A Simple Guide for Simulating Bruker Pulse Sequences in MARSS

Kelley M. Swanberg, Ph.D.
Lunds universitet
2025.12.10

amplitudeUnit: Gauss, Hz, or radians. Convert among the three using gamma and the fact that a full cycle is 2pi radians.

autoMean: Unless you are troubleshooting pulse profiles and the like, this value should be '1'.

B0: acqp file $SF0ppm / 42.58 = 9.3837 for the LBIC 9,4 T

bandwidth: method file `$PVM_SpecSWH`

coherencePathway: [-1, 1, -1, 1, -1] for sLASER

delays: Note that these are the intervals between the end of one pulse and the beginning of the next and NOT the middle of one pulse to the middle of the next. Calculating these can be very different depending on the ParaVision version and pulse sequence employed, but for the most part all the information needed is provided in **backbone.c** (requires access to the uncompiled sequence source code) and the ppg file (often exported with a dataset) for the sequence. For sLASER in ParaVision 360 v. 3.6, the variables behind the timings are nicely defined in a few lines within **backbone.c**, and the visible parameters actually comprising these variables are fairly straightforward to discern elsewhere in this file and the ppg document.

It is good to first define the timings as the typical τ values used to define a pulse sequence (including pulse durations and accounting for the point in the pulse, usually but not always halfway through, at which echo time countdown actually starts) to make sure that they correspond with each other in expected ways (e.g., τ 1 + τ3 + τ5 = τ2 + τ4 for sLASER).

For vanilla sLASER in PV360, ultimately each τ boils down to a combination of the following variables, all accessible in the **method** file exported with a raw dataset:

```
VoxPul1.Length
VoxPul1.RpFac
VoxPul2.Length
VoxPul3.Length
GradStabDelay
PVM_RiseTime
SpoilerDuration[0]
SpoilerDuration[1]
SpoilerDuration[2]
SpoilerDuration[3]
SpoilerDuration[4]
```

And the τ themselves can be built from these variables as follows:

τ1: GradStabDelay + PVM_RiseTime + SpoilerDuration[0] + PVM_RiseTime + GradStabDelay
(+ VoxPul1.Length * RPfac/100 + VoxPul2.Length/2)
τ2: GradStabDelay + PVM_RiseTime + SpoilerDuration[1] + PVM_RiseTime + GradStabDelay
(+ VoxPul2.Length/2 + VoxPul2.Length/2)
τ3: GradStabDelay + PVM_RiseTime + SpoilerDuration[2] + PVM_RiseTime + GradStabDelay
(+ VoxPul2.Length/2 + VoxPul3.Length/2)
τ4: GradStabDelay + PVM_RiseTime + SpoilerDuration[3] + PVM_RiseTime + GradStabDelay
(+ VoxPul3.Length/2 + VoxPul3.Length/2)
τ5: GradStabDelay + PVM_RiseTime + SpoilerDuration[4] + PVM_RiseTime + GradStabDelay
(+ VoxPul3.Length/2)

To convert these proper τ values to the delays to be input in MARSS, simply remove the values corresponding to RF pulse durations (in parentheses above) and convert from milliseconds to seconds.

durations: RF pulse durations are the first values in $VoxPul1, $VoxPul2, … $VoxPuln arrays inside the method file. Note that they need to be converted from milliseconds to seconds for MARSS.

G: gradient strengths in mT/m. Calculate from radiofrequency pulse bandwidths (second value, in Hz, in $VoxPuln arrays from method file), gamma-bar and length of the appropriate simulated voxel dimension according to Gn = BW / (gamma-bar * voxel dimension). Note that if you deal in anisotropic voxels and use different pulses for any of your dimensions (as in PRESS or sLASER) you should start thinking about the axes along which your slice-selective pulses actually lie. This information is now provided (and adjustable) in ParaVision 360.

metabolites: List of spin systems to be simulated; no Bruker-specific considerations.

Npoints: Number of complex free induction decay (FID) points to be density-matrix simulated

phaseShifts: If phase-cycling is not represented, then only the receive phase may need adjustment

referencePeak: Location of water peak; we use 4.7 ppm

rephaseAreas: Calculated from area (mT/m strength x s duration) of dephasing localization gradient during radiofrequency pulse

rfOffsets: Reflective of where RF pulses are centered; note that these are Hz conversions of ppm values from TMS (0 ppm), and not water (wherever the referencePeak was defined), unlike the offsets as we define them in ParaVision

rfPulses: RF pulse shapes. Magnitude (in percent of maximum)-phase (in degrees) vectors describing shaped pulses are found in the exp/stan/nmr/lists/wave folder or similar on the scanner.

Calculated pulse shapes can be approximated in and exported from TopSpin according to their type (discerned by the sharpness, e.g. 3 for Hermite pulses) duration (can be seen directly in the pulse menu on the Sequence card in ParaVision, but also usually available in the method file, as, e.g. `$VoxPul<n>.Length`), flip angle, and number of points. They can also be exported with acquisitions using the sequence in question by adding the high-level variables associated with them (usually but not always `VoxPul1Shape`, `VoxPul2Shape`, `VoxPul3Shape`, etc.; the exact variable names can be discerned from the RF pulse menu in the Sequence card in PV, or the source code itself) within a pargroup in the **parsLayout.h** file of the sequence source, recompiling the sequence, and using it to acquire data.

(Max B1)*gamma-bar (Hz) for a given flip angle can be calculated by 'Analyze Waveform → Calculate gammaB1max' in TopSpin from pulse shapes either simulated or exported as described above. These can be converted to Gauss by dividing out gamma-bar and multiplying by 10,000 Gauss/T. Multiply the percent amplitude value given at each point of the Bruker pulse shape by the maximum such that 100% obviously corresponds with this maximum B1 value. Convert the phase value given at each point of the Bruker pulse shape to radians.