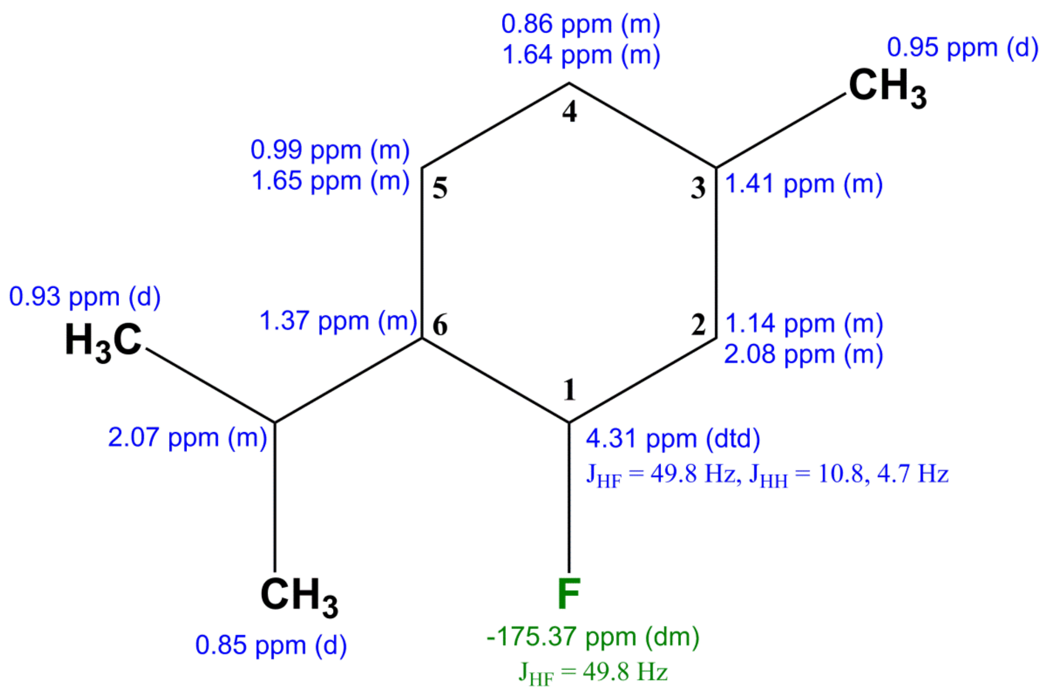
Supplemental Information for Comparison of ^1H-^{19}F Two-Dimensional NMR Scalar Coupling Correlation Pulse Sequences

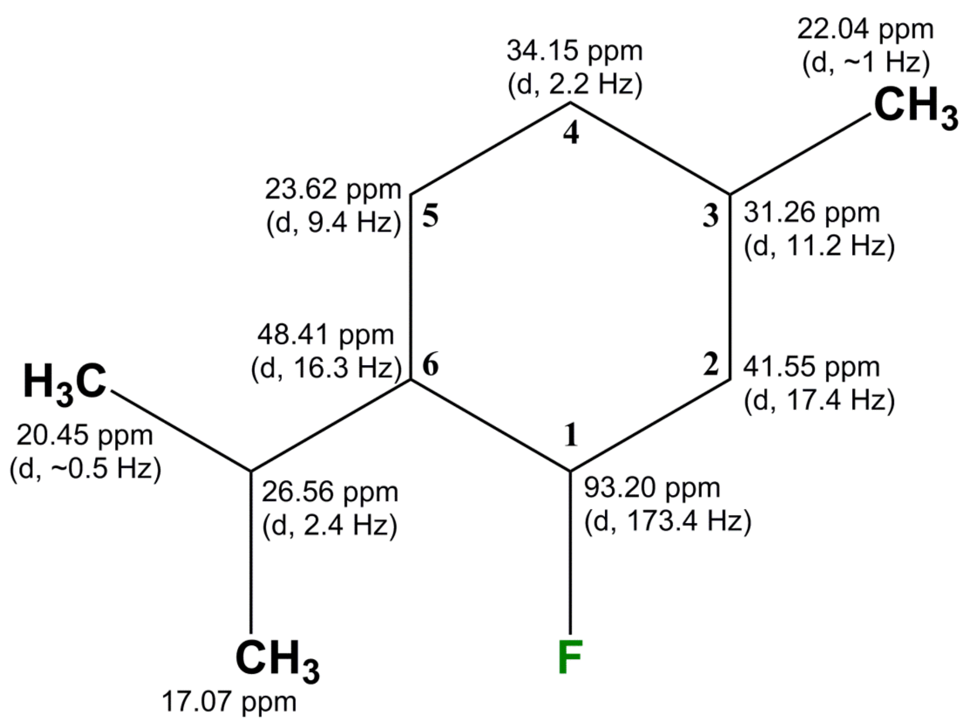
Alexander A. Marchione*, Rebecca J. Dooley, Breanna Conklin
DuPont Central Research and Development

* alexander.a.marchione@dupont.com

Multinuclear Assignment of 2-fluoro-4-methyl-1-(1-methylethyl) cyclohexane ("fluoromenthol")

Spectra acquired in chloroform-d at 30 °C, referenced v. internal tetramethylsilane $\delta = 0$ (^{1}H, ^{13}C) and internal fluorotrichloromethane $\delta = 0$ (^{19}F). ^{13}C spectrum acquired with broadband ^{1}H decoupling; doublets stem from $^{n}J_{FC}$ scalar coupling (n = 1 – 4). The stereochemistry of the analyte was not determined; the reagent in the deoxofluorination was (±)-menthol.





Pulse sequences discussed in the manuscript

The following pulse sequences were successfully compiled and implements on a Varian VNMRS spectrometer running VnmrJ software, version 3.2.

The hetero-COSY is loosely based on a pulse sequence furnished by R. Crouch, Agilent Inc., in 2011. The authors added gradient pulses and changed the pulse phase cycles.

The HETCOR is loosely based on the FH_gHETCOR sequence found in VnmrJ, version 3.2 (Agilent Inc.). An adiabatic inversion pulse in ^{19}F was added, and the pulse phase cycles were modified.

The HMQC and HMBC are loosely based on the gHMQC sequence found in VnmrJ, version 3.2 (Agilent Inc.). An adiabatic inversion pulse in ^{19}F was added, and the pulse phase cycles were modified. π pulses in the decoupler channel were removed. The gradient ratios were modified, and gradients were placed inside the t_1 evolution period in these magnitude-mode experiments. In a variant of the experiment, a strong coupling filter was added. An extra filtering $\pi/2$ pulse was added to the decoupler channel for the HMBC.

The HSQC is loosely based on the gHSQC sequence found in VnmrJ, version 3.2 (Agilent Inc.). Adiabatic inversion pulses were added to the observe channel for ^{19}F-observe experiments. The pulse phase cycles were modified. In a variant of the experiment, a strong coupling filter was added.

Hetero-COSY, cf. Figure 1A

```
/*HF hetero-COSY - 1H-19F (either nucleus set to observe) gradient hetero-
COSY experiment for Varian VNMRS spectrometer

Gradients per Macheteau et al. J. Fluor. Chem. 2000, 104, 149-154.

Loosely based on non-gradient sequence furnished by R. Crouch, Agilent
Inc., in 2011.

To use setup macro, compile this sequence from HF_gCOSY_MRC2014.c

A. Marchione, R. Dooley, B. Conklin Oct. 2013*/

#include <standard.h>


static int ph1[2] = {1, 3},
           ph2[2] = {0, 2};

pulsesequence()
{
      double      pwxlvl, pwx, phase, hsglvl, hsgt, gradratio, gzlvl1, gt1,
gzlvl2, gt2, gstab, pplvl, pp, EDratio;

            char    PFGflg[MAXSTR],
            dn[MAXSTR],
            sspul[MAXSTR];

      pwxlvl = getval("pwxlvl");
      pwx    = getval("pwx");
      pplvl = getval("pplvl");
      pp = getval("pp");
      phase  = getval("phase");
      hsglvl = getval("hsglvl");
      hsgt = getval("hsgt");
      gzlvl1 = getval("gzlvl1");
      gzlvl2 = getval("gzlvl2");
      gt1 = getval("gt1");
      gt2 = getval("gt2");
      gstab = getval("gstab");
      getstr("sspul",sspul);


  assign(ct,v17);

      settable(t1,2,ph1);
      settable(t2,2,ph2);

      getelem(t1,v17,v1);
      getelem(t2,v17,oph);



   if (phase1 == 2)
```

```
     incr(v1);

       status(A);


   if (sspul[0] == 'y')
    {
          zgradpulse(hsglvl,hsgt);
          rgpulse(pw,zero,rof1,rof1);
          zgradpulse(hsglvl,hsgt);
    }
        delay(d1);

    decpower(pplvl);

       status(B);
       decrgpulse(pp,zero,rof1,1e-6);
       delay(d2);
       zgradpulse(gzlvl1,gt1);
       delay(gstab);
       simpulse(pw,pp,zero,v1,1E-06,rof2);
       zgradpulse(gzlvl2,gt2);
       delay(gstab);
         decpower(dpwr);
       status(C);
}
```

HETCOR (cf. Figure 1B)

```
/* H-F HETCOR - H-F (either nucleus set to observe) gradient HETCOR
experiment for Varian VNMRS spectrometer.

Magnitude mode.

In case of 1H observe, shaped 180 pulse is set to "hard", and sequence
functions as basic gHETCOR.

To use setup macro, compile this sequence from HF_gHETCORAD_MRC2014.c

A. Marchione, R. Dooley, B. Conklin Oct. 2013*/


#include <standard.h>

static int ph1[ 4] = {0, 0, 0, 0},
        ph2[ 4] = {0, 2, 1, 3},
          ph3[ 8] = {0, 0, 0, 0, 2, 2, 2, 2},
        ph4[32] = {0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1,
                     2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3},
          ph5[16] = {0, 2, 0, 2, 3, 1, 3, 1, 2, 0, 2, 0, 1, 3, 1, 3};

pulsesequence()
{
      double     gzlvlE = getval("gzlvlE"),
                gtE = getval("gtE"),
                EDratio = getval("EDratio"),
            gstab = getval("gstab"),
            hsglvl = getval("hsglvl"),
            hsgt = getval("hsgt"),
            satdly = getval("satdly"),
                pwxlvl = getval("pwxlvl"),
            pwx = getval("pwx"),
                jFH = getval("jFH"),
            /* inserted following 2 lines */
            pw180 = getval("pw180"),
            tpwr180 = getval("tpwr180"),
                dly1, dly2, phase;
      char  sspul[MAXSTR],
            pw180adR[MAXSTR],
            satmode[MAXSTR];

      getstr("sspul",sspul);
      getstr("pw180adR",pw180adR);
        dly1 = 1.0 / (2.0*jFH);
        dly2 = 1.0 / (3.0*jFH);

      settable(t1,4,ph1);
      settable(t2,4,ph2);
      settable(t3,8,ph3);
        settable(t4,32,ph4);
        settable(t5,16,ph5);
```

```
      getelem(t1,ct,v1);
      getelem(t2,ct,v2);
      getelem(t3,ct,v3);
        getelem(t4,ct,v4);
        getelem(t5,ct,oph);

       decpower(pwxlvl);
status(A);

      if (sspul[0] == 'y')
      {
            zgradpulse(hsglvl,hsgt);
            rgpulse(pw,zero,rof1,rof1);
                 decrgpulse(pwx,zero,rof1,rof1);
            zgradpulse(hsglvl,hsgt);
      }

      delay(d1);

status(B);
      decrgpulse(pwx, v1, rof1, rof1);
        if (d2/2.0==0.0) delay(d2);
        else delay(d2/2.0-2*rof1-pw);
/*        rgpulse(2.0*pw,v3,rof1,rof1);
INSERT ADIABATIC REFOCUSING 180 */
      obspower(tpwr180);
      shaped_pulse(pw180adR, pw180*1E-06, v3, rof1, rof1);
      obspower(tpwr);
        if (d2/2.0==0.0) delay(d2);
        else delay(d2/2.0-2*rof1-pw);
        delay(dly1 - gtE-gstab);
      zgradpulse(gzlvlE*EDratio,gtE);
        delay(gstab);
        decrgpulse(pwx,v2,rof1,rof1);
      rgpulse(pw, v4, rof1, rof1);
      zgradpulse(-1*gzlvlE,gtE);
        decpower(dpwr);
      delay(dly2 - gtE);
status(C);
}
```

[1]H-observe HMQC (cf. Figure 1c)

```
/* gradient HMQC customized for H observe, F decouple experiments - simple
H90-F90-H180-F90 sequence.

Gradient ratios should be 1 : 3 : 0.764

To use setup macro, compile this sequence from HF_gHMQC_MRC2014.c

A. Marchione, R. Dooley, B. Conklin, Oct. 2013
*/


#include <standard.h>

static int     ph1[2] = {0,2},
               ph2[4] = {0,0,2,2},
               ph3[4] = {0,2,2,0};

pulsesequence()
{
  double gtau,
         gzlvl1 = getval("gzlvl1"),
         gt1    = getval("gt1"),
       gzlvl2 = getval("gzlvl2"),
       gt2  = getval("gt2"),
         gzlvl3 = getval("gzlvl3"),
         gt3    = getval("gt3"),
         gstab  = getval("gstab"),
       pwxlvl = getval("pwxlvl"),
       pwx    = getval("pwx"),
       tau    = 1/(2*(getval("j1xh"))),
       hsglvl = getval("hsglvl"),
       hsgt   = getval("hsgt");
  char   sspul[MAXSTR];

  getstr("sspul",sspul);

  gtau =  2*gstab + 2*GRADIENT_DELAY;

  if (tau < (gt3+gstab))
  {
    text_error("tau must be greater than gt3+gstab\n");
    psg_abort(1);
  }

  settable(t1,2,ph1);
  settable(t2,4,ph2);
  settable(t3,4,ph3);

  getelem(t1,ct,v1);
  getelem(t2,ct,v2);
  getelem(t3,ct,oph);

  status(A);
```

```
  decpower(pwxlvl);

  if (sspul[0] == 'y')
  {
    zgradpulse(hsglvl,hsgt);
    rgpulse(pw,zero,rof1,rof1);
    zgradpulse(hsglvl,hsgt);
  }

  delay(d1);

  status(B);

  rgpulse(pw,zero,rof1,rof1);
  delay(tau - rof1 - (2*pw/PI));
  decrgpulse(pwx,v1,rof1,rof1);
// T1 evolution
  delay(d2/2);
  zgradpulse(gzlvl1,gt1);
  delay(gstab - rof1 - pw);
  rgpulse(2.0*pw,zero,rof1,rof1);
  delay(gstab - rof1 - pw);
  zgradpulse(gzlvl2,gt2);
  delay(d2/2);

  decrgpulse(pwx,v2,rof1,rof1);
  zgradpulse(gzlvl3,gt3);
  delay(gstab);
  delay(tau - gt3 - gstab);
  decpower(dpwr);

  status(C);
}
```

1H-observe HMBC (cf. Figure 1e)

```
/* gHMBC for H observe, F decouple.  As gHMQC, with added dec 90 for low
pass filter.  Magnitude mode.

Gradient ratios should be 1 : 3 : 0.764

To use setup macro, compile this sequence from HF_gHMQC_filter_MRC2014.c


A. Marchione, R. Dooley, B. Conklin, Oct. 2013
*/


#include <standard.h>

static int     ph1[2] = {0,2};

pulsesequence()
{
  double gtau,
         gzlvl1 = getval("gzlvl1"),
         gt1    = getval("gt1"),
       gzlvl2 = getval("gzlvl2"),
       gt2  = getval("gt2"),
         gzlvl3 = getval("gzlvl3"),
         gt3    = getval("gt3"),
         gstab  = getval("gstab"),
       pwxlvl = getval("pwxlvl"),
       pwx    = getval("pwx"),
       tau    = 1/(2*(getval("j1xh"))),
       taufilter = 1/(2*(getval("largeJfilter"))),
       hsglvl = getval("hsglvl"),
       hsgt   = getval("hsgt");
  char   sspul[MAXSTR];

  getstr("sspul",sspul);

  gtau =  2*gstab + 2*GRADIENT_DELAY;

  if (tau < (gt3+gstab))
  {
    text_error("tau must be greater than gt3+gstab\n");
    psg_abort(1);
  }

  settable(t1,2,ph1);

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


  status(A);
  decpower(pwxlvl);
```

```
  if (sspul[0] == 'y')
  {
    zgradpulse(hsglvl,hsgt);
    rgpulse(pw,zero,rof1,rof1);
    zgradpulse(hsglvl,hsgt);
  }


  delay(d1);

  status(B);

  rgpulse(pw,zero,rof1,rof1);
//Following two lines filter undesired J
  delay(taufilter - rof1 - (2*pw/PI));
  decrgpulse(pwx,zero,rof1,rof1);

  delay(tau - rof2 - (2*pwx/PI));
  decrgpulse(pwx,v1,rof1,rof1);

// T1 evolution
  delay(d2/2);
  zgradpulse(gzlvl1,gt1);
  delay(gstab - rof1 - pw);
  rgpulse(2.0*pw,zero,rof1,rof1);
  delay(gstab - rof1 - pw);
  zgradpulse(gzlvl2,gt2);
  delay(d2/2);


  decrgpulse(pwx,zero,rof1,rof1);
  zgradpulse(gzlvl3,gt3);
  delay(gstab);
  delay(tau - gt3 - gstab);
  decpower(dpwr);

  status(C);
}
```

1H-observe HSQC (cf. Figure 1d)

```
/* Gradient gHSQC, H observe, F decouple, with adiabatic inversion pulses
in F.  Magnitude mode.

Gradient pulses should be in the ratio of 1:0.941.

To use setup macro, compile this sequence from HF_gHSQCAD_MRC2014.c

A. Marchione, R. Dooley, B. Conklin, Oct. 2013
*/

#include <standard.h>

static int ph1[4]  = {1,1,3,3},
           ph2[2]  = {0,2},
           ph3[8]  = {0,0,0,0,2,2,2,2},
           ph4[4] = {0,0,2,2},
           ph5[4] = {0,2,2,0};

pulsesequence()

{
   double   hsglvl = getval("hsglvl"),
            hsgt = getval("hsgt"),
          gzlvl1     = getval("gzlvl1"),
            gt1        = getval("gt1"),
            gzlvl3     = getval("gzlvl3"),
            gt3        = getval("gt3"),
            gstab      = getval("gstab"),
            EDratio    = getval("EDratio"),
          tpwr180    = getval("tpwr180"),
          pw180      = getval("pw180"),
            pwx180 = getval("pwx180"),
            pwxlvl180 = getval("pwxlvl180"),
            pwx180r = getval("pwx180r"),
            pwxlvl180r = getval("pwxlvl180r"),
            tau,
            evolcorr,
          taug,
          mult = getval("mult"),
          null = getval("null");
   char        pw180ad[MAXSTR], pw180adR[MAXSTR], sspul[MAXSTR],
pwx180ad[MAXSTR],
          pwx180adR[MAXSTR],pwx180ref[MAXSTR];

  getstr("pwx180ad", pwx180ad);
  getstr("pwx180adR", pwx180adR);
  getstr("sspul", sspul);


   tau  = 1/(4*(getval("j1xh")));
  settable(t1, 4, ph1);
  settable(t2, 2, ph2);
  settable(t3, 8, ph3);
```

```
  settable(t4, 4, ph4);
  settable(t5, 4, ph5);

  mod2(ct, v12);
  incr(v12);
  modn(ct,8,v13);
  incr(v13);
  mod4(ct,v14);
  incr(v14);
  mod4(ct,v15);
  incr(v15);

  getelem(t2, v12, v2);
  getelem(t3, v13, v3);
  getelem(t4, v14, v4);
  getelem(t5, v15, oph);



   status(A);

   obspower(tpwr);

   delay(5.0e-5);
      if (sspul[0] == 'y')
      {
        zgradpulse(hsglvl,hsgt);
        rgpulse(pw,zero,rof1,rof1);
        zgradpulse(hsglvl,hsgt);
      }
        delay(d1);

   decpower(pwxlvl);

    status(B);

//INEPT

     rgpulse(pw,zero,rof1,rof1);
     delay(tau - (pw*PI/2) - rof1);
     decpower(pwxlvl180);
     decshaped_pulse(pwx180adR, pwx180, v3, rof1, 1E-06);
     decpower(pwxlvl);
     rgpulse(pw*2,zero,rof1,rof1);
     delay(tau - (pwx*PI) - rof1);
     simpulse(pw,pwx,one,v2,rof1, 2.0E-06);

// T1 evolution

    delay(d2/2);
    rgpulse(2*pw,zero,2.0e-6,2.0e-6);
    delay(d2/2);

// Reverse INEPT
```

```
    zgradpulse(gzlvl1, gt1);
    delay(gstab);
    simpulse(pw, pwx, one, v4, rof1, rof1);
     delay(tau - (pw*PI) - rof1);
     decpower(pwxlvl180);
     decshaped_pulse(pwx180adR, pwx180, v3, rof1, 1E-06);
     decpower(dpwr);
     rgpulse(pw*2,zero,rof1,rof1);
      zgradpulse(gzlvl1/EDratio, gt1);
     delay(tau - gt1-(pw*PI) - rof1);

   status(C);
}
```

^{1}H-observe HSQC with strong coupling filter (cf. Figure 1d)

```
/* Gradient gHSQC, H observe, F decouple, with adiabatic inversion pulses
in F.  Magnitude mode.
Gradient pulses should be in the ratio of 1:0.941.

added echo filter for strong coupling per Kover and Batta J. Magn. Reson.
1999, 138, 89-97.

To use setup macro, compile this sequence from HF_gHSQCAD_filter_MRC2014.c

A. Marchione,R. Dooley, B. Conklin, Oct. 2013
*/

#include <standard.h>

static int ph1[4]  = {1,1,3,3},
           ph2[2]  = {0,2},
           ph3[8]  = {0,0,0,0,2,2,2,2},
           ph4[4] = {0,0,2,2},
           ph5[8] = {0,2,2,0,2,0,0,2};

pulsesequence()

{

   double   tau,taug,phase,
          gzlvl1     = getval("gzlvl1"),
            gt1        = getval("gt1"),
            gzlvl3     = getval("gzlvl3"),
            gt3        = getval("gt3"),
            gstab      = getval("gstab"),
            mult       = getval("mult"),
            hsglvl     = getval("hsglvl"),
            hsgt       = getval("hsgt"),
          tpwr180    = getval("tpwr180"),
          pw180      = getval("pw180"),
            pwx180 = getval("pwx180"),
            pwxlvl180 = getval("pwxlvl180"),
          EDratio    = getval("EDratio");

   char        pw180ad[MAXSTR], pw180adR[MAXSTR], sspul[MAXSTR],
          pwx180ad[MAXSTR],pwx180ref[MAXSTR];

  getstr("pwx180ad", pwx180ad);
  getstr("sspul", sspul);

  tau = 1 / (4*(getval("j1xh")));

  settable(t1, 4, ph1);
  settable(t2, 2, ph2);
  settable(t3, 8, ph3);
  settable(t4, 4, ph4);
  settable(t5, 8, ph5);
```

```
  mod2(ct, v12);
  incr(v12);
  modn(ct,8,v13);
  incr(v13);
  mod4(ct,v14);
  incr(v14);
  modn(ct,8,v15);
  incr(v15);

  getelem(t2, v12, v2);
  getelem(t3, ct, v3);
  getelem(t4, v14, v4);
  getelem(t5, ct, oph);

status(A);

   obspower(tpwr);

      if (sspul[0] == 'y')
      {
        zgradpulse(hsglvl,hsgt);
        rgpulse(pw,zero,rof1,rof1);
        zgradpulse(hsglvl,hsgt);
      }


   delay(d1);


   decpower(pwxlvl);

  status(B);

// INEPT

      rgpulse(pw,zero,rof1,rof1);
      delay(tau - (pw*PI/2) - rof1);
      decpower(pwxlvl180);
      decshaped_pulse(pwx180ad, pwx180, v3, rof1, 1E-06);
      decpower(pwxlvl);
      rgpulse(pw*2,zero,rof1,rof1);
      delay(tau - (pwx*PI) - rof1);
      simpulse(pw,pwx,one,v2,rof1, 2.0E-06);

// T1 evolution

    delay(d2 / 2);
    rgpulse(2*pw,zero,2.0e-6,2.0e-6);
    delay(d2 / 2);

// Gradient selection and reverse INEPT

    zgradpulse(gzlvl1, gt1);
    delay(gstab);
```

```
    simpulse(pw, pwx, one, v4, rof1, rof1);
     delay(tau - (pw*PI) - rof1);
     decpower(pwxlvl180);
     decshaped_pulse(pwx180ad, pwx180, v3, rof1, 1E-06);
     rgpulse(pw*2,zero,rof1,rof1);
      zgradpulse(gzlvl1/EDratio, gt1);
     delay(tau - gt1-(pw*PI) - rof1);
     decpower(pwxlvl);

// Strong coupling filter
    delay(tau*2);
     rgpulse(pw*2,zero,rof1,rof1);
    decrgpulse(pwx, zero, rof1, rof1);
    decrgpulse(pwx, v3, rof1, rof1);
    delay(tau*2);
    decpower(dpwr);

  status(C);
}
```

^{19}F-observe HMQC (cf. Figure 1c)

```
/* F-observe gradient HMQC, adiabatic 19F inversion pulse, magnitude mode

Customized for F observe, H decouple experiments - simple F90-H90-F180-H90
sequence, adiabatic 180 for 19F created on-the-fly

To use setup macro, compile this sequence from FH_gHMQCAD_MRC2014.c

Gradient ratios should be 1 : 3 : 2.251

A. Marchione, R. Dooley, B. Conklin, Oct. 2013
*/

#include <standard.h>

static int     ph1[2] = {0,2},
               ph2[4] = {0,0,2,2},
               ph3[4] = {0,2,2,0};

pulsesequence()
{
  double gtau,
         gzlvl1 = getval("gzlvl1"),
         gt1    = getval("gt1"),
       gzlvl2 = getval("gzlvl2"),
       gt2  = getval("gt2"),
         gzlvl3 = getval("gzlvl3"),
         gt3    = getval("gt3"),
         gstab  = getval("gstab"),
       pwxlvl = getval("pwxlvl"),
       pwx    = getval("pwx"),
       tau    = 1/(2*(getval("j1xh"))),
       hsglvl = getval("hsglvl"),
       hsgt   = getval("hsgt"),
       pw180 = getval("pw180"),
       tpwr180 = getval("tpwr180"),
       satdly = getval("satdly");
  char   pw180adR[MAXSTR],
       sspul[MAXSTR];

  getstr("sspul",sspul);
  getstr("pw180adR",pw180adR);

  gtau =  2*gstab + 2*GRADIENT_DELAY;

  if (tau < (gt3+gstab))
  {
    text_error("tau must be greater than gt3+gstab\n");
    psg_abort(1);
  }

  settable(t1,2,ph1);
  settable(t2,4,ph2);
  settable(t3,4,ph3);
```

```
  assign(zero,v4);
  getelem(t1,ct,v1);
  getelem(t2,ct,v2);
  getelem(t3,ct,oph);

  status(A);
  decpower(pwxlvl);

  if (sspul[0] == 'y')
  {
    zgradpulse(hsglvl,hsgt);
    rgpulse(pw,zero,rof1,rof1);
    zgradpulse(hsglvl,hsgt);
  }

  delay(d1);


  status(B);

  rgpulse(pw,zero,rof1,rof1);
  delay(tau - rof1 - (2*pw/PI));
  decrgpulse(pwx,v1,rof1,rof1);

// T1 evolution period
  delay(d2/2);
  zgradpulse(gzlvl1,gt1);
  delay(gstab - rof1 - pw);
/* Replace following line, inserting adiabatic 180
  rgpulse(2.0*pw,zero,rof1,rof1); */
  obspower(tpwr180);
  shaped_pulse(pw180adR, pw180*1E-06, zero, rof1, rof1);
  obspower(tpwr);
  delay(gstab - rof1 - pw);
  zgradpulse(gzlvl2,gt2);
  delay(d2/2);

  decrgpulse(pwx,v2,rof1,rof1);
  zgradpulse(gzlvl3,gt3);
  delay(gstab);
  delay(tau - gt3 - gstab);
  decpower(dpwr);

  status(C);
}
```

[19]F-observe HSQC (cf. Figure 1d)

```
/* gHSQCAD - Gradient Selected phase-sensitive HSQC, F obs, H decouple
            Adiabatic F inversion
            Magnitude mode

To use setup macro, compile this sequence from FH_gHSQCAD_MRC2014.c

A. Marchione, R. Dooley, B. Conklin, Oct. 2013
*/


#include <standard.h>

static int ph1[4]  = {1,1,3,3},
           ph2[2]  = {0,2},
           ph3[8]  = {0,0,0,0,2,2,2,2},
           ph4[4] = {0,0,2,2},
           ph5[4] = {0,2,2,0};

pulsesequence()

{

   double   tau,taug,phase,
          gzlvl1     = getval("gzlvl1"),
            gt1        = getval("gt1"),
            gzlvl3     = getval("gzlvl3"),
            gt3        = getval("gt3"),
            gstab      = getval("gstab"),
            mult       = getval("mult"),
            hsglvl     = getval("hsglvl"),
            hsgt       = getval("hsgt"),
          tpwr180    = getval("tpwr180"),
          pw180      = getval("pw180"),
          EDratio    = getval("EDratio");

   char        pw180ad[MAXSTR], sspul[MAXSTR], nullflg[MAXSTR],
          pwx180ref[MAXSTR];

  getstr("pw180ad", pw180ad);
  getstr("sspul", sspul);
  getstr("nullflg", nullflg);

  tau = 1 / (4*(getval("j1xh")));

  settable(t1, 4, ph1);
  settable(t2, 2, ph2);
  settable(t3, 8, ph3);
  settable(t4, 4, ph4);
  settable(t5, 4, ph5);

  mod2(ct, v12);
  incr(v12);
  modn(ct,8,v13);
```

```
  incr(v13);
  mod4(ct,v14);
  incr(v14);
  mod4(ct,v15);
  incr(v15);

  getelem(t2, v12, v2);
  getelem(t3, v13, v3);
  getelem(t4, v14, v4);
  getelem(t5, v15, oph);




status(A);
   obspower(tpwr);

      if (sspul[0] == 'y')
      {
        zgradpulse(hsglvl,hsgt);
        rgpulse(pw,zero,rof1,rof1);
        zgradpulse(hsglvl,hsgt);
      }

   delay(d1);
   decpower(pwxlvl);

  status(B);
// INEPT
    rgpulse(pw, zero, rof1, rof1);
    delay(tau);
    obspower(tpwr180);
    shaped_pulse(pw180ad,pw180*1E-06,zero,rof1,rof1);
    obspower(tpwr);
    decrgpulse(pwx*2,v3, rof1, rof1);
    delay(tau - pwx * 0.636);
    simpulse(pw, pwx, one, v2, rof1, rof1);

// T1 evolution

    delay(d2 / 2);
    obspower(tpwr180);
    shaped_pulse(pw180ad,pw180*1E-06,zero,rof1,rof1);
    obspower(tpwr);
    delay(d2 / 2);

// Gradient selection and reverse INEPT
    zgradpulse(gzlvl1, gt1);
    delay(gstab);
    simpulse(pw, pwx, one, v4, rof1, rof1);
    delay(tau);
    obspower(tpwr180);
    shaped_pulse(pw180ad,pw180*1E-06,zero,rof1,rof1);
```

```
    obspower(tpwr);
    decrgpulse(pwx*2, v3, rof1, rof1);
    zgradpulse(gzlvl1/EDratio, gt1);
    delay(tau - gt1 - pwx * 0.636);
    decpower(dpwr);


  status(C);
}
```

^{19}F-observe HSQC with strong coupling filter (cf. Figure 1d)

```
/*gHSQCAD - gradient HSQC, F observe, H decouple, with adiabatic 180's in
19F.  Magnitude mode.

Gradient pulses should be in the ratio of 0.941:1

added echo filter for strong coupling per Kover and Batta J. Magn. Reson.
1999, 138, 89-97.

To use setup macro, compile this sequence from FH_gHSQCAD_filter_MRC2014.c


A. Marchione, R. Dooley. B. Conklin, Oct. 2014

*/

#include <standard.h>

static int ph1[4]  = {1,1,3,3},
           ph2[2]  = {0,2},
           ph3[8]  = {0,0,0,0,2,2,2,2},
           ph4[4] = {0,0,2,2},
           ph5[8] = {0,2,2,0,2,0,0,2};

pulsesequence()

{

   double   tau,taug,phase,
         gzlvl1     = getval("gzlvl1"),
           gt1        = getval("gt1"),
           gzlvl3     = getval("gzlvl3"),
           gt3        = getval("gt3"),
           gstab      = getval("gstab"),
           mult       = getval("mult"),
           hsglvl     = getval("hsglvl"),
           hsgt       = getval("hsgt"),
         tpwr180    = getval("tpwr180"),
         pw180      = getval("pw180"),
         EDratio    = getval("EDratio");

   int      icosel, iphase;
   char        pw180ad[MAXSTR], pw180adR[MAXSTR], sspul[MAXSTR],
         pwx180adR[MAXSTR],pwx180ref[MAXSTR];

  getstr("pw180ad", pw180ad);
  getstr("pw180adR", pw180adR);
  getstr("sspul", sspul);
  tau = 1 / (4*(getval("j1xh")));
  settable(t1, 4, ph1);
  settable(t2, 2, ph2);
  settable(t3, 8, ph3);
  settable(t4, 4, ph4);
  settable(t5, 8, ph5);
```

```
  mod2(ct, v12);
  incr(v12);
  modn(ct,8,v13);
  incr(v13);
  mod4(ct,v14);
  incr(v14);
  modn(ct,8,v15);
  incr(v15);

  getelem(t2, v12, v2);
  getelem(t3, v13, v3);
  getelem(t4, v14, v4);
  getelem(t5, v15, oph);


status(A);
   obspower(tpwr);

      if (sspul[0] == 'y')
      {
        zgradpulse(hsglvl,hsgt);
        rgpulse(pw,zero,rof1,rof1);
        zgradpulse(hsglvl,hsgt);
      }

   delay(d1);

   decpower(pwxlvl);

  status(B);

// INEPT

    rgpulse(pw, zero, rof1, rof1);
    delay(tau);
    obspower(tpwr180);
    shaped_pulse(pw180ad,pw180*1E-06,zero,rof1,rof1);
    obspower(tpwr);
    decrgpulse(pwx*2,v3, rof1, rof1);
    delay(tau - pwx * 0.636);
    simpulse(pw, pwx, one, v2, rof1, rof1);

// T1 evolution

    delay(d2 / 2);
    obspower(tpwr180);
    shaped_pulse(pw180ad,pw180*1E-06,zero,rof1,rof1);
    obspower(tpwr);
    delay(d2 / 2);

// Gradient selection and reverse INEPT

    zgradpulse(gzlvl1, gt1);
```

```
    delay(gstab);
    simpulse(pw, pwx, one, v4, rof1, rof1);
    delay(tau);
    obspower(tpwr180);
    shaped_pulse(pw180ad,pw180*1E-06,zero,rof1,rof1);
    obspower(tpwr);
    decrgpulse(pwx*2, v3, rof1, rof1);
    zgradpulse(gzlvl1/EDratio, gt1);
    delay(tau - gt1 - pwx * 0.636);

// Try adding strong coupling filter per Kover 1999

    delay(tau*2);
    obspower(tpwr180);
    shaped_pulse(pw180ad,pw180*1E-06,zero,rof1,rof1);
    obspower(tpwr);
    decrgpulse(pwx, zero, rof1, rof1);
    decrgpulse(pwx, v3, rof1, rof1);
    delay(tau*2);

    decpower(dpwr);


  status(C);
}
```

Setup macros for pulse sequences

The following macros were successfully used to call in the compiled pulse sequences above on a Varian VNMRS spectrometer running VnmrJ software, version 3.2. The presence of a probe file fully and correctly populated for ^{1}H and ^{19}F is assumed.

hetero-COSY

```
/*macro to set up H-F or F-H heteronuclear COSY expt.

Macro sets fairly wide spectral windows, which can be narrowed after
setup.

A. Marchione, Oct. 2013*/

$wexp=wexp $wnt=wnt wexp='' wnt=''

par2D

string('$obsnuc')
input('What the observe nucleus? H for proton, otherwise F.'):$obsnuc

if ($obsnuc='H') then
      tn='H1'
      dn='F19'
else
      tn='F19'
      dn='H1'
endif

getparam('tpwr',tn):tpwr
getparam('pw90'):pw
getparam('pplvl',dn):pwxlvl
getparam('pp',dn):pwx

$e='0'
exists('hsglvl', 'parameter'):$e
if ($e='0') then create('hsglvl','real') endif
exists('hsgt', 'parameter'):$e
if ($e='0') then create('hsgt','real') endif
exists('gzlvl1', 'parameter'):$e
if ($e='0') then create('gzlvl1','real') endif
exists('gt1', 'parameter'):$e
if ($e='0') then create('gt1','real') endif
exists('gzlvl2', 'parameter'):$e
if ($e='0') then create('gzlvl1','real') endif
exists('gt2', 'parameter'):$e
if ($e='0') then create('gt1','real') endif
exists('sspul', 'parameter'):$e
if ($e='0') then create('sspul','string') endif
exists('EDratio', 'parameter'):$e
if ($e='0') then create('EDratio','string') endif

EDratio=(dfrq/sfrq)
```

```
if (tn='H1') then setsw(10,-0.5) endif
if (tn='F19') then setsw(5,-230) endif
if (dn='H1') then setsw1(dn,9,1) endif
if (dn='F19') then setsw1(dn,-50,-230) endif

if fn > 2048 then fn=2048 np=fn sw=sw endif

dm='nny' dmm='ccw'
refsource1='dfrq'
setdecpars

d1=2 proc='ft' fpmult='n'
ni=128

nt=4
fn1=2k phase=0 av
setLP1 gaussian

gzlvl1=10/gcal
gt1=.0015
gzlvl2=gzlvl1*EDratio
gt2=gt1
hsglvl=3.84/gcal
hsgt=0.0015

ss=8
spin=0 alock='n' wshim='n' pw=pw90
ai axis='pd'
pmode='full'
seqfil='HF_gCOSY_MRC2014' sspul='y' f1coef='1 0 0 1'
f full
hs='n' gain='y' dps
dg
```

HETCOR

```
/*
A macro to set up a gradient, F or H observe 19F-1H HETCOR.
If 19F observe, an adiabatic 180 is created by setpw180ad macro -
precision is 1 ppm

A. Marchione, Oct. 2013*/

string('$obsnuc')
string('$d')

input('What the observe nucleus? H for proton, otherwise F.'):$obsnuc

if ($obsnuc='H') then
      tn='H1'
```

```
      dn='F19'
      $d='y'
else
      tn='F19'
      dn='H1'
endif

FH_gHETCOR

if ($d='y') then
      tn='H1'
      dn='F19'
endif

getparam('tpwr',tn):tpwr
getparam('pw90'):pw
getparam('pplvl',dn):pwxlvl
getparam('pp',dn):pwx

exists('pw180adR','parameter','current'):$e
if ($e=0) then
      create('pw180adR','string')
endif

exists('pw180ad','parameter'):$e
if ($e=0) then
      create('pw180ad','string')
endif

exists('pw180','parameter'):$e
if ($e=0) then
      create('pw180','real')
endif

exists('tpwr180','parameter'):$e
if ($e=0) then
      create('tpwr180','real')
endif

exists('EDratio', 'parameter'):$e
if ($e=0) then create('EDratio','real') endif

EDratio=(sfrq/dfrq)

gzlvlE=10/gcal
gtE=0.0015

hsglvl=3.84/gcal
hsgt=0.0015

if (tn='H1') then setsw(10,-0.5) endif
if (tn='F19') then setsw(5,-230) endif
if (dn='H1') then setsw1(dn,9,1) endif
if (dn='F19') then setsw1(dn,-50,-230) endif
```

```
input('For what value of JHF should the experiment be optimized?'):jFH

real('$downfield')
real('$upfield')
real('swppm')
real('$rval')
string('$swppm')
string('$sval')

if (tn='F19') then

      input('What is the downfield limit for 19F in ppm?  Decimals
ignored.'):$downfield
      input('What is the upfield limit for 19F in ppm?  Decimals
ignored.'):$upfield

      setsw($downfield,$upfield)
      swppm=$downfield-$upfield


      if swppm < 10 then
            write ('line3','You don\'t need an adiabatic pulse in 19F.
Switching to gHETCOR...')
            FH_gHETCOR
            return
      endif
      if swppm < 100 then
            format(swppm,2,0):$sval
            format(swppm,2,0):$rval
      else
            format(swppm,3,0):$sval
            format(swppm,3,0):$rval
      endif

      $swppm=$sval
      swppm=$rval

      setpw180ad(tn,'make','wurst2i',swppm)
      pw180adR=probe+'_F19_ad'+$swppm+'R'

      if tpwr180 > 48 then
            write('line3','The power for the adiabatic pulse is pretty
high - please check.')
      endif

else
      pw180adR='hard'
      pw180=pw*2
      tpwr180=tpwr

endif

seqfil='HF_gHETCORAD_MRC2014'
```

```
if fn > 2048 then fn=2048 np=fn sw=sw endif

dm='nny' dmm='ccw'
refsource1='dfrq'
setdecpars

d1=2 proc='ft' fpmult='n'
ni=128

nt=4
fn1=2k phase=0 av
setLP1 gaussian
f1coef='1 0 0 -1' sspul='y'

dps
```

^{1}H-observe HMQC

```
/*
A macro to set up a gradient, H-observe 1H-19F HMQC with or without a
strong coupling filter

A. Marchione, Oct. 2013 */

tn='H1' dn='F19'

gHMQC

string('$filter')
input('Should a strong coupling filter be used?  Y for yes, otherwise no
'):$filter

exists('gzlvl2','parameter','current'):$e
if ($e=0) then
      create('gzlvl2','real')
endif

exists('gt2','parameter'):$e
if ($e=0) then
      create('gt2','real')
endif

if ($filter='Y') then

      seqfil='HF_gHMQC_filter_MRC2014'
else

      seqfil='HF_gHMQC_MRC2014'
endif
```

```
input('For what value of JHF should the experiment be optimized?'):j1xh

gt1=0.001
gt2=0.001
gt3=0.001

gzlvl1=8/gcal
gzlvl2=gzlvl1*3
gzlvl3=gzlvl1*1.764

dm='nny' dmm='ccp'
setdecpars

sspul='y'

d2=2*(gt1+gstab+(pw*1E-06)+(pwx*0.637*1E-06))

setsw(10,-0.5)
setsw1(dn,-50,-230)

d1=2 proc='ft' fpmult='n'
ni=128

nt=4
fn1=2k phase=0 av
setLP1 gaussian
f1coef='1 0 0 1' sspul='y'


dps
```

1H-observe HSQC

```
/*A macro to set up a gradient H observe, F-decouple basic 19F-1H HSQC
with a adiabatic 180's in 19F, with or without a strong coupling filter.
Magnitude mode.

shape created by setpwx180ad macro - precision is 1 ppm

A. Marchione, Oct. 2013 */

tn='H1' dn='F19'
gHSQCAD

exists('pwx180adR','parameter','current'):$e
if ($e=0) then
        create('pwx180adR','string')
endif

exists('pwx180ad','parameter'):$e
if ($e=0) then
```

```
        create('pwx180ad','string')
endif

exists('pwx180','parameter'):$e
if ($e=0) then
        create('pwx180','real')
endif

exists('pwxlvl180','parameter'):$e
if ($e=0) then
        create('pwxlvl180','real')
endif

exists('gzlvl2','parameter','current'):$e
if ($e=0) then
      create('gzlvl2','real')
endif

exists('gt2','parameter'):$e
if ($e=0) then
      create('gt2','real')
endif




real('$downfield')
real('$upfield')
real('swppm')
real('$rval')
string('$swppm')
string('$sval')

input('For what value of JHF should the experiment be optimized?'):j1xh

input('What is the downfield limit for 19F in ppm?  Decimals
ignored.'):$downfield
input('What is the upfield limit for 19F in ppm?  Decimals
ignored.'):$upfield

setsw($downfield,$upfield)
swppm=$downfield-$upfield


if swppm < 10 then
      write ('line3','You don\'t need an adiabatic pulse in 19F.
Switching to gHSQC...')
      gHSQC
      return
endif
if swppm < 100 then
      format(swppm,2,0):$sval
      format(swppm,2,0):$rval
else
```

```
      format(swppm,3,0):$sval
      format(swppm,3,0):$rval
endif

$swppm=$sval
swppm=$rval


setpwx180ad(dn,'make','wurst2i',swppm)
pwx180adR=probe+'_F19_ad'+$swppm+'R'

if pwxlvl180 > 48 then
      write('line3','The power for the adiabatic pulse is pretty high -
please check.')
endif

string('$filter')
input('Should a strong coupling filter be used?  Y for yes, otherwise no
'):$filter

if ($filter='Y') then
      seqfil='HF_gHSQCAD_filter_MRC2014'
else
      seqfil='HF_gHSQCAD_MRC2014'
endif

gt1=0.001
gt2=0.001
gt3=0.001

gzlvl1=10/gcal
EDratio=sfrq/dfrq

dm='nny' dmm='ccp'
setdecpars

sspul='y'

setsw(10,-0.5)

d1=2 proc='ft' fpmult='n'
ni=128

nt=4
fn1=2k phase=0 ph
setLP1 gaussian
f1coef='1 0 0 1' sspul='y'

dps
```

^{1}H-observe HMBC

```
/*
A macro to set up a gradient, H-observe 1H-19F HMBC with or without a
strong coupling filter

A. Marchione, Oct. 2013 */

tn='H1' dn='F19'

gHMQC

exists('gzlvl2','parameter','current'):$e
if ($e=0) then
      create('gzlvl2','real')
endif

exists('gt2','parameter'):$e
if ($e=0) then
      create('gt2','real')
endif

exists('largeJfilter','parameter'):$e
if ($e=0) then
      create('largeJfilter','real')
endif

seqfil='HF_gHMBC_MRC2014'

input('For what value of JHF should the experiment be optimized?'):j1xh
input('What value of JHF should be filtered out?'):largeJfilter

gt1=0.001
gt2=0.001
gt3=0.001

gzlvl1=8/gcal
gzlvl2=gzlvl1*3
gzlvl3=gzlvl1*1.764

dm='nny' dmm='ccp'
setdecpars

sspul='y'

d2=2*(gt1+gstab+(pw*1E-06)+(pwx*0.637*1E-06))

setsw(10,-0.5)
setsw1(dn,-50,-230)

d1=2 proc='ft' fpmult='n'
ni=128

nt=4
fn1=2k phase=0 av
setLP1 gaussian
```

```
f1coef='1 0 0 1' sspul='y'

dps
```

19F-observe HMQC

```
/*
A macro to set up a gradient, F-observe 19F-1H HMQC with an adiabatic 180
in 19F
shape created by setpw180ad macro - precision is 1 ppm

A. Marchione, Oct. 2013 */

tn='F19' dn='H1'
gHMQC

exists('pw180adR','parameter','current'):$e
if ($e=0) then
      create('pw180adR','string')
endif

exists('pw180ad','parameter'):$e
if ($e=0) then
      create('pw180ad','string')
endif

exists('pw180','parameter'):$e
if ($e=0) then
      create('pw180','real')
endif

exists('tpwr180','parameter'):$e
if ($e=0) then
      create('tpwr180','real')
endif

exists('gzlvl2','parameter','current'):$e
if ($e=0) then
      create('gzlvl2','real')
endif

exists('gt2','parameter'):$e
if ($e=0) then
      create('gt2','real')
endif



real('$downfield')
```

```
real('$upfield')
real('swppm')
real('$rval')
string('$swppm')
string('$sval')

input('For what value of JHF should the experiment be optimized?'):j1xh

input('What is the downfield limit for 19F in ppm?  Decimals
ignored.'):$downfield
input('What is the upfield limit for 19F in ppm?  Decimals
ignored.'):$upfield

setsw($downfield,$upfield)
swppm=$downfield-$upfield


if swppm < 10 then
      write ('line3','You don\'t need an adiabatic pulse in 19F.
Switching to gHMQC...')
      gHMQC
      return
endif
if swppm < 100 then
      format(swppm,2,0):$sval
      format(swppm,2,0):$rval
else
      format(swppm,3,0):$sval
      format(swppm,3,0):$rval
endif

$swppm=$sval
swppm=$rval


setpw180ad(tn,'make','wurst2i',swppm)
pw180adR=probe+'_F19_ad'+$swppm+'R'

if tpwr180 > 48 then
      write('line3','The power for the adiabatic pulse is pretty high -
please check.')
endif

seqfil='FH_gHMQCAD_MRC2014'

gt1=0.001
gt2=0.001
gt3=0.001

gzlvl1=8/gcal
gzlvl2=gzlvl1*3
gzlvl3=gzlvl1*2.251

dm='nny' dmm='ccp'
```

```
setdecpars

setsw1(dn,9,1)

d1=2 proc='ft' fpmult='n'
ni=128

nt=4
fn1=2k phase=0 av
setLP1 gaussian
f1coef='1 0 0 1' sspul='y'

dps
```

19F-observe HSQC

```
/*
A macro to set up a gradient, F-observe 19F-1H HSQC with an adiabatic 180
in 19F, with or without a strong coupling filter.
shape created by setpw180ad macro - precision is 1 ppm

A. Marchione Oct. 2013*/

tn='F19' dn='H1'
gHSQC

exists('pw180adR','parameter','current'):$e
if ($e=0) then
      create('pw180adR','string')
endif

exists('pw180ad','parameter'):$e
if ($e=0) then
      create('pw180ad','string')
endif

exists('pw180','parameter'):$e
if ($e=0) then
      create('pw180','real')
endif

exists('tpwr180','parameter'):$e
if ($e=0) then
      create('tpwr180','real')
endif

exists('gzlvl2','parameter','current'):$e
if ($e=0) then
      create('gzlvl2','real')
endif
```

```
exists('gt2','parameter'):$e
if ($e=0) then
      create('gt2','real')
endif

exists('EDratio','parameter'):$e
if ($e=0) then
      create('gt2','real')
endif


real('$downfield')
real('$upfield')
real('swppm')
real('$rval')
string('$swppm')
string('$sval')

input('For what value of JHF should the experiment be optimized?'):j1xh

input('What is the downfield limit for 19F in ppm?  Decimals
ignored.'):$downfield
input('What is the upfield limit for 19F in ppm?  Decimals
ignored.'):$upfield

setsw($downfield,$upfield)
swppm=$downfield-$upfield


if swppm < 10 then
      write ('line3','You don\'t need an adiabatic pulse in 19F.
Switching to gHSQC...')
      gHSQC
      return
endif
if swppm < 100 then
      format(swppm,2,0):$sval
      format(swppm,2,0):$rval
else
      format(swppm,3,0):$sval
      format(swppm,3,0):$rval
endif

$swppm=$sval
swppm=$rval


setpw180ad(tn,'make','wurst2i',swppm)
pw180adR=probe+'_F19_ad'+$swppm+'R'
pw180ad=probe+'_F19_ad'+$swppm

if tpwr180 > 48 then
      write('line3','The power for the adiabatic pulse is pretty high -
please check.')
```

```
endif

string('$filter')
input('Should a strong coupling filter be used?  Y for yes, otherwise no
'):$filter

if ($filter='Y') then

      seqfil='FH_gHSQCAD_filter_MRC2014'
else

      seqfil='FH_gHSQCAD_MRC2014'
endif

gt1=0.001
gt2=0.001
gt3=0.001

gzlvl1=10/gcal
EDratio=sfrq/dfrq

dm='nny' dmm='ccw'
setdecpars
sspul='y'
phase=0
d1=2
ni=128
nt=8
fn1=2k
setLP1 gaussian
ss=8
ai axis='pd'
pmode='full'
f1coef='1 0 0 1'
av
dps
```