Løsning ved iterasjon

Arne Morten Kvarving

Department of Mathematical Sciences
Norwegian University of Science and Technology

17. September 2009

Problem

- Gitt problemet

$$f(x) = 0$$

  for en eller annen funksjon f.
- Eksempel på slike problem kan være
  - $x^2 - 3x + 1 = 0$
  - $\sin x = 0.5x$
  - $\ln 10x - x = 0$.
- Vi har kun eksakte formler for et fåtall problem på denne formen. Vi må derfor ty til numeriske løsninger.

Fikspunktiterasjoner - repetisjon

- Idé: Gjett på en løsning x_0 og finn en eller annen formel for forbedring av denne løsningen.
- En måte vi kan gjøre dette på er at vi transformerer ligningen over på formen

$$x = g(x).$$

- Gitt en initiell gjetning x_0, lager vi så en iterativ prosess ut av dette ved å sette på iterasjonsnummer

$$x_{n+1} = g(x_n).$$

- Løsningen x av denne ligningen kalles et *fikspunkt* for funksjonen $g(x)$. Vi kaller derfor denne iterasjonsprosessen for *fikspunktiterasjoner*.

Problem med fikspunktiterasjoner

- Som vi har sett må funksjonen $g(x)$ være en kontraksjon på intervallet vi betrakter for å garantere at fikspunktiterasjoner konvergerer.
- Selv om vi har en kontraksjon finner vi ikke nødvendigvis alle løsninger.
- Dette kravet er så sterkt at det begrenser anvendelsesområdet betraktelig.
- Av interesse å finne en mer generell anvendbar metode.

Krav til problem

- En slik metode er *Newtoniterasjoner*, også kjent som *Newton-Raphsoniterasjoner*.
- I fikspunktiterasjoner bruker vi ingen informasjon om den deriverte av funksjonen. Vi vet dog at metoden er avhengig av en kontinuerlig og skranket f' (lokalt) for at vi skal oppnå konvergens.
- Vi antar nå at funksjonen har en kontinuerlig derivert.

Nytt problem

- Vi betrakter nå et problem

$$f(x) = 0$$

  hvor $f \in C^1(\mathbb{R})$.
- Eksempler på et slikt problem kan være

$$f_1(x) = x^2 - 3x + 1 = 0$$
$$f_2(x) = \sin x - 0.5x = 0$$

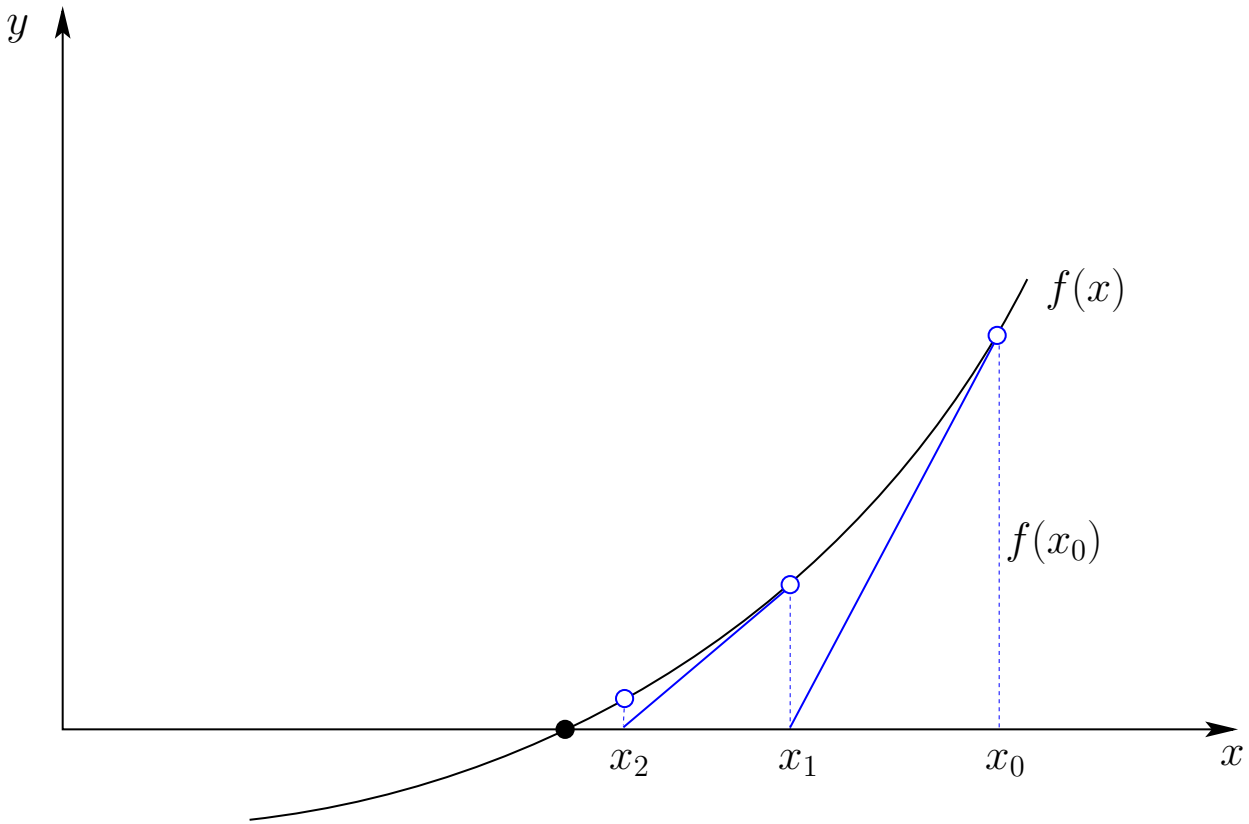
- Et eksempel på et problem vi ikke kan betrakte er

$$f_3(x) = \ln 10x - x = 0.$$

  da vi har et sprang i den deriverte for $x = 0$.

Newtoniterasjoner

- Idé: Vi approksimerer løsningen ved å følge tangenter.



Newtoniterasjoner

- Utleder nå metoden på en litt annen måte for å gjøre matematikken litt enklere.
- Anta at vi har en eller annen iterasjonsverdi x_k. Definerer nå forskyvningsfeilen ϵ ved

$$x = x_k + \epsilon.$$

- Å løse problemet er altså ekvivalent med å finne ϵ.
- Vi Taylorutvikler så f rundt x_k:

$$f(x) = f(x_k + \epsilon) = f(x_k) + \epsilon f'(x_k) + \cdots = 0.$$

Newtoniterasjoner

- Vi trunkerer nå denne serien etter det lineære leddet, i.e. vi betrakter

$$f(x) \approx f(x_k) + \epsilon f'(x_k) = 0.$$

- Løser så denne for ϵ og tar vår neste iteratverdi som

$$x_{k+1} = x_k + \epsilon.$$

- Dette tilsammen gir Newtoniterasjoner;

$$\epsilon = -\frac{f(x_k)}{f'(x_k)}$$

$$x_{k+1} = x_k + \epsilon = x_k - \frac{f(x_k)}{f'(x_k)}.$$

Eksempel

- Betrakter problemet

$$f_1(x) = x^2 - 3x + 1 = 0.$$

  Den deriverte er gitt ved

$$f_1'(x) = 2x - 3.$$

- Setter vi dette inn i iterasjonsskjemaet får vi

$$x_{k+1} = x_k - \frac{x^2 - 3x + 1}{2x - 3}.$$

- Vi vet at de eksakte løsningene er gitt ved

$$x = \frac{3}{2} \pm \sqrt{\frac{5}{4}} \Rightarrow x_1 = 2.61804, x_2 = 0.381966.$$

Eksempel

- Først gjetter vi $x_0 = 0$. Iterasjonstabellen er gitt ved

k	x_k	ϵ	x_{k+1}
0	0	0.3333	0.3333
1	0.3333	0.0476	0.3810
2	0.3810	0.0010	0.3820
3	0.3820	4.591e-7	0.3820
4	0.3820	9.43e-14	0.3820

- Gjetter så $x_0 = 2$. Iterasjonstabellen er gitt ved

k	x_k	ϵ	x_{k+1}
0	2.0000	1.000	3.0000
1	3.0000	-0.333	2.6667
2	2.6667	-0.048	2.6190
3	2.6190	-0.001	2.6180
4	2.6180	-4.591e-7	2.6180

Eksempel

- For referanse; resultatene fra fikspunktiterasjoner for den minste roten.

k	x_k	x_{k+1}
0	0.0000	0.3333
1	0.3333	0.3701
2	0.3701	0.3800
3	0.3800	0.3812
4	0.3812	0.3818
⋮	⋮	⋮
10	0.3820	0.3820

Observasjoner

- Ved å variere den intielle gjetningen oppnår vi begge de to røttene.
- Konvergensen virker til å være betraktelig kjappere enn i fikspunktiterasjoner.
- Hvor kjapp?

Konvergenshastighet

- Betrakt et generelt iterativt skjema på formen

$$x = g(x).$$

Både fikspunktiterasjoner og Newtoniterasjoner passer inn på denne formen, for Newtoniterasjoner er funksjonen $g(x)$ gitt ved

$$g(x) = x - \frac{f(x)}{f'(x)}.$$

- La x_k være en approksimert løsning til en løsning s.
- Definerer som ovenfor forskyvningsfeilen

$$x_k = s + \epsilon.$$

Vi vet at hvis vi kan bestemme ϵ nøyaktig, vil $x_{k+1} = s$.

Konvergenshastighet

- Taylorutvikler og får

$$x_{k+1} = g(x_k) = g(s) + g'(s)\epsilon + \frac{1}{2}g''(s)\epsilon^2 + \dots.$$

- Eksponenten for ϵ foran det første leddet i denne summen som ikke forsvinner kalles for *konvergensorden* for metoden.
- Fra denne definisjonen ser vi at fikspunktiterasjoner har lineær/først ordens konvergens siden vi ikke kan si noe om $g'(s)$ generelt. Men hva med Newton-iterasjoner?

Konvergenshastighet

- Som tidligere nevnt er $g(x)$ gitt ved $x - \frac{f(x)}{f'(x)}$ for Newton. Deriverer

$$g'(s) = 1 - \frac{f'(s)^2 - f(s)f''(s)}{f'(s)^2} = \frac{f(s)f''(s)}{f'(s)^2}$$

- Siden vi vet at $f(s) = 0$ er $g'(s) = 0$ og følgelig er Newtons metode minst av andre orden. Deriverer vi enda en gang finner vi

$$g''(s) = \frac{f''(s)}{f'(s)}.$$

  Denne er ulik null generelt sett, slik at Newtons metode er av andre orden.

Newtoniterasjoner i Matlab - runde 1

- Gitt $x_0 = 0$ og n, antall iterasjoner, kan en Matlab-realisering se ut som:

```
function x = newton1(x0,n)
    x = x0;
    for i=1:n
        eps = -(x^2-3*x+1)/(2*x-3);
        x = x + eps;
    end
```

Avbruddskriterium

- Vi er interessert i å gjøre minst mulig arbeid.
- Vi trenger ofte ikke veldig mange siffer nøyaktighet i svaret.
- Et forslag til et slikt avbruddskriterium kan være

$$|x_{k+1} - x_k| \leq \gamma |x_k|.$$

Newtoniterasjoner i Matlab - runde 2

- Gitt $x_0 = 0$, toleranse γ og n, maks antall iterasjoner, kan en Matlab-realisering se ut som:

```
function x = newton2(x0,n,gamma)
    x = x0;
    for i=1:n
        x_old = x;
        eps = -(x^2-3*x+1)/(2*x-3);
        x = x + eps;
        if abs(x-x_old) <= gamma*abs(x_old)
            break;
        end
    end
```

Funksjonshåndtak

- Vi vil benytte Matlab-koden vår på mange forskjellige problem.
- Matlab har et nyttig verktøy for dette; *funksjonshåndtak*. Disse lar oss sende med funksjoner som parametere til andre funksjoner.
- Disse kan så evalueres med funksjonen *feval*.

Newtoniterasjoner i Matlab - runde 3

- Implementer $f(x)$ og $f'(x)$ som seperate funksjoner

```
function y = test(x)
    y = x^2-3*x+1;

function y = testd(x)
    y = 2*x-3;
```

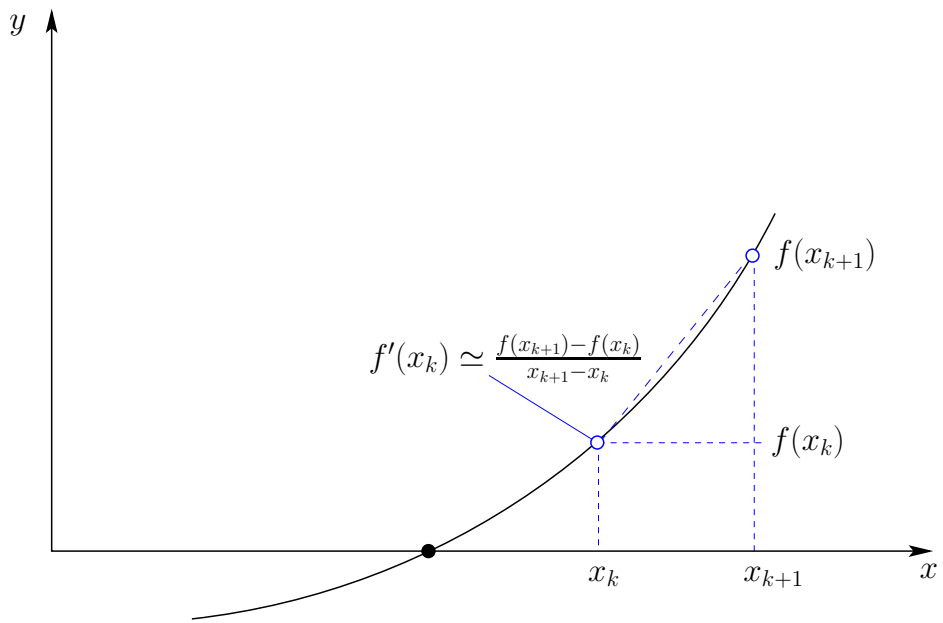
Newtoniterasjoner i Matlab - runde 3 kont

- Selve Newtonløseren ser nå ut som

```
function x = newton3(x0,n,gamma,f,fd)
    x = x0;
    for i=1:n
        x_old = x;
        eps = -feval(f,x)/feval(fd,x);
        x = x + eps;
        if abs(x-x_old) <= gamma*abs(x_old)
            break;
        end
    end
```

Sekantmetoden

- Noen ganger kan det være vanskelig å finne et uttrykk for den deriverte av funksjonen.
- Vi kan/må da ty til en approksimasjon til den deriverte av funksjonen.



Sekantmetoden

- En måte å approksimere den deriverte av funksjonen på er å approksimere den deriverte som en *differanseoperator*.
- Vi bruker

$$f'(x_k) \approx \frac{f(x_k) - f(x_{k-1})}{x_k - x_{k-1}}.$$

- Setter inn og finner

$$x_{k+1} = x_k - f(x_k) \frac{x_k - x_{k-1}}{f(x_k) - f(x_{k-1})}.$$

Sekantmetoden

- Det kan være fristende men vi skriver IKKE

$$x_{k+1} = \frac{x_{k-1} f(x_k) - x_k f(x_{k-1})}{f(x_k) - f(x_{k-1})}.$$

  Hvorfor?
- Det kan vises at metoden har en *superlineær* konvergens, i.e. bedre enn lineær. Orden på metoden er ca 1.62.
- Vi trenger to initielle gjetninger for å sette i gang prosessen.

Sekantmetoden i Matlab

```
function x = sekant(x0,x1,n,gamma,f)
    fx_old = feval(f,x0);
    x_old = x0;
    x = x1;
    for i=1:n
        fx = feval(f,x);
        eps = -fx*(x-x_old)/(fx-fx_old);
        x_old = x;
        x = x + eps;
        fx_old = fx;
        if abs(x-x_old) <= gamma*abs(x_old)
            break;
        end
    end
```