Hand in 6, part 1 of 2 - Projection Methods

1 Solving the ODE-BVP with the Conjugate Gradient Method

Consider the ODE-BVP that we worked with during Lab 6, part 1.
Note: The CG method works for SPD matrices. This matrix is not SPD unless you multiply both sides of the equation with -1. Then you can see that the matrix is SPD by numerically computing the eigenvalues (but you don't have to do that).

1. Rewrite the system so that the matrix is SPD.

2. Implement a python function for the conjugate gradient-method in your code from Lab 6, part 1, by filling in the missing parts in the below function. The script follows the last from the lecture, so that you don't have to think about Arnoldi's.

```
def cg(A,b,x_cg,maxit):
  rk = b-np.dot(A,x_cg)
  p = rk
  for k in np.arange(maxit):
    alpha = np.dot(rk,rk)/np.dot(p,np.dot(A,p))
    x_cg = x_cg + alpha*p
    rkp1 = rk - ...
    beta = ....
    p = rkp1 + ...
    rk = rkp1
  return x_cg
```

3. Use your function for solving the ODE-BVP from Lab 6 part 1, for N=100. Don't use the many different right hand sides, you can set the right boundary condition to 2. How many iterations do you need in order for the numerical solution to be close to the analytical (looking at the plot - you don't need to do any fancy error computations)?

4. Compare to how many iterations you need with the Jacobi method. Is CG more efficient?

This should be included in this part of the hand in: Your CG code question 2, numbers from question 3, a yes or no from question 4.