Available online at www.sciencedirect.com

SCIENCE DIRECT®

Journal of Magnetic Resonance 166 (2004) 174–181

JMR Journal of Magnetic Resonance

www.elsevier.com/locate/jmr

Rational reduction of periodic propagators for off-period observations

Wyndham B. Blanton,* John W. Logan, and Alexander Pines

Materials Sciences Division, Lawrence Berkeley National Laboratory, Berkeley, CA 94720, USA
Department of Chemistry, University of California, Berkeley, CA 94720, USA

Received 7 July 2003; revised 30 September 2003

Abstract

Many common solid-state nuclear magnetic resonance problems take advantage of the periodicity of the underlying Hamiltonian to simplify the computation of an observation. Most of the time-domain methods used, however, require the time step between observations to be some integer or reciprocal-integer multiple of the period, thereby restricting the observation bandwidth. Calculations of off-period observations are usually reduced to brute force direct methods resulting in many demanding matrix multiplications. For large spin systems, the matrix multiplication becomes the limiting step. A simple method that can dramatically reduce the number of matrix multiplications required to calculate the time evolution when the observation time step is some rational fraction of the period of the Hamiltonian is presented. The algorithm implements two different optimization routines. One uses pattern matching and additional memory storage, while the other recursively generates the propagators via time shifting. The net result is a significant speed improvement for some types of time-domain calculations.
© 2003 Published by Elsevier Inc.

Keywords: NMR; Simulations; Propagator reduction; Time-dependent

1. Introduction

Many computational problems in nuclear magnetic resonance (NMR) are solved using a variety of simplification techniques related to the underlying Hamiltonian and its time dependence [1]. Often, the goal is to calculate the density matrix evolution over long time scales (up to ≈ 1 s) in order to observe dynamics as a function of time.

$$\rho(t) = U(t,0)\rho(0)U(t,0)^{-1}. \tag{1}$$

A propagator must be calculated in order to evolve the density matrix in time. Any evolution propagator can be calculated via the *direct method* given by

$$U(t_0 + \Delta t, t_0) = T \exp\left[-\mathrm{i}\int_{t_0}^{t_0+\Delta t} H(t')\,\mathrm{d}t'\right], \tag{2}$$

where T is the Dyson time-ordering operator. When the Hamiltonian is time-independent, the integral can be evaluated analytically in one step (a constant matrix multiplied by Δt). Consequently, the only required $O(N^3)$ operation, where N is the size of the Hilbert space, is a single matrix exponential. Most liquid-state or static solid-state NMR simulations can be calculated very quickly using this method. Additional computational difficulties exist when the Hamiltonian is time-dependent. When the Hamiltonian does not commute with itself at different times, Eq. (2) can be approximated by dividing the time interval Δt into M equally spaced time intervals δt such that $M\,\delta t = \Delta t$.

$$U(t_0 + \Delta t, t_0) = T \prod_{k=0}^{M-1} \exp\left[-\mathrm{i}H(t_0 + t_k)\delta t\right], \tag{3}$$

where T assures that the product is time-ordered, δt is chosen to be much smaller than any time dependence in H, and the elements of t_k are time points where the Hamiltonian is evaluated. Choosing the time points in t_k such that they lie in the middle of each interval δt corresponds to an implementation of the midpoint rule for Riemann sums to approximate the integral in Eq. (2).

* Corresponding author. Fax: 1-5104865744.
E-mail address: bo@theaddedones.com (W.B. Blanton).

1090-7807/$ - see front matter © 2003 Published by Elsevier Inc.
doi:10.1016/j.jmr.2003.10.002

The evaluation of $U(t_0 + \Delta t, t_0)$ using Eq. (3) requires M matrix exponentiations and $M - 1$ matrix multiplications [2]. Both matrix exponentiation and matrix multiplication are $O(N^3)$ operations [2]. All propagators for time-dependent Hamiltonians can be numerically evaluated in this fashion. For Q_i spin-I_i particles, the dimension of the Hilbert space is

$$N = \prod_i (2I_i + 1)^{Q_i}, \tag{4}$$

where the index i runs over each different spin number in the system. Thus, N easily can become large making the number of $O(N^3)$ operations prohibitively time consuming.

Periodic Hamiltonians ($H(t) = H(t + \tau)$, where τ is the period of the modulation) appear frequently in NMR. One of the most common cases of a periodic Hamiltonian is the mechanical spinning of a sample at a constant rate and angle with respect to the static magnetic field (e.g., magic angle spinning, MAS). There are many algorithmic tricks that can be implemented in order to simplify the calculation of periodic Hamiltonians and their corresponding propagators.

Any periodic problem potentially can be solved using a Floquet treatment [3,4]. However, for large Hilbert spaces or if the frequency range during the evolution is large, the Floquet method becomes too computationally demanding. This is particularly true for quadrupolar nuclei under complex radio frequency irradiation and sample spinning [5] where numerous frequency modes are required to achieve a good simulation. Complex pulse sequences are also hard to manipulate in a Floquet treatment, as every pulse and density matrix evolution step must be converted to the large Floquet space. Such techniques also fail to retrieve all the information about the density matrix as it evolves through time as typically only the observable is propagated. In addition, inclusion of relaxation phenomena into a frequency space propagation further complicates the computation by increasing the number of modes required for multi-spin simulation. Many independent large complex simulations would be required to effectively simulate the most realistic spin situation using a frequency-based method.

In this paper we are interested in the time evolution of the density matrix, not the transitions in the underlying Hamiltonian. As a result, frequency-based methods will not be discussed further. The most common method for time propagation under periodic Hamiltonians is using the periodic nature of the resulting propagator [6–8]. The technique is applicable only when the observation time is stroboscopic with the Hamiltonian time dependence. Fig. 1A shows a typical simulation when both the observation and Hamiltonian periodicity coincide. Such simulations are commonplace in many solid-state NMR recoupling experiments [9–11]. Fig. 1B shows the case when the observation and Hamiltonian periods do not coincide. In both cases the initial propagator, U_0 in the figure, are calculated using Eq. (3). This *preparation phase* is common to all time-domain simulations. We present an algorithm that can expedite the further time propagation of the *propagation phase* in Fig. 1B.

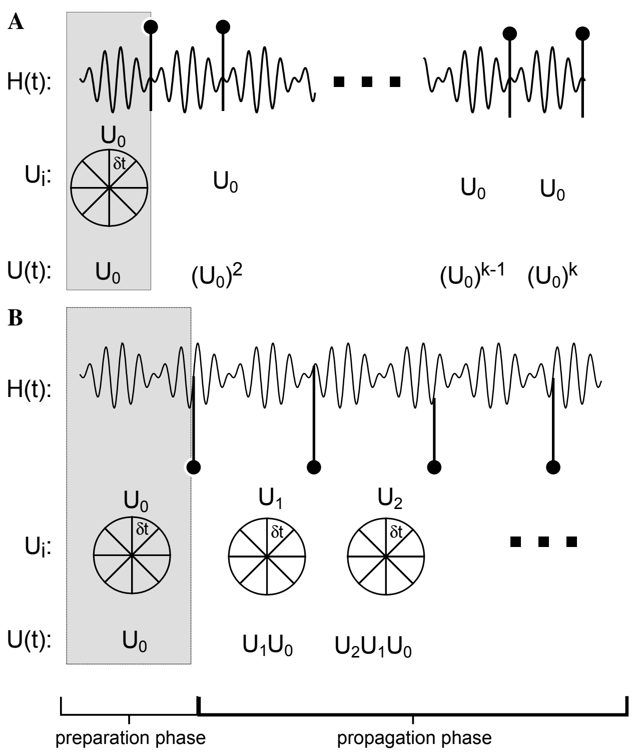

Fig. 1. Two simulation diagrams demonstrating: (A) synchronous observation and Hamiltonian periods and (B) asynchronous observation and Hamiltonian periods. Each dark dot indicated the desired observation point. A basic simulation diagram measuring k evolution points using the periodicity of the Hamiltonian, $H(t) = H(t + \tau)$. The initial calculation of U_0 (the *preparation phase*) is calculated by the smaller time segments δt as in Eq. (3), is reused to calculate the time signal at later times (*propagation phase*), $k\tau$. The preparation phase for (A) is usually the time limiting step. For (B), however, both the preparation phase and propagation phase can be equally time consuming.

Parallels can be drawn between this algorithm (the rational reduction algorithm) and lossless data compression [12]. There are two steps in the algorithm, reduction and grouping. These steps are analogous to data compression and data extraction (decompression), respectively. However, they differ in their optimization goals. The goal of the rational reduction algorithm is to minimize the number of matrix multiplications, whereas in data compression the goal is to minimize the number of bytes required to store a data set.

2. Periodicity and propagator reduction

Throughout this paper, NMR is used as an example, although the technique is general, and it may be applied to any periodic time-dependent Hamiltonian. Unless the overall Hamiltonian (including interactions, rotor spinning, and rf pulses) is periodic, the methods discussed below are invalid. The only way to achieve complete density matrix time evolution is using the direct method. A typical NMR experiment observes the system at integer multiples of time Δt. Given the Hamiltonian is periodic with period τ, and the observation occurs at a rational fraction of the period ($\frac{n}{m}\tau$), there are three general situations depending on the values of m and n. These situations are shown schematically in Fig. 2. Let m and n be positive integers defining the rational fraction n/m. Restricting m and n such that their greatest common divisor is unity ensures optimal propagator generation. For example, the cases where $(m = 9, n = 7)$ and $(m = 81, n = 63)$ are fundamentally equivalent in the discussion below, however, the $(m = 81, n = 63)$ case is reducible to $(m = 9, n = 7)$ which requires much less computation. The denominator m, is the *base factor*, or number of subpropagators per period τ. Here, a *subpropagator* is defined to be the propagator over a τ/m period. The subpropagators are labeled in a compact notation, where

$$U_i \equiv U\left((i+1)\frac{\tau}{m}, i\frac{\tau}{m}\right). \tag{5}$$

The numerator, n, represents the number of subpropagators per observation point, and it is named the *observation factor*. There are m observation points, spanning n modulation periods, between observations that are synchronous with τ. In other words, combining the observations and the Hamiltonian periodicity, an overall period of $n\tau$ can be defined. In all cases, the direct method (Eq. (3)) only has to be used to calculate the m subpropagators (U_i) that span one period τ. Next, the *grouped subpropagators*, U_{Gi}, are constructed from the initial set of subpropagators for $n > 1$.

2.1. $m = 1, n \geqslant 1$

The simplest way to make use of the Hamiltonian's periodicity is to realize that at each interval $n\tau$ the propagator equation can be written as

$$U(n\tau, 0) = U(\tau, 0)^n. \tag{6}$$

Therefore, the computationally expensive product in Eq. (3) only has to be evaluated over the interval $t = 0$ to $t = \tau$. This case provides the dynamics of the system only at specific time points ($t = n\tau$). The $m = 1$, $n \geqslant 1$ case is well suited for rotor synchronized pulse sequences where the rf pulse sequence is commensurate with the sample rotation (in the sense that $M\tau_{\text{seq}} = N\tau_{\text{rot}}$, with integral M, N). Only two propagators have to be calculated and stored ($U(\tau, 0)$ and $U(\tau, 0)^n \equiv U$). The density matrix time evolution is calculated via

$$\rho(t_{i+1}) = U\rho(t_i)U^{-1}. \tag{7}$$

2.2. $m > 1, n = 1$

The second case has $m > 1$ and $n = 1$. The period τ is divided into m subpropagators. The density matrix is advanced in time by one subpropagator for each observation point. There are m subpropagators that need to be stored. The appropriate U_i is chosen in a cyclic fashion during the density matrix time evolution

$$\rho(t_{i+1}) = U_{\text{mod}(i,m)}\rho(t_i)U^{-1}_{\text{mod}(i,m)}. \tag{8}$$

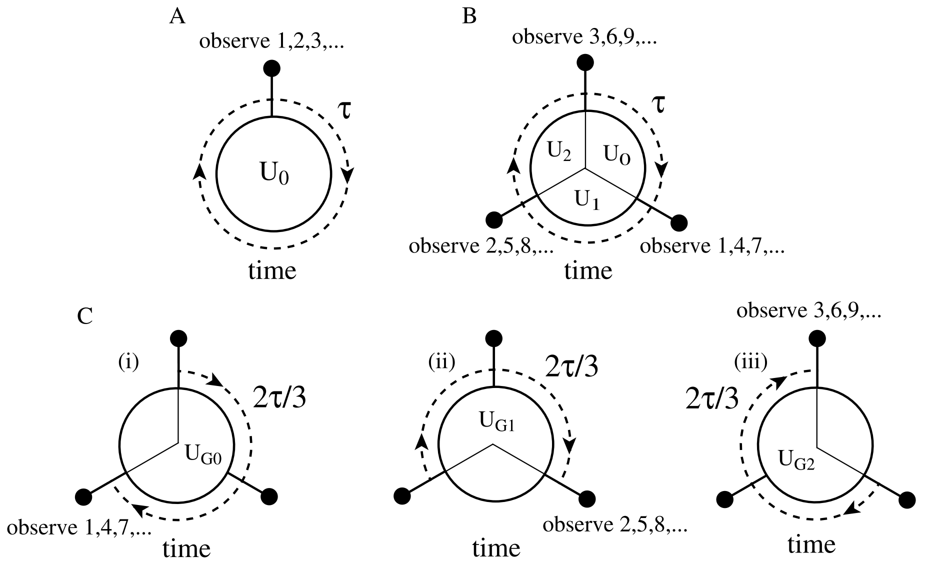

Fig. 2. The three scenarios for observing periodic problems. The (m, n) pairs correspond to the specific examples. (A) $(m = 1, n = 1)$ propagators. (B) $(m = 3, n = 1)$ propagators. (C) Subpropagator groups amenable to rational reduction $(m = 3, n = 2)$. The grouped subpropagators U_{Gi} correspond to series of n consecutive (modulo m) subpropagators. In this example, $U_{G0} = U_1U_0$, $U_{G1} = U_0U_2$, and $U_{G2} = U_2U_1$.

The $m > 1$ and $n = 1$ case is encountered often in NMR applications. Frequency-domain calculation algorithms such as COMPUTE [13] and γ-COMPUTE [14] use a similar calculation method as their starting point. These methods use subpropagators

$$V_i = T\prod_{j=0}^{i} U_i \tag{9}$$

and they use Fourier techniques to retrieve all resonance frequencies. On the other hand, the bandwidth, $\Delta\omega$, for $m > 1$ and $n = 1$ time-domain calculations is restricted to $1/(m\tau)$.

2.3. Rational reduction; $m > 1$, $n > 1$,($\gcd(m,n) = 1$)

In some cases, it may be necessary, or desirable, to observe the system at times $\frac{n}{m}\tau$, where $m > 1$ and $n > 1$. Under these conditions (including $\gcd(m,n) = 1$), the rational reduction algorithm can be performed. This algorithm can reduce substantially the number of matrix multiplications required to generate the U_{Gi} propagators.

Table 1
Subpropagator groups for $m = 9$ and $n = 7$

Subpropagator series	Subpropagator group
$U_6U_5U_4U_3U_2U_1U_0$	U_{G0}
$U_4U_3U_2U_1U_0U_8U_7$	U_{G1}
$U_2U_1U_0U_8U_7U_6U_5$	U_{G2}
$U_0U_8U_7U_6U_5U_4U_3$	U_{G3}
$U_7U_6U_5U_4U_3U_2U_1$	U_{G4}
$U_5U_4U_3U_2U_1U_0U_8$	U_{G5}
$U_3U_2U_1U_0U_8U_7U_6$	U_{G6}
$U_1U_0U_8U_7U_6U_5U_4$	U_{G7}
$U_8U_7U_6U_5U_4U_3U_2$	U_{G8}

To demonstrate how the rational reduction method achieves the same result but with fewer matrix multiplications, consider a normal propagation series where $m = 9$ and $n = 7$ for a periodic system with period τ. A sequence of seven subpropagators is required for each observation, and every nine observations the sequences repeat. These sequences are grouped together and given the names U_{Gi} as shown in Table 1. The rows of Table 1 represent the propagator to get from the previous observation point to the current observation point, and not the total propagator, i.e., the rows contain $U((i+1)\frac{n}{m}\tau, i\frac{n}{m}\tau)$ and not $U((i+1)\frac{n}{m}\tau, 0)$. Once these groups are formed, the problem is transformed into an analogous $m > 1, n = 1$ situation.

The creation of the grouped subpropagators, can be viewed as a renormalization of the original problem. Beginning with the m and n defined in reference to the period τ, the grouping renormalizes the original m and n values. The renormalization maps n to 1_R, and m to m_R, where the subscript R denotes the renormalized periodicity. The numerical value of m is unchanged during the renormalization. However, after the renormalization, m refers to the longer period $n\tau$ as opposed to τ prior to the renormalization. Fig. 3 shows a schematic of the renormalization procedure. Remember, the direct method (Eq. (3)) is only required for a period of τ, and not $n\tau$.

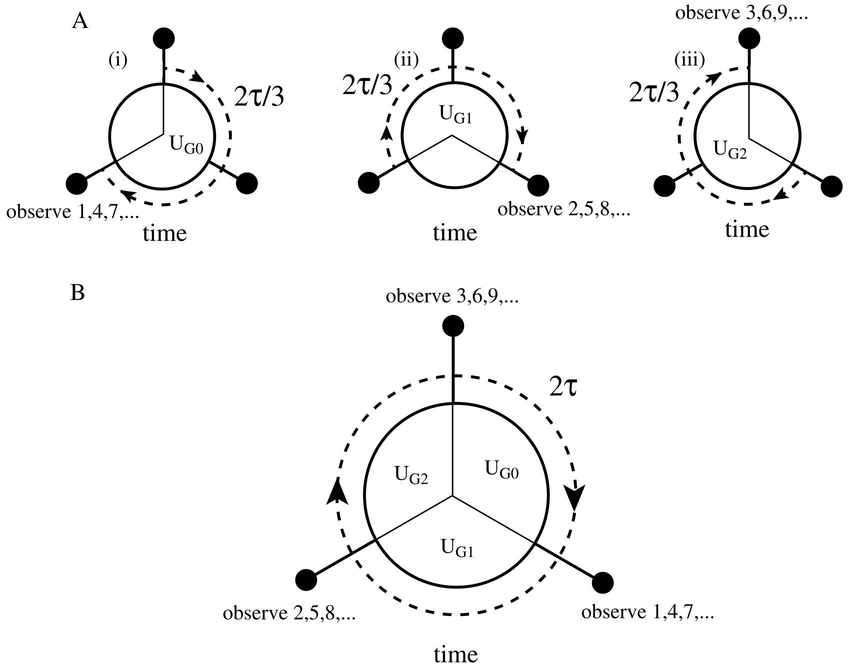

Fig. 3. Schematic of the renormalization from grouping of the subpropagators: (A) before renormalization and (B) after renormalization.

The time evolution of the density matrix can be calculated using the following equation:

$$\rho(t_{i+1}) = U_{\mathrm{Gmod}(i,m)}\rho(t_i)U^{-1}_{\mathrm{Gmod}(i,m)}. \tag{10}$$

The only difference between Eqs. (8) and (10) is the addition of the subscript G denoting the use of a group of subpropagators in the latter equation instead of an individual subpropagator in the former equation.

As shown in Table 1, m subpropagators (U_i), spanning the time interval $t = 0$ to $t = \tau$ in steps of τ/m, are needed to calculate the m grouped subpropagators ($U_{\mathrm{G}i}$). The $U_{\mathrm{G}i}$s, which span time steps of $(n\tau)/m$, can be calculated simply by time-ordered products of the U_is. This naive approach is used as the reference point for evaluating the efficiency of the rational reduction.

The most intensive computation is the calculation of the initial m subpropagators. This calculation is unavoidable, and it uses the direct method (Eq. (3)). The rational reduction algorithm presented here minimizes the number of matrix multiplications used to generate the m grouped subpropagators starting from the U_is. There are two optimization methods used in the first stage of the rational reduction algorithm, *sequence reduction* and *time shifting reduction*.

2.3.1. Case 1: $m > n$, sequence reduction

If $m > n$ (e.g., $m = 9$ and $n = 7$, as shown in Table 1), the n subpropagator sequences for each $U_{\mathrm{G}i}$ are reformulated into new sequences using a pattern matching algorithm. The new sequences contain one matrix for $U_{\mathrm{G}0}$; two matrices for $U_{\mathrm{G}1}$ through $U_{\mathrm{G}(m-2)}$; and one matrix for $U_{\mathrm{G}(m-1)}$, unless $m = n + 1$, in which case there are two matrices. The pattern matching is tantamount to a lossless data compression since the net effect is to encrypt the information in fewer symbols. This reduction method trades computational cost (matrix multiplications) for an increase in memory storage. For example, in Table 1, the sequence U_8U_7 appears six times. If that sequence was stored the first time it was calculated and subsequently recalled as needed, five matrix multiplications would be saved. All possible sequences are considered in the sequence reduction method. After matching the series of n subpropagators with one or two sequences, the number of matrix multiplications needed to generate the $U_{\mathrm{G}i}$ sequences is often dramatically reduced. This point will be described more quantitatively in the following section.

There are several possible types of sequences. Sequences with the form $U_xU_{x-1}\cdots U_1U_0$, called *forward sequences*, are denoted by F_x. Similarly, *backward sequences*, denoted by B_x, have the form $U_{m-1}U_{m-2}\cdots U_{x+1}U_x$. Despite several redundancies in this naming scheme, the rational reduction algorithm does not compute or store redundant matrices. The redundancies include $U_{m-1}U_{m-2}\cdots U_1U_0 \equiv F_{m-1} \equiv B_0$ which is a complete cycle and is named U_{C}; $F_0 \equiv U_0$; and $B_{m-1} \equiv U_{m-1}$. Inverse forward sequences and inverse backward sequences are introduced in order to save additional matrix multiplications.

As previously mentioned, U_{C} can be calculated as either a forward sequence, or a backward sequence. Assuming U_{C} is calculated as a forward sequence. Initially, U_1U_0 must be calculated, and it could be stored as F_1, followed by $F_2 = U_2F_1$, $F_3 = U_3F_2$, etc. The forward sequences are considered "free," in the sense that they require only extra memory, and not additional computational cost. The backward sequences, $U_{m-1}U_{m-2}\cdots$ are not free, as they do not have to be calculated as a part of another calculation. Therefore, assuming the backward sequences are calculated in an analogous fashion as the forward sequences (e.g., $B_{m-2} = U_{m-1}U_{m-2}$, $B_{m-3} = B_{m-2}U_{m-3}$, etc.) the backward sequences cost one matrix storage and one matrix multiplication for each sequence.

Alternatively, U_{C} could be calculated from backward sequences. In which case, the B_xs are free, and the F_xs cost one matrix storage and one matrix multiplication each. The inverse sequences are calculated on-the-fly, as needed. No inverse sequence is used more than once in the calculation of a set $U_{\mathrm{G}i}$s. The inverse sequences are needed only when the corresponding forward or backward sequences are used elsewhere. The inverse sequences do not require any additional $\mathrm{O}(N^3)$ matrix operations since the inverse of a unitary matrix is equal to the adjoint. The inverse can be computed as an $\mathrm{O}(N^2)$ process (complex conjugate transpose), and not an $\mathrm{O}(N^3)$ process (generalized matrix inversion). The inverse sequences are needed only when the corresponding forward or backward sequences are used for other calculations. Finally, only a subset of the forward and backward sequences is necessary to compute the subpropagator groups.

2.3.2. Case 2: $m < n$, time shifting reduction

The second reduction method in the rational reduction algorithm has a simple solution that requires no additional matrices to be stored. The first group is calculated via

$$U_{\mathrm{G}0} = F_{\mathrm{mod}(n-1,m)}(U_{\mathrm{C}})^{\mathrm{floor}(n/m)}. \tag{11}$$

Even though a forward sequence is shown in Eq. (11), it does not need to be stored explicitly. Next, the remaining $m-1$ $U_{\mathrm{G}i}$s are calculated (possibly out of numerical order) by shifting them in time. The time shifting can be implemented by shifting forward or backward one subpropagator each step. The remaining $U_{\mathrm{G}i}$s can be generated recursively using Eq. (12) for forward time shifting, or Eq. (13) for backward time shifting.

$$U_{\mathrm{G}(\mathrm{order_F}(j))} = U_{\mathrm{mod}(\mathrm{mod}(n-1,m)+j,m)}U_{\mathrm{G}(\mathrm{order_F}(j-1))}U^{-1}_{j-1}, \tag{12}$$

$$U_{\mathrm{G}(\mathrm{order_B}(j))} = U^{-1}_{\mathrm{mod}(n-j,m)}U_{\mathrm{G}(\mathrm{order_B}(j-1))}U_{m-j}, \tag{13}$$

where the index j runs from 1 to $m-1$, and order_F and order_B are lists whose elements correspond to the order that the grouped subpropagators are calculated.

$$\text{order}_F(\text{mod}(j\text{mod}(n,m),m)) = j, \tag{14}$$

$$\text{order}_B(\text{mod}(m^2 - j\text{mod}(n,m),m)) = j. \tag{15}$$

An example where $m = 5$ and $n = 12$ is shown in Table 2. The top half of Table 2 demonstrates forward time shifting, while the bottom half demonstrates backward time shifting. There are m different possible starting positions for a sequence of an arbitrary length n. This time shifting method requires $2(m-1)$ matrix multiplications to calculate the final $m-1$ grouped subpropagators.

3. Results and discussion

The two methods for the reduction step (sequence reduction or time shifting reduction) in the rational reduction algorithm do not require any actual propagators. They only use the indices of the subpropagators and the labels for the various types of sequences. Given m and n, the optimization is performed via one of the reduction methods. After the reduction stage determines the optimized scheme to produce the grouped subpropagators, the second step of the algorithm generates the grouped subpropagators. A C++ class *RationalReduction* is given in the *BlochLib* [15] distribution that contains routines to perform both steps of the algorithm.

The reduction step (sequence reduction or time shifting reduction) only needs to be performed once per simulation, independent of both the number of crystallite orientations in a powder average and the dimension of the Hilbert space. For these reasons, the overhead of having to perform the reduction step becomes increasingly negligible as the size of the Hilbert space increases and/or as the number of orientations in a powder average increases.

The results from the sequence reduction for the example where $m = 9$, and $n = 7$ are shown in Table 3. Comparing the left column of Table 1 to the right column of Table 3 for the sequence reduction, or the two left-most columns in Table 2 for time shifting reduction, demonstrate the parallels with data compression.

In the $m = 9, n = 7$ example, one inverse sequence is used. The number of inverse sequences used is zero for $n = 2$, and $m - n - 1$ for $n > 2$. The efficiency of the rational reduction versus the naive method was evaluated in terms of matrix multiplications and memory storage by calculating all combinations of the positive integers m and n less than 160 that satisfy the condition $\gcd(m,n) = 1$.

The contour plot in Fig. 4 illustrates the efficiency of the rational reduction method. The formulae in Eq. (16) return the number of matrix multiplications, $M(m,n)$, used to generate U_C and U_{Gi}. Eq. (16) exactly matches all calculated values, yet it does not have the restriction that $\gcd(m,n) = 1$.

$$M(m,n) = \begin{cases} 3(m-1) + \text{floor}\,(n/m); & m < n, \\ 2(m-1); & m \geqslant n, n = 2, \\ 2(m-1) + \text{ceil}((m+n)/2) - 3; & m \geqslant n, n > 2. \end{cases} \tag{16}$$

The top formula in Eq. (16) is also valid for $m > n$ when the time shifting reduction is used. The naive method for

Table 3
A reduced set of propagators for $m = 9$ and $n = 7$ produced using the rational reduction algorithm (sequence reduction)

Subpropagator group	Reduced series
U_{G0}	F_6
U_{G1}	F_4B_7
U_{G2}	F_2B_5
U_{G3}	U_0B_3
U_{G4}	$F_7U_0^{-1}$
U_{G5}	F_5U_8
U_{G6}	F_3B_6
U_{G7}	F_1B_4
U_{G8}	B_2

Table 2
Time shifting reduction for $m = 5$ and $n = 12$

	Reduced format	Subpropagator group	Index j
Forward time shifting subpropagator sequence			
$U_1U_0U_4U_3U_2U_1U_0U_4U_3U_2U_1U_0$	$F_1U_CU_C$	U_{G0}	
$U_2U_1U_0U_4U_3U_2U_1U_0U_4U_3U_2U_1$	$U_2U_{G0}U_0^{-1}$	U_{G3}	1
$U_3U_2U_1U_0U_4U_3U_2U_1U_0U_4U_3U_2$	$U_3U_{G3}U_1^{-1}$	U_{G1}	2
$U_4U_3U_2U_1U_0U_4U_3U_2U_1U_0U_4U_3$	$U_4U_{G1}U_2^{-1}$	U_{G4}	3
$U_0U_4U_3U_2U_1U_0U_4U_3U_2U_1U_0U_4$	$U_0U_{G4}U_3^{-1}$	U_{G2}	4
Backward time shifting subpropagator sequence			
$U_1U_0U_4U_3U_2U_1U_0U_4U_3U_2U_1U_0$	$F_1U_CU_C$	U_{G0}	
$U_0U_4U_3U_2U_1U_0U_4U_3U_2U_1U_0U_4$	$U_1^{-1}U_{G0}U_4$	U_{G2}	1
$U_4U_3U_2U_1U_0U_4U_3U_2U_1U_0U_4U_3$	$U_0^{-1}U_{G2}U_3$	U_{G4}	2
$U_3U_2U_1U_0U_4U_3U_2U_1U_0U_4U_3U_2$	$U_4^{-1}U_{G4}U_2$	U_{G1}	3
$U_2U_1U_0U_4U_3U_2U_1U_0U_4U_3U_2U_1$	$U_3^{-1}U_{G1}U_1$	U_{G3}	4

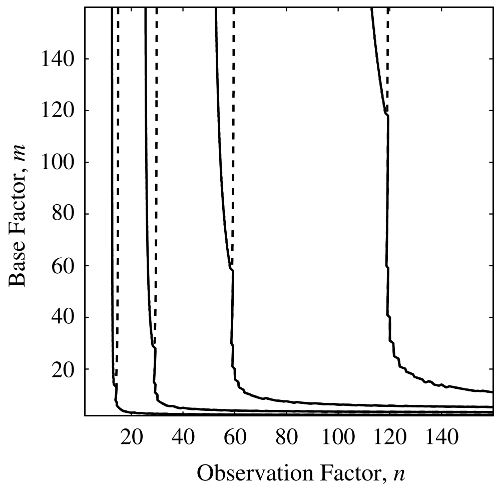

Fig. 4. Effectiveness of the rational reduction method at reducing the number of matrix multiplications for calculating the grouped subpropagators compared to the original method. The percent reduction contour levels are 80, 90, 95, and 97.5%, increasing towards higher m and n values. The dashed contours refer to the efficiency for using the time shifting for cases when $m > n$. The contours are equivalent to a factor of 5, 10, 20, and 40 fewer matrix multiplications for the rational reduction than the naive approach, respectively.

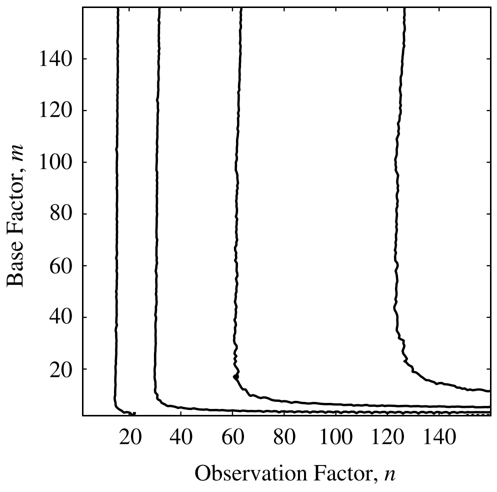

Fig. 5. Relative speed of the calculation of U_C and the U_{Gi}s with and without the rational reduction algorithm. The contour levels indicate a factor of 5, 10, 20, and 40 increase in speed using the rational reduction algorithm versus the naive approach to grouping (increasing towards larger m and n values). A Hilbert space dimension of 100×100 was used. In nearly all m, n pairs, the time shifting reduction was faster than the sequence reduction. The contours in this figure are in good agreement to the predictions made in Fig. 4 using the time shifting reduction. An interpolation procedure was used prior to the calculation of the contour levels since the data is only obtainable when $\gcd(m,n) = 1$.

generating the U_C and the U_{Gi}s takes $m \times n - 1$ matrix multiplications.

The optimal solution from the sequence reduction for $m > n$ and $n > 2$ requires approximately $m + n$ additional matrices (the forward and backward sequences) to be stored in memory while the U_{Gi}s are calculated. Although the time shifting reduction uses no additional memory storage, it requires approximately $(m - n)/2$ more matrix multiplications than the sequence reduction (for $m > n$). The dashed contour lines in Fig. 4 correspond to the reduction efficiency using the time shifting algorithm when $m > n$. A modified form of the sequence reduction could be used when $m < n$, however, the sequence reduction never uses fewer matrix multiplications than the time shifting reduction. This fact, combined with the extra memory requirements for the sequence reduction and more complicated implementation, makes the time shifting reduction the method of choice for $m < n$.

Up to this point the efficiency of the rational reduction has been measured in terms of the number of matrix multiplications. In practice, however, the speed (and accuracy) of the computation is all that matters. Some factors that affect computational performance include the memory requirements and memory management, the matrix multiplication library, the programming language implementation of the algorithm, computer hardware, and the compiler and/or compiler options used. Many of these factors are outside the scope of the discussion of the rational reduction algorithm, nevertheless, measures have been taken to use fast library functions (e.g., ATLAS [16] and *BlochLib* [15]) and to write efficient code for the implementation of the rational reduction algorithm. Fig. 5 shows the actual speed improvements that can be realized by using the rational reduction algorithm when grouping subpropagators. The data in Fig. 5 are in good agreement with Fig. 4, suggesting the time to perform the reduction step is negligible, since the computation time is primarily determined by the number of matrix multiplication that need to be performed (at least for larger Hilbert space dimensions).

The sequence reduction (both the reduction step and the grouping step) is substantially more complicated than the time shifting reduction. As a consequence of the differences in the complexity of the algorithms, the fastest reduction and grouping method may not be the method that uses the fewest matrix multiplications. Some general recommendations for choosing which method to use include: Time shifting reduction should be used whenever $m < n$. For $n = 2$, use the naive approach for grouping. For small n values ($n \lesssim 4$), again the naive approach for grouping may be faster than either of the reduction methods. Time shifting reduction is often the fastest method when $m > n$, despite the fact that it uses more matrix multiplications than the sequence reduction method. Regardless of these recommendations, a quick test to determine which method performs the best for each combination of m, n, and N with $m > n$ and $n > 2$ should be performed. There is a dependence on the size of the Hilbert space, N. The details of the implementation of the rational reduction algorithm, beyond what was mentioned previously, are included in the documentation of the C++ class *RationalReduction* in the *BlochLib* library.

In order to demonstrate the rational reduction in a specific example, a spin system that models manganese decacarbonyl, $Mn_2(CO)_{10}$, was used. The spin system includes two coupled $I = 5/2$ which has a Hilbert space dimension of 36×36. The dipole–dipole coupling between the two Mn nuclei is 305 Hz (calculated using the internuclear distance from X-ray diffraction data [17]). The other couplings used were 3.01 MHz quadrupolar coupling and 65 Hz isotropic J coupling [18]. The quadrupolar–dipole cross-term was neglected. The "isotropic" projection from the 3QMAS (triple quantum version of the multiple-quantum MAS [19]) spectrum was calculated.

Instead of calculating a full 2D data set with evolution under a multiple-quantum coherence during t_1 and evolution under single-quantum coherence during t_2, as is collected experimentally, the desired spectrum was calculated from a 1D data set that traverses the two time domains with a slope of $t_2 = (19/12)t_1$. Calculating this 1D data set across the two time domains reduces the amount of computation to less than what would be required for three slices of the 2D data set (a potential saving of several orders of magnitude).

The ratio 19/12 is determined by ratio of the expansion coefficients for the rank 4 terms of the second-order quadrupolar Hamiltonian under triple- and single-quantum coherences (for $I = 5/2$ nuclei) [19]. After evolution under two different coherences for the appropriate times (according to the aforementioned ratio), the inhomogeneous broadening caused by a the rank 4 terms of the second-order quadrupolar Hamiltonian is removed. The rank 2 terms are removed by MAS, leaving a resonance that is not broadened by anisotropic quadrupolar interactions.

The calculation of the isotropic dimension without using the rational reduction algorithm for 1154 crystallite orientations took approximately 877 s, whereas using rational reduction to group the propagators took 829 s (a 6% improvement). A similar experiment using a two spin $I = 7/2$ system requires $t_2 = (55/101)t_1$ (or $n/m = 55/101$) to measure the isotropic projection. The simulation of this larger system took approximately 5080 s without rational reduction and 3570 s (a 30% improvement) using rational reduction [20].

4. Conclusions

The rational reduction algorithm is applicable to the generation of any set of grouped subpropagators (i.e., $m > 1,\ n > 1,\ \gcd(m,n) = 1$). The algorithm can decrease the time for calculations by reducing the number of matrix multiplications to generate the grouped subpropagators. Rational reduction is useful for time-domain calculations where the elements in the density matrix are of interest, and not just the energy levels of the Hamiltonian.

Fully exploiting the periodicity of a time-dependent Hamiltonian can dramatically reduce the number of matrix multiplications and matrix exponentiations involved in calculating long time spin dynamics. For off-period observations, a new algorithm is presented that minimizes the number of matrix multiplications used to calculate the grouped subpropagators. This rational reduction technique can provide vast speed improvements for the calculations of the grouped subpropagators needed for time-domain observations of the density matrix. Determination of the reduction itself is a fast procedure using only integer labels, and its computation time is generally negligible compared to the computation time of the propagators.

Acknowledgments

J.W.L. thanks Jamie Walls for helpful discussions. This work was supported by the Director, Office of Science, Office of Basic Energy Sciences, Materials Science and Engineering Division, U.S. Department of Energy under Contract No. DE-AC03-76SF00098.

References

[1] P. Hodgkinson, L. Emsley, Prog. Nucl. Magn. Reson. Spectrosc. 36 (2000) 201.
[2] G.H. Golub, C.F. Van Loan, Matrix Computations, third ed., Johns Hopkins University Press, Baltimore, MD, 1996.
[3] J.H. Shirley, Phys. Rev. B 138 (1965) 979.
[4] A. Schmidt, S. Vega, J. Chem. Phys. 96 (1992) 2655.
[5] T. Charpentier, C. Fermon, J. Virlet, J. Magn. Reson. 132 (1998) 181.
[6] U. Haeberlen, J.S. Waugh, Phys. Rev. 175 (1968) 453.
[7] U. Haeberlen, High-Resolution NMR in Solids. Selective Averaging, Academic Press, New York, 1976.
[8] M. Mehring, Principles of High Resolution NMR in Solids, Springer-Verlag, New York, 1983.
[9] M. Eden, M.H. Levitta, J. Chem. Phys. 111 (4) (1999) 1511.
[10] M. Carravetta, M. Eden, X. Zhao, A. Brinkmann, M.H. Levitt, Chem. Phys. Lett. 321 (2000) 205.
[11] X. Zhao, M. Eden, M. Levitt, Chem. Phys. Lett. 234 (2001) 353.
[12] D. Salomon, Data Compression: The Complete Reference, second ed., Springer-Verlag, New York, 2000.
[13] M. Edén, Y.K. Lee, M.H. Levitt, J. Magn. Reson. A 120 (1996) 56.
[14] M. Hohwy, H. Bildsøe, H.J. Jakobsen, N.C. Nielsen, J. Magn. Reson. 136 (1999) 6.
[15] W.B. Blanton, J. Magn. Reson. 162 (2003) 269.
[16] W. Clint, Automatically Tuned Linear Algebra Software (ATLAS), URL: available from http://math-atlas.sourceforge.net.
[17] M. Martin, B. Rees, A. Mitschler, Acta Crystallogr. B 38 (1982) 6.
[18] S. Wi, L. Frydman, J. Chem. Phys. 112 (2000) 3248.
[19] A. Medek, J.S. Harwood, L. Frydman, J. Am. Chem. Soc. 117 (1995) 12779.
[20] All simulations were performed on a 1 GHz Apple PowerBook under OS X 10.2.8 and gcc 3.3.1.