EpidemiOptim: A Toolbox for the Optimization of Control Policies in Epidemiological Models

Cédric Colas   CEDRIC.COLAS@INRIA.FR
Inria Flowers, Univ. de Bordeaux; Bordeaux (FR)

Boris Hejblum   BORIS.HEJBLUM@U-BORDEAUX.FR
Inria SISTM, Inserm U1219 Bordeaux Population Health,
Univ. de Bordeaux; Bordeaux (FR)
Vaccine Research Institute; Créteil (FR)

Sébastien Rouillon   SEBASTIEN.ROUILLON@U-BORDEAUX.FR
Univ. de Bordeaux, GREThA UMR 5113;
Bordeaux (FR)

Rodolphe Thiebaut   RODOLPHE.THIEBAUT@INRIA.FR
Inria SISTM, Inserm U1219 Bordeaux Population Health,
Univ. de Bordeaux; Bordeaux (FR)

Pierre-Yves Oudeyer   PIERRE-YVES.OUDEYER@INRIA.FR
Inria Flowers, Univ. de Bordeaux; Bordeaux (FR)
ENSTA Paris Tech; Paris (FR)

Clément Moulin-Frier   CLEMENT.MOULIN-FRIER@INRIA.FR
Inria Flowers, Univ. de Bordeaux; Bordeaux (FR)
ENSTA Paris Tech; Paris (FR)

Mélanie Prague   MELANIE.PRAGUE@U-BORDEAUX.FR
Inria - SISTM, Inserm U1219 Bordeaux Population Health,
Univ. de Bordeaux; Bordeaux (FR)

Abstract

Modeling the dynamics of epidemics helps to propose control strategies based on pharmaceutical and non-pharmaceutical interventions (contact limitation, lockdown, vaccination, etc). Hand-designing such strategies is not trivial because of the number of possible interventions and the difficulty to predict long-term effects. This task can be cast as an optimization problem where state-of-the-art machine learning methods such as deep reinforcement learning might bring significant value. However, the specificity of each domain—epidemic modeling or solving optimization problems—requires strong collaborations between researchers from different fields of expertise. This is why we introduce EpidemiOptim, a Python toolbox that facilitates collaborations between researchers in epidemiology and optimization. EpidemiOptim turns epidemiological models and cost functions into optimization problems via a standard interface commonly used by optimization practitioners (*OpenAI Gym*). Reinforcement learning algorithms based on Q-

Learning with deep neural networks (DQN) and evolutionary algorithms (NSGA-II) are already implemented. We illustrate the use of EpidemiOptim to find optimal policies for dynamical on-off lockdown control under the optimization of the death toll and economic recess using a Susceptible-Exposed-Infectious-Removed (SEIR) model for COVID-19. Using EpidemiOptim and its interactive visualization platform in Jupyter notebooks, epidemiologists, optimization practitioners and others (e.g. economists) can easily compare epidemiological models, costs functions and optimization algorithms to address important choices to be made by health decision-makers. Trained models can be explored by experts and non-experts via a web interface.

1. Introduction

The recent COVID-19 pandemic highlights the destructive potential of infectious diseases in our societies, especially on our health, but also on our economy. To mitigate their impact, scientific understanding of their spreading dynamics coupled with methods quantifying the impact of intervention strategies along with their associated uncertainty, are key to support and optimize informed policymaking. For example, in the COVID-19 context, large-scale population lockdowns were enforced based on analyses and predictions from mathematical epidemiological models (Ferguson et al., 2005, 2006; Cauchemez et al., 2019; Ferguson et al., 2020). In practice, researchers often consider a small number of relatively coarse and pre-defined intervention strategies and run calibrated epidemiological models to predict their impact (Ferguson et al., 2020). This is a difficult problem for several reasons: 1) the space of potential strategies can be large, heterogeneous and multi-scale (Halloran et al., 2008); 2) their impact on the epidemic is often difficult to predict; 3) the problem is multi-objective by essence: it often involves public health objectives like the minimization of the death toll or the saturation of intensive care units, but also societal and economic sustainability. For these reasons, pre-defined strategies are bound to be suboptimal. Thus, a major challenge consists in leveraging more sophisticated and adaptive approaches to identify optimal strategies.

The use of machine learning in epidemiology is a relatively new area of research. Of note, machine learning was not accepted as a Medline Medical Subject Heading (MeSH) term until end of 2016. However, it has the potential to transform epidemiology (Bi et al., 2019). Machine learning approaches have become widely used in medicine, but remain of limited use in population health (Flaxman & Vos, 2018). In such context, the optimal control of epidemiological models mainly involved Pontryagin's maximum principle or dynamic programming approaches (Pasin et al., 2018). The first approach has recently been used in the context of the COVID-19 pandemics (Lemecha Obsu & Feyissa Balcha, 2020; Perkins & Espana, 2020). However, more flexible approaches are required, especially to tackle the inherent stochasticity of the spread of infectious diseases. Hence, approaches based on reinforcement learning are emerging (Ohi et al., 2020).

Machine learning can indeed be used for the optimization of such control policies, with methods ranging from deep reinforcement learning to multi-objective evolutionary algorithms. In other domains, they have proven efficient at finding robust control policies, especially in high-dimensional non-stationary environments with uncertainty and partial

observation of the state of the system (Deb et al., 2007; Mnih et al., 2015; Silver et al., 2017; Haarnoja et al., 2018; Kalashnikov et al., 2018; Hafner et al., 2019).

Yet, researchers in epidemiology, public health, economics, and machine learning evolve in communities that rarely cross, and often use different tools, formalizations and terminologies. We believe that tackling the major societal challenge of epidemic mitigation requires interdisciplinary collaborations organized around operational scientific tools and goals, that can be used and contributed by researchers of these various disciplines. To this end, we introduce EpidemiOptim, a Python toolbox that provides a framework to facilitate collaborations between researchers in epidemiology, economics and machine learning.

EpidemiOptim turns epidemiological models and cost functions into optimization problems via the standard OpenAI Gym (Brockman et al., 2016) interface that is commonly used by optimization practitioners. Conversely, it provides epidemiologists and economists with easy-to-use access to a variety of deep reinforcement learning and evolutionary algorithms, capable of handling different forms of multi-objective optimization under constraints. Thus, EpidemiOptim facilitates the independent update of models by specialists of each topic, while enabling others to leverage implemented models to conduct experimental evaluations. We illustrate the use of EpidemiOptim to find optimal policies for dynamical on-off lockdown control under the optimization of the death toll and economic recess using an extended Susceptible-Exposed-Infectious-Removed (SEIR) model for COVID-19 from Prague et al. (2020).

1.1 Related Work

We can distinguish two main lines of contributions concerning the optimization of intervention strategies for epidemic response. On the one hand, several contributions focus on providing guidelines and identifying the range of methods available to solve the problem. For example, Yáñez et al. (2019) framed the problem of finding optimal intervention strategies to control the disease spread as a reinforcement learning problem; Alamo et al. (2020) provided a road-map that goes from the access to data sources to the final decision-making step; and Shearer et al. (2020) highlighted that a decision model for epidemic response cannot capture all of the social, political, and ethical considerations that these decisions impact. These contributions reveal a major challenge for the community: developing tools that can be easily used, configured and interpreted by decision-makers. On the other hand, computational contributions proposed actual concrete implementations of such optimization processes. These contributions mostly differ by their definition of epidemiological models (e.g. SEIR: Yaesoubi et al., 2020, agent-based models: Chandak et al., 2020 or a combination of both: Kompella et al., 2020), of optimization methods (e.g. deterministic rules: Tarrataca et al., 2020, Bayesian optimization: Chandak et al., 2020, Deep RL: Arango & Pelov, 2020; Kompella et al., 2020, optimal control: Charpentier et al., 2020, evolutionary optimization: Miikkulainen et al., 2020 or game-theoretic analysis: Elie et al., 2020), of cost functions (e.g. fixed weighted sum of health and economic costs: Arango & Pelov, 2020; Kompella et al., 2020, possibly adding constraints on the school closure budget: Libin et al., 2020, or multi-objective optimization: Miikkulainen et al., 2020), of state and action spaces (e.g. using the entire observed epidemic history: Yaesoubi et al., 2020 or an im-
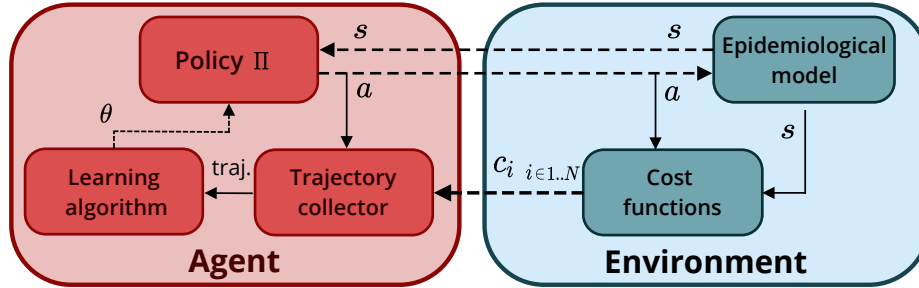
Figure 1: **Epidemic control as an optimization problem.** s, a, c_i refer to environment states, control actions, and the ith cost, while *traj.* is their collection over an episode. Dashed arrows match the input and output of the OpenAI Gym *step* function.

age of the disease outbreak to capture the spatial relationships between locations: Probert et al., 2019), as well as methods for representing the model decisions in a format suitable to decision-makers (e.g. simple summary state representation: Probert et al., 2019 or real-time surveillance data with decision rules: Yaesoubi & Cohen, 2016). See Appendix A for a detailed description of the aforementioned papers.

Given this high diversity of potential methods in the field, our approach aims at providing a standard toolbox facilitating the comparison of different configurations along the aforementioned dimensions in order to assist decision-makers in the evaluation of the range of possible intervention strategies.

1.2 Contributions

This paper makes three contributions. First, we formalize the coupling of epidemiological models and optimization algorithms with a particular focus on the multi-objective aspect of such problems (Section 2). Second, based on this formalization, we introduce the EpidemiOptim library, a toolbox that integrates epidemiological models, cost functions, optimization algorithms and experimental tools to easily develop, study and compare epidemic control strategies (Section 3). Third, we demonstrate the utility of the EpidemiOptim library by conducting a case study on the optimization of lockdown policies for the COVID-19 epidemic (Section 4). We use a recent epidemiological model grounded on real data, cost functions based on a standard economical model of GDP loss, as well as state-of-the-art optimization algorithms, all included in the EpidemiOptim library. This is, to our knowledge, the first contribution that provides a comparison of different optimization algorithm performances for the control of intervention strategies on the same epidemiological model. The user of the EpidemiOptim library can interact with the trained policies via a website based on Jupyter notebooks at `https://epidemioptim.bordeaux.inria.fr/`, exploring the space of cost functions (health cost x economic cost) for a variety of algorithms. The code is made available anonymously at `https://github.com/flowersteam/EpidemiOptim`. Note that our contributions do not include any new optimization algorithm or any epidemiological model.

2. The Epidemic Control Problem as an Optimization Problem

In an *epidemic control problem*, the objective is to find an optimal control strategy to minimize some cost (e.g. health and/or economic cost) related to the evolution of an epidemic. We take the approach of reinforcement learning (RL) (Sutton & Barto, 2018): the control policy is seen as a *learning agent*, that interacts with an epidemiological model that is seen as its *learning environment*, see Figure 1. Each run of the epidemic is a *learning episode*, where the agent alternatively interacts with the epidemic through *actions* and observes the resulting *states* of the environment and their associated *costs*. To define this optimization problem, we define three elements: 1) the state and action spaces; 2) the epidemiological model; 3) the cost function.

State and action spaces.   The state s_t describes the information the control policy can use (e.g. epidemiological model states, cumulative costs up to timestep t, etc.) We also need to define which actions a can be exerted on the epidemic, and how they affect its parameters. For instance, enforcing a lockdown can reduce the transmission rate of the epidemic and slow it down.

The epidemiological model.   The epidemiological model is similar to a *transition function* $\mathcal{T}$: it governs the evolution of the epidemic. From the current state s_t of the epidemic and the current action a_t exerted by the agent, it generates the next state s_{t+1}: $\mathcal{T} : \mathcal{S}, \mathcal{A} \rightarrow \mathcal{S}$ where $\mathcal{S}, \mathcal{A}$ are the state and action spaces respectively. Note that this mapping can be stochastic.

Cost functions and constraints.   The design of cost functions is central to the creation of optimization problems. Epidemic control problems can often be framed as multi-objective problems, where algorithms need to optimize for a set of cost functions instead of a unique one. Costs can be defined at the level of timesteps $C(s, a)$ or at the level of an episode $C_e(\text{trajectory})$. This cumulative cost measure can be obtained by summing timestep-based costs over an episode. For mathematical convenience, RL practitioners often use a discounted sum of costs, where future costs are discounted exponentially: $C_e(\text{trajectory}) = \sum_t \gamma^t c(s_t, a_t)$ with discount factor γ (e.g. it avoids infinite sums when the horizon is infinite). When faced with multiple cost functions $C_i|_{i \in [1..N_c]}$, a simple approach is to aggregate them into a unique cost function C, computed as the convex combination of the individual costs: $\bar{C}(s, a) = \sum_i \beta_i \, C_i(s, a)$. Another approach is to optimize for the Pareto front, i.e. the set of non-dominated solutions. A solution is said to be non-dominated if no other solution performs better on all costs and strictly better on at least one. In addition, one might want to define constraints on the problem (e.g. a maximum death toll over a year).

The epidemic control problem as a Markov Decision Process.   Just like traditional RL problems, our epidemic control problem can be framed as a Markov Decision Process: $\mathcal{M} : \{\mathcal{S}, \mathcal{A}, \mathcal{T}, \rho_0, C, \gamma\}$, where ρ_0 is the distribution of initial states, see Yáñez et al. (2019) for a similar formulation.

Handling time limits.   Epidemics are not episodic phenomena, or at least it is difficult to predict their end in advance. Optimization algorithms, however, require episodes of finite length. This enables the algorithm to regularly obtain fresh data by trying new policies in

the environment. This can be a problem, as we do not want the control policy to act as if the end of the episode was the end of the epidemic: if the epidemic is growing rapidly right before the end, we want the control policy to care and react. Evolution algorithms (EAs) cannot handle such behavior as they use episodic data (θ_e, C_e), where θ_e are the policy parameters used for episode e. RL algorithms, however, can be made time unaware. To this end, the agent should not know the current timestep, and should not be made aware of the last timestep (termination signal). Default RL implementations often ignore this fact, see a discussion of time limits in Pardo et al. (2018).

3. The EpidemiOptim Toolbox

This section presents the toolbox's desiderata and organization.

3.1 Toolbox Desiderata

The EpidemiOptim toolbox aims at facilitating collaborations between the different fields interested in the control of epidemics. Such a library should contain state-of-the-art epidemiological models, optimization algorithms and relevant cost functions. It should enable optimization practitioners to bring their newly-designed optimization algorithm, to plug it to interesting environments based on advanced models grounded in epidemiological knowledge and data, and to reliably compare it to other state-of-the-art optimization algorithms. On the other hand, it should allow epidemiologists to bring their new epidemiological model, reuse existing cost functions to form a complete epidemic control problem and use state-of-the-art optimization algorithms to solve it. This requires a modular approach where algorithms, epidemiological models and cost functions are implemented independently with a standardized interface. The toolbox should contain additional tools to facilitate experiment management (result tracking, configuration saving, logging). Good visualization tools would also help the interpretation of results, especially in the context of multi-objective problems. Ideally, users should be able to interact with the trained policies and observe the new optimal strategy after the modification of the relative importance of costs in real-time. Last, the toolbox should enable reproducibility (seed management) and facilitate statistical comparisons of results.

3.2 Toolbox Organization

Overview. The EpidemiOptim toolbox is centered around two modules: the *environment* and the *optimization algorithm*, see Figure 1. To run an experiment, the user can define its own or select the ones contained in the library. Figure 2 presents the main interface of the EpidemiOptim library. The following sections delve into the *environment* and *algorithm* modules.

Environment: epidemiological models. The *Model* module wraps around epidemiological models, and allows to sample new model parameters from a distribution of models, to sample initial conditions (ρ_0) and to run the model for n steps. Currently, the EpidemiOptim library contains a Python implementation of an extended Susceptible-Exposed-Infectious-Removed (SEIR) model fitted to French data, from Prague et al. (2020).

```python
from epidemioptim.environments.models import get_model
from epidemioptim.environments.cost_functions import get_cost_function
from epidemioptim.environments.gym_envs import get_env
from epidemioptim.optimization import get_algorithm
from epidemioptim.configs.get_params import get_params

config = 'dqn'

# Get the configuration
params = get_params(config_id=config)

# Get the epidemiological model
model = get_model(model_id=params['model_id'], params=params['model_params'])

# Get cost function
cost_function = get_cost_function(cost_function_id=params['cost_id'], params=params['cost_params'])

# Create the optimization problem as a Gym-like environment
env = get_env(env_id=params['env_id'], cost_function=cost_function, model=model, sim_horizon=params['sim_horizon'], seed=params['seed'])

# Get DQN algorithm parameterized by beta
algorithm = get_algorithm(algo_id=params['algo_id'], env=env, params=params)

# Run the training loop
algorithm.learn(num_train_steps=params['num_train_steps'])
```

Figure 2: Main running script of the EpidemiOptim library

Environment: cost functions and constraints. In the EpidemiOptim toolbox, each cost function is a separate class that can compute $c_i(s, a)$, normalize the cost to $[0, 1]$ and generate constraints on its cumulative value. The *Multi-Cost* module integrates several cost functions. It computes the list of costs $c_i(s, a)|_{i \in [1..N_c]}$ and an aggregated cost $\bar{c}$ parameterized by the mixing weights $\beta_i|_{i \in [1..N_c]}$. The current version of EpidemiOptim contains two costs: 1) a health cost computed as the death toll of the epidemic; 2) an economic cost computed as the opportunity loss of GDP resulting from either the epidemic itself (illness and death reduce the workforce), or the control strategies (lockdown also reduces the workforce). We define two additional parameterized constraints. For each of the costs, the cumulative value should stay below a threshold. This threshold can vary and is controlled by the agent. For each episode, the agent selects constraints (thresholds) on the two costs and behaves accordingly.

Environment: OpenAI Gym interface as a universal interface. The learning environment defines the optimization problem. We use a framework that has become standard in the Reinforcement Learning community in recent years: the *OpenAI Gym environment* (Brockman et al., 2016). Gym environments have the following interface: First, they have a *reset* function that resets the environment state, effectively starting a new simulation or episode. Second, they have a *step* function that takes as input the next action from the control policy, updates the internal state of the model and returns the new environment state and the cost associated to the transition, see Figure 1. This standardized interface allows optimization practitioners (RL researchers especially) to easily plug any algorithm, facilitating the comparison of algorithms on diverse benchmark environments. In our library, this class wraps around an epidemiological model and a set of cost functions. It also defines the state and action spaces $\mathcal{S}, \mathcal{A}$. So far, it contains the *EpidemicDiscrete-v0* environment, a Gym-like environment based on the epidemiological model and the bi-objective cost function described above. Agents decide every week whether to enforce a partial lock-

down (binary action) that results in a decreased transmission rate. Note that the step size (here one week) can easily be modified.

Optimization algorithms. The optimization algorithm, or *algorithm* for short, trains learning agents to minimize cumulative functions of costs. Learning agents are characterized by their control policy: a function that takes the current state of the environment as input and produces the next action to perform: $\Pi_\theta : \mathcal{S} \rightarrow \mathcal{A}$ where Π_θ is parameterized by the vector θ. The objective of the algorithm is, thus, to find the optimal set of parameters θ^* that minimizes the cost. These algorithms collect data by letting the learning agent interact with the environment. Currently, the EpidemiOptim toolbox integrates several RL algorithms based on the famous Deep Q-Network algorithm (Mnih et al., 2015) as well as the state-of-the-art multi-objective EA NSGA-II (Deb et al., 2002).

Utils. We provide additional tools for experimentation: experience management, logging and plots. Algorithms trained on the same environment can be easily compared, e.g. by plotting their Pareto front on the same graph. We also enable users to visualize and interact with trained policies via Jupyter notebooks or a web interface accessible to non-experts (`https://epidemioptim.bordeaux.inria.fr/`). Users can select solutions from the Pareto front, modify the balance between different cost functions and modify the value of constraints set to the agent via sliders. They visualize a run of the resulting policy in real-time. We are also committed to the reproducibility and robustness of our results. For this reason, we integrate into our framework a library for statistical comparisons designed for RL experiments (Colas et al., 2019).

4. Case Study: Lockdown On-Off Policy Optimization for the COVID-19 Epidemic

The SARS-CoV-2 virus was identified on January, 7th 2020 as the cause of a new viral disease named COVID-19. As it rapidly spread globally, the World Health Organization (2020) declared a pandemic on March 11th 2020. Exceptional measures have been implemented across the globe as an attempt to mitigate the pandemic (Kraemer et al., 2020). In many countries, the government response was a total lockdown: self-isolation with social distancing, schools and workplaces closures, cancellation of public events, large and small gatherings prohibition, and travel limitation. In France, the first lockdown lasted 55 days from March, 17th to May, 11th. It led to a decrease in gross domestic product of, at least, 10.1% (Mandel & Veetil, 2020) but allowed to reduce the death toll by an estimated 690,000 [570,000; 820,000] people (Flaxman et al., 2020). A second lockdown lasted from October, 30th to December, 15th, illustrating that on-off lockdown strategies seem to be a realistic policy in a population with low herd immunity and in absence, or with low coverage, of a vaccine. In this case study, we investigate alternative strategies optimized by RL and EA algorithms over one year.

4.1 Environment

The epidemiological model. We use a mathematical structural model to understand the large-scale dynamics of the COVID-19 epidemic. We focus on the extended Susceptible-Exposed-Infectious-Removed (SEIR) model from Prague et al. (2020), which was fitted to
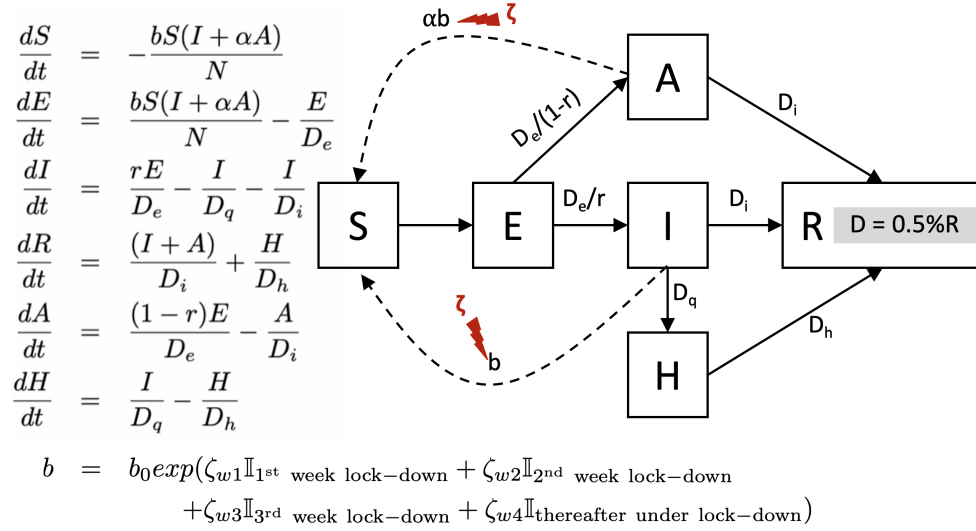
$$\frac{dS}{dt} = -\frac{bS(I + \alpha A)}{N}$$

$$\frac{dE}{dt} = \frac{bS(I + \alpha A)}{N} - \frac{E}{D_e}$$

$$\frac{dI}{dt} = \frac{rE}{D_e} - \frac{I}{D_q} - \frac{I}{D_i}$$

$$\frac{dR}{dt} = \frac{(I + A)}{D_i} + \frac{H}{D_h}$$

$$\frac{dA}{dt} = \frac{(1 - r)E}{D_e} - \frac{A}{D_i}$$

$$\frac{dH}{dt} = \frac{I}{D_q} - \frac{H}{D_h}$$

$$b = b_0 exp(\zeta_{w1}\mathbb{I}_{1^{\text{st}} \text{ week lock-down}} + \zeta_{w2}\mathbb{I}_{2^{\text{nd}} \text{ week lock-down}}$$
$$+ \zeta_{w3}\mathbb{I}_{3^{\text{rd}} \text{ week lock-down}} + \zeta_{w4}\mathbb{I}_{\text{thereafter under lock-down}})$$

Figure 3: A mechanistic model for the dynamics of COVID-19 epidemics, from Prague et al. (2020).

French data. The estimation in this model was region-specific. Here, we use their estimated parameters for the epidemic dynamics in *Île-de-France* (Paris and its surrounding region), see parameters description and mean values in Table 1. The model explains the dynamics between Susceptibles (S), Exposed (E), Ascertained Infectious (I), Removed (R), Non-Ascertained Infectious (A) and Hospitalized (H). The underlying system of differential equations (1) and a schematic view of the mechanistic model can be found in Figure 3. The transmission rate b is modified each week after lockdown with a step function: it decreases from week 1 to week 4 after lockdown and then remains constant for any following week under lockdown. To account for uncertainty in the model parameters, we create a distribution of models resulting from parameter distributions, see details in Appendix Section B and a visualization of 10 sampled models in Appendix Section B.2.

State and action spaces. The state of the agent is composed of the epidemiological states (SEIRAH) normalized by the population size ($N = 12,278,210$), booleans indicating whether the previous and current state are under lockdown, the two cumulative costs and the level of the transmission rate (from week 1 to week 4 and after). As described in the previous section, consecutive weeks of lockdown decrease the transmission b in four stages, while stopping lockdown follows the opposite path (stair-case increase of b). The two cumulative costs are normalized by N and 150 billion euros (B€); respectively, 150 B€ being the approximate maximal cost due to a full year of lockdown. The action occurs weekly and is binary (1 for lockdown, 0 otherwise).

Cost functions. In this case-study, an optimal policy should minimize two costs: a health cost and an economic cost. The health cost is computed as the death toll, it can be evaluated for each environment transition (weekly) or over a full episode (e.g. annually): $C_{\text{h}}(t) = D(t) = 0.005R(t)$.

Param.	Interpretation	Value "Île-de-France"
b_0	Transmission rate of ascertained cases before lockdown	2.23
r	Ascertainement rate	0.043
α	Ratio of transmission between A and I	0.55
D_e	Latent (incubation) period (days)	5.1
D_i	Infectious period (days)	2.3
D_q	Duration from I onset to H (days)	0.36
D_h	Hospitalization period (days)	30
N	Population size	12,278,210
ζ	lockdown effects first (ζ_{w_1}), second (ζ_{w_2}),	(-0.11, -0.50,
	third (ζ_{w_3}) and thereafter weeks ($\zeta_{w'}$)	-1.36, -1.46)
S_0	Initial number of susceptible	$N - E_0 - I_0 - R_0 - A_0 - H_0$
E_0	Initial number of exposed	5004
I_0	Initial number of ascertained	16
R_0	Initial number of removed	0
A_0	Initial number of non ascertained	356
H_0	Initial number of hospitalized	4
D_0	Initial number of death	0
K_0	Initial capital stock in millions euros	1,388,912
L_0	Number of employed individuals in millions	4.58
λ	Activity/employment rate (%)	37.4
Y_0	Initial GDP in million euros / year	424,474
A	Exogenous technical progress	867
$u(t)$	Level of partial unemployment during lockdown (%)	50
γ_k	Capital elasticity	0.37

Table 1: Parameter values for the *Île-de-France* region for the SEIRAH epidemiological model and the economy cost function.

The economic cost is measured in euros as the GDP opportunity cost resulting from diseased or dead workers, as well as people unemployed due to the lockdown policy. At any time t, the GDP can be expressed as a Cobb-Douglas function $F(L(t)) = AK_0^{\gamma_k}L(t)^{1-\gamma_k}$, where K_0 is the capital input before the outbreak that we assume constant during the pandemics, $L(t)$ is the active population employed, γ_k is the capital elasticity and A is the exogenous technical progress (Douglas, 1976). Let $L_0 = \lambda N$ be the initial employed population, where λ is the activity/employment rate. The lockdown leads to partial unemployment and decreases the size of the population able to work (illness, isolation, death), thus $L(t) = (1 - u(t))\lambda(N - G(t))$, where $u(t)$ is the level of partial unemployment due to the lockdown and $G(t) = I(t) + H(t) + 0.005R(t)$ is the size of the ill, isolated or dead population as defined in our SEIRAH model. The economic cost is defined as the difference between the GDP before (Y_0) and after the pandemic, $Y_0 - F(L(t))$ and is given by:

$$C_{\text{eco}}(t) = Y_0 - AK_0^{\gamma_k}((1 - u(t))\lambda(N - G(t)))^{1-\gamma_k}$$

Parameters for values of parameters in the economic model are given in Table 1 and are derived from the National Institute of Statistics and Economical Analysis (Insee, 2020 and Havik et al., 2014). We also consider the use of constraints on the maximum value of cumulative costs.

4.2 Optimization Algorithms

We consider three algorithms: 1) a vanilla implementation of Deep Q-Network (DQN) (Mnih et al., 2015); 2) a goal-conditioned version of DQN inspired from (Schaul et al., 2015); 3) NSGA-II, a state-of-the-art multi-objective evolutionary algorithm (Deb et al., 2002). All algorithms consist in training a control policy mapping the current state s to the next actions a. Each week, the policy observes the state of the pandemic described above and decides whether to enforce a lockdown for the next week. The trained policy is the same for all algorithms: a neural network with one hidden layer of size 64 and ReLU activations. Further background about these algorithms is provided in Appendix Section B.

Optimizing convex combinations of costs: DQN and Goal DQN. DQN only targets one cost function: $\bar{c} = (1-\beta)\, c_{\text{h}} + \beta\, c_{\text{eco}}$. Here the health and economic costs are normalized to have similar ranges (scaled by $1/(65 \times 1e3)$ and $1/1e9$ respectively). We train independent policies for each values of β in $[0., 0.05, .., 1]$. This consists in training a Q-network to estimate the value of performing any given action a in any given state s, where the value is defined as the cumulative negative cost $-\bar{c}$ expected in the future. GOAL DQN, however, trains one policy $\Pi(s, \beta)$ to target any convex combination parameterized by β. We use the method presented in Badia et al. (2020): we train two Q-networks, one for each of the two costs. The optimal action is then selected as the one maximizing the convex combination of the two Q-values:

$$\Pi(s, \beta) = \text{argmax}_a(1 - \beta)\, Q_{\text{s}}(s, a) + \beta\, Q_{\text{eco}}(s, a). \tag{1}$$

By disentangling the two costs, this method facilitates the representation of values that can have different scales and enables automatic transfer for any value of β, see justifications in Badia et al. (2020). During training, agents sample the targeted β uniformly in $[0, 1]$.

Adding constraints with Goal DQN-C. We design and train a variant of GOAL DQN to handle constraints on maximal values for the cumulative costs: GOAL DQN-C. We train a Q-network for each constraint with a cost of 1 each time the constraint is violated, 0 otherwise (Q_h^c, Q_{eco}^c). With a discount factor $\gamma = 1$, this network estimates the number of transitions that are expected to violate the constraints in the future. These additional Q-networks (Q_h^c, Q_{eco}^c) are independent from the Q-networks evaluating future costs (Q_h, Q_{eco}). The action is selected according to Eq. 1 among the actions that are not expected to lead to constraints violations $(Q^c(s, a) < 1)$. If all actions are expected to lead to violations, the agent selects the action that minimizes that violation. During training, agents sample β uniformly, and 50% of the time samples uniformly one of the two constraints in $[1000, 62000]$ for the maximum death toll and $[20, 160]$ for the maximum economic cost, in billions.

EAs to optimize a Pareto front of solutions. We also use NSGA-II, a state-of-the-art multi-objective EA algorithm that trains a population of solutions to optimize a Pareto front of non-dominated solutions (Deb et al., 2002). As others EAs, NSGA-II uses episodic data (θ_e, C_e). To obtain reliable measures of C_e in our stochastic environment, we average costs over 30 evaluations. For this reason, NSGA-II requires more samples than traditional gradient-based algorithms. For a fair comparison, we train two NSGA-II algorithms: one with the same budget of samples as DQN variants (1e6 environment steps), and the other with 15x as much (after convergence).

4.3 Results

Pareto fronts. We aim at providing decision tools to decision-makers and, for this reason, cannot decide ourselves the right value for the mixing parameter β. Instead, we want to present Pareto fronts of Pareto-optimal solutions. Only NSGA-II aims at producing an optimal Pareto front as its result while DQN and GOAL DQN optimize a unique policy. However, one can build Pareto fronts even with DQN variants. We can build populations of policies by aggregating DQN policies trained independently, or evaluating a single GOAL DQN policy on a set of N_{pareto} goals sampled uniformly (100 here). The resulting population can then be filtered to produce the Pareto front. As a result, each run of the DQN algorithm corresponds to 1) training 20 DQN policies for various values of β (the population); 2) evaluating each of them; 3) extracting the Pareto front. For GOAL DQN and GOAL DQN-C, the population is obtained by evaluating the single goal-conditioned policy for 100 uniformly sampled goals.

Figure 4a presents the resulting Pareto fronts for all variants. The Pareto fronts made of several DQN policies (yellow) perform better in the regime of low health cost ($< 10^3$ deaths) where it leads to saving around 20 billion euros over NSGA-II and the GOAL DQN without constraint. Over 25×10^3 deaths, NSGA-II becomes the best algorithm, as it allows to save 10 to 15 billion euros on average for similar health costs. Note that using more objectives might require other visualization tools (see discussion in Ishibuchi et al., 2008; He & Yen, 2015). Running times are about 2 hours for DQN, 4 hours for GOAL DQN and 10 hours for NSGA-II on a single cpu.

In the low health cost regime, the DQN policy attempts to break the first wave of the epidemic, then maintains it low with periodic lockdown (1 or 2 weeks period) until it is
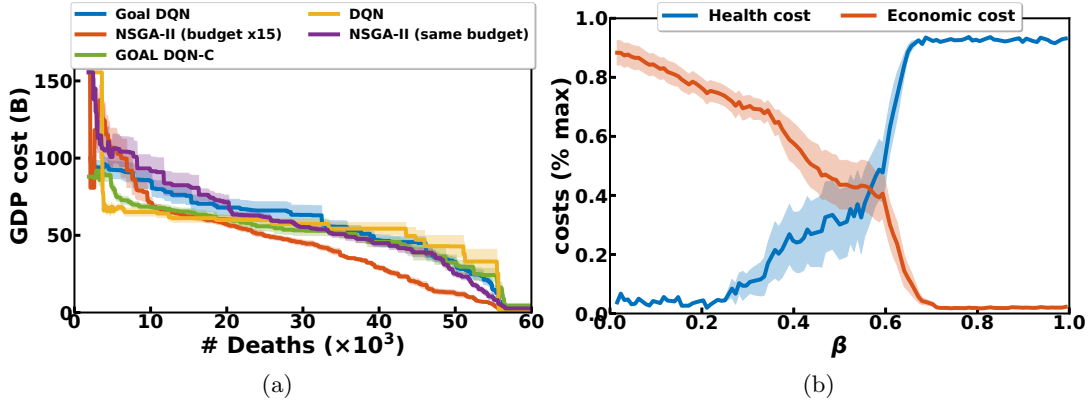
Figure 4: Left: Pareto fronts for various optimization algorithms. The closer to the origin point the better. Right: Evolution of the costs when ɢᴏᴀʟ ᴅǫɴ agents are evaluated on various β. We plot the mean and standard error of the mean (left) and standard deviations (right) over 10 seeds.

sufficiently low. If a second wave appears, it goes back to that strategy, see Figure 5a for an example. ɢᴏᴀʟ ᴅǫɴ with constraints behaves similarly. ɴsɢᴀ-ɪɪ enters a cyclic 1-week period of lockdown but does not stop it even when the number of new cases appears null (Appendix Section C). In the low economic cost regime (below 15 B€), ɴsɢᴀ-ɪɪ waits for the first wave to grow and breaks it by a lockdown of a few weeks (Figure 5b). ᴅǫɴ-based algorithms sometimes find such policies. Policies in the range of health costs between 10^3 and 50×10^3 deaths are usually policies that perform low or high health costs depending on the particular epidemic and its initial state. Appendix Section 4.3 presents comparison statistics and delves into the different strategies and interesting aspects of the four algorithms.

Optimizing convex combinations. When β is low (respectively high), the cost function is closer to the health (respectively economic) cost (see Eq 1). Figure 4b presents the evolution of the two costs when a trained ɢᴏᴀʟ ᴅǫɴ policy (without constraints) is evaluated on different β. As expected, the higher β the lower the health cost and vice-versa. We see that, outside of the $[0.2, 0.8]$ range, the policy falls in extreme regimes: either it always enforces lockdowns (low β) or it never does (high β).

Interactive decision-making. The graphs presented above help make sense of the general performance of the algorithms, in average over multiple runs. However, it is also relevant to look at particular examples to understand a phenomenon. For this purpose, we include visualization Jupyter Notebooks in the EpidemiOptim toolbox. There are three notebooks for ᴅǫɴ, ɢᴏᴀʟ ᴅǫɴs and ɴsɢᴀ-ɪɪ respectively. The two first one help making sense of ᴅǫɴ and ɢᴏᴀʟ ᴅǫɴ algorithms (with or without constraints). After loading a pre-trained model, the user can interactively act on the cost function parameters (mixing weight β or constraints on the maximum values of the cumulative costs). With the ᴅǫɴ algorithm, changing the cost function leads to the selection of a new ᴅǫɴ policy trained for this cost function. With the ɢᴏᴀʟ ᴅǫɴ algorithms, changing the cost function leads to the modification of the goal given to the same policy. In both cases, we run a new simulation and display a new evolution of the epidemic and control strategy. The third notebook helps

to visualize a Pareto front of the NSGA-II algorithm. After loading a pre-trained model, the user can click on solutions shown on the Pareto front, which automatically runs the corresponding model and displays the evolution of the epidemic and control strategy. The user can explore these spaces and visualize the strategies that correspond to the different regimes of the algorithm. We also wrap these notebooks in a web interface accessible to non-experts, available at `https://epidemioptim.bordeaux.inria.fr/`. In addition to running trained models, users can attempt to design their own lockdown policy within the web interface. They can visualize it with respect to the Pareto front of optimized solutions.
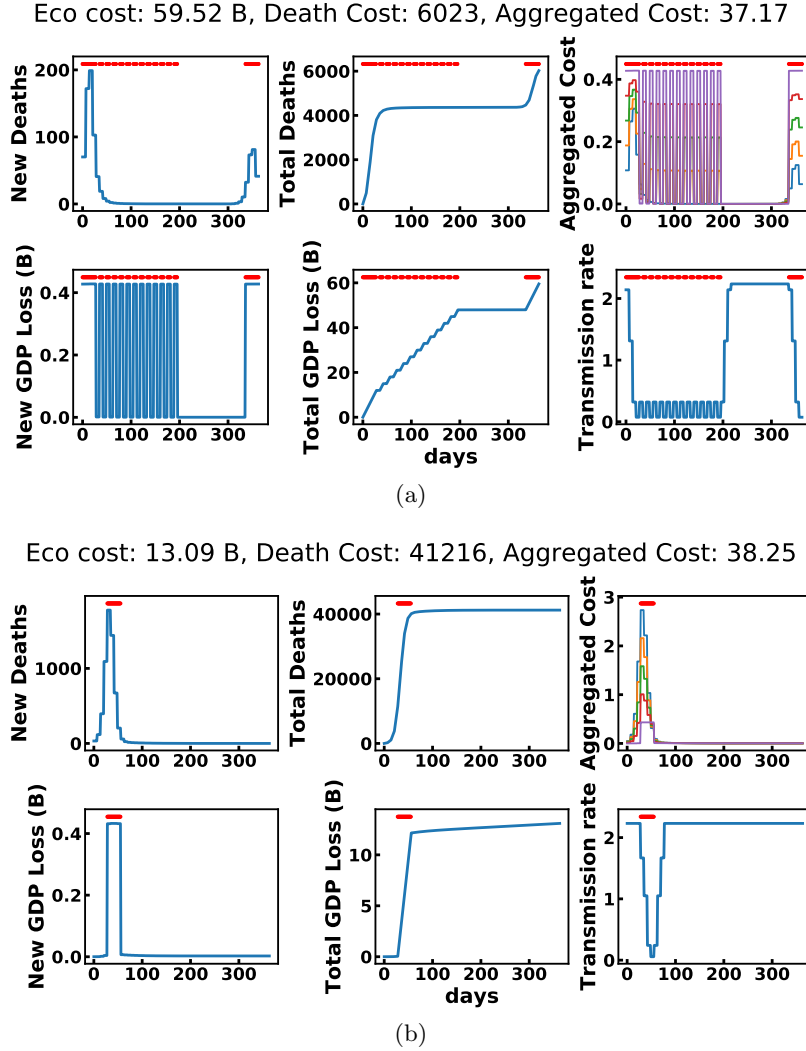
Figure 5: Evolution of cost metrics and strategy over a year of a simulated epidemic with a DQN agent (top) in the low health cost regime and a NSGA-II agent in the low economic cost regime (bottom).

Discussion. This case study proposed to use a set of state-of-the-art algorithms to target the problem of lockdown policy optimization in the context of the COVID-19 epidemic (RL

and EA algorithms). DQN appears to work better in the regime of low health cost, but requires to train multiple policies independently. While the experimenter must train a new DQN policy for each mixture of costs, it only trains one GOAL DQN to target all cost mixtures. Although GOAL DQN is faster to train, it does not bring any performance advantage over DQN. NSGA-II, which is designed for multi-objective optimization, seems to perform better in the regime of low economic cost. Overall, we see emerging two different strategies: one with either a single early lockdown or none at all that seems to rely on herd immunity and one that systematically enforces short-term lockdowns to control the evolution of the epidemic and avoid important epidemic waves. In the last months, few countries have pursued the "herd immunity" strategy and its efficacy is still being questioned (e.g. in Sweden, Orlowski & Goldsmith, 2020). On the other hand, most of the countries that were strongly affected went for the second strategy: enforcing lockdown when the number of cases rises, to contain the epidemic waves. In practice, lockdown is often enforced for at least a month—probably due to political and practical reasons—which contrasts with the shorter lockdowns (one or two weeks) often used by our control policies.

5. Discussion & Conclusion

On the use of automatic optimization for decision-making. We do not think that optimization algorithms should ever replace decision-makers, especially in contexts involving the life and death of individuals such as epidemics. However, we think optimization algorithms can provide decision-makers with useful insights by taking into account long-term effects of control policies. To this end, we believe it is important to consider a wide range of models and optimization algorithms to target the same control problem. This fosters a diversity of methods and reduces the negative impacts of model- and algorithm-induced biases. The EpidemiOptim toolbox is designed for this purpose: it provides a one-stop-shop centralizing both the modeling and the optimization in one place.

The future of EpidemiOptim. This toolbox is designed as a collaborative toolbox that will be extended in the future. To this day, it contains one population-based SEIR epidemiological models and three optimization algorithms. In the future, we plan on extending this library via collaborations with other research labs in epidemiology and machine learning. We plan on integrating agent-based epidemiological models that would allow a broader range of action modalities, and to add a variety of multi-objective algorithms from EAs and RL. We could finally imagine the organization of challenges, where optimization practitioners would compete on a set of epidemic control environments.

Although this toolbox focuses on the use of optimization algorithm to solve epidemic control policies, similar tools could be used in other domains. Indeed, just like epidemics, a large diversity of dynamical systems can be modeled models by ordinary differential equations (e.g. in chemistry, biology, economics, etc.). One could imagine spin-offs of EpidemiOptim for applications to other domains. Furthermore, Chen et al. (2018) recently designed a differentiable ODE solver. Such solvers could be integrated to EpidemiOptim to allow differential learning through the model itself.

Conclusion. This paper introduces EpidemiOptim, a toolbox that facilitates collaboration between epidemiologists, optimization practitioners and others in the study of epidemic

control policies to help decision-making. Of course, others have studied the use of optimization algorithms for the control of epidemics. We see this as a strength, as it shows the relevance of the approach and the importance of centralizing this research in a common framework that allows easy and reproducible comparisons and visualizations. Articulating our understanding of epidemics' evolution with the societal impacts of potential control strategies on the population (death toll, physical and psychological traumas, etc) and the economy will help improve the control of future epidemics.

Acknowledgments

Appendix A. Additional Related Work

Before the current COVID-19 pandemics, Yáñez et al. (2019) framed the problem of finding optimal intervention strategies to control the spread of an epidemic as a reinforcement learning problem, focusing on how to design environments in terms of epidemiological model, intervention strategy, reward function, and state representations. In Alamo et al. (2020), the CONCO-Team (CONtrol COvid-19 Team) provides a detailed SWOT analysis (Strengths, Weaknesses, Opportunities, Threats) and a roadmap that goes from the access to data sources to the final decision-making step, highlighting the interest of standard optimization methods such as Optimal Control Theory, Model Predictive Control, Multi-objective control and Reinforcement Learning. However, as argued in Shearer et al. (2020), a decision model for pandemic response cannot capture all of the social, political, and ethical considerations that impact decision-making. It is therefore a central challenge to propose tools that can be easily used, configured and interpreted by decision-makers.

The aforementioned contributions provide valuable analyses on the range of methods and challenges involved in applying optimization methods to assist decision-making during pandemics events. They do not, however, propose concrete implementations of such systems. Several recent contributions have proposed such implementations. Yaesoubi et al. (2020) applied an approximate policy iteration algorithm of their own (Yaesoubi & Cohen, 2016) on a SEIR model calibrated on the 1918 Influenza Pandemic in San Francisco. An interesting contribution of the paper was the development of a pragmatic decision tool to characterize adaptive policies that combined real-time surveillance data with clear decision rules to guide when to trigger, continue, or stop physical distancing interventions. Arango and Pelov (2020) applied a more standard RL algorithm (Double Deep Q-Network) on a SEIR model of the COVID-19 spread to optimize cyclic lockdown timings. The reward function was a fixed combination of a health objective (minimization of overshoots of ICU bed usage above an ICU bed threshold) and an economic objective (minimization of the time spent under lockdowns). They compared the action policies optimized by the RL algorithm with a baseline on/off feedback control (fixed-rule), and with different parameters (reproductive number) of the SEIR model. Libin et al. (2020) used a more complex SEIR model with coupling between different districts and age groups. They applied the Proximal Policy Optimization algorithm to learn a joint policy to control the districts using a reward function quantifying

the negative loss in susceptibles over one simulated week, with a constraint on the school closure budget. Probert et al. (2019) used Deep Q-Networks (DQN) and Monte-Carlo control on a stochastic, individual-based model of the 2001 foot-and-mouth disease outbreak in the UK. An original aspect of their approach was to define the state at time t using an image of the disease outbreak to capture the spatial relationships between farm locations, allowing the use of convolutional neural networks in DQN. (Kompella et al., 2020) applied the Soft-Actor-Critic algorithm (Haarnoja et al., 2018) on an agent-based epidemiological model with community interactions allowing the spread of the disease to be an emergent property of people's behaviors and the government's policies. Other contributions applied non-RL optimization methods such as deterministic rules (Tarrataca et al., 2020), stochastic approximation algorithms (Yaesoubi et al., 2020), optimal control (Charpentier et al., 2020) or Bayesian optimization (Chandak et al., 2020). This latter paper also proposes a stochastic agent-based model called VIPER (Virus-Individual-Policy-EnviRonment) allowing to compare the optimization results on variations of the demographics and geographical distribution of population. Finally, Miikkulainen et al. (2020) proposed an original approach using Evolutionary Surrogate-assisted Prescription (ESP). In this approach, a recurrent neural network (the Predictor) was trained with publicly available data on infections and NPIs in several countries and applied to predicting how the pandemic will unfold in them in the future. Using the Predictor, an evolutionary algorithm generated a large number of candidate strategies (the Prescriptors) in a multi-objective setting to discover a Pareto front that represents different tradeoffs between minimizing the number of COVID-19 cases, as well as the number and stringency of NPIs (representing economic impact). Elie et al. (2020) propose another original approach based on a game-theoretic analysis, showing how a Mean Field Nash equilibrium can be reached among the population and quantifying the impact of individual choice between epidemic risk and other unfavorable outcomes.

As we have seen, existing contributions widely differ in their definition of epidemiological models, optimization methods, cost functions, state and action spaces, as well as methods for representing the model decisions in a format suitable to decision-makers. Our approach aims at providing a standard toolbox facilitating the comparison of different configurations along these dimensions to assist decision-makers in the evaluation and the interpretation of the range of possible intervention strategies.

Appendix B. Additional Methods

B.1 Optimization Algorithms

The Deep Q-Network algorithm. DQN was introduced in (Mnih et al., 2015) as an extension of the Q-learning algorithm (Watkins & Dayan, 1992) for policies implemented by deep neural networks. The objective is to train a Q-function to approximate the value of a given state-action pair (s, a). This value can be understood as the cumulative measure of reward that the agent can expect to collect in the future by performing action a_t now and following an optimal policy afterward. In the Q-learning algorithm, the Q-function is trained by:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha[r_{t+1} + \gamma \max_a Q(s_{t+1}, a) - Q(s_t, a_t)].$$

Here, α is a learning rate, $[r_{t+1} + \gamma \max_a Q(s_{t+1}, a) - Q(s_t, a_t)]$ is called the temporal difference (TD) error with $r_{t+1} + \gamma \max_a Q(s_{t+1}, a)$ the target. Indeed, the value of the current state-action pair $Q(s_t, a_t)$ should be equal to the immediate reward r_{t+1} plus the value of the next state $Q(s_{t+1}, a_{t+1})$ discounted by the discount factor γ. Because we assume we behave optimally after the first action, action a_{t+1} should be the one that maximizes $Q(s_{t+1}, a)$. Once this function is trained, the agent simply needs to take the action that maximize the Q-value in its current state: $a^* = \max_a Q(s_t, a)$.

Deep Q-Network only brings a few mechanisms to enable the use of large policies based on deep neural networks see Mnih et al. (2015) for details.

Goal-parameterized DQNs. Schaul et al. (2015) introduced Universal Value Function Approximators: the idea that Deep RL agents could be trained to optimize a whole space of cost functions instead of a unique one. In their work, cost functions are parameterized by latent codes called *goals*. An agent in a maze can decide to target any goal position within that maze. The cost function is then a function of that goal. To this end, the agent is provided with its current goal. In the traditional setting, the goal was concatenated to the state of the agent. In a recent work, Badia et al. (2020) used a cost function decomposed into two sub-costs. Instead of mixing the two costs at the cost/reward level and learning Q-functions from this signal, they proposed to learn separate Q-functions, one for each cost, and to merge them afterward: $Q(s, a) = (1 - \beta)\, Q_1(s, a) + \beta\, Q_2(s, a)$. We use this same technique for our multi-objective problems.

Using constraints. In RL, constraints are usually implemented by adding a strong cost whenever the constraint is violated. However, it might be difficult for the Q-function to learn how to model these sudden jumps in the cost signal. Instead, we take inspiration from the work of Badia et al. (2020). We train a separate Q-function for each cost, where the associated reward is -1 whenever the constraint is violated. This Q-function evaluates how many times the constraint is expected to be violated in the future when following an optimal policy. Once this is trained, we can use it to filtrate the set of admissible actions: if $Q_{\text{constraint}}(s, a) > 1$, then we expect action a to lead to at least 1 constraint violation. Once these actions are filtered, the action is selected according to the usual maximization of the traditional Q-function.

NSGA-II. Multi-objective algorithms do not require the computation of an aggregated cost but use all costs. The objective of such algorithms is to obtain the best Pareto front. NSGA-II (Deb et al., 2002) is a state-of-the-art multi-objective algorithm based on EAs (here a genetic algorithm). Starting from a parent population of solutions, parents are selected by a tournament selection involving the rank of the Pareto front they belong to (fitness) and a crowding measure (novelty). Offspring are then obtained by cross-over of the parent solutions and mutations of the resulting parameters. The offspring population is then evaluated and sorted in Pareto fronts. A new parent population is finally obtained by selecting offspring from the higher-ranked Pareto fronts, prioritizing novel solutions as a second metric (low crowding measures).

B.2 Epidemiological Model

The SEIRAH model and the parameter estimations have been extensively described in Prague et al. (2020). In this paper, to account for uncertainty in the model parameters, we use a distribution of models. At each episode, the epidemiological model is sampled from that distribution. The transition function is thus conditioned on a latent code (the model parameters) that is unknown to the agent. This effectively results in a stochastic transition from the viewpoint of the agent. To build the distribution of models, we assume a normal distribution for each of the model parameters, using either the standard deviation from the model inversion for parameters estimated in Prague et al. (2020) or 10% of the mean value for other parameters. Those values are available in appendix Figure 6 features for reproducibility of the experiment. We further add a uniformly distributed delay between the epidemic onset and the start of the learning episode (uniform in [0, 21 days]). This delay models the reaction time of the decision authorities. This results in the distribution of models shown in Figure 7.

	E_0	b	ζ_{1w}	ζ_{2w}	ζ_{3w}	ζ_{4w}	D_q	D_e	D_h	D_i	α	r
E_0	52492	-0,024	-0,144	0,024	0,020	0,008	0,009	0	0	0	0	0
b	-0,024	0,00001	0	0	0	0	0	0	0	0	0	0
ζ_{1w}	-0,144	0	0,004	0	0	0	0	0	0	0	0	0
ζ_{2w}	0,024	0	0	0,003	0	0	0	0	0	0	0	0
ζ_{3w}	0,020	0	0	0	0,003	0	0	0	0	0	0	0
ζ_{4w}	0,008	0	0	0	0	0,0002	0	0	0	0	0	0
D_q	0,009	0	0	0	0	0	0,0001	0	0	0	0	0
D_e	0	0	0	0	0	0	0	0,26	0	0	0	0
D_h	0	0	0	0	0	0	0	0	9	0	0	0
D_i	0	0	0	0	0	0	0	0	0	0,05	0	0
α	0	0	0	0	0	0	0	0	0	0	0,003	0
r	0	0	0	0	0	0	0	0	0	0	0	0,00002

Figure 6: Variance-covariance matrix for parameters of the epidemiological model.

Appendix C. Additional Results

C.1 A Distribution of Models

Figure 7 presents the evolution of the SEIRAH model states for a few models sampled from the distribution of models described in 4.1. Here, no lockdown is enforced.

Now, let us illustrate in detail the results of the case study.

C.2 Pareto Fronts Comparisons

Comparing multi-objective algorithms is not an easy task. The representation of the results itself is not trivial. Figure 4a, for example, presents the standard error of the mean in the GDP cost dimension, but cannot simultaneously represent the error in the other dimension. For this purpose, we can rotate the graph and present the standard error of the mean in the health cost dimension (see Figures 8a and 8b).

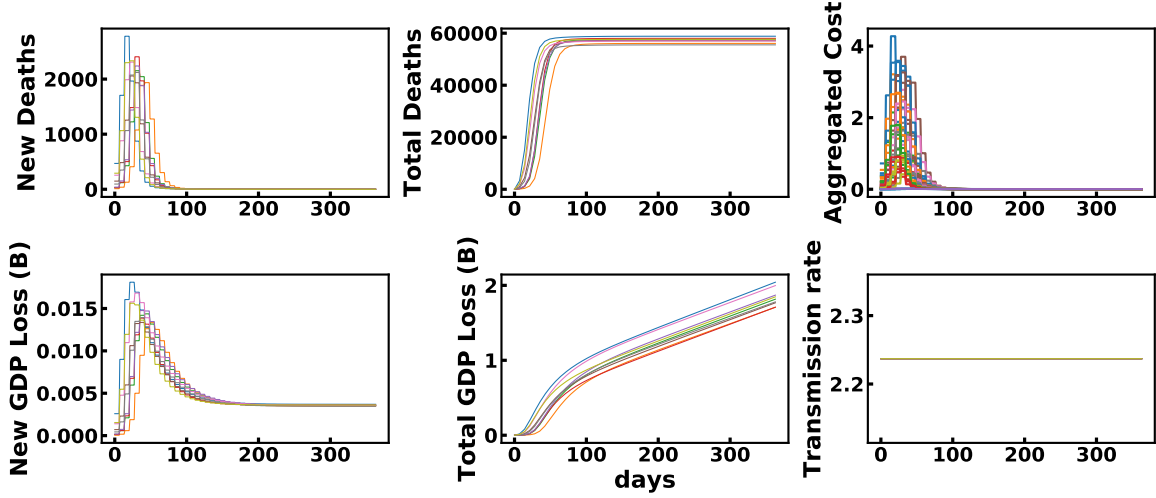Eco cost: 1.85 B, Death Cost: 57649, Aggregated Cost: 45.27

Figure 7: 10 models sampled from the distribution of epidemiological models used in the case-study.

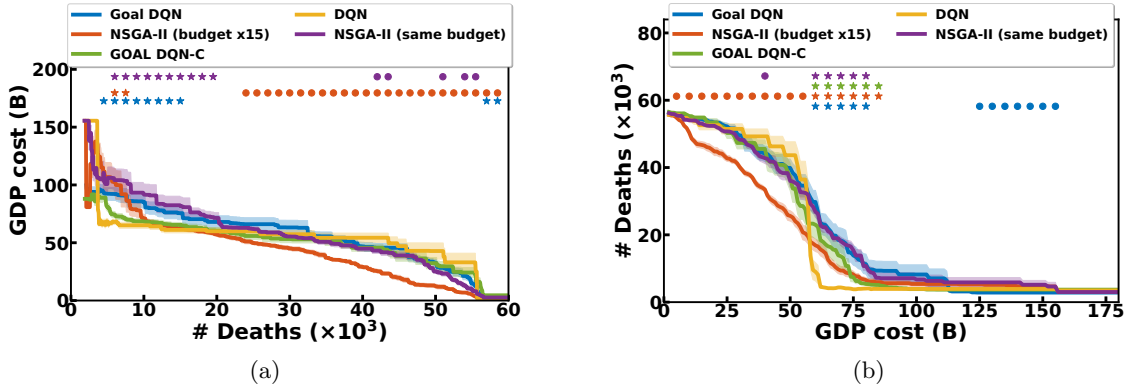|	
(a)	(b)

Figure 8: Pareto fronts for various optimization algorithms. Left: means and standard errors of the mean are computed in the economic cost dimension. right: they are computed in the health cost dimension (over 10 seeds). Stars indicate significantly higher performance w.r.t. the DQN condition while circles indicate significantly lower performance. We use a Welch's t-test with a level of significance $\alpha = 0.05$.

Statistical comparisons. We run each algorithm for 10 seeds to take into account the stochasticity of the learning process and assess the robustness of the algorithms. Rigorous algorithm comparison requires to compute statistical tests to assess the significance of differences in the mean performance (e.g. t-test, see Colas et al., 2019 for a discussion). Here, the two costs require comparisons along two dimensions. One way of handling this is to compare algorithms on one dimension, keeping the other constant. Figures 8a and 8b illustrate this process. In Figure 8a, the circles and stars evaluate the significance of differences (negative and positive w.r.t. the DQN performance respectively) in the economic

cost dimension while the health cost is kept constant. In Figure 8b, we do the reverse and compare health costs, keeping economic costs constant.

Areas under the curve. One way to compare multi-objective algorithms is to compare the area under their normalized Pareto fronts (lower is better). We normalize all Pareto front w.r.t the minimum and maximum values on the two dimensions across all points from all experiments. We can then perform Welch's t-tests to evaluate the significance of the difference between the mean areas metrics of each algorithm.

	DQN	GOAL DQN	GOAL DQN-C	NSGA-II (same)	NSGA-II (x15)
DQN	N/A	0.069	0.55	**0.018**	**0.045**
GOAL DQN		N/A	**0.046**	0.84	**0.0057**
GOAL DQN-C			N/A	0.21	0.35
NSGA-II (same)				N/A	$\mathbf{3.2 \times 10^4}$
NSGA-II (x15)					N/A

Table 2: Comparing areas under the Pareto fronts. The number in row i, column j compares algorithm i and j by providing the p-value of the Welch's t-test assessing the significance of their difference. Bold colored numbers indicate significant differences, green indicates positive differences (column algorithm performs better than row algorithm), while red indicates negative differences (column algorithm performs worse).

Now that we compared algorithms on average metrics, let us delve into each algorithm and look at particular runs to illustrate their respective strategies.

C.3 DQN

We first present strategies found by DQN.
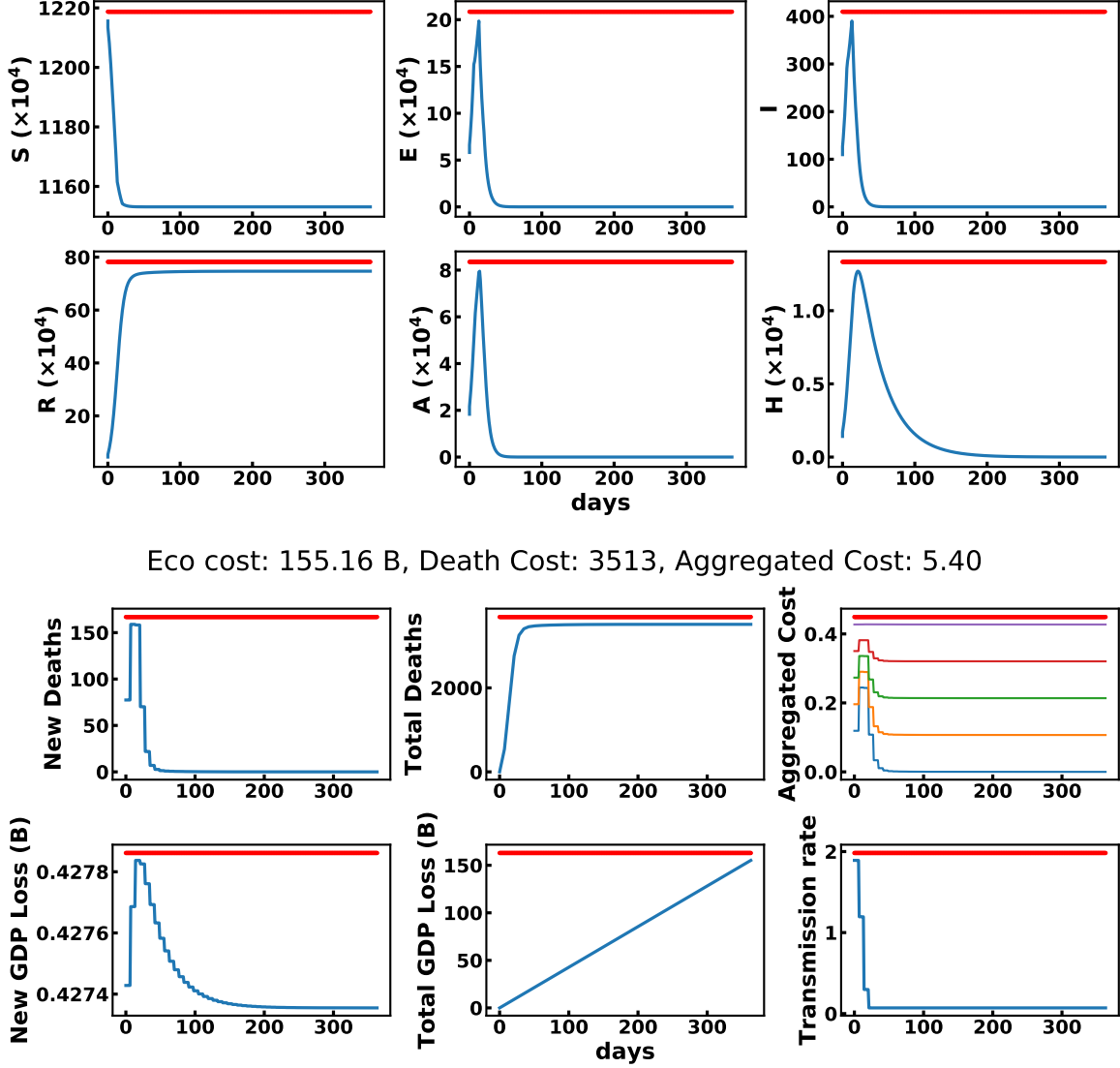
Figure 9: DQN trained with $\beta = 0$.. For one run, this figure shows the evolution of model states (above) and states relevant for optimization (below). The aggregated cost is shown for various values of β in $[0, 0.25, 0.5, 0.75, 1]$. For $\beta = 0$, the agent only cares about the health cost, it always enforces lockdown.
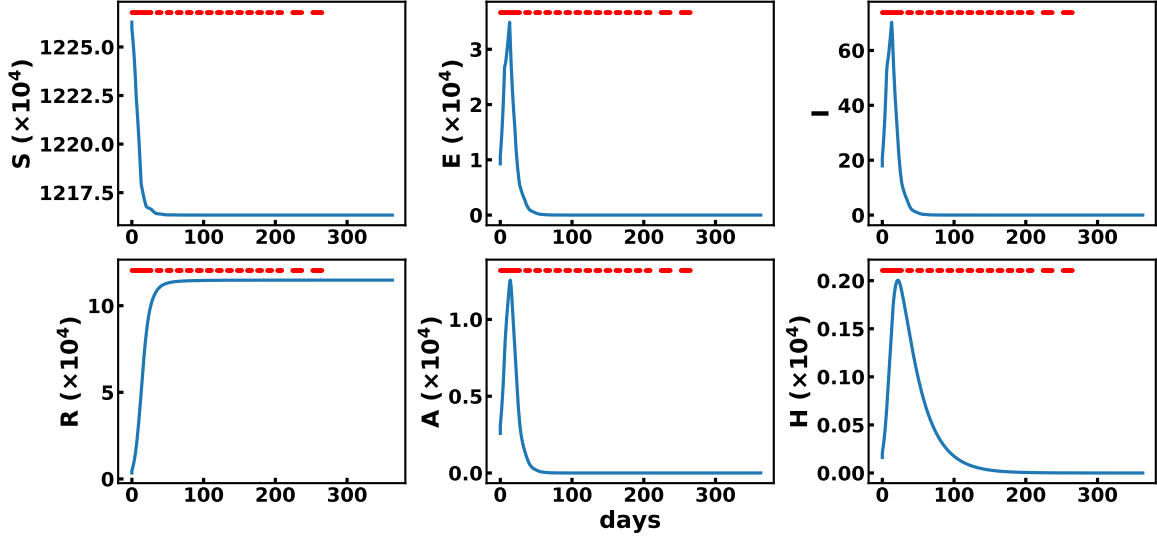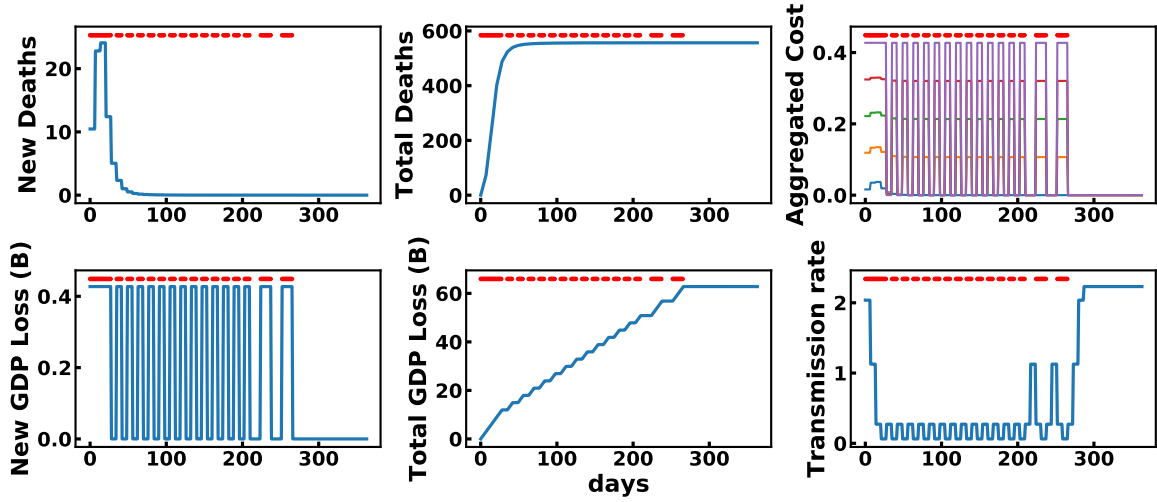
Figure 10: DQN trained with $\beta = 0.5$. Here the strategy is cyclical with a one or two weeks period. For one run, this figure shows the evolution of model states (above) and states relevant for optimization (below). The aggregated cost is shown for various values of β in $[0, 0.25, 0.5, 0.75, 1]$.
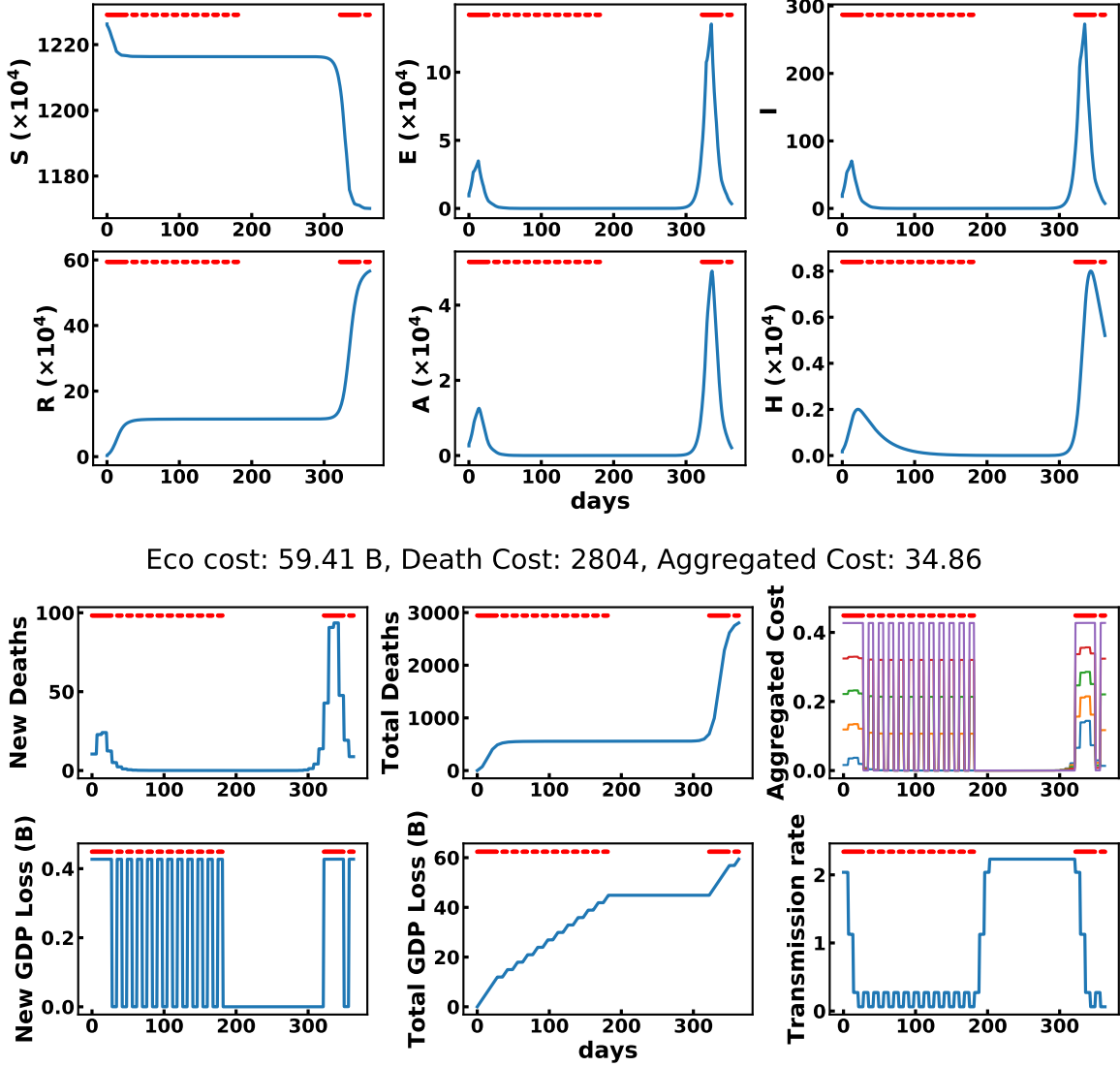
Figure 11: DQN trained with $\beta = 0.55$. Here the agent first lockdown to stop the first wave, but then stops the cyclical lockdown early which induces a second wave later, where the agent also reacts by a lockdown. For one run, this figure shows the evolution of model states (above) and states relevant for optimization (below). The aggregated cost is shown for various values of β in $[0, 0.25, 0.5, 0.75, 1]$.
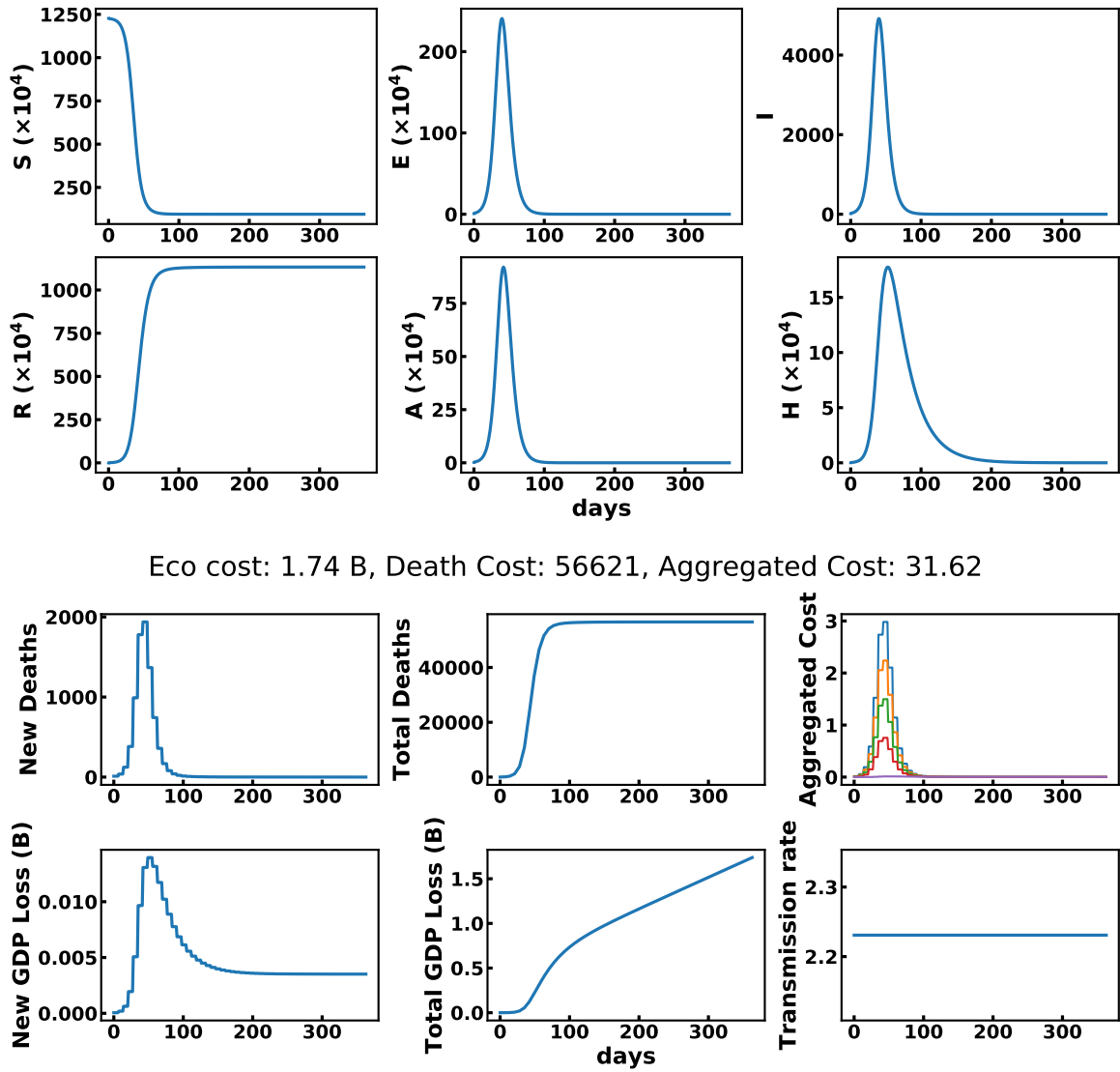
Figure 12: DQN trained with $\beta = 0.65$. Here the agent mostly cares about the economic cost, which results in a no-lockdown policy. For one run, this figure shows the evolution of model states (above) and states relevant for optimization (below). The aggregated cost is shown for various values of β in $[0, 0.25, 0.5, 0.75, 1]$.

C.4 Goal DQN Without Constraints

Now we present a few strategies found by GOAL DQN without constraints.
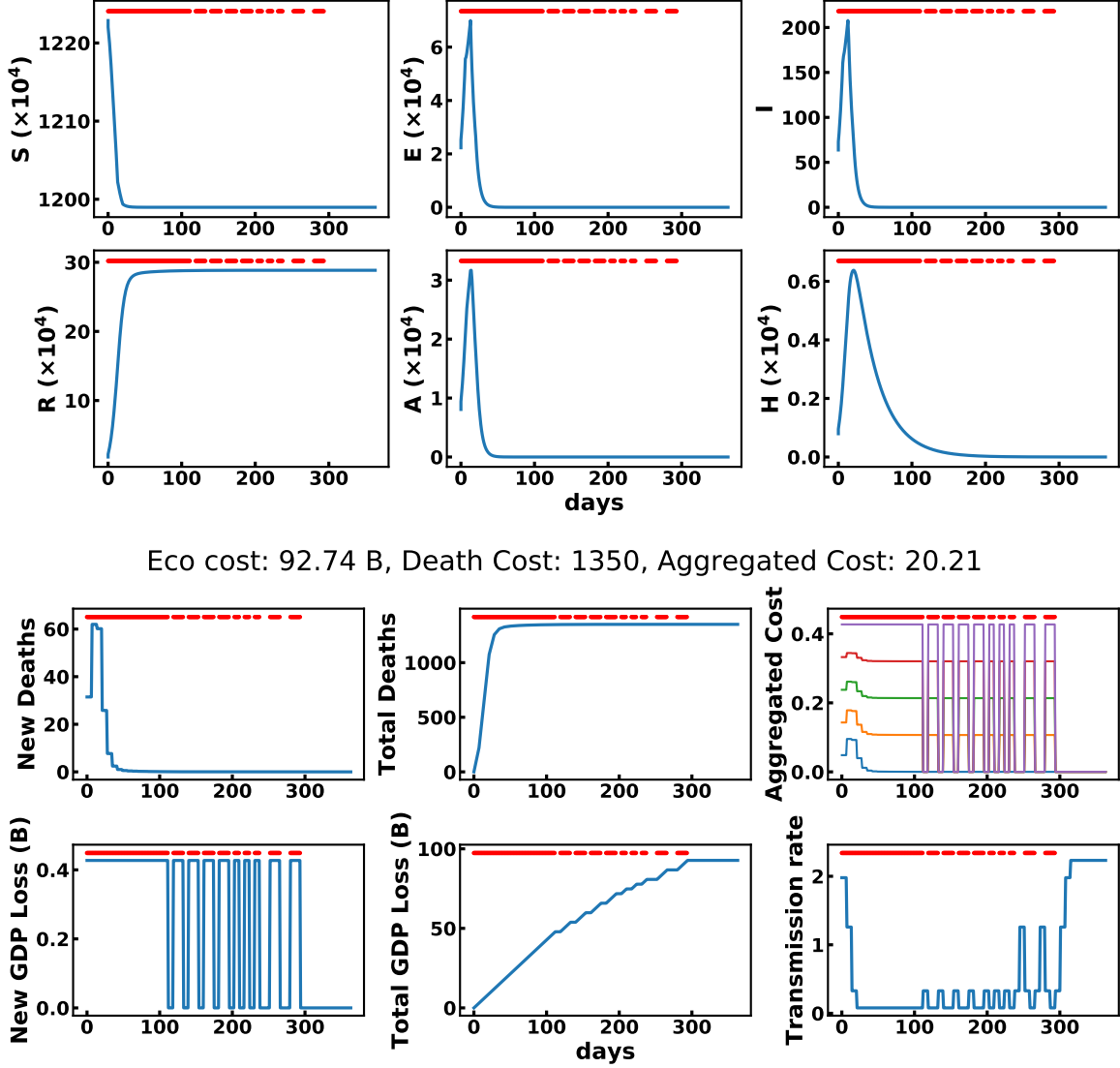
Figure 13: GOAL DQN without constraints evaluated in $\beta = 0.2$. Here the agent starts with a lasting lockdown, then pursues with cyclical lockdowns, which ensures the absence of second wave and, thus, results in low health cost but high economic cost. For one run, this figure shows the evolution of model states (above) and states relevant for optimization (below). The aggregated cost is shown for various values of β in $[0, 0.25, 0.5, 0.75, 1]$.
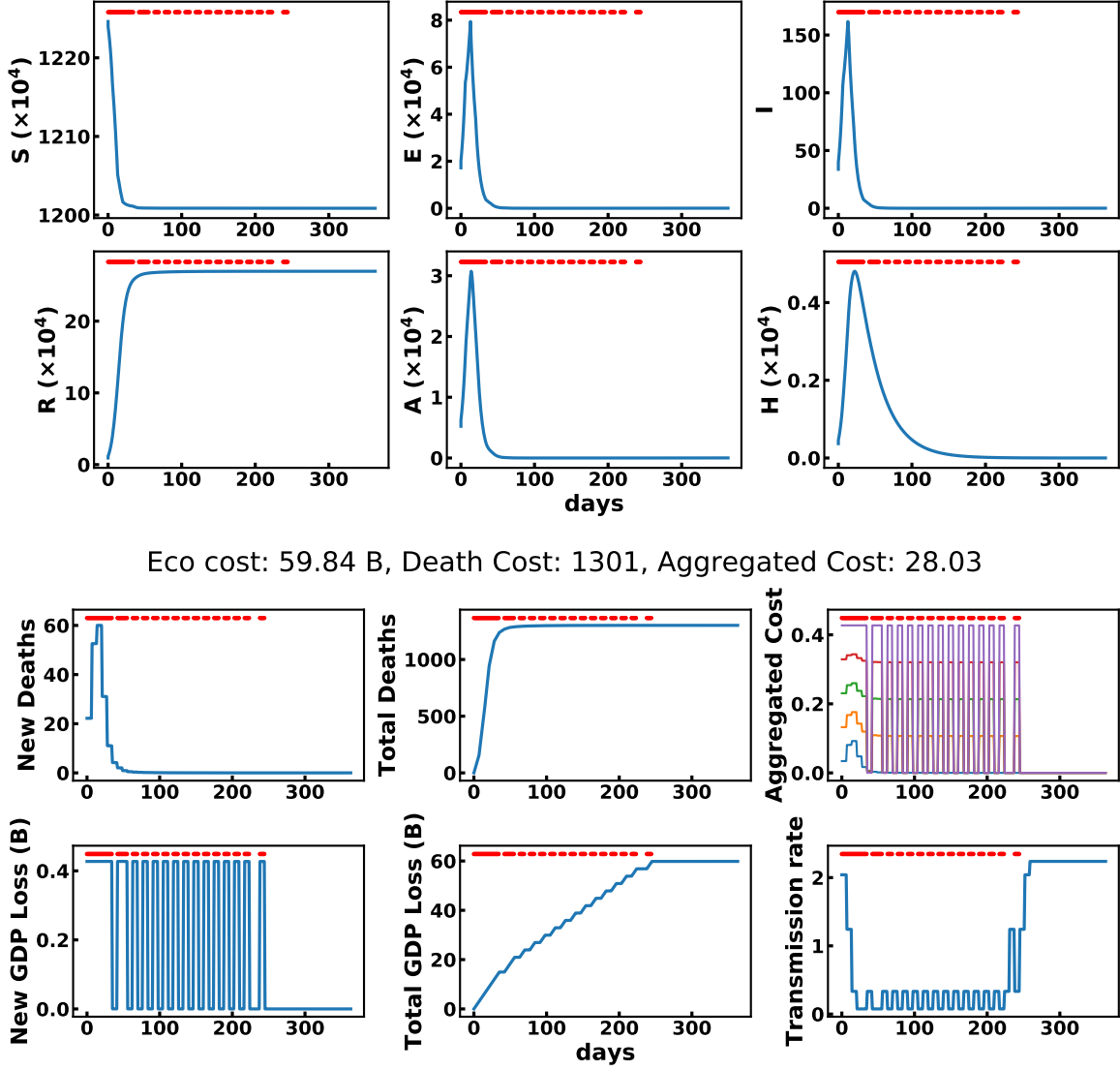
Figure 14: GOAL DQN without constraints evaluated in $\beta = 0.65$. Here we find a cyclical strategy equivalent to the one of shown with a DQN agent trained with $\beta = 0.5$. For one run, this figure shows the evolution of model states (above) and states relevant for optimization (below). The aggregated cost is shown for various values of β in $[0, 0.25, 0.5, 0.75, 1]$.
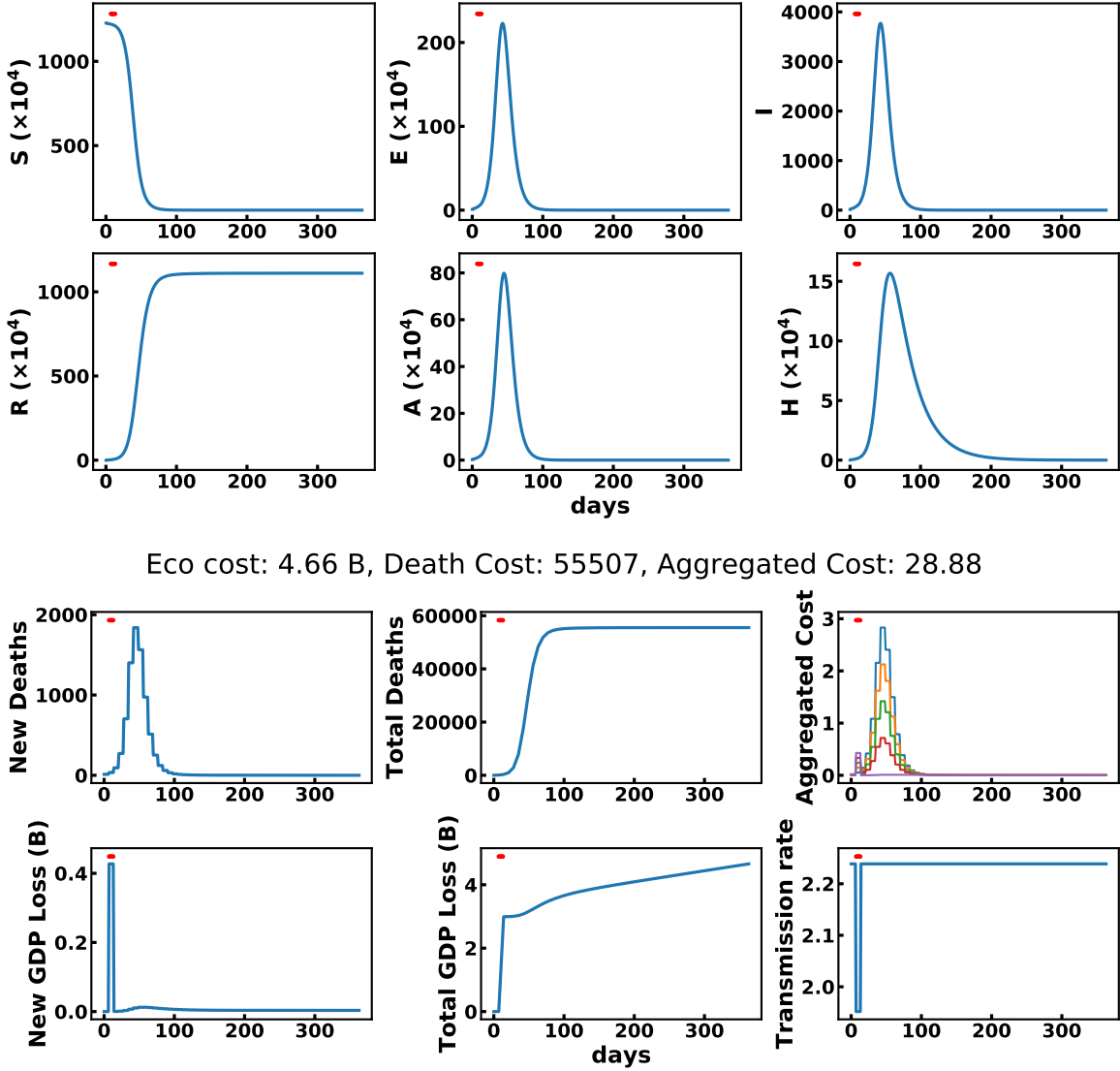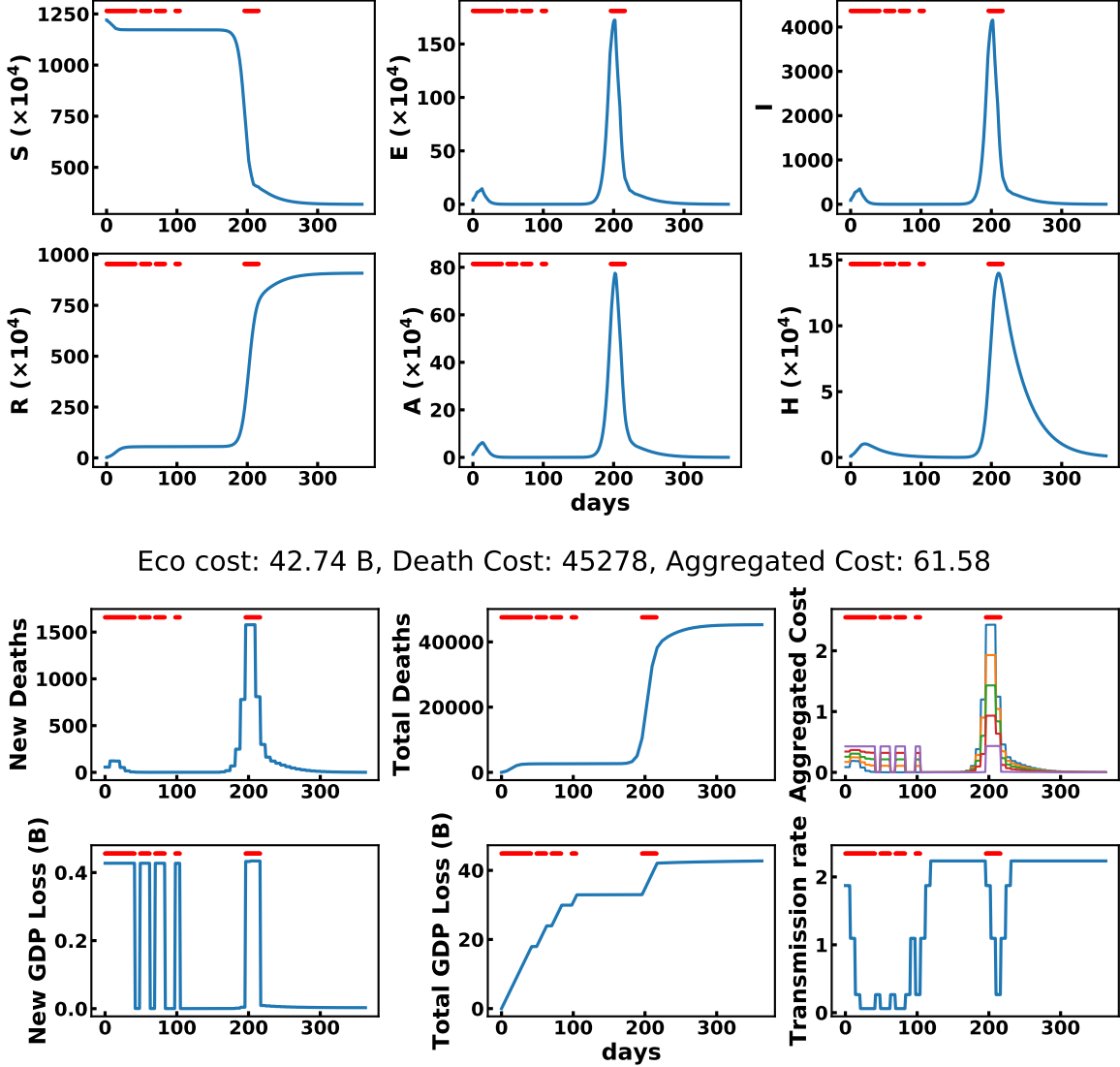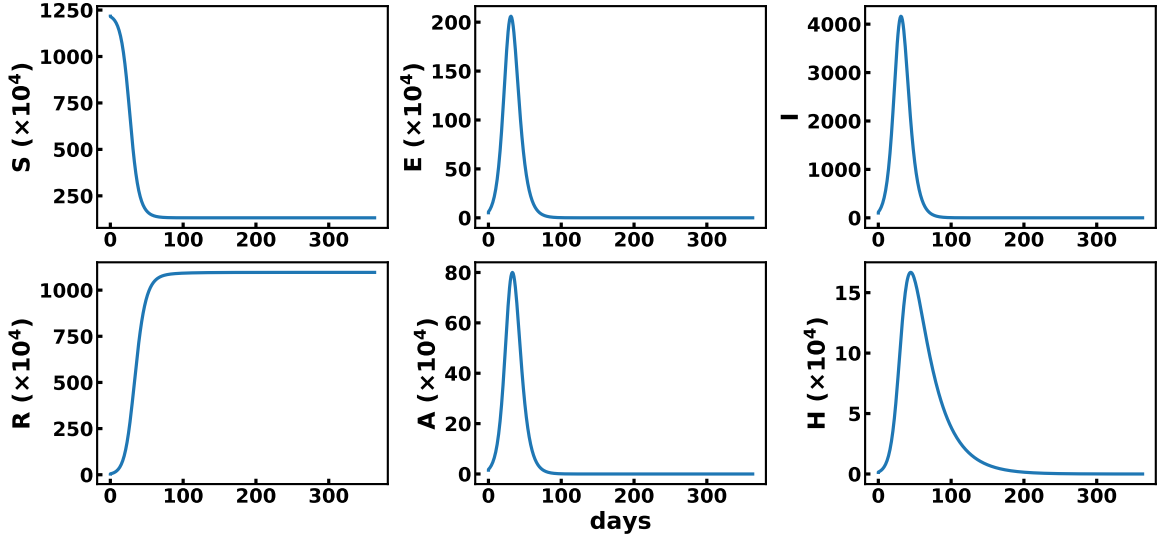
Figure 15: GOAL DQN without constraints evaluated in $\beta = 0.7$. Here the agent only use a lockdown at the very beginning of the epidemic. It is unclear whether this has an impact on the health cost at all. For one run, this figure shows the evolution of model states (above) and states relevant for optimization (below). The aggregated cost is shown for various values of β in $[0, 0.25, 0.5, 0.75, 1]$.

C.5 Goal DQN With Constraints

Now we present a few strategies found by GOAL DQN with constraints and study how it reacts to health and economic constraints.
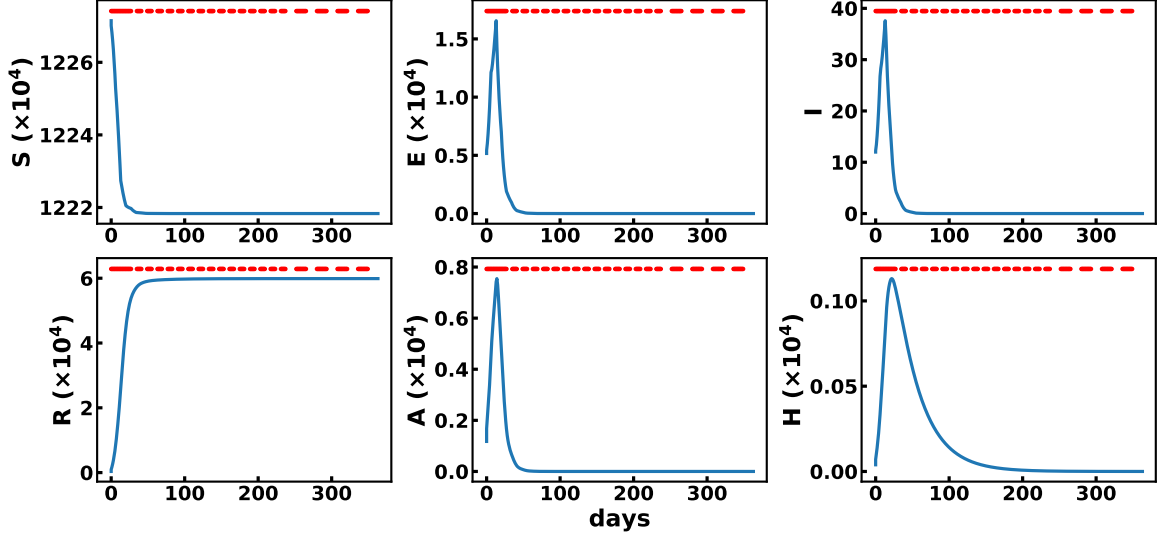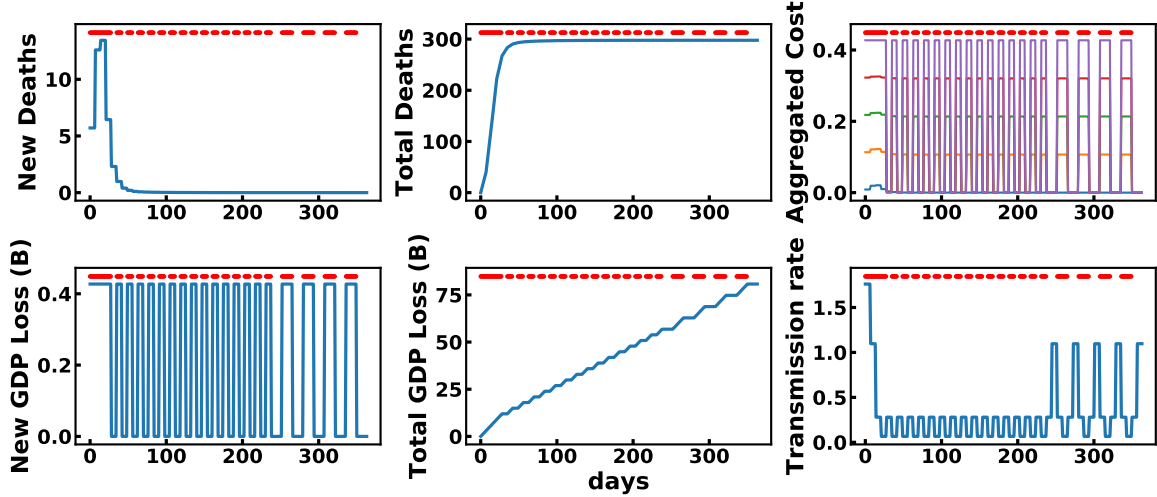
Figure 16: GOAL DQN with constraints evaluated in $\beta = 0.3$, $M_{\text{economic}} = 160B$, $M_{\text{health}} = 62000$ deaths. This boils down to no constraints as they are maximal values. This leads to a cyclical policy. For one run, this figure shows the evolution of model states (above) and states relevant for optimization (below). The aggregated cost is shown for various values of β in $[0, 0.25, 0.5, 0.75, 1]$.

Eco cost: 42.74 B, Death Cost: 45278, Aggregated Cost: 61.58

Figure 17: GOAL DQN with constraints evaluated in $\beta = 0.3, M_{\text{economic}} = 55B, M_{\text{health}} = 62000$ deaths. This is the same β as Figure 16 (previous page). Now there is no constraint on the number of deaths but a strong constraint on the economic cost. The strategy is not cyclical anymore, as the resulting economic cost would be too high. This strategy stays below the economic constraint but still tries to minimize the health cost. For one run, this figure shows the evolution of model states (above) and states relevant for optimization (below). The aggregated cost is shown for various values of β in $[0, 0.25, 0.5, 0.75, 1]$.
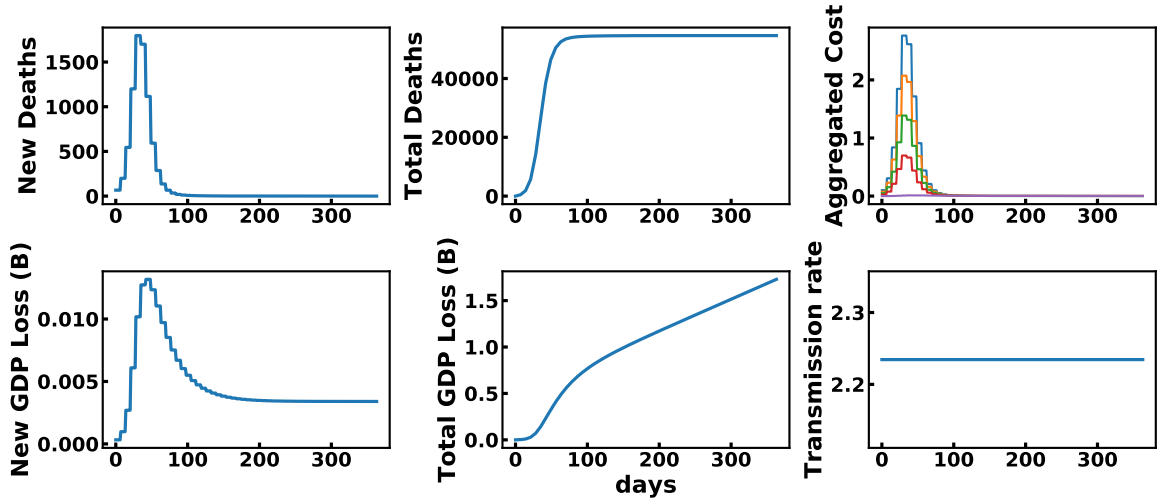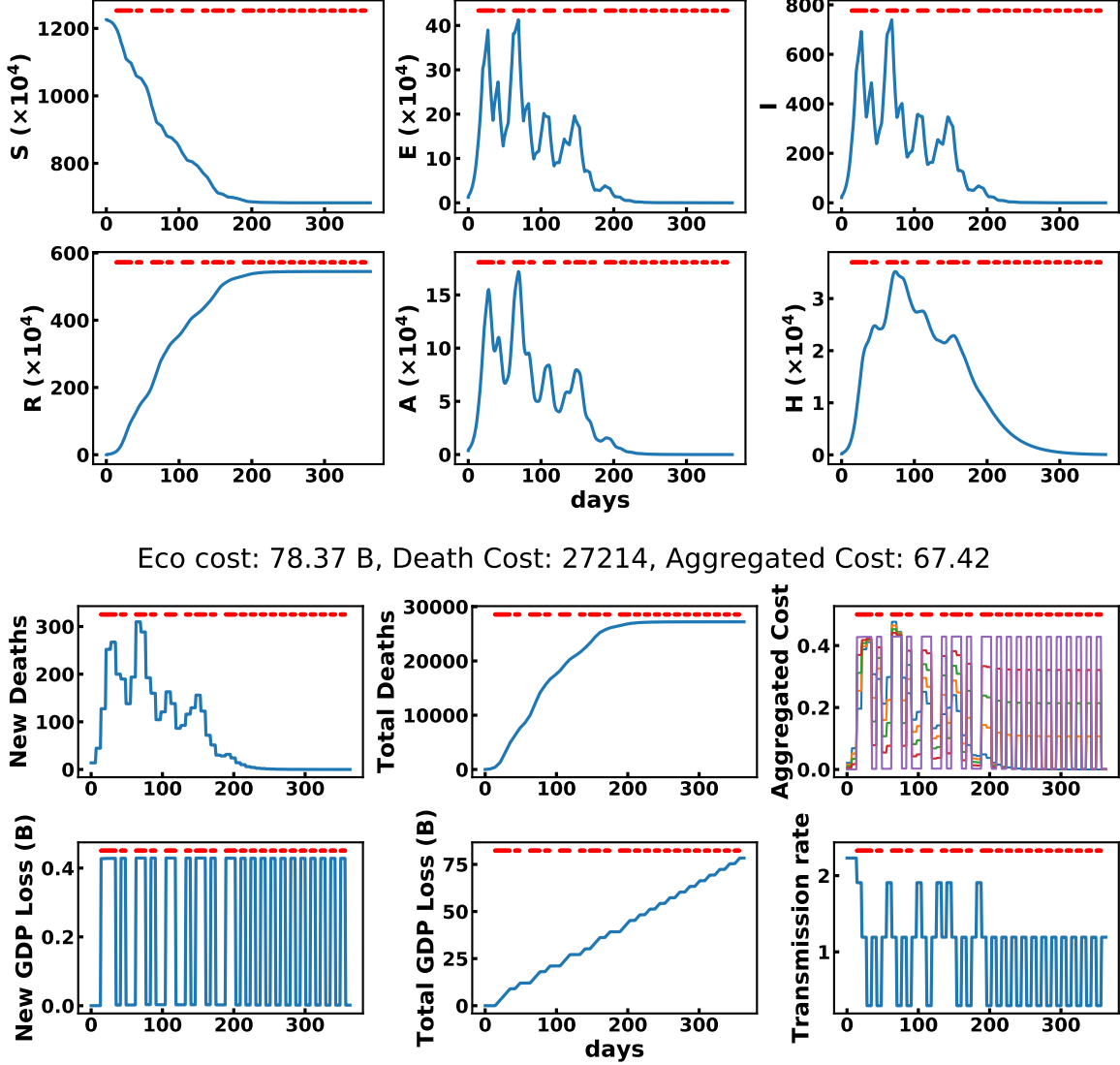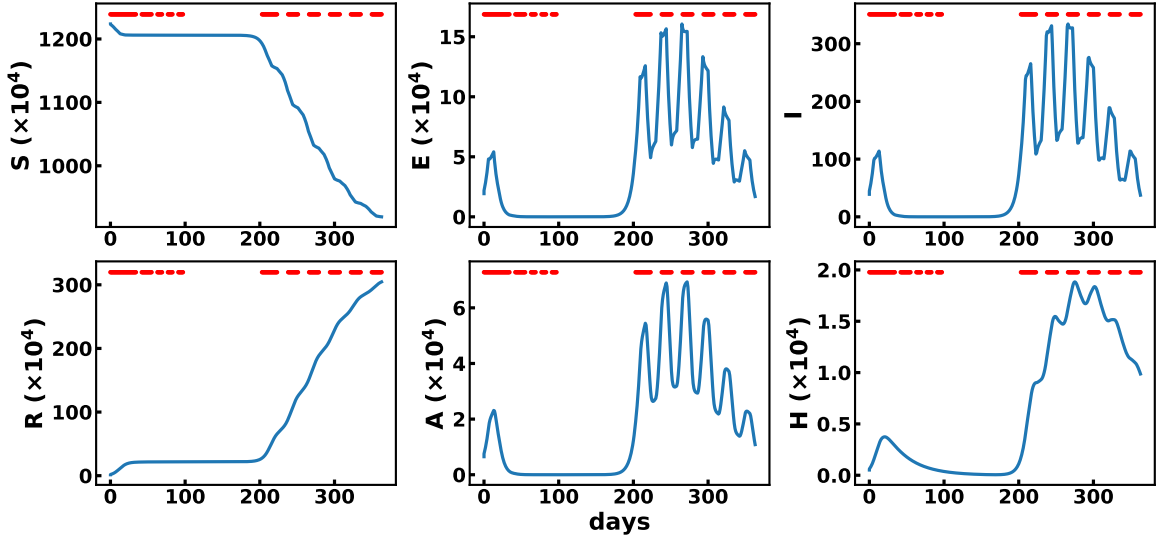
Figure 18: GOAL DQN with constraints evaluated in $\beta = 0.7, M_{\text{economic}} = 160B, M_{\text{health}} = 62000$ deaths. There is no constraint. As the balance favors the economic cost, the strategy does not implement any lockdown. For one run, this figure shows the evolution of model states (above) and states relevant for optimization (below). The aggregated cost is shown for various values of β in $[0, 0.25, 0.5, 0.75, 1]$.
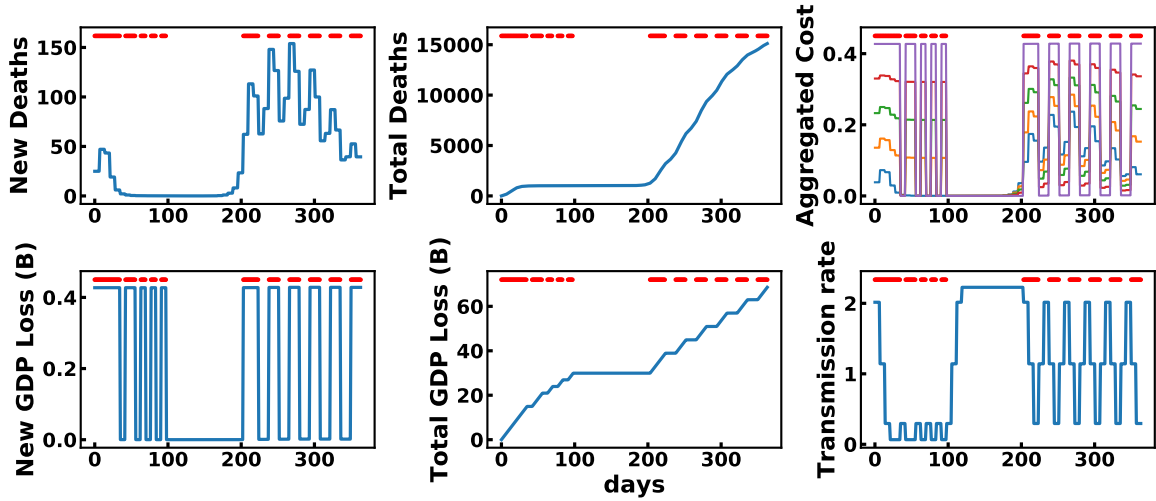
Figure 19: GOAL DQN with constraints evaluated in $\beta = 0.7, M_{\text{economic}} = 160B, M_{\text{health}} = 30500$ deaths. Now we have the same setup as the previous page (Figure 18), except that we have a strong constraint on the number of deaths. The resulting strategy respects the constraint while attempting to minimize the economic cost. For one run, this figure shows the evolution of model states (above) and states relevant for optimization (below). The aggregated cost is shown for various values of β in $[0, 0.25, 0.5, 0.75, 1]$.

Figure 20: GOAL DQN with constraints evaluated in $\beta = 0.3, M_{\text{economic}} = 55B, M_{\text{health}} = 15000$ deaths. Here we have strong constraints on both economic costs. In that case, there is no good solution. This strategy respects the health constraint but violates the economic constraint. For one run, this figure shows the evolution of model states (above) and states relevant for optimization (below). The aggregated cost is shown for various values of β in $[0, 0.25, 0.5, 0.75, 1]$.

C.6 NSGA-II

Now we present a few strategies found by NSGA-II.
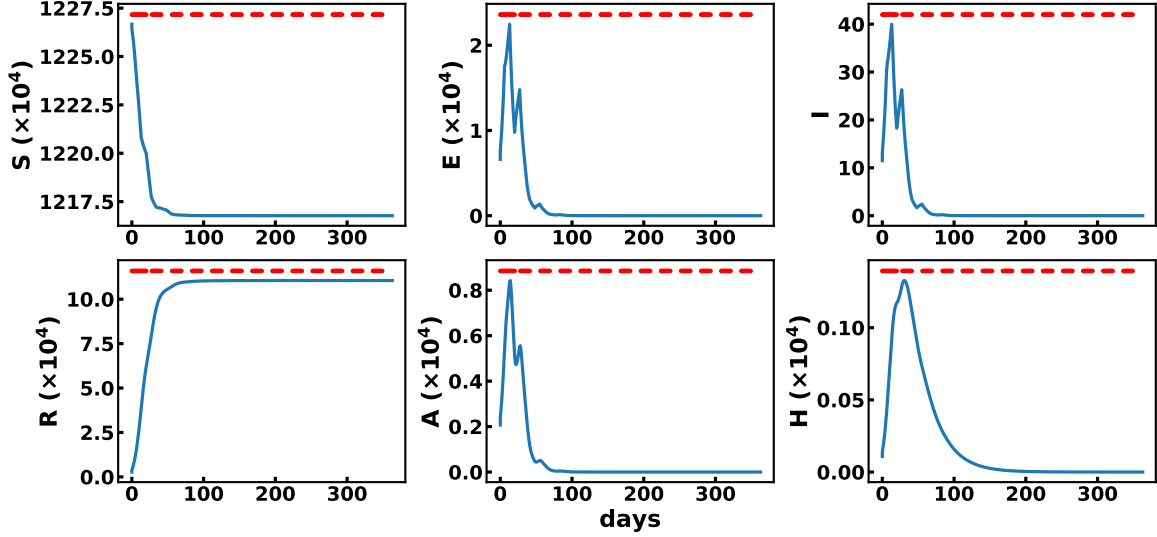
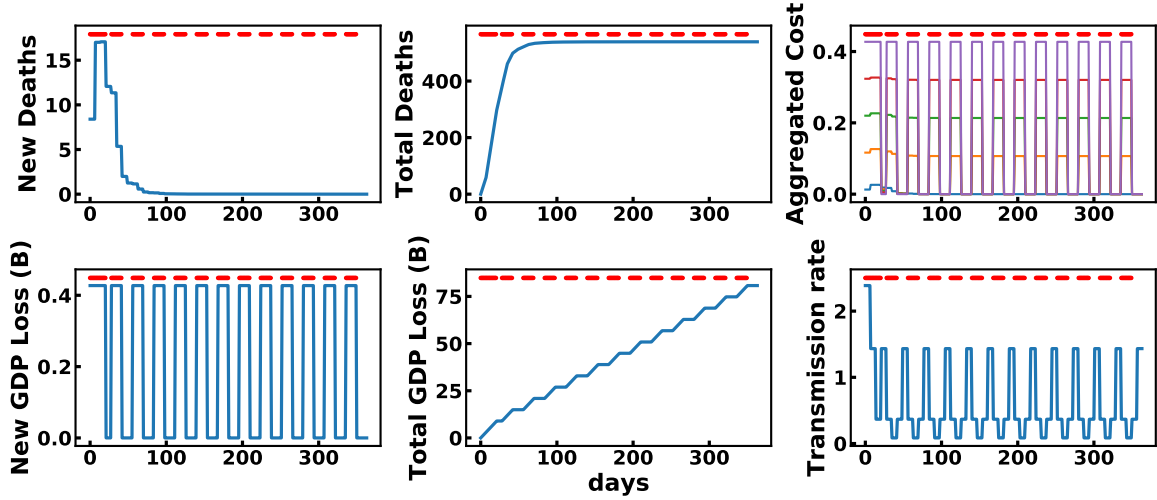Eco cost: 80.76 B, Death Cost: 538, Aggregated Cost: 40.79

Figure 21: NSGA-II, the closest point to [8000 deaths, 60B] in the Pareto front. In this low health cost regime, NSGA-II finds a cyclical strategy with a period of 2 weeks. For one run, this figure shows the evolution of model states (above) and states relevant for optimization (below). The aggregated cost is shown for various values of β in $[0, 0.25, 0.5, 0.75, 1]$.
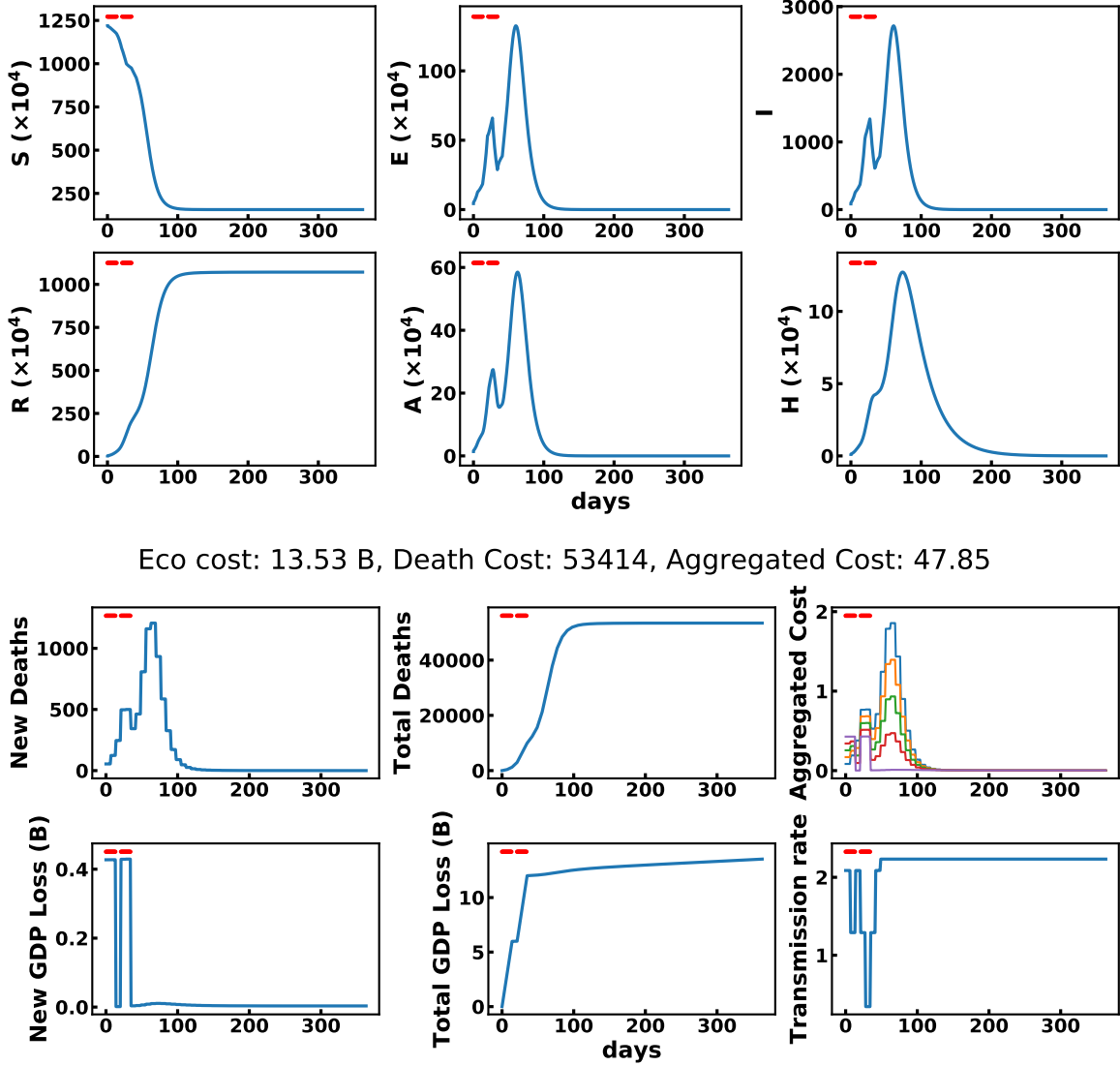
Figure 22: NSGA-II, the closest point to [30000 deaths, 40B] in the Pareto front. Here we find that NSGA-II uses alternative strategies depending on the epidemiological models it faces. The average of these strategies ends up close to 30000, although the two strategies either find high health costs (>50000) or low ones (<1000). These plots show the first alternative, where the strategy aims at breaking the first wave of the epidemic. See the other alternative in Figure 23. For one run, this figure shows the evolution of model states (above) and states relevant for optimization (below). The aggregated cost is shown for various values of β in $[0, 0.25, 0.5, 0.75, 1]$.
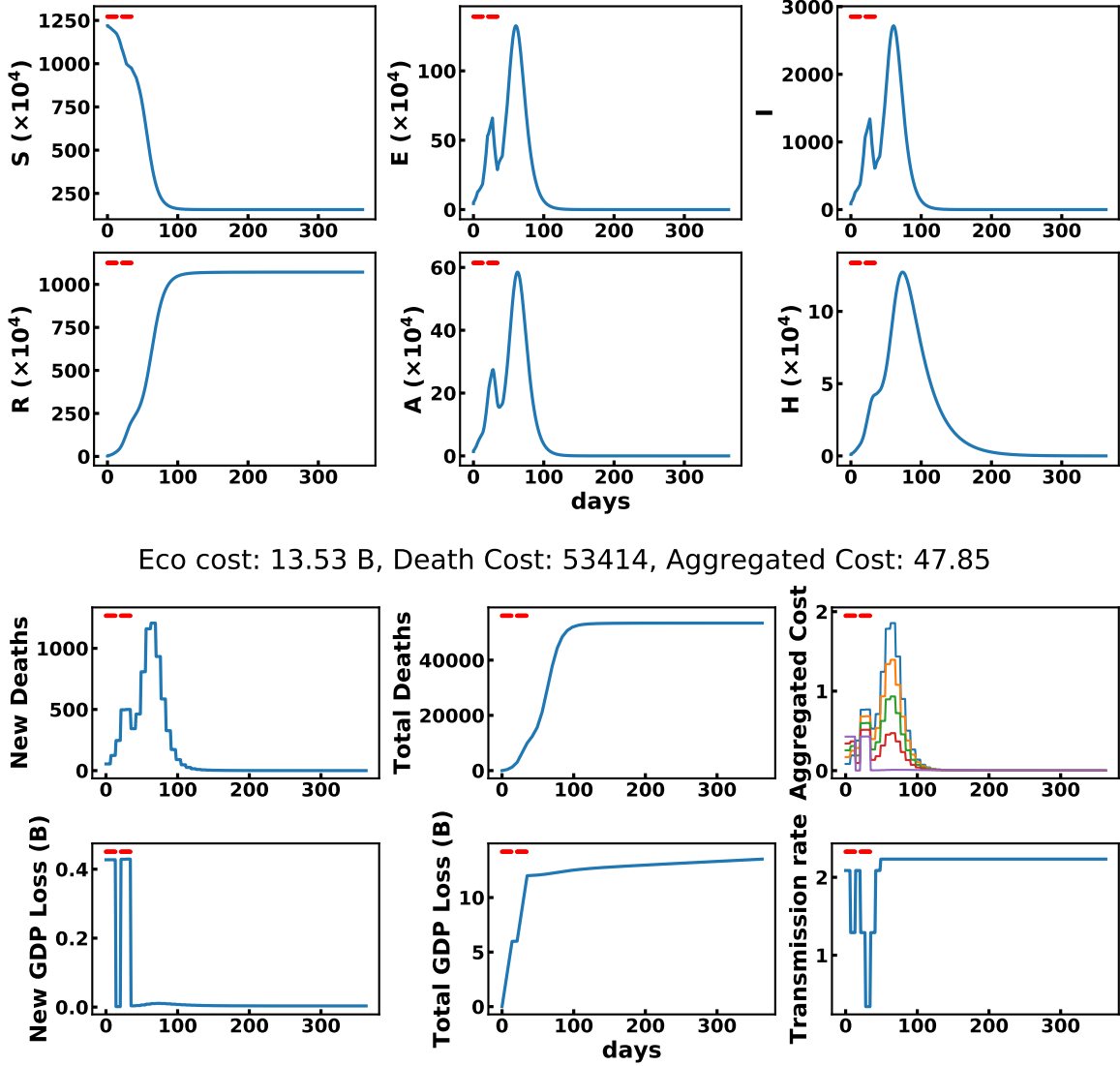
Figure 23: NSGA-II, the closest point to [30000 deaths, 40B] in the Pareto front. Here we find that NSGA-II uses alternative strategies depending on the epidemiological models it faces. The average of these strategies ends up close to 30000, although the two strategies either find high health costs (>50000) or low ones (<1000). These plots show the second alternative, where the strategy is cyclical and achieved low health costs. See Figure 22 for the first alternative. For one run, this figure shows the evolution of model states (above) and states relevant for optimization (below). The aggregated cost is shown for various values of β in $[0, 0.25, 0.5, 0.75, 1]$.

Figure 24: NSGA-II, the closest point to [45000 deaths, 20B] in the Pareto front. NSGA-II seem to find a robust strategy that consists in a single lockdown of a few weeks to break the first wave. For one run, this figure shows the evolution of model states (above) and states relevant for optimization (below). The aggregated cost is shown for various values of β in $[0, 0.25, 0.5, 0.75, 1]$.
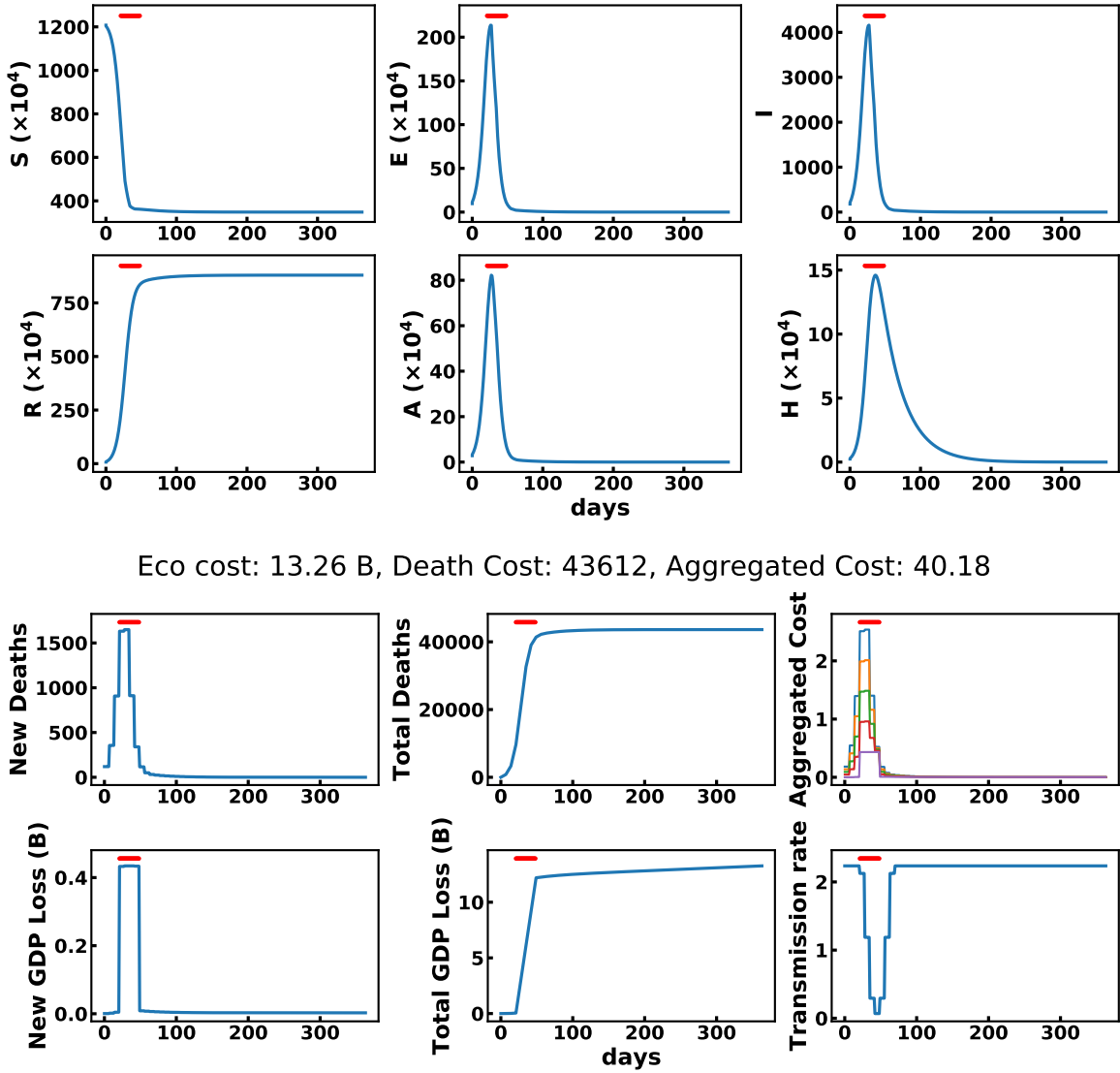
References

Alamo, T., Reina, D. G., & Millán, P. (2020). Data-driven methods to monitor, model, forecast and control covid-19 pandemic: Leveraging data science, epidemiology and control theory. *arXiv preprint arXiv:2006.01731*.

Arango, M., & Pelov, L. (2020). Covid-19 pandemic cyclic lockdown optimization using reinforcement learning. *arXiv preprint arXiv:2009.04647*.

Badia, A. P., Piot, B., Kapturowski, S., Sprechmann, P., Vitvitskyi, A., Guo, D., & Blundell, C. (2020). Agent57: Outperforming the atari human benchmark. *arXiv preprint arXiv:2003.13350*.

Bi, Q., Goodman, K. E., Kaminsky, J., & Lessler, J. (2019). What is machine learning? a primer for the epidemiologist. *American journal of epidemiology*, *188*(12), 2222–2239.

Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., & Zaremba, W. (2016). Openai gym. *arXiv preprint arXiv:1606.01540*.

Cauchemez, S., Hoze, N., Cousien, A., Nikolay, B., et al. (2019). How modelling can enhance the analysis of imperfect epidemic data. *Trends in parasitology*, *35*(5), 369–379.

Chandak, A., Dey, D., Mukhoty, B., & Kar, P. (2020). Epidemiologically and socio-economically optimal policies via bayesian optimization. *arXiv preprint arXiv:2005.11257*.

Charpentier, A., Elie, R., Laurière, M., & Tran, V. C. (2020). Covid-19 pandemic control: balancing detection policy and lockdown intervention under icu sustainability..

Chen, R. T., Rubanova, Y., Bettencourt, J., & Duvenaud, D. K. (2018). Neural ordinary differential equations. In *Advances in neural information processing systems*, pp. 6571–6583.

Colas, C., Sigaud, O., & Oudeyer, P.-Y. (2019). A hitchhiker's guide to statistical comparisons of reinforcement learning algorithms. *arXiv preprint arXiv:1904.06979*.

Deb, K., Karthik, S., et al. (2007). Dynamic multi-objective optimization and decision-making using modified nsga-ii: a case study on hydro-thermal power scheduling. In *International conference on evolutionary multi-criterion optimization*, pp. 803–817. Springer.

Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE transactions on evolutionary computation*, *6*(2), 182–197.

Douglas, P. H. (1976). The cobb-douglas production function once again: Its history, its testing, and some new empirical values. *Journal of Political Economy*, *84*(5), 903–915.

Elie, R., Hubert, E., & Turinici, G. (2020). Contact rate epidemic control of covid-19: an equilibrium view. *Mathematical Modelling of Natural Phenomena*, *15*, 35.

Ferguson, N., Laydon, D., Nedjati Gilani, G., Imai, N., Ainslie, K., Baguelin, M., Bhatia, S., Boonyasiri, A., Cucunuba Perez, Z., Cuomo-Dannenburg, G., et al. (2020). Report 9: Impact of non-pharmaceutical interventions (npis) to reduce covid19 mortality and healthcare demand..

Ferguson, N. M., Cummings, D. A., Cauchemez, S., Fraser, C., Riley, S., Meeyai, A., Iamsirithaworn, S., & Burke, D. S. (2005). Strategies for containing an emerging influenza pandemic in southeast asia. *Nature*, *437*(7056), 209–214.

Ferguson, N. M., Cummings, D. A., Fraser, C., Cajka, J. C., Cooley, P. C., & Burke, D. S. (2006). Strategies for mitigating an influenza pandemic. *Nature*, *442*(7101), 448–452.

Flaxman, A. D., & Vos, T. (2018). Machine learning in population health: opportunities and threats. *PLoS medicine*, *15*(11), e1002702.

Flaxman, S., Mishra, S., Gandy, A., Unwin, H. J. T., Mellan, T. A., Coupland, H., Whittaker, C., Zhu, H., Berah, T., Eaton, J. W., et al. (2020). Estimating the effects of non-pharmaceutical interventions on covid-19 in europe. *Nature*, *584*(7820), 257–261.

Haarnoja, T., Zhou, A., Abbeel, P., & Levine, S. (2018). Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *Proceedings of ICML*.

Hafner, D., Lillicrap, T., Ba, J., & Norouzi, M. (2019). Dream to control: Learning behaviors by latent imagination. In *Proceedings of ICLR*.

Halloran, M. E., Ferguson, N. M., Eubank, S., Longini, I. M., Cummings, D. A., Lewis, B., Xu, S., Fraser, C., Vullikanti, A., Germann, T. C., et al. (2008). Modeling targeted layered containment of an influenza pandemic in the united states. *Proceedings of the National Academy of Sciences*, *105*(12), 4639–4644.

Havik, K., Mc Morrow, K., Orlandi, F., Planas, C., Raciborski, R., Röger, W., Rossi, A., Thum-Thysen, A., Vandermeulen, V., et al. (2014). The production function methodology for calculating potential growth rates & output gaps. Tech. rep., Directorate General Economic and Financial Affairs (DG ECFIN), European . . . .

He, Z., & Yen, G. G. (2015). Visualization and performance metric in many-objective optimization. *IEEE Transactions on Evolutionary Computation*, *20*(3), 386–402.

Insee (2020). Insee website. `https://www.insee.fr/`.

Ishibuchi, H., Tsukamoto, N., & Nojima, Y. (2008). Evolutionary many-objective optimization: A short review. In *2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence)*, pp. 2419–2426. IEEE.

Kalashnikov, D., Irpan, A., Pastor, P., Ibarz, J., Herzog, A., Jang, E., Quillen, D., Holly, E., Kalakrishnan, M., Vanhoucke, V., et al. (2018). Scalable deep reinforcement learning for vision-based robotic manipulation. In *Conference on Robot Learning*, pp. 651–673.

Kompella, V., Capobianco, R., Jong, S., Browne, J., Fox, S., Meyers, L., Wurman, P., & Stone, P. (2020). Reinforcement learning for optimization of covid-19 mitigation policies..

Kraemer, M. U. G., Yang, C.-H., Gutierrez, B., Wu, C.-H., Klein, B., Pigott, D. M., du Plessis, L., Faria, N. R., Li, R., Hanage, W. P., Brownstein, J. S., Layan, M., Vespignani, A., Tian, H., Dye, C., Pybus, O. G., & Scarpino, S. V. (2020). The effect of human mobility and control measures on the COVID-19 epidemic in China. *Science*, *368*(6490), 493–497.

Lemecha Obsu, L., & Feyissa Balcha, S. (2020). Optimal control strategies for the transmission risk of covid-19. *Journal of biological dynamics*, *14*(1), 590–607.

Libin, P., Moonens, A., Verstraeten, T., Perez-Sanjines, F., Hens, N., Lemey, P., & Nowé, A. (2020). Deep reinforcement learning for large-scale epidemic control. *arXiv preprint arXiv:2003.13676*.

Mandel, A., & Veetil, V. P. (2020). The economic cost of covid lockdowns: An out-of-equilibrium analysis. *Economics of Disasters and Climate Change*.

Miikkulainen, R., Francon, O., Meyerson, E., Qiu, X., Canzani, E., & Hodjat, B. (2020). From prediction to prescription: Ai-based optimization of non-pharmaceutical interventions for the covid-19 pandemic. *arXiv preprint arXiv:2005.13766*.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., et al. (2015). Human-level control through deep reinforcement learning. *nature*, *518*(7540), 529–533.

Ohi, A. Q., Mridha, M., Monowar, M. M., & Hamid, M. A. (2020). Exploring optimal control of epidemic spread using reinforcement learning. *Scientific Reports*, *10*.

Orlowski, E. J., & Goldsmith, D. J. (2020). Four months into the covid-19 pandemic, sweden's prized herd immunity is nowhere in sight. *Journal of the Royal Society of Medicine*, *113*(8), 292–298.

Pardo, F., Tavakoli, A., Levdik, V., & Kormushev, P. (2018). Time limits in reinforcement learning. In *International Conference on Machine Learning*, pp. 4045–4054.

Pasin, C., Dufour, F., Villain, L., Zhang, H., & Thiébaut, R. (2018). Controlling il-7 injections in hiv-infected patients. *Bulletin of mathematical biology*, *80*(9), 2349–2377.

Perkins, A., & Espana, G. (2020). Optimal control of the covid-19 pandemic with non-pharmaceutical interventions. *medRxiv*.

Prague, M., Wittkop, L., Clairon, Q., Dutartre, D., Thiébaut, R., & Hejblum, B. P. (2020). Population modeling of early covid-19 epidemic dynamics in french regions and estimation of the lockdown impact on infection rate. *medRxiv*.

Probert, W. J. M., Lakkur, S., Fonnesbeck, C. J., Shea, K., Runge, M. C., Tildesley, M. J., & Ferrari, M. J. (2019). Context matters: using reinforcement learning to develop human-readable, state-dependent outbreak response policies. *Philosophical Transactions of the Royal Society B: Biological Sciences*, *374*(1776), 20180277.

Schaul, T., Horgan, D., Gregor, K., & Silver, D. (2015). Universal value function approximators. In *International conference on machine learning*, pp. 1312–1320.

Shearer, F. M., Moss, R., McVernon, J., Ross, J. V., & McCaw, J. M. (2020). Infectious disease pandemic planning and response: Incorporating decision analysis. *PLoS medicine*, *17*(1), e1003018.

Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., Bolton, A., et al. (2017). Mastering the game of go without human knowledge. *nature*, *550*(7676), 354–359.

Sutton, R. S., & Barto, A. G. (2018). *Reinforcement learning: An introduction*. MIT press.

Tarrataca, L., Dias, C., Haddad, D. B., & Arruda, E. (2020). Flattening the curves: on-off lock-down strategies for covid-19 with an application to brazil. *arXiv preprint arXiv:2004.06916*.

Watkins, C. J., & Dayan, P. (1992). Q-learning. *Machine learning*, *8*(3-4), 279–292.

World Health Organization (2020). Who director-general's opening remarks at the media briefing on COVID-19 – 11 march 2020.. Accessed: 2020-04-02.

Yaesoubi, R., & Cohen, T. (2016). Identifying cost-effective dynamic policies to control epidemics. *Statistics in medicine, 35*(28), 5189–5209.

Yaesoubi, R., Havumaki, J., Chitwood, M. H., Menzies, N. A., Gonsalves, G., Salomon, J., Paltiel, A. D., & Cohen, T. (2020). Adaptive policies for use of physical distancing interventions during the covid-19 pandemic. *medRxiv.*

Yáñez, A., Hayes, C., & Glavin, F. (2019). Towards the control of epidemic spread: Designing reinforcement learning environments.. In *AICS*, pp. 188–199.