**SIVIC Software Development Tutorial**

HMTRC Workshop - March 23-24, 2017

Department of Radiology and Biomedical Imaging, UCSF

Supported by NIBIB P41EB013598

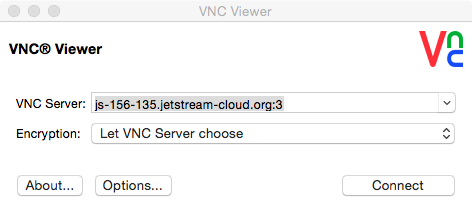
#### NSF Jetstream Development Environment Supported by XSEDE Educational Grant TG-SEE170001

**Goal:** The purpose of this tutorial is to introduce you the SIVIC C++ software framework (svk = SIVIC Kit). The tutorial will cover building the SIVIC package from source and extending the svk by developing a new kinetic model and using the model to fit an example data set.

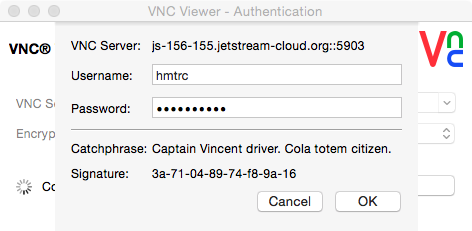
**Introduction:** The SIVIC C++ framework (svk) is an extensible software framework for developing MRS algorithms and applications. In this example you will use the framework to create a new kinetic model that can be run from the svk\_met\_kinetics command line tool in order to fit a simulated dynamic HP 13C data set.

**Setup:** The build environment for this tutorial has been preconfigured on XSEDE Linux VMs for this workshop so there is no need to set up your software environment. Simply use a VNC client on your laptop to log into the remote VM. Login information will be provided at the workshop.

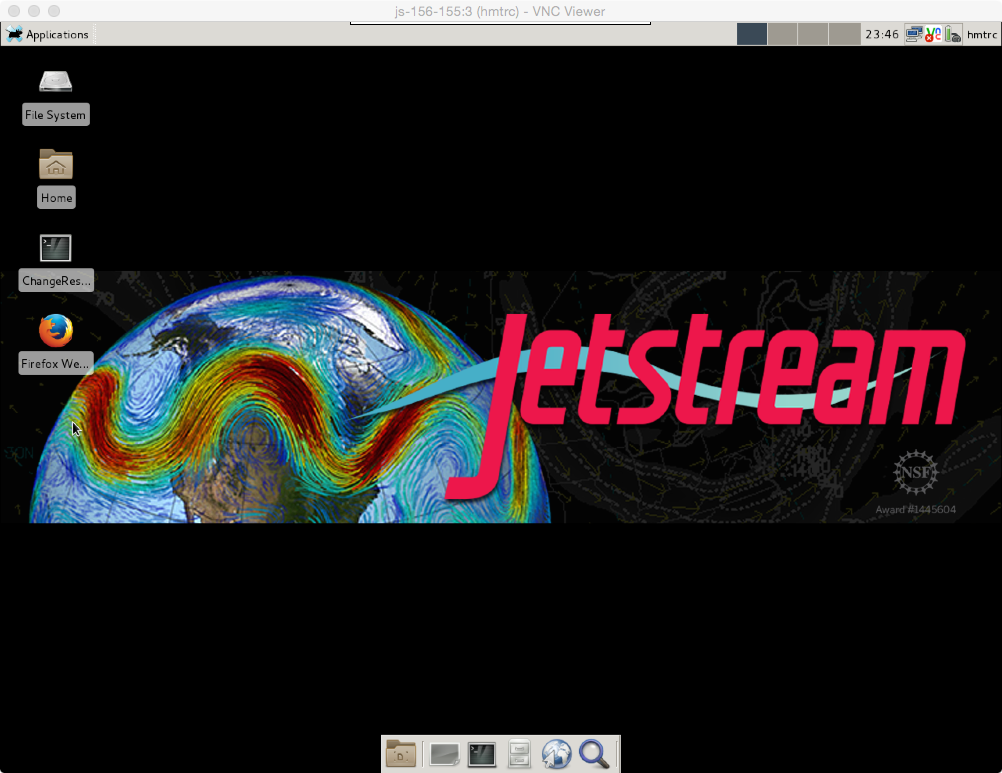
* Download and install VNC Viewer: <https://www.realvnc.com/download/viewer/>
* Open VNC Viewer and enter the URL of your Jetstream Linux VM in the “VNC Server” box., then click “Connect”. The URL and login info will be provided at the worksop.



* Enter the provided username and password:



* This will log you into the Linux desktop on the Jetstream VM:



* Click on the terminal icon at the bottom of the desktop to open a console (arrow above).
* Download sample data from the Jetstream desktop (enter the following commands in a terminal): <https://sourceforge.net/projects/sivic/files/sample_data/HMTRC_2017/development_tuturial.zip/download>

wget <https://sourceforge.net/projects/sivic/files/sample_data/HMTRC_2017/development_tuturial.zip/download>

unzip download

**Download and build SIVIC form source:** Get the sivic project from GITHUB and build the package.

* Make a work directory, cd into it and clone the “xsede” branch of the SIVIC GIT source code repository:

mkdir hmtrc

cd hmtrc

git clone -b xsede https://github.com/SIVICLab/sivic.git

* Build the software in a build directory (this will take about 1 minute):

mkdir build

cd build

make -f ../sivic/Makefile.ctest configure\_dist

make -j 10

* The binary applications are created under the build/applications directory. Run the svk\_met\_kinetics application to confirm that the build worked. You should see the help message from the application:

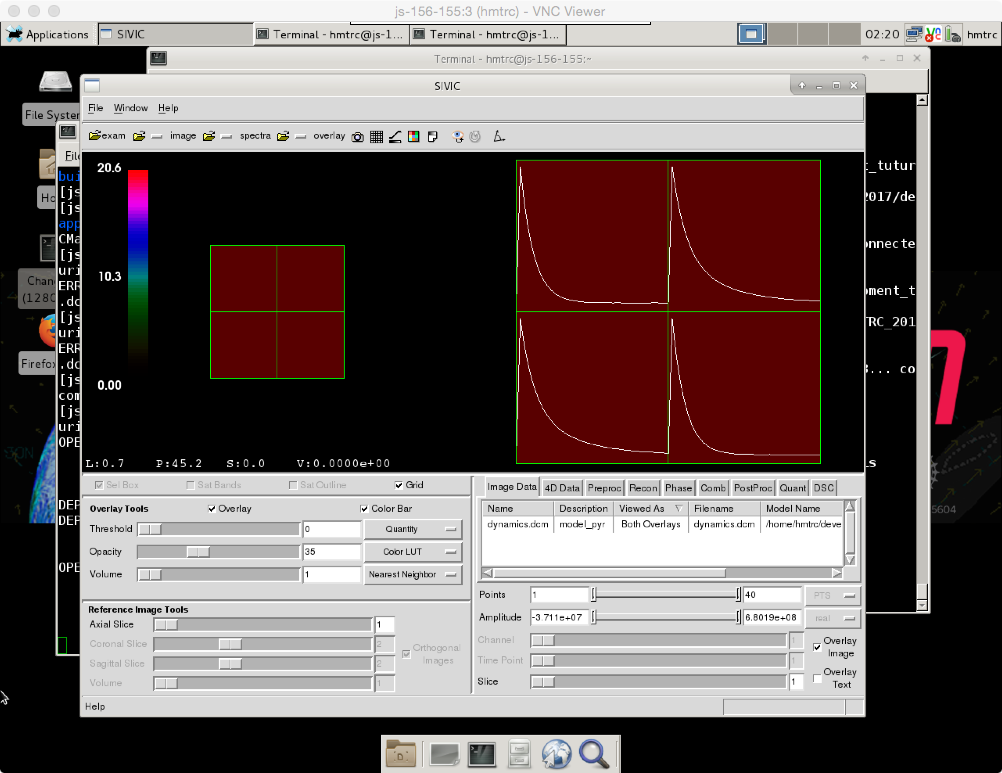
./applications/cmd\_line/Linux\_x86\_64/svk\_met\_kinetics

**Look at Simulated Dynamic Data:** The simulated data represents a single exponentially or bi-exponentially decaying signal in 4 voxels. We’ll create a new model representing bi-exponential decay and fit the decaying edge of the signals to the model:

The output will be a set of 3D parameter maps representing the modeled signal and fitted parameter maps.

* Open the simulated sample dynamic data (~/development\_tutorial/input/dynamics.dcm ) in the SIVIC GUI.

./applications/sivic\_app/Linux\_x86\_64/sivic --id ~/development\_tuturial/input/dynamics.dcm



**Create a new model:** Kinetic models are a subclass of svkKineticModelCostFunction. New models are defining a new model class and defining 6 virtual methods. Copy another model and modify:

virtual InitOutputDescriptionVector()

virtual InitNumberOfSignals()

virtual GetNumberOfParameters()

virtual InitParamBounds()

virtual InitParamScaleFactors()

virtual GetKineticModel()

* cd to the source code directory and make a copy of another kinetic model class:

cd ../sivic/libs/src

cp svk2SiteIMCostFunction.h svkBiexponentialCostFunction.h

* Open the class in an editor (vi, etc.) and modify. Note that a new copy fo the class is available with the downloaded sample data(~/development\_tuturial/src/svkBiexponential.h):

cd ../sivic/libs/src

cp svk2SiteIMCostFunction.h svkBiexponentialCostFunction.h

vi svkBiexponentialCostFunction.h

* + InitOutputDescriptionVector(): Set string descriptions of inputs/output
  + InitNumberOfSignals(): 1 input signal
  + GetNumberOfParameters(): 4 (tmax, k1, k2, fraction)
  + InitParamBounds(): find tmax and fix it, set ranges
  + InitParamScaleFactors(): scale by Tr to point space
  + GetKineticModel(): write the model function.

. . .

this->GetModelSignal(PYR)[t] = this->GetSignalAtTime(PYR, Tmax) \* (

fraction \* exp( -1 \* k1 \* (t-Tmax))

+ (1. - fraction) \* exp( -1 \* k2 \* (t-Tmax))

) ;

* Register the new model in the numerical fitting class (svkMRSKinetics.h). Add an entry in the ModelType enum ( BI\_EXPONENTIAL = 5):

vi svkMRSKinetics.h

- LAST\_MODEL = TWO\_SITE\_IM\_PYR

+ BI\_EXPONENTIAL,

+ LAST\_MODEL = BI\_EXPONENTIAL

* Add a case in svkMRSKinetics so that this model can get instantiated:

vi svkMRSKinetics.cc

#include <svk2SiteIMPyrCostFunction.h>

+#include <svkBiexponentialCostFunction.h>

} else if ( this->modelType == svkMRSKinetics::TWO\_SITE\_IM\_PYR) {

costFunction = svk2SiteIMPyrCostFunction::New();

+ } else if ( this->modelType == svkMRSKinetics::BI\_EXPONENTIAL) {

+ costFunction = svkBiexponentialCostFunction::New();

}

* Add an option for calling the new model in the svk\_met\_kinetics class:

cd ../../applications/cmd\_line/src

vi svk\_met\_kinetics.cc

usemsg += " 4 = 2 Site IM\_PYR \n";

+ usemsg += " 5 = Biexponential decay \n";

usemsg += " -t type Output data type: \n";

modelType = svkMRSKinetics::TWO\_SITE\_IM\_PYR;

+ } else if ( modelTypeInt == 5 ) {

+ modelType = svkMRSKinetics::BI\_EXPONENTIAL;

}

**Your Done! Now you can rebuild the svk\_met\_kinetics application and try it.**

**Rebuild the numerical fitting application with the new model and test it:**

* This time just rebuild the kinetic modeling application:

cd ~/hmtrc/build

make -j 10 svk\_met\_kinetics

* Run it and verify the new usage message:

./applications/cmd\_line/Linux\_x86\_64/svk\_met\_kinetics

* output should show the new –model option 5:

--model type Model to fit data to:

1 = 2 Site Exchange(default)

2 = 2 Site Exchange Perf

3 = 2 Site IM

4 = 2 Site IM\_PYR

5 = Biexponential fit

* Run it with the sample data:

cd ~/development\_tutorial

~/hmtrc/build/applications/cmd\_line/Linux\_x86\_64/svk\_met\_kinetics --i1 input/dynamics.dcm --model 5 -t 6 --tr 3 -o model5 --mask ./input/fitting\_mask.dcm

* Visualize the results. The display below is the fraction map (model5\_fraction.dcm) showing single exponential voxels (f ~1, red) and double exponential voxels (f ~.3, green):

~/hmtrc/build/applications/sivic\_app/Linux\_x86\_64/sivic

--id input/dynamics.dcm --id model5\_pyr\_fit.dcm

--id model5\_pyr\_residual.dcm -i ./input/fitting\_mask.dcm

-i model5\_Tmax.dcm -i model5\_k1.dcm -i model5\_k2.dcm

-i model5\_fraction.dcm -i model5\_rss.dcm

Color Overlay WL icon. Adjust color

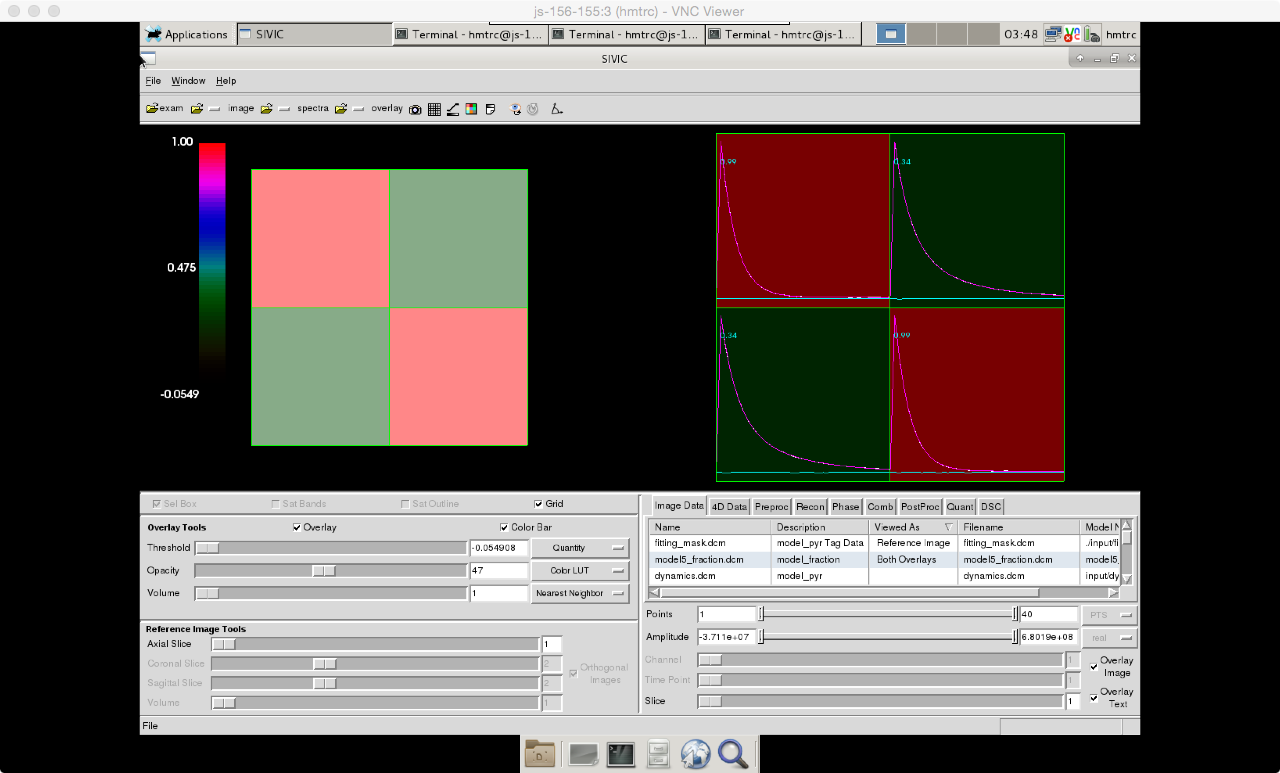
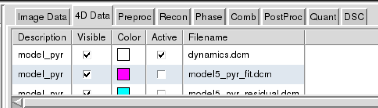


Image Data Tab: Select maps to display as overlays (right click, set as overlay)



4D Data Tab: Select dynamic traces to display on right.