review of the first 18 lectures

1. Exam on Friday 7 October, at 10AM
   - in class or take home?

2. Review Questions
   - rational approximations and roots of polynomials
   - series developments
   - explain terminology, iterators and enumeration
   - numerical integration and fitting data
   - Lagrange interpolation; files and dictionaries

MCS 507 Lecture 19
Mathematical, Statistical and Scientific Software
Jan Verschelde, 5 October 2011

review of the first 18 lectures

exam policy

Friday 7 October is our midterm exam, which could be either

- An in-class conventional exam, open book and notes, but without computer;

or

- A take-home exam due on Monday 10 October at 10AM which must be solved individually, but will need the use of software.

review of the first 18 lectures

1. Exam on Friday 7 October, at 10AM
   - in class or take home?

2. Review Questions
   - rational approximations and roots of polynomials
   - series developments
   - explain terminology, iterators and enumeration
   - numerical integration and fitting data
   - Lagrange interpolation; files and dictionaries

rational approximations

Compute rational approximations for $\sqrt{3}$ using `sympy` and list comprehensions, working with approximations accurate from 3 to 10 decimal places.

take home question: Write a script to compute approximations for $\sqrt{\alpha}$ applying Newton's method on the equation $x^2 - \alpha = 0$. What is a good starting value for Newton's method? The iterations of Newton's method stop when the user given number of correct decimal places in $\sqrt{\alpha}$ are reached.

plotting roots

Roots of random polynomials tend to lie around the unit circle in the complex plane.
Write a script to make a plot of the roots of a random polynomial of degree 100.

take home question: Use Tkinter to make a simple GUI to display plots of roots of random polynomials.
Also plot the roots of the derivative. Where are the roots of the derivative located with respect to the roots of the original polynomial?

review of the first 18 lectures

1. Exam on Friday 7 October, at 10AM
   - in class or take home?

2. Review Questions
   - rational approximations and roots of polynomials
   - series developments
   - explain terminology, iterators and enumeration
   - numerical integration and fitting data
   - Lagrange interpolation; files and dictionaries

Fibonacci numbers

The Fibonacci numbers arise as the coefficients of the Taylor expansion of $g(z) = \dfrac{z}{1 - z - z^2}$ about $z = 0$.

Use `sympy` or Sage to compute the first 10 terms of the series development of $g(z)$ about $z = 0$.
Extract the coefficients of the terms in the series.

take home question: Compare the efficiency of the series development computation for the Fibonacci numbers with the straightforward recursive definition of the Fibonacci numbers as $F_{n+2} = F_{n+1} + F_n$, $F_0 = 0$, $F_1 = 1$.

review of the first 18 lectures

callback functions

What is a callback function?
Give the definition.
Describe an illustration of a good use.

take home question: Add a callback function to the code for the power method to compute the largest eigenvalue:

1. The return value of the callback is a boolean to continue or stop the iteration.
2. Use the callback function to stop when two consecutive approximations are close enough.

OO design of iterator

Consider the power method to compute the largest eigenvalue of a complex matrix.
Describe the object-oriented design of an iterator object.
Define the data attributes and the `next` method.

take home question: Give Python code for a class to compute the largest eigenvalue using an iterator.
The main function in the code generates a random matrix for testing the iterator object.

enumeration and backtracking

Write a recursive Python function for the following:
Given a number d and a list of numbers L,
print all sublists S of L where the sum of the elements of S equals d.

take home question: In chess, a queen can attack all squares that run horizontally, vertically, and diagonally starting at the queen's position. On an 8-by-8 chess board, compute in how many ways you can put 8 queens so they do not attack each other.

review of the first 18 lectures

the midpoint rule

The midpoint rule to approximate an integral is

$$\int_a^b f(x)dx \approx (b-a)f\left(\frac{a+b}{2}\right).$$

Give Python code using `sympy` or Sage to show that the midpoint rule is exact for all linear functions f.

take home question: Write a script using `sympy` or Sage to set up the equations on the weights and nodes of a 2-point quadrature formula $w_1 f(x_1) + w_2 f(x_2)$ to approximate the integral of $f(x)$ over $[a, b]$. With 4 unknowns, we obtain a system of 4 equations. Solve it using `sympy` or Sage.

applying polyfit

Consider

```
>>> L = [(89, 81), (59, 27), (69, 42),
... (47, 84), (43, 17), (31, 79)]
```

Give commands for an interactive Python session to detect whether the (x, y) points in L show an increasing or decreasing trend.

take home question: Make a plot with pylab or pyplot of fits of consecutive degrees. Which degree fits best the data?

review of the first 18 lectures

1. Exam on Friday 7 October, at 10AM
   - in class or take home?

2. Review Questions
   - rational approximations and roots of polynomials
   - series developments
   - explain terminology, iterators and enumeration
   - numerical integration and fitting data
   - Lagrange interpolation; files and dictionaries

interpolating data

Consider a list of tuples (x_i, y_i), for $i = 0, 1, \ldots, n$ with $x_i \neq x_j$ for all $i \neq j$. The Lagrange polynomials are defined as

$$L_i(x) = \prod_{\substack{j = 0 \\ j \neq i}} \frac{x - x_j}{x_i - x_j}$$

and $p(x) = \displaystyle\sum_{i=0}^{n} y_i L_i(x)$ is an interpolating polynomial: $p(x_i) = y_i$, for $i = 0, 1, \ldots, n$. Write Python code using `sympy` to define $L_i(x)$ and p for a given list of tuples.

take home question: Write a module for Lagrange interpolation with `sympy.Poly` objects. As test function, generate a random polynomial q of degree n with integer coefficients and let the list of $n+1$ integer points $(k, q(k))$ for $k = 0, 1, \ldots, n$. The test verifies that $p \equiv q$.

frequency of words

Write a Python script that prompts the user for a file name and that prints a dictionary.
The dictionary is a frequency table for the words on file.
The keys are the words and the values the number of times each word occurs.

take home question: Download a text file from `www.gutemberg.org`, e.g. the “War of the Worlds” and find the 100 most frequently occurring words.