JOURNAL OF MAGNETIC RESONANCE, Series A **101**, 103–105 (1993)

COMMUNICATIONS

POMA: A Complete *Mathematica* Implementation of the NMR Product-Operator Formalism

PETER GÜNTERT, NIKLAUS SCHAEFER, GOTTFRIED OTTING, AND KURT WÜTHRICH

Institut für Molekularbiologie und Biophysik, Eidgenössische Technische Hochschule-Hönggerberg, CH-8093 Zurich, Switzerland

Received September 29, 1992

For weakly coupled spin systems the time evolution of the density operator in a pulsed NMR experiment can be calculated analytically by expressing the Hamiltonian and the density operator in the basis of Cartesian product operators (*1, 2*). This approach, which combines a rigorous quantum mechanical treatment with an intuitive classical description, was the basis for the development of a wide variety of intricate NMR experiments (*2, 3*). All calculations within the product-operator formalism proceed by repeated application of a small number of simple rules, which describe the time evolution of individual product operators under chemical shifts, scalar couplings, and radiofrequency pulses. Although these calculations are straightforward, the large number of terms arising can make them tedious, and computer support is highly desirable. Recently, a partial implementation of the product-operator formalism in *Mathematica* (*4*), a programming language designed for symbolic computation, has been reported (*5*). The present Communication describes a complete, highly flexible *Mathematica* implementation of the product-operator formalism for spin-$\frac{1}{2}$ nuclei, POMA ("*p*roduct *o*perator formalism in *Ma*thematica"), which provides analytical results for the time evolution of weakly coupled spin systems under the influence of free precession, selective and nonselective pulses, and phase cycling. The complexity of the problems that can be treated is in practice limited only by the available computer time and memory. The complete *Mathematica* code is given in Fig. 1. It is apparent that *Mathematica* offers a particularly concise and elegant way to express the transformation rules of the product-operator formalism. If desired, the results of the calculations can be further processed using the built-in capabilities of *Mathematica* to handle mathematical expressions.

In the POMA implementation of the product-operator formalism, a spin operator $I_{k\alpha}$ that corresponds to the α component of spin k ($\alpha = x, y, z$; $k = 1, 2, \ldots$) is represented in the input by **spin**[k, α] and in the output by $X_{k\alpha}$, where the letter X can be set with the command **nucleus**[k] = X. A delay of time period t, during which the spins in the set $\{k, l, \ldots\}$ evolve under chemical shifts with frequencies ω_k, $\omega_l, \ldots$ and under scalar couplings J_{mn}, $J_{pq}, \ldots$ between the spin pairs in the set $\{mn, pq, \ldots\}$ is represented by

$$\textbf{delay}[t, \{\{m, n\}, \{p, q\}, \ldots\}, \{k, l, \ldots\}].$$

By default, all spins evolve under chemical shifts, and no scalar couplings are assumed. A pulse of flip angle β that acts on the spins in the set $\{k, l, \ldots\}$ and is phase cycled according to $\{\phi_1, \phi_2, \ldots\}$ is represented by

$$\textbf{pulse}[\beta, \{\phi_1, \phi_2, \ldots\}, \{k, l, \ldots\}].$$

If a flip angle or a phase angle is given as a pure number, it is assumed to be given in degrees; the phases $\phi = x, y, -x, -y$ correspond to phase angles of 0°, 90°, 180°, and 270°, respectively. If the expression for a flip or a phase angle is neither a pure number nor one of the symbols $x, y, -x, -y$, it is assumed to be given in radians. If the phase is identical in all steps of a phase cycle $\{\phi, \phi, \ldots\}$, it can be represented by a single phase ϕ. By default, a nonselective pulse is assumed; i.e., all spins are affected by the pulse. Thus, for instance, a hard 90° x pulse can be written in shorthand notation as **pulse**[**90**, **x**]. The receiver phase cycle $\{\psi_1, \psi_2, \ldots\}$ is specified as

$$\textbf{receiver}[\{\psi_1, \psi_2, \ldots\}].$$

A receiver phase $\psi \neq x$ is simulated by applying a pulse of flip angle $\Delta\psi$ around the z axis to the product operators present before signal detection, where $\Delta\psi$ is the phase difference with respect to the x axis.

The program repeats the calculation for each individual step of the phase cycle, adds the results, and normalizes the terms by the length of the phase cycle. The final result may be screened for the observable terms, i.e., terms with a single transverse spin operator, using the command

1064-1858/93 $5.00

```
MemberQ[allspins_] ^= True
delay[t_,couples_:{},spins_:allspins][sigma_] := Block[{r,c},
    WriteString["stdout","delay[",InputForm[t],",",couples,",",spins,"]"];
    r = Timing[sigma // shift[t,spins] // coupling[t,couples] // simplifysigma // normalize]; (*Eq. 13*)
    c = r[[1]] /. Second->1; Print[" "]; Print[" (",terms[r[[2]]]," terms, ",N[c,3]," s CPU time)"]; r[[2]]]

shift[t_,spins_:allspins][sigma_] := sigma /. {
    spin[n_,x] :> spin[n,x] Cos[w[n]t] + spin[n,y] Sin[w[n]t] /; MemberQ[spins,n],    (*Eq. 14*)
    spin[n_,y] :> spin[n,y] Cos[w[n]t] - spin[n,x] Sin[w[n]t] /; MemberQ[spins,n]}    (*Eq. 15*)

Attributes[j] = {Orderless}
coupling[t_,couples_:{}][sigma_] := sigma /; couples=={}
coupling[t_,{{m1_,m2_}}][sigma_] := normalize[sigma] /. {
    spin[n1_,k_] spin[n2_,l_] :> spin[n1,k] spin[n2,l] /;
        MemberQ[{x,y},k] && MemberQ[{x,y},l] && {m1,m2}=={n1,n2},
    spin[n1_,x] spin[n2_,z] :> spin[n1,x] spin[n2,z] Cos[Pi j[m1,m2]t] +
        spin[n1,y] Sin[Pi j[m1,m2]t] /; {m1,m2}=={n1,n2} || {m1,m2}=={n2,n1},    (*Eq. 19*)
    spin[n1_,y] spin[n2_,z] :> spin[n1,y] spin[n2,z] Cos[Pi j[m1,m2]t] -
        spin[n1,x] Sin[Pi j[m1,m2]t] /; {m1,m2}=={n1,n2} || {m1,m2}=={n2,n1},    (*Eq. 20*)
    spin[n_,x] :> spin[n,x] Cos[Pi j[m1,m2]t] + spin[n,y] spin[m2,z] Sin[Pi j[m1,m2]t] /;
        m1==n,                                                                    (*Eq. 17*)
    spin[n_,x] :> spin[n,x] Cos[Pi j[m1,m2]t] + spin[n,y] spin[m1,z] Sin[Pi j[m1,m2]t] /; m2==n,
    spin[n_,y] :> spin[n,y] Cos[Pi j[m1,m2]t] - spin[n,x] spin[m2,z] Sin[Pi j[m1,m2]t] /;
        m1==n,                                                                    (*Eq. 18*)
    spin[n_,y] :> spin[n,y] Cos[Pi j[m1,m2]t] - spin[n,x] spin[m1,z] Sin[Pi j[m1,m2]t] /; m2==n }
coupling[t_,couples_][sigma_] := sigma // coupling[t,{First[couples]}] // coupling[t,Rest[couples]]

pulse[b_,p_,spins_:allspins][sigma_] := Block[{r,c},
    WriteString["stdout","pulse[",InputForm[b],",",InputForm[p],",",spins,"]"];
    r = Timing[normalize[pulses[b,p,spins][sigma]]];
    c = r[[1]] /. Second->1; Print[" "]; Print[" (",terms[r[[2]]]," terms, ",N[c,3]," s CPU time)"]; r[[2]]]
pulses[b_,p_List,spins_][sigma_List] := Block[{i},
    Table[ pulses[b,p[[i]],spins] [ sigma[[i]] ], {i,1,Length[p]}]]
pulses[b_,p_List,spins_][sigma_] := Block[{i},
    Table[pulses[b,p[[i]],spins][sigma],{i,1,Length[p]}]]
pulses[b_, p_,spins_] := pulses[b Pi/180,p,spins] /; NumberQ[b] && b!=0
pulses[b_, p_,spins_] := pulses[b,p Pi/180,spins] /; NumberQ[p] && p!=0
pulses[b_,-p_,spins_] := pulses[-b,p,spins]
pulses[b_, p_,spins_][sigma_]:= sigma /. {
    spin[n_,k_]:> spin[n,k] /; p==k,
    spin[n_,z] :> spin[n,z] Cos[b] - spin[n,y] Sin[b] /; p==x && MemberQ[spins,n],    (*Eq. 24*)
    spin[n_,y] :> spin[n,y] Cos[b] + spin[n,z] Sin[b] /; p==x && MemberQ[spins,n],    (*Eq. 25*)
    spin[n_,z] :> spin[n,z] Cos[b] + spin[n,x] Sin[b] /; p==y && MemberQ[spins,n],    (*Eq. 26*)
    spin[n_,x] :> spin[n,x] Cos[b] - spin[n,z] Sin[b] /; p==y && MemberQ[spins,n],    (*Eq. 27*)
    spin[n_,x] :> spin[n,x] Cos[b] + spin[n,y] Sin[b] /; p==z && MemberQ[spins,n],    (*Eq. 28*)
    spin[n_,y] :> spin[n,y] Cos[b] - spin[n,x] Sin[b] /; p==z && MemberQ[spins,n],    (*Eq. 29*)
    spin[n_,z] :> spin[n,z] Cos[b] + spin[n,x] Sin[b] Sin[p] -
        spin[n,y] Sin[b] Cos[p] /; MemberQ[spins,n],                              (*Eq. 35*)
    spin[n_,x] :> -spin[n,z] Sin[b] Sin[p] + spin[n,x] (Cos[b] Sin[p]^2 + Cos[p]^2) +
        spin[n,y] Sin[b/2]^2 Sin[2p] /; MemberQ[spins,n],                         (*Eq. 36*)
    spin[n_,y] :> spin[n,z] Sin[b] Cos[p] + spin[n,x] Sin[b/2]^2 Sin[2p] +
        spin[n,y] (Cos[b] Cos[p]^2 + Sin[p]^2) /; MemberQ[spins,n]                (*Eq. 37*)
    } /. {Cos[-u_] :> Cos[u], Sin[-u_] :> -Sin[u]}

receiver[p_,spins_:allspins][sigma_] := Block[{r,c},
    WriteString["stdout","receiver[",InputForm[p],",",spins,"]"];
    r = Timing[normalize[receivers[p,spins][sigma]]];
    c = r[[1]] /. Second->1; Print[" "]; Print[" (",terms[r[[2]]]," terms, ",N[c,3]," s CPU time)"]; r[[2]]]
receivers[ p_List,spins_][sigma_List] := Block[{i},
    Sum[receivers[p[[i]],spins][sigma[[i]]],{i,1,Length[p]}]/Length[p]]
receivers[ p_,spins_][sigma_List] := Block[{i},
    Sum[receivers[p,spins][sigma[[i]]],{i,1,Length[sigma]}]/Length[sigma]]
receivers[ x,spins_][sigma_] := sigma
receivers[-x,spins_][sigma_] := sigma // pulses[180,z,spins]
receivers[ y,spins_][sigma_] := sigma // pulses[ 90,z,spins]
receivers[-y,spins_][sigma_] := sigma // pulses[-90,z,spins]
receivers[p_,spins_][sigma_] := sigma // pulses[ p,z,spins]

transverse[k_] := MemberQ[{x,y},k]
tfree[u_] := !MatchQ[u,a_. spin[_,?transverse]]
observable[sigma_] := observable[allspins][sigma] /; !FreeQ[sigma,spin[_,_]] || sigma==0
observable[spins_][sigma_] := Block[{r,c},WriteString["stdout","observable[",spins,"]"];
    r = Timing[normalize[observables[spins][sigma]]];
    c = r[[1]] /. Second->1; Print[" "]; Print[" (",terms[r[[2]]]," terms, ",N[c,3]," s CPU time)"]; r[[2]]]
observables[spins_][0] := 0
observables[spins_][sigma_List]:=Block[{i},
    Table[Apply[Plus,observables[spins][sigma[[i]]]],{i,1,Length[sigma]}]]
observables[spins_][sigma_] := Block[{detected,qloc},detected[n_] := MemberQ[spins,n];
    Cases[Apply[List,1+normalize[qloc sigma]],a_?tfree spin[n_?detected,k_?transverse]]
    /. qloc->1 // trigsimplify]

spinlist[sigma_] := sigma // If[Head[#]===Plus,Apply[List,#], {#}]& //
    Map[If[Head[#]===Times,Apply[List,#],{#}]&,#]& //
    Map[Cases[#,spin[_,_]]&,#]& // Map[(# /. List->Times)&,#]& // Union
normalize[sigma_List] := Map[normalize,sigma]
normalize[sigma_] := Block[{e,p}, e=Expand[sigma]; p=spinlist[e]; WriteString["stdout","."];
    (Map[Coefficient[e,#]&,p] /. spin[_,_] -> 0).p]

trigsimplify[u_] := Expand[u] //. {
    c_. Sin[a_]^2 + c_. Cos[a_]^2 :> c,
    c_. Cos[a_]^2 + d_. Sin[a_]^2 :> c Cos[2a] /; c+d==0,
    Sin[a_] Cos[a_] :> 1/2 Sin[2 a],
    c_. Sin[a_] Cos[b_] + c_. Cos[a_] Sin[b_] :> c Sin[a+b],
    c_. Sin[a_] Cos[b_] + d_. Cos[a_] Sin[b_] :> c Sin[a-b] /; c+d == 0,
    c_. Cos[a_] Cos[b_] + d_. Sin[a_] Sin[b_] :> c Cos[a+b] /; c+d == 0,
    c_. Cos[a_] Cos[b_] + c_. Sin[a_] Sin[b_] :> c Cos[a-b]}
simplifysigma[sigma_List] := Map[simplifysigma,sigma]
simplifysigma[sigma_] := Apply[Plus,trigsimplify[Apply[List,1+normalize[sigma]]]]-1

nucleus[_] = 1
terms[sigma_List] := Block[{i},Sum[terms[sigma[[i]]],{i,1,Length[sigma]}]]
terms[sigma_] := Length[1+sigma]-1
spinfree[u_] := FreeQ[u,spin[_,_]]
spinop[u_] := !FreeQ[u,spin[_,_]]
times[u_] := FreeQ[u,j[_,_]] && FreeQ[u,w[_]] && FreeQ[u,_Integer] &&
    FreeQ[u,_Real] && FreeQ[u,Pi]
sort[sigma_] := (Print[TableForm[Complement[Apply[List,1 + Apply[Plus,sigma /. w[n_]->Aw[n]] //. {
    Sin[a_?times b_] :> AA[a,b,Sin], Cos[a_?times b_] :> AA[a,b,Cos],
    a_Times + b_Times :> Select[a,spinfree] (Select[a,spinop] + Select[b,spinop]) /;
        Select[a,spinfree]==Select[b,spinfree]},{1}]]];
    sigma;)
Format[AA[a_,b_Plus,c_]] := SequenceForm[c,"[(",b,") ",a,"]"]
Format[AA[a_,1,c_]] := SequenceForm[c,"[",a,"]"]
Format[AA[a_,b_,c_]] := SequenceForm[c,"[",b," ",a,"]"]
Format[Aw[n_]] := SequenceForm["w",n]
Format[w[n_]] := SequenceForm["w",n]
Format[j[n1_,n2_]] := SequenceForm["J",n1,n2]
Format[spin[n_,k_]] := SequenceForm[nucleus[n],n,k]
```

FIG. 1. Complete *Mathematica* code of the POMA implementation of the product-operator formalism, which is functional with Versions 1.2 and 2.0 of *Mathematica,* albeit with about 2.5 times reduced speed in Version 2.0. The conventions of Sørensen *et al.* (*1*) were used, except that the normalization factors for the product operators were dropped. Equation numbers, e.g., (*Eq. 13*), refer to equations in Sørensen *et al.* (*1*).

$$\textbf{observable}[\{k, l, \ldots\}],$$

where $\{k, l, \ldots\}$ denotes the set of transverse spins that are of interest during the detection period; by default, this includes all spins. Finally, the standard alphabetical ordering of terms used by *Mathematica* can be changed into the convention introduced by Sørensen *et al.* (*1*), i.e., listing in the order "chemical-shift terms–scalar coupling terms–spin operators," by an additional command, **sort**, which rearranges the observable terms accordingly.

In the simulation of a pulsed NMR experiment, the *Mathematica* commands are combined in the order of the pulse sequence by using the postfix operator "//" (in *Mathematica*, $x//f$ has the meaning "apply the operator f to the expression x"; it is equivalent to $f[x]$). For example, the *Mathematica* formulation of the pulse sequence

$$\beta_1\text{—}t_1\text{—}\beta_2\text{—}\cdots\text{—}\beta_n\text{—acquisition}$$

is

pulse[β_1, . . .]//**delay**[t_1, . . .]//**pulse**[β_2, . . .]//· · ·//
pulse[β_n, . . .]//**receiver**[· · ·].

In general, a pulse sequence is applied to the thermal equilibrium state of the spin system, i.e., I_{kz}, or, in *Mathematica* input form, **spin**[k, z], and at the end one retains only the observable terms by using the aforementioned **observable** operator.

As a simple example, we have simulated the 3QF-COSY experiment (*6*),

$$90^\circ_\phi\text{—}t_1\text{—}90^\circ_\phi\text{—}90^\circ_x\text{—acquisition}_\psi,$$

with the phase cycle $\phi = 0, \pi/3, 2\pi/3, \pi, 4\pi/3, 5\pi/3$; $\psi = 3(x, -x)$. In *Mathematica,* this experiment is represented by the sequence of commands

```
spin[1, z]
  //pulse[90, {0, Pi/3, 2Pi/3, Pi, 4Pi/3, 5Pi/3}]//
  delay[t1, {{1, 2}, {1, 3}}]//
  pulse[90, {0, Pi/3, 2Pi/3, 3Pi/3, 4Pi/3, 5Pi/3}]//
  pulse[90, {x, x, x, x, x, x}]//
  receiver[{x, -x, x, -x, x, -x}]//
  observable//Simplify//sort
```

The experiment starts from thermal equilibrium, all pulses are nonselective 90° pulses, and during the delay t_1, spin 1 is scalar coupled to the spins 2 and 3, but spins 2 and 3 are not mutually scalar coupled. **Simplify** is a built-in *Mathematica* command for algebraic simplifications (*4*). The result for this pulse sequence, obtained after 12 seconds of CPU time on a Sun-4 computer with *Mathematica* Version 1.2 and written in *Mathematica* output format is

```
-(Sin[w1 t1]Sin[Pi J12 t1]Sin[Pi J13 t1]
        × (I1z I2z I3x + I1z I2x I3z + I1x I2z I3z)).
```

In the convention of Sørensen *et al.* (*1*), this corresponds to

$$-\sin(\omega_1 t_1)\sin(\pi J_{12} t_1)\sin(\pi J_{13} t_1) \times (I_{1z}I_{2z}I_{3x} + I_{1z}I_{2x}I_{3z} + I_{1x}I_{2z}I_{3z}).$$

Even though the final result is simple, 124 terms had to be handled after the second pulse, which would make a manual derivation of this result rather cumbersome.

ACKNOWLEDGMENTS

We thank Dr. G. Wider and T. Szyperski for helpful discussions. Financial support from the Schweizerischer Nationalfonds Project 31.32033.91 is gratefully acknowledged.

REFERENCES

1. O. W. Sørensen, G. M. Eich, M. H. Levitt, G. Bodenhausen, and R. R. Ernst, *Prog. NMR Spectrosc.* **16,** 163 (1983).
2. R. R. Ernst, G. Bodenhausen, and A. Wokaun, "Principles of Nuclear Magnetic Resonance in One and Two Dimensions," Clarendon Press, Oxford, 1987.
3. H. Kessler, M. Gehrke, and C. Griesinger, *Angew. Chem. Int. Ed. Engl.* **27,** 490 (1988).
4. S. Wolfram, "Mathematica. A System for Doing Mathematics by Computer," Addison-Wesley, Redwood City, California, 1988.
5. J. W. Shriver, *J. Magn. Reson.* **94,** 612 (1991).
6. N. Müller, R. R. Ernst, and K. Wüthrich, *J. Am. Chem. Soc.* **108,** 6482 (1986).

JOURNAL OF MAGNETIC RESONANCE, Series A **105**, 328–329 (1993)

CORRECTIONS AND ADDITIONS

Volume **96,** Number 3 (1992), in the article "Suppression of Cross-Relaxation Effects in TOCSY Spectra via a Modified DIPSI-2 Mixing Sequence," by John Cavanagh and Mark Rance, pages 670–678: The multipulse sequence in lines 30–31 on page 671 has a typesetting error. The correct sequence is

$$R = 180°—\Delta—140°—\overline{320}°—\Delta—\overline{90}°—270°—\Delta—20°—\overline{200}°—\Delta—\overline{85}°—30°—$$
$$\overline{125}°—\Delta—\overline{120}°—300°—\Delta—75°—\overline{255}°—\Delta—\overline{10}°—190°—\Delta—180°—\Delta.$$

The fourth pulse from the end should be $\overline{255}°$ instead of $\overline{225}°$.

JOHN CAVANAGH
MARK RANCE

Volume **101,** Number 1 (1993), Series A, in the Communication "POMA: A Complete *Mathematica* Implementation of the NMR Product-Operator Formalism," by Peter Güntert, Niklaus Schaefer, Gottfried Otting, and Kurt Wüthrich, pages 103–105: Application of the program POMA in different laboratories revealed three typographical errors, which we would like to communicate to potential users of the program. (i) On page 104, the program code of POMA as listed in Fig. 1 uses a sign for the receiver phase angle that is opposite to the convention of G. Bodenhausen, H. Kogler, and R. R. Ernst (*J. Magn. Reson.* **58,** 370–388, 1984). To conform with this convention, lines 7–9 in the right column of Fig. 1 should read:

```
receivers[y,spins_][sigma_] := sigma//pulses[-90,z,spins]
receivers[-y,spins_][sigma_] := sigma//pulses[90,z,spins]
receivers[p_,spins_][sigma_] := sigma//pulses[-p,z,spins]
```

(ii) To be valid input for *Mathematica,* lines 14–15 in the left column on p. 105 should read:

```
spin[1,z]//
    pulse[90,{0,Pi/3,2Pi/3,Pi,4Pi/3,5Pi/3}]//
```

(iii) Since the normalization factors for product operators of O. W. Sørensen, G. M. Eich, M. H. Levitt, G. Bodenhausen, and R. R. Ernst (*Prog. NMR Spectrosc.* **16,** 163, 1983) are not used in POMA, lines 29–30 in the left column on p. 105 should read

```
-(Sin[w1 t1] Sin[Pi J12 t1] Sin[Pi J13 t1]
    × (I1z I2z I3x + I1z I2x I3z + I1x I2z I3z))/4.
```

KURT WÜTHRICH

Volume **102,** Number 2 (1993), Series A, in the article "Design of Phase-Distortionless Broadband Composite-Pulse Sequences via Simulated Annealing," by N. Sunitha Bai, M. Ramakrishna, and R. Ramachandran, pages 235–240: