Exploring the Pareto front of multi-objective COVID-19 mitigation policies using reinforcement learning

A Preprint

Mathieu Reymond
Vrije Universiteit Brussel
Brussels, Belgium
`mathieu.reymond@vub.be`

Conor F. Hayes
National University of Ireland Galway
Galway, Ireland

Lander Willem
University of Antwerp
Antwerp, Belgium

Roxana Rădulescu
Vrije Universiteit Brussel
Brussels, Belgium

Steven Abrams
Hasselt University
Hasselt, Belgium

Diederik M. Roijers
HU University of Applied Sciences Utrecht
Utrecht, the Netherlands

Enda Howley
National University of Ireland Galway
Galway, Ireland

Patrick Mannion
National University of Ireland Galway
Galway, Ireland

Niel Hens
Hasselt University
Hasselt, Belgium

Ann Nowé
Vrije Universiteit Brussel
Brussels, Belgium

Pieter Libin
Vrije Universiteit Brussel
Brussels, Belgium

Abstract

Infectious disease outbreaks can have a disruptive impact on public health and societal processes. As decision making in the context of epidemic mitigation is hard, reinforcement learning provides a methodology to automatically learn prevention strategies in combination with complex epidemic models. Current research focuses on optimizing policies with respect to a single objective, such as the pathogen's attack rate. However, as the mitigation of epidemics involves distinct, and possibly conflicting, criteria (i.a., prevalence, mortality, morbidity, cost), a multi-objective decision approach is warranted to learn balanced policies. To lift this decision-making process to real-world epidemic models, we apply deep multi-objective reinforcement learning and build upon a state-of-the-art algorithm, Pareto Conditioned Networks (PCN), to learn a set of solutions that approximates the Pareto front of the decision problem. We consider the first wave of the Belgian COVID-19 epidemic, which was mitigated by a lockdown, and study different deconfinement strategies, aiming to minimize both COVID-19 cases (i.e., infections and hospitalizations) and the societal burden that is induced by the applied mitigation measures. We contribute a multi-objective Markov decision process that encapsulates the stochastic compartment model that was used to inform policy makers during the COVID-19 epidemic. As these social mitigation measures are implemented in a continuous action space that modulates the contact matrix of the age-structured epidemic model, we extend PCN to this setting. We evaluate the solution set that PCN returns, and observe that it correctly learns to reduce the social burden whenever the hospitalization rates are sufficiently low. In this work, we thus demonstrate that multi-objective reinforcement learning is attainable in complex epidemiological models and provides essential insights to balance complex mitigation policies.

Keywords Multi-objective reinforcement learning · Epidemic control · COVID-19 epidemic models

1 Introduction

As shown by the COVID-19 pandemic, infectious disease outbreaks represent a paramount global challenge that should be tackled by prevention strategies. To this end, understanding the complex dynamics that underlie these epidemics is essential. Epidemiological transmission models allow us to capture and understand such dynamics and facilitate the study of prevention strategies through simulation. However, developing efficient mitigation strategies remains a challenging process, given the non-linear and complex nature of epidemics.

For this reason, reinforcement learning provides a methodology to automatically learn prevention strategies in combination with complex epidemic models [Libin et al., 2021]. Previous research typically focuses on optimising policies with respect to a single objective, such as the pathogen's attack rate, while the mitigation of epidemics is a problem that inherently covers distinct and possibly conflicting criteria (i.a., prevalence, mortality, morbidity, cost). Therefore, optimizing on a single objective requires that these distinct criteria are somehow aggregated into a single metric. Generally, during learning, a decision maker will rely on an expert's assistance to manually design such a metric. However, a decision maker may be unaware of all possible trade-offs between different solutions, which may leave the decision maker unaware of possible solutions that better reflect their preferences, potentially resulting in sub-optimal performance. Furthermore, manually designing such metrics is time consuming, costly and error-prone, as this non-intuitive process requires repetitive and tedious tuning to achieve the desired behaviour [Hayes et al., 2021]. Moreover, taking a single objective approach has several other limiting factors [Hayes et al., 2021, Roijers et al., 2013].

To alleviate the challenging process of defining the learning problem explicitly, we can directly learn optimal behaviours by explicitly taking a multi-objective approach. By assuming that a decision maker will always prefer solutions for which at least one objective improves, it is possible to learn a set of optimal solutions called the *Pareto front*. This enables decision makers to review each solution on the Pareto front before making a decision, thereby being aware of the trade-offs that a solution may imply.

In this work, we investigate the use of *multi-objective reinforcement learning* (MORL) to learn a set of solutions that approximate the Pareto front of multi-objective mitigation strategies. We consider the first wave of the Belgian COVID-19 epidemic, which was mitigated by a hard lockdown [Willem et al., 2021]. When the incidence of confirmed cases were steadily dropping, epidemiological experts were asked to investigate strategies to exit from the stringent lockdown which was imposed. Here, we consider the epidemiological model developed by Abrams et al. [2021] that was constructed to describe the Belgian COVID-19 epidemic, and was fitted to hospitalisation incidence data and serial sero-prevalence data. This model constitutes a stochastic discrete-time age-structured compartmental model that simulates mitigation strategies by varying social distancing parameters concerning school, work and leisure contacts. Based on this model, we *contribute MOBelCov, a novel mutli-objective epidemiological environment*, in the form of a multi-objective Markov decision process (MOMDP). MOBelCov encapsulates the epidemiological model developed by Abrams et al. [2021] to implement state transitions, with an action space that combines a proportional reduction of school, work and leisure contacts at each time step and defines a reward function based on two objectives: the attack rate (i.e., proportion of the population affected by the pathogen) and social burden.

To learn and explore the trade-offs between the attack rate and social burden we build upon a state-of-the-art MORL approach, namely Pareto Conditioned Networks (PCN) [Reymond et al., 2022], which uses a single neural network to learn the policies that belong to the Pareto front. As PCN is an algorithm designed for discrete action-spaces, we extend it towards continuous action-spaces to accommodate MOBelCov's action-space. With this continuous action variant of PCN, we explore the Pareto front of multi-objective COVID-19 mitigation policies.

By evaluating the solution set of mitigation policies returned by PCN, we observe that PCN minimises the social burden in scenarios where hospitalization rates are sufficiently low. Therefore, in this work we illustrate that multi-objective reinforcement learning can be utilised to provide important insights surrounding the trade-offs between complex mitigation polices in real-world epidemiological models.

2 Background

2.1 Multi-Objective Reinforcement Learning

Real-world decisions problems often have multiple and possibly conflicting objectives. Multi-objective reinforcement learning (MORL) can be used to find optimal solutions for sequential decision making problems with multiple objectives [Hayes et al., 2021]. For MORL, problems are typically modelled as a multi-objective Markov decision process (MOMDP), i.e., a tuple, $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{T}, \gamma, \boldsymbol{\mathcal{R}})$, where $\mathcal{S}$, $\mathcal{A}$ are the state and action spaces respectively, $\mathcal{T}\colon \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ is a probabilistic transition function, γ is a discount factor determining the importance of future rewards and $\boldsymbol{\mathcal{R}}\colon \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}^n$ is an n-dimensional vector-valued immediate reward function, where n denotes the number of objectives. For single-objective RL, i.e., when $n = 1$, the goal is to find the policy π^* that maximizes the expected sum of discounted rewards, also called the expected return:

$$\pi^* = \arg\max_{\pi} \mathbb{E}\left[\sum_{t=0}^{h} \gamma^t r_t | \, \pi, s_0\right], \tag{1}$$

where $r_t = \mathcal{R}(s_t, a_t, s_{t+1})$. In contrast, in MORL, $n > 1$ which leads to vectorial returns. In this case, there can be policies for which, without any additional information, it is impossible to know if one is better than the other. For example, we cannot decide which policy between π_1, π_2 is optimal if they lead to expected returns (also called V-values) $\mathbf{V}^{\pi_1} = (0, 1), \mathbf{V}^{\pi_2} = (1, 0)$ respectively. We call these solutions *non-dominated*, i.e., solutions for which it is impossible to improve an objective without hampering another. The set that contains all the non-dominated solutions of the decision problem is called the *Pareto front* $\mathcal{F}$. Our goal is to find the set of policies that lead to all the V-values contained in the Pareto front $\Pi^* = \{\, \pi | \mathbf{V}^\pi \in \mathcal{F}\}$. In general, we call any set of V-values a *solution set*. When a solution set contains only non-dominated V-values, it is referred to as a *coverage set*. In the case that no π exists that has a $\mathbf{V}^\pi$ dominating any of the solutions in a coverage set, then this coverage set is the Pareto front.

2.2 Multi-Objective Metrics

Comparing the learned coverage sets of different algorithms is a non-trivial task, as one algorithm's output might dominate the other in some region of the objective-space, but be dominated in another. Intuitively, one would generally prefer the algorithm that covers a wider range of decision maker preferences. In this work we use several metrics to evaluate our algorithm's performance.

A widely used metric in the literature is called the *hypervolume* [Zitzler et al., 2003]. This metric evaluates the learned coverage set by computing its volume with respect to a fixed specified reference point. This metric is, by definition, the highest for the Pareto front, as no other possible solution can increase its volume (since they are all dominated). One disadvantage of the hypervolume is that it can be difficult to interpret. For example, when working in high-dimensional objective-spaces, adding or removing a single point can lead to wildly different hypervolume values, especially if the point lies close to an extremum of the space.

To alleviate these shortcomings, we additionally evaluate our work on a different metric called the ε-indicator I_ε [Zitzler et al., 2003], which measures how close a coverage set is to the Pareto front $\mathcal{F}$. Intuitively, I_ε shows that any solution of $\mathcal{F}$ is *at most* ε better with respect to each objective o than the closest solution of the evaluated coverage set:

$$I_\varepsilon = \inf_{\varepsilon \in \mathbb{R}} \{\forall\, \mathbf{V}^\pi \in \mathcal{F},\ \exists\, \mathbf{V}^{\pi'} \in \hat{\Pi} :\ ||V^\pi - V_o^{\pi'}||_\infty \leq \varepsilon\} \tag{2}$$

The main disadvantage of this metric is that we need the true Pareto front to compute it, which in the case of our MOMDP (see Section 3) is unknown. To still gain insights from our learned policies, we approximate the true Pareto front using the non-dominated policies across all runs.

We note that the ε-indicator metric is quite pessimistic, as it measures worst-case performance [Zintgraf et al., 2015], e.g., it will still report bad performance as long as a single point of the Pareto front is not correctly modeled, even if all the other points are covered. As such, we also use the $I_{\varepsilon-mean}$ [Reymond et al., 2022] which measures the *average* ε distance of the solutions in $\mathcal{F}$ with respect to the evaluated coverage set.

Figure 1 shows a visual representation of the hypervolume and ε metrics in two dimensions.

3 COVID-19 model and the MOBelCov MOMDP

3.1 Stochastic compartment model for SARS-CoV-2

As an environment to evaluate non-pharmaceutical interventions, we consider the adapted version of the SEIR compartmental model presented by Abrams et al., that was used to investigate exit strategies in Belgium after the first epidemic wave of SARS-CoV-2 [Abrams et al., 2021]. This model constitutes a discrete-time stochastic model, that considers an age-structured population. This model generalises a standard SEIR model[1], extended to capture the different stages of disease spread and history that are associated with SARS-CoV-2 (i.e., pre-symptomatic, asymptomatic, symptomatic with mild symptoms and symptomatic with severe symptoms) and to represent the stages associated

[1]A standard SEIR model divides the population into four different states, i.e., susceptible, exposed, infectious and recovered individuals.

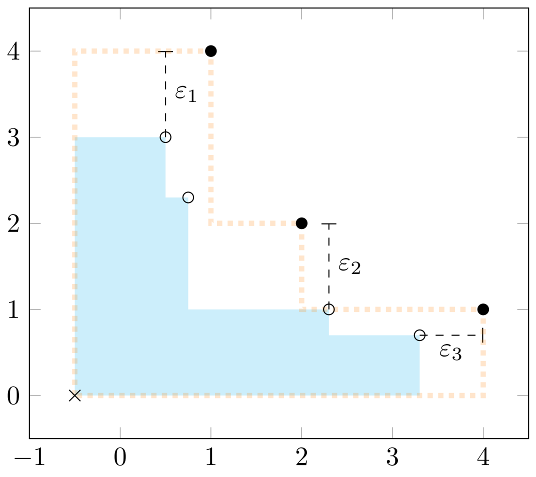


Figure 1: Example of a Pareto front (black dots) and a coverage set (white dots) in a 2-objective environment. The hypervolume metric (in light blue) measures the volume of all dominated solutions with respect to some reference point (cross). The ε metrics first compute the maximum distance between each point in the Pareto front and its closest point in the coverage set (ε_i). We can then take their maximum value to compute the I_ε metric, or their mean value to obtain the $I_{\varepsilon-mean}$ metric of the coverage set.

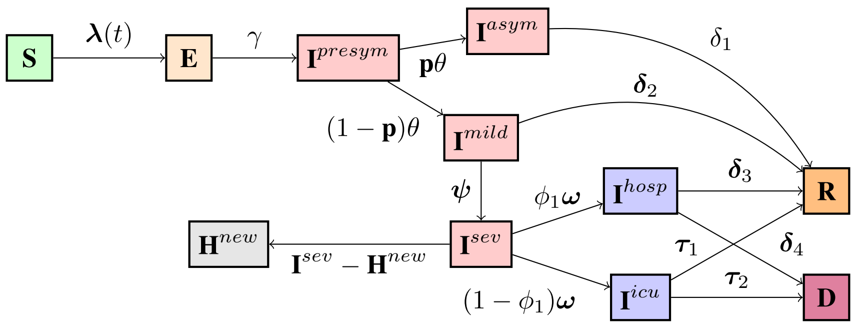


Figure 2: Schematic diagram of the compartmental model for SARS-CoV-2 presented by Abrams et al. [2021] which is used to derive the MOMDP.

with severe disease, i.e., hospitalisation, admission to the intensive care unit (ICU) and death. As we consider an age-structured population, we consider this extended SEIR structure for $K = 10$ age groups, i.e., $[0-10), [10-20), [20-30), [30-40), [40-50), [50-60), [60-70), [70-80), [80-90), [90, \infty)$. Contacts of the different age-groups which impact the propagation rate of the epidemic are modeled using social contact matrices (SCMs). We define a SCM for 6 different social environments: $C_{\text{home}}, C_{\text{work}}, C_{\text{transport}}, C_{\text{school}}, C_{\text{leisure}}, C_{\text{other}}$ for the home, work, transport, school, leisure, other environments respectively. The model is described by a set of ordinary differential equations (ODEs), as described in SI. By formulating this set of differential equations defined above as a chain-binomial process we can obtain stochastic trajectories from this model [Abrams et al., 2021].

3.2 Interventions strategies

In order to model different types of interventions, we follow Abrams et al. [2021]. Firstly, in order to consider distinct exit scenarios, we alter the SCMs to reflect a contact reduction in a particular age group. Secondly, we assume that compliance to the interventions is gradual and model this using a logistic compliance function (see details in SI). We consider a contact reduction function that imposes a proportional reduction of work (including transport) p_w, school p_s and leisure p_l contacts, which is implemented as a linear combination of contact matrices:

$$\hat{C} = C_{\text{home}} + p_w(C_{\text{work}} + C_{\text{transport}}) + p_s C_{\text{school}} + p_l(C_{\text{leisure}} + C_{\text{other}}) \quad (3)$$

3.3 The MOBelCov Environment

In order to apply multi-objective reinforcement learning, we construct the MOBelCov MOMDP based on the epidemiological model introduced in Section 3.1 and graphically depicted in Figure A.1.

Action-space: Our actions concern the installment of a social contact matrix $\hat{C}$, with a particular reduction (see Section A.4). To this end, we use the proportional reduction parameters p_w, p_s, p_l defined in Section A.4. Thus, each $\mathbf{a} \in \mathcal{A}$ is a 3-dimensional continuous vector in $[0, 1]^3$ (i.e., $\mathbf{a} = [p_w, p_s, p_l]$) which impacts the SCM according to Equation 3.

Transition function: The model defined by Abrams et al. [2021] utilises a model transition probability $M(\mathbf{s}'_m \mid \mathbf{s}_m, \hat{C})$ (see SI for details on M), where $\hat{C}$ the currently installed SCM, that progresses the epidemiological model in one timestep. We use this function as the transition function in MOBelCov. For each timestep t, we simulate the model for one week, using $\hat{C}$ obtained from $\mathbf{a}_t$.

As mentioned above and detailed in the SI, the compartment model is described by a set of ODEs. To obtain a stochastic version, the ODEs can be formulated as a chain-binomial process. Based on these cases, we also create and experiment on two versions of the MOBelCov environment: the (deterministic) *ODE model* (i.e., a MOMDP with a deterministic transition function) and the (stochastic) *Binomial model* (i.e., a MOMDP with a stochastic transition function).

In a classical MDP, executing an action a_t in any state s_t leads to a next state s_{t+1} according to the transition function $\mathcal{T}$. At every timestep, the policy is free to choose the action to perform. In our case, this corresponds to a different restriction $[p_w, p_s, p_l]$ every week. However, imposing new social restrictions every week does not seem very realistic. It can be frustrating to having to cope with the ever-changing rules. In case of the exit strategy after the lockdown of the first COVID wave in Belgium, the Belgian government executed a progressive plan that consisted of 3 different stages. Each stage resulted in the progressive lessening of social restrictions.

In order to simulate a more realistic multi-stage exit strategy, we incorporate a *budget per restriction-change* on the MOMDP. Concretely, if the social restriction proposed by the policy is different from the current one in place, we consider this a next stage and reduce our restriction-change budget by one. When the budget reaches 0, we consider the exit strategy to be final, and impose the current social restriction for the remainder of the episode. Finally, we can simulate a no-limit budget setting by setting the budget to the length of the episode.

State-space: The state of the MOMDP is a tuple that consists of 3 elements. The first element, $\mathbf{s}_m$, are the state-variables that make up the epidemiological model (see Figure A.1). They are required to simulate the underlying epidemiological model for a full week.

$\mathbf{s}_m$ directly corresponds to the aggregation of the state variables in the epidemiological model, i.e., a tuple,

$$\{S_k, E_k, I_k^{presym}, I_k^{asym}, I_k^{mild}, I_k^{sev}, I_k^{hosp}, I_k^{icu}, H_k^{new}, D_k, R_k\}, \tag{4}$$

for each age group $k \in \{1, \ldots, K\}$, where S encodes the members of the population who are susceptible to infection and E encodes the members of the population who have been exposed to COVID-19. Moreover, I^{presym}, I^{asym}, I^{mild}, I^{hosp}, I^{icu} are the members of the population infected with COVID-19 and are, respectively, presymptomatic, asymptomatic, have mild symptoms, are hospitalised, or are in the ICU. Finally, H_k^{new} represents the number of newly hospitalised individuals in age group k.

We parameterize the epidemiological model using the mean of the posteriors as reported by Abrams et al. [2021] (details in SI).

The second element of the tuple consists of the social contact matrix $\hat{C}$ that is currently in place. The reason to incorporate it in the state-space is two-fold. First, Abrams et al. [2021] define a compliance function, simulating the time people need to get used to the new rules set in place. As such, during the simulated week, there is a gradual shift from the current $\hat{C}$ to the next one. We thus require to know the current $\hat{C}$.

Secondly, we require the current $\hat{C}$ to determine if we enter a new stage of the multi-stage exit strategy – and thus consume part of the restriction-change budget – or not.

The third and last element of the tuple consists of the restriction-change budget b. We note that we incorporate a separate budget per action-dimension, so p_w, p_s and p_l each have their own budget, $b = [b_w, b_s, b_l]$. It is thus possible that, at timestep t, the budget for one of the dimensions is reduced but not the others.

Therefore, we define a state in MOBelCov as follows:

$$\mathbf{s} = \mathbf{s}_m \cup \hat{C} \cup b \tag{5}$$

Reward function: We define a vector reward function which considers multiple objectives: attack rate (i.e., infections, hospitalisations) and the burden imposed by the interventions on the population.

The attack rate in terms of infections is defined as the difference in susceptibles from the current state and next state [Libin et al., 2021]. Since this is a cost that needs to be minimized, we defined the corresponding reward function as the negative attack rate:

$$\mathcal{R}_{\text{ARI}}(\mathbf{s}, \mathbf{a}, \mathbf{s}') = -(\sum_{k=1}^{K} S_k(\mathbf{s}) - \sum_{k=1}^{K} S_k(\mathbf{s}')). \tag{6}$$

The reward function to reduce the attack rate in terms of hospitalisations is simply defined as the negative number of new hospitalizations:

$$\mathcal{R}_{\text{ARH}}(\mathbf{s}, \mathbf{a}, \mathbf{s}') = -\sum_{k=1}^{K} H_k^{\text{new}}(\mathbf{s}) \tag{7}$$

Finally, we use the missed contacts resulting from the intervention measures as a proxy for societal burden. To quantify missed contacts, we consider the original social contact matrix C and the installed social contact matrix $\hat{C}$, and compute the difference $\hat{C} - C$. The resulting difference matrix quantifies the average frequency of contacts missed. To determine missed contacts for the entire population, we apply the calculated difference matrix to the population sizes of the respective age groups that are currently uninfected (i.e., susceptible and recovered individuals). Therefore we define the social burden reward function $\mathcal{R}_{\text{SB}}$, as follows:

$$\mathcal{R}_{\text{SB}}(\mathbf{s}, \mathbf{a}, \mathbf{s}') = \sum_{i=1}^{K}\sum_{j=1}^{K}(\hat{C} - C)_{ij} S_i(\mathbf{s}) S_j(\mathbf{s}) + \sum_{i=1}^{K}\sum_{j=1}^{K}(\hat{C} - C)_{ij} R_i(\mathbf{s}) R_j(\mathbf{s}), \tag{8}$$

where $S_k(\mathbf{s})$ represents the number of susceptible individuals in age group k in state $\mathbf{s}$ and R_k represents the number of recovered individuals in age group k in state $\mathbf{s}$. In Section 5, we optimize PCN on two different variants for the multi-objective reward function: $[\mathcal{R}_{\text{ARH}}, \mathcal{R}_{\text{SB}}]$ and $[\mathcal{R}_{\text{ARI}}, \mathcal{R}_{\text{SB}}]$.

4 Pareto Conditioned Networks

In multi-objective optimization, the set of optimal policies can grow exponentially in the number of objectives. Thus, recovering them all is an expensive process and requires an exhaustive exploration of the complete state space. To address this problem, we use Pareto Conditioned Networks (PCN), a method that uses a single neural network to encompass all non-dominated policies [Reymond et al., 2022]. The key idea behind PCN is to use supervised learning techniques to improve the policy instead of resorting to temporal-difference learning.

TODO REPHRASE, FROM PCN PAPER: Using neural networks as function approximators in RL comes with many challenges. One of them is that the target (e.g., the optimal action of the policy) is not known in advance — as opposed to classical supervised learning where the ground-truth target is provided. As the behavior of the agent improves over time, the action used as target can change, often leading to hard-to-tune and brittle learners [**??**].

Instead of trying to continuously improve the policy by learning actions that should lead to the highest cumulative reward, PCN flips the problem, by learning actions that should lead to any *desired* cumulative reward (be it high or low). In this way, all past trajectories can be reused for supervision, since their returns are known, as well as the actions needed to reach said returns. We can thus train a policy that, conditioned on a desired return, provides the optimal action to reach said return. By leveraging the generalization properties of neural networks, we can accumulate incrementally better experience by conditioning on increasingly higher reward-goals

PCN uses a single neural network that takes a tuple $\langle \mathbf{s}, \hat{h}, \hat{\mathbf{R}} \rangle$ as input. $\hat{\mathbf{R}}$ represents the *desired return* of the decision maker, i.e. the return PCN should reach at the end of the episode. $\hat{h}$ denotes the *desired horizon* that expresses the number of timesteps that should be executed before reaching $\hat{\mathbf{R}}$. At execution time, both $\hat{h}$ and $\hat{\mathbf{R}}$ are chosen by the decision maker at the start of the episode. Then, at every timestep, the desired horizon is updated according to the perceived reward $\mathbf{r}_t$, $\hat{\mathbf{R}} \leftarrow \hat{\mathbf{R}} - \mathbf{r}_t$ and the desired horizon is decreased by one, $\hat{h} \leftarrow \hat{h} - 1$.

PCN's neural network has an output for each action $a_i \in \mathcal{A}$. Each output represents the confidence the network has that, by taking the corresponding action in $\mathbf{s}$, $\hat{\mathbf{R}}$ will be reached in $\hat{h}$ timesteps. We can draw an analogy with a classification problem where the network should learn to classify $(\mathbf{s}, \hat{h}, \hat{\mathbf{R}})$ to its corresponding label a_i.

Similar to classification, PCN requires a labeled dataset with training examples to learn a mapping from input to label. However, contrary to classification, the data present in the dataset is not fixed. PCN collects data from the trajectories experienced while exploring the environment. Thus, the dataset improves over time, as we collect better and better trajectories.

Concretely, the PCN algorithm is a loop that repeats 3 main steps:

1. Using the current policy to interact with the environment and generate trajectories
2. Update and prune the dataset to incorporate these new trajectories and remove the least interesting trajectories in terms of returns
3. Update the policy by training it on the updated dataset

4.1 Building and updating the dataset

Given a trajectory composed of T transitions $(s_0, a_0, \boldsymbol{r}_0, s_1), \ldots, (s_{T-1}, a_{T-1}, \boldsymbol{r}_{T-1}, s_T)$ we can compute, for each transition at timestep t, the future discounted return $\boldsymbol{R}_t = \sum_{i=t}^{T} \gamma^i \boldsymbol{r}_i$ and the leftover horizon $h_t = T - t$. Since for this trajectory executing action a_t in state s_t resulted in return $\mathbf{R}_t$ in h_t timesteps, we add a datapoint with input $\langle s, \hat{h}, \hat{\mathbf{R}} \rangle = \langle s_t, h_t, \mathbf{R}_t \rangle$ and output $a = a_t$ to the dataset. In other words, when the observed return corresponds to the desired return in that state, then a_t is the optimal action to take.

Since we have a fix-sized dataset, adding this trajectory means we need to remove another one. Our aim is to keep trajectories that span different parts of the objective space, such that we can explore as many types of trade-offs as possible. However, to improve the trade-offs we currently have in our dataset, we need to only keep solutions that are close to our current estimate of the Pareto front. Thus, we assign a priority for each trajectory in our dataset that combine two metrics: the *crowding distance*, which measures the distance of a solution with its closest neighbors, and the *l2-norm* with the closest non-dominated solution, which indicates how close the solution is from our current estimate of the Pareto front.

The dataset is then updated by only keeping the trajectories with the top-N priorities.

4.2 Generating trajectories with the current policy

In the previous section, we explain how PCN updates the dataset given a trajectory. In this section, we explain how PCN produces said trajectory.

It is unrealistic to expect PCN to reliably produce trajectories with high-valued desired returns when it has only been trained on datapoints originating from random trajectories. Rather, we can expect PCN to produce trajectories with returns in the range of the ones from the current training data. Therefore, if we obtain trajectories reaching high returns, PCN will be able to confidently return high-return policies.

PCN leverages the fact that, due to the generalization capabilities of neural networks, the policies obtained from the network will still be reliable even if the desired return is marginally higher than what is present in the training data. In fact, they will perform similar actions to those in the training data, but lead to a higher return. Thus, we incrementally condition the network on better and better returns, in order to obtain trajectories that extend the boundaries of PCN's current coverage set.

More precisely, we randomly select a non-dominated return $\mathbf{R}_{nd}$ and its corresponding horizon $\hat{h}$ from the dataset. By randomly picking a non-dominated return from the entire coverage set we ensure equal chance of improvement to each part of the objective space. To generate trajectories that improve upon $\mathbf{R}_{nd}$, we randomly select an objective o and increase $R_{nd,o}$ by a sample from the uniform distribution $U(0, \sigma_o)$, where σ_o is the standard deviation for the selected objective, using all non-dominated returns from the trajectories in the dataset. This then becomes our desired return $\hat{\boldsymbol{R}}$. By restricting the improvement to at most σ_o, $\hat{\mathbf{R}}$ stays in the range of possible achievable returns and, by only modifying one objective at a time, the changes to the network's input compared to the training data are kept at a minimum.

4.3 Training the network for continuous actions

PCN trains the network as a classification problem, where each class represents a different action. Transitions $x = \langle \mathbf{s}_t, h_t, \mathbf{R}_t \rangle, y = a_t$ are sampled from the dataset, and the ground-truth output y is compared with the predicted output $\hat{y} = \pi(\mathbf{s}_t, h_t, \mathbf{R}_t)$. The predictor (i.e., the policy) is then updated using the cross-entropy loss function:

$$H = -\sum_{a \in \mathcal{A}} y_a \log \pi(a|\mathbf{s}_t, h_t, \mathbf{R}_t) \tag{9}$$

where $y_a = 1$ if $a = a_t$ and $y_a = 0$ otherwise.

While the original PCN algorithm is designed for MOMDPs with discrete actions-spaces, the problem we tackle (see Section 3) is defined in terms of a continuous action-space. We thus extend PCN for the continuous action-space setting. First, we change the output of the neural network such that there is a single output value for each dimension of the action-space. Since the actions should be bound in the domain of possible actions ($[0, 1]$ in the case of MOBelCov, see Section 3), we apply a tanh non-linearity function on this output. Second, the problem becomes a regression problem instead of a classification problem, as the labeled dataset uses continuous labels $y = \mathbf{a}_t$ instead of categories. We thus use a Mean Squared Error (MSE) loss to update the policy:

$$MSE = \frac{1}{|\mathcal{A}|} \sum_{a \in \mathcal{A}} (\hat{y}_a - y_a)^2 \tag{10}$$

Since learning the full set of Pareto-efficient policies Π^* requires that the policies $\pi^* \in \Pi^*$ are deterministic stationary policies [Roijers et al., 2013], we use the output $\hat{\mathbf{y}}$ as action at execution time. However, PCN improves its policy through exploration, by continuously updating its dataset with better trajectories. Thus, at training time, we follow [Lillicrap et al., 2015] and use a stochastic policy by adding random noise sampled from a Normal distribution to the action:

$$\mathbf{a}_t = \pi(\mathbf{s}_t, h_t, \mathbf{R}_t) + \eta s \text{ with } s \sim \mathcal{N}, \tag{11}$$

where η is a hyper-parameter defining the magnitude of noise to be added.

4.4 Coping with stochastic transitions

PCN trains its policy on a dataset that is collected by executing trajectories. It assumes that reenacting a transition from the dataset leads to the same episodic return. When the transition function $\mathcal{T}$ of the MOMDP is deterministic, the whole trajectory can be faithfully reenacted, which guarantees that we obtain the same return. Combined with the fact that PCN's policy is deterministic at execution time, conditioning the policy on a target episodic return is equivalent to conditioning it on the V-value $\mathbf{V}$.

However, when $\mathcal{T}$ is stochastic we lose this guarantee. We cope with this issue by adding small random noise to $\mathbb{R}_t$ when performing gradient descent, which reduces the risk of overfitting [Zur et al., 2009]. Moreover, although the Binomial model of MOBelCov is stochastic, the possible next-states resulting from a state-action pair are similar to each other. This allows PCN to compensate if $\mathbf{r}_t = \boldsymbol{\mathcal{R}}(\mathbf{s}_t, \mathbf{a}_t, \mathbf{s}_{t+1})$ is somewhat worse than expected. This is confirmed by our experiments (see Section 5), where the coverage sets learned by PCN on the ODE model and on the Binomial model are similar.

5 Analysing COVID-19 deconfinement policies

Our goal is to use PCN to learn deconfinment strategies in the MOBelCov environment. We aim to obtain policies that balance between the social burden experienced by the population and the epidemiological objective of minimising the attack rate. To this end we consider two cases for the vectorial reward functions $[\mathcal{R}_{\text{ARH}}, \mathcal{R}_{\text{SB}}]$ and $[\mathcal{R}_{\text{ARI}}, \mathcal{R}_{\text{SB}}]$, to learn and analyse policies under different targets with respect to the considered attack rate.

We now evaluate our extension of PCN for continuous action-spaces on both the ODE and Binomial models. Conform to Abrams et al. [2021], the simulation starts on the 1st of March 2020, by seeding a number of infections in the population. Two weeks later, on the 14th of March, the Belgian government initiated a full lockdown of the country. This is implemented by fixing the actions p_w, p_s, p_l to $0.2, 0, 0.1$ respectively. This lockdown ended on the 4th of May 2020, at which point the government decided on a multi-phase exit strategy to incrementally reduce teleworking, reopen schools and allow leisure activities, such as the re-opening of bars and the cultural sector. It is from this day onward that PCN aims to learn policies for diverse exit strategies, compromising between the total number of daily new hospitalizations and the total number of contacts lost as a proxy for social burden. The simulation lasts throughout the school holidays (from 01/07/2020 to 31/08/2020). Schools are closed during the school holidays, which is simulated by setting $p_s = 0$, regardless of the corresponding value outputted by the policy, i.e., during periods of school closure p_s is ignored.

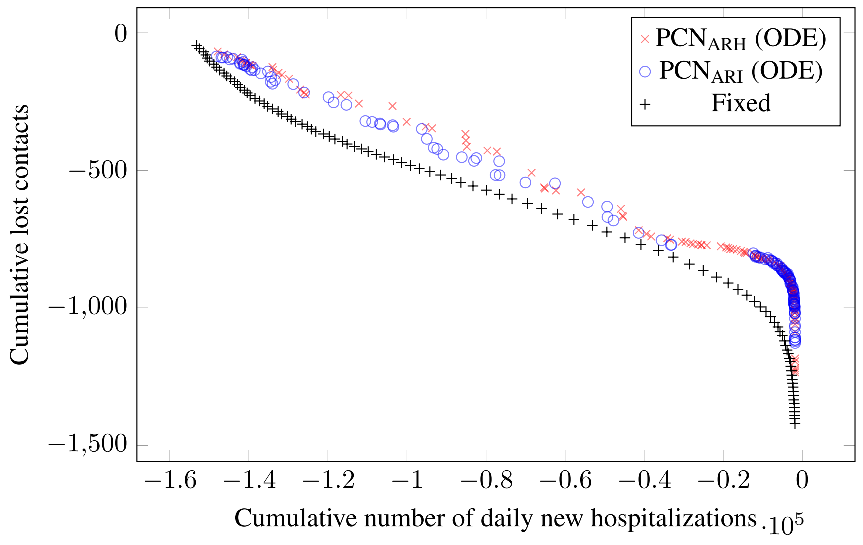


Figure 3: The Pareto front of policies discovered by PCN using the ODE model (Binomial omitted for clarity, see SI), showing the different compromises between the number of hospitalizations and the number of lost contacts. Although PCN_{ARI} was trained on the number of infections, it still shows a competitive coverage set with respect to hospitalizations.

To evaluate the quality of the policies discovered by PCN, we compare it to a baseline. The baseline consists of a set of 100 fixed policies, that iterate over all the possible social restriction levels, with values ranging from 0 to 1. In other words, the fixed policies directly operate in a fine-grained manner on the whole contact reduction function $\hat{C}$. This allows us to obtain a strong baseline for potential exit strategies over the objective space. We note that while such fixed policies are a feasible approach, they do not scale well in terms of action and objective spaces and they will not be able to provide an adaptive restriction level, which is what we aim to provide using PCN.

All experiments are averaged over 5 runs. The hyper-parameters and the neural network architecture can be found in the SI.

5.1 Learned coverage set

We learn a coverage set that ranges from imposing minimal restrictions to enforcing many restrictions (see Figure 4). The coverage set is shown in Figure 3.

We notice that the coverage sets discovered by PCN almost completely dominate the coverage set of the baseline, showing that there are much better alternatives to the fixed policies. This is most visible in the compromising policies, where one has to carefully choose when to remove social restrictions while at the same time minimizing the impact on daily new hospitalizations. In these scenarios, PCN discovers policies that can drastically reduce the total number of new hospitalizations (e.g. more than 20000) for the same social burden. Analysing the executions of such policies (see Figure 4, middle plot) shows a flattened hospitalization curve, with a gradual increase of social freedom during the school holidays such that the peak stays stable and gradually decreases over time.

Interestingly, we notice that the most restrictive policy (i.e. the one that prioritizes hospitalizations over social burden, see Figure 4, bottom plot) still starts to gradually increase p_w and p_l from the end of July onward. This is because by then, the epidemic has mostly faded out, and it is safe to reduce social restrictions. The timing of this reduction is important as reducing restrictions too soon can lead to a new wave. PCN learns the impact of its decisions over time, and correctly infers the timing at which restrictions can be safely lifted.

5.2 ODE versus Binomial models

Next, we compare the performance of PCN when trained on the ODE and Binomial models. The ODE model has the advantage of using a deterministic transition function, for which PCN is suited. However, taking into account stochasticity, using the Binomial model, is important as individual behavioural changes introduce substantial uncertainty in the course of the outbreak and require stochastic model evaluations to evaluate the impact of this uncertainty on the effectiveness of the intervention strategies. Regardless, the results displayed in Table 1 show that the solution sets

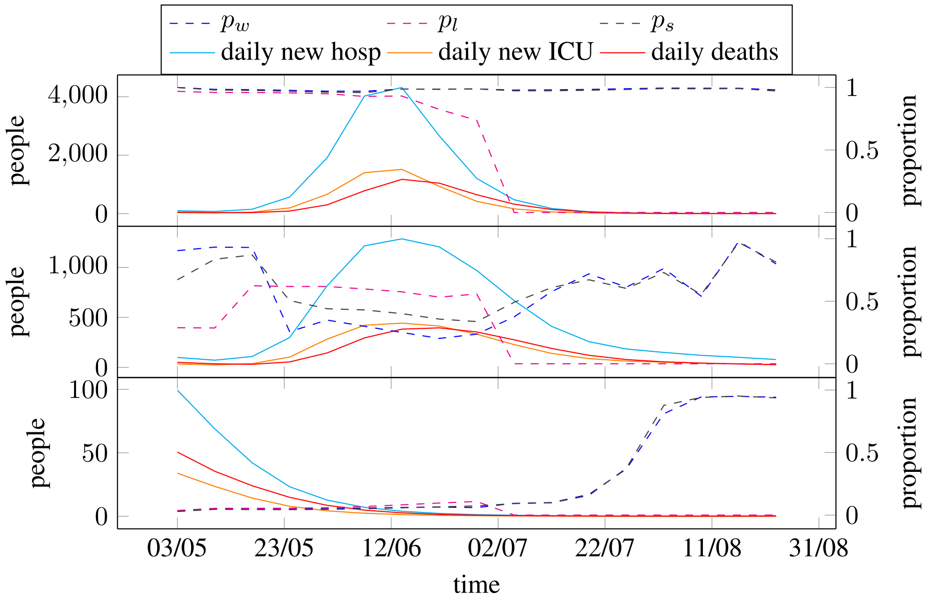


Figure 4: Selection of policies learned by PCN, from most restrictive in terms of social burden (top) to least restrictive (bottom).

	Hypervolume	I_ε	$I_{\varepsilon-mean}$
PCN$_{\text{ARH}}$ (ODE)	$\mathbf{0.679 \pm 0.010}$	$\mathbf{0.056 \pm 0.004}$	$\mathbf{0.010 \pm 0.006}$
PCN$_{\text{ARH}}$ (Binomial)	0.674 ± 0.005	0.085 ± 0.044	0.017 ± 0.005
PCN$_{\text{ARI}}$ (ODE)	0.668 ± 0.009	0.084 ± 0.052	0.021 ± 0.011
PCN$_{\text{ARI}}$ (Binomial)	0.659 ± 0.007	0.100 ± 0.054	0.028 ± 0.013
Fixed	0.6 ± 0.0	0.106 ± 0.0	0.058 ± 0.0

Table 1: Evaluation metrics for the coverage sets comparing hospitalizations with social burden. In general training on the ODE results in slightly better coverage sets than on the Binomial model. Training on infections (ARI) still provides a competitive coverage set in terms of hospitalizations. All PCN coverage sets outperform the baseline.

that PCN finds using the Binomial model are competitive with the ODE model, as indicated by their hypervolumes and $I_{\varepsilon-mean}$ metrics. We do note that the difference in I_ε indicator is more significant, indicating that a few solutions discovered using the ODE model are not found when using the Binomial model, impacting worst-case performance. Overall, we see that our extension of PCN performs efficiently, even with some degree of stochasticity in the transition function.

5.3 R_{ARH} versus on R_{ARI}

We now assess the difference in coverage sets when optimizing on $\mathcal{R}_{\text{ARH}}$ versus $\mathcal{R}_{\text{ARI}}$. Although at a different scale, our experiments show that infections and hospitalizations are correlated. This is confirmed in Figure 3: the coverage set *in terms of hopsitalizations* learned by PCN$_{\text{ARI}}$ is only slightly worse than the one learned by PCN$_{\text{ARH}}$, even though PCN$_{\text{ARI}}$ optimized *on the infection attack rate*. This is expected, as during the initial phase of the epidemic, limited immunity was present in the population (few natural immunity and no vaccines), which induces a tight coupling between infection and hospitalisation cases. Moreover, the opposite also holds: both coverage sets are similar in terms of *infections*. However, there is one notable exception. A number of policies learned by PCN$_{\text{ARH}}$ start removing social restrictions before the epidemic is rooted out, resulting in a rise of hospitalizations at the end of the summer holidays. Since, following Abrams et al., this is when the simulation stops, the total number of hospitalizations remains low. The number of infections, however, starts growing before this rise in hospitalizations and, since its growth is exponential, ends up high. We note that PCN$_{\text{ARH}}$ also learns alternative policies for similar $[\mathcal{R}_{\text{ARH}}, \mathcal{R}_{\text{SB}}]$ that do not exhibit this behavior. This shows that, even though a policy might lead to a return that matches the preferences of the decision maker, its execution might still lead to undesired behavior. We argue that the use of such expert systems for decision making should be paired with careful interpretation.

	I_ε	$I_{\varepsilon-mean}$
PCN_{ARH} (ODE)	0.046 ± 0.012	0.007 ± 0.003
PCN_{ARH} (Binomial)	0.068 ± 0.021	0.024 ± 0.002
PCN_{ARI} (ODE)	0.056 ± 0.013	0.008 ± 0.004
PCN_{ARI} (Binomial)	0.058 ± 0.012	0.011 ± 0.003

Table 2: Comparing the difference in the desired return provided to PCN and the actual return PCN obtained when executing its policy. We see that, regardless of the setting, the learned policy faithfully receives a return similar to its desired return.

5.4 Robustness of policy executions

The dataset of trajectories that PCN is trained on is pruned over time to keep only the most relevant trajectories. The returns of these trajectories are used in Figure 3 to show the discovered coverage set. Each of these returns can be used as desired return for policy execution. We now assess the robustness of the executed policies, by comparing the averaged return obtained over multiple policy executions with the corresponding target return. We show that the executed policies reliably obtain returns that are similar to the desired return used to condition PCN.

Results are shown in Table 2. The I_ε indicators shows that, over all policies, the decision maker will lose at worst 0.046 and 0.068 normalized returns in any of the objectives for the ODE, Binomial versions respectively. On average, it will lose 0.007, 0.024 normalized returns respectively, which corresponds to either a total of $3633, 1059$ more hospitalizations than expected or a total of $262, 76$ less contacts than expected.

6 Related work

Reinforcement learning (RL) has been used in conjunction with epidemiological models to learn policies to limit the spread of diseases and predict the effects of possible mitigation strategies [Probert et al., 2019, Ernst et al., 2006]. For example, RL has been used extensively in modelling and controlling the spread of influenza [Das et al., 2008, Libin et al., 2021, 2018].

RL and Deep RL have been used extensively as a decision making aid to reduce the spread of COVID-19. For example, to learn effective mitigation strategies [Ohi et al., 2020], to learn efficacy of lockdown and travel restrictions [Kwak et al., 2021] and to limit the influx of asymptomatic travellers [Bastani et al., 2021].

Multi-objective methods have also been deployed to learn optimal strategies to mitigate the spread of COVID-19. Wan et al. [Wan et al., 2021] implement a model-based multi-objective policy search method and demonstrate their method on COVID-19 data from China. Given the method is model-based, a model of the transition function must be learned by sampling from the environment. The method proposed by Wan et al. [Wan et al., 2021] only considers a discrete action space which limits the application of their algorithm. Wan et al. [Wan et al., 2021] use linear weights to compute a set of Pareto optimal policies. However, methods which use linear weights can only learn policies on the convex-hull of the Pareto front [Vamplew et al., 2008], therefore the full Pareto front cannot be learned. It is important to note the method proposed by Kompella et al. [Kompella et al., 2020] considers multiple objectives. However, the objectives are combined using a weighted sum with hand-tuned weights which are determined by the authors. The weighted sum is applied by the reward function and a single objective RL method is used to learn a single optimal policy. In contrast to previous work, our approach makes no assumptions regarding the scalarisation function of the user and is able to discover Pareto fronts of arbitrary shape.

7 Conclusion and discussion

Making decisions on how to mitigate epidemics has important ethical implications with respect to public health and societal burden. In this regard, it is crucial to approach this decision making from a balanced perspective, to which end we argue that multi-objective decision making is crucial. In this work, we establish a novel approach, i.e., an expert system, to study multi-faceted policies, and this approach shows great potential with respect to future epidemic control. We are aware of the ethical implications that expert systems have on the decision process and we make the disclaimer that all results based on, or derived from, the expert system that we propose should be carefully interpreted by experts in the field of public health, and in a much broader context of economics, well-being and education. We note that the work in this manuscript was conducted by a multi-disciplinary consortium that includes computer scientists and scientists with a background public health, epidemiology and bio-statistics.

In this work, we focus on the clinical outcomes of the intervention strategies and use the reduced contacts as proxy for the quality of life lost. This could be extended into more formal economic evaluations by assessing the health and monetary benefits and costs of different interventions for various stakeholders. The COVID-19 pandemic demonstrated the broad impact of infectious diseases on sectors outside health care. This stresses the need to capture a societal and thus multi-objective perspective in the decision making process on public health and health care interventions. Our learned policies confirm this, showing that focusing solely on reducing the number of hospitalizations results in taking drastic measures – more than a thousand social interactions lost per person over the span of 4 months – that may have a long-lasting impact on the population.

To conclude, we show that multi-objective reinforcement learning can be used to learn a wide set of high-quality policies on real-world problems, providing the decision maker with insightful and diverse alternatives, and showing the impact of taking extreme measures.

References

Pieter J. K. Libin, Arno Moonens, Timothy Verstraeten, Fabian Perez-Sanjines, Niel Hens, Philippe Lemey, and Ann Nowé. Deep reinforcement learning for large-scale epidemic control. In Yuxiao Dong, Georgiana Ifrim, Dunja Mladenić, Craig Saunders, and Sofie Van Hoecke, editors, *ECML*, pages 155–170, Cham, 2021. Springer International Publishing. ISBN 978-3-030-67670-4.

Conor F. Hayes, Roxana Rădulescu, Eugenio Bargiacchi, Johan Källström, Matthew Macfarlane, Mathieu Reymond, Timothy Verstraeten, Luisa M. Zintgraf, Richard Dazeley, Fredrik Heintz, Enda Howley, Athirai A. Irissappane, Patrick Mannion, Ann Nowé, Gabriel Ramos, Marcello Restelli, Peter Vamplew, and Diederik M. Roijers. A practical guide to multi-objective rl and planning, 2021.

Diederik M Roijers, Peter Vamplew, Shimon Whiteson, and Richard Dazeley. A survey of multi-objective sequential decision-making. *JAIR*, 48:67–113, 2013.

Lander Willem, Steven Abrams, Pieter JK Libin, Pietro Coletti, Elise Kuylen, Oana Petrof, Signe Møgelmose, James Wambua, Sereina A Herzog, Christel Faes, et al. The impact of contact tracing and household bubbles on deconfinement strategies for covid-19. *Nature communications*, 12(1):1–9, 2021.

Steven Abrams, James Wambua, Eva Santermans, Lander Willem, Elise Kuylen, Pietro Coletti, Pieter Libin, Christel Faes, Oana Petrof, Sereina A Herzog, et al. Modelling the early phase of the belgian covid-19 epidemic using a stochastic compartmental model and studying its implied future trajectories. *Epidemics*, 35:100449, 2021.

Mathieu Reymond, Bargiacchi Eugenio, and Ann Nowè. Pareto conditioned networks. In *Proceedings of the 21st International Conference on AAMAS (2022)*, 2022.

Eckart Zitzler, Lothar Thiele, Marco Laumanns, Carlos M Fonseca, and Viviane Grunert Da Fonseca. Performance assessment of multiobjective optimizers: An analysis and review. *IEEE Transactions on evolutionary computation*, 7 (2):117–132, 2003.

Luisa M Zintgraf, Timon V Kanters, Diederik M Roijers, Frans Oliehoek, and Philipp Beau. Quality assessment of morl algorithms: A utility-based approach. In *Benelearn 2015: proceedings of the 24th annual ML conference of Belgium and the Netherlands*, 2015.

Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

Richard M Zur, Yulei Jiang, Lorenzo L Pesce, and Karen Drukker. Noise injection for training artificial neural networks: A comparison with weight decay and early stopping. *Medical physics*, 36(10):4810–4818, 2009.

William JM Probert, Sandya Lakkur, Christopher J Fonnesbeck, Katriona Shea, Michael C Runge, Michael J Tildesley, and Matthew J Ferrari. Context matters: using reinforcement learning to develop human-readable, state-dependent outbreak response policies. *Philosophical Transactions of the Royal Society B*, 374(1776):20180277, 2019.

Damien Ernst, Guy-Bart Stan, Jorge Goncalves, and Louis Wehenkel. Clinical data based optimal sti strategies for hiv: a reinforcement learning approach. In *Proceedings of the 45th IEEE Conference on Decision and Control*, pages 667–672. IEEE, 2006.

Tapas K Das, Alex A Savachkin, and Yiliang Zhu. A large-scale simulation model of pandemic influenza outbreaks for development of dynamic mitigation strategies. *Iie Transactions*, 40(9):893–905, 2008.

Pieter JK Libin, Timothy Verstraeten, Diederik M Roijers, Jelena Grujic, Kristof Theys, Philippe Lemey, and Ann Nowé. Bayesian best-arm identification for selecting influenza mitigation strategies. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 456–471. Springer, Cham, 2018.

Abu Quwsar Ohi, MF Mridha, Muhammad Mostafa Monowar, Md Hamid, et al. Exploring optimal control of epidemic spread using rl. *Scientific reports*, 10(1):1–19, 2020.

Gloria Hyunjung Kwak, Lowell Ling, and Pan Hui. Deep reinforcement learning approaches for global public health strategies for covid-19 pandemic. *PLOS ONE*, 16(5):1–15, 05 2021.

Hamsa Bastani, Kimon Drakopoulos, Vishal Gupta, Ioannis Vlachogiannis, Christos Hadjicristodoulou, Pagona Lagiou, Gkikas Magiorkinis, Dimitrios Paraskevis, and Sotirios Tsiodras. Efficient and targeted covid-19 border testing via rl. *Nature*, 599(7883):108–113, 2021.

Runzhe Wan, Xinyu Zhang, and Rui Song. Multi-objective model-based reinforcement learning for infectious disease control. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 1634–1644, 2021.

Peter Vamplew, John Yearwood, Richard Dazeley, and Adam Berry. On the limitations of scalarisation for multi-objective reinforcement learning of pareto fronts. In *Australasian joint conference on artificial intelligence*, pages 372–378. Springer, 2008.

Varun Kompella, Roberto Capobianco, Stacy Jong, Jonathan Browne, Spencer Fox, Lauren Meyers, Peter Wurman, and Peter Stone. Reinforcement learning for optimization of covid-19 mitigation policies. *arXiv preprint arXiv:2010.10560*, 2020.

Jacco Wallinga, Peter Teunis, and Mirjam Kretzschmar. Using data on social contacts to estimate age-specific transmission parameters for respiratory-spread infectious agents. *American journal of epidemiology*, 164(10): 936–944, 2006.

Norman TJ Bailey. The mathematical theory of infectious diseases and its applications. In *The mathematical theory of infectious diseases and its applications*, pages 413–413. Charles Griffin & Company Ltd 5a Crendon Street, High Wycombe, Bucks HP13 6LE., 1975.

Alireza Beigi, Amin Yousefpour, Amirreza Yasami, JF Gómez-Aguilar, Stelios Bekiros, and Hadi Jahanshahi. Application of reinforcement learning for effective vaccination strategies of coronavirus disease 2019 (covid-19). *The European Physical Journal Plus*, 136(5):1–22, 2021.

Raghav Awasthi, Keerat Kaur Guliani, Saif Ahmad Khan, Aniket Vashishtha, Mehrab Singh Gill, Arshita Bhatt, Aditya Nagori, Aniket Gupta, Ponnurangam Kumaraguru, and Tavpritesh Sethi. Vacsim: Learning effective strategies for covid-19 vaccine distribution using reinforcement learning. *arXiv preprint arXiv:2009.06602*, 2020.

A Compartmental model for SARS-CoV-2

In this work we utilise the compartmental model proposed by Abrams et al. [2021] and extend the model to a multi-objective Markov decision process (MOMDP). The MOMDP used in this work is outlined in the main paper. However, we describe the details of the underlying compartmental model used to model the spread of COVID-19 below.

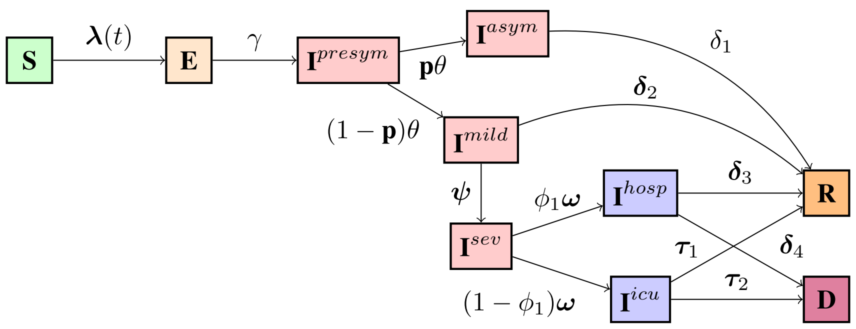


Figure A.1: Schematic diagram of the compartmental model for SARS-CoV-2 presented by Abrams et al. [2021] which is used to derive the MOMDP.

We utilise an adapted SEIR mathematical compartmental model proposed by Abrams et al. to contruct the MOBelCov MOMDP with deterministic and stochastic state transitions. In this model members of the population are susceptible to infection when they are in compartment $\mathbf{S}$[2]. If an individual comes into contact with an infections individual then they

[2]Boldface vector notation is used to denote the multiple age groups for each compartment.

move to the exposed compartment, **E**, at a time specific rate, $\boldsymbol{\lambda}(t)$. After a period of time an exposed individual becomes infectious, where they move to the pre-symptomatic compartment, $\mathbf{I}^{presym}$, at rate γ. Once infected, individuals develop mild symptoms, $\mathbf{I}^{mild}$, with probability $1 - \mathbf{p}$ or do not develop any symptoms, $\mathbf{I}^{asym}$, with probability **p**, where asymptomatic individuals recover, **R**, at rate δ_1. Individuals who experience symptoms can suffer from a mild infection, $\mathbf{I}^{mild}$, and recover at rate $\boldsymbol{\delta}_2$, or they suffer from a more severe infection, $\mathbf{I}^{sev}$, at a rate $\boldsymbol{\psi}$. Individuals with a severe infection are then transferred to a hospital for treatment, $\mathbf{I}^{hosp}$ with probability ϕ_1. However, some individuals become critically ill and are transferred directly to the intensive care unit (ICU) with probability $1 - \psi_1$. Individuals in the hospital, $\mathbf{I}^{hosp}$ and $\mathbf{I}^{icu}$, recover at rate $\boldsymbol{\delta}_3$ or $\boldsymbol{\delta}_4$ and die at rate $\boldsymbol{\tau}_3$ and $\boldsymbol{\tau}_4$ respectively. Figure A.1 outlines the compartmental model defined by Abrams et al. [Abrams et al., 2021].

A.1 Deterministic Compartmental Model

The flows of the deterministic model are defined by a set of ordinary differential equations, which are outlined as follows:

$$
\begin{aligned}
\frac{d\mathbf{S}(t)}{dt} &= -\lambda(t)(\mathbf{S})(t), \\
\frac{d\mathbf{E}(t)}{dt} &= \lambda(t)(\mathbf{S})(t) - \gamma\mathbf{E}(t), \\
\frac{d\mathbf{I}^{presym}(t)}{dt} &= \gamma\mathbf{E}(t) - \theta\mathbf{I}^{presym}(t), \\
\frac{d\mathbf{I}^{asym}(t)}{dt} &= \theta p\mathbf{I}^{presym}(t) - \delta_1\mathbf{I}^{asym}(t), \\
\frac{d\mathbf{I}^{mild}(t)}{dt} &= \theta(1-p)\mathbf{I}^{presym}(t) - \{\psi + \omega_2\}\mathbf{I}^{mild}(t), \\
\frac{d\mathbf{I}^{sev}(t)}{dt} &= \psi\mathbf{I}^{mild}(t) - \omega\mathbf{I}^{sev}(t), \\
\frac{d\mathbf{I}^{hosp}(t)}{dt} &= \phi_1\omega\mathbf{I}^{sev}(t) - \{\delta_3 + \tau_1\}\mathbf{I}^{hosp}(t), \\
\frac{d\mathbf{I}^{icu}(t)}{dt} &= (1-\phi_1)\omega\mathbf{I}^{sev}(t) - \{\delta_4 + \tau_2\}\mathbf{I}^{icu}(t), \\
\frac{d\mathbf{D}(t)}{dt} &= \tau_1\mathbf{I}^{hosp}(t) - \tau_2\mathbf{I}^{icu}(t), \\
\frac{d\mathbf{R}(t)}{dt} &= \delta_1\mathbf{I}^{asym}(t) + \delta_2\mathbf{I}^{mild}(t) + \delta_3\mathbf{I}^{hosp}(t) + \delta_4\mathbf{I}^{icu}(t)
\end{aligned}
$$

where, for example, $\mathbf{S} = (S_1(t), S_2(t), ..., S_k(t))^T$ is the vector representing the susceptible members of the population of each age group k at time t.

In this set of ordinary differential equations, each state variable represents a vector over all age groups for a particular compartment at time t. Infection dynamics are governed by an age-specific force of infection λ:

$$\lambda(k,t) = \sum_{k'=1}^{K} \beta(k,k')I_{k'}(t), \tag{A.1}$$

where k is the respective age group of a total of K age groups, and $\beta(k,k')$ is the time-invariant transmission rate that encodes the average per capita rate at which an infectious individual in age group k makes an effective contact with a susceptible individual in age group k', per unit of time.

Under the social contact hypothesis [Wallinga et al., 2006], we have that:

$$\beta(k,k') = q \cdot C(k,k'), \tag{A.2}$$

where q is a proportionality factor and the matrix C denotes the social mixing behaviour within and between different age groups in the population, and is referred to as a social contact matrix. Following Abrams et al. [2021], we rely on distinct social contact matrices for symptomatic and asymptomatic individuals, respectively C_s and C_a. Therefore it is possible to define the transmission rates for both symptomatic and asymptomatic individuals as follows:

$$\boldsymbol{\beta}_s(k,k') = q_s \cdot C_s(k,k'), \tag{A.3}$$

and

$$\boldsymbol{\beta}_a(k,k') = q_a \cdot C_a(k,k'). \tag{A.4}$$

Moreover, the age-dependent force of infection can be defined as follows:

$$\boldsymbol{\lambda}(t) = \boldsymbol{\beta}_a \times \{\mathbf{I}^{presym}(t) + \mathbf{I}^{asym}(t)\} + \boldsymbol{\beta}_s \times \{\mathbf{I}^{mild}(t) + \mathbf{I}^{sev}(t)\}, \tag{A.5}$$

where $\boldsymbol{\lambda}(t) = (\lambda(1,t), \lambda(1,t), ..., \lambda(K,t))$. For all further information about the different compartments and parameters please refer to the work of Abrams et al. [2021]

To create a version of MOBelCov with a deterministic transition function, M, we focus on the deterministic model proposed by Abrams et al. [2021]. Given that the transitions within the compartmental model are deterministic it is possible to utilise the highlighted model transitions for MOBelCov. Applying the contact matrix $\hat{C}$ to the model state s_m progresses the model for one timestep and returns a new compartmental model state s'_m. Given s_m and $\hat{C}$, the deterministic compartmental model returns s'_m with probability of 1. Therefore, it is possible to use this process as a deterministic transition function, M, for MOBelCov.

A.2 Stochastic Compartmental Model

Intervening in the spread of the virus by, for example, reducing social contacts or government interventions introduces uncertainty in the further course of the outbreak. Therefore, to understand how this uncertainty affect the spread of the disease we introduce a stochastic component model which can model the uncertainty generated by interventions in social contacts.

By formulating the set of differential equations defined above, as a chain-binomial, we can obtain stochastic trajectories from this model [Bailey, 1975]. A chain-binomal model assumes a stochastic model where infected individuals are generated by some underlying probability distribution. For the stochastic model we consider a time interval $(t, t+h]$, where h is defined as the length between two consecutive time points. Similar to Abrams et al. [2021], in this work we set $h = \frac{1}{24}$. Abrams et al. [2021] define the set of differential equations as a chain binomial as follows:

$$\begin{aligned}
\mathbf{S}_{t+h}(k) &= \mathbf{S}_t(k) - \mathbf{E}_{new,t+h}(k),\\
\mathbf{E}_{t+h}(k) &= \mathbf{E}_t(k) + \mathbf{E}_{new,t+h}(k) - \mathbf{I}^{presym}_{new,t+h}(k),\\
\mathbf{I}^{presym}_{t+h}(k) &= \mathbf{I}^{presym}_t(k) + \mathbf{I}^{presym}_{new,t+h}(k) - \mathbf{I}^{asym}_{new,t+h}(k) - \mathbf{I}^{mild}_{new,t+h}(k),\\
\mathbf{I}^{asym}_{t+h}(k) &= \mathbf{I}^{asym}_t(k) + \mathbf{I}^{asym}_{new,t+h}(k) - \mathbf{R}^{asym}_{new,t+h}(k),\\
\mathbf{I}^{mild}_{t+h}(k) &= \mathbf{I}^{mild}_t(k) + \mathbf{I}^{mild}_{new,t+h}(k) - \mathbf{I}^{sev}_{new,t+h}(k) - \mathbf{R}^{mild}_{new,t+h}(k),\\
\mathbf{I}^{sev}_{t+h}(k) &= \mathbf{I}^{sev}_t(k) + \mathbf{I}^{sev}_{new,t+h}(k) - \mathbf{I}^{hosp}_{new,t+h}(k) - \mathbf{I}^{icu}_{new,t+h}(k),\\
\mathbf{I}^{hosp}_{t+h}(k) &= \mathbf{I}^{hosp}_t(k) + \mathbf{I}^{hosp}_{new,t+h}(k) - \mathbf{D}^{hosp}_{new,t+h}(k) - \mathbf{R}^{hosp}_{new,t+h}(k),\\
\mathbf{I}^{icu}_{t+h}(k) &= \mathbf{I}^{icu}_t(k) + \mathbf{I}^{icu}_{new,t+h}(k) - \mathbf{D}^{icu}_{new,t+h}(k) - \mathbf{R}^{icu}_{new,t+h}(k),\\
\mathbf{D}_{t+h}(k) &= \mathbf{D}_t(k) + \mathbf{D}^{hosp}_{new,t+h}(k) + \mathbf{D}^{icu}_{new,t+h}(k),\\
\mathbf{R}_{t+h}(k) &= \mathbf{R}_t(k) + \mathbf{R}^{asym}_{new,t+h}(k) + \mathbf{R}^{mild}_{new,t+h}(k) + \mathbf{R}^{hosp}_{new,t+h}(k) + \mathbf{R}^{icu}_{new,t+h}(k)
\end{aligned}$$

where,

$$
\begin{aligned}
\mathbf{E}_{new,t+h} &\sim \textit{Binomial}\left(\mathbf{S}_t(k), p_t^*(k) = 1 - \{1 - p_t^*(k)\}^{\mathbf{I}_t}\right), \\
p_t^*(k) &= 1 - exp\left[-h \sum_{k'=1}^{K} \beta_{asym}(k,k')\{\mathbf{I}_t^{asym}(k')\} + \beta_{sym}(k,k')\{\mathbf{I}_t^{mild}(k') + \mathbf{I}_t^{sev}(k')\}\right], \\
\mathbf{I}_{new,t+h}^{presym}(k) &\sim Binomial\left(\mathbf{I}_t^{presym}(k), 1 - exp(-hp(k)\theta)\right), \\
\mathbf{I}_{new,t+h}^{mild}(k) &\sim Binomial\left(\mathbf{I}_t^{presym}(k), 1 - exp\left[-h\{1-p(k)\}\theta\right]\right), \\
\mathbf{I}_{new,t+h}^{sev}(k) &\sim Binomial\left(\mathbf{I}_t^{mild}(k), 1 - exp\{-h\psi(k)\}\right), \\
\mathbf{I}_{new,t+h}^{hosp}(k) &\sim Binomial\left(\mathbf{I}_t^{sev}(k), 1 - exp\{-h\phi_1(k)\omega(k)\}\right), \\
\mathbf{I}_{new,t+h}^{icu}(k) &\sim Binomial\left(\mathbf{I}_t^{sev}(k), 1 - exp\left[-h\{1-\phi_1(k)\}\omega(k)\right]\right), \\
\mathbf{D}_{new,t+h}^{hosp}(k) &\sim Binomial\left(\mathbf{I}_t^{hosp}(k), 1 - exp\{-h\tau_1(k)\}\right), \\
\mathbf{D}_{new,t+h}^{icu}(k) &\sim Binomial\left(\mathbf{I}_t^{icu}(k), 1 - exp\{-h\tau_2(k)\}\right), \\
\mathbf{R}_{new,t+h}^{asym}(k) &\sim Binomial\left(\mathbf{I}_t^{asym}(k), 1 - exp\left(-h\delta_2(k)\right)\right), \\
\mathbf{R}_{new,t+h}^{hosp}(k) &\sim Binomial\left(\mathbf{I}_t^{hosp}(k), 1 - exp\{-h\delta_3(k)\}\right), \\
\mathbf{R}_{new,t+h}^{icu}(k) &\sim Binomial\left(\mathbf{I}_t^{icu}(k), 1 - exp\{-h\delta_4(k)\}\right).
\end{aligned}
$$

Given MOBelCov also calculates new hospitalisations, $\mathbf{H}^{new}$, we define $\mathbf{H}^{new}$ for the stochastic compartmental model as follows:

$$\mathbf{H}_{t+h}^{new}(k) = \mathbf{H}_t^{new}(k) + \mathbf{I}_{new,t+h}^{hosp}(k).$$

For more details on this model and the chain-binomial representation of the differential equations, we refer the reader to the work of Abrams et al. [2021].

To create a version of MOBelCov with a stochastic transition function, M, we utilise the stochastic compartmental model outlined above. Given the transitions within the compartmental model are derived by an underlying probability distribution it is possible to utilise the stochastic compartmental model transitions for MOBelCov. As previously outlined in Section A.1 the contact matrix $\hat{C}$ applied the model state s_m progresses the model and returns a new model state s'_m. Given the underlying model dynamics are governed in a probabilistic manner, the model returns s'_m stochastically. Therefore, it is possible to use this process as a stochastic transition function, M, for MOBelCov.

A.3 A Note on Model Parameters

The model is parameterised using the mean of the posteriors as reported by Abrams et al. [2021].

The population size for each of the considered age groups was taken from the Belgian statistical agency STATBEL[3]. To initialise the model, we used the number of confirmed cases until 13 March 2020 [Abrams et al., 2021], as reported by the Belgian agency for public health Sciensano[4].

A.4 Modelling interventions

In order to model different types of interventions, we follow Abrams et al. [2021]. Firstly, to consider distinct exit scenarios, we alter the social contact matrices to reflect a contact reduction in a particular age group. Secondly, we assume that compliance to the interventions is gradual and model this using a logistic compliance function. We use the logistic compliance function in function of time t proposed by Abrams et al.,

$$c(t, t_I) = \frac{\exp(\beta_0^* + \beta_1^*(t - t_I))}{1 + \exp(\beta_0^* + \beta_1^*(t - t_I))}, \tag{A.6}$$

where t_I indicates the time the intervention started [Abrams et al., 2021]. We initialise β_1^* to the value estimated in by Abrams et al. and choose $\beta_0^* = -5$, as an intercept to have $c(t) = 0$ for $t = 0$, in correspondence with Figure F2 in the Supplementary Information of Abrams et al. [2021].

[3] `https://statbel.fgov.be/nl/themas/bevolking/structuur-van-de-bevolking#figures`

[4] `https://epistat.wiv-isp.be/covid/`

B Additional results

B.1 Learned coverage sets

In Figure B.3, we display the coverage set with respect to the number of hospitalizations of the learned policies for both the ODE and Binomial variants, as the Binomial variants were omitted in the main paper for clarity. When trained on $\mathbf{R}_{\text{ARH}}$, the Binomial coverage set is on par with its ODE counterpart. On the other hand, the coverage sets when trained on $\mathbf{R}_{\text{ARI}}$ are slightly worse, as confirmed by the evaluation metrics in Table 1.

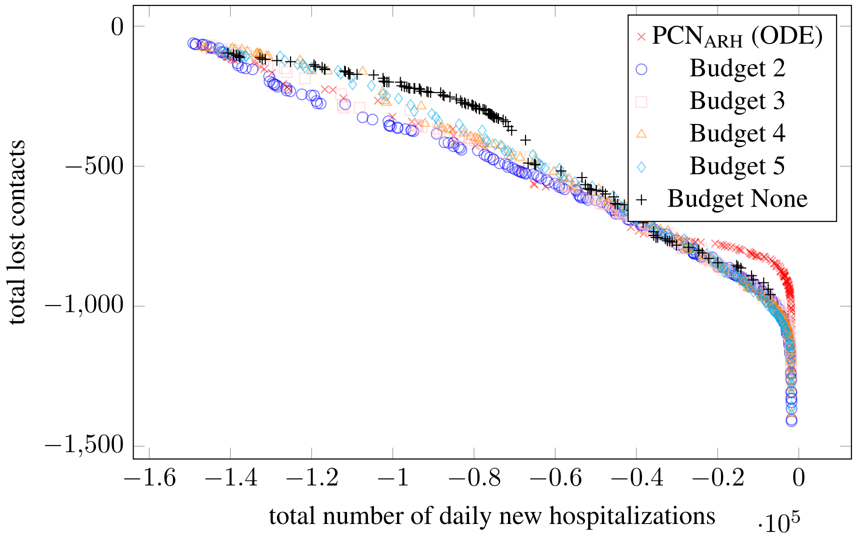


Figure B.1: The Pareto front of policies discovered by PCN, showing the different compromises between the number of hospitalizations and the number of lost contacts. Although PCN_{ARI} was trained on the number of infections, it still shows a competitive coverage set w.r.t. hospitalizations.

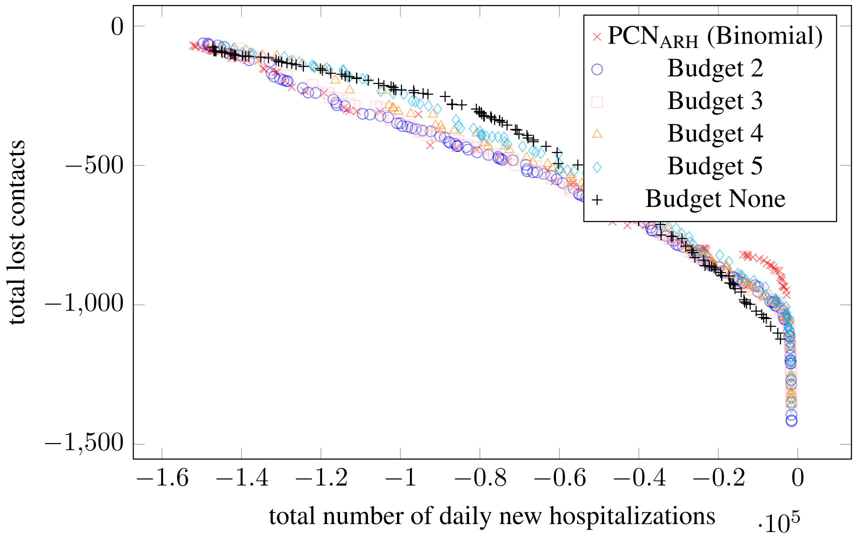


Figure B.2: The Pareto front of policies discovered by PCN, showing the different compromises between the number of hospitalizations and the number of lost contacts. Although PCN_{ARI} was trained on the number of infections, it still shows a competitive coverage set w.r.t. hospitalizations.

Moreover, we also show the coverage sets with respect to the number of infections in Figure B.4. Interestingly, both variants trained on the Binomial model learn a coverage set that almost completely dominates the variants trained on the ODE model. This is due to the stochasticity of the Binomial model. PCN stores trajectories in its dataset, and keep a selection for training. Due to the stochasticity in the transition function, some trajectories might result in higher

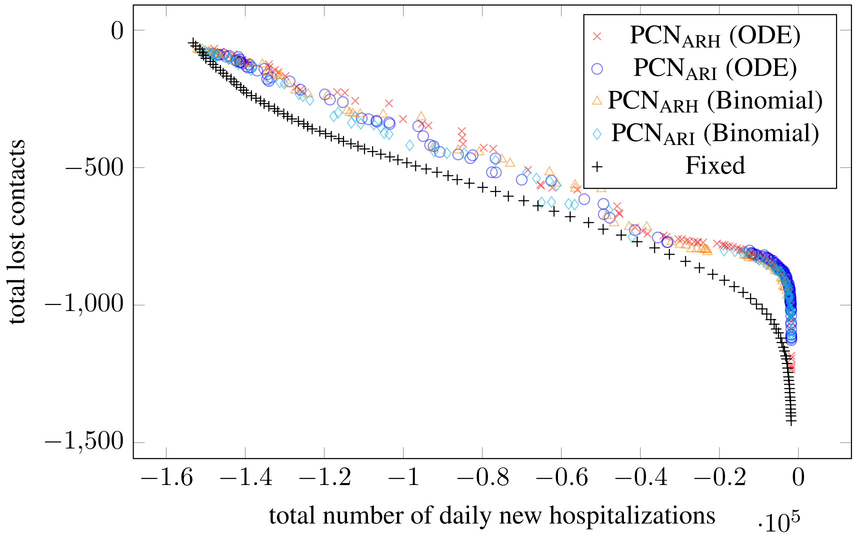
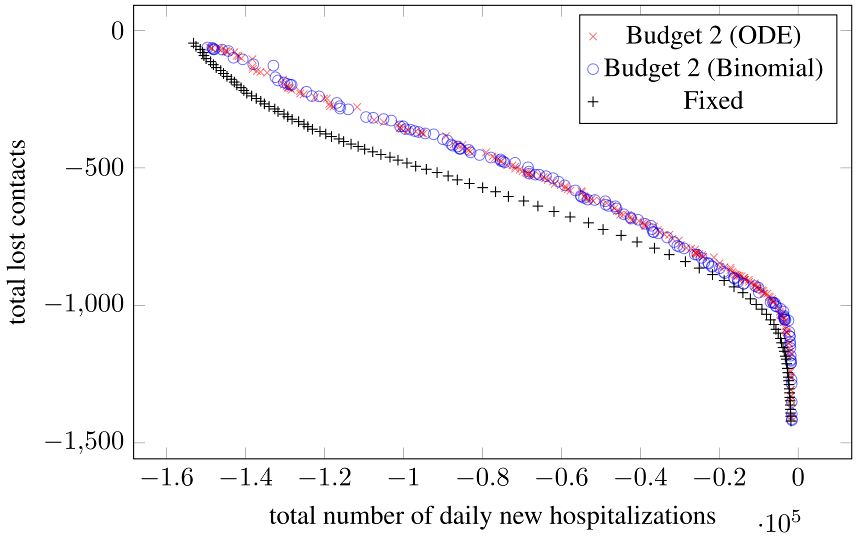


Figure B.3: The Pareto front of policies discovered by PCN, showing the different compromises between the number of hospitalizations and the number of lost contacts. Although PCN$_{\text{ARI}}$ was trained on the number of infections, it still shows a competitive coverage set w.r.t. hospitalizations.

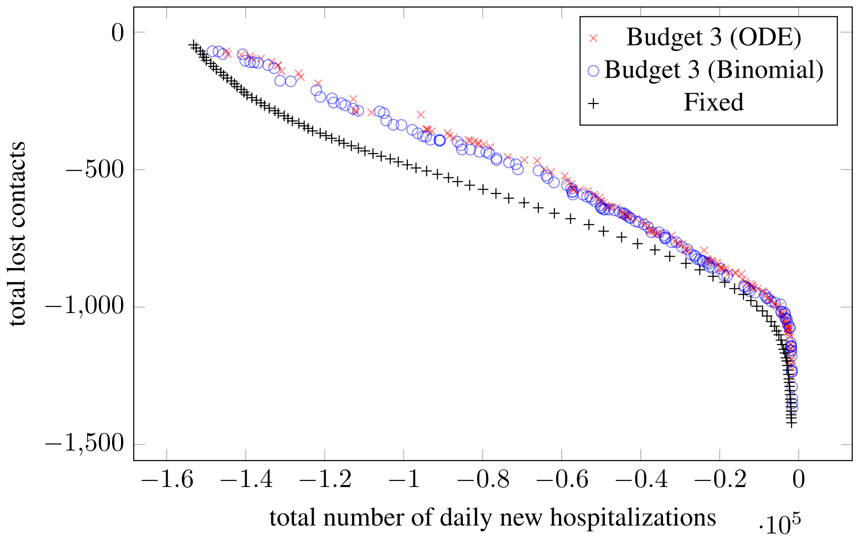


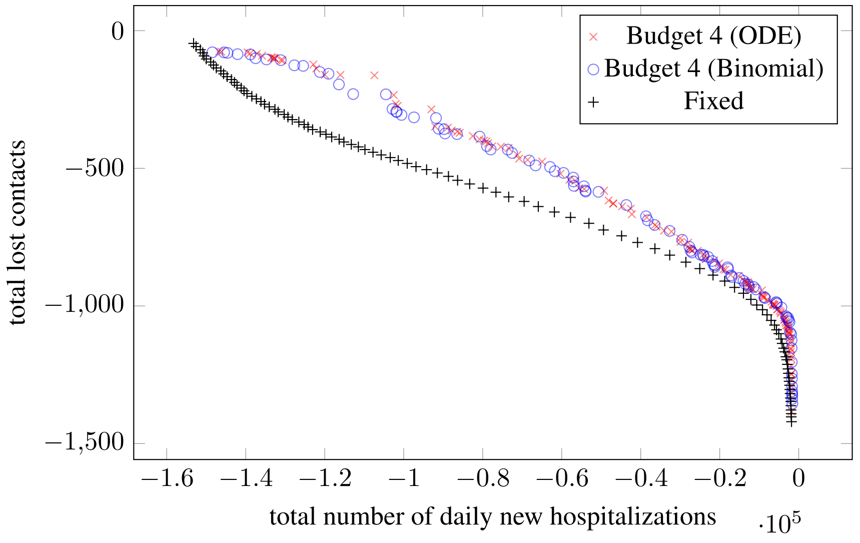
returns, even though the same policy was applied. Due to the high variability in infections, this difference might become significant. This is also why the variants using the Binomial model show a worse I_ε and $I_\varepsilon - mean$ than their ODE counterpart: their desired return stems from a single trajectory, while we evaluate the policies over multiple trajectories. The resulting average return is then different than the desired return. Section 5.3 explains why some policies trained on $\mathbf{R}_{\text{ARH}}$ show suboptimal behavior with respect to the number of infections.

B.2 Policy executions

PCN learns a coverage set containing more than 100 different policies. To gain a better insight about their behavior, and how they differ from each other, we plot executions of each learned policy in Figure B.5. The plots are displayed from the least restrictive policy in terms of social burden to the most restrictive one.

We notice that policies 60 to 140 display similar behavior. At first, they all completely restrict social contact, effectively continuing the lockdown. Afterwards, they progressively lessen these restrictions. What differs in these policies is the timing and the speed at which the restrictions are lessened. These policies correspond to the right-most part of the coverage set displayed in Figure B.3, which shows many alternative for the $[0, 40000]$ hospitalizations segment.





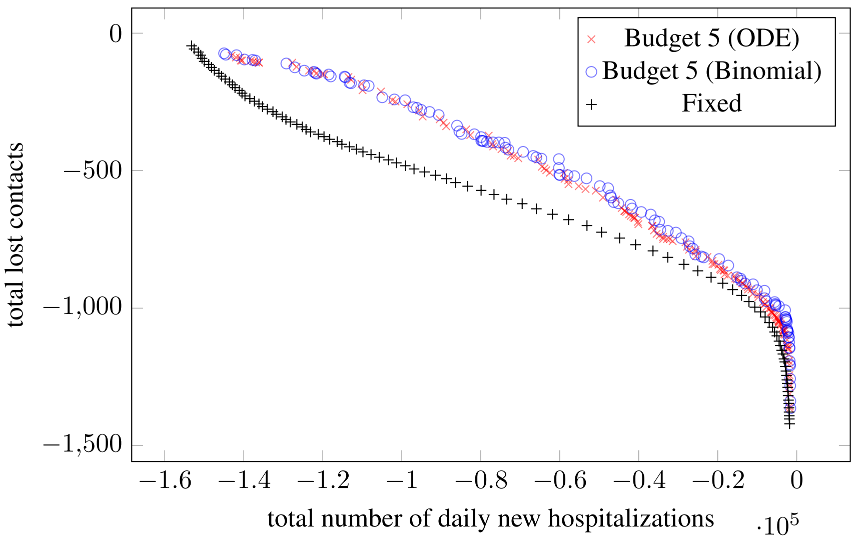
B.3 Experiment parameters

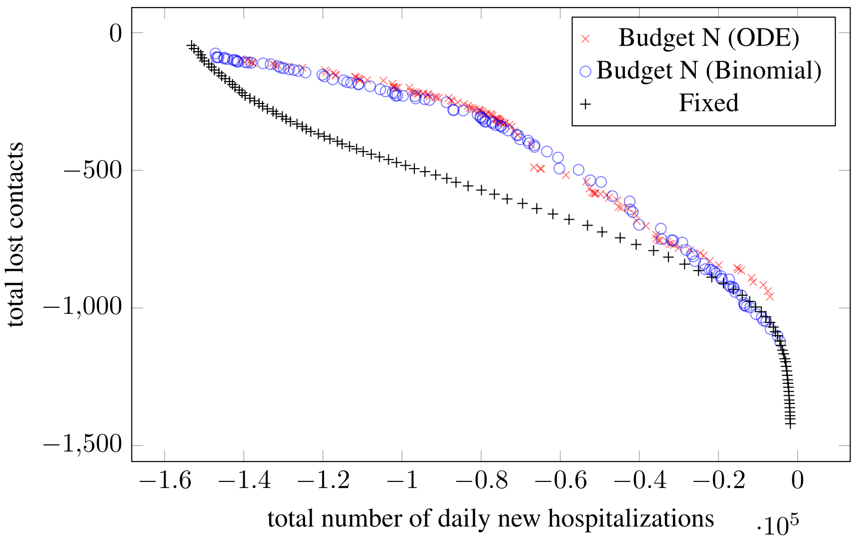
We used the same hyper-parameters across all experiments. Each experiment resulted in 5 independent trials. Finally, we performed a grid-search over possible hyper-parameter values. All hyper-parameters used and their possible values explored during grid-search are displayed in Table B.1.

B.4 Neural network architecture

Next to the hyper-parameter search, we also performed a grid search over 4 different neural network architectures. All the architectures have the same structure. We use a compartment embedding sc_{emb}, a SCM embedding sm_{emb} and a school-holidays embedding sh_{emb} that take as inputs the compartment, the previous p_w, p_s, p_l values (as they fully define the SCM $\hat{C}$) and a boolean flag for school holidays, respectively. All these embeddings have a same-sized embedding of 64, which are multiplied together to form the full state embedding. This state-embedding is used as input for another network, s_{emb}. Additionally, we use a common embedding c_{emb} for the concatenation of the desired return and horizon. Finally, the results of s_{emb} and c_{emb} are multiplied together, before passing through a fully connected network fc that has 3 outputs, one for p_w, p_s, p_l respectively.

All the architectures of the different components are displayed in Table B.2. The variant used in all experiments is `conv1d-big`.





C Related Work

RL has also been used to learn effective mitigation strategies which limit the spread of COVID-19 [Ohi et al., 2020]. Kwak et al. [Kwak et al., 2021] use deep RL, with an SIRD model, to learn efficacy of lockdown and travel restrictions in controlling the COVID-19 pandemic using data collected from various government organisations. Bastani et al. [Bastani et al., 2021] use a RL algorithm called Eva to limit the influx of asymptomatic travellers infected with SARS-CoV-2 to Greece. Kompella et al. [Kompella et al., 2020] define a COVID-19 simulator and a methodology to learn mitigation policies that minimize the economic impact without overwhelming the hospital capacity. RL has also been used to learn effective COVID-19 vaccination distribution strategies [Beigi et al., 2021, Awasthi et al., 2020] and to create decision support systems for mitigating the spread of COVID-19.

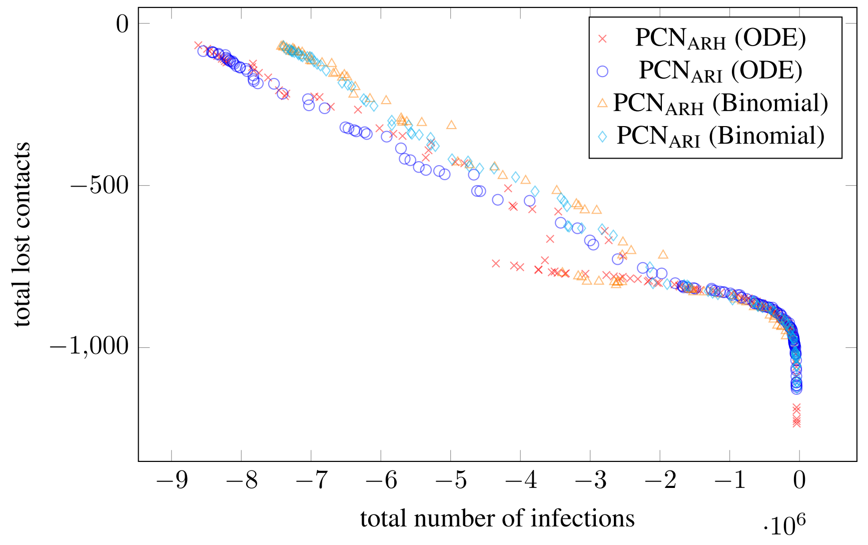


Figure B.4: The Pareto front of policies discovered by PCN, showing the different compromises between the number of hospitalizations and the number of lost contacts. Although PCN_{ARI} was trained on the number of infections, it still shows a competitive coverage set w.r.t. hospitalizations.

hyper-parameter	value	grid-search
learning rate	0.001	
total training timesteps	300000	
batch size	256	$256, 1024$
model updates	50	
episodes between updates	10	
ER size (in episodes)	1000	$400, 500, 1000$
initial random episodes	200	$50, 200$
exploration noise	0.1	$0, 0.1, 0.2$
desired return noise	0.2	$0, 0.1, 0.2$
reward scaling	$[10000, 100]$	

Table B.1: The different hyper-parameters used by our extension of PCN. The right-most column also shows, when applicable, the different values tried during grid-search.

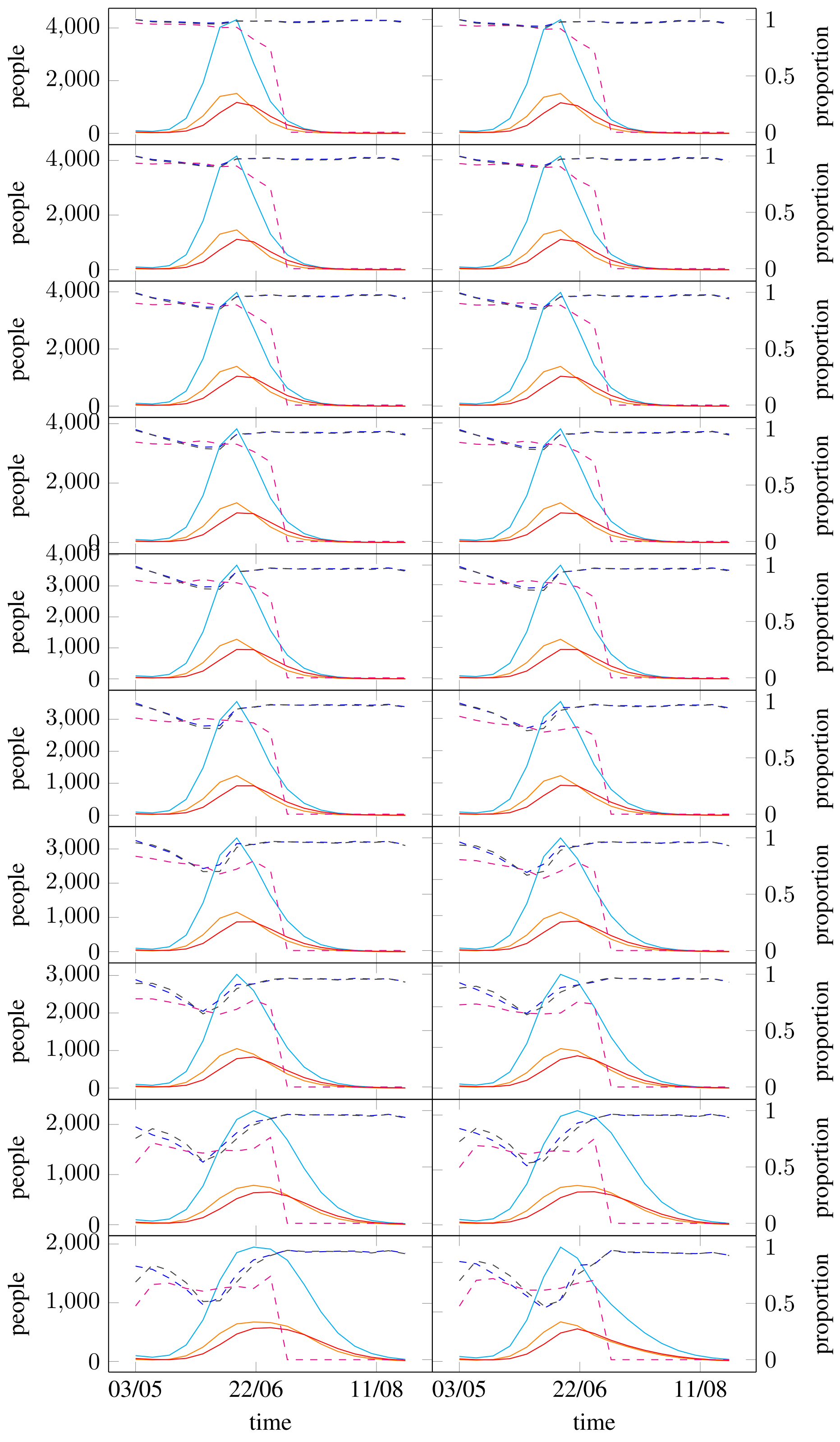


Figure B.5: Execution of policies 1 to 20.

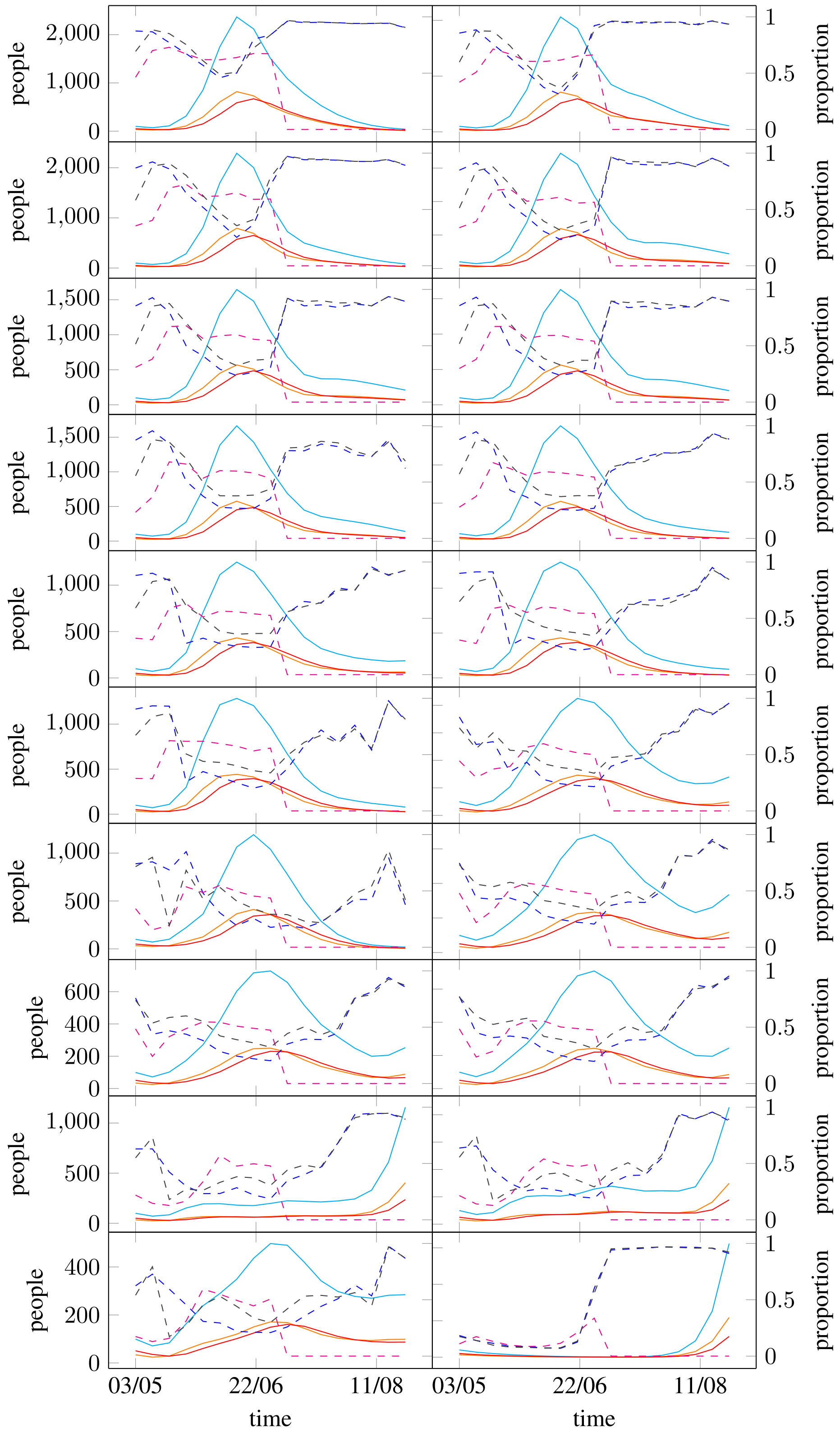


Figure B.5: Execution of policies 21 to 40.

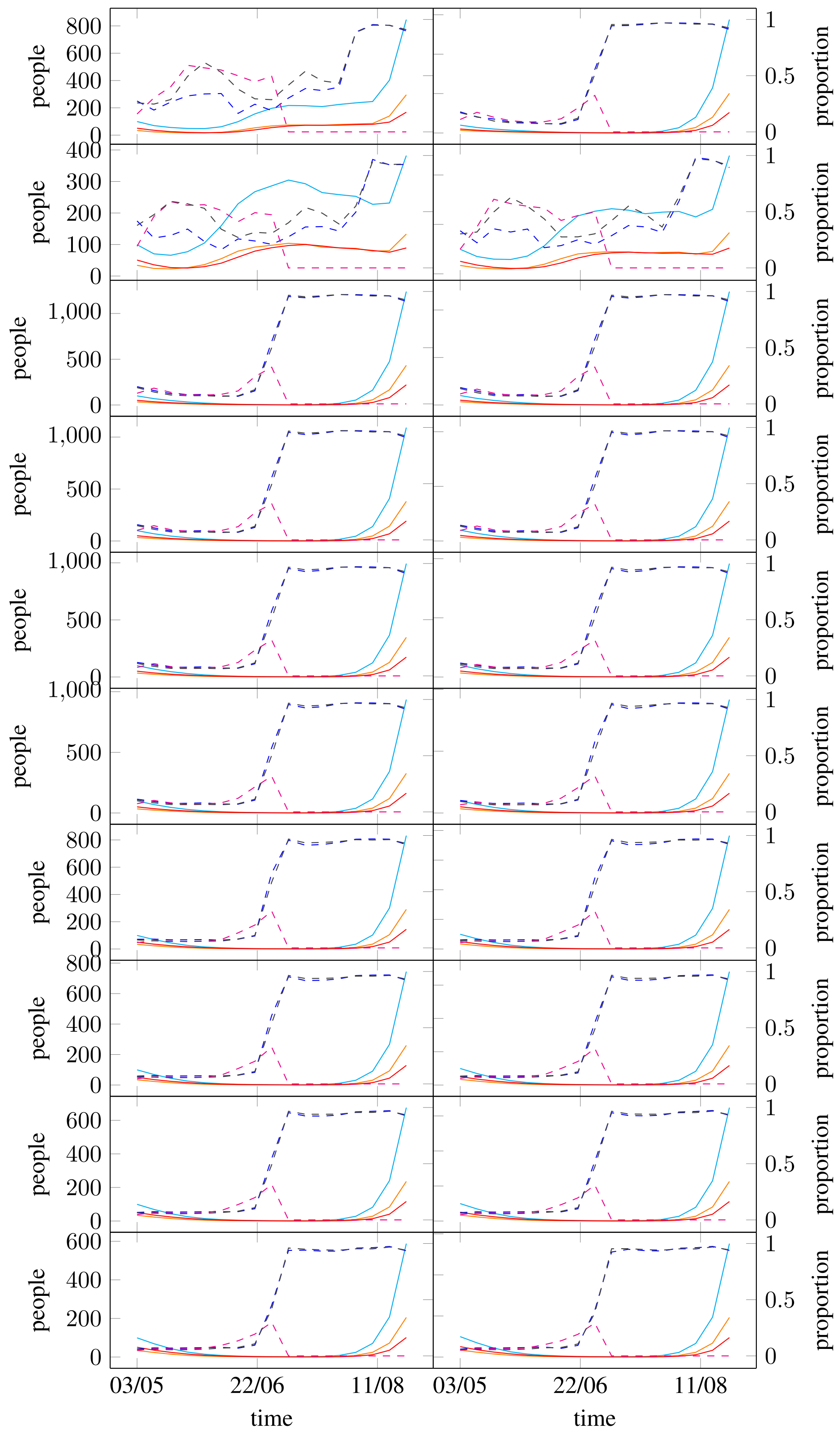


Figure B.5: Execution of policies 41 to 60.

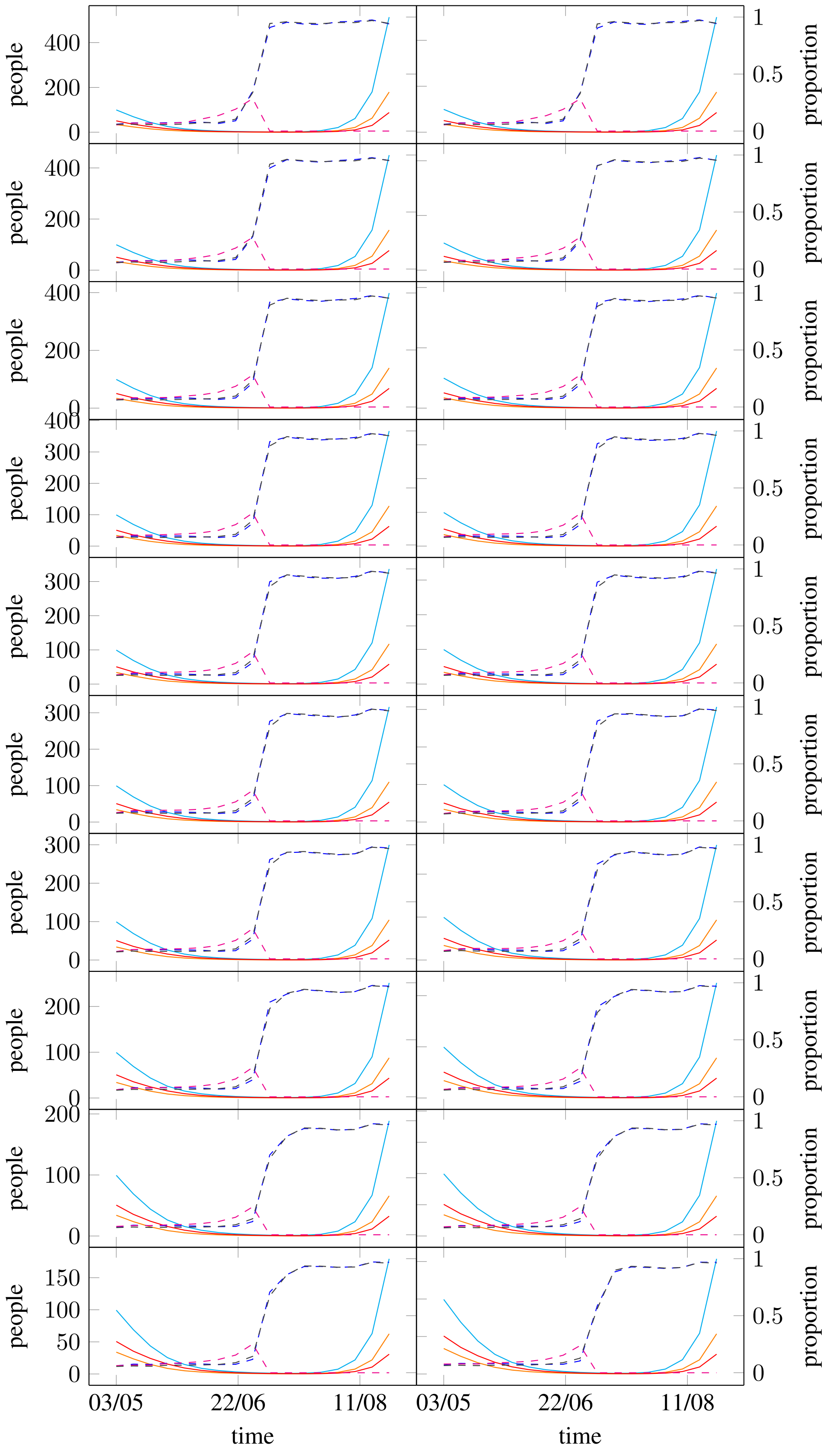


Figure B.5: Execution of policies 61 to 80.

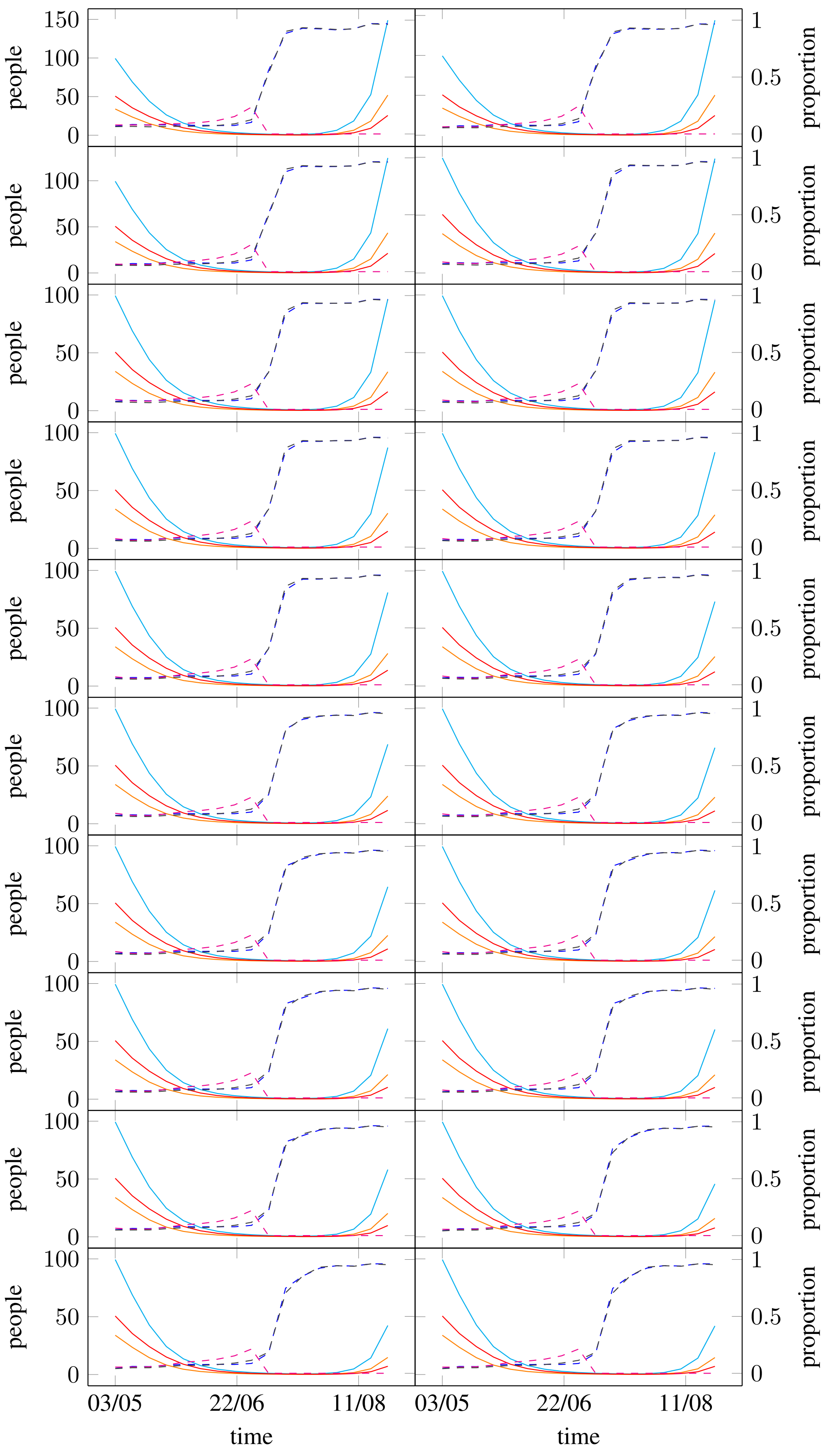


Figure B.5: Execution of policies 81 to 100.

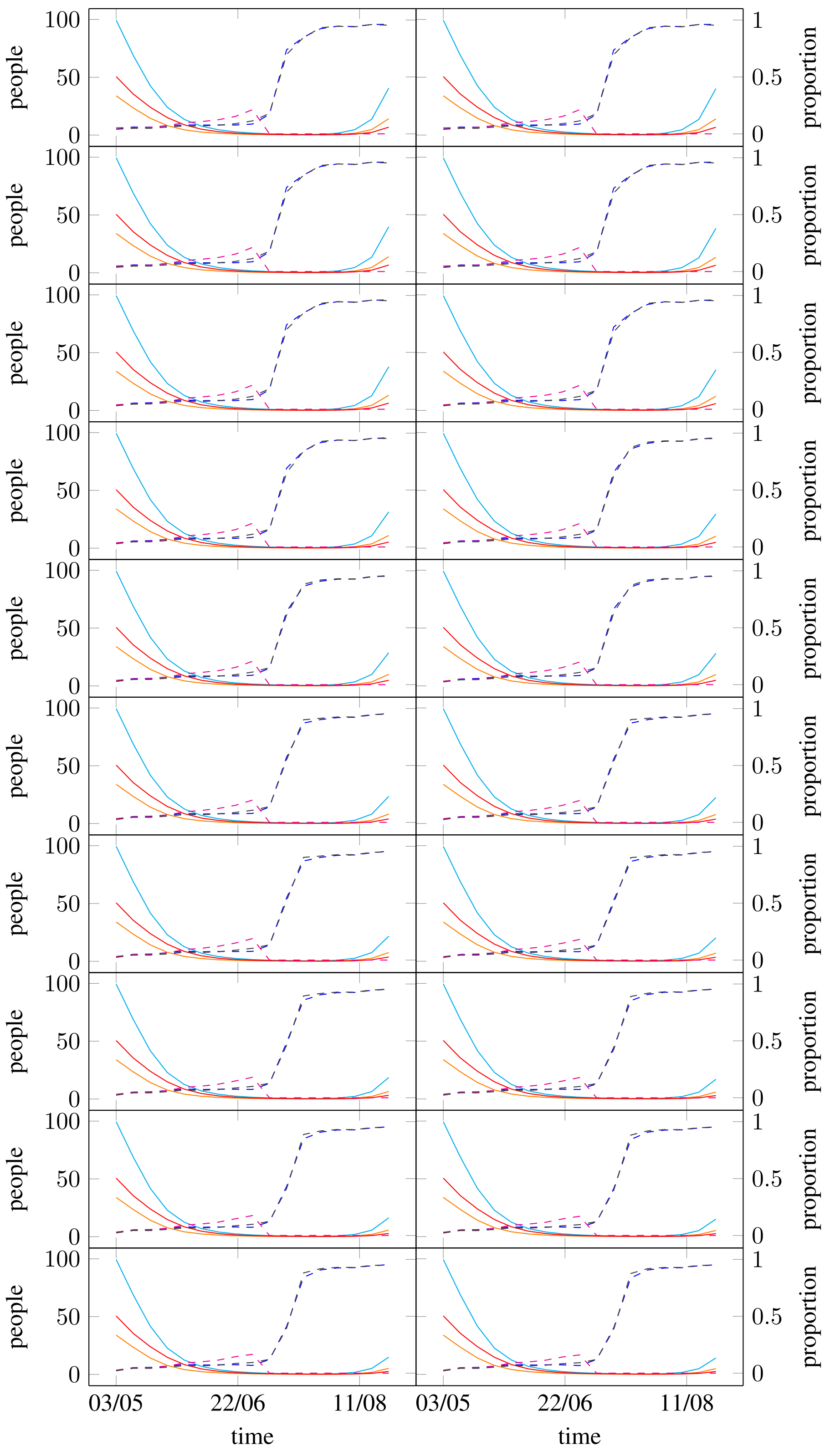


Figure B.5: Execution of policies 101 to 120.

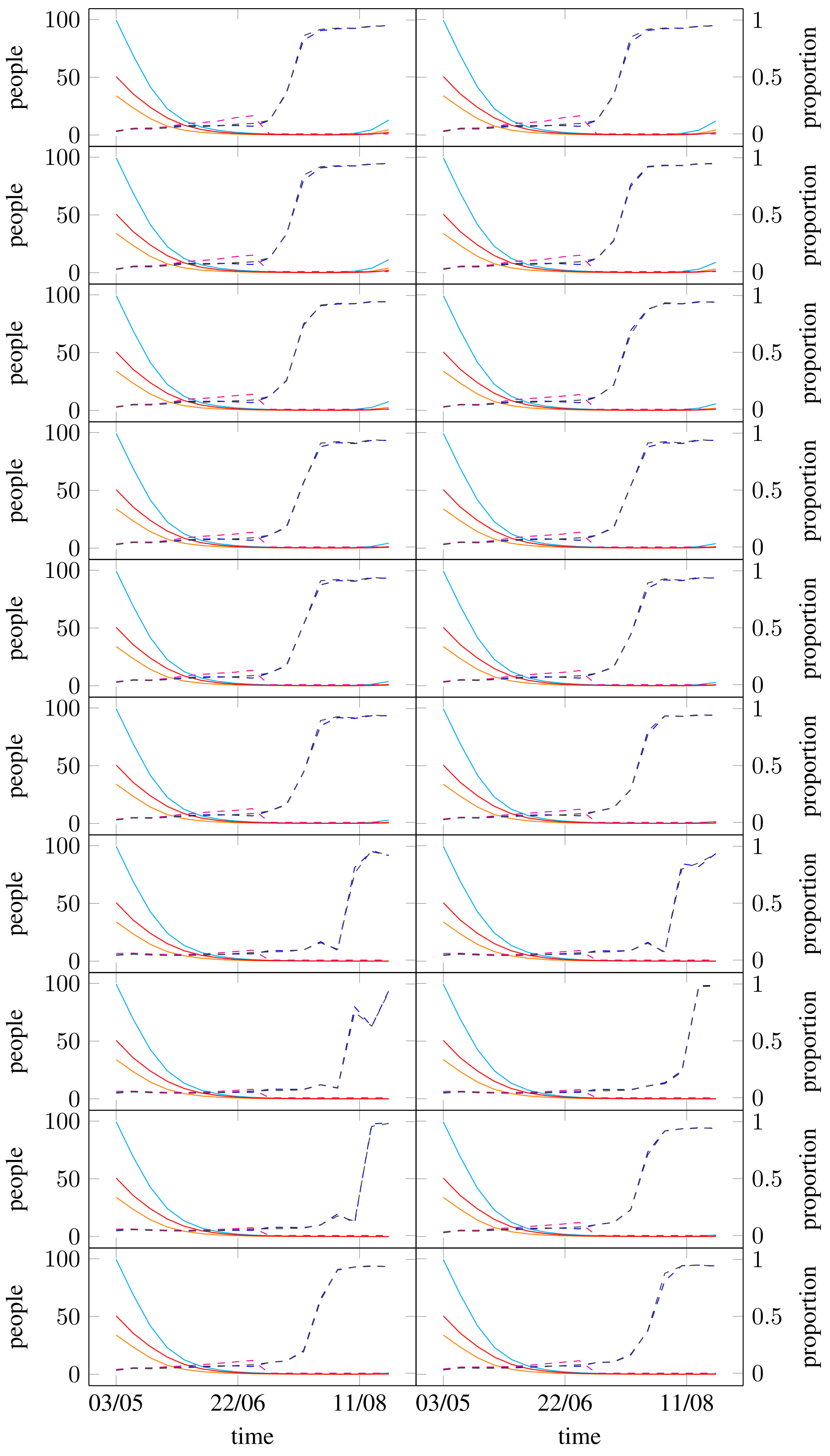


Figure B.5: Execution of policies 121 to 140.

variant	sc_{emb}	sm_{emb}	sh_{emb}	s_{emb}	c_{emb}	fc

conv1d-small	conv1d(10,20)
	relu
	conv1d(20,20)
	relu
	linear(100,64)
	sigmoid
		linear(3,64)
		sigmoid
			linear(1,64)
			sigmoid
				linear(64,64)
				sigmoid
					linear(3,64)
					sigmoid
						linear(64,64)
						relu
						linear(64,3)

conv1d-big	conv1d(10,20)
	relu
	conv1d(20,20)
	relu
	linear(100,64)
	sigmoid
		linear(3,64)
		relu
		linear(64,64)
		sigmoid
			linear(1,64)
			relu
			linear(64,64)
			sigmoid
				linear(64,64)
				relu
					linear(3,64)
					sigmoid
						linear(64,64)
						relu
						linear(64,3)

dense-small	linear(130,64)
	sigmoid
		linear(3,64)
		sigmoid
			linear(1,64)
			sigmoid
				linear(64,64)
				sigmoid
					linear(3,64)
					sigmoid
						linear(64,64)
						relu
						linear(64,3)

dense-big	linear(130,64)
	relu
	linear(64,64)
	sigmoid
		linear(3,64)
		relu
		linear(64,64)
		sigmoid
			linear(1,64)
			relu
			linear(64,64)
			sigmoid
				linear(64,64)
				relu
					linear(3,64)
					sigmoid
						linear(64,64)
						relu
						linear(64,3)

Table B.2: The 4 different neural network architectures explored for our experiments. All the displayed results use the `conv1d-big` variant.