Electronic Supplementary Information

A Chemist's Guide to Multi-Objective Optimization Solvers for Reaction Optimization

Aravind Senthil Vel, [a,*] Daniel Cortés-Borda [a] and François-Xavier Felpin [a,*]

[a] Nantes Université, CNRS, CEISAM UMR 6230, 2 rue de la Houssinière, 44322 Nantes, France.

∗ Corresponding authors.

e-mail: aravind.senthilvel@univ-nantes.fr; ORCID: 0009-0008-6563-5652

e-mail: fx.felpin@univ-nantes.fr; ORCID: 0000-0002-8851-246X; Website: http://felpin.univ-nantes.fr/

Table of content

1. Solvers

The Multi-Objective Optimization (MOO) solvers selected for this study are those known for their capability to approximate the entire Pareto front, having been validated through application in actual reaction optimization scenarios. Further, each of these solvers utilizes a Bayesian approach; starting with the construction of a surrogate model from initial data, the approach continues with the prediction of the next evaluation point through an acquisition function, with this cycle repeating throughout the optimization process.

1.1. MVMOO

MVMOO[1] (Mixed Variable Multi-Objective Optimization) is designed to manage both continuous and categorical variables. It utilizes a Gaussian process (GP) for the surrogate model and incorporates a customized covariance function based on the Gower distance metric to handle categorical variables. The Expected Improvement Matrix (EIM) – Euclidean distance serves as the acquisition function. This solver is available for use through installation in Python and can be accessed at https://github.com/jmanson377/MVMOO?tab=readme-ov-file

1.2. EDBO+

The Experimental Design via Bayesian Optimization (EDBO)[2] platform, initially developed for Single-Objective Optimization (SOO) by the research group led by Abigail G. Doyle, has been expanded to include EDBO+[3] for Multi-Objective Optimization (MOO). EDBO+ leverages a GP for the surrogate model and employs q-Expected HyperVolume Improvement as its acquisition function. It handles categorical variables through one-hot encoding. EDBO+ is accessible in Python via https://github.com/doyle-lab-ucla/edboplus and is also available as a web application (https://www.edbowebapp.com) making it particularly useful for chemists without coding skills and for use in non-autonomous setups. It is important to distinguish that EDBO and EDBO+ are two separate packages.

1.3. Dragonfly

Dragonfly[4,5] is a versatile optimization package capable of addressing both SOO and MOO problems. It supports a wide range of variable types, including continuous (with specified boundaries), discrete numeric, categorical, integer, and discrete Euclidean. Among its notable features are parallel evaluation and constraint handling. Specifically, for MOO, Dragonfly allows users to define the range of interest on the Pareto front, focusing the algorithm's efforts on this specified range rather than the entire front. For surrogate modeling, Dragonfly employs a GP, utilizing a Hamming kernel function for categorical variables. UCB (Upper Confidence Bound) or Thompson Sampling are used as acquisition functions. Uniquely, Dragonfly adopts random scalarizations for Pareto approximation, scalarizing multiple objectives into a single objective weighted by randomly chosen weights in each iteration. Modifications were made to two significant settings for our study. Given Dragonfly's original development for hyperparameter tuning optimization, the default model update occurs every 17 iterations to reduce computational time. However, considering our context, where experimental iteration times are relatively long, updating the model after every iteration is deemed more beneficial for accurate predictions, thus we adjusted the model update frequency to after each iteration (*build_new_model_every* = 1). Additionally, Dragonfly typically employs the Tschebyshev scalarization function by default, which is capable of identifying solutions within non-convex regions. The linear setting, according to theory, cannot access solutions in non-convex regions. To assess performance, we conducted tests using both settings. Dragonfly is available for Python installation at Dragonfly GitHub Repository (https://github.com/dragonfly/dragonfly). Jensen et al.[6] provide a well-structured tutorial for utilizing Dragonfly in both SOO and MOO, accessible at (https://github.com/anirudh-nambiar/make-it-system/tree/main/dragonfly_bayesopt_demo)

Following this, we highlight the advantage of integer and discrete Euclidean variable types supported by Dragonfly. For integer types, it confines its consideration to integer values within a specified range of continuous variable. For discrete Euclidean variables, it optimizes based on predetermined combinations of real values. An example of this could be specifying conditions (e.g. 30 °C and 10 min, 50 °C and 5 min, or 70 °C and 2 min) when optimizing temperature and residence time concurrently. However, this approach, while facilitating optimization, may reduce the robustness of exploration, leading to a potential bias in the search for optimal solutions. This method allows the algorithm to concentrate on a preselected set of combinations, rather than examining every potential permutation within the finite sets of values. This approach streamlines the optimization process by targeting scenarios most pertinent to the desired outcomes, leveraging prior knowledge to enhance efficiency.

1.4. TSEMO

TSEMO - Thompson Sampling Efficient Multi-objective Optimization - utilizes GP as surrogate models and is designed to handle only continuous variables. In the acquisition phase, individual GPs for each objective are sampled using Thompson Sampling. These samples are then processed with NSGA-II to approximate the Pareto front. The subsequent step involves selecting the next evaluation point anticipated to maximize the hypervolume contribution. TSEMO is available as MATLAB code on GitHub (https://github.com/Eric-Bradford/TS-EMO) and requires the Statistics and Machine Learning Toolbox as well as the Optimization Toolbox for its operation.

1.5. EIM-EGO

EIM-EGO (Expected Improvement Matrix for Efficient Global Optimization) leverages GP as surrogate models and adopts a matrix-based, multi-objective variant of the Expected Improvement acquisition function, as introduced by Liu et al[7]. Initially, they presented three methods to combine Matrix EI—Euclidean distance, Hypervolume, and min-max. In this study we employed the Euclidean distance, identical to the acquisition approach in MVMOO. Moreover, our testing revealed no significant performance variance across these transformation methods. The implementation of EIM-EGO is included in the PlatEMO[8], an open-source multi-objective optimization toolbox for MATLAB. It required the Statistics and Machine Learning Toolbox and is designed to process only continuous variables.

2. *In silico* models

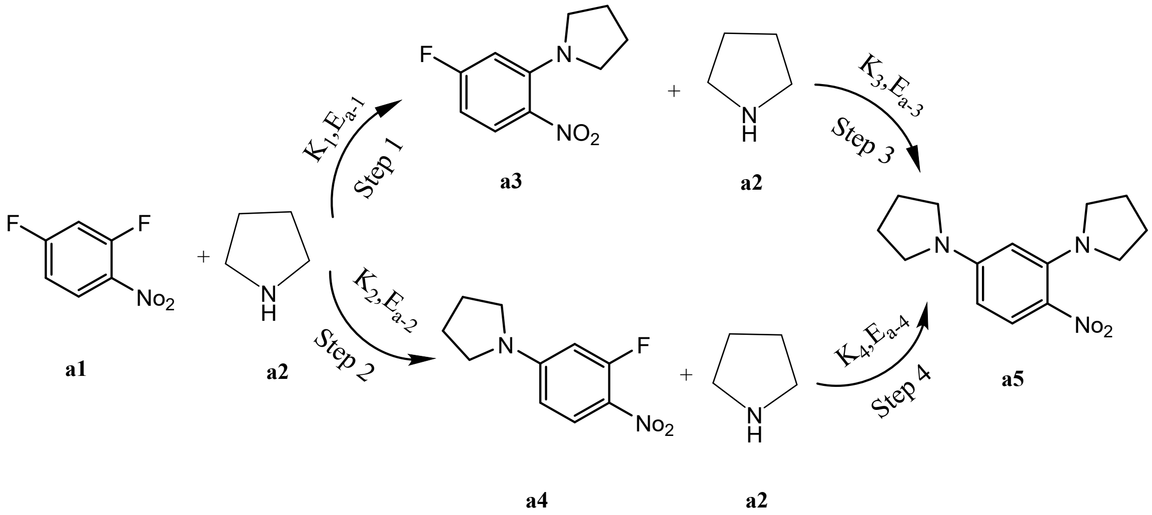
We used 10 distinct *in silico* problems for our analysis, all originating from three separate kinetic/surrogate models. The first kinetic model focuses on the S_NAr reaction involving 2,4-difluoronitrobenzene and pyrrolidine. The second model is a simulated kinetic study used by Reizman[9]. The third model draws from real experimental data we have used in the past, specifically for [3+3] cycloadditions.

2.1. S_NAr

The S_NAr reaction[10] – 2,4-difluoronitrobenzene **a1** reacts with pyrrolidine **a2** to produce one desired product **a3** and two undesired products **a4** and **a5** (Scheme. 1) - is used by Lapkin et al.[11] as a benchmark. The optimization involves maximizing Space time yield (STY) (g/h.L) and minimizing E-factor by varying four continuous variables - residence time **τ** [0.5-2 min], inlet concentration of a1 $\boldsymbol{C_{a1,i}}$ [0.1 -0.5 M], equivalents of **a2** $\mathbf{n_{a2}}$ [1-5 eq.], and reactor temperature **T** [30-120 °C]. We referred to this problem as S_NAr – 1.

$$\max_X (STY, -E)$$

$$where\ X = [\tau, C_{a1,i}, n_{a2}, T]$$



Scheme1. 2,4 – difluronitrobenzene reacts with pyrrolidine

Provided the inlet concentration of a1 $C_{a1,i}$ and the Equivalents of a2 n_2, inlet concentration of a2 $C_{a2,i}$ is calculated as

$$C_{a2,i} = n_{a2}C_{a1,i}$$

Given the reaction conditions, and the inlet concentrations of a1 $C_{a1,i}$ and a2 $C_{a2,i}$, outlet concentrations of the products can be identified by

$$d\tau = \frac{dC_{\_}}{-r_{\_}}$$

$C_{\_}$ represents the concentration and $r_{\_}$ respresents the reaction rate of species (a1-a5). The reaction rates are given as

$$r_{a1} =- (k_1 + k_2)C_{a1}C_{a2}$$

$$r_{a2} =- (k_1 + k_2)C_{a1}C_{a2} - k_3C_{a2}C_{a3} - k_4C_{a2}C_{a4}$$

$$r_{a3} = k_1C_{a1}C_{a2} - k_3C_{a2}C_{a3}$$

$$r_{a4} = k_1C_{a1}C_{a2} - k_4C_{a2}C_{a4}$$

$$r_{a5} = k_3C_{a2}C_{a3} + k_4C_{a2}C_{a4}$$

$k_{\_}$ represents the kinetic constants for each reaction (Step 1 to Step 4) which is calculated using the Arrhenius equation.

$$k_{\_} = k_{\_,ref}exp\left[-\frac{E_{a,r_{\_}}}{R}\left(\frac{1}{T} - \frac{1}{T_{ref} = 90^oC}\right)\right]$$

Table S1. Kinetic parameters used by Muller et al.[10] for the S_NAr reaction

Reaction	$k_{\_,ref}$ $10^{-2} mol^{-1} dm^3 s^{-1}$	E_a $kJ\ mol^{-1}$
1	57.9	33.3
2	2.70	35.3
3	0.865	38.9
4	1.63	44.8

Once the final concentration of all the species is identified, the Space time yield is calculated as the mass of product **a3** leaving per residence time.

$$STY = \frac{C_{a3,o} M_{a3}}{\tau} \left({}^{g}/_{h.L} \right)$$

E-factor is calculated as ratio of mass of waste (outlet undesired products and reactants) to the mass of desired product.

$$E = \frac{Q_{tot}\rho_{eth} + \sum_{n = 1, n \neq 3}^{5} M_{a(n)} C_{a(n),o} Q_{tot}}{M_{a3} C_{a3,o} Q_{tot}}$$

where $M_{a(n)}$ is the Molecular weight of the chemical species an, Q_{tot} is the total volumetric flow rate $\frac{V}{\tau}$, V is the reactor volume assumed to be 5 mL.

We expanded this model for multi-objective optimization by incorporating an additional objective: yield. The calculation for yield can be outlined as follows:

$$yield = \frac{C_{a3,o}}{C_{a1,i}} * 100$$

$$\max_{X} (Yield, STY, -E)$$

$$where\ X = \left[\tau, C_{a1,i}, n_{a2}, T \right]$$

This 3-objective optimization problem is referred to as S_NAr – 2.

To introduce a problem with two distinct categorical variables, we adjusted the kinetic parameters, specifically k_1 and k_2, with three levels for each: k_1= [49.21, 57.9, 66.58] and k_2 = [0.7353, 0.8650, 0.9947]. Increasing k_1 value favours the desired product **a3**, whereas higher k_2 values promote the undesired product **a5**. Thus, the combination $[k_1 = 66.58, k_2 = 0.7353]$ yields a potentially better solution, while $[k_1 = 49.21, k_2 = 0.9947]$ result in a less favourable outcome. This analysis is supported by Figure S1, which illustrates the Pareto front for each set of kinetic parameters, obtained from 100,000 random points. The difference in performance can be assumed to be of different solvents and catalysts. This *in silico* problem is referred to as $S_NAr - 3$.

$$\max_X (STY, -E)$$

$$where\ X = \left[\tau, C_{a1,i}, n_{a2}, T, k_1, k_2\right]$$

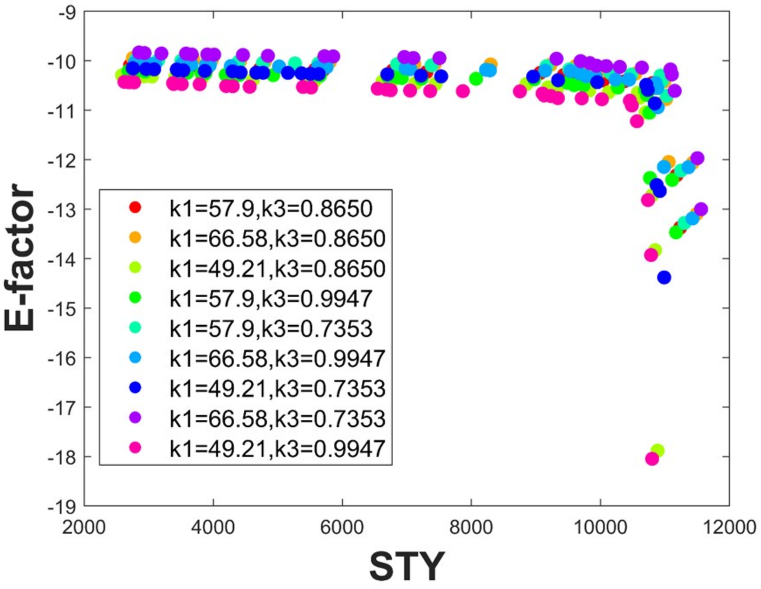


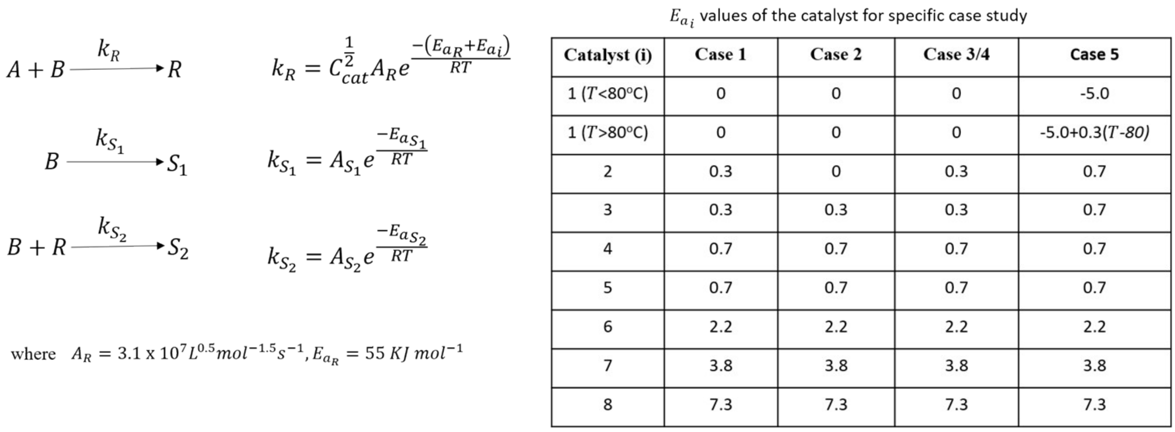
Figure S1. Pareto front for different combinations of kinetic parameters

2.2. Reizman

Reizman[9] introduced a simulated catalytic reaction, depicted in Figure S2, encompassing five distinct case studies. The optimization objective for each case study is the maximization of Yield and Turnover Number (TON), involving three continuous variables: residence time **τ** [1-10 min], temperature ***T*** [30-110 °C], ***C**_{cat}*** [0.5 -2.5 mol%], and one categorical variable catalyst Cat.[1,2,3,4,5,6,7,8]. This model has served as a benchmark in previous studies.[12,13]

$$\max_X (Yield, TON)$$

$$where\ X = \left[\tau, T, C_{cat}, Cat.\right]$$



$$A + B \xrightarrow{k_R} R \qquad k_R = C_{cat}^{\frac{1}{2}} A_R e^{\frac{-(E_{a_R}+E_{a_i})}{RT}}$$

$$B \xrightarrow{k_{S_1}} S_1 \qquad k_{S_1} = A_{S_1} e^{\frac{-E_{a_{S_1}}}{RT}}$$

$$B + R \xrightarrow{k_{S_2}} S_2 \qquad k_{S_2} = A_{S_2} e^{\frac{-E_{a_{S_2}}}{RT}}$$

where $A_R = 3.1 \text{ x } 10^7 L^{0.5} mol^{-1.5} s^{-1}, E_{a_R} = 55\ KJ\ mol^{-1}$

E_{a_i} values of the catalyst for specific case study

Catalyst (i)	Case 1	Case 2	Case 3/4	Case 5
1 (T<80°C)	0	0	0	-5.0
1 (T>80°C)	0	0	0	-5.0+0.3(T-*80)*
2	0.3	0	0.3	0.7
3	0.3	0.3	0.3	0.7
4	0.7	0.7	0.7	0.7
5	0.7	0.7	0.7	0.7
6	2.2	2.2	2.2	2.2
7	3.8	3.8	3.8	3.8
8	7.3	7.3	7.3	7.3

Figure S2. Simulation-based catalytic reaction

Each case study highlights unique scenarios (Table S2). Case study 1 (Reizman - 1) demonstrates that catalyst 1 is more effective compared to other catalysts. Meanwhile, in case study 2 (Reizman - 2), both catalyst 1 and catalyst 2 perform equally well and better than the other catalysts. Case study 3 (Reizman - 3) features a side reaction of Reactant B leading to the formation of the undesired product S_1. In case study 4 (Reizman - 4), Reactant B interacts with the desired product R, creating an undesired product S_2. Finally, case study 5 (Reizman - 5) shows catalyst 1 decomposing when the temperature rises above 80°C.

Table S2. Overview kinetic parameters for different case studies

Case	Catalyst effect	k_{S_1}	k_{S_2}
1	$E_{A_1} > E_{A_{2-8}}$	= 0	= 0
2	$E_{A_1} = E_{A_2} > E_{A_{3-8}}$	= 0	= 0
3	$E_{A_1} > E_{A_{2-8}}$	> 0[a]	= 0
4	$E_{A_1} > E_{A_{2-8}}$	= 0	> 0[b]
5	$E_{A_1} > E_{A_{2-8}}$	= 0	= 0

a: $A_{S_1} = 1\,x\,10^{12} s^{-1}, E_{a_{S_1}} = 100\ KJ\ mol^{-1}$

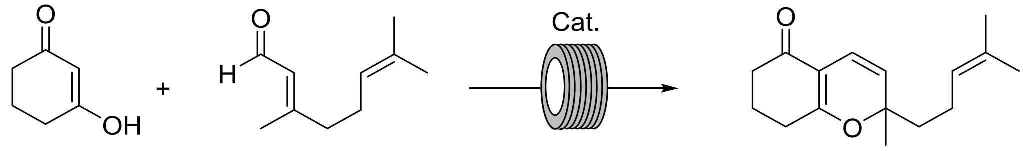
b: $A_R = 3.1\,x\,10^5 L^{0.5} mol^{-1.5} s^{-1}, E_{a_{S_2}} = 50\ KJ\ mol^{-1}$

2.3. [3+3] cycloadditions

This model is a data-based *in silico* model, derived from real experimental work featured in our prior publications[14,15] - 1,3-cyclohexanedione with citral leading to 2*H*-pyran (Scheme S2). For prediction, we utilized GP, which incorporates data from both our published and unpublished studies. The optimization involves optimizing two objective Yield (%) and Throughput (g/h), by adjusting one categorical variable, the catalyst, across five levels: ethanolamine, pyrrolidine, ethylenediamine, butylamine, and piperidine. Additionally, four continuous variables are varied: temperature T [25-50 °C], residence time **τ** [1-10 min], equivalents of **b2** n_{b2} [1-2 eq.], and catalyst loading n_{cat} [0.05 – 0.2 eq.]. This problem is referred to as [3+3] cycloadditions – 1.

$$\max_{X} (Yield, Throughput)$$

$$where\ X = \left[T, \tau, n_{b2}, n_{cat}, Catalyst \right]$$



b1 **b2**

Scheme S2. 1,3-cyclohexanone reacts with citral.

Utilizing this model, we also considered optimizing the same problem while fixing the catalyst to ethanolamine. This choice of catalyst is particularly intriguing due to the Pareto structure exhibiting a notable mix of convex and non-convex regions, reflecting the complexity often encountered in real scenarios. This specific optimization problem is designated as [3+3] cycloadditions – 2.

$$\max_{X} (Yield, Throughput)$$

$$where\ X = \left[T, \tau, n_{b2}, n_{cat} \right]$$

$$Catalyst = Ethanolamine$$

Yield values were used for training the surrogate model. Throughput values were then calculated based on the predicted yields. The data gathered for building this model originated from an optimization study, resulting in a dataset that lacks wide diversity across both categorical and continuous variables. However, for simulation studies aimed at evaluating the performance of optimization algorithms, the precision of the *in silico* model, while not required to be exact, should still fall within an acceptable range. To assess the model's predictive accuracy, we divided the dataset (totaling 216 data points) into 196 for training and 21 for testing, ensuring a stratified split. The model was trained using the Gaussian Process (GP) in

MATLAB with the *fitrgp* function, applying the default settings except for the kernel function, which was set to *ardmatern52*. and predictions were made for the test data to compare against actual values. This process resulted (Figure S3) in a mean absolute error of 4.7894 for the test data and 2.7375 for the training data, with r^2 values of 0.9026 and 0.9590, respectively.

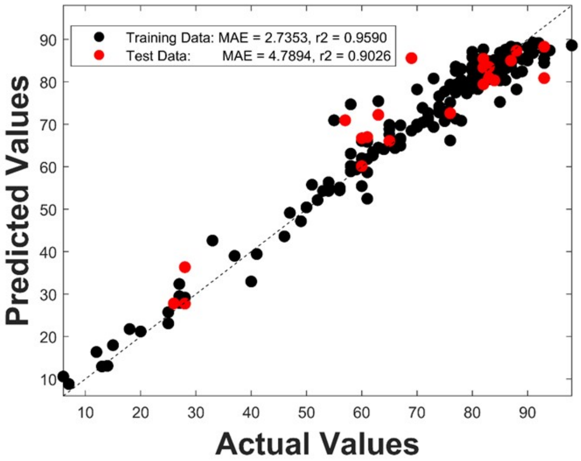


Figure S3. Parity plot for [3+3] cycloaddition model tested using 196 training points

The codes for the *in silico* models, along with the data used to build the [3+3] cycloaddition model, are made available on GitHub.

3. Metrics

3.1. Hypervolume

The hypervolume metric quantifies the space covered by the approximate Pareto set (PS) (current Pareto set), bounded by a predefined reference point. A higher hypervolume value signifies a higher quality of Pareto set. There exist several methods to compute hypervolume, and for our study, we adopted the Monte-Carlo approximation method, similar to the approach utilized by Bourne et al.[16] In the objective space, random points [number of objective*100,000] are generated. The hypervolume is then determined by the proportion of these points that are dominated by the approximate Pareto set. The reference point used is the anti-ideal point (composed of the worst possible outcomes for all objectives), adjusted by 0.01 times the range of the objective space, following the methodology used by Knowles[17].

3.2. Modified Inverted Generational Distance (IGD+)

The Inverted Generational Distance (IGD) is a metric that calculates the average minimum distance between the approximate Pareto points (identified as PS) and the true Pareto points (

PS^*). Originally defined as the Euclidean distance, IGD quantifies the average distance of each true Pareto point (PS^*) to its nearest approximate Pareto point (PS). It is expressed as follows:

$$IGD = \frac{1}{|PS^*|}\sum_{i \in PS^*} \min_{j \in PS} d_{i,j}$$

$$where\ d_{i,j}\ is\ the\ Euclidean\ distance\ from\ i\ to\ j$$

However, this distance calculation is Pareto non-compliant, meaning that at times, it does not accurately reflect the distance. Ideally, we want IGD to be minimized, but there are situations where it might incorrectly interpret the distance. To address this issue, a modified version of IGD (IGD^+), as proposed by Nojima et al.[18], is employed.

$$IGD^+ = \frac{1}{|PS^*|}\sum_{i \in PS^*} \min_{j \in PS} d^+_{i,j}$$

$$where\ d^+_{i,j} = \sqrt{\max(i_1 - j_1, 0)^2 + \ldots + max(i_{PS^*} - j_{PS}, 0)^2}$$

3.3. Worst attainment surface

The attainment surface represents the division between dominated and non-dominated areas within the PS. For any given optimization problem, the final attainment surface varies with each run. The "worst attainment surface" is identified among the set of attainment surfaces from 'n' runs and is characterized by the solutions forming the boundary of the worst-case scenario. In this study, the points constituting the worst attainment surface are derived from the Pareto points at the final iteration of all runs. Specifically, those points that remain non-dominated (when considering minimization) are deemed the worst attainment points for problems aimed at maximization. Given that this metric relies on visual representation, it is applicable primarily to scenarios involving 2 and 3 objectives (2D and 3D). However, interpreting results in 3D can be challenging, so in this study, we have chosen to use this metric exclusively for in silico problems with 2 objectives.

The codes utilized for metric calculations are provided on GitHub.

4. Results

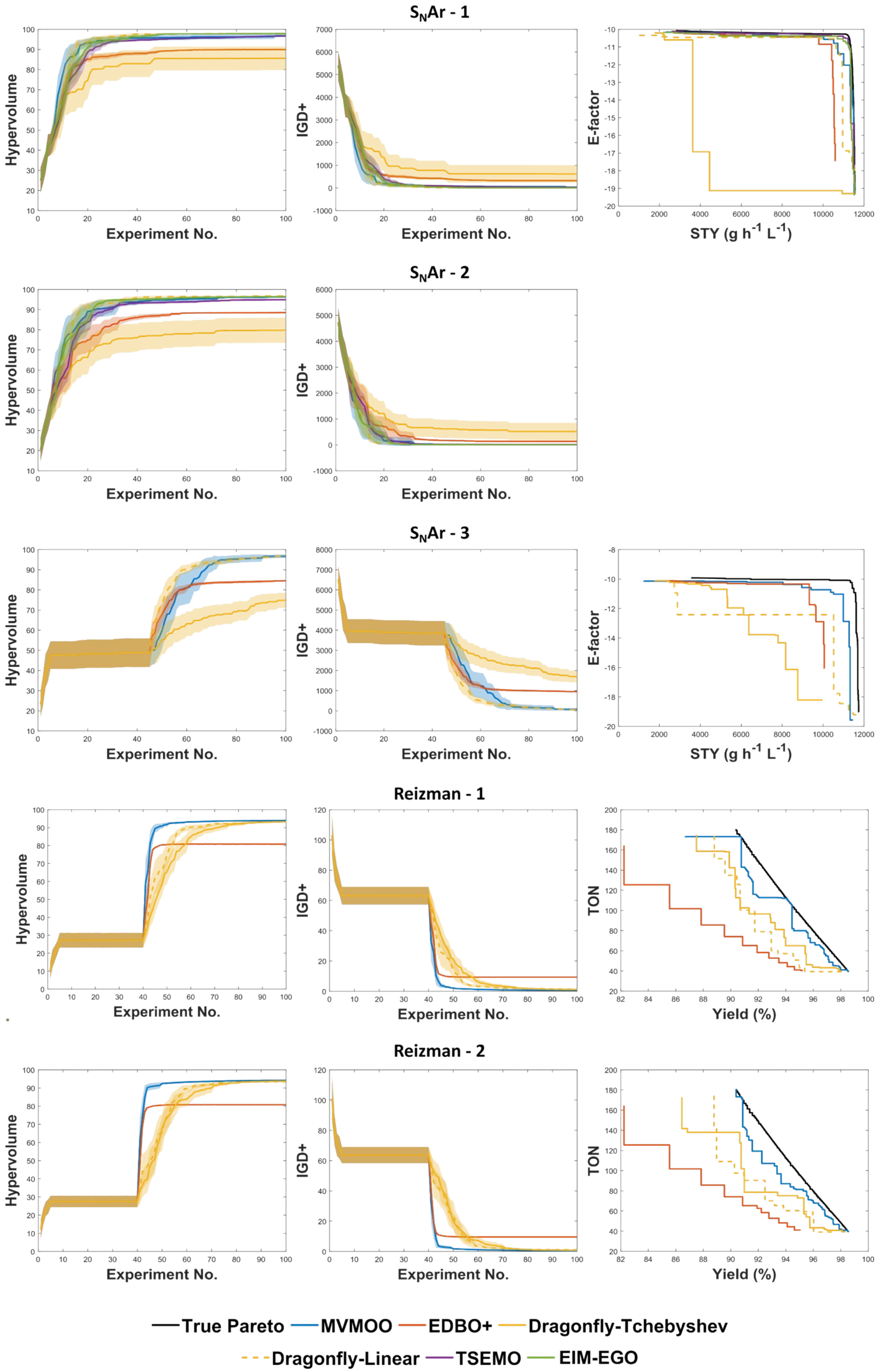
Each *in silico* problem was solved with solvers that could accommodate the problem's variable types. TSEMO and EIM-EGO, which handle only continuous variables, were not used for problems like S_NAr – 2, [3+3] Cycloadditions – 1, and Reizman – 1-5, which required handling both continuous and categorical variables. Additionally, EDBO+ does not naturally process continuous variables within a boundary, so these were formatted as discrete numeric values for compatibility. To ensure reasonable computational time, we opted for discretization that resulted in a search space of approximately 10,000 points. Each solver was run 21 times, with 100 iterations per run. Before optimization, initialization employed Latin Hypercube Sampling (LHS), conducting 5 experiments for each level of the categorical variables present or 5 experiments for problems with only continuous variables. Initial sampling is consistent for across solvers for any given *in silico* problem.

All the solvers were utilized in their default settings except for Dragonfly, which had specific adjustments detailed earlier in the Solver section on Dragonfly. Figure S4 displays the outcomes of the optimization for all the *in silico* problems. The first column corresponds to Hypervolume, the second to IGD+, and the third to the worst attainment surface metric. The worst attainment plot for S_NAr – 2 is omitted due to the challenges in interpretation. For the Hypervolume and IGD+, the plot corresponds to the average value of each iteration resulting from 21 runs, along with the 95% confidence interval.

Calculation of True Pareto

Identifying the true pareto set (PS*) for each *in silico* problem, as depicted in Figure 3, is crucial since it serves as a reference for metrics such as IGD+ and the worst attainment surface. In our study, PS* is determined as the collection of non-dominated points selected from the Pareto points generated by all solvers throughout all runs — essentially, the best points among all the best points.

The results of the different *in silico* problems solved using different solvers, the codes for calling different *in silico* problems, the data used for training [3+3] cycloadditions model, and the codes for calculating metrics are all made available in the GitHub repository (https://github.com/Aravind-vel/Multi_opt_reaction.git).



S$_N$Ar - 1
Hypervolume
IGD+
E-factor
Experiment No.
Experiment No.
STY (g h^{-1} L^{-1})
S$_N$Ar - 2
Hypervolume
IGD+
Experiment No.
Experiment No.
S$_N$Ar - 3
Hypervolume
IGD+
E-factor
Experiment No.
Experiment No.
STY (g h^{-1} L^{-1})
Reizman - 1
Hypervolume
IGD+
TON
Experiment No.
Experiment No.
Yield (%)
Reizman - 2
Hypervolume
IGD+
TON
Experiment No.
Experiment No.
Yield (%)
True Pareto
MVMOO
EDBO+
Dragonfly-Tchebyshev
Dragonfly-Linear
TSEMO
EIM-EGO

Reizman - 3

Reizman - 4

Reizman - 5

[3+3] cycloadditions - 1

[3+3] cycloadditions - 2

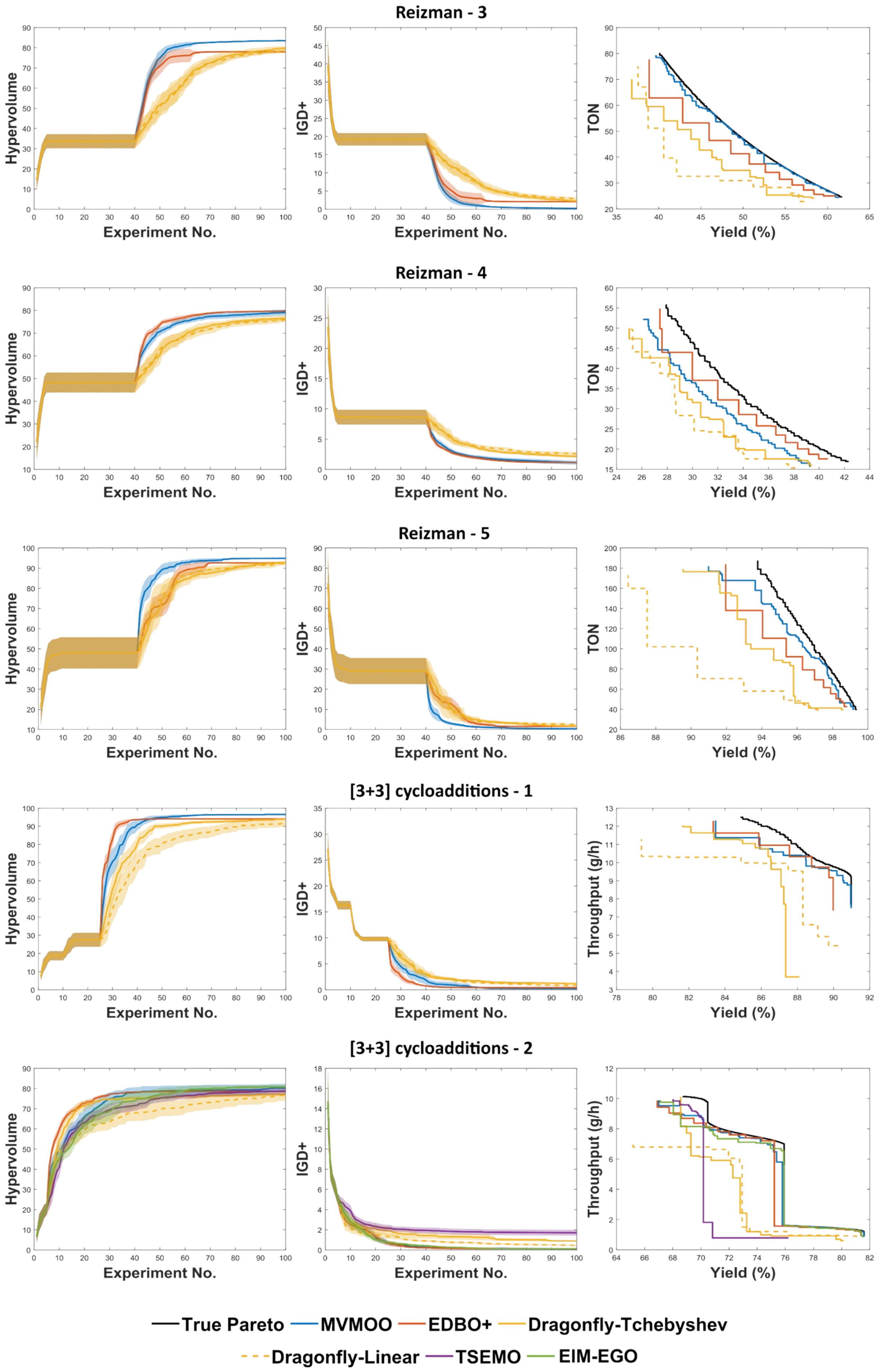


Figure S4. Results for *In Silico* Problems: Hypervolume (1st column), IGD+ (2nd column), Worst Attainment Surface (3rd column)

5. References

1. J. A. Manson, T. W. Chamberlain and R. A. Bourne, *J. Glob. Optim.*, 2021, **80**, 865–886.
2. 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.
3. J. A. G. Torres, S. H. Lau, P. Anchuri, J. M. Stevens, J. E. Tabora, J. Li, A. Borovika, R. P. Adams and A. G. Doyle, *J. Am. Chem. Soc.*, 2022, **144**, 19999–20007.
4. K. Kandasamy, K. R. Vysyaraju, W. Neiswanger, B. Paria, C. R. Collins, J. Schneider, B. Poczos and E. P. Xing, 2020.
5. B. Paria, K. Kandasamy and B. Póczos, 2019.
6. K. Y. Nandiwale, T. Hart, A. F. Zahrt, A. M. K. Nambiar, P. T. Mahesh, Y. Mo, M. J. Nieves-Remacha, M. D. Johnson, P. García-Losada, C. Mateos, J. A. Rincón and K. F. Jensen, *React. Chem. Eng.*, 2022, **7**, 1315–1327.
7. D. Zhan, Y. Cheng and J. Liu, *IEEE Trans. Evol. Comput.*, 2017, **21**, 956–975.
8. Y. Tian, R. Cheng, X. Zhang and Y. Jin, *IEEE Comput. Intell. Mag.*, 2017, **12**, 73–87.
9. B. J. Reizman, Thesis, Massachusetts Institute of Technology, 2015.
10. C. A. Hone, N. Holmes, G. R. Akien, R. A. Bourne and F. L. Muller, *React. Chem. Eng.*, 2017, **2**, 103–108.
11. K. C. Felton, J. G. Rittig and A. A. Lapkin, *Chem.-Methods*, 2021, **1**, 116–122.
12. L. M. Baumgartner, C. W. Coley, B. J. Reizman, K. W. Gao and K. F. Jensen, *React. Chem. Eng.*, 2018, **3**, 301–311.
13. N. Aldulaijan, J. A. Marsden, J. A. Manson and A. D. Clayton, *React. Chem. Eng.*, , DOI:10.1039/D3RE00476G.
14. K. E. Konan, A. S. Vel, A. Abollé, D. Cortés-Borda and F.-X. Felpin, *React. Chem. Eng.*, 2023, **8**, 2446–2454.
15. A. Senthil Vel, K. E. Konan, D. Cortés-Borda and F.-X. Felpin, *Org. Process Res. Dev.*, , DOI:10.1021/acs.oprd.3c00238.
16. P. Müller, A. D. Clayton, J. Manson, S. Riley, O. S. May, N. Govan, S. Notman, S. V. Ley, T. W. Chamberlain and R. A. Bourne, *React. Chem. Eng.*, 2022, **7**, 987–993.
17. J. Knowles, *IEEE Trans. Evol. Comput.*, 2006, **10**, 50–66.
18. H. Ishibuchi, H. Masuda, Y. Tanigaki and Y. Nojima, in *Evolutionary Multi-Criterion Optimization*, eds. A. Gaspar-Cunha, C. Henggeler Antunes and C. C. Coello, Springer International Publishing, Cham, 2015, pp. 110–125.