Electronic Supplementary Material (ESI) for ChemComm.

Electronic Supplementary Information

Ultraclean pure shift NMR

*Pinelopi Moutzouri, Yingxian Chen, Mohammadali Foroozandeh, Peter Kiraly, Andrew R. Phillips, Steven R. Coombes, Mathias Nilsson and Gareth A. Morris**

NB All raw experimental data, pulse sequence code and processing software are freely available for download from DOI 10.15127/1.309229.

Table of Contents

1. **Pulse sequences and phase cycling**

The pulse sequences used in this work are displayed in Fig. S1 and Fig. S2. In all diagrams closed narrow and wide rectangles represent 90° and 180° hard RF pulses respectively. The shaped pulse represents a soft 180° RF pulse.

Conventional pure shift pulse sequence for pseudo 2D acquisition

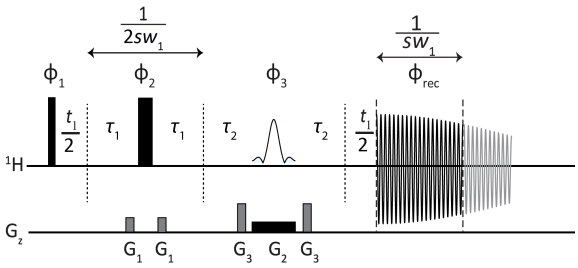


Fig. S1. Pulse sequence for the acquisition of ^{1}H pure shift spectra using the conventional interferogram (pseudo-2D) acquisition.

Phase cycle: $\boldsymbol{\phi_1}$ **:** $x_{16}, -x_{16}$

$\boldsymbol{\phi_2}$ **:** $x_4, y_4, -x_4, -y_4$

$\boldsymbol{\phi_3}$ **:** x, y, -x, -y

$\boldsymbol{\phi_{rec}}$ **:** $\{x, -x, x, -x, -x, x, -x, x\}_4$

Pure shift pulse sequence for SAPPHIRE pseudo 2D acquisition

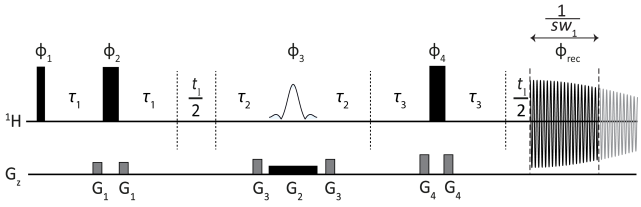


Fig. S2. Pulse sequence for the acquisition of ultraclean ^{1}H pure shift spectra using the SAPPHIRE technique.

Phase cycle: $\boldsymbol{\phi_1}$ **:** x_4, y_4

$\boldsymbol{\phi_2}$ **:** x_2, y_2

$\boldsymbol{\phi_3}$ **:** x, y

$\boldsymbol{\phi_4}$ **:** x

$\boldsymbol{\phi_{rec}}$ **:** x,-x,-x,x,-y,y,y,-y

2. **Sample details**

The sample used for the figures of the main text and all the figures of the ESI, apart from Fig Fig. S5, contained the anti-cholesterol active pharmaceutical ingredient, rosuvastatin, and a small amount of its precursor, BEM*, in DMSO-d_6.

In the final mixture rosuvastatin and BEM had a concentration of 58.3 mM and 1.65 mM respectively. Rosuvastatin and BEM were supplied by AstraZeneca, while DMSO-d_6 was supplied from Cambridge Isotope Laboratories Inc.

* tert-butyl-E-(6-[-[4-(4-fluorophenyl)-6-isopropyl-2-[methyl(methylsulfonyl)amino]pyrimidin-5-yl]-vinyl]-((4R,6S)-2,2-dimethyl[1,3]dioxin-4yl) acetic acid)

The sample of Fig. S5 of the ESI was a 90 mM sample of ethyl isovalerate (supplied from Sigma-Aldrich) in DMSO-d_6.

3. **Experiment details**

Unless stated otherwise, all data were acquired with non-spinning sample on a 500 MHz Bruker Avance NEO spectrometer at 25 °C with an RF amplitude of 24.9 kHz for the ^{1}H pulses.

The ^{1}H 1D spectrum of Fig. 1a was acquired with a 2 s relaxation delay (d1), 8 dummy scans, 512 transients and a 5 kHz spectral window. The ^{1}H pure shift spectra of Fig. 1b/c and 1d were acquired with a 2 s relaxation delay (d1), 8 dummy scans and 512 transients, using a 5 kHz spectral window in the direct dimension and a 39.0625 Hz spectral window in the indirect dimension (t_1) of the pure shift. Both spectra were acquired with a total T_2 weighting of 32 ms which is higher than the minimum T_2 weighting that could have been applied. The spectrum of Fig. 1c was acquired in 9 h and 2 min while the spectrum of Fig. 1d was acquired in 9 h and 38 min. The acquisition time of the spectrum shown in Fig. 1d is longer because an extra chunk has to be acquired for every different pure shift data set in order to compensate for the different duration of the first chunk. The final pure shift spectra in both cases were constructed after the concatenation of 16 chunks leading to a total number of 2048 complex points and were zero filled to 32768 complex points. For the spatial encoding a 50 ms selective rsnob pulse with an RF amplitude of 46.8 Hz was used simultaneously with a field gradient pulse of 2 G/cm. Resolution enhancement was applied to the final 1D pure shift spectra.

The ^{1}H 1D spectrum of Fig. 1e was acquired with a 1 s relaxation delay (d1), 1 transient, and 5 kHz spectral window.

4. **Assignment of pure compounds**

Rosuvastatin in DMSO-d_6

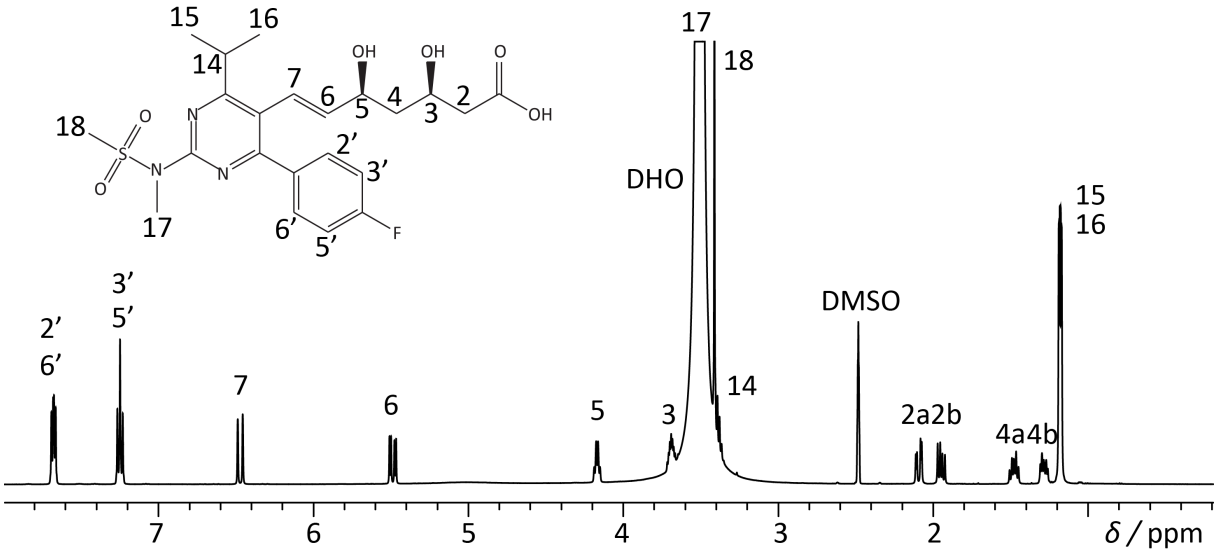


Fig. S3. (a) 500 MHz 1D ^{1}H spectrum for a sample of pure rosuvastatin.

BEM in DMSO-d_6

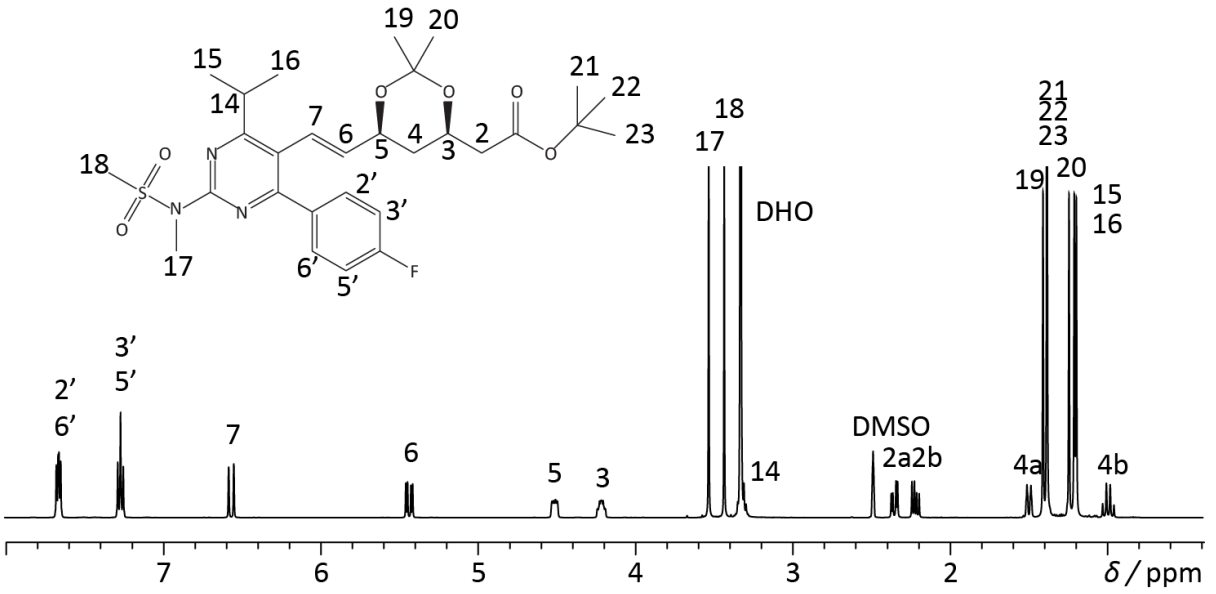


Fig. S4. (a) 500 MHz 1D ^{1}H spectrum for a sample of pure BEM.

Both spectra were acquired with a 5 s relaxation delay (d1), 8 dummy scans, and 256 scans in a Varian/Agilent VNMRS spectrometer. A Gaussian time constant of 0.4096 s was applied. An RF amplitude of 29.9 kHz was used for the ^{1}H pulses.

5. **Comparison with variable sw_1 method**

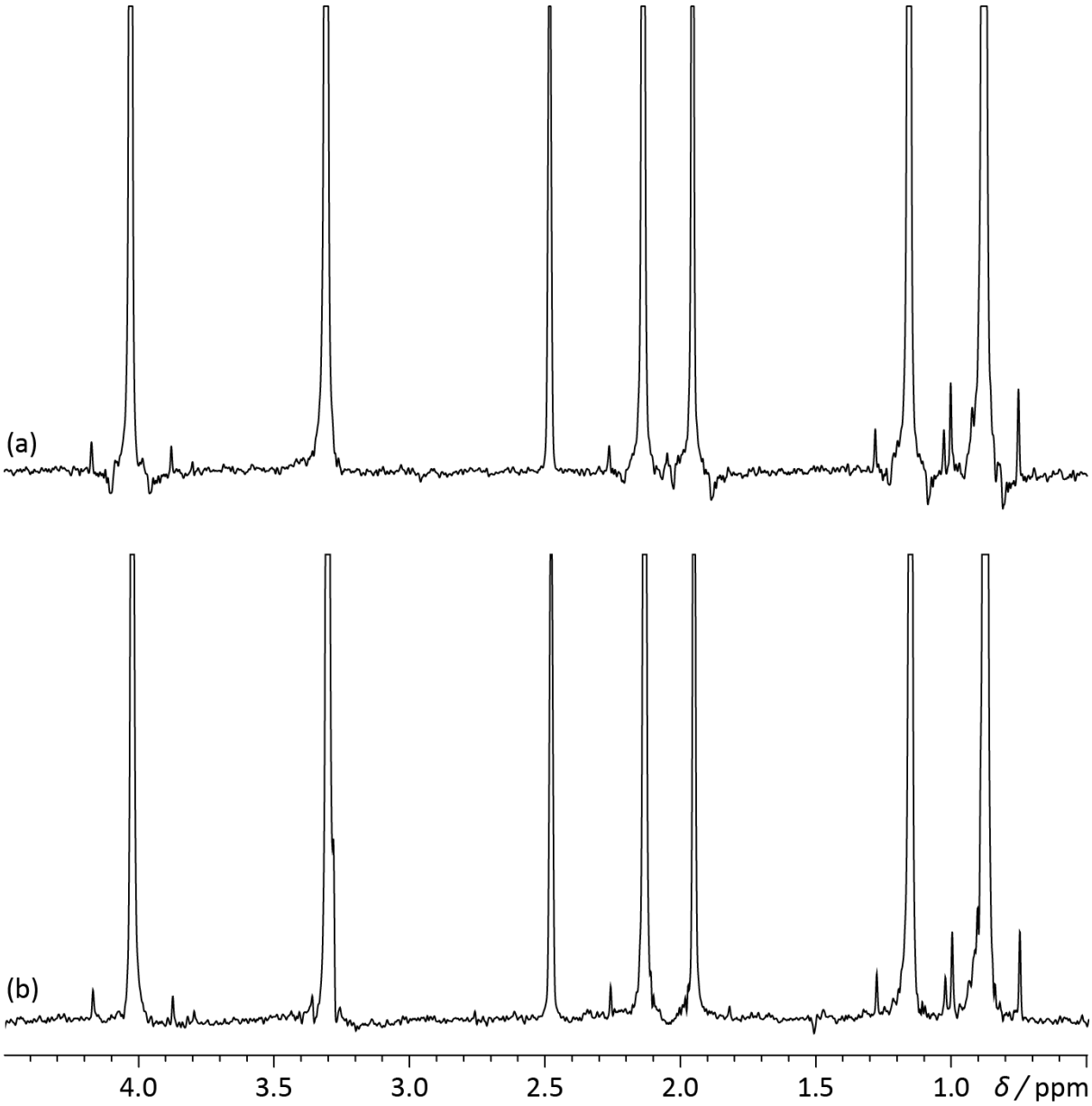


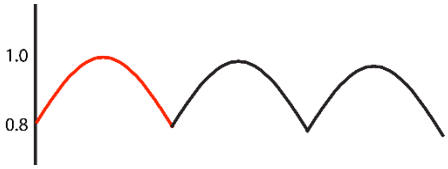
Fig. S5. (a) 500 MHz ^{1}H pure shift spectrum constructed after the averaging of pure shift spectra with different values of sw_1 and (b) 500 MHz ^{1}H pure shift acquired with the new SAPPHIRE technique for a sample of ethyl isovalerate in DMSO-d_6.

The spectra were acquired in a Varian/Agilent VNMRS spectrometer. An RF amplitude of 30.3 KHz was used for the ^{1}H pulses. Both pure shift spectra were acquired with a 5 s relaxation delay (d1), 8 dummy scans, 32 scans, and 32 chunks. Here, a 37 ms rsnob pulse

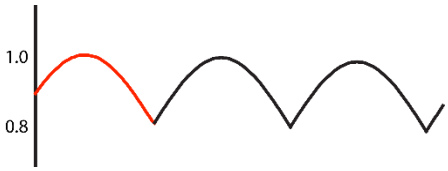
with a bandwidth of 50 Hz was applied simultaneously with a 0.56 G/cm pulsed field gradient. A Gaussian time constant of 0.20 s and a line broadening of 1 Hz were applied.

6. **Principle of the SAPPHIRE method**

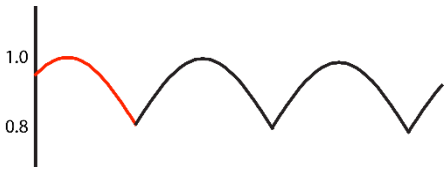
Varying $\Delta\tau_J$ changes the modulation phase, and hence the phases of the sideband signals, by adjusting the size of the first chunk according to the refocusing point of the *J* modulation.



$\Delta\tau_J = 0$ chunk1= chunk



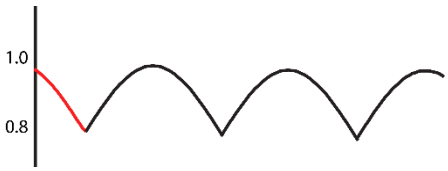
$\Delta\tau_J = 1 / 8\ sw_1$ chunk1= 0.875*chunk



$\Delta\tau_J = 1 / 4\ sw_1$ chunk1=0.75*chunk

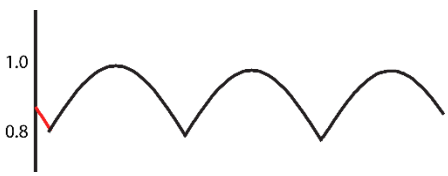
$\Delta\tau_J = 3 / 8\ sw_1$ chunk1=0.625*chunk

$\Delta\tau_J = 1 / 2\ sw_1$ chunk1=0.5*chunk



$\Delta\tau_J = 5 / 8\ sw_1$ chunk1=0.375*chunk

$\Delta\tau_J = 3 / 4\ sw_1$ chunk1=0.25*chunk



$\Delta\tau_J = 7 / 8\ sw_1$ chunk1=0.125*chunk

Fig. S6. Graphical representation of the shifting of the *J* modulation pattern with respect to the incremented *J* evolution time $\Delta\tau_J$. Chunk1 represents the length of the first chunk with respect to the rest of the chunks for an 8-step SAPPHIRE suppression.

Averaging the above signal modulations leads to the following:

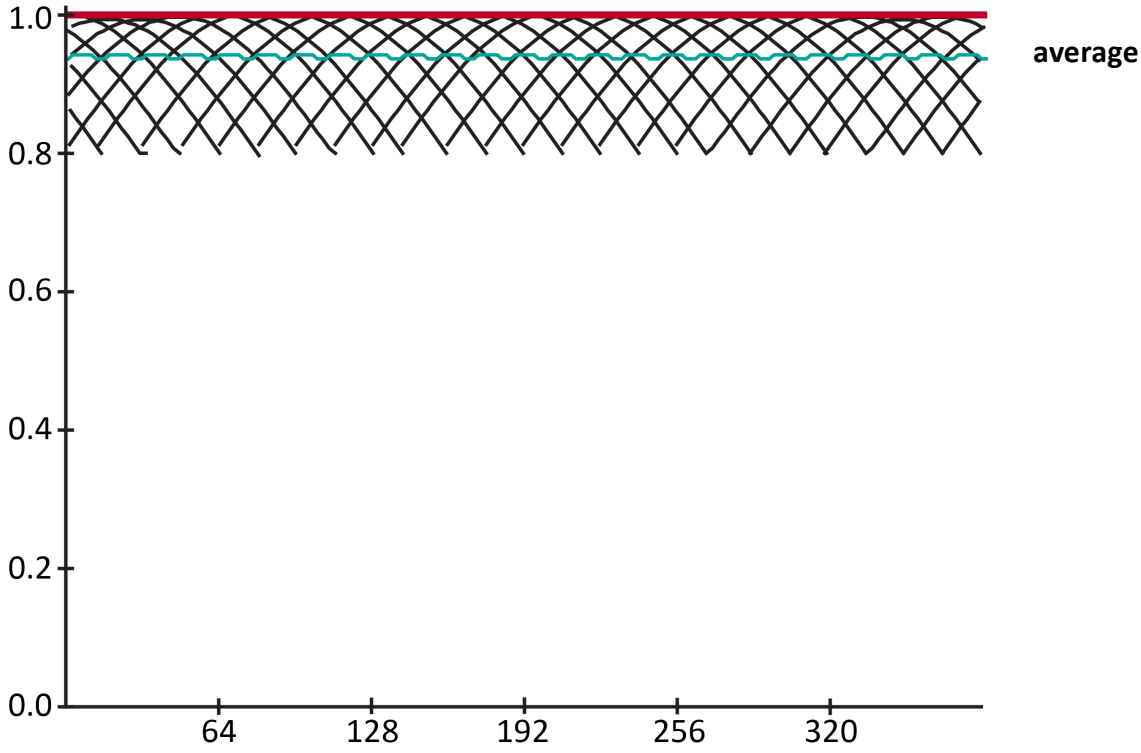


Fig. S7. Graphical representation of the pattern of signal *J* modulation. The *x* axis indicates the amplitude of the modulation while the *y* axis indicates the number of points in the interferogram FID.

7. **Comparing the performance of the sideband suppression**

Different numbers of steps in the SAPPHIRE suppression

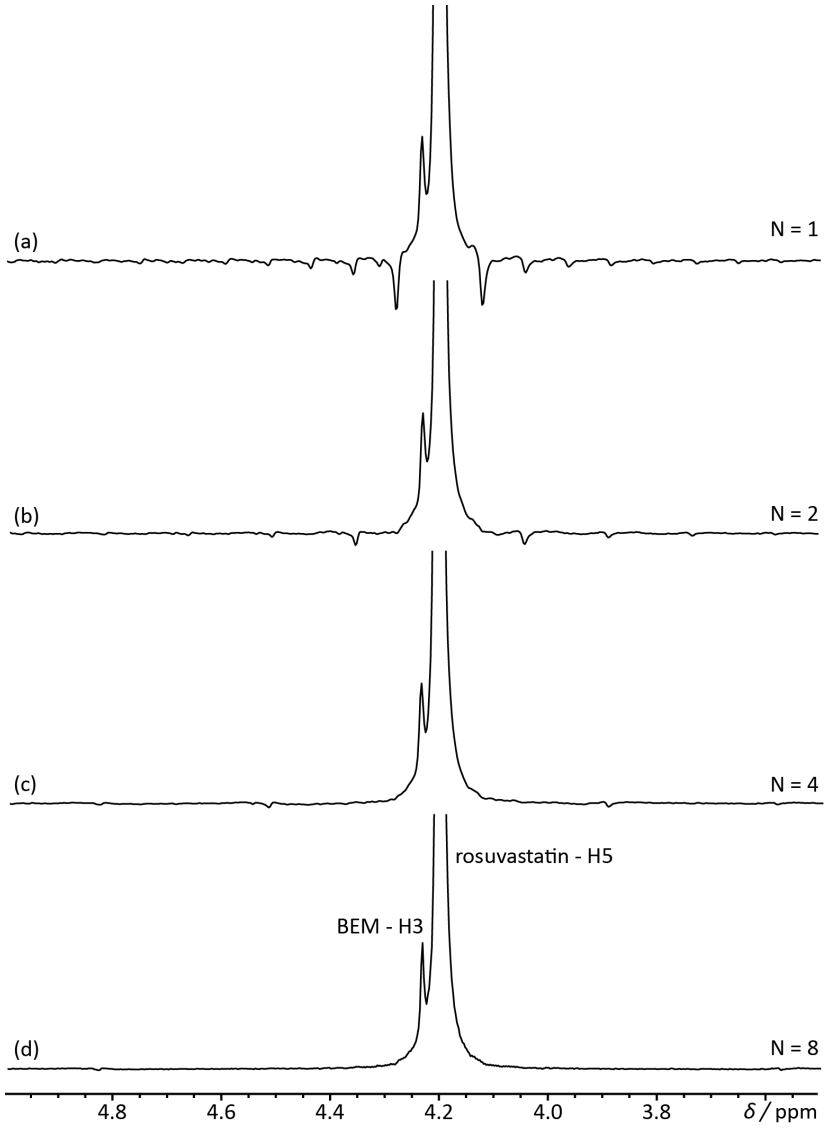


Fig. S8. 500 MHz ^{1}H band-selective pure shift spectra, acquired with the new SAPPHIRE technique, in which (a) one, (b) two, (c) four, and (d) eight different $\Delta\tau_i$ values were averaged.

Different sw_1 values

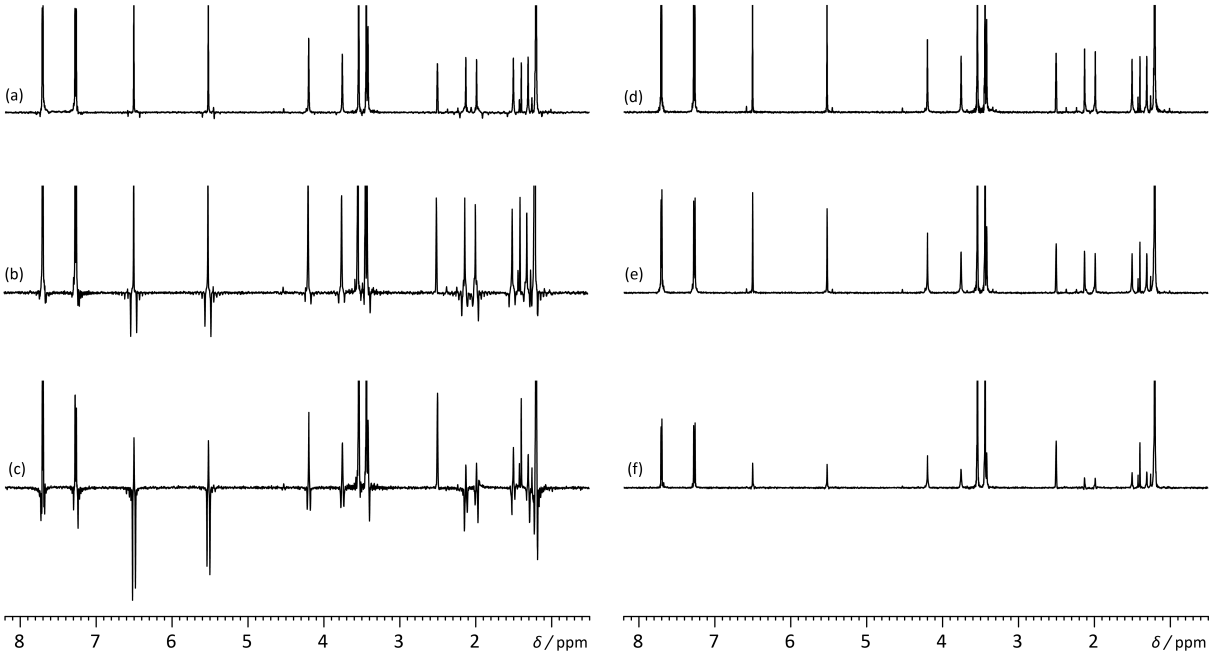


Fig S9. 500 MHz ^{1}H pure shift spectra (a-c) acquired with the pulse sequence of Fig. S1 and (d-f) acquired with the new SAPPHIRE pulse sequence of Fig. S2 with a sw_1 of 39 Hz, 19 Hz and 9 Hz respectively. (a) and (d) were acquired with 16 chunks, (b) and (e) with 8 chunks and (c) and (f) with 4 chunks.

8. Application of the SAPPHIRE technique to different kinds of pure shift experiment

PSYCHE

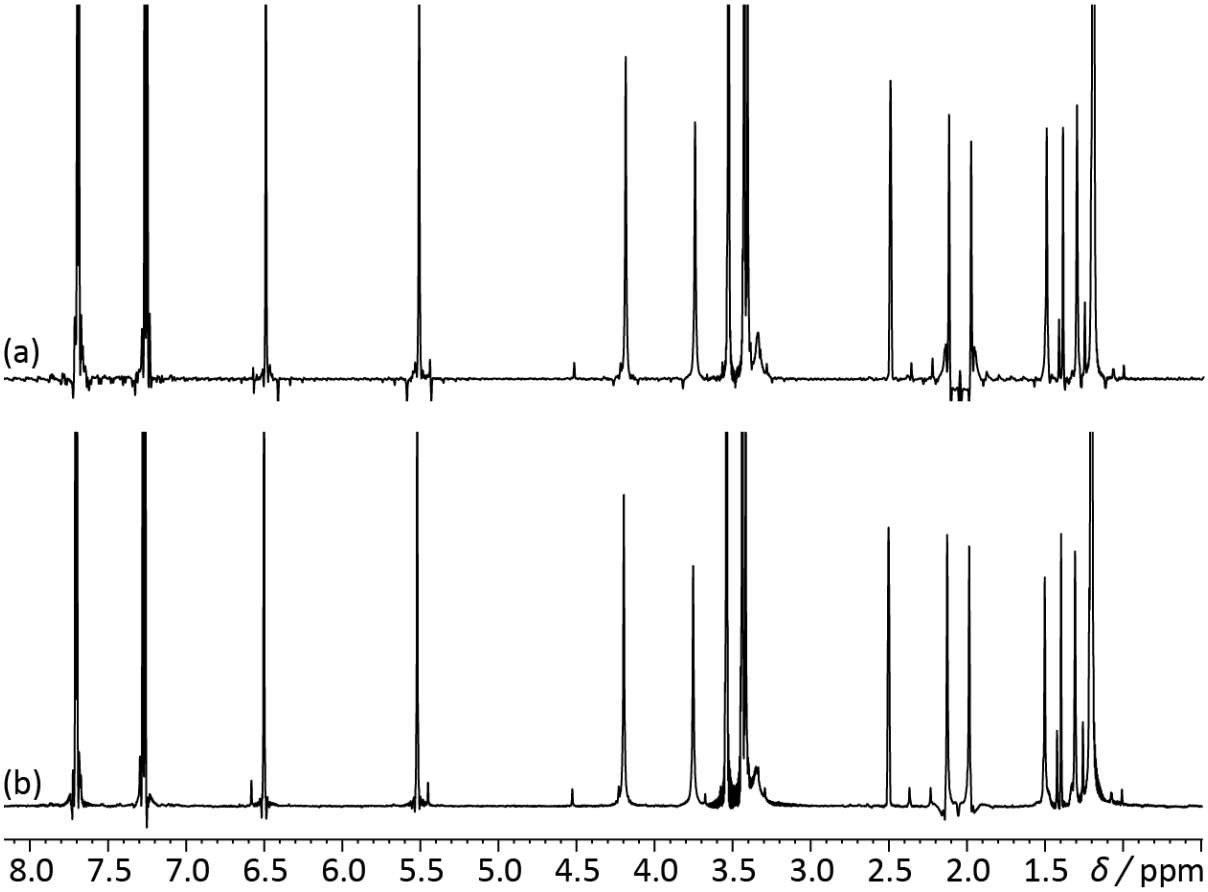


Fig S10. 500 MHz ^{1}H PSYCHE pure shift spectra acquired with (a) the pulse sequence of Fig. S1 and (b) with the new SAPPHIRE technique.

A PSYCHE element was used as the selective spin inversion element with a flip angle β of 20^{o} and a double saltire pulse of 30 ms with a 10 kHz bandwidth. The pure shift spectra were acquired with 512 scans, 16 chunks and a relaxation delay of 2 s. Unlike the Zangger-Sterk method used in the main text, PSYCHE generates spectra which contain small "recoupling" artefacts, in proportion to the fourth power of the flip angle β used, that do not arise from periodic perturbations caused by chunked acquisition. SAPPHIRE suppresses the latter source of periodic sidebands, but faithfully retains the recoupling artefacts.

Band-selective

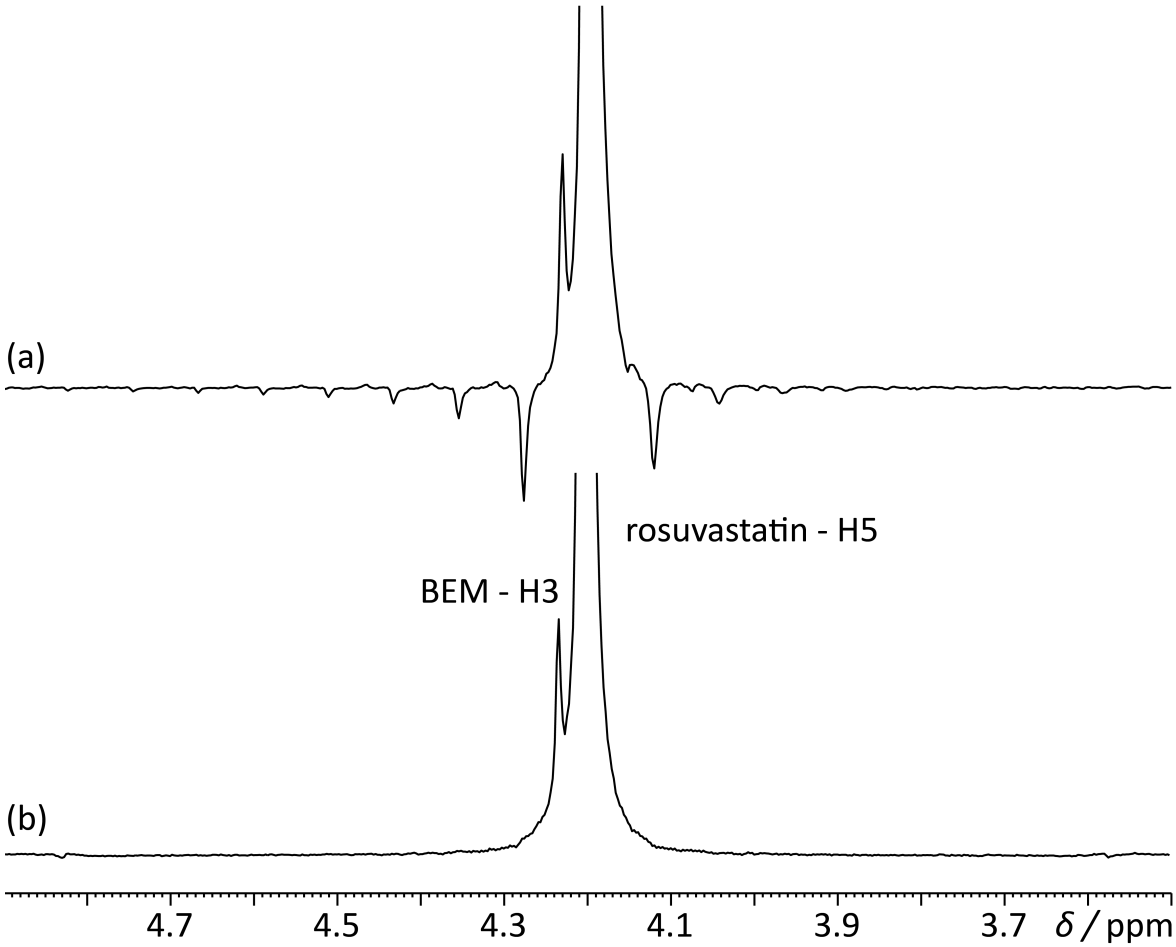


Fig S11. 500 MHz ^{1}H band-selective pure shift spectra acquired with (a) the pulse sequence of Fig. S1 and (b) with the new SAPPHIRE technique for H5 of rosuvastatin.

The band-selective pure shift spectra were acquired with 128 scans.

9. **Strong Coupling**

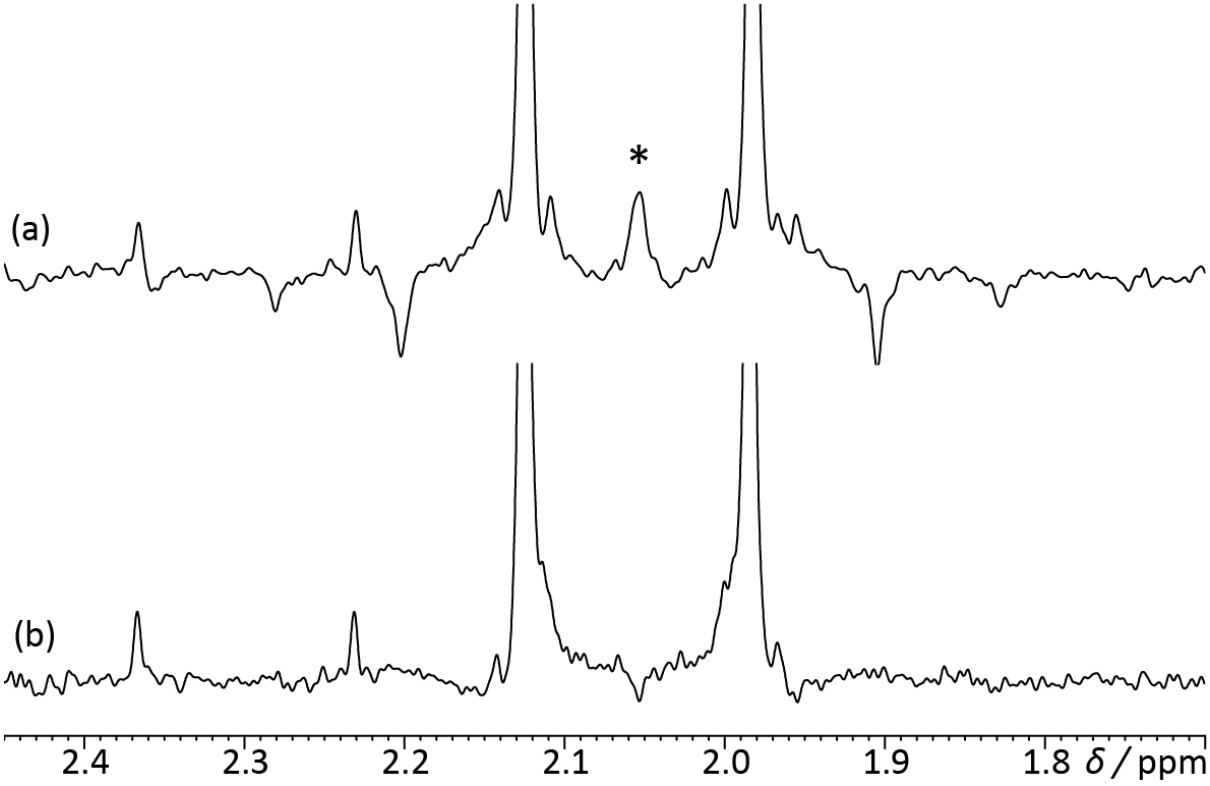


Fig S12. 500 MHz ^{1}H pure shift spectra acquired with (a) the pulse sequence of Fig. S1 and (b) with the new SAPPHIRE technique with an sw_1 of 39 Hz. Both spectra are part of Fig.1 of the main text at the area of the two strongly coupled protons.

10. **Bruker pulse sequence code**

NB All raw experimental data, pulse sequence code and processing software are freely available for download from DOI 10.15127/1.309229.

Acquired with a Zangger-Sterk SSI

```
;sapphire
;
;       Sideband Averaging by Periodic PHase
;       Incrementation of Residual J Evolution
;       for the acquisition of clean ZS pure shift spectra
;
;       Moutzouri Pinelopi
;       NMR Methodology Group
;       University of Manchester
;
;       The pulse sequence involves a 3D acquisition scheme.
;       F3 is the direct dimension. F1 is the incremented dimension for the reconstruction of the pure shift interferogram.
;       F2 is the incremented dimension for the J-evolution.
;
;       The data can be reconstructed using the two following AU programs (downloaded from:
        http://nmr.chemistry.manchester.ac.uk)
;               1) pm_pshift (produces pure shift spectra for each different J-evolution time, adjusting the length of the first chunk
                  appropriately)
;               2) pm_fidadd (averages the pure shift spectra acquired with different J-evolution times)
;
;
;$CLASS=HighRes
;$DIM=3D
;$TYPE=
;$SUBTYPE=
;$COMMENT=

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

define delay tauA
define delay tauAA
define delay tauB
define delay tauBB
define delay tauBBB
define delay tauBBBB
define delay tauC
define delay tauD

"p2=p1*2"
"l0=0"
"l8=0"
"in0=inf1/2"
"in10=inf2"
"d0=inf1/2"
"d10=inf2"
"d30=in0/2"
"d40=in0/2-in10"
"cnst5=(td2/2)+1"

"tauA=0"
"tauAA=inf1/4"
"tauB=d2-p17-2*d16-20u"
"tauBB=d2-p17-2*d16-20u-inf2"
```

```
"tauBBB=d2-p17-2*d16-20u"
"tauBBBB=d2-p17-2*d16-20u+inf2"
"tauC=inf2"
"tauD=(dw*2*cnst4)"


aqseq 312



1 ze
2 50m
        d1 pl1:f1
3 50u UNBLKGRAD


 if "l8 < cnst5"
  {
                "tauA=0"
                "tauB=d2-p17-2*d16-20u"
                "tauA=tauA+(l8*in10)"
                "tauB=tauB-(l8*in10)"
                "d30=in0/2+(l8*in10)"
                "d30=d30+((l0-1)*in0)"
                "tauBBB=d2-p17-2*d16-20u"
                "tauBBB=tauBBB-(l8*in10)"

                p1 ph1

                if "l0==0"
                        {
                                tauA
                        }
                else
                        {
                                tauAA
                        }
                                p16:gp1
                                d16
                                p2 ph2
                                p16:gp1
                                d16

                if "l0==0"
                        {
                                tauA
                        }
                else
                        {
                                tauAA
                        }


                if "l0==0"
                        {

                        }
                else
                        {
                                d30
                        }

                if "l0==0"
                        {
                                tauB
                        }
                else
                        {
                                tauBBB
                        }
```

```
                    d16
                    p17:gp2
                    d16
                20u gron0 pl0:f1
                    ;
                (p11:sp11 ph3):f1
                    ;
                20u groff pl1:f1
                    d16
                    p17:gp2
                    d16

            if "l0==0"
                {
                    tauB
                }
        else
                {
                    tauBBB
                }

                    tauD
                    p18:gp3
                    d16
                    p2 ph4
                    p18:gp3
                    d16 BLKGRAD
            if "l0==0"
                {

                }
            else
                {
                    d30
                }


lab1, goto lab7

        }

else
            {

            "tauC=in10"
            "tauBB=d2-p17-2*d16-20u-in10"
            "tauC=tauC+((l8-cnst5)*in10)"
            "tauBB=tauBB-((l8-cnst5)*in10)"
            "d40=(in0/2-in10)-((l8-cnst5)*in10)"
            "d40=d40+((l0-1)*in0)"
            "tauBBBB=d2-p17-2*d16-20u+in10"
            "tauBBBB=tauBBBB+((l8-cnst5)*in10)"

            p1 ph1
            if "l0==0"
                {

                }
        else
                {
                    tauAA
                }
                    p16:gp1
                    d16
                    p2 ph2
                    p16:gp1
                    d16
             if "l0==0"
                {

                }
```

```
        else
                        {
                                tauAA
                        }

         if "l0==0"
                        {

                        }
                else
                        {
                                d40
                        }

                if "l0==0"
                        {
                                tauBB
                        }
        else
                        {
                                tauBBBB
                        }

                                d16
                                p17:gp2
                                d16
                        20u gron0 pl0:f1
                                ;
                        (p11:sp11 ph3):f1
                                ;
                        20u groff pl1:f1
                                d16
                                p17:gp2
                                d16
                if "l0==0"
                        {
                                tauBB
                        }
        else
                        {
                                tauBBBB
                        }

                 tauD

                if "l0==0"
                        {
                                tauC
                        }
        else
                        {

                        }

                                p18:gp3
                                d16
                                p2 ph4
                                p18:gp3
                                d16 BLKGRAD
        if "l0==0"
                        {
                                tauC
                        }
        else
                        {

                        }

if "l0==0"
                        {
```

```
                        }
                else
                        {
                                d40
                        }


lab2, goto lab7
        }


lab7, go=2 ph31
50m mc #0 to 2
        F1QF(calclc(l0,1))
 F2QF(calclc(l8,1))
exit


ph1 =0 0 0 0 1 1 1 1
ph2 =0 0 1 1 0 0 1 1
ph3 =0 1 0 1 0 1 0 1
ph4 =0 0 0 0 0 0 0 0
ph31=0 2 2 0 3 1 1 3


;POWER LEVEL
;pl0  : zero power (0W)
;pl1  : power level for pulse (default)
;sp11 : power level of ZS selective pulse

;PULSE DURATION
;p1: high power 90 pulse width
;p2: high power 180 pulse width
;p11: duration of ZS selective pulse

;PULSE SHAPE
;spnam11: file name of ZS selective pulse

;GRADIENT DURATION
;p16: CTP gradient pulse width
;p17: CTP gradient pulse width
;p18: CTP gradient pulse width

;GRADIENT SHAPE
;gpnam1: SINE.100
;gpnam2: SINE.100
;gpnam3: SINE.100

;GRADIENT STRENGTH
;gpz1  : CTP gradient [77%]
;gpz2  : CTP gradient [49%]
;gpz3  : CTP gradient [63%]
;gpz0: weak gradient during SSI element (1-4%)

;DELAYS
;d1: relaxation delay; 1-5 * T1
;d16: gradient stabilisation delay
;d2: delay to keep the T2 weighting constant between the pure shift experiments acquired with different evolution time [greater than
1/4SW1+p16+2*d16]


;CONSTANTS
;cnst4: number of points to drop when collecting FID
;cnst5:(td2/2)+1

;OTHERS
;td1: number of chunks to be acquired
;td2: number of different J evolution times to be averaged (N)
;ns: 8 * n, total number of scans
;ds: 8, number of dummy scans
```

;sw1: sw3/n (n has to be an integer number)
;sw2: 1/16sw1 when an N=8 SAPPHIRE cycle is acquired (td2=8), 1/8sw1 when an N=4 SAPPHIRE cycle is acquired (td2=4), 1/4sw1 when an N=2 SAPPHIRE cycle is acquired (td2=2)
;in0: 1/(2 * sw1)
;in10: 1/sw2
;l8: loop counter for F2 dimension
;l0: loop counter for F1 dimension
;FnMODE1: QF
;FnMODE2: QF

Acquired with a PSYCHE SSI

```
;sapphire
;
;		Sideband Averaging by Periodic PHase
;		Incrementation of Residual J Evolution
;		for the acquisition of clean PSYCHE pure shift spectra
;
;		Moutzouri Pinelopi
;		NMR Methodology Group
;		University of Manchester
;
;		The pulse sequence involves a 3D acquisition scheme.
;		F3 is the direct dimension. F1 is the incremented dimension for the reconstruction of the pure shift interferogram.
;		F2 is the incremented dimension for the J-evolution.
;
;		The data can be reconstructed using the two following AU programs (downloaded from:
http://nmr.chemistry.manchester.ac.uk)
;			1) pm_pshift (produces pure shift spectra for each different J-evolution time, adjusting the length of the first chunk
appropriately)
;			2) pm_fidadd (averages the pure shift spectra acquired with different J-evolution times)
;
;$CLASS=HighRes
;$DIM=3D
;$TYPE=
;$SUBTYPE=
;$COMMENT=

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

define delay tauA
define delay tauAA
define delay tauB
define delay tauBB
define delay tauBBB
define delay tauBBBB
define delay tauC
define delay tauD

"p2=p1*2"

"l0=0"
"l8=0"

"in0=inf1/2"
"in10=inf2"

"d0=inf1/2"
"d10=inf2"
```

```
"d30=in0/2"
"d40=in0/2-in10"

"cnst5=(td2/2)+1"

"tauA=0"
"tauAA=inf1/4"
"tauB=d2-p17-2*d16-20u"
"tauBB=d2-p17-2*d16-20u-inf2"
"tauBBB=d2-p17-2*d16-20u"
"tauBBBB=d2-p17-2*d16-20u+inf2"
"tauC=inf2"
"tauD=(dw*2*cnst4)"

"cnst50=(cnst20/360)*sqrt((2*cnst21)/(p40/2000000))"
"p30=1000000.0/(cnst50*4)"

"cnst31= (p30/p1) * (p30/p1)"
"spw40=plw1/cnst31"

"p10=p40"



aqseq 312

1 ze
2 50m
        d1 pl1:f1
3 50u UNBLKGRAD


 if "l8 < cnst5"
  {
                "tauA=0"
                "tauB=d2-p17-2*d16-20u"
                "tauA=tauA+(l8*in10)"
                "tauB=tauB-(l8*in10)"
                "d30=in0/2+(l8*in10)"
                "d30=d30+((l0-1)*in0)"
                "tauBBB=d2-p17-2*d16-20u"
                "tauBBB=tauBBB-(l8*in10)"

                p1 ph1

                if "l0==0"
                        {
                                tauA
                        }
                else
                        {
                                tauAA
                        }
                                p16:gp1
                                d16
                                p2 ph2
                                p16:gp1
                                d16

                if "l0==0"
                        {
                                tauA
```

```
                }
        else
                {
                        tauAA
                }


        if "l0==0"
                {

                }
        else
                {
                        d30
                }

        if "l0==0"
                {
                        tauB
                }
        else
                {
                        tauBBB
                }

                        d16
                        p17:gp2
                        d16
                20u pl0:f1
                        ;
                ( center (p40:sp40 ph3):f1 (p10:gp10) )
                        ;
                20u pl1:f1
                        d16
                        p17:gp2
                        d16

        if "l0==0"
                {
                        tauB
                }
else
                {
                        tauBBB
                }

                        tauD
                        p18:gp3
                        d16
                        p2 ph4
                        p18:gp3
                        d16 BLKGRAD
        if "l0==0"
                {

                }
        else
                {
                        d30
                }
```

```
lab1, goto lab7

        }

else
                {

                "tauC=in10"
                "tauBB=d2-p17-2*d16-20u-in10"
                "tauC=tauC+((l8-cnst5)*in10)"
                "tauBB=tauBB-((l8-cnst5)*in10)"
                "d40=(in0/2-in10)-((l8-cnst5)*in10)"
                "d40=d40+((l0-1)*in0)"
                "tauBBBB=d2-p17-2*d16-20u+in10"
                "tauBBBB=tauBBBB+((l8-cnst5)*in10)"

                p1 ph1
                if "l0==0"
                        {

                        }
        else
                        {
                                tauAA
                        }
                                p16:gp1
                                d16
                                p2 ph2
                                p16:gp1
                                d16
                 if "l0==0"
                        {

                        }
        else
                        {
                                tauAA
                        }

         if "l0==0"
                        {

                        }
                else
                        {
                                d40
                        }

                if "l0==0"
                        {
                                tauBB
                        }
        else
                        {
                                tauBBBB
                        }

                                d16
                                p17:gp2
                                d16
                        20u pl0:f1
                                ;
```

```
                                ( center (p40:sp40 ph3):f1 (p10:gp10) )
                                        ;
                                20u pl1:f1
                                        d16
                                        p17:gp2
                                        d16
                        if "l0==0"
                                {
                                        tauBB
                                }
                else
                                {
                                        tauBBBB
                                }

                         tauD

                        if "l0==0"
                                {
                                        tauC
                                }
                else
                                {

                                }

                                        p18:gp3
                                        d16
                                        p2 ph4
                                        p18:gp3
                                        d16 BLKGRAD
                if "l0==0"
                                {
                                        tauC
                                }
                else
                                {

                                }

  if "l0==0"
                                {

                                }
                        else
                                {
                                        d40
                                }



lab2, goto lab7
                 }

lab7, go=2 ph31
50m mc #0 to 2
                F1QF(calclc(l0,1))
  F2QF(calclc(l8,1))
exit


ph1 =0 0 0 0 1 1 1 1
```

ph2 =0 0 1 1 0 0 1 1
ph3 =0 1 0 1 0 1 0 1
ph4 =0 0 0 0 0 0 0 0
ph31=0 2 2 0 3 1 1 3

;POWER LEVEL
;pl0 : zero power (0W)
;pl1 : power level for pulse (default)
;spw40 : power level of ZS selective pulse

;PULSE DURATION
;p1: high power 90 pulse width
;p2: high power 180 pulse width
;p10 : duration of weak gradient during PSYCHE pulse element
;p40 : duration of double-chirp PSYCHE pulse element

;PULSE SHAPE
;spnam40: file name for PSYCHE pulse element

;GRADIENT DURATION
;p16: CTP gradient pulse width
;p17: CTP gradient pulse width
;p18: CTP gradient pulse width

;GRADIENT SHAPE
;gpnam1: SINE.100
;gpnam2: SINE.100
;gpnam3: SINE.100
;gpnam10: RECT.1

;GRADIENT STRENGTH
;gpz1 : CTP gradient [77%]
;gpz2 : CTP gradient [49%]
;gpz3 : CTP gradient [63%]
;gpz10 : weak gradient during PSYCHE element (1-4%)

;DELAYS
;d1: relaxation delay; 1-5 * T1
;d16: gradient stabilisation delay
;d2: delay to keep the T2 weighting constant between the pure shift experiments acquired with different evolution time [greater than 1/4SW1+p16+2*d16]

;CONSTANTS
;cnst4: number of points to drop when collecting FID
;cnst5:(td2/2)+1
;cnst20: desired flip angle for PSYCHE pulse element (degree) (normally 10-25)
;cnst21: bandwidth of each chirp in PSYCHE pulse element (Hz) (normally 10000)

;OTHERS
;td1: number of chunks to be acquired
;td2: number of different J evolution times to be averaged (N)
;ns: 8 * n, total number of scans
;ds: 8, number of dummy scans
;sw1: sw3/n (n has to be an integer number)
;sw2: 1/16sw1 when an N=8 SAPPHIRE cycle is acquired (td2=8), 1/8sw1 when an N=4 SAPPHIRE cycle is acquired (td2=4), 1/4sw1 when an N=2 SAPPHIRE cycle is acquired (td2=2)
;in0: 1/(2 * sw1)
;in10: 1/sw2
;l8: loop counter for F2 dimension
;l0: loop counter for F1 dimension
;FnMODE1: QF
;FnMODE2: QF

11.**Varian pulse sequence code**

NB All raw experimental data, pulse sequence code and processing software are freely available for download from DOI 10.15127/1.309229.

```
/*sapphire pulse sequence
        Sideband Averaging by Periodic PHase
        Incrementation of Residual J Evolution
        for the acquisition of clean pure shift spectra

        Developed by:
        NMR Methodology Group
        University of Manchester

*/

/*User's Guide for experimental setup and processing

        The pulse sequence involves a 3D acquisition scheme.
        - Direct dimension
        - 1st indirect dimension: for the reconstruction of the pure shift interferogram
        - 2nd indirect dimension: for incrementation of the ΔτJ delay related to the J-evolution


        The 1D final pure shift spectrum can be reconstructed using the two following macros (downloaded from:
        http://nmr.chemistry.manchester.ac.uk)
        1) pm_pshift_v (produces pure shift spectra for each different J-evolution time, adjusting the length of the first
        chunk appropriately)
                Two arguments are needed: 1) number of chunks that will contribute to the final pure shift spectrum (ni-1)
                                          2) number of steps in the SAPPHIRE cycle (ni2) (usually 8, 4 or 2)
        2) pm_fidadd_v (averages the pure shift spectra acquired with different J-evolution times)
                Three arguments are needed: 1) number of the first pure shift spectrum to be added (usually 1)
                                            2) number of the last pure shift spectrum to be added (usually 8)
                                            3) step of addition (usually 1)
*/

/*
General Parameters


        d1              :       relaxation delay
        d2              :       t1 evolution delay (normally zero)
        ni              :       number of chunks to be acquired
        ni2             :       number of different J evolution times to be averaged
        sw1             :       sw/n (n has to be an integer number)
        sw2             :       1/16sw1 when an 8 step cycle is acquired (ni2=8), 1/8sw1 when a 4 step cycle is acquired
                                (ni2=4), 1/4sw1 when a two step cycle is acquired (ni2=2)

*/


#include <standard.h>
#include <Pbox_psg.h>


//phase cycle
static int

ph1[64] = {0,0,0,0, 1,1,1,1},           //v1    excitation 90
ph2[64] = {0,0,1,1, 0,0,1,1},           //v2    hard 180 (1st)
ph3[64] = {0,1,0,1, 0,1,0,1},           //v3    soft 180 (or PSYCHE)
ph4[64] = {0,0,0,0, 0,0,0,0},           //v4    hard 180 (2nd)
ph5[64] = {0,2,2,0,3,1,1,3};            //oph
```

```
pulsesequence()

{

double    droppts = getval("droppts"),                    /* number of dummy points to acquire */
                tpwr = getval("tpwr"),                    /* 1H pulse power*/
                pw = getval("pw"),                        /* 1H pulse width*/
                gzlvl1=getval("gzlvl1"),                  /* CTP selection gradient */
                gzlvl2=getval("gzlvl2"),                  /* weak gradient for slice selection */
                gzlvl3=getval("gzlvl3"),                  /* CTP selection gradient */
                gzlvl4=getval("gzlvl4"),                  /* CTP selection gradient */
                hsglvl=getval("hsglvl"),                  /* steady-state crusher gradient */
          hsgt=getval("hsgt"),                            /* steady-state gradient pulse width */
                gt1=getval("gt1"),                        /* CTP gradient pulse width */
                gt3=getval("gt3"),                         /* CTP gradient pulse width */
                gt4=getval("gt4"),                         /* CTP gradient pulse width */
          gstab=getval("gstab"),                           /* gradient stabilisation delay */
                selpw=getval("selpw"),                     /* pulse length for the soft 180  */
                selpwr=getval("selpwr"),                   /* power level  for the soft 180  */


                tau1   = (1.0/(4.0*sw1))-d3,              /*calculating tau1 (hard pulse echo) */
                tau1a   = d3-(1.0/(4.0*sw1)),
                tau2max   = getval("tau2max"),            /*T2 weighting compensation */
                tau2  = (tau2max-gt3-2.0*gstab)-tau1,      /*calculating tau2 (SSI echo) */
                tau2a  = (tau2max-gt3-2.0*gstab)+tau1,

                deltat1   = d2/2.0-d3;                    /*calculating t1 */

int       kpph = (int)(getval("kpph"));                   //number of steps from the phase cycle to use

char      sspul[MAXSTR],
                lkgate_flg[MAXSTR],
          selshape[MAXSTR];                               /* pulse file for the soft 180 */

          getstr("sspul",sspul);
          getstr("selshape",selshape);
          getstr("lkgate_flg",lkgate_flg);


/******/
          if (kpph==0)
          {
          settable(t1,64,ph1);
          settable(t2,64,ph2);
          settable(t3,64,ph3);
          settable(t4,64,ph4);
          settable(t5,64,ph5);

          }
          if (kpph>0)
          {
          settable(t1,kpph,ph1);
          settable(t2,kpph,ph2);
          settable(t3,kpph,ph3);
          settable(t4,kpph,ph4);
          settable(t5,kpph,ph5);
          }
          if (kpph<0)
          {
          settable(t1,-kpph,ph1);
          settable(t2,-kpph,ph2);
          settable(t3,-kpph,ph3);
          settable(t4,-kpph,ph4);
          settable(t5,-kpph,ph5);
          }


          getelem(t1, ct, v1);
```

```
		getelem(t2, ct, v2);
		getelem(t3, ct, v3);
		getelem(t4, ct, v4);
		getelem(t5, ct, oph);


/*****/

/*Check that sw1 is an integer submultiple of sw */
  if (fabs((sw/sw1)-(double)((int)((sw/sw1)+0.5)))> 0.01)
  {
    text_message("sw1 should be an integer submultiple of sw\n");
  }
/*Check that power deposition in gradient coil is not excessive*/
  if ( (gzlvl2*gzlvl2*selpw/(gradstepsz*gradstepsz))> 0.01) /* maximum 1% of full dissipation */
  {
    abort_message("Slice select gradient gzlvl2 is dangerously high\n");
  }
  if ( (gzlvl1*gzlvl1*gt1*3.0/(gradstepsz*gradstepsz))> 0.01) /* maximum 1% of full dissipation */
  {
    abort_message("Slice select gradient gzlvl2 is dangerously high\n");
  }
  if ( (hsglvl*hsglvl*hsgt*4.0/(gradstepsz*gradstepsz))> 0.01) /* maximum 1% of full dissipation */
  {
    abort_message("Slice select gradient gzlvl2 is dangerously high\n");
  }

if (pw > 0.00005)
{
abort_message("pw is too long..\n");
}

if (tpwr>59.0)
{
abort_message("tpwr is too high..\n");
}

if (gt1>0.005)
{
abort_message("gt1 is too long..\n");
}
if ((selpw>0.025) && (selpwr>29))
{
abort_message("check parameters!!..\n");
}

if (deltat1<0.0)
{
 deltat1=0.0;
}

if (tau1<1.0e-12 && tau1>-1.0e-12) tau1=0.0;


/* BEGIN ACTUAL PULSE SEQUENCE CODE */
status(A);

delay(0.1);
		obspower(tpwr);
if (sspul[A] == 'y')
{
		if ( (lkgate_flg[0] == 'y') || (lkgate_flg[0] == 'k') )  lk_hold();
		delay(0.001);
		zgradpulse(hsglvl*0.7,hsgt);
	rgpulse(pw,zero,rof1,rof1);
		zgradpulse(hsglvl,hsgt);
		delay(0.1);
}
		delay(0.1);
		if ( (lkgate_flg[0] == 'y') || (lkgate_flg[0] == 'k') )  lk_sample();
		delay(d1);
```

```
        if ( (lkgate_flg[0] == 'y') || (lkgate_flg[0] == 'k') )  lk_hold();
        delay(0.001);



status(B);

        rgpulse(pw,v1,rof1,rof1);



        if (tau1>=0.0)

        {

                if (d2==0)

                {
                        delay(tau1);
                }
                else
                {
                        delay(0.25/sw1);
                }

                zgradpulse(gzlvl1,gt1);
                delay(gstab);
                rgpulse(pw*2.0,v2,rof1,rof1);
                zgradpulse(gzlvl1,gt1);
                delay(gstab);

                if (d2==0)

                {
                delay(tau1);
                }
                else
                {
                delay(0.25/sw1);
                }

                delay(deltat1);

                delay(tau2-20.0e-6);
                delay(gstab);
                zgradpulse(gzlvl3,gt3);
                delay(gstab*0.2);
                obspower(selpwr);
                delay(gstab*0.8);
                rgradient('z',gzlvl2);
                delay(20.0e-6);
                shaped_pulse(selshape,selpw,v3,rof1,rof1);
                delay(20.0e-6);
                rgradient('z',0.0);
                delay(gstab);
                zgradpulse(gzlvl3,gt3);
                delay(gstab);
                obspower(tpwr);
                delay(tau2-20.0e-6);

                delay(droppts/sw);
                zgradpulse(gzlvl4,gt4);
                delay(gstab);
                rgpulse(pw*2.0,v4,rof1,rof1);
                zgradpulse(gzlvl4,gt4);
                delay(gstab);
                delay(deltat1);


        }
```

```
else

{
        if (d2==0)

        {

        }
        else
        {
                delay(0.25/sw1);
        }

        zgradpulse(gzlvl1,gt1);
        delay(gstab);
        rgpulse(pw*2.0,v2,rof1,rof1);
        zgradpulse(gzlvl1,gt1);
        delay(gstab);

        if (d2==0)

        {

        }
        else
        {
                delay(0.25/sw1);
        }

        delay(deltat1);

        if (d2==0)

        {
                delay(tau2a-20.0e-6);
        }
        else
        {
                delay(tau2-20.0e-6);
        }

        delay(gstab);
        zgradpulse(gzlvl3,gt3);
        delay(gstab*0.2);
        obspower(selpwr);
        delay(gstab*0.8);
        rgradient('z',gzlvl2);
        delay(20.0e-6);
        shaped_pulse(selshape,selpw,v3,rof1,rof1);
        delay(20.0e-6);
        rgradient('z',0.0);
        delay(gstab);
        zgradpulse(gzlvl3,gt3);
        delay(gstab);
        obspower(tpwr);

        if (d2==0)

        {
                delay(tau2a-20.0e-6);
        }
        else
        {
                delay(tau2-20.0e-6);
        }


        delay(droppts/sw);
```

```
                    if (d2==0)

                    {
                              delay(tau1a);
                    }
                    else
                    {

                    }

                    zgradpulse(gzlvl4,gt4);
                    delay(gstab);
                    rgpulse(pw*2.0,v4,rof1,rof1);
                    zgradpulse(gzlvl4,gt4);
                    delay(gstab);

                    if (d2==0)

                    {
                              delay(tau1a);
                    }
                    else
                    {

                    }

                    delay(deltat1);




          }


delay(rof2-rof1);

/* detection */
status(C);

}
```