerically compute the ground state momentum K_{GS} and the polaron effective mass from the predicted dispersions [56]. Note that we always train all models by the polaron dispersions in one phase and the models have no *a priori* information about the existence of another phase(s). The transition is encoded in the evolution of the polaron band as a function of $\mathbf{x}$. All results are in units of t.

Figure 2 shows the predictions for the pure SSH polaron model ($\beta = 0$, one sharp transition in the polaron phase diagram). The vertical lines show where the training points end and the extrapolation begins. As can be seen, the GP models predict accurately the location of the transition and can be used for quantitative extrapolation in a wide range of the Hamiltonian parameters to strong electron-phonon coupling. All models, including the ones trained by quantum calculations far removed from the transition point, predict accurately the location of the transition. As the coupling to phonons increases, the polaron develops a phonon-mediated next-nearest-neighbor hopping term: $E(K) = -2t\cos(K) + 2t_2(\lambda_{\mathrm{SSH}})\cos(2K)$, where $t_2(\lambda_{\mathrm{SSH}})$ is a function of λ_{SSH} [35]. The transition occurs when the

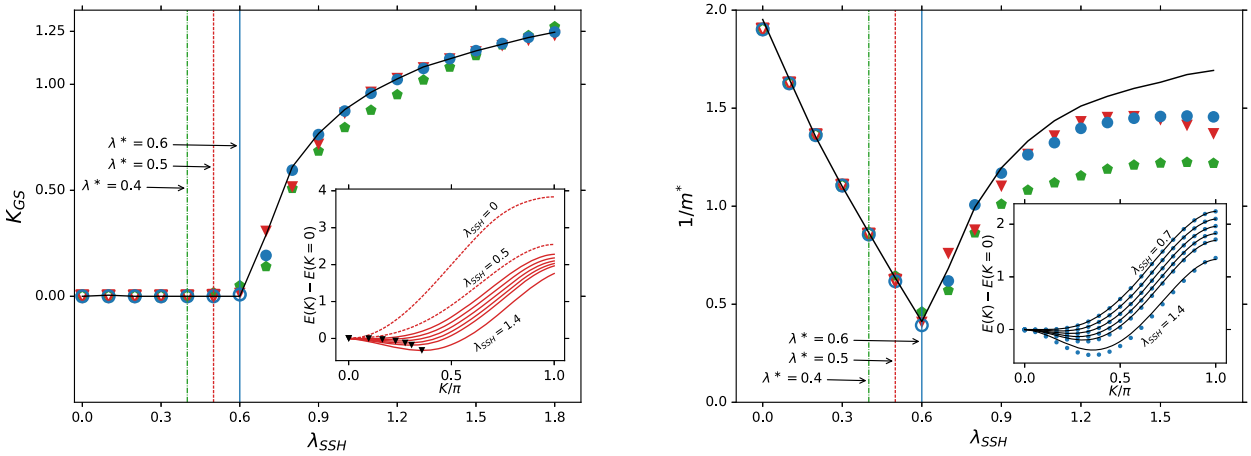

FIG. 2. Extrapolation of the polaron ground state momentum K_{GS} (left) and effective mass m^* (right) across the sharp transition at $\lambda_{\mathrm{SSH}} = 2\alpha^2/t\hbar\omega \approx 0.6$ for the model (2) with $\beta = 0$. The black solid curves are the accurate quantum calculations. The symbols are the predictions of the GP models trained by the full polaron dispersions $E(K)$ at values of $\lambda_{\mathrm{SSH}} \leq \lambda^*$, where λ^* is shown by the vertical lines (solid for circles, dashed for triangles, and dot-dashed for pentagons). The GP models are used for interpolation (open symbols) and extrapolation (full symbols). The algorithm of Fig. 1 yields the kernel $k_{\mathrm{RQ}} \times k_{\mathrm{LIN}} + k_{\mathrm{RBF}}$ for the GP models represented by the triangles and pentagons and $k_{\mathrm{RQ}} \times k_{\mathrm{LIN}} \times k_{\mathrm{MAT}}$ for the circles. Left inset: The polaron dispersions used as input (dashed curves) and predicted by the GP model (solid curves) with $\lambda^* = 0.5$ with the triangles showing the position of the dispersion minimum. Right inset: The polaron dispersions predicted by the GP model trained with $\lambda^* = 0.6$ (solid curves) in comparison with the quantum calculations (symbols).

second term dominates. Figure 2 shows that the GP models trained using the algorithm of Fig. 1 extrapolate accurately this evolution of the polaron energy.

The power of this method is better illustrated with the example of the mixed breathing-mode–SSH model ($\alpha \neq 0$, $\beta \neq 0$) with three phases [37]. The dots in Fig. 3 represent the points of the phase diagram used for training the GP model with the optimized kernels. Remarkably, the model trained by the polaron dispersions all entirely in one phase predicts *both* transitions. The location of the first transition is predicted quantitatively. The second transition is predicted qualitatively. If the model is trained by the polaron properties in two side phases and the prediction is made by extrapolation to low values of λ_{SSH} (lower panel in Fig. 3), both transition lines are predicted quantitatively.

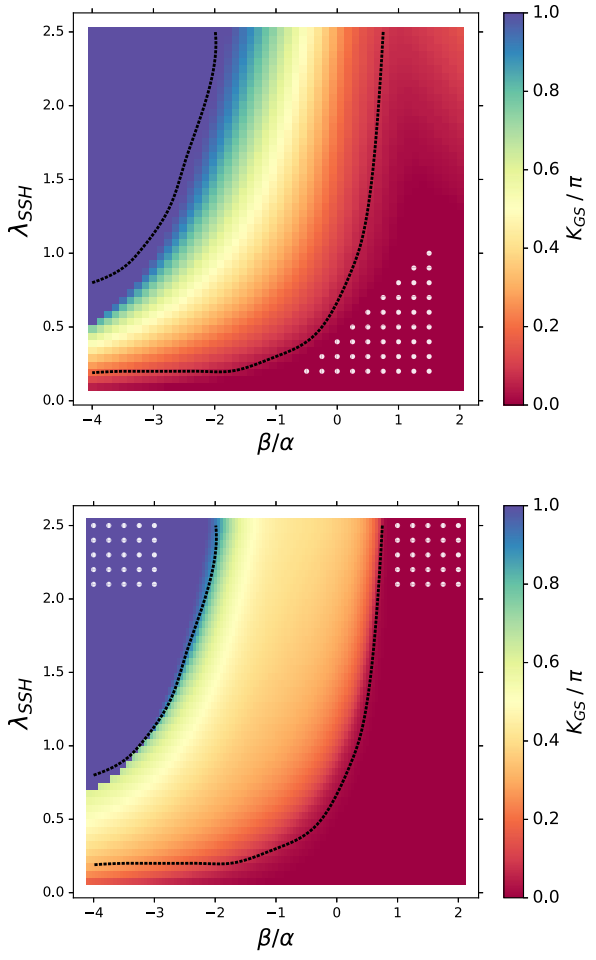

FIG. 3. The polaron ground state momentum K_{GS} for the mixed model (2) as a function of β/α for $\lambda_{\mathrm{SSH}} = 2\alpha^2/t\hbar\omega$. The color map is the prediction of the GP models. The curves are the quantum calculations from Ref. [37]. The models are trained by the polaron dispersions at the parameter values indicated by the white dots. No other information is used. The optimized kernel combination is $(k_{\mathrm{MAT}} + k_{\mathrm{RBF}}) \times k_{\mathrm{LIN}}$ (upper panel) and $(k_{\mathrm{MAT}} \times k_{\mathrm{LIN}} + k_{\mathrm{RBF}}) \times k_{\mathrm{LIN}}$ (lower panel).

As a third independent test, we applied the method to the Holstein polaron model defined by Eq. (2) with $V_{e\text{-ph}} = \mathrm{const}\sum_{k,q} c^{\dagger}_{k+q} c_k (b^{\dagger}_{-q} + b_q)$. Such a model is known to have no transitions [38]. We find that the method presented here can extrapolate accurately the polaron dispersions to a wide range of the Hamiltonian parameters and yields predictions that exhibit no sign of transitions. Since it is often not feasible to explore the entire phase diagram with rigorous quantum calculations, especially for models with many independent parameters, predicting the absence of transitions is as important as locating different phases.

Finally, we demonstrate the method on an analytically soluble model. We consider the Heisenberg model $H = -(J/2)\sum_{i,j} \vec{S}_i . \vec{S}_j$ in the nearest-neighbor approximation. Employing a mean-field description, the resulting free energy density at temperature T is [1,57,58]

$$f(T,m) \approx \frac{1}{2}\left(1 - \frac{T_c}{T}\right) m^2 + \frac{1}{12}\left(\frac{T_c}{T}\right)^3 m^4, \qquad (12)$$

where m is the magnetization and $T_c = 1.25J$ the critical temperature of the phase transition. $T > T_c$ corresponds to the paramagnetic phase, while $T < T_c$ is the ferromagnetic phase.

We train GP models by the results of Eq. (12) in the paramagnetic phase far away from T_c (shaded region in the inset in Fig. 4). We then extrapolate the function $f(T,m)$ across the critical temperature and compute the order parameter m_0 which minimizes $f(T,m)$. Figure 4 demonstrates that m_0 thus predicted can be accurately extrapolated across T_c and far into a different phase. This demonstrates again the general idea behind the technique developed here:

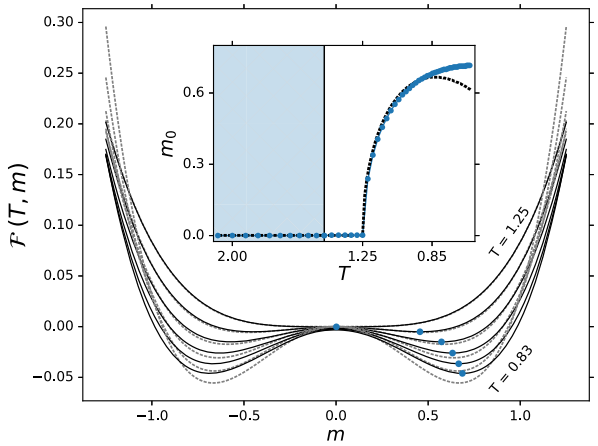

FIG. 4. GP prediction (solid curves) of the free energy density $f(T,m)$ of the mean-field Heisenberg model produced by Eq. (12) (dashed curves). Inset: The order parameter m_0 that minimizes $f(T,m)$: symbols, GP predictions; dashed curve, from Eq. (12). The GP models are trained with 330 points at $1.47 < T < 2.08$ (shaded area) and $-1.25 < m < 1.25$.

use ML to predict the *evolution* of continuous functions that encodes phase transitions.

It is important to point out that the iterative kernel selection algorithm of Fig. 1 must be analyzed before the present method is used for the *quantitative* extrapolation. As the iterations continue, the kernels become more complex, more prone to overfitting, and more difficult to optimize. The quantitative accuracy of the prediction may, therefore, decrease. Supplemental Material [40] illustrates the convergence to Figs. 2–4 with the kernel optimization levels and also the increase of the prediction error after a certain number of levels. To prevent this problem, we stop the kernel optimization when the prediction error is minimal, as explained in Supplemental Material [40]. We emphasize that this does not affect the prediction of the transitions: Once a certain level of Fig. 1 is reached, kernels from the subsequent optimization levels predict the transitions. We have confirmed this for all the results (Figs. 2–4) presented here. Thus, if the goal is to predict the presence or absence of transitions, this method can be used without validation. It is sufficient to check that subsequent levels of the kernel optimization do not produce or eliminate transitions. In order to predict quantitatively the quantum properties by extrapolation, the training data must be divided into the training and validation sets. The models must then be trained with the training set and the error calculated with the validation set. The kernel optimization must then be stopped, when the error is minimal. This is a common approach to prevent the overfitting problem in ML with artificial neural networks.

Summary.—We have presented a powerful method for predicting sharp transitions in Hamiltonian phase diagrams by extrapolating the properties of quantum systems. The method is based on Gaussian process regression with a combination of kernels chosen through an iterative procedure maximizing the predicting power of the kernel. The model thus obtained captures the change of the quantum properties as the system approaches the transition, allowing the extrapolation of the physical properties, even across sharp transition lines.

We believe that the present work is the first example of the application of ML for the extrapolation of physical observables for quantum systems. We have demonstrated that the method is capable of using the properties of the quantum system within a given phase to predict not only the closest sharp transition but also a transition removed from the training points by a separate phase. This makes the present method particularly valuable for searching phase transitions in the parts of the parameter space that cannot be probed experimentally or theoretically. Given that the training of the models and the predictions do not present any numerical difficulty [59], the present method can also be used to guide rigorous theory or experiments in search for phase transitions. Finally, we must note that, although the present extrapolation method works well for all four problems considered, we cannot prove that it is accurate for an arbitrary system so the predictions must always be validated, as is common in machine learning.

We acknowledge useful discussions with Dries Sels and Mathias Sheurer. This work was supported by the Natural Sciences and Engineering Research Council of Canada (NSERC) and by a visiting student fellowship at the Institute for Theoretical Atomic, Molecular, and Optical Physics (ITAMP) at Harvard University and the Smithsonian Astrophysical Observatory (J. S.).

[1] S. Sachdev, *Quantum Phase Transitions* (Cambridge University Press, Cambridge, England, 1999).
[2] L.-F. Arsenault, A. Lopez-Bezanilla, O. A. von Lilienfeld, and A. J. Millis, Phys. Rev. B **90**, 155136 (2014).
[3] L.-F. Arsenault, O. A. von Lilienfeld, and A. J. Millis, arXiv:1506.08858.
[4] T. Ohtsuki and T. Ohtsuki, J. Phys. Soc. Jpn. **85**, 123706 (2016).
[5] L. Wang, Phys. Rev. B **94**, 195105 (2016).
[6] J. Carrasquilla and R. G. Melko, Nat. Phys. **13**, 431 (2017).
[7] E. P. L. van Nieuwenburg, Y.-H. Liu, and S. D. Huber, Nat. Phys. **13**, 435 (2017).
[8] P. Broecker, F. Assaad, and S. Trebst, arXiv:1707.00663.
[9] S. J. Wetzel and M. Scherzer, Phys. Rev. B **96**, 184410 (2017).
[10] S. J. Wetzel, Phys. Rev. E **96**, 022140 (2017).
[11] Y.-H. Liu and E. P. L. van Nieuwenburg, Phys. Rev. Lett. **120**, 176401 (2018).
[12] K. Ch'ng, J. Carrasquilla, R. G. Melko, and E. Khatami, Phys. Rev. X **7**, 031038 (2017).
[13] P. Broecker, J. Carrasquilla, R. G. Melko, and S. Trebst, Sci. Rep. **7**, 8823 (2017).
[14] F. Schindler, N. Regnault, and T. Neupert, Phys. Rev. B **95**, 245134 (2017).
[15] M. J. S. Beach, A. Golubeva, and R. G. Melko, Phys. Rev. B **97**, 045207 (2018).
[16] E. van Nieuwenburg, E. Bairey, and G. Refael, Phys. Rev. B **98**, 060301(R) (2018).
[17] N. Yoshioka, Y. Akagi, and H. Katsura, Phys. Rev. B **97**, 205110 (2018).
[18] J. Venderley, V. Khemani, and E.-A. Kim, Phys. Rev. Lett. **120**, 257204 (2018).
[19] G. Carleo and M. Troyer, Science **355**, 602 (2017).
[20] M. Schmitt and M. Heyl, SciPost Phys. **4**, 013 (2018).
[21] Z. Cai and J. Liu, Phys. Rev. B **97**, 035116 (2018).
[22] Y. Huang and J. E. Moore, arXiv:1701.06246.
[23] D.-L. Deng, X. Li, and S. Das Sarma, Phys. Rev. B **96**, 195145 (2017).
[24] Y. Nomura, A. S. Darmawan, Y. Yamaji, and M. Imada, Phys. Rev. B **96**, 205152 (2017).
[25] D.-L. Deng, X. Li, and S. Das Sarma, Phys. Rev. X **7**, 021021 (2017).
[26] X. Gao and L.-M. Duan, Nat. Commun. **8**, 662 (2017).
[27] G. Torlai, G. Mazzola, J. Carrasquilla, M. Troyer, R. Melko, and G. Carleo, Nat. Phys. **14**, 447 (2018).
[28] T. Hazan and T. Jaakkola, arXiv:1508.05133.

[29] A. Daniely, R. Frostig, and Y. Singer, NIPS **29**, 2253 (2016).
[30] J. Lee, Y. Bahri, R. Novak, S. S. Schoenholz, J. Pennington, and J. Sohl-Dickstein, in Deep Neural Networks as Gaussian Processes (International Conference on Machine Learning, Vancouver, 2018).
[31] K. T. Schütt, H. Glawe, F. Brockherde, A. Sanna, K. R. Müller, and E. K. U. Gross, Phys. Rev. B **89**, 205118 (2014).
[32] L. M. Ghiringhelli, J. Vybiral, S. V. Levchenko, C. Draxl, and M. Scheffler, Phys. Rev. Lett. **114**, 105503 (2015).
[33] F. A. Faber, A. Lindmaa, O. A. von Lilienfeld, and R. Armient, Int. J. Quantum Chem. **115**, 1094 (2015).
[34] F. A. Faber, A. Lindmaa, O. A. von Lilienfeld, and R. Armient, Phys. Rev. Lett. **117**, 135502 (2016).
[35] D. J. J. Marchand, G. De Filippis, V. Cataudella, M. Berciu, N. Nagaosa, N. V. Prokof'ev, A. S. Mishchenko, and P. C. E. Stamp, Phys. Rev. Lett. **105**, 266605 (2010).
[36] B. Lau, M. Berciu, and G. A. Sawatzky, Phys. Rev. B **76**, 174305 (2007).
[37] F. Herrera, K. W. Madison, R. V. Krems, and M. Berciu, Phys. Rev. Lett. **110**, 223002 (2013).
[38] B. Gerlach and H. Löwen, Rev. Mod. Phys. **63**, 63 (1991).
[39] C. E. Rasmussen and C. K. I. Williams, *Gaussian Process for Machine Learning* (MIT Press, Cambridge, 2006).
[40] See Supplemental Material at http://link.aps.org/supplemental/10.1103/PhysRevLett.121.255702 for details of the ML methods, convergence, and the quantum calculations used for training the ML models. Supplemental Material includes Refs. [41–46].
[41] R. S. Sutton and A. G. Barto, *Reinforcement Learning, An Introduction* (MIT Press, Cambridge, 2016).
[42] M. Berciu, Phys. Rev. Lett. **97**, 036402 (2006).
[43] M. Berciu and G. L. Goodvin, Phys. Rev. B **76**, 165109 (2007).
[44] G. L. Goodvin and M. Berciu, Phys. Rev. B **78**, 235120 (2008).
[45] C. P. J. Adolphs and M. Berciu, Phys. Rev. B **90**, 085149 (2014).
[46] M. Berciu and H. Fehske, Phys. Rev. B **82**, 085116 (2010).
[47] D. K. Duvenaud, H. Nickisch, and C. E. Rasmussen, Adv. Neural Inf. Process. Syst. **24**, 226 (2011).
[48] D. K. Duvenaud, J. Lloyd, R. Grosse, J. B. Tenenbaum, and Z. Ghahramani, Proc. Mach. Learn. Res. **28**, 1166 (2013).
[49] G. Schwarz, Ann. Stat. **6**, 461 (1978).
[50] See Supplemental Material at http://link.aps.org/supplemental/10.1103/PhysRevLett.121.255702 for more details on combining kernels in Sec. I. A.
[51] M. Berciu, Phys. Rev. Lett. **97**, 036402 (2006).
[52] J. Sous, M. Chakraborty, C. P. J. Adolphs, R. V. Krems, and M. Berciu, Sci. Rep. **7**, 1169 (2017).
[53] J. Sous, M. Berciu, and R. V. Krems, Phys. Rev. A **96**, 063619 (2017).
[54] J. Sous, M. Chakraborty, R. V. Krems, and M. Berciu, arXiv:1805.06109 [Phys. Rev. Lett. (to be published)].
[55] See Supplemental Material at http://link.aps.org/supplemental/10.1103/PhysRevLett.121.255702 for more details of the MA approach in Sec. III.
[56] See Supplemental Material http://link.aps.org/supplemental/10.1103/PhysRevLett.121.255702 for more details in Sec. II. A.
[57] P. M. Chaikin and T. C. Lubensky, *Principles of Condensed Matter Physics* (Cambridge University Press, Cambridge, England, 1998).
[58] See Supplemental Material at http://link.aps.org/supplemental/10.1103/PhysRevLett.121.255702 for more details in Sec. III. B.
[59] See Supplemental Material at http://link.aps.org/supplemental/10.1103/PhysRevLett.121.255702 for the description of the computational scaling for the ML method adopted here in Sec. I. B.

Supplemental Material for "Extrapolating quantum observables with machine learning: Inferring multiple phase transitions from properties of a single phase"

Rodrigo A. Vargas-Hernández,[1] John Sous,[1,2,3] Mona Berciu,[2,3] and Roman V. Krems[1]

[1] *Department of Chemistry, University of British Columbia, Vancouver, British Columbia, Canada, V6T 1Z1*

[2] *Department of Physics and Astronomy, University of British Columbia, Vancouver, British Columbia, Canada, V6T 1Z1*

[3] *Stewart Blusson Quantum Matter Institute, University of British Columbia, Vancouver, British Columbia, Canada, V6T 1Z4*

(Dated: November 30, 2018)

The purpose of this supplemental material is to provide details of the numerical calculations we present in this work. Sections I and II discuss the machine-learning methods and Sections III – the quantum calculations used to train the ML models.

I. GP REGRESSION WITH KERNEL COMBINATIONS

Gaussian process (GP) regression is a kernel-based probabilistic non-parametric supervised ML algorithm [1]. Within the GP regression framework, the prediction is a normal distribution characterized by a mean $\mu(\cdot)$ and a standard deviation $\sigma(\cdot)$, given as

$$\mu(\boldsymbol{x}_*) = K(\boldsymbol{x}_*, \boldsymbol{x})^\top \left[K(\boldsymbol{x}, \boldsymbol{x}) + \sigma_n^2 I\right]^{-1} \boldsymbol{y} \tag{1}$$

$$\sigma(\boldsymbol{x}_*) = K(\boldsymbol{x}_*, \boldsymbol{x}_*) - K(\boldsymbol{x}_*, \boldsymbol{x})^\top \left[K(\boldsymbol{x}, \boldsymbol{x}) + \sigma_n^2 I\right]^{-1} K(\boldsymbol{x}_*, \boldsymbol{x}). \tag{2}$$

Here,

- $\boldsymbol{x} = (\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_n)^\top$ is a vector of n points in a multi-dimensional parameter space, where the GP model is trained;
- $\boldsymbol{x}_i$ is a vector of variable parameters.

  For the case of the polaron models considered here,
  $\boldsymbol{x}_i \Rightarrow \{\text{polaron momentum } K,\ \text{Hamiltonian parameter } \alpha,\ \text{Hamiltonian parameter } \beta, \text{phonon frequency } \omega\}$.
  For the case of the Heisenberg model considered here, $\boldsymbol{x}_i \Rightarrow \{\text{Temperature } T,\ \text{magnetization } \tilde{m}\}$;
- $\boldsymbol{y} = f(\boldsymbol{x})$ is a vector of quantum mechanics results at the values of the parameters specified by $\boldsymbol{x}_i$

  For the case of the polaron models considered here, $\boldsymbol{y} \Rightarrow$ polaron energy E.
  For the case of the Heisenberg model considered here, $\boldsymbol{y} \Rightarrow$ free energy density;
- $\boldsymbol{x}_*$ is a point in the parameter space where the prediction $\boldsymbol{y}_*$ is to be made;
- $K(\boldsymbol{x}, \boldsymbol{x})$ is the $n \times n$ square matrix with the elements $K_{i,j} = k(\boldsymbol{x}_i, \boldsymbol{x}_j)$ representing the covariances between $\boldsymbol{y}(\boldsymbol{x}_i)$ and $\boldsymbol{y}(\boldsymbol{x}_j)$. The elements $k(\boldsymbol{x}_i, \boldsymbol{x}_j)$ are represented by the kernel function.

The GP models are constructed (in the language of ML "trained") by the quantum mechanics results $\boldsymbol{y}$ at the parameters in $\boldsymbol{x}$. The unknown in this model is the kernel function. The goal of the training is thus to find the best representation for the kernel function $k(\cdot, \cdot)$.

In a standard procedure for training a GP model, one begins by assuming some simple analytical functional form for $k(\cdot, \cdot)$. For example, one assumes *one* of the following functional forms:

$$k_{\text{LIN}}(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^\top \mathbf{x}_j \tag{3}$$

$$k_{\text{RBF}}(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{1}{2} r^2(\mathbf{x}_i, \mathbf{x}_j)\right) \tag{4}$$

$$k_{\text{MAT}}(\mathbf{x}_i, \mathbf{x}_j) = \left(1 + \sqrt{5} r(\mathbf{x}_i, \mathbf{x}_j) + \frac{5}{3} r^2(\mathbf{x}_i, \mathbf{x}_j)\right) \times \exp\left(-\sqrt{5} r(\mathbf{x}_i, \mathbf{x}_j)\right) \tag{5}$$

$$k_{\text{RQ}}(\mathbf{x}_i, \mathbf{x}_j) = \left(1 + \frac{|\mathbf{x}_i - \mathbf{x}_j|^2}{2\alpha \ell^2}\right)^{-\alpha} \tag{6}$$

where $r^2(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i - \mathbf{x}_j)^\top \times M \times (\mathbf{x}_i - \mathbf{x}_j)$ and M is a diagonal matrix with different length-scales ℓ_d for each dimension of $\mathbf{x}_i$. This list represents some of the most commonly used kernel functions.

The parameters of this analytical form are then found by maximizing the log *marginal likelihood* function,

$$\log p(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{\theta}) = -\frac{1}{2} \boldsymbol{y}^\top K^{-1} \boldsymbol{y} - \frac{1}{2} \log |K| - \frac{n}{2} \log(2\pi), \tag{7}$$

where θ denotes collectively the parameters of the analytical function for $k(\cdot, \cdot)$ and $|K|$ is the determinant of the matrix K.

The marginal likelihood can also be used as a metric to compare different kernels. However, care must be taken when kernels with different numbers of parameters are to be compared. The second term of Eq. (7) directly depends on the number of parameters in the kernel, which makes the log marginal likelihood inappropriate to compare kernels with different numbers of parameters. To overcome this issue, we compare the predictive power of different kernels using the Bayesian Information criterion (BIC) [3]

$$\mathrm{BIC}(\mathcal{M}_i) = \log p(\mathbf{y}|\boldsymbol{x}, \hat{\theta}, \mathcal{M}_i) - \frac{1}{2}|\mathcal{M}_i| \log n \tag{8}$$

where $|\mathcal{M}_i|$ is the number of kernel parameters of the kernel $\mathcal{M}_i$. Here, $p(\mathbf{y}|\boldsymbol{x}, \hat{\theta}, \mathcal{M}_i)$ is the marginal likelihood for the optimized kernel $\hat{\theta}$ which maximizes the logarithmic part. The last term in Eq. (8) acts to penalize kernels with larger number of parameters to reduce overfitting, thus making the predicting model more robust.

A. Learning with kernel combinations

Typically, GP regression is used for interpolation of the training points $\boldsymbol{y}(\boldsymbol{x}_i)$. For this problem, it is sufficient to choose one of the kernel functions above. The choice of the function will determine the efficiency of the interpolation model, i.e. the number n of training points required for accurate predictions between the training points. However, any of the kernel functions will work for interpolation.

As discussed in the main text, this is not the case for extrapolation. For accurate extrapolation, one needs to increase the complexity of kernels in order to capture the physical behaviour of the training data $\boldsymbol{y}(\boldsymbol{x}_i)$ well. However, complex kernels come with a risk of overfitting. In addition, the ambiguity as to the choice of the kernel function increases with the complexity of the kernel function. So, the question is, how to increase the complexity of the kernel functions in a systematic way that prevents overfitting and results in a model that captures the physical behaviour of the training results?

Ideally, one should choose a kernel function that captures all of the physical behaviour of the training data. However, as we explained above, hand-crafting the 'best' kernels is not an easy task [1, 5]. In addition, hand-crafting kernels may introduce biases, limiting the generality of the prediction. In this work, we do not use any prior information for the selection of the kernels and we do not hand-craft kernels.

Here, we follow Refs. [2, 5] to increase the complexity of kernels by combining the simple functions (3) - (6) into products and sums. Combining different kernels can enhance the learning capacity of the GP regression [2].

For example, the first kernel combination that we describe here is the addition of two kernels like $k_{MAT} + k_{MAT}$ or $k_{RQ} + k_{MAT}$. This new type of kernel form is capable of learning long-range and short-range correlations between data points. Multiplication of kernels is also another possible algebraic operation, for example, $k_{RQ} \times k_{MAT}$. Multiplying any of the kernels by the linear kernel, *e.g.*, $k_{RBF} \times k_{LIN}$, leads to a GP regression that can learn increasing variations of the data. The dot-product/linear kernel, Eq. (3), can be used to construct polynomial kernels. For example, to describe quadratic functions, one could multiply this kernel by itself: $k_{LIN} \times k_{LIN}$.

It becomes clear that combining kernels in GP regression can provide an advantage in describing a variety of mathematical functions to accurately make predictions. This is the basis behind using GP regression with kernel combinations for extrapolating observables. To build more robust and flexible GP models, we employ the greedy search algorithm and the BIC to algorithmically construct the 'optimal' model. The greedy search is an 'optimal policy' algorithm [4] that selects the kernel assumed optimal based on the BIC at every step in the search. The underlying assumption is that the BIC represents the optimal measure of the kernel performance.

The number of free parameters for each of the simple kernels used in this work are

- $k_{LIN}(\mathbf{x}_i, \mathbf{x}_j) \Rightarrow 1$
- $k_{RBF}(\mathbf{x}_i, \mathbf{x}_j) \Rightarrow d$
- $k_{MAT}(\mathbf{x}_i, \mathbf{x}_j) \Rightarrow d$
- $k_{RQ}(\mathbf{x}_i, \mathbf{x}_j) \Rightarrow 2$,

where d is the dimensionality or number of features of the data. For example, for the results presented in Figure 4 of the main text $d = 2$, $\mathbf{x}_i = [T, m]$.

Every kernel considered in this work is scaled by the constant kernel, $k_c(\mathbf{x}_i, \mathbf{x}_j) = const.$ The total number of parameters of a GP model with any simple kernel considered in this work is thus increased by one due to $k_c(\mathbf{x}_i, \mathbf{x}_j) \times k_X(\mathbf{x}_i, \mathbf{x}_j)$, where k_X is any of the kernels listed above.

As the algorithm depicted in Figure 1 progresses to lower levels, the number of free kernel parameters increases and the kernels become rather complex. We express such kernels as the sum of products of kernels by distributing all products of sums. For example, the kernel used to construct Figure 3 (lower panel) of the main text is,

$$(k_{MAT} \times k_{LIN} + k_{RBF}) \times k_{LIN} = k_{MAT} \times k_{LIN} \times k_{LIN} + k_{RBF} \times k_{LIN},$$

which including the constant kernel is,

$$\big(k_c \times k_{MAT} \times k_{LIN} \times k_{LIN}\big) + \big(k_c \times k_{RBF} \times k_{LIN}\big).$$

B. Numerical difficulty of training and using the GP models

In order to train a GP model, one needs to maximize the log-likelihood function in Eq. (7) by iteratively computing the inverse and the determinant of the correlation matrix K. The dimension of this matrix is $n \times n$, where n is the number of training points. In this work, $n \approx 200 - 1000$, as discussed in the next section. Therefore, training a single GP model presents no numerical difficulty and typically takes seconds to minutes of CPU time.

In order to find the optimal kernels using the algorithm depicted in Figure 1 of the main manuscript, one needs to train many GP models. As the levels in Figure 1 become deeper, kernels become more complex and the algorithm requires the iterative construction of more GP models. For levels 5 to 10 of Figure 1, the kernel optimization may take up to a few hours of CPU time on a single compute core.

Using the model to predict the quantum properties involves the evaluation of the vector - matrix product in Eq. (1). The size of the vector n and the dimension of the matrix is $n \times n$, where $n \approx 200 - 1000$, as before. (Since the matrix K is, at this point known, the prediction may actually be reduced to a scalar product of two vectors of size n). The numerical evaluation of these products presents no computational difficulty.

II. SPECIFIC DETAILS OF THE EXTRAPOLATION METHOD

The value of a quantum observable depends on the parameters of the Hamiltonian. One can learn the behavior of quantum observables using different ML models using the following relation

$$E = \langle \hat{\mathcal{H}}(K, \alpha, \beta, \dots) \rangle \sim \mathcal{F}(K, \alpha, \beta, \dots) \tag{9}$$

where $\mathcal{F}(\cdot)$ is any ML model that can learn the dependence between the Hamiltonian parameters and the quantum observable. In the present work, the quantum observables are the polaron ground state energy and the free-energy density denoted as $E(\cdot)$. We use the algorithm proposed above to learn $\mathcal{F}(\cdot)$ and hence to extrapolate quantum observables.

The results of Figure 2 of the main text are for the polaron model with $\beta = 0$. This figure presents the extrapolation with three different ML models, represented by circles, triangles and pentagons.

For the predictions represented by triangles:

- The GP model is trained with 210 points distributed in the ranges $0 \leq K \leq \pi$, $0 \leq \lambda_{SSH} \leq 0.5$

For the predictions represented by circles:

- The GP model is trained with 245 points distributed in the ranges $0 \leq K \leq \pi$, $0 \leq \lambda_{SSH} \leq 0.6$

For the predictions represented by pentagons:

- The GP model is trained with 175 points distributed in the ranges $0 \leq K \leq \pi$, $0 \leq \lambda_{SSH} \leq 0.4$

The results of Figure 3 of the main text are for the polaron model with $\alpha \neq 0$ and $\beta \neq 0$. This figure presents the extrapolation with the ML models, trained by the quantum calculations at the Hamiltonian parameters shown by white circles in Figure 3 of the main text. For each training point (each white circle), we use 16 energy points in in the range $0 \leq K \leq \pi$ for the total of 900 training points for the upper panel of Figure 3 and 960 training points for the lower panel of Figure 3.

A. Effective mass and ground state momentum from extrapolated results

Given the GP extrapolated $E(K)$, we compute K_{GS} and m^* as follows. K_{GS} is the value of the momentum that minimizes $E(K)$

$$K_{GS}(\alpha, \beta, \cdots) = \arg\min_{K} \; E(K, \alpha, \beta, \cdots) \tag{10}$$

which depends on the Hamiltonian parameters α and β. For all results presented here, we compute K_{GS} by searching for the value where $E(K)$ is minimum. This procedure is depicted in Figure SM 1.

The polaron effective mass is,

$$m^*(\lambda_{SSH}) = \left[\frac{\partial^2 E_K(\lambda_{SSH})}{\partial K^2}\right]^{-1}\Big|_{K=K_{GS}} \tag{11}$$

To compute m^*, we numerically evaluate the second derivative of the extrapolated $E(K)$.

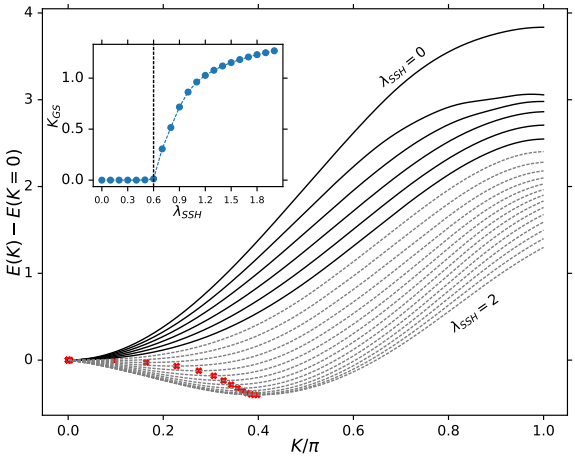

Fig. SM 1: SSH Polaron dispersions predicted (dashed curves) with a GP model trained by the quantum calculations (black solid curves) from Ref. [6] to result in the kernel $k_{RQ} \times k_{LIN} + k_{RBF}$. The red crosses indicate the positions where the polaron dispersion reaches its minimum. Inset: the value of K_{GS} as a function of λ_{SSH}.

B. Prediction accuracy convergence (number of training points)

Figure SM 2 illustrates how the accuracy of the extrapolation improves with the number of training points. We consider the pure SSH polaron model with one sharp transition. The GP models are trained by quantum results at $\lambda_{SSH} \leq 0.4$, which is far below the transition point $\lambda_{SSH} \approx 0.6$, and used to predict the polaron properties after the transition, at $\lambda_{SSH} > 0.6$. In all the cases presented, the kernel search algorithm depicted in Figure 1 of the main manuscript is run for three depth levels.

All models are trained by the quantum results at 5 values of $\lambda_{SSH} \leq 0.4$, but with a different number of points at $0 \leq K \leq \pi$: 15 (triangles), 25 (squares), 35 (circles). Figure SM 2 clearly shows that the accuracy of the prediction dramatically improves with the number of training points.

C. Prediction accuracy convergence (kernel complexity dependence)

For clarity, here, we use the notation "GPL-X" for the kernel with the highest BIC obtained after X depth levels of the algorithm depicted in Figure 1 of the main manuscript. "X" thus denotes the depth of the kernel optimization diagram shown in Figure 1 of the main manuscript. Figures SM 3 and SM 4 show how the accuracy of the prediction of the sharp transitions shown in Figure 3 of the main text improves as the kernel complexity increases.

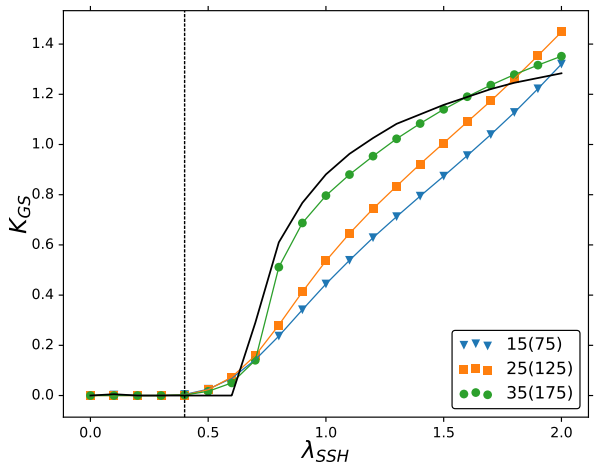

Fig. SM 2: Ground state momentum K_{GS} for the predicted SSH Polaron dispersions with a GP model trained at $\lambda_{SSH} \leq 0.4$ by thee different sets of points: blue triangles – 15 points per value of λ_{SSH} (75 points total); orange squares – 25 points per value of λ_{SSH} (125 points total); green circles – 35 points per value of λ_{SSH} (175 points total). The black solid curve is the rigorous result from Ref. [6].

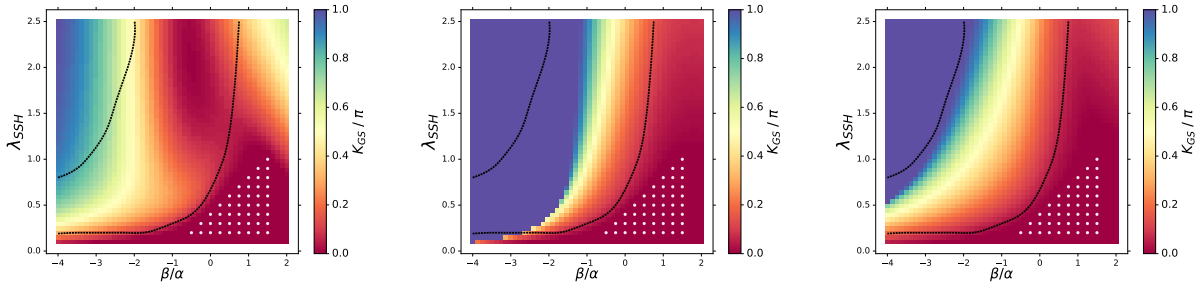

Fig. SM 3: Improvement of the phase diagram shown in Figure 3 (upper panel) of the main manuscript with the kernel complexity increasing as determined by the algorithm depicted in Figure 1 of the main manuscript. The panels correspond to the optimized kernels GPL-0 (left), GPL-1 (center), GPL-2 (right), where "GPL-X" denotes the optimal kernel obtained after X depth levels.

For all of the calculations presented, we verified that increasing X (the number of levels in the kernels optimization) does not change the predictions of the phase transitions. This applies to all results in Figures 2, 3 and 4 in the main manuscript as well as the Holstein model results discussed in the main text. Once a phase transition (or the absence of transitions) is identified, the prediction of the phase transition (or the absence of transitions) is reliable. In other words, once a certain level of kernel optimization is reached, all kernels from the subsequent optimization levels predict the phase transitions or the absence of the phase transitions correctly.

However, as the complexity of the kernels increases with each new level X, it becomes more difficult to find the optimal kernel within a given level X. The optimization algorithm is more likely to be stuck in a local minimum. This does not affect the predictions of the phase transitions. However, the quantitative predictions of the quantum properties in the extrapolated phase may be affected. Both of these points are illustrated in Figure SM 5 (upper right panel). To prevent this problem, in the present work, we stop the optimization algorithm after three levels of optimization for the results in Figures 2, 3 (upper panel) and 4. For the results in Figure 3 (lower panel), we stop the optimization after four levels.

These results show that, if the goal is to predict the presence or absence of phase transitions, the method described here can be used without validation. It is sufficient to ensure that subsequent levels of the kernel optimization do not produce or eliminate phase transitions. If the goal is to predict quantitatively the quantum properties by extrapolation, the training data must be divided into a training and validation sets. The models must then be trained with the training set and the error calculated with the validation set. The kernel optimization must then be stropped at the level of the diagram in Figure 1, where the error is the smallest. This is one of the approaches to prevent the

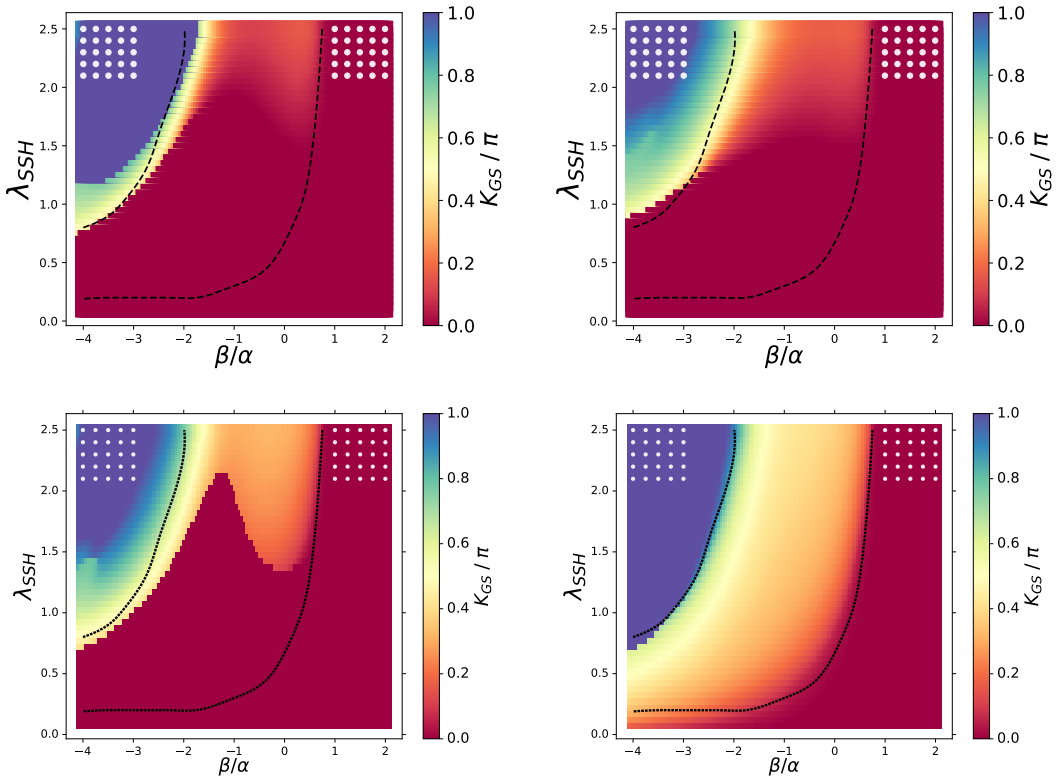

Fig. SM 4: Improvement of the phase diagram shown in Figure 3 (lower panel) of the main manuscript with the kernel complexity increasing as determined by the algorithm depicted in Figure 1 of the main manuscript. The panels correspond to the optimized kernels GPL-0 (upper left), GPL-1 (upper right), GPL-2 (lower left), GPL-3 (lower right), where "GPL-X" denotes the optimal kernel obtained after X depth levels.

overfitting problem in machine learning with artificial neural networks.

III. QUANTUM CALCULATIONS TO OBTAIN TRAINING DATA

A. Polaron models

For the polaron models, we use the Momentum Average (MA) approximation yielding accurate results for the polaron energies in one-dimensional lattices of infinite size [7–9].

The MA approach is a non-perturbative quasi-analytical technique designed to solve the equation of motion for the Green's function $\hat{G}(k,\omega) = \langle k| \left(\omega - \hat{H} + i\eta\right)^{-1} |k\rangle$. We use the Dyson's identity $\hat{G}(\omega) = \hat{G}_0(\omega) + \hat{G}(\omega)\hat{V}\hat{G}_0(\omega)$ to generate the hierarchy of equations of motion. $\hat{G}(\omega) = (\omega - \hat{H} + i\eta)^{-1}$, $\hat{G}_0(\omega) = (\omega - \hat{H}_0 + i\eta)^{-1}$ with $\hat{H}_0 = \hat{H} - \hat{V}_{\mathrm{e-ph}}$, and $\hat{V} = \hat{V}_{\mathrm{e-ph}}$ is the electron-phonon coupling term. This hierarchy consists of an infinite set of coupled equations making an exact solution impossible.

The MA approach acts to guide an insightful approximation/truncation of the hierarchy allowing for efficient yet accurate computations by neglecting the exponentially small diagrams in the expansion. The set of diagrams retained in the hierarchy is identified by considering the variational meaning of MA: one allows for boson excitations only within a finite spatial cut-off from the electron in the polaron cloud [8].

This choice of the variational space depends on the details of the Hamiltonian and states of interest [8]. For the Holstein model, a one-site phonon cloud suffices to provides accurate results for single polarons [7, 8] and for tightly bound bipolarons [10]. For the SSH model, the coupling to phonons is non-local and therefore a bigger cloud is required to yield accurate results. A three-site phonon cloud MA has been shown to be very accurate for such models [6, 12–15].

By design, the MA approach computes the proprieties of polarons in infinite lattices by utilizing the momentum

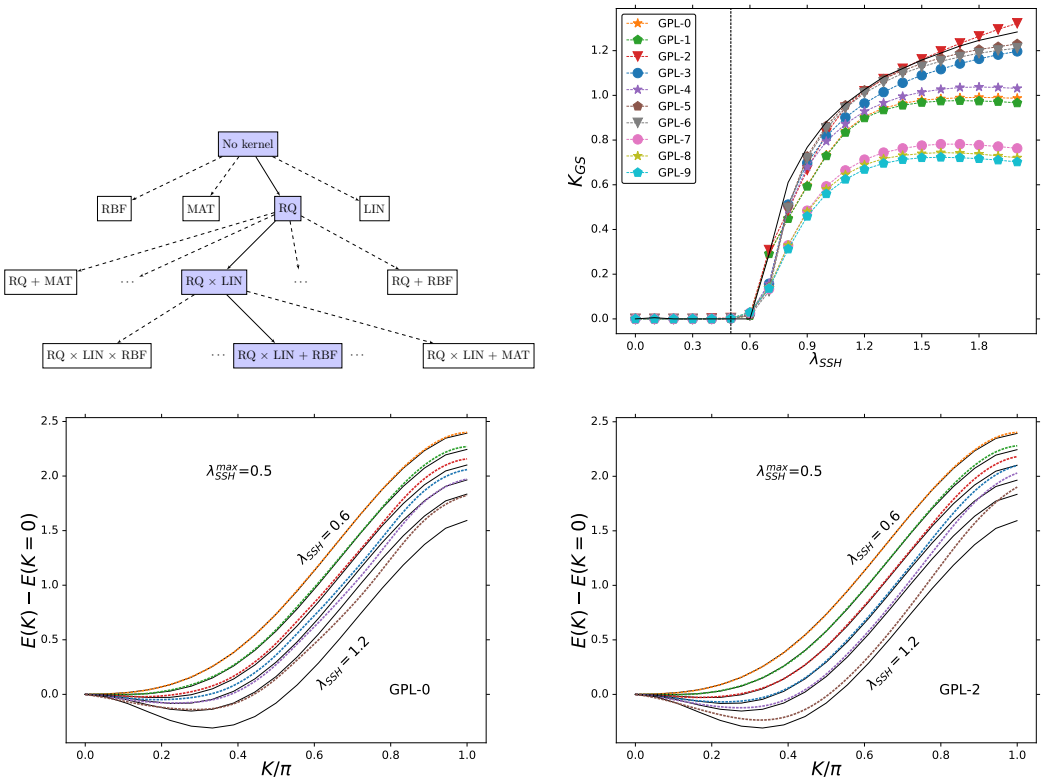

Fig. SM 5: Upper left: Schematic diagram of the kernel combinations with the highest BIC. Upper right: Effect of the increasing kernel complexity on the extrapolation accuracy. "GPL-X" denotes the results with the kernel obtained after X depth levels depicted in the upper left panel (e.g. GPL-0 is k_{RQ} and GPL-1 is $k_{RQ} \times k_{LIN} + k_{RBF}$). Lower panels: polaron dispersions predicted by the GP model with the kernel k_{RQ} (left panel) and kernel $k_{RQ} \times k_{LIN} + k_{RBF}$ obtained at the GPL-2 level (right panel). The dashed curves show the GP model predictions, while the solid curves are the results from Ref. [6]. The GP models are trained by the quantum results at $\lambda_{SSH} \leq 0.5$.

space representation. Therefore, finite size effects have no relevance.

The MA data used in this work are of the three-site variational flavor and have been confirmed to be in quantitative agreement with numerically exact results. The SSH polaron results were verified against the Bold Diagrammatic Quantum Monte Carlo results in Ref. [6], whereas more complicated extensions have been verified against the Variational Exact Diagonalization in Refs. [13–15].

The polaron energy is obtained from the lowest discrete peak in the imaginary part of the Green's function.

B. Mean-free energy of the Heisenberg model

Here we present the derivation of the dimensionless mean-field free energy density of the Heisenberg model we study with the GP method to predict the transition from ferromagnetic to paramagnetic phase. The Heisenberg model Hamiltonian reads

$$\mathcal{H} = -\frac{J}{2} \sum_{\langle i,j \rangle} \bar{S}_i \cdot \bar{S}_j, \tag{12}$$

where $\langle i, j \rangle$ only account for nearest-neighbour interactions between different spins $\bar{S}_i$. The free energy of the Heisenberg model in the mean-field approximation is a function of the magnetization m and the temperature T [16],

$$F(m,T) = \frac{JzNm^2}{2(g\mu_B)^2} - NT \ln \left[2 \cosh \left(\frac{Jzm}{2Tg\mu_B} \right) \right], \tag{13}$$

where m is defined as $m = g\mu_B \langle \bar{S}_i \rangle$ and z is the coordination number. The Boltzmann constant is set to 1 throughout this section.

Taylor expanding $F(m,T)$ near the transition, where m is vanishingly small, we obtain

$$F(m,T) = \frac{JzNm^2}{2(g\mu_B)^2} - NT\left[\ln(2) + \frac{1}{2}\left(\frac{Jz}{2Tg\mu_B}\right)^2 m^2 - \frac{1}{12}\left(\frac{Jz}{2Tg\mu_B}\right)^4 m^4 + \cdots\right]. \tag{14}$$

To find the critical transition temperature T_c, we minimize $F(m,T)$: $\frac{\partial F}{\partial m} = 0$. Solving graphically, we obtain $T_c = \frac{Jz}{4}$ [16]. We then divide $F(m,T)$ by N and T_c after subtracting $F(0,T)$ to obtain the shifted free energy density

$$f(m,T) = \frac{Jz}{2\left(g\mu_B\right)^2}\left[1 - \frac{T_c}{T}\right] m^2 + \frac{4}{3\left(g\mu_B\right)^4}\left(\frac{T_c}{T}\right)^3 m^4. \tag{15}$$

The last step is to define the dimensionless magnetization $\tilde{m} = \frac{2m}{g\mu_B}$, yielding

$$f(\tilde{m},T) = \frac{1}{2}\left[1 - \frac{T_c}{T}\right]\tilde{m}^2 + \frac{1}{12}\left(\frac{T_c}{T}\right)^3 \tilde{m}^4 \tag{16}$$

This is Eq. (12) in the main text, where the tilde over m has been omitted to simplify the notation.

The shape of the magnetization dependence of the free energy density changes with T. At $\frac{T_c}{T} < 1$, the minimizer of $f(\tilde{m},T)$, *i.e.* the order parameter (here denoted as m_0) is $m_0 = 0$; while for $\frac{T_c}{T} > 1$, $m_0 \neq 0$. This is illustrated in Figure 4 of the main text for $Jz = 5$, *i.e.* $T_c = 1.25$. This form of $f(\tilde{m},T)$ can be equivalently obtained through the phenomenological Landau theory of phase transitions [16]. We use GP regression with kernel combinations to extrapolate the free energy density acrtoss the phase transition.

[1] C. E. Rasmussen, and C. K. I. Williams, *Gaussian Process for Machine Learning*. MIT Press, Cambridge (2006).
[2] D. K. Duvenaud, H. Nickisch, and C. E. Rasmussen, *Advances in Neural Information Processing Systems* **24**, 226 (2011).
[3] G. Schwarz, *The Annals of Statistics* **6**, 461 (1978).
[4] R. S. Sutton, and A. G. Barto, *Reinforcement Learning, An Introduction*. MIT Press, Cambridge (2016).
[5] D. K. Duvenaud, J. Lloyd, R. Grosse, J. B. Tenenbaum, and Z. Ghahramani, *Proceedings of the 30th International Conference on Machine Learning Research* **28**, 1166 (2013).
[6] D. J. J. Marchand, G. De Filippis, V. Cataudella, M. Berciu, N. Nagaosa, N. V. Prokof'ev, A. S. Mishchenko and P. C. E. Stamp, *Phys. Rev. Lett.* **105** , 266605 (2010).
[7] M. Berciu, *Phys. Rev. Lett.* **97**, 036402 (2006).
[8] M. Berciu and G.L. Goodvin, *Phys. Rev. B* **76**, 165109 (2007).
[9] G.L. Goodvin and M. Berciu, *Phys. Rev. B* **78**, 235120 (2008).
[10] C.P.J. Adolphs and M. Berciu, *Phys. Rev. B* **90**, 085149 (2014).
[11] F. Herrera, K. W. Madison, R. V. Krems and M. Berciu, *Phys. Rev. Lett.* **110** (22), 223002 (2013).
[12] M. Berciu and H. Fehske, *Phys. Rev. B* **82**, 085116 (2010).
[13] J. Sous, M. Chakraborty, C. P. J. Adolphs, R. V. Krems, and M. Berciu, *Sci. Rep.* **7**, 1169 (2017).
[14] J. Sous, M. Berciu, and R. V. Krems, *Phys. Rev. A* **96**, 063619 (2017).
[15] J. Sous, M. Chakraborty, R. V. Krems, and M. Berciu, arXiv:1805.06109.
[16] P. M. Chaikin, and T. C. Lubensky, *Principles of condensed matter physics*, Cambridge University Press, Cambridge (1998).