

# **A line-broadening free real-time $^{31}\text{P}$ Pure Shift NMR method for phosphometabolomic analysis**

Karl Kristjan Kaup,<sup>a,b</sup> Lauri Toom,<sup>b</sup> Laura Truu,<sup>a</sup> Sten Miller,<sup>a</sup> Marju Puurand,<sup>a</sup> Kersti Tepp,<sup>a</sup> Tuuli Käämbre,<sup>a</sup> Indrek Reile<sup>a</sup>

## **Electronic Supplementary Information**

---

<sup>a</sup> National Institute of Chemical Physics and Biophysics, Akadeemia tee 23, Tallinn 12618, Estonia.

<sup>b</sup> Institute of Chemistry, University of Tartu, Ravila 14A, 50411 Tartu, Estonia.

## Contents

1. Reagents, chemicals and experimental conditions .....	3
2. Rat heart perfusion experiment for $^{18}\text{O}$ labelling of ATP .....	3
3. Method performance at different data chunk and gap lengths .....	4
4. SAPPHIRE implementation .....	6
5. Comparison of the presented method with other Pure Shift techniques .....	7
6. Application to a mixture of $^{18}\text{O}$ -labelled ATP and ADP .....	9
7. Instructions to setting up the experiment .....	9
8. Instructions for data processing .....	10
9. Pulse sequence .....	11
10. Data processing script .....	16
11. Additional au macros .....	21

## 1. Reagents, chemicals and experimental conditions

All standard substances (ATP, CTP, UTP) were bought from Sigma Aldrich. D<sub>2</sub>O and H<sub>2</sub><sup>18</sup>O were received as a donation from Cambridge Laboratories, Inc. Solutions were prepared gravimetrically whenever feasible. pH of all solutions was carefully adjusted before NMR measurements. NMR measurements were conducted either in good quality 5 mm NMR tubes or Shigemi tubes.

All NMR experiments were run on the 700 MHz Bruker Avance III spectrometer, located in the Institute of Chemistry of the University of Tartu. <sup>31</sup>P NMR was carried out with a BBO probe at 6 °C sample temperature. All samples were allowed to temperature equilibrate for 15 min in the probe before analysis. The probe was tuned and matched for every sample and the 90° pulse for <sup>31</sup>P was calibrated. The 90° pulse for <sup>31</sup>P was found to be between 14.7-14.9 μs for all samples. Soft pulses for active and passive spin refocusing were 4.080 ms long RSnob pulses, corresponding to approx. 450 Hz bandwidth. The soft pulse bandwidth has to be selected so that all targeted spins (i.e. active spins) experience a uniform rotation, while untargeted spins (i.e. passive spins) are not perturbed. This is an important consideration when choosing the passive spin refocusing pulse, since this pulse is applied for ~40 times. Any errors from this pulse would lead to imperfect refocusing during consecutive data chunks. Small imperfections can be alleviated by applying a supercycle to the passive spin refocusing pulse, but this will not resolve larger errors, which will eventually lead to faster FID decay and more artefacts.

The pulse program herein was developed and run in Bruker Topspin 3.5pl6 software. The authors expect the code to run on other versions of Topspin 3.x, however this has not been tested. The present implementation relies on Bruker Wavemaker macro for optimization of soft pulses. This macro is not included by default starting from Topspin 3.5pl6. However, soft pulses can be optimized by other means (i.e. Shapetool), if Wavemaker is not available.

The presented data processing macro was developed in Bruker Topspin 3.5pl6 and has also been tested in Topspin 3.6.2. Detailed instructions for applying the code are given below.

The authors are publishing the pulse program and macro codes without any warranties. We have successfully recorded the presented spectra with these programs, but we take no responsibility for potential adverse effects on other users' instrumentation when implementing this method.

## 2. Rat heart perfusion experiment for <sup>18</sup>O labelling of ATP

Wistar line male adult rats 300-400g were anaesthetized by intraperitoneal injection of ketamine (75 mg/kg) and dexmedetomidin (1 mg/kg); the blood was protected against coagulation by injection of heparin. The heart was quickly excised preserving a part of aorta and placed in an aerated perfusion solution, modified Krebs medium, with the following composition: 118 mM NaCl, 5.3 mM KCl, 1.2 mM MgSO<sub>4</sub>, 0.5 mM EDTA, 25 mM NaHCO<sub>3</sub>, 11 mM Glucose, and 2 mM CaCl<sub>2</sub> (pH 7.4). The heart was cannulated via aorta (Langendorf system) and perfused with the solution with a flow rate of 15 ml/min during 5 minutes. Then the heart was perfused for 10 minutes with the same solution where 30% of the H<sub>2</sub><sup>16</sup>O was replaced with H<sub>2</sub><sup>18</sup>O. The perfusate was continuously aerated and its temperature was kept at 37°C. After that the heart was quickly frozen with liquid nitrogen (freeze clamp method) and the heart muscle was used for experiments.

Rats were housed under standard laboratory conditions (at 22°C constant temperature and a 12:12 h light/dark cycle with free access to food and water). Animal experiments were approved by the Estonian National Board of Animal Experiments in accordance with the European Community Directive (86/609/EEC).

#### NMR Sample preparation from freeze clamped heart muscle

The heart tissue was ground in liquid nitrogen using a mortar and pestle. Then weighed and subjected to an extraction solution: 0.6 M HClO<sub>4</sub>, 1 mM EDTA (1 mL extraction solution per 100 mg tissue), stirred vigorously and kept on ice for 5 min. The mixture was neutralized (pH 7.2) with 2 M KH<sub>2</sub>CO<sub>3</sub> and centrifuged at 2253 x g for 10 min, at +4 °C. The supernatant was removed and treated with Chelex® 100 sodium form with constant stirring at +4 °C, overnight. The extract was centrifuged for 5 min 2253 x g at +4 °C the supernatant was removed, frozen in liquid nitrogen and lyophilized. The dry residue was taken up in 300 µL of D<sub>2</sub>O, pH adjusted to 9.5 with NaOH aq and loaded into a Shigemi NMR tube.

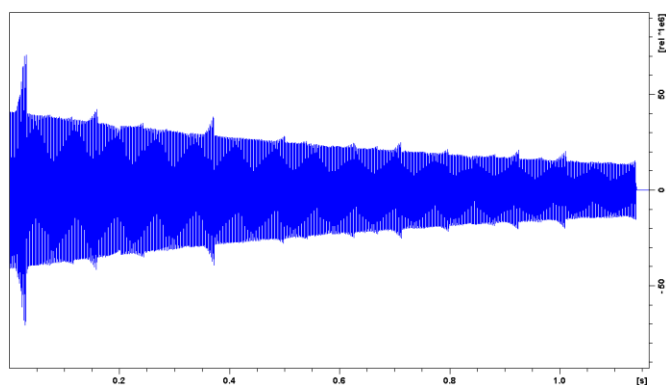
#### Purification of <sup>18</sup>O labelled ATP from the perfused heart sample by preparative LC

ATP was isolated from the primary NMR sample by LC (GE Healthcare ÄKTAPrime Plus), using a Mono Q HR 5/5 ion-exchange column (Pharmacia Biotech) with triethylammonium bicarbonate (TEAB) buffer pH 8.8 (gradient from 0-85%) at a 0.4 ml min<sup>-1</sup> flow-rate. Analytes were detected with a UV detector fixed at 280 nm. ATP containing fractions were combined, lyophilized, reconstituted in 300 µL of D<sub>2</sub>O, pH adjusted to 9.5 with NaOH aq and loaded into a Shigemi NMR tube.

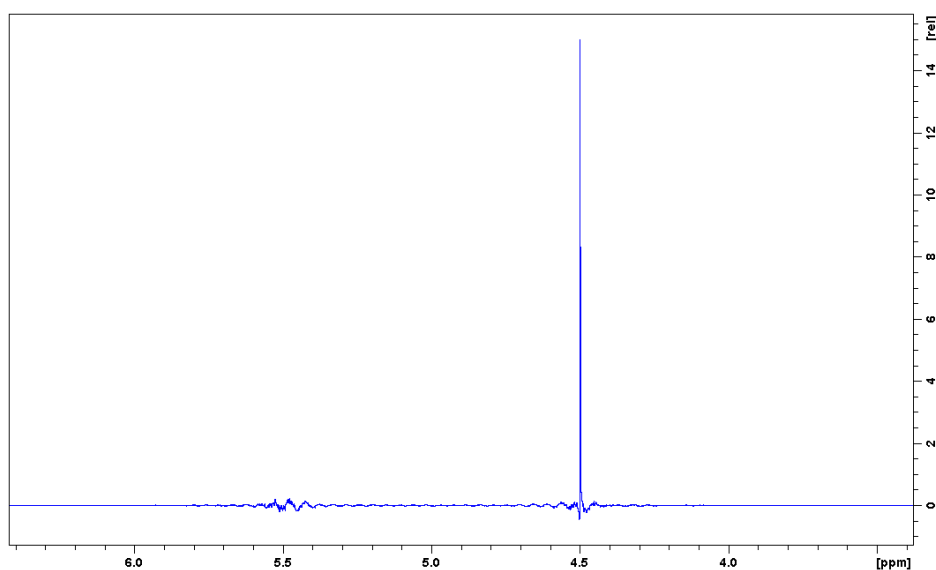
### 3. Method performance at different data chunk and gap lengths

The method herein works by extrapolating the information from within the limited duration data chunks into the gaps in FID by linear extrapolation (LP). A certain number of datapoints has to be predicted, based on the limited number of data points available. In order to gauge, how many points can be reliably predicted from our usual gap length of 20-25 ms, the macro was tested on a simulated FID of a single signal. Chunk length was fixed at 21,28 ms, representing 600 points and the gap length was varied. It followed that if the gap is smaller than approximately a quarter of the chunk length, then LP does not induce additional artefacts. In other words, based on a ~20 ms (600 points) chunk, about 5 ms of data (150 points) can be predicted. In the actual experiment we are only predicting half of the gap in either direction, which relieves the condition further during most of the chunked FID. However, the 1<sup>st</sup> chunk is shorter than the later ones, posing the most critical boundary condition for setting up the experiment.

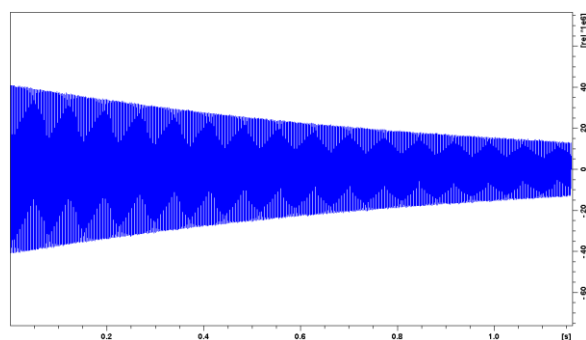
If too many points are predicted, noticeable errors from the expected FID evolution occur (Fig. S1 displays an extreme case). Fourier transform of such FID introduces artefacts (Fig. S2) – similar artefacts occur in real spectra if too many points are being predicted. In turn, if a reasonable number of points are being predicted, good quality FID can be obtained (Fig. S3), which yields a high quality spectrum after FT.



**Figure S1.** Processed FID with gap length equal to chunk length at 600 points.



**Figure S2.** Spectrum from Fourier Transform of FID from S1.



**Figure S3.** Processed FID with gap length equal to a quarter of chunk length (150 points).

The considerations for choosing the length of data chunks are similar to any other PS methods: it is limited by the fact that the slight amount of J-development during the chunk must stay negligible. The difference from most other PS techniques is that on top of J-evolution during data chunks, we need to consider evolution during the gaps: J-evolution is refocused in the middle of chunks, but then evolution will be further elongated by LP until the mid-point of each gap. The chunk cannot be too short, because linear prediction fares better there are more points to extrapolate upon. Linear prediction will also fare better if the gap is shorter (i.e. the ratio of chunk to gap length is bigger). The length of the gap should be as short as possible, but in practice, it is determined mainly by the

length of the soft refocusing pulse that has to uniformly rotate the targeted spins and leave others unperturbed.

The method can handle data with different numbers or signals (frequencies) in the FID-s. As long as the number of extrapolated frequency components, set by the parameter NCOEF in TopSpin, is sufficiently larger than the number of expected signals, its value does not influence spectral quality. Consequently, the data processing macro sets NCOEF=1000, which should also handle well significantly more complex spectra than what is presented herein.

## 4. SAPPHIRE implementation

Chunking sidebands are relieved by using the SAPPHIRE method.<sup>1</sup> The simplest first order SAPPHIRE implementation would be to shift the phase of J-modulation by half of the periodic unit. As the original experiment starts with J-modulation at its maximum value, then the second experiment is manipulated to start at minima. The length of the first chunk is also changed (in the first order case, doubled) accordingly. This is achieved by using multiple gradient spin echo blocks to refocus the chemical shift of active spins while manipulating the J-evolution as needed. These two different experiments cause chunking sidebands with opposite phases, cancelling out at summation (Fig. 4, main text). First order implementation of SAPPHIRE cancels out the first order (highest intensity) sidebands.

The logic behind calculating the delays  $\tau_2$  (*d18* in pulse program),  $\tau_3$  (*d47* in pulse program) and  $\tau_4$  (*d19* in pulse program) is similar to and directly derived from the original SAPPHIRE publication by the Morris group<sup>1</sup>:

$$\begin{aligned}\tau_2: d18 &= \text{larger}(d62/4 - \text{cnst47} * d62/2, 0) \\ \tau_3: d47 &= d62/4 - \text{abs}(d18 - d19) \\ \tau_4: d19 &= \text{larger}(-d62/4 + \text{cnst47} * d62/2, 0)\end{aligned}$$

where *d62* is the time duration of data chunks and *cnst47* sets the SAPPHIRE order, which takes the values of 0, 0.25, 0.5 and 0.75 in present work. SAPPHIRE order is determined by the number of experiments with opposite J-modulation phase. The first order case therefore consists of two experiments (*cnst47* values 0 and 0.5), with first experiment starting with maximum J-modulation phase and second starting at the minimum of J-modulation, cancelling out first order artefacts. Higher SAPPHIRE orders double on the number of experiments from the previous order (1st order consists of 2 experiments, 2nd order of 4 – *cnst47* values 0, 0.25, 0.5, 0.75 - and so on). Higher order artefacts need to be cancelled from all experiments of the previous order and to keep the first order artefacts cancelled, it is necessary for the added experiments to also cancel the first order artefacts within themselves.

We have introduced a slight modification to the SAPPHIRE logic by adding an extra delay  $\tau_1$  in the first PFGSE before acquisition:

$$\tau_1: d31 = (7/8 - \text{cnst47}) * d56/2,$$

where *d56* is the time duration of FID gaps. In our approach, the J-evolution maxima are not at the ends of the chunks (as in the original SAPPHIRE) but rather at the center of the gaps. This extra delay serves to account for the additional gap length - in the original SAPPHIRE sequence, the gap is

neglected and therefore not necessary to be accounted for. However, herein J-evolution will be extrapolated further until the mid-points of the gaps.

Technically our method allows to use SAPPHIRE up to third order, but as the length of the first chunk goes smaller with each higher order, third order implementation requires the use of experiments with very short first chunk lengths which can cause rolling artefacts in the spectra (Similar to Fig. S2) and incorrect cancellation of chunking sidebands. That being said, in certain cases with a very short gap length, it might be useful to extend SAPPHIRE to third order. We have, however, limited us to the 2<sup>nd</sup> order in results that are presented in main text.

## 5. Comparison of the presented method with other Pure Shift techniques

A comparison is presented in Table S1 and graphically in Figure S4 below, followed by a brief description of observations. All pure shift spectra were acquired with identical data chunk lengths (20 ms) and acquisition times, as described in the experimental section of main text. All experiments in Figure S4 and Table S1 were acquired in one measurement session, with no adjustments to instrumental settings, once the experiment sequence was started. All FID-s were acquired in 16k points and zero filled to 32k points before Fourier transformation without any window functions applied.

Table S1. Quantitative comparison of spectra from Figure S4.

Pos in Fig S4	Pure Shift Technique	Line width, Hz	Signal intensity factor	Artefact intensity	Acquisition time
a	Regular 1D	2,0	1,0	-	14 min
b	Real time LP	2,0	1,2	1,8%	14 min
c	Interferogram	1,9	1,4	-7,4%	9 h, 10 min
d	Real time	4,1	0,9	9,5%	14 min

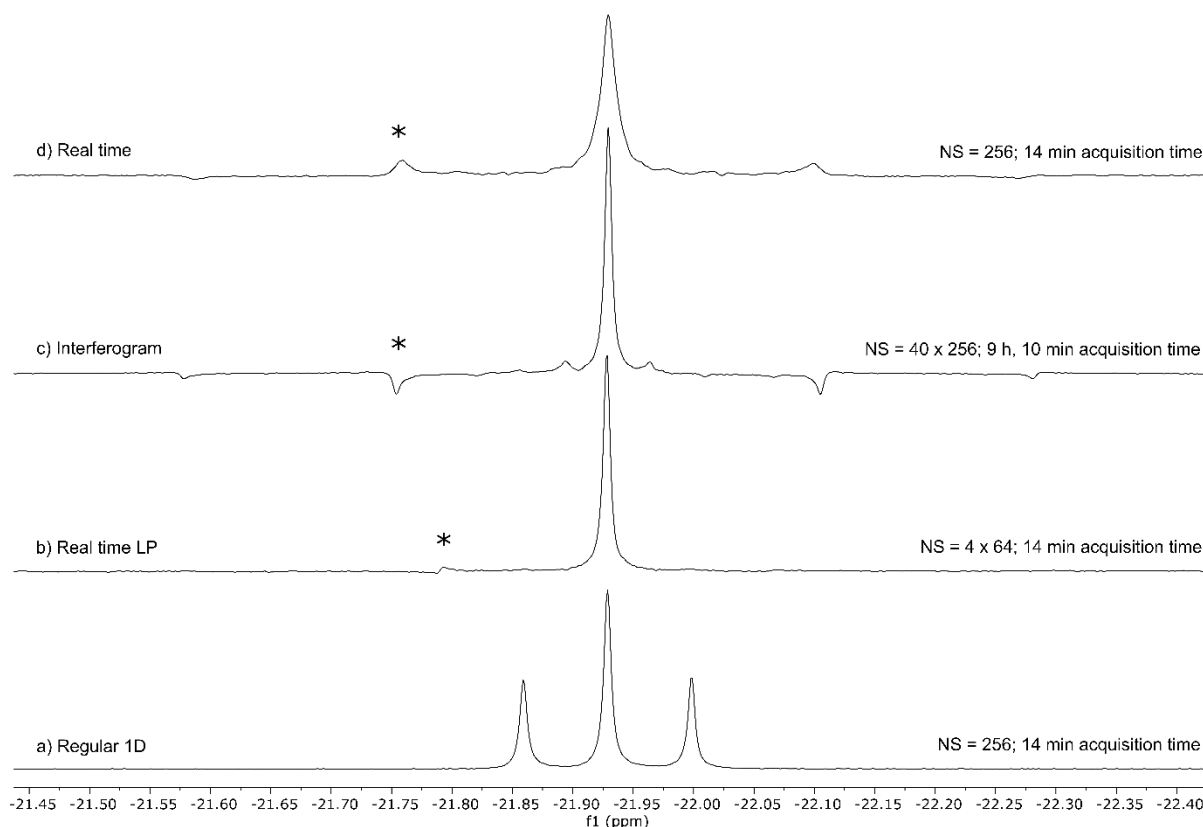


Figure S4. Comparison of regular 1D (trace a)  $^{31}\text{P}$  spectrum of ATP  $\beta$ -phosphate with Pure Shift spectra required with the method presented in this work (trace b), with a pseudo-2D interferogram acquisition scheme (trace c) and traditional real-time acquisition (trace d).

Methods a, b and c yield near-identical line shapes, the apparently narrower line shape from trace c) in Table S1 is the result of data rounding. As expected, the traditional real-time acquisition scheme introduces line broadening, whereas interferogram acquisition introduces a large increase in measurement time. The method herein yields good line shape and artefact suppression, while its signal intensity falls short of the interferogram acquisition scheme. However, we argue that this penalty is compensated by the difference in measurement time.

Note that although recorded data chunk durations were equal for traces b), c) and d) in Figure S4, trace b) displays artefacts at a different distance from main signal. This is due to the fact that the LP procedure effectively increases chunk duration: the period of the artefact causing FID modulation will be the sum of chunk and gap durations (see main text Figure 4).

The first sub-experiment of the semi-real time pure shift experiment (main text Ref. 23) is identical to the interferogram acquisition scheme in its pulse sequence, until acquisition starts. Consequently, the line width and signal intensity properties of the semi-real time experiment are expected to be similar to interferogram acquisition. Since neither include any artefact suppression techniques, we expect also the artefact intensities to be similar. The semi-real time approach (main text Ref. 23) has a sensitivity benefit, compared with the method presented in this work, but this benefit is not sufficient to compensate for the two-fold increase in measurement time for the ATP signals that we analyze in this work.



The SAPPHIRE (main text Ref. 20) acquisition scheme was originally demonstrated for interferogram acquisition, where it yielded the most artefact free pure shift spectra up to date. Up until acquisition starts, the pulse sequences for interferogram SAPPHIRE and the work herein are similar. Consequently, we expect the intensity performance to be similar to our work and acquisition time to be the same as for pseudo-2D interferogram acquisition. The artefact suppression performance of pseudo-2D-SAPPHIRE is likely to be superior compared to the method presenter in this work: repeated refocusing pulses during acquisition and linear prediction are both sources of additional errors that reduce artefact suppression performance.

## 6. Application to a mixture of $^{18}\text{O}$ -labelled ATP and ADP

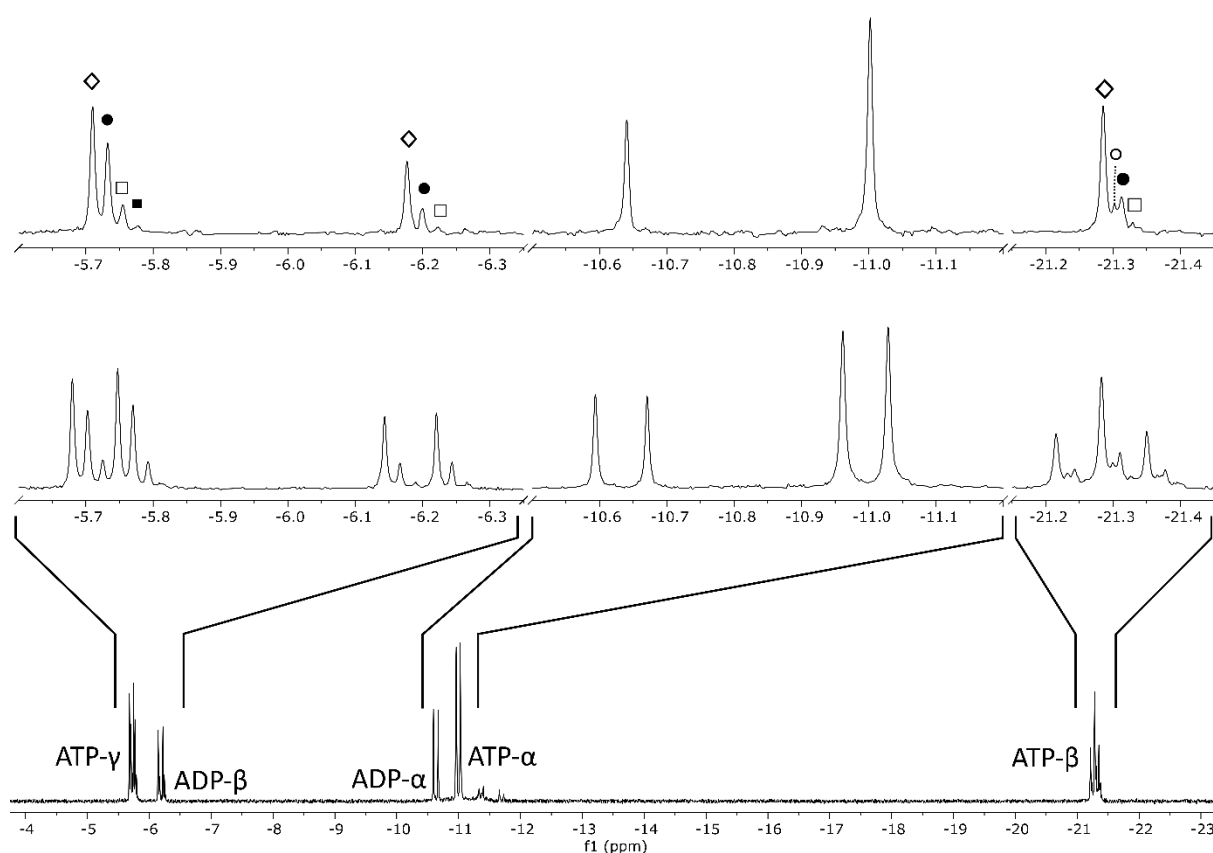


Figure S5. Application of the method for a mixture of ATP and ADP, acquired with 2048 scan (4x512 scans for SAPPHIRE experiments in Pure Shift (upper trace). Note that herein, when measuring ATP- $\gamma$ , both the ADP- $\alpha$  and ATP- $\beta$  regions need to be treated as passive spins, excluding the option to acquire PS spectra for ATP- $\alpha$  and ATP- $\gamma$  simultaneously (I.e., main text, Fig. 2).

## 7. Instructions to setting up the experiment

The experiment is best set up based on an underlying 1D  $^{31}\text{P}$  spectrum. Acquisition time, spectral width, number of acquired points can be directly derived from the underlying regular 1D experiment. Experiment specific considerations are:

- a) The number of acquired scans should be divided into four distinct datasets: the overall number of scans will remain the same, but they will be acquired in 4 distinct 1D experiments, and the FID will be added together after data processing.
- b) These four experiments will differ by the value of parameter *cnst47*, which should take values 0, 0.25, 0.5 and 0.75 in the four experiments. All other acquisition parameters should be kept the same.
- c) The FID chunk length is chosen by the operator with setting the value for *cnst20*. This parameter should be entered as the number of complex points that will make up the chunk. The spectrometer will calculate the time duration of the chunk and display it as *d62*. In present work, we used *cnst20* = 400, corresponding to chunk length of about 20 ms. We recommend using even values for *cnst20*.
- d) The desired bandwidth for active spins refocusing pulse (in ppm, usually an RSnob pulse) is entered as *cnst31*, We used 1.6 ppm bandwidth for all soft pulses.
- e) Set the offset of the active spins refocusing pulse as *cnst32* (as offset from the carrier frequency, i.e. distance from *O1P*, in ppm).
- f) If multiple active spins are targeted at once (i.e. when measuring  $\alpha$  and  $\gamma$  phosphates at once), enter the bandwidth for the 2<sup>nd</sup> excitation region as *cnst33*. Otherwise set *cnst33* = 0.
- g) If multiple active spins are targeted, enter the offset of the 2<sup>nd</sup> excitation region as *cnst34* (as offset from the carrier frequency, i.e. distance from *O1P*, in ppm).
- h) The desired bandwidth for passive spins refocusing pulse (in ppm, usually an RSnob pulse) is entered as *cnst41*, We used 1.6 ppm bandwidth for all soft pulses.
- i) Set the offset of the active spins refocusing pulse as *cnst42* (as offset from the carrier frequency, i.e. distance from *O1P*, in ppm).
- j) If multiple passive spins are targeted at once (i.e. when measuring  $\beta$  phosphate), enter the bandwidth for the 2<sup>nd</sup> excitation range as *cnst43*. Otherwise set *cnst43* = 0.
- k) If multiple active spins are targeted, enter the offset of the 2<sup>nd</sup> excitation region as *cnst44* (as offset from the carrier frequency, i.e. distance from *O1P*, in ppm).
- l) Generate the soft pulses with '*wvm -a*' command and execute the experiment.

## 8. Instructions for data processing

Data processing macro for LP of data into FID gaps, as described in main text, is given below, in SI section 9. We recommend the following workflow:

- a) All four FID-s of the same experiment, corresponding to SAPPHERE orders 0, 0.25, 0.5 and 0.75 should be processed separately.
- b) Open a dataset with a recorded chunked FID. The macro will not alter or harm the acquired data, but it will overwrite the embedded processing parameters in the dataset. This should not be a problem since direct processing of the FID is unlikely to give meaningful results. However, we recommend taking note of the original processing parameters or making a backup copy of the dataset before proceeding.
- c) Execute the macro by calling the name of the Python script in the command line. If no extrapolated data already exists (common when executing the macro for the first time for a particular dataset), a folder missing error is displayed, which can be dismissed. The macro will cycle through several temporary datasets during operation. We recommend not interacting yourself with Topspin before the script has finished. If running the script on a

spectrometer installation, we recommend not having an active acquisition going on in the background. The script generates a copy of the dataset in a new folder with the same dataset name, with ‘\_ekstrap’ added to the end. The expno will remain the same. This way the original acquired data will remain untouched. FT with generic parameters will be performed on the processed dataset, once the script is finished.

- d) The resulting continuous FID can be processed by usual means. However, they represent single SAPPHIRE experiments with chunking artefacts.
- e) In order to cancel artefacts, all four processed FID-s need to be added. The easiest way to do this is by using the *addfid* command of Topspin (consult Topspin manuals, FID-s to be added can be defined with *edc2* command).
- f) The resulting FID should be processed as a regular 1D experiment. We used zero filling to 32k points and no window functions. Final processing can also be performed with MestreNova, but in this case the FID needs to be circular shifted by the group delay before FT in Mestrenova, otherwise extreme frequency dependent phase errors will be displayed in spectrum. Enter the value of Bruker parameter *GRPDLY* into the requested circular shift field in the *FT shift* popup window in MestreNova. This is a byproduct of the way the script is implemented: the script removes the Bruker Digital filter group delay from the beginning of the FID. These datapoints contain no meaningful data and, in principle, can be safely removed. However, this causes a mismatch with the way that the group delay is handled in MestreNova, requiring circular shifting the FID.

## 9. Pulse sequence

The following code should be copied into a file in the Topspin installation directory, subdirectory /exp/stan/nmr/lists/pp/user/

---

```
;IR_1D_PS_LP
;01/06/2021
;
;Avance III Version
;Topspin3.x
;
;$CLASS=HighRes
;$DIM=1D
;$TYPE=PS
;$SUBTYPE=
;$COMMENT=
;
;National Institute of Chemical Physics and Biophysics, Tallinn, Estonia
;
;1D band-selective Pure Shift experiment that produces a discontinuous FID
;that should be processed with the attached linear prediction (LP) script.
;
;Code developed and tested on TopSpin3.5pl6 running on a Bruker Avance III
;spectrometer. The authors make no promises about compatibility with different
;TopSpin versions. Code contains WaveMaker definitions and soft pulses can be
```

```

;generated with WaveMaker. Generating soft pulses by other means should work too.
;
;The authors take no responsibility for any unforeseeable adverse effects on
;spectrometer hardware that may occur while experimenting with this code or its
;modifications.
;
;This code and related work by the authors is based on the following publications:
;(1) P. Kiraly, M. Nilsson and G. A. Morris, J. Magn. Reson., 2018, 293, 19–27.
;(2) P. Moutzouri, Y. Chen, M. Foroozandeh, P. Kiraly, A. R. Phillips, S. R.
; Coombes, M. Nilsson and G. A. Morris, Chem. Commun., 2017, 53, 10188–10191.
;
;The code is published as part of the following publication:
; K. K. Kaup, L. Toom, L. Truu, S. Miller, M. Puurand, K. Tepp, T. Kaambre,
; I. Reile, “A line-broadening free real-time 31P Pure Shift NMR method for
; phosphometabolomic analysis”, Analyst, 2021.

```

```

#include <Avance.incl>
#include <Grad.incl>
#include <Delay.incl>
#include <De.incl>

```

```

dwellmode explicit

```

```

"p17=300u"

```

```

"d17=100u"

```

```

"p2=2*p1"

```

```

"l20=cnst20%8"

```

```

if (l20 == 0)
{
    "cnst35 = cnst20"; [s]
}

```

```

if (l20 != 0)
{
    "cnst35 = cnst20 - l20 + 8"; [s]
}

```

```

;Length of refocusing block (gap)

```

```

"d54=2*p17+2*d17+p22"; [s]

```

```

;Temporal length of the refocusing block's full points

```

```

"cnst54=trunc(d54*1000000,dw)/1000000";[s]

```

```

;Number of full points in the refocusing block

```

```

"cnst21=cnst54*1000000/dw"; [points]

```

```

"l21=(cnst21)%16"

```

```

;Compensation delay to make the block/gap fit into an integer number of

```

```

;points, to be used before and after acquire-refocus block.
;Forces d56 to be even number.
if (l21 == 0)
{
    "d55 = 0"; [s]
}

if (l21 != 0)
{
    "cnst22 = cnst21 - l21"
    "d55 = ( (cnst22 + 16)*dw - d54)/2 "; [s]
}

;Actual length of "blank" space between data chunks
"d56=d54+2*d55"; [s]

;Length of FID data chunks to be recorded; to be defined in points
"d62=cnst35*dw"; [s]

;Length of the first data chunk
"d61=cnst35*dw/2"; [s]

;delays for SAPPHIRE
"d18 = larger(d62/4 - cnst47*d62/2,0)"
"d19 = larger(- d62/4 + cnst47*d62/2,0)"
"d31 = (7/8-cnst47)*d56/2"
"d47 = d62/4 - abs(d18-d19)"
"d63=d62 + (7/8-cnst47)*d56 - cnst47*d62"; [s]

;Number of FID data chunks to be recorded
"l0=aq/(d54+2*d55+d62)"

;Number of loops to be completed in pulse program
"l1=l0-1"

"acqt0=0"
baseopt_echo

1 ze
30m pl12:f2
2 30m do:f2 rpp4
50u cpd2:f2 BLKGRAD
d1
20u pl1:f1
p1 ph1
;First PFGSE
d31
d18 UNBLKGRAD
p16:gp1
d16

```

```

p2 ph2
p16:gp1
d16
d18
d31
;Second PFGSE, selects the active spins
d47
p16:gp2
d16 pl0:f1
(p12:sp12 ph3)
p16:gp2
d16 pl1:f1
d47
;Third PFGSE
d19
p16:gp3
d16
p2 ph5
p16:gp3
d16 pl0:f1 BLKGRAD
d19

; Real time acquisition
ACQ_START(ph30,ph31)

;Acquire first chunk (half the duration of following chunks)
0.1u REC_UNBLK
0.05u DWL_CLK_ON
d63
;First passive spin refocusing block
d55 REC_BLK
p17:gp4
d17 pl0:f1
(p22:sp22 ph4+ph10)
p17:gp4*-1.0
d17 ipp10
d55
;Looping acquisition of full chunks, followed by passive spin refocusing
3 d62 REC_UNBLK
d55 REC_BLK
p17:gp4
d17
(p22:sp22 ph4+ph10)
p17:gp4*-1.0
d17 ipp10
d55
lo to 3 times l1

0.1u REC_UNBLK
d61
d62

```

0.1u REC\_BLK  
DWL\_CLK\_OFF

rcyc=2  
30m do:f2 pl12:f2 mc #0 to 2 F0(zd)

exit

ph1=0 0 0 0 1 1 1 1  
ph2=0 0 1 1  
ph3=0 1  
ph4=0 0 1 1 0 0 1 1  
ph10= (24) 0 7 20 17 20 7 0 0 7 20 17 20 7 0 12 19 32 29 32 19 12 12 19 32 29 32 19 12  
ph5=0  
ph29=0  
ph30=0  
ph31=0 2 2 0 3 1 1 3

;WAVEMAKER DEFINITIONS

;sp12:wvm: rsnob(cnst31 ppm, cnst32 ppm) rsnob(cnst33 ppm, cnst34 ppm) ofs = 0.0 ppm np=2000  
;sp22:wvm: rsnob(cnst41 ppm, cnst42 ppm) rsnob(cnst43 ppm, cnst44 ppm) ofs = 0.0 ppm np=2000

;POWER LEVELS

;pl0 : zero power [O W]  
;pl1 : power level for 31P hard pulse  
;pl12 : f2 channel - power level for 1H decoupling

;PULSES

;p1 : hard 90 deg pulse (high power)  
;p2 : hard 180 deg pulse  
;p12 : 180 deg soft pulse for active spins (low power)  
;p22 : 180 deg soft pulse for passive spins (low power)  
;p16 : homospoil/gradient pulses during initial pulse sequence [1000 us]  
;p17 : homospoil/gradient pulses during real time acquisition [300 us]

;DELAYS

;d1 : relaxation delay; 1-5 \* T1  
;d62 : length of full chunk : = aq/I0 [< 20-25 msec]  
;d63 : length of first/half chunk : = d62/2  
;d16 : recovery delay for gradients during initial pulse sequence [300us]  
;d17 : recovery delay for bipolar gradient pulses during real time acquisition [100us]  
;d18 : SAPPHIRE delay tau\_2  
;d19 : SAPPHIRE delay tau\_4  
;d31 : delay tau\_1  
;d47 : SAPPHIRE delay tau\_3

;SHAPED PULSES

;sp12 shaped pulse power level for 180 selective pulse on active spins  
;sp22 shaped pulse power level for 180 selective pulse on passive spins  
;spnam12: shaped pulse for selective refocusing of active spins [RSnob]  
;spnam22: shaped pulse for selective refocusing of passive spins [RSnob]

```

;GRADIENTS
;gpz1: gradient for the initial hard 180 pulse
;gpz2: gradient for 180deg selective pulse on active spins
;gpz3: gradient for the second hard 180 pulse
;gpz4: gradient for selective passive spin refocusing [8%]
;gpnam1: SINE.100
;gpnam2: SINE.100
;gpnam3: SINE.100
;gpnam4: SINE.100

;OTHERS
;NS: 1 * n
;DS: 2
;l0 : number of chunks acquired during acquisition time
;cnst20 : length of d62 (full chunk) in points
;cnst47 : SAPPHIRE order constant [values < 1, usually 0; 0.25; 0.5; 0.75]
;DIGMOD: baseopt
;cnst31 : bandwidth for active spins refocussing soft pulse, in ppm
;cnst33 : bandwidth for second active spins refocussing soft pulse, in ppm
;cnst41 : bandwidth for passive spins refocussing soft pulse, in ppm
;cnst43 : bandwidth for second passive spins refocussing soft pulse, in ppm
;cnst32 : offset for active spins refocussing soft pulse, in ppm from carrier frequency
;cnst34 : offset for second active spins refocussing soft pulse, in ppm from carrier frequency
;cnst42 : offset for passive spins refocussing soft pulse, in ppm from carrier frequency
;cnst44 : offset for second passive spins refocussing soft pulse, in ppm from carrier frequency

```

---

## 10. Data processing script

The following text should be copied into a file (i.e. PS\_LP.py) in Topspin installation directory, subdirectory /exp/stan/nmr/py/user/

The script can be run from Topspin command line by calling its name (i.e. PS\_LP.py).

Below are also two additional AU macros (*PS\_del\_tempdata* and *PS\_del\_existdata*; see Section 11 below) that are used to delete existing extrapolated data for correct calculation. These AU macros should be copied into two files with the aforementioned names in Topspin installation directory, subdirectory /exp/stan/nmr/au/src/user/.

---

```

"""

```

```

Data processing python script for extrapolating information from
chunked Pure Shift FID chunks into the preceding and following
gaps by linear prediction.

```

```

Code originally published as
K. K. Kaup, L. Toom, L. Truu, S. Miller, M. Puurand, K. Tepp,

```



T. Kaambre, I. Reile, "A line-broadening free real-time 31P Pure Shift NMR method for phosphometabolomic analysis", Analyst, 2021.

This code should be copied to a file (i.e. PS\_LP.py) in Topspin installation directory, subdirectory /exp/stan/nmr/py/user/

Script can be run from Topspin command line by calling its name (i.e. PS\_LP.py).

```
"""
```

```
import time
```

```
"""
```

```
Following shifts the group delay
```

```
"""
```

```
XCMD("xau PS_del_existdata")  
time.sleep(3)
```

```
#Get the necessary parameters for group delay eradication  
grpdly = 2*float(GETPARSTAT("GRPDLY"))  
dataset0 = CURDATA()  
td_base = int(GETPARSTAT("TD"))
```

```
#Get necessary parameters for interpolation later on  
#Get parameters to find the lengths of chunks and break between  
chunk  
DW = float(GETPARSTAT("DW"))/1000000
```

```
CNST = GETPARSTAT("CNST")  
CNST = CNST.split()  
CNST35 = float(CNST[35])
```

```
DELAYS = GETPARSTAT("D")  
DELAYS = DELAYS.split()  
d54 = float(DELAYS[54])  
d55 = float(DELAYS[55])  
d56 = float(DELAYS[56])  
d63 = float(DELAYS[63])  
d62 = float(DELAYS[62])
```

```
#Preparation for TRF to give FID as processed data for further  
processing  
PUTPAR("FT_mod", "no")  
PUTPAR("WDW", "no")  
PUTPAR("BC_mod", "no")  
PUTPAR("ME_mod", "no")  
PUTPAR("PH_mod", "no")
```

```

#Copying the dataset for later
WR(["temp_grpdly", dataset0[1], "733435" , dataset0[3]])
WR(["temp_grpdly", dataset0[1], "1337" , dataset0[3]])

#Cutting the group delay out and circular shifting it into the end

rec_corr = 8

PUTPAR("TDeff", str(grpdly + rec_corr))
shift_grpdly = float(td_base) - float(grpdly)
PUTPAR("SI", str(td_base/2))
TRF()
PUTPAR("NSP", str(grpdly))
XCMD("LS")
#Read the data as a list for processing
grpdly_cs = GETPROCdata(-500,500)
grpdly_csi = GETPROCdata(-500,500,type = dataconst.PROCdata_IMAG )

#Open one of the copies to get the original data. Then we shift it
so that the FID starts without the group delay.
RE(["temp_grpdly", dataset0[1], "733435" , dataset0[3]])
PUTPAR("TDeff", "0")
PUTPAR("SI", str(td_base/2))
TRF()
PUTPAR("NSP", str(grpdly))
XCMD("LS")
data_cs = GETPROCdata(-500,500)
data_csi = GETPROCdata(-500,500,type = dataconst.PROCdata_IMAG )

#Open another file where we store our new group-delay-less FID.
RE(["temp_grpdly", dataset0[1], "1337" , dataset0[3]])
SAVE_ARRAY_AS_1R1I(data_cs, data_csi)

#Copy the FID from processed data into another new dataset's raw
data
GENFID("88282832")
time.sleep(1)
RE(["temp_grpdly", "88282832", "1" , dataset0[3]])

"""

Following makes the chunked FID into separate chunks, forward
predicts them and puts them together.

"""

dataset = CURdata()

```

```

WR(["temp_FIDadd", dataset[1], "1" , dataset[3]])
RE(["temp_FIDadd", dataset[1], "1" , dataset[3]])

TD = GETPARSTAT("TD")
TD = int(TD)

#Set parameters for trf
PUTPAR("PKNL", "FALSE")
PUTPAR("FT_mod", "no")
PUTPAR("WDW", "no")
PUTPAR("BC_mod", "no")
PUTPAR("PH_mod", "no")
PUTPAR("NCOEF", str(1000))

#Creating separate datasets for all the chunks in the original FID
chunks = 1

WR(["temp_FIDadd", "123", "49" , dataset[3]])

#Because the first chunk is different size, we do the procedure on
it separately. In the end we read the forward predicted data into
lists.

rec_corr = 8
PUTPAR("NSP",str(rec_corr))
PUTPAR("DATMOD", "raw")
PUTPAR("TDoff", str((0)))
XCMD("LS")

PUTPAR("TDeff", str((d63)/DW - 2*rec_corr))
PUTPAR("LPBIN", str((d63 + d56/2)/DW - rec_corr))
PUTPAR("ME_mod", "LPfc")
PUTPAR("SI", str(td_base/2))
TRFP()
data_r = GETPROCdata(-500,500)
data_i = GETPROCdata(-500,500,type = dataconst.PROCdata_IMAG)

chunks = 0

#Loop for doing the whole procedure for every chunk.
while ( (d63 + d54) + (d54*(chunks+1)) + (d62*(chunks+1)) ) / DW <
TD:
    RE(["temp_FIDadd", "123", "49" , dataset[3]])
    WR(["temp_FIDadd", dataset[1], str(chunks + 2) , dataset[3]])

```

```

RE(["temp_FIDadd", dataset[1], str(chunks + 2) , dataset[3]])
PUTPAR("TDeff", str((d62)/DW - 2*rec_corr))
LPbfc = int(( (d56/2) / DW) + (d62/DW) - rec_corr)
shiftvalue = ((d54 / DW) + (d63/DW)) + (2*d55/DW) + ((d54 /
DW)*float(chunks)) + (d62/DW)*float(chunks) +
(2*d55/DW)*float(chunks) + rec_corr
PUTPAR("NSP",str(shiftvalue))
PUTPAR("DATMOD", "raw")
PUTPAR("TDoff", str((0)))
XCMD("LS")
PUTPAR("DATMOD", "proc")
PUTPAR("ME_mod", "LPfc")
PUTPAR("LPBIN", str(LPbfc))
TRFP()
PUTPAR("ME_mod", "LPbc")
LPbbc = (d62)/DW - 2*rec_corr
PUTPAR("LPBIN", str(LPbbc))
PUTPAR("TDeff", str((LPbfc)))
TDoff = -d56/(2*DW) - rec_corr
PUTPAR("TDoff", str((TDoff)))
TRFP()
PUTPAR("TDoff", str((0)))
backshiftvalue = ((d56/(2*DW)) + (d63/DW)) + ((d54 /
DW)*float(chunks)) + (d62/DW)*float(chunks) +
(2*d55/DW)*float(chunks) - rec_corr
PUTPAR("NSP", str(backshiftvalue))
XCMD("RS")
chunks += 1
data_r2 = GETPROCdata(-500,500)
data_i2 = GETPROCdata(-500,500,type = dataconst.PROCdata_IMAG)
data_r = [x + y for x, y in zip(data_r, data_r2)]
data_i = [x + y for x, y in zip(data_i, data_i2)]
if (d63+d54)/DW + chunks * ((d62 + d54)/DW) > ( TD - ((d62 +
d54)/DW)) :
    chunks += 1
    RE(["temp_FIDadd", "123", "49" , dataset[3]])
    WR(["temp_FIDadd", dataset[1], str(chunks + 2) ,
dataset[3]])
    RE(["temp_FIDadd", dataset[1], str(chunks + 2) ,
dataset[3]])
    PUTPAR("TDeff", str((d63)/DW- 2*rec_corr))
    LPbfc = (d56/2 / DW) + (d63/DW) - rec_corr
    shiftvalue = ((d54 / DW) + (d63/DW)) + (2*d55/DW) + ((d54
/ DW)*float(chunks)) + (d62/DW)*float(chunks) +
(2*d55/DW)*float(chunks) + rec_corr
    PUTPAR("NSP",str(shiftvalue))
    PUTPAR("DATMOD", "raw")
    XCMD("LS")
    PUTPAR("ME_mod", "LPfc")
    PUTPAR("LPBIN", str(LPbfc))
    TRFP()
    PUTPAR("ME_mod", "LPbc")
    LPbbc = (d63/DW) - 2*rec_corr
    PUTPAR("LPBIN", str(LPbbc))
    PUTPAR("TDeff", str((LPbfc)))
    TDoff = -d56/(2*DW) - rec_corr

```

```

        PUTPAR("TDoff", str((TDoff)))
        TRFP()
        PUTPAR("TDoff", str((0)))
        PUTPAR("DATMOD", "proc")
        backshiftvalue = ((d56/2 / DW) + (d63/DW)) + ((d54 /
DW)*float(chunks)) + (d62/DW)*float(chunks) +
(2*d55/DW)*float(chunks)
        PUTPAR("NSP",str(backshiftvalue))
        XCMD("RS")
        data_r2 = GETPROCdata(-500,500)
        data_i2 = GETPROCdata(-500,500,type =
dataconst.PROCdata_IMAG)
        data_r = [x + y for x, y in zip(data_r, data_r2)]
        data_i = [x + y for x, y in zip(data_i, data_i2)]

#Writing the data together and then storing it in a FID
WR([dataset0[0] + "_ekstrap", "88282832", "1337" , dataset0[3]])
RE([dataset0[0] + "_ekstrap", "88282832", "1337" , dataset0[3]])

SAVE_ARRAY_AS_1R1I(data_r, data_i)
PUTPAR("TDeff", "0")
TRFP()
PUTPAR("NSP", str(float(rec_corr)))
XCMD("RS")
data_r3 = GETPROCdata(-500,500)
data_i3 = GETPROCdata(-500,500,type = dataconst.PROCdata_IMAG)
data_grpdly_r = [x + y for x, y in zip(grpdly_cs, data_r3)]
data_grpdly_i = [x + y for x, y in zip(grpdly_csi, data_i3)]

SAVE_ARRAY_AS_1R1I(data_grpdly_r, data_grpdly_i)
PUTPAR("DATMOD", "proc")
PUTPAR("TDeff", str(TD))
GENFID(dataset0[1])
time.sleep(1)
RE([dataset0[0] + "_ekstrap", dataset0[1], "1" , dataset0[3]])
PUTPAR("SI", str(2*TD))
PUTPAR("WDW", "EM")
PUTPAR("PH_mod", "pk")
PUTPAR("BC_mod", "quad")
PUTPAR("FT_mod", "fqc")
XCMD("xau PS_del_tempdata")

```

---

## 11. Additional au macros

### PS\_del\_tempdata code:

---

```
GETCURDATA
```

```
DELETEEXPNO(name, 88282832 , disk, user);
```

```
DELETENAME("temp_FIDadd", disk, user);
```

```
DELETENAME("temp_grpdly", disk, user);
```

```
QUIT
```

---

```
PS_del_exisdata code:
```

---

```
GETCURDATA
```

```
DELETEEXPNO(strcat(name, "_ekstrap"), expno , disk, user);
```

```
QUIT
```

---