MODELLISTICA E CALCOLO SCIENTIFICO

MOX

POLITECNICO MILANO

MODELING AND SCIENTIFIC COMPUTING

MOX-Report No. 32/2024

Learning epidemic trajectories through Kernel Operator Learning: from modelling to optimal control

Ziarelli, G.; Parolini, N.; Verani, M.

Learning epidemic trajectories through Kernel Operator Learning: from modelling to optimal control

Giovanni Ziarelli[a], Nicola Parolini[a], Marco Verani[a]

[a]*MOX, Dipartimento di Matematica, Politecnico di Milano, Piazza Leonardo da Vinci 32, Milano, 20133, Italy*

Abstract

Since infectious pathogens start spreading into a susceptible population, mathematical models can provide policy makers with reliable forecasts and scenario analyses, which can be concretely implemented or solely consulted. In these complex epidemiological scenarios, machine learning architectures can play an important role, since they directly reconstruct data-driven models circumventing the specific modelling choices and the parameter calibration, typical of classical compartmental models. In this work, we discuss the efficacy of Kernel Operator Learning (KOL) to reconstruct population dynamics during epidemic outbreaks, where the transmission rate is ruled by an input strategy. In particular, we introduce two surrogate models, named KOL-m and KOL-∂, which reconstruct in two different ways the evolution of the epidemics. Moreover, we evaluate the generalization performances of the two approaches with different kernels, including the Neural Tangent Kernels, and compare them with a classical neural network model learning method. Employing synthetic but semi-realistic data, we show how the two introduced approaches are suitable for realizing fast and robust forecasts and scenario analyses, and how these approaches are competitive for determining optimal intervention strategies with respect to specific performance measures.

Author Summary

During epidemic outbreaks, computational models are of paramount importance to provide policy-makers with short-term predictions and scenario analyses. In this respect, machine learning techniques leverage on available data to build computational models suitable for different tasks. Along this line, in this work we propose two approaches based on Kernel Operator Learning (KOL) to approximate the functional relation between plausible restrictions of the transmission rate of the disease and the corresponding evolution of susceptible, infectious, deceased and recovered individuals. We numerically show that our approaches outperform a classical neural

network learning scheme, in terms of trade-off between computational cost and reliability of the results. Moreover, we test the two KOL approaches for optimization tasks (searching for optimal interventions for the minimum eradication and for the minimization of total infected). Finally, the performances of the two proposed KOL methods are compared for different choices of the kernel function. The choice of the Neural Tangent Kernel turns out to be the most effective.

1. Introduction

The recent global SARS-CoV-2 pandemic has underlined the paramount importance of developing mathematical models and numerical schemes for predictions and forecasts of epidemic illnesses: from the perspective of policy-makers, it is often useful to dispose of qualitative and quantitative results for making scenario analyses and forecasts; from the social point of view sharing information about possible outcomes can be beneficial in order to increase social awareness and public knowledge on the illness current spreading and its future evolutions. A typical approach relies on traditional compartmental mathematical frameworks, where specific modelling choices and parameters embody the different features that characterize the epidemic spreading, the virological effects of the illness and the impact of different pharmaceutical interventions. However, in presence of new epidemic outbreaks many key-features could still be unknown or difficult to isolate in clinical trials, and consequently the illness itself could be difficult to be completely described by the classical compartmental models. Indeed, clinical symptoms of different illnesses are multifaceted and strictly depend on the origin of the pathogenic microbial agent responsible of the disease, which can be bacterial, parasitic, fungal, viral or originated by prions, *i.e.* other kinds of toxic proteins, and on the pathway through which the illness naturally diffuses [1, 2]. Moreover, in order to accurately describe the disease through compartmental models, it is fundamental to account for possible DNA or RNA mutations from the wildtype strain in long-term outbreaks, as well as for possible preventive measures and control, including vaccination, treatments, prophylaxis, quarantine, isolation or other measures minimizing social activities (like the use of face-masks, compulsory home-schooling, different levels of lockdowns). In these highly complex and rapidly changing scenarios, the efficacy of compartmental models for making fast scenario-analyses may be severely limited by the delicate and sometimes ad-hoc parameter calibration process that becomes even harder if one aims at embodying age-dependency or other geographical features, see, *e.g.*, [3, 4, 5].

Alongside with scenario analysis, recent upcoming epidemic events have shown the importance of disposing of computational tools measuring the impact of pharmaceutical resources [6] and other Non-Pharmaceutical Interventions (NPIs) [7] so to guide policy-makers in choosing how to intervene limiting the social and economic burden.

From the mathematical perspective, we can leverage on the versatility of optimal control theory in order to derive useful quantitative and qualitative guidelines for minimizing the amount of infectious or deceased individuals [8, 9], the total incidence of the spreading disease [10], or the amount of contacts and, consequently, of cases [11]. Other problems which have been further investigated, analytically and numerically, are more delicate from the mathematical viewpoint such as the minimization of epidemic peaks [12] or the minimization of the eradication time [13].

In view of the above discussion, it is of paramount importance to provide the society with mathematical tools able to output computationally cheap and reliable scenario analyses, so to compare different prevention measures and solve optimal control problems. Among recently developed mathematical frameworks, a prominent position is covered by Operator Learning, which, roughly speaking, deal with the development and application of algorithms designed to build approximation of operators starting from a given set of input/output pairs. In the family of Operator Learning tools, an increasing attention has been devoted to the so called Deep Operator Networks (DeepONets) introduced in [14]. Since then, other ML algorithms exploiting deep neural networks have been developed, such as PINNDeepONets [15] and Fourier Neural Operators [16]. More recently, in [17] Kernel Operator Learning (KOL) has been proposed as a competitive alternative to the previous approaches in terms of cost-accuracy trade-off and the capability of matching (sometimes outperforming) in several benchmark problems the generalization properties of learning methods based on neural networks. Moreover, the simple closed formula of the learnt kernel operator makes KOL very attractive.

In this work we propose two different numerical approaches based on Kernel Operator Learning (KOL) [17], that starting from epidemic data provide surrogate models describing the dependency of different stages of the illness on a given control function representing NPIs. To the best of our knowledge, these approaches are new in the epidemic context. Moreover, the numerical experiments contained in the sequel of the paper show that the presented approaches can be efficiently employed for making fast scenario analyses and solving optimization tasks, since they can be rapidly trained directly from data, circumventing delicate calibration phases. For the sake of simplicity and to present the main features of our approaches, we work with synthetic data generated by standard epidemic differential models governed by control functions modelling NPIs (or any other effect reducing the transmissibility of the illness), which we assume to be given.

As pointed out in [17], the performance of KOL depends on the choice of the kernel employed to perform the regression tasks, see also [18]. Very recently, Neural Tangent Kernels (NTKs) have been introduced in a series of pioneering works (cf. [19, 20, 21]) that opened the door to a very intensive research activity. Loosely speaking, NTK arise from the connection with infinite-width neural networks and, within the infinite-
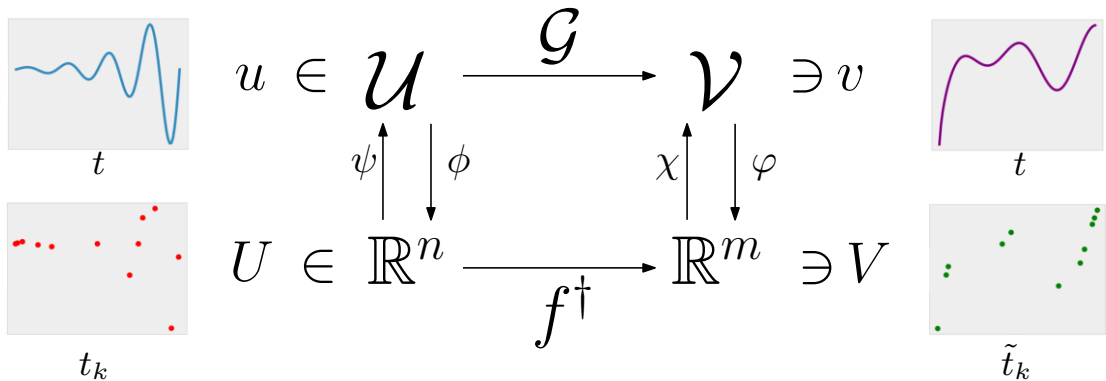
Fig 1. Kernel Operator Learning (KOL) diagram. Operating among RKHS justifies the reduction of the problem to learning the behaviour of the vector-value function $f^\dagger$, operating between input/output observations (cf. Supplementary Information S2).

width limit, the generalization properties of the neural network could be described by the generalization properties of the associated NTK. In view of this connection and encouraged by the results presented in [19, 22], in this paper we employ NTKs in the construction of our kernel operator approaches and we validate its efficacy through a wide campaign of numerical tests.

The paper is organized as follows. In Section 2, we synthetically illustrate and derive the mathematical formulation of our KOL approaches and we briefly recall the compartmental models that we adopt for generating the numerical data to test the methods. In Section 3 we present and discuss different numerical tests for assessing the generalization properties of our approached together with their efficacy in providing solutions to optimization tasks.

2. Materials and Methods

This section is organized in three parts: Section 2.1 is devoted to introducing Kernel Operator Learning in the epidemic context, whilst Section 2.2 briefly recalls the standard compartmental epidemic models that will be employed to produce the synthetic data feeding the KOL. In Section 2.3 we gather some preliminary considerations about the numerical technicalities for retrieving both KOL regressors.

2.1. KOL and epidemic modelling: Basic principles

We briefly summarize some of the principal results on Kernel Operator Learning (KOL) contained in [17], that will be instrumental to build surrogate epidemic models and to solve optimal control problems.

To this aim, let us consider two possibly infinite-dimensional Hilbert spaces $\mathcal{U}, \mathcal{V}$ and assume there exists an unknown operator mapping between the two spaces, *i.e.*

$$\mathcal{G} : \mathcal{U} \to \mathcal{V}. \tag{1}$$

Roughly speaking, the goal of operator learning is to approximate $\mathcal{G}$ based on pairs of input/output that are accessible through finite dimensional linear measurements, as formalized by the following:

Problem. *Let $\{u_i, v_i\}_{i=1}^{N}$ be N samples in $\mathcal{U} \times \mathcal{V}$, i.e.*

$$\mathcal{G}(u_i) = v_i, \text{ with } i = 1, 2 \ldots N. \tag{2}$$

Moreover, define the (bounded and linear) observation operators $\phi : \mathcal{U} \to \mathbb{R}^n$ and $\varphi : \mathcal{V} \to \mathbb{R}^m$ acting on the input and the target functions, respectively. The goal of operator learning is the approximation of the operator $\mathcal{G}$ based on the observation input/output pairs $\{\phi(u_i), \varphi(v_i)\}_{i=1}^{N}$.

In the rest of the section, to ease the reading, we restrict the presentation to the scalar case, while the extension to the vector-valued case, relevant for the numerical examples in the following sections, can be straightforwardly obtained from the scalar one by resorting to the theory of vector-valued Reproducing Kernel Hilbert Spaces (RKHS) in [23]. Therefore, let us consider $\mathcal{U}$ and $\mathcal{V}$ as functional spaces made of scalar functions, where the dependant variable is named t and it is assumed to vary in the interval $D \subset \mathbb{R}$. In this scenario, a standard choice for the observation operators ϕ and φ consists in considering the pointwise evaluation at specific collocation points $\{t_k\}_{k=1}^{m}$ and $\{\tilde{t}_k\}_{k=1}^{n}$ respectively, which in general can be different (see Figure 1). However, for simplicity, in this work we consider the same collocation points, *i.e.*

$$\begin{aligned} \phi : u \to U &:= (u(t_1), u(t_2) \ldots u(t_n))^T \in \mathbb{R}^n, \\ \varphi : v \to V &:= (v(t_1), v(t_2) \ldots v(t_n))^T \in \mathbb{R}^n. \end{aligned} \tag{3}$$

Then, assuming we are given the training dataset $\{U_j, V_j\}_{j=1}^{N}$ where, consistently with our notation, $U_j := (u_j(t_1), u_j(t_2) \ldots u_j(t_n))^T \in \mathbb{R}^n$ and $V_j := (v_j(t_1), v_j(t_2) \ldots v_j(t_n))^T \in \mathbb{R}^n$, we aim at constructing an approximation $\bar{\mathcal{G}}$ of $\mathcal{G}$. More specifically, following [17], endowing $\mathcal{U}$ and $\mathcal{V}$ with a RKHS structure and using kernel regression to identify the maps ψ and χ (cf. Figure 1 and see Supplementary Information S2 for more details), the approximated operator can be written explicitly in closed form as

$$\bar{\mathcal{G}}(u)(t) = K(t, T)K(T, T)^{-1} \left(\sum_{j=1}^{N} S(\phi(u), U_j)\alpha_j \right), \tag{4}$$

where K is the kernel function induced by the RKHS structure of $\mathcal{V}$, the vector $T = [t_1, t_2 \ldots t_n]^T \in \mathbb{R}^n$ contains the collocation points, and $S : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ is a properly chosen vector-valued kernel. Moreover, $K(\cdot, T) : D \to \mathbb{R}^n$ is a row vector such that $K(t, T)_i = K(t, t_i)$, and $K(T, T)$ is an $n \times n$ matrix such that $K(T, T)_{ij} = K(t_i, t_j)$. Parameters $\{\alpha_j\}_{j=1}^N \in \mathbb{R}^n$ are the kernel regression parameters over the input/output training pairs. Since we consider pointwise observation operators, $K(t, T)$ is the composite linear interpolant of nodes in T evaluated at the desired time t. We refer to [17] for a complete discussion of the convergence properties of this approach and for the formal derivation of the learnt operator in a more general functional framework.

Although the above framework is quite general and can be applied to a variety of problems, here, given the goal of the paper, we embody it in the context of reconstructing processes ruled by dynamical systems steered by control variables. More precisely, we consider the following general scalar dynamical system ruled by the control function u

$$\begin{cases} \dot{v}(t) = F(v(t), u(t)), \ \forall t \in (0, t^*] \\ v(0) = v_0 \end{cases} \tag{5}$$

where $F : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ is assumed to be sufficiently smooth to guarantee the well-posedness of the problem [24]. Clearly, we can associate to (5) the mapping $\mathcal{G}$ that given the control u returns the solution v. This mathematical framework fits a typical epidemic context, where it is often necessary to take into account the impact of different NPIs acting in reducing the contact rate among vulnerable people, such as, for instance, partial or total lockdowns, the mandatory use of face-masks in public spaces or the isolation of people showing mild symptoms which could be linked to the illness of concern.

Now, we are ready to introduce our strategies. The first strategy, named KOL-m, consists in directly determining an approximation of the solution map $\mathcal{G}$ from $u(t)$ to $v(t)$, as described in the following.

KOL-m. *Let $\mathcal{U} = \{u \in L^2(0, t^*)\}$ and $\mathcal{V} = C^0([0, t^*])$. Given the input-output data trajectories $\{(\hat{u}_k, \hat{v}_k)\}_{k=1}^N$, where each $\hat{u}_k \in \mathcal{U}$ is a control function and each $\hat{v}_k \in \mathcal{V}$ is the associated (known) state vector function, the Kernel Operator Learning Map method (KOL-m) builds $\bar{\mathcal{G}}_m : \mathcal{U} \to \mathcal{V}$ according to (4).*

Hence, the quantity $\bar{\mathcal{G}}_m(u^*)(t)$ represents an approximation to the solution $v(t)$ of (5) with control variable $u^*(t)$.

Let us now describe the second approach, namely KOL-∂, which approximates the operator that given the prescribed control returns the derivative of the solution to (5).

KOL-∂. *Let $\mathcal{U} = \{u \in L^2(0, t^*)\}$ and $\hat{\mathcal{V}} = L^1(0, t^*)$. Given the input-output data trajectories $\{(\hat{u}_k, \dot{\hat{v}}_k)\}_{k=1}^N$, where each $\hat{u}_k \in \mathcal{U}$ is a control function and each $\dot{\hat{v}}_k \in \hat{\mathcal{V}}$*

is the associated (known) derivative of the state vector function, the Kernel Operator Learning Derivative method (KOL-∂) builds $\bar{\mathcal{G}}_\partial : \mathcal{U} \to \hat{\mathcal{V}}$ according to (4).

Therefore, the quantity $\bar{\mathcal{G}}_\partial (u^*)(t)$ represents an approximation to $\dot{v}(t)$, *i.e.* the derivative of the solution to (5) with control variable $u^*(t)$. Specifically, we have

$$v(t) = v_0 + \int_0^t \bar{\mathcal{G}}_\partial(u)(\tau)d\tau \ \in \mathcal{V} \text{ where } \bar{\mathcal{G}}_\partial : \mathcal{U} \to \hat{\mathcal{V}} \text{ satisfies (4),} \qquad (6)$$

or alternatively

$$\begin{cases} \dot{v}(t) = K(t,T)K(T,T)^{-1}\left(\sum_{j=1}^N S(\phi(u), U_j)\alpha_j \right) \\ \qquad = K(t,T)K(T,T)^{-1}\left(\sum_{j=1}^N S(\phi(u), U_j) \begin{pmatrix} [\mathbf{S}(\mathbf{U},\mathbf{U})^{-1}\hat{\mathbf{V}}_{\cdot,1}]_j \\ [\mathbf{S}(\mathbf{U},\mathbf{U})^{-1}\hat{\mathbf{V}}_{\cdot,2}]_j \\ \vdots \\ [\mathbf{S}(\mathbf{U},\mathbf{U})^{-1}\hat{\mathbf{V}}_{\cdot,n}]_j \end{pmatrix} \right) \quad \forall t \in (0,t^*] \\ v(0) = v_0, \end{cases}$$

$$(7)$$

where $\hat{\mathbf{V}}_{\cdot,k} = [[\varphi(\dot{v}_1)]_k, [\varphi(\dot{v}_2)]_k \ldots [\varphi(\dot{v}_N)]_k]^T$, $\forall k = 1, 2 \ldots n$ is the vector of the evaluations at the k-th point of each output, $\mathbf{U}$ is a vector collecting all the $\{U_j\}_j^N$ and $\mathbf{S}(\mathbf{U},\mathbf{U}) \in \mathbb{R}^{N \times N}$ can be defined as $[\mathbf{S}(\mathbf{U},\mathbf{U})]_{ij} = S(U_i, U_j)$ (cf. Supplementary Material S2). In this case we assume that an accurate approximation of the derivative of each state $(\{v_i\}_{i=1}^N)$ is observable, hence the sequence $\{\varphi(v_i)\}_{i=1}^N$ is available (cf. Figure 1).

We conclude this section with a methodological remark on the application of the KOL-m approach to the reconstruction of compartments when dealing with epidemic problems or other problems where the reconstructed operator has to preserve positivity. This modified version of KOL-m will be employed to derive the numerical results presented in the subsequent sections.

Remark (On the positivity preserving property of KOL-m). *The solutions to the epidemic differential problems are intrinsically positive, since they represent positive fractions of the population corresponding to different states with respect to the illness spreading. However, there is no reason why KOL-m should preserve the positivity of the prediction, even in presence of positive data $\{\hat{\boldsymbol{x}}_k\}$. For this reason, in order to enforce the positivity of the prediction of the learnt operator $\bar{\mathcal{G}}_m$, we proceed as follows: given $\{(\hat{u}_k, \hat{\boldsymbol{x}}_k)\}_{k \leq N}$, with positive $\hat{\boldsymbol{x}}_k$, we apply KOL-m to the modified input-output dataset $\{(\hat{u}_k, \sqrt{\hat{\boldsymbol{x}}_k})\}_{k \leq N}$ so to obtain the intermediate operator $\widetilde{\mathcal{G}}_m$. Then, we set*

$\bar{\mathcal{G}}_m = \widetilde{\mathcal{G}}_m^2$. *The efficiency of this approach is showed in Section 3.*

2.2. Overview of compartmental models

In order to test KOL-m and KOL-∂ we generate synthetic data employing classical compartmental models and use the latter to validate the accuracy of the surrogate models. More precisely, for simplicity and without loss of generality, we restrict our attention to the four classical compartmental models reported in Figure 2, where the control variable u represents the instantaneous reduction of the basic transmission rate β dictated by the virological and transmissibility properties of the illness by reducing the contact rate. We assume that $u_{lb} \leq u(t) \leq u_{ub}$, $\forall t \in [0, t^*]$, where the upper bound value of the control function $u_{ub} = 1$ models total lockdown, instead the lower bound $u_{lb} = 0$ stands for null NPIs. For suitable choices of the function $\mathbf{F} : \mathbb{R}^d \times \mathbb{R} \to \mathbb{R}^d$, all those models (and other high dimensional disease-specific compartmental models, *e.g.* [25, 26, 5]) can be written in the following general form:

$$\begin{cases} \dot{\boldsymbol{x}}(t) = \mathbf{F}(\boldsymbol{x}(t), u(t)), \ \forall t \in (0, t^*] \\ \boldsymbol{x}(0) = \boldsymbol{x}_0. \end{cases} \tag{8}$$

Clearly, the solution of (8) can be written in terms of the operator $\mathcal{G}$, that given the input control $u(t) : [0, t^*] \to \mathbb{R}$ returns the state $\boldsymbol{x}(t) : [0, t^*] \to \mathbb{R}^d$. For each model, $\forall t \in [0, t^*]$ the evolution function $\mathbf{F}$ is sufficiently smooth in order to guarantee existence and uniqueness of the solution of Cauchy problem (8) when u admits at most a countable amount of jump discontinuities (see, *e.g.* [27]). For a more comprehensive review on epidemiological models, we refer, *e.g.*, to [1, 2].

2.3. Numerical aspects of KOL-m and KOL-∂

In this section we collect some useful information on the computational aspects involved in the training process of KOLs. For the purpose of this work we assume that the space of our discrete counterpart of the control functions and the corresponding discrete vectors coming from the evaluation of the solution are of the same dimension, *i.e.* $m = n$ in Figure 1. It is worth to remark that, in the epidemic context, it is natural to consider, as observation operators for the state and the control, the pointwise evaluation, which is coherent with the way open access datasets for epidemic events are often organized and real-time forecasts are delivered (see, *e.g.* [28]).

For what concerns the generation of synthetic data, we employ N different control functions to be observed via the observation operators, together with the N associated compartmental trajectories (vector in $\mathbb{R}^d$) obtained by numerically solving one of the differential systems in Figure 2 using Explicit Euler with time step dt.

From the computational perspective, once the scalar kernel $S(\cdot, \cdot)$ between the two discrete vector spaces has been chosen, we need to solve n linear systems of

SIR model

$$
\begin{cases}
\dot{S} = -\beta(1-u)SI \\
\dot{I} = \beta(1-u)SI - \gamma I \quad \forall t \in (0, t^*] \\
\dot{R} = \gamma I
\end{cases}
$$

$$[S(0), I(0), R(0)]^T = [S_0, I_0, R_0]^T$$

$$\mathcal{R}_0 = \frac{\beta}{\gamma} \qquad \mathcal{R}_u = \frac{\beta(1-u)}{\gamma}$$

SIS model

$$
\begin{cases}
\dot{S} = -\beta(1-u)SI + \gamma I \\
\dot{I} = \beta(1-u)SI - \gamma I
\end{cases} \quad \forall t \in (0, t^*]
$$

$$[S(0), I(0)]^T = [S_0, I_0]^T$$

$$\mathcal{R}_0 = \frac{\beta}{\gamma} \qquad \mathcal{R}_u = \frac{\beta(1-u)}{\gamma}$$

SIRD model

$$
\begin{cases}
\dot{S} = -\beta(1-u)SI + \delta R \\
\dot{I} = \beta(1-u)SI - (\gamma + \varepsilon)I \\
\dot{R} = \gamma I - \delta R \quad \forall t \in (0, t^*] \\
\dot{D} = \varepsilon I
\end{cases}
$$

$$[S(0), I(0), R(0), D(0)]^T = [S_0, I_0, R_0, D_0]^T$$

$$\mathcal{R}_0 = \frac{\beta}{\gamma + \varepsilon} \qquad \mathcal{R}_u = \frac{\beta(1-u)}{\gamma + \varepsilon}$$

SEIRD model

$$
\begin{cases}
\dot{S} = -\beta(1-u)SI + \delta R \\
\dot{E} = \beta(1-u)SI - \phi E \\
\dot{I} = \phi E - (\gamma + \varepsilon)I \quad \forall t \in (0, t^*] \\
\dot{R} = \gamma I - \delta R \\
\dot{D} = \varepsilon I
\end{cases}
$$

$$[S(0), E(0), I(0), R(0), D(0)]^T = [S_0, E_0, I_0, R_0, D_0]^T$$

$$\mathcal{R}_0 = \frac{\beta}{\gamma + \varepsilon + \phi} \qquad \mathcal{R}_u = \frac{\beta(1-u)}{\gamma + \varepsilon + \phi}$$

Fig 2. *SIR*, *SIS*, *SIRD* and *SEIRD* compartmental models. Each model is provided with its respective basic reproduction number ($\mathcal{R}_0$) and the reproduction number depending on the control ($\mathcal{R}_u$).

dimension $N \times N$ where the matrix is $\mathbf{S}(\mathbf{U}, \mathbf{U})$ (cf. Supplementary Material S1). For this purpose, we compute the Cholesky Factorization of the matrix and solve the systems with the standard substitution methods (cf. [29] for more advanced strategies). Finally, in solving the regression problem we add a regularization term, with penalty parameter equal to $1e - 10$.

3. Results and Discussion

In this section we present a wide campaign of numerical tests with the aim of: (a) understanding the impact of the choice of different kernels in KOL-m and KOL-∂ (cf. Section 3.1); (b) comparing our KOL methods with a popular neural-network based model learning method (cf. Section 3.2); (c) assessing the robustness of the introduced approaches for solving two optimization tasks (cf. Section 3.3).

3.1. On the choice of the Kernel

One crucial ingredient, driving the approximation and generalization properties of the corresponding KOL method, is the choice of the scalar kernel S in (35). Which

kernel function is optimal for kernel regression is still an open debate and depends on the specific application. For instance, there exists some innovative approaches which learn the kernels by simulating data driven dynamical systems enabling scalability of Kernel Regression [30]. Here, we consider the following popular choices for S.

- **Linear Kernel**: $S(U_1, U_2) = U_1^T U_2$. This kernel evaluates the alignment in the n-dimensional space between input vectors;

- **Matérn Kernel**: $S(U_1, U_2) = \dfrac{2^{1-\nu}}{\Gamma(\nu)} \left(\sqrt{2\nu} \dfrac{\|U_1 - U_2\|_2}{\rho} \right)^\nu K_\nu \left(\sqrt{2\nu} \dfrac{\|U_1 - U_2\|_2}{\rho} \right)$, where $\nu > 0$ controls the smoothness of the kernel function, ρ is a characteristic length scale, Γ is the Gamma function and K_ν is the modified second kind Bessel function of order ν. This family of kernels is stationary and isotropic, given the Euclidean distance between input points. If $\nu \to \infty$ one obtains the Gaussian kernel. It is often used for image analysis [31] and other machine learning regression tasks [32];

- **RBF kernel**: $S(U_1, U_2) = e^{-\frac{\|U_1 - U_2\|_2^2}{2\sigma^2}}$, known also as Gaussian kernel. It has the interpretation of a similarity measure since it is bounded in $[0, 1]$ and decreases as long as the distance between points increases;

- **Rational Quadratic kernel**: $S(U_1, U_2) = \left(1 + \dfrac{\|U_1 - U_2\|_2^2}{2\alpha l^2} \right)^{-\alpha}$, with $l, \alpha > 0$. It can be regarded as the infinite sum of different RBF kernels with different length scales;

- **Neural Tangent Kernel (NTK)**: Given a neural network regressor $f(x; \theta)$ of depth d_{nn}, width l_{nn} and activation function σ_{nn}, with θ denoting the vector collecting all weights and biases, we define the family of finite-width Neural Tangent Kernels $\{S\}_{\tau > 0} : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ (cf. S1) as

$$S_\tau(x_i, x_j) := \langle \partial_\theta f(x_j; \theta(\tau)), \, \partial_\theta f(x_i; \theta(\tau)) \rangle, \tag{9}$$

where τ represents a fictitious iteration time. It has been proven that, if the initialization of the weights follows the so-called NTK initialization [20], in the infinite-width limit each element in $\{S_\tau\}_\tau$ converges in probability to a stationary kernel independently on τ, *i.e.*

$$S_\tau(x_i, x_j) \underset{\mathbb{P}}{\to} S(x_i, x_j), \; \forall \tau > 0, \; \forall x_i, x_j. \tag{10}$$

Hence, the family of NTKs strictly depends on two parameters: activation function and depth of the associated neural network.

In this paper, with NTK we refer to the infinite-width limit kernel function. We rely on the library presented in [33] for an efficient and user-friendly computation of the considered NTKs. In practice, given the network hyperparameters, we compute the scalar NTK $S : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ and then evaluate it at desired couples of points. More details about the derivation of NTKs associated to neural networks can be found in the Supplementary Material S1 Section.

For each of the above kernels, we evaluate the generalization properties of KOL-m and KOL-∂ trained on the epidemic SIS, SIR and $SEIRD$ trajectories generated by control functions chosen among the following families (see Figure 3) which are representative of possible interventions that policy makers can implement.

For training and testing KOL-m and KOL-∂, we define a mixed dataset constituted by functions belonging to four distinct functional families (see Figure 3):

1. **Linear Pulse** (Figure 3a):

$$u(t) = \begin{cases} u_0, & t \leq t_0, \\ \dfrac{3(u_1 - u_0)}{\Delta t}(t - t_0) + u_0 & t_0 < t \leq t_0 + \frac{\Delta t}{3}, \\ u_1 & t_0 + \frac{\Delta t}{3} < t \leq t_0 + \frac{4\Delta t}{3}, \\ \dfrac{3(u_0 - u_1)}{\Delta t}(t - t_0 - \frac{4\Delta t}{3}) + u_1 & t_0 + \frac{4\Delta t}{3} < t \leq t_0 + \frac{5\Delta t}{3}, \\ u_0 & t > t_0 + \frac{5\Delta t}{3}, \end{cases} \tag{11}$$

   depending on 4 degrees of freedom (dof): u_0, $u_1 \in [0,1]$, $t_0 \in [0, t^*]$ and $\Delta t \in [0, \frac{t^*}{3}]$;

2. **Step function** (Figure 3b):

$$u(t) = \begin{cases} u_0 & t \leq t_0, \\ u_1 & t > t_0, \end{cases} \tag{12}$$

   with 3 dofs: u_0, $u_1 \in [0,1]$, $t_0 \in [0, t^*]$;

3. **Continuous Seasonality** (Figure 3c):

$$u(t) = \frac{u_0}{2}\left(1 + \frac{1}{2}\cos\left(\frac{2\pi t}{t^*} + \frac{\Delta t}{t^*}\frac{\pi}{2}\right)\right), \tag{13}$$

   with 2 dofs: $u_0 \in [0,1]$ and $\Delta t \in [0, \frac{t^*}{3}]$;

11

4. **Double step** (Figure 3d):

$$u(t) = \begin{cases} u_0 & t \leq t_0, \\ u_1 & t_0 < t \leq t_0 + \frac{\Delta t}{2}, \\ u_0 & t_0 + \frac{\Delta t}{2} < t \leq t_0 + \Delta t, \\ u_1 & t > t_0 + \Delta t, \end{cases} \tag{14}$$

with 3 dofs: u_0, $u_1 \in [0, 1]$, $t_0 \in [0, t^*]$.

To test the generalization properties we proceed as follows. For each given input control function $\{u^n\}_n$, $n = 1, 2, \ldots, N_p$ with N_p number of test points, we generate the output data $\{\boldsymbol{x}^{i,n}\}_{i,n}$, $i = 1, 2, \ldots, N_c$ with N_c the number of compartments, by numerically solving the specific systems of ODEs (SIS, SIR, $SIRD$ and $SEIRD$) up to a final time $t^* = 100$, with a discretization step $dt = 1$. The pairs control/output data form the training dataset. We define the prediction relative error as

$$p_{\text{err}} = \frac{1}{N} \sum_{n=1}^{N} \sum_{i=1}^{N_c} \frac{\|\boldsymbol{x}_p^{i,n} - \boldsymbol{x}^{i,n}\|_2}{\|\boldsymbol{x}^{i,n}\|_2}, \tag{15}$$

where $\{\boldsymbol{x}_p^{i,n}\}$ are the predictions generated by the KOL methods. Clearly, the prediction samples belong to a different batch with respect to the training input data.

In all cases, the value of the basic reproduction number is set at $\mathcal{R}_0 = 4$, and the recovery rate $\gamma = 0.05$. The epidemic event starts with 1% of infected (or, respectively, exposed) individuals in all cases. The rest of the population is assumed to be susceptible to the disease at the initial time. For the $SEIRD$ model, the remaining parameters have been chosen as $\delta = 0.4$, $\varepsilon = \phi = 0.05$. The transmission rate β without control is, therefore, computed starting from the definition of $\mathcal{R}_0$ of each model in Figure 2.

For the comparison of the different generalization properties of the KOL acting on different kernels, the training dataset is built employing 500 control functions equally distributed among the above 4 classes of control functions, where the dofs are randomly sampled following a uniform distribution in the respective interval of definition of each parameter. As for the prediction samples, we test the KOL approaches over a dataset built employing 100 samples equally distributed among the same families of control functions, but with different values for the dofs.
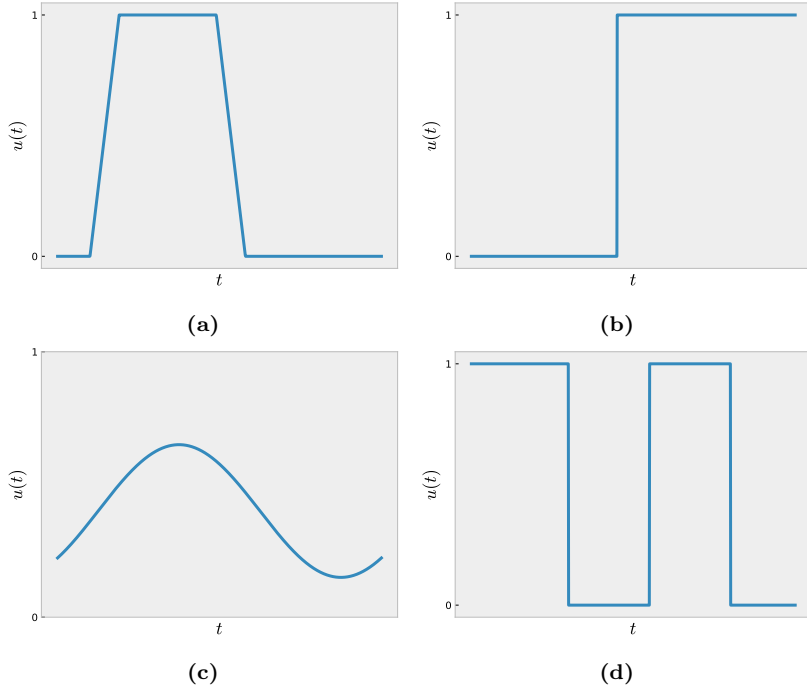
Fig 3. Examples of control functions employed to generate the training-testing dataset for the KOL methods.

Results in Figures 4-6 show the prediction relative error of the different approaches. In particular, for each kernel we trained each KOL-m and KOL-∂ with 20 different training datasets of size 500 as detailed above and we represented the results in the boxplot form. We perform the same error analysis for different compartmental models: SIS (Figure 4), SIR (Figure 5) and $SEIRD$ (Figure 6). The boxplots corresponding to the blue median lines are the one referring to the KOL-∂ approach, whilst the red ones correspond to the KOL-m approach. For those kernels depending on parameters to be tuned, such as RBF, Matérn and Rational Quadratic, we underdid a sensitivity analysis showing similar results exploring the space of parameters (Matérn: $\nu \in [0, 0.1]$, $\rho \in [0.1, 1]$, RBF: $\sigma \in [0, 0.1]$, Rational Quadratic: $\alpha \in [0.01, 0.1]$, $l \in [0.01, 1]$). For what concerns the NTKs, which depend on the choice of the depth d_{nn} and on the activation function σ_{nn} of the associated neural network, we perform the same sensitivity analysis considering both Neural Tangent Kernels with Rectified Linear Unit ($ReLu$) and Sigmoidal activation functions for different depths ($d_{nn} \in \{2, 3, 4\}$). In the boxplots in Figures 4-6 we reported the results of the NTKs corresponding to two-layer networks, which showed to be a suitable trade-off between computational complexity and generalization properties.

We notice that for the SIS and the SIR model, the prediction errors are higher

13

when compared to the one obtained for the $SEIRD$ model. This seems to indicate that KOL methods generalize with lower errors for growing complexity. For the SIS model, the operator approximated via the Linear Kernel does not succeed in surrogating the epidemic dynamics. On the other hand, median errors of KOL methods with $ReLu$-based NTK achieve the lower results, granting errors always lower than 2% across the test samples. Instead, for the $SEIRD$ and SIR model, all the KOLs with the different kernels seem to be good proxies in terms of generalization properties. Specifically, the KOL-∂ methods have better approximation properties, with a particular mention to the NTK-$Relu$ based. With the 5 dimensional model $SEIRD$, among the KOL-m approaches the lowest median error with less IQR is attained by the NTK-sigmoidal one.

In virtue of these results, from now on, we select the NTK-sigmoidal kernel for the KOL-m method, and NTK-$ReLu$ kernel for the KOL-∂ method.

Fig 4. (SIS model) Comparison of the prediction errors over 100 control functions with KOL methods trained on 20 batches of size 500, where the control functions are chosen in the mixed training dataset. Bullet points represent outliers whose prediction error is outside the 1.5 IQR (length of the whiskers) of the set of simulations.
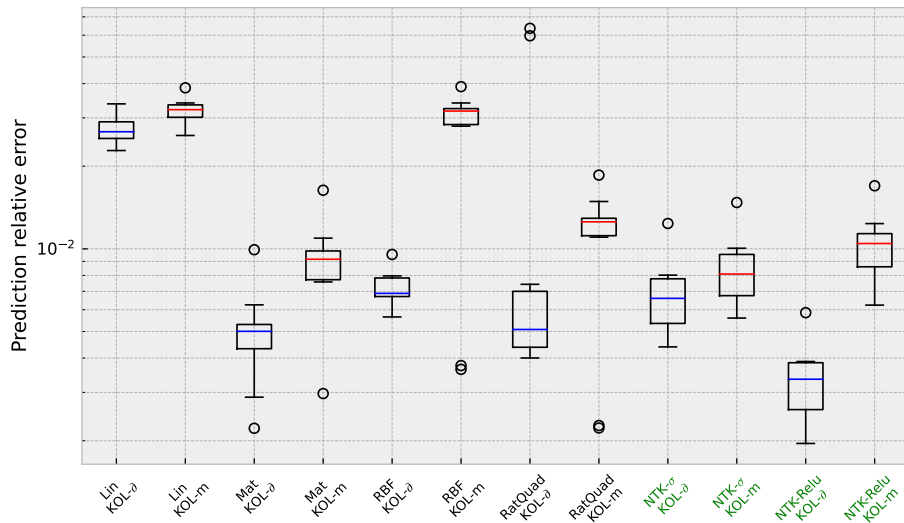
Fig 5. (SIR model) Comparison of the prediction errors over 100 control functions with KOL methods trained on 20 batches of size 500, where the control functions are chosen in the mixed training dataset. Bullet points represent outliers whose prediction error is outside the 1.5 IQR (length of the whiskers) of the set of simulations.
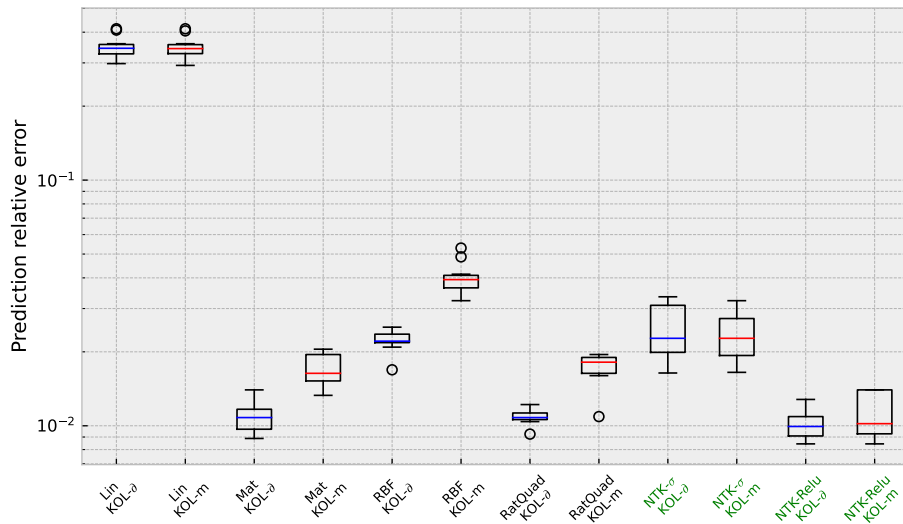

Fig 6. ($SEIRD$ model) Comparison of the prediction errors over 100 control functions with KOL methods trained on 20 batches of size 500, where the control functions are chosen in the mixed training dataset. Bullet points represent outliers whose prediction error is outside the 1.5 IQR (length of the whiskers) of the set of simulations.
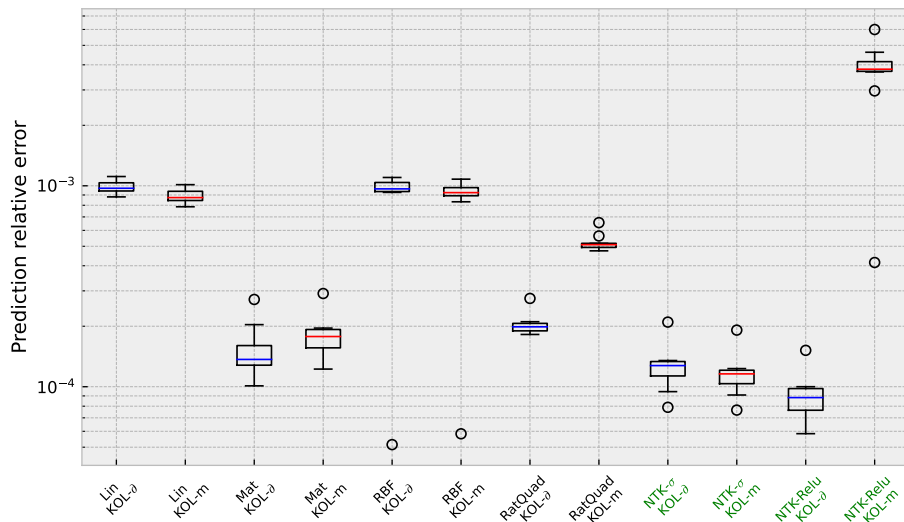
3.2. Comparison between KOL and a Model Learning based on neural networks

In this section, we enlight some important features of our KOL approaches that make them reliable and competitive tools for model learning. To this aim, we compare KOL-m and KOL-∂ with a popular and paradigmatic machine learning approach based on neural networks, namely the approach described in [34], in the sequel called NN-ModL, and shortly described in what follows.

Let $\mathcal{F} = \{\boldsymbol{f}_{nn} : \mathbb{R}^d \times \mathbb{R} \to \mathbb{R}^d\}$ be the space of feed-forward neural network functions. The NN-ModL problem reads as follows:

Problem 1 (NN-ModL). *Solve the constrained optimization problem*

$$\min_{\boldsymbol{f}_{nn} \in \mathcal{F}} \frac{1}{2} \sum_{j=1}^{N} \int_0^{t^*} |\boldsymbol{x}_j(t) - \boldsymbol{x}_{nn,j}(t)|^2 dt, \tag{16}$$

s.t.

$$\begin{cases} \dot{\boldsymbol{x}}_{nn,j}(t) = \boldsymbol{f}_{nn}(\boldsymbol{x}_{nn,j}(t), u_j(t)) & t \in (0, t^*], \ j = 1, 2 \ldots N, \\ \boldsymbol{x}_{nn,j}(0) = \boldsymbol{x}_0, \end{cases} \tag{17}$$

where $\boldsymbol{f}_{nn} \in \mathcal{F}$ and N is the size of the training set.

Hence, NN-ModL reconstructs the map from the control $u(t)$ to the state $\boldsymbol{x}_{nn}(t)$. The problem is recast as a discrete optimization problem where the optimizing variables are weights and biases associated to $\boldsymbol{f}_{nn}$. NN-ModL has shown valuable performances for building reduced models of cardiac electromechanics [35], or for deriving a data-driven model for active force generation in cardiomyocites [36]. As illustrated in [34], the discretized finite-dimensional version of Problem 1 is equivalent to a nonlinear least-square problem, which can been solved employing the Levenberg-Marquardt iterative method [37].

In the sequel, we compare our KOL approaches with NN-ModL in the epidemic context, in terms of: (a) the wall-clock time employed for training the operator learning; (b) the generalization error in the testing phase. The comparison has been drawn by accounting for progressively larger training set sizes for both KOL approaches and NN-ModL in order to approximate the population dynamics generated synthetically by SIR, $SIRD$ and $SEIRD$ models. In particular, we trained NN-ModL with different training sizes and different number of maximum iterations. We performed a sensitivity analysis on the choice of the number of neurons per layers in the NN-ModL case, and concluded, following the *Occam's razor principle of parsimony*, to go for a shallow neural network of 6 neurons.

All the results have been obtained by executing the code in parallel on a 8-core machine Intel i7. In Table 1 we summarize the outputs of the comparison between the NN-ModL and the two KOL approaches, where in each case we considered $dt =$

0.05, $t^* = 5$, $\mathcal{R}_0 = 2$, $\gamma = 0.05$, $\delta = 0.4$ and $\varepsilon = \phi = 0.05$. The initial conditions correspond to those of Section 3.1. The comparison has been done by considering 50, 100 or 200 iterations of the optimization scheme for the NN-ModL, trained with 25, 50 and 100 elements uniformly sampled in the mixed training dataset. Both KOL approaches have also been trained on larger datasets constituted by 200, 500, 800, 1000, 2000, 5000, and 10000 elements. The prediction error has been evaluated according to (15) over the same test set of 100 samples constituted by four batches of dimension 25, each belonging to the four functional families in the mixed dataset. The wall-clock time of the running processes has been measured in seconds.

For each epidemic model, the wall-clock time taken for the training stage of NN-ModL is of orders of magnitude higher than the one needed for KOL approaches, even when considering solely 50 iterations. The NN-ModL approach ends with a prediction error close to 10^{-2} in the three cases after a total wall-clock time ranging from 10^2 (best scenario) and 10^3 seconds (worst scenario). Instead, for all models in less than 8 seconds the KOL-m approach trained with 500 samples reaches a generalization error which is always lower than the corresponding ones reached by NN-ModL. In addition, for all three models, up to 2000-sample training sets, both KOL approaches still employ less wall-clock time than NN-ModL. In particular, in the case of $SEIRD$ model, the prediction error is 3 orders of magnitude lower than the one of NN-ModL. The gain in terms of wall-clock time is more evident for $SEIRD$ model with size 5000 against NN-ModL with size 100.

As expected, further increasing the amount of training samples makes the prediction errors continuously decrease though with a slower rate with respect to the training size. For what concerns the KOL-∂ schemes, they achieve lower prediction errors with respect to the other approaches even with the baseline size of 100 input functions, except for $SIRD$. Moreover, we deduce that in both KOL-m and KOL-∂ approaches, the prediction errors are not affected by the increasing in dimensions of the differential problem going from the 3 dimensional SIR to the 5 dimensional $SEIRD$. Even the computational times surprisingly do not undergo significant changes across the different models.

In conclusion, both KOL approaches return more accurate predictions in a lower computational time with respect to the NN-ModL, trained for different number of iterations. Between the two KOL approaches, KOL-∂ is preferable in terms of accuracy across different epidemic differential models. In general, the KOL approaches turn out to possess a desirable property, *i.e.* the ability of returning reliable predictions in a fast way, which is extremely useful in the epidemic context, for instance when scenario analyses or optimization tasks are required. For the sake of fairness it should be mentioned that, regarding NN-ModL , it is theoretically possible to find a neural-network function able to build an approximation of the target state with any desired accuracy. However, the proof of the existence of such architecture is not

constructive.

	Iterations	Size	SIR		SIRD		SEIRD	
			Error	Time [s]	Error	Time [s]	Error	Time [s]
NN-ModL	50	25	1.5×10^{-2}	279	1.0×10^{-2}	380	1.1×10^{-2}	383
	100	25	1.5×10^{-2}	478	1.0×10^{-2}	615	1.1×10^{-2}	755
	200	25	1.5×10^{-2}	1186	1.0×10^{-2}	1139	1.1×10^{-2}	1673
	50	50	1.2×10^{-2}	576	1.1×10^{-2}	559	1.1×10^{-2}	875
	100	50	1.2×10^{-2}	1363	1.1×10^{-2}	1456	1.1×10^{-2}	1476
	200	50	1.2×10^{-2}	3448	1.1×10^{-2}	2116	1.1×10^{-2}	3046
	50	100	8.8×10^{-3}	893	9.2×10^{-3}	1394	9.5×10^{-3}	1364
	100	100	8.8×10^{-3}	1672	9.2×10^{-3}	2801	9.5×10^{-3}	3262
	200	100	8.8×10^{-3}	1725	9.2×10^{-3}	4511	9.4×10^{-3}	6668
KOL-m		25	8.3×10^{-2}	0.425	8.2×10^{-2}	0.385	7.5×10^{-2}	0.209
		50	5.4×10^{-2}	0.521	5.5×10^{-2}	0.486	4.8×10^{-2}	0.479
		100	1.6×10^{-2}	0.725	1.5×10^{-2}	0.731	4.9×10^{-3}	0.755
		200	1.2×10^{-2}	1.47	1.1×10^{-2}	1.492	2.6×10^{-3}	1.46
		500	5.6×10^{-3}	7.60	5.4×10^{-3}	7.09	9.1×10^{-5}	6.85
		800	5.5×10^{-3}	17.8	5.3×10^{-3}	18.8	7.0×10^{-5}	20.7
		1000	4.7×10^{-3}	32.5	4.5×10^{-3}	25.7	6.4×10^{-5}	29.4
		2000	3.0×10^{-3}	116	2.9×10^{-3}	94.5	3.4×10^{-5}	95.8
		5000	2.3×10^{-3}	572	2.2×10^{-3}	568	1.3×10^{-5}	559
		10000	1.5×10^{-3}	2269	1.4×10^{-3}	2265	8.4×10^{-6}	2288
KOL-∂		25	7.0×10^{-3}	0.420	1.2×10^{-2}	1.27	2.6×10^{-3}	0.501
		50	4.2×10^{-3}	0.524	8.6×10^{-3}	0.872	1.2×10^{-3}	0.856
		100	2.5×10^{-3}	0.744	4.9×10^{-3}	1.17	3.6×10^{-4}	1.11
		200	1.5×10^{-3}	1.54	3.5×10^{-3}	2.14	2.3×10^{-4}	2.05
		500	8.0×10^{-4}	6.32	1.9×10^{-3}	7.85	9.7×10^{-5}	8.26
		800	5.9×10^{-4}	19.3	2.0×10^{-3}	18.9	6.2×10^{-5}	18.1
		1000	5.8×10^{-4}	27.3	1.8×10^{-3}	25.3	5.2×10^{-5}	26.4
		2000	1.2×10^{-4}	93.8	1.5×10^{-3}	90.5	3.9×10^{-5}	94.7
		5000	1.2×10^{-4}	877	1.1×10^{-3}	546	1.9×10^{-5}	888
		10000	1.2×10^{-4}	2206	8.5×10^{-4}	2156	1.3×10^{-5}	3315

Table 1: Wall-clock time comparison for the SIR, $SIRD$ and $SEIRD$ models.

3.3. KOL and epidemic control

In this section we exploit KOL-m and KOL-∂ for the solution of paradigmatic optimal control problems, which are meaningful for making fast and reliable scenario-analyses in the context of epidemic control. More precisely, in Section 3.3.1 we are interested in the minimization of the eradication time prescribed a given epidemic threshold, while in Section 3.3.2 we tackle the minimization of the total amount of infected.

3.3.1. Optimal control for estimating the minimum eradication time

In the sequel, we focus on the minimization of the eradication time prescribed a given epidemic, building upon the theoretical results given in [13]. The presence

of rigorous mathematical results on the existence of an optimal solution, makes the eradication problem a desirable and mathematically solid benchmark to study the reliability of our KOL-based approach. We formulate the problem by considering the standard deterministic SIR model.

Problem 2 (Minimum eradication time). *Let $\mathcal{U} = \{0 \leq u(t) \leq u_{max} \, \forall \, t \in [0, t^*]\}$ be the space of admissible controls. Solve*

$$\min_{u \in \mathcal{U}} \mathcal{J}_{t_e}(u) = \int_0^{t_e(u)} 1 \, dt,$$

subject to the state problem

$$\begin{cases} \dot{S} = -\beta(1-u)SI, \\ \dot{I} = \beta(1-u)SI - \gamma I, \\ \dot{R} = \gamma I, \end{cases}$$

$\forall t \in (0, t^*]$ *given* $[S(0), I(0), R(0)]^T = [S_0, I_0, R_0]^T$, *where* $t_e(u)$ *is the eradication time associated to the control u.*

Given a threshold $\eta > 0$ of infected individuals, the eradication time is defined as the first time when infectious individuals are below the given threshold value, *i.e.*

$$t_e \in (0, t^*] \text{ s.t. } I(t) > \eta \, \forall \, 0 < t < t_e, \, I(t_e) = \eta. \tag{18}$$

It has been proven in [13] that the space of admissible optimal controls can be restricted to

$$\mathcal{A} = \{u : [0, t^*] \to \{0, u_{max}\}, \, u(t) = u_{max} \, H(t - \tau)\}, \tag{19}$$

where τ is the starting intervention time depending on the maximum level of intervention u_{max}, and $H(\cdot)$ is the Heaviside step function. Defining the *maximum control reproduction number* as

$$\mathcal{R}_{u_{max}} = \frac{\beta(1 - u_{max})}{\gamma}, \tag{20}$$

it is possible to find an optimal solution with non-trivial switching time, *i.e.* $\tau > 0$, when considering $\mathcal{R}_{u_{max}} < 1$.

First, we numerically reproduced the results of [13], by considering $\mathcal{R}_0 = 2$, $\gamma = 5$, $dt = 1$, $t^* = 100$, and $u_{max} \in [0, 0.7]$ and solving the above minimization problem constrained by the SIR differential model. In order to match the initial conditions of [13], we consider $S_0 = 2000/2001$ and $I_0 = 1/2001$. We retrieved the same behaviours of the paper of the switching time τ, the eradication time t_e and of the susceptibles evaluated at the eradication time (see Figure 7). We solve the optimal control

19

problem by direct evaluations and comparison of the values of the cost functional associated to the different control strategies in $\mathcal{A}$, that are explored in an exhaustive way by sampling τ in a fine grid of step $\Delta t = 0.01$. This brute force strategy is not always feasible, but in the present context is motivated by the presence of the above mentioned analytical optimal solutions.

Then, building upon the results of Section 3, that show that our KOL approaches can be used as proxy models for a variety of epidemic differential model, we can rewrite Problem 2 by replacing the deterministic SIR state problem with the surrogate KOL-m or KOL-∂ approach. Also in this case, the optimal solution is obtained by subsequent evaluations of control functions over the same fine grid of τ. In this way the optimal solutions obtained with our KOL approaches can be easily compared with the benchmark scenarios, thus obtaining a direct measure of the reliability of the KOL methods.

The KOL approaches have been trained on a specific dataset constituted by 10000 Step functions of different heights, sampled uniformly in $[0, 0.8]$ (see Figure 3b).

The results are collected in Figure 7. More precisely, in Figure 7a we represent the switching times, the eradication time and the susceptibles at the eradication time for the optimal interventions of the strategy with KOL-m as surrogate state problem (point markers) together with the benchmark results (solid lines) depending on the maximum value of the intervention (u_{max}). For the sake of comparison, in the same figures we represent the eradication times and the susceptibles at the eradication time for the trivial strategy implementing a constant intervention fixed at u_{max} (dashed lines). Figure 7b represents the same benchmark quantities of [13] where the different quantities are retrieved by using KOL-∂. The matching of the results shows the ability of our KOL approaches to reconstruct the optimal solution without directly relying on the differential model.

20
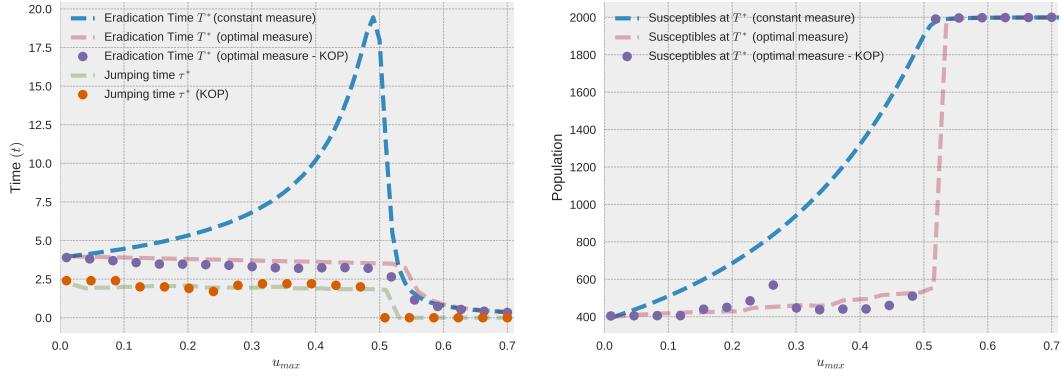
(a) KOL-m.
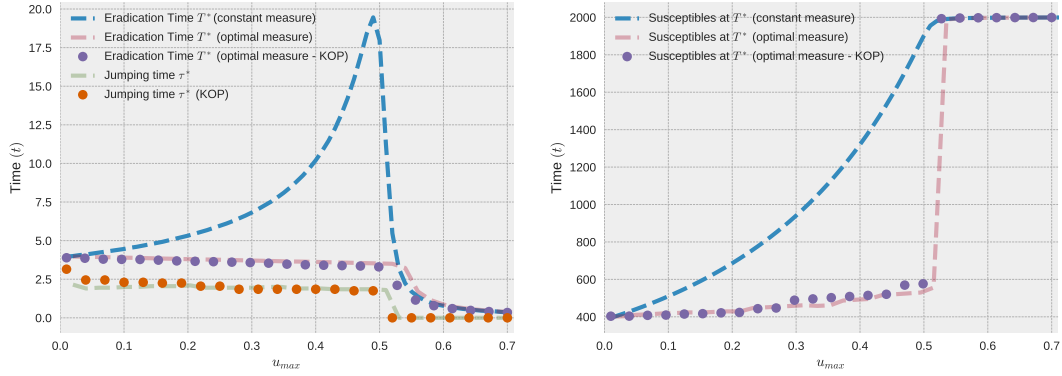
(b) KOL-∂.

Fig 7. Solutions of the optimal control problem searching for minimum eradication time.

3.3.2. Optimal control for minimizing the total amount of infected

Here, we consider a second control problem in which the total amount of infected individuals is minimized, as detailed in the following.

Optimal Control Problem (constrained by ODE).

$$\min_{u \in \mathcal{U}_{ad}} \mathcal{J}_{I,u} := C_I \int_0^{t^*} I(t)^2 dt + C_u \int_0^{t^*} u(t)^2 dt,$$

21

subject to the state problem

$$\begin{cases} \dot{S} = -\beta(1-u)SI, \\ \dot{I} = \beta(1-u)SI - \gamma I, \\ \dot{R} = \gamma I, \end{cases}$$

$\forall t \in (0, t^*]$ *given* $[S(0), I(0), R(0)]^T = [S_0, I_0, R_0]^T$. *The constants* C_I, C_u *are positive and they balance the effects of regularization with respect to the* L^2-*infectious term.*

This is an optimal control problem governed by the SIR model where the control u acts on the transmission rate and incorporates any reducing effects. The optimal configuration obviously depends on the choice of weights C_I and C_u. The cost functional presents the classical Tikhonov regularization term, which has the double-purpose of increasing the convexity of the cost functional and to account for the (economic and social) burden at which higher-level NPIs come. For what concerns the set of admissible controls, we choose the space

$$\mathcal{U}_{ad} = \left\{ u(t) = \sum_{i=1}^{N} u_i \mathbb{1}_{[t_{i-1}, t_i)]}(t), \text{ with } \{t_i\}_{i=0,N} \in (0, t^*] \right\}, \tag{21}$$

containing piecewise constant functions over subintervals of equal length, with the number of subintervals that is a priori chosen. Moreover, choosing the admissible set of controls as in (21) allows to immediately recast the optimal control problems as a discrete optimization problems where the unknowns correspond to the value of the piecewise constant control function in each time slab.

In our numerical exploration, we consider the scenario where $\mathcal{R}_0 = 4$, $t^* = 5$, $dt = 0.05$, initial conditions $[S_0, I_0, R_0]^T = [0.99, 0.01, 0]^T$, $u : [0, t^*] \to [0, 0.7]$ and the number of subintervals $N = \{5, 10, 20\}$. Clearly, larger values of N increase the complexity of the optimization problem, but at the same time they carry more flexibility in designing the optimal control strategy. With the above choice of the parameters, we compare the solutions of the optimal control problem constrained by SIR with the solutions of the following two problems, where the ODE constraint has been replaced by the surrogate operator obtained by KOL-m and KOL-∂, respectively.

Optimal Control Problem (governed by KOL-m).

$$\min_{u \in \mathcal{U}_{ad}} \mathcal{J}_{I,u} = C_I \int_0^{t^*} I(t)^2 dt + C_u \int_0^{t^*} u(t)^2 dt,$$

where the state problem is

$$\begin{pmatrix} S(t) \\ I(t) \\ R(t) \end{pmatrix} = \begin{pmatrix} \bar{\mathcal{G}}_{m,S}(u)(t) \\ \bar{\mathcal{G}}_{m,I}(u)(t) \\ \bar{\mathcal{G}}_{m,R}(u)(t) \end{pmatrix}$$

$\forall t \in (0, T]$ *given* $[S(0), I(0), R(0)]^T = [S_0, I_0, R_0]^T$. *This problem can be equivalently rewritten as an unconstrained minimization problem:*

$$\min_{u \in \mathcal{U}_{ad}} \mathcal{J}_u := C_I \int_0^{t^*} \bar{\mathcal{G}}_{m,I}(u)(t)^2 dt + C_u \int_0^{t^*} u(t)^2 dt.$$

Optimal Control Problem (governed by KOL-∂).

$$\min_{u \in \mathcal{U}_{ad}} \mathcal{J}_{I,u} = C_I \int_0^T I(t)^2 dt + C_u \int_0^T u(t)^2 dt,$$

where the state problem is

$$\begin{pmatrix} \dot{S}(t) \\ \dot{I}(t) \\ \dot{R}(t) \end{pmatrix} = \begin{pmatrix} \bar{\mathcal{G}}_{\partial,S}(u)(t) \\ \bar{\mathcal{G}}_{\partial,I}(u)(t) \\ \bar{\mathcal{G}}_{\partial,R}(u)(t) \end{pmatrix}$$

$\forall t \in (0, T]$ *given* $[S(0), I(0), R(0)]^T = [S_0, I_0, R_0]^T$. *This problem can be equivalently rewritten as an unconstrained minimization problem:*

$$\min_{u \in \mathcal{U}_{ad}} \mathcal{J}_u := C_I \int_0^{t^*} \left(I_0 + \int_0^t \bar{\mathcal{G}}_{\partial,I}(u)(\tau) d\tau \right)^2 dt + C_u \int_0^{t^*} u(t)^2 dt.$$

Both KOL approaches have been trained with a dataset of 800 input control functions belonging to $\mathcal{U}_{ad}$ where the values of the piecewise constant control functions have been sampled from a uniform distribution in $[0, 0.8]$. We considered different values of $C_I > 0$ and $C_u > 0$ and solved the optimization problems using the Sequential Least Square Quadratic Programming methods (SLSQP) [37] as implemented in the python library Scipy [38]. This optimization scheme consists in approximating the original problem with a sequence of quadratic problems, whose objective is a second-order approximation of the Lagrangian of the original problem, and whose constraints are linearized.

In the case of $N = 5$ phases, Figure 8 shows 13 different scenarios of the optimal trajectories reconstructed by solving the problem under SIR (solid line), KOL-m (purple dashed line) and KOL-∂ (red dashed line) constraint, respectively. At first

glance, we see that the optimal controls obtained with the three approaches do not exactly coincide, especially when $C_u \ll C_I$, *i.e.* when the problem is extremely nonconvex. However, comparing the cost functional values associated to the three optimal controls shed a completely different light on the reliability of our approach. Indeed, solving the deterministic SIR problem for each optimal control strategy, and plotting the corresponding cost functional values in Figure 9, we can appreciate that the cost values practically coincide, thus showing the efficacy of our KOL-based approach in solving the optimal control problem. It is worth remarking that in some cases, typically the less convex ones (i.e. C_u much larger than C_I), the KOL approaches succeed in finding cost values lower than the ones obtained using the SIR constraint.

Finally, we compare the three approaches for increasing complexity of the optimization problem, namely for $N = 10, 20$. The results are collected in Figures 10-11, where the optimal control functions are reported, and in Figures 12-13, where the corresponding cost values are compared. Also in these cases, we observe that the cost values achieved by the two KOL approaches are comparable, if not lower, than those obtained employing the SIR constraint. However, with a growing amount of phases the three approaches produce different optimal trajectories due to the increased nonconvexity of the cost functional and, in cascade, of the optimal control problem itself, which exhibit many local minima.

This latest set of results further proves the reliability and the effectiveness of KOL approaches for the most common optimization tasks.

Acknowledgements

Data Availability

The computational framework is available on GitHub together with the numerical results of this work: `https://github.com/giovanniziarelli/KOL`.

Fig 8. Optimal controls for the three optimal control problems fixing $N = 5$.

Fig 9. Cost functionals at the optimal control for different C_I and C_u ($N = 5$). (a) We consider different orders of magnitude for C_I, keeping $C_u = 1e - 1$. (b) We consider different orders of magnitude for C_u, keeping $C_I = 1$.
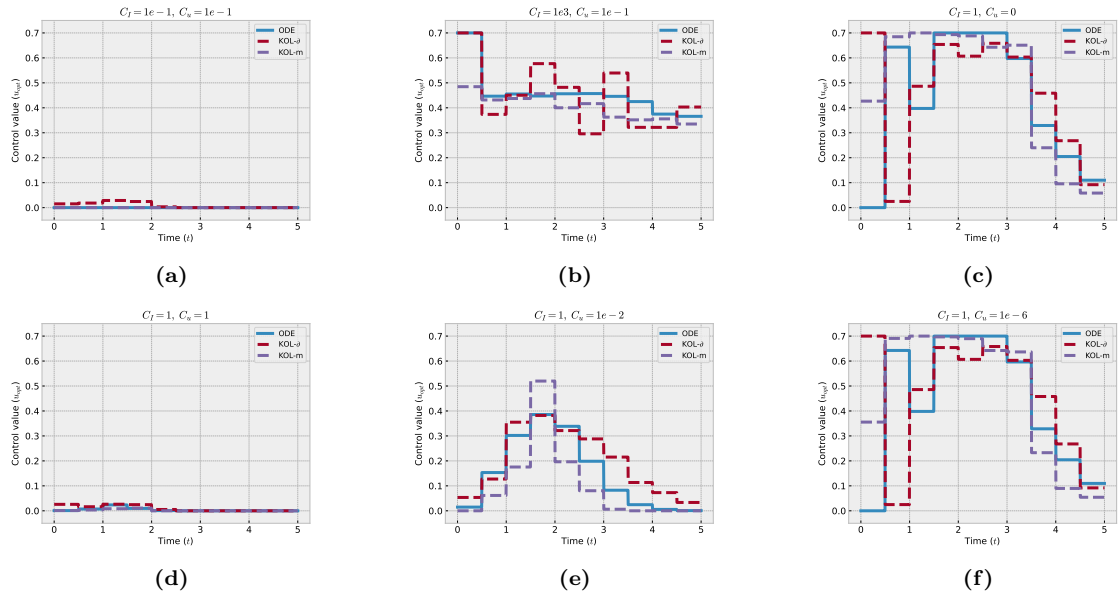
Fig 10. Optimal controls for the three optimal control problems fixing $N = 10$.
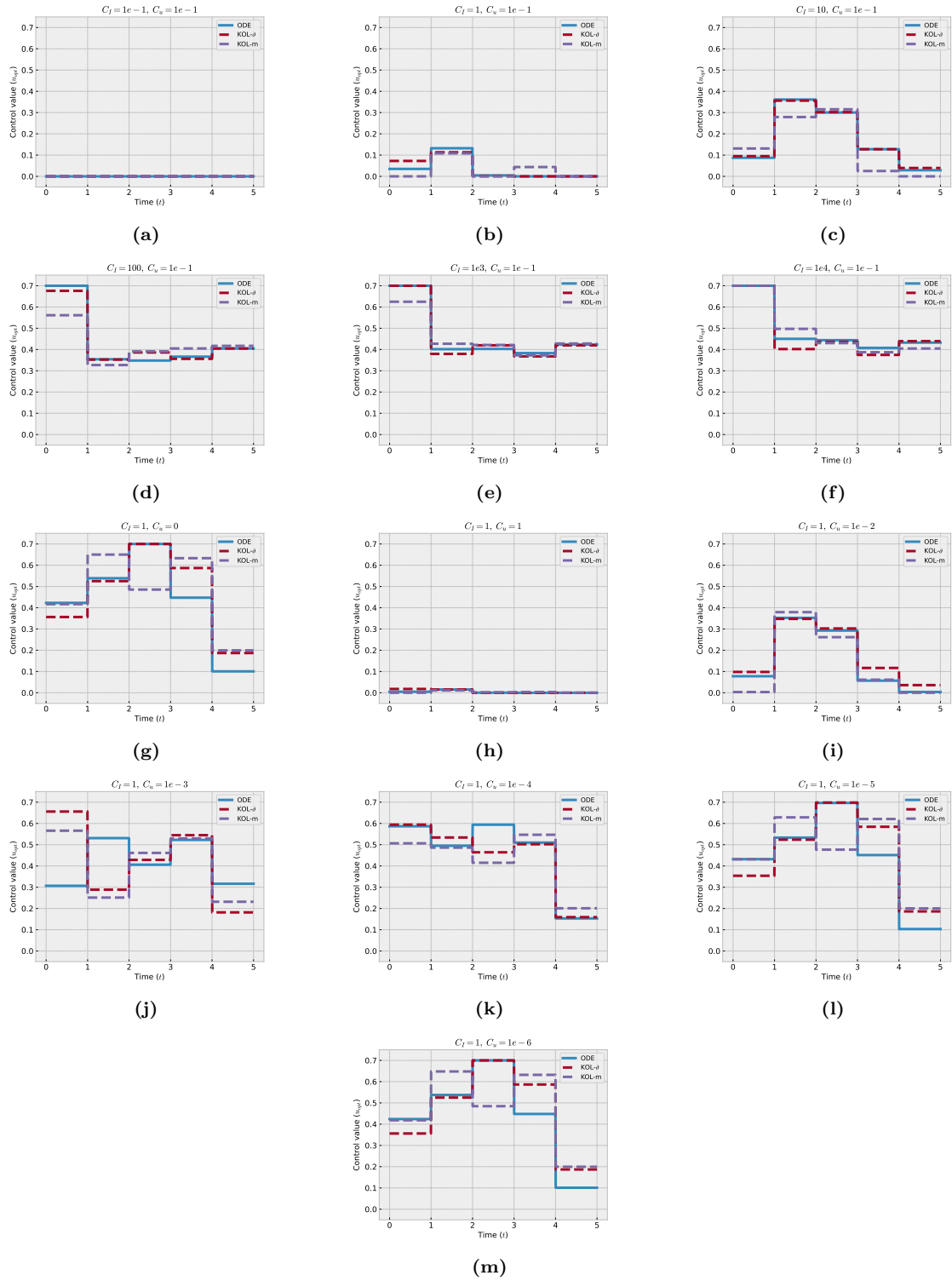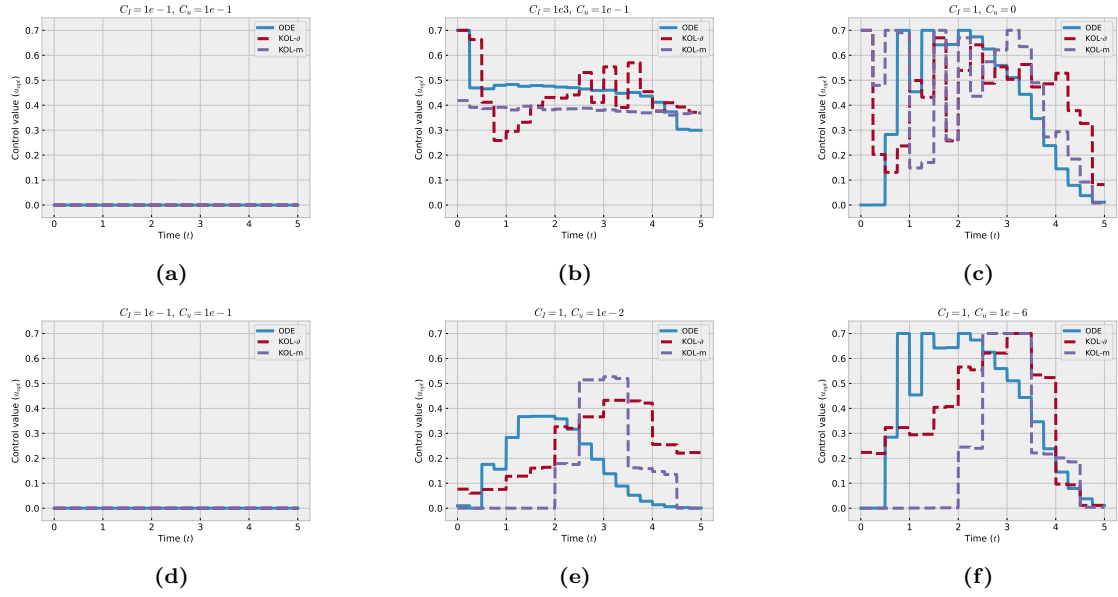
Fig 11. Optimal controls for the three optimal control problems fixing $N = 20$.
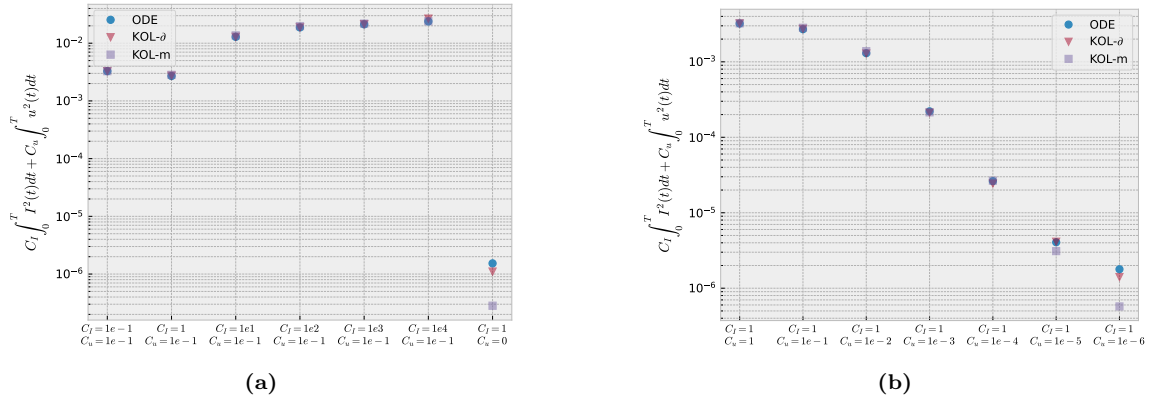

Fig 12. Cost functionals at the optimal control for different C_I and C_u ($N = 10$). (a) We consider different orders of magnitude for C_I, keeping $C_u = 1e - 1$. (b) We consider different orders of magnitude for C_u, keeping $C_I = 1$.

Fig 13. Cost functionals at the optimal control for different C_I and C_u ($N = 20$). (a) We consider different orders of magnitude for C_I, keeping $C_u = 1e - 1$. (b) We consider different orders of magnitude for C_u, keeping $C_I = 1$.
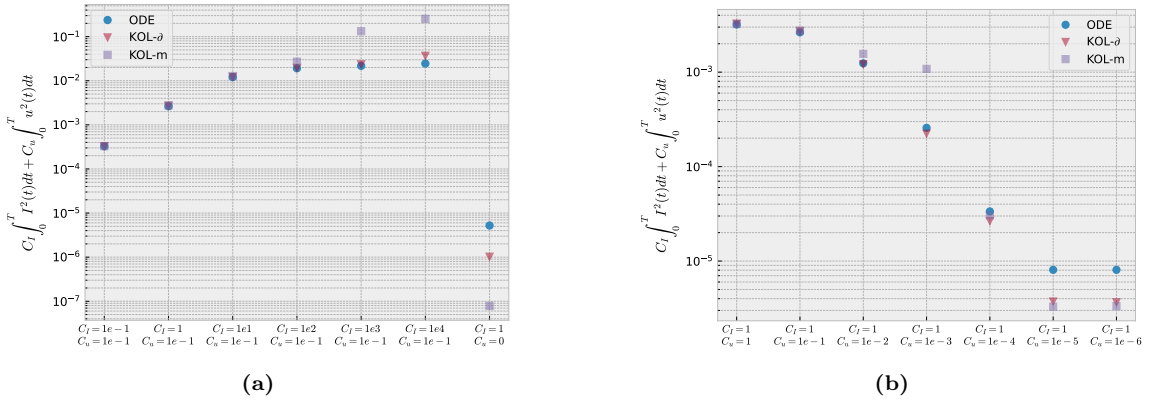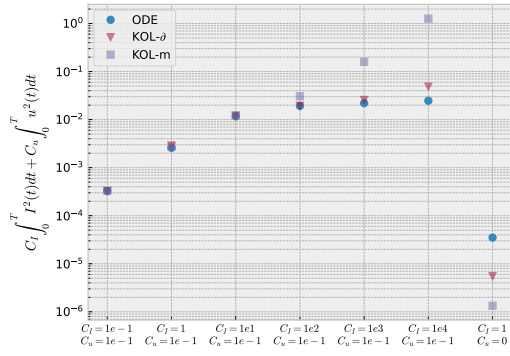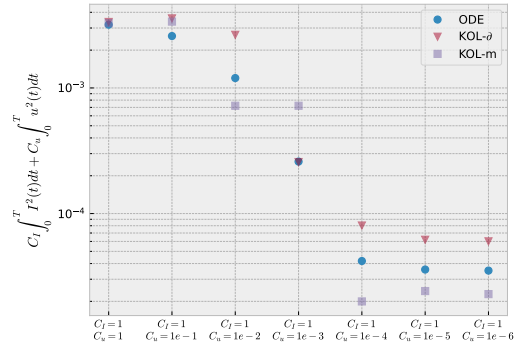
References

[1] Martcheva M. An introduction to mathematical epidemiology. vol. 61. Springer; 2015.

[2] Brauer F, Castillo-Chavez C, Feng Z, et al. Mathematical models in epidemiology. vol. 32. Springer; 2019.

[3] Giordano G, Blanchini F, Bruno R, Colaneri P, Di Filippo A, Di Matteo A, et al. Modelling the COVID-19 epidemic and implementation of population-wide interventions in Italy. Nature medicine. 2020;26(6):855-60.

[4] Parolini N, Ardenghi G, Quarteroni A, et al. Modelling the COVID-19 epidemic and the vaccination campaign in Italy by the SUIHTER model. Infectious Disease Modelling. 2022;7(2):45-63.

[5] Bertuzzo E, Mari L, Pasetto D, Miccoli S, Casagrandi R, Gatto M, et al. The geography of COVID-19 spread in Italy and implications for the relaxation of confinement measures. Nature communications. 2020;11(1):42-64.

[6] Gozzi N, Chinazzi M, Davis JT, Mu K, Pastore y Piontti A, Ajelli M, et al. Anatomy of the first six months of COVID-19 vaccination campaign in Italy. PLoS Computational Biology. 2022;18(5):e1010146.

[7] He M, Tang S, Xiao Y. Combining the dynamic model and deep neural networks to identify the intensity of interventions during COVID-19 pandemic. PLOS Computational Biology. 2023;19(10):e1011535.

[8] Ziarelli G, Parolini N, Verani M, Quarteroni A, et al. Optimized numerical solutions of SIRDVW multiage model controlling SARS-CoV-2 vaccine roll out: An application to the Italian scenario. Infectious Disease Modelling. 2023;8(3):672-703.

[9] Lemaitre JC, Pasetto D, Zanon M, Bertuzzo E, Mari L, Miccoli S, et al. Optimal control of the spatial allocation of COVID-19 vaccines: Italy as a case study. PLoS computational biology. 2022;18(7):e1010237.

[10] Britton T, Leskelä L. Optimal intervention strategies for minimizing total incidence during an epidemic. SIAM Journal on Applied Mathematics. 2023;83(2):354-73.

[11] Dimarco G, Toscani G, Zanella M. Optimal control of epidemic spreading in the presence of social heterogeneity. Philosophical Transactions of the Royal Society A. 2022;380(2224):20210160.

[12] Morris DH, Rossine FW, Plotkin JB, Levin SA. Optimal, near-optimal, and robust epidemic control. Communications Physics. 2021;4(1):78.

[13] Bolzoni L, Bonacini E, Soresina C, Groppi M. Time-optimal control strategies in SIR epidemic models. Mathematical biosciences. 2017;292:86-96.

[14] Lu L, Jin P, Pang G, Zhang Z, Karniadakis GE. Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators. Nature machine intelligence. 2021;3(3):218-29.

[15] Goswami S, Yin M, Yu Y, Karniadakis GE. A physics-informed variational Deep-ONet for predicting crack path in quasi-brittle materials. Computer Methods in Applied Mechanics and Engineering. 2022;391:114587.

[16] Li Z, Kovachki N, Azizzadenesheli K, Liu B, Bhattacharya K, Stuart A, et al. Fourier neural operator for parametric partial differential equations. arXiv preprint arXiv:201008895. 2020.

[17] Batlle P, Darcy M, Hosseini B, Owhadi H. Kernel methods are competitive for operator learning. Journal of Computational Physics. 2024;496:112549.

[18] Genton MG. Classes of kernels for machine learning: a statistics perspective. Journal of machine learning research. 2001;2(Dec):299-312.

[19] Lee J, Schoenholz S, Pennington J, Adlam B, Xiao L, Novak R, et al. Finite versus infinite neural networks: an empirical study. Advances in Neural Information Processing Systems. 2020;33:15156-72.

[20] Jacot A, Gabriel F, Hongler C. Neural tangent kernel: Convergence and generalization in neural networks. Advances in neural information processing systems. 2018;31.

[21] Arora S, Du SS, Hu W, Li Z, Salakhutdinov RR, Wang R. On exact computation with an infinitely wide neural net. Advances in neural information processing systems. 2019;32.

[22] Adlam B, Lee J, Padhy S, Nado Z, Snoek J. Kernel regression with infinite-width neural networks on millions of examples. arXiv preprint arXiv:230305420. 2023.

[23] Carmeli C, De Vito E, Toigo A. Vector valued reproducing kernel Hilbert spaces of integrable functions and Mercer theorem. Analysis and Applications. 2006;4(04):377-408.

[24] Fleming WH, Rishel RW. Deterministic and stochastic optimal control. vol. 1. Springer Science & Business Media; 2012.

[25] Parolini N, Dede' L, Antonietti PF, Ardenghi G, Manzoni A, Miglio E, et al. SUIHTER: A new mathematical model for COVID-19. Application to the analysis of the second epidemic outbreak in Italy. Proceedings of the Royal Society A. 2021;477(2253):20210027.

[26] Marziano V, Guzzetta G, Mammone A, Riccardo F, Poletti P, Trentini F, et al. The effect of COVID-19 vaccination in Italy and perspectives for living with the virus. Nature communications. 2021;12(1):7272.

[27] Yong J, Zhou XY. Stochastic controls: Hamiltonian systems and HJB equations. vol. 43. Springer Science & Business Media; 2012.

[28] Sherratt K, Gruson H, Johnson H, Niehus R, Prasse B, Sandmann F, et al. Predictive performance of multi-model ensemble forecasts of COVID-19 across European nations. Elife. 2023;12:e81916.

[29] Meanti G, Carratino L, Rosasco L, Rudi A. Kernel methods through the roof: handling billions of points efficiently. Advances in Neural Information Processing Systems. 2020;33:14410-22.

[30] Owhadi H, Yoo GR. Kernel flows: From learning kernels from data into the abyss. Journal of Computational Physics. 2019;389:22-47.

[31] De Marchi S, Erb W, Marchetti F, Perracchione E, Rossini M. Shape-driven interpolation with discontinuous kernels: Error analysis, edge extraction, and applications in magnetic particle imaging. SIAM Journal on Scientific Computing. 2020;42(2):B472-91.

[32] Gulian M, Raissi M, Perdikaris P, Karniadakis G. Machine learning of space-fractional differential equations. SIAM Journal on Scientific Computing. 2019;41(4):A2485-509.

[33] Novak R, Xiao L, Hron J, Lee J, Alemi AA, Sohl-Dickstein J, et al. Neural Tangents: Fast and Easy Infinite Neural Networks in Python. In: International Conference on Learning Representations; 2019. .

[34] Regazzoni F, Dede L, Quarteroni A. Machine learning for fast and reliable solution of time-dependent differential equations. Journal of Computational physics. 2019;397:108852.

31

[35] Regazzoni F, Salvador M, Dedè L, Quarteroni A. A machine learning method for real-time numerical simulations of cardiac electromechanics. Computer methods in applied mechanics and engineering. 2022;393:114825.

[36] Regazzoni F, Dedè L, Quarteroni A. Machine learning of multiscale active force generation models for the efficient simulation of cardiac electromechanics. Computer Methods in Applied Mechanics and Engineering. 2020;370:113268.

[37] Nocedal J, Wright SJ. Numerical optimization. Springer; 1999.

[38] Virtanen P, Gommers R, Oliphant TE, Haberland M, Reddy T, Cournapeau D, et al. SciPy 1.0: fundamental algorithms for scientific computing in Python. Nature methods. 2020;17(3):261-72.

[39] Du SS, Zhai X, Poczos B, Singh A. Gradient descent provably optimizes over-parameterized neural networks. arXiv preprint arXiv:181002054. 2018.

[40] Lee J, Xiao L, Schoenholz S, Bahri Y, Novak R, Sohl-Dickstein J, et al. Wide neural networks of any depth evolve as linear models under gradient descent. Advances in neural information processing systems. 2019;32.

[41] Schölkopf B, Herbrich R, Smola AJ. A generalized representer theorem. In: International conference on computational learning theory. Springer; 2001. p. 416-26.

[42] Alvarez MA, Rosasco L, Lawrence ND, et al. Kernels for vector-valued functions: A review. Foundations and Trends® in Machine Learning. 2012;4(3):195-266.

Learning epidemic trajectories through Kernel Operator Learning: from modelling to optimal control. Supplementary Information.

S1. Summary of Neural Tangent Kernel

We introduce the Neural Tangent Kernel in the specific context of neural networks minimizing the Mean Square Error in the input dataset. Consider a set of input data (training set)

$$\mathcal{D} = \{(x_1, y_1), (x_2, y_2) \ldots (x_n, y_n)\}, \tag{22}$$

where $\{x_i\}_i \in \mathbb{R}^d$ represents the sequence of input data, and $\{y_i\}_i \in \mathbb{R}$ the respective labels. Let $f(x; \theta)$ be the neural network regressor, whose weights θ need to be trained solving the minimization problem

$$\min_{\theta \in \mathbb{R}^{n_\theta}} L(\theta) = \sum_{i=1}^{n} \frac{1}{2}(f(x_i; \theta) - y_i)^2 = \sum_{i=1}^{n} l(f(x_i; \theta), y_i). \tag{23}$$

We solve the optimization problem via gradient flow, which is the continuous counterpart of the usual full-batch gradient descent method used in machine learning,

$$\partial_\tau \theta(\tau) = -\partial_\theta L(\theta(\tau)) = -\sum_{i=1}^{n} \partial_\theta f(x_i; \theta(\tau))(f(x_i; \theta(\tau)) - y_i), \tag{24}$$

where τ is a fictious time variable accounting for the iteration progress. Therefore,

$$\begin{aligned}
\partial_\tau f(x_j; \theta(\tau)) &= \partial_\theta f(x_j; \theta(\tau))\partial_\tau \theta(\tau) \\
&= -\sum_{i=1}^{n} \partial_\theta f(x_j; \theta(\tau))^T \partial_\theta f(x_i; \theta(\tau))(f(x_i; \theta(\tau)) - y_i),
\end{aligned} \tag{25}$$

and we define the Neural Tangent Kernel $K : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ as

$$K_\tau(x_i, x_j) := \langle \partial_\theta f(x_j; \theta(\tau)), \partial_\theta f(x_i; \theta(\tau)) \rangle. \tag{26}$$

This kernel is symmetric and positive semi-definite by construction. We remark that the NTK depends on the specific topology of the considered neural network and on the choice of the activation function. Considering wider neural networks, it can be proven [39, 40] that

$$K_\tau(x_i, x_j) \underset{\mathbb{P}}{\to} K^\infty(x_i, x_j), \tag{27}$$

33

where in the infinite-width limit $K^\infty(x_i, x_j) \approx K_0(x_i, x_j)$, meaning that the average infinite-width limiting NTK is close to the NTK computed with weights and biases at the initialization. Therefore, during the training processes of wide neural networks the trajectories of the cost functional ruled by K_τ are close to the approximated linearized ones (ruled by K_∞).

S2. KOL: algebraic derivation

We set the learning problem in the specific Reproducing Kernel Hilbert Spaces (RKHS) framework, so to employ the tools deriving from the Kernel Regression theory, thus enhancing the computational efficiency and capitalizing on the linearity of the resulting functional framework. Let $\mathcal{U}$ be an RKHS of functions $u : \Omega \to \mathbb{R}$ where the kernel introduced is $Q : \Omega \times \Omega \to \mathbb{R}$, and $\mathcal{V}$ is an RKHS of functions $v : D \to \mathbb{R}$ endowed with the following kernel $K : D \times D \to \mathbb{R}$. Then, ψ and χ are defined as optimal recovery maps as

$$\begin{aligned}
\psi(U) &:= \arg\min_{w \in \mathcal{U}} \|w\|_Q \ \text{ s.t. } \ \phi(w) = U, \\
\chi(V) &:= \arg\min_{w \in \mathcal{V}} \|w\|_K \ \text{ s.t. } \ \phi(w) = V,
\end{aligned} \tag{28}$$

with $\|w\|_Q = \sqrt{\langle\langle w, Q(\cdot, t)\rangle, \langle w, Q(\cdot, t)\rangle\rangle}$ and $\|w\|_K = \sqrt{\langle\langle w, K(\cdot, t)\rangle, \langle w, K(\cdot, t)\rangle\rangle}$ are the norms defined in the respective Hilbert spaces exploiting the kernel reproducibility property. Assuming that ϕ and φ are pointwise evaluations at specific collocation points as previously discussed, the optimal recovery maps have explicit closed forms deriving from kernel interpolation theory [41]:

$$\psi(U)(t) = Q(t, T)Q(T, T)^{-1}U, \ \ \chi(V)(t) = K(t, T)K(T, T)^{-1}V, \tag{29}$$

where $Q(T, T)$ and $K(T, T)$ are symmetric and definite positive matrices such that $Q(T, T)_{ij} = Q(T_i, T_j)$ and $K(T, T)_{ij} = K(T_i, T_j)$, whilst $Q(t, T)_i = Q(t, T_i)$ and $K(t, T)_i = K(t, T_i)$ represent row-vectors. Following the Kernel-based approach proposed in [17], the operator learning scheme simplifies to determine an approximation of the mapping between two finite-dimensional Euclidean spaces $f^\dagger : \mathbb{R}^n \to \mathbb{R}^n$, defined as

$$f^\dagger := \varphi \circ \mathcal{G} \circ \psi, \tag{30}$$

cf. Figure 1, where the reconstruction maps $\psi : \mathbb{R}^n \to \mathcal{U}$ and $\chi : \mathbb{R}^n \to \mathcal{V}$ need to be properly defined.

Finally, we aim at approximating the $f^\dagger$ function through the use of a vector-valued kernel. Indeed, consider $\Gamma : \mathbb{R}^n \times \mathbb{R}^n \to \mathcal{L}(\mathbb{R}^n)$ the matrix valued kernel following the notation of [42]. We call the RKHS induced by the considered kernel $\mathcal{H}_\Gamma$, and the respective induced norm is $\|\cdot\|_{\mathcal{H}_\Gamma}$. Hence, $f^\dagger$ can be approximated with the map $\bar{f}$ solving the following optimization problem:

$$\bar{f} := \arg\min_{f \in \mathcal{H}_\Gamma} \|f\|_\Gamma \ \text{ s.t. } \ f(\phi(u_i)) = \varphi(v_i), \ i = 1, 2 \dots N. \tag{31}$$

We introduce the following block vectors $\mathbf{U}$ and $\mathbf{V}$ as

$$\mathbf{U} = \begin{pmatrix} U_1 \\ U_2 \\ \vdots \\ U_{N-1} \\ U_N \end{pmatrix} \in \mathbb{R}^{nN}, \ \mathbf{V} = \begin{pmatrix} V_1 \\ V_2 \\ \vdots \\ V_{N-1} \\ V_N \end{pmatrix} \in \mathbb{R}^{nN}, \tag{32}$$

where $\{U_i\}_i := \{\phi(u_i)\}_i, \ \forall \, u_i \in \mathcal{U}$ and $\{V_i\}_i := \{\varphi(v_i)\}_i, \ \forall \, v_i \in \mathcal{V}$. Then, we define with a slight abuse of notation the matrix $\Gamma : \mathbb{R}^{nN} \times \mathbb{R}^{nN} \to \mathbb{R}^{nN \times nN}$ as

$$\Gamma(\mathbf{U}, \mathbf{U}) = \begin{bmatrix} \Gamma(U_1, U_1) & \Gamma(U_1, U_2) & \dots & \Gamma(U_1, U_N) \\ \Gamma(U_2, U_1) & \Gamma(U_2, U_2) & \dots & \Gamma(U_2, U_N) \\ \vdots & \vdots & & \vdots \\ \Gamma(U_N, U_1) & \Gamma(U_N, U_2) & \dots & \Gamma(U_N, U_N) \end{bmatrix}, \tag{33}$$

where each $\Gamma(U_i, U_j) \in \mathbb{R}^{n \times n}, \ \forall i, j = 1, 2 \dots N$ is an independent block, and the following matrix

$$\Gamma(U, \mathbf{U}) = \begin{bmatrix} \Gamma(U, U_1) & \Gamma(U, U_2) & \dots & \Gamma(U, U_N) \end{bmatrix} \in \mathbb{R}^{n \times nN}. \tag{34}$$

In this work we assume to deal with uncorrelated input samples, therefore the analysis can be simplified by relying on diagonal kernels for Γ. In particular, let $S : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ be a scalar kernel (see Section 2.1 where we compare different practical choices for S). Thus, each block in (33) is a diagonal block, *i.e.*

$$\Gamma(U_i, U_j) = S(U_i, U_j)I = \begin{bmatrix} S(U_i, U_j) & 0 & \dots & 0 \\ 0 & S(U_i, U_j) & \dots & 0 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \dots & S(U_i, U_j) \end{bmatrix}. \tag{35}$$

Hence, the problem of learning the operator from prescribed input-output couples can be recast as an optimal recovery problem:

$$\bar{f}_j := \arg\min_{g \in \mathcal{H}_S} \|g\|_S \text{ s.t. } g(\phi(u_i)) = \varphi(v_i)_j, \ \forall \, i = 1, 2 \dots N, \ j = 1, 2 \dots n, \tag{36}$$

where the RKHS endowed with the S kernel is named $(\mathcal{H}_S, \|\cdot\|_S)$. In this finite dimensional case, it is possible to employ the fundamental result from kernel theory known as representer theorem [41]. Therefore, each component of the dicrete representation

can be written explicitly as

$$\bar{f}_j(U) = S(U, \mathbf{U})S(\mathbf{U}, \mathbf{U})^{-1}\mathbf{V}_{\cdot,j}, \tag{37}$$

where $S(U, \mathbf{U}) : \mathbb{R}^n \times \mathbb{R}^{nN} \to \mathbb{R}^N$ is a row vector, $S(\mathbf{U}, \mathbf{U}) : \mathbb{R}^{nN} \times \mathbb{R}^{nN} \to \mathbb{R}^{N \times N}$ and $\mathbf{V}_{\cdot,j} = [[V_1]_j, [V_2]_j \ldots [V_N]_j]^T \in \mathbb{R}^N$. Equation (37) in the alternative kernel-methods-like form

$$\bar{f}(U) = \sum_{j=1}^N \underbrace{S(U, U_j)}_{\in \mathbb{R}} \underbrace{\alpha_j}_{\in \mathbb{R}^n}. \tag{38}$$

Indeed, by the representer theorem it holds that

$$V(U) = \Gamma(U, \mathbf{U})\Gamma(\mathbf{U}, \mathbf{U})^{-1}\mathbf{V} = \Gamma(U, \mathbf{U})\boldsymbol{\alpha} = \sum_{j=1}^N \underbrace{\Gamma(U, U_j)}_{\in \mathbb{R}^{n \times n}} \alpha_j = \sum_{j=1}^N \underbrace{S(U, U_j)}_{\in \mathbb{R}} \alpha_j, \tag{39}$$

where

$$\boldsymbol{\alpha} = \begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \underbrace{\alpha_N}_{\in \mathbb{R}^n} \end{pmatrix} = \begin{pmatrix} \alpha_1^1 \\ \alpha_1^2 \\ \vdots \\ \alpha_1^n \\ \alpha_2^1 \\ \alpha_2^2 \\ \vdots \\ \alpha_2^n \\ \vdots \\ \alpha_N^1 \\ \alpha_N^2 \\ \vdots \\ \alpha_N^n \end{pmatrix} \in \mathbb{R}^{nN} \text{ s.t } \Gamma(\mathbf{U}, \mathbf{U})\boldsymbol{\alpha} = \mathbf{V}. \tag{40}$$

The matrix in system in (40) can be written explicitly as

$\Gamma(\mathbf{U}, \mathbf{U}) =$

$$\begin{bmatrix}
S(U_1,U_1) & 0 & \cdots & 0 & S(U_1,U_2) & 0 & \cdots & 0 & \cdots & S(U_1,U_N) & 0 & \cdots & 0 \\
0 & S(U_1,U_1) & \cdots & 0 & 0 & S(U_1,U_2) & \cdots & 0 & \cdots & 0 & S(U_1,U_N) & \cdots & 0 \\
\vdots & & & \vdots & \vdots & & & \vdots & \cdots & \vdots & & & \vdots \\
0 & 0 & \cdots & S(U_1,U_1) & 0 & 0 & \cdots & S(U_1,U_2) & \cdots & 0 & 0 & \cdots & S(U_1,U_N) \\
S(U_2,U_1) & 0 & \cdots & 0 & S(U_2,U_2) & 0 & \cdots & 0 & \cdots & S(U_2,U_N) & 0 & \cdots & 0 \\
0 & S(U_2,U_1) & \cdots & 0 & 0 & S(U_2,U_2) & \cdots & 0 & \cdots & 0 & S(U_2,U_N) & \cdots & 0 \\
\vdots & & & \vdots & \vdots & & & \vdots & \cdots & \vdots & & & \vdots \\
0 & 0 & \cdots & S(U_2,U_1) & 0 & 0 & \cdots & S(U_2,U_2) & \cdots & 0 & 0 & \cdots & S(U_2,U_N) \\
\vdots & & & \vdots & \vdots & & & \vdots & \cdots & \vdots & & & \vdots \\
S(U_N,U_1) & 0 & \cdots & 0 & S(U_N,U_2) & 0 & \cdots & 0 & \cdots & S(U_N,U_N) & 0 & \cdots & 0 \\
0 & S(U_N,U_1) & \cdots & 0 & 0 & S(U_N,U_2) & \cdots & 0 & \cdots & 0 & S(U_N,U_N) & \cdots & 0 \\
\vdots & & & \vdots & \vdots & & & \vdots & \cdots & \vdots & & & \vdots \\
0 & 0 & \cdots & S(U_N,U_1) & 0 & 0 & \cdots & S(U_N,U_2) & \cdots & 0 & 0 & \cdots & S(U_N,U_N)
\end{bmatrix} \tag{41}$$

37

Hence, we can reorder the system as

$$
\begin{cases}
S(U_1, U_1)\alpha_1^1 + S(U_1, U_2)\alpha_2^1 + S(U_1, U_3)\alpha_3^1 + \ldots + S(U_1, U_N)\alpha_N^1 = V_1^1 \\
S(U_2, U_1)\alpha_1^1 + S(U_2, U_2)\alpha_2^1 + S(U_2, U_3)\alpha_3^1 + \ldots + S(U_2, U_N)\alpha_N^1 = V_2^1 \\
S(U_1, U_1)\alpha_1^1 + S(U_1, U_2)\alpha_2^1 + S(U_1, U_3)\alpha_3^1 + \ldots + S(U_1, U_N)\alpha_N^1 = V_1^1 \\
S(U_3, U_1)\alpha_1^1 + S(U_3, U_2)\alpha_2^1 + S(U_3, U_3)\alpha_3^1 + \ldots + S(U_3, U_N)\alpha_N^1 = V_3^1 \\
\vdots \\
S(U_N, U_1)\alpha_1^1 + S(U_N, U_2)\alpha_2^1 + S(U_N, U_3)\alpha_3^1 + \ldots + S(U_N, U_N)\alpha_N^1 = V_N^1
\end{cases}
, \quad (42)
$$

and we can define the matrix

$$
\mathbf{S}(\mathbf{U}, \mathbf{U}) =
\begin{bmatrix}
S(U_1, U_1) & S(U_1, U_2) & S(U_1, U_3) & \ldots & S(U_1, U_N) \\
S(U_2, U_1) & S(U_2, U_2) & S(U_2, U_3) & \ldots & S(U_2, U_N) \\
\vdots & & & & \vdots \\
S(U_N, U_1) & S(U_N, U_2) & S(U_N, U_3) & \ldots & S(U_N, U_N)
\end{bmatrix}
\in \mathbb{R}^{N \times N}. \quad (43)
$$

We get that

$$
\alpha_i =
\begin{pmatrix}
[\mathbf{S}(\mathbf{U}, \mathbf{U})^{-1}\mathbf{V}_{\cdot,1}]_i \\
[\mathbf{S}(\mathbf{U}, \mathbf{U})^{-1}\mathbf{V}_{\cdot,2}]_i \\
\vdots \\
[\mathbf{S}(\mathbf{U}, \mathbf{U})^{-1}\mathbf{V}_{\cdot,n}]_i
\end{pmatrix}, \quad \forall i = 1, 2 \ldots N. \quad (44)
$$

Finally, we rewrite equation (39) as

$$
\bar{f}_j(U) = V(U)_j = \left[\sum_{i=1}^{N} S(U, U_i)\alpha_i \right]_j = \left[\sum_{i=1}^{N} S(U, U_i)
\begin{pmatrix}
[\mathbf{S}(\mathbf{U}, \mathbf{U})^{-1}\mathbf{V}_{\cdot,1}]_i \\
[\mathbf{S}(\mathbf{U}, \mathbf{U})^{-1}\mathbf{V}_{\cdot,2}]_i \\
\vdots \\
\hdashline
[\mathbf{S}(\mathbf{U}, \mathbf{U})^{-1}\mathbf{V}_{\cdot,j}]_i \\
\vdots \\
[\mathbf{S}(\mathbf{U}, \mathbf{U})^{-1}\mathbf{V}_{\cdot,n}]_i
\end{pmatrix}
\right]_j,
$$
$$(45)$$

that, by rearranging the terms is exactly equation (37).

Combining equations (29) and (37) we obtain the approximation operator as

$$
\bar{\mathcal{G}} := \chi \circ \bar{f} \circ \phi. \quad (46)
$$

38

Finally, recalling the definition of the operator in equation (38), $\bar{\mathcal{G}} : \mathcal{U} \to \mathcal{V}$ is

$$\bar{\mathcal{G}}(u) = \chi \left(\sum_{j=1}^{N} S(\phi(u), U_j) \alpha_j \right). \tag{47}$$

MOX Technical Reports, last issues

Dipartimento di Matematica

Politecnico di Milano, Via Bonardi 9 - 20133 Milano (Italy)

31/2024   Gambarini, M.; Agate, G.; Ciaramella, G.; Miglio, E.; Maran, S.
Modeling and optimization for arrays of water turbine OWC devices

30/2024   Gregorio, C.; Rea, F.; Ieva, F.; Scagnetto, A.; Indennidade, C.; Cappelletto, C.; Di Lenarda, A.;
Barbati, G.
Flexible approaches based on multi-state models and microsimulation to perform real-world cost-effectiveness analyses: an application to PCSK9-inhibitors

29/2024   Palummo, A.;, Arnone, E.; Formaggia, L.; Sangalli, L.M.
Functional principal component analysis for incomplete space-time data

28/2024   Magri, M.; Riccobelli, D.
Modelling of initially stressed solids: structure of the energy density in the incompressible limit

27/2024   Antonietti, P.F.; Beirao da Veiga, L.; Botti, M.; Vacca, G.; Verani, M.
A Virtual Element method for non-Newtonian fluid flows

26/2024   Bucelli, M.; Regazzoni, F.; Dede', L.; Quarteroni, A.
Robust radial basis function interpolation based on geodesic distance for the numerical coupling of multiphysics problems

25/2024   Enrico Ballini e Luca Formaggia e Alessio Fumagalli e Anna Scotti e Paolo Zunino
Application of Deep Learning Reduced-Order Modeling for Single-Phase Flow in Faulted Porous Media

22/2024   Gatti, F.; de Falco, C.; Fois, M.; Formaggia, L.
A scalable well-balanced numerical scheme for a depth-integrated lava flow model

21/2024   Caldana, M.; Antonietti P. F.; Dede' L.
Discovering Artificial Viscosity Models for Discontinuous Galerkin Approximation of Conservation Laws using Physics-Informed Machine Learning

19/2024   Torzoni, M.; Manzoni, A.; Mariani, S.
A multi-fidelity surrogate model for structural health monitoring exploiting model order reduction and artificial neural networks