Chapter 1.3 - Iteration Methods: Newton Raphson

15 MIN · YOUTUBE · HTTPS://WWW.YOUTUBE.COM/WATCH?V=MUPN3LTC4YQ
https://www.youtube.com/watch?v=MUPN3ltC4yQ

SUMMARY

Iteration methods are crucial in computing, serving as foundational techniques for solving complex problems, particularly in data-driven modeling and scientific computation. The Newton-Raphson method is highlighted as a classic iterative scheme for finding roots of functions, showcasing the importance of iteration in various computational tasks.

- *Iteration is fundamental in computing, allowing computers to solve complex problems through loops.*
- *The Newton-Raphson method is a widely used technique for root finding, starting with an initial guess and iterating towards a solution.*
- *Iterative processes can exhibit different behaviors: monotone convergence, oscillatory convergence, or divergence.*
- *Fixed points in iteration schemes are critical, representing values where input equals output, often indicating convergence.*
- *The Newton-Raphson formula involves using the function's derivative to update guesses iteratively.*
- *Python code examples demonstrate how to implement the Newton-Raphson method efficiently.*
- *Iteration schemes are foundational for many algorithms in machine learning and scientific computing.*
- *The course emphasizes the significance of understanding iteration as a core concept in computational problem-solving.*

we're going to talk about iteration methods and as they are some one of the more F foundational things that we do in Computing is we do iterations is is what your computer is actually in fact best at is is essentially setting up into iteration loops and you'll see that this can solve many problems for us as we advance forward in looking at our model in data driven modeling and scientific comput computation techniques many of them are foundationally set upon

iteration schemes and so today I want to talk about iteration but also related to Newton rafson method which is one of the classic iteration schemes for finding roots of functions so let's go to it so iteration is fundamental it's one of the most common things that our computer can actually do for solving complex problem problems the computer iterates towards a solution of some form so as you'll see as we progress in this course

you'll see that iteration plays this foundational role in almost everything we do iteration also is foundationally built upon a for Loop so in other words iteration is structured so that you do four and then you would do an iteration type structure within that for Loop so what kind of things can we solve well obviously we can think think think about root solving F ofx equal Z that's what we're going to do with Newton rafson today but you can

also do things like solve ax equal to B in other words the most common linear algebra problem that you might have or even solve something like differential equations you'll find that as you time step forward in this differential equation as you advance solution in time it's fundamentally built upon an iteration scheme of sorts so almost everything we're going to do is foundationally built upon it some kind of iteration scheme because this is ultimately what you're training your

computer to do is iterate for you to get to a solution so here's the generic structure of iteration there's some function G where you put in some value of P let's call it K it's indexed by K so this is the K value of it and you put in this Ki value and out comes P of k + 1 by the way the P of K can be a vector it can be a matrix it's an input through some

function G that produces an output an updated value so that updated value is the if you have the kith value the updated value the k + one value okay and generically here's how we're going to do it you're going to take for instance an initial value like p 0 which is going to generate P1 you're going to take P1 to generate P2 P2 to generate P3 three and so forth so this is kind of the structure we're looking at in

iteration is that you just plug in values and the way you would do something like this is just simply by a for Loop for instance so some interesting things one of things that's very interesting to this is to look for fix points in these iteration schemes and this is going to be directly related for us in terms of root finding so fixed points are when you plug in a value and what comes out is the same value so you

put in P of K you go through G you get back exactly the same values that you started with in other words you've achieved what's called a fix point and in fact in many cases in Computing convergence to a fix point is actually the goal of the computation and that is the solution we are looking for so let's talk about what this might look like so here for instance is some function G of X the function here is this curve and

and this point here is where we put in a value of p and the value output value is also P so this point is very special for us it's along the line yals X in other words Y is the output X is the input is when they are equal okay and that only happens for this curve at this point right here and so we are looking for this point and trying to find it so how would we do this in some kind of

iteration scheme so let me show you some of these these iteration schemes or iteration type behaviors so what I'm giving you here is four different functions and kind of behaviors that can happen with iteration so here in this this screen here this function y ofx which you can see here I start out some value of p of0 and the output I put in P of Z to the function is P1 so to find where P1 is I bring it

back to the yal X line plug in P1 to the system this is the next Point here where my fingertip is that's where P1 is okay if I want to find what the next Val put in P1 to that function it gives me back P2 which is now here and notice What's Happening Here my values progressively as I March forward the P not P1 P2 are going to converge to this point here where the dot is so this is a

monotone convergence of this iteration scheme okay the second one over here is known as oscillatory convergence I put in p 0 I go to my Axis here and say this is where the value is but my output Y is now my P1 so I plug in P1 to the system and then my output is P2 and what happens is I'm essentially iterating around this curve and I'm going to eventually end up at this point by essentially walking around here and

getting closer and closer and closer into that point but now not in an monotone way but in an oscillatory way you can also diverge and of course this is not ideal most of the time when you're doing iterations is you would like the convergence however if you are diverging this is where you want a good if statement saying well if the value gets really large that means you're probably diverging so here is for instance this curve this G of X I plug

in p 0 the next value is P1 okay that's up here I can plug that into the function now I'm here PL it that value in to get P2 that's now out here what I'm doing is I'm moving away from this crossing point so the iteration scheme itself even though there is a fix point the fix point is what's called unstable in other words when I'm near it the iteration scheme just moves away from that point okay so that does it in a monotone

way and you can also just like above do it in an oscillatory way you can move away from this fix point so these iteration Behavior are really important for us to understand because as we go forward in Computing we want to know is are we are we you know are we going to go ahead and converge to something or are we going to diverge and those are important things to for us to understand and also under what conditions do these

things happen Okay so the most classic way to do this and by the way a lot of this we're going to just talk in detail about throughout this course because we're just introducing the first iteration scheme here which is going to be this Newton rapson but you'll see throughout that iteration becomes the center point of almost all of our scientific Computing operations as we move forward in the course so let's start off with the simplest example which is Newton

rafson which is looking for zeros of a function so here's F ofx and R which means the root I'm looking for the roots right those are the zero the values that are the roots of this function okay so the question is how would I find this in practice now in the last lecture we actually did it through bisection but new rapson was a is a much in some sense a much more commonly used technique that we that is used in practice because in

the Newton iteration or sorry in the bisection we actually had to know we had to first of all figure out the left point and the right Point first before we started the algorithm whereas Newton rafson just starts with a guess and starts to iterate towards the solution so here's how it might work there's a function I give you this is the F of xn here it is this function here that you see and I'm going to give it some guess

let's say I'm at The X of nth Point what I'm going to do from this point is I'm going to decide to get to look for the solution by taking off on the tangent at this point now the tangent takes me up to here this point x of n + one and then I can evaluate the function at that Point that's the iteration scheme you start off with a guess or you start off with a value X of N and what you're

going to do is your next guess is just going to be well what's the derivative when does the derivative hit the where the where this zero line is and ask the question am I above or below this line right and how far okay so what I'm going to do is use this as a very simple way to construct the Newton rafson scheme so so first of all what we know is that the slope is given by rise over

run okay but by the way the slope is also the derivative at this point so the derivative at this point in fact is the slope and the and the rise overrun here so look at the rise the rise goes It goes the second point is zero the first point F ofx of n right that's where I'm looking at hitting this point right here and the run is X of n + 1 - x of n so that's it that's the slope

formula and that slope is equal to the derivative at that point so all I'm doing is saying the slope is equal to the derivative which is rise over one which is this formula here and what I'm going to do now is solve for x of n + one from this and when you do this and rearrange here is your formula this is the Newton rafson scheme it says that the next point is equal to the current

Point minus the function evaluated X of N and its derivative evalu of x of n that's it so that is the Newton rsom scheme it's guaranteed to converge if you are sufficiently close it's not clear what sufficiently close means so it's one of those theorems that's not clear that it's that useful because sufficiently close means you might have to be within 10 to- 16 or it could be you're within a billion so it it doesn't say in the theorem how what sufficiently

close means really so in practice it's not a very good theorem but let's execute this on a simple example and in fact it's the example we used with bsection of the last lecture which is f ofx e the x minus Tan x so first of all I'm going to need the derivative so I can simply take the derivative of this which is e the xus secant 2 x so I have those because that is the structure we need to compute the up

update rule so here is what I have the point x of n plus one is X of n whatever I had and then this is the function evaluated X of N and it's derivative evaluative X of n so this is the iteration scheme we're going to set up we're going to write a python code for doing this for solving this transcendental the roots of this transcendental equation okay so here is where we're going to do we're going to go ahead and

write this it's as simple as this here so first of all I'm going to take a guess my guess is -4 so I guess where the zero is and I just so I guess4 and now I'm going to go into a loop and what the loop does here's the loop for J equal 1 0 to 100 the value of x is equal to so notice what I'm doing here I'm actually just coding in this line this formula right here it

says the next point is equal to what the current point is minus the value of the function over the value of the derivative and then I can plot what that value is now if that value is less than 10 and minus 5 in other words if I'm close to being zero I break out of this Loop okay okay the other thing I could do excuse me sorry so that's that's the Newton standard Newton wphs and iteration and what I'm doing here is I'm

just rewrite I'm writing over the previous values with this little structure I could keep a vector of all the values as I've iterating by doing this here this is just a different version of the same python code which is X is going to be NP a pend what that's going to do is I'm going to make an array and keep all the values that I have and so what I'm doing is instead of overwriting the previous value I'm just

adding on to my Vector so my first value is -4 the next one is the next value the next value and so forth so the NP aen command allows me to add on to this so X is NP append my previous value of x and then here this here is my Newton rafson formula the current value minus the function value over the derivative and then I write out the value of the function and then I can compute the

actual functional value and if it's less than 10 - 5 break okay so if I do this here is sort of what the iteration looks like so after only four iterations I guessed a value of -4 and then what it does is it starts Marching In and by the time it hits four iterations I'm within 10us 6 so this is a very fast way to get to this zero right and again luckily for us it actually is a converging iteration

right Newton rson is not guaranteed to converge unless you're sufficiently close so it wasn't clear if my -4 was sufficiently close but it looks like it did the trick okay and by the way this took four iterations versus when I did bsection in the last exam in the last lecture it took 13 iterations so this is a pretty efficient way to do this and a very simple code to write right I mean this is as simple as it gets right there

okay that's very few lines of code again you have access to this on the GitHub page so you can just have both these codes available to you so again what this really setting up for us is iteration schemes are going to play a huge role for us in terms of how we manipulate and solve many of our problems both in machine learning and traditional scientific Computing iteration is sort of this foundational piece it's going to sort of dictate so

many of the algorithms we're going to actually execute and practice for solving scientific problems so again you can find everything here at this website and also download the Clone that GitHub page so you'll have access to it and again iteration methods going to be foundational and now you have Newton rson which is probably the most sort kind of one of the oldest and most important in practice in many ways