In [55]:

```
import numpy as np
import scipy.sparse as sparse
import scipy.linalg as sla
import scipy.sparse.linalg as spla
import matplotlib.pyplot as plt
%matplotlib inline
```

Let's make a *random* sparse matrix

First we'll set the density so that

$$density = \frac{nnz(A)}{n^2}$$

In [75]:

```
n = 100
density = 10.0 / n # 5 points per row
nnz = int(n*n*density)
```

Now make the entries:

In [76]:

```
row = np.random.random_integers(low=0, high=n-1, size=nnz)
col = np.random.random_integers(low=0, high=n-1, size=nnz)
data = np.ones(nnz, dtype=float)

A = sparse.coo_matrix((data, (row, col)), shape=(n, n))
print(A.dtype)
```

```
float64
```

But let's make it positive definite:

In [77]:

```
A.data[:] = -1.0                  # -1 for off-diagonals
rowsum = -np.array(A.sum(axis=1)) + 1 # positive rowsum
rowsum = rowsum.ravel()
A.setdiag(rowsum)
```

In [78]:

```
u = np.random.rand(n)
v = np.random.rand(n)
```

In [79]:

```
A = A.tocsc()
%timeit s = spla.splu(A)
```

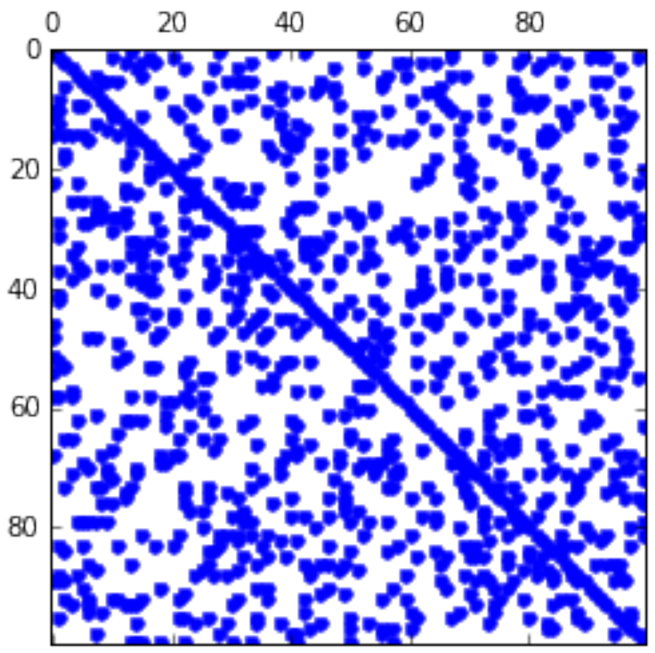
```
1000 loops, best of 3: 399 µs per loop
```

In [80]:

```
plt.spy(A, marker='.')
```

Out[80]:

```
<matplotlib.lines.Line2D at 0x10d8002b0>
```



In [81]:

```
B = A.todense()
```

In [82]:

```
%timeit p, L, U = sla.lu(B)
```

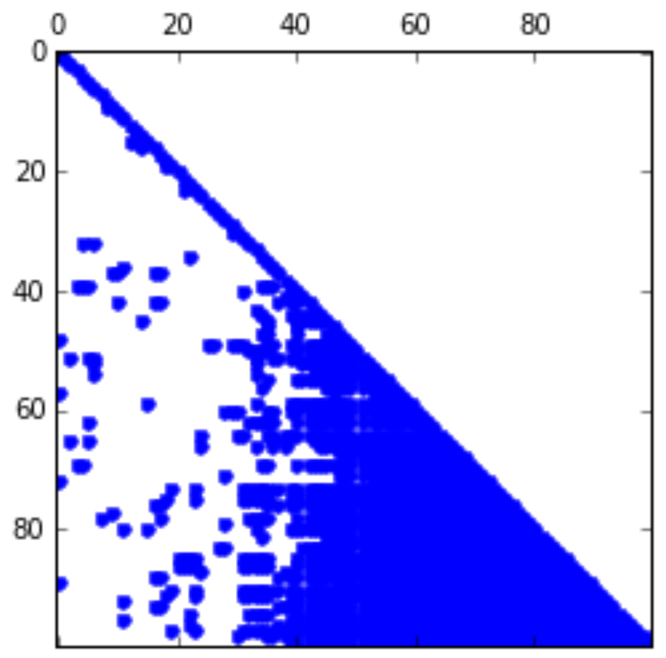
```
1000 loops, best of 3: 216 µs per loop
```

```
In [83]:
```

```
plt.spy(s.L, marker='.')
```

```
Out[83]:
```

```
<matplotlib.lines.Line2D at 0x10d89a0f0>
```



```
In [ ]:
```

```
In [ ]:
```