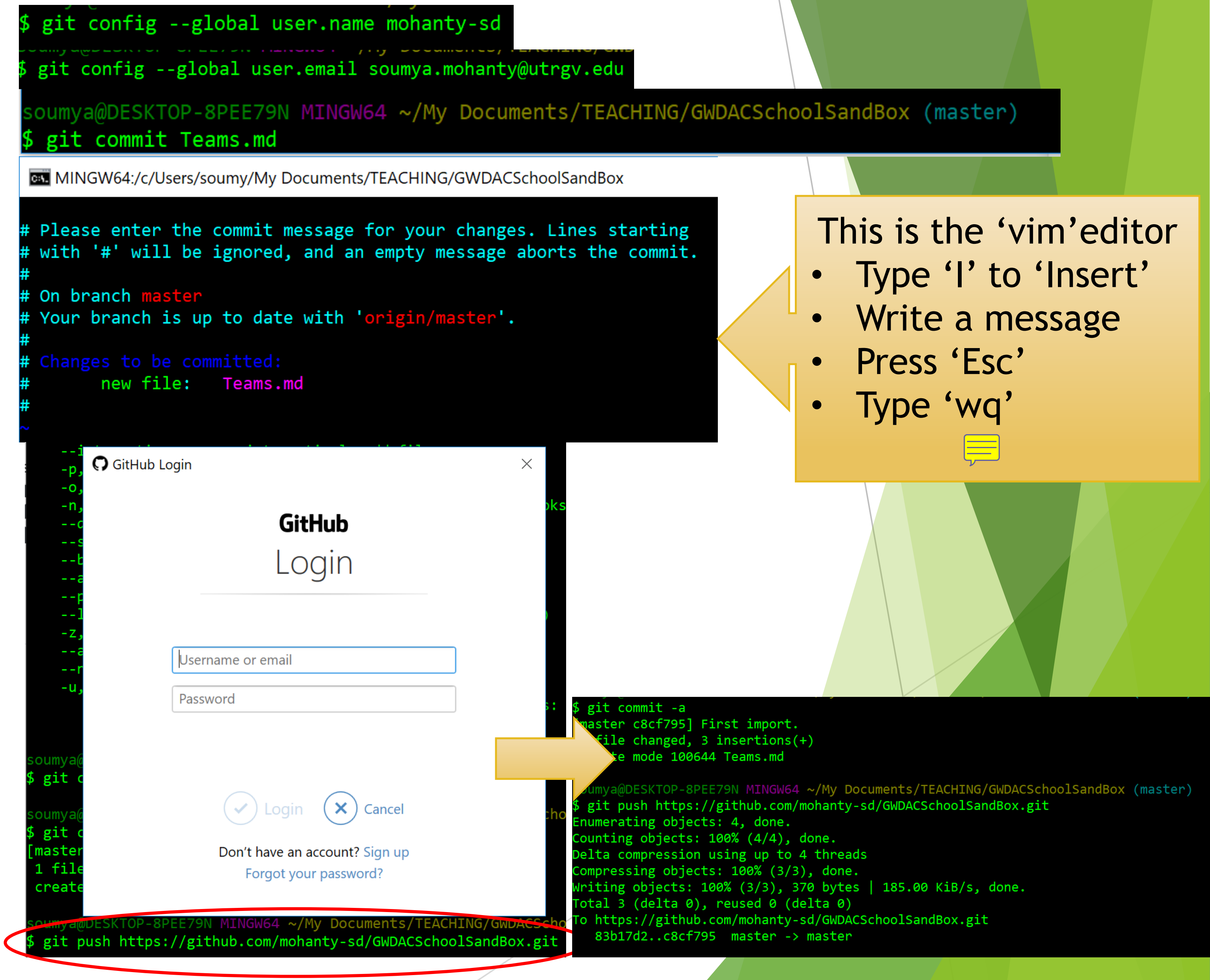
Topic 1: Lab

Intro lab cont…

- Got to your **GWDACSchoolSandBox** folder
- **Team leaders only**
  - **git pull**
  - Edit the file called Teams.md in your local branch and add the name of your team members
    - **(1) Group 1st, leader: Shucheng Yang ; members:**
  - **Confirm with instructor …**
  - Save the file and push your changes to the main repository as shown
  - **git commit → git push**
- **When instructed:** Everyone do **git pull**



Signal generation

- Write matlab code to generate different types of **discrete time signals**
- Each team will write code to generate one type of signal
- Team leaders will write the code and explain to their team the meaning of their code
- Team members can help:
  - If you know programming, try to implement the code in parallel so that there is a check
  - If you don’t know programming, copy the code and try to learn OR learn Matlab using the free mathworks.com coursework

Signal generation

- Follow the example of the code **GWDACSchoolSandBox/DSP/crcbgenqcsig.m**
  - Do **git pull** in **GWDACSchoolSandBox** to get the latest update
  - Write your code in the same format as this function
  - Learn elements of good coding: Good documentation, Clean and understandable code
  - Script showing how to use the function: **DSP/testcrcbgenqcsig.m**
- Once your code is running well:
  - Use: **git pull → git add→ git commit → git push**
  - **Remember the advice**: Pull before Push

OPTIMIZATION: NON-LINEAR MODEL

Quadratic chirp

$$f(t) = A\sin(2\pi\Phi(t))$$

Instantaneous phase:

$$\Phi(t) = a_1 t + a_2 t^2 + a_3 t^3$$

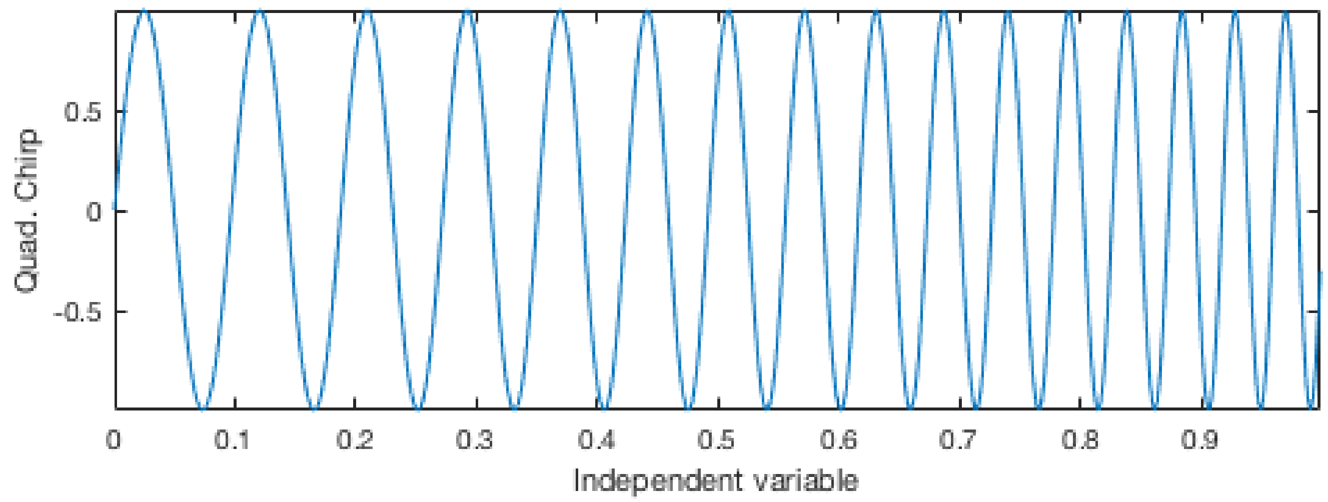
Parameters of the signal:

$$A, a_1, a_2, a_3$$

Instantaneous frequency:

$$f(t) = \frac{d\Phi}{dt} = a_1 + 2a_2 t + 3a_3 t^2$$

$f(t)$ increases with t
$1/f(t)$ (Instantaneous period) decreases with t



Example taken from textbook ("Swarm intelligence methods for Statistical Regression", Chapter 1)

Format of a Matlab function definition

function <output arguments> = <function name>(Input arguments)

```
function sigVec = crcbgenqcsig(dataX,snr,qcCoefs)
```

- dataX : vector of time stamps $(t_0, t_1, \ldots, t_{M-1})$ at which the samples of the signal $s(t)$ are to be computed.
- qcCoefs: vector of three coefficients [a1, a2, a3] that parametrize the phase of the signal $\Phi(t) = a_1 t + a_2 t^2 + a_3 t^3$
- snr: A special way to define the parameter A

$$\Phi(t) = a_1 t + a_2 t^2 + a_3 t^3$$

```
phaseVec = qcCoefs(1)*dataX + qcCoefs(2)*dataX.^2 + qcCoefs(3)*dataX.^3;
```

$$\sin(2\pi\Phi(t))$$

```
sigVec = sin(2*pi*phaseVec);
```

$$A\sin(2\pi\Phi(t))$$

```
sigVec = snr*sigVec/norm(sigVec);
```

Elements of good coding

```
function sigVec = crcbgenqcsig(dataX,snr,qcCoefs)
% Generate a quadratic chirp signal
% S = CRCBGENQSIG(X,SNR,C)
% Generates a quadratic chirp signal S. X is the vector of
% time stamps at which the samples of the signal are to be computed. SNR is
% the matched filtering signal-to-noise ratio of S and C is the vector of
% three coefficients [a1, a2, a3] that parametrize the phase of the signal:
% a1*t+a2*t^2+a3*t^3.

%Soumya D. Mohanty, May 2018

phaseVec = qcCoefs(1)*dataX + qcCoefs(2)*dataX.^2 + qcCoefs(3)*dataX.^3;
sigVec = sin(2*pi*phaseVec);
sigVec = snr*sigVec/norm(sigVec);
```

Function name should be descriptive but short: CRCBook-Generate-Quadratic-Chirp-Signal

First comment is used by Matlab to generate Contents report

Second line shows usage format (input and output arguments); Displayed with command "help crcbgenqcsig'

Describe what the code does and what is the meaning of each input and output argument

Author of the code (add additional lines for multiple authors), Date of creation

Variable names should be descriptive. C++ convention: thisIsAVariableName . Quadratic Chirp Coefficients

More signals

- Sinusoidal signal
  - $s(t) = A\sin(2\pi f_0 t + \phi_0)$
  - Parameters: A, f_0, ϕ_0
- Linear chirp signal
  - $s(t) = A\sin(2\pi(f_0 t + f_1 t^2) + \phi_0)$
  - Parameters: A, f_0, f_1, ϕ_0
- Sine-Gaussian signal
  - $s(t) = A\exp\left(-\frac{(t-t_0)^2}{2\sigma^2}\right)\sin(2\pi f_0 t + \phi_0)$
  - Parameters: $A, t_0, \sigma, f_0, \phi_0$

More signals

- Frequency modulated (FM) sinusoid
  - $s(t) = A\sin(2\pi f_0 t + b\cos(2\pi f_1 t))$
  - Parameters: A, b, f_0, f_1
- Amplitude modulated (AM) sinusoid
  - $s(t) = A\cos(2\pi f_1 t) \times \sin(f_0\, t + \phi_0)$
  - Parameters: A, f_0, f_1, ϕ_0
- AM-FM sinusoid
  - $s(t) = A\cos(2\pi f_1 t) \times \sin(2\pi f_0 t + b\cos(2\pi f_1 t))$
  - Parameters: A, b, f_0, f_1

Linear transient chirp signal

- $s(t) = \begin{cases} 0; \quad t \notin [t_a, t_a + L] \\ A\sin(2\pi(f_0(t - t_a) + f_1(t - t_a)^2) + \phi_0) \end{cases}$
- Parameters: $A, t_a, f_0, f_1, \phi_0, L$

Plots

- Make plots of each signal
- You have to choose a **sampling interval (or period)** Δ
$$t = n\Delta, \qquad n = 0,1,\dots,N-1$$
- **Sampling frequency** $= 1/\Delta$
- Generate the signal for this set of time stamps and make a plot

Choosing the sampling frequency: Nyquist Sampling theorem

- What is the bandwidth of your signal?
  - A good starting guess: highest **instantaneous frequency** in the signal
  - Note: Instantaneous frequency is not the same as Fourier frequency!
- Example:
  - N samples with sampling interval Δ
  - Quadratic chirp instantaneous frequency increases with time
  - $\Rightarrow$ Maximum instantaneous frequency is at $t = n\Delta$

$$f(t) = a_1 + 2a_2 t + 3a_3 t^2$$

- Nyquist theorem $\Rightarrow$ Sampling rate is $\geq 2 \times \text{Max. instantaneous frequency}$
- **Anti-aliasing:** When doing actual data analysis, we low pass filter our signals and data such that a given sampling frequency becomes the Nyquist frequency
  - Example: LIGO data is low pass filtered to a maximum Fourier frequency of 8192 Hz before it is sampled at 16384 Hz

FFT

- Assuming you are generating the signals with the proper sampling frequency, make plots of the periodogram of each signal
- **Periodogram**: Magnitude of the FFT

Frequencies in a DFT

- Generate the correct frequency values for your periodogram plots
  - Positive frequency components of FFT go from index number:
    - 1 to `floor(N/2)+1`
  - Negative frequency components go from index number:
    - `floor(N/2)+2 to N`
- Frequency spacing is $1/(N\Delta)$ where N is the number of samples and Δ is the sampling interval
- See **testcrcbgenqcsig.m** for an example

Advanced Lab Topic 1

Time frequency analysis

- Use Matlab's spectrogram function to make time-frequency plots of the signals that have been coded so far
- Each team should pick the signal function written by the next team (proceed in a ring)
  1. Read the signal generation function help (use "**help <functionName>**" in Matlab) and the associated **test<functionName>.m** script if needed
  2. If the help/test script are not well documented, inform the author of the function/script to make them better
     1. Authors of each function should add their names to the function file as shown in **DSP/crcbgenqcsig.m**
  3. Generate signal time series with appropriate Nyquist sampling frequency
  4. Make spectrograms: When successful, add spectrogram generation to **test<functionName>.m** script
- See **DSP/SpecgrmQCDemo.m** for an example
- **Highly recommended**: See the documentation of **spectrogram** in Matlab (start with "help spectrogram" and follow up with the hyperlink at the end of the help)

Filtering

- **Use the function for generating a sinusoid** to generate a signal containing the **sum of three sinusoids** with the following parameters
- Number of samples: 2048
- Sampling frequency: 1024 Hz
  - What is the maximum frequency of the discrete time sinusoid you can generate with this sampling frequency?

	Signal 1	Signal 2	Signal 3
A	10	5	2.5
f_0	100	200	300
ϕ_0	0	$\pi/6$	$\pi/4$

- Use Matlab's `fir1` function to design 3 **different** filters such that filter #i allows only signal #i to pass through
- Apply each filter to the signal and show the periodogram of the input and outputs (All teams do the same exercise; split filter design among team members)

Filtering

- The main thing about Matlab's filter design functions is that frequencies are specified relative to half the sampling frequency of the input data

  >> help fir1

  FIR filter design using the window method. B = fir1(N,Wn) designs an N'th order lowpass FIR digital filter and returns the filter coefficients in length N+1 vector B. The cut-off frequency Wn must be between 0 < Wn < 1.0, with 1.0 corresponding to half the sample rate.

- fir1 is the simplest FIR filter design method (fir2 and firls are more sophisticated methods): Good for designing standard low pass, high pass and bandpass filters.
- For the exercise, you have to design all of the above three types of filters with the appropriate frequency limits
- Study the script **DSP/LowPassFilterDemo.m** to see how a low pass filter is designed and how it is applied (to the quadratic chirp signal)