**Proceedings of the ASME 2021**
**International Design Engineering Technical Conferences and**
**Computers and Information in Engineering Conference**
**IDETC/CIE2021**
**August 17-20, 2021, Virtual, Online**

# DETC2021-70985

# UNDERSTANDING RESILIENCE OPTIMIZATION ARCHITECTURES WITH AN OPTIMIZATION PROBLEM REPOSITORY

**Daniel Hulse***
NASA Ames Research Center †
Moffett Field, California, 94035

**Hongyang Zhang, Christopher Hoyle**
Oregon State University
Corvallis, Oregon, 97330

## ABSTRACT

*Optimizing a system's resilience can be challenging, especially when it involves considering both the inherent resilience of a robust design and the active resilience of a health management system to a set of computationally-expensive hazard simulations. While prior work has developed specialized architectures to effectively and efficiently solve combined design and resilience optimization problems, the comparison of these architectures has been limited to a single case study. To further study resilience optimization formulations, this work develops a problem repository which includes previously-developed resilience optimization problems and additional problems presented in this work: a notional system resilience model, a pandemic response model, and a cooling tank hazard prevention model. This work then uses models in the repository at large to understand the characteristics of resilience optimization problems and study the applicability of optimization architectures and decomposition strategies. Based on the comparisons in the repository, applying an optimization architecture effectively requires understanding the alignment and coupling relationships between the design and resilience models, as well as the efficiency characteristics of the algorithms. While alignment determines the necessity of a surrogate of resilience cost in the upper-level design problem, coupling determines the overall applicability of a sequential, alternating, or bilevel structure. Additionally, the application of decomposition strategies is dependent on there being limited interactions between variable sets, which often does not hold when a resilience policy is parameterized in terms of actions to take in hazardous model states rather than specific given scenarios.*

## 1 INTRODUCTION

There is an increasing interest in incorporating resilience in the design of complex engineered systems to minimize the impact of hazards which may inevitably occur during the operation of the system [1, 2]. Resilience characterizes the system's dynamic hazard response, which can be defined in terms of resistance, absorption, restoration, and recovery [3], the system's ability to prevent and mitigate hazards [4], or system-specific measures [5–7]. To account for these resilience attributes while designing a system, many decision frameworks, including value modelling [8, 9], multiobjective decision analysis [10, 11], and expected cost modelling [12–14], have been developed to enable one to trade the resilience of a given concept with other design considerations, such as performance and efficiency.

Using these decision-making frameworks, mathematical optimization techniques can be used to automatically explore the design space and find the set of variables which optimally balances design, operational, and resilience costs. General resilience optimization formulations have presented it as a sequential problem: first allocating resilience to subsystems and then optimizing the reliability and health management of those systems to achieve the required resilience [15, 16]. More recent work has formulated a two-stage optimization problem in which

*Address all correspondence to this author.

the system's control response is optimized before and after a set of hazards occur [17].

However, applications of resilience optimization often use approaches and architectures more specific to the problem than these general formulations. The most often-used architecture is the two-stage optimization approach, which has origins in stochastic programming [18–20] and is generally used for the resilient design of infrastructure networks while accounting for post-disaster reconfiguration and planning [21–23]. Bilevel approaches, which have their origin in the optimal solution of competitive games [24, 25] analogous to the resilience optimization problem [26], have also been used in the design of power stations [27], service centers [28], and solution of energy allocation problems [29], when the resilience system response over a set of scenarios must be specified up-front. Resilience optimization is additionally often used to design prognostics and health management systems using different architectures, including the integrated optimization of health management, system design, and mission [30], power plant condenser parameter and maintenance policy design [31], and sensor network design [32–34].

However, despite these many frameworks and applications, there has been less study into how characteristics of the resilience optimization problem formulation impact the development of solution strategy and/or choice of optimization architecture. In the broader field of optimization and multidisciplinary design optimization [35], the formulation of test problems has been key to developing [36] and studying the effectiveness of optimization algorithms [37, 38] and architectures [39–41]. Thus, there is an opportunity develop example resilience optimization problems to compare, study, and better understand optimization approaches.

In prior work, the authors identified the integrated optimization of design, operations, and contingency management as a type of codesign problem and used an integrated drone design architecture, mission, and flight resilience model to compare all-in-one, sequential, and bilevel architectures for resilience optimization [42]. While this comparison constituted a first step into understanding the comparative advantages of resilience optimization approaches, it was limited because it only considered a narrow set of architectures on a single problem, which limited its ability to inform general recommendations about the applicability of approaches to new problems.

### 1.1 Contribution

To resolve these limitations, this work develops a repository of resilience optimization problems and uses it to demonstrate and compare optimization approaches. To advance this goal, this work collects and categorizes previously-developed resilience optimization and presents three additional problems:

1. a notional resilience model in which a system's fault recovery is optimized with its operational profile,
2. a pandemic response model in which response thresholds are optimized to minimize infections and economic burden, and
3. a cooling tank model where design buffer and reconfiguration are optimized for performance and fault mitigation.

Taking insights from the optimization of these problems and the previously-developed models in the repository, this work then develops an overall understanding of the applicability of optimization architectures and decomposition strategies to resilience optimization problems. To contextualize this work, Section 2 describes the general resilience optimization problem and optimization architectures used in this work. Section 3 then describes the repository, along with the three newly-developed problems. Overall insights gained in the development of the repository are then outlined in Section 4, with conclusions in Section 5.

## 2 BACKGROUND

To understand the resilience optimization problem and the strategies used in the optimization problem repository, this section presents the general formulation of the resilience optimization problem used throughout the repository (Section 2.1) and optimization architectures and decomposition structures used in the repository (Section 2.2).

### 2.1 Resilience Optimization

Previously-presented cost formulations of the resilience optimization problem balance the cost of hazardous scenarios against the design and operational costs (e.g., manufacturing and performance). This enables one to find the variable values which simultaneously minimize the impact of hazards as it effects the overall merit of the design. This problem may be stated:

$$\begin{aligned} &min_{\mathbf{x}}\ C_{D/O}(\mathbf{x}) + C_R(\mathbf{x}) \\ &where\ C_R = \sum_{s \in S} n * r_s * C_s(\mathbf{x}) \end{aligned} \tag{1}$$

where $\mathbf{x}$ is the design vector, $C_{D/O}$ is the cost of design and/or operations, and $C_R$ is the cost of hazards, which is taken as an expectation over the set of scenarios $S$, where $r_s$ is the rate of the scenario, $C_s$ is the cost of the scenario, and $n$ is the life of the system. Depending on the intent of the problem formulation, the design vector $\mathbf{x}$ in the resilience optimization problem can take different forms. To distinguish these formulations, consider that the design vector $\mathbf{x}$ may be made up of two components: $\mathbf{x_{D/O}}$, the design and operational variables which define the inherent resilience of the system (e.g., design buffer), and $\mathbf{x_R}$, the resilience variables which define the actions the system takes over the set of hazardous scenarios–the active resilience of the system. There are thus three types of resilience optimization problems, as shown in Figure 1: Resilience-based Design Optimization (RDO), Resilience Policy Optimization (RPO), and Integrated Resilience Optimization (IRO).

In RDO, the design and operations of the system $\mathbf{x_{D/O}}$ are optimized as decision variables, resulting in a design and/or mis-

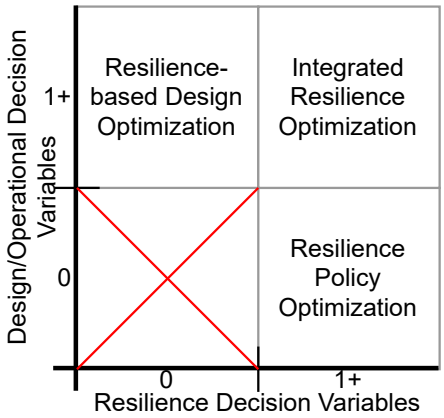


**FIGURE 1**. RESILIENCE OPTIMIZATION FORMULATIONS

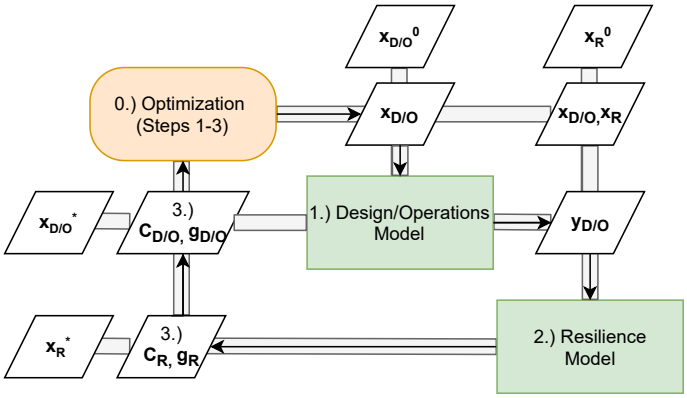


**FIGURE 2**. ALL-IN-ONE OPTIMIZATION ARCHITECTURE

sion profile which is inherently resilient to faults (see: [32–34, 43]). In RPO, the resilience variables $\mathbf{x_R}$ are optimized as decision variables, resulting in an optimal contingency management policy over the set of resilience scenarios for a given system design (see: [44–47]). In IRO, the design/operational variables $\mathbf{x_{D/O}}$ and resilience variables $\mathbf{x_R}$ are optimized together, resulting in an optimally-resilient set of design/operational and resilience variables [42]. Because of the interacting design and operational models, dedicated optimization architectures are most often used in IRO formulations (e.g., [15, 42]), while RPO and RDO problems are typically solved in a single-level.

## 2.2 Integrated Resilience Optimization Architectures

In the Integrated Resilience Optimization formulation of the resilience-based design problem previously presented in Ref. [42], the design/operational variables and resilience variables are optimized over a set of fault scenarios quantified in a single cost function. In a generic form, this may be stated:

$$\begin{aligned} &min_{\mathbf{x_{D/O}},\mathbf{x_R}}\ C_{D/O}(\mathbf{x_{D/O}}) + C_R(\mathbf{x_{D/O}},\mathbf{x_R}) \\ &where\ C_R = \sum_{s\in S} n * r_s * C_s(\mathbf{x_{D/O}},\mathbf{x_R}) \\ &s.t.\ g_{D/O}(\mathbf{x_{D/O}}) \le 0, g_R(\mathbf{x_{D/O}},\mathbf{x_R}) \le 0 \\ &h_{D/O}(\mathbf{x_{D/O}}) = 0, h_R(\mathbf{x_{D/O}},\mathbf{x_R}) = 0 \end{aligned} \tag{2}$$

where $C_{D/O}$, $h_{D/O}$, and $g_{D/O}$ are design and operational cost objectives and constraints (the result of the design cost model at design variables $\mathbf{x_{D/O}}$) and $C_R$, $h_R$, and $g_R$ are the resilience cost model in terms of the design/operational variables and resilience variables $\mathbf{x_R}$. While the design and operational cost models may be of arbitrary form, the resilience cost model is given as the expected cost of the set of scenarios $S$ with cost $C_s$ and per-use rate $r_s$ over the number of uses of the system $n$ ($n$ and $r_s$ may also be given as functions of design, operational, and resilience variables if needed). Typically, the equality constraints $h$ represent simulations, which may not be given directly to the algorithm and are instead considered a part of the model. In this case, the design vector $\mathbf{x}$ is broken into two sets–the optimized decision variables $\mathbf{x}$ and corresponding response variables $\mathbf{y}$ which result from a simulation at $\mathbf{h}(\mathbf{x})$. This problem then has the form:

$$\begin{aligned} &min_{\mathbf{x_{D/O}},\mathbf{x_R}}\ C_{D/O}(\mathbf{x_{D/O}},\mathbf{y_{D/O}}) + C_R(\mathbf{x_{D/O}},\mathbf{x_R},\mathbf{y_{D/O}},\mathbf{y_R})) \\ &where\ C_R = \sum_{s\in S} n * r_s * C_s(\mathbf{x_{D/O}},\mathbf{x_R},\mathbf{y_{D/O}},\mathbf{y_R}) \\ &s.t.\ g_{D/O}(\mathbf{x_{D/O}},\mathbf{y_{D/O}}) \le 0, g_R(\mathbf{x_{D/O}},\mathbf{x_R},\mathbf{y_{D/O}},\mathbf{y_R}) \le 0 \end{aligned} \tag{3}$$

where $\mathbf{x_{D/O}}$, $\mathbf{x_R}$,$\mathbf{y_{D/O}}$, $\mathbf{y_R}$ are the decision variables and corresponding response variables of the design/operational and resilience models. This formulation corresponds to the all-in-one architecture in Figure 2 where the design/operational variables and resilience variables are optimized in a single problem.

### 2.2.1 Bilevel Architecture

In the bilevel architecture, a "lower level" optimization of the resilience variables is nested inside an "upper level" optimization of the the design and operational variables, as shown in Figure 3. In this structure, the upper level problem has the form:

$$\begin{aligned} &min_{\mathbf{x_{D/O}}}\ C_{D/O}(\mathbf{x_{D/O}},\mathbf{y_{D/O}}) + C^*_R(\mathbf{x_{D/O}},\mathbf{y_{D/O}})) \\ &s.t.\ g_{D/O}(\mathbf{x_{D/O}},\mathbf{y_{D/O}}) \le 0, g^*_R(\mathbf{x_{D/O}},\mathbf{y_{D/O}}) \le 0 \end{aligned} \tag{4}$$

where $C^*_R(\mathbf{x_{D/O}})$ and $g^*_R(\mathbf{x_{D/O}})$ are the optimal (or best) responses from the lower-level resilience optimization, which itself has the form:

$$\begin{aligned} &min_{\mathbf{x_R}}\ C_R = \sum_{s\in S} n * r_s * C_s(\mathbf{x_{D/O}},\mathbf{x_R},\mathbf{y_{D/O}},\mathbf{y_R}) \\ &s.t.\ g_R(\mathbf{x_{D/O}},\mathbf{x_R},\mathbf{y_{D/O}},\mathbf{y_R}) \le 0 \end{aligned} \tag{5}$$

where $\mathbf{x_{D/O}}$ and $\mathbf{y_{D/O}}$ are the design/operational variables from the upper level used as inputs for the lower-level optimization.

### 2.2.2 Alternating Architecture

In an alternating architecture, the "upper-level" and "lower-level" optimization of design/operational and resilience variables are conducted itera-

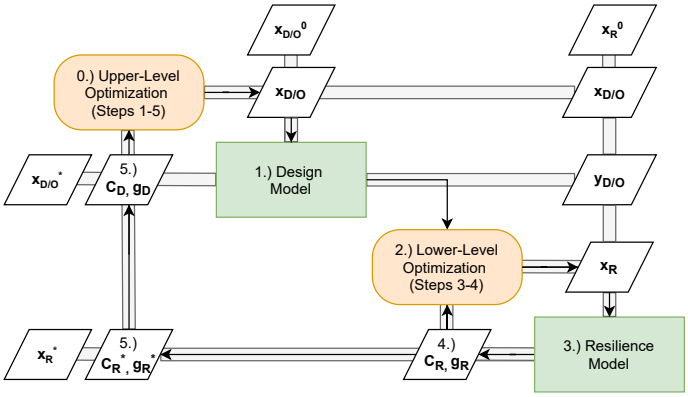


**FIGURE 3**. BILEVEL OPTIMIZATION ARCHITECTURE

tively one after another until each set of variables stops improving, as shown in Figure 4. This architecture was taken from the study of codesign architectures (where it is also referred to as the iterated sequential method) [48, 49]. In this formulation, the upper-level design optimization problem has the form:

$$min_{\mathbf{x_{D/O}}}\ C_{D/O}(\mathbf{x_{D/O}}, \mathbf{y_{D/O}}) + C_R(\mathbf{x_{D/O}}, \mathbf{y_{D/O}})) \tag{6}$$
$$s.t.\ g_{D/O}(\mathbf{x_{D/O}}, \mathbf{y_{D/O}}) \leq 0, g_R(\mathbf{x_{D/O}}, \mathbf{y_{D/O}}, \mathbf{x_R^{*-1}}, \mathbf{y_R^{*-1}}) \leq 0$$

where $\mathbf{x_R^{*-1}}$ and $\mathbf{y_R^{*-1}}$ are the optimal variables from the previous iteration of the lower level resilience optimization. This lower-level resilience optimization problem in turn has the form:

$$min_{\mathbf{x_R}}\ C_R = \sum_{s \in S} n * r_s * C_s(\mathbf{x_{D/O}^{*-1}}, \mathbf{x_R}, \mathbf{y_{D/O}^{*-1}}, \mathbf{y_R}) \tag{7}$$
$$s.t.\ g_R(\mathbf{x_{D/O}^{*-1}}, \mathbf{x_R}, \mathbf{y_{D/O}^{*-1}}, \mathbf{y_R}) \leq 0$$

where $\mathbf{x_{D/O}^{*-1}}$ and $\mathbf{y_{D/O}^{*-1}}$ are the optimal variables from the upper-level design/operational optimization. These optimization problems are each run iteratively until either there is a lack of improvement in the overall solution after both problems are solved, or some other exit condition is met (e.g., number of iterations, tolerances on variables, etc.). As shown in 4, the upper-level resilience model may be considered an optional part of the architecture (the effect of which is explored here), as can the iteration between upper and lower-level models (in which it is a sequential architecture, which was previously explored in Ref. [42] without an upper-level resilience cost).

#### 2.2.3 Decomposition Structures

Because of the computational cost and potentially large dimensionality of the lower-level problem, it can be helpful to decompose it into problems which can be solved independently. In two-stage approaches, (e.g. [17]), the resilience optimization is decomposed to each scenario independently, because the variables optimized are how the system responds to each situation individually (rather than in aggregate). Previous work explored the ability to decompose the resilience optimization to independent sets of scenarios [42], a decomposition which can also be performed over design variables, as long as they can also mapped to fault scenarios [50]. While these approaches are not explored in the new problems presented in this paper, they are reflected in the repository (see Table 1) and will be discussed in Section 4.

## 3 Problem Repository

As shown in Table 1, the resilience optimization problem repository contains a number of different formulations of problems and corresponding solution strategies. In general, the problems explored thus far have been of low dimensionality for ease of comparison. The problem set covers both multilevel Integrated Resilience Optimization problems (Notional Example, cooling Tank, Drone models) as well as single-level Resilience-based Design Optimization (EPS model) and Resilience Policy Optimization (Pandemic Management and Monopropellant System models) formulations. It additionally covers the exploration of AAO, Bilevel, Alternating, and Sequential optimization architectures as well as monolithic and by-scenario set decomposition strategies (although not a two-stage/by-scenario example). Of the problems in Table 1, the Drone, EPS, and Monopropellant system design problem have been described in previous work (Refs. [13, 42, 50], respectively). While these problems involved optimizing discrete variables using relatively simple search strategies (e.g. exhaustive search), the examples added in this work include continuous and discrete problems which must be searched with more sophisticated approaches.

The following subsections describe these three new problems, which are used in the repository to study the characteristics of resilience optimization problems and compare the effectiveness of given solution strategies. Section 3.1 describes a notional resilience optimization problem in which the problem is specified in a few simple equations and six decision variables, to illustrate considerations in continuous nonlinear programming formulations. Section 3.2 describes a pandemic management and response optimization problem which provides an intuitive demonstration of the optimization of the complex post-hazard dynamics involved in resilience. Section 3.3 then describes the optimization of a cooling tank system, which illustrates considerations in problems where the design model is a continuous-variable problem while the resilience level is a discrete-variable problem. Further information is available in the repository itself [53].

### 3.1 Notional System

To tractably demonstrate the use of optimization architectures in continuous-variable problems, a notional example was developed which formulates the resilience optimization problem in a simple set of equations. This problem has the resilience curve shown in Figure 5. As shown, the system has a nominal performance $x_p$, which then, because of a hazardous event, drops by an amount $x_a$ given by the slack in the system $x_s$. This con-

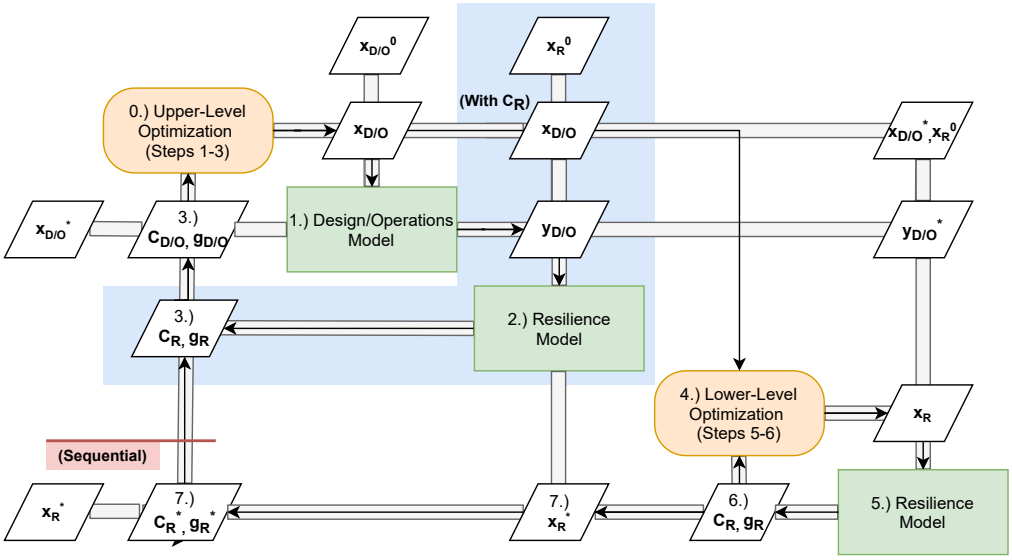


**FIGURE 4**. ALTERNATING AND SEQUENTIAL RESILIENCE OPTIMIZATION ARCHITECTURES AND VARIANTS

| Problem | Des. Vars | Res. Vars | Architecture | Decomposition | Algorithms Used | Model Type | Sim. Framework |
|---|---|---|---|---|---|---|---|
| Notional Example | 4 (C) | 2 (C) | AAO, Bilevel, Alt. (both) | Monolithic | Trust-Region | Equations | Stand-alone |
| Pandemic Management | N/A | 6 (C) | AAO | Monolithic | Differential Evolution | Dynamic | Stand-alone |
| Cooling Tank | 2 (C) | 54 (D) | Bilevel, Alt. (with $C_R$) | Monolithic | Powell's (D)/EA (R) | Dynamic | fmdtools [51] |
| Drone | 3 (D) | 2 (D) | AAO, Bilevel, Seq. (no $C_R$) | Monolithic, Scenario-Set | Exhaustive Search | Dynamic | fmdtools [51] |
| EPS | 14 | N/A | AAO | Scenario-Set | Line search | Static | IBFM [52] |
| Monopropellant System | N/A | 12 (D) | AAO | Monolithic | EA | Static | IBFM [52] |

**TABLE 1**. OVERVIEW OF OPTIMIZATION REPOSITORY PROBLEMS

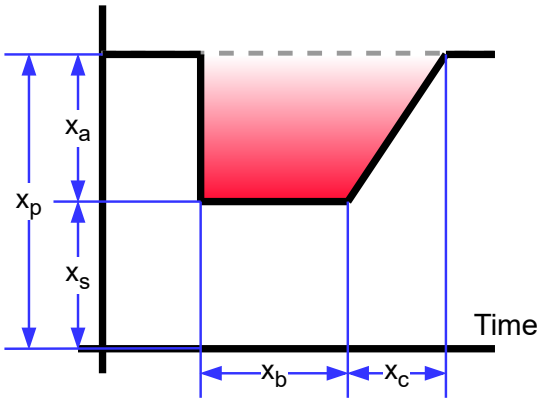


**FIGURE 5**. NOTIONAL SYSTEM UNDER THE GIVEN FAULT

tinues for the time $x_b$ until the system recovers, which takes time $x_c$. Cost functions and and constraints were constructed to form the optimization problem shown below:

$$
\begin{aligned}
min_{\mathbf{x}}\ f(\mathbf{x}) =& C_D(x_p,x_a,x_r,x_s) + C_R(x_r,x_a,x_b,x_c) \\
s.t.\ C_D =& -a\sqrt{x_a+0.1} + b\sqrt{x_s+0.1} + c/\sqrt{x_r} \\
C_R =& x_r * n * (d * x_a * (x_b + x_c/2) + e/x_c + f/x_b) \\
h_{D1} =& x_p - (x_s + x_a) = 0 \\
g_{D2} =& \frac{(x_p-1)^2}{c} - x_r < 0 \\
where \quad & 100 > x_p > 0, 2 > x_a > 0, 100 > x_r > 10^-10, \\
& 2 > x_s > 0, 100 > x_b > 0, 100 > x_c > 0 \\
and \quad & [a,b,c,n,d,e,f] = [1e6, 5e5, 100, 1e5, 50, 50, 20]
\end{aligned}
\tag{8}
$$

where $x_r$ is the rate of the fault, and $a$, $b$, $c$, $d$, $e$, $f$, and $n$ are all problem constants. In this problem, the system produces revenue from performing $x_s$ operations subject to a hazards and $x_a$ operations not subject to hazards, and is also subject to the costs of maintaining a reliable system. The constraint $h_{D1}$ relates slack, performance, and the drop in performance due to the fault while $g_{D2}$ specifies a peak performance level, which bounds reliability.

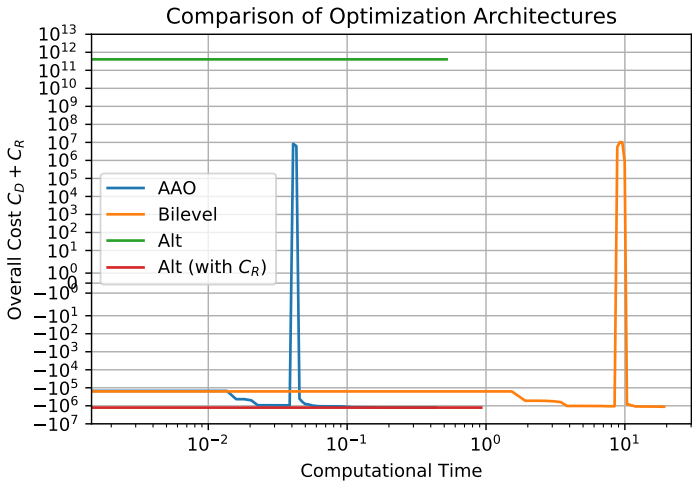


**FIGURE 6**. ARCH. PERFORMANCE ON NOTIONAL PROBLEM

**3.1.1 Optimization** This problem is a nonlinear programming problem, which can be solved using any number of NLP methods. In this work, this problem is solved using Python's trust-region algorithm in the SciPy package (see: [54]) using all-in-one, bilevel, alternating, and sequential architectures. While a full description of the implementation is out of the scope of this section, it is important to know that the bilevel method was run for 50 upper-level iterations with 20 corresponding lower-level iterations (since other convergence criteria were not met during optimization at either level) and the alternating approaches were set to terminate when the improvement between upper and lower-level optimizations was below a tolerance of $f_{tol} = 10^4$. The starting point used was $x = [1, 0.5, 10^{-4}, 0.5, 1, 1]$. Figure 6 shows the progression of the alternating, all-in-one, and alternating architectures through the search space. As shown, while both the all-in-one and alternating (with $C_R$) architectures complete the optimization in reasonable computational time, the bilevel approach takes an order of magnitude longer to approach the same solution while the alternating approach with out the resilience cost is ineffective at optimizing upper and lower-level costs. These results are also reflected in the final results comparison in Table 2, which additionally shows the performance of sequential architectures with and without the resilience cost $C_R$ in the upper level. As shown, the sequential architecture with the resilience cost in the upper level performs nearly as well as the all-in-one strategy, however removing the resilience cost from the upper level makes the architecture ineffective.

The comparative performance of these architectures can be explained by the characteristics of the problem and optimization methods. Because the design and resilience optimization problems are only loosely coupled (resilience variables do not significantly impact the upper-level cost), the sequential and alternating approach performs nearly as well as a monolithic approach in terms of solution found and computational time. However, this is only the case when the resilience cost is included in the upper-level model, which is consistent with Ref. [42]. Additionally, in this approach the bilevel structure performs poorly for a few reasons: First, establishing a gradient in the upper level is unnecessarily costly because each point evaluated to find the gradient using the finite difference method in the algorithm corresponds to a full optimization of the lower-level; second, the problem is difficult to solve in the upper level because of the constraints, meaning many of the iterations used solving the lower-level optimization are wasted because they are unrelated to establishing feasibility in the upper-level. As a result, the bilevel architecture, which must perform lower-level optimizations for each upper-level design point, proceeds inefficiently.

**TABLE 2**. NOTIONAL EXAMPLE OPTIMIZATION RESULTS

| Architecture | $x_p$ | $x_a$ | $x_r$ | $x_s$ | $x_b$ | $x_c$ | $f^*$ | time |
|---|---|---|---|---|---|---|---|---|
| All-in-One | 1.5 | 1.1 | 0.0022 | 0.41 | 0.62 | 10 | -1.3e+06 | 0.96 |
| Bilevel | 1.5 | 0.8 | 0.0024 | 0.7 | 0.71 | 10 | -1.2e+06 | 17 |
| Alt. (No $C_R$) | 1e+02 | 80 | 1e+02 | 20 | 0.071 | 10 | 4.1e+11 | 0.79 |
| Alternating | 1.5 | 1.1 | 0.0022 | 0.41 | 0.62 | 10 | -1.3e+06 | 1.6 |
| Sequential | 1.3 | 0.78 | 0.00083 | 0.51 | 0.72 | 10 | -1.2e+06 | 0.5 |
| Seq. (No $C_R$) | 1e+02 | 80 | 1e+02 | 20 | 0.071 | 10 | 4.1e+11 | 0.35 |

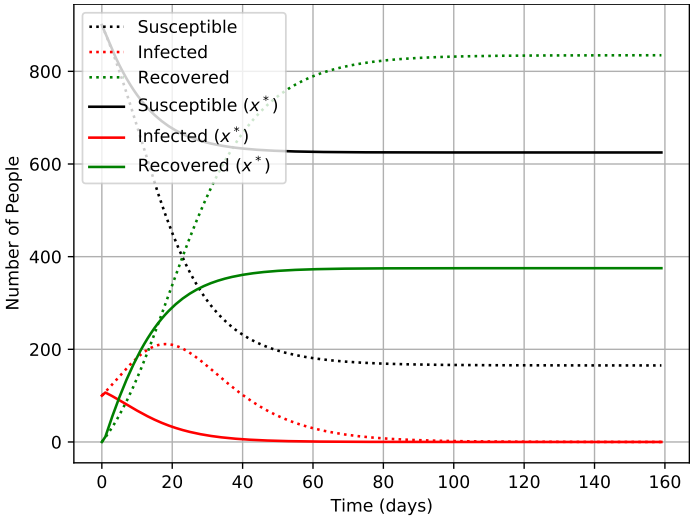


**FIGURE 7**. SIR MODEL SIMULATION COMPARISON

## 3.2 Pandemic Management Problem

While the notional example in Section 3.1 captures the idea of resilience optimization in a tractable model, the representation of failure and recovery dynamics is very simple. Often, rather than having an analytic model of failure dynamics, one has a set of differential equations which must be simulated over time. This section better illustrates the optimization of these dynamics by considering a pandemic management problem where the underlying model is a set of simulated differential equations for susceptible, infected, and recovered populations. In this problem, there are actions public health officials can take to manage

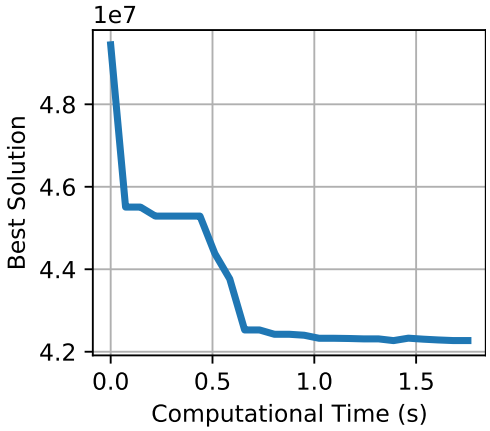


**FIGURE 8**. OPTIMIZATION OF PANDEMIC RESPONSE

the impacts of a pandemic–minimizing the number of infections while minimizing the impact to the economy. This can be formulated as an optimization problem where the goals are to minimize the total cost of the pandemic due to infections, preventative measures, and increased treatment measures. The cost function and constraints are as follows:

$$min_{\mathbf{x}}\ f(\mathbf{x}) = C_R(a,n,v,m,\alpha,IR) \quad (9)$$
$$s.t.\ \mathbf{x} \in [\mathbf{x_{min}}, \mathbf{x_{max}}]$$

where $a$, $n$, $v$, $m$, $\alpha$, and $IR$ are the variables of the problem $x$ with descriptions and bounds provided in Table 3. This objective is quantified in a context of a SIR model, which models the number of Susceptible ($S$), Infected ($I$), and Recovered ($R$) for a given population size ($N$), as a function of average contact rate ($A$) and recovery rate ($b$):

$$\frac{dS}{dt} = \frac{-ASI}{N} - V, \quad \frac{dI}{dt} = \frac{ASI}{N} - \frac{cI}{b}, \quad \frac{dR}{dt} = \frac{cI}{b} + V \quad (10)$$

Two policies are embedded in this model. The first policy, *policy 1*, is that if the proportion of infected people is larger than a given threshold value $\alpha$, the *contact rate* is dropped to $A = a$ and $V$ uninfected people are provided with a vaccine (otherwise $V = 0$). The second policy, *policy 2*, is that if infection rate is larger than a threshold value $IR$, the *recovery rate* is increased by adding additional medical staff $c$ (otherwise $c$=1). The resulting resilience cost function is:

$$C_R = I * c_I + (A_1 - a) * N * t_{PL1} * c_{PL1} + t_{PL2} * C_{PL2} \quad (11)$$

where $I$ is the total number of infections over the simulation, $c_I$ is the cost of infections (100000), $A_1$ is the initial contact rate (0.2), $a$ is the contact rate under policy 1, $N$ is the population (1000), $t_{PL1}$ is the time over which policy 1 is taken, $C_{PL1}$ is the per-person cost for policy 1 (10000), $t_{PL2}$ is the time over which policy 2 is taken, and $C_{PL2}$ is the cost of taking policy 2 (10000).

**TABLE 3**. PANDEMIC PROBLEM OPTIMAL PARAMETERS.

| Variable | Description | Range | $x^*$ |
|---|---|---|---|
| $a$ | policy1 contact rate | [0 - 0.2] | 0.06 |
| $n$ | medical staff per timestep | [1 - 5] | 1 |
| $V$ | vaccine per timestep | [8 - 10] | 9.4 |
| $m$ | default medical staff | [8 - 10] | 8 |
| $\alpha$ | policy 1 threshold | [0, 200] | 8.6 |
| $IR$ | policy2 threshold | [-0,500] | 17.9 |
| $C$ | total cost | N/A | $42 * 10^6$ |

#### 3.2.1 Optimization

While the variables of this problem are continuous, the fact that the policies in the model are triggered conditionally makes the problem discontinuous and piece-wise-flat, making it difficult to get gradient information. As a result, it cannot be solved using a gradient-based or direct search method. SciPy's differential evolution method (see: [54]) is thus used to explore the design space and find the optimal solution. The progression of this method is shown in Figure 8, with final results shown in Table 3. As shown, these variables, with a low contact rate in policy 1 (0.06) and a low threshold for policy 1 (8.6%–essentially the starting condition for the pandemic) reflect an aggressive suppression-based response to the pandemic, resulting in the improved pandemic response shown in Figure 7. As shown, this response results in shorter-duration pandemic with fewer infections than taking no response.

This formulation was solely focused on optimizing the failure response of the progression of a single scenario. While many methods were tried on this problem (trust-region, SQSLP, Powell, etc.), only the differential evolution algorithm was able to effectively optimize the function, because of the discontinuous nature of the variables. Another property of the differential evolution algorithm is its ability to parallelize the optimization of the model. Future work should develop an integrated resilience optimization formulation of this problem (with a design/operational level) to investigate how parallelism affects the ability of optimization architectures to effectively solve the problem at both levels.

### 3.3 Cooling Tank Problem

To illustrate resilience optimization considerations in problems where the upper level is continuous and lower-level is combinatorial, this case study focuses on the optimization of a cooling tank system shown in Figure 9 along with its fault behaviors under a leak. The purpose of this system is to keep an external heat source from overheating. The level of the tank provides a coolant to absorb and transfer heat, an outflow valve lets out warm coolant, and an inflow valve which brings in cold coolant. This system is subject to faults, such as the shown leak fault, in which the tank drains due to the loss of coolant. In this situation, no action may be taken to mitigate the fault from outside

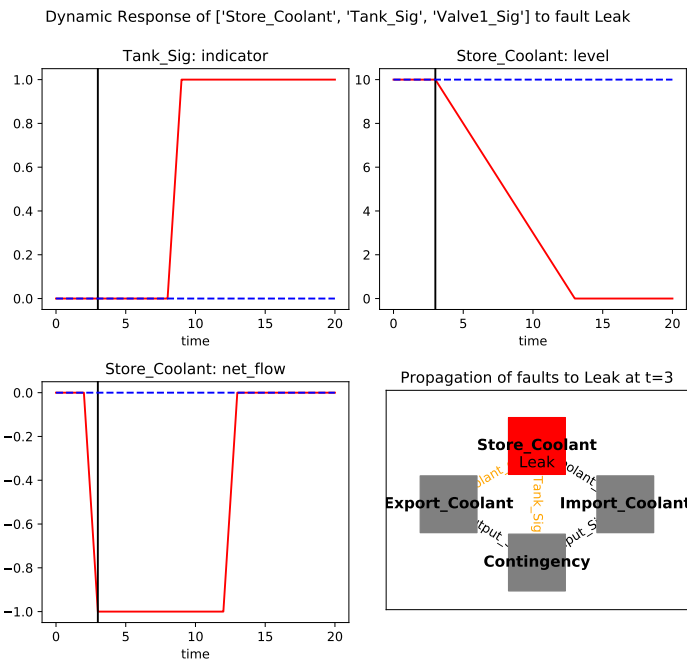


**FIGURE 9**. TANK SYSTEM UNDER LEAK FAULT

the system until $t = 20$, meaning the system must be designed in such a way to be optimally resilient in this window. To optimize the resilience of this system, the size of the tank $x_T$ and the size of the valve inlet $x_I$ are the design variables while the response signals to reconfigure the input valve $\mathbf{x_{ip}}$ and output valve $\mathbf{x_{ip}}$ are the resilience variables. The contingency management input states are $\vec{z} = [i,t,o]$, where $i$ is a given state in the input valve (leak, blockage, or nominal), $t$ is the level of the tank (low, high, or nominal), and $o$ is the state of the output valve (leak, blockage, or nominal). The resulting optimization problem is:

$$
\begin{aligned}
min_{\mathbf{x}}\ f(\mathbf{x}) =& C_D(x_T, x_I) + C_R(\mathbf{x_{ip}}, \mathbf{x_{op}}) \qquad (12)\\
s.t.\ C_D =& 1000(x_T - 10)^2 + 1000(x_T - 10) + 10000x_I^2\\
C_R =& \sum_{s \in S} n * r_s * C_s(\mathbf{x_{ip}}, \mathbf{x_{op}})\\
g_R =& \sum_{f \in N} f \leq 0\\
where\ x_T \in& (10-100), x_I \in (0-1),\\
& x_{ip,z} \in [-1,0,1], x_{op,z} \in [-1,0,1]
\end{aligned}
$$

where $C_D(x_T, x_I)$ is the design cost of both implementation and efficiency (which increases quadratically with buffer size) $C_R(\mathbf{x_{ip}}, \mathbf{x_{op}})$ is the resilience cost which is taken over the set of fault simulations, and $g_R$ is the (resilience-level) constraint determining whether the given set of variables results in a nominal mission profile in the nominal scenario. The scenario costs take

**TABLE 4**. TANK PROBLEM FAULT SCENARIOS.

| Scenario | Rate | Cost | Expected Cost |
|---|---|---|---|
| Import Coolant Leak | 1.7e-06 | 2.1e+06 | 3.5e+05 |
| Import Coolant Blockage | 1.7e-06 | 2.1e+06 | 3.5e+05 |
| Store Coolant Leak | 1.7e-06 | 1e+06 | 1.7e+05 |
| Export Coolant Leak | 1.7e-06 | 1e+06 | 1.7e+05 |
| Export Coolant Blockage | 1.7e-06 | 1e+05 | 1.7e+04 |

**TABLE 5**. TANK DESIGN PROBLEM VARIABLES

| Variable | Values | Description |
|---|---|---|
| $x_T$ | (10 - 100) | Tank Buffer Size |
| $x_I$ | (0 - 1) | Input Valve Margin |
| $x_{ip,z}$ | [-1,0,1] | Input Valve Turn in state $j$ |
| $x_{op,z}$ | [-1,0,1] | Output Valve Turn in state $j$ |

the form:

$$
\begin{aligned}
C_s = \sum_{t=0}^{t_e} & 10^4 * \mathbb{1}_{l(t)-x_T>0} + 10^6 * \mathbb{1}_{l(t)\leq 0}\\
& + 10^5 \mathbb{1}_{b(t)\leq 0} + 100(\mathbb{1}_{iv(t)\neq 0} + \mathbb{1}_{ov(t)\neq 0})
\end{aligned}
$$

where $l(t)$ is the level of coolant in the tank, $b(t)$ is the amount of useful unspent buffer coolant in the tank, and $ov(t)$ and $iv(t)$ are the policy used by the input and output valves. As shown, the primary failure costs result from the tank overfilling, emptying, and no longer having enough buffer coolant to cool the heat source. The valve policy costs are used to discourage unnecessary usage of valve reconfiguration in cases where it is not needed, such as in policy states that are not entered in the simulations in the set of considered scenarios. These costs are summed over the number of timesteps in the simulation where the condition is present to account for the increased risk from greater time exposure resulting from each condition. Five scenarios are included in the set of scenarios $S$ in the resilience model, which constitute blockages and leaks in each system, as shown in Table 4, along with their rates and costs (assuming no mitigation).

**3.3.1 Optimization** This problem is difficult to solve because of the high resilience model dimensionality (54 variables) and the mix of variable types (continuous in the design model and discrete in the resilience model). Additionally, because variables are state-based (and not scenario-based), some variables may be coupled (e.g., raising a level when it is too low may cause the level to become too high, resulting in a new set of actions). Thus, this work uses a custom evolutionary algorithm in a monolithic resilience model to generate and refine solutions. This makes it difficult to solve design and resilience models in tandem, since the result of the lower-level optimization may not necessarily be continuous (or "act" like a continuous function to an upper-level solver). To solve this problem, the design model

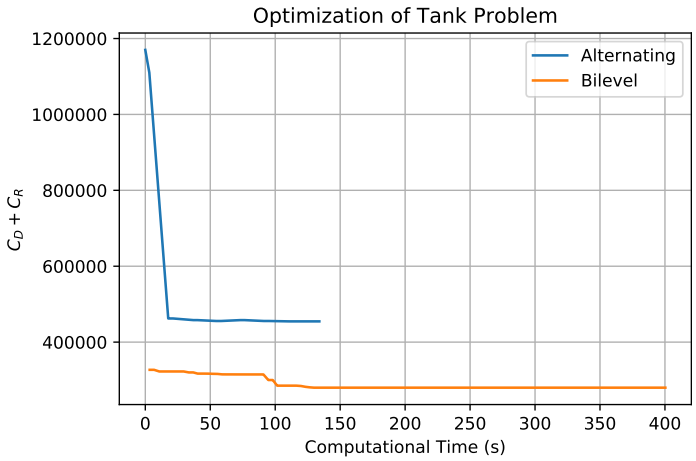


**FIGURE 10**. ARCH. PERFORMANCE ON TANK PROBLEM

in this work is searched using Powell's method in SciPy [54], a gradient-free direct search method which iteratively performs line-searches along the feasible directions of the search space determined first by the coordinate directions of the space and then by the results of previous searches.

The alternating architecture and bilevel architectures were implemented on this problem using a population size of 50 and a number of iterations of 20 in the alternating architecture and a population size of 10 and number of iterations of 20 in the bilevel architecture. Both strategies used a seeding method in the evolutionary algorithm which enabled the best populations found at the end of each lower-level optimization to be saved and carried over to the next optimization. The progression of these algorithms through the search space is shown in Figure 10. As shown, the alternating architecture very quickly reaches a plateau where each individual optimization does not improve the design significantly, while the bilevel architecture searches the space more effectively, ultimately converging to a lower-cost design, as shown Table 6. As shown, the alternating architecture converges to a tank size of 20, the size which *by design* mitigates leak faults by making it impossible for the tank to drain completely, while the bilevel architecture converges to a tank size of 18, using pipe buffer and a corresponding lower-level resilience policy to make up the difference for the extra 2 time-steps. This difference in performance and in design found (and corresponding policy) between the architectures is a result of the coupling relationship between the upper and lower-level problems. That is, the pipe buffer has no intrinsic value outside its ability to be leveraged by a lower-level policy, which must be optimized in the lower-level to be effective. This is why the alternating structure reduces pipe margin to 0 while the bilevel has a pipe margin of 1 which it leverages with a corresponding optimized resilience policy.

## 4 Discussion

The problems in this repository demonstrate how solving resilience optimization problems requires structuring the optimization architecture to most efficiently optimize the system, which is important because of the the large comparative simulation time of the resilience model: Unless there is a complex interfacing design optimization problem, the resilience optimization is likely the primary driver of computational expense in resilience optimization because of the many dynamical simulations needed to quantify the appropriate metrics. This is especially true as the resilience model becomes more elaborate and the problem more complex, since the computational cost increases by $\mathcal{O}(E*T*S)$, where $E$ is the number of equations in the resilience model, $T$ is the number of time-steps, and $S$ is the number of scenarios. While computational expense can also be reduced in the model, appropriate optimization architectures enable quicker and more effective solution without sacrificing model fidelity. As was illustrated in the notional example and tank problem presented here (and the drone model presented in previous work), the choice and development of optimization architecture can influence the effectiveness of the overall approach. This difference in effectiveness in part depends on the *alignment* and *coupling* of the upper and lower-level problems, as summarized in Figure 11. Alignment refers to whether the upper-level and lower-level objectives oppose, support, or are invariant to each other (i.e. if $\frac{\mathbf{d\hat{C}_D}}{\mathbf{dx_D}} \approx \frac{\mathbf{d\hat{C}}}{\mathbf{dx_D}}$ where $\frac{\mathbf{dC}}{\mathbf{dx_D}} = \frac{\mathbf{dC_D}}{\mathbf{dx_D}} + \frac{\mathbf{dC_R}}{\mathbf{dx_D}}$). When the upper and lower-level problems are not aligned (which constitutes all problems where there is a trade-off between design cost and resilience), the resilience model (or a surrogate) must be included in alternating and se-

**TABLE 6**. TANK PROBLEM PERFORMANCE COMPARISON

| Approach | $x_t^*$ | $x_l^*$ | $f^*$ | Time (s) |
|---|---|---|---|---|
| Bilevel | 18 | 1 | 2.8e+05 | 4e+02 |
| Alternating (with $C_R$) | 20 | 3.4e-08 | 4.5e+05 | 1.3e+02 |

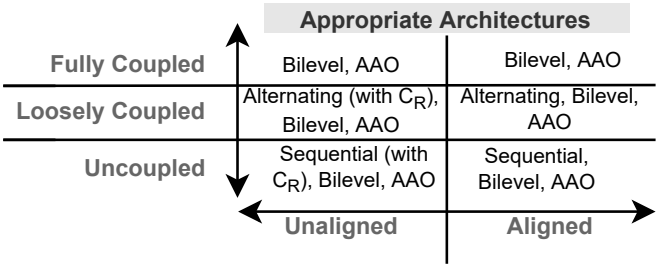


**FIGURE 11**. ARCHITECTURE APPLICABILITY

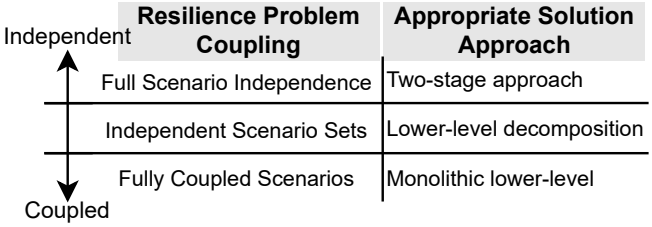


**FIGURE 12**. DECOMPOSITION APPROACH APPLICABILITY

quential architectures for them to perform adequately. Coupling, on the other hand, refers to the degree to which upper-level variables are constrained with lower-level variables. In an uncoupled situation, the lower-level optimization is merely a refinement of the upper-level optimization, meaning there is a direct path from $[\mathbf{x}_\mathbf{D}^*, \mathbf{x}_\mathbf{R}^\mathbf{0}]$ to $[\mathbf{x}_\mathbf{D}^*, \mathbf{x}_\mathbf{R}^*]$ justifying a sequential architecture. In a loosely-coupled situation, the optimal choice of design variables $\mathbf{x}_\mathbf{D}^*$ may depend on the choice of resilience variables, however, the choices do not directly depend on each other (i.e., Eq. 13 holds) which justifies an alternating approach. Finally, in a fully-coupled situation where Eq. 13 does not hold, the design and resilience variables must be jointly explored, which requires a bilevel or all-at-once approach.

$$C(\mathbf{x}+\delta\mathbf{x}) \approx C(\mathbf{x}+\delta\mathbf{x}_\mathbf{D}) + C(\mathbf{x}+\delta\mathbf{x}_\mathbf{R}) - C(\mathbf{x}) \qquad (13)$$

The optimization method used can additionally affect the efficiency and effectiveness of a given architecture. In the exhaustive search used in the Drone model in Ref. [42], for example, the bilevel architecture was able to reduce (some) computational cost by reducing the number of iterations by reducing the space of the search and enabling a lower-level decomposition strategy. However, as shown in the notional example presented here, the large number of lower-level optimizations necessary to approximate a gradient in the upper-level problem can increase the computational cost by orders of magnitude when using a gradient-based solver. Thus, to perform efficiently, the choice of architecture must be connected to the solution strategy–whether it be because an architecture is inherently quicker with a given strategy, or because upper and lower-level problems require different methods to solve efficiently.

Finally, several of the problems in the repository (Drone, EPS, Monopropellant system) use a decomposition strategy to reduce the dimensionality of the problem and number of resilience simulations needed. However, these decomposition strategies can only be used when different variables or sets of variables are uncoupled, as shown in Figure 12. While problem formulations where features or control policies which map to specific failure scenarios lend themselves to decomposition strategies, formulations where the actions of a resilience policy is optimized over interacting states do not. Thus, while decomposition can increase the efficiency of a given strategy, the problem must be formulated appropriately for them to be effective.

## 5 CONCLUSIONS

Effective application of resilience optimization methods requires knowledge of the underlying optimization problem formulation. Because resilience optimization problems can be formulated in a number of different ways, characterizing the effectiveness of given methods requires considering the performance of these methods over the range of possible formulations. This work studied previously-developed resilience optimization problems along with three new problems formulated in this work, which lead to insights about the applicability of optimization approaches. First, the ability for optimization architectures to optimize effectively depends on the *alignment* and *coupling* of the design and resilience problems. Second, the performance of bilevel approaches vary widely based on the algorithms used at each level, since they can enable different solution strategies at each level but also require a full optimization of the lower-level at each upper-level design point. Finally, while decomposition strategies can lower solution time, they only apply when variables can be mapped to independent scenarios and scenario-sets–when the resilience variables interact with each other (e.g., through model states), a monolithic strategy is needed.

These insights constitute a beginning towards the characterization and systematic study of resilience optimization problems and approaches. However, there are a few limitations that deserve mentioning. The repository is limited to problems developed in the authors' previous work, which leaves out other problems and solution strategies (e.g. two-stage, etc.) previously presented in the literature. Future work should replicate these problems and frameworks to perform a broader comparison of strategies. Additionally, many formulations in this repository are built around simulations, which make them more difficult to leverage for the study of optimization architectures because of computational expense. Thus, it would be helpful to formulate more simple continuous problems to enable architecture comparison, analysis of problem characteristics, and develop specialized methods for the efficient optimization of resilience.

## ACKNOWLEDGMENT

This research was partially conducted at NASA Ames Research Center. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not constitute or imply its endorsement by the United States Government.

## REFERENCES

[1] Linkov, I., Bridges, T., Creutzig, F., Decker, J., Fox-Lent, C., Kröger, W., Lambert, J. H., Levermann, A., Montreuil, B., Nathwani, J., et al., 2014. "Changing the resilience paradigm". *Nature Climate Change,* ***4***(6), pp. 407–409.

[2] Punzo, G., Tewari, A., Butans, E., Vasile, M., Purvis, A., Mayfield, M., and Varga, L., 2020. "Engineering resilient complex systems: the necessary shift toward complexity science". *IEEE Systems Journal,* ***14***(3), pp. 3865–3874.

[3] Cottam, B., Specking, E., Small, C., Pohl, E., Parnell, G. S., and Buchanan, R. K., 2019. "Defining resilience for engineered systems". *Engineering Management Research,* ***8***(2), pp. 11–29.

[4] Yodo, N., and Wang, P., 2016. "Engineering resilience quantification and system design implications: A literature survey". *Journal of Mechanical Design,* ***138***(11).

[5] da Silva, F. S., and Matelli, J. A., 2019. "Development of metrics for resilience quantification in energy systems". In Proceedings of the Annual Conference of the PHM Society, Vol. 11.

[6] Matelli, J. A., and Goebel, K., 2018. "Conceptual design of cogeneration plants under a resilient design perspective: Resilience metrics and case study". *Applied Energy,* ***215***, pp. 736–750.

[7] Ouyang, M., Dueñas-Osorio, L., and Min, X., 2012. "A three-stage resilience analysis framework for urban infrastructure systems". *Structural safety,* ***36***, pp. 23–31.

[8] Specking, E., Cottam, B., Parnell, G., Pohl, E., Cilli, M., Buchanan, R., Wade, Z., and Small, C., 2019. "Assessing engineering resilience for systems with multiple performance measures". *Risk Analysis,* ***39***(9), pp. 1899–1912.

[9] Small, C., Parnell, G., Pohl, E., Goerger, S. R., Cottam, B., Specking, E., and Wade, Z., 2017. "Engineered resilient systems with value focused thinking". In INCOSE international symposium, Vol. 27, Wiley Online Library, pp. 1371–1385.

[10] Margolis, J. T., Sullivan, K. M., Mason, S. J., and Magagnotti, M., 2018. "A multi-objective optimization model for designing resilient supply chain networks". *International Journal of Production Economics,* ***204***, pp. 174–185.

[11] Wade, Z., Parnell, G. S., Goerger, S. R., Pohl, E., and Specking, E., 2019. "Designing engineered resilient systems using set-based design". In *Systems Engineering in Context*. Springer, pp. 111–122.

[12] MacKenzie, C. A., and Hu, C., 2019. "Decision making under uncertainty for design of resilient engineered systems". *Reliability Engineering & System Safety,* ***192***, p. 106171.

[13] Hulse, D., Hoyle, C., Goebel, K., and Tumer, I. Y., 2019. "Quantifying the resilience-informed scenario cost sum: A value-driven design approach for functional hazard assessment". *Journal of Mechanical Design,* ***141***(2).

[14] Moslehi, S., and Reddy, T. A., 2018. "Sustainability of integrated energy systems: A performance-based resilience assessment methodology". *Applied energy,* ***228***, pp. 487–498.

[15] Youn, B. D., Hu, C., and Wang, P., 2011. "Resilience-driven system design of complex engineered systems". *Journal of Mechanical Design,* ***133***(10).

[16] Yodo, N., and Wang, P., 2016. "Resilience allocation for early stage design of complex engineered systems". *Journal of Mechanical Design,* ***138***(9).

[17] Wu, J., and Wang, P., 2020. "Risk-averse optimization for resilience enhancement of complex engineering systems under uncertainties". In International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, American Society of Mechanical Engineers. DETC2020-22226.

[18] Kall, P., Wallace, S. W., and Kall, P., 1994. *Stochastic programming*. Springer.

[19] Dantzig, G. B., 1955. "Linear programming under uncertainty". *Management science,* ***1***(3-4), pp. 197–206.

[20] Beale, E. M., 1955. "On minimizing a convex function subject to linear inequalities". *Journal of the Royal Statistical Society: Series B (Methodological),* ***17***(2), pp. 173–184.

[21] Faturechi, R., Levenberg, E., and Miller-Hooks, E., 2014. "Evaluating and optimizing resilience of airport pavement networks". *Computers & Operations Research,* ***43***, pp. 335–348.

[22] Miller-Hooks, E., Zhang, X., and Faturechi, R., 2012. "Measuring and maximizing resilience of freight transportation networks". *Computers & Operations Research,* ***39***(7), pp. 1633–1643.

[23] Mazidi, M., Rezaei, N., Ardakani, F. J., Mohiti, M., and Guerrero, J. M., 2020. "A hierarchical energy management system for islanded multi-microgrid clusters considering frequency security constraints". *International Journal of Electrical Power & Energy Systems,* ***121***, p. 106134.

[24] Biswas, A., and Hoyle, C., 2019. "A literature review: Solving constrained non-linear bi-level optimization problems with classical methods". In ASME 2019 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, American Society of Mechanical Engineers Digital Collection.

[25] Von Stackelberg, H., 1934. *Market structure and equilibrium*. Springer Science & Business Media.

[26] Rismiller1, S., Cagan, J., and McComb, C., 2020. "Stochastic stackelberg games for agent-driven robust design". In International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, American Society of Mechanical Engineers. DETC2020-22153.

[27] Piacenza, J. R., Faller, K. J., Bozorgirad, M. A., Cotilla-Sanchez, E., Hoyle, C., and Tumer, I. Y., 2020. "Understanding the impact of decision making on robustness during complex system design: More resilient power systems". *ASCE-ASME J Risk and Uncert in Engrg Sys Part B Mech Engrg,* ***6***(2).

[28] Zhang, X., Hu, Z., and Mahadevan, S., 2020. "Bilevel optimization model for resilient configuration of logistics service centers". *IEEE Transactions on Reliability*.

[29] Biswas, A., Chen, Y., Gibson, N., and Hoyle, C., 2020. "Bilevel flexible-robust optimization for energy allocation problems". *ASCE-ASME J Risk and Uncert in Engrg Sys Part B Mech Engrg,* ***6***(3).

[30] Mehr, A. F., Tumer, I., and Barszcz, E., 2005. "Optimal design of integrated systems health management (ishm) for improving the safety of nasas exploration missions: A multidisciplinary design approach". In Sixth World Congress on Structural and Multidisciplinary Optimization, Rio de Janeiro, May.

[31] Yu, B. Y., Honda, T., Zubair, S., Sharqawy, M. H., and Yang, M. C., 2013. "A framework for system design optimization based on maintenance scheduling with prognostics and health management". In International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, Vol. 55881, American Society of Mechanical Engineers, p. V03AT03A035.

[32] Wang, P., Youn, B. D., Hu, C., Ha, J. M., and Jeon, B., 2015. "A probabilistic detectability-based sensor network design method for system health monitoring and prognostics". *Journal of Intelligent Material Systems and Structures,* ***26***(9), pp. 1079–1090.

[33] Malere, J. P., 2017. Application of linear programming to optimize the cost-benefit of an ivhm system. Tech. rep., SAE Technical Paper.

[34] Hoyle, C., Tumer, I. Y., Mehr, A. F., and Chen, W., 2009. "Health management allocation during conceptual system design". *Journal of Computing and Information Science in Engineering,* ***9***(2).

[35] Padula, S., Alexandrov, N., and Green, L., 1996. "Mdo test suite at nasa langley research center". In 6th Symposium on Multidisciplinary Analysis and Optimization, p. 4028.

[36] Dorigo, M., and Gambardella, L. M., 1997. "Ant colonies for the travelling salesman problem". *biosystems,* ***43***(2), pp. 73–81.

[37] Larranaga, P., Kuijpers, C. M. H., Murga, R. H., Inza, I., and Dizdarevic, S., 1999. "Genetic algorithms for the travelling salesman problem: A review of representations and operators". *Artificial Intelligence Review,* ***13***(2), pp. 129–170.

[38] Antosiewicz, M., Koloch, G., and Kamiński, B., 2013. "Choice of best possible metaheuristic algorithm for the travelling salesman problem with limited computational time: quality, uncertainty and speed". *Journal of Theoretical and Applied Computer Science,* ***7***(1), pp. 46–55.

[39] Alexandrov, N., and Kodiyalam, S., 1998. "Initial results of an mdo method evaluation study". In 7th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, p. 4884.

[40] Balling, R., and Wilkinson, C., 1997. "Execution of multidisciplinary design optimization approaches on common test problems". *AIAA journal,* ***35***(1), pp. 178–186.

[41] Tedford, N. P., and Martins, J. R., 2010. "Benchmarking multidisciplinary design optimization algorithms". *Optimization and Engineering,* ***11***(1), pp. 159–183.

[42] Hulse, D., Biswas, A., Hoyle, C., Tumer, I. Y., Kulkarni, C., and Goebel, K. Exploring architectures for integrated resilience optimization.

[43] Maul, W. A., Kopasakis, G., Santi, L. M., Sowers, T. S., and Chicatelli, A., 2008. "Sensor selection and optimization for health assessment of aerospace systems". *Journal of Aerospace Computing, Information, and Communication,* ***5***(1), pp. 16–34.

[44] Rausch, R. T., Goebel, K. F., Eklund, N. H., and Brunell, B. J., 2007. "Integrated in-flight fault detection and accommodation: A model-based study". *Journal of Engineering for Gas Turbines and Power,* ***129***(4), pp. 962–969.

[45] Balachandran, S., and Atkins, E., 2017. "Markov decision process framework for flight safety assessment and management". *Journal of Guidance, Control, and Dynamics,* ***40***(4), pp. 817–830.

[46] Müller, S., Gerndt, A., and Noll, T., 2019. "Synthesizing failure detection, isolation, and recovery strategies from nondeterministic dynamic fault trees". *Journal of Aerospace Information Systems,* ***16***(2), pp. 52–60.

[47] Yildiz, A., Akcal, M. U., Hostas, B., and Ure, N. K., 2019. "Switching control architecture with parametric optimization for aircraft upset recovery". *Journal of Guidance, Control, and Dynamics,* ***42***(9), pp. 2055–2068.

[48] Silvas, E., Hofman, T., Murgovski, N., Etman, L. P., and Steinbuch, M., 2016. "Review of optimization strategies for system-level design in hybrid electric vehicles". *IEEE Transactions on Vehicular Technology,* ***66***(1), pp. 57–70.

[49] Allison, J. T., and Herber, D. R., 2014. "Multidisciplinary design optimization: multidisciplinary design optimization of dynamic engineering systems". *AIAA journal,* ***52***(4), pp. 691–710.

[50] Hulse, D., Hoyle, C., Tumer, I. Y., and Goebel, K., 2019. "Decomposing incentives for early resilient design: Method and validation". In International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, Vol. 59193, American Society of Mechanical Engineers, p. V02BT03A015.

[51] Hulse, D., Walsh, H., Dong, A., Hoyle, C., Tumer, I., Kulkarni, C., and Goebel, K., 2020. "fmdtools: A fault propagation toolkit for resilience assessment in early design".

[52] McIntire, M. G., Keshavarzi, E., Tumer, I. Y., and Hoyle, C., 2016. "Functional models with inherent behavior: Towards a framework for safety analysis early in the design of complex systems". In ASME International Mechanical Engineering Congress and Exposition, Vol. 50657, American Society of Mechanical Engineers, p. V011T15A035.

[53] Hulse, D., Zhang, H., and Biswas, A., 2021. "Designengrlab/resil_opt_examples".

[54] Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., et al., 2020. "Scipy 1.0: fundamental algorithms for scientific computing in python". *Nature methods,* ***17***(3), pp. 261–272.