

TMV036, Analys och linjär algebra, del A, 2008/09

Programmering och Matlab Laboration 2

Laboration 2 redovisas på studio 7.1 På studio 7.2 kommer vi att ha sista redovisnings tillfälle i grupper för alla laborationer. De som inte hinner vid detta tillfälle måste redovisa senare till Alexei.

Ni får inte poäng för kursen utan att redovisa laborationer.

Allmän uppgift är: att göra om programmet från Laboration 1 för att lösa ekvationer av typ $f(x)=0$ med hjälp av Newtons metod. Det finns beskrivning av Newtons metod i Adams på sid. 246, en länk med en detaljerad text om Newtons metod och lämpliga exempel av Jan på kursens hemsida. Vi hade också en föreläsning med genomgång av Newtons metod och programmerings detaljer i laborationen. Ni kan utnyttja för Laboration 2 mest av texter ni skrivit för Laboration 1 och bara ändra dem lite i den del (cirka 4 rader!) som skiljer Newtons metod och metoden med fixpunktiterationer.

- i) En ny detalj i programmering blir att $f(x)$ måste innehålla en global parameter a .
- ii) En annan möjlighet är att definiera era exempel-funktioner som funktioner av två variabler t.ex. $f(x, a)$ inklusive parametern a . För att anropa dem som funktion av bara en variabel x i huvudprogrammet, som är lämpligt för Newtons program, kan konstruktionen med anonymous funktion användas. Läs beskrivningen av den metoden i detaljerad beskrivning under.

Välj själv vilken variant av funktion med parameter du vill använda.

Detaljerad uppgift:

1. Skriv en funktion i Matlab med namn till exempel NewtonMin som löser en godtycklig ekvation $f(x)=0$ med hjälp av Newtons metod. Det enkelt att göra om texten av funktionen för fixpunktiterationen för detta.
Skillnaden blir att nya funktionen NewtonMin behöver ett funktions argument f_{prim} till för derivatan av funktionen f : `NewtonMin(f, fprim, x0, tol, Nmax)`.
Man måste också ändra den viktiga raden där står beräkningen av nästa approximation från gamla approximationen av lösningen.
2. Skriv ett huvudprogram som använder er funktion för Newtons metod och löser ekvationen $f(x)=0$ för en godtycklig funktion f .
3. Funktionen $f(x)$ måste innehålla en parameter. Till exempel parametern a i funktioner $f(x) = 0.5 + \sin(ax)$ eller $f(x) = 0.5 - \exp(-ax^2)$. Den parametern måste beskrivas som global parameter i huvudprogrammet och få ett värde där. Ni kan igen utnyttja ert huvudprogram från laboration 2 och bara göra om den lite.

Fortsättningen på nästa sida !

4. Hitta ett lämpligt eget exempel av funktionen $f(x)$ i ekvationen (**inte samma som i boken och inte samma som gruppen bredvid !**) och skriv två funktionsfiler: en för funktionen $f(x)$ och en för dess derivata $f'(x)$.

a) Funktioner $f(x)$ och $f'(x)$ **måste ha en parameter a som kan varieras** och kan definieras som global variabel i huvudprogrammet och på samma sätt i funktionsfiler för $f(x)$ och för $f'(x)$. Ni måste förstås kunna skriva ner en formel för $f'(x)$. I exempel ovan $f'(x) = a \cos(ax)$ och $f'(x) = -2ax \exp(-ax^2)$.

b) Alternativt, definiera din exempel funktion **som funktion av två variabler t.ex.**

$f(x, a)$ inklusive parametern a (samma för derivatan). Att anropa den som funktion av bara en variabel x i är lämpligt för Newtons metod program. Det kan göras med hjälp av anonymous funktions konstruktion.

Du kan skapa i ditt huvudprogram en handle (referens till funktion) med bara en variabel x av din tvåvariabels funktion $f(x, a)$ med följande kommandot:

`MyAnonymFunk=@(x) f(x, a)`. Konstanten a måste vara definierad i huvud programmet före detta kommando, x här är bara en formel symbol.

`MyAnonymFunk` kan senare användas (utan `@` !) när du anropar den funktion för Newtons metod och också direkt `y=MyAnonymFunk(x)`, när du ritar graf till $f(x, a)$. Det görs för att undvika skriva flera onödiga argument vid anropet. Samma typ av handle måste skapas för derivatan f' av f .

Välj själv vilken variant av funktion med parameter du vill använda: parametern som global variabel, eller anonymous funktioner.

5. Skriv en funktionsfil för Newtons metod (t.ex. `MyNewton`) som har format `MyNewton(f, fprim x0, tol, Nmax)` där f är funktionen i ekvationen, f_{prim} är dess derivata, x_0 är initial approximation, tol är toleransen, och N_{max} är ett maximalt antal steg i iterationer.
6. Skriv också ett huvudprogram där du ritar graf till din exempelfunktion $y=f(x)$, hittar en startapproximation från grafen och anropar `MyNewton` för din exempelfunktion. Grafen måste ha en skärnings punkt med x -axeln för att vi löser ekvationen $f(x)=0$. Initiala approximationen x_0 kommer att väljas med hjälp av kommandot `ginput` exakt på samma sätt som i `Laborration1`.
7. Kolla att beräkningar funkar korrekt (med ett enkelt exempel där ni kan rätt svar) och sen ändra `MyNewton` funktionen så att beräknings processen även illustreras på samma sätt som på Fig. 4.51 på sidan 246 i Adams. Nämligen funktionen `MyNewton(f, fprim x0, tol, Nmax)` skall rita en bruten linje mellan konsekventa iterationer som. Den är lite annorlunda än spiraler i fixpunktiterationer och är också lätt att programmera med `plot` kommandot.
8. Använd Matlabs kommandot `fzero` för att lösa samma ekvation och se att ni fick rätt svar.

Newton's Method

We want to find a **root** of the equation $f(x) = 0$, that is, a number r such that $f(r) = 0$. Such a number is also called a **zero** of the function f . If f is differentiable near the root, then tangent lines can be used to produce a sequence of approximations to the root that approaches the root quite quickly. The idea is as follows. (See Figure 4.51.) Make an initial guess at the root, say $x = x_0$. Draw the tangent line to $y = f(x)$ at $(x_0, f(x_0))$, and find x_1 , the x -intercept of this tangent line. Under certain circumstances x_1 will be closer to the root than x_0 was. The process can be repeated over and over to get

numbers x_2, x_3, \dots , getting closer and closer to the root r . The number x_{n+1} is the x -intercept of the tangent line to $y = f(x)$ at $(x_n, f(x_n))$.

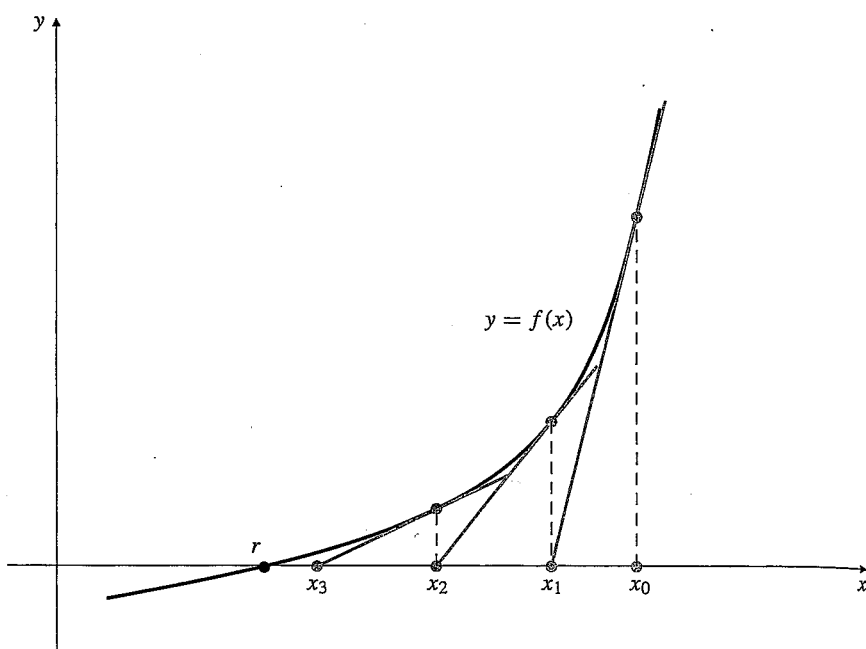


Figure 4.51

The tangent line to $y = f(x)$ at $x = x_0$ has equation

$$y = f(x_0) + f'(x_0)(x - x_0).$$

Since the point $(x_1, 0)$ lies on this line, we have $0 = f(x_0) + f'(x_0)(x_1 - x_0)$. Hence

$$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)}.$$

Similar formulas produce x_2 from x_1 , then x_3 from x_2 , and so on. The formula producing x_{n+1} from x_n is

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}.$$

and is known as the **Newton's Method formula**. We usually use a calculator or computer to calculate the successive approximations x_1, x_2, x_3, \dots , and observe whether these numbers appear to converge to a limit. If $\lim_{n \rightarrow \infty} x_n = r$ exists, and if f/f' is continuous near r , then r must be a root of f because

$$r = \lim_{n \rightarrow \infty} x_{n+1} = \lim_{n \rightarrow \infty} x_n - \lim_{n \rightarrow \infty} \frac{f(x_n)}{f'(x_n)} = r - \frac{f(r)}{f'(r)},$$

from which it follows that $f(r) = 0$. This method is known as **Newton's Method** or **The Newton-Raphson Method**.