Supporting Information

Optical Control of Phosphatidic Acid Signaling

Reika Tei[#a], Johannes Morstein[#b], Andrej Shemet[b], Dirk Trauner[*b], and Jeremy M. Baskin[*a]

[a] Department of Chemistry and Chemical Biology and Weill Institute for Cell and Molecular Biology, Cornell University, Ithaca, New York 14850, USA

[b] Department of Chemistry, New York University, New York, New York 10003, USA

* Correspondence: dirktrauner@nyu.edu and jeremy.baskin@cornell.edu

# These authors contributed equally to this work.

Table of Contents

Supplementary Figures

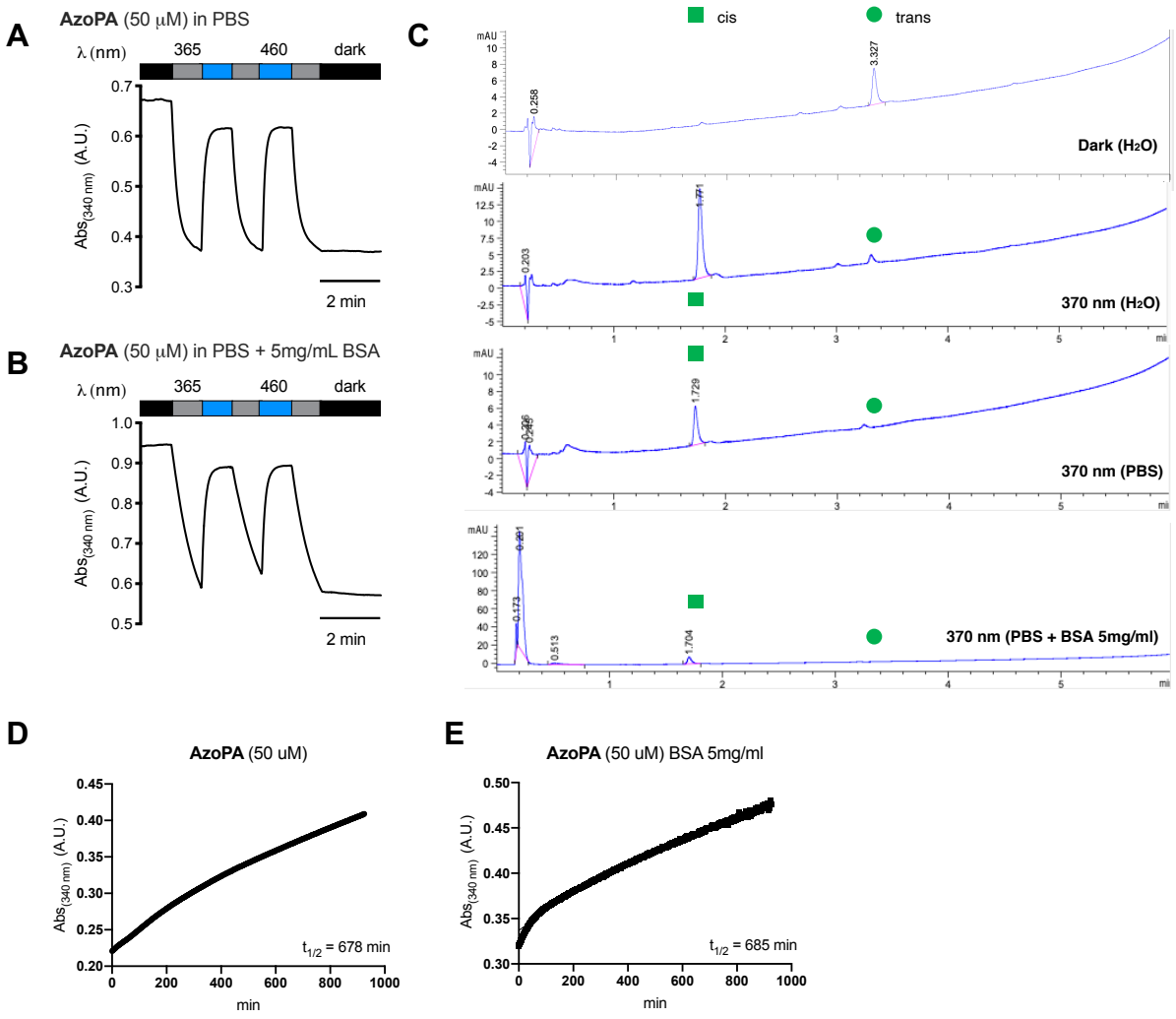
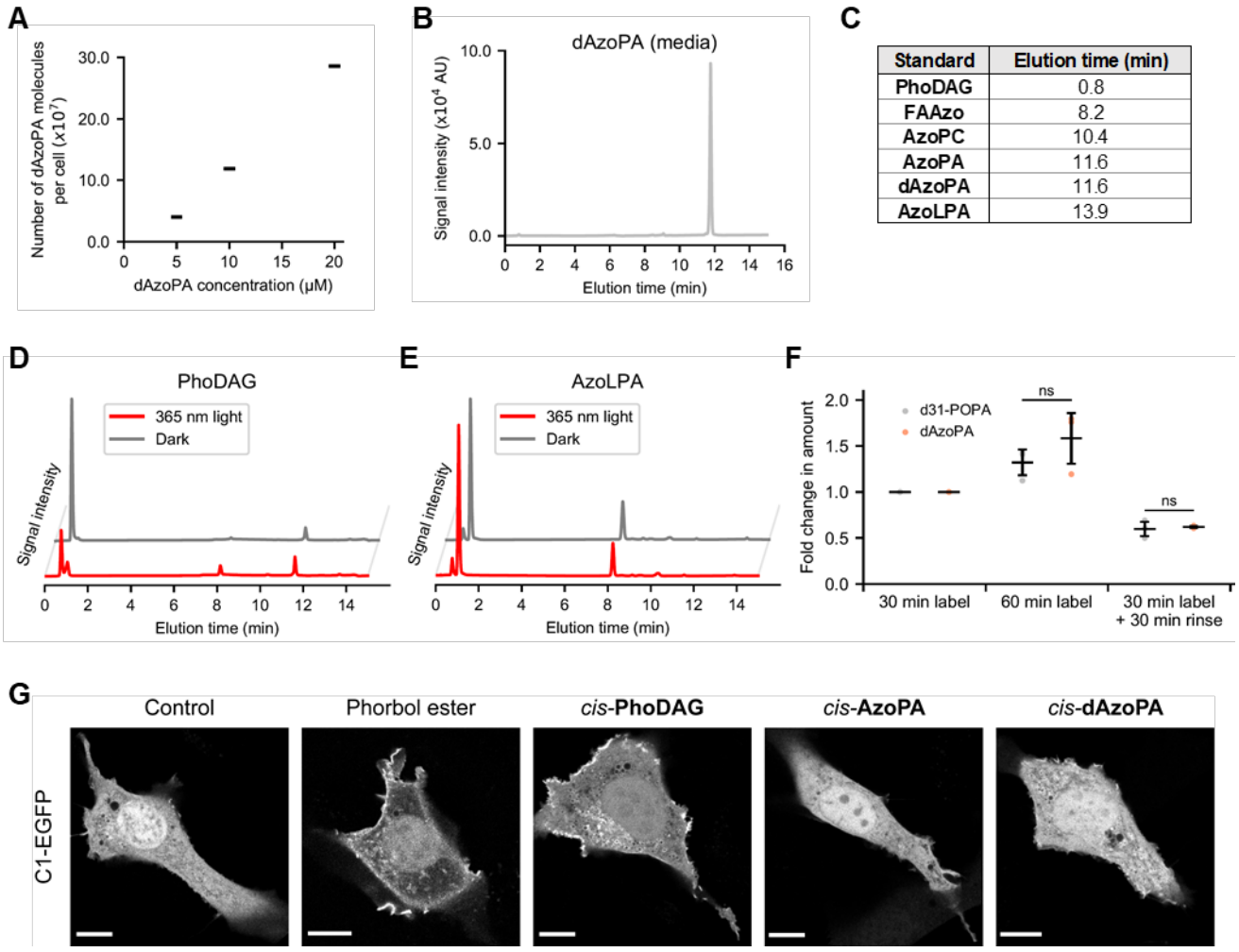


Figure S1 | Effect of 5 mg/mL Bovine Serum Albumin (BSA) on photoswitching properties of **AzoPA** (50 μM). (A,B) Reversible switching of **AzoPA** (50 μM) in the absence (A) and presence (B) of BSA. (C) 254 nm LCMS traces **AzoPA** (50 μM) before or after irradiation with 370 nm light and in the absence and presence of BSA. (D,E) Thermal relaxation of **AzoPA** (50 μM) in the absence (D) and presence (E) of BSA.



Standard	Elution time (min)
PhoDAG	0.8
FAAzo	8.2
AzoPC	10.4
AzoPA	11.6
dAzoPA	11.6
AzoLPA	13.9

Figure S2 | Characterization of cellular uptake and metabolism of PA and photolipids. (A) Quantification of **dAzoPA** uptake in NIH 3T3 cells. Cells were treated with the indicated amount of lipid for 1 h, followed by lipid extraction and quantification by HPLC comparing to synthetic standards, to determine the relative amount of **dAzoPA**. To obtain the average number of molecules per cell, this number was divided by the number of cells on the plate. (B) HPLC chromatogram of lipid extracts from the culture media of NIH 3T3 cells after incubation with 20 μM *cis*-**dAzoPA** for 1 h. (C) Observed HPLC elution times of synthetic photolipid standards. (D–E) HPLC chromatogram of lipid extracts from NIH 3T3 cells treated with 20 μM **PhoDAG** (D) or **AzoLPA** (E) in light-induced *cis* forms (red) or dark-adapted *trans* forms (gray) for 1 h. Prior to the injection, the samples were irradiated with 460 nm light to equilibrate azobenzene moieties

into the *trans* forms and normalize the absorption. (F) Amount of a deuterium-labeled version of POPA (d31-POPA) and **dAzoPA** detected in NIH 3T3 cells treated with 20 μM of either compound for 30 min, 60 min or 30 min followed by a 30 min rinse (chase). For the rinse, cells were washed with PBS three times and incubated for an additional 30 min in fresh media. ns, not significant (Student's t-test). (G) Confocal microscopy images of live cells expressing the fluorescent DAG biosensor C1-EGFP. NIH 3T3 cells transfected with C1-EGFP were treated with either phorbol 12,13-dibutyrate (phorbol ester, 100 nM, 10 s) or the indicated photolipid (20 μM, 20 min). Scale bars: 10 μm.

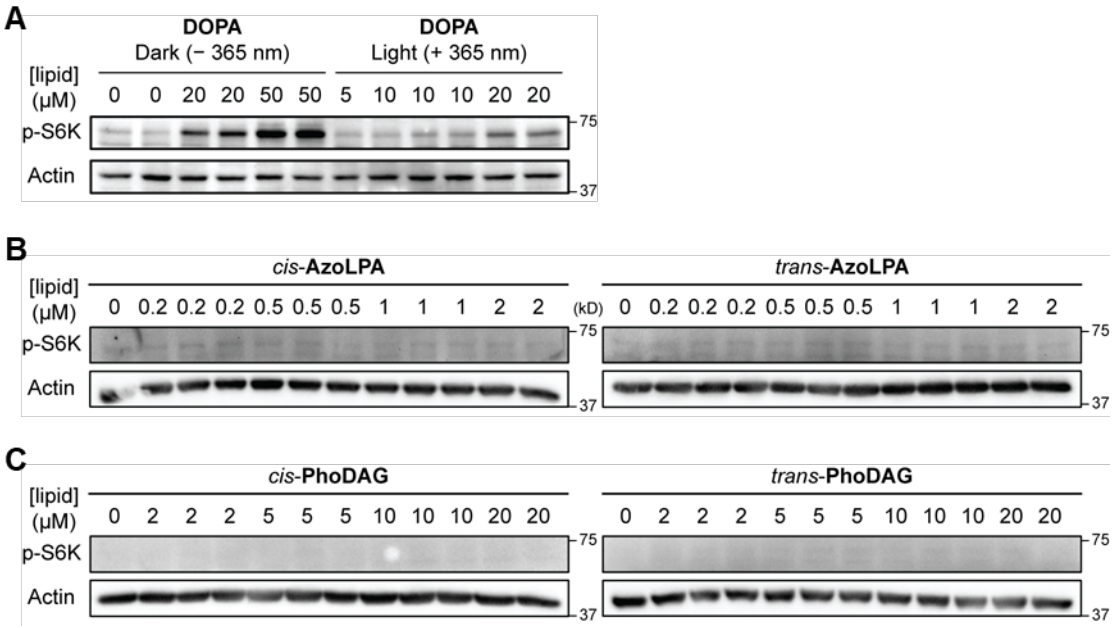


Figure S3 | Evaluation of p-S6K activity in NIH 3T3 cells. (A) Western blot analysis of cells treated with POPA at different concentrations (0–100 μM) for 1 h with or without irradiation of 365 nm light (5 ms every 15 s). (B–C) Western blot analysis of cells treated with **AzoLPA** (B) or **PhoDAG** (C) in light-induced *cis* form or dark-adapted *trans* form at different concentrations (0–20 μM) for 30 min.

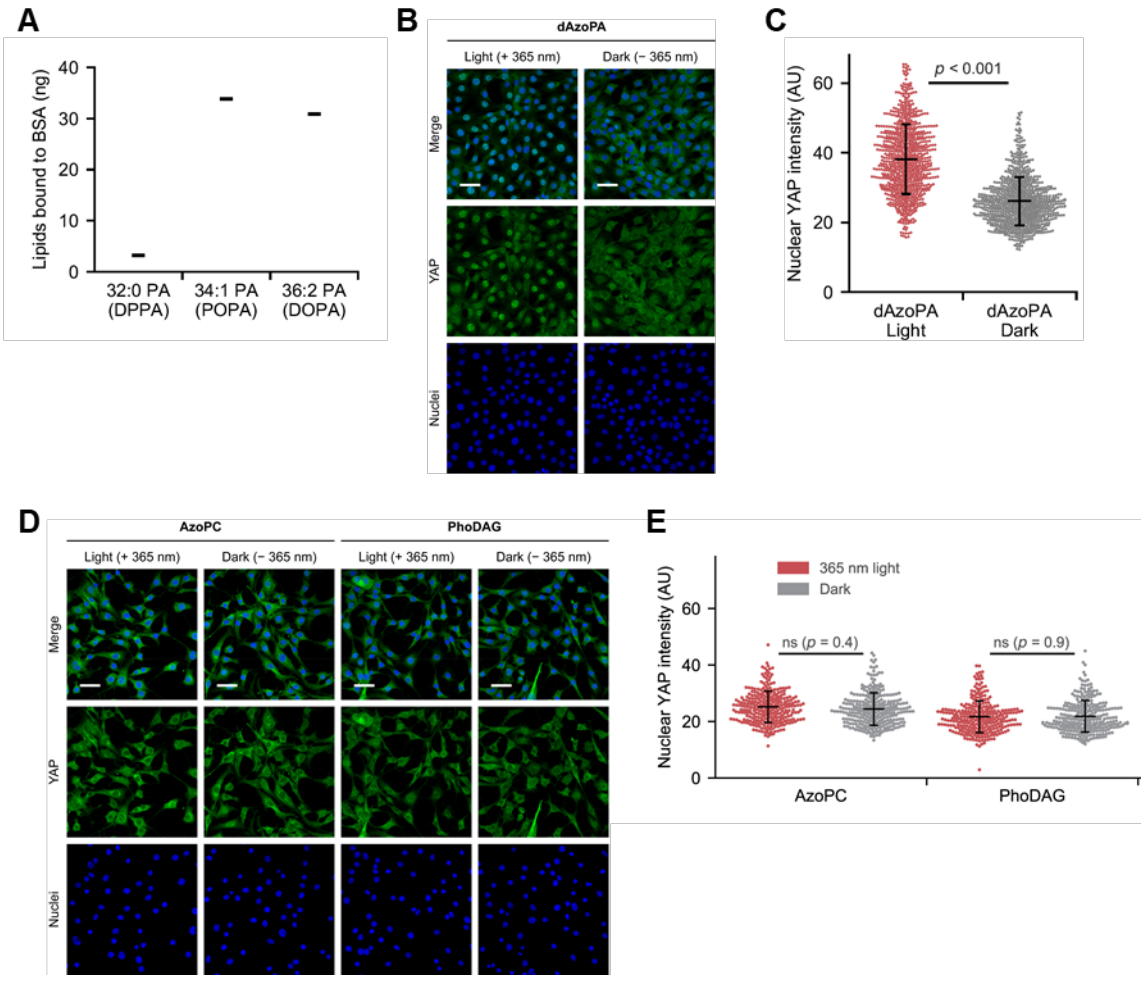


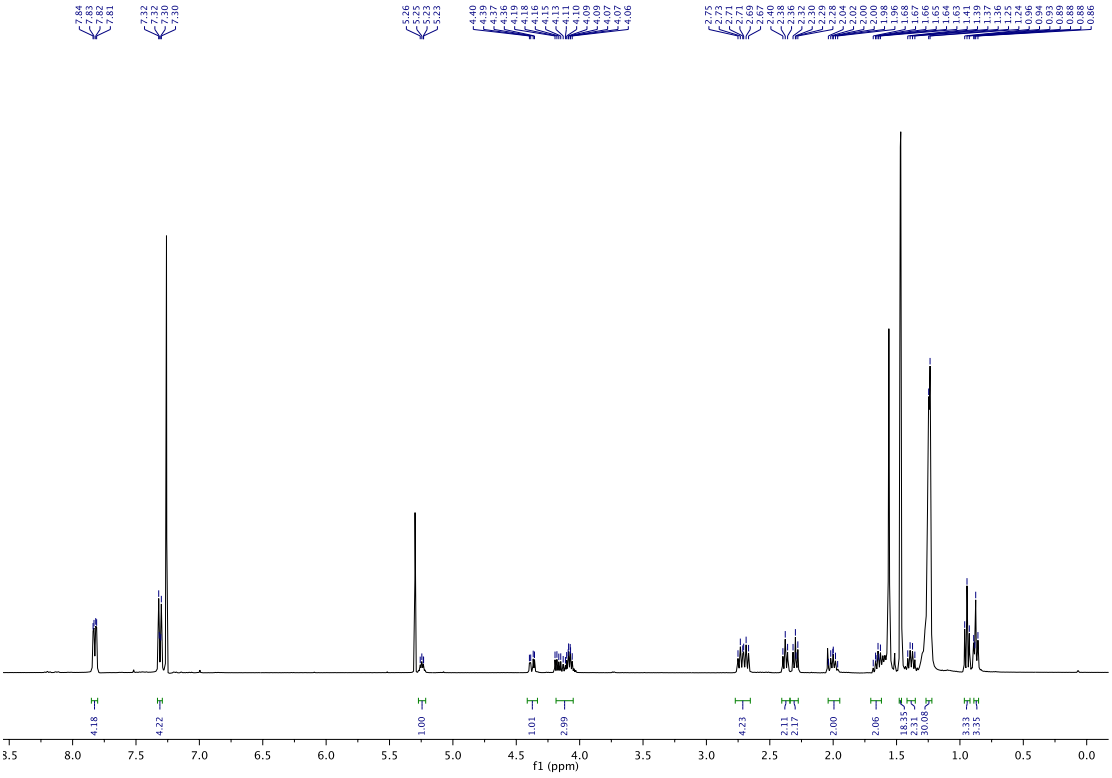
Figure S4 | Evaluation of nuclear YAP localization in NIH 3T3 cells. (A) Quantification of the amount of the PA analogs with different acyl tails that were complexed to BSA. Equivalent amounts of each lipid and BSA were mixed to form lipid-BSA complexes in PBS, and complexes were then separated from uncomplexed lipids by size-exclusion chromatography. Lipids were then extracted from purified lipid-BSA complexes and quantified by LC–MS analysis. (B) Confocal microscopy images of cells treated with **dAzoPA** (10 μM) for 1 h with or without 365 nm light (5 ms every 15 s) and immunostained for YAP. (C) Quantification of nuclear YAP levels in A. The plots show mean nuclear YAP intensity in each cell. Black horizontal bars indicate mean (n = 810) and vertical error bars indicate standard deviation. (D) Confocal microscopy images of cells treated with the indicated lipids (10 μM) in light-induced *cis* form or dark-adapted *trans* form for 1

h and immunostained for YAP. Intermittent 365 nm light (5 ms every 15 s) was additionally applied during the incubation of cells with the lipids. (E) Quantification of nuclear YAP levels in C. Black horizontal bars indicate mean (n = 280) and vertical error bars indicate standard deviation. Green, YAP; blue, DAPI (nuclei). Scale bars: 50 μm.

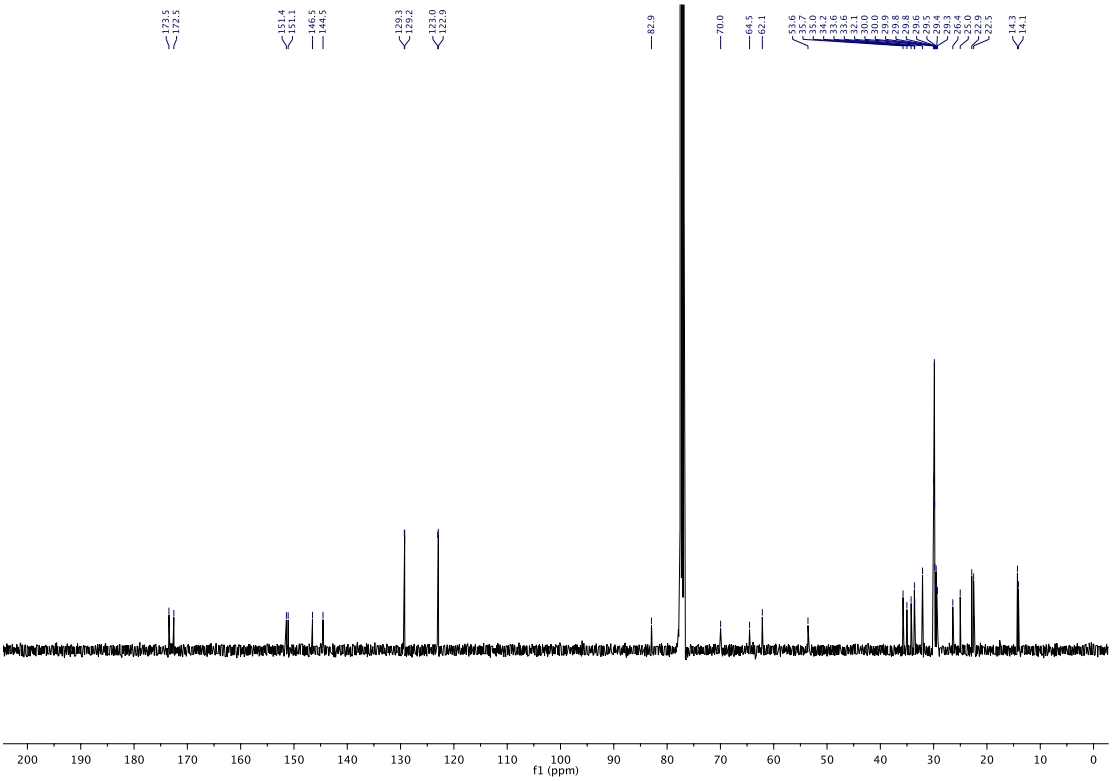
^{1}H,^{13}C, and ^{31}P NMR Spectra

Di-*tert*-butyl AzoPA (1)

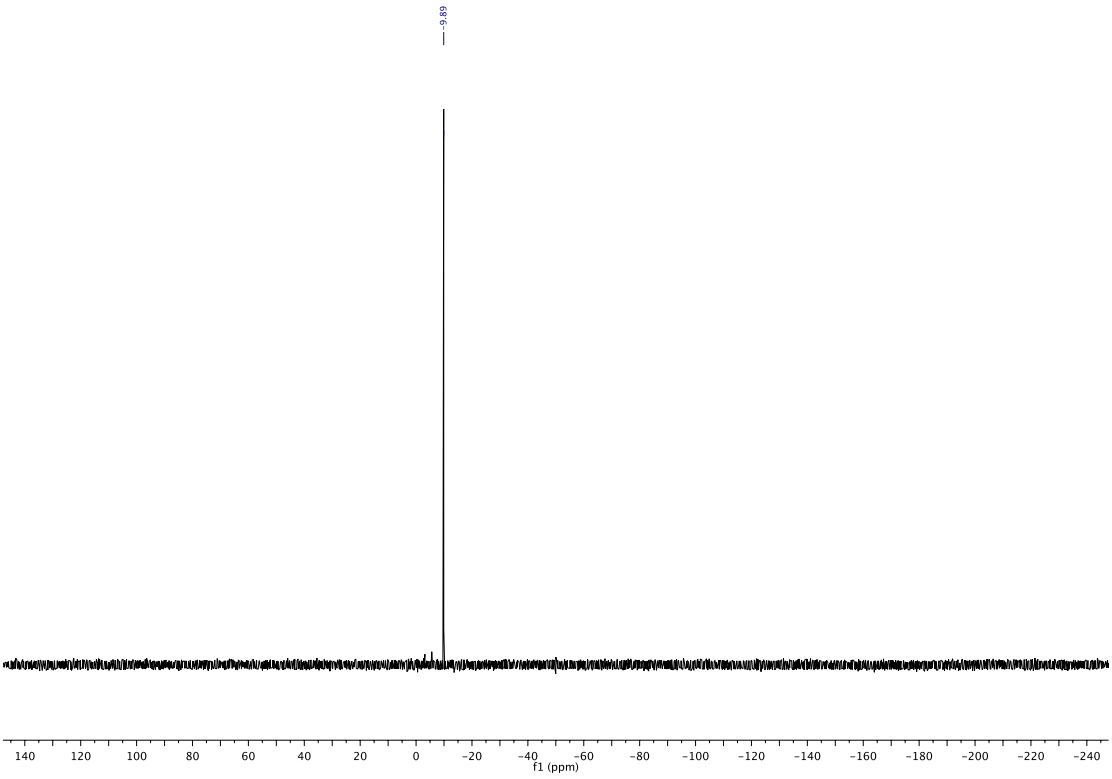
^{1}H NMR



^{13}C NMR

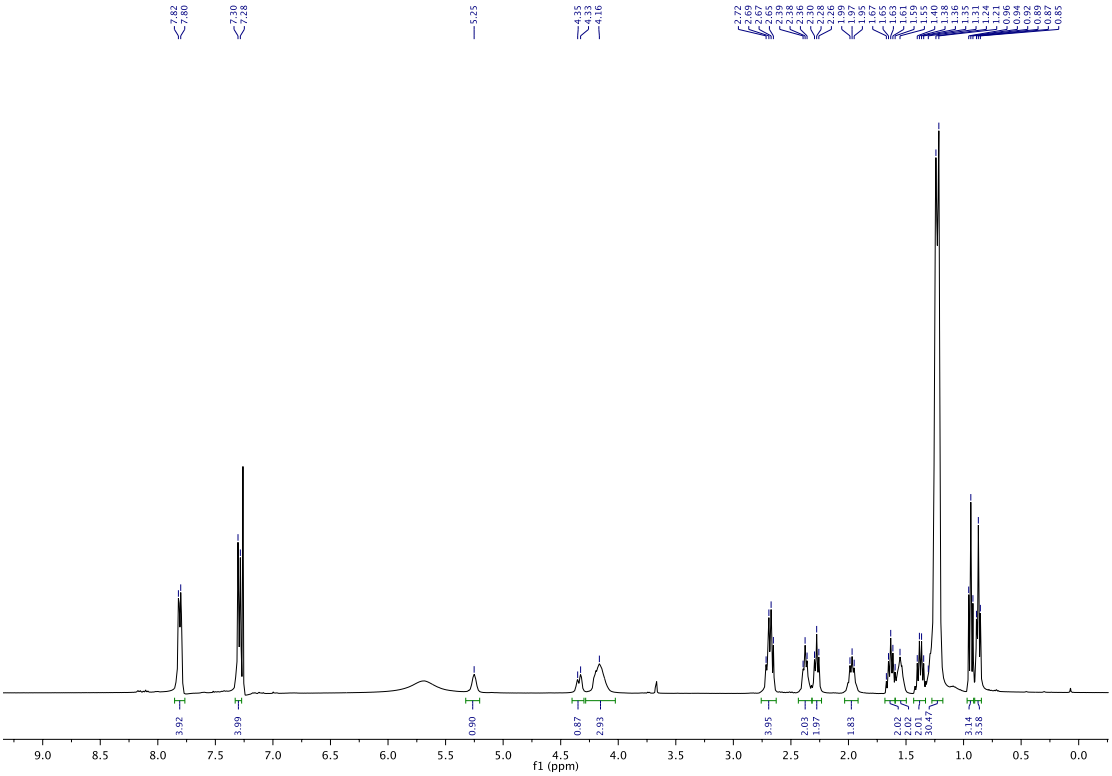


^{31}P NMR

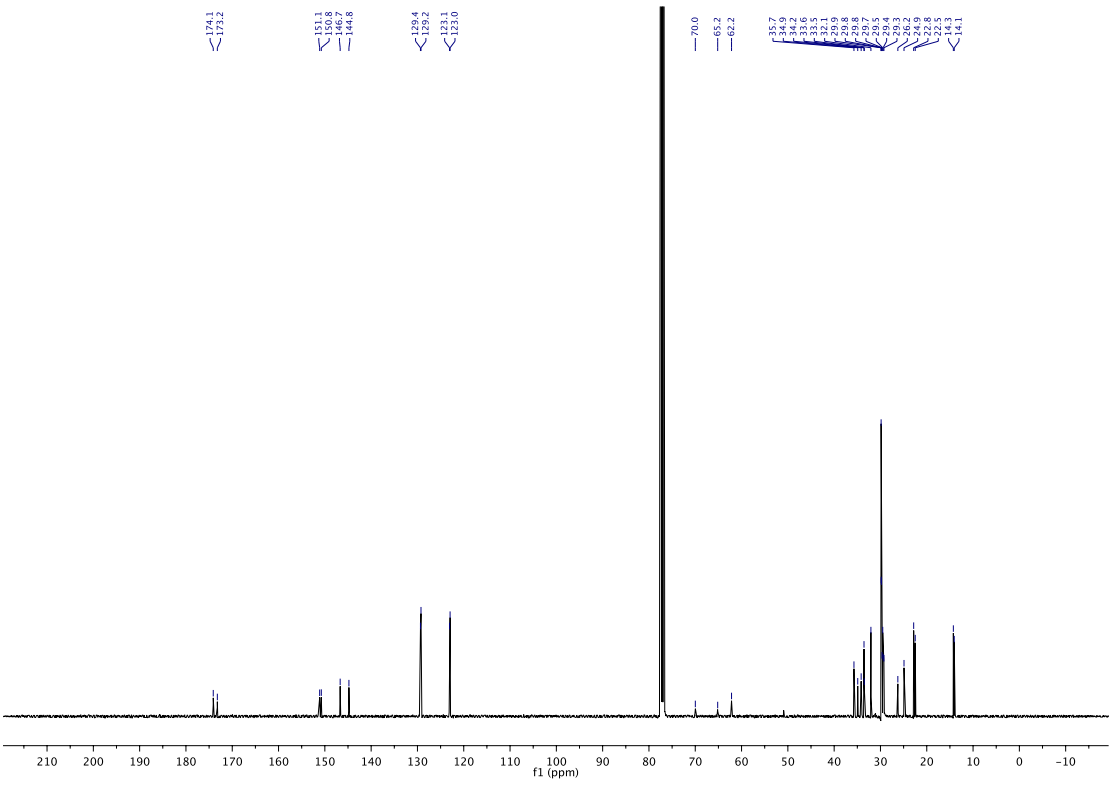


AzoPA

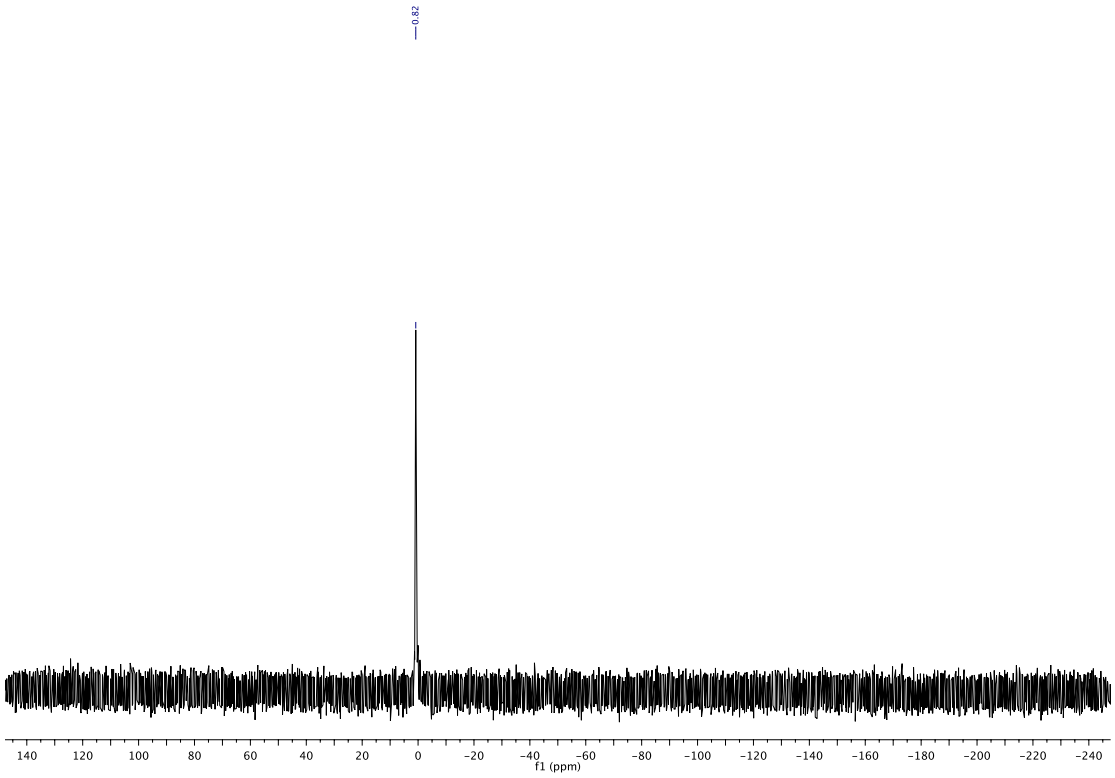
^{1}H NMR



^{13}C NMR

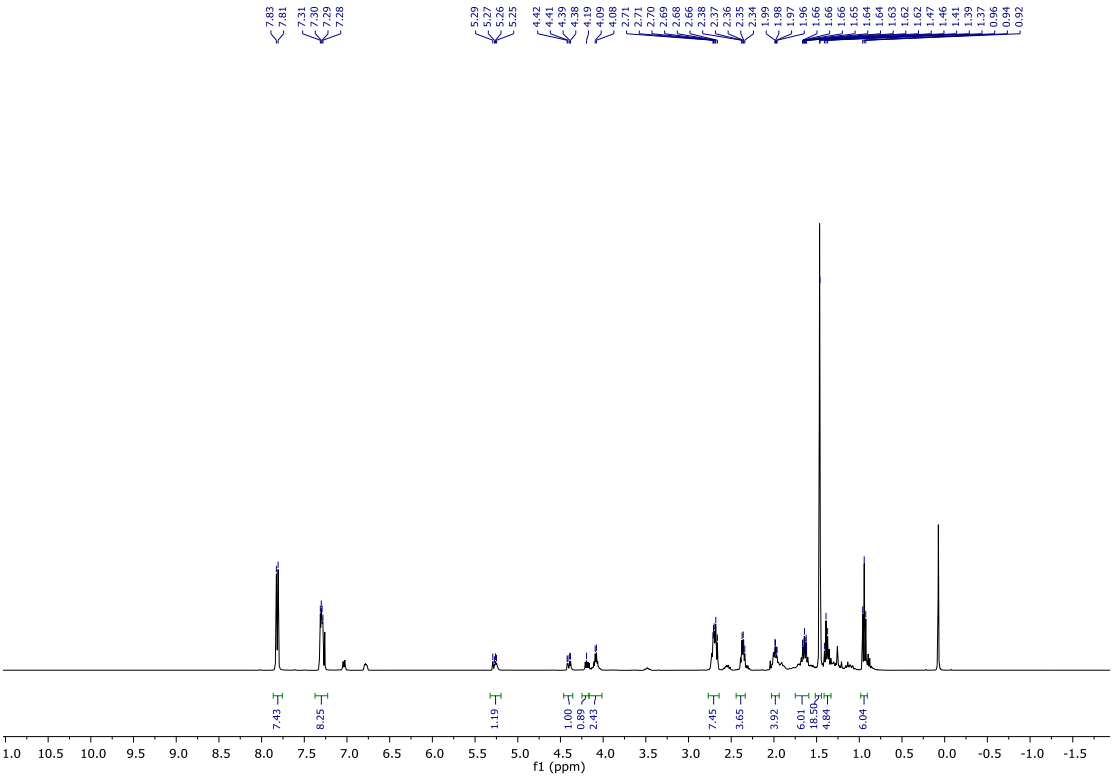


^{31}P NMR

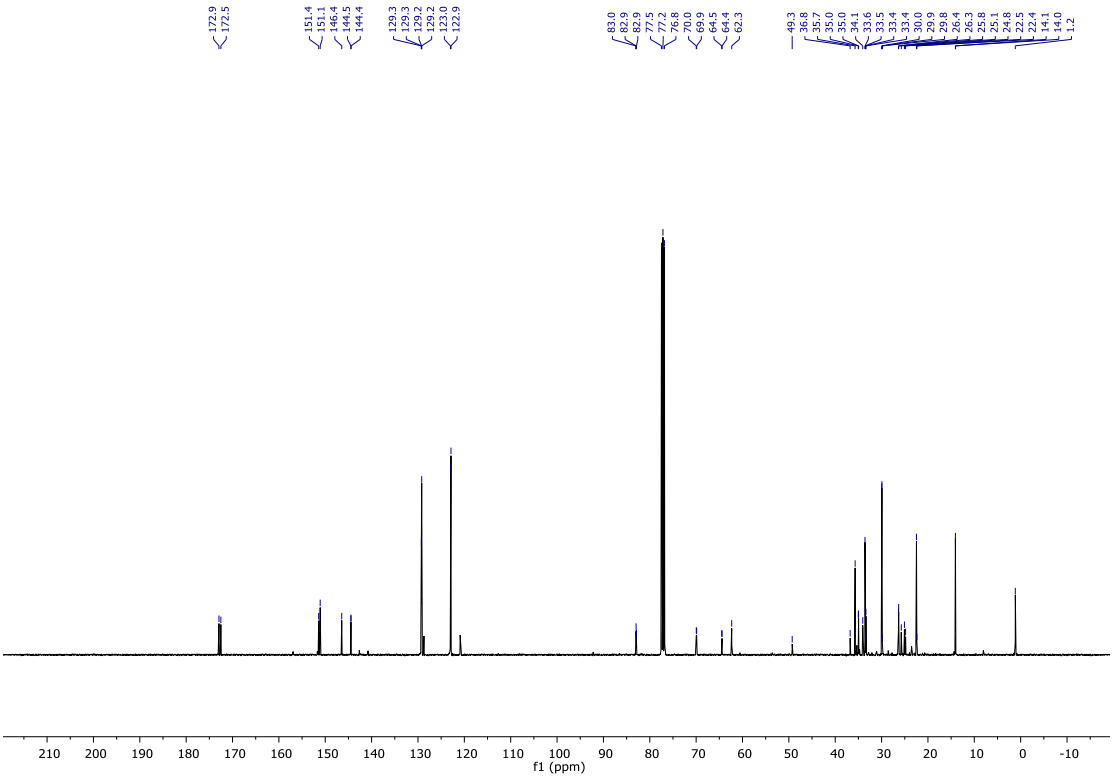

0.82
140
120
100
80
60
40
20
0
-20
-40
-60
-80
-100
-120
-140
-160
-180
-200
-220
-240
f1 (ppm)

Di-*tert*-butyl dAzoPA

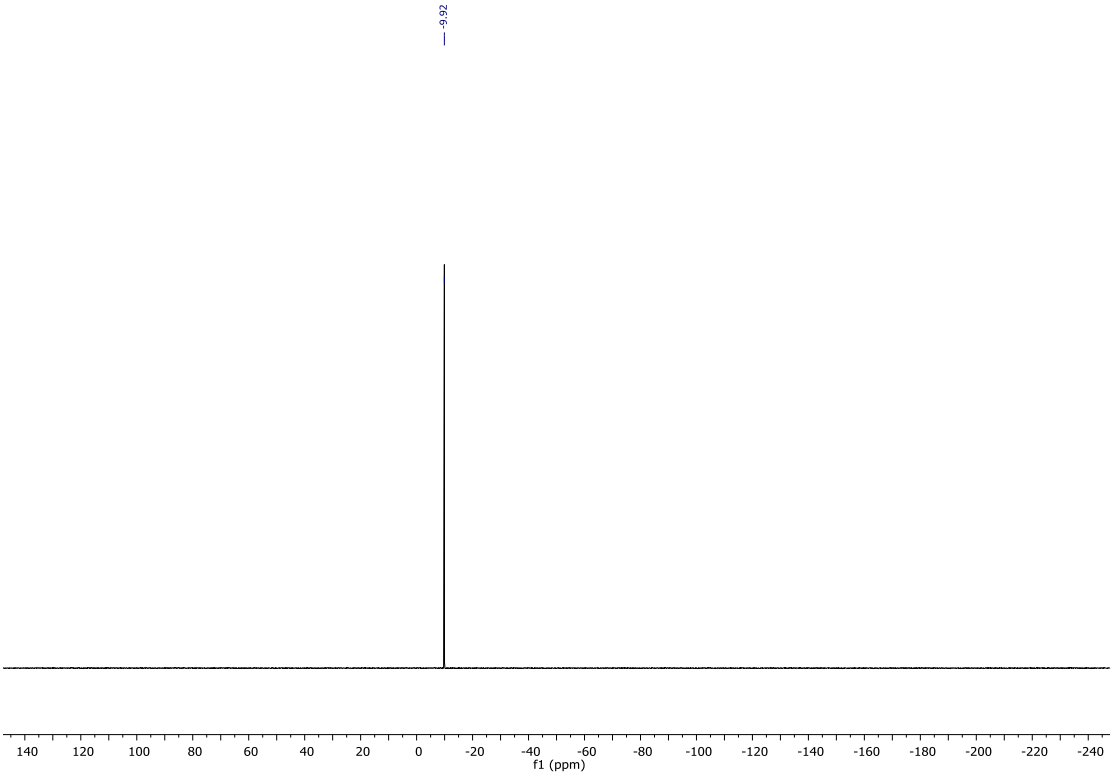
^{1}H NMR



^{13}C NMR

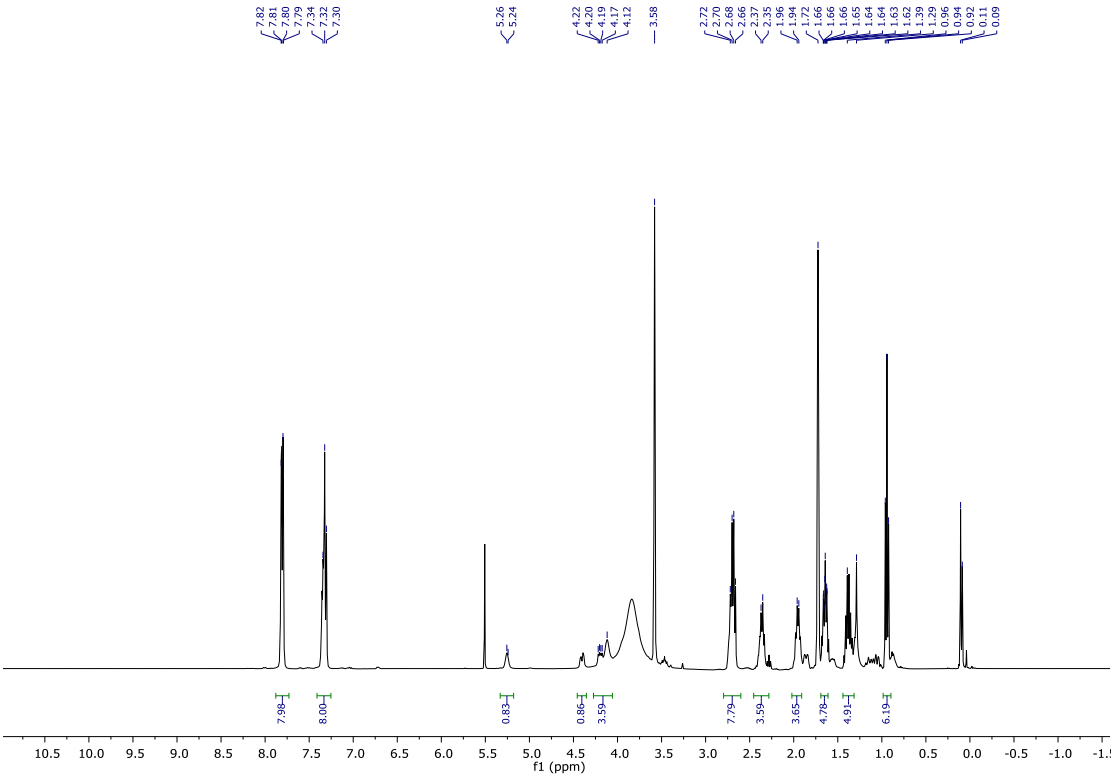


^{31}P NMR

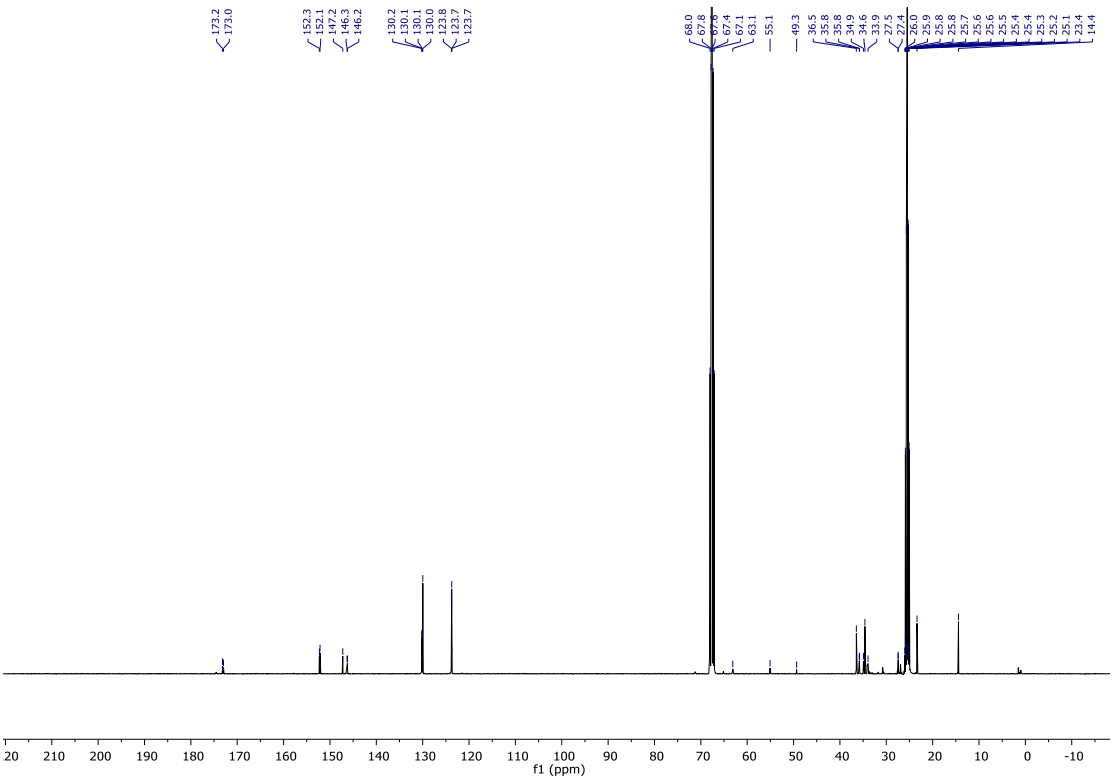

-9.92
140
120
100
80
60
40
20
0
-20
-40
-60
-80
-100
-120
-140
-160
-180
-200
-220
-240
f1 (ppm)

dAzoPA

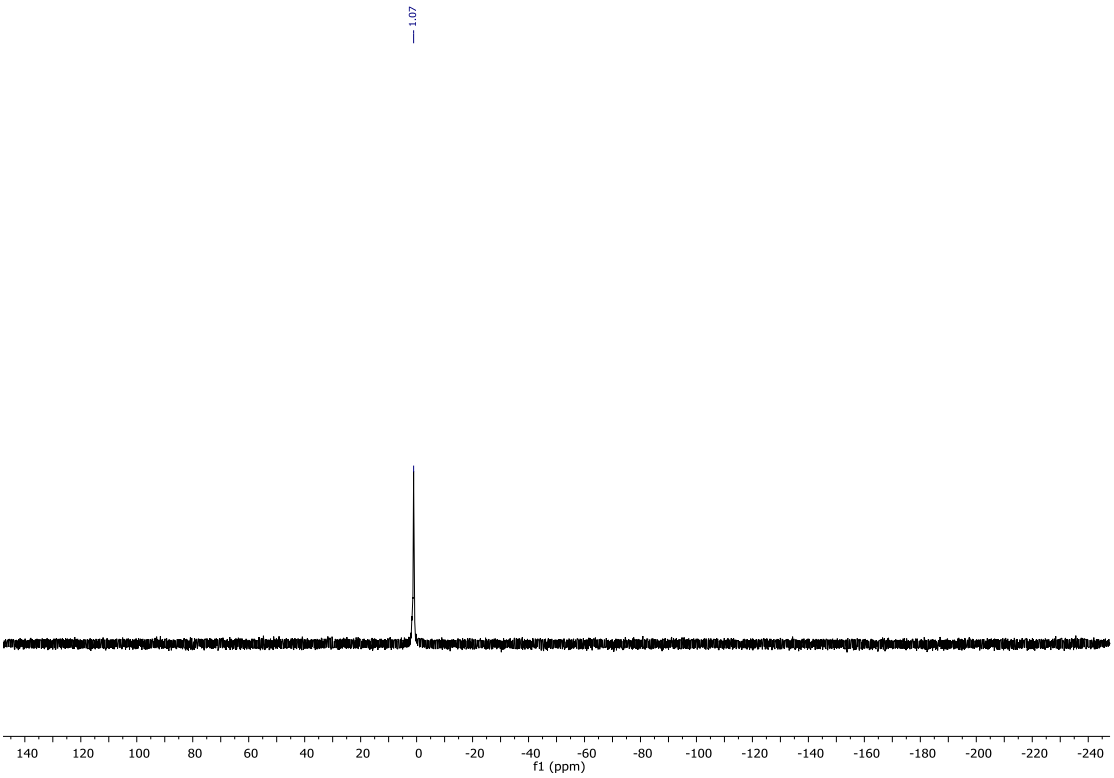
^{1}H NMR



^{13}C NMR



^{31}P NMR


1.07
140
120
100
80
60
40
20
0
-20
-40
-60
-80
-100
-120
-140
-160
-180
-200
-220
-240
f1 (ppm)

Python Scripts

Intensity_per_cell.py

```
from ij import IJ
from ij.plugin.frame import RoiManager
import os
import csv

def measure():
	file_path = u"C:/Users/Reika Tei/Box/Reika/Programming/Extract_nuclear_intensity/input"
	file_list = os.listdir(r'C:/Users/Reika Tei/Box/Reika/Programming/Extract_nuclear_intensity/input')
	output_path = u"C:/Users/Reika Tei/Box/Reika/Programming/Extract_nuclear_intensity/output"

	for file_name in file_list:
		root,ext = os.path.splitext(file_name)
		if ext == '.tif':
			image = IJ.openImage(file_path + "/" + file_name)
			image.show()

			## count cell numnber using particle analyzer ##
			IJ.run("Duplicate...", "duplicate")
			image_mask = IJ.getImage()
			IJ.run("Auto Threshold", "method=Yen white")
			IJ.run("Analyze Particles...", "size=100.00-1000.00 circularity=0.10-1.00 add slice")
			image_mask.close()

			## Count cell number as roi_number ##
			rm = RoiManager.getInstance()
			roi_number = rm.getIndexes()
			rm.reset()
			rm.close()
			image.setC(2)
			IJ.run("Measure")
			IJ.saveAs("Results", output_path + "/" + root + "_" + str(len(roi_number)) + ".txt")
			IJ.run("Clear Results", "")
			IJ.run("Close")
			IJ.run("Close")

def analysis():
```

```
	file_path = u"C:/Users/Reika Tei/Box/Reika/Programming/Extract_nuclear_intensity/output"
	file_list = os.listdir(r'C:/Users/Reika Tei/Box/Reika/Programming/Extract_nuclear_intensity/output')
	output_path = u"C:/Users/Reika Tei/Box/Reika/Programming/Extract_nuclear_intensity"

	pS6_dict = {}
	for file_name in file_list:
		root,ext = os.path.splitext(file_name)

		if ext == ".txt":
			cond = root.split(",")[0]
			cellcount = int(root.split("_")[1])
			pS6_list = []

			with open(file_path + "/" + file_name, 'r') as input_file:
				contents = input_file.readlines()
				for line in contents[1:]:
					Number, Area, Mean, IntDen, RawIntDen = line.split()
					pS6_int = float(IntDen)
					pS6_per_cell = pS6_int/cellcount
					pS6_list.append(pS6_per_cell)

			if cond not in pS6_dict:
				pS6_dict[cond] = pS6_list
			else:
				pS6_dict[cond].extend(pS6_list)

	output_file = open(output_path + "/analysis.csv", 'w')
	csvwriter = csv.writer(output_file, lineterminator = "\n")
	csvwriter.writerow(["Compound", "Average pS6 per cell"])

	for key in pS6_dict:
		csvwriter.writerow([key] + pS6_dict[key])

	output_file.close()

measure()
analysis()
```

Nuclear_intensity.py

```
from ij import IJ
from ij.plugin.frame import RoiManager
import os

def measure():
	file_path = u"C:/Users/Reika
Tei/Box/Reika/Programming/Extract_nuclear_intensity/input"
	file_list = os.listdir(r'C:/Users/Reika
Tei/Box/Reika/Programming/Extract_nuclear_intensity/input')
	output_path = u"C:/Users/Reika
Tei/Box/Reika/Programming/Extract_nuclear_intensity/output"

	for file_name in file_list:
		root,ext = os.path.splitext(file_name)
		if ext == '.tif':
			image = IJ.openImage(file_path + "/" + file_name)
			image.show()

			##	identify each nucleus using particle analyzer ##
			IJ.run("Duplicate...", "duplicate")
			image_mask = IJ.getImage()
			IJ.run("Auto Threshold", "method=Huang2 white")
			IJ.run("Analyze Particles...", "size=100.00-1000.00 circularity=0.10-
1.00 exclude add slice")
			image_mask.close()

			## measure the YAP intensity in each nucleus ##
			rm = RoiManager.getInstance()
			roi_number = rm.getIndexes()
			for i in roi_number:
				rm.setSelectedIndexes([i])
				image.setC(2)
				IJ.run("Measure")

			rm.reset()
			rm.close()
			IJ.run("Close")
			IJ.saveAs("Results", output_path + "/" + root + ".txt")
			IJ.run("Clear Results", "")

def analysis():
	file_path = u"C:/Users/Reika
Tei/Box/Reika/Programming/Extract_nuclear_intensity/output"
```

```
        file_list = os.listdir(r'C:/Users/Reika
Tei/Box/Reika/Programming/Extract_nuclear_intensity/output')
        output_path = u"C:/Users/Reika
Tei/Box/Reika/Programming/Extract_nuclear_intensity"

        YAP_dict = {}
        for file_name in file_list:
                root,ext = os.path.splitext(file_name)

                if ext == '.txt':
                        comp = root.split(",")[0]
                        YAP_list = []

                        with open(file_path + "/" + file_name, 'r') as input_file:
                                contents = input_file.readlines()
                                for line in contents[1:]:
                                        Number, Area, Mean, IntDen, RawIntDen = line.split()
                                        YAP_mean = float(Mean)
                                        YAP_list.append(YAP_mean)

                        if comp not in YAP_dict:
                                YAP_dict[comp] = YAP_list
                        else:
                                YAP_dict[comp].extend(YAP_list)

        output_file = open(output_path + "/analysis.csv", 'w')
        csvwriter = csv.writer(output_file, lineterminator = "\n")
        csvwriter.writerow(["Compound", "Nuclear YAP"])

        for key in YAP_dict:
                csvwriter.writerow(key.split("\t"))
                csvwriter.writerow(YAP_dict[key])

        output_file.close()

measure()
analysis()
```