MddNMR

Version 2.6, January 2018

Reconstruction of NMR spectra from non-uniformly sampled signal using multi-dimensional decomposition (MDD) and Compressed Sensing (CS)

The User manual

Developed by
Orekhov, Vladislav
Jaravine, Victor
Mayzel, Maxim
Kazimierczuk, Krzysztof

University of Gothenburg
Gothenburg, Sweden

Overview

MddNMR is a program for processing of non-uniformly sampled (NUS) multidimensional NMR spectra. The package contains also a routine to produce NUS schedule that can be used to setup N-dimensional NUS NMR experiments. Potentially any pulse sequence can be run in the NUS mode. In the NUS acquisition, only a fraction of full (conventional) data set is recorded. MddNMR uses multi-dimensional decomposition (MDD) and compressed sensing (CS) to replenishing missing data points in the full matrix followed by regular FT processing of the complete data.

Copyright

Citing the software

When presenting results obtained using the software, please cite at least one of the following papers:

1. Orekhov, V.Y. and V.A. Jaravine, Analysis of non-uniformly sampled spectra with Multi-Dimensional Decomposition. Prog. Nucl. Magn. Reson. Spectrosc., 2011, 59, p 271-292.
2. Kazimierczuk, K. and V.Y. Orekhov, Accelerated NMR Spectroscopy by Using Compressed Sensing, Angew. Chem.-Int. Edit., 50, p. 5556-5559.
3. Qu, X.; Mayzel, M.; Cai, J.-F.; Chen, Z.; Orekhov, V. Accelerated NMR Spectroscopy with Low-Rank Reconstruction. Angew. Chemie - Int. Ed. 2015, 54 (3).

Downloads and updates

The software is available for download at http://mddnmr.spektrino.com/download
or upon request from:

Vladislav Y. Orekhov
Professor
Swedish NMR Center at Gothenburg University
Box 465, Gothenburg, SE 40530, Sweden
E-mail: orov@nmr.gu.se

All users of the program are encouraged to join news-group "mddnmr" at http://groups.google.com/group/mddnmr. The group is a forum for the MDD, CS, and software related discussions, as well as a billboard to inform the users about updates and bug fixes.

Installation

Note that to run the software you must have functioning nmrPipe package (Delaglio, F., et al., 1995, J. Biomol. NMR, 6, 277-293). In order to run GUI *qMDD*, you also need python with additional libraries. For this, we suggest to install, for example, latest EPD python package from http://www.enthought.com/ , which is free for academic use.

Mddnmr software is distributed as a compressed Unix tar archives, e.g. mddnmr2.0_29Jun2011.tgz . Current version supports Linux (32 and 64 Bit) and Mac (Intel) OS X 10.4 and later. The corresponding binaries are automatically selected during installation. The step-by-step Installation procedure is the following

1. Uncompress and unfold the archive.
2. Read content of Copyright file.
3. Change directory to mddnmr2.xx and run command
   ./Install

4. Add several lines into to your .cshrc file, as suggested by the terminal output produced by the script.

5. [Optional] Download and install examples by unfolding corresponding tar archives in your preferable data location directory.

General concept

Traditionally, multi-dimensional NMR experiments are collected on regular grid of equally spaced points in the time domain. The signal is processed by Discrete Fourier transform (DFT). NUS or sparse data are generally processed by other methods. Sparse recording of spectra can save a lot of time, especially for high-resolution nD datasets with extensive phase cycling.

Processing of a regular NMR spectrum includes several steps: (i) conversion of the FID and parameters into *nmrPipe* format; (ii) Fourier transform in the directly detected dimension; (iii) Fourier transform in all indirect dimensions; viewing of the result and, if needed, fine-tuning of the processing parameters. If spectrum is recorded in the NUS mode, the indirect dimensions cannot be Fourier transformed right away and *mddNMR* software intervenes between steps (ii) and (iii). Steps i-iii are performed using *nmrPipe*. The role of the *mddNMR* is to replenish complete data matrix with reconstructed points (Fig. 1). The software offers four general possibilities:

(a) *Direct Fourier transform.* The complete data matrix is obtained by setting all missing points to zero. This spectrum has the lowest power among all possible spectra compatible with measured data.

(b) *Multi-dimensional decomposition (MDD)* Spectrum constructed using MDD model.

(c) *Compressed sensing (CS).* The sparsest spectrum, which is compatible with measured data.

(d) *Low-rank method (LR)*. The FID containing lowest number of components, which is compatible with measured data.

In this manual, usage of the software is described by several commented examples. In addition, complete set of parameters and formats of essential files are given in Appendixes. Description of underlying mathematical algorithms and processing protocols can be found in our papers listed above, and references cited therein.

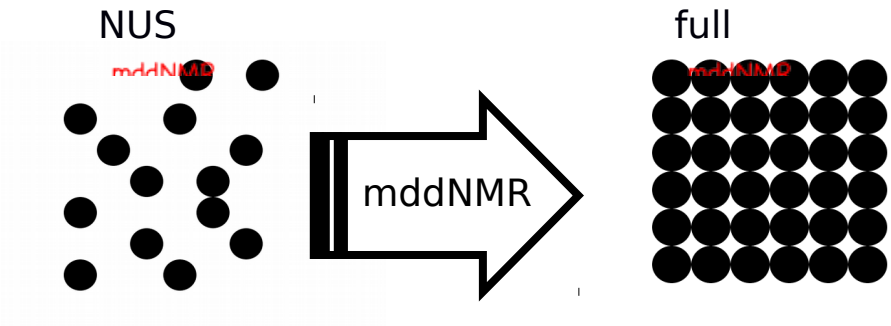


Figure 1. The software replenishes time domain data points in the indirect dimensions that are missing in the NUS set and produces the full data set amenable for regular Fourier transform

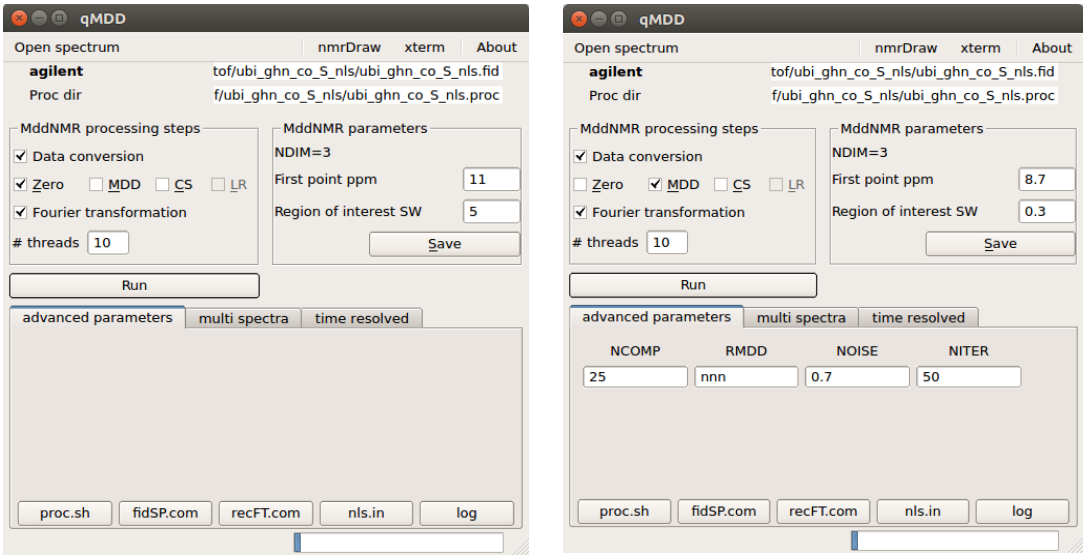
qMDD graphical user interface

The primary mode of mddnmr usage, which gives access to the full set of the software functionalities, is by C-shell scripts. There is, however, a graphical user interface (GUI) that simplifies work with the program, and is especially recommended for beginners. It is started with command

qMDD

A window is opened, which invites to select a spectrum for processing. For example, you may select ubi_ghn_co_S_nls.fid, which is one of the example experiments in http://mddnmr.spektrino.com/download (data type is recognized by directory name, e.g. .fid for Variant/Agilent). Working directory named *ubi_ghn_co_S_nls.proc* is created, which is the place where you find all files discussed below. Answer "yes"

for the question (if any) about overwriting existing processing files. You are ready to process the spectrum by pressing button "RUN" (figure below, left).



When button "RUN" is pressed, *qMDD* runs script *proc.sh*, which is found in the working directory (*ubi_ghn_co_S_nls.proc*). Active "Stop" button and sliding bar in the low-right corner of the window indicate calculations in progress. Details of the progress can be seen also in *log* window, which is opened with the corresponding button in the bottom right part of the program panel. When the script is successfully finished, look at the resulting spectrum in *nmrDraw* by pressing button *"nmrDraw"* or starting *nmrDraw* in a terminal window. In *nmrDraw* file look at three projections of the 3D spectrum stored in *H1.C13.dat, 1H.N15.dat* and *N15.C13.dat* or the 3D spectrum located in directory *ft*. In the figure above (left), you may notice that the calculations were performed with "Zero*"* mode, which is essentially normal Fourier transform with all missing data points set to zero., This is the fastest and most robust method, albeit it provides the poorest results due to massive aliasing artefacts. Nevertheless, "Zero*"* mode (minimal power reconstruction) is useful since it allows fast look at the spectrum and adjustment of *nmrPipe* processing parameters, e.g. phases.
mddNMR software uses *nmrPipe* for spectral data conversion and traditional processing. There are two *nmrPipe* scripts that deal with these: *fidSP.com* and *recFT.com*. The former is responsible for conversion of the spectrum to the *nmrPipe* format and processing of the directly detected dimension. The latter script processes all indirect spectral dimensions after the missing data in the time domain interferogram is replenished by *mddNMR*. The scripts can be viewed and edited from "Advanced parameters" by pressing the corresponding buttons. For example, in the projection 1H.N15.dat you see that the phase is off in the ^{1}H direction. Phases for all dimensions can be set in *fidSP.com* by modifying parameters for var2pipe (bru2pipe) -xP0, -xP1, -yP0, -yP1, etc. Set value for -xP0 to 48 (deg). Do not forget saving the script before closing the editor window. Rerun the calculation to see effect of the phase change.
Check the MDD box in order to try MDD algorithm for the same data. The GUI allows modification of several most important parameters. This is done in the "Advanced" display (fig. right). For example, you may set parameter and CEXP to "yyn" in order to activate R-MDD mode for the indirect dimensions (13C, 15N) of the experiment. Prior to pressing *"RUN"*, you may define a small region of interest in the directly detected (amide proton) dimension by setting *"First point ppm"* and *"Region of Interest SW ppm"*, as shown in the figure (right). This reduces the calculation time proportionally to the region size. Check also parameter *"# threads"*, which specify how many computational tasks can run simultaneously on your computer. You need to press *"Save"* to activate the changes. Meanings and recommended values for the parameters are indicated in contextual help, which appears when the mouse cursor is placed on the parameter field.

Adjusting conventional processing parameters

mddNMR software uses nmrPipe for spectral data conversion and traditional processing. There are two scripts that deal with these: *fidSP.com* and *recFT.com*. The former is responsible for conversion of the spectrum to the nmrPipe format and processing of the directly detected dimension. The latter processes all indirect spectral dimensions after the missing data in the time domain interferogram is replenished by *mddNMR*. The scripts can be viewed and edited from "Advanced" display by pressing the corresponding buttons. The procedure can be illustrated on *gNhsqc_S.fid* spectrum example. Load the spectrum using "Browse" button and answer "Yes" to the question (if any) to discard the existing processing scripts. Process the spectrum with "FT". Inspection of the spectrum in nmrDraw shows that phase in the directly detected dimension requires adjustment by ca. 70 degrees. Press "*fidSP.com*" button in the "Advanced" display and set the phase correction as shown below (in the highlighted line). Save the script and rerun the calculations.

```
var2pipe -in fid -noaswap -aqORD 0 \
 -xN           2048           -yN            192             \
 -xT           1024           -yT            96              \
 -xMODE        Complex        -yMODE         Rance-Kay       \
 -xSW          13008.100      -ySW           2600.300        \
 -xOBS         800.128        -yOBS          81.085          \
 -xCAR         4.755          -yCAR          116.641         \
 -xP0          156.1          -yP0             0.0           \
 -xP1            0.0          -yP1             0.0           \
 -xLAB         H1             -yLAB          N15             \
 -ndim 2       -aq2D States \
 | nmrPipe -fn SOL                                          \
 | nmrPipe -fn SP -off 0.450 -end 0.970 -pow 2 -c 0.500 \
 | nmrPipe -fn ZF -auto                                     \
 | nmrPipe -fn FT -auto                                     \
 | nmrPipe -fn PS -hdr                                      \
 | nmrPipe -fn PS -p0 70 -p1 0 -di                          \
 | nmrPipe -fn EXT -x1 11ppm -xn 5ppm -sw -round 16      \
 | pipe2xyz -z -out ft/data%03d.DAT -ov -nofs -verb
```

Compressed Sensing

CS processing is activated by selecting "CS" checkbox. You may try CS on *gNhsqc_S.fid* spectrum example. **Note, that from version 2.5 *mddNMR* can be used to process 4D spectra with CS.** First, process the spectrum with "FT". Adjust the phase for the directly detect dimension as described in the previous section. Check the CS box and press "RUN" to calculate the spectrum. The result, which is stored in *test.dat* file, can be viewed in nmrDraw. "Advanced" view (see figure below) allows checking and editing the essential parameters for CS. For example, one can chose to use Iterative Reweighted Least Squares (IRLS) or Iterative Soft Thresholding (IST) algorithms (for 4D spectra only IST is allowed). Meanings and recommended values for the parameters are indicated in contextual help, which appears when the mouse cursor is placed on the parameter field.

Result of CS calculations is better, if Virtual Echo is used (see below).

CS with Virtual Echo

Compressed sensing is based on the assumption of sparsity (compressibility) of NMR spectra. Sparsity can be enhanced by a trick called Virtual Echo leading to better reconstruction quality (especially at low sampling levels). For details see: Mayzel, M.; Kazimierczuk, K.; Orekhov, V. Y. The Causality Principle in the Reconstruction of Sparse NMR Spectra. Chem. Commun. 2014, 50 (64), 8947–8950.
For 4D VE is turned on by default (hardcoded) to save calculation time.

Two practical aspects of VE should be mentioned:

1. To use VE set **CS_VE** to **y** and **phase** parameter to appropriate value corresponding phase in the indirect dimensions (for half-dwell time, set **phase** to **0** and **f180** to **y**). For instance, 2D spectrum requiring phase correction by 90 deg in F1 can be processed with VE if following lines are added into a script:

```
setenv CS_VE   y
setenv phase '90 0'
```

2. Calculation times and memory consumption are different for VE:

algorithm	Memory consumption (vs non-VE)	Calculation time (vs non-VE)
2D IRLS	ca. 2x higher	ca. 4x higher
3D IRLS	ca. 4x higher	ca. 4x higher
2D IST	ca. the same	ca. the same
3D IST	ca. the same	2x lower
4D IST (VE is obligatory!)	ca. the same	4x lower

Master script proc.sh

The role of GUI *qMDD* is to collect and check input from the user and to produce all necessary files for the calculations. The actual calculations are performed by C-shell script called *proc.sh* (or alike). The GUI supports basic and most frequently used features of the software, while more advanced processing may require editing of the master script. The script can be viewed and modified in “Advanced” view by pressing button *proc.sh* or in any text editor. As soon as the script is ready, it can run in a terminal window or be by pressing “Run” button. For example, *proc.sh* script for MDD/CS processing of *gNhsqc_S.fid* spectrum described in the previous section is

```
#!/bin/tcsh

setenv FST_PNT_PPM 11

setenv ROISW 5

setenv NUS_POINTS 96

setenv NDIM 2

setenv MDDTHREADS 4

setenv METHOD MDD



# CS algorithm related parameters

setenv CS_alg IRLS
```

```
setenv CS_norm 0

setenv CS_lambda 1.0

setenv CS_niter 10



# MDD algorithm related parameters

setenv SRSIZE 0.1

setenv NCOMP 25

setenv NITER 50

setenv MDD_NOISE 0.7

setenv CEXP nn

setenv lambda 0.001



mddnmr4pipeN.sh 1 2 3 4 5
```

The master script sets all parameters that have to be altered from defaults. This is done by setting C-shell environment variables using command *sentenv* . The actual calculations are started by command *mddnmr4pipeN.sh* at the end of the script. As arguments, the command takes list of tasks to be performed (1 – 5), which are defined by unique numbers (described in *mddnmr* manual).

Advanced processing

Several examples presented in this manual illustrate most of the software features. The commands are typically arranged into short C-shell (Unix) scripts. The master script, which is called *proc.sh* in this manual and in all examples, first sets several parameters (most of the parameters have good defaults values, and are not set explicitly). The parameters are set as Unix environment variables with command *setenv* (see examples). They can be changed by modifying the script. Finally the processing is performed by *mddnmr4pipeN.sh* command. As line parameters *mddnmr4pipeN.sh* takes a list of *steps*, which are integer numbers. Typically steps 1 - 5 are executed sequentially because output from a previous step serves as an input for the next one. The steps are:

0 – print full list of parameters recognized by the program.
1 – conversion of ser/fid to nmrPipe format; processing of the direct dimension and extraction of region of interest (ROI) (see nmrPipe script defined by parameter *fidSP*).
2 – preparing input for MDD/CS calculations.
3 – MDD/CS calculations over all sub-regions of the ROI.
33 - estimate memory required for the reconstruction
4 – full spectrum matrix is produced.
5 – the full time domain reconstruction obtained in step 4 is processed using an nmrPipe script (see parameter *REC2FT*)
42 – MDD mode only: MDD shapes obtained on step 3 are processed by nmrPipe and written into Unified Spectral Format (USF3) (see parameters Proc3D_* and Proc4D_*).

Several steps can be executed in one line, e.g.

mddnmr4pipeN.sh 1 2 3 4 5

or can be done by consecutive calls of *mddnmr4pipeN.sh,* for example:

mddnmr4pipeN.sh 1 2 3

mddnmr4pipeN.sh 4 5

In the first case, the program passes the data from one step to another in memory. In the latter example, intermediate results are stored in files in working directory MDD. This can be useful to skip lengthy calculations on initial steps when changes are required for the later downstream steps only. For example, lengthy MDD calculations on step 3 may be performed only once for all possible nmrPipe processing of the reconstructed spectrum obtained at steps 4 and 5.

Input files

There is at least one file, which needs to be prepared for the processing. Name of this file is conveyed to *mddNMR* by parameter *fidSP*. The file is an nmrPipe script that performs conversion from spectrometer to nmrPipe data formats using programs bruk2pipe or var2pipe, Fourier transform of the directly detected dimension and storing of region of interest (ROI) to disk. In this manual and in all examples, the script is called *fidSP.com*. The script is automatically generated by *qMDD* GUI or can be produced by command *nus2pipe* from *mddNMR* software or can be prepared by editing the data conversion script produced by programs *bruker/varian* from nmrPipe package. For example, for experiment 57 from the examples, correct *fidSP.com* file is produced by

nus2pipe -f 57 -t Bruker

Below is an example of the *fidSP* file for experiment 57.

```
bruk2pipe -in ./ser -bad 0.0 -aswap -DMX -decim 2000      \
                    -dspfvs 20 -grpdly 67.9862518310547   \
-xN      2048       -yN      1       -zN     3712         \
-xT      1024       -yT      0       -zT      856         \
-xMODE  DQD         -yMODE Complex   -zMODE Complex       \
-xSW    10000.000   -ySW    2500.000 -zSW    2500.000    \
-xOBS     600.130   -yOBS    150.903 -zOBS     60.811     \
-xCAR       4.702   -yCAR    175.327 -zCAR    115.840    \
-xP0      -46.9     -yP0       0.0   -zP0       0.0      \
-xP1       22.4     -yP1       0.0   -zP1       0.0      \
-xLAB    1H     -yLAB     13C      -zLAB     15N         \
-ndim 3 -aq2D States \
 | nmrPipe -fn POLY -time                                 \
 | nmrPipe -fn SP -off 0.450 -end 0.970 -pow 2 -c 0.500  \
 | nmrPipe -fn ZF -auto                                  \
 | nmrPipe -fn FT -auto                                  \
 | nmrPipe -fn PS -hdr                                   \
 | nmrPipe -fn PS -p0 0 -p1 0 -di                        \
 | nmrPipe -fn EXT -sw                                   \
 #| nmrPipe -fn POLY -auto                               \
 | pipe2xyz -z -out ft/data%03d.DAT -ov -nofs -verb
```

Script *fidSP.com* may be modified, for instance, for adjusting phase in the indirect dimension or chemical shift references. On step 1, script *fidSP.com* is used as a template for generating several scripts (FTx.sh*) , which are actually used in the processing.

Name of another important nmrPipe script is set by parameter *REC2FT*. This is an nmrPipe script that performs regular processing of the full data matrix in all indirect dimensions. User may need to change some parameters, e.g. phase corrections, weighting functions, linear prediction, etc. To do this, make and/or edit a local copy of the script *recFT.com*, which is located in the *${MDD_NMR}/com* directory.

Real dimensions

The processing of NMR data with real dimensions is possible in both CS and MDD modes. CS performs a separate processing on each of 2D or 3D spectra in a series (thus forming pseudo-3D or pseudo-4D input and output). MDD treats all data together, which provides superior reconstruction, if peaks do not change their positions. To process pseudo-3D or pseudo-4D data, the *dtype* parameter has to be set to define which

dimensions are ‘real’, i.e. enumerate spectra in a stack. For serial 2D data, the parameter should be set to rcr, for serial 3D data to rccr.

Please note the following :

1. Only 1st indirect dimension can be real
2. qMDD does yet not support script setup for real dimensions
3. Sampling schedule should include all indirect dimensions, also the real one.

Below are the example pre- (*fidSP*) and post-processing (*recFT*) scripts for pseudo-3D (series of 2D) data:

```
fidSP:

bruk2pipe -in ./ser -bad 0.0 -noswap -DMX -decim 1584 -dspfvs 21 -grpdly 76 \
-xN     1536       -yN       1           -zN          400           \
-xT     768        -yT       1           -zT          200           \
-xMODE  Complex    -yMODE    QF          -zMODE       Echo-AntiEcho \
-xSW    12626.263  -ySW      4409.222    -zSW         2553.626            \
-xOBS   899.890    -yOBS     899.890     -zOBS        91.185        \
-xCAR   4.698      -yCAR     4.698       -zCAR        117.500       \
-xP0    -122.1     -yP0      0.0         -zP0         90.0          \
-xP1      0.0      -yP1      0.0         -zP1         0.0           \
-xLAB   1H         -yLAB     T2          -zLAB        N15           \
-ndim 3 -aq2D States \
| nmrPipe -fn POLY -time                                \
| nmrPipe -fn SP -off 0.48 -end 0.95 -pow 2 -c 0.5     \
| nmrPipe -fn ZF -auto                                 \
| nmrPipe -fn FT -auto                                 \
| nmrPipe -fn PS -hdr                                  \
| nmrPipe -fn PS -p0 0 -p1 0 -di                       \
| nmrPipe -fn POLY -auto -xn 5.0ppm -ord 1              \
| nmrPipe -fn EXT -sw                                  \
| pipe2xyz -z -out ft/data%03d.DAT -ov -nofs -verb
```

recFT:

```
#!/bin/tcsh -f
cat $1                                          \
| nmrPipe  -fn TP -auto                         \
| nmrPipe  -fn ZTP -auto                         \
| nmrPipe  -fn SP -off 0.48 -end 0.95 -pow 2 -c 0.5 \
| nmrPipe  -fn ZF -auto                             \
| nmrPipe  -fn FT                                 \
| nmrPipe  -fn PS -hdr                            \
| nmrPipe  -fn PS -p0 0.0 -p1 -0.0 -di            \
| nmrPipe  -fn TP  -auto                          \
| pipe2xyz -out $proc_out -x -verb -ov
echo $proc_out ready
exit
```

Processing parameters

The parameters are typically set in the master C-shell script using command *setenv*. Full list of parameters with their current values can be viewed by command

mddnmr4pipeN.sh 0

Since most of the parameters have good default values, typically only few parameters need to set explicitly. For an illustration, let us look at the commented master script for processing scripts for experiment 57, which is one of the examples provided with the software. The experiment is a 3D HNCO spectrum recorded on Bruker spectrometer using NUS acquisition mode in TopSpin 3.0.

```
# input/output files
setenv FID ../57          # location of directory with the experiment
setenv fidSP fidSP.com    # script for conversion to nmrPipe
setenv in_file nls.in     # parameters of the NUS schedule
```

```
setenv selection_file nls.hdr_3  # nus schedule

setenv REC2FT recFT.com   # pipe procession of the indirect dimensions
setenv proc_out ft/test%03d.dat  # nmrPipe template for the final spectrum

# Definition of a small region of interest (ROI) in the direct dimension
setenv FST_PNT_PPM 8 # first point in ppm
setenv ROISW 0.5     # ROI size in ppm

#MDD related parameters
setenv MDDTHREADS 2   # maximal number of parallel processes
setenv NCOMP    25    # number of components per sub-region
setenv NITER    100   # number of iterations
setenv SRSIZE   0.1   # approximate size of sub-region in ppm
setenv MDD_NOISE 0.7  # factor for adding residuals to the MDD reconstruction
setenv lambda   0.01  # MDD lambda
setenv CT_SP    nnn   # parameter CT_SP in file nls.in is overridden
setenv CEXP     nnn   # parameter CEXP in file nls.in is overridden

# start actual calculations
mddnmr4pipeN.sh  1 2 3 4 5
```

In the script above, only first four parameters, which are typed in bold, are needed to be set explicitly. The remaining parameters are there mostly for display.

Table 1. Parameters recognized by *mddNMR* software.

Parameter name	**Default value**	**Meaning and references for examples**
CEXP		If set, override value in nls.in file
CS_alg	IRLS	CS algorithm: IRLS – iterative reweighted least squares, or IST – iterative soft thresholding
CS_lambda	1.0	CS regularization (default is ok for all studied cases). For IRLS, to make spectrum less sparse decrease lambda (applicable, if noise seems to be 'peaky', i.e. turned into sparse representation). For IST, CS_lambda has to be =<2 (values<1 make spectrum less sparse).
CS_niter	10	Number of iterarations for CS (default is ok for IRLS). Change to 100-10000 for IST.
CS_norm	0	Norm for CS IRLS algorithm: 0 - 1
CS_VE	y	Virtual echo on/off. If turned on (y), please set phase parameter to a proper value
CS_ZF	2	Frequency domain "over-digitization" in CS algorithm (best results for 2)
CT_SP		
DATAMAP_FILE		
DIM_MERGE		If defined, dimensions to merge. Used, e.g. to process 4D spectrum with 3D MDD
dtype		Defines which dimensions are "real" (r) and which are complex (c). For example, for the serial 2D HSQC relaxation experiment dtype should be set to rcr. Only first indirect dimension can be real. Direct dimension (last) is real by default.
f180		If set, overrides value in nls.in file
FID		Directory with experimental data
FIX_FREQ		Reserved
FIX_FREQ_FILE		Reserved
FST_PNT_PPM	10	Start (highfield) of the region of interest (ppm) in the directly detected dimension

ft4	./XYZA/ft4.xyza	Reserved
FT4DX	FTx.sh	Reserved
FTX_2D	./XYZA/ft4sp.xyza.2D	Reserved
FTXTREC	./XYZA/ft4trec.xyza	Reserved
in_file		Name and location of NUS parameter file.
lambda	0.005	Tikhonov regularization for MDD: 0.001-0.1
MAP_FACTOR	1	Reserved
MDD_DIR	./MDD	Reserved
MDD_FILE	./MDD/region	Reserved
MDD_NMR	…./mddnmr2.0	Location of the software directory
MDD_NMR_COM	…./mddnmr2.0/com	
MDD_NOISE	0.7	A factor that scales residuals of the MDD and IST calculations as they are added to the reconstructed spectrum: 0 - 1
MDD_STDERR	stderr	If set to a file name, all error terminal messages are redirected to the file
MDD_STDOUT	stdout	If set to a file name, all terminal messages are redirected to the file
MDD_WORK_DIR	.	Processing working directory
MDDRUNS	./regions	reserved
MDDTHREADS	2	Maximal number of threads, i.e. number of processes that can be run on your computer at the same time. The parameter relates to number of processors and cores on the computer.
METHOD	FT	Processing method: FT, MDD, CS
NCOMP	30	Number of MDD components per sub-region.
ndim		If set, override value in nls.in file
NI		If set, override value in nls.in file
NIMAX		If set, override value in nls.in file
NIMIN		If set, override value in nls.in file
NITER	300	Number of iteration for MDD calculations
NUS_POINTS		Reserved
NUS_TABLE_ORDER		Reserved
OVLP	3	Overlap between sub-regions in points
phase		If set, override value in nls.in file. **Important to set it properly for CS_VE='y'**
PHASE_ORDER		If set, reshuffle FID's inside each hyper-complex point, e.g. setting '1 3 2 4' results in swapping of the 2nd and 3rd FID's. The parameter is analogous to *-aqORD* flag in nmrPipe, which does not work for NUS processing. See also programs *fid_shuffle and ser_shuffle* .
Proc3D_X	shapeProc3D_%c.sh	Reserved
Proc3D_Y	shapeProc3D_Y.sh	Script for processing MDD shapes for Y dimension
Proc3D_Z	shapeProc3D_Z.sh	Script for processing MDD shapes for Z dimension
Proc4D_A	shapeProc4D_A.sh	Script for processing MDD shapes for A dimension
Proc4D_YZ	shapeProc4D_YZ.sh	Script for processing MDD merged shapes for YZ dimensions in a 4D spectrum
proc_out	./ft/tdrec%03d.dat	nmrPipe template for the final output

		spectrum
REC2FT	recFT.com	Script to process indirect dimensions in a 3D spectrum
RECHEAD	./XYZA/ft4sp.xyza	reserved
ROISW	6	Size of the region of interest in the directly detected dimension, ppm
RUNQUE		reserved
seed	2345	Random seed for MDD calculations
selection_file		NUS schedule file, typically comes with the experiment
SHAPEMAP		Co-processing: setting correspondence of dimensions between two experiments
SHAPEMAP_FILE	./MDD/regionMAP	Co-processing: file with the reference MDD shapes
soft_mode	p	Reserved
SPARSE		If set, overrides value in nls.in file
SpecParFile	./XYZA/_t.hdr	Reserved
SRSIZE	0.18	Approximate size of sub-region in ppm
SW		If set, overrides value in nls.in file
XDimSize	1	Reserved

The MDD solver has one parameter that mostly affects quality of the solution, namely, number of components (parameter NCOMP) per sub-region. Guidelines on correct setting for the parameter can be found in our papers. In most cases, however, a default value of ca 30 for a sub-region strip of 0.1-0.2 ppm (parameter SRSIZE) in the directly detected ^{1}H dimension is a good guess. In short, the number of components must be 20-50% larger then number of expected cross-peaks for 2D's and triple resonance backbone experiments, or number of diagonal peaks for 3-4D NOESY/TOCSY experiments. Note that number of components refers to a sub-region. NCOMP value must be sufficient for a sub-region with maximal expected number of peaks.

MDD shapes

The MDD model looks for an approximation of a M-dimensional spectral matrix by the sum of a small number of tensor products of one-dimensional vectors:

$$\boldsymbol{S}_{MDD} = \Sigma_\beta \; {}^{\beta}a \, {}^{\beta}F^{1} \otimes \ldots {}^{\beta}F^{M-1} \otimes {}^{\beta}F^{M} \qquad (1)$$

where the model spectrum $\boldsymbol{S}_{MDD}$ is the sum of fixed number of components N_c enumerated by index $\beta=1\ldots N_c$. Each component is given by the product of normalized vectors ${}^{\beta}F^{m}$ for every spectral dimension $m=1\ldots M$, referred to below as shapes, and the component amplitude ${}^{\beta}a$. The term shape is introduced here in relation to the spectral line shape; its several synonyms are present in the literature, i.e. loads, modes, factors, etc. Symbol $\otimes$ denotes the outer product operation, which produces M-dimensional matrix from M one-dimensional shapes.

A simple approach is to think about a component as the representation of a cross peak in a multi-dimensional spectrum. The shapes then are traditional line-shapes of the peak in all dimensions. The actual situation, however, is more complex, since the components do not always have a one-to-one correspondence to peaks. In general, a peak showing complex structure, e.g. in an E.COSY spectrum, may require several components for its description. It also can be the other way around, as in 3D NOESY spectrum - one component may accommodate several cross peaks. It is important to emphasize that the MDD (but not R-MDD) model does not make any assumptions about the shape vectors ${}^{\beta}\boldsymbol{F}^{m}$. Thus it can be equally well applied to data in the time or frequency domains, as well as combination of both.

The reconstructed spectra are produced by summation over all components (Eq. 1). Thus typically, dealing with the individual components is not needed. However, the shapes can be stored in both time and frequency domains.

```
# Storing MDD shapes using step 42
mddnmr4pipeN.sh  1 2 3 42
```

Upon completion of step 42, two files in XML format (see also USF3 format) are produced, which contain shapes in frequency and time domain. The shapes are also stored in nmrPipe format in directory SHAPES and can be viewed using nmrDraw. For example, columns in file SHAPES/sh_Y_03.dat contain the processed shapes for first indirect dimension from the 3rd sub-region of the spectrum.

Parallel calculation for faster MDD and CS processing

MDD and CS calculations may be lengthy. The computation time rapidly increases with amount of experimental data, number of iterations, number of components (for MDD), and size of the final spectrum (for CS). Calculations for different sub-regions are independent and can be performed in parallel on several CPUs that are available on one computer or within a local network. On one computer, parallel calculations are organized simply by setting parameter MDDTHREADS to the number of CPUs. In order to distribute calculations over a network, e.g. for a Linux cluster, step 3 may be performed off-line. First steps 1 and 2 are performed.

```
# Preparing input for sub-regions
mddnmr4pipeN.sh  1 2
```

This produces files *regions.runs* and *MDD/regionXX.mdd*, which are the only input for standalone MDD and CS solvers, *mddsolver* and *cssolver*, respectively. File *regions.runs* is a C-shell script. Each line in it contains a command for running calculations for one region. The commands may run in parallel on one computer or be distributed over the network together with *MDD/regionXX.mdd* for corresponding regions. When calculations are complete, the results, which are files MDD/regionXX.res or MDD/regionXX.cs, need to be collected to the original MDD directory followed by the spectrum reconstruction and final nmrPipe processing (steps 4 and 5).

```
# time domain spectrum reconstruction and final nmrPipe processing
mddnmr4pipeN.sh  4 5
```

GUI *qMDD* provides a simple possibility to distribute calculations using password-free *ssh* access. If box "*Send job to remote host*" in GUI is checked, the following lines are added to the master script *proc.sh* .

```
mddnmr4pipeN.sh  1 2
ssh login@host "mkdir -p tmpxxx/"
scp -C -r MDD regions.runs login@host:~/tmpxxx/
ssh login@host "cd tmpxxx/; queMM.sh regions.runs"
scp login@host:"~/tmpxxx/MDD/*.[rc]*" MDD/

mddnmr4pipeN.sh  4 5
```

The procedure runs script *queMM.sh* (part of *mddNMR* package) on a remote host, which distributes calculations over specified set of computers in the network. If the master script is ran from a directory, which is shared with other nodes in the network, *ssh* is not needed and the lines above are simplified to:

```
mddnmr4pipeN.sh  1 2
queMM.sh regions.runs
mddnmr4pipeN.sh  4 5
```

Note that the header of *queMM.sh* should be edited for every new network.

Examples

Examples, can be downloaded from http://mddnmr.spektrino.com/download and are described in the Appendix. For each example there is a compressed tar archive with two directories containing the spectrum and the script[s] for its processing (*.proc). For large data sets, the spectrum may be in a separate tar archive, which allows skipping download of large examples data files.

Table 1. List of examples:

	File name	Protein & citation	Experiment	Spectrometer	comment
1	gNhsqc_S.tgz	Azurin	2D HSQC	Varian	BioPack
2	ubi_ghn_co_S.tgz	Ubiquitin	3D HNCO	Varian	BioPack
4	284_hncoca.tgz	Ubiquitin	3D HNcoCA	Bruker	
4	57.tgz	Ubiquitin	3D HNCO	Bruker	
5	BPgnoesyNhsqc_S.tgz		3D 15N NOESY-HSQC	Varian	BioPack
6	az_HNCA_high_res.tgz	Azurin [1]	3D HNCA	Varian	BioPack*

(1) Jaravine, V.; Ibraghimov, I.; Orekhov, V. Y. *Nature Methods* **2006**, *3*, 605.

* The experiment was recorded in full, i.e. without NUS. This allows sparsifying of the data set and checking how quality of the spectrum degrades as fewer and fewer data are used for the reconstruction of the full fid matrix.

To speed up the calculations and save disk space in the examples, select (in *qMDD* or *proc.sh*) a narrow strip of about 0.2 ppm in the direct acquisition dimension. Scripts in the examples may serve as templates for processing of spectra of similar type. For example, scripts for the HNcoCA example can be used, after minor modifications, for most of the backbone experiments.

APPENDIX

I. NUS schedule

NUS schedule is typically produced by spectrometer software and is saved in a text file together with the experiment. Major spectrometer software vendors use program *nussampler*, which is also part of the *mddNMR* distribution. *nussampler*,can be evoked from a terminal command line:

nussampler nls.in

nussampler takes parameters from the input file *nls.in* (this name is fixed for most cases), generates a sampling scheme and writes it to file (*nls.hdr_3* on Varian or *nuslist* on Bruker). Both files *nls.in* and *nls.hdr_3* are needed for NUS spectral processing and must be stored together usually in the directory with *fid* or *ser* files. Typically the *nls.in* file is produced by executing the script from a GUI in spectrometer software, e.g. BioPack (Varian). The *nls.in* text file can be edited manually, and the above command for generating of a schedule can be typed from Unix command line.

Random number generation: now the program uses the "Mersenne Twister" pseudorandom number generator developed by by Takuji Nishimura and Makoto Matsumoto[1] (in previous versions it was function rand() from stdlib.h). The generator is tested to be platform independent; test generated the same nuslist on Linux 32-bit, Linux 64-bit, Windows 32-bit, Windows 64-bit.

File nls.in – setting NUS parameters

Each line of NUS parameter file *nls.in* start with a keyword followed by a list of parameters values. All the keywords are mandatory, but the lines order is not important. Below and in Appendix 2. several examples of input files are explained; location of *nls.in* file needs to be specified by parameter *in_file* in the master script (*proc.sh*).

```
file    nls.in
NDIM    3
seed    54321
SPARSE  y
sptype  poissongap
f180    nnn
CT_SP   nyn
CEXP    yyn
NIMAX   40 30 1
NIMIN   0 0 0
NI      7 30 1
SW      1824.818 2112.825  8389.262
T2      1.0 0.05  1
Jsp     0 0 0
phase   0 0 0
PERMUTE_NUSTABLE 2 1
FIRST_POINT_ZERO 1
nholes  -1
```

NDIM – number of dimensions, e.g. 3 for 3D spectrum

seed – seed for random number generator. If seed and other parameters are not changed output NUS schedule table will be always the same.

SPARSE – y|n - during the processing, the flag toggles processing of a real sparse spectrum ('y') vs sparsifying of a full spectrum ('n'). When setting up a BioPack experiment on a spectrometer, it toggles between NUS and regular sampling.

sptype - option for type of random selection (default:poissongap)

1 M. Matsumoto and T. Nishimura, "Mersenne Twister: A 623-Dimensionally Equidistributed Uniform Pseudo-Random Number Generator", ACM Transactions on Modeling and Computer Simulation, Vol. 8, No. 1, January 1998, pp 3-30. http://www.math.sci.hiroshima-u.ac.jp/~m-mat/MT/emt.html

sptype = poissongap | reg | shuffle | norepeatshuffle (only the first char is checked)

poissongap [default] Poisson Gap sampling, all gaps in accordance with the Poisson distribution with exponentially and J-coupling matched options applicable.

reg - regular =uniform sampling, if selected other options may be ignored
shuffle - incremental matched sampling, no check for point repetition (so schedule may contain point repeats).
ovewriteshuffle - old name, which is now equal to shuffle (kept for back-compatibility).
norepeatshuffle - incremental matched sampling, points are checked for no repetition, takes longer due to that, and sometimes will not find schedule especially for very short T2, e.g. < 1ms (increase T2, if so).

f180 – flags to specify 180 degree linear phase. Set y or n for every dimension. Direct dimension is the last one. The flag is important only for dimensions with CT_SP equal 'y'.
CEXP – y|n toggle R-MDD / MDD mode for a dimension, with 'y' time domain shape in the dimension is expected to be autoregressive. In other words, we assume that the FID in the dimension is a complex exponent. CEXP=y may be used, for example, for HNCO and HNcoCA experiments, but not for the NOESY's.
CT_SP - n|y toggles mirror image processing for dimensions with CEXP='y' . Set 'n' for the first indirect dimension. CT_SP have to be stet to 'y' for the constant-time second indirect dimension (typically 15N) in the triple resonance experiments.
NIMIN - min indices [for evolution increments]
NIMAX – full indirect sizes of the spectrum (you may put 1 for the (last) direct dimension)
NI - Multiplication of all NI values gives total number of (hyper-complex) points in the indirect dimensions. Put 1 for the direct dimension, which is the last number.
SW – spectral windows for dimensions in Hz
T2 – estimate of transverse relaxation times for each dimension for NUS. Use large value for dimension with constant-time (CT) evolution
Jsp – estimated resolved J-coupling for each dimension
phase – zero order phase correction for indirect dimensions. Used for dimensions with CT_SP=y

Optional parameters
PERMUTE_NUSTABLE 2 1 – for Bruker the table is permuted
FIRST_POINT_ZERO – if 1 makes the point with zero evolution times at the top of NUS schedule
nholes – if negative, no check for empty rows/columns in the R-MDD nD matrix.

Alternatively, the parameters are entered directly as arguments to the program, using option '-p', one by one, enclosed in '' (= sign can be omitted):

```
nussampler -p 'file=nls' 'NDIM=3' 'SPARSE=y' 'seed=4321' 'sptype=shuffle' 'f180=nnn' 'CT_SP=nyn' 'CEXP=yyn'
'NIMIN=0 0 0' 'NIMAX=120 224 1' 'NI=12 224 1' 'SW=1500 2500 10000'  'T2=0.035 1 1' 'Jsp=0 0 0' 'phase=0 0 0'
'PERMUTE_NUSTABLE 2 1' 'nholes -1'
```

File nls.hdr.3 – NUS table

The number of entries in the NUS table is equal to product of NI values for all indirect dimensions, i.e. NI x NI2 [x NI3 ...]. Each line consists of indexes for all indirect dimensions, i.e. two numbers per line for 3D experiment. The points are selected from a regular grid, thus values of the indexes in the table are integers from zero to NImax-1, NI2max-1,[NI3max-1 ...], respectively. By default, the NUS schedule is given in random order, that is evolution time points are not ordered. This allows stopping an experiment at any time without losing digital resolution.

Algorithm for the generation of the NUS schedule

NUS or sparse schedule suitable for MDD/CS processing selects for detection only a fraction of points from a complete Nyquist grid. Sampling on the grid is a special case of more general NUS that allows sampling at arbitrary selected time points. The selection of points in *mddNMR* software (see *nussampler*) is governed by a multi-dimensional probability density function for all indirect evolution dimensions. For example for a 3D experiment, the function is defined on a two-dimensional grid (t1, t2) determined by spectral widths and maximal acquisition times (t1max, t2max) in the two indirect dimensions. The distribution is obtained as a

product of the two envelopes, P(t1,t2) = P1(t1) x P2(t2). The envelope functions P1(t1) and P2(t2) are devised to match the signal intensity in the indirect dimensions for a particular system and experiment. Currently two possibilities are implemented: (i) mono-exponential relaxation- P (t1) = exp(-t/T2); (ii) modulation by the one-bond J-coupling- P(t2) = cos(t pi/J). The J-modulation can be combined with the relaxation decay. The transverse relaxation time T2 and value of the J-coupling are parameters of the procedure and are defined in the ".in" file. For a given probability distribution, we use the following procedure to generate the NUS schedules. First, a pair of integer indices is randomly selected that corresponds to the acquisition times (t1, t2). Then the pair is added to the sampling schedule table if the corresponding value of the probability distribution P(t1,t2) is larger than a randomly generated number ranging between 0 and 1; otherwise, the index pair is discarded. This process is repeated until the sampling table contains the requested number of data points for each step. Thus, a NUS schedule is a table of evolution delays (t1, t2) spanning maximal acquisition times and spectral widths in the indirect dimensions.

Algorithm for "Poisson Gap" random number generation

The algorithm for Poisson Gap random number generation is adapted from Knuth. It ensures that the gaps between points in the nD distribution follow Poisson distribution[2].

```
algorithm poisson random number (Knuth):
init:
  Let L ← e^(-λ), k ← 0 and p ← 1.
do:
  k ← k + 1.
  Generate uniform random number u in [0,1] and let p ← p × u.
while p > L.
return k − 1.
```

II. Unified Spectral Format (usf3)

Unified spectral format (USF3) is an XML format for compact storage and handling of spectra. It was originally intended to present results of spectra decomposition by programs *MddNMR* and *PRODECOMP*; it is also suggested as a general frame for compact storing of regular multidimensional and hyper-dimensional spectra of any dimensionality. USF3 is one of the data storage formats supported by CCPN. The latest formal description of the format (2011-05-13) can be found at http://www.ccpn.ac.uk/ccpn/projects/extendnmr/shape-data-format or requests from Rasmus Fogh (CCPN), Vladislav Orekhov (Swedish NMR Center), or Martin Billeter (University of Gothenburg).

III NUS Implementations on NMR spectrometers

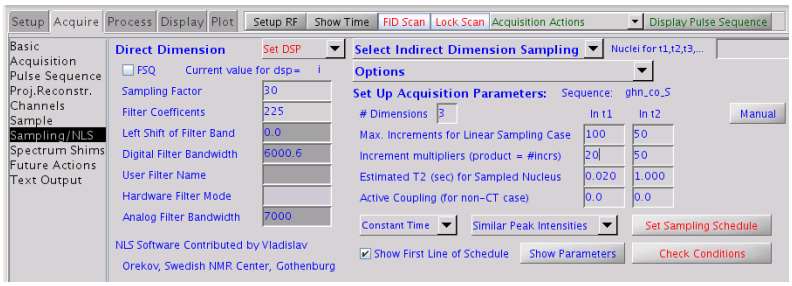
The flexibility of the pulse programming languages of VnmrJ and TopSpin has allowed straightforward and generic implementations of acquisition of NUS nD data. See vendor software manuals for exact details, e.g. documentation for BioPack or TopSpin of version 3.0 and higher. The NUS scheme is generically applicable for most, if not all, of existing pulse sequences. Uniformly incremented evolution delays in the pulse sequence are substituted by the values from the NUS table for every FID. New evolution delays are produced by multiplying the indexes by corresponding dwell time (i.e. 1/sw). For every combination of evolution times all FID's comprising one hyper-complex point must be recorded as one block. Thus, the block contains 2,4, and 8 FID's for 2D, 3D, and 4D spectra respectively. This corresponds to standard Varian/Agilent convention, but is different for old Bruker pulse sequences.

BioPack: Varian/Agilent spectrometer

The BioPack implementation (see Varian/Agilent documentation for details) features automatic creation of a new NUS version of any multi-dimensional pulse sequence by selecting *Sparse NLS Sampling* (figure below). After specifying the number of increments, etc., a single button is used to generate all the NUS files, including *.in file, table of increments (*hdr_3) via the *Set Sampling Schedule* button in the same page. Acquisition is performed in the normal manner. Saving of the data using the BioPack macro "BPsvf" also saves the NUS files and a script to permit processing by the MDD software.

---

2 https://en.wikipedia.org/wiki/Poisson_distribution



In the VnmrJ command line, NUS version of any experiment can be produced using macro *BP_NLSinit(<dims>)*, where <dims> stands for the number of dimensions in the experiment, thus for HNCO in the command line type

BP_NLSinit(3)

The macro prepares NUS version of the pulse sequence (look at *ghn_co_S.c*) and adds few additional parameters, which can be viewed in the "*text output"* tab using command *dgnls*. Set *SPARSE='n'* , *phase=1,2 , phase2=1,2* and use parameters *nt, ni, ni2* and command *time* to adjust time of the experiment. Note that parameters *ni,* and *ni2* define only total duration of the NUS experiment, the size of the sampling grid is defined by parameters *nimax* and *ni2max*. Set *SPARSE='y'* and calculate the sampling schedule by using macro

BP_NLSset

The macro creates two files in the experiment directory *~/vnmrsys/expXX* : *nls.in* and *nls.hdr_3*. The former contains parameters that are used for the generating the NUS schedule

```
nls.in
       file /home/bcbp/vnmrsys/exp4/nls
       NDIM 3
       SPARSE y
       seed 4321
       sptype shuffle
       nholes 0
       f180 nnn
       CT_SP nyn
       CEXP yyn
       NIMAX 50 50 1
       NIMIN 0 0 0
       NI 5 50 1
       SW 2500 3000 10000
       T2 0.02 1 1
       Jsp 0 0 0
```

File *nls.hdr_3* contains the sampling schedule, i.e. the list of selected points from the 2D grid (13C, 15N).

```
nls.hdr_3
       0 0
       36 45
       11 47
       3 26
       9 44
       8 38
       5 4
       …
```

Files *nls.in* and *nls.hdr_3* are needed to run and process the experiment.

Varian/Agilent spectrometer: default sparse sampling

The data acquired in Agilent default NUS mode (available in VnmrJ 4.0 and higher) that can be set up by clicking "Sparse" checkbox in the Sampling tab can also processed with mddnmr. Note, that this approach does not modify the pulse sequence code and is available also outside Biopack.

To make such data accessible with qMDD, a few modifications are required:

- set SPARSE parameter on (SPARSE='y')
- after the experiment, set *ni* in proc.sh script to the actual number of NUS points, not the full grid size
- rename sampling.sch to nls.hdr_3 in the data folder
- create nls.in file manually according to the template above

TopSpin: Bruker spectrometer

Consult the user manual for TopSpin 3.0 and later versions.

IV. Examples of MDD/CS processing

The examples include: (i) spectra with relatively small number of signals (up to 100-300) and limited dynamic range (up to ca 100), examples are 2D 13C HSQC, 3D HNcoCA and HNCO; (ii) spectra of NOESY-HSQC or TOCSY-HSQC type.

Example of backbone experiment

2D 15N HSQC (ubiquitin, BioPack)

This example illustrates MDD and CS processing of a 2D spectrum. Extract files from tar archive *gNhsqc_S*_30Jun2011.tgz*.tgz* and change directory to *gNhsqc_S/gNhsqc_S.proc.* Run master script *proc.sh* and look at the resulting spectrum using nmrDraw. Phase in the directly detected dimension requires correction, which can be done by changing one number in file *fidSP.com* . Namely, change line

```
| nmrPipe -fn PS -p0 0 -p1 0 -di    \
```

to

```
| nmrPipe -fn PS -p0 70 -p1 0 -di   \
```

and rerun *proc.sh* script. Open *proc.sh* in a text editor and change parameter METHOD to MDD or CS in order to compare results for different methods. To shorten the calculations you may reduce the region of interest in ppm's by changing parameters FST_PNT_PPM and ROISW. The spectrum has been recorded with random NUS 38%, i.e. 96 points were recorded out of 256. You may check how quality of the spectrum degrades as fewer data points are used for calculations. For this, set parameter NI in *pros.sh* to a value smaller than 96, e.g. for 25% NUS add line

setenv NI 64 1

By setting variable NI in *proc.sh* we override the values stored in file nls.in.

Processing of the spectrum can also be performed using GUI *qMDD*. Start *qMDD* in directory *gNhsqc_S* and (if asked) choose to overwrite existing files, open spectrum named *gNhsqc_S.fid,* process the spectrum by pressing button "Run", and look at the resulting spectrum in nmrDraw (press corresponding button in the GUI). In order to correct the phase in the directly detected dimension, you need to edit *fidSP.com* script, which can be done in "Advanced" by pressing *fidSP.com* button. You may, for example change line

```
-xP0 156.1 -yP0 0.0 \
```

to

```
-xP0 226.1 -yP0 0.0 \
```

3D HNCO (ubiquitin, TopSpin 3.0)

This example illustrate MDD and CS processing of a 3D triple-resonance spectrum. Extract files from tar archive 57hnco_30Jun2011.tgz and change directory to 57hnco */57.proc.* Run the master script *proc.sh* and check the resulting spectrum in nmrDraw. The spectrum was recorded with 25% NUS. Results for less data can be checked as described in the HSQC example above. The spectrum can also be processed using GUI *qMDD.* Note that phase for directly detect dimension is set to 45^o in *fidSP.com*, which is different from the value provided by *qMDD.*

3D HNCO (ubiquitin, BioPack)

This example illustrate MDD and CS processing of a 3D triple-resonance spectrum. Extract files from tar archive ubi_ghn_co_S_nls_30Jun2011.tgz and change directory to ubi_ghn_co_S_nls

/ubi_ghn_co_S_nls.proc. Run the master script *proc.sh* and check the resulting spectrum in directory *ft* and 2D projections in 1H.C13.dat N15.1H.dat N15.C13.dat . The spectrum was recorded with 6% NUS. Note the fid reshuffling from *phase2,phase* to *phase,phase2* in the master script[3]:

if(! -f fid) fid_shuffle ../ubi_ghn_co_S_nls.fid/fid fid 4 1 3 2 4

3D HNcoCA (ubiquitin, TopSpin 3.0)

This example illustrate MDD and CS processing of a 3D triple-resonance spectrum. Extract files from tar archive 284_hncoca_30Jun2011.tgz and change directory to 284_hncoca/284.*proc.* Run the master script *proc.sh* and check the resulting spectrum in directory *ft,* file *284.tf3* and 2D projections in *1H.C13.da,t N15.1H.dat, N15.C13.dat* . The spectrum was recorded with 9% NUS. Note that scripts *fidSP.com* and *recFT.com* have been adjusted relative to the setting provided by GUI *qMDD*. Thus, phase in the directly detected dimension is corrected in *fidSP.com.*

3D NOESY (BioPack)

This example shows processing of 3D 15N NOESY-HSQC spectrum of a 15 kDa protein. Extract files from tar archive BP_gnoesyNhsqc_S_30Jun2011.tgz and change directory to BP_gnoesyNhsqc_S/ BP_gnoesyNhsqc_S.*proc.* The experiment has been run in sparse level of 20%. Run master script *proc.sh* and check the resulting spectrum in directory *ft,* file *284.tf3* and 2D projections in *1H.C13.da,t N15.1H.dat, N15.C13.dat*. The spectrum can be also processed using GUI *qMDD*.

3D HNCA (azurin, BioPack, full spectrum)

This example shows MDD processing of a 3D spectrum, which was recorded in full. Thus, spectrum reconstructed from a small fraction of data points can be compared with the full spectrum processed using traditional DFT. A high resolution HNCA experiment[4] was recorded for globular 14 kDa protein azurin. Extract files from tar archive az_HNCA_high_res_30Jun2011.tgz and change directory to *az_HNCA_high_res.proc.*

Since the spectrum was recorded without NUS, GUI *qMDD* processes it as a regular full spectrum. In order to "sparse" the spectrum, in *qMDD* choose **not** to overwrite the existing files. In *Advanced* display, check script *proc.sh* (also shown below) and file *nls.in*. Script *proc.sh* produces NUS schedule (10%) "on the fly" by running program *nussampler* on *nls.in* file. Note that parameter SPARSE in *nls.in* file is set to "n". This tells *mddNMR* that sparse data are to be extracted from a full spectrum. For spectra recorded in real NUS mode, the parameter must be set to "y". Below you see content of commented script *proc.sh:*

```
#!/bin/csh

setenv FID ../az_HNCA_high_res  # input experiment (without .fid )
setenv in_file nls.in           # local copy of nls.in with NUS schedule parameters
setenv selection_file nls.hdr_3 # NLS schedule file to be produced by nussampler

setenv fidSP  fidSP.com # nmrPipe script for fid conversion and DFT of the direct dim
setenv REC2FT recFT.com # nmrPipe script to process reconstruction after mdd calculations

# MDD related  --------------------
setenv METHOD  MDD      # toggle MDD mode
setenv MDDTHREADS  2    # allows up to 2 simultaneous calculations
setenv FST_PNT_PPM 8.75 # leftmost point of region of interest (ROI)
setenv ROISW 0.15       # full ROI size (ppm)
setenv SRSIZE 0.1       # recommended size of sub-region
setenv NITER 50         # number of iteration
setenv NCOMP 30         # default number of components for one sub-region
setenv lambda 0.002     # Tikhonov regularization parameter

setenv MDD_NOISE 0.2 # scales residuals as they are added to the reconstructed spectrum
setenv proc_out ft/test%03d.dat  # nmrPipe template for the final 3D spectrum

#####################################################################
################ end of definitions ##################################
```

3 Since processing of the directly detected dimension is performed by *nmrPipe* in 2D mode (script *FTx.sh.2D*), this reshuffling cannot be performed by *var2pipe*. It cannot be done either by setting *mddNMR* parameter *PHASE_ORDER* to 1 3 2 4, because decoding of the Echo-Anti-Echo signal is performed by *nmrPipe* prior to the processing by *mddNMR*.

4 The experiment was used for our publication Jaravine, V., et al. Nature Methods 2006, 3, p 605

```
nussampler $in_file   # the spectrum has been recorded in full and is "sparsed"
                      # for processing; so calculate NUS table here
                      # check/edit file nls.in for the NUS schedule parameters

# process spectrum with mdd
mddnmr4pipeN.sh   1 2 3 4 5

# make 2D projections of the final 3D spectrum
proj3D.tcl -in $proc_out


## uncomment the following lines to process the full spectrum for comparison
# FTx.sh XYZA/FTx.xyza                    # make FT for directly detected dim for ref
spectrum
# recFT.com XYZA/FTx.xyza ft/ref%03d.dat   # FT of Y and Z dimensions
# cd ft
# proj3D.tcl -in ref%03d.dat            # make 2D projections of reference spectrum
```

pseudo-3D T2-HSQC (large protein, TopSpin 3.5)

This example shows both MDD and CS processing on the serial 2D data with 8 different settings of the relaxation delay (8 points in the real pseudo-dimension). Note the parameter ***dtype*** set to *rcr* which describes which of the dimensions are real. NUS schedule describes both indirect time dimension and relaxation dimension. Only 25 NUS points per relaxation delay are used.

While CS divides data into separate subsets for each relaxation delay, MDD processes whole pseudo-3D matrix jointly. Note, that the conversion script has QF mode in first indirect dimension and that recFT script is one-dimensional.

V. Additional tools in the package

Programs

fid_shuffle

program to shuffle (=re-order) 1D FIDs in Varian fid
Use: fid_shuffle <input fid> <output fid> <array size> <n1> <n2> ... <n arr size> }
<input fid> - data file
<array size> <n1> <n2> ... <n arr size> - size of reshufled block and new order of 1D's
Example 3D phase2,phase to phase,phase2 : <input fid> <output fid> 4 1 3 2 4
Example 4D phase3,phase2,phase to phase,phase2,phase3 : <input fid> <output fid> 8 1 5 3 7 2 6 4 8

ser_shuffle

program to shuffle (=re-order) 1D FIDs in Bruker ser
Use: ser_shuffle <input ser> <output ser> <FID size in 4 byte words> <NF -FID's in block> <n1> <n2> ... <n NF> }
change order of FID's within block; initial order is 1 2 3 4 5 6 7 ...
Example: ... 4 1 3 - only 1st and 3rd FID's out of 4 are passed to the output
Example 3D phase2,phase to phase,phase2 : ... 4 1 3 2 4
Example 4D phase3,phase2,phase to phase,phase2,phase3 : ... 8 1 5 3 7 2 6 4 8

mddsolver, cssolver

Standalone MDD and CS solvers respectively.

Shell scripts

queMM.sh

allows to do parallel calculations of step 3 on multi-CPU localhost or a network cluster (over password-free ssh); Edit parameters in this script setup for your local network.

recFT.com

default template nmrPipe script, for processing of YZ dimensions; it is normally copied to each *.proc directory and manually edited, e.g. to set indirect phases if different from defaults (0 0).

nussampler

NUS generator; described above

nus2pipe

A python script that converts Agilent and Bruker parameters to nmrPipe conversion script *fidSP.com*. Run the script without arguments to get usage info.

VI. Copyright and Legal Information

Date: May 26, 2011

DESCRIPTION
MddNMR is a program for processing uniformly and non-uniformly sampled (sparse) NMR spectra.
The following is legal information pertaining to the use of the MddNMR program. It applies to all MddNMR source files, executable (binary) files, configuration and documentation files contained in the official MddNMR archives. (Certain portions refer to custom versions of the software, there are specific rules listed below for these versions also.) All of these are referred to here as "the software".

THIS NOTICE MUST ACCOMPANY ALL OFFICIAL OR CUSTOM MddNMR FILES. IT MAY NOT BE REMOVED OR MODIFIED. THIS INFORMATION PERTAINS TO ALL USE OF THE PACKAGE WORLDWIDE. THIS DOCUMENT SUPERSEDES ALL PREVIOUS LICENSES OR DISTRIBUTION POLICIES.

IMPORTANT LEGAL INFORMATION
While the use MddNMR is essentially free of any costs for noncommercial purposes, commercial users and software developers, who wish to bundle MddNMR with other software, will be asked for support of the research and development of MddNMR. For commercial purposes of MddNMR please contact the copyright holders. Permission is granted to use the MddNMR program and all associated files in this package for making calculations. Use of the software for academic and educational purposes is free. The user retains all rights to the results and may use them for any noncommercial purpose. The following legal information exclusively concerns distribution and use of the software for noncommercial purposes.

This software package and all of the files in this archive are copyrighted by the authors, which are represented by Prof. Vladislav Orekhov for distribution, copyright and other legal issues (VO). The software may only be distributed and/or modified according to the guidelines listed below. The spirit of the guidelines below is to provide the MddNMR package freely to as many users as possible, prevent MddNMR users and developers from being taken advantage of, enhance the life quality of those who come in contact with MddNMR. This legal document was created so these goals could be realized. You are legally bound to follow these rules, but we hope you will follow them as a matter of ethics, rather than fear of litigation.

No portion of this package may be separated from the package and distributed separately other than under the conditions specified in the guidelines below. This package may only be bundled in other software packages with an explicit permission of the copyright holders. This package may only be posted in the Internet and/or included in software compilations using media such as, but not limited to, floppy disk, CD-ROM, tape backup, optical disks, hard disks, or memory cards with the explicit permission of the copyright holders.

CUSTOM VERSIONS
With a separate agreement on the custom version (CVA) a user may be granted the privilege to modify and compile the source code (SC) for their own use in any fashion they see fit. What you do with the software in your home or lab is your business, however, in such cases the activity is usually limited by the agreement or defined by the collaborative project. If it is allowed in the CVA and the user wishes to distribute a modified version of the software, documentation or other parts of the package (here after referred to as a "custom

version") they must follow the guidelines listed below. These guidelines have been established to promote the growth of MddNMR and prevent difficulties for users and developers alike. Please follow them carefully for the benefit of all concerned when creating a custom version. Without the explicit permission of the copyright holders you may not: make any portion of the SC public via Internet or other media, transfer the SC or its parts to other people of organizations, incorporate any portion of the MddNMR source code in any software other than a custom version of MddNMR. Authors who contribute source to MddNMR may still retain all rights to use their contributed code for any purpose as described below. The user is encouraged to send enhancements and bug fixes to the MddNMR authors, but the authors are in no way required to utilize these enhancements or fixes. By sending material to the authors of MddNMR software, he authorizes the MddNMR authors to use the materials any way they like. The contributor still retains rights to the donated material, but by donating you grant equal rights to the MddNMR authors. The MddNMR authors don't have to use the material, but if we do, you do not acquire any rights related to MddNMR. We will give you credit if applicable.

CONDITIONS FOR DISTRIBUTION OF CUSTOM VERSIONS
The permission to distribute compiled custom version of the software may be granted in advance with explicit permission from VO. Note that no part of the MddNMR source code can be distributed or made public. Typical conditions include but not limited to the following: - mark your version clearly on all modified files stating this to be a modified and unofficial version; - Include clear and obvious information on how to obtain the official MddNMR. - Include contact and support information for your version. - Include all credits and credit screens for the official version. - Include a copy of this document. The MddNMR authors are not obligated to provide you or your users any technical support.

GENERAL RULES FOR ALL DISTRIBUTION
All requests to acquire the software should be sent to VO, who normally distributes the software on behalf of all co-authors. The permission to distribute this package under certain very specific conditions is granted in advance to other persons or organizations, provided that the above and following conditions are met. The software archives must not be renamed or re-archived using a different method without the explicit permission of the authors. The full software package, as described in the next section, must always be distributed. All forms of commercial and non-profit distribution are only allowed with explicit permission of the copyright holders represented by VO. Clear reference to the copyright holders (at least to VO) must be present in any description/synopsis of software. The copyright holders reserve the right to withdraw distribution privileges from any group, individual, or organization for any reason.

DEFINITION OF "MddNMR PACKAGE" MddNMR is distributed as a number of archives containing executables, installation scripts, examples, and documentation. MddNMR is officially distributed for PC (LINUX) and Intel MAC (OS X 10.4 and later). Other systems may be added in the future. Distributors may support different platforms but for each platform they support the full package must be distributed.

DISCLAIMER
This software is provided as is without any guarantees or warranty. Although the authors have attempted to find and correct any bugs in the package, they are not responsible for any damage or losses of any kind caused by the use or misuse of the package. The authors are under no obligation to provide service, corrections, or upgrades to this package.
[End of Legal Information]