Phys 129 lectures

28Apr22

Note to students

These are my raw, unedited lecture notes. They are intended to remind me what to say during lecture, so you are reading a very terse conversation between me and myself. You will not find good explanations here, though you will find some useful examples. I am providing the notes to you as an index to the course content, so that you can, for example, find where a particular topic is discussed in the recorded lectures. This will work most of the time, but there are places where the order of the material in the recordings is somewhat different than what you see here.

At the beginning of each lecture, you will see the names of the video files in which the corresponding content is covered. You can see the video file names on the course web page by moving the mouse cursor over the lecture links. The time at which a particular topic appears in a recorded lecture is sometimes shown in square brackets, for example like this: [12:46].

WARNING: The recorded lectures are from spring 2020, and some of these notes are from before then. While the content they contain is still useful, you must refer to the course web page for the current versions of due dates, assignment guidelines, and course rules and procedures. In addition, some of the programs and hardware discussed in the recordings have been replaced with newer versions.

If you print this file and bring it to class, you can take your own notes on the page next to mine.

1.

==========

Online:

unpack_rpi.mp4
rpi_install.mp4
passwords.mp4
homework_overview.mp4

==========

Introduction

textbooks

```
   attendance required
   lab times, 5223 hours

Install Raspbian
   assemble RPi
      use cover with opening for jumper wires
      insert SD card

   run through rpi_install.txt - stop before update script

shutting down safely

password security
   hash functions (md5, sha1, sha256)
   password hashing and shadow file (salting)

Preview web page

Homework guidelines on course web page
   tar

Homework #1 on course web page
   document password
   preview HW #1

keyboards, mice, monitors
   UCSB surplus


2.
========================
Online:
   course_info.mp4
   course_intro.mp4
   numbers_and_files.mp4
========================

Turn on networking
Run update script

Intro to 129L presentation

Late hw -10% 24 h, 0 after
grading                      summer with final exam
   attendance    10          10
   homework      65          45
   final project 25          45

Office hours: Tuesday 5:00-6:00

fixres
vm

<Ctrl-Alt> F1

Introduction to files
   xxd
      binary numbers
      hexadecimal numbers
      octal numbers
   ASCII
```

```
      cat
      less

   Python
      python3
      ipython3
      spyder3

      1 + 1
      0b11110000
      0xf0
      bin(240)
      hex(240)
      oct(240)
      0o10
################################################################################

3.
   ===============
   Online:
      theshell.mp4
   ===============

   ls (with options: -a, -l, -F)
   type -a
   man
      searching in man page
   help

   Directories
   Filesystem
      pwd
      cd
      mkdir
      rmdir

   Typing tricks
      tab completion
      <ESC>-.
      history, up and down arrows, !N
      .bash_history

   Shell aliases
      alias foo='echo "Hi there!"'
      .bashrc
         # begins shell comments
      type -a

   Shell variables, environment variables
      var='hello there'
      echo $var
      export FOO='this is foo'
      python3
         import os
         a = os.getenv('FOO')
         print(a)

      export var
      env

   Control characters - demonstrate with vi
```

```
   ^D - End of transmission 0x04
   ^C - Interrupt
   ^Z - Stop process
       jobs
       fg
   ^S - Pause terminal output (XOFF)
   ^Q - Resume terminal output (XON)

cat
   >  (file redirection)
   >> (append)
   concatenate using multiple files on command line
   cat file.txt | cat | cat | rev

Files
   file
   cp
   mv
   rm
   less


4.
   =================
   Online:
      more_shell.mp4
      flashdrive.mp4
   =================

   grep

   wc

   Pipes
      ls | grep

   Shell wildcards
      *
         ls *.txt
         ls /etc/*.conf
      ?
         ls .??*

   File extensions not so important in UNIX

   command substitution
      $()
      ``
      cat `ls *.txt`
      cat $(ls *.txt)

   Files, processes, kernel
      htop
      ps
      echo "Hi there!" > /dev/pts/0

   Standard input, output, error (file descriptors vs. files)
   cat
      stdin -> stdout
      takes stdin from command line arguments, more than one possible
      concatenate
```

```
    grep foo * 2>/dev/null

    Permissions
       id -a
       chmod
       sudo
       /etc/passwd
       /etc/shadow
       /etc/group
       /dev/snd/*

    PATH
       run proctemp script from ~
    CDPATH

    Text editor
       vi
          .vimrc
          :r in vi
          :r! in vi

    Format flash drives
       rsync

    Intro to shell scripts
       #!/bin/sh
       clear
       echo "Hello, world!"
################################################################################

5.
    ==============
    Online:
       python1.mp4
    ==============

    Machine code, assembly language, compiler, interpreter

    Python scripts
       #!/usr/bin/env python3

       #
       # hello.py - Say hello
       #
       # 29Apr19  Everett Lipman
       #

       print('Hello, world!')

    Comments in code

    6 ways to execute Python:
       interpreter, #!, command line, IDE,
       ipython, jupyter

    Implicit and explicit typecasting
       type()
       a = 5
       type(a)
       b = 5.5
       type(b)
```

```
   c = 'foo'
   type(c)
   d = '5.5'
   d
   x = float(d)
   x = float(c) -> error

print
   introduce print formatting
   a = 5.5
   print('%d' % a)
   print('%f' % a)
   print('%.3f' % a)

   scientific notation
   b = 5.5e10
   print('%.3f' % b)

   print('first: %f  second: %e' % (a,b))
   tuples
   c = (a,b)
   print('first: %f  second: %e' % c)

   print(..., end='')
   \n in strings
      print('Hello\n\nThere\n')

formatting need not be inside a print call
   s = 'The number is %.3f' % 3.1415

man 3 printf

dir()
   a = 5.5
   dir(a)
   dir(float)
   b = 5 + 2j
   dir(b)
   b.conjugate()

import
   sin(3)
   import math
   dir(math)
   math.sin(3)
   from math import *
   sin(3)
   PYTHONSTARTUP
      startup3.py
   np.sin(3) - will operate on arrays
   np.sin(np.arange(100))

Python help()
   help(np.arange)
   help(print)
   import math
   help(math.sin)
   help('modules')

PYTHONPATH
```

6.

```
==================
Online:
   programming1.mp4
==================

range(), lists, tuples
   iterables vs. lists
   list()
   a = list(range(10))
   b = tuple(range(10)
   a[3] = 100
   b[3] = 100
      tuples immutable

   range(1, 6, 1)
   a = np.arange(10)
   a
   b = np.arange(1,100,2)

   len()

for loops

while loops
   import time
   while True:
      print('still going...')
      time.sleep(0.5)

   i = 0
   while i < 5:
      i = i + 1
      print(i)

input
   instr = input('string, please: ')
   print(instr)
   type(instr)
   instr = input('number, please: ')
   conversion int(), float()
   a = int(instr)
   type(a)

Exit codes
   ls foo.txt
   echo $?

subprocess
   subproc_ls.py
   demonstrate in Python shell

Functions and methods [33:25]
   def add(a,b):
      c = a + b
      return(c)
   add strings
      operator overloading
      __add__() method
   docstring
```

```
      """, '''
      a = """This is a
             multi-line string."""
      a
      print(a)

Reading and writing files
   file_readlines.py
   help(open)
   careful_write.py
   open files for append
   with/as
      with open('lines.txt', 'r') as infile:
         lines = infile.readlines()
         print(lines)

      with open('out.txt', 'w') as outfile:
         outfile.write('Hello, file!\n')

Clarify files vs. file descriptors
   file is a place to send data to or get data from
   disk vs. virtual (in kernel)

Conditionals
   1 < 2
   1 > 2
   i == 6 vs i = 6

   for i in range(10):
      print(i)
      if i > 5:
         print('i > 5 !!!')
         break
      else:
         print('i <= 5')

i = -1
while i <= 5:
   i = i + 1
   print(i)

if 1 < 2 and True:
   print('hi')

if 1 < 2 or True:
   print('hi')

5 // 2 and 5 / 2

5 % 2

Python objects [1:14:25]
   strings
      string methods
      help(str)
      split() with and without argument
         a = 'Hello, there, this is a string.'
         a.split()
         a.split(',')
         a.split('ll')
   list methods
```

```
        a = list(range(6))
        dir(a)
        a.reverse()
        a.append()
    in
################################################################################

7.
    ==================
    Online:
       programming2.mp4
    ==================

    Integer overflow
       r = range(1,10000,2)
       list(r)
       b = 0
       for a in r:
          b = b + a*a
       b

       a = np.arange(1,10000,2)
       b = a*a
       b.sum()  # -837059544 on RPi

    a = np.arange(1, 100000, 2)
    type(a[0])  # <class 'numpy.int32'> on RPi
    a.dtype
    b = np.arange(1, 100000, 2, dtype='int64')
    type(b[0])  # <class 'numpy.int64'> on RPi
    a.sum()
    b.sum()

    Integer calculation with proper size [12:46]
       a = np.arange(1, 10000, 2, dtype='int64')
       b = a*a
       b.sum()  # 166666665000 on RPi

    Parallel calculation with numpy
       a = np.arange(1000000)
       b = np.sqrt(a)
       never use loops if numpy function can do the job

    Arbitrary precision integers
       21580943287502943875934 * 42549872943879274
          -> 918266394892314543987627518927329991916
       918266394892314543987627518927329991916 / 42549872943879274
       918266394892314543987627518927329991916 // 42549872943879274

    Floating point numbers
       binary representation
       mantissa, exponent
       IEEE 754
       float.hex()
          def md(x):
             print('%.55f' % x)

          a = 2.0
          a = 4.0
          a = 3.5
             md(a)
```

```
         a.hex()

      a = 0.7
         md(a)
         a.hex()
            '0x1.6666666666666p-1'
           = (1 + 6/16 + 6/256 + 6/4096 + ...)*2^{-1}
          ~= (1 + 0.375 + 0.0234375 + 0.00146484375) * 0.5
           = 0.69951171875

         a.as_integer_ratio()

   sys.float_info
   print (np.finfo(np.float))

   Representation error and roundoff error
      a = [0.1] * 10
      a
      sum(a)
      sum(a) == 1.0

List comprehensions
   [ i + i for i in range(10) ]

Bytes and character strings
   chr(65), chr(0x41), ord('A'), hex(ord('A'))
   a = b'Abytestring', list(a)
   b = [102, 111, 111]
   ''.join([chr(x) for x in b])

Names and objects
   immutable types
      integers
      tuples
   a = list(range(10))
   b = list(range(10))
   a
   b
   is keyword
      a is b
   a.reverse()
   a
   b
   c = b
   c is b
   b
   c.reverse()
   b
   d = c.copy()
   d.reverse()
   c

Scoping and local variables
   a = 1
   def add(x, y):
      a = x + y
      return(a)
   a
   def add(x, y):
      global a
      a = x + y
```

```
      return(a)
   a

Exceptions and error handling
   error_handling.py
   errprint.py
   try:
      1/0
   except Exception as foo:
      print(foo)
```

8.

```
==================
Online:
   programming3.mp4
==================

Slicing, zero first
   a = list(range(100))
   b = list(range(10,110,1))
   a[10:20]
   b[10:20]
   a[:50]
   a[50:]
   a[-10]
   a[-10:]
   a[10:20:2]
   a[::-1]
      [i:j:k] - if i omitted, 0 for k > 0, n-1 for k<0.
              - if j omitted, n for k > 0, -n-1 for k<0.
              - if k omitted, 1

   assignment with slicing
      a[30:40] = range(10)
      a[40:60] = 5 -> error
         a[40:60] = [5]*20
         [5]*20
         a
         a[40:60] = [5]*10
         a
         len a

      a = np.arange(100)
      a[40:60] = 5
      a

Dictionaries [11:19]
   person = {}
   person['color'] = 'blue'
   person['age'] = 47
   person['height'] = 1.776
   person['food'] = 'pizza'
   person['car'] = 'Toyota'

keys() method

sorted(iterable)
   iterable may be dict
   a = list('hi there')
   sorted(a)
```

```
    List sort() method
       a = list('hi there')
       a.sort()
       a

    None
       months list
       months dictionary keyed on numbers

    Debugging with print()
       reportloop.py

    ps, htop, kill
       kill cat in another terminal
       kill -9 bash
       kill -STOP cat
       man 7 signal
       echo $$

    Symbolic links
       demonstrate ln -s
       mv, rm a symlink
       /usr/bin/python3
       absolute and relative symbolic links

    Optimization [55:05]
       don't do more computation than necessary
          polynomial evaluation
             a + bx + cx^2 + dx^3 : 3 additions, 6 multiplications
             a + x(b + x(c + xd)) : 3 additions, 3 multiplications
       optimize inner loops
       use more memory
          swapping two elements in a list vs. creating a swapped list
          lookup tables
       avoid function calls
       recursion - not good in Python

    time shell builtin
       time -p sleep 2

    time module [1:12:42]
       time.perf_counter()
       timeloop.py
       pass
##############################################################################

9.
    ================
    Online:
       graphics1.mp4
    ================

    prime.c and prime.py

    prime.py optimization
       roundoff - // vs. / with 86795643218674325
          a = 86795643218674325.0
          a.as_integer_ratio()
       optimize inner loop by removing else: - 10% improvement
```

```
Pythonic factorization
   a = 1232324123
   np.sqrt(a)
   b = np.arange(35105) + 2
   b
   c = a % b
   b[c == 0]
   a/_
   23957*51439

apt-get and aptitude
   sudo aptitude install sysvbanner

argc, argv
   arguments.py

Operator precedence
   1+1*3
   6/2*3
   https://docs.python.org/3.9/reference/expressions.html

Numpy arrays
   np.arange()
   np.linspace()
   a = np.loadtxt('wind.dat')
   type(a)
   dir(a)
   help(a)
   a.dtype
   a.shape
   type(a[0])
   a[0].shape
   a[0][0]
   a[0][2]
   type(a[0][0])
   a.T
      different view of a, not a new object
      b = a.T
      c = a.copy().T
      a[1][1] = 99
      a
      b
      c
   a = np.zeros(100, dtype='complex')

Plotting [59:08]
   drows = np.loadtxt('wind.dat')
   wdat = drows.T

   f1, ax1 = plt.subplots()
   ax1.plot(wdat[0], wdat[1], 'o')
   ax1.set_xlim(-5,25)
   ax1.set_ylim(0,12)
   ax1.errorbar(wdat[0],wdat[1],yerr=wdat[2],fmt='o', capsize=3)
   # ax1.plot(wdat[0], wdat[1], 'o')
   f1.show()

   f2, ax2 = plt.subplots()
   ax2.plot( np.sin(np.linspace(0,np.pi,1000)) )
   f2.show()
   ax2.plot( np.cos(np.linspace(0,np.pi,1000)) )
```

```
      f2.show()

   simple_plot.py
      demonstrate saving .eps files [1:10:07]
      view .eps file with gv [1:11:29]


10.
   ================
   Online:
      graphics2.mp4
   ================

   Summer only:
   Project
      guidelines
      https://www.adafruit.com/category/105
      Use of packaged equation solvers, ray tracers, etc. in projects
      ===================
      Online: project.mp4
      ===================

   Graphics [00:18]
      pixels, screen memory, 24-bit color
      xmag demonstration

      X = 100
      Y = 100
      f1, ax1 = plt.subplots()

      pvals = np.zeros((X, Y, 3), dtype='uint8')
      pvals[50,50,:] = (255,0,0)

      ax1.imshow(pvals, interpolation='none')
      f1.show()

      for i in range(16,64,1):
         for j in range(16,64,1):
            pvals[i,j,:] = (0,0xee,0)

      ax1.imshow(pvals, interpolation='none')
      f1.show()

      set color block with slicing instead of loop
         pvals[20:40, 20:40, :] = (0,0,0xff)
         ax1.imshow(pvals, interpolation='none')
         f1.show()

   img.py [22:13]
   cmapimg.py [39:16]

   Complex numbers [46:22]
      a = 5+2j
      type(a)
      a = 5+j does not work
      b = complex(5)
      b
      a = 5+1j
      a*a
      a+a
      5+1j * 5+1j  (unexpected result)
```

```
      (5+1j)*(5+1j)

   Mandelbrot and Julia sets [50:58]

   Fifos (queues)
      from collections import deque
      a = [1,2,3]
      q = deque(a)
      q.append()
      q.popleft()

   Stacks [1:00:45]
      append(), pop()
      dc

   PostScript [1:06:50]
      gs
      hello, world
         stack-based syntax
         coordinate system
      gv
      rt345.eps [1:13:23]
      Blue Book and PLRM

      vector graphics
         curveto.ps
         bigletter.ps

      EPS
         rt345.eps
         PS translate and scale [1:35:25]
         gv coordinate display

      petal.ps [1:42:10]
##############################################################################

11.
   ==================
   Online:
      networking1.mp4
   ==================

   School year only:
   Project
      guidelines
      https://www.adafruit.com/category/105
      Use of packaged equation solvers, ray tracers, etc. in projects
      ====================
      Online: project.mp4
      ====================

   Network protocol stack
      application  email, http
      transport    TCP, UDP
      network      IP
      link         ethernet
      physical     cable, radio wave, optical fiber, etc.

      ifconfig
         ip command Linux-specific
         netstat -ie
```

```
   IP
      IP numbers
      port numbers
      /etc/services
   UDP
   TCP

Basics of DNS
   host

traceroute
ping

URLs

nc -Cv web.physics.ucsb.edu 80
   GET /~phys129/lipman/ HTTP/1.0

CRLF in HTTP and SMTP

wget
   update script function


12.
   ==================
   Online:
      networking2.mp4
   ==================

   nc -Cv elo.physics.ucsb.edu 25
      HELO ealrpi1
      MAIL FROM:mrsmtp@example.com
      RCPT TO:lipm1@elo.physics.ucsb.edu
      DATA
      Subject: fake mail

      This is fake mail
      .
      QUIT

   RFCs
      http://www.rfc-editor.org/rfc-index.html
      SMTP 821
      HTTP 2068
         RFC 7230, section 3.1.1 "Request Line"

   nc -lv 1024 < file
   nc 127.0.0.1 1024

   netstat --inet
   lsof -i

   Buffer overflows and stack smashing

   client.py
      also can write to socket

   server.py
      also can read from socket
```

```
    Python Requests library and Beautiful Soup
       http://docs.python-requests.org/en/master/
       https://www.crummy.com/software/BeautifulSoup/bs4/doc/

       import requests
       r = requests.get('http://web.physics.ucsb.edu/~phys129/lipman/')
       type(r)
       dir(r)
       r.content
       r.encoding
       r.text
       print(r.text)

    BRING HARDWARE TO NEXT LECTURE
################################################################################

13.
    ======================
    Online:
       data_acquisition.mp4
    ======================

    --> Air Can

    School year only:
       Detailed project scope due to me

    generators
       def gen():
          yield 1
          yield 2
          yield 3
       a = gen()
       a
       next(a)
       next(a)
       next(a)
       next(a)

    Object-oriented programming vs. imperative programming

    stripchart.py
       classes

    I2C bus

    MCP9808
       https://www.adafruit.com/product/1782
       Wiring diagram on course web page
       tempdemo.py

    ADS1015 and solar cell
       adcdemo.py
       fastadc.py
          Real-time kernels


14.
    ==============
    Online:
```

```
      fourier.mp4
    ==============

    Sampling
      Nyquist theorem
      aliasing
      convolution and signal recovery from samples
        (f*g)(t) = \int_{-\infty}^\infty f(\tau)g(t - \tau) d\tau

    /etc/init.d, /etc/init
      man service

    FFT
      evaluate polynomial at n complex nth roots of 1
      O( n^2 ) -> O[ nlog(n) ]

    matplotlib.mlab.psd() and numpy.fft
      fft_spectrum.py
      psd_spectrum.py
################################################################################

15.
    =============
    Online:
      random.mp4
    =============

    man random
      RNG period
    dir(np.random)
    help(np.random)
    np.random.random()
      np.random.random(100)
      np.random.random((10,10))
    np.random.seed()
    np.random.randint()
    np.random.uniform()

    Monte Carlo
      area/value of pi
      simple integration

      For N evaluations, error goes as 1/sqrt(N)
      for any number of dimensions.  For grid
      (trapezoidal rule), error goes as 1/N^{2/d}
      in d dimensions.

    LaTeX
      latex texample.tex
      latex texample.tex
      dvips texample.dvi
      ps2pdf texample.ps


16.
    ========================
    Online:
      fork_exec_tracing.mp4
    ========================

    fork() and threading
```

```
      fork_example.py

      starting a new process
         help exec
         sleep 3
         exec sleep 3
         man exec, execve
         shell forks, then calls wait(2)

         import os
         help(os.execv)
         -> in second terminal, ps -uxw | tail -5
         os.execv('/bin/sleep', ['/bin/sleep', '10'])
         -> ps -uxw | tail -5; show that process name changed

      import threading
      dir(threading)

      thread_example.py

   Homework problems
      fork_ls.py demo
      threadchart.py demo

   System call tracing
      man strace
      strace ls
         trace output goes to stderr
      strace ls nosuchfile 2> foo
      trp
         run on date
         run on errprint.py
         run on fork_example.py
################################################################################

17.
   ====================
   Online:
      distributions1.mp4
   ====================

   Coin toss/binomial mean and SD
      P(N,k) = \frac{N!}{k!(N-k)!} p^k q^(N-k)
      mean = Np
      \sigma = \sqrt(Npq)

   Binomial sampling example
      Dice game: win $5 on 6, lose $1 on any other roll
      you play:
      N = 300
      w = 50
      p = 1/6, uncertainty?
      Npq = 300*(1/6)*(5/6) = 41.667
      \sigma = 6.45
      w = 50 +/- 6.45
      p = 0.167 +/- 0.022
      ----------------------------
      cousin plays:
      N = 100
      w = 31, fair?
      Np = 16.67, sqrt(Npq) = 3.73
```

```
   (31 - 16.67)/3.73 = 3.8 sd
   Both tails: 1/6900
   Probability of w = 31 or greater is 1/13,800
   ----------------------------
   Best estimate for cousin:
   N = 100, w = 31
   p = w/N = 0.31
   sqrt(Npq) = 4.6, 4.6/100 = 0.046
   p = 0.31 +/- 0.046
   ----------------------------
   Other examples
      election polling
      count fish in lake
      all error bars touch line

Poisson distribution
   Binomial as Np -> 0, q -> 1
   P(n) = (\mu^n/n!)e^{-\mu}
   \sigma = \sqrt(Np) (q \approx 1)
   histogram fluctuation for uniform random numbers
      a = np.random.random(10000)
      f1, ax1 = plt.subplots()
      ahist = ax1.hist(a, 10)
      f1.show()
      compare np.sqrt(Npq) to np.sqrt(Np)

      f2, ax2 = plt.subplots()
      ahist = ax2.hist(a, 100)
      f2.show()
      compare np.sqrt(Npq) to np.sqrt(Np)

Nonuniform random numbers
   Gaussian distribution
   1/sigma*sqrt(2\pi) exp(-(x-mu)^2/2\sigma^2)

   a = np.random.random(10000)
   f1, ax1 = plt.subplots()
   ahist = ax1.hist(a,50)
   f1.show()

   b = np.random.normal(1.0, 0.1, 10000)
   f2, ax2 = plt.subplots()
   bhist = ax2.hist(b,50,(0.6,1.4))
   f2.show()

   from scipy import stats
   x = np.linspace(0.6, 1.4, 1000)
   ax2.plot(x, 160*stats.norm.pdf(x, 1.0, 0.1))
   f2.show()


18.
   ====================
   Online:
      integration.mp4
      distributions2.mp4
   ====================

   Discrete integration
      careful with dx
      find the integral of y = x from 0 to 1:
```

```
        N = 10000
        dx = 1.0/N
        x = np.linspace(0, 1.0-dx, N)
        x
        y = x
        ydx = y*dx
        ydx.sum()
        y.sum()*dx

     trapezoidal rule
     simpson's rule (parabola)
     other methods

     numint.py

  For the error function, \sigma = 1/\sqrt(2).
  So we integrate one tail by giving the number
  of standard deviations divided by \sqrt(2)
  as the argument to erfc().  With N = 100,
  p = 1/6, w = 31, we are 3.846\sigma from
  the mean, so:

  from scipy import special
  0.5*special.erfc( 3.846/np.sqrt(2) )
     6.0030881107545723e-05 = 1 in 16658.

  from scipy import stats
  x = np.arange(101)
  a = stats.binom.pmf(x, 100, 1.0/6.0)
  a[31:].sum()
     0.00029578488275368661 = 1 in 3381.
  f, ax = plt.subplots()
  ax.plot(a, drawstyle='steps-post')
  f.show()

  Integer histograms
     a = np.random.randint(1,7,1000)
     f, ax = plt.subplots()
     ax.hist(a)
     f.show()

     inthist.py

  Arbitrary nonuniform random numbers demo
     g(y) = 2y, f(x)dx = dx = g(y)dy = 2ydy (2y is normalized on 0-1)
     dx = 2ydy
     x = y^2, y = sqrt(x)

     x = np.random.random(10000)
     f1, ax1 = plt.subplots()
     xhist = ax1.hist(x,50)
     f1.show()

     y = np.sqrt(x)
     f2, ax2 = plt.subplots()
     yhist = ax2.hist(y,50)
     f2.show()
################################################################################

19.
  ==============
```

```
Online:
   diffeq1.mp4
==============

Finite differencing
   draw curve with straight line approximation to tangent
   [x(t + dt) - x(t)]/dt

   Q/C - iR = 0
      i = -dQ/dt
   dQ/dt + Q/RC = 0
   C = 10 uF, R = 20k, RC = 0.2

   Q(t + dt) - Q(t) = -Q(t)dt/RC
   Q(t + dt) = Q(t)*(1 - dt/RC)

   capfd.py

Second derivative finite difference
   draw curve with straight line approximation to tangent
      three points, two slopes
      { [x(t+2dt) - x(t+dt)]/dt - [x(t+dt) - x(t)]/dt }/dt
      { x(t+2dt) - 2x(t+dt) + x(t) }/dt^2

   m d^2x/dt^2 + kx = 0
   d^2x/dt^2 + w^2x = 0

   x(t+2dt) = 2x(t+dt) - x(t)(1 + w^2*dt^2)

   shmfd.py

Senior thesis overview (Laplace's equation in einzel lens)


20.
==============
Online:
   diffeq2.mp4
==============

2-d finite differencing
   D = del, p = phi
   Laplace eqn
   E = -Dp
   D.E = D.(-Dp) = d^2p/dx^2 + d^2p/dy^2 = \rho/\epsilon
       = 0 in free space

   {p(x+dx,y) - 2p(x,y) + p(x-dx)}/dx^2} +
   {p(x,y+dy) - 2p(x,y) + p(x,y-dy)}/dy^2 = 0
   4 p(x,y) = p(x+dx,y) + p(x-dx,y) + p(x,y+dy) + p(x,y-dy)

   p(x,y) = [p(x+dx,y) + p(x-dx,y) + p(x,y+dy) + p(x,y-dy)]/4

Relaxation
   laplace_cap.py

a = np.arange(9).reshape((3,3)) + 1
np.roll(a,-1,axis=0)
np.roll(a,1,axis=0)
np.roll(a,-1,axis=1)
```

```
    lapcap_roll.py
    lapcap_live.py

    Sparse matricies
      -4p(x,y) + p(x+dx,y) + p(x-dx,y) + p(x,y+dy) + p(x,y-dy) - b = 0

       Ax = b;  b from boundary points

       x = A^{-1}b

    ESCI evaluations
################################################################################
```