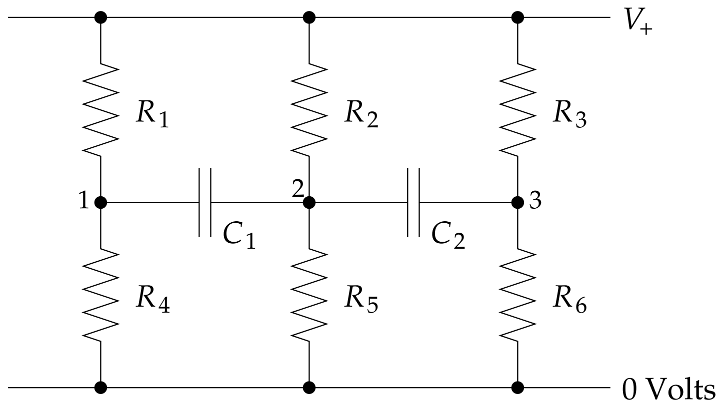
UNIVERSITY OF SUSSEX
Scientific Computing
Tutor: Dr. Ilian Iliev, Office: Pev III 4C5

Assignment 1
Deadline: 12pm on Monday, October 31st, 2016.
Penalties will be imposed for submissions beyond this date.
Final submission date: Tuesday, November 1st, 2016.
No submissions will be accepted beyond this date.

1. Consider the following RC circuit:



The voltage V_+ is time-varying and sinusoidal of the form $V_+ = x_+ e^{i\omega t}$ with x_+ a constant. The resistors in the circuit can be treated using Ohm's law as usual. For the capacitors the charge Q and voltage V across them are related by the capacitor law $Q = CV$, where C is the capacitance. Differentiating both sides of this expression gives the current I flowing in on one side of the capacitor and out on the other:

$$I = \frac{dQ}{dt} = C\frac{dV}{dt}.$$

(a) Assuming the voltages at the points labeled 1, 2, and 3 are of the form $V_1 = x_1 e^{i\omega t}$, $V_2 = x_2 e^{i\omega t}$, and $V_3 = x_3 e^{i\omega t}$, apply Kirchhoff's law at each of the three points, along with Ohm's law and the capacitor law, to show that the constants x_1, x_2, and x_3 satisfy the equations

$$\left(\frac{1}{R_1} + \frac{1}{R_4} + i\omega C_1\right)x_1 - i\omega C_1 x_2 = \frac{x_+}{R_1},$$
$$-i\omega C_1 x_1 + \left(\frac{1}{R_2} + \frac{1}{R_5} + i\omega C_1 + i\omega C_2\right)x_2 - i\omega C_2 x_3 = \frac{x_+}{R_2},$$
$$-i\omega C_2 x_2 + \left(\frac{1}{R_3} + \frac{1}{R_6} + i\omega C_2\right)x_3 = \frac{x_+}{R_3}.$$

(b) Write a program to solve for x_1, x_2, and x_3 when

$$R_1 = R_3 = R_5 = 1\,\mathrm{k}\Omega,$$
$$R_2 = R_4 = R_6 = 2\,\mathrm{k}\Omega,$$
$$C_1 = 1\,\mu\mathrm{F}, \qquad C_2 = 0.5\,\mu\mathrm{F},$$
$$x_+ = 3\,\mathrm{V}, \qquad \omega = 1000\,\mathrm{s}^{-1}.$$

Notice that the matrix for this problem has complex elements. You will need to define a complex array to hold it, but you can still use the `solve` function just as before to solve the equations—it works with either real or complex arguments. Using your solution have your program calculate and print the amplitudes of the three voltages V_1, V_2, and V_3 and their phases in degrees. (Hint: You may find the functions `polar` or `phase` in the `cmath` package useful. If `z` is a complex number then "`r,theta = polar(z)`" will return the modulus and phase (in radians) of `z` and "`theta = phase(z)`" will return the phase alone.) [40]

2. **Ill-conditioned matrices:** Consider the system $Hx = b$ where H is the Hilbert matrix of size n, defined by

$$H[i,j] = \frac{1}{i+j-1}$$

and $b_i = \sum_j H_{ij}$ (so that the vector $x = [1, 1, \ldots, 1]$ solves the system). You can use the provided function `hilb(n)`to generate the Hilbert matrix of size $n \times n$. For $n = 2 : 20$, use the in-build NumPy linear solver `solve(A,b)` to find x and plot the maximal error in x as a function of n (use a logarithmic scale in the error). Notice how the error grows quickly, and becomes of the same order as the solution for $n > 12$, at which point your solutions of $Hx = b$ become meaningless, although of course the computer will cheerfully continue to give you results for larger n unless you stop it from doing so.

Try to do the same as above using the provided code for Gauss elimination with partial pivoting (`gaussPivot.py`). Compare the results with the in-build code above. What go you conclude? [30]

3. Interpolate the following function:

$$f(x) = \begin{cases} -1 & \text{if } x < 0 \\ +1 & \text{if } x \geq 0 \end{cases}$$

with 11 equally spaced points in the interval $[-\pi, \pi]$, as follows:

(a) Set up function values using the `linspace` command. Calculate (using the in-build `scipy.interpolate.ba` the values of the interpolation polynomial of maximal order to interpolate $f(x)$ (e.g. the Lagrangian interpolation polynomial) at the points $x = -1.5, 1$ and 3. Compare with the true values of $f(x)$ at the same points. What is the order of the interpolating polynomial?

(b) Now use the in-build Python function `interp1d` to interpolate the function $f(x)$ again, but now using cubic spline. Give again the values at $x = -1.5, 1$ and 3 for the interpolation and compare them with the true values of $f(x)$ at those points.

(c) Make a figure for $f(x)$ and its interpolations. On the figure plot the data points as circles and plot the exact function and interpolating functions from (b) and (c) as lines with different colours on a finer spacing (with 1000 points in the interval $[-\pi, \pi]$).

(d) Fit the same data from (a) above with the function $a * tanh(b * x + c)$. Add some random noise to the data (as real data from experimental measurements would have), as follows:

$$y_{noisy} = y + 0.1 * np.random.random(len(x))$$

Repeat the fit with the above function using the noisy data and plot both fits against the noisy data. How much did the parameters a, b and c change? What do you conclude?

Note: This last fit is related to the results of Planck and WMAP satellites on the timing and duration of the Epoch of Reionization, see e.g. `http://adsabs.harvard.edu/abs/2016A%26A...594A..13P`. [30]