In-line separation of multicomponent reaction mixtures using a new semi-continuous supercritical fluid chromatography system.

Daniel E. Fitzpatrick,[†] Robbie J. Mutton,[†] Steven V. Ley[†]*

[†]Innovative Technology Centre, Department of Chemistry, University of Cambridge, Lensfield Road, Cambridge CB2 1EW, U.K.

Supporting Information

1.0 Description of process design, materials and equipment

Please refer to Figure 2 in the main text for a schematic layout of the SFC system.

Consumables and materials

CO_2 (liquid withdrawal) cylinders were purchased from BOC, UK. Methanol (HPLC grade) was purchased from Fisher Scientific, UK. All reagent starting materials were purchased from Fluorochem, Sigma Aldrich and Alfa Aesar, and were reagent grade. All solvents were filtered through a 40 μm filter prior to use.

Chromatography columns from Princeton Chromatography Inc (USA) with the following polymeric stationary phases were purchased from Crawford Scientific (UK); diethylamino (DEAP), 2-ethylpyridine (2-EP), propyl-pyridyl-urea (PPU). All columns (10 x 250 mm) had a particle size of 5 μm, DEAP and 2-EP phases have a pore size of 60 Å and PPU of 100 Å.

Supercritical fluid chromatography equipment

The system consisted of the following standard Jasco units unless otherwise stated (Figure S1). A PU-2088 CO_2 pump equipped with chilled pump head, powered by a Julabo FL1201 recirculation chiller; a PU-2088 co-solvent pump; a HE-2068 heater and HC-2068 heater controller; a CO-4065 column oven, containing the fluid mixing column and a 4-way flow splitting manifold; four PU-2086 semi-preparative HPLC pumps; four UV-2075 UV detectors equipped with preparative scale, high pressure cells; four BP-2080 automatic back pressure regulators; two CO-4062 column ovens, each containing two 6-position 7-port Rheodyne switching valves and used exclusively as heated value units.

The outputs from each valve was directed into a waste vessel or one of five cyclone collection vessels located in a fume cabinet, allowing for containment of CO_2 gas, which in turn was vented directly to the atmosphere outside the laboratory. The system was fitted with a pressure relief valve calibrated to 34.35 MPa to protect equipment in the event of a pressure surge. A manual

pressure regulator was fitted at the splitting manifold to reduce pressure fluctuations within the system between operations.

Crude reaction mixture was introduced to the SFC system via four 2-position 6-port Rheodyne valves, connected in series (refer to Figure 3a in the main text). Using a small auxiliary pump, a constant stream of mixture was directed through the injection system with any excess being returned to the reservoir. Loaded loops (100 µl) were injected sequentially to the column system at specified repeating intervals.

All equipment except for the SFC automatic back pressure regulators were controlled using an Internet-based control system (refer to reference 24 in the main text). The set point parameters for the BPRs were configured manually prior to process start up.

Figure S1. Photograph of the SFC equipment.

2.0 Selection of separation conditions

The selection of separation conditions for all examples followed the same general process. This process consisted of 3 separate steps, as summarised for the isolation of 1-bromoethylbenzene below.

Isolation of 1-bromoethylbenzene

Step 1 – Column selection

We maintained column sets of three functionalised silica materials (see above), with each set containing four columns so that the full use of the system would be possible no matter which

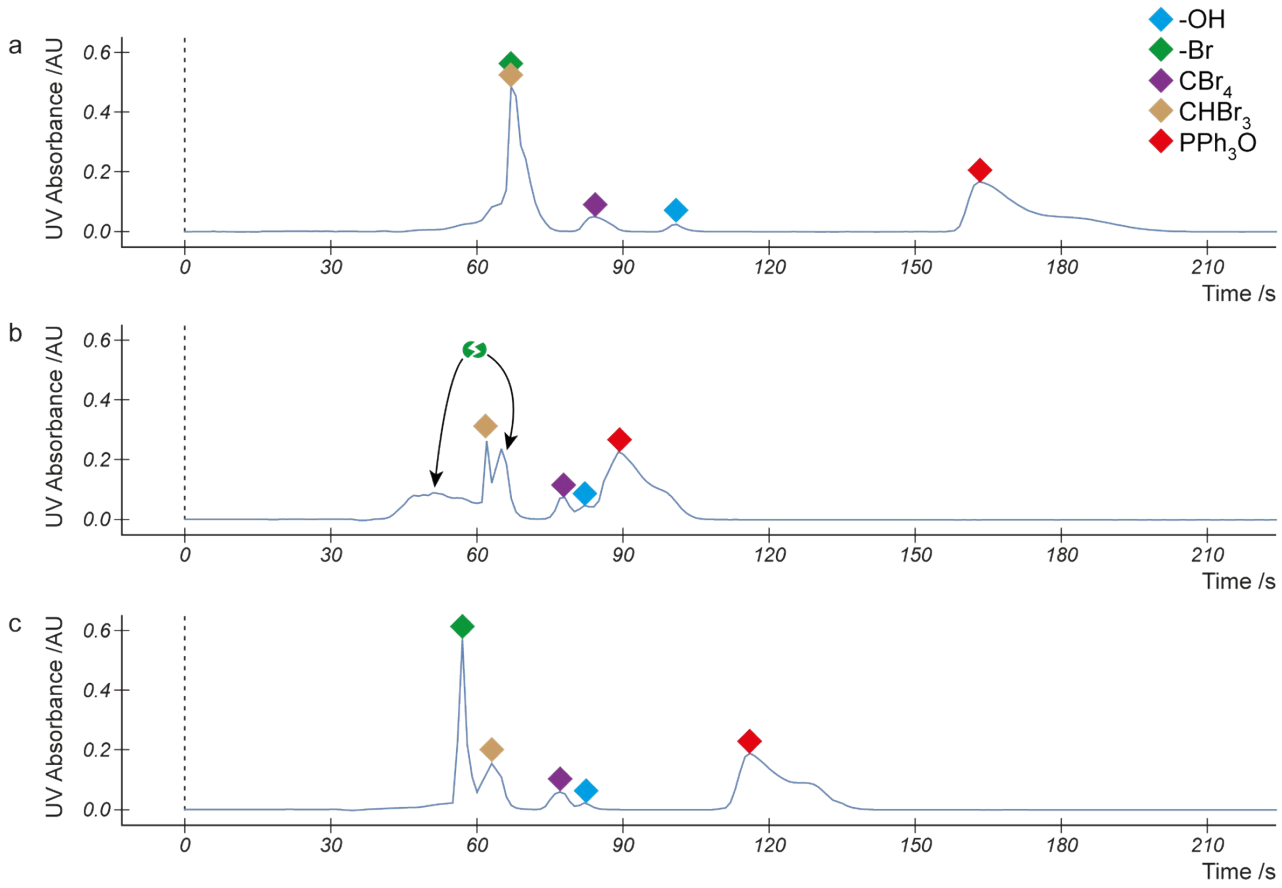


Figure S2. UV traces collected during a column screening process (220 nm, 65 °C, 10 % MeOH, 10 MPa). Breakdown of product was observed through the DEAP column. Columns: (a) PPU; (b) DEAP; (c) 2-EP.

material were selected. The pressure drop across each column was consistent at approximately 5.5 MPa for all separations described in this chapter.

During the screening process, one column from each set was connected to the first three column lines. The fourth line also held a column to maintain an overall pressure balance; however, no injection occurred into this line. The column screening procedure for all reactions was identical: after the mobile phase was pumped through the system for 10 minutes to allow for system equilibration, downstream UV-Vis detectors were then set to baseline and injections occurred simultaneously across the three columns lines. The output from detectors, set to monitor at 220 nm, was recorded and plotted by the control system so that the relative performance of columns could be compared easily.

For our first Appel reaction mixture, the mobile phase composition was 10 % methanol, 90 % CO_2. To ensure the system was operated under supercritical conditions, back pressure was set to 10 MPa and oven temperature set to 65 °C. UV traces for each of the column lines following injection are shown in Figure S2. Eluting bands were collected from each column to aid with the identification of peaks.

It was clear that separation over 2-EP columns gave the best results, with little overlap between peaks and good separation of expected reaction mixture components into individual bands. Accordingly we selected this column set for our separation.

Step 2 – Wavelength selection

The UV-Vis detectors downstream of columns were capable of monitoring wavelengths between 190 nm and 600 nm, giving some flexibility with how eluting peaks could be observed. This was particularly useful when eluting compounds had relatively low UV activity, such as bromoform and tetrabromomethane in our example mixture.

Having selected a suitable column, we proceeded to identify a detector wavelength that would give the most useful feedback. Before doing so, the system was depressurised and the columns for lines one, two and four were replaced with those containing 2-EP. Having re-equilibrated the system (65 °C, 10 MPa back pressure and 10 % MeOH in CO_2), detectors were configured to monitor for different wavelengths and reaction mixture was injected into each of the four columns simultaneously.

The best results were obtained when monitoring at 210 nm (Figure S3a). This wavelength also enabled us to see a new leading peak that was not visible at 220 nm during the column screen. An additional benefit at this wavelength could be realised if the recycle of unreacted starting material were a priority; under these conditions, the peak corresponding to the alcohol starting material a higher absorbance and thus would be more straightforward to collect via UV

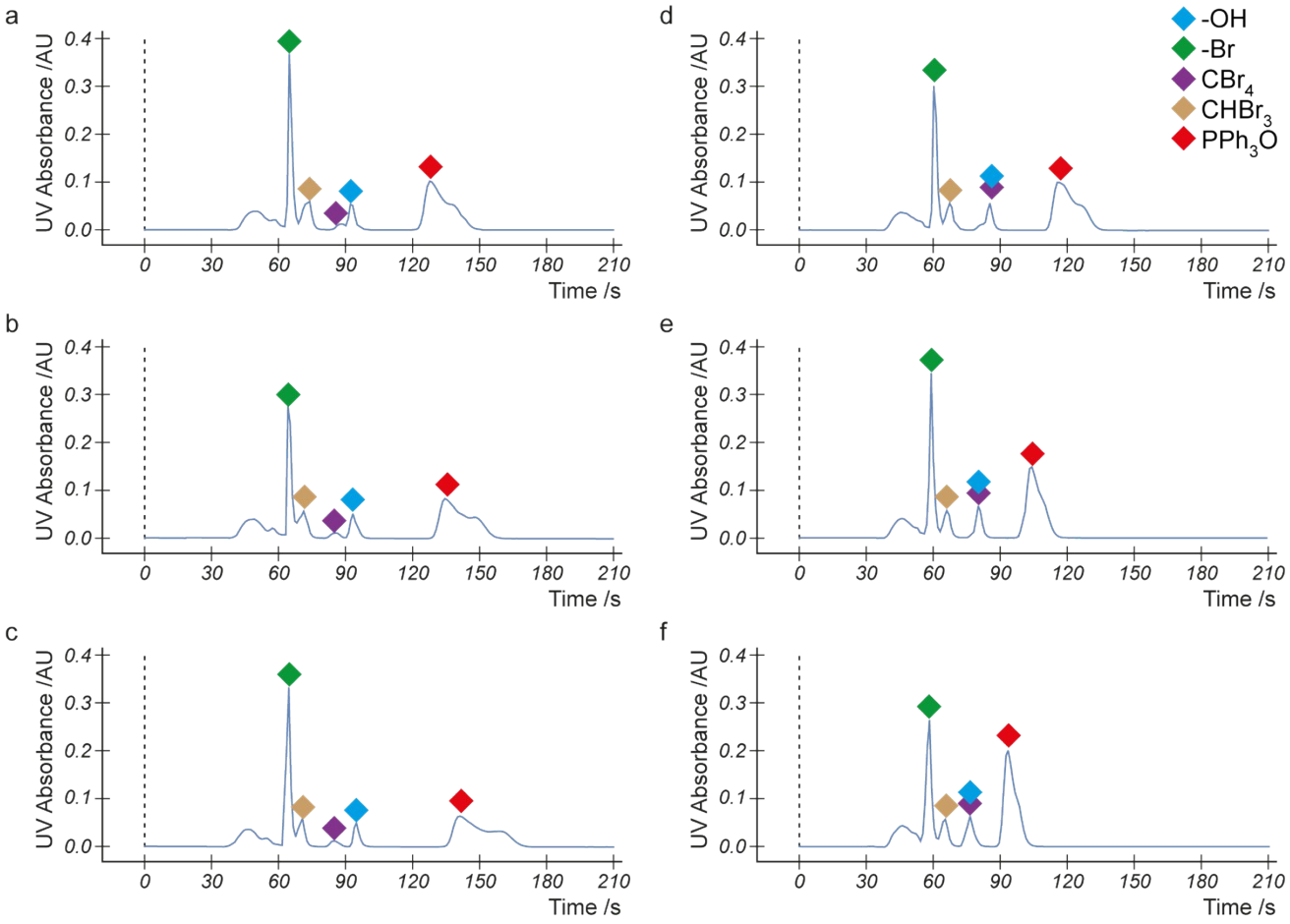


Figure S3. UV traces obtained during wavelength screening and method development procedures. (a) Wavelength screen at 210 nm (10 % MeOH, 10 MPa). (b) 9 % MeOH, 11 MPa. (c) 9 % MeOH, 10 MPa. (d) 11 % MeOH, 10 MPa. (e) 12.5 % MeOH, 10 MPa. (f) 14 % MeOH, 10 MPa. The column temperature was fixed at 65 °C for all separations.

detection. In this work we did not exploit this, as we had configured the reaction such that the alcohol was not the limiting reagent.

Step 3 – Separation optimisation

Having selected a suitable column and wavelength, we turned our attention to the optimisation of separation conditions. Our objective was simple: the system must elute the target compound quickly in as high a purity as possible. SFC-based processes have a number of parameters that can be adjusted to achieve this, including the composition of the mobile phase, oven temperature and system pressure differentials.

Initial conditions used during the column selection and wavelength screening processes had resulted in a reasonable separation and thus provided a solid starting point for further optimisation.

During the first optimisation run (Figure S3b), we attempted to exploit the relationship between pressure and solvation power for supercritical carbon dioxide systems by decreasing concentration of MeOH while increasing back pressure to 11.0 MPa. While these conditions maintained good separation between the first peak and our target material, there was an increased overlap of the target product with bromoform. The total separation time remained approximately the same at 2.6 minutes (*cf.* 2.5 minutes during UV screening).

In subsequent trials (Figure S3c-f) we focussed on adjusting mobile phase composition. Optimal conditions were obtained with 12.5 % MeOH, an oven temperature of 65 °C and a back pressure of 10 MPa (Figure S3e), giving a total separation time of 2.0 minutes. While there was very slight peak overlap at these conditions, they achieved the best balance between purity and overall separation speed. It is worth noting that these conditions led to a short time difference between when the first peak started to elute and the final peak finished eluting, presenting opportunities to increase throughput by stacking injections into columns.

Isolation of AZ82 intermediate

We followed the same steps as the example above to develop a suitable separation procedure for the isolation of methyl (2S)-2-((*tert*-butoxycarbonyl)amino)-3-iodopropanoate.

Following an initial column screening process (Figure S4a-c), DEAP columns were identified as potentially the most suitable. However, having modified conditions to separate the target peak from that for triphenylphosphine oxide, we observed what appeared to be product breakdown in the UV trace (Figure S4d). This was confirmed by ^{1}H NMR analysis, which indicated that 31 % of the collected product had undergone elimination to form the alkene compound shown in Figure S4e, likely driven by the basicity of the column.

We repeated a column screening process using just PPU and 2-EP columns (Figure S5). Given the results from the previous screening, the proportion of MeOH in the mobile phase was lowered to 10 % to ensure that a difference between the columns could be discerned.

It was found that the 2-EP stationary phase resulted in the best separation. Having switched all lines to hold 2-EP columns, we proceeded to screen UV wavelengths. Owing to the lower UV activity of the alcohol starting material and product relative to triphenylphosphine and its oxide,

a wider screen of wavelengths was performed (190 nm to 380 nm). It was found that monitoring at 200 nm provided the best shape and magnitude of the product peak and immediate neighbours.

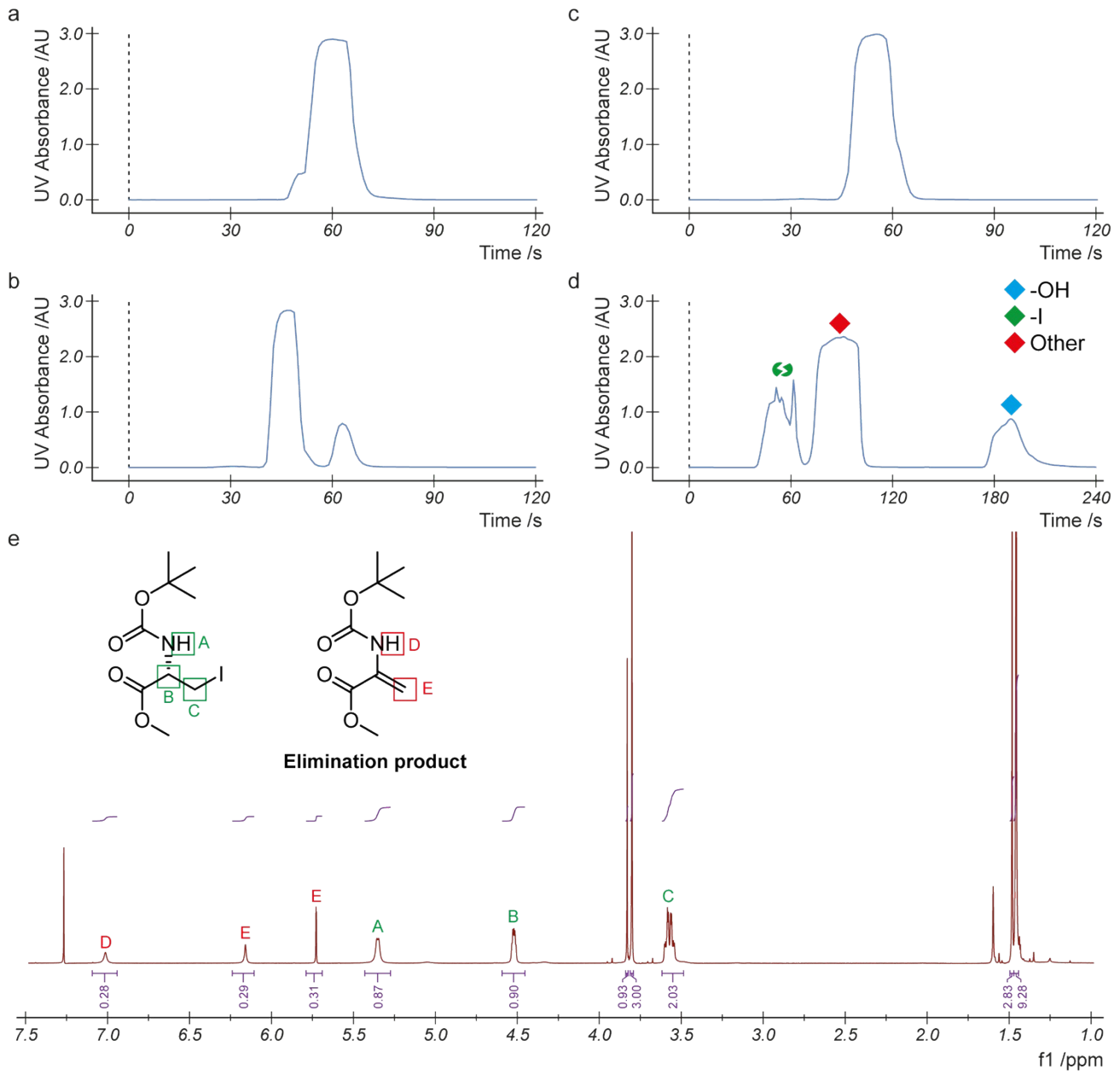


Figure S4. UV traces for the initial column screening process (85 °C, 30 % MeOH, 12 MPa) for the reaction producing the AZ82 intermediate: (a) PPU columns; (b) DEAP; (c) 2-EP. Other plots: (d) UV trace showing product breakdown at 65 °C, 10 % MeOH and 12 MPa back pressure; (e) ^{1}H NMR of first fraction when using DEAP columns, showing 31 % breakdown of product into the alkene elimination product.

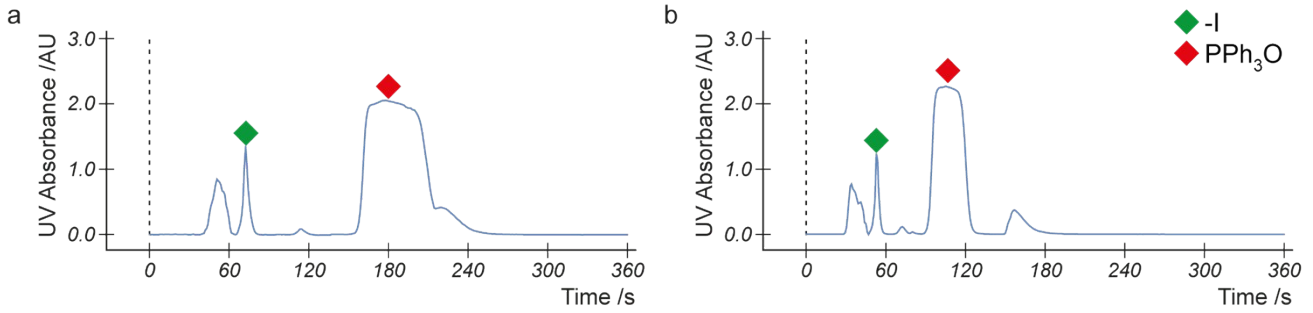


Figure S5. UV traces (200 nm) obtained during the second column screening process (65 °C, 10 % MeOH, 10 MPa): (a) PPU; (b) 2-EP.

The operating conditions used during the column and wavelength screening processes yielded excellent results, with no peak overlap between our target compound and its neighbours. It also gave good separation of other mixture components which, while not applied for this example, could aid with the isolation of unreacted starting materials if required. Accordingly no further optimisation of separating conditions was required.

Isolation of isonicotinamide

As with the two separations described above, a screening process was conducted to identify optimal conditions within the SFC system to separate our two component product mixture. It was found that best results were obtained using DEAP columns at 65 °C, with a MeOH concentration of 22.5 % and a back pressure of 12.0 MPa.

3.0 Control sequence for separation of telescoped intermediate

The telescoped synthesis of isoniazid involved the control of 24 independent machines, including pumps, reactor systems, valves and an IR spectrometer (Mettler Toledo Flow IR), and necessitated equipment querying on a per-second basis owing to the rapid separation times in the SFC. As described in the main text, an Internet-based control system was used to facilitate such a challenging example. Please refer to Figure 7 in the main text for a schematic of equipment.

The control strategy was broken into three segments corresponding to process start up, operation at steady state and finally equipment shutdown. A flowchart visualising each step is shown in below.

The synthesis was initiated by loading equipment and automation code into two separate experiments on the control server. One experiment dealt with operation of reaction equipment associated with steps one and two (*Reaction*, as labelled in the flow chart), while the second controlled only the SFC system (*SFC*). This structure enabled us to have independent control over each segment of the telescoped process, simplifying actions should any operational delays arise (for example if replacement of the carbon dioxide feed cylinder were required). Two reservoirs were added before and after the SFC system to act as buffers, helping to minimise process disruption in the event of a pause.

Process start up

Our telescoped process began with the *Reaction* script. At the commencement of the experiment, the flow rate of the pump (P1) and temperature of the reactor column (R1) for the first reaction step were both set and a listener function created to wait for the reactor temperature to reach its set point. P1 was started at this time to ensure that there was sufficient back pressure in the system during initial reactor heating, thus avoiding any boiling of MeOH. While the reactor warmed, solution owing through the system was directed to waste by a valve (V2) downstream of the reactor.

As soon as the desired temperature had been reached, the feed to the pump was switched to the reagent solution thus beginning the first transformation. Simultaneously a listener was created to monitor data captured from IR for the presence of isonicotinamide in the solvent stream leaving the column. This listener was triggered when the infrared absorption at 1686 cm^{-1} rose above a

0.10 AU threshold for a period longer than 12 seconds, minimising the impact that detector noise had on system operation.

When triggered, this listener directed reaction mixture to the SFC injection reservoir and initiated the *SFC* experiment. The *SFC* scripts then began the process of equilibrating the SFC columns and stabilising system back pressure. Following a seven minute delay, the UV-Vis detectors on each column line were set to baseline and reaction mixture held in the first reservoir was recirculated through injection loops. The injection cycle began 30 seconds later.

Seven minutes following the IR trigger, the *Reaction* script adjusted the temperature set point of the seven reactor coils to 130 °C for the second reaction step and pressurised the system by pumping solvent through pumps associated with the second reaction step (P3 and P4). As soon as the temperatures of the seven reactor slots had stabilised, the control system switched supply valves for P3 and P4 (V3 and V4) to reagent lines, initiating the second reaction. After 22 minutes - the residence time for the second step - the position of the second step product collection valve (V5) was adjusted to collect the product stream.

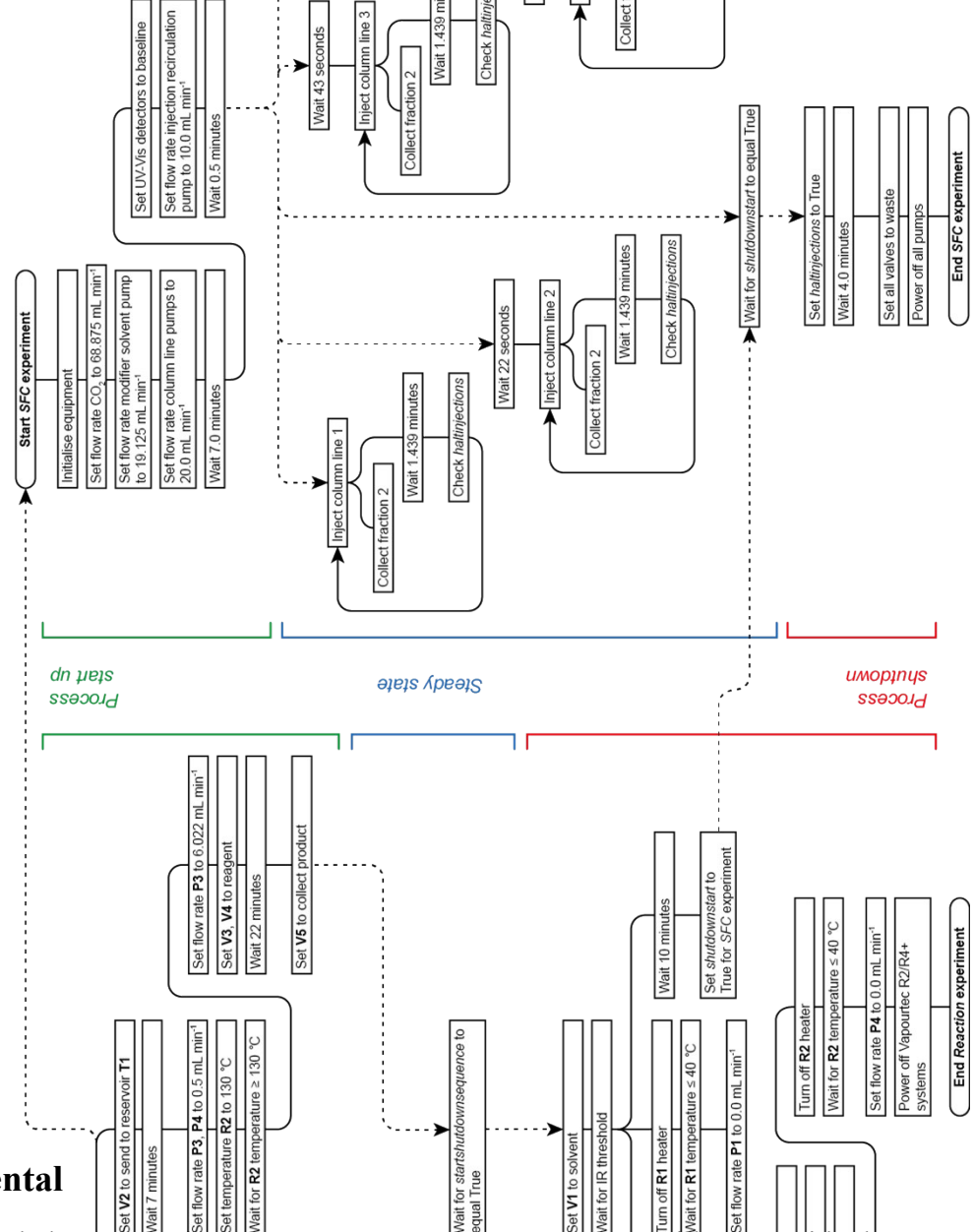
At this time a new listener was created to monitor the value of the *startshutdownsequence* experiment variable; the system had now reached steady state.

Equipment shutdown

When the shutdown sequence was initiated, the *Reaction* script set the position of V1 to solvent and waited for the signal from IR to fall below a 0.10 AU threshold for 12 consecutive seconds. At this time the stream leaving the first reaction step was directed to waste and column heating was ceased.

As soon as reactor temperature had fallen below 40 °C, P1 was halted. After a period of 10 minutes following the infrared trigger, a signal was sent to the *SFC* experiment to initiate system shutdown. At this time, the SFC script halted the injection cycle and waited for an additional 4 minutes before shutting down remaining equipment.

15 minutes following the infrared trigger, the control system started the shutdown of the second reaction step by switching V3 and V4 to pump solvent through the reactor coils. When 21 minutes had passed, heaters of the seven reactor slots were turned off and the product stream was directed to waste. As soon as all reactor readings had fallen below 40 °C, P4 was halted, both reactor systems (Vapourtec R2+/R4 units) were powered down and the experiment was terminated.



4.0 General Experimental

^{1}H-NMR spectra were recorded on a Bruker Avance DPX-600 spectrometer with the residual solvent peak as the internal reference ($CDCl_3$ = 7.26 ppm, d_6-DMSO = 2.50 ppm). ^{1}H resonances are reported to the nearest 0.01 ppm. ^{13}C-NMR spectra were recorded on the same spectrometers with the central resonance of the solvent peak as the internal reference ($CDCl_3$ = 77.16 ppm, d_6-DMSO = 39.52 ppm). All ^{13}C resonances are reported to the nearest 0.1 ppm. The multiplicity of 1H signals are indicated as: s = singlet, d = doublet, dd = doublet of doublet, ddd = doublet of doublet of doublet, t = triplet, q = quadruplet, sext = sextet, m = multiplet, br. = broad, or combinations of thereof. Coupling constants (J) are quoted in Hz and reported to the nearest 0.1 Hz. Where appropriate, averages of the signals from peaks displaying multiplicity were used to calculate the value of the coupling constant. Unless stated otherwise, reagents were obtained from commercial sources and used without purification. The removal of solvent under reduced pressure was carried out on a standard rotary evaporator.

Infrared spectra were recorded neat or as a thin film on a PerkinElmer Spectrum One FTIR spectrometer using Universal ATR sampling accessories. Letters in parentheses refer to the relative absorbency of the peak: w = weak, less than 30 % of the most intense peak; m = medium, ca. 31 - 69 % of the most intense peak; s = strong, greater than 70 % of the most intense peak.

Flash chromatography over silica gel was performed on a Biotage SP1 purification system with prepacked SiliCycle silica cartridges (4, 12, 25 or 40 g). Melting points were performed on a Stanford Research Systems MPA100 (OptiMelt) automated melting point system using a gradient of 1 °C min^{-1}, and are uncorrected.

High resolution mass spectrometry (HRMS) within ±5 ppm was carried out on a Waters Micromass LCT Premier spectrometer using time of flight with positive ESI, or by Mr Paul Skelton on a Bruker BioApex 47e FTICR spectrometer using positive ESI or EI at 70 eV, to within ±5 ppm of the theoretically calculated value.

Optical rotation measurements were recorded on a Perkin-ElmerModel 343 digital polarimeter using a Na/halogen lamp (589 nm) as the light source over a pathlength of 100 mm. $[\alpha]_D^{28.0}$ values are reported in (deg mL)(g $dm)^{-1}$ at specified concentrations (c) in g (100 $mL)^{-1}$ at temperature (T).

Elemental composition microanalysis was performed by the Microanalysis Laboratories at the Department of Chemistry, University of Cambridge and results are reported to two decimal places.

*Synthesis of **1-Bromoethylbenzene.***

Triphenylphosphine (2.098 g, 8.0 mmol, 0.8 eq.) was dissolved in MeCN (50 mL) and the solution cooled to 0 °C. Carbon tetrabromide (3.316 g, 10.0 mmol, 1.0 eq.) was added slowly and the mixture left to stir at room temperature for 10 minutes before being cooled to 0 °C. 1-Phenylethanol (1.21 mL, 10.0 mmol) was added dropwise to the solution, and the resulting mixture left to stir at 0 °C for 60 minutes. The reaction mixture was allowed to warm to room temperature, and stirred for an additional 1.5 hours.

A sample of 2 mL crude reaction mixture was purified using the supercritical fluid chromatography (SFC) unit, to recover 81 % of the title compound (92 % purity, by HPLC, NMR) in the crude mixture as a yellow/brown liquid.

IR (neat) ν/cm^{-1}: 2468 (bw), 2158 (m), 2033 (m), 1977 (m), 1494 (w), 1454 (m), 1376 (w), 1211 (m), 1178 (m), 1044 (m), 1025 (m), 963 (m), 911 (w), 760 (s), 693 (s);

^{1}H NMR (600 MHz, $CDCl_3$) δ/ppm = 7.45 (d, J = 7.2 Hz, 2H), 7.36 (t, J = 7.5 Hz, 2H), 7.30 (t, J = 7.2 Hz, 1H), 5.23 (q, J = 7.0 Hz, 1H), 2.06 (d, J = 7.0 Hz, 3H);

^{13}C NMR (150 MHz, $CDCl_3$) δ /ppm = 143.19, 128.63, 128.30, 126.76, 49.53, 26.79;

HRMS m/z calc. for C_8H_8Br $[M - H]^-$ 182.9809, found 182.9806, Δ = -1.6 ppm;

Microanalysis calc. (found) for C_8H_8Br C 51.92% (51.94%), H 4.90% (4.86%), Br 43.18% (42.62%).

Recorded NMR data were consistent with those reported previously (R. M. Denton, J. An, B. Adeniran, A. J. Blake, W. Lewis and A. M. Poulton, *J. Org. Chem.*, 2011, **76**, 6749-6767).

Synthesis of ***Methyl (2S)-2-((tert-butoxycarbonyl)amino)-3-iodopropanoate.***

Triphenylphosphine (3.147 g, 12.0 mmol, 1.2 eq.) and imidazole (0.817 g, 12.0 mmol, 1.2 eq.) were dissolved in DCM (50 mL) and the solution cooled to 0 °C. Iodine (3.046 g, 12.0 mmol, 1.2 eq.) was added slowly and the stirred mixture stirred at room temperature for 10 minutes. During this time a yellow suspension was observed to form. The mixture was cooled to 0 °C before a solution of methyl (*tert*-butoxycarbonyl)-D-serinate (2.190 g, 10.0 mmol) in DCM (2.0 m, 5.0 mL) was added dropwise. The solution was left to stir at 0 °C for 60 minutes, then room temperature for 1.5 hours.

A sample of 1.0 mL crude reaction mixture was diluted with DCM (1.0 mL) and then purified using the SFC unit, to recover 89 % of the title compound (>99 % purity by HPLC, NMR) from the crude mixture as a light yellow oil that solidified in the refrigerator.

IR (neat) ν/cm^{-1}: 3349 (m), 2982 (w), 1732 (s), 1690 (s), 1522 (s), 1438 (m), 1394 (m), 1368 (m), 1314 (s), 1273 (s), 1249 (m), 1222 (s), 1206 (m), 1156 (s), 1141 (s), 1059 (s), 1010 (m), 980 (m), 863 (m), 791 (m);

^{1}H NMR (600 MHz, $CDCl_3$) δ/ppm = 5.35 (d, J = 6.2 Hz, 1H, NH), 4.38-4.57 (m, 1H), 3.79 (s, 3H), 3.53-3.59 (m, 2H), 1.45 (s, 9H);

^{13}C NMR (150 MHz, $CDCl_3$) δ /ppm = 170.02, 154.80, 80.47, 53.65, 52.98, 28.25, 7.83;

HRMS m/z calc. for $C_9H_{17}INO_4$ $[M + H]^+$ 330.0202, found 330.0192, Δ = -3.0 ppm;

$[\alpha]_D^{28.0}$ = -41.5 (c 1.0 in $CHCl_3$);

Microanalysis calc. (found) for $C_9H_{17}INO_4$ 32.84% (32.84%), H 4.90% (4.82%), N 4.26% (4.14%).

Recorded NMR data were consistent with those reported previously (D. Crich and M. Y. Rahaman, *J. Org. Chem.*, 2009, **74**, 6792-6796).

Synthesis of ***Isonicotinamide.***

A solution of pyridine-4-carbonitrile in IPA/H2O (0.5 M, 1:1 *v/v*) was pumped at 0.278 mL min^{-1} through an Omnifit column (100 mm length, ⌀ 10 mm) heated to 117 °C. The column was packed with 2.5 g manganese dioxide, and small plugs of celite were placed at both ends of the column. Pressure was provided by a 75 psi back pressure regulator.

A sample of 1.0 mL crude reaction mixture was purified using the SFC unit using conditions described above, to recover 97 % of the title compound (>99 % purity by HPLC, NMR) as a white solid.

Mp: 129 - 131 °C (DCM) [Lit. 128 - 130 °C];

IR (neat) ν/cm^{-1}: 3312 (m), 3058 (bm), 2520 (bw), 2159 (m), 1978 (m), 1677 (s), 1619 (s), 1599 (s), 1553 (s), 1490 (m), 1395 (s), 1223 (s), 1149 (m), 1129 (m), 1061 (m), 1001 (s), 845 (s), 817 (s), 758 (s), 672 (s);

^{1}H NMR (600 MHz, d_6-DMSO) δ/ppm = 8.71 (dd, J = 4.4, 1.6 Hz, 1H), 8.24 (s, 1H, NH_2), 7.76 (dd, J = 4.4, 1.6 Hz, 1H), 7.72 (s, 1H, NH_2);

^{13}C NMR (150 MHz, d_6-DMSO) δ /ppm = 166.29, 150.19, 141.26, 121.38;

HRMS m/z calc. for $C_6H_7N_2O$ $[M + H]^+$ 123.0553, found 123.0551, Δ = -1.62 ppm;

Microanalysis calc. (found) for $C_6H_7N_2O$ C 59.01% (58.76%), N 22.94% (22.88%), H 4.95% (4.90%).

All recorded data were consistent with those reported previously:

C. Battilocchio, J. M. Hawkins and S. V. Ley, *Org. Lett.*, 2014, **16**, 1060-1063.

M. A. M. Abu-Youssef, R. Dey, Y. Gohar, A. A. Massoud, L. Öhrström and V. Langer, *Inorg. Chem.*, 2007, **46**, 5893-5903.

C. Crisóstomo, M. G. Crestani and J. J. García, *Inorganica Chim. Acta*, 2010, **363**, 1092-1096.

Synthesis of ***Pyridine-4-carbohydrazide.***

A solution of isonicotinamide in MeOH (0.011 M), obtained from the SFC unit, was pumped at 6.022 mL min^{-1} to a tee junction where it was mixed with a solution of hydrazine monohydrate in MeOH (1.32 M) pumped at 0.500 mL min^{-1}. The resulting mixture was heated to 130 °C in seven 20 mL reactor coils. System pressure was maintained using a 150 psi back pressure regulator.

Solvent was removed from a 100 mL sample of crude reaction mixture under reduced pressure and residues were purified over silica (DCM/MeOH gradient elution, 0 % for 1 CV, 0 - 10 % over 15 CV, 10 % for 20 CV) to afford the title compound (0.079 g, 0.575 mmol, 27 %) as a white solid.

Mp: 171 - 173 °C (DCM) [Lit. 171 - 173 °C];

IR (neat) ν/cm^{-1}: 3104 (bm), 1549 (bs), 1459 (bs), 1411 (s), 1305 (bs), 1220 (m), 1138 (bm), 994 (s), 952 (m), 887 (m), 844 (s), 815 (m), 740 (m), 670 (s);

^{1}H NMR (600 MHz, d_6-DMSO) δ/ppm = 10.09 (s, 1H, NH), 8.70 (dd, J = 4.4, 1.6 Hz, 2H), 7.72 (dd, J = 4.4, 1.6 Hz, 2H), 4.62 (s, 2H, NH_2);

^{13}C NMR (150 MHz, d_6-DMSO) δ /ppm = 163.85, 150.18, 140.23, 120.97;

HRMS m/z calc. for $C_6H_8N_3O$ $[M + H]^+$ 138.0662, found 138.0659, Δ = -2.02 ppm;

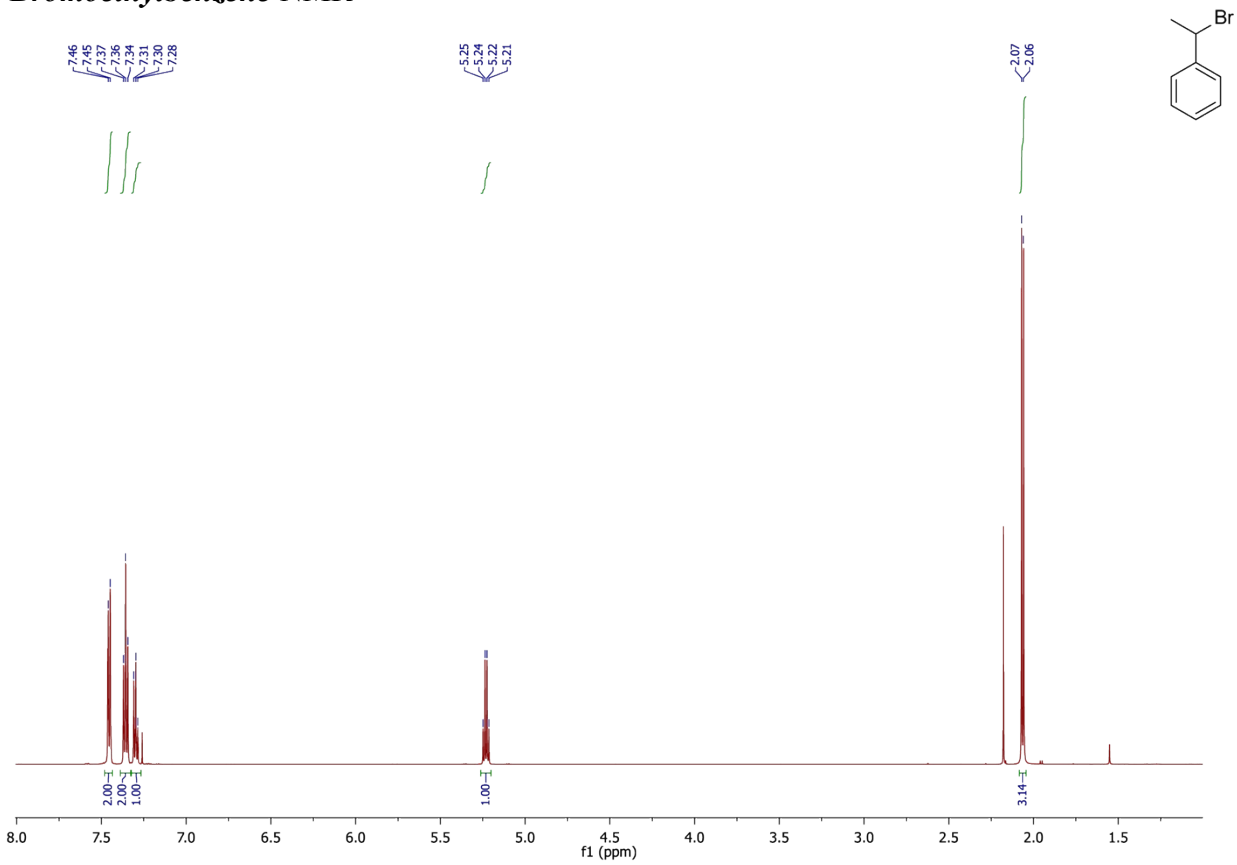
Microanalysis calc. (found) for $C_6H_8N_3O$ C 52.55% (52.25%), N 30.64% (30.43%), H 5.15% (5.15%).

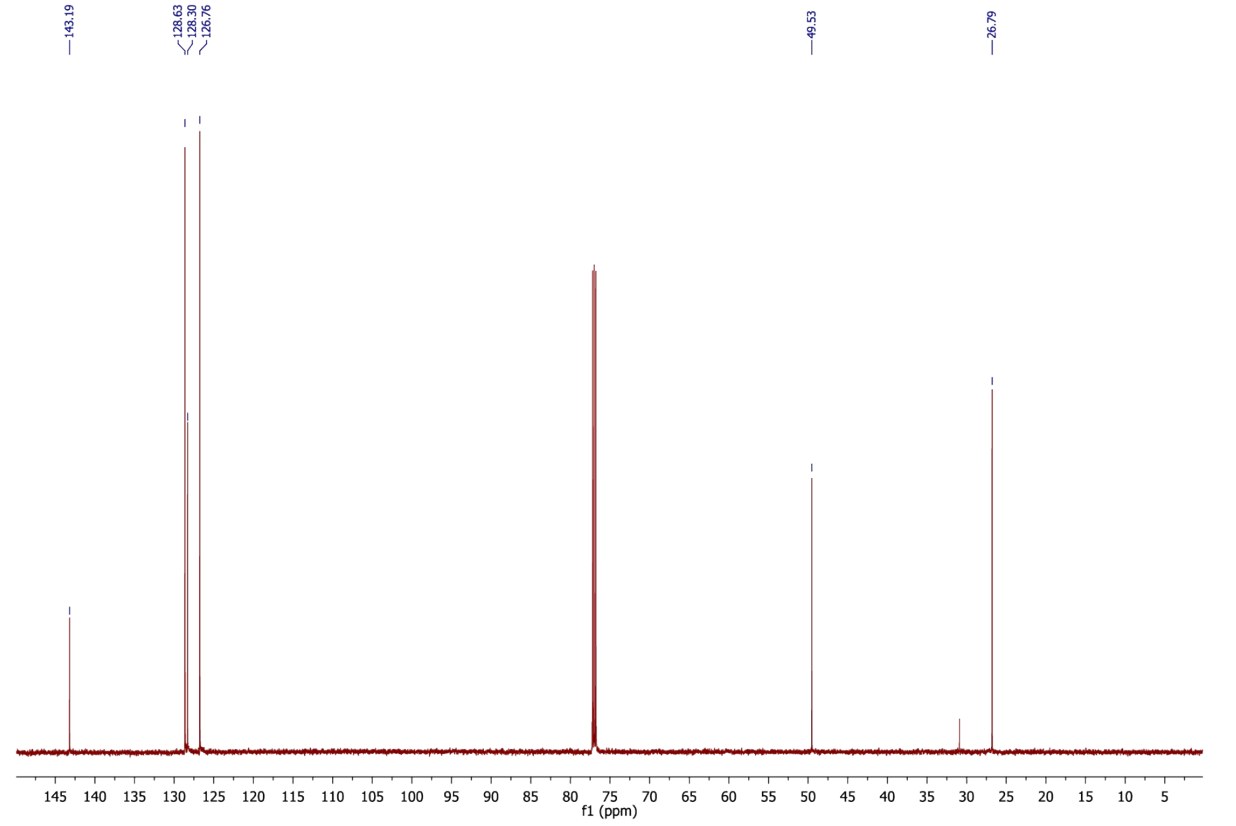
All recorded data were consistent with those reported previously:

D. Kaushik, S. A. Khan and G. Chawla, *Eur. J. Med. Chem.*, 2010, **45**, 3960-3969.

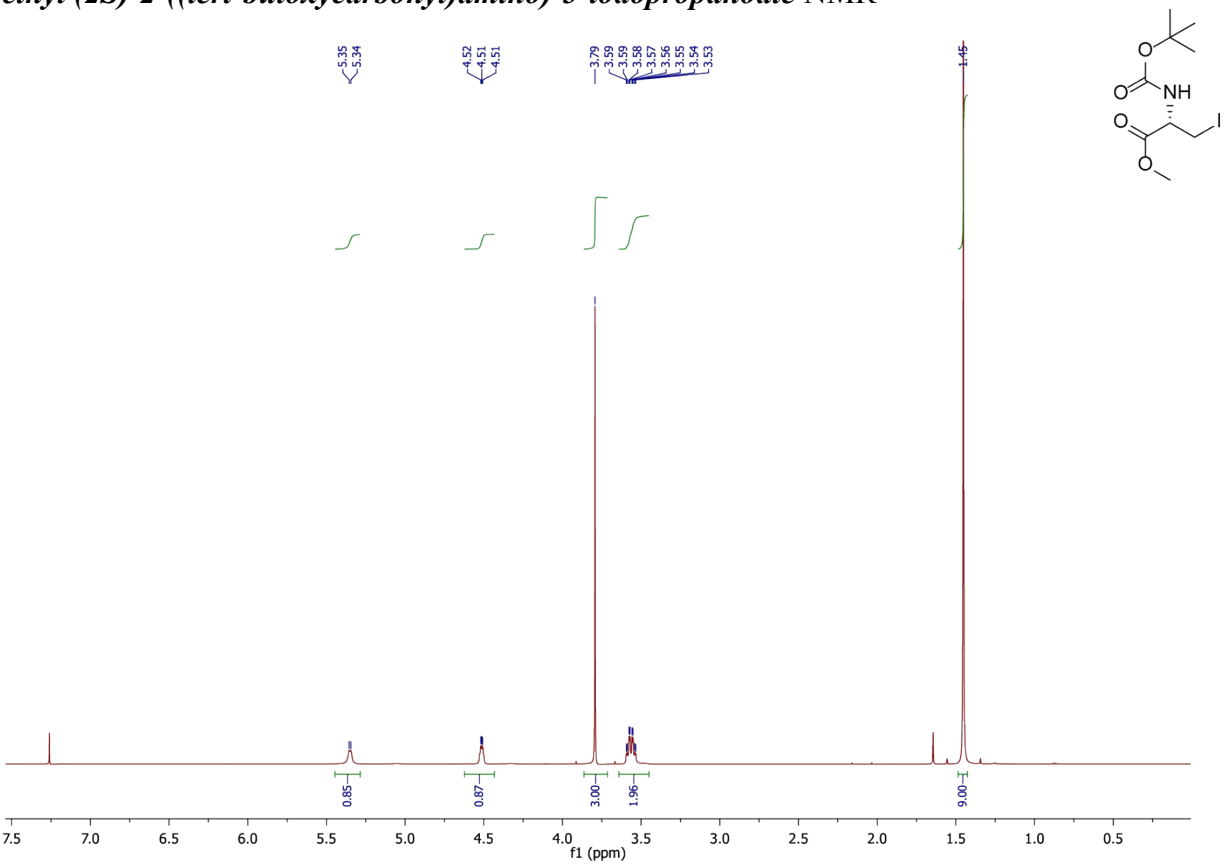
Data were compared with those held by the National Institute of Advanced Industrial Science and Technology (AIST), Japan. SDBD record number 1853.

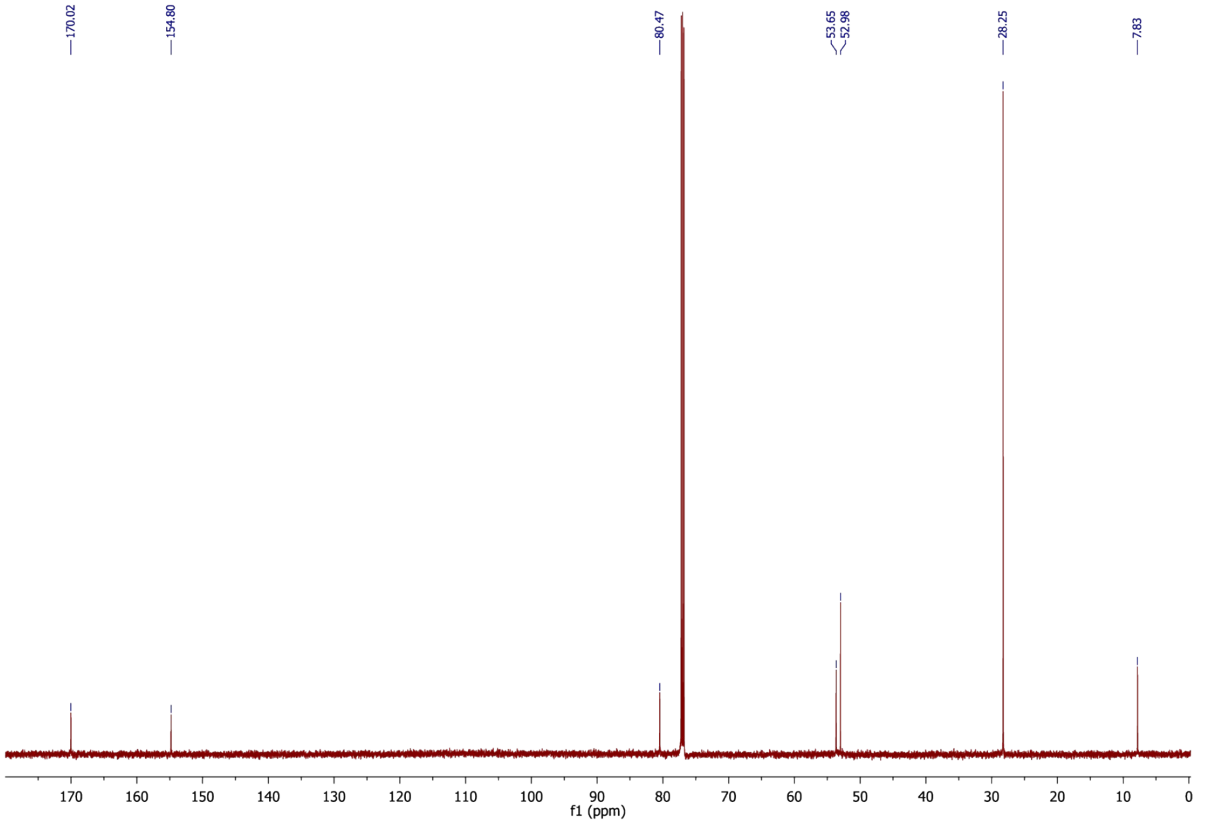
1-Bromoethylbenzene NMR



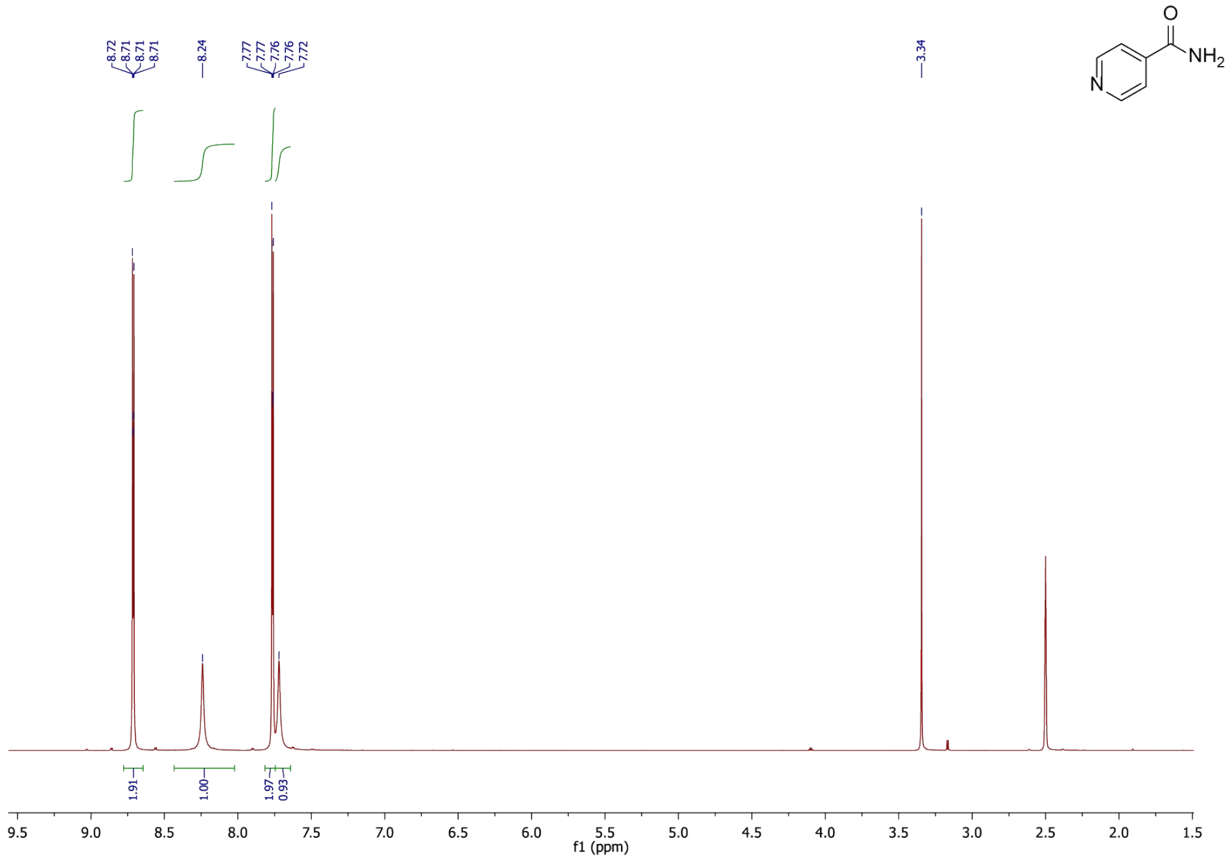


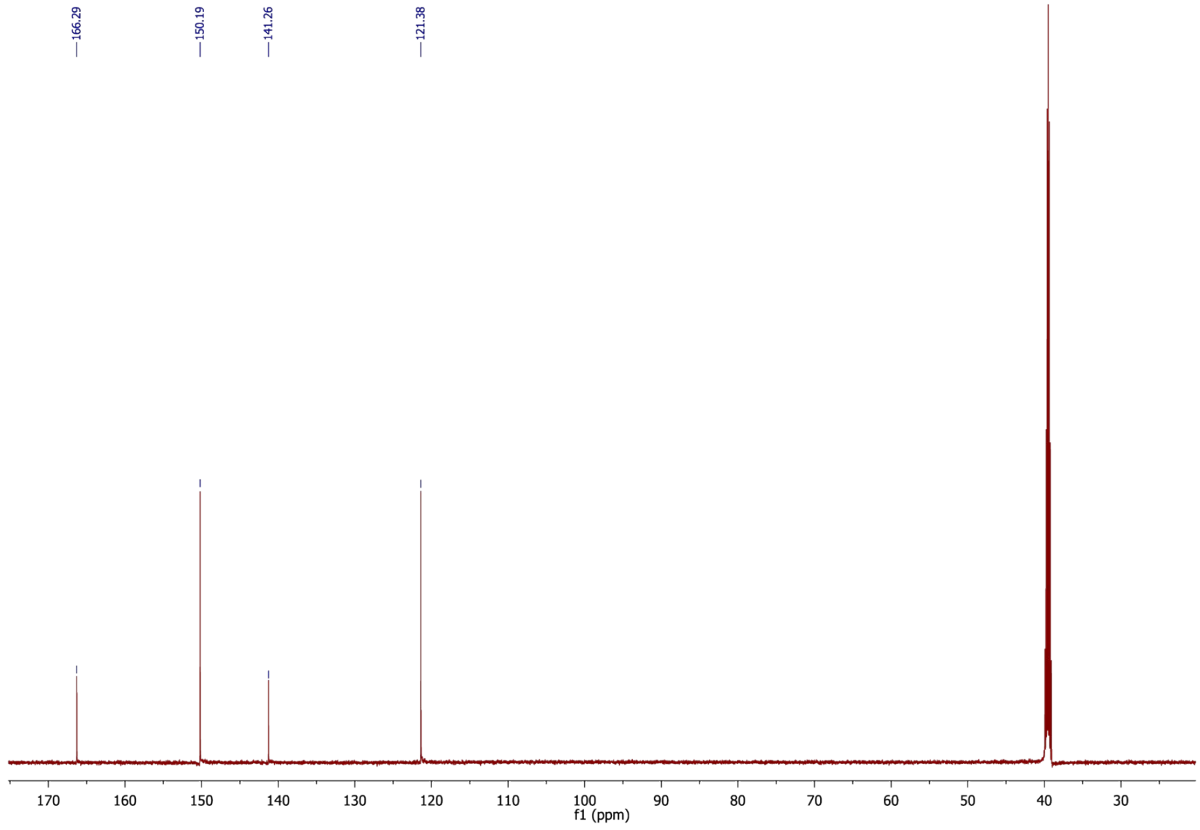
Methyl (2S)-2-((tert-butoxycarbonyl)amino)-3-iodopropanoate NMR



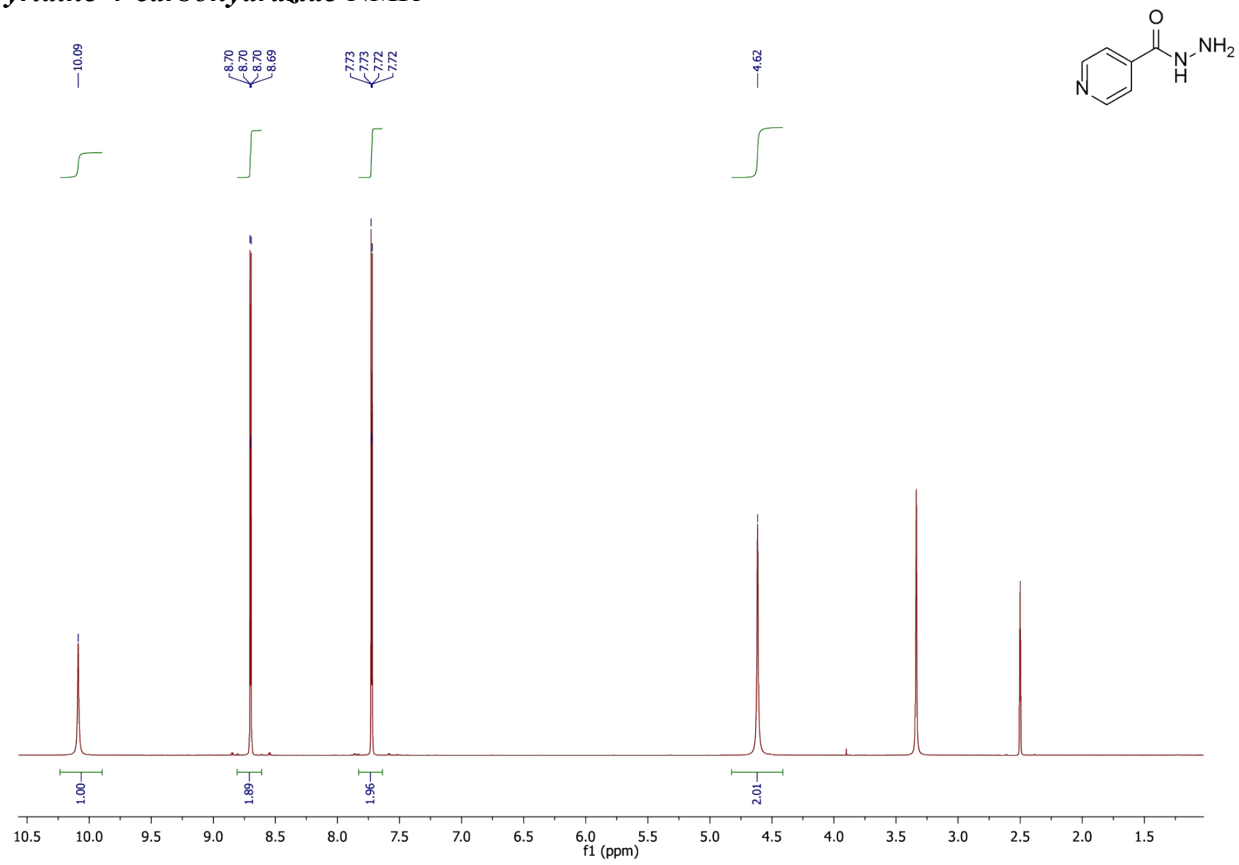


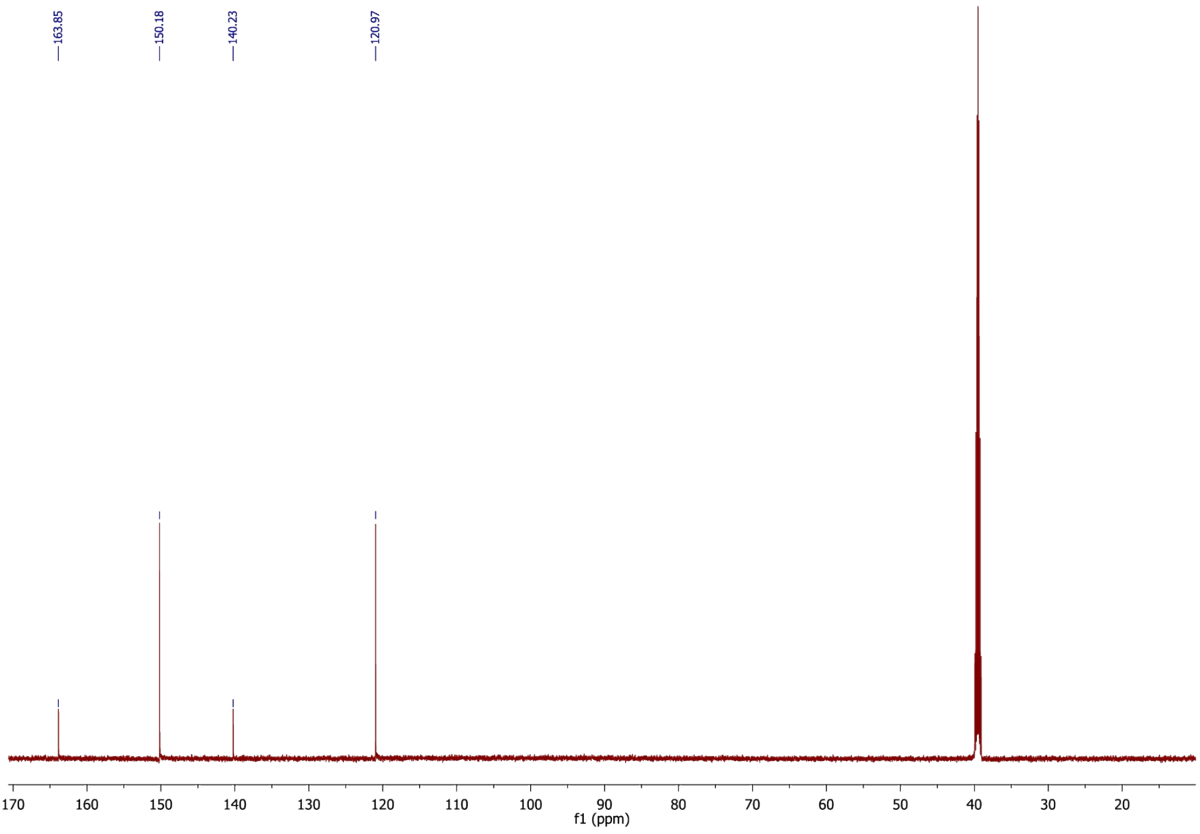
Isonicotinamide NMR





Pyridine-4-carbohydrazide NMR





Appendix A – Example automation code for collection of second fraction using UV-Vis

```php
<?php
    function experiment_start(){
        //Define variables
        newvariable('col1peakcount', 0);
        newvariable('col2peakcount', 0);
        newvariable('col3peakcount', 0);
        newvariable('col4peakcount', 0);
        newvariable('injectiontime', 4);
        newvariable('haltinjections', 0);    //set to 1 to stop injection sequence

        newvariable('col1uvrise', 0.01);
        newvariable('col1uvfall', 0.01);
        newvariable('col2uvrise', 0.01);
        newvariable('col2uvfall', 0.01);
        newvariable('col3uvrise', 0.01);
        newvariable('col3uvfall', 0.01);
        newvariable('col4uvrise', 0.01);
        newvariable('col4uvfall', 0.01);

        //Equipment initialisation
        executeequipmentfunction('pumpmod', 'init', []);
        executeequipmentfunction('pumpco2', 'init', []);

        executeequipmentfunction('pumpcol1', 'init', []);
        executeequipmentfunction('pumpcol2', 'init', []);
        executeequipmentfunction('pumpcol3', 'init', []);
        executeequipmentfunction('pumpcol4', 'init', []);

        executeequipmentfunction('ovencolumns', 'init', []);

        executeequipmentfunction('ovencol1', 'init', []);
        executeequipmentfunction('ovencol2', 'init', []);
        executeequipmentfunction('arduino', 'init', []);

        executeequipmentfunction('loadknauer', 'init', []);

        executeequipmentfunction('uvcol1', 'init', [200]);
        executeequipmentfunction('uvcol2', 'init', [200]);
        executeequipmentfunction('uvcol3', 'init', [200]);
        executeequipmentfunction('uvcol4', 'init', [200]);

        //Set all valves to waste
        executeequipmentfunction('ovencol1', 'setValvePosition', [6, 6]);
        executeequipmentfunction('ovencol2', 'setValvePosition', [6, 6]);

        //Oven temperatures
        executeequipmentfunction('ovencolumns', 'setTemperature', [65]);
        executeequipmentfunction('ovencol1', 'setTemperature', [65]);
        executeequipmentfunction('ovencol2', 'setTemperature', [65]);

        //Large pumps on
        executeequipmentfunction('pumpmod', 'setFlowRate', [8.5]);
        executeequipmentfunction('pumpco2', 'setFlowRate', [76.5]);

        //Column pumps on
        executeequipmentfunction('pumpcol1', 'setFlowRate', [20.0]);
        executeequipmentfunction('pumpcol2', 'setFlowRate', [20.0]);
        executeequipmentfunction('pumpcol3', 'setFlowRate', [20.0]);
        executeequipmentfunction('pumpcol4', 'setFlowRate', [20.0]);

        //Give 6 mins for steady state
        $time = time() + 360;
        addlistener("time()", "greaterthanequalto", $time, "uvzero()");
        addlistener("time()", "greaterthanequalto", $time, "startloading()");

        //Give an extra 30s before injection start
        $time = $time + 30;
        addlistener("time()", "greaterthanequalto", $time, "injectcol4()");

        $injectiontime = getvariable('injectiontime');

        $time = $time + round((60 * (float)$injectiontime) / 4);
        addlistener("time()", "greaterthanequalto", $time, "injectcol3()");

        $time = $time + round((60 * (float)$injectiontime) / 4);
```

```
        addlistener("time()", "greaterthanequalto", $time, "injectcol2()");

        $time = $time + round((60 * (float)$injectiontime) / 4);
        addlistener("time()", "greaterthanequalto", $time, "injectcol1()");
    }

    function uvzero(){
        executeequipmentfunction('uvcol1', 'performAutoZero', []);
        executeequipmentfunction('uvcol2', 'performAutoZero', []);
        executeequipmentfunction('uvcol3', 'performAutoZero', []);
        executeequipmentfunction('uvcol4', 'performAutoZero', []);

        $time = time() + 5;
        addlistener("time()", "greaterthanequalto", $time, "startuvlisteners()");
    }

    function startloading(){
        executeequipmentfunction('loadknauer', 'setFlowRate', [4]);
    }

    function startuvlisteners(){
        addlistener("uvrisecol1()", "equals", "1", "nullfunction()");
        addlistener("uvrisecol2()", "equals", "1", "nullfunction()");
        addlistener("uvrisecol3()", "equals", "1", "nullfunction()");
        addlistener("uvrisecol4()", "equals", "1", "nullfunction()");
    }

    //------------------------------------------------------------------------------------------------ Injections
    function injectcol1(){
        //Only inject if haltinjections is zero
        $halt = getvariable('haltinjections');
        if($halt == 0){
            executeequipmentfunction('arduino', 'switchposition', ["1", "1"]);

            //Return to load
            $time = time() + 5;
            addlistener("time()", "greaterthanequalto", $time, "executeequipmentfunction('arduino', 'switchposition',
['1', '0'])");
        }

        //Add listener for next injection
        $injectiontime = getvariable('injectiontime');

        $time = time() + round(60 * (float)$injectiontime);
        addlistener("time()", "greaterthanequalto", $time, "injectcol1()");

        $time = $time + 20;
        addlistener("time()", "greaterthanequalto", $time, "updatevariable('col1peakcount', 0)");
    }

    function injectcol2(){
        //Only inject if haltinjections is zero
        $halt = getvariable('haltinjections');
        if($halt == 0){
            executeequipmentfunction('arduino', 'switchposition', ["2", "1"]);

            //Return to load
            $time = time() + 5;
            addlistener("time()", "greaterthanequalto", $time, "executeequipmentfunction('arduino', 'switchposition',
['2', '0'])");
        }

        //Add listener for next injection
        $injectiontime = getvariable('injectiontime');

        $time = time() + round(60 * (float)$injectiontime);
        addlistener("time()", "greaterthanequalto", $time, "injectcol2()");

        $time = $time + 20;
        addlistener("time()", "greaterthanequalto", $time, "updatevariable('col2peakcount', 0)");
    }

    function injectcol3(){
        //Only inject if haltinjections is zero
        $halt = getvariable('haltinjections');
        if($halt == 0){
            executeequipmentfunction('arduino', 'switchposition', ["3", "1"]);

            //Return to load
            $time = time() + 5;
```

```
            addlistener("time()", "greaterthanequalto", $time, "executeequipmentfunction('arduino', 'switchposition',
['3', '0'])");
        }

        //Add listener for next injection
        $injectiontime = getvariable('injectiontime');

        $time = time() + round(60 * (float)$injectiontime);
        addlistener("time()", "greaterthanequalto", $time, "injectcol3()");

        $time = $time + 20;
        addlistener("time()", "greaterthanequalto", $time, "updatevariable('col3peakcount', 0)");
    }

    function injectcol4(){
        //Only inject if haltinjections is zero
        $halt = getvariable('haltinjections');
        if($halt == 0){
            executeequipmentfunction('arduino', 'switchposition', ["4", "1"]);

            //Return to load
            $time = time() + 5;
            addlistener("time()", "greaterthanequalto", $time, "executeequipmentfunction('arduino', 'switchposition',
['4', '0'])");
        }

        //Add listener for next injection
        $injectiontime = getvariable('injectiontime');

        $time = time() + round(60 * (float)$injectiontime);
        addlistener("time()", "greaterthanequalto", $time, "injectcol4()");

        $time = $time + 20;
        addlistener("time()", "greaterthanequalto", $time, "updatevariable('col4peakcount', 0)");
    }

    //------------------------------------------------------------------------------------------------ UV monitors
    function uvrisecol1(){
        $uvvalue = getequipmentdata('uvcol1')['absorbance'];

        //Check if is number
        if(is_numeric($uvvalue)){
            $uvvalue = floatval($uvvalue);

            //Check if above cutoff
            if($uvvalue >= floatval(getvariable('col1uvrise'))){
                //Add one to peak count variable
                updatevariable('col1peakcount', (int)getvariable('col1peakcount') + 1);

                //Wait for fall in uv
                addlistener("uvfallcol1()", "equals", "1", "nullfunction()");

                //Check if collection needed
                if(getvariable('col1peakcount') == 3){ //3
                    executeequipmentfunction('ovencol1', 'setValvePosition', [1, 6]);
                }

                return "1";
            }
        }

        return 0;
    }

    function uvfallcol1(){
        $uvvalue = getequipmentdata('uvcol1')['absorbance'];

        //Check if is number and under uv fall cutoff
        if(is_numeric($uvvalue)){
            $uvvalue = floatval($uvvalue);
            if($uvvalue <= floatval(getvariable('col1uvfall'))){
                //Add one to peak count variable
                updatevariable('col1peakcount', (int)getvariable('col1peakcount') + 1);

                //Check if switch needed
                if(getvariable('col1peakcount') == 4){ //4
                    $time = time() + 5;
                    addlistener("time()", "greaterthanequalto", $time, "returnvalvecol1()");
                }
```

```
            //Wait for uv reset
            addlistener("uvresetcol1()", "equals", "1", "nullfunction()");

            return "1";
        }
    }

    return 0;
}

function returnvalvecol1(){
    executeequipmentfunction('ovencol1', 'setValvePosition', [6, 6]);

    return 0;
}

function uvresetcol1(){
    $uvvalue = getequipmentdata('uvcol1')['absorbance'];

    //Check if is number and under uv rise cutoff
    if(is_numeric($uvvalue)){
        $uvvalue = floatval($uvvalue);

        if($uvvalue <= floatval(getvariable('col1uvrise'))){
            addlistener("uvrisecol1()", "equals", "1", "nullfunction()");

            return "1";
        }
    }

    return 0;
}

function uvrisecol2(){
    $uvvalue = getequipmentdata('uvcol2')['absorbance'];

    //Check if is number
    if(is_numeric($uvvalue)){
        $uvvalue = floatval($uvvalue);

        //Check if above cutoff
        if($uvvalue >= floatval(getvariable('col2uvrise'))){
            //Add one to peak count variable
            updatevariable('col2peakcount', (int)getvariable('col2peakcount') + 1);

            //Wait for fall in uv
            addlistener("uvfallcol2()", "equals", "1", "nullfunction()");

            //Check if collection needed
            if(getvariable('col2peakcount') == 3){
                executeequipmentfunction('ovencol1', 'setValvePosition', [6, 1]);
            }

            return "1";
        }
    }

    return 0;
}

function uvfallcol2(){
    $uvvalue = getequipmentdata('uvcol2')['absorbance'];

    //Check if is number and under uv fall cutoff
    if(is_numeric($uvvalue)){
        $uvvalue = floatval($uvvalue);
        if($uvvalue <= floatval(getvariable('col2uvfall'))){
            //Add one to peak count variable
            updatevariable('col2peakcount', (int)getvariable('col2peakcount') + 1);

            //Check if switch needed
            if(getvariable('col2peakcount') == 4){
                $time = time() + 5;
                addlistener("time()", "greaterthanequalto", $time, "returnvalvecol2()");
            }

            //Wait for uv reset
            addlistener("uvresetcol2()", "equals", "1", "nullfunction()");

            return "1";
```

```
        }
    }

    return 0;
}

function returnvalvecol2(){
    executeequipmentfunction('ovencol1', 'setValvePosition', [6, 6]);

    return 0;
}

function uvresetcol2(){
    $uvvalue = getequipmentdata('uvcol2')['absorbance'];

    //Check if is number and under uv rise cutoff
    if(is_numeric($uvvalue)){
        $uvvalue = floatval($uvvalue);

        if($uvvalue <= floatval(getvariable('col2uvrise'))){
            addlistener("uvrisecol2()", "equals", "1", "nullfunction()");

            return "1";
        }
    }

    return 0;
}

function uvrisecol3(){
    $uvvalue = getequipmentdata('uvcol3')['absorbance'];

    //Check if is number
    if(is_numeric($uvvalue)){
        $uvvalue = floatval($uvvalue);

        //Check if above cutoff
        if($uvvalue >= floatval(getvariable('col3uvrise'))){
            //Add one to peak count variable
            updatevariable('col3peakcount', (int)getvariable('col3peakcount') + 1);

            //Wait for fall in uv
            addlistener("uvfallcol3()", "equals", "1", "nullfunction()");

            //Check if collection needed
            if(getvariable('col3peakcount') == 3){
                executeequipmentfunction('ovencol2', 'setValvePosition', [1, 6]);
            }

            return "1";
        }
    }

    return 0;
}

function uvfallcol3(){
    $uvvalue = getequipmentdata('uvcol3')['absorbance'];

    //Check if is number and under uv fall cutoff
    if(is_numeric($uvvalue)){
        $uvvalue = floatval($uvvalue);
        if($uvvalue <= floatval(getvariable('col3uvfall'))){
            //Add one to peak count variable
            updatevariable('col3peakcount', (int)getvariable('col3peakcount') + 1);

            //Check if switch needed
            if(getvariable('col3peakcount') == 4){
                $time = time() + 5;
                addlistener("time()", "greaterthanequalto", $time, "returnvalvecol3()");
            }

            //Wait for uv reset
            addlistener("uvresetcol3()", "equals", "1", "nullfunction()");

            return "1";
        }
    }

    return 0;
```

```
}

function returnvalvecol3(){
    executeequipmentfunction('ovencol2', 'setValvePosition', [6, 6]);

    return 0;
}

function uvresetcol3(){
    $uvvalue = getequipmentdata('uvcol3')['absorbance'];

    //Check if is number and under uv rise cutoff
    if(is_numeric($uvvalue)){
        $uvvalue = floatval($uvvalue);

        if($uvvalue <= floatval(getvariable('col3uvrise'))){
            addlistener("uvrisecol3()", "equals", "1", "nullfunction()");

            return "1";
        }
    }

    return 0;
}

function uvrisecol4(){
    $uvvalue = getequipmentdata('uvcol4')['absorbance'];

    //Check if is number
    if(is_numeric($uvvalue)){
        $uvvalue = floatval($uvvalue);

        //Check if above cutoff
        if($uvvalue >= floatval(getvariable('col4uvrise'))){
            //Add one to peak count variable
            updatevariable('col4peakcount', (int)getvariable('col4peakcount') + 1);

            //Wait for fall in uv
            addlistener("uvfallcol4()", "equals", "1", "nullfunction()");

            //Check if collection needed
            if(getvariable('col4peakcount') == 3){
                executeequipmentfunction('ovencol2', 'setValvePosition', [6, 1]);
            }

            return "1";
        }
    }

    return 0;
}

function uvfallcol4(){
    $uvvalue = getequipmentdata('uvcol4')['absorbance'];

    //Check if is number and under uv fall cutoff
    if(is_numeric($uvvalue)){
        $uvvalue = floatval($uvvalue);
        if($uvvalue <= floatval(getvariable('col4uvfall'))){
            //Add one to peak count variable
            updatevariable('col4peakcount', (int)getvariable('col4peakcount') + 1);

            //Check if switch needed
            if(getvariable('col4peakcount') == 4){
                $time = time() + 5;
                addlistener("time()", "greaterthanequalto", $time, "returnvalvecol4()");
            }

            //Wait for uv reset
            addlistener("uvresetcol4()", "equals", "1", "nullfunction()");

            return "1";
        }
    }

    return 0;
}

function returnvalvecol4(){
    executeequipmentfunction('ovencol2', 'setValvePosition', [6, 6]);
```

```
        return 0;
    }

    function uvresetcol4(){
        $uvvalue = getequipmentdata('uvcol4')['absorbance'];

        //Check if is number and under uv rise cutoff
        if(is_numeric($uvvalue)){
            $uvvalue = floatval($uvvalue);

            if($uvvalue <= floatval(getvariable('col4uvrise'))){
                addlistener("uvrisecol4()", "equals", "1", "nullfunction()");

                return "1";
            }
        }

        return 0;
    }

    function nullfunction(){
        return 0;
    }

    function experiment_shutdown(){
        removealllisteners();

        //Set all valves to waste
        executeequipmentfunction('ovencol1', 'setValvePosition', [6, 6]);
        executeequipmentfunction('ovencol2', 'setValvePosition', [6, 6]);

        //Large pumps off
        executeequipmentfunction('pumpmod', 'setFlowRate', [0.0]);
        executeequipmentfunction('pumpco2', 'setFlowRate', [0.0]);

        //Column pumps off
        executeequipmentfunction('pumpcol1', 'setFlowRate', [0.0]);
        executeequipmentfunction('pumpcol2', 'setFlowRate', [0.0]);
        executeequipmentfunction('pumpcol3', 'setFlowRate', [0.0]);
        executeequipmentfunction('pumpcol4', 'setFlowRate', [0.0]);

        //Loop load pump off
        executeequipmentfunction('loadknauer', 'setFlowRate', [0.0]);

        stopexperiment();
    }
?>
```