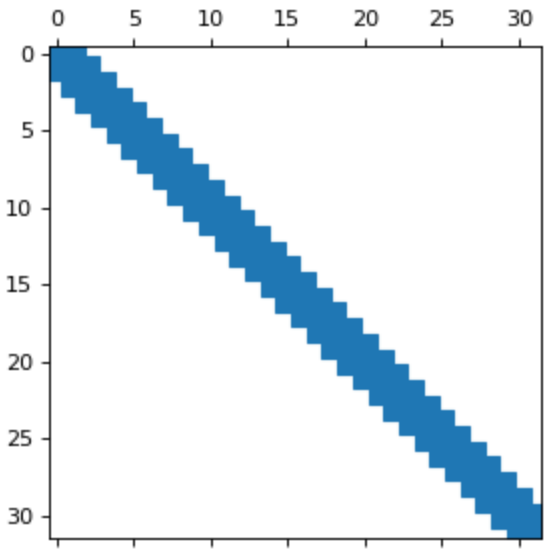
In [1]:
```
import numpy as np
import matplotlib.pyplot as plt
import scipy.sparse
import time
```

In [2]:
```
def rhs_f(x):
    return 20*np.cos(2*np.pi*x)

def create_A(n):
    subdiag = np.ones(n-1)
    maindiag = -2*np.ones(n)
    superdiag = np.ones(n-1)

    A = scipy.sparse.diags([subdiag, maindiag, superdiag],  [-1, 0, 1], shape=(n, n),
    return A

def g_true(x):
    return 5/(np.pi**2)*(1-np.cos(2*np.pi*x))
```

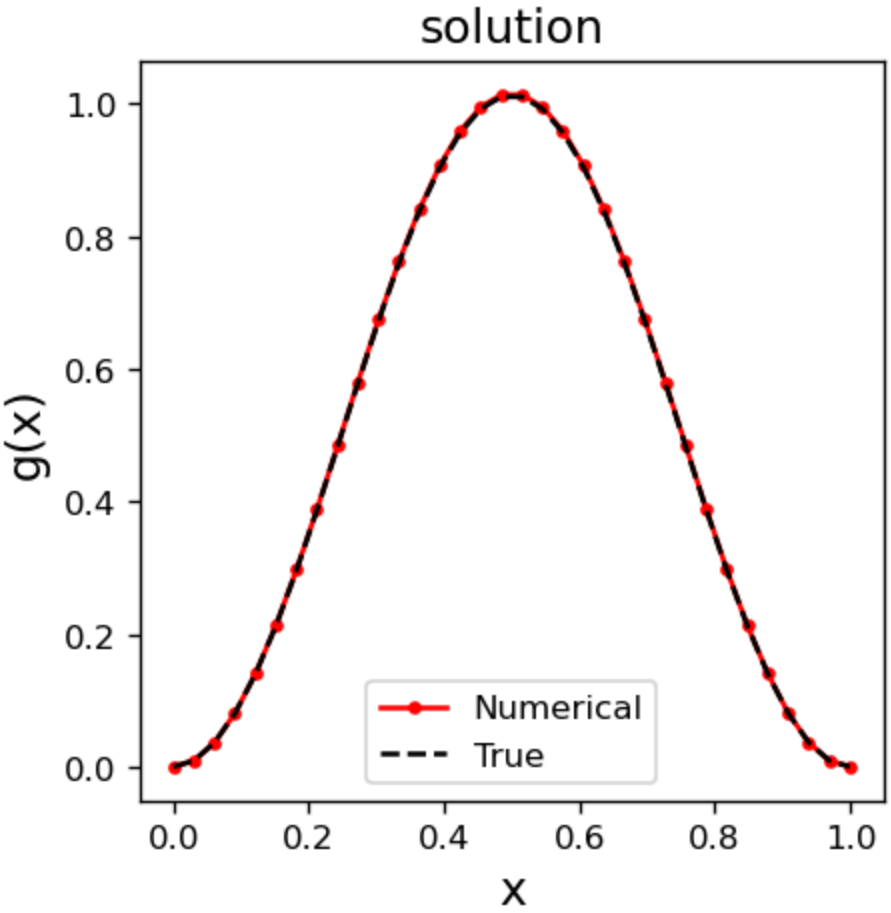
In [3]:
```
# start with just 32 to visualize
n = 32

A = create_A(n)

fig = plt.figure(figsize=(4,4),dpi = 80)
plt.spy(A)
plt.show()
```



In [4]:
```
h = 1/(n+1)
# all points
x_array_full = np.linspace(0,1,n+2)
# just interior points
x_array_int = x_array_full[1:-1]
# rhs
f_array = rhs_f(x_array_int)
# matrix A
A = create_A(n)*1/h**2
LU = scipy.sparse.linalg.splu(A)
g_array = LU.solve(f_array)
```

```
g_array_full = np.zeros(np.size(x_array_full))
g_array_full[1:-1] = g_array

g_array_true = g_true(x_array_full)

fig = plt.figure(figsize=(4,4),dpi = 120)
plt.plot(x_array_full,g_array_full,'r.-',label="Numerical")
plt.plot(x_array_full,g_array_true,'k--',label="True")
plt.xlabel('x',fontsize=14)
plt.ylabel('g(x)',fontsize=14)
plt.title('solution',fontsize=14)
plt.legend()
plt.show()
```

```
C:\ProgramData\Anaconda3\lib\site-packages\scipy\sparse\linalg\_dsolve\linsolve.py:34
7: SparseEfficiencyWarning: splu converted its input to CSC format
  warn('splu converted its input to CSC format', SparseEfficiencyWarning)
```



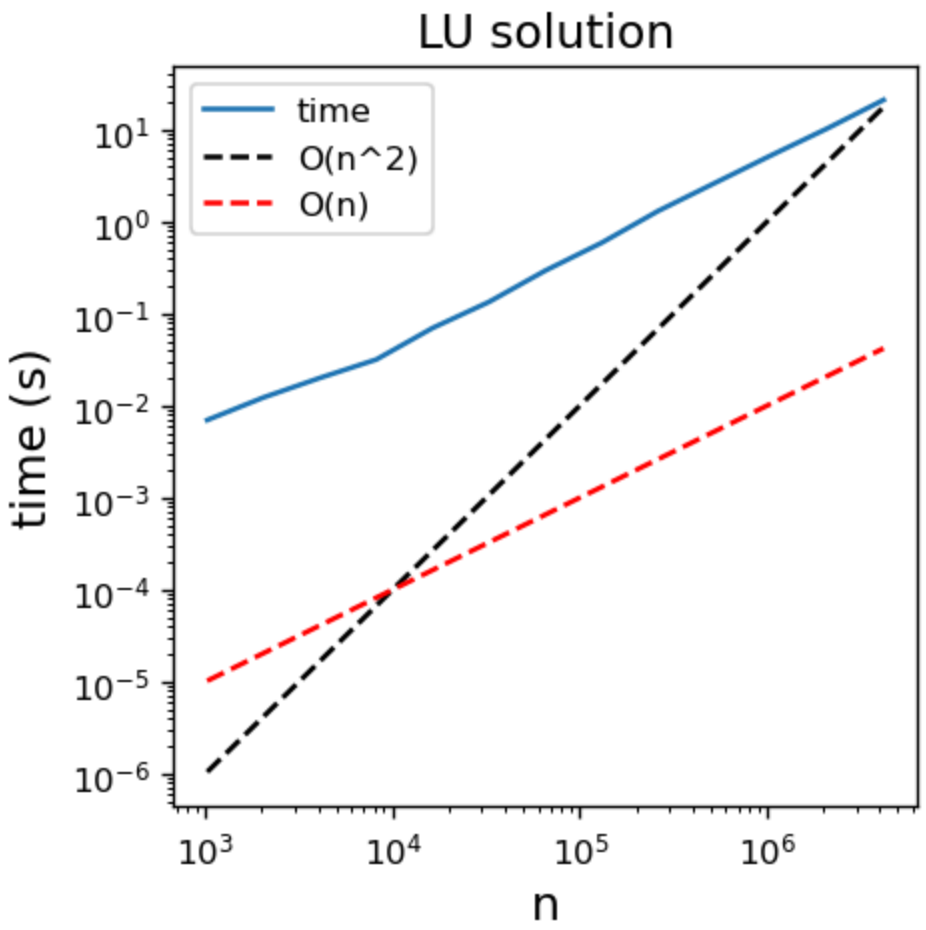
In [5]:
```
# number of interior points
n_array = 2**(np.arange(10,23,1))
timings_array = np.zeros(np.size(n_array))

for k in range(0,np.size(n_array)):
    n = n_array[k]
    print(n)
    h = 1/(n+1)
    # all points
    x_array_full = np.linspace(0,1,n+2)
```

```
    # just interior points
    x_array_int = x_array_full[1:-1]
    # rhs
    f_array = rhs_f(x_array_int)
    # matrix A
    A = create_A(n)*1/h**2

    # timings
    mytstart = time.time()

    LU = scipy.sparse.linalg.splu(A)
    g_array = LU.solve(f_array)

    mytend = time.time()

    timings_array[k] = mytend-mytstart
```

```
1024
2048
4096
8192
16384
32768
65536
131072
262144
524288
1048576
2097152
4194304
```

In [10]:

```
fig = plt.figure(figsize=(4,4),dpi = 120)
plt.loglog(n_array,timings_array,label='time')
plt.loglog(n_array,(n_array/1e6)**2,'k--',label='O(n^2)')
plt.loglog(n_array,n_array/1e8,'r--',label='O(n)')
plt.xlabel('n',fontsize=14)
plt.ylabel('time (s)',fontsize=14)
plt.title('LU solution',fontsize=14)
plt.legend()
plt.show()
```


LU solution
time
O(n^2)
O(n)
time (s)
n
10^{1}
10^{0}
10^{-1}
10^{-2}
10^{-3}
10^{-4}
10^{-5}
10^{-6}
10^{3}
10^{4}
10^{5}
10^{6}

In [ ]: