Vespa – RFPulse
User Manual and Reference

Version 0.8.6

Release date: February 21st, 2016

Developed by:

Brian J. Soher, Ph.D.
Philip Semanchuk

Duke University Medical Center,
Department of Radiology, Durham, NC

Karl Young, Ph.D.
David Todd, Ph.D.
Jerry Matson, Ph.D.

University of California, San Francisco
Department of Radiology, San Francisco, CA

Developed with support from NIH, grant # EB008387-01A1

Table of Contents

Overview of the Vespa Package

The Vespa package enhances and extends three previously developed magnetic resonance spectroscopy (MRS) software tools by migrating them into an integrated, open source, open development platform. Vespa stands for Versatile Simulation, Pulses and Analysis. The original tools that have been migrated into this package include:

- GAVA/Gamma - software for spectral simulation
- MatPulse – software for RF pulse design
- IDL_Vespa – a package for spectral data processing and analysis

The new Vespa project addresses current software limitations, including: non-standard data access, closed source multiple language software that complicates algorithm extension and comparison, lack of integration between programs for sharing prior information, and incomplete or missing documentation and educational content.

Introduction to Vespa-RFPulse

Vespa-RFPulse is a graphical control and visualization program written in the Python programming language that provides a user friendly front end for calculation of Shinnar-LeRoux (SLR) frequency selective RF pulses, and many other calculations and manipulations. Although RFPulse is meant to be intuitive and is entirely menu-driven, a cursory reading of the information below is recommended. A description of the original MatPulse program (beta version 1.0) is provided in reference 1 (below). The RF pulses and gradient waveforms created by RFPulse follow the general procedures and nomenclature provided in references 2 and 3. Along with SLR based pulses, analytical pulse shapes, such as hyperbolic secant, are also available, as are additional transformation steps such as pulse interpolation and re-scaling.

1. Matson, G.B. An integrated program for amplitude-modulated RF pulse generation and re-mapping with shaped gradients. Magn. Reson. Imaging 12, 1205-1225, 1994.

2. J. Pauly, P. Le Roux, D. Nishimura, and A. Macovski, "Parameter Relations for the Shinnar-Le Roux Selective Excitation Pulse Design Algorithm", IEEE Trans Med Imaging 10: 53 - 65 (1991).

3. S. Conolly, D. Nishimura, A. Macovski, and G. Glover, "Variable-Rate Selective Excitation", J Magn Reson 78: 440 - 458 (1988).

What can RFPulse do?

1) Create new RF pulse projects from a list of design algorithms
2) Store pulse projects and their design parameters into a database
3) Re-load previous pulse projects
4) Display pulse results for each step of the design process in a flexible plotting tool
5) Compare side-by-side results from one or more pulse projects
6) Output results in text or graphical format, including MR manufacturer platform formats
7) Export/Import Vespa pulse projects to/from other users
8) Be an open source test bed for your own algorithms and transformations.

What is an RF pulse project? A ‘pulse project’ consists of one or more design steps. Each pulse project has a single creation step. You can further refine the pulse by adding additional general transformation steps. Each step of the design process (creation plus other transformations) contains time and frequency domain results for the RF pulse up to that point. These results can be visualized in plots to the GUI at any time. Changes to any given step in the design process trigger a “downstream” calculation of all subsequent transformations.

You can open multiple pulse projects and compare them side-by-side. Any pulse project tab can be copied into a new pulse project tab. This allows you to make minor changes to a copy of an RF pulse in order to see the effects on the final results.

RFPulse can create (and expects to work with) four separate 'vectors' which you will see mentioned and explained in more detail in later sections:

The first, **B1**, represents an amplitude modulated RF pulse.
The second, **B2**, represents a re-mapped pulse (reshaped for use with shaped gradients).

The third, **G2**, represents the shaped gradient waveform for use with **B2**.
The fourth, **F2**, represents the frequency offsets for an offset slice (used with **B2** and **G2**).
In addition, the constant gradient for a conventional selective pulse is labeled **G1**.

The following chapters run through the operation of the Vespa-RFPulse program both in general and screen by screen.

In this manual, command line instructions will appear in a fixed-width font on individual lines, for example:

```
~/Vespa-RFPulse/ % ls
```

Specific file and directory names will appear in a fixed-width font within the main text.

Online Resources:

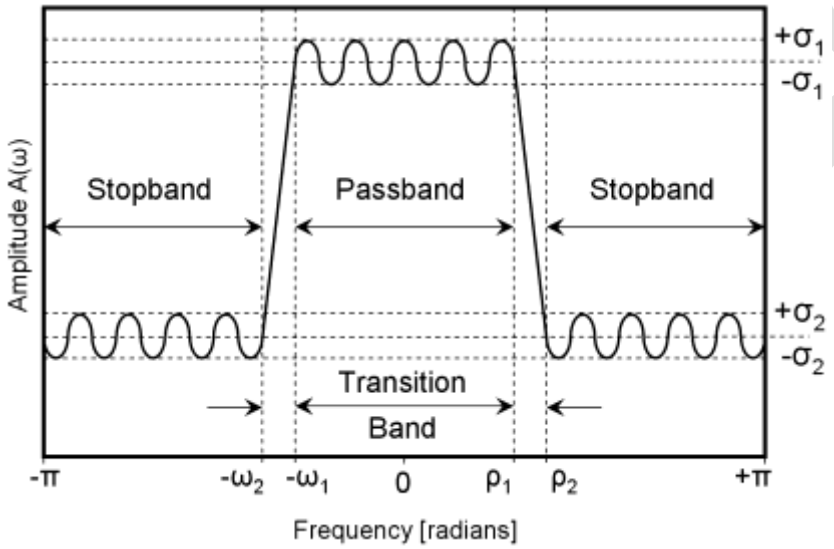
The Vespa project and each of its applications have wikis with extensive information about how to use, and develop new functionality for, each application. These can be accessed through the main portal at

http://scion.duhs.duke.edu/vespa/

Case Studies in RF Pulse Design

Note. The following case studies discussion was generously contributed by Dr. Jerry Matson based on the functionality of his MatPulse program. Not all features discussed below are implemented in the Vespa-RFPulse application.

Although your MRI instrument has a slew of RF pulses installed and available for use in new experiments, there are many reasons why you may want to consider designing new RF pulses. The figure below shows a generic pulse profile to define terms used to describe a pulse profile.



The following sections list some of the reasons why you may want to consider creating your own RF pulse designs.

Case 1 – Improved Selectivity

Many MRI pulses are designed with short TE sequences in mind, and the pulses are designed with a reduced length (at the expense of increased transition bands which creates some loss in selectivity and sensitivity) to accommodate short TEs. Somewhat longer pulses that provide for shorter transition bands may provide improved S/N for signals with longer T2s. In addition, pulse profiles with minimal transition bands are helpful for slice selection without slice gaps. RFPulse provides menus for designing longer RF pulses to reduce the transition bands.

Case 2 – Reduced Contamination

Especially for spectroscopy, contamination from signal from outside the excitation bandwidth can be a problem (e.g., leading to lipid or water signals from outside the voxel interfering with spectroscopic signals from within the voxel. Selective pulses designed for improved out-of-band (stopband) suppression can reduce this problem. The SLR pulse design tab in RFPulse allows users to select the amount of stopband (reject band) suppression, and facilitates exploring the tradeoffs involved between increased stopband suppression, increased transition band, and increased pulse length.

Case 3 – Observation of Spectroscopic Signals Close to Water

Most MRI instruments use Gaussian pulses for water suppression. However, Gaussian-shaped pulses generate large transition zones, leading to suppression of other spectroscopy signals close to water. The use of pulses with improved selectivity (e.g., SLR pulses as designed with RFPulse) can reduce this problem.

Case 4 – Lowered Peak Voltage Pulses

The peak voltage (or power) that can be applied by the MRI instrument to the coil is limited. This can prevent a desired RF pulse from being implemented on the instrument. For example, spin echo pulses require 3 to 4 times the peak voltage required by a 90 degree pulse producing the same bandwidth. In addition, hyperbolic secant inversion pulses require high peak voltage. There are several approaches to lowering the peak voltage of these pulses.

Spin echo pulses: An RF pulse can be expressed in terms of a pair of polynomials, designated A and B (see e.g. the paper by Pauli et al. cited above). For an SLR pulse, the B polynomial represents the magnitude of the selection profile that will result from the pulse, while the A polynomial contains the phase information for the pulse. For a so-called minimum phase pulse, the roots of the A polynomial lie within the unit circle in the complex plane.

Reflection of one of the roots of the A polynomial across the unit circle - altering the value of the root from z to 1/(z*) – doesn't alter the profile produced by the pulse, but may lower the peak voltage required. Typically, the peak voltage can be reduced to around 0.6 of the original voltage required. While the peak voltage is decreased, the price to be paid is that the SAR for the pulse is increased.

A second approach to lowering the peak voltage of a spin echo pulse is to use the VERSE technique (see, for example, the paper by Matson cited above), which can be thought of as lengthening the center portion of the pulse to enable the voltage of this section of the pulse to be reduced. The FOCI technique is similar, and can be considered as a particular implementation of the VERSE method. A flexible implementation of the VERSE method is known as "remapping". This method enables the RF pulse and associated gradient to be altered to lower the peak voltage of the pulse. While this method does not necessarily increase the overall length of the pulse, depending on the parameters used, it may increase the SAR.

Hyperbolic secant inversion pulses: The hyperbolic secant pulse represents an adiabatic pulse, where the performance of the pulse does not change once the total power of the pulse is above a certain threshold. The advantage of this type of pulse is that it can perform well even in an inhomogeneous B1 field.

The voltage shape of the hyperbolic secant pulse can be expressed as $\mathrm{sech}(\beta\tau)$, where τ is the normalized time extending from -1 to 1. This shape produces a high voltage at the center of the pulse. Garwood et al. have shown that, by using a shape function of the form $\mathrm{sech}(\beta\tau^n)$, the peak voltage of the pulse can be reduced, while the pulse retains its adiabatic character.

The original hyperbolic secant pulse is thus designated HS1, while the reduced voltage pulses are designated HSn, where n indicates the power to which τ is raised. RFPulse

enables HSn pulses to be designed over a range of values of n. Although larger values of n lower the peak voltage, the selectivity of the pulse is degraded at higher values of n.

Case 5 – Reduced SAR

Due to the high peak voltage required, both spin echo and hyperbolic secant pulses generate high SAR, and lower SAR versions of these pulses may be desired. For spin echo pulses, the remapping method (discussed in the previous section) can be employed with parameters chosen to lower the SAR (and lengthen the pulse duration) of a spin echo pulse. For a hyperbolic inversion pulse, the pulse can be implemented as an HSn pulse (discussed in the previous section) to lower the SAR.

Case 6 – More Exotic Pulses

While RFPulse can already be used to create Multiband pulse designs, future additions to RFPulse will include the following:

Multiband pulses: in which two separate selection bands are included within a single pulse design. For example, in a spectroscopy editing experiment, editing may be desired simultaneously in two separate regions, or suppression in two regions (e.g., water and lipid).

Increased bandwidth pulse designs: useful in spectroscopy to suppress the "two compartment artifact', which causes signal loss for J-coupled metabolites unless the experiment is conducted with very short TE.

Special concatenated RF pulses: used in a variety of experiments. For example, concatenated RF pulses are used in arterial spin labeling experiments such as the TILT experiment and in so-called pseudo continuous arterial spin labeling experiments.

B1 insensitive pulse design: particularly at high magnetic field, the B1 field becomes inhomogeneous, and it may be desirable to use RF pulse designs that perform uniform tipping even in the presence of inhomogeneous RF fields. RFPulse provides optimization methods to obtain such designs.

Using RFPulse – A User Manual

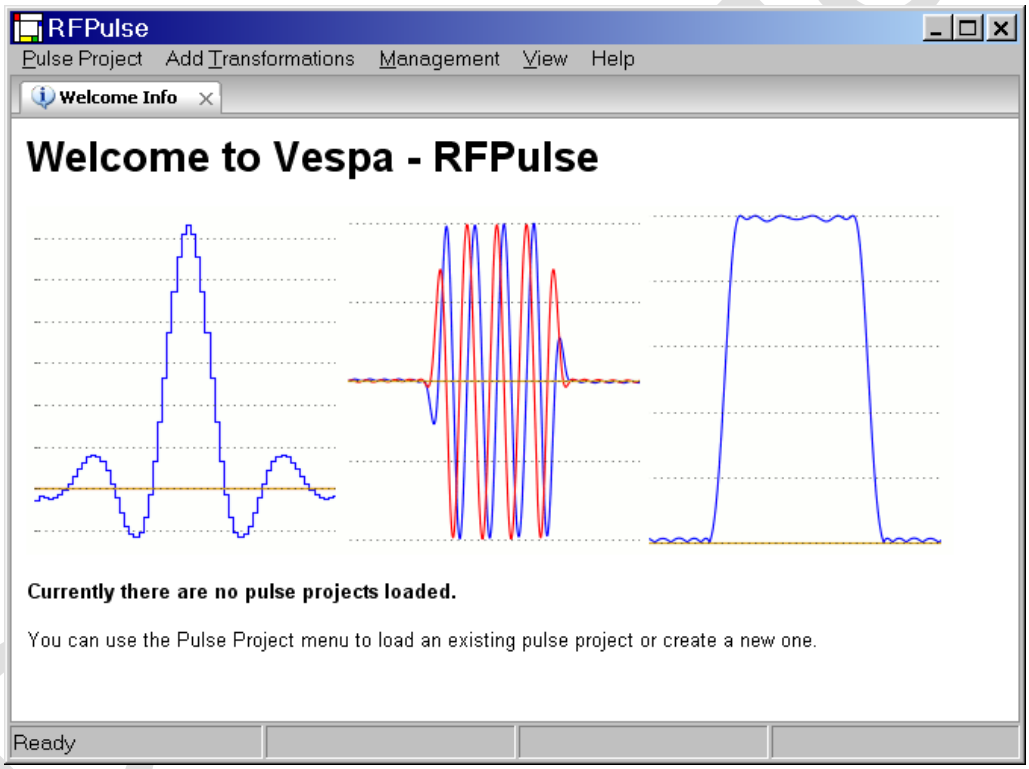
This section assumes Vespa-RFPulse has been downloaded and installed. See the Vespa Installation guide on the Vespa main project wiki for details on how to install the software and package dependencies. http://scion.duhs.duke.edu/vespa.

In the following, screenshots are based on running Vespa-RFPulse on the Windows OS, but aside from starting the program, the basic commands are the same on all platforms.

1. How to launch Vespa-RFPulse

Double click on the Simulation icon that the installer created on your Desktop.

Shown below is the Vespa-RFPulse main window as it appears on first opening. No actual RF pulse project windows are open, only the 'Welcome' banner is displayed.



Use the **RFPulse** menu to open an existing pulse project or to create a new project for designing an RF pulse.

Shown below is a screen shot of a Vespa-RFPulse session with two pulse project tabs opened side by side for comparison. The functionality of all tools, algorithms and transformation will be described further in the following sections.


Pulse Project - Add Transformations Management View Help
Pulse Project 1
Pulse Project 2
Pulse Params
Tip Angle:
Time Steps:
Duration:
Bandwidth:
degrees
increments
milliseconds
kilohertz
Single or Dual Band
Single Band
Dual Band
Non-Coalesced Phase Type
Linear
Filter
Passband Ripple:
Reject Ripple:
Percent
Run
Freq Profile
Abs Freq Profile
Ext Profile
Ext Absolute Profile
Waveform [Volts]
Waveform [Mags]
Waveform [deg]
Usage Type:
Excite
Grad Refoc Profile
Grad Value:
Basic Info
SLR
Frequency Profile
Absolute Frequency Profile
RF Pulse Waveform
Grad Refocused Profile
Magnitude (normal)
Microtesla

2. Quick Guide – The Nuts and Bolts of RFPulse

The creation and modification of RF pulses for use in MR experiments is as much an art as a science. We hope that the modularity and flexibility of design steps as presented in RFPulse will allow and encourage you to play with, and better understand, the effects of changing parameter values in each step of the design. Also, since each step is on a separate tab within the pulse project, you can see the results of each step.

To facilitate your initial usage, we offer here a quick description of how to get started using and learning from RFPulse.

We start with the assumption that you have already installed and launched the RFPulse application. You should see the Welcome screen displayed with no pulse project tabs displayed.

The easiest interaction requires just a few basic steps.

- From the **PulseProject** menu, select New to create a new pulse project.
- In the Basic Info tab, fill in the project name and investigator fields.
- Select **Add Transformations→Create→SLR** and a new "SLR" tab appears.
- Hit the **Run** button and a time domain RF pulse waveform is displayed in the right panel.
- (optional) select other plot control options at the bottom of the tab to see other results plots.
- (optional) vary parameters and hit **Run** again to observe desired pulse performance.
- Select **PulseProject →Save** menu item to store your results to the Vespa database so you can re-open them later.

From here you can change parameters on the SLR creation tab to refine the shape, duration or other features of the pulse. You can also select other transformations from the **Add Transformations** menu. Each transformation appears on a new tab and allows you to further refine your pulse. (The motivations for manipulating RF pulses have been briefly discussed in the Case Studies section).

There are two types of transformations, "create" transformations and "general" transformations. As shown above, pulse projects start with a create transformation. Create transformations are the algorithms by which an initial RF pulse is designed (e.g. Shinnar-LeRoux, hyperbolic secant, etc.). Each pulse project has just one create transformation. The create transformations are selected from the **Add Transformations→Create** menu. General transformations are listed below the **Create** submenu. General transformations operate on the results of the previous transformation (i.e., the transformation in the tab to the left).

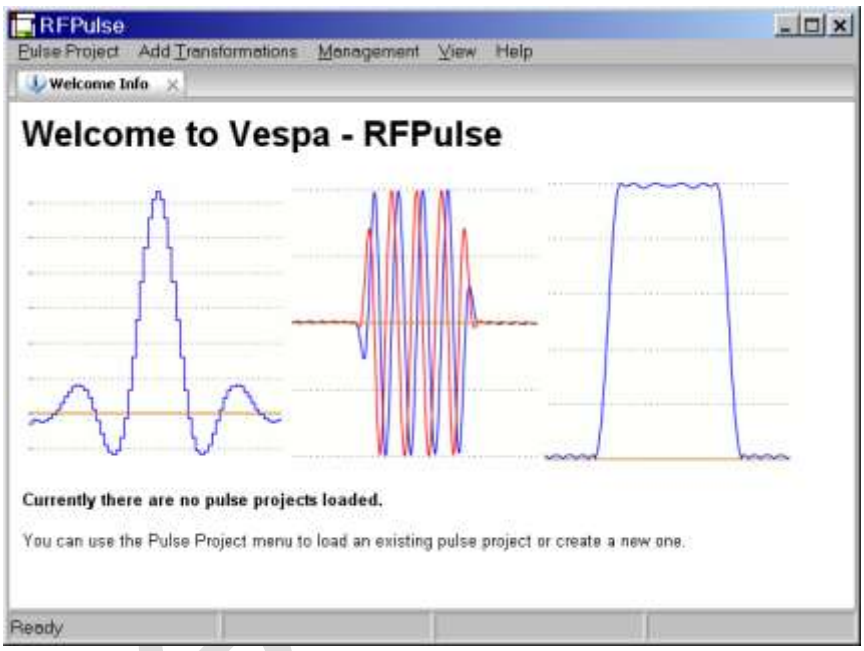
Notes

1. Another typical workflow might be to load a saved pulse project from the database, add, edit or delete general transformations, and then save the changed results back to the database.
2. A third typical workflow might be to load a saved pulse project from the database, and copy that into a new pulse project by using the **PulseProject→Copy Tab to New Tab** menu item. Then modify the RF pulse in the copied tab and compare the results plots to the original pulse project results.
3. Changes can be made to parameter settings in any transformation tab in a pulse project. When you click the Run button, RFPulse calculates the effects of these changes in all

transformation tabs “downstream” from the transformation tab changed. **Note** – as of this writing, there is no “undo” functionality in the transformation tabs in the application. Although, for existing pulse projects that have been loaded into a tab, changes to that project are not saved to the database until you select the **PulseProject→Save** menu manually.

3. The RFPulse Main Window

This is a view of the main Vespa-RFPulse user interface window. It is the first window that appears when you run the program. It contains a tabbed window into which pulse projects can be loaded, a menu bar and status bar. One or more pulse projects can be loaded at a time. Each project is displayed in its own tab located at the top of the window. Each project contains input data and results from one pulse project design. As described above, a pulse project is a group of transformation steps that create and modify an RF pulse. Each pulse project tab contains a group of tabs at the bottom of the page that represent the transformations in the project. Transformation tabs contain the input values and results for that step of the pulse design.



The pulse project window is initially populated with a welcome text window, but no pulse projects are loaded. From the RFPulse menu bar you can 1) load a previously run pulse project from the Vespa database into a tab, or 2) create a new pulse project and set it up and run it. In either case a tab will appear at the top of the window for each pulse project that is loaded or created. The Management menu allows you to access pop-up dialogs to create, edit, view, delete and import/export pulse projects and Machine Settings (which will be explained fully later).

The status bar provides information about where the cursor is located within the various plots and images in the interface throughout the program. It also reports short messages that reflect current processing while events are running.

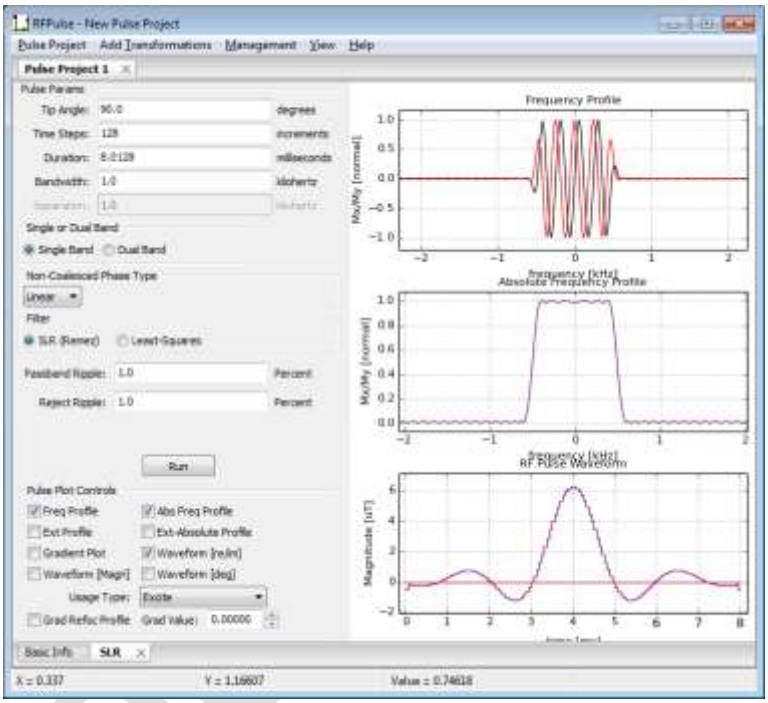
On the Menu Bar

PulseProject→New	Opens a new pulse project tab.
PulseProject →Copy Tab to New	This will open a new pulse project tab and populate it with the same values that are listed in the current pulse project. This is a short cut for varying design parameters to get different results while retaining the ability to compare back to a previous results set without having to save them both to the data base.
PulseProject →Open	Runs the pulse project Browser dialog, from which you can choose a pulse project to open from the database to open.
PulseProject →Run	Runs all steps in the current pulse project tab. Works as if the user had manually hit the Run button in each transformation tab.
PulseProject →Save	Saves the pulse project in the current tab to the database. **Note** - pulse project results are not automatically saved to data base after a Run button is hit.
PulseProject →Save As	Saves the pulse project in the current tab to the database under a new name and unique ID.
PulseProject→Close	Closes the current tab.

PulseProject→Third Party Export	Opens a dialog that will export the final result of the active pulse project into a file suitable for import to various MR manufacturers' platforms.
PulseProject →Exit	Closes current pulse project tab. Will prompt for save if necessary.
Add Transformations →Create →SLR	Appends an SLR (create transformation) tab to current pulse project
Add Transformations →Create →Hyperbolic-Secant	Appends an Hyperbolic-Secant (create transformation) tab to current pulse project
Add Transformations →Create →Gaussian	Appends a Gaussian (create transformation) tab to current pulse project
Add Transformations →Create →Import from File	Appends an Import from File (create transformation) tab to current pulse project
Add Transformations →Interpolate and Rescale	Appends an Interpolate and Rescale (general transformation) tab to current pulse project.
Management →Manage Pulse Projects	Launches the Manage pulse projects dialog. Allows you to view, clone, delete, import and export pulse projects.
Management →Manage Machine Settings Templates	Launches the Manage Machine Settings dialog. Allows you to create, edit, view, and delete templates for machine settings information.
View→<various>	Changes plot options in the selected transformation tab of the active pulse project tab, including: display zero line, turn x-axis on/off or choose units, selecting data type and various output options for all plot windows.
Help→User Manual	Launches the user manual (from vespa/docs) into a PDF file reader.
Help→RFPulse/Vespa Online Help	Online wiki for the RFPulse application and Vespa project
Help→About	Giving credit where credit is due.

4. The Pulse Project Window

The pulse project window offers considerable flexibility. You can open multiple projects simultaneously and they can be moved around, arranged and "docked" as desired by left-click and dragging the desired tab to a new location inside the main window. The tabs can be positioned side-by-side, top-to-bottom or stacked. They can also be arranged in any mixture of these positions.



The pulse project window can be populated with one or more pulse project tabs, each of which contains the results of one pulse project. Pulse project tabs can be closed using the X box on the tab or with a middle-click on the tab itself. When a pulse project Tab is closed, the pulse project is removed from memory, but can be reloaded from the database at a future time - assuming it was previously saved.

5. The Pulse Project Tab

A pulse project tab is one page in the pulse project window. Each tab contains one entire pulse project design. A pulse project tab can be used to run a new pulse project and view the results of that design. It can also be used to load an existing pulse project from the database to view results, or to change parameters or other modifications to the pulse project.

Each pulse project tab will contain sub-tabs at the bottom of the page that describe the transformations used to design the RF pulse. There will typically be two or more of these "transformation tabs". The first tab is standard to all pulse projects and is called "Basic Info". The second tab is always a "create" transformation tab such as "SLR" or "Hyperbolic-Secant" (see Appendix B). Additional tabs, containing "general" transformations such as Interpolate-Rescale (see Appendix B) can be appended after the "create" transformation tab. The pulse project tab can be resized by adjusting the RFPulse application window and each transformation tab has a vertical splitter bar through the middle that can be used to resize the right and left sides. Parameters for each transformation tab are displayed on the left side of the tab, the right side of the tab contains a plotting canvas to visually display the current RF pulse waveform and profile results. Typically, when a transformation tab is added to the pulse project, there are no results to be displayed until the **Run** button has been clicked.

A new pulse project is typically created, set up and run. After this, other changes to parameters can be made and the Project re-run until you are satisfied with the pulse performance. Results from running a pulse project are only saved to the database when specifically requested by you. Transformation tabs are updated to display results after each click on the Run button. If there are transformation tabs "downstream" from the tab where the **Run** button was hit, the program

will automatically propagate changes through those tabs as well (ie. essentially automatically triggering their Run functions). Pulse projects can be modified in any tab multiple times.

All saved pulse projects start life as “private”. That means they're only in your database; no one else has seen/accessed them. Once you export a pulse project to share with another user, the pulse project is designated as “public”. That means the pulse project result has been shared with the world. Public objects are “frozen” and become mostly unchangeable. Once a private pulse project has become public, it can never become private again.

There are two ways that a public pulse project can be re-used. First, cloning a public pulse project (using the Manage Pulse Projects dialog which is described more fully in Section 6) will create a new, private pulse project in the database with exactly the same properties as the original but with a different name. This clone can then be opened into the pulse project window and modified. Second, any pulse project can be loaded into the pulse project window, whether it is public or private. From there, the **Pulse Project→Copy to New Tab** menu item will create a new pulse project tab with a copy of the current tab. This copy will be private and ready to be modified. Note that an important difference between cloning and copying tabs is that the clone is automatically saved into the database on creation, while the copied tab is only created in memory until you specifically save it.

The View menu on the main menu bar can be used to modify the display of the plots in the active pulse project tab. The resulting modifications only affect the settings in the active pulse project tab. More specifically, only the plot options in the selected transformation tab in the active pulse project tab are affected. These options are preserved as you switch from transformation tab to transformation tab. The following functions are on the View menu item:

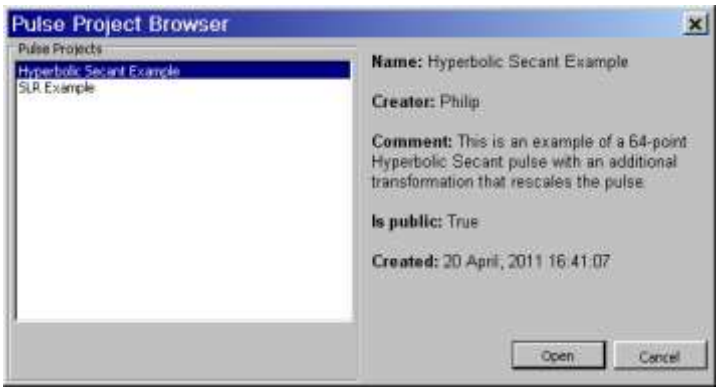
On the Menu Bar

View (this menu affects the plots in the currently active PulseProject/Transformation tab)

→Show ZeroLine	toggle zero line off/on in waveform and profile plots
→Xaxis →Show	turns off/on vertical grid lines and values along the x-axis
→Data Type	select **Real** or **Real+Imaginary** in waveform and frequency profile plots. Typically the real part is plotted in blue, imaginary in red. Magnitude plots are plotted in magenta. However, these colors can be set by the user in the “ini” file for RFPulse to display in any color matplotlib will support.
→Output→<format>	writes the plot panel to file as either PNG, SVG, EPS or PDF format

5.1 Loading an existing pulse project

The pulse project Browser dialog is launched from **Pulse Project→Open** menu which is shown below. A list of pulse project names is shown on the left. When a pulse project listed in the browser is clicked on once, its comment and other information are displayed on the right.



When the Open button is clicked (or a pulse project’s name is double-clicked on), the program loads the information for that pulse project from the database into a pulse project object in memory. This object then creates a pulse project tab and sets up the transformation tabs that describes its creation. You might

notice a delay when opening larger pulse projects.

5.2 Running a new pulse project

When you select the **Pulse Project→New** menu option, a new pulse project tab is created in the pulse project window and the Basic Info transformation tab loaded. This tab requires you to type in a project name before the project can be saved. A project investigator and/or a comment can optionally be added.

A list of available transformations is on the **Add Transformations** menu. Create transformations are found under the **Add Transformations →Create** sub-menu. All the other transformations directly under the **Add Transformations** menu are general transformations. A detailed description of each transformation and its general usage can be found in Appendix B.

As noted previously, a 'pulse project' object consists of one or more transformation objects. Each pulse project object must start with a single create transformation but can contain zero or more general transformation tabs. Each transformation object contains results for that step in the RF pulse design.

For more detail on how to get started on creating and manipulating RF pulses, see **Section 2 – Quick Guide and the Case Study** section on pulse design in this manual.

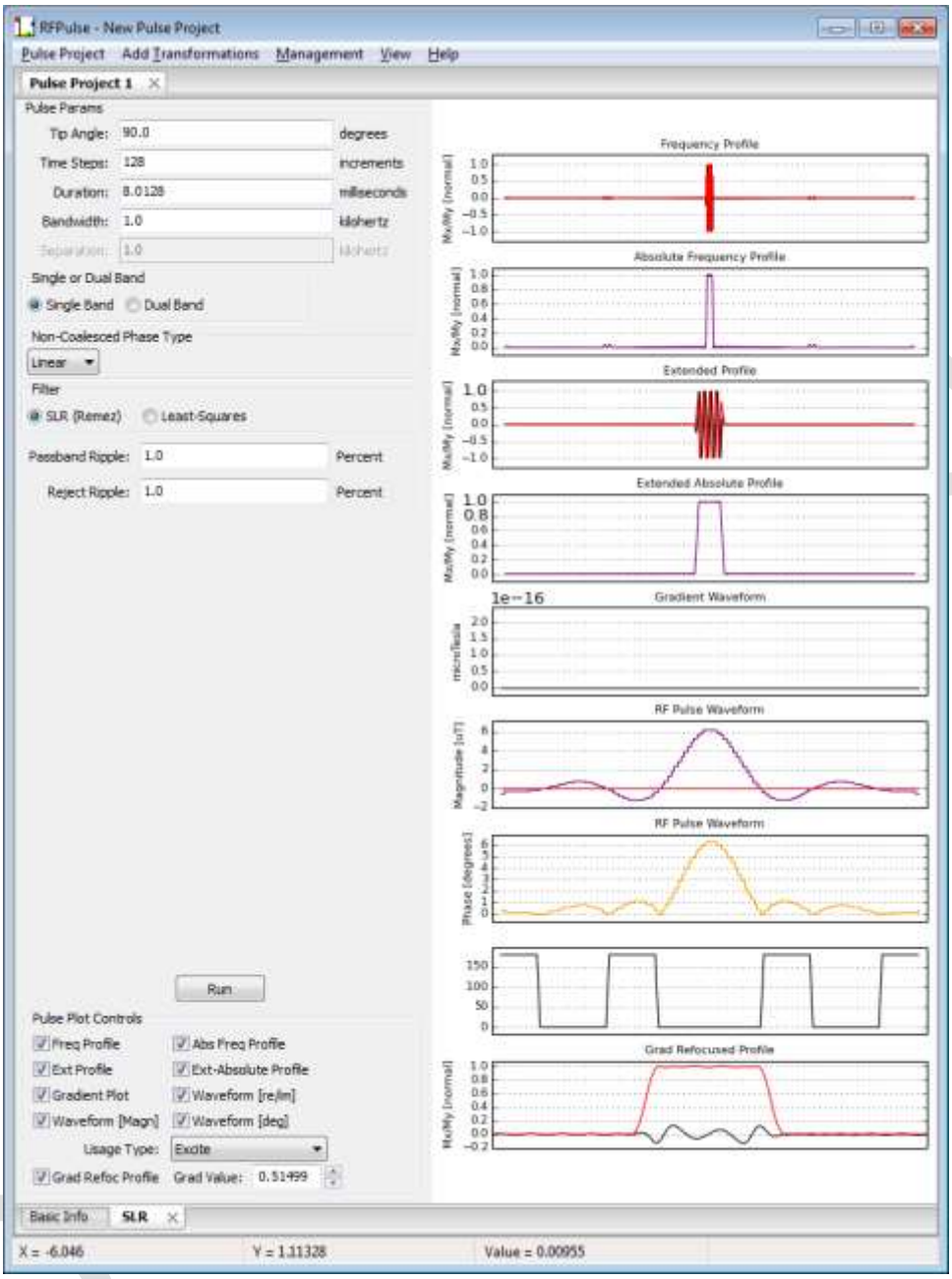
5.3 Visualizing Pulse Project Results

Transformations each contain a plot on the right hand side. This plot can display any (and all) of the eight results plots, in any combination, by selecting from the check boxes in the **Plot Control** panel on the left side of the tabbed window. Turning a check box on or off adds or removes the plot from the figure on the right.

You can modify other aspects of the data display from the **View** menu.

The plots displayed in each transformation tab are the cumulative results for all transformation tabs to the left of, and including, the current tab.

When a transformation tab is added to a pulse project, RFPulse won't display results until you click the Run button. Changing transformation parameters on the left panel does not typically change the results plotted until after the Run button is hit.



On the Plot Control Panel

Freq Profile (checkbox) Selects RF pulse frequency profile to be plotted. This result comes from running the RF waveform through the Bloch equations to calculate the complex RF profile at ±1 times the Nyquist frequency and at the calculation resolution set in the Basic Info tab.

Abs Freq Profile (checkbox) Selects RF pulse absolute frequency profile to be plotted. The magnitude plot of the frequency profile plot described above.

Ext Profile (checkbox) Selects the RF extended frequency profile to be plotted. This result comes from running the RF waveform through the Bloch equations to calculate the complex RF profile at ±4 times the Nyquist frequency and at the calculation resolution set in the Basic Info tab. This allows you to check a wider bandwidth range, albeit at a coarser spectral

resolution than for the non-extended profile. This allows you to view out-of-band excitation.

Ext-Absolute Profile (checkbox) Selects the RF extended absolute frequency profile to be plotted. This is the magnitude plot of the extended frequency profile plot described above

Gradient Plot (checkbox) Selects the time domain gradient waveform affiliated with the RF pulse to be plotted. If none needed for the given pulse, it plots zeros.

Waveform (checkbox) Selects the time domain RF pulse waveform to be plotted.

Waveform [Magn] (checkbox) Selects the magnitude format of the time domain RF pulse waveform to be plotted.

Waveform [deg] (checkbox) Selects the phase (in degrees) format of the time domain RF pulse waveform to be plotted.

Grad Refoc Profile (checkbox) **Note. This option only appears when the Usage Type is set to "Excite".** This profile can be used to visualize the residual phase in the M_{xy} plane after a refocus gradient has been applied to the pulse. When the plot is first opened, RFPulse attempts to automatically calculate the optimal rephrase gradient value. That is the floating point number (between 0.0-1.0) in the Grad Value field. The user can type in any value here to see the resultant M_{xy} waveform. When the user types 0.0 in this field, RFPulse will run the automated optimization calculation again.

The **Usage Type** drop list widget requires a more careful explanation. For any given RF pulse, running the complex time waveform through the Bloch equations results in a description of the magnetization along the Z axis as well as in the X,Y plane. Depending on what the pulse will be used for, you may want to check the pulse performance of either the M_z or $M_{x,y}$ and sometime both directions. Changing the Usage Type widget allows you to specify what is plotted along the vertical axis (a.k.a. the y-axis) in each plot in the figure.

Usage Type effects on Y-axis Choice

Excite The M_x (real part) and M_y (imaginary part, if turned on) magnetization is plotted in all profile plots, with the assumption of initial magnetization along the z axis.

Inversion The M_z magnetization is plotted in the real part and zeros in the imaginary part (if turned on) in all profile plots, with the assumption of initial magnetization along the z axis.

Saturation The $-M_z$ magnetization is plotted in the real part and zeros in the imaginary part (if turned on) in all profile plots, with the assumption of initial magnetization along the z axis.

Spin Echo The M_x (real part) and $-M_y$ (imaginary part, if turned on) magnetization is plotted in all profile plots, with the assumption of initial magnetization along the y axis

The mouse can be used to zoom in on the X and Y dimensions in all plots and to set vertical reference spans along the x-axis. The left mouse button draws a box which designates the region into which to zoom the plot. Click and drag the left mouse button in the window and a yellow box will appear. Drag the mouse to size the zoom box appropriately; release the left button and the plot will zoom into that region. XRange, YRange, plot value, and ΔX and ΔY values will be shown in the status bar while dragging. Click and release the left mouse button in place and the plot will zoom out to its max setting.

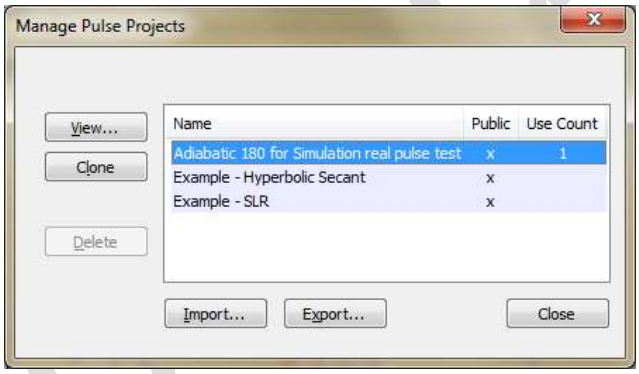
In a similar fashion, two vertical cursors can be set inside each plot window. Click and drag the right mouse button then release to set the two cursors anywhere in the window. This Cursor Span will display as a light gray span. Click and release the right mouse button in place and the cursor span will be turned off.

6. Management Dialogs

The Management dialogs allow you to create, delete, edit, import, export or view pulse projects and machine settings templates. These dialogs thus allow you to manage the data in the RFPulse database. It also provides the means for users to share information between themselves via XML files created using the Import/Export functions. Note.

6.1 Manage Pulse Project dialog

Access this dialog by clicking on the **Management→Manage Pulse Projects** menu item. The dialog opens and blocks other activity until it is closed. An example of this dialog is shown in the figure. Pulse project names are listed in the window on the right. You can View, Delete, Clone, Import or Export pulse projects. These functions are summarized below.



View: Displays a textual summary of the pulse project.

Clone: Copies the currently selected pulse project(s) to a new pulse project with a different unique id and name. The new pulse project is automatically saved in the database. This is typically used on pulse projects that have been designated as "public" in order to have a modifiable copy with which to work.

Delete: Removes the selected pulse project(s) from the database.

Import: Allows you to select an XML file that contains a pulse project. If the UUID in the file is unique, it is added to the pulse project database.

Export: You select a pulse project from the list. A second dialog allows you to browse for the output filename, select if output should be compressed and allows an additional export comment to be typed in.

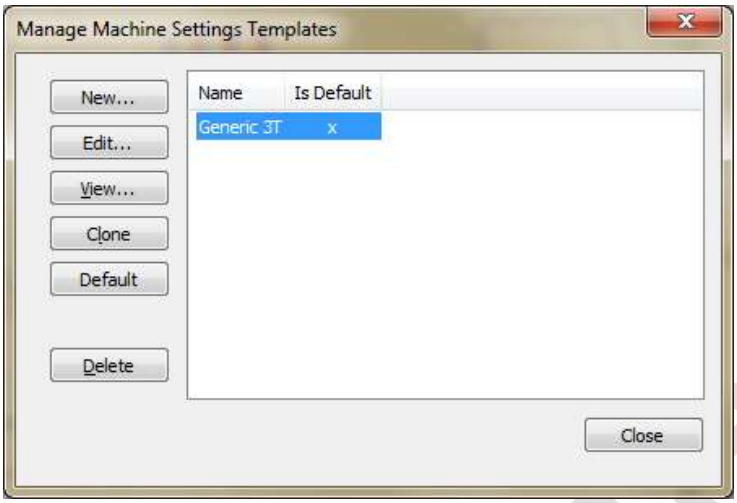
Under the Hood

- The action of exporting a pulse project (or other objects) caused it to be marked as "frozen" in the database. This means that no changes can be made. This is for the sake of consistency as results are shared. However, a frozen pulse project can still be deleted from the database if needed. This file can be imported into another Vespa-RFPulse installation using the Import function. If additional changes are desired a new pulse project, cloned from the original "frozen" object, can be created then modified.
- RFPulse PulseProject results can be used at part of a Simulation Pulse Sequence. The "Use Count" column above shows how many times that pulse is in use within Vespa. Because that pulse is a dependency for another application, it can not be deleted until all usages of it have also been deleted. However, it can still be cloned and import/exported.

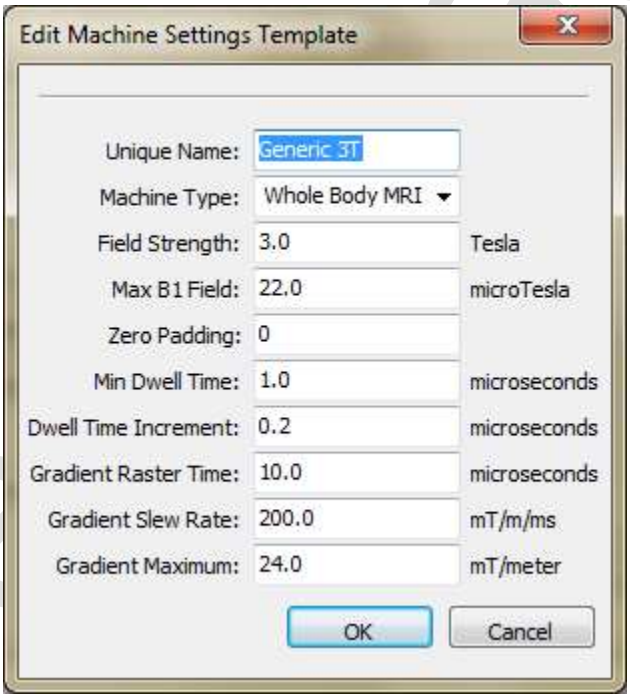
6.2 Manage Machine Settings Templates dialog

Access this dialog by clicking on the **Management→Manage Machine Settings Templates** menu item. Actions that can be taken on the dialog include: New, Edit, View, Clone, Set Default and Delete. An example of this dialog is shown below. The "Is Default" column indicates which template is used as the default settings for the machine settings object in a newly created pulse

project object. The Set Default button is used to designate the currently selected template as the default template. Only one template can be set as the default at one time.



New: A dialog will pop up that gives you a blank form to fill out. Fill in the settings in the widget fields and hit the OK or Cancel button. See the sample in the figure below.



Edit: The highlighted metabolite is opened in the machine settings editor. All fields are editable.

View: Similar to Edit but no fields are editable.

Clone: Select a template in the list, hit clone and a copy of that template is made that is now fully editable.

Delete: Deletes selected template(s).

Set Default: Used to designate the currently selected template as the default template that sets the machine settings in a new pulse project object.

7. Results Output

7.1 Plot results to image file formats

Results plots in each transformation tab can all be saved to file in PNG (portable network graphic), PDF (portable document file) or EPS (encapsulated postscript) formats to save the results as an image. The Vespa-RFPulse **View** menu lists commands that only apply to the active transformation tab in the active pulse project tab. Select the **View→Output→** option and further select the **Plot to PNG, Plot to PDF** or **Plot to EPS** menu item. You will be prompted to pick an output filename to which will be appended the appropriate suffix.

7.2 Plot results to vector graphics formats

Results in each transformation tab can be saved to file in SVG (scalable vector graphics) or EPS (encapsulated postscript) formats to save the results as a vector graphics file that can be decomposed into various parts. This is particularly desirable when creating graphics in PowerPoint or other drawing programs. At the time of writing this, only the EPS files were readable into PowerPoint.

The Vespa-RFPulse **View** menu lists commands that only apply to the active transformation tab in the active pulse project tab. Select the **View→Output→** option and further select either the **Plot to SVG**, or **Plot to EPS** menu item. You will be prompted to pick an output filename to which will be appended the appropriate suffix.

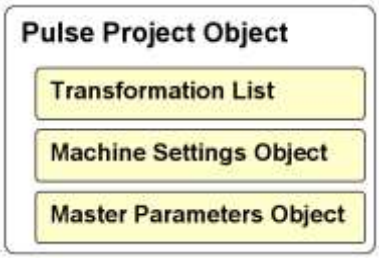
Appendix A. RFPulse Design

A.1 What is under the hood?

A.1.1 Vespa-RFPulse Basic Concepts

This is a combination of logical concepts and constraints that determine how RFPulse works. These rules are enforced through the application and, to some extent, the database.

The main objects in the system are pulse projects, transformations, machine settings and master parameters. There are two types of transformations, “create“ and “general“, which will be described in more detail later. Pulse projects are the primary objects; everything else is secondary. Here's how they're related --



- Each pulse project has one or more transformations. Transformations are the building blocks of a pulse project design. The pulse project object contains a list of all transformation objects that are part of the RF pulse design. The pulse project also contains the machine settings and master parameter settings used.
- Each transformation step of the design contains the results of the RF pulse at that step, as well as the parameters used to create that result. The pulse project can be saved at any time, but changes made in the GUI will not be copied into internal objects and thus saved until after the “Run” button is hit (for a given transformation).
- Each pulse project contains exactly one “create” transformation. This transformation is the first one selected by you, and is positioned as the second transformation tab in the GUI after the standard Basic Info tab.
- Each pulse project can add any number of general transformations after the create transformation.
- Each pulse project has only one set of machine settings.
- Each pulse project has only one set of master parameters.

We expect users to share data via RFPulse’s export and import functions. For this reason, the RFPulse pulse project object has a universally unique id (UUID) . Machine settings used in a given pulse project are stored in that project when it is exported and are thus available when imported.

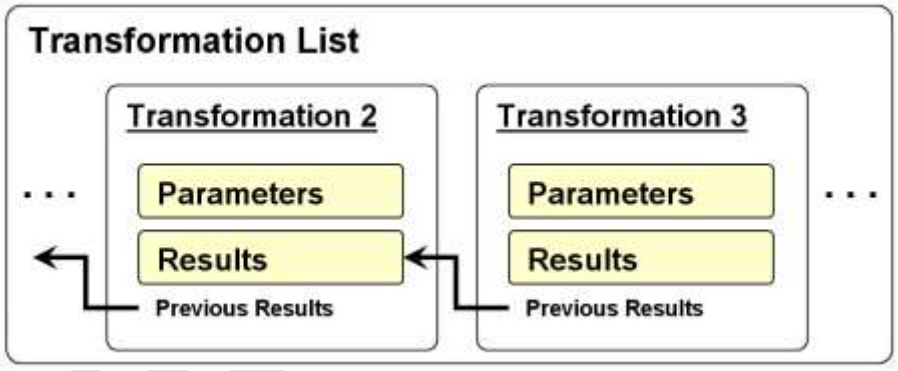
A.1.2 Pulse Projects

Pulse projects are the main focus of the RFPulse application. A Pulse Project’s *raison d'etre* is to help you design an RF pulse through a series of transformation steps. These transformations first create and subsequently modify an RF pulse to achieve the desired results. This set of transformations is the pulse project’s *transformation list*.

Currently, that list is implemented as a linear group of transformation objects. The first transformation object is always Basic Info. This is a special transformation object in that it occurs prior to any RF pulse waveform results being created. This object modifies meta-data within the pulse project and sets up global parameters that can affect pulse creation/modification algorithms in subsequent steps. Specifically, it allows you to select the Machine Settings and Master Parameters for the pulse design. The second transformation is always a create transformation that implements an RF pulse design algorithm, such as Shinnar-Le Roux or hyperbolic-secant, that creates an initial RF pulse waveform result. Subsequent steps always contain a general transformation which uses the waveform result of the previous step as an input to be modified by the current transformation step.

The transformation design was selected to allow more modularity in RF pulse design and increased flexibility in the display of intermediate results. Each transformation tab in the pulse project has a manageable number of parameter widgets in its GUI. These can be modified and applied to the local set of results without having to run all previous steps. Results are plotted in the transformation tab to allow you to visually inspect the effects of their settings for each design step.

The figure below is a visual representation of two steps in the transformations that make up an imaginary pulse project. Each transformation contains the parameters used in its algorithm, a set of results from running its algorithm, and a reference to the results in the previous transformation step (used as an input).



Transformations can be added or removed from the pulse project, typically by closing the associated tab in the GUI, with a few caveats. New transformation steps can only be added to the end of the current project. Any general transformation can be closed/removed, however this triggers an automatic recalculation of the steps “downstream” of that transformation. The create transformation can not be removed unless all general transformations have been previously closed/removed. This is due to the fact that the general transforms would have no results on which to act without an initial create transformation.

The take-home lesson from this section is that the Vespa-RFPulse application provides a modular design platform for optimizing RF pulses. Various steps in RF pulse creation and modification can be included in a given design, or not. In the appendix that follows, we will specify what each transformation step included does, are and how they are typically used.

Other useful information:

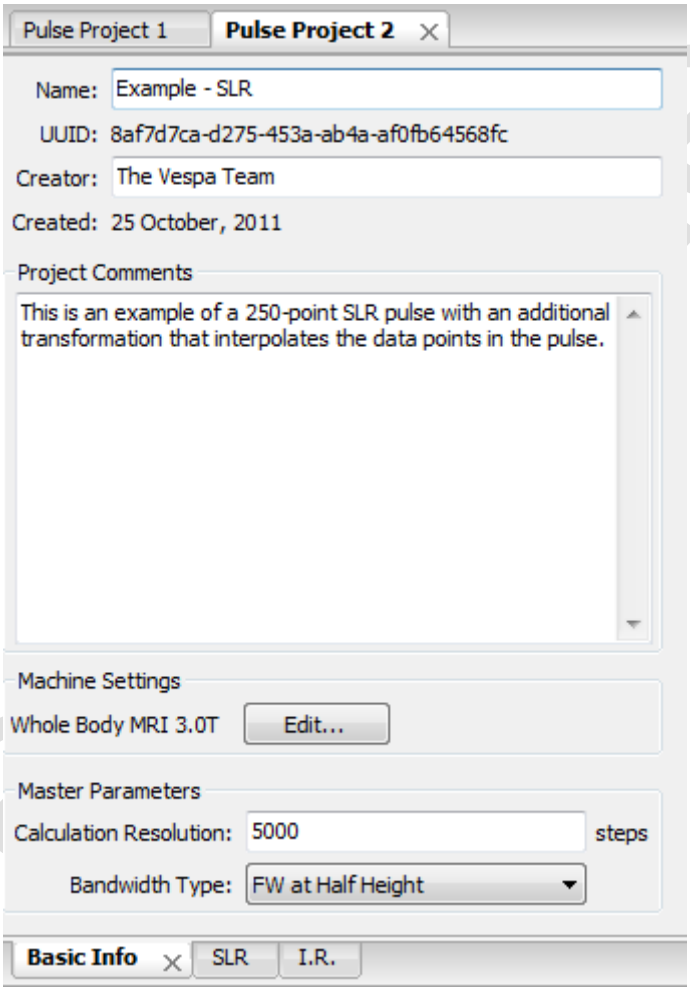
- **Anytime a tab's GUI is out of sync with the parameter values used for the last Run of the transformation, the tab's name will contain an asterisk ('*') next to it. Hitting Run will clear this.**

Appendix B. RFPulse Transforms

This section provides some basic information about the transformations (both “create” and “general” types) available in the RFPulse application. A description of the user interface and the parameters that can be set for each transformation is given. A brief overview of the algorithm applied when the Run button is hit is also presented.

B.1 Basic Info Tab

B.1.1 Tab Diagram



B.1.2 General Usage

This is a special transformation tab, neither “create” nor “general” type, which you must always fill in so that RFPulse can correctly identify the pulse project to/from the database. The name field MUST be filled in before the pulse project can be saved to the database.

B.1.3 Widgets and Parameters

Name – text, must be a unique text string within the group of pulse project names in the database

Investigator – text, (optional)

UUID and Created – not editable, just given FYI

Machine Settings – label, listing the name of the currently selected machine settings.

Edit – button, opens an editable dialog that displays the current machine settings for modification.

Calculation Resolution – integer, the number of points used in the Bloch equation calculations to create pulse profiles. More points result in a more finely sampled bandwidth but at the expense of longer calculation times.

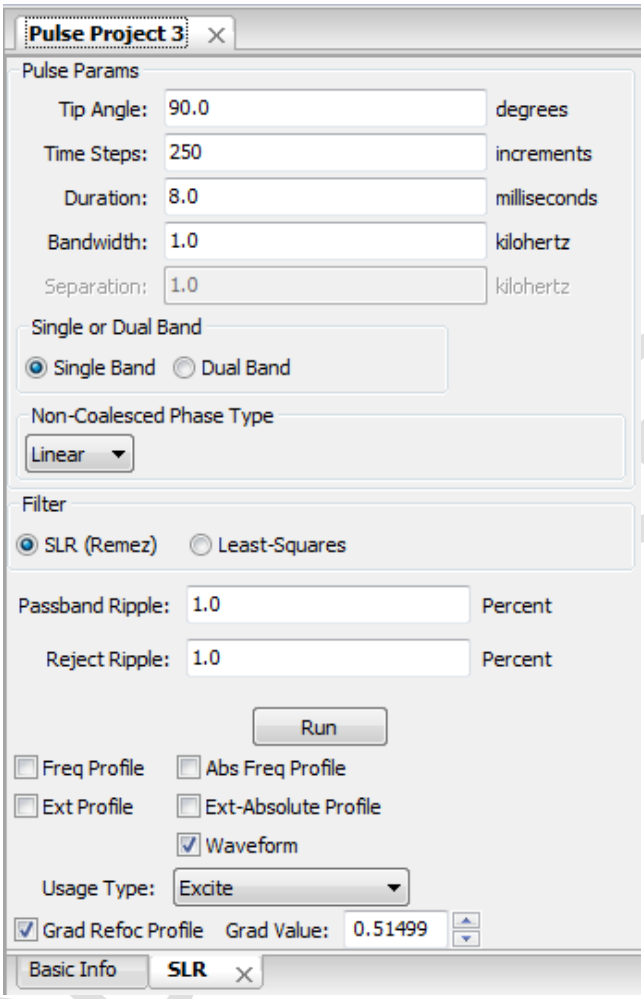
Bandwidth Type – droplist

B.1.4 Algorithm Applied at Run Time

There is no Run button on this tab. To proceed to the next step you select a “create” transformation from the **Transformation→Create** menu.

B.2 SLR Tab (create transformation)

B.2.1 Tab Diagram



B.2.2 General Usage

This is a “create” transformation tab. The SLR tab provides for creation of computer optimized SLR shaped (amplitude-modulated) RF pulses typically denoted as **B1**.

B.2.3 Widgets and Parameters

Tip Angle – float, effective tip angle of the pulse in degrees. As of this writing, this parameter is limited to shallow tip angles (just above zero, e.g. 0.001, to 30 degrees), 90 degree, or 180 degree angles. Note that the pulse is shaped for the specified tip angle. To see the effect of the pulse at the wrong power (that is, producing the wrong tip), the pulse must be re-scaled and the profile recalculated with the Bloch Equations.

Time Steps – integer, number of steps in the pulse. The number of pulse steps is usually selected in powers of two to speed the calculations. The profile shape is not improved by increasing the number of steps beyond around 128. However, the smoothness of the pulse (and minimization of out-of-band effects) is improved with more steps. For 90 degree pulses the minimum number of steps recommended is 64, while for 180 degree pulses the recommended minimum is 128. Note that the calculations are

more rapid for a small number of steps (e.g., 64 steps), and a larger number of steps may be selected after the other pulse parameters have been decided upon.

Duration – float, in milliseconds, total time of the pulse. Values selected may not be achievable due to the minimum dwell time and dwell time increment set in the machine settings. In this case, an achievable value will be suggested.

Bandwidth – float, in kHz, width of the pass band in kHz as measured at full width half max amplitude value.

Separation – float, in kHz (optional). This control is only activated if Dual Band is selected. This is the separation in kHz between the two passbands as measured at full width half max amplitude values.

Single or Dual Band – radio button, sets whether one or two passbands will be included in the design of the pulse. Single Band produces conventional slice-selective pulses. The Dual Band selection can be used to produce a dual band pulse such as a dual band saturation pulse.

Non-Coalesced Phase Type – dropbox, selections 'linear', 'max' or 'min'. Refocused (i.e. linear phase pulse) is generally preferred for an excitation or spin echo pulse, while Max Phase or Min Phase are preferable for inversion or saturation pulses

Filter – radio button, selects whether the SLR algorithm uses a Remez or least-squares optimization algorithm.

Passband Ripple – float, in percent, of the RF pulse profile. See notes below in 'Reject Ripple"

Reject Ripple – float, as a percent of the RF pulse profile amplitude. These 'ripple' parameters are used to estimate a transition band, which is the parameter actually used in the calculations. Although the ripple estimates are quite accurate for 180, 90, and very shallow tip pulses, they would be in error for pulses calculated at intermediate angles

B.2.4 Algorithm Applied at Run Time

When you hit the **Run** button …

The following description is based on content from the "Handbook of MRI Pulse Sequences, Bernstein MA, Kevin F, King KF, Xiaohong JZ, Academic Press, 2004".

The SLR algorithm was developed to solve the difficult inverse problem of finding what RF pulse to apply, given a desired slice profile and the initial condition of the magnetization. For small flip angles, the shape of an excitation pulse can be approximated by an inverse Fourier transform of the slice profile but this approximation breaks down for pulses in the range of 30-90° or larger.

Iterative numerical optimization methods can be used for large tip angles, but the SLR method allows for a direct, non-iterative solution in certain situations. Characteristics such as RF bandwidth, pulse duration, tip angle, percent ripple in the passband, and percent ripple in the stopband can be specified as well as acceptable tradeoffs between these parameters. The algorithm returns the exact RF pulse through a straightforward computational process.

The SLR algorithm uses two key concepts: The SU(2) group theoretical representation of rotations and the hard pulse approximation. Two typical ways of describing rotations in three-dimensional space are:

1) via 3 x 3 orthogonal rotation matrices and 3 x I vectors, i.e. via the special orthogonal 3D group SO(3)

2) via 2 x 2 unitary matrices, and 2 x 1 complex vectors called spinors, i.e. via the special unitary group SU(2) (Pauly et al.1991).

These two representations are completely equivalent ways of describing macroscopic rotations such as those experienced by the magnetization vector. But the SU(2) representation offers considerable mathematical simplification, and as a result is used in the SLR algorithm.

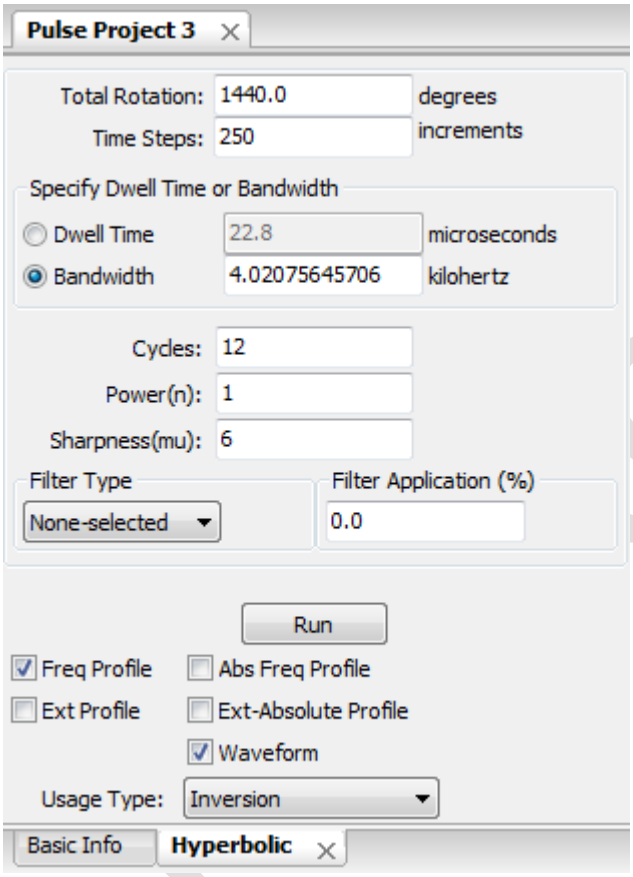
The second important conceptual component of the SLR algorithm is the hard pulse approximation (Pauly et al. 1991), which is useful in particular in the case of non-adiabatic pulses. The hard pulse approximation states that any shaped, or soft, pulse $B_1(t)$ can be approximated by a series of short hard pulses followed by periods of free precession. The larger the number of hard pulses used, combined with the resultant decrease in the duration of the free precision periods, the more accurate the approximation.

By describing rotations in the SU(2) representation and using the hard pulse approximation, the effect of soft pulses on the magnetization vector can be mathematically described by two polynomials with complex coefficients. The mathematical process that converts an RF pulse into the two polynomials is called the 'forward SLR transform'. It is important that the inverse SLR transform also can be calculated. The inverse transform yields the RF pulse, given the two complex polynomials corresponding to the desired magnetization. In digital signal processing (DSP), these polynomials are filters for which there are well-established and powerful design tools available. In the SLR algorithm, the inverse SLR transform is used in conjunction with finite impulse response (FIR) filter design tools to design RF pulses directly (Pauly et al. 1991).

Vespa-RFPulse uses these DSP tools to turn user input parameters into the appropriate complex polynomials and apply the inverse SLR transformation to generate a corresponding RF pulse.

B.2 Hyperbolic-Secant Tab (create transformation)

B.2.1 Tab Diagram



B.2.2 General Usage

This is a “create” transformation tab.

B.2.3 Widgets and Parameters

Total Rotation – float, in degrees,

Time Steps – integer, number of steps in the RF pulse

Dwell Time or Bandwidth – float, microseconds or kHz, respectively. Either the bandwidth or the dwell time must be provided (when one parameter is given, the other is automatically calculated). However, the width or dwell time calculations are only approximate

Cycles – float, number of cycles before truncating pulse.

Power(n) – integer, power of hyperbolic secant function

Sharpness(mu) – float, parameter that defines the sharpness of the function

Filter Type – droplist, apodization filter type – Hamming or Cosine

Filter Application – float, percentage, how much to apply the filter (0.0 to 100.0)

B.2.4 Algorithm Applied at Run Time

When you hit the **Run** button …

Hyperbolic secant pulses are adiabatic pulses that are typically used for inversion and are relatively insensitive to inhomogeneities in the B1 field and transmitter power setting, and result in very uniform frequency profiles. For adiabatic pulses both the amplitude and frequency of the pulse are typically varied. The pulses are adiabatic in that the amplitude is large enough and the frequency variation is slow enough (the adiabatic condition) that the longitudinal magnetization follows the direction of the effective magnetic field generated by the combined B0 and B1 fields. More specifically the adiabatic condition is that the precession of the magnetization vector about the effective field vector is more rapid than the change in angle of the effective field vector.

For a hyperbolic secant inversion pulse the initial effective field is aligned with B0 and the final effective field is aligned in the opposite direction. Hyperbolic secant pulses have the remarkable property that once B1 is strong enough, inversion is achieved over a frequency selective range (slice) such that further increases in B1 do not result in increases in flip angle.

Due to the frequency dependent phase that is generated by a hyperbolic secant pulse, they can not be used as refocusing pulses. A disadvantage of hyperbolic secant pulses and adiabatic pulses generally is that they typically involve high SAR, though as described in the case studies section above there are techniques for lowering the SAR associated for a given hyperbolic secant pulse.

Hyperbolic secant pulses constitute one of the few analytic solutions of the Bloch equations (Silver MS, Joseph RI, Hoult DI, "Selective spin inversion in nuclear magnetic resonance and coherent optics through an exact solution of the Bloch-Riccati equation", Phys Rev A, 1985, Apr;31(4):2753-2755.) and are actually solitons (yes, this is a real word) or non-dispersive solutions. For power(n)=1 they have the form:

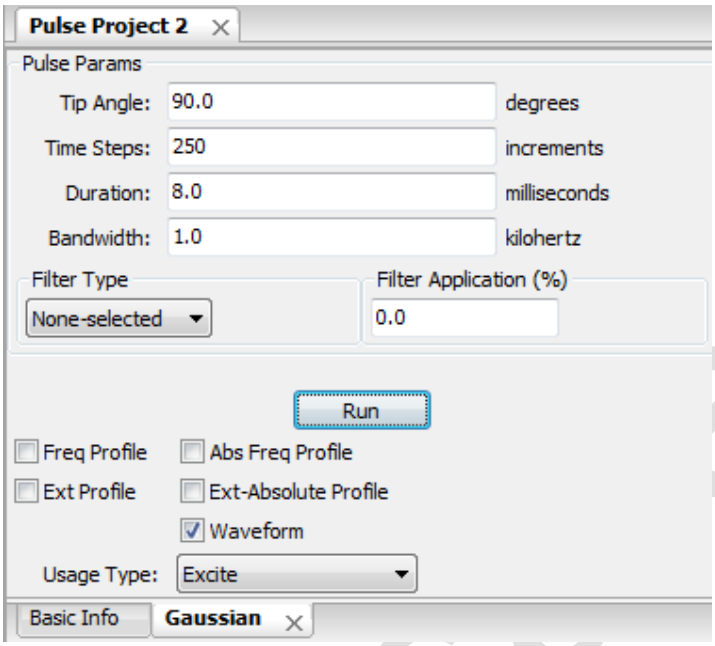
$$B_1(t) = A(t)e^{-i\omega_1(t)t}$$
$$A(t) = A_{ii}\,\mathrm{sech}(\beta t)$$
$$\omega(t) = -\mu\beta\tanh(\beta t)$$

where *A(t)* is the modulated, time dependent amplitude and $\omega(t)$ is the modulated, time dependent frequency, and μ and β are parameters. For the form of higher order hyperbolic secant functions see Tannus A and Garwood M, "Improved Performance of Frequency-Swept Pulses Using Offset-Independent Adiabaticity", 1996, J. Mag. Resonance, A 120 133-137.

RFPulse takes the specified user input parameters and generates a hyperbolic secant pulse of the above form.

B.3 Gaussian Tab (create transformation)

B.3.1 Tab Diagram



B.3.2 General Usage

This is a “create” transformation tab. The Gaussian tab provides for creation of filtered or un-filtered Gaussian shaped RF pulses typically denoted as **B1**.

B.3.3 Widgets and Parameters

Tip Angle – float, effective tip angle of the pulse in degrees. Note that the pulse is shaped for the specified tip angle at the center of the pulse. To see the effect of the pulse at the wrong power (that is, producing the wrong tip), the pulse must be re-scaled and the profile recalculated with the Bloch Equations.

Time Steps – integer, number of steps in the pulse. The number of pulse steps is usually selected in powers of two to speed the calculations. The profile shape is not improved by increasing the number of steps beyond around 128. However, the smoothness of the pulse (and minimization of out-of-band effects) is improved with more steps. Note that the calculations are more rapid for a small number of steps (e.g., 64 steps).

Duration – float, in milliseconds, total time of the pulse. Values selected may not be achievable due to the minimum dwell time and dwell time increment set in the machine settings. In this case, an achievable value will be suggested.

Bandwidth – float, in kHz, width of the pass band in kHz as measured at full width half max amplitude value.

Filter Type – drop-down menu. Selections are None, Cosine and Hamming. This control is used to select the type of apodization filtering applied to the calculated Gaussian waveform in the time domain. This selection is used in conjunction with the Filter Application (%) widget, see below.

Filter Application (%) – float, sets the extent, in percentage of all points in the time waveform, on which the apodization filter is to be applied. This filter can be used to smoothly transition a pulse to zero at the outer edges.

B.3.4 Algorithm Applied at Run Time

Overview – The primary advantage of Gaussian-shaped RF pulses is that they have a simple form and are easy to generate. Historically, Gaussian pulse cascades have been used in spectroscopy experiments for water suppression. However, Gaussian pulses do not produce linear responses, and also do not generate sharp selective profiles. Thus, with SLR pulses now easy to generate, there is little incentive to continue to use Gaussian-shaped pulses.

But, having said that, here is what you get from this Create transformation when you hit the **Run** button …

The general formula for a Gaussian shape is:

$$y(t) = exp(-[t / (2*sigma)]^2)$$

where sigma is one standard deviation for the data. We typically are defining the bandwidth of our pulses as the Full Width at Half Height. So, for a normalized pulse, when y = ½ we can solve for the time (*t*) that yields a given bandwidth = sigma.

This formulation holds true for a Gaussian pulse that is put through a Fourier transform, and for small tip angles, this approximation holds true for the Bloch Equation processing as well. As tip angles get larger (20 – 180 degrees or more) this approximation breaks down and the expected bandwidth is not typically achieved using the closed form.

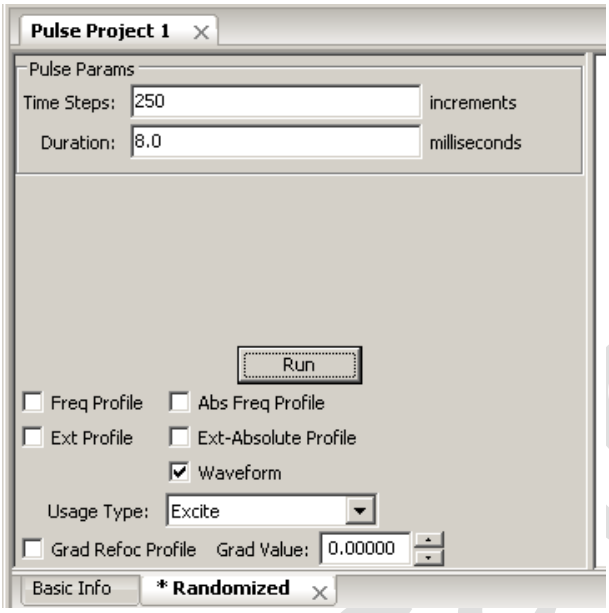
However, with the output plots that can be displayed for M_{xy} and M_z magnetizations, users can iteratively dial in a bandwidth that achieve the actual FWHM bandwidth desired.

Note. That for angles greater than 90 degrees, users may want to display results in the Inversion Mode to measure the FWHM tip angle for the longitudinal magnetization, M_z.

All Gaussian Pulses are created to be symmetric. For an odd number of points, the pulse maximum is located at N/2, for even numbers of points, the central two points N/2 and (N/2)+1 share the same value. This is achieved by time shifting the x-axis by half the width of one dwell period when digitizing the waveform.

B.4 Randomized Tab (create transformation)

B.4.1 Tab Diagram



B.4.2 General Usage

This is a “create” transformation tab. The primary purpose of this create transformation is to provide a starting input for the Optimal Control methods transformation. The Randomized tab provides for creation of a complex vector of random points in the range of +/- 1 micro Tesla.

B.4.3 Widgets and Parameters

Time Steps – integer, number of steps in the pulse.

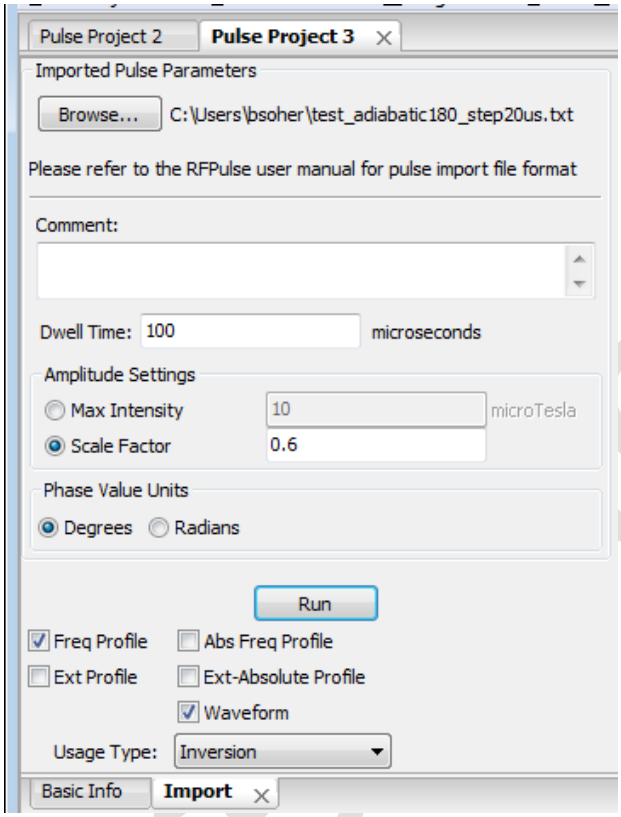
Duration – float, in milliseconds, total time of the pulse. Values selected may not be achievable due to the minimum dwell time and dwell time increment set in the machine settings. In this case, an achievable value will be suggested.

B.4.4 Algorithm Applied at Run Time

Overview – The primary purpose of this create transformation is to provide a starting input for the Optimal Control transformation. The Randomized tab provides for creation of a complex vector of random points in the range of +/- 1 micro Tesla.

B.5 Import from File (create transformation)

B.5.1 Tab Diagram



B.5.2 General Usage

This is a “create” transformation tab. Users can use this “create” tab to load a waveform from a text file into a PulseProject in RFPulse where it can be examined and stored. The format for the text files are given below. Note – there is an additional “Comment” field provided in this tab, it is highly recommended that it be filled in with details as to the origin of the waveform being loaded. This will serve as the only provenance that this project will maintain for how this pulse was created.

B.5.3 Widgets and Parameters

Browse (and filename label) – button, allows users to browse to find the file to be imported.

Comment – text, a text description of where the imported pulse came from. Note – this is the only provenance that will be kept regarding the source of this waveform. It is a separate comment from the main comment in the Basic Info tab. Use it!

Dwell Time – float, in microseconds, duration of each step in the imported pulse.

Max Intensity/Scale Factor – radio button, this is an exclusive selection for the method by which the RF pulse waveform amplitudes will be treated. The “Scale Factor” method will scale the amplitude values, being read in (e.g. microTesla values), by a floating point scale factor designated in the field to the right. The default value for this method is 1.0, which will in actuality have no effect on the amplitude of the imported waveform. The second method for scaling the imported waveforms amplitudes is called “Max Intensity”. After the waveform is read in from file, it is normalized to the max term in it and then scaled by the floating point microTesla amount listed in the field to the right.

Phase Value Units – radio button, this is an exclusive selection for the units of the phase values stored in the second column of the imported file. The choices are either degrees (default) or radians.

B.5.4 Algorithm Applied at Run Time

Overview – This create transformation allows users to import waveforms of their own making into the RFPulse application environment for examination and extension. Because nothing will be known inherently about the origin of the imported pulse, there is a separate comment field in this tab to allow users to comment on where this waveform came from and how it was made. This will be the **only** provenance kept for this project, so please fill in as much information as possible.

Import File Format Description

The following describes the file format that must be followed in order to import a waveform:

- The file must exist.
- It is expected to be a text file
- Lines beginning with ';' or '#' are considered comment lines and are ignored
- Comments can NOT be included on the same line as waveform points, but can be interleaved between waveform lines (though that is not recommended)
- Lines with only whitespace are ignored
- Waveform information should be included in the file as follows:
    - One point of the waveform is on each line
    - Each point consists of an amplitude value and phase value separated by space(s)
    - Amplitude values are in microTesla
    - Phase values are in degrees OR radians (default is in degrees)
    - Waveform points are assumed to be equally spaced as described by the "Dwell Time" field in the transformation tab

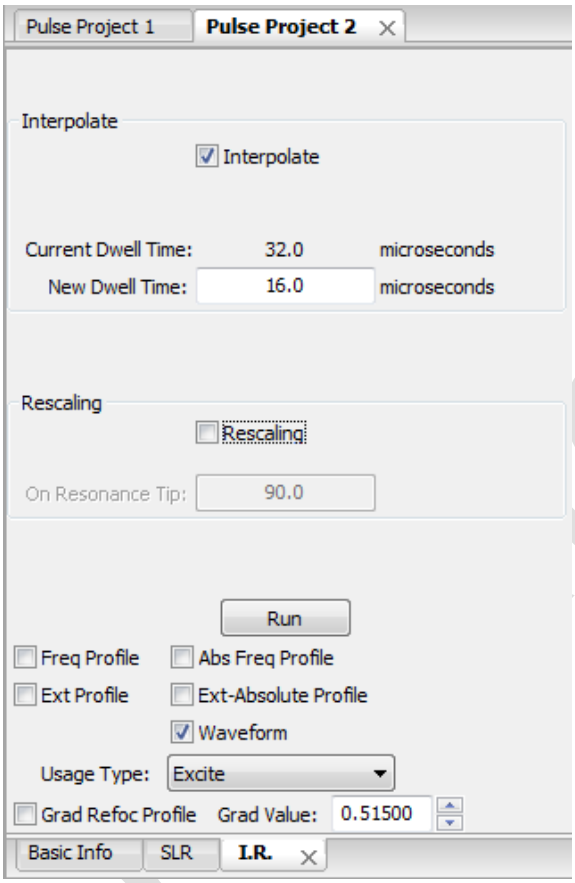
At run time, a number of values are checked both in the widgets in the tab and whether any values for a waveform can be read in from the listed file. When the Run operation is done, the waveform and processed Bloch equation results for Mxy, Mz can be viewed in the plot canvas on the right.

Other useful information:

- The comment line is only updated after the Run button is hit. So, making changes to it without hitting Run will result in their loss when the project is closed.
- When the user Browses for a file name and then hits OK, the Import tab is marked as "out of sync" as indicated by an asterisk in the transformation tab. This is the case even if the user selects the same filename as before. Only if the user hits the Cancel button within Browse will the tab remain in sync. Hitting the Run button will reload the file into the PulseProject and restore the tab into sync.
- Because this "create" transformation only reads in finished results from some other source, it may be possible for the waveform amplitudes listed in the file to exceed the Machine Settings set on the Basic Info tab. This can be worked around by creating new Machine Settings or by scaling the imported waveform using the "Max Intensity" method.

B.6 Interpolate-Rescale Tab (general transformation)

B.6.1 Tab Diagram



B.6.2 General Usage

This is a "general" transformation tab. It modifies the results from the previous tab. A "create" tab must be added to the project prior to using this transformation.

B.6.3 Widgets and Parameters

Interpolate – checkbox, turns Interpolation algorithm on/off.

Current Dwell Time – float, the dwell time from the previous results, i.e. the results from the previous transformation (or tab). If there are no previous results, then this value will be zero.

New Dwell Time – float, the target dwell time for the interpolation. If this new dwell time violates the minimum dwell time or the dwell time increment (set in the Machine Settings), an error is reported.

Rescaling – checkbox, turns Rescaling algorithm on/off.

On Resonance Tip – float, equal to the new value for either the net rotation or total rotation.

B.6.4 Algorithm Applied at Run Time

When you hit the **Run** button …

Interpolate will perform a linear interpolation to recalculate the waveform. The new dwell time will be used to create the time axis, and the number of time points will be changed to span the duration - up to the last point. The interpolation routine is performed in one direction and assumes the points are at the beginning of each time slice.

For data points added after the last point in the waveform, the last B1 value will be used (extended).

Rescaling changes the on-resonance tip angle (either the net tip angle or the total tip rotation) of the pulse by rescaling the amplitude of the pulse.

Net angle is calculated as: (integral of the waveform)*gamma

Total rotation is calculated as: (integral of the absolute value of the waveform)*gamma

The choice of net tip angle or total tip rotation is determined by this rule:

- If the sum of the absolute values of the imaginary components of the waveform (B1) is greater than 0.01, then it uses the total rotation, otherwise it uses net angle.

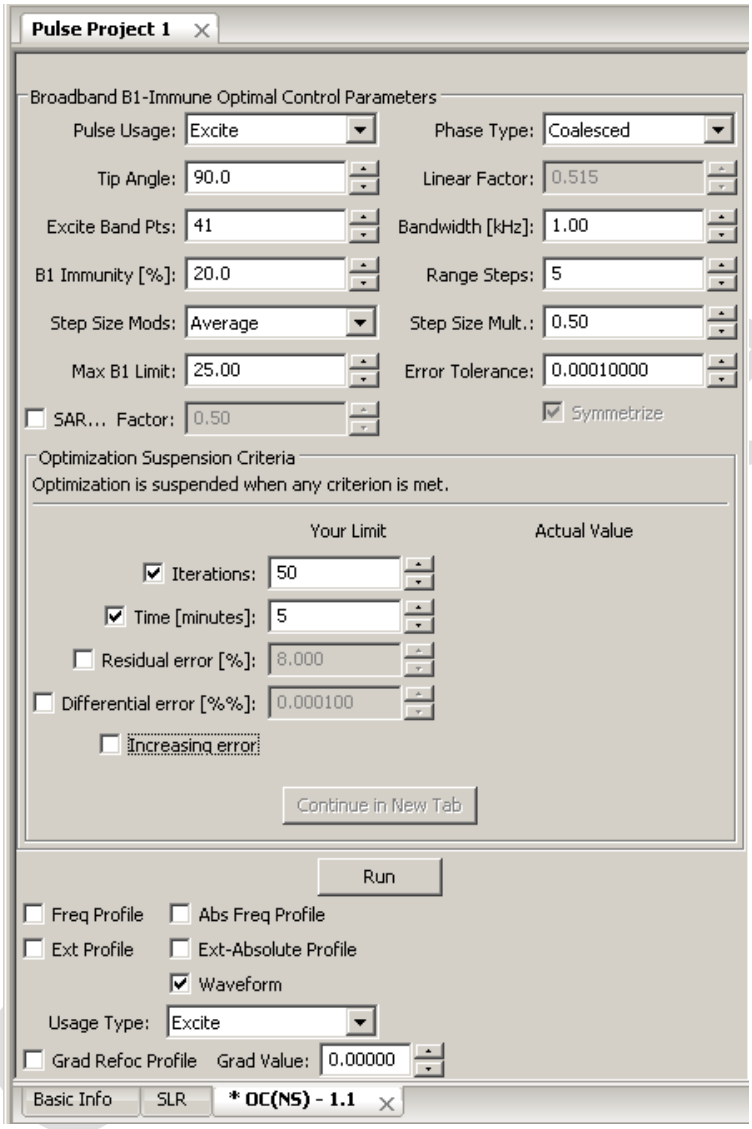
For analytic pulse types (e.g. the Hyperbolic-Secant) rescaling can be used to change the amplitude of the pulse for various magnetic fields, by changing the total rotation significant amounts if needed.

Amplitude modulated pulses (e.g. SLR pulses) can only be adjusted by a few percent before they start to degrade in performance. In this case, rescaling can be used to see how much it degrades, so one can see how it performs under various intensities - that may be caused by susceptibility issues, limitations in the field coils, etc.

Note: For amplitude modulated pulses (such as an SLR pulse) the tip angle specified on the interpolation and rescaling transformation widget (the on-resonance tip) is the tip angle at the center of the profile. However, due to ripples in the passband, the tip angle at the profile center is usually at an extremity (either maximum or minimum) of the allowed tip angle as specified by the ripple amplitude set by the user when the pulse was designed. Thus, the tip angle at the center of the profile may be different from the tip angle used for the design of the pulse. This, however, does not invalidate the initial design, which provides for the correct average tip angle over the profile for which the pulse was created.

B.7 Optimal Control – Non-Selective Tab (general transformation)

B.7.1 Tab Diagram



B.7.2 General Usage

This is a “general” transformation tab. It modifies the results from the previous tab. A “create” tab must be added to the project prior to using this transformation.

Optimal control methods (a.k.a. OC) can be used to significantly improve and/or extend the functionality of an existing RF pulse. As such, it needs a ‘pulse waveform’ with which to start. This could be the output from an SLR transform, a hyperbolic secant transform, or even just a waveform with random numbers (hence, the Randomized create tab). The usage of this transformation is quite different from other transformations, so please take the time to read the documentation below both on why to use optimal control, what the widget settings affect, and what the underlying algorithm contains.

The major difference between this transformation and other general transformations is that OC is typically a multi-step process. The OC optimization does not have well-defined stopping criteria, that is, it does not have an automated way of telling when the waveform best meets the user’s specifications. In the end, the

decision to stop is based on the user's iterative manual examination of the state of the result. A typical OC session looks like this:

set parameters → start OC run → stop and examine result → change parameters → continue OC run → repeat …

In RFPulse, each repeat of the steps above will be displayed as a new tab. This enables you to determine the provenance for how each pulse ultimately is created.

Add the first OC Non-Selective tab to the pulse project from the Transformations menu. The new tab's label will read "OC(NS) 1.1" Set up the parameters and stop criteria based time, iterations and/or residual errors. This first tab is started using the Run button as usual.

When the optimization stops, results are shown in the plot window to the right which can be viewed interactively. At this point the "Continue in New Tab" button at the bottom of the tab becomes active. If you click this button, a new tab is created to the right of the current tab and its label is set to "OC(NS) 1.2". As further iteration are performed, the tab labels increment 1.3, 1.4, etc.

At any point, you may decide that the current results are sufficient and stop hitting the "Continue" button. At this point, another general transformation could be added to the project. At any point in the OC optimization iteration, you can go back one or more OC iterations (tabs) and delete that tab. As when deleting any tab, this will destroy the results in downstream tabs. But, this would be one way to "restart" the optimization from a mid-point result and take it in a different direction by altering the optimization parameters.

Note. The optimal control methods in this transformation are described in a 2009 Journal of Magnetic Resonance publication by Matson, et.al. However, this paper is freely available to the public at:

http://www.ncbi.nlm.nih.gov/pmc/articles/PMC2724660/?tool=pubmed

Much of the documentation below has been summarized from this publication or kindly provided directly from Dr. Jerry Matson at UC San Francisco.

Practical Note – in discussions with Dr. Matson, he said that in a typical OC session, he will set up his optimization parameters (in MatPulse) and then let it run for a few minutes. He then suspends the run and checks on the progress. If it appears that it is heading in the right direction, he then continues the optimization for a longer period. Some of his optimizations he has let run for hours or even overnight to optimize to a point that he is satisfied with the results. This is a very subjective comment and only meant to give you an indication of how long these optimizations *could* run.

So, why would you want to use OC?

This optimal control routine for non-selective pulses (originally developed in in MatPulse and since ported to Vespa-RFPulse) was developed with non-slice-selective (NSS) pulses in mind. The strength of the design method is that pulses with a selected immunity to B1 inhomogeneity, and with immunity to a selected range of resonance offsets, can be designed. Examples of such pulses suitable for MRI sequences such as 3D MP RAGE are presented in reference 1 (1). Other authors have demonstrated similar pulses, although not generated by optimal control (2,3).

However, the optimal control routine is useful for many other kinds of non-selective pulses that operate over a range of B1 strengths and resonance offsets, such as those typically used in high resolution spectroscopy (2-6). In addition, the routine may be initiated with hyperbolic secant pulses to develop efficient non-selective inversion pulses (1), and may also be used to design non-selective spin echo pulses, again with selected immunities to B1 inhomogeneity and to resonance offset.

The number of frequency points needed for the optimization can be highly variable. For hyperbolic secant style inversion pulses, where the profile is typically rather smooth, a small number of frequency points can be sufficient. On the other hand, high resolution spectroscopy excitation and inversion pulses can have rapidly oscillating profiles, and may require a large number of points. Viewing the plotted waveform and profile plot of a partially optimized pulse can provide a guide on whether a sufficient number of frequency points are being used.

The number of points needed for the pulse itself is also highly variable. Again, the smoothness (or lack of smoothness) of a partially optimized pulse can be a guide on whether a sufficient number of points to define the pulse are being used.

A pulse that does not seem to converge to a useable design probably needs a longer dwell time, or a larger B1 amplitude, or both. For a pulse that converges quickly, the dwell time or maximum B1 strength may be able to be reduced.

A description of the controls for this method is given in the next section with brief descriptions of the aspects of the optimization that they control.

References

1. Liu, H, Matson GB. Radiofreuqency pulse designs for three-dimensional MRI providing uniform tipping in inhomogeneous B1 fields. Magn Reson Med 2010;66(5):1254-1266.

2. Boulant N, Mangin JF, Amadon A. Counteracting radio frequency inhomogeneity in the human brain at 7 Tesla using strongly modulating pulses. Magn Reson Med 2009;61(5);1165-1172..

3. Moore J, Jankiewicz M, Zeng H, Anderson AW, Gore JC. Composite RF pulses for B1+-insensitive volume excitation at 7 Tesla. J Magn Reson 2010;205(1):50-62.

4. Skinner TE, Kobzar K, Luy B, Bendall MR, Bermel W, Khaneja N, Glaser SJ. Optimal control design of constant amplitude phase-modulated pulses: application to calibration-free broadband excitation. J Magn Reson 2006;179(2):241-249.

5. Skinner TE, Reiss TO, Luy B, Khaneja N, Glaser SJ. Reducing the duration of broadband excitation pulses using optimal control with limited RF amplitude. J Magn Reson 2004;167(1):68-74.

6. Skinner TE, Reiss TO, Luy B, Khaneja N, Glaser SJ. Application of optimal control theory to the design of broadband excitation pulses for high-resolution NMR. J Magn Reson 2003;163(1):8-15.

B.7.3 Widgets and Parameters

Pulse Usage – drop menu, [Excite, Saturation, Inversion, Spin-echo], default is Excite. This is how you will use the pulse you are making. What you select from the drop down affects what is displayed as choices from the "phase type", i.e. which of these choices {Coalesced, Non-Coalesced, Linear Factor} is available. This information is also used as part of the optimization.

Phase Type – drop menu, [Coalesced, Non-coalesced, Linear]. The choices that are displayed depend on what Pulse Usage is selected:

- For **Excite** we have two choices:
  - Coalesced: The phase of all magnetization in the transverse plane is the same.
  - Linear: Implies that there is a linear distribution of phases in the transverse plane, i.e. that could be refocused using a gradient. NOTE: Linear is a type of Non-Coalesced pulse (the other types are Min and Max phase).
  - Linear Factor (float): Default 0.5. This is thought of as the fraction of the area under the gradient waveform - applied during the pulse - that needs to be applied in the opposite direction to achieve refocusing. Mathematically it is the numerical 'factor' required to obtain refocusing in the expression:

    $$\text{profile} \times \exp(-i\, 2\pi\, f\, \tau\, \text{factor})$$

- For all the other pulse usages we don't have a choice. i.e.
  - Saturation → Non-Coalesced
  - **Inversion** → Coalesced
  - **Spin-echo** → Coalesced

Tip Angle – float, in degrees, The target tip angle for this optimization. Defaults to 90 (degrees).

Linear Factor – float, unitless (optional). A value applied to optimization when ‘Phase Type – Excite’ pulse usage is selected. See above.

Excite Band Pts – int, default 41, the number of points used to generate ‘zero error profiles’ used in the optimization to represent the desired pass-band performance. The more points used, the more comparisons need to be made, but using too few points might results in undersampling the results in the excitation band.

Bandwidth – float, in kHz, width of the pass band in kHz as measured at full width half max amplitude value.

B1 Immunity – float, percentage. This, along with the “Range Steps” field, determines the specific profiles that will be fit for this optimization. It starts at $-B1_{immunity}$ with steps size of $2*B1_{immunity}/(Steps -1)$, all the way up to $+B1_{immunity}$. For example if Steps=5 and B1 Immunity Range is 20 then OC will use 5 profiles at -20, -10, 0, 10, and 20. OC will minimize the average residuals over these profiles (for the number of Excitation Band Points specified). Default value for B1 Immunity Range is 20.

Range Steps – integer, see above.

Step Size Mods – drop menu, [Average, Fixed, Compute], default is Average. This is the method that the optimization used to modify the step size during computation. Average: The new step size multiplier will depend on the percent success of last 100 steps (what percent led to and increase or a decrease in the residual errors). Fix Size: the step size multiplier will not be changed. Compute: Jerry's observation is that this did not work very well. Interacts with Step Size Mult, see further explanation below.

Step Size Mult – This is a multiplier that is used to decide the step size to take along the minimization gradient to get to the next point (for checking residuals). Typical usage, or suggested, range is (0.1 to 1.0) although larger values are allowed. This value is called Step Size Multiplier to indicate the fact that while this number may not change as you get close to converging the actual step size will get smaller. It is actually used as a multiplier for the fitting process and not as the actual step size. How this might change during the fit is determined by choice of [Average, Fixed Size, or Compute].

Max B1 Limit – float, in microtesla, default 25 uT. Used as a cutoff value for the maximum B1 field. If the pulse output from the current iteration has fields above the max, the pulse will be "topped" at this value. The alternative would be to put a fitting penalty on values above the max, but this apparently works as well, perhaps by not giving any advantage to going higher that the max.

Error Tolerance – float, The maximum allowed fractional residual increase (or something close). Range is 1e-6 to 1e-3. Default is 1e-4.

SAR … – check box, used to indicate application of additional subroutines, that take the SAR Factor as an input. Jerry was not happy with the results this gave and perhaps someone else may have a better way of applying this. Defaults to not checked.

SAR … Factor – float, default 0.5, input to 'Limit SAR' subroutines. Ignored unless 'Limit SAR' is checked. Range is 0.1 to 1.0.

Note. The optimization suspends when it meets any of the five criteria described below. These criteria are not mutually exclusive.

Iterations – integer, unitless, check box and integer value. Defaults to checked and 50. If selected it uses the number specified to determine how many iterations should be executed before the optimization is finished. This is an absolute number, i.e. it is not in units of 100 iterations, etc.

Time – integer, in minutes, check box and integer value. Defaults to checked and 5 minutes. If selected it uses the duration specified to determine if the optimization should stop.

Residual Error [%] – check box and percentage. Tells optimization to stop once the residual error is below a certain %. Default of check box is unchecked. Data input field default is 0.2 (percent). Real Number.

Differential Error [%%] – check box and percentage. Default unchecked and 1e-4 (percent of a percent). Tells optimization to stop if the error has not changed by more than the specified amount in the last 200 iterations.

Increasing Error – check box, default is unchecked. Indicates that the optimization should be stopped if the error begins to increase. It looks at the last 200 (to 400) iterations.

Note. When a criterion is reached and the optimization stops, the current values for all five criteria are listed in their respective "Actual Value" column. These numbers can be useful for determining the progress of the optimization and which/whether parameters should be changed.

B.7.4 Algorithm Applied at Run Time

When you hit the **Run** button …

The following description is based on content from the 2009 Journal of Magnetic Resonance publication by Matson, et.al. However, this paper is freely available to the public at:

http://www.ncbi.nlm.nih.gov/pmc/articles/PMC2724660/?tool=pubmed

In early MRI development, optimal control methods provided one of the most effective means for design of frequency selective RF pulses. Although optimal control methods have continued to be used for some RF pulse applications, the much more efficient Shinnar Le-Roux (SLR) methods became the RF pulse design method of choice for MRI and in vivo spectroscopy applications that required frequency selective RF pulses. Despite the effectiveness of the SLR design, there are a number of RF frequency selective pulse design criteria that are not amenable to the SLR method. In general, the SLR method does not provide for extension of the pulse bandwidth without increase of $B1_{max}$.

Recently there has been resurgence in the use of optimal control methods for the design of RF pulses, promoted primarily by the research groups of Glaser and Skinner. The Glaser and Skinner groups (G&S) have collaborated to both promote the efficacy of optimal control methods and demonstrate improved, broadband excitation by optimal control (BEBOP) pulses and broadband inversion by optimal control (BIBOP) pulses. G&S recently demonstrated that excitation pulses could be made more efficient (shorter) by allowing a linear spread of phase with frequency. If the phase spread is small, it can be corrected by conventional processing (first order phase correction). In addition, this type of pulse is useful in spin echo experiments, where the timings can be adjusted to re-coalesce the magnetizations at the time of the echo. Among their innovations, they demonstrated that limiting the peak power (or $B1_{max}$) could be incorporated into the optimal control method, and that the cross product of the forward magnetization and vector Lagrange multiplier could be used to correct the B1 field.

The OC algorithms developed by Matson et.al., and that are used in the Vespa-RFPulse application, demonstrate that optimal control methods as described by G&S can be extended to either improve the bandwidth and selectivity of conventionally designed frequency selective pulses under conditions of limited available $B1_{max}$ and fixed length, or reduce the length of the pulse while maintaining the bandwidth and selectivity.

The optimal control method is well described by reference 5 (above) and references therein. The optimal control method starts with an initial RF sequence. Very briefly, as applied to RF pulse optimization, optimal control attempts to make the forward path (in which the RF sequence takes the initial magnetization towards the desired state) as close as possible to the reverse path (in which the desired state is taken by the reverse of the RF sequence towards the initial magnetization). The vector Lagrange multiplier at the final time point is identified as the desired final magnetization to be run in the reverse direction towards the initial magnetization. At each time point, the difference between the forward and reverse paths provides the direction of change needed to be applied to the RF sequence to bring the paths closer together (i.e., to be optimized). As discussed in ref 5, this provides a very efficient optimization algorithm.

Optimal control theory applied to RF pulse design has been described a number of times in the literature. In general, the OC algorithm follows that outlined in ref 5. There are, however, some differences. In the nomenclature of reference 5, **M**(t) is the magnetization at time point t, where t is incremented from 0 to the pulse length tp by δt, ωe is the B1 field, and λ represents the vector Lagrange multiplier.

The outline of the optimal control routines, following the nomenclature in ref 5, is:

i. Choose an initial RF sequence $\omega e^{(k)}$.

ii. Evolve **M** forward in time from the initial state

iii. Evolve λ backwards in time, starting from the target state **F**

iv. Correct $\omega e^{(k)}(t)$ at all time points by $\omega e^{(k+1)}(t) = \omega e^{(k)}(t) + \varepsilon [\mathbf{M}(t) \times \lambda(t)]$

v. If the difference between **M**(tp) and **F** is increased, reduce ε and repeat the calculation of **M**(tp)

vi. For any $\omega > \omega_{max}$, set $\omega = \omega_{max}$

vii. Repeat steps (ii) through (vi) until a desired convergence is reached.

For step (v), a tolerance can be set to allow a slight increase in the difference between **M**(tp) and F, as we found that the residual (absolute linear difference between the states **M**(tp) and **F**) was often increased slightly early in the optimization. In addition, the algorithm limited the number of times ε could be sequentially reduced before continuing to another iteration.

In addition, as the cross product **M**(t) x λ(t) depended equally on the time points t and t+δt, with the exception of the end points ω(0) and ω(tp), ω(t) was corrected by:

$$\varepsilon/2 [\mathbf{M}(t) \times \lambda(t) + \mathbf{M}(t+\delta t) \times \lambda((t+\delta t))]$$

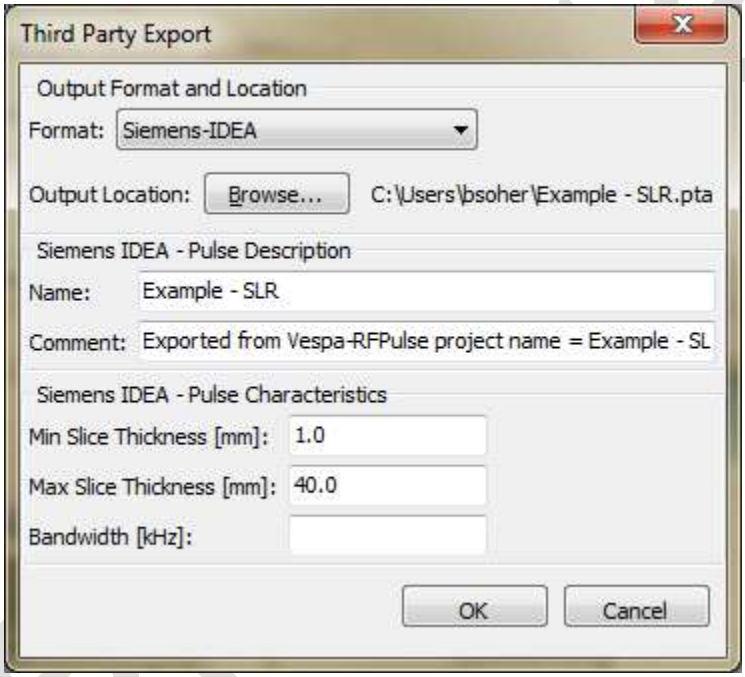
The algorithm allows three options for setting the value of ε. The user could set a fixed value for ε; the user could set the program to choose a near-optimal value for ε every hundred iterations; the user could allow the program to adjust ε every hundred iterations, where the program increased ε if the residual never increased, or decreased ε if the residual increased more than a few times in the previous hundred iterations. As the optimal value for ε appeared to change significantly over even a few iterations in the Matson paper, the latter option, which could be viewed as averaging over a number of iterations, was used for the most part for the optimizations in the Matson paper.

Appendix C. Third Party Export

This section describes the Third Party Export dialog launched by selecting the **PulseProject →ThirdPartyExport**… menu item. This dialog converts RFPulse waveform results into various third party formats and exports the converted results to a file. At the moment, RFPulse supports three formats:

1) Siemens IDEA single RF pulse ASCII format.
2) Siemens Vision single RF pulse ASCII format.
3) Generic ASCII magnitude/phase representation.
4) Annotated ASCII magnitude/phase representation.

The same dialog is used to output all formats; an example is shown below:



C.1 General Functionality

The third party export dialog acts on the pulse project that is active when the dialog is launched. The GUI reformats itself depending on the selection in the **Format** pull down list. All formats save ONLY the waveform result from the last transformation tab in the project.

Specify the file to which you wish to export results by clicking the Browse… button. This selection is used slightly differently in each format. The Output Location typically defaults to the last location where something was saved. The differences in default file names will be discussed specifically for each format in the sections below.

Parameters specific to the format are listed in the next sections (if any) and are also described more fully below.

C.2 Siemens-IDEA Format

The Siemens Pulsetool utility can import RF pulse or gradient waveforms that are saved in the ASCII file format described in the Siemens IDEA manual. Typically, only one RF pulse or gradient waveform is stored in each file. At the moment, only RF pulses are exportable from the RFPulse application. The first figure in this section shows the Third Party Export dialog configured for the Siemens-IDEA format output.

C.2.1 Format Specific GUI Fields

Siemens-IDEA format ASCII files have some user-set and automatically-calculated parameter header lines at the top of the file. These lines are followed by magnitude-phase value pairs, one pair per line, for each time step in the RF waveform. We refer you to the Siemens IDEA manual for more specific details for each automatically-calculated header parameter. User-set header parameters that are required include:

Name (text) This value defaults to the pulse project name, but can not contain spaces or periods, so these are replaced by underscores automatically. This value is also used as the default filename for the output location.

Comment (text) The default comment contains the pulse project name, UUID, and pulse duration as reminders to you once it is imported into IDEA. This also allows the "Name" header variable and/or filename to be changed without losing provenance regarding the pulse project origin of this result. This comment is contained on only one line in the output file, so don't hit return in entering your comment.

Min Slice Thickness (float, [mm]) See the Siemens IDEA manual for additional description of this parameter. Vespa-RFPulse defaults this value to 1.0.

Max Slice Thickness (float, [mm]) See the Siemens IDEA manual for additional description of this parameter. Vespa-RFPulse defaults this value to 40.0.

Bandwidth (float, [kHz]) User should type in this value based on their visual inspection of their pulse and it will be provided to the header. This value is used in the calculation of the REFGRAD value.

C.2.2 Example Siemens-IDEA Output File

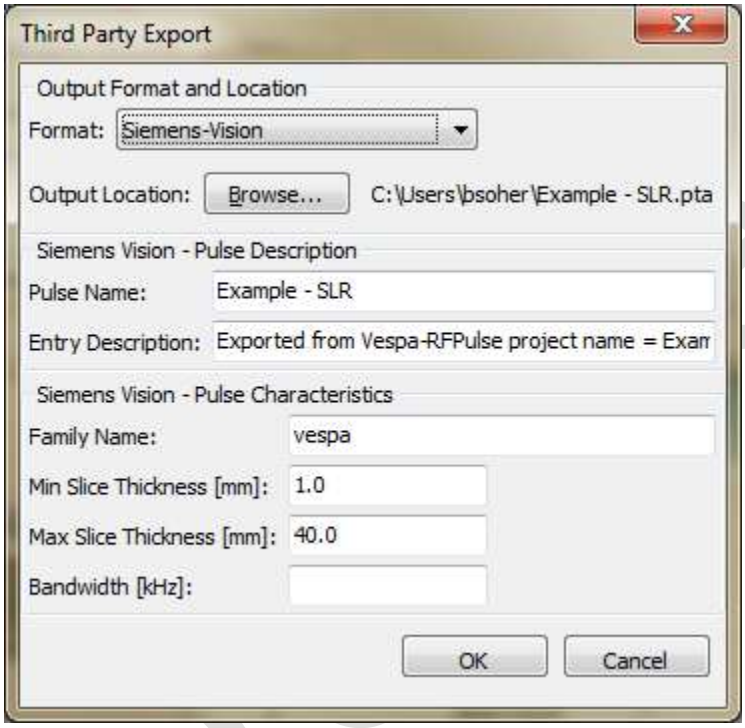
Here is a short example of the data output to a Siemens-IDEA format output file. Note that the COMMENT parameter is typically all on a single line but formatted here for easier reading.

```
PULSENAME:      Simple_64pt_SLR
COMMENT:        Exported from Vespa-RFPulse project name = Simple 64pt SLR, UUID = a1cec249-446c-
                41e2-a2fd-578a3894ad7d, and pulse duration = 8.0 [ms]
REFGRAD:        0.000003670
MINSLICE:       1.000000000
MAXSLICE:       40.000000000
AMPINT: 8.436948615
POWERINT:       7.329981263
ABSINT: 12.869012534


0.043329947     3.141592654     ; (0)
0.026630458     3.141592654     ; (1)
0.029410526     3.141592654     ; (2)
0.027636394     3.141592654     ; (3)
0.020032857     3.141592654     ; (4)
0.006048280     3.141592654     ; (5)
0.014027609     0.000000000     ; (6)
0.038877678     0.000000000     ; (7)
0.066164780     0.000000000     ; (8)
...
```

C.3 Siemens-Vision Format

This is a format that is compatible with older Siemens MR scanners. The code for this format was derived from Jerry Matson's MatPulse program rather than directly from Siemens documentation. Many of the fields are the same (and have similar meaning) as those for Siemens-IDEA format.



C.3.1 Format Specific GUI Fields

Siemens-Vision format ASCII files have some user-set and automatically-calculated parameter header lines at the top of the file. These lines are followed by magnitude-phase value pairs, one pair per line, for each time step in the RF waveform. We refer you to the Siemens Vision documentation for more specific details for each automatically-calculated header parameter. User-set header parameters that are required include:

Name (text) This value defaults to the pulse project name, but can not contain spaces or periods, so these are replaced by underscores automatically. This value is also used as the default filename for the output location.

Comment (text) The default comment contains the pulse project name, uuid, and pulse duration as reminders to you once it is imported into IDEA. This also allows the "Name" header variable and/or filename to be changed without losing provenance regarding the pulse project origin of this result. This comment is contained on only one line in the output file, so don't hit return in entering your comment.

Family Name (text) This value is used to group related pulses in the Siemens-Vision pulse libraries. It may not contain spaces or periods.

Min Slice Thickness (float, [mm]) See Siemens Vision documentation for additional description of this parameter. Vespa-RFPulse defaults this value to 1.0.

Max Slice Thickness (float, [mm]) See Siemens Vision documentation for additional description of this parameter. Vespa-RFPulse defaults this value to 40.0.

Bandwidth (float, [kHz]) User should type in this value based on their visual inspection of their pulse and it will be provided to the header. This value is used in the calculation of the Reference Grad value.

C.3.2 Example Siemens-Vision Output File

Here is a short example of the data output to a Siemens-Vision format output file. Note that the Entry_Description parameter is typically all on a single line but formatted here for easier reading. Also the waveform results are entered sequentially, one value per line, all magnitude values first followed by the phase values.

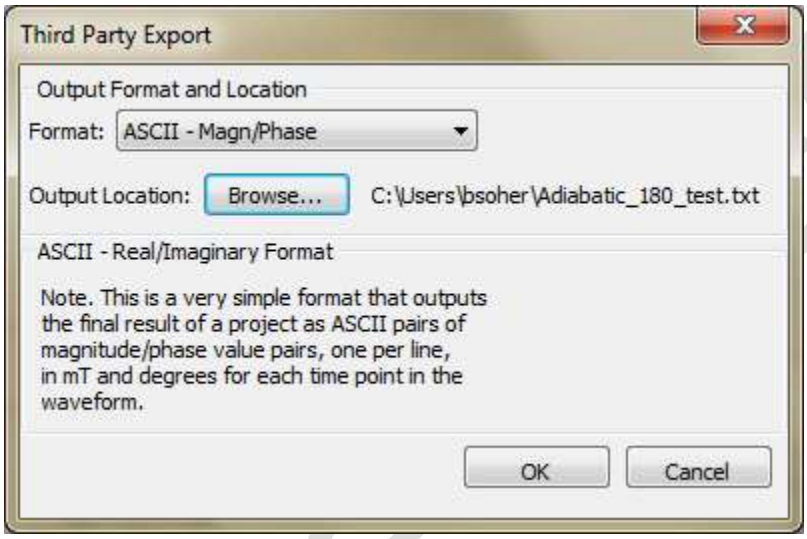
```
Begin_Entry:   VESPA_RFPulse_Generated_Pulse
Entry_Type:    6
Pulse_Name:    Simple_64pt_SLR
Entry_Description:     Exported from Vespa-RFPulse project name=Simple 64pt SLR,
                        UUID=a1cec249-446c-41e2-a2fd-578a3894ad7d, and pulse duration = 8.0 [ms]
Num_Points:    64
Family_Name:   vespa
Slice_Thick_Min:       1.000000000
Slice_Thick_Max:       40.000000000
Reference_Grad:        0.000003670
Power_Integral:        7.329981263
Ampl_Integral: 8.436948615
Envelope_Mode: 1
Entry_Values:
0.043329947
0.026630458
0.029410526
0.027636394
0.020032857
0.006048280
0.014027609
0.038877678
0.066164780
0.092508011
0.114093901
0.127025923
0.127717372
0.113967653
0.085044517
0.042370597
0.010457088
0.067715754
…

3.141592654
3.141592654
3.141592654
3.141592654
3.141592654
3.141592654
0.000000000
0.000000000
0.000000000
0.000000000
0.000000000
0.000000000
0.000000000
0.000000000
0.000000000
0.000000000
0.000000000
3.141592654
3.141592654
…
```

C.4 ASCII – Magn/Phase Format

This is a very simple ASCII output format that basically writes the waveform to a text file as a pair of magnitude and phase terms (expressed as floating point numbers). One time point is written per line. The magnitude term is saved in microTeslas, and phase is expressed in radians. There is no header information in this file format so you are encouraged to name each file descriptively as a reminder as to what is in each file.

Note. This Export format is compatible with the RFPulse Import create transformation.



C.4.1 Format Specific GUI Fields

There are no format specific fields for this format.

C.4.2 Example ASCII – Magn/Phase Output File

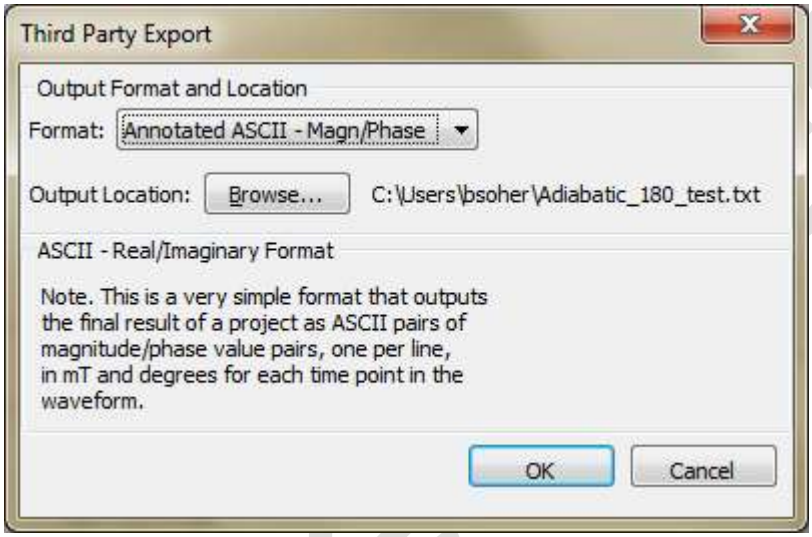
Here is a short example of the data output to an ASCII – Magn/phase format output file. Values are separated by a space.

```
0.043329947 3.141592654
0.026630458 3.141592654
0.029410526 3.141592654
0.027636394 3.141592654
0.020032857 3.141592654
0.006048280 3.141592654
0.014027609 0.000000000
0.038877678 0.000000000
0.066164780 0.000000000
0.092508011 0.000000000
0.114093901 0.000000000
0.127025923 0.000000000
0.127717372 0.000000000
0.113967653 0.000000000
0.085044517 0.000000000
0.042370597 0.000000000
0.010457088 3.141592654
0.067715754 3.141592654
…
```

C.5 Annotated ASCII – Magn/Phase Format

This is a very simple ASCII output format that basically writes the waveform to a text file as a pair of magnitude and phase terms (expressed as floating point numbers). One time point is written per line. The magnitude term is saved in microTeslas and phase is expressed in radians. There is a brief text header in this format on lines starting with '#' symbols. This header information gives a brief description of the PulseProject from which this waveform was exported.

Note. This Export format is compatible with the RFPulse Import create transformation.



C.5.1 Format Specific GUI Fields

There are no format specific fields for this format.

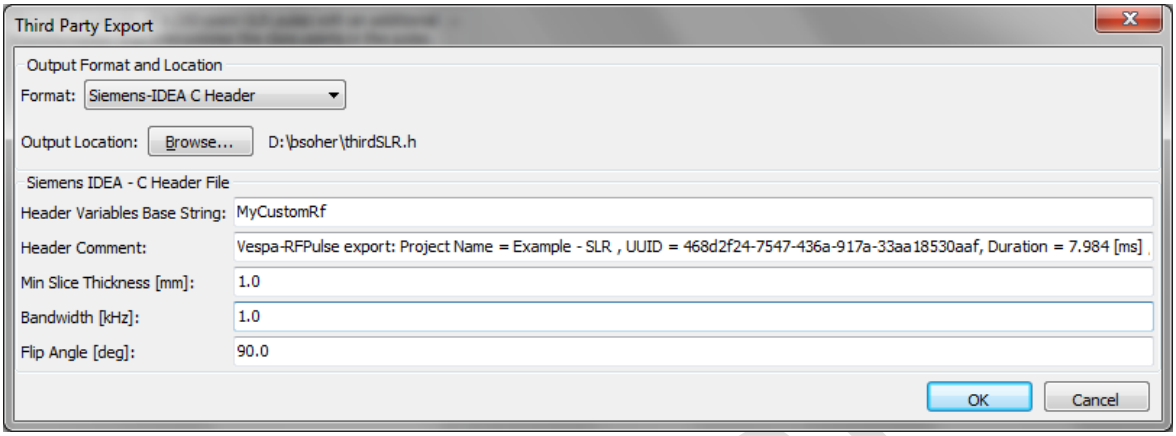
C.5.2 Example ASCII – Magn/Phase Output File

Here is a short example of the data output to an Annotated ASCII – Magn/phase format output file. Values are separated by a space.

```
# Exported from Vespa-RFPulse
# For more information, visit http://scion.duhs.duke.edu/vespa/
# Export Timestamp: 2011-10-25T15:37:16
# Pulse Name: Example - SLR
# Project UUID: 468d2f24-7547-436a-917a-33aa18530aaf
# Pulse Total Duration: 7.984 [ms]
# ---------------------------------------
0.000793201 180.000000000
0.000469102 180.000000000
0.000145003 180.000000000
0.000150651 180.000000000
0.000156299 180.000000000
0.000161669 180.000000000
0.000167039 180.000000000
0.000171922 180.000000000
0.000176805 180.000000000
0.000181311 180.000000000
0.000185816 180.000000000
…
```

C.6 Siemens-IDEA C Header Format

This creates a C language header (*.h) file that contains a representation of the pulse sequence envelope for use in importing it into a Siemens pulse sequence.



C.6.1 Format Specific GUI Fields

Fill in the Bandwidth and Flip Angle fields by hand, so make a note before launching dialog. These are needed to calculate some of the variables reported in the header file.

C.6.2 Example Siemens-IDEA C Header format

Here is a short example of the output from this format. Note that there is a lot more example code appended to the bottom of the actual file to help you remember how to implement it in an actual Siemens pulse sequence file as an Arbitrary RF pulse.

```
// =------------------=
// Arbitrary pulse file
// =------------------=
//
// General statistics (as exported):
//    Duration:     7.984 ms
//    Max B1:       0.006 kHz
//    Num. Steps:   499
//    SAR*:         0.000
//    Ref. Grad:    3.663 mT/m
//    Ref. Grad:    0.156 kHz/mm for 1H
//    Flip Angle:   90.0 deg.
// * - Relative to a 1 ms pi-pulse.
//
// User comment: Vespa-RFPulse export: Project Name = Example - SLR ,
//               UUID = 468d2f24-7547-436a-917a-33aa18530aaf,
//               Duration = 7.984 [ms] , Bandwidth = 1.0 [kHz],
//               and Tip Angle = 90.0 [deg]
//
// Hint: do not forget to include these libraries and definitions:
//    #include "MrServers\MrMeasSrv\SeqIF\libRT\libRT.h"
//    #include "MrServers\MrMeasSrv\SeqFW\libSSL\SSL_local.h"
//    static sSample MyCustomRfPulseArray[499];

float MyCustomRfRefGrad = 3.6631;
float MyCustomRfMinSlice = 1.0;
float MyCustomRfMaxSlice = 200.0;
float MyCustomRfAmpInt = 58.334696817; // calculated for 1H
float MyCustomRfPowerInt = 56.272891895;
float MyCustomRfAbsInt = 99.408069681;
```

```
MyCustomRfPulseArray[0].flAbs = float(0.12606);   MyCustomRfPulseArray[0].flPha = float(3.14159);
MyCustomRfPulseArray[1].flAbs = float(0.07456);   MyCustomRfPulseArray[1].flPha = float(3.14159);
MyCustomRfPulseArray[2].flAbs = float(0.02305);   MyCustomRfPulseArray[2].flPha = float(3.14159)
…

MyCustomRfPulseArray[497].flAbs=float(0.07456); MyCustomRfPulseArray[497].flPha=float(3.14159);
MyCustomRfPulseArray[498].flAbs=float(0.12606); MyCustomRfPulseArray[498].flPha=float(3.14159);

/*

// The following code is an example of how to make use of this header file
// in a pulse sequence. Instructions on where to put the code have single line
// comment characters in front of them. Actual code lines are not commented out.
// Copy these sections into your file as instructed to activate them.

// 1. Put header file in the same directory as the pulse sequence.
//
// 2. In the "global" part of the sequence (i.e. before fSEQInit, where the
//     pulses & gradients are defined), add:
//
// <------------------------------------ BEGIN --------------------------------->

static int MyCustomRfNumSamples = 499;
static sSample MyCustomRfPulseArray[499];

static sRF_PULSE_ARB   MyCustomRfPulse("MyCustomRfPulse");
static sFREQ_PHASE     MyCustomRfPulseSet( "MyCustomRfPulseSet" );
static sFREQ_PHASE     MyCustomRfPulseNeg( "MyCustomRfPulseNeg" );

// <------------------------------------ END ----------------------------------->
//
// 3. Somewhere in the fSEQPrep function, around where you prepare the pulses, add:
// (This may not be "optimal" or the most elegant way. e.g., you may be able to
// define SLR90NumSamples already in SLR90.h. I was not sure about that one because
// I wanted my pulse arrays to be global, i.e. static, and did not want to deal with
// dynamic memory allocation which I am never sure about)
//
// <------------------------------------ BEGIN --------------------------------->

#include "thirdSLR.h"

// Prepare Excitation Pulse
MyCustomRfPulse.setTypeUndefined(); // Whatever. Never understood what this is good for really.
MyCustomRfPulse.setSamples(MyCustomRfNumSamples);
MyCustomRfPulse.setDuration( 7984 );   // in us. Put whatever you want here
MyCustomRfPulse.setFlipAngle( 90.0 ); // In degrees. Put whatever you want here
MyCustomRfPulse.setInitialPhase( 0.0 );
MyCustomRfPulse.setThickness( pMrProt->spectroscopy().voI().thickness() ); // For example, if the
pulse works along the "slice" direction

if (!MyCustomRfPulse.prepArbitrary(pMrProt,pSeqExpo, MyCustomRfPulseArray, MyCustomRfAmpInt))
{
    cout << "ERROR: "<< MyCustomRfPulse.getNLSStatus() << endl;
    return (false);
};

MyCustomRfPulseGrad                                                                    =
(10.0/MyCustomRfPulse.getThickness())*MyCustomRfRefGrad*(5120.0/MyCustomRfPulse.getDuration());
// mT/m
lFrequency = specMiddleFreq;
lFrequency += (long)( 0.5 + MyCustomRfPulseGrad * larmorconst * VOI.getSliceShift() );

MyCustomRfPulseSet.setFrequency( lFrequency );
MyCustomRfPulseNeg.setFrequency( 0L );

dPhaseExcite = - lFrequency * (360.0/1e6) * MyCustomRfPulse.getDuration() * 0.5;
MyCustomRfPulseSet.setPhase( dPhaseExcite );
MyCustomRfPulseNeg.setPhase( dPhaseExcite );

// <------------------------------------ END ----------------------------------->

*/
```

Appendix D. Object State in Applications

This section describes important concepts in Vespa that have significant practical issues in how you make use of all the applications in the package. We have defined a number of terms that describe certain conditions of the prior information and results that are stored within the Vespa database. These terms include: ‘private’, ‘public’, ‘in use’ and ‘frozen’. Our definition of these terms, and their practical implementation within Vespa applications, go a long way towards providing accurate workflow provenance of how experiments or pulse projects were created. They also help to keep users from deleting or changing important information. This section will help you understand what these terms mean and how to use them effectively within Vespa.

D.1 Background and Design Philosophy

Our overall goal when designing Vespa has been to try to help you organize your data and workflow. Each application has a number of modules that can be combined in different ways as part of your investigations. For example, in Simulation an experiment contains one pulse sequence and one or more metabolites. There are a variety of pulse sequences and metabolites and these can be combined in many ways to create experiments. The design philosophy behind Vespa has been to enable great flexibility in each application while still providing a complete description of how each usage ended up with the results it did. This historical record your actions is called the ‘provenance’.

In the database, each pulse sequence, metabolite and pulse project is stored just once, but may be referred to by many other objects. For instance, the three sample experiments installed with Vespa Simulation all refer to the metabolite creatine. A change to the definition of creatine would be a change in all of the experiments that refer to it. This would damage the provenance that Vespa wants to protect.

D.2 State Definitions and Usage

To avoid damaging provenance Vespa classifies items into states called ‘private’, ‘public’, ‘in use’ and ‘frozen’. These states determine which database items can be changed/deleted and which cannot. There are also very simple steps for creating editable copies of uneditable items. Definitions and advice for each state is given below.

D.2.1 Private and Public

Objects Affected: experiments, metabolites, pulse sequences, pulse projects

All of these objects start life private. That means they're only in your database; no one else has seen them.

Once exported, objects become public. That means that their definition has been shared with the world. Public objects are frozen (see below). Furthermore, the objects to which they refer (directly and indirectly) also become public (and frozen). For instance, if an experiment refers to a pulse sequence that refers to a pulse project, all three of those objects become public when the experiment is exported.

Once a private object has become public, it can never become private again. Cloning, however, will create a new, private object with exactly the same properties (but a different UUID).

D.2.2 In Use

Objects Affected: metabolites, pulse sequences, pulse projects

When you select a metabolite or pulse sequence for use in an experiment, that experiment refers to the object for as long as the experiment exists in your database. Metabolites and pulse sequences that are referred to by an experiment are in use by that experiment.

Objects that are in use may not be deleted and are frozen (see below).

There's no limit to the number of references an object may have.

Once all of the experiments referring to an object are deleted, the object is no longer in use.

D.2.3 Frozen

Objects Affected: experiments, metabolites, pulse sequences, pulse projects

Frozen objects are mostly un-editable -- only the name and comments can be changed. Objects are frozen for one of two reasons.

In use objects (metabolites, pulse sequences and pulse projects) are frozen because they're referred to by an experiment, and changing the underlying objects that the experiment uses would corrupt the experiment.

Public objects are frozen because once you've shared an object with others (or you've imported an object that they've shared with you), you need to be able to trust that you're talking about exactly the same object.

Here's a table summarizing when an object is frozen.

Private?	In Use?	Frozen?
Yes	No	No
Yes	Yes	Yes
No (public)	No	Yes
No (public)	Yes	Yes

Note that no objects can refer to experiments, so experiments can never be 'in use'.

Note that frozen only refers to whether or not the fundamental attributes of the object can be edited. It doesn't affect whether or not it can be deleted.

Appendix E. RFPulse Deprecation

The Pulse application was created to replace RFPulse in Vespa. Initially, we introduced it as a standalone application for people to try out and report bugs. As of release 0.8.6, March 2016, it officially became the 'RF pulse application' in Vespa. When this happened, we transferred all results from RFPulse to Pulse in order to maintain backwards compatibility. We also updated the database to tell Vespa-Simulation to get RF pulse results used in PulseSequence objects from the Pulse application not RFPulse. We explain here, both how and why it was done.

When version 0.8.6 was released, part of the upgrade code converted and transferred all RFPulse PulseProject objects into Pulse PulseDesign objects. Each PulseDesign object has an "Import from RFPulse" transform kernel that contains the waveform, gradient and time axis information from the final RFPulse transformation object. This information is encoded in "xdr zlib base64" formats similarly to what we push numpy arrays into in our XML outputs. These byte strings are converted upon the Run button being triggered into rf waveform, gradient and time axis array results in the PulseDesign object.

By formulating RFPulse results into Pulse in this manner, we lose the information about how the pulse was created, but we are still able to use and store these pulses in Vespa-Simulation PulseSequence objects. This was deemed sufficient for backward compatibility in this case.

Although RFPulse is deprecated, it will remain part of the Vespa package for the foreseeable future. It will remain as a 'standalone' application in that its results cannot be used upstream in Vespa-Simulation as part of a PulseSequence. The main reason to leave it at all is to allow users to revisit previously created PulseProject objects to see how pulses were created.