Computer Simulations in Solid-State NMR. III. Powder Averaging

MATTIAS EDÉN

*Physical Chemistry Division, Arrhenius Laboratory, Stockholm University, SE-*106 91 *Stockholm, Sweden*

ABSTRACT: As the final part in a series of articles on numerical simulations in solid-state NMR (Concepts Magn Reson 17A: 117–154, 2003 and Concepts Magn Reson Part A 18A: 1–23, 2003), aspects of simulations of NMR responses from powders are discussed. The underlying equations for powder averaging are derived, and it is demonstrated how powder averages may be estimated numerically. Orientational symmetry in solid-state NMR is summarized and exploited to achieve more efficient calculations. Explicit computer code in C/C++ is given for simulation of NMR spectra from powders containing (1) two homonuclear spins-1/2 in a static sample and (2) a heteronuclear two spin-1/2 system under magic-angle-spinning conditions. © 2003 Wiley Periodicals, Inc. Concepts Magn Reson Part A 18A: 24–55, 2003

KEY WORDS: solid-state NMR; numerical simulation; computer algorithm; powder average; orientational average; orientational symmetry; static solid; magic-angle-spinning; dynamically inhomogeneous Hamiltonian

INTRODUCTION

NMR is applied extensively for obtaining structural and dynamical information in a wide range of solid systems. Internuclear distances and molecular geometries may be probed directly through the measurement of anisotropic NMR interactions in solid samples. For example, the determination of internuclear distances may be obtained from dipolar couplings and the estimate of interbond and dihedral angles may be made by combining information of orientations of the chemical shift, dipolar or quadrupolar tensors (*1–5*).

Single crystals usually provide the most accurate structural information. However, many compounds do not allow for growing large crystals suitable for analysis. This applies, for instance, to many biomolecular systems. In general, a more feasible alternative to using single crystals is performing NMR experiments on powders. The spectral resolution is degraded significantly, but many experimental techniques are available to circumvent this problem, including magic-angle-spinning (MAS) and multiple-pulse techniques (*1–7*).

This article deals with the problem of numerically simulating solid-state NMR spectra from powders under different experimental conditions. It constitutes the third and final part of a series of publications: The two preceding articles, Ref. (*8*) and Ref. (*9*), will be referred to as "Part I" and "Part II," respectively. Part

Received 5 October 2001; revised 22 October 2002; accepted 20 November 2002

Correspondence to: M. Edén; E-mail: mattias@physc.su.se.

Concepts in Magnetic Resonance Part A, Vol. 18A(1) 24–55 (2003)

Published online in Wiley InterScience (www.interscience.wiley.com). DOI 10.1002/cmr.a.10065

I outlined theoretical formalisms for calculating NMR time-domain signals and frequency-domain spectra for so-called dynamically inhomogeneous problems (*10*). Part II described how these may be implemented numerically in computer programs, relevant for simulating the NMR response from a single crystal. However, the NMR time-domain signal and frequency-domain spectrum depend on the orientation of the molecule with respect to the static magnetic field direction. When simulating spectra of *powders* comprising a large number of microscopically small crystals, it is necessary to do a procedure called *powder averaging*, involving an average over the signals from all crystallites. In this article, we incorporate the results of Parts I and II to address the problem of calculating spectra from nuclear spins evolving under dynamically inhomogeneous Hamiltonians in powders.

This article is organized as follows: First, the reasons for the orientational dependence of the NMR signal is explained and the basic equations for performing powder averages are derived. Next, orientational symmetry in solid-state NMR is explained as well as how powder averaging may be implemented numerically. Finally, we address two representative problems in solid-state NMR of powders: Calculation of the NMR spectrum of (1) a homonuclear system of two coupled spins-1/2 in a static sample and (2) a heteronuclear system of two coupled spins-1/2 under MAS conditions. The latter case is generic for simulations of separated local field (SLF) experiments (*2–4*, *11–13*). A general introduction is given to each case and explicit C/C++ programs are listed in Appendices B and C. The complete source code used in this article is available from Ref. (*14*).

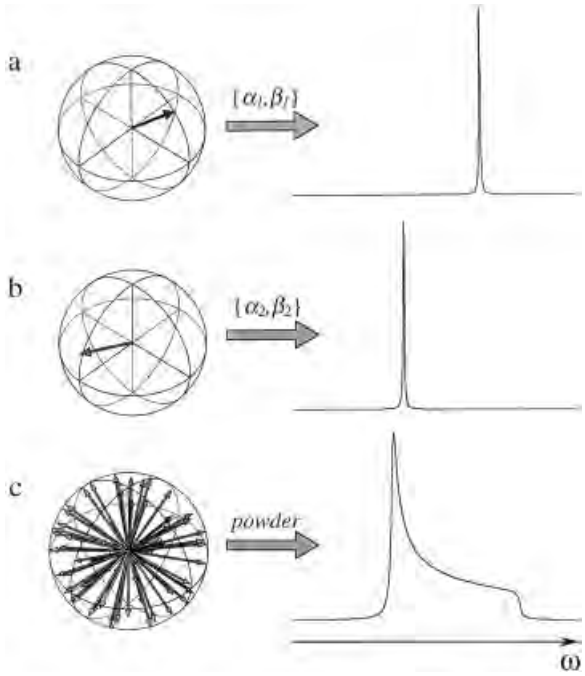

Figure 1 The spectum generated from an NMR interaction depends on the orientation of its spatial tensor with respect to the static magnetic field direction. Here, we assumed for simplicity that the orientation may be associated with only two Euler angles $\{\alpha, \beta\}$ so that it it can be represented as a point on the unit sphere (see Fig. I-4). (a) and (b) show two such orientations and their corresponding NMR spectra, which are *different* from each other. (c) The spectrum from a static powder corresponds to the sum of spectra from a large number of randomly oriented crystallites, each representing a certain tensor orientation. A broad spectrum, spread over a large frequency range, is obtained.

PRINCIPLES OF POWDER AVERAGING

Orientational Dependence of the NMR Signal

The NMR frequency ω_Λ for an anisotropic NMR interaction Λ depends on the orientation of its spatial second-rank tensor with respect to the direction of the static magnetic field $\vec{\mathbf{B}}_0$, as depicted in Fig. 1. The orientation is denoted $\Omega = \{\alpha, \beta, \gamma\}$, and is, depending on the experimental situation, parameterized by various Euler angles, such as Ω_{PL} or Ω_{PR}. In a solid sample, the orientation of each spatial tensor of the spins in a rigid molecular fragment may be associated with the orientation of the crystal with respect to the magnetic field. The spatial tensor orientation changes when the orientation of the crystal changes. Therefore, we will use "crystal" (or "molecular") orientation synonymously with "tensor" orientation.

This means that the NMR time-domain signal obtained from an interaction also depends on orientation. The orientational dependence propagates from the spatial tensor over to the time-domain signal as follows:

$$[A_{20}(t;\,\Omega)]^L \;\rightarrow\; \hat{\mathbf{H}}(t;\,\Omega) \;\rightarrow\; \hat{\mathbf{U}}(t,\,t_0;\,\Omega) \;\rightarrow\; \hat{\rho}(t;\,\Omega) \;\rightarrow\; s(t;\,\Omega) \quad [1]$$

as can be shown by combining Eqs. [I-60], [II-2], [II-3], and [II-4]. In contrast to the mathematical formalism used in Parts I and II, henceforth, the dependence of NMR quantities on molecular orientation will be indicated *explicitly*.

Powder Averages

Assume that the NMR sample consists of a powder comprising a very large number N_{cryst} of small crystallites. A finely ground powder for NMR studies typically comprises 10^7–10^8 randomly oriented crystallites. Each of these has a certain orientation Ω with respect to the static magnetic field direction. In this ensemble, we may find n_j crystallites with orientation Ω_j. Assuming that N_{cryst} is infinite, there is an infinite number of such "subsets" Ω_j, each of which produces an NMR signal $s(t; \Omega_j)$. Due to the large number of different crystallite orientations, each giving a characteristic and orientation-dependent signal, the NMR spectrum from such powders are typically spread over a large range of frequencies, as illustrated in Fig. 1c.

In this section, we derive an expression for the *powder-averaged time-domain signal*, which is the average contribution from all subsets Ω_j. It is denoted

$$\bar{s}(t) \equiv \langle s(t; \Omega) \rangle_{\Omega} \qquad [2]$$

where $\langle \cdots \rangle_{\Omega}$ represents an average over all orientations and is calculated as follows:

$$\bar{s}(t) = N_{\text{cryst}}^{-1} \sum_j n_j s(t; \Omega_j) \qquad [3]$$

$$= \sum_j \mathcal{P}(\Omega_j) s(t; \Omega_j) \qquad [4]$$

Here, $\mathcal{P}(\Omega_j) = n_j / N_{\text{cryst}}$ is the probability of finding a crystallite with orientation Ω_j in the powder. In other words, the powder-averaged time-signal corresponds to the sum of the signals deriving from each orientation, weighted by the probability of finding that particular orientation.

Every statistical distribution may be described in terms of a *probability density function* (*15*), denoted $p(\Omega) \equiv p(\alpha, \beta, \gamma)$. It is defined such that the probability of finding an orientation Ω within the interval of orientations $\Omega_a \leq \Omega \leq \Omega_b$ corresponds to

$$\mathcal{P}(\Omega_a \leq \Omega \leq \Omega_b) = \int_{\Omega_a}^{\Omega_b} d\Omega \; p(\Omega) \qquad [5]$$

$\mathcal{P}(\Omega_a \leq \Omega \leq \Omega_b)$ may be interpreted as the area enclosed by the density function over the range of orientations. The probability of finding an orientation within the full range of Euler angles must be unity, i.e., the probability density function is normalized such that the integral of $p(\Omega)$ over the full range of orientations, denoted V, is unity

$$\int_V d\Omega \; p(\Omega) = 1 \qquad [6]$$

with $V \equiv \{0 \leq \alpha < 2\pi, \; 0 \leq \beta \leq \pi, \; 0 \leq \gamma < 2\pi\}$.

Assume that the crystals are randomly oriented. Provided that N_{cryst} is large, the powder is referred to as being *isotropic*, and all orientations Ω_j are represented in the full range of Euler angles. Now consider an orientation Ω_j and an infinitely small change $d\Omega$ around that value, corresponding to a simultaneous change in each of the Euler angles α, β, and γ, i.e., $d\Omega = (d\alpha, d\beta, d\gamma)$. $\mathcal{P}(\Omega_j)$ denotes the probability of finding an orientation within $[\Omega_j, \Omega_j + d\Omega]$, i.e., the probability that simultaneously α is in the range $[\alpha_j, \alpha_j + d\alpha]$, β is in the range $[\beta_j, \beta_j + d\beta]$, and γ is in the range $[\gamma_j, \gamma_j + d\gamma]$. Equation [5] may be reformulated as

$$\mathcal{P}(\Omega_j) = p(\Omega_j) d\Omega \qquad [7]$$

which means that the probability of finding a crystallite with orientation Ω_j is given by the product of the probability density (evaluated at Ω_j) and the differential $d\Omega$. Combination of Eqs. [4] and [7] gives

$$\bar{s}(t) = \sum_j p(\Omega_j) s(t; \Omega_j) d\Omega \qquad [8]$$

For infinitesimally small changes $d\Omega_j$, the summation over all orientations Ω_j may be replaced by an *integration* over the full range of Euler angles. This results in the fundamental equation for the powder-averaged time signal:

$$\bar{s}(t) = \int_V d\Omega \; p(\Omega) s(t; \Omega) \qquad [9]$$

$$= \int_0^{2\pi} d\alpha \int_0^{\pi} d\beta \int_0^{2\pi} d\gamma \; p(\alpha, \beta, \gamma) s(t; \alpha, \beta, \gamma) \qquad [10]$$

The probability density depends on the orientational order in the sample, and the appropriate expression for $p(\alpha, \beta, \gamma)$ is needed when evaluating Eq. [10]. A general property of a probability density func-

tion depending on several variables, $p(x, y)$, is that if the distributions over the individual variables x and y are statistically independent of each other ("uncorrelated") one may factorize $p(x, y)$ according to $p(x, y) = p(x)p(y)$ (*15*). Consequently, the two probability densities $p(x)$ and $p(y)$ may be handled separately. This applies to the probability density function $p(\alpha, \beta, \gamma)$ of an *isotropic powder*, and we may therefore split the probability density into a product of three separate functions, each depending solely of one of the variables α, β, and γ:

$$p(\alpha, \beta, \gamma) = p(\alpha)p(\beta)p(\gamma) \qquad [11]$$

In the following, we will simplify the expression for the powder-averaged time-domain signal (Eq. [10]) by separately evaluating the expressions for each of the individual probability densities $p(\alpha)$, $p(\beta)$, and $p(\gamma)$.

Because the dependence of $p(\gamma)$ may be handled separately, we can exploit the geometric picture of Fig. I-4, namely, that a molecular orientation $\{\alpha, \beta\}$ may be represented as a point on the surface of the unit sphere (i.e., a sphere with radius $r = 1$).

For uniform distributions of crystallites in the sample, the probability density of the statistical variable is given by the inverse of its range of definition (*15*). This may directly be applied to $p(\alpha)$ and $p(\gamma)$:

$$p(\alpha) = p(\gamma) = (2\pi)^{-1} \qquad [12]$$

Finding the expression for $p(\beta)$ requires some care because the geometric interpretation of $\{\alpha, \beta\}$ as a point on the unit sphere involves a nonlinear mapping of the variable β from one-dimensional region $0 \leq \beta \leq \pi$ onto the spherical surface. This implies that $p(\beta)$ is not a constant factor as in the cases $p(\alpha)$ and $p(\gamma)$. Below, we derive an expression for $p(\beta)$ using geometric arguments.

The probability of finding an orientation in the range $\{[\alpha, \alpha + d\alpha], [\beta, \beta + d\beta]\}$ is given by the product $p(\alpha)p(\beta)d\alpha d\beta$. This corresponds to the area confined by the shaded region $d\mathcal{A}$ on the sphere shown in Fig. 2(a), divided by the total spherical surface area. $d\mathcal{A}$ is usually referred to as the "solid angle" (*4, 16–18*) of the orientation $\{\alpha, \beta\}$. For infinitesimally small changes $d\alpha$ and $d\beta$, the solid angle may be approximated as a rectangle of sides l_α and l_β. Its area is then given by

$$d\mathcal{A} = l_\alpha l_\beta \qquad [13]$$

Each of the sides of the rectangle makes the third side of two isosceles triangles, as illustrated in Fig. 2(b). l_α and l_β may be calculated from the "law of cosines" (*15*). This states that if AB, AC, and BC are the lengths of each side of a triangle, then BC may be calculated from AB, AC, and the angle θ between the vectors $\overrightarrow{\mathbf{AB}}$ and $\overrightarrow{\mathbf{AC}}$, according to

$$(BC)^2 = (AB)^2 + (AC)^2 - 2AB \cdot AC \cdot \cos\theta \qquad [14]$$

From Fig. 2(b) then it follows that l_α is obtained from the two known distances $\sin\beta$ and the angle $d\alpha$:

$$l_\alpha^2 = 2\sin^2\beta(1 - \cos d\alpha) \qquad [15]$$

Using the trigonometric identity $1 - \cos x = 2\sin^2(x/2)$ and solving for l_α gives

$$l_\alpha = 2\sin\beta\sin\frac{d\alpha}{2} \qquad [16]$$

Because the function $\lim_{x\to 0}\sin x = x$ (*15*), we accordingly approximate $\sin(d\alpha/2)$ as $(d\alpha/2)$. Then, the following expression is obtained for the distance l_α:

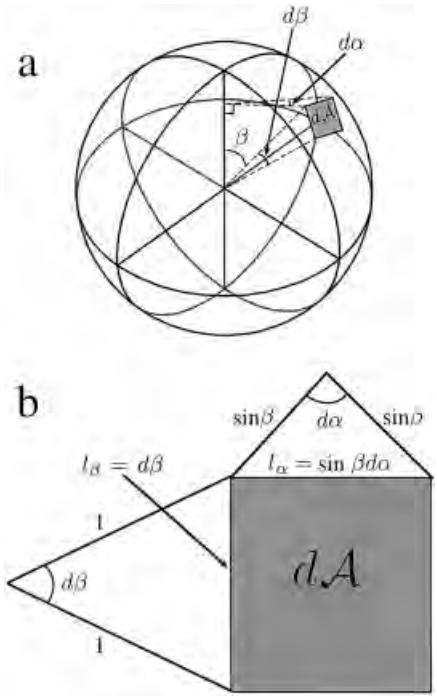

Figure 2 (a) Area $d\mathcal{A}$ on the unit sphere resulting from small changes in the angles α and β around the point parameterized by the orientation $\{\alpha, \beta\}$. (b) $d\mathcal{A}$ may be approximated as a rectangle with sides of length l_α and l_β. Each distance l_α and l_β is the third side of an isoceles triangle. Note that the figure is not drawn to scale: $d\alpha$ and $d\beta$ are *much smaller* than the radius of the sphere.

$$l_\alpha = \sin\beta d\alpha \qquad [17]$$

Identical arguments may be applied for obtaining the following expression for the distance l_β:

$$l_\beta = d\beta \qquad [18]$$

Finally, inserting the expressions for l_α and l_β into Eq. [13] gives the following expression for the solid angle of the orientation $\{\alpha, \beta\}$:

$$d\mathcal{A} = \sin\beta d\alpha d\beta \qquad [19]$$

The probability $\mathcal{P}(\alpha, \beta)$ of finding the orientation $\{\alpha, \beta\}$ is finally calculated as the ratio between its solid angle and the surface area of the unit sphere (4π):

$$\mathcal{P}(\alpha, \beta) = \frac{1}{4\pi} d\mathcal{A} = \frac{1}{4\pi} \sin\beta d\alpha d\beta \qquad [20]$$

This shows that the probability of finding an orientation within the range $\{[\alpha, \alpha + d\alpha], [\beta, \beta + d\beta]\}$ scales as $\sin\beta$. Consequently, a larger *fraction* of the crystallites in the sample are oriented near the equator of the sphere ($\beta = \pi/2$) than at the poles ($\beta = 0$ and $\beta = \pi$). This means that there is a larger contribution to the powder-averaged signal from orientations with $\beta \approx \pi/2$ than with $\beta \approx 0$. This is also realized by inspecting Fig. 2: For constant $d\alpha$ and $d\beta$, the solid angle is larger near the equator than at the poles. After substituting the expression for $p(\alpha)$ into Eq. [20], we get

$$p(\beta) = \frac{1}{2} \sin\beta \qquad [21]$$

Finally, substituting Eqs. [12] and [21] into Eq. [10] gives the following expression for the powder-averaged time signal from samples with randomly oriented crystallites

$$\bar{s}(t) = \frac{1}{8\pi^2} \int_0^{2\pi} d\alpha \int_0^{\pi} d\beta \sin\beta \int_0^{2\pi} d\gamma\, s(t; \alpha, \beta, \gamma) \qquad [22]$$

Note that for partially oriented samples different expressions are obtained for the probability densities $p(\alpha)$, $p(\gamma)$, and $p(\beta)$ than those given by Eqs. [12] and [21]. Further, the integral in Eq. [22] may often be simplified: As discussed in the next section, in many cases the time signal is only dependent on one or two orientational variables, and additionally, the integration *ranges* may be restricted further.

In the "frequency-domain" simulations discussed in Part II, the NMR *spectrum* is calculated *directly*, implying that the spectral frequencies and amplitudes are the relevant functions for powder averaging:

$$\bar{S}(\omega) = \frac{1}{8\pi^2} \int_0^{2\pi} d\alpha \int_0^{\pi} d\beta \sin\beta \int_0^{2\pi} d\gamma S(\omega(\Omega)) \qquad [23]$$

Orientational Variables

Here, we specify with which particular set of Euler angles (Ω_{PL}, Ω_{ML}, Ω_{PR}, or Ω_{MR}) that Ω should be identified for different spin systems and experimental conditions. In the most general cases, the NMR signal depends on all three angles of the triplet $\Omega = \{\alpha, \beta, \gamma\}$.

For a *single interaction* in *static* powders, the angle Ω_{PL}^{Λ} is random, and we should identify $\Omega \equiv \Omega_{PL}^{\Lambda}$ in Eq. [22]. An example of such a case is a powdered sample in which each molecule contain a single (isolated) nuclear site experiencing a chemical shift anisotropy (CSA) interaction. If the initial frame is chosen to coincide with the PAS of the CSA tensor, the laboratory-frame frequency $\omega_\Lambda = [A_{20}^{\Lambda}]^L$ is given by Eq. [I-94]. However, this expression is *independent* of the angle γ_{PL}. From Eq. [1] it follows that neither the Hamiltonian nor the propagator depends on this orientational angle, and consequently neither the time-domain signal. This implies that a calculation of the NMR signal (using Eq. [22]) with a static powder needs only an average over the two angles α_{PL} and β_{PL} of the full triplet Ω_{PL}^{Λ}. However, note that this scenario is only valid within the high-field approximation (*1–5*) discussed in Part I; at zero magnetic field B_0, the $m \neq 0$ components of the Hamiltonian must be taken into account (i.e., Eq. [I-59] must be used) and the time signal will then be dependent on all three orientational variables. Here, we disregard such exotic cases.

In the case of *several interactions* in *static* powders, it is desirable to exploit a molecular frame into which the principal axis system (PAS) of each interaction is initially transformed. For example, this situation applies to a powder containing *spin pairs*. For such a system, one typically needs to include the CSA tensor of each site, as well as the through-space dipolar interaction. The PAS of these interactions do in general not coincide, and it is desirable to first transform each interaction into a common coordinate sys-

tem, i.e., a molecular frame. Consequently, the angle Ω_{ML} is the random orientational angle in a powder and, therefore, the target for powder averaging. Also in this case, averaging over only two Euler angles is required because Eq. [I-97] does not depend on γ_{ML}.

In *rotating* powders, using reference frames as in Eq. [I-100], the relevant set of angles for averaging is Ω_{MR}, as then the final transformation between the rotor and laboratory frame is identical for all crystallites in the powder. As may be seen from Eq. [I-105], averaging needs to be performed over the set of all three Euler angles $\{\alpha_{MR}, \beta_{MR}, \gamma_{MR}\}$; however, the next section will provide us with a convenient way of doing this. Three-angle averaging also applies to the case of a *single interaction* involving an *asymmetric* ($\eta \neq 0$) tensor if the transformation is performed from its PAS directly to the rotor frame. In this case the appropriate set of angles is Ω_{PR}.

Powder averaging of the NMR signal from a *single symmetric* tensor ($\eta = 0$) represents a particularly simple case. In a *static* solid, the signal only depends on β_{PL}. This is because the *only nonzero* component of the tensor in its PAS is A_{20} (Eq. [I-88]) and the expression for the tensor component in the laboratory frame (Eq. [I-94]) simplifies according to

$$[A^{\Lambda}_{20}]^L = \sum_{m=-2}^{2} [A^{\Lambda}_{2m}]^P d^2_{m0}(\beta^{\Lambda}_{PL}) \exp\{-im\alpha^{\Lambda}_{PL}\} \quad [24]$$

$$= [A^{\Lambda}_{20}]^P d^2_{00}(\beta_{PL}) \quad [25]$$

The same argument applied to Eq. [I-102] shows that in a *rotating* solid the signal only depends on the *two* angles β_{PR} and γ_{PR}, as is realized by modifying the sequence of transformations given in Eq. [I-100], and instead transform the PAS of the tensor *directly* into the rotor frame:

$$[\overleftrightarrow{\mathbf{A}}_{\Lambda}{}^{(2)}]^L = [\overleftrightarrow{\mathbf{A}}_{\Lambda}{}^{(2)}]^P \hat{\mathbf{D}}^{(2)}(\Omega_{PR}) \hat{\mathbf{D}}^{(2)}(\Omega_{RL}(t)) \quad [26]$$

The laboratory frame $m = 0$ component for a symmetric tensor is then

$$[A^{\Lambda}_{20}]^L = \sum_{m=-2}^{2} [A^{\Lambda}_{20}]^P D^2_{0m}(\Omega^{\Lambda}_{PR})\, D^2_{m0}(\Omega_{RL}(t)) \quad [27]$$

Using the definition of the time-dependent Euler transformation angles $\Omega_{RL}(t) = \{-\omega_r t, \theta_m, 0\}$ (Eq. [I-99]) and applying Eq. [I-49] twice gives

$$[A^{\Lambda}_{20}]^L = \sum_{m=-2}^{2} [A^{\Lambda}_{20}]^P d^2_{0m}(\beta_{PR}) \exp\{-im\gamma_{PR}\} d^2_{m0}(\theta_m) \exp\{im\omega_r t\} \quad [28]$$

It follows that the expression for $[A^{\Lambda}_{20}]^L$ depends on β_{PR} but is *independent* of α_{PR} as in the case of the static powder. However, unlike the scenario with a static sample, Eq. [28] also carries a dependence on γ_{PR} due to the additional transformation to the rotor frame.

Above, the orientational dependence of the NMR signal was motivated by inspecting the equations relating the spatial tensor from its PAS to the laboratory frame. Here, an alternative picture is given: The static magnetic field may be represented as a vector and its direction can therefore be specified by two Euler angles. The NMR signal depends only on the relative orientation between the static field and the interaction tensor. Instead of transforming the PAS of the tensor into the laboratory frame (parameterized by Ω_{PL}), one may equally well *transform* the *field vector* into the PAS of the interaction; the latter transformation can be achieved using *only two* Euler angles, as the orientation of a vector may, in *any* coordinate system, be specified by two Euler angles (see Fig. I-4). Therefore, for static solids the powder averaging is only required over two orientational variables.

So, why then are all three angles $\{\alpha, \beta, \gamma\}$ in general needed when calculating powder averages of rotating samples? The reason is that a direct transformation between the molecular and laboratory frames is *time dependent.* Powder averaging over time-dependent angles would be very inconvenient, if at all possible. The additional transformation employed from the molecular frame to the rotor frame solves this problem by *separating* the *time* dependence from the *orientational* dependence. The price paid, however, is a lower symmetry of the problem compared to the static case. This is because parameterizing the transformation from an arbitrarily chosen molecular frame to the rotor frame requires in general all angles of the Euler triplet.

Nevertheless, as demonstrated in the next section, in many cases in rotating solids the powder averaging may in practice be reduced to integration over only two Euler angles due to the presence of an additional symmetry. This symmetry is absent for a singe crystal but is present in isotropic powders. Moreover, the integration *ranges* may often be reduced because the NMR signal possesses certain symmetry properties with respect to the orientational variables.

ORIENTATIONAL SYMMETRY IN SOLID-STATE NMR

Carousel Symmetry in Rotating Solids

In the following, we consider an isotropic powder undergoing MAS with a dynamically inhomogeneous Hamiltonian, assuming that several anisotropic interactions are involved and that all tensors are expressed in a molecular frame, such that the powder averaging involves an integration over the orientational variable $\Omega_{MR} = \{\alpha_{MR}, \beta_{MR}, \gamma_{MR}\}$. Throughout this section, we will carefully indicate the orientational dependence of all entities, but for brevity we identify $\{\alpha, \beta, \gamma\} \equiv \{\alpha_{MR}, \beta_{MR}, \gamma_{MR}\}$. *Hence,* Ω *here denotes the angles* Ω_{MR} *relating the molecular and rotor frames.*

For powder averaging in rotating solids, the Euler angle γ plays a special role and provides a useful property for the spin Hamiltonian. For isotropic powders, this opens up a route to handling the average over γ through a closed analytic expression. This holds in general for simulation of any spin system subjected to sample spinning, as long as no RF fields are applied and the initial density operator is *not* dependent on the orientation of the molecule at the start of the experiment (*19–24*). This applies to all dynamically inhomogeneous cases treated in Parts I and II. As far as powder averaging is concerned, simulations of MAS and static experiments may then be treated on equal footing; "*explicit*" averaging is only required over two Euler angle components.

The expression for the spatial tensor component $\omega_\Lambda(t; \Omega) = [A^\Lambda_{20}(t; \Omega)]^L$ is given by Eq. [I-102]. This frequency enters the spin dynamics when the Hamiltonian for the spin interaction is constructed. To separate out the dependencies of ω_Λ on the various orientational variables α, β, and γ, we insert Eq. [I-49] into Eq. [I-102]. This gives

$$\omega_\Lambda(t; \Omega) = \sum_{m',m=-2}^{2} [A^\Lambda_{2m'}]^M d^2_{m'm}(\beta) d^2_{m0}(\theta_m) \times \exp\{-im'\alpha\} \exp\left\{im\omega_r\left(t - \frac{\gamma}{\omega_r}\right)\right\} \quad [29]$$

The last exponential factor may be interpreted as follows: The angle γ is equivalent to a shift in *time* by γ/ω_R. Note that the angles γ and $\omega_r t$ correspond to rotations around the *same* axis, namely, the *z*-axis of the rotor frame.

In an isotropic powder, all values of Ω are represented, so that given an arbitrary orientation $\Omega^a = \{\alpha, \beta, \gamma\}$ we may also find the orientation $\Omega^b = \{\alpha, \beta, 0\}$. If these two Euler triplets are substituted into Eq. [29] one obtains the following relationship between the interaction frequency ω^a_Λ of the crystallite orientation Ω^a and that of Ω^b:

$$\omega^a_\Lambda(t; \alpha, \beta, \gamma) = \omega^b_\Lambda(t - \gamma/\omega_r; \alpha, \beta, 0) \quad [30]$$

Hence, the spatial tensors from two crystallite orientations differing *only* in the value of γ give the *same NMR frequencies as the sample rotates but at different points in time,* as is illustrated in Fig. 3. This relationship between the time dependence of the tensor and its orientation γ is referred to as "*carousel symmetry*" (*24*). The terminology stems from the similarity to the situation of a carousel in an amusement park: Two persons riding on the carousel follow the same circular path, and see the same views, but at shifted time points. We stress that carousel symmetry only applies to powders having crystallites that are isotropically distributed with respect to γ, i.e., *all* values $0 \le \gamma < 2\pi$ are represented for *each* pair $\{\alpha, \beta\}$ in the powder.

For later calculations, it is useful to separate out the dependence on the angle γ in the expression for the Fourier components of the interaction frequency. We rewrite Eq. [29] as follows:

$$\omega_\Lambda(t; \Omega) = \sum_{m=-2}^{2} \omega^{(m)}_\Lambda(t; \Omega) \exp\{im\omega_r t\} \quad [31]$$

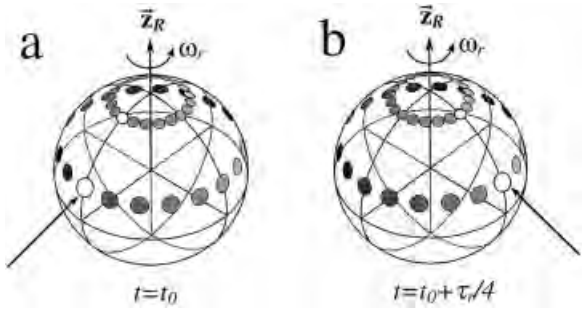

Figure 3 Visualization of "carousel symmetry" in a powder under sample rotation around the rotor frame axis $\vec{z}_R$. Each "tensor orientation" is depicted as a circle on the spherical surface. Two carousels are shown, each comprising orientations with *fixed* $\{\alpha, \beta\}$ values. Note that in this figure a rotation around the $\vec{z}_R$-axis corresponds to a *change* in the Euler angle γ: Different members of the carousel *only* differ in their values of γ. (a) shows the positions of the carousel members at $t = t_0$ and (b) the same carousels at $t = t_0 + \tau_r/4$, i.e., after the sample has rotated by $\pi/2$, corresponding to a quarter of a rotational period. The arrows indicate the *same* carousel member (labeled by a "white circle") at the two different time points.

with the Fourier components obtained from Eq. [29] according to

$$\omega_\Lambda^{(m)}(t;\,\Omega) = \sum_{m'=-2}^{2} [A_{2m'}^{\Lambda}]^M d_{m'm}^2(\beta) d_{m0}^2(\theta_m) \exp\{-im'\alpha\} \exp\{-im\gamma\} \quad [32]$$

As the dependence on γ only appears in the rightmost factor, we have the following relationship between the Fourier components $\omega_\Lambda^{(m)}$ from different carousel members (i.e., orientations only differing in the value of γ):

$$\omega_\Lambda^{(m)}(t;\,\alpha,\,\beta,\,\gamma) = \omega_\Lambda^{(m)}(t;\,\alpha,\,\beta,\,0)\exp\{-im\gamma\} \quad [33]$$

Consequences of Carousel Symmetry. We continue to examine the consequences of the carousel symmetry in isotropic powders undergoing MAS. The generic expression for the NMR time-domain signal was shown in Part I to be

$$s(t) = \sum_{u,v=1}^{\mathcal{N}} \sum_{k=-\infty}^{\infty} a_{uv}^{(k)} \exp\{i\omega_{uv}^{(k)} t\} \quad [34]$$

with the spinning sideband amplitudes

$$a_{uv}^{(k)} = c_{uv}^{(k)} \langle u|\hat{\rho}(0)|v\rangle\langle v|\hat{\mathbf{Q}}|u\rangle \quad [35]$$

given by a product involving matrix elements of the initial density operator and observable as well as $c_{uv}^{(k)}$. The latter is defined as a coefficient in the Fourier expansion of the periodic phase $\Phi'_{uv}(t,\,0;\,\Omega)$:

$$\exp\{i\Phi'_{uv}(t,\,0;\,\Omega)\} = \sum_{k=-\infty}^{\infty} c_{uv}^{(k)}(\Omega)\exp\{ik\omega_r t\} \quad [36]$$

Now, consider the orientational dependence of the various components of Eq. [34]: If the NMR signal is generated directly by applying an RF pulse to the spin ensemble at thermal equilibrium, $\hat{\rho}(0)$ is *independent* on the molecular orientation. The same applies to the observable operator $\hat{\mathbf{Q}}$. The spectral frequencies, $\omega_{uv}^{(k)} = \omega_{uv}^{(0)} + k\omega_r$, depend according to Eq. [II-12] on the difference between the $m = 0$ Fourier components of the Hamiltonian eigenvalues $\omega_v^{(0)}$ and $\omega_u^{(0)}$. However, we defined these to comprise the *isotropic* parts of the NMR interactions, which in turn are *independent of molecular orientation*, as discussed in Part I. Next, consider the coefficient $c_{uv}^{(k)}$, which depends on the dynamic phase $\Phi'_{uv}(t,\,0)$. This, on the other hand, involves the Hamiltonian eigenvalue Fourier components $\omega_v^{(m)}$, which according to Eq. [II-14] comprise the *anisotropic* (and, hence, orientation-*dependent*) parts of the spin interactions,

$$\Phi'_{uv}(t,\,t_0;\,\Omega) = (i\omega_r)^{-1} \sum_{m\neq 0} m^{-1}\omega_{uv}^{(m)}(\Omega)\Big(\exp\{im\omega_r t\} - \exp\{im\omega_r t_0\}\Big) \quad [37]$$

with the difference in Fourier coefficients defined by

$$\omega_{uv}^{(m)}(\Omega) = \omega_v^{(m)}(\Omega) - \omega_u^{(m)}(\Omega) \quad [38]$$

As only the factor $c_{uv}^{(k)}$ is dependent on the molecular orientation in Eq. [35], we focus in the following on examining its expression further. To this end, we define an auxiliary function $\xi_{uv}(t;\,\Omega)$ according to (*25*)

$$\xi_{uv}(t;\,\Omega) = (i\omega_r)^{-1} \sum_{m\neq 0} m^{-1}\omega_{uv}^{(m)}(\Omega)\exp\{im\omega_r t\} \quad [39]$$

Note the similarity between the function $\xi_{uv}(t;\,\Omega)$, dependent on *one* time index, and the dynamic phase $\Phi'_{uv}(t,\,t_0;\,\Omega)$, which depends on *two* time indices. Using the definition of ξ_{uv}, we may (for $t_0 = 0$) express $\Phi'_{uv}(t,\,0;\,\Omega)$ as follows:

$$\Phi'_{uv}(t,\,0;\,\Omega) = \xi_{uv}(t;\,\Omega) - \xi_{uv}(0;\,\Omega) \quad [40]$$

This is easily verified by comparing Eqs. [37] and [39]. Moreover, it follows by combining Eqs. [39] and [I-106] that ξ_{uv} is a real function.

Analogously to the proof that the dynamic phase Φ'_{uv} is time periodic (see page 148 of Part I), one may show that $\xi_{uv}(t;\,\Omega)$ possesses the same periodicity. Hence, it may be expanded in a Fourier series (analogously to Eq. [36]):

$$\exp\{i\xi_{uv}(t;\,\Omega)\} = \sum_{k=-\infty}^{\infty} C_{uv}^{(k)}(\Omega)\exp\{ik\omega_r t\} \quad [41]$$

The Fourier coefficients $C_{uv}^{(k)}$ in Eq. [41] are related to the coefficients $c_{uv}^{(k)}$ in Eq. [36]. This is not surprising in view of Eq. [40]. We examine this relationship by first exponentiating Eq. [40] (*25*):

$$\exp\{i\Phi'_{uv}(t, 0; \Omega)\} = \exp\{i\xi_{uv}(t; \Omega) - i\xi_{uv}(0; \Omega)\} \quad [42]$$

$$= \exp\{i\xi_{uv}(t; \Omega)\}\exp\{-i\xi_{uv}(0; \Omega)\} \quad [43]$$

Since ξ_{uv} (0; Ω) is a real number, $\exp\{-i\xi_{uv}(0; \Omega)\} = \exp\{i\xi_{uv}(0; \Omega)\}^*$, and we may write Eq. [43] as

$$\exp\{i\Phi'_{uv}(t, 0; \Omega)\} = \exp\{i\xi_{uv}(t; \Omega)\}\exp\{i\xi_{uv}(0; \Omega)\}^* \quad [44]$$

Next, we insert the Fourier expansion (Eq. [41]) for each of the exponential factors to obtain

$$\exp\{i\Phi'_{uv}(t, 0; \Omega)\} = \underbrace{\left(\sum_{k=-\infty}^{\infty} C_{uv}^{(k)}(\Omega)\exp\{ik\omega_r t\}\right)}_{\exp\{i\xi_{uv}(t;\Omega)\}} \times \underbrace{\left(\sum_{k'=-\infty}^{\infty} C_{uv}^{(k')}(\Omega)\exp\{ik'\omega_r \cdot 0\}\right)^*}_{\exp\{i\xi_{uv}(0;\Omega)\}^*} \quad [45]$$

$$= \sum_{k=-\infty}^{\infty}\sum_{k'=-\infty}^{\infty} C_{uv}^{(k)}(\Omega)C_{uv}^{(k')}(\Omega)^*\exp\{ik\omega_r t\} \quad [46]$$

Note the time index $t = 0$ in the rightmost factor of Eq. [45], making all exponential functions vanish in the Fourier expansion. Equations [36] and [46] are two forms of the Fourier series of the factor $\exp\{i\Phi'_{uv}(t, 0)\}$; however, because a Fourier series is unique, we may equate the coefficients of the two series. Then, we get the following relationship between the components $c_{uv}^{(k)}$ and $C_{uv}^{(k)}$ (*24–26*):

$$c_{uv}^{(k)}(\Omega) = \sum_{k'=-\infty}^{\infty} C_{uv}^{(k)}(\Omega)C_{uv}^{(k')}(\Omega)^* \quad [47]$$

Next, we derive a useful relationship between the Fourier coefficients $C_{uv}^{(k)}(\Omega)$ from *different carousel members,* i.e., for crystallite orientations *only differing* in the Euler angle γ. From the carousel symmetry (Eq. [30]) follows the relationship

$$\xi_{uv}(t; \alpha, \beta, \gamma) = \xi_{uv}(t - \gamma/\omega_r; \alpha, \beta, 0) \quad [48]$$

This may be verified by evaluating $\xi_{uv}(t - \gamma/\omega_r; \alpha, \beta, 0)$ (the right side of Eq. [48]) using Eq. [39]:

$$\xi_{uv}(t - \gamma/\omega_r; \alpha, \beta, 0) = (i\omega_r)^{-1}\sum_{m\neq 0} m^{-1}\omega_{uv}^{(m)}(\alpha, \beta, 0)\exp\left\{im\omega_r\left(t - \frac{\gamma}{\omega_r}\right)\right\} \quad [49]$$

$$= (i\omega_r)^{-1}\sum_{m\neq 0} m^{-1}\underbrace{\omega_{uv}^{(m)}(\alpha, \beta, 0)\exp\{-im\gamma\}}_{\omega_{uv}^{(m)}(\alpha,\beta,\gamma)} \times \exp\{im\omega_r t\} \quad [50]$$

Using the relationship between the Fourier components $\omega_u^{(m)}$ of orientations $\{\alpha, \beta, 0\}$ and $\{\alpha, \beta, \gamma\}$ (Eq. [33]) results in

$$\xi_{uv}(t - \gamma/\omega_r; \alpha, \beta, 0) = (i\omega_r)^{-1}\sum_{m\neq 0} m^{-1}\omega_{uv}^{(m)}(\alpha, \beta, \gamma)\exp\{im\omega_r t\} \quad [51]$$

which, by definition, equals $\xi_{uv}(t; \alpha, \beta, \gamma)$, as was to be shown.

If both sides of Eq. [48] are Fourier expanded, we obtain for the left side

$$\xi_{uv}(t; \alpha, \beta, \gamma) = \sum_{k=-\infty}^{\infty} C_{uv}^{(k)}(\alpha, \beta, \gamma)\exp\{ik\omega_r t\} \quad [52]$$

and for the right side

$$\xi_{uv}(t - \gamma/\omega_r; \alpha, \beta, 0) = \sum_{k=-\infty}^{\infty} C_{uv}^{(k)}(\alpha, \beta, 0)\exp\left\{ik\omega_r\left(t - \frac{\gamma}{\omega_r}\right)\right\} \quad [53]$$

$$= \sum_{k=-\infty}^{\infty} C_{uv}^{(k)}(\alpha, \beta, 0)\exp\{-ik\gamma\}\exp\{ik\omega_r t\} \quad [54]$$

Since Eq. [48] holds, we may equate Eqs. [52] and [54] to show that the coefficient $C_{uv}^{(k)}(\alpha, \beta, \gamma)$ originating from the orientation $\{\alpha, \beta, \gamma\}$ is related to that from the orientation $\{\alpha, \beta, 0\}$ according to (*24, 26*)

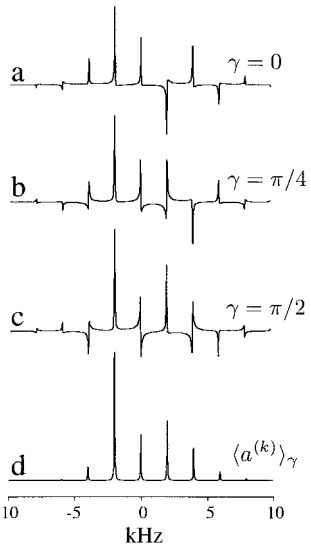

Figure 4 Simulated MAS spectra at $\omega_r/2\pi = 2$ kHz generated with a CSA tensor for various tensor orientations ($\omega_{aniso}/2\pi = 5$ kHz and $\eta = 0.35$). The spectra in (a)–(c) correspond to *fixed* values $\{\alpha = \pi/3, \beta = \pi/4\}$ but *variable* γ according to (a) $\gamma = 0$, (b) $\gamma = \pi/4$, and (c) $\gamma = \pi/2$. The spectra are not purely absorptive. The spectrum in (d) is the result after *averaging* over all values of γ. Note that all peaks are positive and purely absorptive, as expressed by Eq. [70].

$$C_{uv}^{(k)}(\alpha, \beta, \gamma) = C_{uv}^{(k)}(\alpha, \beta, 0)\exp\{-ik\gamma\} \quad [55]$$

Hence, the coefficients $C_{uv}^{(k)}$ originating from any carousel member is related to that from $\gamma = 0$. Combining Eqs. [47] and [55] then allows us to express $c_{uv}^{(k)}(\Omega)$ as follows (*24, 26*):

$$c_{uv}^{(k)}(\Omega) = \sum_{k'=-\infty}^{\infty} C_{uv}^{(k)}(\alpha, \beta, \gamma)C_{uv}^{(k')}(\alpha, \beta, \gamma)^* \quad [56]$$

$$= \sum_{k'=-\infty}^{\infty} C_{uv}^{(k)}(\alpha, \beta, 0)C_{uv}^{(k')}(\alpha, \beta, 0)^* \times \exp\{-i(k - k')\gamma\} \quad [57]$$

Note that the γ-dependence in Eq. [57] is factored out as a single complex exponential. Remembering that the sideband amplitude $a_{uv}^{(k)}$ in the NMR spectrum is related to $c_{uv}^{(k)}$ by a constant factor (Eq. [35]), this implies that for a fixed pair of $\{\alpha, \beta\}$, the kth-order sidebands, arising from each member of the carousel, are of *equal magnitudes* but *different phases.* The amplitudes are in general complex numbers, leading to spectral peaks not being purely absorptive (*24*). This is, for example, obvious in the simulated spectra shown in Figs. 4(a)–(c). Each of these spectra are generated from each of three members of a "carousel," i.e., single crystallite orientations with fixed values of α and β but different values of γ.

Carousel Averaging. We are now prepared for the key result of this section; a fast route to calculate averages over the angle γ for samples having carousel symmetry. We concluded earlier that the only orientational-dependent part of the NMR time-domain signal is the coefficient $c_{uv}^{(k)}(\alpha, \beta, \gamma)$, which together with the product $a_{uv} = \langle u|\hat{\rho}(0)|v\rangle\langle v|\hat{\mathbf{Q}}|u\rangle$ defines the amplitudes of the spinning sidebands in the spectrum. Since a_{uv} is independent on the molecular orientation, we may focus on averaging $c_{uv}^{(k)}(\Omega)$ with respect to γ, i.e., to evaluate the integral

$$\langle c_{uv}^{(k)}(\Omega)\rangle_\gamma = \frac{1}{2\pi}\int_0^{2\pi} d\gamma c_{uv}^{(k)}(\Omega). \quad [58]$$

This is referred to as "*carousel averaging,*" as the integration is carried out over all orientations γ of a given carousel. Inserting Eq. [57] into the carousel average gives

$$\langle c_{uv}^{(k)}(\Omega)\rangle_\gamma = \sum_{k'=-\infty}^{\infty} C_{uv}^{(k)}(\alpha, \beta, 0)C_{uv}^{(k')}(\alpha, \beta, 0)^* \times \frac{1}{2\pi}\int_0^{2\pi} d\gamma \exp\{-i(k - k')\gamma\} \quad [59]$$

This expression involves the integral of the function $\exp\{-i(k - k')\gamma\}$ over all orientations γ, the result of which may easily be seen by first considering a general complex exponential function $\exp\{iPx\}$, expressed as $\exp\{iPx\} = \cos Px + i \sin Px$. As for each positive value taken by the sine and cosine functions in the range $0 \leq x \leq 2\pi$, there is a corresponding negative value of the same amplitude; hence, the integral vanishes (or, viewed equivalently, the area enclosed by the functions over the range $0 \leq x \leq 2\pi$ is zero). This holds *unless* $P = 0$ (or, more strictly, unless P is an integer multiple of π), for which the cosine part gives the constant value $\int_0^{2\pi} dx = 2\pi$. In the case of Eq. [59] this requires that the Fourier indices k and k' must be *equal* to give a *nonvanishing* contribution, i.e.,

$$\frac{1}{2\pi}\int_0^{2\pi} d\gamma\ \exp\{-i(k-k')\gamma\} = \delta(k,k') \qquad [60]$$

The carousel average may then be expressed (*24*, *26*)

$$\langle c_{uv}^{(k)}(\Omega)\rangle_\gamma = \sum_{k'=-\infty}^{\infty} C_{uv}^{(k)}(\alpha,\beta,0)C_{uv}^{(k')}(\alpha,\beta,0)^* \times \delta(k,k') \qquad [61]$$

$$= C_{uv}^{(k)}(\alpha,\beta,0)C_{uv}^{(k)}(\alpha,\beta,0)^* \qquad [62]$$

Note that because the expression $C_{uv}^{(k)}C_{uv}^{(k)}*$ is the product of a complex number and its conjugate we may write

$$\langle c_{uv}^{(k)}(\Omega)\rangle_\gamma = |C_{uv}^{(k)}(\alpha,\beta,0)|^2 \qquad [63]$$

This shows that the carousel average of $c_{uv}^{(k)}$ may be calculated by taking the square modulus of the Fourier coefficient $C_{uv}^{(k)}$ from the expansion of $\exp\{i\xi_{uv}(t;\Omega)\}$. However, the *same* average may alternatively be obtained by taking the square modulus of the coefficient $c_{uv}^{(k)}$ itself:

$$\langle c_{uv}^{(k)}(\Omega)\rangle_\gamma = |c_{uv}^{(k)}(\alpha,\beta,0)|^2 \qquad [64]$$

This may be proven from the relationship between $c_{uv}^{(k)}$ and $C_{uv}^{(k)}$ (Eq. [47]), and noting that the sum over all Fourier coefficients $C_{uv}^{(k)}$ is equal to the function $\exp\{i\xi_{uv}(0;\Omega)\}$,

$$\exp\{i\xi_{uv}(0;\Omega)\} = \sum_{k'=-\infty}^{\infty} C_{uv}^{(k')}(\Omega) \qquad [65]$$

as may be shown by inserting $t = 0$ in Eq. [41]. For brevity, in the following we do not explicitly indicate the dependence on Ω. From Eq. [65] it follows that Eq. [47] may be expressed

$$c_{uv}^{(k)} = C_{uv}^{(k)}\exp\{i\xi_{uv}(0)\}* \qquad [66]$$

Using this expression to evaluate the square modulus $|c_{uv}^{(k)}|^2 = c_{uv}^{(k)}c_{uv}^{(k)}*$ gives

$$|c_{uv}^{(k)}|^2 = (C_{uv}^{(k)}\exp\{i\xi_{uv}(0)\}*)\cdot(C_{uv}^{(k)}*\exp\{i\xi_{uv}(0)\}) \qquad [67]$$

$$= C_{uv}^{(k)}C_{uv}^{(k)}*\underbrace{\exp\{i\xi_{uv}(0)\}*\exp\{i\xi_{uv}(0\}}_{=1} \qquad [68]$$

$$= |C_{uv}^{(k)}|^2 \qquad [69]$$

justifying the use of Eq. [64] for calculating the carousel average.

Finally, by combining Eqs. [64] and [35] we may calculate the carousel average *sideband amplitude* $a_{uv}^{(k)}$ according to

$$\langle a_{uv}^{(k)}(\Omega)\rangle_\gamma = a_{uv}|c_{uv}^{(k)}(\alpha,\beta,0)|^2 \qquad [70]$$

Note that here we assumed the orientation $\gamma = 0$, but that the coefficient $c_{uv}^{(k)}$ from *any* value of γ may be used. Two important conclusions may be drawn from Eq. [70]:

1. After averaging over γ, the sideband amplitudes are *positive* and purely *real* (*24*). This is evidenced by the spectrum in Fig. 4(d), corresponding to the carousel average of the spectra from all members of the given carousel.
2. The γ-*averaged* sideband amplitudes may be obtained from the components $c_{uv}^{(k)}$ evaluated for *one* orientation $\{\alpha, \beta, \gamma = 0\}$. This "implicit" averaging over γ is appropriate for the frequency domain simulations discussed in Part II and summarized as a flowchart in Fig. II-5. An example of the incorporation of Eq. [70] into this algorithm is illustrated by the computer code in Appendix C and discussed further in a later section.

The sideband amplitudes averaged over *all* orientations may accordingly be calculated from Eq. [23] as an integral of the function $|c_{uv}^{(k)}(\alpha,\beta,0)|^2$ over the full ranges of orientations α and β

$$\langle a_{uv}^{(k)}(\Omega)\rangle_\Omega = \frac{1}{4\pi}\,a_{uv}\int_0^{2\pi} d\alpha \int_0^{\pi} \sin\beta d\beta |c_{uv}^{(k)}(\alpha,\beta,0)|^2 \qquad [71]$$

Note that the carousel averaging assists in bringing down the explicit integration (Eq. [23]) from three to two orientational variables: This leads to faster and more accurate numerical computations.

Classification of Orientational Symmetry

By "orientational symmetry" we mean the following: Assume a set of orientations $\{\Omega\}$ comprising N_{sym}

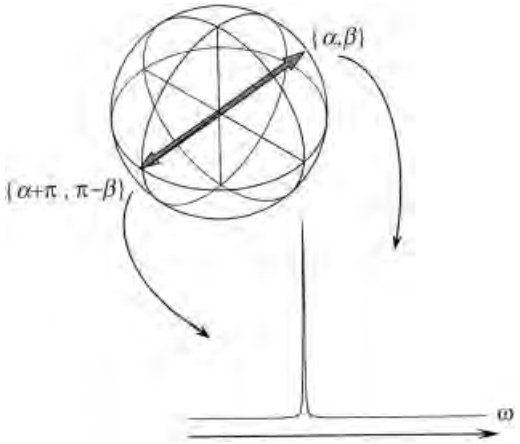

Figure 5 Example of orientational symmetry in solid-state NMR: Two tensors at two orientations that are related by inversion through the center of the sphere produce identical NMR spectra. The consequence of this orientational symmetry for powder averaging is that it is sufficient to calculate the spectrum for only *one* of the orientations.

values, where each angle in the set is related to the others by a certain mathematical operation. Each angle defines the orientation of a crystallite in the sample. If it is a priori known that all orientations in $\{\Omega\}$ produce *identical* NMR signals, it is sufficient to calculate $s(t; \Omega)$ for *only one* of them. In many cases, the NMR signal possesses such symmetries with respect to the orientational variables. Carousel symmetry is one such example. This allows for a reduction of the integration range when calculating powder averages and hence to faster computations—in the present example by a factor of N_{sym}.

As an illustration of orientational symmetry, consider the following case: Assume that the NMR time-domain signal depends on the two orientational angles α and β, for which $\{0 \leq \alpha < 2\pi, 0 \leq \beta \leq \pi\}$. This corresponds to integration over the full spherical surface. Consider two orientations related by inversion through the center of the sphere: $\Omega = \{\alpha, \beta\}$ and $\Omega' = \{\alpha + \pi, \pi - \beta\}$, depicted in Fig. 5. *If* the signal fulfills $s(t; \Omega) = s(t; \Omega')$ for all values of Ω, it is possible to reduce the integration range of β to $0 \leq \beta \leq \pi/2$, giving integration over *only* the upper hemisphere. The NMR spectrum is related by a Fourier transformation of $s(t; \Omega)$, and also the *spectra* from the two orientations are identical, i.e., $S(\omega; \Omega) = S(\omega; \Omega')$ (Fig. 5). Hence, the same symmetry consequences apply to the frequency-domain calculations discussed in Part II.

Such orientational symmetries are common in solid-state NMR (Fig. 6). As discussed in detail in Refs. (*21, 23, 26*), the symmetry of the NMR signal may conform to one of the following four cases:

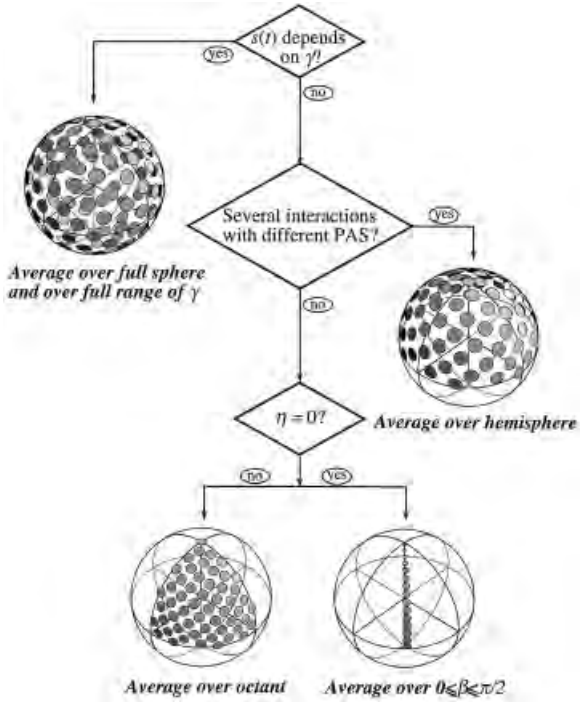

Figure 6 Overview of orientational symmetry in solid-state NMR and its implications for powder averaging (discussed in the *Classification of Orientational Symmetry* section). The ranges of the orientational variables α and β required in the powder averaging procedure are indicated in each case. These ranges depend on the number of interactions involved in the calculation and if the tensors are axially symmetric ($\eta = 0$) or axially asymmetric ($\eta \neq 0$).

1. The signal depends on all angles of the Euler triplet. No further reduction of the full integration range $\{0 \leq \alpha < 2\pi, 0 \leq \beta \leq \pi, 0 \leq \gamma < 2\pi\}$ is then in general possible. This applies to spin systems experiencing time-dependent RF fields, or if the initial density operator is orientationally dependent at the start of the experiment. The former case applies when simulating experiments employing multiple-pulse sequences, for example, for the purpose of spin decoupling (aiming at removing the effects of the dipolar couplings) (*1, 2, 4*). The latter case applies at the start of acquisition of the second time dimension ("$t_2 = 0$") at different values of evolution intervals ("t_1") in two-dimensional NMR experiments, leading to a dependence of $\hat{\rho}(t_1, t_2 = 0)$ on the molecular orientation. The procedure of carousel averaging discussed in the previous section has to be modified to account for this situation (*20, 23*), as it was assumed that $\langle u|\hat{\rho}(0)|v\rangle$ in Eq. [35] is orientation independent. This invalidates the use of Eq.

[70] for performing carousel averaging in the case of two-dimensional NMR experiments.

Whenever the signal depends on only the two angles $\{\alpha, \beta\}$, higher orientational symmetries are *always* present. This holds for all static cases and for all rotating solids having carousel symmetry (provided that the calculation first employs Eq. [70] for averaging over γ). For all these cases, the generic integral for the powder average is

$$\bar{s}(t) = \frac{1}{K} \int_0^{\alpha_{max}} d\alpha \int_0^{\beta_{max}} \sin\beta d\beta \; s(t; \alpha, \beta) \quad [72]$$

where α_{max}, β_{max}, and the normalization constant K depend on the particular orientational symmetry of the NMR signal. Depending on the spin system, one of the three following cases apply.

2. *Several* interactions, employing an arbitrary molecular frame: This restricts the integration over a *hemisphere* (*26*). Equation [72] is used with $\alpha_{max} = 2\pi$, $\beta_{max} = \pi/2$, and $K = 2\pi$:

$$\bar{s}(t) = \frac{1}{2\pi} \int_0^{2\pi} d\alpha \int_0^{\pi/2} \sin\beta d\beta \; s(t; \alpha, \beta) \quad [73]$$

Two explicit examples of such cases are considered below.

3. *Single axially asymmetric* ($\eta \neq 0$) interaction, or several interactions with coinciding principal axis systems. If the transformations are performed directly from the PAS of the interaction, by identifying Ω_{PL} (static sample) or Ω_{PR} (rotating sample) with the orientational variable, it is sufficient to integrate over an *octant* of the spherical surface (*26*). This gives $\alpha_{max} = \pi/2$, $\beta_{max} = \pi/2$, and $K = \pi/2$:

$$\bar{s}(t) = \frac{2}{\pi} \int_0^{\pi/2} d\alpha \int_0^{\pi/2} \sin\beta d\beta \; s(t; \alpha, \beta) \quad [74]$$

An example of such a case is the simulation of the spectrum from the CSA interaction of one nuclear spin in a powdered sample.

The final scenario is a special case of 3.

4. *One axially symmetric* ($\eta = 0$) interaction (e.g., a dipolar interaction), or several axially symmetrical interactions with coinciding principal axis systems: By directly transforming from the PAS of the interaction(s), it is sufficient to integrate along the spherical arc $\{\alpha = \text{arbitrary}; 0 \leq \beta \leq \pi/2\}$:

$$\bar{s}(t) = \int_0^{\pi/2} \sin\beta d\beta \; s(t; \beta) \quad [75]$$

This follows directly by applying the arguments given in section "*Orientational Variables*" to case 3 above.

Examples. We illustrate the consequences of orientational symmetry for powder averaging by considering a heteronuclear ^{13}C–^{1}H spin pair from a molecular fragment of an organic solid. Assume that the ^{13}C spectrum is to be calculated from a static sample in the form of a finely ground powder. Three types of spin interactions need to be taken into account: The chemical shift interactions of each nuclear spin (including isotropic as well as anisotropic shifts) and the heteronuclear ^{13}C–^{1}H dipolar interaction. We know that the total spin Hamiltonian is dynamically inhomogeneous because all interactions commute with each other (see the discussion in Part I).

Assume that a molecular frame is chosen such that its z-axis coincides with the C–H internuclear vector, i.e., that the PAS of the dipolar interaction coincide with that of the molecular frame. If the PAS of each CSA tensor is transformed into the molecular frame and this is used as the starting reference frame for the subsequent powder averaging, the system conforms to case 2 above. Consequently, powder averaging is performed over α_{MR} and β_{MR} using Eq. [73], i.e., integrating over a hemisphere.

Next, assume that the ^{13}C–^{1}H dipolar interaction is *removed* by the use of a heteronuclear decoupling sequence (*1, 2, 4*) applied to the proton during the signal acquisition. If the decoupling sequence involves changing the RF amplitude, phase, or frequency, an accurate numerical simulation must take the RF pulses explicitly into account. As discussed in Part I, because the RF Hamiltonian does not commute with the Hamiltonian for the ^{1}H chemical shift and that for the heteronuclear dipolar coupling, the total spin Hamiltonian is dynamically homogeneous. This requires a "small-step" integration of the Schrödinger equation (thus breaking the problem down to a sequence of piece-wise time-*in*dependent Hamiltonians), as explained in Part II. As far as powder averaging is concerned, the spin system now conforms to case 1, and averaging needs to be performed over the full ranges of all three Euler angles.

We assume instead for simplicity that the heteronuclear decoupling may be treated "phenomenologi-

cally" in the sense that it effectively removes all effects of the dipolar coupling (without considering explicitly the decoupling *mechanism*) and furthermore leaves all other spin interactions unaffected. While this certainly is a crude approximation, it often accurately accounts for the observed spin dynamics and leads in this case to a significant simplification of the spin dynamics calculation. Only two interactions remain, i.e., the chemical shift interactions of each spin. Also, as discussed in more detail below, due to the commutation of spin operators from different spin species the ^{1}H chemical shift need not be considered in the calculation of the ^{13}C spectrum. In other words, the ^{1}H chemical shift does not affect the ^{13}C spin dynamics, and the entire simulation reduces to calculating the powder spectrum from a single chemical shift interaction. If the ^{13}C CSA tensor is axially asymmetric, this case conforms to 3 and powder averaging is performed over the variables α_{PL} and β_{PL} over the ranges $\{0 \leq \alpha \leq \pi/2,\ 0 \leq \beta \leq \pi/2\}$. However, if the tensor happens to be axially symmetric, case 4 is relevant with averaging over only the single variable β_{PL}.

Finally, assume instead the MAS spectrum is to be calculated from a sample comprising an ensemble of "isolated" quadrupolar spins. If each spin of the ensemble spin is subject to the first-order quadrupolar interaction, but has negligible second-order quadrupolar effects (*27*), its Hamiltonian is dynamically inhomogeneous. While this is an uncommon situation in quadrupolar NMR, it is relevant, for example, to many cases of ^{2}H and ^{7}Li. For powder averaging, the averaging over γ_{PR} is first carried out through Eq. [70]. Once this is done, the orientational symmetry with respect to α_{PR} and β_{PR} conforms to either case 3 or 4, depending on whether the electric field gradient tensor (*27*) is axially symmetric or axially asymmetric, respectively.

In the following, we focus on cases involving *at most* two orientational variables in the powder average calculation. In particular, this applies to all dynamically inhomogeneous problems for which we outlined spin dynamic computational algorithms in Part II and Part III. However, in the case of sample rotation it is required that the carousel averaging (Eq. [70]) is employed before averaging over α and β.

NUMERICAL ESTIMATES OF POWDER AVERAGES

The Powder Average as a Weighted Sum

There exists no general analytic solution to the integrals in Eqs. [22] and [72]. Instead, one has to resort to a numerical approximation of the exact powder-averaged NMR response. The integral is usually estimated as follows: First, a set of orientations are selected. Then, the NMR time-domain signal or frequency-domain spectrum is calculated for *each* orientation, and the powder average is approximated as a *weighted summation* over the entire set of signals:

$$\bar{s}^{\mathcal{S}}(t) = \sum_{j=1}^{N^{\mathcal{S}}} w_j^{\mathcal{S}} s(t;\ \Omega_j^{\mathcal{S}}), \qquad [76]$$

Here, $\bar{s}^{\mathcal{S}}(t)$ represents the *estimated* powder average using the discrete set $\mathcal{S} = \{w_j^{\mathcal{S}},\ \Omega_j^{\mathcal{S}}\}$ comprising $N^{\mathcal{S}}$ orientations and weights. The weights should be normalized so that their sum equals unity. The goal is to obtain a good approximation $\bar{s}^{\mathcal{S}}(t)$ of the exact powder average $\bar{s}(t)$, using the smallest possible number of orientations $N^{\mathcal{S}}$. This is desirable because the computation time grows linearly with $N^{\mathcal{S}}$ and the calculation of the time signal or spectrum may be time consuming even for a single orientation.

Selecting the Orientations

In this section, we briefly comment on some arguments used when devising schemes for generating sets of orientations for the purpose of powder averaging. For details on the principles behind the various powder methods we refer to Refs. (*16–18, 26, 28–35*); here, we will mainly be interested in *using* them.

The most intuitive approach is to select the orientations randomly because this corresponds to the situation in the physical sample. However, this is an inefficient strategy from a computational standpoint: Often, 10^4–10^5 random orientations are needed for obtaining accurate powder averages. Instead, the problem of finding better solutions is usually attacked by aiming for "small" sets that provide accurate powder averages by aiming at *uniform* distributions of the orientations using geometric arguments (*16–18, 28–32, 34*).

When devising methods for selecting orientations, the link between the Euler angles $\{\alpha, \beta\}$ and the polar angles $\{\theta, \phi\}$ is often exploited (see Fig. I-4): The implementation of powder averaging is then to integrate the orientationally dependent NMR signal over the spherical surface. Depending on the region over which the distributions are generated, powder methods may be classified either as "*planar*" or "*spherical*" (*18, 34*).

The aim of the *spherical* approaches is to find as uniform a distribution as possible over the surface of the unit sphere. There exist a vast number of solutions

to this problem (*16–18, 31, 34*). As an example, we describe the *REPULSION* method (*18*). The orientations are selected as follows: First, $N^{\mathcal{P}}$ "particles" are placed onto the spherical surface, the location of each being parameterized by two Euler angles. Each particle is associated with a point charge, giving rise to a repulsive force to its neighbors. The particles are subsequently freely moved on the surface until the total repulsive potential is minimized. This results in a uniform distribution of orientations, with nearly equal weights.

The *planar* methods (*28–30, 34*) first sample points over the two-dimensional planar region $\{0 \leq x < 2\pi,\ 0 \leq y \leq \pi\}$, where uniform distributions are easier to arrange them over the spherical surface. These samples are subsequently mapped onto the sphere, i.e., the $\{x, y\}$ coordinates are converted into the Euler angles $\{\alpha, \beta\}$. This requires taking the weighting factor sin β factor in Eq. [22] into account. The planar methods may be implemented in many different versions depending on (1) how the planar region is *sampled* and (2) how these samples are subsequently *mapped* onto the sphere. The most common approach is to divide the two-dimensional integration range into a grid of rectangles of equal size, and then choosing the samples $\{x_i, y_j\}$ at the vertices ("closed form") or at the centers ("open form") of the rectangles (*34*).

One of the simplest and most commonly used planar methods is the "STEP method." It divides the planar region $\{0 \leq x < 2\pi,\ 0 \leq y \leq \pi\}$ into equal meshes and maps the $\{x, y\}$ coordinates onto the sphere by identifying $\alpha = x$ and $\beta = y$. The weight associated with $\{\alpha_j, \beta_j\}$ is given by $w_j = \sin \beta_j$.

Another commonly used planar technique is the *ZCW* method (*28–30*), named after inventors Zaremba, Conroy, and Wolfsberg. It is used for integration over both two and three Euler components.

Appendix A contains the formulas for generating powder sets of the STEP and ZCW methods under various symmetries, as well as short C routines for implementing them. Some STEP and ZCW sets are visualized in Fig. 7. Other sets of orientations generated from a large variety of methods may be downloaded from the websites given in Refs. (*36–38*).

Various general criteria have been suggested for probing the "efficiency" of a powder method. The "ideal" scheme should provide sets of orientations that are completely uniformly distributed over the spherical surface. This gives equally weighted orientations: $w_j = 1/N^{\mathcal{P}}$. In practice, however, it is difficult to produce completely uniform distributions and it is, therefore, necessary to compensate for the nonideality by assigning a weight to each orientation that truly reflects its solid angle (*16–18*) in the weighted summation (Eq. [76]).

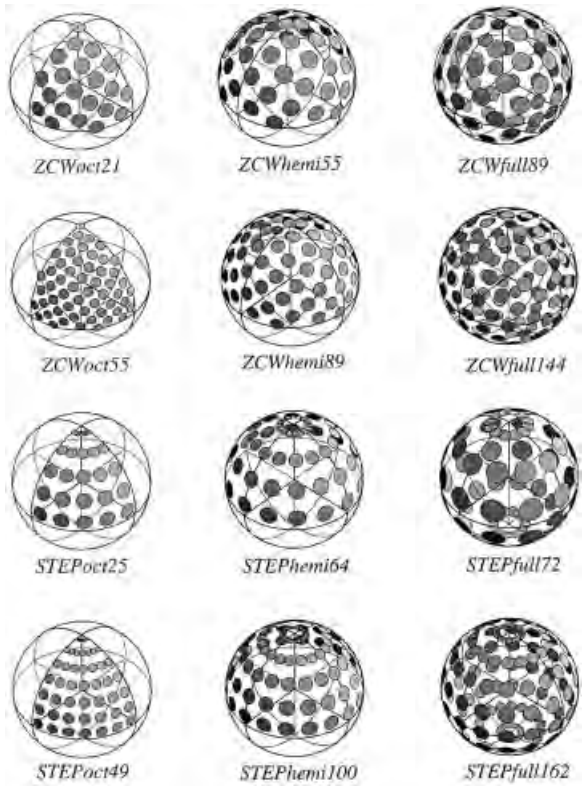

Figure 7 Selection of orientations plotted on the surface over various regions of the sphere. These sets were generated by the STEP and ZCW methods by the routines given in Appendix A. The total number of orientations comprised by each set is indicated beneath its sphere, and the "size" of each orientation reflects its weight w_j, as defined by Eqs. [105] and [114] for the STEP and ZCW schemes, respectively.

Computer Implementations of Powder Averaging

In a simulation program, Eq. [76] lends itself to a convenient implementation in terms of loops over the orientational variables. For each orientation, the NMR time-signal/spectrum is calculated as described in Part II. Then, the powder averaged signal/spectrum is updated by adding the result from each orientation multiplied by its weight according to Eq. [76]. Two possible implementations of this procedure are shown as flowcharts in Fig. 8. The orientations and weights must be stored in the memory of the computer. This can be arranged in either of the following ways:

1. The most flexible approach is to once store the angles and weights in a file and retrieve them at

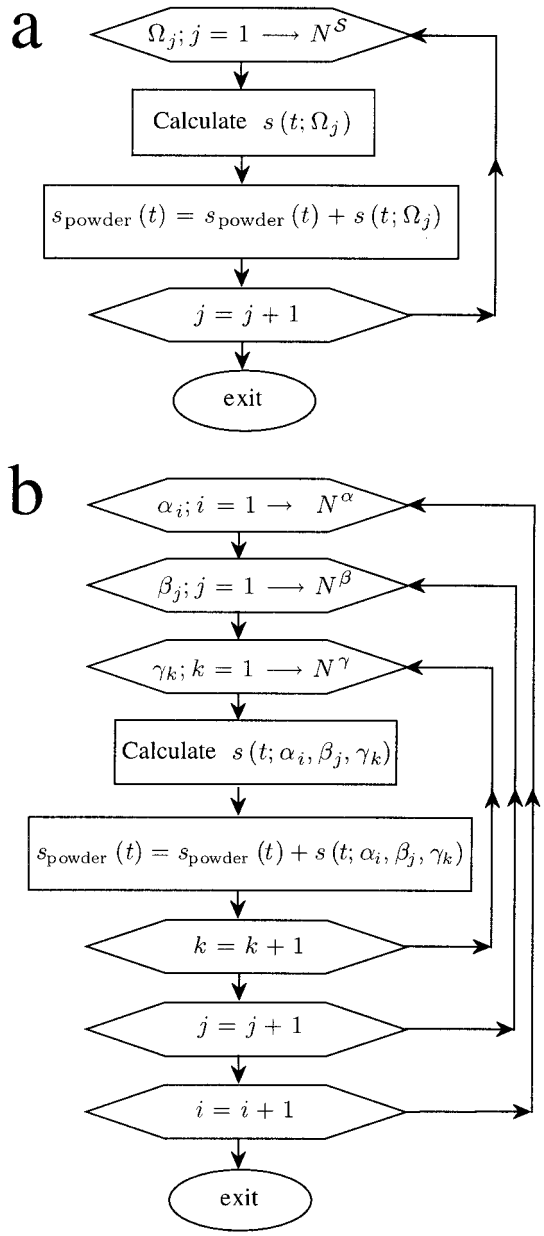

Figure 8 Flowcharts for two different loop implementations of powder averaging by (a) assigning the angles of the Euler triplet simultaneously for each orientation and (b) independently changing each of the angles α, β, and γ. In both cases, for each combination of Euler angles α_i, β_j, and γ_k the time-domain signal is calculated for the given orientation and the powder-averaged signal (s_{powder}), estimated as a weighted sum, is updated. In (b), the values α_i and β_j in the two outer loops are fixed while the index k is incremented. Similarly, a total of $N^\gamma N^\beta$ orientations are used for a fixed value α_i, before the outermost loop variable is incremented.

the start of each simulation. This procedure may be implemented in various ways, for example, by storing each of the lists of angles $\{\alpha_j\}$, $\{\beta_j\}$, and $\{\gamma_j\}$ as floating-point numbers in one-dimensional arrays. Object-oriented languages, such as C++, offer more elegant solutions: By defining a "Euler angle data type," the Euler angle triplet Ω may contain all three angles in a single object. A primitive version of such an "Euler angle class" (`euler`) was introduced in Part II and will be used later.

2. If the expressions defining the orientations and weights are attainable through closed formulas, they may be calculated directly when needed within the powder loop. Then, the powder averaging (Eq. [76]) may be implemented in terms of *nested loops*, as shown in Fig. 8(b). This involves running over all values of the variable in the inner loop while keeping the value of the variable in the outer loop fixed. It is important to arrange the loops so as to avoid unnecessary calculations that slow down the performance of the program. For example, an implementation of the STEP method should employ the loop over β as the outermost: For *each* value of β_j, there are $N^\alpha N^\gamma$ orientations $\{\alpha_i, \beta_j, \gamma_k\}$ with $i = 1, 2, \ldots, N^\alpha$ and $k = 1, 2, \ldots, N^\gamma$; the weight of *all* these orientations is the same, as it is proportional to $\sin \beta_j$. The evaluation of trigonometric functions are relatively time-consuming operations and should therefore be done only once for each value β_j.

Of the two approaches outlined above, we recommend using the former: Depending on the experimental conditions to be simulated, specialized spin dynamics routines may be needed. The structure of the powder loop, on the other hand, is the same for all simulations of powders. Strategy 1 provides better portability of the code and gives higher flexibility when executing the simulation program because any given set of orientations from any powder scheme can conveniently be selected from a parameter file, instead of being determined when the program is compiled. Usually, this also leads to faster execution of the program because the orientations and the weights themselves need not be calculated. Approach 2 has, however, one advantage: During runtime, it requires less computer memory because each orientation is generated *directly* ("on the fly") when it is needed, and not stored in arrays before the powder loops start.

PRACTICAL RECOMMENDATIONS FOR POWDER AVERAGING

Accuracy of the Simulated Data

When simulating NMR spectra of powders, it is important to ensure that the particular scheme for generating the orientational angles is implemented with care so as to give correct results for any given situation. The simulation of accurate line shapes is especially important when interaction parameters are extracted by iterative fitting of simulations to experimental data. Simulation artifacts induced by insufficient or erroneous powder averaging give systematic errors in the determination of the interaction parameters. Such aspects of powder averaging when extracting interaction parameters through numerical fitting procedures are discussed in Refs. (*39, 40*).

For calculated spectra of *rotating solids,* the integration error primarily shows up as erroneous *amplitudes* of the spinning *sidebands.* The breadth of the sideband pattern, as well as the relative sideband amplitudes within the manifold, reflect the anisotropy frequency and asymmetry parameter of the NMR interaction tensor. Hence, when fitting simulations to experimental results erroneous sideband intensities produce errors in the estimate of these parameters.

In the case of *static solids,* the anisotropy and asymmetry parameters are reflected in the *shapes* of the *powder patterns,* as illustrated in Fig. I-7. As discussed further below, simulations of spectra of static powders require in general a larger number of orientations than those of rotating samples because the broad lines of static spectra require a certain minimum number of orientations to be faithfully represented as a histogram of stick spectra of the type discussed in Part II. Figure 9 illustrates the effects of including insufficient number of orientations in the powder average calculation of a spectrum from a static sample: The line shapes are not smooth but "rippled."

The number of orientations necessary to produce converged results is, of course, not known a priori. Apart from the efficiency provided by the set of angles, it depends on many factors, for example, the size and relative orientations of the NMR interaction tensors involved in the simulation, as well as on the experimental conditions that are simulated. Usually, it is necessary to perform a series of calculations with identical parameters but using different numbers of orientations in the powder average, until the simulated data no longer change perceptibly when the size of the sets of orientations are increased. For example, from the simulated spectra in Fig. 9 we conclude that 2584 orientations is required for that particular case.

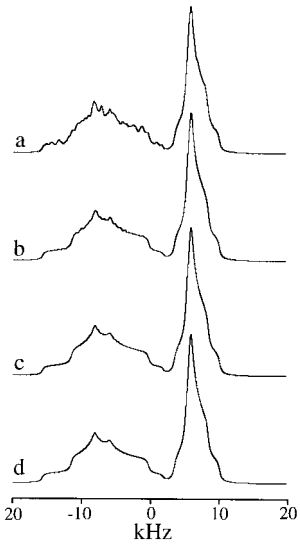

Figure 9 Effect of including insufficient number of orientations in the calculation of a static ^{13}C NMR powder spectrum of glycine. The spectra are obtained from (a) 233, (b) 987, (c) 2584, and (d) 46,368 ZCW orientations generated over a hemisphere (see Fig. 7 for a graphic visualization of such sets). The lineshape distortions ("rippling") is in particular evident in (a). The spectrum (c) is the output of the computer program given in Appendix B and may be taken as "converged" as it is indistinguishable from the spectrum obtained by the larger number of orientations in (d).

Choice of Powder Method

Powder schemes display different convergence properties depending on the given situation. In some applications there are large differences in performance between different methods, whereas in other cases they are of similar quality. Selection of a particular method for generating the sets of orientations and weights depends on the given situation, which usually requires gaining insight by comparing different schemes for several representative test simulations. Here, we limit ourselves to briefly indicate our choice of method in three generic cases, assuming averaging over α and β only. For cases also requiring averaging over γ we recommend using sets of the three-angle ZCW method as given in Refs. (*29, 30*).

1. Simulating spectra of static powders is a challenging problem from a powder averaging standpoint because the line shapes are broad and must be accurately represented over a large

frequency range. Often, many thousands of orientations are needed to give smooth line shapes. Here, we recommend using either the method of Alderman et al. (*16*) or the two-angle ZCW scheme (*28–30, 32*). They have proven to be efficient for a wide range of problems, and large sets of orientations may easily be generated. See Appendix A for a ZCW implementation.

2. When simulating the spinning sideband patterns generated from dynamically inhomogeneous systems under MAS, a powder averaging technique called Gaussian spherical quadrature (*26, 41, 42*) has been demonstrated to be particularly efficient, allowing for the calculation of accurate sideband amplitudes using sets comprising less than 100 orientations. Other well-adapted schemes to these problems are ZCW (*28–30*) and REPULSION (*18*).
3. The spectral peaks from strongly coupled spin systems under MAS, or from experiments employing off-magic angle spinning, are often much broader than those of case (2). Usually, a few hundred orientations are required for converged amplitudes and shapes of the spinning sidebands. Here, we recommend using either the ZCW method or REPULSION.

EXAMPLE I: POWDER PATTERN OF A STATIC SOLID

This section outlines how the NMR powder spectrum from two homonuclear spins may be simulated, including the chemical shift, dipolar, and J interactions. A static sample of $^{13}C_2$-labeled glycine in the form of a finely ground powder represents such a case. A ball-and-stick structure of a glycine molecule is shown in Fig. I-1. In the following, we use the labels "1" and "2" for the ^{13}C spins of the carboxyl (COOH) and methylene sites (CH_2), respectively. The experimental ^{13}C spectrum recorded at a magnetic field of 9.4 T (corresponding to a ^{13}C Larmor frequency of $\omega_0/2\pi = -100.66$ MHz) is shown in Fig. 10. During the signal acquisition, high-power proton decoupling (*1, 2, 4*) was employed to remove the heteronuclear ^1H–^{13}C dipolar couplings. Therefore, we assume in the following discussion that we may consider the two ^{13}C spins as isolated from the surrounding protons. The NMR signal was recorded after first generating observable transverse magnetization by a $(\pi/2)_y$ pulse as discussed in Part I. In the framework of the density operator formalism, the effect of the RF pulse is to convert the thermal equilibrium spin pair ensemble into an ensemble state represented by $\hat{\rho}(0) = \hat{\mathbf{I}}_x \equiv \hat{\mathbf{I}}_{1x} + \hat{\mathbf{I}}_{2x}$ at the start of signal acquisition. The transverse magnetization is subsequently allowed to evolve under the various spin interactions, described by a sum of time-independent Hamiltonians, whose expressions are derived below. The detected NMR time-domain signal corresponds to the expectation value of the operator $\hat{\mathbf{I}}^+$: $s(t) = \langle \hat{\mathbf{I}}^+ \rangle(t)$. Subsequent Fourier transformation produces the frequency-domain spectrum $S(\omega)$.

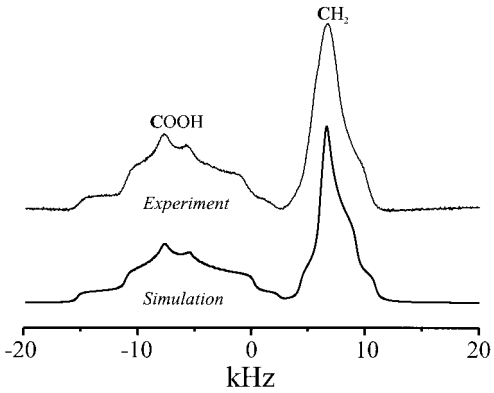

Figure 10 (a) Experimental ^{13}C spectrum of a static powder of 99% $^{13}C_2$-labeled glycine acquired at a magnetic field of 9.4 T. (b) Simulation of the spectrum using the program in Appendix B and spin interaction parameters taken from Ref. (*48*).

In the remainder of this section, we derive the form of the spin Hamiltonian representative for the two coupled ^{13}C spins in a molecule of glycine and discuss an explicit computer implementation for calculating the NMR response from a *powder* of glycine. The source code is given in Appendix B.

Hamiltonian

According to the theory given in Part I, we start deriving the explicit form of the Hamiltonian from the irreducible spherical tensor (IST) formalism (Eq. [I-60]). The Hamiltonian including all interactions for the spin pair 1–2 is given by

$$\hat{\mathbf{H}} = \hat{\mathbf{H}}^1_{\mathrm{CS}} + \hat{\mathbf{H}}^2_{\mathrm{CS}} + \hat{\mathbf{H}}^{12}_D + \hat{\mathbf{H}}^{12}_J \qquad [77]$$

where $\hat{\mathbf{H}}^1_{\mathrm{CS}}$ and $\hat{\mathbf{H}}^2_{\mathrm{CS}}$ represent the chemical shift Hamiltonians of spins 1 and 2, respectively, whereas $\hat{\mathbf{H}}^{12}_D$ and $\hat{\mathbf{H}}^{12}_J$ correspond to the dipolar and J-coupling Hamiltonians, respectively.

The homonuclear dipolar Hamiltonian is obtained by inserting the relevant tensor expressions from Table I-2 into Eq. [I-60], resulting in

$$\hat{\mathbf{H}}_D^{12} = C^D[A_{20}^{12}]^L \hat{\mathbf{T}}_{20}^{12} \quad [78]$$

$$= \omega_D^{12} \frac{1}{\sqrt{6}} (3\hat{\mathbf{I}}_{1z}\hat{\mathbf{I}}_{2z} - \hat{\mathbf{I}}_1 \cdot \hat{\mathbf{I}}_2) \quad [79]$$

where ω_D^{12} is the dipolar coupling frequency. Because this problem involves several interactions, we employ transformations via a molecule-fixed frame (Eq. [I-95]):

$$\omega_D^{12} = \sqrt{6} b^{12} \sum_{m=-2}^{2} D_{0m}^2(\Omega_{PM}^{12}) D_{m0}^2(\Omega_{ML}) \quad [80]$$

where b^{12} is the dipolar coupling constant in Table I-3.

Similarly, the J-coupling Hamiltonian is (ignoring the second-rank contributions of the J-tensor for simplicity)

$$\hat{\mathbf{H}}_J^{12} = 2\pi J^{12} \hat{\mathbf{I}}_1 \cdot \hat{\mathbf{I}}_2 \quad [81]$$

Next, we consider the chemical shifts and construct the Hamiltonian for spin 1. The expression for spin 2 is identical, with the label "2" replacing "1." The chemical shift Hamiltonian $\hat{\mathbf{H}}_{\mathrm{CS}}^1$ is conveniently separated into its isotropic and anisotropic parts, deriving from the $l = 0$ and $l = 2$ ISTs, respectively:

$$\hat{\mathbf{H}}_{\mathrm{CS}}^1 = \hat{\mathbf{H}}_{\mathrm{iso}}^1 + \hat{\mathbf{H}}_{\mathrm{CSA}}^1 \quad [82]$$

From the calculation given on page 128 of Part I, it follows that the isotropic part may be written

$$\hat{\mathbf{H}}_{\mathrm{iso}}^1 = \omega_{\mathrm{iso}}^1 \hat{\mathbf{I}}_{1z} \quad [83]$$

with the isotropic chemical shift frequency ω_0^1 given by Eq. [I-84]. A similar treatment of the CSA Hamiltonian results in

$$\hat{\mathbf{H}}_{\mathrm{CSA}}^j = \omega_{\mathrm{CSA}}^1 \sqrt{\frac{2}{3}} \hat{\mathbf{I}}_{1z} \quad [84]$$

with the anisotropic chemical shift frequency

$$\omega_{\mathrm{CSA}}^1 = \sum_{m,m'=-2}^{2} [\omega_{2m'}^1]^P D_{m'm}^2(\Omega_{PM}^1) D_{m0}^2(\Omega_{ML}) \quad [85]$$

and $\omega_{2m'}^1$ are the PAS components of the CSA tensor in frequency units (discussed on page 132 of Part I).

When the expressions for the various spin Hamiltonians are summed up in Eq. [77], we get the total Hamiltonian for the spin pair of a single molecular orientation Ω_{ML}.

Numerical Implementation

Overall Structure. The C code for the computer implementation that carries out the explicit calculations is given in Appendix B. In this section we explain the code. It follows the general steps for the spin dynamics calculation outlined in Part I, using the code given in Part II and with an overall structure conforming to the general hierarchy of a C/C++ program (*43–45*).

The program starts with including some files that defines general utility functions needed for the calculations. These involve: (1) the file `math.h` of the standard C library (*43*), containing functions for general mathematical operations; (2) `complex.h` and `matrix.h` [obtained from the GAMMA library (*46, 47*)], for handling complex numbers and matrices, respectively; (3) `aux.h`, comprising routines for dynamic memory allocations and Fourier transforms (*14*); (4) `spinOperators.h`, defining the matrix representations for the spin operators used; (5) `spinDynamics.h`, which corresponds to the routines for calculating the NMR response in the time and/or frequency domains from a single crystallite in the powder. These routines were given explicitly in Part II; (6) `powderSchemes.h`, comprising the functions given in Appendix A.

Next, the data-type `spinPars` is defined: A variable of this type contains three matrices, each of which represents a second-rank tensor of one interaction (in this case, two CSA tensors and one dipolar tensor).

After these initializations, the remaining of the code is structured into three routines as follows:

```
void getPowderSpectrumStatic( const spinPars &data,const matrix &Hiso, const matrix &rho0,
                              const matrix &Q, complex *ampPowder,int n_points,double sw,
                              euler *ang_list,double *weight_list,int n_angles)
{
 . . .
}

void getPowderSignalStatic( const spinPars &data,const matrix &Hiso, const matrix &rho0,
                            const matrix &Q, complex* signalPowder,int n_points,double sw,
                            euler *ang_list,double *weight_list,int n_angles)
{
 . . .
}

main()
{
 . . .
}
```

The program is designed to generate the powder NMR spectrum using either the direct method in the time-domain or the frequency-domain algorithm (see Part II), corresponding to using the subroutines `getPowderSignalStatic( )` and `getPowderSpectrumStatic( )`, respectively. The routine `main( )` is the heart of the program and its structure follows the general guidelines for an NMR simulation program, as outlined in Part I and II: (1) First, the relevant input parameters are processed; (2) next, the spin dynamics calculations are initiated; (3) finally, the output of these calculations undergo additional processing (e.g., line broadening).

In the following, we discuss each of steps 1–3 in detail.

Initialization. The code for carrying out the initialization steps are located in the routine `main( )`. For the sake of keeping the code brief, the input parameters are given directly in the source code. However, as discussed in Part II, a more convenient solution for feeding the input data into the computer program is, in general, to provide them from a parameter file because a simulation requiring any change in these parameters currently necessitates recompiling the code.

The initialization starts by defining the interaction parameters needed for the subsequent construction of the interaction tensors in their respective PAS. Note that the actual numbers used for the frequencies [taken from Ref. (*48*)] are input in Hz. These are, however, directly converted into angular frequencies through Eq. [I-24], which is performed by the simple routine `HzToRads`, located in the included file `aux.h`, and defined as

```
double HzToRads(double Hz)
                ×{ return Hz*(2.*Pi); }
```

Further, the orientation of each PAS relative to the molecular frame is needed. The latter is in the present code (arbitrarily) chosen such that its z-axis is along the internuclear 1–2 vector, i.e., coincident with the PAS of the dipolar coupling. The corresponding transformation angles of the two CSA tensors and the dipolar tensor is represented by the variables `PM_CSA_1`, `PM_CSA_2`, and `PM_D`, respectively. These are of data type `euler`, defined in Part II.

Next, the matrices for the spin operators are constructed and the initial density operator (represented by the variable `rho0` in the code) and observable operator (`Q`) defined.

The definition of input parameters concludes by choosing a frequency span of the spectrum (`sw`) and the number of spectral coordinates (`n_points`) (it also represents the number of time points sampled in the direct time-domain calculation).

The next initialization step involves allocating memory for the arrays needed in the simulation; these include arrays of spectral frequency coordinates and amplitudes, as well as lists containing the sets of orientations and weights used in the powder averaging procedure. As far as the latter is concerned, we note that the present simulation conforms to category 2 on page 36. The current code constructs 2584 ZCW orientations for the powder calculation, generated over the upper hemisphere using the routine given in Appendix A.

Next, each tensor is constructed in its principal axis system. This is carried out by the routine `L2TensorSetup`, explained in the section *Initial Steps of Numerical Simulations* of Part II. The tensors are subsequently transformed to the molecular frame, effected by the `L2tensor_transform` routine (also defined in Part II). The resulting tensors are subsequently stored in the variable data.

Finally, the sum of the Hamiltonians for the isotropic spin interactions is calculated (i.e., the two isotropic chemical shifts and the J coupling). Note that, as opposed to the Hamiltonians for the anisotropic interactions (which depend on the molecular orientation), the isotropic Hamiltonian needs only be formed *once*. The matrix representing the Hamiltonian for the sum of the isotropic interactions are subsequently employed as input parameters to the spin dynamics routines, together with the initial density operator and observable operator.

Spin Dynamics Calculations and Powder Averaging. At this point, everything is prepared for starting the spin dynamics calculations and powder averaging. The particular choice of simulation strategy, i.e., the "time-domain" or "frequency-domain" approach, may be selected through the variables `sim_flag`. The two routines `getPowderSignalStatic` and `getPowderSpectrumStatic` have similar structures: Both involve a powder loop over the variable `ang_index`, which runs over all `n_angles` of crystallite orientations. For each orientation Ω_{ML} (represented by `ML` in the code), each tensor component $[A_{20}^{\Lambda}]^L$ is calculated using the routine `L2tensorToLabm0`, given in Part II. Then, the sum of Hamiltonians for all anisotropic interactions (`Haniso`) is constructed. Next, the total Hamiltonian is formed, containing both the isotropic and anisotropic contributions.

Finally, the relevant data are provided as input to the selected spin dynamics calculation routine. This is the essential difference between the two routines `getPowderSignalStatic` and `getPowderSignalStatic`; the former calls the subroutine `getSignalStatic`, which calculates the time-domain signal for each orientation, while the routine `getPowderSpectrumStatic` uses the frequency-domain routine `getSpectrumStatic`. The source code of the spin dynamics routines was explained in detail in Part II.

After each spin dynamics calculation is completed for each molecular orientation within the loop, the estimate of the powder-averaged NMR signal is updated. For example, the frequency-domain routine returns the arrays containing the frequencies $\omega_{uv}(\Omega_{ML})/2\pi$ (stored in the array `freq_list`) and amplitudes $a_{uv}(\Omega_{ML})$ (`amp_list`). Then, the powder-averaged amplitudes, stored in the array `ampPowder`, are updated by adding $a_{uv}(\Omega_{ML})$ multiplied by its weight $w(\Omega_{ML})$ (`weight_list[angle_index]`) to the appropriate "bin" of the amplitude histogram representing the NMR spectrum (see Part II for a discussion about this mapping). The procedure for updating the powder-averaged time-domain signal is analogous, with the distinction that the loop counter in the routine `getPowderSignalStatic` instead runs over all points of the time-domain signal.

Postprocessing. After exiting the spin dynamics calculations, Lorentzian line broadening is applied to the spectrum (in this case corresponding to 500 Hz full width at half height broadening). Finally, the powder-averaged data are stored on disk. The simulated spectrum, shown in Fig. 11, is in reasonable agreement with the experimental one. The largest discrepancies are in the features of the peak originating from the methylene site, which are presumably due to insufficient proton decoupling during the experiment.

By changing the input parameters, the program may also be used for simulating various limiting spin systems, such as the powder pattern generated from a single CSA interaction as well as the spectrum from two dipolar-coupled spins-1/2 in the absence of CSA. "Direct method" simulations of the two-spin system under MAS conditions may also be carried out using the routine `getSignalPeriodic` (given in Part II), without extensive modification of the program. Further extensions to encompass simulations of larger spin systems will require the construction of additional spatial tensors and larger matrices for the spin operators [and is readily carried out using the matrix routines of the GAMMA library (*46, 47*)].

EXAMPLE 2: HETERONUCLEAR SPIN PAIR UNDER MAS

In this example, we consider a heteronuclear IS spin system, where S═^{13}C and I═^{1}H. Assume that the *S-spin NMR spectrum* is to be calculated under MAS conditions, i.e., the sideband pattern generated from the combined effects of the S-spin chemical shift interaction and the heteronuclear dipolar coupling. In this case, it is assumed that the ^{13}C spin is close in space only to *one* proton, so that additional ^{13}C–^{1}H couplings need not be considered. Further, we assume that the ^{1}H is not coupled to other protons in its surroundings. This approximation rarely holds in organic solids, as the protons are typically tightly coupled to each other. However, by applying homonuclear proton decoupling pulse sequence (*1, 2, 4*) during the acquisition of the ^{13}C signal one may reduce the problem to an isolated heteronuclear ^{13}C–^{1}H spin pair. In the following, we calculate the one-dimensional NMR spectrum assuming that detectable ^{13}C transverse magnetization is first generated by a $(\pi/2)_y$ pulse. This correspond to $\hat{\rho}(0) = \hat{\mathbf{S}}_x$ at the start of signal acquisition, employing the observable $\hat{\mathbf{Q}} = \hat{\mathbf{S}}^+$.

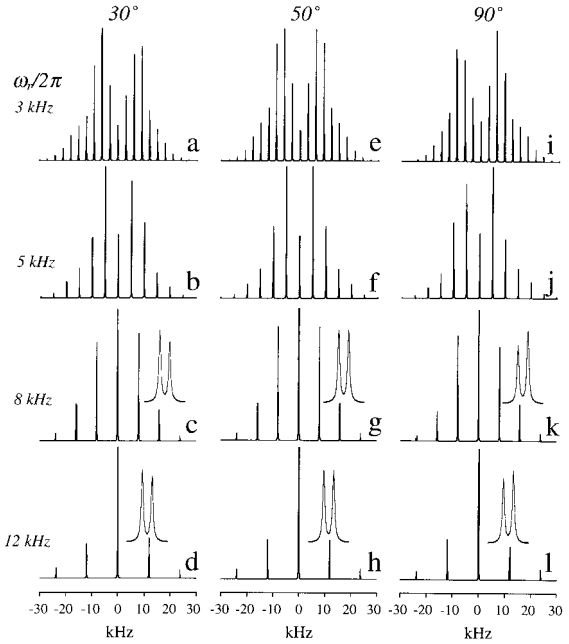

Figure 11 S-spin MAS sideband patterns generated from the combined effects of its CSA and the heteronuclear dipolar interaction to another spin I. The spectra were calculated for different spinning frequencies and relative tensor orientations using the program listed in Appendix C. The left, middle, and right panels correspond to the cases $\Omega_{PM}^{S} = \{0, \pi/6, 0\}$, $\{0, (5/18)\pi, 0\}$, and $\{0, \pi/2, 0\}$, respectively. All other parameters (given in the text) are kept fixed. Each inset spectrum is a magnification of the center band in the manifold. The spectrum in (b) is obtained using the exact input parameters in the code of Appendix C. Note how the sideband intensities depend on the relative orientations between the dipolar and CSA tensors and that each sideband is split into a doublet due to the heteronuclear J-coupling. This fact may be used to determine the tensor magnitudes and relative orientations by fitting numerically calculated spectra to experimental results.

Simulations such as the one outlined below are relevant to SLF experiments (*2–4, 11–13*), which may be used to measure the magnitudes and relative orientations between two interactions through their effects on the spectral sideband patterns they generate. Usually, SLF experiments are implemented in a two-dimensional fashion, giving a spectrum with the two interactions (in the present case, IS dipolar couplings and S-spin chemical shifts) separated along each of the spectral dimensions.

In the following, we first discuss the spin Hamiltonian to be used in the numerical simulation and then consider the C program given in Appendix C.

Hamiltonian

When considering all interactions of the spin system, the following Hamiltonians are obtained from Eq. [I-60] and Table I-2:

$$\hat{\mathbf{H}}_{\mathrm{CS}}^{I}(\mathrm{t}) = \left(\omega_{\mathrm{iso}}^{I} + \sqrt{\frac{2}{3}}\, \omega_{\mathrm{CSA}}^{I}(\mathrm{t}) \right) \hat{\mathbf{I}}_z \qquad [86]$$

$$\hat{\mathbf{H}}_{\mathrm{CS}}^{S}(\mathrm{t}) = \left(\omega_{\mathrm{iso}}^{S} + \sqrt{\frac{2}{3}}\, \omega_{\mathrm{CSA}}^{S}(\mathrm{t}) \right) \hat{\mathbf{S}}_z \qquad [87]$$

$$\hat{\mathbf{H}}_{D}^{IS}(t) = \omega_{D}^{IS}(t) \sqrt{\frac{2}{3}}\, \hat{\mathbf{I}}_z \hat{\mathbf{S}}_z \qquad [88]$$

$$\hat{\mathbf{H}}_{J}^{IS} = 2\pi J_{\mathrm{iso}}^{IS} \hat{\mathbf{I}}_z \hat{\mathbf{S}}_z \qquad [89]$$

These expressions are derived as in the previous example. The total Hamiltonian is dynamically inhomogeneous, as it is given by a sum of four commuting terms. This may be verified by explicit evaluation of the various commutators involving pairs of the operators $\hat{\mathbf{I}}_z$, $\hat{\mathbf{S}}_z$, and $\hat{\mathbf{I}}_z\hat{\mathbf{S}}_z$.

Further, only the S-spin chemical shift and heteronuclear coupling Hamiltonians affect the S-spin evolution because the I-spin chemical shift Hamiltonian commutes with all other terms, *as well as* with the observable operator $\hat{\mathbf{S}}^{+}$. This implies that the S-spin NMR spectrum is invariant to the I-spin chemical shift. This may be shown as follows: We express the total Hamiltonian of the heteronuclear spin pair as a sum of two commuting terms:

$$\hat{\mathbf{H}}_{\mathrm{tot}}(t) = \hat{\mathbf{H}}(t) + \hat{\mathbf{H}}_{\mathrm{CS}}^{I}(\mathrm{t}) \qquad [90]$$

where $\hat{\mathbf{H}}(t)$ includes the S-spin chemical shift Hamiltonian, as well as the dipolar and J coupling Hamiltonians:

$$\hat{\mathbf{H}}(t) = \hat{\mathbf{H}}_{\mathrm{CS}}^{S}(t) + \hat{\mathbf{H}}_{D}^{IS}(t) + \hat{\mathbf{H}}_{J}^{IS} \qquad [91]$$

The aim now is to show that the time evolution of the density operator is governed solely by the Hamiltonian in Eq. [91].

The formal expression for the propagator arising from a dynamically inhomogeneous Hamiltonian is given in Eq. [I-141]. Therefore, $\hat{\mathbf{H}}_{\mathrm{tot}}(t)$ generates the following accumulated propagator from the start of signal acquisition ($t_0 = 0$) out to an arbitrary time point t:

$$\hat{\mathbf{U}}_{\mathrm{tot}}(t, 0) = \exp\left\{ -i \int_{0}^{t} dt' \hat{\mathbf{H}}_{\mathrm{tot}}(t') \right\} \qquad [92]$$

This may be factorized as a product of two exponential operators (i.e., propagators)

$$\hat{\mathbf{U}}_{\text{tot}}(t,0) = \exp\left\{-i\int_0^t dt'\hat{\mathbf{H}}(t')\right\}\exp\left\{-i\int_0^t dt'\hat{\mathbf{H}}^I_{\text{CS}}(\text{t})\right\} \quad [93]$$

because $\hat{\mathbf{H}}(t)$ and $\hat{\mathbf{H}}^I_{\text{CS}}(t)$ commute with each other. The time evolution of the density operator is according to Eq. [II-3] given by

$$\hat{\rho}(t) = \hat{\mathbf{U}}_{\text{tot}}(t,0)\hat{\rho}(0)\hat{\mathbf{U}}_{\text{tot}}(t,0)^\dagger \quad [94]$$

$$= \exp\left\{-i\int_0^t dt'\hat{\mathbf{H}}(t')\right\}\exp\left\{-i\int_0^t dt'\hat{\mathbf{H}}^I_{\text{CS}}(t')\right\}\hat{\mathbf{S}}_x \times \exp\left\{-i\int_0^t dt'\hat{\mathbf{H}}^I_{\text{CS}}(t')\right\}^\dagger \exp\left\{-i\int_0^t dt'\hat{\mathbf{H}}(t')\right\}^\dagger \quad [95]$$

Next, we apply a theorem used on page 142 of Part I, stating that if two operators $\hat{\mathbf{A}}$ and $\hat{\mathbf{B}}$ commute, i.e., if $[\hat{\mathbf{A}}, \hat{\mathbf{B}}] = 0$, then their corresponding exponential operators *also* commute: $[\exp\{i\hat{\mathbf{A}}\}, \exp\{i\hat{\mathbf{B}}\}] = 0$ (*49*). This follows because the exponential operator $\exp\{i\hat{\mathbf{A}}\}$ has the same eigenbasis as $\hat{\mathbf{A}}$ and the fact that operators sharing the same eigenbasis commute. In other words, we may deduce the commutation properties of exponential operators by evaluating the commutators of the operators in their exponents. Consequently, because $\hat{\mathbf{H}}^I_{\text{CS}}(t)$ commutes with the initial density operator $\hat{\mathbf{S}}_x$, so does its propagator $\exp\{-i\int_0^t dt'\hat{\mathbf{H}}^I_{\text{CS}}(t')\}$ and we may rearrange Eq. [95] as follows:

$$\hat{\rho}(t) = \exp\left\{-i\int_0^t dt'\hat{\mathbf{H}}(t')\right\}\hat{\mathbf{S}}_x \times \underbrace{\exp\left\{-i\int_0^t dt'\hat{\mathbf{H}}^I_{\text{CS}}(t')\right\}\exp\left\{-i\int_0^t dt'\hat{\mathbf{H}}^I_{\text{CS}}(t')\right\}^\dagger}_{=\hat{1}} \times \exp\left\{-i\int_0^t dt'\hat{\mathbf{H}}(t')\right\}^\dagger \quad [96]$$

However, because the product of an operator with its inverse (see Eq. [I-4]) is equal to the unity operator,

$$\exp\left\{-i\int_0^t dt'\hat{\mathbf{H}}^I_{\text{CS}}(t)\right\}\exp\left\{-i\int_0^t dt'\hat{\mathbf{H}}^I_{\text{CS}}(t)\right\}^\dagger = \hat{1} \quad [97]$$

Equation [96] reduces to

$$\hat{\rho}(t) = \exp\left\{-i\int_0^t dt'\hat{\mathbf{H}}(t')\right\}\hat{\mathbf{S}}_x\exp\left\{-i\int_0^t dt'\hat{\mathbf{H}}(t')\right\}^\dagger \quad [98]$$

which is *independent* of the I-spin chemical shift Hamiltonian $\hat{\mathbf{H}}^I_{\text{CS}}(t)$. This shows that the density operator describing the S-spin dynamics, and hence the S-spin NMR spectrum, is independent of the I-spin chemical shift. Henceforth, we may ignore $\hat{\mathbf{H}}^I_{\text{CS}}(t)$ and focus only on the Hamiltonian in Eq. [91] for the numerical simulation. Also, note that Eq. [96] may *not* be rearranged so as to make the propagator $\exp\{-i\int_0^t dt'\hat{\mathbf{H}}(t')\}$ and its inverse $\exp\{-i\int_0^t dt'\hat{\mathbf{H}}(t')\}^\dagger$ cancel, as these do *not commute* with $\hat{\mathbf{S}}_x$.

Numerical Implementation

The code is given in Appendix C. The spin interaction parameters chosen in this example are typical for an aliphatic CH group in an organic solid at a field of 11.7 T. The variable `n` corresponds to the number of divisions of the rotational period τ_r (defined in Part II) and 987 ZCW orientations (generated over a hemisphere) are employed for the powder averaging. The S-spin spectrum generated from the program with this particular choice of parameters is shown in Fig. 11(b). Note that in the program spin operators of index "1" correspond to those of the S-spin (i.e., ^{13}C).

The "initializations" are performed as in the previous example. Then, the routine `getISPowderRotating` carries out the spin dynamics calculations, finally returning to the *powder average* sideband amplitudes. This is the computer implementation of the frequency-domain simulations discussed in Part II and shown in the flowchart of Fig. II-5, with the distinction that the present calculations in addition include the carousel averaging over γ_{MR}, as discussed above. Instead of calculating the spectral amplitude from each single orientation $a^{(k)}_{uv}(\Omega_{MR})$, the γ_{MR}-averaged sideband amplitude $\langle a^{(k)}_{uv}(\Omega_{MR})\rangle_{\gamma_{MR}}$ is generated directly using Eq. [70].

The routine `getISPowderRotating()` starts by constructing a matrix `w0`, holding the values of the Fourier coefficients $\omega^{(0)}_{uv}$ of the isotropic interactions for all eigenstate indices u and v. This is calculated from Eqs. [I-194] and [II-12] by means of the two

matrices `Iz1_diff` and `Izz12_diff`. The elements of the former comprise differences between matrix elements of the operator $\hat{\mathbf{S}}_z$ such that `Iz1_diff(u, v)` corresponds to $(\hat{\mathbf{S}}_z)_{vv} - (\hat{\mathbf{S}}_z)_{uu}$. The matrix `Izz12_diff` is defined analogously, i.e., the element `Izz12_diff(u, v)` is calculated from $(\hat{\mathbf{I}}_z\hat{\mathbf{S}}_z)_{vv} - (\hat{\mathbf{I}}_z\hat{\mathbf{S}}_z)_{uu}$.

Next, the powder loop over Ω_{MR} follows, comprising the following main operations for each orientation Ω_{MR} (compare with the flowchart of Fig. II-5):

1. The Fourier components $\omega_\Lambda^{(0)}$ of the CSA and dipolar tensors are formed using the routine `getSpatialFourierComponents` given in Part II.
2. The eigenvalue Fourier components $\omega_{uv}^{(m)}$ are calculated (Eq. [I-194]).
3. Each of the the phases $\Phi'_{uv}(t_p, 0)$ (Eq. [37]) are constructed for all time points $t_p = 1, 2, \ldots, n$ by means of the subroutine `getPhase` given in Appendix C.
4. The coefficients $c_{uv}^{(k)}$ (Eq. [36]) are calculated by Fourier transforming the list of exponentials $\exp\{i\Phi'_{uv}(t_p, 0)\}$.
5. The powder-averaged amplitudes are updated by adding the γ-averaged amplitude (Eq. [70]) at the appropriate frequency coordinate, i.e., the frequency of the array `freqPowder` being closest to $\omega_{uv}^{(k)}$, following the procedure outlined in Part II.

As in Example 1, after exiting the routine for generating the spectrum broadening is applied to the resulting spectral peaks. Finally, the spectrum is stored on disk. The current implementation is restricted to simulation of a heteronuclear IS-spin system and various limiting cases thereof, such as the spectrum of a single CSA tensor. However, by changing the spin operators this program may also be used for simulating the spectrum of an isolated quadrupolar nucleus (*27*) as long as second-order quadrupolar effects may be ignored. It is also straightforward to include additional spins to simulate larger heteronuclear systems, for instance, simulations of one S-spin coupled to N_I I-spins, where N_I is kept as an arbitrary input parameter. This requires forming the relevant spin operators and constructing all spatial tensors in an "automated fashion."

SUMMARY

Numerical simulations are important ingredients in the development and analysis of solid-state NMR experiments. In Part I [Ref. (*8*)], we outlined the basic theory for simulating the NMR spectrum from a single molecular orientation. In Part II [Ref. (*9*)], we discussed numerical implementations for carrying out computer simulations. The emphasis was on calculating the time evolution of the spin density matrix during a particular experiment, from which the NMR time-domain signal and spectrum may be obtained. In these articles, we focused on applications to cases where the quantum mechanical description of the experiment involved spin evolution under a Hamiltonian that was either time independent (e.g., the case of a static solid sample) or time periodic and self-commuting (e.g., a heteronuclear spin pair under MAS). Extensions to simulations to more complicated cases (e.g., strongly coupled spin systems under MAS) were briefly outlined in Part II.

In this article, which built directly on Parts I and II, we discussed theoretical and practical aspects of *powder averaging* in solid-state NMR: why the NMR signal is dependent on the spatial molecular orientation, the fundamental equations for calculating powder averages, and how they may be implemented on a computer. We also discussed and classified orientational symmetry of the NMR domain time signal, which is useful in practice to speed up calculations of powder averages. Further, the concept of "carousel symmetry" was demonstrated as a route for analytically calculating the average over one of the orientational variables to further improve the accuracy and speed of the numerical simulation. Finally, to exemplify the philosophy of writing of a computer program, source code for two simulation programs were presented and discussed. They are intended to provide a platform for subsequent modifications to other simulations.

ACKNOWLEDGMENTS

I would like to thank L. Frydman, C.V. Grant, A. Sebald, and S. Vega for helpful comments on the manuscript and M.H. Levitt for many discussions and for providing *Mathematica* routines used for generating some of the figures. A postdoctoral fellowship from The Swedish Foundation for International Cooperation in Research and Higher Education (STINT) is appreciated.

APPENDIX A

Here, we outline the specific implementations of the STEP and ZCW methods used in the code for Examples 1 and 2, respectively. There also exist many other versions of these schemes (*18, 26, 29, 30, 32, 34*).

Figure 7 illustrates graphically some STEP and ZCW sets, with each orientation visualized as a point of the unit sphere (also see Fig. I-4).

STEP Scheme

The STEP method divides the planar region $\{x, y\}$ into a grid of evenly spaced coordinates $\{x_i, y_j\}$. These are mapped onto the sphere, i.e., converted into Euler angles describing an orientation that may be represented on the unit sphere. The mappings used here are

$$\alpha_i^{\text{STEP}} = x_i \qquad [99]$$

and

$$\beta_j^{\text{STEP}} = y_j \qquad [100]$$

Assuming that N^α and N^β samples are used for x and y, respectively, the resulting set comprises $N^\alpha N^\beta$ orientations, each given by

$$\alpha_i^{\text{STEP}} = \frac{2\pi}{N^\alpha a_3}(a_1 i + a_2), \qquad i = 0, 1, 2, \ldots N^\alpha - 1 \qquad [101]$$

$$\beta_j^{\text{STEP}} = \frac{\pi}{2N^\beta b}(2j + 1), \qquad j = 0, 1, 2, \ldots N^\beta - 1 \qquad [102]$$

where the numbers $\{a_l\}$ are components of a vector $\vec{\mathbf{a}}$, which depends on the integration range and how the planar range is divided. There are many possibilities (*34*). The sampling of the grid points $\{x_i, y_j\}$ employed here corresponds to

$$\vec{\mathbf{a}} = \begin{cases} (1, 0, 1) & \text{for full sphere and hemisphere} \\ (2, 1, 8) & \text{for octant} \end{cases} \qquad [103]$$

Likewise, the number b depends on the integration range as follows:

$$b = \begin{cases} 1 & \text{for full sphere} \\ 2 & \text{for hemisphere and octant} \end{cases} \qquad [104]$$

The weight associated with the angle $\{\alpha_i, \beta_j\}$ is given by

$$w_j^{\text{STEP}} = N_{\text{STEP}} \sin\{\beta_j\} \qquad [105]$$

with the normalization constant

$$N_{\text{STEP}} = \left(N^\alpha \sum_{j=0}^{N^\beta - 1} \sin\{\beta_j\} \right)^{-1} \qquad [106]$$

When using these equations to generate the STEP sets, it is recommended to use equal increments (i.e., "steps") in the two integration variables α and β. This implies using $N^\alpha = N^\beta$ for the octant sets, $N^\alpha = 4N^\beta$ for the hemispheric sets, and $N^\alpha = 2N^\beta$ for the full sphere.

ZCW Scheme

It is out of the scope of this article to justify the generic equations for the ZCW sets. Detailed explanations may be found in Refs. (*28–30, 32*). The ZCW partitions are generated from numbers F_M of the Fibonacci series (*15*). These are given by the recursion

$$F_M = F_{M-1} + F_{M-2}, \qquad M = 0, 1, 2, \ldots \qquad [107]$$

with $F_0 = 8$ and $F_1 = 13$. For a given integer M, the corresponding ZCW set comprises

$$N_M = F_{M+2} \qquad [108]$$

samples over the planar range $\{x, y\}$. Next, the mappings

$$\alpha_i^{\text{ZCW}} = x_i \qquad [109]$$

and

$$\beta_j^{\text{ZCW}} = \arccos\{y_j\} \qquad [110]$$

are used, giving the Euler angles

$$\alpha_j^{\text{ZCW}} = \frac{2\pi}{c_3} \bmod\{jF_M/N_M, 1\}, \qquad j = 0, 1, 2, \ldots, N_M - 1 \qquad [111]$$

$$\beta_j^{\text{ZCW}} = \arccos[c_1(c_2 \bmod\{j/N_M, 1\} - 1)], \qquad j = 0, 1, 2, \ldots, N_M - 1 \qquad [112]$$

As for the STEP implementation, the numbers $\{c_l\}$ are components of a vector $\vec{\mathbf{c}}$, given by

$$\vec{\mathbf{c}} = \begin{cases} (1, 2, 1) & \text{for full sphere} \\ (-1, 1, 1) & \text{for hemisphere} \\ (2, 1, 8) & \text{for octant} \end{cases} \qquad [113]$$

The weights are equal for all angles in a ZCW set; they are calculated as the inverse of the number of orientations:

$$w_j^{\mathrm{ZCW}} = N_M^{-1} \qquad [114]$$

Implementations in C

The C code implementations of the equations above for producing sets of orientations for the STEP (`genSTEP`) and ZCW (`genZCW`) methods are given below. Each routine may generate orientations according to the choice of the input string `sphereType`, which is to be specified as "`full`," "`hemi`," or "`oct`" for integration over the full sphere, the upper hemisphere, and octant, respectively. The routine `genZCW` returns a list of Euler angles (of data type `euler`, defined in Part II) and their corresponding weights, stored in `angle_list` and `weight_list`, respectively. The routine `getNumberZCW` calculates the number of orientations from the value of the input parameter `M`. The routine for generating the STEP angle sets is implemented similarly.

The ZCW routines are used in the programs given in Appendices B and C.

```
//*********  ZCW ROUTINES  ***********
int getNumberZCW(int M)
//returns the number of ZCW angles for the given integer M=2,3,4,. . .
{
 int j,gM=5,gMminus1=3;
 int sum=5;
 for(j=0;j<=M;j++) {
  sum=(gM+gMminus1);
  gMminus1=gM;
  gM=sum;
 }
 return sum;
}

void genZCW(euler *angle_list,double *weight_list,int M,char *sphereType)
//constructs the set containing getNumberZCW(M) ZCW orientations {angle_j,weight_j}
//for the given choice of symmetry: sphereType is either of:{ full, hemi, oct }
{
 double c[3];                  //define the vector 'c', and assign its elements
 if (!strcmp(sphereType,"full")) {c[0]=1.;c[1]=2.;c[2]=1.;};
 if (!strcmp(sphereType,"hemi")) {c[0]=-1.;c[1]=1;c[2]=1.;};
 if (!strcmp(sphereType,"oct")) {c[0]=-1.;c[1]=1;c[2]=4.;};

 int N=getNumberZCW(M);       //total number of angles
 int g2=getNumberZCW(M-2);
 for(int m=0;m<=(N-1);m++ ) {
  angle_list[m].beta=acos( c[0]*(c[1]*fmod(m/double(N),1.)-1.) );
  angle_list[m].alpha=2.*Pi*( fmod( (m*g2/double(N)),1.) )/c[2];
  angle_list[m].gamma=0.;
  weight_list[m]=1./double(N);
 }
}

//*********  STEP ROUTINE  ***********
void genSTEP(euler *angle_list,double *weight_list,int N_alpha,int N_beta,char *sphereType)
//constructs the set containing (N_alpha*N_beta) STEP orientations {angle_j,weight_j} for the given
//choice of symmetry: sphereType is either of:{ full, hemi, oct }
{
 double b,a[3],norm_step=0.;            //define b and the elements of the vector 'a'
 if (!strcmp(sphereType,"full")) {a[0]=1.;a[1]=0.;a[2]=1.;b=1.;};
 if (!strcmp(sphereType,"hemi")) {a[0]=1.;a[1]=0.;a[2]=1.;b=2.;};
 if (!strcmp(sphereType,"oct")) {a[0]=2.;a[1]=1.;a[2]=8.;b=2.;};

 double inc_alpha=2.*Pi/( N_alpha*a[2] ); //calculate incrementation in alpha angle
 double inc_beta=Pi/( 2.*N_beta*b );      //calculate incrementation in beta angle
 //calculate the normalization factor
 for(int j=0;j<N_beta;j++) norm_step += sin( inc_beta*( 2.*j+1.) );
 norm_step=( 1./(N_alpha*norm_step) );
 //assign the angles and weights
 for(int j=0;j<N_beta;j++ ) {
  for(int i=0;i<N_alpha;i++ ){
   angle_list[j*N_alpha+i].alpha=inc_alpha*( a[1]+ i*a[0] );
   angle_list[j*N_alpha+i].beta=inc_beta*( 2.*j+1.);
   angle_list[j*N_alpha+i].gamma=0.;
   weight_list[j*N_alpha+i]=norm_step*sin( inc_beta*( 2.*j+1.) );
  }
 }
}
```

APPENDIX B

```
#include <math.h>          //standard C-library of mathematical functions
#include <complex.h>       //from GAMMA library
#include <matrix.h>        //from GAMMA library
#include <aux.h>           //general auxiliary routines
#include <spinOperators.h> //for constructing the spin operators
#include <spinDynamics.h>  //spin dynamics calculation routines
#include <powderSchemes.h> //generate orientations for powder averaging

// STATIC.cc
// Simulation of the powder spectrum from a homonuclear spin-pair, experiencing a time-independent
// Hamiltonian. The calculation may be carried out both in time- and frequency-domains.

struct spinPars{ //data type containing all spin parameters
 matrix CSA_1_M;
 matrix CSA_2_M;
 matrix D_M;
};

void getPowderSpectrumStatic( const spinPars &data,const matrix &Hiso, const matrix &rho0,
                              const matrix &Q, complex *ampPowder,int n_points, double sw,
                              euler *ang_list,double *weight_list,int n_angles)
// Simulation of powder spectrum in the frequency domain
// INPUT: The spin parameters, isotropic part of the Hamiltonian (Hiso), initial density operator (rho0),
//        observable (Q), spectral window (sw) and the list of powder angles and weights
// OUTPUT: The spectrum containing n-points of the powder averaged amplitudes (ampPowder)
{
 int m,n_transitions,n_states=rho0.rows();
 double freq,*freq_list,res=sw/double(n_points);
 double wCSA_1,wCSA_2,wD;
 complex *amp_list;
 euler ML;                                      //the M=>L transformation ("powder") angle
 matrix Haniso;                                 //the total Hamiltonian for the anisotropic interactions
 freq_list=d_array1(n_states*n_states);         //lists of frequencies for one molecular orientation
 amp_list=c_array1(n_states*n_states);          //list of amplitudes for one molecular orientation

 for(int angle_index=0;angle_index<n_angles;angle_index++) { //powder loop

  ML=ang_list[angle_index];                    //get M=>L transformation angle
  wCSA_1=L2tensorToLab_m0( data.CSA_1_M,ML );  //calculate the Lab-frame m=0 component of CSA tensor #1
  wCSA_2=L2tensorToLab_m0( data.CSA_2_M,ML );
  wD=L2tensorToLab_m0( data.D_M,ML );          //calculate the Lab-frame m=0 component of DD-tensor
  //construct the Hamiltonian for the anisotropic parts for the current orientation
  Haniso=(wCSA_1*sqrt(2./3.)*Iz1)+(wCSA_2*sqrt(2./3.)*Iz2)+(wD*T12_20);
  //call the spin dynamics calculation routine
  getSpectrumStatic( Hiso+Haniso,rho0,Q,freq_list,amp_list,n_transitions );
  for(int j=0;j<n_transitions;j++) {
   freq=freq_list[j];
   m=int(freq/res + (sign(freq)*0.5) )+n_points/2-1;
   //update the powder averaged amplitude, provided the 'bin_index' m is in the proper range
    if ((m>=0 && (m<n_points)) ampPowder[m]+=weight_list[angle_index]*amp_list[j];
  }

 }                              //closes powder loop
 del_d_array1(freq_list);       //free allocated memory
 del_c_array1(amp_list);
}

void getPowderSignalStatic( const spinPars &data,const matrix &Hiso, const matrix &rho0,
                            const matrix &Q, complex* signalPowder,int n_points,double sw,
                            euler *ang_list,double *weight_list,int n_angles)
// Simulation of powder signal by the direct method
// INPUT: The spin parameters, isotropic part of the Hamiltonian (Hiso), initial density operator (rho0),
//        observable (Q), spectral window (sw) and the list of powder angles and weights
// OUTPUT: n_points of the powder averaged time signal (signalPowder)
{
 double wCSA_1,wCSA_2,wD;
 complex *signal_list;                          //the signal from one molecular orientation
 euler ML;                                      //M=>R transformation angle
 matrix Haniso;                                 //the Hamiltonian for sum of anisotropies
 int n_states=rho0.rows();
 signal_list=c_array1(n_points);                //number of eigenstates

 for(int angle_index=0;angle_index<n_angles;angle_index++) { //powder loop

  ML=ang_list[angle_index];                    //get M=>L transformation angle
  wCSA_1=L2tensorToLab_m0( data.CSA_1_M,ML );  //calculate the Lab-frame m=0 component for CSA_1
  wCSA_2=L2tensorToLab_m0( data.CSA_2_M,ML );  //calculate the Lab-frame m=0 component for CSA_2
```

```
  wD=L2tensorToLab_m0( data.D_M,ML );  //calculate the Lab-frame m=0 component for dipolar coupling
  Haniso=(wCSA_1*sqrt(2./3.)*Iz1)+(wCSA_2*sqrt(2./3.)*Iz2)+(wD*T12_20);
  //call the spin dynamics calculation routine and update the powder average estimate of signal
  getSignalStatic( Hiso+Haniso,rho0,Q,signal_list,n_points,(1./sw) );
  for(int j=0;j<n_points;j++) signalPowder[j] += weight_list[angle_index]*signal_list[j];

 }
 del_c_array1(signal_list);  //free allocated memory
}

main()
{
//*****************************************************************
//*********                 INITIALIZATION                **********
//*****************************************************************
 int ZCWLevel,n_points,sim_flag;
 char out_file[100];
 double sw,width;
 double CSiso_1,CSiso_2,CSAaniso_1,CSAeta_1,CSAaniso_2,CSAeta_2,Jiso,b_12; //spin interaction parameters
 euler PM_CSA_1,PM_CSA_2,PM_D;
 matrix Q,rho0;

 //****************  INPUT PARAMETERS  *******************
 CSiso_1=HzToRads(-6204.);CSiso_2=HzToRads(7204.);Jiso=HzToRads(53.); //parameters for isotropic parts
 CSAaniso_1=HzToRads(7500.);CSAeta_1=0.88;           //CSA parameters of spin 1 (carboxyl)
 CSAaniso_2=HzToRads(1956.);CSAeta_2=0.98;           //CSA parameters of spin 2 (methylene)
 b_12=HzToRads(-2135.);                              //dipolar coupling
 setEuler( PM_CSA_1, -0.7,88.5,52.5 );               //CSA_1 P=>M angles in degrees
 setEuler( PM_CSA_2, 99.4,146.0,138.9 );             //CSA_2 P=>M angles in degrees
 setEuler( PM_D, 0.,0.,0. );                         //dipolar P=>M angles
 SOsetup(2);                                         //construct spin operators for two spins
 Q=Ip1+Ip2;                                          //observable operator
 rho0=Ix1+Ix2;                                       //initial density operator
 sw=40000.;                                          //spectral window (Hz)
 n_points=8192;                                      //number of points in the spectrum/signal
 width=500.;                                         //width (in Hz) of the line-broadening used
 ZCWlevel=12;                                        //choice of ZCW set
 char sphereSymm[10]="hemi";                         //orientations generated over hemi sphere
 sim_flag=0;                                         //sim_flag=0 => time-domain; sim_flag=1 => freq-domain
 cout << '\n' << "out_file=";cin >> out_file;        //get the output file from the user

 spinPars data;
 matrix CSA_1_M,CSA_2_M,D_M,Hiso;
 complex *ampPowder,*signalPowder;
 double *freqPowder,*ampPowder_re,*weight_list;
 euler *ang_list;                                   //list of weighted euler angles
 int n_angles=getNumberZCW(ZCWlevel);               //calculate the number of ZCW orientations
 cout << '\n' << "n_angles=" << n_angles << '\n';
 ang_list=eul_array1(n_angles);                     //allocate memory for lists of powder angles
 weight_list=d_array1(n_angles);                    //allocate memory for lists of weights
 genZCW(ang_list,weight_list,ZCWlevel,sphereSymm);  //generate powder angles
 freqPowder=d_array1(n_points):                     //allocate memory for spectral frequency coordinates
 ampPowder=c_array1(n_points);                      //allocate memory for spectral amplitudes
 signalPowder=c_array1(n_points);                   //allocate memory for values of powder averge signal
 ampPowder_re=d_array1(n_points);
 for(int j=0;j<n_points;j++) freqPowder[j]=-sw/2+(j+1)+sw/double(n_points); //generate the frequencies

 //construct the tensors in their PAS, and transform P=>M
 CSA_1_M=L2tensor_transform( L2TensorSetup( CSAaniso_1, CSAeta_1 ),PM_CSA_1 );
 CSA_2_M=L2tensor_transform( L2TensorSetup( CSAaniso_2, CSAeta_2 ),PM_CSA_2 );
 D_M=L2tensor_transform( L2TensorSetup( 2.*b_12, 0. ),PM_D );
 Hiso=(CSiso_1*Iz1)+(CSiso_2*Iz2)+(Jiso*I12);          //Hamiltonian for all isotropic interactions
 data.CSA_1_M=CSA_1_M;data.CSA_2_M=CSA_2_M;data.D_M=D_M;  //store the tensors in the variable 'data'

//*************************************************************
//***   DO SPIN DYNAMICS SIMULATION & APPLY LINEBROADENING   ****
//*************************************************************
 if (sim_flag) {                                    //choose between frequency and time-domain simulation
  getPowderSpectrumStatic(data,Hiso,rho0,Q,ampPowder,n_points,sw,ang_list,weight_list,n_angles);
  lineBroaden( ampPowder,width,n_points,1./sw,0);
 }
 else {
  getPowderSignalStatic(data,Hiso,rho0,Q,signalPowder,n_points,sw,ang_list,weight_list,n_angles);
  apodize(signalPowder,width,n_points,1.0/sw,0);          //apply decay function
  FFT(signalPowder,ampPowder,n_points,1);                 //transform the frequency domain
 }
 re_list( ampPowder,ampPowder_re,n_points );              //take real part of spectrum
 save_data( freqPowder,ampPowder_re,n_points,out_file );  //store data on disk
 cout << '\n';
}
```

APPENDIX C

```
#include <math.h>               //standard C-library of mathematical functions
#include <complex.h>            //from GAMMA
#include <matrix.h>             //from GAMMA
#include <aux.h>                //general auxiliary routines
#include <spinOperators.h>      //for constructing the spin operators
#include <powderSchemes.h>      //generate orientations for powder averaging

// CSA+IS.cc
// Simulation of the powder S-spin spectrum from an IS system under MAS

struct spinPars{ //data type containing all spin parameters
 double CSiso_1;
 double CSiso_2;
 double Jiso;
 matrix CSA_1_M;
 matrix CSA_2_M;
 matrix D_M;
};

double getPhase( const matrix &FourComp, double wr, double t )
// OUTPUT: the phase accumulated from t=0 to t=t of an interaction with Fourier components as input in FourComp.
//         The symmetry of the w^(m) components with respect to sign reversal of m is used
//         to reduce the summation range to m=1,2.
{
 double sum=0.;
 for(int m=1;m<=2;m++) sum += ( Im(FourComp.get(0,2-m)*(exp(I*m*wr*tb)-1.))/(m*wr) );
 return 2.*sum;
}

void getISPowderRotating( spinPars data,const matrix &rho0, const matrix &Q,double wr,int n,complex *ampPowder,
                          int n_points,double sw, euler *ang_list,double *weight_list,int n_angles)
// simulation of S-spin powder spectrum in the frequency domain under MAS
// INPUT: the spin parameters, initial density operator (rho0), observable (Q), spectral window (sw),
//        the parameter n, and a list of powder angles and weights
// OUTPUT: the powder averaged amplitudes (ampPowder) of size n_points
{
 int u,v,k,bin_index,n_states=rho0.rows();
 double freq,w0_uv,tau=2*Pi/(wr*double(n));
 double res=sw/double(n_points);                                  //frequency resolution
 complex phase,a_uv,*amp_list,*exp_phase_uv,*c_uv;
 euler MR;                                                        //M=>R transformation angle
 matrix w_uv,CSA_M,CSA_FourierComponents,D_M,D_FourierComponents;
 amp_list=c_array1(n_states*n_states);                            //amplitudes from one molecular orientation
 exp_phase_uv=c_array1(n);                                        //exponents of the phases
 c_uv=c_array1(n);                                                //array of c_uv for one molecular orientation

 matrix Iz1_diff(n_states,n_states);
 matrix Izz12_diff(n_states,n_states);
 matrix w0(n_states,n_states);                                   //matrix of isotropic m=0 eigenvalues
 for(u=0;u<n_states;u++) {                                       //u loop over states
  for(v=0;v<n_states;v++) {                                      //v loop over states
   //get the difference between the matrix elements of the spin operators,
   //and construct the m=0 Hamiltonian eigenvalues
   Iz1_diff(u,v)=(Iz1.get(v,v)-Iz1.get(u,u) );
   Izz12_diff(u,v)=(Izz12.get(v,v)-Izz12.get(u,u) );
   w0(u,v)=( Iz1_diff(u,v)*data.CSiso_1 + Izz12_diff(u,v)*data.Jiso );
  }
 }

 for(int angle_index=0;angle_index<n_angles;angle_index++) {    //powder loop
   MR=ang_list[angle_index];                                     //get the current M=>R angle
   CSA_FourierComponents=getSpatialFourierComponents( data.CSA_1_M,MR,MagAng );
   D_FourierComponents=getSpatialFourierComponents( data.D_M,MR,MagAng );
   for(u=0;u<n_states;u++) {                                     //u loop over states
    for(v=0;v<n_states;v++) {                                    //v loop over states

     a_uv=rho0.get(u,v)*Q.get(v,u);                              //calculate the amplitude a_uv
     if ( norm(a_uv)>1e-8 ) {                                    //only continue from here if |a|>10e-8
      //calculate the isotropic and anisotropic parts of the eigenvalue difference for states u & v
      w_uv=( Iz1_diff(u,v)*sqrt(2./3.)*CSA_FourierComponents+
             sqrt(2./3.)*Izz12_diff(u,v)*D_FourierComponents );
      w0_uv=Re(w0.get(u,v));
      for(k=0;k<n;k++) exp_phase_uv[k]=exp( I*getPhase( w_uv,wr,(k+1)*tau) );
       FFT( exp_phase_uv,c_uv,n,0 );                             //calculate c^ (k)_uv by an FFT
       for(k=0;k<n;k++){
        freq=( w0_uv+(wr*double(k-n/2+1)) )/(2.*Pi);             //get current frequency position
        bin_index=int( freq/res+(sign(freq)*0.5) )+n_points/2-1; //update powder amplitudes
        if ((bin_index>=0) && (bin_index<n_points))
```

```
         ampPowder[bin_index] += (weight_list[angle_index]*a_uv*square_norm(c_uv[k]));
       }
      }
    }  //closes v loop over states
  }   //closes u loop over states

 }  //closes powder loop
 del_c_array1(amp_list);  //free allocated memory
 del_c_array1(exp_phase_uv);
 del_c_array1(c_uv);
}

main()
{
//*****************************************************************
//*********                  INITIALIZATION               **********
//*****************************************************************
 int n,ZCWlevel,n_points;
 double sw,width,wr,CSiso_1,CSiso_2,CSAaniso_1,CSAeta_1,Jiso,b_12;
 char out_file[100];
 euler PM_CSA_1,PM_D;
 matrix Q,rho0;

 //********  INPUT PARAMETERS  *******
 CSiso_1=HzToRads(0.0);Jiso=HzToRads(200.);          //S-spin is labeled 1
 CSAaniso_1=HzToRads(-1900.);CSAeta_1=0.4;            //I-spin is labeled 2
 setEuler( PM_CSA_1,0.,30.,0. );
 b_12=HzToRads(-21000.);setEuler( PM_D,0.,0.,0. );
 wr=HzToRads(5000.);                                  //rotational frequency
 n=32;                                                //divide rotational period into n=32 segments
 S0setup(2);                                          //construct spin operators for two spins
 Q=Ip1;                                               //observable operator; only observe S-spin
 rho0=Ix1;                                            //initial density operator
 sw=80000.;                                           //spectral window
 n_points=16384;                                      //total number of spectral points
 width=50.;                                           //broadening (FWHH in Hz)
 ZCWlevel=10;                                         //choice of ZCW set
 char sphereSymm[10]="hemi";
 cout << '\n' << "out_file=";cin.getline(out_file,100);
 //*****************************************************************

 spinPars data;
 matrix CSA_M,D_M;
 complex *ampPowder;
 double *freqPowder,*ampPowder_re,*weight_list;
 euler *ang_list;

 int n_angles=getNumberZCW(ZCWlevel);                  //construct powder angles & allocate memory
 cout << '\n' << "n_angles=" << n_angles << '\n';
 ang_list=eul_array1(n_angles);                        //allocate memory for lists of powder angles
 weight_list=d_array1(n_angles);                       //allocate memory for lists of weights
 genZCW(ang_list,weight_list,ZCWlevel,sphereSymm);     //generate powder angles
 freqPowder=d_array1(n_points);                        //allocate memory for spectral frequency coordinates
 ampPowder=c_array1(n_points);                         //allocate memory for spectral amplitudes
 ampPowder_re=d_array1(n_points);
 for(int j=0;j<n_points;j++) freqPowder[j]=-sw/2+(j+1)*sw/double(n_points); //generate frequency array
 //construct the tensors in their PAS, and transform P=>M
 CSA_M=L2tensor_transform( L2TensorSetup( CSAaniso_1, CSAeta_1 ),PM_CSA_1 );
 D_M=L2tensor_transform( L2TensorSetup( 2.*b_12, 0. ),PM_D );
 data.CSiso_1=CSiso_1;data.Jiso=Jiso;                  //store all interactions in 'data'
 data.CSA_1_M=CSA_M;data.D_M=D_M;

 //****  CALL ROUTINE FOR SPIN DYNAMICS CALCULATION  *****
 getISPowderRotating( data,rho0,Q,wr,n,ampPowder,n_points,sw,ang_list,weight_list,n_angles);

 //****  POST PROCESSING: LINE BROADENING  *****
 lineBroaden( ampPowder,width,n_points,1./sw,0);
 re_list( ampPowder,ampPowder_re,n_points );           //take real part of amplitudes
 save_data( freqPowder,ampPowder_re,n_points,out_file );  //store spectrum on disk
 cout << '\n';
}
```

REFERENCES

1. Haeberlen U. High Resolution NMR in Solids. Selective Averaging. New York: Academic Press; 1976.
2. Munowitz M. Coherence and NMR. New York: Wiley; 1988.
3. Ernst RR, Bodenhausen G, Wokaun A. Principles of Nuclear Magnetic Resonance in One and Two Dimensions. Oxford, UK: Clarendon Press; 1987.
4. Schmidt-Rohr K, Spiess HW. Multidimensional Solid-State NMR and Polymers. New York: Academic; 1994.
5. Mehring M, Weberuß VA. Object-Oriented Magnetic Resonance. Classes and Objects, Calculations and Computations. London: Academic Press; 2001.
6. Andrew ER, Bradbury A, Eades RG. Removal of dipolar broadening of nuclear magnetic resonance spectra of solids by specimen rotation, Nature 1959; 183:1802–1803.
7. Lowe IJ. Free induction decays of rotating solids. Phys Rev Lett 1959; 2:285–287.
8. Edén M. Computer simulations in solid-state NMR: I. Spin dynamics theory. Concepts Magn Reson Part A 2003; 17A:117–154.
9. Edén M. Computer simulations in solid-state NMR: II. Implementations for static and rotating samples. Concepts Magn Reson 2003; 18A:1–23.
10. Maricq MM, Waugh JS. NMR in rotating solids. J Chem Phys 1979; 70:3300–3316.
11. Hester RK, Cross VR, Ackerman JL, Waugh JS. Structure determination of dipolar coupling measurements of dilute nuclear spins in a polycrystalline solid. J Chem Phys 1975; 63:3606.
12. Stoll ME, Vega AJ, Vaughan RW. Heteronuclear dipolar modulated chemical shift spectra for geometrical information in polycrystalline solids. J Chem Phys 1976; 65:4093–4098.
13. Munowitz MG, Griffin RG. Two-dimensional nuclear magnetic resonance in rotating solids: An analysis of line shapes in chemical shift-dipolar spectra. J Chem Phys 1982; 76:2848–2858.
14. http://www.fos.su/se/physical/mattias.
15. Abramowitz M, Stegun IA, editors. Handbook of Mathematical Functions with Formulas, Graphs and Mathematical Tables. New York: Dover; 1972.
16. Alderman DW, Solum MS, Grant DM. Methods for analyzing spectroscopic line shapes. NMR solid powder patterns. J Chem Phys 1986; 84:3717–3725.
17. Wang D, Hanson GR. A new method for simulating randomly oriented powder spectra in magnetic resonance: The Sydney Opera House (SOPHE) method. J Magn Reson A 1995; 117:1–8.
18. Bak M, Nielsen NC. REPULSION—A novel approach to efficient powder averaging in solid-state NMR. J Magn Reson 1997; 125:132–139.
19. Charpentier T, Fermon C, Virlet J. Efficient time propagation technique for MAS NMR simulation: Application to quadrupolar nuclei. J Magn Reson 1998; 132:181–190.
20. Charpentier T, Fermon C, Virlet J. Numerical and theoretical analysis of multiquantum magic-angle-spinning experiments. J Chem Phys 1998; 109:3116–3130.
21. Levitt MH, Edén M. Numerical simulation of periodic NMR problems: Fast calculation of carousel averages. Mol Phys 1998; 95:879–890.
22. Hohwy M, Bildsøe H, Jakobsen HJ, Nielsen NC. Efficient spectral simulations in NMR of rotating solids. The γ-COMPUTE algorithm. J Magn Reson 1999; 136:6–14.
23. Hodgkinson P, Emsley L. Numerical simulation of solid-state NMR experiments. Prog NMR Spectrosc 2000; 36:201–239.
24. Levitt MH. Why do spinning sidebands have the same phase? J Magn Reson 1989; 82:427–433.
25. Antzutkin ON, Shekar SC, Levitt MH. Two-dimensional sideband separation in magic-angle-spinning NMR. J Magn Reson A 1995; 115:7–19.
26. Edén M, Levitt MH. Computation of orientational averages in solid-state NMR by Gaussian spherical quadrature. J Magn Reson 1998; 132:220–239.
27. Vega AJ. In: Encyclopedia of NMR. New York: Wiley; 1995.
28. Zaremba SK, Good lattice points, discrepancy, and numerical integration. Ann Mat Pura Appl 1966; 4:73:293–317.
29. Conroy H. Molecular Schrödinger equation. VIII. A new method for the evaluation of multidimensional integrals. J Chem Phys 1967; 47:5307–5318.
30. Cheng VB, Suzukawa HH, Wolfsberg M. Investigations of a nonrandom numerical method for multidimensional integration. J Chem Phys 1973; 59:3992–3999.
31. Mombourquette MJ, Weil JA. Simulation of magnetic resonance powder spectra. J Magn Reson 1992; 99:37–44.
32. Koons JM, Hughes E, Cho HM, Ellis PD. Extracting multitensor solid-state NMR parameters from lineshapes. J Magn Reson A 1995; 114:12–23.
33. Varner SJ, Vold RL, Hoatson GL. An efficient method for calculating powder patterns. J Magn Reson A 1996; 123:72–80.
34. Ponti A. Simulation of magnetic resonance static powder lineshapes: A quantitative assessment of spherical codes. J Magn Reson 1999; 138:288–297.
35. Ponti A. Simulation of one-dimensional magnetic resonance powder lineshapes reduced to area computation. Chem Phys Lett 1999; 302:224–230.
36. http://www.csrsrc.mi.cnr.it/~ponti.
37. Simpson program. University of Aarhus, Denmark. http://nmr.imsb.au.dk.
38. http://www.soton.ac.uk/~mhl.
39. Olivieri AC. Rigorous statistical analysis of errors in chemical-shift-tensor components from spinning sidebands in solid-state NMR. J Magn Reson 1996; 123:207–210.
40. Hodgkinson P, Emsley L. The reliability of the deter-

mination of tensor parameters by solid-state NMR. J Chem Phys 1997; 107:4808–4816.
41. Lebedev VI. Values of the nodes and weights of ninth to seventeenth order Gauss–Markov quadrature formulae invariant under the octahedron group with inversion. Z Vychisl Mat Fiz 1975; 15:48–54.
42. Lebedev VI, Skorokhodov AL. Quadrature formulas of orders 41, 47 and 53 for the sphere. Sov Phys Dokl 1992; 45:587–592.
43. Schildt H. C: The complete reference. Berkeley, CA: Osborne McGraw-Hill; 1995.
44. Press WH, Flannery BP, Teukolsky SA, Vetterling VT. Numerical recipies in C. The Art of Scientific Computing. Cambridge, UK: Cambridge University Press; 1986.
45. Stroustrup B. The C++ Programming Language. New York: Addison-Wesley; 1991.
46. Smith SA, Levante TO, Meier BH, Ernst RR. Computer simulations in magnetic resonance. An object-oriented programming approach. J Magn Reson A 1994; 106: 75–105.
47. GAMMA program. Florida State University, Tallahassee, FL. http://gamma.magnet.fsu.edu.
48. Brinkmann A, Schmedt auf der Günne J, Levitt MH. Homonuclear zero-quantum recoupling in fast magic-angle-spinning nuclear magnetic resonance. J Magn Reson 2002; 156:79–96.
49. Sakurai JJ. Modern Quantum Mechanics. New York: Addison-Wesley; 1994.

BIOGRAPHY

Mattias Edén was born in 1971 in Stockholm, Sweden. He received his B.S. in chemistry from Stockholm University in 1994 and continued there with doctoral studies in the group of Malcolm H. Levitt. Their work primarily involved pulse sequence design for determining molecular structures by solid-state NMR, as well as the development of methods for numerically simulating NMR experiments. He did postdoctoral work in the field of solid-state NMR on quadrupolar nuclei with Lucio Frydman at the University of Illinois at Chicago (2000) and at the Weizmann Institute of Science in Israel (2001). Currently, Dr. Edén is an assistant professor at Stockholm University, focusing on solid-state NMR methodology development for structural investigations of inorganic materials.