Matematisk modellering og numeriske metoder

Lektion 15

Morten Grud Rasmussen

1. november, 2013

1 Numerisk analyse

[Bogens afsnit 19.1 side 788]

1.1 Grundlæggende numerik

Groft sagt handler numerisk analyse om at bringe matematiske problemer på en form, hvor man arbejder med rigtige tal. De sidste to gange så vi på nogle metoder, hvorved man kan bringe differentialligninger på en form, hvor de indgående størrelser er rigtige tal. Nogle af disse tal skulle dog frembringes ved at regne nogle integraler ud, og her påstod vi, at dette kunne gøre "numerisk."

Det har stor betydning for både hastighed og præcision, hvordan disse numeriske udregninger foretages. I gennemgangen af de numeriske metoder for løsning af differentialligninger så vi, at nogle af begrænsningerne i præcisionen lå i selve metoderne, som fordrede, at man at man gik over til en diskret ("skridtvis") repræsentation af den ønskede funktion. En endnu mere fundamental begrænsning af præcisionen ligger imidlertid i, at enhver konkret beregning kun kan inkludere et endeligt antal cifre, ligegyldigt om det foregår på en computer, i en lommeregner, i hovedet eller på en blok papir.

Da både udregningsmetoden og rækkefølgen i en udregning påvirker præcisionen, har det stor betydning, hvordan man opstiller sin model og implementerer den eksempelvis på en computer. Resten af afsnittet vil derfor omhandle generelle emner relateret til præcisionen af konkrete udregninger.

Flydende-komma-repræsentation af reelle tal

Når et tal skal lagres i en computers (eller lommeregners) hukommelse, er man nødt til at tage en beslutning om, hvor meget plads, der skal afsættes til det og i hvilken form, tallet skal gemmes. Er der tale om et naturlig tal, et helt tal, et rationelt tal, eller et reelt tal? Hvor stort eller lille må det være? Hvis det er et reelt tal, hvordan skal det så gemmes: med et et fast antal cifre før og

efter kommaet, så der er fast afstand mellem hvert nabotal, eller med "flydende komma," således at små tal har kortere afstand til deres naboer end store tal? Fast afstand? Nabotal? Hvad snakker manden om?

Jo, som sagt har vi afsat (allokeret, som det også kaldes) en hvis pladsmængde til tallet, og da computere jo som udgangspunkt kun bruger 0 og 1, og den mindste pladsenhed repræsenterer netop sådan ét 0 eller 1 (kaldet en *bit*), så vil en given pladsmængde udgøres af et vist antal bits, og hvis vi eksempelvis har 32 bits til rådighed, så kan vi vælge at gemme et naturligt tal mellem 1 og 2^{32} – eller mere naturligt et ikke-negativt helt tal mellem 0 og $2^{32}-1$, idet bitsene så blot skal tolkes som 1'er og 0'er i et tal skrevet i totalssystemet (det *binære* talsystem). Vil vi gerne have et fortegn på, så "koster" det en bit, og vi kan i stedet skrive tal fra 0 til $2^{31}-1$ med et "$+$" eller "$-$" foran, og er vi snedige, kan vi vælge at bruge "-0" til noget specielt (eksempelvis "∞" eller $+2^{31}$ eller noget helt tredje). Rationelle tal kan nu repræsenteres som en brøk af et helt tal (med fortegn) samt et naturligt tal (helt, positivt og uden fortegn), eller som de "reelle tal": enten som et fast-komma-tal med et fastsat antal cifre før kommaet og et (eventuelt andet) fast antal cifre efter kommaet, eller som et flydende-komma-tal, med et fast antal *betydende cifre*[1] og et givet (fast) interval for kommaplaceringen. Ligegyldigt hvad vi gør, ændrer vi dog ikke på, at vi kun kan repræsentere et endeligt antal tal, og at antallet er (højst) 2 opløftet til antallet af bits, vi har allokeret. Det korte af det lange er, at ligegyldigt, hvordan vi gør, så vil der være et endeligt antal mulige tal, og ethvert tal vil derfor have et nabotal – det største og det mindste tal har kun ét nabotal, mens alle andre både har en nabo "til venstre" og "til højre."

I praksis er flydende-komma-tal de mest fleksible, og vi vil derfor kigge nærmere på dem i det følgende. Idéen er, at man skriver et reelt tal a som

$$a = \pm m \cdot 2^n, \tag{1}$$

hvor m er et fast-komma-tal udelukkende med cifre efter kommaet (kaldet *mantissen*) og n er et helt tal med fortegn (kaldet *eksponenten*). Der findes en række forskellige standarder for flydende komma (eksempelvis de såkaldte `single`- og `double`-typer), og bortset fra forskellige teknikaliteter afviger de hovedsagligt i antallet af bits, der er allokeret til hhv. m og n. Da der som sagt er grænser for, hvad man kan repræsentere, vil (1) i langt de fleste tilfælde være en *approksimation*. Det mindste tal, som er større end 1, kan skrives $1+\mathrm{eps}$, hvor eps kaldes *nøjagtigheden*. Det er værd at bemærke, at der er samme antal tal mellem $2^0=1$ og $2^1=2$ som mellem $2^{20}=1048576$ og $2^{21}=2097152$.

Hvis et beregningsresultat enten er for stort eller for småt til at blive repræsenteret som et flydende-komma-tal med de givne valg af præcision for n og m, så kaldes det *underflow* hhv. *overflow*, og i første tilfælde sættes resultatet normalt til 0. I andet tilfælde afhænger det af det pågældende setup.

Afrunding

En kilde til fejl er *afhugning* eller *afrunding* (til et bestemt antal cifre eller et bestemt antal *betydende* cifre). En afhugning $\mathrm{chp}(x)$ af et tal x går ud på at fjerne alle cifre, som ligger udenfor bestemmelserne, mens en afrunding $\mathrm{rnd}(x)$ af tallet x er $\mathrm{chp}(x)+(\mathrm{chp}(2x)-2\,\mathrm{chp}(x))$. Resultatet af en afrunding $\mathrm{rnd}(x)$ er generelt tættere på udgangspunktet x end $\mathrm{chp}(x)$ og anbefales derfor. Fejl som

[1] Antallet af betydende cifre er antallet af cifre til venstre for eventuelt foranstillede 0'er. Eksempelvis er der tre betydende cifre i $1.00\cdot 10^9$, $100\cdot 10^7$, 100 og 0.00100.

skyldes afrunding eller afhugning kaldes *afrundingsfejl*. Antag, at vi afrunder til k betydende cifre i 10-talssystemet. Så er den *relative* fejl $\frac{x-\mathrm{rnd}(x)}{x}$ højst $\frac{1}{2}\cdot 10^{1-k}$.

Afrundingsfejl kan ødelægge en udregning fuldstændigt, hvis de enkelte regneoperationer foretages i en uhensigtsmæssig rækkefølge. Derudover er computeres aritmetik ikke eksakt, hvilket kan introducere yderligere fejl.

Tab af betydende cifre

Ved subtraktion af to tal af ca. samme størrelse reduceres antallet af betydende cifre, eksempelvis $2.75000 - 2.71828 = 0.03172$, hvor tallene på venstre side har 6 betydende cifre ("6S"), mens højresiden har 4 betydende cifre. Ofte kan man dog ved lidt snedighed undgå subtraktioner, som det følgende eksempel viser.

Eksempel 1.1 (Bogens Eksempel 2 side 791)**.** Vi vil finde rødderne i $x^2 + 40x + 2 = 0$ med 4S-præcision. Løsningsformlen for en andengradsligning $ax^2 + bx + c = 0$ er som bekendt

$$x_- = \frac{-b - \sqrt{b^2 - 4ac}}{2a} \qquad \text{og} \qquad x_+ = \frac{-b + \sqrt{b^2 - 4ac}}{2a}$$

så

$$x_- x_+ = \frac{-b - \sqrt{b^2 - 4ac}}{2a} \cdot \frac{-b + \sqrt{b^2 - 4ac}}{2a} = \frac{(-b)^2 - \sqrt{b^2 - 4ac}^2}{4a^2} = \frac{b^2 - b^2 + 4ac}{4a^2} = \frac{c}{a},$$

og vi kan altså undgå subtraktionen mellem $-b$ og $\pm\sqrt{b^2 - 4ac}$ (som er to tal af samme størrelse, såfremt $4ac$ er lille i forhold til b^2) ved at vælge den rod $x_\pm$, hvor $-b$ og $\pm\sqrt{a^2 - 4ac}$ har samme fortegn, og finde den anden rod vha. $x_- x_+ = \frac{c}{a}$. I det konkrete tilfælde får vi at $4ac$ og b^2 giver $4 \cdot 1 \cdot 2 = 8 \ll 40^2$, $-b = -40 < 0$ og vi bør derfor først udregne x_-:

$$\begin{aligned} x_- &= \frac{-40.00 - \sqrt{1600 - 4.000 \cdot 1.000 \cdot 2.000}}{2.000 \cdot 1.000} \\ &= \frac{-40.00 - \sqrt{1600 - 8.000}}{2.000} = \frac{-40.00 - \sqrt{1592}}{2.000} \\ &= \frac{-40.00 - 39.90}{2.000} = \frac{-79.90}{2.000} = -39.95. \end{aligned}$$

Dernæst fås $x_+ = \frac{c}{ax_-}$:

$$x_+ = \frac{2.000}{-39.95 \cdot 1.000} = -0.05006.$$

Havde vi brugt den almindelige formel, ville vi have fået

$$x_+ = \frac{-40.00 + 39.90}{2.000} = \frac{-0.10}{2.000} = -0.05000.$$

Fejl i numeriske værdier

Hvis x er den eksakte værdi af et ønsket tal og $\bar{x}$ er den numeriske approksimation (eksempelvis resultatet af en numerisk beregning), så definerer vi $\bar{x}$'s *fejl* til

$$\varepsilon = x - \bar{x},$$

således at den nøjagtige værdi er lig approksimationen plus fejlen.[2] Tilsvarende defineres den *relative fejl* til

$$\varepsilon_r = \frac{\varepsilon}{x} = \frac{x - \bar{x}}{x},$$

således at den relative fejl er fejlen divideret med den sande værdi. Problemet er selvfølgelig, at x og dermed ε normalt vil være ukendte størrelser, men heldigvis er det ofte muligt at lave *fejlvurderinger*, sådan som vi så med den relative fejl ved en afrunding. De to vigtigste fejlvurderinger ser ud på følgende måde:

$$|\varepsilon| \leq \beta \qquad \text{og} \qquad |\varepsilon_r| \leq \beta_r,$$

hvor β og β_r er kendte størrelser.

Udbredelse af fejl

Som nævnt flere gange betyder rækkefølgen af udregninger meget. Hvordan en fejl udbreder sig, altså hvordan en fejlbehæftiget størrelse påvirker de efterfølgende udregninger, kan i mange tilfælde bestemmes ud fra følgende to simple regler.

Sætning 1.2 (Bogens Theorem 1, side 793)**.** *Lad β_x og β_y være grænser på fejlen for x og y, og β_{xr} og β_{yr} være grænser for den relative fejl for x og y. Lad ε_{x+y} og ε_{x-y} være fejlen for hhv. $\bar{x} + \bar{y}$ og $\bar{x} - \bar{y}$, og $\varepsilon_{(xy)r}$ og $\varepsilon_{\frac{x}{y}r}$ være den relative fejl for $\bar{x}\bar{y}$ og $\frac{\bar{x}}{\bar{y}}$. Da er*

$$|\varepsilon_{x+y}| \leq \beta_x + \beta_y \qquad |\varepsilon_{x-y}| \leq \beta_x + \beta_y$$
$$|\varepsilon_{(xy)r}| \leq \beta_{xr} + \beta_{yr} + \beta_{xr}\beta_{yr} \approx \beta_{xr} + \beta_{yr} \qquad |\varepsilon_{\frac{x}{y}r}| \leq (\beta_{xr} + \beta_{yr})(1 + \beta_{yr} + \tfrac{(\varepsilon_y)^2}{y - \varepsilon_y}) \approx \beta_{xr} + \beta_{yr},$$

hvor "$\approx$" gælder for små værdier af β_{xr} og β_{yr}.

Bevis. Overlades til læseren. □

Algoritmer og stabilitet

En *algoritme* er en opskrift på en beregning. En algoritme kaldes *stabil*, hvis små ændringer i inputtet ikke resulterer i store ændringer i resultatet.

1.2 At løse ligninger vha. iterationer

[Bogens afsnit 19.2 side 795]
En ligning i én ukendt kan altid omskrives til formen $f(x) = 0$, for et passende valg af f, ved blot at flytte alle udtryk over på venstre side. Vi vil nu gennemgå tre forskellige metoder til at løse en sådan ligning vha. iteration – dvs. gentagne anvendelser af den samme operation, typisk med outputtet af én anvendelse af operationen som input til den efterfølgende anvendelse af operationen.

[2] Man kunne også have valgt $-\varepsilon$ eller $|\varepsilon|$, men så ville sidste del af sætningen skulle ændres tilsvarende.

Fikspunktiteration

I denne metode skal problemet først omformuleres fra $f(x) = 0$ til $g(x) = x$. Dette kan gøres på flere måder, og det skal vise sig, at den konkrete formulering (det konkrete udtryk for g) har stor betydning for metodens succes. Herefter foretages et kvalificeret gæt på en løsning (eksempelvis kan man skitsere grafen for f og herudfra finde et godt bud på en værdi af x, hvor $f(x) = 0$). Lad os kalde dette bud for x_0. Vi finder nu næste skridt i iterationen ved

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

altså i første omgang fås $x_1 = g(x_0)$, herefter $x_2 = g(x_1) = g(g(x_0))$ osv. Håbet er nu, at der findes et $x_\infty \in \mathbb{R}$ så $x_n \to x_\infty$ for $n \to \infty$, hvilket vil betyde, at $x_\infty = g(x_\infty)$ og altså $f(x_\infty) = 0$. Det viser sig, at der findes et simpelt, tilstrækkeligt krav på g, for at dette er tilfældet. Mere præcist gælder følgende sætning.

Sætning 1.3 (Bogens Theorem 1 side 797)**.** *Lad s være en løsning til $x = g(x)$ og antag, at g er kontinuert differentiabel i et interval J omkring s. Hvis $|g'(x)| \leq K < 1$ i J, så konvergerer følgen $\{x_n\}_{n=0}^{\infty}$ mod $x_\infty = s$, såfremt $x_0 \in J$.*

Bevis. Vi vil blot skitsere idéen i beviset. Lad g opfylde betingelserne i sætningen. Da der findes et t mellem x_n og s således at

$$\frac{g(x_n) - g(s)}{x_n - s} = g'(t),$$

så er $|x_{n+1} - s| = |g(x_n) - g(s)| = |g'(t)||x_n - s| \leq K|x_n - s| < |x_n - s|$. Altså er afstanden mellem x_{n+1} og s skarpt mindre end afstanden mellem x_n og s, og vi ser, at følgen $\{x_n\}_{n=0}^{\infty}$ nærmer sig s. □

Eksempel 1.4 (Bogens Example 1 side 796)**.** Vi vil benytte fikspunktmetoden til at finde en numerisk løsning[3] til $f(x) = 0$, hvor $f(x) = x^2 - 3x + 1$. En skitse af grafen vil afsløre, at der findes en løsning omkring $x = 2.6$ og omkring $x = 0.4$. Hvis vi omskriver $f(x) = 0$ til $g(x) = \frac{1}{3}(x^2 + 1) = x$, så er $g'(x) = \frac{2x}{3}$ og $|g'(x)| \leq \frac{2}{3}$ for $x \in [0, 1]$. Vælger vi derfor $x_0 = 0.4$, vil $x_n \to x_\infty$ for $n \to \infty$ med $f(x_\infty) = 0$. Det kan vises, at x_6 er korrekt til 5 betydende cifre (5S). Derimod er $g'(2.6) = 1.7333 > 1$, og vi kan altså ikke finde løsningen omkring 2.6 med dette valg af g. Hvordan ville det forholde sig, hvis vi i stedet havde valgt $\tilde{g}(x) = 3 - \frac{1}{x}$?

Newtons metode

Antag nu, at f er kontinuert differentiabel. Da en tangent til en kontinuert differentiabel funktion er en god tilnærmelse til funktionen omkring tangentens skæringspunkt, så vil en tangent nær et nulpunkt s, $f(s) = 0$, skære x-aksen tæt på s (prøv at skitsere situationen). Newtons metode går ud på at udnytte dette, ved at vælge et x_0 tæt på s, udregne tangenten i x_0's skæring med x-aksen, kalde dette skæringspunkt for x_1 (som gerne skulle være *endnu* tættere på s), finde tangenten i x_1's skæring med x-aksen osv., således at x_{n+1} er skæringspunktet med x-aksen for tangenten i

[3] En *numerisk løsning* vil – som i tilfældet for numeriske løsninger til differentialligninger – betyde en numerisk approksimation af den "rigtige" (analytiske) løsning.

x_n. Da tangenten har konstant hældning $f'(x_n)$ som også kan udregnes som hældningen mellem $(x_n, f(x_n))$ og $(x_{n+1}, 0)$, så er

$$f'(x_n) = \frac{f(x_n) - 0}{x_n - x_{n+1}} \qquad \text{eller} \qquad x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}.$$

Det er klart, at metoden bryder sammen, hvis $f'(x_n) = 0$ (algebraisk, fordi vi ikke må dividere med 0, men mere konceptuelt, hvad går så galt? Tegn!). Bemærk, at Newtons metode faktisk er en fikspunktsmetode med $g(x) = x - \frac{f(x)}{f'(x)}$. Skal en numerisk løsning findes, kan man vælge at stoppe, når $f(x_n)$ er tilpas lille, eller når $|x_{n+1} - x_n|$ er tilpas lille. Her kan man vælge "tilpas lille" i absolut værdi eller – mest relevant i sidstnævnte tilfælde – lille *i forhold til* $|x_{n+1}|$.

Eksempel 1.5 (Example 3 i bogen side 800)**.** Antag, at vi vil finde kvadratroden af 2. Det er den positive løsning til $x^2 - 2 = 0$, og vi kan altså sætte $f(x) = x^2 - 2$. Så er $f'(x) = 2x$ og med $x_0 = 1.5$ fås

$$x_1 = 1.5 - \frac{1.5^2 - 2}{2 \cdot 1.5} = \frac{1}{2}\Big(1.5 + \frac{2}{1.5}\Big) = 1.41667, \qquad x_{n+1} = \frac{1}{2}\Big(x_n + \frac{2}{x_n}\Big),$$

og altså $x_2 = 1.41422$, $x_3 = 1.41421$ osv. Læg mærke til den algebraiske omskrivning af udtrykket, som gør, at vi undgår subtraktion!

Sekantmetoden

Hvis det af den ene eller anden grund er problematisk at udregne $f'(x)$, så findes der en modifikation af Newtons metode, som springer f' over ved at skifte tangenter ud med sekanter. En sekant er en linje gennem to punkter på grafen for en funktion, og vi skal altså ikke blot have ét godt gæt som input, men to tal, som begge ligger tæt på, hvor vi forventer at finde et nulpunkt. Til gengæld er det såre simpelt at modificere Newtons metode med sekanthældningen. Da $f'(x_n) \approx \frac{f(x_n) - f(x_{n-1})}{x_n - x_{n-1}}$ for x_{n-1} tæt på x_n, så sættes

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

som i parentes bemærket ikke bør omskrives til $x_{n+1} = \frac{x_{n-1}f(x_n) - x_n f(x_{n-1})}{f(x_n) - f(x_{n-1})}$, da det giver en numerisk set meget uheldig subtraktion.

Eksempel 1.6. Lad os finde $\sqrt{2}$ vha. sekantmetoden, hvor vi sætter $f(x) = x^2 - 2$ og vælger $x_0 = 1.5$ og $x_1 = 1.4$. Så er

$$x_2 = 1.4 - f(1.4)\frac{1.4 - 1.5}{f(1.4) - f(1.5)} = 1.4 - (-0.04) \cdot \frac{-0.1}{-0.04 - 0.25} = 1.41379,$$

$x_3 = 1.41422$ og $x_4 = 1.41421$.

Konvergenshastighed

Når vi nu ser på tre forskellige metoder til at løse samme problem, må der selvfølgelig være forskel i styrker og svagheder. Én af parametrene, man kan kigge på, er konvergenshastighed. Men

hvordan sammenligner man konvergenshastigheder uden at betragte konkrete tilfælde? En meget brugbar måde at sammenligne sådanne hastigheder er at betragte deres *orden*. Lad g være en funktion, som definerer en iterationsmetode ved $x_{n+1} = g(x_n)$ (overvej, hvad g er i fikspunktiterationsmetoden og i Newtons metode – i sekantmetoden skal man omformulere begrebet lidt, da iterationen her ikke blot afhænger af x_n men også x_{n-1}). Lad nu ε_n være fejlen i x_n (altså $\varepsilon_n = s - x_n$). Antag, at g er flere gange differentiabel, så vi kan opskrive en Taylor-udvikling af g omkring løsningen s:

$$\begin{aligned} x_{n+1} = g(x_n) &= g(s) + g'(s)(x_n - s) + \frac{1}{2}g''(s)(x_n - s)^2 + \cdots \\ &= g(s) - g'(s)\varepsilon_n + \frac{1}{2}g''(s)\varepsilon_n^2 + \cdots . \end{aligned}$$

Ordenen af iterationsmetoden er nu eksponenten k af ε_n i det første led, hvor $g^{(k)}(s) \neq 0$. Hvis ε_n er lille, så dominerer dette led, og vi får altså for en k'te-ordens iterationsmetode, at

$$x_{n+1} \approx g(s) + \frac{(-1)^k}{k!}g^{(k)}(s)\varepsilon_n^k \qquad \text{eller} \qquad \varepsilon_{n+1} \approx -\frac{(-1)^k}{k!}g^{(k)}(s)\varepsilon_n^k,$$

så fejlen bliver altså hurtigt mindre, hvis k er stor.

Det skulle nu være oplagt, at ordenen af fikspunktsiterationsmetoden på helt afgørende vis afhænger af valget af g. For Newtons metode kan man dog nemt vise, at hastigheden er (mindst) andenordens:

$$g(x) = x - \frac{f(x)}{f'(x)} \qquad \text{så} \qquad g'(x) = 1 - \frac{f'(x)^2 - f(x)f''(x)}{f'(x)^2} = \frac{f(x)f''(x)}{f'(x)^2},$$

så sætter vi s ind, ser vi, at $g'(s) = 0$, idet ($g(s) = 0$). Vi opsummerer i følgende sætning.

Sætning 1.7 (Bogens Theorem 2 side 802)**.** *Hvis f er to gange differentiabel og $f'(s)$ ikke er 0, hvor $f(s) = 0$ er en løsning, så er Newtons metode mindst af orden 2.*

Differentierer man g igen får man

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

og dermed $\varepsilon_{n+1} \approx -\frac{f''(s)}{2f'(s)}\varepsilon_n^2$. Vi kan bruge dette udtryk til at vurdere fejlen i et givet skridt:

Eksempel 1.8 (Bogens Example 6 side 802)**.** Vi vil estimere fejlen i det $n+1$'ste skridt i iterationen, hvor vi leder efter den positive løsning til $2\sin(x) = x$ vha. Newtons metode. Vi vælger $x_0 = 2$, sætter $f(x) = x - 2\sin(x)$ og får $x_1 = 1.901$. Vi vil nu estimere $\frac{f''(s)}{2f'(s)}$:

$$\frac{f''(s)}{2f'(s)} \approx \frac{f''(x_1)}{2f'(x_1)} = \frac{2\sin(x_1)}{2(1 - 2\cos(x_1))} \approx 0.57,$$

så $|\varepsilon_{n+1}| \approx 0.57\varepsilon_n^2 \approx 0.57^{2^{n+1}-1}\varepsilon_0^{2^{n+1}}$, hvor sidste circa-tegn opnås ved at gentage tricket igen og igen. Vi mangler nu blot at finde et estimat for ε_0. Men

$$\varepsilon_1 - \varepsilon_0 = (\varepsilon_1 - s) - (\varepsilon_0 - s) = -x_1 + x_0 \approx 0.10$$

samtidig med at $\varepsilon_1 \approx -0.57\varepsilon_0^2$, så $-0.57\varepsilon_0^2 - \varepsilon_0 \approx 0.10$, som har de to løsninger $\varepsilon_0 = -0.11$ og $\varepsilon_0 = -1.65$. Sidstnævnte løsning er ikke kompatibel med den løsning, vi leder efter, så fejlen i $n+1$'ste led er altså estimeret udlukkende ud fra x_0 og x_1 til $0.57^{2^{n+1}-1}0.11^{2^{n+1}}$.