NEURAL OPERATORS FOR THE COMMITTOR PROBLEM*

BILL CHEN†

PROJECT ADVISOR: MARIA CAMERON‡

Abstract. Transition path theory is a widely used mathematical framework for quantifying rare noise-driven transitions between two metastable states A and B in systems modeled by stochastic differential equations. Central to this framework is the committor function, the solution to the stationary backward Kolmogorov equation with certain boundary conditions. The committor at a point x is the probability that the process starting at x will first reach metastable state B rather than A. In this work, an operator learning approach proposed in Li et al. (2020) is investigated in the context of the committor problem for overdamped Langevin dynamics. The parameters chosen for the numerical tests are relevant to applications in chemical physics, including the temperature which controls the noise amplitude and the parameters which regulate the potential energy landscape. The solution operator to the committor problem with these parameters is represented as a Fourier Neural Operator. This approach yields a family of solutions allowing the user to evaluate the committor at a whole range of parameters values, in contrast to standard numerical methods. Accuracy and efficacy of the approach are demonstrated on a number of benchmark test problems.

Key words. Transition Path Theory, Committor, Neural Network, Neural Operator

AMS subject classifications. 68T07, 35Q92, 65C30

1. Introduction. In recent years, the field of scientific computing has been transformed by the introduction and rapid development of neural network-based partial differential equation (PDE) solvers such as physics-informed neural networks (PINNs) [11] and variational neural networks [6, 9]. These formulations often are more efficient than traditional methods. For instance, a PDE depends on parameters that alter its solution, such as the viscosity of a fluid or the temperature of the state space. Traditional methods for solving partial differential equations require reapplication of numerical methods for each parameter value, which can be computationally expensive [10].

However, a neural-network based framework known as operator learning resolves this issue by solving PDEs for a family of parameter values [10]. First, the solution operator is represented by means of a neural network which is trained to match the solution of the PDE at a number of parameter values, using training data that is generated by a high-accuracy method. Importantly, the neural network in [10] has linear layers interlacing the nonlinear activation functions in a certain form motivated by the Fourier transform. Second, as the solution operator is learned, it can then be evaluated at a wide range of parameter value to yield the desired solution. This approach was tested on 1-D Burgers equation, 2-D Darcy flow, and 2-D Navier-Stokes equation in [10]. In both 2-D cases, the spatial domain was the unit square.

The committor problem, which arises from the study of quantifying rare transitions in systems governed by stochastic differential equations, is both an interesting and important test problem for operator learning. Solving the committor problem is crucial for the application of the widely used transition path theory framework [4] to quantify rare events in stochastic systems. Crucially, once the committor is found, the transition channels and transition rate can be calculated. The committor problem is also challenging, as it is the solution to the boundary-value problem for the

*Submitted to the editors 9/15/2024.

Funding: Acknowledgement: NSF REU grant DMS-2149913

†NORTHWESTERN UNIVERSITY (BILLCHEN2025@U.NORTHWESTERN.EDU).

‡UNIVERSITY OF MARYLAND, COLLEGE PARK (MARIAKC@UMD.EDU).

stationary backward Kolmogorov equation with a mix of Dirichlet inhomogeneous and Neumann homogeneous boundary conditions. Put in the self-adjoint form, the stationary backward Kolmogorov equation for the overdamped Langevin dynamics takes the form $\nabla \cdot (\exp[-\beta V(x)]\nabla q) = 0$ where β is the inverse temperature and $V(x)$ is the potential energy function that often varies widely throughout the phase space of the system and tends to infinity as $\|x\| \to \infty$ fast enough such that the invariant probability density exists. Therefore, the natural computational domain is a sublevel set $\{x \mid V(x) < C\}$ for some large enough constant C with removed disjoint regions A and B between which the transitions need to be quantified. Hence, the computational domain is nonconvex and not rectangular, introducing difficulty.

In this work, we adapt and test the operator learning to the committor problem for the overdamped Langevin dynamics. We choose the inverse temperature β and terms controlling the extent of ruggedness of the rugged Mueller potential (a popular test problem [6, 9, 12]) as the parameters for the solution operator to the final test problem. We demonstrate that the learned operator can learn highly accurate solutions for a range of parameter values.

The rest of the paper is organized as follows. The necessary background on transition path theory and neural networks is given in Section 2. The theoretical foundations for operator learning are described in Section 3. The numerical tests are presented in Section 4. The conclusions are summarized in Section 5.

2. Background.

2.1. Transition Path Theory. In this work, we will use the overdamped Langevin dynamics as the basic model. The overdamped Langevin equation is a common model for molecular motion:

$$dx = -\nabla V(x)dt + \sqrt{2\beta^{-1}}dw, \tag{2.1}$$

where

- V(x) is a smooth function interpreted as the potential energy,
- β^{-1} is the temperature in the units of Boltzmann's constant,
- w is the standard Brownian motion.

The invariant probability measure for equation (2.1) is the Gibbs density:

$$\rho(x) = Z^{-1}e^{-\beta V(x)}, \quad Z = \sum e^{-\beta V(x)} \tag{2.2}$$

In transition path theory (TPT) [4, 5], two disjoint regions A and B are chosen, which usually surround minima of the potential function $V(x)$ between which transitions are studied (see 1).

The key function of TPT is the committor $q(x)$, defined as the probability that the process starting at x will reach first B rather than A, or:

$$q(x) = \mathbb{P}(\tau^B < \tau^A \mid x_0 = x) \tag{2.3}$$

where τ^A and τ^B are the first hitting times of A and B and x_0 denotes the initial point of the process. The committor can be written as the solution to the boundary value problem for the backward Kolmogorov equation:

$$\begin{cases} -\nabla V \cdot \nabla q + \beta^{-1}\Delta q = 0, & x \in \Omega \\ q = 0, & x \in \partial A, \\ q = 1, & x \in \partial B. \end{cases} \tag{2.4}$$

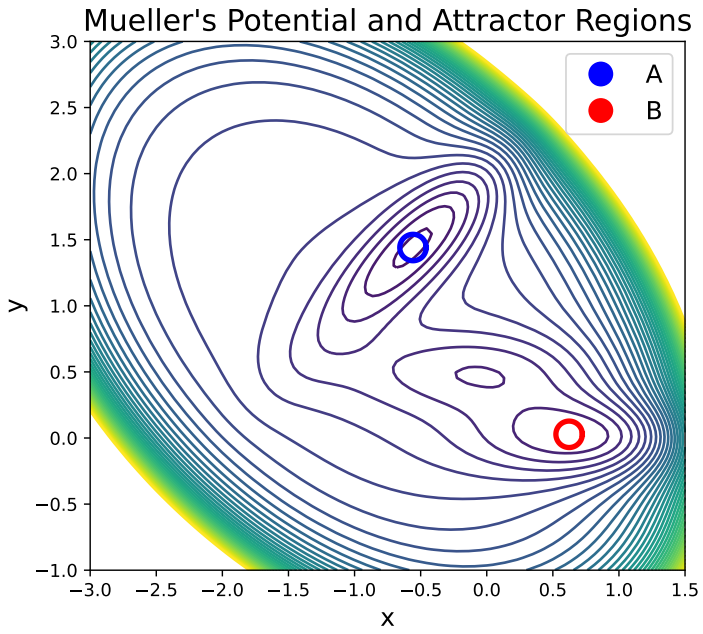


Fig. 1: The regions A and B surround the two deepest minima of Mueller's Potential.

The equivalent adjoint representation of the equation uses the invariant probability density as follows:

$$\begin{cases} \beta e^{\beta V} \nabla \cdot (e^{-\beta V} \nabla q) = 0, & x \in \Omega \\ q = 0, & x \in \partial A \\ q = 1, & x \in \partial B \end{cases} \tag{2.5}$$

Once the committor is computed, one can readily calculate the transition current and the transition rate [4, 5].

2.2. Neural Networks. In general, a neural network is a composition of functions in which affine mappings are interlaced with nonlinear activation functions applied entrywise. In this work, we will use neural networks of the form

$$\mathcal{N}(x) = \mathcal{L}_{L+1} \circ \cdots \circ \sigma \circ \mathcal{L}_1, \tag{2.6}$$

defining $\mathcal{L}_j = A_j \phi_\theta(x) + b_j$, where ϕ is an operator with parameters θ, A_j is a weight matrix, b_j is a bias vector, and σ is an nonlinear activation function (e.g. ReLU, tanh, etc.) applied entrywise. The size of a neural network refers to the number of trainable parameters and the depth refers to the number of activation functions.

The trainable parameters of the model are the weights A_j, bias terms b_j, and the parameters θ. A loss function is defined to evaluate the performance of the neural

network. Training a neural network refers to solving the minimization problem in the parameter space with respect to the loss function. Optimization algorithms such as stochastic gradient descent or ADAM are typically used applied for this purpose [7].

3. Operator Learning and Fourier Neural Operator.

3.1. Operator Learning Framework. Throughout this paper, we define a neural operator using the same framework as [8, 10].

Let $D \subset \mathbb{R}^d$, $\mathcal{A} = \mathcal{A}(D; \mathbb{R}^{d_a})$, $\mathcal{U} = \mathcal{U}(D; \mathbb{R}^{d_u})$, where D is the domain of $\mathcal{A}$ and $\mathcal{U}$, function spaces defining the coefficient function space and solution space for the PDE respectively, d is the dimension of D, d_a is the dimension of the codomain of $a \in \mathcal{A}$, and d_u is the dimension of the codomain of $u \in \mathcal{U}$.

We consider a family of PDEs with functions of the form $f(x, u, \mathcal{D}u, \mathcal{D}^2u, \{a_j(x)\}_{j=1}^{d_a})$ where $\mathcal{D}^j u,\ j = 1, 2$ denotes the sets of partial derivatives of order j of u and $\{\{a_j(x)\}_{j=1}^{d_a}\}$ denotes the set of coefficient functions of the PDE. Let $\mathcal{U}$ be the solution space of the PDE and let $G : \mathcal{A} \to \mathcal{U}$ be the solution operator.

Frequently, only a finite sub-sample of a_j, u_j are available. Therefore, let $D_n = \{x_1, \ldots, x_n\}$ be a discrete set of points (mesh) such that a_j, u_j are known for any $x_i \in D$. This reflects the limitations of measurements and simulations when using data.

3.2. Architecture. A neural operator is a specific architecture of a neural network. We consider the neural operator defined on $(\mathcal{A},\mathcal{U})$, where the neural operator is trained to emulate the solution operator G. Define the architecture of a neural network $\mathcal{N}$ as follows:

$$\mathcal{N}(a) = Q \circ \mathcal{L}_L \circ \cdots \circ \mathcal{L}_1 \circ P \tag{3.1}$$

where P is a projective layer from $\mathbb{R}^{d_a} \to \mathbb{R}^{d_v}$, $d_v > d_a$, and Q is a projective layer from $\mathbb{R}^{d_v} \to \mathbb{R}^{d_u}$. $\mathcal{L}$ is the kernel layer defined as $\sigma(W\nu_t x + \mathcal{K}(a; \phi)\nu_t(x))$, where σ is an activation function, W is a weight matrix, K is the parametrized operator that maps $\mathcal{A}$ into the set of bounded linear operators on D to $\mathcal{V}$, i.e. $K : \mathcal{A} \times \Theta \to L(D, \mathcal{V}), \mathcal{V} = \mathcal{V}(D, \mathbb{R}^{d_v})$, where $\mathcal{V}$ is a function space defined analogously to $\mathcal{A},\mathcal{U}$ from before, which can be thought of as an intermediate stage between $\mathcal{A}$ and $\mathcal{U}$.

A neural operator maps the coefficient a_j through L layers, where the output of a_j after being mapped through t layers is denoted as v_t. v_t is recursively defined as $v_{t+1} = \mathcal{L}_{t-1}(v_t)$.

The standard form of K is an integral operator of the form $(\mathcal{K}(a; \theta)v_t)(x) = \int_D k_\theta(x, y; a(x), a(y))v(y)dy$, where k is a kernel function parameterized by θ. For brevity, we write $\mathcal{K}(a; \theta)$ as $\mathcal{K}_\theta(a)$.

3.3. Fourier Neural Operator. The Fourier Neural Operator (FNO) is a specific form of a neural operator, in which the kernel operator K is specified to be a convolution operator evaluated in Fourier space [10]. K takes the Fourier Transform of the data, performs the convolution (which in Fourier space is equivalent to multiplication), and then takes the inverse Fourier Transform of the result. K may also be interpreted as a convolutional neural network in Fourier space.

Let $k_\phi(x, y; a(x), a(y)) = k_\theta(x - y)$, meaning the kernel operator is parametrized as $\mathcal{K}_\theta v(x) = \int k_\theta(x - y)v(y)dy$. Let $\mathcal{F}$ denote the Fourier Transform and $\mathcal{F}^{-1}$ denote the inverse Fourier Transform. Then $\mathcal{K}_\theta(v(x)) = \mathcal{F}^{-1}((P_\theta \cdot (\mathcal{F}\nu_t))$. where $P_\theta = \mathcal{F}(k_\theta)$. Each $\mathcal{L}$ (layer) of the Fourier Neural Operator is written as $\sigma(W\nu_t + \mathcal{F}^{-1}(P_\theta \cdot \mathcal{F}(\nu_t)))$.

3.4. Implementation. The kernel function k admits a Fourier basis. We take at most k_{max} modes of this function, which gives a multi-dimensional representation of P_θ. As discussed previously, values of the function $a_j \in \mathcal{A}$ typically are only known for a mesh. The Fourier Transform of functions defined on this data may be evaluated via the Fast Fourier Transform. Since P_θ has k_{max} terms, we also take only the k_{max} modes of the Fourier Transform of v_t.

3.5. Advantages of Fourier Neural Operator. The Fourier Neural Operator has the following advantages for scientific computing [8, 10].

- Complexity:
  The source of computational complexity of the Fourier Neural Operator is largely dominated by the complexity of the Fourier Transform ($\mathcal{O}(n \log n)$), which is more efficient than directly evaluating the convolution operator.
- Data-driven:
  Training does not require a parameterized model of the underlying differential equation, but rather only a set of data generated according to the underlying equations.
- Operator Learning:
  The model is able to predict solutions for different parameters from the same general family, in contrast to finite element methods which must generate solutions for each set of parameter values.
- Mesh-invariant:
  Via the Fourier projection, the neural network is able to approximate global functions from projection along the Fourier basis. This property can be observed empirically. Additionally, Fourier Neural Operators display high accuracy with coarse meshes (relatively few points in the mesh), and can also be successfully trained for solutions with more detailed meshes than the training data possesses.
- Approximation Theory: It was derived in [8] that the Fourier Neural Operator can approximate continuous operators with ϵ accuracy. That is, by expanding the size or depth of the neural network, the maximum of the difference between the FNO and the solution at any point is bounded by an arbitrary ϵ. In addition, it can be shown that the size (number of parameters) of a Fourier neural network for the Darcy Flow PDE scales sub log-linearly with ϵ [8]. In particular, the similarity of the Darcy Flow PDE with the committor problem suggests optimism for similar performance.

4. Computational Results. In this section, we detail the results of using the Fourier Neural Operator on various test cases. The average test error is reported in each table. We take the perspective of learning the solution to different parameters by varying the input function along several parameters. The input functions are defined to be functions that contain information about the relevant parameters of the test problem, defined in the same general area as the mesh. The input functions can be thought of as the coefficient functions $a \in \mathcal{A}$. Training data solutions are generated using traditional numerical methods with high accuracy. The parameters of interest in the committor problem for the overdamped Langevin SDE are the inverse temperature β and the parameters of the potential function $V(x)$ (if any exist).

All results are computed using PyTorch via the neural operator Python package on an AMD Ryzen 5 4600U processor, with modifications to the neural network architecture as appropriate [1] [3]. Data for test problems was generated via M. Cameron's transition path theory finite element mesh code [2]. My implementation

Training Epochs	l_1/MAE	RMSE
500	.0017	.0001
1000	.0010	3.9577e-05
2000	.0008	2.7614e-05

Table 1: Double Well test errors, 200 Samples

can be found at [3].

In general, the specifications used for the neural network are $\sigma = \text{gelu} = x\Phi(x)$, where Φ is the cumulative distribution function of $N(0,1)$, Cosine Annealing Scheduler (varies learning rate in a periodic manner),ADAM optimizer [7], 4 Fourier Layers, Tucker Factorization, 16 Fourier modes, h_1 Sobolev norm loss.

For the committor function, we add a final sigmoid layer, to ensure that the output is contained in $[0,1]$. Various loss functions are used to evaluate the model, including

$$\text{MAE (mean absolute error)} = \sum_{\text{all FEM points}} |q_{FNO}(x_j) - q_{FEM}(x_j)| \tag{4.1}$$

$$\text{RMSE (root mean absolute error} = (\sum_{\text{all FEM points}} (q_{FNO}(x_j) - q_{FEM}(x_j))^2)^{1/2} \tag{4.2}$$

$$\text{wMAE (weighted mean absolute error)} = \sum_{\text{all FEM points}} w(x_j) * |q_{FNO}(x_j) - q_{FEM}(x_j)| \tag{4.3}$$

The wMAE is a special loss function adapted to the committor problem, with higher weight placed on points where the transition is uncertain. The weights are given by $w(x_j) = exp(-\beta V(x_j))q_{FEM}(x_j) * (1 - q_{FEM}(x_j))$, where $w(x_j)$ is then normalized via $\sum w_j$ [14].

4.1. Test Problem: Double Well Potential. The first test problem is a 1-dimensional committor problem, where the potential function is defined as

$$V(x) = (x^2 - 1)^2. \tag{4.4}$$

The potential function has local minima at $x = \pm 1$ and a local maximum at $x = 0$. The regions A and B are defined as $A = \{x \leq -1\}$ and $B = \{x \geq 1\}$. 1000 values of β are selected in the interval [0.10] in increments of .01. The input function is the constant β.

The committor for those values of β are solved using the Chebyshev Spectral Method [13], which obtains a machine-precision solution. The computed solution represented as a Chebyshev sum can be evaluated at any point using Clenshaw's method. For more details, see [12] for a similar 1-dimensional committor test case.

We randomly select 20 samples from the above set and use the committor functions for those input functions as the training set, and do the same for 200 samples as the test set on which the performance of the model is evaluated. The deterministic Adam optimizer is set to learning rate $\eta = 8e - 5$, and no batching is used.

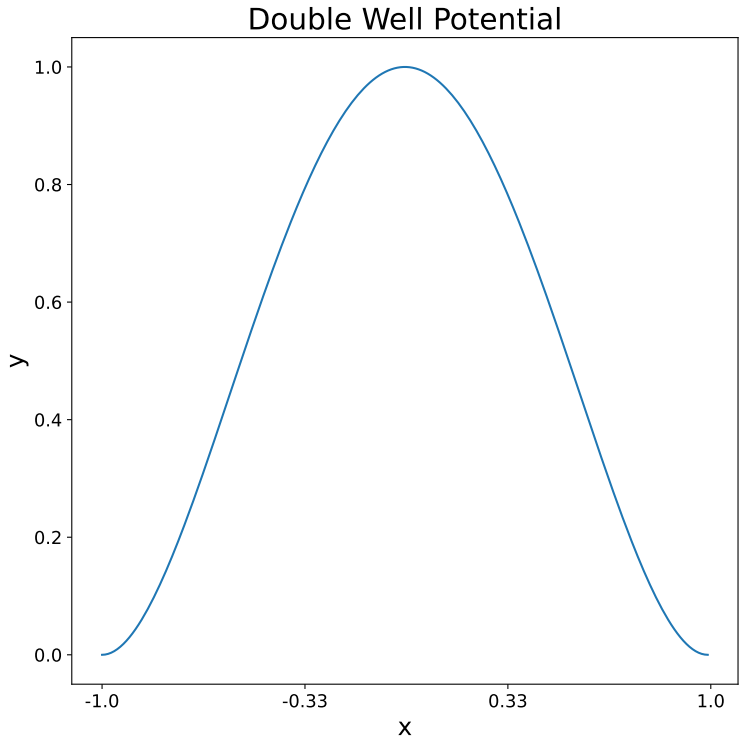


Fig. 2: Graph of the Double Well Potential (4.4)

As seen in (1), the test errors are low after training, indicating that the FNO model is able to solve differential equations with noises (i.e. stochastic differential equations).

4.2. Test Problem: Parametric Double Well. The next test problem deals with a two-parameter family of 1-dimensional committor problems on the interval $[-1, 1]$. The first parameter is the inverse temperature β, while the second parameter is the position of the maximum of the double-well potential (In the previous example, the maximum was fixed at $x = 0$). The double-well potential in this parametric family are twice continuously differentiable and are defined using cubic splines by constructing piecewise functions p_1, p_2 which satisfy the conditions

- $p_2(a) = 1, p_2'(a) = 0, p_2(1) = p_2'(1) = 0$
- $p_1(a) = 1, p_1'(a) = 0, p_1(-1) = p_1'(-1) = 0$

where a is the point where the maximum is achieved. These conditions give a similar shape to the double well potential, but with varying maxima. The functions p_1, p_2 satisfying these conditions are given by

(4.5)

$$p_1(x;a) = -\frac{2}{(a+1)^3}(x+1)^2\left(x - \frac{3a+1}{2}\right), p_2(x;a) = -\frac{2}{(a-1)^3}(x-1)^2\left(x - \frac{3a-1}{2}\right)$$

(defined on the interval $[-1, 1]$).

100 values of β are selected from the interval $[0, 10]$ in increments of .01, and

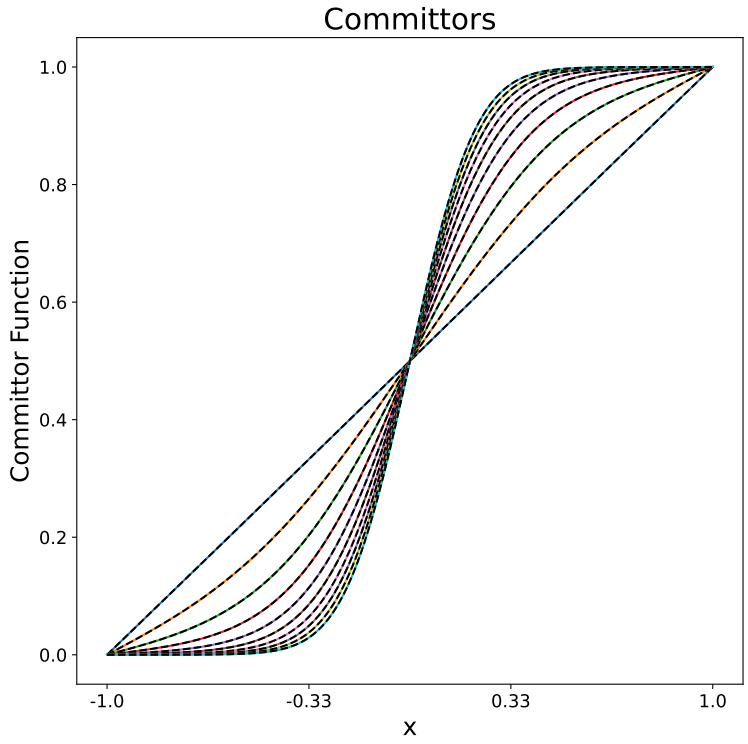


Fig. 3: Committor functions of the Double Well Potential graphed against dotted ground truth committors for equispaced values of β between 0 and 10 spaced by 1

Epochs	l^1/MAE	RMSE
500	.0082	.0005
1000	.0046	.0002
2000	.0040	.0004

Table 2: Parametric Double Well test errors, 384 Samples

18 values of a are selected from the interval $[-.9, .9]$ in increments of .1. Taking a too close to the endpoints $-1, 1$ results in sharp transitions. Each combination of values defines an input function $e^{-\beta V}$, which captures information about both β and a. As before, we use the Chebyshev Spectral Method to solve the committor for these values.

We randomly select 52 samples from the above set to train the model on, and 384 samples to test on. The deterministic Adam optimizer is set to learning rate $\eta = 8e - 4$, and no batching is used.

In table (2), the test errors are low after training, indicating that the FNO model is able to solve stochastic differential equations of increased complexity.

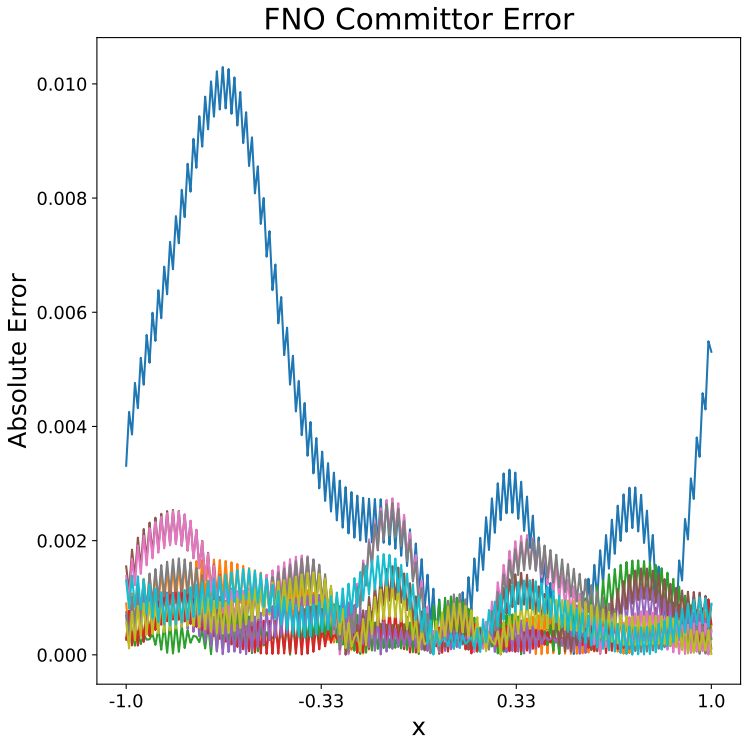


Fig. 4: The absolute error plot for equispaced values of β (Double Well)

4.3. Test Problem: Mueller's Potential. The next test problem is the committor problem with the 2-D Mueller's potential shown in (1) and listed in (4.6), which is often used as a test problem in the chemical physics literature, e.g. [12]. The regions A and B are the balls of radius .1 surrounding the two deepest local minima near $(-.57, 1.43)$ and $(.56, .044)$

$$V(x_1, x_2) = \sum_{i=1}^{4} D_i \exp(a_i(x_1 - X_i)^2 + b_i(x_1 - X_i)(x_2 - Y_i) + c_i(x_2 - Y_i)^2) \tag{4.6}$$

$$[a_1, a_2, a_3, a_4] = [-1, -1, -6.5, 0.7]$$
$$[b_1, b_2, b_3, b_4] = [0, 0, 11, 0.6]$$
$$[c_1, c_2, c_3, c_4] = [-10, -10, -6.5, 0.7]$$
$$[D_1, D_2, D_3, D_4] = [-200, -100, -170, 15]$$
$$[X_1, X_2, X_3, X_4] = [1, 0, -.5, -1]$$
$$[Y_1, Y_2, Y_3, Y_4] = [0, .5, 1.5, 1]$$

80 values of β selected in the interval $[.05, .2]$ in increments of .0015. The input function is the constant function β. The committor solutions are generated using the

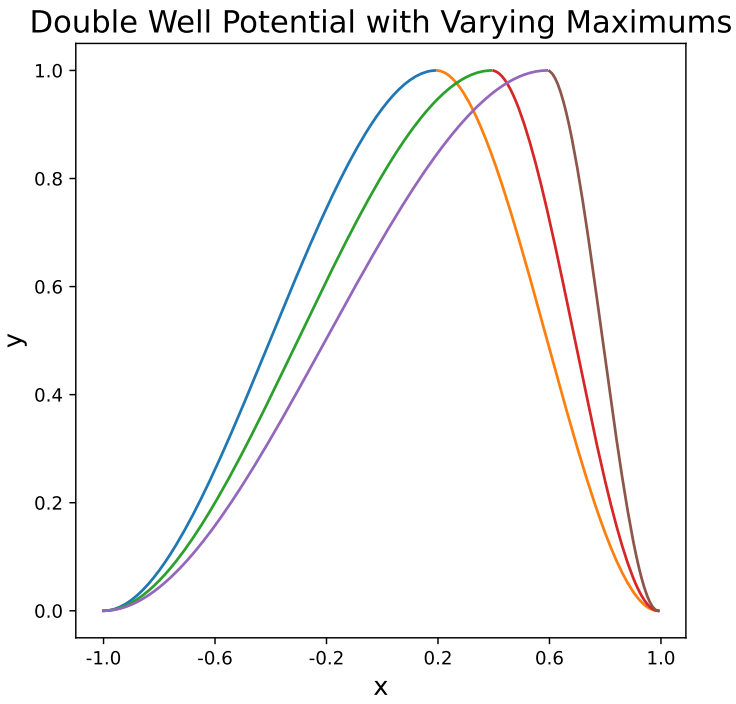


Fig. 5: Parametric Double Well Potentials, where a ranges over $.2, .4, .6$. The differing colors denote the component piecewise functions.

Epochs	l^1/MAE	wMAE
500	.0096	.0204
1000	.0048	.0089
1500	.0040	.0087
2000	.0035	.0071

Table 3: Mueller's Potential test errors, 20 Samples

finite element method (FEM) method [2] as the training data with a mesh has 6776 points and 13268 triangles.

The training set uses 80 samples and the test set uses 20 samples. The maximum error of the model solutions was approximately $\eta = 1e-4$. The deterministic Adam optimizer is set to learning rate $\eta = 8e-5$, and a batch size of 4 is used.

Again, from table (3), the test errors are low after training, indicating that the FNO model is able to solve stochastic differential equations in two-dimensional potential spaces, compared to one. One additional observation is that the wMAE is higher than the MAE, which indicates that the model has more difficulty learning the committor near transition points (i.e. q is near .5).

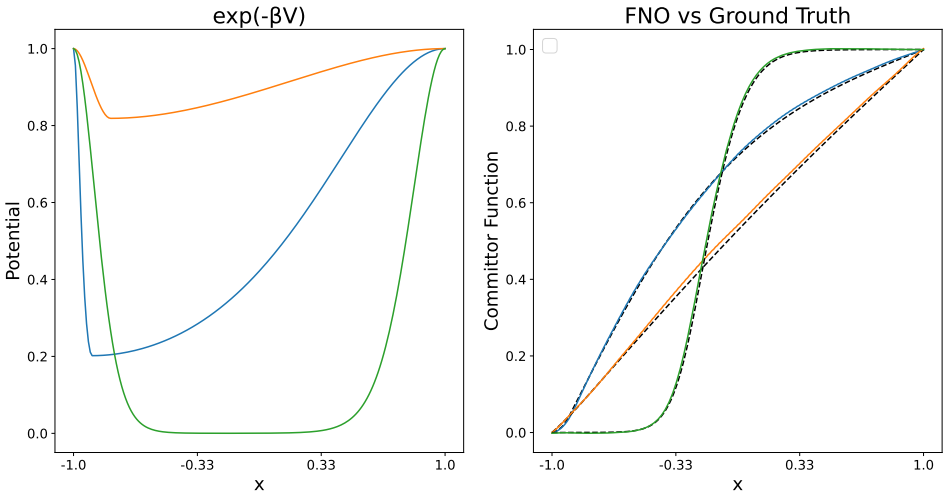
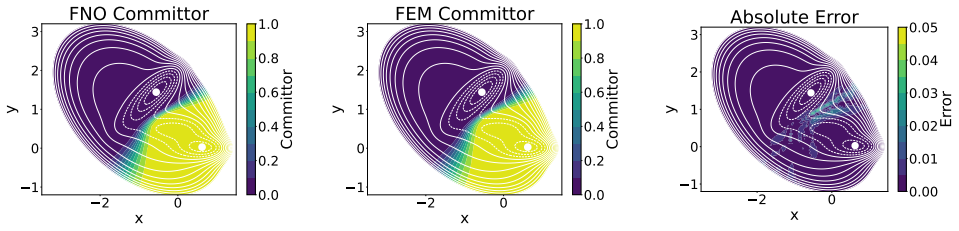


Fig. 6: A plot of the input function $\exp(-\beta V)$ with V from (4.5) and a plot of the FNO committor with dotted-line Ground Truth committors (Parametric Double Well)



(a) Committor via FNO (b) Committor via FEM (c) Absolute Error.

Fig. 7: Comparison of committors generated for $\beta = .18$ (Mueller's Potential)

4.4. Test Problem: A Parametric Rugged Mueller's Potential. A even more challenging test problem is the Rugged Mueller's potential (4.7), which is the sum of Mueller's potential and a periodic oscillatory function. The attractor regions similarly to the Mueller's Potential Test Problem. This test problem was also used in a number of recent works featuring the numerical solution to the committor problem via various machine learning techniques [12, 9]

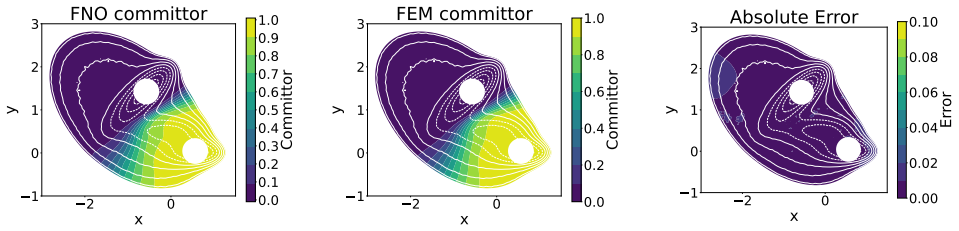
$$V(x_1, x_2; \gamma, k) = V_0(x_1, x_2) + \gamma \sin(2k\pi x_1) \sin(2k\pi x_2) \tag{4.7}$$

where $V_0(x_1, x_2)$ is Mueller's potential (4.6) The parameters γ, k control the amplitude and the frequency of the ruggedness, respectively. The mesh consists of 4611 points, with 8924 triangles. Input functions are defined as $e^{-\beta V}$, with the corresponding

Epochs	l^1/MAE	wMAE
5628	3.9173e-05	.0078
6000	4.3385e-0	.0087
6020	3.2658e-05	.0065

Table 4: Rugged Mueller's Potential test errors, 200 Samples

committors solved using FEM once again [2]. For the data, we take the (FEM) solutions from random samples with 10 values of β in the interval $[.05, .2]$ in increments of .015, 25 values of γ in the interval $[0, 10]$ in increments of .4, and 2 values of k in the interval $[4, 6]$ in increments of .5. 200 samples from the set above are used as a training dataset, and 800 are used as a test dataset.



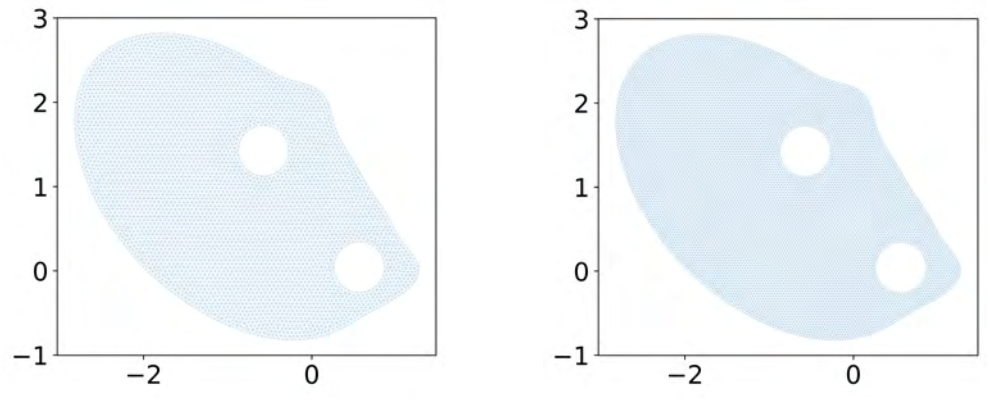
(a) Committor via FNO (b) Committor via FEM (c) Absolute Error.

Fig. 8: Comparison of committors generated for $\beta = .13$, $\gamma = 5.8$, $k = 3.6$ (Rugged Mueller's Potential)

In table (3), the error of the committor is higher compared to Mueller's potential, especially when β and γ are relatively large. Nevertheless, it is still comparable to the errors obtained in [6]. [6] obtains MAE of .02 to .05, depending on the sampling method and data size. Overall, the low test errors indicate that the FNO is able to solve stochastic differential equations with complex two-dimensional potentials.

In addition, the FNO model experiences additional difficulty when learning points near the transition area of the committor, as shown by the relatively high wMAE in comparison. Another note is that the training process on the laptop (AMD Ryzen 5 4600U processor) took a rather long time, around 8 hours. Nonetheless, once the training is completed, one can resample the committor function at any set of parameters (β, γ, k) in the range and expect an accurate result.

4.5. Mesh Invariance. As noted earlier, the FNO model is mesh-invariant. Practically, this means that the FNO can be trained on coarse meshes, and then be applied to find solutions with more detailed meshes. One potential concern is that when spectral methods are applied to complex domains, spurious oscillations may arise (for example, due to the Gibbs phenomenon). Although such oscillations are not observed in the testing/training resolution of (8), for instance, it is interesting to see if such oscillations or instabilities are still suppressed in higher resolution, and in



(a) Mesh with 4611 Points

(b) Mesh with 9240 Points

Fig. 9: Comparison of resolutions

general to observe the performance of the FNO when utilizing mesh invariance.

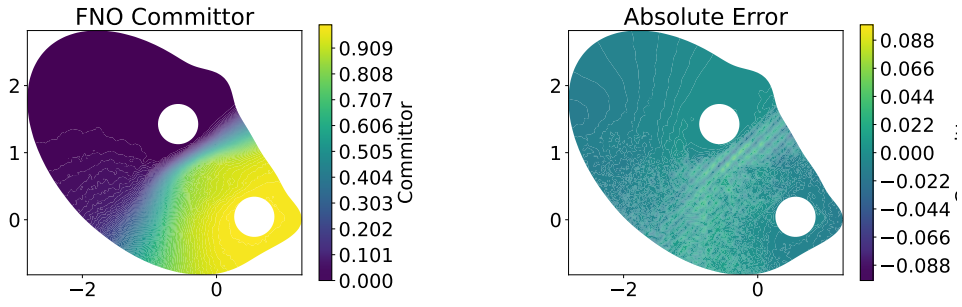
To investigate this, a case-study on this property is presented with the Rugged Mueller's potential. The size of the mesh used to generate the committor to the Rugged Mueller's potential is governed by a scaling parameter h. By adjusting it from $h = .05$ to $h = .035$, the size of the mesh increases from 4611 points to 9240 points, approximately doubling the resolution, as seen in (9). Parameter values that were not trained on, along with data generated using this higher resolution mesh, are used to evaluate the model. Note that an abridged version of the committor diagrams are presented, since we are mainly concerned about the qualitative behavior of the committor. We observe that in (10), little to no instability are present, but in (11), some instability can be detected. Although instability/higher errors in unseen data for the Rugged Mueller's potential is not unexpected (due to the complexity of the underlying potential), the instability appears to be more severe than seen in (8), for instance. Further investigation of these phenomenon to ascertain the source of this instability may yield more concrete insight here.

5. Conclusions. In conclusion, we have demonstrated Fourier Neural Operator is able to approximate committors with high accuracy, even when solving for the solution operator for more than one parameter and in two dimensions.

There are several optimizations and theoretical results that remain to be explored. Adequate results were observed on nonuniform meshes, but further research and implementation of a nonuniform fast Fourier transform and its inverse, although not well documented, may increase speed and error on nonuniform applications such as the committor.

A natural extension of the Fourier method would be to represent functions as wavelets, and evaluate them through some type of convolutional neural network analogous to an FNO. An advantage of this process would be that wavelets can represent irregular/sharp functions with fewer terms than Fourier terms (such as in the Gibbs phenomenon).

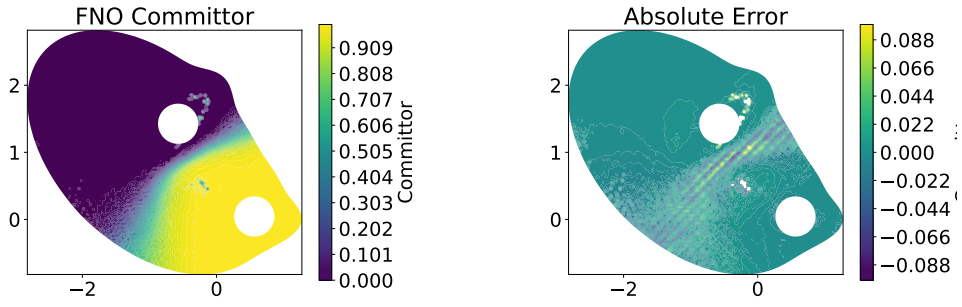
Another worthwhile area of research would be deriving a bound on the scaling of the size or depth of the neural operator for the specific case of the neural operator,



(a) Committor via FNO

(b) Absolute error

Fig. 10: High resolution FNO output and error for $\beta = .075, \gamma = 10.5, k = 4.9$ (Rugged Mueller's)



(a) Committor via FNO

(b) Absolute error

Fig. 11: High resolution FNO output and error for $\beta = .15, \gamma = 10.5, k = 6.1$ (Rugged Mueller's)

akin to [8] for the similar Darcy Flow equation. [8].

Additionally, exploring the theoretical guarantees and practical performance of the FNO's mesh invariance property may also yield insights that enhance the computational efficiency of the FNO, as briefly explored in (4.5).

Appendix A. Approximation Theory. One major component of this project was attempting to develop an approximation theory for the Fourier Neural Operator. No formal result was proven because of the mathematical sophistication required, but some intuition for a general outline of a proof was achieved via [8].

- Design a spectral method for the committor. One such method could potentially be via diffusion maps
- Prove that a Fourier Neural Operator can approximate each step of the method with some bound on size/depth of the neural network
- Compose these results to have an emulation of FNOs for a spectral method, which gives an efficient approximation theory result.

Acknowledgments. The author thanks Professor Maria Cameron, my fellow REU students in the stochastic systems project at UMD, the teaching assistants (Shashank Sule, Perrin Ruth, Luis Suarez), as well as two anonymous referees for valuable feedback and discussions. This work was supported by the NSF REU grant DMS-2149913.

REFERENCES

[1] *Neural operator code*, https://github.com/neuraloperator/neuraloperator.
[2] M. Cameron, *Transition path finite element mesh code*, https://github.com/mar1akc/transition_path_theory_FEM_distmesh/tree/main.
[3] B. Chen, *My fourier neural operator implementations/code*, https://github.com/bill1006/CommittorFNO.
[4] W. E and E. Vanden-Eijnden, *Towards a theory of transition paths*, Journal of Statistical Physics, 123 (2006).
[5] W. E and E. Vanden-Eijnden, *Transition-path theory and path-finding algorithms for the study of rare events*, Annual Review of Physical Chemistry, 61 (2010).
[6] Y. Khoo, J. Lu, and L. Ying, *Solving for high dimensional committor functions using artificial neural networks*, CoRR, abs/1802.10275 (2018), http://arxiv.org/abs/1802.10275, https://arxiv.org/abs/1802.10275.
[7] D. P. Kingma and J. Ba, *Adam: A method for stochastic optimization*, Computing Research Repository, abs/1412.6980 (2014).
[8] N. Kovachki, S. Lanthaler, and S. Mishra, *On universal approximation and error bounds for fourier neural operators*, Journal of Machine Learning Research, (2021).
[9] Q. Li, B. Lin, and W. Ren, *Computing committor functions for the study of rare events using deep learning*, The Journal of Chemical Physics, 151 (2019), https://aip.scitation.org/doi/10.1063/1.5110439.
[10] Z. Li, N. Kovachki, K. Azizzadenesheli, B. Liu, K. Bhattacharya, A. Stuart, and A. Anandkumar, *Fourier neural operator for parametric partial differential equations*, International Conference on Learning Representations, (2021).
[11] M. Raissi, P. Perdikaris, and G. E. Karniadaki, *Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations*, Journal of Computational Physics, 378 (2019), pp. 686–707.
[12] Z. Song, M. K. Cameron, and H. Yang, *A finite expression method for solving high-dimensional committor problems*, arXiv, (2023).
[13] L. N. Trefethen, *Spectral Methods in MATLAB*, SIAM, 2000.
[14] J. Yuan, A. Shah, C. Bentz, and M. Cameron, *Optimal control for sampling the transition path process and estimating rates*, arXiv, (2023).