

**NPTEL  
NATIONAL PROGRAMME ON  
TECHNOLOGY ENHANCED LEARNING**

**IIT BOMBAY**

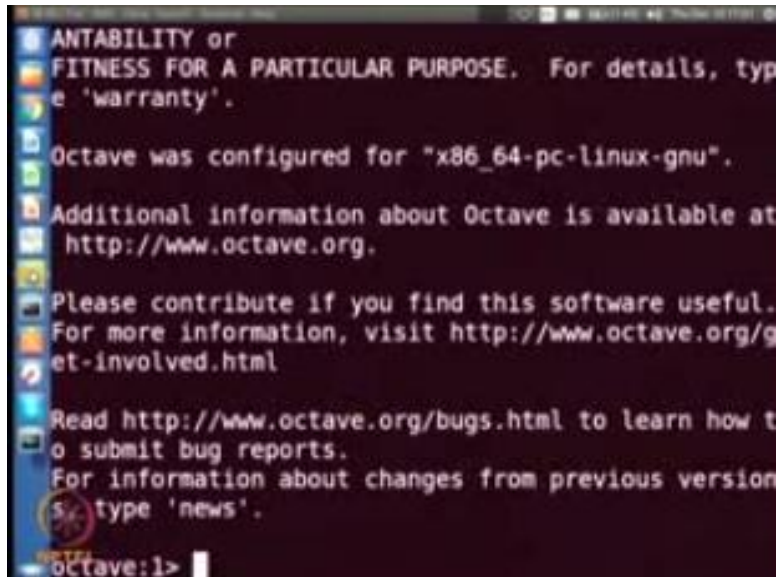
**CDEEP  
IIT BOMBAY**

**Phase field modeling;  
the materials science,  
mathematics and  
computational aspects  
Prof. M P Gururajan  
Department of Metallurgical Engineering  
and materials Science, IIT Bombay**

**Module No.5  
Lecture No.23  
GNU Octave:  
Interactive mode**

Welcome so we are trying to work with octave I showed you how to use octave like a calculator now we are going to go one step further and we are going to see how to use obtain both as a calculator and as a plotter this is the program which can also plot and show to do that of course we have to go and invoke octave so this time I am going to do it from the icon.

(Refer Slide Time: 00:44)



```
ANTABILITY or
FITNESS FOR A PARTICULAR PURPOSE. For details, type
'e 'warranty'.

Octave was configured for "x86_64-pc-linux-gnu".

Additional information about Octave is available at
http://www.octave.org.

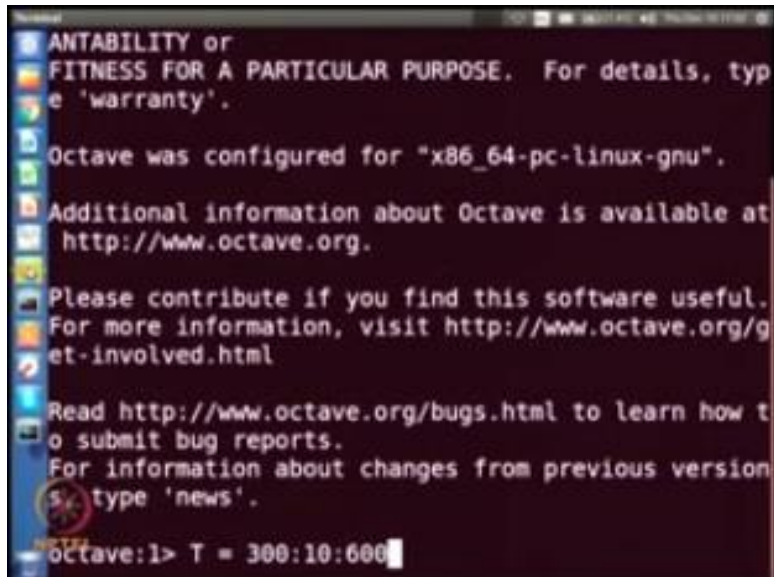
Please contribute if you find this software useful.
For more information, visit http://www.octave.org/get-involved.html

Read http://www.octave.org/bugs.html to learn how to
submit bug reports.
For information about changes from previous versions,
type 'news'.

octave:1>
```

So you go find out the octave I can double click it got up there I wanted to calculate the vacancy concentration in aluminum for temperature range 300 to 600 and I wanted to calculate it for every  $10^0$ , so first I am going to define a variable called temperature T and I am going to define it to be a vector it will go from 300 to 600 it will go 300, 320, 330, 340, 350, etcetera up to 600 but how do I do that here is how you do that.

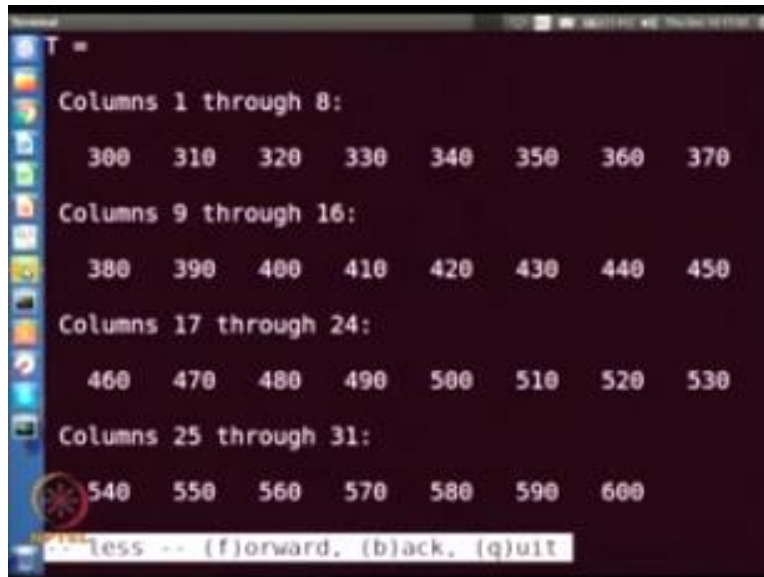
(Refer Slide Time: 01:20)



```
ANTABILITY or  
FITNESS FOR A PARTICULAR PURPOSE. For details, typ  
e 'warranty'.  
Octave was configured for "x86_64-pc-linux-gnu".  
Additional information about Octave is available at  
http://www.octave.org.  
Please contribute if you find this software useful.  
For more information, visit http://www.octave.org/g  
et-involved.html  
Read http://www.octave.org/bugs.html to learn how t  
o submit bug reports.  
For information about changes from previous version  
s, type 'news'.  
octave:1> T = 300:10:600
```

So you first say T that is my variable name is equal to 300 then colon I say increase in steps of 10 till what point till it reaches 600 so I am going to enter.

(Refer Slide Time: 01:36)



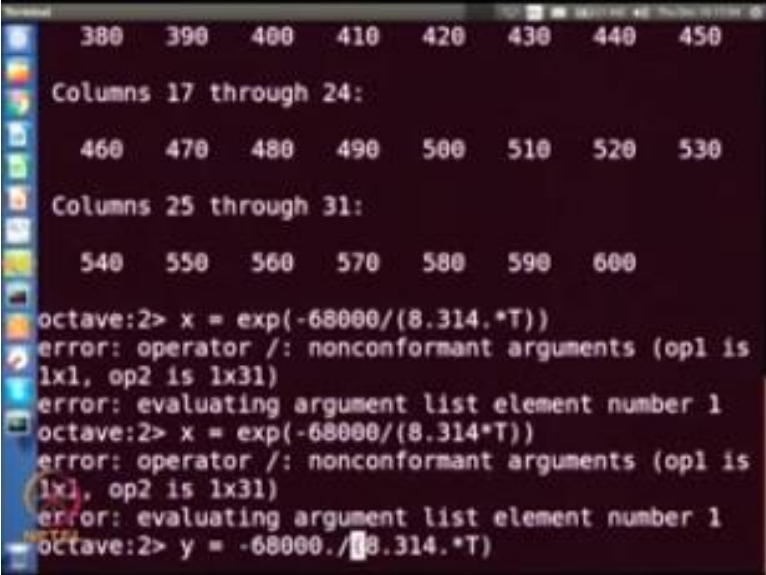
```
T =  
Columns 1 through 8:  
 300  310  320  330  340  350  360  370  
Columns 9 through 16:  
 380  390  400  410  420  430  440  450  
Columns 17 through 24:  
 460  470  480  490  500  510  520  530  
Columns 25 through 31:  
 540  550  560  570  580  590  600  
less -- (f)orward, (b)lack, (q)uit
```

So you can see it has columns and the columns are worth 300, 310, 320, 333, 340, etcetera and okay let us go forward so it goes up to 600 right so if I say Q now it will come out so I have temperature now stored from 300 to 600 in steps of 10 now I want to calculate the vacancy concentration let me call the vacancy concentration as X, X is equal to what exponential this is where I said you know when we looked up the exponential it said that it will calculate it for a matrix or vector.

This is what it means I have temperature stored as a vector and when I say exponential of that quantity then it is going to do for every component of that vector okay so that is what we are going to do how do I calculate exponential right of course  $e^{-68000/8.314}$ , it is always a good idea to put this parenthesis first and write the remaining command so that you will never get syntax errors you know that you open parenthesis and you have not closed it so you will not get in and then I am going to multiply the star is for multiplication.

If you say dot star then that means multiply for each of the components of the given vector and that is T for us right so let us see what happens.

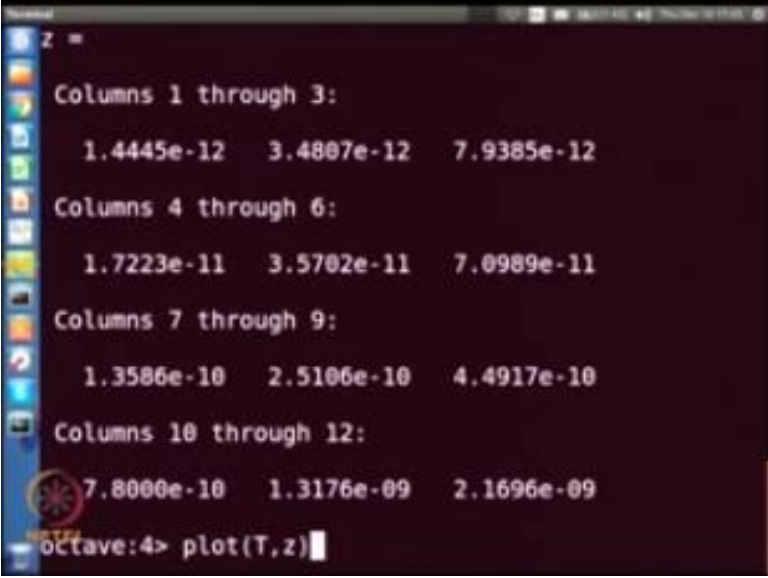
(Refer Slide Time: 03:09)



```
380 390 400 410 420 430 440 450
Columns 17 through 24:
460 470 480 490 500 510 520 530
Columns 25 through 31:
540 550 560 570 580 590 600
octave:2> x = exp(-68000/(8.314.*T))
error: operator /: nonconformant arguments (op1 is
1x1, op2 is 1x31)
error: evaluating argument list element number 1
octave:2> x = exp(-68000/(8.314*T))
error: operator /: nonconformant arguments (op1 is
1x1, op2 is 1x31)
error: evaluating argument list element number 1
octave:2> y = -68000./[8.314.*T]
```

If I enter okay so some non conformant arguments okay I do not understand what it is let me see maybe have made a mistake here no I cannot do this okay so I am going to do this so let me make a temporary variable let us see if this works this is  $-68000/8.314 \cdot T$ , let us see if this works.

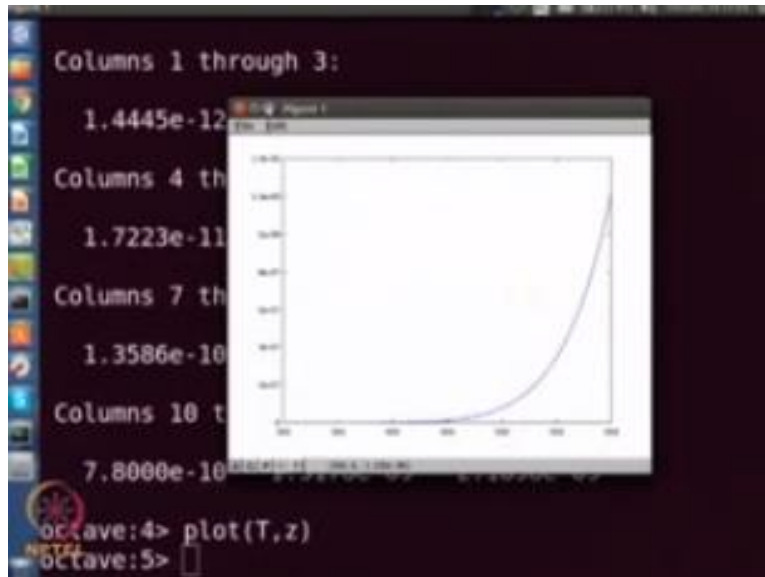
(Refer Slide Time: 03:45)



```
Z =  
Columns 1 through 3:  
    1.4445e-12    3.4807e-12    7.9385e-12  
Columns 4 through 6:  
    1.7223e-11    3.5702e-11    7.0989e-11  
Columns 7 through 9:  
    1.3586e-10    2.5106e-10    4.4917e-10  
Columns 10 through 12:  
    7.8000e-10    1.3176e-09    2.1696e-09  
octave:4> plot(T,z)
```

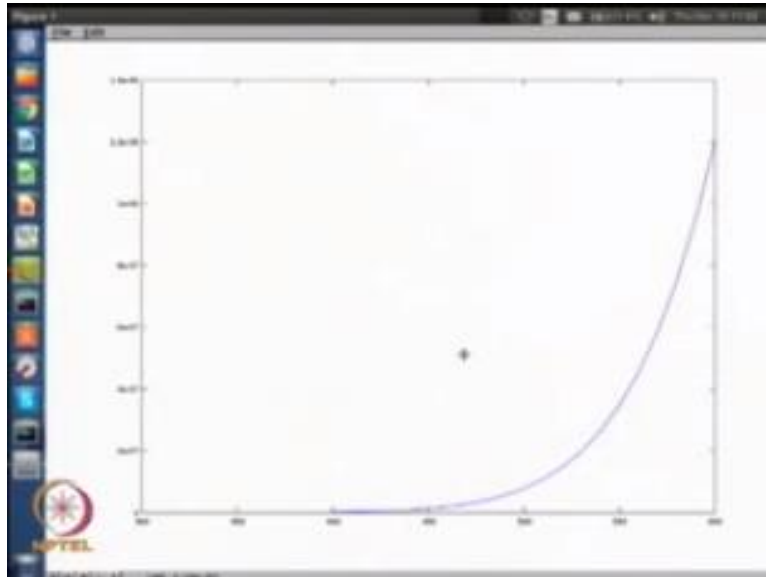
Okay this works very good, so we now have a vector Y which basically calculated  $-\delta h/rt$ , very good so now I say that some quantity z is nothing but exponential (Y), us see if this works well this works very good okay now what do I do I need to plot which is the x axis that is temperature which is the Y axis that is Z.

(Refer Slide Time: 04:18)



So let me enter I have the vacancy concentration you can see that the vacancy concentration is very low of course it increases exponentially because that is what the functional dependence we saw right.

(Refer Slide Time: 04:30)



So I hope you are able to see the numbers I do not know if they are clearly visible so it starts from somewhere 0 300 is very, very small it is not quite 0 you saw that it is 10 power minus 12 or something but this is five orders of magnitude high, so by the time you reach something like 525 or something you are in  $10^{-7}$  and by the time you reach something like 600 you are in  $10^{-6}$ , so from -12 to -6, so 6 orders of magnitude your vacancy concentration has increased of course this is exponential dependence.

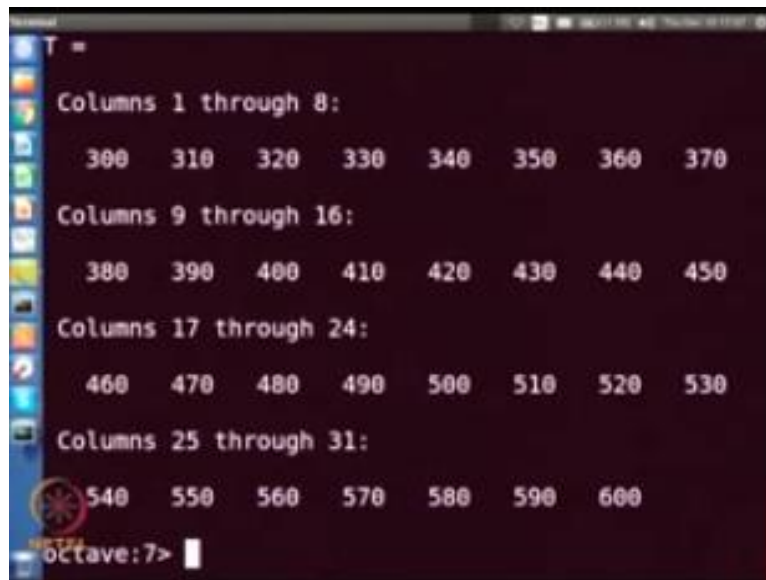
Because that is what the dependence that we are trying to plot okay so you can use octave you can do calculation, calculations which you would have thought or if you had programmed in a programming language like C for example first you have to generate a loop and the loop has to go in steps of 10 and then you have to calculate that value for everything then you have to store them and then that has to be written to a file in a format which some plotting program can read and then plot okay.

So that is how you would have done it if you didn't have access to simplify the program like okay so it is very high level as you can see it is also interactive what it means is that if I type something it immediately tells me what the answer is right I say from 300 to 600 you go in steps



of 10 then it immediately shows me what all these steps it is going to take but there is away to avoid that sometimes I do not want things onto the screen. So I will show you a small trick which helps avoid printing to the screen and that is something like this okay.

(Refer Slide Time: 06:08)

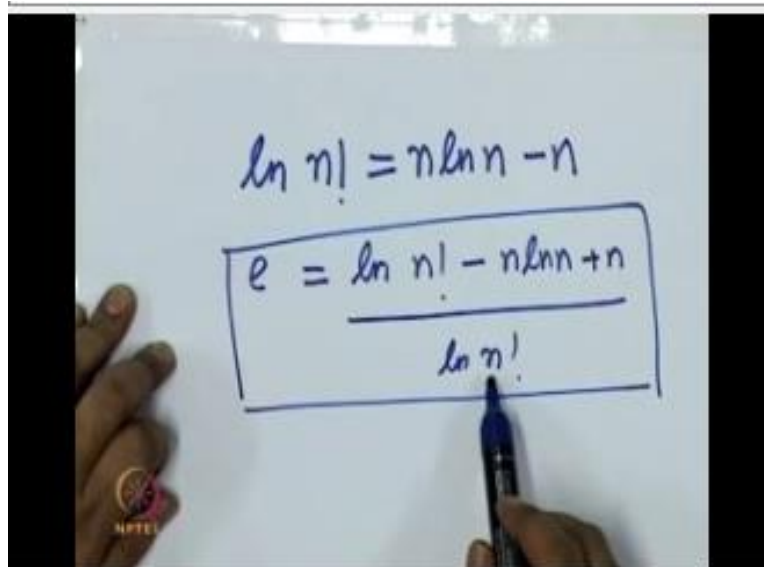


```
T =  
Columns 1 through 8:  
  300  310  320  330  340  350  360  370  
Columns 9 through 16:  
  380  390  400  410  420  430  440  450  
Columns 17 through 24:  
  460  470  480  490  500  510  520  530  
Columns 25 through 31:  
  540  550  560  570  580  590  600  
octave:7>
```

So if I say 300 to 600 go in steps of 10 but do not show me the output on the screen so you just have to put a semicolon at the end if you say semicolon and then enter then it will not show anything on the screen but it has produced this T okay, so if you just say T and then enter it will show you that this is what it is stored ok so the semicolon basically suppresses output to the screen sometimes we do not want to see all these big things on the screen so you can put a semicolon and forget about it so it will not show you on the screen okay.

Now very good so we can do a small exercise you remember the Sterling approximation that we looking at earlier the sterling approximation said that.

(Refer Slide Time: 07:00)

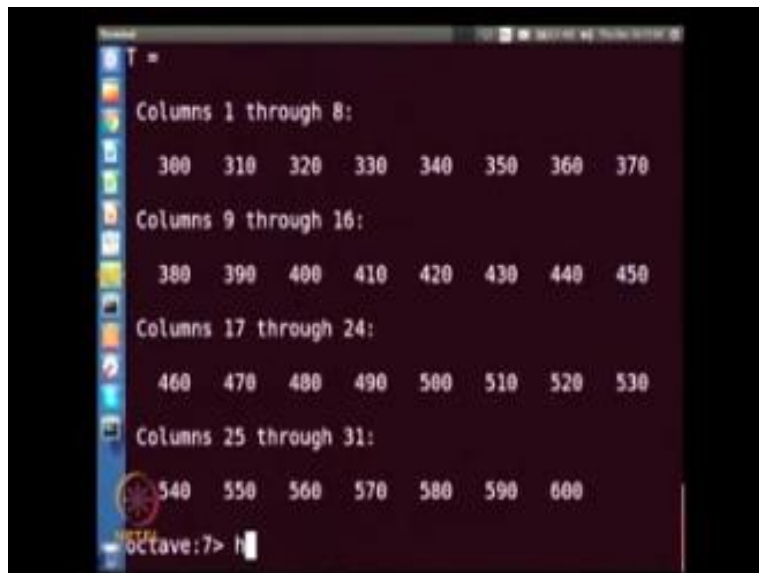

$$\ln n! = n \ln n - n$$
$$e = \frac{\ln n! - n \ln n + n}{\ln n!}$$

Logarithm of N! It what did it say logarithm of n! n log n-n right, we said that this approximation is good for very large numbers and things like that so let me find out how good this approximation is okay, so we are going to take some numbers from 2 to 20 and see how the approximation is performed for doing that you need to know what the factorial is right you need to calculate for 2! 3! 4! Etcetera up to 20! You have to take logarithm on that and you have to compare that with this number.

So the way I am going to compare is that I am going to define a quantity called error the error is something like this it is a relative error so I am going to take  $\log n! - n \log n + n / \log n!$  because logarithm of n! is the actual value, ok this is approximation so when I make this approximation relatively how much error am I making is what my error is going to be, okay so this is the quantity I want to do and I want to plot this error as a function of n, okay.

So I want to have a plot of e versus n this is what I want to do, well we know how to do that provided we know how to do the factorial calculation on up there okay. So my first step would be to ask for help from up there, and I am going to ask help factorial I hope L! Will give me help on factorial okay.

(Refer Slide Time 08:53)



```
T =  
Columns 1 through 8:  
 300  310  320  330  340  350  360  370  
Columns 9 through 16:  
 380  390  400  410  420  430  440  450  
Columns 17 through 24:  
 460  470  480  490  500  510  520  530  
Columns 25 through 31:  
 540  550  560  570  580  590  600  
octave:7> |
```

So how do I do that help let us ask for help on factorial right, very good so there is a function called factorial and they do factorial so it says written the factorial n, where n is a positive integer, okay so if n is a scalar then this is equivalent to product of 1 to n so of course you know that that is what! Is 3! is  $3 \times 2 \times 1$  right.

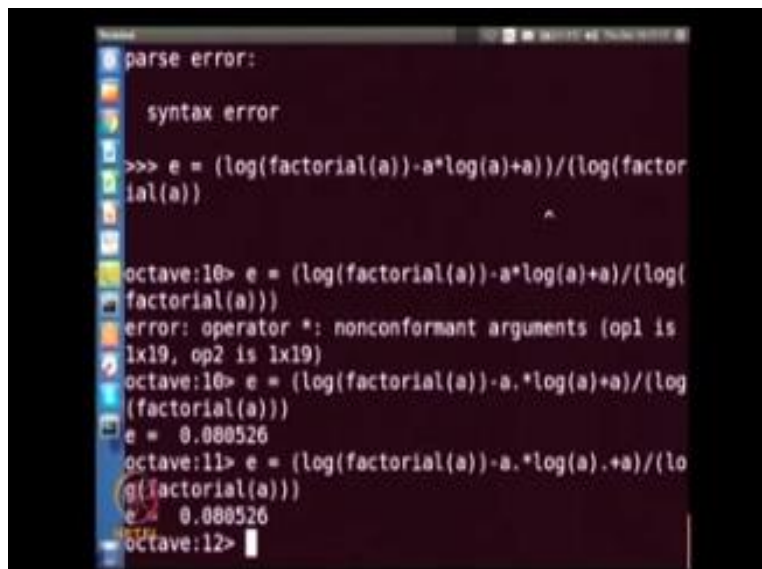
So for vector arguments also it works so now you know what vector argument is, and for non integers see the generalized factorial function  $\gamma$ , see this is one of the reasons why I like octave and is strongly recommend that you actually go through the manual and the all the available information, so you can actually learn up there as you are learning if you are paying attention to the manual and document and other information you will also learn lots of mathematics, and on top of it because this is the open source software you will even be able to see the code that generates this result.

So you will be able to learn a lot about algorithms and coding and everything ok so this is a one package, and this is a great thing about free open source software diet that they are great pedagogical implements, so they are very great from the point of view of learn, okay instead of you know somebody teaching you all the stuff they are available and, the worst that you can do

by experimenting with them is that maybe you will make your computer hang right nothing worse is going to happen.

So you can learn lots of information and lots of concepts and lots of material on your own, just by putting one of these programs on your computer and playing with them and trying to read about them and trying to learn about them, okay so very good, so we are going to do this factorial thing, so factorial n exists.

(Refer Slide Time 10:58)



```
parse error:
syntax error
>>> e = (log(factorial(a))-a*log(a)+a)/(log(factorial(a)))
octave:10> e = (log(factorial(a))-a*log(a)+a)/(log(factorial(a)))
error: operator *: nonconformant arguments (op1 is 1x19, op2 is 1x19)
octave:10> e = (log(factorial(a))-a.*log(a)+a)/(log(factorial(a)))
e = 0.080526
octave:11> e = (log(factorial(a))-a.*log(a).+a)/(log(factorial(a)))
e = 0.080526
octave:12>
```

So what am I going to do? I am going to first calculate first I am going to define a variable, let me call that variable as a what is a goes from 2: 20 okay, so as you can see previously I said 300:10: 600, so then it went from 300 to 600 insteps of 10, if I skip the middle column if I say 2 and 20, the initial and final alone I give then it increments by one by unit, so it goes 2 3 4 etcetera okay.

So if you are incrementing things by one which is the most default way of increasing numbers, then you do not have to give the increment value in between, in other words to is 2:20 is equivalent to giving it as 2 : 1 : 2 :20 okay so they are the same all right, so I have this variable I

called it as A and A is from 2 to 20, I can calculate the error let me define e equal to I am going to do this divided that, you know I always put the parentheses first so that I do not miss them log factorial ( a ) x - a × log of a + a no that is plus, you written log n - n is the approximation.

So minus if I do so and first one because minus a second one with a plus divided of course log! A it is quite complicated let us see how it works? did I forget something parentheses don't think so let us see, ha I did forget some parenthesis, okay so if you get to this then the way to come out of this is to put control C, okay so I did the forget some parentheses I think it's here ok.

So let us see if it works now, no there is an extra parenthesis so it says that there is some syntax error not able to figure out so let me rewrite it again, e is equal to divided by this, and this part is log ! a, one-run pieces I missed ok so log ! a ok – a, if you pay attention it actually shows me where the parentheses are, I mean maybe it does keep track of it for me and tell me okay.

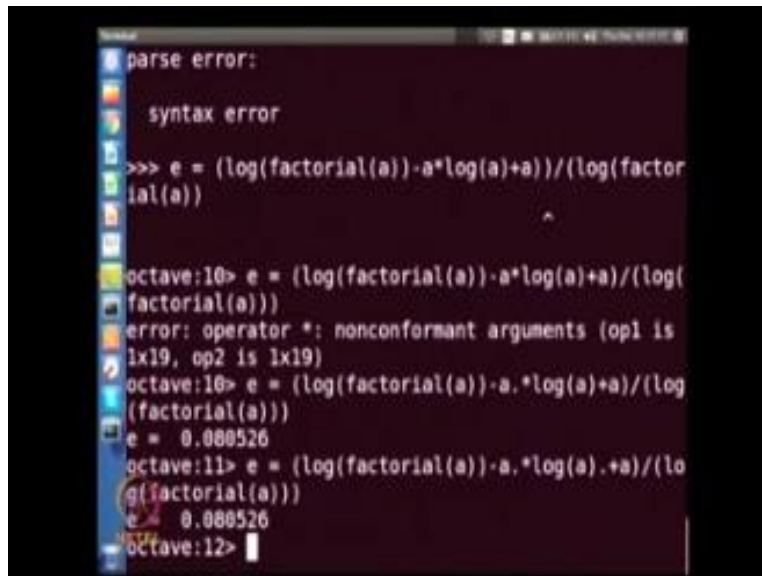
So let us see if it works? Okay so this is telling me the non-core conformant argument means, that you are dealing with a vector and you want to do this for every element of the vector in which case you cannot use star, you have to use dot star ok, so I am showing you this error messages because most of the times when you do calculations some sometimes you will get this kind of error messages, then you should be able to guess what the error could be and then correct it right.

So for example here it showed that this is where I made a mistake, because I thought that I Mr. parentheses I put it here and it is telling there is an error here, it says in touch syntax error parse error which means this parenthesis is extra, similarly here it is telling operator star non conformant argument operation 1 is 1 into 19, operation 2 is 1 in to 19 which means okay, so it is talking about some vector operators here and the dart is missing so that is what I understand.

Okay so we have error but this is not the forum in which I want, let me see what has no this is not the forum in which I want, I want the error for each of the A's, okay so I am going to go back and redo this calculation, the way I am going to do is like I did in the case of plotting the vacancy

concentration, so I am going to introduce two vectors I am going to deal with vectors, so let me first create the  $\log n !$ ,

(Refer Slide Time: 15:52)



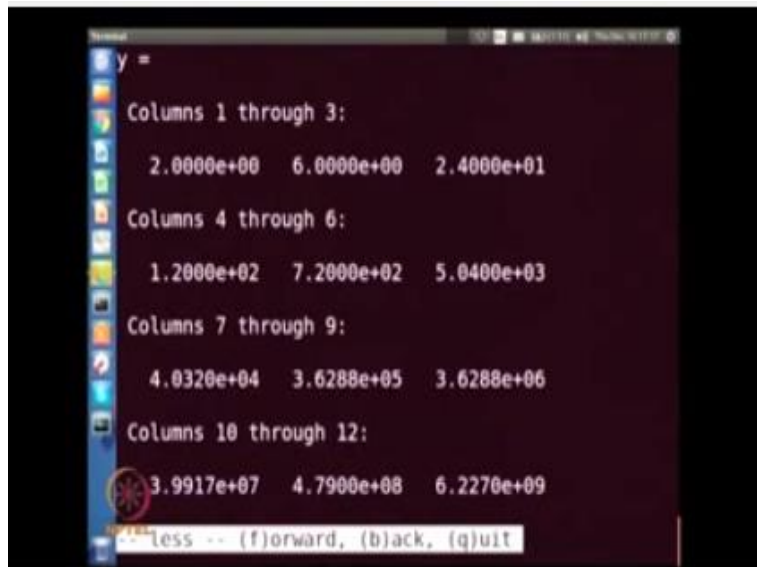
```
parse error:
syntax error
>>> e = (log(factorial(a))-a*log(a)+a)/(log(factorial(a)))
^
octave:10> e = (log(factorial(a))-a*log(a)+a)/(log(factorial(a)))
error: operator *: nonconformant arguments (op1 is 1x19, op2 is 1x19)
octave:10> e = (log(factorial(a))-a.*log(a)+a)/(log(factorial(a)))
e = 0.080526
octave:11> e = (log(factorial(a))-a.*log(a).+a)/(log(factorial(a)))
e = 0.080526
octave:12>
```

Okay so how do I do that let me call this  $Y = !a$  well now,  $y$  is a vector okay, so now I can say

(Refer Slide Time: 16:00)

```
parse error:
syntax error
>>> e = (log(factorial(a))-a*log(a)+a)/(log(factorial(a)))
^
octave:10> e = (log(factorial(a))-a*log(a)+a)/(log(factorial(a)))
error: operator *: nonconformant arguments (op1 is 1x19, op2 is 1x19)
octave:10> e = (log(factorial(a))-a.*log(a)+a)/(log(factorial(a)))
e = 0.088526
octave:11> e = (log(factorial(a))-a.*log(a).+a)/(log(factorial(a)))
e = 0.088526
octave:12> y = factorial(a)
```

(Refer Slide Time: 16:01)

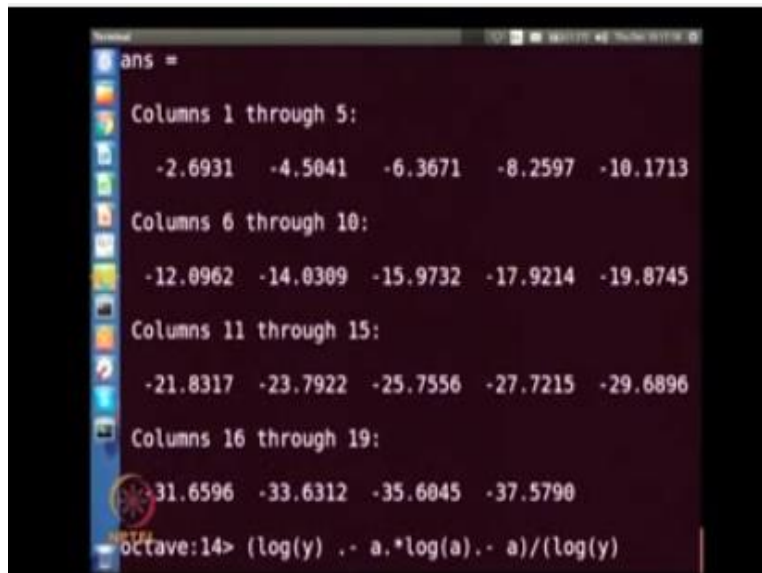


```
Terminal
y =
Columns 1 through 3:
  2.0000e+00  6.0000e+00  2.4000e+01
Columns 4 through 6:
  1.2000e+02  7.2000e+02  5.0400e+03
Columns 7 through 9:
  4.0320e+04  3.6288e+05  3.6288e+06
Columns 10 through 12:
  3.9917e+07  4.7900e+08  6.2270e+09
less -- (f)orward, (b)ack, (q)uit
```

Let me call this  $y = \text{factorial}(a)$ , well now  $y$  is a vector okay, so now I can say  $\log(y) - a$  or  $*\log(a)$  minus okay, so what did I plot so logarithm of factorial  $n$  factorial minus  $a \log a - a$  right this is the approximation. So now let me take this quantity and divide by logarithm of factorial of  $a$  right that is the  $\log(y)$  and that quantity I want to call as error  $E$ , I have made a mistake again so where did I make the mistake so let me call this as error probably I must okay very good okay I get some number again this is not what I want okay, so let me do it I have a okay if I say clear all things go away so  $a$  is gone so let me redo it  $a$  is to 2 to 20.



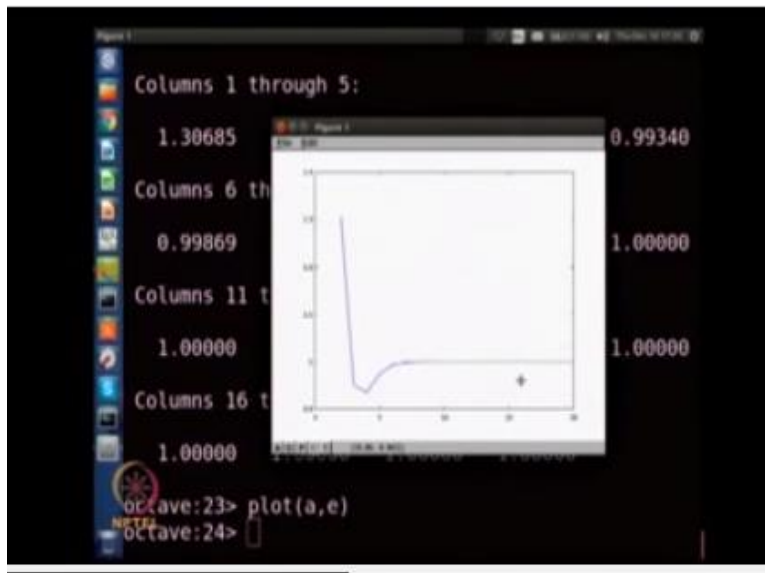
(Refer Slide Time: 16:56)



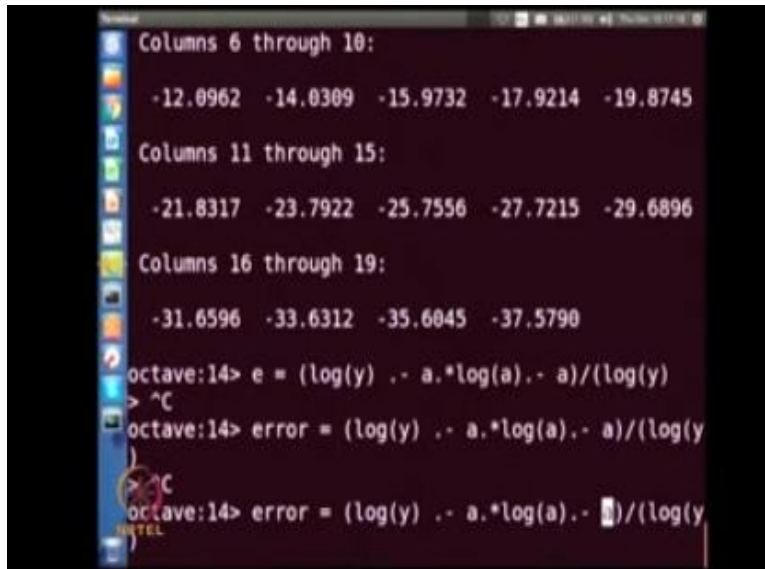
```
ans =  
Columns 1 through 5:  
-2.6931 -4.5041 -6.3671 -8.2597 -10.1713  
Columns 6 through 10:  
-12.0962 -14.0309 -15.9732 -17.9214 -19.8745  
Columns 11 through 15:  
-21.8317 -23.7922 -25.7556 -27.7215 -29.6896  
Columns 16 through 19:  
-31.6596 -33.6312 -35.6045 -37.5790  
octave:14> (log(y) .* a.*log(a) ./ a) ./ (log(y))
```

y is factorial (a) okay, now  $\log(y)$  is log of factorial(y), now a so let me call this as z this is the approximation  $\log(a) - a$ , ok so now I have  $y - z$  okay and divide by y so each component of y I have to take after subtract the z component from there and divided by the corresponding y component and that is what I called it as error. So i have the error so I can now plot a by its error right a is nothing but the n 2 3 4 etcetera up to 20 and the e is nothing but the error that you calculate for those quantities.

(Refer Slide Time: 19:06)

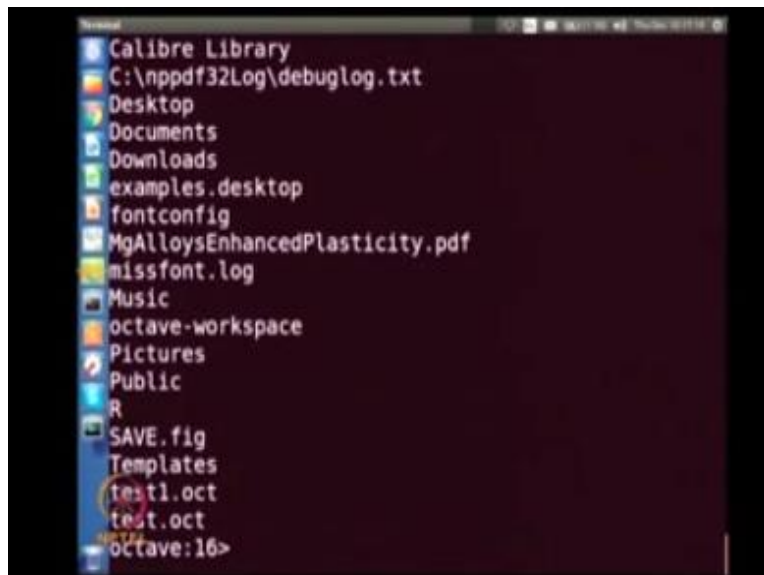


(Refer Slide Time: 17:39)



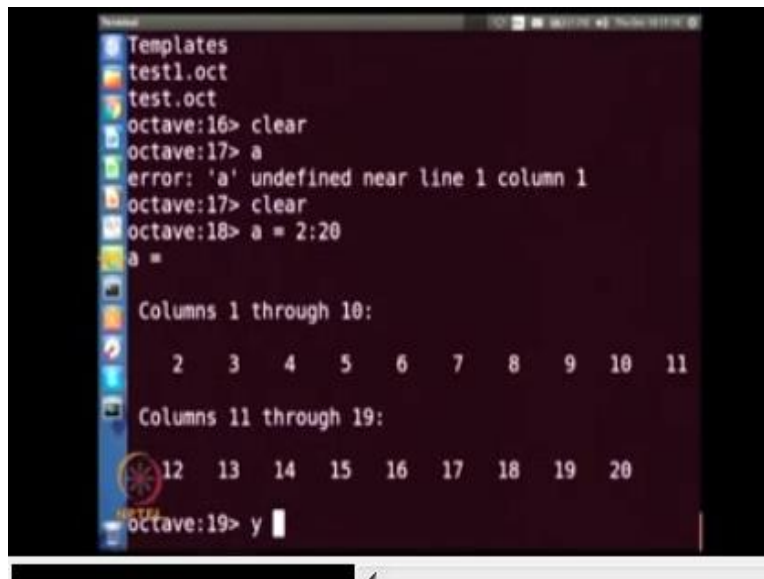
```
Columns 6 through 10:  
-12.0962 -14.0309 -15.9732 -17.9214 -19.8745  
Columns 11 through 15:  
-21.8317 -23.7922 -25.7556 -27.7215 -29.6896  
Columns 16 through 19:  
-31.6596 -33.6312 -35.6045 -37.5790  
octave:14> e = (log(y) .* a.*log(a) ./ a)/(log(y)  
> ^C  
octave:14> error = (log(y) .* a.*log(a) ./ a)/(log(y)  
> ^C  
octave:14> error = (log(y) .* a.*log(a) ./ a)/(log(y)
```

(Refer Slide Time: 17:42)



So if I plot you can see that initially the error is high and then it is coming down but why does it come down to one you should not be getting one okay, there is something still wrong okay so, let me calculate again  $a$  is to 2 to 20 and the factorial( $a$ ) I saw let me call as  $y$  is equal to factorial( $a$ ) okay, this is correct now logarithm ( $y$ ) let me call that as  $z$  is equal to this quantity, let me call a quantity called  $b$  that is nothing but the  $a$  times  $\log(a) - a$ .

(Refer Slide Time: 18:06)



```
Templates
test1.oct
test.oct
octave:16> clear
octave:17> a
error: 'a' undefined near line 1 column 1
octave:17> clear
octave:18> a = 2:20
a =

Columns 1 through 10:

    2    3    4    5    6    7    8    9   10   11

Columns 11 through 19:

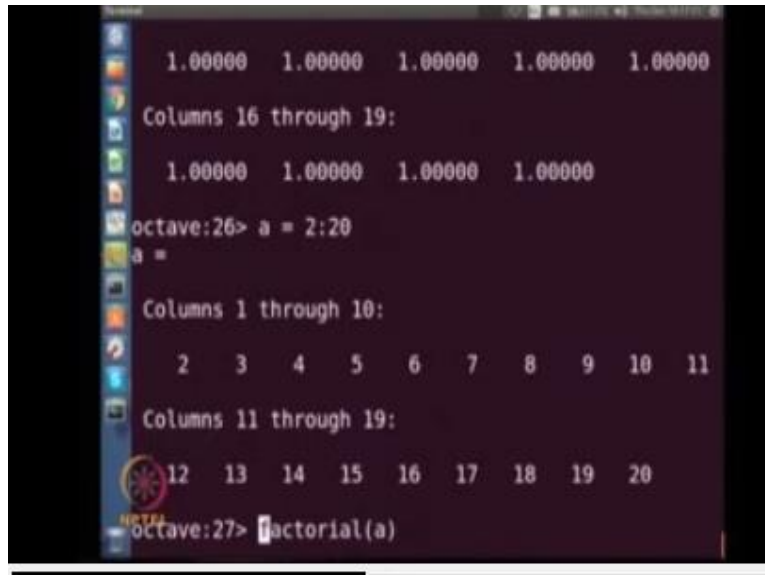
   12   13   14   15   16   17   18   19   20
octave:19> y
```

Now I had the quantity called the  $z$  which was  $\log(y)$  and I now have this quantity okay, so and this quantity is called the  $b$  right the  $z$  is basically  $\log$  of factorial( $n$ ) so  $z - b$  okay, is this quantity and  $z - b$  divided by okay so  $e$  is equal to  $z - b$  divided by these add value okay now I am getting it right. So let us plot this quantity plot  $a$ ,  $e$  right now I have got it correct okay what happens as the  $n$  value is increasing 2 3 4 5 etcetera you can see that the relative error is coming down and it is going towards 0.

(Refer Slide Time: 19:12)

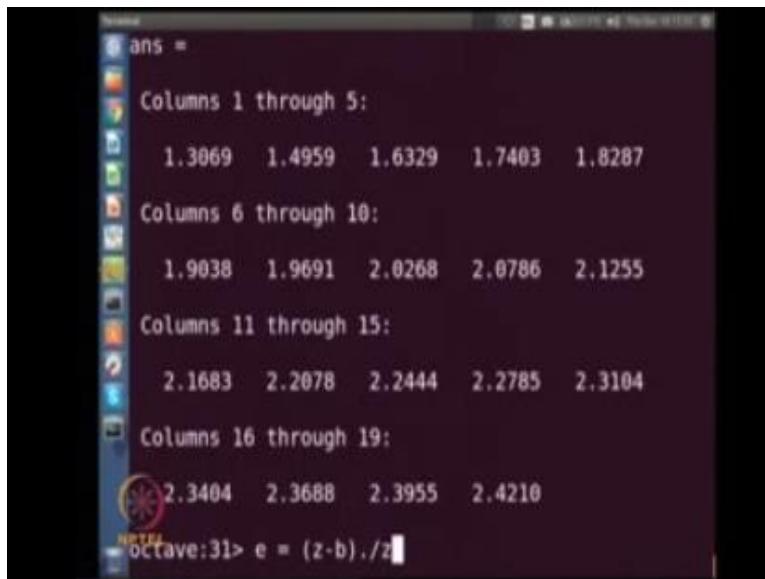
```
e =  
Columns 1 through 5:  
1.30685 0.95069 0.93562 0.97461 0.99340  
Columns 6 through 10:  
0.99869 0.99979 0.99997 1.00000 1.00000  
Columns 11 through 15:  
1.00000 1.00000 1.00000 1.00000 1.00000  
Columns 16 through 19:  
1.00000 1.00000 1.00000 1.00000  
octave:23> plot(a,e)
```

(Refer Slide Time: 19:57)



```
1.00000  1.00000  1.00000  1.00000  1.00000
Columns 16 through 19:
1.00000  1.00000  1.00000  1.00000
octave:26> a = 2:20
a =
Columns 1 through 10:
2  3  4  5  6  7  8  9  10  11
Columns 11 through 19:
12  13  14  15  16  17  18  19  20
octave:27> factorial(a)
```

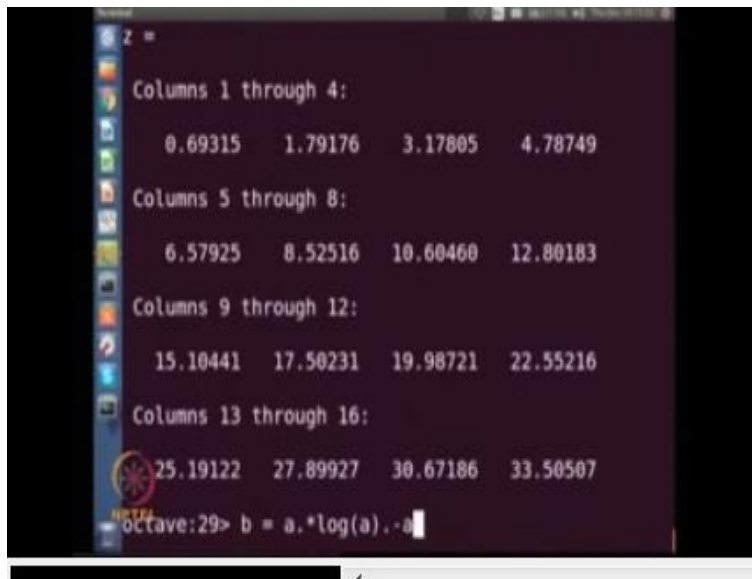
(Refer Slide Time: 21:10)



```
ans =  
Columns 1 through 5:  
    1.3069    1.4959    1.6329    1.7403    1.8287  
Columns 6 through 10:  
    1.9038    1.9691    2.0268    2.0786    2.1255  
Columns 11 through 15:  
    2.1683    2.2078    2.2444    2.2785    2.3104  
Columns 16 through 19:  
    2.3404    2.3688    2.3955    2.4210  
octave:31> e = (z-b)./z
```



(Refer Slide Time: 20:24)

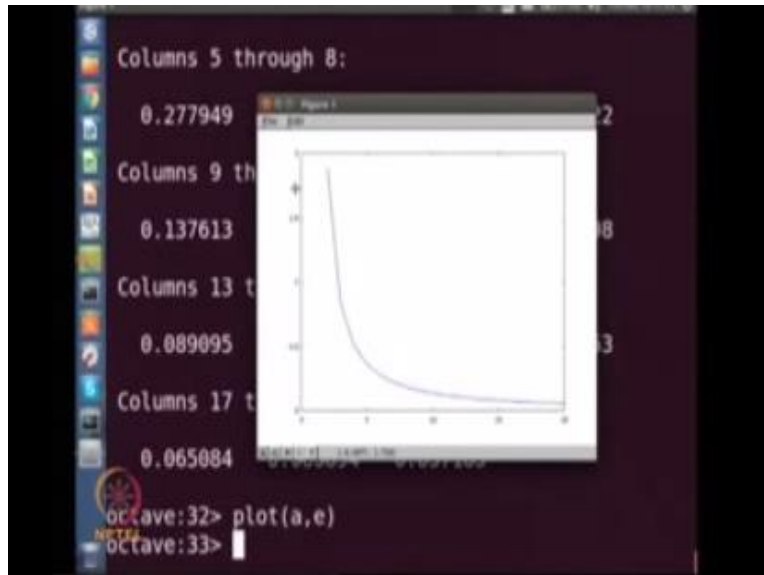


The image shows a terminal window with a dark background and a light-colored text. The terminal displays the following content:

```
Z =  
Columns 1 through 4:  
    0.69315    1.79176    3.17805    4.78749  
Columns 5 through 8:  
    6.57925    8.52516   10.60460   12.80183  
Columns 9 through 12:  
   15.10441   17.50231   19.98721   22.55216  
Columns 13 through 16:  
   25.19122   27.89927   30.67186   33.50507  
octave:29> b = a.*log(a).^a
```

The terminal window has a standard Linux-style taskbar on the left with various application icons. The title bar at the top of the window is partially visible, showing the text "Z =".

(Refer Slide Time: 21:44)



So it is a good exercise for you now to go back and do it for  $n$  values ranging from 2 to 100, you can try for 2 to 150 okay, you will get the similar results but then if you try to calculate for 2 to 200 for example your computer is going to give up okay, so you can try this exercise go to octave try to define  $a$  to be from 2 to 200 and try to do the same kind of error calculation and see what happens and it gives up and if you know programming you know why it gives up if not it is a good idea to look into why it gives up they will come back to this problem sometime later.

So to summarize you can use octave as a calculator and you can also use it to plot functions and so it is very handy even if you do not use it for phase field modeling, you can work with octave as the advanced scientific calculator and plot okay. So in the next one of course I am going to show you one more way of interacting with octave and that is known as the script mode of working with octaves so that is what I will show you next. Thank you

**NPTEL**

**Principal Investigator**

**IIT Bombay**

Prof. R.K Shevgaonkar

**Head CDEEP**

Prof. V.M Gadre

**Producer**

Arun Kalwankar

**Digital Video Cameraman**

**&Graphics Designer**

Amin B Shaikh

**Online Editor**

**&Digital Video Editor**

Tushar Deshpande

**Jr. Technical Assistant**

Vijay Kedare

**Teaching Assistants**

Arijit Roy

G Kamalakshi

**Sr. Web Designer**

Bharati Sakpal

**Research Assistant**

Riya Surange

**Sr. Web Designer**

Bharati M. Sarang

**Web Designer**

Nisha Thakur

**Project Attendant**

Ravi Paswan

Vinayak Raut

**NATIONAL PROGRAMME ON TECHNOLOGY**

**ENHANCED LEARNING**

**(NPTEL)**

**Copyright NPTEL CDEEP IIT Bombay**