International Journal of Rail Transportation

ISSN: 2324-8378 (Print) 2324-8386 (Online) Journal homepage: https://www.tandfonline.com/loi/tjrt20

Investigating Bayesian Optimization for rail network optimization

Bob Hickish, David I. Fletcher & Robert F. Harrison

To cite this article: Bob Hickish, David I. Fletcher & Robert F. Harrison (2019): Investigating Bayesian Optimization for rail network optimization, International Journal of Rail Transportation, DOI: 10.1080/23248378.2019.1669500

To link to this article: https://doi.org/10.1080/23248378.2019.1669500

Published online: 14 Oct 2019.

INTERNATIONAL JOURNAL OF RAIL TRANSPORTATION
https://doi.org/10.1080/23248378.2019.1669500

Taylor & Francis
Taylor & Francis Group

OPEN ACCESS Check for updates

Investigating Bayesian Optimization for rail network optimization

Bob Hickish[a], David I. Fletcher[a] and Robert F. Harrison[b]

[a]Department of Mechanical Engineering, The University of Sheffield, Sheffield, UK; [b]Department of Automatic Control and Systems Engineering, The University of Sheffield, Sheffield, UK

ABSTRACT

Optimizing the operation of rail networks using simulations is an on-going task where heuristic methods such as Genetic Algorithms have been applied. However, these simulations are often expensive to compute and consequently, because the optimization methods require many (typically $>10^4$) repeat simulations, the computational cost of optimization is dominated by them. This paper examines Bayesian Optimization and benchmarks it against the Genetic Algorithm method. By applying both methods to test-tasks seeking to maximize passenger satisfaction by optimum resource allocation, it is experimentally determined that a Bayesian Optimization implementation finds 'good' solutions in an order of magnitude fewer simulations than a Genetic Algorithm. Similar improvement for real-world problems will allow the predictive power of detailed simulation models to be used for a wider range of network optimization tasks. To the best of the authors' knowledge, this paper documents the first application of Bayesian Optimization within the field of rail network optimization.

ARTICLE HISTORY
Received 3 April 2019
Revised 6 September 2019
Accepted 15 September 2019

KEYWORDS
Bayesian Optimization; Genetic Algorithm; rail; network; optimization; simulation

1. Introduction

Improving the operation of rail networks is an on-going challenge for rail service providers such as train operating companies and infrastructure managers. In Great Britain, there is evidence that service providers have recognized the importance of improving the performance of their network from the passenger perspective. For example, customer experience is identified as one of four strategic goals by the Technical Leadership Group [1]. To assist with improving network operation, detailed models, for example by Yao et al. [2] and Landex and Nielsen [3], have been developed to predict network performance from the passenger perspective. The optimization of detailed rail network models will enable service providers to improve the performance of their network from the passenger perspective. However, the two well-established methods within the field of rail network optimization [Mathematical Programming and Genetic Algorithms (GAs)] are poorly suited for this task. Mathematical Programming requires specific model formulations that make it difficult to adequately capture the detail of individual passenger journeys [4]. GAs can be applied with any model

CONTACT David I. Fletcher d.i.fletcher@sheffield.ac.uk Department of Mechanical Engineering, The University of Sheffield, Mappin Street, Sheffield S1 3JD, UK

formulation (e.g. an agent-based simulation of individual passenger journeys) to find solutions that are approaching optimal, although without any formal certainty that an optimum is found, that is ‘good’ solutions are identified. However, because a GA requires many computations of a model (e.g. an agent-based simulation) there is a need to maintain a low computational cost for each model evaluation. This requirement is in conflict with the use of detailed simulation models because, in the case where many passengers are being simulated, it is difficult to adequately capture passenger journeys at low computational expense.

Bayesian Optimization (BO) has the potential to find similarly good solutions to a GA, but in fewer model computations because it uses a predictive model of the search space to target the selection of new candidate solutions for evaluation. GAs are well suited to tasks with cheap-to-compute models and large numbers of optimization variables, while BO has the potential to be computationally cheaper for tasks with an expensive-to-compute model and few optimization variables (many established implementations currently perform well for less than approximately 25). This reduction in computational requirement could enable the wider use of more detailed models when optimizing a rail network, or allow good solutions to be found at lower computational cost than with a GA. BO might therefore be an additional optimization tool for rail network operators, especially considering that implementations are being developed to increase the upper limit on the number of optimization variables and reduce the computational cost of calculating its inherent predictive model. This paper investigates BO and experimentally compares it against a GA on a range of test-tasks involving a passenger rail network where there is a large number of operational options, that is potential or *candidate* solutions to the network operation problem. Following a brief review of GA and BO methods, this paper compares the computational costs of both methods. To the best of the authors’ knowledge, this paper documents the first application of BO within the field of rail network optimization.

1.1. Genetic Algorithm for optimizing rail networks

Recently GAs have been applied to tasks involving rail network operation, for example timetabling [5,6], train control [7], and resource allocation [8]. However, typically these tasks have required the evaluation of 10^4–10^5 candidate solutions to find a good solution. The computational cost of the optimization procedure has often been kept reasonable by ensuring that the models used for evaluating candidate solutions require less than 1 second of computation time. Where this is not the case, the total computational cost of optimizing an expensive-to-compute model using a GA may be intractable. It is therefore desirable to find a method that can find a good solution using fewer candidate solution evaluations. This would allow models with a greater computational expense to be used. To keep the computational expense of the optimization procedure reasonable an alternative approach is to consider only a small network. For example, Wei and Yuan [5] demonstrate the use of a GA implementation on a network consisting of a single line and 13 stations that required $\sim 1.5 \times 10^3$ evaluations. However, there are 2560 mainline stations in Great Britain (GB) [9] and a method that will scale to realistically sized networks is desirable.

1.2. Applications of Bayesian Optimization

There are many examples of BO being used to select the hyperparameter values of expensive-to-compute machine learning algorithms [10,11]. Applications outside the field of machine learning are less common but can be categorized into two purposes:

- To maximize the agreement between a model and observed data by optimally fitting model parameters.
- To maximize the performance of a real-world entity by optimizing design and operational model parameters.

In this paper, the BO and GA methods are compared for the latter purpose. An example of this type of application is the use of BO by Candelieri et al. [12] to maximize the performance of a simulated water distribution network by optimizing the pump schedule. Candelieri et al. use BO because of its advantages when applied with expensive-to-compute simulations. For an alternative case, Lisicki et al. [13] report that BO finds a solution which performs approximately 50% better than that found by random search with an equal number of evaluations. Neither Candelieri et al. or Lisicki et al. make a quantitative comparison of BO against a sophisticated optimization method such as GA and at the time of writing, the only identified application of BO in a transportation network setting is by Schultz and Sokolov [14] who optimize the parameters of transportation network simulators to maximize agreement with observed data.

An explicit comparison between GAs and BO is presented by Trotter et al. [15] to compare both approaches for maximizing the performance of a distributed computing system. However, it can be inferred that this comparison is not made for an equal budget of candidate solution evaluations and is therefore difficult to generalize from. Chandrashekaran et al. [16] report the number of candidate evaluations in a comparative optimization of a speech recognition model. However, the use of a very small population size again limits the generalizability of the conclusions.

2. Genetic Algorithm and Bayesian Optimization

As a basis for equitable comparison of BO and GA we consider the general task of selecting the optimum value of a vector, $\boldsymbol{x}$, which maximizes a non-negative objective function, $f(\boldsymbol{x})$. In the case of a rail network, the objective function quantifies the performance of the network. The vector, $\boldsymbol{x}$, comprises n elements and exists in the n-dimensional *search space, X*, bounded by the upper and lower bounds (constraints) of each optimization variable, $x_i, i = 1, 2, \ldots n$, leading to the optimization task given by (1) subject to constraints (2) and (3).

$$\boldsymbol{x}^* = \arg\max_{\boldsymbol{x} \in X} f(\boldsymbol{x}) \tag{1}$$

$$g_j(\boldsymbol{x}) = 0 \; for \; j = 1, 2, \; \ldots \; , \; p \tag{2}$$

$$h_k(\boldsymbol{x}) \leq 0 \; for \; k = 1, 2, \; \ldots \; , \; q \tag{3}$$

Where: $\boldsymbol{x}^*$ is the global optimum and there are p equality constraints on $\boldsymbol{x}$, defined by $g_j(\boldsymbol{x})$, and q inequality constraints, defined by $h_k(\boldsymbol{x})$. For a case requiring minimization of $f(\boldsymbol{x})$, maximization of $-f(\boldsymbol{x})$ would be used.

A brief description of both GAs and BO is given in the following subsections. More comprehensive descriptions of GAs can be found in Goldberg [17] and Mitchell [18], and of BO in Shahriari et al. [19]. Note that within the GA literature, the objective function is usually referred to as a 'fitness function', however '*objective function*' is used here for both cases for consistency.

2.1. Genetic Algorithm

A GA is a simple computational model of the process of natural selection in an evolving population. At every iteration, a GA evaluates the objective function for every candidate solution within a population. By selecting the candidates with the best objective function scores and 'mating' them, the population at later generations exhibits more characteristics of candidates with good objective function scores and converges towards the optimum. 'Mutation' is used to allow a GA to 'explore' the search space. The GA method can be applied to many types of optimization task with any number of optimization variables and, because it requires many (typically >10^4) evaluations of $f(\boldsymbol{x})$, it is well suited to tasks where $f(\boldsymbol{x})$ is cheap-to-compute. Algorithm 1 presents pseudo-code for a simple GA. An important control parameter of the algorithm is the population size, P. The algorithm iterates depending on a conditional statement at line 4 that is often related to the objective function scores of the candidates found so far, or, the computational resources used. The number of algorithm iterations used by a GA, I^{GA}, is the final value of i in Algorithm 1. At every iteration all of the candidates in a generation are evaluated so the number of objective function evaluations, η^{GA}, required, is given by:

$$\eta^{GA} = I^{GA} \cdot P \tag{4}$$

```
Algorithm 1: Genetic Algorithm ($P$)
1. Initialize population, $G_1$, with $P$ candidates
2. Evaluate objective function for every candidate in $G_1$
3. $i = 1$
4. while Is-Not-Terminated($i, G_i$)
5.    $G_{i+1}$ = Evolve($G_i$)//select, mate and mutate candidates
6.    Evaluate objective function for every candidate in $G_{i+1}$
7.    $i = i + 1$
8. end
9. Return best candidate evaluated so far
```

2.2. Bayesian Optimization

To estimate the global maximum of an objective function, a BO implementation creates an approximation of it, called a *proxy function* (also referred to as a *surrogate model* or *response surface*). In comparison to the objective function, the proxy function is cheaper to compute and is continuous so it is ‘easier’ to find its maximum. To create the proxy function, a probabilistic model is inferred from previous evaluations of the objective function’s value at different locations in the search space. For this model, it is common [10] to use a Gaussian Process (GP) regression model as is the case considered here. The proxy function, $\mu(\boldsymbol{x})$, and its uncertainty, $\sigma(\boldsymbol{x})$, are respectively the mean and standard deviation of the GP model. At each iteration of a BO algorithm, the objective function is evaluated and the new data is used to update the probabilistic model and, hence, the proxy function. Information from the proxy function is used to create an *acquisition function*, whose global maximum indicates where in the search space the objective function should next be evaluated. The acquisition function is important to the success of BO because it ‘guides’ the search, but finding its maximum increases the computational expense of the whole process, particularly for tasks with more than approximately 25 dimensions [20]. Nonetheless for tasks with less than approximately 25 dimensions, it is often cheaper than evaluating an expensive-to-compute objective function and consequently the BO method is often well suited to task of this type. For higher dimension tasks, BO implementations are still being developed that keep the cost of maximizing the acquisition function reasonable, for example Li et al. [21]. Algorithm 2 presents pseudo-code outlining BO and is further described in the appendix. BO only uses one objective function evaluation per iteration; therefore, the number of expensive-to-compute evaluations, η^{BO}, is equal to the value of i at the end of Algorithm 2.

```
Algorithm 2: Bayesian Optimization()

Initialize candidate, $\boldsymbol{x}_1$
$y_1 = f(\boldsymbol{x}_1)$//sample objective function
$D = [\boldsymbol{x}_1, y_1]$//data set of corresponding x and y values
Calculate $\mu(\boldsymbol{x})$ and $\sigma(\boldsymbol{x})$ using $D$
$i = 1$
while Is-Not-Terminated $(i, D)$
    Create acquisition function, $\alpha(\boldsymbol{x})$, using $\mu(\boldsymbol{x})$ and $\sigma(\boldsymbol{x})$
    $\boldsymbol{x}_{i+1} = \arg\max_{\boldsymbol{x} \in X} \alpha(\boldsymbol{x})$
    $y_{i+1} = f(\boldsymbol{x}_{i+1})$
    $D = \{D, [\boldsymbol{x}_{i+1}, y_{i+1}]\}$//augment new data to data set
    Calculate $\mu(\boldsymbol{x})$ and $\sigma(\boldsymbol{x})$ using $D$
    $i = i + 1$
end
Return best candidate evaluated so far
```

2.3. Comparing the computational cost of Genetic Algorithms and Bayesian Optimization

Here, the total computational cost of each method is decomposed into the cost of all the objective function evaluations, Γ, and the cost of computing the algorithm (excluding objective function evaluations), Π. Γ is the product of the number of evaluations, η, and the cost of a single evaluation, γ. Note that when variables apply to both approaches, superscripts are used to denote variables specific to an approach. Because of the multiplier, P, in (4), it is expected that $\eta^{GA} > \eta^{BO}$ and consequently that $\Gamma^{GA} > \Gamma^{BO}$. However, because BO involves the expensive step of maximizing an acquisition function at every iteration, $\Pi^{GA} < \Pi^{BO}$ resulting in a trade-off between the low algorithm cost, high objective function evaluation cost of GAs and the high algorithm cost, low objective function evaluation cost of BO. The case of BO being cheaper than a GA is captured by (5) which demonstrates that the value of γ is important for determining the best method. Note that γ is constant between the approaches and is assumed to be constant for all $\boldsymbol{x}$.

$$\Pi^{BO} + \gamma \cdot \eta^{BO} < \ \Pi^{GA} + \ \gamma \cdot \eta^{GA} \tag{5}$$

Because the number of objective function evaluations and the number of algorithm iterations are related, the value of Π is dependent on η. In the case of a GA this is a linear dependency. However in the case of BO with a GP model, calculating the model with i data points requires inversion of a square matrix of dimension i. The computational cost of this is $O(i^3)$. Consequently, the value of Π^{BO} may become fourth order for large η^{BO}. However, developing methods to compute large matrix inversions at reduced cost is an active area of research [22,23] so this is not seen as a fundamental limitation of BO. Furthermore, although not utilized here with BO, Gardner et al. [24] present a method using parallel computing techniques that can reduce the cost of computing a GP model can be reduced to $O(i^2)$.

3. Experimental comparison of a Genetic Algorithm and Bayesian Optimization

For an experimental comparison, specific GA and BO implementations were applied to a range of test-tasks involving an expensive-to-compute objective function that simulates passengers using a rail network and captures their satisfaction. For an unambiguous comparison, the globally optimal solutions ($\boldsymbol{x}^*$) must be known. For this reason, the test-tasks have been chosen so that $\boldsymbol{x}^*$ can be calculated analytically.

3.1. The test-tasks

Two examples of demands in rail network operation are:

- The allocation of finite rolling stock between scheduled train paths [25].
- The choice of which areas of the rail network should receive investment for increased line speed [26].

These general demands are synthesized with a family of rail networks to create a family of test-tasks that involve the allocation of a limited number of identical carriages between trains and the setting of line speeds around the network. The number of carriages allocated to a train determines its passenger capacity but, for these test-tasks, does not affect any other characteristics of the train. For the purpose of the test-tasks, any number of carriages greater than zero can be allocated to a train provided that the limit on the total number of carriages available is not exceeded. The passenger capacity of a train affects the performance of the network because it is related to the comfort and duration of passenger journeys. A track section linking adjoining stations in the test-task network is defined as a *line* that is homogenous and bi-directional. In these test-tasks, the line speed can be one of two alternatives: a 'basic' level and an 'upgraded' level. The line speed affects passenger journey times and hence the network performance. For the test-tasks, the carriage allocations and line speeds are the optimization variables whose values are chosen to maximize network performance from the perspective of passengers and for the greatest number of passengers.

The family of networks used in the test-tasks are chosen to have a high degree of symmetry so that the global optimum can be calculated analytically. A radial network design is used where the central station is connected to outer stations by two lines. The networks are named B2, B3, and B4 where the number refers to the number of links in the network.

Figure 1 shows the topographies of the networks with a circle representing a station and a connecting edge representing a railway line. Each line within the network has one train operating upon it and the trains do not transfer between lines. Table 1 displays the number of lines and trains in the test-networks and how this controls the number of optimization variables. The number of trains within the network is equal to the number of lines within the network so we only state the number of trains from here on in. The

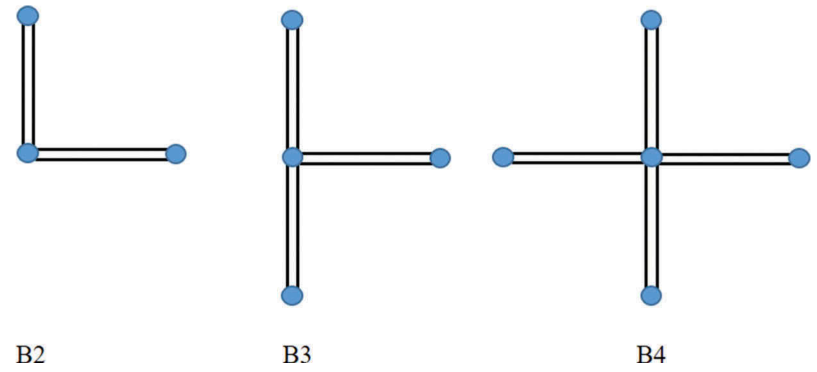

Figure 1. Topographical representation of three different sized networks from the family of networks used for test-tasks. A circle represents a station. An edge between two stations represents a bi-directional line upon which trains travel. The name of the network topography is displayed below each network.

Table 1. The number of lines and trains for the three different sized networks used in test-tasks.

Network topography name	B2	B3	B4
Number of lines	4	6	8
Number of trains	4	6	8
Number of optimization variables	8	12	16

trains operate to a symmetrical timetable and all lines have a dedicated platform at their connected station. The passengers using the network (defined as the passenger load) have a symmetrical origin-destination-time matrix. It is recognized that these networks are simplistic and idealized relative to real world networks, however their properties are sufficient to test the relative performance of the GA and BO methods in preparation for application to more realistic cases.

Although the BO implementation tested here cannot be scaled to tasks representing the whole GB network in this way, that is every individual line represented by an optimization variable, there are opportunities at regional and individual train operator scales. While these tasks present a computational challenge at present, new developments such as a BO implementation allowing more optimization variables might be used [21].

3.2. Formal definition of the test-tasks

Here the general definition of constrained optimization given by (1), (2) and (3) is modified to the family of test-tasks described in the previous section. The tasks can be described as optimizing the distribution of M identical carriages amongst R trains and selecting the line speed of each of the L lines from S discrete choices. The vector, $\boldsymbol{x}$, is $R + L$ dimensional with $x_1, x_2, \ldots .x_R$ describing the number of carriages allocated to trains 1 to R and $x_{R+1}, x_{R+2} \ldots x_{R+L}$ describing the line speeds of lines 1 to L. The form of $\boldsymbol{x}$ is therefore shown by:

$$\boldsymbol{x} = [x_1, x_2, \ldots .x_R,\ x_{R+1}, x_{R+2} \ldots x_{R+L}] \tag{6}$$

The general objective function, $f(\boldsymbol{x})$, is modified to the specific function $F(\boldsymbol{x}; \lambda, \theta)$ which quantifies network performance from the passenger perspective where λ describes the network parameters other than carriage allocations and line speeds (e.g. station locations, train performance, timetable) and θ describes the passenger load. Neither λ or θ are optimized. The test-tasks have no constraints placed on the choice of line speed (i.e. all lines can have the maximum line speed). Furthermore, because there is no penalty associated with increasing the line speed, the globally optimal solution will have all line speeds maximized – in real-world application costs such as energy and wear and tear are associated with higher line speeds so a cost function penalizing these aspects could be accommodated. However, in this task, there is a constraint on the carriage allocations that a maximum of M carriages can be distributed between all the trains, captured by:

$$\sum_{k=1}^{k=R} x_k \leq M \tag{7}$$

A negative number of carriages cannot be allocated, this is captured by (8), and the limit on the number of choices of line speed is captured by (9).

$$0 \leq x_k \ \ for\ 1 \leq k \leq R \tag{8}$$

$$1 \leq x_k \leq S\ for\ R + 1 \leq k \leq R + L \tag{9}$$

The formal definition of the family of test-tasks can therefore be written as (10) subject to (7)–(9).

$$\boldsymbol{x}^* = \arg\max_{\boldsymbol{x}\in X} F(\boldsymbol{x}; \lambda, \theta) \tag{10}$$

3.3. Quantifying solution performance

The objective function $F(\boldsymbol{x}; \lambda, \theta)$ quantifies the performance of a network, described by $\boldsymbol{x}$ and λ and carrying passenger load, θ, from the passenger perspective. To calculate the value of the objective function, an agent-based simulation is used that models individual passengers using a rail network to make their journeys. The quality of the individual passenger journey is quantified and all passenger journeys aggregated to give a network score. This model has been developed by the authors and is fully described by Hickish et al. [27]. The value of $F(\boldsymbol{x}; \lambda, \theta)$ is expressed on the percentage scale from 0 to 100%. 100% relates to the known global maximum, 0% relates to the known global minimum. The experimental parameter, F^*, is introduced to describe the *target performance* of the solution. F^* is the smallest value of $F(\boldsymbol{x}; \lambda, \theta)$ that must be found by an implementation before terminating with a result. Candidates ($\boldsymbol{x}$) for which $F(\boldsymbol{x}; \lambda, \theta) \geq F^*$ are referred to as *acceptable solutions.*

3.4. Methodology

Experiments were carried out in MATLAB 2017b using its proprietary optimization functions, here denoted *ga* and *bayesopt*. The number of objective function evaluations used by *ga* and *bayesopt* was measured for eight 'jobs', that is a specific combination of test-task and F^* that is input to a function. A test-task is defined by the number of trains in the network and the number of carriages to be allocated. To collect experimental data, a job is submitted to a function and the algorithm iterates until an acceptable solution is found or a limit on the number of objective function evaluations is reached. The number of objective function evaluations required by the algorithm is recorded for the experiment. Because the algorithms are non-deterministic it is necessary to repeat, independently, each experiment multiple times and describe distributions. For comparison between the GA and BO methods, identical tasks are submitted to the algorithms. Because the control parameters of each implementation differ, attainment of identical terminal objective function values is used to ensure like-for-like comparison.

For the GA experiments, the default *ga* settings were used with a cross-over rate of 0.5, a mutation rate proportional to the initial range of values in the population (shrinking to zero at the final generation), a uniform stochastic selection function and a population size of 30. For the BO experiments, the *bayesopt* default Matern 5/2 kernel was used, with the Probability-of-Improvement acquisition function and an exploration ratio of 0.5. Computational cost is measured using wall clock time. This is machine specific, but gives an indication of the relative behaviour of the algorithms and is comparable across jobs since they were performed on the same machine, an Intel Xeon Dual Processor @ 2.4 GHz.

3.5. Results

Figure 2 is a box and whisker plot comparing the number of evaluations required by *ga* and *bayesopt*, η^{GA} and η^{BO}, for eight jobs. Each box and whisker represents, on a logarithmic scale, a distribution from 32 repeats of an experiment. Distributions for the same job are plotted next to each other and separated by vertical dashed lines to allow easy comparison. The job and method that each distribution relates to is displayed on the x-axis. The notation 'J1', 'J2', 'J3', etc. can be cross-referenced against Table 2 to observe the experimental parameters defining the job. A full factorial design for three experimental parameters and two levels has been used. An arbitrary limit of 1.6×10^4 objective function evaluations was used which corresponds to approximately 48 hours of objective function computation time per experiment. This limit only affects the median and quartile results of 'J6,GA', but still allows a discernible difference with 'J6,BO'. The box notches indicate a 95% confidence interval of the median. When comparing distributions for the same job but different methods, there are no cases in which the notches overlap indicating that, on average, $\eta^{GA} > \eta^{BO}$ with approximately 95% confidence [28]. The features of the 'J6,GA' plot are indiscernible because for 31 of

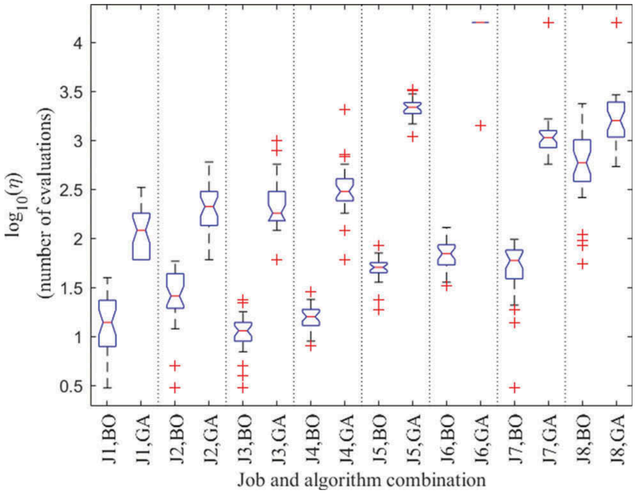

Figure 2. Box plots showing the distribution of the number of objective function evaluations required, η, for eight different jobs. The y-axis is on a $\log_{10}$ scale. Each box plot represents a distribution of 32 repeat experiments. Table 2 displays the experimental parameters of each job described by the 'J number'. The whiskers extend to a maximum of the inter-quartile range below and above the 25th and 75th percentiles respectively. Data outside this range is considered an outlier and is shown by a cross.

Table 2. The experimental parameters of the eight jobs for which the BO and GA methods are compared in Figure 2.

	J1	J2	J3	J4	J5	J6	J7	J8
Trains	4	4	4	4	8	8	8	8
Carriages	8	8	48	48	8	8	48	48
F^* (%)	90	95	90	95	90	95	90	95

the experiments, the GA implementation did not find an acceptable solution within the limit of objective function evaluations. Comparing all eight jobs, the mean of the factor differences between η^{BO} and η^{GA} is 43 with a standard deviation of 76. Furthermore, excluding J6 where the comparison is not valid, the inter-quartile ranges of the BO distributions are narrower (the log scale of the y-axis means this is true even for J7), indicating that BO is more consistent in the number of objective function evaluations required to find an acceptable solution.

3.5.1. The number of objective function evaluations and target performance

Following the comparison to a GA, the relationship between the number of objective function evaluations required by *bayesopt* and the target solution performance was investigated. Five different tasks were considered and the y-axis value of Figure 3 is the logarithm of the median from 24 repeats of the experiment.

The data shows a positive relationship between F^* and $\log_{10}(\eta^{BO})$ that is at least linear. This means that the relationship between F^* and η^{BO} is at least exponential. The increase in η^{BO} is more sensitive to the number of trains in the task than the number of carriages. It can be seen that for the two most difficult problems (six and eight trains, 48 carriages) there is no data for $F^* > 99\%$ and 95.5% respectively. This is because after 4 days of computation, F^* had not been increased and the experiment terminated. While this appears limiting it is worth nothing that, in the case of the eight train task, relaxation of the target performance to 95% enables the target solution performance to be reached in only 1 hour and 20 min of computation. The rapid increase in η^{BO} for solutions close to the optimum is believed to be because of the GP model and not inherent to BO. This is further explained in the discussion section.

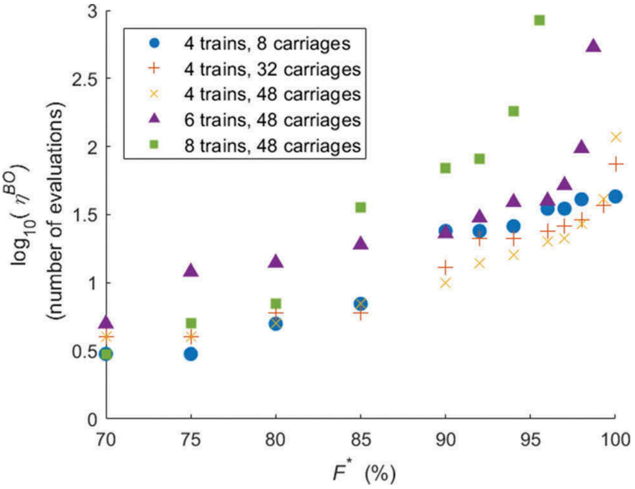

Figure 3. Scatter plots to show the number of objective function evaluations required by Bayesian Optimization, η^{BO}, at varying target performance, F^*. Data is shown for five different tasks. The values plotted are the median of a distribution of 24 repeat experiments. The y-axis displays η^{BO} on a $\log_{10}$ scale.

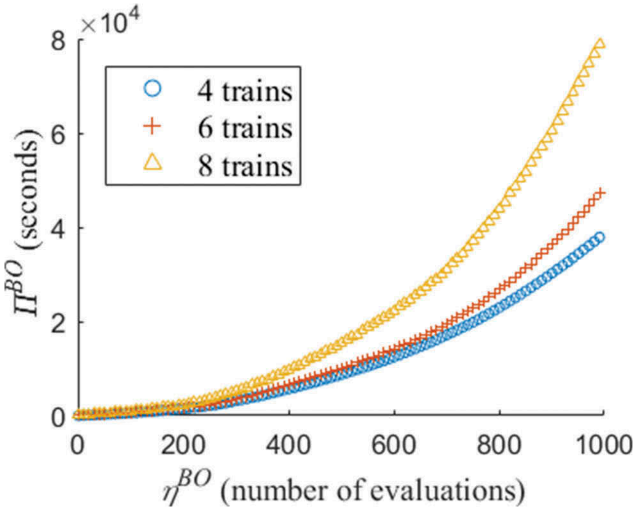

Figure 4. The total algorithm computation time of the Bayesian Optimization implementation, Π^{BO}, at increasing objective function evaluations, η^{BO}. This relationship is investigated for three different tasks. The value displayed by each marker is the median from eight repeat experiments.

3.5.2. Algorithm computation time

To investigate the relationship between η^{BO} and the algorithm time of BO, Π^{BO}, Figure 4 displays the measured total computation time for the *bayesopt* algorithm at varying iterations with different numbers of trains. The computation time required was found to be insensitive to the number of carriages so this is not displayed. For each task the data plotted is the median from eight repeat experiments. Figure 4 shows that the computational cost of the *bayesopt* implementation grows rapidly at later iterations of the algorithm. This is thought to be specific to the use of a GP model within BO and the associated matrix inversion, rather than inherent to the BO approach. The data also shows that the algorithm computation time increases faster for tasks with more trains. This is because increasing the number of trains in the task increases the GP model matrix dimensions and the associated cost of inverting it. Taking the logarithm of Π^{BO} and η^{BO}, to determine the order of computational cost as a function of η^{BO}, $O(\eta^{b})$, gives $b = 2.73$. However, testing to larger values of η^{BO} would be required to confirm that b is not larger.

4. Discussion

The results in Figure 2 show that for all the test-tasks, the BO implementation finds acceptable solutions in significantly less objective function evaluations than the GA. It is not expected that this is unique to the test-tasks. However, the results in Figure 3 indicate that increasing the target performance of the solution leads to an at least exponential increase in the number of objective evaluations required by BO. This is thought to be because in general, for jobs with a high target performance, a proxy function which accurately models the objective function is required. This demands a higher density of evaluations but when many evaluations become clustered in one region the ability of BO to effectively select new candidates is reduced [29]. In addition to the increase in objective function evaluations, the results in Figure 4 indicate that there is a super-linear relationship between the number of evaluations and the algorithm cost. This is thought to be because of the matrix inversion when calculating a GP. Both these effects are thought to be a consequence of implementing BO with a GP model rather than inherent in the BO method. Investigating the effect of using a different probabilistic model within the BO is an area for future investigation.

The results indicate that for certain tasks BO may find 'good' solutions in significantly fewer objective function evaluations than a GA. For tasks involving expensive-to-compute objective functions, this leads to a reduction in total computational expense. When a GP regression model is used as the probabilistic model within BO, increasing the quality of the solution required significantly increases this total expense. However if accelerated techniques for calculating GP models, such as the one developed by Gardner et al. [24], can be implemented with the BO method, this effect might be reduced. The threshold solution quality and computational cost of one evaluation (γ) for which BO is cheaper than GA, are task specific. As identified by McLeod et al. [29] it is likely that in applications where the target performance for the solution is high, a multi-strategy optimization method would be most effective, that is switching from BO to GA (or another alternative).

5. Conclusions

GAs are a well-established optimization method. However, because they typically require, for real-world applications, in the order of 10^4 objective function evaluations or more, there is a pressure to keep the computational cost of an objective function evaluation low. BO uses information from *all* previous evaluations of the objective function to guide the selection of new candidate solutions so that the most beneficial ones are targeted. This means that BO has the potential to find solutions of a similar quality to a GA but in fewer objective function evaluations and, for tasks with expensive-to-compute objective functions, be computationally cheaper. This was experimentally confirmed using a range of test-tasks where the mean factor difference between the numbers of evaluations required by the methods was 43 with standard deviation of 76. However, due to the overhead in the algorithm of the BO implementation tested, a super-linear relationship was found between the total algorithm cost and the number of objective function evaluations required. Furthermore, the relationship between the number of objective function evaluations required and improving solution quality is at least exponential. This is thought to be an effect of using a GP model within the BO algorithm and not inherent to the BO method itself. This means that the BO implementation tested is better applied to tasks involving expensive-to-compute objective functions where approximate answers are satisfactory and the budget for computational expense is small. Two approaches which may improve the solutions found by BO are to either: improve the probabilistic model used within BO, or, sequentially combine BO with GA for a multi-strategy optimization method. The experiments in this paper are conducted using a family of tasks with up to 16 optimization variables. To further evaluate the suitability of BO for the optimization of expensive-to-compute transportation network models, investigation of a wider range of tasks (e.g. train scheduling or driving tasks) and a greater number of optimization variables is a target for the next stages of research in this area.

Nomenclature

$f(\boldsymbol{x})$	A general objective function
$\mu(\boldsymbol{x})$	A proxy function
$\sigma(\boldsymbol{x})$	An uncertainty function
$\alpha(\boldsymbol{x})$	An acquisition function
$F(\boldsymbol{x}; \lambda, \theta)$	A specific objective function
$\boldsymbol{x}^*$	Solution vector
$\boldsymbol{x}$	Candidate vector
n	Number of optimization variables in a task
λ	Fixed network parameters
θ	Passenger load
m	Number of carriages to be allocated
R	Number of trains
L	Number of lines
S	Number of line speed choices
F^*	Target performance of solution
η	Number of objective function evaluations used
γ	One objective function evaluation computational cost (seconds)
Γ	Evaluating the objective function computational cost seconds)
Π	Algorithm computational cost (seconds)
I	Number of algorithm iterations
P	GA population size

Disclosure statement

No potential conflict of interest was reported by the authors.

Funding

Funding received from Network Rail and EPSRC grant number EP/M508135/1.

ORCID

Bob Hickish http://orcid.org/0000-0003-2526-7863
David I. Fletcher http://orcid.org/0000-0002-1562-4655
Robert F. Harrison http://orcid.org/0000-0002-9323-8637

References

[1] Rail technical strategy capability delivery plan [Internet]. London (UK): Rail Safety and Standards Board; 2017 [cited 2018 Nov 7]. Available from: https://www.rssb.co.uk/rts/Documents/2017-01-27-rail-technical-strategy-capability-delivery-plan-brochure.pdf

[2] Yao X, Zhao P, Qiao K. Simulation and evaluation of urban rail transit network based on multi-agent approach. J Ind Eng Manage. 2013;6(1):367–379.

[3] Landex A, Nielsen OA. Simulation of disturbances and modelling of expected train passenger delays. In: Allan J, Brebbia CA, Rumsey AF, editors. Computers in railways X. Southampton (UK): WIT Press; 2006. p. 85–93.

[4] Kanai S, Shiina K, Harada S, et al. An optimal delay management algorithm from passengers' viewpoints considering the whole railway network. J Rail Transp Plan Manage. 2011;1(1):25–37.

[5] Wei L, Yuan ZZ. A robust timetabling model for a metro line with passenger activity information. Inf. 2017;8(3):102–122.

[6] Xu XM, Li KP, Li X. A multi-objective subway timetable optimization approach with minimum passenger time and energy consumption. J Adv Transp. 2016;50(1):69–95.

[7] Goodwin JCJ, Fletcher DI, Harrison RF. Multi-train trajectory optimisation to maximise rail network energy efficiency under travel-time constraints. Proc Inst Mech Eng F Rail Rapid Transit. 2016;230(4):1318–1335.

[8] Fan B, Roberts C, Weston P. A comparison of algorithms for minimising delay costs in disturbed railway traffic scenarios. J Rail Transp Plan Manage. 2012;2(1):23–33.

[9] Estimates of Station Usage 2016–2017 [Internet]. London (UK): Office of Rail and Road; 2017 [cited 2018 Nov 7]. Available from: http://orr.gov.uk/__data/assets/pdf_file/0020/26129/estimates-of-station-usage-2016-17-key-facts.pdf

[10] Snoek J, Larochelle H, Adams RP. Practical Bayesian optimization of machine learning algorithms. In: Pereia F, Burges CJC, Bottou L, et al., editors. Proceedings of the 25th International Conference on Neural Information Processing Systems; 2012 Dec 3; Lake Tahoe (NV). New York (USA): Curran Associates; 2012. p. 2951–2959.

[11] Jordan MI, Mitchell TM. Machine learning: trends, perspectives, and prospects. Science. 2015;349(6245):255–260.

[12] Candelieri A, Perego R, Archetti F. Bayesian optimization of pump operations in water distribution systems. J Glob Optim. 2018;71(1):213–235.

[13] Lisicki M, Lubitz W, Taylor GW. Optimal design and operation of Archimedes screw turbines using Bayesian optimization. Appl Energy. 2016;183:1404–1417.

[14] Schultz L, Sokolov V. Bayesian optimization for transportation simulators. Procedia Comput Sci. 2018;130:973–978.

[15] Trotter M, Liu GY, Wood T. Into the storm: descrying optimal configurations using genetic algorithms and Bayesian optimization. In: Frantz CK, Iannucci S, Pitt J, editors. IEEE 2nd International Workshops on Foundations and Applications of Self* Systems; 2017 Sep 18; Tucson (AZ). Washington (DC): IEEE Computer Society Conference Publishing Services; 2017. p. 175–180.

[16] Chandrashekaran A, Lane I Automated Optimization of decoder hyper-parameters for online LVCSR. In: Cardinal P, Hansen JHL, editors. IEEE Workshop on Spoken Language Technology; 2016 Dec 13; San Diego (CA). [Washington (DC)]: IEEE; 2016. p. 454–460.

[17] Goldberg DE. Genetic algorithms in search, optimization, and machine learning. Reading (MA): Addison-Wesley; 1989.

[18] Mitchell M. An introduction to genetic algorithms. Cambridge, (MA): MIT Press; 1996.

[19] Shahriari B, Swersky K, Wang Z, et al. Taking the human out of the loop: A review of Bayesian optimization. Proce IEEE. 2016;104(1):148–175.

[20] Kandasamy K, Schneider J, Piczos B. High dimensional Bayesian optimisation and bandits via additive models. In: Bach F, Beli D, editors. Proceedings of the 32nd International Conference on Machine Learning; 2015 Jul 6; [Lille (France)]: JMLR; 2015. p. 295–304.

[21] Li C, Gupta S, Rana S, et al. High dimensional Bayesian optimization using dropout. In: Sierra C, editor. Proceedings of the 26th International Joint Conference on Artificial

Intelligence; 2017 Aug 19; Melbourne (Australia). [place unknown]: International Joint Conferences on Artificial Intelligence Organization; 2017. p. 2096–2102.

[22] Li JJ, Ranka S, Sahni S. Strassen's matrix multiplication on GPUs. In: Mutka M, Shieh CK, Cheng ST, et al., editors. 17th IEEE International Conference on Parallel and Distributed Systems; Tainan, Taiwan: IEEE; 2011 Dec 7; Washington (DC): IEEE Computer Society Conference Publishing Services; 2011. p. 157–164.

[23] Ballard G, Benson AR, Druinsky A, et al. Improving the numerical stability of fast matrix multiplication. SIAM J Matrix Anal Appl. 2016;37(4):1382–1418.

[24] Gardner J, Pleiss G, Weinberger KQ, et al. GPyTorch: blackbox matrix-matrix Gaussian process inference with GPU acceleration. In: Bengio S, Wallach H, Larochelle H, et al., editors. Neural Information Processing Systems 2018; Montreal, Canada: Neural Information Processing Systems Foundation; 2018.

[25] Abbink E, van Den Berg B, Kroon L, et al. Allocation of railway rolling stock for passenger trains. Transp Sci. 2004;38(1):33–41.

[26] Railway Upgrade Plan 2017/2018 [Internet]. London (UK): Network Rail; 2018 [cited 2018 Nov 7]. Available from: https://cdn.networkrail.co.uk/wp-content/uploads/2017/08/Railway-Upgrade-Plan-Update-2017-2018.pdf

[27] Hickish B, Fletcher DI, Harrison RF. Maximising passenger satisfaction through optimised train movements. In: How F, Iwnicki S, Odetunde S, et al., editors. The Stephenson Conference; 2017 Apr 25–27; London (UK): The Institution of Mechanical Engineers; 2017. p. 181–192.

[28] Chambers J, Cleveland W, Kleiner B, et al. Comparing data distributions. In: Chambers J, editor. Graphical methods for data analysis. Belmont (CA): Wadsworth International Group; 1983. p. 62–63.

[29] McLeod M, Osborne M, Roberts S. Optimization, fast and slow: optimally switching between local and Bayesian optimization. In: Dy J, Krause A, editors. Proceedings of the 35th International Conference on Machine Learning; 2018 Jul 10; Stockholm (Sweden). [place unknown]: PMLR; 2018. p. 3443–3452.

Appendix

Figure 5 illustrates the early stages of the BO procedure, given in Algorithm 2 pseudo-code for the general BO approach. The x-axis represents the value of the optimization variable, $\boldsymbol{x}$, and the left ordinate represents both the value of the objective function, $f(\boldsymbol{x})$, its proxy function, $\mu(\boldsymbol{x})$, and the uncertainty about the proxy function, $\mu(\boldsymbol{x}) \pm \sigma(\boldsymbol{x})$. The right ordinate represents the value of the acquisition function, $\alpha(\boldsymbol{x})$. The first stage is to evaluate $f(\boldsymbol{x})$ at the initial candidate, $\boldsymbol{x}_1$, whose location is shown by a square marker (line 2 of Algorithm 2). Stage 2 shows that this information is used to create an initial $\mu(\boldsymbol{x})$ that models what is known about $f(\boldsymbol{x})$ at this stage (line 4 of Algorithm 2). A corresponding $\sigma(\boldsymbol{x})$ is also calculated and combined with $\mu(\boldsymbol{x})$ to calculate the acquisition function. Stage 3 shows that $\alpha(\boldsymbol{x})$ increases further away from $\boldsymbol{x}_1$ because of the increased uncertainty on the value of $f(\boldsymbol{x})$. Stage 3 also shows the maximum of the acquisition function with a triangle marker. The x-axis value of this maximum is $\boldsymbol{x}_2$ (line 7 of Algorithm 2) which is then used to evaluate $f(\boldsymbol{x})$ in Stage 4, shown by a square marker (line 9 of Algorithm 2). Stage 4 shows a circle marker to represent that data from previous samples remain in the probabilistic model and because there is now more data, $\mu(\boldsymbol{x})$ better approximates $f(\boldsymbol{x})$ than in Stage 2 with reduced $\sigma(\boldsymbol{x})$ in the region around the second observation. Stages 5 and 6 demonstrate that maximizing $\alpha(\boldsymbol{x})$ and calculating $\mu(\boldsymbol{x})$ and $\sigma(\boldsymbol{x})$ are two important processes that are repeated in every iteration of the BO approach.

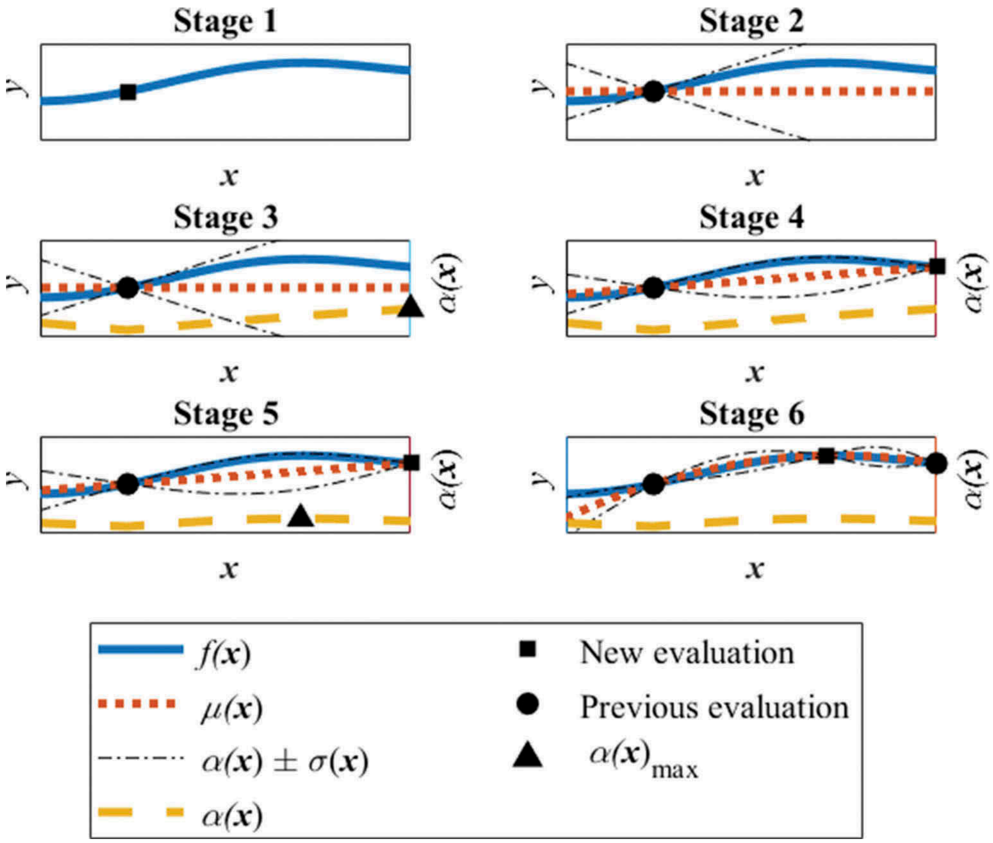

Figure 5. Illustration of six of the early stages in the BO approach. The objective function, $f(\boldsymbol{x})$, is shown by a solid line, the proxy function, $\mu(\boldsymbol{x})$, is shown by a dotted line and the uncertainty about the proxy function, $\mu(\boldsymbol{x}) \pm \sigma(\boldsymbol{x})$, is shown by a dash-dot line. The values of $f(\boldsymbol{x})$, $\mu(\boldsymbol{x})$ and $\mu(\boldsymbol{x}) \pm \sigma(\boldsymbol{x})$ are plotted on the left ordinate. The acquisition function, $\alpha(\boldsymbol{x})$, is shown by a dashed line and its value plotted on the right ordinate. The x-axis represents the variable, $\boldsymbol{x}$, being optimized.