Electronic Supplementary Material (ESI) for Digital Discovery.

Use of open-source software platform to develop dashboards for control and automation of flow chemistry equipment

Supporting Information

C. Johan van der Westhuizen[1], Jurie du Toit[2], Nicole Neyt[1], Darren Riley[3], Jenny-Lee Panayides[1*]

1) Council for Scientific and Industrial Research (CSIR), Future Production: Chemicals Cluster, Meiring Naude Road, Brummeria, Pretoria.
2) Council for Scientific and Industrial Research (CSIR), Future Production: Manufacturing Cluster, Meiring Naude Road, Brummeria, Pretoria.
3) Department of Chemistry, Faculty of Natural and Agricultural Sciences, University of Pretoria, Lynnwood Road, Pretoria, South Africa
* Corresponding author: JPanayides@csir.co.za

Supporting Information

Table of Contents

1 Introduction

The implementation of the dashboards (Node-RED and Graphana) was done using a Raspberry Pi. The autonomous closed-loop optimisation was done using Python, with the Summit library for the optimisation algorithm.

The Raspberry Pi was selected as a low-cost hardware solution. It runs Ubuntu operating system, which is also free.

Node-RED is an ideal solution where a low-cost control platform is required. The programming of the Node-RED code also enables "easier" programming for beginners and novices. The Node-RED community was developed various subflows and custom nodes that can be downloaded for free.

The Python optimization script has been included in this manuscript to provide a framework for others to develop their own closed-loop procedure. *Summit*,[1] an open-source python library, provides an intuitive syntax for using different optimisation algorithms as well as developing models of previously generated experimental data.

2 Experimental

Materials used for the reactions and calibrations were purchased from SIGMA aldrich. Isovanillin (621-59-0) 99 % purity and >95% purity, Allylbromide (106-95-6) reagent grade, 97%, contains ≤1000 ppm propylene oxide as stabilizer. Reagent grade potassium carbonate and HPLC grade DMF was purchased from RadChem.

3 Raspberry Pi Installation

The requirements for the hardware installation are:

1. Raspberry Pi
2. Micro-SD card (>= 32GB)
3. PC to download and install the Rasbian / Ubuntu Image on the micro-SD card

Follow the instruction in one of the following links to install the Raspbian Image to the micro-SD card:

https://projects.raspberrypi.org/en/projects/raspberry-pi-setting-up/2

https://www.raspberrypi.org/documentation/computers/getting-started.html

If the Ubuntu Operating System is preferred, the following link can be followed to install the Ubuntu Image on the micro-SD card:

https://www.techradar.com/how-to/how-to-install-ubuntu-on-the-raspberry-pi

4 Node-RED Installation

Once the hardware and the Operating System are installed, the Node-RED environment/platform can be installed.

Use the following link for a description to install the Node-RED platform on the Raspberry Pi:

https://nodered.org/docs/getting-started/raspberrypi

Make sure that all the steps are followed to allow Node-RED to start on boot-up.

5 Node-RED Application Description

5.1 Overview

Users are encouraged to first become familiar with Node-RED by completing the tutorials and going through the documentation on the Node-RED website (https://nodered.org/docs/). YouTube is another valuable platform that contains several tutorials and demonstrations of the Node-RED platform. The following playlists for Node-RED tutorials are suggested:

- https://www.youtube.com/playlist?list=PLyNBB9VCLmo1hyO-4fIZ08gqFcXBkHy-6
- https://www.youtube.com/playlist?list=PLKYvTRORAnx6a9tETvF95o35mykuysuOw

The software to implement the automated optimization of the flow chemistry processes are broken up into the following parts:

1. UI components and software flows to implement the dashboard where the parameters can be set, and to present data received from the flow chemistry equipment on graphs and gauges.
2. Sub-Flows implementing the connection to different manufacturers of flow chemistry equipment.
3. Main software flows implementing the process of repeated testing/experiments with different parameters until an optimum solution is found. This part is where the parameters are being set using the JSON "script"
4. The external Python program is used in conjunction with the previous point to calculate the new parameters for the next test run/optimization cycle.

5.2 Dashboard

Flows are implemented to display variables, which is retrieved from the flow chemistry equipment, using UI components e.g. LEDs, on gauges or graphs. Manual control of the parameters of the flow chemistry equipment is also possible, allowing the user to set temperatures, flow rates, etc. by using UI components such as buttons, click—boxes, switches, etc. The dashboard also provides the ability for the user to record data to a database for later analysis and presentation.

The flow to implement this can be found below:

DB_Recording.json

Paste the JSON code into the import clipboard. Make sure to download and install the following palettes: MySQL (node-red-node-mysql), ui_led (node-red-contrib-ui-led), dashboard (node-red-dashboard). The mysql node will need to be configured with the credentials of an account from the database, to allow the parameters to be stored and accessed.

5.3 Flow Chemistry Equipment Sub-Flow Implementation

In Node-RED it is possible to create sub-flows. This is used to make your own custom UI components that will perform certain functions.

It is in these sub-flows where the user can implement the code for custom flow chemistry equipment. In this setup, flow chemistry equipment from Uniqsis was used. It is, however, possible to implement code to control any flow chemistry equipment if the connection interface is known and available.

5.3.1 Uniqsis HotChip

This implementation is for the Uniqsis HotChip device.

Attached is the Node-RED Sub-Flow for this device.

HotChip_Flow.json

The Sub-Flow accepts a text input “ON” or “OFF” to switch the state of the heater. It also implements a UI button to switch the HotChip’s heater manually. Once the input is processed, or the button input is handled, the state the heater should be in, is stored in a Global variable. The command is also sent to Uniqsis HotChip device.

The HTTP REST API interface was used to control this device. The msg.payload contains the command to be sent to the device, and the msg.topic contains the IP-address of the particular HotChip.

This sub-flow also periodically requests the temperature of the HotChip and displays it on a UI-gauge component. The temperature is also sent to the output of the Sub-Flow.

Here is an example of the HotChip output:

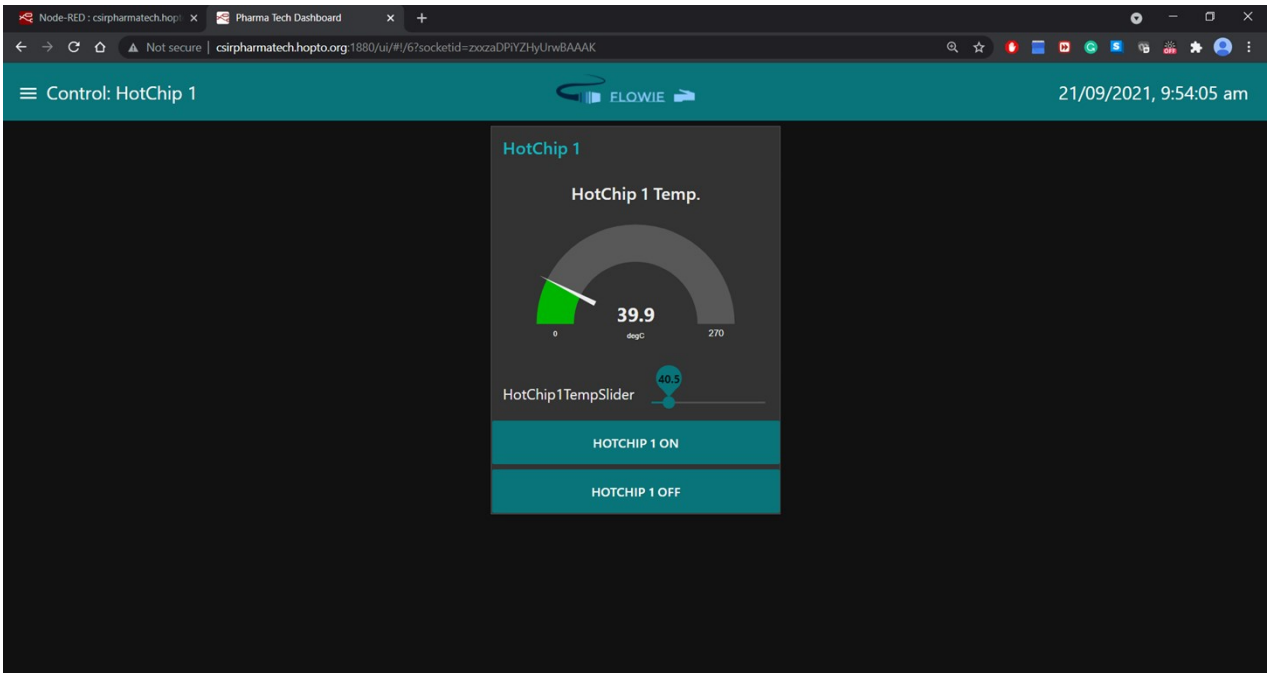


Figure S1. Dashboard for Uniqsis HotChip.

5.3.2 Uniqsis HotCoil

This implementation is for the Uniqsis HotCoil device. Attached is the Node-RED Sub-Flow for this device.

HotCoil_Flow.json

The Sub-Flow accepts a text input “ON” or “OFF” to switch the state of the heater. It also implements a UI button to switch the HotChoil’s heater manually. Once the input is processed, or the button input is handled, the state the heater should be in, is stored in a Global variable. The command is also sent to the Uniqsis HotChoil device.

The HTTP REST API interface was used to control this device. The msg.payload contains the command to be sent to the device, and the msg.topic contains the IP-address of the particular HotChip.

This Sub-Flow also periodically requests the temperature of the HotCoil, and displays it on a UI-gauge component. The temperature is also sent to the output of the Sub-Flow.

Here is an example of the HotCoil output:

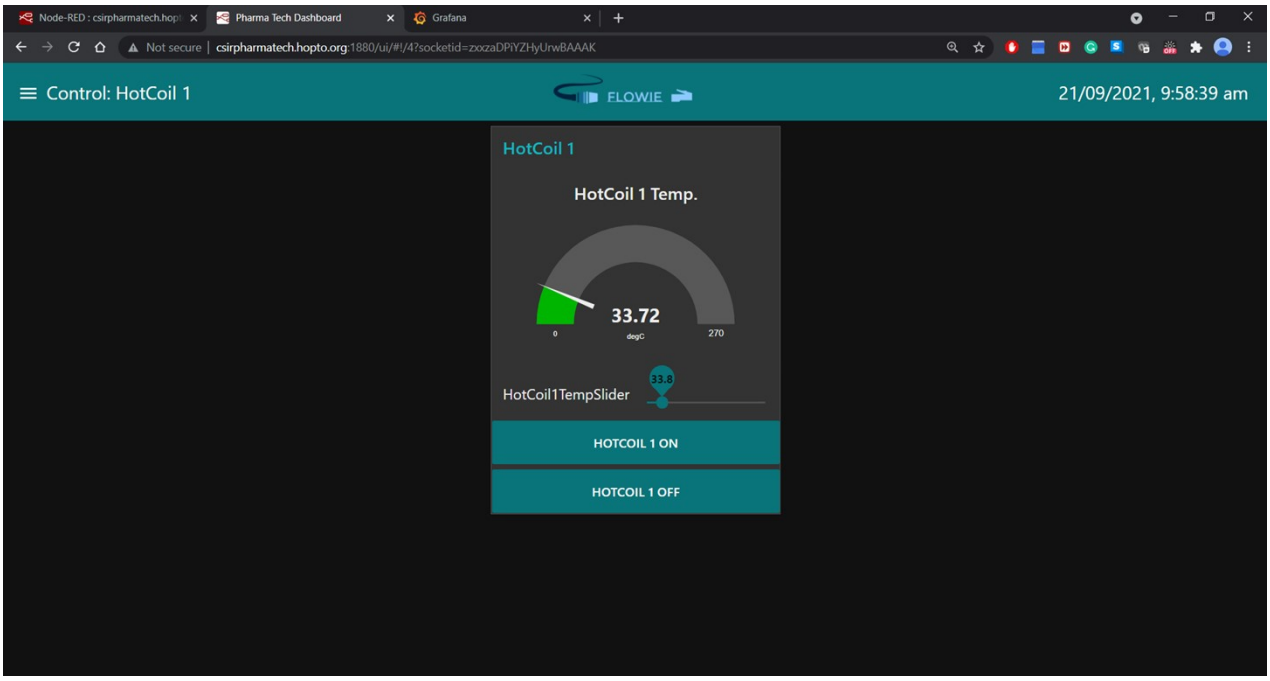


Figure S2. Dashboard for Uniqsis HotCoil.

5.3.3 Flow Syn Reactor

This implementation is for the Uniqsis Flow Syn reactor. Attached is the Node-RED Sub-Flow.

FlowSynMaxi_Flow.json

Make sure to download and install the following palette: throttle (node-red-contrib-throttle).

The Sub-Flow accepts several text input “commands” to set the following:

- 4 x Pump Flow Rates
- 4 x Reactor Temperatures
- 4 x Flow Syn Binary Valves
- 4 x Flow Syn Inj Valves
- 1 x CW (Collect/Waste) Valve
- Heater

For each of the input commands, there are also UI components with which the flow chemistry equipment can be manually manipulated. Once the input is processed, or the UI components are used to manually manipulate the device, the new parameter/setting is stored in a Global variable. The command is also sent to the Uniqsis Flow Syn Maxi device.

The HTTP REST API interface was used to control this device. The msg.payload contains the command to be sent to the device, and the msg.topic contains the IP address of the particular Flow Syn Maxi device.

This sub-flow also periodically requests the parameters of the Flow Syn device and displays it by using several UI components, e.g. gauge, graph, text, etc.

Here is an example of the Flow Syn reactor output:

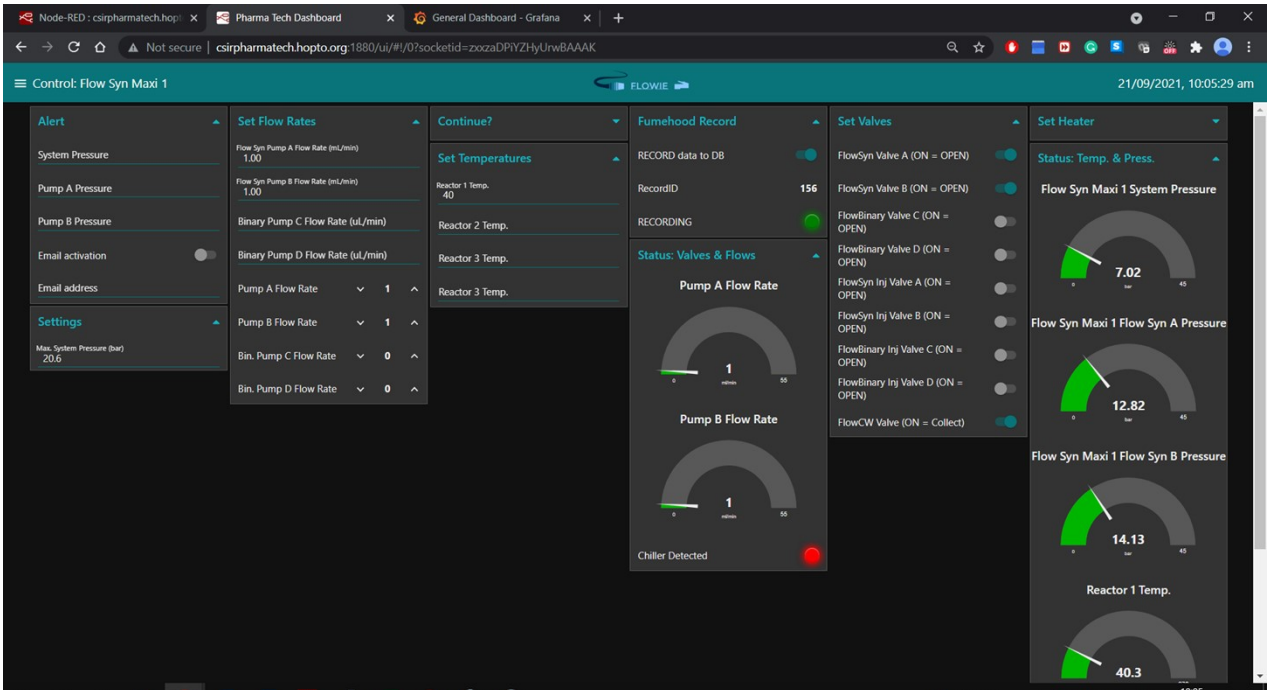


Figure S3. Dashboard for Uniqsis FlowSyn Reactor.

5.3.4 A complete Node-RED workflow for closed-loop automation of FlowSyn reactor

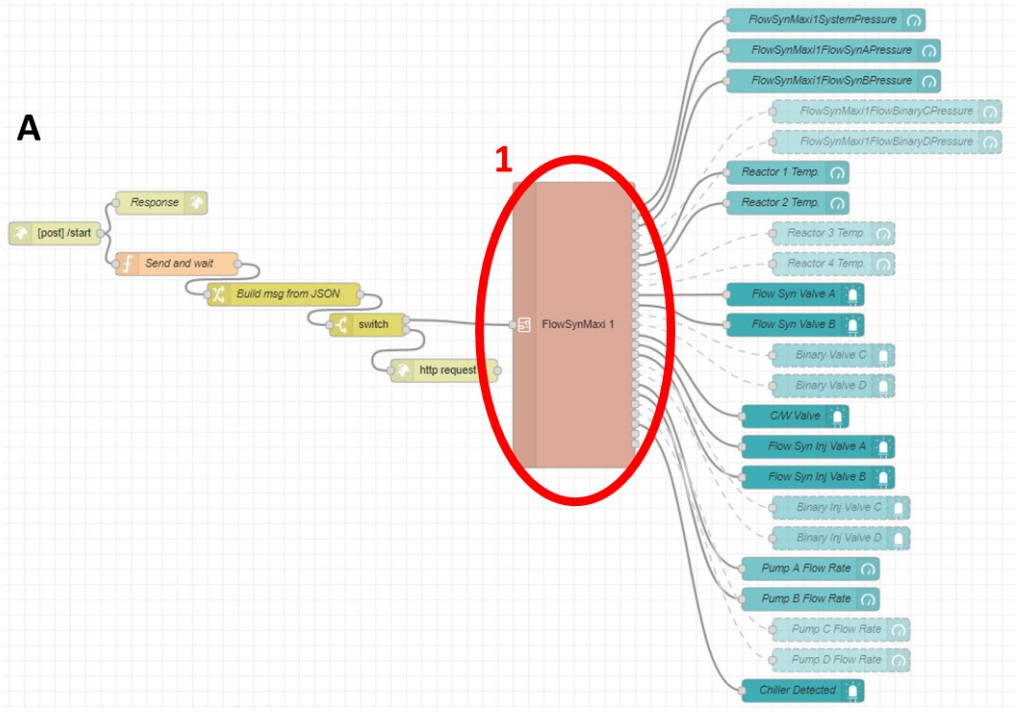
The flow was used for the autonomous closed-loop optimisation process:

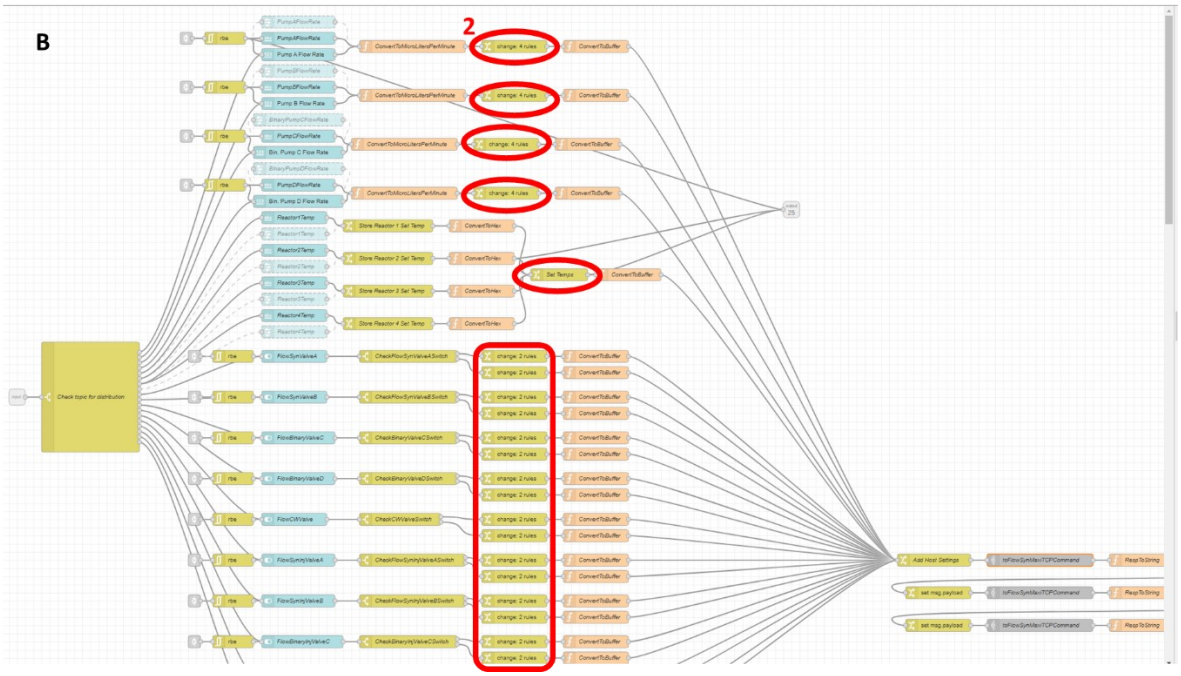
Closed-loop_dashboard.json

Make sure to download and install the following palettes: MySQL (node-red-node-mysql), ui_led (node-red-contrib-ui-led), dashboard (node-red-dashboard), e-mail (node-red-node-email), throttle (node-red-contrib-throttle), base64 (node-red-node-base64).

5.3.5 Adding proprietary commands

The proprietary commands from Uniqsis have been removed. Please contact Uniqsis for any equiries pertaining to these. Once these commands are know they can be added to the nodes for the Uniqsis equipment. For the example of the FlowSyn (Maxi) reactor, the subflow which is responsible for controlling the reactor is opened – Figure S4 A. The proprietary commands are incorporated into the 'change' nodes (Figure S4 B) and trigger nodes (Figure S4 C).


A
1
[post] /start
Response
Send and wait
Build msg from JSON
switch
http request
FlowSynMaxi 1
FlowSynMaxi1SystemPressure
FlowSynMaxi1FlowSynAPressure
FlowSynMaxi1FlowSynBPressure
FlowSynMaxi1FlowBinaryCPressure
FlowSynMaxi1FlowBinaryDPressure
Reactor 1 Temp.
Reactor 2 Temp.
Reactor 3 Temp.
Reactor 4 Temp.
Flow Syn Valve A
Flow Syn Valve B
Binary Valve C
Binary Valve D
C/W Valve
Flow Syn Inj Valve A
Flow Syn Inj Valve B
Binary Inj Valve C
Binary Inj Valve D
Pump A Flow Rate
Pump B Flow Rate
Pump C Flow Rate
Pump D Flow Rate
Chiller Detected


B
2

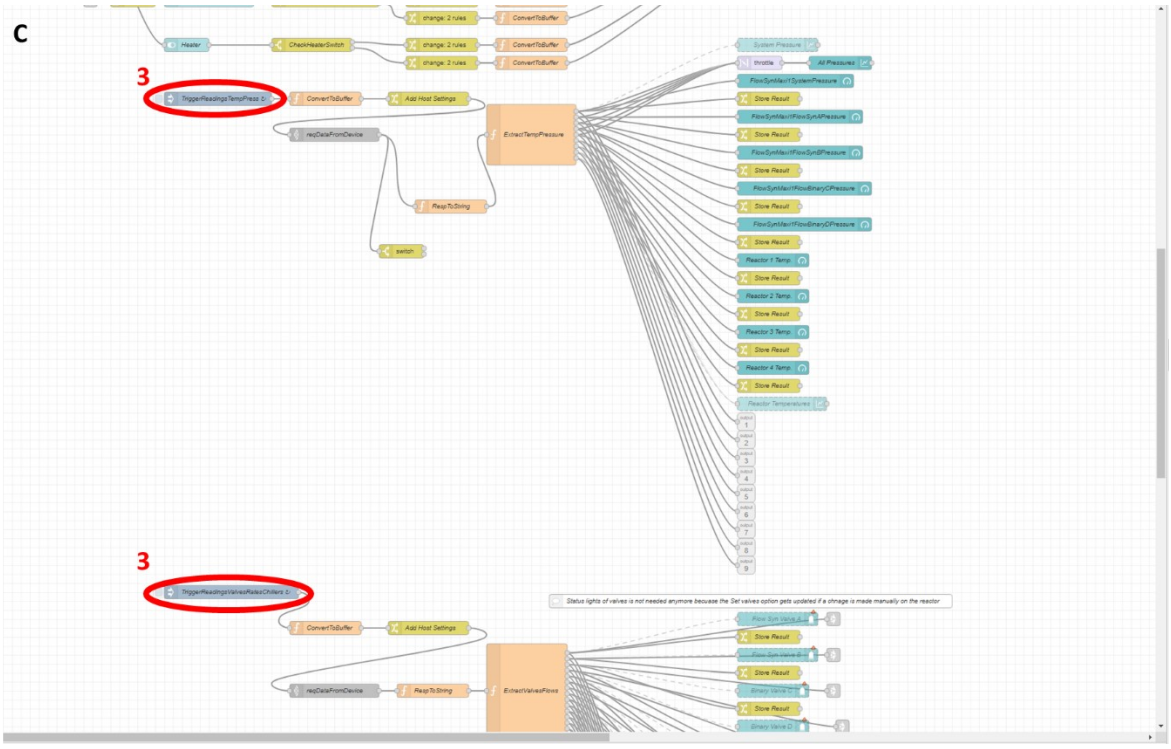


Figure S4. Location where proprietary Uniqsis commands need to be added for subflow, titled 'FlowSynMaxi 1', to function correctly.

5.4 Python Scripts for Closed-loop optimisation

The python code provided below was used for the closed-loop optimisation of STY. The *Summit* python library was used for the optimisation of the process.

Python code for Summit:

Summit_code.txt

Python class for reaction sequence execution and data processing:

Allylation_code.txt

Python code files can also be obtrained here: https://github.com/JohanvdWesthuizen/FlowChem-ClosedLoopOpt

The code titled 'Allyllation_code.txt' provides the python script which *Summit* uses to run the experiments in an automated closed-loop fashion. During each iteration of the closed-loop cycle, the script sends an HTTP request to the Node-RED server containing a JSON message to execute a sequence of commands. Three JSON messages are sent at different stages of each experiment iteration. The JSON files are provided below. When the temperature of the reactor needs to be set to a specified value, the python script will wait for an HTTP response from Node-RED before continuing with the script. This is to allow the reactor sufficient time to heat up (or cool down) to the specified temperature.

This JSON file is used as a template for sending commands to FlowSyn reactor:

json1.json

This JSON file is used for sending commands to the FlowSyn reactor to change the valves to solvents:

json1-solv_chg.json

This JSON file is used for sending commands to the FlowSyn reactor to clean up the reactor before the next reaction is executed:

json1_CleanUp.json

5.5 Grabbing data from HPLC computer

For the closed-loop automation, the python script connects to the computer which controls the HPLC via File Transfer Protocol (FTP) to grab the data from the HPLC analysis. Thus, an FTP server was setup for this HPLC computer using FileZilla Server. The software can be download for free and the graphical user interface (GUI) was used to set up a user account for secure access. Filezilla Server can be downloaded here:

https://filezilla-project.org/download.php?type=server

After each HPLC analysis is performed, the data is reported in a CSV file which reports the retention time and associated area of each peak. The area peak of selected peaks, along with the associated calibration plots are used to determine the concentration of the reagents and product.

5.6 Relay to trigger dilution and HPLC analysis

As described in the manuscript, a relay is used to trigger the Syrris Sampler and Dilutor (SAD) to (i) take a sample from the flow stream of the reactor, (ii) perform a dilution and (iii) inject it into the HPLC flow stream for analysis.

A SparkFun Qwiic Single Relay (https://www.sparkfun.com/products/15093) was connected to the Raspberry Pi GPIO pins using a Qwiic-Female Jumper cable (https://www.sparkfun.com/products/17261). A custom cable was made which linked the relay, SAD and HPLC. When analysis was required, the Raspberry

Pi sends a command to the relay using I^2C protocol and the relay performed a contract closure for two pins of the SAD I/O which triggers the dilution to begin. When the dilution was completed and the sample injected into the HPLC flow steam, the SAD I/O would trigger a contact closure between two pins of the remote port (DB-9 connection) at the back of the HPLC to start analysis.

6 SQL Database Description

The SQL server is responsible for the storage of the parameters of the reactor. This server can be run on the Raspberry Pi (as was done in this project), or on any other computer on the network.

Find attached the SQL text file which can be used to create the database used.

pharma_schema_structure.sql

7 Graphana Installation and Application Description

7.1 Graphana Installation

Use the following link for a description for the installing of Graphana on different operating systems:

https://grafana.com/docs/grafana/latest/installation/

7.2 Graphana Application

While Node-RED is able to show various parameters of the reactor in a timeline in a graph, we found this to slow down the dashboard. Because of this, it was decided to use Graphana for the visualisation of the parameters as it is more suited to this task than Node-RED.

Find attached the JSON file which can be used to create the dashboard which was used in this project.

GraphanaDashboard_export.json

Figure S5. Graphana Dashboard for used for representing parameters of the Uniqsis FlowSyn Reactor on a timeline.

8 Network Settings

For a user to access the dashboard from the Node-RED or Graphana server, the IP address of the server is needed. We have tested two ways to circumvent this issue. One approach to simplify this issue is to assign the server a static IP address. The second approach utilises a Domain Name System (DNS) which stores the dynamic IP address of the Raspberry Pi so that the IP address does not need to be memorised, but instead a domain name such as 'flowchemdash.org' can be used.

No-IP (https://www.noip.com/) provides a platform to store the dynamic IP address of the Raspberry Pi so that a simple and custom hostname can instead be used, preventing a user from needed to memorise an IP address which may change.

9 Automated Optimisation Experimental Data

The Node-RED flow used for the closed-loop optimisation can be found in section 4.3.4. A photo for the physical set up of the Uniqsis reactor, Syrris Sampler and Dilutor (SAD) and HPLC is shown in the figure below.

Figure S6. Photo of automated flow reactor.

10 HPLC Chromatogram and Calibration

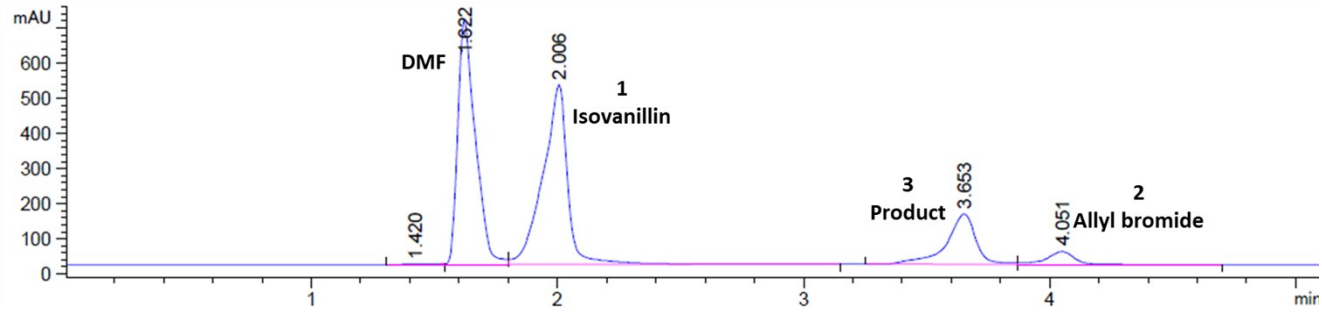


Figure S7. Typical HPLC chromatogram for the allylation reaction. Retention times (min): DMF = 1.62; **1** Isovanillin = 2.01; **2** Allyl bromide = 4.05; **3** Product = 6.66.

Calibration was performed by injected a known concentration of **1**, **2** or **3** into the Syrris Sampler and Dilutor.

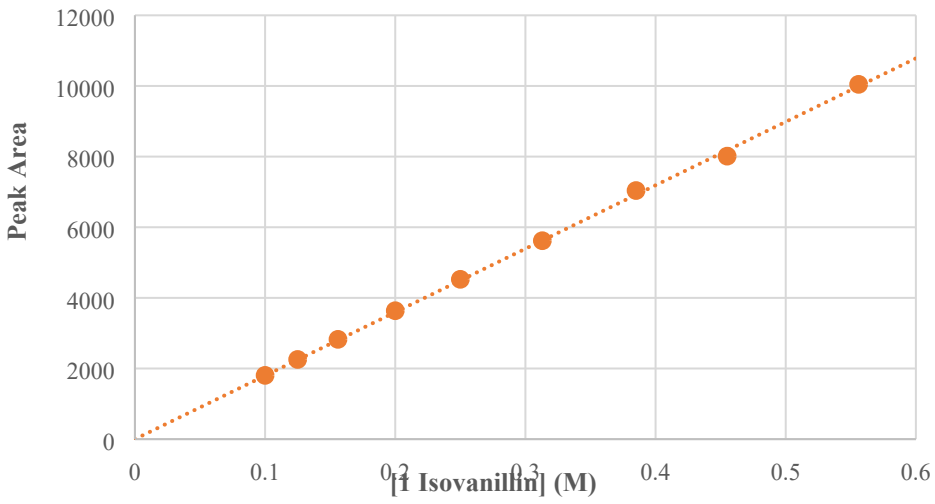


Figure S8. HPLC calibration plot for the reagent Isovanillin **1**. Dilution was performed using Syrris 'Sampler and Dilutor'.

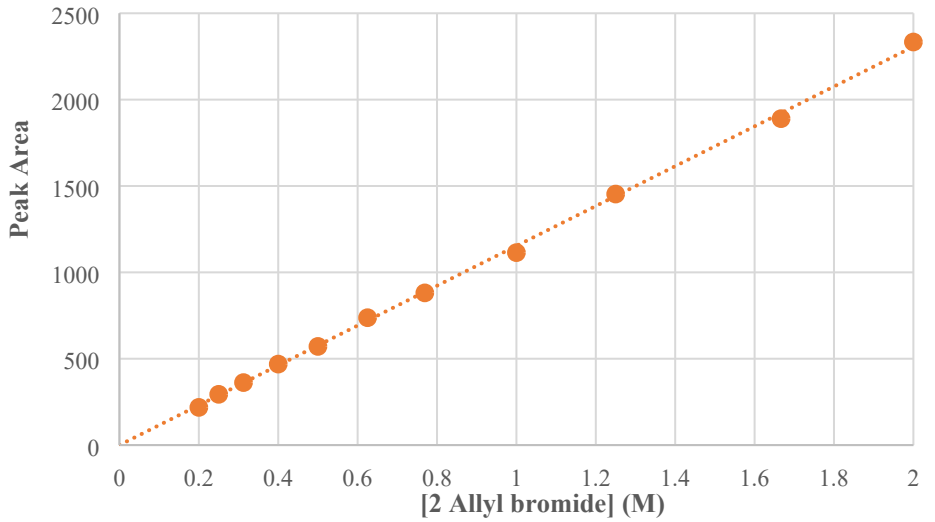


Figure S9. HPLC calibration plot for the reagent Allyl bromide **2**. Dilution was performed using Syrris 'Sampler and Dilutor'.

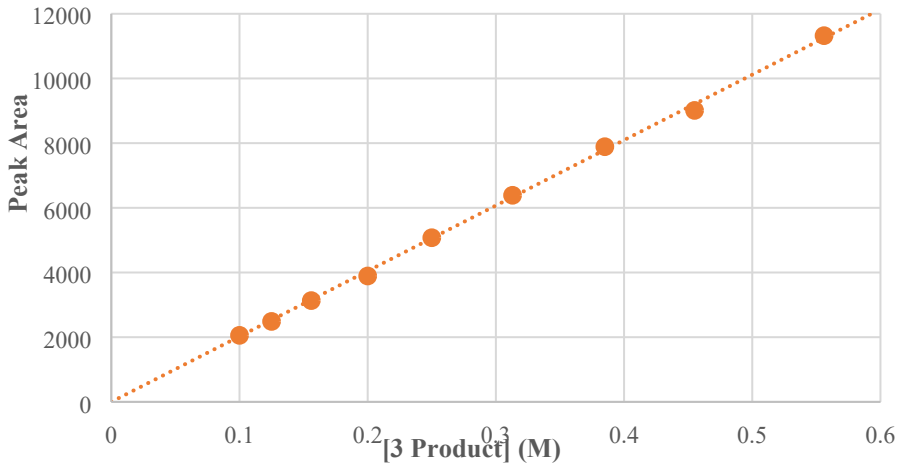


Figure S10. HPLC calibration plot for Product **3**. Dilution was performed using Syrris 'Sampler and Dilutor'.

11 Repeated use of single K_2CO_3 Column

To demonstrate the degradation of the K_2CO_3 column an autonomous set up was done which repeated the same conditions to note the yield of the product **3** - **Figure S9**. The conditions used for each run were:

- Column temperature: 80 °C;
- Residence time: 4 min;
- Equivalence of Allyl bromide **2**: 1.005

Because of the decreasing yield with each run that is performed, it was decided to replace the column after every three experimental runs to improve the realibility of the results.

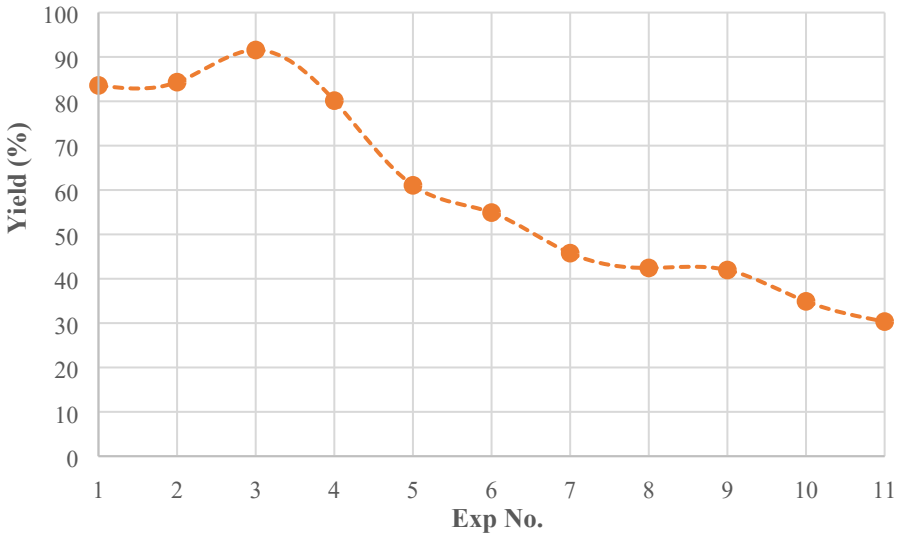


Figure S11. Plot for the yield of **3** using the same packed column of K_2CO_3. The experiment was repeated 11 times using consistent conditions. (Temp: 80 °C; Res. Time: 4 min; Eqv. Of **2**: 1.005)

Table S1. Experimental conditions, yields of reagents and

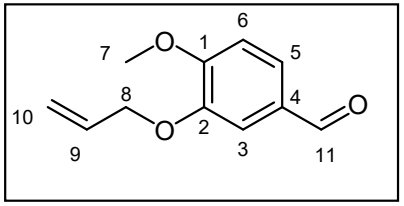
Exp No.	T (°C)	Res time (min)	Eqv	Pump A flow rate (ml/min)	Pump B flow rate (ml/min)	Iso conc (M)	Allyl conc (M)	Prod conc. (M)	Theoritcal Prod. Conc (M)	Prod yield (%)	STY ($g.L^{-1}.h^{-1}$)
1	86	28	1.078	0.14	0.11	0.000	0.020	0.257	0.277	92.7	105.8
2	73	16.28	1.193	0.23	0.2	0.000	0.039	0.256	0.265	96.8	181.7
3	73.5	16.28	1.309	0.22	0.21	0.000	0.059	0.246	0.253	97.1	174.2
4	69.8	16.28	0.988	0.25	0.18	0.043	0.010	0.258	0.288	89.8	183.1
5	67.2	4.43	1.01	0.91	0.67	0.052	0.025	0.240	0.285	84.1	624.1
6	61.3	4	1.005	1.01	0.74	0.133	0.168	0.171	0.286	60.0	494.4
7	71.3	4	1.005	1.01	0.74	0.067	0.023	0.255	0.286	89.4	736.1
8	78.5	4	1.005	1.01	0.74	0.065	0.023	0.263	0.286	92.1	758.6
9	75.5	4	1.005	1.01	0.74	0.075	0.040	0.242	0.286	84.7	697.4
10	90	4	1.005	1.01	0.74	0.054	0.008	0.228	0.286	80.0	658.8
11	80.2	4	1.005	1.01	0.74	0.091	0.093	0.179	0.286	62.6	515.5
12	73.1	4	1.005	1.01	0.74	0.138	0.186	0.134	0.286	46.9	386.2
13	90	4	1.005	1.01	0.74	0.063	0.019	0.253	0.286	88.7	730.3
14	90	4	1.005	1.01	0.74	0.061	0.030	0.224	0.286	78.2	644.5
15	90	4	1.005	1.01	0.74	0.188	0.000	0.159	0.286	55.5	457.3
16	84.1	4	1.005	1.01	0.74	0.060	0.000	0.240	0.286	84.0	692.0
17	40	4	1.387	0.87	0.88	0.198	0.248	0.059	0.246	23.9	169.7
18	83.4	4	1.005	1.01	0.74	0.077	0.035	0.223	0.286	78.0	642.8
19	83.4	4	1.005	1.01	0.74	0.083	0.036	0.217	0.286	75.9	624.9
20	83.4	4	1.005	1.01	0.74	0.078	0.050	0.183	0.286	64.1	528.4
21 (Blockage)	90	4	1.387	0.87	0.88	0.019	0.047	0.060	0.246	24.3	172.1
22	77.9	4	1.005	1.01	0.74	0.068	0.000	0.240	0.286	84.1	693.0
23	77.6	4	1.005	1.01	0.74	0.077	0.019	0.232	0.286	81.2	668.8
24	77.5	4	1.005	1.01	0.74	0.100	0.022	0.238	0.286	83.3	686.0
25	77.3	4	1.005	1.01	0.74	0.098	0.024	0.274	0.286	96.0	791.1
26	77.1	4	1.005	1.01	0.74	0.091	0.015	0.260	0.286	90.9	748.4
27	77	4	1.005	1.01	0.74	0.092	0.010	0.260	0.286	89.3	735.5
28	76.9	4	1.005	1.01	0.74	0.089	0.025	0.242	0.286	84.8	698.3

29	76.8	4	1.005	1.01	0.74	0.090	0.019	0.234	0.286	82.0	675.2
30	76.8	4	1.005	1.01	0.74	0.093	0.024	0.235	0.286	82.3	677.7
31	76.8	4	1.005	1.01	0.74	0.083	0.019	0.249	0.286	87.2	718.2
32	76.8	4	1.005	1.01	0.74	0.086	0.013	0.252	0.286	88.3	727.0
33	76.7	4	1.005	1.01	0.74	0.087	0.020	0.252	0.286	88.3	727.5
21 (Repeat)	90	4	1.387	0.87	0.88	0.055	0.106	0.226	0.246	91.9	651.7

12 Batch experimental data

Batch reaction – Product material was used for HPLC calibration of product.

3-(Allyloxy)-4-methoxybenzaldehyde



Potassium carbonate (12.13 g, 87.8 mmol, 2.5 equiv) was added to DMF (70 mL) and the solution was heated at 60 °C for 20 min. Isovanillin (5.34 g, 35.1 mmol, 1 equiv) was added to the solution along with allylbromide (7.6 mL, 87.8 mmol, 2.5 equiv). The reaction mixture was then stirred at 60 °C under an Argon atmosphere for 20 h. The reaction mixture was cooled to room temperature and the inorganic solids were filtered off through a celite pad. The DMF was removed and the mixture was purified by column chromatography (25 % EtOAc:Hexane). The product **3-(Allyloxy)-4-methoxybenzaldehyde,** was obtained as a light yellow oil in 89 %. ***R*f = 0.48** (silica gel, 3:1 Hexanes–EtOAc); **^{1}H** NMR (400 MHz, Chloroform-*d*): δ ppm 9.85 (d, *J* = 7.4 Hz, 1H, H_{11}), 7.63 – 7.33 (m, 2H, H_5 & H_3), 7.00 (dd, *J* = 8.4, 5.8 Hz, 1H, H_6), 6.22 – 5.92 (m, 1H, H_9), 5.59 – 5.21 (m, 2H, H_{10}), 4.79 – 4.53 (m, 2H, H_8), 4.11 – 3.83 (m, 3H, H_7); **^{13}C** NMR (101 MHz, Chloroform-*d*): δ ppm 190.8 (C=O), 154.8 (C_2), 148.5 (C_1), 132.4 ($OCH_2\underline{C}H=CH_2$), 129.9 (C_4), 126.8 (C_5), 118.6 ($OCH_2CH=\underline{C}H_2$), 110.7 (C_6), 69.7 ($O\underline{C}H_2CH=CH_2$), 56.1 ($O\underline{C}H_3$); **IR ν_{max}/cm^{-1}** 2935 (C-H str, vw), 2840 (C-H str, vw), 1680 (C=O str, s), 1649, 1583, 1507, 1461 (C-H bend, s), 1434 (C-H bend, s), 1396, 1339, 1259, 1233, 1190, 1162, 1130, 1013 (C-O str, s), 928, 864, 807, 780, 756, 734, 639, 576, 461, 421; **HRMS *m/z* (ES+)** 193.082 **[M + H]$^{+}$** ($C_{11}H_{12}O_3$ requires 192.08 g/mol).

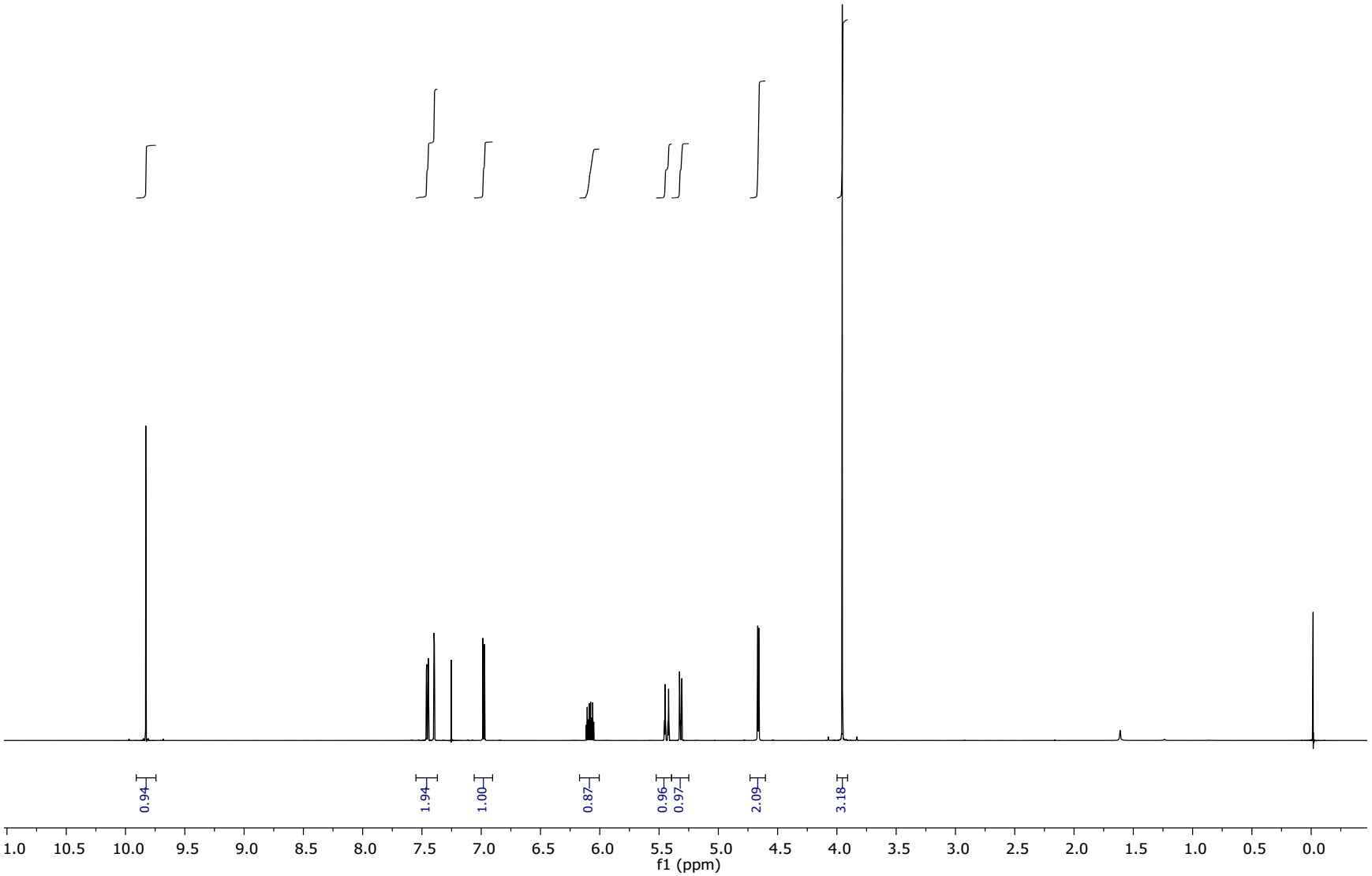


Figure S12. ^{1}H NMR Spectrum for **3-(Allyloxy)-4-methoxybenzaldehyde**

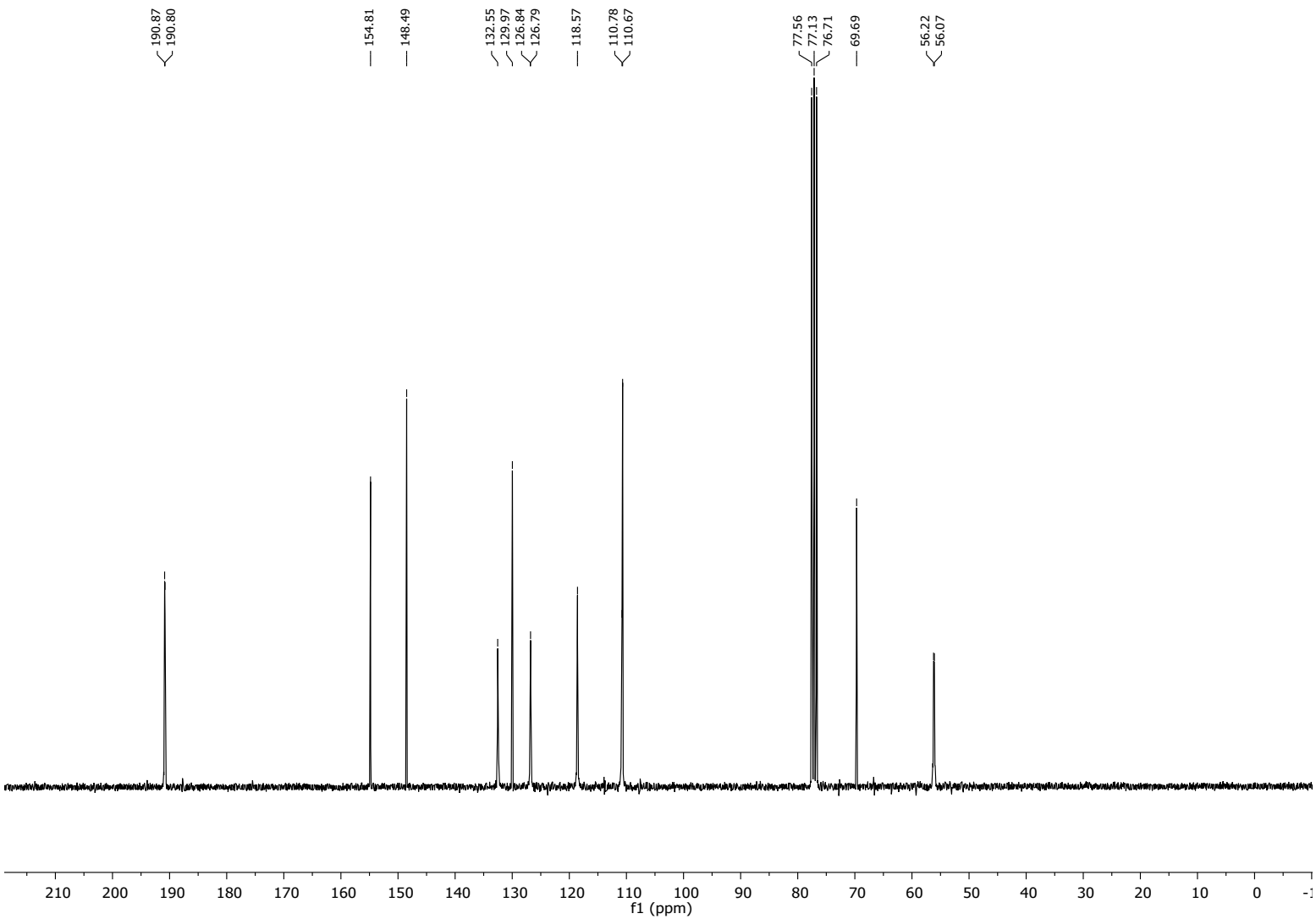


Figure S13. ^{13}C NMR Spectrum for **3-(Allyloxy)-4-methoxybenzaldehyde**

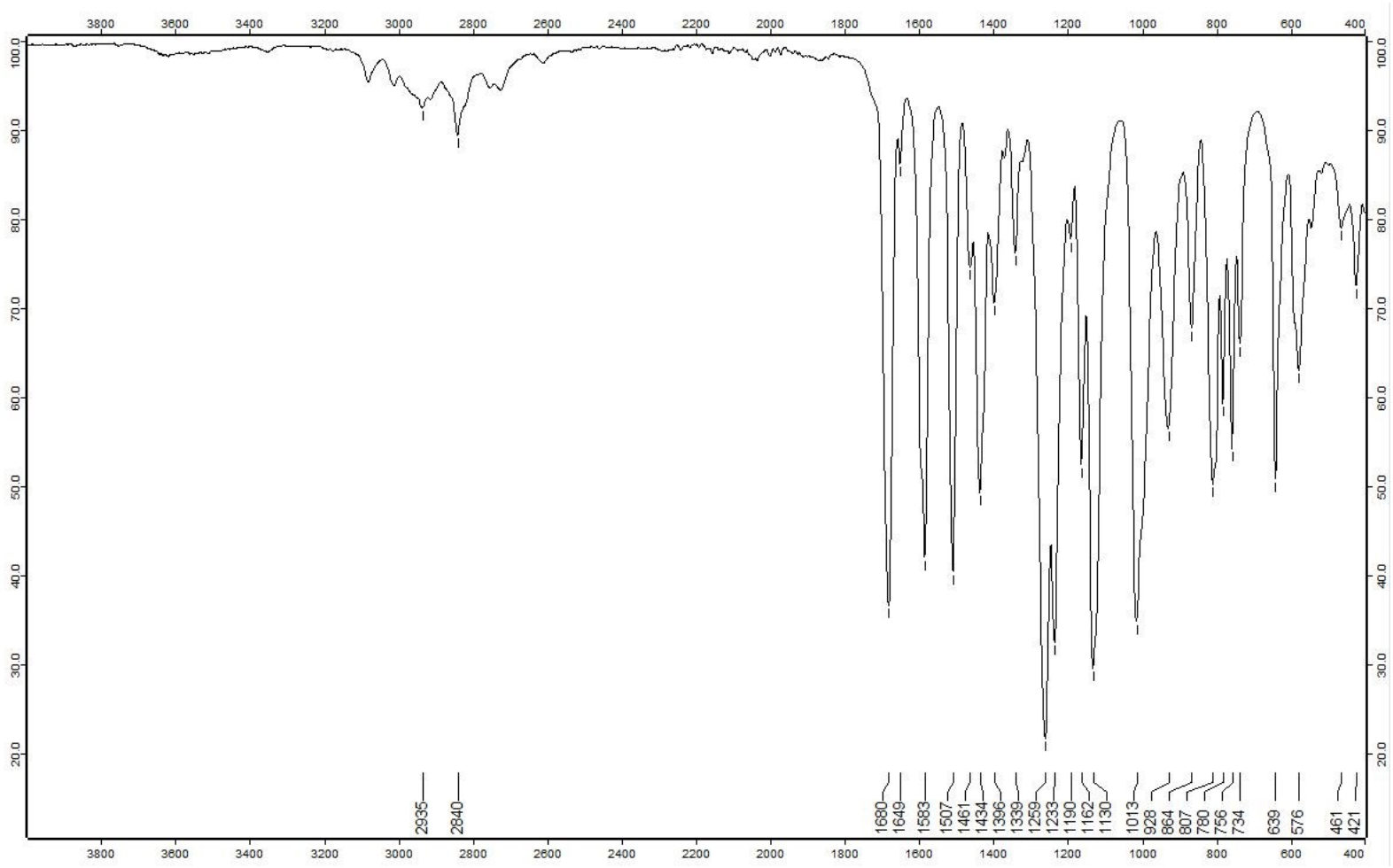


Figure S14. IR Spectrum for **3-(Allyloxy)-4-methoxybenzaldehyde**

13 References

1. Felton, K. C., Rittig, J. G. & Lapkin, A. A. Summit: Benchmarking Machine Learning Methods for Reaction Optimisation. *Chemistry–Methods* **1**, 116–122 (2021).