

DigiChemTree enables programmable light-induced carbene generation for on demand chemical synthesis

Abhilash Rana,¹ Ruchi Chauhan,¹ Amirreza Mottafeqh,² Dong Pyo Kim,² and Ajay K. Singh^{*1}

- [1] Abhilash Rana, Ruchi Chauhan, Ajay K. Singh*, Department of organic Synthesis & Process Chemistry, CSIR-Indian Institute of Chemical Technology, Hyderabad 500007 (India) E-mail: ajaysingh015@iict.res.in; ajaysingh015@gmail.com and Academy of Scientific and Innovative Research (AcSIR) Ghaziabad 201002 (India)
- [2] Center for Intelligent Microprocess of Pharmaceutical Synthesis, Department of Chemical Engineering, Pohang University of Science and Technology (POSTECH), Pohang 37673, Republic of Korea
- Abhilash Rana and Ruchi Chauhan equally contributed the work

Table of contents:

S1.	General	S3
	S1.1. Materials.	
	S1.2 Analysis.	
S2.	Automated system overview	S4
S3	Case study-1: Carbene insertion (C-O bond formation) single objective multi-variant auto-optimization.	S20
S4	Case study-2: Carbene insertion (C-S bond formation) single objective multi-variant auto-optimization.	S46
S5	Case study-3: Carbene insertion (C-N bond formation) single objective multi-variant auto-optimization.	S66
S6	Case study-4: Carbene insertion (C-C bond formation, cyclopropanation) single objective multi-variant auto-optimization.	S87
S7	Case study-5: Carbene insertion (C-C bond formation, cyclopropanation) single objective multi-variant auto-optimization.	S108
S8	Case study-6: Carbene insertion (N-C-O bond formation, oxazole) single objective multi-variant auto-optimization.	S128
S9	Case study-7: Carbene insertion (C=C bond formation, fumarate) single objective multi-variant auto-optimization.	S142
S10.	Case study-8: On demand chemical synthesis.	S156
S11.	Troubleshooting	S208
S12.	Supplementary references	S209

Supplementary Methods

S1. General

S1.1. Materials: Most of the reagents and chemicals are bought from Sigma-Aldrich and used as such without any further purification. Common organic chemicals and salts were purchased from AVRA chemicals, India. Water used for the experiments was deionized water (18.2 mS conductivity). All work-up and purification procedures were carried out with reagent-grade solvents in air. Analytical thin-layer chromatography (TLC) was performed using analytical chromatography silica gel 60 F254 precoated plates (0.25 mm). The developed chromatogram was analysed by UV lamp (254 nm). PTFE (id = 500 μm) tubing, T-junction, high-purity PFA tubing was purchased from Upchurch IDEX HEALTH & SCIENCE. The photo-batch reactor bought from Lelesil Mumbai India was slightly modified for the continuous flow reaction. Visible light (Blue, Red, Green LED) reactor was bought from the Smartchemsynth Machine Pvt. Ltd, Hyderabad.

S1.2. Analysis: High-resolution mass spectra (HRMS) were obtained from a JMS-T100TD instrument (DART) and Thermo Fisher Scientific Exactive (APCI). Nuclear magnetic resonance (NMR) spectra were recorded on a Bruker 600, 500, 400 or 300 MHz in CDCl_3 or $\text{DMSO-}d_6$ solvent. Chemical shifts for ^1H NMR are expressed in parts per million (ppm) relative to tetramethylsilane (δ 0.00 ppm). Chemical shifts for ^{13}C NMR are expressed in ppm relative to CDCl_3 (δ 77.0 ppm). Data are reported as follows: chemical shift, multiplicity (s = singlet, d = doublet, dd = doublet of doublets, t = triplet, q = quartet, quin = quintet, sext = sextet, m = multiplet), coupling constant (Hz). GC/MS analysis was conducted on a Shimadzu technology GCMS-QP2010 instrument equipped with a HP-5 column (30 m \times 0.25 mm, Hewlett-Packard) and inbuilt MS 5975C VL MSD system with triple axis detector.

S2. Automated system overview

S2.1. Automated system components

Table S1. Outlines the hardware components utilized in the automated platform, with color coding representing distinct modules (green: operation module, blue: reaction module, yellow: analysis module, purple: fraction sample collection).

Type	Manufacturer	Module	Function	Communication protocol	Details in section
Syringe pump	HARVARD	Operational module	Solvent delivery	Serial, USB	2.2.1
HPLC pump	KNAUER	Operational module	Solvent delivery	LAN	2.2.1
Continuous Syringe	SMART CHEM SYNTH	Operational module	Solvent delivery	Serial, USB	2.2.1
Power supply	OWON	Reaction module	Reaction	Serial, USB	2.2.2
Blue LED	SMART CHEM SYNTH	Reaction module	Reaction	Serial, USB	2.2.2
In-line IR	METTLER TOLEDO	Analysis module	Sample analysis	Serial, USB	2.2.3
3D printer	ENDER	Collection module	Fraction sample collection	Serial, USB	2.2.4

S2.2. Automated system hardware module.

S2.2.1 Operational module

The HPLC pumps (figure S1a) and Harvard syringe pump (figure S1b) were directly interfaced to the main computer via an RS-232 interface or LAN. These pumps will help in introducing reactants and reagent solutions into the micro reactor with varied flow rate as the central computer's prescribed flow rate (fr). This control protocol also includes starting/stopping of the pump. The

central computer and the HPLC pump communicated through serial communication using ASCII code to exchange information about flow rate, operational status, and duration. This facilitated the transmission of essential details from the main computer to the pumps, enabling them to commence their operations as required. The syringe pump (figure S1c) The parameters for the serial communication of the syringe pump with the computer are given as follows: Baud Rate - 9600, Flow control – none. The basic control protocols include starting/stopping the pump and setting different parameters. Arduino (figure S1d) connected to syringe pump directly interfaced to the main computer via an RS-232 interface or LAN and helps in controlling flow rate. It has inbuilt proximity sensor and pulse will generate through Arduino program.

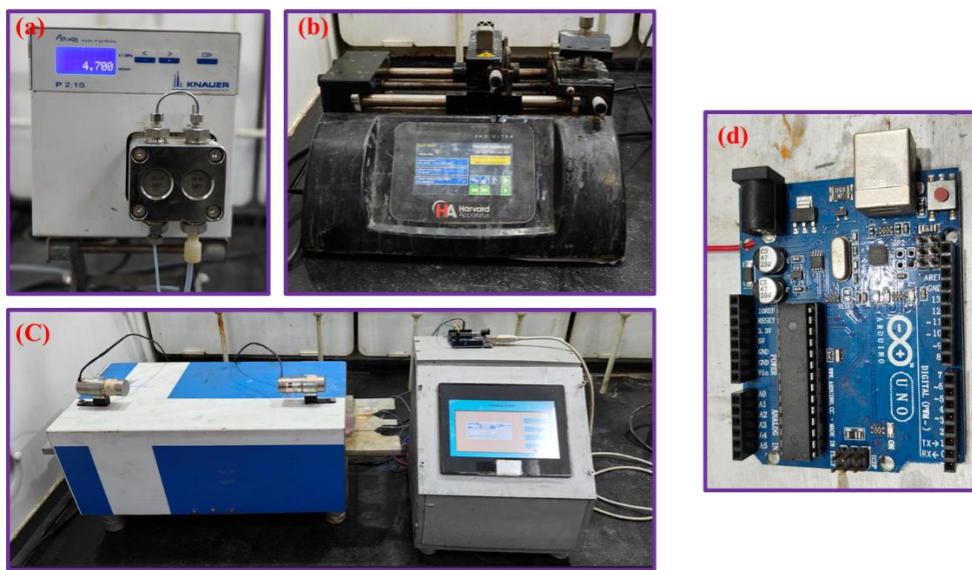


Figure S1. Picture of working syringe pump and HPLC pump (a) HPLC pump (Knauer) (b) syringe pump (Harvard) (c) Continuous syringe pump with Arduino (smarthchemsynth) (d) Arduino.

S2.2.2 Reaction module

We employed a commercially accessible cylindrical blue LED (figure S2b) for the photochemical reaction, utilizing a programmable power supply manufacture by Owon (figure S2a) to regulate the lamp's power (measured in watts) by managing both current and voltage. The Blue LED produces light in the range of wavelength 450-495 nm and it can be operated up to 5 A current, and 14.5 v voltage corresponds to about 70 w wattage. Additionally, rotating fan was attached to the bottom of cylindrical blue LED light to maintain the room temperature throughout the reaction. The central computer and the power supply communicated through serial communication using ASCII code to exchange information about varied voltage, current and operational status.

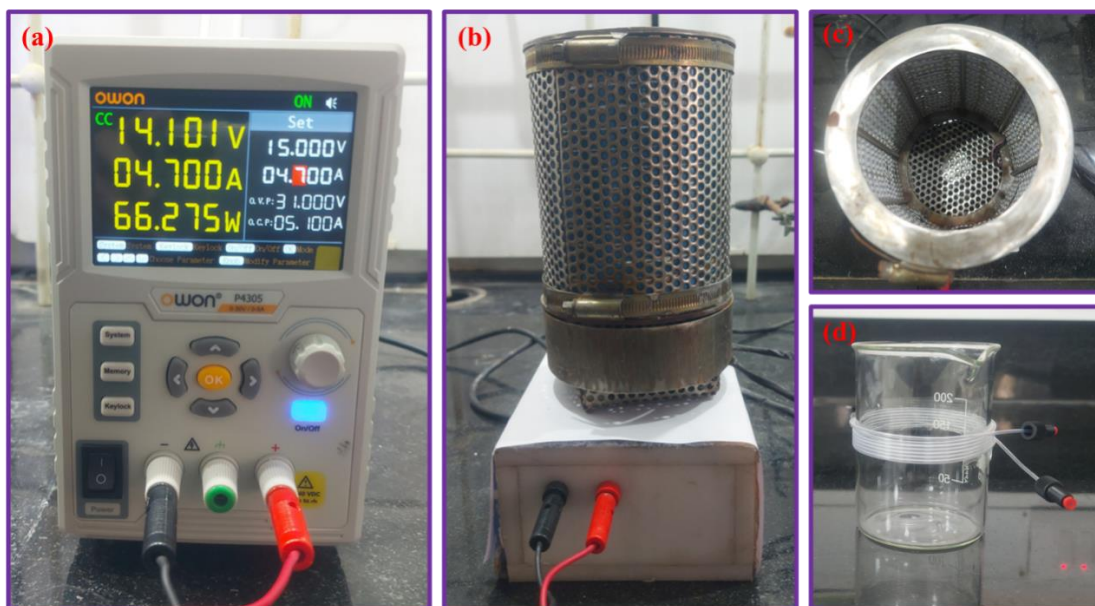


Figure S2. Manual designed photo-flow reactor; (a) power supply (b-c) blue LED module with cooling fan (1-70W, 460 nm max); (d) 1mL PFA coiled reactor (1.2-meter length, OD = 1.58 mm and id = 1.0 mm).

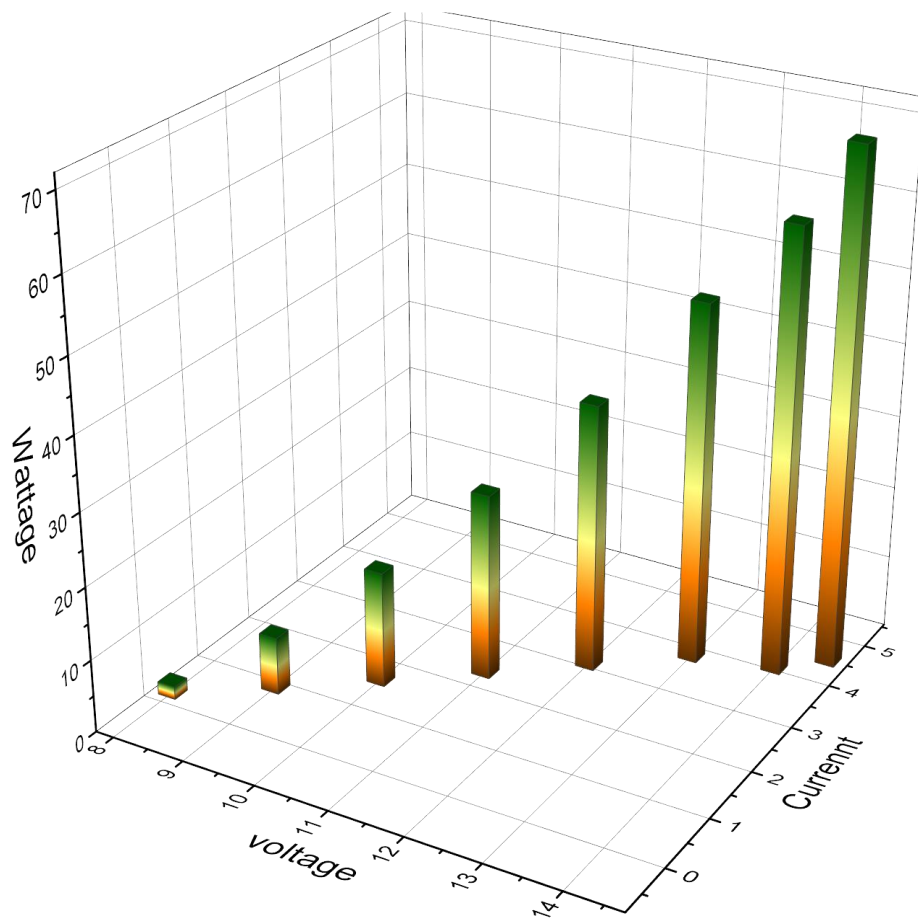


Figure S3. 3D graph elucidating the correlation among voltage (V), current (Amp), and the emission of blue light (Wattage).

S2.2.3 Analysis module

In-line FTIR analysis

In-situ reaction analysis was accomplished by recording IR (functional group, shifting peak in product) spectra using a benchtop ReactIR 15 base unit with DS Micro Flow Cell (Mettler toledo). It works through sensing the amount of infrared radiation after passing through a sample, generating a 2D graph with absorption on the Y-axis and wavenumber on the X-axis, having optical range from 4000 to 650 cm^{-1} . The system collected spectra with 16 scans at resolution 4 cm^{-1} .



Figure S4. Actual photograph of ReactIR 15 flow cell setup.

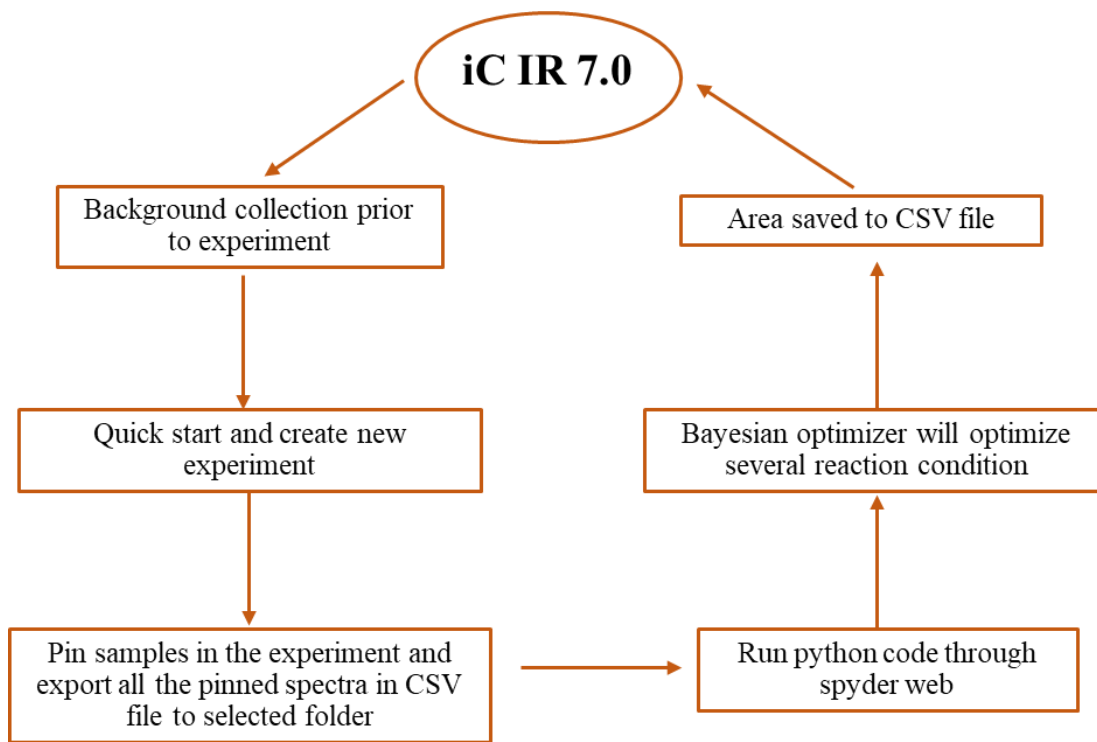


Figure S5. Work flow of In-line FTIR measurement when running reaction optimization.

S2.2.4. Collection module.

We employed a commercially accessible 3D printer as an automated collector. The Python program on the central system interfaced with the 3D printer and give commands to the stepper motors on the X, Y, and Z axis of the printer for movement. This information is transmitted via USB serial communication. PFA tubing (od = 1/16, id = 1mm) is connected to the printer nozzle and move along x-axis, and a test tube stand is placed on the printing platform which move along Y-axis to collect the fraction of reaction mixture output (see figure S6). This configuration enables precise control of the 3D printer, facilitating tasks such as sample collection, reactor stabilization, and washing in diverse scenarios. With this 3D printer setup, we can collect maximum 22 fraction of out coming reaction mixture in one run of experiment.

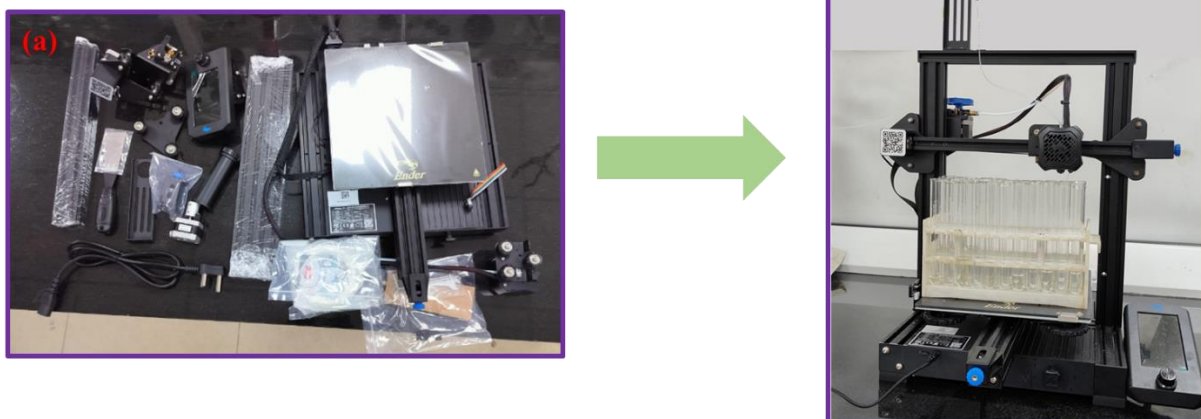


Figure S6. (a) Original photograph of dismantle 3D printer; (b) assembled photograph of 3D printer as auto-fraction collector.

S2.3. Automated system software

Table S2. List of the software required for the automated system.

Type	Version	Function
Windows 10	Professional	Control system
Python	3.9, 32-bit	Control of platform
Spyder web	3.7	run Bayesian Optimizer algorithm
Arduino IDE	1.8.19	Control temperature
iC IR	7.0	Control benchtop In-line FTIR

S2.4. Bayesian optimization

The machine learning algorithm utilized in this project is Bayesian Optimization (BO), implemented through the open-source package. Bayesian optimization is an algorithm which uses Baye's theorem to find the maximum of objective. It is a robust machine learning technique, especially suited for optimizing chemical reactions during the initial phases of process development its efficiency lies in the ability to navigate extensive reaction spaces effectively, forecasting optimal reaction conditions with high yields by assessing only a limited number of experiments. The selected initial points are the variable space (reaction conditions), the objectives to attain (e.g., maximize yield and/or conversion, minimize costs), the total number of experiments to perform, the information of communication port (e.g., pump = COM 7, printer = COM 9) and the overall experimental cost (indication how many cycles of experiment to be performed in each run). The initial points are chosen through Latin hypercube sampling for Euclidean and integral variables, while for discrete numeric values, selection is done uniformly at random. The variable space encompasses the range within which the objective is intended to be optimized. It accommodates both discrete and continuous variables. In Bayesian optimization incorporation of multiple variables of the same type (in a single line, as in String varA, varB, varC) is feasible, leaving the determination of the most effective among them to the model. Each time the BO calls the objective function, the platform is run, and the parameter settings are sent to the platform. The model then waits until the result is sent back before updating and deciding on the next experiment.

S2.4.1. Step-by-step instructions for setting up the experiments and operating the automated system.

Step 1

Background collection: This marks the beginning of our new experiment. We'll start by selecting the wavenumber range and then proceed to click on "collect background." It's crucial to ensure

that our solvent is running through the In-line IR. Upon clicking "collect background," a new dialogue box will appear, where we'll select 128 scans and proceed to collect the background.

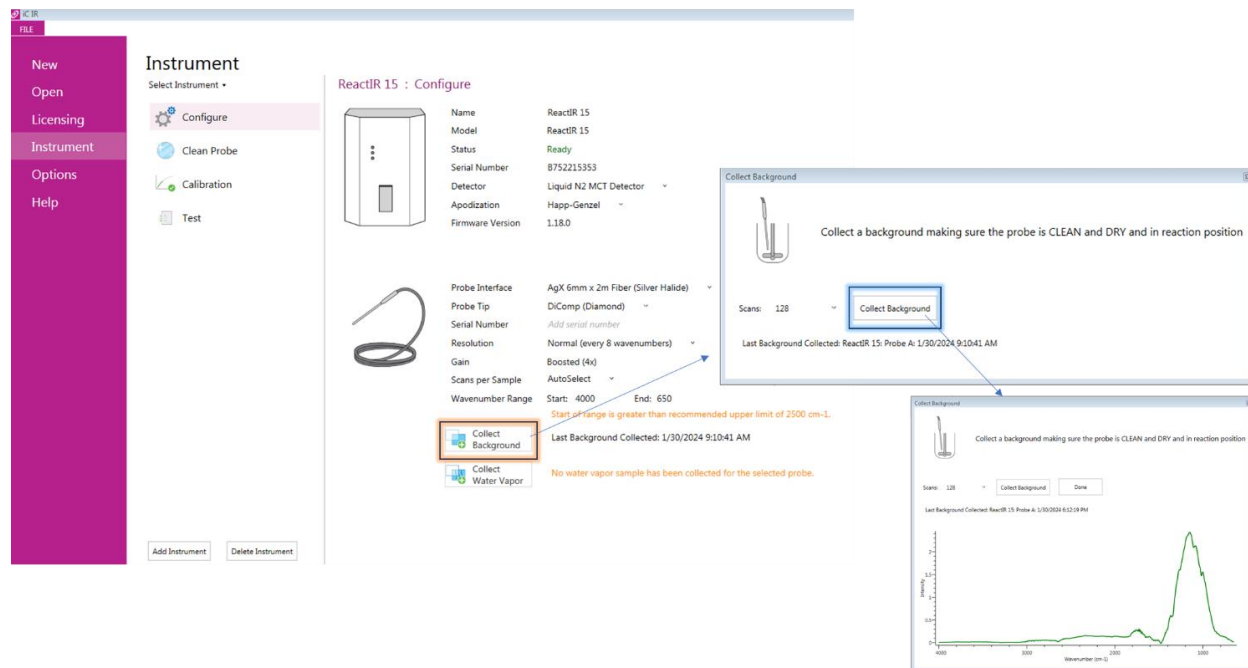


Figure S7. Home page of the iC IR 7.0 containing step-by-step instructions on how to take solvent background before starting experiment.

Step 2

After background collection create new experiment by clicking on tab this dialogue box will appear.



Type your experiment name and select the folder to save your experiment at your desired location then select duration (1h-2day) and scan of sample interval (15sec) for one run.

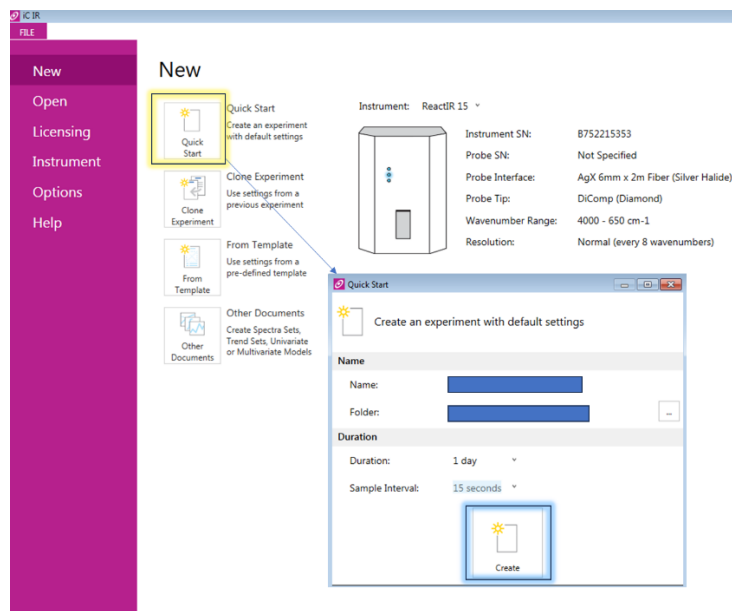


Figure S8. User Settings page, where the name of the experiments, folder and the location they are saved to are defined.

Step 3

Upon creating the new experiment, you'll encounter this dialogue box. Firstly, **(a)** pin all the samples by clicking on "smart pin," followed by clicking on "pin 6 samples." Secondly, **(b)** export all the pinned spectra by clicking on "spectra" and then exporting all the files in CSV format only, which will be in an Excel file. Thirdly, **(c)** after clicking on "export," a new dialogue box will appear showing all the CSV files exported from the In-line IR. Create a text document in that folder and save it with a AB.py extension to save the Python code.

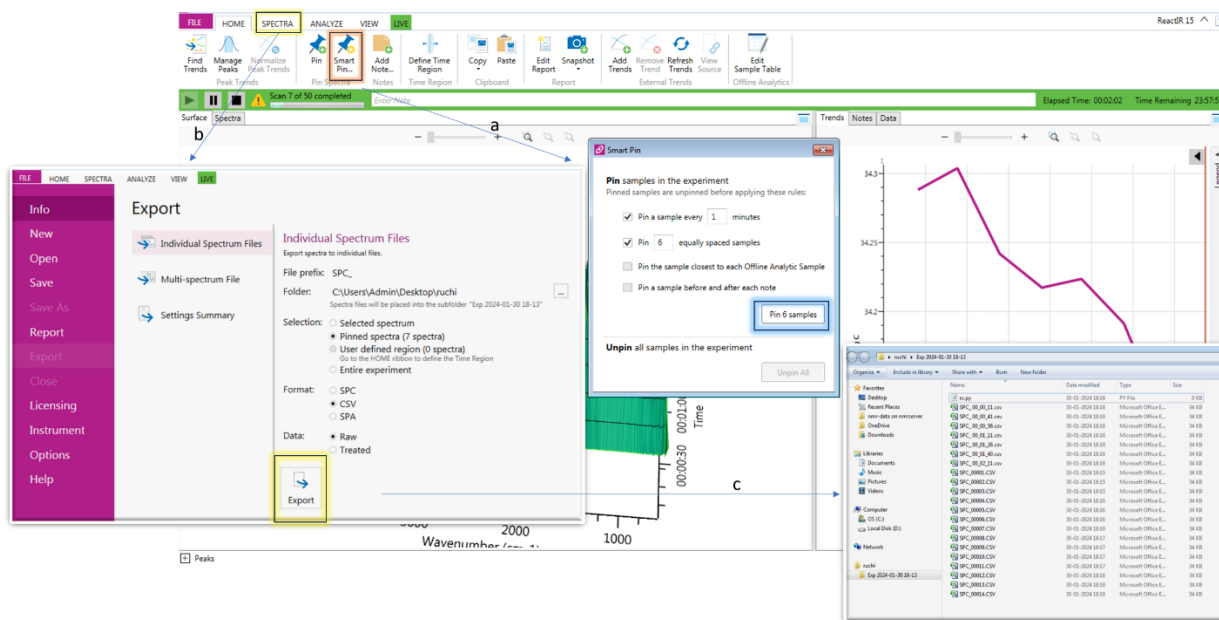


Figure S9. User Settings page, where in first step (a) pin all the sample, in second step (b) export spectrum file in CSV format and in third step (c) you can see all the spectrum file in excel format.

Step 4

Open spyder web app in central system where we will write our python code or open the previously saved python code. In this python code we have to define path in **mypath** and then we write the communication port for syringe pump, power supply and auto collector. Then we will give reaction condition variable such as flow rate (residence time), voltage, current and select no of cycles (1-22) in one run.

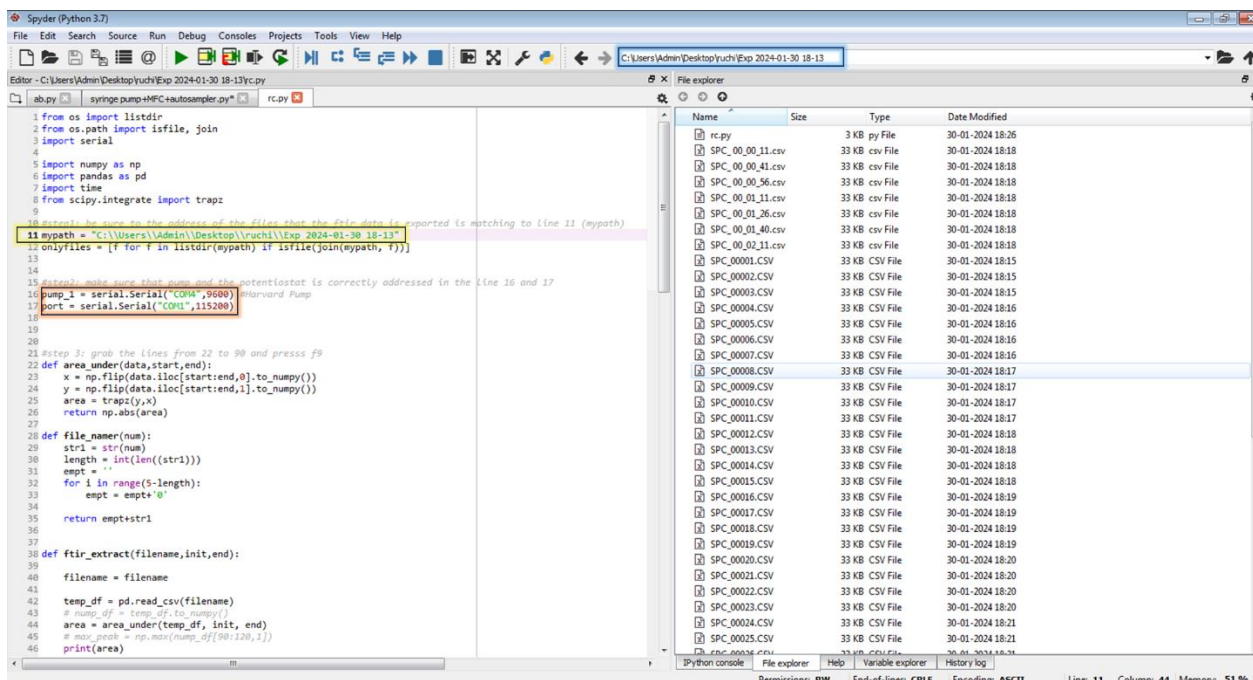


Figure S10. Reaction parameter settings page, where you will define the file path and communication port with the instruments attached to central system.

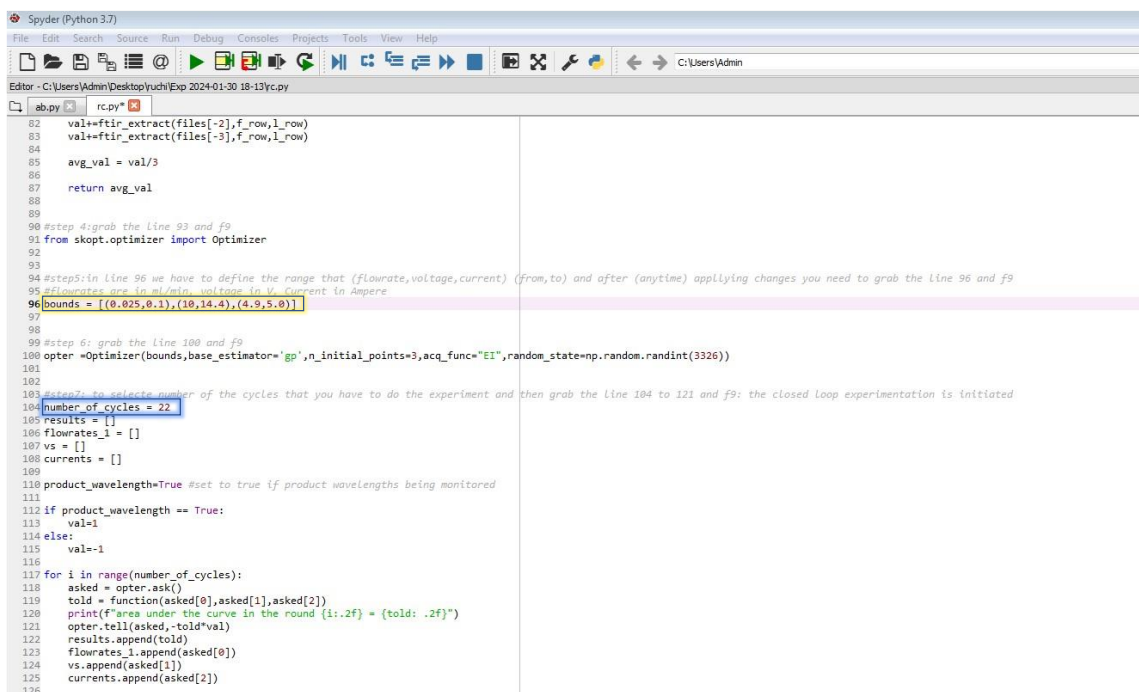
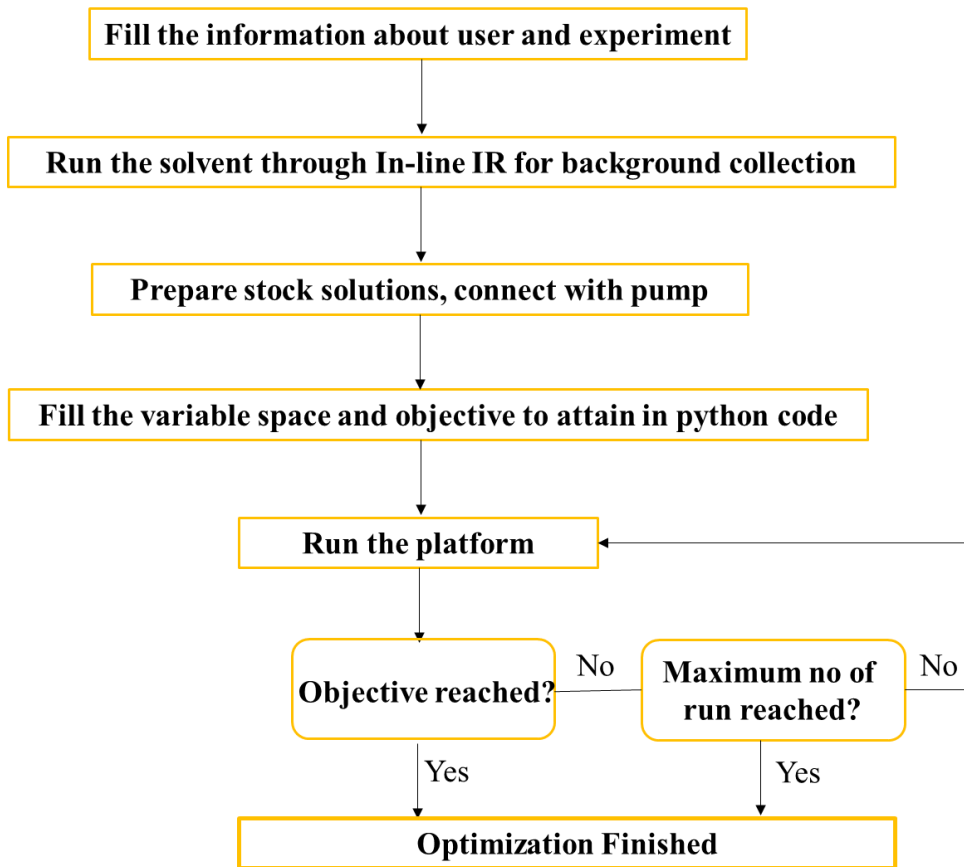


Figure S11. Reaction parameter settings page, where you will define the boundary of flow rate, voltage, current and communication port with the instruments attached to central system.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
1		flowrate	voltages	currents	area-resulwatt																			
2	0	0.102368	14.08762	4.968193	0.08926	65.321																		
3	1	0.104702	14.27705	4.963058	0.054675	68.01																		
4	2	0.112409	14.2009	4.941309	0.111543	67.218																		
5	3	0.066445	14.47059	4.901553	0.114956	70.786																		
6	4	0.125	14	4.9	0.025144	69.682																		
7	5	0.066	14.39582	5	0.049361	69.801																		
8	6	0.066156	14.49334	4.902017	0.138695	70.825																		
9	7	0.125	14.5	5	0.096633	71.413																		
10	8	0.066	14.16656	4.905818	0.045791	65.241																		
11	9	0.069702	14.5	4.901639	0.11917	70.892																		
12	10	0.124355	14.5	4.929517	0.097555	71.456																		
13	11	0.125	14.27383	4.930149	0.037493	68.225																		
14	12	0.066	14	4.982374	0.106264	63.896																		
15	13	0.082366	14	4.906796	0.025258	63.644																		
16	14	0.124833	14.13302	5	0.059424	65.357																		
17	15	0.105972	14.5	4.943228	0.091063	70.934																		
18	16	0.094977	14.5	4.9	0.113633	70.818																		
19	17	0.066536	14	5	0.076992	70.673																		
20	18	0.115403	14.5	4.9	0.141776	70.629																		
21	19	0.087432	14.00118	4.95748	0.099524	63.033																		
22	20	0.094102	14.5	4.9	0.252021	70.557																		
23	21	0.106282	14.49823	4.999693	0.233256	71.301																		
24																								

Figure S12. Example layout on excel file automatically generated from iC IR and bayesian optimizer.

S2.4.2. General Procedure for optimization with automated system.



List of starting material (2a-2g): Previously, reported protocol has been used to synthesized the diazo-compound.¹

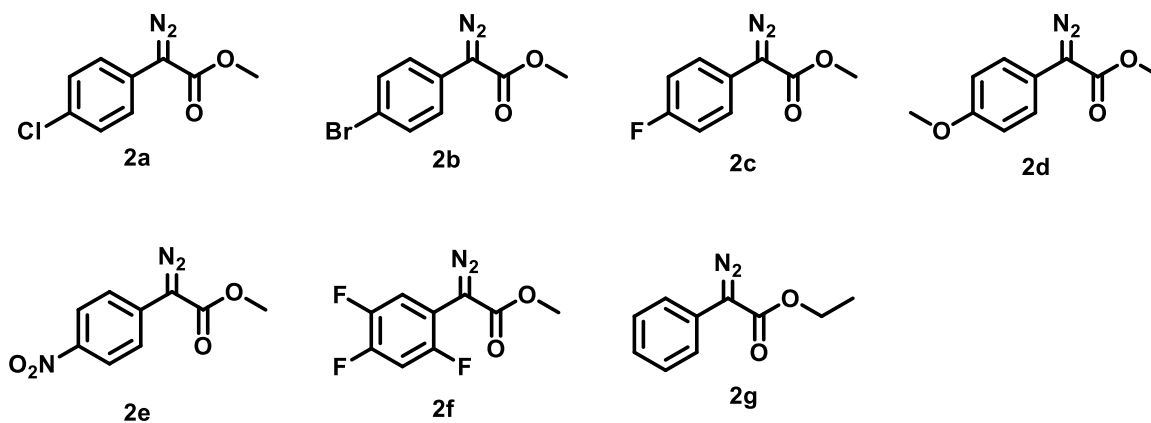
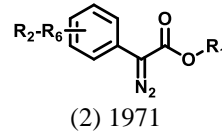
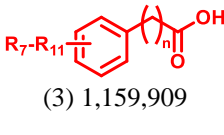
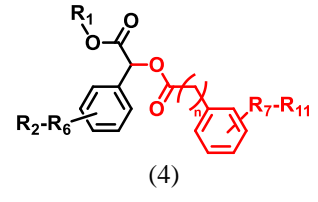
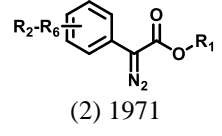
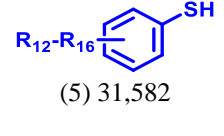
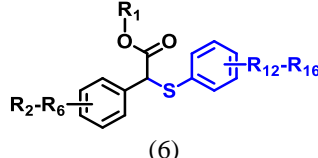
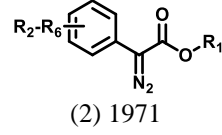
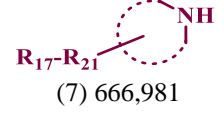
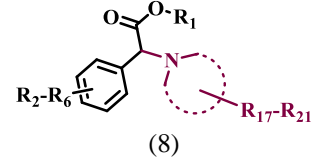
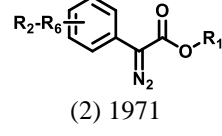
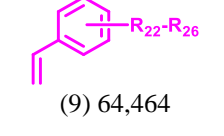
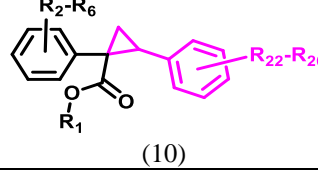
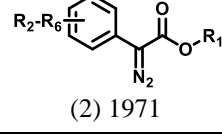
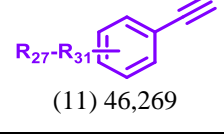
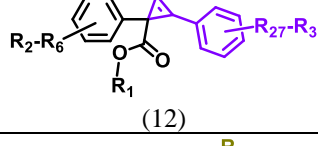
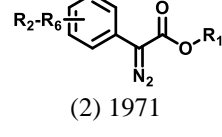
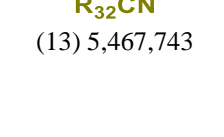
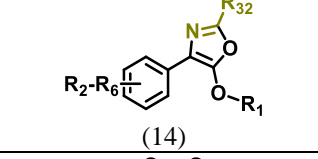
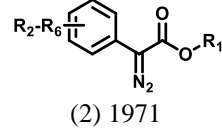
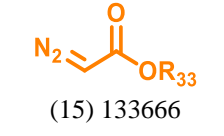
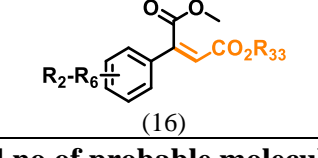


Figure S13. Starting material diazo ester reported earlier.

Table S3. Literature study of the probable molecules of product and their corresponding reported known molecule.

Starting material	Substrate	product	Probable molecule	Reported molecule
 (2) 1971	 (3) 1,159,909	 (4)	1971 × 1159909 = 2,286,180,639	5675
 (2) 1971	 (5) 31,582	 (6)	1971 × 31582 = 62,248,122	2590
 (2) 1971	 (7) 666,981	 (8)	1971 × 666981 = 1,314,619,551	2410
 (2) 1971	 (9) 64,464	 (10)	1971 × 64464 = 127,058,544	954
 (2) 1971	 (11) 46,269	 (12)	1971 × 46269 = 91,196,199	396
 (2) 1971	 (13) 5,467,743	 (14)	1971 × 5646743 = 11,129,730,45 3	385
 (2) 1971	 (15) 133666	 (16)	1971 × 133666 = 263,455,686	113
Total no of probable molecule = 1.5 × 10¹⁰				
			Total no of known molecule = 12523	

* Data extracted from Reaxys on 07 march 2024

$$\% \text{ known molecule} = \frac{12523}{15000000000} \times 100$$

$$= \sim 0.00000083\%$$

S3 Case study-1 carbene insertion (C-O bond formation) single objective multi-variant auto-optimization.

S3.1. Background collection

The collection of solvent background or an air background before automated experiment stand out as most important step in obtaining finest quality infrared data from a fully operational React IR 15. Each sample measurement will use the background to 'ratio out' all infrared absorbing materials in the optical path and the source solvent peaks. It will give infrared fingerprint end-result in form of absorbance versus wavenumber of only the reaction mixture components. For background collection set the scan to 128 scans at 4 cm⁻¹ resolutions.

Sample preparation: We prepared 0.5 M stock solutions of compound **2a**, reagent **3a**, and product **4a** in ethyl acetate, with each solution loaded into separate syringes. Our analysis utilized an In-line FTIR system. The experimental procedure began by introducing ethyl acetate solvent using a syringe pump for 10 min, during which data was recorded. Subsequently, the stock solution of compound **2a** was introduced into the In-line FTIR for another 10 minutes. This process was repeated for the reaction product **4a** and reagent **3a**, each pumped for 10 minutes. After collecting all relevant data, we identified the signature peak present in the product. Using this peak as a reference, we employed a Bayesian optimizer to fine-tune and optimize the reaction conditions for improved results.

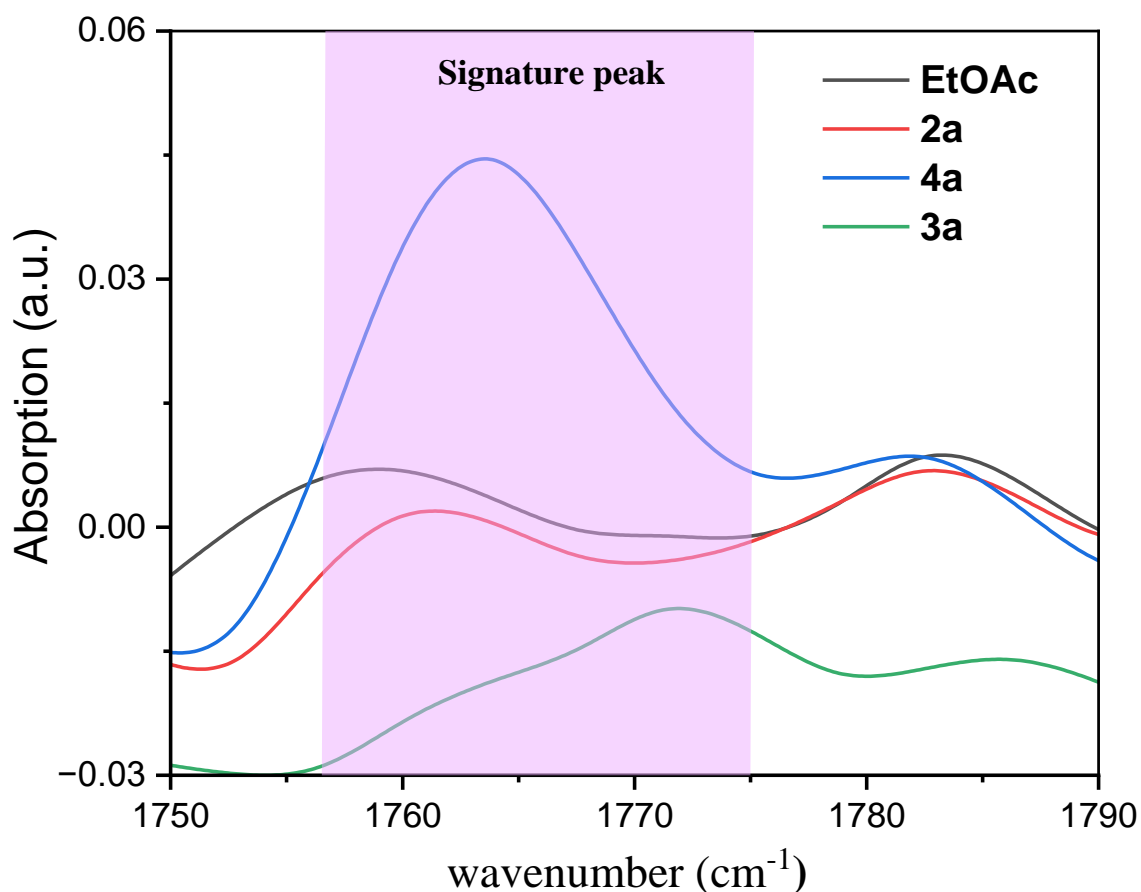


Figure S14. In-line IR background analysis of EtOAc, 0.5 M stock solution of **2a**, **3a**, and **4a** in EtOAc.

The distinctive keto peak associated with benzoic acid (**3a**) in the initial substrate undergoes a noticeable shift in the resulting product (**4a**), aligning within the range of 1756 to 1775 cm^{-1} . This specific peak, falling within this range, is designated as the signature peak for our analysis. During the optimization of the reaction through Bayesian methods, this signature peak serves as a pivotal reference point. We utilize it to calculate the area under the curve in the In-line FT-IR spectra, providing a quantitative measure of the area in the forthcoming reaction mixture.

Micro reactor system: As shown in Figure. S2d, we had utilized pump to deliver the solution at varied flow rates. Stock solution starting material and reagent were introduced through a T-mixer with the same flow rate to maintain a stoichiometric ratio. Subsequently, they passed through a PFA tubing (od 1/16, id 1mm, l = 1.3 m, vol = 1 mL) for the carbene insertion reaction. The cylindrical reactor was enveloped by cylindrical blue light LED for irradiation. To enhance light absorption, the entire reactor was covered with aluminum foil.

S3.2. General procedure for the auto-optimized synthesis of compound 3a.

In our approach we are employing a python-coded based Bayesian optimization strategy for further refinement. Initial steps involved the preparation of stock solutions for compound **2a** (0.1 M in EtOAc) and reagent **3a** (0.2 M in EtOAc), each filled into separate syringes and connected to a syringe pump. The solutions were then directed through a PFA tubular reactor (inner diameter = 1 mm, length = 1.3 m, volume = 1 mL) surrounded by a cylindrical-shaped blue LED light source. Once the solution setup was complete, input ranges for variable flow rates, voltages, and current were specified in the Python code designed for the reaction. The Bayesian optimizer systematically explored these varying reaction conditions, aiming to achieve maximum yield. The optimization process involved running 50-60 experiments in a 24 h time frame, and the results were tabulated in the optimization table (Table S4).

S3.2.1. Python code for optimizing condition of carbene insertion reaction (C-O bond formation).

```
ab1.py x
1 from os import listdir
2 from os.path import isfile, join
3 import serial
4
5 import numpy as np
6 import pandas as pd
7 import time
8 from scipy.integrate import trapz
9
10 #step1: be sure to the address of the files that the ftir data is exported is matching to line 11 (mypath)
11 mypath = "C:\\Users\\Admin\\Desktop\\ruchi\\Exp 2023-09-22 14-27"
12 onlyfiles = [f for f in listdir(mypath) if isfile(join(mypath, f))]
13
14
15 #step2: make sure that pump and the potentiostat is correctly addressed in the line 16 and 17
16 pump_1 = serial.Serial("COM4",9600) #Harvard Pump
17 port = serial.Serial("COM1",115200)
18 printer = serial.Serial("COM10", 115200, timeout=1)
19
20
21
22 #step 3: grab the lines from 22 to 90 and press f9
23 def area_under(data,start,end):
24     x = np.flip(data.iloc[start:end,0].to_numpy())
25     y = np.flip(data.iloc[start:end,1].to_numpy())
26     area = trapz(y,x)
27     return np.abs(area)
28
29 def file_namer(num):
30     str1 = str(num)
31     length = int(len((str1)))
32     empt = ''
33     for i in range(5-length):
34         empt = empt+'0'
35
36     return empt+str1
37
38
39 def ftir_extract(filename,init,end):
40
41     filename = filename
42
43     temp_df = pd.read_csv(filename)
44     # nump_df = temp_df.to_numpy()
45     area = area_under(temp_df, init, end)
46     # max_peak = np.max(nump_df[90:120,1])
47     print(area)
```

```

48
49     return area
50
51
52 def function(flowrate_1,v,i):
53
54     #set the pumps with the flowrate as the desired flowrate for the function
55
56     fr_1 = flowrate_1 #ml/min
57     pump_1.write(('irate '+str(fr_1)+' ml/min\r\n').encode())
58
59     time.sleep(0.1)
60
61     #set the com port for potentiostat and set the voltage and current
62     vol = v
63     curr = i
64     port.write(('VOLT '+str(vol)+'\r\n').encode()) #to change the voltge we need to use "VOLT 1" command
65     port.write(('CURR '+str(curr)+'\r\n').encode()) #to change the current we need to use "CURR 1" command
66
67
68     #pumps run
69     pump_1.write((b'irun\r\n'))
70     time.sleep(0.1)
71     time.sleep(3636)
72
73 def function2(flowrate_1,v,i):
74     time.sleep(180)
75
76     files = [f for f in listdir(mypath) if isfile(join(mypath, f))]
77
78     val = 0
79
80     #change wavelengths as per product here.
81     f_row=21 #first row of range for wavelength as per IR CSV
82     l_row=28 #last row of range for wavelength as per IR CSV
83
84     val += ftir_extract(files[-1],f_row,l_row)
85     val+=ftir_extract(files[-2],f_row,l_row)
86     val+=ftir_extract(files[-3],f_row,l_row)
87
88     avg_val = val/3
89
90     return avg_val
91
92
93 #step 4:grab the line 93 and f9
94 from skopt.optimizer import Optimizer

```



```

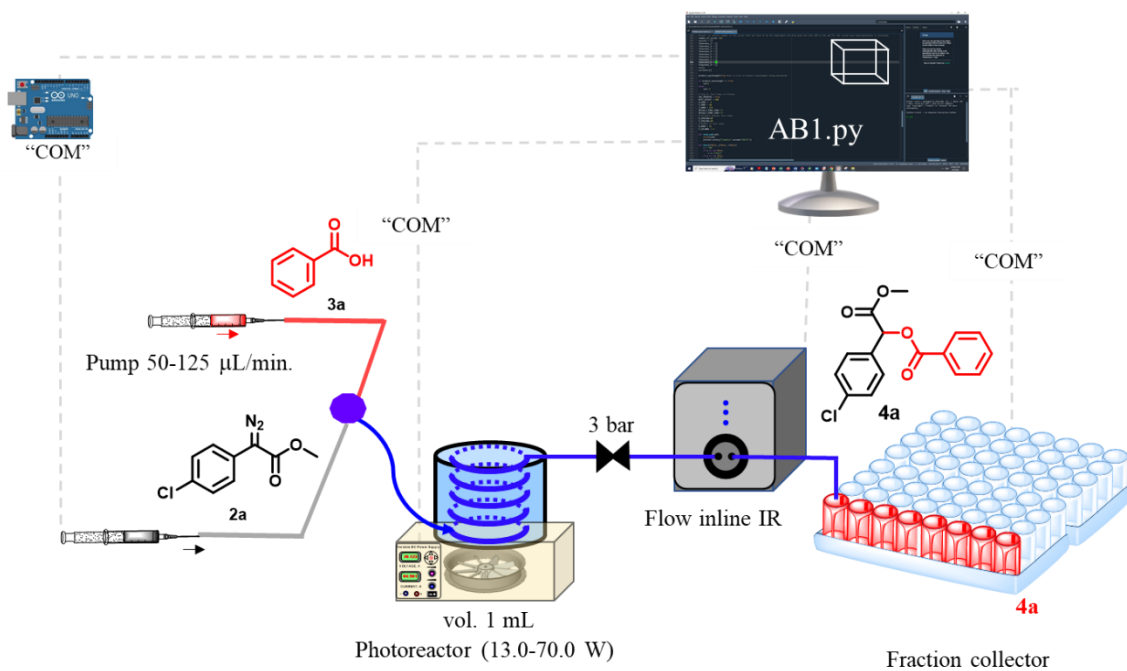
95
96
97 #step5:in line 96 we have to define the range that (flowrate,voltage,current) (from,to) and after (anytime) applying changes you need to grab the line 96 and f9
98 #flowrates are in ml/min, voltage in V, Current in Ampere
99 bounds = [(0.05,0.125),(10,14.5),(4.9,5.0)]
100
101
102 #step 6: grab the line 100 and f9
103 opter =Optimizer(bounds,base_estimator='gp',n_initial_points=3,acq_func="EI",random_state=np.random.randint(3326))
104
105
106 #step7: to selecte number of the cycles that you have to do the experiment and then grab the line 104 to 121 and f9: the closed loop experimentation is initiated
107 number_of_cycles = 22
108 results = []
109 flowrates_1 = []
110 vs = []
111 currents = []
112
113 product_wavelength=True #set to true if product wavelengths being monitored
114
115 if product_wavelength == True:
116     val=1
117 else:
118     val=-1
119
120
121 # Step 8: Test Tubes on Printer
122 USE_PRINTER = True
123 REST_HEIGHT = 200
124 X_HOME = -5
125 Y_HOME = 20
126 Z_HOME = 175
127 DEFAULT_PUMP_TIME="1"
128 # Distance between test tubes
129 X_SPACING=20
130 Y_SPACING=20
131 # Number of test tubes
132 X_ROWS = 11
133 Y_COLUMNS = 4
134
135 def send_cmd(cmd):
136     print(cmd)
137     printer.write(f"{cmd}\n".encode("ASCII"))
138
139 def move(x=None, y=None, z=None):

```

```

140 s = "G0"
141 if x is not None:
142     s += f"X{x}"
143 if y is not None:
144     s += f"Y{y}"
145 if z is not None:
146     s += f"Z{z}"
147
148 s+= "F5000"
149 send_cmd(s)
150
151 def printer_positions():
152     for j in range(Y_COLUMNS):
153         for i in range(X_ROWS):
154             if j%2==1:
155                 yield (X_HOME + (X_ROWS - 1 - i) * X_SPACING, Y_HOME + j * Y_SPACING, Z_HOME)
156             else:
157                 yield (X_HOME + i * X_SPACING, Y_HOME + j * Y_SPACING, Z_HOME)
158
159 # Run this.
160 tube_location = list(printer_positions())
161
162
163 for i in range(number_of_cycles):
164     move(*tube_location[2*i])
165     asked = opter.ask()
166     function(asked[0],asked[1],asked[2])
167     move(*tube_location[2*i+1])
168     told = function2(asked[0],asked[1],asked[2])
169
170     print(f"area under the curve in the round {i:.2f} = {told: .2f}")
171     opter.tell(asked,-told*val)
172     results.append(told)
173     flowrates_1.append(asked[0])
174     vs.append(asked[1])
175     currents.append(asked[2])
176
177     dict1 = {"flowrate_1":flowrates_1,"voltages":vs, "currents":currents,"area-results":results}
178     df2 = pd.DataFrame(dict1)
179     df2.to_csv("output round "+str(i)+".csv")
180
181
182
183 pump_1.write(b'stop\r\n')
184
185
186

```

Table S4. Auto-optimization table of carbene insertion reaction into O-H bonds of carboxylic acid.

No. of experiments	Flow rate mL/min	Voltage (V)	Light intensity (W)	area-results	Yield (%)	Space time yield (g mL ⁻¹ h ⁻¹)
1	0.066608	10.882021	24.45	0.088501	22.30199	0.013547726
2	0.029862	14.306068	65.808	0.200355	50.48887	0.013750297
3	0.065704	12.421805	40.39	0.171524	43.22355	0.025900525
4	0.099795	10.154336	16.488	0.227276	57.27289	0.052125911
5	0.075772	13.720454	57.344	0.256559	64.65212	0.044677424
6	0.041725	10	13.469	0.019331	4.871431	0.001853775
7	0.046836	11.089352	24.981	0.097945	24.68185	0.010542825
8	0.095982	10	13.83	0.026577	6.697226	0.005862467
9	0.098594	14.4	66.616	0.381375	96.10539	0.086416489
10	0.027592	13.123195	49.103	0.092042	23.19437	0.005836805
11	0.094381	12.699958	43.77	0.071	17.89184	0.015400604
12	0.1	13.164905	49.799	0.270311	68.11758	0.062123235
13	0.025	12.628105	42.957	0.135531	34.15342	0.007786979
14	0.040461	13.437396	53.558	0.200978	50.64587	0.01868879

15	0.076298	12.251856	38.395	0.12583	31.70879	0.022064265
16	0.090613	12.933006	46.788	0.166475	41.95121	0.034668386
17	0.028235	10	15.458	0.039305	9.904745	0.002550535
18	0.061374	14.4	67.262	0.330339	83.24446	0.046594855
19	0.102368	14.08762	62.363	0.08926	22.49326	0.077462861
20	0.104702	14.27705	65.017	0.054675	13.77794	0.08534457
21	0.119409	14.2009	63.843	0.111543	28.10851	0.074518828
22	0.066445	12.17059	37.334	0.114956	28.96858	0.017554331
23	0.125	11.8592	33.843	0.025144	6.336214	0.007223285
24	0.066	13.39582	52.746	0.099361	25.03868	0.015071283
25	0.056156	14.49334	68.059	0.238695	60.15044	0.032096196
26	0.125	12.5	41.184	0.196633	49.55094	0.056488073
27	0.066	14.16656	63.265	0.200791	50.59875	0.035006859
28	0.069702	14.5	68.049	0.26917	67.83005	0.04311836
29	0.094102	14.5	67.904	0.379021	95.51218	0.081969534
30	0.124355	13.5	53.919	0.097555	24.58357	0.027880665
31	0.125	14.27383	64.7	0.267493	67.40745	0.076844498
32	0.066	12.9345	46.537	0.106264	26.77822	0.016118345
33	0.082366	12	35.361	0.025258	6.364942	0.004781204
34	0.124833	14.13302	62.821	0.208424	52.52224	0.065533216
35	0.115972	14.5	68.02	0.351063	88.46685	0.093568259
36	0.036977	12	35.328	0.153633	38.71507	0.013055891
37	0.066536	14	60.984	0.276992	69.80117	0.042355933
38	0.115403	14.5	68.081	0.291776	73.5267	0.077385039
39	0.087432	14.00118	60.736	0.199524	50.27946	0.046119953
40	0.106282	13.49823	53.844	0.233256	58.77983	0.05697481
41	0.123375	14.15004	62.812	0.293364	73.92687	0.083181036
42	0.072075	14.492	67.649	0.272292	68.61679	0.048416336
43	0.067177	14.00765	60.664	0.209733	52.8521	0.032380065
44	0.110513	12.11331	36.53	0.10991	27.697	0.027915211
45	0.120569	14.34578	66.173	0.279088	70.32936	0.077333411
46	0.089661	14.12967	55.222	0.258843	65.22768	0.053337216

47	0.099916	13.9295	60.131	0.341064	85.94713	0.078317941
48	0.125	14.5	68.339	0.285321	71.90006	0.087711599
49	0.027135	13.5	54.216	0.127316	32.08326	0.007939683
50	0.096834	13.9295	60.131	0.337922	85.15536	0.075202916
51	0.039347	13.56446	54.622	0.263359	66.3657	0.023814975
52	0.078418	14.5	67.875	0.315884	79.60184	0.056929023
53	0.099858	14.5	69.733	0.396378	99.88607	0.090966741
54	0.11909	14.1	62.209	0.247817	62.44916	0.067826082
55	0.082793	14.2	63.659	0.298597	75.24557	0.056815836
56	0.088865	14.5	68.005	0.339088	85.44918	0.07231565
57	0.068636	14.41542	66.722	0.282448	71.17607	0.044553396
58	0.075903	14	60.802	0.192995	48.63418	0.033666296
59	0.084141	14.2	63.801	0.211859	53.38785	0.040968016
60	0.10001	11.43321	29.232	0.099703	25.12486	0.022916168
61	0.099987	13.437396	53.422	0.25504	64.26933	0.058606013
62	0.095871	13.537396	55.717	0.260815	65.72462	0.057465893

Reaction condition: compound **2a** (0.1 M in EtOAc); reagent **3a** (0.2 M in EtOAc); 1 mL reactor volume.

Graph: We have plotted 3D graph of optimization table S4, we have taken flow rate on X-axis, blue light intensity (watt) on Y-axis and product yield on Z axis.

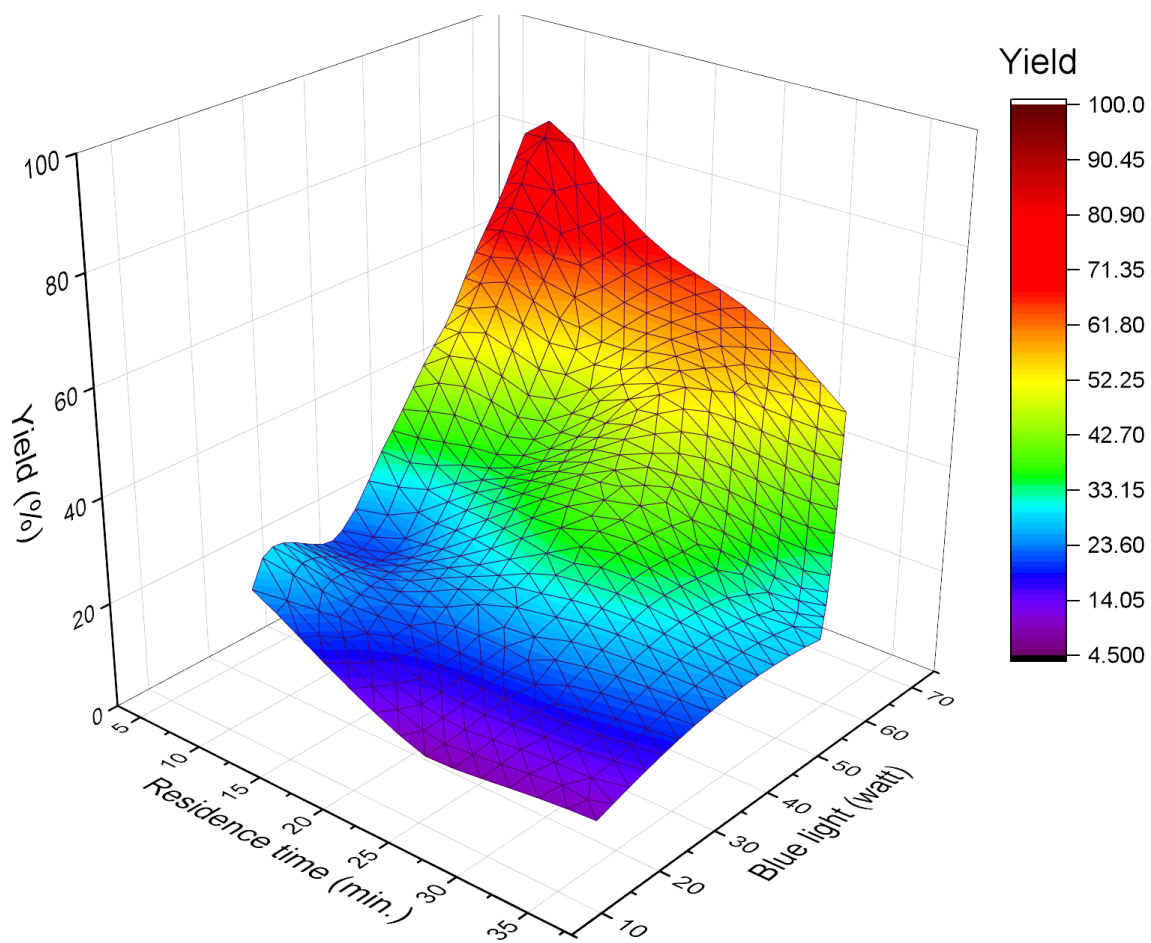


Figure S15. AI based system to auto-optimize and navigate this complexity and identify the optimal conditions for the photo activated C-O bond formation reaction. Compound **2a** (0.1 M in EtOAc); reagent **3a** (0.2 M in EtOAc); 1 mL reactor volume.

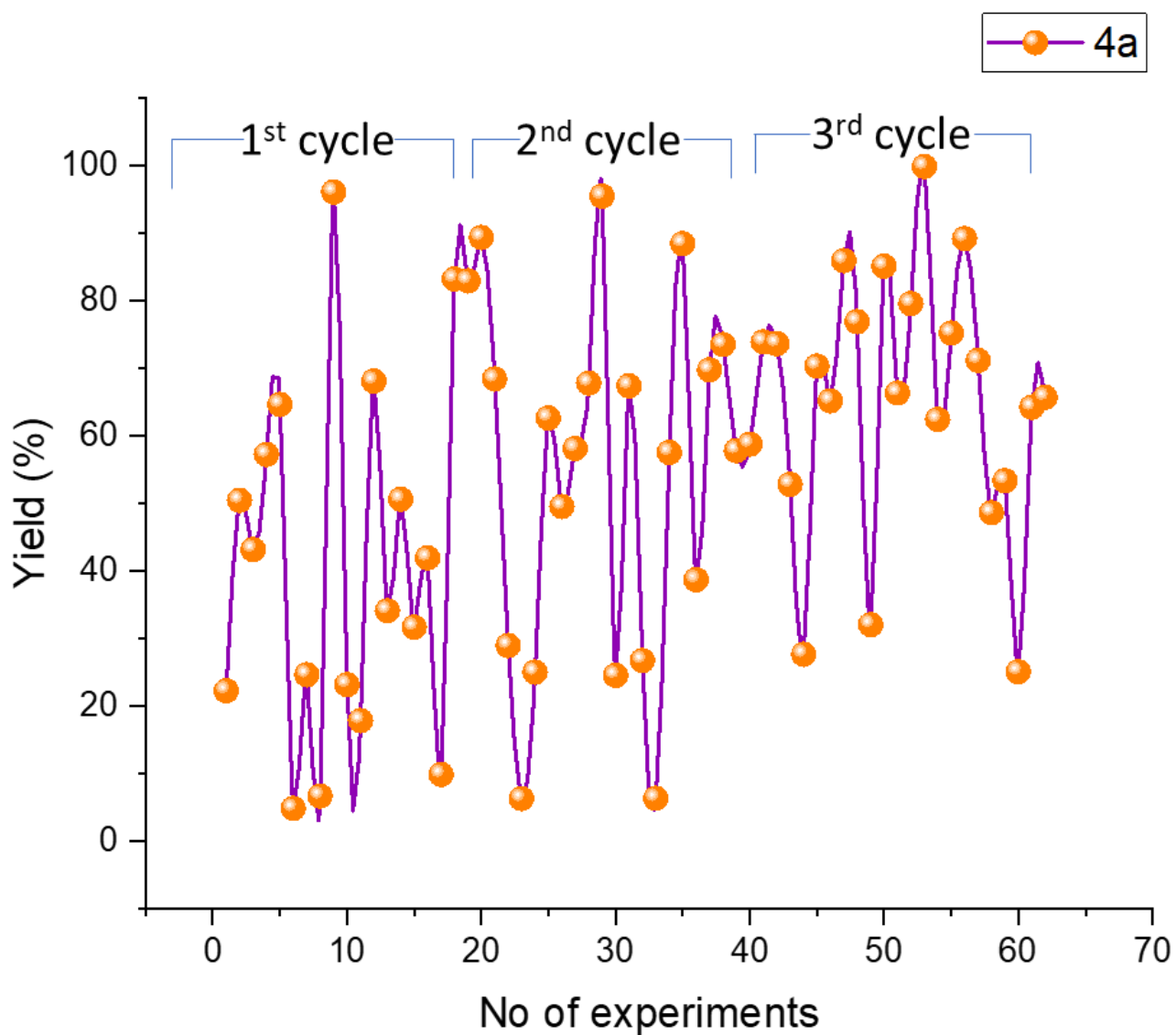


Figure S16. 2D graph between no of experiments performed versus yield (%) for the AI based auto-optimization of photo activated O-H reaction. Compound **2a** (0.1 M in EtOAc); reagent **3a** (0.2 M in EtOAc); 1 mL reactor volume.

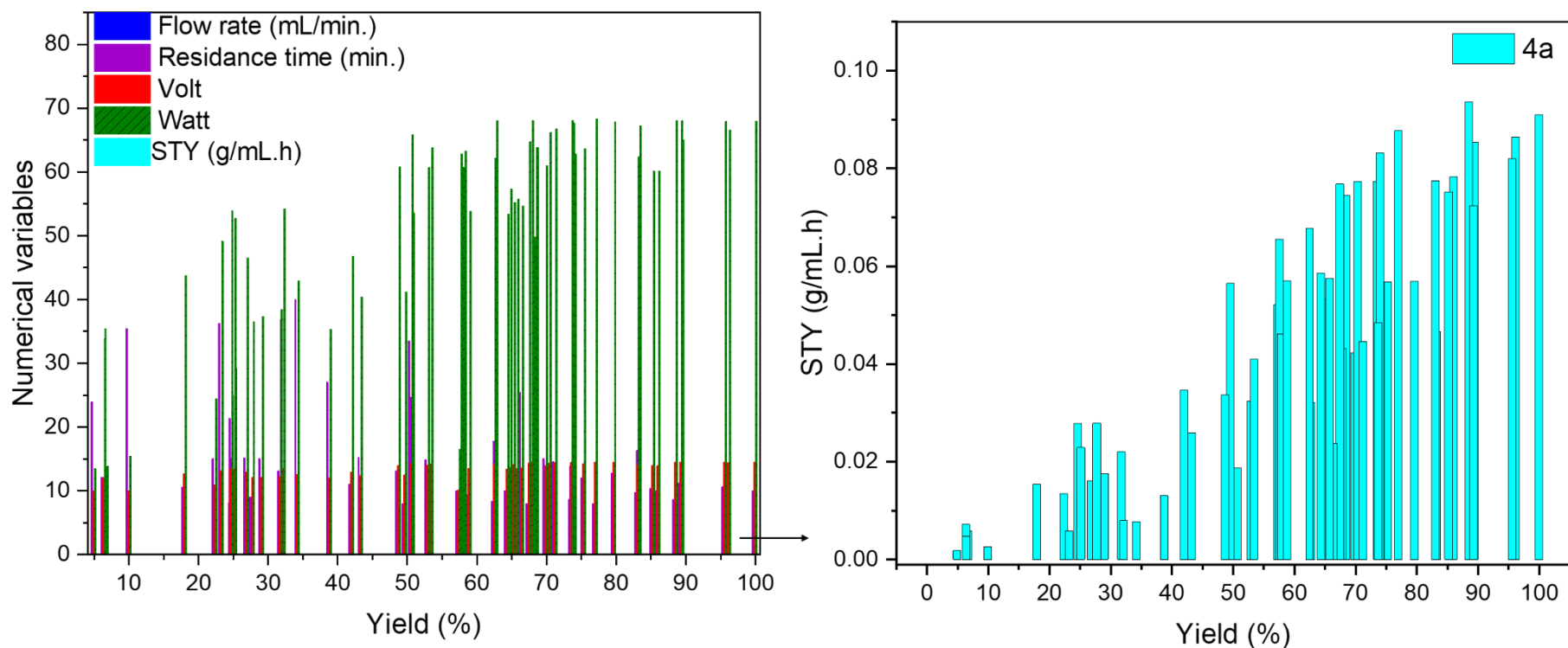


Figure S17. AI based system navigated multi-numerical variable complexity and identify the optimal conditions for the photo activated O-H reaction. Compound **2a** (0.1 M in EtOAc); reagent **3a** (0.2 M in EtOAc); 1 mL reactor volume.

S3.3. Procedure for the synthesis, of compound **4a** for longer time.

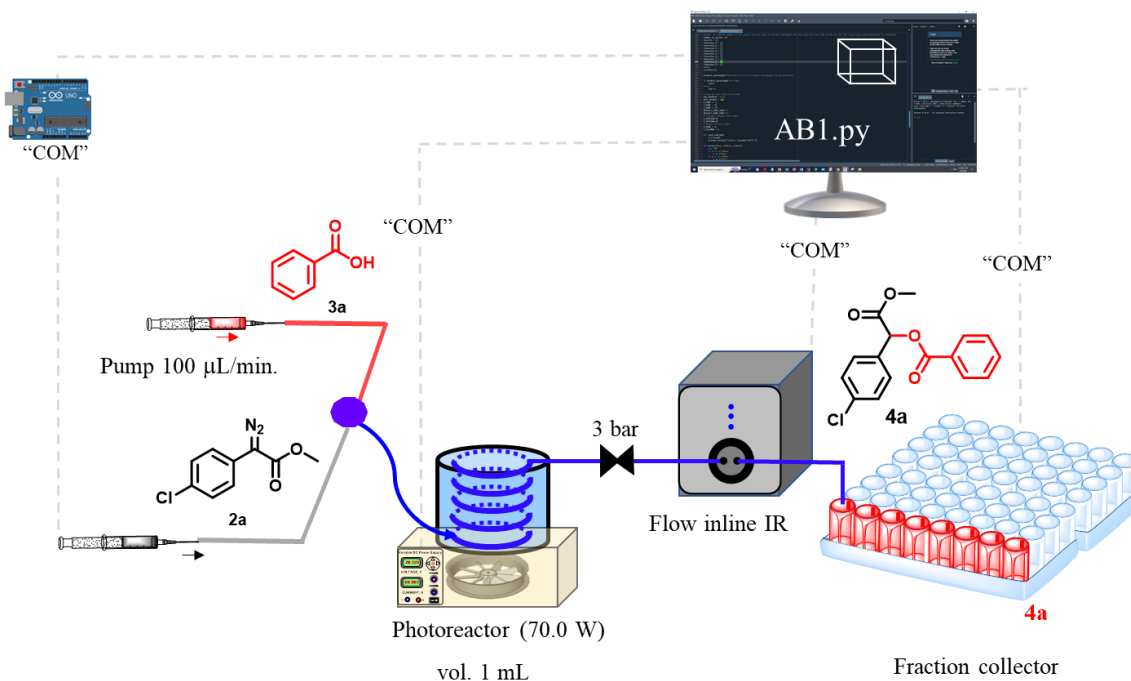


Figure S18. Schematic presentation of continuous flow for the synthesis, of compound **4a**.

To prepare the stock solution of compound **2a** (2.20 g, 0.01 mol) was dissolved in ethyl acetate (100 mL) charged in one syringe, another syringe charged with the reagent **3a** (2.56 g, 0.02 mol) dissolved in ethyl acetate (100 mL). These two syringes connected via pump and out-put is further connected with the T₁-mixer. Both syringes were running with the 50 µL/min. flow rate each to maintain the stoichiometry and then passed through PFA tubular reactor (id = 1000 µm, l = 1.3 m, vol. = 1 mL) under blue light (70 W) exposure. The room temperature of the reactor was controlled by fan attached to the bottom of the photochemical reactor. The out-put of the tubular reactor was connected with spring based back pressure regulator (~3 bar) to maintain the evaporation. Then out coming first one hour of the product mixture **4a** was discarded and next 5 h of the product mixture [30 mL; **2a** (0.32 g)] was collected in HPLC bottle. The EtOAc reaction mixture washed it with 10% aq. NaOH and separated through the separating funnel. The organic EtOAc phase was concentrated under reduced pressure to give the product **4a** (0.45 g in 5 h, 99.88%) as a colorless oil. ¹H NMR (400 MHz, CDCl₃) δ 8.12

– 8.10 (m, 2H), 7.60 – 7.51 (m, 3H), 7.47 – 7.38 (m, 4H), 6.15 (s, 1H), 3.74 (s, 3H). **¹³C NMR (126 MHz, CDCl₃)** δ 168.97, 165.71, 135.36, 133.65, 132.55, 129.99, 129.14, 129.03, 128.55, 74.14, 52.82. **IR (ν_{max}):** 3023.51, 1754.46, 1724.80, 1255.08, 815.32 cm⁻¹. **HRMS (ESI):** *m/z* calcd for C₁₆H₁₄ClO₄ [M+H]⁺ 305.7335, found 305.7332.

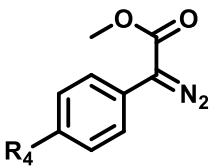
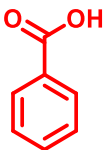
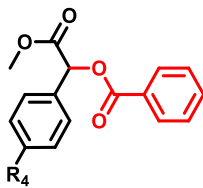
Space time yield in previous reported batch process²

Productivity = 33 mg in 12 h

$$\text{Productivity (per h)} := \frac{0.033}{12} = 0.00275 \text{ g/h}$$

$$\begin{aligned} \text{Space time yield} &:= \frac{\text{productivity}}{\text{volume of reactor}} \\ &= \frac{0.0028}{4} = 0.0007 \text{ g mL}^{-1} \text{ h}^{-1} \end{aligned}$$

Table S5: Comparative table for the carbene insertion into O-H bonds of carboxylic acid.

Entry	Compound 2a	Reagent 3a	Product 4a	Comparative result
1				99.8%, 10 min. (our study) Blue light, overnight, 91% 2 Guanidine (5 mol%), Rh(OAc) ₄ (2.5 mol%), 3 h, 89% 3

S3.5. General procedure for scope of substrate for carbene insertion into the O-H bonds of carboxylic acid for synthesis of compound 4b–4p.

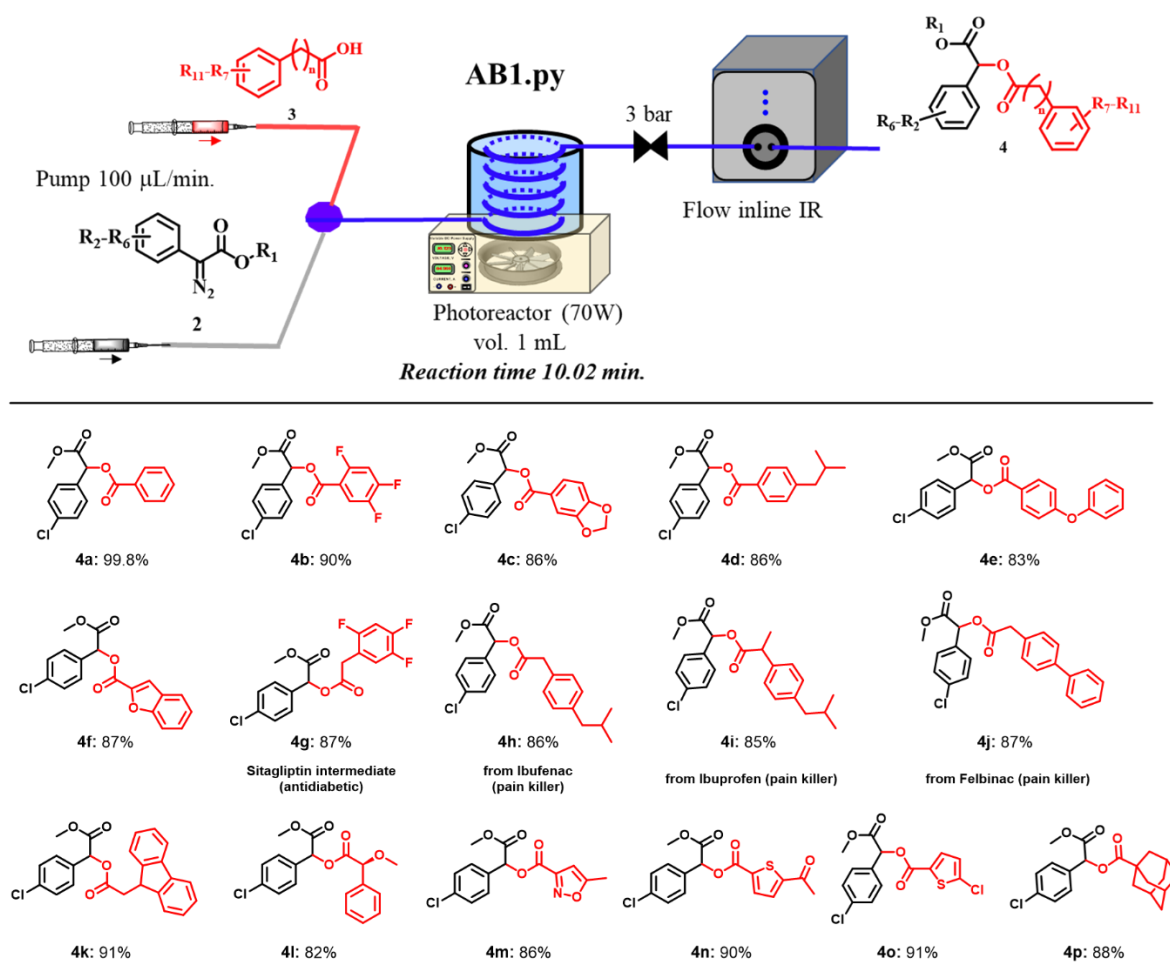
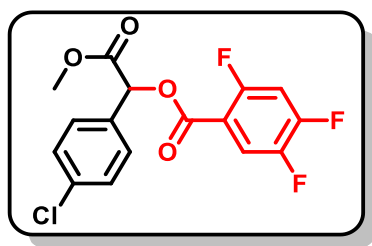


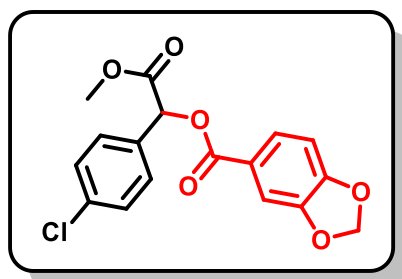
Figure S19. Scope of substrate for the photo activated O-H reaction. Compound **2** (0.1 M in EtOAc); reagent **3** (0.2 M in EtOAc); 1 mL reactor volume.

Example 2. 1-(4-chlorophenyl)-2-methoxy-2-oxoethyl 2,4,5-trifluorobenzoate (**4b**):



The title compound was synthesised following the general procedure described in section 3.3, and involve corresponding reactant exchange with compound **2a** and compound **3b** (2,4,5-trifluorobenzoic acid). The crude mixture was concentrated under vacuum and the product was purified by flash chromatography; $R_f = 0.5$ (10% ethylacetate/hexane); to give compound **4b** (0.48 g in 5 h, 90%) as a colorless oil. $^1\text{H NMR}$ (400 MHz, CDCl_3) δ 7.90 – 7.83 (m, 1H), 7.51 – 7.49 (m, 2H), 7.42 – 7.39 (m, 2H), 7.07 – 7.01 (m, 1H), 6.13 (s, 1H), 3.76 (s, 3H). $^{13}\text{C NMR}$ (101 MHz, CDCl_3) δ 168.57, 161.79, 159.44, 157.34, 154.87, 152.81, 147.69, 145.64, 135.69, 131.89, 129.31, 129.04, 120.50, 120.33, 107.62, 107.46, 107.40, 107.23, 74.78, 53.10. $^{19}\text{F NMR}$ (376 MHz, CDCl_3) δ -107.92, -107.95, -107.96, -107.99, -123.52, -123.55, -123.59, -123.61, -140.79, -140.85, -140.90. **IR** (ν_{max}): 3021.92, 2925.33, 1746.25, 1623.42, 1518.64, 1216.49, 824.87 cm^{-1} . **HRMS (ESI):** m/z calcd for $\text{C}_{16}\text{H}_{11}\text{ClF}_3\text{O}_4$ $[\text{M}+\text{H}]^+$ 359.0292, found 359.0298.

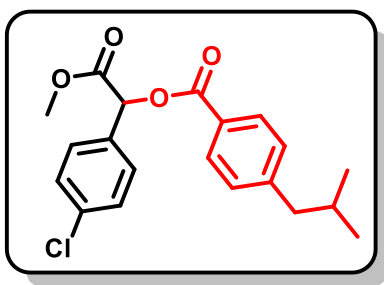
Example 3. 1-(4-chlorophenyl)-2-methoxy-2-oxoethyl benzo[d][1,3]dioxole-5-carboxylate (**4c**):



The title compound was synthesised following the general procedure described in section 3.3, and involve corresponding reactant exchange with compound **2a** and compound **3c**

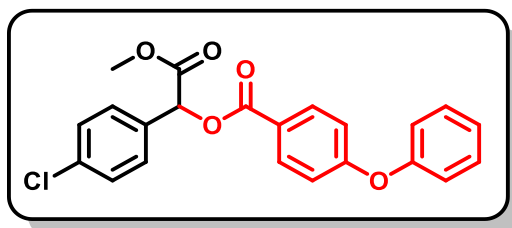
(benzo[*d*][1,3]dioxole-5-carboxylic acid). The crude mixture was concentrated under vacuum and the product was purified by flash chromatography; $R_f = 0.5$ (10% ethylacetate/hexane); to give compound **4c** (0.45 g in 5 h, 86%) as a colorless oil $^1\text{H NMR}$ (400 MHz, CDCl_3) δ 7.74 – 7.72 (m, 1H), 7.52 – 7.49 (m, 3H), 7.41 – 7.39 (m, 2H), 6.85 (d, $J = 8.2$ Hz, 1H), 6.10 (s, 1H), 6.05 (s, 2H), 3.75 (s, 3H). $^{13}\text{C NMR}$ (101 MHz, CDCl_3) δ 169.13, 165.13, 152.34, 147.95, 135.44, 132.66, 131.09, 129.21, 129.08, 126.16, 123.01, 109.85, 108.24, 102.06, 74.15, 52.90. IR (ν_{max}): 2955.71, 2920.24, 1754.04, 1718.03, 1256.96, 1036.39, 831.60 cm^{-1} . HRMS (ESI): m/z calcd for $\text{C}_{17}\text{H}_{14}\text{ClO}_6$ $[\text{M}+\text{H}]^+$ 349.0473, found 349.0475.

Example 4. 1-(4-chlorophenyl)-2-methoxy-2-oxoethyl 4-isobutylbenzoate (4d):



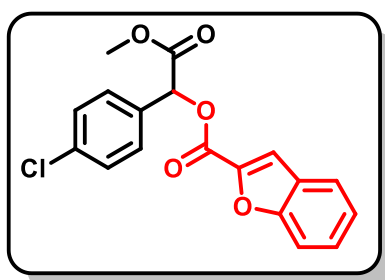
The title compound was synthesised following the general procedure described in section 3.3, and involve corresponding reactant exchange with compound **2a** and compound **3d** (4-isobutylbenzoic acid). The crude mixture was concentrated under vacuum and the product was purified by flash chromatography; $R_f = 0.4$ (10% ethylacetate/hexane); to give compound **4d** (0.46 g in 5 h, 86%) as a colorless oil. $^1\text{H NMR}$ (400 MHz, CDCl_3) δ 8.04 – 8.01 (m, 2H), 7.53 – 7.51 (m, 2H), 7.39 – 7.37 (m, 2H), 7.22 (d, $J = 8.3$ Hz, 2H), 6.14 (s, 1H), 3.73 (s, 3H), 2.52 (d, $J = 7.2$ Hz, 2H), 1.91 – 1.83 (m, 1H), 0.89 (d, $J = 6.6$ Hz, 6H). $^{13}\text{C NMR}$ (101 MHz, CDCl_3) δ 169.02, 165.73, 148.11, 135.25, 132.68, 129.90, 129.27, 129.07, 129.00, 126.57, 73.97, 52.71, 45.42, 30.12, 22.29. IR (ν_{max}): 3022.14, 2960.22, 1753.90, 1721.69, 1492.80, 1215.16, 1016.94 cm^{-1} . HRMS (ESI): m/z calcd for $\text{C}_{20}\text{H}_{22}\text{ClO}_4$ $[\text{M}+\text{H}]^+$ 361.1201, found 361.1209.

Example 5. 1-(4-chlorophenyl)-2-methoxy-2-oxoethyl 4-phenoxybenzoate (4e):



The title compound was synthesised following the general procedure described in section 3.3, and involve corresponding reactant exchange with compound **2a** and compound **3e** (4-phenoxybenzoic acid). The crude mixture was concentrated under vacuum and the product was purified by flash chromatography; $R_f = 0.6$ (10% ethylacetate/hexane); to give compound **4e** (0.49 g in 5 h, 83%) as a colorless oil. $^1\text{H NMR}$ (400 MHz, CDCl_3) δ 8.11 – 8.08 (m, 2H), 7.54 – 7.52 (m, 2H), 7.42 – 7.38 (m, 4H), 7.22 – 7.18 (m, 1H), 7.09 – 7.00 (m, 4H), 6.15 (s, 1H), 3.76 (s, 3H). $^{13}\text{C NMR}$ (101 MHz, CDCl_3) δ 169.06, 165.19, 162.51, 155.48, 135.36, 132.65, 132.22, 130.13, 129.15, 129.03, 124.72, 123.25, 120.23, 117.40, 74.05, 52.83. IR (ν_{max}): 3022.03, 1754, 1720.35, 1491.57, 1215.73, 865.28 cm^{-1} . HRMS (ESI): m/z calcd for $\text{C}_{22}\text{H}_{18}\text{ClO}_5$ $[\text{M}+\text{H}]^+$ 397.0837, found 397.0847.

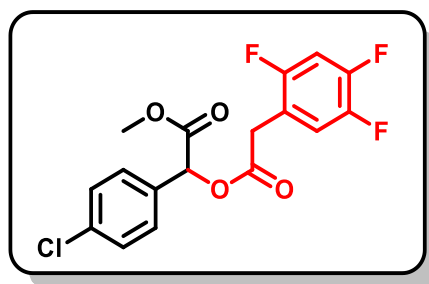
Example 6. 1-(4-chlorophenyl)-2-methoxy-2-oxoethyl benzofuran-2-carboxylate (4f):



The title compound was synthesised following the general procedure described in section 3.3, and involve corresponding reactant exchange with compound **2a** and compound **3f** (benzofuran-2-carboxylic acid). The crude mixture was concentrated under vacuum and the product was purified by flash chromatography; $R_f = 0.5$ (10% ethylacetate/hexane); to give compound **4f** (0.45 g in 5 h, 87%) as a colorless oil. $^1\text{H NMR}$ (400 MHz, CDCl_3) δ 7.67 –

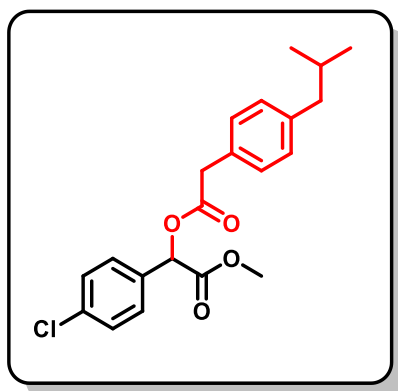
7.65 (m, 2H), 7.57 – 7.51 (m, 3H), 7.45 – 7.37 (m, 3H), 7.30 – 7.25 (m, 1H), 6.19 (s, 1H), 3.75 (s, 3H). **¹³C NMR (101 MHz, CDCl₃)** δ 168.53, 158.52, 156.04, 144.36, 135.54, 132.01, 129.17, 129.11, 128.12, 126.81, 123.99, 123.02, 115.41, 74.21, 52.94. **IR (ν_{max}):** 2956.04, 2854.56, 1730.43, 1566.86, 1172.31, 888.28 cm⁻¹. **HRMS (ESI):** *m/z* calcd for C₁₈H₁₄ClO₅ [M+H]⁺ 345.0524, found 345.0531.

Example 7. Methyl 2-(4-chlorophenyl)-2-(2-(2,4,5-trifluorophenyl)acetoxy)acetate (4g):



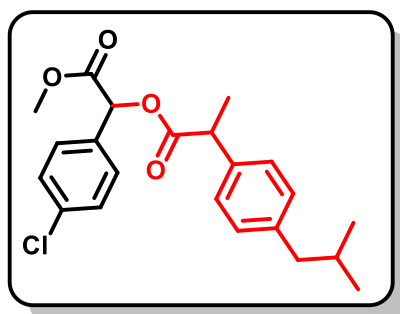
The title compound was synthesised following the general procedure described in section 3.3, and involve corresponding reactant exchange with compound **2a** and compound **3g** (2-(2,4,5-trifluorophenyl)acetic acid). The crude mixture was concentrated under vacuum and the product was purified by flash chromatography; **R_f** = 0.4 (10% ethylacetate/hexane); to give compound **4g** (0.48 g in 5 h, 87%) as a colorless oil. **¹H NMR (400 MHz, CDCl₃)** δ 7.39 – 7.34 (m, 4H), 7.22 – 7.15 (m, 1H), 6.97 – 6.90 (m, 1H), 5.94 (s, 1H), 3.77 (d, *J* = 3.8 Hz, 2H), 3.71 (s, 3H). **¹³C NMR (101 MHz, CDCl₃)** δ 169.27, 168.63, 157.40, 157.32, 154.95, 154.88, 150.95, 150.82, 150.69, 148.46, 148.32, 148.20, 148.00, 147.87, 145.60, 145.57, 145.47, 145.44, 135.54, 131.94, 129.17, 128.99, 119.33, 119.28, 119.13, 119.09, 117.10, 117.05, 117.00, 116.92, 116.87, 116.82, 105.83, 105.62, 105.55, 105.34, 74.36, 52.89, 33.32. **¹⁹F NMR (471 MHz, CDCl₃)** δ -118.27, -118.30, -134.23, -134.27, -142.50, -142.54, -142.57. **IR (ν_{max}):** 3024.90, 2958.73, 1749.06, 1633.89, 1215.42, 979.95, 850.85. **HRMS (ESI):** *m/z* calcd for C₁₇H₁₃ClF₃O₄ [M+H]⁺ 373.0449, found 373.0456.

Example 8. Methyl 2-(4-chlorophenyl)-2-(2-(4-isobutylphenyl)acetoxy)acetate (4h):



The title compound was synthesised following the general procedure described in section 3.3, and involve corresponding reactant exchange with compound **2a** and compound **3h** (2-(4-isobutylphenyl)acetic acid). The crude mixture was concentrated under vacuum and the product was purified by flash chromatography; $R_f = 0.4$ (10% ethylacetate/hexane); to give compound **4h** (0.49 g in 5 h, 88%) as a colorless oil. $^1\text{H NMR}$ (400 MHz, CDCl_3) δ 7.37 – 7.32 (m, 4H), 7.20 (d, $J = 8.1$ Hz, 2H), 7.10 (d, $J = 8.1$ Hz, 2H), 5.91 (s, 1H), 3.74 (d, $J = 3.8$ Hz, 2H), 3.69 (s, 3H), 2.45 (d, $J = 7.2$ Hz, 2H), 1.89 – 1.79 (m, 1H), 0.89 (d, $J = 6.6$ Hz, 6H). $^{13}\text{C NMR}$ (101 MHz, CDCl_3) δ 171.00, 168.85, 140.75, 135.27, 132.28, 130.43, 73.92, 52.76, 45.09, 40.47, 30.23, 22.39. IR (ν_{max}): 3020.64, 2957.88, 1748.49, 1492.75, 1215.51, 1016.03 cm^{-1} . HRMS (ESI): m/z calcd for $\text{C}_{21}\text{H}_{24}\text{ClO}_4$ $[\text{M}+\text{H}]^+$ 375.8685, found 375.8690.

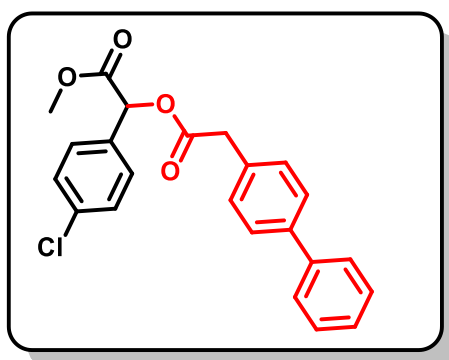
Example 9. 1-(4-chlorophenyl)-2-methoxy-2-oxoethyl 2-(4-isobutylphenyl)propanoate (4i):



The title compound was synthesised following the general procedure described in section 3.3, and involve corresponding reactant exchange with compound **2a** and compound **3i** (2-(4-isobutylphenyl)propanoic acid). The crude mixture was concentrated under vacuum and the

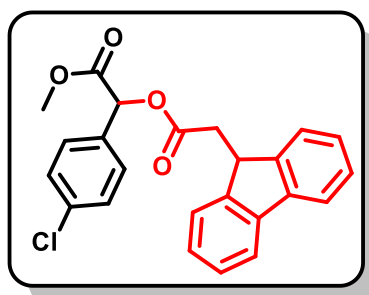
product was purified by flash chromatography; $R_f = 0.5$ (10% ethylacetate/hexane); to give compound **4i** (0.49 g in 5 h, 85%) as a colorless oil. $^1\text{H NMR}$ (400 MHz, CDCl_3) δ 7.31 (d, $J = 1.5$ Hz, 2H), 7.27 – 7.19 (m, 4H), 7.13 – 7.06 (m, 2H), 5.89 (d, $J = 3.3$ Hz, 1H), 3.90 – 3.83 (m, 1H), 3.65 (d, $J = 31.5$ Hz, 3H), 2.44 (dd, $J = 10.1, 7.2$ Hz, 2H), 1.89 – 1.79 (m, 1H), 1.56 (dd, $J = 7.2, 2.0$ Hz, 3H), 0.89 (t, $J = 6.8$ Hz, 6H). $^{13}\text{C NMR}$ (101 MHz, CDCl_3) δ 173.89, 173.82, 168.94, 168.74, 140.78, 140.71, 137.02, 136.92, 135.18, 135.08, 132.44, 132.39, 129.40, 129.33, 128.98, 128.91, 128.82, 128.71, 127.46, 127.37, 73.90, 73.71, 52.72, 52.59, 45.10, 45.06, 44.96, 44.82, 30.24, 22.41, 18.49, 18.35. IR (ν_{max}): 3020.99, 2956.85, 2872.11, 1744.81, 1599.53, 1215.33, 1016.16 cm^{-1} . HRMS (ESI): m/z calcd for $\text{C}_{22}\text{H}_{26}\text{ClO}_4$ $[\text{M}+\text{H}]^+$ 389.1514, found 389.1520.

Example 10. Methyl 2-(2-([1,1'-biphenyl]-4-yl)acetoxy)-2-(4-chlorophenyl)acetate (4j):



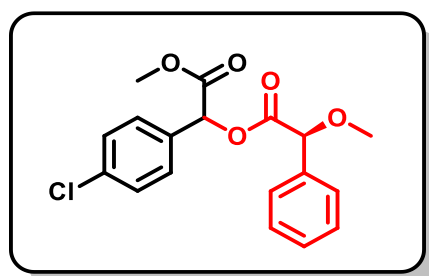
The title compound was synthesised following the general procedure described in section 3.3, and involve corresponding reactant exchange with compound **2a** and compound **3j** (2-([1,1'-biphenyl]-4-yl)acetic acid). The crude mixture was concentrated under vacuum and the product was purified by flash chromatography; $R_f = 0.5$ (10% ethylacetate/hexane); to give compound **4j** (0.52 g in 5 h, 87%) as a colorless oil. $^1\text{H NMR}$ (400 MHz, CDCl_3) δ 7.58 – 7.55 (m, 4H), 7.45 – 7.32 (m, 9H), 5.94 (s, 1H), 3.87 – 3.77 (m, 2H), 3.70 (s, 3H). $^{13}\text{C NMR}$ (101 MHz, CDCl_3) δ 170.82, 168.84, 140.74, 140.29, 135.36, 132.22, 132.18, 129.83, 129.07, 128.94, 128.81, 127.38, 127.09, 74.04, 52.80, 40.45. IR (ν_{max}): 3024.43, 1744.86, 1489.15, 1215.88, 1013.88 cm^{-1} . HRMS (ESI): m/z calcd for $\text{C}_{23}\text{H}_{20}\text{ClO}_4$ $[\text{M}+\text{H}]^+$ 395.1045, found 395.1055.

Example 11. Methyl 2-(2-(9H-fluoren-9-yl)acetoxy)-2-(4-chlorophenyl)acetate (4k):



The title compound was synthesised following the general procedure described in section 3.3, and involve corresponding reactant exchange with compound **2a** and compound **3k** (2-(9H-fluoren-9-yl)acetic acid). The crude mixture was concentrated under vacuum and the product was purified by flash chromatography; $R_f = 0.45$ (10% ethylacetate/hexane); to give compound **4k** (0.55 g in 5 h, 91%) as a colorless oil. $^1\text{H NMR}$ (400 MHz, CDCl_3) δ 7.71 (d, $J = 7.5$ Hz, 2H), 7.50 (dd, $J = 10.7, 7.6$ Hz, 2H), 7.43 – 7.29 (m, 8H), 6.04 (s, 1H), 4.49 (t, $J = 6.9$ Hz, 1H), 3.78 (s, 3H), 3.01 (dd, $J = 16.5, 6.8$ Hz, 1H), 2.90 (dd, $J = 16.5, 7.1$ Hz, 1H). $^{13}\text{C NMR}$ (101 MHz, CDCl_3) δ 171.66, 168.85, 145.93, 140.86, 140.81, 135.32, 132.15, 129.05, 129.01, 127.59, 127.30, 127.25, 124.46, 120.00, 74.07, 52.83, 43.37, 38.15. IR (ν_{max}): 3019.04, 2922.32, 1741.47, 1489.45, 1216.03, 1144.72 cm^{-1} . HRMS (ESI): m/z calcd for $\text{C}_{24}\text{H}_{20}\text{ClO}_4$ $[\text{M}+\text{H}]^+$ 407.1045, found 407.1053.

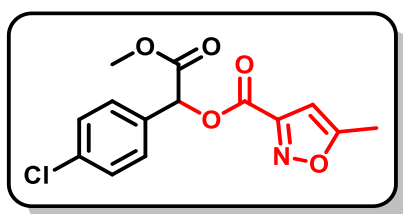
Example 12. 1-(4-Chlorophenyl)-2-methoxy-2-oxoethyl (2S)-2-methoxy-2-phenylacetate (4l):



The title compound was synthesised following the general procedure described in section 3.3, and involve corresponding reactant exchange with compound **2a** and compound **3l** ((R)-3-

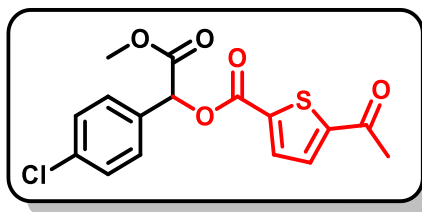
methoxy-3-phenylpropanoic acid). The crude mixture was concentrated under vacuum and the product was purified by flash chromatography; $R_f = 0.6$ (10% ethylacetate/hexane); to give compound **4l** (0.44 g in 5 h, 82%) as a colorless oil. $^1\text{H NMR}$ (400 MHz, CDCl_3) δ 7.50 – 7.48 (m, 2H), 7.41 – 7.33 (m, 7H), 5.93 (s, 1H), 4.90 (s, 1H), 3.59 (s, 3H), 3.49 (s, 3H). $^{13}\text{C NMR}$ (101 MHz, CDCl_3) δ 170.10, 168.34, 135.70, 135.47, 132.02, 129.14, 129.00, 128.96, 128.69, 127.46, 82.49, 74.15, 57.71, 52.76. IR (ν_{max}): 2929.68, 1750.89, 1598.38, 1217.54, 1164.43, 1016.87, 828.38 cm^{-1} . HRMS (ESI): m/z calcd for $\text{C}_{22}\text{H}_{18}\text{ClO}_5$ $[\text{M}+\text{H}]^+$ 363.0994, found 363.0999.

Example 13. 1-(4-chlorophenyl)-2-methoxy-2-oxoethyl 5-methylisoxazole-3-carboxylate (4m):



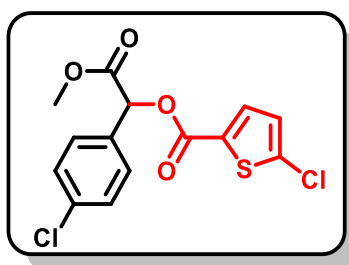
The title compound was synthesised following the general procedure described in section 3.3, and involve corresponding reactant exchange with compound **2a** and compound **3m** (5-methylisoxazole-3-carboxylic acid). The crude mixture was concentrated under vacuum and the product was purified by flash chromatography; $R_f = 0.6$ (10% ethylacetate/hexane); to give compound **4m** (0.38 g in 5 h, 82%) as a colorless oil. $^1\text{H NMR}$ (400 MHz, CDCl_3) δ 7.87 – 7.83 (m, 2H), 7.77 – 7.68 (m, 2H), 6.83 (d, $J = 0.8$ Hz, 1H), 6.50 (s, 1H), 4.12 (s, 3H), 2.86 (d, $J = 0.7$ Hz, 3H). $^{13}\text{C NMR}$ (101 MHz, CDCl_3) δ 171.80, 168.21, 159.34, 155.55, 135.69, 131.62, 129.25, 129.15, 128.83, 128.04, 111.21, 102.55, 74.70, 72.26, 53.07, 12.40. IR (ν_{max}): 3021.10, 2957.90, 1746.23, 1598.19, 1207.20, 1006.59, 808.66 cm^{-1} . HRMS (ESI): m/z calcd for $\text{C}_{14}\text{H}_{13}\text{ClNO}_5$ $[\text{M}+\text{H}]^+$ 310.0477, found 310.0478.

Example 14. 1-(4-chlorophenyl)-2-methoxy-2-oxoethyl 5-acetylthiophene-2-carboxylate (4n):



The title compound was synthesised following the general procedure described in section 3.3, and involve corresponding reactant exchange with compound **2a** and compound **3n** (5-acetylthiophene-2-carboxylic acid). The crude mixture was concentrated under vacuum and the product was purified by flash chromatography; $R_f = 0.6$ (10% ethylacetate/hexane); to give compound **4n** (0.47 g in 5 h, 90%) as a colorless oil. $^1\text{H NMR}$ (400 MHz, CDCl_3) δ 7.84 (d, $J = 4.0$ Hz, 1H), 7.64 (d, $J = 4.0$ Hz, 1H), 7.49 – 7.46 (m, 2H), 7.40 – 7.37 (m, 2H), 6.09 (s, 1H), 3.74 (s, 3H), 2.58 (s, 3H). $^{13}\text{C NMR}$ (101 MHz, CDCl_3) δ 190.79, 168.43, 160.69, 149.64, 138.16, 135.61, 134.49, 131.85, 131.76, 129.22, 128.99, 74.54, 53.00, 27.07. IR (ν_{max}): 3021.89, 1724.77, 1673.48, 1524.47, 1214.90, 1017.28 cm^{-1} . HRMS (ESI): m/z calcd for $\text{C}_{16}\text{H}_{14}\text{ClO}_5\text{S}$ $[\text{M}+\text{H}]^+$ 353.0245, found 353.0250.

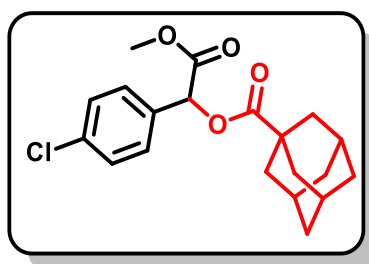
Example 15. 1-(4-chlorophenyl)-2-methoxy-2-oxoethyl 5-chlorothiophene-2-carboxylate (**4o**):



The title compound was synthesised following the general procedure described in section 3.3, and involve corresponding reactant exchange with compound **2a** and compound **3o** (5-chlorothiophene-2-carboxylic acid). The crude mixture was concentrated under vacuum and the product was purified by flash chromatography; $R_f = 0.5$ (10% ethylacetate/hexane); to give compound **4o** (0.47 g in 5 h, 91%) as a colorless oil. $^1\text{H NMR}$ (400 MHz, CDCl_3) δ 7.68 (d, J

= 4.1 Hz, 1H), 7.48 – 7.46 (m, 2H), 7.40 – 7.37 (m, 2H), 6.95 (d, $J = 5.0$ Hz, 1H), 6.08 (s, 1H), 3.74 (s, 3H). **^{13}C NMR (126 MHz, CDCl_3)** δ 168.53, 160.11, 138.53, 135.47, 134.15, 132.03, 130.39, 129.13, 128.95, 127.54, 74.20, 52.87. **IR (ν_{max}):** 3022.57, 1755.23, 1717.81, 1422.40, 1215.11, 1089.70 cm^{-1} . **HRMS (ESI):** m/z calcd for $\text{C}_{14}\text{H}_{11}\text{Cl}_2\text{O}_4\text{S}$ $[\text{M}+\text{H}]^+$ 344.9750, found 344.9756.

Example 16. 1-(4-chlorophenyl)-2-methoxy-2-oxoethyl (1s,3s)-adamantane-1-carboxylate (**4p**):



The title compound was synthesised following the general procedure described in section 3.3, and involve corresponding reactant exchange with compound **2a** and compound **3p** (adamantyl acid). The crude mixture was concentrated under vacuum and the product was purified by flash chromatography; $R_f = 0.5$ (10% ethylacetate/hexane); to give compound **4p** (0.48 g in 5 h, 88%) as viscous oil. **^1H NMR (400 MHz, CDCl_3)** δ 7.42 – 7.40 (m, 2H), 7.35 – 7.32 (m, 2H), 5.88 (s, 1H), 3.68 (s, 3H), 2.02 (s, 3H), 1.97 (s, 6H), 1.71 (s, 6H). **^{13}C NMR (101 MHz, CDCl_3)** δ 176.70, 169.12, 135.10, 132.72, 129.00, 128.80, 73.24, 52.65, 40.75, 38.69, 36.46, 27.90. **IR (ν_{max}):** 2907.80, 2853.75, 1754.93, 1730.89, 1215.77, 1072.97 cm^{-1} . **HRMS (ESI):** m/z calcd for $\text{C}_{20}\text{H}_{24}\text{ClO}_4$ $[\text{M}+\text{H}]^+$ 363.1358, found 363.1364.

S4. Case study-2: Carbene insertion (C-S bond formation) single objective multi-variant auto-optimization.

S4.1 Background collection

Sample Preparation: We prepared 0.5 M stock solutions of compound **2a**, reagent **5a**, and product **6a** in ethyl acetate, each loaded into separate syringes. For our analysis, we utilized an In-line FTIR system. The experimental protocol commenced with the initial introduction of ethyl acetate solvent for 10 minutes using a syringe pump, during which data was recorded. Subsequently, the stock solution of compound **2a** was introduced into the In-line FTIR for another 10 minutes. This procedure was repeated for the reaction product **6a** and reagent **5a**, with each pumped for 10 minutes. After collecting all relevant data, we identified the signature peak present in the product. Using this peak as a reference, we employed a Bayesian optimizer to fine-tune and optimize the reaction conditions for enhanced outcomes.

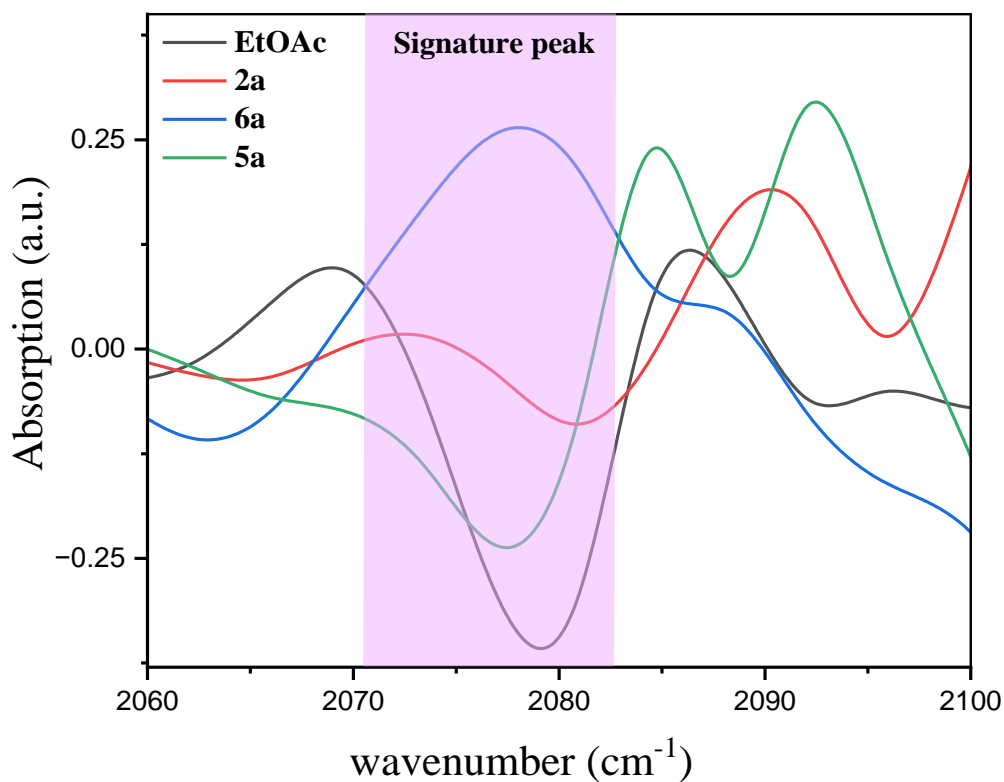


Figure S20. In-line IR background analysis of EtOAc, 0.5 M stock solution of **2a**, **5a**, and **6a** in EtOAc.

The distinctive shifting peak associated with product (**6a**) aligning within the range of 2070 to 2085 cm^{-1} was observed. This specific peak, falling within this range, is designated as the signature peak for our analysis. During the optimization of the reaction through Bayesian methods, this signature peak serves as a pivotal reference point. We utilize it to calculate the area under the curve in the In-line FT-IR spectra, providing a quantitative measure of the area in the forthcoming reaction mixture.

S4.2. General procedure for the auto-optimized synthesis of compound 6a.

In our approach we are employing a Python-coded based Bayesian optimization strategy for further refinement. Initial steps involved the preparation of stock solutions for compound **2a** (0.1 M in EtOAc) and reagent **5a** (0.2 M in EtOAc), each filled into separate syringes and connected to a syringe pump. The solutions were then directed through a PFA tubular reactor (inner diameter = 1 mm, length = 1.3 m, volume = 1 mL) surrounded by a cylindrical-shaped blue LED light source. Once the solution setup was complete, input ranges for variable flow rates, voltages, and current were specified in the Python code designed for the reaction. The Bayesian optimizer systematically explored these varying reaction conditions, aiming to achieve maximum yield. The optimization process involved running 70-80 experiments and the results were tabulated in the optimization table (Table S6).

S4.2.1. Python code for optimizing condition of carbene insertion reaction (C-S bond formation).

```
ab2.py
1 from os import listdir
2 from os.path import isfile, join
3 import serial
4
5 import numpy as np
6 import pandas as pd
7 import time
8 from scipy.integrate import trapz
9
10 #step1: be sure to the address of the files that the ftir data is exported is matching to line 11 (mypath)
11 mypath = "C:\\Users\\Admin\\Desktop\\ruchi\\Exp 2023-10-09 10-24"
12 onlyfiles = [f for f in listdir(mypath) if isfile(join(mypath, f))]
13
14
15 #step2: make sure that pump and the potentiostat is correctly addressed in the line 16 and 17
16 pump_1 = serial.Serial("COM4",9600) #Harvard Pump
17 port = serial.Serial("COM1",115200)
18 printer = serial.Serial("COM11", 115200, timeout=1)
19
20
21
22 #step 3: grab the lines from 22 to 90 and press f9
23 def area_under(data,start,end):
24     x = np.flip(data.iloc[start:end,0].to_numpy())
25     y = np.flip(data.iloc[start:end,1].to_numpy())
26     area = trapz(y,x)
27     return np.abs(area)
28
29 def file_namer(num):
30     str1 = str(num)
31     length = int(len((str1)))
32     empt = ''
33     for i in range(5-length):
34         empt = empt+'0'
35
36     return empt+str1
37
38
39 def ftir_extract(filename,init,end):
40
41     filename = filename
42
43     temp_df = pd.read_csv(filename)
44     # nump_df = temp_df.to_numpy()
45     area = area_under(temp_df, init, end)
46     # max_peak = np.max(nump_df[90:120,1])
47     print(area)
```



```

48
49     return area
50
51
52 def function(flowrate_1,v,i):
53
54     #set the pumps with the flowrate as the desired flowrate for the function
55
56     fr_1 = flowrate_1 #ml/min
57     pump_1.write(('irate '+str(fr_1)+' ml/min\r\n').encode())
58
59     time.sleep(0.1)
60
61     #set the com port for potentiostat and set the voltage and current
62     vol = v
63     curr = i
64     port.write(('VOLT '+str(vol)+'\r\n').encode()) #to change the voltge we need to use "VOLT 1" command
65     port.write(('CURR '+str(curr)+'\r\n').encode()) #to change the current we need to use "CURR 1" command
66
67
68     #pumps run
69     pump_1.write(('irun\r\n'))
70     time.sleep(0.1)
71     time.sleep(2400)
72
73 def function2(flowrate_1,v,i):
74     time.sleep(180)
75
76     files = [f for f in listdir(mypath) if isfile(join(mypath, f))]
77
78     val = 0
79
80     #change wavelengths as per product here.
81     f_row=17 #first row of range for wavelength as per IR CSV
82     l_row=23 #last row of range for wavelength as per IR CSV
83
84     val += ftir_extract(files[-1],f_row,l_row)
85     val+=ftir_extract(files[-2],f_row,l_row)
86     val+=ftir_extract(files[-3],f_row,l_row)
87
88     avg_val = val/3
89
90     return avg_val
91
92
93 #step 4:grab the line 93 and f9
94 from skopt.optimizer import Optimizer

```

```

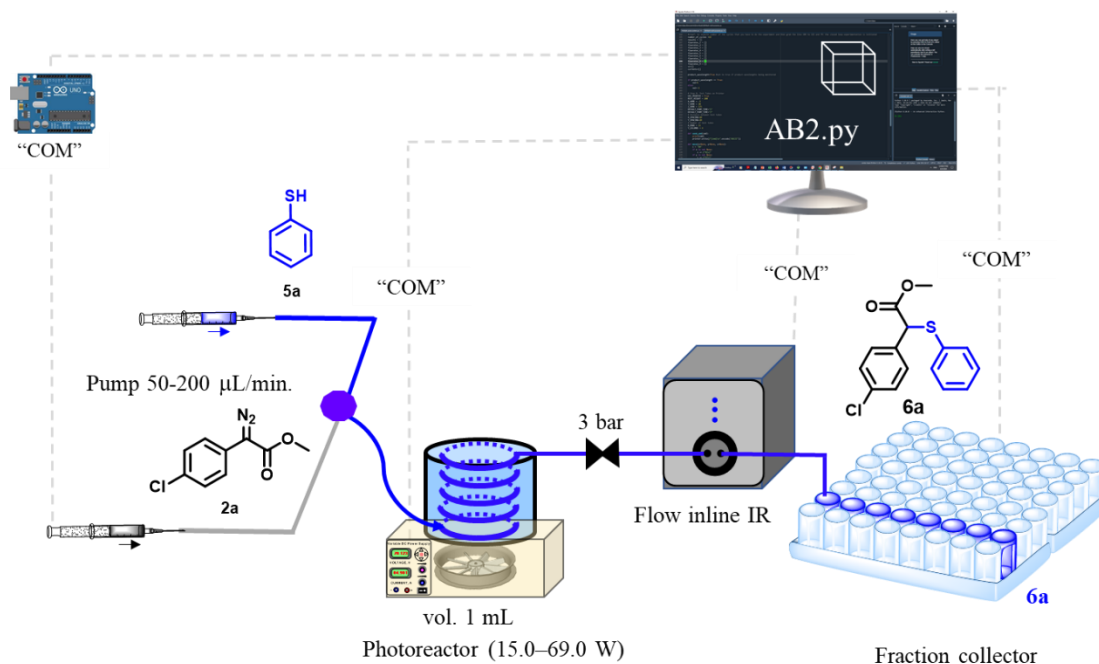
95
96
97 #step5:in line 96 we have to define the range that (flowrate,voltage,current) (from,to) and after (anytime) applying changes you need to grab the line 96 and f9
98 #flowrates are in ml/min, voltage in V, Current in Ampere
99 bounds = [(0.05,0.2),(10,14.5),(4.9,5.0)]
100
101
102 #step 6: grab the line 100 and f9
103 opter =Optimizer(bounds,base_estimator='gp',n_initial_points=3,acq_func="EI",random_state=np.random.randint(3326))
104
105
106 #step7: to selecte number of the cycles that you have to do the experiment and then grab the line 104 to 121 and f9: the closed loop experimentation is initiated
107 number_of_cycles = 22
108 results = []
109 flowrates_1 = []
110 vs = []
111 currents = []
112
113 product_wavelength=True #set to true if product wavelengths being monitored
114
115 if product_wavelength == True:
116     val=1
117 else:
118     val=-1
119
120
121 # Step 8: Test Tubes on Printer
122 USE_PRINTER = True
123 REST_HEIGHT = 200
124 X_HOME = -5
125 Y_HOME = 20
126 Z_HOME = 175
127 DEFAULT_PUMP_TIME="1"
128 # Distance between test tubes
129 X_SPACING=20
130 Y_SPACING=20
131 # Number of test tubes
132 X_ROWS = 11
133 Y_COLUMNS = 4
134
135 def send_cmd(cmd):
136     print(cmd)
137     printer.write(f"{cmd}\n".encode("ASCII"))
138
139 def move(x=None, y=None, z=None):

```

```

140     s = "G0"
141     if x is not None:
142         s += f"X{x}"
143     if y is not None:
144         s += f"Y{y}"
145     if z is not None:
146         s += f"Z{z}"
147
148     s+= "F5000"
149     send_cmd(s)
150
151 def printer_positions():
152     for j in range(Y_COLUMNS):
153         for i in range(X_ROWS):
154             if j%2==1:
155                 yield (X_HOME + (X_ROWS - 1 - i) * X_SPACING, Y_HOME + j * Y_SPACING, Z_HOME)
156             else:
157                 yield (X_HOME + i * X_SPACING, Y_HOME + j * Y_SPACING, Z_HOME)
158
159 # Run this.
160 tube_location = list(printer_positions())
161
162
163 for i in range(number_of_cycles):
164     move(*tube_location[2*i])
165     asked = opter.ask()
166     function(asked[0],asked[1],asked[2])
167     move(*tube_location[2*i+1])
168     told = function2(asked[0],asked[1],asked[2])
169
170     print(f"area under the curve in the round {i:.2f} = {told: .2f}")
171     opter.tell(asked,-told*val)
172     results.append(told)
173     flowrates_1.append(asked[0])
174     vs.append(asked[1])
175     currents.append(asked[2])
176
177     dict1 = {"flowrate_1":flowrates_1,"voltages":vs, "currents":currents,"area-results":results}
178     df2 = pd.DataFrame(dict1)
179     df2.to_csv("output round "+str(i)+".csv")
180
181
182
183 pump_1.write(b'stop\r\n')
184
185
186

```

Table S6. Auto-optimization table of carbene insertion reaction into S-H bonds of thiophenol.

No. of experiments	Flow rate mL/min	Voltage (V)	Light intensity (W)	area-results	Yield (%)	Spacetime yield (g mL ⁻¹ h ⁻¹)
1	0.12187	14.0141	61.3	0.639877	62.1628401	0.066363879
2	0.15077	14.1614	63.583	0.725702	70.500557	0.093113272
3	0.14389	14.3314	65.894	0.771888	74.987459	0.094519922
4	0.17094	14.5	65.85	0.94132	91.4474572	0.136936568
5	0.1	12.3572	39.777	0.536517	52.1216116	0.045658532
6	0.16593	14.0518	62.246	0.886884	86.1591028	0.125236288
7	0.2	14.5	68.778	0.790801	76.8248211	0.134597087
8	0.1894	14.5	68.778	0.910954	88.4974578	0.146830026
9	0.17392	14.5	68.778	0.978956	95.1037234	0.144894251
10	0.2	14	60.96	0.672707	65.3522124	0.114497076
11	0.175	14.5	68.3	0.97634	94.849584	0.145404412
12	0.14684	14.5	68.213	0.909719	88.3774799	0.113681579
13	0.16295	14.5	68.169	0.953319	92.6131374	0.132199882
14	0.2	14	61.002	0.758361	73.6733365	0.129075686
15	0.15126	14.5	68.368	0.919708	89.3478923	0.118389317

16	0.13402	14.2839	66.944	0.797798	77.5045665	0.090991539
17	0.15638	14.0233	61.295	0.669423	65.0331781	0.089088222
18	0.12526	14.0342	61.455	0.745254	72.4000163	0.079442916
19	0.1	14.5	68.281	0.900546	87.4863403	0.076638034
20	0.11313	14.0352	61.319	0.61147	59.4031537	0.058869642
21	0.1	11.0324	25.032	0.206977	20.107424	0.017614103
22	0.16846	14.0379	61.305	0.851698	82.7408495	0.122101466
23	0.18174	12.3572	39.777	0.205725	19.9857946	0.031818232
24	0.18955	11.7409	32.482	0.466023	45.2732528	0.075174335
25	0.13922	14.1921	63.566	0.75473	73.3205917	0.089419389
26	0.14375	14.5	67.904	0.870399	84.5576163	0.106479178
27	0.18405	10.0002	15.42	0.103039	10.0100439	0.016138974
28	0.2	14	60.844	0.794354	77.1699884	0.13520182
29	0.18552	14.3822	66.263	0.894818	86.9298759	0.14127454
30	0.10265	14.104	62.297	0.769329	74.738857	0.067206227
31	0.16804	14.4279	66.956	0.89149	86.606567	0.127487499
32	0.17293	14.5	68.961	0.977279	94.9408061	0.143822675
33	0.1	14.4738	67.545	0.691974	67.2239651	0.058888193
34	0.12095	14.0335	61.184	0.622655	60.4897553	0.064090227
35	0.12869	14.3928	66.309	0.896598	87.1027996	0.098193111
36	0.1	13.466	54.12	0.584978	56.8295061	0.049782647
37	0.1653	14.5	67.41	0.931312	90.475199	0.131010621
38	0.1583	14.0035	60.003	0.89358	86.8096066	0.120379576
39	0.17399	14.0821	61.355	0.855364	83.0969945	0.126652484
40	0.1736	14.0805	61.186	0.830028	80.63565	0.122625536
41	0.19014	14.052	60.143	0.717789	69.7318435	0.116147199
42	0.17339	14.4756	68.784	1.008769	97.9786321	0.148819392
43	0.10246	14.0741	61.785	0.789538	76.6853989	0.068828949
44	0.1083	13.3572	52.066	0.597804	58.0628649	0.055084704
45	0.15031	13.3544	52	0.649264	63.0610165	0.083033424
46	0.16092	14.4505	67.288	0.928714	90.203136	0.127155681
47	0.18975	11.1796	26.503	0.190577	18.5101582	0.03076777
48	0.1005	10	15.48	0.061864	6.00866015	0.005289904

49	0.18795	11.6288	30.954	0.394747	38.3405627	0.063125513
50	0.16037	10.909	23.563	0.227011	22.0488806	0.030975176
51	0.19911	12.9452	46.563	0.687769	66.8008888	0.116514351
52	0.19707	13.7912	57.233	0.794465	77.1639433	0.133210677
53	0.10353	13.9953	60.109	0.823241	79.9588677	0.07251652
54	0.1882	13.6429	54.882	0.765239	74.3253118	0.122535087
55	0.11839	13.1852	49.839	0.751321	72.9734992	0.075680553
56	0.19909	14.499	68.131	0.903623	87.7661243	0.153066613
57	0.10173	10.0506	15.849	0.092939	9.02687938	0.008044347
58	0.11477	12.3255	39.218	0.464055	45.0722357	0.045314959
59	0.12176	12.3888	39.803	0.521891	50.6896686	0.054066493
60	0.19812	12.2467	38.122	0.419045	40.7005527	0.070637079
61	0.14555	12.194	36.286	0.272067	26.4250314	0.033692391
62	0.18235	12.5432	39.924	0.590353	57.3391722	0.091592791
63	0.18723	12.0402	34.157	0.234019	22.7295461	0.03727952
64	0.2	12.29	37.001	0.427683	41.5395351	0.072777266
65	0.12925	10.4935	19.748	0.193939	18.8366989	0.021327476
66	0.10105	10.525	20.027	0.360808	35.0441719	0.031021031
67	0.16517	14.4505	67.288	0.895807	87.0069803	0.12588946
68	0.14869	14.3822	66.263	0.842513	81.8306979	0.106586481
69	0.13311	13.3572	52.066	0.573379	55.6905397	0.064937597
70	0.10145	10.5483	20.261	0.195224	18.961507	0.016851129
71	0.17331	10.5219	20.043	0.288514	28.0224779	0.042543603
72	0.15263	10.5261	20.094	0.355223	34.5017181	0.046130136
73	0.1	13.466	54.12	0.692583	67.2684578	0.058927169
74	0.1	14.5	67.41	0.819158	79.5622985	0.069696573
75	0.19775	14.4028	66.182	0.858552	83.3885166	0.144453093
76	0.11677	10.5449	20.13	0.190198	18.4733471	0.018896483
77	0.12524	11.6656	31.411	0.290907	28.2549027	0.030998522
78	0.1297	14.4989	67.952	0.796329	77.3449879	0.08787721
79	0.17576	14.5	68.3	0.975133	94.7116708	0.145823544
80	0.14697	10.0002	15.42	0.118176	11.4780716	0.014777526
81	0.18865	11.7409	32.482	0.225615	21.9132914	0.036213336

82	0.11754	14.4997	67.304	0.77118	74.9023428	0.077123227
83	0.17818	10.0003	16.025	0.114992	11.1688195	0.017432928
84	0.11092	10	16.01	0.175406	17.0366456	0.016553813

Reaction condition: compound **2a** (0.1 M in EtOAc); reagent **5a** (0.2 M in EtOAc); 1 mL reactor volume.

Graph: We have plotted 3D graph of optimization table S6, we have taken flow rate on X-axis, blue light intensity (watt) on Y-axis and product yield on Z axis.

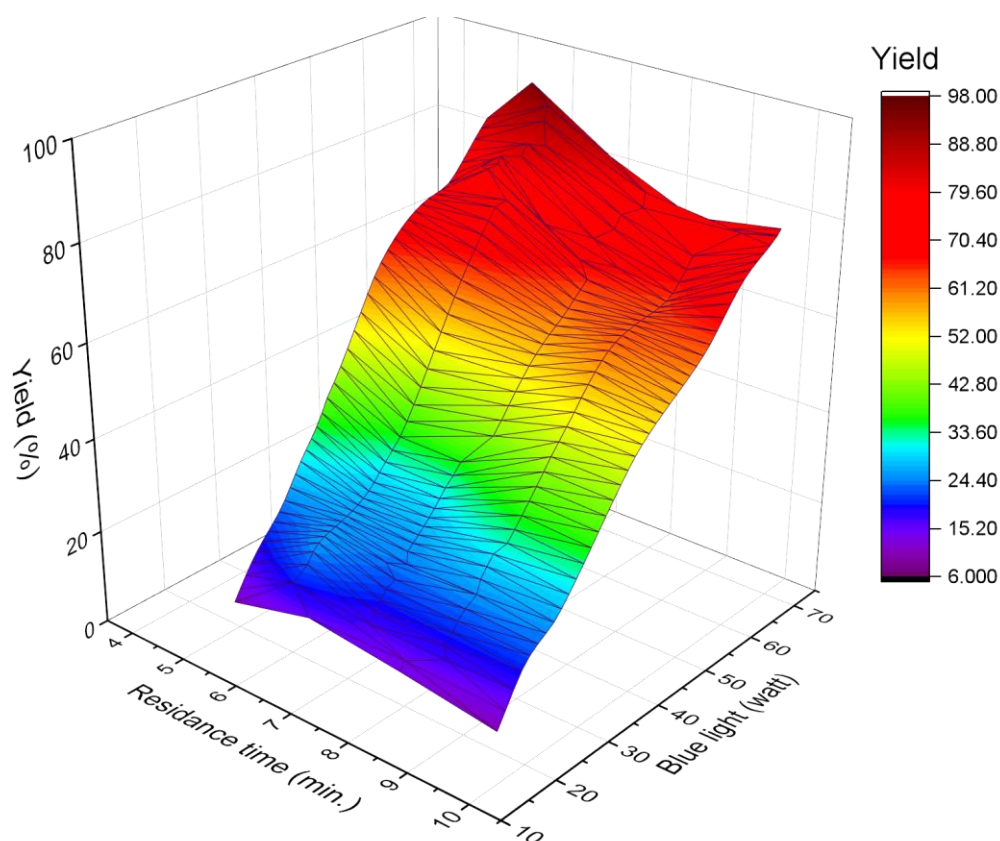


Figure S21. AI based system to auto-optimize and navigate this complexity and identify the optimal conditions for the photo activated S-H insertion reaction. Compound **2a** (0.1 M in EtOAc); reagent **5a** (0.2 M in EtOAc); 1 mL reactor volume.

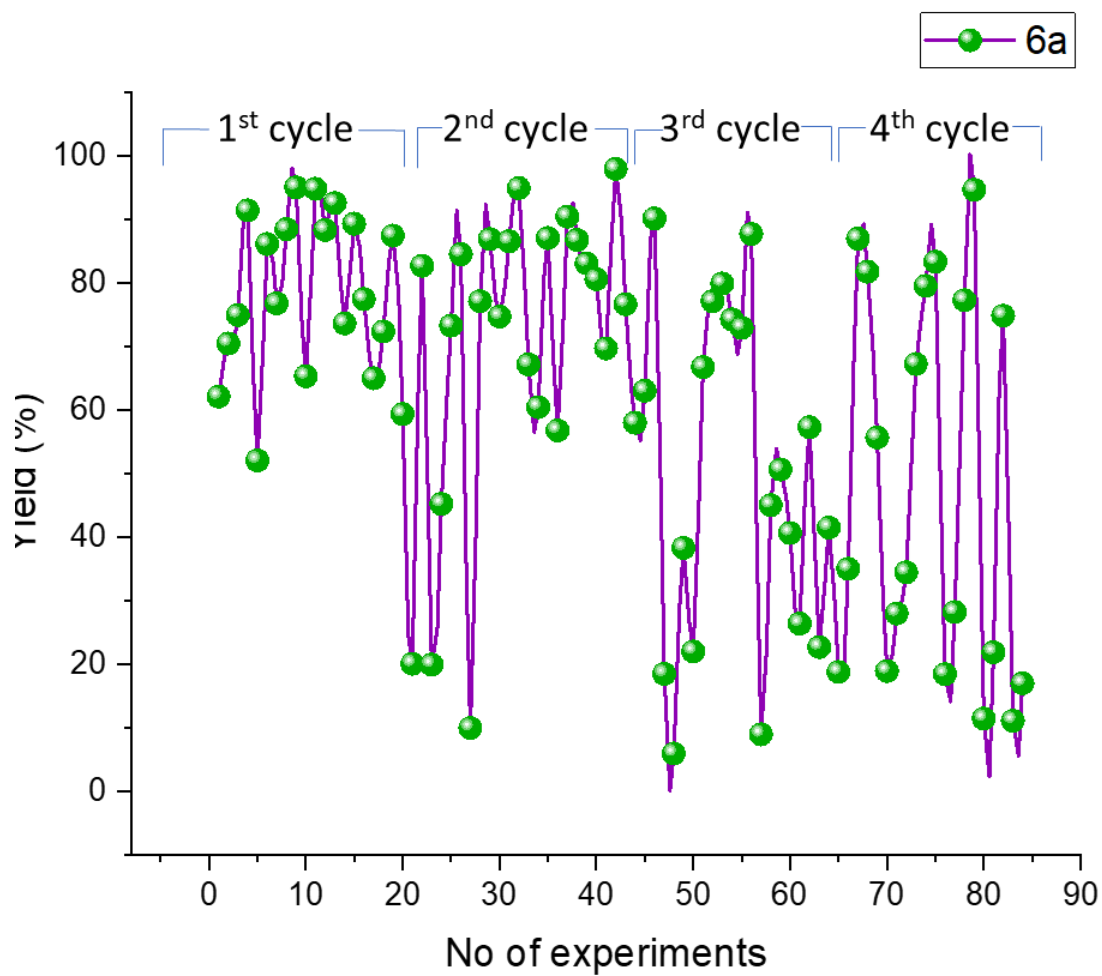


Figure S22. 2D graph between no of experiments performed versus yield (%) for the AI based auto-optimization of photo activated S-H insertion reaction. Compound **2a** (0.1 M in EtOAc); reagent **5a** (0.2 M in EtOAc); 1 mL reactor volume.

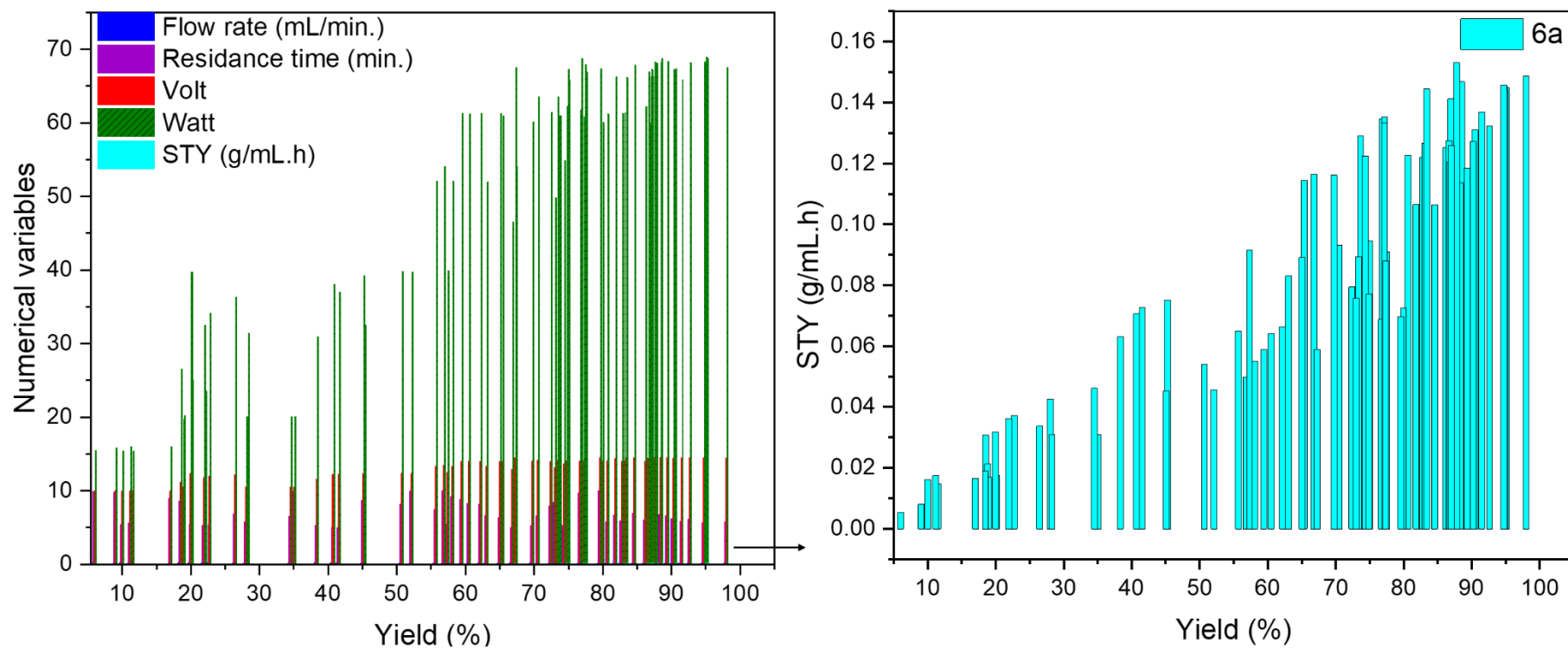


Figure S23. AI based system navigated multi-numerical variable complexity and identify the optimal condition for the photo activated S-H insertion reaction. Compound **2a** (0.1 M in EtOAc); reagent **5a** (0.2 M in EtOAc); 1 mL reactor volume.

S4.3. General procedure of running AI-optimized condition for longer time for synthesis of compound (6a).

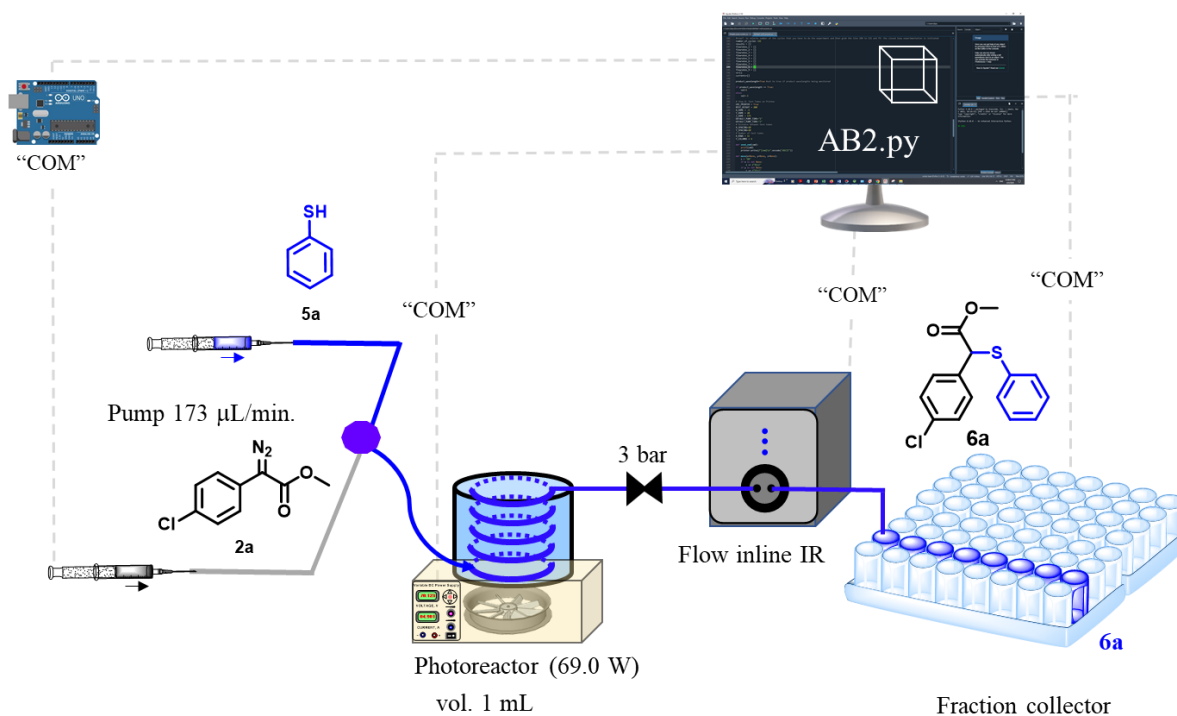


Figure S24. Schematic presentation of continuous flow for the synthesis, of compound **6a**.

To prepare the stock solution of compound **2a** (2.20 g, 0.01 mol) was dissolved in ethyl acetate (100 mL) charged in one syringe, another syringe charged with the reagent **5a** (1.9 mL, 0.02 mol) dissolved in ethyl acetate (100 mL). These two syringes connected via pump and out-put is further connected with the T1-mixer and both syringes were running with the 173 µL/min. flow rate to maintain the stoichiometry and then passed through PFA tubular reactor (id = 1000 µm, l = 1.3 m, vol. = 1 mL) under blue light (69W) exposure. The room temperature of the reactor was controlled by fan attached to the bottom of the photochemical reactor. The out-put of the tubular reactor was connected with spring based back pressure regulator (~3 bar) to maintain the evaporation. Then outcoming first one hour of the product mixture **6a** was discarded and next 5 h of the product mixture [52 mL; **2a** (0.546 g)] was collected in HPLC bottle. The EtOAc reaction mixture washed it with 10% aq. NaOH and separated through the separating funnel. The organic EtOAc phase was concentrated under reduced pressure to give

the product **6a** (0.74 g in 5 h, 97.98%) as a colorless oil. **¹H NMR (400 MHz, CDCl₃)** δ 7.37 – 7.33 (m, 4H), 7.29 – 7.25 (m, 5H), 4.86 (s, 1H), 3.67 (s, 3H). **¹³C NMR (101 MHz, CDCl₃)** δ 170.55, 134.32, 133.16, 133.10, 129.98, 129.14, 128.89, 128.40, 55.73, 52.88. **IR (ν_{max}):** 3021.35, 2952.59, 1737.18, 1487.9, 1214.19, 829.21 cm⁻¹. **HRMS (ESI):** *m/z* calcd for **C₁₅H₁₄ClO₂S** [M+H]⁺ 293.0398, found 293.0404. Verified the analytical data with those reported in the literature.⁴

Space time yield in previous reported batch process ⁴

Productivity = 130 mg in 12 h

$$\text{Productivity (per h)} := \frac{0.130}{12} = 0.011 \text{ g/h}$$

$$\text{Space time yield} := \frac{\text{productivity}}{\text{volume of reactor}}$$

$$= \frac{0.011}{1} = 0.011 \text{ g mL}^{-1} \text{ h}^{-1}$$

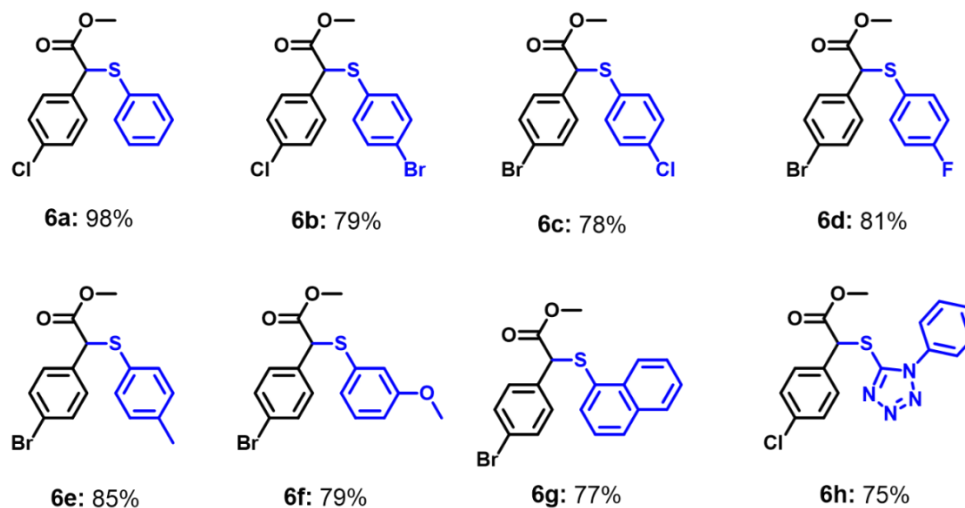
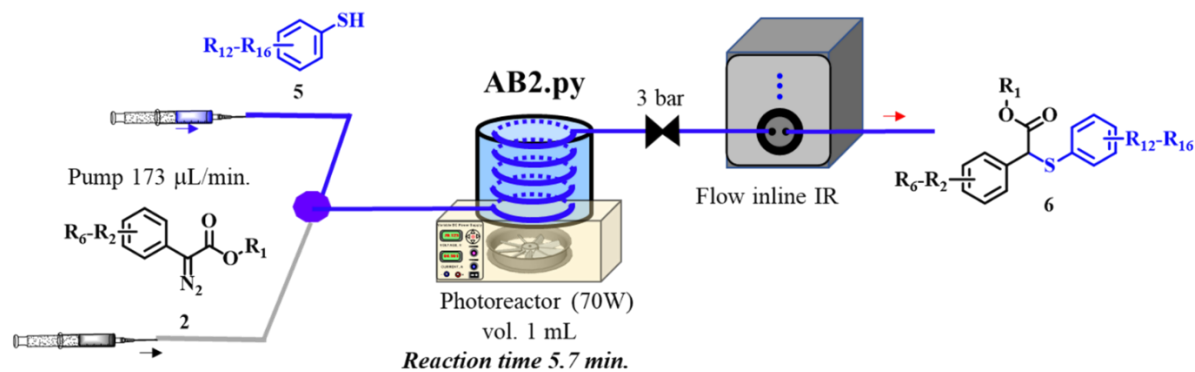
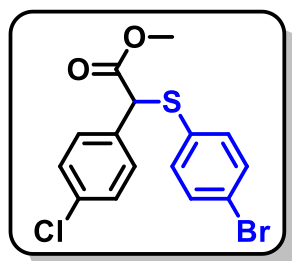


Figure S25. Scope of substrate for the photo activated S-H insertion reaction. Compound **2** (0.1 M in EtOAc); reagent **5** (0.2 M in EtOAc); 1 mL reactor volume.

S4.3.2. General procedure for scope of substrate for carbene insertion into the S–H bonds of organic aromatic thiol for synthesis of compound **6b–6h**.

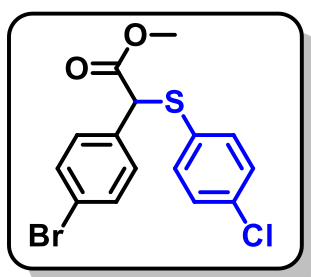
Example 18. Methyl 2-((4-bromophenyl)thio)-2-(4-chlorophenyl)acetate (**6b**).



The title compound was synthesised following the general procedure described in section 4.3, and involve corresponding reactant exchange with compound **2a** and compound **5b** (4-bromobenzenethiol). The crude mixture was concentrated under vacuum and the product was

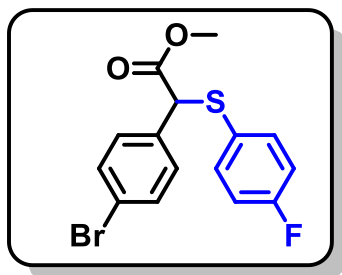
purified by flash chromatography; $R_f = 0.5$ (5% ethylacetate/hexane); to give compound **6b** (0.76 g in 5 h, 79%) as a colorless oil. $^1\text{H NMR}$ (400 MHz, CDCl_3) δ 7.39 – 7.32 (m, 4H), 7.30 – 7.25 (m, 2H), 7.22 – 7.17 (m, 2H), 4.83 (s, 1H), 3.68 (s, 3H). $^{13}\text{C NMR}$ (126 MHz, CDCl_3) δ 170.23, 137.70, 134.68, 132.24, 129.94, 128.98, 122.88, 55.62, 52.98. IR (ν_{max}): 3020.98, 1736.81, 1483.63, 1214.01, 817.91, 667.71 cm^{-1} . HRMS (ESI): m/z calcd. for $\text{C}_{15}\text{H}_{13}\text{BrClO}_2\text{S}$ $[\text{M}+\text{H}]^+$ 370.9503, found 370.9509.

Example 19. Methyl 2-(4-bromophenyl)-2-((4-chlorophenyl)thio)acetate (**6c**).



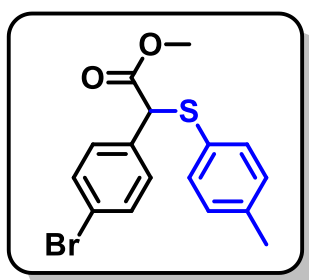
The title compound was synthesised following the general procedure described in section 4.3, and involve corresponding reactant exchange with compound **2b** and compound **5c** (4-chlorobenzenethiol). The crude mixture was concentrated under vacuum and the product was purified by flash chromatography; $R_f = 0.5$ (5% ethylacetate/hexane); to give compound **6c** (0.75 g in 5 h, 78%) as a colorless oil. $^1\text{H NMR}$ (400 MHz, CDCl_3) δ 7.42 (d, $J = 8.4$ Hz, 2H), 7.28 – 7.25 (m, 4H), 7.21 (d, $J = 8.6$ Hz, 2H), 4.81 (s, 1H), 3.66 (s, 3H). $^{13}\text{C NMR}$ (126 MHz, CDCl_3) δ 170.15, 134.75, 134.59, 134.47, 131.91, 131.43, 130.26, 129.30, 122.66, 55.79, 52.96. IR (ν_{max}): 3019.95, 2952.20, 1735.81, 1497.43, 1213.23, 817.44, 626.54 cm^{-1} . HRMS (ESI): m/z calcd for $\text{C}_{15}\text{H}_{13}\text{BrClO}_2\text{S}$ $[\text{M}+\text{H}]^+$ 370.9503, found 370.9510.

Example 20. Methyl 2-(4-bromophenyl)-2-((4-fluorophenyl)thio)acetate (**6d**).



The title compound was synthesised following the general procedure described in section 4.3, and involve corresponding reactant exchange with compound **2b** and compound **5d** (4-fluorobenzenethiol). The crude mixture was concentrated under vacuum and the product was purified by flash chromatography; $R_f = 0.5$ (5% ethylacetate/hexane); to give compound **6d** (0.74 g in 5 h, 81%) as a colorless oil. $^1\text{H NMR}$ (500 MHz, CDCl_3) δ 7.42–7.40 (m, 2H), 7.34–7.32 (m, 2H), 7.27–7.24 (m, 2H), 6.96–6.92 (m, 2H), 4.77 (s, 1H), 3.64 (s, 3H). $^{13}\text{C NMR}$ (101 MHz, CDCl_3) δ 170.25, 164.38, 161.90, 136.37, 136.29, 134.63, 131.81, 130.29, 127.85, 127.82, 122.52, 116.34, 116.13, 56.26, 52.83. $^{19}\text{F NMR}$ (471 MHz, CDCl_3) δ -111.71. IR (ν_{max}): 3021.30, 1737.68, 1437.42, 1216.36, 828.38, 668.88 cm^{-1} . HRMS (ESI): m/z calcd for $\text{C}_{15}\text{H}_{13}\text{BrFO}_2\text{S}$ $[\text{M}+\text{H}]^+$ 354.9798, found 354.9802.

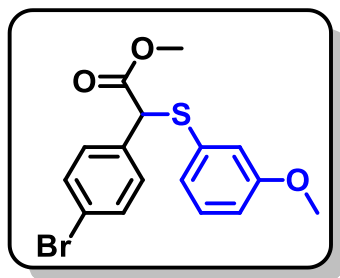
Example 21. Methyl 2-(4-bromophenyl)-2-(p-tolylthio)acetate (**6e**).



The title compound was synthesised following the general procedure described in section 4.3, and involve corresponding reactant exchange with compound **2b** and compound **5e** (4-methylbenzenethiol). The crude mixture was concentrated under vacuum and the product was purified by flash chromatography; $R_f = 0.5$ (5% ethylacetate/hexane); to give compound **6e** (0.77 g in 5 h, 85%) as a colorless oil. $^1\text{H NMR}$ (400 MHz, CDCl_3) δ 7.42 (d, $J = 8.5$ Hz, 2H),

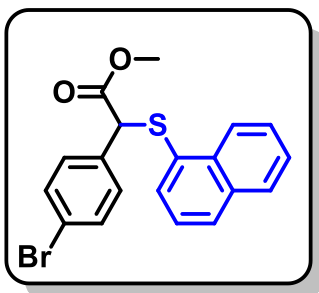
7.30 – 7.24 (m, 4H), 6.79 (d, $J = 8.9$ Hz, 2H), 4.70 (s, 1H), 3.77 (s, 3H), 3.66 (s, 3H). ^{13}C NMR (101 MHz, CDCl_3) δ 170.57, 160.44, 136.48, 135.02, 131.72, 130.36, 123.05, 122.33, 114.63, 56.67, 55.36, 52.75. IR (ν_{max}): 3020.59, 1735.43, 1437.72, 1214.76, 1010.62, 826.89 cm^{-1} . HRMS (ESI): m/z calcd for $\text{C}_{16}\text{H}_{16}\text{BrO}_2\text{S}$ $[\text{M}+\text{H}]^+$ 351.0049, found 351.0055.

Example 22. Methyl 2-(4-bromophenyl)-2-((3-methoxyphenyl)thio)acetate (**6f**).



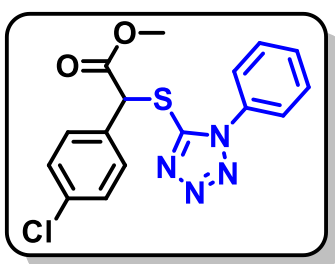
The title compound was synthesised following the general procedure described in section 4.3, and involve corresponding reactant exchange with compound **2b** and compound **5f** (3-methoxybenzenethiol). The crude mixture was concentrated under vacuum and the product was purified by flash chromatography; $R_f = 0.5$ (5% ethylacetate/hexane); to give compound **6f** (0.74 g in 5 h, 79%) as a yellow oil. ^1H NMR (500 MHz, CDCl_3) δ 7.44 – 7.42 (m, 2H), 7.31 – 7.30 (m, 2H), 7.18 – 7.14 (m, 1H), 6.93 – 6.91 (m, 1H), 6.86 – 6.85 (m, 1H), 6.80 – 6.78 (m, 1H), 4.87 (s, 1H), 3.71 (s, 3H), 3.67 (s, 3H). ^{13}C NMR (101 MHz, CDCl_3) δ 170.39, 159.74, 134.82, 134.33, 131.82, 130.25, 129.89, 124.70, 122.47, 117.58, 114.29, 55.56, 55.26, 52.90. IR (ν_{max}): 3019.75, 1737.02, 1481.61, 1215.45, 669.75 cm^{-1} . HRMS (ESI): m/z calcd for $\text{C}_{16}\text{H}_{16}\text{BrO}_3\text{S}$ $[\text{M}+\text{H}]^+$ 366.9998, found 366.9993.

Example 23. Methyl 2-(4-bromophenyl)-2-(naphthalen-1-ylthio)acetate (**6g**).



The title compound was synthesised following the general procedure described in section 4.3, and involve corresponding reactant exchange with compound **2b** and compound **5g** (3-methoxybenzenethiol). The crude mixture was concentrated under vacuum and the product was purified by flash chromatography; $R_f = 0.5$ (5% ethylacetate/hexane); to give compound **6g** (0.77 g in 6 h, 77%) as a colorless oil. $^1\text{H NMR}$ (400 MHz, CDCl_3) δ 7.83 (d, $J = 1.4$ Hz, 1H), 7.77 – 7.74 (m, 1H), 7.72 – 7.69 (m, 2H), 7.50 – 7.42 (m, 3H), 7.41 – 7.38 (m, 2H), 7.33 – 7.31 (m, 2H), 4.95 (s, 1H), 3.64 (s, 3H). $^{13}\text{C NMR}$ (101 MHz, CDCl_3) δ 170.47, 134.80, 133.56, 132.77, 132.22, 131.88, 130.50, 130.29, 129.72, 128.75, 127.74, 127.66, 126.74, 126.69, 122.55, 55.73, 52.92. **IR** (ν_{max}): 3053.68, 2951.64, 1738.16, 1489.43, 1212.16, 634.22 cm^{-1} . **HRMS** (ESI): m/z calcd for $\text{C}_{19}\text{H}_{16}\text{BrO}_2\text{S}$ $[\text{M}+\text{H}]^+$ 387.0049, found 387.0053.

Example 24. Methyl 2-(4-chlorophenyl)-2-((1-phenyl-1H-tetrazol-5-yl)thio)acetate (**6h**).



The title compound was synthesised following the general procedure described in section 4.3, and involve corresponding reactant exchange with compound **2a** and compound **5h** (1-phenyl-1H-tetrazole-5-thiol). The crude mixture was concentrated under vacuum and the product was purified by flash chromatography; $R_f = 0.5$ (5% ethylacetate/hexane); to give compound **6h** (0.70 g in 6 h, 75%) as a colorless oil. $^1\text{H NMR}$ (400 MHz, CDCl_3) δ 7.56 – 7.49 (m, 5H),

7.45 – 7.42 (m, 2H), 7.32 – 7.29 (m, 2H), 5.78 (s, 1H), 3.76 (s, 3H). **¹³C NMR (126 MHz, CDCl₃)** δ 169.17, 152.44, 135.38, 133.34, 132.35, 130.39, 129.93, 129.92, 129.38, 123.77, 54.07, 53.70. **IR (ν_{max}):** 3018.80, 2955.24, 1741.03, 1216.96, 1009.64, 832.24 cm⁻¹. **HRMS (ESI):** *m/z* calcd for C₁₆H₁₄ClN₄O₂S [M+H]⁺ 361.0521, found 361.0526

Table S7. Comparative result for the carbene insertion into S-H bonds of thiols.

Entry	Compound 2a	Reagent 5a	Product 6a	Comparative result
1				98%, 5.7 min. (our study) [(CH ₃ CN) ₄ Cu]PF ₆ (5 mol%), 12-24 h, 89% 4 Fe(OTf) ₂ (15 mol%), 24-48 h, 35 % 5

S5. Case study-3: Carbene insertion (C-N bond formation) single objective multi-variant auto-optimization.

S5.1. Background collection

Sample Preparation: We prepare 0.5 M stock solution of compound **2a** in ethyl acetate, 0.5 M stock solution of reagent **7a** in ethyl acetate and 0.5 M stock solution of product **8a** in ethyl acetate. The three of the solution taken in three different syringes.

Delivery system: The HPLC pumps and syringe pump were directly interfaced to the main computer via an RS-232 interface or LAN. These pumps will help in introducing reactants and reagent solutions into the microreactor with varied flow rate as the central computer's prescribed flow rate (fr). The central computer and the HPLC pump communicated through serial communication using ASCII code to exchange information about flow rate, operational status, and duration. This facilitated the transmission of essential details from the main computer to the pumps, enabling them to commence their operations as required. To conduct our analysis, we utilized an In-line FTIR system. The experimental protocol involved the initial introduction of the solvent, ethyl acetate, for a duration of 10 min. using a syringe pump, during which data was recorded. Subsequently, a previously prepared stock solution of compound **2a** was similarly introduced into the In-line FTIR for 10 min. This process was repeated for the reaction product **8a** and reagent **7a**, each pumped for 10 min. Upon collecting all relevant data, we identified the signature peak present in the product. Using this peak as a reference, we employed a Bayesian optimizer to fine-tune and optimize the reaction conditions for improved results.

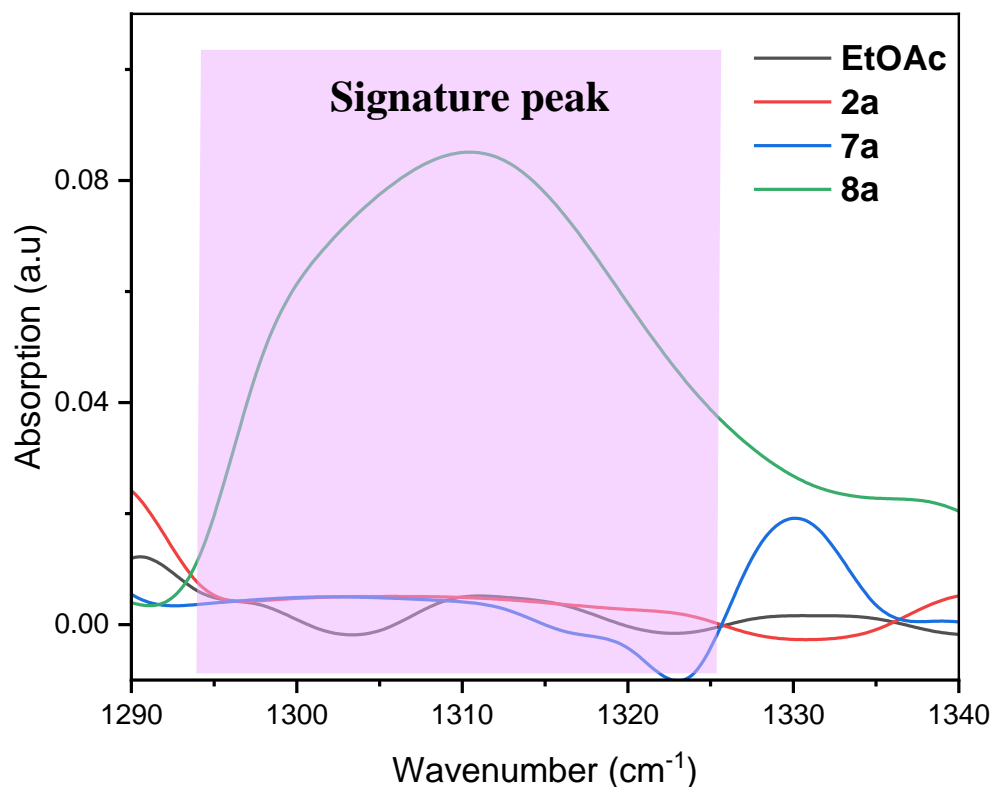


Figure S26. In-line IR background analysis of EtOAc, 0.5 M stock solution of **2a**, **7a**, and **8a** in EtOAc.

The distinctive C-N stretching associated with product (**8a**) aligning within the range of 1295 to 1325 cm^{-1} was observed. This specific peak, falling within this range, is designated as the signature peak for our analysis. During the optimization of the reaction through Bayesian methods, this signature peak serves as a pivotal reference point. We utilize it to calculate the area under the curve in the In-line FT-IR spectra, providing a quantitative measure of the area in the forthcoming reaction mixture.

S5.2. General procedure for the auto-optimized synthesis of compound **8a**.

Initial steps involved the preparation of stock solutions for compound **2a** (0.1 M in EtOAc) and reagent **7a** (0.3 M in EtOAc), each filled into separate syringes and connected to a syringe pump. The solutions were then directed through a PFA tubular reactor (inner diameter = 1 mm,

length = 1.3 m, volume = 1 mL) surrounded by a cylindrical-shaped blue LED light source. Once the solution setup was complete, input ranges for variable flow rates, voltages, and current were specified in the Python code designed for the reaction. The Bayesian optimizer systematically explored these varying reaction conditions, aiming to achieve maximum yield. The optimization process involved running 40-44 experiments and the results were tabulated in the optimization table (Table S8).

S5.2.1. Python code for optimizing condition of carbene insertion reaction (C-N bond formation).

```
ab3.py
1 from os import listdir
2 from os.path import isfile, join
3 import serial
4
5 import numpy as np
6 import pandas as pd
7 import time
8 from scipy.integrate import trapz
9
10 #step1: be sure to the address of the files that the ftir data is exported is matching to line 11 (mypath)
11 mypath = "C:\\Users\\Admin\\Desktop\\ruchi\\Exp 2024-02-27 13-13"
12 onlyfiles = [f for f in listdir(mypath) if isfile(join(mypath, f))]
13
14 #step2: make sure that pump and the potentiostat is correctly addressed in the line 16 and 17
15 pump_1 = serial.Serial("COM15",9600) #Syringe Pump
16 port = serial.Serial("COM1",115200)
17 printer = serial.Serial("COM14", 115200, timeout=1)
18
19 r_1=13.36 #radius of syringe used (50ml,14.25|20ml,9.6|10ml,7.25|5ml,6.03|3ml,4/3|1ml,2.39|60ml,13.36)
20 pump_1_k=(13.36/r_1)**2
21
22 #step 3: grab the lines from 22 to 90 and presss f9
23 def area_under(data,start,end):
24     x = np.flip(data.iloc[start:end,0].to_numpy())
25     y = np.flip(data.iloc[start:end,1].to_numpy())
26     area = trapz(y,x)
27     return np.abs(area)
28
29 def file_namer(num):
30     str1 = str(num)
31     length = int(len((str1)))
32     empt = ''
33     for i in range(5-length):
34         empt = empt+'0'
35
36     return empt+str1
37
38
39 def ftir_extract(filename,init,end):
40
41     filename = filename
42
43     temp_df = pd.read_csv(filename)
44     # nump_df = temp_df.to_numpy()
45     area = area_under(temp_df, init, end)
46     # max_peak = np.max(nump_df[90:120,1])
47     print(area)
```

```

48
49     return area
50
51
52 def function(flowrate_1,v,i):
53
54     #set the pumps with the flowrate as the desired flowrate for the function
55
56     fr_1 = flowrate_1*1000*pump_1_k    #mL/min to microliter/min and syringe correction factor
57     send_syr_command(f'fr{fr_1}')
58     time.sleep(0.1)
59
60     #set the com port for potentiostat and set the voltage and current
61     vol = v
62     curr = i
63     port.write(('VOLT '+str(vol)+'\r\n').encode())    #to change the voltge we need to use "VOLT 1" command
64     port.write(('CURR '+str(curr)+'\r\n').encode())    #to change the current we need to use "CURR 1" command
65
66
67     #pumps run
68     time.sleep(0.1)
69     time.sleep(1000)
70
71 def function2():
72     time.sleep(1000)
73
74     files = [f for f in listdir(mypath) if isfile(join(mypath, f))]
75
76     val = 0
77
78     #change wavelengths as per product here.
79     f_row=16 #first row of range for wavelength as per IR CSV
80     l_row=24 #Last row of range for wavelength as per IR CSV
81
82     val += ftir_extract(files[-1],f_row,l_row)
83     val+=ftir_extract(files[-2],f_row,l_row)
84     val+=ftir_extract(files[-3],f_row,l_row)
85
86     avg_val = val/3
87
88     return avg_val
89
90
91 #step 4:grab the line 93 and f9
92 from skopt.optimizer import Optimizer
93
94

```

```

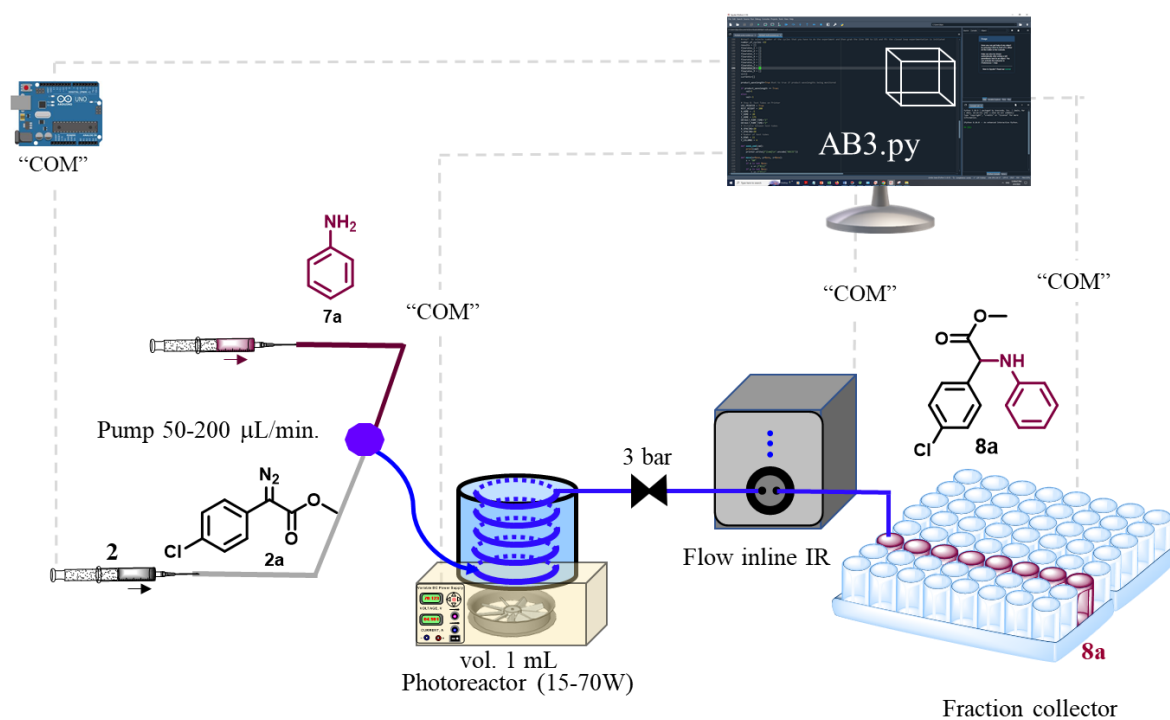
95 def send_syr_command(command):
96     pump_1.write(command.encode() + b'\n') # Send the command to Arduino with a newline character
97     pump_1.flush() # Flush the serial buffer
98
99
100 #step5: in line 96 we have to define the range that (flowrate,voltage,current) (from,to) and after (anytime) applying changes you need to grab the line 96 and f9
101 #flowrates are in ml/min, voltage in V, Current in Ampere
102 bounds = [(0.05,0.2),(10,14.5),(4.9,5.0)]
103
104 #step 6: grab the line 100 and f9
105 opter =Optimizer(bounds,base_estimator='gp',n_initial_points=3,acq_func="EI",random_state=np.random.randint(3326))
106
107
108 #step7: to selecte number of the cycles that you have to do the experiment and then grab the line 104 to 121 and f9: the closed loop experimentation is initiated
109 number_of_cycles = 22
110 results = []
111 flowrates_1 = []
112 vs = []
113 currents = []
114
115 product_wavelength=True #set to true if product wavelengths being monitored
116
117 if product_wavelength == True:
118     val=1
119 else:
120     val=-1
121
122
123 # Step 8: Test Tubes on Printer
124 USE_PRINTER = True
125 REST_HEIGHT = 200
126 X_HOME = -5
127 Y_HOME = 20
128 Z_HOME = 175
129 DEFAULT_PUMP_TIME="1"
130 # Distance between test tubes
131 X_SPACING=20
132 Y_SPACING=20
133 # Number of test tubes
134 X_ROWS = 11
135 Y_COLUMNS = 4
136
137 def send_cmd(cmd):
138     print(cmd)
139     printer.write(f"{cmd}\n".encode("ASCII"))
140
141 def move(x=None, y=None, z=None):

```

```

142 s = "G0"
143 if x is not None:
144     s += f"X{x}"
145 if y is not None:
146     s += f"Y{y}"
147 if z is not None:
148     s += f"Z{z}"
149
150 s+= "F5000"
151 send_cmd(s)
152
153 def printer_positions():
154     for j in range(Y_COLUMNS):
155         for i in range(X_ROWS):
156             if j%2==1:
157                 yield (X_HOME + (X_ROWS - 1 - i) * X_SPACING, Y_HOME + j * Y_SPACING, Z_HOME)
158             else:
159                 yield (X_HOME + i * X_SPACING, Y_HOME + j * Y_SPACING, Z_HOME)
160
161 # Run this.
162 tube_location = list(printer_positions())
163
164 time.sleep(2)
165
166 for i in range(number_of_cycles):
167     move(*tube_location[2*i])
168     asked = opter.ask()
169     print (asked[0])
170     function(asked[0],asked[1],asked[2])
171     move(*tube_location[2*i+1])
172     told = function2()
173
174     print(f"area under the curve in the round {i:.2f} = {told: .2f}")
175     opter.tell(asked,-told*val)
176     results.append(told)
177     flowrates_1.append(asked[0])
178     vs.append(asked[1])
179     currents.append(asked[2])
180
181     dict1 = {"flowrate_1":flowrates_1,"voltages":vs, "currents":currents,"area-results":results}
182     df2 = pd.DataFrame(dict1)
183     df2.to_csv("output round "+str(i)+".csv")
184
185
186
187 send_syr_command("stop")    #turns off syringe pump flow
188

```


Table S8. Auto-optimization table of carbene insertion reaction (C-N bond formation).

No. of experiments	Flow rate mL/min	Voltage (V)	Light intensity (W)	area-results	Yield (%)	Spacetime yield (g mL ⁻¹ h ⁻¹)
1	0.118968	12.38799	40.459	1.196524	71.6165410	0.0702906
2	0.093580	10.18322	17.197	0.279182	16.7101331	0.012900
3	0.111095	13.78164	59.183	1.514104	90.6249622	0.0830608
4	0.05	11.07727	25.851	0.628177	37.5988264	0.0155095
5	0.066191	14.23180	65.638	1.482614	88.7401704	0.0484595
6	0.061313	10.04277	15.985	0.343314	20.5487147	0.0103943
7	0.115638	13.65416	57.251	1.346334	80.5832479	0.0768775
8	0.069548	12.82883	17.25	0.234918	14.0607889	0.0080677
9	0.099196	11.69255	69.716	1.506151	90.1489571	0.0737751
10	0.093306	10.18078	17.163	0.291460	17.4450195	0.0134288
11	0.092505	14.3295	68.113	1.537078	92.0000251	0.0702116
12	0.051560	14.48697	69.75	1.456641	87.1855572	0.0370862
13	0.111049	13.75428	58.757	1.311133	78.4763496	0.0718964
14	0.118626	10.00002	15.663	0.210988	12.6248993	0.0123556
15	0.050115	10.06854	16.238	0.178406	10.6783159	0.0044149

16	0.062578	13.06313	49.698	1.021805	61.158672	0.0315746
17	0.100030	12.22525	38.894	1.029414	61.6143808	0.0508472
18	0.05	10.10525	16.638	0.240613	14.4016513	0.0059406
19	0.111160	13.82081	60.249	1.373565	82.2131677	0.0753953
20	0.093994	11.74773	33.39	0.921292	55.1429018	0.0427608
21	0.122704	14.31194	67.567	1.225070	73.3251847	0.0742279
22	0.057572	10.62465	21.511	0.980855	58.7079763	0.0278848
23	0.086404	14	62.807	1.494519	89.4526871	0.0637649
24	0.148922	11	25.125	0.133867	8.01247153	0.0098441
25	0.052426	12	36.174	1.244392	74.4816629	0.0322148
26	0.15	11	25.18	0.156175	9.34772818	0.0115678
27	0.05	10.0673	15.564	0.303881	18.1884792	0.0075027
28	0.05	10	15.604	0.378768	22.6707418	0.0093516
29	0.15	13	48.67	1.093202	65.4323299	0.0809725
30	0.063750	11.6785	32.683	1.165559	69.7631768	0.0366912
31	0.052816	12.9356	47.678	1.242195	74.3501624	0.0323970
32	0.05	13.3951	53.794	1.265565	75.7489373	0.0312464
33	0.055842	11	29.217	0.797093	47.7090838	0.0219795
34	0.101936	14.0832	67.409	1.504863	90.0718120	0.0757479
35	0.05	13.0828	49.985	0.773444	46.2935985	0.0190961
36	0.132422	11	26.081	0.482431	28.8753526	0.0315458
37	0.05	10.085	16.437	0.105391	6.30806718	0.0026020
38	0.067590	12.315	40.073	1.017212	60.8840568	0.0339503
39	0.089018	14	63.507	1.528104	91.4628977	0.0671702
40	0.053891	11.315	28.469	0.783364	46.8873703	0.0208464
41	0.139535	13.38552	58.881	1.272373	76.1564519	0.0876686
42	0.1282496	11.937	35.17	0.748672	44.8109424	0.0474126
43	0.15	13.2041	51.367	0.640692	38.3479029	0.0474555
44	0.068291	14	62.656	1.513058	90.562340	0.0510233

Reaction condition: compound **2a** (0.1 M in EtOAc); reagent **7a** (0.3 M in EtOAc); 1 mL reactor volume.

Graph: We have plotted 3D graph of optimization table S8, we have taken flow rate on X-axis, blue light intensity (watt) on Y-axis and product yield on Z axis.

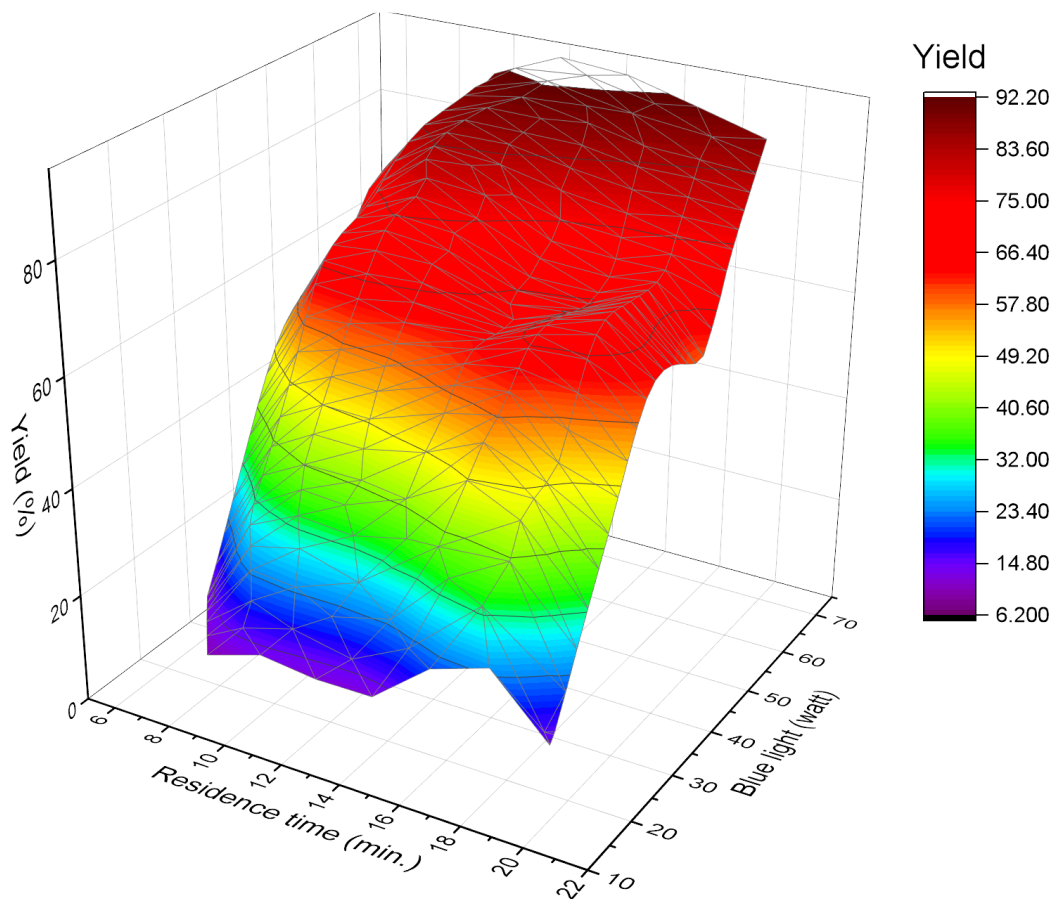


Figure S27. AI based system to auto-optimize and navigate this complexity and identify the optimal conditions for the photo activated N-H insertion reaction. Compound **2a** (0.1 M in EtOAc); reagent **7a** (0.3 M in EtOAc); 1 mL reactor volume.

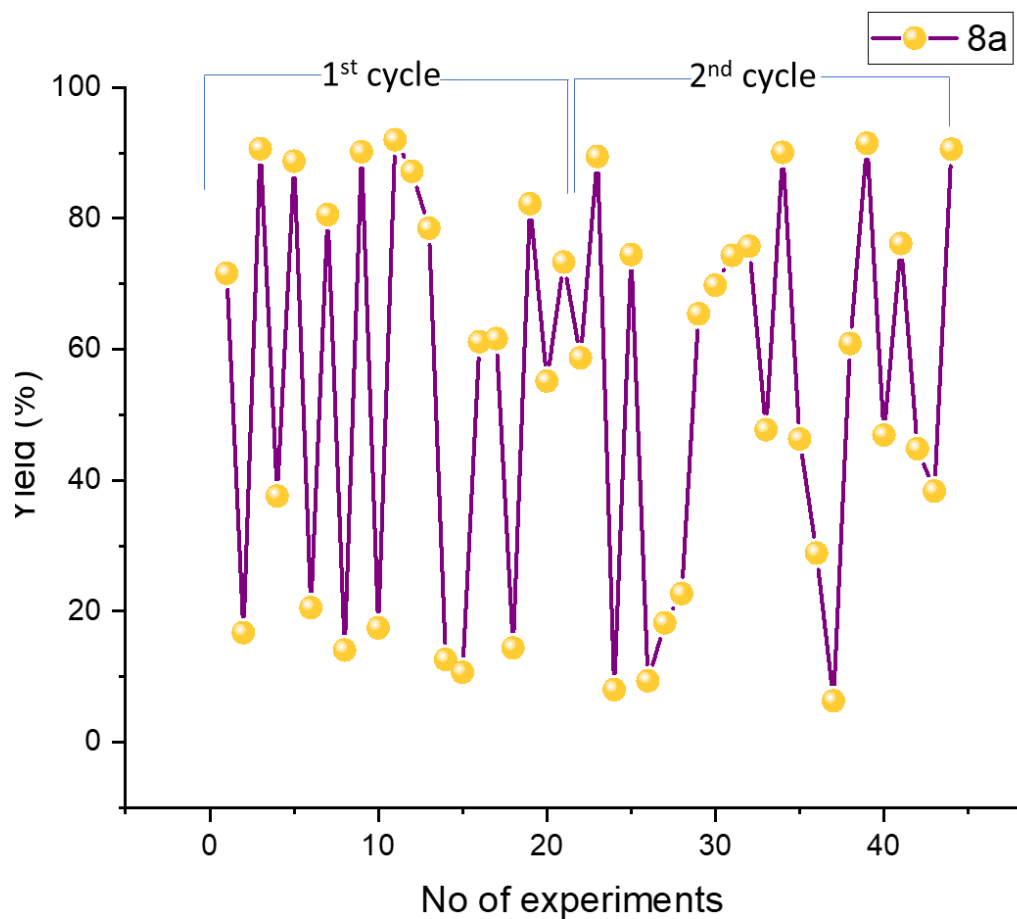


Figure S28. 2D graph between no of experiments performed versus yield (%) for the AI based auto-optimization of photo activated N-H insertion reaction. Compound **2a** (0.1 M in EtOAc); reagent **7a** (0.3 M in EtOAc); 1 mL reactor volume.

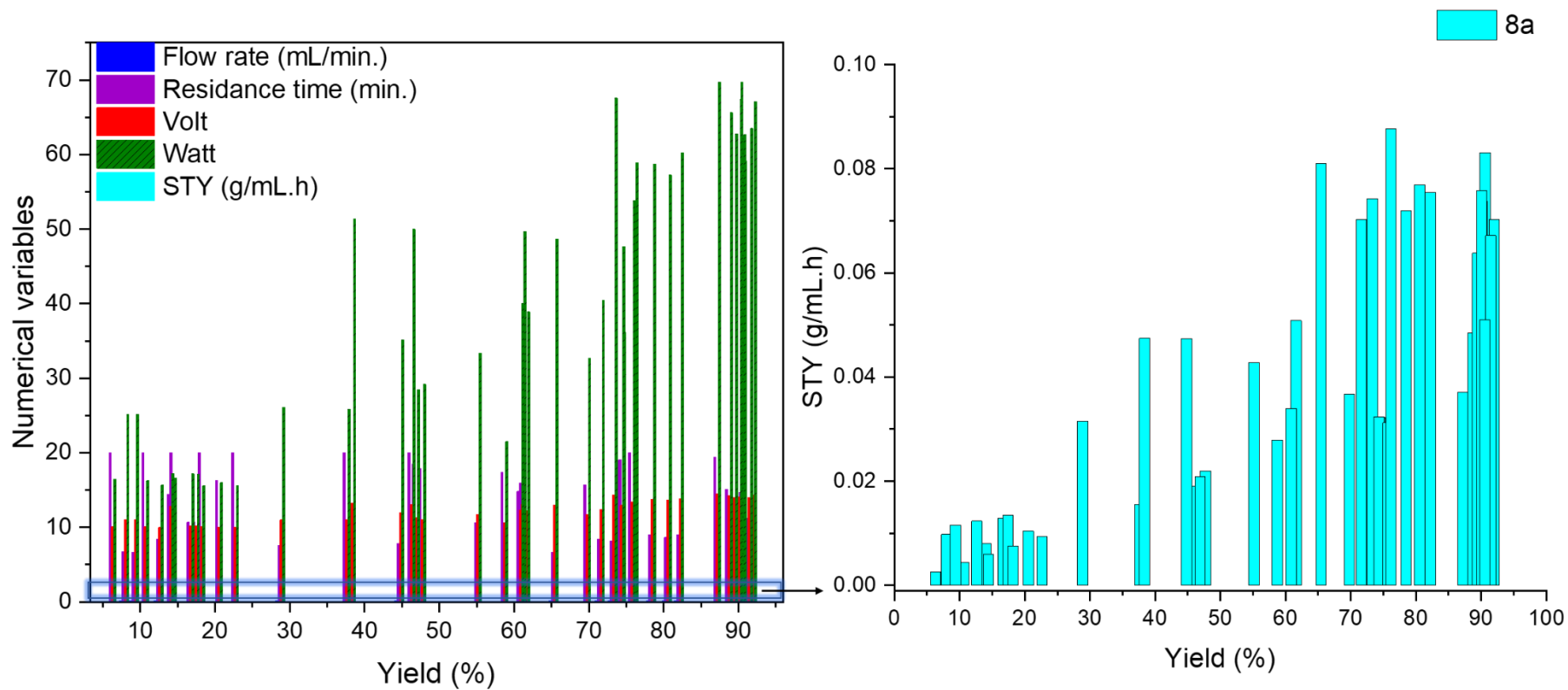


Figure S29. AI based system navigated multi-numerical variable complexity and identify the optimal conditions for the photo activated N-H insertion reaction. Compound **2a** (0.1 M in EtOAc); reagent **7a** (0.3 M in EtOAc); 1 mL reactor volume.

S5.3. General procedure of running AI optimized condition for longer time for synthesis of compound 8a.

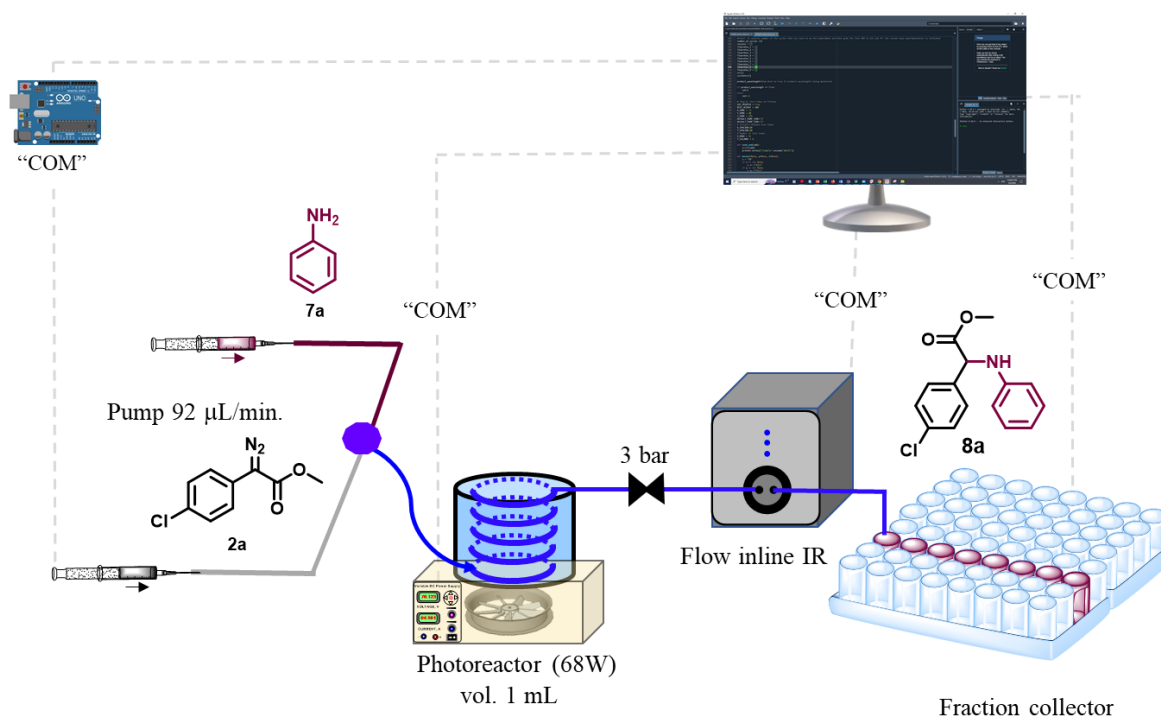
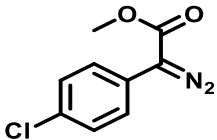

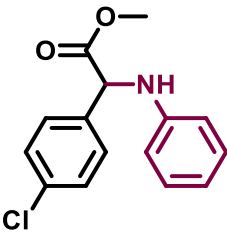


Figure S30. Schematic presentation of continuous flow for the synthesis, of compound **8a**.

To prepare the stock solution of compound **2a** (2.20 g, 0.01 mol) was dissolved in ethyl acetate (100 mL) charged in one syringe, another syringe charged with the reagent **7a** (2.9 mL, 0.03 mol) dissolved in ethyl acetate (100 mL). These two syringes connected via pump and out-put is further connected with the T₁-mixer and both syringes were running with the 46 $\mu\text{L}/\text{min}$. each flow rate to maintain the stoichiometry and then passed through PFA tubular reactor (id = 1000 μm , l = 1.3 m, vol. = 1 mL) under blue light (68.0 \pm 2 W) exposure. The room temperature of the reactor was controlled by fan attached to the bottom of the photochemical reactor. The out-put of the tubular reactor was connected with spring based back pressure regulator (\sim 3 bar) to maintain the evaporation. First one hour of the product mixture was discarded and next 5 h of the product mixture [28 mL; **2a** (0.294 g)] was collected in HPLC bottle The organic EtOAc phase was concentrated under reduced pressure and purified through the regular batch process protocols to give the product **8a** (0.35 g in 5 h, 92 %) as a colourless oil. **¹H NMR (400 MHz,**

CDCl₃) δ 7.43 – 7.40 (m, 2H), 7.30 – 7.27 (m, 2H), 7.11 – 7.07 (m, 2H), 6.71 – 6.67 (m, 1H), 6.52 – 6.49 (m, 2H), 5.03 (s, 1H), 5.00 (s, 1H), 3.69 (s, 3H). **¹³C NMR (126 MHz, CDCl₃)** δ 171.86, 145.67, 136.29, 134.17, 129.33, 129.10, 128.67, 118.37, 113.50, 60.12, 52.98. **IR (v_{max}):** 3418.02, 3020.09, 1740.08, 1214.36, 688.27 cm⁻¹. **HRMS (ESI):** *m/z* calcd for **C₁₅H₁₅ClNO₂** [M+H]⁺ 276.0786, found 276.0782. Verified the analytical data with those reported in the literature.⁶

Table S9. Comparative result for the carbene insertion into N-H bonds of amines.

Entry	Compound 2a	Reagent 5a	Product 6a	Comparative result
1				92%, 11 min. (our study) Blue light, chiral spiro phosphamides (2 mol%), 1 h 6 White light, 80 min., 83% 1

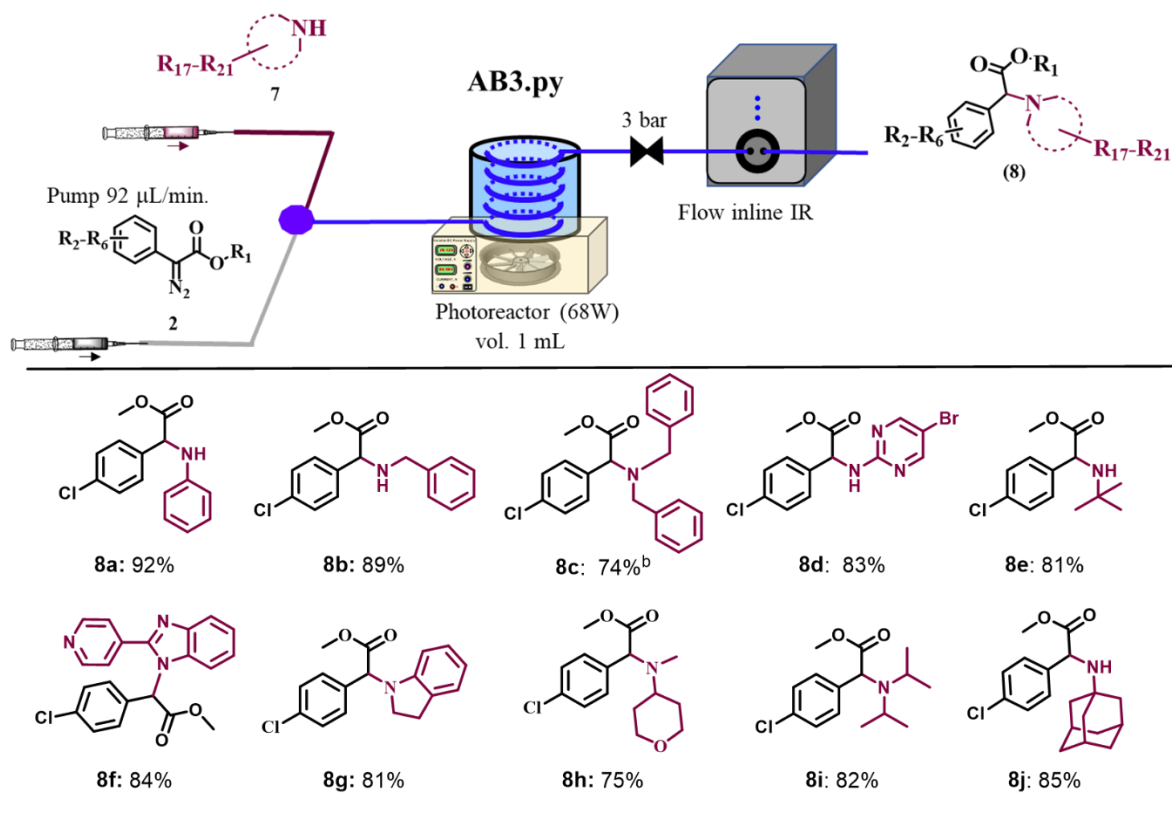
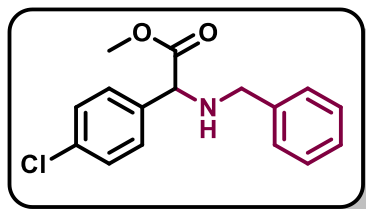


Figure S31. Scope of substrate for the photo activated N-H insertion reaction. Compound **2a** (0.1 M in EtOAc); reagent **7a** (0.3 M in EtOAc); 1 mL reactor volume.

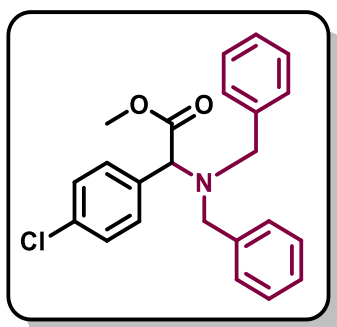
S5.3.1. General procedure for scope of substrate for carbene insertion into the N–H bonds of organic aromatic and aliphatic amines for synthesis of compound 8b–8j.

Example 26. Methyl 2-(benzylamino)-2-(4-chlorophenyl)acetate (**8b**).



The title compound was synthesized following the general procedure described in section 5.3 and involved corresponding reactant exchange with compound **2a** and compound **7b** (benzyl amine). The crude mixture was concentrated under vacuum, and the product was purified by flash chromatography; to give compound **8b** (0.74 g in 5 h, 89%) as a colorless oil. **¹H NMR (400 MHz, CDCl₃)** δ 7.34 – 7.22 (m, 9H), 4.36 (s, 1H), 3.70 (d, *J* = 6.2 Hz, 2H), 3.66 (s, 3H), 2.32 (s, 1H). **¹³C NMR (101 MHz, CDCl₃)** δ 173.02, 139.28, 136.61, 133.96, 129.05, 128.89, 128.51, 128.31, 127.27, 63.64, 52.36, 51.30. **IR (ν_{max}):** 3340.60, 2951.52, 1736.67, 1489.77, 1169.20, 769.20 cm⁻¹. **HRMS (ESI):** *m/z* calcd for C₁₆H₁₇ClNO₂ [M+H]⁺ 290.0948, found 290.0967.

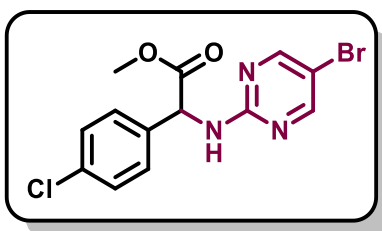
Example 27. Methyl 2-(4-chlorophenyl)-2-(dibenzylamino)acetate (**8c**).



The title compound was synthesized following the general procedure described in section 5.3 and involved corresponding reactant exchange with compounds **2a** and **7c** (dibenzylamine). The crude mixture was concentrated under vacuum, and the product was purified by flash

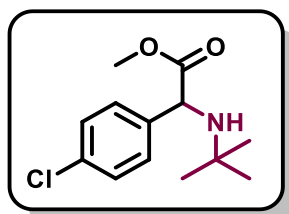
chromatography; to give compound **8c** (0.79 g in 5 h, 74%) as a colourless liquid. **¹H NMR (400 MHz, CDCl₃)** δ 7.34 – 7.21 (m, 14H), 4.57 (s, 1H), 3.77 (s, 3H), 3.73 (d, *J* = 2.9 Hz, 4H). **¹³C NMR (101 MHz, CDCl₃)** δ 172.28, 139.32, 135.32, 133.87, 130.26, 128.88, 128.69, 128.48, 127.28, 65.31, 54.35, 51.67. **IR (ν_{max}):** 3024.08, 1736.91, 1451.25, 1213.67, 688.73 cm⁻¹. **HRMS (ESI):** *m/z* calcd for C₂₃H₂₃ClNO₂ [M+H]⁺ 380.1412, found 380.1408.

Example 28. Methyl 2-((5-bromopyrimidin-2-yl)amino)-2-(4-chlorophenyl)acetate (**8d**).



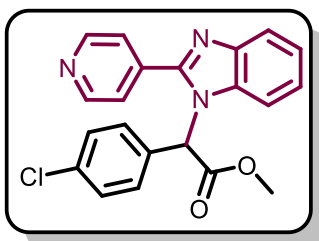
The title compound was synthesized following the general procedure described in section 5.3 and involved corresponding reactant exchange with compound **2a** and compound **7d** (5-bromopyrimidin-2-amine). The crude mixture was concentrated under vacuum, and the product was purified by flash chromatography; to give compound **8d** (0.83 g in 5 h, 83%) as a yellow liquid. **¹H NMR (500 MHz, CDCl₃)** δ 8.21 (s, 2H), 7.40 – 7.38 (m, 2H), 7.33 – 7.32 (m, 2H), 6.61 (s, 1H), 5.56 (d, *J* = 6.8 Hz, 1H), 3.74 (s, 3H). **¹³C NMR (101 MHz, CDCl₃)** δ 171.30, 159.18, 158.36, 135.38, 134.50, 129.11, 128.84, 107.86, 58.01, 52.93. **IR(ν_{max}):** 3251.65, 3001.76, 2952.73, 1742.79, 1577.46, 1429.20, 1170.80 cm⁻¹. **HRMS (ESI):** *m/z* calcd for C₁₃H₁₂BrClN₃O₂ [M+H]⁺ 355.9796, found 355.9792.

Example 29. Methyl 2-(*tert*-butylamino)-2-(4-chlorophenyl)acetate (**8e**).



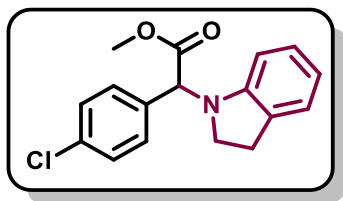
The title compound was synthesized following the general procedure described in section 4.8.2 and involved corresponding reactant exchange with compound **2a** and compound **7e** (*tert*-butyl amine). The crude mixture was concentrated under vacuum, and the product was purified by flash chromatography; to give compound **8e** (0.56 g in 5 h, 81%) as a colorless oil. **¹H NMR (500 MHz, CDCl₃)** δ 7.37 – 7.27 (m, 4H), 4.46 (s, 1H), 3.68 (s, 3H), 2.03 (s, 1H), 1.08 (s, 9H). **¹³C NMR (101 MHz, CDCl₃)** δ 175.11, 139.37, 133.50, 128.76, 59.08, 52.58, 51.36, 29.52. **IR (ν_{max}):** 3341.84, 2928.86, 1736.93, 1488.43, 1167.16, 823.38 cm⁻¹. **HRMS (ESI):** *m/z* calcd for C₁₃H₁₉ClNO₂ [M+H]⁺ 256.1099, found 256.1103.

Example 30. Methyl 2-(4-chlorophenyl)-2-(2-(pyridin-4-yl)-1H-benzo[d]imidazol-1-yl)acetate (**8f**).



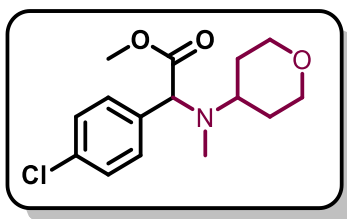
The title compound was synthesized following the general procedure described in section 5.3 and involved corresponding reactant exchange with compound **2a** and compound **7f** (2-(pyridin-4-yl)-1H-benzo[d]imidazole). The crude mixture was concentrated under vacuum, and the product was purified by flash chromatography; to give compound **8f** (0.92 g in 5 h, 84%) as an off white solid and **mp**: 166-168 °C. **¹H NMR (400 MHz, CDCl₃)** δ 8.77 (d, *J* = 6.0 Hz, 2H), 7.88 (d, *J* = 8.1 Hz, 1H), 7.56 – 7.55 (m, 2H), 7.37 – 7.33 (m, 3H), 7.25 – 7.23 (m, 1H), 7.15 – 7.09 (m, 3H), 6.37 (s, 1H), 3.76 (s, 3H). **¹³C NMR (101 MHz, CDCl₃)** δ 168.01, 151.50, 150.58, 143.19, 137.46, 135.06, 134.64, 131.44, 129.23, 128.53, 124.32, 123.55, 123.51, 120.76, 120.51, 112.40, 61.38, 53.44. **IR (ν_{max}):** 2958.15, 1747.03, 1411.75, 1212.68, 1004.48 cm⁻¹. **HRMS (ESI):** *m/z* calcd for C₂₁H₁₇ClN₃O₂ [M+H]⁺ 378.1007, found 378.1009.

Example 31. Methyl 2-(4-chlorophenyl)-2-(indolin-1-yl)acetate (**8g**).



The title compound was synthesized following the general procedure described in section 5.3 and involved corresponding reactant exchange with compound **2a** and compound **7g** (indoline). The crude mixture was concentrated under vacuum, and the product was purified by flash chromatography; $R_f = 0.5$ (30% ethylacetate/hexane); to give compound **8g** (0.70 g in 5 h, 81%) as a white solid and **mp**: 105-107 °C. $^1\text{H NMR}$ (400 MHz, CDCl_3) δ 7.39 (s, 4H), 7.13 – 7.06 (m, 2H), 6.76 – 6.72 (m, 1H), 6.47 (d, $J = 7.9$ Hz, 1H), 5.29 (s, 1H), 3.78 (s, 3H), 3.66 (dd, $J = 17.9, 8.6$ Hz, 1H), 3.23 – 3.17 (m, 1H), 3.05 – 2.89 (m, 2H). $^{13}\text{C NMR}$ (101 MHz, CDCl_3) δ 171.30, 150.63, 134.33, 133.76, 130.25, 130.03, 128.92, 127.24, 124.77, 118.64, 106.90, 63.21, 52.15, 49.90, 28.20. **IR** (ν_{max}): 2954.36, 1714.79, 1213.16, 1013.78, 838.25 cm^{-1} . $^1\text{HRMS}$ (ESI): m/z calcd for $\text{C}_{17}\text{H}_{17}\text{ClNO}_2$ $[\text{M}+\text{H}]^+$ 302.0942, found 302.0970.

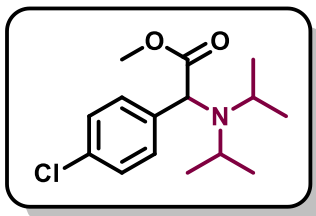
Example 32. Methyl 2-(4-chlorophenyl)-2-(methyl(tetrahydro-2H-pyran-4-yl)amino)acetate (**8h**).



The title compound was synthesized following the general procedure described in section 5.3 and involved corresponding reactant exchange with compound **2a** and compound **7h** (N-methyltetrahydro-2H-pyran-4-amine). The crude mixture was concentrated under vacuum, and the product was purified by flash chromatography; to give compound **8h** (0.62 g in 5 h, 75%) as a colourless liquid. $^1\text{H NMR}$ (400 MHz, CDCl_3) δ 7.38 – 7.36 (m, 2H), 7.33 – 7.30 (m, 2H), 4.47 (s, 1H), 4.00 – 3.95 (m, 2H), 3.70 (s, 3H), 3.32 – 3.21 (m, 2H), 2.74 – 2.66 (m, 1H), 2.24 (s, 3H), 1.72 – 1.59 (m, 4H). $^{13}\text{C NMR}$ (101 MHz, CDCl_3) 172.32, 135.51, 133.93,

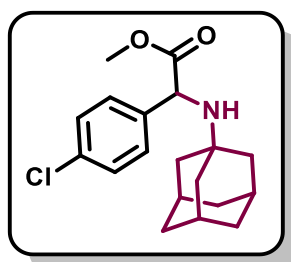
129.92, 128.70, 68.63, 67.57, 67.36, 56.94, 51.93, 33.42, 29.69, 28.64. **IR** (ν_{\max}): 2952.50, 2846.49, 1738.35, 1489.25, 1206.43, 1011.50, 816.53 cm^{-1} . **HRMS (ESI)**: m/z calcd for $\text{C}_{15}\text{H}_{21}\text{ClNO}_3$ $[\text{M}+\text{H}]^+$ 298.1210, found 298.1230.

Example 33. Methyl 2-(4-chlorophenyl)-2-(diisopropylamino)acetate (**8i**).



The title compound was synthesized following the general procedure described in section 5.3 and involved corresponding reactant exchange with compounds **2a** and **7i** (diisopropylamine). The crude mixture was concentrated under vacuum, and the product was purified by flash chromatography; to give compound **8i** (0.66 g in 5 h, 82%) as a white solid. **$^1\text{H NMR}$ (400 MHz, CDCl_3)** δ 7.29 – 7.22 (m, 4H), 4.73 (s, 1H), 3.73 (s, 3H), 3.30 – 3.20 (m, 2H), 1.10 (d, $J = 6.6$ Hz, 6H), 0.96 (d, $J = 6.7$ Hz, 6H). **$^{13}\text{C NMR}$ (126 MHz, CDCl_3)** δ 175.62, 138.16, 133.05, 129.78, 128.33, 61.04, 51.69, 45.95, 23.24, 21.55. **IR** (ν_{\max}): 2963.52, 2871.93, 1740.85, 1459.80, 1140.85, 1015.15, 821.93 cm^{-1} . **HRMS (ESI)**: m/z calcd for $\text{C}_{15}\text{H}_{23}\text{ClNO}_2$ $[\text{M}+\text{H}]^+$ 284.1412, found 284.1411.

Example 34. Methyl 2-(((3s,5s,7s)-adamantan-1-yl)amino)-2-(4-chlorophenyl)acetate (**8j**).



The title compound was synthesized following the general procedure described in section 5.3 and involved corresponding reactant exchange with compound **2a** and compound **7j** (adamantly amine). The crude mixture was concentrated under vacuum, and the product was purified by flash chromatography; to give compound **8j** (0.76 g in 5 h, 85%) as a white solid.

¹H NMR (400 MHz, CDCl₃) δ 7.37 – 7.26 (m, 4H), 4.57 (s, 1H), 3.67 (s, 3H), 2.04 (d, *J* = 7.0 Hz, 4H), 1.64 – 1.55 (m, 12H). **¹³C NMR (101 MHz, CDCl₃)** δ 175.05, 139.49, 133.33, 128.68, 128.63, 56.90, 52.45, 51.31, 43.07, 36.51, 29.56. **IR (ν_{max}):** 3328.25, 2904.43, 2848.81, 1736.79, 1487.26, 1163.15, 821.61 cm⁻¹. **HRMS (ESI):** *m/z* calcd for C₁₉H₂₅ClNO₂ [M+H]⁺ 334.1574, found 334.1592.

S6 Case study-4: Carbene insertion (C-C bond formation, cyclopropanation) single objective multi-variant auto-optimization.

5.6.1 Background collection

Sample Preparation: We prepare 0.5 M stock solution of compound **2a** in DCE, 0.5 M stock solution of reagent **9a** in DCE and 0.5 M stock solution of product **10a** in DCE. The three of the solution taken in three different syringes. To conduct our analysis, we utilized an In-line FTIR system. The experimental protocol involved the initial introduction of the solvent, DCE, for a duration of 10 min. using a syringe pump, during which data was recorded. Subsequently, a previously prepared stock solution of compound **2a** was similarly introduced into the In-line FTIR for 10 min. This process was repeated for the reaction product **10a** and reagent **9a**, each pumped for 10 min. Upon collecting all relevant data, we identified the signature peak present in the product. Using this peak as a reference, we employed a Bayesian optimizer to fine-tune and optimize the reaction conditions for improved results.

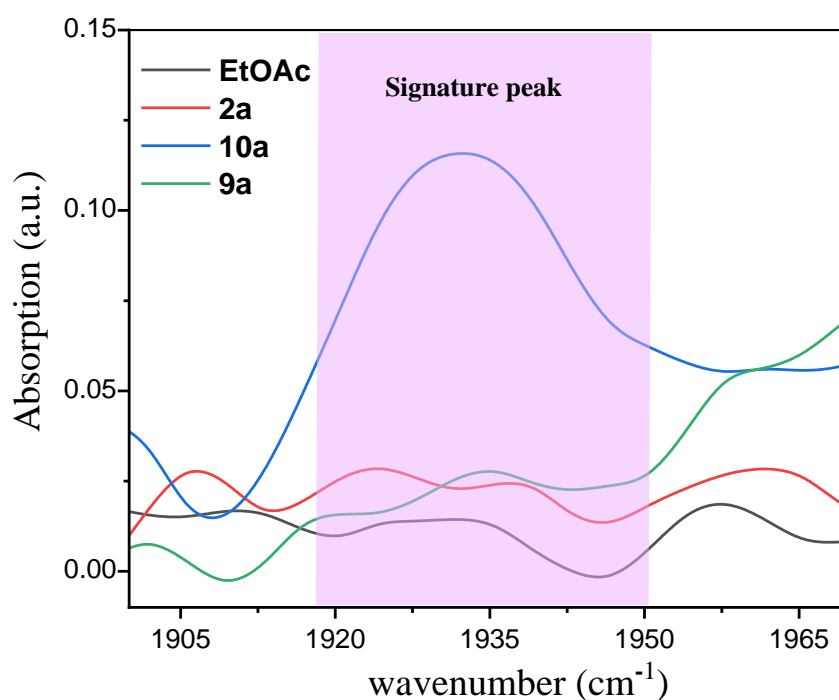


Figure S32. In-line IR background analysis of DCE, 0.5 M stock solution of **2a**, **9a**, and **10a** in DCE.

The distinctive shifting peak associated with product (**10a**) aligning within the range of 1910 to 1950 cm^{-1} was observed. This specific peak, falling within this range, is designated as the signature peak for our analysis. During the optimization of the reaction through Bayesian methods, this signature peak serves as a pivotal reference point. We utilize it to calculate the area under the curve in the In-line FT-IR spectra, providing a quantitative measure of the area in the forthcoming reaction mixture.

S6.2. General procedure for the auto-optimized synthesis of compound 10a.

Initial steps involved the preparation of stock solutions for compound **2a** (0.1 M in DCE) and reagent **9a** (0.2 M in DCE), each filled into separate syringes and connected to a syringe pump. The solutions were then directed through a PFA tubular reactor (inner diameter = 1 mm, length = 5.0 m, volume = 4 mL) surrounded by a cylindrical-shaped blue LED light source. Once the solution setup was complete, input ranges for variable flow rates, voltages, and current were specified in the Python code designed for the reaction. The Bayesian optimizer systematically explored these varying reaction conditions, aiming to achieve maximum yield. The optimization process involved running 50-60 experiments and the results were tabulated in the optimization table (Table S10).

S6.2.1. Python code for optimizing condition for C-C bond formation (cyclopropanation).

```
ab4.py
1 from os import listdir
2 from os.path import isfile, join
3 import serial
4
5 import numpy as np
6 import pandas as pd
7 import time
8 from scipy.integrate import trapz
9
10 #step1: be sure to the address of the files that the ftir data is exported is matching to line 11 (mypath)
11 mypath = "C:\\Users\\Admin\\Desktop\\ruchi\\Exp 2023-10-31 18-27"
12 onlyfiles = [f for f in listdir(mypath) if isfile(join(mypath, f))]
13
14
15 #step2: make sure that pump and the potentiostat is correctly addressed in the Line 16 and 17
16 pump_1 = serial.Serial("COM4",9600) #Harvard Pump
17 port = serial.Serial("COM1",115200)
18 printer = serial.Serial("COM6", 115200, timeout=1)
19
20
21
22 #step 3: grab the lines from 22 to 90 and press f9
23 def area_under(data,start,end):
24     x = np.flip(data.iloc[start:end,0].to_numpy())
25     y = np.flip(data.iloc[start:end,1].to_numpy())
26     area = trapz(y,x)
27     return np.abs(area)
28
29 def file_namer(num):
30     str1 = str(num)
31     length = int(len((str1)))
32     empt = ''
33     for i in range(5-length):
34         empt = empt+'0'
35
36     return empt+str1
37
38
39 def ftir_extract(filename,init,end):
40     filename = filename
41
42     temp_df = pd.read_csv(filename)
43     # nump_df = temp_df.to_numpy()
44     area = area_under(temp_df, init, end)
45     # max_peak = np.max(nump_df[90:120,1])
46     print(area)
47
```

```

48     return area
49
50
51 def function(flowrate_1,v,i):
52     #set the pumps with the flowrate as the desired flowrate for the function
53
54     fr_1 = flowrate_1 #ml/min
55     pump_1.write(('irate '+str(fr_1)+' ml/min\r\n').encode())
56
57     time.sleep(0.1)
58
59     #set the com port for potentiostat and set the voltage and current
60     vol = v
61     curr = i
62     port.write(('VOLT '+str(vol)+'\r\n').encode()) #to change the voltge we need to use "VOLT 1" command
63     port.write(('CURR '+str(curr)+'\r\n').encode()) #to change the current we need to use "CURR 1" command
64
65
66
67     #pumps run
68     pump_1.write((b'irun\r\n'))
69     time.sleep(0.1)
70     time.sleep(4800)
71
72 def function2(flowrate_1,v,i):
73     time.sleep(180)
74
75     files = [f for f in listdir(mypath) if isfile(join(mypath, f))]
76
77     val = 0
78
79     #change wavelenghts as per product here.
80     f_row=9 #first row of range for wavelength as per IR CSV
81     l_row=19 #Last row of range for wavelength as per IR CSV
82
83     val += ftir_extract(files[-1],f_row,l_row)
84     val+=ftir_extract(files[-2],f_row,l_row)
85     val+=ftir_extract(files[-3],f_row,l_row)
86
87     avg_val = val/3
88
89     return avg_val
90
91
92 #step 4:grab the line 93 and f9
93 from skopt.optimizer import Optimizer
94

```

```

95
96 #step5:in line 96 we have to define the range that (flowrate,voltage,current) (from,to) and after (anytime) applying changes you need to grab the line 96 and f:
97 #flowrates are in ml/min, voltage in V, Current in Ampere
98 bounds = [(0.05,0.15),(10,14.5),(4.9,5.0)]
99
100
101 #step 6: grab the line 100 and f9
102 opter =Optimizer(bounds,base_estimator='gp',n_initial_points=3,acq_func="EI",random_state=np.random.randint(3326))
103
104
105 #step7: to selecte number of the cycles that you have to do the experiment and then grab the line 104 to 121 and f9: the closed loop experimentation is initiate
106 number_of_cycles = 22
107 results = []
108 flowrates_1 = []
109 vs = []
110 currents = []
111
112 product_wavelength=True #set to true if product wavelengths being monitored
113
114 if product_wavelength == True:
115     val=1
116 else:
117     val=-1
118
119
120 # Step 8: Test Tubes on Printer
121 USE_PRINTER = True
122 REST_HEIGHT = 200
123 X_HOME = -5
124 Y_HOME = 20
125 Z_HOME = 175
126 DEFAULT_PUMP_TIME="1"
127 # Distance between test tubes
128 X_SPACING=20
129 Y_SPACING=20
130 # Number of test tubes
131 X_ROWS = 11
132 Y_COLUMNS = 4
133
134 def send_cmd(cmd):
135     print(cmd)
136     printer.write(f"{cmd}\n".encode("ASCII"))
137
138 def move(x=None, y=None, z=None):

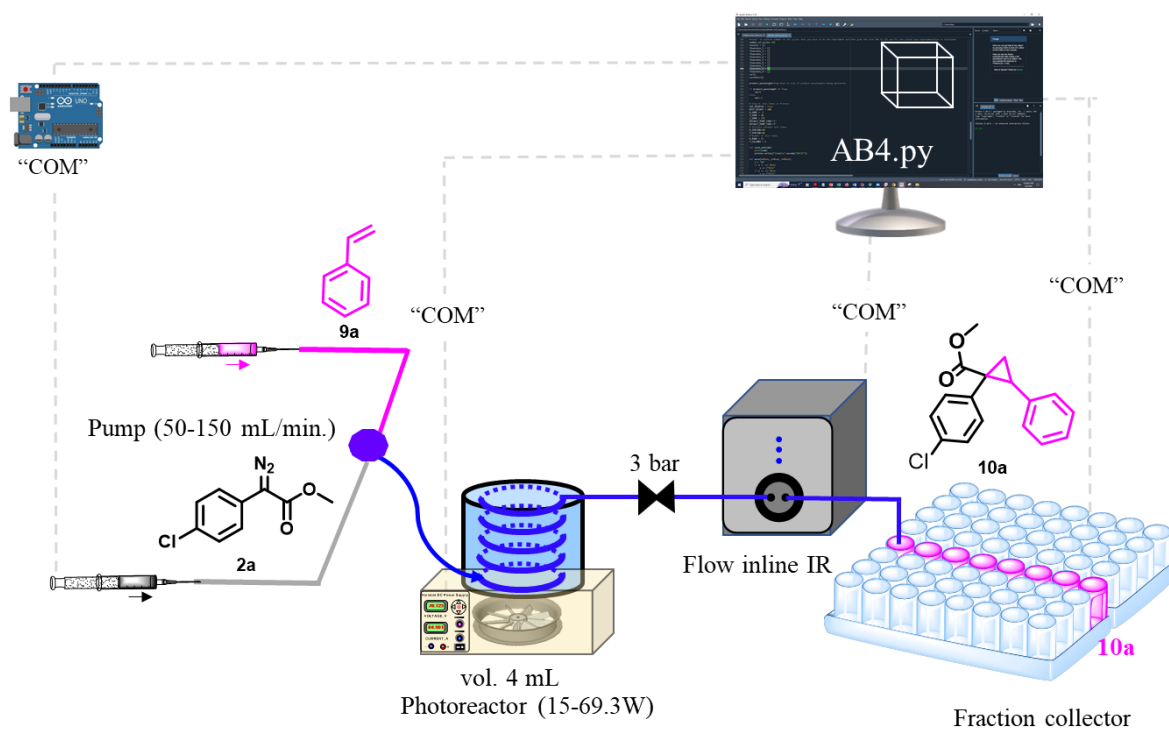
```

```

139 s = "G0"
140 if x is not None:
141     s += f"X{x}"
142 if y is not None:
143     s += f"Y{y}"
144 if z is not None:
145     s += f"Z{z}"
146
147 s+= "F5000"
148 send_cmd(s)
149
150 def printer_positions():
151     for j in range(Y_COLUMNS):
152         for i in range(X_ROWS):
153             if j%2==1:
154                 yield (X_HOME + (X_ROWS - 1 - i) * X_SPACING, Y_HOME + j * Y_SPACING, Z_HOME)
155             else:
156                 yield (X_HOME + i * X_SPACING, Y_HOME + j * Y_SPACING, Z_HOME)
157
158 # Run this.
159 tube_location = list(printer_positions())
160
161
162 for i in range(number_of_cycles):
163     move(*tube_location[2*i])
164     asked = opter.ask()
165     function(asked[0],asked[1],asked[2])
166     move(*tube_location[2*i+1])
167     told = function2(asked[0],asked[1],asked[2])
168
169     print(f"area under the curve in the round {i:.2f} = {told: .2f}")
170     opter.tell(asked,-told*val)
171     results.append(told)
172     flowrates_1.append(asked[0])
173     vs.append(asked[1])
174     currents.append(asked[2])
175
176     dict1 = {"flowrate_1":flowrates_1,"voltages":vs, "currents":currents,"area-results":results}
177     df2 = pd.DataFrame(dict1)
178     df2.to_csv("output round "+str(i)+".csv")
179
180
181
182 pump_1.write(b'stop\r\n')
183
184
185

```

Table S10. Auto-optimization table of C-C bond formation (cyclopropanation).



No. of experiments	Flow rate mL/min.	Voltage (V)	Light intensity (W)	area- results	Yield (%)	Space time yield (g mL ⁻¹ h ⁻¹)
1	0.05238	11.3274	27.907	0.221759	15.04351243	0.001690215
2	0.11695	12.0946	36.33	0.883698	59.94760912	0.015038322
3	0.0759	14.4634	67.166	0.965877	65.52240341	0.010667408
4	0.13854	14	60.298	1.178701	79.95979035	0.023761515
5	0.05488	12.406	40.357	0.586033	39.75484522	0.004679845
6	0.10281	14.484	67.22	1.280536	86.8679929	0.019156777
7	0.09679	14.0054	60.698	1.104894	74.9529292	0.015561319
8	0.05326	14.1323	62.576	0.860929	58.40302362	0.006672119
9	0.10344	14.0963	62.093	1.268002	86.01772127	0.019085509
10	0.06814	14.4914	67.943	1.050196	71.24236934	0.010412806
11	0.09225	14.4998	67.938	1.119739	75.95997262	0.015030675
12	0.05962	14.5	67.889	1.076398	73.01983999	0.009338135
13	0.1326	14.2594	64.379	1.344057	91.17708048	0.025933223
14	0.09069	10.0003	15.612	0.104695	7.102216975	0.001381595

15	0.08657	14.5	64.141	0.86797	58.88066544	0.010933707
16	0.09243	12.5787	42.061	0.566808	38.45067481	0.007623321
17	0.06643	14.5	58.947	0.863185	58.55606437	0.008343791
18	0.14817	14.0122	59.401	0.97105	65.87332531	0.020936167
19	0.05161	14.0002	60.993	0.980286	66.49986981	0.007361765
20	0.07201	14.0001	60.965	0.938318	63.65287767	0.009831916
21	0.14721	14.4998	60.856	1.198318	81.29055295	0.025668748
22	0.1287	13.7505	57.251	1.108318	75.18520381	0.020755574
23	0.14672	14.4002	62.576	1.206441	81.84159463	0.025756728
24	0.11723	14.5	68.44	1.244494	84.42300408	0.021228869
25	0.06946	12.406	40.357	0.32261	21.88496316	0.003260678
26	0.15	10	15.58	0.069309	4.701729369	0.001512781
27	0.05	10	15.58	0.294156	19.95472311	0.002140144
28	0.05058	14.1178	62.919	0.412608	27.99017662	0.003036769
29	0.128	14.5	68.353	1.37646	93.37520968	0.025637098
30	0.09281	14.5	68.251	1.167093	79.17233598	0.015761427
31	0.05643	14.5	68.179	0.679223	46.07659507	0.005577219
32	0.05563	10.002	15.602	0.132377	8.980086695	0.001071561
33	0.14537	10.0003	15.612	0.289655	19.64938782	0.006127046
34	0.14994	13.0001	47.886	0.798213	54.14854499	0.017415325
35	0.14993	10.0356	15.895	0.277093	18.79721676	0.006045182
36	0.15	14.5	67.947	0.979234	66.42850506	0.021373372
37	0.14897	10.7891	22.869	0.293017	19.87745653	0.006351655
38	0.14945	11.2435	27.424	0.330208	22.4003903	0.007180899
39	0.05023	10.7667	22.585	0.504851	34.24768463	0.00368996
40	0.09691	13.0001	47.886	0.921254	62.49530347	0.012991021
41	0.05134	12.3099	39.229	0.813469	55.18346951	0.006077041
42	0.14539	14.1487	62.167	1.265278	85.83293271	0.026767991
43	0.05	14.5	67.831	0.983382	66.70989382	0.007154636
44	0.05	13.1051	41.708	0.871355	59.1102944	0.006339579
45	0.05	14.5	67.802	1.223662	83.00981927	0.008902803
46	0.11106	14.5	67.744	1.27144	86.25094561	0.020547019
47	0.15	12.5787	42.061	0.927784	62.93828047	0.020250392

48	0.15	13.1051	48.708	0.957731	64.96980148	0.020904034
49	0.1251	14.5	69.301	1.406376	95.40462773	0.02560083
50	0.15	13.3803	52.251	1.129606	76.6293224	0.024655484
51	0.15	11.126	26.133	0.46734	31.70304294	0.010200454
52	0.15	14.5	67.947	1.313304	89.09088268	0.028664992
53	0.13	13.3538	52.209	1.04556	70.92787602	0.019778238
54	0.06674	14.4746	67.719	1.186773	80.50737233	0.011525218
55	0.15	12.6835	43.512	0.99783	67.69000587	0.021779259
56	0.15	13.3591	55.044	1.033796	70.12983905	0.022564276
57	0.14801	10.0012	15.5	0.037879	2.569605777	0.000815802
58	0.09802	12.9662	46.924	0.988208	67.03727621	0.014094782
59	0.14937	14.4998	60.391	1.096345	74.37298887	0.023829005
60	0.14409	14.1585	61.436	1.12544	76.34671257	0.023596711

Reaction condition: compound **2a** (0.1 M in DCE); reagent **9a** (0.2 M in DCE); 4 mL reactor volume.

Graph:

We have plotted 3D graph of optimization table S10, we have taken flow rate on X-axis, blue light intensity (watt) on Y-axis and product yield on Z axis.

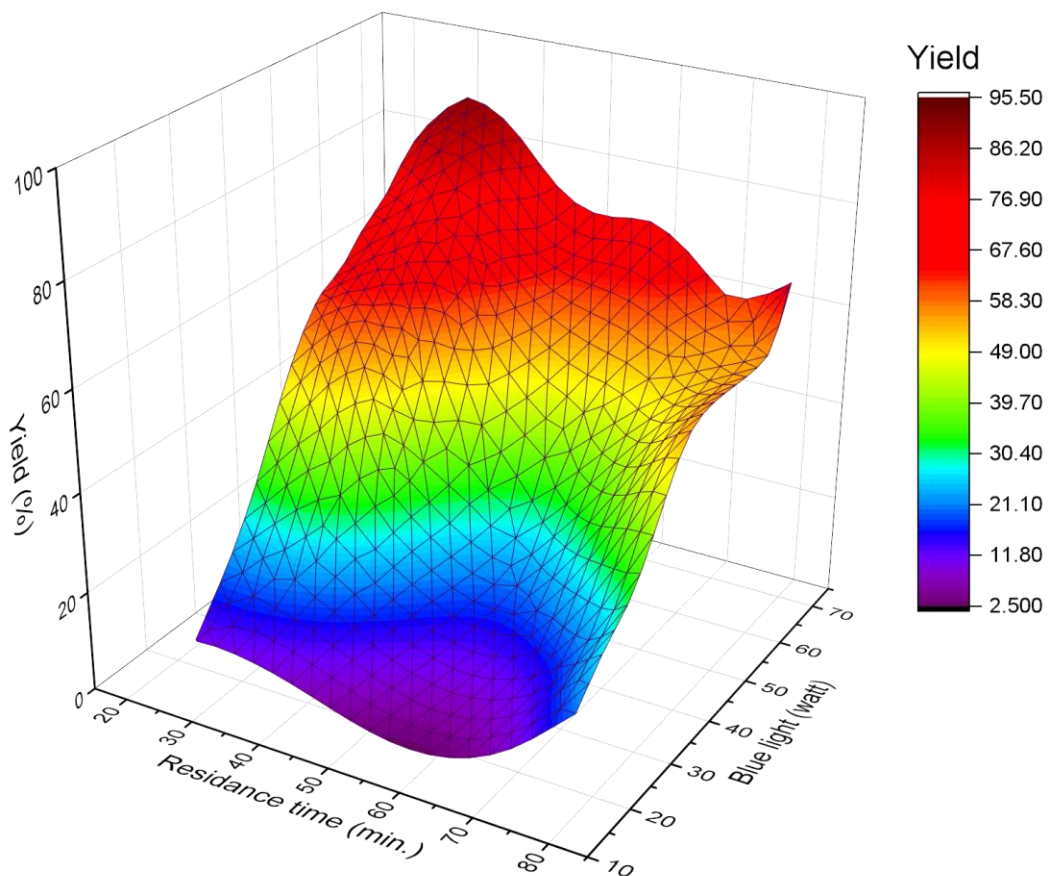


Figure S33. AI based system to auto-optimize and navigate this complexity and identify the optimal conditions for the photo activated cyclopropanation reaction. Compound **2a** (0.1 M in DCE); reagent **9a** (0.2 M in DCE); 4 mL reactor volume.

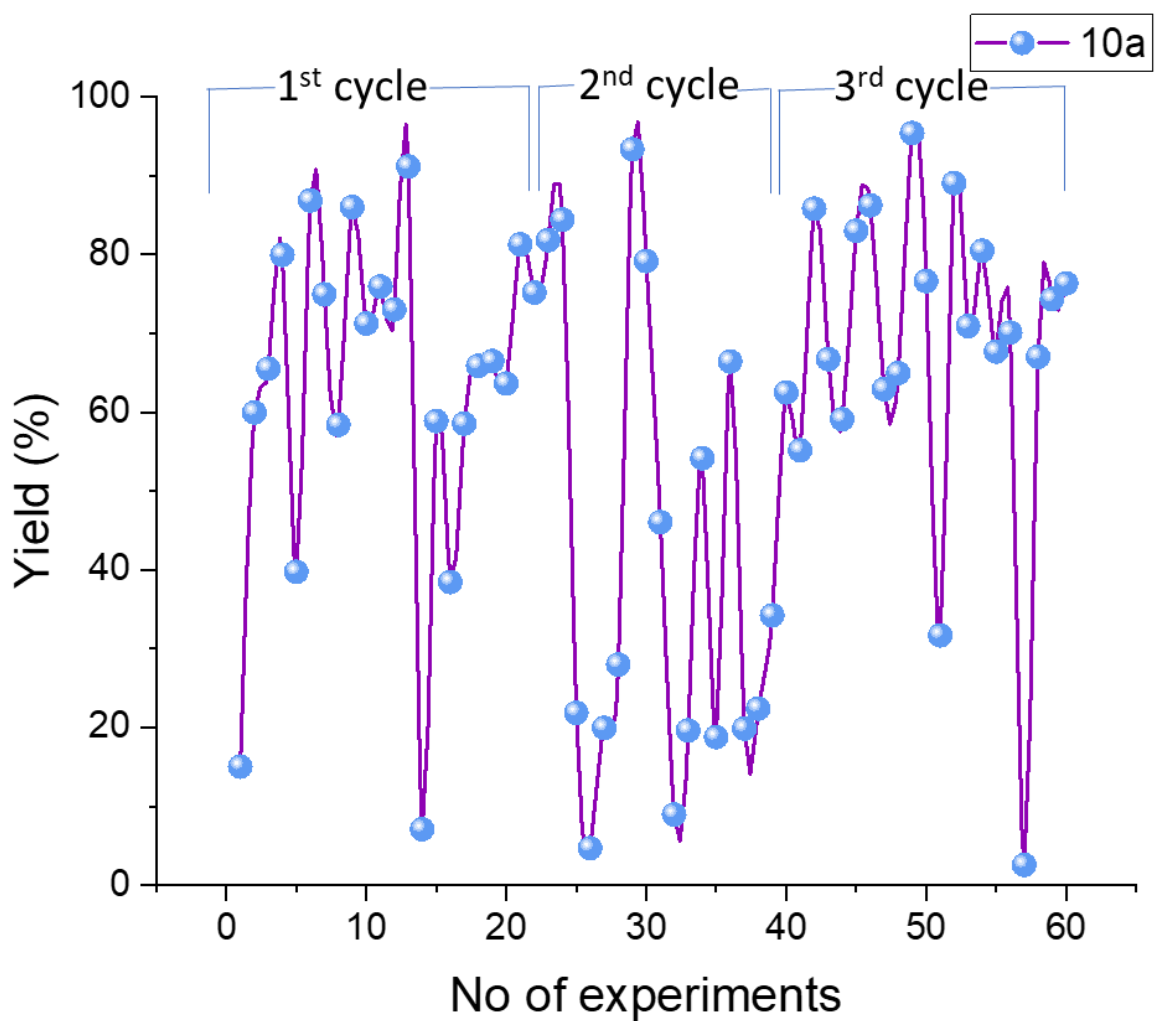


Figure S34. 2D graph between no of experiments performed versus yield (%) for the AI based auto-optimization of photo activated cyclopropanation reaction. Compound **2a** (0.1 M in DCE); reagent **9a** (0.2 M in DCE); 4 mL reactor volume.

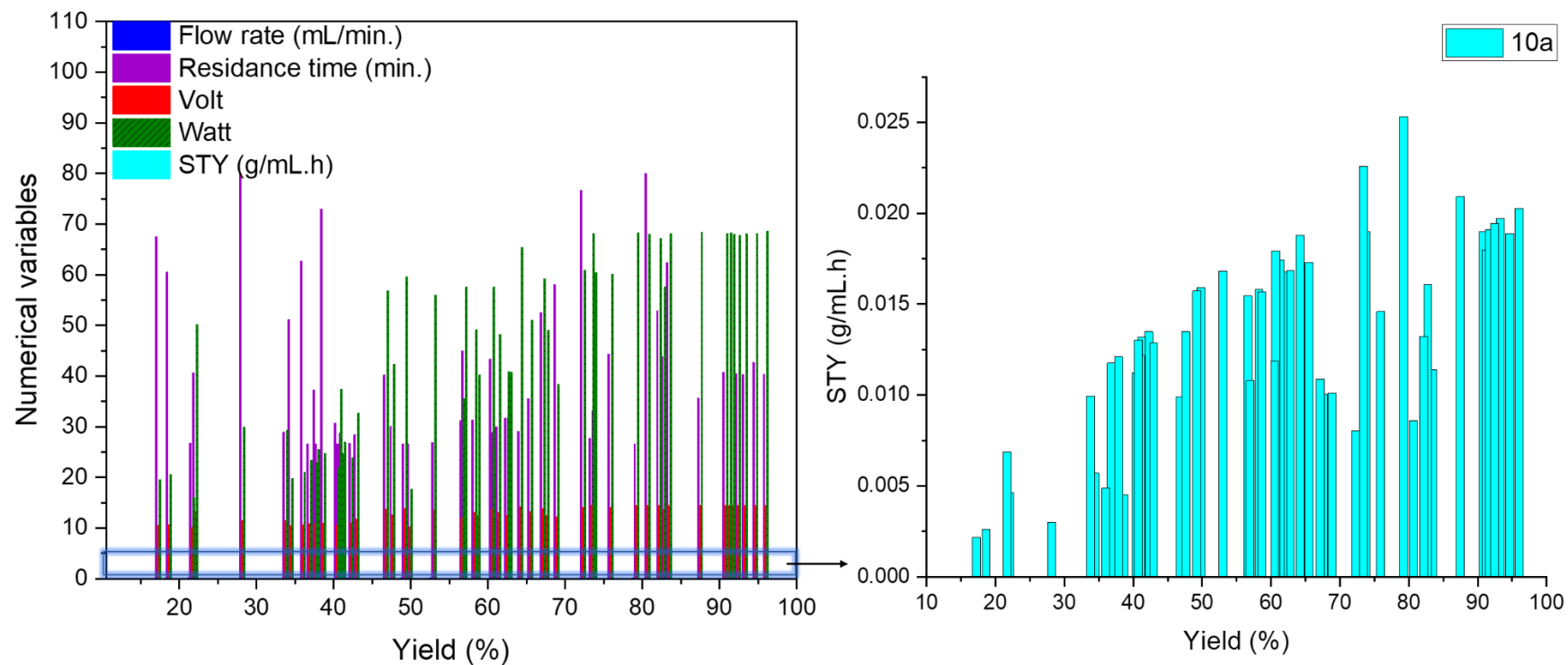


Figure S35. AI based system navigated multi-numerical variable complexity and identify the optimal condition for the photo activated cyclopropanation reaction. Compound **2a** (0.1 M in DCE); reagent **9a** (0.2 M in DCE); 4 mL reactor volume.

S6.3. General procedure for AI optimized carbene insertion into the double bonds of aromatic alkenes for longer run to synthesis compound **10a**.

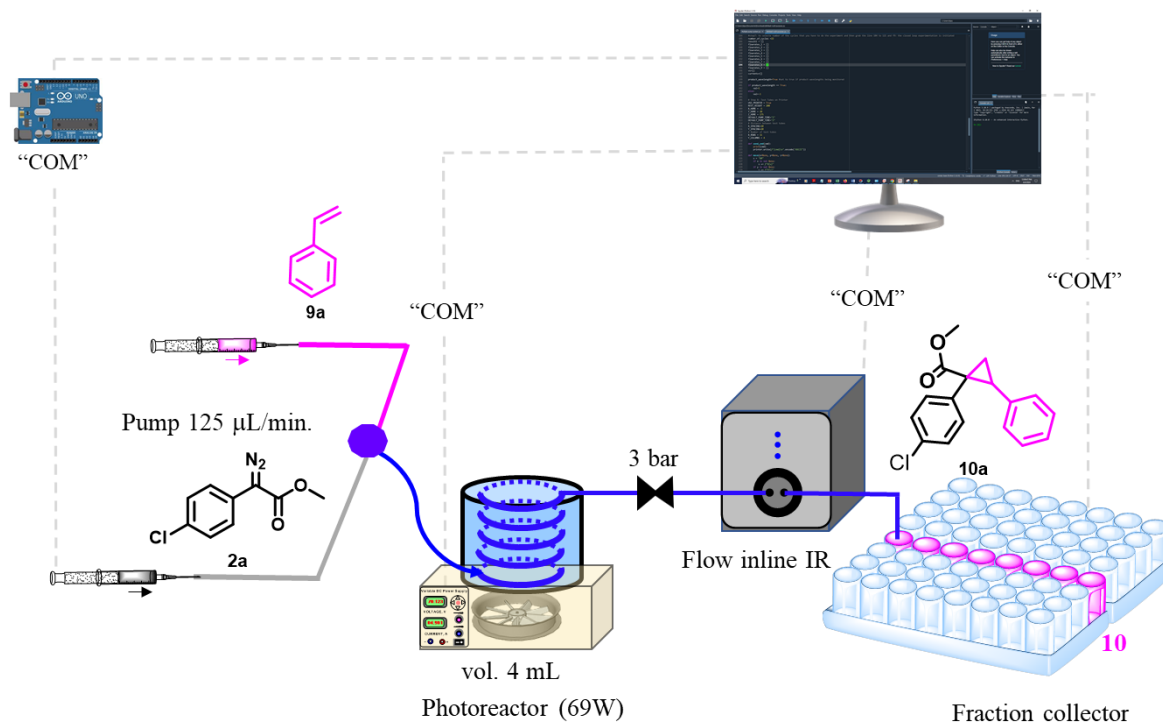


Figure S36. Schematic presentation of continuous flow for the synthesis, of compound **9a**.

To prepare the stock solution of compound **2a** (2.20 g, 0.01 mol) was dissolved in ethyl acetate (100 mL) charged in one syringe, another syringe charged with the reagent **9a** (2.4 mL, 0.02 mol) dissolved in ethyl acetate (100 mL). These two syringes connected via pump and out-put is further connected with the T-mixer and both syringes were running with the 125 µL/min. flow rate to maintain the stoichiometry and then passed through PFA tubular reactor (id = 1000 µm, l = 5 m, vol. = 4 mL) under blue light (69W) exposure. The room temperature of the reactor was controlled by fan attached to the bottom of the photochemical reactor. The out-put of the tubular reactor was connected with spring based back pressure regulator (~3 bar) to maintain the evaporation. First one hour of the product mixture was discarded and next 5 h of the product mixture [37.5 mL; **2a** (0.393 g)] was collected in HPLC bottle. The organic EtOAc phase was concentrated under reduced pressure to give the product **10a** (0.507 g in 5 h, 95%) as a colorless oil. $^1\text{H NMR}$ (400 MHz, CDCl_3) δ 7.12 – 7.07 (m, 5H), 6.97 – 6.93 (m, 2H), 6.79 – 6.77 (m,

2H), 3.67 (s, 3H), 3.11 (dd, $J = 9.3, 7.4$ Hz, 1H), 2.14 (dd, $J = 9.3, 5.0$ Hz, 1H), 1.84 (dd, $J = 7.3, 5.0$ Hz, 1H). **^{13}C NMR (101 MHz, CDCl_3)** δ 173.99, 135.99, 133.56, 133.33, 133.07, 131.63, 129.46, 128.15, 128.07, 128.02, 126.68, 52.80, 36.82, 33.31, 20.47. **IR (ν_{max}):** 3024.29, 2953.82, 1716.75, 1494.38, 1213.56, 825.68 cm^{-1} . **HRMS (ESI):** m/z calcd for $\text{C}_{17}\text{H}_{16}\text{ClO}_2$ $[\text{M}+\text{H}]^+$ 286.0833, found 286.0842. Verified the analytical data with those reported in the literature.⁷

Space time yield in previous reported batch process⁸

Productivity = 100.4 mg in 12 h

$$\text{Productivity (per h)} := \frac{0.100}{12} = 0.008 \text{ g/h}$$

$$\text{Space time yield} := \frac{\text{productivity}}{\text{volume of reactor}}$$

$$= \frac{0.008}{3} = 0.0026 \text{ g mL}^{-1} \text{ h}^{-1}$$

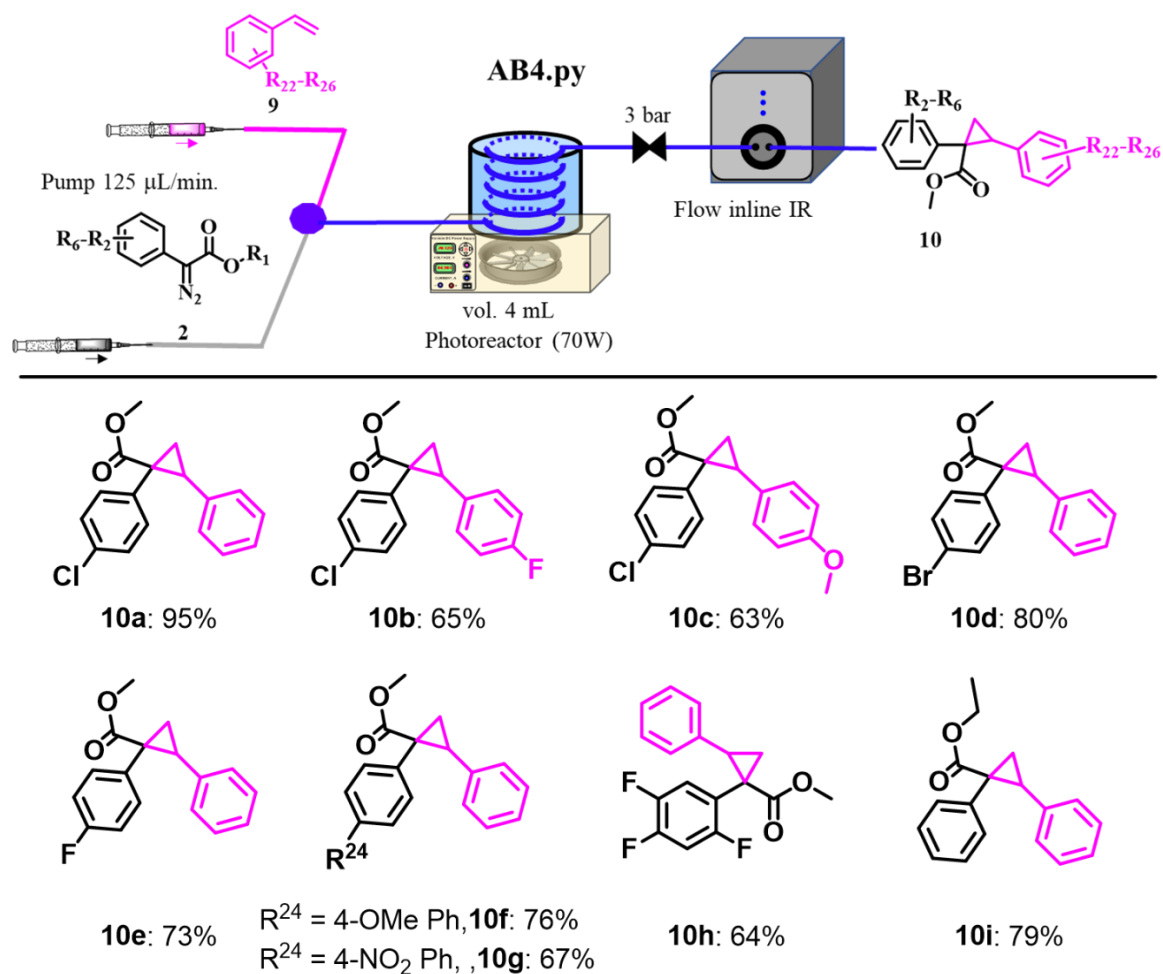
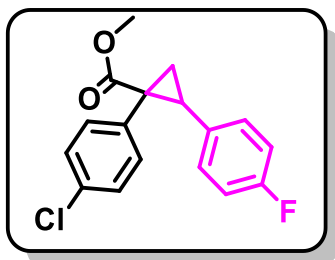


Figure S37. Scope of substrate for the photo activated cyclopropanation reaction. Compound **2a** (0.1 M in DCE); reagent **9a** (0.2 M in DCE); 4 mL reactor volume.

S6.3.1. General procedure for scope of substrate for synthesis of compound **10b–10i**.

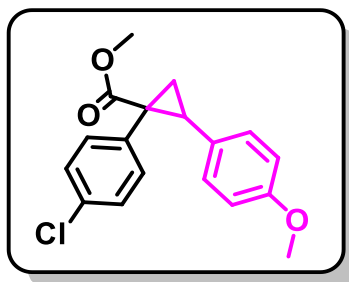
Example 36. Methyl 1-(4-chlorophenyl)-2-(4-fluorophenyl)cyclopropane-1-carboxylate (**10b**).



The title compound was synthesised following the general procedure described in section 6.3, and involve corresponding reactant exchange with compound **2a** and compound **9b** (1-fluoro-

4-vinylbenzene). The crude mixture was concentrated under vacuum and the product was purified by flash chromatography; $R_f = 0.5$ (5% ethylacetate/hexane); to give compound **10b** (0.37 g in 5 h, 65%) as a colorless oil. $^1\text{H NMR}$ (400 MHz, CDCl_3) δ 7.13–7.10 (m, 2H), 6.96–6.93 (m, 2H), 6.81–6.71 (m, 4H), 3.66 (s, 3H), 3.09 (dd, $J = 9.3, 7.3$ Hz, 1H), 2.13 (dd, $J = 9.4, 5.1$ Hz, 1H), 1.78 (dd, $J = 7.3, 5.1$ Hz, 1H). $^{13}\text{C NMR}$ (101 MHz, CDCl_3) δ 173.83, 162.92, 160.48, 133.34, 133.28, 133.20, 131.75, 129.57, 129.49, 128.17, 115.07, 114.85, 52.81, 36.65, 32.50, 20.55. $^{19}\text{F NMR}$ (471 MHz, CDCl_3) δ -114.38 (s). IR (ν_{max}): 3021.76, 1715.56, 1437.27, 1214.76, 836.50 cm^{-1} . HRMS (ESI): m/z calcd for $\text{C}_{17}\text{H}_{15}\text{ClFO}_2$ $[\text{M}+\text{H}]^+$ 305.0739, found 305.0748.

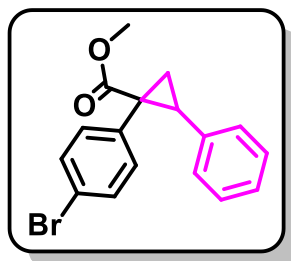
Example 37. Methyl 1-(4-chlorophenyl)-2-(4-methoxyphenyl)cyclopropane-1-carboxylate (**10c**).



The title compound was synthesised following the general procedure described in section 6.3, and involve corresponding reactant exchange with compound **2a** and compound **9c** (1-methoxy-4-vinylbenzene). The crude mixture was concentrated under vacuum and the product was purified by flash chromatography; $R_f = 0.5$ (5% ethylacetate/hexane); to give compound **10c** (0.375 g in 5 h, 63%) as a colorless oil. $^1\text{H NMR}$ (400 MHz, CDCl_3) δ 7.11–7.08 (m, 2H), 6.97–6.94 (m, 2H), 6.70–6.68 (m, 2H), 6.64–6.61 (m, 2H), 3.68 (s, 3H), 3.64 (s, 3H), 3.08–3.04 (m, 1H), 2.13–2.10 (m, 1H), 1.78–1.75 (m, 1H). $^{13}\text{C NMR}$ (101 MHz, CDCl_3) δ 190.83, 173.95, 158.31, 133.69, 133.32, 132.89, 132.03, 131.69, 130.08, 129.07, 128.51, 127.99, 127.84, 114.37, 113.42, 55.22, 55.13, 52.63, 36.42, 32.81, 20.43. IR (ν_{max}): 3020.53,

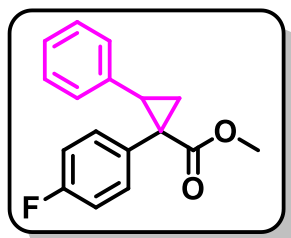
1714.16, 1437.97, 1214.91, 833.50 cm^{-1} . **HRMS (ESI):** m/z calcd for $\text{C}_{18}\text{H}_{18}\text{ClO}_3$ $[\text{M}+\text{H}]^+$ 317.0939, found 317.0942.

Example 38. Methyl 1-(4-bromophenyl)-2-phenylcyclopropane-1-carboxylate (**10d**).



The title compound was synthesised following the general procedure described in section 6.3, and involve corresponding reactant exchange with compound **2b** and compound **9a** (styrene). The crude mixture was concentrated under vacuum and the product was purified by flash chromatography; R_f = 0.5 (5% ethylacetate/hexane); to give compound **10d** (0.494 g in 5 h, 80%) as a colorless oil. **$^1\text{H NMR}$ (400 MHz, CDCl_3)** δ 7.23 (d, J = 8.4 Hz, 2H), 7.08 – 7.05 (m, 3H), 6.88 (d, J = 8.4 Hz, 2H), 6.77 – 6.75 (m, 2H), 3.63 (s, 3H), 3.10 (t, J = 8.3 Hz, 1H), 2.12 (dd, J = 9.3, 5.0 Hz, 1H), 1.82 (dd, J = 7.2, 5.1 Hz, 1H). **$^{13}\text{C NMR}$ (101 MHz, CDCl_3)** δ 173.78, 135.88, 134.03, 133.64, 130.94, 128.08, 127.97, 126.63, 121.26, 52.73, 36.82, 33.21, 20.37. **IR (ν_{max}):** 3021.33, 1715.74, 1491.81, 1214.63, 668.65 cm^{-1} . **HRMS (ESI):** m/z calcd for $\text{C}_{17}\text{H}_{16}\text{BrO}_2$ $[\text{M}+\text{H}]^+$ 331.0328, found 331.0337.

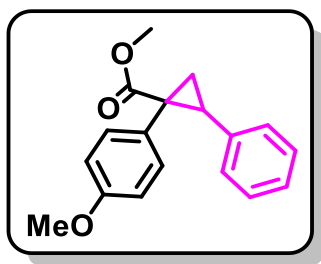
Example 39. Methyl 1-(4-fluorophenyl)-2-phenylcyclopropane-1-carboxylate (**10e**).



The title compound was synthesised following the general procedure described in section 6.3, and involve corresponding reactant exchange with compound **2c** and compound **9a** (styrene).

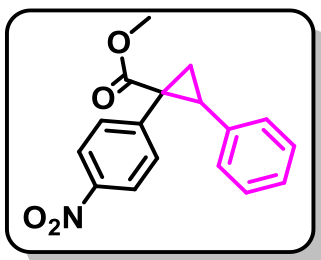
The crude mixture was concentrated under vacuum and the product was purified by flash chromatography; $R_f = 0.5$ (5% ethylacetate/hexane); to give compound **10e** (0.370 g in 6 h, 73%) as a colorless oil. $^1\text{H NMR}$ (500 MHz, CDCl_3) δ 7.05 – 7.01 (m, 3H), 6.97 – 6.94 (m, 2H), 6.78 – 6.73 (m, 4H), 3.59 (s, 3H), 3.10 (dd, $J = 9.3, 7.3$ Hz, 1H), 2.12 (dd, $J = 9.3, 5.0$ Hz, 1H), 1.82 (dd, $J = 7.3, 5.0$ Hz, 1H). $^{13}\text{C NMR}$ (101 MHz, CDCl_3) δ 173.92, 162.89, 160.45, 135.99, 133.49, 133.41, 130.66, 130.63, 127.99, 127.76, 126.41, 114.67, 114.46, 52.46, 36.54, 33.09, 20.36. $^{19}\text{F NMR}$ (377 MHz, CDCl_3) δ -114.84. IR (ν_{max}): 3023.59, 2954.03, 1715.89, 1510.89, 1216.92 cm^{-1} . HRMS (ESI): m/z calcd for $\text{C}_{17}\text{H}_{16}\text{FO}_2$ $[\text{M}+\text{H}]^+$ 271.1129, found 271.1139.

Example 40. Methyl 1-(4-methoxyphenyl)-2-phenylcyclopropane-1-carboxylate (**10f**).



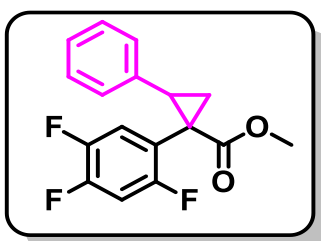
The title compound was synthesised following the general procedure described in section 6.3, and involve corresponding reactant exchange with compound **2d** and compound **9a** (styrene). The crude mixture was concentrated under vacuum and the product was purified by flash chromatography; $R_f = 0.5$ (5% ethylacetate/hexane); to give compound **10f** (0.403 g in 5 h, 76%) as a colorless oil. $^1\text{H NMR}$ (400 MHz, CDCl_3) δ 7.04 – 7.00 (m, 3H), 6.92 – 6.90 (m, 2H), 6.76 – 6.73 (m, 2H), 6.62 (d, $J = 8.6$ Hz, 2H), 3.62 (s, 3H), 3.59 (s, 3H), 3.06 (t, $J = 8.2$ Hz, 1H), 2.11 – 2.07 (m, 1H), 1.81 – 1.71 (m, 1H). $^{13}\text{C NMR}$ (101 MHz, CDCl_3) δ 174.64, 158.56, 136.59, 133.03, 128.21, 127.82, 126.87, 126.37, 113.26, 55.07, 52.63, 36.79, 33.30, 20.81. IR (ν_{max}): 3028.49, 2952.42, 1714.73, 1513.38, 1247.84, 934.88 cm^{-1} . HRMS (ESI): m/z calcd for $\text{C}_{18}\text{H}_{19}\text{O}_3$ $[\text{M}+\text{H}]^+$ 283.1329, found 283.1335.

Example 41. Methyl 1-(4-nitrophenyl)-2-phenylcyclopropane-1-carboxylate (**10g**).



The title compound was synthesised following the general procedure described in section 6.3, and involve corresponding reactant exchange with compound **2e** and compound **9a** (styrene). The crude mixture was concentrated under vacuum and the product was purified by flash chromatography; $R_f = 0.5$ (5% ethylacetate/hexane); to give compound **10g** (0.373 g in 5 h, 67%) as a colorless oil. $^1\text{H NMR}$ (500 MHz, CDCl_3) δ 7.99 – 7.97 (m, 2H), 7.21– 7.18 (m, 2H), 7.09 – 7.07 (m, 3H), 6.80 – 6.7 (m, 2H), 3.68 (s, 3H), 3.21 (dd, $J = 9.3, 7.4$ Hz, 1H), 2.22 (dd, $J = 9.3, 5.2$ Hz, 1H), 1.96 (dd, $J = 7.4, 5.2$ Hz, 1H). $^{13}\text{C NMR}$ (101 MHz, CDCl_3) δ 172.97, 146.94, 142.76, 135.21, 132.89, 128.23, 128.01, 127.06, 122.98, 52.94, 37.04, 33.64, 20.06. IR (ν_{max}): 3023.03, 2953.82, 1720.22, 1519.83, 1214.04 cm^{-1} . HRMS (ESI): m/z calcd for $\text{C}_{17}\text{H}_{16}\text{NO}_4$ $[\text{M}+\text{H}]^+$ 298.1074, found 298.1082.

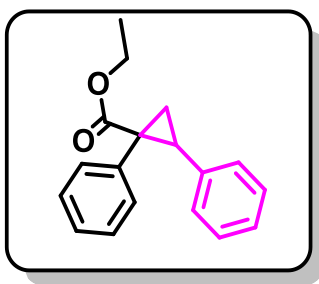
Example 42. Methyl 2-phenyl-1-(2,4,5-trifluorophenyl)cyclopropane-1-carboxylate (**10h**).



The title compound was synthesised following the general procedure described in section 6.3, and involve corresponding reactant exchange with compound **2k** and compound **9a** (styrene). The crude mixture was concentrated under vacuum and the product was purified by flash chromatography; $R_f = 0.5$ (5% ethylacetate/hexane); to give compound **10h** (0.368 g in 5 h, 64%) as a colorless oil. $^1\text{H NMR}$ (400 MHz, CDCl_3) δ 7.11 – 7.09 (m, 3H), 6.88 – 6.85 (m,

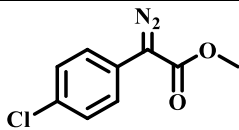
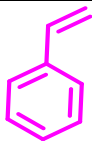
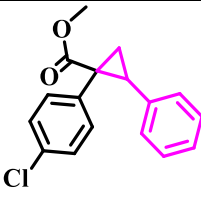
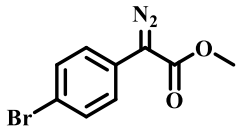
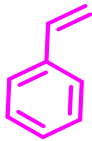
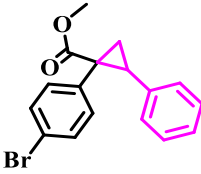
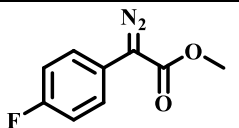
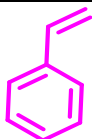
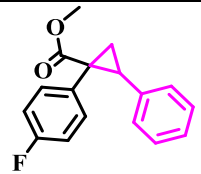
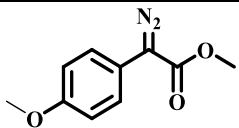
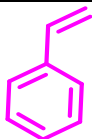
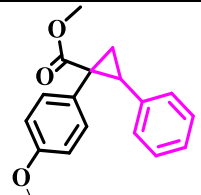
3H), 6.67 – 6.60 (m, 1H), 3.68 (s, 3H), 3.22 (dd, $J = 9.3, 7.6$ Hz, 1H), 2.10 (dd, $J = 9.4, 5.3$ Hz, 1H), 1.86 (dd, $J = 7.5, 5.4$ Hz, 1H). **^{13}C NMR (126 MHz, CDCl_3)** δ 172.59, 158.68, 158.62, 156.71, 156.64, 150.53, 150.42, 150.32, 148.62, 148.53, 148.43, 148.32, 147.15, 147.13, 147.05, 147.03, 145.50, 145.21, 145.18, 145.11, 145.08, 135.62, 135.19, 129.30, 128.13, 127.94, 127.89, 127.17, 126.94, 120.23, 120.19, 120.08, 120.04, 119.60, 119.56, 119.51, 119.47, 119.43, 105.40, 105.24, 105.18, 105.02, 52.75, 52.10, 33.60, 33.10, 32.27, 19.72, 18.86. **^{19}F NMR (471 MHz, CDCl_3)** δ -113.59, -134.16, -134.20, -143.40, -143.44, -143.47. **IR (ν_{max}):** 3027.15, 2926.40, 1725.42, 1516.46, 1264.81, 885.38 cm^{-1} . **HRMS (ESI):** m/z calcd for $\text{C}_{17}\text{H}_{14}\text{F}_3\text{O}_2$ $[\text{M}+\text{H}]^+$ 307.0940, found 307.0949.

Example 43. Methyl 1,2-diphenylcyclopropane-1-carboxylate (**10i**).



The title compound was synthesised following the general procedure described in section 6.3, and involve corresponding reactant exchange with compound **2l** and compound **9a** (styrene). The crude mixture was concentrated under vacuum and the product was purified by flash chromatography; $R_f = 0.5$ (5% ethylacetate/hexane); to give compound **10i** (0.395 g in 5 h, 79%) as a colorless oil. **^1H NMR (300 MHz, CDCl_3)** δ 7.12 – 7.09 (m, 3H), 7.05 – 7.00 (m, 5H), 6.78 – 6.74 (m, 2H), 4.19 – 4.06 (m, 2H), 3.09 (dd, $J = 9.3, 7.3$ Hz, 1H), 2.11 (dd, $J = 9.3, 4.9$ Hz, 1H), 1.86 (dd, $J = 7.3, 4.9$ Hz, 1H), 1.17 (t, $J = 7.1$ Hz, 3H). **^{13}C NMR (101 MHz, CDCl_3)** δ 173.77, 136.59, 134.99, 132.05, 128.18, 127.80, 127.72, 127.02, 126.38, 61.33, 37.73, 33.04, 20.27, 14.30. **IR (ν_{max}):** 3028.05, 2983.04, 1711.52, 1451.75, 1252.14, 1164.61 cm^{-1} . **HRMS (ESI):** m/z calcd for $\text{C}_{18}\text{H}_{19}\text{O}_2$ $[\text{M}+\text{H}]^+$ 267.1380, found 267.1391.

Table S11. Comparative result for C-C bond formation (cyclopropanation).

Entry	Compound 2a	Reagent 9a	Product 10a	Comparative result
1				94.7%, 32 min. (our study) Bis(imino)pyridine iron complexes, 12 h, 88% ⁸ Borane-catalyzed, 18-24 h, 83% ⁹ Tris(pentafluorophenyl)borane-catalyzed, 69% ¹⁰
2				75%, 32 min. (our study) Rh ₂ (S-TPPTTL) ₄ , 13 h, 61% ¹¹ Rh ₂ (S-biTISP) ₂ , 22 h, 86% ¹²
3				73%, 32 min. (our study) bismuth-rhodium complexes, 22 h, 90% (based on ¹ H NMR), 32% ee ¹³
4				76%, 32 min. (our study) Ag, 16 h, 95% ¹⁴ bis(imino)pyridine iron complex, 12 h, 83% ⁸

S7 Case study-5: Carbene insertion (C-C bond formation, cyclopropenation) single objective multi-variant auto-optimization.

S7.1. Background collection

Sample Preparation: We prepare 0.5 M stock solution of compound **2a** in DCE, 0.5 M stock solution of reagent **11a** in DCE and 0.5 M stock solution of product **12a** in DCE. The three of the solution taken in three different syringes.

To conduct our analysis, we utilized an In-line FTIR system. The experimental protocol involved the initial introduction of the solvent, DCE, for a duration of 10 min. using a syringe pump, during which data was recorded. Subsequently, a previously prepared stock solution of compound **2a** was similarly introduced into the In-line FTIR for 10 min. This process was repeated for the reaction product **12a** and reagent **11a**, each pumped for 10 min. Upon collecting all relevant data, we identified the signature peak present in the product. Using this peak as a reference, we employed a Bayesian optimizer to fine-tune and optimize the reaction conditions for improved results.

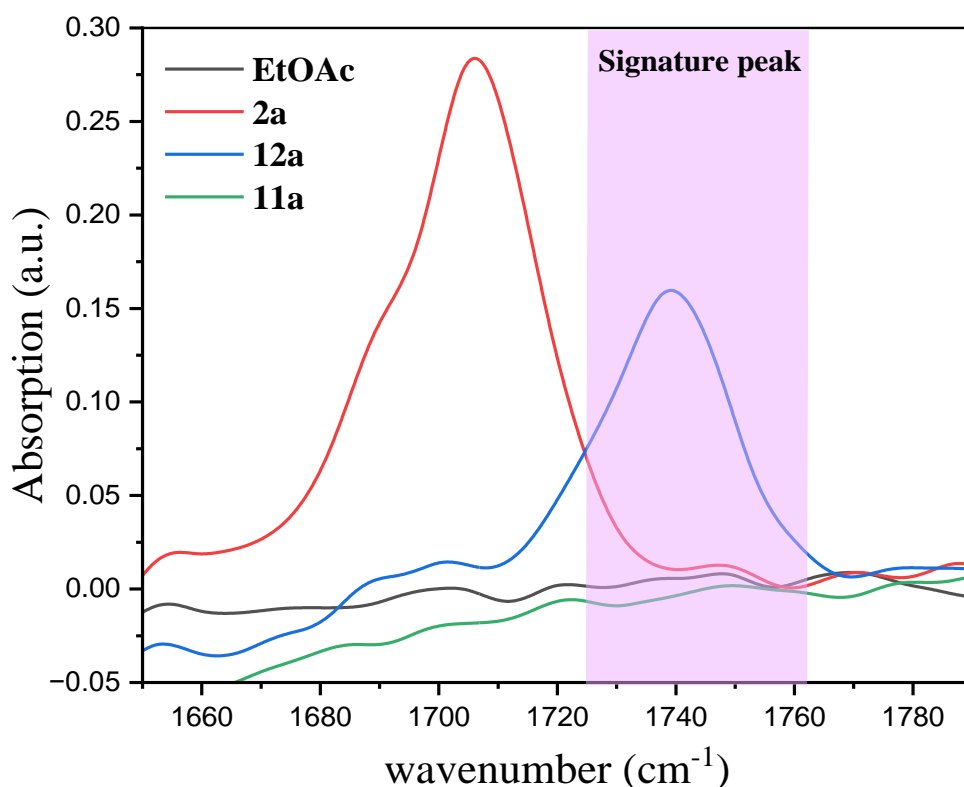


Figure S38. In-line IR background analysis of DCE, 0.5 M stock solution of **2a**, **11a**, and **12a** in DCE.

The distinctive shifting peak associated with product (**12a**) aligning within the range of 1730 to 1770 cm^{-1} was observed. This specific peak, falling within this range, is designated as the signature peak for our analysis. During the optimization of the reaction through Bayesian methods, this signature peak serves as a pivotal reference point. We utilize it to calculate the area under the curve in the In-line FT-IR spectra, providing a quantitative measure of the area in the forthcoming reaction mixture.

S7.2. General procedure for the auto-optimized synthesis of compound 12a.

In our approach we are employing a Python-coded based Bayesian optimization strategy for further refinement. Initial steps involved the preparation of stock solutions for compound **2a** (0.1 M in DCE) and reagent **11a** (0.2 M in DCE), each filled into separate syringes and connected to a syringe pump. The solutions were then directed through a PFA tubular reactor (inner diameter = 1 mm, length = 5.1 m, volume = 4 mL) surrounded by a cylindrical-shaped blue LED light source. Once the solution setup was complete, input ranges for variable flow rates, voltages, and current were specified in the Python code designed for the reaction. The Bayesian optimizer systematically explored these varying reaction conditions, aiming to achieve maximum yield. The optimization process involved running 50-60 experiments and the results were tabulated in the optimization table (Table S12).

S7.2.1. Python code for optimizing condition of C-C bond formation, (cyclopropanation)

```
ab5.py x
1 from os import listdir
2 from os.path import isfile, join
3 import serial
4
5 import numpy as np
6 import pandas as pd
7 import time
8 from scipy.integrate import trapz
9
10 #step1: be sure to the address of the files that the ftir data is exported is matching to line 11 (mypath)
11 mypath = "C:\\Users\\Admin\\Desktop\\ruchi\\Exp 2023-11-04 15-53"
12 onlyfiles = [f for f in listdir(mypath) if isfile(join(mypath, f))]
13
14
15 #step2: make sure that pump and the potentiostat is correctly addressed in the line 16 and 17
16 pump_1 = serial.Serial("COM4",9600) #Harvard Pump
17 port = serial.Serial("COM1",115200)
18 printer = serial.Serial("COM6", 115200, timeout=1)
19
20
21
22 #step 3: grab the lines from 22 to 90 and press f9
23 def area_under(data,start,end):
24     x = np.flip(data.iloc[start:end,0].to_numpy())
25     y = np.flip(data.iloc[start:end,1].to_numpy())
26     area = trapz(y,x)
27     return np.abs(area)
28
29 def file_namer(num):
30     str1 = str(num)
31     length = int(len((str1)))
32     empt = ''
33     for i in range(5-length):
34         empt = empt+'0'
35
36     return empt+str1
37
38
39 def ftir_extract(filename,init,end):
40     filename = filename
41
42     temp_df = pd.read_csv(filename)
43     # nump_df = temp_df.to_numpy()
44     area = area_under(temp_df, init, end)
45     # max_peak = np.max(nump_df[90:120,1])
46     print(area)
47
```

```

48     return area
49
50
51 def function(flowrate_1,v,i):
52
53     #set the pumps with the flowrate as the desired flowrate for the function
54
55     fr_1 = flowrate_1 #ml/min
56     pump_1.write(('irate '+str(fr_1)+' ml/min\r\n').encode())
57
58     time.sleep(0.1)
59
60     #set the com port for potentiostat and set the voltage and current
61     vol = v
62     curr = i
63     port.write(('VOLT '+str(vol)+'\r\n').encode()) #to change the voltge we need to use "VOLT 1" command
64     port.write(('CURR '+str(curr)+'\r\n').encode()) #to change the current we need to use "CURR 1" command
65
66
67     #pumps run
68     pump_1.write((b'irun\r\n'))
69     time.sleep(0.1)
70     time.sleep(4800)
71
72 def function2(flowrate_1,v,i):
73     time.sleep(180)
74
75     files = [f for f in listdir(mypath) if isfile(join(mypath, f))]
76
77     val = 0
78
79     #change wavelengths as per product here.
80     f_row=13 #first row of range for wavelength as per IR CSV
81     l_row=21 #last row of range for wavelength as per IR CSV
82
83     val += ftir_extract(files[-1],f_row,l_row)
84     val+=ftir_extract(files[-2],f_row,l_row)
85     val+=ftir_extract(files[-3],f_row,l_row)
86
87     avg_val = val/3
88
89     return avg_val
90
91
92 #step 4:grab the line 93 and f9
93 from skopt.optimizer import Optimizer
94

```

```

95
96 #step5:in line 96 we have to define the range that (flowrate,voltage,current) (from,to) and after (anytime) applying changes you need to grab the line 96 an
97 #flowrates are in ml/min, voltage in V, Current in Ampere
98 bounds = [(0.05,0.15),(10,14.5),(4.9,5.0)]
99
100
101 #step 6: grab the line 100 and f9
102 opter =Optimizer(bounds,base_estimator='gp',n_initial_points=3,acq_func="EI",random_state=np.random.randint(3326))
103
104
105 #step7: to selecte number of the cycles that you have to do the experiment and then grab the line 104 to 121 and f9: the closed loop experimentation is initi
106 number_of_cycles = 22
107 results = []
108 flowrates_1 = []
109 vs = []
110 currents = []
111
112 product_wavelength=True #set to true if product wavelengths being monitored
113
114 if product_wavelength == True:
115     val=1
116 else:
117     val=-1
118
119
120 # Step 8: Test Tubes on Printer
121 USE_PRINTER = True
122 REST_HEIGHT = 200
123 X_HOME = -5
124 Y_HOME = 20
125 Z_HOME = 175
126 DEFAULT_PUMP_TIME="1"
127 # Distance between test tubes
128 X_SPACING=20
129 Y_SPACING=20
130 # Number of test tubes
131 X_ROWS = 11
132 Y_COLUMNS = 4
133
134 def send_cmd(cmd):
135     print(cmd)
136     printer.write(f"{cmd}\n".encode("ASCII"))
137
138 def move(x=None, y=None, z=None):

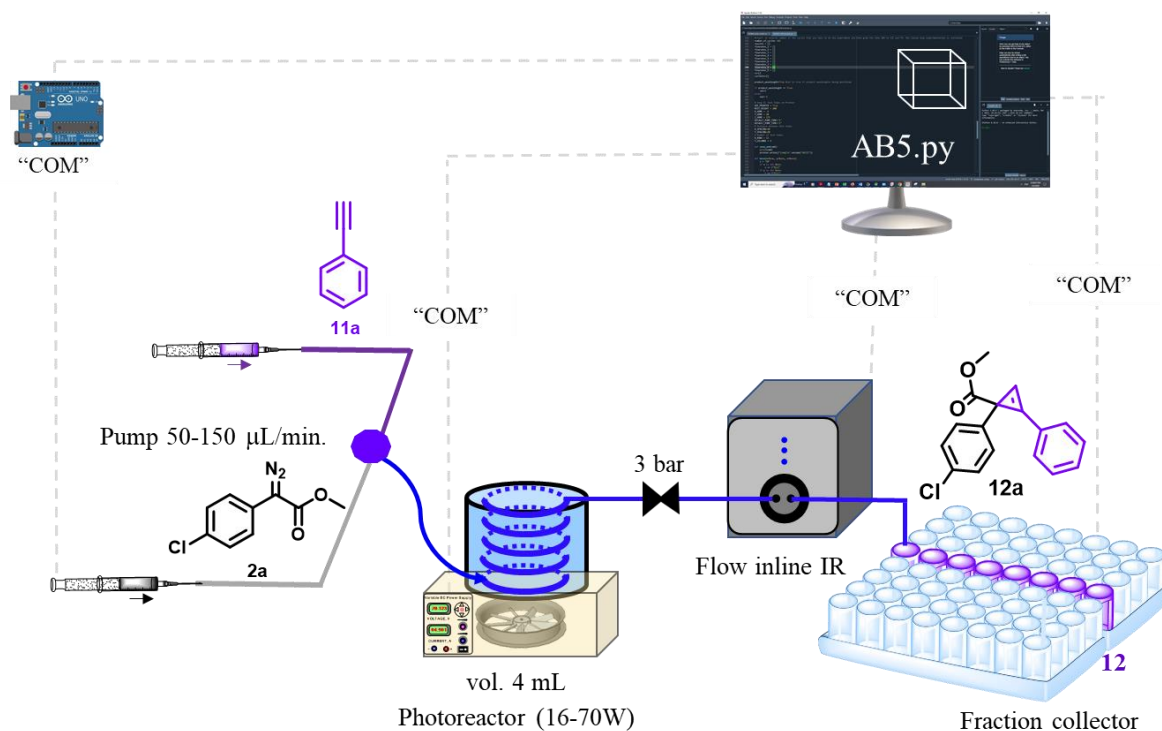
```



```

139 s = "G0"
140 if x is not None:
141     s += f"X{x}"
142 if y is not None:
143     s += f"Y{y}"
144 if z is not None:
145     s += f"Z{z}"
146
147 s+= "F5000"
148 send_cmd(s)
149
150 def printer_positions():
151     for j in range(Y_COLUMNS):
152         for i in range(X_ROWS):
153             if j%2==1:
154                 yield (X_HOME + (X_ROWS - 1 - i) * X_SPACING, Y_HOME + j * Y_SPACING, Z_HOME)
155             else:
156                 yield (X_HOME + i * X_SPACING, Y_HOME + j * Y_SPACING, Z_HOME)
157
158 # Run this.
159 tube_location = list(printer_positions())
160
161
162 for i in range(number_of_cycles):
163     move(*tube_location[2*i])
164     asked = opter.ask()
165     function(asked[0],asked[1],asked[2])
166     move(*tube_location[2*i+1])
167     told = function2(asked[0],asked[1],asked[2])
168
169     print(f"area under the curve in the round {i:.2f} = {told: .2f}")
170     opter.tell(asked, -told*val)
171     results.append(told)
172     flowrates_1.append(asked[0])
173     vs.append(asked[1])
174     currents.append(asked[2])
175
176     dict1 = {"flowrate_1":flowrates_1,"voltages":vs, "currents":currents,"area-results":results}
177     df2 = pd.DataFrame(dict1)
178     df2.to_csv("output round "+str(i)+".csv")
179
180
181
182 pump_1.write(b'stop\r\n')
183
184
185

```

Table S12. Auto-optimization table of C-C bond formation, (cyclopropenation)

No. of experiments	Flow rate mL/min	Voltage (V)	Light intensity (W)	area-results	Yield (%)	Space time yield (g mL ⁻¹ h ⁻¹)
1	0.120701	14.1092	60.443	0.454837	73.79562	0.018972347
2	0.090227	14.0756	60.142	0.467826	75.90304	0.014587312
3	0.066047	10.6181	20.55	0.114833	18.63123	0.002621043
4	0.052175	14.1067	60.91	0.445654	72.30571	0.008035533
5	0.05922	10.4612	19.499	0.106311	17.24857	0.00217571
6	0.127527	13.1035	49.136	0.358988	58.24447	0.015821093
7	0.130375	10.6304	21.864	0.249116	40.41815	0.01122407
8	0.093723	14.4455	68.115	0.583323	94.642	0.018893381
9	0.069852	12.4586	49.071	0.41648	67.57234	0.010053734
10	0.15	14.5	68.251	0.48846	79.25083	0.025320639
11	0.149893	10.9074	23.886	0.260363	42.24293	0.01348699
12	0.099249	14.5	68.105	0.574865	93.26972	0.019717253
13	0.05	14.5	68.077	0.497185	80.66643	0.008590974
14	0.15	11.2061	26.928	0.254111	41.22857	0.013172528
15	0.11223	14.5	68.352	0.539139	87.47331	0.020910486

16	0.144445	14.5	68.164	0.452476	73.41256	0.022586684
17	0.064064	14.5	68.091	0.514453	83.4681	0.011389749
18	0.054786	10.9936	24.807	0.238132	38.63604	0.004508601
19	0.098561	13.1959	50.192	0.135787	22.03094	0.004625063
20	0.107359	10.7921	22.836	0.231832	37.61388	0.008601343
21	0.099419	13.6901	56.831	0.288356	46.78469	0.009907242
22	0.099154	14.4518	69.581	0.591693	96.00162	0.020275353
23	0.078101	10.4687	19.751	0.212029	34.40092	0.005722277
24	0.149026	13.6182	55.984	0.326608	52.99094	0.016820669
25	0.128138	12.0086	35.58	0.349058	56.63337	0.015457169
26	0.133455	13.0499	48.233	0.377999	61.32894	0.017433312
27	0.138758	11.0478	25.215	0.373725	60.6355	0.017921097
28	0.15	10.2395	17.684	0.307325	49.86234	0.015931019
29	0.09817	14.4834	68.2	0.559436	90.76642	0.018979449
30	0.091267	13.7496	57.585	0.510067	82.75648	0.016087754
31	0.126242	12.456	40.893	0.384914	62.45087	0.016792756
32	0.068906	12.2385	38.326	0.424445	68.86463	0.010107247
33	0.092488	14.5	68.324	0.56245	91.25543	0.017977269
34	0.112444	13.2561	51.079	0.403716	65.50143	0.015687966
35	0.075546	14.4222	67.192	0.506462	82.17159	0.013222475
36	0.088877	13.7497	57.553	0.351022	56.95202	0.010781474
37	0.092205	13.7496	57.553	0.373007	60.51901	0.01188573
38	0.097674	14.4875	67.973	0.565181	91.69853	0.019077477
39	0.125538	12.4198	40.238	0.361648	58.67605	0.015689738
40	0.125824	12.4595	40.787	0.387129	62.81025	0.016833468
41	0.12398	12.5461	42.079	0.403667	65.49348	0.017295347
42	0.139188	10.9791	24.766	0.253061	41.05821	0.012172546
43	0.15	12.1401	37.379	0.251172	40.75173	0.013020177
44	0.15	10.8394	23.41	0.227043	36.83689	0.011769385
45	0.133045	12.5478	42.296	0.293369	47.59803	0.01348861
46	0.15	13.8612	59.614	0.303369	49.2205	0.015725949
47	0.149999	11.0341	25.511	0.233369	37.86326	0.01209723
48	0.050002	11.4698	29.911	0.173369	28.12848	0.002995803

49	0.098806	14.4781	67.772	0.569669	92.4267	0.019451826
50	0.149282	10.0468	16.022	0.133369	21.63863	0.006880448
51	0.140899	11.6944	32.679	0.264725	42.95065	0.012890129
52	0.137499	14.2234	65.412	0.395556	64.17815	0.018796038
53	0.063767	10.58	21.028	0.221869	35.99742	0.004889304
54	0.138262	11.4412	29.286	0.208132	33.76865	0.009944801
55	0.076134	13.8555	59.2	0.413512	67.09079	0.010879806

Reaction condition: compound **2a** (0.1 M in DCE); reagent **11a** (0.2 M in DCE); 4 mL reactor volume.

Graph:

We have plotted 3D graph of optimization table S12, we have taken flow rate on X-axis, blue light intensity (watt) on Y-axis and product yield on Z axis.

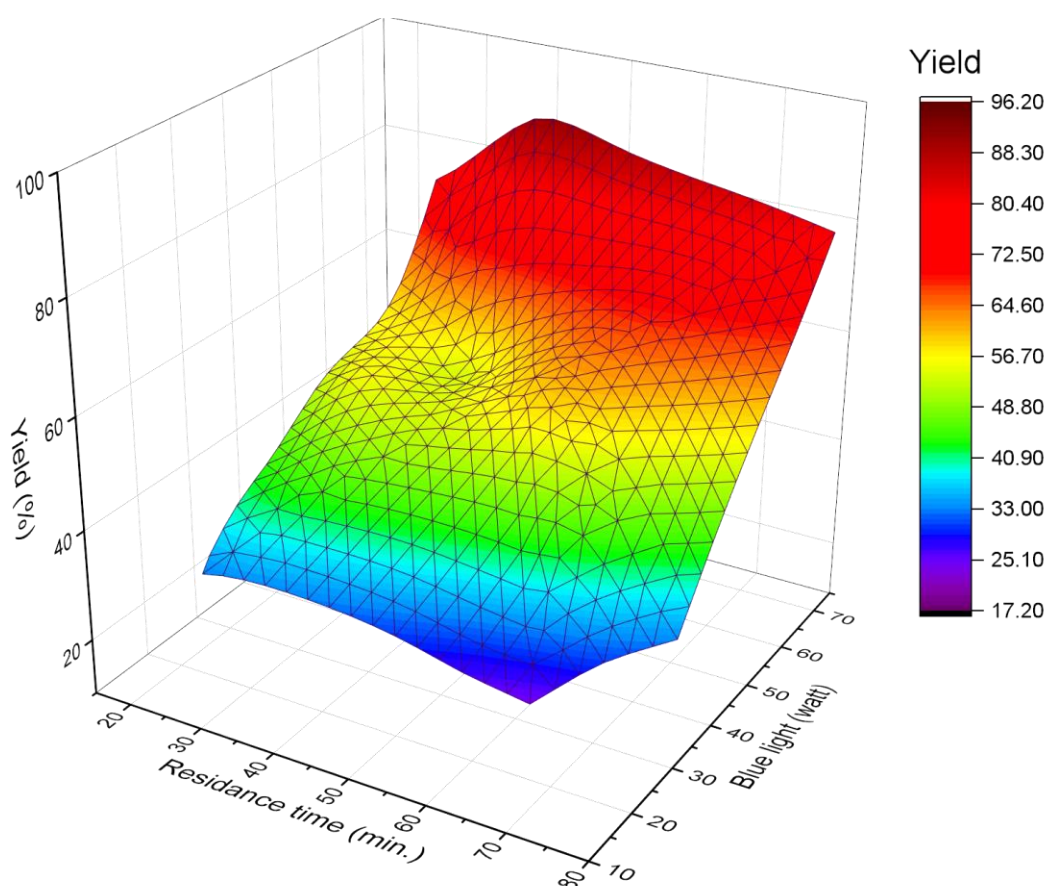


Figure S39. AI based system to auto-optimize and navigate this complexity and identify the optimal conditions for the photo activated cyclopropanation reaction. Compound **2a** (0.1 M in DCE); reagent **11a** (0.2 M in DCE); 4 mL reactor volume.

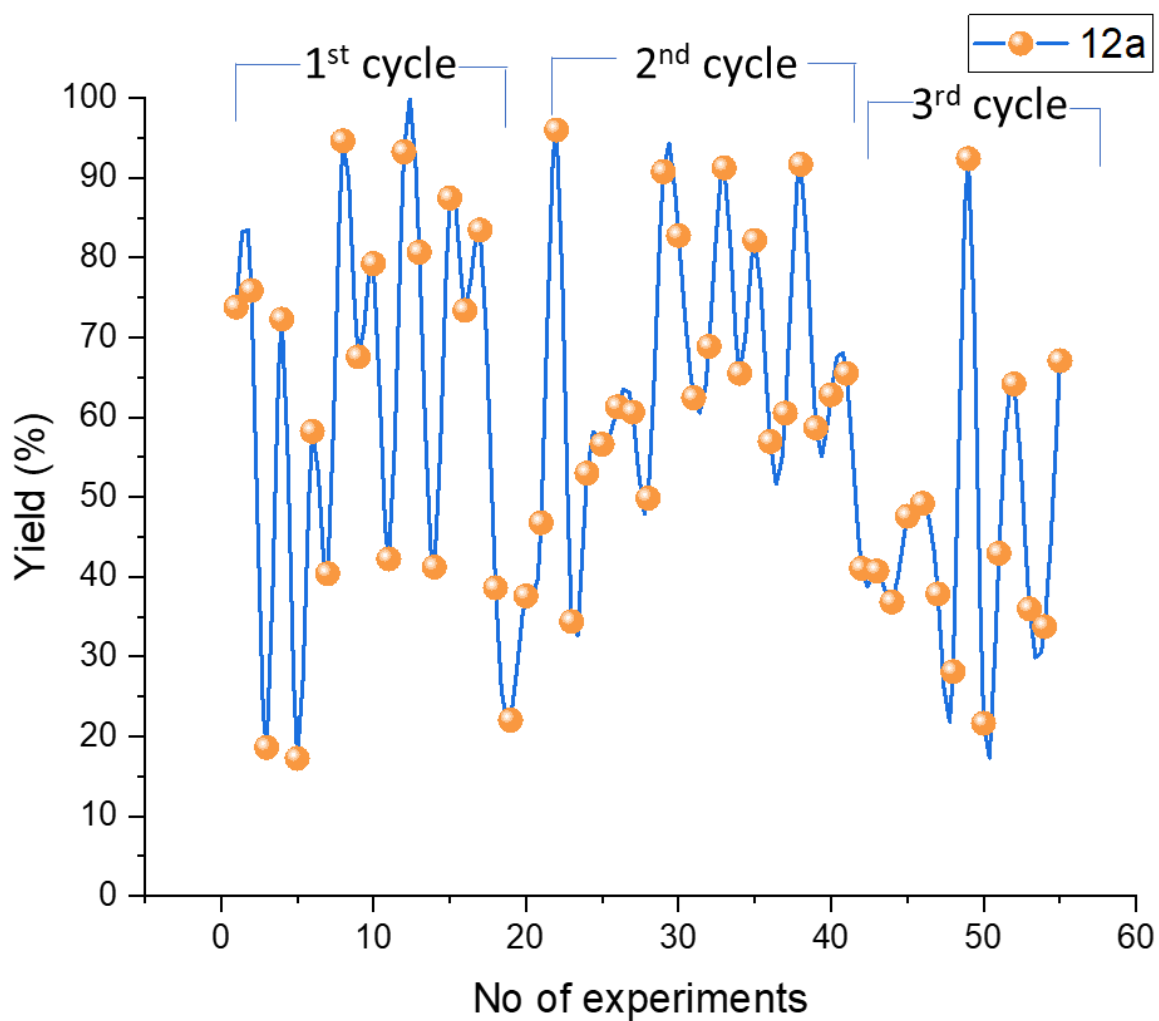


Figure S40. 2D graph between no of experiments performed versus yield (%) for the AI based auto-optimization of photo activated cyclopropanation reaction. Compound **2a** (0.1 M in DCE); reagent **11a** (0.2 M in DCE); 4 mL reactor volume.

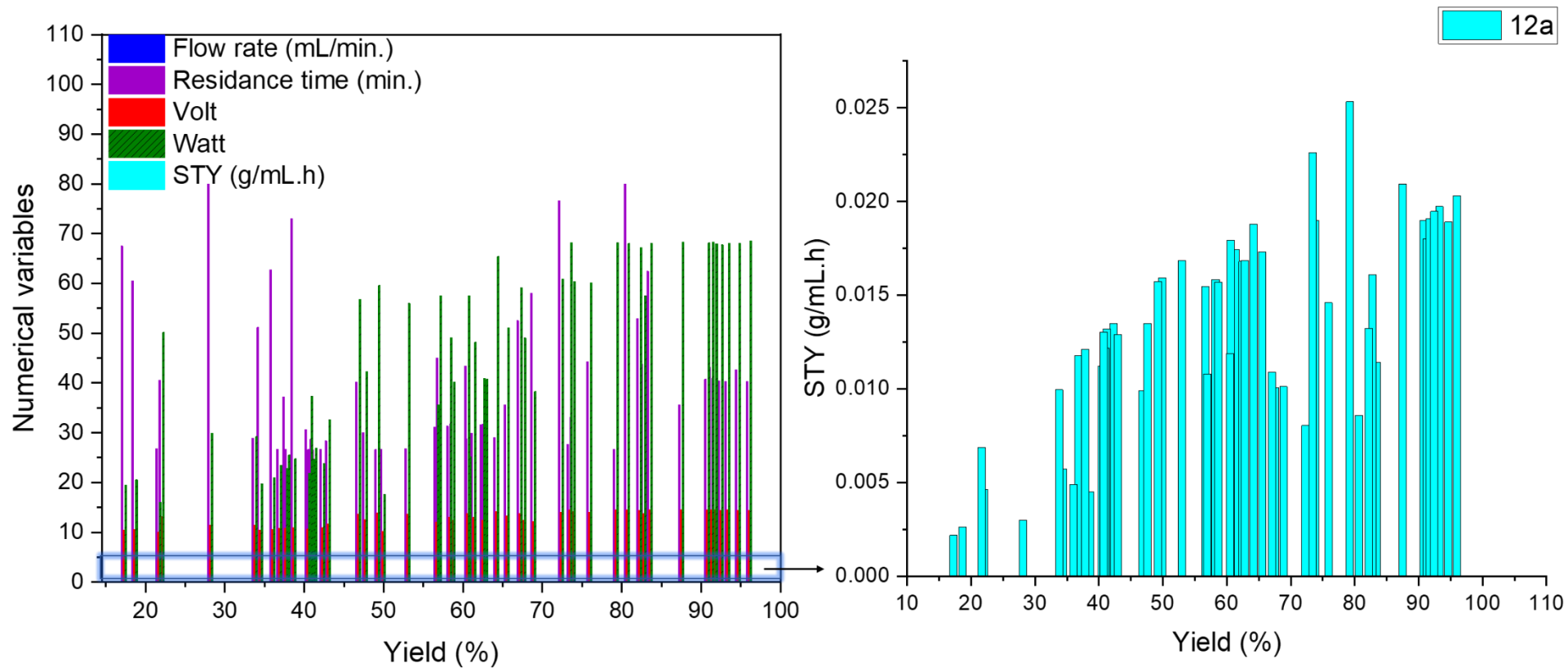


Figure S41. AI based system navigated multi-numerical variable complexity and identify the optimal condition for the photo activated cyclopropanation reaction. Compound **2a** (0.1 M in DCE); reagent **11a** (0.2 M in DCE); 4 mL reactor volume.

S7.3. General procedure for AI optimized carbene insertion into the triple bonds of aromatic alkynes for synthesis of compound **12a**

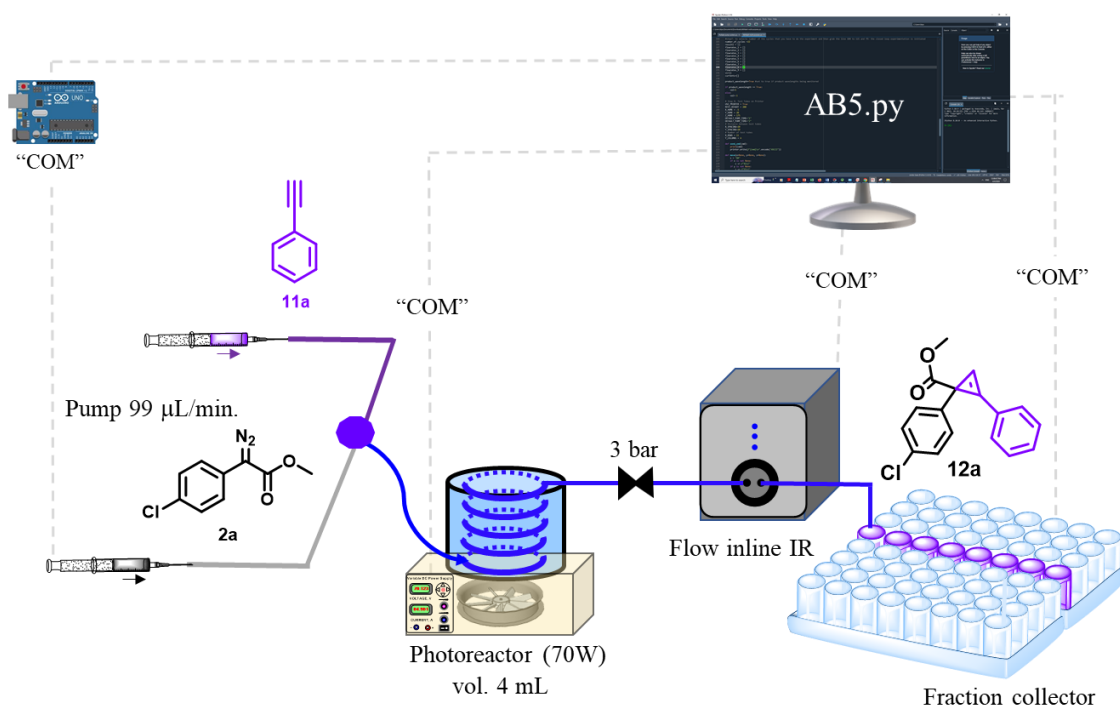


Figure S42. Schematic presentation of continuous flow for the synthesis, of compound **12a**.

To prepare the stock solution of compound **2a** (2.20 g, 0.01 mol) was dissolved in ethyl acetate (100 mL) and connected with pump. Another pump was connected with the reagent **11a** (2.3 mL, 0.02 mol) dissolved in ethyl acetate (100 mL). These two pump out-put is further connected with the T₁-mixer and pumps were running with the flow rate 50 μL/min.each to maintain the stoichiometry and then passed through PFA tubular reactor (id = 1000 μm, l = 5 m, vol. = 4 mL) under blue light (68.581 W) exposure. The room temperature of the reactor was controlled by fan attached to the bottom of the photochemical reactor. The out-put of the tubular reactor was connected with spring based back pressure regulator (~3 bar) to maintain the evaporation. First one hour of the product mixture was discarded and next 5 h of the product mixture [30 mL; **2a** (0.315 g)] was collected in HPLC bottle. The organic EtOAc phase was concentrated under reduced pressure to give the product **12a** (0.410 g in 5 h, 96%) as a yellow

oil. **$^1\text{H NMR}$ (400 MHz, CDCl_3)** δ 7.59 – 7.57 (m, 2H), 7.42 – 7.39 (m, 3H), 7.34 – 7.31 (m, 2H), 7.25 – 7.22 (m, 2H), 7.17 (s, 1H), 3.70 (s, 3H). **$^{13}\text{C NMR}$ (126 MHz, CDCl_3)** δ 174.65, 139.48, 132.32, 130.25, 129.94, 129.68, 129.01, 128.23, 125.11, 117.09, 99.86, 52.32, 32.98. **IR** (ν_{max}): 3021.78, 1720.27, 1595.83, 1490.51, 1214.17, 830.32 cm^{-1} . **HRMS (ESI)**: m/z calcd for $\text{C}_{17}\text{H}_{14}\text{ClO}_2$ $[\text{M}+\text{H}]^+$ 285.0677, found 285.0686. Verified the analytical data with those reported in the literature.¹⁵

Space time yield in previous reported batch process ¹⁵

Productivity = 67 mg in 16 h

$$\text{Productivity (per h)} := \frac{0.067}{16} = 0.0041 \text{ g/h}$$

$$\text{Space time yield} := \frac{\text{productivity}}{\text{volume of reactor}}$$

$$= \frac{0.0041}{2} = 0.002 \text{ g mL}^{-1} \text{ h}^{-1}$$

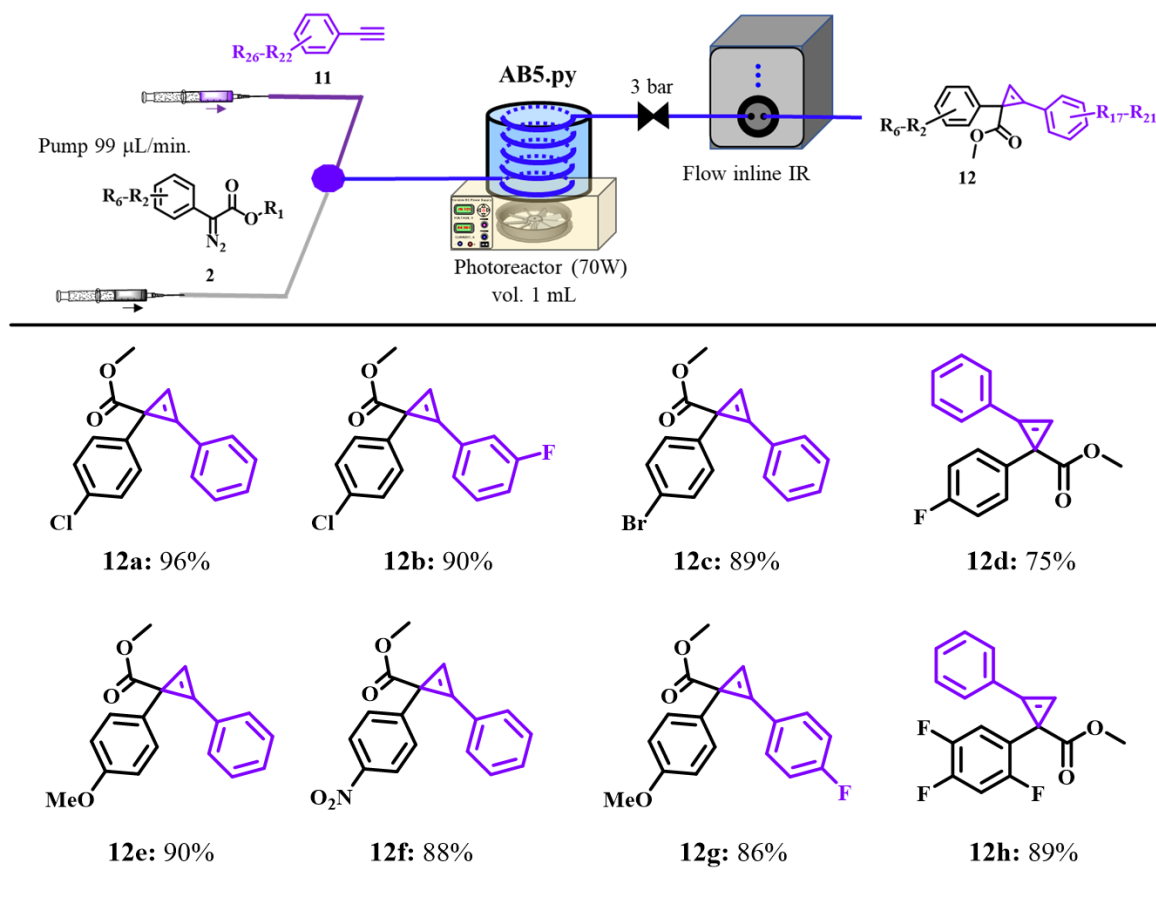
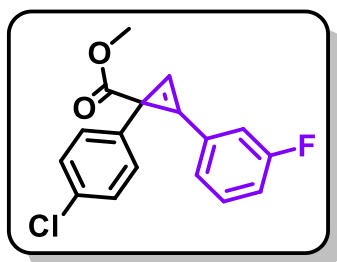


Figure S43. Scope of substrate for the photo activated cyclopropanation reaction. Compound **2a** (0.1 M in DCE); reagent **11a** (0.2 M in DCE); 4 mL reactor volume.

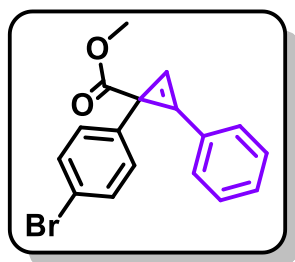
S.7.3.1. General procedure for scope of substrate for carbene insertion into the S–H bonds of organic aromatic thiol for synthesis of compound 12b-12h.

Example 45. Methyl 1-(4-chlorophenyl)-2-(3-fluorophenyl)cycloprop-2-ene-1-carboxylate (**12b**).



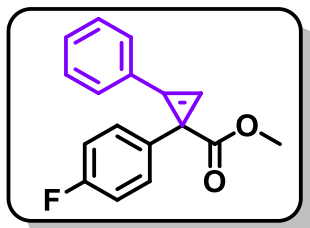
The title compound was synthesised following the general procedure described in section 7.3, and involve corresponding reactant exchange with compound **2a** and compound **11b** (1-ethynyl-3-fluorobenzene). The crude mixture was concentrated under vacuum and the product was purified by flash chromatography; $R_f = 0.4$ (5% ethylacetate/hexane); to give compound **12b** (0.410 g in 5 h, 90%) as a yellow oil. $^1\text{H NMR}$ (500 MHz, CDCl_3) δ 7.42 – 7.35 (m, 2H), 7.32 – 7.23 (m, 6H), 7.12 – 7.08 (m, 1H), 3.71 (s, 3H). $^{13}\text{C NMR}$ (101 MHz, CDCl_3) δ 174.26, 164.08, 161.62, 139.02, 132.52, 130.72, 130.64, 129.59, 129.28, 128.32, 127.11, 125.66, 125.64, 117.45, 117.24, 116.60, 116.38, 101.44, 52.40, 33.28. $^{19}\text{F NMR}$ (471 MHz, CDCl_3) δ -112.83 (s). IR (ν_{max}): 3021.73, 1726.97, 1587.32, 1214.51, 1014.81 cm^{-1} . HRMS (ESI): m/z calcd for $\text{C}_{17}\text{H}_{13}\text{ClFO}_2$ $[\text{M}+\text{H}]^+$ 303.0583, found 303.0597.

Example 46. Methyl 1-(4-bromophenyl)-2-phenylcycloprop-2-ene-1-carboxylate (**12c**).



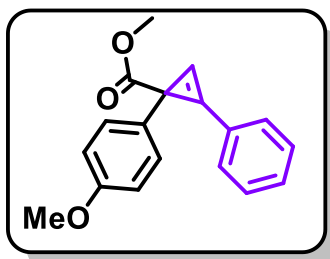
The title compound was synthesised following the general procedure described in section 7.3, and involve corresponding reactant exchange with compound **2b** and compound **11a** (ethynylbenzene). The crude mixture was concentrated under vacuum and the product was purified by flash chromatography; $R_f = 0.4$ (5% ethylacetate/hexane); to give compound **12c** (0.440 g in 5 h, 89%) as a colorless oil. $^1\text{H NMR}$ (400 MHz, CDCl_3) δ 7.59 – 7.57 (m, 2H), 7.42 – 7.39 (m, 3H), 7.33 (d, $J = 12.0$ Hz, 2H), 7.24 (d, $J = 8.5$ Hz, 2H), 7.17 (s, 1H), 3.70 (s, 3H). $^{13}\text{C NMR}$ (101 MHz, CDCl_3) δ 174.57, 140.01, 131.18, 130.28, 130.07, 129.95, 129.03, 125.09, 120.45, 117.04, 99.78, 52.34, 33.06. IR (ν_{max}): 3136.19, 2950.74, 1719.37, 1587.93, 1216.09, 629.14 cm^{-1} . HRMS (ESI): m/z calcd for $\text{C}_{17}\text{H}_{14}\text{BrO}_2$ $[\text{M}+\text{H}]^+$ 329.0172, found 329.0180.

Example 47. Methyl 1-(4-fluorophenyl)-2-phenylcycloprop-2-ene-1-carboxylate (**12d**).



The title compound was synthesised following the general procedure described in section 7.3, and involve corresponding reactant exchange with compound **2c** and compound **11a** (ethynylbenzene). The crude mixture was concentrated under vacuum and the product was purified by flash chromatography; $R_f = 0.4$ (5% ethylacetate/hexane); to give compound **12d** (0.302 g in 5 h, 75%) as a yellow oil. $^1\text{H NMR}$ (500 MHz, CDCl_3) δ 7.64 – 7.62 (m, 2H), 7.47 – 7.38 (m, 5H), 7.22 (s, 1H), 7.01 – 6.98 (m, 2H), 3.73 (s, 3H). $^{13}\text{C NMR}$ (101 MHz, CDCl_3) δ 174.96, 162.85, 160.42, 136.79, 130.22, 129.95, 129.90, 129.04, 125.30, 117.41, 115.07, 114.86, 100.23, 52.33, 32.95. $^{19}\text{F NMR}$ (377 MHz, CDCl_3) δ -116.21. IR (ν_{max}): 3024.19, 2952.42, 1718.44, 1507.81, 1217.41, 1019.92, 837.24 cm^{-1} . HRMS (ESI): m/z calcd for $\text{C}_{17}\text{H}_{14}\text{FO}_2$ $[\text{M}+\text{H}]^+$ 269.0972, found 269.0990.

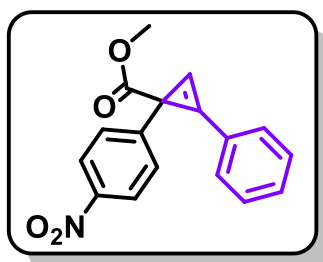
Example 48. Methyl 1-(4-methoxyphenyl)-2-phenylcycloprop-2-ene-1-carboxylate (**12e**).



The title compound was synthesised following the general procedure described in section 7.3, and involve corresponding reactant exchange with compound **2d** and compound **11a** (ethynylbenzene). The crude mixture was concentrated under vacuum and the product was purified by flash chromatography; $R_f = 0.4$ (5% ethylacetate/hexane); to give compound **12e**

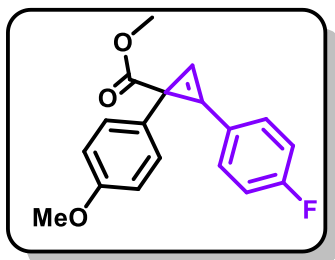
(0.380 g in 5 h, 90%) as a yellow oil. **¹H NMR (400 MHz, CDCl₃)** δ 7.60 – 7.58 (m, 2H), 7.40 – 7.30 (m, 5H), 7.18 (s, 1H), 6.81 (d, *J* = 8.7 Hz, 2H), 3.72 (s, 3H), 3.68 (s, 3H). **¹³C NMR (101 MHz, CDCl₃)** δ 175.36, 158.32, 133.16, 130.02, 129.93, 129.40, 128.96, 125.59, 117.71, 113.62, 100.63, 55.25, 52.23. **IR (ν_{max}):** 3021.06, 1718.02, 1606.03, 1511.72, 1214.64 cm⁻¹. **HRMS (ESI):** *m/z* calcd for C₁₈H₁₇O₃ [M+H]⁺ 281.1172, found 281.1185.

Example 49. Methyl 1-(4-nitrophenyl)-2-phenylcycloprop-2-ene-1-carboxylate (**12f**).



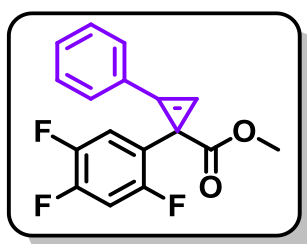
The title compound was synthesised following the general procedure described in section 7.3, and involve corresponding reactant exchange with compound **2e** and compound **11a** (ethynylbenzene). The crude mixture was concentrated under vacuum and the product was purified by flash chromatography; **R_f** = 0.4 (5% ethylacetate/hexane); to give compound **12f** (0.390 g in 5 h, 88%) as a yellow oil. **¹H NMR (400 MHz, CDCl₃)** δ 8.16 – 8.13 (m, 2H), 7.60 – 7.56 (m, 4H), 7.47 – 7.44 (m, 3H), 7.19 (s, 1H), 3.73 (s, 3H). **¹³C NMR (101 MHz, CDCl₃)** δ 173.75, 148.59, 146.45, 130.63, 129.99, 129.13, 129.05, 124.45, 123.33, 116.18, 98.99, 52.46, 33.25. **IR (ν_{max}):** 3022.46, 2925.68, 1722.41, 1520.33, 1215.42 cm⁻¹. **HRMS (ESI):** *m/z* calcd for C₁₇H₁₄NO₄ [M+H]⁺ 296.0917, found 296.0927.

Example 50. Methyl 2-(4-fluorophenyl)-1-(4-methoxyphenyl)cycloprop-2-ene-1-carboxylate (**12g**).



The title compound was synthesised following the general procedure described in section 7.3, and involve corresponding reactant exchange with compound **2a** and compound **11c** (1-ethynyl-4-fluorobenzene). The crude mixture was concentrated under vacuum and the product was purified by flash chromatography; $R_f = 0.4$ (5% ethylacetate/hexane); to give compound **12g** (0.385 g in 5 h, 86%) as a yellow oil. $^1\text{H NMR}$ (400 MHz, CDCl_3) δ 7.60 – 7.58 (m, 2H), 7.40 – 7.30 (m, 5H), 6.81 (d, $J = 8.8$ Hz, 2H), 3.72 (s, 3H), 3.68 (s, 3H). $^{13}\text{C NMR}$ (101 MHz, CDCl_3) δ 175.36, 158.32, 133.16, 130.02, 129.93, 129.40, 128.96, 125.59, 117.71, 113.62, 100.63, 55.25, 52.23. $^{19}\text{F NMR}$ (376 MHz, CDCl_3) δ -110.11, IR (ν_{max}): 2954.04, 2844.40, 1723.50, 1602.75, 1247.68, 1029.28 cm^{-1} . HRMS (ESI): m/z calcd for $\text{C}_{18}\text{H}_{16}\text{FO}_3$ $[\text{M}+\text{H}]^+$ 299.1078, found 299.1089.

Example 51. Methyl 2-phenyl-1-(2,4,5-trifluorophenyl)cycloprop-2-ene-1-carboxylate (**12h**).



The title compound was synthesised following the general procedure described in section 7.3, and involve corresponding reactant exchange with compound **2k** and compound **11a** (ethynylbenzene). The crude mixture was concentrated under vacuum and the product was purified by flash chromatography; $R_f = 0.4$ (5% ethylacetate/hexane); to give compound **12h** (0.408 g in 5 h, 89%) as a yellow oil. $^1\text{H NMR}$ (500 MHz, CDCl_3) δ 7.66 – 7.65 (m, 2H), 7.48

– 7.42 (m, 3H), 7.22 (s, 1H), 7.03 – 6.98 (m, 1H), 6.93 – 6.88 (m, 1H), 3.71 (s, 3H). **¹³C NMR (176 MHz, CDCl₃)** δ 174.06, 157.56, 156.13, 149.99, 148.57, 147.43, 145.97, 130.64, 130.05, 129.14, 125.39, 125.29, 124.77, 118.69, 117.83, 117.75, 105.78, 105.62, 105.50, 98.98, 52.69, 29.83, 29.28. **¹⁹F NMR (471 MHz, CDCl₃)** δ -115.94, -115.96, -135.30, -135.34, -142.94, -142.98, -143.01. **IR (ν_{max}):** 3071.13, 2956.10, 1724.39, 1512.59, 1247.95, 880.71 cm⁻¹. **HRMS (ESI):** *m/z* calcd for C₁₇H₁₂F₃O₂ [M+H]⁺ 305.0784, found 305.0797.

Table S13. Comparative result for C-C bond formation, (cyclopropenation).

Entry	Compound 2a	Reagent 11a	Product 12a	Comparative result
1				95%, 40 min, (our study) Blue LED, 16 h, 80% ¹⁵ dirhodium tetraacetate, 10 h, 72% ¹⁶
2				89%, 40 min, (our study) Cu, 16h, 81% ¹⁷ dirhodium tetraacetate, 10h, 69% ¹⁶
3				75%, 40 min, (our study) tris(pentafluorophenyl)borate, 24h, 75% ¹⁸ Irradiation, 16 h, 56% ¹⁵
4				90%, 40 min, (our study) tris(pentafluorophenyl)borate, 24h, 50% ¹⁸ dirhodium tetraacetate, 10 h, 50% ¹⁹
5				88%, 40 min, (our study) dirhodium tetraacetate, 64% ²⁰

S8. Case study-6: Carbene insertion (N-C-O bond formation, oxazole) single objective multi-variant auto-optimization.

S.8.1 Background collection

Sample Preparation: We prepare 0.5 M stock solution of compound **2a** in DCE, 0.5 M stock solution of reagent **13a** in DCE and 0.5 M stock solution of product **14a** in DCE. The three of the solution taken in three different syringes.

To conduct our analysis, we utilized an In-line FTIR system. The experimental protocol involved the initial introduction of the solvent, DCE, for a duration of 10 min. using a syringe pump, during which data was recorded. Subsequently, a previously prepared stock solution of compound **2a** was similarly introduced into the In-line FTIR for 10 min. This process was repeated for the reaction product **14a** and reagent **13a**, each pumped for 10 min. Upon collecting all relevant data, we identified the signature peak present in the product. Using this peak as a reference, we employed a Bayesian optimizer to fine-tune and optimize the reaction conditions for improved results.

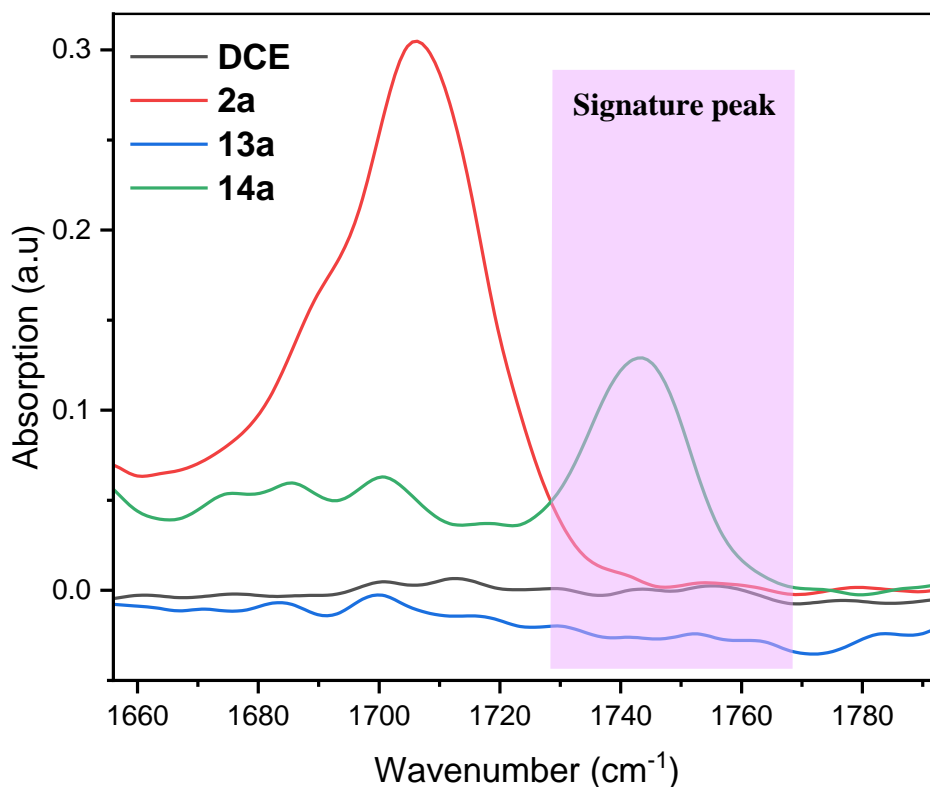


Figure S44. In-line IR background analysis of DCE, 0.5 M stock solution of **2a**, **13a**, and **14a** in DCE.

The distinctive shifting peak associated with product (**14a**) aligning within the range of 1730 to 1770 cm^{-1} . This specific peak, falling within this range, is designated as the signature peak for our analysis. During the optimization of the reaction through Bayesian methods, this signature peak serves as a pivotal reference point. We utilize it to calculate the area under the curve in the In-line FT-IR spectra, providing a quantitative measure of the area in the forthcoming reaction mixture.

S8.2. General procedure for the auto-optimized synthesis of compound 14a.

In our approach we are employing a Python-coded based Bayesian optimization strategy for further refinement. Initial steps involved the preparation of stock solutions for compound **2a**

(0.1 M in DCE) and reagent **13a** (0.2 M in DCE), each filled into separate syringes and connected to a syringe pump. The solutions were then directed through a PFA tubular reactor (inner diameter = 1 mm, length = 1.3 m, volume = 1 mL) surrounded by a cylindrical-shaped blue LED light source. Once the solution setup was complete, input ranges for variable flow rates, voltages, and current were specified in the Python code designed for the reaction. The Bayesian optimizer systematically explored these varying reaction conditions, aiming to achieve maximum yield. The optimization process involved running 40-45 experiments and the results were tabulated in the optimization table (Table S14).

S8.2.1. Python code for optimized condition for N-C-O bond formation, (oxazole)

```
ab6.py
1 from os import listdir
2 from os.path import isfile, join
3 import serial
4
5 import numpy as np
6 import pandas as pd
7 import time
8 from scipy.integrate import trapz
9
10 #step1: be sure to the address of the files that the ftir data is exported is matching to line 11 (mypath)
11 mypath = "C:\\Users\\Admin\\Desktop\\ruchi\\Exp 2024-03-02 14-49"
12 onlyfiles = [f for f in listdir(mypath) if isfile(join(mypath, f))]
13
14
15 #step2: make sure that pump and the potentiostat is correctly addressed in the line 16 and 17
16 pump_1 = serial.Serial("COM4",9600) #Harvard Pump
17 port = serial.Serial("COM1",115200)
18 printer = serial.Serial("COM9", 115200, timeout=1)
19
20
21
22 #step 3: grab the lines from 22 to 90 and press f9
23 def area_under(data,start,end):
24     x = np.flip(data.iloc[start:end,0].to_numpy())
25     y = np.flip(data.iloc[start:end,1].to_numpy())
26     area = trapz(y,x)
27     return np.abs(area)
28
29 def file_namer(num):
30     str1 = str(num)
31     length = int(len((str1)))
32     empt = ''
33     for i in range(5-length):
34         empt = empt+'0'
35
36     return empt+str1
37
38
39 def ftir_extract(filename,init,end):
40
41     filename = filename
42
43     temp_df = pd.read_csv(filename)
44     # nump_df = temp_df.to_numpy()
45     area = area_under(temp_df, init, end)
46     # max_peak = np.max(nump_df[90:120,1])
47     print(area)
```

```

48
49     return area
50
51
52 def function(flowrate_1,v,i):
53
54     #set the pumps with the flowrate as the desired flowrate for the function
55
56     fr_1 = flowrate_1 #ml/min
57     pump_1.write(('irate '+str(fr_1)+' ml/min\r\n').encode())
58
59     time.sleep(0.1)
60
61     #set the com port for potentiostat and set the voltage and current
62     vol = v
63     curr = i
64     port.write(('VOLT '+str(vol)+'\r\n').encode()) #to change the voltage we need to use "VOLT 1" command
65     port.write(('CURR '+str(curr)+'\r\n').encode()) #to change the current we need to use "CURR 1" command
66
67
68     #pumps run
69     pump_1.write((b'irun\r\n'))
70     time.sleep(0.1)
71     time.sleep(1000)
72
73 def function2(flowrate_1,v,i):
74     time.sleep(1400)
75
76     files = [f for f in listdir(mypath) if isfile(join(mypath, f))]
77
78     val = 0
79
80     #change wavelengths as per product here.
81     f_row=15 #first row of range for wavelength as per IR CSV
82     l_row=25 #last row of range for wavelength as per IR CSV
83
84     val += ftir_extract(files[-1],f_row,l_row)
85     val+=ftir_extract(files[-2],f_row,l_row)
86     val+=ftir_extract(files[-3],f_row,l_row)
87
88     avg_val = val/3
89
90     return avg_val
91
92
93 #step 4:grab the line 93 and f9
94 from skopt.optimizer import Optimizer

```

```

95
96
97 #step5:in line 96 we have to define the range that (flowrate,voltage,current) (from,to) and after (anytime) applying changes you need to grab the line 96 and f9
98 #flowrates are in ml/min, voltage in V, Current in Ampere
99 bounds = [(0.05,0.150),(10.0,14.5),(4.9,5.0)]
100
101
102 #step 6: grab the line 100 and f9
103 opter =Optimizer(bounds,base_estimator='gp',n_initial_points=3,acq_func="EI",random_state=np.random.randint(3326))
104
105
106 #step7: to selecte number of the cycles that you have to do the experiment and then grab the line 104 to 121 and f9: the closed loop experimentation is initiated
107 number_of_cycles = 22
108 results = []
109 flowrates_1 = []
110 vs = []
111 currents = []
112
113 product_wavelength=True #set to true if product wavelengths being monitored
114
115 if product_wavelength == True:
116     val=1
117 else:
118     val=-1
119
120
121 # Step 8: Test Tubes on Printer
122 USE_PRINTER = True
123 REST_HEIGHT = 200
124 X_HOME = -5
125 Y_HOME = 20
126 Z_HOME = 175
127 DEFAULT_PUMP_TIME="1"
128 # Distance between test tubes
129 X_SPACING=20
130 Y_SPACING=20
131 # Number of test tubes
132 X_ROWS = 11
133 Y_COLUMNS = 4
134
135 def send_cmd(cmd):
136     print(cmd)
137     printer.write(f"{cmd}\n".encode("ASCII"))
138
139 def move(x=None, y=None, z=None):

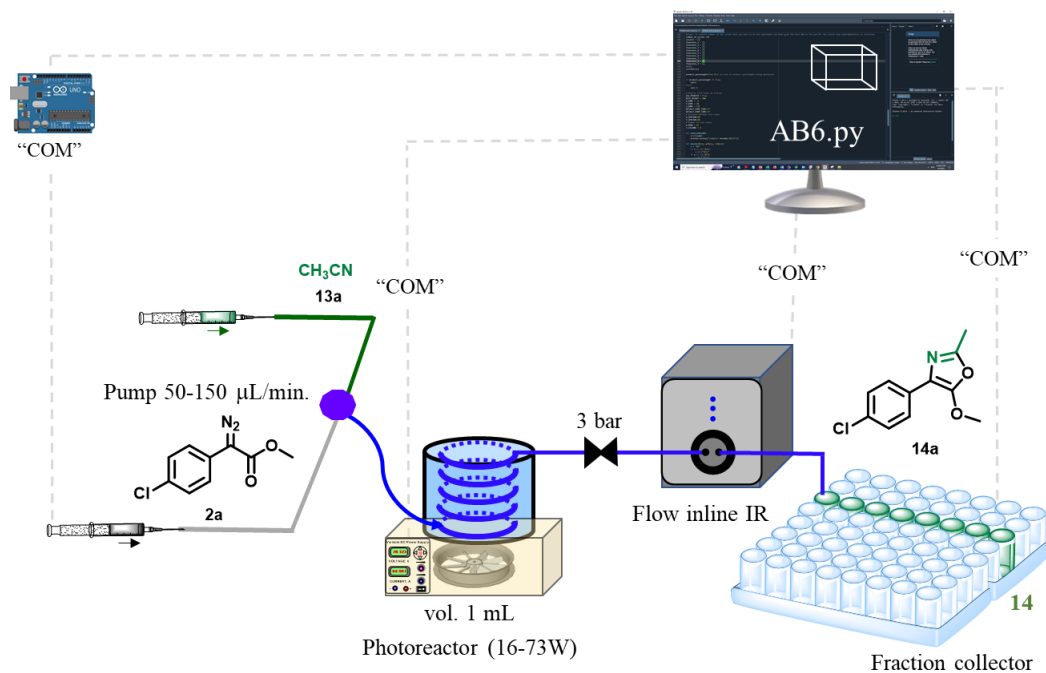
```

```

140 s = "G0"
141 if x is not None:
142     s += f"X{x}"
143 if y is not None:
144     s += f"Y{y}"
145 if z is not None:
146     s += f"Z{z}"
147
148 s+= "F5000"
149 send_cmd(s)
150
151 def printer_positions():
152     for j in range(Y_COLUMNS):
153         for i in range(X_ROWS):
154             if j%2==1:
155                 yield (X_HOME + (X_ROWS - 1 - i) * X_SPACING, Y_HOME + j * Y_SPACING, Z_HOME)
156             else:
157                 yield (X_HOME + i * X_SPACING, Y_HOME + j * Y_SPACING, Z_HOME)
158
159 # Run this.
160 tube_location = list(printer_positions())
161
162
163 for i in range(number_of_cycles):
164     move(*tube_location[2*i])
165     asked = opter.ask()
166     function(asked[0],asked[1],asked[2])
167     move(*tube_location[2*i+1])
168     told = function2(asked[0],asked[1],asked[2])
169
170     print(f"area under the curve in the round {i:.2f} = {told: .2f}")
171     opter.tell(asked, -told*val)
172     results.append(told)
173     flowrates_1.append(asked[0])
174     vs.append(asked[1])
175     currents.append(asked[2])
176
177     dict1 = {"flowrate_1":flowrates_1,"voltages":vs, "currents":currents,"area-results":results}
178     df2 = pd.DataFrame(dict1)
179     df2.to_csv("output round "+str(i)+".csv")
180
181
182
183 pump_1.write(b'stop\r\n')
184
185
186

```

Table S14. Auto-optimization table of carbene insertion reaction (N-C-O bond formation, oxazole).



No. of experiments	Flow rate mL/min	Voltage (V)	Light intensity (W)	area-results	Yield (%)	Space time yield (g mL ⁻¹ h ⁻¹)
1	0.060718	12.40142	41.056	1.155747	66.97147	0.027204
2	0.10674	10.77376	22.862	0.55897	32.39033	0.02313
3	0.107843	10.36182	18.909	0.267647	15.50919	0.011189
4	0.108263	10.79726	23.103	0.543014	31.46576	0.02279
5	0.099346	10.72808	22.398	0.541964	31.40494	0.020873
6	0.068879	10.66267	21.772	0.510561	29.58525	0.013633
7	0.060402	14.4994	69.755	1.570414	91.00002	0.036772
8	0.149805	14.34272	67.192	1.231955	71.38748	0.071544
9	0.053698	14.13864	64.004	1.410356	81.72518	0.029359
10	0.145992	13.77436	58.806	1.178274	68.27688	0.066685
11	0.129902	14.38545	67.638	1.292653	74.90471	0.065096
12	0.15	14.44019	68.469	1.252034	72.55097	0.072805
13	0.13959	12.22695	38.597	0.937301	54.31332	0.050721
14	0.147351	14.00744	62.093	1.237856	71.72945	0.07071
15	0.070144	14.49956	69.252	1.496397	86.71099	0.04069

16	0.15	12.64771	44.051	1.159221	67.17278	0.067408
17	0.141495	14.42064	16.392	0.123688	7.167289	0.006785
18	0.15	14.47181	68.795	1.170652	67.83517	0.068073
19	0.05	14.30231	66.266	1.542238	89.36728	0.029893
20	0.060666	14.03974	62.491	1.458827	84.53391	0.034309
21	0.146306	13.80666	59.163	1.196568	69.33694	0.067866
22	0.053558	13.26608	51.711	1.147437	66.48996	0.023824
23	0.101353	12.36236	40.263	0.970972	56.26446	0.03815
24	0.111683	10.08164	68.24	1.389288	80.50436	0.06015
25	0.050659	14.27468	66.303	1.43998	83.44181	0.028279
26	0.05	10.52857	20.456	0.350086	20.28625	0.006786
27	0.05	15	73.955	1.561132	90.46212	0.03026
28	0.15	13.76539	58.639	1.178978	68.31763	0.068557
29	0.063485	10.41661	19.403	0.406903	23.57859	0.010014
30	0.139424	10.02272	15.941	0.291171	16.87234	0.015738
31	0.062383	14.73195	70.464	1.57821	91.45174	0.038167
32	0.082665	14.57888	70.718	1.457362	84.44903	0.046703
33	0.064109	14.40021	67.925	1.564911	90.68111	0.038892
34	0.146834	10.72916	22.411	0.51166	29.64892	0.029125
35	0.05799	14.59723	70.971	1.55543	90.13171	0.034967
36	0.077781	14.85116	74.015	1.550575	89.85043	0.046754
37	0.149019	10.96034	24.693	0.691132	40.04866	0.039926
38	0.06201	14.99412	73.149	1.543011	89.4121	0.037092
39	0.09372	13.86823	60.073	1.510841	87.54798	0.054892
40	0.060974	14.27468	66.303	0.979244	56.74375	0.023147
41	0.064852	14.76182	73.203	1.320306	76.50714	0.033193
42	0.13607	13.16311	50.48	0.771156	44.68577	0.040678
43	0.053277	10.40976	19.38	0.261369	15.14544	0.005398
44	0.147724	10.71269	22.349	0.495395	28.70643	0.02837

Reaction condition: compound **2a** (0.1 M in DCE); reagent **13a** (0.2 M in DCE); 1 mL reactor volume.

Graph:

We have plotted 3D graph of optimization table S14, we have taken flow rate on X-axis, blue light intensity (watt) on Y-axis and product yield on Z axis.

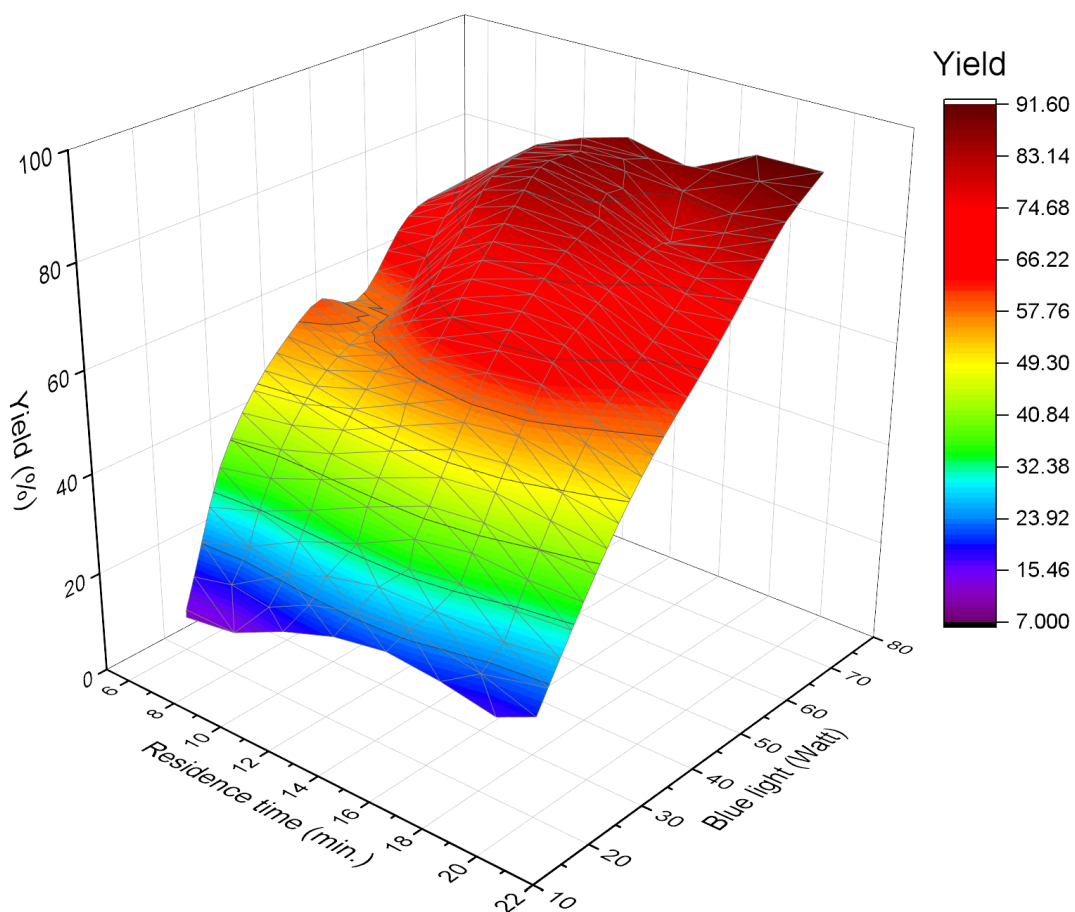


Figure S45. AI based system to auto-optimize and navigate this complexity and identify the optimal conditions for the photo activated [3+2] cycloaddition reaction. Compound **2a** (0.1 M in DCE); reagent **13a** (0.2 M in DCE); 1 mL reactor volume.

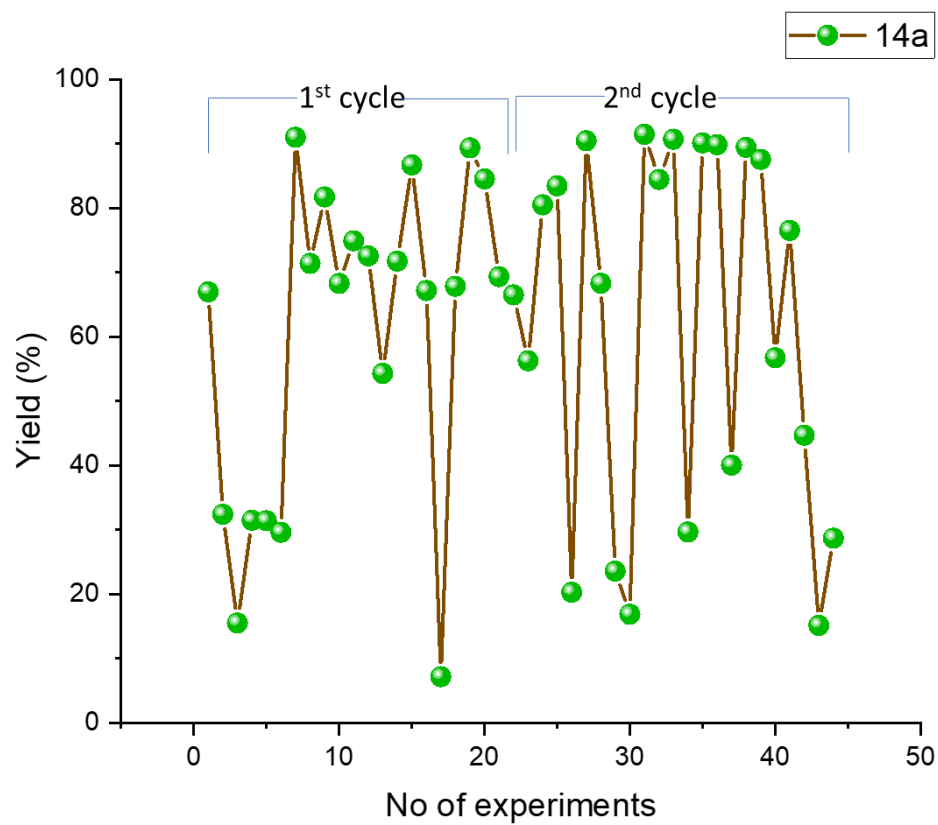


Figure S46. 2D graph between no of experiments performed versus yield (%) for the AI based auto-optimization of photo activated [3+2] cycloaddition reaction. Compound **2a** (0.1 M in DCE); reagent **13a** (0.2 M in DCE); 1 mL reactor volume.

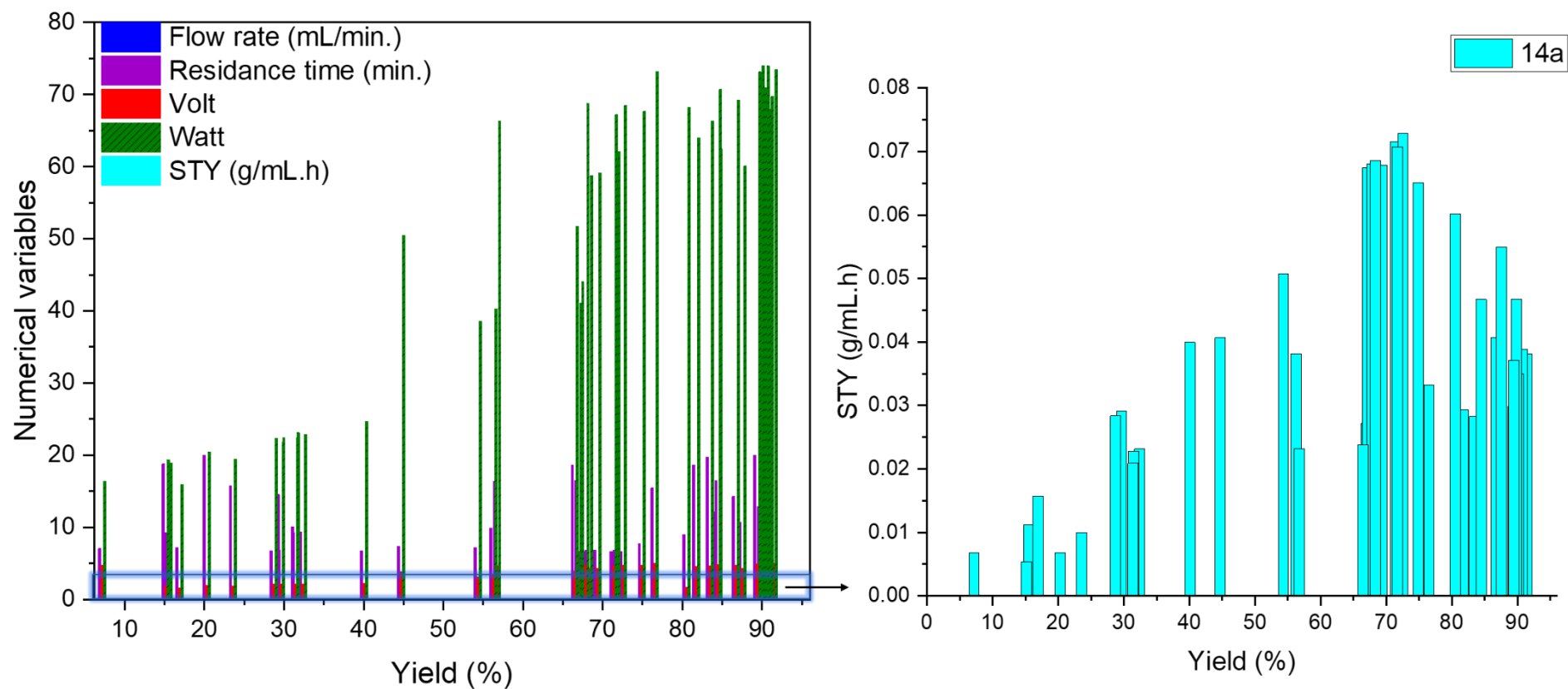


Figure S47. AI based system navigated multi-numerical variable complexity and identify the optimal conditons for the photo activated [3+2] cycloaddition reaction. Compound **2a** (0.1 M in DCE); reagent **13a** (0.2 M in DCE); 1 mL reactor volume.

S.8.3. General procedure for AI optimized carbene insertion into the triple bonds of $C\equiv N$ for synthesis of compound 14a.

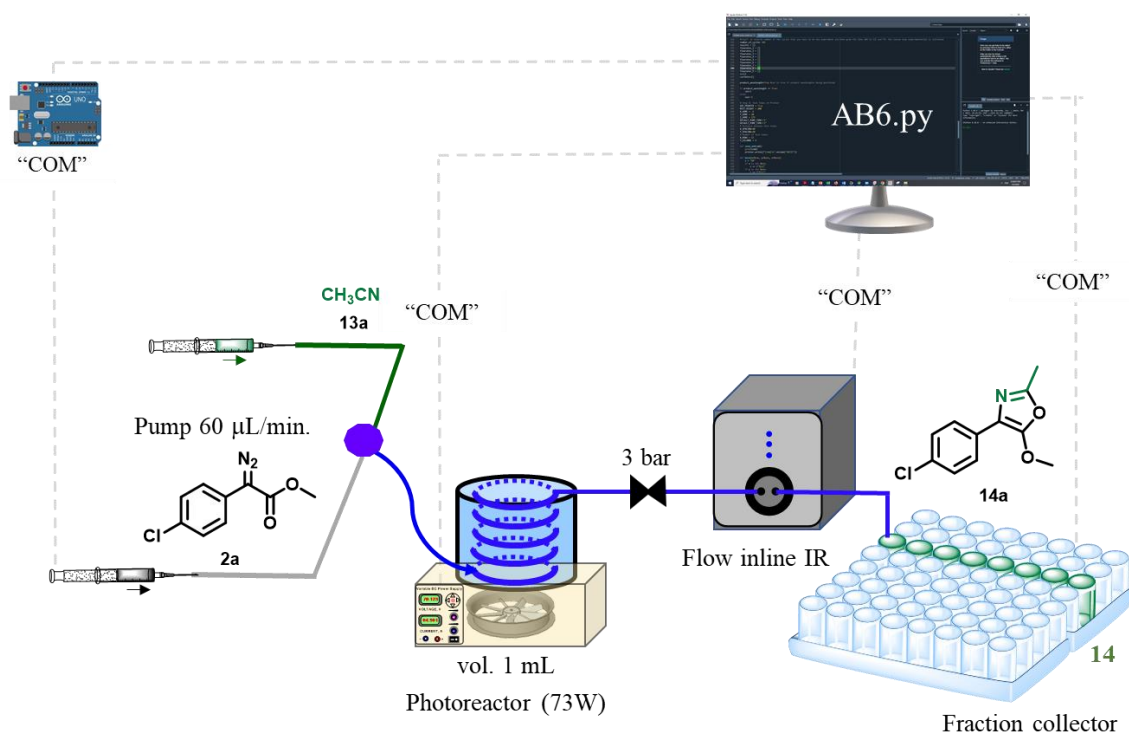


Figure S48. Schematic presentation of continuous flow for the synthesis, of compound **14a**.

To prepare the stock solution of compound **2a** (2.20 g, 0.01 mol) was dissolved in DCE (100 mL) and connected with pump. Another pump was connected with the reagent **13a** (1.09 mL, 0.026 mol) dissolved in DCE (100 mL). These two pump out-put is further connected with the T₁-mixer and pumps were running with the flow rate 31 µL/min. each to maintain the stoichiometry and then passed through PFA tubular reactor (inner diameter = 1 mm, length = 1.3 m, volume = 1 mL) under blue light (73.464 W) exposure. The room temperature of the reactor was controlled by fan attached to the bottom of the photochemical reactor. The out-put of the tubular reactor was connected with spring based back pressure regulator (~3 bar) to maintain the evaporation. First one hour of the product mixture was discarded and next 5 h of the product mixture [18.6 mL; **2a** (0.195 g)] was collected in HPLC bottle. The organic EtOAc phase was concentrated under reduced pressure to give the product **13a** (0.190g in 5 h, 91%)

as a colourless oil. **¹H NMR (500 MHz, CDCl₃)** δ 7.70 – 7.67 (m, 2H), 7.32 – 7.30 (m, 2H), 4.00 (s, 3H), 2.38 (s, 3H). **¹³C NMR (126 MHz, CDCl₃)** δ 154.28, 151.75, 131.56, 130.07, 129.22, 128.56, 126.01, 113.27, 59.84, 14.11. **IR (ν_{max}):** 3019.92, 1741.94, 1646.73, 1214.78, 1013.60, 745.47 cm⁻¹. **HRMS (ESI):** *m/z* calcd for C₁₁H₁₁ClNO₂ [M+H]⁺ 224.0473, found 224.0451. Verified the analytical data with those reported in the literature.²¹

Space time yield in previous reported batch process ²¹

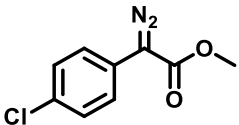
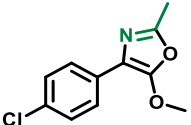
Productivity = 39.8 mg in 24 h

$$\text{Productivity (per h)} := \frac{0.39}{24} = 0.0016 \text{ g/h}$$

$$\text{Space time yield} := \frac{\text{productivity}}{\text{volume of reactor}}$$

$$= \frac{0.0016}{1} = 0.0016 \text{ g mL}^{-1} \text{ h}^{-1}$$

Table S15. Comparative result for the carbene insertion (N-C-O bond formation, oxazole).

Entry	Compound 2a	Reagent 13a	Product 14a	Comparative result
1		CH ₃ CN		91.4%, 16.6 min. (our study) Blue LED, RVC (+), Pt (-), 1.5 h, 88% ²² Et ₃ SiCl, Blue LED, 8 h, 83% ²³ Blue LED, 24 h, 89% ²¹

S9. Case study-7: Carbene insertion (C=C bond formation, fumarate) single objective multi-variant auto-optimization.

S9.1 Background collection

Sample Preparation: We prepare 0.5 M stock solution of compound **2a** in DCE, 0.5 M stock solution of reagent **15** in DCE and 0.5 M stock solution of product **16a** in DCE. The three of the solution taken in three different syringes.

To conduct our analysis, we utilized an In-line FTIR system. The experimental protocol involved the initial introduction of the solvent, DCE, for a duration of 10 min. using a syringe pump, during which data was recorded. Subsequently, a previously prepared stock solution of compound **2a** was similarly introduced into the In-line FTIR for 10 min. This process was repeated for the reaction product **16a** and reagent **15**, each pumped for 10 min. Upon collecting all relevant data, we identified the signature peak present in the product. Using this peak as a reference, we employed a Bayesian optimizer to fine-tune and optimize the reaction conditions for improved results.

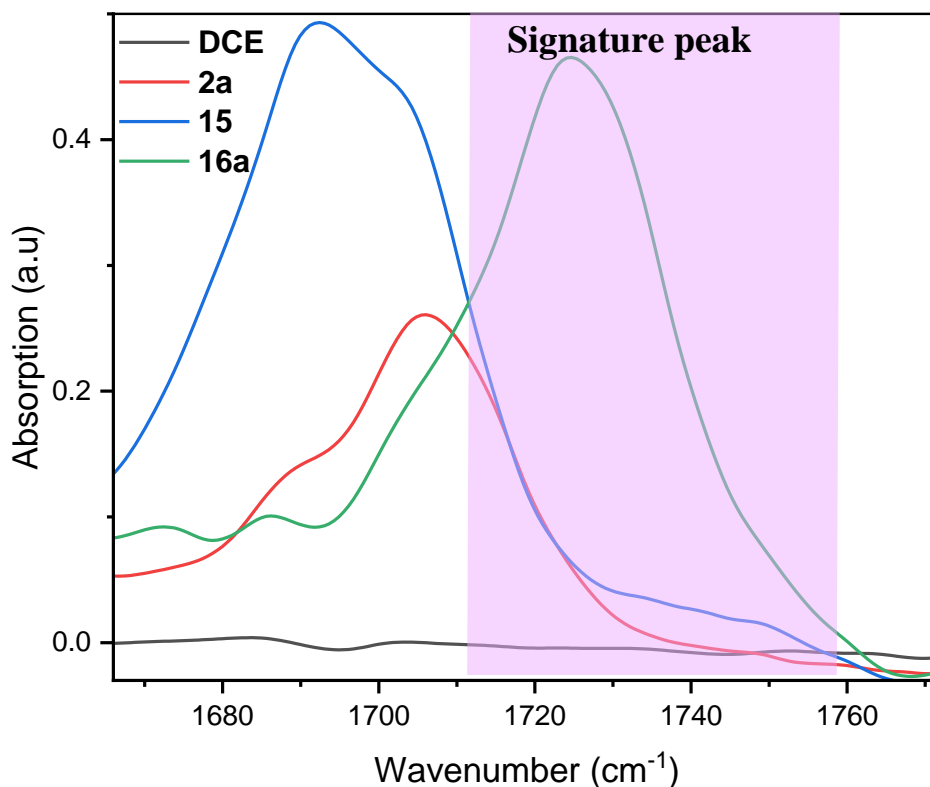


Figure. S49. In-line IR background analysis of DCE, 0.5 M stock solution of **1a**, **15a**, and **16a** in DCE.

The distinctive shifting peak associated with product (**16a**) aligning within the range of 1715 to 1760 cm^{-1} . This specific peak, falling within this range, is designated as the signature peak for our analysis. During the optimization of the reaction through Bayesian methods, this signature peak serves as a pivotal reference point. We utilize it to calculate the area under the curve in the In-line FT-IR spectra, providing a quantitative measure of the area in the forthcoming reaction mixture.

S8.2. General procedure for the auto-optimized synthesis of compound 16a.

Initial steps involved the preparation of stock solutions for compound **2a** (0.1 M in DCE) and reagent **15** (0.1 M in DCE), each filled into separate syringes and connected to a syringe pump.

The solutions were then directed through a PFA tubular reactor (inner diameter = 1 mm, length = 1.3 m, volume = 1 mL) surrounded by a cylindrical-shaped blue LED light source. Once the solution setup was complete, input ranges for variable flow rates, voltages, and current were specified in the Python code designed for the reaction. The Bayesian optimizer systematically explored these varying reaction conditions, aiming to achieve maximum yield. The optimization process involved running 40-45 experiments and the results were tabulated in the optimization table (Table S16).

S8.2.1. Python code for optimized condition for C=C bond formation, fumarate

```
ab7.py
1 from os import listdir
2 from os.path import isfile, join
3 import serial
4
5 import numpy as np
6 import pandas as pd
7 import time
8 from scipy.integrate import trapz
9
10 #step1: be sure to the address of the files that the ftir data is exported is matching to line 11 (mypath)
11 mypath = "C:\\Users\\Admin\\Desktop\\ruchi\\Exp 2024-03-06 13-11"
12 onlyfiles = [f for f in listdir(mypath) if isfile(join(mypath, f))]
13
14 #step2: make sure that pump and the potentiostat is correctly addressed in the Line 16 and 17
15 pump_1 = serial.Serial("COM17",9600) #Syringe Pump
16 port = serial.Serial("COM1",115200)
17 printer = serial.Serial("COM9", 115200, timeout=1)
18
19 r_1=13.36 #radius of syringe used (50ml,14.25|20ml,9.6|10ml,7.25|5ml,6.03|3ml,4/3|1ml,2.39|60ml,13.36)
20 pump_1_k=(13.36/r_1)**2
21
22 #step 3: grab the lines from 22 to 90 and presss f9
23 def area_under(data,start,end):
24     x = np.flip(data.iloc[start:end,0].to_numpy())
25     y = np.flip(data.iloc[start:end,1].to_numpy())
26     area = trapz(y,x)
27     return np.abs(area)
28
29 def file_namer(num):
30     str1 = str(num)
31     length = int(len((str1)))
32     empt = ''
33     for i in range(5-length):
34         empt = empt+'0'
35
36     return empt+str1
37
38
39 def ftir_extract(filename,init,end):
40
41     filename = filename
42
43     temp_df = pd.read_csv(filename)
44     # nump_df = temp_df.to_numpy()
45     area = area_under(temp_df, init, end)
46     # max_peak = np.max(nump_df[90:120,1])
47     print(area)
```

```

48
49     return area
50
51
52 def function(flowrate_1,v,i):
53
54     #set the pumps with the flowrate as the desired flowrate for the function
55
56     fr_1 = flowrate_1*1000*pump_1_k #mL/min to microliter/min and syringe correction factor
57     send_syr_command(f'fr{fr_1}')
58     time.sleep(0.1)
59
60     #set the com port for potentiostat and set the voltage and current
61     vol = v
62     curr = i
63     port.write(('VOLT '+str(vol)+'\r\n').encode()) #to change the voltage we need to use "VOLT 1" command
64     port.write(('CURR '+str(curr)+'\r\n').encode()) #to change the current we need to use "CURR 1" command
65
66
67     #pumps run
68     time.sleep(0.1)
69     time.sleep(20)
70
71 def function2():
72     time.sleep(20)
73
74     files = [f for f in listdir(mypath) if isfile(join(mypath, f))]
75
76     val = 0
77
78     #change wavelengths as per product here.
79     f_row=11 #first row of range for wavelength as per IR CSV
80     l_row=15 #last row of range for wavelength as per IR CSV
81
82     val += ftir_extract(files[-1],f_row,l_row)
83     val+=ftir_extract(files[-2],f_row,l_row)
84     val+=ftir_extract(files[-3],f_row,l_row)
85
86     avg_val = val/3
87
88     return avg_val
89
90
91 #step 4:grab the line 93 and f9
92 from skopt.optimizer import Optimizer
93
94

```

```

95 def send_syr_command(command):
96     pump_1.write(command.encode() + b'\n') # Send the command to Arduino with a newline character
97     pump_1.flush() # Flush the serial buffer
98
99
100 #step5:in line 96 we have to define the range that (flowrate,voltage,current) (from,to) and after (anytime) appling changes you need to grab the line 96 and f9
101 #flowrates are in ml/min, voltage in V, Current in Ampere
102 bounds = [(0.05,0.15),(10.0,14.5),(0.0,5.0)]
103
104 #step 6: grab the line 100 and f9
105 opter =Optimizer(bounds,base_estimator='gp',n_initial_points=3,acq_func="EI",random_state=np.random.randint(3326))
106
107
108 #step7: to selecte number of the cycles that you have to do the experiment and then grab the line 104 to 121 and f9: the closed loop experimentation is initiated
109 number_of_cycles = 22
110 results = []
111 flowrates_1 = []
112 vs = []
113 currents = []
114
115 product_wavelength=True #set to true if product wavelengths being monitored
116
117 if product_wavelength == True:
118     val=1
119 else:
120     val=-1
121
122
123 # Step 8: Test Tubes on Printer
124 USE_PRINTER = True
125 REST_HEIGHT = 200
126 X_HOME = -5
127 Y_HOME = 20
128 Z_HOME = 175
129 DEFAULT_PUMP_TIME="1"
130 # Distance between test tubes
131 X_SPACING=20
132 Y_SPACING=20
133 # Number of test tubes
134 X_ROWS = 11
135 Y_COLUMNS = 4
136
137 def send_cmd(cmd):
138     print(cmd)
139     printer.write(f"{cmd}\n".encode("ASCII"))
140

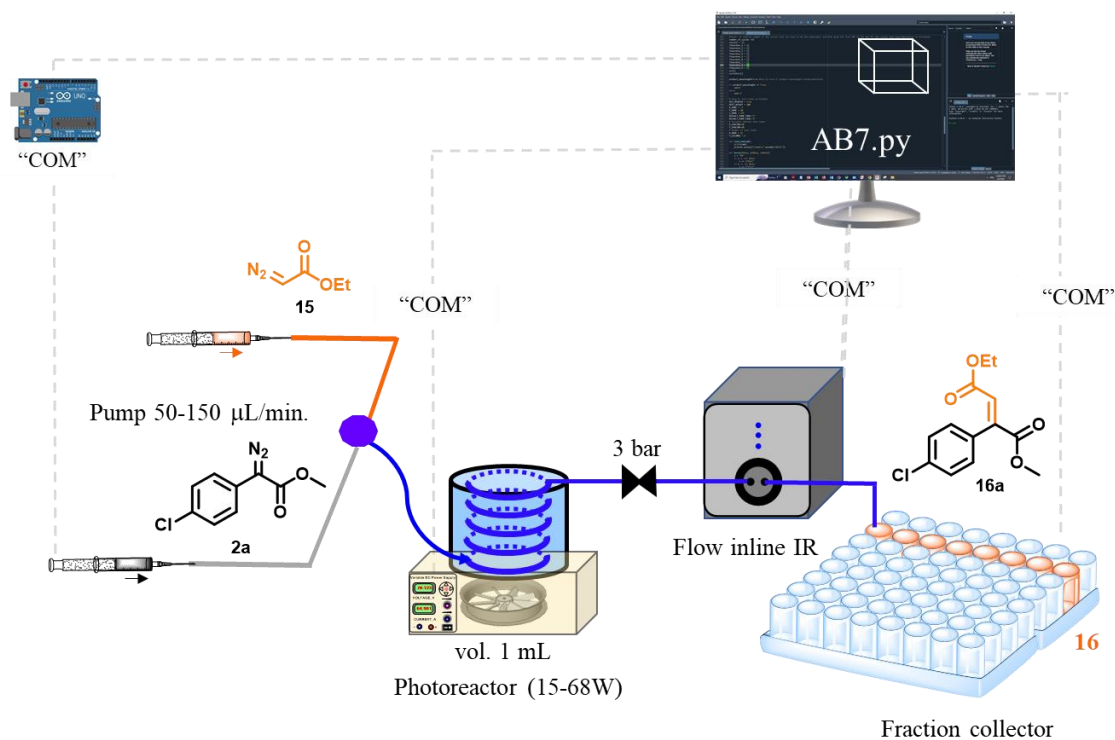
```

```

141 def move(x=None, y=None, z=None):
142     s = "G0"
143     if x is not None:
144         s += f"X{x}"
145     if y is not None:
146         s += f"Y{y}"
147     if z is not None:
148         s += f"Z{z}"
149
150     s += "F5000"
151     send_cmd(s)
152
153 def printer_positions():
154     for j in range(Y_COLUMNS):
155         for i in range(X_ROWS):
156             if j%2==1:
157                 yield (X_HOME + (X_ROWS - 1 - i) * X_SPACING, Y_HOME + j * Y_SPACING, Z_HOME)
158             else:
159                 yield (X_HOME + i * X_SPACING, Y_HOME + j * Y_SPACING, Z_HOME)
160
161 # Run this.
162 tube_location = list(printer_positions())
163
164 time.sleep(2)
165
166 for i in range(number_of_cycles):
167     move(*tube_location[2*i])
168     asked = opter.ask()
169     print (asked[0])
170     function(asked[0],asked[1],asked[2])
171     move(*tube_location[2*i+1])
172     told = function2()
173
174     print(f"area under the curve in the round {i:.2f} = {told: .2f}")
175     opter.tell(asked,-told*val)
176     results.append(told)
177     flowrates_1.append(asked[0])
178     vs.append(asked[1])
179     currents.append(asked[2])
180
181     dict1 = {"flowrate_1":flowrates_1,"voltages":vs, "currents":currents,"area-results":results}
182     df2 = pd.DataFrame(dict1)
183     df2.to_csv("output round "+str(i)+".csv")
184
185
186
187 send_syr_command("stop")    #turns off syringe pump flow

```

Table S16. Auto-optimization table of cross coupling reaction (C=C bond formation, fumarate).



No. of experiments	Flow rate mL/min	Voltage (V)	Light intensity (W)	area-results	Yield (%)	Space time yield (g mL ⁻¹ h ⁻¹)
1	0.114805	11.92934	34.102	0.470179	36.42071	0.033618
2	0.098673	12.30966	38.515	0.854069	66.1574	0.052485
3	0.103371	11.69926	31.712	0.830144	64.3041	0.053443
4	0.053233	14.41624	67.034	1.090795	84.49451	0.036163
5	0.15	12.46267	41.012	0.60424	46.80531	0.056447
6	0.136292	11.89984	34.436	0.627558	48.61151	0.053268
7	0.0808	13.69994	57.96	1.106513	85.71207	0.055682
8	0.070825	11.19236	26.491	0.542513	42.0238	0.02393
9	0.052078	10.03053	15.605	0.349642	27.08376	0.01134
10	0.109072	12.22021	37.699	0.508583	39.39557	0.034548
11	0.100732	12.0365	35.578	0.421314	32.63555	0.026431
12	0.114871	11.90975	34.56	0.456292	35.345	0.032643
13	0.148623	14.25763	65.083	1.000345	77.48811	0.092593

14	0.053555	10.00233	15.532	0.192328	14.89802	0.006415
15	0.05	12.69937	44.027	0.739746	57.30175	0.023035
16	0.129758	11.12454	26.086	0.44318	34.32936	0.035814
17	0.123543	12.8464	45.873	0.795025	61.58378	0.06117
18	0.142808	11.61169	31.302	0.302434	23.42699	0.026898
19	0.080479	11.63094	31.389	0.381501	29.5516	0.019121
20	0.081458	14.41624	67.034	1.148959	88.99998	0.058288
21	0.065678	12.35186	39.644	0.714388	55.33753	0.029221
22	0.068097	12.28934	39.054	0.431773	33.44574	0.018311
23	0.094075	10.32873	18.363	0.20385	15.79052	0.011943
24	0.054858	12.70035	44.104	0.90043	69.74862	0.030763
25	0.132587	14.37057	66.907	0.816223	63.22577	0.067399
26	0.15	10	15.53	0.124816	9.668397	0.01166
27	0.149923	11.35554	28.524	0.250264	19.38581	0.023367
28	0.15	10	16.018	0.193638	14.99947	0.018089
29	0.15	10	16.018	0.19419	15.04221	0.018141
30	0.081458	14.32353	66.382	1.09276	84.64676	0.055437
31	0.14972	10.04417	15.585	0.164063	12.70858	0.015298
32	0.124635	14.5	68.203	0.653962	50.6568	0.050762
33	0.050661	10.01346	15.52	0.189311	14.6643	0.005973
34	0.148887	14.32353	66.382	0.96474	74.73017	0.089456
35	0.059379	10	15.499	0.156635	12.13321	0.005792
36	0.050761	11.45305	29.297	0.368252	28.5253	0.011642
37	0.052621	12.00522	35.367	0.75496	58.48026	0.024741
38	0.090012	10.00458	15.458	0.35354	27.38574	0.019819
39	0.079081	14.4398	67.541	1.03432	80.11992	0.050941
40	0.129892	10.00877	15.502	0.417781	32.3619	0.033797
41	0.080521	14.47602	68.587	1.140182	88.32009	0.057177
42	0.078882	10.02039	14.682	0.251696	19.4967	0.012365
43	0.094613	14.48282	68.427	1.068802	82.79091	0.062978
44	0.094114	14.49842	68.595	1.089417	84.38781	0.063854

Reaction condition: compound **2a** (0.1 M in DCE); reagent **15** (0.1 M in DCE); 1 mL reactor volume.

Graphs:

We have plotted 3D graph of optimization table S16, we have taken flow rate on X-axis, blue light intensity (watt) on Y-axis and product yield on Z axis.

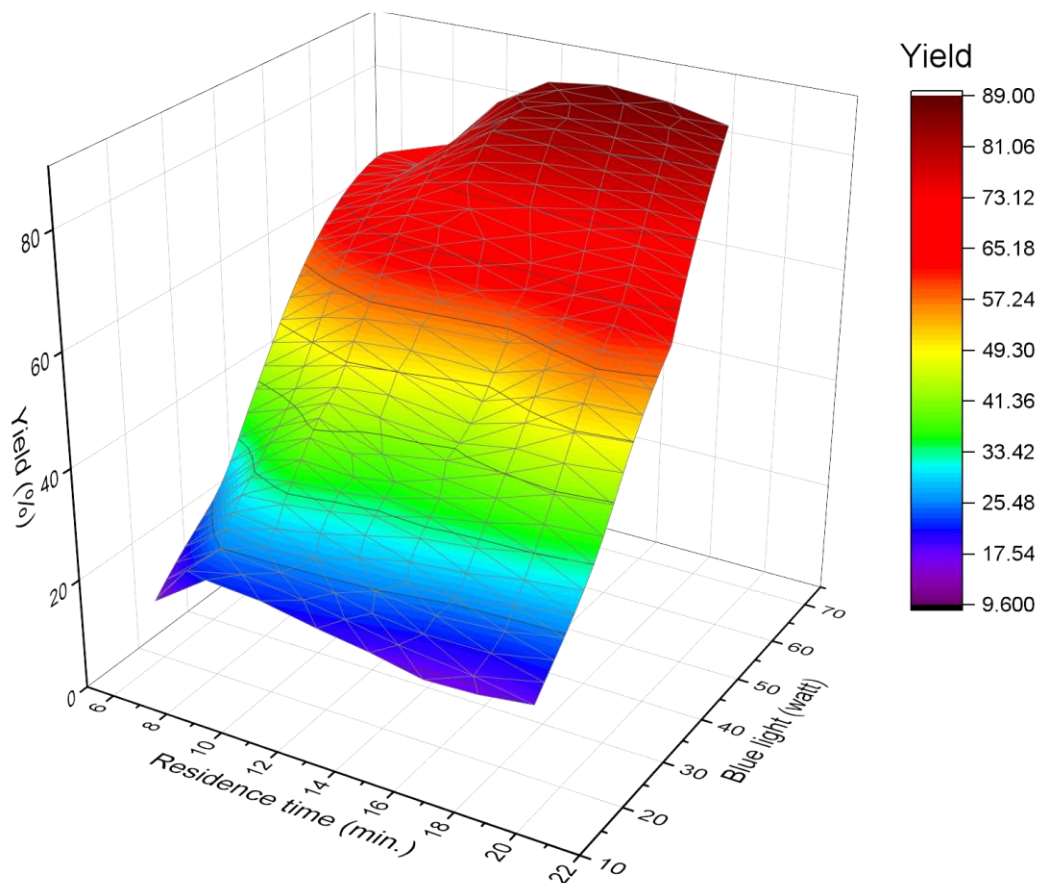


Figure S50. AI based system to auto-optimize and navigate this complexity and identify the optimal conditions for the photo activated cross-coupling reaction. Compound **2a** (0.1 M in DCE); reagent **15** (0.1 M in DCE); 1 mL reactor volume.

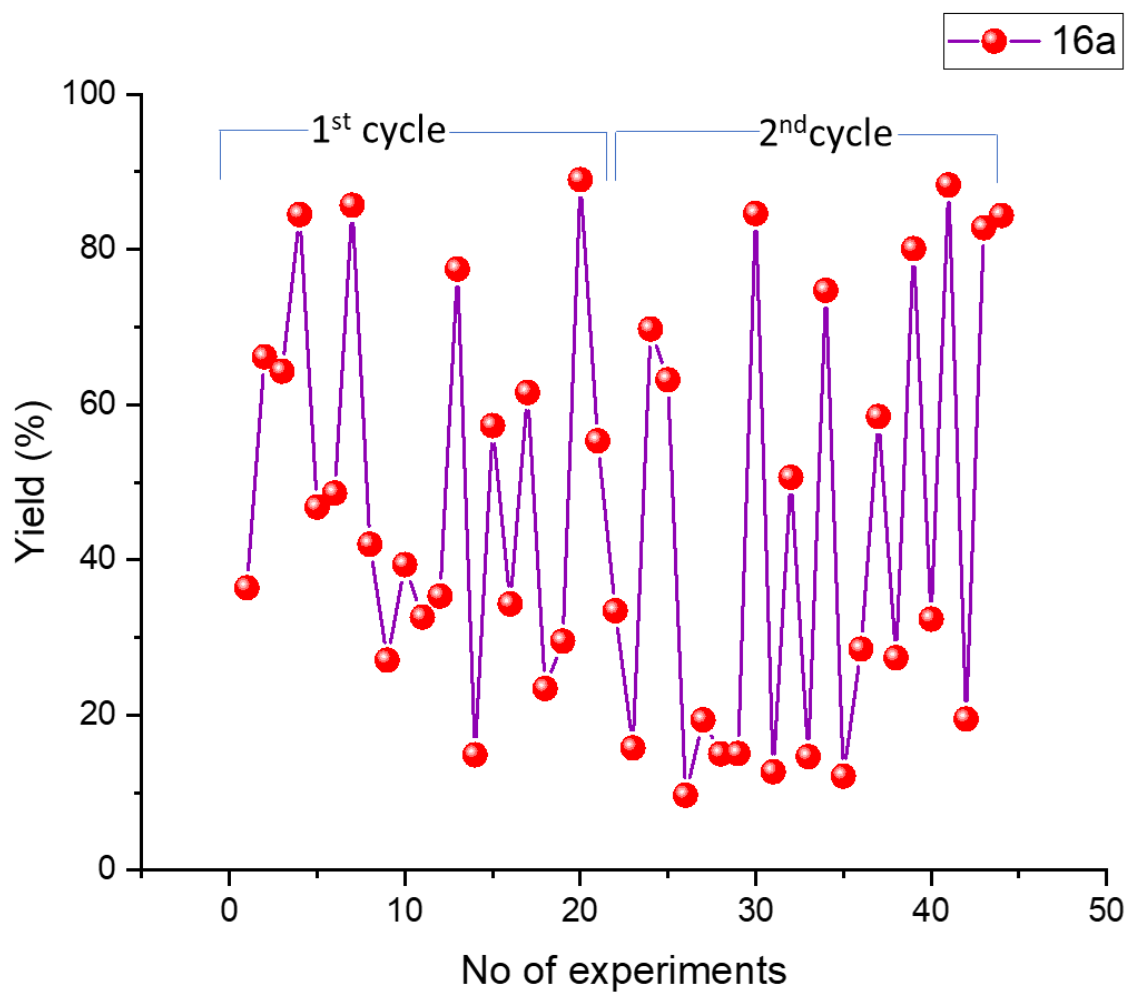


Figure S51. 2D graph between No of experiments performed versus Yield (%) for the AI based auto-optimization for the photo activated cross-coupling reaction. Compound **2a** (0.1 M in DCE); reagent **15** (0.1 M in DCE); 1 mL reactor volume.

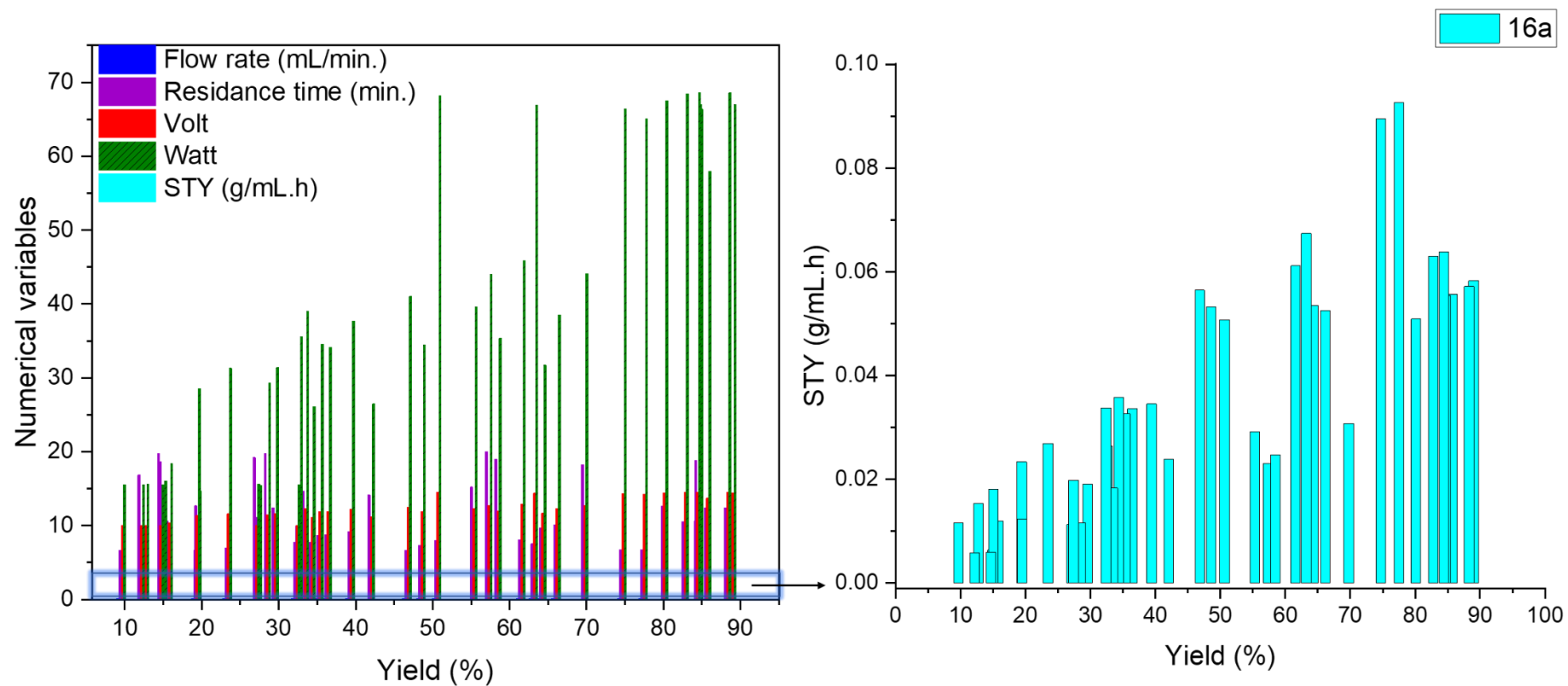


Figure S52. AI based system navigated multi-numerical variable complexity and identify the optimal condition for the photo activated cross-coupling reaction. Compound **2a** (0.1 M in DCE); reagent **15** (0.1 M in DCE); 1 mL reactor volume.

S.8.3. General procedure for AI optimized cross coupling reaction of ethyl diazoacetate and aryldiazoester for synthesis of compound 16a.

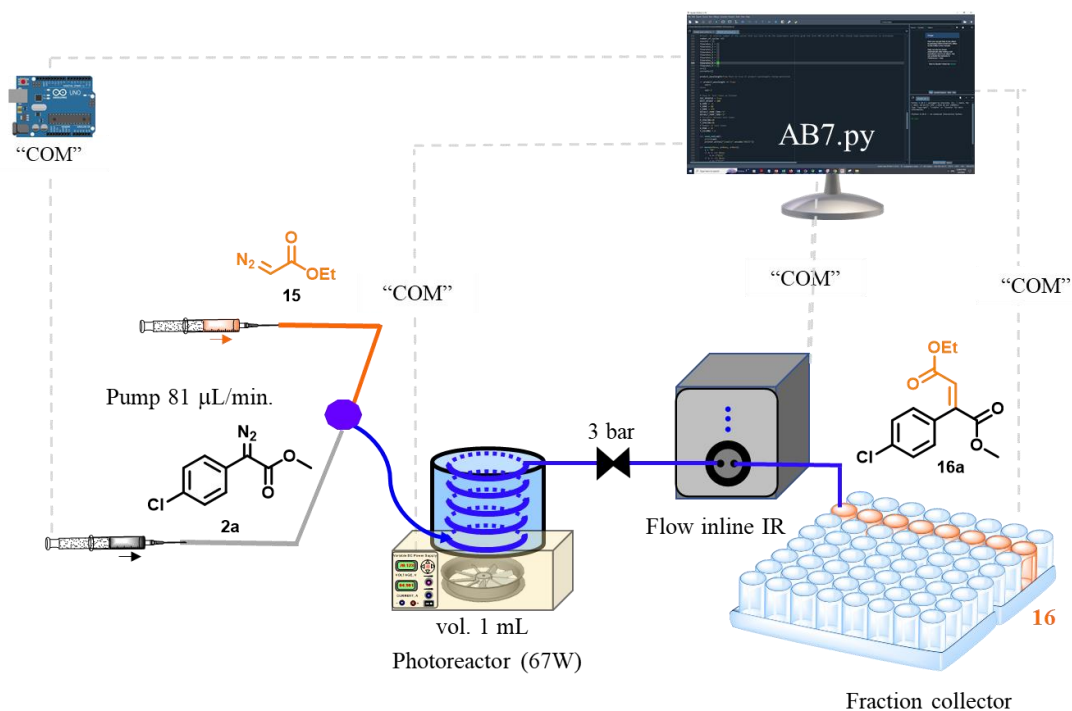


Figure S53. Schematic presentation of continuous flow for the synthesis, of compound **16**.

To prepare the stock solution of compound **2a** (2.20 g, 0.01 mol) was dissolved in DCE (100 mL) and connected with pump. Another pump was connected with the reagent **15** (1.32 mL, 0.01 mol) dissolved in DCE (100 mL). These two pump out-put is further connected with the T₁-mixer and pumps were running with the flow rate 40 µL/min.each to maintain the stoichiometry and then passed through PFA tubular reactor (inner diameter = 1 mm, length = 1.3 m, volume = 1 mL) under blue light (67.034 W) exposure. The room temperature of the reactor was controlled by fan attached to the bottom of the photochemical reactor. The out-put of the tubular reactor was connected with spring based back pressure regulator (~3 bar) to maintain the evaporation. First one hour of the product mixture was discarded and next 5 h of the product mixture [24.3 mL; **2a** (0.255 g)] was collected in HPLC bottle. The organic EtOAc phase was concentrated under reduced pressure to give the product **16a** (0.29 g in 5 h, 89%) as

a colourless oil. **¹H NMR (400 MHz, CDCl₃)** δ 7.35–7.32 (m, 2H), 7.21–7.16 (m, 2H), 7.03 (s, 1H), 4.06 (q, *J* = 7.1 Hz, 2H), 3.79 (s, 3H), 1.10 (t, *J* = 7.1 Hz, 3H). **¹³C NMR (126 MHz, CDCl₃)** δ 166.34, 164.93, 142.65, 134.63, 132.35, 130.36, 129.81, 128.06, 61.02, 52.97, 13.84. **IR (ν_{max}):** 2985.96, 1719.69, 1439.07, 1242.01, 832.82 cm⁻¹. **HRMS (ESI):** *m/z* calcd for C₁₃H₁₄ClO₄ [M+H]⁺ 269.0575, found 269.0590. Verified the analytical data with those reported in the literature.²⁴

Space time yield in previous reported batch process²⁴

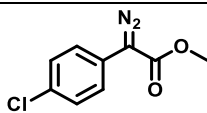
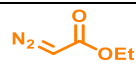
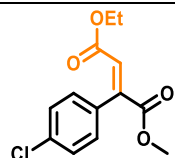
Productivity = 41.5 mg in 12 h

$$\text{Productivity (per h)} := \frac{0.041}{12} = 0.0034 \text{ g/h}$$

$$\text{Space time yield} := \frac{\text{productivity}}{\text{volume of reactor}}$$

$$= \frac{0.0034}{2} = 0.0017 \text{ g mL}^{-1} \text{ h}^{-1}$$

Table S17. Comparative result for the carbene insertion (C=C bond formation, fumarate).

Entry	Compound 2a	Reagent 15a	Product 16a	Comparative result
1				89%, 12.3 min. (our study) Blue LED, 12 h, 93% ²⁴ Artificial metalloenzymes, 22 h, 68% ²⁵

S10. Case study-8: On demand chemical synthesis.

After individually optimizing the experimental conditions and validating our codes for carbene insertion reactions with varied nucleophiles photo chemically, we aimed to integrate all the python codes (AB1, AB2, AB3, AB4, AB5, AB6, AB7) with optimized conditions. We drew inspiration from the natural grafting technique observed in hibiscus plants, where the root remains the same, but a portion of one plant is placed into the branch of another, resulting in the growth of the same kind of flower with different colours. Similarly, in our "digichemtree" approach, we maintain the same starting material, 4-chloro phenyl diazoester (**2a**), but graft our tree with varied nucleophiles (**3a**, **5a**, **7a**, **9a**, **11a**, **13**, **15**) to obtain our respective carbene-inserted products photo chemically. The neural network remains unchanged from our individual experiments, but we've now integrated a total of 9 pumps to flow the starting material and nucleophiles. The outlets of these pumps are connected to an 8-port valve for mixing purposes. The resulting stock solution then passes through a 1 mL PFA tubular reactor with a blue LED and tunable power supply. For fraction collection, we've utilized our tailor-made 3D printer module. This integrated system employs 4 syringe pumps and 5 HPLC pumps. As a model reaction, we've conducted 7 chemical transformations with 7 different nucleophiles.

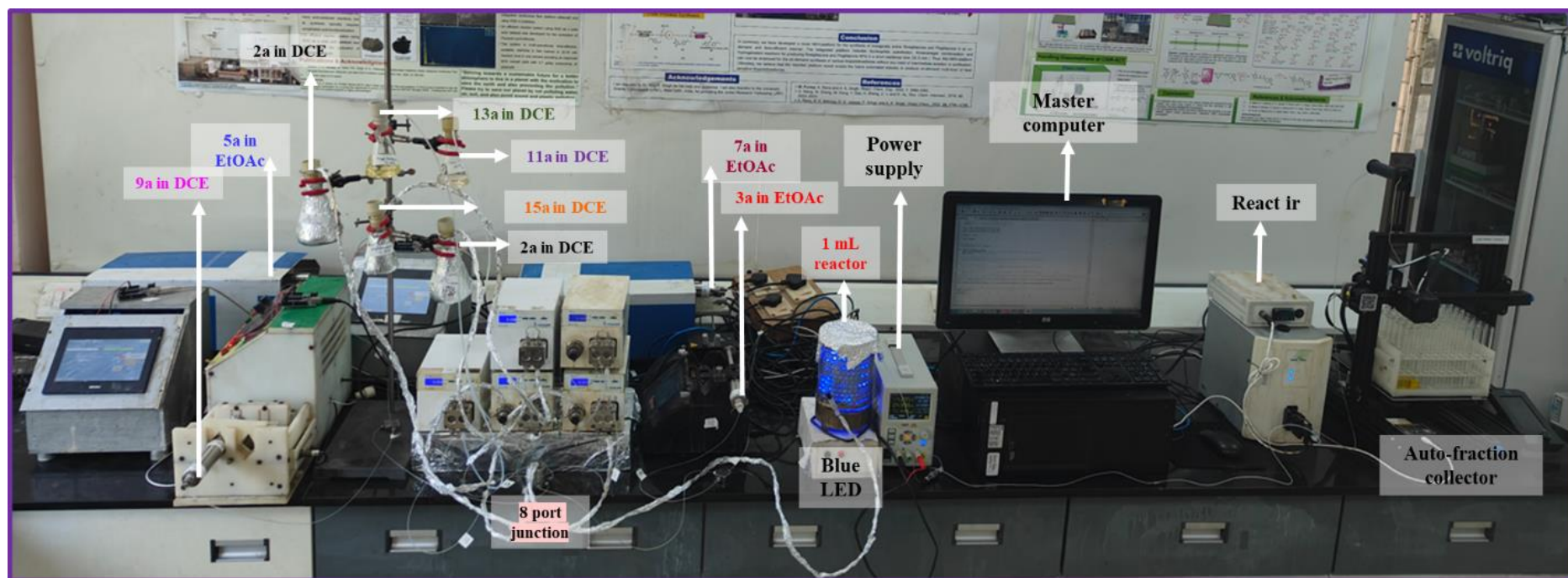


Figure S54. Actual picture of the setup.

S10.1. Python code for integrated optimized condition for carbene insertion reactions.

AB1.PY CODE (for benzoic acid)

```
AB1.py x
1
2 """
3 Created on Wed Aug 30 14:48:13 2023
4
5 @author: Admin
6 """
7
8
9 from os import listdir
10 from os.path import isfile, join
11 import serial
12
13 import numpy as np
14 import pandas as pd
15 import time
16 from scipy.integrate import trapz
17
18 #step1: be sure to the address of the files that the ftir data is exported is matching to line 11 (mypath)
19 mypath = "C:\\Users\\Admin\\Desktop\\ruchi\\Exp 2024-04-11 15-26"
20 onlyfiles = [f for f in listdir(mypath) if isfile(join(mypath, f))]
21
22
23 #step2: make sure that pump and the potentiostat is correctly addressed in the Line 16 and 17
24 pump_1 = serial.Serial("COM10",9600) #hwSyringe Pump
25 #pump_2 = serial.Serial("COM16",9600) #Syringe Pump
26 #pump_3 = serial.Serial("COM19",9600) #Syringe Pump
27 #pump_4 = serial.Serial("COM15",9600) #Syringe Pump
28 #pump_5 = serial.serial_for_url("COM29",9600) #HPLC Pump LAN
29 #pump_6 = serial.serial_for_url("socket://169.254.225.18:10001",9600)
30 port = serial.Serial("COM1",115200)
31 #pump_7 = serial.Serial("COM28",9600) #HPLC Pump
32 #pump_8 = serial.Serial("COM30",9600) #HPLC Pump
33 pump_9 = serial.Serial("COM5",9600) #HPLC Pump
34 printer = serial.Serial("COM14", 115200, timeout=1)
35
36 r_1=14.25 #radius of syringe used (50ml,14.25|20ml,9.6|10ml,7.25|5ml,6.03|3ml,4/3|1ml,2.39|60ml,13.36)
37 r_2=14.25
38 r_3=14.25
39 r_4=14.25
40 pump_1_k=(14.25/r_1)**2
41 #pump_2_k=(14.25/r_2)**2
42 #pump_3_k=(14.25/r_3)**2
43 #pump_4_k=(14.25/r_4)**2
44
45 #step 3: grab the lines from 22 to 90 and presss f9
46 def area_under(data,start,end):
```

```

47 x = np.flip(data.iloc[start:end,0].to_numpy())
48 y = np.flip(data.iloc[start:end,1].to_numpy())
49 area = trapz(y,x)
50 return np.abs(area)
51
52 def file_namer(num):
53     str1 = str(num)
54     length = int(len((str1)))
55     empt = ''
56     for i in range(5-length):
57         empt = empt+'0'
58
59     return empt+str1
60
61
62 def ftir_extract(filename,init,end):
63
64     filename = filename
65
66     temp_df = pd.read_csv(filename)
67     # nump_df = temp_df.to_numpy()
68     area = area_under(temp_df, init, end)
69     # max_peak = np.max(nump_df[90:120,1])
70     print(area)
71
72     return area
73
74
75 def function(flowrate_1,flowrate_2,flowrate_3,flowrate_4,flowrate_5,flowrate_6,flowrate_7,flowrate_8,flowrate_9,v,i):
76
77     #set the pumps with the flowrate as the desired flowrate for the function
78
79     fr_1 = flowrate_1*1000*pump_1_k #mL/min to microliter/min and syringe correction factor
80     #fr_2 = flowrate_2*1000*pump_2_k #mL/min to microliter/min and syringe correction factor
81     #fr_3 = flowrate_3*1000*pump_3_k #mL/min to microliter/min and syringe correction factor
82     #fr_4 = flowrate_4*1000*pump_4_k #mL/min to microliter/min and syringe correction factor
83     #fr_5 = flowrate_5*1000 #mL/min
84     #pump_5.write(('flow: '+ str(fr_5) + '\r').encode())
85
86     #time.sleep(0.1)
87     #fr_6=flowrate_6*1000 #converting mL/min to uL/min as pump takes this as input val
88     #pump_6.write(('flow: '+ str(fr_6) + '\r').encode())
89
90     #time.sleep(0.1)
91     #fr_7 = flowrate_7*1000 #mL/min
92     #pump_7.write(('flow: '+ str(fr_7) + '\r').encode())

```

```

93
94 #time.sleep(0.1)
95 #fr_8=flowrate_8*1000 #converting ml/min to ul/min as pump takes this as input val
96 #pump_8.write(('flow:'+ str(fr_8) + '\r').encode())
97
98 time.sleep(0.1)
99 fr_9=flowrate_9*1000 #converting ml/min to ul/min as pump takes this as input val
100 pump_9.write(('flow:'+ str(fr_9) + '\r').encode())
101 time.sleep(0.1)
102
103 #set the com port for potentiostat and set the voltage and current
104 vol = v
105 curr = i
106 port.write(('VOLT '+str(vol)+'\r\n').encode()) #to change the voltage we need to use "VOLT 1" command
107 port.write(('CURR '+str(curr)+'\r\n').encode()) #to change the current we need to use "CURR 1" command
108
109
110 #pumps run
111 #pump_5.write(b'on\r')
112 #time.sleep(0.1)
113 #pump_6.write(b'on\r')
114 #time.sleep(0.1)
115 #pump_7.write(b'on\r')
116 #time.sleep(0.1)
117 #pump_8.write(b'on\r')
118 #time.sleep(0.1)
119 pump_9.write(b'on\r')
120 time.sleep(0.1)
121
122
123 send_syr_command_1(f'fr_{fr_1}')
124 #send_syr_command_2(f'fr_{fr_2}')
125 #send_syr_command_3(f'fr_{fr_3}')
126 #send_syr_command_4(f'fr_{fr_4}')
127 time.sleep(6000)
128 def function2():
129     time.sleep(6000) # Last 3 minutes
130     files = [f for f in listdir(mypath) if isfile(join(mypath, f))]
131
132     val = 0
133
134     #change wavelengths as per product here.
135     f_row=17 #first row of range for wavelength as per IR CSV
136     l_row=24 #last row of range for wavelength as per IR CSV
137

```



```

138 val += ftir_extract(files[-1],f_row,l_row)
139 val += ftir_extract(files[-2],f_row,l_row)
140 val += ftir_extract(files[-3],f_row,l_row)
141
142 avg_val = val/3
143
144 return avg_val
145
146
147 #step 4:grab the line 93 and f9
148 from skopt.optimizer import Optimizer
149
150 def send_syr_command_1(command):
151     pump_1.write(command.encode() + b'\n') # Send the command to Arduino with a newline character
152     pump_1.flush() # Flush the serial buffer
153
154 #def send_syr_command_2(command):
155     #pump_2.write(command.encode() + b'\n') # Send the command to Arduino with a newline character
156     #pump_2.flush() # Flush the serial buffer
157
158 #def send_syr_command_3(command):
159     #pump_3.write(command.encode() + b'\n') # Send the command to Arduino with a newline character
160     #pump_3.flush() # Flush the serial buffer
161 #def send_syr_command_4(command):
162     #pump_4.write(command.encode() + b'\n') # Send the command to Arduino with a newline character
163     #pump_4.flush() # Flush the serial buffer
164 #step5:in line 96 we have to define the range that (flowrate,voltage,current) (from,to) and after (anytime) applying changes you need to grab the line 96 and f9
165 #flowrates are in ml/min, voltage in V, Current in Ampere
166 bounds = [(-0.050,-0.049),(0.1,0.2),(0.1,0.2),(0.1,0.2),(0.1,0.2),(0.1,0.2),(0.1,0.2),(0.1,0.2),(0.049,0.050),(14.49,14.50),(4.99,5.0)]
167 #step 6: grab the line 100 and f9
168 opter =Optimizer(bounds,base_estimator='gp',n_initial_points=3,acq_func="EI",random_state=np.random.randint(3326))
169
170
171 #step7: to selecte number of the cycles that you have to do the experiment and then grab the line 104 to 121 and f9: the closed loop experimentation is initiated
172 number_of_cycles =5
173 results = []
174 flowrates_1 = []
175 #flowrates_2 = []
176 #flowrates_3 = []
177 #flowrates_4 = []
178 #flowrates_5 = []
179 #flowrates_6 = []
180 #flowrates_7 = []
181 #flowrates_8 = []
182 flowrates_9 = []
183 vs = []

```

```

184 currents = []
185
186 product_wavelength=True #set to true if product wavelengths being monitored
187
188 if product_wavelength == True:
189     val=1
190 else:
191     val=-1
192
193 # Step 8: Test Tubes on Printer
194 USE_PRINTER = True
195 REST_HEIGHT = 200
196 X_HOME = -5
197 Y_HOME = 20
198 Z_HOME = 175
199 DEFAULT_PUMP_TIME="1"
200 # Distance between test tubes
201 X_SPACING=20
202 Y_SPACING=20
203 # Number of test tubes
204 X_ROWS = 11
205 Y_COLUMNS = 4
206
207 def send_cmd(cmd):
208     print(cmd)
209     printer.write(f"{cmd}\n".encode("ASCII"))
210
211 #-----code for changing column for run---
212 column_num=1 #column number of run to try from
213 Y_HOME=Y_HOME+(column_num-1)*Y_SPACING
214 #-----
215
216
217 def move(x=None, y=None, z=None):
218     s = "G0"
219     if x is not None:
220         s += f"X{x}"
221     if y is not None:
222         s += f"Y{y}"
223     if z is not None:
224         s += f"Z{z}"
225
226     s+= "F5000"
227     send_cmd(s)
228

```

```

229 def printer_positions():
230     for j in range(Y_COLUMNS):
231         for i in range(X_ROWS):
232             if j%2==1:
233                 yield (X_HOME + (X_ROWS - 1 - i) * X_SPACING, Y_HOME + j * Y_SPACING, Z_HOME)
234             else:
235                 yield (X_HOME + i * X_SPACING, Y_HOME + j * Y_SPACING, Z_HOME)
236
237 # Run this.
238 tube_location = list(printer_positions())
239
240 time.sleep(2)
241
242 try:
243     for i in range(number_of_cycles):
244         move(*tube_location[2*i])
245         asked = opter.ask()
246         function(asked[0],asked[1],asked[2],asked[3],asked[4],asked[5],asked[6],asked[7],asked[8],asked[9],asked[10])
247         move(*tube_location[2*i+1])
248         told= function2()
249
250         print(f"area under the curve in the round {i:.2f} = {told:.2f}")
251         opter.tell(asked,-told*val)
252         results.append(told)

```

```

253     flowrates_1.append(asked[0])
254     #flowrates_2.append(asked[1])
255     #flowrates_3.append(asked[2])
256     #flowrates_4.append(asked[3])
257     #flowrates_5.append(asked[4])
258     #flowrates_6.append(asked[5])
259     #flowrates_7.append(asked[6])
260     #flowrates_8.append(asked[7])
261     flowrates_9.append(asked[8])
262     vs.append(asked[9])
263     currents.append(asked[10])
264 #dict1 = {"flowrate_2":flowrates_2,"flowrate_3":flowrates_3,"flowrate_4":flowrates_4,"flowrate_5":flowrates_5,"flowrate_6":flowrates_6,"flowrate_7":flowrates_7,"flowrate_8":flowrates_8,"flowrate_9":flowrates_9,"voltage":vs,"currents":currents,"area-results":results}
265 dict1 = {"flowrate_1":flowrates_1,"flowrate_9":flowrates_9,"voltages":vs,"currents":currents,"area-results":results}
266 df2 = pd.DataFrame(dict1)
267 df2.to_csv("output round "+str(i)+".csv")
268 finally:
269     pump_1.write(b'stop\r')
270     #pump_2.write(b'stop\r')
271     #pump_3.write(b'stop\r')
272     #pump_4.write(b'stop\r')
273     #pump_5.write(b'off\r')
274     #pump_6.write(b'off\r')
275     #pump_7.write(b'off\r')
276     #pump_8.write(b'off\r')
277     pump_9.write(b'off\r')
278

```

AB2.py (for thiophenol)

```
AB2.py x
1
2 """
3 Created on Wed Aug 30 14:48:13 2023
4
5 @author: Admin
6 """
7
8
9 from os import listdir
10 from os.path import isfile, join
11 import serial
12
13 import numpy as np
14 import pandas as pd
15 import time
16 from scipy.integrate import trapz
17
18 #step1: be sure to the address of the files that the ftir data is exported is matching to line 11 (mypath)
19 mypath = "C:\\Users\\Admin\\Desktop\\ruchi\\Exp 2024-04-11 11-04"
20 onlyfiles = [f for f in listdir(mypath) if isfile(join(mypath, f))]
21
22
23 #step2: make sure that pump and the potentiostat is correctly addressed in the line 16 and 17
24 #pump_1 = serial.Serial("COM10",9600) #hwSyringe Pump
25 pump_2 = serial.Serial("COM16",9600) #Syringe Pump
26 #pump_3 = serial.Serial("COM19",9600) #Syringe Pump
27 #pump_4 = serial.Serial("COM15",9600) #Syringe Pump
28 #pump_5 = serial.serial_for_url("COM29",9600) #HPLC Pump LAN
29 #pump_6 = serial.serial_for_url("socket://169.254.225.18:10001",9600)
30 port = serial.Serial("COM1",115200)
31 #pump_7 = serial.Serial("COM28",9600) #HPLC Pump
32 #pump_8 = serial.Serial("COM30",9600) #HPLC Pump
33 pump_9 = serial.Serial("COM5",9600) #HPLC Pump
34 printer = serial.Serial("COM14", 115200, timeout=1)
35
36 r_1=14.25 #radius of syringe used (50ml,14.25|20ml,9.6|10ml,7.25|5ml,6.03|3ml,4/3|1ml,2.39|60ml,13.36)
37 r_2=14.25
38 r_3=14.25
39 r_4=14.25
40 #pump_1_k=(14.25/r_1)**2
41 pump_2_k=(14.25/r_2)**2
42 #pump_3_k=(14.25/r_3)**2
43 #pump_4_k=(14.25/r_4)**2
44
45 #step 3: grab the lines from 22 to 90 and press f9
46 def area_under(data,start,end):
```

```

47 x = np.flip(data.iloc[start:end,0].to_numpy())
48 y = np.flip(data.iloc[start:end,1].to_numpy())
49 area = trapz(y,x)
50 return np.abs(area)
51
52 def file_namer(num):
53     str1 = str(num)
54     length = int(len((str1)))
55     empt = ''
56     for i in range(5-length):
57         empt = empt+'0'
58
59     return empt+str1
60
61
62 def ftir_extract(filename,init,end):
63
64     filename = filename
65
66     temp_df = pd.read_csv(filename)
67     # nump_df = temp_df.to_numpy()
68     area = area_under(temp_df, init, end)
69     # max_peak = np.max(nump_df[90:120,1])
70     print(area)
71
72     return area
73
74
75 def function(flowrate_1,flowrate_2,flowrate_3,flowrate_4,flowrate_5,flowrate_6,flowrate_7,flowrate_8,flowrate_9,v,i):
76
77     #set the pumps with the flowrate as the desired flowrate for the function
78
79     #fr_1 = flowrate_1*1000*pump_1_k #mL/min to microliter/min and syringe correction factor
80     fr_2 = flowrate_2*1000*pump_2_k #mL/min to microliter/min and syringe correction factor
81     #fr_3 = flowrate_3*1000*pump_3_k #mL/min to microliter/min and syringe correction factor
82     #fr_4 = flowrate_4*1000*pump_4_k #mL/min to microliter/min and syringe correction factor
83     #fr_5 = flowrate_5*1000 #mL/min
84     #pump_5.write(('flow:'+ str(fr_5) + '\r').encode())
85
86     #time.sleep(0.1)
87     #fr_6=flowrate_6*1000 #converting mL/min to uL/min as pump takes this as input val
88     #pump_6.write(('flow:'+ str(fr_6) + '\r').encode())
89
90     #time.sleep(0.1)
91     #fr_7 = flowrate_7*1000 #mL/min
92     #pump_7.write(('flow:'+ str(fr_7) + '\r').encode())

```

```

93
94 #time.sleep(0.1)
95 #fr_8=flowrate_8*1000 #converting ml/min to ul/min as pump takes this as input val
96 #pump_8.write(('flow:'+ str(fr_8) + '\r').encode())
97
98 time.sleep(0.1)
99 fr_9=flowrate_9*1000 #converting ml/min to ul/min as pump takes this as input val
100 pump_9.write(('flow:'+ str(fr_9) + '\r').encode())
101 time.sleep(0.1)
102
103 #set the com port for potentiostat and set the voltage and current
104 vol = v
105 curr = i
106 port.write(('VOLT '+str(vol)+'\r\n').encode()) #to change the voltge we need to use "VOLT 1" command
107 port.write(('CURR '+str(curr)+'\r\n').encode()) #to change the current we need to use "CURR 1" command
108
109
110 #pumps run
111 #pump_5.write(b'on\r')
112 #time.sleep(0.1)
113 #pump_6.write(b'on\r')
114 #time.sleep(0.1)
115 #pump_7.write(b'on\r')
116 #time.sleep(0.1)
117 #pump_8.write(b'on\r')
118 #time.sleep(0.1)
119 pump_9.write(b'on\r')
120 time.sleep(0.1)
121
122
123 #send_syr_command_1(f'fr{fr_1}')
124 send_syr_command_2(f'fr{fr_2}')
125 #send_syr_command_3(f'fr{fr_3}')
126 #send_syr_command_4(f'fr{fr_4}')
127 time.sleep(3300)
128 def function2():
129 time.sleep(3300) # Last 3 minutes
130 files = [f for f in listdir(mypath) if isfile(join(mypath, f))]
131
132 val = 0
133
134 #change wavelengths as per product here.
135 f_row=17 #first row of range for wavelength as per IR CSV
136 l_row=24 #last row of range for wavelength as per IR CSV
137

```

```

138 val += ftir_extract(files[-1],f_row,l_row)
139 val += ftir_extract(files[-2],f_row,l_row)
140 val += ftir_extract(files[-3],f_row,l_row)
141
142 avg_val = val/3
143
144 return avg_val
145
146
147 #step 4:grab the line 93 and f9
148 from skopt.optimizer import Optimizer
149
150 #def send_syr_command_1(command):
151     #pump_1.write(command.encode() + b'\n') # Send the command to Arduino with a newline character
152     #pump_1.flush() # Flush the serial buffer
153
154 def send_syr_command_2(command):
155     pump_2.write(command.encode() + b'\n') # Send the command to Arduino with a newline character
156     pump_2.flush() # Flush the serial buffer
157
158 #def send_syr_command_3(command):
159     #pump_3.write(command.encode() + b'\n') # Send the command to Arduino with a newline character
160     #pump_3.flush() # Flush the serial buffer
161 #def send_syr_command_4(command):
162     #pump_4.write(command.encode() + b'\n') # Send the command to Arduino with a newline character
163     #pump_4.flush() # Flush the serial buffer
164 #step5:in line 96 we have to define the range that (flowrate,voltage,current) (from,to) and after (anytime) appling changes you need to grab the line 96 and f9
165 #flowrates are in ml/min, voltage in V, Current in Ampere
166 bounds = [(-10.0,-5.0),(0.086,0.087),(0.1,0.2),(0.1,0.2),(0.1,0.2),(0.1,0.2),(0.1,0.2),(0.1,0.2),(0.086,0.087),(14.49,14.50),(4.99,5.0)]
167 #step 6: grab the line 100 and f9
168 opter =Optimizer(bounds,base_estimator='gp',n_initial_points=3,acq_func="EI",random_state=np.random.randint(3326))
169
170
171 #step7: to selecte number of the cycles that you have to do the experiment and then grab the line 104 to 121 and f9: the closed loop experimentation is initiated
172 number_of_cycles =5
173 results = []
174 #flowrates_1 = []
175 flowrates_2 = []
176 #flowrates_3 = []
177 #flowrates_4 = []
178 #flowrates_5 = []
179 #flowrates_6 = []
180 #flowrates_7 = []
181 #flowrates_8 = []
182 flowrates_9 = []
183 vs = []

```

```

184 currents = []
185
186 product_wavelength=True #set to true if product wavelengths being monitored
187
188 if product_wavelength == True:
189     val=1
190 else:
191     val=-1
192
193 # Step 8: Test Tubes on Printer
194 USE_PRINTER = True
195 REST_HEIGHT = 200
196 X_HOME = -5
197 Y_HOME = 20
198 Z_HOME = 175
199 DEFAULT_PUMP_TIME="1"
200 # Distance between test tubes
201 X_SPACING=20
202 Y_SPACING=20
203 # Number of test tubes
204 X_ROWS = 11
205 Y_COLUMNS = 4
206
207 #-----code for changing column for run---
208 column_num=2 #column number of run to try from
209 Y_HOME=Y_HOME+(column_num-1)*Y_SPACING
210 #-----
211
212 def send_cmd(cmd):
213     print(cmd)
214     printer.write(f"{cmd}\n".encode("ASCII"))
215
216 def move(x=None, y=None, z=None):
217     s = "G0"
218     if x is not None:
219         s += f"X{x}"
220     if y is not None:
221         s += f"Y{y}"
222     if z is not None:
223         s += f"Z{z}"
224

```



```

225     s+= "F5000"
226     send_cmd(s)
227
228 def printer_positions():
229     for j in range(Y_COLUMNS):
230         for i in range(X_ROWS):
231             if j%2==1:
232                 yield (X_HOME + (X_ROWS - 1 - i) * X_SPACING, Y_HOME + j * Y_SPACING, Z_HOME)
233             else:
234                 yield (X_HOME + i * X_SPACING, Y_HOME + j * Y_SPACING, Z_HOME)
235
236 # Run this.
237 tube_location = list(printer_positions())
238
239 time.sleep(2)
240
241 try:
242     for i in range(number_of_cycles):
243         move(*tube_location[2*i])
244         asked = opter.ask()
245         function(asked[0],asked[1],asked[2],asked[3],asked[4],asked[5],asked[6],asked[7],asked[8],asked[9],asked[10])
246         move(*tube_location[2*i+1])
247         told= function2()
248
249         print(f"area under the curve in the round {i:.2f} = {told:.2f}")
250         opter.tell(asked,-told*val)
251         results.append(told)
252         #flowrates_1.append(asked[0])
253         flowrates_2.append(asked[1])
254         #flowrates_3.append(asked[2])
255         #flowrates_4.append(asked[3])
256         #flowrates_5.append(asked[4])
257         #flowrates_6.append(asked[5])
258         #flowrates_7.append(asked[6])
259         #flowrates_8.append(asked[7])
260         flowrates_9.append(asked[8])
261         vs.append(asked[9])
262         currents.append(asked[10])
263
264         #dict1 = {"flowrate_2":flowrates_2,"flowrate_3":flowrates_3,"flowrate_4":flowrates_4,"flowrate_5":flowrates_5,"flowrate_6":flowrates_6,"flowrate_7":flowrates_7,"flowrate_8":flowrates_8,"flowrate_9":flowrates_9
265         dict1 = {"flowrate_2":flowrates_2,"flowrate_9":flowrates_9,"voltages":vs,"currents":currents,"area-results":results}
266         df2 = pd.DataFrame(dict1)
267         df2.to_csv("output round "+str(i)+".csv")
268 finally:
269     #pump_1.write(b'stop\r')
270     pump_2.write(b'stop\r')
271     #pump_3.write(b'stop\r')
272     #pump_4.write(b'stop\r')
273     #pump_5.write(b'off\r')
274     #pump_6.write(b'off\r')
275     #pump_7.write(b'off\r')
276     #pump_8.write(b'off\r')
277     pump_9.write(b'off\r')
278

```

AB3.py (for aniline)

```
AB3.py* X
1
2 """
3 Created on Wed Aug 30 14:48:13 2023
4
5 @author: Admin
6 """
7
8
9 from os import listdir
10 from os.path import isfile, join
11 import serial
12
13 import numpy as np
14 import pandas as pd
15 import time
16 from scipy.integrate import trapz
17
18 #step1: be sure to the address of the files that the ftir data is exported is matching to line 11 (mypath)
19 mypath = "C:\\Users\\Admin\\Desktop\\ruchi\\Exp 2024-04-11 11-30"
20 onlyfiles = [f for f in listdir(mypath) if isfile(join(mypath, f))]
21
22
23 #step2: make sure that pump and the potentiostat is correctly addressed in the line 16 and 17
24 #pump_1 = serial.Serial("COM10",9600) #hwSyringe Pump
25 #pump_2 = serial.Serial("COM16",9600) #Syringe Pump
26 pump_3 = serial.Serial("COM19",9600) #Syringe Pump
27 #pump_4 = serial.Serial("COM15",9600) #Syringe Pump
28 #pump_5 = serial.serial_for_url("COM29",9600) #HPLC Pump LAN
29 #pump_6 = serial.serial_for_url("socket://169.254.225.18:10001",9600)
30 port = serial.Serial("COM1",115200)
31 #pump_7 = serial.Serial("COM28",9600) #HPLC Pump
32 #pump_8 = serial.Serial("COM30",9600) #HPLC Pump
33 pump_9 = serial.Serial("COM5",9600) #HPLC Pump
34 printer = serial.Serial("COM14", 115200, timeout=1)
35
36 r_1=14.25 #radius of syringe used (50ml,14.25|20ml,9.6|10ml,7.25|5ml,6.03|3ml,4/3|1ml,2.39|60ml,13.36)
37 r_2=14.25
38 r_3=14.25
39 r_4=14.25
40 #pump_1_k=(14.25/r_1)**2
41 #pump_2_k=(14.25/r_2)**2
42 pump_3_k=(14.25/r_3)**2
43 #pump_4_k=(14.25/r_4)**2
44
```

```

45 #step 3: grab the lines from 22 to 90 and presss f9
46 def area_under(data,start,end):
47     x = np.flip(data.iloc[start:end,0].to_numpy())
48     y = np.flip(data.iloc[start:end,1].to_numpy())
49     area = trapz(y,x)
50     return np.abs(area)
51
52 def file_namer(num):
53     str1 = str(num)
54     length = int(len((str1)))
55     empt = ''
56     for i in range(5-length):
57         empt = empt+'0'
58
59     return empt+str1
60
61
62 def ftir_extract(filename,init,end):
63
64     filename = filename
65
66     temp_df = pd.read_csv(filename)
67     # nump_df = temp_df.to_numpy()
68     area = area_under(temp_df, init, end)
69     # max_peak = np.max(nump_df[90:120,1])
70     print(area)
71
72     return area
73
74
75 def function(flowrate_1,flowrate_2,flowrate_3,flowrate_4,flowrate_5,flowrate_6,flowrate_7,flowrate_8,flowrate_9,v,i):
76
77     #set the pumps with the flowrate as the desired flowrate for the function
78
79     #fr_1 = flowrate_1*1000*pump_1_k #ml/min to microliter/min and syringe correction factor
80     #fr_2 = flowrate_2*1000*pump_2_k #ml/min to microliter/min and syringe correction factor
81     fr_3 = flowrate_3*1000*pump_3_k #ml/min to microliter/min and syringe correction factor
82     #fr_4 = flowrate_4*1000*pump_4_k #ml/min to microliter/min and syringe correction factor
83     #fr_5 = flowrate_5*1000 #ml/min
84     #pump_5.write(('flow:'+ str(fr_5) + '\r').encode())
85

```

```

86 #time.sleep(0.1)
87 #fr_6=flowrate_6*1000 #converting ml/min to ul/min as pump takes this as input val
88 #pump_6.write(('flow:'+ str(fr_6) + '\r').encode())
89
90 #time.sleep(0.1)
91 #fr_7 = flowrate_7*1000 #ml/min
92 #pump_7.write(('flow:'+ str(fr_7) + '\r').encode())
93
94 #time.sleep(0.1)
95 #fr_8=flowrate_8*1000 #converting ml/min to ul/min as pump takes this as input val
96 #pump_8.write(('flow:'+ str(fr_8) + '\r').encode())
97
98 time.sleep(0.1)
99 fr_9=flowrate_9*1000 #converting ml/min to ul/min as pump takes this as input val
100 pump_9.write(('flow:'+ str(fr_9) + '\r').encode())
101 time.sleep(0.1)
102
103 #set the com port for potentiostat and set the voltage and current
104 vol = v
105 curr = i
106 port.write(('VOLT '+str(vol)+'\r\n').encode()) #to change the voltge we need to use "VOLT 1" command
107 port.write(('CURR '+str(curr)+'\r\n').encode()) #to change the current we need to use "CURR 1" command
108
109
110 #pumps run
111 #pump_5.write(b'on\r')
112 #time.sleep(0.1)
113 #pump_6.write(b'on\r')
114 #time.sleep(0.1)
115 #pump_7.write(b'on\r')
116 #time.sleep(0.1)
117 #pump_8.write(b'on\r')
118 #time.sleep(0.1)
119 pump_9.write(b'on\r')
120 time.sleep(0.1)
121
122
123 #send_syr_command_1(f'fr_{fr_1}')
124 #send_syr_command_2(f'fr_{fr_2}')
125 send_syr_command_3(f'fr_{fr_3}')
126 #send_syr_command_4(f'fr_{fr_4}')
127 time.sleep(6000)
128 def function2():
129     time.sleep(6000) # Last 3 minutes
130     files = [f for f in listdir(mypath) if isfile(join(mypath, f))]

```

```

131
132     val = 0
133
134     #change wavelengths as per product here.
135     f_row=17 #first row of range for wavelength as per IR CSV
136     l_row=24 #Last row of range for wavelength as per IR CSV
137
138     val += ftir_extract(files[-1],f_row,l_row)
139     val += ftir_extract(files[-2],f_row,l_row)
140     val += ftir_extract(files[-3],f_row,l_row)
141
142     avg_val = val/3
143
144     return avg_val
145
146
147 #step 4:grab the line 93 and f9
148 from skopt.optimizer import Optimizer
149
150 #def send_syr_command_1(command):
151     #pump_1.write(command.encode() + b'\n') # Send the command to Arduino with a newline character
152     #pump_1.flush() # Flush the serial buffer
153
154 #def send_syr_command_2(command):
155     #pump_2.write(command.encode() + b'\n') # Send the command to Arduino with a newline character
156     #pump_2.flush() # Flush the serial buffer
157
158 def send_syr_command_3(command):
159     pump_3.write(command.encode() + b'\n') # Send the command to Arduino with a newline character
160     pump_3.flush() # Flush the serial buffer
161 #def send_syr_command_4(command):
162     #pump_4.write(command.encode() + b'\n') # Send the command to Arduino with a newline character
163     #pump_4.flush() # Flush the serial buffer
164 #step5:in line 96 we have to define the range that (flowrate,voltage,current) (from,to) and after (anytime) applying changes you need to grab the line 96 and f9
165 #flowrates are in ml/min, voltage in V, Current in Ampere
166 bounds = [(-10.0,-5.0),(0.1,0.2),(-0.05,-0.049),(0.1,0.2),(0.1,0.2),(0.1,0.2),(0.1,0.2),(0.1,0.2),(0.049,0.050),(14.49,14.5),(4.99,5.00)]
167 #step 6: grab the line 100 and f9
168 opter =Optimizer(bounds,base_estimator='gp',n_initial_points=3,acq_func="EI",random_state=np.random.randint(3326))
169
170

```

```

171 #step7: to selecte number of the cycles that you have to do the experiment and then grab the line 104 to 121 and f9: the closed loop
172 number_of_cycles =5
173 results = []
174 #flowrates_1 = []
175 #flowrates_2 = []
176 flowrates_3 = []
177 #flowrates_4 = []
178 #flowrates_5 = []
179 #flowrates_6 = []
180 #flowrates_7 = []
181 #flowrates_8 = []
182 flowrates_9 = []
183 vs = []
184 currents = []
185
186 product_wavelength=True #set to true if product wavelengths being monitored
187
188 if product_wavelength == True:
189     val=1
190 else:
191     val=-1
192
193 # Step 8: Test Tubes on Printer
194 USE_PRINTER = True
195 REST_HEIGHT = 200
196 X_HOME = -5
197 Y_HOME = 20
198 Z_HOME = 175
199 DEFAULT_PUMP_TIME="1"
200 # Distance between test tubes
201 X_SPACING=20
202 Y_SPACING=20
203 # Number of test tubes
204 X_ROWS = 11
205 Y_COLUMNS = 4
206
207 #-----code for changing column for run---
208 column_num=3 #column number of run to try from
209 Y_HOME=Y_HOME+(column_num-1)*Y_SPACING
210 #-----
211

```

```

212 def send_cmd(cmd):
213     print(cmd)
214     printer.write(f"{cmd}\n".encode("ASCII"))
215
216 def move(x=None, y=None, z=None):
217     s = "G0"
218     if x is not None:
219         s += f"X{x}"
220     if y is not None:
221         s += f"Y{y}"
222     if z is not None:
223         s += f"Z{z}"
224
225     s += "F5000"
226     send_cmd(s)
227
228 def printer_positions():
229     for j in range(Y_COLUMNS):
230         for i in range(X_ROWS):
231             if j%2==1:
232                 yield (X_HOME + (X_ROWS - 1 - i) * X_SPACING, Y_HOME + j * Y_SPACING, Z_HOME)
233             else:
234                 yield (X_HOME + i * X_SPACING, Y_HOME + j * Y_SPACING, Z_HOME)
235
236 # Run this.
237 tube_location = list(printer_positions())
238
239 time.sleep(2)
240
241 try:
242     for i in range(number_of_cycles):
243         move(*tube_location[2*i])
244         asked = opter.ask()
245         function(asked[0],asked[1],asked[2],asked[3],asked[4],asked[5],asked[6],asked[7],asked[8],asked[9],asked[10])
246         move(*tube_location[2*i+1])
247         told= function2()
248
249         print(f"area under the curve in the round {i:.2f} = {told:.2f}")
250         opter.tell(asked,-told*val)
251         results.append(told)
252         #flowrates_1.append(asked[0])
253         #flowrates_2.append(asked[1])
254         flowrates_3.append(asked[2])
255         #flowrates_4.append(asked[3])
256         #flowrates_5.append(asked[4])
257         #flowrates_6.append(asked[5])
258         #flowrates_7.append(asked[6])
259         #flowrates_8.append(asked[7])
260         flowrates_9.append(asked[8])
261         vs.append(asked[9])
262         currents.append(asked[10])
263
264         #dict1 = {"flowrate_2":flowrates_2,"flowrate_3":flowrates_3,"flowrate_4":flowrates_4,"flowrate_5":flowrates_5,"flowrate_6":flowrates_6,"flowrate_7":flowrates_7,"flowrate_8":flowrates_8,"flowrate_9":flowrates_
265         dict1 = {"flowrate_3":flowrates_3,"flowrate_9":flowrates_9,"voltages":vs,"currents":currents,"area-results":results}
266         df2 = pd.DataFrame(dict1)
267         df2.to_csv("output round "+str(i)+".csv")
268 finally:
269     #pump_1.write(b'stop\r')
270     #pump_2.write(b'stop\r')
271     pump_3.write(b'stop\r')
272     #pump_4.write(b'stop\r')
273     #pump_5.write(b'off\r')
274     #pump_6.write(b'off\r')
275     #pump_7.write(b'off\r')
276     #pump_8.write(b'off\r')
277     pump_9.write(b'off\r')
278

```

AB4.py (for styrene)

```
AB4.py x
1
2 """
3 Created on Wed Aug 30 14:48:13 2023
4
5 @author: Admin
6 """
7
8
9 from os import listdir
10 from os.path import isfile, join
11 import serial
12
13 import numpy as np
14 import pandas as pd
15 import time
16 from scipy.integrate import trapz
17
18 #step1: be sure to the address of the files that the ftir data is exported is matching to line 11 (mypath)
19 mypath = "C:\\Users\\Admin\\Desktop\\ruchi\\Exp 2024-04-11 12-01"
20 onlyfiles = [f for f in listdir(mypath) if isfile(join(mypath, f))]
21
22
23 #step2: make sure that pump and the potentiostat is correctly addressed in the line 16 and 17
24 #pump_1 = serial.Serial("COM10",9600) #hwSyringe Pump
25 #pump_2 = serial.Serial("COM16",9600) #Syringe Pump
26 #pump_3 = serial.Serial("COM19",9600) #Syringe Pump
27 #pump_4 = serial.Serial("COM15",9600) #Syringe Pump
28 #pump_5 = serial.Serial("COM29",9600) #HPLC Pump LAN
29 #pump_6 = serial.Serial("socket://169.254.225.18:10001",9600)
30 #port = serial.Serial("COM1",115200)
31 #pump_7 = serial.Serial("COM28",9600) #HPLC Pump
32 #pump_8 = serial.Serial("COM30",9600) #HPLC Pump
33 #pump_9 = serial.Serial("COM5",9600) #HPLC Pump
34 #printer = serial.Serial("COM14", 115200, timeout=1)
35
36 r_1=14.25 #radius of syringe used (50mL,14.25|20mL,9.6|10mL,7.25|5mL,6.03|3mL,4/3|1mL,2.39|60mL,13.36)
37 r_2=14.25
38 r_3=14.25
39 r_4=14.25
40 #pump_1_k=(14.25/r_1)**2
41 #pump_2_k=(14.25/r_2)**2
42 #pump_3_k=(14.25/r_3)**2
43 #pump_4_k=(14.25/r_4)**2
44
```



```

45 #step 3: grab the lines from 22 to 90 and press f9
46 def area_under(data,start,end):
47     x = np.flip(data.iloc[start:end,0].to_numpy())
48     y = np.flip(data.iloc[start:end,1].to_numpy())
49     area = trapezoid(y,x)
50     return np.abs(area)
51
52 def file_namer(num):
53     str1 = str(num)
54     length = int(len(str1))
55     empt = ''
56     for i in range(5-length):
57         empt = empt+'0'
58
59     return empt+str1
60
61
62 def ftir_extract(filename,init,end):
63
64     filename = filename
65
66     temp_df = pd.read_csv(filename)
67     # nump_df = temp_df.to_numpy()
68     area = area_under(temp_df, init, end)
69     # max_peak = np.max(nump_df[90:120,1])
70     print(area)
71
72     return area
73
74
75 def function(flowrate_1,flowrate_2,flowrate_3,flowrate_4,flowrate_5,flowrate_6,flowrate_7,flowrate_8,flowrate_9,v,i):
76
77     #set the pumps with the flowrate as the desired flowrate for the function
78
79     #fr_1 = flowrate_1*1000*pump_1_k #mL/min to microliter/min and syringe correction factor
80     #fr_2 = flowrate_2*1000*pump_2_k #mL/min to microliter/min and syringe correction factor
81     #fr_3 = flowrate_3*1000*pump_3_k #mL/min to microliter/min and syringe correction factor
82     fr_4 = flowrate_4*1000*pump_4_k #mL/min to microliter/min and syringe correction factor
83     #time.sleep(0.1)
84     #fr_5 = flowrate_5*1000 #mL/min
85     #pump_5.write(('flow:'+ str(fr_5) + '\r').encode())
86

```

```

87 #time.sleep(0.1)
88 #fr_6=flowrate_6*1000 #converting ml/min to ul/min as pump takes this as input val
89 #pump_6.write(('flow:'+ str(fr_6) + '\r').encode())
90
91 #time.sleep(0.1)
92 #fr_7 = flowrate_7*1000 #mL/min
93 #pump_7.write(('flow:'+ str(fr_7) + '\r').encode())
94
95 time.sleep(0.1)
96 fr_8=flowrate_8*1000 #converting ml/min to ul/min as pump takes this as input val
97 pump_8.write(('flow:'+ str(fr_8) + '\r').encode())
98
99 #time.sleep(0.1)
100 #fr_9=flowrate_9*1000 #converting ml/min to ul/min as pump takes this as input val
101 #pump_9.write(('flow:'+ str(fr_9) + '\r').encode())
102 #time.sleep(0.1)
103
104 #set the com port for potentiostat and set the voltage and current
105 vol = v
106 curr = i
107 port.write(('VOLT '+str(vol)+'\r\n').encode()) #to change the voltge we need to use "VOLT 1" command
108 port.write(('CURR '+str(curr)+'\r\n').encode()) #to change the current we need to use "CURR 1" command
109
110
111 #pumps run
112 #pump_5.write(b'on\r')
113 #time.sleep(0.1)
114 #pump_6.write(b'on\r')
115 #time.sleep(0.1)
116 #pump_7.write(b'on\r')
117 #time.sleep(0.1)
118 pump_8.write(b'on\r')
119 time.sleep(0.1)
120 #pump_9.write(b'on\r')
121 #time.sleep(0.1)
122
123

```

```

124 #send_syr_command_1(f'fr{fr_1}')
125 #send_syr_command_2(f'fr{fr_2}')
126 #send_syr_command_3(f'fr{fr_3}')
127 send_syr_command_4(f'fr{fr_4}')
128 time.sleep(15000)
129 def function2():
130     time.sleep(15000) # Last 3 minutes
131     files = [f for f in listdir(mypath) if.isfile(join(mypath, f))]
132
133     val = 0
134
135     #change wavelengths as per product here.
136     f_row=17 #first row of range for wavelength as per IR CSV
137     l_row=24 #Last row of range for wavelength as per IR CSV
138
139     val += ftir_extract(files[-1],f_row,l_row)
140     val += ftir_extract(files[-2],f_row,l_row)
141     val += ftir_extract(files[-3],f_row,l_row)
142
143     avg_val = val/3
144
145     return avg_val
146
147
148 #step 4:grab the line 93 and f9
149 from skopt.optimizer import Optimizer
150
151 #def send_syr_command_1(command):
152     #pump_1.write(command.encode() + b'\n') # Send the command to Arduino with a newline character
153     #pump_1.flush() # Flush the serial buffer
154
155 #def send_syr_command_2(command):
156     #pump_2.write(command.encode() + b'\n') # Send the command to Arduino with a newline character
157     #pump_2.flush() # Flush the serial buffer
158
159 #def send_syr_command_3(command):
160     #pump_3.write(command.encode() + b'\n') # Send the command to Arduino with a newline character
161     #pump_3.flush() # Flush the serial buffer

```

```

162 def send_syr_command_4(command):
163     pump_4.write(command.encode() + b'\n') # Send the command to Arduino with a newline character
164     pump_4.flush() # Flush the serial buffer
165 #step5:in line 96 we have to define the range that (flowrate,voltage,current) (from,to) and after (anytime) applying changes you need to grab the line 96 and f9
166 #flowrates are in ml/min, voltage in V, Current in Amperes
167 bounds = [(-10.0,-5.0),(0.1,0.2),(0.1,0.2),(0.014,0.015),(0.1,0.2),(0.1,0.2),(0.1,0.2),(0.014,0.015),(0.1,0.2),(14.49,14.5),(4.99,5.0)]
168 #step 6: grab the line 100 and f9
169 opter =Optimizer(bounds,base_estimator='gp',n_initial_points=3,acq_func="EI",random_state=np.random.randint(3326))
170
171
172 #step7: to selecte number of the cycles that you have to do the experiment and then grab the line 104 to 121 and f9: the closed loop experimentation is initiated
173 number_of_cycles =5
174 results = []
175 #flowrates_1 = []
176 #flowrates_2 = []
177 #flowrates_3 = []
178 flowrates_4 = []
179 #flowrates_5 = []
180 #flowrates_6 = []
181 #flowrates_7 = []
182 flowrates_8 = []
183 #flowrates_9 = []
184 vs = []
185 currents = []
186
187 product_wavelength=True #set to true if product wavelengths being monitored
188
189 if product_wavelength == True:
190     val=1
191 else:
192     val=-1
193
194 # Step 8: Test Tubes on Printer
195 USE_PRINTER = True
196 REST_HEIGHT = 200
197 X_HOME = -5
198 Y_HOME = 20
199 Z_HOME = 175
200 DEFAULT_PUMP_TIME="1"
201 # Distance between test tubes
202 X_SPACING=20
203 Y_SPACING=20
204 # Number of test tubes

```

```

205 X_ROWS = 11
206 Y_COLUMNS = 4
207
208 #-----code for changing column for run---
209 column_num=4 #column number of run to try from
210 Y_HOME=Y_HOME+(column_num-1)*Y_SPACING
211 #-----
212
213 def send_cmd(cmd):
214     print(cmd)
215     printer.write(f"{cmd}\n".encode("ASCII"))
216
217 def move(x=None, y=None, z=None):
218     s = "G0"
219     if x is not None:
220         s += f"X{x}"
221     if y is not None:
222         s += f"Y{y}"
223     if z is not None:
224         s += f"Z{z}"
225
226     s+= "F5000"
227     send_cmd(s)
228
229 def printer_positions():
230     for j in range(Y_COLUMNS):
231         for i in range(X_ROWS):
232             if j%2==1:
233                 yield (X_HOME + (X_ROWS - 1 - i) * X_SPACING, Y_HOME + j * Y_SPACING, Z_HOME)
234             else:
235                 yield (X_HOME + i * X_SPACING, Y_HOME + j * Y_SPACING, Z_HOME)
236
237 # Run this.
238 tube_location = list(printer_positions())
239

```

```

240 time.sleep(2)
241
242 try:
243     for i in range(number_of_cycles):
244         move(*tube_location[2*i])
245         asked = opter.ask()
246         function(asked[0],asked[1],asked[2],asked[3],asked[4],asked[5],asked[6],asked[7],asked[8],asked[9],asked[10])
247         move(*tube_location[2*i+1])
248         told= function2()
249
250         print(f"area under the curve in the round {i:.2f} = {told:.2f}")
251         opter.tell(asked,-told*val)
252         results.append(told)
253         #flowrates_1.append(asked[0])
254         #flowrates_2.append(asked[1])
255         #flowrates_3.append(asked[2])
256         flowrates_4.append(asked[3])
257         #flowrates_5.append(asked[4])
258         #flowrates_6.append(asked[5])
259         #flowrates_7.append(asked[6])
260         flowrates_8.append(asked[7])
261         #flowrates_9.append(asked[8])
262         vs.append(asked[9])
263         currents.append(asked[10])
264
265         #dict1 = {"flowrate_2":flowrates_2,"flowrate_3":flowrates_3,"flowrate_4":flowrates_4,"flowrate_5":flowrates_5,"flowrate_6":flowrates_6,"flowrate_7":flowrates_7,"flowrate_8":flowrates_8,"flowrate_9":flowrates_5
266         dict1 = {"flowrate_4":flowrates_4,"flowrate_8":flowrates_8,"voltages":vs,"currents":currents,"area-results":results}
267         df2 = pd.DataFrame(dict1)
268         df2.to_csv("output round "+str(i)+".csv")
269 finally:
270     #pump_1.write(b'stop\r')
271     #pump_2.write(b'stop\r')
272     #pump_3.write(b'stop\r')
273     pump_4.write(b'stop\r')
274     #pump_5.write(b'off\r')
275     #pump_6.write(b'off\r')
276     #pump_7.write(b'off\r')
277     pump_8.write(b'off\r')
278     #pump_9.write(b'off\r')
279

```

AB5.py (for phenyl acetylene)

```
AB5.py x
1
2 """
3 Created on Wed Aug 30 14:48:13 2023
4
5 @author: Admin
6 """
7
8
9 from os import listdir
10 from os.path import isfile, join
11 import serial
12
13 import numpy as np
14 import pandas as pd
15 import time
16 from scipy.integrate import trapz
17
18 #step1: be sure to the address of the files that the ftir data is exported is matching to line 11 (mypath)
19 mypath = "C:\\Users\\Admin\\Desktop\\ruchi\\Exp 2024-04-11 13-33"
20 onlyfiles = [f for f in listdir(mypath) if isfile(join(mypath, f))]
21
22
23 #step2: make sure that pump and the potentiostat is correctly addressed in the Line 16 and 17
24 #pump_1 = serial.Serial("COM10",9600) #hwSyringe Pump
25 #pump_2 = serial.Serial("COM16",9600) #Syringe Pump
26 #pump_3 = serial.Serial("COM19",9600) #Syringe Pump
27 #pump_4 = serial.Serial("COM15",9600) #Syringe Pump
28 #pump_5 = serial.Serial("COM29",9600) #HPLC Pump
29 #pump_6 = serial.Serial("COM28",9600) #HPLC Pump
30 #pump_7 = serial.Serial("COM28",9600) #HPLC Pump
31 #pump_8 = serial.Serial("COM30",9600) #HPLC Pump
32 #pump_9 = serial.Serial("COM5",9600) #HPLC Pump
33 #printer = serial.Serial("COM14", 115200, timeout=1)
34
35
36 r_1=14.25 #radius of syringe used (50ml,14.25|20ml,9.6|10ml,7.25|5ml,6.03|3ml,4/3|1ml,2.39|60ml,13.36)
37 r_2=14.25
38 r_3=14.25
39 r_4=14.25
40 #pump_1_k=(14.25/r_1)**2
41 #pump_2_k=(14.25/r_2)**2
42 #pump_3_k=(14.25/r_3)**2
43 #pump_4_k=(14.25/r_4)**2
44
```

```

45 #step 3: grab the lines from 22 to 90 and press f9
46 def area_under(data,start,end):
47     x = np.flip(data.iloc[start:end,0].to_numpy())
48     y = np.flip(data.iloc[start:end,1].to_numpy())
49     area = trapz(y,x)
50     return np.abs(area)
51
52 def file_namer(num):
53     str1 = str(num)
54     length = int(len((str1)))
55     empt = ''
56     for i in range(5-length):
57         empt = empt+'0'
58
59     return empt+str1
60
61
62 def ftir_extract(filename,init,end):
63
64     filename = filename
65
66     temp_df = pd.read_csv(filename)
67     # nump_df = temp_df.to_numpy()
68     area = area_under(temp_df, init, end)
69     # max_peak = np.max(nump_df[90:120,1])
70     print(area)
71
72     return area
73
74
75 def function(flowrate_1,flowrate_2,flowrate_3,flowrate_4,flowrate_5,flowrate_6,flowrate_7,flowrate_8,flowrate_9,v,i):
76
77     #set the pumps with the flowrate as the desired flowrate for the function
78
79     #fr_1 = flowrate_1*1000*pump_1_k #mL/min to microliter/min and syringe correction factor
80     #fr_2 = flowrate_2*1000*pump_2_k #mL/min to microliter/min and syringe correction factor
81     #fr_3 = flowrate_3*1000*pump_3_k #mL/min to microliter/min and syringe correction factor
82     #fr_4 = flowrate_4*1000*pump_4_k #mL/min to microliter/min and syringe correction factor
83     time.sleep(0.1)
84     fr_5 = flowrate_5*1000 #mL/min
85     pump_5.write(('flow: '+ str(fr_5) + '\r').encode())
86

```



```

87  #time.sleep(0.1)
88  #fr_6=flowrate_6*1000 #converting ml/min to ul/min as pump takes this as input val
89  #pump_6.write(('flow:'+ str(fr_6) + '\r').encode())
90
91  #time.sleep(0.1)
92  #fr_7 = flowrate_7*1000  #ml/min
93  #pump_7.write(('flow:'+ str(fr_7) + '\r').encode())
94
95  time.sleep(0.1)
96  fr_8=flowrate_8*1000 #converting ml/min to ul/min as pump takes this as input val
97  pump_8.write(('flow:'+ str(fr_8) + '\r').encode())
98
99  #time.sleep(0.1)
100 #fr_9=flowrate_9*1000 #converting ml/min to ul/min as pump takes this as input val
101 #pump_9.write(('flow:'+ str(fr_9) + '\r').encode())
102 #time.sleep(0.1)
103
104 #set the com port for potentiostat and set the voltage and current
105 vol = v
106 curr = i
107 port.write(('VOLT '+str(vol)+'\r\n').encode()) #to change the voltge we need to use "VOLT 1" command
108 port.write(('CURR '+str(curr)+'\r\n').encode()) #to change the current we need to use "CURR 1" command
109
110
111 #pumps run
112 pump_5.write(b'on\r')
113 time.sleep(0.1)
114 #pump_6.write(b'on\r')
115 #time.sleep(0.1)
116 #pump_7.write(b'on\r')
117 #time.sleep(0.1)
118 pump_8.write(b'on\r')
119 time.sleep(0.1)
120 #pump_9.write(b'on\r')
121 #time.sleep(0.1)
122
123

```

```

125 #send_syr_command_2(f'fr{fr_2}')
126 #send_syr_command_3(f'fr{fr_3}')
127 #send_syr_command_4(f'fr{fr_4}')
128 time.sleep(15000)
129 def function2():
130     time.sleep(15000) # Last 3 minutes
131     files = [f for f in listdir(mypath) if isfile(join(mypath, f))]
132
133     val = 0
134
135     #change wavelengths as per product here.
136     f_row=17 #first row of range for wavelength as per IR CSV
137     l_row=24 #Last row of range for wavelength as per IR CSV
138
139     val += ftir_extract(files[-1],f_row,l_row)
140     val += ftir_extract(files[-2],f_row,l_row)
141     val += ftir_extract(files[-3],f_row,l_row)
142
143     avg_val = val/3
144
145     return avg_val
146
147
148 #step 4:grab the line 93 and f9
149 from skopt.optimizer import Optimizer
150
151 #def send_syr_command_1(command):
152     #pump_1.write(command.encode() + b'\n') # Send the command to Arduino with a newline character
153     #pump_1.flush() # Flush the serial buffer
154
155 #def send_syr_command_2(command):
156     #pump_2.write(command.encode() + b'\n') # Send the command to Arduino with a newline character
157     #pump_2.flush() # Flush the serial buffer
158
159 #def send_syr_command_3(command):
160     #pump_3.write(command.encode() + b'\n') # Send the command to Arduino with a newline character
161     #pump_3.flush() # Flush the serial buffer
162 #def send_syr_command_4(command):
163     #pump_4.write(command.encode() + b'\n') # Send the command to Arduino with a newline character
164     #pump_4.flush() # Flush the serial buffer
165 #step5:in line 96 we have to define the range that (fLowrate,voltage,current) (from,to) and after (anytime) applying changes you need to grab the line 96 and f9
166 #flowrates are in ml/min, voltage in V, Current in Amperes
167 bounds = [(-10.0,-5.0),(0.1,0.2),(0.1,0.2),(0.1,0.2),(0.012,0.013),(0.1,0.2),(0.1,0.2),(0.012,0.013),(0.1,0.2),(14.49,14.50),(4.99,5.00)]
168 #step 6: grab the line 100 and f9
169 opter =Optimizer(bounds,base_estimator='gp',n_initial_points=3,acq_func="EI",random_state=np.random.randint(3326))

```

```

170
171
172 #step7: to selecte number of the cycles that you have to do the experiment and then grab the line 104 to 121 and f9: the closed loop experimentation is initiated
173 number_of_cycles =5
174 results = []
175 #flowrates_1 = []
176 #flowrates_2 = []
177 #flowrates_3 = []
178 #flowrates_4 = []
179 flowrates_5 = []
180 #flowrates_6 = []
181 #flowrates_7 = []
182 flowrates_8 = []
183 #flowrates_9 = []
184 vs = []
185 currents = []
186
187 product_wavelength=True #set to true if product wavelengths being monitored
188
189 if product_wavelength == True:
190     val=1
191 else:
192     val=-1
193
194 # Step 8: Test Tubes on Printer
195 USE_PRINTER = True
196 REST_HEIGHT = 200
197 X_HOME = -5
198 Y_HOME = 20
199 Z_HOME = 175
200 DEFAULT_PUMP_TIME="1"
201 # Distance between test tubes
202 X_SPACING=20
203 Y_SPACING=20
204 # Number of test tubes
205 X_ROWS = 11
206 Y_COLUMNS = 4
207
208 #-----code for changing column for run---

```

```

209 column_num=5 #column number of run to try from
210 Y_HOME=Y_HOME+(column_num-1)*Y_SPACING
211 #-----
212
213 def send_cmd(cmd):
214     print(cmd)
215     printer.write(f"{cmd}\n".encode("ASCII"))
216
217 def move(x=None, y=None, z=None):
218     s = "G0"
219     if x is not None:
220         s += f"X{x}"
221     if y is not None:
222         s += f"Y{y}"
223     if z is not None:
224         s += f"Z{z}"
225
226     s+= "F5000"
227     send_cmd(s)
228
229 def printer_positions():
230     for j in range(Y_COLUMNS):
231         for i in range(X_ROWS):
232             if j%2==1:
233                 yield (X_HOME + (X_ROWS - 1 - i) * X_SPACING, Y_HOME + j * Y_SPACING, Z_HOME)
234             else:
235                 yield (X_HOME + i * X_SPACING, Y_HOME + j * Y_SPACING, Z_HOME)
236

```

```

237 # Run this.
238 tube_location = list(printer_positions())
239
240 try:
241     for i in range(number_of_cycles):
242         move(*tube_location[2*i])
243         asked = opter.ask()
244         function(asked[0],asked[1],asked[2],asked[3],asked[4],asked[5],asked[6],asked[7],asked[8],asked[9],asked[10])
245         move(*tube_location[2*i+1])
246         told= function2()
247
248         print(f"area under the curve in the round {i:.2f} = {told:.2f}")
249         opter.tell(asked,-told*val)
250         results.append(told)
251         #flowrates_1.append(asked[0])
252         #flowrates_2.append(asked[1])
253         #flowrates_3.append(asked[2])
254         #flowrates_4.append(asked[3])
255         flowrates_5.append(asked[4])
256         #flowrates_6.append(asked[5])
257         #flowrates_7.append(asked[6])
258         flowrates_8.append(asked[7])
259         #flowrates_9.append(asked[8])
260         vs.append(asked[9])
261         currents.append(asked[10])
262
263         #dict1 = {"flowrate_2":flowrates_2,"flowrate_3":flowrates_3,"flowrate_4":flowrates_4,"flowrate_5":flowrates_5,"flowrate_6":flowrates_6,"flowrate_7":flowrates_7,"flowrate_8":flowrates_8,"flowrate_9":flowrates_9}
264         dict1 = {"flowrate_5":flowrates_5,"flowrate_8":flowrates_8,"voltages":vs,"currents":currents,"area-results":results}
265         df2 = pd.DataFrame(dict1)
266         df2.to_csv("output round "+str(i)+".csv")
267 finally:
268     #pump_1.write(b'stop\r')
269     #pump_2.write(b'stop\r')
270     #pump_3.write(b'stop\r')
271     #pump_4.write(b'stop\r')
272     pump_5.write(b'off\r')
273     #pump_6.write(b'off\r')
274     #pump_7.write(b'off\r')
275     pump_8.write(b'off\r')
276     #pump_9.write(b'off\r')
277

```

AB6.py (for acetonitrile)

```
AB6.py x
1
2 """
3 Created on Wed Aug 30 14:48:13 2023
4
5 @author: Admin
6 """
7
8
9 from os import listdir
10 from os.path import isfile, join
11 import serial
12
13 import numpy as np
14 import pandas as pd
15 import time
16 from scipy.integrate import trapz
17
18 #step1: be sure to the address of the files that the ftir data is exported is matching to line 11 (mypath)
19 mypath = "C:\\Users\\Admin\\Desktop\\ruchi\\Exp 2024-04-11 13-46"
20 onlyfiles = [f for f in listdir(mypath) if isfile(join(mypath, f))]
21
22
23 #step2: make sure that pump and the potentiostat is correctly addressed in the line 16 and 17
24 #pump_1 = serial.Serial("COM10",9600) #hwSyringe Pump
25 #pump_2 = serial.Serial("COM16",9600) #Syringe Pump
26 #pump_3 = serial.Serial("COM19",9600) #Syringe Pump
27 #pump_4 = serial.Serial("COM15",9600) #Syringe Pump
28 #pump_5 = serial.serial_for_url("COM29",9600) #HPLC Pump LAN
29 pump_6 = serial.serial_for_url("socket://169.254.189.57:10001",9600)
30 port = serial.Serial("COM1",115200)
31 #pump_7 = serial.Serial("COM28",9600) #HPLC Pump
32 pump_8 = serial.Serial("COM30",9600) #HPLC Pump
33 #pump_9 = serial.Serial("COM5",9600) #HPLC Pump
34 printer = serial.Serial("COM14", 115200, timeout=1)
35
36 r_1=14.25 #radius of syringe used (50ml,14.25|20ml,9.6|10ml,7.25|5ml,6.03|3ml,4/3|1ml,2.39|60ml,13.36)
37 r_2=14.25
38 r_3=14.25
39 r_4=14.25
40 #pump_1_k=(14.25/r_1)**2
41 #pump_2_k=(14.25/r_2)**2
42 #pump_3_k=(14.25/r_3)**2
43 #pump_4_k=(14.25/r_4)**2
44
```

```

45 #step 3: grab the lines from 22 to 90 and press f9
46 def area_under(data,start,end):
47     x = np.flip(data.iloc[start:end,0].to_numpy())
48     y = np.flip(data.iloc[start:end,1].to_numpy())
49     area = trapz(y,x)
50     return np.abs(area)
51
52 def file_namer(num):
53     str1 = str(num)
54     length = int(len((str1)))
55     empt = ''
56     for i in range(5-length):
57         empt = empt+'0'
58
59     return empt+str1
60
61
62 def ftir_extract(filename,init,end):
63
64     filename = filename
65
66     temp_df = pd.read_csv(filename)
67     # nump_df = temp_df.to_numpy()
68     area = area_under(temp_df, init, end)
69     # max_peak = np.max(nump_df[90:120,1])
70     print(area)
71
72     return area
73
74
75 def function(flowrate_1,flowrate_2,flowrate_3,flowrate_4,flowrate_5,flowrate_6,flowrate_7,flowrate_8,flowrate_9,v,i):
76
77     #set the pumps with the flowrate as the desired flowrate for the function
78
79     #fr_1 = flowrate_1*1000*pump_1_k #mL/min to microliter/min and syringe correction factor
80     #fr_2 = flowrate_2*1000*pump_2_k #mL/min to microliter/min and syringe correction factor
81     #fr_3 = flowrate_3*1000*pump_3_k #mL/min to microliter/min and syringe correction factor
82     #fr_4 = flowrate_4*1000*pump_4_k #mL/min to microliter/min and syringe correction factor
83     #time.sleep(0.1)
84     #fr_5 = flowrate_5*1000 #mL/min
85     #pump_5.write(('flow:'+ str(fr_5) + '\r').encode())
86

```

```

87  time.sleep(0.1)
88  fr_6=flowrate_6*1000 #converting ml/min to ul/min as pump takes this as input val
89  pump_6.write(('flow:'+ str(fr_6) + '\r').encode())
90
91  #time.sleep(0.1)
92  #fr_7 = flowrate_7*1000  #ml/min
93  #pump_7.write(('flow:'+ str(fr_7) + '\r').encode())
94
95  time.sleep(0.1)
96  fr_8=flowrate_8*1000 #converting ml/min to ul/min as pump takes this as input val
97  pump_8.write(('flow:'+ str(fr_8) + '\r').encode())
98
99  #time.sleep(0.1)
100 #fr_9=flowrate_9*1000 #converting ml/min to ul/min as pump takes this as input val
101 #pump_9.write(('flow:'+ str(fr_9) + '\r').encode())
102 #time.sleep(0.1)
103
104 #set the com port for potentiostat and set the voltage and current
105 vol = v
106 curr = i
107 port.write(('VOLT '+str(vol)+'\r\n').encode()) #to change the vltge we need to use "VOLT 1" command
108 port.write(('CURR '+str(curr)+'\r\n').encode()) #to change the current we need to use "CURR 1" command
109
110
111 #pumps run
112 #pump_5.write(b'on\r')
113 #time.sleep(0.1)
114 pump_6.write(b'on\r')
115 time.sleep(0.1)
116 #pump_7.write(b'on\r')
117 #time.sleep(0.1)
118 pump_8.write(b'on\r')
119 time.sleep(0.1)
120 #pump_9.write(b'on\r')
121 #time.sleep(0.1)
122
123

```



```

124 #send_syr_command_1(f'fr{fr_1}')
125 #send_syr_command_2(f'fr{fr_2}')
126 #send_syr_command_3(f'fr{fr_3}')
127 #send_syr_command_4(f'fr{fr_4}')
128 time.sleep(8400)
129 def function2():
130     time.sleep(8400) # Last 3 minutes
131     files = [f for f in listdir(mypath) if isfile(join(mypath, f))]
132
133     val = 0
134
135     #change wavelengths as per product here.
136     f_row=17 #first row of range for wavelength as per IR CSV
137     l_row=24 #Last row of range for wavelength as per IR CSV
138
139     val += ftir_extract(files[-1],f_row,l_row)
140     val += ftir_extract(files[-2],f_row,l_row)
141     val += ftir_extract(files[-3],f_row,l_row)
142
143     avg_val = val/3
144
145     return avg_val
146
147
148 #step 4:grab the line 93 and f9
149 from skopt.optimizer import Optimizer
150
151 #def send_syr_command_1(command):
152     #pump_1.write(command.encode() + b'\n') # Send the command to Arduino with a newline character
153     #pump_1.flush() # Flush the serial buffer
154
155 #def send_syr_command_2(command):
156     #pump_2.write(command.encode() + b'\n') # Send the command to Arduino with a newline character
157     #pump_2.flush() # Flush the serial buffer
158

```

```

159 #def send_syr_command_3(command):
160     #pump_3.write(command.encode() + b'\n') # Send the command to Arduino with a newline character
161     #pump_3.flush() # Flush the serial buffer
162 #def send_syr_command_4(command):
163     #pump_4.write(command.encode() + b'\n') # Send the command to Arduino with a newline character
164     #pump_4.flush() # Flush the serial buffer
165 #step5: in line 96 we have to define the range that (flowrate,voltage,current) (from,to) and after (anytime) applying changes you need to grab the line 96 and f9
166 #flowrates are in ml/min, voltage in V, Current in Ampere
167 bounds = [(-10.0,-5.0),(0.1,0.2),(0.1,0.2),(0.1,0.2),(0.1,0.2),(0.029,0.030),(0.1,0.2),(0.029,0.030),(0.1,0.2),(14.49,14.50),(4.99,5.00)]
168 #step 6: grab the line 100 and f9
169 opter =Optimizer(bounds,base_estimator='gp',n_initial_points=3,acq_func="EI",random_state=np.random.randint(3326))
170
171
172 #step7: to selecte number of the cycles that you have to do the experiment and then grab the line 104 to 121 and f9: the closed loop experimentation is initiated
173 number_of_cycles =5
174 results = []
175 #flowrates_1 = []
176 #flowrates_2 = []
177 #flowrates_3 = []
178 #flowrates_4 = []
179 #flowrates_5 = []
180 flowrates_6 = []
181 #flowrates_7 = []
182 flowrates_8 = []
183 #flowrates_9 = []
184 vs = []
185 currents = []
186
187 product_wavelength=True #set to true if product wavelengths being monitored
188
189 if product_wavelength == True:
190     val=1
191 else:
192     val=-1
193

```

```

194 # Step 8: Test Tubes on Printer
195 USE_PRINTER = True
196 REST_HEIGHT = 200
197 X_HOME = -5
198 Y_HOME = 20
199 Z_HOME = 175
200 DEFAULT_PUMP_TIME="1"
201 # Distance between test tubes
202 X_SPACING=20
203 Y_SPACING=20
204 # Number of test tubes
205 X_ROWS = 11
206 Y_COLUMNS = 4
207
208 #-----code for changing column for run---
209 column_num=6 #column number of run to try from
210 Y_HOME=Y_HOME+(column_num-1)*Y_SPACING
211 #-----
212
213 def send_cmd(cmd):
214     print(cmd)
215     printer.write(f"{cmd}\n".encode("ASCII"))
216
217 def move(x=None, y=None, z=None):
218     s = "G0"
219     if x is not None:
220         s += f"X{x}"
221     if y is not None:
222         s += f"Y{y}"
223     if z is not None:
224         s += f"Z{z}"
225
226     s+= "F5000"
227     send_cmd(s)
228
229 def printer_positions():
230     for j in range(Y_COLUMNS):
231         for i in range(X_ROWS):
232             if j%2==1:
233                 yield (X_HOME + (X_ROWS - 1 - i) * X_SPACING, Y_HOME + j * Y_SPACING, Z_HOME)
234             else:
235                 yield (X_HOME + i * X_SPACING, Y_HOME + j * Y_SPACING, Z_HOME)
236

```

```

237 # Run this.
238 tube_location = list(printer_positions())
239
240 try:
241     for i in range(number_of_cycles):
242         move(*tube_location[2*i])
243         asked = opter.ask()
244         function(asked[0],asked[1],asked[2],asked[3],asked[4],asked[5],asked[6],asked[7],asked[8],asked[9],asked[10])
245         move(*tube_location[2*i+1])
246         told= function2()
247
248         print(f"area under the curve in the round {i:.2f} = {told:.2f}")
249         opter.tell(asked,-told*val)
250         results.append(told)
251         #flowrates_1.append(asked[0])
252         #flowrates_2.append(asked[1])
253         #flowrates_3.append(asked[2])
254         #flowrates_4.append(asked[3])
255         #flowrates_5.append(asked[4])
256         flowrates_6.append(asked[5])
257         #flowrates_7.append(asked[6])
258         flowrates_8.append(asked[7])
259         #flowrates_9.append(asked[8])
260         vs.append(asked[9])
261         currents.append(asked[10])
262
263         #dict1 = {"flowrate_2":flowrates_2,"flowrate_3":flowrates_3,"flowrate_4":flowrates_4,"flowrate_5":flowrates_5,"flowrate_6":flowrates_6,"flowrate_7":flowrates_7,"flowrate_8":flowrates_8,"flowrate_9":flowrates_9
264         dict1 = {"flowrate_6":flowrates_6,"flowrate_8":flowrates_8,"voltages":vs,"currents":currents,"area-results":results}
265         df2 = pd.DataFrame(dict1)
266         df2.to_csv("output round "+str(i)+".csv")
267 finally:
268     #pump_1.write(b'stop\r')
269     #pump_2.write(b'stop\r')
270     #pump_3.write(b'stop\r')
271     #pump_4.write(b'stop\r')
272     #pump_5.write(b'off\r')
273     pump_6.write(b'off\r')
274     #pump_7.write(b'off\r')
275     pump_8.write(b'off\r')
276     #pump_9.write(b'off\r')
277

```

AB7.py (for diazoetherate)

```
AB7.py
1
2 """
3 Created on Wed Aug 30 14:48:13 2023
4
5 @author: Admin
6 """
7
8
9 from os import listdir
10 from os.path import isfile, join
11 import serial
12
13 import numpy as np
14 import pandas as pd
15 import time
16 from scipy.integrate import trapz
17
18 #step1: be sure to the address of the files that the ftir data is exported is matching to line 11 (mypath)
19 mypath = "C:\\Users\\Admin\\Desktop\\ruchi\\Exp 2024-04-11 13-57"
20 onlyfiles = [f for f in listdir(mypath) if isfile(join(mypath, f))]
21
22
23 #step2: make sure that pump and the potentiostat is correctly addressed in the line 16 and 17
24 #pump_1 = serial.Serial("COM10",9600) #hwSyringe Pump
25 #pump_2 = serial.Serial("COM16",9600) #Syringe Pump
26 #pump_3 = serial.Serial("COM19",9600) #Syringe Pump
27 #pump_4 = serial.Serial("COM15",9600) #Syringe Pump
28 #pump_5 = serial.serial_for_url("COM29",9600) #HPLC Pump LAN
29 #pump_6 = serial.serial_for_url("socket://169.254.225.18:10001",9600)
30 port = serial.Serial("COM1",115200)
31 pump_7 = serial.Serial("COM28",9600) #HPLC Pump
32 pump_8 = serial.Serial("COM30",9600) #HPLC Pump
33 #pump_9 = serial.Serial("COM5",9600) #HPLC Pump
34 printer = serial.Serial("COM14", 115200, timeout=1)
35
36 r_1=14.25 #radius of syringe used (50ml,14.25|20ml,9.6|10ml,7.25|5ml,6.03|3ml,4/3|1ml,2.39|60ml,13.36)
37 r_2=14.25
38 r_3=14.25
39 r_4=14.25
40 #pump_1_r=(14.25/r_1)**2
41 #pump_2_r=(14.25/r_2)**2
42 #pump_3_r=(14.25/r_3)**2
43 #pump_4_r=(14.25/r_4)**2
44
```

```

45 #step 3: grab the lines from 22 to 90 and press f9
46 def area_under(data,start,end):
47     x = np.flip(data.iloc[start:end,0].to_numpy())
48     y = np.flip(data.iloc[start:end,1].to_numpy())
49     area = trapz(y,x)
50     return np.abs(area)
51
52 def file_namer(num):
53     str1 = str(num)
54     length = int(len((str1)))
55     empt = ''
56     for i in range(5-length):
57         empt = empt+'0'
58
59     return empt+str1
60
61
62 def ftir_extract(filename,init,end):
63
64     filename = filename
65
66     temp_df = pd.read_csv(filename)
67     # nump_df = temp_df.to_numpy()
68     area = area_under(temp_df, init, end)
69     # max_peak = np.max(nump_df[90:120,1])
70     print(area)
71
72     return area
73
74
75 def function(flowrate_1,flowrate_2,flowrate_3,flowrate_4,flowrate_5,flowrate_6,flowrate_7,flowrate_8,flowrate_9,v,i):
76
77     #set the pumps with the flowrate as the desired flowrate for the function
78
79     #fr_1 = flowrate_1*1000*pump_1_k #mL/min to microliter/min and syringe correction factor
80     #fr_2 = flowrate_2*1000*pump_2_k #mL/min to microliter/min and syringe correction factor
81     #fr_3 = flowrate_3*1000*pump_3_k #mL/min to microliter/min and syringe correction factor
82     #fr_4 = flowrate_4*1000*pump_4_k #mL/min to microliter/min and syringe correction factor
83     #time.sleep(0.1)
84     #fr_5 = flowrate_5*1000 #mL/min
85     #pump_5.write(('flow:'+ str(fr_5) + '\r').encode())
86

```

```

87  #time.sleep(0.1)
88  #fr_6=flowrate_6*1000 #converting ml/min to ul/min as pump takes this as input val
89  #pump_6.write(('flow:'+ str(fr_6) + '\r').encode())
90
91  time.sleep(0.1)
92  fr_7 = flowrate_7*1000  #ml/min
93  pump_7.write(('flow:'+ str(fr_7) + '\r').encode())
94
95  time.sleep(0.1)
96  fr_8=flowrate_8*1000 #converting ml/min to ul/min as pump takes this as input val
97  pump_8.write(('flow:'+ str(fr_8) + '\r').encode())
98
99  #time.sleep(0.1)
100 #fr_9=flowrate_9*1000 #converting ml/min to ul/min as pump takes this as input val
101 #pump_9.write(('flow:'+ str(fr_9) + '\r').encode())
102 #time.sleep(0.1)
103
104 #set the com port for potentiostat and set the voltage and current
105 vol = v
106 curr = i
107 port.write(('VOLT '+str(vol)+'\r\n').encode())  #to change the vltge we need to use "VOLT 1" command
108 port.write(('CURR '+str(curr)+'\r\n').encode())  #to change the current we need to use "CURR 1" command
109
110
111 #pumps run
112 #pump_5.write(b'on\r')
113 #time.sleep(0.1)
114 #pump_6.write(b'on\r')
115 #time.sleep(0.1)
116 pump_7.write(b'on\r')
117 time.sleep(0.1)
118 pump_8.write(b'on\r')
119 time.sleep(0.1)
120 #pump_9.write(b'on\r')
121 #time.sleep(0.1)
122

```

```

123 #send_syr_command_1(f'fr{fr_1}')
124 #send_syr_command_2(f'fr{fr_2}')
125 #send_syr_command_3(f'fr{fr_3}')
126 #send_syr_command_4(f'fr{fr_4}')
127 time.sleep(6600)
128 def function2():
129     time.sleep(6600) # Last 3 minutes
130     files = [f for f in listdir(mypath) if isfile(join(mypath, f))]
131
132     val = 0
133
134     #change wavelengths as per product here.
135     f_row=17 #first row of range for wavelength as per IR CSV
136     l_row=24 #last row of range for wavelength as per IR CSV
137
138     val += ftir_extract(files[-1],f_row,l_row)
139     val += ftir_extract(files[-2],f_row,l_row)
140     val += ftir_extract(files[-3],f_row,l_row)
141
142     avg_val = val/3
143
144     return avg_val
145
146
147 #step 4:grab the line 93 and f9
148 from skopt.optimizer import Optimizer
149
150 #def send_syr_command_1(command):
151     #pump_1.write(command.encode() + b'\n') # Send the command to Arduino with a newline character
152     #pump_1.flush() # Flush the serial buffer
153
154 #def send_syr_command_2(command):
155     #pump_2.write(command.encode() + b'\n') # Send the command to Arduino with a newline character
156     #pump_2.flush() # Flush the serial buffer
157
158 #def send_syr_command_3(command):
159     #pump_3.write(command.encode() + b'\n') # Send the command to Arduino with a newline character
160     #pump_3.flush() # Flush the serial buffer
161 #def send_syr_command_4(command):

```



```

162 #pump_4.write(command.encode() + b'\n') # Send the command to Arduino with a newline character
163 #pump_4.flush() # Flush the serial buffer
164 #step5: in line 96 we have to define the range that (flowrate,voltage,current) (from,to) and after (anytime) applying changes you need to grab the line 96 and f9
165 #flowrates are in mL/min, voltage in V, Current in Ampere
166 bounds = [(-10.0,-5.0),(0.1,0.2),(0.1,0.2),(0.1,0.2),(0.1,0.2),(0.039,0.040),(0.039,0.040),(0.1,0.2),(14.49,14.50),(4.99,5.00)]
167 #step 6: grab the line 100 and f9
168 opter =Optimizer(bounds,base_estimator='gp',n_initial_points=3,acq_func="EI",random_state=np.random.randint(3326))
169
170
171 #step7: to selecte number of the cycles that you have to do the experiment and then grab the line 104 to 121 and f9: the closed loop experimentation is initiated
172 number_of_cycles =5 #max is 5 cycles for 10 tet tubes
173
174 results = []
175 #flowrates_1 = []
176 #flowrates_2 = []
177 #flowrates_3 = []
178 #flowrates_4 = []
179 #flowrates_5 = []
180 #flowrates_6 = []
181 flowrates_7 = []
182 flowrates_8 = []
183 #flowrates_9 = []
184 vs = []
185 currents = []
186
187 product_wavelength=True #set to true if product wavelengths being monitored
188
189 if product_wavelength == True:
190     val=1
191 else:
192     val=-1
193
194
195
196 # Step 8: Test Tubes on Printer
197 USE_PRINTER = True
198 REST_HEIGHT = 200
199 X_HOME = -5
200 Y_HOME = 20
201 Z_HOME = 175
202 DEFAULT_PUMP_TIME="1"

```

```

203 # Distance between test tubes
204 X_SPACING=20
205 Y_SPACING=20
206 # Number of test tubes
207 X_ROWS = 11
208 Y_COLUMNS = 4
209
210 #-----code for changing column for run---
211 column_num=7 #column number of run to try from
212 Y_HOME=Y_HOME+(column_num-1)*Y_SPACING
213 #-----
214
215
216 def send_cmd(cmd):
217     print(cmd)
218     printer.write(f"{cmd}\n".encode("ASCII"))
219
220 def move(x=None, y=None, z=None):
221     s = "G0"
222     if x is not None:
223         s += f"X{x}"
224     if y is not None:
225         s += f"Y{y}"
226     if z is not None:
227         s += f"Z{z}"
228
229     s+= "F5000"
230     send_cmd(s)
231
232 def printer_positions():
233     for j in range(Y_COLUMNS):
234         for i in range(X_ROWS):
235             if j%2==1:
236                 yield (X_HOME + (X_ROWS - 1 - i) * X_SPACING, Y_HOME + j * Y_SPACING, Z_HOME)
237             else:
238                 yield (X_HOME + i * X_SPACING, Y_HOME + j * Y_SPACING, Z_HOME)
239
240 # Run this.
241 tube_location = list(printer_positions())
242

```

```

243 try:
244     for i in range(number_of_cycles):
245         move(*tube_location[2*i])
246         asked = opter.ask()
247         function(asked[0],asked[1],asked[2],asked[3],asked[4],asked[5],asked[6],asked[7],asked[8],asked[9],asked[10])
248         move(*tube_location[2*i+1])
249         told= function2()
250
251         print(f"area under the curve in the round {i:.2f} = {told:.2f}")
252         opter.tell(asked,-told*val)
253         results.append(told)
254         #flowrates_1.append(asked[0])
255         #flowrates_2.append(asked[1])
256         #flowrates_3.append(asked[2])
257         #flowrates_4.append(asked[3])
258         #flowrates_5.append(asked[4])
259         #flowrates_6.append(asked[5])
260         flowrates_7.append(asked[6])
261         flowrates_8.append(asked[7])
262         #flowrates_9.append(asked[8])
263         vs.append(asked[9])
264         currents.append(asked[10])
265
266         #dict1 = {"flowrate_2":flowrates_2,"flowrate_3":flowrates_3,"flowrate_4":flowrates_4,"flowrate_5":flowrates_5,"flowrate_6":flowrates_6,"flowrate_7":flowrates_7,"flowrate_8":flowrates_8,"flowrate_9":flowrates_9
267         dict1 = {"flowrate_7":flowrates_7,"flowrate_8":flowrates_8,"voltages":vs,"currents":currents,"area-results":results}
268         df2 = pd.DataFrame(dict1)
269         df2.to_csv("output round "+str(i)+".csv")
270 finally:
271     #pump_1.write(b'stop\r')
272     #pump_2.write(b'stop\r')
273     #pump_3.write(b'stop\r')
274     #pump_4.write(b'stop\r')
275     #pump_5.write(b'off\r')
276     #pump_6.write(b'off\r')
277     pump_7.write(b'off\r')
278     pump_8.write(b'off\r')
279     #pump_9.write(b'off\r')
280

```

S10.2. General procedure of Integrated AI optimized carbene insertion reaction into diazo ester photochemically.

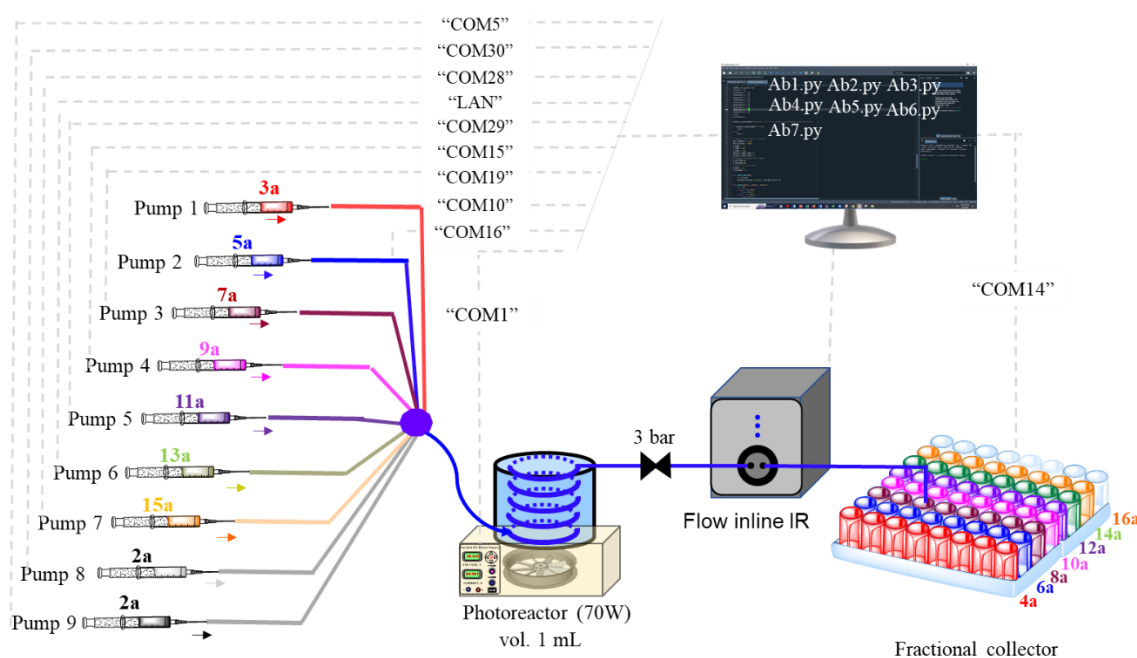


Figure S55. Schematic diagram of Integrated AI optimized carbene insertion reaction into diazo ester photochemically.

Our interest turned to integrating our protocol into a unified flow process akin to grafting techniques observed in hibiscus plants. In this process, reminiscent of the grafting of branches onto a common trunk, we maintain the same starting material and combine it with different substrates to yield respective products, each designated with a distinct colour. We coined this process "digichemtree," where we sequentially run seven different reagents, each mixed with the starting material to obtain a single product at a time.

Before commencing the experiment, we prepared 0.1 M stock solutions of compound **2a** in both EtOAc and DCE, each stored in separate conical flasks covered with aluminum foil. Similarly, we prepared 0.2 M stock solutions of compounds **3a**, **5a**, and **7a** in EtOAc, each loaded into syringes and connected to pumps. Additionally, 0.2 M stock solutions of compounds **9a**, **11a**, **13a**, and **15a** in DCE were prepared, also stored in separate conical flasks

and connected to pumps. After ensuring all connections were secure, we checked communication ports with the master computer. Our integrated continuous flow setup, coupled with AI, was now prepared for operation. Upon executing the code AB1.py, pumps 1 and 9 were activated. Pump 1 dispensed the 0.2 M solution of **3a** in EtOAc, while pump 9 in EtOAc dispensed the 0.1 M solution of **2a**, each at an optimized flow rate of 50 $\mu\text{L}/\text{min}$. The resulting reaction mixture flowed through a 1 mL PFA tubular reactor, exposed to blue LED illumination (70 watts), with a residence time of 10 minutes under 3 bar pressure. Upon completion, the first drop of product emerged, and subsequent solution was collected in individual test tubes over a period of 6000 seconds using tailor made auto fraction collector. Following the first test tube of product, the subsequent ones were reserved for external analysis.

After the successful execution of the AB1.py experiment code, we proceeded to test the AB2.py code. This time, pumps 2 and 9 were activated. Pump 2 delivered the 0.2 M solution of **5a** in EtOAc, while pump 9 dispensed the 0.1 M solution of **2a** in EtOAc, each at an optimized flow rate of 86 $\mu\text{L}/\text{min}$. The resulting reaction mixture traversed through a 1 mL PFA tubular reactor, subjected to intense blue LED illumination (70 watts), with a residence time of 11 minutes under 3 bar pressure. As the reaction concluded, the initial drop of product appeared, and the ensuing solution was meticulously collected in individual test tubes over a span of 3300 seconds utilizing a tailor-made auto fraction collector. Following the first test tube of product, the subsequent ones were reserved for external analysis.

Energized by the successful outcomes of the preceding experiments, we were keen to explore our AB3.py code for a photochemical N-H insertion reaction. Initiating the AB3.py experiment code prompted pump 3 to dispense the 0.2 M solution of **7a** in EtOAc, while pump 9 delivered the 0.1 M solution of **2a** in EtOAc, each flowing at an optimized rate of 46 $\mu\text{L}/\text{min}$. The resultant reaction mixture journeyed through a 1 mL PFA tubular reactor, bathed in vibrant blue LED illumination (70 watts), with a residence time of 10.86 minutes under 3 bar pressure.

Upon completion, the droplet of product emerged, and the successive solution was gathered into individual test tubes over a period of 6000 seconds via the tailor-made auto fraction collector. Following the first test tube of product, the subsequent ones were reserved for external analysis.

Having accomplished three successful experimental runs, we were tantalized by the prospect of investigating the feasibility of the remaining experiment codes. We started with clicking on experiment code AB4.py, now pump 4 and pump 8 is activated, pumping the 0.2 M solution of reagent **9a** in DCE with pump 4 and 0.1 M solution of compound **2a** in DCE each flowing at an optimized rate of 15 $\mu\text{L}/\text{min}$. The resultant reaction mixture journeyed through a 1 mL PFA tubular reactor, bathed in vibrant blue LED illumination (70 watts), with a residence time of 32.0 min under 3 bar pressure. Upon completion, the droplet of product emerged, and the successive solution was gathered into individual test tubes over a period of 15000 seconds via the tailor-made auto fraction collector. Following the first test tube of product, the subsequent ones were reserved for external analysis. Next we have run experiment code AB5.py This time, pumps 5 and 8 were activated. Pump 5 delivered the 0.2 M solution of **11a** in DCE, while pump 8 dispensed the 0.1 M solution of **2a** in DCE, each at an optimized flow rate of 12.5 $\mu\text{L}/\text{min}$. The resulting reaction mixture traversed through a 1 mL PFA tubular reactor, subjected to intense blue LED illumination (70 watts), with a residence time of 40 min under 3 bar pressure. As the reaction concluded, the initial drop of product appeared, and the ensuing solution was meticulously collected in individual test tubes over a span of 3300 seconds utilizing a tailor-made auto fraction collector. Following the first test tube of product, the subsequent ones were reserved for external analysis.

Then for [3+2] cycloaddition reaction have done utilizing experiment code AB6.py, in which upon executing, pumps 6 and 8 were activated. Pump 6 dispensed the 0.2 M solution of **13a** in DCE, while pump 8 dispensed the 0.1 M solution of **2a** in DCE, each at an optimized flow rate

of 31 $\mu\text{L}/\text{min}$. The resulting reaction mixture flowed through a 1 mL PFA tubular reactor, exposed to blue LED illumination (70 watts), with a residence time of 16.6 minutes under 3 bar pressure. Upon completion, the first drop of product emerged, and subsequent solution was collected in individual test tubes over a period of 8400 seconds using tailor made auto fraction collector. Following the first test tube of product, the subsequent ones were reserved for external analysis.

In last for cross coupling reaction experiment code AB7.py is subjected and pump 7 and pump 8 is activated, pumping the 0.2 M solution of reagent **15a** in DCE with pump 8 and 0.1 M solution of compound **2a** in DCE each flowing at an optimized rate of 40 $\mu\text{L}/\text{min}$. The resultant reaction mixture journeyed through a 1 mL PFA tubular reactor, bathed in vibrant blue LED illumination (70 watts), with a residence time of 12.5 minutes under 3 bar pressure. Upon completion, the droplet of product emerged, and the successive solution was gathered into individual test tubes over a period of 6600 seconds via the tailor-made auto fraction collector. Following the first test tube of product, the subsequent ones were reserved for external analysis.

S12. Troubleshooting

Question: Where to buy AI-system for the reaction optimization?

Answer: No need to buy from any place the python code is available in supporting information.

Question: Where to buy the 3D printer and Arduino board?

Answer: We bought through Amazon.

Question: Any other company photo-flow reactor will work?

Answer: Yes, it'll work but need to change the python code accordingly.

Question: Does other tube id work?

Answer: Other higher tube id (e.g., 2-5 mm) also work, but further need to optimize to get the optimal condition.

Question: Does code will work for the production 100 kg scale?

Answer: Yes, it'll work but further auto-optimization may give you better reaction condition.

Question: Does current code will work with any other company hardware set-up?

Answer: Yes, it'll work but slight modification needed in python code.

Question: Can we replace the In-line IR with other analysis unit?

Answer: Yes, you can replace with HPLC, In-line NMR, GC-MS, LC-MS, XAS etc but the analysis code need to rewrite as per the analysis system manual.

Question: Do we need the super computer to run the auto-reaction optimization?

Answer: We don't need any supercomputer. Any normal PC will work for the reaction auto-optimization and on demand synthesis.

Question: Optimization will have effected by the current failure?

Answer: Yes, before starting the experiment make sure system have proper battery back-up.

Question: How many pump we can connect with one AI system?

Answer: Infinite.

S13. Supplementary references:

- 1 Rana, A., Chauhan, R. & Singh, A. K. Thermal/photochemical micro-flow probe system for direct C–H bond functionalization of biologically active molecules. *React. Chem. Eng.* **9**, 1313-1319, doi:10.1039/D4RE00083H (2024).
- 2 Jurberg, I. D. & Davies, H. M. L. Blue light-promoted photolysis of aryldiazoacetates. *Chem. Sci.* **9**, 5112-5118, doi:10.1039/C8SC01165F (2018).
- 3 Tan, F. *et al.* Asymmetric Catalytic Insertion of α -Diazo Carbonyl Compounds into O–H Bonds of Carboxylic Acids. *ACS Catal.* **6**, 6930-6934, doi:10.1021/acscatal.6b02184 (2016).
- 4 Keipour, H., Jalba, A., Delage-Laurin, L. & Ollevier, T. Copper-Catalyzed Carbenoid Insertion Reactions of α -Diazoesters and α -Diazoketones into Si–H and S–H Bonds. *J. Org. Chem.* **82**, 3000-3010, doi:10.1021/acs.joc.6b02998 (2017).
- 5 Keipour, H., Jalba, A., Tanbouza, N., Carreras, V. & Ollevier, T. α -Thiocarbonyl synthesis via the FeII-catalyzed insertion reaction of α -diazocarbonyls into S–H bonds. *Org. Biomol. Chem.* **17**, 3098-3102, doi:10.1039/C9OB00261H (2019).
- 6 Pan, J. B. *et al.* A Spiro Phosphamide Catalyzed Enantioselective Proton Transfer of Ylides in a Free Carbene Insertion into N–H Bonds. *Angew. Chem.* **135**, e202300691 (2023).
- 7 Wang, B., Howard, I. G., Pope, J. W., Conte, E. D. & Deng, Y. Bis(imino) pyridine iron complexes for catalytic carbene transfer reactions. *Chem. Sci.* **10**, 7958-7963 (2019).
- 8 Wang, B., Howard, I. G., Pope, J. W., Conte, E. D. & Deng, Y. Bis(imino)pyridine iron complexes for catalytic carbene transfer reactions. *Chem. Sci.* **10**, 7958-7963, doi:10.1039/C9SC02189B (2019).

- 9 Dasgupta, A. *et al.* Borane-catalyzed stereoselective C–H insertion, cyclopropanation, and ring-opening reactions. *Chem* **6**, 2364-2381 (2020).
- 10 Mancinelli, J. P. & Wilkerson-Hill, S. M. Tris(pentafluorophenyl)borane-Catalyzed Cyclopropanation of Styrenes with Aryldiazoacetates. *ACS Catal.* **10**, 11171-11176, doi:10.1021/acscatal.0c03218 (2020).
- 11 Sharland, J. C. *et al.* Asymmetric synthesis of pharmaceutically relevant 1-aryl-2-heteroaryl- and 1,2-diheteroarylcyclopropane-1-carboxylates. *Chem. Sci.* **12**, 11181-11190, doi:10.1039/D1SC02474D (2021).
- 12 Davies, H. M. & Venkataramani, C. Dirhodium tetraproline-catalyzed asymmetric cyclopropanations with high turnover numbers. *Org. Lett.* **5**, 1403-1406 (2003).
- 13 Singha, S., Buchsteiner, M., Bistoni, G., Goddard, R. & Fürstner, A. A New Ligand Design Based on London Dispersion Empowers Chiral Bismuth–Rhodium Paddlewheel Catalysts. *J. Am. Chem. Soc.* **143**, 5666-5673 (2021).
- 14 Chen, L. *et al.* An Inexpensive and Recyclable Silver-Foil Catalyst for the Cyclopropanation of Alkenes with Diazoacetates under Mechanochemical Conditions. *Angew. Chem. Int. Ed.* **54**, 11084-11087, doi:<https://doi.org/10.1002/anie.201504236> (2015).
- 15 Hommelsheim, R., Guo, Y., Yang, Z., Empel, C. & Koenigs, R. M. Blue-Light-Induced Carbene-Transfer Reactions of Diazoalkanes. *Angew. Chem. Int. Ed.* **58**, 1203-1207, doi:<https://doi.org/10.1002/anie.201811991> (2019).
- 16 Li, C., Zeng, Y. & Wang, J. Au-catalyzed isomerization of cyclopropenes: a novel approach to indene derivatives. *Tetrahedron Lett.* **50**, 2956-2959 (2009).
- 17 Chen, L., Leslie, D., Coleman, M. G. & Mack, J. Recyclable heterogeneous metal foil-catalyzed cyclopropanation of alkynes and diazoacetates under solvent-free

- mechanochemical reaction conditions. *Chem. Sci.* **9**, 4650-4661, doi:10.1039/C8SC00443A (2018).
- 18 Stefkova, K., Heard, M. J., Dasgupta, A. & Melen, R. L. Borane catalysed cyclopropanation of arylacetylenes. *Chem. Comm.* **57**, 6736-6739 (2021).
- 19 Muriel, B. & Waser, J. Azide radical initiated ring opening of cyclopropenes leading to alkenyl nitriles and polycyclic aromatic compounds. *Angew. Chem. Int. Ed.* **60**, 4075-4079 (2021).
- 20 Chuprakov, S., Rubin, M. & Gevorgyan, V. Direct palladium-catalyzed arylation of cyclopropenes. *J. Am. Chem. Soc.* **127**, 3714-3715 (2005).
- 21 Saha, A. *et al.* Photoinduced [3+2] Cycloaddition of Carbenes and Nitriles: A Versatile Approach to Oxazole Synthesis. *Angew. Chem. Int. Ed.* **62**, e202308916, doi:<https://doi.org/10.1002/anie.202308916> (2023).
- 22 Maiti, D., Saha, A., Guin, S., Maiti, D. & Sen, S. Unveiling catalyst-free electro-photochemical reactivity of aryl diazoesters and facile synthesis of oxazoles, imide-fused pyrroles and tetrahydro-epoxy-pyridines via carbene radical anions. *Chem. Sci.* **14**, 6216-6225, doi:10.1039/D3SC00089C (2023).
- 23 Bai, J. *et al.* Visible light-induced synthesis of polysubstituted oxazoles from diazo compounds. *Org. Biomol. Chem.* **21**, 5511-5515, doi:10.1039/D3OB00878A (2023).
- 24 Xiao, T., Mei, M., He, Y. & Zhou, L. Blue light-promoted cross-coupling of aryldiazoacetates and diazocarbonyl compounds. *Chem. Comm.* **54**, 8865-8868, doi:10.1039/C8CC04609C (2018).
- 25 Upp, D. M. *et al.* Engineering Dirhodium Artificial Metalloenzymes for Diazo Coupling Cascade Reactions**. *Angew. Chem. Int. Ed.* **60**, 23672-23677, doi:<https://doi.org/10.1002/anie.202107982> (2021).