CLASS: CSE7345

SMUID: 47812509

Quest 8 - Regression Transpose

A. Simple

In [25]:
```
import numpy as np
import timeit
import random
```

In [47]:
```
# function to create matrix
def create_matrix(a):
    return np.random.randint(low=1, high=5, size=(a,a))

# function for simple matrix transpose
def simple(mat):
    return [[mat[j][i] for j in range(len(mat))] for i in range(len(mat[0]))]
```

In [48]:
```
m=[200,2000,20000,25000]
#list to store time
simpleList=[]
for i in m:
    z=create_matrix(i)
    simpleMethod = timeit.timeit('simple(z)','from __main__ import simple,z', num
    simpleList.append(simpleMethod)
print (simpleList)
```

```
[0.0267331600189209, 1.4580609798431396, 89.75500679016113, 143.99370098114014]
```

B. Zip

In [51]:
```
# function using zip matrix transpose
def zipp(mat):
    t_matrix = zip(*mat)
```

In [52]:
```
zipList=[]
for i in m:
    z=create_matrix(i)
    zipMethod = timeit.timeit('zipp(z)','from __main__ import zipp,z', number=1)
    zipList.append(zipMethod)
print (zipList)
```

```
[0.004498958587646484, 0.5977120399475098, 23.70847797393799, 36.9933550357818
6]
```

C. Numpy

In [67]:
```
# function using numpy matrix transpose
def numpytranspose(mat):
    return np.transpose(mat)
```

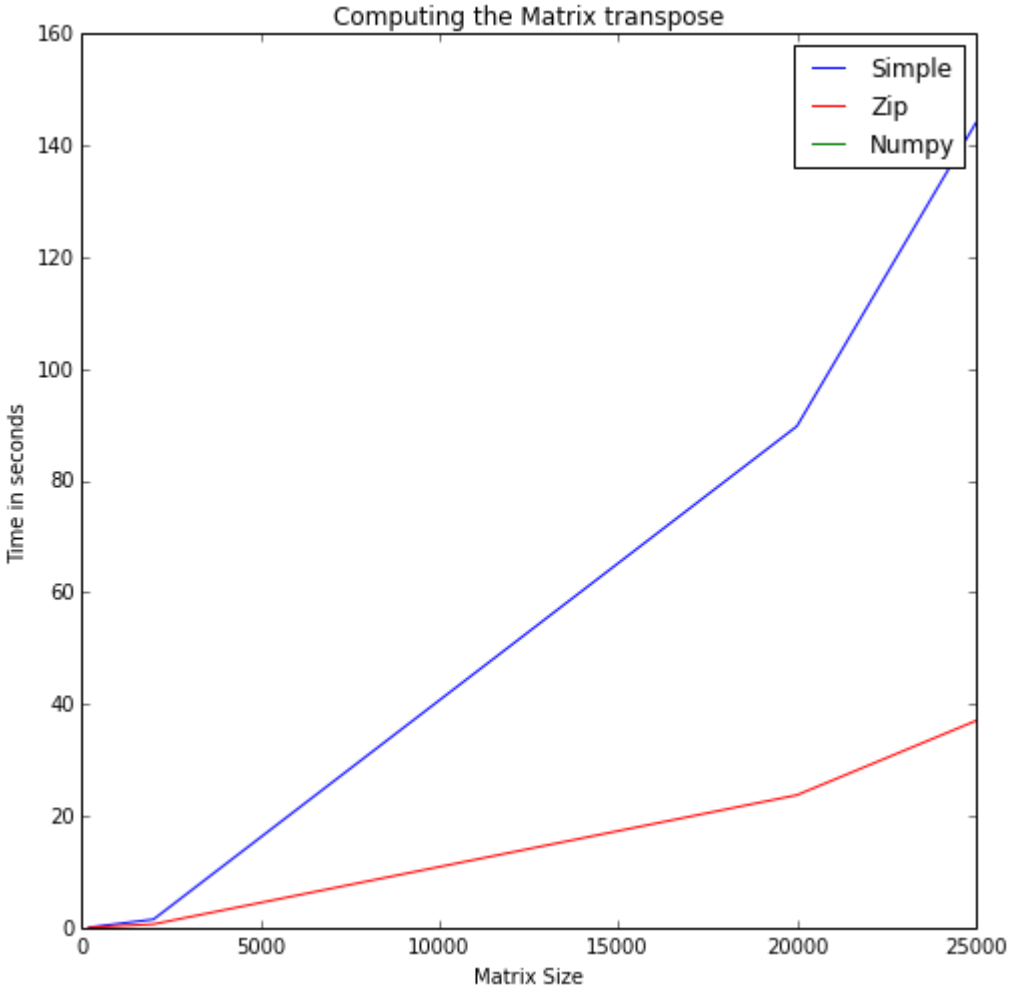
In [68]:
```
numpyList=[]
for i in m:
    z=create_matrix(i)
    NumpyMethod = timeit.timeit('numpytranspose(z)',
                                'from __main__ import numpytranspose,z', number=1
    numpyList.append(NumpyMethod)
print (numpyList)
```

```
[1.4066696166992188e-05, 1.5974044799804688e-05, 1.0013580322265625e-05, 9.0599
06005859375e-06]
```

Graph

In [64]:
```
import matplotlib.pyplot as plt
%matplotlib inline
```

In [65]:
```
plt.figure(figsize=(8,8))
plt.plot(m, simpleList, color='blue', label = "Simple")
plt.plot(m, zipList, color='red', label = "Zip")
plt.plot(m, numpyList, color='green', label = "Numpy")
plt.xlabel('Matrix Size')
plt.ylabel('Time in seconds')
plt.title('Computing the Matrix transpose')
plt.legend(numpoints=2)
plt.show()
```



Q. What are the implications?

A. From the graph, we can confirm that using numpy for matrix transpose is much faster.

Q. What is the time complexity of matrix transposition?

A. Matrix transposition using simple method takes almost double time as compared with zip method, numpy on the other hand takes only few seconds.

Q. Do your results confirm the theory?

A. Matrix transposition is time consuming function if not used without numpy package.