Vespa – Pulse
User Manual and Reference

Version 0.10.0

Release date: October 7th, 2019

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 Analysis, Pulses and Analysis". Applications in the Vespa package (and the original tools from which they were migrated) include:

- Vespa-Simulation – software for spectral simulation (GAVA/Gamma)
- Vespa-Pulse – software for RF pulse design (MatPulse)
- Vespa-Analysis – spectral data processing and analysis (MIDAS/IDL-Vespa)
- Vespa-Priorset – used to create 'fake' MRS data sets from Simulation results

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.

Installation

Vespa is a Python package. It requires that a Python environment be installed with certain required additional modules (all easily obtained) and then Vespa can be automatically installed from the PyPI web site.

Full instructions for installation can be accessed online at:

https://scion.duhs.duke.edu/vespa/project/wiki/HowToInstallVespa

Vespa is an actively developed project that has frequent releases. Once you have it installed, it is easy to upgrade as described here:

https://scion.duhs.duke.edu/vespa/project/wiki/Upgrading

Other Online Resources

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

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

Introduction to Vespa-Pulse

Vespa-Pulse is a platform that allows users to create, compare and analyze radio frequency (RF) pulses for use in magnetic resonance (MR) applications. It is program written in the Python programming language for easy prototyping and cross platform compatibility. Pulse has GUI interface to allow users to visualize the RF waveform at all steps in the creation/modification process. Users can contribute their own create/modify algorithms using the integrated design/test dialog.

What can Pulse do?

1) Create new RF pulse designs from a list of user defined algorithms
2) Store pulse designs and their design parameters into a database
3) Re-load previous pulse designs
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 designs
6) Output results in text or graphical format, including MR manufacturer platform formats
7) Export/Import Vespa pulse designs to/from other users
8) Be an open source test bed for your own algorithms and transformations.

What is an RF pulse Design? A ‘pulse Design’ consists of one or more transform steps. Each pulse Design has a single Create transform step. You can further refine the pulse by adding additional Modify transform steps. Each step of the design process (creation plus other transforms) 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 transforms.

Designs are displayed as a tab in a Notebook. Each Design tab has a series of sub-tabs that display each transform step applied to the Design. You can open multiple pulse Design tabs in the Notebook and compare them side-by-side. Any pulse Design tab can be copied into a new pulse Design 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.

A pulse Design result stores a complex RF waveform and a gradient vector in 1, 2 or 3 dimensions, which will be explained in more detail in later sections. The effects of various gradients and/or frequency offsets can be explored interactively using various Bloch transform and display capabilities in each tab of the Design Notebook.

The following chapters run through the operation of the Vespa-Pulse 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-Pulse/ % 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/

Using Pulse – A User Manual

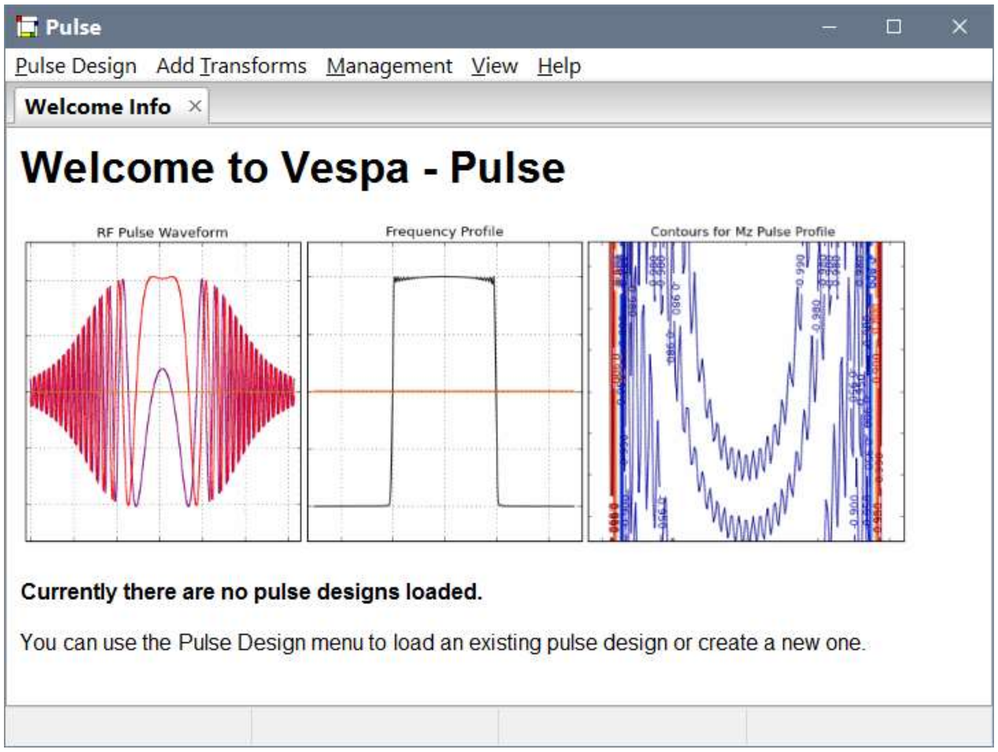
This section assumes Vespa-Pulse has been downloaded and installed. See the Vespa Installation guide on the Vespa main project wiki for details on how to install the software and package dependencies. http://scion.duhs.duke.edu/vespa.

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

1. How to launch Vespa-Pulse

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

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



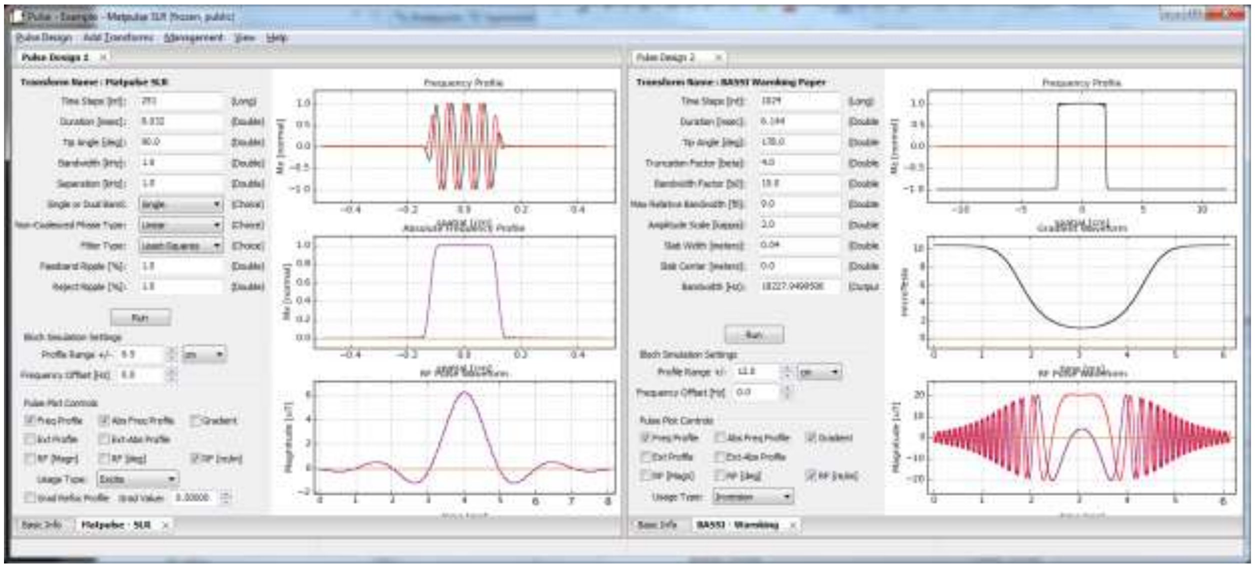
Welcome to Vespa - Pulse

Currently there are no pulse designs loaded.

You can use the Pulse Design menu to load an existing pulse design or create a new one.

Use the **PulseDesign** menu to open an existing pulse Design or to create a new tab for designing an RF pulse.

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


Pulse - Example - Matpulse SLR (frozen, public)

Pulse Design | Add Transforms | Management | View | Help

Pulse Design 1

Transform Name : Matpulse SLR

Time Steps [int]:	251	[Long]
Duration [msec]:	8.032	[Double]
Tip Angle [deg]:	90.0	[Double]
Bandwidth [kHz]:	1.6	[Double]
Separation [kHz]:	1.0	[Double]
Single or Dual Band:	Single	[Choice]
Non-Coalesced Phase Type:	Linear	[Choice]
Filter Type:	Least Squares	[Choice]
Passband Ripple [%]:	1.0	[Double]
Reject Ripple [%]:	1.0	[Double]

Run

Bloch Simulation Settings

Profile Range +/- 0.5 cm

Frequency Offset [Hz] 0.0

Pulse Plot Controls

[✓] Freq Profile [✓] Abs Freq Profile [ ] Gradient

[ ] Ext Profile [ ] Ext-Abs Profile

[ ] RF [Magn] [ ] RF [deg] [✓] RF [re/im]

Usage Type: Excite

[ ] Grad Refoc Profile Grad Value: 0.00000

Basic Info | Matpulse - SLR

Pulse Design 2

Transform Name : BASSI Warnking Paper

Time Steps [int]:	1024	[Long]
Duration [msec]:	6.144	[Double]
Tip Angle [deg]:	170.0	[Double]
Truncation Factor [beta]:	4.0	[Double]
Bandwidth Factor [kk]:	10.0	[Double]
Max Relative Bandwidth [%]:	9.0	[Double]
Amplitude Scale [kappa]:	2.0	[Double]
Slab Width [meters]:	0.04	[Double]
Slab Center [meters]:	0.0	[Double]
Bandwidth [Hz]:	18227.9499506	[Output]

Run

Bloch Simulation Settings

Profile Range +/- 12.0 cm

Frequency Offset [Hz] 0.0

Pulse Plot Controls

[✓] Freq Profile [ ] Abs Freq Profile [✓] Gradient

[ ] Ext Profile [ ] Ext-Abs Profile

[ ] RF [Magn] [ ] RF [deg] [✓] RF [re/im]

Usage Type: Inversion

Basic Info | BASSI - Warnking

2. Quick Guide – The Nuts and Bolts of Pulse

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 Pulse 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 design, 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 Pulse.

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

The easiest interaction requires just a few basic steps.

- From the **PulseDesign** menu, select **New** to create a new pulse Design tab.
- In the “Basic Info” tab, fill in the design name and investigator fields, select a nuclei other than 1H if you want.
- Select **Add Transforms→Create→Matpulse-SLR multinuc** and a new “Matpulse 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 **PulseDesign →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 Modify type transforms from the **Add Transforms** menu. Each transform appears on a new tab and allows you to further refine your pulse. (The motivations for manipulating RF pulses is briefly discussed in the Case Studies appendix).

There are two types of transforms, “create” transforms and “modify” transforms. As shown above, pulse designs start with a create transform. Create transforms are the algorithms by which an initial RF pulse waveform is designed (e.g. Shinnar-LeRoux, hyperbolic secant, etc.). Each pulse design has just one create transform. A create transform is selected from the **Add Transforms→Create** sub-menu. Modify transforms are listed below the **Create** menu entry. Modify transforms operate on the results of the previous transform (i.e. the transform in the tab to the left).

Notes

1. A second typical workflow to the one above might be to load a saved pulse design from the database, add, edit or delete modify transformations, and then save the changed results back to the database.
2. A third typical workflow might be to load a saved pulse design from the database, and copy that into a new pulse design by using the **PulseDesign→Copy Tab to New** menu

item. Then modify the RF pulse in the copied tab and compare the results plots to the original pulse design results.

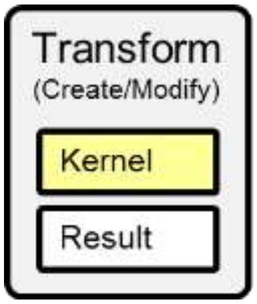
3. Changes can be made to parameter settings in any transform tab in a pulse design. When you click the Run button, Pulse calculates the effects of these changes in all transform tabs “downstream” from the transform tab changed. **Note** – as of this writing, there is no “undo” functionality in the transform tabs in the application. Although, for existing pulse designs that have been loaded into a tab, changes to that design are not saved to the database until you select the **PulseDesign→Save** menu manually.

3. Transforms and Designs – Basics of Pulse Functionality

The process of creating an RF pulse in Pulse is made up of a series of steps called ‘transforms’. There are a group of standard transforms that come installed with Pulse. However, users can create their own transforms to expand and personalize the Pulse application using the Transform Kernel Editor. All standard transforms were created using the Transform Kernel Editor. All transforms (standard or user created) are stored in the Vespa data base for use in Vespa-Pulse.

Below, we will describe how transforms are created, tested, stored and subsequently used in the main Pulse application.

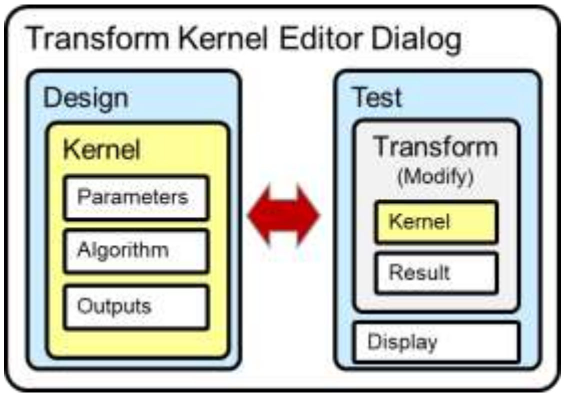
What is a Transform? A transform is a Python object that contains two sub-objects, a Transform Kernel and a Result. This is shown in the figure to the right.



- The Transform Kernel object contains the scientific code that is executed during the transform step. It also contains the input parameters and any output values separate from the RF waveform or gradient being designed.
- The Result object contains the values for the complex RF waveform and its gradient (if any) after the Kernel code has been applied.
- There are two types of Transforms, Create and Modify. Create Transforms contain a Create Transform Kernel and are used initially to ‘create’ an RF waveform result. They do not require that the previous Transform (if any) have an RF waveform in its Result.
- Modify Transforms contain a Modify Transform Kernel. These are used to ‘modify’ the previous transform’s Result to some new configuration. These require that the previous Transform have an RF waveform and gradient in its Result.

How do we create a Transform? Well, actually the only part of the Transform that we need to create and store for later use is the Transform Kernel object. That is the part of the Transform that contains the scientific code, and parameters it needs to create or modify an RF waveform. *So the **Kernel** is what we will refer to from now on*.

- Use the **Manage Transform Kernels** menu item under the top level **Management** menu to launch a dialog that lists all transforms in the Vespa database. You have the option to Edit, View, Clone or Delete existing transforms.

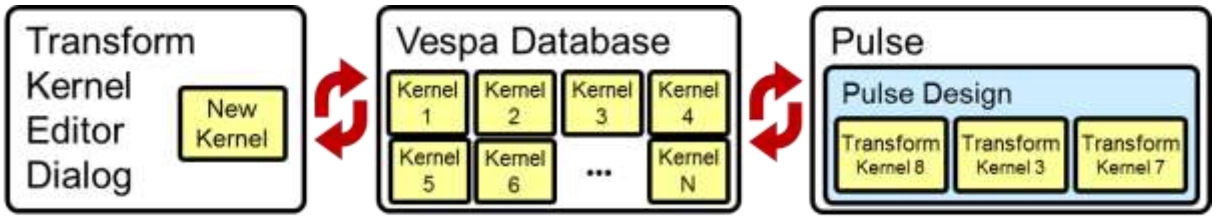


- The **New…** button launches the Transform Kernel Editor dialog. In this dialog you can design and test new transform code. When you are finished you can hit OK and your new (or updated) kernel is saved to the Vespa database for use in the main Pulse application. This functionality is shown graphically in the figure to the right.
- The full functionality of the Transform

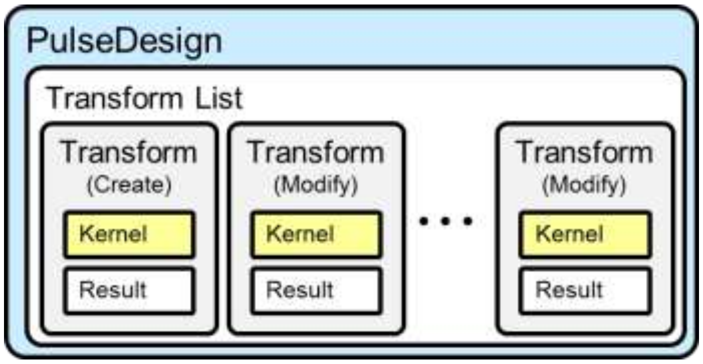
Kernel Editor is described in more detail later, but for now, suffice to say that the Editor will allow you to fully test and display results from your pulse design code. You can save a draft and return to make more changes later. And when you are ready, you can start using it in the main Pulse application.

- We expect users to share data via Pulse's export and import functions. For this reason, each Transform Kernel object has a universally unique id (UUID) which is used to maintain a unique set of Kernels within a Vespa database.

And how does all this work together? As shown in the figure below, the Transform Kernel Editor, Vespa database and main Pulse program work together as a system to allow you to extend and personalize the application. You can also export Transform Kernels to share with others, or import functionality that others send to you.



And what is this Pulse Design thing? Pulse Designs are the organizational object of the Pulse application. A Pulse Design's *raison d'etre* is to help you design an RF pulse through a series of transform steps. These transforms first create and subsequently modify an RF pulse to achieve the desired results. This set of transforms is the pulse design's *transform list*. See figure below.



The transform list is implemented as a linear group of transform objects. The first transform object is always Basic Info. This is a special transform object that occurs prior to any RF pulse waveform results being created. It allows the user to enter meta-data for the pulse design and sets up global parameters that can affect pulse calculation and display in subsequent steps. Specifically, it allows you to select the Machine Specs 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. It creates an initial RF pulse waveform result. Subsequent steps always contain a 'modify' transformation which uses the waveform result of the previous step as an input to be modified by the current transformation step. A Pulse Design 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 transform tab).

The transform design was selected to allow more modularity in RF pulse design and increased flexibility in the display of intermediate results. Each transform tab in the pulse design 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 transform tab to allow you to visually inspect the effects of their settings for each design step.

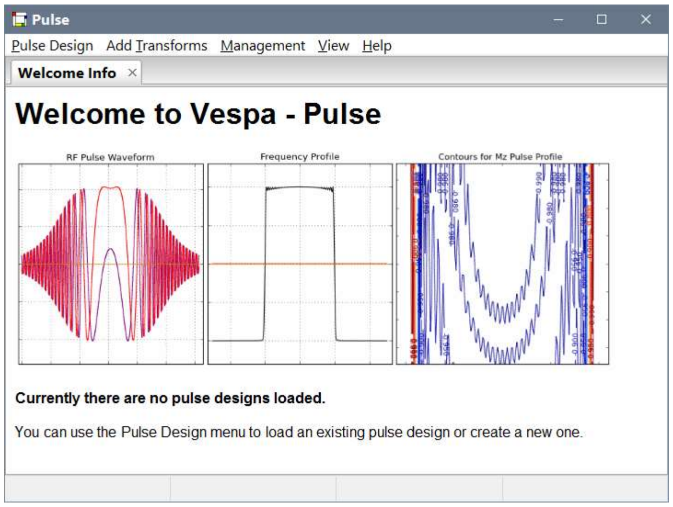
Transforms can be added or removed from the pulse design, typically by closing the associated tab in the GUI, with a few caveats. New transform steps can only be added to the end of the current project. Any modify transform can be closed/removed, however this triggers an automatic recalculation of the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

The take-home lesson from this section is that the Vespa-Pulse 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.

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

4. The Pulse Main Window

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



Currently there are no pulse designs loaded.

You can use the Pulse Design menu to load an existing pulse design or create a new one.

The pulse design window is initially populated with a welcome text tab, but no pulse designs are loaded. From the Pulse menu bar you can 1) load a previously run pulse design from the Vespa database into a tab, or 2) create a new pulse design set of transforms and run it. In either case a tab will appear at the top of the window for each pulse design that is loaded or created. The Management menu allows you to access pop-up dialogs to create, edit, view, delete and import/export new transform algorithms, pulse designs 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

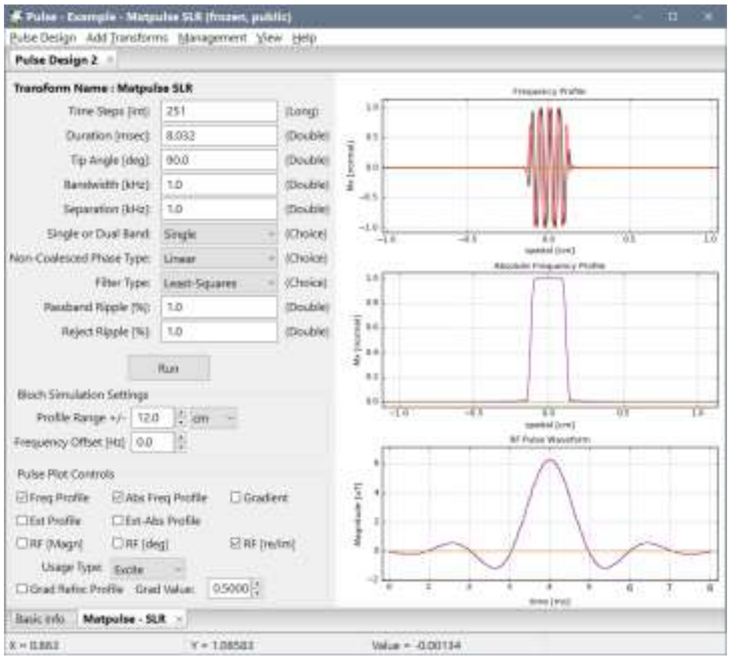
PulseDesign→New	Opens a new pulse design tab.
PulseDesign →Copy Tab to New	This will open a new pulse design tab and populate it with the same values that are listed in the current pulse design. This is a short cut for varying design parameters to get different results while retaining the ability to compare back to a previous results set without having to save them both to the data base.
PulseDesign →Open	Runs the pulse design Browser dialog, from which you can choose a pulse design to open from the database to open.
PulseDesign →Run	Runs all steps in the current pulse design tab. Works as if the user had manually hit the Run button in each transformation tab.
PulseDesign →Save	Saves the pulse design in the current tab to the database. **Note** - pulse design results are not automatically saved to data base after a Run button is hit.
PulseDesign →Save As	Saves the pulse design in the current tab to the database under a new name and unique ID.
PulseDesign→Close	Closes the current tab.
PulseDesign→Third Party Export	Opens a dialog that will export the final result of the active pulse design into a file suitable for import to various MR manufacturers' platforms.
PulseDesign →Exit	Closes current pulse design tab. Will prompt for save if necessary.

Add Transforms	This menu contains a list of all transform algorithms that exist in the Pulse database. Create transforms are under the Create sub-menu and Modify transforms are listed below the Create menu item. There are a number of standard transforms included in the Pulse installation. These are listed below. Other, user defined, transforms may also be listed in the Add Transform menu but are not described here. Note. As of version 0.9.4, Pulse allows users to select the gyromagnetic nuclei value used within the transform kernels and Bloch simulation. Older kernels may not make use of this 'system gamma' value. The new kernels (create and modify) that use this value are indicated by the term 'multinuc' in their label.
→Create→Matpulse SLR	Appends an SLR (create transformation) tab to current pulse design
→Create→Matpulse Hyperbolic-Secant	Appends a Hyperbolic-Secant (create transformation) tab to current pulse design
→Create→Matpulse Gaussian	Appends a Gaussian (create transformation) tab to current pulse design
→Create→Import from File	Appends an Import from File (create transformation) tab to current pulse design
→Contour Plot for B1 Immunity	Plots the pulse profile for a range +/- of B1 power settings set by the user. This will show how the pulse performs within an inhomogeneous coil.
→Interpolate and Rescale	Appends an Interpolate and Rescale (modify transformation) tab to current pulse design.
Management	
→Manage Pulse Designs	Launches the Manage Pulse Designs dialog. Allows you to view, clone, delete, import and export pulse designs.
→Manage Transform Kernels	Launches the Manage Transform Kernels dialog. Allows you to create, view, clone, or delete transform kernels.
→Manage Machine Specs Templates	Launches the Manage Machine Specs dialog. Allows you to create, edit, view, and delete templates for machine specs information.
View→<various>	Changes plot options in the selected transformation tab of the active pulse design 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→Pulse/Vespa Online Help	Online wiki for the Pulse application and Vespa project
Help→About	Giving credit where credit is due.

5. The Pulse Design Window

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


Pulse - Example - Matpulse SLR (frozen, public)
Pulse Design Add Transforms Management View Help
Pulse Design 2
Run
Basic Info
Matpulse - SLR

Transform Name : Matpulse SLR
Time Steps [int]: 251 (Long)
Duration [msec]: 8.032 (Double)
Tip Angle [deg]: 90.0 (Double)
Bandwidth [kHz]: 1.0 (Double)
Separation [kHz]: 1.0 (Double)
Single or Dual Band: Single (Choice)
Non-Coalesced Phase Type: Linear (Choice)
Filter Type: Least-Squares (Choice)
Passband Ripple [%]: 1.0 (Double)
Reject Ripple [%]: 1.0 (Double)

Bloch Simulation Settings
Profile Range +/- 12.0 cm
Frequency Offset [Hz] 0.0

Pulse Plot Controls
☑ Freq Profile ☑ Abs Freq Profile ☐ Gradient
☐ Ext Profile ☐ Ext-Abs Profile
☐ RF [Magn] ☐ RF [deg] ☑ RF [re/im]
Usage Type: Excite
☐ Grad Refoc Profile Grad Value: 0.5000

X = 0.863 Y = 1.08583 Value = -0.00134

6. The Pulse Design Tab

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

Each pulse design 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 “transform tabs”. The first tab is standard to all pulse designs and is called “Basic Info”. The second tab is always a “create” transform tab such as “SLR” or “Hyperbolic-Secant” (see Appendix B). Additional tabs, containing “modify” transform such as Interpolate-Rescale (see Appendix B) can be appended after the “create” transform tab. The pulse design tab can be resized by adjusting the Pulse application window and each transform tab has a vertical splitter bar through the middle that can be used to resize the right and left sides. Parameters for each transform 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 transform tab is added to the pulse design, there are no results to be displayed until the **Run** button has been clicked.

A new pulse design 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 design are <u>only</u> saved to the database when specifically requested by you. Transform tabs are updated to display results after each click on the Run button. If there are transform 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 designs start life as “private”. That means they're only in your database; no one else has seen/accessed them. Once you export a pulse design to share with another user, the pulse design is designated as “public”. That means the pulse design result has been shared with the world. Public objects are “frozen” and become mostly unchangeable. Once a private pulse design has become public, it can never become private again.

There are two ways that a public pulse design can be re-used. First, cloning a public pulse design (using the Manage Pulse Projects dialog which is described more fully in Section 6) will create a new, private pulse design 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 design window and modified. Second, any pulse design can be loaded into the pulse design window, whether it is public or private. From there, the **Pulse Project→Copy to New Tab** menu item will create a new pulse design 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 design tab. The resulting modifications only affect the settings in the active pulse design tab. More specifically, only the plot options in the selected transformation tab in the active pulse design 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:

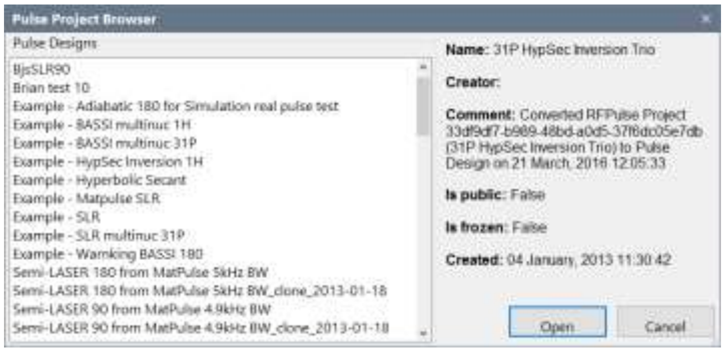
<u>On the Menu Bar</u>

View (this menu affects the plots in the currently active PulseDesign/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 Pulse to display in any color matplotlib will support.
→Output→<format>	writes the plot panel to file as either PNG, SVG, EPS or PDF format

6.1 Loading an existing Pulse Design

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



BjsSLR90
Brian test 10
Example - Adiabatic 180 for Simulation real pulse test
Example - BASSI multinuc 1H
Example - BASSI multinuc 31P
Example - HypSec Inversion 1H
Example - Hyperbolic Secant
Example - Matpulse SLR
Example - SLR
Example - SLR multinuc 31P
Example - Warnking BASSI 180
Semi-LASER 180 from MatPulse 5kHz BW
Semi-LASER 180 from MatPulse 5kHz BW_clone_2013-01-18
Semi-LASER 90 from MatPulse 4.9kHz BW
Semi-LASER 90 from MatPulse 4.9kHz BW_clone_2013-01-18

Name: 31P HypSec Inversion Trio

Creator:

Comment: Converted RFPulse Project 33df9df7-b989-48bd-a0d5-37f6dc05e7db (31P HypSec Inversion Trio) to Pulse Design on 21 March, 2016 12:05:33

Is public: False

Is frozen: False

Created: 04 January, 2013 11:30:42

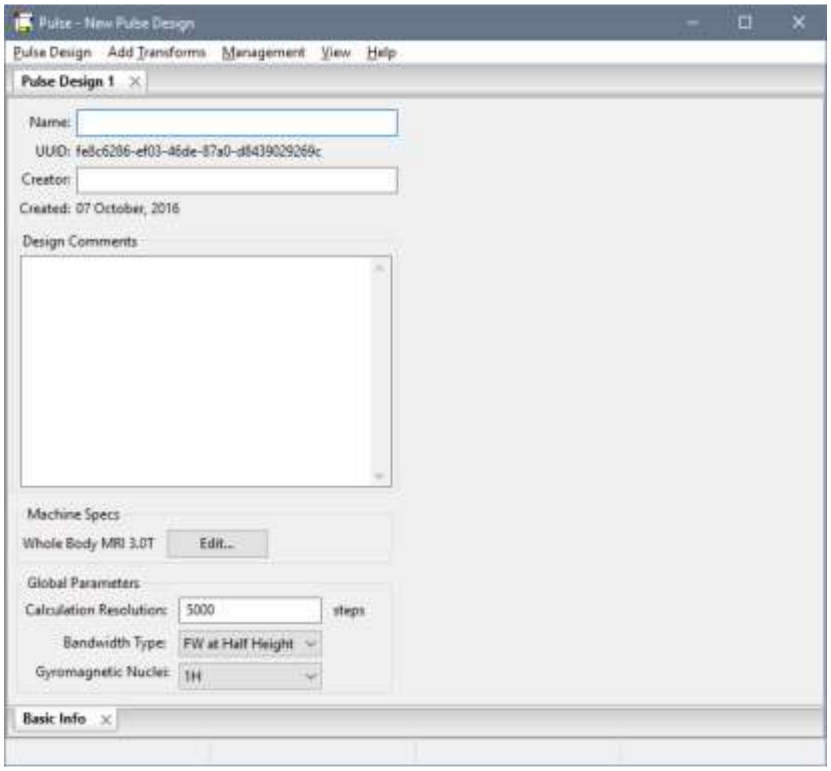
When the Open button is clicked (or a pulse design’s name is double-clicked on), the program loads the information for that pulse design from the database into a pulse design object in memory. This object then creates a pulse design tab and sets up the transformation tabs that describes its creation.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

6.2 Creating a New Pulse Design – the Basic Info Tab

As noted previously, a ‘pulse design’ object consists of one or more transformation objects. Each pulse design 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.

When you select the **Pulse Design→New** menu option, a new pulse design tab is created in the pulse design notebook. Every pulse design has a Basic Info transformation tab. This tab is where the user provides information about the pulse and can also select certain design and simulation settings to help inform the create and modify transform tabs as they are added.

The Basic Info 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. We highly suggest that you add some musings and tidbits of information to the Comments box so you know what you were thinking about 12 months from now when you come back to this design and wonder what the heck you were thinking.



Name:
UUID: fe8c6286-ef03-46de-87a0-d8439029269c
Creator:
Created: 07 October, 2016
Design Comments
Machine Specs
Whole Body MRI 3.0T
Global Parameters
Calculation Resolution: 5000 steps
Bandwidth Type: FW at Half Height
Gyromagnetic Nuclei: 1H

The Machine Specs widget allows you to select from a list of hardware/software settings templates previously set up, either provided with Pulse or created by users using the Manage Machine Settings dialog. See Section 7.3 for info on how to create your own specification templates. These include values for field strength, B1 max value and gradient waveform limits that can be used to limit the algorithms within create and modify transform tabs. Note that not all transform tabs make use of these Machine Specs settings, but they are available if users want them as part of their algorithms.

The Global Parameters section lets users change parameters used globally to calculate RF waveforms and the Bloch simulations for spectral analysis of the results. The Calculation Resolution sets the spectral resolution across the RF pulse profile. The higher the number the more points are calculated across the bandwidth, but the longer the simulations may take. This is generally not too much of a penalty up to 5000 points or so. The Bandwidth Type is typically set to Full Width Half Height, but has other settings should some algorithm in the future want to use it. The Gyromagnetic Nuclei allows the user to specify what value the program provides for the gyromagnetic ratio in transform algorithms and for the Bloch simulation calculation.

Note. Early versions of the transforms provided in Pulse were all hard set to 1H gyromagnetic ratio values. More recently, we rewrote the Bloch simulator and new versions of transforms to accept this system value. These transforms have the term 'multinuc' in their labels. If you use an old (1H) transform and set this widget value to a non-1H setting, the Bloch simulation will not scale correctly.

After filling in the Basic Info widgets, a list of available transformations is on the **Add Transforms** menu. Create transformations are found under the **Add Transforms** →**Create** sub-menu. All the other transformations directly under the **Add Transforms** menu are general transformations. A detailed description of each transformation and its general usage can be found in Appendix B.

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.

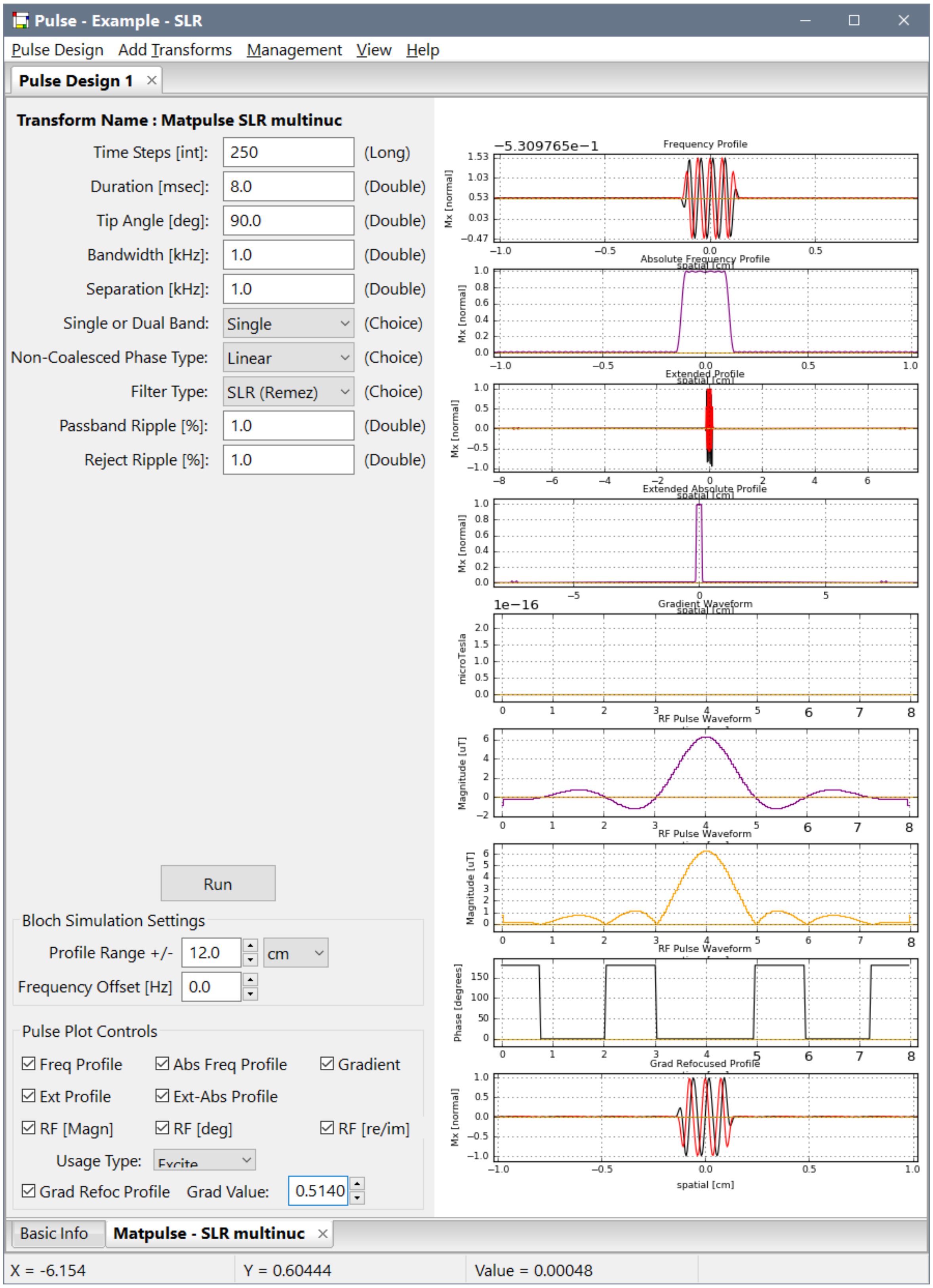
6.3 Visualizing Pulse Design Results

Transforms 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 design, Pulse 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.



Transform Name : Matpulse SLR multinuc

Time Steps [int]:	250	(Long)
Duration [msec]:	8.0	(Double)
Tip Angle [deg]:	90.0	(Double)
Bandwidth [kHz]:	1.0	(Double)
Separation [kHz]:	1.0	(Double)
Single or Dual Band:	Single	(Choice)
Non-Coalesced Phase Type:	Linear	(Choice)
Filter Type:	SLR (Remez)	(Choice)
Passband Ripple [%]:	1.0	(Double)
Reject Ripple [%]:	1.0	(Double)

Bloch Simulation Settings

Profile Range +/- 12.0 cm

Frequency Offset [Hz] 0.0

Pulse Plot Controls

☑ Freq Profile ☑ Abs Freq Profile ☑ Gradient

☑ Ext Profile ☑ Ext-Abs Profile

☑ RF [Magn] ☑ RF [deg] ☑ RF [re/im]

Usage Type: Excite

☑ Grad Refoc Profile Grad Value: 0.5140

X = -6.154 Y = 0.60444 Value = 0.00048

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.
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.
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
RF [Magn]	(checkbox) Selects the magnitude format of the time domain RF pulse waveform to be plotted.
RF [deg]	(checkbox) Selects the phase (in degrees) format of the time domain RF pulse waveform to be plotted.
RF [Re/Im]	(checkbox) Selects 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, Pulse 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, Pulse 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.

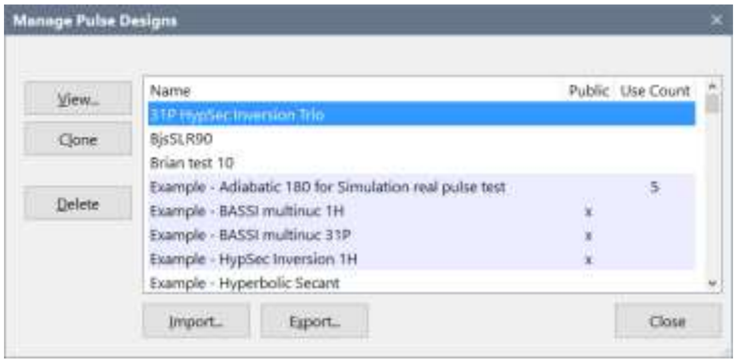
In a similar fashion, two vertical cursors can be set inside each plot window. Click and drag the right mouse button then release to set the two cursors anywhere in the window. This Cursor Span will display as a light gray span. Click and release the right mouse button in place and the cursor span will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

7. Management Dialogs

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

7.1 Manage Pulse Designs dialog

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



Name	Public	Use Count
31P HypSec Inversion Trio		
BjsSLR90		
Brian test 10		
Example - Adiabatic 180 for Simulation real pulse test		5
Example - BASSI multinuc 1H	x	
Example - BASSI multinuc 31P	x	
Example - HypSec Inversion 1H	x	
Example - Hyperbolic Secant		

View: Displays a textual summary of the pulse design.

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

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

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

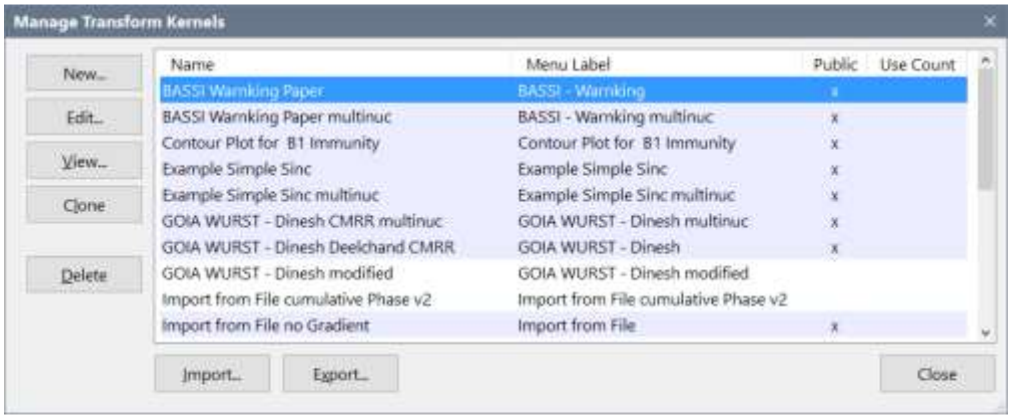
Export: You select a pulse design from the list. A second dialog allows you to browse for the output filename, select if output should be compressed and allows an additional export comment to be typed in.

Under the Hood

- The action of exporting a pulse design (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 design can still be deleted from the database if needed. This file can be imported into another Vespa-Pulse installation using the Import function. If additional changes are desired a new pulse design, cloned from the original "frozen" object, can be created then modified.
- Pulse PulseDesign 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.

7.2 Manage Transform Kernels dialog

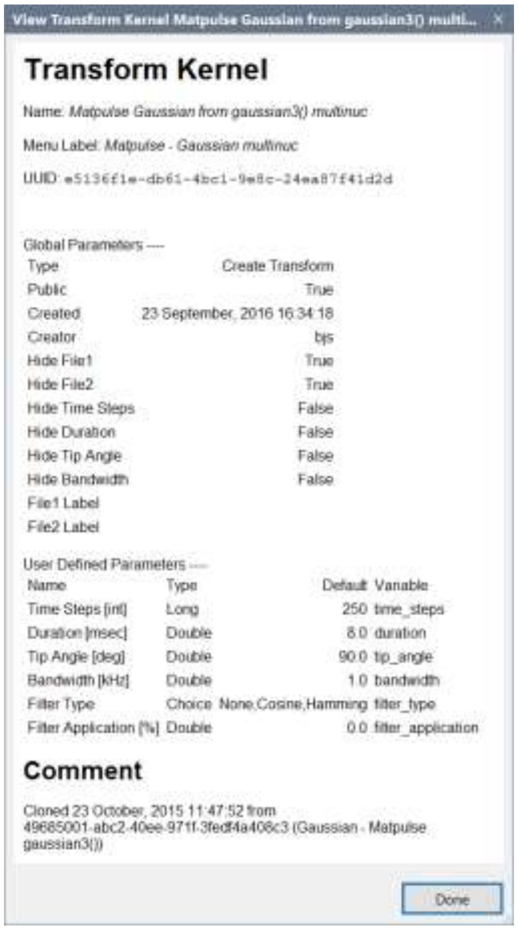
Access this dialog by clicking on the **Management→Manage Transform Kernels** menu item. Actions that can be taken on the dialog include: New, Edit, View, Clone, Delete, Import and Export. An example of this dialog is shown below. The New, Edit, View, Import and Export buttons all launch secondary dialogs as part of their functionality. Clone and Delete only affect the list in the main transform kernel management dialog.



Manage Transform Kernels

Name	Menu Label	Public	Use Count
BASSI Warnking Paper	BASSI - Warnking	x	
BASSI Warnking Paper multinuc	BASSI - Warnking multinuc	x	
Contour Plot for B1 Immunity	Contour Plot for B1 Immunity	x	
Example Simple Sinc	Example Simple Sinc	x	
Example Simple Sinc multinuc	Example Simple Sinc multinuc	x	
GOIA WURST - Dinesh CMRR multinuc	GOIA WURST - Dinesh multinuc	x	
GOIA WURST - Dinesh Deelchand CMRR	GOIA WURST - Dinesh	x	
GOIA WURST - Dinesh modified	GOIA WURST - Dinesh modified		
Import from File cumulative Phase v2	Import from File cumulative Phase v2		
Import from File no Gradient	Import from File	x	

The “Name” column displays the long descriptive name given for the kernel by the user. The “Menu Label” column shows the short name used in the Pulse application Add Transforms menu. The "Public" column indicates if a sequence has ever been exported (or imported from someone else). Pulse Sequences with the Public column marked ‘x’ cannot be edited except in the Name and Comment fields. However, you can still Clone it and make changes to the copy. The "Use Count" column indicates how many local Experiments use this sequence. While in use by any Experiments, the sequence cannot be deleted.



View Transform Kernel Matpulse Gaussian from gaussian3() multi...

Transform Kernel

Name: *Matpulse Gaussian from gaussian3() multinuc*

Menu Label: *Matpulse - Gaussian multinuc*

UUID: `e5136f1e-db61-4bc1-9e8c-24ea87f41d2d`

Global Parameters ----

Type	Create Transform
Public	True
Created	23 September, 2016 16:34:18
Creator	bjs
Hide File1	True
Hide File2	True
Hide Time Steps	False
Hide Duration	False
Hide Tip Angle	False
Hide Bandwidth	False
File1 Label	
File2 Label	

User Defined Parameters ----

Name	Type	Default	Variable
Time Steps [int]	Long	250	time_steps
Duration [msec]	Double	8.0	duration
Tip Angle [deg]	Double	90.0	tip_angle
Bandwidth [kHz]	Double	1.0	bandwidth
Filter Type	Choice	None,Cosine,Hamming	filter_type
Filter Application [%]	Double	0.0	filter_application

Comment

Cloned 23 October, 2015 11:47:52 from 49685001-abc2-40ee-971f-3fedf4a408c3 (Gaussian - Matpulse gaussian3())

View: Select a transform kernel from the main list. If more than one is selected the first on in the list is viewed. This button pops up a secondary dialog that contains a brief overview of the kernel structure. See figure right for example of View.

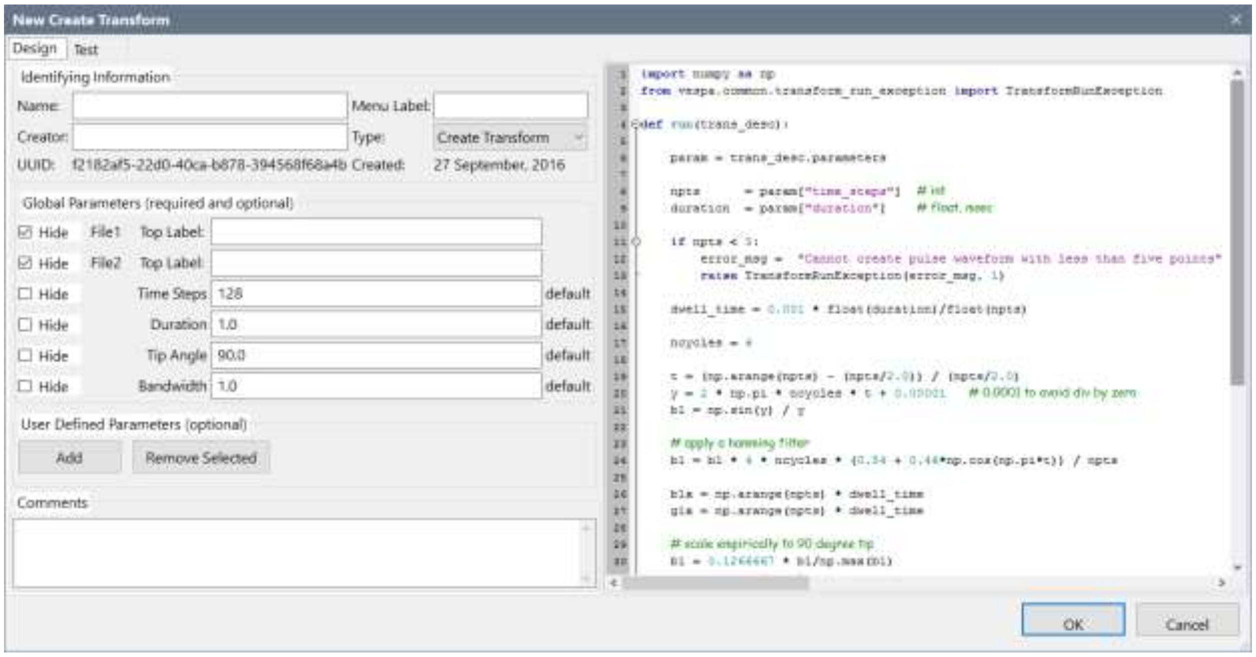
Clone: This option allows a user to make a copy of an existing transform kernel. This is most useful when an existing transform kernel is “public” or otherwise not editable because it is referenced by an existing Design. Select a transform kernel in the list, hit clone and a copy of that transform kernel is made that is now fully editable. The new transform kernel has the name of the original transform kernel followed by the date and the word "_clone".

Delete: Only transform kernels that are not referenced by a Design should be deleted. To reconstruct any given Design, that object must refer to the original kernel used to create it. The "Use Count" column indicates if a kernel is in use by a Design. If not in use by a Design, the highlighted kernel(s) in the list can be deleted from the database.

Import: Pops up a secondary dialog. The user selects an XML file that contains one or more Vespa Pulse transform kernels. Any kernels in the file are added to the database, provided that they aren't in the database already. Transform kernels with UUIDs that match those already in the database are simply ignored. Please be sure to import/export pulse transform kernels with the “Manage Transform Kernels” utility to ensure proper operation.

Export: Select one or more transform kernel(s) from the main list. A second dialog pops up that allows the user to browse for the output filename, select if output should be compressed and allows an additional export comment to be typed in. Note that the action of exporting an object causes it to be marked as “frozen” in the database and “public” in the transform kernel management dialog. This means that it cannot be changed. This is for the sake of consistency as results are shared. However, a frozen transform kernel can still be deleted from the database if needed. This file can be imported into another Vespa-Pulse installation using the Import function. Please be sure to import/export pulse sequences with the “Manage Transform Kernels” utility to ensure proper operation.

New: This button starts another dialog called the “Transform Kernel Editor”. This allows the user to design and test a transform kernel. The user must provide general descriptive information about the kernel. They must describe how to lay out the kernel input parameters in the GUI of a Transform tab. The user must also provide Python code for the algorithm that is performed within the Transform step. When the dialog opens, there is default code for a simple pulse in the code window as an example to help you get started. You can keep this code, alter it, replace it or delete it entirely. Other existing transform kernels can also be sources of examples and inspiration from which to start.



Name:
Menu Label:
Creator:
Type: Create Transform
UUID: f2182af5-22d0-40ca-b878-394568f68a4b Created: 27 September, 2016

☑ Hide	File1	Top Label:		
☑ Hide	File2	Top Label:		
☐ Hide		Time Steps	128	default
☐ Hide		Duration	1.0	default
☐ Hide		Tip Angle	90.0	default
☐ Hide		Bandwidth	1.0	default

```
import numpy as np
from vespa.common.transform_run_exception import TransformRunException

def run(trans_desc):

    param = trans_desc.parameters

    npts      = param["time_steps"]   # int
    duration  = param["duration"]     # float, msec

    if npts < 5:
        error_msg =  "Cannot create pulse waveform with less than five points"
        raise TransformRunException(error_msg, 1)

    dwell_time = 0.001 * float(duration)/float(npts)

    ncycles = 4

    t = (np.arange(npts) - (npts/2.0)) / (npts/2.0)
    y = 2 * np.pi * ncycles * t + 0.00001    # 0.00001 to avoid div by zero
    b1 = np.sin(y) / y

    # apply a hamming filter
    b1 = b1 * 4 * ncycles * (0.54 + 0.46*np.cos(np.pi*t)) / npts

    b1x = np.arange(npts) * dwell_time
    gx = np.arange(npts) * dwell_time

    # scale empirically to 90 degree tip
    b1 = 0.1266667 * b1/np.max(b1)
```

The New Pulse Sequence Editor dialogis shown above. It consists of a Notebook widget with two tabs: 1) the Design tab (shown) and 2) the Display tab (not shown). To create a transform kernel, fill in the various sections of the Design tab. When you hit the OK button (bottom right), your Design is validated, and if sufficient data has been entered, the dialog saves your transform kernel to the Vespa database, the Transform Kernel Editor dialog goes away and you should see your new transform kernel listed in the main Transform Kernel Manager list of kernels. Note. The ‘validation’ process does not test if the algorithm or parameters produce correct (or any) results, but rather if enough fields are filled with the right type of data for the object to be saved. If you do not wish to save your pulse sequence, hit Cancel.

When you have filled in the Design tab with a valid set of meta-data, parameters and algorithm code, you can switch to the Test tab. At this point the editor also validates your design to see if any Design fields are missing or values are incorrectly set in your fields. It then creates an actual Transform sub-tab within the Test tab based on the parameters you've provided. You will see your kernel displayed in a Transform exactly as it will be in a Pulse Design notebook (eventually) in the main Pulse program. In the Test tab, you can look to see that your parameters display properly, change their values, and hit the Run button to see if your algorithm code runs properly.

If you are making a new 'Create' kernel, your algorithm is expected to produce an RF waveform. If you are making a new 'Modify' kernel, your algorithm will be passed a simple sinc function to simulate an RF waveform result from a previous Transform. In either case, at the completion of the Run button, a Bloch simulation of the RF waveform/gradient results is run and displayed for you to evaluate the performance of your algorithm. If an error is detected at any point in the process, it is reported in the Console window (lower left) of the Test tab.

The "Design" and "Test" tabs allow you to test your transform kernel before running it in a Pulse Design in the main application. Effectively, it allows you to run a single Transform step of a Pulse Design to test your algorithm in isolation. More information on these is provided below.

Design Tab – Data input fields

Name: This is a field to enter a longer name (don't go crazy) that will be displayed in the Transform Kernel Manager dialog. It must be a unique string from all other kernel names

Menu Name: Type a short name that will be displayed in the **Add Transforms** menu. It must be a unique string from all other kernel menu names.

Creator: The name of the person creating the transform kernel

Type: This is a drop menu. At this point the only choices are for 'Create' or 'Modify'. This is the type of kernel you are designing.

Global Parameters: This is a list of parameters that are frequently used in transform kernels including: Time Steps, Duration, Tip Angle and Bandwidth. Two 'File' widgets can also be enabled. Click on the 'Hide' check box to NOT display a parameter. Enter a default value in each widget that you want to use. The 'default value' for a Field parameter is actually a label displayed above the Browse button for that File. Make sure that the default values you use will give a valid result in your algorithm are the right type (int, float, etc).

Your User Defined Parameter Definitions: This section allows you to add/remove parameters which will eventually be passed into the transform kernel algorithm. The Transform GUI can be populated with additional parameters you need to perform your algorithm. However, they have to be describable as Float, Long, String or Choice (drop menu) widget types so that the main program can display them properly. By hitting the Add button, a row of widgets will appear that contain four fields used to describe the GUI for one parameter: A data type (selected from a drop-list), a "Name" string, a "Default Value" string, and a "Variable Name" string that you will use in your algorithm. The Name string will be used by the Transform tab as a label to describe this field when the pulse sequence is selected for an Experiment. The data type shows up as a label in the far right hand side as a reminder. The default value is inserted as the initial value that is displayed in that field. The Variable string does not appear in the GUI, but will be used as a key name in the dictionary for the value of this parameter sent to the algorithm.

The default value for a Choice parameter is used as the list of items for the drop down menu. It should be entered into the text field as a comma separated string such as “one,two,three”. Don’t use spaces in this string unless you want them in the menu item. The Choice parser does no trimming of white space, just separates the items between commas. The value returned for a Choice parameter is the drop list index. For example, if we selected the list item “two” from the above example, the variable would have the value of 1, since it is the second in the zero indexed list. You can easily simulate a Boolean variable with a Choice parameter by setting the default value to “Off,On” which returns 0 or 1 respectively.

The Remove Selected button can be used to remove unwanted parameters while designing a pulse sequence. Select the check box on the left side of each row of parameters you want to remove, and then hit the Remove Selected button.

Note1: Parameters are displayed in the order that they are entered into the list. You can not move parameters around once they are entered other than to delete them. Please be careful to organize your parameters before you start to enter them.

Note2: By selecting a data type for a user-specified parameter in the drop down menu, the user will be reminded to enter a variable of that type. Please select your default types and values accordingly.

Comments: A field where you can enter a lot of text to remind yourself why you made this transform kernel when you check back on it 3 months from now. This is also a good place to put information for users on how to use this transform kernel.

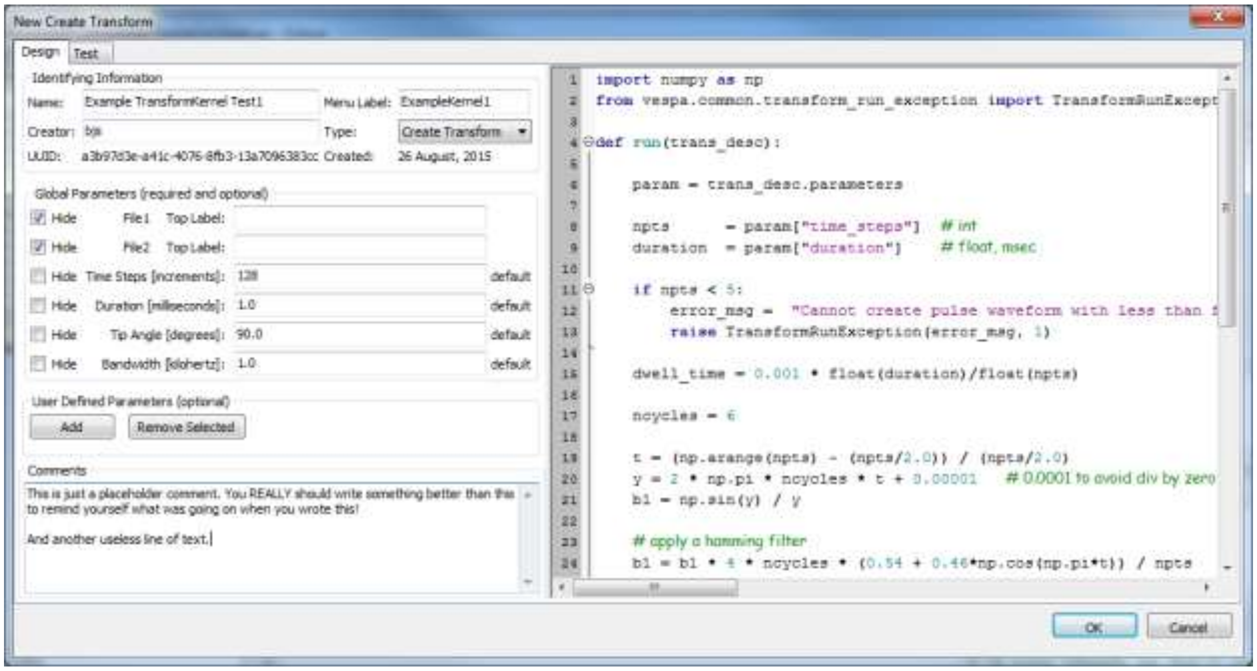
Code Window: The whole right hand pane is a text editor into which you must enter the code of your transform kernel algorithm.

Test Tab

When the user clicks on the Test tab, the settings in the Design tab are validated, and if passed, then the Test tab is updated to reflect the transform kernel design. If there are any missing fields or other errors in the Design tab, the user is prompted to fix these prior to switching to the Test tab.

Note: A similar validation takes place when the user hits the OK button. Only a validated pulse sequence can be saved into the database. However, the validation only checks to see if all necessary data is available in a reasonable format, NOT if it is functional transform kernel code.

An example is shown below of how the default settings in the Design tab are represented on the Test tab. Note that the default values have been used in the Test tab and the Run button has been hit to display the results shown.



Identifying Information

Name: Example TransformKernel Test1 Menu Label: ExampleKernel1
Creator: bjs Type: Create Transform
UUID: a3b97d3e-a41c-4076-8fb3-13a7096383cc Created: 26 August, 2015

Global Parameters (required and optional)

[✓] Hide	File1 Top Label:		
[✓] Hide	File2 Top Label:		
[ ] Hide	Time Steps [increments]:	128	default
[ ] Hide	Duration [milliseconds]:	1.0	default
[ ] Hide	Tip Angle [degrees]:	90.0	default
[ ] Hide	Bandwidth [kilohertz]:	1.0	default

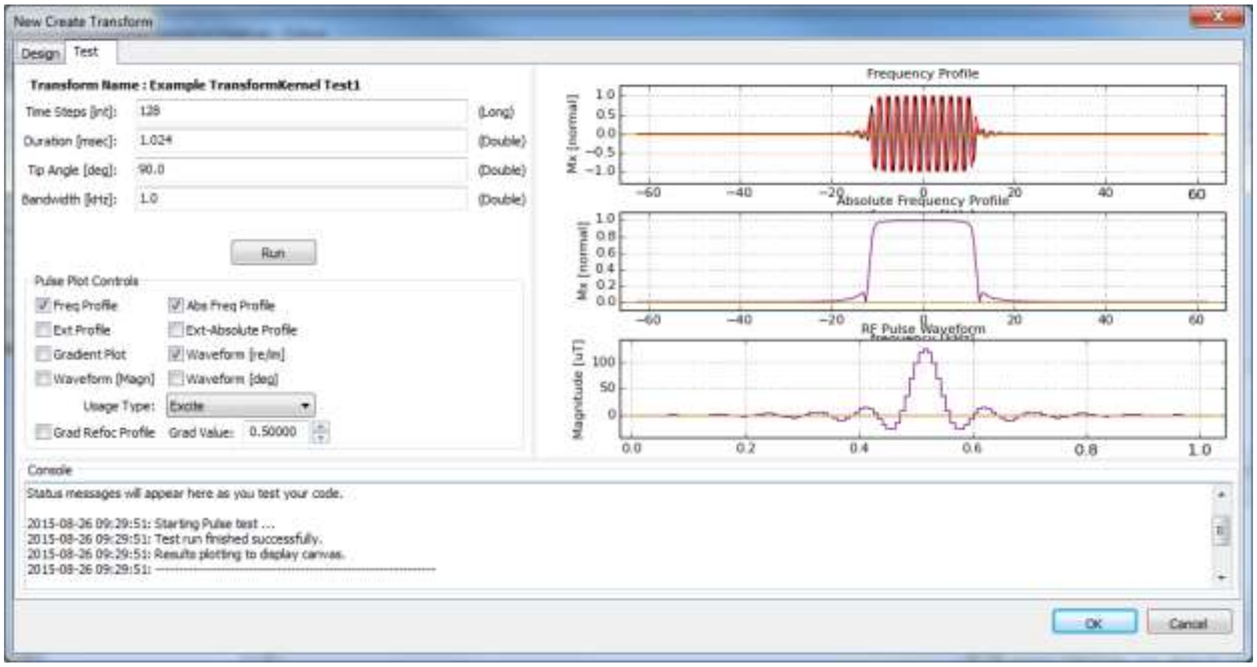
User Defined Parameters (optional)

Comments

This is just a placeholder comment. You REALLY should write something better than this to remind yourself what was going on when you wrote this!

And another useless line of text.

```
import numpy as np
from vespa.common.transform_run_exception import TransformRunExcept

def run(trans_desc):

    param = trans_desc.parameters

    npts      = param["time_steps"]   # int
    duration  = param["duration"]     # float, msec

    if npts < 5:
        error_msg =  "Cannot create pulse waveform with less than f
        raise TransformRunException(error_msg, 1)

    dwell_time = 0.001 * float(duration)/float(npts)

    ncycles = 6

    t = (np.arange(npts) - (npts/2.0)) / (npts/2.0)
    y = 2 * np.pi * ncycles * t + 0.00001     # 0.0001 to avoid div by zero
    b1 = np.sin(y) / y

    # apply a hamming filter
    b1 = b1 * 4 * ncycles * (0.54 + 0.46*np.cos(np.pi*t)) / npts
```



Transform Name : Example TransformKernel Test1

Time Steps [int]:	128	(Long)
Duration [msec]:	1.024	(Double)
Tip Angle [deg]:	90.0	(Double)
Bandwidth [kHz]:	1.0	(Double)

Pulse Plot Controls

[✓] Freq Profile [✓] Abs Freq Profile
[ ] Ext Profile [ ] Ext-Absolute Profile
[ ] Gradient Plot [✓] Waveform [re/im]
[ ] Waveform [Magn] [ ] Waveform [deg]
Usage Type: Excite
[ ] Grad Refoc Profile Grad Value: 0.50000

Console

```
Status messages will appear here as you test your code.

2015-08-26 09:29:51: Starting Pulse test ...
2015-08-26 09:29:51: Test run finished successfully.
2015-08-26 09:29:51: Results plotting to display canvas.
2015-08-26 09:29:51: ----------------------------------------------------
```

Transform Tab: The top two thirds of the Test tab example above contains a Transform tab representation of your transform kernel design identical to that which will display (eventually) in the main program.

User Parameters: The top left third contains the name of the transform kernel, and a vertical list of the user parameters (label, value, type) that you can use to control your algorithm.

Run Test Button: Runs the Transform in the tab. The Start and End times should be reported in the Console window. Any additional exceptions that are raised should be reported between these messages. Note. This test assumes that the system gamma value is for 1H. In the main program, you can select other values if you've incorporated it into your transform kernel.

Pulse Plot Control: The middle left third contains widgets to control the plots displaying the RF waveform and gradient results on the right hand side of the Transform tab.

Pulse Plot: This is the whole right hand top two thirds of the Test tab. The test RF waveform and gradient results are plotted to this display area. The Left mouse button can be used to draw a zoom box in both x and y directions. Multiple zooms can be performed. Left clicking once in place will zoom you all the way out to the maximum x-axis extent and fit the y-axis to approximately the min/max data range.

Console: The full bottom third of the Test tab contains a console window which displays descriptions of what happens when the Run button is pressed in the Transform tab above. In the example above, it only reports that the Pulse test started and finished successfully. If the algorithm had crashed before finishing, the line and exception report thrown would be reported in this window.

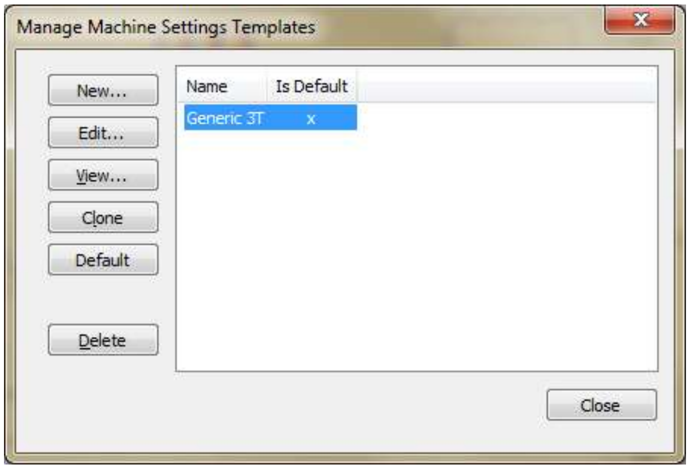
Console: The place where text messages about each Test Run are printed.

Edit: The first highlighted sequence is opened in a form similar to the New Transform Kernel dialog. Note: Only the transform kernel Name, and Comment are editable if the transform kernel is "public" or referred to by one or more Pulse Designs. The name is editable because Pulse Designs save Transform Kernels references by UUID which are not editable. Use the "Clone" option to create a copy of a Transform Kernel that is editable.

If the transform kernel is editable, the existing values of the transform kernel object are populated into the Design and Test tabs on startup. The name of the transform kernel from the main dialog is shown in the dialog title. The transform kernel settings can be edited and tested just like a New transform kernel would be. Hitting OK saves any changes into the database. Cancel quits the dialog without saving changes.

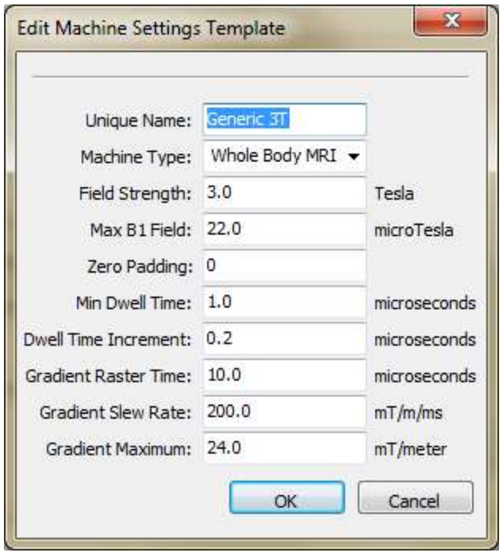
7.3 Manage Machine Specs Templates dialog

Access this dialog by clicking on the **Management→Manage Machine Specs 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 specs for the machine specs object in a newly created pulse design 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.



Name	Is Default
Generic 3T	x

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



Unique Name:	Generic 3T	
Machine Type:	Whole Body MRI	
Field Strength:	3.0	Tesla
Max B1 Field:	22.0	microTesla
Zero Padding:	0	
Min Dwell Time:	1.0	microseconds
Dwell Time Increment:	0.2	microseconds
Gradient Raster Time:	10.0	microseconds
Gradient Slew Rate:	200.0	mT/m/ms
Gradient Maximum:	24.0	mT/meter

Edit: The highlighted metabolite is opened in the machine specs 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 specs in a new pulse design object.

8. Results Output

8.1 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-Pulse **View** menu lists commands that only apply to the active transformation tab in the active pulse design 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.

8.2 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-Pulse **View** menu lists commands that only apply to the active transformation tab in the active pulse design 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.

8.3 Write RF waveform/gradient results to file format

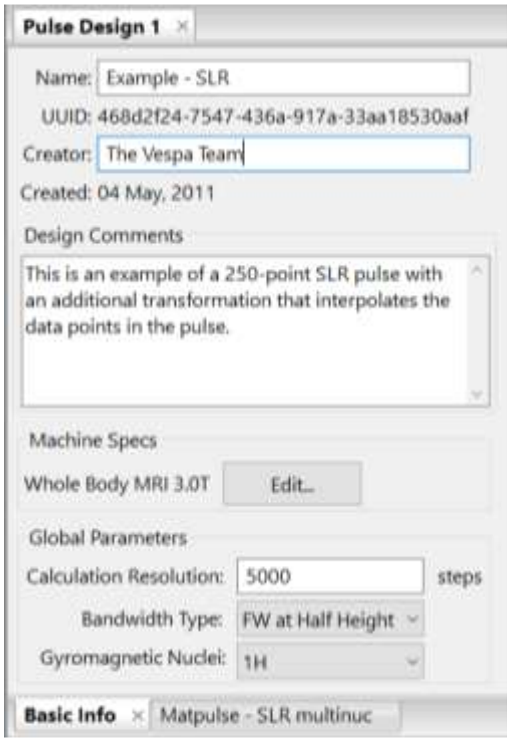
See Appendix B for 3rd Party Export options.

Appendix A. List of Standard Transforms

This section provides some basic information about the transformations (both “create” and “general” types) available in the Pulse 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.

A.1 Basic Info Tab

A.1.1 Tab Diagram



Name: Example - SLR
UUID: 468d2f24-7547-436a-917a-33aa18530aaf
Creator: The Vespa Team
Created: 04 May, 2011

This is an example of a 250-point SLR pulse with an additional transformation that interpolates the data points in the pulse.

Whole Body MRI 3.0T

Calculation Resolution: 5000 steps
Bandwidth Type: FW at Half Height
Gyromagnetic Nuclei: 1H

A.1.2 General Usage

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

A.1.3 Widgets and Parameters

<u>Name</u> – text, must be a unique text string within the group of pulse design names in the database

<u>Investigator</u> – text, (optional)

<u>UUID and Created</u> – not editable, just given FYI

<u>Machine Settings</u> – label, listing the name of the currently selected machine settings.

<u>Edit</u> – 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

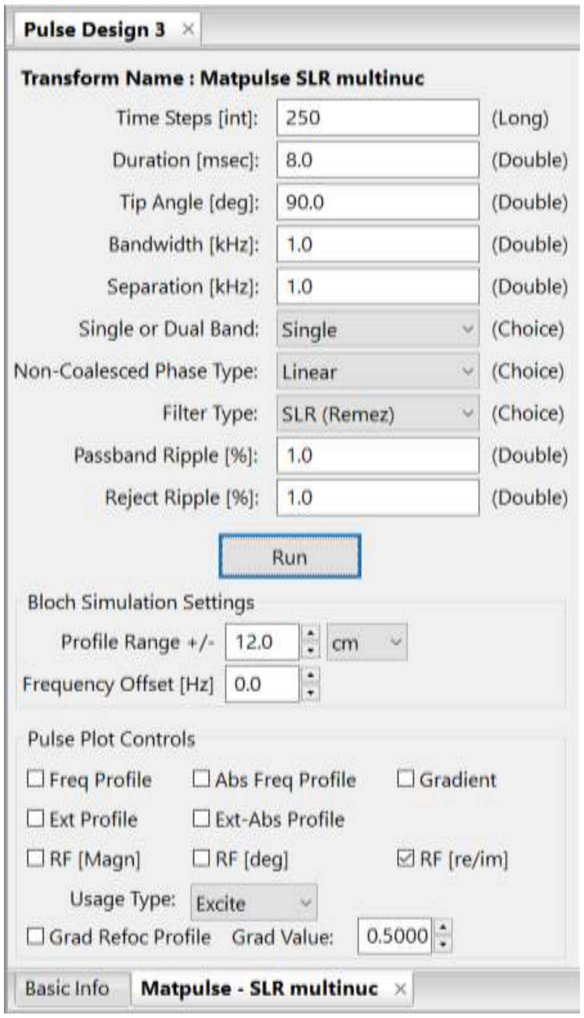
Gyromagnetic Nuclei – droplist, a list of nuclei (1H, 19F, 31P, etc.) that the use can select to set the system gamma value. This value is passed into each transform kernel calculation for the user to include in their algorithm. It is also used but the Bloch simulation engine. So for the proper pulse profile to be displayed in the plot windows, both calculations should use the same gamma value. Older transforms may not use the system gamma, in which case the 1H value is the default.

A.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.

A.2 SLR Tab (create transformation)

A.2.1 Tab Diagram



Pulse Design 3

Transform Name : Matpulse SLR multinuc

Time Steps [int]:	250	(Long)
Duration [msec]:	8.0	(Double)
Tip Angle [deg]:	90.0	(Double)
Bandwidth [kHz]:	1.0	(Double)
Separation [kHz]:	1.0	(Double)
Single or Dual Band:	Single	(Choice)
Non-Coalesced Phase Type:	Linear	(Choice)
Filter Type:	SLR (Remez)	(Choice)
Passband Ripple [%]:	1.0	(Double)
Reject Ripple [%]:	1.0	(Double)

Run

Bloch Simulation Settings

Profile Range +/- 12.0 cm

Frequency Offset [Hz] 0.0

Pulse Plot Controls

☐ Freq Profile ☐ Abs Freq Profile ☐ Gradient

☐ Ext Profile ☐ Ext-Abs Profile

☐ RF [Magn] ☐ RF [deg] ☑ RF [re/im]

Usage Type: Excite

☐ Grad Refoc Profile Grad Value: 0.5000

Basic Info | **Matpulse - SLR multinuc**

A.2.2 General Usage

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

A.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

A.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.

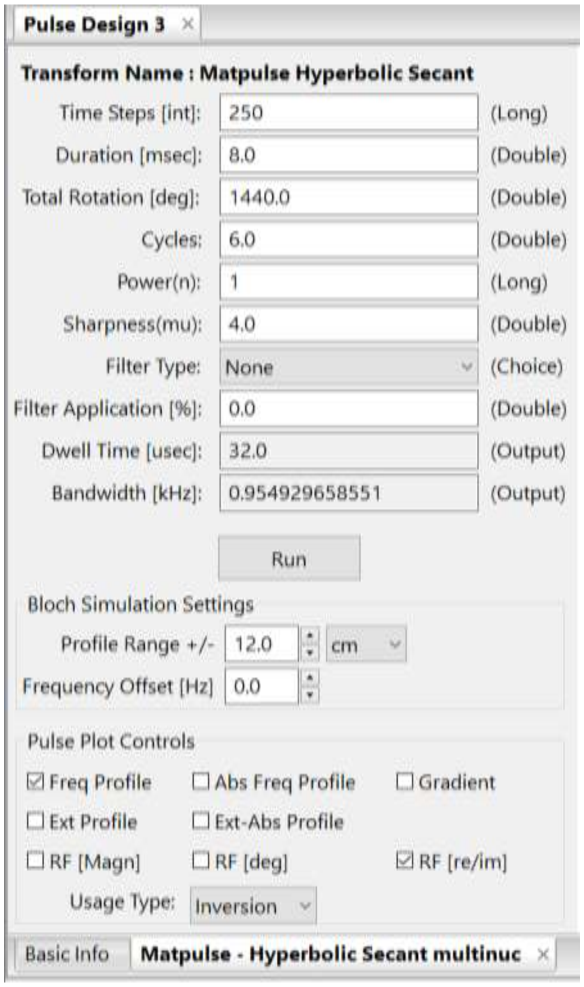
The second important conceptual component of the SLR algorithm is the hard pulse approximation (Pauly et al. 1991), which is useful in particular in the case of non-adiabatic pulses. The hard pulse approximation states that any shaped, or soft, pulse $B_1(t)$ can be approximated by a series of short 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-Pulse 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.

A.3 Hyperbolic-Secant Tab (create transform)

A.3.1 Tab Diagram


Pulse Design 3

Transform Name : Matpulse Hyperbolic Secant

Time Steps [int]:	250	(Long)
Duration [msec]:	8.0	(Double)
Total Rotation [deg]:	1440.0	(Double)
Cycles:	6.0	(Double)
Power(n):	1	(Long)
Sharpness(mu):	4.0	(Double)
Filter Type:	None	(Choice)
Filter Application [%]:	0.0	(Double)
Dwell Time [usec]:	32.0	(Output)
Bandwidth [kHz]:	0.954929658551	(Output)

Bloch Simulation Settings

Profile Range +/- 12.0 cm

Frequency Offset [Hz] 0.0

Pulse Plot Controls

☑ Freq Profile ☐ Abs Freq Profile ☐ Gradient

☐ Ext Profile ☐ Ext-Abs Profile

☐ RF [Magn] ☐ RF [deg] ☑ RF [re/im]

Usage Type: Inversion

A.3.2 General Usage

This is a “create” transform tab.

A.3.3 Widgets and Parameters

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

Duration – float, in milli-seconds,

Total Rotation – float, in degrees,

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)

Dwell Time (Output) – float, in degrees, calculated by the algorithm

Bandwidth – float, in kHz, calculated by the algorithm

A.3.4 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:

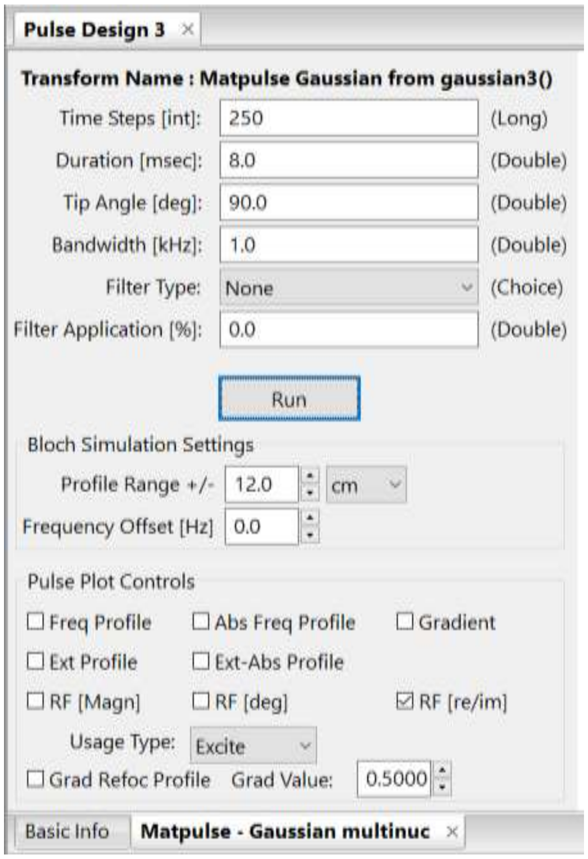
$$
\begin{aligned}
B_1(t) &= A(t)e^{-i\omega_1(t)t} \\
A(t) &= A_0 \operatorname{sech}(\beta t) \\
\omega(t) &= -\mu\beta \tanh(\beta t)
\end{aligned}
$$

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.

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

A.4 Gaussian Tab (create transform)

A.4.1 Tab Diagram


Pulse Design 3

Transform Name : Matpulse Gaussian from gaussian3()

Time Steps [int]:	250	(Long)
Duration [msec]:	8.0	(Double)
Tip Angle [deg]:	90.0	(Double)
Bandwidth [kHz]:	1.0	(Double)
Filter Type:	None	(Choice)
Filter Application [%]:	0.0	(Double)

Run

Bloch Simulation Settings

Profile Range +/- 12.0 cm

Frequency Offset [Hz] 0.0

Pulse Plot Controls

☐ Freq Profile ☐ Abs Freq Profile ☐ Gradient

☐ Ext Profile ☐ Ext-Abs Profile

☐ RF [Magn] ☐ RF [deg] ☑ RF [re/im]

Usage Type: Excite

☐ Grad Refoc Profile Grad Value: 0.5000

Basic Info | Matpulse - Gaussian multinuc

A.4.2 General Usage

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

A.4.3 Widgets and Parameters

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.

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.

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.

A.4.4 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.

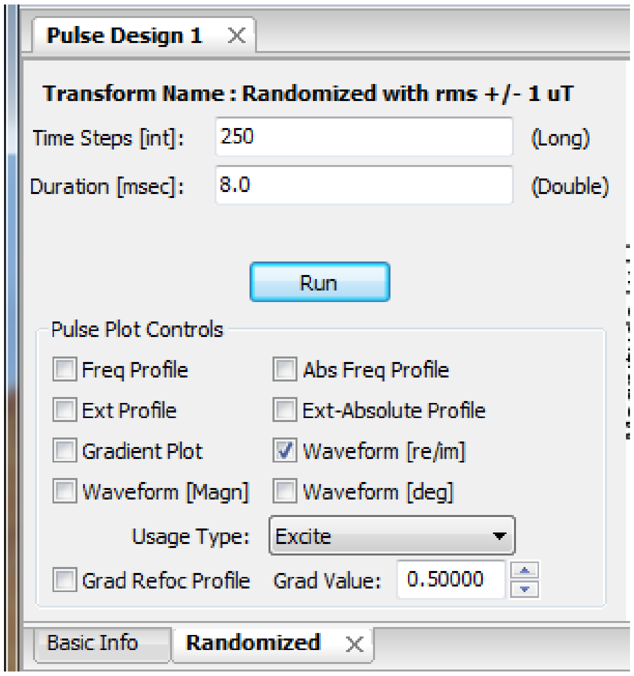
However, with the output plots that can be displayed for M_{xy} and M_z magnetizations, users can iteratively dial in a bandwidth that achieve the actual FWHM bandwidth desired.

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.

A.5 Randomized Tab (create transform)

A.5.1 Tab Diagram


Pulse Design 1

Transform Name : Randomized with rms +/- 1 uT

Time Steps [int]: 250 (Long)

Duration [msec]: 8.0 (Double)

[ ] Freq Profile [ ] Abs Freq Profile
[ ] Ext Profile [ ] Ext-Absolute Profile
[ ] Gradient Plot [✓] Waveform [re/im]
[ ] Waveform [Magn] [ ] Waveform [deg]

Usage Type: Excite

[ ] Grad Refoc Profile Grad Value: 0.50000

A.5.2 General Usage

This is a “create” transform tab. The primary purpose of this create transform 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.

A.5.3 Widgets and Parameters

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

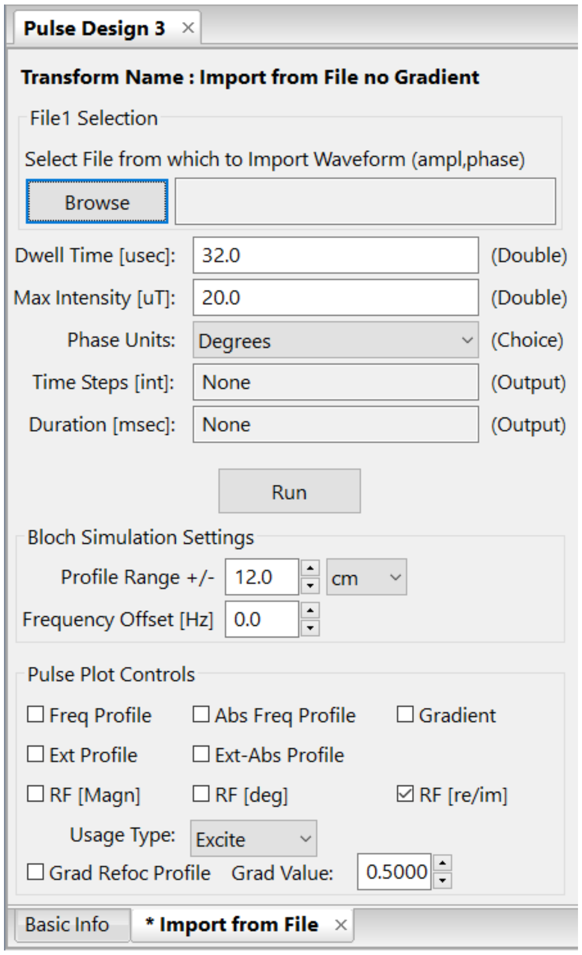
Duration – float, in milliseconds, total time of the pulse. Values selected may not be achievable due to the minimum dwell time and dwell time increment set in the machine settings. In this case, an achievable value will be suggested.

A.5.4 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.

A.6 Import from File (create transform)

A.6.1 Tab Diagram



Pulse Design 3

Transform Name : Import from File no Gradient

File1 Selection

Select File from which to Import Waveform (ampl,phase)

Browse

Dwell Time [usec]:	32.0	(Double)
Max Intensity [uT]:	20.0	(Double)
Phase Units:	Degrees	(Choice)
Time Steps [int]:	None	(Output)
Duration [msec]:	None	(Output)

Run

Bloch Simulation Settings

Profile Range +/- 12.0 cm

Frequency Offset [Hz] 0.0

Pulse Plot Controls

☐ Freq Profile ☐ Abs Freq Profile ☐ Gradient

☐ Ext Profile ☐ Ext-Abs Profile

☐ RF [Magn] ☐ RF [deg] ☑ RF [re/im]

Usage Type: Excite

☐ Grad Refoc Profile Grad Value: 0.5000

Basic Info | * Import from File

A.6.2 General Usage

This is a “create” transform tab. Users can use this “create” tab to load a waveform from a text file into a PulseDesign in Pulse 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 <u>highly</u> 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.

A.6.3 Widgets and Parameters

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

<u>Comment</u> – 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!

<u>Dwell Time</u> – float, in microseconds, duration of each step in the imported pulse.

<u>Max Intensity/Scale Factor</u> – 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.

A.6.4 Algorithm Applied at Run Time

Overview – This create transformation allows users to import waveforms of their own making into the Pulse 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

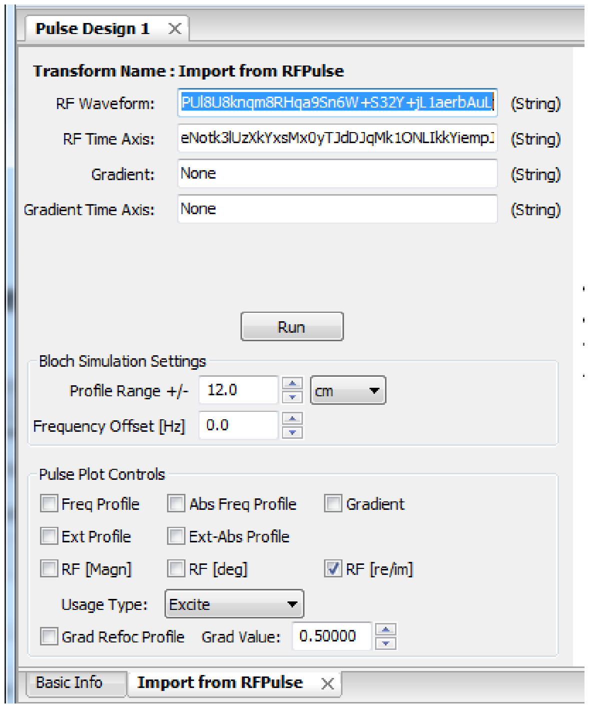
At run time, a number of values are checked both in the widgets in the tab and whether any values for a waveform can be read in from the listed file. When the Run operation is done, the waveform and processed Bloch equation results for Mxy, Mz can be viewed in the plot canvas on the right.

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 PulseDesign 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.

A.7 Import from RFPulse (create transform)

A.7.1 Tab Diagram



Pulse Design 1

Transform Name : Import from RFPulse

RF Waveform: PUl8U8knqm8RHqa9Sn6W+S32Y+jL1aerbAuL (String)

RF Time Axis: eNotk3lUzXkYxsMx0yTJdDJqMk1ONLIkkYiempI (String)

Gradient: None (String)

Gradient Time Axis: None (String)

Run

Bloch Simulation Settings

Profile Range +/- 12.0 cm

Frequency Offset [Hz] 0.0

Pulse Plot Controls

- [ ] Freq Profile
- [ ] Abs Freq Profile
- [ ] Gradient
- [ ] Ext Profile
- [ ] Ext-Abs Profile
- [ ] RF [Magn]
- [ ] RF [deg]
- [✓] RF [re/im]

Usage Type: Excite

- [ ] Grad Refoc Profile

Grad Value: 0.50000

Basic Info | Import from RFPulse

A.7.2 General Usage

This is a “create” transform tab. The transform kernel used in this tab was created to transfer a copy of RFPulse results to Pulse when Pulse replaced RFPulse in the Vespa package. It is not anticipated that users will use this transform in daily usage of Pulse. This documentation is provided for completeness.

A.7.3 Widgets and Parameters

The final form of each RFPulse PulseProject, the RF waveform and time axis and any Gradient waveform and time axis, was encoded from their numpy array forms into a byte array format and stored in each of the parameter fields shown above. There had to be a RF waveform and time axis for a result to be transferred, but most RFPulse PulseProjects did not have specific Gradient waveforms. This is indicated by a ‘None’ in the respective field.

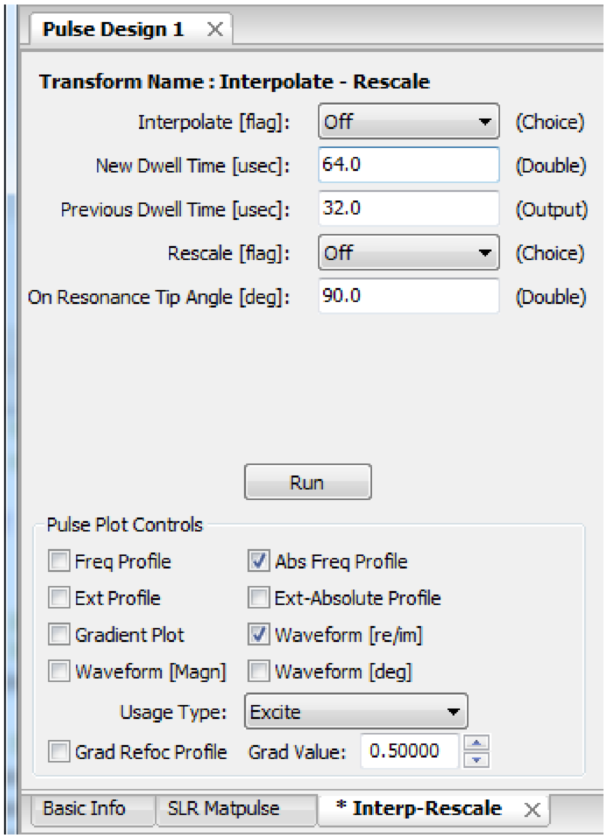
It is recommended that users do NOT change any of the values in these string fields as this will alter the pulse results obtained.

A.7.4 Algorithm Applied at Run Time

Overview – Each encoded string is decoded back into a numpy array and saved into a PulseProject result object. If the Gradient waveform and time axis entries have a string ‘None’ in them, a Python None value is returned.

A.8 Interpolate-Rescale Tab (modify transform)

A.8.1 Tab Diagram



Pulse Design 1

Transform Name : Interpolate - Rescale

Interpolate [flag]:	Off	(Choice)
New Dwell Time [usec]:	64.0	(Double)
Previous Dwell Time [usec]:	32.0	(Output)
Rescale [flag]:	Off	(Choice)
On Resonance Tip Angle [deg]:	90.0	(Double)

Pulse Plot Controls

☐ Freq Profile	✓ Abs Freq Profile
☐ Ext Profile	☐ Ext-Absolute Profile
☐ Gradient Plot	✓ Waveform [re/im]
☐ Waveform [Magn]	☐ Waveform [deg]

Usage Type: Excite

☐ Grad Refoc Profile Grad Value: 0.50000

A.8.2 General Usage

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

A.8.3 Widgets and Parameters

Interpolate – choice, turns Interpolation algorithm on/off.

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.

Current Dwell Time – float, the dwell time from the previous results, i.e. the results from the previous transform tab. This value is only reported after this transform has been run (and had access to the previous transform). One workaround is to run this transform with both Choice widgets set to ‘Off’. This will populate this field without changing the previous result.

Rescaling – choice, turns Rescaling algorithm on/off.

On Resonance Tip – float, equal to the new value for either the net rotation or total rotation. See below for more info on what is reported here.

A.8.4 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.

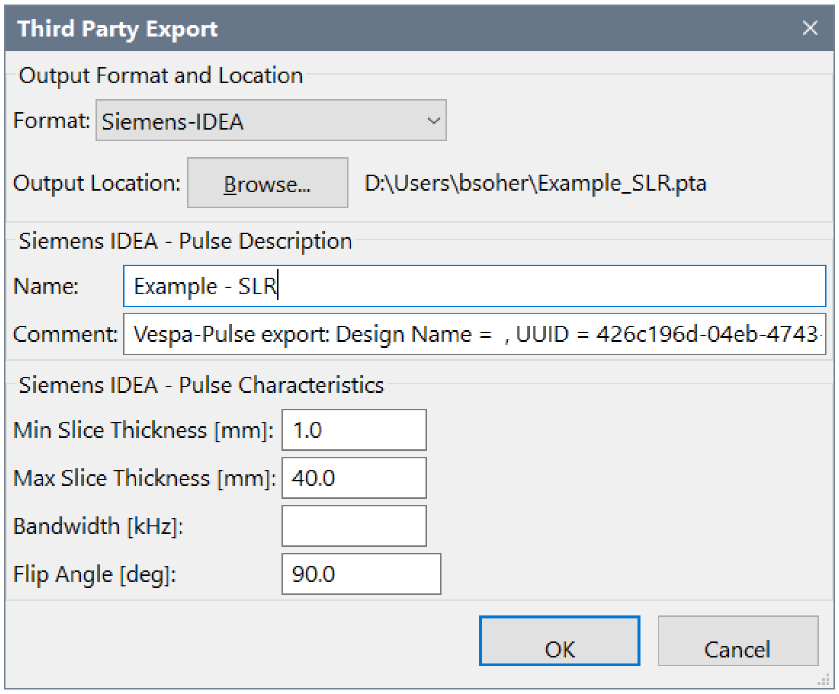
Note: For amplitude modulated pulses (such as an SLR pulse) the tip angle specified on the interpolation and rescaling transformation widget (the on-resonance tip) is the tip angle at the center of the profile. However, due to ripples in the passband, the tip angle at the profile center is usually at 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.

Appendix B. Third Party Export

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


Third Party Export
Output Format and Location
Browse...
Siemens IDEA - Pulse Description
Siemens IDEA - Pulse Characteristics
OK
Cancel

Format: Siemens-IDEA

Output Location: D:\Users\bsoher\Example_SLR.pta

Name: Example - SLR

Comment: Vespa-Pulse export: Design Name = , UUID = 426c196d-04eb-4743

Min Slice Thickness [mm]: 1.0

Max Slice Thickness [mm]: 40.0

Bandwidth [kHz]:

Flip Angle [deg]: 90.0

B.1 General Functionality

The third party export dialog acts on the pulse design 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.

B.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 Pulse application. The first figure in this section shows the Third Party Export dialog configured for the Siemens-IDEA format output.

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

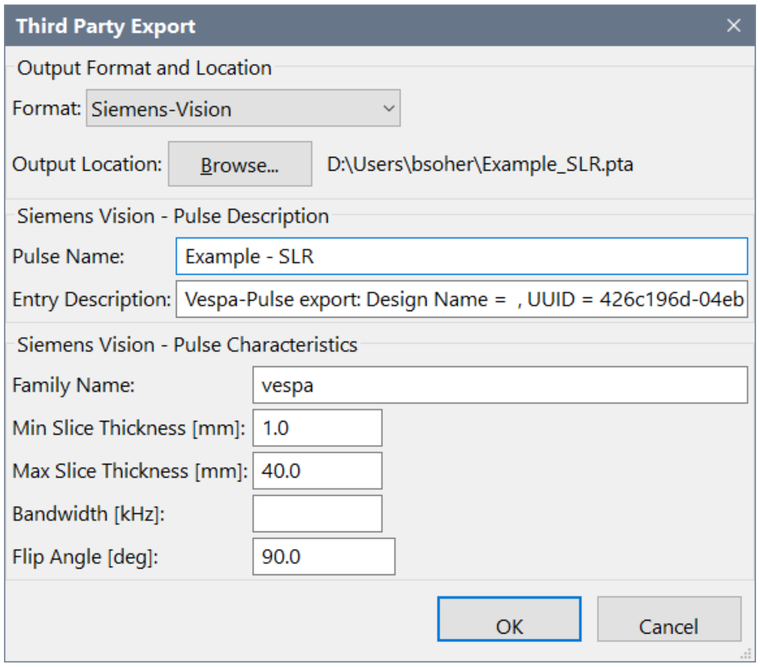
B.2.2 Example Siemens-IDEA Output File

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-Pulse 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)
...
```

B.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.



Output Format and Location

Format: Siemens-Vision

Output Location: Browse... D:\Users\bsoher\Example_SLR.pta

Siemens Vision - Pulse Description

Pulse Name: Example - SLR

Entry Description: Vespa-Pulse export: Design Name = , UUID = 426c196d-04eb

Siemens Vision - Pulse Characteristics

Family Name: vespa

Min Slice Thickness [mm]: 1.0

Max Slice Thickness [mm]: 40.0

Bandwidth [kHz]:

Flip Angle [deg]: 90.0

B.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 design 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 design 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 design 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-Pulse defaults this value to 1.0.

Max Slice Thickness	(float, [mm]) See Siemens Vision documentation for additional description of this parameter. Vespa-Pulse 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.

B.3.2 Example Siemens-Vision Output File

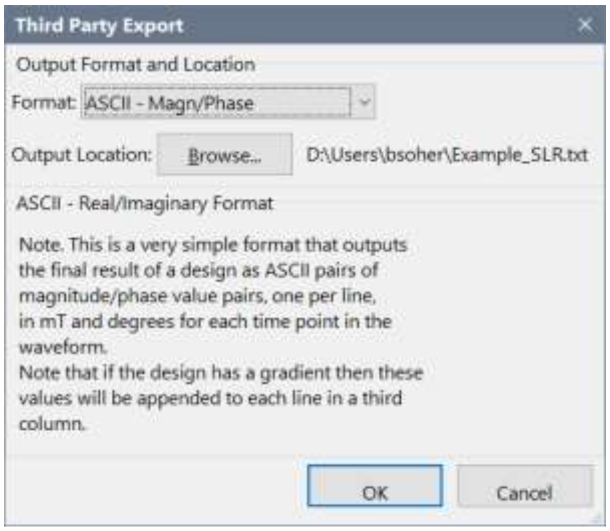
Here is a short example of the data output to a Siemens-Vision format output file. Note that the Entry_Description parameter is typically all on a single line but formatted here for easier reading. Also the waveform results are entered sequentially, one value per line, all magnitude values first followed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

```
Begin_Entry:   VESPA_Pulse_Generated_Pulse
Entry_Type:    6
Pulse_Name:    Simple_64pt_SLR
Entry_Description:     Exported from Vespa-Pulse 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
3.141592654
3.141592654
...
```

B.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). A single time point is written per line. 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. If the pulse design includes a gradient, these values will be appended as a third value on each line. Magnitude units are [mT], Phase units are [rad], Gradient units are [mT/m] (multiply by 0.1 to get Gauss/cm).

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



Format: ASCII - Magn/Phase

Output Location: D:\Users\bsoher\Example_SLR.txt

ASCII - Real/Imaginary Format

Note. This is a very simple format that outputs the final result of a design as ASCII pairs of magnitude/phase value pairs, one per line, in mT and degrees for each time point in the waveform.
Note that if the design has a gradient then these values will be appended to each line in a third column.

B.4.1 Format Specific GUI Fields

There are no format specific fields for this format.

B.4.2 Example ASCII – Magn/Phase Output File

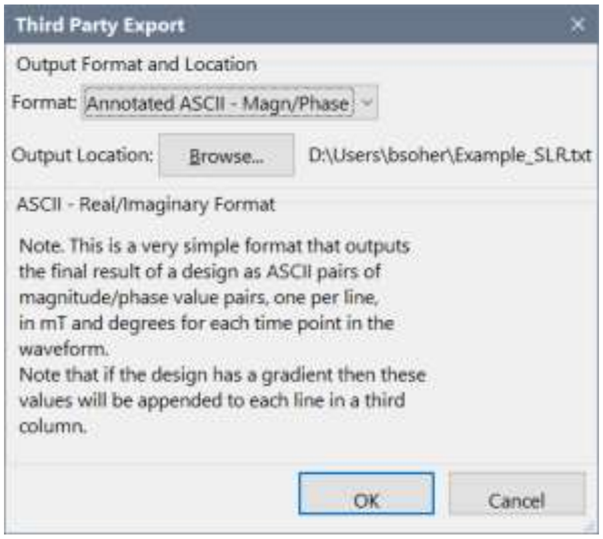
Here is a short example of the data output to an ASCII – Magn/phase format output file. Values are separated by a space.

```
0.026630458 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
…
```

B.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). A single time point is written per line. There is a brief text header in this format on lines starting with '#' symbols. This header information gives a brief description of the PulseDesign from which this waveform was exported. Note. If the pulse design includes a gradient, these values will be appended as a third value on each line. Magnitude units are [mT], Phase units are [rad], Gradient units are [mT/m] (multiply by 0.1 to get Gauss/cm).

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



Format: Annotated ASCII - Magn/Phase

Output Location: D:\Users\bsoher\Example_SLR.txt

ASCII - Real/Imaginary Format

Note. This is a very simple format that outputs the final result of a design as ASCII pairs of magnitude/phase value pairs, one per line, in mT and degrees for each time point in the waveform.
Note that if the design has a gradient then these values will be appended to each line in a third column.

B.5.1 Format Specific GUI Fields

There are no format specific fields for this format.

B.5.2 Example ASCII – Magn/Phase Output File

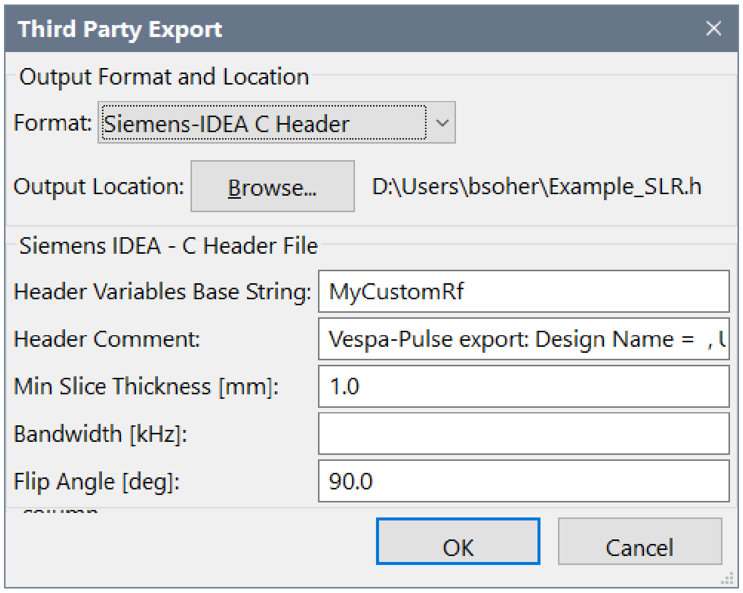
Here is a short example of the data output to an Annotated ASCII – Magn/phase format output file. Values are separated by a space. Note. If the pulse design includes a gradient, these values will be appended as a third value on each line.

```
# Exported from Vespa-Pulse
# 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
...
```

B.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.



Format: Siemens-IDEA C Header

Output Location: D:\Users\bsoher\Example_SLR.h

Header Variables Base String: MyCustomRf

Header Comment: Vespa-Pulse export: Design Name = , l

Min Slice Thickness [mm]: 1.0

Bandwidth [kHz]:

Flip Angle [deg]: 90.0

B.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.

B.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-Pulse 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 C. 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 designs 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.

C.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 design 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.

C.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.

C.2.1 Private and Public

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

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 design, 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).

C.2.2 In Use

Objects Affected: metabolites, pulse sequences, pulse designs

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.

C.2.3 Frozen

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

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 designs) 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

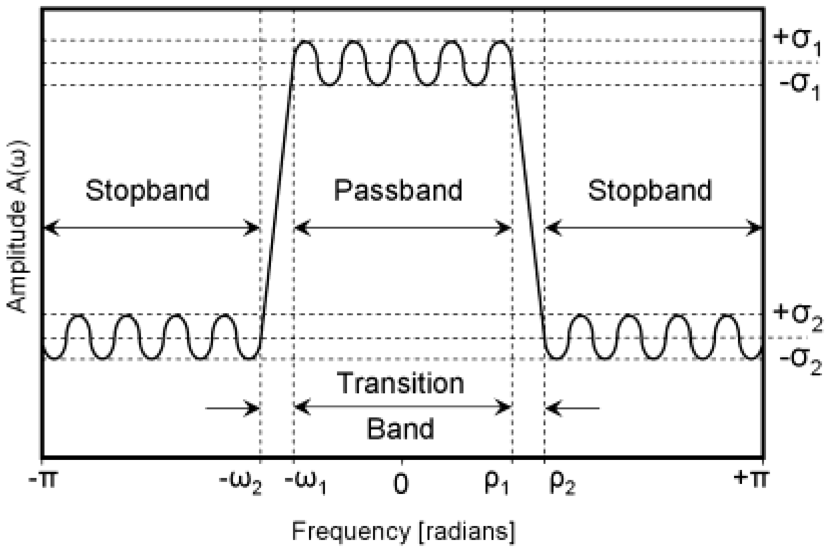
Note that no objects can refer to experiments, so experiments can never be 'in use'.

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 D. Case Studies in 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-Pulse 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. Pulse 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 Pulse 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 Pulse) 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. Pulse 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 Pulse can already be used to create Multiband pulse designs, future additions to Pulse 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. Pulse provides optimization methods to obtain such designs.

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, 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.

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

We structured the upgrade/conversion of RFPulse results into Pulse in this manner to ensure that we are still able to provide Vespa-Simulation PulseSequence objects that use real RF pulses with the waveforms that they expect. However, this method of upgrade means that we lose the information about how the pulse was created originally in RFPulse. 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.