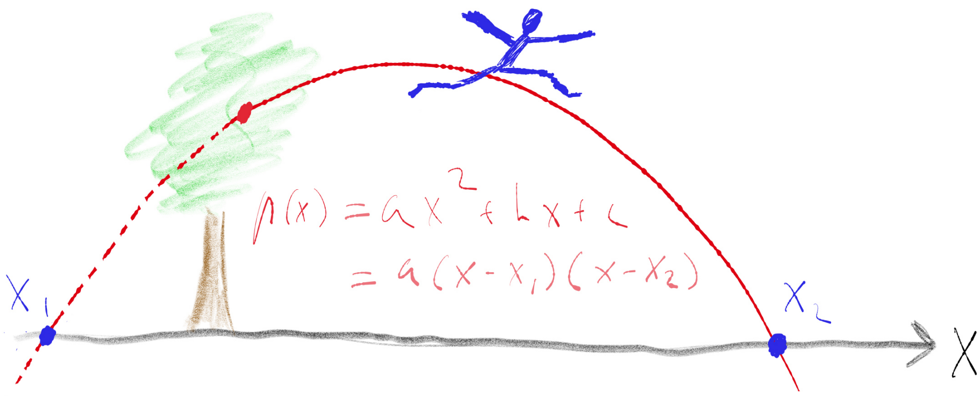
OPPFRISKNINGSKURSET!

5 - LØSNING AV UMULIGE LIKNINGER

Fysiske modeller formuleres stort sett som likninger som involverer fysiske størrelser. I de enkleste tilfellene er modellen en algebraisk likning, slik som $s = s_0 + vt$ eller $\frac{1}{2}mv^2 = mgh$, og løsningen et tall. Dersom man kun påvirket av tyngden og i vakuum hopper ut av et tre og vil regne ut hvor man lander, består dette problemet i bunn og grunn av å løse en annengradslikning.

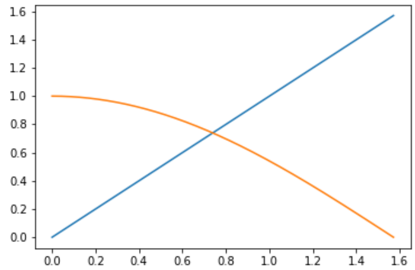


På skolen trente du derfor veldig mye på å løse algebraiske likninger som $x^2 + 2x + 1 = 0$. Men denne tolv år lange løsningstreningsleiren gir av ymse grunner et litt skjevt bilde av virkeligheten. Det mange likninger som ikke har noen løsning (slik som $x^2 = 2$ der x er et rasjonalt tall), og de du på skolen lærte ikke har løsning, har gjerne det allikevel (slik som $x^2 + 2x + 2 = 0$). Likninger kan ha uendelig mange løsninger (ikke bare to slik som annengradslikningene), og det kan tenkes at likningens løsninger eksisterer, men at det er håpløst å regne dem ut.

Vi skal denne uken studere det siste tilfellet, flesker defor til med en ganske vanskelig oppgave.

$\boxed{1}$ Prøv å løse likningen $x = \cos x$.

Her er et plot av funksjonene $f(x) = x$ og $g(x) = \cos x$. Det er lett å se at de to funksjonene krysser hverandre, så løsningen til likningen eksisterer, men den er ikke så lett å finne uten en kalkulator eller en datamaskin.
Løsningen kalles *dottietallet* og er oppkalt etter en professor i fransk. Professor Dottie var gift med en matematiker, og oppdaget at hvis hun skrev inn et tilfeldig tall på kalkulatoren hans og trykket gjentatte ganger på cosinusknappen, endte hun alltid opp på omtrent 0.739085.



$\boxed{2}$ Gjenta Dotties eksperiment i Python. (Nå er det antagelig på tide å lære seg for-løkke.)

Før vi går videre kan vi ta et sitat av en stor matematiker:

> ***"It is impossible to exaggerate the extent to which modern applied mathematics has been shaped and fueled by the general availability of fast computers with large memories. Their impact on mathematics, both applied and pure, is comparable to the role of the telescopes in astronomy and microscopes in biology."***
>
> ***— Peter Lax, Siam Rev. Vol. 31 No. 4***

Konseptet "datamaskin som kan generere prediksjon" er ganske gammelt,[1] og et helt uunværlig verktøy for å løse små og store matematiske problemer. En av de klassiske teknikkene for å løse uløselige likninger kalles **fikspunktiterasjonen**, og det var denne Professor Dottie snublet over ved å trykke på cosinusknappen gjentatte ganger. Hun oppdaget at rekursjonen

$$x_{n+1} = \cos x_n$$

lager en følge som sakte men sikkert konvergerer mot den korrekte løsningen $L \approx 0.739085$. Fikspunktiterasjonen har vært i bruk i tusener av år,[2] og er et eksempel noe som kalles en **numerisk likningsløser**. Dette er noe som produserer en følge $\{x_n\}$ som sakte men sikkert jobber seg inn mot løsningen til en likning. Det er nyttig å kjenne til numeriske likningsløsere, slik at man ikke blir stående fast bare fordi man støter på en likning som er umulig å løse med penn og papir.

$\boxed{3}$ Gjenta professor Dotties eksperiment i Python, men legg inn noe som teller hvor mange iterasjoner det tar før det stabiliserer seg på $L \approx 0.739085133215161$.

[1] https://en.wikipedia.org/wiki/Antikythera_mechanism
[2] https://en.wikipedia.org/wiki/Fixed-point_iteration

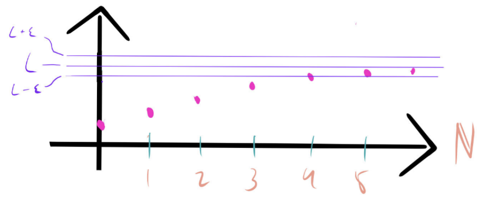
Nå er det antagelig på tide å innføre en ny type funksjon som kalles en **følge**. Dette er en funksjon fra $\mathbb{N}$ til $\mathbb{R}$ eller $\mathbb{C}$. Vanligvis er følgen gitt ved en **rekursjon**:

$$x_{n+1} = f(x_n)$$

og en initialverdi x_0. Spørsmålet er nesten alltid hva følgen konvergerer til. Vi sier at en følge $\{x_n\}$ er **konvergent** dersom det finnes en L slik at det for enhver $\epsilon > 0$ finnes N slik at $n > N$ impliserer

$$|x_n - L| < \epsilon.$$

Dette er en definisjon som får mange studenter til å fryse på ryggen. Men det er dessverre den eneste måten å gjøre det på slik at alt blir presist. Tenk på L som løsningen til $x = \cos x$, og på ϵ som vilkårlig valgt bitte lite tall. Konvergens betyr at alle ledd etter ledd nummer N ligger nærmere L enn ϵ, og at du kan finne en slik N uansett hvor liten ϵ velges.



Fikspunktiterasjonen er gitt ved rekursjonen

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

og den produserer en følge som av og til konvergerer mot en løsning av likningen

$$x = g(x).$$

Om fikspunktiterasjonen konvergerer til noe fornuftig, kommer an på g og på x_0. Nøyaktig når den konvergerer, skal vi komme tilbake til. Nå tar vi noen flere eksempler.

$\boxed{4}$ Likningen

$$x \ln x = 1$$

kan skrives om til formen

$$x = g(x)$$

på minst to forskjellige måter. Finn de to åpenbare, og prøv fikspunktiterasjonen

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

på begge. Hvor mange iterasjoner før du når **maskinpresisjon**, altså $\epsilon \approx 10^{-16}$?[3]

[3] https://en.wikipedia.org/wiki/Double-precision_floating-point_format

Det finnes mange numeriske likningsløsere. En annen klassiker er **Newtons metode**. Denne baserer seg på at likningen som skal løses er skrevet på formen

$$f(x) = 0.$$

Den leter altså etter nullpunkter til funksjoner, og er gitt ved

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

$\boxed{5}$ Løs likningen $x = \cos x$ med Newtons metode og tell antall iterasjoner slik som i oppgave 3. (Du må skrive om likningen litt.) Hva ser du?

Det første elementet x_0 i følgen kalles **initialgjetningen**. Alle numeriske likningsløsere trenger en slik for å komme igang. Noen numeriske likningsløsere er mer sensitive på initialgjetningen enn andre, og forskjellige numeriske likningsløsere oppfører ganske ulikt.

$\boxed{6}$ Løs likningen $x = \cos x$ med både Newtons metode og fikspunktiterasjonen og eksperimenter med forskjellige initialgjetninger. Hva skjer om $x_0 = 4$ eller større?

$\boxed{7}$ Gjenta for $x = \ln x$.

Nå tar vi en oppgave fra den virkelige verden. Denne er visst en klassiker borte på maskin.

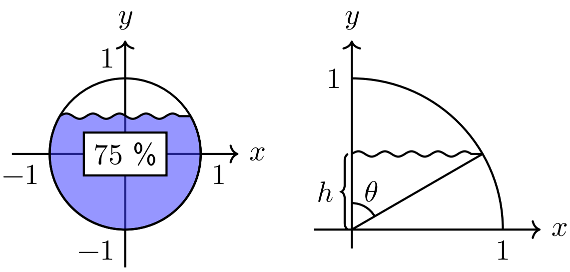
$\boxed{8}$ La h være vannstanden i et kloakkrør med radius 1 m, og la $h = \cos\theta$.

Når vannet fyller 75 % av rørets tverrsnitt, løser θ ligningen

$$2\sin 2\theta + \pi - 4\theta = 0.$$

Finn vannstanden h med fikspunktiterasjonen og Newtons metode.

(Figur: Marius Thaule.)



$\boxed{9}$ Likningen $x^3 + x^2 + x + 1 = 0$ har en reell og to komplekse røtter. Du kan finne disse numerisk, men du må fortelle Python at du har til hensikt å jobbe med komplekse tall og skrive initialgjetningen som $1 + 0 * 1j$. Prøv å finne alle.

Når vi først er inne på det: Det finnes en klasse av funksjoner som er spesielt gunstige. De heter **polynomer**,[4] og er funksjoner på formen

$$p(x) = a_n x^n + a_{n-1}x^{n-1} + a_{n-2}x^{n-2} + \cdots + a_2x^2 + a_1x + a_0.$$

Dersom $a_n \neq 0$ sier vi at polynomet har orden n. Det var Rene Descartes som introduserte denne notasjonen på 1600-tallet en gang. Fordelen med polynomer er at de er enkle å analysere. Tallene $a_n, a_{n-1}, \cdots, a_0$ kalles **koeffisientene**.

Vi sier at en tallmengde er algebraisk lukket dersom alle polynomer med koeffisienter fra tallmengden har et nullpunkt. Dette er ikke sant for $\mathbb{R}$, for eksempel finnes det ingen $x \in \mathbb{R}$ slik at

$$x^2 + 2x + 2 = 0,$$

men den neste tallmengden i matryosjkaen vår, $\mathbb{C}$, er derimot algebraisk lukket. Jean-Robert Argand (en amatørmatematiker!)[5] beviste i 1806 at vi alltid kan faktorisere slik:

$$z^n + a_{n-1}z^{n-1} + \cdots + a_1z + a_0 = \prod_{i=1}^{n}(z - z_i)$$

der $z_i \in \mathbb{C}$ er løsninger av likningen

$$z^n + a_{n-1}z^{n-1} + \ldots + a_1z + a_0 = 0.$$

Dette er altfor vanskelig å bevise for oss, men kan være artig å kjenne til fra tid til annen.

10 Vi vet jo at det kan være vanskelig å finne nullpunktene til et polyom, men nå vet vi ihvertfall at de finnes. Her kan numeriske likningsløsere være til hjelp. Faktoriser polynomet

$$x^3 + x^2 - 3x - 3 = 0.$$

11 Her er en oppgave som slenger ut noen spørsmål vi kan svare på mot slutten av semesteret. Polynomene

$$p(x) = x^4 - 10x^3 + 35x^2 - 50x + 24$$

og

$$q(x) = x^4 - 9x^3 + 27x^2 - 31x + 12$$

har begge nullpunkt i $x = 1$. Prøv å finne nullpunktene med Newtons metode med startverdi $x_0 = 1/2$, og noter deg hvor mange iterasjoner som trengs for både p og q.

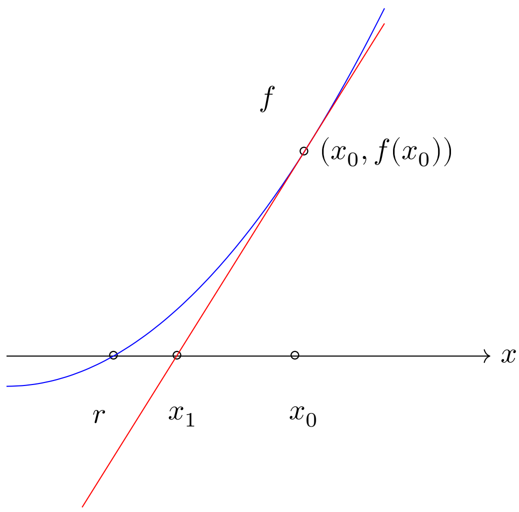
[4]Studenter har flere ganger foreslått at jeg bytter fornavn til Poly.

[5]`https://en.wikipedia.org/wiki/Fundamental_theorem_of_algebra`

Her kommer en utledning av Newtons metode. Newtons metode baserer seg på at likningen er skrevet på formen

$$f(x) = 0.$$

Den leter etter nullpunkter til funksjoner. La oss si at nullpunktet vi leter etter kalles r og at vi har en tilnærming x_0 til r, og så slår vi tangenten til f i x_0. Punktet der tangenten skjærer x-aksen, kaller vi x_1. Dette punktet kan vi finne ved å sette opp likningen for tangenten til f i x_0:



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

og så kreve at $y = 0$ i denne likningen:

$$-f(x_0) = f'(x_0)(x_1 - x_0)$$

Løser vi denne likningen for x_1, får vi at

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

Dersom f og x_0 ser ut slik som i figuren, vil x_1 ligge nærmere nullpunktet enn x_0, og det trengs ikke så stor fantasi for å se at x_2 vil legge seg nærmere nullpunktet enn x_1. Newtons metode er definert som den rekursive følgen

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

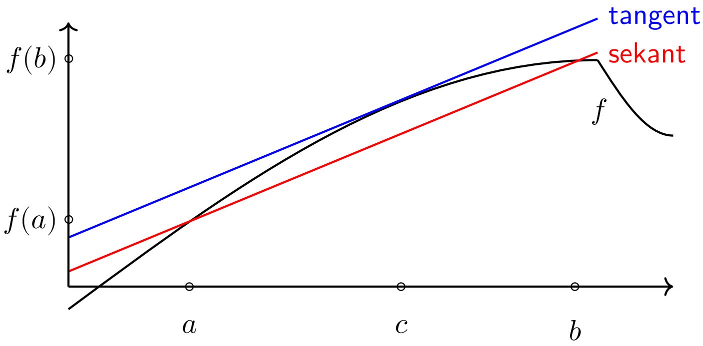
Denne utledningen er litt for vanskelig til at du og jeg ville klart å komme på det selv som nittenåringer. Derfor lager jeg heller en oppgave som heter

$\boxed{12}$ Denne utledningen kan du bli spurt om på eksamen og da må du kunne den.

Hvis du husker **sekantsetningen**, som sier at dersom en funksjon f er deriverbar på et intervall som inneholder a og b, finnes en c mellom a og b slik at

$$f'(c) = \frac{f(b) - f(a)}{b - a}$$

er det mulig for deg å forstå slike ting som hvorfor fikspunktiterasjonen noen ganger konvergerer fort, noen ganger sakte, og noen ganger ikke i det hele tatt.



Dersom g er deriverbar på et intervall som inneholder både x_n og r, sier sekantsetningen at det må finnes en s mellom disse to slik at

$$g'(s) = \frac{g(x_n) - g(r)}{(x_n - r)} = \frac{x_{n+1} - r}{(x_n - r)}$$

som gir at

$$x_{n+1} - r = g'(s)(x_n - r).$$

Denne likningen forteller oss noe om avstanden mellom iterasjonene og løsningen r, og vi ser nå at det kan være lurt å ha peiling på hvorvidt $|g'| < 1$ før vi programmerer opp og trykker på kjør. Hvis $|g'| > 1$ kan det være det ikke konvergerer i det hele tatt.

$\boxed{13}$ I oppgave 4 over, var de to åpenbare omskrivningene $x = 1/\ln x$ og $x = e^{1/x}$. Bruk din nye kunnskap til å spå på forhånd (1) hvilken av disse omskrivningene som kommer til å funke og (2) hvilke startverdier som funker for den omskrivningen som funker.

En liknende analyse av Newtons metode vil forklare hvorfor denne som regel konvergerer fortere. Men dette er noe for komplisert for oppfrisk.