

Machine Learning applied to Prediction, Control and Planning from Dynamic Epidemiological Models



Oliver Bent
Somerville College
University of Oxford

A thesis submitted for the degree of
Doctorate in Philosophy

Michaelmas 2020

To Leif, for the love of your parents and your life unlived

Acknowledgements

Firstly, thank you to Stephen Roberts for all your untiring support and encouragement in every aspect of this work.

I've also had the pleasure of interacting with many groups of researchers in bringing together this thesis, I would not have imagined how the seed of an initial idea may have been cultivated by my interactions with you, specifically I've learnt a lot from those within the malaria modelling consortium, thank you for everything you have taught me. Professionally, thank you Aisha for nurturing an idea and continuing to mother the results, and thank you Sekou for being an unwavering gardener for so much of this work.

On a more personal level, my new family has changed the world for me, and my daughter Eva has taught me so much outside what is written here. Doriane, while we were not prepared, we have been in this together and I thank you from the bottom of my heart for your unwavering patience. We have many more unwritten chapters to begin.

Thank you Pascal and Thierry for taking in a desolate English student over Christmas and providing an endless supply of chocolate for me to do battle with during this work.

To my parents Manti and Paul, thank you for your care and joy for things I wish I could experience with you more often, and for always being there for me. You have been in my thoughts during isolation restrictions in our respective countries, however my childhood memories have been so important to me during this time, and this has often saved me.

Lastly, I have to acknowledge 2020 has been a very difficult year, a trial and a call to arms. There is so much we may have lost but you have all shown me we can and we will adapt as there is an inner human resilience, although the author acknowledges this is easier said than done.

Abstract

We are ever aware of the global impact of infectious disease transmission in shaping the reality of the world around us. For much of human existence transmitted infections have been the principle cause for loss of human life. Through their very nature, the transient threats of pandemics or endemic disease transmission remain a constant challenge to epidemiologists and are also at the forefront of shaping public policy decisions.

In order to bring the benefits of Machine learning (ML) to tasks such as optimal prediction, control and planning from epidemiological models of disease transmission, we need to define how principles, advancements and knowledge can be directed towards real-world progress.

This work outlines steps towards making better decisions based on learning from epidemiological models under uncertainty. It considers how we can automate and scale epidemiological model value using Machine learning. The thesis investigates a range of methods from Multi-Armed Bandit approaches to sequential decision making using Reinforcement learning (RL) based on simulation. We use Gaussian Process Regression for sample efficiency in learning value-functions, improving model predictions via Bayesian optimisation of epidemiological model parameters and develop surrogate model descriptions for sharing, abstraction and combination of models as learning environments. Finally, the necessary computational infrastructure for sharing and running models at scale is considered.

Contents

Glossary	1
Acronyms	3
1 Background	5
1.1 Motivation	6
1.2 Mathematical Modelling in Epidemiology	9
1.3 Uncertainty Quantification	12
1.4 Model-Based Decision Making	15
1.4.1 Malaria Elimination and Control Background	16
1.4.2 Potential of Machine Learning	17
1.5 Contributions of Thesis and Outline	18
2 Exploring One-Step Prediction and Control with Online Learning	19
2.1 Online Learning	19
2.2 Stochastic Multi-Armed Bandit	21
2.2.1 Regret	22
2.2.2 Infinitely-Many Armed Bandits	23
2.2.3 Optimism in the Face of Uncertainty	24
2.3 Running Epidemiological Simulations as Bandits	24
2.3.1 Time	25
2.3.2 State	26
2.3.3 Action	27
2.3.4 Reward	27
2.3.5 Bandit Approaches for Simulated Learning	30
2.4 Example: Bandit Experiments on OpenMalaria Simulations	34
2.4.1 Simulated Interventions and Outputs	34
2.4.2 System Implementation and Deployment	37
2.4.3 Comparison with Results from Expert Humans	37

2.4.4	Evaluation	41
2.5	Extensions to Online Learning	42
3	Learning Models for Prediction from Epidemiological Simulation	44
3.1	Machine Learning Surrogates of Epidemiological Models	47
3.1.1	Gaussian Process Regression	48
3.1.2	Action-Value Function Example	53
3.2	Epidemiological Model Calibration	55
3.2.1	Bayesian Optimisation	56
3.2.2	Loss Function	58
3.3	Uncertainty quantification (UQ)	59
3.3.1	Sensitivity Analysis	60
3.3.2	Calibration Schedule	62
3.4	Example: Calibration and Uncertainty Quantification of EMOD Malaria models	62
3.4.1	Calibration	63
3.4.2	Uncertainty Quantification and Prediction	64
3.4.3	Sensitivity Analysis	64
4	Infrastructure and Computation for Prediction, Control and Plan- ning from Epidemiological Models	69
4.1	An Initial Computational Approach to Epidemiological Simulation . .	70
4.1.1	Engineering and Deployment Challenges	71
4.2	A Revised Infrastructure	72
4.2.1	Terms	72
4.2.2	Design Principles	73
4.2.3	Platform Components	75
4.3	Performance Results	79
4.4	Example: Technical Approach for Competitions	83
4.4.1	Environment Web Service	83
4.4.2	Competition API	84
4.4.3	Example Code	86
4.5	Related Work	87

5	Machine Learning for Epidemiological Planning via Competition	89
5.1	Planning from Learnt Models	91
5.2	KDD Cup 2019 and Other Competitions	94
5.2.1	Evaluation Framework	95
5.2.2	Submission and Scoring	97
5.2.3	Other Competitions	100
5.3	Collaboration from Competition	103
5.3.1	Learning Surrogates from Multiple Epidemiological Models . .	103
5.3.2	Covariance-based Weighted Committee	104
5.3.3	Consideration of Other Ensemble Methods	106
5.4	Example: Analysis of Successful Competition Approaches	106
5.5	Conclusions on Competitions	109
5.5.1	Human-in-the-loop	109
6	Multi-Step Prediction and Control with Reinforcement Learning	110
6.1	Reinforcement Learning	111
6.1.1	State	113
6.1.2	Action	113
6.1.3	Reward	114
6.1.4	Markov Decision Processes	114
6.1.5	Cumulated Return	115
6.1.6	Policy	116
6.1.7	Value Function	116
6.1.8	Partially Observable Markov Decision Process (POMDP) . . .	118
6.2	Learning Sequential Surrogate Models from Episodic Simulators . . .	118
6.2.1	Prediction - Learning a Value Function	120
6.2.2	Simulation Based Search - Decision Trees	121
6.2.3	Monte Carlo Tree Search (MCTS)	123
6.2.4	Gaussian Process Regression with Upper Confidence Bound selection in Monte Carlo Tree Search for Learning Sequential Surrogates (GP-UCB-MCTS)	124
6.3	Control - Optimal Policies and Planning	128
6.3.1	Optimal Policy Learning	129
6.4	Comparing Predictions from Multi-Step and One-Step Methods with direct Experience	132

6.5	Example: Model-free Reinforcement learning with Covid-19 Compartmental Model	133
6.5.1	Lives Saved and NPI Cost	134
6.5.2	Results of Agent Approaches to Epidemiological Control . . .	135
7	Conclusions and Related Work	137
7.1	Discussion	137
7.2	Related Work	138
7.3	Future Directions	139
	Bibliography	141

List of Figures

1.1	Yearly new cases of malaria per 1,000 people [129] in malaria endemic regions.	7
1.2	Flattening the Curve, health-system preparedness for viral pandemics [76].	7
1.3	Four Examples of current global epidemics under various states of control.	10
	(a) Global Malaria Deaths by Age Group [164]	10
	(b) Global HIV Deaths by Age Group [165]	10
	(c) Global Covid-19 Deaths by Continent [122]	10
	(d) Global Paralytic Polio Cases by Continent [138]	10
1.4	Model schematics for a single disease; malaria, with varying model approaches and types.	13
	(a) MASH extended Ross-Macdonald model [182]	13
	(b) Optima compartmental malaria model [176]	13
	(c) OpenMalaria deterministic vector model [108]	13
	(d) EMOD agent based model [68]	13
2.1	Action a_t selected by a Bandit algorithm receives feedback or reinforcement via reward r_t based on the epidemiological simulators outputs .	25
2.2	A batch of arm pulls for one-step decision making, the state transitions $\mathcal{S}_t \rightarrow \mathcal{S}_{t+1}$ improve estimates of the underlying continuous action value function $Q(a)$, the underlying techniques for inference are discussed further in Chapter 3.	26
2.3	Comparison of GP-ULCB Samples over subsequent batches from Open-Malaria Simulations for the problem outlined in Section 2.4.	32
	(a) First (Random) Batch	32
	(b) Third Batch	32
	(c) Fifth Batch	32
	(d) Seventh Batch	32

2.4	Boxplots indicating reward distributions ($\mathbb{P}[r a]$) of a fixed number of actions run through OpenMalaria simulation rounds as Bandit arms. Rewards are relative, based on the percentage improvement over the currently implemented deterministic policy.	35
2.5	Comparison of Bandit Approaches on OpenMalaria Simulations and their Top 3 Policies. $Q(\mathbf{a})$ mean action-value surfaces ($m(\mathbf{a})$)	39
(a)	Current and Expert Human recommended policies	39
(b)	GP-ULCB	39
(c)	Genetic Algorithm	39
(d)	Batch Policy Gradient	39
(e)	UCB	39
2.6	Slices of action-value function $Q(a)$ for actions (a) as a combination of Insecticide Treated Net and Indoor Residual Spraying coverage changing through time (days), for OpenMalaria Simulations	43
3.1	Multiple model ensemble predictions generated for Covid-19 model submissions in the United States of America, by the Centres for Disease Control and Prevention (CDC)	46
3.2	Comparison of posterior with selected (a) and learnt (b) MLE parameters	52
(a)	Posterior GP inference based on underlying epidemiological function and judicious choice of $\mathcal{GP}$ parameters $\theta_{\mathcal{GP}}$: $l = 0.25$, $\sigma_f^2 = 100$, $\sigma_n^2 = 0.1$. Including 5 draws from the posterior distribution of functions	52
(b)	Based on maximised log likelihood optimisation of GP parameters: $\mathcal{GP}$ parameters $\theta_{\mathcal{GP}}$: $l = 0.38$, $\sigma_f^2 = 1320$, $\sigma_n^2 = 0.02$. Including 5 draws from the posterior distribution	52
3.3	Learnt $\mathcal{GP}$ parameters $\theta_{\mathcal{GP}}$: $l = 0.64$, $\sigma_f^2 = 3380$, $\sigma_n^2 = 15.6$	54
(a)	$Q(\mathbf{a})$ mean function for OpenMalaria Simulation experiments [39]	54
(b)	$Q(\mathbf{a})$ standard deviation bound for OpenMalaria Simulation experiments	54
3.4	Calculation of Expected Improvement for Learnt $\mathcal{GP}$ parameters $\theta_{\mathcal{GP}}$: $l = 0.38$, $\sigma_f^2 = 1320$, $\sigma_n^2 = 0.02$, EI is maximised at $x = 1$	58
3.5	Visualisations of the samples generated across the parameter set during a short run-calibration	65

(a)	Calibration improvements sequence using Bayesian Optimisation, for Namawala Tanzania. Blue dots: loss function score, Red line: minimum loss.	65
(b)	Bayesian Optimisation Calibration Samples for Namawala Tanzania, projected onto 2 components by PCA	65
3.6	Comparison of full and vector EMOD models, based on their generated monthly Entomological Inoculation Rate (EIR) distributions, against the provided observed points of monthly Entomological Inoculation Rate (EIR) in Namawala, Tanzania.	66
(a)	Namawala Modelled Entomological Inoculation Rate (EIR) Vector Model	66
(b)	Namawala Modelled Entomological Inoculation Rate (EIR) Malaria Model	66
3.7	Comparison of full and vector models with learnt parameters	67
(a)	5 Year Predictions for Daily Infections for Location from Calibrated Vector Model, Learnt $\mathcal{GP}$ parameters $\theta_{\mathcal{GP}}$: $l = 0.10$, $\sigma_f^2 = 586$, $\sigma_n^2 = 36.6$	67
(b)	5 Year Predictions for Daily Infections for Location from Calibrated Malaria Model, Learnt $\mathcal{GP}$ parameters $\theta_{\mathcal{GP}}$: $l = 0.06$, $\sigma_f^2 = 1820$, $\sigma_n^2 = 69.1$	67
3.8	Histograms of parameter θ_m samples during calibration	68
4.1	Conceptual architecture of a platform for Machine learning based prediction, control and planning from epidemiological models [210].	75
4.2	Dashboard view showing generated intervention plans from two different epidemiological models for the same location, with the ability to filter out intervention plans using prevalence, budget, and the number of deaths averted, in order to meet the strategic plan.	79
4.3	Combined results from [32] showing AI (optimal), Current, and Human Expert recommended solutions.	80
4.4	Performance of a single cluster, with increasing numbers of jobs and workers. Three different resolutions correspond to three levels of computational expense in the simulation.	81
4.5	Blockchain throughput and average latency performance	82

5.1	Inspired by the work Planning, Learning and Acting [194]. <i>Blue</i> : Chapters 2 and 6 refer to learning a value function and policy based on simulated experience from an epidemiological model. <i>Yellow</i> : Chapters 3 and 6 refer to the task of prediction and necessarily learning a model based on epidemiological simulations and experience. <i>Red</i> : this Chapter (5) discusses the task of planning from a model to learn a policy which in turn may be used for further simulations to generate more experience.	93
5.2	Sequential Decision Making Task - Competition Framing	95
5.3	KDD Cup analysis	99
	(a) Number of daily CSV file submissions up until the submission of code for verification and final scoring.	99
	(b) Cumulative normalised scores of top 10 teams on all competition environments (additionally included for comparison the relative performance of algorithm GP-UCB-MCTS detailed in Section 6.2.4).	99
5.4	Two model covariance weighted prediction of daily malaria infections in Namawala, Tanzania.	105
5.5	Sample of successful approaches for the malaria competition sequential decision making task	107
	(a) Bayesian optimisation with a forward boosting network [135]	107
	(b) Thompson sampling bandit method	107
	(c) Fast search with prior particles (FSPP)	107
	(d) Gridded random search	107
5.6	Two dimensional histograms for the yearly actions generated by competitor agent submissions during the verification phase (Section 5.2.1.3)	108
6.1	Reinforcement Learning in Environments of epidemiological model simulations. Reward $R_t = r_t$, State $S_t = s_t$, Action $A_t = a_t$	112
6.2	Example of cumulated return as a two dimensional response surface, as a result of a sequence of decisions for 5 years.	123
6.3	Visual description of generation of sequential surrogate $Q(S, A)$ with regards to the mean $m(\mathbf{a})$ for each state.	127
	(a) Discretised 2D action-value function for $Q(\mathbf{a})$ showing mean $m(\mathbf{a})$	127
	(b) Transitions based on branching (B) of order three for High, Medium and Low value actions	127

6.4	Value-based Q-learning, Policy-based Reinforce, and Actor-Critic (A2C) approaches to learning from a sequential malaria surrogate competition environment.	132
6.5	Ability of ‘Best’ multi-step policies (GP-UCB-MCTS Section 6.2.4) to consistently outperform static policies (GP-ULCB Section 2.3.5.1) for the same simulation environment	133
6.6	Compartmental Epidemiological model SIRS-D used for this Covid-19 model-free learning example [43].	134
6.7	Normalised episodic reward (y-axis) of greedy policies learnt by different agent algorithms on a compartmental Covid-19 model environment, over 20 runs of 1000 episodes	136

List of Tables

2.1	Top three actions with respect to C_{DA} evaluated by each bandit model. Action: $\{a_{ITN}\%, a_{IRS}\%\}$, C_{DA} : Cost per DALY Averted USD, DA: DALYs Averted, C_{int} : Intervention Costs USD.	40
3.1	Best Parameter Sets for Respective Models: Namawala Tanzania. . .	63
3.2	First order Sobol indices for respective model parameters: Namawala Tanzania.	64
5.1	Relative scores of top ten submissions for the same environment when agent code is submitted for evaluation on an unseen environment (KDD Cup) and generated CSV files are uploaded by competitors for a live environment (Indaba Hackathon and NeurIPS)	102

Listings

4.1	OpenAI gym-like environment	84
4.2	Example Random Agent Submission	86
4.3	Submission file generation using the competition library	87
5.1	Sequential Bird Function for Test Environment	96

Glossary

action a single or combination of interventions performed at a particular time during simulation, which may lead to a state transition in the sequential decision making task.

agent observes and acts in an environment to achieve specific goals/targets.

Bandit a state-less one-step decision making agent.

control optimise the future by finding the best policy.

environment represented by simulations from model(s), may be either fully or partially observable.

epidemiology is the study (scientific, systematic, and data-driven) of the distribution (frequency, pattern) and determinants (causes, risk factors) of health-related states and events (not just diseases) in specified populations (neighbourhood, school, city, state, country, global) [1].

episode consists of all the states observed by a learning agent from initial to terminal state.

experiment a group of jobs with common properties for example with same geographic location or user.

Gaussian process a Gaussian process is a collection of random variables, any finite number of which have a joint Gaussian distribution [156].

intervention a single actuation or change to impact the observed future of the problem domain.

intervention plan a sequence of interventions to be implemented.

job a single model simulation.

model a single or combination of mathematical abstractions to describe a complex physical system.

planning policy improvement based on agent interaction with a model of the environment.

policy this may be a fixed deterministic action based on a defined intervention plan, or stochastic and learnt from observation. Ultimately the goal of Reinforcement learning is to learn a reactive policy which is a function of a Markov observation.

prediction evaluating the future given a policy.

regret the difference between the maximum achievable reward and that actually obtained during a learning episode.

reward a scalar signal used to reinforce particular decisions or behaviour. The magnitude and sign of rewards influences an agent's actions as it attempts to maximise cumulative reward.

simulation the result of a particular computational instantiation of a model run.

state an agent interpretable observation from an epidemiological model.

surrogate a data-driven model approximation, learnt from epidemiological model samples.

uncertainty quantification determining unknown model output distributions, often characterised by the first and second central moments of the distribution.

Acronyms

ACL Access Control Logic.

ACM Association for Computing Machinery.

AI Artificial Intelligence.

API Application Programming Interface.

CDF cumulative density function.

CPDA Cost Per DALY Averted.

CSV Comma Separated Value.

DALY Disability Adjusted Life Year.

DLT Distributed Ledger Technology.

EI Expected Improvement.

EIR Entomological Inoculation Rate.

GMEP Global Malaria Eradication Program.

GP Gaussian Process.

GPR Gaussian Process Regression.

HLF Hyperledger Fabric.

HPC High Performance Compute.

IRS Indoor Residual Spraying.

ITN Insecticide Treated Net.

MAB Multi-Armed Bandit.

MC Monte Carlo.

MCMC Markov Chain Monte Carlo.

MCTS Monte Carlo Tree Search.

MDP Markov Decision Process.

ML Machine learning.

MLE Maximum Likelihood Estimation.

MMC Malaria Modelling Consortium.

MSP Member Service Providers.

NICE National Institute of Clinical Excellence.

NLP Natural Language Processing.

NPI Non-Pharmaceutical Intervention.

NRMSE Normalised Root Mean Square Error.

PDF probability density function.

POMDP Partially Observable Markov Decision Process.

RL Reinforcement learning.

SA Sensitivity Analysis.

TD Temporal Difference.

UCB Upper Confidence Bound.

UQ Uncertainty quantification.

WVE Workflow & Validation Engine.

YLD years of life with disability.

YLL years of life lost.

Chapter 1

Background

The effort invested in ‘getting it right’ should be commensurate with the importance of the decision

Daniel Kahneman, Nobel Laureate
2002

The success of computational machines for tasks without exhaustive programming, through ‘learning’ from data, observations or examples, comprises the field of Machine learning (ML).

Often a Machine learning task involves a prediction or decision, based on some previous experience and/or new observations, and any particular computational machine or system may consist of many ML based elements.

For example to provide an over-simplification for the system of a self-driving car, the autonomous vehicle and machine may perform object detection, recognition and planning. Identifying obstacles, classifying them as a potentially vulnerable pedestrian, and then modifying its inferred position, velocity and acceleration to minimize the risk the self-driving car poses to the pedestrian.

The ideas developed in this thesis were seeded from the Reinforcement learning (RL) branch of ML approaches. From the inception of game playing algorithms by Arthur Samuel in 1959 [173], the field of ML has continued to demonstrate performance on game-like abstractions of problems, with less evidence of RL being utilised in high impact real-world problem domains. The promise of learning optimal policies based on delayed rewards via reinforcement poses potential in guiding high impact decisions or strategy. For real world applications which may positively impact lives, this is a possible solution commensurate with the time to extend and expand ML

techniques to such problems. Hence the material of this thesis focuses on the application and extension of learning machines to the domain of epidemiology and the control of infectious disease transmission.

1.1 Motivation

Broadly an epidemiologist is interested in the whole spectrum of improving public health through quantitative understanding of health and disease conditions in human populations. Our focus is on the ability of ML approaches to impact decision making, which will be biased towards a narrower field of epidemiology, specifically communicable or infectious disease transmission, where mathematical models are developed and used to inform public health policy decision making. Temporal control of such mathematical epidemiological models will be derived from our ability to leverage ML approaches for decision making over time, in tandem with the mechanistic, physical or mathematical models of epidemiology, which imbue our current understanding of disease transmission. The promise of a ML-based approach is one of augmenting and enhancing the expert ability of the epidemiologist to provide evidence which supports a best-known or optimal decision for those ultimately responsible. This promise stems from the fact that we may be able to solve decisions at a level of complexity infeasible to individual, expert or even groups of humans. The space of possible interventions¹ and intervention plans² to address these problems are often vast and complex, making it daunting for decision makers to explore, process and reason. Through creating a pipeline for directly putting data and multiple epidemiological models to work through ML approaches, we can scale the process of the epidemiologist, generating increased value from their understanding of the domain.

Often targets in the reduction of disease transmission are stated as goals or even a trajectory, as seen in Figure 1.1, which includes possible future projections of yearly cases in malaria endemic regions. The three scenario projections indicate possible trajectories which would either meet or fall short of the 2030 target. Additionally by looking at the control task with a finer temporal resolution, it may be necessary not just to target an X% reduction in the number of disease cases, but in fact limit the maximum number of cases or ‘flatten the curve’ as is demonstrated in Figure 1.2. This requires limiting daily cases below the healthcare system capacity through

¹actuators that may be implemented to impact the observed future of the studied disease.

²sequences of interventions to be implemented.

protective measures, consequently reduction in the total number of new cases is not the primary objective or target.



Figure 1.1: Yearly new cases of malaria per 1,000 people [129] in malaria endemic regions.

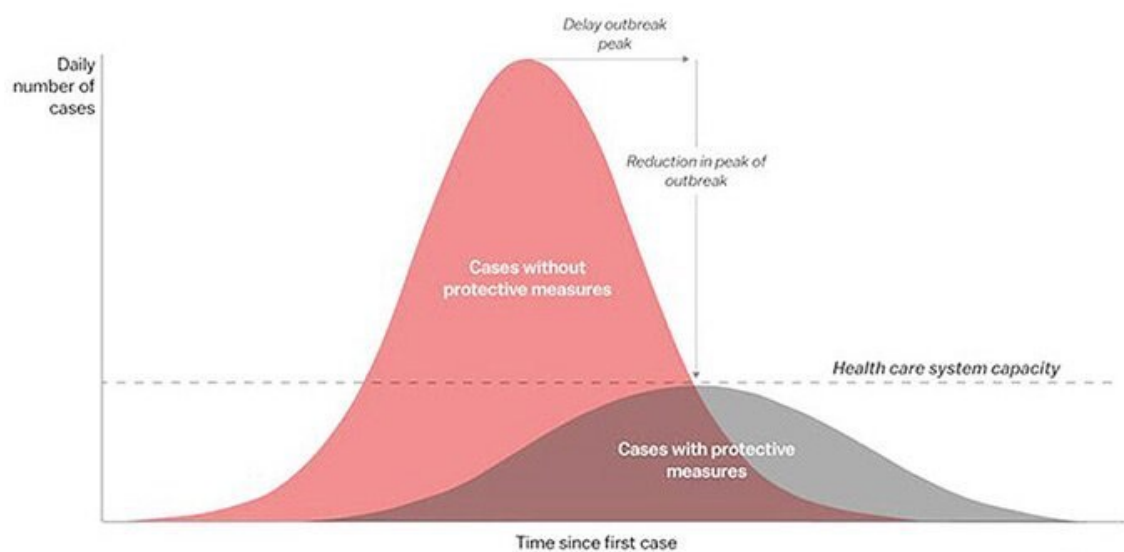


Figure 1.2: Flattening the Curve, health-system preparedness for viral pandemics [76].

Disease transmission may be controlled through a variety of public health interventions, such interventions may be pharmaceutical or non-pharmaceutical based on whether they derive from active chemical compounds (including vaccination) or not. These two categories may also be described by corresponding pathways that the interventions disrupt transmission. Pharmaceutical interventions aim to treat or inoculate against the disease and so effectively reduce the reservoir of infected, infectious and susceptible humans in the population. Non-pharmaceutical interventions provide control by reducing the probability of disease transmission between humans or the pathogen with vectors (such as the mosquito when considering malaria).

The coordination of interventions to limit transmission across geographies on a continental or global scale will not be handled with the ML approaches discussed in this work, though spatial interactions for disease transmission are implicit within some of the epidemiological models used, methods for extensive coordinated geospatial control remain a challenging extension. We will ourselves focus on decision making for a single location, whether this is an administrative level 0 (country), level 1 (province) or level 2 (district) analysis. These country centric administrative levels often match what is currently practical for national infectious disease programs, though cumulative benefits may be afforded through coordination, cooperation and communication between countries and continents, this remains a complex geopolitical problem.

Additionally the possibility for heterogeneity in the impact of interventions on individuals as opposed to an entire population will not be considered in this work. Specifically one may be interested in which sub-population or demographic may be vulnerable, or could greatest benefit from targeted interventions down to an individual level. Planning targeted interventions based on population demographics or cohorts is an important additional dimension to the decision making process, and a worthwhile analysis which may be extended with the epidemiological model based ML approaches to be presented.

At the time of writing the importance of our ability to plan in a time-sensitive manner has unfortunately been made drastically apparent. On 11th March 2020 the WHO situation report announced the novel Covid-19 virus global *pandemic*. When this was declared the number of confirmed cases surpassed 118,000 with over 4,000 deaths world-wide, and the pandemic continued to have orders of magnitude further impacts on the global community. Individual, group and global experiences exposed the result of (potentially) not taking immediate action to implement intervention plans to limit transmission through containment and control. Figure 1.3c gives a snapshot of the number of deaths from Covid-19 continuing to proliferate despite

global attempts at control. These numbers are stark, while the other examples of Figure 1.3 serve to give some contrast and a contemporary perspective of global deaths from infectious diseases. All these diseases have significant global epidemiological modelling efforts behind them, supported through programs such as the Gates Foundation³ and Global Fund⁴, along with local country led efforts.

Of these diseases depicted in Figure 1.3, malaria (Figure 1.3a) is perhaps the oldest disease, believed to have pre-dated human evolution [50]. In contrast to Covid-19 it has a disproportionate impact on the young (under five) population [131]. Recent years have seen a decline in cases but not in line with targets, with the major reported contribution to success in malaria control being the cost-effective intervention of Insecticide Treated Net (ITN) distribution [151].

Successes in the treatment of HIV and keeping the virus suppressed have led to reductions in disease deaths (Figure 1.3b) each year [175]. Notably these should be compared to the level of geographic disease deaths that have been witnessed as a result of the Covid-19 pandemic in 2020 as in sub-Saharan Africa there is evidence of up to a doubling of mortality risk for HIV patients who contracted Covid-19 [59].

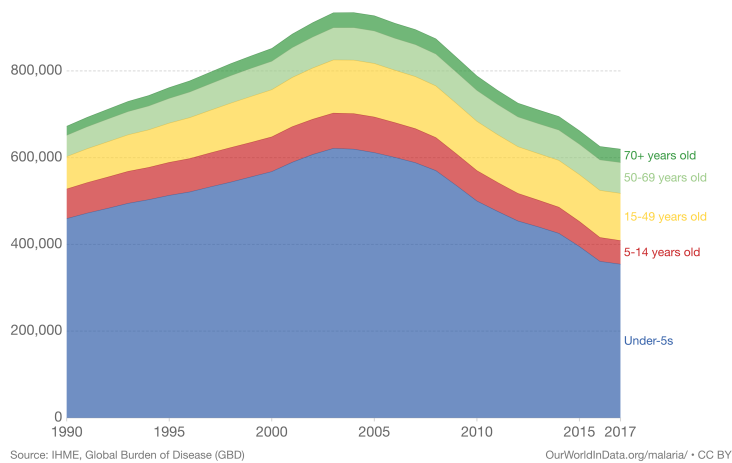
Finally the example of Polio (Figure 1.3d) serves to demonstrate where a vaccine has locally eliminated the disease while global eradication remains very close, though challenges due to asymptomatic cases ensure a non-zero prevalence in parts of the globe [114]. Complex dynamics and uneven access to vaccination mean that the disease has not been fully eradicated, which is a question we will continue to ponder for other infectious diseases into the future.

1.2 Mathematical Modelling in Epidemiology

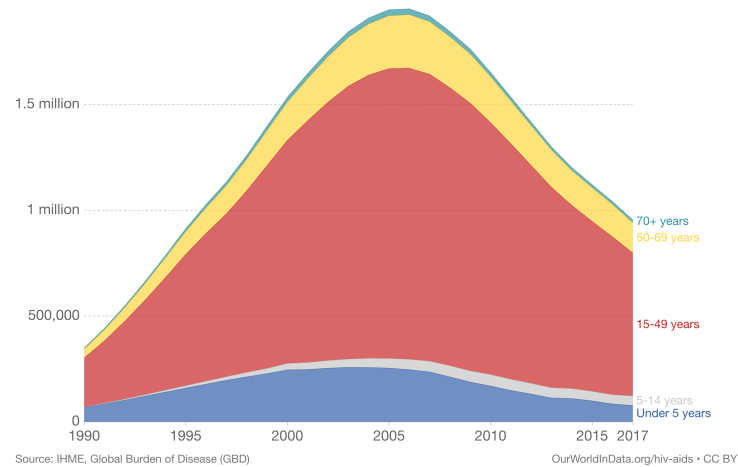
The first documented use of mathematical modelling in epidemiology was by Daniel Bernoulli in 1760, which aimed to motivate smallpox inoculation to increase population life expectancy [160]. Greater progress in quantitative approaches to understanding infectious disease transmission has continued in the 20th century, initially through the application of the laws of mass action to model malaria transmission by Ronald Ross [167], onwards to the introduction of compartmental models for the study of epidemics by Kermack and McKendrick in 1927 [103]. The extended Ross-Macdonald malaria model further developed by George Macdonald was used in the guidance of the Global Malaria Eradication Program (GMEP) (1955-1969); the Ross-MacDonald

³<https://www.gatesfoundation.org/>

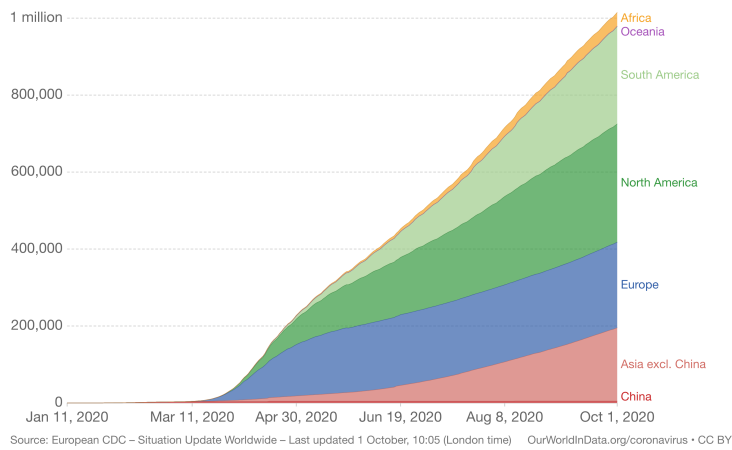
⁴<https://www.theglobalfund.org/>



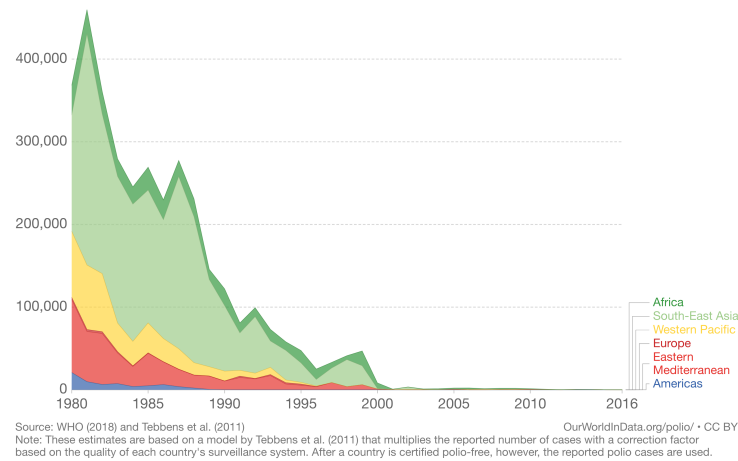
(a) Global Malaria Deaths by Age Group [164]



(b) Global HIV Deaths by Age Group [165]



(c) Global Covid-19 Deaths by Continent [122]



(d) Global Paralytic Polio Cases by Continent [138]

Figure 1.3: Four Examples of current global epidemics under various states of control.

malaria model making the conceptual leap to capture both mass action for transmission and the mosquito (vector) dynamics. Understandably subsequent investment and research effort in the domain of infectious disease modelling has led to increasing numbers of epidemiologists developing models, collecting data and proposing policies through possible intervention plans based on new quantitative understanding of epidemiological systems. Conversely if the partial success of the GMEP had been complete and repeated, it would have possibly resulted in widespread malaria eradication [142]. Humanity would have less focus on our control of communicable disease transmission, and as such modelling it. Unfortunately, the previous examples of Figure 1.3 remain persistent causes for great loss of human life.

The outputs of epidemiological models are important for providing evidence to develop feasible intervention plans, and ultimately recommend policies to achieve a target or set of targets. The model developer is often tasked with proposing improved intervention plans in the face of competing priorities to match the real-world problem. For example, trying to reduce programme costs while improving health outcomes, where saving lives is necessarily a costly endeavour. This requires the ability to efficiently sample model inputs to simulate the outcomes of different interventions given a set of constraints. Running highly descriptive mathematical models is computationally expensive and the search for improved intervention plans will have further complexity tied to the number of chosen interventions.

Recommended intervention plans generated by one or more modellers are shared with the relevant decision makers for potential adoption and implementation. Decision makers are faced with the daunting challenge of making decisions based on existing models, without being technical experts and without having access to all evidence about all possible options. In many cases data and models are not shared in a reusable, repeatable and reproducible manner, making it difficult for decision makers to compare output from different models. While in cases where results are presented from multiple transmission models the recommended intervention plans can conflict [212], leaving no clear resolution for making a decisive action. Furthermore, with multiple stakeholders, coordinating the development of effective intervention plans becomes a significant operational challenge. Finally the task of decision making under uncertainty, where outcomes are not completely specified (or known), further complicates the intervention planning process.

To conclude we would like to point to the heterogeneity in epidemiological modelling approaches which have been developed. Figure 1.4 provides model schematics for four separate modelling efforts aiming to provide decision making tools for malaria

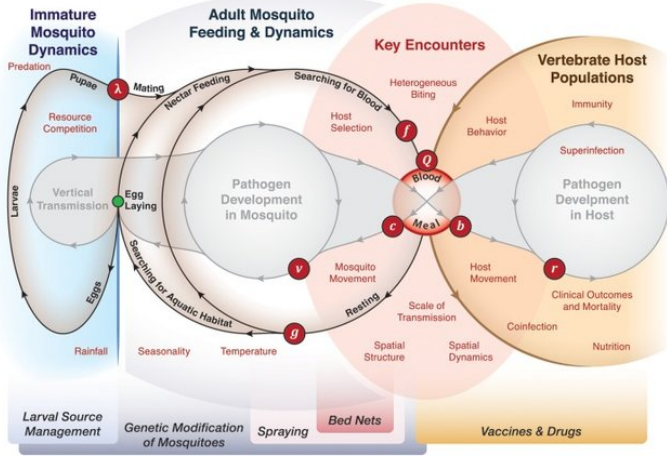
control. Our perspective is that heterogeneity is beneficial; again there is no one conclusive model design and comparative, competitive and parallel research efforts help to progress disease understanding along with optimal intervention plans for locations. For example malaria modelling efforts are coordinated under the Malaria Modelling Consortium (MMC) for the collaborative exchange and joint studies into the efficacy of intervention plans [42].

1.3 Uncertainty Quantification

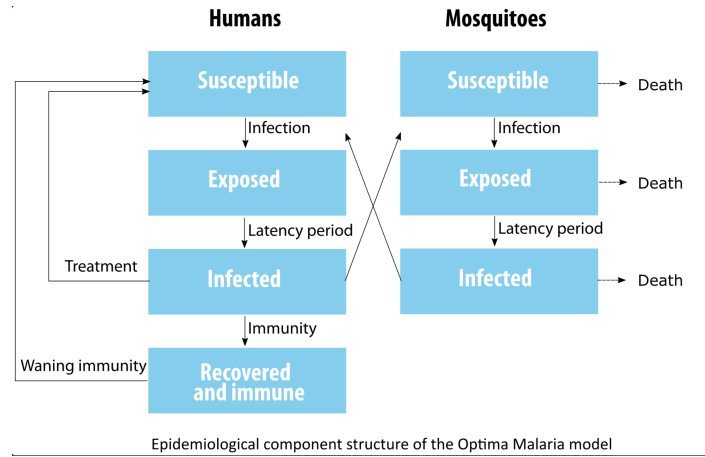
The major forms of mathematical models for dynamical systems may be classified into three groups. These three groups are directly applicable to the domain of epidemiological mathematical modelling:

- **Black-box** models are entirely data-driven models and their structure is not guided by any underlying physical mechanism. Such model types are common in Machine learning or often considered as statistical models; in either case we'll take the perspective that model parameters are learnt from observation.
- **White-box** models are considered to be mechanistic and attempt to describe the underlying physical mechanisms. As an example of **white-box** models, agent-based models are constructed based on interactions between multiple agents of the epidemiological system and aim to create a closer resemblance to real-world dynamics.
- **Grey-box** models consist of mixtures of phenomenological (black-box) and mechanistic (white-box) model components.

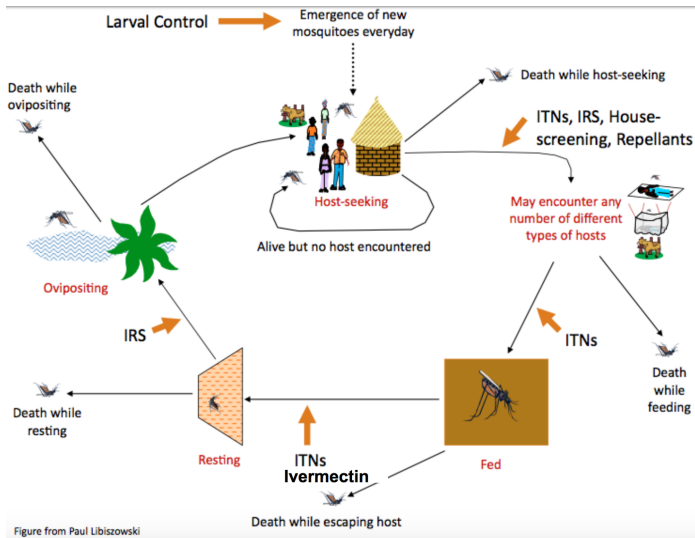
For the purposes of this work, all models will agnostically be considered as black-boxes. The epidemiological models are therefore effectively a set of exposed parameters, which allows us to switch seamlessly between the use of different model types and between diseases. This ground-up approach of extensibility and scale is perhaps rooted in the premise that much of Machine learning considers models as black-boxes across agnostic applications. In the case of epidemiology, challenges in reproducibility, and lack of sharing or a set of standard benchmarks, limit the amount of work which effectively compares the results of different modelling approaches. We will not therefore spend significant time focusing on the analysis of epidemiological model structures, instead we will look at mappings between inputs and outputs from various



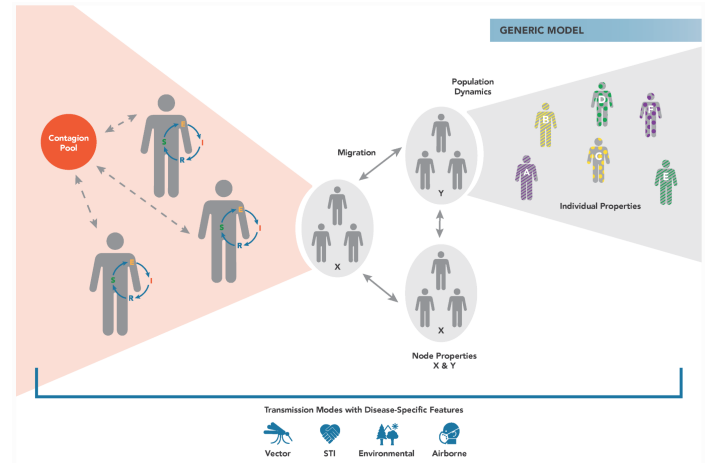
(a) MASH extended Ross-Macdonald model [182]



(b) Optima compartmental malaria model [176]



(c) OpenMalaria deterministic vector model [108]



(d) EMOD agent based model [68]

Figure 1.4: Model schematics for a single disease; malaria, with varying model approaches and types.

models for forecasts or predictions under uncertainty, ultimately to be exploited for the task of decision making.

While the internal mechanisms of the models used will remain as black-boxes, different approaches will return deterministic or stochastic results. This is of note with regards to how uncertainty may be handled from the model. A deterministic epidemiological model will principally take the compartmental form of Kermack and McKendrick [103]. Such models are deterministic in the sense that, once we know the population numbers of each compartment (initial conditions) and the rate constants between the compartments, the system is fully determined and has a single outcome or prediction, while for any real world scenario, reducing the problem in such a way is unlikely to be realistic, or have strong predictive performance. We may inject some randomness into deterministic models through quantifying uncertainty in the model parameters or initial conditions. Deterministic compartmental models tend to be suited for large well mixed populations, with relatively small numbers of recovered individuals. Typically a deterministic approach may be evaluated much faster, as differential equations have efficient solvers with a single outcome being computed. Stochastic simulations may need many more computational cycles in order to model the atomic transitions of a stochastic agent-based model in comparison.

Taking a common footing for both modelling approaches through uncertainty quantifications can lead us to the following classifications. Uncertainty in predictions arising from the uncertainty in the model parameters is called *epistemic* uncertainty. This could lead to a distribution of outputs from a deterministic model if parameter distributions were defined on the model parameters, or based on a stochastic models own description of epistemic uncertainty. This kind of uncertainty may be better quantified with more data or observations. Consequently, epistemic uncertainty should be higher in regions of no or little training data and lower in regions of more training data. Uncertainty arising from the inherent noise (or imprecision) in the disease dynamics is an example of *aleatoric* uncertainty. It cannot be reduced if we get more observations, in principal we have to accept that the future is uncertain.

The utility of uncertainty quantification becomes apparent when we wish to make comparable decisions across various scenarios. Often in such risk adverse or high impact scenarios we may want to minimise uncertain outcomes, taking ‘safe’ decisions in preference to potential greater improvements with greater uncertainty or risk.

1.4 Model-Based Decision Making

As a specific disease example, malaria presents one of the most complex domains for intervention planning in global health. The disease modelling and simulation efforts available for use in malaria control and eradication are among the most highly developed for any disease. Researchers developing recommendations for intervention plans effective in the fight against malaria may have access to several domain models developed by organisations across the globe [15,78,183]. They rely on the recommendations provided by their own models, and do not access the insights or the potential validations possible from comparisons with models developed by others.

There are legitimate concerns in the presentation of quantitative results based on models, and doing so with the appropriate uncertainty bounds, limitations, and caveats. It is of extremely high value for decision makers to have this type of information, systematically identifying where there are unknowns, and when there are multiple available options, with the goal of confidently generating decisions backed by multiple evidence producers. Finally, the generation of insights based on models, whether a single model or an ensemble analysis, is a continual research challenge. The intellectual process alone may be daunting, before considering the computational complexity in generating resource intensive results.

Modelling consortia have previously conducted comparison exercises of respected models in their field when applied to the same questions in the same countries. The Malaria Modelling Consortium (MMC) has conducted such model comparisons to assess whether there was consensus in messaging on predictions of the impact and cost-effectiveness of the RTS,S malaria vaccine [202] and the role of mass drug administration on elimination [143]. In this case the different models predicted differing magnitudes of the effects of mass drug administration, but consensus answers were reached for several factors contributing to the effectiveness of mass drug administration. The models also reached consensus about RTS,S administration and priority implementation areas. This is despite fundamental differences in the types of the model developed, and differences in datasets used for model calibration to the studied location. More generally malaria intervention plans include a mix of intervention types such as the distribution of the Insecticide Treated Net (ITN), process of Indoor Residual Spraying (IRS), vector larviciding of bodies of water, and malaria vaccinations. The space of possible interventions, combinations of interventions, and combinations of sequences of interventions is daunting and inefficient for human decision makers to explore without adequate decision support tools and coordination.

1.4.1 Malaria Elimination and Control Background

Firstly, malaria as a broad term describes the disease associated with infection from the *Plasmodium* parasite. Five different *Plasmodium* species may infect humans with complex lifecycles and vector dynamics. The evolution of these *Plasmodium* species is believed to have preceeded our own evolution as man [150]. Malaria may therefore be responsible for the greatest number of cumulative human deaths over our existence. With several key milestones in human history, the serendipitous discovery of quinine as a treatment in the 1600s was the first successful use of a chemical compound to treat an infectious disease [3]. As knowledge grew into the causes of infection, initially with research into bacteria, the *Plasmodium* parasite and its vector the mosquito were uncovered in the late 1800s [57]. With this newly discovered knowledge about the transmission cycle of malaria along with rapid advancements through industrialisation and economic growth, the US was able to eliminate malaria in the US in the 1950s [9].

We have therefore made significant successes in accumulating knowledge towards the isolated elimination of malaria, to the point that it would appear a solved problem to replicate. However a ‘broad-brush’ approach in applying this knowledge to other nations was shown to be flawed in the Global Malaria Eradication Program (GMEP) from 1955-1969. This effort made apparent that a single strategy can not be applicable everywhere and that a long-term commitment with a flexible strategy that includes community involvement, integration with health systems, and the development of agile surveillance systems is needed [132]. Substantial gains were made through the GMEP though these were lost in resurgences, sometimes of epidemic proportions. Now after renewed investment in global malaria eradication, malaria remains a significant global health problem, with over 500,000 deaths a year concentrated in sub-Saharan Africa [10, 178]. It is estimated that 200,000 people are suffering from malaria globally at any time. This is a major economic burden on the countries where malaria is still endemic, the poorest countries as a result suffering a 1.3% reduction in their economic growth.

As the economics of the problem prohibit its elimination, analyses have highlighted that the gap between the costs of elimination and of control [178] are hindering global eradication, as we are making sub-optimal decisions limited by cost. Low-cost interventions, in the form of bed nets and insecticide spraying, have become viable mechanisms to try to overcome the infrastructural issues with solving malaria. At present we have not achieved eradication through this ‘broad-brush’ approach of distribution alone.

Finally, those experiencing and living with malaria day-to-day may be disengaged from the interventions they receive. For example, if someone is required to use an uncomfortable bed net, when they are habituated to malaria symptoms. Or, if sleeping under a bed net would not be required with adequate local investment in sanitation and drainage, to reduce vector numbers. Furthermore, the distribution of bed nets has empirically been shown to be effective only to a certain amount [119]. Malaria is perhaps (socially) not the biggest problem faced by an individual in an endemic country, but this makes it very difficult to prevent its transmission, so populations must not be disengaged and interventions not disconnected from the goals of proposed solutions.

In the 2014-2020 Malaria Reduction Strategic plan, the Ugandan Ministry of Health's National Malaria Control Programme set the following three goals:

1. by 2020, reduce annual malaria deaths from the 2013 levels to near zero.
2. by 2020, reduce malaria morbidity to 30 per 1000 cases.
3. by 2020, reduce the malaria parasite prevalence to less than 7%.

In 2014, the prevalence was estimated at 19%, and by all estimates progress was made, yet the 2020 targets were not achieved. The current estimation is 9% prevalence nationwide (with significant local variation). With the advent of Machine learning based approaches, can recommendations for intervention plans which aim to reduce the prevalence from 9% to 5% over the course of five years countrywide, be developed? The primary components of the intervention programs are the Insecticide Treated Net (ITN) and Indoor Residual Spraying (IRS), though there is significant consideration to improvements in case management, malaria in pregnancy, and urban malaria control [204].

1.4.2 Potential of Machine Learning

Machine learning methods are well-suited to efficiently exploring the space of all possible combinations of interventions. Epidemiological model simulations provide the opportunity to play forward different interventions and/or plans to predict the future state of the world. They do not require any physical investment or action, and observations of expected outcomes may be made repeatedly in-silico. This may be thought of as analogous to experimentation, though here we consider it a form of online sequential learning, naïvely the ability to make observations from the simulation and

update model knowledge. Any decision making agent may formalise this process of online learning via reinforcement. A reinforcement, or reward signal may thus be used to learn policies based on observations of the impact of actions on that environment. This environment may consist of simulations or epidemiological models, or an agent interacting directly with the ‘real-world’. Conventional benchmarks and examples of Reinforcement learning (RL) approaches in research [67] have avoided complex physical simulation models. The approach developed in this work given the emerging availability of epidemiological models offers an opportunity to greatly enhance the decision making process via model-based evidence.

1.5 Contributions of Thesis and Outline

Throughout this thesis the use, extension and deployment of Machine learning approaches to high impact decision making, with regards to disease control interventions, will be outlined. From developing Bandit learning approaches as one-step control methods (Chapter 2); supervised Machine learning techniques for epidemiological model predictions and Uncertainty quantification (Chapter 3); introducing an infrastructure to run epidemiological model simulations at scale (Chapter 4); a competition based set-up for heterogeneous Reinforcement learning approaches to solving the epidemiological control problem along with methods of combining information from different Machine learning and epidemiological approaches (Chapter 5); and a full development of the application of sequential decision making via Reinforcement learning methods for the purposes of control and planning from epidemiological models (Chapter 6).

Chapter 2

Exploring One-Step Prediction and Control with Online Learning

We have no idea about the ‘real’ nature of things. The function of modelling is to arrive at descriptions which are useful.

Richard Bandler and John Grinder,
1979

Having already discussed the necessity, origin and use of epidemiological models for decision making we’ll now use this chapter to lay the framework to extend the use of such models towards learning from simulation. A model is a physical, mathematical, or logical representation of a system entity, phenomenon, or process. A simulation is the implementation of a model over time. A simulation brings a model to life and shows how a particular object or phenomenon will behave. It is useful for testing, analysis or training where real-world systems or concepts can be represented by a model [149]. With such a description of the real-world we may now look to train computational systems to learn from simulation.

2.1 Online Learning

Online controlled experiments may make use of simulation to set up experimentation *in silico*. The theory of controlled experiments dates back to Sir Ronald A. Fishers’ experiments in the 1920s, and is formalised in his text the *Design of Experiments* [73]. Online controlled experiments started to be used in the late 1990s with the adoption of our global system of connected computation, the Internet. Today no longer just websites, but major technology companies including Amazon, Facebook

and Google to name some of the most immediately prominent, run thousands to tens if not hundreds of thousands of experiments daily, each testing user interface (UI) changes, enhancements to algorithms (search, ads, personalisation, recommendation, etc.), changes to apps, content management systems and other areas.

Extensive scientific research, effort and time may go into creating an exquisitely complex and rich computational model of disease transmission. This will likely be used in the paradigm of *A/B testing* by an epidemiologist (also likely the model creator). The term *A/B testing* hopes to capture the process by which experimentation is performed through a direct comparison, or hypothesis test, between two outcomes *A* or *B*.

In our case we will compare simulated evidence to inform a decision or policy, in contrast to other applications which may look to guide design, healthcare or marketing decisions [172]. It may be sufficient in these applications to present that option *B* outperforms option *A* and this can be argued to provide a comparative basis as evidence for a decision. Although this may be reaching too hasty a conclusion, in the sense that any such decision may only be considered optimal (iff one of the two options being considered is the optimal decision), though we may demonstrate that the *A/B* choice offers a simulated improvement. From this simple two case scenario we are bound to making a decision under limited information, bias or even misinformation whether intended or not. Additionally, without formally stating uncertainty bounds there may be scenarios where the chosen decision under-performs the expected outcome. For the case of making a model-based decision in epidemiology, a sensible proposition for an *A* policy would be the current disease control interventions in place with a *B* proposition made relative to this current policy (*A*). Therefore an *A/B test* would assess whether there is sufficient model-based evidence to motivate a change in policy to the proposition *B*. If we understand sufficiently well the current state and control interventions for the disease, a proposition of a policy for which there is quantitative evidence to suggest improvement is a significant step. Especially, if the monetary resource is available to promote such a decision, model support in an *A/B* fashion represents the baseline of how most implemented intervention plans are supported by quantitative evidence [79].

Although there is more that can be done, it is proposed that reaching specific policy goals, such as elimination in defined contexts, requires increasingly non-generic advice from modelling. For example research in epidemiological model development looks to motivate the importance of measuring baseline epidemiology, intervention coverage, vector ecology and program operational constraints in predicting expected

outcomes for different combinations of interventions [41]. This does not deviate from the baseline process, that all of these goals may be studied as a series of A/B tests without accelerating towards critical solutions, which are needed to solve immediate problems in a timely but principled manner. The rest of this Chapter will begin the work of how we may improve our exploration of novel control policies from published models.

2.2 Stochastic Multi-Armed Bandit

In formalising an extension to A/B testing we'll move towards associative, or one-step Reinforcement learning in the form of the Multi-Armed Bandit (MAB). For direct applicability we'll consider a form of the Multi-Armed Bandit under a limited computational budget, specifically a discrete number of epidemiological model runs (n), which may be known or unknown, though each model run is considered to be computationally expensive.

To provide background to the approach of the Multi-Armed Bandit (it is named after the canonical example of the problem), we could interpret this class of problems as analogous to making selections of one-armed bandits (slot-machines) given a fixed budget of coins or tokens, either known or unknown and the objective being to maximise winnings given unknown stochastic distributions of payouts. Optimal approaches should thus balance exploration and exploitation. Where exploratory actions serve to improve estimates of stochastic rewards, exploitative actions should maximise cumulative reward given the current information and 'model' of the world.

We will consider the MAB problem as a tuple of $\langle \mathcal{A}, \mathcal{R} \rangle$ consisting of K arms or actions a labelled as integers $\{1, 2, \dots, K\}$. At each round $t = 1, 2, \dots, n$ an arm is selected:

$$a_t \in \mathcal{A} \in \{1, 2, \dots, K\}$$

The reward received by an action is a random sample drawn from the unknown reward distributions of the arms $\mathcal{R}^a(r) = \mathbb{P}[r|a]$. Having selected an action a_t at round t the environment generates a reward $r_t \sim \mathcal{R}^{a_t}$ which is assumed to be independent of previous actions or rewards. The goal of any player or algorithm in this setting should be to maximise cumulative stationary rewards across the model runs, where cumulative reward (R_t) is simply the arithmetic sum of rewards across rounds:

$$R_t = \sum_{\tau=1}^t r_{\tau} \quad (2.1)$$

Up until the computational budget of the episode whereby $t = n$. The *action-value* is the mean reward from action a and taken from the reward distribution of the arm, therefore if we drop the dependence on round (t) due to stationarity of rewards, we may state the following:

$$Q(a) = \mathbb{E}[r|a] \quad (2.2)$$

In addition the *optimal value* V^* , for the optimal arm or action a^* is:

$$V^* = Q(a^*) = \max_{a \in \mathcal{A}} Q(a) \quad (2.3)$$

If any player, algorithm or agent had access to these unknown distributions *a priori* they would be able to achieve a maximised expected cumulative reward of:

$$nV^* = nQ(a^*) = \sum_{\tau=1}^n \mathbb{E}[r_{\tau}|a^*] \quad (2.4)$$

2.2.1 Regret

To reiterate that for our framing of the MAB problem we assume a fixed finite computational budget for a number of model runs (n), we will define regret as the difference between the maximum achievable reward and that actually obtained during the learning episode consequently of length n . At any instant t the opportunity loss l_t for that step is:

$$l_t = \mathbb{E}[V^* - Q(a_t)] \quad (2.5)$$

and the total regret (L_t) up until step t may be defined as:

$$L_t = \mathbb{E} \left[\sum_{\tau=1}^t (V^* - Q(a_{\tau})) \right] \quad (2.6)$$

At this stage we should note that if we seek solutions to maximise expected cumulative reward this is equivalent to minimising expected cumulative regret. While a *gap* of Δ_a exists for each action (a) between the *optimal value* V^* and the *action-value* $Q(a)$:

$$\Delta_a = V^* - Q(a) \quad (2.7)$$

Now we shall make use of the *count* $N_t(a)$ to track the number of selections of action a by the MAB approach up until round t , such that we may extend total regret as defined in Equation 2.6 using *gaps* and *counts*:

$$\begin{aligned} L_t &= \sum_{a \in \mathcal{A}} \mathbb{E}[N_t(a)] (V^* - Q(a)) \\ &= \sum_{a \in \mathcal{A}} \mathbb{E}[N_t(a)] \Delta_a \end{aligned} \tag{2.8}$$

Good solutions to the bandit problem will have a low cumulative regret, ensuring small counts for large gaps, though this is tricky as with the problem of maximising cumulative reward, reward distributions and hence gaps are unknown. The relative performance of solutions to Bandit problems are often analysed through bounding of cumulative regret. For example an algorithm which takes only exploratory actions across an episode (n), will have linear regret, conversely if an algorithm never explores it will also have linear total regret [48]. Achieving sub linear total regret is therefore a key measure of useful performance for approaches. The concept is included here for an understanding of the theoretical objectives.

2.2.2 Infinitely-Many Armed Bandits

In addition to the notions of exploration and exploitation in the Multi-Armed Bandit, if we are to consider the problem where the number of possible actions (K) exceeds our number of model runs, computational budget or episode length (n) and tends towards infinity, we have to add the concept of discovery. As in such a problem type it will not be possible for any solution to sample all possible actions, discovery infers the finite set of actions which are sampled, allowing exploration/exploitation mechanisms to proceed on this finite set. Berry [34] first addressed the bandit problem with infinitely many arms (assuming Bernoulli or Discrete reward distributions of actions). There may be instances, where there is no relationship between the distribution of rewards, in which case a random search would be a suitable method for such subset discovery.

For the specific problems considered in this work it is reasonable to consider that all cases represent infinitely many possible actions, where there is a structural relationship between their reward distributions, due to the underlying epidemiological models encoding some unknown causal relationship between actions and rewards. Any reduction in the numbers of actions is likely to inflate the aleatoric uncertainty which the epidemiological model simulates, though this is not guaranteed. However, it does push forwards capturing as many actions as possible.

furthering the decision of capturing infinitely many possible actions. Finally expressing that the process of discovery in simulating new actions further adds to the infinite possible actions.

Therefore balancing, exploration, exploitation and discovery is a consideration provided by the MAB framework. In addition to slot-machine selection we may consider another example to further describe this balancing act. These mechanisms have analogies to decisions in social activities; one may choose to spend time with their closest connections (exploitation); someone with a degree of 2 separation in their social network (exploration); or finally a group or person considered to have a completely different social network to their own (discovery).

2.2.3 Optimism in the Face of Uncertainty

In order to achieve sub-linear regret, techniques to provide solutions to the Multi-Armed Bandit problem tend to focus on the notion of *optimism in the face of uncertainty*. This concept formalises the process for constructing sequences of decisions, to minimise regret and in so doing approach a^* given uncountably many, unknown distributions, of continuous rewards. This notion is guided by the idea that the higher the uncertainty in an action-value $Q(a)$ the more important it is to explore, and optimistically it could turn out to be the best action! Instantiations of algorithms which encourage exploration use various heuristics: ϵ -greedy [194], Leader [91] or Upper-Confidence Bounds [18] based on empirical arm evaluations, all of which achieve sub-linear, logarithmic regrets. While all are valid approaches, we note that empirical evaluation of arms does not lead to the most sample efficient mechanism for large problems. The problems tackled in this work will take the Infinitely-many armed Bandit form, and as such it is necessary to leverage structure in the problem.

2.3 Running Epidemiological Simulations as Bandits

Finding an optimal policy from running simulations can be posed as a stochastic Multi-Armed Bandit problem. For example, this approach has been used to develop algorithms aiding the design of clinical trials [209], where actions should be made to balance exploitation (positive patient outcomes) and exploration (searching for actions which may lead to a clinical ‘breakthrough’). In this chapter we ground this approach in the determination of the execution of simulations to infer high-performing actions as fixed or deterministic policies for simulated populations.

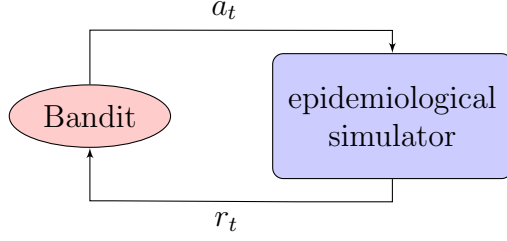


Figure 2.1: Action a_t selected by a Bandit algorithm receives feedback or reinforcement via reward r_t based on the epidemiological simulators outputs

2.3.1 Time

A clear distinction of simulation time (t_s) and algorithmic time (t) should be made. In the MAB scenario the notion of time is purely algorithmic and representative of the number of arm pulls or rounds that have been taken. This approach is suitable for taking a one-step, or static decision, from simulated learning, requiring that any change in simulation input is decided before the simulation is performed. For example, a simulation could be run over a 5 year simulated intervention period on a daily time-step resolution. A Bandit approach to the selection of an optimal intervention plan would select from predefined intervention plans, posed as an action a_t to be simulated, receiving a scalar reward based on the epidemiological simulator output r_t , as described in Figure 2.1. The action selection task is not informed by any observation of the simulation state and is reduced to simulating uncountably many intervention plans, which provide stochastic rewards based on the simulators final output. The MAB problem being a single state framing or single epidemiological model online learning problem, with no observed state transitions in this case.

Some epidemiological simulators directly lend themselves to this approach, as they have not been designed to provide observable intermediate simulation values, only a final output result. Through our common generalisation of maintaining the simulator as a black-box, this would follow as the most suitable approach, not requiring any change to the model's internal mechanisms. While simulation time (t_s) may often be on the order of days, the time frame for decision-making could range from years to weeks. Traditionally decisions are made over longer time frames but increasingly there is pressure for this time horizon to be shortened, specifically in response to global pandemic preparedness.

2.3.2 State

In the MAB framing of the problem there is no state transition between simulations, this could be reduced to the notion that each simulation upon completion returns to its initial state. We are trying to solve the problem of making one-step control policy recommendations for the simulation intervention period. To consider the single state of the epidemiological model set-up, this may be defined by the model's parametrisation θ_m (which may be uncertain), ultimately defining a model hyper-state, as this is not an observation made directly from the black-box simulator. Each action provides no observation of a physical state of the system, while it does provide information with regards to high performing policies given the model's hyper-state or parametrisation. For example in Figure 2.2 successive actions (a_{1-5}), may lead to improved inference of a continuous reward distribution, and the transition from $\mathcal{S}_t \rightarrow \mathcal{S}_{t+1}$ defines an improved estimation given a stationary model hyper-state $\mathcal{S}(\theta_m)$.

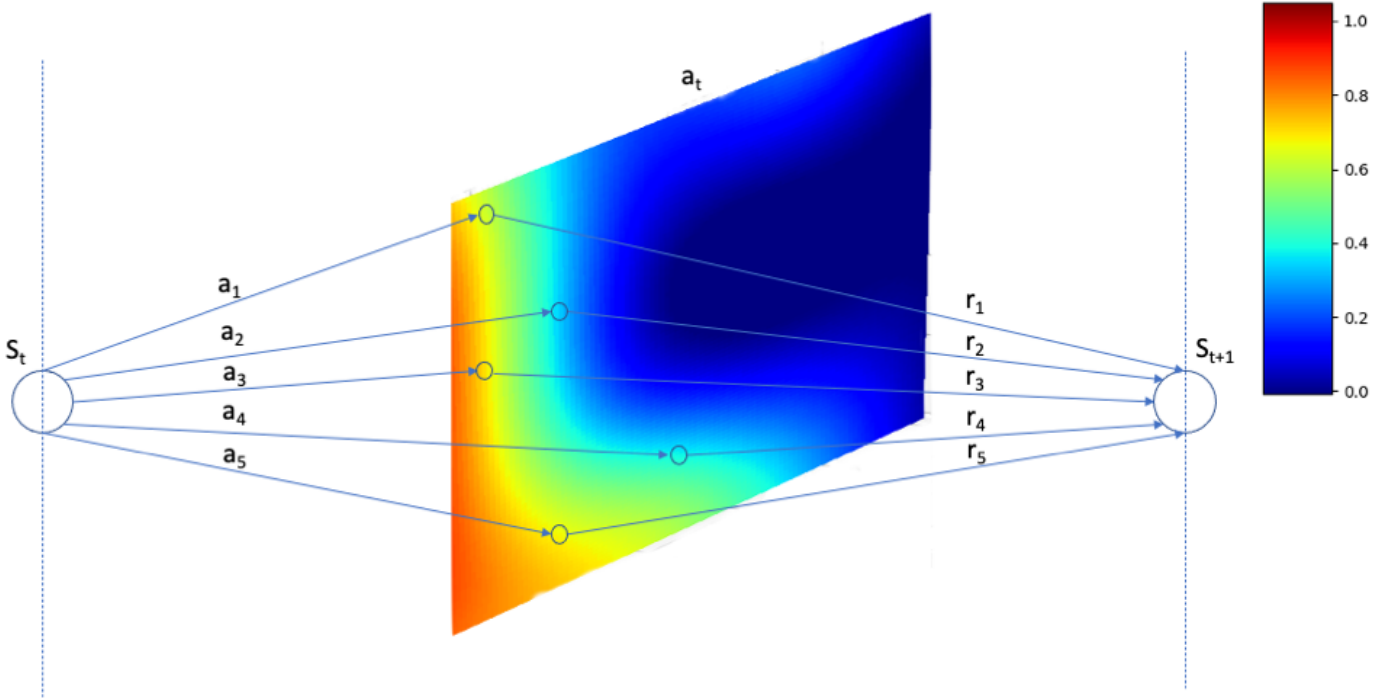


Figure 2.2: A batch of arm pulls for one-step decision making, the state transitions $\mathcal{S}_t \rightarrow \mathcal{S}_{t+1}$ improve estimates of the underlying continuous action value function $Q(a)$, the underlying techniques for inference are discussed further in Chapter 3.

2.3.3 Action

For the purpose of decision making from epidemiological models we'll define an action (a) as an abstraction based on possible epidemiological model control inputs, the most direct translation of this for the purpose of actionable decision making being a modelled set of interventions. The action-space ($\mathcal{A}$) may be composed of many interventions and therefore be high dimensional. for example efforts to categorise the number of Non-Pharmaceutical Interventions deployed to control the transmission of Covid-19, such as the WNTRAC database, currently contains 6000 non-pharmaceutical interventions across 261 countries and territories [190]. While in this case multi-dimensional inputs may be translated through a functional mapping into variations of a single model parameter, such as the infection rate (β) of a compartmental model. Hence often abstractions are required to map a translated intervention to a model change. Here we will not consider specific functional mappings of interventions into model changes, which will be taken as predefined by the epidemiological model developer. Action based inputs to the epidemiological model will therefore, for generality, be considered as encoded in a purely black-box manner. A component of an action will contain a proposed intervention, its relative strength, coverage or impact on the simulated population ($[0 - 1]$) and timing in the simulation (t_s). Our action-value function based search, achieved through the MAB approach will aim to return the action (a^*) which maximises simulated reward.

2.3.4 Reward

The calculated reward distribution across actions ($\mathcal{R}^{a_t}$) will necessarily have to make use of available model outputs. It is assumed that in order to achieve optimal control of a disease, we aim to reduce the negative outcomes from the disease, where numbers of cases, deaths, complications etc. may have been modelled, depending on the specific design and original intended use of the model. Any such reward observed from each action $r_t \sim \mathcal{R}^{a_t}$ will be considered as stochastic, based on stochastic epidemiological model outputs, where mechanisms for stochasticity may come from mechanistic model design, utilising probabilistic state transitions or systems of differential equations, where model parameter uncertainty is propagated. We'll continue to consider models to have stochastic returns, as we aim to make decisions on uncertain futures and necessitate, that the simulations used provide estimations of uncertainty.

A reward signal could take the simplistic view of these raw stochastic model outputs, for example deaths or infected cases, while to mirror the process of making

public health decisions from models we'll use a readily available quantitative framework. Such a common metric across diseases and models was developed in the first global burden of disease report (1990) [118], which introduced the term Disability Adjusted Life Year (DALY). For the purposes of learning from simulated outcomes, this provides a meaningful single scalar value for the decision making process. When making a proposal for mixes of interventions over time, we may now look to maximise the health outcomes with regards to DALYs, while also regulate improved health outcomes with regards to cost.

2.3.4.1 DALYs

The Disability Adjusted Life Year (DALY) [130], is a measure defined by the total years of life lost (YLL) due to fatality linked with contraction of a disease, and the number of years of life with disability (YLD) as a result of the disease. Where possible the magnitude of any reward is determined through an economic cost-effectiveness analysis of stochastic epidemiological simulation output, which we'll base on the concept of DALYs. Upon completion simulators will return outputs either on an individual or human population level. A full example of performing this calculation from simulation outputs is covered in Section 2.4.1.

2.3.4.2 Costs

Simulated costs are included so that decisions may be made to prioritise the highest impact. As such we should look to promote the most cost-effective solutions on a total simulated population. Where DALYs provide a health outcome for a particular action, different combinations of interventions and coverages may have greatly varying costs. Again, for generality this will be considered monetary cost. For example administering an Influenza vaccine in the UK may have a base cost of £5.96. In addition to the base scenario there may be possible cost savings or further complications, e.g. avoiding a visit to the GP (General Practitioner), costing £31, in the case that a vaccine is administered and a symptomatic case experienced. Or perhaps a £31 cost for treating a side effect of administering the vaccine. Finally, a severe case of hospitalised influenza costs £1,029. All these values have been taken from [206], as an example of enumerated costs to be considered when evaluating cost-effectiveness. If possible, all such events should be priced comprehensively and estimated based on expected outcomes.

Other non-monetary units may be used to place a cost on interventions, but the concept remains the same. This could be taken from an optimal control perspective

of a linearised system with quadratic cost, being a function of state and control variables [100]. For example, an optimal and minimum cost solution to the Linear-Quadratic Regulator problem [27] looks to minimise the energy of Equation 2.9 based on control inputs ($u(t)$) and outputs as state observations ($x(t)$) over simulated time (t_s).

$$\frac{1}{2} \int_0^{t_s} [x^T(t)Qx(t) + u^T(t)Ru(t)] dt \quad (2.9)$$

Where we will take Q and R as stationary matrices, Equation 2.9 may be tuned for the task of decision making from epidemiological models, placing costs on model control inputs or actions ($u(t)$) and model outputs or states ($x(t)$), entries in R act as a direct penalty on inputs ($u(t)$), and entries in Q place costs on negative modelled outcomes such as infections or deaths.

2.3.4.3 Cost-Effectiveness

Finally, to complete a general reward calculation, any Bandit algorithm may receive rewards based on the cost effectiveness of an action, balancing both positive and negative outcomes. For this purpose we define cost effectiveness as the ratio of the relative cost to perform a combination of interventions to the health impact realised from that combination of interventions. The health impact is defined as the DALYs averted (DA), the difference between the DALYs simulated under the action (a_t) and the DALYs simulated under a null action, zero input or intervention. Therefore a calculation of DA requires a minimum of two simulations from a model, in which the action is simulated and compared to the simulated outcomes from no action under the same model parametrisation (θ).

Different decision makers may have different upper limits on considering cost-effectiveness of decisions. For example the National Institute of Clinical Excellence (NICE) in the United Kingdom will consider interventions to be cost-effective if they cost less than £20,000 per DALY averted. This would add a constraint on which solutions may be feasible for the geography being modelled. Generally, for the application of these techniques we'll make the assumption that the most cost-effective solutions (V^*) are optimal, for decision-makers, as they are the most cost efficient (for resource constrained environments) [88].

2.3.5 Bandit Approaches for Simulated Learning

Each Bandit algorithm outlined in this section performs sequential batch exploration of actions, towards optimisation of an unknown stochastic reward function $\mathcal{R}^a(r)$. At each batch for the round (t) we will choose $j = 1, 2, \dots, B$ up until the batch size (B) actions $a_t^j \in \mathcal{A}$. Due to the computational expense of calculating $\mathbf{r}_t \sim \mathcal{R}^{a_t}$ and the size of the entire action space $\mathcal{A}$, we wish to find solutions of maximal reward in as few batches (n) as possible. The goal being to approximate $a^* = \operatorname{argmax}_{a \in \mathcal{A}_c} Q(a)$ without prohibitively expensive computation for all possible actions, therefore using a subset $\mathcal{A}_c \in \mathcal{A}$ of the action space. *Note:* we have used n to define both the number of batches and previously the number of rounds for the bandit problem, this is based on the assumption that evaluation of a batch or a single run requires the same computational time under the ability to perform parallel computations, depending on the availability of the infrastructure, which would necessarily inform the batch size (B), normally of the order number of computational cores. Therefore the descriptions are equivalent while we'll use the extension of the bandit to a batch of actions for these computationally expensive, large action-space problems.

2.3.5.1 Upper/Lower Confidence Bound (GP-ULCB)

Based on previous discussion on sub-linear total regret we've extended a Gaussian Process Upper/Lower Confidence Bound (GP-ULCB) algorithm, inspired by Gaussian Process Regression (GPR) and work on Upper Confidence Bound (UCB) solutions to the Multi-Armed Bandit problem [17, 18]. This is a formulation which combines the natural confidence bounds of Gaussian processes with stochastic MAB problems, and the variants have already been proposed in the form of GP-UCB [186] and GP-UCB-PE [54]. The use of Gaussian Process Regression is detailed in Chapter 3.1.1, but it is here used to perform inference on a continuous action-value function $Q(a)$, based on successive batches of simulated rewards.

The algorithm is initialised with a random sample of a discrete action space ($\mathcal{A}_c$). Subsequent actions are chosen to further explore the action-value function regressed by GPR on all preceding simulation runs. The choice of using both *upper* and *lower* confidence bounds (UCB and LCB respectively) was made due to the large variance often observed in rewards. Specifically, multiple minima and maxima may occur in the action-value function, motivating a search for both optimal and poor actions. In this case increasing information about poor performing actions may help to avoid taking poor decisions, especially as we have a limited computational budget, increasing the

value assigned to exploring poor decisions may be considered necessary for high impact decision making. Should the mixing factor f_m be increased to unity, this approach would be reduced to an Upper Confidence Bound (UCB) only search. Repetition of simulated actions is avoided in a batch via the masking factor f_c , which defines the minimum distance between selected actions within a batch.

Result: $Q(\mathbf{a})$

Input: random discretised actions $\mathbf{a}_0 \in \mathcal{A}_c$;

GP priors $m_0 = \mathbf{0}$, l , σ_f^2 , σ_n^2 [Section 3.1.1];

B = batch size, f_m = mixing factor, f_c = masking factor [Section 2.3.5.1];

for $t = 1, 2, \dots, n$ **do**

 reset: $\mathbf{a}_{\text{UCB}}, \mathbf{a}_{\text{LCB}}, \mathcal{A}_c$;

for $j = 1, 2, \dots, B$. **do**

if $j < B \times f_m$ **then**

$a_t^j = \operatorname{argmax}_{a \in \mathcal{A}_c} [m_{t-1}(\mathbf{a}) + \beta \sqrt{k_{t-1}(\mathbf{a}, \mathbf{a}')}];$

 mask: $\mathbf{a}_{\text{UCB}}$ where $|a - a_t^j| < l \times f_c$;

 update: $\mathbf{a}_{\text{UCB}} \notin \mathcal{A}_c$;

else

$a_t^j = \operatorname{argmin}_{a \in \mathcal{A}_c} [m_{t-1}(\mathbf{a}) - \beta \sqrt{k_{t-1}(\mathbf{a}, \mathbf{a}')}];$

 mask: $\mathbf{a}_{\text{LCB}}$ where $|a - a_t^j| < l \times f_c$;

 update: $\mathbf{a}_{\text{LCB}} \notin \mathcal{A}_c$;

end

end

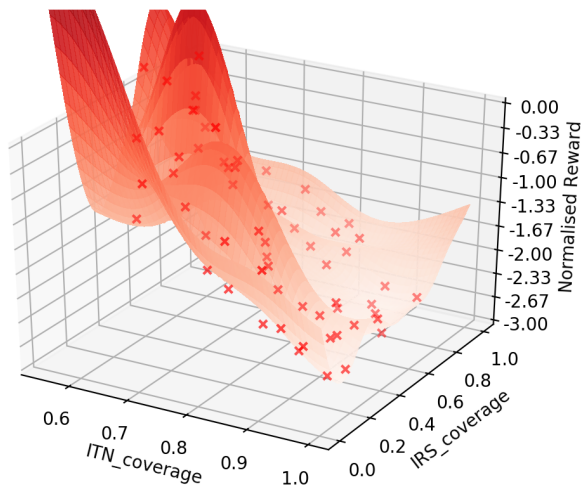
 Return: $\mathbf{r}_t \sim \mathcal{R}^{a_t}$ [Section 2.2];

 Update $\mathcal{GP}$ posterior for $Q(\mathbf{a})$: mean $m_t(\mathbf{a})$, covariance $k_t(\mathbf{a}, \mathbf{a}')$ including l [Section 3.1.2] ;

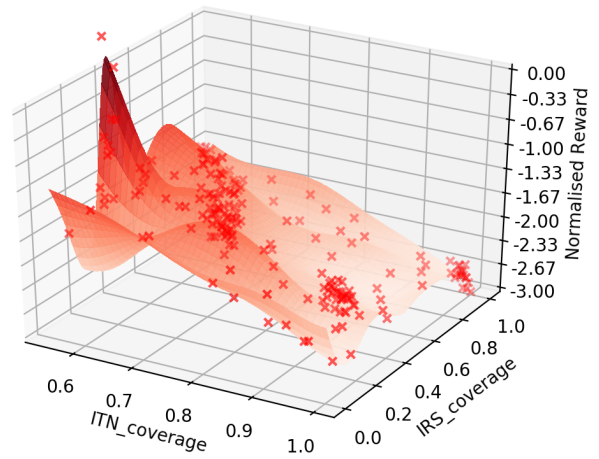
end

Algorithm 1: GP-ULCB

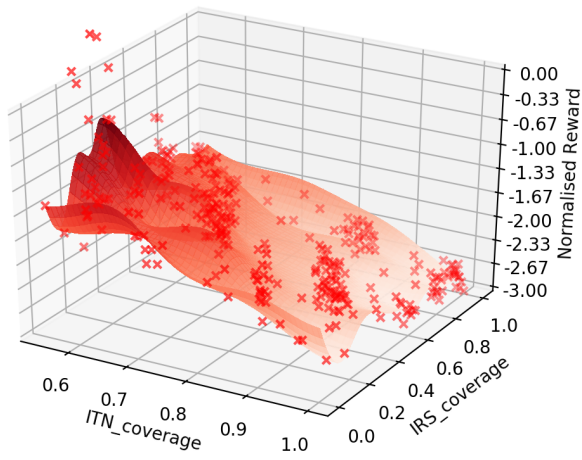
In Figure 2.3 we demonstrate the accumulation of batch samples in Algorithm 1, with a batch size of 64 applied to the specific problem to be outlined in Section 2.4. Qualitatively samples can be seen to shape the expectation ($m_t(\mathbf{a})$) in the value function $Q(\mathbf{a})$, with a 0.5 mixing factor (f_m), clusters of similar high and low performing solutions emerge. In this case a fixed masking factor (f_c) of 0.1 has been used. On inspection an adaptive schedule for the masking factor could help to spread out clusters of samples during earlier batches to encourage exploration, with subsequent annealing of this factor placing a focus on exploitation within batches for later rounds.



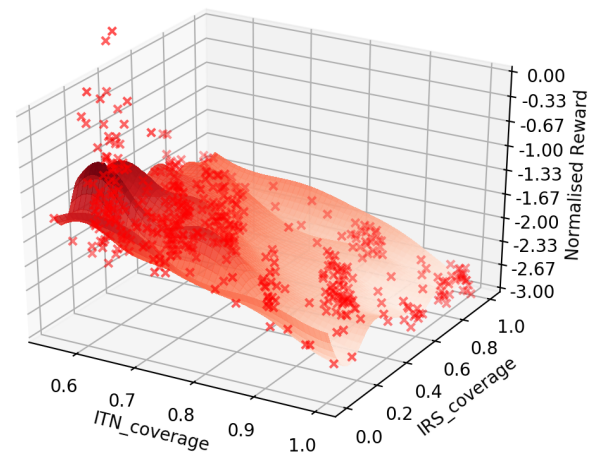
(a) First (Random) Batch



(b) Third Batch



(c) Fifth Batch



(d) Seventh Batch

Figure 2.3: Comparison of GP-ULCB Samples over subsequent batches from Open-Malaria Simulations for the problem outlined in Section 2.4.

2.3.5.2 Genetic Algorithms

A genetic algorithm (GA) is a biologically inspired, population-based search technique [93], specifically a meta-heuristic inspired by the process of natural selection. Here it is outlined as a method for exploration of the action-value function, where the genetic algorithm population size is equivalent to the batch size, a proposed batch of actions being run in parallel equivalent to a single bandit run, of our total budget (n). We use the reward generated for an action $r \sim \mathcal{R}^a$ as the measure of its fitness, and as our epidemiological environments return uncertain stochastic rewards for each action, there is noise in the fitness measure.

Given an evaluated population, in this case a set of actions and their stochastic rewards, the GA will then derive the next generation of the population. In this work we begin this process with roulette wheel selection [84] to select candidate actions. This biases selection of good actions to pass their ‘genetic material’ to the subsequent generation. The probability of selection p^j of the j^{th} action in a generation (i.e. run t), is defined in (2.10), where f^j is the fitness ($Q(a)$ normalised between $[0, 1]$) and P is the population size or total number of actions simulated.

$$p^j = \frac{f^j}{\sum_{k=1}^P f^k} \quad (2.10)$$

Each action in the subsequent generation is derived from two actions selected via this approach. Mimicking biological crossover of chromosomes, the two selected actions are mixed based on the comprising interventions, and the resulting action selected at random. Finally, a random subset of the interventions of each derived action is perturbed by adding noise. This sequence of processes defines how actions from the current generation are used to derive the next generation.

2.3.5.3 Batch Policy Gradient

A final proposed approach for online bandit learning from epidemiological simulations is a modified policy gradient [195] algorithm. This is a policy learning method, which for our single state problem reduces to learning an action. Policy gradient solutions tend to be preferred for problems with continuous or very large action spaces, while being robust to stochasticity. In this implementation $\mathbb{P}[r|a]$ is approximated by a neural network, with new policies sampled through ϵ -greedy exploration. The rewards r were normalised $[0, 1]$ from the batch results and the network (f) trained to update its weights ($\mathbf{w}$) associated with each policy $\mathbf{a} \in \mathcal{A}_c$, using gradient descent on the negative log-loss of the batch normalised rewards.

During training the actions $\mathbf{a}^j$ of Equation 2.11 will be chosen with probability $1 - \epsilon$:

$$\mathbf{a}^j = \underset{\mathbf{a} \sim \mathcal{A}_c}{\operatorname{argmax}}(f(\mathbf{w})) \quad (2.11)$$

and a random policy will be sampled from $\mathcal{A}_c$ with probability ϵ . Similar to the GP-ULCB algorithm, each $\mathbf{a}^j$ of batch t is sampled sequentially such that $\mathbf{a}^{j-1} \notin \mathcal{A}_c$. *Note:* for the specific instantiation of this approach presented in Section 2.4 a three layer fully connected network architecture has been used.

2.4 Example: Bandit Experiments on OpenMalaria Simulations

The following work demonstrates the results of the approach on simulations using an OpenMalaria epidemiological model calibrated to perform predictions for Rachonyo South in Kenya [189].

2.4.1 Simulated Interventions and Outputs

For this example we'll consider defining actions in terms of the main control interventions used in malaria intervention planning, such as mass-distribution of long-lasting insecticide-treated nets (ITNs); Indoor Residual Spraying (IRS) with pyrethroids; and the prompt and effective treatment of malaria. There are many others implemented across malaria models but for this example case study we'll restrict to two major components, ITNs and IRS, which are direct intervention strategies, targeting vector (mosquito) numbers. We'll define our actions a such that the first component is the deployment of nets, specifying the population coverage ($a_{\text{ITN}} \in (0, 1]$). The second component of the action is the application of seasonal spraying, specifying the proportion of population covered by this intervention ($a_{\text{IRS}} \in (0, 1]$). As observed, spraying regimens of IRS are conducted through alternating the intervention between April/June each simulated year [188]. Since the intervention planning task is framed as how much of the simulated population should be covered by a particular intervention, the action space $\mathcal{A}$ is constructed through $a_t \in \mathcal{A} = \{a_{\text{ITN}}, a_{\text{IRS}}\}$. Upon termination, the OpenMalaria model provides outcomes for each individual in the population for the executed simulation. If an individual experienced a malaria episode (ME_k), the simulation results allow years of life with disability (YLD) to be quantified (See (2.12)). Additionally, if an individual contracted malaria and subsequently died (D_z), either directly or indirectly from malaria, the simulation output

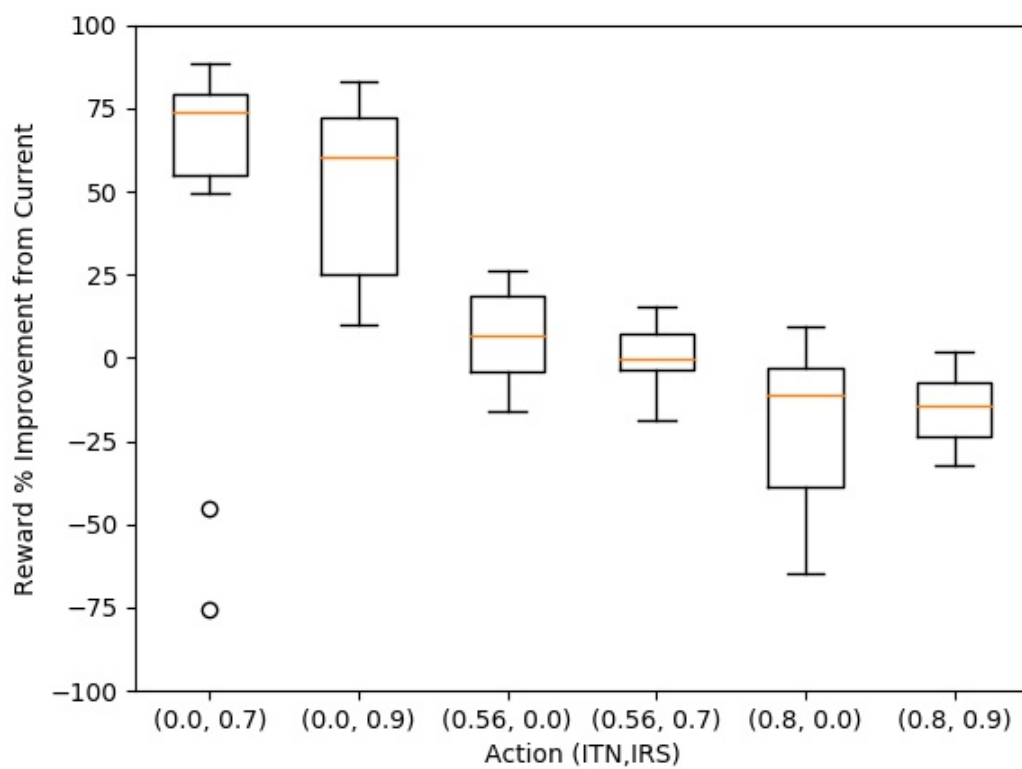


Figure 2.4: Boxplots indicating reward distributions ($\mathbb{P}[r|a]$) of a fixed number of actions run through OpenMalaria simulation rounds as Bandit arms. Rewards are relative, based on the percentage improvement over the currently implemented deterministic policy.

allows years of life lost (YLL) to be calculated (See (2.13)). We also use a discount factor $\gamma = 0.97$ to discount the value of future years of life lost, and a life expectancy of 46.6 years [134]. The work of [45] may be referred to for an explicit mapping of OpenMalaria simulation outputs to the calculation of DALYs. This is where values for the functions **Duration** (duration in years of malaria episode) and **Weight** (Age dependant weight of disability experienced by malaria episode $\in [0, 1]$) have been collected.

$$\text{YLD} = \sum_{k=0}^K \text{Duration}(\text{ME}_k) * \text{Weight}(\text{Age}(\text{ME}_k)) \quad (2.12)$$

$$\text{YLL}_z = \max(0, \text{LifeExpectancy} - \text{Age}(\text{D}_z)) \quad (2.13)$$

$$\text{YLL} = \sum_{z=0}^Z \text{YLL}_z \times \gamma^{\text{YLL}_z} \quad (2.14)$$

$$\text{DALY} = \text{YLL} + \text{YLD} \quad (2.15)$$

In this work we simulate two types of costs, the cost to treat and manage malaria episodes, and the cost to implement interventions which minimise malaria prevalence. We call these healthcare system costs (HSC), and intervention costs (IC) respectively. For each malaria episode that a patient seeks treatment, hospitals incur costs to treat the disease, to manage the patient's recovery process, and to deal with the patient's death, if that were to occur. As such, the HSC can be broken down in terms of total in-hospital treatment costs (TTC), the total in-hospital recovery cost (TRC), and the cost for in-hospital mortality (See (2.16-2.18)). We use values from the literature [198] to define the costs implemented in this work.

$$\text{TTC} = \sum_{k=0}^K \text{Cost}(\text{Treatment}(\text{InHospital}(\text{ME}_k))) \quad (2.16)$$

$$\text{TRC} = \sum_{k=0}^K \text{Cost}(\text{Recovery}(\text{InHospital}(\text{ME}_k))) \quad (2.17)$$

$$\text{HSC} = \text{TTC} + \text{TRC} + \sum_{z=0}^Z \text{Cost}(\text{InHospital}(\text{D}_z)) \quad (2.18)$$

The cost of the intervention (C_{int}) is the sum of the numbers of individuals covered by each intervention by the average cost of deploying the intervention to an individual. For the region in Kenya under consideration it costs \$8.52 per net and \$0.73 per person covered by spraying intervention. The average cost to seek hospital treatment

per person is \$0.60, and this value is associated with transportation and consumables. [188]. So the cost effectiveness will be quantified as the cost per DALY averted (C_{DA}):

$$C_{\text{DA}} = \frac{\text{HSC}_{\text{int}} - \text{HSC}_{\text{noint}} + C_{\text{int}}}{\text{DA}} \quad (2.19)$$

In summary, actions consist of proportions of the interventions of IRS and ITN which may be distributed to the simulated humans, and rewards indicate the cost effectiveness of the deployment over 5 simulated years in terms of cost per DALY averted (which should be minimised).

2.4.2 System Implementation and Deployment

For this work we used OpenMalaria commit a50730b. The simulation environment was run on a 4 node cluster of machines, each with 64 hyper-threaded cores (2.20GHz Intel Xeon® CPU E5-2660). On these processors, running one instance of an OpenMalaria simulation, for a representative human population size (100,000), returns results in the time-frame of days. Running experiments in batches can thus take advantage of the natural parallelism of the deployment environment. Parallelism was implemented using Python’s *multiprocessing* package. This package supports spawning processes using an API, and was used to execute OpenMalaria simulations, passing the parametrised model as an argument. As previously mentioned (Section 2.3.5) this results in a natural expression of the batch size B for each bandit model equal to the number of available processor cores.

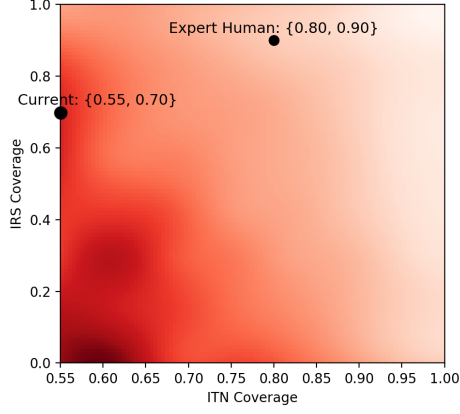
2.4.3 Comparison with Results from Expert Humans

Published studies [188] give a direct reference to human expert decisions made using OpenMalaria as a research tool, specifically to compare the cost-effectiveness of different interventions in Rachuonyo South District. Their findings stated that the current intervention plan, expressed as an action of 56% a_{ITN} , 70% a_{IRS} was the most cost-effective with regards to C_{DA} , while they also recommended that increasing this to 80% a_{ITN} and 90% a_{IRS} (including a school-based screen and treat program) would have the greatest health impact for DALYs averted. In this work we use the same parameterisation θ_m of OpenMalaria, but instead explore an automated answer to a less constrained problem for the decision maker: *Given the current intervention strategy, what policy decisions can be made for the next 5 years to improve cost-effectiveness?*

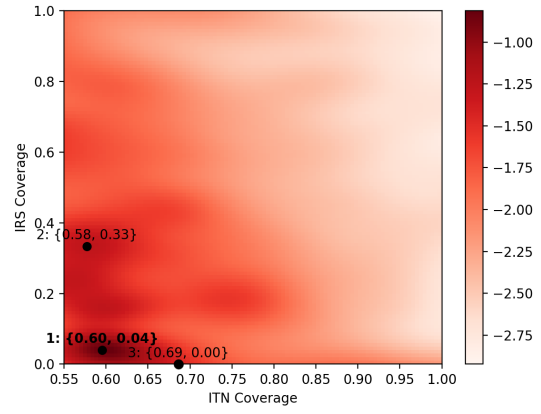
Our results indicate that all three bandit models extract the same top three performing emergent control interventions, with respect to our primary evaluation metric, C_{DA} (Table 2.1):

- Maintain a_{ITN} and stop a_{IRS} ,
- Maintain a_{ITN} and reduce a_{IRS} ,
- Increase a_{ITN} and stop a_{IRS} .

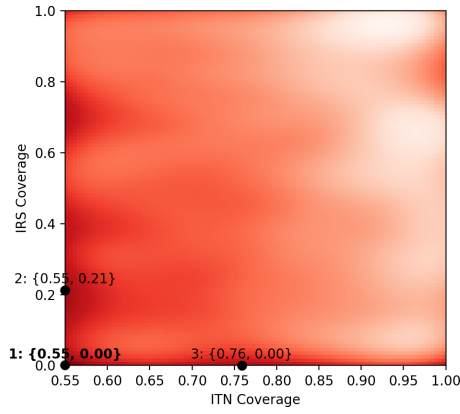
These findings are extracted from the action-value function ($Q(a)$) surface maxima of the posterior mean $m(\mathbf{a})$ based on Gaussian Process regression of rewards $\mathbf{r}_t$ collected by each respective bandit. Figure 2.5 demonstrates visualisations of $\log(Q(a))$ action-value surfaces, generated from each Bandit, after 8 iterations of a 64 batch size and running a total 512 simulations. The surfaces describe each Bandit's exploration of the available actions a_{ITN} and a_{IRS} for a human decision maker in Rachuonyo South.



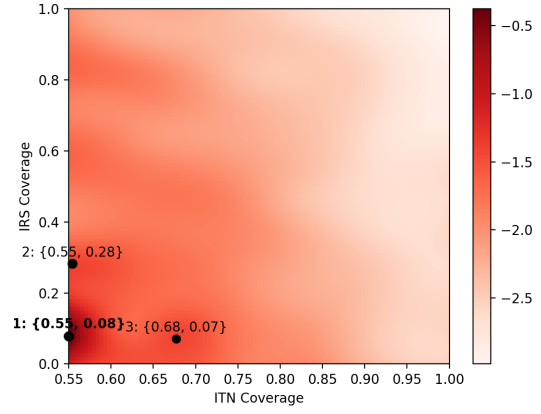
(a) Current and Expert Human recommended policies



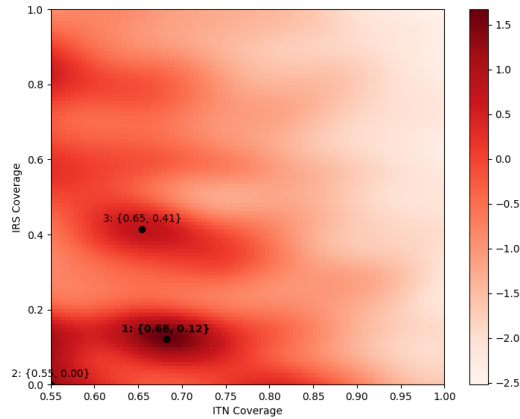
(b) GP-ULCB



(c) Genetic Algorithm



(d) Batch Policy Gradient



(e) UCB

Figure 2.5: Comparison of Bandit Approaches on OpenMalaria Simulations and their Top 3 Policies. $Q(a)$ mean action-value surfaces ($m(a)$)

GP-ULCB 6.2.4			Genetic Algorithm			Batch Policy Gradient			UCB		
Action $\mathbf{a}$	C_{DA}	DA	C_{int}	Action $\mathbf{a}$	C_{DA}	DA	C_{int}	Action $\mathbf{a}$	C_{DA}	DA	C_{int}
$\{\mathbf{60}, \mathbf{4}\}$	28.9	11800	514000	$\{\mathbf{55}, \mathbf{0}\}$	25.5	8690	458000	$\{\mathbf{55}, \mathbf{8}\}$	27.6	11600	477000
$\{\mathbf{58}, \mathbf{33}\}$	30.1	12300	519000	$\{\mathbf{55}, \mathbf{21}\}$	26.8	11600	477000	$\{\mathbf{55}, \mathbf{28}\}$	30.2	11000	489000
$\{\mathbf{69}, \mathbf{0}\}$	30.9	13000	579000	$\{\mathbf{76}, \mathbf{0}\}$	28.3	11700	632000	$\{\mathbf{68}, \mathbf{7}\}$	30.3	13800	589000
								$\{\mathbf{68}, \mathbf{12}\}$	33.0	12400	409000
								$\{\mathbf{55}, \mathbf{0}\}$	33.1	6620	345000
								$\{\mathbf{65}, \mathbf{41}\}$	33.3	12300	405000

Table 2.1: Top three actions with respect to C_{DA} evaluated by each bandit model. Action: $\{a_{\text{ITN}}\%, a_{\text{IRS}}\%\}$, C_{DA} : Cost per DALY Averted USD, DA: DALYs Averted, C_{int} : Intervention Costs USD.

2.4.4 Evaluation

The methods shown give a comprehensive evaluation, exploring the cost-effectiveness for an action consisting of the two main malaria interventions for a single location. Such fundamental insight is often missing from empirical studies, which are grounded in determining how much of a single intervention may be implemented to maximise a particular performance metric. Interestingly, at the surface the results could challenge the current sentiment in the community - that policy makers in Sub-Saharan Africa should maximise ITN coverage *before* looking into other intervention strategies.

One 2017 study even states that:

‘Coverage of ITNs was consistently the most cost-effective intervention across a range of transmission settings and was found to occur early in the cost-effectiveness scale-up pathway. RS, RTS,S and SMC vaccinations entered the cost-effective pathway once ITN coverage had been maximised. [220]’

Instead, our results suggest that when the cost-effectiveness of strategies is considered, after achieving a threshold of nets deployed for the environment, it may be more cost-effective to start spraying a small proportion of households (approx. 20-30%), instead of continuing to scale up the deployment of insecticide treated nets.

If additional investment is available, then these resources should be allocated to scale up the coverage of bed nets and further maximise health outcomes. With unlimited resources there are health benefits in the maximization of nets first, however if policy makers face tougher budget constraints, there are other factors that should influence their decisions and any smaller investment can more effectively be used to spray households. We have not explored if interventions could be deployed at different times of year, or even if multiple actions could have been concurrently deployed in a population. Further, the current simulation did not permit interventions to be targeted to specific subsets of the population (e.g. households with young children). Finally, this work is specific to one studied location in Western Kenya, and the generalisation of the insights from multiple simulated learning strategies gathering insights across expansive environments e.g. Sub-Saharan Africa has not been explored, while providing a case study of application of online learning to the task of one-step decision making for intervention planning.

Applying simulated learning strategies to the problem of epidemiological intervention planning and using relevant real data for studied regions is a challenge. In the

sense that new models are difficult to configure, if one does not already exist, while the acquisition of the requisite data for calibrating the models may be prohibitively expensive, if it does not already exist. Therefore, before even running epidemiological models which infrastructure can support at scale, significant investment has been made to make the available models and associated data available for research. The current expense of both real and simulated data should be appreciated when understanding the problem and so we look to techniques which can be ‘data-efficient’ in learning effective plans as recommendations. We will aim to continue to present progress to a scalable distributed infrastructure, that supports evaluation of intervention plans and the resources which makes this possible. In [32] this work has been presented by framing the process of finding an optimal malaria intervention plan as a stochastic multi-armed bandit problem, and implementation of three bandit-based strategies to find optimal actions for a single environment.

2.5 Extensions to Online Learning

The techniques presented have been selected and designed with the view to deployment on larger action spaces as detailed in the Section 2.4.4. More compute time, expansive environments and plans are a requirement for the real-world human decision maker, while other data sources exist outside of the OpenMalaria simulation environment, for example malaria mapping projects, which could serve as input to new models, perhaps requiring function approximation to be discussed in Chapter 3. There is also the opportunity to embed online learning approaches deeper into the simulation environment, passing control of simulation parameters in order to allow balancing of computational expense with efficient policy space exploration. This work is viewed as an emerging application for deploying further novel exploration techniques.

Figure 2.6 for example explores the changes in the action-value function generated simply from a perturbation in the timing of the intervention combinations. We’ll look to expand on this concept of dynamic policy learning in Chapter 6.

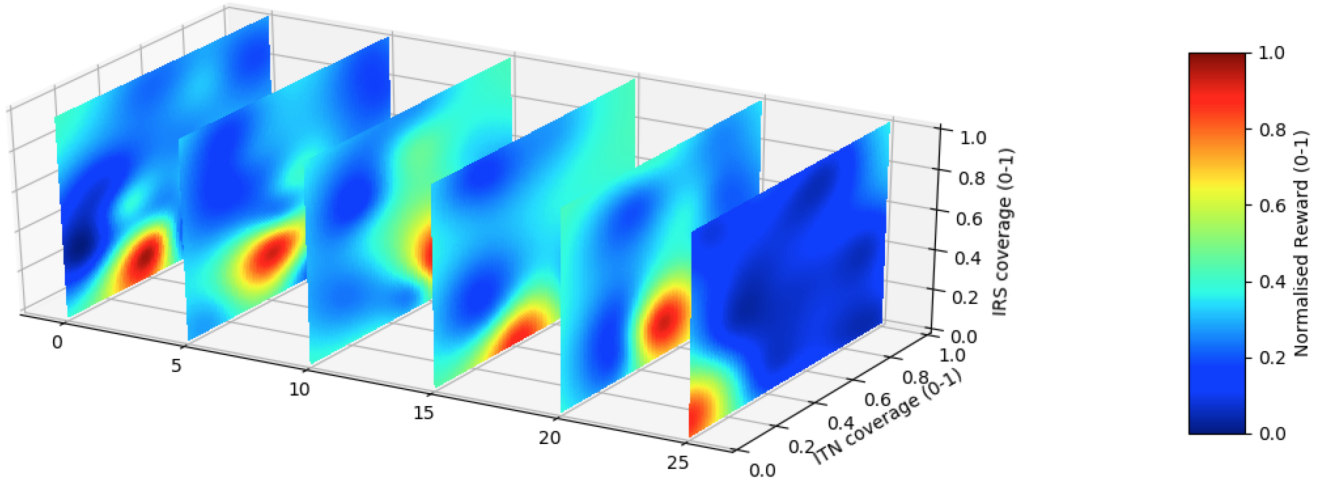


Figure 2.6: Slices of action-value function $Q(a)$ for actions (a) as a combination of Insecticide Treated Net and Indoor Residual Spraying coverage changing through time (days), for OpenMalaria Simulations

Chapter 3

Learning Models for Prediction from Epidemiological Simulation

The sciences do not try to explain, they hardly even try to interpret, they mainly make models. By a model is meant a mathematical construct which, with the addition of certain verbal interpretations, describes observed phenomena. The justification of such a mathematical construct is solely and precisely that it is expected to work - that is, correctly to describe phenomena from a reasonably wide area.

John von Neumann, 1995

This chapter will cover a more ‘traditional’ approach to Machine learning (ML) and is based on function approximation [77], statistical learning theory [208] or supervised learning [110] applied to epidemiological modelling. While such approaches are already used for epidemiological modelling to develop phenomenological ‘black-box’ models which describe disease transmission [115], this will not be the focus. Instead, with access to training data and examples or experience from epidemiological models, we concentrate on how the machinery of data-driven inference may operate in tandem with mechanistic model based insight. We’ll demonstrate a set of data-efficient ML techniques using training examples from epidemiological model runs to learn functional approximations of computationally expensive epidemiological models. Such ML models learnt from an underlying epidemiological model will be given the term *surrogate* model in this work. The surrogate model is a black-box approximation which will afford us benefits in our handling of tasks of prediction, control and planning.

The chapter is separated into three parts, each dealing with one of the following aspects:

1. Surrogate model descriptions for speed, sharing and analysis of computationally expensive models.
2. Function approximation for uncertainty quantification and model calibration for improved predictions.
3. Combination of epidemiological models through committees based on learnt function approximations.

For existing phenomenological prediction approaches, models will aim to generate projections of the future state of the system, based on the available data (or training examples). It may be reasonable for several prediction tasks that we assume no significant change to the underlying epidemiological process, for example due to already imposed intervention plans, evolution of the disease or climate variations. While a challenge remains that if such changes have not already been observed, it is a fruitless task to aim to predict outcomes in the presence of these changes.

The current practice to generate new predictions under significant changes is to rely on the centuries of research, evolving the ‘gold-standard’ of mechanistic epidemiological models for disease progression. We will therefore place our focus on how Machine learning (ML) may be used to infer predictions across multiple policies, where model simulations act as observations under changing intervention plans. Later such ML models learnt from simulations will be used directly in the control and planning process.

As discussed, ML techniques may be used for predictions based on real-world observations alone. Significantly repeated and relevant examples of this have been demonstrated in the context of the COVID-19 Pandemic, due to the unprecedented amount of data [166], examples of shared research efforts [157] and global impact [136]. However, all of the observed and collected data allows us only to learn an approximate model based on a single experience or set of observations. Through the addition of model-based observations, these may provide additional training examples or sets of experience to guide planning and control. Such an approach to simulating predictions under varying intervention plans is novel for epidemiological applications but is at the heart of Reinforcement learning (RL) for multi-step decision making. Another point which should be made is that if desired to replicate the prediction task for other diseases; the number of real-world observations may be further limited; of far lower

quality; and not openly shared, making a repeatable approach to modelling predictions impossible without guidance from mechanistic models. An approach is needed to make such models repeatedly usable, along with the appropriate characterisation of model uncertainty.

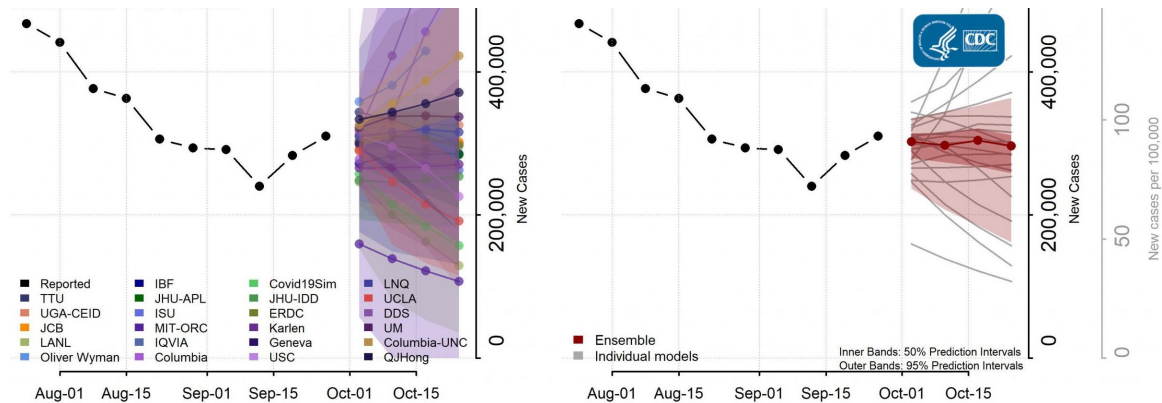


Figure 3.1: Multiple model ensemble predictions generated for Covid-19 model submissions in the United States of America, by the Centres for Disease Control and Prevention (CDC) ¹

Figure 3.1 demonstrates that while prediction models for diseases, such as Covid-19, may be developed in great number, their predictions don't directly agree. This heterogeneity is encouraged with different mechanisms, modelling assumptions and approaches underlying the various models. An ensemble prediction may be chosen to meaningfully combine estimates for an uncertain future, in the case of Figure 3.1 this is the median prediction, along with 95% confidence interval. While each prediction is based on the same observational data, some models include additional data with regards to interventions such that a prediction may be made under the current policy. Though these predictions are not exposed as a function of varying policies or intervention plans, should they not be fixed or deterministic, as is the assumption for all presented predictions.

To look further into Figure 3.1, uncertainty is based on each model's prediction as a median and quantile range. The model ensemble is based on the aggregation of these uncertainty measures, with equal contribution from each submitted approach. We'll discuss methods to achieve this without making the assumption that all models are created equal in Section 5.3.1. The often quoted adage of statistician George Box is directly applicable to this scenario [40]

'All models are *wrong* but some are useful'

¹<https://www.cdc.gov/coronavirus/2019-ncov/cases-updates/forecasts-cases.html>

This may be accompanied with the addition that in the case of ensemble forecasts or committee predictions some models may also be more useful than others.

Through the steps outlined we can decompose forecasts as functions of intervention plans, performing ensemble combinations across possible predictions conditioned on intervention, along with providing an unbiased multi-model based estimator for combined predictions under uncertainty. Through this chapter we hope to motivate that such model structures may undergo semi-automated calibration for predictions, given a base model structure, easing the burden of computation and use of shared models, along with providing insights into the sensitivity of models to changes in input parameters in a manner which is efficient and utilises computations already performed during the calibration process. Mechanistic models may be thought to impose interventions as additional parameterisations to the baseline model parameters and so through a baseline calibration, we may extend calibrated models for the purposes of control and planning in Chapters 5 and 6.

3.1 Machine Learning Surrogates of Epidemiological Models

Due to heterogeneity, computational cost and lack of a common description of uncertainty for any given epidemiological model, surrogate model approximations provide advantages, which may outweigh the fact that we are introducing errors due to approximation. Though it should be noted these errors are based on our approximation of a model which itself has it's own approximation error. Hence it may be argued that a surrogate ML model may accelerate our ability to explore epidemiological control and planning strategies, which nevertheless will always be lower bounded by the epidemiological model's unknown approximation error. ML function approximations provide the following benefits:

- Approximate simulation results may be queried to provide instant feedback for decision-maker or user.
- Any algorithm may query a surrogate model to perform it's own simulated planning steps.
- As introduced in Chapter 2 functional approximation may be used to guide 'true' simulated action selection.

For this work we'll focus on the most data-efficient strategy which naturally incorporates uncertainty quantification in its definition, Gaussian Process Regression (GPR).

3.1.1 Gaussian Process Regression

The Gaussian Process (GP) [218] is a well-known statistical learning model, used extensively for probabilistic non-linear regression. The task of regression aligns with parametric, phenomenological, or black-box epidemiological model design for prediction, specifically how predictions as forecasts may be learnt of future epidemiological state. Although Gaussian Process Regression is not a standard modelling approach in epidemiology, there are examples of its use in geospatial modelling [207]. This work will focus on motivating its use in a pipeline for temporal decision-making. Gaussian Process Regression (GPR) targets a functional approximation for an underlying stochastic process, a natural fit for capturing uncertainty from epidemiological model training examples.

The outputs used from epidemiological simulations will be considered as stochastic, either through mechanisms of stochasticity within the model or based on stochastic distributions of model input parameters, samples of which generate a stochastic output. Firstly a general overview of Gaussian Process Regression (GPR) is necessary before generating the appropriate mapping for prediction tasks.

A Gaussian process is fully specified by its mean function $m(\mathbf{x})$ and covariance function $k(\mathbf{x}, \mathbf{x}')$. This is a natural generalisation of the Gaussian distribution, whose mean and covariance is a vector and matrix, respectively. The Gaussian Process entertains a probability distribution over some (real) underlying stochastic process $f(\mathbf{x})$. Defining:

$$m(\mathbf{x}) = \mathbb{E}[f(\mathbf{x})] \quad (3.1)$$

$$k(\mathbf{x}, \mathbf{x}') = \mathbb{E}[(f(\mathbf{x}) - m(\mathbf{x}))(f(\mathbf{x}') - m(\mathbf{x}'))] \quad (3.2)$$

the Gaussian process is specified as

$$f(\mathbf{x}) \sim \mathcal{GP}(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}'))$$

These random variables represent the value of the stochastic process $f(\mathbf{x})$ at a location $\mathbf{x} \in \mathcal{X}$. The location variable in our application will be over states and actions. We discuss this further in Chapter 6. Judicious choices of kernel function and parameter values need to be made for the successful application of GPR.

3.1.1.1 Choice of Covariance Kernel Function

At the heart of GPR we expect a level of smoothness in our inference, that is to say neighbouring observations have a reasonable correlation, this correlation weakening from unity with distance between observations. Specifically the value of $f(\mathbf{x})$ is strongly correlated to values close to $\mathbf{x}$ eg. $f(\mathbf{x} + \delta\mathbf{x})$ where $\mathbf{x}' = \mathbf{x} + \delta\mathbf{x}$. Intuitively this captures a wide range of well behaved dynamic systems where we expect similar input conditions to lead to similar outcomes. We consider such responses to be captured by stationary functions such that the expected response of our system is not changing over time, while the remaining choices of the form of the kernel remain somewhat an 'art form', as in the designing of many Machine learning models (eg. neural network architectures). A common form exists which we'll use as a running example in this work.

This is the Matérn covariance function, specifically the $\nu = 5/2$ covariance function, which is a popular choice in modelling physical systems [55], and included here for the purposes of demonstration of results. Alternative kernel methods and selections or combinations may always be used, but this is not the contribution of the work and so we will stick to describing a single kernel use for the purpose of demonstration. Enumerating all of our hyper-parameters σ_f^2 , signal variance is the maximum allowable variance and l the length-scale of the covariance kernel, over which we condition correlations between observations. Finally we specify the value of the kernel function in terms of the input distance $r = |\mathbf{x} - \mathbf{x}'|$.

$$k_{\nu=5/2}(r) = \sigma_f^2 \left(1 + \frac{\sqrt{5}r}{l} + \frac{5r^2}{3l^2} \right) e^{-\frac{\sqrt{5}r}{l}} \quad (3.3)$$

$k(\mathbf{x}, \mathbf{x}')$ is a covariance function, used to create the square matrix K , indicating the values of the covariance function at pairs of evaluations.

3.1.1.2 Prediction with Stochastic Epidemiological Observations

The specification of the covariance function implies a distribution over functions. To see this, we can draw samples from the distribution of functions evaluated at any number of points; in detail, we choose a number of input points denoted X_* .

$$f_* \sim \mathcal{N}(\mathbf{0}, K(X_*, X_*))$$

In which X_* is a vector of our test inputs (or locations) of the regression, f_* . Therefore, without observations or training inputs, this fully specifies our Gaussian process prior.

We consider an underlying stochastic or noisy epidemiological function $y = f(\mathbf{x}) + \epsilon$, ϵ being additive independently identically distributed (i.i.d) Gaussian noise with variance σ_n^2 . Here we've assumed apriori that the underlying stochastic process is unbiased and $m(\mathbf{x})$ is $\mathbf{0}$. Without other reasonable prior knowledge, e.g. periodicity, this is a reasonable assumption, along with our beliefs of the underlying process that we have specified through the GP covariance function. Our stochastic epidemiological function (y) is therefore distributed such that:

$$y \sim \mathcal{N}(\mathbf{0}, K(X, X) + \sigma_n^2 I)$$

This now allows us to specify the joint distribution of the observed training values X and the function values at the test locations f_* under the Gaussian process prior.

$$\begin{bmatrix} \mathbf{y} \\ \mathbf{f}_* \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} K(X, X) + \sigma_n^2 I & K(X, X_*) \\ K(X_*, X) & K(X_*, X_*) \end{bmatrix}\right) \quad (3.4)$$

If there are n training points and n_* test points then $K(X, X_*)$ denotes the matrix of the covariance function evaluations at all pairs of training and test points and X the vector of our training inputs or results from epidemiological runs. The covariance pairings of $K(X, X)$, $K(X_*, X_*)$ and $K(X_*, X)$ follow respectively from test points (X_*) and training points (X).

Through constructing the conditional distribution based on the assumption of Gaussian prior observations this gives us the following:

$$\mathbf{f}_* | X, y, X_* \sim \mathcal{GP}(m(\mathbf{x}_*), k(\mathbf{x}_*, \mathbf{x}_'))$$

In the case of a zero mean function prior this reduces the prediction functions to:

$$m(\mathbf{x}_*) = K(X_*, X)[K(X, X) + \sigma_n^2 I]^{-1} \mathbf{y} \quad (3.5)$$

$$k(\mathbf{x}_*, \mathbf{x}_') = K(X_*, X_*) - K(X_*, X)[K(X, X) + \sigma_n^2 I]^{-1} K(X, X_*) \quad (3.6)$$

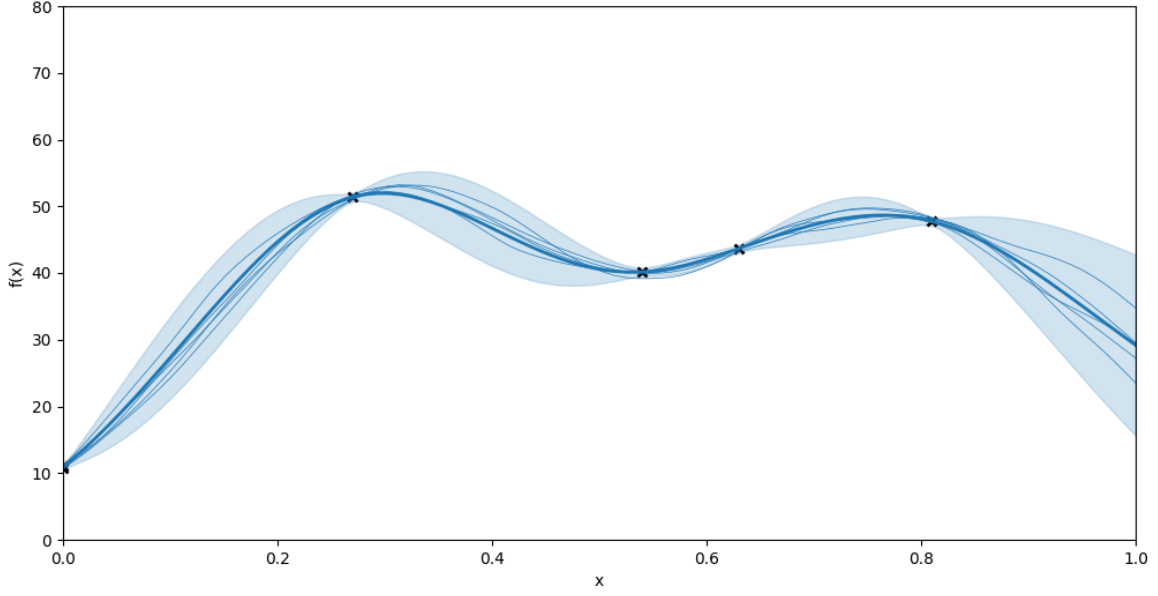
The above results have direct solutions conditioned on our GP hyper-parameters l , σ_f^2 and σ_n^2 , which we shall term $\theta_{\mathcal{GP}}$.

While judicious approximations can be made for these values we'll rely on an optimisation scheme to fit them based on observation across tasks. This may be done through maximisation of the log marginal likelihood:

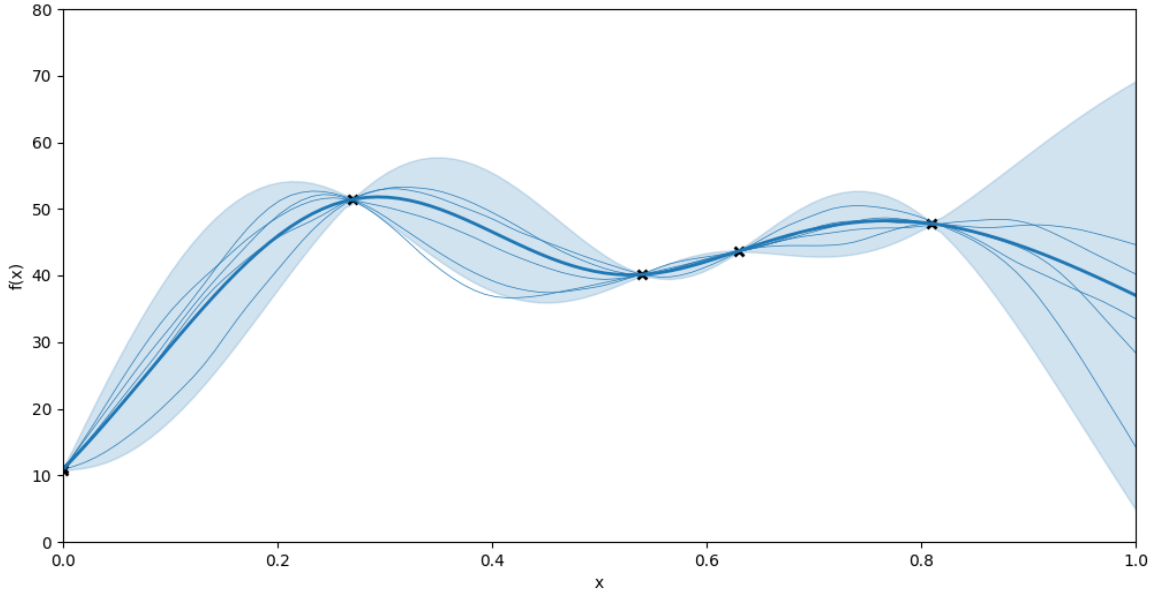
$$\log p(\mathbf{y}|X, \theta_{\mathcal{GP}}) = -\frac{1}{2}\mathbf{y}^T(K(X, X) + \sigma_n^2 I)^{-1}\mathbf{y} - \frac{1}{2}\log |K(X, X) + \sigma_n^2 I| - \frac{n}{2}\log(2\pi) \quad (3.7)$$

Performing Maximum Likelihood Estimation (MLE) based on maximisation through gradient based optimisation, recovers an empirical Bayes estimate to $\theta_{\mathcal{GP}}$. While full Bayesian inference may be performed to generate the posterior parameter distribution, the computations are analytically intractable, approximate methods are therefore deployed, for example Markov Chain Monte Carlo (MCMC) algorithms for sampling the $\mathcal{GP}$ prior [203].

We may see in Figure 3.2 the difference in GP inference based on judicious or expert guided choices (Figure 3.2a) of $\theta_{\mathcal{GP}}$ and those learnt through the maximised log likelihood optimisation (Figure 3.2b). Our learnt optimal parameter values inflate the epistemic uncertainty of our GPR based on a higher signal variance (σ_f^2). We'll further discuss the topic of Uncertainty quantification (UQ) in Section 3.3.



(a) Posterior GP inference based on underlying epidemiological function and judicious choice of $\mathcal{GP}$ parameters $\theta_{\mathcal{GP}}$: $l = 0.25$, $\sigma_f^2 = 100$, $\sigma_n^2 = 0.1$. Including 5 draws from the posterior distribution of functions



(b) Based on maximised log likelihood optimisation of GP parameters: $\mathcal{GP}$ parameters $\theta_{\mathcal{GP}}$: $l = 0.38$, $\sigma_f^2 = 1320$, $\sigma_n^2 = 0.02$. Including 5 draws from the posterior distribution

Figure 3.2: Comparison of posterior with selected (a) and learnt (b) MLE parameters

3.1.2 Action-Value Function Example

This section serves as an extension to the application of Gaussian processes introduced in Chapter 2 for learning the action-value function $Q(a)$ for the Multi-Armed Bandit (MAB) problem. Despite stochastic simulation results, the value function of similar actions should be highly correlated and fit the purposes for the use of the method for our application.

$$Q(a) = \mathbb{E}[r|a] = m(a) \quad (3.8)$$

If we consider that any simulated action (a) and set of actions ($\mathbf{a}$) return a stochastic scalar reward with mean function $m(\mathbf{a})$ and covariance $k(\mathbf{a}, \mathbf{a}') = \mathbb{E}[(R(\mathbf{a}) - m(\mathbf{a}))(R(\mathbf{a}') - m(\mathbf{a}'))]$, a Gaussian Process can be specified by these mean and covariance functions, where we now consider actions as inputs to the GP.

$$Q(\mathbf{a}) \sim \mathcal{GP}(m(\mathbf{a}), k(\mathbf{a}, \mathbf{a}'))$$

Gaussian Process Regression is a supervised learning technique, in which the stochastic scalar rewards $r_t \sim \mathcal{R}^{a_t}$ are used to train a Gaussian process to infer with confidence bounds the action-value function $Q(\mathbf{a})$ [218].

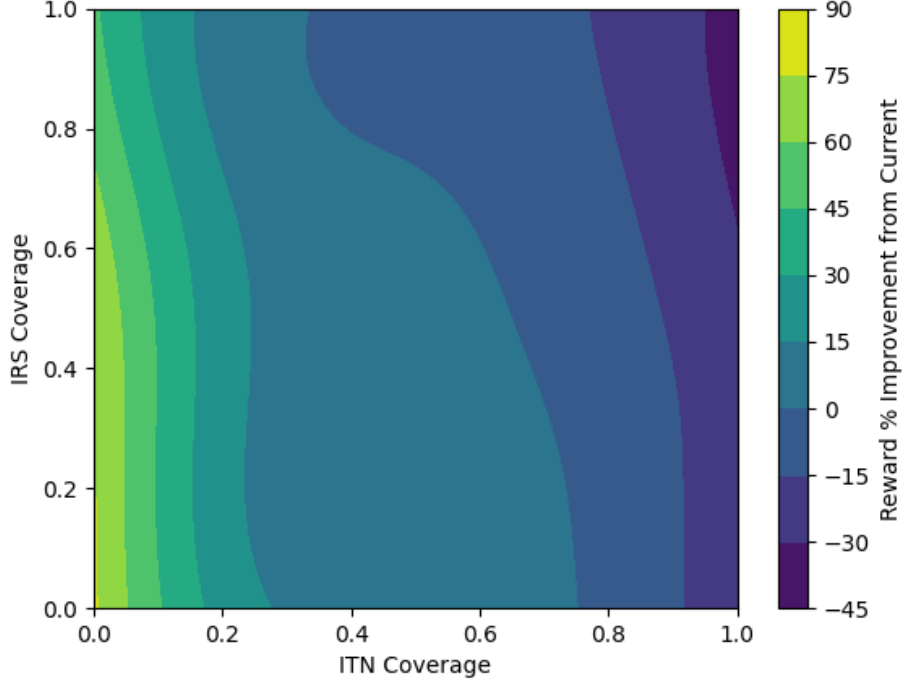
The learnt parameters describe the posterior distribution over $Q(\mathbf{a})$: Or over run t of batch k .

$$m_{t+1}(\mathbf{a}) = \mathbf{k}_i(\mathbf{a})^T (\mathbf{K}_i + \sigma^2 \mathbf{I})^{-1} R_i \quad (3.9)$$

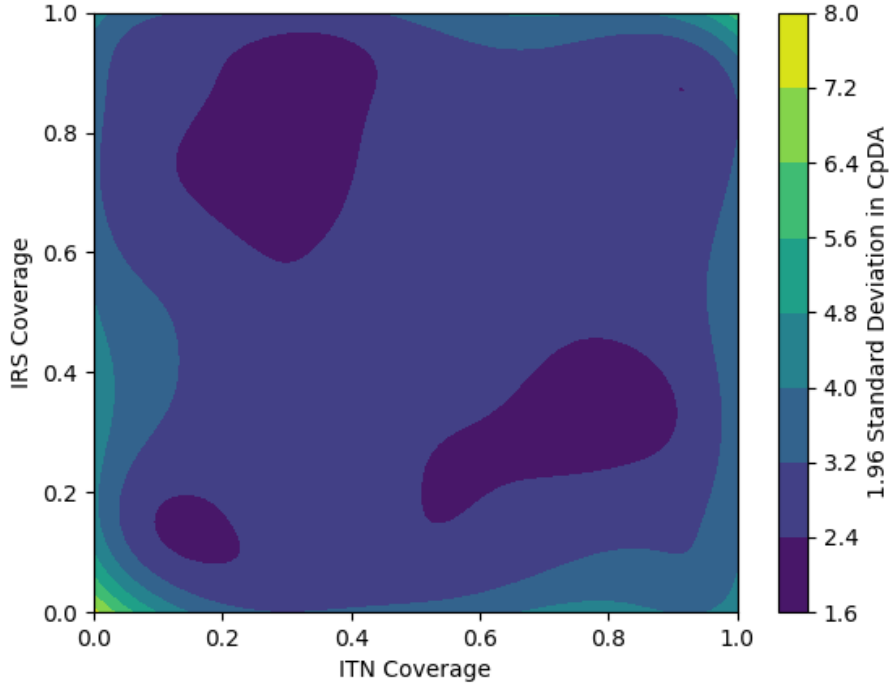
$$k_{i+1}(\mathbf{a}) = k(\mathbf{a}, \mathbf{a}') - \mathbf{k}_i(\mathbf{a})^T (\mathbf{K}_i + \sigma^2 \mathbf{I})^{-1} \mathbf{k}_i(\mathbf{a}) \quad (3.10)$$

At each location $\mathbf{a} \in A$, $\mathbf{k}_t(\mathbf{a}) = [k(\mathbf{a}_t^k, \mathbf{a})]_{\mathbf{a}_t^k \in A_c}$ and $\mathbf{K}_t = [k(\mathbf{a}, \mathbf{a}')]_{\mathbf{a}, \mathbf{a}' \in A_c}$, here σ^2 is the likelihood variance of the GP posterior, where covariance function or kernel $k(\mathbf{a}, \mathbf{a}')$ may be of Matérn form from Section 3.1.1.1.

As can be demonstrated by a two dimensional response surface for the action-value function $Q(a)$ in Figure 3.3a this is the mean function and the expected performance of a set of two interventions with regards to a performance criterion, in this case the Cost Per DALY Averted (CPDA) as outlined in Section 2.3.4.3, along with the upper confidence interval of $Q(a)$ shown in Figure 3.3b, where regions of lower inferred uncertainty in the response surface move down the colour scale to dark blue. This notion of learning a surrogate model or response surface will be extended to sequential decision making in Chapter 6.



(a) $Q(a)$ mean function for OpenMalaria Simulation experiments [39]



(b) $Q(a)$ standard deviation bound for OpenMalaria Simulation experiments

Figure 3.3: Learnt $\mathcal{GP}$ parameters $\theta_{\mathcal{GP}}$: $l = 0.64$, $\sigma_f^2 = 3380$, $\sigma_n^2 = 15.6$

3.2 Epidemiological Model Calibration

Up to this point we have assumed a calibrated epidemiological model for the purposes of model-based prediction. In essence a calibrated model implies that the model parameters (θ_m) have been selected, either as single values or distributions. This calibration will likely have been performed by the model developer for the specific disease and location, for which the model has been specified. The process of calibration takes the designed model structure and exposes meaningful parameters (θ_m) to a tuning such that a model achieves higher predictive performance. A model developer may have designed a model with physical parameters in mind and reference the literature for these parameter values, and simply look for the model structure to represent observations. Alternatively a grid-search of parameter values may be performed, which exhaustively evaluates model runs and fixed interval perturbations of the parameter values, specifying a hyper-dimensional grid in the number of parameters. The vertex associated with the parameter set which provides a best-fit (or minimum loss 3.2.2) then being chosen. In this work we'll advocate for an algorithmic approach to achieve best-fit, which may both improve the efficiency of our search for a parameter set, under computational constraints, while also providing additional information with regards to parameter sensitivity and allows us to incorporate additional information around parameter distributions.

For the purposes of this section we'll assume a predefined model structure with exposed model parameters (θ_m), while we may consider the mechanisms of the model as hidden under the black-box abstraction. Simply being given access to a disease model structure will almost always necessitate tuning or *calibration* in order to provide meaningful predictions. Therefore we'll look to develop mechanisms which can achieve this in an efficient manner, enabling large scale use and comparison of epidemiological models.

Any parameter set being calibrated for a mechanistic epidemiological model is often in the order of 10 [123]. Where these parameters will normally have a consistent epidemiological meaning and directly govern the model dynamics, the following list of parameter names consists of examples with more or less obvious physical interpretations; death rate [102], transmission rate [124], incubation period [12], vaccination rate [7], vector mortality rate [128], biting rate [52], Entomological Inoculation Rate (EIR) [227], etc. When performing calibrations for models of the same disease and same location, a mapping of any common parameter sets and to get similar model performance is noted to be a complex task [71]. So we will consider the task of

separately calibrating single models for a new geography. To perform such calibrations reference observational data is required (cases, deaths, vector numbers etc.). We may then interrogate further the impact of parameter values with regards to this observational data and model outputs.

3.2.1 Bayesian Optimisation

Bayesian optimisation is a principal global optimisation method driven by the exploration-exploitation paradigm, used in black-box optimisation of computationally expensive functions. Priors on our parameters may build in human expert knowledge from epidemiologists; and an acquisition function [214] which guides the exploration process, reducing unnecessary evaluations of expensive objective functions while maximising information [141]. Importantly Bayesian optimisation leverages approximations in the form of a surrogate model or response surface to remove the computational expense of direct evaluations of the objective function. This surrogate model is almost always a Gaussian process as we developed in Section 3.1.1. While other surrogate modelling methods exist for sequential model-based optimisation, we’ve developed a consistent set of approaches based on the GP posterior and will not divert, though methods such as Parzen Estimator approaches exist [33]. As epidemiological models tend not to be written with closed form solutions, we are led to approaches which treat the environment against which a candidate solution is being evaluated as a ‘black-box’. Specifically we seek to approximate an optimal solution given that the:

- Objective function ($f(x)$) is *unknown, non-linear, non-convex, non-derivative* and *stochastic*.
- Candidate solutions (x) are a *high dimensional* vector from the set of possible actions. ($\mathcal{A} \subset \mathbb{R}^d$), typically with $d \leq 20$ [75].
- Evaluation of $f(x)$ is computationally very expensive.

Therefore framing the process of exploring solutions as an optimisation problem:

$$\max_{x \in \mathcal{A} \subset \mathbb{R}^d} f(x) \quad (3.11)$$

For the purposes of calibration, candidate solutions (x or θ_m) exist across dimensions (d) (often a rectangular hyperspace with uniform ranges $x \in \mathcal{R}^d : a_i \leq x_i \leq b_i$), which are evaluated towards the maximisation of some objective function $f(x)$. The

objective function will return a scalar reward which quantifies the performance of the solution.

In outline as in Algorithm 2 we'll use Bayesian optimisation as a sequential optimisation schedule. Through each subsequent model run t , a new set of inputs to the model simulation are chosen $\mathbf{x}_{t+1}$ as the result of combining our GP surrogate model with the acquisition function which will be detailed in the following section.

```

for  $t = 1, 2, \dots, n$  do
    select new  $\mathbf{x}_{t+1}$  under acquisition function  $\alpha$  [Section 3.2.1.1];
     $\mathbf{x}_{t+1} = \underset{\mathbf{x} \in \mathcal{A}}{\operatorname{argmax}} \alpha(\mathbf{x}; \mathcal{D}_t)$ ;
    run model simulation to provide scalar output  $y_{t+1}$ ;
    new observations  $\mathcal{D}_{t+1} = \{\mathcal{D}_t, (\mathbf{x}_{t+1}, y_{t+1})\}$ ;
    Update  $\mathcal{GP}$  Posterior: mean and covariance
end

```

Algorithm 2: Bayesian optimisation Steps

Subsequent evaluations serve to reduce the *epistemic* uncertainty in the GP surrogate model of the loss function (Section 3.2.2) and improve our understanding of the captured *aleatoric* uncertainty.

3.2.1.1 Acquisition Function

Bayesian optimisation is guided by an acquisition function (α), this may have several functional forms. Popular acquisition functions are Upper Confidence Bound (UCB), Expected Improvement (EI) and Thompson Sampling - they may be grouped in the *optimistic*, *improvement-based* and *information-based* search methods respectively [177]. We've dealt with *optimistic* search in Chapter 2 and specifically the action-selection method of UCB for the Multi-Armed Bandit (MAB) which is analogous to our acquisition function for Bayesian optimisation.

The EI acquisition function is defined by the following equation:

$$\alpha_{EI} = \mathbb{E} [\max(f(\mathbf{x}_*) - f(\mathbf{x}^+), 0)] \quad (3.12)$$

$f(\mathbf{x}^+)$ is the best observed loss and $\mathbf{x}^+$ the parameters associated with this loss. Based on our GP model:

$$\alpha_{EI} = \begin{cases} (m(\mathbf{x}_*) - f(\mathbf{x}^+) - \epsilon)P(Z) + \sigma(\mathbf{x}^*)\rho(Z) & \text{if } \sigma(\mathbf{x}_*) > 0 \\ 0 & \text{if } \sigma(\mathbf{x}_*) = 0 \end{cases} \quad (3.13)$$

$m(\mathbf{x}^*)$ is the prediction mean and $\sigma(\mathbf{x}^*)$ the prediction standard deviation at the test location $\mathbf{x}^*$ of the Gaussian process. P is the cumulative density function (CDF) and ρ the probability density function (PDF) of the Gaussian distribution and

$$Z = \begin{cases} \frac{m(\mathbf{x}_*) - f(\mathbf{x}^+) - \epsilon}{\sigma(\mathbf{x}_*)} & \text{if } \sigma(\mathbf{x}_*) > 0 \\ 0 & \text{if } \sigma(\mathbf{x}_*) = 0 \end{cases} \quad (3.14)$$

Exploration and exploitation may be balanced via the P and ρ terms in the summation of α_{EI} with ϵ weighting this balance, a greater ϵ leading to greater exploration. Figure 3.4 plots our EI acquisition function trace as an orange line calculated from our learnt GP introduced previously in Figure 3.2b, in this case $\epsilon = 0.01$ and the function is maximised at $x = 1$ which would be selected as the next sample.

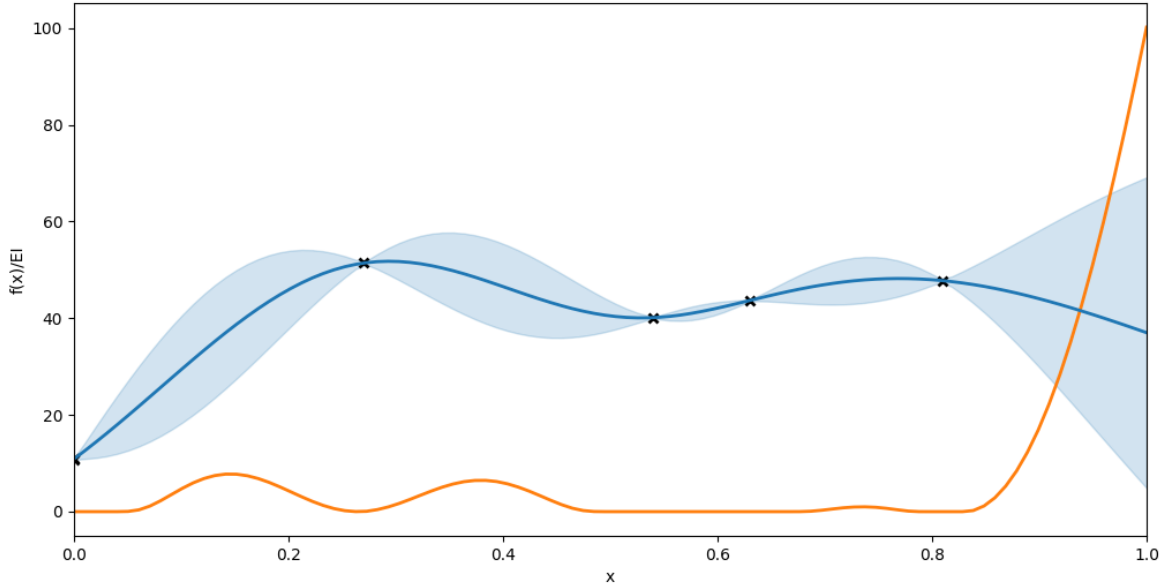


Figure 3.4: Calculation of Expected Improvement for Learnt $\mathcal{GP}$ parameters $\theta_{\mathcal{GP}}$: $l = 0.38$, $\sigma_f^2 = 1320$, $\sigma_n^2 = 0.02$, EI is maximised at $x = 1$

3.2.2 Loss Function

A loss function ($\mathcal{L}$) maps our parameter estimates to a scalar value ($\mathcal{L} : R^d \mapsto \mathcal{R}$) from function outputs against which Bayesian optimisation may be performed. In this framing of a loss function for calibration, the optimisation will look to minimise the loss. The search for a minimum Normalised Root Mean Square Error (NRMSE) provides a point estimate or single estimate of the calibration parameters.

$$\mathcal{L} = NRMSE = \frac{\sqrt{\frac{1}{n} \sum_{t=1}^n (y_t - \hat{y}_t)^2}}{\max_t(y_t) - \min_t(y_t)} \quad (3.15)$$

Such a loss function is standard with epidemiological model calibration [185] and is not an innovation but included for completeness as to which we'll demonstrate results of a Bayesian optimisation process applied to such a generic function.

3.3 Uncertainty quantification (UQ)

An accurate description of model output uncertainty, uncertainty quantification is key to ensuring that a model is useable [171], firstly for prediction and subsequently for the purposes of planning and control. We've previously introduced that epistemic uncertainty should be based on how uncertain the model is in its own predictions, while aleotoric uncertainty captures natural variations or noise in the stochastic process being modelled. Hopefully any epidemiological model based uncertainty quantification may accurately capture both types of uncertainty. Through the Bayesian optimisation steps (2) of calibrating a model, with the Gaussian process surrogate model fitting, we have gained a quantification of uncertainty based on the error function measure for input parameters, while using an NRMSE error returns a point estimate for the best-fit calibration result, equivalent to Maximum Likelihood Estimation (MLE). Such a point estimate may be sampled to quantify our appropriate measures of uncertainty in the outputs of consideration, namely the first and second central moments:

$$\mathbb{E}[y] = \int_{-\infty}^{\infty} y \rho(y) dy \quad (3.16)$$

$$\mathbb{V}[y] = \int_{-\infty}^{\infty} (y - \mathbb{E}[y])^2 \rho(y) dy \quad (3.17)$$

As we extensively work with Gaussian distributions in this work, it should be noted that a symmetric 2σ bound corresponds to our $\sim 95\%$ confidence interval for any arbitrary model output distribution $\rho(y)$ and the respective 95% bounds correspond to the integral:

$$0.975 = \int_{-\infty}^{upper} \rho(y) dy \quad (3.18)$$

$$0.025 = \int_{-\infty}^{lower} \rho(y) dy \quad (3.19)$$

$$95\% \text{ ci} = [lower, upper] \quad (3.20)$$

Having used this process we may now take the surrogate model to efficiently perform a Sensitivity Analysis (SA) on the chosen calibration parameters.

3.3.1 Sensitivity Analysis

Sensitivity Analysis (SA) describes how we may portion uncertainty in the output of a model to uncertainty in model inputs. We may leverage the surrogate model learnt through calibration for these purposes, especially as significant computations may have already been run to generate the training examples for the surrogate model. Polynomial chaos is another technique widely used in the field of Uncertainty quantification (UQ) [58], here due to its wide applicability we will continue to describe the use of a Gaussian process model surrogate, while the SA discussed is extensible to other surrogate modelling methods of choice.

The computation of Sobol indices may be performed directly from the Gaussian process surrogate. While analytic solutions exist [121] both using the predictor (Gaussian process mean) locally [125], and the global model covariance structure [137], these analytical solutions make modelling assumptions around the kernel composition, specifically to be of squared exponential form. While analytical results provide evidence for the tractability of solutions and the natural expression of uncertainty through the Gaussian process, for purposes of flexible extensions we'll rely on numerical methods. Numerical solutions may be agnostically applied to the surrogate model of an underlying computationally expensive epidemiological model, practically serving the purpose of this analysis pipeline while adding negligible additional computational cost.

The Sobol index is a variance-based measure of sensitivity. Sobol sensitivity indices quantify how much of the total variance of the model output ($\mathbb{V}[y]$) each uncertain parameter or input (θ_m^i) is responsible for [184]. Such that the first order Sobol index is calculated as:

$$S_i = \frac{\mathbb{V}[\mathbb{E}[y|\theta_m^i]]}{\mathbb{V}[y]} \quad (3.21)$$

and measures the effect of each parameter θ_m^i , where $\mathbb{E}[y|\theta_m^i]$ is the expected value of the output (y) when parameter θ_m^i is fixed and therefore an expected reduction in variance in the model output when we fix the model parameter.

$$\sum_{i=0}^n S_i \leq 1 \quad (3.22)$$

As above the sum of the first order Sobol indices is only equal to one when no interactions are present between parameters [83], meaning that the summation of higher order indices is zero. Higher order Sobol indices exist as the partial variances of sets of model parameters and may be interrogated further but for the purposes of ordering parameter importance, to reduce the dimensionality of our problem, first order indices provide all the information required. In neglecting the calculation of higher order indices we conveniently side-step a downstream problem in calculating higher order interactions which suffer from the curse of dimensionality as the number of model parameters increases [94].

A Quasi-Monte Carlo method is proposed for evaluating sensitivity, based on a sampled parameter set and offering an improvement upon standard Monte Carlo with increased coverage over the sampled parameter space, when compared with a true random sample [112]. In this instance Saltelli sampling from the parameter distributions generates model parameters using the Saltelli extension of the Sobol sequence [14].

$$S_i = \frac{U_i - \mathbb{E}^2[y]}{\mathbb{V}[y]} \quad (3.23)$$

$$U_i = \int_{-\infty}^{\infty} \mathbb{E}^2(y | \theta_m^i = \hat{\theta}_m^i) \rho(\hat{\theta}_m^i) d\hat{\theta}_m^i \quad (3.24)$$

In Equation 3.24, by fixing the parameter to a generic value ($\hat{\theta}_m^i$), its dependence may be eliminated by integrating over its PDF ($\rho(\hat{\theta}_m^i)$). Furthermore, estimates of $\mathbb{E}[y]$ may be generated from the samples eg.:

$$\mathbb{E}[y] \approx \frac{1}{N_S} \sum_{t=1}^{N_S} y_t \quad (3.25)$$

and a numerical estimate for U_i through Saltelli's numerical scheme [170], meaning that we complete our calculation for S_i via established numerical methods linked to the authors whose methods carry their names.

The first-order Sobol index is a variance based method of sensitivity analysis and looks at the contribution of each model parameter to the total model output uncertainty. As such an ordering of each parameter by Sobol index and subsequent rejection of parameters with low contribution may increase the efficiency of the calibration process through reducing down the size of the calibration parameter set.

Algorithm 3 may be iterated in a loop with the Bayesian optimisation process to improve optimisation results, such iteration will be termed a calibration schedule. A

Result: $S_i, \theta_m(\text{reduced})$
Input: $\mathcal{GP}$ surrogate of $\mathcal{L}$ function after n steps;
Generate: Saltelli samples equivalent to N_S samples [170];
Sample: $\mathcal{GP}$ at locations N_S ;
Compute: S_i [Equation 3.23 and Equation 3.24];
Return: S_i and $\theta_m(\text{ranked}) \rightarrow \theta_m(\text{reduced})$;

Algorithm 3: Sensitivity Analysis Steps

threshold value of first-order Sobol index value is chosen such that a ranked list of modelled parameters may be reduced.

3.3.2 Calibration Schedule

It should be noted that the steps outlined for calibration, uncertainty quantification and sensitivity analysis in this Chapter may be consumed as an iterative process. For example an initial calibration with a wide set of parameters may be practically narrowed by interrogating the parameter values which are most significant through a ranking of their first-order Sobol indices. Additionally any learnt parameter fit may be used to initialise further optimisation runs in calibration, for example to generate a parameter posterior distribution via Markov Chain Monte Carlo (MCMC) methods. Along with parameter distributions, sensitivity analysis and uncertainty quantification, all of the steps practically have an epidemiologists insights in the loop, while we can offer benefits in speed up for each of the steps. The validation and checking performed by a human in the loop is not replaceable - especially when the generated insights may be used to update the model structure. Though the automation proposed here is a greedy approach, sequentially reducing the parameter numbers and ranges, this in-sequence reduction could discard parameters which would have greater importance over narrower ranges. Therefore, the proposed methods are pragmatic and may return sub-optimal solutions, though nevertheless allow for the use of models without complete expert guidance.

3.4 Example: Calibration and Uncertainty Quantification of EMOD Malaria models

This section aims to give a complete illustration of the techniques employed for purposes of generating predictions given limited observations for a location. The dataset

used are Entomological Inoculation Rate (EIR) observations taken for Namawala district Tanzania [227] for a single year and are indicative of the most comprehensive sources of data for calibrating malaria disease transmission models. For this study we'll use two different model structures provided as examples for vector based disease transmission, both are mechanistic in design, one intended to model generic vector based disease transmission and the other developed specifically for the malaria infectious cycle. Importantly, both models have a one-to-one parameter mapping and while the structures vary, a reduced set has been chosen which is consistent for both models.

3.4.1 Calibration

The following parameters were selected for the calibration runs, both models taking the same demographic and climate base information for the location. The chosen model parameters (θ_m) and ranges consist of the following:

- Transmission Rate probability(β), prior uniform distribution, *range*: [0.5,0.99]
- Rain Scale Factor (c_{rsf}), prior uniform distribution, *range*: [0.5,1]
- Initial Prevalence ($P(0)$), prior uniform distribution, *range*: [0.1,0.2]
- Larval Habitat Multiplier (c_{larv}), prior uniform distribution, *range*: [0.5,1]

The calibration process may be visualised in Figures 3.5a and 3.5b, where sequences of parameter set evaluations generate a reduced loss (NRMSE) and are also projected into common areas of low loss in the surrogate model, using principal component analysis [222].

The learnt parameter sets on termination of the calibration run are summarised in table 3.1, the more complex malaria specific model achieving a marginally better fit to the observations EIR data.

Vector Model					Malaria Model				
β	c_{rsf}	$P(0)$	c_{larv}	E	β	c_{rsf}	$P(0)$	c_{larv}	E
0.88	0.50	0.10	0.50	0.197	0.87	0.56	0.18	0.75	0.152

Table 3.1: Best Parameter Sets for Respective Models: Namawala Tanzania.

The models which have been used are widely available malaria model descriptions based on the EMOD stochastic agent-based disease transmission kernel ², while the results associated with these parameter values may be shown against the observations in Figures 3.6b and 3.6a.

3.4.2 Uncertainty Quantification and Prediction

In Figure 3.7 we have generated samples for a 5 year prediction based on the calibrated models for the number of daily infections in the simulated population, for which a one dimensional GP has been learnt for the forecast under the current policy.

These GPs may now be combined through a covariance weighted prediction to generate the combined prediction of Figure 5.4, using techniques discussed later in Section 5.3.1.

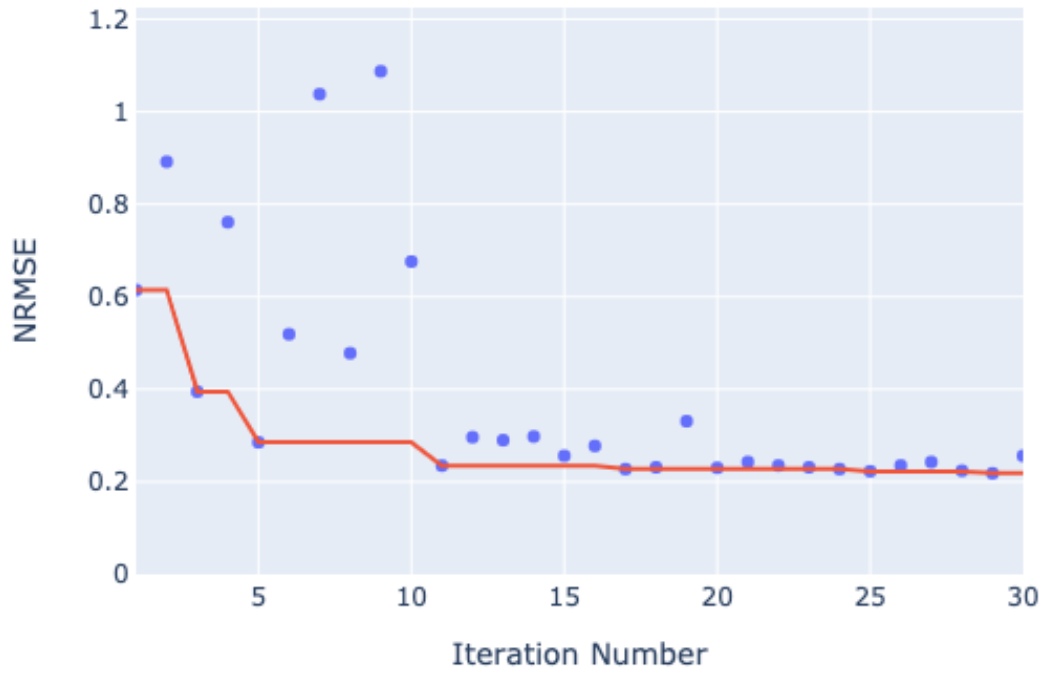
3.4.3 Sensitivity Analysis

Performing a global sensitivity analysis of the models used would allow for the number of parameters involved in calibration to be reduced. In this example it was achieved with a sensitivity analysis based on first-order Sobol index calculations on the calibration surrogate model (eg. a dimension $d = 4$ Gaussian Process learnt from the error function for calibration). A reduced parameter set will lead to improved performance of the calibration approach, in terms of both computational speed and final error (NRMSE). For the four parameter set example given in the table 3.2, it can be seen that model sensitivity is far greater for the c_{rsf} and c_{larv} parameters with regards to a best-fit of model uncertainty to observed uncertainty, placing a greater importance on climate-based factors for this particular model with little contribution to overall model uncertainty based on the parameters of β and $P(0)$. We observe high sensitivity to parameters which perturb climate-based inputs to this particular model family as to those more directly linked with disease transmission dynamics.

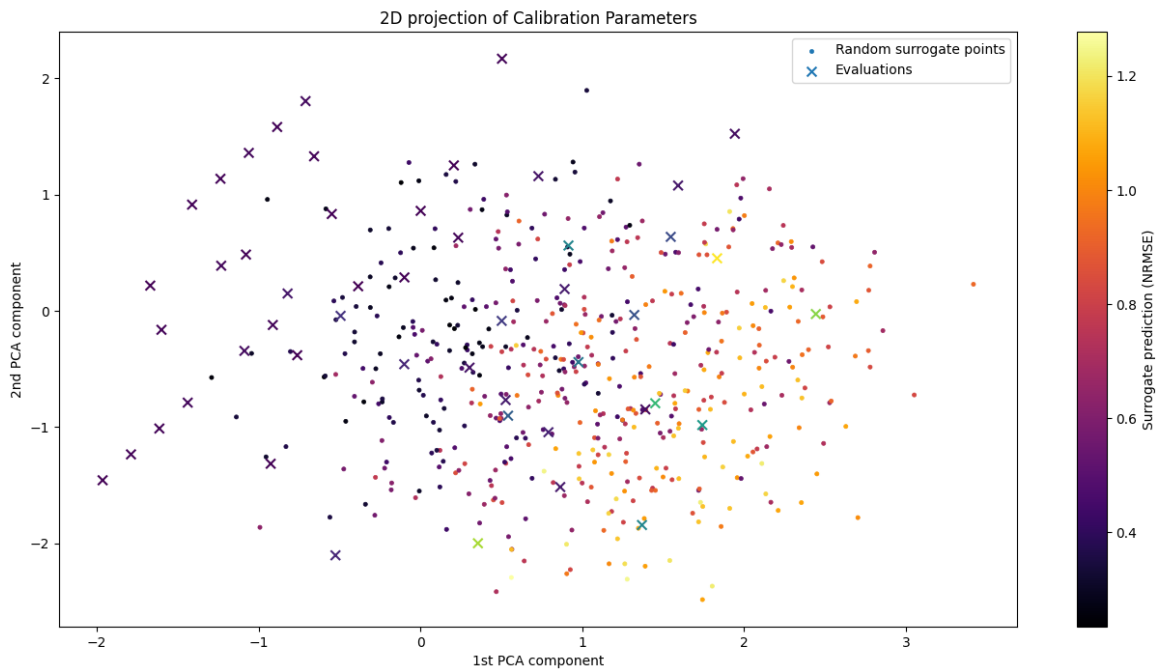
Vector Model				Malaria Model			
β	c_{rsf}	$P(0)$	c_{larv}	β	c_{rsf}	$P(0)$	c_{larv}
0.012	0.456	-0.001	0.445	0.020	0.380	-0.001	0.399

Table 3.2: First order Sobol indices for respective model parameters: Namawala Tanzania.

²<https://github.com/InstituteForDiseaseModeling/EMOD>

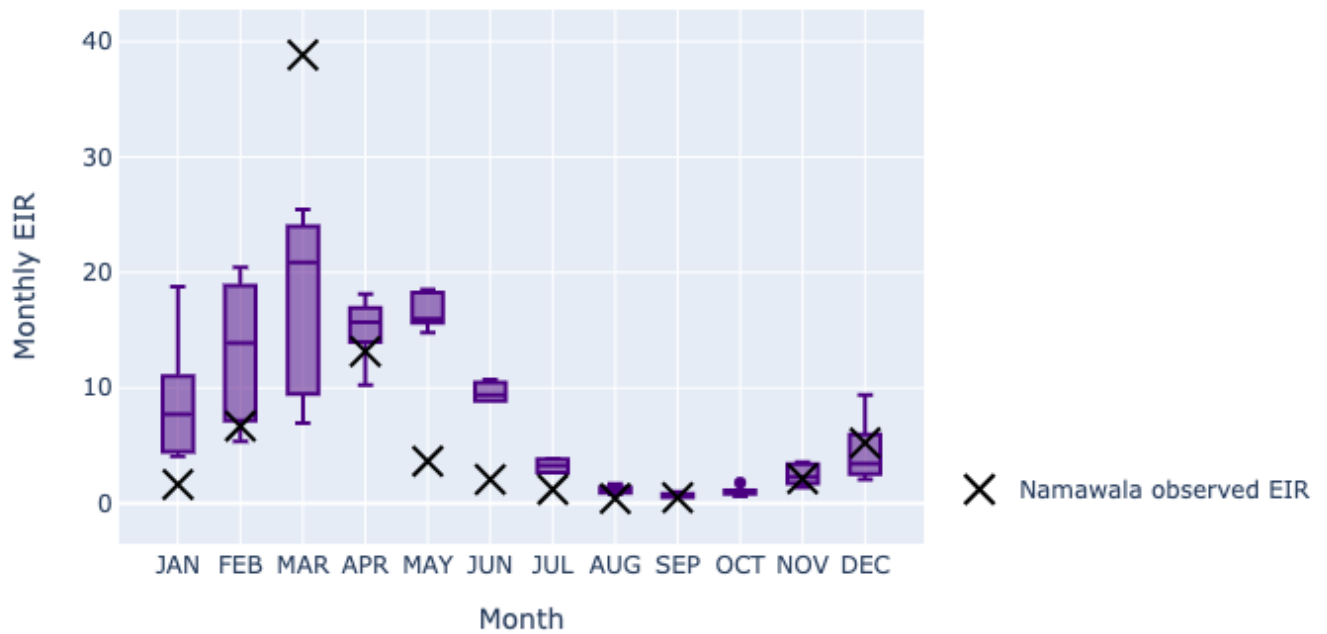


(a) Calibration improvements sequence using Bayesian Optimisation, for Namawala Tanzania. Blue dots: loss function score, Red line: minimum loss.

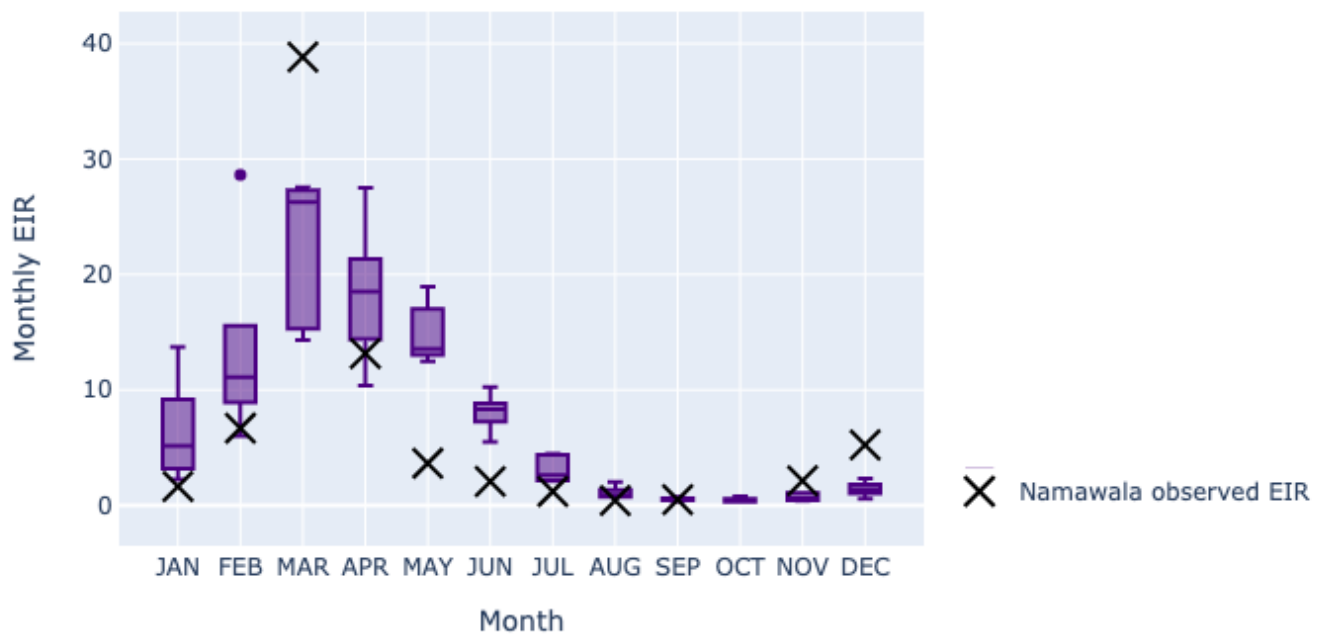


(b) Bayesian Optimisation Calibration Samples for Namawala Tanzania, projected onto 2 components by PCA

Figure 3.5: Visualisations of the samples generated across the parameter set during a short run-calibration

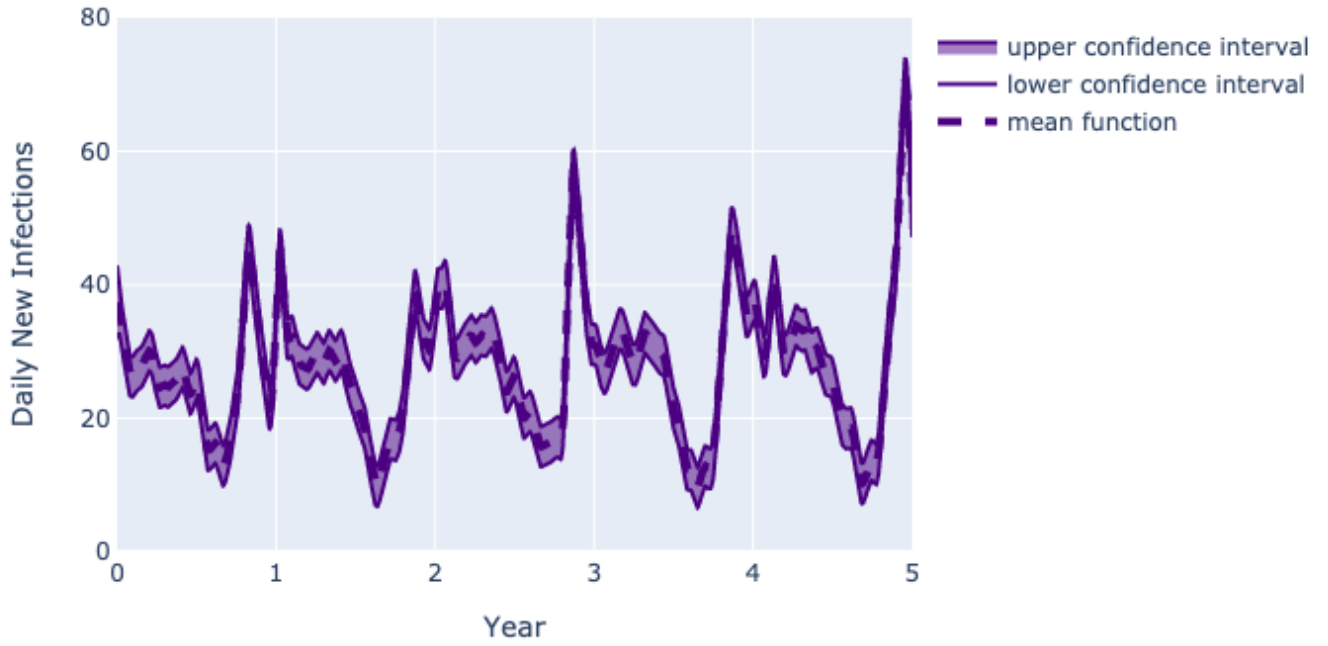


(a) Namawala Modelled Entomological Inoculation Rate (EIR) Vector Model

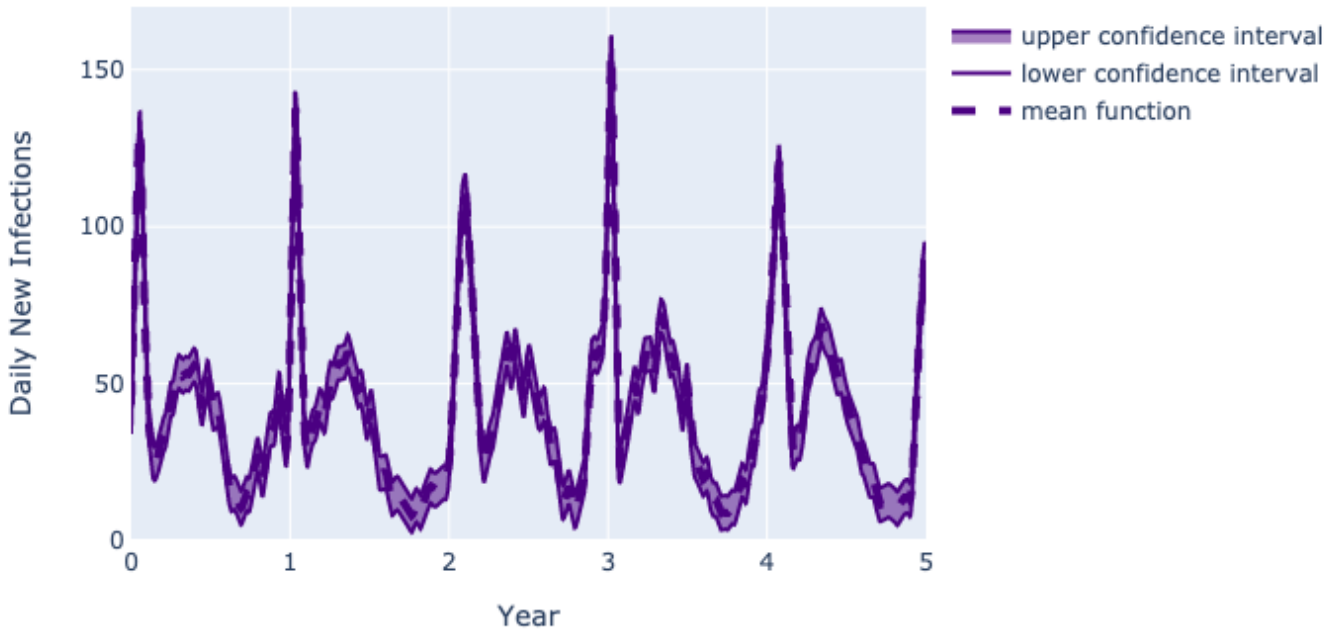


(b) Namawala Modelled Entomological Inoculation Rate (EIR) Malaria Model

Figure 3.6: Comparison of full and vector EMOD models, based on their generated monthly Entomological Inoculation Rate (EIR) distributions, against the provided observed points of monthly Entomological Inoculation Rate (EIR) in Namawala, Tanzania.



(a) 5 Year Predictions for Daily Infections for Location from Calibrated Vector Model, Learnt $\mathcal{GP}$ parameters $\theta_{\mathcal{GP}}$: $l = 0.10$, $\sigma_f^2 = 586$, $\sigma_n^2 = 36.6$



(b) 5 Year Predictions for Daily Infections for Location from Calibrated Malaria Model, Learnt $\mathcal{GP}$ parameters $\theta_{\mathcal{GP}}$: $l = 0.06$, $\sigma_f^2 = 1820$, $\sigma_n^2 = 69.1$

Figure 3.7: Comparison of full and vector models with learnt parameters

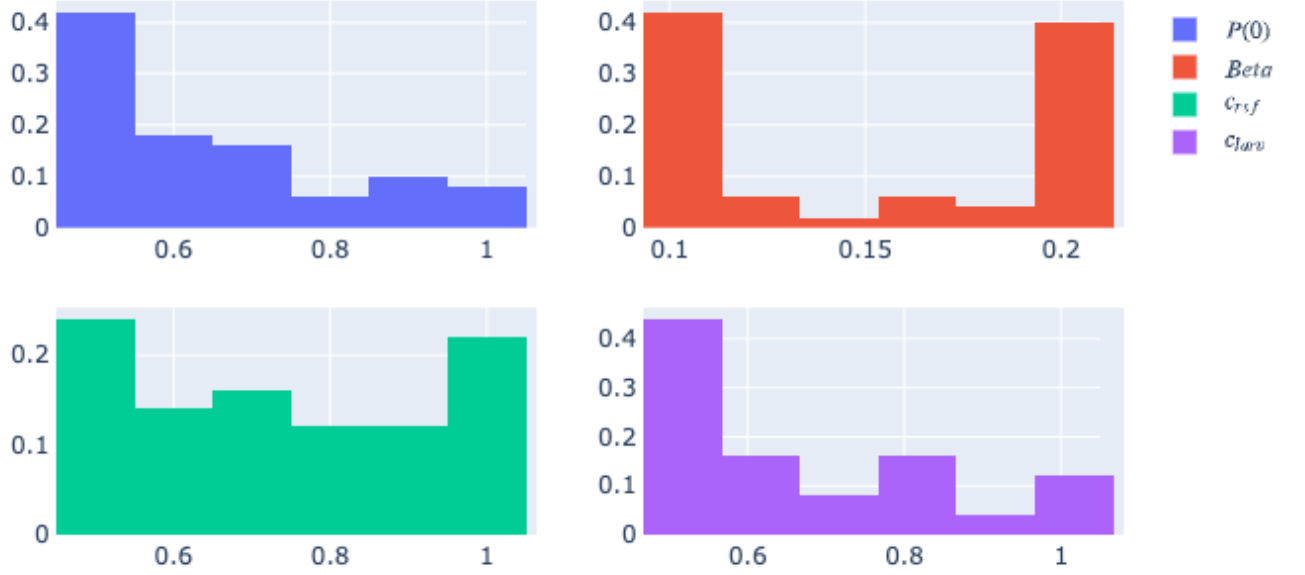


Figure 3.8: Histograms of parameter θ_m samples during calibration

Finally, the histogram of parameter samples during calibration shown in Figure 3.8 is included in the extension whereby the user wishes to infer parameter posterior distributions to the model. These histograms may be helpful in the formation of prior distributions and the variational inference [37] approach may then be initialised around the MLE parameter estimates provided by the Bayesian optimisation framing used here. Given the presentation of methods which limit the number of expensive epidemiological model evaluations required, we now move on in the next chapter to discuss a flexible infrastructure to support the required number of epidemiological model runs for developing useful predictions.

Chapter 4

Infrastructure and Computation for Prediction, Control and Planning from Epidemiological Models

If we suppose that many natural phenomena are in effect computations, the study of computer science can tell us about the kinds of natural phenomena that can occur.

Rudy Rucker, 2005

Computational epidemiology [85] brings the possibilities afforded by computational methods in simulating challenging physical systems [72] specifically to those which dictate disease spread, intervention strategies and epidemic preparedness. At the core of this work we have assumed a pre-developed epidemiological model, the practice of computational epidemiology may marry together the processes of both developing an epidemiological model and designing a computational infrastructure to support the computational requirements of running model simulations. A canonical example of this for the reader is the work of the Insitute of Disease Modelling (IDM), who have their own computational infrastructure (COMPS)¹ to support their own epidemiological model (EMOD) [35]. Instead we will focus on an infrastructure which may agnostically achieve the proposed tasks given any abstract epidemiological model format and the minimum required information for executing a model simulation.

¹<https://comps.idmod.org/>

To motivate the significance of the discussion of computation in congress with combating disease transmission, this year (2020) the Covid-19 pandemic received the singular support of the largest volunteer supercomputer on earth [111] as part of the Folding@home project [23], a close relative of the Search for Extra Terrestrial Intelligence (SETI) @home project [11], which itself stopped distributing tasks after close to 20 years of using distributed computing for this further sighted research effort².

More generically High Performance Compute (HPC) clusters, which aggregate their compute nodes using high bandwidth networking and support a high-degree of inter-process communication, are ubiquitous across scientific research [8]. This work presents results from single instances to multiple nodes but there is no reason that this could not be permanently housed in a HPC setting, simply by the application justifying the compute being continuously available if necessary. Therefore these methods may be developed and used on individual laptops, shared servers and all the way up the scale to shared HPC systems.

In Chapters 2 and 3 we have extensively discussed the design of algorithms to perform inference to limit the impacts of computational expense. We'll proceed to outline a design for a system to combine the insights provided by computational simulation with Machine learning approaches to move towards programmatic relevance of the work.

4.1 An Initial Computational Approach to Epidemiological Simulation

With regards to introducing online learning from simulation we have given the example of using an epidemiological model for decision making, which happened to be rather computationally expensive with regards to necessitating specific infrastructure for it to be run. This will serve as the motivation in assessing shortfalls and a general set of specifications to be met with a generic infrastructure. For this purpose we will quickly refresh the details from Section 2.4.2. In the original deployment of this infrastructure [32], the work packaged commit a50730b of the OpenMalaria repository³ with a suite of Python scripts. These Python scripts served both as the low-level wrappers of the OpenMalaria executable, and the implementation of Bandit based decision making algorithms. To run the necessary experiments an XML

²<https://setiathome.berkeley.edu/>

³<https://github.com/SwissTPH/openmalaria/>

formatted scenario file was acquired for the target region. This scenario file was dynamically modified by these Python scripts to enable Bandit based control of the evaluated experiments. Through the use of the *multiprocessing* package which supports the spawning of processes, the OpenMalaria executable could be run in parallel as batches on the available cores. The output generated by these executables was parsed by the same script housing the Bandit algorithm, and consequently used to derive the evaluation of the simulated malaria intervention plan. This model simulation environment was run on a 4 node cluster of machines, each with 64 hyper threaded cores (2.20GHz Intel Xeon® CPU E5-2660). On these processors, running one instance of an OpenMalaria simulation, for a representative human population size e.g., 100,000 people, returns results in a time-frame upwards to 24 hours. The final results of the experiments were stored as numpy arrays and saved to disk on each of the machines where experiments were performed.

4.1.1 Engineering and Deployment Challenges

Having successfully implemented and used such an approach to demonstrate the work, we were able to identify key issues with the existing approach, as outlined below.

- *Limitations in Scalability.* Scalability via Python’s multiprocessing means that experiments are limited in the pool of computers which could be used. Further, the number of computers available, the number of cores, and the amount of RAM were all fixed, so the computation environment did not support flexibly scaling up (or down) as demand would require. For each computer in the cluster, the entire codebase is independently maintained through a manual process to ensure that the appropriate versions of the requisite applications and code are present.
- *Separation Between Epidemiological Models and Algorithms (or Agents).* In the original deployment there is very little separation between the agent algorithm and the underlying epidemiological model used to run simulations.
- *Limited Simulation Flexibility.* There is very limited flexibility in the types of models which may be simulated, and the results of simulations are not stored in a form which permits them to be readily accessible to a broad audience (which could include decision-makers or model developers). Since in this case data and compute were partitioned on four machines in the cluster, synthesising the results generated for a single location (defining a set of model simulations in

this work) requires an extra, time-intensive step to synchronize data across all machines.

- *Duplication of Simulations.* It is likely that simulations are duplicated during learning, as there is no coordination between independent simulation environments prior to running an experiment through a model.

We'll now continue to detail a platform for disease prediction, control and planning, design and implementation that address the issues above and extend the original experiments to work with new capabilities that have supported the presented extensions in the following Chapters 5 and 6.

4.2 A Revised Infrastructure

This section provides an overview on the updated system architecture and deployment, as displayed in full in Figure 4.1.

4.2.1 Terms

To be precise in the language used we've laid out common system terms to be used in the proceeding sections:

- *Job:* A single model simulation.
- *Experiment:* A group of *jobs* with common properties, for example with same geographic location or user.
- *Intervention plan:* A list of intervention actions with their percentage coverages and deployment time.
- *Decision variables:* Disease control outcomes which we often related to rewards (Section 6.1.3) e.g. prevalence, Cost Per DALY Averted (CPDA), number of deaths averted etc.

Along with the current implementation supporting five different types of user, including:

- *Policy maker:* A person who is required to make decisions based on epidemiological model based evidence.

- *Expert modeller*: A user who makes and maintains the epidemiological models utilised for decision making.
- *Novice modeller*: A person who is not familiar with epidemiological model development, and is able to learn from interacting with deployed models.
- *Machine learning practitioner*: A person who creates agent based learning algorithms, for example as a competitor in Chapter 5 or the most likely group reading this work.
- *Admin*: A person who manages the whole technology stack presented.

4.2.2 Design Principles

To design what we will term the platform for disease prediction, control and planning, and address the challenges outlined, the following seven paradigms were leveraged.

4.2.2.1 Microservices

Microservices is an architectural style inspired by service-oriented computing that has recently started gaining popularity [65]. For our specification this provides a general design principle to breakdown the system into its component parts, then deployed as microservices, enabling flexible deployment and improved support for testing of interdependent components. The use of microservices also has the added benefit of permitting teams of developers to work more collaboratively, as they promote distinct segments of the codebase, exposing possibilities for readily sharing components of the work.

4.2.2.2 Frameworks

This involved the transfer of the original (Python) codebase to Java to take advantage of the Spring Boot framework⁴. The Spring Boot framework enables the reuse of existing, well-vetted design patterns for microservice deployment [213].

4.2.2.3 Containers

Containers are a lightweight virtualisation method for running multiple isolated Linux systems under a common host operating system and a new ecosystem has emerged around the Docker⁵ platform to enable container based computing [98]. Docker is not

⁴<https://spring.io/projects/spring-boot>

⁵<https://docker.io>

extensively used in the HPC community and we'll harness containers to deploy the developed microservices. Using tools like Docker makes it easy to deploy microservices at scale in heterogenous compute environments, achieving an approach which may be accessible, at whatever scale, for others to also replicate the work.

4.2.2.4 Cloud Computing

Due to its wide accessibility as a commodity we can harness cloud computing as Infrastructure as a Service (IaaS) or Software as a Service (SaaS) to deploy system components at scale, in the location or specific data center best suited for the target users.

4.2.2.5 Application Programming Interface (API)

An Application Programming Interface provides users with a library/module which exposes the microservices for their agent algorithms to interact directly with. At present, we have implemented a Python API which is deployed via github, and intended to be installed via the pip utility. This is extensively demonstrated with regards to the competition setting in Chapter 5.

4.2.2.6 Distributed Ledger Technology (DLT)

DLT deployed as a blockchain provides a decentralized and distributed platform that is useful in storing transactions aimed at improving trust, transparency and integrity among the participants in a fragmented ecosystem (e.g. disease modelling community). We can demonstrate the ability to store, share and maintain auditable logs and records of each step in the simulation process, showing how to validate results generated by computational workers [39]. This provides modellers with the ability to track who and how their models are accessed and used. If the compute is distributed, trust issues could arise on the results of a given job. The Hyperledger Fabric [13] allows for a platform to perform validation of these results.

4.2.2.7 User Interface

Consideration has been given for a front-end platform that allows sharing, calibration and running of models at scale, providing an access point for users who may not consider all of the lower level technical details of the microservices which make up this platform for disease prediction, control and planning.

4.2.3 Platform Components

Using the introduced design approaches, the platform was framed into the following ten component classes contained within Figure 4.1. Each of these platform components could be deployed onto distributed computers running on a network.

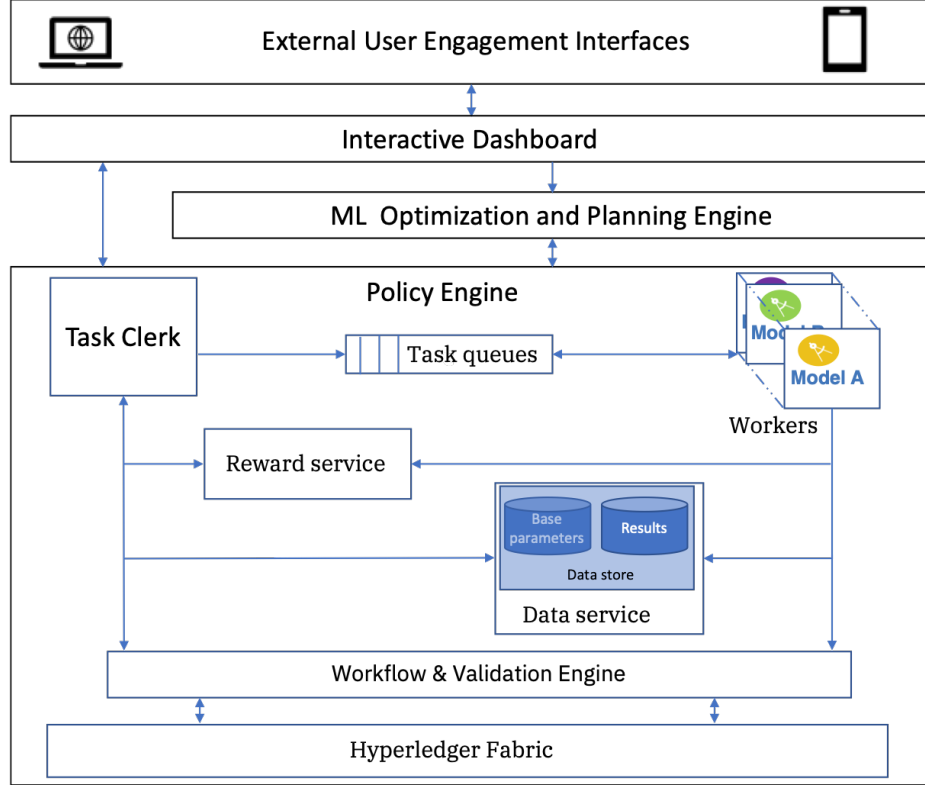


Figure 4.1: Conceptual architecture of a platform for Machine learning based prediction, control and planning from epidemiological models [210].

4.2.3.1 Workers

Using containers, we have packaged epidemiological models in a manner that is easy to deploy at scale and in multiple types of computing environments. In this container we also couple the model with software to communicate the desired input file, and to process the output files as needed. Together, these tools are referred to as a *worker*, and multiple workers can be deployed on a single machine or even distributed across the Internet. The output files resulting from the model execution are further processed by the *reward service* (Section 4.2.3.5). We have deployed multiple formats of disease models as workers at scale, a few of which constitute the examples in this work.

4.2.3.2 Task clerk

We permit users to define a specific instantiation of a model that they would like to evaluate. These parameters are used to “germinate” a seed input file for the model. The resulting file is then sent as a *job* to the worker so that it can be processed. Parameters define properties of an intervention plan, for example in our widely used base malaria example: the portion of the households where Insecticide Treated Net (ITN) and Indoor Residual Spraying (IRS) interventions are deployed, and the corresponding simulation time at which each of these interventions is applied.

4.2.3.3 Data Store

The seed input file, as well as the results for all evaluated intervention plans are stored in a central repository. Other data stored include the geographical information for a location based model, definitions of *decision variables* and log data. This data is also stored for subsequent reporting in the case that simulation of the same intervention plan is requested in the future.

4.2.3.4 Data Service

The *task clerk* and all the distributed *workers* access the *data store* via a common data service. The data service is currently implemented as a Java web-service with API endpoints that the clients use to access the *data store*. A database agnostic data-service is desirable and needs further development.

4.2.3.5 Reward Service

The output files resulting from model execution in the *workers* are sent to the *reward service* which computes relevant output predictions of the *decision variables* over a number of years. These variables include prevalence, the Cost Per DALY Averted (CPDA), deaths, deaths averted and the total cost of interventions.

4.2.3.6 ML Optimization and Planning Engine

This engine is currently implemented as a Python web-service that is exposed to agent based learning algorithms. The engine receives ranges of intervention coverages and deployment times from users and sends sampled intervention plans to the *task clerk* for processing. The engine receives *worker* output (predicted *decision variables*) through the *data service* and this informs the next intervention plans sample to be simulated.

4.2.3.7 Task Queue

To tie together the *task clerk* and the *workers*, we use a messaging fabric. The current implementation harnesses Advanced Message Queuing Protocol (AMQP⁶) as implemented in RabbitMQ⁷ to provide a task queue. The *task clerk* posts *jobs* to the queue which are subsequently picked up by idle *workers* of the appropriate type. When the work is complete, the results are saved in the *data store* via the *data service*. This instantiation permits *workers* to be deployed in a wide range of environments, with little requirements on coordination. Additional work would explore the use of workload manager frameworks like SLURM⁸ to more effectively schedule where and when the work would be done. In the rapidly expanding field of parallel processing, these job schedulers are the ‘operating systems’ of modern HPC systems, specifically they allocate computing resources and control the execution of processes on those resources [159].

4.2.3.8 Hyperledger Fabric (HLF)

The Hyperledger Fabric [13] blockchain component in Figure 4.1 is used due to the sensitivity of data and need of trust among participants (e.g. modellers, decision makers, users, etc) in the ecosystem. To realise this a permissioned blockchain HLF has been used, which supports private data collections for sensitive data with execution of business logic in the form of generic chaincodes (or so called smart contracts). In addition, HLF algorithms and protocols provide mechanisms to allow only identified participants in a business network to have consensus in all transactions while achieving data provenance, finality and immutability [13].

4.2.3.9 Workflow & Validation Engine (WVE)

The Workflow & Validation Engine component in Figure 4.1 is used to interact with workflows that track and monitor all user activities and their interactions with ML algorithms while using the infrastructure. This is realised by the use of workflows that are defined as generic chaincodes built on top of HLF [13], the chaincodes are further used to enforce interactions among modellers, workers and policy makers by specifying Access Control Logic (ACL). In addition, the chaincodes logic includes specific participants in the network encoded as actors for each *job* with defined set of actions and permissions, more description on workflow orchestration in [38]. The

⁶<https://www.amqp.org/>

⁷<https://www.rabbitmq.com/>

⁸<https://slurm.schedmd.com/>

actors are managed by HLF Member Service Providers (MSP) [13] which provides access to *jobs* metadata by using HLF secure authentication.

Leveraging on the chaincodes to control the design and execution of *jobs* provides a reliable mechanism to collect metadata from *jobs* and to generate content for transactions logs. The metadata is further used to validate *job* results and provide a means to trust computational results, more description on the trust and validation mechanisms in [39, 154]. To enforce data privacy for sensitive *jobs* between two or more participants in the network we leverage HLF channels [13] which provide secure private means of interactions among participants in the network.

4.2.3.10 Interactive Dashboard

To allow for easier interaction between the systems and the end users, an Angular 7⁹ based dashboard is presented to the users. The dashboard supports each of the user types as follows:

- (a) The *expert modeller* is able to upload models and decision variable definition repositories and associated input files for different locations. They are also able to view results of their models and comparisons with other models. Lastly they are able to define an action space (see Section 6.1.2) where interventions will be sampled by the *ML optimization and planning engine*.
- (b) The *admin* receives on-boarding requests of models and decision variables from the *expert modeller* and manually packages the resources required by the model in to a docker image for deployment in docker containers and exposes the decision variables as a web-service. The URL for the created services are updated using the dashboard.
- (c) The *Machine learning practitioner* uploads agent algorithms that sample the action space.
- (d) The *novice modeller* studies how the model output changes with varying intervention plans.
- (e) The *policy maker* views intervention plan simulations generated prior by the *expert modeller*, with the ability to vary different decision variable ranges so as to filter through these intervention plans to find one or more that meets their goals as shown in Figure 4.2. They can also define a set of intervention plans and

⁹<https://angular.io/>

investigate the results by running a single *job* and finally they are able to view historical interventions and outcomes data.

Figure 4.2 gives a view of the Interactive Dashboard which captures and presents model based results from the experiments. This dashboard, developed with some target users making national level malaria policy decisions, was intended to help a policy maker to gain intuition on model performance for proposed intervention plans. It also presents agent derived recommendations for relevant intervention plans for those locations.

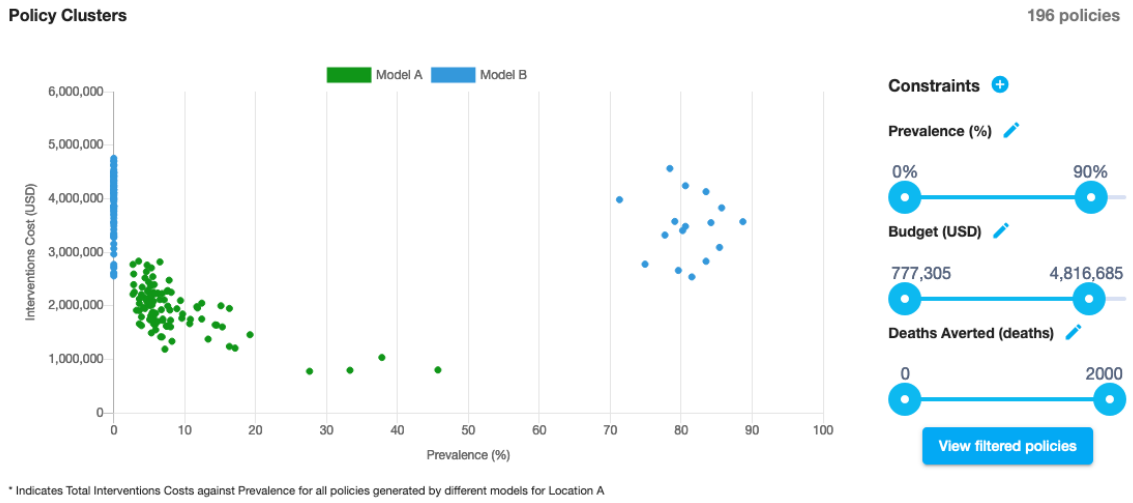


Figure 4.2: Dashboard view showing generated intervention plans from two different epidemiological models for the same location, with the ability to filter out intervention plans using prevalence, budget, and the number of deaths averted, in order to meet the strategic plan.

4.3 Performance Results

This platform permits workers to be added (or subtracted) on demand, thereby scaling the compute available to evaluate intervention plans. In the original work, scaling was supported by running additional experiments independently on additional computers. In this case, all resources are partitioned across the machines which precluded past results from being shared across experiments at run time, and introduced duplicity in experiments which were run. Finally idle compute resources on one machine were not available for use on another.

In our approach, the same compute could be pooled to support multiple experiments. Further, computational resources running on different networks could also

be pooled together. This was demonstrated with workers deployed in a Kubernetes cluster in the United Kingdom, HPC Virtual Machines in a data centre in Canada, and on a bare metal machine in the United States of America. A total of 200 workers spread across these networks were used to re-run the experiments developed in [32] as shown in Figure 4.3. This result of using Bandit approaches for one-step control, later deemed agents in Chapter 6, questions what is the most cost-efficient combination of two interventions (ITN and IRS). The presentation is of a response surface where better policies based on simulation results are dark red. The principle take-away being that as shown in [32], there are distinct recommendations provided by the independently implemented ML approaches in comparison with both the current and recommended policies.

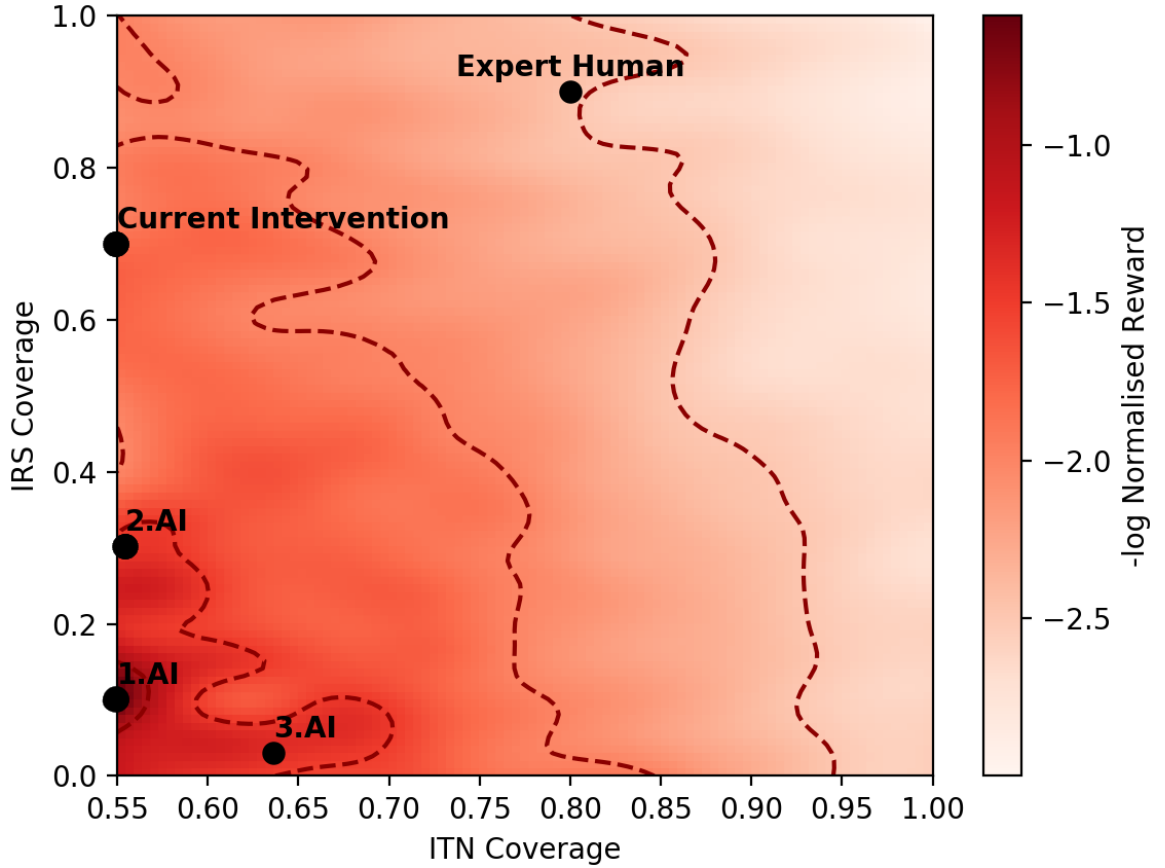


Figure 4.3: Combined results from [32] showing AI (optimal), Current, and Human Expert recommended solutions.

In Figure 4.4 we show results of running the workers on a single 40-core machine, configured to flexibly run an increasing number of workers based on the number of jobs submitted. We observe that the average job processing time gradually increases,

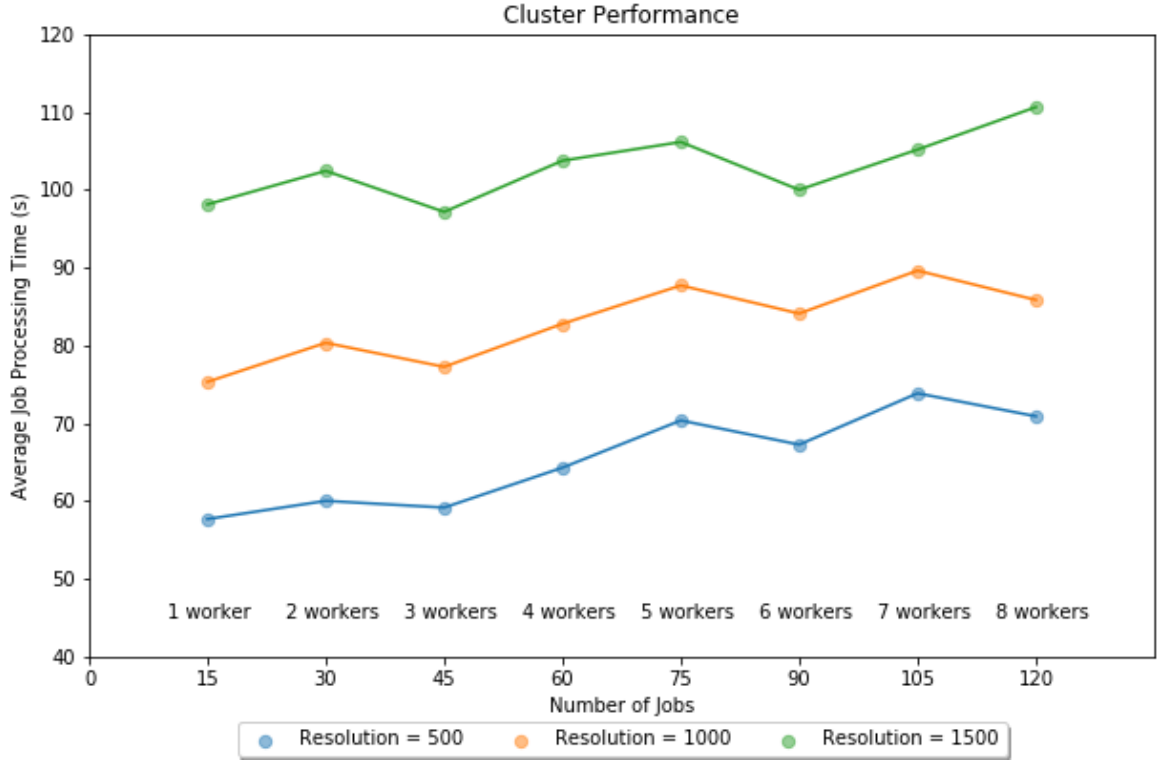


Figure 4.4: Performance of a single cluster, with increasing numbers of jobs and workers. Three different resolutions correspond to three levels of computational expense in the simulation.

with increasing job number while scaling the number of workers proportionately. The measured increase in average processing time for the infrastructure does not currently prohibit its utility for scaling to run greater numbers of experiments within the platform.

In addition to cluster performance, we also collected *Throughput* and *Latency* performance characteristics of the blockchain Workflow & Validation Engine (WVE) component. *Throughput* is the total number of transactions per second that are successfully processed by the blockchain network. *Latency* is the total time taken when a request is sent and confirmed in the network to the time when blockchain clients (e.g. task clerk or clients) receive a response. Figure 4.5 shows results collected using the Hyperledger Caliper benchmarking tool supported by the Linux foundation [95]. The blockchain network setup included four organizations, each with two peers and one communication channel with workflow and validation chaincodes installed. Figure 4.5 shows throughput increasing linearly to a saturation point of 100 transactions(tx)/second(sec) with a constant send rate increment of 15 tx/sec and varying block size. It is also observed that average latency increased steadily as the

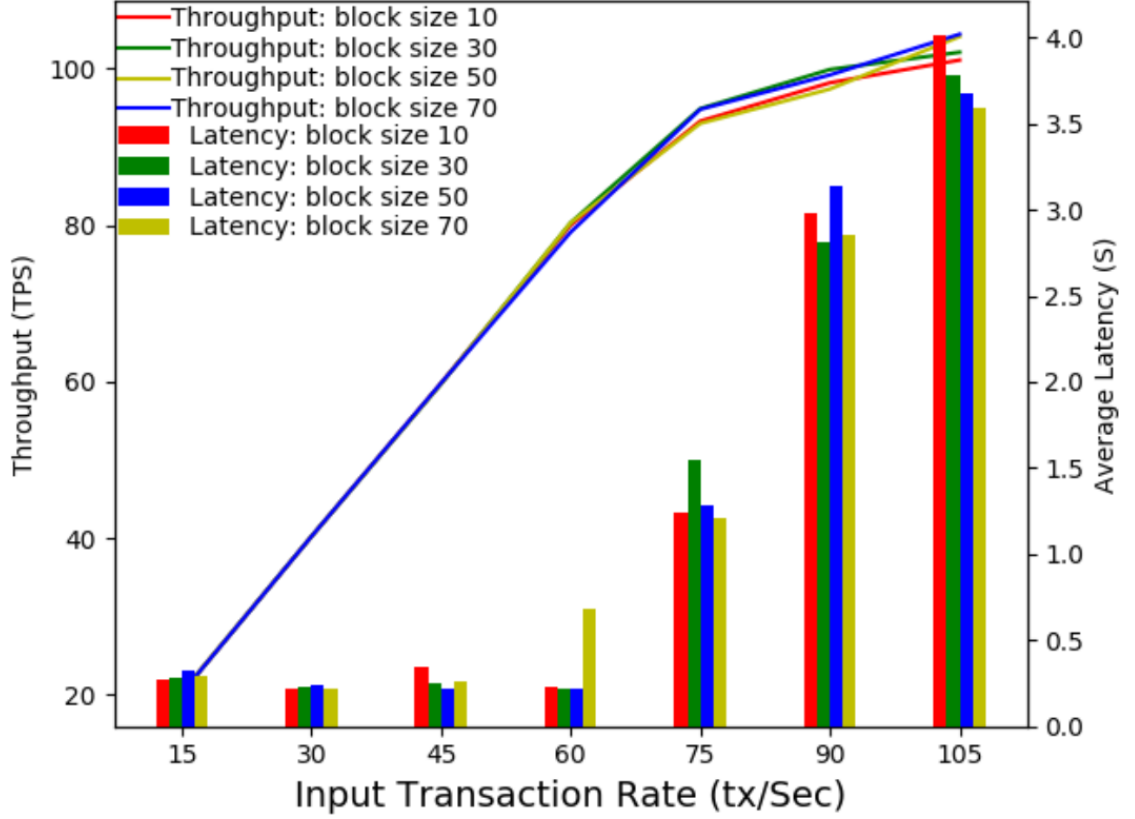


Figure 4.5: Blockchain throughput and average latency performance

send rate increased to approximately 4.0 seconds towards saturation point. Altering HLF block size from default 10 transactions to 70 transactions did not have significant impact on the throughput but increased latency as shown in Figure 4.5. The system's average throughput value of 100 tx/sec is lower than other HLF throughput and latency performance statistics, with results ranging from 1000 tx/sec to 3500 tx/sec [96,199,200]. This striking performance difference is due to the complex workflow use to track, monitor and validate experiments executions with fine grained access control implemented by multiple chaincodes. The work of [21,199] proposes different performance improvement metrics which include introduction of a membership service provider cache and bulk read/write during multi-version concurrency control validation, which were not implemented in the current version of the platform, and may be considered in future work.

Insights from experiments in both Figures 4.4 and 4.5 may be contrasted in the following manner; while simulation times are orders of magnitude greater than validation latencies via the HLF, the scaling of the infrastructure is more immediately

sensitive to the number of validations performed than the number of jobs run. There is a significant increase in latency after a validation transaction rate of greater than 60, with only marginal penalty for requesting a greater number of those jobs. Therefore, the current platform should perform validation requests in a manner that avoids this, limiting for what may be a negligible change in performance around the threshold. Conversely, increasing the total number of jobs on a single cluster does not incur significant penalty, with no obvious threshold to avoid. In scaling the current platform to greater numbers of jobs and strategies which minimize the number of jobs on each cluster may provide marginal benefits in total compute time, and should be considered further.

4.4 Example: Technical Approach for Competitions

This section will now outline the use of the design approach of the Application Programming Interface (API) to harness the results of epidemiological simulations at scale for the competitive outline of Chapter 5. Through pioneering projects like the OpenAI Gym [46], there is an established paradigm of how the Reinforcement learning community expects to interface their agent algorithms with simulation environments. In this work we implement the gym paradigm to provide a common interface for development and testing of agent algorithms to be submitted as solutions to the competition simulation environments.

Our formulation for these environments is novel in that it features a client-server architecture which supports sharing of resource intensive elements, as well as providing a mechanism to support transparency and trust [158], the elements of which have been discussed in Section 4.2.

4.4.1 Environment Web Service

To provide access to the environment in a scalable manner, a new client-server architecture was developed which hosts the environments as another model type which is exposed as a web service. The web service is written as a modular codebase, developed with the Python based microservice *flask* [87], and leverages the *blueprint* [145] module’s functionality to provide an extensive platform to concurrently host multiple environments. Each subdirectory contains the code and/or data required to implement the environment, with only a one-line stub in the base web service code to import that environment object into the common namespace. The surrogate model

is used as the description of the epidemiological model for the purposes of planning and easily shared with multiple participants, providing a single competitive reference environment, which is archived and may be returned to, while not overly abstracting the problem.

4.4.2 Competition API

The inference for competitor interaction with the competition environments took inspiration from OpenAI Gym, launched in 2016 [140], a tool-kit designed to create a unified API in which many Reinforcement learning benchmark problems could be implemented, so allowing for direct comparisons between different RL approaches. OpenAI Gym has itself drawn inspiration from earlier competitions and benchmarks, for example the Arcade Learning Environment [24]. Additionally, OpenAI Gym currently provides extensions to gym environments in areas such as control theory [127] [22] and simulated robotics [69]. Users may also implement their own gym environments for any arbitrary task via the unified Python-based gym API [163]. Although this service from OpenAI Gym was not used directly, the competition API mirrored its form to allow cross compatibility with existing Reinforcement learning code implementations that participants may have had access to. This version of the environments used are not fully ‘gym-compliant’ as they do not currently import the gym object, and accordingly do not harness gym-defined *spaces* [158].

The API assumes that an agent class has been implemented by the end-user to allow their chosen RL algorithm to interface with the competition environments. The primary tasks of the environment are to produce observations given the current state and to ingest actions from the agents to produce reward signals, mapping the required functions for an environment class to our underlying surrogate models in the following manner:

Listing 4.1: OpenAI gym-like environment

```
nextstate, reward, done, _ = env.evaluateAction(env_action)
```

- `env` - Instantiates an environment object with a limited number of evaluations (runs)
- `env.reset()` - Starts a new sequence or policy evaluation of the environment object

- observation (nextstate) - This returns the users next state based on the evaluated action
- reward - The reward given by the environment associated with the performed action
- done - A Boolean value denoting if the end of the episode has been reached
- info (-) - This remains unassigned in the API implementation

With the existence of significant work to create open RL benchmarks, this enabled competitors to leverage much published code in exploring solutions to the Malaria environments. The novel environment for epidemiological models was intended to allow data scientists and RL practitioners to expand the type of Machine learning approaches for policy learning and bring Reinforcement learning to other real-world applications, outside of games and simulated dynamics. The environment is coupled with a language specific API which implements a client library for participants to access methods which evaluate an intervention or a sequence of interventions (which we have referred to as a deterministic policy). These methods are analogous to the step method common in gym environments. A reward (a scalar value) is provided as a result for either the intervention or the policy. This reward value contains the signal, which enables a participant’s algorithm to learn the utility in performing certain interventions as they engage in the challenge activity.

The API is designed to integrate with an agent class implemented by the end-user allowing their chosen RL algorithm to interface with the competition environments. The principle functionality of an environment is to produce observations and representative transitions based on actions as input from external agents, along with providing a scalar reward signal to these agents and mapping the required functions for an environment class to our underlying surrogate models in the following manner:

The first API was implemented in Python, however leveraging the *reticulate* package, it has been used to support R language competition activities as well. In the future it’s envisioned to have implementations in multiple languages natively [158]. The web service architecture will make such libraries easy to implement and to support as HTTP functionality is pervasive in modern programming languages.

4.4.3 Example Code

Code for random policy generation and submission using the Competition API was shared for competitors to accelerate their developments and test submission of a solution. This gives an example of using the API to evaluate an entire policy in Listing 4.2.

Following the code example, a submission file may be generated using the competition library, by passing the CustomAgent class implemented by the competitor (eg. Listing 4.2) and generating a CSV file (example.csv) which is uploaded onto the site to generate a public leaderboard score, through the EvaluateChallengeSubmission class of Listing 4.3.

These examples therefore serve to demonstrate that along with other examples on the competition website [30] and documentation, a few lines of code written by the competitor within a *generate* method were sufficient to participate.

Listing 4.2: Example Random Agent Submission

```
from sys import exit, exc_info, argv
import numpy as np
import pandas as pd

from netsapi.challenge import *

class CustomAgent:
    def __init__(self, environment):
        self.environment = environment

    def generate(self):
        best_policy = None
        best_reward = -float('Inf')
        candidates = []
        try:
            # Agents should make use of 20 episodes in each training run,
            # if making sequential decisions
            for i in range(20):
                self.environment.reset()
                policy = {}
                for j in range(5): #episode length
                    policy[str(j+1)]=[random.random(),random.random()]
                candidates.append(policy)

            rewards = self.environment.evaluatePolicy(candidates)
            best_policy = candidates[np.argmax(rewards)]
            best_reward = rewards[np.argmax(rewards)]
```

```

except (KeyboardInterrupt, SystemExit):
    print(exc_info())

return best_policy, best_reward

```

Listing 4.3: Submission file generation using the competition library

```

EvaluateChallengeSubmission(ChallengeSeqDecEnvironment, CustomAgent,
    "example.csv") #How scoring files generated by participants

```

4.5 Related Work

There are a variety of methods and platforms that provide scalable compute resources and infrastructures which can be applied to support decision making from epidemiological models.

Deep Learning as a Service (DLaaS) [36] is a unified platform that provides developers, data scientists/engineers, with a mechanism to use popular deep learning libraries (e.g., Caffe, Torch and TensorFlow) in one-go, with minimal effort, helping to minimise unnecessary effort to configure the infrastructures needed for deep learning based computations. To do so, DLaaS leverages the benefits of cloud service and API economies, enabling the training of deep learning models using one or more deep learning libraries with user specified amounts of resources or budget, as well as providing additional utilities for the user like control of the training progress.

Like DLaaS, MLaaS is another architecture for a machine learning as a service type of model. It provides compute infrastructure and Machine learning libraries to create flexible and scalable models [161]. MLaaS can support multiple data sources and create a variety of models using various algorithms, parameters and training sets via web-based APIs. However, MLaaS's architecture is currently designed for specific analytical libraries (limited to predictive modelling), making it difficult to extend it for our use in our infrastructure.

A similar concept has been discussed in other numerous and varied approaches in the expanded literature e.g. [20, 51, 139]. PredictionIO [51], for example, is the first open standard architecture that combines multiple Machine learning processes into a distributed and scalable system where each Machine learning process can be accessed via APIs and a dedicated GUI as well, including integration with a variety of big data platforms (e.g. in Hadoop and Spark). PredictionIO also provides a template gallery, allowing users to reuse templates for various Machine learning applications.

Another open-source platform that compiles popular statistical analysis libraries of R is OpenCPU [139]. However, OpenCPU lacks key features such as flexibility, ease-of-use, and economy of scale, etc. and hence it is difficult (or requires expertise) to configure (including acquisition of the requisite data), execute, and interpret data, models and results.

Following this detailed overview of an infrastructure and systems approach for prediction and control from epidemiological models, the next chapter will consider the results of exposing the development of ML approaches in optimisation and planning to open competition.

Chapter 5

Machine Learning for Epidemiological Planning via Competition

Only dead mathematics can be taught where the attitude of competition prevails: living mathematics must always be a communal possession.

Mary Everest Boole, 1931

Competition is a theme tightly woven into the advancement and demonstration of Artificial Intelligence. Significant efforts have been taken by organisations to demonstrate publicly above-human performance machine intelligence in games. Notable examples include Deep Blue (Chess) [49], Alpha GO (Go) [179] and Alpha Zero (Chess, Shogi, Go) [180]. In addition grand challenges have been posed as open competitions (often with significant funding or investment) such as the Netflix prize (\$1M) [28] and the DARPA Grand Challenges (\$Multi M) [201]. These efforts all raise awareness and stimulate immediate and further research about their respective topics.

There is also an emerging class of problems promoted through dedicated Machine Learning Platforms such as Kaggle ¹ and CodaLab ², inviting engagement from a global community of Data Scientists to engage in problems that matter [211]. Machine learning challenges are not always focussed on algorithmic advancements or theoretical contributions. They tend to generate solutions of a more practical nature, specific to a proposed problem, rather than a generalisable result, with the aim of the challenge to insight the imagination of others to pose novel solutions to the problem.

¹<https://www.kaggle.com/>

²<https://competitions.codalab.org/>

Of greatest relevance to our objectives in epidemiological control, we look to the subset of competitions which are based on Reinforcement learning (RL). The first RL challenges were launched in 2006 [217] out of a necessity to create benchmarks and common environments to support results, sharing and publication. This use of the competitive framework has harboured a competitive spirit in demonstration and publication of research in RL, especially in dominant topics such as Deep Reinforcement learning, where the Arcade Environment [24] for Atari 2600 games is a common benchmark in Deep RL, in much the same way that the benchmark data set Imagenet [62] shaped Deep learning research in Computer vision.

A set of problems may have arisen in the RL domain, from a focus on reporting benchmark performance, but no standardised evaluation or reporting framework [105] has accompanied such benchmarks and this is potentially damaging for research, leading to concerns regarding the reproducibility, reusability, and robustness of Reinforcement learning algorithms or approaches [92].

Direct comparisons between different RL algorithms are inherently difficult as the authors of algorithms may spend a lot of time tuning them to work well in specific domains and overfit to environments [216], even when they are (by design) supposed to learn in an active manner from zero experience. What is required is a way to perform unbiased comparisons across various learning tasks, as was proposed early on in Reinforcement learning competitions. Unbiased comparisons for static data sets are a well-established practice in other areas of data science but the interactive nature of RL means that no fixed data sets or experience are possible. However, more standard notions of Machine learning evaluation (for static data sets), based on a train-test split [90], may still be applied to entire learning environments, whereby a competitor develops an algorithm or learning agent on a limited set of representative training environments, and then reports learning performance on another set of similar, but non identical testing environments. This principle is adopted for many RL competitions, often over multiple generalised domains. Each domain corresponds to a different type of problem. Considering, for example the well-used example of helicopter control [2], training and testing are performed on specific domain instances of the problem, such as helicopters with different physical characteristics. The competition format can then provide an unbiased comparison between algorithms, that truly tests their ability to learn [64].

Our motivation is hence not just to tackle a new class of problem through RL, but to frame the entire problem in a competition. This provides the benefit of unbiased

evaluation of new approaches to the epidemiological domain, which may provide novel answers.

The rest of this chapter will report on the mechanics of executing a global series of Reinforcement learning competitions, motivated by the sequential decision-making task for optimal malaria control, abstracted into a set of reusable environments through the techniques for learning a model developed in Chapter 6. The competition environment code-base presented here is open source, along with accompanying educational material [158] and the infrastructure required to run the competitions at scale is discussed in Chapter 4.

5.1 Planning from Learnt Models

Planning is defined formally as policy improvement based on agent interaction with a model of the environment [192]. This is especially important as we may not be learning by interacting in real-time with the ‘real’ environment for challenges in epidemiological decision making. The process of planning necessitating that plans should be developed in-silico, based on model evidence, before implementation. We propose that a competitive framework may be used to perform the task of planning in a distributed manner, each competitor developing an agent to interact with a model of the ‘real’ environment for the epidemiological decision making task. A learnt surrogate model of underlying epidemiological models is what is shared with competitors. The reasons that a learnt surrogate model is developed for sharing are due to the following:

- Lowering the computational cost associated with directly running epidemiological models (Chapter 6)
- Abstraction allows for ease of sharing without requiring expert knowledge to run a model (Chapter 4)
- Quantification of uncertainty may be consistently applied (Chapter 3)
- Combination of insights from multiple epidemiological models may be generated (Chapter 5)

There are several examples of competitions and leading AI research achievements where it is thought access to the latest computational hardware [4], along with highly motivated specialised teams [174], led to the most successful contributions to challenging problems [5]. Although all global challenges will necessarily not have a substantial

monetary budget or team of experienced researchers tackling the problem. Through sharing an abstracted model with limited numbers of evaluations and requiring little computational resource may lower the barriers to contribution and increase the numbers of researchers able to work on challenging problems [135] [6]. Our goal is to generate examples of distributed policy learning from epidemiological environments, not generating one single strong learner or team, but achieving strength through heterogeneity and lowering the barriers to access or contribution.

As in Figure 5.1 an integration of planning into the life-cycle of learning allows us to generate a union with the objective of control, in addition to the control policies learned through direct RL methods. This life-cycle of learning was initially posed in the Dyna architecture [193], an integrated approach to learning, planning and reacting. High performing algorithms or agents to the planning task may also learn from further experience or direct interaction with epidemiological models, substituting planning based learning to performing direct RL and consequently generate further experience. In this way it can be seen that the process of learning may be entirely deployed in an iterative manner for further improvements, based on the principle of sharing epidemiological models as environments, along with the straightforward deployment of learning agents which algorithmically look to solve these environments, either directly or via model-based approximations. We'll now proceed to further detail the instantiations of model-based planning based on the set-up of a global series of Machine learning competitions.

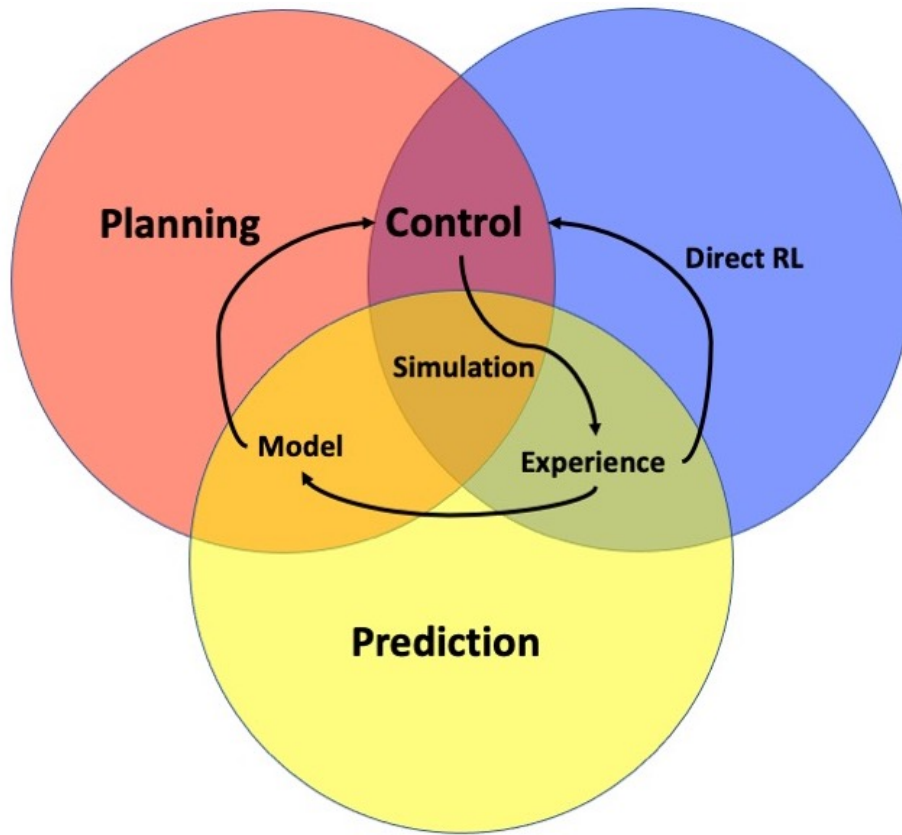


Figure 5.1: Inspired by the work Planning, Learning and Acting [194]. *Blue*: Chapters 2 and 6 refer to learning a value function and policy based on simulated experience from an epidemiological model. *Yellow*: Chapters 3 and 6 refer to the task of prediction and necessarily learning a model based on epidemiological simulations and experience. *Red*: this Chapter (5) discusses the task of planning from a model to learn a policy which in turn may be used for further simulations to generate more experience.

5.2 KDD Cup 2019 and Other Competitions

The KDD Cup is a renowned Machine learning competition which began in 1997 [168] and has produced several notable examples of benchmark data sets and Machine learning challenges [197] [28] [66] [109]. This work presented here was accepted as the KDD Cup—Humanity RL track competition 2019 [30]. Over a four month period the competition attracted 248 teams, 296 competitors and a total 735 submissions, along with sponsorship from the Association for Computing Machinery (ACM) for prize money totalling \$25k. The competition was advertised to competitors as follows:

“ We are looking for participants to apply machine learning tools to determine novel solutions which could impact malaria policy in Sub Saharan Africa. Specifically, how should combinations of interventions which control the transmission, prevalence and health outcomes of malaria infection, be distributed in a simulated human population....”

This competition was uniquely framed as a Reinforcement learning problem, the first in the history of the KDD Cup since its inception in 1997 [101]. Participants were expected to submit high performing solutions to the sequential decision making task shown in Figure 5.2. For this competition, actions (defined as coverages of Insecticide Treated Net (ITN) and Indoor Residual Spraying (IRS) interventions as in Section 2.4.1) receive stochastic and delayed rewards, which are resource constrained by both the monetary and computational costs associated with implementing and observing the impact of an action. The environments used by the competitors were developed as sequential surrogate models as outlined in Chapter 6. Due to the open nature of the competition, submissions were encouraged from participants who may not have had previous experience in RL problems, with supplementary material, tutorials and Q&A sessions provided and administered through the competition platform ³.

This activity was executed as a three-phase, four-month long activity, featuring multiple distinct environments as well as a single-blind submission. During the first two phases, the environments were used for both training and evaluation on a shared leaderboard, while in the final proving phase, submissions were evaluated on an environment unseen by the participants. The reward utilised for the activity was the measure of cost effectiveness of the considered intervention policies in terms of Cost Per DALY Averted (CPDA).

³https://compete.hexagon-ml.com/practice/rl_competition/37/

Action

$A_S = [a_{ITN}, a_{IRS}]$ where $a_{ITN} \in [0, 1]$ and $a_{IRS} \in [0, 1]$

Reward

$R_\pi \in (-\infty, \infty)$

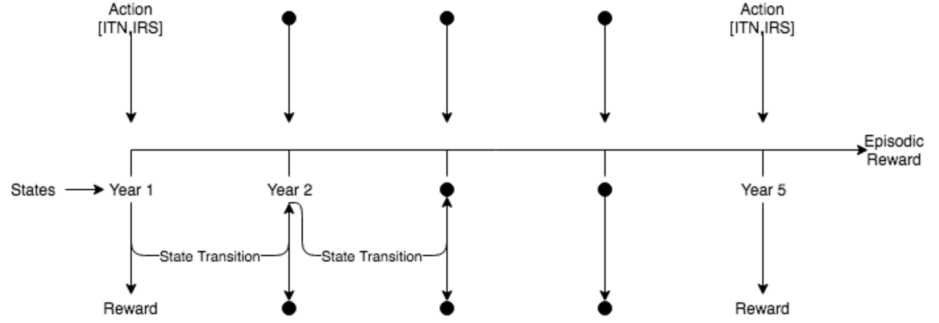


Figure 5.2: Sequential Decision Making Task - Competition Framing

We now proceed to provide further detail into the competitive process and its evaluation with regards to submission of algorithmic approaches or agents towards our proposition of distributed planning for the task of epidemiological decision making.

5.2.1 Evaluation Framework

In distinction to previous Reinforcement learning (RL) competitions [64] and [217] which served as guidance for the work, a new evaluation framework was proposed for the KDD Cup competition [230], which required that final evaluation was performed on a ‘held-out’ environment to which competitors did not have access. For this purpose it was necessary to engineer the infrastructure and framework to accept code submissions on unseen environments requiring:

1. That competitors adhere to specific code submission requirements and,
2. There was an automated and secure infrastructure to execute these submissions and report on results.

The evaluation framework was applied to three phases of competition which will be detailed before discussing further submission and scoring.

5.2.1.1 Feedback Phase - Test Environment

The feedback phase was used to provide the chance for competitors to learn, read the ‘Getting started code’ examples, follow tutorials, blogs and engage in the discussion

forum, principally consisting of running the example code given previously in Chapter 4 specifically Listing 4.4.3 and generating a submission file to post a public leader-board score. After a period of a month over which approaches were iterated, improved and refined, a selection was then made such that the top scoring 100 submissions on the public leader-board were able to proceed to the next phase (for details on scoring see Section 5.2.2). Those competitors with a top 100 score, were required to make a valid code submission to proceed to the following check phase. The upper bound of 100 submissions also provided a reasonable limit to the instantiations of code which would have to be run, along with providing a check that the formatting of the submissions was clear to competitors before their final evaluation submission. The test environment used was in fact an analytic function, based on the Bird function [126], taken from the optimisation literature. This multi-modal function can mimic some of the dynamics observed in epidemiological simulation models but be run efficiently at scale on a new web-service for all of the competitors. The additional practical reason for this was that further testing and validations were being run on the sequential surrogate models used in latter stages. In order to generate a sequence from the Bird function and conserve a notion of resource efficiency in the reward signal, this took the functional form of Listing 5.1 (where $x \rightarrow a_{ITN}$, $y \rightarrow a_{IRS}$ and $ff(x, y) \rightarrow R$).

Listing 5.1: Sequential Bird Function for Test Environment

```
def ff(x,y):
    x = np.clip(x,0,1)
    y = np.clip(y,0,1)
    x *= -10
    y *= -10
    return - random.uniform(.9,1.1)*(sin(y)*exp((1-cos(x))**2) +
        cos(x)*exp((1-sin(y))**2) + (x-y)**2)

def ffprime(x,y,oldx,oldy):
    return ff(x*(1-oldx), y*(1-oldy))
```

5.2.1.2 Check Phase - Prove Environment

The top 100 participants ranking on the public leader-board from the feedback phase were taken forward to have access to the proving environment. With no public leader-board in the check phase, a private dashboard logs a competitor's progress, so that approaches may be iterated upon, while the results of this are not shared with other competitors. Unlike the feedback phase, competitors lose all visibility of what other competitors are doing. To progress to the final verification phase competitors had

to submit their final code and dependencies to be run in the verification phase. The prove environment was generated based on samples from a low-resolution mechanistic malaria model. This produced the appropriate sequential simulation surrogate model structure from thousands of simulations selected, based on Monte Carlo Tree Search runs from the epidemiological model, specifically using the method of GP-UCB-MCTS (outlined in Section 6.2.4). To achieve generality in their approach each competitor was expected to develop approaches with good performance in both the test and prove environments, before having their submissions ranked on the final unseen evaluation environment.

5.2.1.3 Verification Phase - Evaluation Environment

Final submissions for the verification phase were run, validated and verified by the organisers on an evaluation environment. The evaluation environment was modelled based on high-resolution observations from the same mechanistic malaria model scenario as used for the check phase, additionally a hidden environment was used for further blind validation of the submissions. The hidden environment was a combination based on combining all samples from both the prove and evaluation environments and relearning a new sequential surrogate model. The top ten participants of the competition were honoured with prizes, and emphasis was placed by the organisers on fair evaluation, including provision for detection of any anomalous behaviour in submissions, which would fall outside the rules of the competition. A final ranking and leader-board of all the submitted solutions was openly posted on the competition platform. Figure 5.3b gives the cumulative normalised scores of the top ten submissions over all the three stated environments (including the fourth hidden environment).

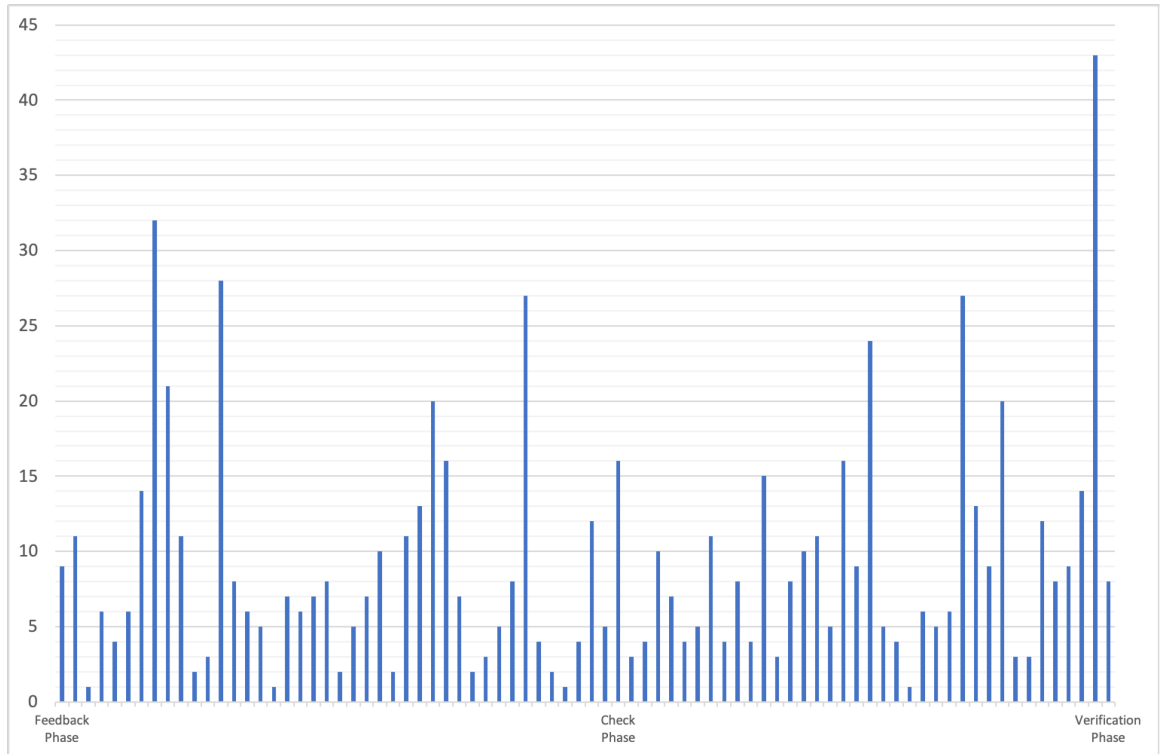
5.2.2 Submission and Scoring

We now proceed to provide more details on the submission and scoring of each phase. Each submission was evaluated based on ten runs, with each run consisting of ten episodes, and the learnt greedy policy over a run was submitted along with its episodic reward. This is indicative of the limited experience for any algorithm to learn a high performing policy from direct query of the underlying simulation models, although participants had access to a computationally less expensive surrogate model description of the underlying epidemiological models. This framing of the problem is novel compared to existing RL benchmarks, which do not typically penalise the number of evaluations in reporting the performance of an approach. There is no sharing of

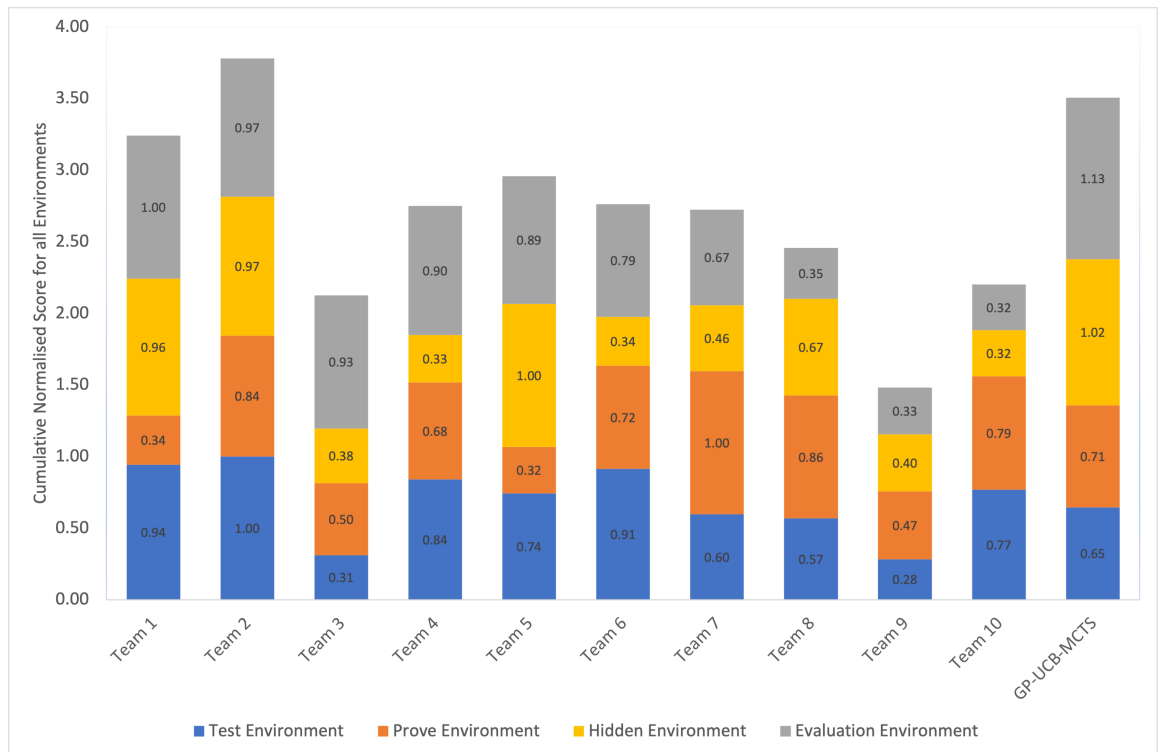
information allowed between runs, so every submission learns from zero initial experience during each run. In the live challenge format (Section 5.2.3) only a single environment was used, and participants' scores were determined from Comma Separated Value (CSV) files generated by their algorithms. In the multi-phase format, participants submitted the same formatted CSV file except for in the last phase.

In the last phase of the multi-phase format, participants submitted a compressed file containing their algorithm code for an independent assessment on remote machines. Code submissions were limited to a list of standard Python libraries which could be pre-installed on the infrastructure, limiting not only the requirement of the machines to be connected to the internet for package installs but also any malicious behaviour which may be hidden in the submissions. If submitted code could not be run, the team was contacted. If minor remediation or sufficient information was not provided in order to run the code, the submission was ultimately removed, though in the 2019 KDD Cup competition this was only the case for a single submission. The reported score was taken as the median of the episodic rewards from the ten greedy policies learnt over the ten runs.

Figure 5.3a shows the distribution of CSV file submissions over the competition duration, while Figure 5.3b gives insights into the relative performance of the top ten teams over the three stages, ultimately it is not clear that single solutions were the best performing for all environments, though in Section 5.2.3 we interrogate this further with regards the effects of tuning an agent submission to a single environment.



(a) Number of daily CSV file submissions up until the submission of code for verification and final scoring.



(b) Cumulative normalised scores of top 10 teams on all competition environments (additionally included for comparison the relative performance of algorithm GP-UCB-MCTS detailed in Section 6.2.4).

Figure 5.3: KDD Cup analysis

5.2.3 Other Competitions

In 2019, culminating with the NeurIPS conference [31], a series of similarly themed events were performed at conferences with different audiences. In all cases competitions have been framed in the same manner, where the participants have been asked to write an agent algorithm which interacts with the challenge environment in fixed number of samples, learns and recommends a policy (the sequence of interventions) identified by their solution as best. Each intervention has its own behaviour and parametrisation in the model as they act on individual simulated humans in the environment.

In the framing, interventions are defined as the coverage of Insecticide Treated Net (ITN) and distribution of Indoor Residual Spraying (IRS) at given times in a five year period. The state of the environment changes as a result of the interventions which have been performed, as well as the dynamics which have been implemented. A state signal from the underlying environment is presented as a variable from which the algorithm may learn a policy. Accordingly, the rewards derived from interventions were learned as a function of the interventions selected, and the time in which they were selected (See Figure 5.2). The participant’s code was given a fixed compute budget of 2000 samples from the environment which could be used for learning. The problem was framed at a level we assert is accessible to the community. With several possible approaches to the problem, its framing will remain deliberately non-prescriptive. Also, due to our API abstraction for tackling the problem in a machine learning guise, little domain knowledge is required of malaria transmission and control, or its simulation (which was left to the competitor’s interests).

Because of the selection of the client-server approach to access the environment and subsequently accessing the remotely hosted environments, it is not considered a distinct advantage to have considerable local compute to run the models in a competitor’s considered algorithm. Such model complexity would be discouraged by design as it is unlikely a competitor can generate enough data in the competition time frame to train such a model, with each submission taking hours to run. As all models are learning from surrogates of physics-based simulators developed by epidemiological modellers, the results found should be interpretable by the malaria modelling community. The complexity of a particular strategy should also be confined to real-world feasibility, which is ensured through constraints on timing, cost and number of actions which may be performed. All of which for simplicity were contained in the evaluation metrics, to generate a reward signal.

To date, the malaria scientific community has no clear answer for an optimal strategy in a particular location, which we believe may efficiently be found through computationally intelligent solutions, and challenges like ours. As such, submitted algorithms may act as a black-box in terms of application to epidemiological decision-making problems, with regards to this framing, algorithmic results which outperform other strategies in simulation may be ranked and presented. In continuation from the challenge, these results may be subsequently validated by epidemiologists, with a long term focus on implementation of the results.

The data used for these competitions is stationary based on the state observation with stochastic returns. Also, due to the high dimensionality of combinations of actions consisting of an intervention strategy, the possible set of solutions can not be exhaustively searched through the platform, so there is little concern for over-fitting the possible data in the environment. Specifically, competitors will have to grapple with the paradigm of exploration and exploitation of high performing strategies under limited resources in terms of both compute and time. We hope that given the stochastic multi-dimensional nature of the problem, this will discourage over-tuning of any particular model for a simulator due to the noisy results returned.

5.2.3.1 Indaba Hackathon

The Deep Learning Indaba Conference [29] was used as a chance to engage directly with 40 teams over the course of a conference in Nairobi, Kenya. Competitors used solely the evaluation environment (Section 5.2.1.3) to post submissions to a public leaderboard published around the conference venue. This live competition setting had different incentives as the reward for participation was principally knowledge based, the problem framing being used as an instructional environment for Reinforcement learning. The top three submissions at the end of the conference were honoured in the closing ceremony receiving certificates and small prizes. It is clear from Table 5.1 that submissions obtained a far higher performance through direct tuning on the previously hidden evaluation environment. Daily lab sessions in excess of two hours interacting with the teams suggested that techniques tended to focus on tuned improvements on Q-learning and Policy Gradient based approaches as an Introduction to Reinforcement learning [29].

5.2.3.2 NeurIPS

This particular competition did not gain the same number of submissions, organised as a live competition it used the same concept and is included for completeness. The

event differed from the other two competitions in a lack of funding and incentives for competitors such as prizes and it serves as an example that for substantial competitive submissions, additional considerations should be made to ensure attention in a busy conference schedule over a one week period. Essentially this was not an event which could compete for attendees time at the venue and it was reportedly difficult to attract attention over the course of the event. The organiser was unable to travel to the venue, running operations remotely and this did not alleviate some of the challenges in gaining submissions. The work could still be presented at the conference competitions workshop but with four RL competitions competing for participants’ time, all had a lack of engagement compared to what was expected (typically each attracting less than 10 submissions), this work for comparison registering 8 teams, 6 competitors and 2 submissions [31]. Framed as a live, multi-day activity, this challenge format was executed as a single phase event with data from a mechanistic malaria model used to provide the input loaded by the web service. Synthetic data was used to bootstrap the malaria model for this activity and the model output was an approximation of the problem domain based on epidemiological model simulations. The reward signals in this activity were the parasite prevalence, and a measure of cost effectiveness of the considered intervention policies (CPDA).

Position	Top 10 Scores 2019 Competitions		
	KDD Cup	Indaba Hackathon	NeurIPS
1	27.513	44.997	23.754
2	26.572	43.575	-
3	25.581	42.436	-
4	24.791	38.344	-
5	24.552	36.564	-
6	21.686	33.579	-
7	18.369	33.232	-
8	9.756	31.760	-
9	8.987	30.411	-
10	8.808	30.223	-
last (position)	-2.548(30)	10.486(17)	23.754(1)

Table 5.1: Relative scores of top ten submissions for the same environment when agent code is submitted for evaluation on an unseen environment (KDD Cup) and generated CSV files are uploaded by competitors for a live environment (Indaba Hackathon and NeurIPS)

5.3 Collaboration from Competition

At the start of this chapter we acknowledged that competition inherently serves to increase the pace of research and innovation, given the ground work we’ve laid in a demonstration of how competition may be harnessed towards achieving this for planning from epidemiological models. We’ll now include an extension to the family of approaches within ML which maximise algorithmic performance via collaboration, sharing or more commonly the ensemble. Ensemble methods improve the predictive performance of a single model by training multiple models and combining their predictions [169], this combination will be considered as a weighted vote learnt by the ensemble method [63]. Through a principled combination of competitive submissions, we now also classify shared epidemiological models as a competitive submission, in the spirit of the received competitive agent algorithms which we’ve so far focussed on in this chapter.

5.3.1 Learning Surrogates from Multiple Epidemiological Models

A benefit of learning surrogate model descriptions of heterogeneous, complex and computationally expensive epidemiological models is that we may generate principled combinations of multiple models for each disease and location. Such an ensemble model should combine all information from the respective sub-models so as to improve our uncertainty quantification estimates from the combined model predictions. The combined prediction is learnt based on modelling a combined value-function from the separate value-functions learnt across models. The reason to generate this value-function across models is not just to provide a description of a single future, forecast or prediction, but to bring the benefits of epidemiological based modelling to the tasks of control and planning, where the predictions are conditioned on the action taken.

There are several methods in Machine learning which consider boosting or combining ML models. Adaptive mixtures of local experts [97] is a popular example and one of the first, though we will not draw any further attention to it here as we select to train models on the same data sets, instead of switching between surrogate predictions we focus on combining them. This has been achieved in gating of networks [155] by assigning probabilities to different experts (epidemiological models) based on the inputs (which we consider as actions). We consider combining models using non-constant weighting functions [205], and look at linearly weighted combination of models in which the weighting functions are dependent on the actions (and states

for the sequential surrogates to be developed in Chapter 6). Weighting functions can be derived either by evaluating the action dependent variance of each model or by estimating how likely it is that a given model has seen data in the region of the input space close to the input pattern. The latter solution is closely related to the mixture of experts approach and we will not utilise this method due to our own framing of the learning tasks we describe, whereby all models have access to the same training data or environments, which is supported by our common infrastructural approach to support sharing models, *data* and compute. In the case that additional epidemiological models are added, weighting functions can easily be modified without any change to the existing models, which have been pre-trained/calibrated on the same data. Furthermore, it is possible to incorporate algorithms or agents (derived from environments) using ensemble methods if we have access to estimates of the competence of each agent with regards to the learnt policies. We now focus on a single method to perform the model ensemble, which uniquely, does not rely on a model's own measure of uncertainty but compares the level of congruence or agreement with another models; a covariance-based weighted committee [144].

5.3.2 Covariance-based Weighted Committee

We consider the committee vote of multiple learnt surrogate models. These models may be Gaussian processes or any other surrogate Machine learning model whereby the analysis to inform the surrogate model committee is performed in a black-box manner based on actions and predictions. We calculate the covariance from samples, as we have no prior information about the joint distributions of the surrogates. Sampling the surrogate model is far more efficient than directly sampling the epidemiological model as discussed in Chapter 3. We also assume that our surrogates are unbiased models, the linear weighting of each model is denoted $\delta_i > 0$, such that the committee estimate $\hat{C}$, is a linear combination of the vector of input ($\mathbf{x}$) dependant weights $\boldsymbol{\delta}(\mathbf{x})$ and the corresponding surrogate model outputs $f(\mathbf{x})$:

$$\hat{C}(\mathbf{x}) = \boldsymbol{\delta}^T(\mathbf{x})f(\mathbf{x}) \quad (5.1)$$

Simplifying notation, removing explicit input dependence (i.e. $\boldsymbol{\delta}(\mathbf{x}) \rightarrow \boldsymbol{\delta}$), the committee variance (V_C) is approximated through:

$$V_C = \boldsymbol{\delta}^T \boldsymbol{\Sigma} \boldsymbol{\delta} \quad (5.2)$$

Where Σ is our covariance matrix over the surrogate model errors at each input ($\mathbf{x}$), assuming zero bias in the predictions the δ weighting is learnt through:

$$\delta = \frac{\Sigma^{-1}\mathbf{u}}{\mathbf{u}^T \Sigma^{-1}\mathbf{u}} \quad (5.3)$$

Here $\mathbf{u}$ is a vector of ones. In order to impose the constraint that all $\delta_i \geq 0$ and ensure the ensemble weights sum to unity, δ_i are transformed via the softmax function:

$$\sigma(\delta_i) = \frac{e^{\delta_i}}{\sum_{j=1}^K e^{\delta_j}} \quad (5.4)$$

This provides a weighting set which sum to one and reside in the interval $[0,1]$. An example of the weighting of two surrogate predictions for malaria infections (from Figure 3.7) is given below in Figure 5.4.

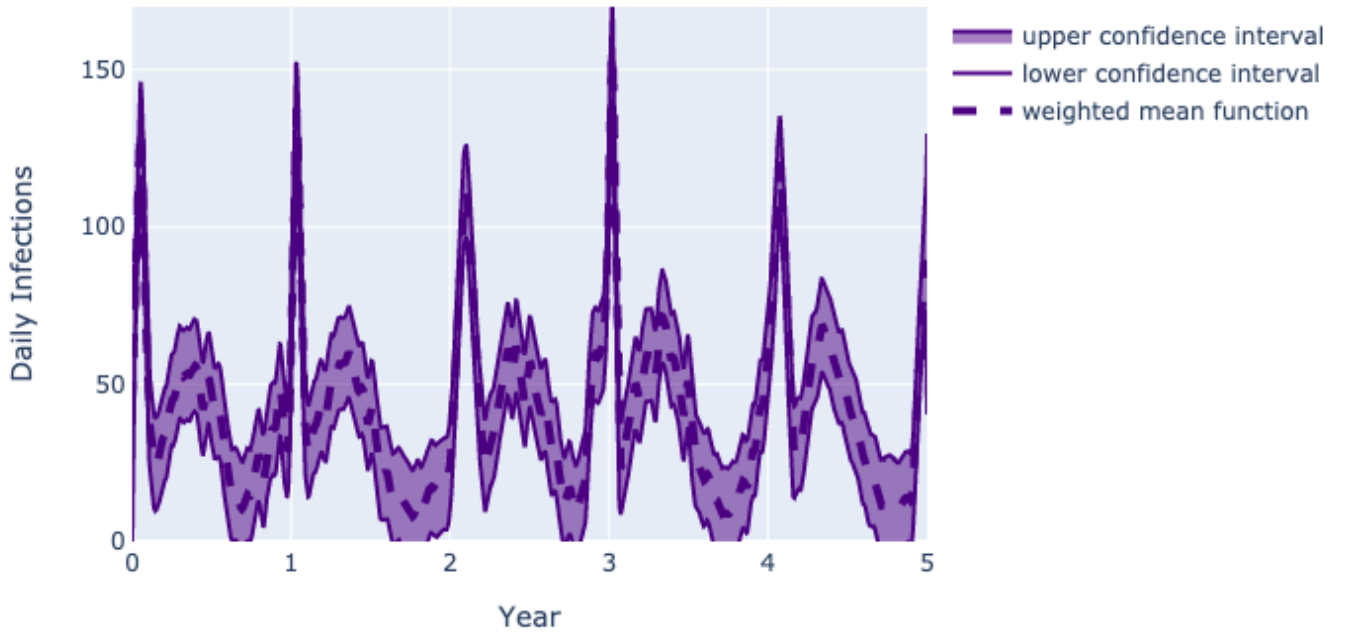


Figure 5.4: Two model covariance weighted prediction of daily malaria infections in Namawala, Tanzania.

The separate predictions which generate the two model covariance weighted prediction were presented in Section 3.4.2, which had presented the individual calibrated predictions of EMOD ⁴ Vector and Malaria models for Namawala Tanzania.

⁴<https://github.com/InstituteForDiseaseModeling/EMOD>

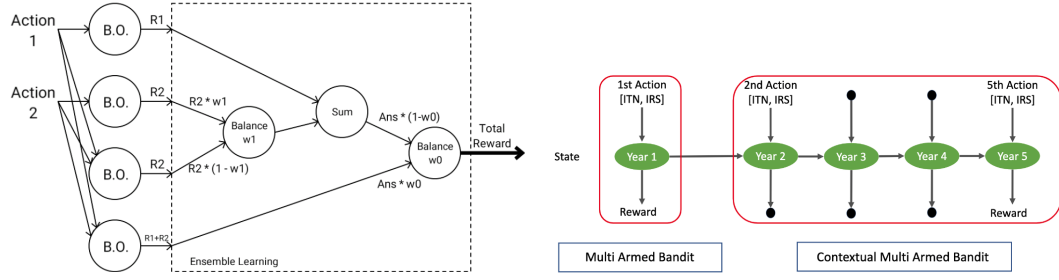
5.3.3 Consideration of Other Ensemble Methods

A variance based weighting would reduce our covariance matrix Σ to a diagonal variance matrix and our combined estimate to the Maximum Likelihood Estimation (MLE) or an information weighted mean. This may capture some of the concepts introduced above, though we necessarily assume model independence. If models are trained/calibrated on the same data, this may not be a reasonable assumption. It does however provide a ready mechanism for ensemble estimates especially given direct computations on the covariance functions of our Gaussian process surrogate models, although naturally weighting to models which predict lower uncertainties (or with lower learnt covariance functions). Different models may provide information in different regions, for example a gating across interventions could be used, whereby epidemiological models may not generate outputs across all interventions or actions.

5.4 Example: Analysis of Successful Competition Approaches

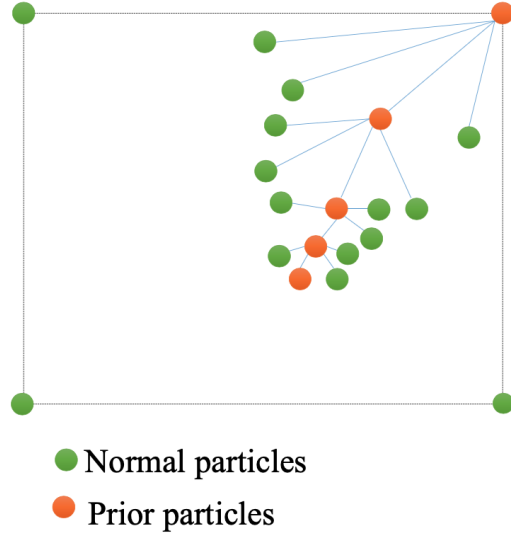
In this section we provide an overview of the top approaches taken by competitors, these were influenced by material shared to competitors, which focussed on the techniques of Q-learning (Section 6.3.1.1), and Policy gradients (Section 6.3.1.2) which we are yet to discuss and also a Genetic algorithm (Section 2.3.5.2) method which we have already covered, all as examples using the competition environments to generate submissions for the stages described in Section 5.2.1.

Algorithmic innovations looked to increase the sample efficiency of the approaches and improve performance within the limited number of episodes for the competitions. The main way this was achieved was by mirroring the work used to generate the environments (unknown to the competitors) through using model estimations centred on Gaussian process surrogates and Bayesian optimisation to maximise episode reward based on policies learnt during the training runs from the provided environment. While such approaches would introduce additional approximation error, based on approximating an approximation, under the problem framing these approaches proved to be the most successful model-based learning approximations.

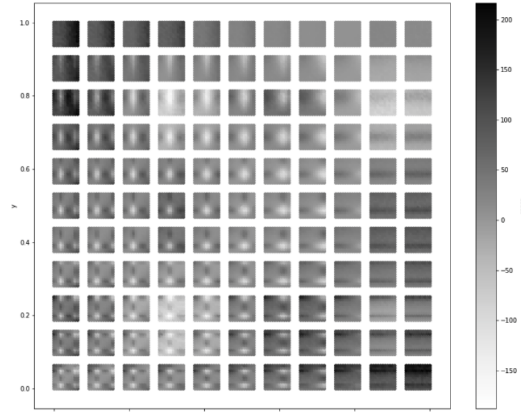


(a) Bayesian optimisation with a forward boosting network [135]

(b) Thompson sampling bandit method



(c) Fast search with prior particles (FSPP)



(d) Gridded random search

Figure 5.5: Sample of successful approaches for the malaria competition sequential decision making task

The repositories are included for Figures 5.5a⁵ 5.5b⁶ 5.5c⁷ and 5.5d⁸, each contains the agent code from which their approaches were evaluated and a workshop poster for the KDD cup 2019 workshop [230].

⁵https://github.com/bach1292/KDD_Cup_2019_LOLS_Team

⁶<https://github.com/quanjunc/KDD-CUP-RL-8th-solution>

⁷<https://github.com/federerjiang/KDD2019-Humanity-RL-Track->

⁸<https://github.com/luosuiqian/submission>

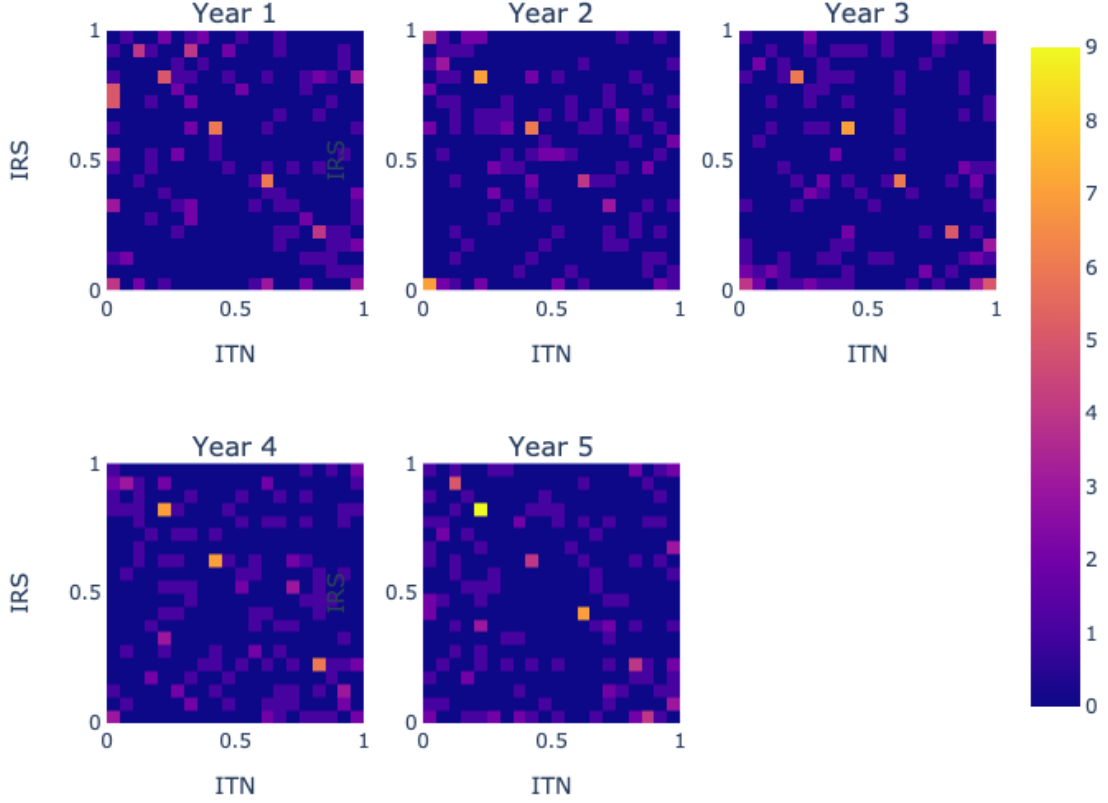


Figure 5.6: Two dimensional histograms for the yearly actions generated by competitor agent submissions during the verification phase (Section 5.2.1.3)

We now consider the results of submitted approaches for the verification phase. In Figure 5.6 we see consensus in actions chosen for each year based on the agent samples. High frequency actions tended to balance the components of ITN and IRS interventions with reasonable frequencies of selections of ‘bang-bang’ policies based on one of the components of the actions being either on-or-off e.g. $[0, 1]$, $[1, 0]$ or $[0, 0]$ in $[ITN, IRS]$, curiously this is an optimal control strategy for infectious disease transmission with Non-Pharmaceutical Intervention (NPI) under linear costs (Section 2.3.4.2) [116]. This visual description combines information to generate a unique set of insights, that policies for malaria control based on the used epidemiological models have a clear pattern in algorithmically searched solutions, qualitatively suggesting that cost-efficient strategies should inversely weight ITN deployments to IRS coverages, with greater initial IRS coverages in this particular model environment.

5.5 Conclusions on Competitions

In the health domain, competitions such as Learning to Run [106] and AI for Prosthetics [107] competitions created by the Stanford Neuromuscular Biomechanics Lab promote the creation of a controller, that enables a physiologically-based human model to navigate complex obstacles within a short duration of time and simulate changes in gait that occur after receiving a prosthesis in order to develop approaches that may be generalized to health conditions impacting movement. Additionally, the Personalized Medicine: Redefining Cancer Treatment competition [113] set the task of classifying clinically actionable genetic mutations. More specifically this task may be stated as; given the genetic mutations of a cancer tumour, classify the mutations which accelerate tumour growth as separate from the neutral mutations.

In contrast, we developed a novel competition applying Machine learning to the global challenge of malaria elimination, using agent-based models to learn from simulation and demonstrate novel strategies for the elimination of malaria in an endemic location. We believe this challenge builds upon interest across the Machine learning community for interpretable, scalable, and repeatable models which solve real world challenges, with higher level motivation provided by the promise of Machine learning assisting in the elimination of a disease, which for centuries has been missing the medical breakthrough to achieve this goal. Such a challenge applying Machine learning to epidemiological models, specifically malaria, has never been run to the knowledge of the organisers. This applied form of Machine learning treating such models as a simulator, is also novel.

5.5.1 Human-in-the-loop

The environment abstractions of learnt model approximations presented have been adequate for promoting engagement, evaluation and demonstrating potential for new benchmarks and approaches and there is a critical need to support human-in-the-loop evaluation for any final application or decision, which could be made from such a process. To this end, the evaluation of a domain specialist or policy maker is required, effectively pairing an intelligent software agent not only with the environment and compute to explore new policies, but also with direct feedback from a domain specialist. By providing a scalable platform, that supports such evaluations, this could encourage the community to benchmark performance on tasks outside of the traditional Reinforcement learning playground and lead to a better understanding of AI guided decision making processes [226].

Chapter 6

Multi-Step Prediction and Control with Reinforcement Learning

I simply wish that in a matter which so closely concerns the wellbeing of the human race, no decision shall be made without all the knowledge which a little analysis and calculation can provide.

Daniel Bernoulli, 1760

In Chapter 2 the Multi-Armed Bandit (MAB) framing was presented as a tool in the search for an optimal action-value function $Q(a)$. This use of online learning for prediction and control will now be further developed for the purposes of sequential or multi-step decision making. Performing predictions and learning control policies for epidemiological models is a decision making task, which may be constructed as a sequence of actions, generating possible future outcomes and for which there may be delayed rewards. Through the sequential decision making approach we'll use formalisms to perform predictions across possible futures (from the current observation) and learn a behaviour function based on observations to guide control (via a policy). This development of temporally situated decision making from epidemiological models will be driven through the paradigm of Reinforcement learning (RL).

Ever important in our description of making decisions across time is that this decision-making is problem dependant. To further detail the problem framing, decisions for epidemiology control and planning may be made on varying discrete time intervals. While based on empirical observation and through discussions with domain experts and policy makers, a one year decision cadence is often used for endemic diseases (HIV, Tuberculosis, Malaria) supported by organisations such as the Global

fund ¹. This may be guided by monetary constraints, implementation challenges, or simply by the fact that yearly plans, with perhaps a longer horizon overview, are the digested format for the users, although evidently for emerging disease outbreaks the challenge is more immediate and the cadence of decision making reflects this. Ultimately the task is to drive the epidemiological system to asymptotic stability, or eventual eradication, with any delays potentially compounding the number of negative outcomes.

Typically an epidemiological disease model used for decision making will proceed on a simulated time frame of days. As changes in epidemiological state are not required at any finer resolution for the purposes of population level decision making or assessing health impacts, any particular task will aggregate the data such that it is relevant for the decision making task.

6.1 Reinforcement Learning

By positioning epidemiological models as an abstraction of the real world, they may be posed as a simulation environment for an Reinforcement learning (RL) agent to interact with. The following sections will attempt to handle this framing of the sequential decision making task, broken down into the elements of state in Section 6.1.1, action in Section 6.1.2 and reward in Section 6.1.3, whereby a Reinforcement learning agent is the algorithmic instantiation of our approach to ‘search’ for optimal control policies.

As in Figure 6.1, we acknowledge the abstraction that a mechanistic model is driven by disease specific data. The presence of this data may also act as real-world experience, but is necessary should the model not be calibrated as in Chapter 3. The presence of real world experience may also guide an agent’s learning in combination with simulated experience, while we acknowledge that real world experience is significantly more expensive than that which may be generated from a useful epidemiological model. Here we use the term ‘expensive’ in the direct sense of loss of life and of physical resource. For effective application of RL we’ll often be required to simulate much more experience than may be feasible to observe in the real-world. We’ll now define the observations required by an agent in the RL framing.

¹<https://www.theglobalfund.org/>

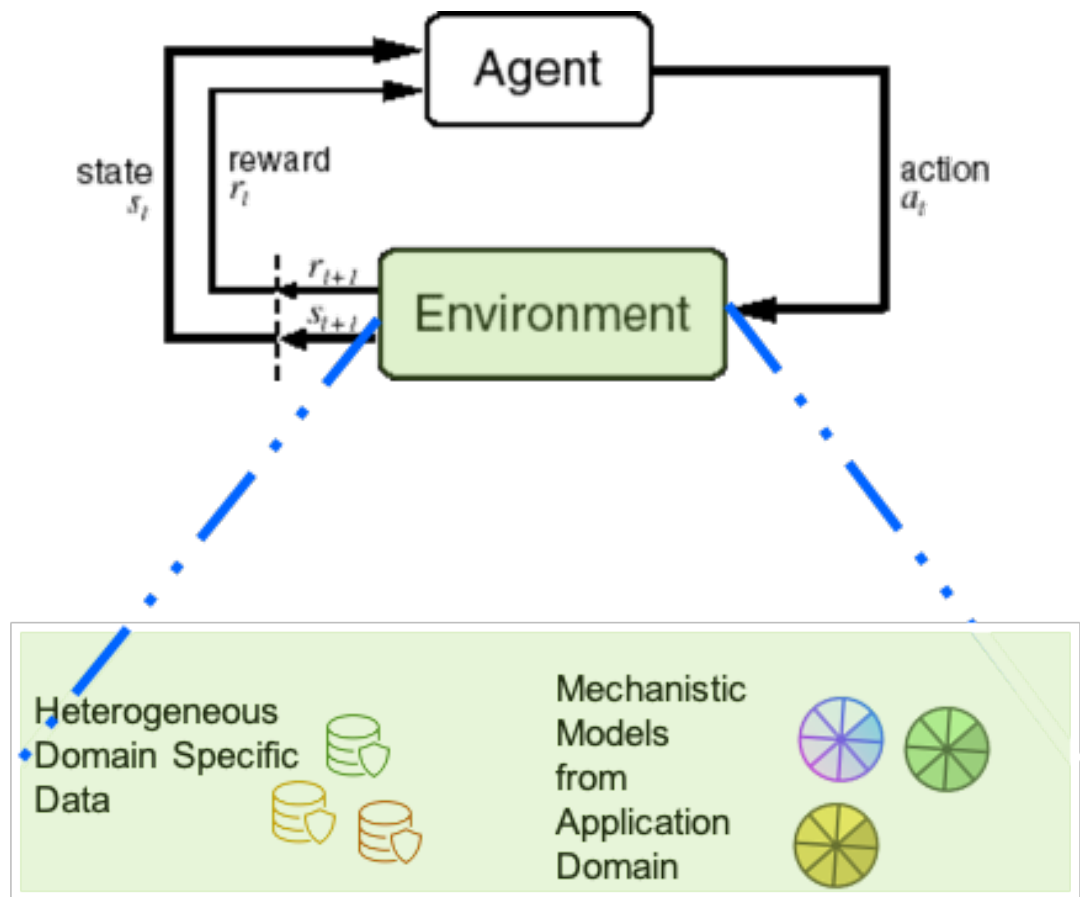


Figure 6.1: Reinforcement Learning in Environments of epidemiological model simulations. Reward $R_t = r_t$, State $S_t = s_t$, Action $A_t = a_t$

6.1.1 State

A state signal is one which provides useful observations to our learning agent. For the purposes of sequential decision making we require that the state S_t is Markov or has the Markov property [120], which may be summarised in the following quote:

‘The future is independent of the past given the present’

Which may also be written as:

$$P[S_{t+1}|S_t] = P[S_{t+1}|S_1, S_2, \dots, S_t]$$

The current state (S_t) hence provides all information based on the history ($S_1, S_2, \dots, S_t$) and is sufficient to perform a prediction of the next state (S_{t+1}). The analogy to epidemiological models would be that this state observation would be sufficient information to restart the model from the point it was stopped, generating the same model prediction as a complete model run for the same time period under the same simulated policy.

The state domain of an epidemiological model may be vast for such purposes, where typically inputs to models may consist of structured files containing, population demographics, epidemiological parameters, climate predictions, etc. This is a problem for our framing, which suffers Bellman’s curse of dimensionality [25], meaning that the number of possible states $n_S = |\mathcal{S}|$ grows exponentially with number of state variables. Therefore, useful encodings of state will be required, which don’t necessarily retrieve a one to one mapping to the original model but allow us to proceed through the planning process.

A probability of transition $\mathcal{P}_{ss'}^a$ from state s to s' under action a defines our Markov Decision Process (MDP).

$$\mathcal{P}_{ss'}^a = P[S_{t+1} = s' | S_t = s, A_t = a]$$

We revisit this in more detail in Section 6.1.4.

6.1.2 Action

Actions are the mechanisms by which agents may perturb the environment. Given our framing that this environment will be a dynamical system, defined by an epidemiological model, such actions will consist of possible inputs to the model. Specifically for the task of control we aim to perturb the environment based on discrete interventions

which may be performed in the epidemiological system. Previously, the definition of an action in Section 2.3.3, of a modelled set of interventions, will be relaxed such that the timing of interventions is not defined a priori before being simulated. Now the agent, as opposed to the Bandit framing of Chapter 2 has the chance to perform action selection through interaction with the environment, e.g. during learning. An action may receive delayed rewards in any sequential decision making task, and we do not expect to reap the benefits of our actions immediately. Instead, any learning method should be able to identify the sequences of actions which lead to positive outcomes. For the purposes of describing a value function (in Section 6.1.7) we'll consider actions as a finite set $\mathcal{A}$.

6.1.3 Reward

Much the same as in Chapter 2, reward for sequential decision making is as per that for a Multi-Armed Bandit. However, intermediate rewards may be received during interaction, whether delayed or not. The guiding principal in the use of a reward signal for agent learning is captured in Sutton's reward hypothesis [194]:

‘All of what we mean by goals and purposes can be well thought of as maximization of the expected value of the cumulative sum of a received scalar signal (reward)’

We assume a reward transition function exists, mapping state and action pairs to received stochastic rewards, such that the reward upon taking action a in state s is:

$$\mathcal{R}_s^a = \mathbb{E}[R_{t+1}|S_t = s, A_t = a] \quad (6.1)$$

This reward transition function is necessarily unknown by the agent and must be learnt via interaction, typically via an unknown Markov Decision Process (MDP).

6.1.4 Markov Decision Processes

Here we combine the concepts of the preceding Sections 6.1.1, 6.1.2 and 6.1.3, extending the Markov property to the complete Markov Decision Process definition, containing States (S), Actions (A) and Rewards (R). For all $S_t, A_t, S_{t+1}, R_{t+1}$ we then have:

$$P(S_{t+1}, R_{t+1}|S_t, A_t) = P(S_{t+1}, R_{t+1}|S_t, A_t, R_t, S_{t-1}, A_{t-1}, \dots, R_1, S_0, A_0) \quad (6.2)$$

The result of this summary is that we may restrict our attempts at optimal control to the search for reactive policies (Section 6.1.6). A reactive policy being one that depends only on the current state observation ($S_t = s$) and defines the behaviour function of which action ($A_t = a$) is selected given the observation.

A Markov Decision Process (MDP) is a tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{R}, \mathcal{P}, \gamma \rangle$, where $\mathcal{S}$ is our state space, $\mathcal{A}$ is a finite set of actions, $\mathcal{R}$ a Markov reward process, $\mathcal{P}$ defines state transition probabilities and γ the discount factor $\in [0, 1]$. Therefore, in solving the sequential decision making task from epidemiological model based experience, we'll require that all environments conserve the Markov property, or may be reduced to a form which is tractable as a MDP.

6.1.5 Cumulated Return

The cumulated return G_t is defined as the summed reward after time t is discounted via the factor (γ).

$$G_t = R_{t+1} + \gamma R_{t+2} + \dots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \quad (6.3)$$

In a finite horizon task, by which we mean that the number of steps k in the learning episode is finite, this becomes a finite summation and it may be argued that all multi-step decision making tasks in the context of epidemiological prediction, control and planning, are finite horizon tasks. We do not consider systems ad infinitum - perhaps as part of the human condition, we are most interested in the task of decision making over a single human lifetime, though often far shorter horizons, where upon achieving disease elimination, eradication may return our systems to asymptotically stable solutions or no further cases. If we are required to control endemics over a longer horizon this is possibly a far more expensive task, although inexpensive control measures may exist, the length of the planning horizon should influence the decision making process. For an infinite horizon task that is episodic, instead of continuing, we represent termination to an absorbing state. Such an absorbing state may be mapped to eradication of the disease for the analysis that we perform. The discount factor (γ) may therefore be thought of as placing more weight on rewards which are achieved sooner, while often conditioning against the Infinitely delayed splurge [147] phenomenon observed in infinite horizon agent learning tasks. In the absorbing state example of eradication, this particular goal would have to be associated with a zero reward to ensure returns remain finite. For a short horizon task of less than one year,

it is likely that the discount factor will be set to unity $\gamma = 1$ and as such not influence the summation in calculating the cumulated return (Equation 6.1.5).

6.1.6 Policy

A policy defines an agent's behaviour function. For the purposes of learning within a MDP, policies may be reduced to the reactive type. A reactive policy is one which provides a mapping from the state observed to the action to be taken immediately at that time step. The policy at time step t , $\pi_t(s, a)$ defines a mapping from states to action probabilities such that $\pi(s, a) = \text{probability that } A_t = a \text{ when } S_t = s, P(a|s)$. The above specifies a reactive stochastic policy, while for every MDP there exists at least one optimal deterministic reactive policy whereby $\pi(s) = a$. Reinforcement learning methods specify how the agent learns an effective policy as a result of experience. Roughly, the agent's goal is to accumulate as much reward as possible over the long run, therefore reinforcing reactive policies which maximise the reward for the problem. Significantly, it should be noted that policies are stationary, i.e. a function of state observation and independent of time, further meaning that the behaviour function has no knowledge of epidemiological simulation time, only the observations it receives, in principal policies whether deterministic or stochastic are implemented based on observation. Thus our sequence of decisions may be built from a policy which proceeds by evaluating which actions should be chosen based on observations. We'll now look to address the mechanisms by which such a policy may be quantified in the search for optimality or control.

6.1.7 Value Function

Value functions are the primary tool for reasoning about future reward in a MDP. The base form is the state-value function under a fixed policy π , such that value, the expected cumulated reward, is given as:

$$v^\pi(s) = \mathbb{E}_\pi [G_t | S_t = s] = \mathbb{E}_\pi \left[\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} | S_t = s \right] \quad (6.4)$$

in which this may be viewed as a prediction based on following a fixed policy regarding future reward conditioned on the current state. For example, in Chapter 3 a single prediction may be based on the current intervention plan and a forward epidemiological model run. The associated cumulated return (Section 6.1.5) would describe the value function of the state (in this case the set of epidemiological model

inputs) at the start of model observations. Therefore, the value function for a single prediction will provide expected reward under the fixed policy - the model forecast, and the outcomes of which are quantified by our reward function for the observed states. This is useful for characterising states which lead to high future reward.

Additionally, the action-value of a policy π will be defined as follows:

$$q^\pi(s, a) = \mathbb{E}_\pi[G_t | S_t = s, A_t = a] = \mathbb{E}_\pi \left[\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} | S_t = s, A_t = a \right] \quad (6.5)$$

In this instance, the action-value function may guide the future reward associated with taking action a in state s , which moves us towards learning a reactive policy based on the action-value function $q(s, a)$. This is the foundation for the value-based Reinforcement learning technique of Q-learning [215] which is introduced in Section 6.3.1.1.

For the purposes of guiding optimal decision making, the properties of optimality can now be defined for our respective value functions:

$$v_*(s) = \max_{\pi} v_{\pi}(s) \quad (6.6)$$

$$q_*(s, a) = \max_{\pi} q_{\pi}(s, a) \quad (6.7)$$

Where Equation 6.6 is the condition for the optimal state-value function and Equation 6.7 the optimal action-value function. An optimal policy (π_*) may subsequently be defined by an optimal action-value function as follows:

$$\pi_*(a|s) = \begin{cases} 1 & \text{if } a = \operatorname{argmax}_{a \in \mathcal{A}} q_*(s, a) \\ 0 & \text{otherwise} \end{cases} \quad (6.8)$$

The Bellman equations (Equations 6.9 and 6.10) are used to decompose the calculation of value functions into iterative direct solutions [26]:

$$v_{\pi}(s) = \sum_{a \in \mathcal{A}} \pi(a|s) \left(\mathcal{R}_s^a + \gamma \sum_{s' \in \mathcal{S}} \mathcal{P}_{ss'}^a v_{\pi}(s') \right) \quad (6.9)$$

$$q_{\pi}(s, a) = \mathcal{R}_s^a + \gamma \sum_{s' \in \mathcal{S}} \mathcal{P}_{ss'}^a \sum_{a' \in \mathcal{A}} \pi(a'|s') q_{\pi}(s', a') \quad (6.10)$$

Most iterative solutions for the Bellman equations such as Value Iteration are solved through dynamic programming [25]. We shall continue with a focus on *model-free* approximations to these solutions, which specify agent interactions in an *unknown*

environment. Specifically, this means that the transition probabilities $\mathcal{P}_{ss'}^a$ and the Markov reward process $\mathcal{R}_s^a$ for the environment will always be assumed to be unknown, fitting our specifications of interacting with epidemiological models in a black-box manner. This outline will be extended to *model-free* methods, which approximate the environment transitions via samples or observations. The methods used in the search for optimal control policies under unknown environments will be developed further from Section 6.2 onwards.

6.1.8 Partially Observable Markov Decision Process (POMDP)

Practically all environments which we interact with, composed from epidemiological models, are abstractions made for the process of decision making. Further we may consider all models as POMDPs and generate state approximations to generically apply learning algorithms to them. Any approximation is required to capture the history of the model run, as is done by the Markov state measure. A MDP has a history-less state measure and at a minimum the POMDP should be able to define the sequential decision making task in terms of the history [117]. The size of the history will scale in a manner which is worse than the dimensionality of the state measure and may also be referred to as the curse of history: the number of belief-contingent plans increases exponentially with the planning horizon [146]. In this case, an agent or belief state is distinguished from the environment state. If an agent always has access to their own state measure, or beliefs, there are greater opportunities for transfer of approaches, along with providing a general framework for planning under-uncertainty. This was addressed in Chapter 5, where we proposed a plurality of agents to be deployed for epidemiological planning tasks.

6.2 Learning Sequential Surrogate Models from Episodic Simulators

This section will specify learning an agent model or environment from epidemiological model runs. This is necessary as the form of a heterogeneous model will not necessarily be a Markovian environment from which to learn control policies and so it will be necessary to efficiently learn a model using the supervised learning methods from Chapter 3. Learning a model approximation may increase our sample efficiency for the application of more general value or policy learning methods (Sections 6.3.1.1 and 6.3.1.2). Also, we may generate consistent descriptions of the epidemiological model uncertainty, under which reasoning is being performed, for the purposes of planning

or control. Although learning a surrogate model and then performing an agent based approach, which learns a value function or policy as an approximation to the model, introduces two sources of approximation error. Namely the sources of error from the two processes of approximation. We'll refer to any surrogate model learnt from epidemiological simulations as $\mathcal{M}$. The surrogate model $\mathcal{M}_{\theta_M}$ is a representation of the MDP tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{R}, \mathcal{P} \rangle$, based on the parametrisation θ_M , assuming that by defining our own surrogate model structure this means $\mathcal{S}$ and $\mathcal{A}$ are known. Therefore through $\langle \mathcal{R}_{\theta_M}, \mathcal{P}_{\theta_M} \rangle$, we will attempt to approximate the true (unknown) transitions $\mathcal{P}_{\theta_M} \approx \mathcal{P}$ and $\mathcal{R}_{\theta_M} \approx \mathcal{R}$.

Hence,

$$S_{t+1} \sim \mathcal{P}_{\theta_M}(S_{t+1}|S_t, A_t) \quad (6.11)$$

$$R_{t+1} = \mathcal{R}_{\theta_M}(R_{t+1}|S_t, A_t) \quad (6.12)$$

Here we assume (as is common practice) state transitions and rewards to be independent [187], such that:

$$\mathbb{P}[S_{t+1}, R_{t+1}|S_t, A_t] = \mathbb{P}[S_{t+1}|S_t, A_t] \mathbb{P}[R_{t+1}|S_t, A_t] \quad (6.13)$$

Estimating $\mathcal{M}_{\theta_M}$ may be reduced to a supervised learning problem based on tuples of experience $\langle S_1, A_1, R_2, \dots, S_T \rangle$ [16], decomposing a sequence of decisions as follows:

$$\begin{aligned} S_1, A_1 &\rightarrow R_2, S_2 \\ S_2, A_2 &\rightarrow R_3, S_3 \\ &\vdots \\ S_{T-1}, A_{T-1} &\rightarrow R_T, S_T \end{aligned}$$

Learning $\mathcal{R}_{\theta_M}$ as an approximation to $\mathcal{R}$ thence becomes a regression problem and we will continue to use Gaussian Process Regression (GPR) for this task. In tandem, learning the approximation of $\mathcal{P}_{\theta_M}$ is a density estimation problem [60], for which we'll leverage the Gaussian Process structure to generate state transition approximations in the surrogate model.

6.2.1 Prediction - Learning a Value Function

Before learning a model ($\mathcal{M}_{\theta_M}$) of our unknown epidemiological simulator, we need to introduce the mechanism by which we may evaluate policies based on model runs, as this will be equally useful for our purpose of generating the environment abstraction. This mechanism is via learning a value function to define the expected response of the underlying models based on their returns. Now, as we do not have unlimited evaluations to perform over multiple dimensions we must use approximation, therefore the efficiency of samples will have to be improved given a fixed computational budget of B simulations. We have proposed this problem for learning the action-value function in one-step planning introduced in Chapter 2. In continuing under a fixed computational budget, we'll now introduce extensions for multi-step planning, in order to provide a common framework to query heterogeneous disease models at scale and approximate a sequential surrogate model, which conserves the Markovian property for sequential decision making approaches.

In undertaking the prediction task for model-free Reinforcement learning via Monte Carlo (MC) methods of sampling, experience may be generated from complete model runs. In the model-free learning sense we treat the epidemiological model as a black-box, with no knowledge of what we assume to be an underlying MDP, therefore with no knowledge of $\mathcal{R}_s^a$ and $\mathcal{P}_{ss'}^a$, and these will be constructed as approximations in the process of learning model $\mathcal{M}_{\theta_M}$. Monte Carlo searches will learn from complete episodes and therefore a one-shot simulation from a model is not a barrier to our decomposition of the sequential decision making task.

The value function, based on MC methods, is considered as an empirical mean of the return rather than an expectation and it is employed for episodic MDPs which terminate [196]. This is realistic for our problems as the episode, or decision making time frame needs to be finite to be meaningful. We will therefore consider non-discounted returns given the decision making cadence.

Alternatively, Temporal Difference (TD) learning offers another example of a model-free Reinforcement learning method, though not readily applicable to all epidemiological simulations, as Temporal Difference learning is an online learning technique which performs updates before knowing the final outcome. TD learning may be more appropriate in non-terminating environments, where TD updates may be performed based on the subsequent step reward and state $R_{t+1} + \gamma V(S_{t+1})$.

MC methods do not exploit the Markov property itself, so are effective in non-Markov environments and for generating our approximations from model simulations which may not provide a Markov state. Additionally MC, being based on simulation

returns, is a zero bias, high variance estimator which naturally pairs it with the Gaussian process in the presence of smoothness between nearby observations. Further MC is a method which generally works well for ‘black-box’ models [181]. Finally, MC sampling may be used to break Bellman’s curse of dimensionality in the size of the state space and the curse of history in the case of a planning from a POMDP [99] [146].

In any Monte Carlo (MC) method the law of large numbers dictates that our observations will convergence to our true expected underlying values [228], in our case $V(s) \rightarrow v_\pi(s)$ as $N(s) \rightarrow \infty$, with updates performed on $V(s)$ after each episode. In this way the number of visitations of each state $N(S_t) \leftarrow N(S_t)+1$ and the state-value function is updated subject to its current value ($V(S_t)$) and the observed cumulated return (G_t), with the state visitation count ($N(S_t)$).

$$V(S_t) \leftarrow V(S_t) + \frac{1}{N(S_t)}(G_t - V(s_t)) \quad (6.14)$$

In order to generate useful observations of epidemiological state for decision making, we’ll need to further develop an often used framework of the decision tree, where the nodes in a decision tree capture a state approximation for the planning task. Finally, this Section on MC predictions or policy evaluations provides the basis for evaluating our *Rollout* policies required in the complete planning approach of Monte Carlo Tree Search (MCTS) laid out in Section 6.2.3.

6.2.2 Simulation Based Search - Decision Trees

In order to learn from black-box epidemiological models we’ll impose a tree structure on the decision making task. Search or decision trees are a common structure used in supervised Machine learning [44], for the purpose of sequential decision making each node of the search tree describes a Markov state S . At this node we will store the state-action value function $q(s, a)$, along with a visitation count for each action a and the total count for the node $N(s) = \sum_a N(s, a)$, correspondingly each branch of the tree provides a description of our state transitions $\mathcal{P}_{ss'}^a$. Initialisation of the search tree will be considered as setting $Q(s, a) = 0$ and $N(s, a) = 0$.

For this common framing across epidemiological models for decision making we will make abstractions about the task which determine the structure of the solutions developed. In our case this requires that the cadence at which interventions are to be deployed, along with the number of possible actions, need to be specified and will fully define the structure of the learnt surrogate model. Hence it needs to be guided by judicious choices with respect to the application domain. This choice of

decision making cadence and number of actions directly effects the number of nodes and transitions in the decision tree.

In the following description it is assumed that for a chosen episode of length k the task is to generate a sequence of k actions. In this case the episode length defines the depth of the decision tree and is related to what we have informally referred to as the decision making cadence for the modelled time frame. Subsequently, the history (which is the sequence of actions) may be used to define the state, or each node, in the search tree. This is the simplest encoding of state for the environment being learnt, though at the same time by retaining the history of the sequential decision making task, especially where each action is of dimension d , it is necessarily to acknowledge an increased complexity to the number of total states described by the history and will look at approximations to break this curse of dimensionality. If we continue to assume a finite set of actions $\mathcal{A}$, the MDP states may be constructed based on the history of actions as follows:

$$S = (a_1, a_2, \dots, a_i) \mid 1 \leq i \leq k, \forall [a_i \in \mathcal{A}] \quad (6.15)$$

In any state, actions may correspond to the implemented interventions and their coverage in the time frame until the next decision. With this definition of $s \in \mathcal{S}$ and $\mathcal{A}$ the transition function $\mathcal{P}_{ss'}^a$ reduces to a deterministic transition function, as actions map to a single state.

For further illustration consider $s = (a_1, a_2, \dots, a_i)$ with the selection of action a_t governing transition to the next state $s' = (a_1, a_2, \dots, a_i, a_t)$. Terminal states will necessarily be of length k , the length of the sequence of actions, and the reward for each state will be based on the episodic returns as in Section 6.2.1. The reward of the terminal state will equate to the return for the sequence of actions and to generate such a reward this may be specified on episodic simulation returns, through for example the cost-effectiveness of the policy (the Cost Per DALY Averted (CPDA) calculation is described in Section 2.3.4.3).

We assume a consistent initial state between epidemiological model runs, this is the root node in the search tree and may be thought of as the start of the intervention period of the model run, or even the current state, "Now" as in Figure 6.2. This should be consistent between different epidemiological models, which aim to describe the same disease and location over the same time frame, for the same decision making task. This consistency in generating comparisons and learning sequences of interventions is fundamental for an 'apples to apples' comparison of relevant intervention plans. At a minimum we may wish to assume the epidemiological models

are calibrated with the same data for the same location and may or may not have been validated by an epidemiologist. We will now move on to the algorithmic use of this notion of a search tree in the generation of a model $\mathcal{M}$, which may be used for planning or which may itself be used directly in the task of control in extension to the Multi-Armed Bandit (MAB) for control of Chapter 2.

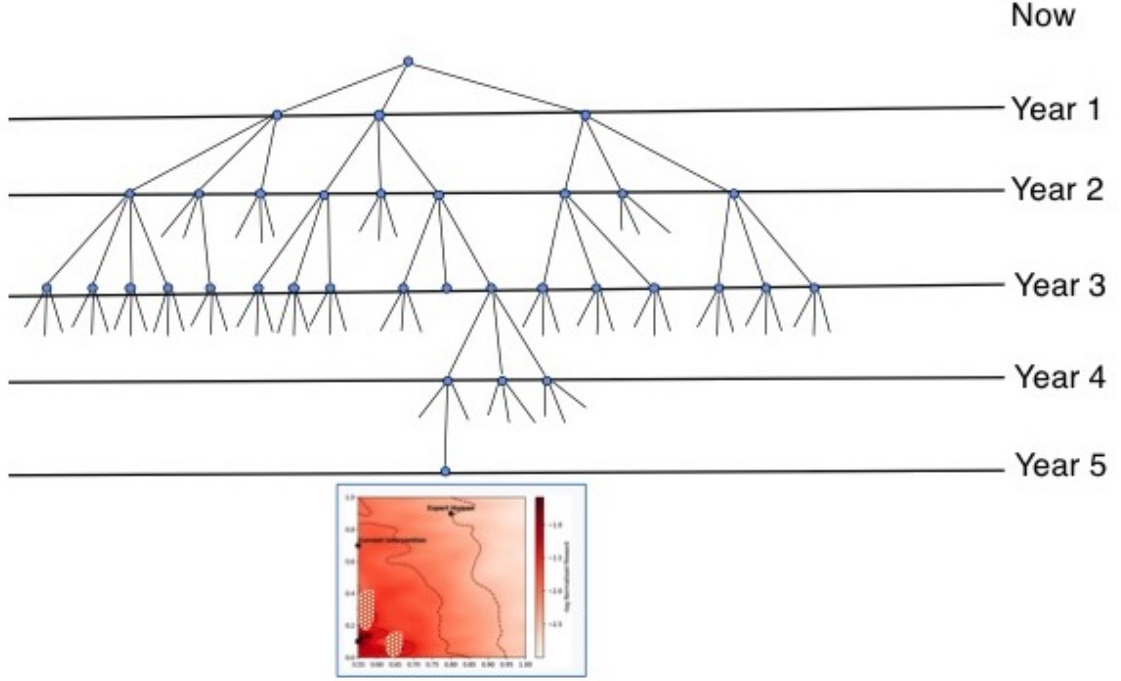


Figure 6.2: Example of cumulated return as a two dimensional response surface, as a result of a sequence of decisions for 5 years.

6.2.3 Monte Carlo Tree Search (MCTS)

By combining generative predictions via Monte Carlo (MC) methods and the structure of a search tree, Monte Carlo Tree Search has had success as a planning approach in large state spaces including the ultra-large spaces found in challenging games, most notably the game of Go [47, 56, 81]. Any direct optimal solution to such problems is infeasible due to the large size of the state-action space, therefore approximate solvers are used, in the form of MC samples. Further, we wish to restrict samples to relevant regions of the state-action space, which is achieved through the Monte Carlo Tree Search approach.

MCTS can be broken down into the following four steps:

1. *Selection*: Starting from the root state, select successive actions, appended to the history as states, until a leaf state is reached. The root state is taken as the start of the epidemiological model run and a leaf is any state that has a potential child from which no simulation (*rollout*) has yet been run.
2. *Expansion*: Unless the leaf state is the terminal state, create one (or more) child nodes and choose a next action from one of them. Child nodes are any valid interventions from the leaf state. The process of expansion may be guided by the following notions:
 - in-tree: actions select child states to maximise $Q(s, a)$
 - out-tree: pick actions randomly to explore child states
3. *Simulation*: Complete one random *rollout* from a child node. The *rollout* policy may be as simple as choosing uniform random actions until a terminal state is reached. Any simulation will consist of a sequence of actions of length k .
4. *Backpropagation*: Use the result of the rollout to update information in the nodes on the path using MC updates: $Q(s, a) \leftarrow Q(s, a) + \frac{1}{N(s, a)}(G_t - Q(s, a))$. In this case recording state-action counts $N(s, a)$.

The success of this technique in deterministic games is well documented. To stretch the concept of such a game tree to the process of intervention planning from simulation, we need to be able to efficiently combine the processes of selection and expansion to maximise our exploration of regions of interest. This is detailed in the following section through decomposing the problem into a sequence of Multi-Armed Bandit decisions, aiming to achieve efficiency in the number of simulated runs or experience through learning a model $\mathcal{M}$ as part of the search tree.

6.2.4 Gaussian Process Regression with Upper Confidence Bound selection in Monte Carlo Tree Search for Learning Sequential Surrogates (GP-UCB-MCTS)

Optimism in the face of uncertainty [48] will provide our guiding principle to extending the Monte Carlo Tree Search (MCTS) approach for the generation of surrogate models in sequential decision making. The generation of a sequential surrogate model using GP-UCB-MCTS (Algorithm 4) is presented, to produce a surrogate model from simulation results based on inference performed for the state action value function

$q(s, a)$ at each node. The value $q(S, A)$ is estimated using regression across a continuous action-space at each node, in order to reduce to a finite set of transitions, and bound the size of the history which may describe the system. $Q(S, A)$ needs to contain both discrete actions and be compressed through density estimation techniques in order to bound the size of the search tree. This search tree will have a depth k , though approximations to the number of states will govern the number of branches in the tree, based on the number of possible transitions. Value function approximation is performed using Gaussian Process Regression (GPR), in order to scale the efficiency of model free methods learning from experience. This inference will improve convergence with fewer samples. Rather than a look-up table, we now have a function that is learnt as a $\mathcal{GP}$, which may be constructed in batches of simulations.

We introduce a Monte Carlo Tree Search (MCTS) approach to generating a sequential surrogate model using the GP-UCB-MCTS algorithm, inspired by Monte Carlo Tree Search and work on the Upper Confidence Bound (UCB) solution to the Multi-Armed Bandit (MAB) problem [17, 18] along with the Upper Confidence applied to Trees (UCT) algorithm [82]. This is a formulation which combines the natural confidence bounds of Gaussian processes, for uncertain sequential decision making in stochastic simulation problems. Treating each action selection as a Bandit problem from a state node in the search tree, this structure may be generated from samples, for example, using MCTS with simulators which do not allow for direct manipulation of sequential decisions as they are episodic in formulation. In creating a representative sequential surrogate model the aim is to increase the speed of learning for Reinforcement learning (RL) based approaches, which may perform planning from the surrogate $\mathcal{M}$. This has been used to motivate development of agent based insight into decision making from computationally expensive simulations later in the Chapter (Section 6.3), along with external developments based on sharing of efficient sequential surrogate models which is presented in Chapter 5.

Algorithm 4 is initialised with a random sample of the action space ($\mathcal{A}$) for each node in the random searches. Subsequent actions are chosen to further explore the search tree nodes which each contain their own learnt GP model based on MC returns from preceding simulation runs. In order to balance exploration and exploitation in generating the tree we make use of the Upper Confidence Bound (UCB) heuristic introduced in Chapter 2.

Further heuristics have been included for practical realisations of this approach, such as a branching factor (B) commonly of value three has been used along with a uniform discretisation of the action space ($\mathcal{A}$) for each node, as visualised in Figure

6.3. In this sequential surrogate model described by a tree, each node consists of mean and covariance for the estimated action value function ($Q(S, A)$). This approximation of $q(s, a)$ for the epidemiological model may be sampled at query time by any agent code. Additionally an ensemble may be generated across $Q(S, A)$ for different epidemiological models where the same tree structure is learnt, eg. using the same intervention sets and decision time frame, through the use of the techniques described in Section 5.3.1.

Result: $Q(S, A)$

Initialisation;

Search Tree parameters: B = branching or transition number; Tree depth k [Section 6.2.4];

$\mathcal{GP}$ priors $\theta_{\mathcal{GP}}$ [Section 3.1.1.2];

All $Q(s, a) = 0$ [for other initialisation values see Section 6.2.4.1];

for $t = 1, 2, \dots, \text{end}$ **do**

for $j = 1, 2, \dots, k$ **do**

Selection;

if $Q(S^j, A) = 0$ **then**

A_t^j random action $\in \mathcal{A}$;

else

$A_t^j = \underset{a \in \mathcal{A}}{\operatorname{argmax}} \left[m_{t-1}(\mathbf{a}) + \beta \sqrt{k_{t-1}(\mathbf{a}, \mathbf{a}')} \right]$ [Section 2.3.5.1];

$S^j \rightarrow S^{j+1}$ based on compression for B choices [Section 6.2.4];

end

Expansion;

if $j=k$ **then**

Simulation;

$\mathbf{a}_t$;

else

 Repeat *Selection*;

end

end

 Return: G_t ;

 State-action count: $N(S, A) \leftarrow N(S, A) + 1$;

Backpropagation;

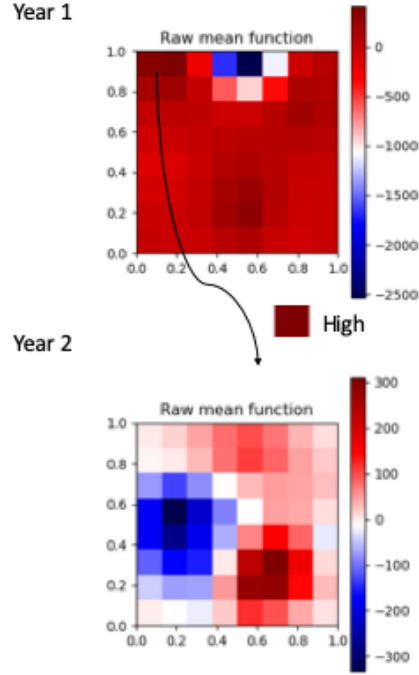
$Q(S^j, A^j) \leftarrow Q(S^j, A^j) + \frac{1}{N(S^j, A^j)}(G_t - Q(S^j, A^j))$;

 Update posterior $Q(S_t, A_t)$: mean $m_t(\mathbf{a})$, covariance $k_t(\mathbf{a}, \mathbf{a}')$;

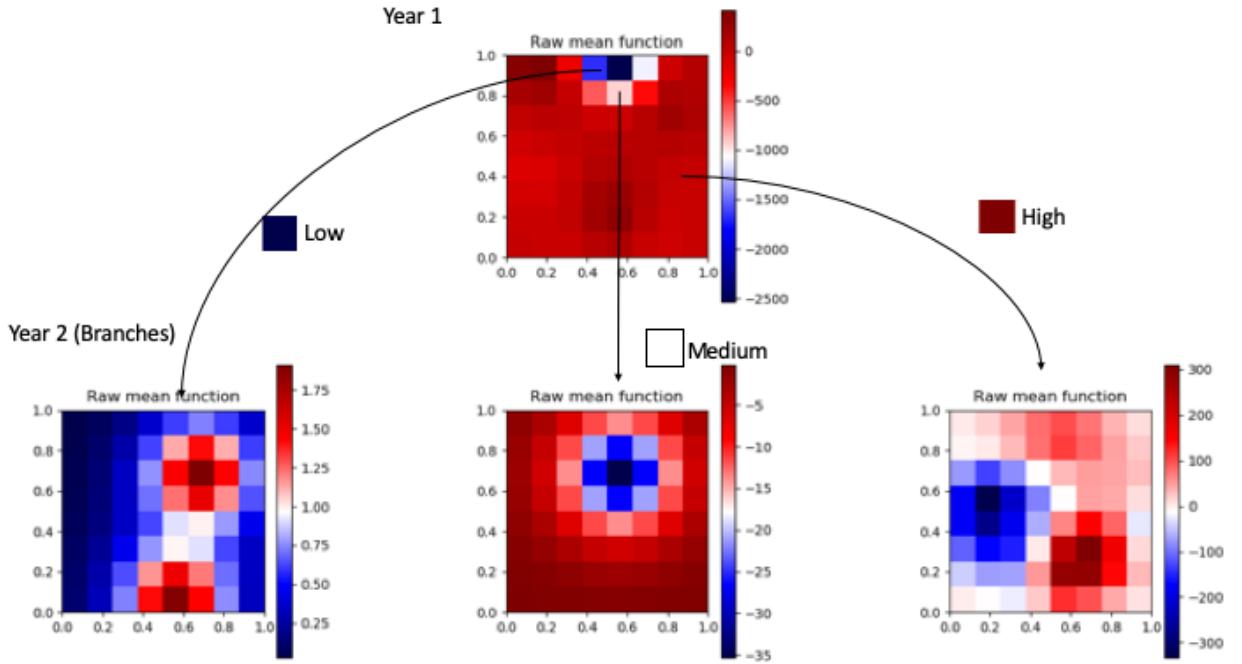
end

$Q(S, A) \rightarrow q_*(S, A)$

Algorithm 4: GP-UCB-MCTS



(a) Discretised 2D action-value function for $Q(\mathbf{a})$ showing mean $m(\mathbf{a})$



(b) Transitions based on branching (B) of order three for High, Medium and Low value actions

Figure 6.3: Visual description of generation of sequential surrogate $Q(S, A)$ with regards to the mean $m(\mathbf{a})$ for each state.

6.2.4.1 Incorporating domain knowledge into MCTS

This section is included to highlight that generating new search trees based on available epidemiological models may be sped up in the multi-model scenario. The most immediate mechanism to achieve this during the learning process, including domain knowledge, is to initialise the tree based on the learnt nodes for other models, assuming the same structure persists. Therefore, the initialisation of $Q(S, A)$ for each of the nodes, $N(S, A) = 0$ will help to focus the search to policies we believe, from other models, to be better, without altering the convergence properties of the Monte Carlo approach. In this case the learnt nodes from another model may act as a GP prior for the action selection task in further Monte Carlo simulations. While it is also conceived that trees may be later combined at each node based on the committee predictions, as introduced in Section 5.3.1, boosting our ability to quantify uncertainty in the problem and therefore have an effective multi-model description for the decision making task.

6.3 Control - Optimal Policies and Planning

Now we return to the task of learning a policy in extension to the developments of Section 6.2 in learning a surrogate representation of the sequential decision making task from epidemiological models. We've already stated that Reinforcement learning (RL) is complicated by the fact that the reinforcement, in the form of rewards (Section 6.1.3), is often intermittent and delayed [61]. The agent may perform a long sequence of actions (Section 6.1.2) before receiving any reward. This leads to the difficulty in solving the temporal credit assignment problem to attribute credit or blame to actions when the reinforcement is finally received [191]. The policy (Section 6.1.6) that an agent learns via interaction represents a particular plan that indicates the best action to take in every possible state that it observes.

There are two fundamental approaches to designing an agent for problems in which no information is given with regards to the transition probabilities and rewards ($\mathcal{P}_{ss'}^a$ and $\mathcal{R}_s^a$ respectively), this defines the domain of model-free Reinforcement learning. In the first approach the agent attempts to learn the transition probabilities and rewards and then construct an optimal policy off-line, using a method such as Bellman's value iteration in Equation 6.6. For the second approach, the agent attempts to learn an optimal policy by constructing an evaluation function for use in selecting the best action to take in a given state [148]. The agent through learning this evaluation function cannot predict what the state transition resulting from a given action will

be, its evaluations based on determining whether this new state leads to greater future rewards or return compared to the selection of another action.

6.3.1 Optimal Policy Learning

Learning an optimal policy requires the specification of an action-value function in addition to the value function of the prediction task (Section 6.2.1). We’ve shown in the previous section how we may generate a sequential surrogate model from epidemiological simulations and this may be used for planning as outlined previously in Chapter 5. The purpose of this section is to specify common domains of learning methods, specifically model-free methods, in the hope that generalisable approaches may be developed. Results will be shown based on learning from computationally less expensive sequential surrogate models, learnt from sets of experience and also in extension to less computationally expensive, population level, deterministic compartmental models.

6.3.1.1 Q-Learning

Q-learning is perhaps the most widely introduced value-based learning method in RL. It is a model-free approach to solving the Bellman equation (Equation 6.10), finding an optimal reactive policy for the value function through assessing the value of an state/action pair. Q-Learning performs its own updates of the form:

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha \left[R_t(S_t, A_t) + \gamma \operatorname{argmax}_a (Q(S_{t+1}, a)) - Q(S_t, A_t) \right] \quad (6.16)$$

In Equation 6.16 we introduce a new parameter $\alpha \in [0, 1]$ the learning rate, and continue to use the discount factor γ as previously defined in Section 6.1.4. Q-learning performs off-policy updates, whereby our next action is chosen using a behaviour policy which will encourage exploration, e.g. ϵ -greedy. Although $Q(S_t, A_t)$ updates are performed based on following the target policy where the state value function of the next state is based on the greedy policy:

$$\pi(S_{t+1}) = \operatorname{argmax}_a (Q(S_{t+1}, a)) \quad (6.17)$$

This method of off-policy updates offers the promise of learning about the target policy (necessarily a greedy policy), which is ultimately the final policy we evaluate for the purposes of optimality (Equation 6.8), while following a behaviour policy which encourages exploration during the process of learning an the optimal control policy. Q-learning control therefore uses off-policy updates to converge on the optimal

action-value function $Q(s, a) \rightarrow q_*(s, a)$. With base implementations of Q-learning often referred to as tabular learning, the action-value function being calculated as a look-up for discrete action-state pairs, this exhibits provably exponential sample complexity, with a linearly decaying learning rate [70]. The sample efficiency of Q-learning methods may be improved through function approximation methods for the value functions, for example as previously described in multiple applications of this work using Gaussian processes [86]. An approximation to the value function will take the parametrised version $Q_\theta(s, a)$ the use of which shall be extended in Section 6.3.1.3.

6.3.1.2 Policy Gradient Methods

These approaches sit in the family of function approximation methods. Instead of approximating a value-function, policy gradient approaches look to parametrise a policy $\pi_\theta(s, a) = \mathbb{P}[a|s, \theta]$. In this case θ captures these parameters, the parameterised function returns the highest probability action based on the greatest expected cumulative reward, in the case of a stochastic reactive policy. The *REINFORCE* algorithm [219] (Algorithm 5) is a canonical example of this, and performs updates based on the discounted cumulative reward, giving a policy gradient:

$$\nabla_{\theta_n} J(\theta_n) = \mathbb{E}_{\pi_\theta} [\nabla_{\theta_n} \log \pi_{\theta_n}(s, a) Q^{\pi_{\theta_n}}(s, a)] \quad (6.18)$$

Equation 6.18 is the result of the policy gradient theorem [195] for multi-step MDPs, where θ_n is the parametrisation of the policy function approximation which returns a stochastic policy, $J(\theta_n)$ defines an objective function on the performance of θ_n , α is a step-size parameter in the gradient ascent task for the differentiable policy and $\nabla_{\theta_n} J(\theta_n)$ defines the policy gradient or gradient of the policy function approximation.

Function approximation is performed to parametrise the policy, learning a policy for the environment, not a value function, while the action space $\mathcal{A}$ is naturally continuous, as opposed to the discrete form assumed for value-based learning methods. Policy gradient methods generalise the likelihood ratio to multi-step MDPs [148], where the return G_t is an unbiased sample of $q^{\pi_\theta}(s, a)$

In policy gradients we require a function approximation and training method that is suitable for non-stationary, non-i.i.d data. Commonly, as in Chapter 2, this will be a neural network.

Result: θ_n
 randomly initialise θ_n
for *each episode* $\{S_1, A_1, R_2, \dots, S_{T-1}, A_{T-1}, R_T\} \sim \pi_{\theta_n}$ **do**
 for $t = 1, \dots, T-1$ **do**
 $\theta_n \leftarrow \theta_n + \alpha \nabla_{\theta_n} \log \pi_{\theta_n}(S_t, A_t) G_t$
 end
end
 result: θ_n

Algorithm 5: REINFORCE Algorithm [219]

6.3.1.3 Actor Critic Methods

Actor critic methods form part of the basis of reinforcement learning and were one of the earliest approaches [221], though naturally their description follows separate introductions of value and policy learning control methods. The premise of actor critic control may be thought of as providing assistance to a policy gradient learning method via function approximation of the action-value function, which is the critic, to the actor of the policy gradient method. Often referred to in the literature as A2C approaches, actor critic methods may reduce the variance in *REINFORCE* by also learning a critic to approximate $q^{\pi_{\theta}}(s, a)$, therefore $Q_{\theta_v}(s, a) \approx Q^{\pi_{\theta_n}}(s, a)$, introducing another set of parameters to be learnt θ_v . Updating the policy gradient theorem to include this approximation provides us with policy gradients as per Equation 6.19:

$$\nabla_{\theta} J(\theta) \approx \mathbb{E}_{\pi_{\theta}} [\nabla_{\theta} \log \pi_{\theta}(s, a) Q_{\theta_v}(s, a)] \quad (6.19)$$

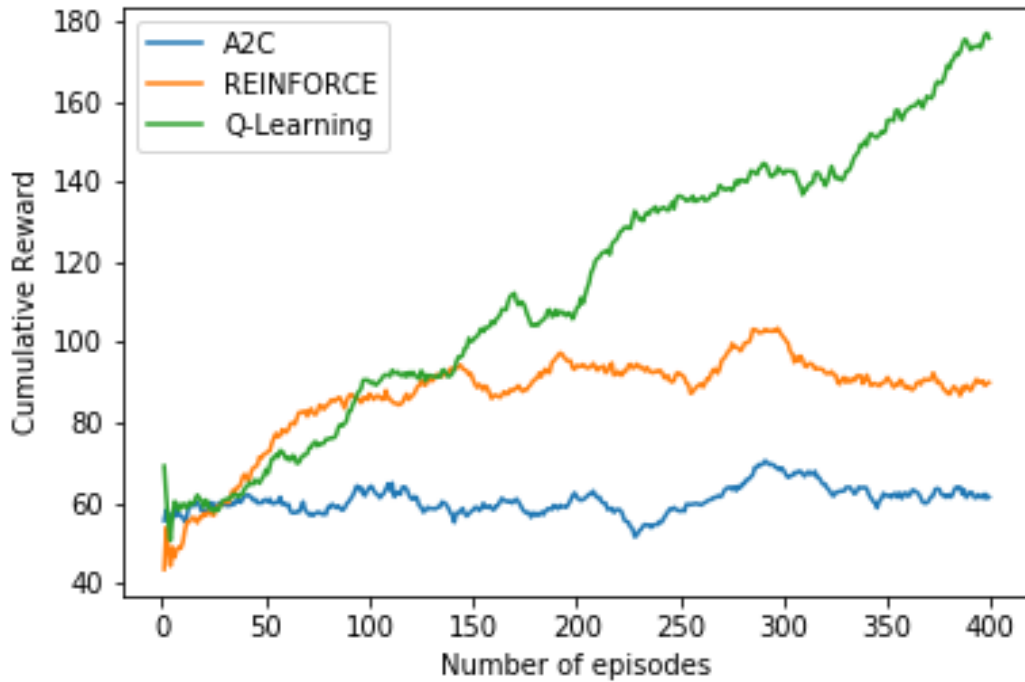


Figure 6.4: Value-based Q-learning, Policy-based Reinforce, and Actor-Critic (A2C) approaches to learning from a sequential malaria surrogate competition environment.

Figure 6.4 shows the learning curves of the three detailed approaches on a short number of episodes of training for a surrogate environment generated through GP-UCB-MCTS. In this case for limited experience the off-policy lower variance approach of tabular q-learning demonstrates the best performance and would be a candidate for selecting further episodes of real experience to augment the learnt policy.

6.4 Comparing Predictions from Multi-Step and One-Step Methods with direct Experience

As we explore more nuanced decision-making, varying the timing of interventions has been previously ignored due to complexity. Specifically incorporating timing adds to the combinatorial complexity of the problem, further stressing the need for data efficiency and also the volume of data needed to make useful inference. When making the best policy decision we can see in Figure 2.6, that for decisions of cost-effectiveness, optimal decisions may vary slightly even across the course of the month. Such responses have not been previously studied and are now being supported by the presented infrastructure as well as extending into the space of geospatial decision making. We may observe empirically in Figure 6.5 that the search for multi-step as opposed to single-step control policies provide high programmatic relevance when

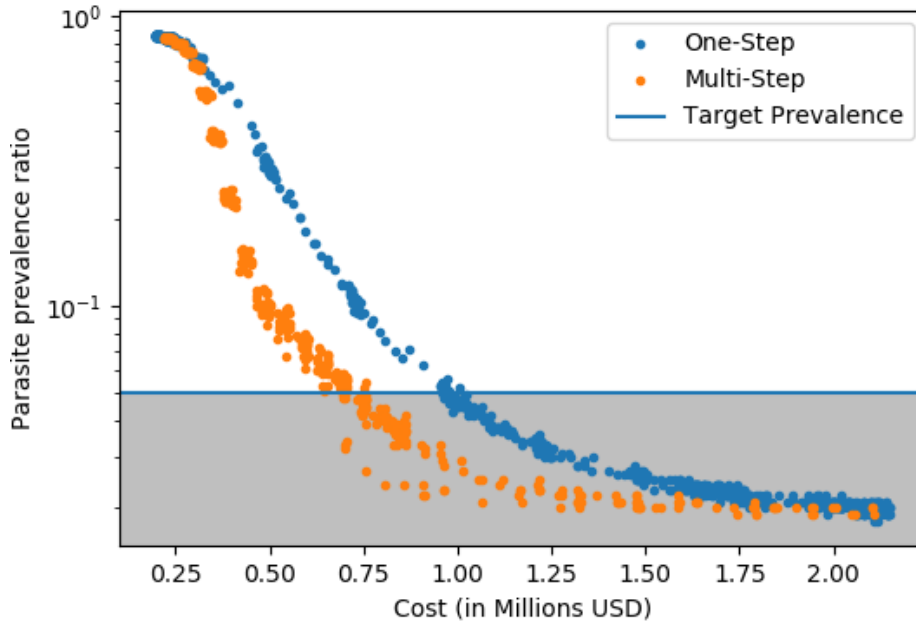


Figure 6.5: Ability of ‘Best’ multi-step policies (GP-UCB-MCTS Section 6.2.4) to consistently outperform static policies (GP-ULCB Section 2.3.5.1) for the same simulation environment

we evaluate the cost of the policy based on it reaching a desired performance. In this case, for an underlying malaria model, we are able to reduce the cost by 25% to reach the same final target prevalence, or in the presence of the same budget reach a final prevalence of 50% of the target. Given the highlighted evidence of the benefits afforded by multi-step control and casting the generation of intervention plans as a sequential decision making task, we’ll now progress the search for refined approaches and systems to accelerate the development of optimal policies for epidemiological control.

6.5 Example: Model-free Reinforcement learning with Covid-19 Compartmental Model

Compartmental models [43] provide a fundamental epidemiological model structure for decision making, their composition of rate based differential equations avoid the computational burden associated with complex interaction driven agent-based epidemiological models which as we’ve discussed may act as a barrier to model use. Though not without its own challenges in providing useful insights based on calibration of the dynamical system, our approach to such modelling has been outlined in Section 3.2. We place particular emphasis on the learning of parameter distributions

to propagate uncertainty in deterministic models. In the example below, of direct RL from an epidemiological model, we use a single rate parameter set, $\{\beta, \gamma, \zeta, \mu\}$, as per the model schematic in Figure 6.6.

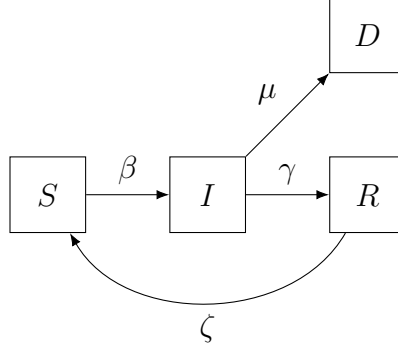


Figure 6.6: Compartmental Epidemiological model SIRS-D used for this Covid-19 model-free learning example [43].

We explore intervention plans from this SIRS-D model using actions which perturb the model transmission rate, β . The change in transmission rate has a functional mapping to a Non-Pharmaceutical Intervention (NPI) index, namely the Oxford stringency index [89], which aims to capture the stringency level of imposed NPIs on a scale of $[0, 100]$. A further functional mapping takes this stringency level $\in [0, 100]$ to a change in the transmission rate β . There we used the mapping developed in [133]. Decisions regarding a change in implemented NPIs will be made on a two week cadence. In order to learn via Reinforcement learning (RL) it's now necessary to detail a representative reward signal, observable along with the epidemiological states (compartments S: Susceptible, I: Infected, D: Dead and R: Recovered) at the same two week cadence as actions are taken.

6.5.1 Lives Saved and NPI Cost

To further enumerate the outcomes of the four-state compartmental model we've taken the following proportions for Infected (I) individuals, taken from [224] :

- Mild to moderate: 81% (mild symptoms up to mild pneumonia):
- Severe: 14% (dyspnea, hypoxia, or more than 50% lung involvement on imaging)
- Critical: 5% (respiratory failure, shock, or multiorgan system dysfunction)

Each of the Infected states has a corresponding contribution to loss of productive life, or a disease weight for the time period they are experienced, taken from [225]:

- Mild to moderate: 0.051
- Severe: 0.133
- Critical: 0.655

In addition [153] estimates an average of ten years of life lost (YLL) per Covid-19 Death (D), given that one Disability Adjusted Life Year (DALY) averted may be evaluated at \$100,000 [162], along with an estimation of costs per individual for the most stringent NPI index of \$1000, giving a linear cost based on the stringency index of the intervention $-10 \times [0, 100]$.

The goal of the agent is to maximize the expected return, a sum of the intervention costs and the cost savings in terms of DALYs averted (or lives saved) from Infected and Dead compartment values. This problem will be constrained as a finite horizon task control task of 12 steps, or a simulated six months.

6.5.2 Results of Agent Approaches to Epidemiological Control

We'll now compare the results of a variety of agent algorithms learning through repeated interactions with the Covid-19 based compartmental model (Figure 6.6), abstracted as an OpenAI gym environment (as in Chapter 5), with the distinction that this is not a surrogate model, but agent learning acting on direct model simulations through the environment.

To generate the results of Figure 6.7, agent algorithms were run for 1000 episodes. In this figure we show the final performance of each of the six implemented algorithms. Deep Q-Learning (Q-learning with function approximation) has demonstrated the greatest variation in learnt greedy policy but finds the best policy. Though we note that the expected performance was significantly poorer over 20 runs. Based on the distributions of final rewards, Genetic algorithms and Random search prove the most robust in finding effective greedy policies, though sub-optimal based on comparison with the other agent reward distributions from 20 runs of 1000 episodes.

As discussed in previous sections of this work we hope this Figure further communicates the striking heterogeneity in agent based learning approaches for control from epidemiological models. Indeed it highlights that there is not a single learner or answer to the task, but information needs to be shared through a collaborative approach on such challenging 'real-world' problems.

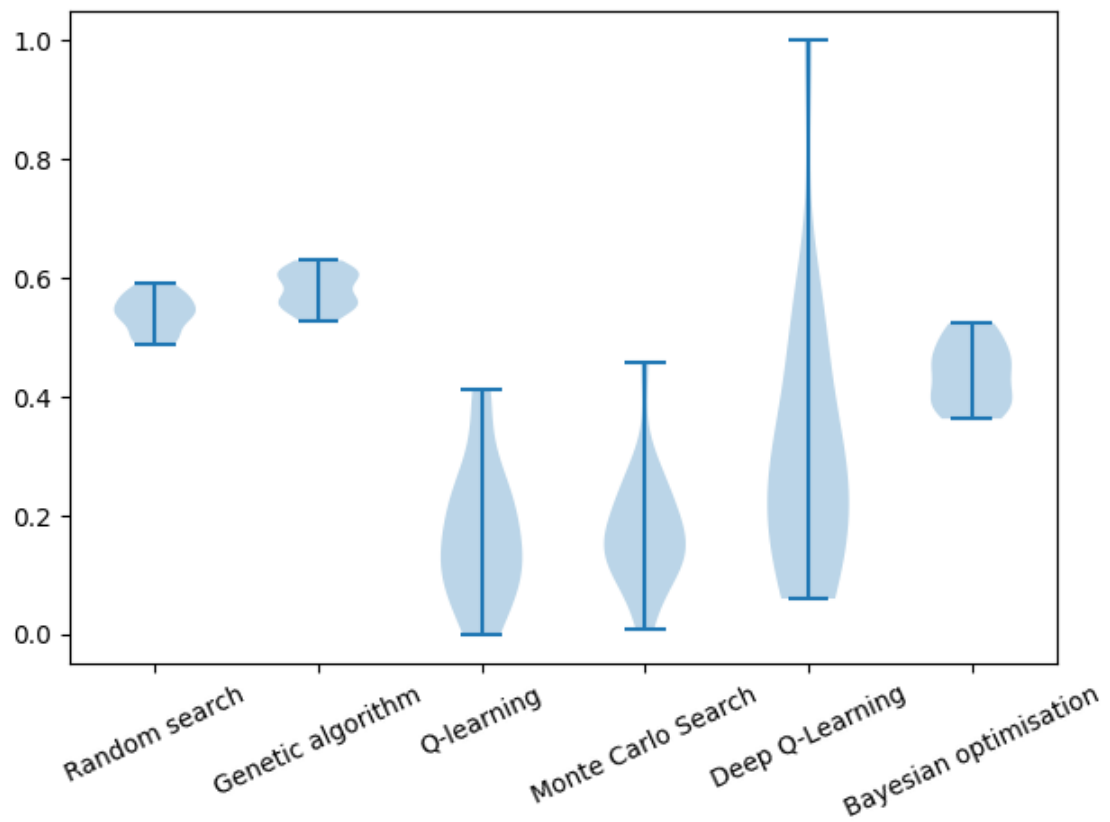


Figure 6.7: Normalised episodic reward (y-axis) of greedy policies learnt by different agent algorithms on a compartmental Covid-19 model environment, over 20 runs of 1000 episodes

Chapter 7

Conclusions and Related Work

7.1 Discussion

Throughout this work we've presented the need for organisation around the use of epidemiological models for decision making. This perspective centres around the possibilities of Machine learning (ML) methods to learn in an online fashion from model simulations to generate either single-step or sequential multi-step control policies, based on framing the learning tasks as Multi-Armed Bandit (MAB) and Reinforcement learning (RL) problems.

At the core all epidemiological models have been treated as computationally expensive black-box functions, and we may overcome some of the challenges of learning from such systems through the use of Bayesian inference. This has centred around the use of Gaussian processes as function approximations to expensive dynamical systems. The Gaussian process may be used as a surrogate model, and these surrogate models may define the value function for the single state Bandit problem or for the learning of reactive policies from a tree-based surrogate which captures the sequential decision making task for model-free RL methods.

These surrogate models have also opened up the possibilities for agent algorithms to be deployed for large scale real-world challenges demonstrated in their use for a global series of Machine learning competitions. The notion of competition effectively frames concepts as we attempt to boost, ensemble and combine performance from algorithms but also the heterogeneity in approaches to epidemiological models which are available.

Furthermore, in supporting many different model structures the ability to generate useful predictions under uncertainty has been explored as an extension to Bandit learning in the form of sequential optimisation through Bayesian optimisation. This focus being as an approach to learning epidemiological parameter sets or calibration of

models which may have stochastic outputs or to inject uncertainty into deterministic models based on learning parameter posterior distributions.

Finally, acknowledging the computational burden of developing strategies for epidemiological control insilico, we have described a scalable infrastructure that leverages abstractions to support effective deployment of existing epidemiological models and their associated data to achieve the broad goals of epidemiological control, prediction and planning.

7.2 Related Work

We must acknowledge an explosion in attention with regards to the problems explored in this work due to the global Covid-19 pandemic. Discussions around the relevance of Machine learning approaches in tandem with epidemiological specialists have often been called into question. Our perspective is that this has been a catalyst for impactful applied research.

Pharmaceutical intervention discovery has employed crowd sourcing in pandemics, with the use of Machine learning and computational tools to rapidly triage compounds and design synthetic routes, reducing the time it takes to find therapeutic treatments with the most promise [53] [80]. Machine learning based models, trained on specific biomolecules, have also offered inexpensive and rapid implementation methods for the discovery of effective viral therapies [104] along with direct applications to the design of clinical trials and experiments through the use of causal inference and RL methods [229].

Model predictions ensemble machine learning modelling for highly effective predictions in a malaria prediction of drug resistance [74], together with the development of many highly parametrised ML based approaches to contrast with the established epidemiological model literature [19] [223].

Non-Pharmaceutical Intervention (NPI) repositories have been extended through automated extraction of global NPI details from Wikipedia articles using Natural Language Processing (NLP) [190] which run in parallel to intensive manual data collection projects [89].

Though combining all of these approaches to generate optimal intervention plans remains a gap, perhaps in part due to the scale of the task and the multiple components, approaches to which have been laid out in this thesis. We’ve seen how much collaboration is needed along with the pace of development required in solving problems at this scale.

7.3 Future Directions

We have extensively explored options afforded by generating reactive policies as a disruptive mechanism to the conventional intervention plans supported in a global health setting. Existing plans being necessarily static and outlined in a funding application defining a deterministic policy based on a chosen intervention program. We'll propose future directions which could further extend the performance of stochastic reactive policies for decision making in the context of epidemiological control.

As a starting point future work will focus on tying in the insights of subgroup analysis to automation of model-based insight for epidemiological decision making. Data driven insights into the subgroups who are most vulnerable or experience heterogeneous treatment effects may allow for the development of optimal intervention plans which achieve far greater impact based on targeted interventions across time.

We have spoken extensively of the impact of sequential decisions, though this work has assumed a chosen study location and the epidemiological model would handle geospatial interactions between disease driving agents. The impact of targeting and coordinating interventions on a geospatial resolution for global benefit remains as an extension for the reader and this work.

Learning how interventions may impact epidemiological models for which we have significantly more data given the global pandemic [152] will mean possible updates, based on data driven insights, of intervention models which directly impact the processes of prediction, control and planning from epidemiological models.

Reward functions for the decision making process have been included as a necessity for the methods to be applied, we've not sought consensus on a universal function which makes the most sense for all these decision making tasks. Instead we acknowledge that this remains highly problem dependent, to introduce capabilities which allow best possible approximations to the reward function which defines an expert made decision would be a valuable contribution in line with the other techniques presented.

Practical questions remain in the problem of the 'value of information', the availability of data is never as complete as we would like. In order to make informed decisions there are many remaining challenges globally. The disparate quality of case data in Sub-Saharan Africa compared with the rest of the world for the Covid-19 pandemic remains as a stark example. There is clear evidence of the need for global coordination and cooperation, in our interconnected world we need approaches which can effectively break networks of disease interaction but not just isolate the humans.

We hope for a future where unity in global approaches helps us to overcome our most pressing challenges.

References

- [1] What is Epidemiology? — Teacher Roadmap — Career Paths to Public Health — CDC.
- [2] Pieter Abbeel, Adam Coates, Morgan Quigley, and Andrew Y Ng. An application of Reinforcement learning to aerobatic helicopter flight. In *Advances in neural information processing systems*, pages 1–8, 2007.
- [3] Jane Achan, Ambrose O Talisuna, Annette Erhart, Adoke Yeka, James K Tibenderana, Frederick N Baliraine, Philip J Rosenthal, and Umberto D’Alessandro. Quinine, an old anti-malarial drug in a modern world: role in the treatment of malaria. *Malaria Journal*, 10(1):144, 2011.
- [4] Nur Ahmed and Muntasir Wahed. The de-democratization of AI: Deep learning and the compute divide in Artificial Intelligence research, 2020.
- [5] Mohammed Al Quraishi. Alphafold at CASP13. *Bioinformatics*, 35(22):4862–4865, 2019.
- [6] Zahid Alam Khan, Zhengyong Feng, M Irfan Uddin, Noor Mast, Syed Atif Ali Shah, Muhammad Imtiaz, Mahmoud Ahmad Al-Khasawneh, and Marwan Mahmoud. Optimal policy learning for disease prevention using Reinforcement learning. *Scientific Programming*, 2020, 2020.
- [7] Murray E Alexander, Christopher Bowman, Seyed M Moghadas, Randy Summers, Abba B Gumel, and Beni M Sahai. A vaccination model for transmission dynamics of influenza. *SIAM Journal on Applied Dynamical Systems*, 3(4):503–524, 2004.
- [8] Jamie J Alnasir. Distributed computing in a pandemic: A review of technologies available for tackling Covid-19, 2020.

- [9] CDC Centers for Disease Control and Prevention. CDC - Malaria - About Malaria - History - Elimination of Malaria in the United States (1947-1951).
- [10] CDC Centers for Disease Control and Prevention. CDC - Malaria - Malaria Worldwide - Impact of Malaria.
- [11] David P Anderson, Jeff Cobb, Eric Korpela, Matt Lebofsky, and Dan Werthimer. SETI@ home: an experiment in public-resource computing. *Communications of the ACM*, 45(11):56–61, 2002.
- [12] Roy M Anderson. The epidemiology of HIV infection: variable incubation plus infectious periods and heterogeneity in sexual activity. *Journal of the Royal Statistical Society: Series A (Statistics in Society)*, 151(1):66–93, 1988.
- [13] Elli Androulaki, Artem Barger, Vita Bortnikov, Christian Cachin, Konstantinos Christidis, Angelo De Caro, David Enyeart, Christopher Ferris, Genady Laventman, Yacov Manevich, Srinivasan Muralidharan, Chet Murthy, Binh Nguyen, Manish Sethi, Gari Singh, Keith Smith, Alessandro Sorniotti, Chrysoula Stathakopoulou, Marko Vukolic, Sharon Weed Cocco, and Jason Yellick. Hyperledger fabric: A distributed operating system for permissioned blockchains. *CoRR*, abs/1801.10228, 2018.
- [14] GEB Archer, Andrea Saltelli, and IM Sobol. Sensitivity measures, ANOVA-like techniques and the use of bootstrap. *Journal of Statistical Computation and Simulation*, 58(2):99–120, 1997.
- [15] S M Niaz Arifin, Gregory R Madey, and Frank H Collins. *Spatial agent-based simulation modeling in public health: design, implementation, and applications for malaria epidemiology*. John Wiley & Sons, 2016.
- [16] Christopher G Atkeson and Juan Carlos Santamaria. A comparison of direct and model-based Reinforcement learning. In *Proceedings of international conference on robotics and automation*, volume 4, pages 3557–3564. IEEE, 1997.
- [17] Peter Auer, Nicolò Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the Multiarmed Bandit problem. *Machine Learning*, 47(2-3):235–256, 2002.
- [18] Peter Auer and Ronald Ortner. UCB revisited: Improved Regret bounds for the stochastic Multi-Armed Bandit problem. *Periodica Mathematica Hungarica*, 61(1-2):55–65, 2010.

- [19] Seyed Mohammad Ayyoubzadeh, Seyed Mehdi Ayyoubzadeh, Hoda Zahedi, Mahnaz Ahmadi, and Sharareh R Niakan Kalhori. Predicting Covid-19 incidence through analysis of Google trends data in Iran: Data mining and Deep learning pilot study. *JMIR public health and surveillance*, 6(2):e18828, 2020.
- [20] A. Baldominos, E. Albacete, Y. Saez, and P. Isasi. A scalable Machine learning online service for big data real-time analysis. In *2014 IEEE Symposium on Computational Intelligence in Big Data (CIBD)*, pages 1–8, Dec 2014.
- [21] Arati Baliga, Nitesh Solanki, Shubham Verekar, Amol Pednekar, Pandurang Kamat, and Siddhartha Chatterjee. Performance characterization of hyper-ledger fabric. In *2018 Crypto Valley Conference on Blockchain Technology (CVCBT)*, pages 65–74. IEEE, 2018.
- [22] Andrew G Barto, Richard S Sutton, and Charles W Anderson. Neuron-like adaptive elements that can solve difficult learning control problems. *IEEE Transactions on Systems, Man, and Cybernetics*, (5):834–846, 1983.
- [23] Adam L Beberg, Daniel L Ensign, Guha Jayachandran, Siraj Khaliq, and Vijay S Pande. Folding@Home: Lessons from eight years of volunteer distributed computing. In *2009 IEEE International Symposium on Parallel & Distributed Processing*, pages 1–8. IEEE, 2009.
- [24] Marc G Bellemare, Yavar Naddaf, Joel Veness, and Michael Bowling. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 47:253–279, 2013.
- [25] Richard Bellman. Dynamic programming. Technical report, RAND corp, Santa Monica, CA, 1956.
- [26] Richard E Bellman and Stuart E Dreyfus. *Applied dynamic programming*. Princeton University Press, 2015.
- [27] Alberto Bemporad, Manfred Morari, Vivek Dua, and Efstratios N Pistikopoulos. The explicit linear quadratic regulator for constrained systems. *Automatica*, 38(1):3–20, 2002.
- [28] James Bennett, Stan Lanning, et al. The Netflix Prize. In *Proceedings of KDD cup and workshop*, volume 2007, page 35. New York, 2007.

- [29] Oliver Bent and Sekou Remy. Indaba Malaria Hackathon 2019. <http://www.deeplearningindaba.com/hackathon-2019.html>, 2019.
- [30] Oliver Bent and Sekou Remy. KDD Cup 2019. <https://www.kdd.org/kdd2019/kdd-cup>, 2019.
- [31] Oliver Bent and Sekou Remy. NeurIPS Live Malaria Challenge 2019. <https://neurips.cc/Conferences/2019/CallForCompetitions>, 2019.
- [32] Oliver Bent, Sekou L Remy, Stephen Roberts, and Aisha Walcott-Bryant. Novel Exploration Techniques (NETs) for Malaria Policy Interventions. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18)*, pages 7735–7740, 2018.
- [33] James Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. Algorithms for hyper-parameter optimization. *Advances in neural information processing systems*, 24:2546–2554, 2011.
- [34] Donald A Berry, Robert W Chen, Alan Zame, David C Heath, and Larry A Shepp. Bandit problems with infinitely many arms. *The Annals of Statistics*, pages 2103–2116, 1997.
- [35] Anna Bershteyn, Jaline Gerardin, Daniel Bridenbecker, Christopher W Lorton, Jonathan Bloedow, Robert S Baker, Guillaume Chabot-Couture, Ye Chen, Thomas Fischle, Kurt Frey, et al. Implementation and applications of EMOD, an individual-based multi-disease modeling platform. *Pathogens and disease*, 76(5):fty059, 2018.
- [36] Bishwaranjan Bhattacharjee, Scott Boag, Chandani Doshi, Parijat Dube, Ben Herta, Vatche Ishakian, K. R. Jayaram, Rania Khalaf, Avesh Krishna, Y. B. Li, Vinod Muthusamy, Ruchir Puri, Yufei Ren, Florian Rosenberg, Seetharami R. Seelam, Yi Wang, J. M. Zhang, and L. Zhang. IBM Deep learning service. *IBM Journal of Research and Development*, 61(4):10, 2017.
- [37] David M Blei, Alp Kucukelbir, and Jon D McAuliffe. Variational inference: A review for statisticians. *Journal of the American statistical Association*, 112(518):859–877, 2017.
- [38] Nelson Bore, Andrew Kinai, Juliet Mutahi, David Kaguma, Fred Otieno, Sekou L. Remy, and Komminist Weldemariam. On using blockchain based

- workflows. In *IEEE International Conference on Blockchain and Cryptocurrency, ICBC 2019, Seoul, Korea (South), May 14-17, 2019*, pages 112–116, 2019.
- [39] Nelson Kibichii Bore, Ravi Kiran Raman, Isaac M. Markus, Sekou L. Remy, Oliver Bent, Michael Hind, Eleftheria Kyriaki Pissadaki, Biplav Srivastava, Roman Vaculín, Kush R. Varshney, and Komminist Weldemariam. Promoting distributed trust in machine learning and computational simulation. In *IEEE International Conference on Blockchain and Cryptocurrency, ICBC 2019, Seoul, Korea (South), May 14-17, 2019*, pages 311–319, 2019.
- [40] G.E.P. Box. Robustness in the strategy of scientific model building. In Robert L. Launer and Graham N. Wilkinson, editors, *Robustness in Statistics*, pages 201 – 236. Academic Press, 1979.
- [41] Oliver J Brady, H. Charles J. Godfray, Andrew J Tatem, Peter W Gething, Justin M Cohen, F. Ellis McKenzie, T. Alex Perkins, Robert C Reiner, Lucy S Tusting, Marianne E Sinka, Catherine L Moyes, Philip A Eckhoff, Thomas W Scott, Steven W Lindsay, Simon I Hay, and David L Smith. Vectorial capacity and vector control: Reconsidering sensitivity to parameters for malaria elimination. *Transactions of the Royal Society of Tropical Medicine and Hygiene*, 110(2):107–117, 2016.
- [42] Oliver J Brady, Hannah C Slater, Peter Pemberton-Ross, Edward Wenger, Richard J Maude, Azra C Ghani, Melissa A Penny, Jaline Gerardin, Lisa J White, Nakul Chitnis, et al. Role of mass drug administration in elimination of plasmodium falciparum malaria: a consensus modelling study. *The Lancet Global Health*, 5(7):e680–e687, 2017.
- [43] Fred Brauer. Compartmental models in epidemiology. In *Mathematical epidemiology*, pages 19–79. Springer, 2008.
- [44] Leo Breiman, Jerome Friedman, Charles J Stone, and Richard A Olshen. *Classification and regression trees*. CRC press, 1984.
- [45] Olivier Jt Briët, Melissa a Penny, Diggory Hardy, Taiwo S Awolola, Wim Van Bortel, Vincent Corbel, Roch K Dabiré, Josiane Etang, Benjamin G Koudou, Patrick K Tungu, and Nakul Chitnis. Effects of pyrethroid resistance on the cost effectiveness of a mass distribution of long-lasting insecticidal nets: a modelling study. *Malaria journal*, 12(1):77, 2013.

- [46] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.
- [47] Bernd Brügmann. Monte carlo go. Technical report, Citeseer, 1993.
- [48] Sébastien Bubeck and Nicolo Cesa-Bianchi. Regret analysis of stochastic and nonstochastic Multi-Armed Bandit problems. *arXiv preprint arXiv:1204.5721*, 2012.
- [49] Murray Campbell, A Joseph Hoane Jr, and Feng-hsiung Hsu. Deep Blue. *Artificial intelligence*, 134(1-2):57–83, 2002.
- [50] Richard Carter and Kamini N Mendis. Evolutionary and historical aspects of the burden of malaria. *Clinical microbiology reviews*, 15(4):564–594, 2002.
- [51] Simon Chan, Thomas Stone, Kit Pang Szeto, and Ka Hou Chan. Predictionio: a distributed Machine learning server for practical software development. In Qi He, Arun Iyengar, Wolfgang Nejdl, Jian Pei, and Rajeev Rastogi, editors, *22nd ACM International Conference on Information and Knowledge Management, CIKM'13, San Francisco, CA, USA, October 27 - November 1, 2013*, pages 2493–2496. ACM, 2013.
- [52] Nakul Chitnis, James M Hyman, and Jim M Cushing. Determining important parameters in the spread of malaria through the sensitivity analysis of a mathematical model. *Bulletin of mathematical biology*, 70(5):1272, 2008.
- [53] John Chodera, Alpha A Lee, Nir London, and Frank von Delft. Crowdsourcing drug discovery for pandemics. *Nature Chemistry*, 12(7):581–581, 2020.
- [54] Emile Contal, David Buffoni, Alexandre Robicquet, and Nicolas Vayatis. Parallel Gaussian process optimization with upper confidence bound and pure exploration. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 8188 LNAI(PART 1):225–240, 2013.
- [55] Dan Cornford, Ian T Nabney, and Christopher KI Williams. Modelling frontal discontinuities in wind fields. *Journal of nonparametric statistics*, 14(1-2):43–58, 2002.

- [56] Rémi Coulom. Efficient selectivity and backup operators in Monte Carlo tree search. In *International conference on computers and games*, pages 72–83. Springer, 2006.
- [57] Francis Eg Cox. History of the discovery of the malaria parasites and their vectors. *Parasites and Vectors*, 3(1):5, 2010.
- [58] Thierry Crestaux, Olivier Le Maitre, and Jean-Marc Martinez. Polynomial chaos expansion for sensitivity analysis. *Reliability Engineering & System Safety*, 94(7):1161–1172, 2009.
- [59] Mary-Ann Davies et al. HIV and risk of Covid-19 death: a population cohort study from the Western Cape Province, South Africa. *MedRxiv*, 2020.
- [60] Peter Dayan and Geoffrey E Hinton. Using expectation-maximization for reinforcement learning. *Neural Computation*, 9(2):271–278, 1997.
- [61] Thomas Dean, Ken Basye, and John Shewchuk. Chapter 3 - reinforcement learning for planning and control. In Steven Minton, editor, *Machine Learning Methods for Planning*, pages 67 – 92. Morgan Kaufmann, 1993.
- [62] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.
- [63] Thomas G Dietterich. Ensemble methods in machine learning. In *International workshop on multiple classifier systems*, pages 1–15. Springer, 2000.
- [64] Christos Dimitrakakis, Guangliang Li, and Nikolaos Tziortziotis. The Reinforcement Learning Competition. 2014.
- [65] Nicola Dragoni, Saverio Giallorenzo, Alberto Lluch Lafuente, Manuel Mazzara, Fabrizio Montesi, Ruslan Mustafin, and Larisa Safina. Microservices: yesterday, today, and tomorrow. *Present and ulterior software engineering*, pages 195–216, 2017.
- [66] Gideon Dror, Noam Koenigstein, Yehuda Koren, and Markus Weimer. The Yahoo! music dataset and KDD-cup 2011. In *Proceedings of KDD Cup 2011*, pages 3–18. PMLR, 2012.

- [67] Yan Duan, Xi Chen, Rein Houthooft, John Schulman, and Pieter Abbeel. Benchmarking Deep Reinforcement learning for continuous control. In *International Conference on Machine Learning*, pages 1329–1338, 2016.
- [68] Philip A Eckhoff and Edward A Wenger. The EMOD Individual-Based Model. *Spatial Agent-Based Simulation Modeling in Public Health: Design, Implementation, and Applications for Malaria Epidemiology*, pages 185–208, 2016.
- [69] Tom Erez, Yuval Tassa, and Emanuel Todorov. Infinite horizon model predictive control for nonlinear periodic tasks. *Manuscript under review*, 4, 2011.
- [70] Eyal Even-Dar, Yishay Mansour, and Peter Bartlett. Learning rates for Q-learning. *Journal of machine learning Research*, 5(1), 2003.
- [71] Casey Ferris, Benoit Raybaud, and Gregory Madey. Openmalaria and emod: a case study on model alignment. In *Proceedings of the Conference on Summer Computer Simulation*, pages 1–9, 2015.
- [72] Richard P Feynman. Simulating physics with computers. *Int. J. Theor. Phys*, 21(6/7), 1982.
- [73] Ronald Aylmer Fisher. Design of experiments. *Br Med J*, 1(3923):554–554, 1936.
- [74] Colby T Ford and Daniel Janies. Ensemble machine learning modeling for the prediction of artemisinin resistance in malaria. *F1000Research*, 9(62):62, 2020.
- [75] Peter I Frazier. A tutorial on Bayesian Optimization. *arXiv preprint arXiv:1807.02811*, 2018.
- [76] Marcia Frellick. Flattening the Curve: Viral Graphic Shows COVID-19 Containment Needs, 2020.
- [77] Jerome H Friedman. An overview of predictive learning and function approximation. In *From statistics to neural networks*, pages 1–61. Springer, 1994.
- [78] Manoj Gambhir and Chathurika Hettiarachchige. Making sense of consensus: comparative modelling of malaria interventions. *The Lancet Global Health*, 5(7):e638 – e639, 2017.
- [79] MG Garner, SA Hamilton, et al. Principles of epidemiological modelling. *Revue Scientifique et Technique-OIE*, 30(2):407, 2011.

- [80] Yiyue Ge, Tingzhong Tian, Sulin Huang, Fangping Wan, Jingxin Li, Shuya Li, Hui Yang, Lixiang Hong, Nian Wu, Enming Yuan, et al. A data-driven drug repositioning framework discovered a potential therapeutic agent targeting Covid-19. *BioRxiv*, 2020.
- [81] Sylvain Gelly and David Silver. Combining online and offline knowledge in UCT. In *Proceedings of the 24th International Conference on Machine Learning, ICML '07*, page 273280, New York, NY, USA, 2007. Association for Computing Machinery.
- [82] Sylvain Gelly and Yizao Wang. Exploration exploitation in Go: UCT for Monte-Carlo Go. 2006.
- [83] Graham Glen and Kristin Isaacs. Estimating Sobol sensitivity indices using correlations. *Environmental Modelling & Software*, 37:157–166, 2012.
- [84] David E Goldberg and Kalyanmoy Deb. A Comparative Analysis of Selection Schemes Used in Genetic Algorithms. *Foundations of Genetic Algorithms*, 1:69–93, 1991.
- [85] Pam Frost Gorder. Computational epidemiology. *Computing in science & engineering*, 12(1):4–6, 2009.
- [86] Robert Grande, Thomas Walsh, and Jonathan How. Sample efficient Reinforcement learning with Gaussian processes. In *International Conference on Machine Learning*, pages 1332–1340. PMLR, 2014.
- [87] Miguel Grinberg. *Flask web development: developing web applications with Python*. O’Reilly Media, Inc., 2018.
- [88] Resign Gunda and Moses John Chimbari. Cost-effectiveness analysis of malaria interventions using disability adjusted life years: A systematic review, jul 2017.
- [89] Thomas Hale and Samuel Webster. Oxford Covid-19 government response tracker. 2020.
- [90] Mark A Hall. Correlation-based feature selection of discrete and numeric class machine learning. 2000.
- [91] James Hannan. Approximation to Bayes risk in repeated play. *Contributions to the Theory of Games*, 3:97–139, 1957.

- [92] Peter Henderson, Riashat Islam, Philip Bachman, Joelle Pineau, Doina Precup, and David Meger. Deep Reinforcement learning that matters. *arXiv preprint arXiv:1709.06560*, 2017.
- [93] John H Holland. Genetic Algorithms - Computer programs that "evolve" in ways that resemble natural selection can solve complex problems even their creators do not fully understand, 1992.
- [94] Toshimitsu Homma and Andrea Saltelli. Importance measures in global sensitivity analysis of nonlinear models. *Reliability Engineering & System Safety*, 52(1):1–17, 1996.
- [95] Hyperledger. Hyperledger caliper. <https://hyperledger.github.io/caliper/>.
- [96] IBM. Behind the Architecture of Hyperledger Fabric. <https://www.ibm.com/blogs/research/2018/02/architecture-hyperledger-fabric/>.
- [97] Robert A Jacobs, Michael I Jordan, Steven J Nowlan, and Geoffrey E Hinton. Adaptive mixtures of local experts. *Neural computation*, 3(1):79–87, 1991.
- [98] Douglas M Jacobsen and Richard Shane Canon. Contain this, unleashing docker for HPC.
- [99] Leslie Pack Kaelbling, Michael L Littman, and Anthony R Cassandra. Planning and acting in partially observable stochastic domains. *Artificial intelligence*, 101(1-2):99–134, 1998.
- [100] Rudolf Emil Kalman et al. Contributions to the theory of optimal control. *Bol. soc. mat. mexicana*, 5(2):102–119, 1960.
- [101] KDD. KDD Cup Archives. <https://www.kdd.org/kdd-cup>, 1997.
- [102] Matt J Keeling and Pejman Rohani. Estimating spatial coupling in epidemiological systems: a mechanistic approach. *Ecology Letters*, 5(1):20–29, 2002.
- [103] William Ogilvy Kermack, A. G. McKendrick, and Gilbert Thomas Walker. A contribution to the mathematical theory of epidemics. *Proceedings of the Royal Society of London. Series A, Containing Papers of a Mathematical and Physical Character*, 115(772):700–721, 1927.

- [104] A Keshavarzi Arshadi, J Webb, M Salem, E Cruz, S Calad-Thomson, N Ghadirian, J Collins, E Diez-Cecilia, B Kelly, H Goodarzi, et al. Artificial intelligence for Covid-19 drug discovery and vaccine development. *Artif. Intell.*, 3:65, 2020.
- [105] Khimya Khetarpal, Zafarali Ahmed, Andre Cianflone, Riashat Islam, and Joelle Pineau. Re-evaluate: Reproducibility in evaluating Reinforcement learning algorithms. 2018.
- [106] Łukasz Kidziński, Sharada Prasanna Mohanty, Carmichael F Ong, Zhewei Huang, Shuchang Zhou, Anton Pechenko, Adam Stelmaszczyk, Piotr Jarosik, Mikhail Pavlov, Sergey Kolesnikov, et al. Learning to run challenge solutions: Adapting Reinforcement learning methods for neuromusculoskeletal environments. In *The NIPS’17 Competition: Building Intelligent Systems*, pages 121–153. Springer, 2018.
- [107] Łukasz Kidzinski, Carmichael Ong, Sharada Prasanna Mohanty, Jennifer Hicks, Sean Carroll, Bo Zhou, Hongsheng Zeng, Fan Wang, Rongzhong Lian, Hao Tian, et al. Artificial intelligence for prosthetics: Challenge solutions. *The NeurIPS’18 Competition: From Machine Learning to Intelligent Conversations*, page 69, 2019.
- [108] Gerry F Killeen and Thomas A Smith. Exploring the contributions of bed nets, cattle, insecticides and excitorepellency to malaria control: a deterministic model of mosquito host-seeking behaviour and mortality. *Transactions of the Royal Society of Tropical Medicine and Hygiene*, 101(9):867–880, 2007.
- [109] Ron Kohavi, Carla E Brodley, Brian Frasca, Llew Mason, and Zijian Zheng. KDD-cup 2000 organizers’ report: Peeling the onion. *Acm Sigkdd Explorations Newsletter*, 2(2):86–93, 2000.
- [110] Sotiris B Kotsiantis, I Zaharakis, and P Pintelas. Supervised machine learning: A review of classification techniques. *Emerging artificial intelligence applications in computer engineering*, 160(1):3–24, 2007.
- [111] Nane Kratzke. Volunteer down: How Covid-19 created the largest idling super-computer on earth. *Future Internet*, 12(6), 2020.

- [112] Sergei Kucherenko, Balazs Feil, Nilay Shah, and Wolfgang Mauntz. The identification of model effective dimensions using global sensitivity analysis. *Reliability Engineering & System Safety*, 96(4):440–449, 2011.
- [113] Marin Kukovacec, Toni Kukurin, and Marin Vernier. Personalized medicine: Redefining cancer treatment. *Text Analysis and Retrieval 2018 Course Project Reports*, page 69.
- [114] Heidi J Larson and Isaac Ghinai. Lessons from polio eradication. *Nature*, 473(7348):446–447, 2011.
- [115] Andrew B Lawson. *Statistical methods in spatial epidemiology*. John Wiley & Sons, 2013.
- [116] Feng Lin, Kumar Muthuraman, and Mark Lawley. An optimal control theory approach to non-pharmaceutical interventions. *BMC infectious diseases*, 10(1):1–13, 2010.
- [117] Michael Lederman Littman. *Algorithms for sequential decision making*. Brown University Providence, RI, 1996.
- [118] Alan D Lopez and Christopher CJL Murray. The global burden of disease, 1990–2020. *Nature medicine*, 4(11):1241–1243, 1998.
- [119] Grace Manu, Ellen Abrafi Boamah-Kaali, Lawrence Gyabaa Febir, Emmanuel Ayipah, Seth Owusu-Agyei, and Kwaku Poku Asante. Low Utilization of Insecticide-Treated Bed Net among Pregnant Women in the Middle Belt of Ghana. *Malaria Research and Treatment*, 2017:1–7, jul 2017.
- [120] Andrei Andreevich Markov. The theory of algorithms. *Trudy Matematicheskogo Instituta Imeni VA Steklova*, 42:3–375, 1954.
- [121] Amandine Marrel, Bertrand Iooss, Beatrice Laurent, and Olivier Roustant. Calculations of Sobol indices for the Gaussian process metamodel. *Reliability Engineering & System Safety*, 94(3):742–751, 2009.
- [122] Esteban Ortiz-Ospina Max Roser, Hannah Ritchie and Joe Hasell. Coronavirus pandemic (Covid-19). *Our World in Data*, 2020. <https://ourworldindata.org/coronavirus>.

- [123] Kevin A McCarthy, Edward A Wenger, Grace H Huynh, and Philip A Eckhoff. Calibration of an intrahost malaria model and parameter ensemble evaluation of a pre-erythrocytic vaccine. *Malaria journal*, 14(1):1–10, 2015.
- [124] A Ñ McLean and Roy M Anderson. Measles in developing countries part i. epidemiological parameters and patterns. *Epidemiology & Infection*, 100(1):111–133, 1988.
- [125] Andreas Miliadis Argeitis and Tamara Kurdyakova. Analytical calculation of Sobol sensitivity indices for Gaussian processes with a squared exponential covariance function. Workingpaper, 2018.
- [126] Sudhanshu K Mishra. Global optimization by differential evolution and particle swarm methods: Evaluation on some benchmark functions. *Available at SSRN 933827*, 2006.
- [127] Andrew William Moore. Efficient memory-based learning for robot control. 1990.
- [128] Sean Moore, Sourya Shrestha, Kyle W Tomlinson, and Holly Vuong. Predicting the effect of climate change on African trypanosomiasis: integrating epidemiology with parasite and vector biology. *Journal of the Royal Society Interface*, 9(70):817–830, 2012.
- [129] Trevor Mundel. how do you decide what to focus on, 2018. <https://www.gatesfoundation.org/TheOptimist/Articles/how-do-you-decide-what-to-invest-in>.
- [130] Christopher J. L. Murray and Alan D. Lopez. The global burden of disease: a comprehensive assessment of mortality and disability from diseases, injuries and risk factors in 1990 and projected to 2010. *Harvard University Press*, 1:1–35, 1996.
- [131] Christopher JL Murray and Alan D Lopez. Global health statistics: Global burden of disease and injury series. *Volumes I and II. Boston: Harvard School of Public Health*, 1996.
- [132] José A Nájera, Matiana González-Silva, and Pedro L Alonso. Some lessons for the future from the Global Malaria Eradication Programme (1955-1969). *PLoS medicine*, 8(1):e1000412, jan 2011.

- [133] Jacques Naude, Bruce Mellado, Joshua Choma, Fabio Correa, Salah Dahbi, Barry Dwolatzky, Leslie Dwolatzky, Kentaro Hayasi, Benjamin Lieberman, Caroline Maslo, et al. Worldwide Effectiveness of Various Non-Pharmaceutical Intervention Control Strategies on the Global Covid-19 Pandemic: A linearised control model. *medRxiv*, 2020.
- [134] INDEPTH Network. INDEPTH model life tables for sub-Saharan Africa. *Aldershot (England): INDEPTH Network*, 2004.
- [135] Van Bach Nguyen, Belaid Mohamed Karim, Bao Long Vu, Jörg Schlötterer, and Michael Granitzer. Policy learning for malaria control. *arXiv preprint arXiv:1910.08926*, 2019.
- [136] Maria Nicola, Zaid Alsafi, Catrin Sohrabi, Ahmed Kerwan, Ahmed Al-Jabir, Christos Iosifidis, Maliha Agha, and Riaz Agha. The socio-economic implications of the coronavirus pandemic (Covid-19): A review. *International journal of surgery (London, England)*, 78:185, 2020.
- [137] Jeremy E Oakley and Anthony O’Hagan. Probabilistic sensitivity analysis of complex models: a Bayesian approach. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 66(3):751–769, 2004.
- [138] Sophie Ochmann. Polio. *Our World in Data*, 2017. <https://ourworldindata.org/polio>.
- [139] Jeroen Ooms. The opencpu system: Towards a universal interface for scientific computing through separation of concerns. *CoRR*, abs/1406.4806, 2014.
- [140] OpenAI. OpenAI Gym Environments. <https://gym.openai.com/envs/>, accessed 2019.
- [141] Michael A Osborne, Roman Garnett, and Stephen J Roberts. Gaussian Processes for Global Optimization. *3rd International Conference on Learning and Intelligent Optimization LION3*, (x):1–15, 2009.
- [142] Emilio Pampana et al. A textbook of malaria eradication. *A textbook of malaria eradication. 2nd edition.*, 1969.
- [143] Melissa A Penny, Robert Verity, Caitlin A Bever, Christophe Sauboin, Katya Galactionova, Stefan Flasche, Michael T White, Edward A Wenger, Nicolas

- Van de Velde, Peter Pemberton-Ross, et al. Public health impact and cost-effectiveness of the RTS, S/AS01 malaria vaccine: a systematic comparison of predictions from four mathematical models. *The Lancet*, 387(10016):367–375, 2016.
- [144] William D Penny, Dirk Husmeier, and Stephen J Roberts. Covariance-based weighting for optimal combination of model predictions. 1999.
- [145] Joel Perras. *Flask Blueprints*. Packt Publishing Ltd, 2015.
- [146] Joelle Pineau, Geoffrey Gordon, and Sebastian Thrun. Anytime point-based approximations for large POMDPs. *Journal of Artificial Intelligence Research*, 27:335–380, 2006.
- [147] Loren Kerry Platzman. *Finite memory estimation and control of finite probabilistic systems*. PhD thesis, Massachusetts Institute of Technology, 1977.
- [148] Doina Precup. Eligibility traces for off-policy policy evaluation. *Computer Science Department Faculty Publication Series*, page 80, 2000.
- [149] Defense Acquisition University (US). Press. *Test and Evaluation Management Guide*. Defense Acquisition University Press, 2001.
- [150] Franck Prugnolle, Patrick Durand, Benjamin Ollomo, Linda Duval, Frédéric Arieu, Céline Arnathau, Jean Paul Gonzalez, Eric Leroy, and François Renaud. A fresh look at the origin of *Plasmodium falciparum*, the most malignant malaria agent. *PLoS Pathogens*, 7(2), 2011.
- [151] Joseph Pryce, Marty Richardson, and Christian Lengeler. Insecticide-treated nets for preventing malaria. *Cochrane Database of Systematic Reviews*, (11), 2018.
- [152] Zhaozhi Qian, Ahmed M. Alaa, and Mihaela van der Schaar. When and how to lift the lockdown? Global Covid-19 scenario analysis and policy assessment using compartmental Gaussian processes, 2020.
- [153] Troy Quast, Ross Andel, Sean Gregory, and Eric A Storch. Years of life lost associated with Covid-19 deaths in the United States. *Journal of public health (Oxford, England)*, 2020.

- [154] Ravi Kiran Raman, Kush R. Varshney, Roman Vaculín, Nelson Kibichii Bore, Sekou L. Remy, Eleftheria Kyriaki Pissadaki, and Michael Hind. Constructing and compressing frames in blockchain-based verifiable multi-party computation. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2019, Brighton, United Kingdom, May 12-17, 2019*, pages 7500–7504, 2019.
- [155] Carl E Rasmussen and Zoubin Ghahramani. Infinite mixtures of Gaussian process experts. In *Advances in neural information processing systems*, pages 881–888, 2002.
- [156] Carl Edward Rasmussen. Gaussian processes in machine learning. In *Summer School on Machine Learning*, pages 63–71. Springer, 2003.
- [157] Evan L Ray, Nutchua Wattanachit, Jarad Niemi, Abdul Hannan Kanji, Katie House, Estee Y Cramer, Johannes Bracher, Andrew Zheng, Teresa K Yamana, Xinyue Xiong, et al. Ensemble forecasts of coronavirus disease 2019 (Covid-19) in the US. *MedRxiv*, 2020.
- [158] Sekou L Remy and Oliver Bent. A global health gym environment for RL applications. In *NeurIPS 2019 Competition and Demonstration Track*, pages 253–261. PMLR, 2020.
- [159] Albert Reuther, Chansup Byun, William Arcand, David Bestor, Bill Bergeron, Matthew Hubbell, Michael Jones, Peter Michaleas, Andrew Prout, Antonio Rosa, and Jeremy Kepner. Scalable system scheduling for HPC and big data. *Journal of Parallel and Distributed Computing*, 111:76 – 92, 2018.
- [160] Daniel Bernoulli Reviewed and Sally Blower. An attempt at a new analysis of the mortality caused by smallpox and of the advantages of inoculation to prevent it. *Rev. Med. Virol*, 14:275–288, 2004.
- [161] Mauro Ribeiro, Katarina Grolinger, and Miriam A. M. Capretz. Mlaas: Machine learning as a service. In Tao Li, Lukasz A. Kurgan, Vasile Palade, Randy Goebel, Andreas Holzinger, Karin Verspoor, and M. Arif Wani, editors, *14th IEEE International Conference on Machine Learning and Applications, ICMLA 2015, Miami, FL, USA, December 9-11, 2015*, pages 896–902. IEEE, 2015.

- [162] Lisa A Robinson, James K Hammitt, and Lucy O’Keeffe. Valuing mortality risk reductions in global benefit-cost analysis. *Journal of Benefit-Cost Analysis*, 10(S1):15–50, 2019.
- [163] David Rohde, Stephen Bonner, Travis Dunlop, Flavian Vasile, and Alexandros Karatzoglou. RecoGym: A Reinforcement Learning Environment for the Problem of Product Recommendation in Online Advertising. Technical report, 2018.
- [164] Max Roser and Hannah Ritchie. Malaria. *Our World in Data*, 2013. <https://ourworldindata.org/malaria>.
- [165] Max Roser and Hannah Ritchie. HIV / AIDS. *Our World in Data*, 2018. <https://ourworldindata.org/hiv-aids>.
- [166] Max Roser, Hannah Ritchie, Esteban Ortiz-Ospina, and Joe Hasell. Coronavirus pandemic (Covid-19). *Our World in Data*, 2020.
- [167] Ronald Ross. *The prevention of malaria*. London, Murray, 1911.
- [168] Saharon Rosset and Aron Inger. KDD-cup 99: knowledge discovery in a charitable organization’s donor database. *ACM SIGKDD Explorations Newsletter*, 1(2):85–90, 2000.
- [169] Omer Sagi and Lior Rokach. Ensemble learning: A survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 8(4):e1249, 2018.
- [170] Andrea Saltelli. Making best use of model evaluations to compute sensitivity indices. *Computer physics communications*, 145(2):280–297, 2002.
- [171] Andrea Saltelli, Gabriele Bammer, Isabelle Bruno, Erica Charters, Monica Di Fiore, Emmanuel Didier, Wendy Nelson Espeland, John Kay, Samuele Lo Piano, Deborah Mayo, et al. Five ways to ensure that models serve society: a manifesto, 2020.
- [172] Claude Sammut and Geoffrey I. Webb, editors. *A/B Testing*, pages 1–1. Springer US, Boston, MA, 2017.
- [173] Arthur L Samuel. Some studies in machine learning using the game of checkers. *IBM Journal of research and development*, 3(3):210–229, 1959.

- [174] Leticia San Martín-Rodríguez, Marie-Dominique Beaulieu, Danielle D’Amour, and Marcela Ferrada-Videla. The determinants of successful collaboration: a review of theoretical and empirical studies. *Journal of interprofessional care*, 19(sup1):132–147, 2005.
- [175] Robin Schaefer, Simon Gregson, Elizabeth Fearon, Bernadette Hensen, Timothy B Hallett, and James R Hargreaves. HIV prevention cascades: a unifying framework to replicate the successes of treatment cascades. *The Lancet HIV*, 6(1):e60–e66, 2019.
- [176] Nick Scott, S Azfar Hussain, Rowan Martin-Hughes, Freya JI Fowkes, Cliff C Kerr, Ruth Pearson, David J Kedziora, Madhura Killedar, Robyn M Stuart, and David P Wilson. Maximizing the impact of malaria funding through allocative efficiency: using the right interventions in the right locations. *Malaria journal*, 16(1):1–14, 2017.
- [177] Bobak Shahriari, Kevin Swersky, Ziyu Wang, Ryan P Adams, and Nando De Freitas. Taking the human out of the loop: A review of Bayesian optimization. *Proceedings of the IEEE*, 104(1):148–175, 2015.
- [178] Rima Shretta, Anton L. V. Avanceña, and Arian Hatefi. The economics of malaria control and elimination: a systematic review. *Malaria Journal*, 15(1):593, dec 2016.
- [179] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of Go with deep neural networks and tree search. *nature*, 529(7587):484–489, 2016.
- [180] David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dhharshan Kumaran, Thore Graepel, et al. A general Reinforcement learning algorithm that masters chess, shogi, and Go through self-play. *Science*, 362(6419):1140–1144, 2018.
- [181] David Silver and Joel Veness. Monte-Carlo planning in large POMDPs. In J. Lafferty, C. Williams, J. Shawe-Taylor, R. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems*, volume 23, pages 2164–2172. Curran Associates, Inc., 2010.

- [182] David L Smith, Katherine E Battle, Simon I Hay, Christopher M Barker, Thomas W Scott, and F Ellis McKenzie. Ross, Macdonald, and a theory for the dynamics and control of mosquito-transmitted pathogens. *PLoS pathog*, 8(4):e1002588, 2012.
- [183] Thomas Smith, Gerry F Killeen, Nicolas Maire, Amanda Ross, Louis Molineaux, Fabrizio Tediosi, Guy Hutton, Jürg Utzinger, Klaus Dietz, and Marcel Tanner. Mathematical modeling of the impact of malaria vaccines on the clinical epidemiology and natural history of *Plasmodium falciparum* malaria: Overview. *The American journal of tropical medicine and hygiene*, 75(2-suppl):1–10, 2006.
- [184] Ilya M Sobol. Global sensitivity indices for nonlinear mathematical models and their Monte Carlo estimates. *Mathematics and computers in simulation*, 55(1-3):271–280, 2001.
- [185] Robert C Spear. Large simulation models: calibration, uniqueness and goodness of fit. *Environmental Modelling & Software*, 12(2-3):219–228, 1997.
- [186] Niranjan Srinivas, Andreas Krause, Sham M Kakade, and Matthias Seeger. Gaussian process optimization in the Bandit setting: No Regret and experimental design. *arXiv preprint arXiv:0912.3995*, 2009.
- [187] Malcolm Strens. A Bayesian framework for Reinforcement learning. In *ICML*, volume 2000, pages 943–950, 2000.
- [188] Erin M. Stuckey, Jennifer Stevenson, Katya Galactionova, Amrish Y. Baidjoe, Teun Bousema, Wycliffe Odongo, Simon Kariuki, Chris Drakeley, Thomas A. Smith, Jonathan Cox, and Nakul Chitnis. Modeling the cost effectiveness of malaria control interventions in the highlands of western Kenya. *PLoS ONE*, 9(10), 2014.
- [189] Erin M Stuckey, Jennifer C Stevenson, Mary K Cooke, Chrispin Owaga, Elizabeth Marube, George Oando, Diggory Hardy, Chris Drakeley, Thomas A Smith, Jonathan Cox, and Nakul Chitnis. Simulation of malaria epidemiology and control in the highlands of western Kenya. *Malaria Journal*, 11(1):357, 2012.
- [190] Parthasarathy Suryanarayanan, Ching-Huei Tsou, Ananya Poddar, Diwakar Mahajan, Bharath Dandala, Piyush Madan, Anshul Agrawal, Charles Wachira, Osebe Mogaka Samuel, Osnat Bar-Shira, et al. Wntrac: Artificial intelligence

assisted tracking of non-pharmaceutical interventions implemented worldwide for covid-19. *arXiv preprint arXiv:2009.07057*, 2020.

- [191] Richard S Sutton. Temporal credit assignment in Reinforcement learning. 1985.
- [192] Richard S Sutton. Integrated architectures for learning, planning, and reacting based on approximating dynamic programming. In *Machine learning proceedings 1990*, pages 216–224. Elsevier, 1990.
- [193] Richard S Sutton. Dyna, an integrated architecture for learning, planning, and reacting. *ACM Sigart Bulletin*, 2(4):160–163, 1991.
- [194] Richard S Sutton and Andrew G Barto. Reinforcement learning: an introduction cambridge. *MA: MIT Press.[Google Scholar]*, 1998.
- [195] Richard S Sutton, David Mcallester, Satinder Singh, and Yishay Mansour. Policy Gradient Methods for Reinforcement Learning with Function Approximation. In *Advances in Neural Information Processing Systems 12*, pages 1057–1063, 1999.
- [196] Csaba Szepesvári. Algorithms for Reinforcement learning. *Synthesis lectures on artificial intelligence and machine learning*, 4(1):1–103, 2010.
- [197] Mahbod Tavallaei, Ebrahim Bagheri, Wei Lu, and Ali A Ghorbani. A detailed analysis of the KDD cup 99 data set. In *2009 IEEE symposium on computational intelligence for security and defense applications*, pages 1–6. IEEE, 2009.
- [198] Fabrizio Tediosi, Nicolas Maire, Melissa Penny, Alain Studer, and Thomas a Smith. Simulation of the cost-effectiveness of malaria vaccines. *Malaria journal*, 8:127, 2009.
- [199] Parth Thakkar, Senthil Nathan, and Balaji Viswanathan. Performance Benchmarking and Optimizing Hyperledger Fabric Blockchain Platform. *CoRR*, abs/1805.11390, 2018.
- [200] The Linux Foundation. Hyperledger Blockchain Performance Metrics White Paper - Hyperledger. <https://www.hyperledger.org/resources/publications/blockchain-performance-metrics\#definitions>.

- [201] Sebastian Thrun, Mike Montemerlo, Hendrik Dahlkamp, David Stavens, Andrei Aron, James Diebel, Philip Fong, John Gale, Morgan Halpenny, Gabriel Hoffmann, et al. Stanley: The robot that won the DARPA Grand Challenge. *Journal of field Robotics*, 23(9):661–692, 2006.
- [202] Halidou Tinto, Walter Otieno, Samwel Gesase, Hermann Sorgho, Lucas Otieno, Edwin Liheluka, Innocent Valéa, Valentine Sing’oei, Anangisye Malabeja, Daniel Valia, et al. Long-term incidence of severe malaria following RTS, S/AS01 vaccination in children and infants in Africa: an open-label 3-year extension study of a phase 3 randomised controlled trial. *The Lancet Infectious Diseases*, 19(8):821–832, 2019.
- [203] Michalis K Titsias, Neil Lawrence, and Magnus Rattray. Markov chain Monte Carlo algorithms for Gaussian processes. *Inference and Estimation in Probabilistic Time-Series Models*, 9, 2008.
- [204] Tinashe A Tizifa, Alinune N Kabaghe, Robert S McCann, Henk van den Berg, Michele Van Vugt, and Kamija S Phiri. Prevention efforts for malaria. *Current tropical medicine reports*, 5(1):41–50, 2018.
- [205] Volker Tresp and Michiaki Taniguchi. Combining estimators using non-constant weighting functions. In G. Tesauro, D. S. Touretzky, and T. K. Leen, editors, *Advances in Neural Information Processing Systems 7*, pages 419–426. MIT Press, 1995.
- [206] David Turner, A Wailoo, K Nicholson, N Cooper, A Sutton, and K Abrams. Systematic review and economic decision modelling for the prevention and treatment of influenza A and B. In *NIHR Health Technology Assessment programme: Executive Summaries*. NIHR Journals Library, 2003.
- [207] Jarno Vanhatalo and Aki Vehtari. Sparse log Gaussian processes via MCMC for spatial epidemiology. In *Gaussian processes in practice*, pages 73–89, 2007.
- [208] Vladimir Vapnik. *The nature of statistical learning theory*. Springer science & business media, 2013.
- [209] Sofía S Villar, Jack Bowden, and James Wason. Multi-Armed Bandit models for the optimal design of clinical trials: benefits and challenges. *Statistical science: a review journal of the Institute of Mathematical Statistics*, 30(2):199, 2015.

- [210] Charles Wachira, Sekou L Remy, Oliver Bent, Nelson Bore, Samuel Osebe, Komminist Weldemariam, and Aisha Walcott-Bryant. A platform for disease intervention planning. In *2020 IEEE International Conference on Healthcare Informatics (ICHI) - to appear*. IEEE, 2020.
- [211] Kiri Wagstaff. Machine learning that matters. *arXiv preprint arXiv:1206.4656*, 2012.
- [212] Dorothy I Wallace, Ben S Southworth, Xun Shi, Jonathan W Chipman, and Andrew K Githeko. A comparison of five malaria transmission models: benchmark tests and implications for disease control. *Malaria Journal*, 13(1):1–16, 2014.
- [213] Craig Walls. *Spring Boot in action*. Manning Publications, 2016.
- [214] Ziyu Wang, Masrour Zoghi, Frank Hutter, David Matheson, and Nando De Freitas. Bayesian Optimization in High Dimensions via Random Embeddings. *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 1778–1784, 2013.
- [215] Christopher John Cornish Hellaby Watkins. Learning from delayed rewards. 1989.
- [216] Shimon Whiteson, Brian Tanner, Matthew E Taylor, and Peter Stone. Protecting against evaluation overfitting in empirical Reinforcement learning. In *2011 IEEE symposium on adaptive dynamic programming and Reinforcement learning (ADPRL)*, pages 120–127. IEEE, 2011.
- [217] Shimon Whiteson, Brian Tanner, and Adam White. The Reinforcement learning competitions. *AI Magazine*, 31(2):81–94, 2010.
- [218] Christopher KI Williams and Carl Edward Rasmussen. Gaussian processes for regression. In *Advances in neural information processing systems*, pages 514–520, 1996.
- [219] Ronald J Williams. Simple statistical gradient-following algorithms for connectionist Reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992.
- [220] Peter Winskill, Patrick GT Walker, Jamie T Griffin, and Azra C Ghani. Modelling the cost-effectiveness of introducing the RTS,S malaria vaccine relative

- to scaling up other malaria interventions in sub-Saharan Africa. *BMJ Global Health*, 2(1):e000090, 2017.
- [221] Ian H Witten. An adaptive optimal controller for discrete-time Markov environments. *Information and control*, 34(4):286–295, 1977.
- [222] Svante Wold, Kim Esbensen, and Paul Geladi. Principal component analysis. *Chemometrics and intelligent laboratory systems*, 2(1-3):37–52, 1987.
- [223] Yuexin Wu, Yiming Yang, Hiroshi Nishiura, and Masaya Saitoh. Deep learning for epidemiological predictions. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 1085–1088, 2018.
- [224] Zunyou Wu and Jennifer M McGoogan. Characteristics of and important lessons from the coronavirus disease 2019 (Covid-19) outbreak in China: summary of a report of 72 314 cases from the Chinese center for disease control and prevention. *Jama*, 323(13):1239–1242, 2020.
- [225] Grant MA Wyper, Ricardo MA Assunção, Edoardo Colzani, Ian Grant, Juanita A Haagsma, Giske Lagerweij, Elena Von der Lippe, Scott A McDonald, Sara M Pires, Michael Porst, et al. Burden of disease methods: a guide to calculate Covid-19 disability-adjusted life years. 2020.
- [226] Deshraj Yadav, Rishabh Jain, Harsh Agrawal, Prithvijit Chattopadhyay, Taranjeet Singh, Akash Jain, Shiv Baran Singh, Stefan Lee, and Dhruv Batra. EvalAI: Towards Better Evaluation Systems for AI Agents. Technical report.
- [227] Edmund I. Yamba, Adrian M. Tompkins, Andreas H. Fink, Volker Ermert, Mbouna D. Amelie, Leonard K. Amekudzi, and Olivier J. T. Briët. Monthly Entomological Inoculation Rate Data for Studying the Seasonality of Malaria Transmission in Africa. *Data*, 5(2):31, mar 2020.
- [228] K. Yao and J. Gao. Law of large numbers for uncertain random variables. *IEEE Transactions on Fuzzy Systems*, 24(3):615–621, 2016.
- [229] William R Zame, Ioana Bica, Cong Shen, Alicia Curth, Hyun-Suk Lee, Stuart Bailey, James Weatherall, David Wright, Frank Bretz, and Mihaela van der Schaar. Machine learning for clinical trials in the era of Covid-19. *Statistics in Biopharmaceutical Research*, 12(4):506–517, 2020.

- [230] Wenjun Zhou, Taposh Dutta Roy, and Iryna Skrypnyk. The KDD Cup 2019 Report. *SIGKDD Explor. Newsl.*, 22(1):817, May 2020.