Stochastic Optimal Control Methodologies in Risk-informed Community Resilience Planning

Saeed Nozhati[1,2], Bruce R. Ellingwood[1,2], Edwin K.P. Chong[3,4]

[1] Department of Civil and Environmental Engineering, Colorado State University, Fort Collins, CO 80523-1372, USA

[2] NIST Center of Excellence for Risk-Based Community Resilience Planning, Colorado State University, Fort Collins, CO, USA

[3] Department of Electrical and Computer Engineering, Colorado State University, Fort Collins, CO 80523-1373, USA

[4] Department of Mathematics, Colorado State University, Fort Collins, CO 80523-1874, USA

A B S T R A C T

The absorptive and restorative abilities of a community are two key elements of the community resilience following disasters. The recovery of communities relies on an efficient restoration planning of damaged critical infrastructure systems, household units, and impaired supporting social and economic functions. These interdependent systems form a dynamic system of systems (SoSs) that changes continuously during restoration. Therefore, an effective and practical recovery planning process for a community can be modeled as a sequential dynamic optimization problem under uncertainty. This paper seeks to enhance our understanding of dynamic optimization concepts and their role in formulating post-disaster, community-level recovery strategies. Various methods of classic dynamic programming and reinforcement learning are examined and applied. Simulation-based, approximate dynamic programming techniques are introduced to overcome the curse of dimensionality that is characteristic of a large scale and multi-state system of systems. The paper aims not only to study the unexplored topic of dynamic optimization in community resilience, but also to be a practical reference for policymakers, practitioners, engineers, and operations analysts to harness the power of dynamic optimization toward assessing and achieving community resilience.

Keywords: Community resilience, Dynamic programming, Optimal control, Reinforcement learning, Rollout algorithm, Stochastic optimization.

1. Introduction

Extreme natural hazards pose economic, physical, societal, and psychological threats to the wellbeing of modern urban communities. Consequently, community resilience research aimed at mitigating impacts of extreme natural hazard events and returning communities quickly to normalcy has expanded significantly in recent years. Much of this research has focused on understanding the factors that make a community resilient during and immediately following an extreme natural hazard. However, evaluating the post-event restoration/recovery phase is a non-trivial task and has received far less attention, in large measure because a community requires the functioning of numerous interdependent networks. Any analysis of post-disaster restoration must include a study of these networks, and their interactions, over the course of recovery. One of the cornerstones of community resilience, broadly defined, is an efficient restoration policy, whereby the effective delivery of the desired amount of goods or supplies is assured in the aftermath of a hazard [1].

Loosely speaking, various studies in the community resilience literature can be categorized into one (or more) of the following divisions:

(a) Studies that introduce a more comprehensive definition of resiliency. These type of studies focus on the concepts of resilience at the community level and the implications for the future of resilience engineering (e.g., [2-5]).

(b) Studies that strive to understand and model different networks at the community level scale, focusing on modeling of interdependencies within and between networks to capture the cascading effects. These studies also define new metrics based on the problem on hand and compute the restoration of a community over time. Some of these studies also try to find a mathematical model to describe the stochastic behavior of post-disaster recovery (e.g., [6-10]).

(c) Studies that attempt to identify optimal (or near-optimal) *ex-ante* mitigations or *ex-post* strategies. Such studies usually employ and/or develop different optimization methods. These

studies may also develop new metrics and objective functions that embed policymakers' preferences (e.g., [11-14]).

Optimization methodologies at the community level that satisfy community stakeholders as well as consider practical constraints require more attention. However, developing such methodologies at the community level raises challenging computational issues because a community with several interdependent networks is a dynamic time-dependent system of systems (SoSs) over the restoration period.

This paper reviews and introduces dynamic, approximate dynamic, and reinforcement learning algorithms to optimize community resilience planning. Such algorithms have been advancing rapidly in the field of optimal control for the decision-making problems under uncertainties. Unfortunately, the leverage of these strong techniques in community resilience and recovery optimization is still a largely unexplored topic. With this in mind, this study makes the following main contributions:

- It introduces the concepts of dynamic optimization, sequential closed-loop optimization, and the value of community resilience formulation based on these concepts.
- It formulates post-disaster community recovery within a Markov decision process framework and leverages its theory in the optimization formulation.
- A step-by-step description of several classical dynamic programming and reinforcement learning algorithms is presented. The advantages and challenges of each algorithm when used in community resilience assessment are probed in. These classical algorithms are applied to a hypothetical case study.
- The study discusses the twin *curses of dimensionality* and *modeling* for moderate sized and very large communities, and introduces approximate dynamic programming (ADP) techniques to overcome the challenges they pose for resilience assessment. Simulation-based methodologies are introduced and applied to a real case study of a community of moderate size to reveal the applicability of these algorithms in a real-world context.

One of the classical studies of community resilience [4] identifies resiliency with four attributes: robustness, redundancy, resourcefulness, and rapidity. The methodologies in this paper focus significantly on the optimization of stochastic post-disaster recovery, i.e., the resourcefulness and rapidity dimensions identified by Bruneau, et al [4]. Nevertheless, they can

also be reformulated and applied for *ex-ante* mitigation plans to cover up all aspects of community resilience planning. Moreover, we focus only on the systems (or SoSs) which can be modeled by Markov or semi-Markov processes in the realm of control optimization.

The remainder of this study is structured as follows. In Section 2, the motivations and the background of dynamic optimization and Markov decision process formulation for the community recovery are presented. In Section 3, several classic dynamic programming and reinforcement learning methodologies are introduced. In Section 4, a class of ADP algorithms is probed in details. In Section 5, we present some complementary points.

2. Motivation

Interdependent critical infrastructure systems (ICISs), such as transportation, energy, water, supply chains, household units, and healthcare systems, are cornerstones for the functioning of society. Therefore, the functionality of ICISs is essential to the well-being of the community. The operability of ICISs can vary significantly over the recovery phase. Once a policymaker (or a decision maker) decides to repair a malfunctioned component, the outcome of this decision can potentially affect the functionality of a network or several networks. Therefore, the outcome of the decision may have a broad impact on the community as a whole, produced by interdependencies within and among networks. Furthermore, the outcome of a repair action is not fully predictable. Therefore, a decision maker deals with a stochastic dynamic SoSs that changes sequentially, beginning immediately following the occurrence of the hazard until the end of recovery. With this in mind, the efficient restoration of ICISs requires a comprehensive decision-making framework to consider different sources of uncertainty to support policymakers. The identification of optimal strategies for this dynamic SoSs needs strong optimization methodologies to satisfy community stakeholders' objectives and constraints.

A broad spectrum of real optimization problems can be explained in various ways of naming and classifying optimization methods (see Fig. 1). Broadly speaking in the field of operations research, the stochastic complex problems belong to two main branches; *static* optimization (also referred to as parametric optimization) and *dynamic* optimization (also referred to as control optimization) [15].

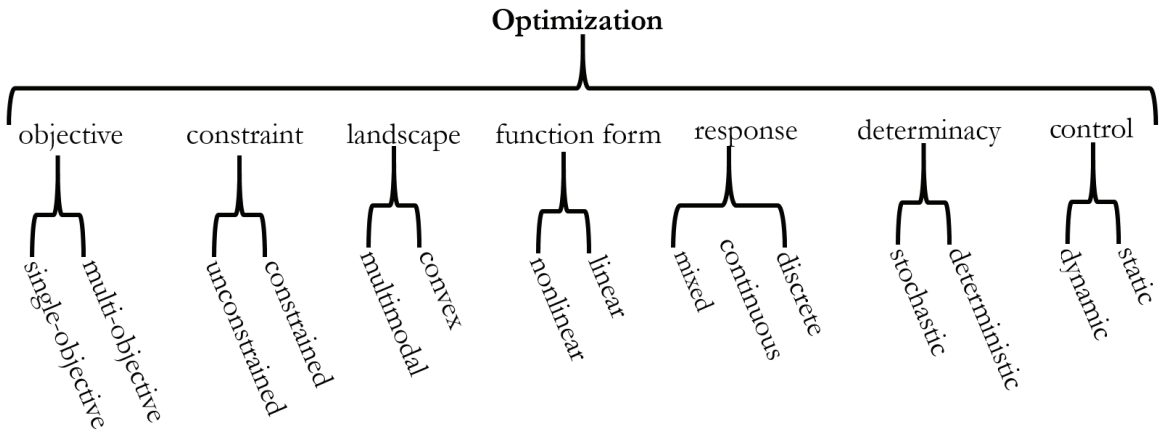

Fig. 1. Classification of optimization problems.

Static optimization is employed to determine values of decision variables (or parameters) that optimize some objective functions. It is called static because the solution is a set of "static" parameters for all states. Classically, this type of optimization is performed utilizing mathematical programming, such as linear, non-linear, and integer programming. Some researchers have studied the community resilience problem with the aid of these methods by simplifying and set assumptions on the real problem [13].

On the other hand, dynamic (or control) optimization identifies a set of decisions to be made in the various states that a system or SoSs visits so that the chosen decisions optimize some performance measure of the SoSs. It is called dynamic, inasmuch as the solution depends upon the dynamic states and we may have a different solution per each state. The dynamic optimization methodologies are more appropriate than static ones since a community is a dynamic SoSs. Classically, control optimization is generally performed via dynamic programming [15]. The next section introduces and summarizes the dynamic programming concept, the cornerstone of all methodologies presented in this study.

3. Classic dynamic programming and reinforcement learning techniques

3.1.Background

Dynamic programming (DP) and reinforcement learning (RL) are strong methods for situations in which decisions are made consecutively to a system or SoSs over an extended

period of time, to accomplish the desired goal (e.g., the policymakers' preferences). DPs technique breaks a complicated problem and tries to solve sub-problems in a recursive manner, based on Bellman's *principle of optimality* [16]. This principle suggests that an optimal policy can be formed in a piecemeal manner and the optimization of the future is independent of the past [16].

A comprehensive decision-making framework must consider the consequences of each decision in the long run and balance the desire for low present costs with the undesirability of high future costs (also called as *challenge of delayed rewards*) [16, 17]. This lookahead property complicates the solution exceedingly when a policymaker must be "far-sighted" until the end of the recovery process. The DP-based methods capture this trade-off so that it orders the decisions based on the sum of the present cost (or reward) and expected discounted future costs. The cost function is additive over the recovery process.

3.1.1. Closed-loop vs. open-loop optimization

The core of DP and RL-based methods is the closed-loop optimization in which the decisions are made in stages so that the outcome of earlier decisions is controlled and taken into account when making new decisions. This formulation gathers the information between time slots and profoundly enhances the quality of decisions. Conversely, in the open-loop formulation, a series of decisions is selected once without waiting to observe the succeeding demand levels [16].

In disaster resilience planning, the *coupled* resilience is occurred when extreme events are narrow in time; then, one or more than one drops of functionality can happen over the recovery process due to several events (e.g., strong aftershocks, post-earthquake tsunamis or fires, etc.) [18]. This interaction of the recovery process between narrowly spaced events is unpredictable and imposes disturbances on the problem. The coupled resilience signifies the importance of closed-loop formulation for community resilience assessment when the outcomes of the open-loop formulation can possibly become inaccurate owing to narrowly spaced events. The difference between closed and open-loop formulations is called Value of Information (VoI). The VoI indicates how much the information between time slots can be worth to a policymaker. In the deterministic optimization case without any random disturbances, there is no difference between closed and open-loop formulations [16].

3.1.2. Model-based vs. model-free

DP-based methods can be implemented when a model of the system is available. An important advantage of DP-based methods is that few assumptions are imposed on the system or SoSs, which can generally be stochastic and nonlinear. In contrast, linear programming or classical automatic control methods restrict the system with the assumptions like linearity or determinism. These restrictive assumptions may become serious impediments in community resilience planning. While DP methods require a system model, that model need not be an analytical model. The methods can also interact with a simulation (generative) model. Deriving generative models is usually easier than constructing an analytical model, especially for large-scale and complex problems like community resilience assessment. Indeed, for very large communities exposed to extreme hazards, a model of the SoSs may not be achievable; For such communities, constructing generative or analytical models may be extremely computationally expensive, or there may be inadequate information to understand the performance of large SoSs. In such situations, model-free RL methods that utilize only data obtained from the community and require no prior knowledge of the community (see Fig. 2), may be alternatives [19].

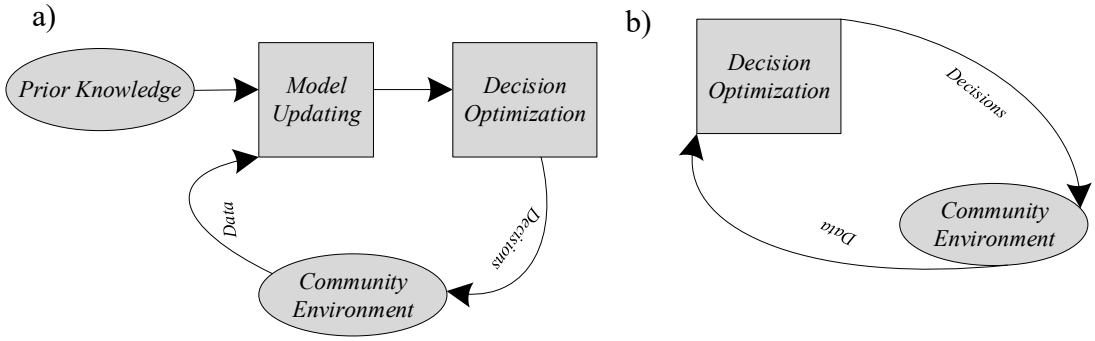

Fig. 2. Schematic representation of a) model-based methods and b) model-free approaches.

3.2.Markov decision processes

DP and RL-based problems can be formalized with the aid of MDPs [19]. MDPs are mainly utilized to model dynamic decision-making problems with multiple-periods under stochastic circumstances. In the most basic form, an MDP can be defined by (X, U, f, ρ, γ), where

X is the state space, U is the actions space of the decision maker (also called as controller or agent), and f (f: $X\times U\times X\rightarrow[0,1]$) is the transition function that defines how the state changes as a result of decisions. At the discrete time step k, the decision maker applies the action u_k to the community in the state x_k, and the state changes to x_{k+1} based on $x_{k+1} = f(x_k,u_k)$. That state x_{k+1} must summarize past information of the community that related to future optimization. For countable state space, the probability of the next state, x' per pair of (x_k,u_k) is:

$$P\left(x_{k+1}=x' \mid x_k,u_k\right)=f\left(x_k,u_k,x'\right) \qquad s.t. \qquad \sum_{x'} f\left(x_k,u_k,x'\right)=1 \tag{1}$$

Simultaneously, the decision maker receives the reward of r_{k+1} based on ρ: $X\times U\times X\rightarrow\mathbb{R}$ so that $||\rho||_\infty=\sup_{x,u,x'}(x,u,x')$ is finite. The rewards are also stochastic because they depend upon the next random state. Thus, we are interested into the *expected* rewards. Note that ρ is a deterministic function of ($x_k,u_k,\ x_{k+1}$), which means that once the next state is determined, r_{k+1} is completely defined. Based on the Markov property, the pair (x_k,u_k) must define the probability density of the next state.

The decision maker makes a decision based on the policy π: $X\rightarrow U$, $u_k = \pi(x_k)$. The identification of the optimal policy is the final object of DP and RL techniques. An optimal policy must maximize the return from any arbitrary initial state x_0. The return is a cumulative summation of rewards along a trajectory starting at x_0. It depicts the reward obtained by the policymaker in the long run. The finite-horizon discounted return is:

$$R^{\pi}(x_0)=\sum_{k=0}^{K_{\max}}\gamma^k r_{k+1}=\sum_{k=0}^{K_{\max}}\gamma^k\rho(x_k,\pi(x_k)) \tag{2}$$

where $K_{\max}= \max_x K(x)$ and $\gamma\in[0,1)$ is the discount factor. The discount factor determines how "far-sighted" the policymaker is in acknowledging the future rewards, and also reflects the uncertainty increment in future rewards. These control optimization methodologies can handle the challenge of delayed future rewards by constructing an underlying MDP and deriving benefit from lookahead property of the methods. The decision maker can determine the value of the future rewards by adjusting the discount factor based on the policy makers' preferences. The bigger γ (e.g., higher than 0.95) is recommended [19]. The decision maker can decrease γ to increase the convergence rate with the cost of lowering the quality of decisions.

3.3. Numerical example

In this section, a general formulation of a MDP and the application of classic DP and RL methods for community resilience modeling are presented. A hypothetical system with six components is considered. While this system is trivial in a real-world context, it enables the features of the analysis to be clearly visualized and is comparable in scope to the problems studied in [20, 21]. Furthermore, the size of this community enables us to apply the classic methods that are contingent upon the size of the problem and fall into the curse of dimensionality. These six components can be assumed as household units, elements of an infrastructure system like an electrical power network, or any other critical facility within a community. The decision maker should know the functionality of each component in the aftermath of a hazard. The functionality of a component depends on the level of damage it sustains and the functionality of the components on which it depends. These interdependent effects can be captured in different ways [14, 22, 23]. The decision maker knows the functionality and the level damage of each component either with inspection following the hazard or with the aid of simulation prior to the hazard. The latter case involves two steps; first, the hazard intensity measure (*IM*) must be computed at the location of each component; second, the level of damage can be evaluated by the fragility curves, which describe the probability that the component experiences a particular level of damage as a function of IM. If n fragility curves are available for each component, the component can be in (n+1) different states per *IM*. Let $x_i(t)$ represent all information regarding component i at time t following the hazard, including the instant damage state and the post-disaster lifetime of the component. For this example, we consider three fragility curves of minor, major, and collapse (n=3). The vector $X(t)$ describes jointly the states of the components at time t. This vector represents the state of the system, SoSs, or the whole community. Note that we do not formulate the MDP at the component level, but the states of all components form one possible state of the MDP at the community level.

$$X(t)=(x_1(t), x_2(t), \ldots, x_M(t)) \qquad s.t. \qquad |X(t)|=M \tag{3}$$

where M is the number of components within the community (e.g., M=6 for our problem). Therefore, the state space depends on the number of components. As alluded to earlier, three

fragility curves per each component including undamaged state produce four total states for each component ($n+1$). Thus, the size of the state space at the SoSs level is $(n+1)^M$, equal to 4,096 for our problem.

A decision maker can apply $j_i(t)$ different actions to component i at time t, e.g., do nothing, minor repair, major repair, and complete repair. Let $A(t)$ represent the all possible actions that a decision maker can apply at the community level at time t.

$$A(t) = \left(a_1(t), a_2(t), \ \ldots, a_M(t)\right) \qquad s.t. \qquad \left|A(t)\right| = \prod_{i=1}^{M} j_i(t) \tag{4}$$

in which $a_i(t)$ is the action on component i at time t. The stack of the actions for all components produces the action at the community level. However, a decision maker generally cannot apply repair actions to an arbitrary number of components due to the restriction in the number of units of resources. The resource units (RU) denotes the group of tools, vehicles, crews, etc, required to repair or replace a damaged component. For example, in the typical case when a fixed number, R, of RUs are available to repair M components, and $M>>R$, the decision maker has no choice except "do nothing" for (M-R) number of components. Practically, the number of RUs varies over the recovery process, $R(t)$. For example, Ouyang and Dueñas-Osorio [24] noted that the number of RUs increased in a roughly linear fashion in the aftermath of Hurricane Ike in 2008.

At each decision-making time, RUs can be assigned to the components in two different ways. First, if each component requires only one RU, the impact that several RUs may have in reducing the repair time of a damaged component is not considered. In this case, the dimension of action space at time t is:

$$|A(t)| = \binom{M}{R(t)} \tag{5}$$

Second, the decision maker may assign more than one RUs to a component that has suffered significant damage to restore it in the shortest possible amount of time. In this case, a precise mapping function between the reduction in repair time and the number of RUs must be available. Few studies have identified appropriate mapping functions; some [20] have suggested that this mapping function must be computed from empirical data. Note that in the first case, the

mapping is one-to-one. The second case can also be interpreted in the way that R is the maximum number of RUs and the assignment of "do nothing" may be more profitable in some cases. In this case, the decision maker may choose to employ less than R Rus or assign more than one RU to a damaged component. While the first case is usually more common in the literature review [7, 14, 25], it can be interpreted as a special case of the second case, in which the decision maker has more flexibility. In the second case, then,

$$|A(t)| = \sum_{i=0}^{R(t)} \binom{M}{R(t)-i} \tag{6}$$

Therefore, in addition to the number of damaged components, the number of available RUs, $R(t)$, plays a significant role in the action space dimension, as shown in Figure 3. From the computational budget perspective, very low and very high number of RUs are desirable for the first case (Fig. 3a). However, depending on the number of components, the number of RUs can lead to an enormous action space in the second case, as depicted by Fig. 3b, which can be intractable for real-world problems. For our example problem we assume that R equals to 2 and the action space is defined as in the second case; therefore, $|A(t)|=22$.

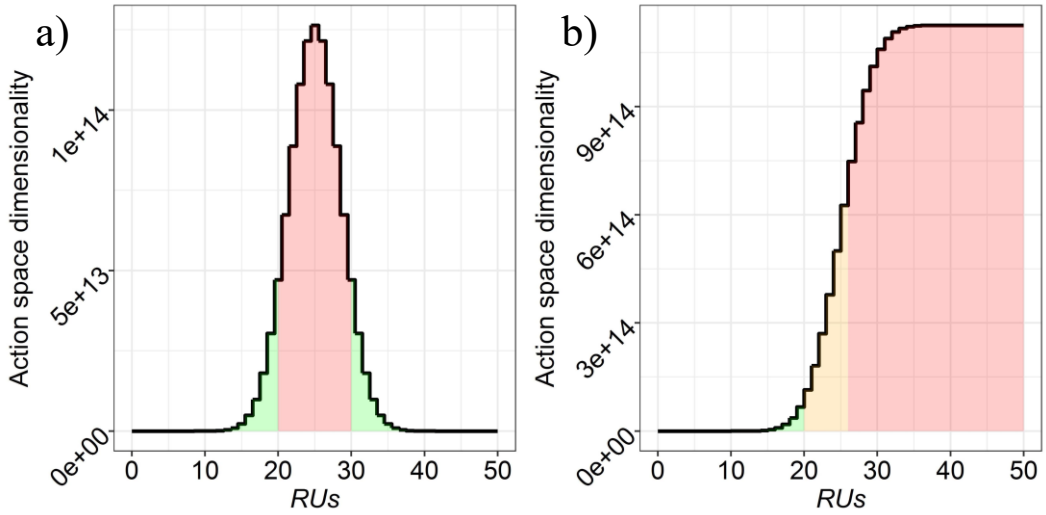

Fig. 3. The action space dimensionality a) one-to-one mapping b) general case

Unlike uncontrolled Markov chains, in MDPs each control mechanism has its own transition rules like the transition probability matrices (TPM). To utilize the model-based DP

methods, we assume and synthesize the analytical models from the past studies [26, 27]. Two different actions of do nothing (DN) and fully repair (FR) are considered for our hypothetical case ($j_i(t)$=2). Note that the concatenation of all components' damage state *history* following the hazard forms the MDP state at the community level. The TPMs associated with each action, the transition rules for do nothing (TPM_{DN}) and fully repair (TPM_{FR}) are presented in Table 1.

Note also the community can possible degrade owing to TPM_{DN}. The degradation under a "do nothing" action can potentially consider the coupled recovery process due to the interdependent hazards. It means the intact state of x_{in}=(1,1,1,1,1,1) in which all components are intact is not an absorbing state. Therefore, there is always a chance that the system would degrade. Strictly speaking, the agent deals with an infinite-horizon decision-making problem.

There is an associated reward for each transition in a MDP. This reward is referred to as the immediate reward or transition reward and the decision maker can represent immediate rewards in a transition reward matrix (TRM). In the MDP formulation, TPM and TRM include all information the agent requires to evaluate the restoration policy. TRM and TPM must be computed based on the type of the components and possibly the hazard. For example, Lin and Wang [7] introduced a methodology to compute TPM for residential buildings. A negative value for the reward is equivalent to a cost, represented in Table 1 and adding a constant to each cost would not change the subsequent policy [17].

Table 1
Cost and transition probability matrices at the component level

Damage state description at time t	Action cost ($/ft^2)	State risk cost ($/ft^2)
1 Undamaged	30	0
2 Minor damaged	75	90
3 Major damaged	85	250
4 Collapsed	95	350

$$TPM_{DN}: \quad T_1 = \begin{bmatrix} 0.95 & 0.05 & 0.0 & 0.0 \\ 0.0 & 0.9 & 0.1 & 0.0 \\ 0.0 & 0.0 & 0.85 & 0.15 \\ 0.0 & 0.0 & 0.0 & 1.0 \end{bmatrix}$$

$$TPM_{FR}: \quad T_2 = \begin{bmatrix} 1.0 & 0.0 & 0.0 & 0.0 \\ 0.95 & 0.05 & 0.0 & 0.0 \\ 0.90 & 0.10 & 0.0 & 0.0 \\ 0.85 & 0.15 & 0.0 & 0.0 \end{bmatrix}$$

3.4. The Bellman equations and value function

The value function describes the long-term value of states if a community experiences them. The state value functions (V-functions, also called as cost-to-go function) indicate the total amount of return a policymaker can expect over the course of recovery. While the immediate rewards are given by the TRMs, the values must be computed from the series of decisions a policymaker makes during the restoration process. The state-action value functions (Q-functions)

describe the value of a specific decision in a specific state under a particular policy of the policymaker. The Q-function Q^{π}: $X \times U \rightarrow \mathbb{R}$ of a policy π is:

$$Q^{\pi}(x,u) = \underset{x' \sim f(x,u,.)}{\mathrm{E}} \left\{ \rho(x,u,x'), \gamma R^{\pi}(x') \right\} \tag{7}$$

The best Q-function, also known as the optimal Q-function, for any policy is:

$$Q^{*}(x,u) = Q^{\pi}(x,u) \tag{8}$$

Since the ultimate goal of policymaker is the optimal policy (π^{*}), the policymaker should select an action based on π^{*} with the largest optimal Q-value at each state of the community:

$$\pi^{*}(x) = \arg\max_{u} Q^{*}(x,u) \tag{9}$$

Bellman showed that a recursive relationship between the value of a state and the values of its successor states exists [28]. The Bellman equations for Q^{π} and Q^{*} are:

$$Q^{\pi}(x,u) = \underset{x' \sim f(x,u,.)}{\mathrm{E}} \left\{ \rho(x,u,x') + \gamma Q^{\pi}(x', \pi(x')) \right\} \tag{10}$$

$$Q^{*}(x,u) = \underset{x' \sim f(x,u,.)}{\mathrm{E}} \left\{ \rho(x,u,x') + \gamma \max_{u'} Q^{*}(x', \pi(x')) \right\} \tag{11}$$

3.5. Value iteration

The optimal V-functions and Q-functions can be computed by value iteration. These techniques utilize the Bellman equations iteratively to obtain the optimal policy. Our hypothetical example provides knowledge of the transition and reward functions. Therefore, we initiate the computation of optimal recovery strategies with model-based DP algorithm of Q-iteration. It is followed by the classical reinforcement learning techniques that do not require an explicit analytical model.

3.5.1. Model-based value iteration

Let $\mathcal{Q}$ denote the space of bounded real-valued Q-functions and T: $\mathcal{Q} \rightarrow \mathcal{Q}$ computes the right-hand side of the Bellman optimality, Eq. (11). Therefore, for an arbitrary Q-function:

$$[T(Q)](x,u) = \underset{x' \sim f(x,u,.)}{\mathrm{E}} \left\{ \rho(x,u,x') + \gamma \max_{u'} Q(x',u') \right\} \tag{12}$$

The Q-iteration method begins with an arbitrary Q-function, Q_0 , and at each iteration ℓ, the Q-function is updated:

$$Q_{\ell+1} = T(Q_\ell) \qquad s.t. \qquad \| T(Q) - T(Q') \|_\infty = \gamma \| Q - Q' \|_\infty \tag{13}$$

The optimal Q-function, Q^*, is a fixed point of T (i.e., $Q^* = T(Q^*)$). Therefore, Q-iteration asymptotically converges to Q^* as $\ell \rightarrow \infty$ [19]. However, a decision maker can choose conservatively a finite number L of iterations that provides a suboptimality bound $\varepsilon_{QI} > 0$ [19]:

$$L = \left\lceil \frac{\varepsilon_{QI}(1-\gamma)^2}{2 \| \rho \|_\infty} \right\rceil \tag{14}$$

Conceptually, DP assumes $Q_{\ell+1}(x,u)$ determines $Q_\ell(x,u)$ for all states. This is often referred to as backward DP.

An appealing feature of dynamic programming methods is that the optimal policy typically is the same regardless of initial condition [16]. Lest the reader be confused by this claim, we want to make it clear that we are *not* saying that the initial condition does not matter. Indeed, the optimal action at the initial state is a function of the initial condition. Moreover, the overall objective function value will depend on what action is performed at the initial state. Instead, what we are saying is that this appealing feature of dynamic programming methods is that when we do Q-iteration, we need not concern ourselves with what initial condition to apply to the training procedure. If the procedure converges, the resulting policy will be the same regardless of what initial condition was assumed. The user can, therefore, set whatever initial condition they wish. In our simulations shown later, we set the initial condition for the training process to be x_0=(4,4,4,4,4,4).

When the algorithm determines the optimal action at each community's state, it assigns the RUs to the selected components. Each damaged component requires a period of repair to be restored. This repair period is a random variable and the outcome of the decision is not fully predictable. We assume that the distributions of repair times [22, 29] can be described as exponential distributions, represented in Table 2. Section 4 discusses the repair time distribution in details.

Table 2.
The expected restoration times based on the level of damage

	Undamaged	Minor Damaged	Major Damaged	Collapsed
Component	0	30	120	230

Table 3 summarizes a few selected states and their corresponding optimal actions, computed by the Q-iteration algorithm based on maximum two RUs. The optimal action represents the number of components that the agent should assign the RUs.

Table 3.
Sample states and their corresponding optimal actions

State number	State description	Optimal action
1	(1,1,1,1,1,1)	(0,0)
700	(1,3,3,4,3,4)	(4,6)
769	(1,4,1,1,1,1)	(0,2)
4096	(4,4,4,4,4,4)	(1,3)

To show the convergence of the algorithm to the fully restored state, we consider the component damage state of 4, 3, 2, and 1 as 0, 33, 66, and 100% functionality, respectively. For example, the state of (1,2,3,4,1,2) represents 60.83% functionality at the community level. Fig. 4 shows the restoration time of the hypothetical community of Section 3.3. The mean with one and two standard deviation bands are computed based on 1000 different random numbers generation. The Q-iteration algorithm asymptotically converges to the intact state x_{in} after seven iterations. It is worth emphasizing that the community can possibly leave this state owing to the degradation in the system. The degradation can be seen in some recovery trajectories, which indicates a recovery process trajectory is not necessarily monotonic. However, the mean curve is monotonically increasing.

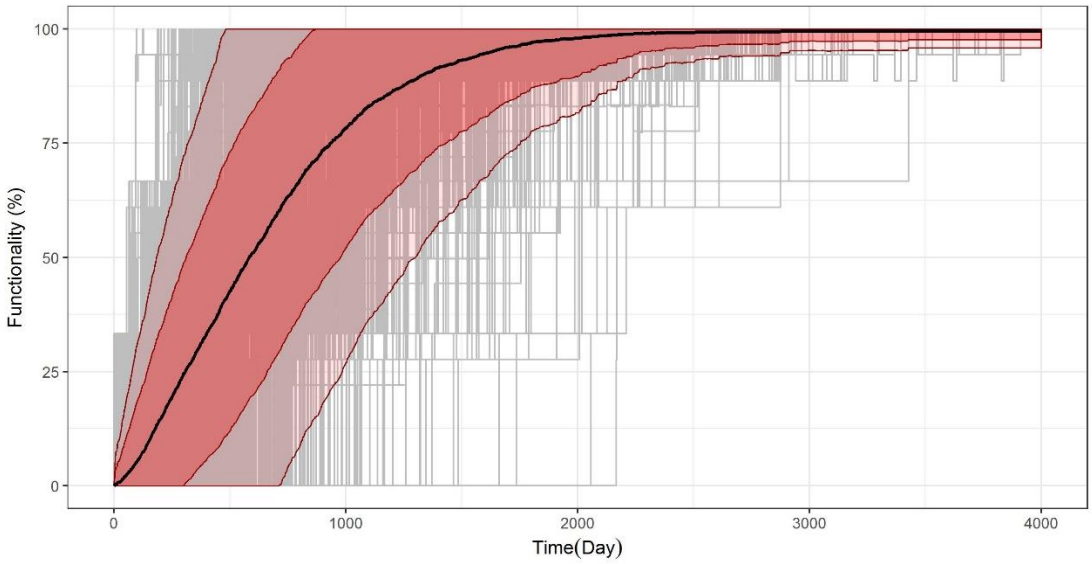

Fig. 4. The Q-iteration restoration policies with mean and ± (2) σ

Fig. 5 presents the reward (or cost) of the restoration decisions during the recovery period. In the beginning, the slope of the expected required cost increases drastically with a significant dispersion immediately following the occurrence of the hazard. However, as the level of damage over the community decreases because of the repair policies, the expected required cost decreases over time. The expected cost cannot reach zero because there is always cost to maintain components even after the recovery process, although it is much less than the restoration costs.

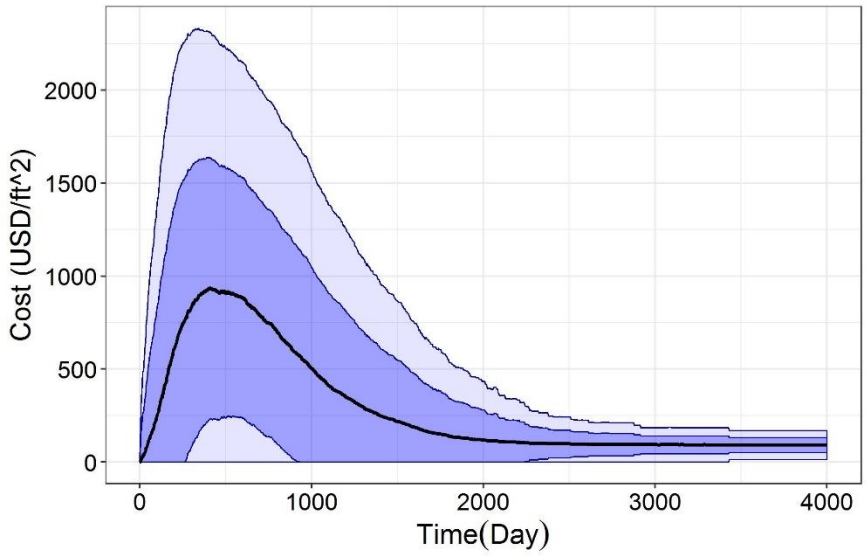

Fig. 5. The expected Q-iteration reward and ± (2) σ

3.5.2. Model-free value iteration (Q-learning)

One of the classical model-free reinforcement learning approaches is Q-learning [17], and indeed Q-learning is perhaps the most widely used approach in this class. The required data tuples of Q-learning, $(x_k,u_k,x_{k+1},r_{k+1})$, indicates that Q-learning does not require a model. It initiates with an arbitrary initial Q-function Q_0 and updates it based only on observed state transitions and rewards. Thus, the decision-maker can update the quality of each decision and Q values using such a data tuple after each transition, as follows:

$$Q_{k+1}(x_k,u_k) \leftarrow Q_k(x_k,u_k) + \alpha_k[r_{k+1} + \gamma \max_{u'} Q_k(x_{k+1},u') - Q_k(x_k,u_k)] \tag{15}$$

where $\alpha_k \in (0,1]$ is the learning rate and the term $[r_{k+1} + \gamma \max_{u'} Q_k(x_{k+1},u') - Q_k(x_k,u_k)]$ is the temporal difference (TD). The TD indicates the difference between the updated assessment $\gamma \max_{u'} Q_k(x_{k+1},u')$ of the optimal Q-value of (x_k,u_k), and the current evaluation $Q_k(x_k,u_k)$. Q-learning can be considered as a sample-based, stochastic approximation procedure since the process of updating the Q-values produces a single sample of the random quantity, the expectation of which is computed by the Q-iteration mapping (Eq. (12)). It can be shown that the solution from the Q-learning algorithm asymptotically converges to Q^* as k goes to infinity,

provided that the state and action spaces are finite and discrete, and that two conditions are satisfied [19]; first, the sum $\sum_{k=0}^{\infty}\alpha_k$ provides an infinite value, while the sum $\sum_{k=0}^{\infty}\alpha_k^2$ provides a finite value (e.g., $\alpha_k=\frac{1}{k}$); and, second, all the state-action pairs frequently. The second condition can be satisfied if the decision maker enables to explore an arbitrary action in every visited state and exploits his/her current knowledge to achieve high-quality performance (e.g., choosing greedy decisions in the Q-function). In the RL field, this is called the *exploration-exploitation* trade-off [19]. Classically, Q-learning is implemented with a ε-greedy exploration or modifications such as adaptive ε-greedy exploration or Boltzmann selection, in which the decision maker explores the new Q values to increase the quality of recovery decisions. The classical method of ε-greedy exploration balances exploration with exploitation in Q-learning, exploration probability of $\varepsilon_k\in(0,1)$ at step k as follows;

$$u_k=\begin{cases}u\in\arg\max_{\bar{u}}Q_k(x_k,\bar{u}) & \textit{with probablity}\ 1-\varepsilon_k\\ a\ \textit{uniformly random action in}\ U & \textit{with probablity}\ \varepsilon_k\end{cases} \tag{16}$$

Alternatively, Boltzmann exploration can be used to select an action u at step k with probability:

$$P(u\mid x_k)=\frac{e^{\frac{Q_k(x_k,u)}{\xi_k}}}{\sum_{\bar{u}}e^{\frac{Q_k(x_k,\bar{u})}{\xi_k}}} \tag{17}$$

in which the *temperature* $\xi_k\geq 0$ adjusts the randomness of the exploration. The reduction of temperature makes the decision maker greedy in the decision-making process. The exploration and the learning rate have significant effects on the performance of Q-learning [19].

Fig. 6 shows the performance of Q-learning with classic ε-greedy exploration for one recovery trajectory. Without random exploration, Q-learning is not guaranteed to converge and might iterate on a sub-optimal policy [30]. We have applied the Q-learning method to the

problem, defined in Section 3.3. Fig. 6 depicts the performance of Q-learning with ε-greedy exploration method. As Fig. 6 shows, the functionality at the community level drops irregularly due to the random exploration of the applied Q-learning method over the training steps. In this case, the decision-maker selects some actions randomly to explore the Q values for different states and actions. As mentioned before, the second condition to guarantee the convergence of Q-learning is that all state-action pairs should be visited frequently. In actuality, owing to the lack of data and time, the analyst is unable to satisfy this condition completely and should decide a repair action based on a partial-trained Q-learning algorithm. In this case, the agent might make an unjustifiable restoration decision. Hence, we do not recommend a random exploration for the post-disaster community recovery process in which the policymakers deal with public health and safety as well as expensive critical infrastructure systems. One potential alternative method in the literature is "optimism in the face of uncertainty" in which the decision maker commences with a value function that is larger than true returns [19]. Thus, greedy decision selection explores novel actions since the return assessments have been corrected downwards for any decisions already made. Moreover, safer exploration methods like deep Q-learning and safe Q-learning for the system under the risk of extreme events are recommended [30].

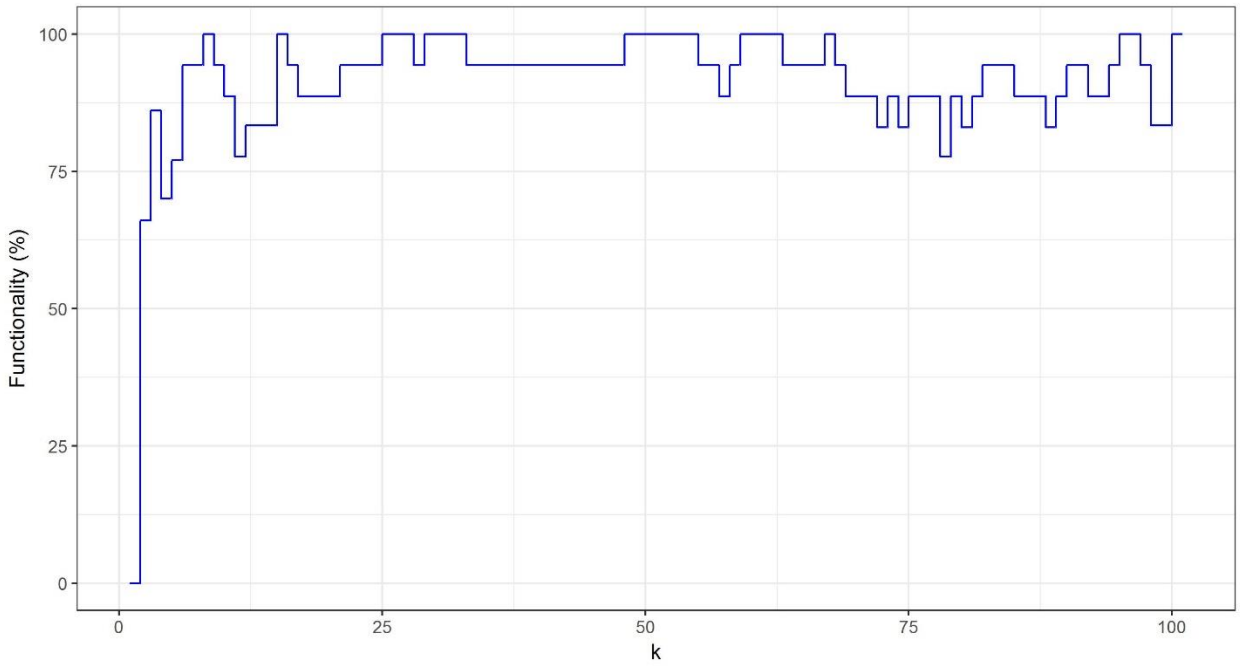

Fig. 6. Q-learning restoration

3.6.Policy iteration

In this section, another class of DP/RL methods— policy iteration - is considered. In this class of methods, the value functions of policies are computed to assess and improve the admissible policies [16]. As with Q-iteration methods, the policy iteration can be categorized as model-based and model-free. We first present model-based policy iteration in Section 3.6.1, followed by model-free policy iteration in Section 3.6.2.

3.6.1. Model-based policy iteration (policy evaluation for Q-functions)

Model-based policy evaluation for Q-functions utilizes reward and transition functions iteratively. Similar to Q-iteration mapping, define a policy evaluation of T^{π}: $\mathcal{O} \rightarrow \mathcal{O}$. For any arbitrary Q-function, this mapping function determines the right-hand side of the Bellman equation as follows:

$$[T^{\pi}(Q)](x,u) = \underset{x' \sim f(x,u,.)}{E} \left\{ \rho(x,\mathrm{u},\mathrm{x}') + \gamma \mathrm{Q}(x', \pi(x') \right\} \tag{18}$$

The algorithm updates an arbitrary Q_0^{π} iteratively as follows:

$$Q_{\ell+1}^{\pi} = T^{\pi}(Q_{\ell}^{\pi}) \qquad s.t. \qquad \left\| T^{\pi}(Q) - T^{\pi}(Q') \right\|_{\infty} \leq \gamma \left\| Q - Q' \right\|_{\infty} \tag{19}$$

For a discount factor of γ<1, the mapping function T^{π} has a unique fixed point (i.e., $Q^{\pi} = T^{\pi}(Q^{\pi})$). This iteration converges to Q^{π}. For small problems in which $X \times U$ is up to several thousand with finite states and action spaces, the linear system of Bellman equations can be solved to obtain Q^{π} [19].

Practically, the number of iterations required for the policy iteration method to converge is smaller than for Q-iteration methods. This does not necessarily mean that the Q-iteration methods are computationally more expensive than policy iteration owing to the exhaustive policy evaluation that is required for every individual policy iteration [19].

A desirable feature of policy iteration in the community resilience planning problems is that it can start with a policy. In reality, public and private entities usually have their specific

recovery policies. These policies can be based on entities' regulations, policymakers' judgments, and/or their analyses. The class of policy iterations can potentially begin with these policies and improve them to reach a strict optimal restoration policy asymptotically. A high-quality restoration policy with respect to the policy makers' preferences, provided by entities helps the policy iteration to converge quicker.

3.6.2. Model-free policy iteration (SARSA)

In the realm of model-free policy iteration methods, we focus on the most common method, called SARSA (**S**tate, **A**ction, **R**eward, (next) **S**tate, and (next) **A**ction). The required data tuples of SARSA is $(x_k, u_k, r_{k+1}, x_{k+1}, u_{k+1})$. Thus, like Q-learning, SARSA does not require an explicit model of the problem. It updates the Q-functions in the light of its required data tuple as follows;

$$Q_{k+1}(x_k, u_k) = Q_k(x_k, u_k) + \alpha_k [r_{k+1} + \gamma Q_k(x_{k+1}, u_{k+1}) - Q_k(x_k, u_k)] \tag{20}$$

in which $\alpha_k \in (0,1]$. The method is implemented in a manner very similar to Q-learning. However, the temporal difference (the term between brackets) in SARSA embeds the Q-value of the next possible state, whereas Q-learning considers the maximum. In other words, SARSA fulfills the policy evaluation on the followed (initial) policy. This is the difference between "on-policy" and "off-policy" methods. SARSA is called "on-policy" because it evaluates the recovery policy utilized to restore the community. Conversely, in "off-policy" methods like Q-learning, one policy is applied while another recovery policy is being evaluated [17]. In SARSA, as in Q-learning the decision-maker can use ε-greedy or Boltzmann exploration. However, safer exploration methods are again recommended.

Of greatest interest in community recovery management is the applicability of SARSA in large-scale problems. SARSA does not stall because the Q-function has not entirely converged. For example, in community recovery problems, this convergence might be extremely time-consuming not only because of the large dimensionality of the problem but also because of the lack of a high-quality policy to initiate the analysis. However, SARSA tries to improve the policy at hand before the full convergence. Another desirable property of SARSA is that it tries to improve the current policy after every sample owing to the greedy part of the method.

3.7. Computational cost

The computational cost of each algorithm represents the applicability of the method for large-scale problems. Hence, this section evaluates the complexities of the mentioned mode-based methods to assess their applicability for real communities that embed several real-size networks.

Let |.| define the cardinality of the argument set ".". In this study, $|X|$ and $|U|$ represent the number of states and actions that both are finite. In the Q-iteration method, presented in Section 3.5.1, in each iteration for a pair of (x,u) the cost of updating Q-value is $|X|^2\ |U|\ (2+|U|)$. Therefore, the total cost of L iterations is [19]:

$$L|X|^2\ |U|\ (2+|U|) \tag{21}$$

To reduce the total cost, L can be selected with applying a suboptimality bound of $\varepsilon_{QI}> 0$ in Eq. (14). In the policy evaluation, represented in Section 3.6., four functions of f, h, Q^{π}, and ρ are evaluated at each iteration. The cost of each policy evaluation for the L number of iterations is:

$$L|X|^2\ |U| \tag{22}$$

As mentioned in Section 3.6.1, the policy can be evaluated by solving the linear system of the Bellman equations. This procedure also requires computations of order $O(|X|^3\ |U|^3)$ [19].

For practical community resilience problems, $|X|$ and $|U|$ are enormous. Therefore, owing to the cost complexities of the Q-iteration and policy iteration, classic model-based methods of DP are quite impractical in a real-world context. Approximate Dynamic Programming (ADP) techniques usually are required, described in the next section.

4. Approximate dynamic programming techniques

4.1. ADP fundamentals

As alluded to previously, the DP methods of Q-iteration and policy iteration are guaranteed to generate optimal solutions for Markov Decision Problems (MDPs) and semi-Markov Decision Problems (SMDPs) [17]. However, they are often computationally intractable and for real-sized communities a complete solution is impossible. When the number of state or

action variables increases, the exponential increase in computational demand often is referred to as the *curse of dimensionality* [16]. Furthermore, computations of the transition probabilities and rewards in an analytical form for realistic community resilience assessment is often intractable. In actuality, *all* information to describe a large-scale community is unavailable in advance, or maybe unavailable until shortly before a decision is required. Owing to the constraint of the amount of time and complexities, the theoretical model of a community is often unachievable. Therefore, when used for modeling and managing community recovery, DP is said to be plagued by the twin curses of dimensionality and modeling . Additionally, policymakers are required to make a rational decision immediately following the occurrence of a hazard event. These stringent time constraints make a solution even more difficult. The on-line simulation-based methods are one way to circumvent these twin curses.

In on-line methods, optimal decisions are identified only for the visited states in the real-world, thereby eliminating unnecessary computational cost on the unreached states. On the other hand, in off-line computations, the policy is calculated for all the states and stored. Thereafter, the policymaker chooses an optimal action from the stored policy based on the observed evolution of the community. Q-learning and SARSA are on-line methods since they only deal with the reached state at the instant, while the Q-iteration and policy iteration are off-line methods.

Practically speaking, policymakers should expect a suboptimal recovery strategy that balances a desire for low computation with sufficient performance. These stochastic modeling and algorithmic strategies for solving large and complex problems fall under the broad umbrella of approximate dynamic programming (ADP). Although ADP techniques overcome the curse of dimensionality, the more valuable objective of ADP is *learning what to learn, and how to learn it, to make better decisions over time* [31]. ADP has been used in a wide range of decision-making problems in control theory [32], operations research [33], and reinforcement learning/artificial intelligence [17]. .

The fundamental idea of ADP is to approximate the true value functions (V or Q-functions) reasonably with different statistical methods. ADP methods often step forward in time instead of the classic backward-stepping DP (cf. Section 3.5) or the combination of stepping forward and sweeping backward to calculate the value state. To this end, numerous approaches have been developed to assess the value function approximation. In this section, we summarize

(see Figure 7) the main approaches of ADP with emphasis on the methods that are either utilized or are applicable for civil infrastructure management.

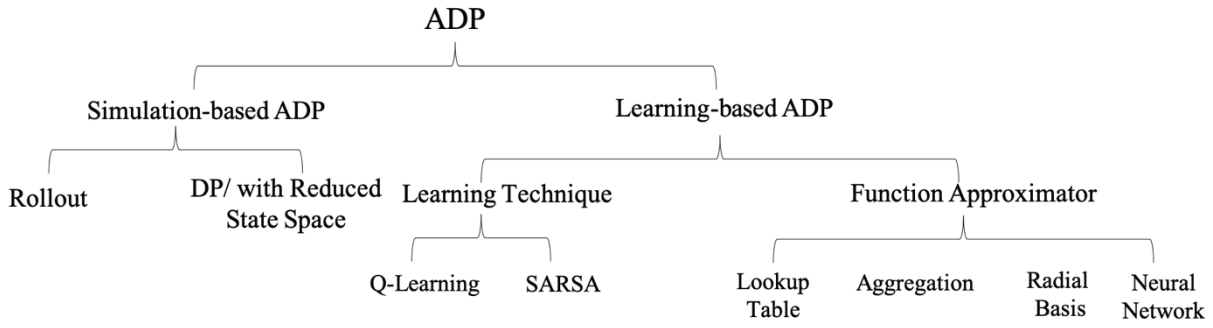

Fig. 7. A block diagram of ADP classification, adopted from [34].

A learning-based ADP method utilizes either a learning technique or functional approximation to approximate state-action rewards (Q functions). Q-learning and SARSA, discussed in Sections 3.5.2 and 3.6.2, are examples of learning-based ADP. A learning-based ADP method requires minimum knowledge of the problem. As mentioned before, the random exploration of these techniques can be too risky for the community's well-being. Memazrzadeh and Pozzi [30] addressed this issue proposing a model-based safe exploration method called Safe Q-learning. Andriotis and Papakonstantinou [35] proposed the Deep Centralized Multi-agent Actor-Critic (method to provide efficient life-cycle policies over the life cycle of a single infrastructure system.

In contrast, function approximators utilize lookup tables (the simplest form), multilevel aggregation, or parametric and non-parametric approximator like radial basis functions and neural networks. The limitations of the lookup method are discussed in Ref. [31] and it is not recommended for huge state and action spaces problems like community recovery management. Multilevel aggregation tries to represent the state space with a coarser structure. In this case, the challenge is determining the right level of aggregation. Some researchers proposed that a varying level of aggregation with iteration is suitable for many problems [31]. Fereshtehnejad and Shafieezadeh [26] applied this technique through the counting process to compute the optimal management of a bridge with five components. Their method appears to be applicable only to similar small problems; at the community level, the agent must model the community at an extremely coarse granularity, which leads to a loss of accuracy in representing the original

problem. Furthermore, this method assigns the same action to the elements in the same condition level; this assignment is unreasonable in community resilience assessment because infrastructure elements may be in different states of damage following a hazard and only a limited number of RUs may be available during recovery. ,.

One of the most prevalent methods in ADP is radial basis functions. These functions capture prominent quantities of the state variable and construct a surrogate model around these quantities. In the realm of civil engineering, Medury and Madanat [36] utilized this technique to provide optimal maintenance, rehabilitation and replacement (MR&R) policies for a system of facilities over a planning horizon. While basis functions are easy to implement, their selection may affect the quality of decisions significantly [19, 31]. Alternatively, non-parametric techniques like kernel-based approximators can be applied. Unlike in the parametric case, the numbers of parameters and the form of the approximator are derived completely from the available data. Therefore, their performance tends to be weak when there is a lack of sufficient data.

Note that the learning-based ADP like Q-learning can also benefit from a suitable approximator. For example, Q-learning (Eq. (15)) estimates the Q-value of each state-action pair separately. The performance of this method can be poor if there are insufficient data for some states, and a suitable approximator can support Q-learning by making reasonable decisions in neighboring states of available Q-values. This method of *generalization* can potentially enhance the performance of learning-based ADP techniques [19]. The selection of an accurate model and sufficient data are always of concern when using parametric or non-parametric approximators. The class of simulation-based ADP called rollout does not require an explicit approximator, and thus is of special interest in assessing community resilience under extreme natural hazards.

4.2. Rollout

In community resilience problems, planners and policymakers usually have models of infrastructure systems at some level. In the presence of such community models, simulation-based methods are recommended (cf Figure 7) [34]. Hence, this type of ADP can potentially outperform other methods by leveraging the power of the model. Rollout belongs to a class of simulation-based methods in which the state and action space reduction is possible and state-action costs are approximated with the help of simulation [34]. Rollout computes the near-

optimal recovery process in an on-line manner (cf. Section 4.1). It can be categorized as an on-line implementation of the policy improvement step of policy iteration via simulation [16, 37]. Owing to its on-line character, rollout can be easily implemented regardless of the size of the state space; hence, it is suitable for large state space problems such as community resilience assessment. Rollout is initiated by a base policy, which can be random, based on experts' judgments, based on importance analyses, or above all the recovery strategies of the regional entities [14]. Thereafter, rollout successively improves the existing model, exploiting its advantages and improving current policies for the community and human considerations.

Rollout evaluates the Q-value, ($\hat{Q}^{\pi_b}(x,u)$), by simulating a number N_{MC} of trajectories. The trajectories are generated by the base policy of π_b with the length of K.

$$\hat{Q}^{\pi_b}(x,u)=\frac{1}{N_{MC}}\sum_{i_0=1}^{N_{MC}}\left[\rho(x,u,x_{i_0,1})+\sum_{k=1}^{K}\gamma^k\rho(x_{i_0,k},\pi_b(x_{i_0,k}),x_{i_0,k+1})\right] \qquad s.t. \quad x_{i_0,k+1}\sim f(x_{i_0,k},\pi_b(x_{i_0,k}),.) \quad (23)$$

The rollout policy can be shown to outperform the base policy, given that improvement is feasible [37]. Furthermore, while the rollout policy is not necessarily an optimal, it provides a framework to support the policymakers with better informed decisions than their current strategies (i.e., π_b).

Nozhati et al. [14, 39] applied the rollout algorithm to compute near-optimal recovery strategies for the electrical power (EPN) and potable water (WN) networks of the Gilroy, CA community, given the occurrence of a *Mw* 6.9 earthquake on the San Andreas Fault at an epicentral distance of approximately 12 km from the center of the city.. This stochastic decision-making problem was modeled in the context of MDP. Fig. 8 depicts the EPN and WN of Gilroy. Details of the analysis, including the modeling of the hazard and the interdependent networks can be found in [14, 38, 39]. As mentioned in Section 3, this model is not necessarily an analytical model (e.g., TPM) and can be a generative model. In fact, an analytical model is not achievable for the real-case problems (curse of modeling). This is a main motivation for the simulation-based representation of MDP.

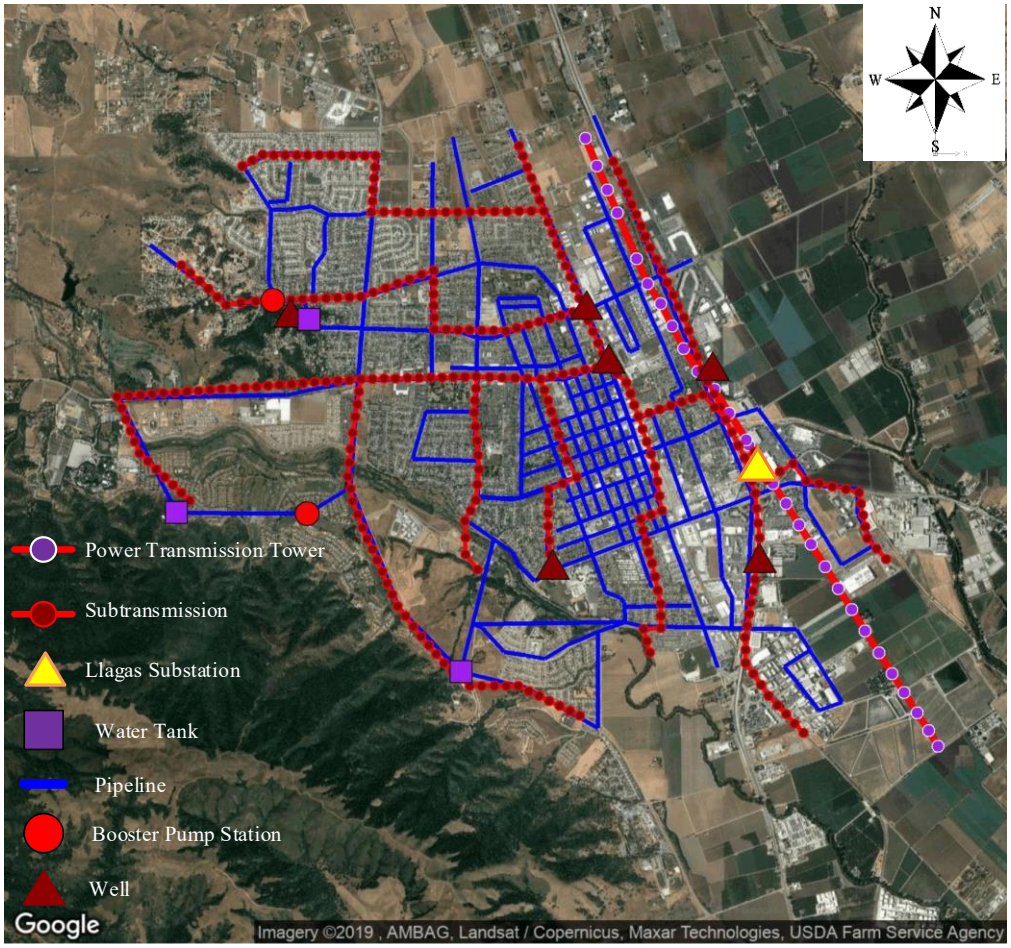

Fig. 8. The modeled electrical power and water network of Gilroy

4.3. Simulation-based representation of MDP

The simulation-based representation of an MDP serves well for large state and action spaces, which is a characteristic feature of community resilience problem [17]. A simulation-based representation of an MDP is (X, U, P, ρ, γ) where cardinalities $|X|$ and $|U|$ are large. The tuple P is the generative model (or simulator) of the problem that mimics the behavior of the community and the networks. Given the pair of (x,u), the simulator P provides the new random state of x_{t+1}. Hence, there is no need for the analytical model. Several approaches to model the networks and their interdependencies, ranging from the flow analysis to graph theory and fragility-based modeling of a community over the recovery process, are available [6, 8, 14, 40]. These methods in the literature can potentially serve as a simulator for the simulation-based MDP. A key factor common to all mentioned methods is the repair time, a random variable

which depends on the level of damage at each damaged location and plays a significant role in modeling post-disaster recovery.

In preemptive assignment, the policymaker can reassign RUs to different locations, provided that reassignment receives more rewards. Therefore, the non-preemption is a special case of preemptive scheduling. The decision-maker can implement the reassignment if at least one of the RUs is available. Once a damaged component is repaired, there may be cascading effects throughout the community. From the MDP perspective, the underlying Markov chain would transition to a new state. The time spent at a particular state before transitioning to another state is called the *sojourn* time (a.k.a. *dwelling* or *holding* time) [41, 42]. The statistical property of this sojourn time and the distribution of residual repair time in the succeeding states depends on the initial repair time distributions. Nozhati et al. [14, 39] assumed the repair times are described by exponential distributions. With this assumption, the sojourn times are also exponentially distributed:

$$t_1(t) \sim Exp(\lambda_1),....,t_n(t) \sim Exp(\lambda_n) \rightarrow \hat{t}_{sj}(t) = \min(t_1(t),...,t_n(t)) \sim Exp(\sum_{i=1}^{n} \lambda_i) \quad (24)$$

where t_i(t) is the random repair time of the i^{th} component and $\hat{t}_{sj}(t)$ is the sojourn time of the community at time *t.* As mentioned in Section 3.3, the components' damage state *history* following the hazard forms the MDP state at the community level. However, owing to the memoryless property of the exponential distribution, the damage state and the sojourn time at each time suffice to simulate the future evolution of the problem. Strictly speaking, the exponential distribution assumption and consequently the memoryless property of the sojourn time satisfy the Markovian property of the underlying process [41, 42].

The methods presented above remain applicable with other distributions, with some cautions. For example, HAZUS-MH [43] proposes normal distributions for the repair time with the means and standard deviations proportional to the level of damage. The conditional residual repair time is a conditional normal distribution with a smaller standard deviation, leading to the conclusion the fact that *lower damage states leads to lower dispersion in residual repair time*. However, the normal distribution is supported by the entire real line, and thus is inappropriate either for the system's lifetime or restoration time [44]. Although the negative part often can be ignored, given that the coefficient of variation (COV) is much smaller than one [44], this is not the case for the

community recovery process because of high dispersions in the repair time,. As a result, different distributions like lognormal or Weibull distributions [7, 10] often have been used. Note that in these cases, the sojourn time is no longer exponential distribution and the underlying decision process is semi-Markov rather than Markov.

4.4. Rollout results

The reward function in the study of Nozhati et al. [39] is based on providing electricity and water to the maximum number of people in Gilroy, CA in the shortest possible time. This objective function mimics a common resilience index in the literature, based on the area under the curve describing recovery of functionality [14]:

$$\textbf{Resilience} = \int_0^{T_{LC}} \frac{Q(t)}{T_{LC}} dt \tag{25}$$

where $Q(t)$ is the functionality of a system at time, t, and T_{LC} is the control time of the system during the recovery.

Therefore, as Fig. 9 shows, rollout is aimed at providing a strategy with a larger area under the curve than the given base policy. If the base policy is not strictly optimal, the rollout policy always outperforms the base policy. The resilience index (Eq 25) for the base policy in Fig. 9 is 22,395. It means that if the policymaker follows the base policy 22,395 people benefit from electricity and water per day, while 24,224 number of people have electricity and water provided that the policymaker follows the rollout strategy to recover the EPN and WN. Fig. 9 also highlights the lookahead property of the rollout approach, mentioned in Section 3. Rollout identifies conservative repair decisions over the first 15 days following the earthquake, while it improves the base policy when the entire recovery horizon is considered. The lookahead property of rollout in this study is a single-step lookahead property, in which the agent tries all possible decision at a single time slot and then the base policy is simulated. With a sufficient computational budget, the agent can increase the step of lookahead property to reach a higher-qualified rollout policy. Ref. [16] describes more details and computational issues in limited lookahead policies.

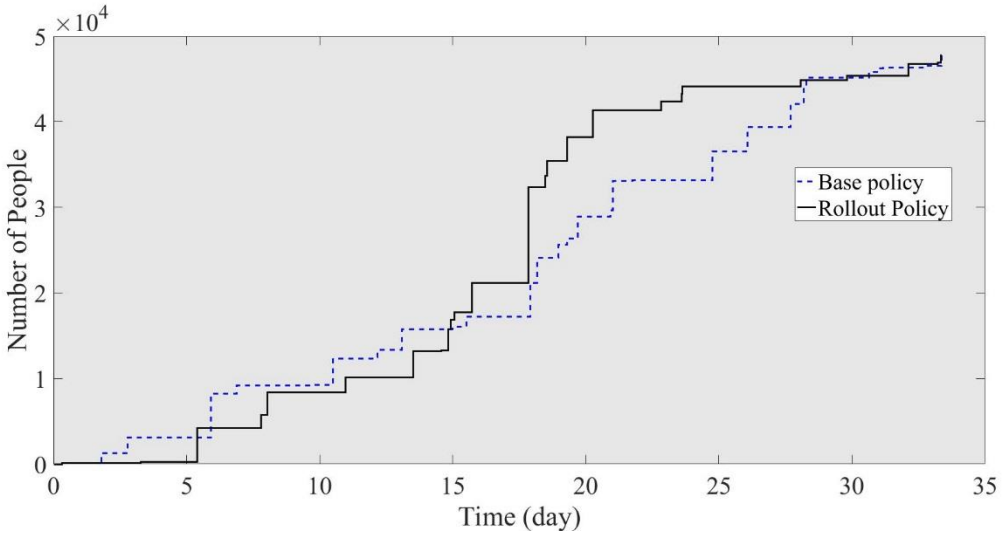

Fig. 9. The performance of base and rollout policy to provide electricity and water to people

Like any optimization methodology, rollout has some limitations. In the stochastic case, rollout can be computationally expensive, and if the required number of points to estimate the value function is increased, rollout can be intractable. In these cases, a suitable approximator to estimate the Bellman equation may be computationally less expensive than rollout. Furthermore, the performance of rollout depends on the quality of the base policy. Since the primary goal of rollout is to enhance the base policy than to be close to the strict optimal, a naive base policy can decrease the quality of the rollout policy. This issue can be addressed to some extent by initiating the process with a reasonable base policy and using a multistep lookahead property and.

5. Complementary discussion

We now discuss different points and issues that policymakers and practitioners might encounter by using the mentioned methodologies or formulating the recovery management in the optimal control context.

5.1. Reward function and encoding risk

Designing a comprehensive reward function is of central importance in the optimal control context. This reward function must reflect policymakers' preferences. Therefore, reward

functions may be multi-objective in nature, with conflicting objectives. Furthermore, the methods described herein are based on the expected value of the reward or cost in the decision-making process. The expected value may not always be the most appropriate measure to represent the preferences and risks in community resilience assessment. In the context of DP and RL, the methods of minimax and maximin approaches can be alternatives [39]. These standard methods can address different risk behaviors by replacing mean-based optimization with worst-case optimization. However, the methods cannot reflect risk aversion behavior in public decision-making. More comprehensive methods like dual stochastic programming can be an option [45]. Nevertheless, the maximization of expected reward can be potentially appropriate on condition that the reward function is comprehensively defined so that it encodes the risk preferences of community stakeholders [16].

One of the strengths of the ADP and RL methods is that they can be enhanced significantly by expert judgment and domain knowledge related to civil infrastructure management For example, Nozhati et al. [14] took advantage of the series arrangement of the EPN and WN to reduce the action space at each time slot. Therefore, those RL methods that are model-free can benefit from prior knowledge. Sometimes the prior knowledge can be encoded in the reward function [19].

5.2. State augmentation

Sometimes the community states and policymakers' decisions impact future states with some time lag. For example, if the system state x_{k+1} depends not only on the preceding state x_k and action u_k but also on earlier states and decisions. This issue can be addressed by state augmentation, in which the state at time k is enlarged to embed all the data that is available to the policymaker at time k so that it can subsequently be used to advantage in making the decision of u_k. State augmentation and definition of new composite states can be found in Ref. [16]. This case can occur in community resilience problems when policymakers deal with several physical systems. Suppose that two interdependent networks A and B in a community (e.g., EPN and WN) are represented by interdependent Markov chains that are different in terms of state variables and state-space cardinalities over the recovery period. The Markovian property of the marginal processes of X and Y for the individual networks does not necessarily guarantee that their joint process (X, Y) is Markovian [46]. Therefore, in community resilience planning, in addition to the

current states of individual networks, the policymakers' must be aware of including *memory* from the past transitions to completely and accurately model the stochastic dynamics of the community [46].

5.3.Evaluating an ADP strategy

There are several methods a policymaker might use to evaluate the rationality of an ADP solution [31]. The policymaker might *compare the solution with an optimal MDP*, assuming that one can be obtained In other cases, the policymaker might simplify the problem to the point that it can be solved by Q-iteration or policy-iteration methods. The policymaker can *compare the solution with an optimal deterministic problem* [14], in which the common problem of state explosion does not arise. Finally, the policymaker can *compare the solution with a myopic strategy*[1], since it is known that myopic (greedy) solutions are less costly than those with the look-ahead property and often provide a good sense about the accuracy of the solution [16].

6. Summary and conclusions

Approximate dynamic programming and reinforcement learning techniques are powerful algorithmic strategies that can be utilized in a wide range of problems involving decision-making under uncertainty. These methods are largely unexplored in the crucial problem of planning for community resilience under extreme natural events. This study has reviewed and appraised common ADP and RL methods, with regard to their usefulness in community resilience assessment and has led to the following observations:

The Q-iteration method is suitable for small networks (e.g., up to ten components). In this case, it provides an efficient solution if a model of the community is available. Although it is computationally expensive, it can provide an optimal strategy. The model-based policy iteration method is also feasible for small communities. This method can evaluate and improve the current policy and provide optimal scheduling. RL techniques, including Q-learning and SARSA, do not require any prior knowledge of the community and can be suitable methods when a model of the community is unavailable. However, the process of exploration embedded

[1] Myopic decisions can be made by policymakers in different phases and they are not restricted to the recovery process. The word *myopia* is used by scientists to represent the inclination to overweight short-term effects or consequences in comparison to long-term impacts concerning disasters. Myopia can potentially influence policymakers to undervalue the low-probability/high-consequence catastrophes [47].

in these methods can be highly risky for community restoration planning. Furthermore, a minimum number of steps is required to converge to optimality. Generalization of these methods with a suitable approximator is recommended if they are to be applied in community resilience. Finally, a class of ADP methods called rollout possesses several desirable features for the problem of community recovery management. It can handle large state space problems; it leverages current recovery strategies by using them to initiate the rollout analysis, and it enables various sources of uncertainties to be considered. On the other hand, it can be computationally expensive in the high level of stochasticity.

One final note: we do not claim that the methods summarized in this paper are a panacea for modeling community recovery. To apply or design a methodology in the context of community resilience, a policymaker must consider three points; 1) *reliability*; the method should provide the correct solutions with defined errors, 2) *productivity*; the method should implement the intended solutions rationally and orderly in a timely fashion; and 3) *simplicity*, the method should allow policymakers and community stakeholders to understand community resilience and strategies for its enhancement. No decision analysis method should be expected to be universally applicable in addressing complex community resilience assessment issues.

Acknowledgements

The research herein has been funded by the National Science Foundation under Grant CMMI-1638284. This support is gratefully acknowledged. Any opinions, findings, conclusions, or recommendations presented in this material are solely those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

[1] Abolghasemi H, Radfar MH, Tabatabaee M, Hosseini-Divkolayee NS, Burkle FM. Revisiting blood transfusion preparedness: experience from the Bam earthquake response. Prehospital Disaster Med 2008; 23(05):391–4.

[2] Woods DD. Four concepts for resilience and the implications for the future of resilience engineering. Reliability Engineering and System Safety 2015; 141:5–9.

[3] Koliou M, van de Lindt J, McAllister TP, Ellingwood BR, Dillard M, Cutler H. A Critical Appraisal of Community Resilience Studies: Progress and Challenges.

Sustainable and Resilient Infrastructure 2018; 3(1). doi.org/10.1080/23789689.2017.1418547.

[4] Bruneau M, Chang SE, Eguchi RT, Lee GC, O'Rourke TD, Reinhorn AM, Von Winterfeldt D. A framework to quantitatively assess and enhance the seismic resilience of communities. Earthquake Spectra 2003; 19(4): 733-752.

[5] Sharma N, Tabandeh A, Gardoni P. Resilience analysis: A mathematical formulation to model resilience of engineering systems. Sustainable and Resilient Infrastructure 2018; 3(2): 49-67.

[6] Guidotti R, Gardoni P, Rosenheim N. Integration of physical infrastructure and social systems in communities' reliability and resilience analysis. Reliability Engineering and System Safety 2019; 185: 476-492.

[7] Lin P, Wang N. Stochastic post-disaster functionality recovery of community building portfolios I: Modeling. Structural Safety 2017; 69: 96-105.

[8] Masoomi H, van de Lindt JW, Peek L. Quantifying socioeconomic impact of a Tornado by estimating population outmigration as a resilience metric at the community level. J. Struct. Eng 2018; 144(5): 04018034.

[9] Masoomi H, van de Lindt JW. Restoration and functionality assessment of a community subjected to tornado hazard. Structure and Infrastructure Engineering 2018; 14(3): 275-291.

[10] Masoomi H. A resilience-based decision framework to determine performance targets for the built environment (Doctoral dissertation, Colorado State University) 2018.

[11] Zhang W, Nicholson C. A multi-objective optimization model for retrofit strategies to mitigate direct economic loss and population dislocation. Sustainable and Resilient Infrastructure 2016; 1(3-4): 123-136.

[12] Gomez C, Baker JW. An optimization-based decision support framework for coupled pre-and post-earthquake infrastructure risk management. Structural Safety 2019; 77:1-9.

[13] Xu M, Ouyang M, Mao Z, Xu X. Improving repair sequence scheduling methods for postdisaster critical infrastructure systems. Computer-Aided Civil and Infrastructure Engineering 2019; 34(6): 506-522.

[14] Nozhati S, Sarkale Y, Ellingwood B, Chong EKP, Mahmoud H. Near-optimal planning using approximate dynamic programming to enhance post-hazard community resilience management. Reliability Engineering and System Safety 2019; 18: 116-126.

[15] Gosavi A. Simulation-based optimization. Berlin: Springer 2015.

[16] Bertsekas, D. P. Dynamic programming and optimal control (Vol. 1, No. 2). Belmont, MA: Athena scientific 1995.

[17] Sutton RS, Barto AG. Introduction to reinforcement learning (Vol. 2, No. 4). Cambridge: MIT press 1998.

[18] Cimellaro GP, Solari D, Arcidiacono V, Renschler CS, Reinhorn, AM, Bruneau M. Community resilience assessment integrating network interdependencies. In Proc. Tenth US National Conf. on Earthquake Engineering (10NCEE) 2014.

[19] Busoniu L, Babuska R, De Schutter B, Ernst D. Reinforcement learning and dynamic programming using function approximators. CRC press 2017.

[20] Zhang X, Mahadevan S, Sankararaman S, Goebel K. Resilience-based network design under uncertainty. Reliability Engineering and System Safety 2018, 169: 364-379.

[21] Hillier FS. Introduction to operations research 9th edition. Tata McGraw-Hill Education 2001.

[22] Nozhati S, Rosenheim N, Ellingwood BR, Mahmoud H, Perez M. Probabilistic framework for evaluating food security of households in the aftermath of a disaster. Structure and Infrastructure Engineering 2019; 15(8): 1060-1074.

[23] Nozhati S, Ellingwood BR, Mahmoud H, van de Lindt JW. Identifying and analyzing interdependent critical infrastructure in post-earthquake urban reconstruction. In 11th US National Conference on Earthquake Engineering: Integrating Science Engineering and Policy 2018.

[24] Ouyang M, Duenas-Osorio L. Multi-dimensional hurricane resilience assessment of electric power systems. Structural Safety 2014; 48: 15-24.

[25] Ouyang M, Dueñas-Osorio L, Min X. A three-stage resilience analysis framework for urban infrastructure systems. Structural safety 2012; 36: 23-31.

[26] Fereshtehnejad E, Shafieezadeh A. A randomized point-based value iteration POMDP enhanced with a counting process technique for optimal management of multi-state multi-element systems. Structural Safety 2017; 65: 113-125.

[27] Meidani, H, Ghanem R. Random Markov decision processes for sustainable infrastructure systems. Structure and Infrastructure Engineering 2015; 11(5): 655-667.

[28] Bellman R. Dynamic programming. Science 1966; 153(3731): 34-37.

[29] Sarkale Y, Nozhati, S, Chong, EKP, Ellingwood B. R., Mahmoud H. Solving Markov decision processes for network-level post-hazard recovery via simulation optimization and rollout. In 2018 IEEE 14th International Conference on Automation Science and Engineering (CASE) 2018; 906-912.

[30] Memarzadeh M, Pozzi M. Model-free reinforcement learning with model-based safe exploration: Optimizing adaptive recovery process of infrastructure systems. Structural Safety 2019; 80: 46-55.

[31] Powell WB. What you should know about approximate dynamic programming. Naval Research Logistics (NRL) 2009; 56(3): 239-249.

[32] Bertsekas DP, Tsitsiklis JN. Neuro-dynamic programming (Vol. 5). Belmont, MA: Athena Scientific 1996.

[33] Birge JR, Louveaux F. Introduction to stochastic programming. Springer Science & Business Media 2011.

[34] Katanyukul T, Duff WS, Chong EKP. Approximate dynamic programming for an inventory problem: Empirical comparison. Computers & Industrial Engineering 2011; 60(4): 719-743.

[35] Andriotis CP, Papakonstantinou KG. Managing engineering systems with large state and action spaces through deep reinforcement learning. Reliability Engineering and System Safety 2019; 191:106483.

[36] Medury A, Madanat S. Incorporating network considerations into pavement management systems: A case for approximate dynamic programming. Transportation Research Part C: Emerging Technologies 2013, 33:134-150.

[37] Bertsekas DP. Rollout algorithms for discrete optimization: A survey. Handbook of combinatorial optimization 2013; 2989-3013.

[38] Nozhati S, Ellingwood BR, Mahmoud H. Understanding community resilience from a PRA perspective using binary decision diagrams. Risk analysis 2019.

[39] Nozhati S, Sarkale Y, Chong, EKP, Ellingwood, BR. Optimal Stochastic Dynamic Scheduling for Managing Community Recovery from Natural Hazards. Reliability Engineering and System Safety 2019. https://doi.org/10.1016/j.ress.2019.106627.

[40] He X, Cha EJ. Modeling the damage and recovery of interdependent critical infrastructure systems from natural hazards. Reliability Engineering and System Safety 2018; 177:162-175.

[41] Aslett LJ. MCMC for inference on phase-type and masked system lifetime models (Doctoral dissertation, Trinity College Dublin) 2012.

[42] Ibe O. Markov processes for stochastic modeling. Academic Press 2013.

[43] Department of Homeland Security Emergency Preparedness and Response Directorate, FEMA, Mitigation Division. Multi-hazard loss estimation methodology, earthquake model: HAZUS-MH MR1, advanced engineering building module. Washington DC; 2003.

[44] Limnios N. Fault trees. John Wiley & Sons 2013.

[45] Shapiro A. Analysis of stochastic dual dynamic programming method. European Journal of Operational Research 2011; 209(1): 63-72.

[46] Rahnamay Naeini M. Stochastic dynamics of cascading failures in electric-cyber infrastructures (Doctoral dissertation, University of New Mexico) 2014. Available from https://digitalrepository.unm.edu/ece_etds/213

[47] Applied Technology Council, & Applied Technology Council. Critical Assessment of Lifeline System Performance: Understanding Societal Needs in Disaster Recovery. US Department of Commerce, National Institute of Standards and Technology 2016.