Mini-project 1: Deep Q-learning for Epidemic Mitigation

Simon Sondén & Tobias Oberdörfer

1 Question 1

Question 1.a) study the behavior of the model when epidemics are unmitigated

Run the epidemic simulation for *one episode* (30 weeks), *without epidemic mitigation* (meaning no action is taken, i.e. all values in the action dictionary are set to `False`) and **produce three plots** (see Figure 1):

1. A plot of variables $s^{[w]}_{\text{total}}, e^{[w]}_{\text{total}}, i^{[w]}_{\text{total}}, r^{[w]}_{\text{total}}, d^{[w]}_{\text{total}}$ over time, where time is measured in weeks and all the variables share the y axis scaling.

2. A plot of variables $i^{[w]}_{\text{total}}, d^{[w]}_{\text{total}}$ over time, where time is measured in weeks and all the variables share the y axis scaling.

3. A set of plots of variables $i^{[w]}_{\text{city}}, d^{[w]}_{\text{city}}$ over time, where time is measured in weeks (one subplot per-city, variables share the y-scaling per-city).

Discuss the evolution of the variables over time.

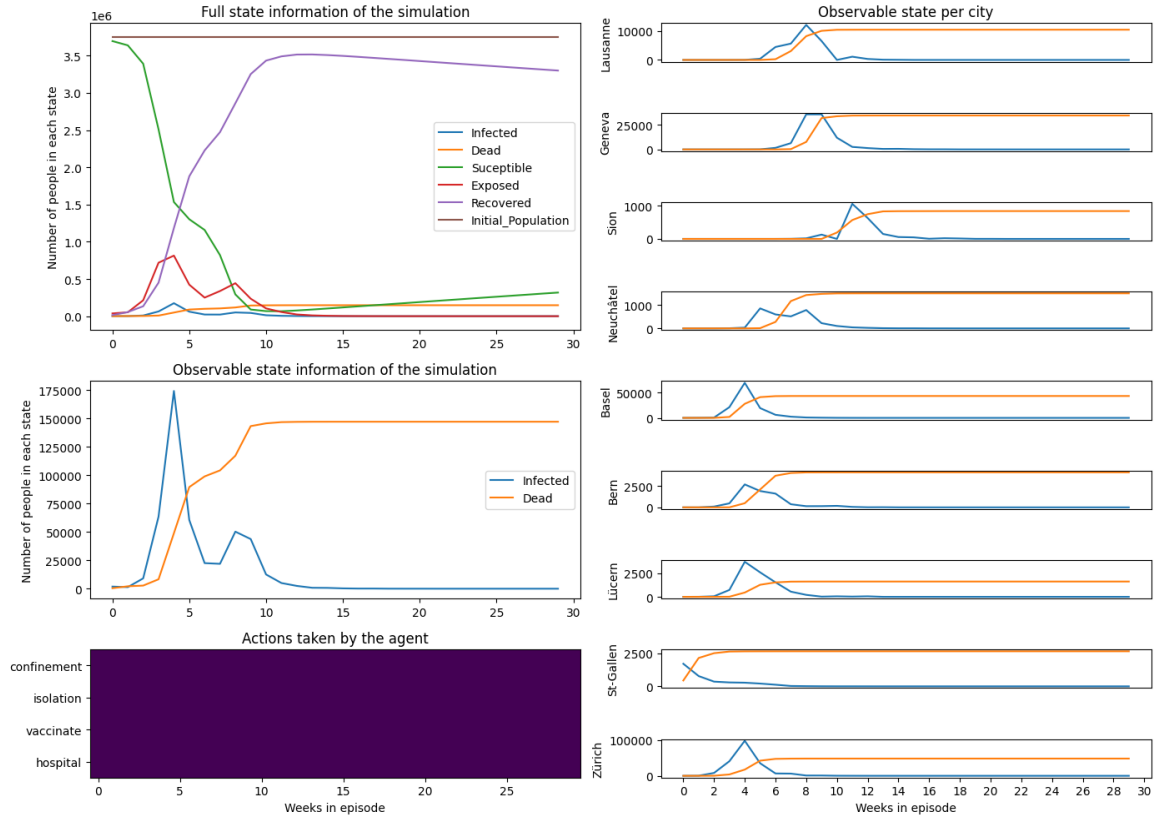


Figure 1: All three subplots of Q1 in one for space efficiency reasons.

1. Over the initial seven weeks, the virus spreads unchecked, leading to a rise in infections and a decline in susceptible individuals. By around week seven or eight, the number of exposed people starts to decrease, while recoveries continue to increase. The graph of the full state information may not clearly depict it, but the number of infections and deaths steadily increases until approximately week five, after which the pool of potential carriers and potential fatalities diminish significantly by week eight.

2. Not implementing preventive measures results in a rapid surge of infections, subsequently causing a sharp increase in deaths, albeit with a delay, as can be seen on the observable state plot. Additionally, the simulation shows that the number of infected individuals (and consequently susceptible individuals, although not seen but only inferred) eventually decreases, leading to a decrease in new infections. Consequently, it could be inferred that within the 30-week simulation, individuals do not revert back to a susceptible state.

3. In a manner resembling the previous plot of the observable state, we observe a rapid rise in infections within each city once the first infected individuals arrive. Similarly, the number of deaths also increases rapidly, reaching near maximum levels within two to three weeks, albeit with a time lag of one to two weeks compared to the onset of infections.

2 Professor Russo's Policy

Question 2.a) Implement Pr. Russo's Policy

Implement Pr. Russo's Policy **as a python class** (*we recommend that you subclass the* `Agent` *abstract class provided with the project files, and as is demonstrated in the tutorial notebook*). Run the epidemic simulation for one episode using Pr. Russo's Policy to pick actions and **produce four plots** (see Figure 2):

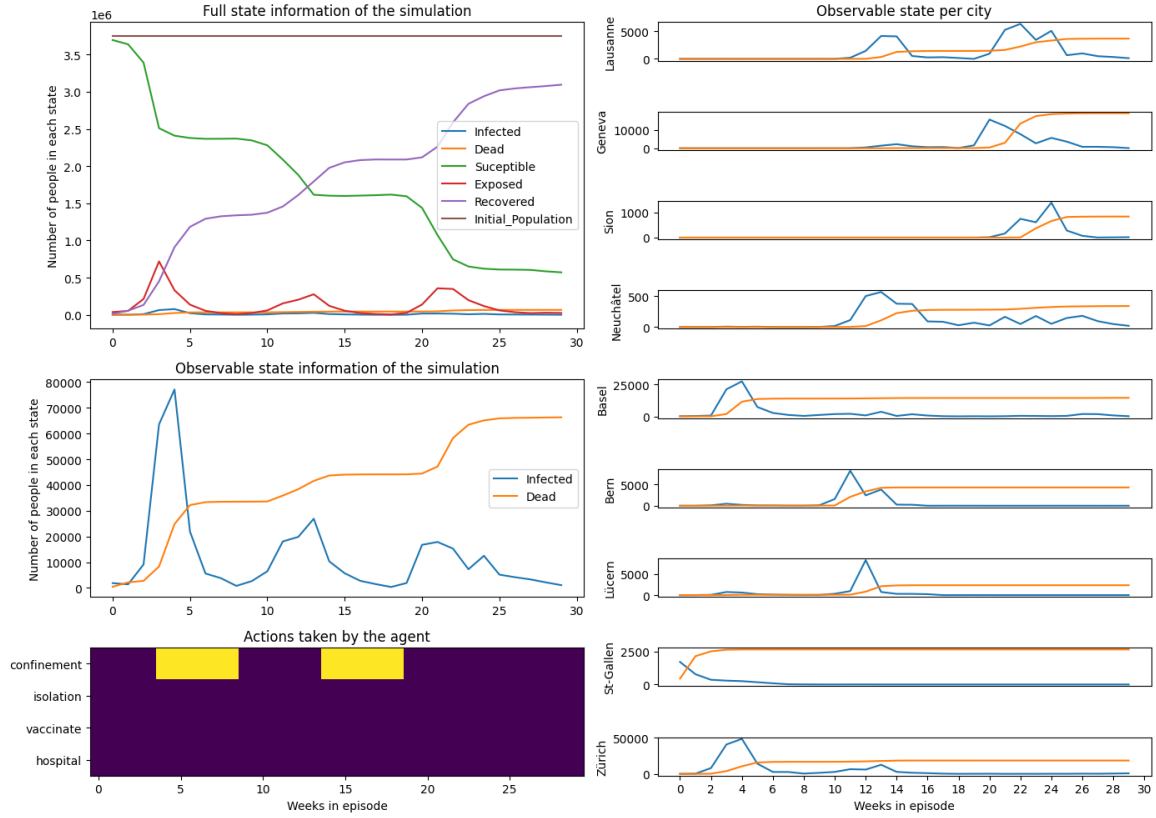


Figure 2: All necessary subplots for Q2.

1. A plot of variables $s^{[w]}_{\text{total}}, e^{[w]}_{\text{total}}, i^{[w]}_{\text{total}}, r^{[w]}_{\text{total}}, d^{[w]}_{\text{total}}$ over time, where time is measured in weeks and all the variables share the y axis scaling.

2. A plot of variables $i^{[w]}_{\text{total}}, d^{[w]}_{\text{total}}$ over time, where time is measured in weeks and all the variables share the y axis scaling.

3. A set of plots of variables $i^{[w]}_{\text{city}}, d^{[w]}_{\text{city}}$ over time, where time is measured in weeks (one subplot per-city, variables share the y-scaling per-city).

4. A plot of the action taken by the policy over time (whether the policy chooses to confine or not).

Discuss how the epidemic simulation responds to Pr. Russo's Policy (focus on how it differs from the unmitigated scenario):

1. We can clearly see from the number of people that are exposed, that confinement reduces the speed at which the virus spreads. This impact can also be seen in both the amount of susceptible and recovered people in the state of the simulation, which both flatten out whenever Russo's policy confines people.

2. Looking at only the observable state we can again clearly see the waves that Russo's policy creates within those 30 weeks. Whenever new confinement is enacted the number of infected people sharply decreases for roughly four weeks, while the deaths do not increase for a similar period but are again delayed by up to two weeks.

3. By examining the observable state within each city, we can observe varying numbers of waves (up to four smaller ones in certain simulation seeds) when no confinement measures are implemented. Furthermore, due to Russo's policy considering the total number of infected individuals, cities like Bern experience a significant increase in infections. However, the implementation of confinement measures in Bern is delayed because the number of infected individuals is not sufficient to trigger them. Only when the threshold of infected individuals is reached through the addition of other cities does Russo's policy respond with confinement measures.

4. When looking at the policy decision we nicely see the times confinement was activated, which resulted in the three waves of infections in the total state graph seen above.

Question 2.b) Evaluate Pr. Russo's Policy

In order to be able to make meaningful conclusions about the behavior of the policy, you will need properly evaluate its behavior. **For each episode, save the following values**: **number of total confined days**, the **cumulative reward** (the sum of all rewards collected during the episode) and the **number of total deaths**. Once those values are logged for each episode, **plot a histogram** (see Figure 3).

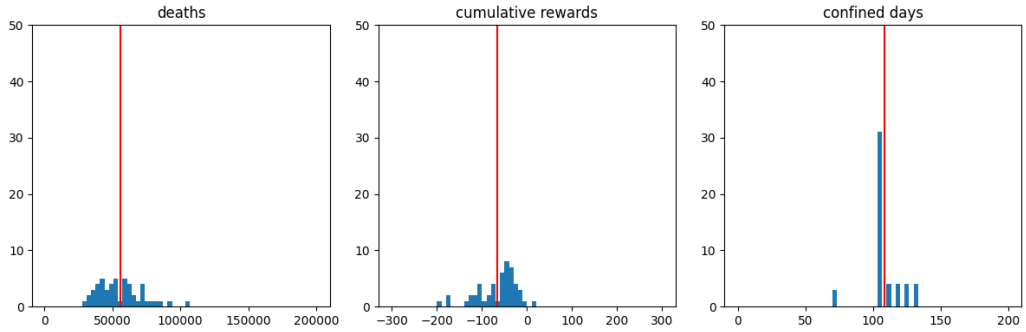


Figure 3: Histograms Q2b)

Average	Deaths	Cumulative reward	Number of confined days
Russo's Policy	55673	-65.67	108

Table 1: Average values of final metrics for the Russo's Policy

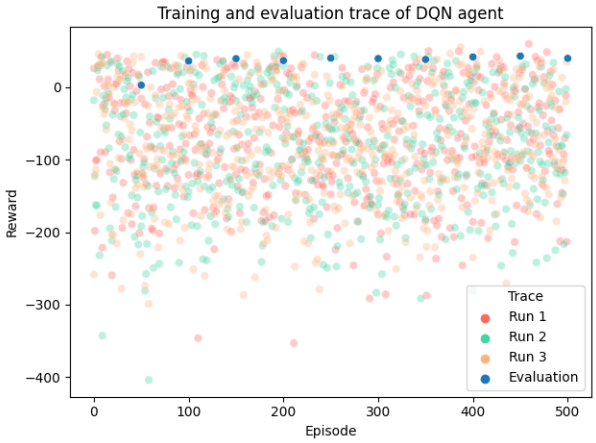
3 Deep Q-Learning with a binary action space

Question 3.a) implementing Deep Q-Learning

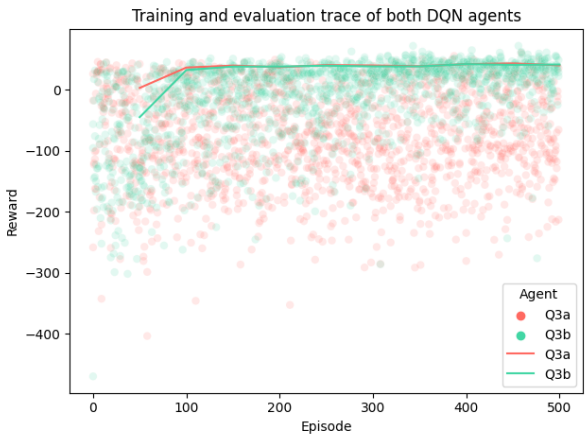
Implement and train the Deep Q-Learning agent $\pi_{\mathbf{DQN}}$ for 500 training episodes, with $\epsilon = 0.7$. For each episode, log the cumulative reward. Once you have successfully evaluated your implementation, plot the training trace and the eval trace. We ask you to average the eval trace across three full training processes.
The plot for the simple DQN with $\epsilon = 0.7$ can be seen in Fig 4a.

Does your agent learn a meaningful policy?
The agent demonstrates policy learning, evident from the notable increase in the evaluation trace. This learning process occurs swiftly, approaching the optimal policy achieved within the initial 300 episodes on average across multiple seeds.

Record three example episodes where actions are picked by $\pi^*_{\mathbf{DQN}}$ you obtained. Plot one of those episodes, using the same plotting procedure as in question *2.a)* and interpret the policy.



(a) Q3a traces; simple DQN $\epsilon = 0.7$



(b) Training & eval traces for 3a and 3b plotted together.

Figure 4: Training and evaluation traces averaged over three runs simple DQN agent.

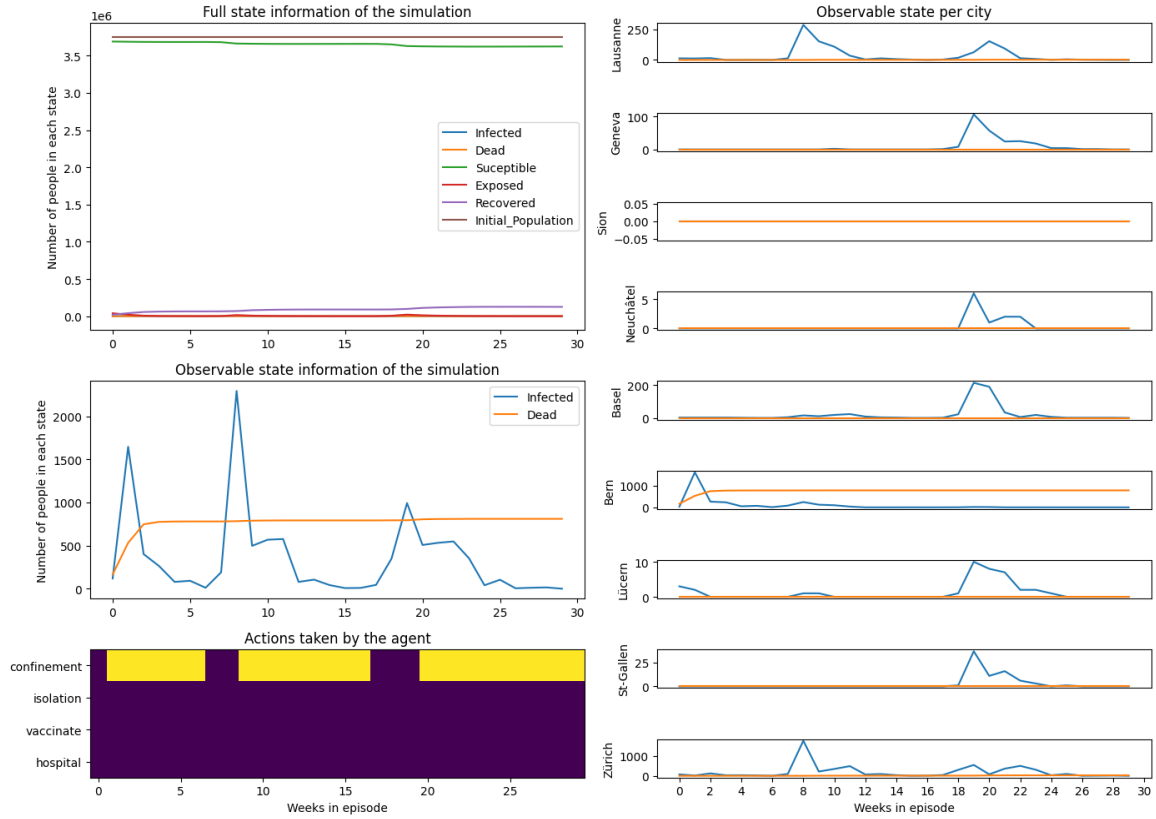


Figure 5: Sample simulation run for DQN with simple $\epsilon = 0.7$.

From looking at a few full episodes using this policy in Figure 5, it seems that this agent confines whenever the infected amount reaches a threshold of around 1000 people and then confines them for quite some time. How long this confinement lasts, or better under which condition it is lifted, is not directly clear, but likely depends on the total amount of infected people. Further, it seems that the model might have learned the delayed effect of people dying, as it sometimes does not confine the last week of the simulation (not shown in this run).

Question 3.b) decreasing exploration

Implement and train Deep Q-Learning agent for 500 training episodes with decreasing ϵ with $\epsilon_0 = 0.7$ and $\epsilon_{\text{min}} = 0.2$. Plot the evaluation and training traces on a shared plot with *Q3.b)*, compare and discuss the results between them. Which policy gets the best results and why?

Despite the difficulty in discerning from Figure 4b, the agent utilizing decreasing epsilon during training achieves a marginally better policy on average across various seeds. This can likely be attributed to reduced

randomness in expectations towards the training's conclusion. Consequently, the agent becomes biased towards attaining a satisfactory policy rather than investing excessive time in exploring the state space for potentially higher rewards. This effect is further evident when examining the distribution of training points.

Question 3.c) evaluate the best performing policy against Pr. Russo's policy

Run the best performing policy π^*_{DQN} through the evaluation code that you wrote to evaluate Pr. Russo's policy in question 2.b). generate the same histogram plots and compare the results Did the reinforcement learning policy outperform Pr. Russo's, if so in what sense?

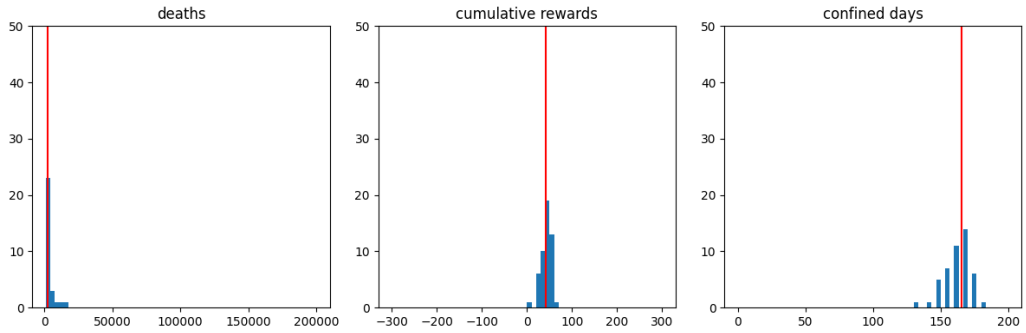


Figure 6: Histograms of Q3c); best simple DQN implementation.

Average	Deaths	Cumulative reward	Number of confined days
DQN confinement policy	2286	42.29	166

Table 2: Average values of final metrics for the DQN policy

The learned policy demonstrates a significant performance advantage over Pr. Russo's policy, despite the similarity observed in the actions taken during a single simulation run. Across the 50 simulations, the DQN agent with epsilon decay achieves an average of approximately 2.3k deaths, while Russo's policy results in around 65k deaths, indicating a reduction of more than an entire order of magnitude. Moreover, the total cumulative reward notably increases to approximately 40, depending on the chosen seed. This improvement is mainly accomplished by increasing the average duration of confinement for individuals in the simulation from an average of 110 days under Russo's policy to roughly 165 days for the DQN agent.

4 Dealing with a more complex action Space

4.1 Toggle-action-space multi-action agent

Question 4.1.a) (Theory) Action space design

Why would one want to use such an action-observation space, rather than directly compute $Q(s, a)$ for each action?

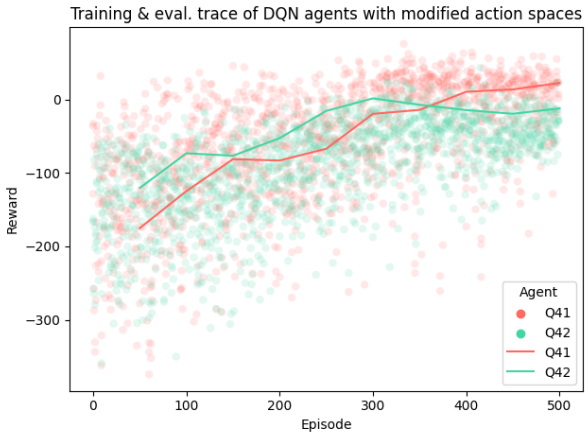
The main point is that such an action-observation space reduces the number of possible actions from 2^4 to "only" four possible actions. Another possible advantage of using such an action-observation space is that it makes the toggling cost more explicit because only one action can be toggled per week. Further, it allows for the possibility to change the state of the system, in terms of currently active actions, explicitly by an external actor, which is not as easily possible using directly computed Q(s, a).

The impact on the network architecture is minimal, the difference is the observation space which now contains additional neurons per action, i.e. the state the action is currently in (only four new neurons for us) and having four output neurons instead of a single one. With the same learning rate of 4.2, this agent seems to learn a bit slower and would benefit from running for 2000 episodes with a learning rate that is between the normal DQN and the factorized Q value one. This slower learning might be attributed to the fact that not all policies are possible for this agent, meaning on average this agent explores fewer ideal policies.

Question 4.1.b) Toggle-action-space multi-action policy training

Implement the toggled-action and observation spaces and train your Deep Q-Learning agent on it. Plot the training and evaluation traces (we ask you to average across 3 training runs).

(a) Training and evaluation traces from 4.1.b



(b) Training and evaluation traces for Q42 together with Q41.

Figure 7: Traces of Q41 left, together with those of Q42 right.

Is the agent properly learning?

Figure 7a presents the training and evaluation traces of the toggle-action-space agent. Observing the evaluation cumulative reward over the training duration, we notice that this agent exhibits slower learning compared to the previously examined DQN agents. Nevertheless, the agent manages to acquire a respectable policy, evident from the gradually increasing cumulative reward, averaging around 20. Furthermore, it appears that extending the training duration would likely enhance the agent's performance.

To better understand the behavior of the policy, plot one of the episodes and interpret it.

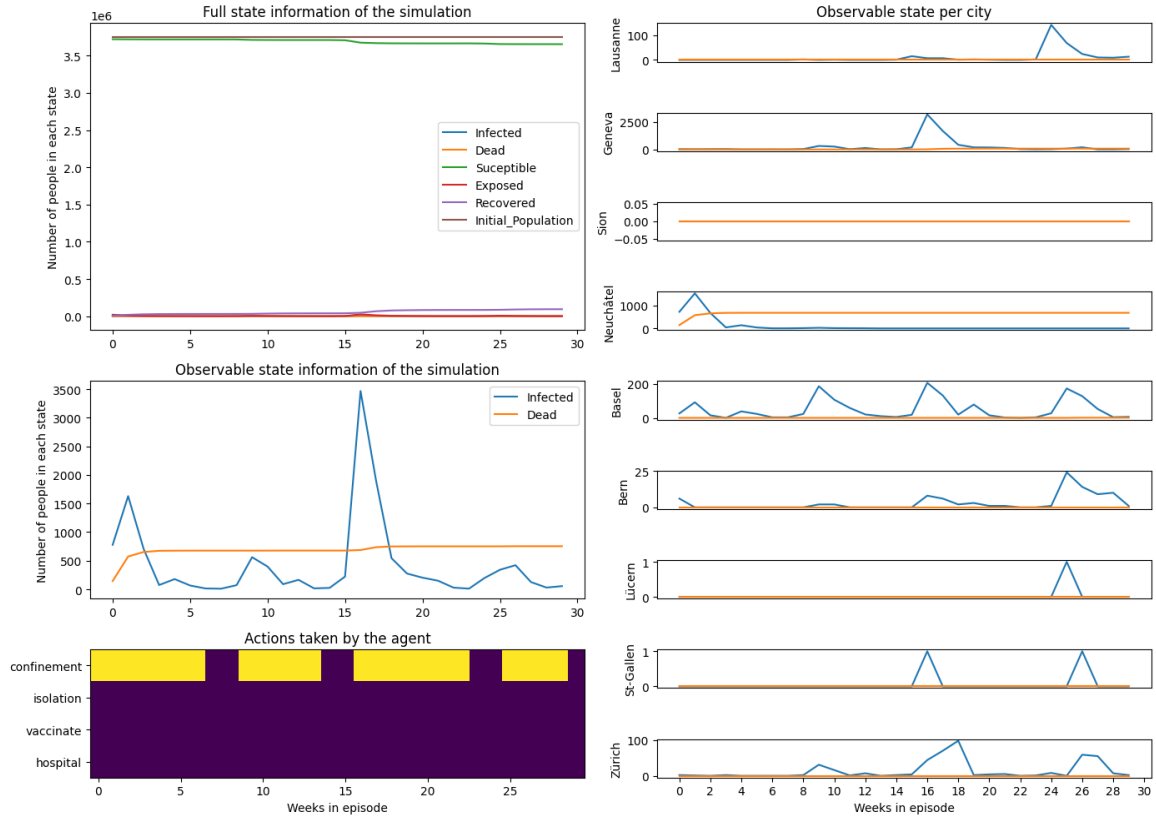


Figure 8: All necessary subplots of Q41 to interpret the policy.

Because of the higher dimension of the action space, interpreting can be more difficult, but it seems that this DQN agent learns a very similar base policy to the agent only allowed to do confinement, namely confine for long times whenever there is an increase of infections (see Figure 8). At the same time this agent often always starts with an initial confinement even with low numbers of infections at the start. This toggle-action space agent additionally seems to add sporadically hospital beds (not shown in this run of the policy), whenever the

infections are above a threshold of around 1000 people infected.

Question 4.1.c) Toggle-action-space multi-action policy evaluation

Plot the histograms and discuss the results

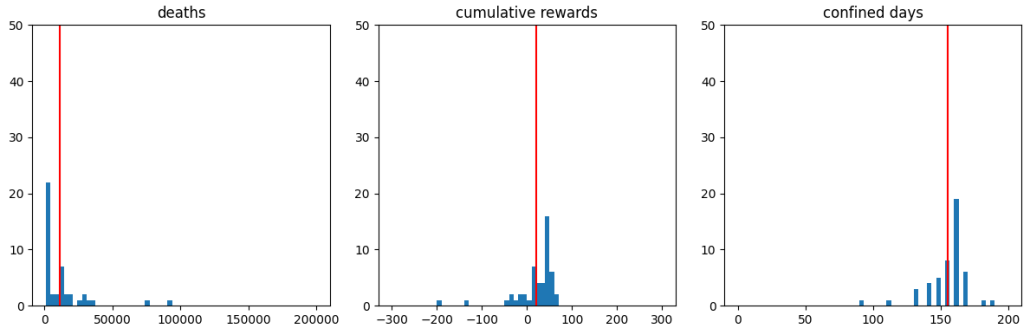


Figure 9: Histograms of Q41c); toggle-action space DQN agent.

The predominant strategy employed involves toggling confinement on and off, particularly when the number of infections is relatively low. As a result, there is a substantial number of days spent in confinement. This strict lockdown approach significantly reduces the number of infections and consequently leads to a low death toll. However, the frequent use of confinement adversely affects the overall reward, even if only a few deaths occur.

Table 3: Average values of final metrics for the DQN Toggle policy

Average	Deaths	Cumulative reward	Number of confined days
DQN Toggle policy	11207	19.99	155

How does the policy perform compared to the binary action policy evaluated in question 3.c?

On average, the toggle-action-space agent performs slightly worse than the binary-action-space agent. This outcome is largely anticipated since the toggle-action policy predominantly employs the same actions as the binary policy. The binary policy, with its smaller search space, benefits from a longer training time to refine its policy, leading to its superior performance.

Question 4.1.d) (Theory) question about toggled-action-space policy, what assumption does it make?

Could you **think of an action space for which toggling the actions would not be suitable**? Discuss.

Sometimes it is not allowed to choose the same action, for example in chess or shogi, hence it is not sensical to toggle actions. Another action space where toggling actions are not suitable is whenever two actions at the same time can complement each other well, because the toggle-action-space technique only allows two actions to happen with a delay, making the possibly optimal policy unreachable.

4.2 Factorized Q-values, multi-action agent

Question 4.2.a) multi-action factorized Q-values policy training

Implement the multi-action factorized Q-values agent and observation spaces and train your Deep Q-Learning agent on it. Plot the evaluation and training traces on a graph together with the traces from the toggle-action-space training. **Does it successfully learn?**

Similarly to the simple DQN we can see in Figure 7b that this agent does learn a policy (although if it does so successfully and to which degree depends heavily on the seed), which can also be seen by the rewards that are achieved until the end of training for the 20 evaluation episodes.

Run a few episodes of the best policy π^*_{factor}, to better understand the behavior of the learned policy. **Plot one of those episodes and interpret the policy. Is the policy realistic?**

In Figure 10, we observe an example simulation episode of the factorized Q-value agent. It is worth noting that this agent does not consistently learn the same policy in every run, resulting in varying evaluation rewards ranging from 20 to 40 depending on the seed. In many cases, the learned policy tends to initiate vaccination early, although not always in the first week. The effective policy (not shown in the above simulation) strives to strike a balance between minimizing the cost of hospital beds and limiting the maximum expected number

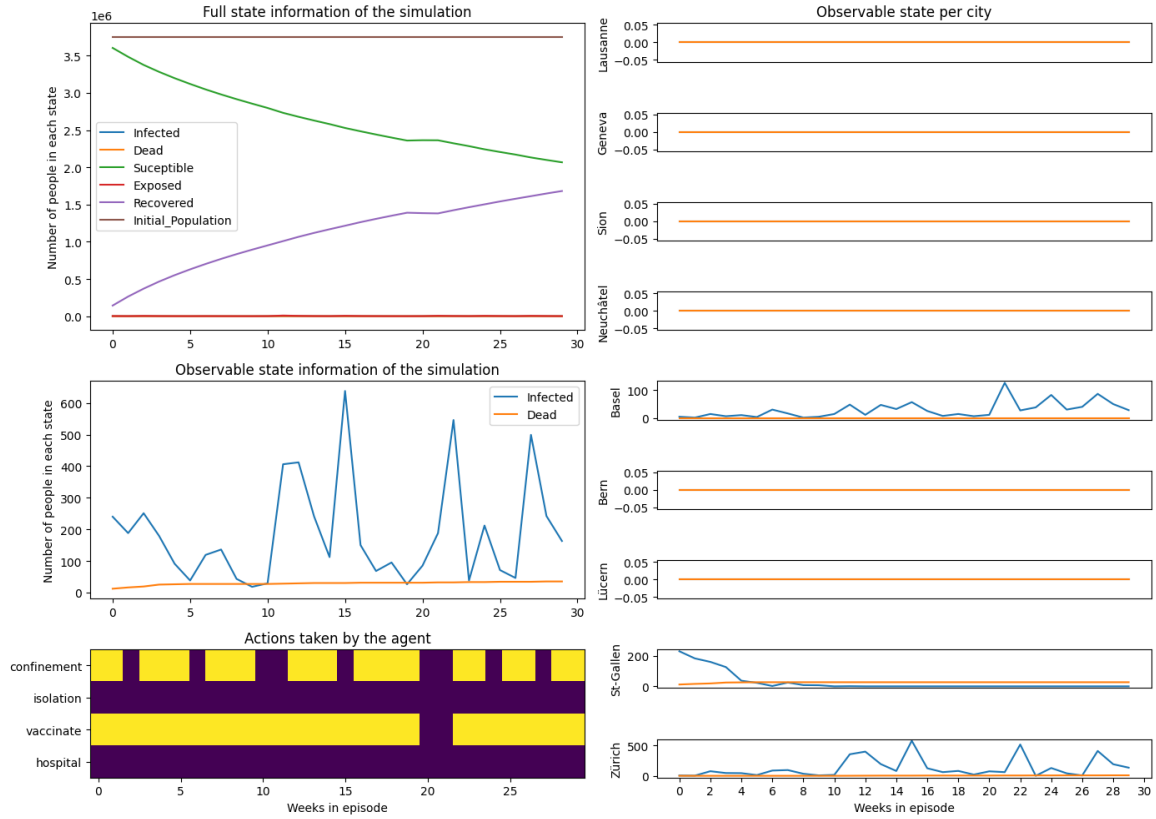


Figure 10: Sample of Factorized DQN policy

of simultaneous infections. Conversely, the less optimal policy (as depicted in Figure 10) frequently resorts to vaccination while switching rapidly between confinement and non-confinement measures. It is noteworthy that both policies rarely employ isolation as a strategy, based on our observations of the agent's learning behavior.

Question 4.2.b) multi-action factorized Q-values policy evaluation

Evaluate the best policy (π^*_{Factor}) trained in question *4.2.a)* **Plot the histograms**, discuss the results and compare them to the policy implemented in *4.1.b)*. **How does it compare to the toggled policy**?

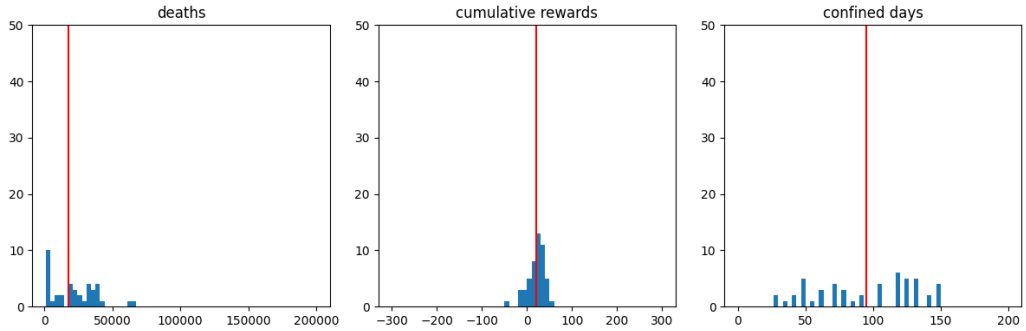


Figure 11: Histogram for Factorized Q-values

Average	Deaths	Cumulative reward	Number of confined days
DQN Factorized policy	17303	20	94

Table 4: Average values of final metrics for the factorized Q-value DQN policy

In good runs this policy is much better than the toggled policy with a reward around 40, less deaths and no confinement days. Whenever the multi-action agent does not manage to find a good policy (as the one plotted in Figure 11), it will behave along the lines of to the toggled policy, and learn something similar to the long confinement policy as in question 3b), but with vaccinations mixed in.

Question 4.2.c) (Theory) Factorized-Q-values, what assumption does it make?

What assumptions does the use of the factorized-Q-value policy make on the action space? Could you think of an action space for which factorizing Q-values would not be suitable?

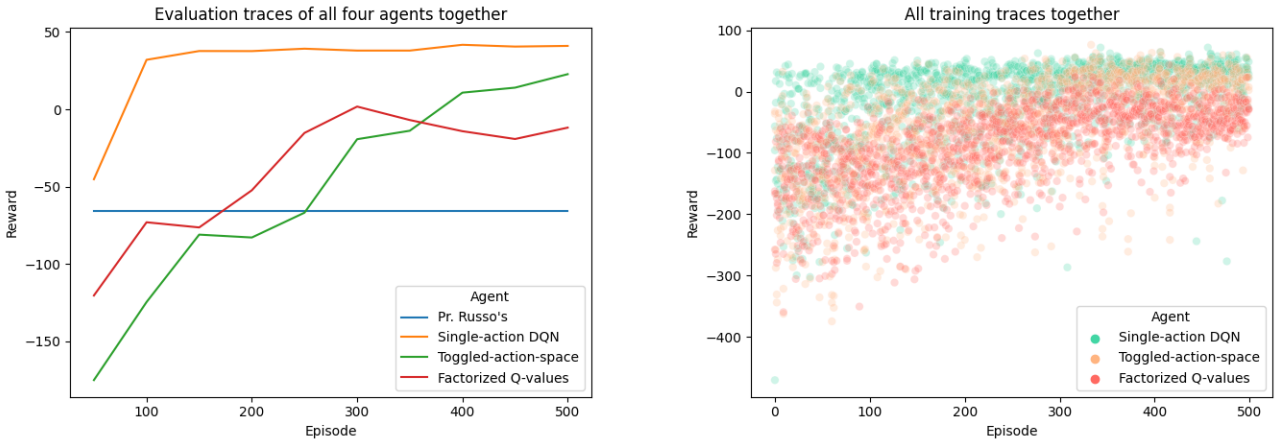
The assumption of such a factorized-Q-value policy on the action space is again that the actions are seperable into sub-actions, i.e. factorizable into those four actions, as well as that the optimal policy might require to activate multiple actions at the same time.

Any action space were actions are not factorizable is an action space for which factorizing Q-values is not suitable. An example is when the actions can have opposites like in a car that can driver left or right, but not both at the same time. In this case factorized Q-value imply that both actions left and right can be taken at the same time, even if this is not the case.

5 Wrapping Up

Question 5.a) (Result analysis) Comparing the training behaviors

Compare the evaluations and training curves of **Pr. Russo's Policy**, **single-action DQN**, **factorized Q-values** and **toggled-action-space** policies. Discuss the performance differences, what do you observe? How do the two approaches compare? What approach performs best? Why?



(a) Evaluation traces of all models plotted together.

(b) Training traces of all models plotted together.

Figure 12: Evaluation traces averaged over three runs, while all training rewards are shown together.

It is easy to see in Figure 12a that all RL models perform much better than Pr. Russo's policy after some time of learning. The evaluation traces of the three RL models show that the simple single-action DQN with epsilon decay is learning much faster than the multi-action agents and in the end on average better than the other two agents. The reason for this very likely boils down to the fact that this single-action DQN has a smaller action-space to explore, while the most optimal policy is contained within this action space of pure confinement. (Would policies with other actions be much better, then we would expect to see the other agents learn different policies.) From the training traces in Figure 12b it is possible to see the curse of dimensionality in effect, both the toggle-action-space agent in orange and factorized Q-value agent in red have a wider spread of training rewards than the single-action agent in green. Hence the bigger the action space the more training time is necessary.

Question 5.b) (Result analysis) Comparing policies

Run the evaluation procedure with each (π_{DQN}, π_{toggle}, π_{factor} as well as the original π_{russo}) trained policy for 50 episode and compute the metrics (all averaged as empirical means over all runs) of **total confined days**, **total isolation days**, **total vaccination days**, **total additional hospital bed days**, **number of total deaths** and the **cumulative reward**.

The final column of the table below holds the most crucial information, indicating the cumulative rewards. It is evident that the single-action DQN agent outperforms the others with a cumulative reward of approximately 43. However, it is worth noting that the factorized Q-Values agent achieves the same cumulative reward for certain seeds. This higher reward achieved by the single-action DQN agent is likely due to its exclusive use of confinement, effectively maintaining the average death toll around 1.5 thousand.

	Confined	Isolation	Vaccinate	Hospital	Deaths	Cum. Reward
Pr. Russo's	108.50	*	*	*	55,673.38	-65.67
Single-action DQN	166.60	*	*	*	1,504.40	43.57
Toggled-action-space	156.10	0.0	0.0	23.1	10,668.42	20.13
Factorized Q-values	101.22	0.0	180.6	0.0	13,652.24	22.04

Table 5: Table of key metrics for all different models averages over 50 runs. * means policy cannot use action, green is metric and agent that performed best under this metric.

Additionally, it is noteworthy that both agents capable of isolation never utilize this action. This observation leads to the conclusion that either isolation is generally too costly or these agents failed to discover a policy in which isolation proves worthwhile considering the associated cost.

Question 5.c) (Interpretability) Q-values

For both π_{DQN} and π_{factor}, produce a visualization of the estimated Q-values as a heat-map of the evolution with time. Discuss your results. How interpretable is your policy?

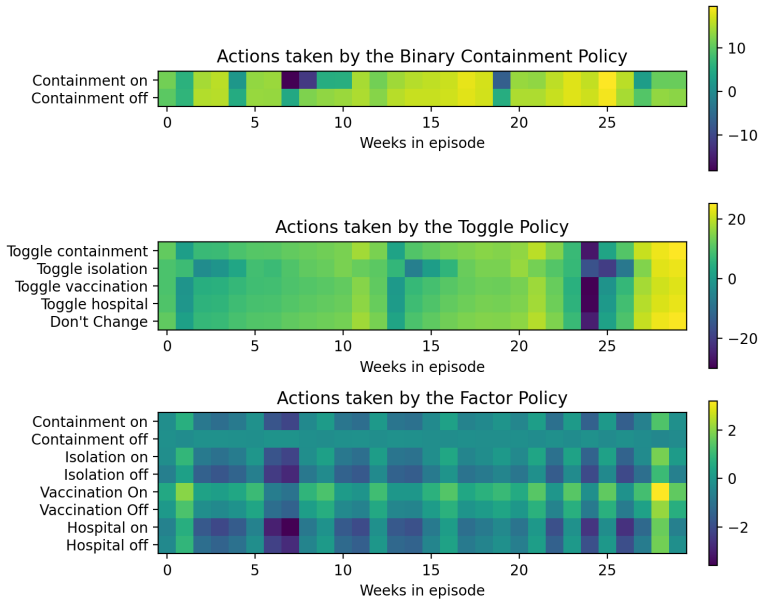


Figure 13: Heat map over Q-values for all DQN policies.

An observable repeating pattern can be discerned in Figure 13 for the evolution of Q-values over time for all the different models. This pattern is likely a result of similar states recurring, such as the number of infected approaching a threshold value. Once this threshold is reached, a response is triggered and maintained until a lower threshold is attained, at which point it is deactivated. Moreover, there is a subtle distinction between the toggle policy and factorized policy, where the Q-values' significance varies depending on the state. Consequently, all the values undergo dramatic changes.

Question 5.d) (Theory)

Is cumulative reward an increasing function of the number of actions?

Adding more actions provides the agent with increased choice, including not taking the new actions, and the cumulative reward should, in theory, be a non-strictly increasing function of the number of actions. However, in practice, this notion does not hold well due to the curse of dimensionality, which refers to the exponential increase in exploration required as the action space grows larger. Consequently, finding the optimal policy in a larger action space becomes significantly more challenging, increasing the likelihood of getting trapped in suboptimal policies.

The multi-action-space agent clearly demonstrates this phenomenon, achieving a better policy than the simple DQN agent for some seeds, but for others significantly worse. This is also evident in the training plot of Figure 12b, where both agents with more choice converge slower and to a worse policy. The curse of dimensionality poses challenges for exploring the expanded action space, hindering the agent's ability to consistently find the optimal policy.