Artificial Intelligence and Augmented Intelligence for Automated Investigations for Scientific Discovery

Bayesian Optimisation in Chemistry

Project Report

Project Dates: 01/07/2021 - 15/09/2021

Prof. Jeremy Frey

Project Student: Rubaiyat Mohammad Khondaker, University of Cambridge
Supervised by: Dr. Stephen Gow, University of Southampton

Report Date: 28/09/2021

AI3SD-Intern-Series:Report_14_Khondaker

Bayesian Optimisation in Chemistry
AI3SD-Intern-Series:Report_14_Khondaker
Report Date: 28/09/2021
DOI: 10.5258/SOTON/AI3SD0153
Published by University of Southampton

Network: Artificial Intelligence and Augmented Intelligence for Automated Investigations for Scientific Discovery
This Network+ is EPSRC Funded under Grant No: EP/S000356/1

Principal Investigator: *Professor Jeremy Frey*
Co-Investigator: *Professor Mahesan Niranjan*
Network+ Coordinator: *Dr Samantha Kanza*

Contents

1 Project Details

Title	Bayesian Optimisation in Chemistry
Project Reference	AI3SD-FundingCall3_016
Supervisor Institution	University of Southampton
Project Dates	01/07/2021 - 15/09/2021
Keywords	Bayesian Optimisation, Reaction Optimisation, Photovoltaic, Expected Improvement

2 Project Team

2.1 Project Student

Name and Title	Mr. Rubaiyat Mohammad Khondaker
Employer name / University Department Name	Department of Mathematics, University of Cambridge
Work Email	rmk47@cam.ac.uk

2.2 Project Supervisor

Name and Title	Dr. Stephen Gow
Employer name / University Department Name	Department of Chemistry, University of Southampton
Work Email	S.R.Gow@soton.ac.uk
Website Link (if available)	https://www.southampton.ac.uk/chemistry/about/staff/srg1f20.page

2.3 Researchers & Collaborators

- Prof. Jeremy Frey, Department of Chemistry, University of Southampton. https://www.southampton.ac.uk/chemistry/about/staff/jgf.page
- Prof. Mahesan Niranjan, Department of Electronics and Computer Science, University of Southampton. https://www.ecs.soton.ac.uk/people/mn1p07.

3 Publication

NB: This report was formalised into a journal paper which has been published in the Journal of Cheminformatics: Robustness under parameter and problem domain alterations of Bayesian optimization methods for chemical reactions[1]. This report serves as a preprint of that paper.

4 Lay Summary

Optimisation problems are common throughout chemistry, for example optimising the yield of a chemical reaction. Since chemical reactions can take hours or days to complete, it is infeasible to search through the entire set of possible conditions. Instead, a small subset of the search space is tested, with Design of Experiments (DOE)[2] and Generalised Subset Design[3] being two common techniques used to determine such a subset. Bayesian optimisation, an iterative global optimisation algorithm, has found success in hyperparameter tuning for machine learning models[4], where researchers face a similar issue of long model evaluation times. As a result, it has been recently applied to problems in chemistry, including the development of the EDBO (Experimental Design for Baysian Optimisation) optimiser[5], which was used throughout the project. We aimed at verifying the results obtained from the EDBO paper, and then testing the robustness of the optimiser by applying it to other datasets, including one outside the scope of its original domain of reaction yield optimisation.

5 Aims and Objectives

Using the provided EDBO optimiser, we intended to:

- Verify the results obtained from the EDBO paper, by running the optimiser on the provided datasets.
- Determine what effect altering the settings used for the optimiser, in particular acquisition function and batch size, had on its performance.
- Determine the robustness of the optimiser on other reaction yield datasets found from the literature.
- Test the optimiser on a problem different to reaction yield optimisation, in order to determine its applicability across problem domains.

6 Background

6.1 Bayesian Optimisation

Once an initial reaction pathway is determined, chemists wish to find reaction conditions that provide the best percentage yield, to minimise the amount of input material required to produce the desired product. Bayesian optimisation is a technique that aims to find optimal conditions while running only a small number of actual experiments.

Abstractly, we are given some black-box objective function $f(x)$ that we wish to minimise, which is expensive to evaluate. This could be in terms of time, resources or both. We wish to optimise the objective function while minimising the number of evaluations needed.

To do this, after taking some initial observations, we build a statistical model of the data. In the case of EDBO, this was done using a Gaussian Process Regression[6], which fits a mean and covariance function to the data (in lieu of a multidimensional gaussian random variable). This is termed the surrogate model.

After this, we need to choose the next point for evaluation, which is done using the acquisition function. This is an algorithm that takes in the surrogate model, and assigns a 'usefulness' measure to each point in the domain. There are many ways to define such a function; two examples are Expected Improvement (EI)[7] and Thompson Sampling (TS)[8].

As a brief summary of EI - Given the current best observed value x_+, and our objective function $f(x)$, define the Improvement Utility as

$$\mathbb{I}(\mathbf{x}) = \begin{cases} f(\mathbf{x}) - x_+ & \text{if } f(\mathbf{x}) \geq x_+ \\ 0 & \text{if } f(\mathbf{x}) < x_+ \end{cases}$$

Then, given the mean and variance at the point $\mathbf{x}$, the Expected Improvement is simply the average value of the Improvement Utility, according to the probability distribution defined by $\mathbf{x}$. In the case of a Gaussian Process Regression, an explicit symbolic expression can be found, which is described in the Shields paper. Note that the EDBO optimiser includes an exploration parameter δ that is subtracted from the Improvement Utility, and sets the Expected Improvement to 0 if the standard deviation σ is less than δ.

The key benefit of the acquisition function is that it's much easier to evaluate than the objective function, and so is easier to optimise. In particular, since the project was focused on discrete bayesian optimisation, due to only having access to tables of data, acquisition functions $acq(\mathbf{x})$ were optimised simply by evaluating them over the entire finite domain, and finding $\text{argmax}_{\mathbf{x}\in X} acq(\mathbf{x})$.

Then, we evaluate the objective function at the point selected by the acquisition function, and update the surrogate model with the new data. This concludes a round of optimisation.

Finally, the process is iterated until a fixed experiment budget is reached, a sufficiently high objective value is found, or it seems unlikely that further optimisation will provide useful improvement. This concludes a full run of the optimiser.

The following figure provides an example of a round of optimisation:

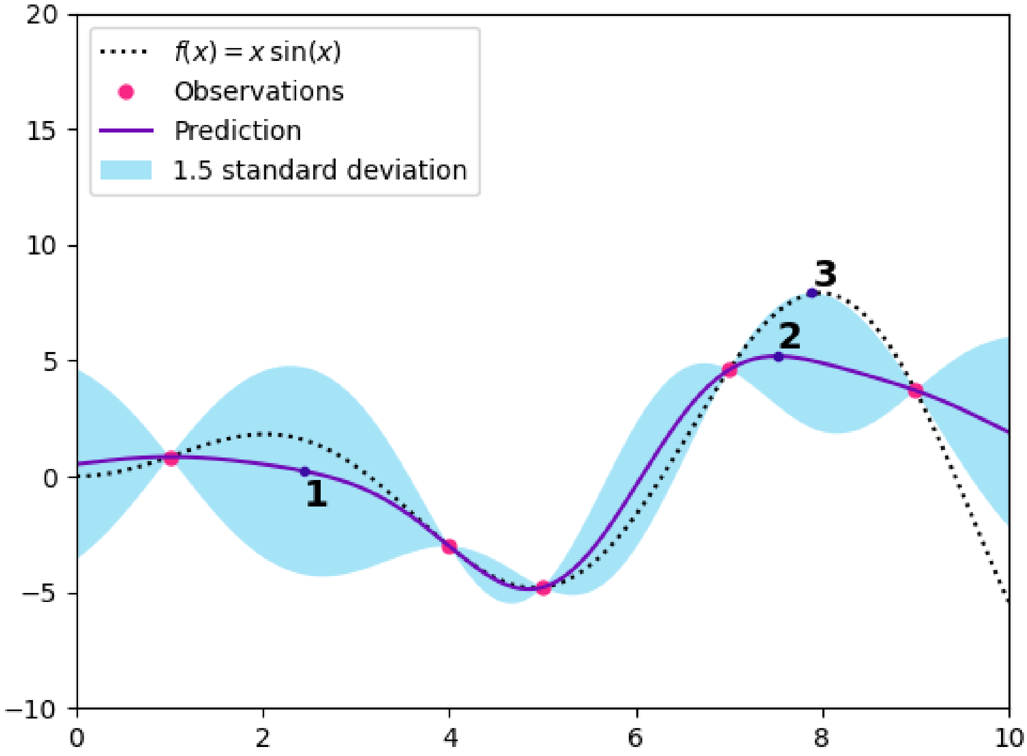

Figure 1: An example of what a single round of bayesian optimisation might look like, for optimising a one-dimensional function. The surrogate model assigns each point in the domain a mean, and an uncertainty - note that the previous observations, represented by pink dots, have zero uncertainty. From this, several different acquisition functions could be used - maximal uncertainty (1), maximal predictive mean (2), or maximal predictive mean + uncertainty (3). Depending on the choice of function, one of these points will end up being the next observation. The model would then be recomputed at the end of the round.

6.2 Batched Bayesian Optimisation

Conventionally, Bayesian Optimisation is done sequentially - the black-box function is evaluated at the chosen point, the model is immediately updated with the new data, and then the acquisition function is re-computed and maximised.

However, chemists have the ability to run multiple reactions in parallel, with techniques such as High-Throughput Experimentation[9] allowing the evaluation of hundreds of different conditions at a time. Thus, in order to maximise the information gained by the model in a given amount of time, batched Bayesian optimisation is carried out, where multiple points are chosen at each iteration to evaluate the objective function at.

Of course, this requires more guesswork on the part of the model at each iteration - the second evaluation point must be chosen without knowledge of the objective value at the first point. One common strategy, called Kriging Believer[10] is to estimate the value at the first point as the mean of the predictive distribution. Then, update the model with the estimate and determine the second point by re-computing and maximising the acquisition function. This is repeated until every desired point has been selected - the true values are then evaluated using the objective function, which replace the estimated values, and the round of optimisation concludes.

One disadvantage of this method is that it's computationally sequential, since computing the $k + 1$th point requires estimates of the previous k points. While this is necessary for a deterministic acquisition function like Expected Improvement, Thompson Sampling, which samples the posterior distribution given by the model and takes the point with highest objective value, can be parallelised easily by taking multiple samples of the distribution, and so has a marked speed increase over EI for large batch sizes. For example, investigating the Harvard dataset, a round of TS took approximately 4 times as long as a round of EI for a batch size of 10.

Note that while this does reduce the computational cost, which is useful for the purposes of the project given that only simulations of the optimiser are run, in lab applications the limiting factor is likely to be the evaluation of the objective function itself (e.g. running the chemical reaction with the given conditions).

7 Methodology

7.1 Numerical Encoding

Before starting the optimisation algorithm, the search space must be numerically encoded. In every dataset, this consisted exclusively of computing molecular descriptors for chemicals given by SMILES strings. Three techniques were used for this - Density Functional Theory (DFT) data (provided by authors of the EDBO paper); Mordred[11], an open-source molecular descriptor calculator; 512-bit Morgan Fingerprint Encoding[12], built-in to the rdkit Python module.

Once each chemical factor was encoded as a vector of descriptors, the encoding for the full configuration is obtained by concatenating all vectors. By listing the configuration vectors as the rows of a matrix, the search space is fully constructed. The EDBO optimiser then performs some pre-processing of the data by removing columns that are highly correlated, to save on memory.

7.2 Testing Optimiser performance

A full run of optimisation proceeds as follows

1. A dataset, acquisition function, batch size and experimental budget are provided.

2. The optimiser numerically encodes the search space.
3. The optimiser selects initial domain points to evaluate, by taking a random sample of the search space of size equal to the batch size. A seed can be supplied here to ensure reproducibility.
4. The optimiser conducts rounds of optimisation until the experimental budget is reached.
5. The top n values obtained are output, where n was normally either 1 or 5.

In each case, the optimiser was tested by doing 50 full runs with different starting seeds, to gauge performance across a range of initial domain points.

Due to a lack of access to physical labs during the project, the optimiser couldn't be tested in the lab with a given reaction and search space. Instead, simulations were run given a pre-existing database of conditions and objective values.

7.3 Dataset-specific configuration

Mordred encoding was not used for the Harvard Clean Energy Project dataset[13] due to time and memory constraints. For all other datasets, the size of the search space came from the multidimensional nature of the problem, with possibilities for each factor multiplying together. However, the Harvard dataset was uniquely one-dimensional due to the only factor being candidate organic photovoltaic chemical, meaning a sample size of 10,000 required computing molecular descriptors for 10,000 molecules. Mordred encoding in this case was prohibitively slow, so instead 512-bit Morgan Fingerprint encoding was used.

8 Results

8.1 EDBO Paper

This paper tested the optimiser on the Suzuki-Miyaura[14] and Buchwald-Hartwig Aryl Amination[15] datasets. Density Functional Theory (DFT) data provided in the EDBO Github repository (https://github.com/b-shields/edbo) was used for encoding, based on results from the paper that suggested this format minimised worst-case loss of the optimiser.

8.1.1 Batch Size

The authors of the EDBO paper remarked that Expected Improvement with a batch size of 5 performed equally well to sequential Expected Improvement with the same experiment budget of 50. So, after verifying that the code provided from the EDBO paper functioned properly on the provided Suzuki and Aryl Amination datasets, we wished to determine to what extent the batch size used affected the optimiser performance, for a range of different sizes. In particular, since a larger size meant fewer rounds of optimisation (with a similar total experimental budget) and more guesswork each round, we hypothesised that performance would degrade as batch size increased.

We opted to keep the experimental budget roughly constant, near 50. Of course, with a batch size of 3, it wasn't possible to give a budget of exactly 50, since 50 does not divide evenly into 3, and the batch size couldn't be altered during a simulation.

Therefore, we opted to give a budget as close as possible, of size 51. Similar strategies were adopted for other batch sizes that didn't divide exactly into 50:

- 3 had a budget of 51.
- 4 had a budget of 48.

- 6 had a budget of 48.
- 7 had a budget of 49.
- 8 had a budget of 48.
- 9 had two separate runs, with budgets of 45 and 54, since these were much further from 50 than any other batch size. In the end, it was decided to only display the 54 run, since performance didn't seem to be largely affected by batch size in any case.

We decided to test three different acquisition functions - EI, Thompson Sampling, and Random (which corresponded to ignoring the underlying surrogate model and instead picking points at random), to serve as a control. The results are displayed below.

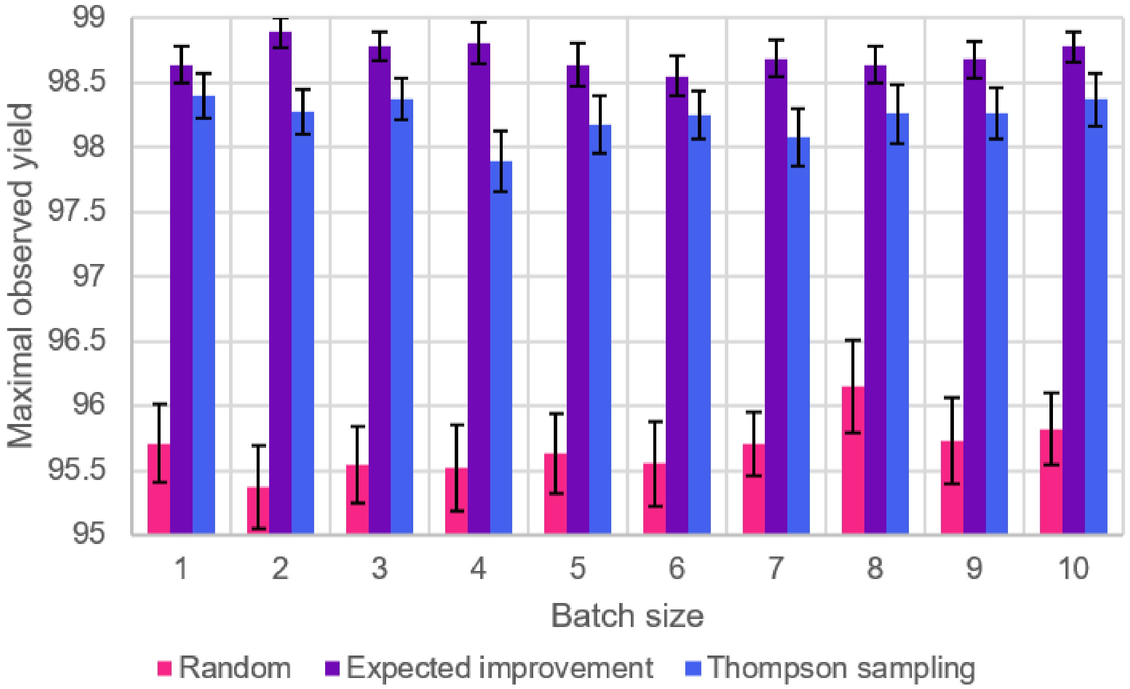

Figure 2: Graph of average optimiser performance (taken as the maximum observed yield after a full run, averaged over 50 runs), with standard error in the mean for the error bars, for the Suzuki reaction. Results indicate that there do not seem to be significant differences in performance across batch sizes. As an example, using Welch's t-test to compare sample means with sequential EI, $p > 0.05$ for every other batch size when continuing to use EI. EI does consistently outperform Thompson Sampling, but importantly both methods significantly outperform the random control ($p < 0.05$ in all cases, using Welch's t-test).

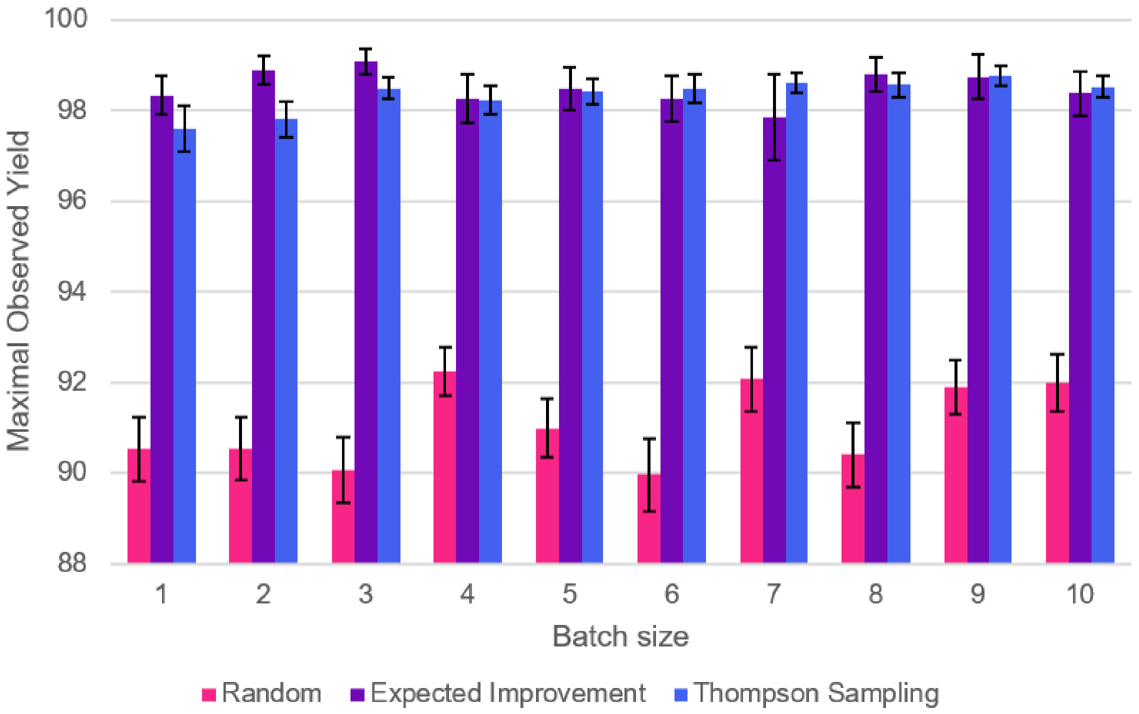

Figure 3: Graph of average optimiser performance (taken as the maximum observed yield after a full run, averaged over 50 runs), with standard error in the mean for the error bars, for the Aryl Amination reaction. As before, results indicate no significant dependence of performance on batch size, with both methods significantly outperforming the random control. Interestingly, Thompson Sampling occasionally outperforms Expected Improvement for certain sizes, which is likely an artifact of the search space of this particular reaction.

8.1.2 Altering set of initial experiments

After seeing optimiser performance was robust with respect to batch size, we moved on to testing to what extent the initial set of experiments given to the optimiser was important. The idea we explored was restricting this set to be chosen purely from the lowest 10% of experiments, ordered by reaction yield. These results are shown in Figure 3.

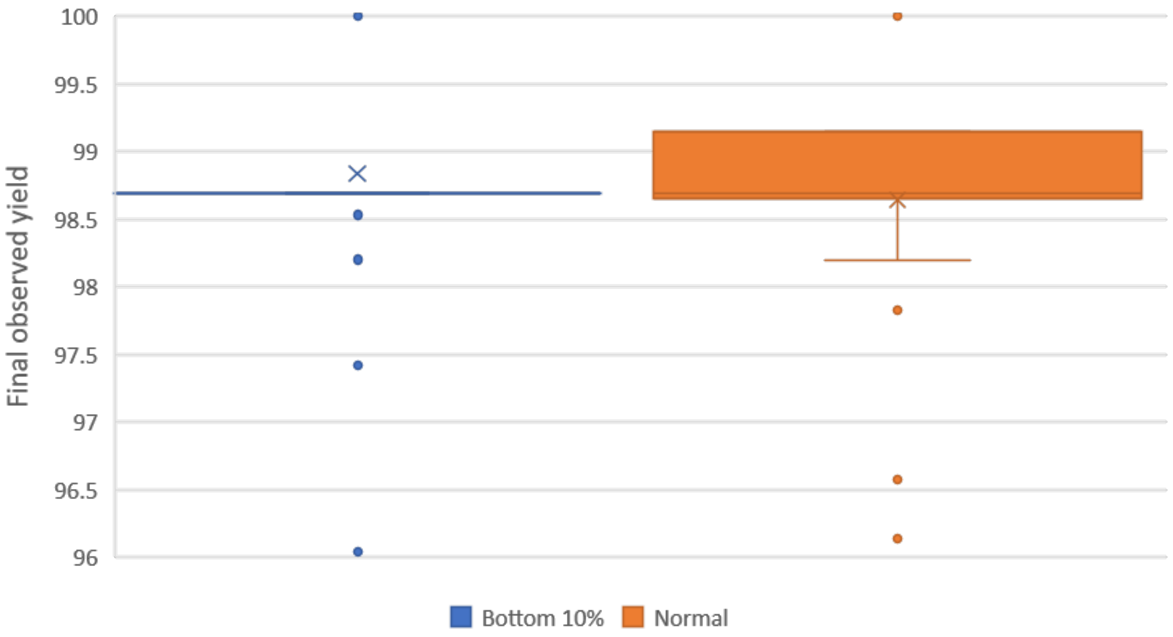

Figure 4: Box plots of optimiser performance for selecting from bottom 10% versus selecting normally. These were taken for the Suzuki-Miyaura reaction, with a batch size of 5, and an experimental budget of 50, and an acquisition function of Expected Improvement, with 50 full runs conducted. While average performance is very similar, it is interesting to note that the 'bottom 10%' method had identical lower and upper quartiles of 98.69%, suggesting the optimiser consistently found the same local maximum with this method.

8.2 Further reaction yield datasets

We then decided to further test the optimiser on two unseen datasets, still within the domain of reaction yield optimisation. In both cases, structures of relevant compounds were given visually in the paper withouot the associated SMILEES - to obtain this, they were drawn using an online tool (http://www.cheminfo.org/flavor/malaria/Utilities/SMILES_generator___checker/index.html) that allowed output of the required SMILES. Once this was obtained, Mordred was used to calculate molecular descriptors.

Firstly, we looked at a paper investigating the rate constants associated with different Iridium Photocatalysts[16], consisting of $24 \times 48 = 1152$ combinations in total. We decided to use an experiment budget of 50, with a batch size of 5, and again comparing Expected Improvement, Thompson Sampling and the random control, with 50 full runs of each. In this case, both Expected Improvement and Thompson Sampling found the global maximum quite frequently, and often found the second-highest maximum in other runs. Thus, we report the results in the following table:

Acquisition Function	#Runs finding global maximum	#Runs finding second-highest
Random	11	19
Expected Improvement	23	27
Thompson Sampling	19	28

Next, we looked at a paper that was investigating Palladium-catalysed cross-coupling reactions[9], and this is where we ran into our first issues with the optimiser. Multiplying together the possible Electrophiles, Nucleophiles, Catalysts and Bases gave $6 \times 11 \times 6 \times 8 = 3,168$ configurations in a full search space. However, the dataset provided by the paper contained only 1536 entries, indicating that some were excluded.

Again, we used an experimental budget of 50, with a batch size of 5, comparing Expected Improvement, Thompson Sampling and the random control, with 50 full runs of each.

Initially, it was coded so that combinations that were 'missing' were given an area count of 0, but this led to fairly poor performance by the optimiser. This could have been due to missing combinations being labelled as 0 interfering with the model, especially if a missing combination was close to an optimal one in the search space.

After this, the search space was modified to include only the 1536 combinations present in the dataset, by manually providing the allowable configurations. As indicated by the figure, this lead to markedly improved performance, which suggested that EDBO handled 'missing values' poorly overall and needed to be told the allowed domain points in advance. These results are shown below.

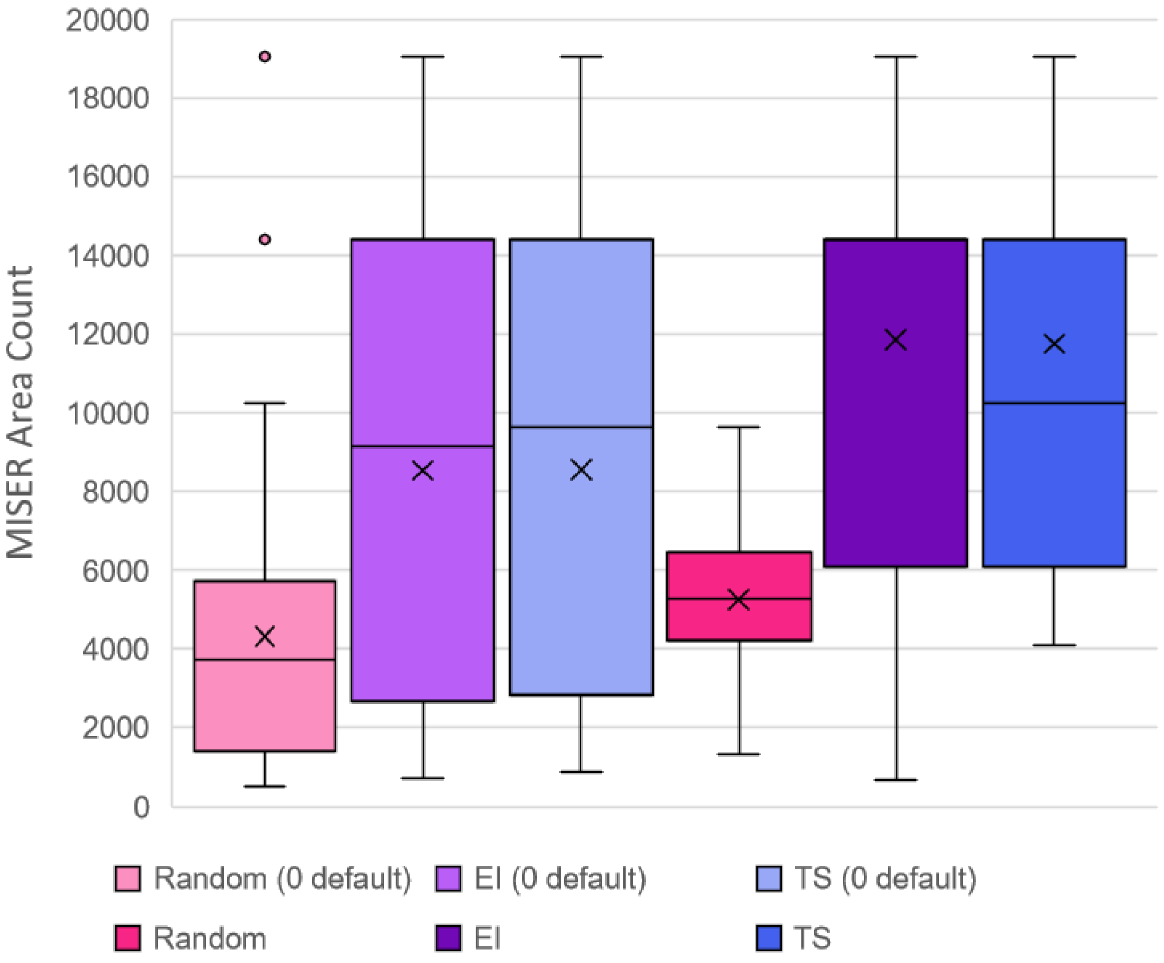

Figure 5: Box plots for the Palladium-catalysed cross-coupling dataset. The "0 default" labels indicate those runs where missing combinations were coded as 0. Note that for regular Expected Improvement, the upper quartile of the dataset was equal to the median, so an additional median line was not drawn.

8.3 Harvard Clean Energy Project

8.3.1 Initial Results

Finally, after thoroughly testing the optimiser on multiple reaction yield datasets, we decided to see how it would fare on a very different kind of optimisation problem - the Harvard Clean Energy Project (CEP) Dataset[13].

The CEP was a computational screening of 2 million+ molecules, with quantum chemistry calculations, in order to determine their theoretical power conversion efficiency (PCE) values for use in organic photovoltaics. This was both qualitatively and quantitatively different to what had been studied before, in that we were attempting to optimise a physical property of a single molecule, as opposed to the reaction yield from a combination of several molecules. In particular, the dataset was one-dimensional, since the only factor was which molecule was being tested for its PCE, as opposed to the previous multidimensional sets.

The full dataset was too large to be loaded on a single computer, so we instead took a specific random sample of the dataset of size 10,000 to investigate throughout. The numerical encoding of the dataset produced a unique challenge, because the search space was one-dimensional, with the only factor being the candidate organic photovoltaic chemical. This was in contrast to the multidimensional reaction yield datasets we had investigated previously, whose size came from multiplying together combinations of different factors and not from a single factor having thousands of possibilities. Therefore, Mordred encoding was prohibitively slow, leading us to use 512-bit Morgan Fingerprint encoding.

In this case, we used an experimental budget of 100, with a batch size of 10, comparing Expected Improvement, Thompson Sampling and the random control, with 50 full runs of each.

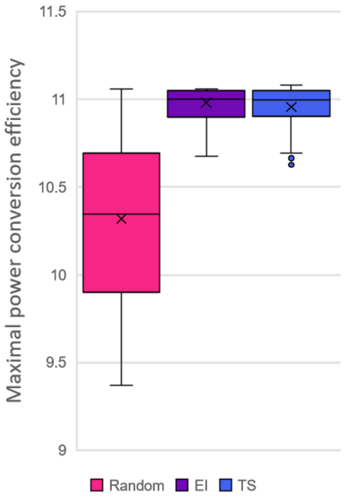

Figure 6: In this case, Expected Improvement and Thompson Sampling were remarkably similar, though both significantly outperformed random selection, which was surprising given that this was an entirely different problem for the optimiser to tackle.

Since the optimiser performed so well on this initial subset, we wanted to see whether we could find a different subset of the same size where it performed more poorly. Subsets were determined by taking a random sample, which could be seeded, so we searched a few hundred seeds - the worst seed we found had these results:

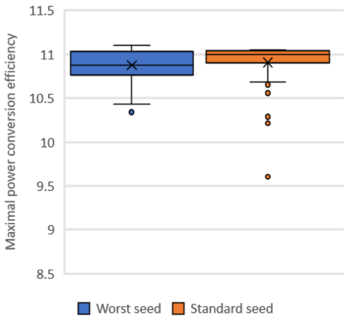

Figure 7: To compare the subsets, 50 full runs of the optimiser were conducted, with a batch size of 10, experiment budget of 100, and with Expected Improvement as the acquisition function. While optimiser performance is slightly worse in the left subset, it wasn't significantly so, indicating performance was fairly stable across different subsets.

8.3.2 Investigating different Acquisition Functions

After confirming that the EDBO optimiser could be applied successfully to this problem, we shifted to focusing on modifying the acquisition functions used in order to improve performance. In particular, we decided to look at improving the top 5 values returned by the optimiser, as opposed to merely the top value.

Our motivation for this was based on the calculated values being theoretical - therefore, it would be useful to have a large selection of molecules each of which had a good PCE as opposed to a single molecule with excellent PCE and the rest mediocre, to reduce the chances of relying heavily on an artefact of the PCE model.

The modification to the EI algorithm we explored was changing what value was being compared to for the sake of the Improvement Utility. Using the notation from earlier in the report, we altered the x_+ value used. Two modification strategies were evaluated:

- EI-k: Setting x_+ to be the kth highest value observed, with EI-1 representing ordinary Expected Improvement.
- E3I: Exploration Enhanced Expected Improvement[17]. In summary, it samples the surrogate model distribution multiple times, each time calculating Expected Improvement by setting x_+ to be the sample maximum, and then averaging the results. This tends to encourage more exploration of the sample space early on, and approaches normal Expected Improvement with more iterations.

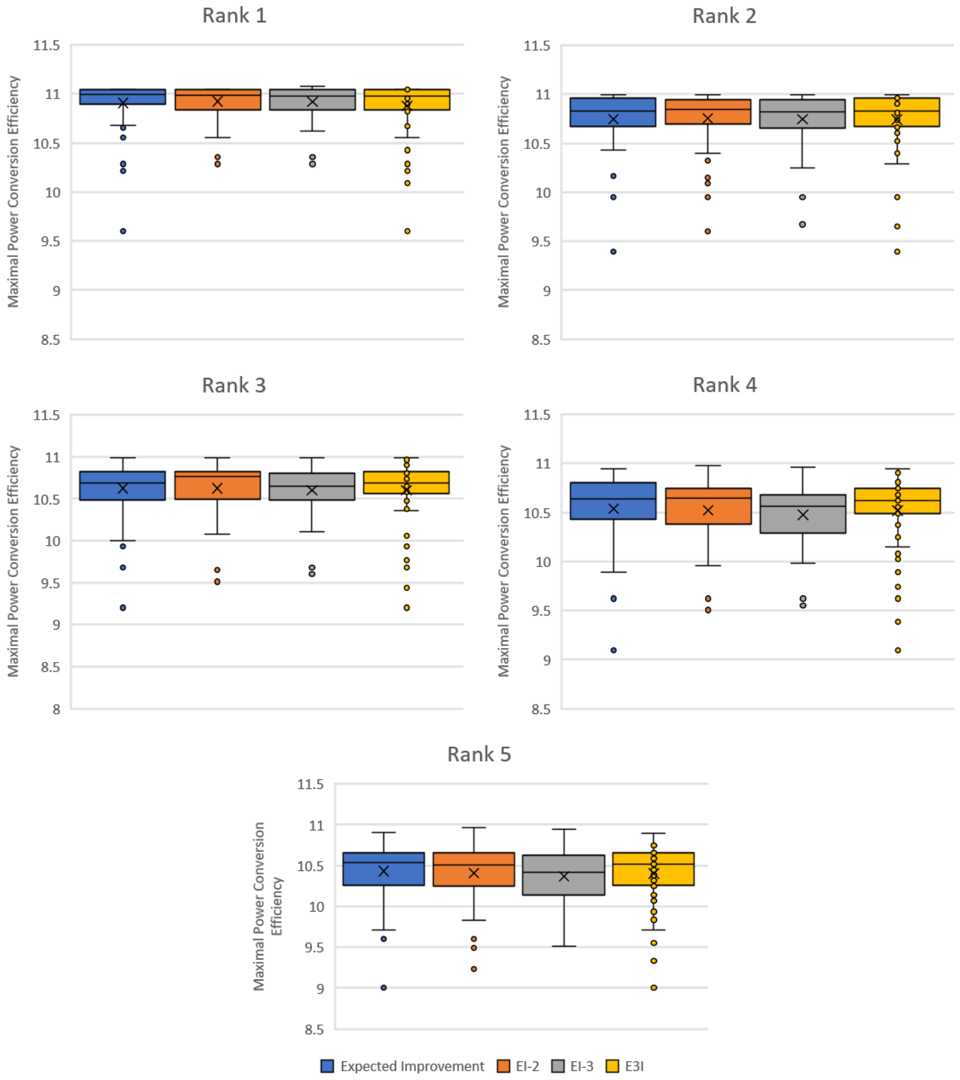

Figure 8: In each case, 50 full runs of the optimiser were conducted, with a batch size of 10, experiment budget of 100. As is evident from the figures, each strategy performed fairly similarly, and no alternative strategy consistently performed better than standard Expected Improvement.

9 Conclusions & Future Work

The EDBO optimiser did not only perform well with its original settings and datasets, but was robust to a wide variety of changes made. From the original paper, performance was not severely impacted by either modifying the batch size, nor altering the initial experiments selected by the optimiser. Furthermore, the algorithm continued to perform well on subsequent reaction yield datasets, showing its applicability in its original problem domain. Moreover, the algorithm performed well when used for a different class of problem - optimising a physical property of a single molecule - evidenced by its performance on the CEP dataset. In general, this suggests both EDBO and similar optimisers designed for one chemical problem domain may find applicability in a large number of other domains, with minimal configuration, lessening the need for problem-specific algorithms.

However, there are certainly limitations to the approach. In particular, as evidenced by the Palladium-catalysed cross-coupling dataset, the algorithm poorly handles 'missing' domain points, and instead requires a fixed search space determined before optimisation begins. Future work aimed at allowing this to instead adapt to both missing values and new candidate configurations would be greatly useful. In addition, efforts could be made to improve the acquisition function used to provide a selection of useful candidates, avoiding a sharp drop-off in quality across the top values.

Finally, it would be enlightening to test the optimiser in a physical lab setting to provide some hands-on data of its applicability to real-world reaction yield optimisation, especially if yields surpassing the literature could be accomplished. This would also allow the optimiser to be tested at a larger scale, with hundreds of thousands of potential conditions, since the experiment budget itself would remain manageable, which might provide insights unobtainable from the datasets shown here. Using a more powerful computer, or computing cluster, to analyse a larger sample of the CEP dataset would be a further test of the robustness of the algorithm in an unfamiliar problem domain, and could allow further room for experimentation on the acquisition function used.

10 Outputs, Data & Software Links

All code and data used for the project, as well as an explanatory poster, are available at `https://github.com/Pseudonium/edbo`.

This report has also been writen up as a journal paper which has been published in the Journal of Cheminformatics: Robustness under parameter and problem domain alterations of Bayesian optimization methods for chemical reactions[1].

References

[1] R. M. Khondaker, S. Gow, S. Kanza, J. G. Frey and M. Niranjan, *Journal of Cheminformatics*, 2022, **14**, 59.

[2] R. Carlson and J. E. Carlson, *Design and Optimization in Organic Synthesis: Second Revised and Enlarged Edition*, Elsevier, 2005.

[3] I. Surowiec, L. Vikström, G. Hector, E. Johansson, C. Vikström and J. Trygg, *Analytical Chemistry*, 2017, **89**, 6491–6497.

[4] J. Snoek, H. Larochelle and R. P. Adams, Advances in Neural Information Processing Systems, 2012.

[5] B. J. Shields, J. Stevens, J. Li, M. Parasram, F. Damani, J. I. M. Alvarado, J. M. Janey, R. P. Adams and A. G. Doyle, *Nature*, 2021, **590**, 89–96.

[6] C. E. Rasmussen and C. K. I. Williams, *Gaussian processes for machine learning*, MIT Press, Cambridge, Mass, 2006.

[7] J. Mockus, *IFAC Proceedings Volumes*, 1975, **8**, 428–431.

[8] D. J. Russo, B. V. Roy, A. Kazerouni and Z. Wen, 96.

[9] A. B. Santanilla, E. L. Regalado, T. Pereira, M. Shevlin, K. Bateman, L.-C. Campeau, J. Schneeweis, S. Berritt, Z.-C. Shi, P. Nantermet, Y. Liu, R. Helmy, C. J. Welch, P. Vachal, I. W. Davies, T. Cernak and S. D. Dreher, *Science*, 2015, **347**, 49–53.

[10] D. Ginsbourger, R. Le Riche and L. Carraro, in *Computational Intelligence in Expensive Optimization Problems*, ed. Y. Tenne and C.-K. Goh, Springer, Berlin, Heidelberg, 2010, pp. 131–162.

[11] H. Moriwaki, Y.-S. Tian, N. Kawashita and T. Takagi, *Journal of Cheminformatics*, 2018, **10**, 4.

[12] D. Rogers and M. Hahn, *Journal of Chemical Information and Modeling*, 2010, **50**, 742–754.

[13] J. Hachmann, R. Olivares-Amaya, S. Atahan-Evrenk, C. Amador-Bedolla, R. S. Sánchez-Carrera, A. Gold-Parker, L. Vogt, A. M. Brockway and A. Aspuru-Guzik, *The Journal of Physical Chemistry Letters*, 2011, **2**, 2241–2251.

[14] D. Perera, J. W. Tucker, S. Brahmbhatt, C. J. Helal, A. Chong, W. Farrell, P. Richardson and N. W. Sach, *Science*, 2018, **359**, 429–434.

[15] D. T. Ahneman, J. G. Estrada, S. Lin, S. D. Dreher and A. G. Doyle, *Science*, 2018, **360**, 186–190.

[16] V. Mdluli, S. Diluzio, J. Lewis, J. F. Kowalewski, T. U. Connell, D. Yaron, T. Kowalewski and S. Bernhard, *ACS Catalysis*, 2020, **10**, 6977–6987.

[17] J. Berk, V. Nguyen, S. Gupta, S. Rana and S. Venkatesh, Machine Learning and Knowledge Discovery in Databases, Cham, 2019, pp. 621–637.