lab3

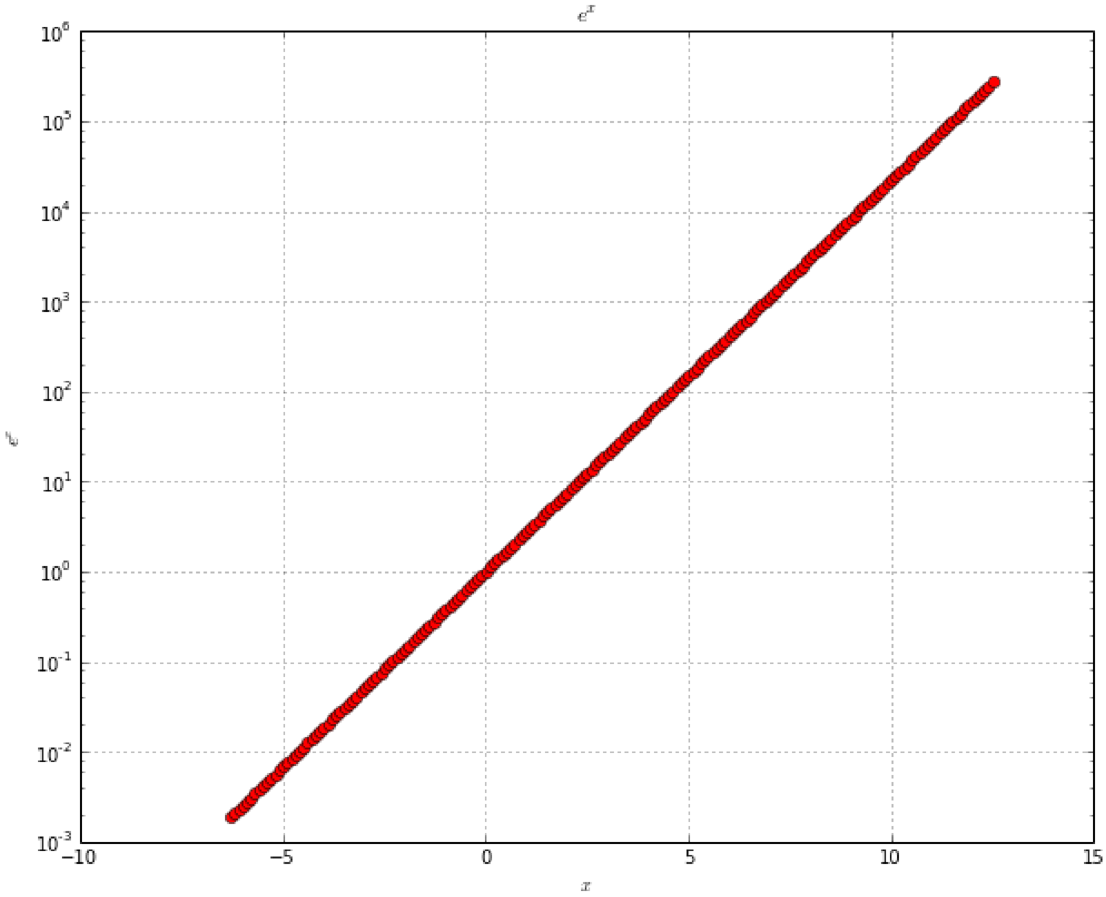
February 14, 2018

```
In [41]: from __future__ import division
         % matplotlib inline

         import numpy as np
         import matplotlib
         from matplotlib import pyplot as plt
         from scipy.integrate import quad


         size=(10,8)
```

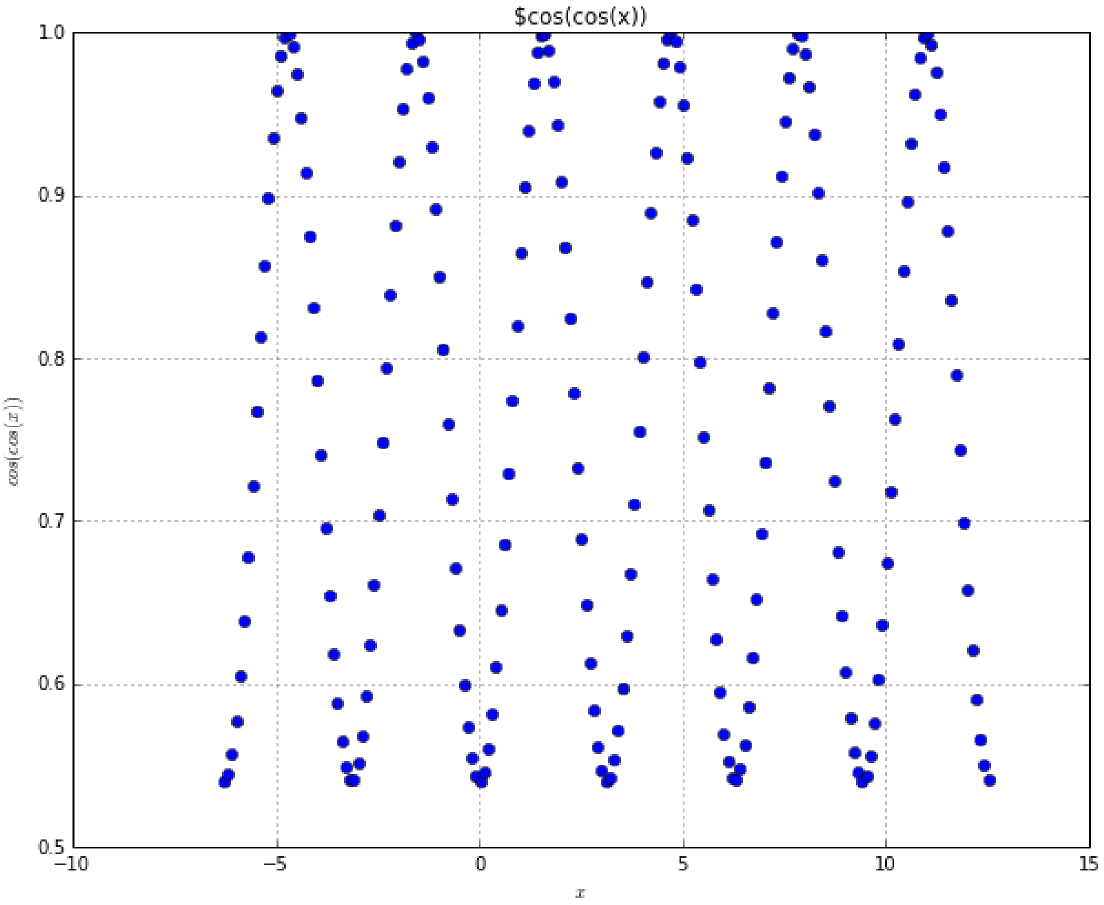
```
In [42]: #functions

         def exponential(x):
             return np.exp(x)

         def coscos(x):
             return np.cos(np.cos(x))

         #a0_exp= (1/(2*np.pi))*(np.e^(2*np.pi)-1)
         #def a_nexp(n):

         def expcos(x,k):
             return (exponential(x))*np.cos(k*x)

         def expsin(x,k):
             return (exponential(x))*np.sin(k*x)

         def cccos(x,k):
             return (coscos(x))*np.cos(k*x)

         def ccsin(x,k):
             return (coscos(x))*np.sin(k*x)
```

```
In [86]: valuerange= np.arange(-2*np.pi, 4*np.pi, 0.1)

         fig1= plt.figure(1, figsize= size)
         axes1= fig1.add_subplot(1,1,1)
```

```
axes1.grid(True)
axes1.set_xlabel("$x$")
axes1.set_ylabel("$e^x$")
axes1.set_title("$e^x$")
random=axes1.semilogy(valuerange, exponential(valuerange),"ro")
```



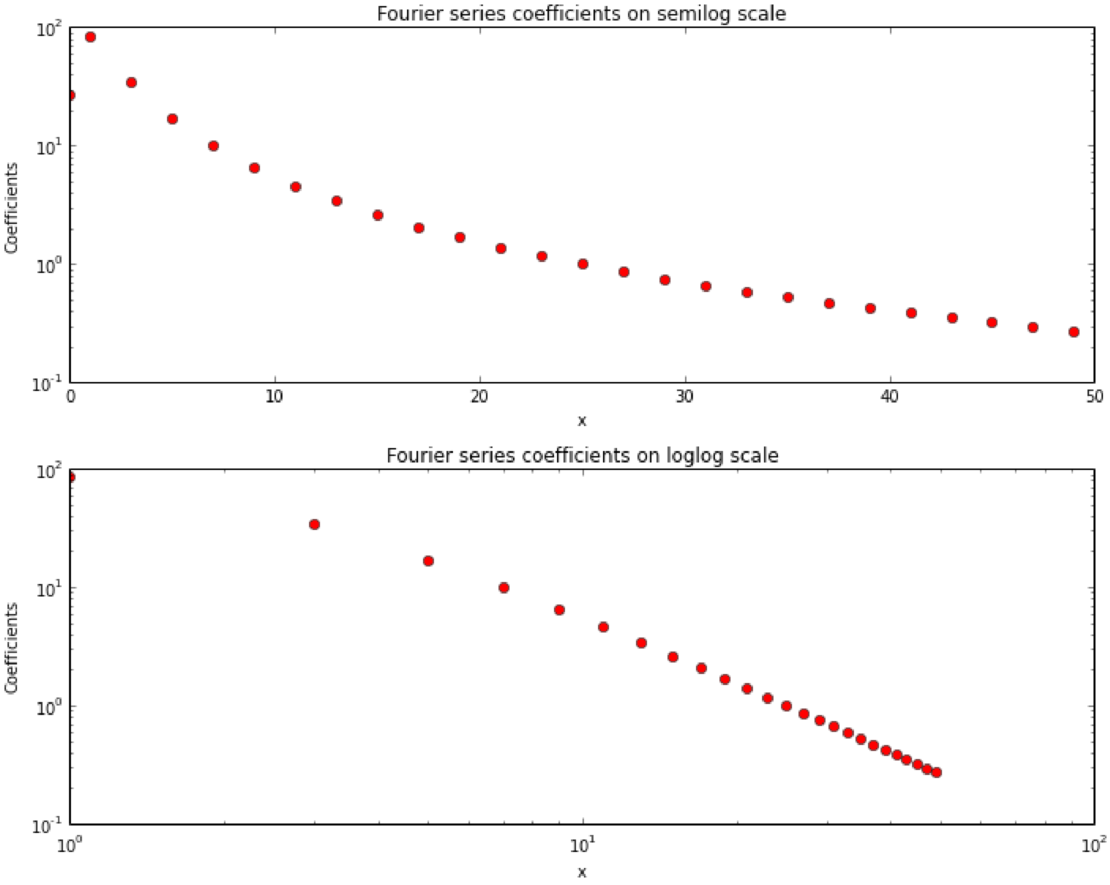
```
In [44]: fig2= plt.figure(2, figsize= size)
         axes2= fig2.add_subplot(111)
         axes2.grid(True)
         axes2.set_xlabel("$x$")
         axes2.set_title("$cos(cos(x))")
         axes2.set_ylabel("$cos(cos(x))$")
         graph= axes2.plot(valuerange, coscos(valuerange),"bo")
```



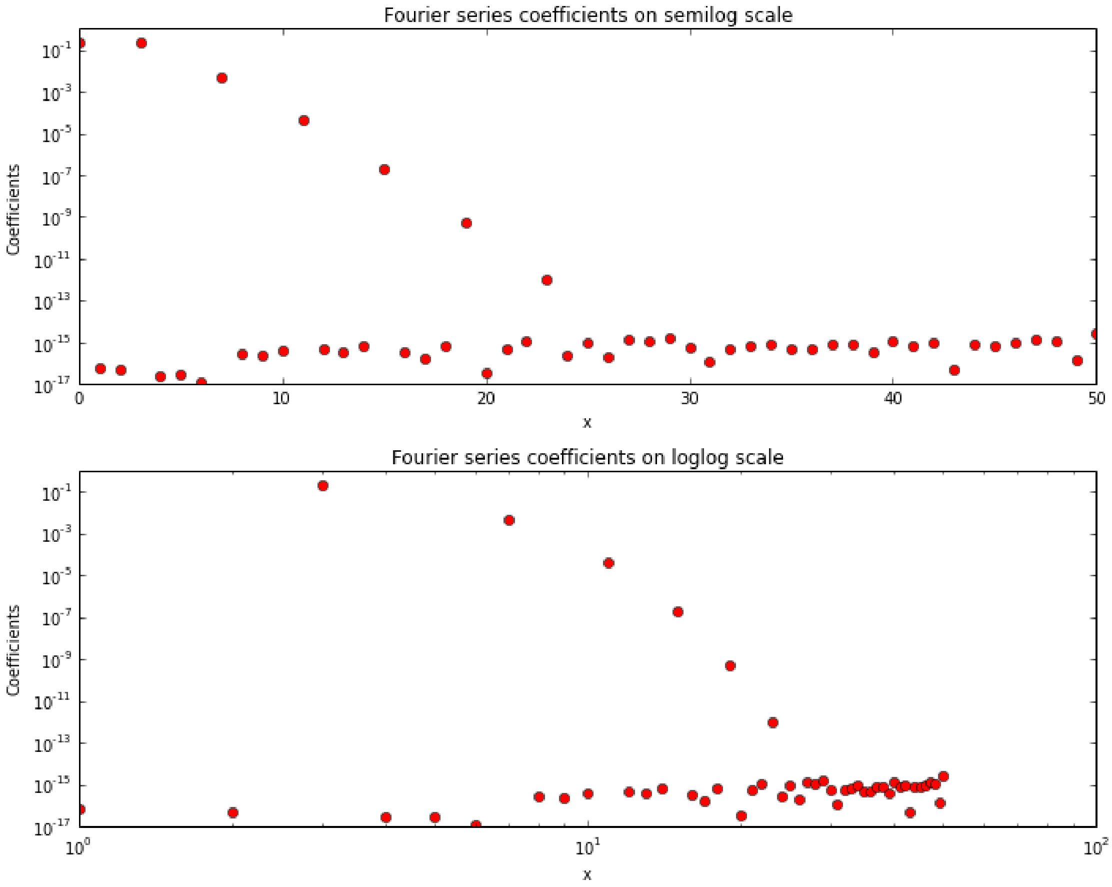
Analytical computation and plotting of the functions. The $cos(cos(x))$ function is π periodic and the e^x function is aperiodic. The equations used for the computation of the fourier series assume that these functions are π periodic. Thus, the fourier coefficients represent the e^x function from $[0, \pi]$ repeated over intervals of length π intervals

```
In [67]: # to find the vector of 25 coefficients

         #find the coefficients for exponential

         coffexp=[]
         a0= quad(exponential, 0, 2*np.pi)[0]

         a0= a0/(2*np.pi)
         coffexp.append(a0)
         for i in range(1,26):
             ai= quad(expcos, 0, 2*np.pi, args=(i))[0]
             bi= quad(expsin, 0, 2*np.pi, args=(i))[0]
             coffexp.append(ai)
             coffexp.append(bi)
```

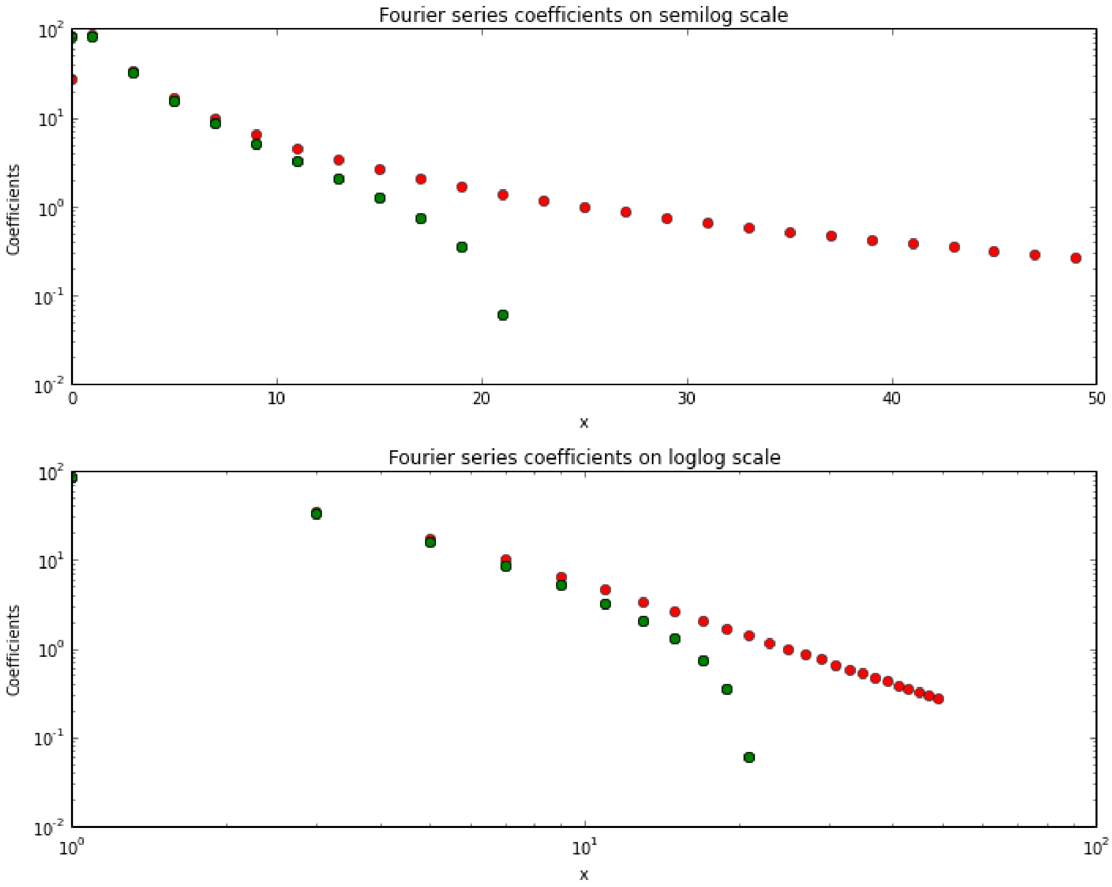
```
coffexp= np.array(coffexp)/(np.pi)


fig3= plt.figure(3, figsize= size)
axes30= fig3.add_subplot(211)
axes30.set_xlabel("x")
axes30.set_ylabel("Coefficients")
axes30.set_title("Fourier series coefficients on semilog scale")
graph= axes30.semilogy( coffexp, "ro")

axes31= fig3.add_subplot(212)
axes31.set_xlabel("x")
axes31.set_ylabel("Coefficients")
axes31.set_title("Fourier series coefficients on loglog scale")
graph= axes31.loglog( coffexp, "ro")
plt.tight_layout()
#plt.subplot(212)
#plt.loglog( coffexp, "ro")
#plt.xlabel("x")
#plt.ylabel("Coefficients")
#plt.title("Fourier series coefficients on loglog scale")
```



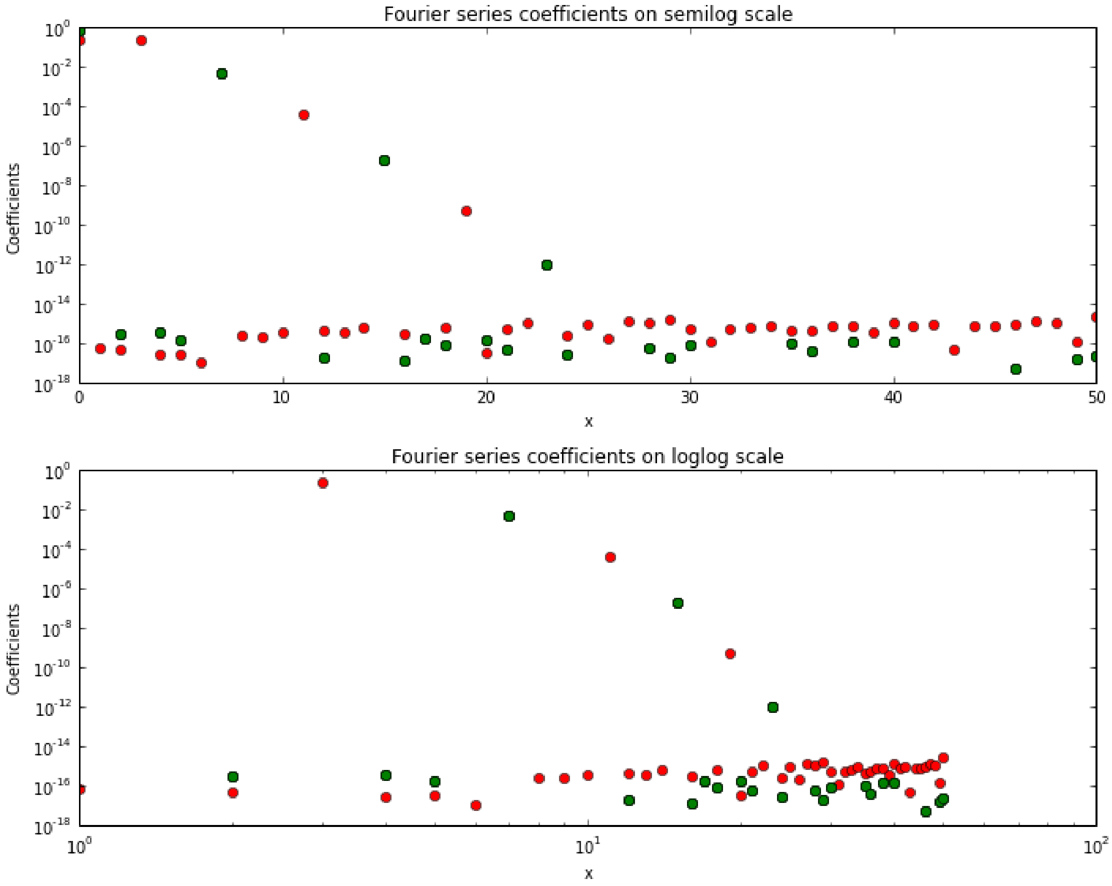
```
In [58]: # to find the vector of 25 coefficients

         #find the coefficients for exponential

         coffcc=[]
         a0= quad(coscos, 0, 2*np.pi)[0]

         a0= a0/(2*np.pi)
         coffcc.append(a0)
         for i in range(1,26):
             ai= quad(cccos, 0, 2*np.pi, args=(i))[0]
             bi= quad(ccsin, 0, 2*np.pi, args=(i))[0]
             coffcc.append(ai)
             coffcc.append(bi)

         coffcc= abs(np.array(coffcc))/(np.pi)


         fig4= plt.figure(4, figsize= size)
         axes40= fig4.add_subplot(2,1,1)
         axes40.set_xlabel("x")
         axes40.set_ylabel("Coefficients")
         axes40.set_title("Fourier series coefficients on semilog scale")
         graph= axes40.semilogy( coffcc, "ro")

         axes41= fig4.add_subplot(2,1,2)
         axes41.loglog( coffcc, "ro")
         axes41.set_xlabel("x")
         axes41.set_ylabel("Coefficients")
         axes41.set_title("Fourier series coefficients on loglog scale")

         plt.tight_layout()
```



```
In [47]: x= np.linspace(0, 2*np.pi, 401)
         x= x[:-1]
         A= np.zeros((400,51))
         A[:,0]= 1
         for i in range(1,26):
             A[:,2*i -1]= np.cos(i*x)
             A[:,2*i]= np.sin(i*x)

         b_exp= exponential(x)
         b_cc= coscos(x)

         c_exp= np.linalg.lstsq(A,b_exp)[0]
         c_cc= np.linalg.lstsq(A,b_cc)[0]

In [70]: graph= axes30.semilogy(c_exp, "go")
         graph= axes31.loglog(c_exp, "go")
         fig3

   Out[70]:
```



```
In [81]: graph= axes40.semilogy(c_cc, "go")
         graph= axes41.loglog(c_cc, "go")
         fig4

   Out[81]:
```

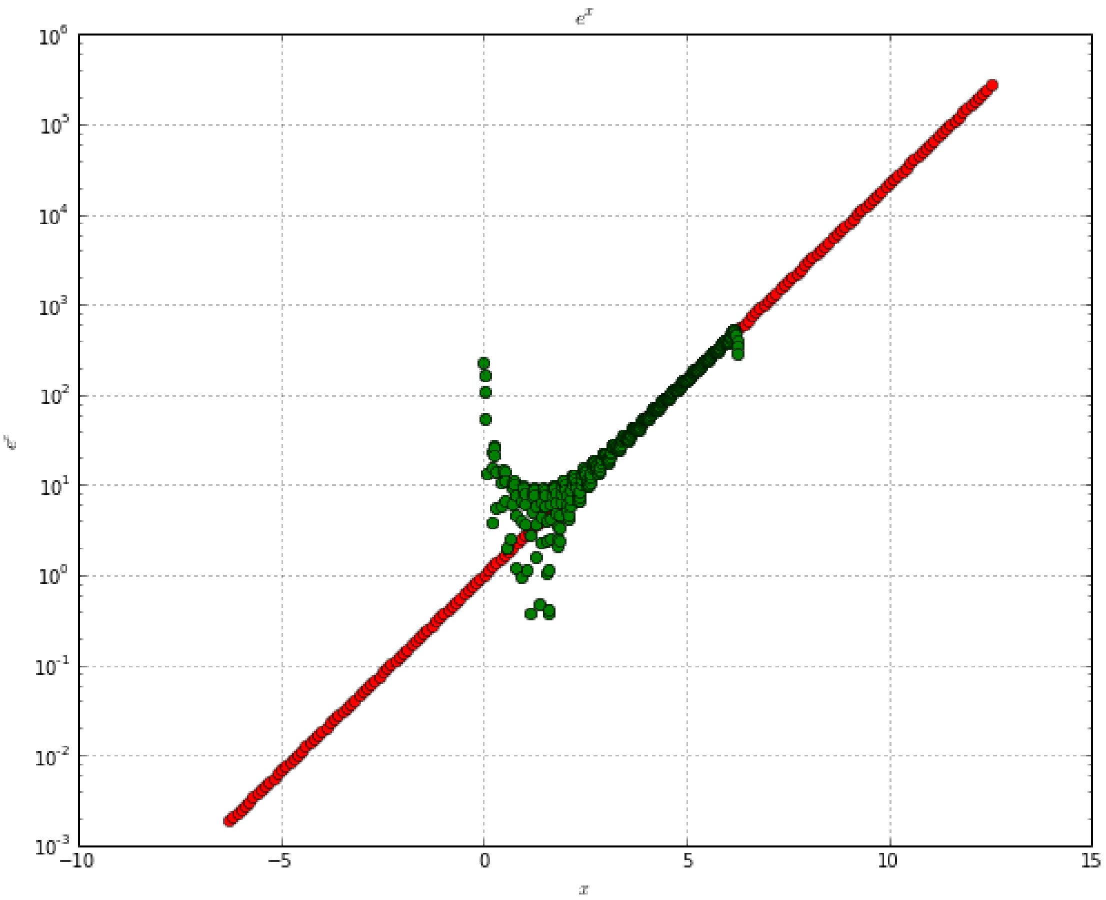


```
In [50]: print max(abs(c_exp-coffexp))
         print max(abs(c_cc-coffcc))
```

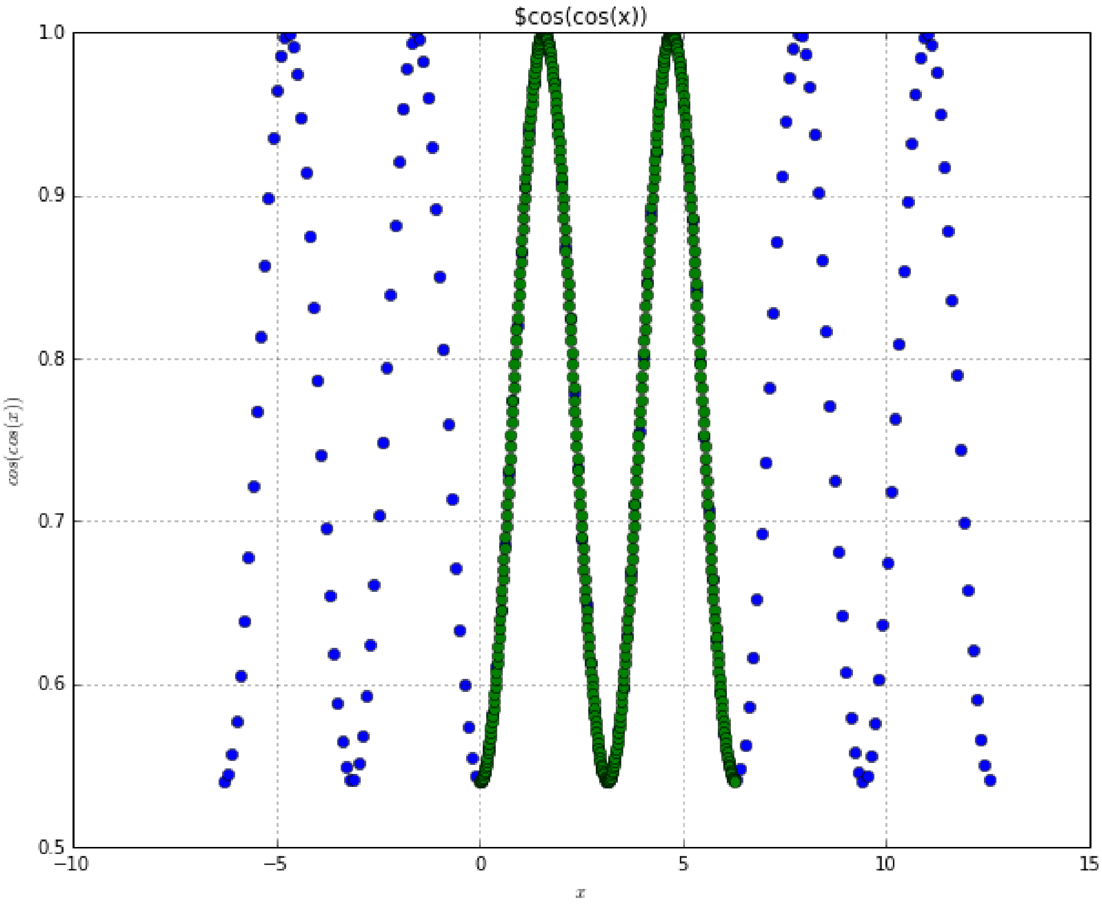
```
57.322959969
0.521627698042
```

```
In [92]: expvalues= np.dot(A,c_exp)
         graph= axes1.semilogy(x, expvalues, 'go')
         fig1
```

```
   Out[92]:
```



```
In [93]: ccvalues= np.dot(A,c_cc)
         graph=axes2.plot(x, ccvalues, "go")
         fig2

   Out[93]:
```


$cos(cos(x))
1.0
0.9
0.8
0.7
0.6
0.5
$cos(cos(x))$
−10
−5
0
5
10
15
x