APPLIED PROGRAMMING LAB (Week - 4) Bessel Index Approximation

Shubhangi Ghosh
EE15B129
Department of Electrical Engineering

February 7, 2017

1 INTRODUCTION:

The bessel function of the first type is denoted by $J_v(x)$, and is often seen in cylindrical geometry problems. For large x,

$$J_v(x) \simeq \sqrt{\frac{2}{\pi x}} cos(x - \frac{v\pi}{2} - \frac{\pi}{4}) \tag{1}$$

In this report, we compute the phase, $\phi = \frac{v\pi}{2} + \frac{\pi}{4}$, associaed with this approximation of the Bessel function and then estimate v, which should ideally be equal to 1.
We do the above by two different models, described as follows.

1.1 Models:

The models used to estimate Bessel function in this code are:

1. **Model 1:**

$$Acos(x_i) + Bsin(x_i) \simeq J_1(x_i) \tag{2}$$

We estimate phase, ϕ as:

$$\phi = \frac{A}{\sqrt{A^2 + B^2}} \tag{3}$$

Also,

$$\phi = \frac{v\pi}{2} + \frac{\pi}{4} \tag{4}$$

$$\therefore v = \frac{2}{\pi}(\phi - \frac{\pi}{4}) \tag{5}$$

2. **Model 2:**

$$A\frac{cos(x_i)}{\sqrt{x_i}} + B\frac{sin(x_i)}{\sqrt{x_i}} \simeq J_1(x_i) \tag{6}$$

   We estimate phase, ϕ as:

$$\phi = \frac{A}{\sqrt{A^2 + B^2}} \tag{7}$$

   v is estimated the same way.

2 BRIEF CODE EXPLANATION:

2.1 CODE:

```
#Importing header files
import numpy as np
from scipy import *
from matplotlib.pyplot import *
from scipy.integrate import quad
from math import *
from scipy.special import jv
#Extracting subvector starting from x0 from a vector x
#x is an increasing vector
def subvec(x,x0):
v1 = x[np.where(x>=x0)] #indices of x where value >= x0 are extracted
v2 = jv(1,v1) #computing bessel fn for extracted vector
return v1,v2 #returning extracted vector and its bessel fn
#returns cos fn divide by sqrt(x)
def cos_with_amp(x):
return np.cos(x)/x**(0.5)
#returns sin fn divide by sqrt(x)
def sin_with_amp(x):
return np.sin(x)/x**(0.5)
#adds randomised noise to passed vector
def vec_with_noise(x,eps):
x = x+eps*randn(len(x)) #noise*randomised vector of length of x
return x
#Function to compute 'v' as in given expression in question
def calcnu(x,x0,c,eps,model):
v = [] #v array
err = [] #error array
print "A and B for model %c with noise = %f:\n" %(model,eps)
```

```
print "A\t\t\tB\n" #printing out header for values of A and B as computed in expression
for i in x0:
b = subvec(x,i)[1] #extracted subvector from linerly spaced vector
b = vec_with_noise(b,eps) #adding noise
A = zeros((len(subvec(x,i)[0]),2)) # allocate a matrix A with zeroes
if model=='b':
A[:,0] = np.cos(subvec(x,i)[0]) #first column is cos
A[:,1] = np.sin(subvec(x,i)[0]) #first column is sin
elif model=='c':
A[:,0] = cos_with_amp(subvec(x,i)[0]) #first column is cos with amplitude 1/sqrt(x)
A[:,1] = sin_with_amp(subvec(x,i)[0]) #first column is sin with amplitude 1/sqrt(x)
a = np.linalg.lstsq(A,b)[0][0] #a is first val of lst sq vector
b = np.linalg.lstsq(A,b)[0][1] #b is second val of lst sq vector
print "%7.6f\t%7.6f\n" %(a,b) #printing out values of A and B as computed in expression
phi = math.acos(a/(a**2+b**2)**(0.5)) #calculating phase from a and b: a/sqrt(a^2 + b^2
vi = (phi-math.pi/4)*(2/math.pi) #phi = v*pi/2 + pi/4
v.append(vi) #v vector
err.append(np.fabs(1-vi)) # error
#plotting v
plot(x0,v,c)
return err #returning error
#Function to plot 'v' i.e Bessel Index for different no. of measurements in same range
def bessel_idx(num):
x = linspace(0,20,num) #linearly spaced vector
diff = 20.0/(num-1) #linear difference
x0 = arange(diff, 18+diff, diff) #points to extract subvector from
title('Estimation of Bessel index $v$')
xlabel('$x_0$')
ylabel('$v$')
#computaion and plots acc to diff models
err1=calcnu(x,x0,'bo',0,'b')
err2=calcnu(x,x0,'go',0,'c')
err3=calcnu(x,x0,'ro',0.01,'c')
legend(('$\epsilon=0$, model (b)','$\epsilon=0$, model (c)','$\epsilon=1.0e-02$, model
savefig('v_bessel'+str(num)+'.pdf',format='pdf') #save figure as pdf
show()
title('Error in Estimation of Bessel index $v$')
xlabel('$x_0$')
ylabel('Error')
#computaion and plots of error acc to diff models
plot(x0,err1,'bo')
plot(x0,err2,'go')
plot(x0,err3,'ro')
legend(('$\epsilon=0$, model (b)','$\epsilon=0$, model (c)','$\epsilon=1.0e-02$, model
savefig('err_bessel'+str(num)+'.pdf',format='pdf') #save figure as pdf
show()
```

```
#printing maximum error for all models
print "Max error for model(b) without noise: %f" %(amax(err1))
print "Max error for model(c) without noise: %f" %(amax(err2))
print "Max error for model(c) with noise=0.01: %f" %(amax(err3))
#Main Function
bessel_idx(41) #bessel for 41 points
bessel_idx(61) #bessel for 61 points
bessel_idx(81) #bessel for 81 points
```

2.2 FUNCTIONS USED IN CODE:

1. $subvec(x, x0)$:
   This function extracts a subvector from the increasing vector x, with all the elements $\geqslant x0$.

2. $cos\_with\_amp(x)$:
   The function returns $\frac{cos(x)}{\sqrt{x}}$.

3. $sin\_with\_amp(x)$:
   The function returns $\frac{sin(x)}{\sqrt{x}}$.

4. $vec\_with\_noise(x, eps)$:
   This function adds a randomised noise to the vector x. The vector returned by this function is:

   $$x = x + eps * randomised\, vector\, of\, length(x)$$

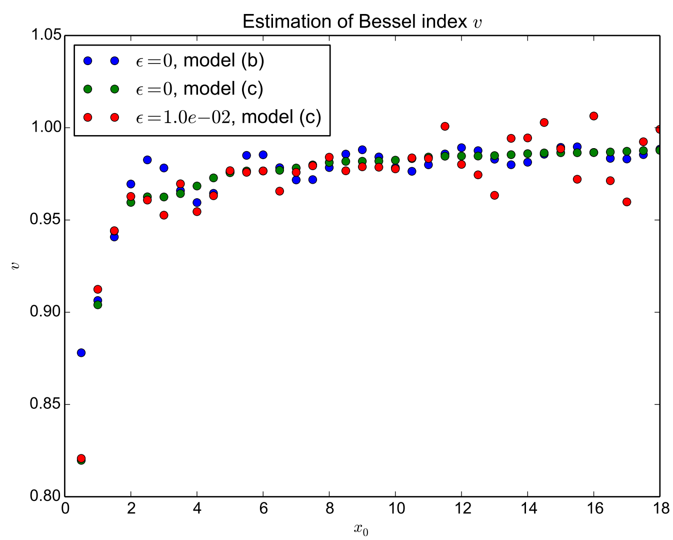
5. $calcnu(x, x0, c, eps, model)$:
   The arguments of this function are:
   (i) $x-$The vector over the range of which Bessel function is computed.
   (ii) $x0-$The vector which contains the points at which subvector is extracted.
   (iii) $c-$The colour specification of the plot of $v\, vs.\, x0$.
   $(iv)$$eps-$The amount of noise to be added to input vector
   (v) $model-$The model used to compute A, $B\, and\, v$.

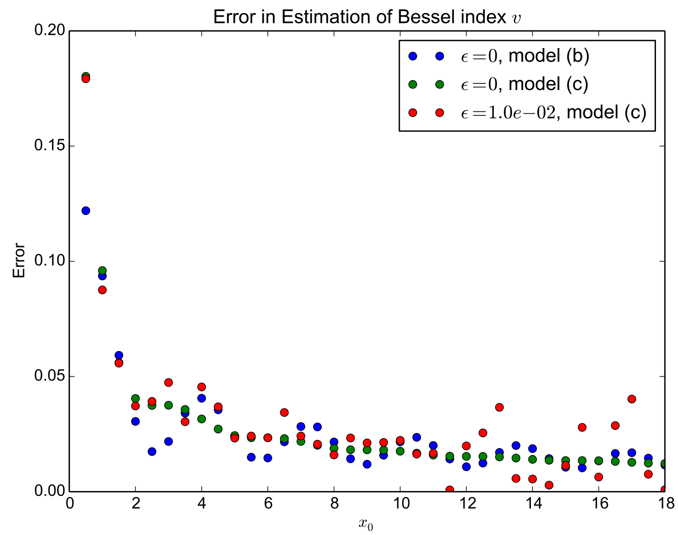
   This function:
   (i)Extracts subector from x for all elements of $x0$.
   (ii)Creates a matrix A, with first column as $cos(x_i)$ and second column as $sin(x_i)\forall\, x_i$in the subvector.
   (iii)Computes A and B according to the model passed as function argument.
   (iv)Computes phase, ϕ, according to the given model.
   (v)Computes v,from ϕ as $v = \frac{2}{\pi}(\phi - \frac{\pi}{4})$ as in Eqn.(4).
   (vi)Prints A and B for different $x0$.
   (vii)Plots $v\, vs.\, x0$,according to given colour specification.
   $(viii)$Computes error for all $x0$ and returns error array.

6. $bessel\_idx(num)$:
   This function:
   (i)Creates x, a linearly spaced array from 0 to 20 with num samples.
   (ii)Creates $x0$, a linearly spaced vector from $(0, 18]$ with same linear spacing as x.This contains the points from which subvector is extracted.
   (iii)Plots estimated bessel index, v $vs.x0$, and error $i.e.1 - v$ $vs.x0$ for model (b) without noise, model (c) without noise and model (c), with noise $= 0.01$.

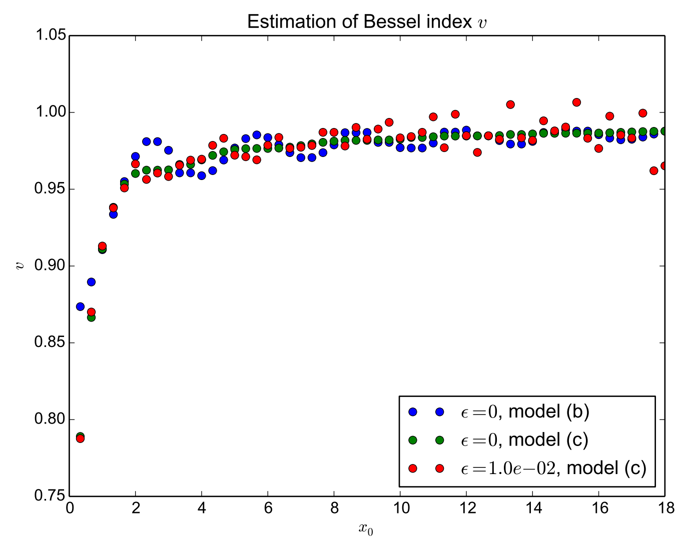
2.3 PLOTS:

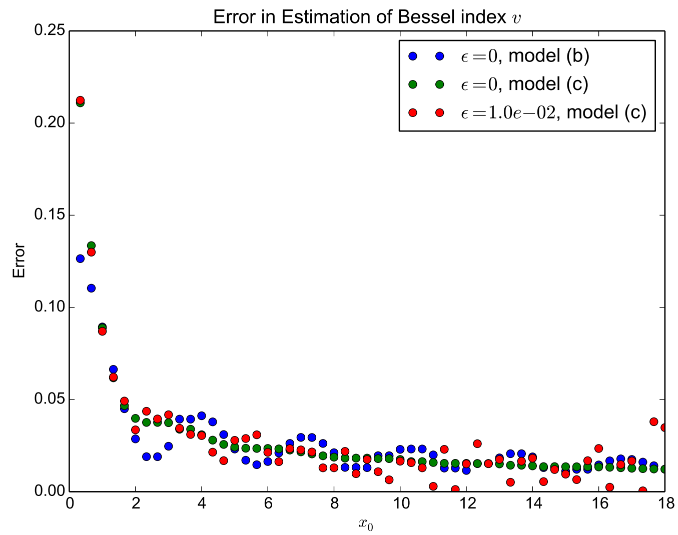
2.3.1 Plot of Bessel Index $v\, vs.\, x0$ and $error\, vs.\, x0$ for 41 samples in [0,20]



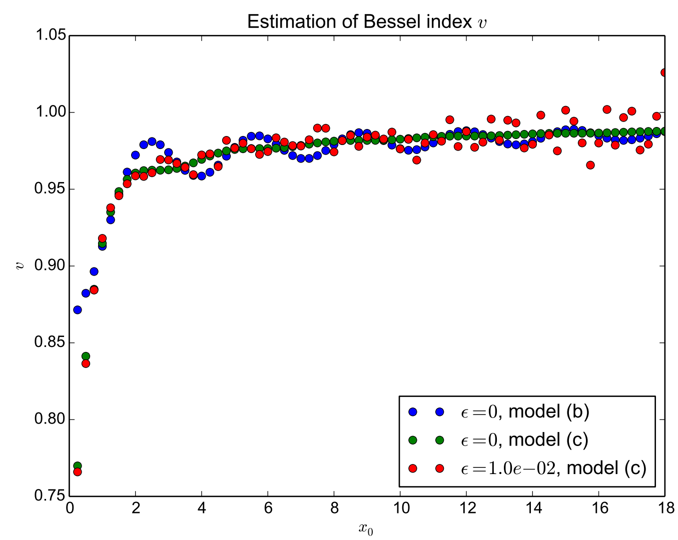


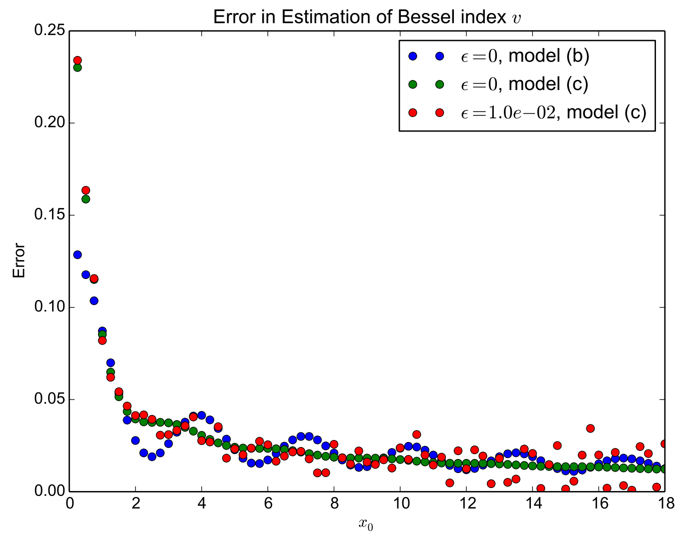
2.3.2 Plot of Bessel Index $v\, vs.\, x0$ and $error\, vs.\, x0$ for 61 samples in [0,20]





2.3.3 Plot of Bessel Index $v\, vs.\, x0$ and $error\, vs.\, x0$ for 81 samples in [0,20]





2.4 ERROR ANALYSIS:

2.4.1 41 SAMPLES

1. Max error for model(b) without noise: 0.121958
2. Max error for model(c) without noise: 0.180269
3. Max error for model(c) with noise=0.01: 0.171130

2.4.2 61 SAMPLES

1. Max error for model(b) without noise: 0.126442
2. Max error for model(c) without noise: 0.210968
3. Max error for model(c) with noise=0.01: 0.209321

2.4.3 81 SAMPLES

1. Max error for model(b) without noise: 0.128559
2. Max error for model(c) without noise: 0.230132
3. Max error for model(c) with noise=0.01: 0.233080

3 RESULTS AND DISCUSSION:

1. Since the approximation in Eqn. (1) is only valid for large x, we see that as we move towards a larger value of $x0$, the estimation of Bessel index v,nears the expected value, $i.e.1$.

2. But on adding noise, if we estimate the Bessel index only for the larger samples, we get a biased and erroneous result due to fewer number of samples being considered.

3. Model (c) , where $sin(x_i)$ and $cos(x_i)$ are scaled with amplitude $\frac{1}{\sqrt{x_i}}$ is more accurate than Model (b) as we see from the plots a more consistent and less erroneous output.

4. Increased number of measurements helps us estimate the error better, since with lesser measurements we may miss some points where the value of error is higher.

5. At higher values of x_0,we'd get a more accurate value of Bessel Index, v,if it were possible to take measurements noiselessly. After adding noise, the Bessel index estimation at higher values of $x_0$is affected more due to lower number of samples, thereby giving biased output. Hence, we should attempt reach a balanced value of x_0, which is large enough such that the Bessel index approximation from Eqn. (1) is valid and also we have enough samples such that the effect of noise is negated.