Contents

Illustrations

Tables

Contributors

Sean McGregor *Oregon State University, School of Electrical Engineering and Computer Science, 1148 Kelley Engineering Center, Corvallis, OR 97331-5501*
Rachel Houtman *Oregon State University, College of Forestry*
Ronald Metoyer *Notre Dame University, College of Engineering*
Thomas Dietterich *Oregon State University, School of Electrical Engineering and Computer Science*

1

Connecting Conservation Research and Implementation: Building a Wildfire Assistant

Sean McGregor,[a] Rachel Houtman,[b] Ronald Metoyer,[c] and Thomas Dietterich[d]

Abstract

Effective management of shared public ecosystems like national forests requires balancing beneficial uses (e.g., biodiversity and timber production). For forest wildfire suppression decisions, the balancing process tends to favor myopic policies that increase the risk of catastrophic fires through time. This work introduces WildFireAssistant as a visual analytics system combining a 100 year forest simulator with black box optimization. WildFireAssistant incorporates long-term impacts into decision-making by facilitating forest managers in their exploration of policies for different stakeholder groups. Since a critical feature for WildFireAssistant is the ability to interactively change the reward function (e.g., to explore the differences between policies optimized for biodiversity and timber), developing methods that support fast simulation and optimization are required. To this end, we develop a Model-Free Monte Carlo surrogate that pre-computes state transitions for fast visualization of policy functions. We then demonstrate a black box optimization method that utilizes the surrogate to support interactive optimization of user-selected reward functions.

1.1 Introduction

Nearly 1.6 billion people depend on forests for their livelihood. In many cases their economic activities place them into direct conflict with the

[a] Oregon State University School of Electrical Engineering and Computer Science
[b] Oregon State University College of Forestry
[c] University of Notre Dame College of Engineering
[d] Oregon State University School of Electrical Engineering and Computer Science

best interests of 80 percent of all terrestrial animals, plants, and insects (United Nations, 2017). A common approach to balancing forest economic value with the inherent value of natural ecology is to publicly manage forests according to stakeholder objectives. Defining objectives serving all stakeholders is a challenging political process. For example, a period in the 1990s is referred to as the "Timber Wars" in the US Pacific Northwest because of the troubling and occasionally violent (Egan, 2005) conflicts that arose between stakeholder groups over forest management policies during that period. This is typical of many ecosystem management problems—the complexity of ecosystem dynamics and the broad array of interested parties makes it difficult to identify politically-feasible policies.

A particularly challenging and controversial area of forest management is forest wildfire suppression decisions. Wildland fires pose immediate risks to wildlife, fire fighters, timber values, air quality, and recreation, but fires also produce long term rewards in the form of biodiversity, reduced future fire risk, increased timber harvest, and more accessible forests. Despite a consensus in the forestry community on the necessity of allowing some wildfires to burn (Forest History Society, 2017), forest managers still suppress the vast majority of wildfires (United States Department of Agriculture, 2015; Tephens and Ruth, 2005).

The institutional constraints preventing forest managers from allowing more wildfires to burn is evident in their decision support tools. Forest managers are required to use decision support processes to *guide* and *document* wildfire management decisions (National Interagency Fire Center, 2009). The Wildland Fire Decision Support System (WFDSS) fulfills this requirement with a web-based platform that includes components for situational assessment, hazard and risk analysis, and documentation of decision rationale. WFDSS does not include a component estimating the future benefits of wildfires.

To help forest managers incorporate the future benefits of wildfires into their decision making, we developed the WildFireAssistant system as a visual analytic environment for reward specification and policy optimization. Forest managers can modify the rewards realized over century time spans to represent different stakeholders, optimize a policy, and explore the results in a visualization. Forest managers can also document their decisions by saving the WildFireAssistant visualizations.

Developing WildFireAssistant posed several algorithmic challenges that are the focus of this work. The high-fidelity ecology, weather, and fire components of our Oregon Centennial Wildfire Simulator (OCWS)

(Houtman et al., 2013) are extremely expensive computationally. Without a fast simulator, the forest manager cannot adequately explore the space of reward and policy functions. To address this problem we develop a method for pre-computing state transitions and avoid simulation when the forest manager interacts with WildFireAssistant. Further, we perform policy optimization on top of the pre-computed state transitions with a Bayesian policy search method. These two components reduce optimization time from hundreds of hours (fully-parallelized) to less than a minute (without parallelization).

This work proceeds in four parts. First, we formally introduce the forest wildfire management problem. Second, we introduce the system architecture for WildFireAssistant as shown in Figure 1.1. The third part then addresses the computational challenges associated with supporting interactive visualization and optimization of policies. Finally, we quantitatively evaluate the quality of WildFireAssistant visualization and optimization.

1.2 Wildfire Suppression Decisions in Context

Endert et al. (2014) recommend shifting from "human in the loop" design to "human *is* the loop" design. The difference between the two viewpoints is one in which humans assist the computational process (i.e., humans are *in* the loop) to one in which the computing process supports the human. For the cultural and practical considerations of forestry decisions, "human *is* the loop" is the proper design philosophy. The official training course for "Advanced Incident Management with WFDSS" (The National Wildfire Coordinating Group, 2009) states,

> Decision-making is not a science but an art. It requires judgment not calculation. There is no unit of measure which can weigh the substantive consequences of a decision against the political consequences, or judge the precise portions of public opinion and congressional pressure, or balance short-range against long-range, or private against public considerations.

This advice reveals the wider culture of forest policy research, which typically performs case study policy optimizations that either minimize market costs (like suppression expenses; Houtman et al. (2013)), or maximize a non-market benefit (like species abundance; Ager et al. (2007)). If the study includes a non-market reward with another reward component, then the specification of their relative importance requires extensive post-optimization analysis and justification. Despite the potential

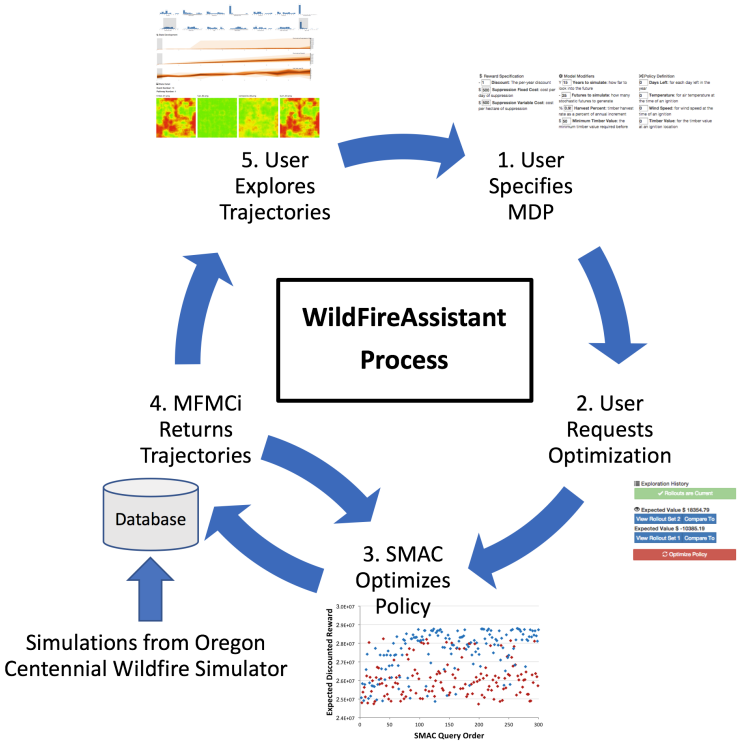


Figure 1.1 A complete view of the WildFireAssistant process. We begin by loading high-fidelity simulations into a state transition database. Next the user enters the optimization and exploration loop by specifying the reward function and requesting an optimized policy. Then the black box optimization algorithm SMAC produces a newly optimized policy by repeatedly calling a Model-free Monte Carlo with independencies (MFMCi) surrogate model. After optimizing the policy, the MFMCi surrogate returns one more batches of trajectories for visualization. After exploring the trajectories with visualization, the user may then request an optimized policy for a different reward function.

for controversy, there are considerable advantages to jointly optimizing multi-objective reward functions. For instance, Dilkina et al. (2016) showed considerable efficiency gains in jointly optimizing wildlife reserve land acquisition for multiple species.

In the wildfire simulations of the OCWS, we can optimize policies for any combination of biodiversity, fire suppression expenses, air quality, recreation, and timber. The OCWS defines a finite horizon discounted

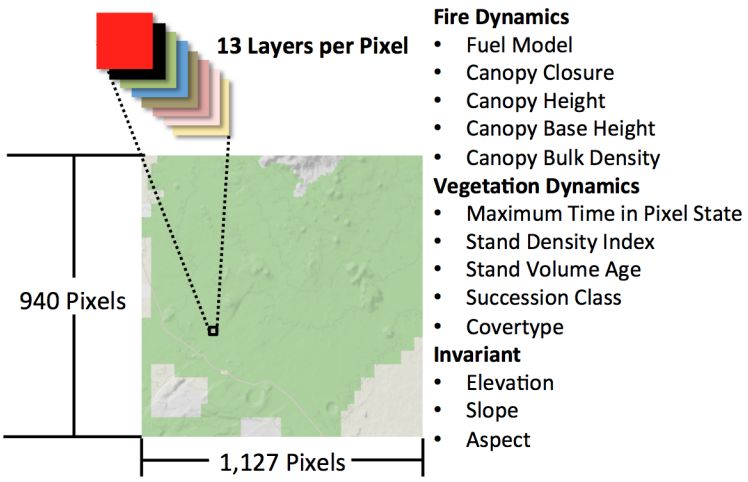


Figure 1.2 The landscape totals approximately one million pixels, each of which has 13 state variables that influence the spread of wildfire on the landscape.
(Map is copyright of OpenStreetMap contributors)

Markov Decision Process (MDP) with a designated start state distribution (Bellman, 1957; Puterman, 1994) $\mathcal{M} = \langle S, A, P, R, \gamma, P_0, h \rangle$. S is a finite set of states of the world; A is a finite set of possible actions that can be taken in each state; $P : S \times A \times S \mapsto [0, 1]$ is the conditional probability of entering state s' when action a is executed in state s; $R(s, a)$ is the reward received after performing action a in state s; $\gamma \in (0, 1)$ is the discount factor, P_0 is the distribution over starting states, and h is the horizon. The optimization goal is to find a policy, π, that selects actions maximizing the expected discounted sum of rewards of the MDP.

Figure 1.2 shows a snapshot of the landscape as simulated by the OCWS. The landscape is comprised of approximately one million pixels, each with 13 layers summarizing the invariant properties of the location (i.e., elevation, slope, aspect) and the dynamic vegetation attributes (i.e., tree properties and density). When a fire is ignited by lightning, the policy must choose between two actions: *Suppress* (fight the fire) and *Let Burn* (do nothing). Hence, $|A| = 2$.

The simulator spreads wildfires with the FARSITE fire model (Finney, 1998) according to the surrounding pixel variables (X) and the hourly weather. Weather variables include *hourly wind speed, hourly wind direction, hourly cloud cover, daily maximum/minimum temperature, daily maximum/minimum humidity, daily hour of maximum/minimum tem-*

perature, daily precipitation, and *daily precipitation duration.* These are generated by resampling from 25 years of observed weather (Western Regional Climate Center, 2011).

After computing the extent of the wildfire on the landscape, the simulator applies a cached version of the Forest Vegetation Simulator (Dixon, 2002) to update the vegetation of the individual pixels. Finally, a harvest scheduler selects pixels to harvest for timber value.

Simulating the spread of fire is computationally expensive. Generating a single 100-year trajectory of wildfire simulations takes up to 7 hours to complete, which makes it very difficult to explore the policy space for many different reward functions. With fast simulations, we can support interactively changing the reward function and re-optimizing the results. We describe a method for quickly simulating wildfires after introducing WildFireAssistant.

1.3 WildFireAssistant for Forest Managers

Visual analytics combines automated analysis techniques with interactive visualizations for understanding, reasoning, and decision making (Keim et al., 2008). WildFireAssistant is a visual analytics system that meets the following requirements:

i. **Modifiability**: users should be able to modify the reward function to represent the interests of various stakeholders.
ii. **Automatic Optimization**: users should be able to optimize policies for the updated reward functions without the involvement of a machine learning researcher.
iii. **Visualization**: users should be able to visualize the system so that it is easy for users to explore the behavior of the system when it is controlled by the optimized policies.
iv. **Interactivity**: users should be able modify reward functions, optimize policies, and visualize the results many times within a single user session.

These requirements support a visual analytics knowledge generation model (Sacha et al., 2014) through a Parameter Space Analysis (PSA) process (Sedlmair et al., 2014). PSA is defined as a set of techniques for understanding the relationship between input parameters and outputs by systematically varying model input parameters, generating outputs for each combination of parameters, and investigating the relation

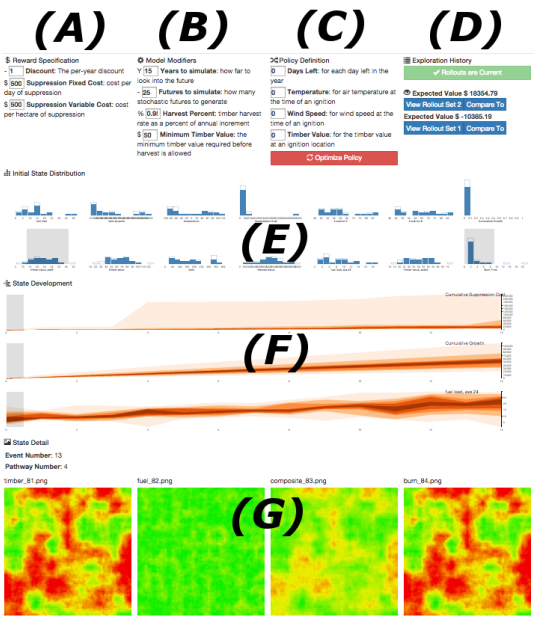


Figure 1.3 A high level overview of the Markov Decision Process visualization MDPvis. The top row has the three parameter controls for (A) the reward specification, (B) the model modifiers, and (C) the policy definition. A fourth panel gives the history of Monte Carlo trajectory sets generated under the parameters of panels (A) through (C). Changes to the parameters enable the optimization button found under the policy definition and the Monte Carlo trajectories button found under the Exploration History section. The visualization has two buttons in the History panel for each set of Monte Carlo trajectories, one for visualizing the associated Monte Carlo trajectories and another for comparing two sets of trajectories. Below the control panels are visualization areas for (E) histograms of the initial state distribution, (F) fan charts for the distribution of variables over time, and (G) a series of individual states rendered by the simulator as images. For a readable version of the visualization we invite users to load the visualization in their browser by visiting *MDPvis.github.io*.

between parameter settings and corresponding outputs. In the case of MDPs, the policy, transition, and reward functions can all be parameterized and subjected to PSA.

Throughout the design and development process, we worked closely with forestry economics collaborators to design WildFireAssistant as a version of the MDPvis visualization system shown in Figure 1.3.

MDPVIS includes parameter control areas and three visualization panels. The parameter controls allow the forest manager to change reward parameters and request a newly optimized policy as shown in Figure 1.1. The reward parameters and the policies they produce are saved in an exploration history for subsequent reference and generation of decision documentation.

The first two visualization areas show sets of trajectories and provides ways for applying filters to the current trajectory set. This supports a global-to-local (Sedlmair et al., 2014) exploration process where the user starts with an overview of all trajectories before drilling down into specific trajectories.

To render distributions of time series, we represent state variables as fan charts giving the deciles of each variable across trajectory timesteps. To produce the fan chart, we first plot a lightly colored area whose top and bottom represent the largest and smallest value of the variable in each time step. On top of this lightest color, we plot a series of colors increasing in darkness with nearness to the trajectory set's median value for the timestep.

Fan charts show both the diversity of outcomes and the probability of particular ranges of values. For instance, when a forest manager is examining the risk of a catastrophic fire, they can view the fan chart displaying the distribution of fire sizes through time. The worst-case scenarios will be shown by the maximum extent of the fan chart in each time step.

After generating trajectories for multiple parameter sets, the user can place the visualization into "comparison mode," which displays the difference between two trajectory sets in the visualization areas. When in comparison mode, the fan chart color extents are plotted by subtracting a color's maximum (minimum) extent from the corresponding maximum (minimum) extent of the other fan chart. By rendering the shift in decile boundaries the user can see how different policies shape the probability of different outcomes through time. When the policies are optimized for different reward functions, it is possible to examine how the outcomes predicted for different stakeholder groups differ when executing their optimized policies.

The final visualization area gives a "state detail" view, which is an extensible two dimensional array of images or videos shown at the bottom of MDPVIS. By supporting extensibility, it is possible to introduce ecology-specific visualizations like GIS maps.

To prototype the visualization we interacted with an idealized fast

wildfire simulator that supported **automatic optimization** and **Interactivity** (McGregor et al., 2016, 2015). Despite it being a simplistic wildfire domain, we discovered bugs in the fire spread implementation, harvest planner, random number generator, optimizer, and other components. Even after fixing these bugs the prototyping simulator is not sufficient for real world policy. In the next section we integrate the OCWS and restore *interactive* visualization through additional modeling effort.

1.4 Adding Interactivity to WildFireAssistant

We now turn to integrating the OCWS with WildFireAssistant. Our starting point is running the simulator for every new policy query, which takes up to 7 hours to simulate. How can we support interactivity when the simulator is so expensive?

Our approach is to develop a surrogate model that can substitute for the simulator. We start by designing a small set of "seed policies" and invoking the slow simulator to generate several 100-year trajectories for each policy. This gives us a database of state transitions of the form (s_t, a_t, r_t, s_{t+1}), where s_t is the state at time t, a_t is the selected action, r_t is the resulting reward, and s_{t+1} is the resulting state. Given a new policy π to visualize, we apply the method of Model-Free Monte Carlo (MFMC) (Fonteneau et al., 2013) to simulate trajectories for π by stitching together state transitions according to a specified distance metric Δ. Given a current state s and desired action $a = \pi(s)$, MFMC searches the database to find a tuple $(\tilde{s}, \tilde{a}, r, s')$ that minimizes the distance $\Delta((s, a), (\tilde{s}, \tilde{a}))$. It then uses s' as the resulting state and r as the corresponding one-step reward. We call this operation "stitching" (s, a) to $(\tilde{s}, \tilde{a})$. MFMC is guaranteed to give reasonable simulated trajectories under assumptions about the smoothness of the transition dynamics and reward function and provided that each matched tuple is removed from the database when it is used. Algorithm 1 provides the pseudocode for MFMC generating a single trajectory.

We selected the MFMC surrogate in lieu of two other approaches for quickly generating trajectories. First, we could write our own simulator for fire spread, timber harvest, weather, and vegetative growth that computes the state transitions more efficiently. For instance, Arca et al. (2013) use a custom-built model running on GPUs to calculate fire risk maps and mitigation strategies. However, developing a new simulator requires additional work to design, implement, and (especially) val-

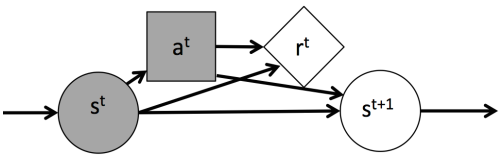
Algorithm 1: MFMC for a single trajectory. When generating multiple trajectories for a single trajectory set, the state transitions from D must not be reused (Fonteneau et al., 2010).

Input Parameters: Policy π, horizon h, starting state s_0, distance metric $\Delta(.,.)$, database D;
Returns: $(s,a,r)_1, ..., (s',a',r')_h$;
$t \leftarrow \emptyset$;
$s \leftarrow s_0$;
while *length(t)*$< h$ **do**
  $a \leftarrow \pi(s)$;
  $X \leftarrow \underset{(\widetilde{s},\widetilde{a},r,s') \in D}{\operatorname{argmin}} \Delta((s,a),(\widetilde{s},\widetilde{a}))$;
  $r \leftarrow X^r$;
  append$(t,(s,a,r))$;
  $s \leftarrow X^{s'}$;
  $D \leftarrow D \setminus X$;
end
return(t);

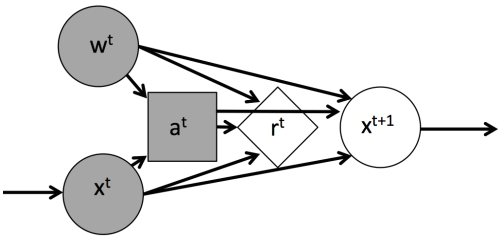
idate the simulator. This cost can easily overwhelm the resulting time savings when deploying to new conservation domains. Further, forest managers trust the models incorporated into the OCWS simulator. It is critically important that the WildfireAssistant employ highly-accurate state-of-the-art simulators in order to be trusted by decision makers.

A second approach would be to learn a parametric surrogate model from data generated by the slow simulator. For instance, Abbeel et al. (2005) learn helicopter dynamics by updating the parameters of a function designed specifically for helicopter flight. Designing a suitable parametric model that can capture weather, vegetation, fire spread, and the effect of fire suppression would require a major modeling effort.

The advantage of MFMC surrogates over these techniques is the ability to build the transition database from state transitions of simulators trusted in the forestry community. Further, it is possible to build on standard database technologies that are guaranteed to be fast. Despite these advantages, MFMC has never been applied to high-dimensional MDPS. In high-dimensional spaces (i.e., where the states and actions are described by many features), MFMC breaks because of two related problems. First, distances become less informative in high-dimensional



(a) The standard MDP transition.



(b) MDP transition with *exogenous* (w) and *Markovian* variables (x).

Figure 1.4 MDP probabilistic graphical models.

spaces. Second, the required number of seed-policy trajectories grows exponentially in the dimensionality of the space. Our main technical contribution is to introduce a modified algorithm, MFMCi, that reduces the dimensionality of the distance matching process by factoring out certain exogenous state variables and removing the features describing the action. In many applications, this can very substantially reduce the dimensionality of the matching process to the point that MFMC is again practical.

We now describe how we can factor the state variables of an MDP in order to reduce the dimensionality of the MFMC stitching computation. State variables can be divided into Markovian and Time-Independent random variables. A time-independent random variable x_t is exchangeable over time t and does not depend on any other random variable (including its own previous values). A (first-order) Markovian random variable x_{t+1} depends on its value x_t at the previous time step. In particular, the state variable s_{t+1} depends on s_t and the chosen action a_t. Variables can also be classified as endogenous and exogenous. The variable x_t is exogenous if its distribution is independent of $a_{t'}$ and $s_{t'} \setminus \{x'_t\}$ for all $t' \leq t$. Non-exogenous variables are endogenous. In conservation problems, Markovian variables are typically the state of the landscape from one time step to another.

Let us factor the MDP state s into two vectors of random variables: w, which contains the time-independent, exogenous state variables and x,

which contains all of the other state variables (see Figure 1.4). Time independent exogenous random variables can include wildfire ignitions, the introduction of an invasive species, macroeconomic conditions, weather, and government budgets. In our wildfire suppression domain, the state of the forest from one time step to another is Markovian, but our policy decisions also depend on exogenous weather events such as rain, wind, and lightning.

We can formalize this factorization as follows.

Definition 1.1 A Factored Exogenous MDP is an MDP such that the state (x, w) and next state (x', w') are related according to

$$Pr(x', w'|x, w, a) = Pr(w')Pr(x'|x, w, a). \tag{1.1}$$

This factorization allows us to avoid computing similarity in the complete state space s. Instead we only need to compute the similarity of the Markov state x. Without the factorization, MFMC stitches (s, a) to the $(\tilde{s}, \tilde{a})$ in the database D that minimizes a distance metric Δ, where Δ has the form $\Delta((s, a), (\tilde{s}, \tilde{a})) \mapsto \mathbb{R}^+$. Our new algorithm, MFMCi, makes its stitching decisions using only the Markov state. It stitches the current state x by finding the tuple $(\tilde{x}, \tilde{w}, a, r, x')$ that minimizes the lower-dimensional distance metric $\Delta_i(x, \tilde{x})$. MFMCi then adopts $(\tilde{x}, \tilde{w})$ as the current state, computes the policy action $\tilde{a} = \pi(\tilde{x}, \tilde{w})$, and then makes a transition to x' with reward r. The rationale for replacing x by $\tilde{x}$ is the same as in MFMC, namely that it is the nearest state from the database D. The rationale for replacing w by $\tilde{w}$ is that both w and $\tilde{w}$ are exchangeable draws from the exogenous time-independent distribution $P(w)$, so they can be swapped without changing the distribution of simulated paths.

There is one subtlety that can introduce bias into the simulated trajectories. What happens when the action $\tilde{a} = \pi(\tilde{x}, \tilde{w})$ is not equal to the action a in the database tuple $(\tilde{x}, \tilde{w}, a, r, x', w')$? One approach would be to require that $a = \tilde{a}$ and keep rejecting candidate tuples until we find one that satisfies this constraint, however, this method introduces a bias. Consider again the graphical model in Figure 1.4. When we use the value of a to decide whether to accept $\tilde{w}$, this couples $\tilde{w}$ and $\tilde{x}$ so that they are no longer independent.

We can remove this bias by changing how we generate the database D to ensure that for every state $(\tilde{x}, \tilde{w})$, there is always a tuple $(\tilde{x}, \tilde{w}, a, r, x', w')$ for every possible action a. To do this, as we execute a trajectory follow-

Algorithm 2: Populating D for MFMCi by sampling whole trajectories.

Input Parameters: Policy π, horizon h, trajectory count n, transition simulator f_x, reward simulator f_r, exogenous distribution $P(w)$, stochasticity distribution $P(z)$
Returns: nh transition sets B
$D \leftarrow \emptyset$
while $|D| < nh$ **do**
  $x = f_x(\cdot, \cdot, \cdot, \cdot)$ // Draw initial Markov state
  $l = 0$
  while $l < h$ **do**
    $B \leftarrow \emptyset$
    $w \sim P(w)$
    $z \sim P(z)$
    for $a \in A$ **do**
      $r \leftarrow f_r(x, a, w, z)$
      $x' \leftarrow f_x(x, a, w, z)$
      $B \leftarrow B \cup \{(x, w, a, r, x')\}$
    end
    append(D,B)
    $x \leftarrow B^{x'}_{\pi(x,w)}$
    $l \leftarrow l + 1$
  end
end
return(D)

ing policy π, we simulate the result state (x', w') and reward r for each possible action a and not just the action $a = \pi(x, w)$ dictated by the policy. This requires drawing more samples during database construction, but it restores the independence of $\tilde{w}$ from $\tilde{x}$.

It is convenient to collect together all the simulated successor states and rewards into *transition sets*. Let $B(x, w)$ denote the transition set of tuples $\{(x, w, a, x', r)\}$ generated by simulating each action a in state (x, w). Given a transition set B, it is useful to define selector notation as follows. Subscripts of B constrain the set of matching tuples and superscripts indicate which variable is extracted from the matching tuples. Hence, $B^{x'}_a$ denotes the result state x' for the tuple in B that matches

action a. With this notation, Algorithm 2 describes the process of populating the database with transition sets.

Algorithm 3: MFMCi

Input Parameters: Policy π, horizon h, starting state x_0, distance metric $\Delta_i(\cdot,\cdot)$, database D
Returns: $(x_0, w, a, r)_1, \ldots, (x', w', a', r')_h$
$t \leftarrow \emptyset$
$x \leftarrow x_0$
while *length(t)*$< h$ **do**
  $\hat{B} \leftarrow \text{argmin}_{B \in D} \Delta_i(x, B^{x'})$
  $\hat{w} \leftarrow \hat{B}^w$
  $a \leftarrow \pi(x, \hat{w})$
  $r \leftarrow \hat{B}^r_a$
  append$(t, (x, w, a, r))$
  $D \leftarrow D \setminus \hat{B}$
  $x \leftarrow B^{x'}_a$
end
return(t)

Algorithm 3 gives the pseudo-code for MFMCi. To estimate the cumulative return of policy π, we call MFMCi n times without database replacement and compute the mean of the cumulative rewards of the resulting trajectories.

Fonteneau et al. (2010) apply MFMC to estimate the expected cumulative return of a new policy π and derived bias and variance bounds on the MFMC value estimate. We reworked this derivation (McGregor et al., 2017a) to provide analogous bounds for MFMCi. The MFMC bounds depend on assuming Lipschitz smoothness of the state, policy, and reward spaces and bound the error that can be introduced at each time step by assuming the database contains transitions across the whole state-action space. MFMCi improves on these bounds by eliminating the need for the Lipschitz assumption on the policy space and exogenous state space variables.

With the MFMCi surrogate, it becomes possible to quickly generate whole trajectories from arbitrary policies. In our experimental section, we will empirically demonstrate how MFMCi improves on an MFMC baseline for visualizing policies within WildFireAssistant. Then we will optimize policies from the MFMCi surrogate with a fast black box opti-

mization method and evaluate the resulting policies directly within the OCWS.

1.5 Experimental Evaluation

In our experiments we evaluate the quality of trajectory visualizations and then an optimization method invoking the MFMCi surrogate.

1.5.1 Evaluation of Visualization Fidelity

We constructed three policy classes that map fire ignitions to fire suppression decisions. We label these policies INTENSITY, FUEL, and LOCATION. The INTENSITY policy suppresses fires based on the weather conditions at the time of the ignition and the number of days remaining in the fire season. The FUEL policy suppresses fires when the landscape accumulates sufficient high-fuel pixels. The LOCATION policy suppresses fires that are ignited in the top half of the landscape, but allows fires on the bottom half of the landscape to burn. These policies approximate scenarios in which the goal is to prevent catastrophic fires or the destruction of structures located on the north side of the landscape.

We focus sampling database states that likely to be entered by future policy queries by seeding the database with one trajectory for each of 360 policies whose parameters are chosen according to a grid over the INTENSITY policy space. The INTENSITY policy parameters include a measure of the weather conditions at the time of ignition known as the Energy Release Component (ERC) and a measure of the seasonal risk in the form of the calendar day. These measures are drawn from $\big[0, 100\big]$ and $\big[0, 180\big]$, respectively.

One of our goals is to determine whether MFMCi can generalize from state transitions generated from the INTENSITY policy to accurately simulate state transitions under the FUEL and LOCATION policies. We selected these policy classes because they are functions of different components of the Markov and exogenous state such that knowing the action selected within one of the policy classes will not indicate the action selected by the other policy classes. We evaluate MFMCi by generating 30 trajectories for each policy from the ground truth simulator.

For our distance metric Δ_i, we use a weighted Euclidean distance computed over the mean/variance standardized values of the following landscape features: *Canopy Closure, Canopy Height, Canopy Base Height,*

Canopy Bulk Density, Stand Density Index, High Fuel Count, and Stand Volume Age. All of these variables are given a weight of 1. An additional feature, the time step (*Year*), is added to the distance metric with a very large weight to ensure that MFMCi will only stitch from one state to another if the time steps match. Introducing this non-stationarity ensures we exactly capture landscape growth stages for all pixels that do not experience fire.

Our choice of distance metric features is motivated by our key insight that the risk profile (the likely size of a wildfire) and the vegetation profile (the total tree cover) are easy to capture in low dimensions. If we instead attempt to capture the likely size of a *specific* fire on a landscape, we need a distance metric that accounts for the exact spatial distribution of fuels on the landscape. Our distance metric successfully avoids directly modeling spatial complexity while still generating accurate visualizations and estimators of the discounted reward.

We quantitatively evaluate the quality of the surrogate MDPvis fan charts by defining visual fidelity error as the difference in vertical position between the MFMCi median and the Monte Carlo sampled median. More formally, we define $\text{error}(v, t)$ as the offset between the correct location of the median and its MFMCi-modeled location for state variable v in time step t. We normalize the error by the height of the fan chart for the rendered policy ($H_v(\pi)$). The weighted error is thus $\sum_{v \in S} \sum_{t=0}^{h} \frac{\text{error}(v,t)}{H_v(\pi)}$.

This error is measured for 20 variables related to the counts of burned pixels, fire suppression expenses, timber loss, timber harvest, and landscape ecology.

We evaluated the visual fidelity for both MFMC (including exogenous variables in the distance metric) and MFMCi. We also compare against two baselines that explore the upper and lower bounds of the visual error. First, we show that the lower bound on visual error is not zero. Although each policy has true quantile values at every time step, estimating these quantiles with 30 trajectories is inherently noisy. The computational expense of the ground truth simulator prevents estimating its visual fidelity error directly, instead we find the average achievable visual fidelity error by bootstrap resampling the 30 ground truth trajectories. Second, we check whether the error introduced by stitching is worse than visualizing a set of random database trajectories. Thus the bootstrap resample forms a lower bound on the error, and comparison to the random trajectories detects stitching failure.

Figures 1.6 and 1.7 plot the visual fidelity error when simulating a

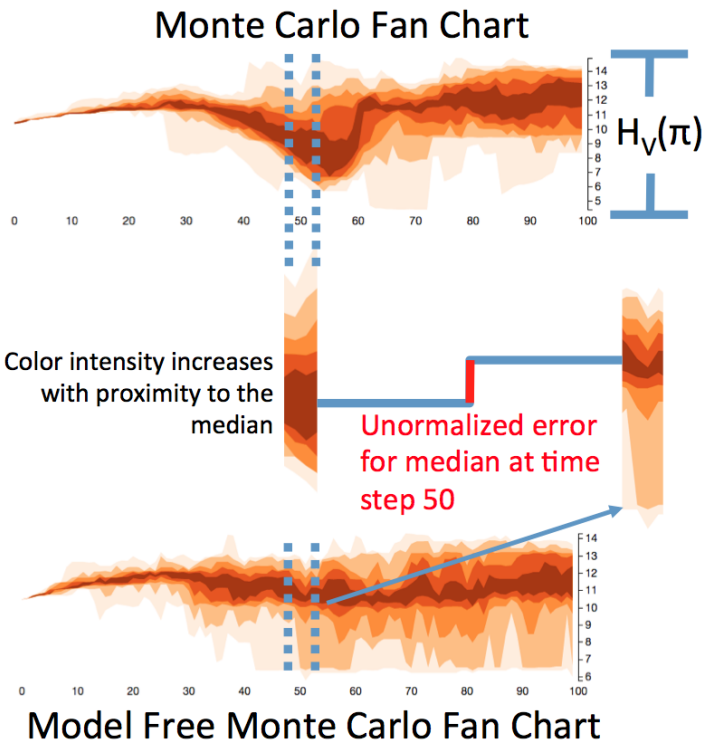


Figure 1.5 Top: A fan chart generated by Monte Carlo simulations from the expensive simulator. Bottom: A fan chart generated from the MFMC surrogate model. x axis is the time step and y axis is the value of the state variable at each time step. Each change in color shows a quantile boundary for a set of trajectories generated under policy π. Middle: Error measure is the distance between the median of the Monte Carlo simulations (left) and the median of the MFMC/MFMCi surrogate simulations (right). The error is normalized across fan charts according to $H_v(\pi)$, which is the Monte Carlo fan chart height for policy π and variable v.

LOCATION and FUEL policies, respectively, from the database of INTENSITY policy trajectories. The ideal learning curve should show a rapid decrease in visual fidelity error as $|D|$ grows. When D is small, the error is very high. MFMC is unable to reduce this error as D grows, because its distance metric does not find matching fire conditions for similar landscapes. In contrast, because the MFMCi methods are matching on the smaller set of Markov state variables, they are able to find good matching trajectories.

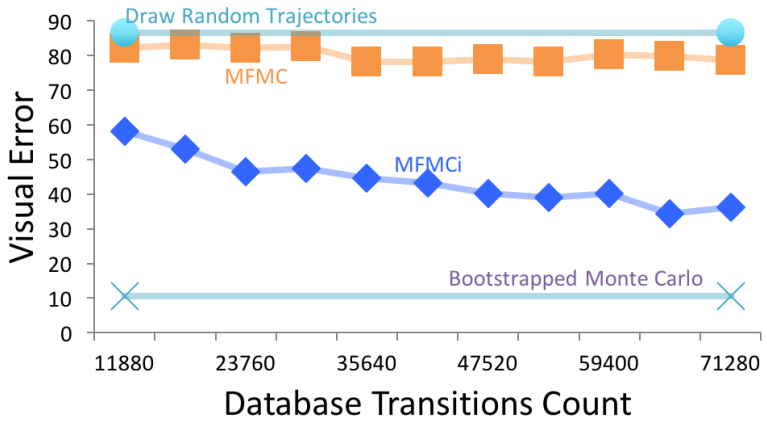


Figure 1.6 Visual fidelity errors for an ignition *location* policy class. Fires are always suppressed if they start on the top half of the landscape, otherwise they are always allowed to burn.

In summary, our experiments show that MFMCi generalizes across policy classes and that it requires only a small number of database trajectories to accurately reproduce the median of each state variable at each future time step. Having specified a surrogate that quickly generates trajectories, we now turn to supporting optimization from these trajectories.

1.5.2 Evaluation of Policy Optimization

WildFireAssistant requires an optimization method that is fast and can optimize any reward function specified by forest managers. We employ SMAC (Hutter et al., 2010), a black-box optimization algorithm originally developed for tuning the hyperparameters of algorithms.

SMAC is similar to Bayesian methods for black box optimization. However, unlike those methods, SMAC does not employ a Gaussian process to model the black box function. Instead, it fits a random forest ensemble (Breiman, 2001). This has three important benefits. First, it does not impose any smoothness assumption on the black box function. We will see below that wildfire policies are naturally expressed in the form of decision trees, so they are highly non-smooth. Second, SMAC does not require the design of a Gaussian process kernel. This makes it more suitable for application by end users such as forest managers. Third, the CPU time required for SMAC scales as $O(n \log n)$ where n is

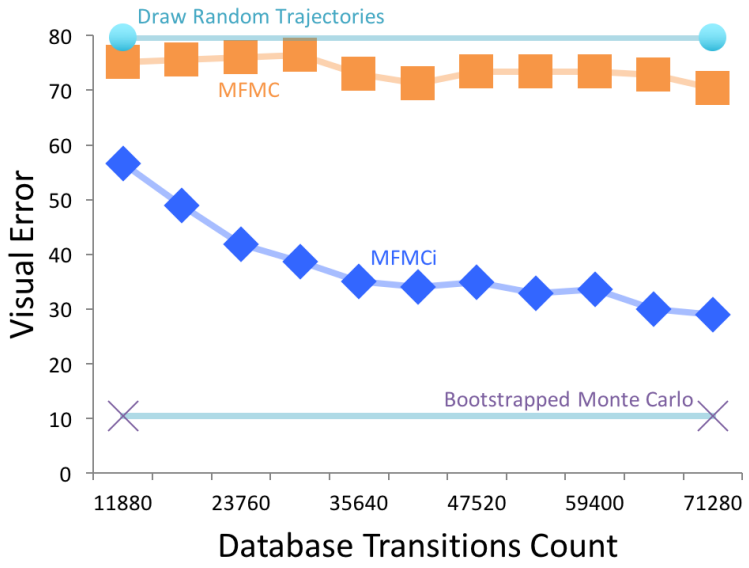


Figure 1.7 Visual fidelity errors for a *fuel* accumulation policy class. Fires are always suppressed if the landscape is at least 30 percent in high fuels, otherwise the fire is allowed to burn.

the number of evaluations of the black box function, whereas standard GP methods scale as $O(n^3)$ because they must compute and invert the kernel matrix.

Let Π be a class of deterministic policies with an associated parameter space Θ. Each parameter vector $\theta \in \Theta$ defines a policy $\pi_\theta : S \mapsto A$ that specifies what action to take in each state. Let $\tau = \langle s_0, s_1, \ldots, s_h \rangle$ be a trajectory generated by drawing a state $s_0 \sim P_0(s_0)$ according to the starting state distribution and then following policy π_θ for h steps. Let $\rho = \langle r_0, \ldots, r_{h-1} \rangle$ be the corresponding sequence of rewards. Both τ and ρ are random variables because they reflect the stochasticity of the starting state and the probability transition function. Let V_θ define the expected cumulative discounted return of applying policy π_θ starting in a state $s_0 \sim P_0(s_0)$ and executing it for h steps:

$$V_\theta = \mathbb{E}_\rho[r_0 + \gamma r_1 + \gamma^2 r_2 + \cdots + \gamma^{h-1} r_{h-1}]$$

The goal of "direct policy search" is to find θ^* that maximizes the value

of the corresponding policy:

$$\theta^* = \underset{\theta \in \Theta}{\operatorname{argmax}}\, V_\theta.$$

Sequential model-based optimization methods (Kushner, 1964; Mockus, 1994; Zilinskas, 1992; Hutter et al., 2010; Srinivas et al., 2010; Wilson et al., 2014; Wang et al., 2016) construct a model of V_θ called a Policy Value Model and denoted $PVM(\theta)$. The PVM estimates both the value of V_θ and a measure of the uncertainty of that value. The basic operation of sequential model-based optimization methods is to select a new point θ at which to query the PVM, observe the value of V_θ at that point (e.g., by simulating a trajectory τ using π_θ), and then update the PVM to reflect this new information. In Bayesian Optimization, the PVM is initialized with a prior distribution over possible policy value functions and then updated after gathering observations by applying Bayes rule. The new points θ are selected by invoking an *acquisition function*.

SMAC (Hutter et al., 2010) is a sequential model-based optimization method in which the PVM is a random forest of regression trees. The estimated value of V_θ is obtained by "dropping" θ through each of the regression trees until it reaches a leaf in each tree and then computing the mean and the variance of the training data points stored in all of those leaves. In each iteration, SMAC evaluates V_θ at 10 different values of θ, adds the observed values to its database R of (θ, V_θ) pairs, and then rebuilds the random forest.

SMAC chooses 5 of the 10 θ values with the goal of finding points that have high "generalized expected improvement". The (ordinary) expected improvement at point θ is the expected increase in the maximum value of the PVM that will be observed when we measure V_θ under the assumption that V_θ has a normal distribution whose mean is μ_θ (the current PVM estimate of the mean at θ) and whose variance is σ_θ^2 (the PVM estimate of the variance at θ). The expected improvement at θ can be computed as

$$EI(\theta) := \mathbb{E}\big[I(\theta)\big] = \sigma_\theta\big[z \cdot \Phi(z) + \phi(z)\big], \tag{1.2}$$

where $z := \frac{\mu_\theta - f_{max}}{\sigma_\theta}$, f_{max} is the largest known value of the current PVM, Φ denotes the cumulative distribution function of the standard normal distribution, and ϕ denotes the probability density function of the standard normal distribution (Jones et al., 1998).

The generalized expected improvement (GEI) is obtained by computing the expected value of $I(\theta)$ raised to the g-th power. In SMAC, g is

set to 2. Hutter et al. (2010) show that this can be computed as

$$GEI(\theta) = \mathbb{E}\big[I^2(\theta)\big] = \sigma_\theta^2\big[(z^2+1)\cdot\Phi(z) + z\cdot\phi(z)\big]. \tag{1.3}$$

Ideally, SMAC would find the value of θ that maximizes $GEI(\theta)$ and then evaluate V_θ at that point. However, this would require a search in the high-dimensional space of Θ, and it would also tend to focus on a small region of Θ. Instead, SMAC employs the following heuristic strategy to find 10 candidate values of θ. First, it performs a local search in the neighborhood of the 10 best known values of θ in the PVM. This provides 10 candidate values. Next, it randomly generates another 10,000 candidate θ vectors from Θ and evaluates the GEI of each of them. Finally, it chooses the 5 best points from these 10,010 candidates and 5 points sampled at random from the 10,000 random candidates, and evaluates V_θ at each of these 10 points. This procedure mixes "exploration" (the 5 random points) with "exploitation" (the 5 points with maximum GEI), and it has been found empirically to work well.

Hutter et al. 2010 prove that the SMAC PVM is a consistent estimator of V and that given an unbounded number of evaluations of V, it finds the optimal value θ^*.

In our experimental evaluation, we optimize and validate policies for the two different reward functions with components shown in Table 1.1. These two reward functions may approximate a home owner, and a proposed composite reward function that may be representative of some combination of home owner, timber company, naturalist, and recreational stakeholder interests.

	Suppression Costs	Timber Revenues	Ecology Target	Air Quality	Recreation Target
Composite	✓	✓	✓	✓	✓
Home Owners	-	-	-	✓	✓

Table 1.1 *Components of each reward function. The "home owner" constituency only cares about air quality and recreation. The "composite" reward function takes an unweighted sum of all the costs and revenues realized by forest stakeholders. Additional reward functions can be specified by forest managers interactively within WildFireAssistant.*

The reward functions are compositions of five different reward components. The *Suppression* component gives the expenses incurred for suppressing a fire. Fire suppression expenses increase with fire size and the number of days the fire burns. Without fire suppression effort, the fire suppression costs are zero, but the fire generally takes longer to self-extinguish. *Timber* harvest values sum the sale price for board feet of timber. *Ecological* value is a function of the squared deviation from an officially-specified target distribution of vegetation on the landscape known as the "restoration target." *Air Quality* is a function of the number of days a wildfire burns. When fires burn, the smoke results in a large number of home-owner complaints. We encode this as a negative reward for each smoky day. Finally, the *recreation* component penalizes the squared deviation from a second vegetation target distribution—namely, one preferred by people hiking and camping. If we optimize for any single reward component, the optimal policy will tend to be one of the trivial policies "suppress-all" or "letburn-all". When multiple reward components are included, the optimal policy still tends to either suppress or let burn most fires by default, but it tries to identify exceptional fires where the default should be overridden. See Houtman et al. (2013) for a discussion of this phenomenon.

A natural policy class in this domain takes the form of a binary decision tree as shown in Figure 1.8. At each level of the tree, the variable to split on is fixed in this policy class. With the exception of the very first split at the root, which has a hard-coded threshold, the splitting thresholds $\theta_1, \ldots, \theta_{14}$ are all adjusted by the policy optimization process. Moving from the top layer of the tree to the bottom, the tree splits first on whether the fire will be extinguished within the next 8 days by a "fire-ending weather event" (i.e., substantial rain or snowfall). The threshold value of 8 is fixed (based on discussions with forest managers and on the predictive scope of weather forecasts). The next layer splits on the state of fuel accumulation on the landscape. The fuel level is compared either to θ_1 (left branch, no weather event predicted within 8 days) or θ_2 (right branch; weather predicted within 8 days). When the fuel level is larger than the corresponding threshold, the right branch of the tree is taken. The next layer splits on the intensity of the fire at the time of the ignition. In this fire model, the fire intensity is quantified in terms of the Energy Release Component (ERC), which is a composite measure of dryness in fuels. Finally, the last layer of the tree asks whether the current date is close to the start or end of the fire season. Our study region in Eastern Oregon is prone to late spring and early fall rains,

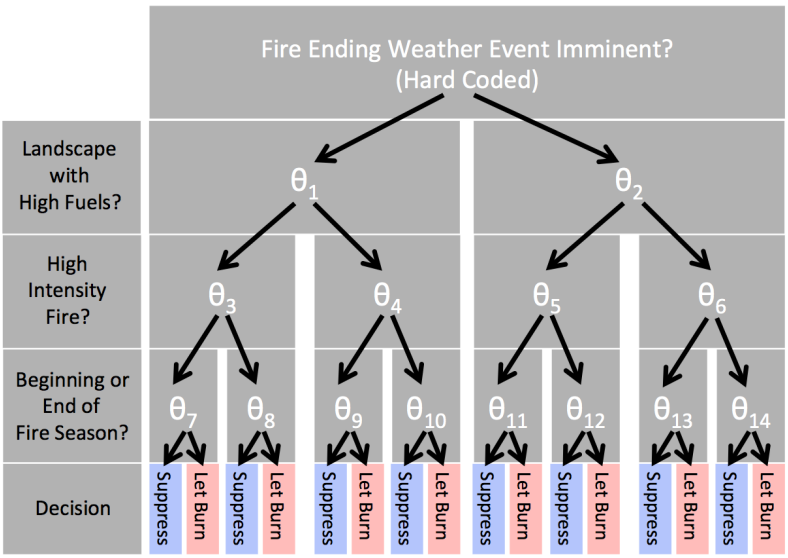


Figure 1.8 The layers of the decision tree used to select wildfires for suppression. The top layer splits on whether the fire will likely be extinguished by a storm in the next 8 days regardless of the suppression decision. The next layers then have 14 parameters for the number of pixels that are in high fuel (parameters 1 and 2, $[0, 1000000]$), the intensity of the weather conditions at the time of the fire (3 through 6, $[0, 100]$), and a threshold that determines whether the fire is close to either the start or end of the fire season (7 through 14, $[0, 90]$).

which means fires near the boundary of the fire season are less likely burn very long.

We apply SMAC with its default configuration. Optimization details are provided in McGregor, et al. (2017b). Every query to the MFMCi surrogate generates 30 trajectories, and the mean cumulative discounted reward of these is returned as the observed value of V_θ.

Figure 1.9 shows the results of applying SMAC to find optimal policies for the two reward functions. The left column of plots show the order in which SMAC explores the policy parameter space. The vertical axis is the estimated cumulative discounted reward, and the point that is highest denotes the final policy output by SMAC. Line dashes are policy parameter vectors chosen by the GEI acquisition function whereas circles are parameter vectors chosen by SMAC's random sampling process. Notice that in all cases, SMAC rapidly finds a fairly good policy. The right column of plots gives us some sense of how the different policies

behave. Each plot shows the percentage of fires that each policy suppresses. In 1.9(b), we see that the highest-scoring policies allow almost all fires to burn, whereas in 1.9(d), the highest-scoring policies suppress about 80% of the fires.

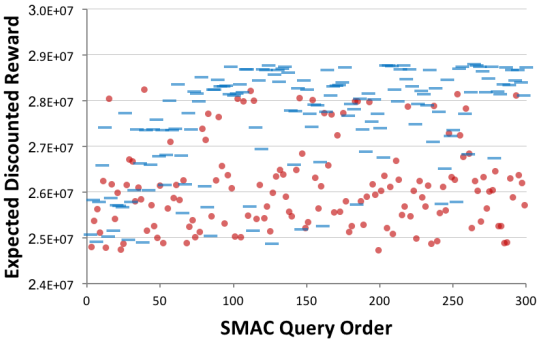
Let us examine these policies in more detail. The optimal policy for the *composite* reward allows most wildfires to burn, but the *home owner* reward policy suppresses most wildfires. The *home owner* constituency reward function seeks to minimize smoke (which suggests suppressing all fires) and maximize recreation value (which suggests allowing fires to burn the understory occasionally). We can see in 1.9(d) that the best policy found by SMAC allows 20% of fires to burn and suppresses the remaining 80%.

These results agree with our intuition and provide evidence that SMAC is succeeding in optimizing these policies. However, the expected discounted rewards in Figure 1.9 are estimates obtained from the modified MFMC surrogate model. To check that these estimates are valid, we invoked each optimized policy on the full simulator at least 50 times and measured the cumulative discounted reward under each of the reward functions. We also evaluated the Suppress All and Let Burn All policies for comparison. The results are shown in Figure 1.10.

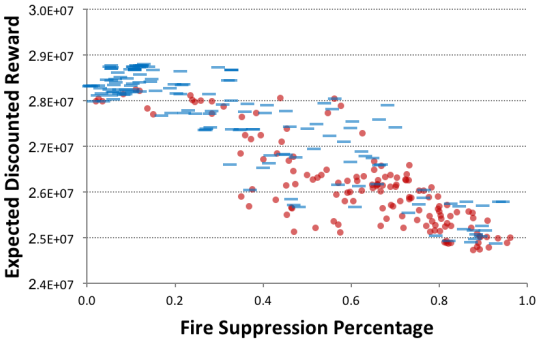
Each panel of boxplots depicts the range of cumulative returns for each policy on one reward function. For the policy that SMAC constructed, we also plot a dashed red line showing the MFMCi estimate of the return. In all cases, the estimates are systematically biased high. This is to be expected, because any optimization against a noisy function will tend to exploit the noise and, hence, over-estimate the true value of the function. Nonetheless, the MFMCi estimates all fall near the inter-quartile range of the full simulator estimates. This confirms that the MFMC estimates are giving an accurate picture of the true rewards.

1.6 Conclusion

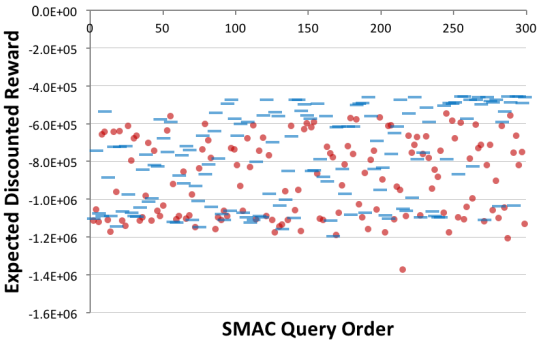
Previous work on high-fidelity wildfire management (Houtman et al., 2013) has focused only on *policy evaluation*, in which the full simulator was applied to evaluate a handful of alternative policies. We report the first successful *policy optimization* for wildfire suppression at scale. Moreover, the MFMCi surrogate executes quickly enough to support optimization within WildFireAssistant, which provides a basis for forest



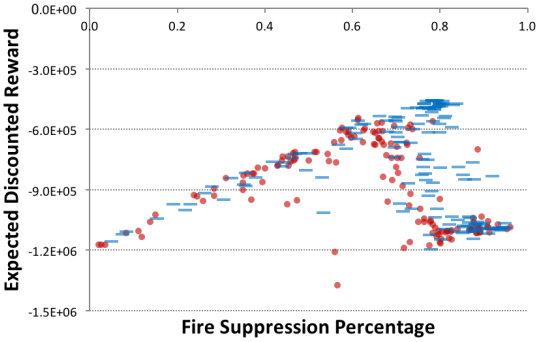
(a) Composite reward function.



(b) Composite reward function.

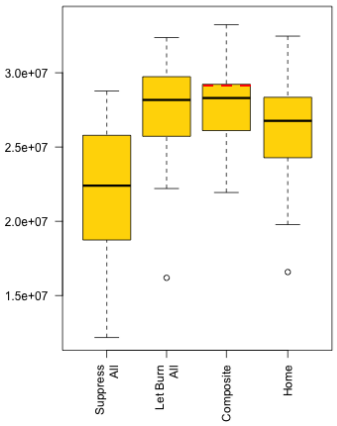


(c) Home owner reward function.

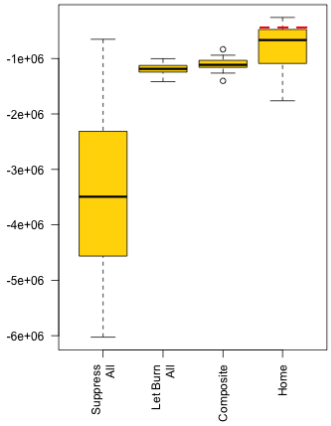


(d) Home owners reward function.

Figure 1.9 Average reward achieved for 30 trajectories. Horizontal lines are selected by the EI heuristic and circles are randomly sampled points.



(a) Policies for composite reward function.



(b) Policies for home owner reward function.

Figure 1.10 Each set of box charts show the performance of various policies for a constituency. The individual box plots show the expected discounted reward for each of the policies optimized for a constituency, as well as the hard-coded policies of suppress-all and let-burn-all. The dashed lines indicate the expected discounted reward estimated by MFMCi.

managers to discover the long term consequences of present-day decisions.

Our forestry economics collaborators inform us that WildFireAssistant has greatly accelerated development of their MDP research. For future work we would like to deploy MDPvis to the US Forest Service. We believe we can build on our proof-of-concept by optimizing for entire US Forest Service regions.

Acknowledgment

This material is based upon work supported by the National Science Foundation under grant numbers 1331932 and 0832804. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

Abbeel, Pieter, Ganapathi, Varun, and Ng, Andrew Y. 2005. Learning Vehicular Dynamics, with Application to Modeling Helicopters. *Advances in Neural Information Processing Systems (NIPS)*, 1–8.

Ager, Alan A., Finney, Mark A., Kerns, Becky K., and Maffei, Helen. 2007. Modeling wildfire risk to northern spotted owl (Strix occidentalis caurina) habitat in Central Oregon, USA. *Forest Ecology and Management*, **246**(1 SPEC. ISS.), 45–56.

Arca, Bachisio, Ghisu, Tiziano, Spataro, William, and Trunfio, Giuseppe a. 2013. GPU-accelerated Optimization of Fuel Treatments for Mitigating Wildfire Hazard. *Procedia Computer Science*, **18**, 966–975.

Bellman, Richard. 1957. *Dynamic Programming*. New Jersey: Princeton University Press.

Breiman, Leo. 2001. Random Forests. *Machine learning*, **45**(1), 5–32.

Dilkina, Bistra, Houtman, Rachel, Gomes, Carla P, Montgomery, Claire A, Mckelvey, Kevin S, Kendall, Katherine, Graves, Tabitha A, Bernstein, Richard, and Schwartz, Michael K. 2016. Trade-offs and efficiencies in optimal budget-constrained multispecies corridor networks. *Conservation Biology*, **31**(1), 192–202.

Dixon, G. 2002. *Essential FVS : A User's Guide to the Forest Vegetation Simulator*. Fort Collins, CO: USDA Forest Service.

Egan, Timothy. 2005 (December). *6 Arrested Years After Ecoterrorist Acts*.

Endert, Alex, Hossain, M. Shahriar, Ramakrishnan, Naren, North, Chris, Fiaux, Patrick, and Andrews, Christopher. 2014. The human is the loop: new directions for visual analytics. *Journal of Intelligent Information Systems*, **43**(3), 411–435.

Finney, Mark A. 1998. *FARSITE: fire area simulator model development and evaluation*. Missoula, MT: USDA Forest Service, Rocky Mountain Research Station.

Fonteneau, Raphael, Murphy, Susan A, Wehenkel, Louis, and Ernst, Damien. 2010. Model-Free Monte Carlo-like Policy Evaluation. *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics (AISTATS 2010)*, 217–224.

Fonteneau, Raphael, Murphy, Susan a, Wehenkel, Louis, and Ernst, Damien. 2013. Batch Mode Reinforcement Learning based on the Synthesis of Artificial Trajectories. *Annals of Operations Research*, **208**(1), 383–416.

Forest History Society. 2017. *U.S. Forest Service Fire Suppression.*

Houtman, Rachel M., Montgomery, Claire A., Gagnon, Aaron R., Calkin, David E., Dietterich, Thomas G., McGregor, Sean, and Crowley, Mark. 2013. Allowing a Wildfire to Burn: Estimating the Effect on Future Fire Suppression Costs. *International Journal of Wildland Fire*, **22**(7), 871–882.

Hutter, Frank, Hoos, Holger H., and Leyton-Brown, Kevin. 2010. Sequential Model-Based Optimization for General Algorithm Configuration (extended version). *University of British Columbia, Department of Computer Science*, 507–523.

Jones, Donald R, Schonlau, Matthias, and Welch, William J. 1998. Efficient Global Optimization of Expensive Black-Box Functions. *Journal of Global Optimization*, **13**, 455–492.

Keim, Daniel, Andrienko, Gennady, Fekete, Jean-daniel, Carsten, G, Melan, Guy, Keim, Daniel, Andrienko, Gennady, Fekete, Jean-daniel, and Carsten, G. 2008. Visual Analytics : Definition , Process and Challenges To cite this version : Visual Analytics : Definition , Process , and Challenges. *Information Visualization - Human-Centered Issues and Perspectives*, 154–175.

Kushner, H. J. 1964. A new method of locating the maximum point of an arbitrary multipeak curve in the presence of noise. *Journal of Basic Engineering*, **86**, 97–106.

McGregor, Sean, Buckingham, Hailey, Dietterich, Thomas G., Houtman, Rachel, Montgomery, Claire, and Metoyer, Ron. 2015. Facilitating Testing and Debugging of Markov Decision Processes with Interactive Visualization. In: *IEEE Symposium on Visual Languages and Human-Centric Computing.*

McGregor, Sean, Buckingham, Hailey, Dietterich, Thomas G., Houtman, Rachel, Montgomery, Claire, and Metoyer, Ronald. 2016. Interactive visualization for testing Markov Decision Processes: MDPVIS. *Journal of Visual Languages and Computing.*

McGregor, Sean, Houtman, Rachel, Montgomery, Claire, Metoyer, Ronald, and Dietterich, Thomas G. 2017a. Factoring Exogenous State for Model-Free Monte Carlo. *ArXiv.*

McGregor, Sean, Houtman, Rachel, Montgomery, Claire, Metoyer, Ronald, and Dietterich, Thomas G. 2017b. Fast Optimization of Wildfire Suppression Policies with SMAC. *ArXiv.*

Mockus, J. 1994. Application of Bayesian approach to numerical methods of global and stochastic optimization. *Journal of Global Optimization*, **4**, 347–365.

National Interagency Fire Center. 2009. Guidance for Implementation of Federal Wildland Fire Management Policy. *Federal Guidance*, 1–20.

Puterman, Martin. 1994. *Markov Decision Processes: Discrete Stochastic Dynamic Programming.* 1st edn. Wiley-Interscience.

Sacha, D, Stoffel, a, Stoffel, F, Kwon, B, Ellis, G, and Keim, D. 2014. Knowledge Generation Model for Visual Analytics. *Visualization and Computer Graphics, IEEE Transactions on*, **PP**(99), 1.

Sedlmair, M., Heinzl, C., Bruckner, S., Piringer, H., and Möller, T. 2014. Visual parameter space analysis: A conceptual framework. *IEEE Transactions on Visualization and Computer Graphics*, **20**(12).

Srinivas, Niranjan, Krause, Andreas, Kakade, Sham M., and Seeger, Matthias. 2010. Gaussian Process Optimization in the Bandit Setting: No Regret and Experimental Design. *Proceedings of the 27th International Conference on Machine Learning (ICML 2010)*, 1015–1022.

Tephens, Scott L, and Ruth, Lawrence W. 2005. Federal Forest-Fire Policy in the United States. *Ecological Applications*, **15**(March 2004), 532–542.

The National Wildfire Coordinating Group. 2009. S-520 Advanced Incident Management. *Manual for Wildland Fire Decision Support System (WFDSS)*, 1–8.

United Nations. 2017. *Goal 15 : Sustainably manage forests, combat desertification, halt and reverse land degradation, halt biodiversity loss.*

United States Department of Agriculture. 2015. The Rising Cost of Fire Operations: Effects on the Forest Services Non-Fire Work. *United States Department of Agriculture*, 13.

Wang, Ziyu, Hutter, Frank, Zoghi, Masrour, Matheson, David, and de Freitas, Nando. 2016. Bayesian optimization in high dimensions via random embeddings. *Journal of Artificial Intelligence Research*, **55**, 361–387.

Western Regional Climate Center. 2011. *Remote Automated Weather Stations (RAWS).* Reno, NV: Western Regional Climate Center.

Wilson, Aaron, Fern, Alan, and Tadepalli, P. 2014. Using Trajectory Data to Improve Bayesian Optimization for Reinforcement Learning. *Journal of Machine Learning Research*, **15**, 253–282.

Zilinskas, A. 1992. A review of statistical models for global optimization. *Journal of Global Optimization*, **2**, 145–153.