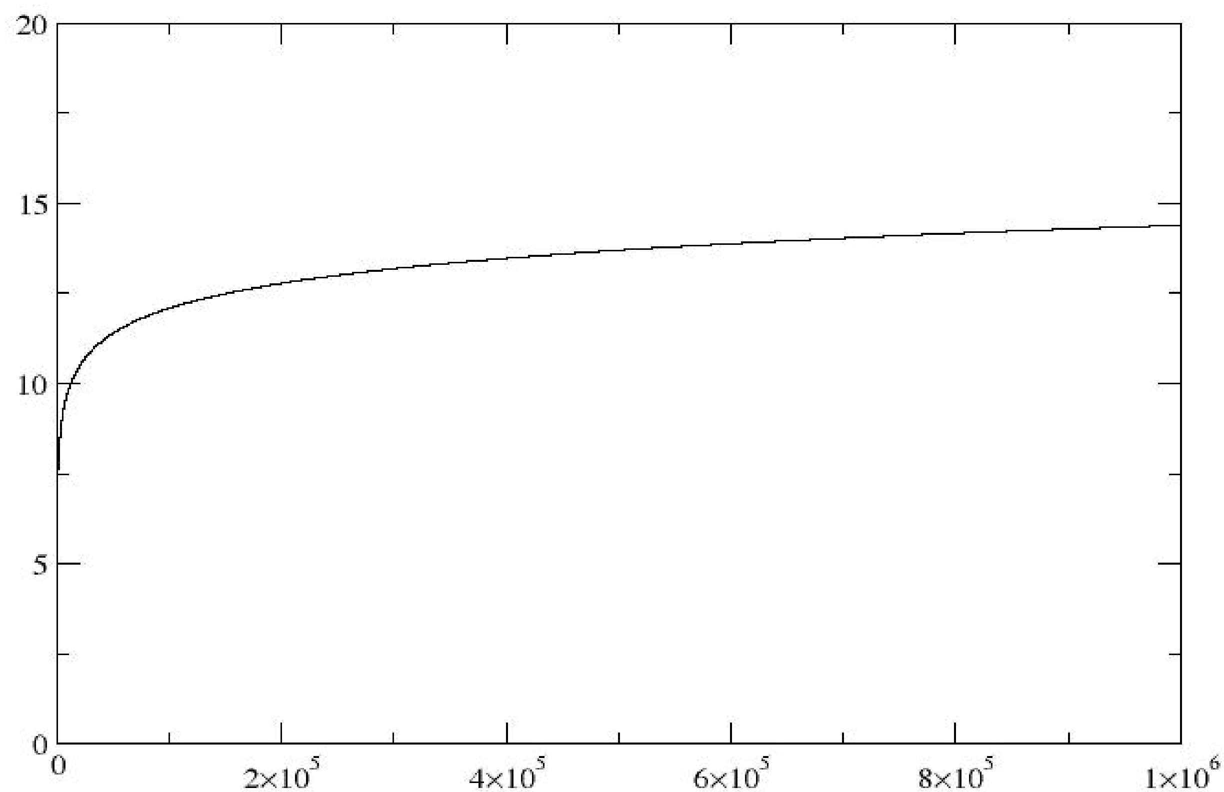
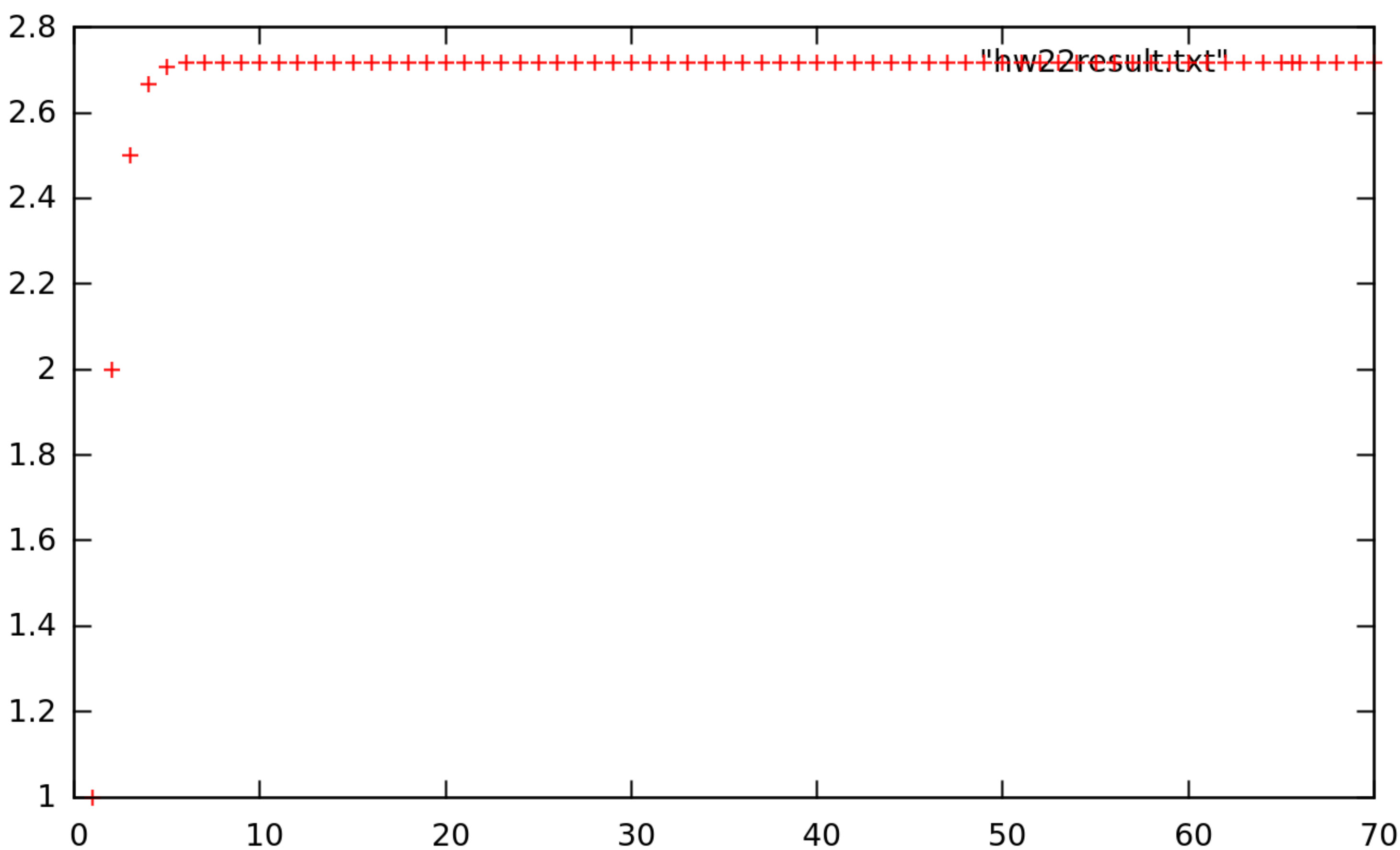
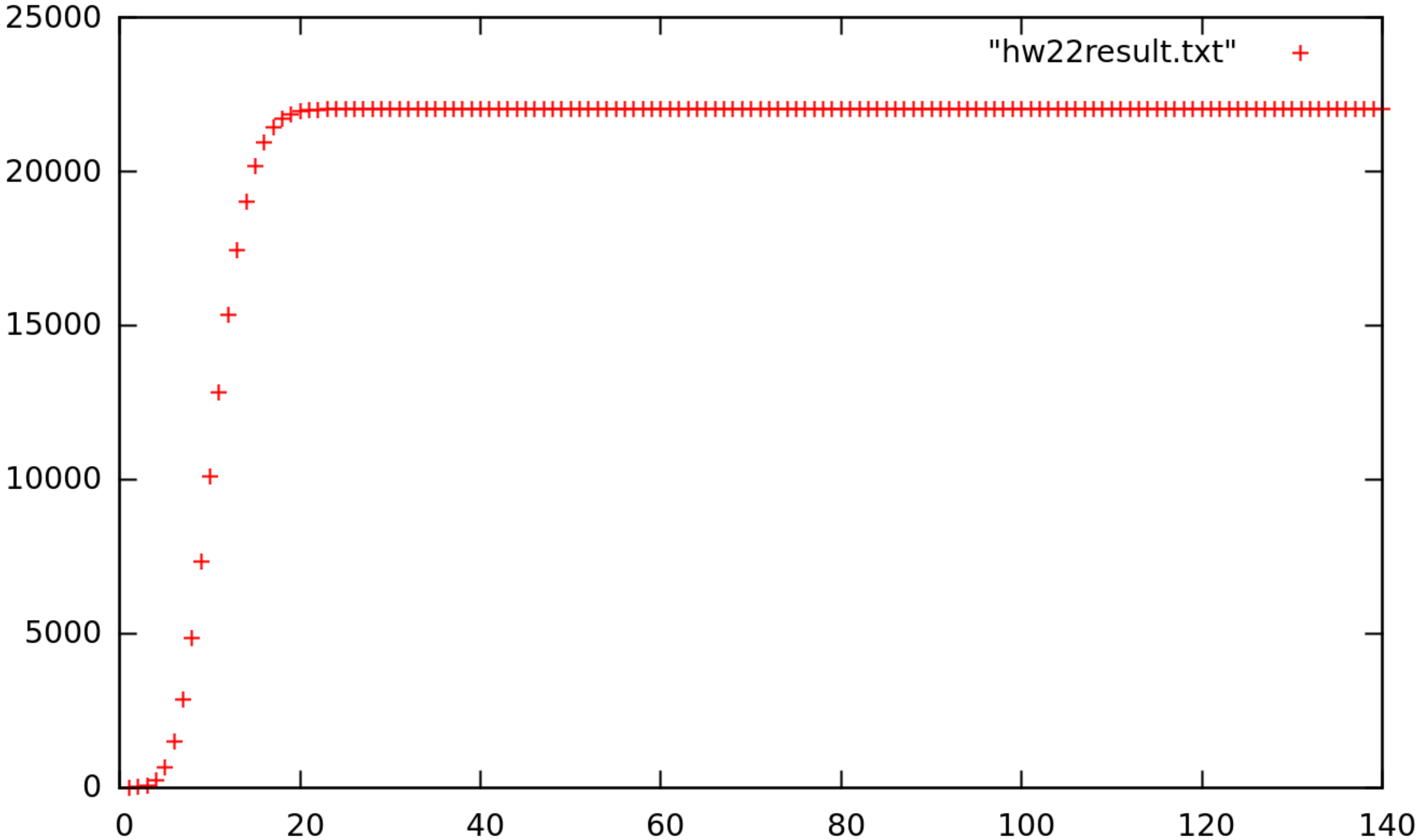
```
# sigam(1/x) when x goes to positive infinity
# Farhad Ramezanghorbani

step=1                                  # set step to 1
sigma=0.0                               # set sigma of 1/x to 0
error=10**(-6)                          # def err (which is subtraction-
                                        # of two consequent sigma)
result=open("hw21result.txt","w")       # open an empty .txt for writing the result
def div(x):                             # define division of 1/x as function
	return 1.0/x
while 1:                                # open an infinite loop to calculate-
	sigma+=div(step)                    # sigma until the subtraction of two-
	if div(step)<=error:                # consequent sigma is equal to defined err
		break
	output=str(sigma)+"  "+str(step)+"\n"    # output as two columns (sigma vs #step)
	result.write(output)                # write result in txt file
	step+=1

print sigma

# as we can see Sigma(1/x) goes to infinity when x goes to infinity but the slope of graph-
# will decrease after 10*5 steps-
# I have defined the error in which the infinit loop of while will stop when the division of-
#two consequent sigma is less than 10 to the power of -6.
```

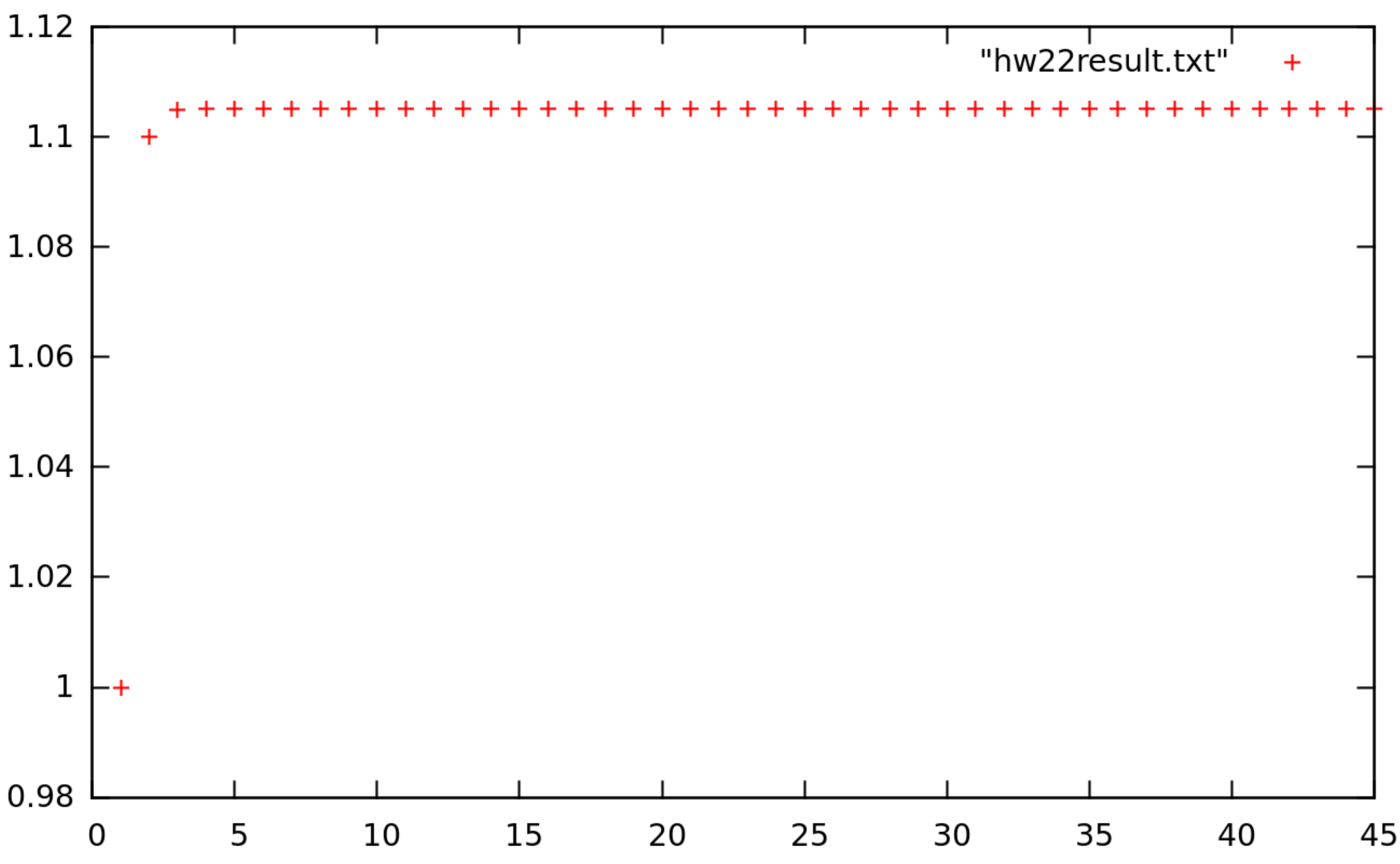


```
# Taylor expansion of e**x in 0 neighborhood
# for different inputs of x

import math                                   # call the math functions
n=0.0                                         # set counter of infinite loop
expo=0.0                                      # set initial # of expo of x to 0
error=10**(-100)                              # set error for breaking the infinite loop
x=float(raw_input("enter x of e**x "))        # ask x from user

result=open("hw22result.txt","w")             # open an empty txt for writing the results

def div_fact(x):                              # define div_fact function which is-
    return float(x**n/math.factorial(n))      # each term of Taylor expansion

while 1:                                      # open an infinite loop
    expo+=div_fact(x)                         # add each term to expo of x till-
    if div_fact(x)<=error:                    # the difference between two consequence term-
        break                                 # is less than error
    n+=1                                      # go to next term of Taylor expansion
    output=str(n)+"      "+str(expo)+"\n"     # add number of terms vs expo in two column
    result.write(output)                      # write the outputs to txt file
```

as we can see in the first plot the out put of Taylor expansion will goes to near real number of e**1 after 6 or 7 terms of the expansion
for x=1 but when we enter 10 for x, what we can see is Taylor expansion needs more terms (about 17-18) to be near the e**10
so when we write Taylor in 0 neighborhood, for getting the best output our x should be near zero, as you
can see in third graph, when we enter 0.1 we will be near e**(0.1) after 3 or 4 term.




"hw22result.txt"
25000
20000
15000
10000
5000
0
0
20
40
60
80
100
120
140


"hw22result.txt"
1.12
1.1
1.08
1.06
1.04
1.02
1
0.98
0
5
10
15
20
25
30
35
40
45