solver_scaling

September 22, 2021

1 Example of Solving Linear Systems of Different Sizes

[1]:

```
#import packages for example
import numpy as np
from timeit import timeit
import matplotlib.pyplot as plt
```

1.1 Setting up and solving a linear system

Set linear system size

[2]:

```
N=10;
```

Setup random dense linear system with reference solution

[3]:

```
A = np.random.rand(N,N);
xref = np.ones(N);
b = np.dot(A,xref);
```

Solve the linear system by some method

[4]:

```
x = np.linalg.solve(A,b)
```

1.2 Verify the result

[5]:

```
np.linalg.norm(x-xref,2)
```

[5]:

```
1.2498271287878353e-14
```

1.3 Measure time for solution

[6]:

```
t = timeit(lambda: np.linalg.solve(A,b), number=10)
print(f"{t} ms".format(t * 1000/ 10))
```

```
0.0005191999999993868 ms
```

1.4 Investigate scaling of solve time vs system size

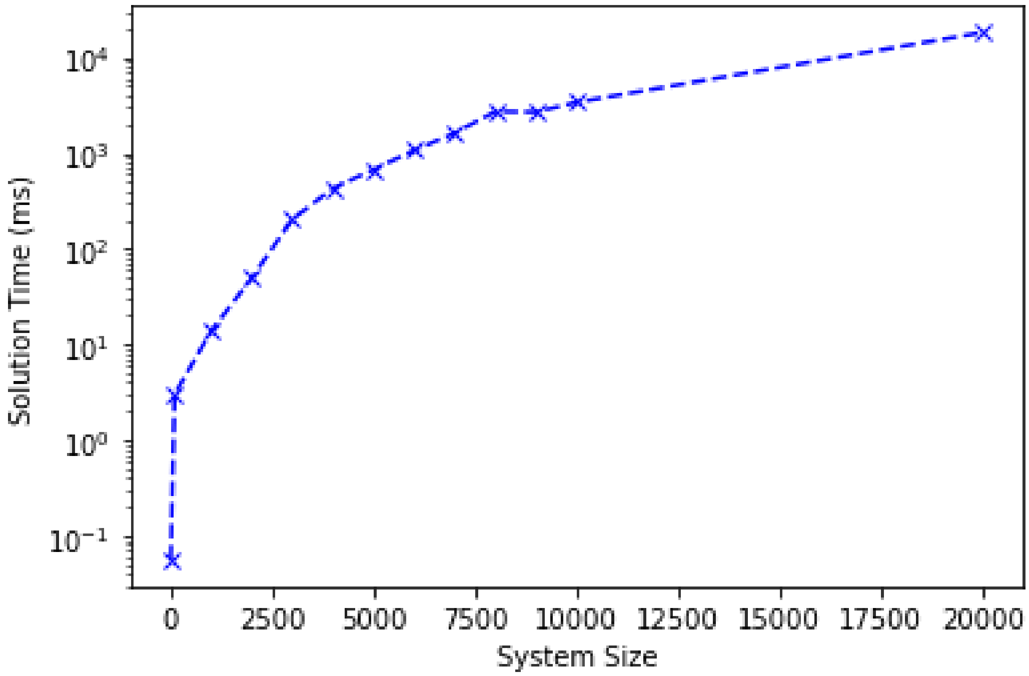
[17]:

```
systemSize=[10,100,1000,2000,3000,4000,5000,6000,7000,8000,9000,10000,20000];
solnTime=[];
for i in systemSize:
    N=i
    A = np.random.rand(N,N);
    xref = np.ones(N);
    b = np.dot(A,xref);

    t = timeit(lambda: np.linalg.solve(A,b), number=10)

    solnTime.append(t*1000/10)

plt.semilogy(systemSize,solnTime,'bx--')
plt.xlabel("System Size");
plt.ylabel("Solution Time (ms)");
```


10^4
10^3
10^2
10^1
10^0
10^{-1}
Solution Time (ms)
System Size