MATLAB Programming for Numerical Computation

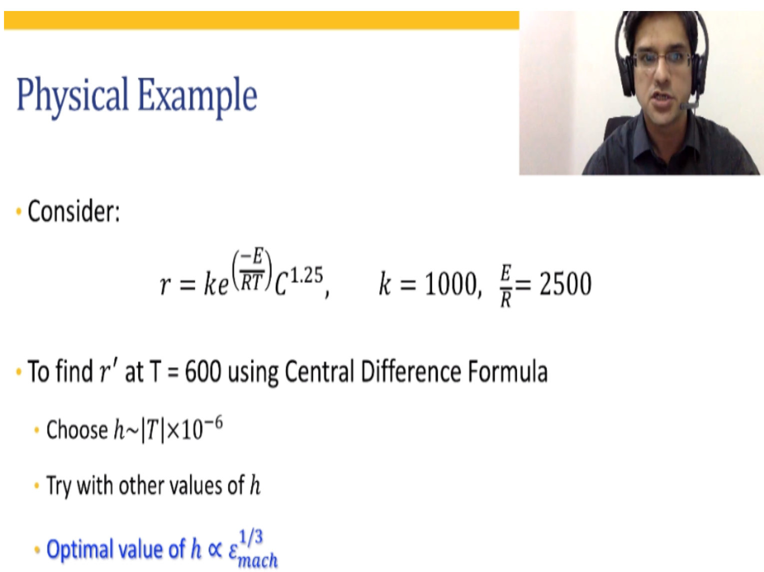
Dr. Niket Kaisare
Department of Chemical Engineering
Indian Institute of Technology, Madras

Module No. #03
Lecture No. #3.3
Numerical Differentiation – Partial Derivatives

Hello and welcome to MATLAB programming for numerical computations. We are in the first part of module 3, in this part, we are considering numerical differentiation in the first 2 lectures. We considered numerical differentiation in single variable. We look at obtaining df/dx and d square f/dx square using forward, backward and central difference formula.

In today's lecture we are going to extend an idea to finding partial derivatives. Partial derivatives when the function f is a function of multiple variables, so f is a function of x1, x2 and so on. Before we do that we will go over the example that we took in the previous lecture. Practical example of rate of reaction.

(Refer Slide Time: 1:00)



We wanted to find its derivative with respect to temperature t at t = 600. We use the central difference formula with h=10 ^ - 4. At the end of this lecture I have suggested that as a practice what you could do is, you could try to solve this problem for various different values of h. In today's lecture we will spend a few minutes time to look back at that same example. Modifying the examples to choose different values of h and obtain what is the optimal value of h that we need to use.

We are not going to look into the theory behind it. We are interested in practical aspects of how you will apply this theory in order to solve problems using MATLAB. In order to do that I am going to give you a certain set of steps. What we will find is that the judicious way of choosing h for central difference formula is, choose h as temperature t * 10 ^ - 6.

If we were using forward or backward difference formulae, then the optimal value of h would be t * 10 ^ - 8. Where does this come from? This comes from the fact that the optimal value of h is proportional to the machine precision to the power 1 / 3. The temperature is not of an order 1 but is of order 100.

We need to proportionately increase h as well ok. So this is the practical tip that we are going to use and we will see how this problem works out by going to MATLAB and trying to solve this problem.

(Video starts 02:44) So, let us head on over to MATLAB okay. So this was the code that we used in the previous lecture. What I am going to do is, now modify the code for different values of h. Different values of h that I am going to take are as follows 1e- 3 to 1e- 7. I am going to multiply this with t okay. So, the value of the independent variable temperature at which we want to find out the first derivative that is the value that we are going to multiply our h with okay.

Now because now our step size is a vector, we have seen this in the previous lecture. What we need to do in order to obtain all these results at one go. What we need to do is, we need to replace this slash with dot slash okay. This guy is a scalar. So, scalar when we want to divide that by a vector, we need a dot slash.

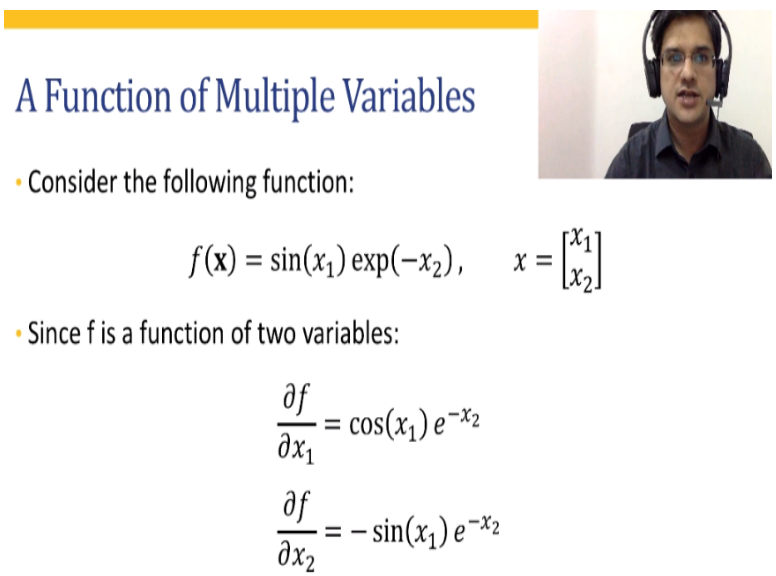
So in the parenthesis what we get is, going to be a vector. So the thing that I have highlighted is a vector, exp function, exp works on a vector as well and it’s element by element exponent of that vector. So, this is also going to give us a 1/5 vector. We are doing scalar multiplication with k and c. So we will not need to change this star to dot star at all okay.

Same changes we need to make in this expression as well we change a slash to dot slash and no other difference we have, no other change we need to make at all. And finally numVal, the numerator is a vector, the denominator is also a vector. So need an element by element division and we obtain that by a dot slash over here. None of the other things need to be changed, okay.

So let me save this and run this and let us see after running what are the various errors that we get for the various values of h. So, when we run this, I am displaying the various errors. We see that the best case errors are over here okay. For these 2 values of h we have the best case errors. If I type out h what I get is that I get best case errors for 6*10 ^- 3 and 6*10 ^- 4. So h we do not choose blindly. (Video ends 05:26)

But we need to choose our optimal h as machine precision to the power 1/3 multiplied by the absolute value of the independent variable of interest. The take home message is then there is an optimal value of h for which the central difference or the forward difference formula gives us the best results. So let us go on to this particular lecture.

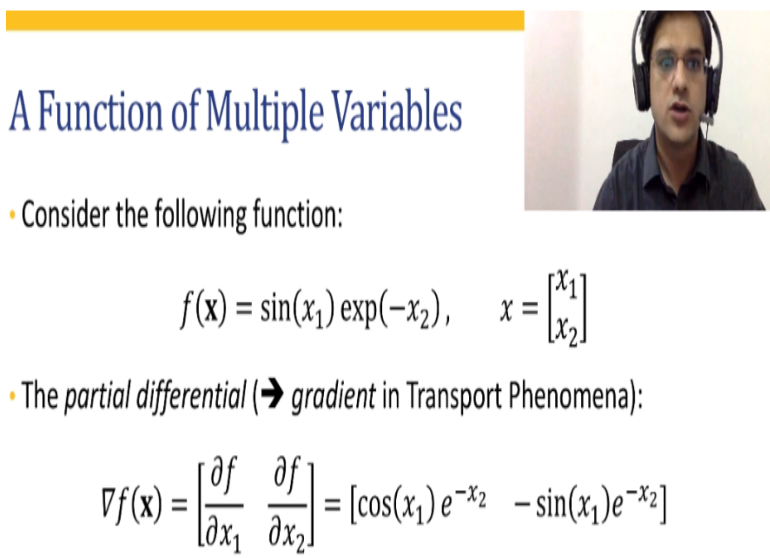
(Refer Slide Time: 05:52)



In this particular lecture we want to find out df /dx. When f is a function of multiple variables as a matter of fact, we have done this already in the previous example. In previous example if you think about it the rate expression is a function of both temperature and concentration.

(Video Starts 06:12) So what we had done in this particular MATLAB code was this is we held the concentration value constant at 1.0 and varied temperature to t + h, t- h and found the central derivative. (Video Ends 06:28)

So that is what a definition of partial differentiation is. Partial differentiation means δf/δx1 keeping x2, x3 and so on. Constant δf /δx2 keeping x1, x3, and x4 so on and so forth okay. So that is the definition of partial derivatives. In order to calculate δf/δx1, we hold x2 constant and vary our x1 in order to calculate δf/δx1, we hold our x1 constant and vary our x2.
(Refer Slide Time: 07:04)



We will notice that a partial differentiation is kind of like finding the gradient of a scalar. The gradient of the scalar is nothing but δf/δx1 and δf/δx2 return as a row vector. Let us go to MATLAB and solve this problem for finding partial derivatives. (Video Starts 07:24). Okay so, our a, the value of x at which we want to find δf/δx. So we want to find δf/δx at x= a, a is going to be a vector.

Let us say we wanted to find this at 0.5 and 1. So, our a is 0.5 and 1. Our trueVal or our h steps size should also be a two value vector 1e- 6 and 1e- 6 as well okay. So, trueVal cosx1 * e ^ - x2. So cos (a1) * exp -a2 and the second guy is -sinx1 e ^ - x2- sin a1 * exp- a2 okay.

Partial derivative with respect to first variable okay, we will choose our x is going to be equal to, x is going to be equal to a and x1 is going to be equal to (a1 + h1) okay. Now our f1 is going to be equal to sin of x1 * exp of- x2 okay. So, this 1 is at (x1 + h) and x2. Likewise we need to calculate our f2 at (x1- h) and x2.

So let us copy and paste this x1 = (a- h) and f2 is sin(x1) - e ^ - x2. And our numDiff 1 is going to be nothing but f2, sorry, (f1- f2) divided by 2h by the central difference formula. Then copy, paste this and say that. We want to find partial derivative with respect to second variable is this function f (x). Rather than we give this function in this particular form, its recommended that we give our half a separate function file where, we define sin x1 and e ^ - x2 as that function.

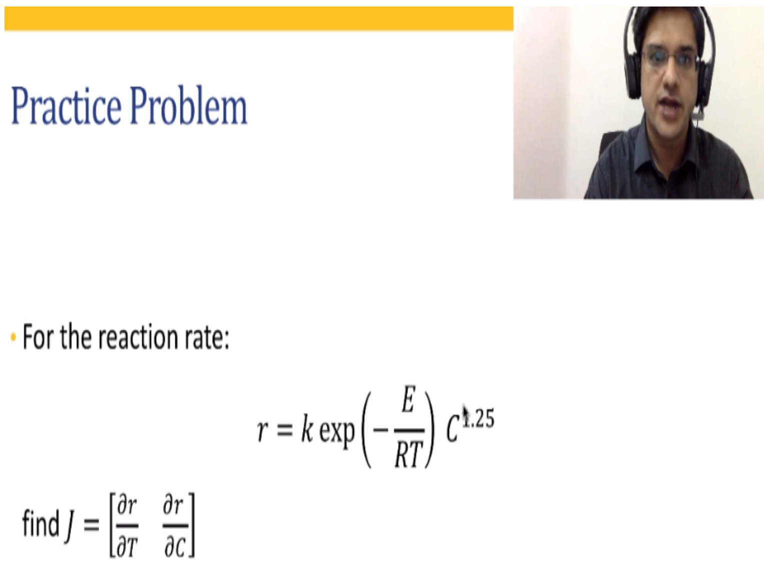
So for the second partial derivative, the partial derivative in the second variable that is what I am going to do. So I am going to create a function. Function fval = myFunM3. My fun for module 3 and my input argument is x. Now x is a vector. x are not 2 values, these are not 2 variables but x is a single variable okay.

And it is a vector variable and what I need to do is, we need to write fval is going to be nothing but sin(x1) * exp of -x2; end and save this as myFunM3 okay. What we want to do over here is, we do it in a slightly different way. What we will do is, we will define our h temp, h temp_h = 0, h2 okay.

This is a better way of doing things because when we do(x + temp h), the first guy is going to be just x1; the second guy is going to be (x2 + h2) okay. Our f1 is going to be nothing but our myFunM3 computed at x1, (x2 + h) which is nothing but (a + temp h) okay. And our f2 is nothing but myFunM3 (a - temp h) and we will have numDiff2 as (f1 - f2) / 2 * h2 okay.

And the same thing that we need to do over here. This also should be 2 * h1 okay. So this is the partial derivatives that's display results. So the first thing we want to display is error. err = abs (trueVal- numDiff). So let us save this and let us run this. Clear all clc, partial differentiation. Let us hope that this runs without an error okay.

So this the result if you see this particular guy difference between trueVal and numDiff turns out to be of the order of 10 ^ - 11 and the numerical derivative is 0.32 and – 0.17 okay. So this is how we take the partial derivatives okay. Now let us move on. (Video ends 14:00)
(Refer Slide Time: 14:06)



So will take the same example for rate of reaction. Where, k, as before was 1000, e/r was 2500. And we need to find δr / δt and δr / δc at t= 600 and c= 1. That is what we need to do over here. What I will recommend you do is, you can pause this video for a short period, you can go on to MATLAB.

(Video starts 14:36). And you have this MATLAB code over here. What I recommend you do is, you can take this particular part of that code and try to modify it to solve this problem okay. Okay so, what I will do is create a new script. I will copy this partial differentiation okay. So our h value is going to be 1e- 6 * t. And the other guy is going to be 1e- 6, that is going to be our h values okay. numDiff1 is going to be myRate x1- myRate x2 divided by, let me call this as h1 and numDiff2. I will call this as also the same but divided by h2, divided by h2 okay.

So what exactly am I doing over here, well let me, will just first save this, reaction partial diff is the name of this script and the name of that function is function r = myRate x okay. e = x1, c = x2 and as we had over here, our r is this guy. Let us go over here and remove this from here and move this to this file, constants, current value of t and c, rate calculation okay.

I will save this as myRate okay. So when I call this myRate with x1, what it should give me? What it should give me is, r calculated at (t + h) and c and when I use this at x2, it should give me r calculated at (t- h) and c okay. So that is what I will do. x1 is going to be equal to c sorry, (t + h1) and c okay, c is going to be kept constant over here. So (t- h1) and c okay. So as you can see the value of c is kept constant over here.

And value of t becomes (t + h) and (t- h) that goes in the numerator divided by 2 * h should go in the denominator over here okay. Likewise what we do over here is, x1 = t. That t remains constant at our original value and (c + h2) and x2 is nothing but t and (c- h2) okay. And this becomes numDiff.

Display results, partial derivative num2str, numDiff okay. Let us save this and let us run this okay. And when we have run this the partial derivatives of r was 0.1077 and with respect to temperature and with respect to concentration it is 38.76 so these are the partial derivatives of r with respect to t and c. (Video ends 20:20)

With that I come to the end of this lecture and this also concludes numerical derivative parts of this module. In the next lecture of this module, I will start with numerical integration. See you in the next lecture. Thanks and bye.