Available online at www.sciencedirect.com
ScienceDirect
Journal of Magnetic Resonance 187 (2007) 19–26
JMR Journal of Magnetic Resonance
www.elsevier.com/locate/jmr

matNMR: A flexible toolbox for processing, analyzing and visualizing magnetic resonance data in Matlab®

Jacco D. van Beek *

Physical Chemistry, ETH Zurich, CH-8093 Zurich, Switzerland

Received 9 March 2007; revised 23 March 2007
Available online 7 April 2007

Abstract

matNMR (matnmr.sourceforge.net) is a toolbox for processing, analyzing and visualizing magnetic-resonance data within the Matlab® environment (www.mathworks.com) that aims for control, flexibility and extendability. Processing can be done using either a graphical user interface or with command-line scripts, both of which allow user-defined processing or analysis functions to be incorporated at any time. The direct access to data points during processing, and the extensive library of mathematical and visualization routines provided by Matlab, afford the high degree of control and flexibility needed in modern magnetic-resonance research.

Keywords: Magnetic resonance; Processing toolbox; Matlab; Visualization; Data analysis

1. Introduction

Nowadays, a large number of software packages are available for processing nuclear magnetic resonance data [1], ranging from packages supplied by spectrometer manufacturers [2–5] to commercial packages [6–14] and free software [15–22] often made available to the scientific community by single researchers or groups. The algorithms used in these programs are typically well-known and the processing capabilities are often similar, although most packages will try and target a specific user base by including some specialized processing or analysis features, implementing near-complete automation features or even reduced functionality for increased simplicity. Here, we describe matNMR, an open-source processing toolbox that works within the Matlab® [23] environment.

Matlab® is an established interactive matrix-oriented computation and visualization environment, which has found a broad range of application within scientific and engineering communities. It is currently available for various different computer platforms. The extensive library of standard mathematical computation and visualization routines make it ideally suited for a processing toolbox that wants to be flexible, easily extendable, and offer a high degree of control. Matlab offers matrix-optimized routines for most processing actions required in magnetic resonance and allows creating extensive graphical user interfaces (GUIs). More specialized routines are available through additional toolboxes or can be programmed by the user. The interactive Matlab *workspace* provides a flexible interface between the user and the computer memory, which allows direct access to all variables that are created during a session.

Matlab provides various tools for direct access to matrix-based data, which is essential for preparing experimental data for quantitative analysis and subsequently analysing the data. matNMR provides an easy interface to processing of 1D and 2D spectra and the many tools in Matlab for graphical visualization. Various common and specialized processing features, that are needed for 1D and 2D magnetic resonance data, are implemented and multiple NMR data formats are supported. Storing of the processing history with

* Fax: +41 44 632 1621.
E-mail address: jabe@nmr.phys.chem.ethz.ch

1090-7807/$ - see front matter © 2007 Elsevier Inc. All rights reserved.
doi:10.1016/j.jmr.2007.03.017

the spectrum or FID is useful for *a posteriori* accountability and macros can easily be defined for standardized processing protocols. matNMR is designed for efficiently processing and plotting series of datasets.

Over the past years matNMR has slowly evolved into a complete processing, analysis and visualization toolbox. With more than 15,000 downloads in the last 7 years, matNMR has clearly shown its value to the field of NMR. This contribution is aimed at presenting the package to the wider NMR community. It is not intended to show all features and implementations of published algorithms; only a short overview of features is given instead. For more detailed information the reader is referred to the HTML manual, available on the website, http://matnmr.sourceforge.net.

2. Features

2.1. General implementation

Matlab is an interpreted language that has specialized in numerical calculation and visualization. Common to other interpreted languages, it provides a command-line interface (shell) that allows direct operations on numbers, arrays and variables, but it also forms a complete high-level programming environment. The environment created by Matlab is a virtual workspace in which variables are directly accessible to the user and can be changed at will. The memory management is dynamic and all taken care of by Matlab, in contrast to compiled programming languages where the programmer needs to actively allocate and de-allocate memory. A wide variety of commonly-used data types is available to the user and new object classes may also be defined. Matlab offers various options for programming: *functions* have a defined number of input and output parameters and generally do not share variables with the Matlab workspace. *Scripts* have no input or output parameters and directly access the workspace. Finally, interfacing to and from external C or Fortran programs can be implemented, where the programs do not access the workspace directly and requires external compiling.

In order to achieve maximum flexibility for the user, matNMR has been written mainly in the form of scripts. This allows for the application of user-defined routines at any time, to make immediate changes to the code without the need for recompiling and, if needed, direct access to matNMR-internal variables during processing. For processing speed matNMR relies on optimized routines provided by Matlab; matNMR mainly performs some book keeping to build the user interface. Note that programming in high-level languages, like Matlab, is comparatively easy, which means a low barrier for non-experts to implement new processing or analysis functionality.

All processing is done in memory, which works well for small to medium-sized datasets. On a typical modern personal computer with 1 Gb RAM memory, hypercomplex 2D spectra up to 2048×2048 points can be readily processed. When limiting the number of "undo" steps, larger matrices are feasible. Currently, 3D spectra cannot be processed with matNMR (only series of 2D spectra organized in a 3D matrix). With the onset of larger memory in modern desktop computers, full 3D processing in memory in fact becomes possible and is planned to be implemented in matNMR.

2.2. An intuitive interface

matNMR offers both GUI and script-based processing. The user-friendly graphical interface allows all processing features to be accessed from a menubar and control buttons. After each action the data is redisplayed on the screen. The GUI consists of a main window from which all processing is done (Fig. 1a) with separate modules to perform various analyses, e.g. spectral deconvolution, fitting of exponential curves or tensors (Fig. 1b), and a viewer for 2D data with a variety of high-level 2D and 3D contour, surface and line plots (Fig. 2). Usage of the GUI principally requires little knowledge of Matlab. Script-based processing allows incorporation of standard processing algorithms into user-defined routines and, hence, does require a working knowledge of Matlab.

2.3. Interaction with Matlab

It is useful to stress that matNMR is a toolbox that interfaces between the user and the Matlab workspace: whilst some software packages are designed to process NMR data in a more or less fixed fashion, matNMR makes no restrictions on which functions to use, nor in what order. It is designed to simplify processing in the workspace, without getting rid of the valuable features of the workspace. For example, importing datasets from disk using the GUI will result in them being stored in the workspace as variables. There they can be manipulated by the user or be loaded into the GUI for processing, which is a separate step. Although single datasets can also be read and stored directly from disk into the GUI, thus limiting the interaction with Matlab to a bare minimum, the workspace is a prominent aspect of matNMR. Advanced users will appreciate the features Matlab has to offer and will be able to integrate matNMR effectively into user-defined routines.

A powerful feature of interpreted languages is the evaluation of strings into commands. For example, a string 'fft(UserFunc1(UserFunc2(UserVar(1:3:end))))' will evaluate into the execution of two user-defined functions (called UserFunc1 and UserFunc2) and a fast Fourier transform (fft). UserFunc2 operates on the variable UserVar, indexed at points defined by the range $[1, 4, 7, \ldots, \text{end}]$. UserFunc1 then operates on the output of UserFunc2 and the fft routine on the output of the UserFunc1 routine. This highly flexible way to generate commands, through interpretation by the Matlab command-line parser, is supported throughout matNMR, ensuring an optimal integration of matNMR into the Matlab environment.

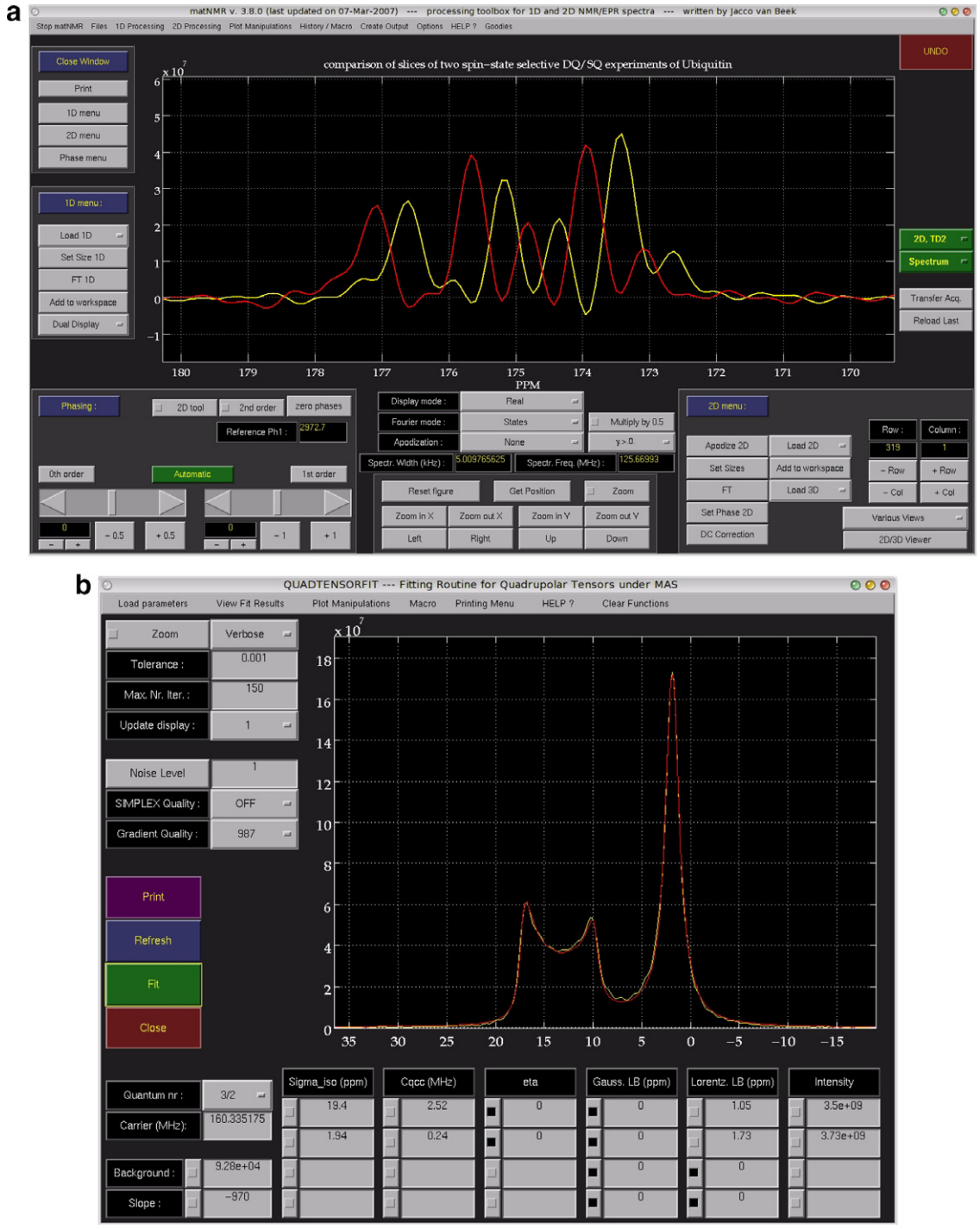

Fig. 1. (a) Layout of the main processing GUI in matNMR: a small number of functions can be accessed directly by pushing the corresponding buttons, others are accessible from the menubar or from the context menu (right-clicking the mouse). The plot shows a comparison of slices taken from two spin-state selective DQ/SQ correlation spectra of Ubiquitin [33], indicating that the two lines of the J-doublet can be selected. The echo/anti-echo time-domain data was converted to a STATES modulation, processed with 4096×1024 points, and referenced externally using a spectrum of adamantane. (b) The quadrupolar-tensor fitting routine showing a fit of the trigonal and tetrahedral sites in borax. The ^{11}B spectrum spectrum of borax was taken at 13 kHz MAS, 11.75 T field strength and was referenced to boric acid at 19.6 ppm.

2.4. Processing strategy

matNMR uses a consecutive plane-directed processing approach, which typically consists of several actions applied successively to one dimension of a spectrum, possibly followed by several actions applied to the second dimension. No transpositions are needed for 2D processing as the user can readily change the dimension to operate upon and access individual rows and columns. In contrast to several other software packages that focus on automated

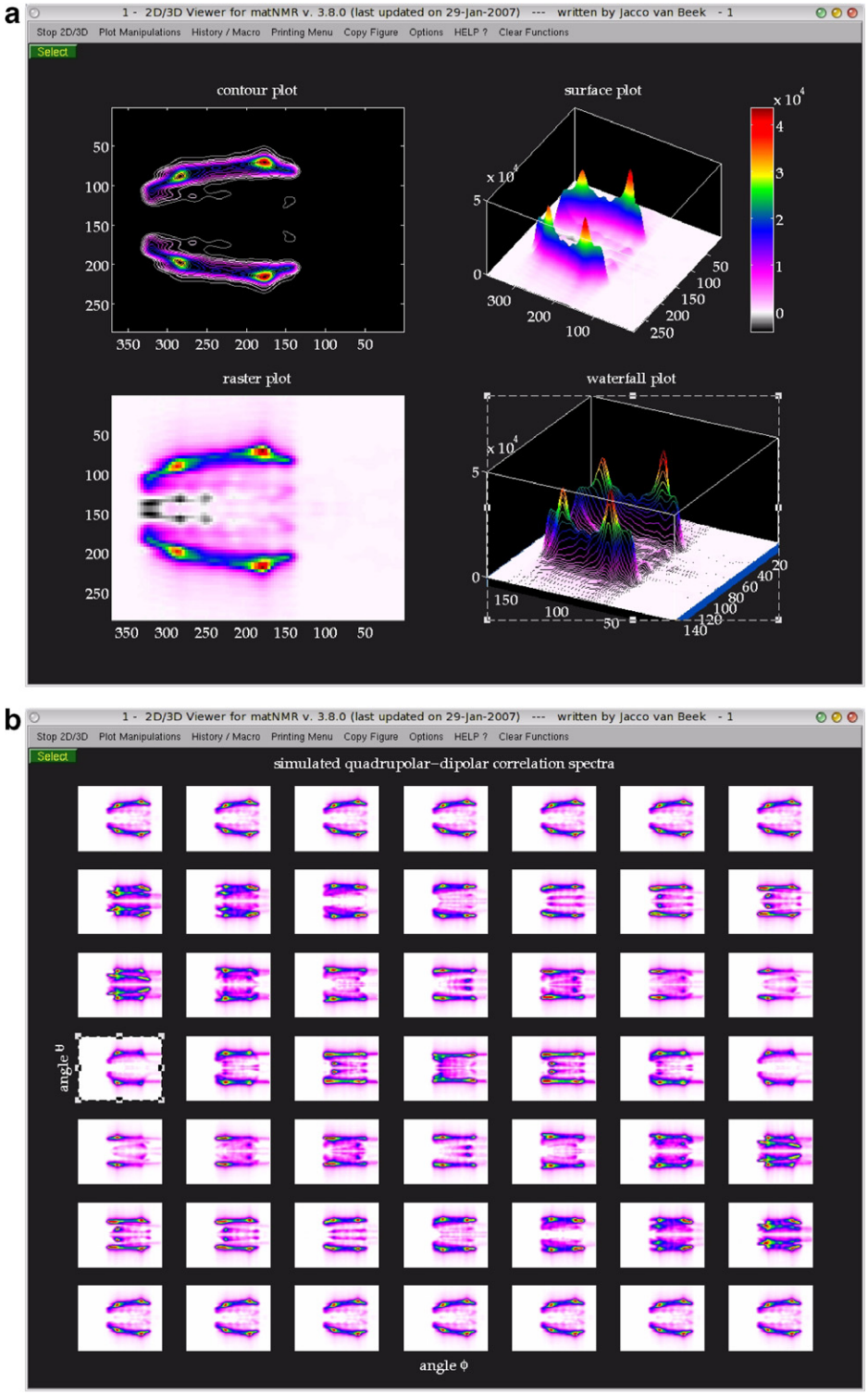

Fig. 2. Example of automation and visualization in matNMR using the 2D/3D viewer: (a) multi-plot configuration with various plot types. (b) Series of 49 simulated 2D FIDs that was loaded into Matlab and concatenated into a 3D matrix. The spectra were processed using a single macro and displayed as 3D surface plots into a multiplot window with a chosen grid of 7×7 axes. By using macros and the array of 2D matrices, this took only a fraction of the time needed to process and plot each single 2D spectrum separately.

processing, several features were implemented interactively. We feel this affords more control during processing, especially for spectra with wide lines or distortions, and ultimately yields more accurate processing.

2.5. Processing features

matNMR supports many standard processing techniques required for modern 1D and 2D processing and

reads multiple NMR data formats from commercial spectrometer manufacturers. A summary of features is given in Table 1. For most common NMR file formats, basic spectral information can be read from the standard parameter files. Furthermore, processed spectra can be imported for both Bruker and Varian/Chemagnetics SpinSight formats and also ASCII data from SIMPSON [24] simulations can be read directly. In addition, Matlab itself supports reading of generic binary and ASCII files, in case other formats are required. Several well-known schemes for phase-sensitivity are supported for 2D spectra (Table 1). Routines are provided to convert different data ordering formats, as used by the various spectrometer manufacturers, into one that is used by matNMR.

2.6. Spectral analysis

A small number of routines for standard spectral analyses are implemented in matNMR: spectral deconvolution with mixed Gaussian/Lorentzian lineshapes, (multi-)exponential fitting of relaxation or diffusion curves, fitting of static CSA tensors from powder patterns, fitting tensors from magic-angle spinning (MAS) sideband amplitudes and fitting of half-integer quadrupolar tensors under MAS from the central transition lineshape. Fig. 1b shows an example of the interface for the fitting routines. All parameters relevant to a fit are displayed in the window and the experimental and fitted data are plotted to allow additional visual inspection of the quality of fit. All routines use the non-linear fitting algorithms supplied by Matlab, typically using a downhill simplex method for coarse optimization, followed by a gradient minimization.

Gaussian and Lorentzian peaks and exponential curves are readily calculated, but the tensor-fitting routines are computationally more demanding. Efficient routines have been implemented that yield results equivalent to full density-operator simulations for single spins, by using analytical expressions for frequencies and amplitudes. Static powder patterns are calculated in frequency space using an analytical lineshape function [25]. Spinning sideband manifolds and second-order quadrupolar patterns are calculated through powder averaging of analytical expressions for the frequency under MAS. Using efficient sampling schemes the calculation of a spectrum typically takes on

Table 1
Selected features implemented in matNMR

Features	Details	Comments
Supported file formats	Varian	VNMR, SpinSight
	Bruker	TopSpin, XWinNMR, UXNMR, WinNMR, Aspect 2000/3000
	Chemagnetics	SpinSight, CMXW
	JEOL	JEOL generic format
	TecMag	MacNMR, NTNMR
	SMIS	
	SIMPSON ASCII	Raw ASCII data is readable directly by Matlab
Processing features	Removal Bruker digital filter	
	Data shifting	Both data shifting and resetting of time-origin for whole-echo experiments
	DC offset correction	
	Solvent deconvolution	
	Apodization	Exponential, Gaussian, Cos^2, Hamming, includes functions for whole-echo acquisition
	Linear prediction	Forward/backward, two algorithms
	Fourier transform	Real, complex, TPPI, Bruker qseq, sine FT, inverse FT. For 2D: STATES, TPPI, STATES-TPPI, Echo/Anti-Echo
	Phase correction	Automatic or interactive
	Baseline correction	Interactive
	Shearing transformation	Frequency or time domain
	Various projections	Sum, (anti-)diagonal, horizontal/vertical stack plots, skyline
Macros/History	Full processing history	Recorded and stored with every dataset
	Macros	For processing and plotting actions
Analysis	Spectral deconvolution	Mixed Lorentzian/Gaussian lineshapes
	Relaxation and diffusion-curve fitting	Flexible (multi-)exponential functionals
	CSA tensor fitting	From static spectra or MAS sideband amplitudes
	Quadrupolar tensor fitting	Central transition under MAS
2D/3D Viewer	Multi-window multi-plot routine	
	Various plot types	Contours, 3D surface plots, 2D/3D bar plots, stack, raster, volume
	High-level graphics	Zoom, 3D rotation, color maps, shading, lighting
	Various others	Peak picking, conditional probabilities

the order of a second to finish. The exact timings depend on the numerical accuracy of the powder integration, which can be defined manually.

2.7. Visualization, output and automation

Matlab provides an extensive environment for state-of-the-art graphics using its own two rendering algorithms or OpenGL®. A variety of plot types (multiple contour and 3D surface plots, line and bar plots, etc.) can be used with zooming, panning, rotation, color maps, lighting and shading options available to enhance plots. matNMR provides a convenient interface to plot 2D spectra in commonly used forms. For 3D spectra Matlab supports several high-level volume-visualization routines. From the 2D/3D viewer, shown in Fig. 2, most functions required to change plots can be accessed, allowing changes to colormaps, shading, all axis properties, tick properties, etc. Multiple pre-defined grids of subplots can be chosen from, making this multi-window multi-plot routine a versatile tool for creating high-quality output for publications.

Matlab allows plots to be printed directly, or converted into PostScript® (ps or eps), multiple bitmap graphics formats or a variety of other formats (e.g. Windows metafile, WMF) for cross-platform exchange. The matNMR GUI provides a convenient interface to many of these output options. It also achieves (near-)WYSIWYG (what-you-see-is-what-you-get) behavior, which, unfortunately, is not a standard feature in Matlab.

Rendering of graphics is typically slow compared to processing data and the generation of a plot. Limiting the number of rendering steps can bring a significant gain in time, especially for multi-plot windows. In matNMR both processing and plotting macros may be recorded for automated processing and plot manipulations to limit the number of graphics rendering steps. The recorded actions are translated by matNMR into an array of numbers and these can be stored in the workspace, and on disk, as any other variable. Macros can be reprocessed interactively or automatically.

Processing a series of spectra, either in blocks or different spectra, is conveniently supported in the GUI: loading, grouping, processing and plotting the series is efficiently implemented through special menu options and the use of macros. Series of 2D spectra can be grouped into a 3D matrix and macros can be applied to all subspectra. Fig. 2b shows an example of a set of 49 simulated 2D spectra, processed using a single macro and plotted from a 3D matrix in a chosen grid of 7×7 axes.

2.8. Processing history

Often, a record of the exact data processing is not kept. For applications where data is analyzed quantitatively a processing record is vital, however, and also as a control mechanism for students' work it can be quite useful. matNMR automatically records all processing steps applied to a given dataset. The processing history can be printed to go into a lab journal or stored alongside the final spectrum or the FID. Thus, true accountability is achieved.

2.9. License and distribution

The source code for matNMR is published under the GNU general public license and may be used, altered and distributed accordingly. The project is currently hosted at http://matNMR.sourceforge.net, where regular updates are posted. The code, which consists of readable text files, is small (<600 kb compressed using gzip, excluding HTML manual) and runs on all platforms that are supported by Matlab. No compilation steps are required.

This paper describes features for matNMR 3.8.0, which runs on Matlab 6.5 and higher. Previous releases are available on the website that run on older versions of Matlab. Note that currently matNMR does not work with free Matlab-like software packages like Octave [26] and SciLab [27].

3. Discussion

In light of the many choices available for NMR data processing, a comparison between matNMR and other packages is useful. matNMR was developed because of a need for detail and control during processing and visualization, which was not provided by the spectrometer software. For many uses its features may not necessarily be appreciated, though, and other solutions may be preferred: (1) the available spectrometer software provide all-round processing features. Their advantage is that they are well-known throughout the NMR community, because most people use them to some extent, but they typically suffer from lack of flexibility and functionality and it is often unclear what algorithms are used. (2) Most of the commercial packages mentioned above [6–14] focus on effortless GUI-based processing and several have implemented data archiving schemes for spectra and even drawings of molecules, which can be useful for industry applications. Processing with matNMR or script-based software is inherently more demanding, because the user is required to know when to apply what function. (3) The best-established free NMR processing package to date is certainly NMRpipe [20]. In the last decade it has become a standard for processing of high-resolution multi-dimensional NMR spectra of proteins. It uses script-based processing of up to 4-dimensional spectra and some accompanying software is provided for visualization and analyses of spectra. Furthermore, indicative of its success, a number of algorithms and analysis tools have been developed by the NMR community to work within the NMRpipe framework or with NMRpipe-processed data.

Naturally, comparable features are found in most of the processing software and, all things equal, much depends on personal preference. The main advantages of matNMR over other processing software are the control and flexibility

offered during processing, and the extent of Matlab's functionality. The former stems from the fact that the data is not hidden to the user, but is freely-accessible at all times through the Matlab workspace. Direct access to the data is not granted in most other packages. The ease of access to the data is also much enhanced by the GUI and presents a clear advantage over script-based processing software. In selected cases the user may also wish to know what the processing software does exactly. Most other packages do not provide the source code, and if so it is often difficult for non-specialists to read such code. The high-level programming language of Matlab simplifies this considerably, and hence also allows non-specialists to add new routines. Whilst such features may not be needed for standard applications, in our experience they have been vital for quantitative analysis of solid-state NMR data: accurate processing and correction of spectral artefacts require detailed visual inspection.

Working with multiple programs and computer platforms for processing and analysis can be very tedious. Hence, the matNMR project offers a good basis for further extension of its functionality by the NMR community to obtain an all-round open-source platform-independent research tool. Such extensions could, for example, include 3D processing and more specialized analysis routines that today are achievable by e.g. completely general spin-simulation software [28–31] or by one of the many other specialized tools currently available [1].

In several ways the choice of Matlab as a platform is both a strength and a potential weakness: (1) Matlab provides a low-barrier access to an extensive library of advanced mathematical and graphical routines. Conversely, the speed of calculation may not be as high as for problem-specific optimized code. (2) The workspace affords much-desired flexibility but as always such freedom can also lead to chaos if the user creates too many variables in a session and the overview is lost. (3) Generally, processing in RAM memory is fast, but the maximum size of the datasets that can be processed is thus limited. Also, Matlab's memory management is such that the user must be aware of the memory limitations of the computer: processing many large datasets can lead to very large memory usage by Matlab, especially if multiple ''undo-steps'' are configured (since each ''undo-step'' stores a copy of the dataset in the workspace). (4) Matlab provides excellent graphical capabilities, but its rendering speed for complicated plots is comparatively slow. This can be circumvented, in part, by data reduction and defining macros for processing and plotting actions, which limit the number of rendering steps, but this does require a slightly different way of working compared to a GUI-based step-by-step approach.

Note, though, that programming and maintaining a similar package in a compiled language, with issues of cross-platform compatibility and ever-changing operating systems, libraries and compilers, is much more demanding. In our experience, the functionality of Matlab and the ease of programming by far outweigh a possible loss of speed compared to well-optimized compiled code, especially since NMR data are relatively small, the ''mathematics'' of processing is not complicated and CPU speeds and RAM memory sizes have increased dramatically in the last decade.

4. Conclusions

A flexible toolbox for data processing, analysis and visualization in Matlab is presented. A large variety of NMR data formats and standard 1D and 2D processing techniques are supported. matNMR combines a user-friendly graphical interface, or script-based processing if wanted, with the computational and graphical capabilities that Matlab has to offer. The matrix-based Matlab software is ideally suited for a hands-on approach to processing, subsequent numerical analysis and creating high-quality plots. The user is free to determine the desired degree of interaction with the matNMR GUI, Matlab and user-defined routines. From user experiences it seems that this approach suits both Matlab and NMR novices, as well as experts.

Finally, to highlight the flexibility of the Matlab environment, we present the application of a mobile spectrometer: external instruments can be controlled and interfaced from the Matlab workspace, using additional toolboxes. A complete Matlab-based spectrometer that controls both pulse generation and acquisition was constructed from only a computer, an appropriate rf generating board and an ampiflier. The data processing was afforded by the matNMR toolbox [32].

Acknowledgments

Many people have contributed to the development of matNMR. Dr. Aswin Verhoeven, Prof. Arno Kentgens and Dr. Bernd Fitzinger are specifically acknowledged for the many suggestions they made. Dr. Giorgia Zandomeneghi, Theofanis Manolikas and Dr. René Verel are acknowledged for the spectra of Fig. 1. Prof. Beat Meier is thanked for supporting the development of matNMR.

References

[1] For a comprehensive list see www.spectroscopynow.com.
[2] www.varianinc.com.
[3] www.bruker-biospin.de.
[4] www.jeol.com.
[5] www.tecmag.com.
[6] www.mestrec.com.
[7] www.acornnmr.com.
[8] www.umatek.com.
[9] www.npnmr.com.
[10] www.accelrys.com.
[11] www.inmr.net.
[12] www.perchsolutions.com.
[13] www.nmrtec.com.
[14] www.acdlabs.com.
[15] J.C. Hoch, A.S. Stern, NMR data processing, Wiley, London, 1996, www.rowland.org/rnmrtk.

[16] U. Günther, C. Ludwig, H. Rüterjans, NMRlab—advanced NMR data processing in Matlab, J. Magn. Res. 145 (2000) 201–208.
[17] www.grandinetti.org.
[18] J.-L. Pons, T. Malliavin, M. Delsuc, Gifa v. 4: a complete package for NMR data set processing, J. Biomol. NMR 8 (1996) 445–452.
[19] www.nmrtools.com.
[20] F. Delaglio, S. Grezesiek, G. Vuister, G. Zhu, J. Pfeifer, A. Bax, NMRpipe: a multidimensional spectral processing system based on unix pipes, J. Biomol. NMR 6 (1995) 277–293.
[21] www.umanitoba.ca/chemistry/nmr.
[22] www.abcis.cbs.cnrs.fr/NPK.
[23] Mathworks Inc., www.mathworks.com.
[24] M. Bak, J. Rasmussen, N.C. Nielsen, Simpson: a general simulation program for solid-state NMR spectroscopy, J. Magn. Res. 147 (2000) 296–330.
[25] N. Bloembergen, J. Rowland, On the nuclear magnetic resonance in metals and alloys, Acta Met. 1 (1953) 731.
[26] www.octave.org.
[27] www.scilab.org.
[28] S. Smith, T. Levante, B.H. Meier, R.R. Ernst, Computer simulations in magnetic resonance: an object oriented programming approach, J. Magn. Res. A 106 (1994) 75–105.
[29] M. Veshtort, R. Griffin, Spinevolution: a powerful tool for the simulation of solid and liquid state NMR experiments, J. Magn. Res. 178 (2006) 248–282.
[30] P. Hodgkinson, "pNMRsim: a general simulation program for large problems in solid-state NMR", URL: http://www.dur.ac.uk/paul.hodgkinson/pNMRsim.
[31] W.B. Blanton, Blochlib: a fast NMR C++ tool kit, J. Magn. Res. 162 (2003) 269–283.
[32] A. Kentgens, Radboud universiteit Nijmegen, private communication.
[33] R. Verel, T. Manolikas, A.B. Siemer, B.H. Meier, Improved resolution in 13C solid-state spectra through spin-state-selection, J. Magn. Res. 184 (2007) 322–329.