Defeating the Matrix

Ilya Kuprov,
School of Chemistry,
University of Southampton

Abstract

These are personal recollections and musings, written for the 50th Anniversary of the *Journal of Magnetic Resonance*. They are distilled from twenty years of writing simulation code, and filtered through hindsight and sarcastic intransigence. To me, the biggest recent achievements of the magnetic resonance community in the field of theory and simulation have been the successful war on the exponential scaling, the powerful and general simulation software, and the return to elegant notation. It appears that our future will be defined by computers. Three aspects are pertinent: simulation as the experiment is designed, optimal control as the experiment proceeds, and machine learning at the data processing stage.

Prologue

A project student poked his head through the door, *"Hey, Ilya – I think I am done with the HOESY simulation for cyprinol."* I blinked and looked at him, *"What do you mean, done?"* The student opened his laptop, *"Well, I just copied in all the shifts and couplings and coordinates, and the instructions are pretty neat – that's the spectrum."* 27 carbons and 48 protons, I thought… strongly coupled, HOESY needs full Redfield superoperator. You just typed the parameters in, you sweet summer child; twenty seconds later, you got the spectrum simulated – on your laptop. Another student walked in. *"What's up, mate?"* – *"Do you have PCM solvent in that cupboard? We used it for calculations, but I can't find a bottle anywhere…"*

I chuckled and poured myself some coffee. It has not always been thus. A vivid memory from the first week of my PhD project was watching one of Peter Hore's postdocs printing out coupling operators for a seven-spin system… on poster paper. And then sitting there, circling non-zeroes with a pencil, to hand-code multiplications in Fortran. Two decades prior, Jack Freed's group clocked up \$7,000 in supercomputing charges to simulate slow-motion EPR spectra for two spins [1]. For a long time, the biggest objection to superoperator methods [2] was *"it will crash the computer"*.

Here's the big problem: a spin has two states, two spins have four, three spins have eight, and so on – it goes as 2^k. In a simulation of a k-spin system, the operators will be matrices of dimension 2^k. If we must accurately deal with an ensemble of such systems, the dimension rises to 4^k, more if particles like ^{14}N are present – they have a higher spin. A double-precision complex number requires 16 bytes of memory; this puts naïve storage requirements for a ten-spin ensemble simulation into terabytes, and wall clock time into centuries. This was mostly solved by 2010 [3,4], but the story begins much earlier.

The quest for beauty and efficiency

Humans rightly equate beautiful with desirable, and throughout the history of natural sciences it was always the elegant theory that was the right one. At some point in 1960-es, we lost track – the mismatch between the meagre capacity of early computers and the extraordinary skill of the theorists begat monstrous expressions that filled whole pages of *JMR* and *JCP* (Figure 1). This was particularly visible in the angular momentum theory – to pack the calculation into the tiny quantities of computer memory that were practically available, it was really necessary to go through that horror.

Both the notation [5] and the computational complexity scaling [6] required extensive training. Peter summarised it neatly in the coffee room one day: *"Ilya, three people will understand your paper. Do you really want to produce something like that?"* – it became clear that something must be done about the sort of thing we are looking at in Figure 1. Eliminating that is what *modern* theory and computing is about. We are hardly ever doing anything truly new, but we are digging ourselves out of sums of nested integrals over nested commutators, and rediscovering elegance, flexibility, and power of high-level analytical mathematics because now the computer is good – we can hand things over to that computer before they turn ugly. So, what has changed? Several things, and very slowly.

We moved to higher levels of abstraction and better notation – first Abragam [7], then the three wise men [8], and then *Matlab* got everyone accustomed to the operator notation. Higher-level functions, such as matrix exponentials and logarithms, became elementary. A good example is the expression for the exact rotating frame Hamiltonian [9] used by *Spinach*:

$$\mathbf{H}_1^{(\mathrm{R})} = \frac{i}{T}\ln_{(\mathrm{P})}\left[e^{-i(\mathbf{H}_0+\mathbf{H}_1)T}\right] \tag{1}$$

where T is the period of $\exp(-i\mathbf{H}_0 t)$ and $\ln_{(\mathrm{P})}$ is the principal value of the matrix logarithm. A stable numerical method for the matrix log only appeared around the turn of this century [10], at which point Equation (1) replaced a horrifying and often divergent series over nested commutator integrals [11].

Outside undergraduate classrooms, we are using energy level diagrams less and less – unlike those, the group theory language (product operators and such) does not actually depend on the number of spins in the system. At the same time, computers eliminated a lot of analytical spaghetti, like the explicit formulae for Wigner *D* functions. Don't get me started… here's the group theory level expression for the entire *D* matrix *via* Pauli matrices of the same rank, trivially computable numerically:

$$\mathfrak{D}(\alpha,\beta,\gamma) = \exp[-i\mathbf{S}_\mathrm{Z}\alpha]\exp[-i\mathbf{S}_\mathrm{Y}\beta]\exp[-i\mathbf{S}_\mathrm{Z}\gamma] \tag{2}$$

and here's the explicit formula for just one element, from that paperback monstrosity [12]:

$$\begin{aligned}\mathfrak{D}^{(l)}_{m,m'}(\alpha,\beta,\gamma) &= (-1)^{l-m'}\sqrt{(l+m)!(l-m)!(l+m')!(l-m')!}\times \\ &\times\sum_k \frac{(-1)^k e^{-im\alpha}\left(\cos[\beta/2]\right)^{m+m'+2k}\left(\sin[\beta/2]\right)^{2l-m-m'-2k} e^{-im'\gamma}}{k!(l-m-k)!(l-m'-k)!(m+m'+k)!}\end{aligned} \tag{3}$$

where even working out the summation index range is not trivial. As computers are becoming more powerful, such things are drifting into history – once we realise that Equation (2) is directly and easily computable and differentiable, its pristine simplicity is reclaimed from the jaws of Equation (3).

We finally got the bloody rotations shaken down into something that resembles order [13]… I still have an irrational fear of Euler angles, and I am infinitely grateful to David Siminovitch [14] for explaining why. Interaction tensor reporting conventions are also now agreed upon [15]. There must be a separate energy level in helium for Ken Stevens, but even his appalling operators have by now been tamed [16] and contained inside well designed software packages [17] that provide two-way translation into the beautiful and neat irreducible spherical tensors.

We began making approximations at the level of the state space – nobody in magnetic resonance needs fifty-quantum coherences, and the people designing perpetual motion machines of the third kind [18] will never defeat friction… erm, "decoherence". Once the description moves to the level of Equations (1) and (2), one realises that faithful representations of a Lie algebra are not the only ones possible – corners may be cut by neglecting unimportant and unpopulated states [3,4]. It was this that eventually vanquished the exponential scaling in liquid state NMR [19]. In solids and elsewhere, the work is ongoing [20].

We started appreciating that every operation has a cost, and eliminating expensive ones. Look at the Frobenius scalar product in the density operator formalism:

$$\mathrm{Tr}\left[\mathbf{A}^{\dagger}\mathbf{B}\right]=\sum_{nk}a_{kn}^{*}b_{kn}=\mathbf{A}^{*}\odot\mathbf{B} \tag{4}$$

where $\odot$ stands for element-wise multiplication followed by total summation. The expression on the left has cubic scaling with matrix dimension; the expression on the right has quadratic scaling. Three orders of magnitude faster for a ten-spin system! *Spinach* [21] uses dozens of such shortcuts.

Decent programming languages and libraries arrived – *Matlab* is a masterpiece. And the computers of course. *NVidia* gets a special mention: a single Titan V card does ~10^{13} multiplications per second – that is an outrageous number! Faced with the ready availability of such things, theorists became more cautious about publishing incomputable (generalised cumulant expansion gets the prize) or unscalable (most matrix factorisations) propositions. Universities became aware of the need to maintain good facilities – the two departments that work well at my place are Gardening and Supercomputing.

The future is with three-letter expressions that a machine can take care of: if plus and minus are elementary operations, then so is adiabatic elimination! Make an operator, call it your favourite letter, and I'll get *Spinach* to generate it for you. The transition to numerics should be where creative thinking stops and computational slogging begins. Realistically these days, that's the level of the equation of motion and the basis set. Leave the rest to the computer.

The art of computing

Chemists rarely appreciate that 64 bits is not a lot of accuracy – just sixteen decimal places. There are expressions in spin dynamics that are not even computable in 64-bit arithmetic. This story goes back to 2009, when a bug was spotted simultaneously in *Spinach* and *EasySpin* – both packages had correctly programmed functions which nonetheless failed to compute Clebsch-Gordan coefficients of high ranks. The reason was that CG coefficients are full of factorials [12]. The largest factorial that fits into a 64-bit representation is $170! \approx 7.26\times10^{306}$, with a lovely accuracy of plus or minus 1.6×10^{291}. To some extent, this could be mitigated by using logs of factorials, but that still left one unavoidable summation operation at the very end, where astronomically large numbers were subtracted to leave the result that is, by definition, inside [–1,+1]. That simply wasn't possible in 64-bit arithmetic. About a week of joint head scratching between Oxford and Seattle yielded a page of carefully orchestrated calls to arbitrary precision arithmetic functions that was returning accurate answers for all reasonable ranks.

There are many such subtleties. Consider memory access. The distance between the CPU and the memory banks in a computer is about 10 cm, and the wavefront velocity in copper wire is about 200,000 km/s – requiring a minimum of 1 ns for a round trip. During that nanosecond, a modern CPU with 16 cores, 4 GHz clock frequency and 4 multiplications per cycle, can accomplish 256 multiplications! Instead, it has to sit and wait. It is not *that* bad, of course – there are three levels of caches that are closer electrically than the main RAM, there are pre-fetchers and branch predictors that proactively fetch data before it is needed, but… all of this relies on your memory access being (ideally) linear, or (at least) predictable. If it is not, welcome to a factor-of-200 slowdown. This is not hypothetical: the infinity-norm (maximum absolute row sum) for a Hermitian matrix is equal to its 1-norm (maximum absolute column sum), but depending on how your matrices are stored and compressed (column-wise or row-wise), one operation would be addressing memory linearly, the other randomly. One of the two identical numbers can be an order of magnitude slower to compute. For the same reason, matrix transpose operation is surprisingly expensive.

In any serious simulation software, such caveats are everywhere. Around 2011, Walter Kockenberger was using parts of *Spinach* for his DNP simulations… he mentioned at a conference that he was packing the microwave irradiation stage into a single matrix exponential. My heart skipped a beat: "*What do you mean, single exponential?!"* – the estimates raced through my head: typical hyperfine coupling is MHz, typical irradiation time is minutes, is he exponentiating a matrix with the norm of roughly 10^9? *"Oh, but the answer looks right,"* – Walter continued pleasantly – *"we just published it [22]."* Well, it better damn well be right or I am toast, I thought and reached for my laptop. How the hell did my exponentiation module survive that? Turns out it did: the program diligently estimated the norm, and concluded that it needs to scale the time step down by… 2^{30} and then to square the propagator… thirty times. Thankfully, squaring a unitary operator is unproblematic, and the safety catch I had completely forgotten about had kicked in: when an extreme norm was detected, all internal tolerances were automatically tightened up. In subsequent testing, it turned out that Walter was down to the last two decimal places of numerical accuracy, but the answer was indeed correct. *"Yeah, we just computed it in Spinach"* – the student remarked casually as she was presenting her poster. Good for you, I thought.

The four horsemen of incomputability

This wisdom took decades to crystallise – from the point of view of numerical computing, some operations in magnetic resonance are in the whole separate category of horrible. Every time you use them, a kitten dies. A kitten with big eyes (Figure 2).

1. **Diagonalisation.** We were all brought up with energy level diagrams – get the Hamiltonian, solve the secular equation… this works for a two-spin system. With more spins, two nasty properties of the diagonalisation operation manifest themselves:

    (a) Spin Hamiltonians are always sparse, but their eigenvectors are not – memory and bandwidth will be needed store and move every single one of those $16 \cdot 2^N$ bytes.

(b) Diagonalisation is expensive – O(N^3) operations for a matrix of dimension N. This makes it unaffordable beyond about 14 spins.

This means that diagonalisation is a dirty word in computing. A few papers have the phrase "diagonalisation-free" in the title as their principal selling point [23,24]. Besides, do you really need to know *every energy and every conceivable dynamical mode* in the system? I thought you just wanted the spectrum or the trajectory… that's a lot less information! *Never diagonalise.*

2. **Kronecker product.** Krons appear naturally in physics: ensemble simulations require averaging over all degrees of freedom that are not involved in a particular process. The result is the following structure for the evolution generator terms, for example in metabolic MRI:

$$[\text{space dynamics}]\otimes[\text{reaction kinetics}]\otimes[\text{spin dynamics}] \tag{5}$$

where the space part contains further krons between spatial dimensions and the spin part contains further krons between spins. This object is a hand grenade. Its outside structure is simple: we just wrote it in one line. But an attempt to actually compute the resulting matrix for a realistic spin system in a realistic MRI phantom will overflow the memory.

This may be too much to ask at this point in history, and we are only starting to explore this in *Spinach*, but… *never open the krons* in Equation (5). You don't need to: all relevant elementary operations may be carried out by having the closed direct product structure act on vectors [25].

3. **Exact norm.** To satisfy the Nyquist sampling condition – two points per period of the fastest oscillation – we need to know the largest frequency in the system. It is called "2-norm", and… it's the largest singular value. You can probably hear the beast of diagonalisation howling in the distance. Fear not, there are actually faster methods for extreme singular values [26], but they are still very expensive relative to the cost of a matrix-vector product. The secret here is to realise that *an upper bound* on the 2-norm is actually sufficient – who cares of we oversample the trajectory a bit. Upper bounds are massively cheaper: Hager's famous estimate [27] typically needs five matrix-vector multiplications, meaning that the cost is negligible relative to the other mathematics that is going on. The sage advice here: *avoid operations that cost more than a few matrix-vector products.*

4. **Matrix exponential.** We have yet to vanquish this one. There are places in magnetic resonance where it is unavoidable, although relations like the following offer hope

$$\mathbf{x}(t+\Delta t)=\exp\left[-i\mathbf{L}\Delta t\right]\mathbf{x}(t)=\sum_{n=0}^{\infty}\frac{(-i\Delta t)^n}{n!}\mathbf{L}\Big(\ldots\Big(\mathbf{L}\big(\mathbf{L}\mathbf{x}(t)\big)\Big)\Big). \tag{6}$$

Note that there are only matrix-vector products on the right hand side, and the time step may be subdivided to ensure rapid convergence of the sum.

The worst possible way to exponentiate an actual matrix is *via* diagonalisation, and most of the other "dubious ways" [28] are bad in one way or another in finite precision arithmetic. Chebyshev diverges for dissipative systems, Newton has a finite convergence radius, Padé loses sparsity, all three are too memory-hungry, *etc.* The only completely bullet-proof, sparsity-preserving, and low-memory method I have seen is Taylor with scaling and squaring. Use it if you must… you will need a norm estimate, for which see above. However, *never exponentiate the generator if you only plan to take one time step* – use Equation (6) or something similar [29].

So here's your straight and narrow path: do not open krons, do not diagonalise, use cheap norm estimators, and do not exponentiate matrices – unless you absolutely have to. This is not in any magnetic resonance book. I am writing one (thank you, whoever you are, for sending *Springer* in my general direction), but it will take till the end of 2020 or thereabouts.

Where next?

I am tempted to say that we are done – magnetic resonance simulations have always been a technical matter of persuading the well-known equations of motion to get themselves computed, exactly or approximately, and we are there: just read the book and use the software. But that is not the case, of course. Plenty of new opportunities have recently emerged, and two of them deserve a mention.

1. **Optimal control theory.** Optimal control pulses [30] are a generalisation of composite pulses [31], although Malcolm still says that I am computing too much and thinking too little. He is probably right, but I think the move from analytical to numerical modelling is inevitable – the spin systems and the objectives we are dealing with are becoming more and more complicated. The early successes of optimal control theory involve ultra-broadband excitation, highly efficient coherence transfer, B_0 and B_1 inhomogeneity resilience, highly selective excitation, and effective Hamiltonian design [32].

   In situations when difficult experiments must be performed on imperfect hardware (*i.e.* always), the GRAPE algorithm [33] is a godsend. I work with laser spectroscopists, who are less spoiled with slow timescales than we are. When they see a 5-fold improvement in performance, they simply say *"whatever that was, we want more of it"*. For complicated, demanding, and tightly constrained experiment design, optimal control theory is the way forward.

2. **Deep neural networks.** Never in the history of science has numerical regression been so famo… erm, I meant to say "machine learning", of course. Silly me. The biggest question is why does it work so well, and… *how* does it actually work. We published some neural networks recently [34], but we still haven't got a damn clue about what happens inside, and it's not for lack of trying! Whosoever finds out how those things solve [34] what is technically an insoluble problem, will have my eternal gratitude – as well as a Fields Medal, I suppose.

Overall, it is my humble opinion that the continuing exponential improvement in computing power is the biggest scientific gold mine of our time. It is already defining, and will continue to define the future of magnetic resonance. Do sign up for that theory and programming course you've been planning to attend – now is the right time.

References

[1] J.H. Freed, G.V. Bruno, C.F. Polnaszek, Electron spin resonance line shapes and saturation in the slow motional region, The Journal of Physical Chemistry, 75 (1971) 3385-3399.

[2] H. Primas, C.N. Banwell, On the analysis of high-resolution nuclear magnetic resonance spectra, Molecular Physics, 6 (1963) 225-256.

[3] M.C. Butler, J.-N. Dumez, L. Emsley, Dynamics of large nuclear-spin systems from low-order correlations in Liouville space, Chemical Physics Letters, 477 (2009) 377-381.

[4] I. Kuprov, N. Wagner-Rundell, P. Hore, Polynomially scaling spin dynamics simulation algorithm based on adaptive state-space restriction, Journal of Magnetic Resonance, 189 (2007) 241-250.

[5] I. Scholz, B.H. Meier, M. Ernst, Operator-based triple-mode Floquet theory in solid-state NMR, The Journal of chemical physics, 127 (2007) 204504.

[6] Z. Tošner, R. Andersen, B. Stevensson, M. Edén, N.C. Nielsen, T. Vosegaard, Computer-intensive simulation of solid-state NMR experiments using SIMPSON, Journal of Magnetic Resonance, 246 (2014) 79-93.

[7] A. Abragam, A. Abragam, The principles of nuclear magnetism, Oxford university press, 1961.

[8] R.R. Ernst, G. Bodenhausen, A. Wokaun, Principles of nuclear magnetic resonance in one and two dimensions, Clarendon Press Oxford, 1987.

[9] D. Goodwin, I. Kuprov, Auxiliary matrix formalism for interaction representation transformations, optimal control, and spin relaxation theories, The Journal of chemical physics, 143 (2015) 084113.

[10] N.J. Higham, Accuracy and stability of numerical algorithms, Siam, 2002.

[11] U. Haeberlen, J. Waugh, Coherent averaging effects in magnetic resonance, Physical Review, 175 (1968) 453.

[12] D.A. Varshalovich, A.N. Moskalev, V.K.m. Khersonskii, Quantum theory of angular momentum, World Scientific, 1988.

[13] L.J. Mueller, Tensors and rotations in NMR, Concepts in Magnetic Resonance Part A, 38 (2011) 221-235.

[14] M. Siemens, J. Hancock, D. Siminovitch, Beyond Euler angles: Exploiting the angle–axis parametrization in a multipole expansion of the rotation operator, Solid state nuclear magnetic resonance, 31 (2007) 35-54.

[15] R.K. Harris, E.D. Becker, S.M. Cabral De Menezes, P. Granger, R.E. Hoffman, K.W. Zilm, Further conventions for NMR shielding and chemical shifts (IUPAC recommendations 2008), Solid State Nuclear Magnetic Resonance, 33 (2008) 41-56.

[16] C. Rudowicz, C. Chung, The generalization of the extended Stevens operators to higher ranks and spins, and a systematic review of the tables of the tensor operators and their matrix elements, Journal of Physics: Condensed Matter, 16 (2004) 5825.

[17] S. Stoll, A. Schweiger, EasySpin, a comprehensive software package for spectral simulation and analysis in EPR, Journal of magnetic resonance, 178 (2006) 42-55.

[18] A.M. Tyryshkin, J.J.L. Morton, S.C. Benjamin, A. Ardavan, G.A.D. Briggs, J.W. Ager, S.A. Lyon, Coherence of spin qubits in silicon, Journal of Physics: Condensed Matter, 18 (2006) S783-S794.

[19] L.J. Edwards, D. Savostyanov, Z. Welderufael, D. Lee, I. Kuprov, Quantum mechanical NMR simulation algorithm for protein-size spin systems, Journal of Magnetic Resonance, 243 (2014) 107-113.

[20] J.-N. Dumez, M.C. Butler, L. Emsley, Numerical simulation of free evolution in solid-state nuclear magnetic resonance using low-order correlations in Liouville space, The Journal of chemical physics, 133 (2010) 224501.

[21] H. Hogben, M. Krzystyniak, G. Charnock, P. Hore, I. Kuprov, Spinach–a software library for simulation of spin dynamics in large spin systems, Journal of Magnetic Resonance, 208 (2011) 179-194.

[22] A. Karabanov, G. Kwiatkowski, W. Köckenberger, Quantum mechanical simulation of cross effect DNP using Krylov–Bogolyubov averaging, Applied magnetic resonance, 43 (2012) 43-58.

[23] Z. Szekeres, P.G. Mezey, P.R. Surján, Diagonalization-free initial guess to SCF calculations for large molecules, Chemical physics letters, 424 (2006) 420-424.

[24] I. Kuprov, Diagonalization-free implementation of spin relaxation theory for large spin systems, Journal of Magnetic Resonance, 209 (2011) 31-38.

[25] A.J. Allami, M.G. Concilio, P. Lally, I. Kuprov, Quantum mechanical MRI simulations: solving the matrix dimension problem, submitted, (2019).

[26] G.W. Stewart, A Krylov-Schur algorithm for large eigenproblems, SIAM Journal on Matrix Analysis and Applications, 23 (2002) 601-614.

[27] W.W. Hager, Condition estimates, SIAM Journal on scientific and statistical computing, 5 (1984) 311-316.

[28] C. Moler, C. Van Loan, Nineteen dubious ways to compute the exponential of a matrix, SIAM review, 20 (1978) 801-836.

[29] R.B. Sidje, Expokit: A software package for computing matrix exponentials, ACM Transactions on Mathematical Software (TOMS), 24 (1998) 130-156.

[30] N. Khaneja, T. Reiss, C. Kehlet, T. Schulte-Herbrüggen, S.J. Glaser, Optimal control of coupled spin dynamics: design of NMR pulse sequences by gradient ascent algorithms, Journal of magnetic resonance, 172 (2005) 296-305.

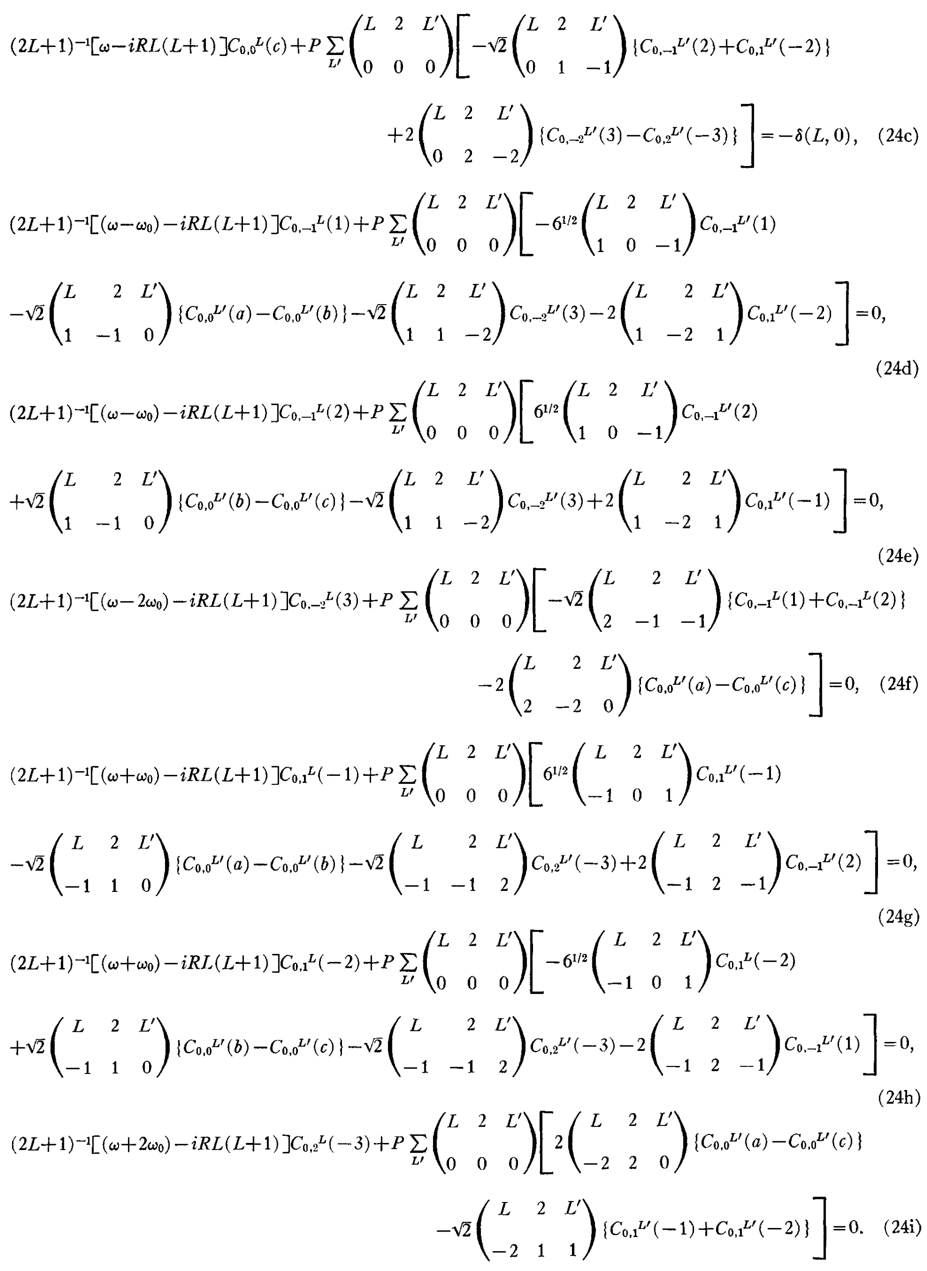
[31] M.H. Levitt, Composite pulses, eMagRes, (2007).

[32] N.C. Nielsen, C. Kehlet, S.J. Glaser, N. Khaneja, Optimal control methods in NMR spectroscopy, eMagRes, (2007).

[33] P. De Fouquieres, S. Schirmer, S. Glaser, I. Kuprov, Second order gradient ascent pulse engineering, Journal of Magnetic Resonance, 212 (2011) 412-417.

[34] S.G. Worswick, J.A. Spencer, G. Jeschke, I. Kuprov, Deep neural network processing of DEER data, Science advances, 4 (2018) eaat5218.

[35] J.H. Freed, G.V. Bruno, C. Polnaszek, ESR line shapes for triplets undergoing slow rotational reorientation, The Journal of Chemical Physics, 55 (1971) 5270-5281.



$$
\begin{aligned}
&(2L+1)^{-1}[\omega-iRL(L+1)]C_{0,0}{}^{L}(c)+P\sum_{L'}\begin{pmatrix} L & 2 & L' \\ 0 & 0 & 0 \end{pmatrix}\Bigg[-\sqrt{2}\begin{pmatrix} L & 2 & L' \\ 0 & 1 & -1 \end{pmatrix}\{C_{0,-1}{}^{L'}(2)+C_{0,1}{}^{L'}(-2)\} \\
&\qquad +2\begin{pmatrix} L & 2 & L' \\ 0 & 2 & -2 \end{pmatrix}\{C_{0,-2}{}^{L'}(3)-C_{0,2}{}^{L'}(-3)\}\Bigg]=-\delta(L,0),
\end{aligned}
\tag{24c}
$$

$$
\begin{aligned}
&(2L+1)^{-1}[(\omega-\omega_0)-iRL(L+1)]C_{0,-1}{}^{L}(1)+P\sum_{L'}\begin{pmatrix} L & 2 & L' \\ 0 & 0 & 0 \end{pmatrix}\Bigg[-6^{1/2}\begin{pmatrix} L & 2 & L' \\ 1 & 0 & -1 \end{pmatrix}C_{0,-1}{}^{L'}(1) \\
&-\sqrt{2}\begin{pmatrix} L & 2 & L' \\ 1 & -1 & 0 \end{pmatrix}\{C_{0,0}{}^{L'}(a)-C_{0,0}{}^{L'}(b)\}-\sqrt{2}\begin{pmatrix} L & 2 & L' \\ 1 & 1 & -2 \end{pmatrix}C_{0,-2}{}^{L'}(3)-2\begin{pmatrix} L & 2 & L' \\ 1 & -2 & 1 \end{pmatrix}C_{0,1}{}^{L'}(-2)\Bigg]=0,
\end{aligned}
\tag{24d}
$$

$$
\begin{aligned}
&(2L+1)^{-1}[(\omega-\omega_0)-iRL(L+1)]C_{0,-1}{}^{L}(2)+P\sum_{L'}\begin{pmatrix} L & 2 & L' \\ 0 & 0 & 0 \end{pmatrix}\Bigg[6^{1/2}\begin{pmatrix} L & 2 & L' \\ 1 & 0 & -1 \end{pmatrix}C_{0,-1}{}^{L'}(2) \\
&+\sqrt{2}\begin{pmatrix} L & 2 & L' \\ 1 & -1 & 0 \end{pmatrix}\{C_{0,0}{}^{L'}(b)-C_{0,0}{}^{L'}(c)\}-\sqrt{2}\begin{pmatrix} L & 2 & L' \\ 1 & 1 & -2 \end{pmatrix}C_{0,-2}{}^{L'}(3)+2\begin{pmatrix} L & 2 & L' \\ 1 & -2 & 1 \end{pmatrix}C_{0,1}{}^{L'}(-1)\Bigg]=0,
\end{aligned}
\tag{24e}
$$

$$
\begin{aligned}
&(2L+1)^{-1}[(\omega-2\omega_0)-iRL(L+1)]C_{0,-2}{}^{L}(3)+P\sum_{L'}\begin{pmatrix} L & 2 & L' \\ 0 & 0 & 0 \end{pmatrix}\Bigg[-\sqrt{2}\begin{pmatrix} L & 2 & L' \\ 2 & -1 & -1 \end{pmatrix}\{C_{0,-1}{}^{L}(1)+C_{0,-1}{}^{L}(2)\} \\
&\qquad -2\begin{pmatrix} L & 2 & L' \\ 2 & -2 & 0 \end{pmatrix}\{C_{0,0}{}^{L'}(a)-C_{0,0}{}^{L'}(c)\}\Bigg]=0,
\end{aligned}
\tag{24f}
$$

$$
\begin{aligned}
&(2L+1)^{-1}[(\omega+\omega_0)-iRL(L+1)]C_{0,1}{}^{L}(-1)+P\sum_{L'}\begin{pmatrix} L & 2 & L' \\ 0 & 0 & 0 \end{pmatrix}\Bigg[6^{1/2}\begin{pmatrix} L & 2 & L' \\ -1 & 0 & 1 \end{pmatrix}C_{0,1}{}^{L'}(-1) \\
&-\sqrt{2}\begin{pmatrix} L & 2 & L' \\ -1 & 1 & 0 \end{pmatrix}\{C_{0,0}{}^{L'}(a)-C_{0,0}{}^{L'}(b)\}-\sqrt{2}\begin{pmatrix} L & 2 & L' \\ -1 & -1 & 2 \end{pmatrix}C_{0,2}{}^{L'}(-3)+2\begin{pmatrix} L & 2 & L' \\ -1 & 2 & -1 \end{pmatrix}C_{0,-1}{}^{L'}(2)\Bigg]=0,
\end{aligned}
\tag{24g}
$$

$$
\begin{aligned}
&(2L+1)^{-1}[(\omega+\omega_0)-iRL(L+1)]C_{0,1}{}^{L}(-2)+P\sum_{L'}\begin{pmatrix} L & 2 & L' \\ 0 & 0 & 0 \end{pmatrix}\Bigg[-6^{1/2}\begin{pmatrix} L & 2 & L' \\ -1 & 0 & 1 \end{pmatrix}C_{0,1}{}^{L}(-2) \\
&+\sqrt{2}\begin{pmatrix} L & 2 & L' \\ -1 & 1 & 0 \end{pmatrix}\{C_{0,0}{}^{L'}(b)-C_{0,0}{}^{L'}(c)\}-\sqrt{2}\begin{pmatrix} L & 2 & L' \\ -1 & -1 & 2 \end{pmatrix}C_{0,2}{}^{L'}(-3)-2\begin{pmatrix} L & 2 & L' \\ -1 & 2 & -1 \end{pmatrix}C_{0,-1}{}^{L'}(1)\Bigg]=0,
\end{aligned}
\tag{24h}
$$

$$
\begin{aligned}
&(2L+1)^{-1}[(\omega+2\omega_0)-iRL(L+1)]C_{0,2}{}^{L}(-3)+P\sum_{L'}\begin{pmatrix} L & 2 & L' \\ 0 & 0 & 0 \end{pmatrix}\Bigg[2\begin{pmatrix} L & 2 & L' \\ -2 & 2 & 0 \end{pmatrix}\{C_{0,0}{}^{L'}(a)-C_{0,0}{}^{L'}(c)\} \\
&\qquad -\sqrt{2}\begin{pmatrix} L & 2 & L' \\ -2 & 1 & 1 \end{pmatrix}\{C_{0,1}{}^{L'}(-1)+C_{0,1}{}^{L'}(-2)\}\Bigg]=0.
\end{aligned}
\tag{24i}
$$

Figure 1. *A page from a 1971 paper by probably the strongest theorists we have in magnetic resonance [35]. The paper was forty years ahead of its time – the computers that could process the underlying rotation group representations directly in matrix notation only appeared around 2010.*

Figure 2. *The kitten that dies when you diagonalise a Hamiltonian.*