Numerical Simulation of Magnetic Resonance Experiments: Concepts and Applications to Static, Rotating and Double Rotating Experiments *

M. Baldus, T. O. Levante, and B. H. Meier
Laboratorium für Physikalische Chemie, ETH-Zentrum, 8092 Zürich, Switzerland

Z. Naturforsch. **49a,** 80–88 (1994); received August 31, 1993

An object-oriented programming environment for numerical simulation of magnetic resonance spectra is introduced and applied to NQR and NMR of quadrupolar nuclei. Using a Floquet approach it is possible to perform simulations of spin systems that are described by explicitly time-dependent Hamiltonians in full analogy to simulations of time-independent systems. Applications to magic angle spinning and double rotation are discussed.

Much of the success of magnetic resonance methods in numerous applications can be attributed to the flexibility of the experiment. In many cases it can be tailored to address a specific property of the sample to be characterized. Often it is beneficial or indispensable to simulate numerically the response of the spin system to pulse sequences and sample rotation. Examples where numerical simulations are useful include the prediction of spectra, the evaluation of the effect of non-idealities on the spectra, optimization of pulse sequences and the evaluation of the basic spin system parameters (e.g. chemical shielding, dipolar and scalsar coupling constants, quadrupolar coupling constant and asymmetry) from experimental spectra (e.g. by least squares analysis). A general magnetic resonance simulation package must preserve this flexibility and allow to simulate all practically feasible experiments, as well as many more that are more difficult to realize on a spectrometer. This requirement is not easy to fulfill in a single simulation program, although flexible approaches have been described [1, 2]. Therefore, most simulations are presently performed using task-specifically written simulation programs. The usual computer languages do, however, not contain the basic data types that are used to theoretically describe magnetic resonance experiments. One has to resort to explicit matrix element manipulations while the underlying theory is formulated much more elegantly, e.g. by using quantum-mechanical operators and superoperators, spin and spacial tensors. The object-oriented programming tool kit GAMMA (General Approach to Magnetic resonance Mathematical Analysis) [3], that we discuss in the following, is a novel approach to magnetic resonance simulations. An existing computer programming language, C++ [4], is extended by defining new data types such as operators, superoperators, Floquet-operators and tensors that are used in the theoretical description of magnetic resonance experiments [5]. This approach enables the spectroscopist to formulate a problem to be solved numerically by a computer in a language that mimics the abstract formalism used for the analytical description. In addition to magnetic resonance specific data types predefined in GAMMA the entire power of the underlying programming language C++ is at the spectroscopist disposition to realize a specific simulation. GAMMA is not a task-oriented simulation program but an open-ended platform that allows the simulation of the spin dynamics (using a density operator formalism) in an arbitrary magnetic resonance experiment.

In the following, selected features of GAMMA will be exemplified by applications to NQR and to solid state NMR spectroscopy of quadrupolar nuclei. Furthermore, a numerical Floquet approach for the simulation of spectra under time-dependent Hamiltonians, as they appear, for example, in magic angle spinning (MAS) and double rotation (DOR) experiments, will be discussed.

As a simple introductory example that illustrates the GAMMA approach to magnetic resonance simulations, the free induction decay of a quadrupolar nu-

---

* Presented at the XII International Symposium in Nuclear Quadrupole Resonance, Zürich, July 19–23, 1993, Switzerland.

Reprint requests to Dr. B. H. Meier, Laboratorium für Physikalische Chemie, ETH Zentrum, Universitätsstr. 22, 8092 Zürich, Switzerland.

0932-0784 / 94 / 0100-0080 $ 01.30/0. – Please order a reprint rather than making your own copy.

Analytical	**Computer Code**
1. Declare Spin System	
One spin system: Q	spin_sys Q(1);
considered isotope: ^{23}Na	Q.isotope (0, "23 Na");
2. Declare Symbols	
quadrupolar coupling constant: $\frac{\omega_Q}{2\pi}$	float nuq = 617000;
asymmetry: η	float eta = 0;
space Tensor: A	space_T A = A2 (0, nuq, eta);
quadrupolar spin Tensor: T	spin_T T = T_Q (Q, 0);
density operator: $\hat{\sigma}$	gen_op s;
Observable (complex number): $\langle \hat{I}^+ \rangle$	complex a;
time interval for evolution: τ	float t = 1e-5;
3. Define the Hamiltonian	
$H = \sum_{m=-2}^{2} (-1)^m A_{2-m} T_{2m}$	H = T_prod (T, A, 2);
4. Define Initial Density Matrix	
$\hat{\sigma}(0) = I_y$	$s = Iy(Q, 0)$;
5. Let the Density Operator evolve under the Hamiltonian	
$\hat{\sigma}(\tau) = e^{-2\pi i H \tau} \sigma(0) e^{2\pi i H \tau}$	s = exp (−2*P*I***I**H*t)*s*exp (2*P*I***I**H*t)
6. Calculate the Expectation Value for the (complex) Magnetization	
$\langle \hat{I}^+ \rangle = Tr\{\hat{I}^+ \cdot \hat{\sigma}\}$	a = trace ($I_p(Q)$*s);

Fig. 1. Comparison between the analytical description and the GAMMA code required to calculate the expectation value of the magnetization under the NQR Hamiltonian of (1)

cleus in zero magnetic field shall be calculated. In the laboratory frame, the Hamiltonian described in terms of irreducible tensor operators, is given by

$$H_Q = \sum_{m=-2}^{2} (-1)^m A_{2-m}^{(Q)} T_{2m}^{(Q)}, \tag{1}$$

where the elements of the spin tensor operator are defined as:

$$\begin{aligned} T_{20}^{(Q)} &= \frac{1}{\sqrt{6}} [3 I_z^2 - I(I+1)], \\ T_{2\pm 1}^{(Q)} &= \mp \frac{1}{2} [I^\pm I_z + I_z I^\pm], \\ T_{2\pm 2}^{(Q)} &= \frac{1}{2} \cdot (I^\pm)^2, \end{aligned} \tag{2}$$

and the elements of spatial tensor are given by

$$\begin{aligned} A_{20}^{(Q)} &= \sqrt{\frac{3}{8}} \frac{\omega_Q}{2\pi} [(3\cos^2\beta - 1) + \eta \sin^2\beta \cos 2\alpha], \\ A_{2\pm 1}^{(Q)} &= \frac{1}{2} \frac{\omega_Q}{2\pi} \sin\beta \, e^{\mp i\gamma} [\mp (3 - \eta \cos 2\alpha) \\ &\quad \cdot \cos\beta - i\eta \sin 2\alpha], \\ A_{2\pm 2}^{(Q)} &= \frac{1}{2} \frac{\omega_Q}{2\pi} e^{\mp 2i\gamma} \left(\frac{3}{2} \sin^2\beta + \frac{\eta}{2} (1 + \cos^2\beta) \right. \\ &\quad \left. \cdot \cos 2\alpha \pm i\eta \cos\beta \sin 2\alpha \right). \end{aligned} \tag{3}$$

Note that all Hamiltonians are given in frequency units. The Euler angles α, β, γ describe the orientation of the principal axes system of the quadrupolar interaction with respect to the laboratory frame, η denotes the asymmetry and the quadrupolar coupling constant is defined as

$$\omega_Q = \frac{e^2 q Q}{2I(2I-1)\hbar}. \tag{4}$$

The time evolution of the expectation value of the transverse magnetization (FID) under the Hamiltonian of (1) can be evaluated analytically [6]. Here, however, we describe a numerical evaluation as an example for the use of the GAMMA environment. The GAMMA code required to calculate the expectation value $\langle I^+ \rangle$ is compared with the analytical description in Fig. 1 where, for simplicity, a special crystal orientation, $\alpha = \beta = \gamma = 0$ was used. As a consequence of the introduction of magnetic resonance specific data types, a one to one correspondence between analytical formulas and computer code is achieved. In paragraphs 1 and 2 of Fig. 1 the spin system and the necessary operators and parameters are declared and initialized. In addition to the standard C++ data type *float*, the GAMMA-defined data types *spin_sys* (representing a spin system), *space_T* (representing a spatial tensor operator), *spin_T* (representing a spin tensor operator), *gen_op* (a general quantum mechanical operator) and *complex* (a complex number) have been used. The spatial tensor operator A is initialized using the GAMMA function $A2$ that generates the spatial tensor operator of (3). The spin tensor, whose elements are given by (2), is initialized by the GAMMA function $T\_Q(Q,0)$ where the parameter Q (of type *spin_sys*) denotes the spin system (specified in paragraph 1 of Fig. 1). The second parameter to T_Q is the number of the spin (in Q) for which the tensor operator is required. In the present one-spin example this value is zero, denoting the zeroth spin (note that in the C++ language, numbering starts at 0, while for the analytical description one usually starts at 1). In paragraph 3 of Fig. 1 the Hamiltonian is constructed as the tensor product of the spin and spatial tensor operators (see (5)). In GAMMA the tensor product is denoted by *T_prod*. The third parameter to *T_prod*, which is set to 2 (see Fig. 1), indicates that only rank 2 tensors are of importance and that it is not necessary to treat the other ranks. In paragraph 4, the initial density operator is set equal to I_y. The time-evolution of this density operator is obtained the usual way, as shown in paragraph 5 of Figure 1 and the expectation value for the polarization is calculated in paragraph 6. It should be noted that the computer code on the right hand side of Fig. 1, enclosed by the standard C++ "main() {"and"}" statements, is the complete simulation program.

The data types and functions on them used in the program of Fig. 1 are examples for the features of the so called mathematical shell of GAMMA. A more complete description is given elsewhere [3].

Technically, the magnetic resonance data types have been implemented using the *class* concept of C++ (also used in other object oriented programming languages), that allows to extend the built-in abstract data types (for example *float*, *integer*, *character*) by user-defined (abstract) data types. An abstract data type is defined by a set of objects (take quantum-mechanical operators as an example) as well as the functions defined on them (for operators, functions would include addition, multiplication, trace operations etc.). For an introduction to the concept of abstract data types, we refer to the literature [4, 7–11]. In the present context, the most important feature of an abstract data type is, that it handles objects as complete entities. No knowledge of the internal data representation is needed. For an operator (class *gen_op*) this means, that two operators A and B can be multiplied or added using a single statement of the form $A+B$ and without accessing single elements of a matrix representation. This feature made it possible to formulate the simulation of Fig. 1 without ever using an explicit matrix representation of the operators involved.

It is convenient to use, in addition to the mathematical shell, a spectroscopic shell that contains the higher level concepts used by the spectroscopist. These include ideal, real or shaped pulses, the acquisition of an FID or the explicit definition of a relaxation superoperator.This shell is constructed around the mathematical shell. A practical example of its use will be presented below. An important feature of C++, that allows for a flexible and user-friendly spectroscopic language, is the concept of function overloading. The function that calculates and FID can be called *FID* irrespectively whether the calculation is done in Hilbert space, Liouville space or in Floquet space. Depending on the data type of the detection operator and the Hamiltonian used, the correct code for *FID* is chosen internally.

An even higher level of abstraction could be realized in a spectrometer operation shell that uses the mathematical and spectroscopic shell to mimic a magnetic resonance spectrometer. Spin system, spectrometer characteristics and pulse program are specified and a spectrum is obtained from the computer. In this mode, GAMMA could be used to debug pulse programs and to save valuable instrument time. Clearly, the implementation of the spectrometer operation shell depends on the pulse programmer language of the target spectrometer. This shell is not implemented in GAMMA presently.

The shell structure of GAMMA is summarized in Figure 2. In very much the same manner as the programming language shell (the compiler) encloses the processor-dependent machine language, the two GAMMA shells and the spectrometer operation shell enclose the compiler shell. Advantages and disadvantages of using the GAMMA shells instead of the compiler shell are the same as the advantages and disadvantages of using the compiler shell instead of the machine language. The most important advantages are, that programming becomes easier, more compact and less error-prone by using the higher shell. A disadvantage lies in the unavoidable computational overhead that is added by the higher shell and that makes the execution of the program slower. This drawback can, at least partially, be compensated by optimization techniques that are automatically applied by the higher shell and which might be time consuming to

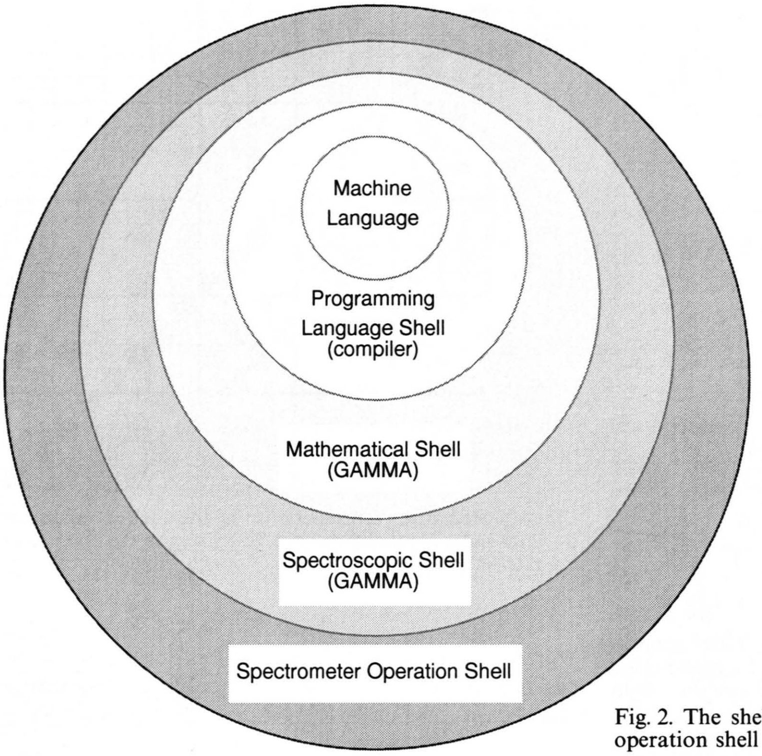

Fig. 2. The shell structure of GAMMA. The spectrometer operation shell is presently not implemented.

exploit in directly programming the lower level shell. A number of schemes for automatic optimization are implemented in GAMMA, in particular for the data type *matrix* that is used by the data type *gen_op* and is involved in many of the time-consuming steps of a simulation [3].

Using the programming schemes sketched above, we proceed to calculate the spectra of a quadrupolar nuclei in the presence of a magnetic field. The calculation is most easily performed in the laboratory frame of reference where the Hamiltonian is given by $H = H_Q + H_0$ with the Zeeman interaction $H_0 = \omega_0 I_z$ and the quadrupolar interaction H_Q defined in (1). ω_0 denotes the Larmor frequency of the nucleus. In high magnetic fields it is permissible to go to an interaction frame representation with respect to the dominant Zeeman interaction and to neglect the nonsecular contributions. The average Hamiltonian to first order is then given by [12]

$$\bar{H}_Q = A^{(Q)}_{20} T^{(Q)}_{20} + \frac{1}{2\omega_0} \sum_{m=\pm 1, \pm 2} \frac{1}{m} A^{(Q)}_{2m} A^{(Q)}_{2-m} [T^{(Q)}_{2m}, T^{(Q)}_{2,-m}]. \tag{5}$$

The central transition of a spin 3/2 in a powdered sample for different values of the asymmetry parameter η has been calculated using the GAMMA program given in Appendix A. For the calculation of the central transition, only the second term in (5) needs to be considered. The resulting spectra are given in Figure 3. The calculations have been performed using the spectroscopic shell of GAMMA. To illustrate its properties, a few statements from the program, listed in full in Appendix A, shall be analyzed. The application of an (ideal) y-pulse on the density operator is done by *sigma = lypuls* (*Q*, *sigma*, *90.*); where *Q* denotes the spin system (data type *spin_sys*), sigma the density operator (data type *gen_op*) and "90." the pulse angle (datatype *float*). The FID is collected using *FID* (*sigma, detect, H, dwtime, NP, spect*); where the detection operator (data type *gen_op*), the Hamiltonian (data type *gen_op*), the dwell time (data type *float*), the number of points (data type *integer*) and an array to store the data, spect (data type *data_1D*) are amongst the parameters of the GAMMA-function *FID*.

So far, time-independent Hamiltonians have been used. There are, however, a number of situations where the Hamiltonian is explicitly time-dependent

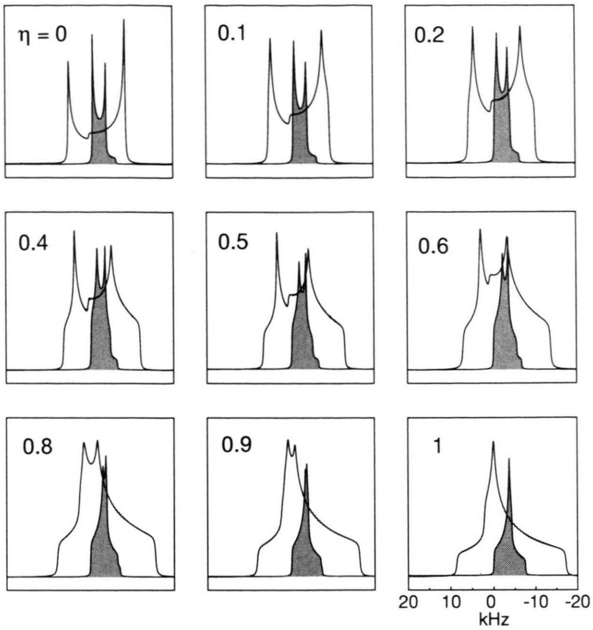

Fig. 3. Comparison between static and MAS (shaded spectra) powder lineshapes of the central transition of a spin 3/2 for different asymmetry parameters η. For the calculation in GAMMA the first order average Hamiltonian of (5) was used. A Larmor frequency of 132.29 MHz and a quadrupolar coupling constant of 617 kHz were assumed. The MAS spinning frequency was set to 20 kHz.

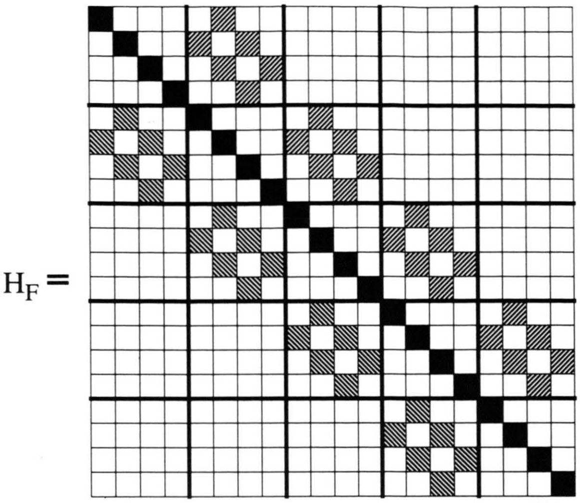

Fig. 4. Schematical representation of the Floquet operator for an rf pulse applied for a spin 3/2 system in the absence of an external magnetic field for a Floquet dimension of 2.

including sample rotation, double rotation or time-dependent radiofrequency irradiation. In NMR, rf pulses can usually be described by a time-independent Hamiltonian by going to a rotating frame representation. For pulsed NQR spectroscopy however, this is not possible because the rotating as well as the counter-rotating component of a linearly polarized rf field are resonant with transitions of the spin system [6, 13]. Therefore, the time-dependence of the Hamiltonian must be retained. Often, the Hamiltonian can, to excellent approximation, be split-up into non-interacting virtual spin 1/2 contributions that may be evaluated separately in appropriate interaction frames. Here, this property shall, however, not been exploited, but the entire time-dependence shall be retained in a simulation that illustrates the use of time-dependent operators. The solution of the equation of the motion for a time-dependent Hamiltonian [5]

$$\sigma(t) = T \exp\left(-i \int_0^t H(t')\,\mathrm{d}t'\right) \cdot \sigma(0)\, T' \exp\left(i \int_0^t H(t')\,\mathrm{d}t'\right), \qquad (6)$$

where T denotes the Dyson time-ordering operator, can numerically be evaluated either by approximating the continuously time-dependent Hamiltonian by a piecewise time-constant Hamiltonian, or by transforming into a Floquet space [14, 15], where the Hamiltonian is represented by a time-independent but infinite-dimensional matrix

$$H(t) \rightarrow H_F, \qquad \sigma(t) \rightarrow \sigma_F(t). \qquad (7)$$

In practice, the matrix dimension can be truncated to a finite value for the numerical evaluation. Explicit matrix representations for Floquet Hamiltonians have been given by Vega [15] and a formal introduction of a Liouville space will be presented elsewhere. Here, we demonstrate the embedding of a Floquet-operator data type, *floq_op*, into the GAMMA framework. The application of an rf-pulse in pure NQR spectroscopy shall be used as an example. A schematical matrix representation of a Floquet operator that corresponds to the Hilbert space operator

$$H = H_Q + 2\gamma B_1 \cos(\omega t) I_x \qquad (8)$$

describing a NQR pulse for a spin 3/2 is shown in Fig. 4, where filled squares correspond to nonzero matrix elements. In GAMMA, a Floquet operator that is characterized by the (truncated) dimension N of the Floquet space, the Hilbert space dimension hs and the basic Fourier frequency omega, is initialized by *floq_op HF* (*N, hs, omega*); and the Floquet Hamiltonian is constructed by inserting the non-zero operators $H^{(n)}$ (Hn), that are defined by the Fourier expansion of the Hamiltonian according to

$$H(t) = \sum_{n=-\infty}^{\infty} H^{(n)} \exp(i n \omega t), \tag{9}$$

into the n'th side-diagonal using the GAMMA function *HF.put_sdiag* (*Hn, n*). Similarly, the initial Floquet density operator sigmaf (data type *floq_op*) is defined and constructed from the (Hilbert space) operator sigma (data type *gen_op*) via: *sigmaf.put_block* (*sigma, 0, 0*). With these definitions, the time-evolution of the (Floquet) density operator and the detection of the magnetization can be performed in complete analogy to Hilbert space calculations because the GAMMA functions (e.g. *evolve* and *FID*) automatically recognize that the Hamiltonian and density operator are defined in Floquet space and perform the appropriate Floquet space operations. The pure NQR spectrum, using quadrature detection, of a spin 3/2 after a pulse of length t_p is shown in Fig. 5 as a function of t_p. The spectrum consists of two lines with opposite amplitude at $\pm 3\omega_Q$. The maximum signal amplitude ("90° pulse") for a rf-field strength of ω_1 is found for $t_p = \dfrac{\pi}{2\sqrt{3}\,\omega_1}$, as expected [6]. The opposite sign of the amplitude of the two resonance lines is understood by the well known fact that the NQR FID is sine-modulated and linearly polarized along the direction of the irradiating field with $\langle I_y \rangle = \langle I_z \rangle = 0$ [6].

Sample spinning and double rotation are important application where time-dependent Hamiltonians invariably occur. The time-dependence of the first order average Hamiltonian is contained in the spatial tensor $A^{(Q)}$ of (5). For the MAS case, we obtain

$$A^{(Q)}_{2m}(t) = \sum_{m'=\pm1\pm2} A^{(Q)}_{2m'}\, d_{mm'}(\theta) \exp(-i m' \omega_r t). \tag{10}$$

Because the product of two spatial tensor components appear in the second term of (5), the time dependence of the quadrupolar interaction (to first order average Hamiltonian theory) is described by Fourier components at $k\omega_r$ with $k = \pm 4 \pm 3 \pm 2 \pm 1$ and 0. A schematic representation of the matrix representation of the Floquet Hamiltonian for a spin 3/2 is given in Figure 6. The MAS lineshapes for the central transition and a spinning speed of 20 kHz are shown as shaded spectra in Fig. 3, where Larmor frequency and quadrupolar constant were set to 132.29 MHz and 617 kHz, respectively. Because the spinning frequency is much larger than the quadrupole interaction (for the central transition), the spectra of Fig. 3 agree with the simulations by Amoureux et al. [16] for the case of infinitely fast spinning. The minor differences in the two sets of spectra come from a different assumption for the linewidth. If the spinning speed is of the same order of magnitude as the quadrupolar interaction, the lineshape becomes more complicated. An example

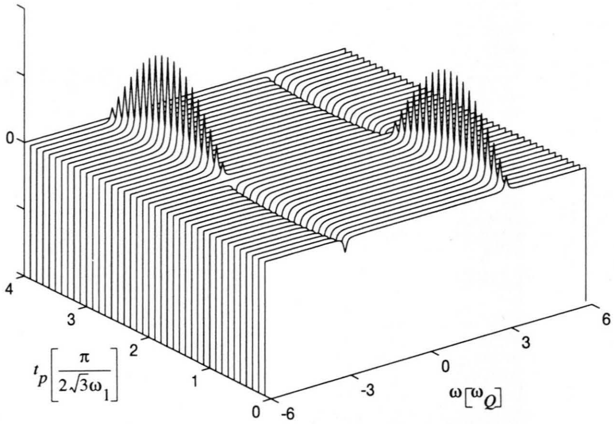

Fig. 5. Pure pulsed NQR spectrum of a spin 3/2 as a function of the duration t_p of a linearly polarized rf pulse.

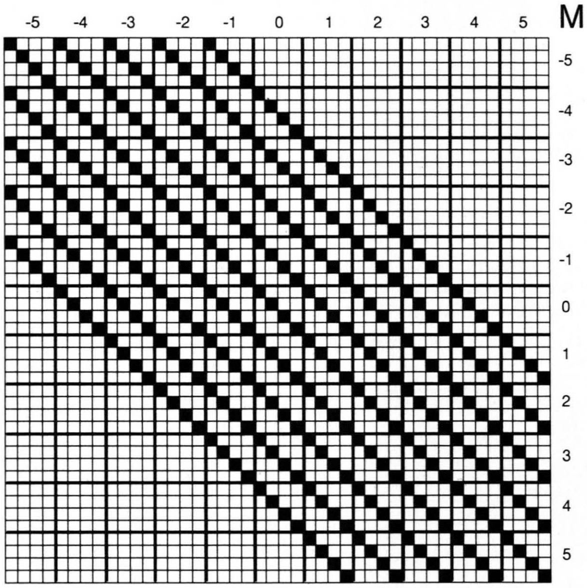

Fig. 6. Schematic matrix representation of the Floquet Hamiltonian described by the secular terms of (5) for MAS using a Floquet space with dimension M = 5.

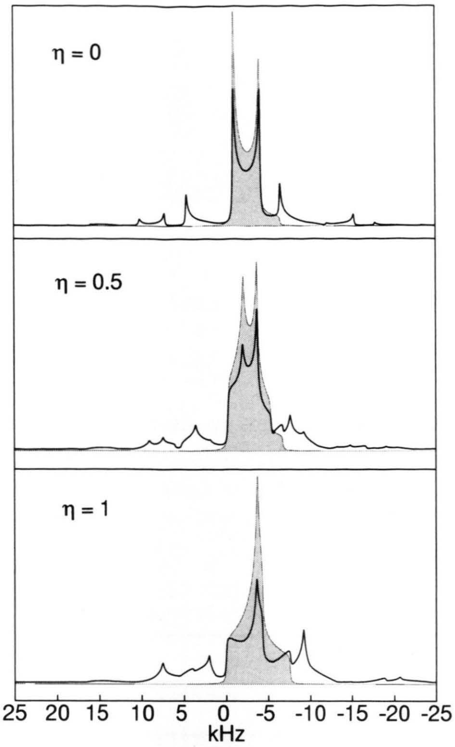

Fig. 7. Comparison between the MAS powder lineshapes for the central transition of a spin 3/2 at a spinning speed of 5.5 kHz and 20 kHz (shaded spectra) for three different asymmetry parameters. Quadrupole coupling constant and Larmor frequency are given in Fig. 3.

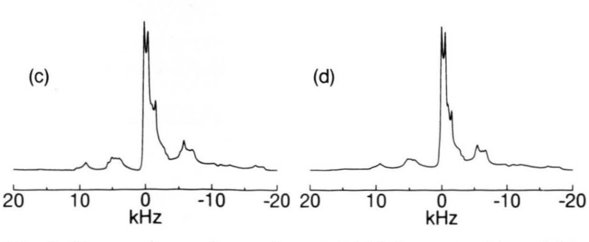

Fig. 8. Comparison of experimental MAS spectra (a) and (c) and simulated spectra (b) and (d) of anhydrous Na_2HPO_4 for spinning frequencies of: (a), (b): 7200 Hz and (c), (d): 5230 Hz. For the simulation, three sites with quadrupolar frequencies of 229, 355, and 617 kHz and asymmetries of 0.21, 0.69, 0.217 were used.

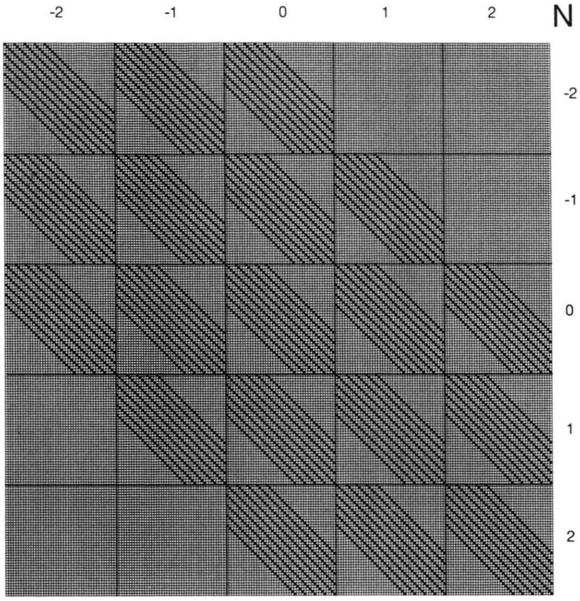

Fig. 9. Schematic matrix representation of the (2 mode) Floquet quadrupolar Hamiltonian under DOR where Floquet dimensions of N = 2 and M = 5 for the ω_1 and ω_2 dimensions, respectively, were used.

is given in Fig. 7 where the spectra for MAS at 5.5 kHz are compared to the MAS spectra in the fast spinning limit.

Experimental MAS lineshapes of anhydrous Na_2HPO_4 have been reproduced by a GAMMA simulation using the superposition of three magnetically inequivalent ^{23}Na sites for two different spinning speeds. A comparison between experimental and simulated spectra is made in Figure 8.

DOR experiments can be described in terms of two, in general non-commensurable, spinning frequencies ω_1 and ω_2. The Floquet space concept can be generalized [17] to describe more than one time-dependence. In GAMMA, a data type *flog2_op* is defined which allows the construction of 2 mode Floquet operators in analogy to the single mode treatment discussed above. The appropriate Fourier components Hkl at $k\omega_1 + l\omega_2$ that again represent matrices of the Hilbert space dimension, are inserted at the appropriate side diagonal (k, l) of the Floquet Hamiltonian using the GAMMA function *HF.put_sdiags* (Hkl, k, l). As in the MAS treatment, nonzero Fourier components appear at $k, l = \pm 4 \pm 3 \pm 2 \pm 1$ and 0. The (truncated) matrix representation of the (2 mode) Floquet Hamiltonian is shown in Figure 9.

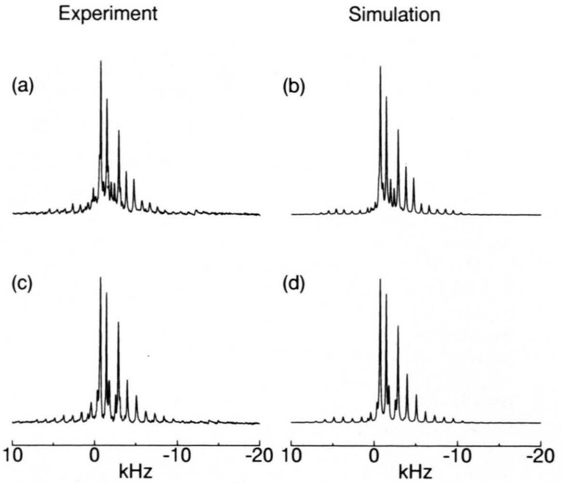

Fig. 10. Comparison between measured (a) and (c) and calculated (b) and (d) unsynchronized DOR spectra on anhydrous Na_2HPO_4 for spinning speeds of (a), (b): [5470 Hz, 925 Hz] and (c) and (d): [5350 Hz, 1100 Hz]. The spin system parameters are given in the captions to Fig. 8.

The function overloading concept allows to perform simulations of DOR spectra using essentially the same GAMMA code as for the MAS simulation. A comparison between simulated and recorded unsynchronized DOR spectra of Na_2HPO_4 is presented in Fig. 10 for two different sets of spinning speeds and using the same parameters as for the MAS simulation of Figure 8. The powder spectra of Fig. 10 have been obtained by averaging over about 1000 crystallite orientations in less than 7 hours of computer time on a SUN SparcStation 2.

In summary, a novel concept for the simulation of a wide range of magnetic resonance experiments was applied. The spin dynamics under time-dependent Hamiltonians was evaluated using a Floquet approach that allows one to describe time-dependent and time-independent experiments using the same formalism.

Distribution

The GAMMA environment, as described in [3], will be made available to the public without charge under the conditions of the GNU General Public Licence [18]. Users of GAMMA are requested to cite [3] if they use the GAMMA library for publications. The Floquet data types are still under development and will not be included in the first GAMMA release.

Acknowledgements

Helpful discussions with Professor R. R. Ernst are gratefully acknowledged. This work has been supported in part by the Swiss National Science Foundation and the Stipendien-Fond der Basler Chemischen Industrie. We also thank Dr. S. Steuernagel for measuring the unsynchronized DOR spectra at Bruker Analytische Meßtechnik, Rheinstetten/Germany.

[1] W. Studer, J. Magn. Reson. **7,** 424 (1988).
[2] F. Stickney de Bouregas and J. S. Waugh, J. Magn. Reson. **96,** 280 (1992).
[3] S. Smith, T. Levante, B. H. Meier, and R. R. Ernst, J. Magn. Reson. A (in press).
[4] B. Stroustrup, The C_{++} programming language. Second ed. edition, Addison-Wesley, Reading, Massachusetts 1991.
[5] R. R. Ernst, G. Bodenhausen, and A. Wokaun, Principles of Nuclear Magnetic Resonance in One and Two Dimensions, Clarendon press, Oxford 1987.
[6] A. Abragam, The Principles of Nuclear Magnetism, Clarendon press, Oxford 1961.
[7] C. Ghezzi and M. Jazayeri, Programming language concepts. Wiley, New York, 2 edition, 1987.
[8] P. Thomas, H. Rodinson, and J. Emms, Abstract Data Types. Oxford Applied Mathematics and Computing Science Series. Clarendon Press, Oxford 1988.
[9] S. C. Dewhurst and K. T. Stark, Programming in C_{++}. Prentice Hall, Englewood Cliffs, New Jersey 1989.
[10] M. A. Ellis and B. Stoustrup, The annotated C_{++} reference manual. Addison-Wesley, Reading, Massachusetts 1991.
[11] P. Coad and E. Yourdon, Object-oriented design. Yourdon Press, Prentice Hall, Englewood Cliffs, New Jersey 1991.
[12] B. Q. Sun, J. H. Baltisberger, Y. Wu, A. Samoson, and A. Pines, Solid State Nuclear Magnetic Resonance **1,** 267 (1992).
[13] M. S. Krishnan, F. P. Temme, and B. C. Sanctuary, Mol. Phys. **78,** 1385 (1993).
[14] J. H. Shirley, Phys. Rev. **138 B,** 979 (1965).
[15] S. Vega, E. T. Olejniczak, and R. G. Griffin, J. Chem. Phys. **80,** 4832 (1984).
[16] J. P. Amoureux, C. Fernandez, and P. Granger. Interpretation of quadrupolar powder spectra: Static and MAS experiments. In: P. Granger and R. K. Harris (eds.), Multinuclear Magnetic Resonance in Liquids and Solids – Chemical Applications, pp. 409–424, Kluwer Academic Publishers, 1990.
[17] S. Chu, Adv. Chem. Phys. **73,** 739 (1989).
[18] The GNU General Public License is available from the Free Software Foundation, Cambridge, Massachusetts 02139, USA, and from the ftp-server prep.ai.mit.edu.

Appendix A

```
/* GAMMA source code for the simulation of a quadrupolar powder
   pattern using the secular first order average Hamiltonian */

# include "gamma.h"

 main ( )
{
   spin_system Q (1);                                        // Set up a spin system
   Q.isotope (0, "23Na");                                    // nucleus: 23Na
   float nuq = 3.702e6;                                      // quadrupolar coupling constant
                                                             // in frequency units
   float eta = 0;                                            // asymmetry parameter
   double dwtime = 5E-5;                                     // dwell time
   int NP = 4096;                                            // number of points in FID
   block_1D spect (NP);                                      // single crystal FID
   block_1D specsum (NP);                                    // powder sum
   double phi, theta;                                        // Euler angles for powder loop

   String outFileName = "powderspec";                        // output filename

   space_T A_pas = A2 (0, nuq, ta);                          // space tensor in principle axis sys.
   space_T A;                                                // rotated tensor
   spin_T T=T_Q (Q, 0);                                      // quadrupolar spin tensor
   gen_op H;                                                 // Hamiltonian

   gen_op sigma;                                             // density operator
   sigma = sigma_eq (Q);                                     // thermal equilibrium
   sigma = Iypuls (Q, sigma, 90.);                           // 90° pulse

   matrix cen_Trans (4, 4, 0);                               // built detection operator
   cen_Trans.put (0.5, 1, 2);                                // for the
   gen_op detect (cen_Trans);                                // central transition 1/2 → -1/2

   spin_op S1_1, S2_2;                                       // spin operators needed below

   S1_1 = TTQ.component (2, 1)*TTQ.component (2, -1) - TTQ.component (2, -1)*TTQ.component (2, 1);
   S2_2 = TTQ.component (2, 2)*TTQ.component (2, -2) - TTQ.component (2, -2)*TTQ.component (2, 2);

for (int b = 1; b < 28657; b++)                              // powder average
  {
     theta = 180./28675.*b;                                  // theta in [0, 180]
     phi = double (360./28675) * ((10946*b)%28675));         // phi   in [0, 360]

     A = T_rot (A_pas, phi, theta, 0.);                      // rotate spatial part

     H   = A.component (2, 1)*A.component (2, -1)*S2_1;      // built up Hamiltonian
     H += A.component (2, 2)*A.component (2, -2)*S2_2*0.5;

     FID (sigma, detect, H, dwtime, NP, spect);              // aquisition
     exponential_multiply (spect, -10);                      // apodization
     spect = FFT (spect);                                    // Fourier Transform
     spect* = sin (theta*PI/180);
     specsum + = spect;                                      // powder summation
  }
 MATLAB (outFileName, outFileName, specsum, 1);              // output on Matlab
                                                             // Matlab is commercial product
                                                             // from: The MathWorks Inc.

}
```