Moving Real Data from MATLAB to GNU Radio

The purpose of this document is to explain, and demonstrate with an example, how to move real data from MATLAB to GNU Radio, say to be used as a source. We will verify that the procedure works by sending the signal to the speaker in GNU Radio.

Steps Required:

1. Create a signal in MATLAB, using the sampling frequency that you will be using in gnuradio.
2. In MATLAB, write the signal to a data file using the m-file: write_float_binary, given below.
3. If you need to move the data file to another computer or another directory: Look for the data file in the "Current Directory" of MATLAB.
    a. If it is visible, try a "copy" and "paste" to put the file where you want it.
    b. If it is not visible, go to the top menu bar of MATLAB, and select "File → Open." At the bottom of the dialog box, select "All Types" for "Files of Type", to make the data file visible. Then right-click on the desired file, and select "Send", to send it to a ram-stick (to be moved to another computer) or to another directory, to be used in gnuradio.
4. Verification: Send the signal to a sink in gnuradio.

Example: Say we want to create a sinusoidal signal, with amplitude .5 and frequency 300 Hz, in MATLAB, to be played as a tone in gnuradio with a sampling frequency of 48 K samples/sec.

Step 1: MATLAB code for creating the sinusoidal signal named *tone_300*, using a function in the form of an m-file named *tone*.

```
>> fs = 48*10^3;              %  set the sampling frequency
>> ts = 1/fs;                 %  set the time-between samples
>> [t, tone_300] = tone(ts, 300, 10, .5);        %  .5 cos(2*pi*300*t), for 10 s.
```

MATLAB code for the m-file: tone

```
function [t,x] = tone(ts, f, dur, amp)
```

```
%
% Generates a sinusoid, x(t), with
%     time-between-samples    = ts
%     frequency, in Hz        = f
%     duration                = dur
%     amplitude               = amp
%
t = [0: ts : dur-ts];
x = amp*cos(2*pi*f*t);
```

Step 2: MATLAB code for writing the signal to a file named *test_tone_300.dat*.

>> write_float_binary(tone_300, 'test_tone_300.dat')

MATLAB code for the m-file: write_float_binary

```
function v = write_float_binary (data, filename)

% Ref: gnuradio
%
% Usage: write_float_binary (data, filename)
%
%  open filename and write data to the file as 32 bit floats
%
  fid = fopen (filename, 'w');

  if (fid < 0)
    v = 0;
  else
    v = fwrite (fid, data, 'float');
    fclose (fid);
  end

end
```

Step 3. Follow the steps given above for step 3 to move the file named *test_tone_300.dat* to the desired computer or directory. (This step seems to be slightly different depending on your computer operating system and/or the version of MATLAB you are using.)

Step 4. Modify the dial_tone example from gnuradio to read a signal from the file in Step 3, and send it to the computer sound card, with the following steps:

a. save the dial_tone script under a new name;
b. replace the current source definition lines (src0 = …, src1 = …) with the line:

src = gr.file_source (gr.sizeof_float, 'test_tone_300.dat')

c. leave the current destination (dst = audio.sink(48000)) as it is;
d. replace the current connection lines (self.connect …) with the line:
   self.connect (src, dst)

Now execute the code, and you should hear the 300 Hz. Tone.

Further check: you can also modify the original dial_tone example so that the sound generated by gnuradio is a single sinusoid, at frequency 300 Hz. and amplitude 0.5. The sound should be the same as it was when the sinusoid was generated in MATLAB.

Note: If the signals don't sound the same, you are probably getting distortion due to the amplitude of the signals. Try decreasing the amplitude of the signals and make the comparison again.