

```

import numpy as np
import scipy.sparse as sp
from scipy.sparse.linalg import spsolve
import scipy.linalg as linalg
import matplotlib.pyplot as plt

def with_dense_matrix(N):
    D = N**2 #size of matrix A
    y0 = 1 #boundary condition for y = 0

    A = (-4) * np.eye(D) + np.eye(D, k=-1) + np.eye(D, k=1) + np.eye(D, k=N) + np.eye(D, k=-N)
    for n in np.arange(N, D, step=N):
        A[n, n-1] = 0
        A[n-1, n] = 0

    b = np.zeros(D)
    b[:N] = -y0

    np.savetxt('dense_matrix_solved.dat', linalg.solve(A, b))

    # One problem I see here is that the boundary is not included in the plot.
    # We could solve tha by simply appending to the results array, but it would
    # make the solution so much uglier. That is why I omitted it.

def with_sparse_matrix(N):
    D = N**2
    y0 = 1

    # we need to define the format of the sparse matrix as reshaping
    # etc. is not available
    # with the default format
    A = (-4) * sp.eye(D, format='csr') + sp.eye(D, k=-1) + sp.eye(D, k=1) + sp.eye(D, k=N) +
    sp.eye(D, k=-N)
    for n in np.arange(N, D, step=N):
        A[n, n-1] = 0
        A[n-1, n] = 0

    b = np.zeros(D)
    b[:N] = -y0

    np.savetxt('sparse_matrix_solved.dat', spsolve(A, b))

if __name__ == "__main__":
    N = 50 #size of square grid

    with_dense_matrix(N)
    with_sparse_matrix(N)

    DS = np.array(np.loadtxt('dense_matrix_solved.dat'))
    SS = np.array(np.loadtxt('sparse_matrix_solved.dat'))

    _, ax = plt.subplots(1, 2)
    # makes a heat map from the values in the results matrix
    # origin lower sets the start to be from the lower left (else it would
    # start at the top)
    # extent tells the axis to go from 0 to 1
    # cmap chooses a nice looking color map for the heat map
    im1 = ax[0].imshow(np.reshape(DS, (N, N)), extent=[0,1,0,1], cmap='jet', origin='lower')
    im2 = ax[1].imshow(np.reshape(SS, (N, N)), extent=[0,1,0,1], cmap='jet', origin='lower')
    ax[0].set_title("Dense matrix")
    ax[1].set_title("Sparse matrix")
    plt.colorbar(im1, ax=ax[0], shrink=0.5)
    plt.colorbar(im2, ax=ax[1], shrink=0.5)
    plt.show()

```