A modular library for fast prototyping of solution-state nuclear magnetic resonance experiments

Michał Górka, Wiktor Koźmiński

Biological and Chemical Research Centre, Faculty of Chemistry, University of Warsaw, Żwirki i Wigury 101, 02-089 Warsaw, Poland

Correspondence to: Wiktor Koźmiński (kozmin@chem.uw.edu.pl)

Abstract. We present a framework library (Modular Elements, ME) for the development of pulse sequences for Bruker spectrometers. It implements a two-level abstraction approach–the lower level comprises basic functional elements of pulse sequences and the higher one often-reused blocks comprising several evolution periods. The low-level abstractions reduce code duplication between variants of experiments such as hard-pulse and selective variants of individual NMR experiments. The high-level modules enable further reuse of pulse program code and aid in the construction of complex experiments. We show the library's functionality by presenting pulse programs that can be switched between standard and TROSY variants, hard and shaped pulses and can seamlessly incorporate real-time homodecoupling. Adaptability is further demonstrated in a configurable 4D NOESY program.

1 Introduction

NMR is an extraordinarily powerful and adaptable spectroscopic method, with just the solution-state variant being capable of discerning the structure and dynamics of molecules ranging in size from simple organic compounds to large protein complexes such as a proteasome (Sprangers and Kay, 2007). The variety of experimental objects and the great number of parameters that can be measured has led to the proliferation of not only general experimental schemes, such as an ^{1}H, ^{15}N HSQC (Bodenhausen and Ruben, 1980) or a HNCO (Kay et al., 1990b; Ikura et al., 1990), but also their variants and thus the pulse sequences, implementing them as computer code. As an example, for the often used HNCO experiment, the non-exhaustive list of meaningful implementation choices is: the experiment can use hard pulses or avoid saturating water using selective pulses (Schanda et al., 2006); the final transfer element can be a simple spin-echo (Palmer et al., 1991), a set of three echoes implementing a sensitivity-enhanced transfer or one of many TROSY variants (Salzmann et al., 1999b; Nietlispach, 2005), with possible optimizations (Salzmann et al., 1999a; Schulte-Herbrüggen and Sørensen, 2000); radiation damping can be suppressed with bipolar gradients (Sklenar, 1995). Even without implementing all specialised experiment variants, the standard library supplied with the TopSpin software (Bruker) contains over a thousand pulse programs.

A common problem with pulse sequences, especially in biological NMR, is thus the requirement to code multiple variants of a given sequence. If this is done in separate files (as in the TopSpin built-in library) it results in a lot of code repetition and if made using conditional statements, it can substantially complicate the structure of the file, making trouble-shooting harder. Similarly, many pulse sequences share large amounts of code, often with no or minimal changes. Because this repeated code is scattered across different sequences and variants of experiments adding new variants (using different soft pulses, adding homodecoupling) requires applying the same modification across a large part of the whole pulse sequence library, which is tedious and error-prone. It is possible to implement such a library using standard systems programming language like C or Python, but we decided to use the native programming language of the spectrometer system, since any user writing pulse sequence needs to be familiar with it and requiring knowledge of separate programming language and its tooling would be an unnecessary hurdle to adoption. Here we show that by abstracting certain functionality using the somewhat limited macro and "define" functionality built into the TopSpin software, the above-described problems can still be avoided and the code can be made more readable and easier to modify. Here, we present the Modular Elements (ME) library for Bruker spectrometers. Although the library is specific to a particular hardware vendor, the modular approach it implements is more general and can be implemented on other instruments. A previous implementation of a modular library for pulse program implementation (NMR blocks) can be found in (Zawadzka-Kazimierczuk, 2012) for Varian/Agilent spectrometers, where evolution periods such as INEPT or COS-INEPT where abstracted as C functions. Alternative approaches to a modular library include domain specific pulse program generators, like GENESIS (Yong et al., 2022) for NOAH supersequences. Specialized libraries combining custom pulse programs and various tools (Favier and Brutscher, 2019; Vallet et al., 2020; Lukavsky and Puglisi, 2001), are suitable for routine use, but have limited applicability in the prototyping of new sequences.

2 General approach to pulse sequence modularisation

We categorise the library's functionality as low- and high-level. Low-level functionality encompasses the creation of variables and functions (technically functional macros), abstracting the basic building blocks of pulse sequences like pulses, gradients and delays. A pulse function can expand to a 90 degree proton hard pulse or an HN selective excitation pulse depending on global settings. A gradient function can expand to "no operation" in standard HSQC, or a selection gradient in the gradient-selected HSQC variant, with its corresponding delay variable containing a zero or correspondingly non-zero length of time. Decoupling functions for protons and deuterium can likewise be enabled and disabled depending on whether a TROSY variant is desired and if the sample is deuterated. This functionality simplifies the writing of pulse sequences implementing multiple variants of a given NMR experiment and gives its user the ability to easily test and compare the effectiveness of the variants for a given sample and the commonization of parameters across variants enables faster optimisation.

High-level functionality is implemented as modules that are included in the pulse sequences and can be classified as general modules and specific modules. General modules implement elements common to almost all pulse sequences. The functionally most significant ones are the preparation and acquisition modules. The preparation module gives the user the option to turn on functionality such as solvent presaturation or a combination of N/C pulses and pulse field gradients for spoiling of residual magnetisation on those nuclei. The acquisition module enables switching between standard or homodecoupled data acquisition. The specific modules are abstract blocks of pulse sequence elements that appear in many pulse sequences in an almost identical form. Two main types of specific modules are proximal and distal modules, abstracting the functionality of blocks including and following first excitation (distal) and right before acquisition (proximal). Despite a large variety of possible implementations, the proximal/distal fragments differentiate variants of a pulse sequence (for example a standard hard-pulse HNCO, selective/BEST-HNCO, hard pulse and BEST TROSY-HNCO (Solyom et al., 2013)) and the actual code is usually repeatable across different sequences. HNCO, HNCACO and HNCOCA (Yang and Kay, 1999) have very similar proximal and distal parts; HN(CA)CONH and HabCabCONH (Kazimierczuk et al., 2010) have different proximal blocks, but the distal block is still very similar for all sequences listed. With the use of the low-level functionality described above a single proximal module can abstract the initial two transfer periods (first with transverse H magnetisation and second with transverse N/C magnetisation), with the choice of N/C nucleus and choice of evolved J coupling (CO in HNCO) made using define directives in the main pulse sequence. NOESY experiments are particularly susceptible to modularisation, with the NOE transfer period naturally splitting them into proximal and distal blocks. Standard 2D experiments of the HSQC, TROSY and HMQC type have thus been implemented as proximal modules, that can be used on their own as 2D experiments or included in a 3 or 4D NOESY (Kay et al., 1990a) with the chosen distal modules, which can themselves be modified 2D experiments or simpler blocks.

3 Library implementation

Description of implementation details and design choices requires a quick recapitulation of TopSpin pulse programs' language specifics. TopSpin has two types of variables: user-adjustable numbered variables (*d1*..*d63* for delays, *cnst1*...*cnst63* for floating point constants, similarly for integer constants ("loopcounters") *lN*, pulse lengths *pN*, ...) and named variables (pulses, delays and loopcounters only, also lists of various kinds), which can only be manipulated within a pulse program. Some less-documented observations on the limitations of named variables are compiled in SI. TopSpin implements limited functionality for defining text-substitution macros ("--traditional" mode of the GNU C preprocessor cpp (Stallman and GCC Developer Community, 2012)), which can be used everywhere outside a "relation" (variable value calculations using a subset of C syntax), due to their implementation as text in quotes (treated as string literals by cpp and ignored for macro expansion), though this limitation can be overcome (see the file "notes on TopSpin.txt" in the ME library). The user can provide custom option choices to a pulse program using the ZGOPNTS variable to define appropriate macros.

3.1 Low-level modularisation

3.1.1 Variables

With no user-adjustable named variables, two approaches to making them consistent across different pulse programs are possible - indirection through named variables or introducing a convention attaching constant meaning to numbered variables. Due to the limited number and type of named variables, we predominantly use the latter option (with sets of variables described in files such as delays.incl, pulse.incl, ...) with some focused use of indirection - for example, proximal type modules use *timeHX* and *timeXY* for J coupling evolution times between the H, X and Y nuclei. Default values for all such variables can by set using the me.set_parameters.py TopSpin program. For variables that don't ordinarily have calculations performed on them (pulse phases *phN*, gradient programs *gpN*) we implemented full indirection, where the user can use phFree1 or phFree3 without worrying as to which phN variables are used by other parts of a pulse program.

3.1.2 Pulses

The most important low-level abstractions are pulse functions. They are implemented using function-like macros of cpp and have the general form of nucleus_type(phase), where nucleus can be a general specifier like H/C/N or more specific like HN/HC/CA/CO and type is classified based on the desired functionality, with the main ones being: excitation (for the excitation of longitudinal magnetization), flipback (acting on transverse magnetization), refocussing, inversion (inverting longitudinal magnetization), excitation_UR and flipback_UR (implementing universal rotations). The pulse macros will have different replacement text based on global settings (usually ZGOPTNS). A proton pulse "H_excitation(ph)" will be replace by a hard pulse "p1 ph pl1" by default, but with a "-DH_SHAPED" option will instead be replaced by "p54:sp54 ph" for a selective soft pulse and the associated named variable pH_excitation will be set to have the same value as *p1* or *p54*.

Pulse programs should account for the effective evolution time during pulse (which can be as much as 1 ms for longer selective pulses) to give correctly phased spectra and optimal J coupling evolution times. This library only accounts for linear phase slope using the modelling method described in (Lescop et al., 2010), that is treating a pulse as sequence (delay, ideal pulse, delay), which accounts for the phase slope of many commonly used pulses and can be explicitly optimized for during pulse design (Gershenzon et al., 2008; Asami et al., 2018). This phase slope is compensated for using variables such as *eH_excitation*, which for the hard pulse above would be set to $\frac{2p1}{\pi}$. We assume that the flipback and flipback_UR pulses act as if they were time-reversed excitation pulses and so the effective evolution time for a flipback pulse acting on transverse magnetization is also *eH_excitation*. A H_excitation_UR pulse of phase x will give an effective time of *eH_excitation* for z magnetization, *eH_flipback* for y magnetization and *eH_excitation* + *eH_flipback* for x magnetization. By compensating delays using the above mentioned variables, the whole sequence can be switched from a hard pulse implementation to a shaped pulse version, whether to account for field inhomogeneity or perform band-selective excitation.

3.1.3 Code blocks

There are many small blocks of code that can be included/excluded in a pulse program based on a sequence variant. To limit the number of conditional statements in the main pulse program, many are defined as macros that will expand to pulse program code based on options, for example “H2O_FLIPBACK(ph2)” will be replaced by “(11:sp1 ph2):f1” in a pulse sequence with water flipback and by whitespace if using selective pulses. Similarly DECOUPLE_H_ON and DECOUPLE_H_OFF macros will turn on proton decoupling in a standard HNCO experiment but will have no effect in TROSY-HNCO.

3.2 High-level modularization

TopSpin pulse programs follow a defined sequential structure that complicates the implementation of high-level modules as individual files and, in general, is:

1) configuration and compile-time calculations
2) a "zd" or "ze" statement
3) pulse program body (pulses and delays) and real-time calculations
4) signal acquisition block
5) loop statements for scans of a FID and points of a multidimensional experiment
6) phase program definitions

3.2.1 General modules

The general modules fit into this sequential structure as follows:

1a) configuration and compile-time calculations
1b) **init. incl**
1c) configuration and compile-time calculations continued
2) a "zd" or "ze" statement
3a) real-time calculations
3b) **start.incl**
3c) pulse program body (pulses and delays) and real-time calculations
4) **end.incl**
5) loop statements for scans of a FID and points of a multidimensional experiment
6a) **phasecycles.incl**
6b) phase program definitions

The general modules have numerous conditional statements and imports evaluating the option provided in point 1) above and using the built-in ZGOPTNS variable and interact with the specific modules (this is covered below). The init.incl module provides the library’s core functionality by defining macros for functions and variable descriptions. start.incl executes the relaxation delay (with possible solvent presaturation) and optional operations, such as crushing residual C or N magnetization (gradient pulse after an excitation pulse) or inverting N magnetization before the relaxation delay in BEST-TROSY. For non-protein experiments an ASAP (Kupče and Freeman, 2007) period would be added here, but the relevant code is experimental and provided in a commented-out form due to the method's potential to damage probeheads. The end.incl module handles acquisition with the option for real-time homodecoupling - here provided with ^{13}C-GBIRD^{r,X} (Garbow et al., 1982; Haller et al., 2022) and BASHD (Brüschweiler et al., 1988; Krishnamurthy, 1997) types.

3.2.2 Specific modules

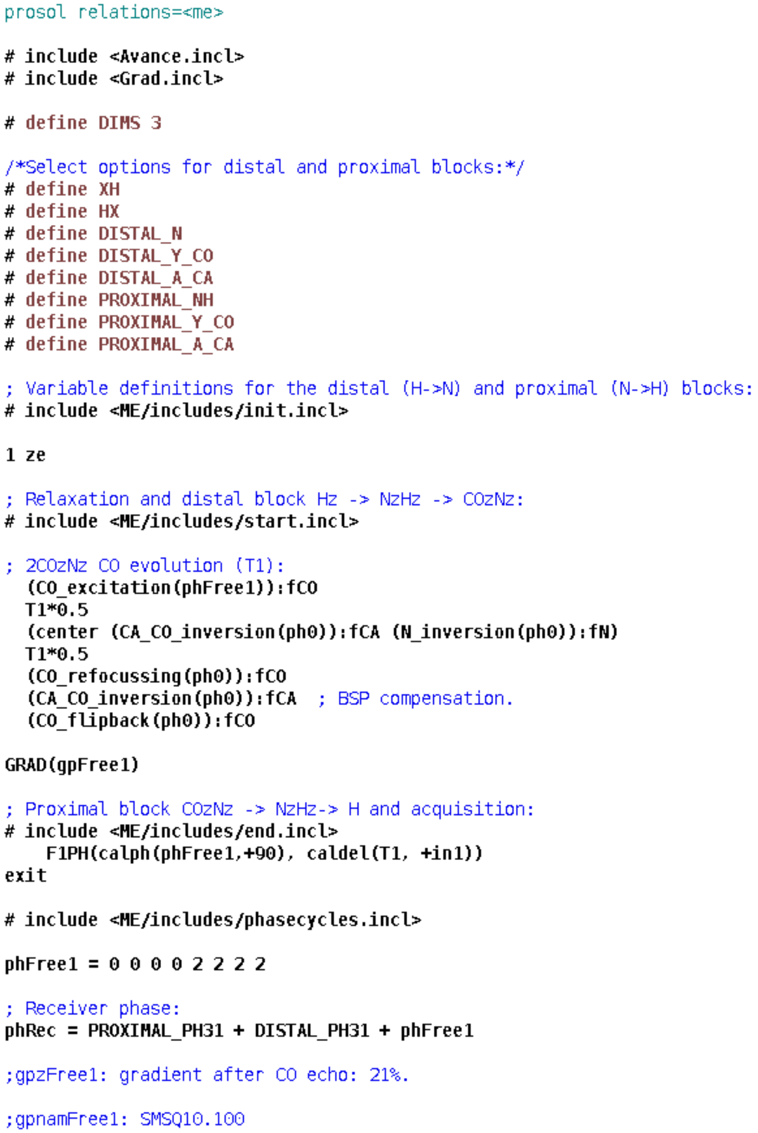
In contrast to the general modules, specific modules implement a specific form of proximal or distal block and serve to localize the relevant code in a single file. The biggest hurdle to writing self-contained modules for TopSpin is the sequential pulse program structure necessitating the separation of related code segments in the post-preprocessing file. To mitigate this problem, each module is entirely enclosed in a conditional statement with alternative conditions (an if...elif...else structure) and including the file once will only insert a selected part of the module into a file. Since the 4 general modules already perform the sequential separation of code, each of them sets the appropriate conditions (defines a macro) and imports the distal_2D.incl and proximal_2D.incl which themselves import the selected specific modules at each of the 4 positions in the pulse program. Thus, the initialization phase statements (variable declarations, some calculations, macro definitions) are included in init.incl, runtime calculations of both types of modules are included through start.incl, together with the main body (pulses and delay statements) of the distal. Similarly, the main body of the proximal module is included through the end.incl before the latter's acquisition portion. Phase cycles of both modules are inserted into a pulse program file through phasecycles.incl with some basic logic, allowing for coordinating the cycles between them if two modules are used.

For triple-resonance experiments (in the implementation limited to amide protons, but should be possible to extend to aliphatic/aromatic groups) the proximal module hx.incl and the distal module hx.incl provide the ability to compartmentalize the relatively standard blocks for both out-and-back and straight through type experiments and a more detail description in the context of a HNCO experiment is provided below. Although sub-optimal in some circumstances the library provides default 2 step phase cycles for each of the modules, leaving the implementation of 8 step and longer cycles for the central part of the program. A more detailed description of individual modules is provided in library documentation. In the supplemental we provide a detailed step-by-step description of the proximal HSQC module and the way it is used in the 2D experiment pulse program.

A specific module separate from the proximal-distal type can also be based on the same structure and either manually included in the pulse program after each general module or in a specific module itself - se.incl is module implementing the sensitivity-enhanced COS-INEPT and TROSY transfers and is imported in both the hsqc_se.incl and hx.incl modules.

4 Application examples

4.1 HNCO



```
prosol relations=<me>

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

# define DIMS 3

/*Select options for distal and proximal blocks:*/
# define XH
# define HX
# define DISTAL_N
# define DISTAL_Y_CO
# define DISTAL_A_CA
# define PROXIMAL_NH
# define PROXIMAL_Y_CO
# define PROXIMAL_A_CA

; Variable definitions for the distal (H->N) and proximal (N->H) blocks:
# include <ME/includes/init.incl>

1 ze

; Relaxation and distal block Hz -> NzHz -> COzNz:
# include <ME/includes/start.incl>

; 2COzNz CO evolution (T1):
  (CO_excitation(phFree1)):fCO
  T1*0.5
  (center (CA_CO_inversion(ph0)):fCA (N_inversion(ph0)):fN)
  T1*0.5
  (CO_refocussing(ph0)):fCO
  (CA_CO_inversion(ph0)):fCA  ; BSP compensation.
  (CO_flipback(ph0)):fCO

GRAD(gpFree1)

; Proximal block COzNz -> NzHz-> H and acquisition:
# include <ME/includes/end.incl>
    F1PH(calph(phFree1,+90), caldel(T1, +in1))
exit

# include <ME/includes/phasecycles.incl>

phFree1 = 0 0 0 0 2 2 2 2

; Receiver phase:
phRec = PROXIMAL_PH31 + DISTAL_PH31 + phFree1

;gpzFree1: gradient after CO echo: 21%.

;gpnamFree1: SMSQ10.100
```

Fig. 1. Pulse program code for the implementation of the HNCO experiment.

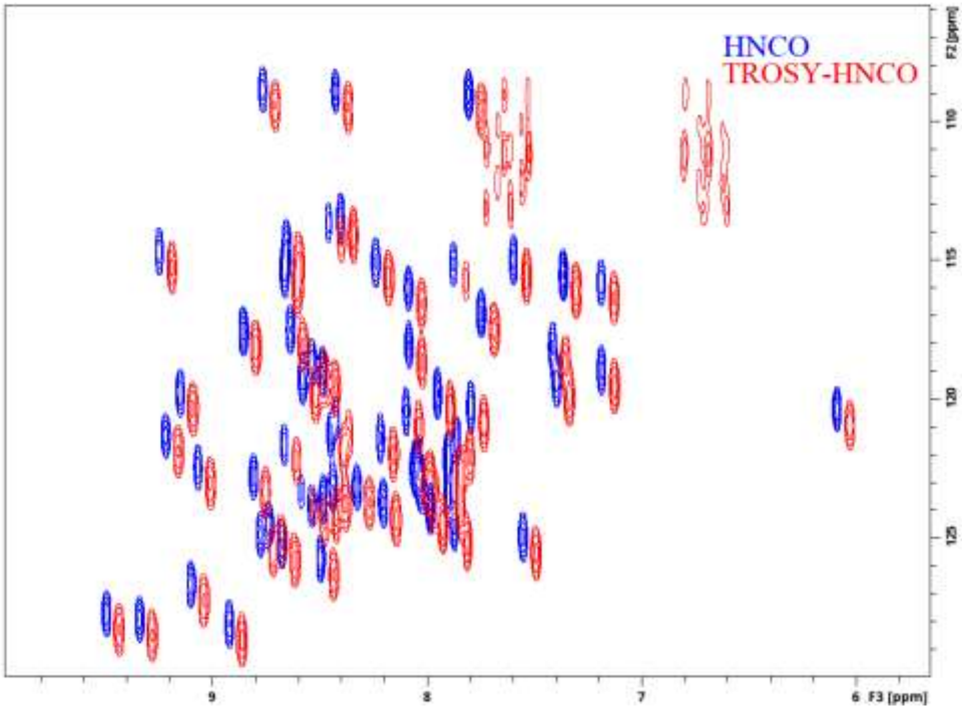


Fig. 2. Experimental demonstration of the implementations of the HNCO and TROSY-HNCO experiments for ubiquitin 8 (kDa) at 25 °C. Spectra were recorded as ^{1}H-^{15}N planes with maximum evolution times of 85.2 ms (^{1}H) and 9.87 ms (^{15}N) and processed using cosine squared window functions.

HNCO is one of the simplest triple-resonance experiments and thus a good candidate to demonstrate the strengths and limitations of the presented approach to library building. We present its ME NMR implementation in Fig. 1. We use a custom prosol file (used mostly for automatic precalculation of pulse parameters) to free up a number of variables. Evolution delays and increments are defined explicitly due to the proximal module's numbered variables (here *td2* and *in2*) being dimensionality-dependant. The block of defines specifies options for ME library - specifying the proximal (xh.incl) and distal (hx.incl) modules and the couplings to be evolved (Y is $^{2}J_{NCO}$) and decoupled (A is $^{2}J_{NCA}$). After importing the first two general modules, which include the distal modules, two evolution periods, the carbonyl echo is implemented using the library's low-level functionality. Since channels and pulses aren't selected explicitly, this block will function with split CA and CO channels (with the right spectrometer configurations and "CACO_SPLIT" defined in ZGOPTNS) or using a single carbon channel and frequency-offset pulses. The rest of the pulse program includes the end.incl module (with the two proximal echoes and acquisition) and standard configuration of gradients and phasecycles. To demonstrate the libraries functionality in Fig 2. we present 2D spectra (recorded as HN(CO) experiments) of a standard variant of the experiment (no ZGOPTNs) and a TROSY-HNCO (adding the TROSY define to ZGOPTNS) selecting only the H_{β} and N_{β} component (the lower right component using standard display convention). It's possible to choose a ^{13}C-GBIRD^{r,X} appending the "ACQ_BIRD_C" option to ZGOPTNS, with an example of line narrowing demonstrated in Fig. 3.

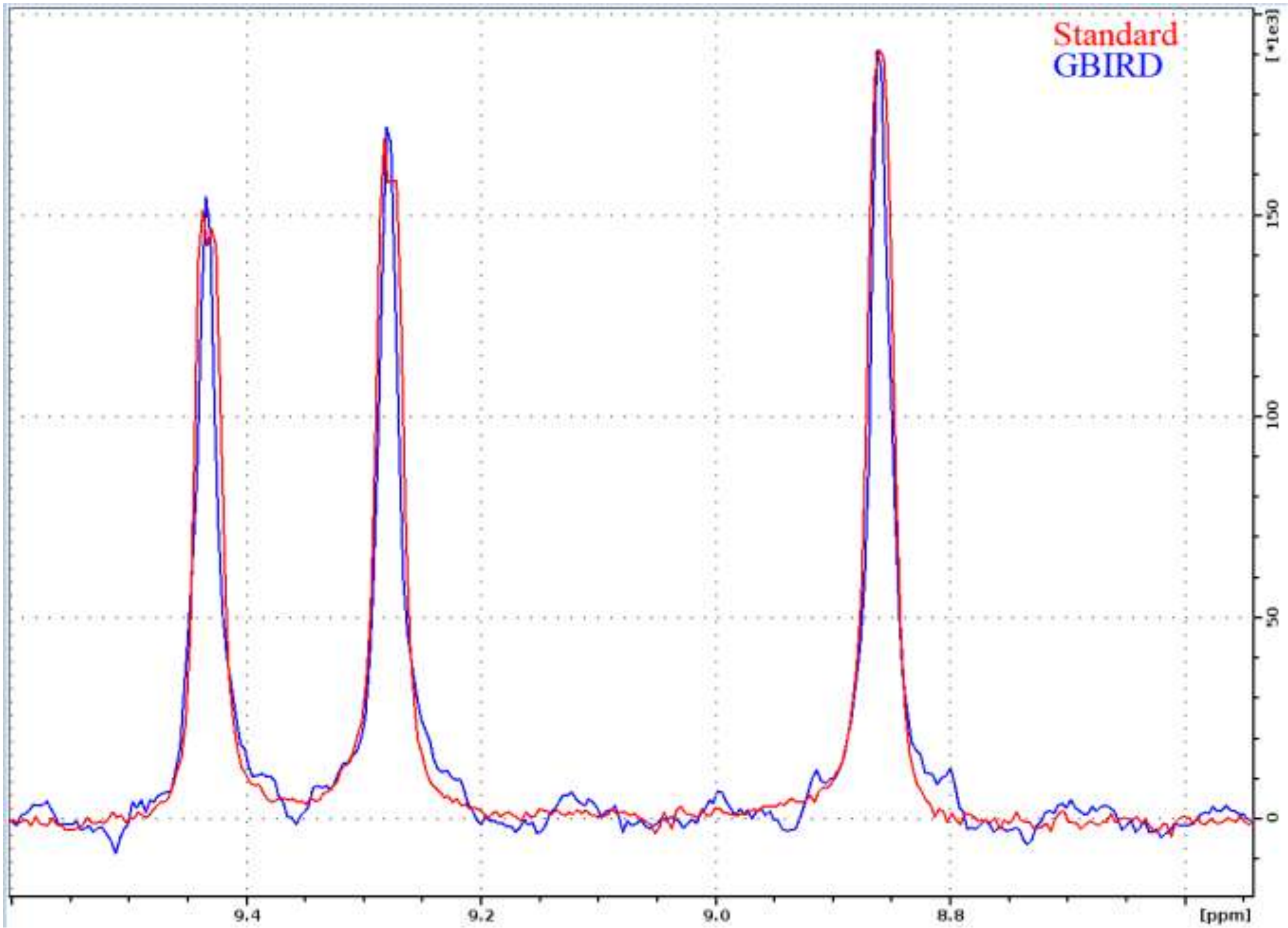
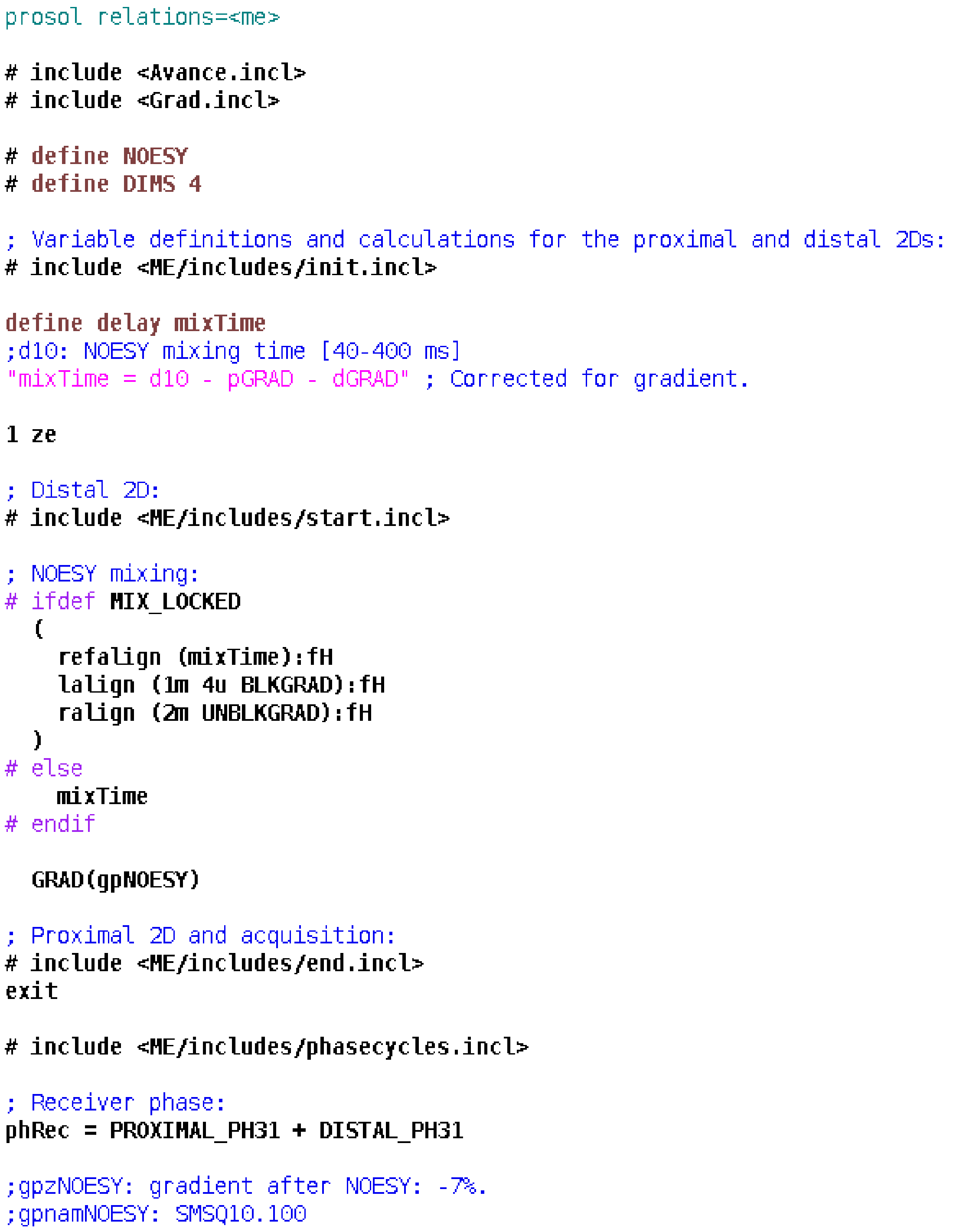


Fig. 3. 1D slices (for $N = 128.5$ ppm) through ^{1}H-^{15}N planes recorded for a TROSY-HNCO with standard acquisition and TROSY-HNCO with ^{13}C-GBIRD^{r,X} demonstrating the effectiveness of the homodecoupling and the resultant line narrowing. Both spectra were acquired for ubiquitin 8 (kDa) at 25 °C with maximum evolution times of 340.7 ms (^{1}H) and 9.87 ms (^{15}N) and processed using a cosine squared window function in the N dimension and sine, squared shifted by $\frac{\pi}{2}$ in the H dimension. The GBIRD spectrum was shifted right by 4 Hz (shift was possibly induced by sample heating) and scaled up to match the amplitude of the standard TROSY-HNCO. For the GBIRD spectrum, 18 chunks were acquired with a 11.96 ms inter-chunk delay, 3.5 ms $^{2}J_{HC}$ evolution time and using a 120 µs BIP-720-100-10 (Smith et al., 2001) pulse for ^{13}C inversion. Linewidths at half height are (from left to right) 19.6 Hz, 19.5 Hz and19.1 Hz for the standard spectrum and 13.2 Hz, 13.2 Hz and 13.7 Hz for the homodecouple spectrum (TopSpin peakw function).

4.2 4D NOESY



```
prosol relations=<me>

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

# define NOESY
# define DIMS 4

; Variable definitions and calculations for the proximal and distal 2Ds:
# include <ME/includes/init.incl>

define delay mixTime
;d10: NOESY mixing time [40-400 ms]
"mixTime = d10 - pGRAD - dGRAD" ; Corrected for gradient.

1 ze

; Distal 2D:
# include <ME/includes/start.incl>

; NOESY mixing:
# ifdef MIX_LOCKED
  (
    refalign (mixTime):fH
    lalign (1m 4u BLKGRAD):fH
    ralign (2m UNBLKGRAD):fH
  )
# else
    mixTime
# endif

  GRAD(gpNOESY)

; Proximal 2D and acquisition:
# include <ME/includes/end.incl>
exit

# include <ME/includes/phasecycles.incl>

; Receiver phase:
phRec = PROXIMAL_PH31 + DISTAL_PH31

;gpzNOESY: gradient after NOESY: -7%.
;gpnamNOESY: SMSQ10.100
```

Fig. 4. Pulse program code for the implementation of a 4D NOESY experiment.

The modular nature of the library is exemplified by the 4D NOESY pulse program in Fig. 4. Apart from the basic structure described above in the case of HNCO it only contains a mixing period joining the proximal and distal module, with the evolved heteronuclei and experiment types selected by the user using ZGOPTNS. A HC,NH-HMQC-NOESY-HSQC with sensitivity enhancement in the last dimension (Fig. 5.) can be changed to a HC,CH-HMQC-NOESY-HSQC (Fig. 6.) pulse program by changing the “PROXIMAL_N” option to “PROXIMAL_C” and adding the gradient selection option (“GS”, which isn’t a default for non-sensitivity-enhanced HSQC.

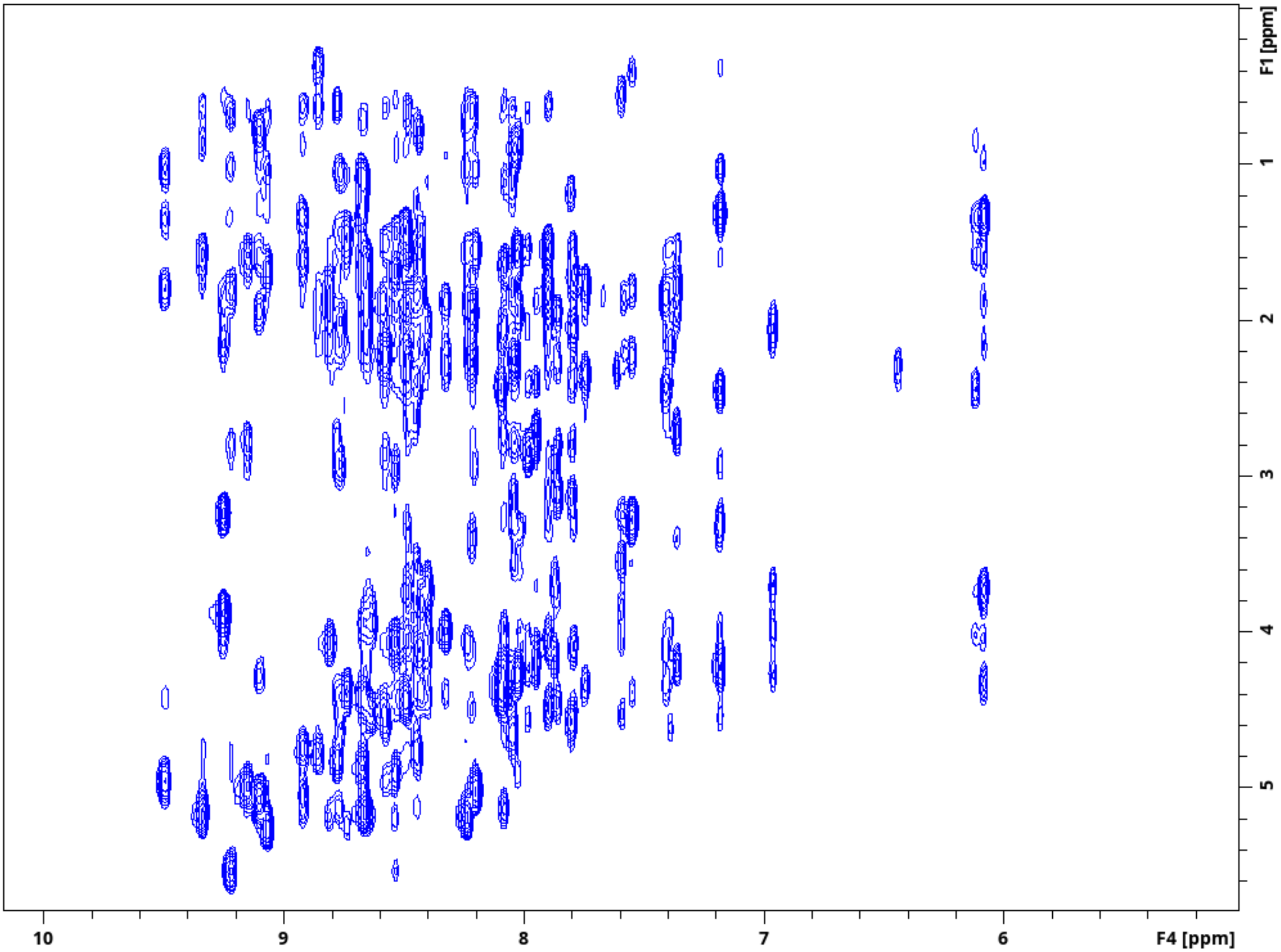


Fig. 5. ^{1}**H-**^{1}**H planes recorded using a 4D HC,NH-HMQC-NOESY-HSQC experiment for ubiquitin 8 (kDa) at 25 °C. Spectra were recorded with maximum evolution times of 85,2 ms (**^{1}**H direct dimension) and 6.99 ms (**^{1}**H indirect dimension) and processed using cosine squared window functions.**

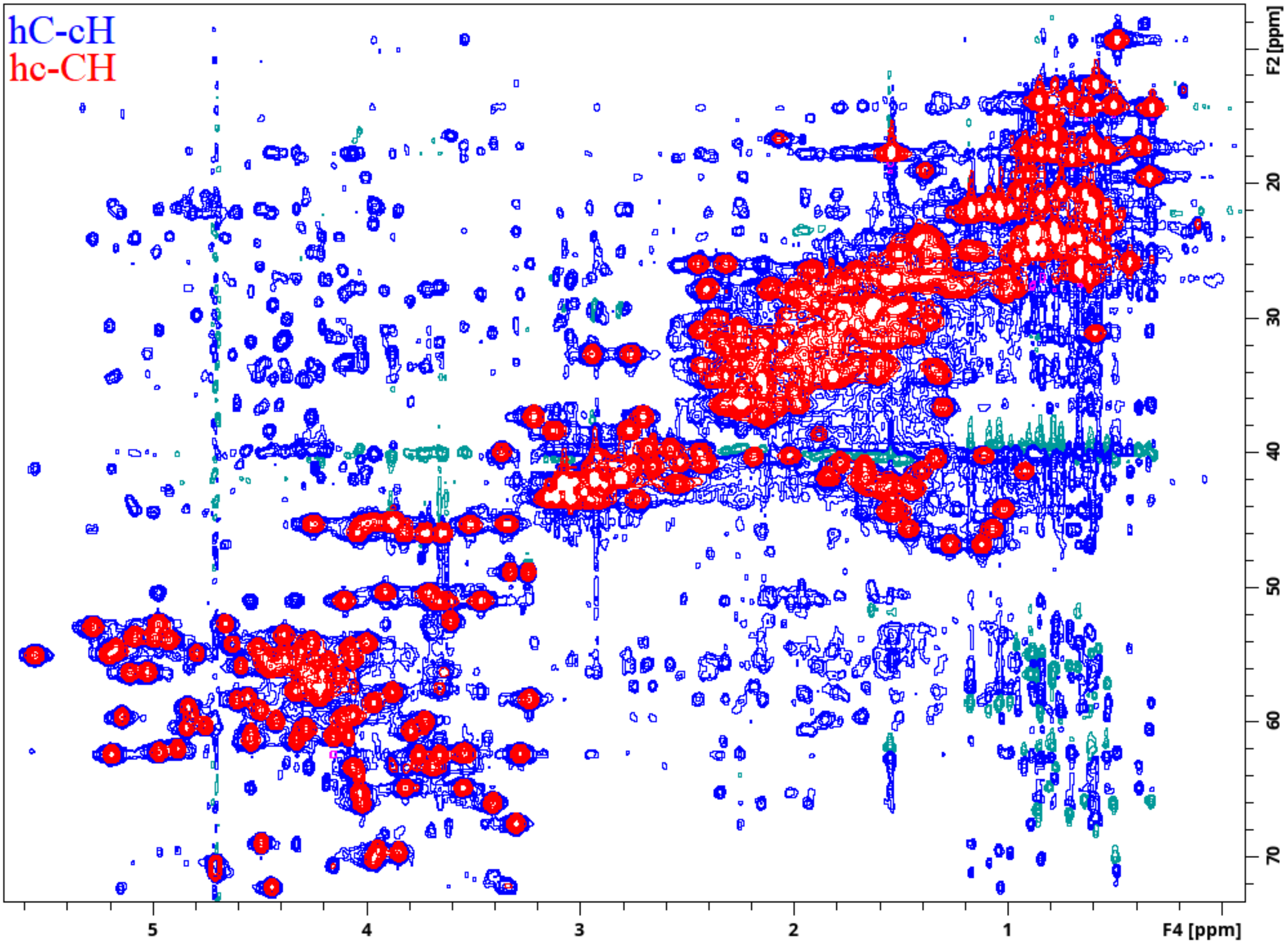


Fig. 6. Two different ^{1}H-^{13}C 2D planes recorded using a ME implementation of a 4D HC,CH-HMQC-NOESY-HSQC experiment for ubiquitin 8 (kDa) at 25 °C. Spectra were recorded with maximum evolution times of 85.2 ms (^{1}H direct dimension) and 7.96 ms (both ^{13}C dimensions) and processed using cosine squared window functions.

4.3 ^{1}H-^{15}N correlation – shaped pulses

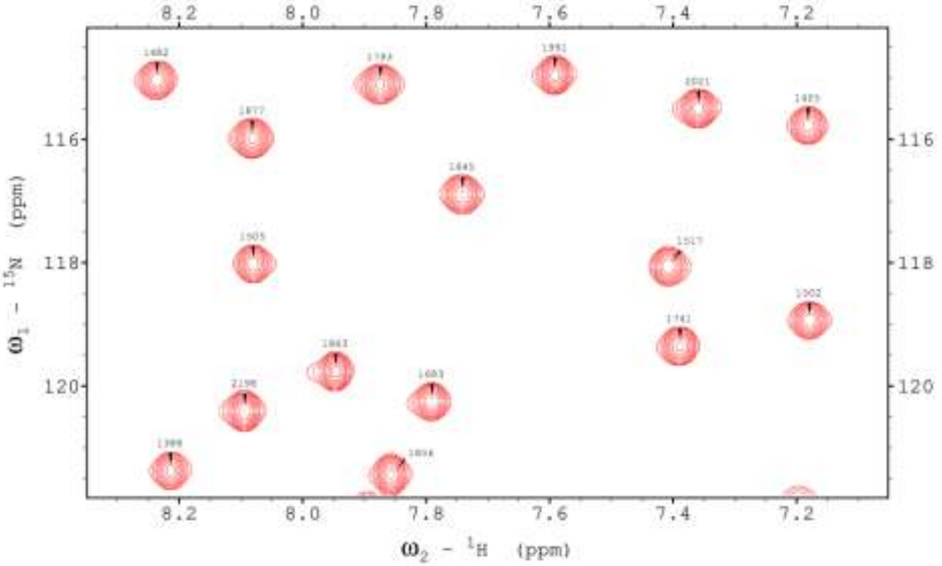


Fig. 7. ^{1}H,^{15}N TROSY spectrum recorded using a ME implementation with hard pulses and water flipback for ubiquitin 8 (kDa) at 25 °C. The spectrum was recorded with maximum evolution times of 85,2 ms (^{1}H) and 39.5 ms (^{15}N) and processed using cosine squared window functions.

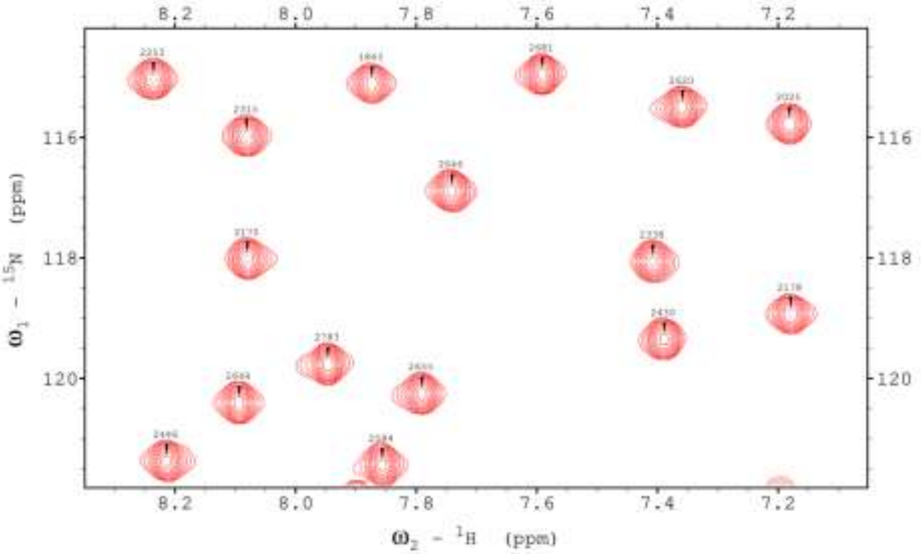


Fig. 8. ^{1}H,^{15}N TROSY spectrum recorded using a ME implementation with shaped pulses (E400B and RE-BURP) for ubiquitin 8 (kDa) at 25 °C. The spectrum was recorded with maximum evolution times of 85,2 ms (^{1}H) and 39.5 ms (^{15}N) and processed using cosine squared window functions.

Since BEST-type experiments utilizing shaped pulses can bring improved sensitivity especially at higher scan repetition rates (Schanda et al., 2006) we demonstrate the library's inherent ability to automatically adapt to the substantial chemical shift and coupling evolution during the 90-degree universal rotation E400B (Veshtort and Griffin, 2004) (using a time-reversed version of the original pulse for excitation) pulses with the length of 1073.1 us (equivalent to an ideal pulse followed by a 611.7 us delay) and refocussing pulse RE-BURP (Geen and Freeman, 1991) with length of 1108.8 us (modelled as an ideal refocussing pulse flanked by 554 us delays) in Fig. 7. and 8. With a relaxation delay of 0.65 s all peaks in the selected region are over 20% stronger in the shaped pulse version. Full datasets for a number of different relaxation delays are provided, as in the data availability section.

5 Materials & methods

For all experiments we used a 2 mM ^{13}C, ^{15}N-double labelled human ubiquitin (ASLA Biotech) in a 5 mm Shigemi NMR microtube. All spectra were acquired using a Bruker Avance IIIHD 800 MHz spectrometer with a 5 mm TCI z-gradient cryo-probe. Pulse lengths for 90 degree hard pulses were 10.47 μs for ^{1}H, 12.3 μs for ^{13}C and 33.22 μs for ^{15}N. Full acquisition and processing parameters are provided in the dataset linked below in the Data availability section. Acquisition and library testing was, performed using the TopSpin 3.6.5 Service Pack 2 software (Bruker). Data processing and plotting (aside from Fig. 7. and 8.) was carried out in TopSpin. Figures 7 and 8 were prepared using the NMRFAM-SPARKY software (Goddard and Kneller, 2004; Lee et al., 2015).

6 Conclusions

We have described a framework library implementing a two-level approach to pulse program modularization and demonstrated its utility. We hope it can be used by others either directly for the streamlining of pulse program code or as an inspiration for similar frameworks. Although the usefulness of the modularization approach is most obvious for the case of protein experiments presented here, it should extend to nucleic acids and, to a more limited extent, small molecules. In the latter case, the ability to modularize preparation period operations (presaturation, ASAP), WATERGATE (Piotto et al., 1992; Sklenar et al., 1993) type solvent suppression and real-time acquisition should be particularly useful.

Code availability

The initial version of the ME library code and documentation is available online at: https://doi.org/10.5281/zenodo.10841749. Current library version is available online at https://github.com/nmr-cnbch/MEnmr_pubcode or from the authors upon request.

Data availability

All data used in the preparation of this article is available online at: https://doi.org/10.5281/zenodo.10578330.

Author contributions

MG and WK designed the general workflow of the ME library. MG wrote the library code and performed the experiments. MG wrote the manuscript with input from WK.

Financial support

This research was supported by the Polish National Science Centre grant PRELUDIUM 2015/19/N/ST4/00863 to MG.

Competing interests

The authors declare that they have no conflict of interest.

References

Asami, S., Kallies, W., Günther, J. C., Stavropoulou, M., Glaser, S. J., and Sattler, M.: Ultrashort Broadband Cooperative Pulses for Multidimensional Biomolecular NMR Experiments, Angewandte Chemie, 130, 14706–14710, https://doi.org/10.1002/ange.201800220, 2018.

Bodenhausen, G. and Ruben, D. J.: Natural abundance nitrogen-15 NMR by enhanced heteronuclear spectroscopy, Chemical Physics Letters, 69, 185–189, https://doi.org/10.1016/0009-2614(80)80041-8, 1980.

Brüschweiler, R., Griesinger, C., Sørensen, O. W., and Ernst, R. R.: Combined use of hard and soft pulses for ω1 decoupling in two-dimensional NMR spectroscopy, Journal of Magnetic Resonance (1969), 78, 178–185, https://doi.org/10.1016/0022-2364(88)90171-0, 1988.

Favier, A. and Brutscher, B.: NMRlib: user-friendly pulse sequence tools for Bruker NMR spectrometers, J Biomol NMR, 73, 199–211, https://doi.org/10.1007/s10858-019-00249-1, 2019.

Garbow, J. R., Weitekamp, D. P., and Pines, A.: Bilinear rotation decoupling of homonuclear scalar interactions, Chemical Physics Letters, 93, 504–509, https://doi.org/10.1016/0009-2614(82)83229-6, 1982.

Geen, H. and Freeman, R.: Band-selective radiofrequency pulses, Journal of Magnetic Resonance (1969), 93, 93–141, https://doi.org/10.1016/0022-2364(91)90034-Q, 1991.

Gershenzon, N. I., Skinner, T. E., Brutscher, B., Khaneja, N., Nimbalkar, M., Luy, B., and Glaser, S. J.: Linear phase slope in pulse design: Application to coherence transfer, Journal of Magnetic Resonance, 192, 235–243, https://doi.org/10.1016/j.jmr.2008.02.021, 2008.

Goddard, T. D. and Kneller, D. G.: SPARKY 3, 2004.

Haller, J. D., Bodor, A., and Luy, B.: Pure shift amide detection in conventional and TROSY-type experiments of 13C,15N-labeled proteins, J Biomol NMR, 76, 213–221, https://doi.org/10.1007/s10858-022-00406-z, 2022.

Ikura, M., Kay, L. E., and Bax, A.: A novel approach for sequential assignment of 1H, 13C, and 15N spectra of proteins: heteronuclear triple-resonance three-dimensional NMR spectroscopy. Application to calmodulin, Biochemistry, 29, 4659–4667, 1990.

Kay, L. E., Clore, G. M., Bax, A., and Gronenborn, A. M.: Four-dimensional heteronuclear triple-resonance NMR spectroscopy of interleukin-1 beta in solution, Science, 249, 411–414, https://doi.org/10.1126/science.2377896, 1990a.

Kay, L. E., Ikura, M., Tschudin, R., and Bax, A.: Three-dimensional triple-resonance NMR spectroscopy of isotopically enriched proteins, Journal of Magnetic Resonance (1969), 89, 496–514, https://doi.org/10.1016/0022-2364(90)90333-5, 1990b.

Kazimierczuk, K., Zawadzka-Kazimierczuk, A., and Koźmiński, W.: Non-uniform frequency domain for optimal exploitation of non-uniform sampling, Journal of Magnetic Resonance, 205, 286–292, https://doi.org/10.1016/j.jmr.2010.05.012, 2010.

Krishnamurthy, V. V.: Application of Semi-Selective Excitation Sculpting for Homonuclear Decoupling During Evolution in Multi-Dimensional NMR, Magnetic Resonance in Chemistry, 35, 9–12, https://doi.org/10.1002/(SICI)1097-458X(199701)35:1<9::AID-OMR930>3.0.CO;2-R, 1997.

Kupče, E. and Freeman, R.: Fast multidimensional NMR by polarization sharing, Magnetic Resonance in Chemistry, 45, 2–4, https://doi.org/10.1002/mrc.1931, 2007.

Lee, W., Tonelli, M., and Markley, J. L.: NMRFAM-SPARKY: enhanced software for biomolecular NMR spectroscopy, Bioinformatics, 31, 1325–1327, https://doi.org/10.1093/bioinformatics/btu830, 2015.

Lescop, E., Kern, T., and Brutscher, B.: Guidelines for the use of band-selective radiofrequency pulses in hetero-nuclear NMR: Example of longitudinal-relaxation-enhanced BEST-type 1H–15N correlation experiments, Journal of Magnetic Resonance, 203, 190–198, https://doi.org/10.1016/j.jmr.2009.12.001, 2010.

Lukavsky, P. J. and Puglisi, J. D.: RNAPack: An Integrated NMR Approach to RNA Structure Determination, Methods, 25, 316–332, https://doi.org/10.1006/meth.2001.1244, 2001.

Nietlispach, D.: Suppression of anti-TROSY lines in a sensitivity enhanced gradient selection TROSY scheme, J Biomol NMR, 31, 161–166, https://doi.org/10.1007/s10858-004-8195-7, 2005.

Palmer, A. G., Cavanagh, J., Wright, P. E., and Rance, M.: Sensitivity improvement in proton-detected two-dimensional heteronuclear correlation NMR spectroscopy, Journal of Magnetic Resonance (1969), 93, 151–170, https://doi.org/10.1016/0022-2364(91)90036-S, 1991.

Piotto, M., Saudek, V., and Sklenář, V.: Gradient-tailored excitation for single-quantum NMR spectroscopy of aqueous solutions, J Biomol NMR, 2, 661–665, https://doi.org/10.1007/BF02192855, 1992.

Salzmann, M., Wider, G., Pervushin, K., and Wüthrich, K.: Improved sensitivity and coherence selection for [15N,1H]-TROSY elements in triple resonance experiments, J Biomol NMR, 15, 181–184, https://doi.org/10.1023/A:1008358030477, 1999a.

Salzmann, M., Wider, G., Pervushin, K., Senn, H., and Wüthrich, K.: TROSY-type Triple-Resonance Experiments for Sequential NMR Assignments of Large Proteins, J. Am. Chem. Soc., 121, 844–848, https://doi.org/10.1021/ja9834226, 1999b.

Schanda, P., Van Melckebeke, H., and Brutscher, B.: Speeding Up Three-Dimensional Protein NMR Experiments to a Few Minutes, J. Am. Chem. Soc., 128, 9042–9043, https://doi.org/10.1021/ja062025p, 2006.

Schulte-Herbrüggen, T. and Sørensen, O. W.: Clean TROSY: Compensation for Relaxation-Induced Artifacts, Journal of Magnetic Resonance, 144, 123–128, https://doi.org/10.1006/jmre.2000.2020, 2000.

Sklenar, V.: Suppression of Radiation Damping in Multidimensional NMR Experiments Using Magnetic Field Gradients, Journal of Magnetic Resonance, Series A, 114, 132–135, https://doi.org/10.1006/jmra.1995.1119, 1995.

Sklenar, V., Piotto, M., Leppik, R., and Saudek, V.: Gradient-Tailored Water Suppression for 1H-15N HSQC Experiments Optimized to Retain Full Sensitivity, Journal of Magnetic Resonance, Series A, 102, 241–245, https://doi.org/10.1006/jmra.1993.1098, 1993.

Smith, M. A., Hu, H., and Shaka, A. J.: Improved Broadband Inversion Performance for NMR in Liquids, Journal of Magnetic Resonance, 151, 269–283, https://doi.org/10.1006/jmre.2001.2364, 2001.

Solyom, Z., Schwarten, M., Geist, L., Konrat, R., Willbold, D., and Brutscher, B.: BEST-TROSY experiments for time-efficient sequential resonance assignment of large disordered proteins, J Biomol NMR, 55, 311–321, https://doi.org/10.1007/s10858-013-9715-0, 2013.

Sprangers, R. and Kay, L. E.: Quantitative dynamics and binding studies of the 20S proteasome by NMR, Nature, 445, 618–622, https://doi.org/10.1038/nature05512, 2007.

Stallman, R., M. and GCC Developer Community: Using the GNU Compiler Collection, Free Software Foundation, Boston, 2012.

Vallet, A., Favier, A., Brutscher, B., and Schanda, P.: ssNMRlib: a comprehensive library and tool box for acquisition of solid-state nuclear magnetic resonance experiments on Bruker spectrometers, Magnetic Resonance, 1, 331–345, https://doi.org/10.5194/mr-1-331-2020, 2020.

Veshtort, M. and Griffin, R. G.: High-Performance Selective Excitation Pulses for Solid- and Liquid-State NMR Spectroscopy, ChemPhysChem, 5, 834–850, https://doi.org/10.1002/cphc.200400018, 2004.

Yang, D. and Kay, L. E.: TROSY Triple-Resonance Four-Dimensional NMR Spectroscopy of a 46 ns Tumbling Protein, J. Am. Chem. Soc., 121, 2571–2575, https://doi.org/10.1021/ja984056t, 1999.

Yong, J. R. J., Kupče, E., and Claridge, T. D. W.: Modular Pulse Program Generation for NMR Supersequences, Anal. Chem., 94, 2271–2278, https://doi.org/10.1021/acs.analchem.1c04964, 2022.

Zawadzka-Kazimierczuk, A.: New methods of protein NMR spectra analysis using the techniques of high dimensionality, Doctoral dissertation, University of Warsaw, 2012.