Designing a Relevant Lab for Introductory Signals and Systems

Edward A. Lee
UC Berkeley

A computer without networking, audio, video, or real-time services.

Objectives

- Introduce applications before the theory fully supports them.
- Connection between a mathematical (declarative) and a computational (imperative) view of systems.
- Use of software to perform operations that could not possibly be done by hand, operations on real signals such as sounds and images.

Technology

- Matlab
  - imperative programming language
  - finite signals (matrices and vectors)
  - discrete signals
- Simulink
  - block diagram language
  - infinite signals
  - continuous-time semantics

We view these as complementary.

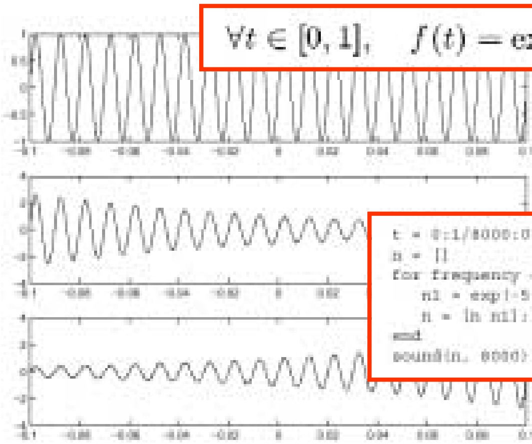
Organization

- 3 hour scheduled sessions, once a week
- 11 labs in 15 week session
  - 1 organizational, 1 technological, 2 review sessions
- In-lab section
  - takes about 1 hour, completed with signoff
- Independent section
  - takes 1-6 hours, completed with a report
- Tightly synchronized with the lectures.

Lab 1 (audio)

- Arrays and vectorization in Matlab
- Construct finite sound signals

$$\forall t \in [0, 1], \quad f(t) = \exp(-5t)\sin(2\pi \times 440t).$$

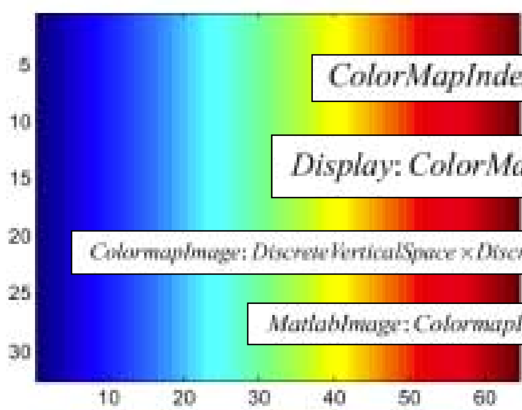


```
t = 0:1/8000:0.5;
n = [];
for frequency = [494 440 392 440 494 494 494]
   n1 = exp(-5*t).*sin(2*pi*frequency*t);
   n = [n n1];
end
sound(n, 8000);
```

Edward A. Lee

Lab 2 (images)

- Colormaps



$$ColorMapIndexes = \{0, \cdots, 255\}$$

$$Display: ColorMapIndexes \to Intensity^3$$

$$ColormapImage: DiscreteVerticalSpace \times DiscreteHorizontalSpace \to ColorMapIndexes.$$

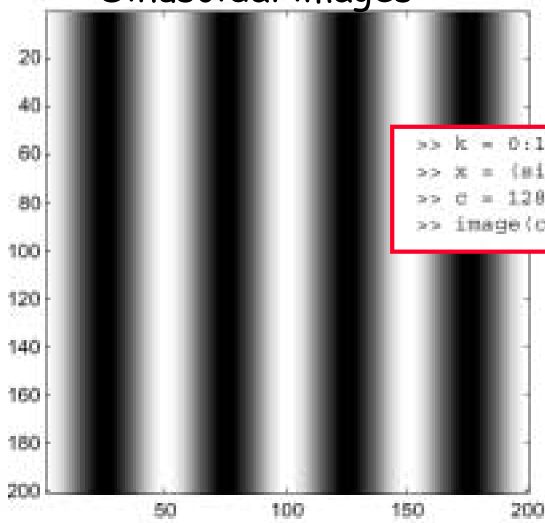
$$MatlabImage: ColormapImages \times Colormaps \to Images$$

Figure C.3: An image of the default colormap.

Edward A. Lee

Lab 2 (images)

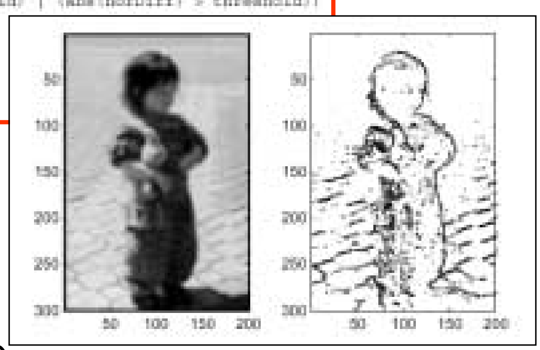
- Sinusoidal images




```
>> k = 0:199;
>> x = (sin(k*2*pi/50 + pi/2) +1);
>> c = 128 * repmat(x, 200, 1);
>> image(c), axis image
```


Lab 2 (images)

```
edges = ones(size(bwImage))*255;
threshold = 40;
for row = 2:300
   for col = 2:200
      vertDiff = bwImage(row, col) - bwImage(row - 1, col);
      horDiff = bwImage(row, col) - bwImage(row, col - 1);
      if ((abs(vertDiff) > threshold) | (abs(horDiff) > threshold))
         edges(row,col) = 0;
      end
   end
end
image(edges), axis image
```



- Images
  - blurring (averaging)
  - edge detection (first-order differences)

Lab 3 (state)

- State machines
  - Tamagotchi virtual pet

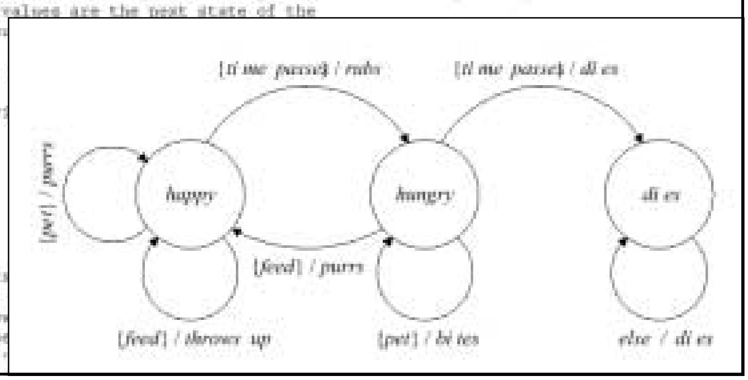
It starts out *happy*. If you *pet* it, it *purrs*. If you *feed* it, it *throws up*. If *time passes*, it gets *hungry* and *rubs* against your legs. If you feed it when it is hungry, it purrs and gets happy. If you pet it when it is hungry, it *bites* you. If time passes when it is hungry, it *dies*.

```
% PET - A fun
% The first a
% The second
% The two returned values are the next state of the
% pet, and the outp

function [newstate,

% The default behav
newstate = state;
out = 'absent';

switch(state)
case 'happy'
    switch(in)
    case 'pet'
        out = 'purr
    case 'feed'
        out = 'thro
    case 'time passe
        newstate = '
```



Lab4 (feedback control)

- Closed loop control of the virtual pet

Design a deterministic state machine that you can put into a feedback composition with your non-deterministic cat so that the cat is kept alive and time passes. Give the state transition diagram for your state machine and write a Matlab function that implements its *update* function. Write a Matlab program that implements the feedback composition.

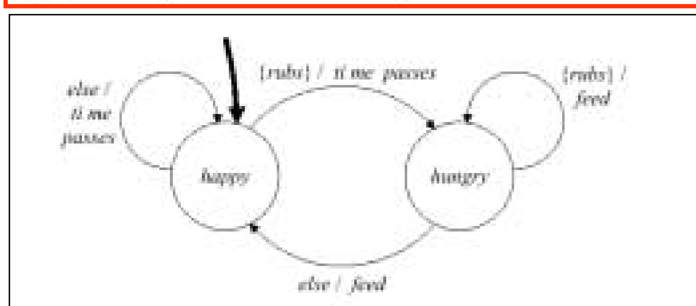
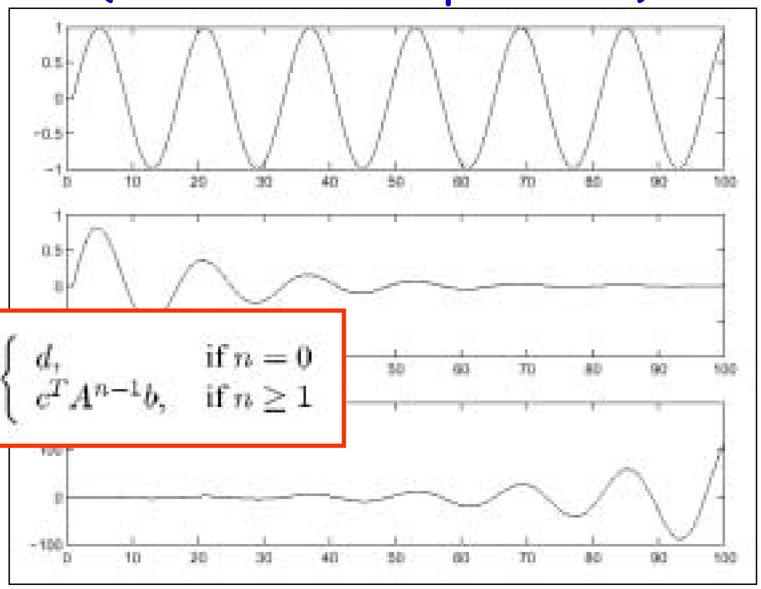


Figure C.9: Controller that keeps the nondeterministic cat alive.

Lab 5 (difference equations)



$$h(n) = \begin{cases} d, & \text{if } n = 0 \\ c^T A^{n-1} b, & \text{if } n \geq 1 \end{cases}$$

Lab 6 (differential equations)

The purpose of this lab is to experiment with models of continuous-time systems that are described as differential equations. The exercises aim to solidify state-space concepts while giving some experience with software that models continuous-time systems.

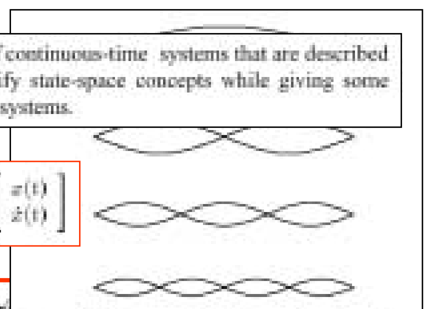


Figure C.5: Four modes of vibration of a guitar string.

$$\dot{z}(t) = \begin{bmatrix} \dot{x}(t) \\ \ddot{x}(t) \end{bmatrix} = \begin{bmatrix} a_{1,1} & a_{1,2} \\ a_{2,1} & a_{2,2} \end{bmatrix} \begin{bmatrix} x(t) \\ \dot{x}(t) \end{bmatrix}$$

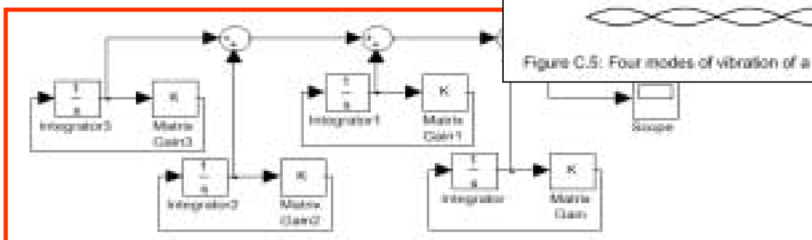
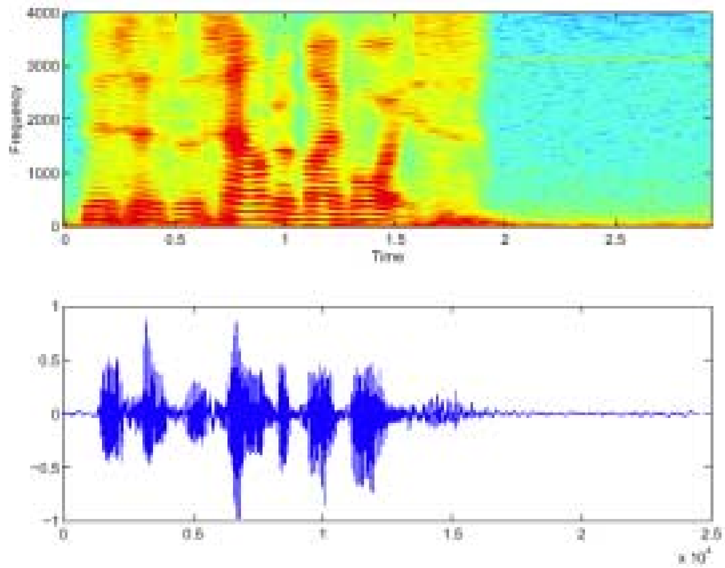


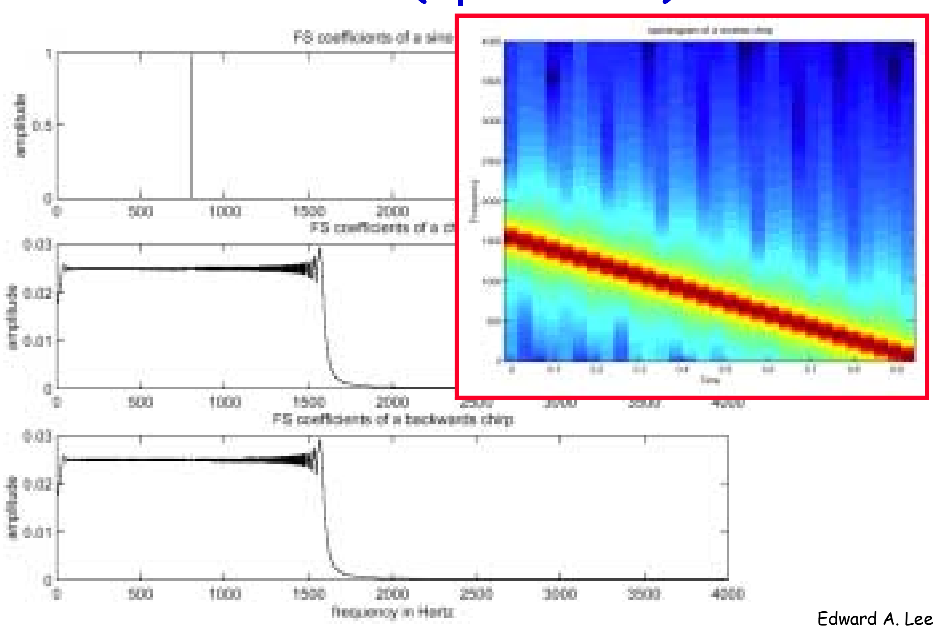
Figure C.12: A block diagram generating a plucked string sound with a fundmental and three harmonics.

Lab 7 (spectrum)



Edward A. Lee

Lab 7 (spectrum)



Edward A. Lee

Lab 8 (comb filters)

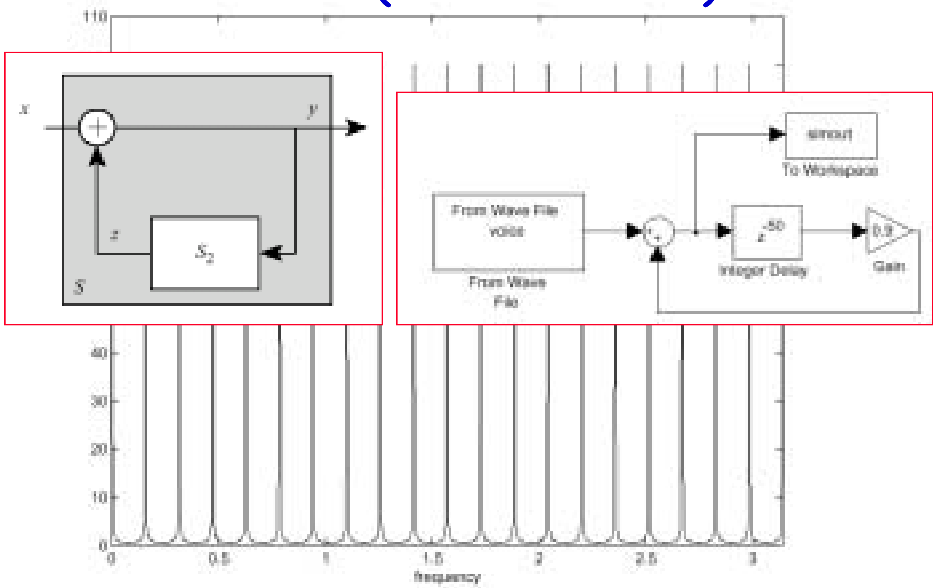
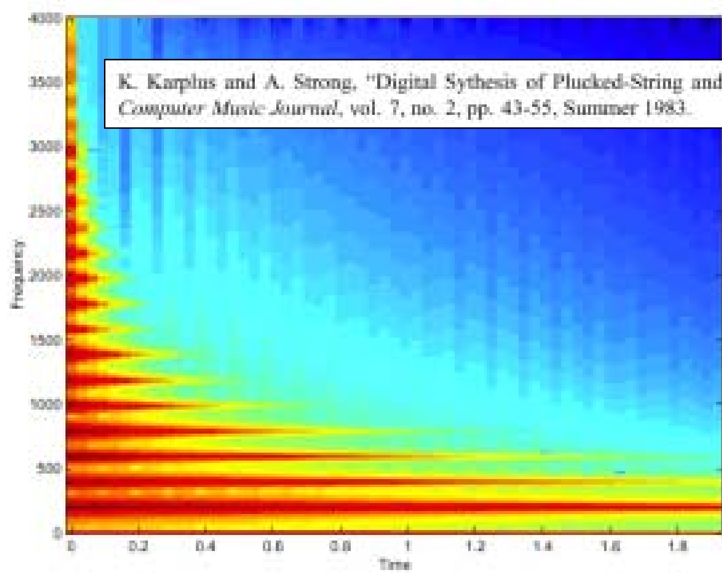


Figure C.25: Frequency response of the comb filter.

Edward A. Lee

Lab 9 (plucked string model)



K. Karplus and A. Strong, "Digital Sythesis of Plucked-String and Drum Timbres," *Computer Music Journal*, vol. 7, no. 2, pp. 43-55, Summer 1983.

Edward A. Lee

Lab 10 (modulation/demodulation)

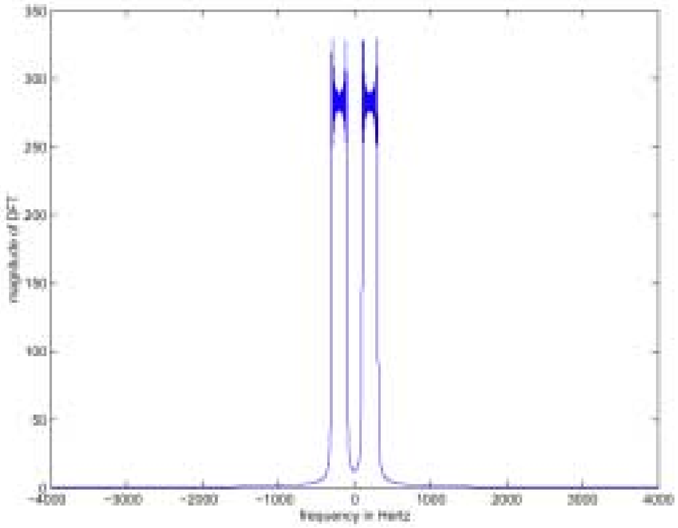


Figure C.28: Magnitude of the DFT of a chirp from 100 to 300 Hz, plotted from -4000 to 4000 Hz.

Lab 10 (modulation/demodulation)

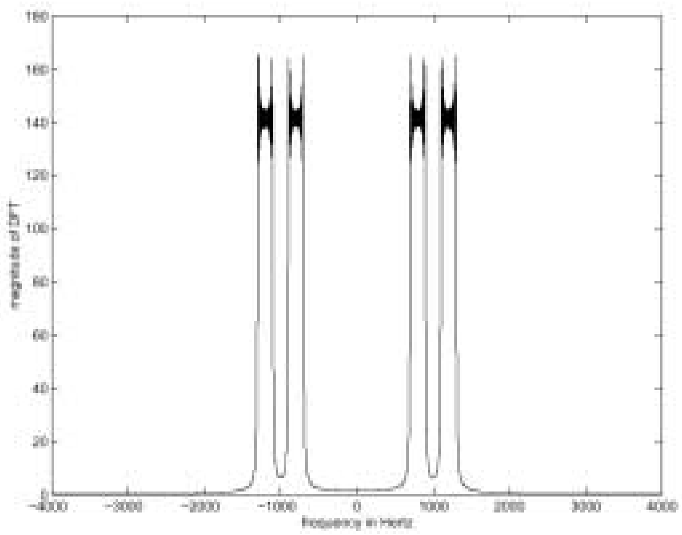


Figure C.29: Magnitude of the DFT of the chirp signal multiplied by a carrier at 1 kHz.

Lab 10 (modulation/demodulation)

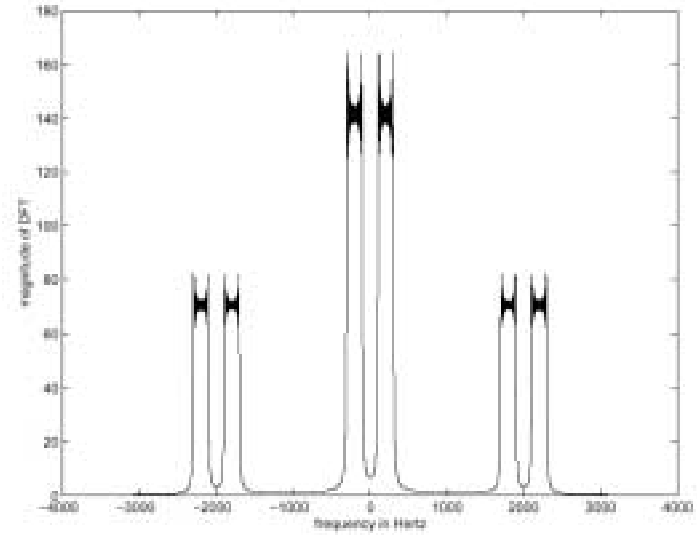


Figure C.30: Magnitude of the DFT of the chirp signal multiplied by a carrier at 1 kHz twice.

Lab 10 (modulation/demodulation)

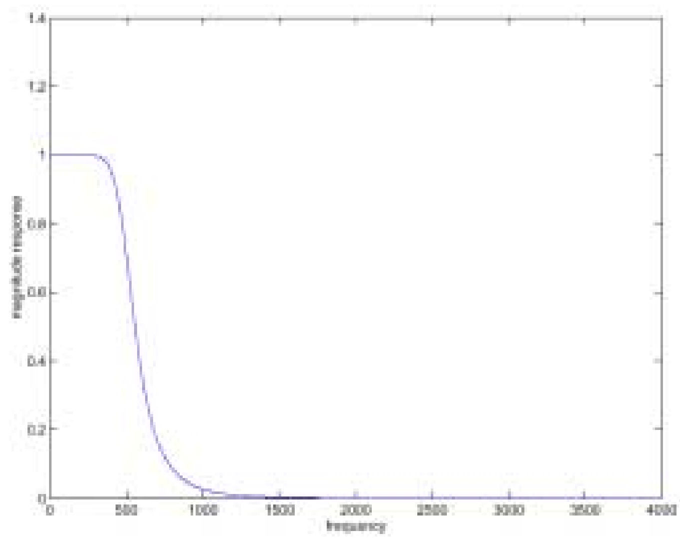
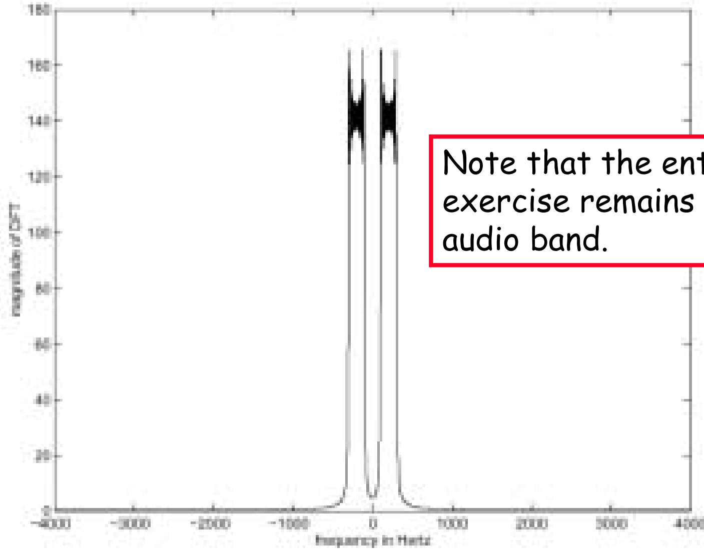


Figure C.31: Magnitude response of a 5-th order Butterworth filter with a cutoff frequency of 500 Hz.

Lab 10 (modulation/demodulation)



Note that the entire exercise remains in the audio band.

Figure C.32: Magnitude of a demodulated and filtered chirp.

Edward A. Lee

Lab 11 (sampling & aliasing)

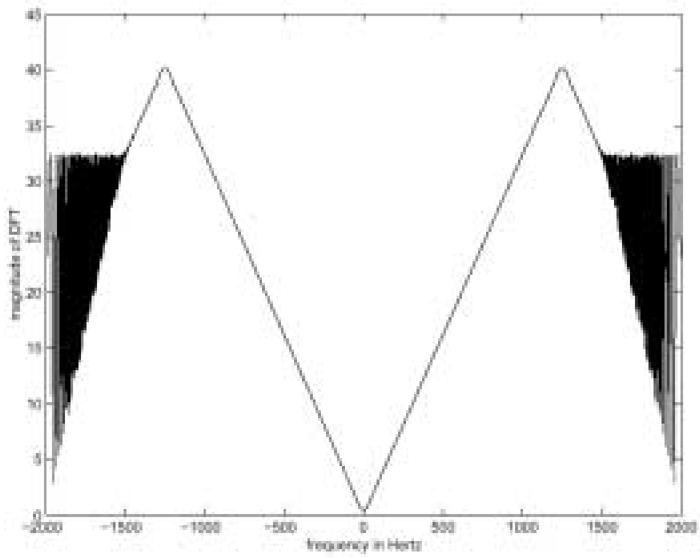


Figure C.35: Magnitude of the DFT of the downsampled chirp.

Edward A. Lee

Areas for Improvement

- Extensive capabilities of the tools can be intimidating.
- Requires some programming background.
- On-line help for Matlab is much better than for Simulink.
- Simulink's discrete-time models are continuous-time models in disguise.