Supporting Information

Towards fully optimised and automated ESR spectroscopy

J.-B. Verstraete, J. R. J. Yong, D. L. Goodwin, W. K. Myers, M. Foroozandeh

Contents

1 Materials and methods

The ESR-POISE dependencies include the `XeprAPI` package, as adapted for Python 3.7 & 3.8 by Sam Schott (`https://github.com/OE-FET/XeprAPI`). In this work, the Bruker BioSpin spectrometer control software Xepr, version 2.8b.5, was used.

Shaped pulse generation and simulation were carried out using mrpypulse package (`https://github.com/foroozandehgroup/mrpypulse`) in Python and MRChirpLab package (`https://github.com/foroozandehgroup/MRChirpLab`) in MATLAB [1].

ESR-POISE comes with three derivative-free optimisation algorithms: the multidimensional search (MDS) method [2], the Nelder-Mead simplex method [3], and BOBYQA (Bound Optimization BY Quadratic Approximation), a trust-region interpolation method [4, 5].

For the DEER experiment, a resonator MD5 was used and for all other examples a resonator MS2.

For statistical analysis of different optimisation algorithms, all optimisations were conducted with five repetitions and in an interleaved fashion to minimise the effect of instrumental drift; however, full grid searches (entries labelled "brute force") were only carried out once due to their long running time.

All experiments used a video gain of 30 dB (except for DEER, at 48 dB), a receiver bandwidth of 1 GHz and a signal recording time resolution of 0.5 ns.

For the signal phasing, flip angle and field tweak optimisation studies, the echo was recorded with 256 points and 64 averages per scan. A 16-step phase cycle was used. The MPFU phasing optimisation used the same setup, but with 4 averages per scan.

The DEER sequence was set up with Gaussian pulses [6] as observer pulses of lengths 34 ns (65% amplitude) and 100 ns (100% amplitude), $\tau_1 = 200$ ns, $\tau_2 = 1000$ ns, and a HS pump pulse with the following properties (see below for shape equation): duration $\tau_\mathrm{p} = 100$ ns, bandwidth $\Delta F = 350$ MHz and time resolution $t_\mathrm{res} = 0.625$ ns. The DEER optimisation was conducted by recording 2 points of the traces with 1024 averages per scan. The DEER full traces were recorded with 1024 point, 256 average per scans, an 8-step phase cycle and 8 τ_1-averaging steps of 8 ns. MATLAB was used for the summing and phasing of τ-averaged DEER data. The Python package DEERLab[7] was used to subtract background signal from the DEER traces.

For the MPFU phasing optimisation study, the echo was recorded with 256 points and 4 averages per scan. The study was conducted under the same conditions (resonator, sample, temperature) as the DEER optimisation.

Table S1: Parameters of individual pulses in CHORUS experiment, namely the adiabaticity factor (Q), duration (τ_p), and time-bandwidth product (TBP). All pulses have a bandwidth (ΔF) of 350 MHz, a time resolution (t_res) of 0.625 ns, and a smoothing factor of 12.5% (quarter-sine smoothing).

Parameter	**Chirped pulse 1**	**Chirped pulse 2**	**Chirped pulse 3**
τ_p (ns)	80	280	240
TBP (ns)	28	98	84
Q	0.441	4	4

The CHORUS sequence was 640 ns long (full pulse details are given in Table S1). An additional delay of 80 ns was inserted between the two refocusing pulses and at the end of the sequence to account for the instrument dead time. The sequence was run with a 64-step phase cycle, as described in our previous work.[8]

For the optimisation of CHORUS pulse amplitudes alone, the echo was recorded with 128 points and 64 averages per scan. On the other hand, for the CHORUS pulse compensation optimisation, the echo was recorded with 128 points and 32 averages per scan. The echo was cut at its middle point (where the maximum in intensity occurs) to avoid first-order phase errors; to make this step more precise, the time resolution of the echo was artificially increased using spline interpolation. The amplitude of the first point was subsequently halved to avoid added baseline error. The spectrum was then obtained by Fourier transformation and zero-filling to 8 times more points than the time-domain data. Zeroth-order phase correction was performed by searching for the phase correction ϕ_0 (going from $-180°$ to $180°$ in steps of $0.25°$) which maximised the intensity of the spectrum.. Finally, the spectrum was interpolated along the x-axis of the reversed field sweep.

The field-sweep data used as part of the cost function to achieve compensated CHORUS sequence was recorded with a Hahn echo using low intensity Gaussian pulses of lengths 50 ns and 100 ns, over a 100 G, with 1024 points, 256 averages per scan, and a video gain of 42 dB. To compare the field sweep data with CHORUS spectra, the field sweep data was transformed to the frequency domain, normalised, and its baseline adjusted by putting its first point to zero.

General data processing was performed in Xepr and Python 3.

2 Phase optimisation

2.1 Coarse setup

- One parameter: signal phase
- Experiment: Hahn echo (π-pulse twice as long as $\frac{\pi}{2}$-pulse)
- Cost function: $-\int \Re(\text{echo})$

Table S2: Results of signal phase optimisation. f_{best} represents the value of the cost function at its minimum, and n_{fev} represents the total number of function evaluations (i.e. measurements).. The phase values were implemented with a tolerance of 25 (a.u.).

Algorithm	f_{best}	Optimal parameter	n_{fev}	Time (s)
MDS	-1.68 ± 0.01	1843 ± 22	18 ± 1	68 ± 3
Nelder–Mead	-1.68 ± 0.01	1900 ± 47	20 ± 1	75 ± 4
BOBYQA	-1.67 ± 0.01	1866 ± 25	11 ± 1	42 ± 5

2.2 Fine adjustment

- One parameter: signal phase, using the optimum from the coarse setup as the initial point
- Experiment: Hahn echo (π-pulse twice as long as $\frac{\pi}{2}$-pulse)
- Cost function: $\int \Im(\text{echo})$

Table S3: Results of signal phase optimisation. f_{best} represents the value of the cost function at its minimum, and n_{fev} represents the total number of function evaluations (i.e. measurements). The phase values were implemented with a tolerance of 5 (a.u.).

Algorithm	f_{best}	Optimal parameter	n_{fev}	Time (s)
MDS	3.61 ± 3.32	1893 ± 35	15 ± 1	55 ± 4
Nelder–Mead	3.55 ± 1.99	1880 ± 37	20 ± 1	76 ± 4
BOBYQA	3.76 ± 1.87	1900 ± 14	8 ± 1	31 ± 4

The final cost function values in this instance were found to be quite variable, as forcing the imaginary part to zero means that the *relative* noise level increases as the algorithm converges to an optimum.

3 Flip angle optimisation

- Two parameters:: pulse length, attenuation
- Experiment: Hahn echo (π-pulse twice as long as $\frac{\pi}{2}$-pulse)
- Cost function: $-\int |\text{echo}|$

Table S4: Results of flip angle optimisation. f_{best} represents the value of the cost function at its minimum, and n_{fev} represents the total number of function evaluations (i.e. measurements). The pulse length values were implemented with a tolerance of 2 ns and the attenuation values with a tolerance of 1 dB.

		Optimal parameters			
Algorithm	f_{best}	Length (ns)	Attenuation (ns)	n_{fev}	Time (s)
MDS	-3.14	24	4	11	82 ± 1
Nelder–Mead	-3.16 ± 0.05	24 ± 1	3.8 ± 0.4	16 ± 2	120 ± 20
BOBYQA	-3.80 ± 0.01	6	2	12	90 ± 1
Brute force	-3.97	6	1	198	1459

4 DEER pump pulse optimisation

4.1 Pulse shapes and simulation

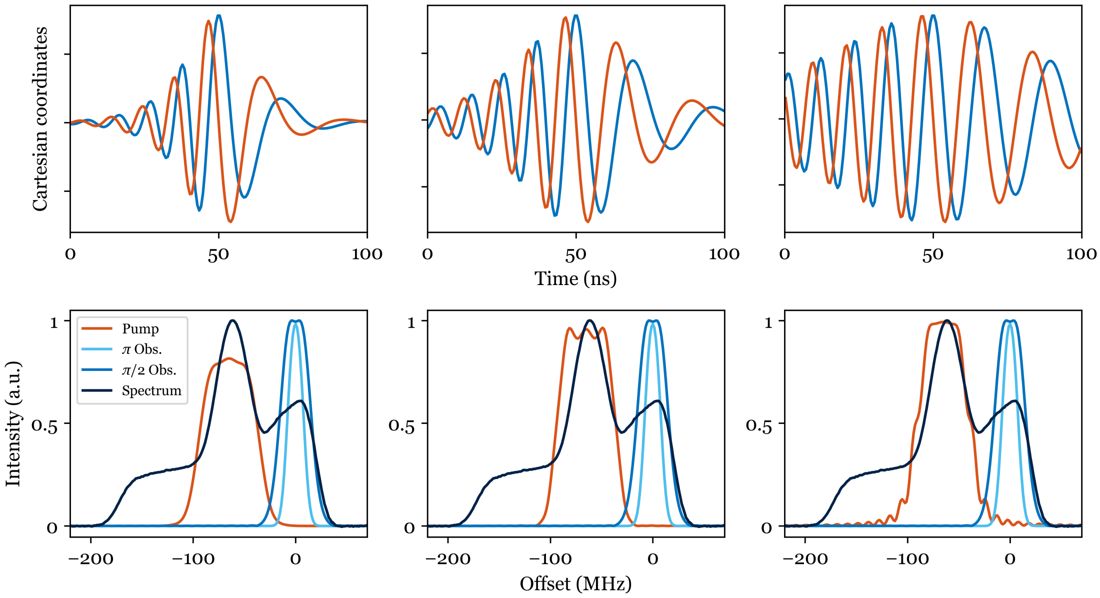

Figure S1: Top: Cartesian representations of pump pulse for $\beta = 10$, 6, and 3 (left to right). Bottom: simulation of observer and pump pulses profiles superposed with the field sweep measurement. These simulations do not offer the most suitable value. Additionally, they rely on assumptions on actual B_1 values and approximations such as disregarding hardware response despite their well-known impact on frequency-swept pulses.[9–11]

The amplitude $\omega_1(t)$ of an HS pulse can be defined over $t \in [\delta_\mathrm{t} - \tau_\mathrm{p}/2, \delta_\mathrm{t} + \tau_\mathrm{p}/2])$ as

$$\omega_1(t) = \omega_{1,\mathrm{max}} \operatorname{sech}\left(\frac{\beta(t-\delta_\mathrm{t})}{\tau_p}\right) \tag{S1}$$

where $\omega_{1,\mathrm{max}}$ represents the peak amplitude of the pulse, τ_p the pulse duration, and δ_t the time corresponding to the centre of the pulse. The selectivity factor β can be defined as c/τ_p, where c is some constant.[12] Here, we define β as a constant so that it can be directly optimised. The pulse phase $\phi(t)$ can be expressed as

$$\phi(t) = \phi_0 + \frac{\pi(\Delta F)\tau_\mathrm{p}}{\beta} \ln\left[\cosh\left(\frac{\beta(t-\delta_\mathrm{t})}{\tau_\mathrm{p}}\right)\right] + 2\pi\delta_\mathrm{f}(t-\delta_\mathrm{t}) \tag{S2}$$

where ΔF is the bandwidth, ϕ_0 is the overall phase offset (typically used for phase cycling), and δ_f the frequency offset of the pulse (which can, for example, be used to control the position of the pump pulse in a DEER experiment).

4.2 Optimisation

- One parameter: HS pump pulse selectivity parameter β
- Experiment: 4-pulse DEER
- Cost function: $-\eta_{2p}$

Table S5: Results of DEER pump pulse selectivity optimisation. f_{best} represents the value of the cost function at its minimum, and n_{fev} represents the total number of function evaluations (i.e. measurements). The β values were implemented with a tolerance of 1.

Algorithm	f_{best}	Optimal parameter	n_{fev}	Time (s)
MDS	-1.49 ± 0.02	5.4 ± 0.5	7	159
Nelder–Mead	-1.50 ± 0.04	5.6 ± 0.5	7	168 ± 11
BOBYQA	-1.51 ± 0.04	6.0 ± 0.6	7	150 ± 13

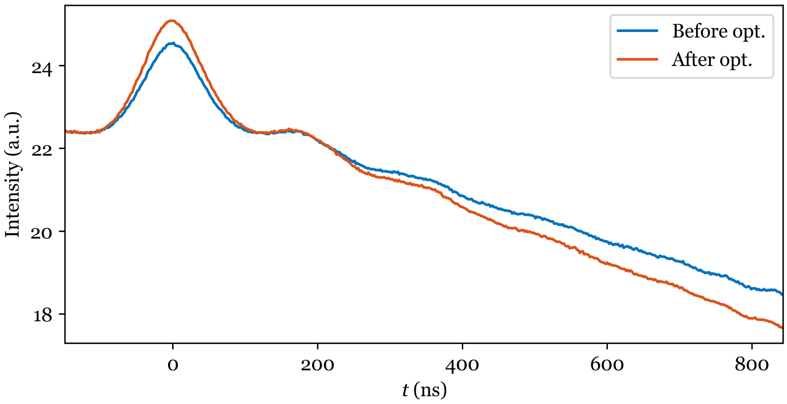

Figure S2: Raw DEER traces obtained before and after optimisation.

5 CHORUS amplitudes optimisation

- Three parameters: pulse amplitudes
- Experiment: CHORUS
- Cost function: $-\int |\text{spectrum}|$

Table S6: Results of CHORUS amplitudes optimisation, including a comparison against our previous approach which uses the intensity of the echo to adjust the two refocusing pulses. f_{best} represents the value of the cost function at its minimum, and n_{fev} represents the total number of function evaluations (i.e. measurements). The pulse amplitudes values were implemented with a tolerance of 1%.

Algorithm	f_{best}	Optimal parameters	n_{fev}	Time (s)
MDS	-5.00	29.2 ± 0.4, 85.6 ± 0.5, 96.2 ± 0.4	42 ± 5	588 ± 68
Nelder–Mead	-5.01 ± 0.01	29.4 ± 0.5, 81.4 ± 0.5, 90.6 ± 0.8	38 ± 4	523 ± 59
BOBYQA	-4.99	29.6 ± 0.5, 87.2 ± 0.7, 99.0 ± 0.9	20 ± 1	285 ± 16
Brute force [8]	-4.98	29, 67, 73	2560	4100

6 CHORUS resonator compensation

6.1 Theory

Resonator compensation relies on the use of a transfer function $h(t)$, describing the effect of the hardware on a pulse, can create a compensated pulse $f(t)$ that, when applied, has the effect of the intended pulse $g(t)$,

$$g(t) = f(t) * h(t), \tag{S3}$$

where $*$ indicates convolution. Using the convolution theorem, Equation (S3) can be rewritten in the frequency domain

$$\begin{aligned} \mathcal{F}\{g(t)\} &= \mathcal{F}\{f(t) * h(t)\}, \\ \implies G(s) &= F(s) \cdot H(s), \end{aligned} \tag{S4}$$

where $\cdot$ indicates point-wise product. Therefore, by knowing $h(t)$, and hence $H(s)$, one can calculate the compensated pulse shape

$$f(t) = \mathcal{F}^{-1}\left\{\frac{\mathcal{F}\{g(t)\}}{H(s)}\right\}. \tag{S5}$$

We considered the general form of the transfer function $H(s)$ as

$$H(s) = \frac{1 + bs}{1 + as}, \qquad a, b \in \mathbb{C}. \tag{S6}$$

6.2 Simulation

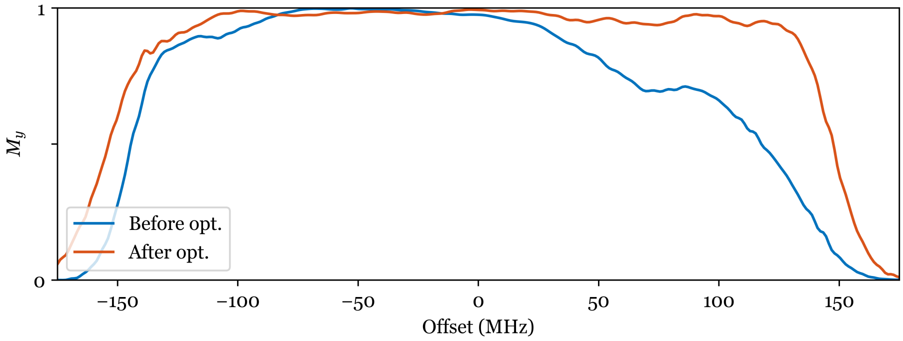

Figure S3: CHORUS compensation simulation results over 300 spin-$\frac{1}{2}$s (resonator profile used for $H(s)$: x-LiPc at X-band [8], centred on resonator maximum. simulation of the transfer function effect on the pulses [9, 13]: EasySpin 5.2.28 [14]). Powell's Derivative-Free Optimization solvers (PDFO)[15] was used to run optimisations with BOBYQA. The optimiser was forced to stop after 50 function evaluations to emulate experimental implementations.

6.3 Optimisation

- Five parameters: amplitude of the first pulse, two complex-valued coefficients of the transfer function (two real parameters each)
- Experiment: CHORUS
- Cost function: ||CHORUS spectrum − field-sweep spectrum||$_2$

Table S7: Results of CHORUS resonator compensation optimisation. The cost function value differs slightly from the main text as the data is noisy and the measurement for the main text figure was done after the optimisation. The second line represents the experiment conducted at −100 MHz. f_{best} represents the value of the cost function at its minimum, and n_{fev} represents the total number of function evaluations (i.e. measurements).

Algorithm	f_{best}	Optimal parameters	n_{fev}	Time (s)
BOBYQA	1.00	38.5, −0.185, 0.410, −0.530, 0.430	113	1357
BOBYQA	1.85	35.0, 0.090, −0.875, −0.170, 0.825	125	1501

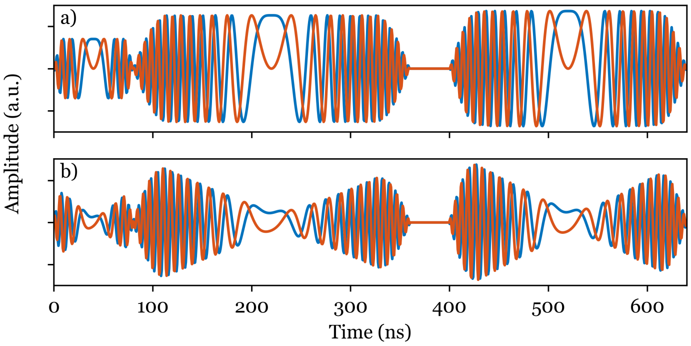

Figure S4: Cartesian representation of the CHORUS sequences: (a) before, (b) and after ESR-POISE compensation.

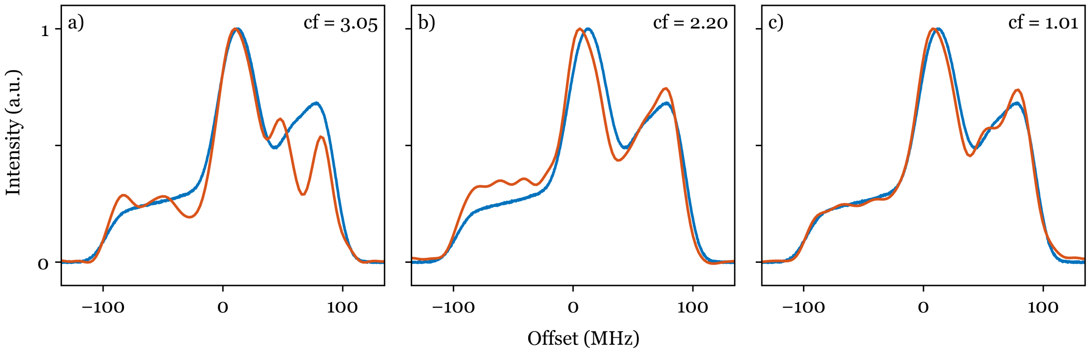

Figure S5: Normalised spectra obtained with CHORUS sequences (red) and the field-sweep (blue): (a) without optimisation, (b) after amplitude optimisation, (d) after amplitude optimisation and with ESR-POISE compensations. The value of the cost function (cf) is shown for all cases. The carrier frequency was set to 9.46 GHz.

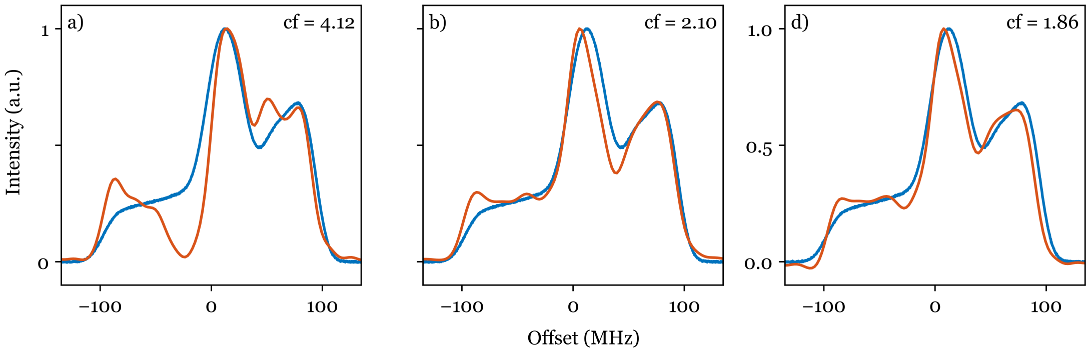

Figure S6: The same as Figure S5, but with a 100 MHz higher carrier frequency.

7 Additional example 1: Field tweak

- One parameter: magnetic field
- Experiment: Hahn echo (π-pulse twice as long as $\frac{\pi}{2}$-pulse)
- Cost function: $-\int |\text{echo}|$

Table S8: Results of field optimisation. f_{best} represents the value of the cost function at its minimum, and n_{fev} represents the total number of function evaluations (i.e. measurements). In this case, f_{best} did not vary significantly between different optimisation runs. However, the brute-force value of f_{best} slightly differs, probably due to instrumental drift during a long experiment. The field values were implemented with a tolerance of 0.25 G.

Algorithm	f_{best}	Optimal parameter	n_{fev}	Time (s)
MDS	-1.321 ± 0.002	3407.55 ± 0.29	17 ± 1	138 ± 20
Nelder–Mead	-1.321 ± 0.002	3407.40 ± 0.12	16 ± 1	141 ± 9
BOBYQA	-1.321 ± 0.002	3407.35 ± 0.12	9 ± 1	82 ± 7
Brute force	-1.314	3407.35	141	1259

8 Additional example 2: MPFU channels phase

MPFU (Microwave Pulse Forming Unit) channels are used when long delays in the sequence do not allow to fully load the sequence into the AWG (typically for RIDME [16, 17] experiment). Imbalance in their phases causes quadrature error in the phase cycling which leads to signal loss. The user has to adjust each channel phase manually but with ESR-POISE, one can set them up one after the other in four consecutive optimisations.

8.1 Basic case

The goal is to adjust the phase to get the real part of the echo.

- One parameter: channel phase
- Experiment: Hahn-echo
- Cost function: $-\int \Re(\text{echo})$

Table S9: Results of channel phase optimisation. f_{best} represents the value of the cost function at its minimum, and n_{fev} represents the total number of function evaluations (i.e. measurements). The phase was implemented with a tolerance of 1%.

Algorithm	f_{best}	Optimal parameter	n_{fev}	Time (s)
MDS	-8.47 ± 0.02	54.6 ± 1.5	12	36
Nelder–Mead	-8.45 ± 0.02	55.2 ± 0.4	13 ± 1	40 ± 3
BOBYQA	-8.43 ± 0.02	55.2 ± 1.0	7	20 ± 1
Brute force	-8.49	55	100	306

8.2 Consecutive optimisations

- One parameter: channel phase

  To optimise all four phases, each phase was optimised separately one after the other (as opposed to all four being optimised simultaneously).
- Experiment: Hahn echo with a phase cycle which requires an additional channel for each optimisation
- Cost function: $-\int \Re(\text{echo})$

Table S10: Results of four consecutive channel phase optimisations. f_{best} represents the value of the cost function at its minimum after the last optimisation, and n_{fev} represents the total number of function evaluations (i.e. measurements) for 4 consecutive optimisations. The phase was implemented with a tolerance of 1%. The optimal parameters are not shown here as the optimisation criterion enforces the relative channel phases with respect to one another, while the actual values may change.

Algorithm	f_{best}	n_{fev}	Time (s)
MDS	-3.69 ± 0.01	49 ± 1	166 ± 1
Nelder–Mead	-3.70	52 ± 1	175 ± 5
BOBYQA	-3.70 ± 0.01	31 ± 1	104 ± 3
Brute force	3.67	400	1372

References

[1] MATLAB, version 9.9.0.1467703 (R2020b), The MathWorks Inc., Natick, Massachusetts, 2020.

[2] J. J. E. Dennis, V. Torczon, Direct search methods on parallel machines, SIAM Journal on Optimization 1 (4) (1991) 448–474.

[3] J. A. Nelder, R. Mead, A simplex method for function minimization, The Computer Journal 7 (4) (1965) 308–313.

[4] M. J. D. Powell, The BOBYQA algorithm for bound constrained optimization without derivatives, University of Cambridge DAMTP 2009/NA06 (2009).

[5] C. Cartis, J. Fiala, B. Marteau, L. Roberts, Improving the flexibility and robustness of model-based derivative-free optimization solvers, ACM Transactions on Mathematical Software 45 (3) (2019) Article 32.

[6] M. Teucher, E. Bordignon, Improved signal fidelity in 4-pulse DEER with Gaussian pulses, Journal of Magnetic Resonance 296 (2018) 103–111.

[7] L. Fábregas Ibáñez, G. Jeschke, S. Stoll, DeerLab: a comprehensive software package for analyzing dipolar electron paramagnetic resonance spectroscopy data, Magnetic Resonance 1 (2) (2020) 209–224.

[8] J.-B. Verstraete, W. K. Myers, M. Foroozandeh, Chirped ordered pulses for ultra-broadband ESR spectroscopy, The Journal of Chemical Physics 154 (9) (2021) 094201.

[9] A. Doll, S. Pribitzer, R. Tschaggelar, G. Jeschke, Adiabatic and fast passage ultra-wideband inversion in pulsed EPR, Journal of Magnetic Resonance 230 (2013) 27–39.

[10] A. Doll, G. Jeschke, Fourier-transform electron spin resonance with bandwidth-compensated chirp pulses, Journal of Magnetic Resonance 246 (2014) 18–26.

[11] P. E. Spindler, P. Schoeps, A. Bowen, B. Endeward, T. F. Prisner, Shaped pulses in EPR, eMagRes 5 (4) (2016) 1477–1492.

[12] K. E. Cano, M. A. Smith, A. J. Shaka, Adjustable, broadband, selective excitation with uniform phase, Journal of Magnetic Resonance 155 (1) (2002) 131–9.

[13] T. Kaufmann, T. J. Keller, J. M. Franck, R. P. Barnes, S. J. Glaser, J. M. Martinis, S. Han, Dac-board based x-band epr spectrometer with arbitrary waveform control, Journal of Magnetic Resonance 235 (2013) 95–108.

[14] S. Stoll, A. Schweiger, Easyspin, a comprehensive software package for spectral simulation and analysis in epr, Journal of Magnetic Resonance 178 (1) (2006) 42–55.

[15] T. M. Ragonneau, Z. Zhang, PDFO: Cross-platform interfaces for Powell's derivative-free optimization solvers (version 1.1), https://www.pdfo.net (2021).

[16] L. V. Kulik, S. A. Dzuba, I. A. Grigoryev, Y. D. Tsvetkov, Electron dipole–dipole interaction in ESEEM of nitroxide biradicals, Chemical Physics Letters 343 (3) (2001) 315–324.

[17] S. Milikisyants, F. Scarpelli, M. G. Finiguerra, M. Ubbink, M. Huber, A pulsed EPR method to determine distances between paramagnetic centers with strong spectral anisotropy and radicals: The dead-time free RIDME sequence, Journal of Magnetic Resonance 201 (1) (2009) 48–56.