An Efficient Approach for Assessing Parameter Importance in Bayesian Optimization

Frank Hutter
Freiburg University
fh@informatik.uni-freiburg.de

Holger H. Hoos and Kevin Leyton-Brown
University of British Columbia
{hoos,kevinlb}@cs.ubc.ca

Abstract

We describe a method for quantifying the importance of a blackbox function's input parameters and their interactions, based on function evaluations obtained by running a Bayesian optimization procedure. We focus on high-dimensional functions with mixed discrete/continuous as well as conditional inputs, and therefore employ random forest models. We derive the first exact and efficient approach for computing efficient marginal predictions over subsets of inputs in random forests, enabling an exact quantification of parameter importance in the functional ANOVA framework. We demonstrate these techniques by assessing the importance of parameters in several recent applications of Bayesian optimization.

1 Introduction

Bayesian optimization has achieved considerable success on application problems such as determining good hyperparameter settings of machine learning methods [4, 18, 21, 2] and configuring solvers for hard combinatorial problems [16, 13]. Much less work has been done to develop methods for providing scientists with answers to questions like the following: "How important is each of the parameters, and how do their values affect performance? Which parameter interactions matter? How do the answers to these questions depend on the data set under consideration?" Because answers to such questions can drive new scientific discoveries, recent Bayesian optimization workshops have identified the need for more work on methods that do not only optimize a given blackbox function but also provide some information about its characteristics. Recent work on Bayesian optimization has targeted the case where most parameters are truly unimportant [7, 22], and several application studies have yielded evidence that some parameters indeed tend to be much more important than others [3, 14, 8]. Identifying such important parameters becomes particularly necessary for highly parameterized algorithms, such as deep belief networks (32 parameters [4]); combinatorial solvers (up to 76 parameters [13]), complex vision architectures (238 parameters [5]); and combined model selection and hyperparameter optimization (e.g., 768 parameters in Auto-WEKA [21]).

To assess the importance of input parameters to a blackbox function $f : \Theta_1 \times \cdots \times \Theta_n \to \mathbb{R}$, in this work, we investigate the classic technique of functional analysis of variance (functional ANOVA) [19, 12, 17, 11]: we fit a model $\hat{f}$ to f and partition $\hat{f}$'s variance $\mathbb{V}$ into additive components $\mathbb{V}_U$ associated with each subset of input parameters $U \subseteq \{1, \ldots, n\}$. In order to do this tractably, we must be able to efficiently compute marginalizations of $\hat{f}$ over arbitrary parameter subsets $U \subseteq \{1, \ldots, n\}$. This has been shown to be possible for Gaussian process models $\hat{f}$ with certain kernels [see, e.g., 17]. However, our problem domain of Bayesian optimization of highly parameterized algorithms is characterized by high-dimensional discrete/continuous data, conditional parameters, many data points, non-Gaussian noise, and non-stationarity, for which we and others have found random forest models to be more appropriate [20, 9, 13]. Because to date, efficient marginalizations had not been available for random forests, researchers had to revert to sampling-based techniques to compute approximate functional ANOVA decompositions [9]. Here, we provide the first efficient and exact method for computing marginal predictions and deriving functional ANOVA sensitivity indices for random forests. We demonstrate their use by studying the parameter spaces of several algorithms.

2 Notation and Definitions

Let f be a function with n input parameters with domains $\Theta_1, \ldots, \Theta_n$. We use positive integers to denote the parameters, and N to refer to the set $\{1, \ldots, n\}$ of all parameters. The *parameter space* is $\boldsymbol{\Theta} = \Theta_1 \times \cdots \times \Theta_n$. A *parameter configuration* is a vector $\boldsymbol{\theta} = \langle \theta_1, \ldots, \theta_n \rangle$ with $\theta_i \in \Theta_i$. A *partial instantiation* of a subset $U = \{u_1, \ldots, u_m\} \subseteq N$ of A's parameters is a vector $\boldsymbol{\theta}_U = \langle \theta_{u_1}, \ldots, \theta_{u_m} \rangle$ with $\theta_{u_i} \in \Theta_{u_i}$. The *extension set* $X(\boldsymbol{\theta}_U)$ *of* $\boldsymbol{\theta}_U$ of a partial parameter instantiation $\boldsymbol{\theta}_U$ is the set of parameter configurations that are consistent with it. More formally, let $\boldsymbol{\theta}_U = \langle \theta_{u_1}, \ldots, \theta_{u_m} \rangle$ be a partial instantiation of the parameters $U = \{u_1, \ldots, u_m\} \subseteq N$; $X(\boldsymbol{\theta}_U)$ is then the set of parameter configurations $\boldsymbol{\theta}_{N|U} = \langle \theta'_1, \ldots, \theta'_n \rangle$ such that $\forall j (j = u_k \Rightarrow \theta'_j = \theta_{u_k})$.

Definition 1 (Marginal)**.** *Let $f : \boldsymbol{\Theta} \to \mathbb{R}$ be a blackbox function, $U \subseteq N$, and $T = N \setminus U$. The* marginal $a_U(\boldsymbol{\theta}_U)$ of f over T *is then defined as*

$$a_U(\boldsymbol{\theta}_U) = \mathbb{E}[f(\boldsymbol{\theta}_{N|U}) \mid \boldsymbol{\theta}_{N|U} \in X(\boldsymbol{\theta}_U)] = \int f(\boldsymbol{\theta}_{N|U}) \cdot p(\boldsymbol{\theta}_T)\, d\boldsymbol{\theta}_T.$$

Since we aim for marginals representing the input space evenly, we use the uniform distribution $p(\boldsymbol{\theta}_T)$ for any T.

3 Functional ANOVA

Analysis of variance (ANOVA) partitions the observed variation of a response value into components due to each of several factors. *Functional* ANOVA decomposes the variance of a function $f : \boldsymbol{\Theta} \to \mathbb{R}$ across its domain into additive components that only depend on subsets of its parameters N:

$$f(\boldsymbol{\theta}) = \sum_{U \subseteq N} f_U(\boldsymbol{\theta}_U). \tag{1}$$

The components $f_U(\boldsymbol{\theta}_U)$ are defined as follows:

$$f_U(\boldsymbol{\theta}_U) = \begin{cases} \int f(\boldsymbol{\theta}) \cdot p(\boldsymbol{\theta})\, d\boldsymbol{\theta} & \text{if } U = \emptyset; \\ a_U(\boldsymbol{\theta}_U) - \sum_{W \subsetneq U} f_W(\boldsymbol{\theta}_W) & \text{otherwise.} \end{cases} \tag{2}$$

The constant $f_\emptyset$ is the mean of f across its domain. The unary functions $f_{\{j\}}(\boldsymbol{\theta}_{\{j\}})$ are called *main effects* and capture the effect of varying parameter j, averaging across all instantiations of all other parameters. The functions $f_U(\boldsymbol{\theta}_U)$ for $|U| > 1$ capture exactly the interaction effects between all variables in U (excluding all lower-order main and interaction effects of $W \subsetneq U$).

By definition, the variance of f across its domain $\boldsymbol{\Theta}$ is

$$\mathbb{V} \quad = \quad \int (f(\boldsymbol{\theta}) - f_\emptyset)^2 \cdot p(\boldsymbol{\theta})\, d\boldsymbol{\theta}. \tag{3}$$

Assuming a uniform prior, functional ANOVA decomposes this variance into contributions by all subsets of variables [see, e.g., 11, for a derivation]:

$$\mathbb{V} \quad = \quad \sum_{U \subset N} \mathbb{V}_U, \quad \text{where} \quad \mathbb{V}_U = \int f_U(\boldsymbol{\theta}_U)^2 \cdot p(\boldsymbol{\Theta}_U)\, d\boldsymbol{\theta}_U. \tag{4}$$

The importance of all main and interaction effects f_U can thus be quantified by the fraction of variance they explain: $\mathbb{F}_U = \mathbb{V}_U / \mathbb{V}$. Key to computing these so-called *sensitivity indices* is access to the marginals $a_U(\boldsymbol{\theta}_U)$. In our setting, f is an expensive blackbox function, so there is generally no hope of evaluating it across its entire domain in order to compute $a_U(\boldsymbol{\theta}_U)$. However, if we have a predictive model $\hat{f}$ that fits f well on average across the parameter space, the difference between the true marginals $a_U(\boldsymbol{\theta}_U)$ and the predicted marginals $\hat{a}_U(\boldsymbol{\theta}_U)$ under $\hat{f}$ will also be low, and so will the difference between sensitivity indices based on f and $\hat{f}$. We show that for regression tree models $\hat{f}$ we can compute marginal predictions $\hat{a}_U(\boldsymbol{\theta}_U)$ and sensitivity indices exactly and efficiently.

4 Efficient Marginals and Functional ANOVA Indices in Random Forests

The simplest way to study the importance of a parameter θ_i is to investigate its impact in the context of a fixed instantiation of all other parameters. However, this only yields *local* information on θ_i's

importance; we are interested in its *global* impact across various instantiations of the other parameters. We can obtain this global information by studying the marginal predictions $\hat{a}_{\{i\}}(\theta_i)$ for $\theta_i \in \Theta_i$.

We now show that when using random forest models $\hat{f}$, marginal predictions $\hat{a}_U(\boldsymbol{\theta}_U)$ can be computed in linear time for arbitrary subsets of parameters $U \subseteq N$. Random forests [6] are ensembles of regression trees. Each regression tree partitions the input space through sequences of branching decisions that lead to each of its leaves. We denote the partitioning as $\mathcal{P}$ and observe that each equivalence class $P_i \in \mathcal{P}$ is associated with a leaf of the regression tree and with a constant c_i. Let $\Theta_j^{(i)} \subset \Theta_j$ denote the subset of domain values of parameter j that is consistent with the branching decisions leading to the leaf associated with P_i. Then, for trees with axis-aligned splits, P_i is the Cartesian product $P_i = \Theta_1^{(i)} \times \cdots \times \Theta_n^{(i)}$. The predictor $\hat{f} : \boldsymbol{\Theta} \to \mathbb{R}$ encoded by the regression tree is $\hat{f}(\boldsymbol{\theta}) = \sum_{P_i \in \mathcal{P}} \mathbb{I}(\boldsymbol{\theta} \in P_i) \cdot c_i$, where $\mathbb{I}(x)$ is the indicator function. A random forest simply predicts the average over the predictions obtained from its component regression trees.

Our approach for computing marginal predictions $\hat{a}_U(\boldsymbol{\theta}_U)$ of a random forest works in two phases: a preprocessing phase that has to be carried out only once and a prediction phase that is carried out once per requested marginal prediction. Both phases require only linear time given a random forest as input. (Constructing the random forest is a separate problem, but is also cheap: quadratic in the number of data points T in the worst case, and proportional to $T \log^2 T$ in the more realistic best case of balanced trees [15].)

The key idea behind our algorithm is to exploit the fact that each of the regression trees in a given forest defines a partitioning $\mathcal{P}$ of the configuration space $\boldsymbol{\Theta}$, with piecewise constant predictions c_i in each $P_i \in \mathcal{P}$, and that the problem of computing sums over an arbitrary number of configurations thus reduces to the problem of identifying the ratio of configurations that fall into each partition. The random forest prediction then simply averages the individual tree predictions.

In the full version of this paper, we derive the following result:

Theorem 2. *Given a random forest F with B trees of up to L leaves that defines a predictor $\hat{f} : \boldsymbol{\Theta} \to \mathbb{R}$ for a configuration space with n parameters and maximal categorical domain size D, the time and space complexity of computing a single marginal prediction of F is $O(B \cdot L \cdot \max\{D + n, n \log D\})$. Additional marginal predictions cost additional space $O(1)$ and time $O(B \cdot L \cdot n \log D)$.*

Finally, plugging these marginal computations into the functional ANOVA framework and performing simple dynamic programming to compute $f_U(\boldsymbol{\theta}_U)$ via Equation 2 yields the following result:

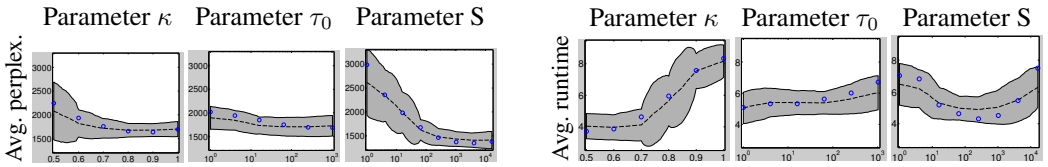
Theorem 3. *Given a configuration space $\boldsymbol{\Theta}$ consisting of n categorical[1] parameters of maximal domain size D and a regression tree $\mathcal{T}$ with L leaves that defines a predictor $\hat{f} : \boldsymbol{\Theta} \to \mathbb{R}$, it is possible to exactly compute the sensitivity indices $\mathbb{F}_U$ of all subsets U of $\boldsymbol{\Theta}$'s parameters N of arity up to K, with space complexity $O(L \cdot D + L \cdot n)$ and time complexity $O\left(L \cdot D + \sum_{k=1}^{K} \binom{n}{k} \cdot D^k (L \cdot n \log d + 2^k)\right)$.*

We compute sensitivity indices for each of the trees in a random forest and return means and variances over the trees' sensitivity indices to express our model uncertainty in regions of sparse data.

5 Empirical Evaluation

We start with a simple 3-dimensional hyperparameter space, for which ground truth data is available for a 288-point grid of the three hyperparameters (κ, τ_0, and S) of an online variational Bayes algorithm for Latent Dirichlet Allocation (Online LDA [10]). This data was made available as part of a previous study applying Bayesian optimization to this algorithm's hyperparameters [18]. For each of the grid points, perplexity and runtime of the LDA algorithm are available. We used only half the data points (sampled uniformly at random) to construct random forests models and plot their marginal predictions in Figure 1, comparing to the true marginals at the grid points. It is clear that parameter S is most important for perplexity, and our functional ANOVA analysis can quantify

[1] For continuous parameters j with $\boldsymbol{\Theta}_j = [l_j, u_j]$, we have to sum over all intervals of $[l_j, u_j]$ defined by the split points in $\bigcup_{P_i \in \mathcal{P}} \{\min \boldsymbol{\Theta}_j^{(i)}, \max \boldsymbol{\Theta}_j^{(i)}\}$. The number of such intervals can in principle grow as large as the number of leaves, leading to an increased worst-case time complexity $O\left(L \cdot D + \sum_{k=1}^{K} \binom{n}{k} \cdot L^k (L \cdot n \log d + 2^k)\right)$.



(a) Main effects for perplexity

(b) Main effects for runtime

Figure 1: Main effects for Online LDA's perplexity and runtime. Each plot shows the marginals of one hyperparameter (identified by the column header); the y-axis in each plot shows marginals (across all instantiations of the other two hyperparameters) when varying that single parameter across the x-axis. The dashed black line and grey-shaded area indicate predicted marginal means $\pm$ two standard deviations, the blue circles indicate true marginals (computable only with ground truth).

this: 65% of the variance of perplexity is due to S and another 18% is due to an interaction effect between S and κ (not shown due to space constraints). Hyperparameter κ most influences runtime (causing 54% of its variation), followed by S (causing 21% of its variation). We note that when the function evaluations are available, this quantitative analysis takes milliseconds and the visualizations (enabled by marginal predictions) can provide valuable intuitions to the algorithm designer even in low-dimensional hyperparameter spaces.

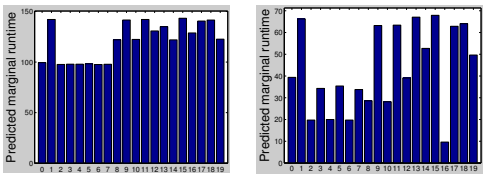


Figure 2: Main effect of SPEAR's variable selection heuristic (with 20 possible values) for two different instance distributions. Left: BMC; right: SWV.

We also applied our variance decomposition methods to assess parameter importance of seven highly parametric state-of-the-art solvers for 10 prominent combinatorial problem benchmarks from propositional satisfiability (SAT), mixed integer programming (MIP) and answer set programming (ASP), covering applications in formal verification, industrial process optimization, computational sustainability, and database query optimization. For all these benchmarks, most variation was due to a few parameters. Indeed, main effects explained a substantial fraction of overall performance variation (20–88%), and likewise did pairwise interaction effects (up to 45%). One particularly interesting case was the performance of the SAT solver SPEAR [1] on SAT-encoded formal verification instances. Here, 87% of the variance was explained by only one out of SPEAR's 26 parameters (namely, its variable selection heuristic). Figure 2 (left) shows that several standard activity heuristics (labelled 0,2,3,4,5,6,7) performed well for this dataset, whereas other ad-hoc heuristics performed poorly. In contrast, for SPEAR's performance on software verification (SWV) instances (see Figure 2, right side), a simple heuristic (labelled 16) initially expected to perform poorly turned out to be very effective. Before seeing these results, SPEAR's developer did not have any intuition about which variable selection heuristic would work well for SWV. Our automatically-derived result helped him realize that a special property of the SWV SAT encoding creates instances suited perfectly for the simple heuristic. This example illustrates how this analysis approach can help algorithm designers (or more abstractly, users with a blackbox function) gain new insights.

The full version of this paper contains many more experiments, including an application to assess which of the 768 parameters of the recent Auto-WEKA framework [21] were important on 21 datasets. Not surprisingly, model class was the most important parameter, followed by (in various orders) the base classifier to use inside a meta-classifier, feature search and feature evaluation methods.

6 Conclusion

We introduced an efficient approach for assessing the importance of the inputs to a blackbox function and applied it to quantify the effect of several algorithms' parameters. Our key technical advance is the derivation of the first exact and efficient algorithm for computing marginal predictions over input dimensions in random forests, thus enabling the practical use of the functional ANOVA framework.

The methods introduced here offer a principled, scientific way for algorithm designers and users to gain deeper insights into the way in which design choices controlled by parameters affect the overall performance of a given algorithm. In future work, we plan to extend our approach to detect dominated parameter values. We also plan to exploit our efficiently derived sensitivity indices inside of Bayesian optimization algorithms, e.g., to focus optimization in the space of the most important parameters or to speed up the subsidiary high-dimensional optimization to select promising configurations.

References

[1] D. Babić and F. Hutter. Spear theorem prover. Solver description, SAT competition, 2007.

[2] R. Bardenet, M. Brendel, B. Kégl, and M. Sebag. Collaborative hyperparameter tuning. In *Proc. of ICML-13*, 2013.

[3] J. Bergstra and Y. Bengio. Random search for hyper-parameter optimization. *JMLR*, 13:281–305, 2012.

[4] J. Bergstra, R. Bardenet, Y. Bengio, and B. Kégl. Algorithms for Hyper-Parameter Optimization. In *Proc. of NIPS-11*, 2011.

[5] J. Bergstra, D. Yamins, and D. D. Cox. Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures. In *Proc. of ICML-12*, 2013.

[6] L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[7] B. Chen, R.M. Castro, and A. Krause. Joint optimization and variable selection of high-dimensional Gaussian processes. In *Proc. of ICML-12*, 2012.

[8] C. Fawcett and H.H. Hoos. Analysing differences between algorithm configurations through ablation. In *Proc. of MIC'13*, 2013.

[9] R.B. Gramacy, M. Taddy, and S.M. Wild. Variable selection and sensitivity analysis using dynamic trees, with an application to computer code performance tuning. *Ann. Appl. Stat.*, 7(1):51–80, 2013.

[10] M. D. Hoffman, D. M. Blei, and F. R. Bach. Online learning for latent dirichlet allocation. In *Proc. of NIPS-10*, 2010.

[11] G. Hooker. Generalized functional ANOVA diagnostics for high dimensional functions of dependent variables. *Journal of Computational and Graphical Statistics*, 16(3), 2007.

[12] J. Z. Huang. Projection estimation in multiple regression with application to functional anova models. *The Annals of Statistics*, 26(1):242–272, 1998.

[13] F. Hutter, H. H. Hoos, and K. Leyton-Brown. Sequential model-based optimization for general algorithm configuration. In *Proc. of LION-5*, 2011.

[14] F. Hutter, H. H. Hoos, and K. Leyton-Brown. Identifying key algorithm parameters and instance features using forward selection. In *Proc. of LION-7*, 2013.

[15] F. Hutter, L. Xu, H.H. Hoos, and K. Leyton-Brown. Algorithm runtime prediction: Methods and evaluation. *AIJ*, 2013. To appear; preprint available on arXiv: CoRR abs/1211.0906.

[16] F. Hutter. *Automated configuration of algorithms for solving hard computational problems*. PhD thesis, Univ. of British Columbia, Dept. of Computer Science, Vancouver, Canada, October 2009.

[17] D. R. Jones, M. Schonlau, and W. J. Welch. Efficient global optimization of expensive black box functions. *Journal of Global Optimization*, 13:455–492, 1998.

[18] J. Snoek, H. Larochelle, and R.P. Adams. Practical Bayesian optimization of machine learning algorithms. In *Proc. of NIPS-12*, 2012.

[19] I. M. Sobol. Sensitivity estimates for nonlinear mathematical models. *Mathematical Modeling and Computational Experiment*, 1(4):407–414, 1993.

[20] M. A. Taddy, R. B. Gramacy, and N. G. Polson. Dynamic trees for learning and design. *Journal of the American Statistical Association*, 106(493):109–123, 2011.

[21] C. Thornton, F. Hutter, H. H. Hoos, and K. Leyton-Brown. Auto-WEKA: Combined selection and hyperparameter optimization of classification algorithms. In *Proc. of KDD-13*, 2013.

[22] Z. Wang, M. Zoghi, F. Hutter, D. Matheson, and N. de Freitas. Bayesian optimization in high dimensions via random embeddings. In *Proc. of IJCAI-13*, 2013.