Matematikk for musikere

Øyvind Hammer
Norsk nettverk for Teknologi, Akustikk og Musikk

Oslo, 10. mai 1995

Innhold

Kapittel 1

Innledning

Det er en kjensgjerning at mange komponister og musikere ikke har den aller beste bakgrunn i og interesse for matematikk. Den russiske nevrofysiologen Luria gjorde en gang en stor studie der han så på sammenhengen mellom musikalitet og matematiske evner. Blant de tusener av mennesker som var med i hans materiale, fant han ikke en eneste person som scoret høyere enn gjennomsnittet både på den musikalske og den matematiske testen. Jeg antar imidlertid at han ikke testet f.eks. Xenakis!

Når man arbeider med computermusikk, er det likevel tvingende nødvendig å kjenne endel enkle matematiske prinsipper. Dette blant annet fordi man oftest er nødt til å spesifisere matematiske funksjoner for å styre forløp i lyden. I dette kapitlet skal vi repetere enkle forhold fra skolematematikken, men også se på noen resultater fra mer avansert matematikk som er viktige for oss.

Jeg har valgt ut emner som har direkte relevans for praktisk arbeid med computermusikk. Dette betyr at sentrale felter som differensial- og integralregning er helt utelatt. Nesten alle krav til stringens er dessuten kastet på båten, så beviser må du finne annetsteds.

Kapittel 2

Funksjoner

Den viktigste ideen i matematikken må vel være funksjonsbegrepet. Vi skal se litt på dette, og dessuten presentere en zoo av funksjoner som man trenger i sin verktøykasse når man skal konstruere musikalske forløp.

2.1 Hva er en funksjon?

En funksjon er en regel som anvendes på et tall x som vi putter inn i funksjonen, og som leverer et annet tall $f(x)$. Selve regelen kalles *funksjonsforskriften.*

Eksempler:

$$f(x) = 1$$

vil levere tallet 1 uansett hvilken x man putter inn (dette kalles en konstant funksjon).

$$f(x) = x$$

vil sende ut samme tallet som vi sendte inn (dette kalles identitetsfunksjonen).

$$f(x) = x + 1$$

vil addere 1 til tallet vi sendte inn (dette er en lineær funksjon, se under).

Vi kan betrakte en funksjon som en maskin som omformer tall. Når vi arbeider med musikkprogrammet MAX, vil en funksjon i praksis bety en liten boks på skjermen med en inngang og en utgang, hvor vi har skrevet funksjonsforskriften inne i boksen.

Vi skal se på endel spesielle funksjoner under, men først må man ha klart for seg hva en *funksjonsgraf* er. En slik graf er en framstilling av funksjonsverdiene plassert i et *koordinatsystem.* Vi plotter alle "fornuftige" x (de som tilhører funksjonens *definisjonsområde*) og deres tilhørende funksjonsverdier $f(x)$, slik man er kjent med fra alle mulige kurver i avisene. En kurve fra børsen framstiller tiden langs x-aksen, og aksjekursen til ethvert tidspunkt som $f(x)$-verdier.

Med blant funksjonene hører også funksjoner av flere variable, f.eks.

$$f(x, y) = x + y$$

og funksjoner som opererer på andre typer objekter enn vanlige tall (vektorer, matriser, komplekse tall).

2.2 Lineære funksjoner

En lineær funksjon er på formen

$$f(x) = ax + b,$$

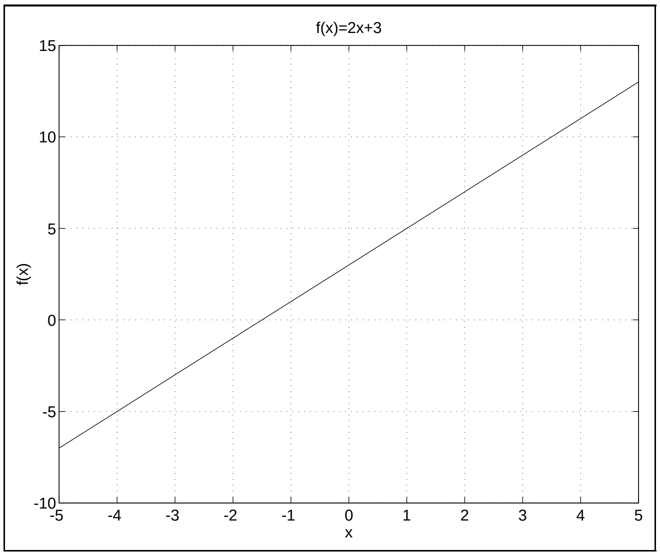
der a og b er faste tall som vi velger. Eksempel:

$$f(x) = 2x + 3.$$

Her er $a = 2$ og $b = 3$. Det betyr at funksjonen multipliserer x med 2 og legger til 3.

Det finnes endel spesialtilfeller av lineære funksjoner, f.eks. nullfunksjonen $f(x) = 0$ der $a = 0$ og $b = 0$, identitetsfunksjonen $f(x) = x$ der $a = 1$ og $b = 0$, konstante funksjoner $f(x) = b$ og multiplikasjon med konstant $f(x) = ax$.

Lineære funksjoner har fått sitt navn fordi deres graf alltid er en *rett linje*, som vist i figur 2.1.



Figur 2.1: Lineær funksjon

Legg merke til at konstanten a (her 2) styrer hvor *bratt* kurven er, mens konstanten b (her 3) bestemmer verdien $f(0)$.

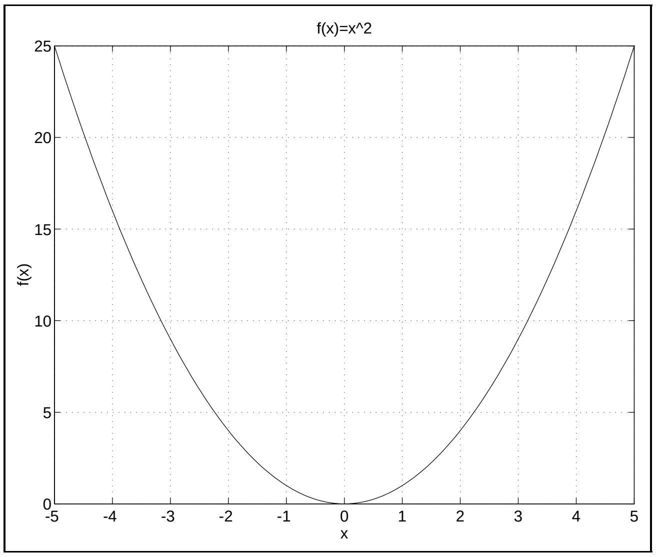
Lineære funksjoner er på mange måter spesielle, og lette å håndtere. For eksempel er det bare lineære funksjoner og forsinkelsesfunksjoner (se under) som slipper igjennom lyd uten at klangfargen endres.

2.3 Polynomer

Dersom vi multipliserer et tall x med seg selv $(x \cdot x)$ sier vi at vi har opphøyet tallet i andre potens, og vi skriver $f(x) = x^2$.

Eksempel: $3^2 = 3 \cdot 3 = 9$.

Grafen til x^2 er en såkalt *parabel*, og er vist i figur 2.2.



Figur 2.2: Parabel

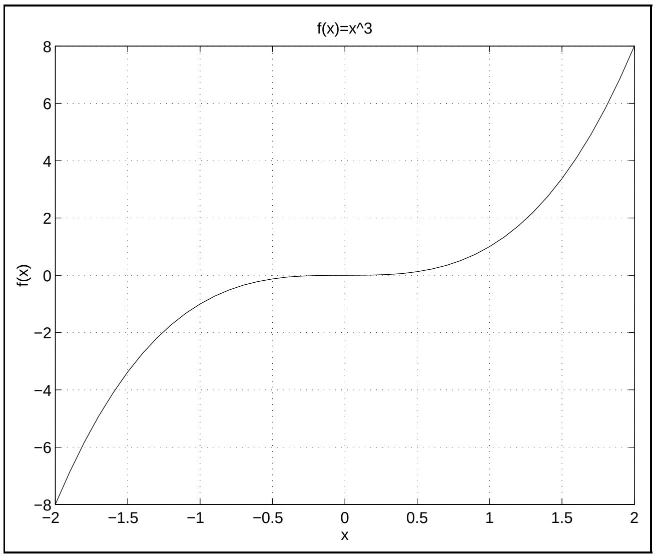
Her er det at par ting å legge merke til. For det første ser vi at alle negative x-verdier ender opp som positive $f(x)$. For det andre er det tydelig at grafen er *symmetrisk om x=0*; formelt skriver vi $f(x) = f(-x)$ (kan du finne ut av det?). Vi sier også at x^2 er en *like* funksjon.

Figur 2.3 viser grafen til x^3.

x^3 er *antisymmetrisk*, slik at $f(x) = -f(-x)$. Vi sier også at x^3 er en *odde* funksjon. Det gjelder generelt at like tall i potensen gir like funksjoner, og odde tall i potensen odde funksjoner.

Vi skal nå se på funksjoner som er en sum av de typer funksjoner vi har sett på hittil. For eksempel er $3x^2 + 2x + 1$ en sum av funksjonen x^2 (riktignok med en faktor 3 foran), og den lineære funksjonen $2x + 1$. En slik funksjon kalles et *polynom*. Polynomer har en *grad* etter den høyeste potensen som inngår.

Eksempler:
$5x + 2$ og $3x$ er polynomer av 1. grad.
$4x^2 - 7x + 3$ er et polynom av 2. grad.
$7x^{14} + 9x^8 - x + 64$ er et polynom av 14. grad.



Figur 2.3: Tredjegradspolynom

Det finnes mange forskjellige spesielle klasser av polynomer, hvor koeffisientene er konstruert etter bestemte regler. F.eks. er det de såkalte *Bessel-polynomene* som bestemmer styrken på deltonene i spektret ved FM-syntese.

Eksempel 1: Simulering av rørforsterker

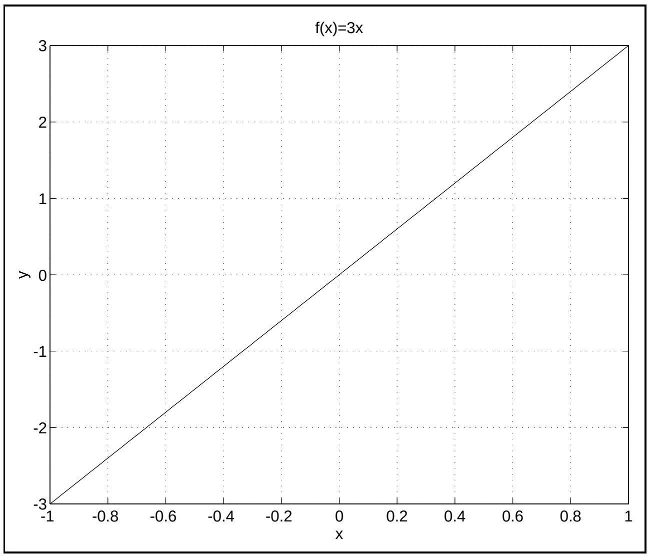
En forsterker tar et inngangssignal x og forsterker det med en faktor a, som styres av en volumkontroll. En forsterker kan dermed beskrives ved en såkalt *overføringsfunksjon*. Ideelt sett er forsterkeren *lineær*, det vil si at den har en lineær overføringsfunksjon $f(x) = ax$. La oss anta at $a = 3$ (det spiller ingen rolle), slik at en ideell forsterker har $f(x) = 3x$ som overføringsfunksjon. Lyden går da gjennom forsterkeren uten å endres i det hele tatt, bortsett fra at amplityden tredobles.

Alle virkelige forsterkere har imidlertid en større eller mindre grad av forvrengning, slik at $f(x)$ avviker fra en lineær funksjon. En typisk effekt er *klipping*, det vil si at når inngangssignalet får for stor amplityde går forsterkeren i *metning* og nekter å slippe igjennom mer lyd.

Transistorforsterkere klipper *hardt*, det vil si at signalet er noenlunde intakt opp til et visst nivå, men så kuttes det brått. Den stygge, sprakende forvrengningen er velkjent for de fleste.

Forsterkere med radiorør klipper derimot mere *bløtt*, det vil si at det er en gradvis overgang fra ikke-klipping til klipping. Dette lager en spesiell sound som særlig gitarister er glade i. Det finnes til og med egne effektbokser som etterlikner slik "tube amp sound". Vi skal lage en slik boks på en datamaskin, og da må vi lage oss en overføringsfunksjon som maskinen kan sende lyden igjennom for å oppnå den riktige effekten. Den norske komponisten Anders Vinjar har holdt på med nettopp dette.

La oss forsøke med et polynom. I utgangspunktet ønsker vi at forsterkeren skal være lineær, så vi setter $f(x) = 3x$ (figur 2.4).



Figur 2.4: Overføringsfunksjon for lineær forsterker

Det ser jo fornuftig ut, men så må vi prøve å introdusere litt "soft clipping". Dette innebærer at grafen må flate ut gradvis i endene. På den måten vil store x (nær pluss og minus 1) bli begrenset.

Etter litt tenking finner vi ut at vi skal forsøke å trekke fra en x^3, så vi får overføringsfunksjonen $f(x) = 3x - x^3$. Det er ikke så dumt, for som vi ser av grafen for x^3 over ligger den nær null når x er nær null, slik at vår lineæritet ikke ødelegges for små x. For større x blir imidlertid utslagene store, og de vil slå riktig vei dersom vi trekker x^3 fra $3x$ (figur 2.5).

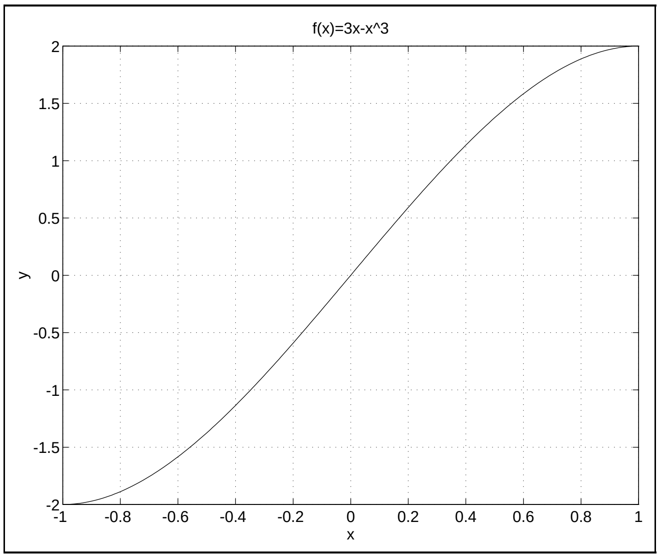
Dette tredjegradspolynomet kan vi sette inn i en MAX-patch eller i et C-Sound-program og oppnå en utmerket "tube amp". Vi har nå sett et eksempel på en lydbehandlingsmetode som kalles *ulineær kurveforming* eller nonlinear waveshaping. I ulineær kurveforming benyttes ofte en spesiell type polynomer som heter *Chebychev-polynomer*. Chebychev-polynomer har pene egenskaper som gjør det lett å forutsi hvordan klangfargen endres.

2.4 Hyperbler og røtter

Funksjonen $f(x) = \frac{1}{x}$ kalles en *hyperbel*. Her må vi huske på at definisjonsområdet ikke inneholder null, fordi divisjon med null er forbudt.

De motsatte funksjonene av potensene over kalles *røtter*. Det tallet som opphøyet i andre potens gir x, heter *kvadratroten* til x, og vi skriver $f(x) = \sqrt{x}$. Kvadratroten til 9 er således 3, kvadratroten til 16 er 4. Igjen må vi passe på definisjonsområdet. Det er forbudt å ta kvadratroten av et negativt tall, for det finnes ingen tall som opphøyet i andre potens gir et tall mindre enn null.

Det tallet som opphøyet i tredje potens gir x, heter *kubikkroten* eller *tredje-roten* av x, og vi skriver $f(x) = \sqrt[3]{x}$. Her er det tillatt å putte inn negative x (finn ut hvorfor!).



Figur 2.5: Overføringsfunksjon for ulineær forsterker

Tilsvarende kan vi definere høyere røtter. De fleste programmeringsspråk har imidlertid ikke egne funksjoner for annet enn kvadratrøtter. Det er lett å lure fram høyere røtter ved å benytte seg av måten ikke-heltallige potenser er definert på. Vi har for en n'te rot at

$$\sqrt[n]{x} = x^{\frac{1}{n}},$$

slik at f.eks. fjerderoten av x kan skrives som $x^{0.25}$.

2.5 Eksponensialfunksjoner, logaritmer

Hvis vi bruker potenser som beskrevet over, men setter x i *eksponenten* i stedet for i grunntallet, får vi en *eksponensialfunksjon.* Dette er en meget nyttig funksjon og er blant de mest brukte til musikkformål.

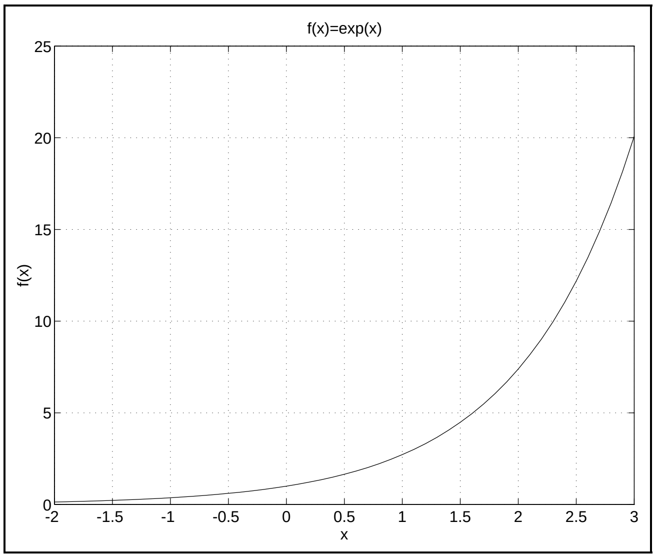
Eksempler på slike funksjoner er $f(x) = 2^x$, $f(x) = 3^{4x}$, $f(x) = 3.14^{2x+5}$.

I disse eksemplene brukte vi 2, 3 og 3.14 som grunntall (det tallet vi opphøyer i noe). Siden vi alltids kan skru til disse funksjonene som vi vil ved å multiplisere med faktorer etc., er det på en måte likegyldig hvilket grunntall vi bruker. I matematikken viser det seg at alle formler blir mye enklere dersom vi alltid bruker et magisk tall som heter e som grunntall. Vi skal ikke gå mer inn på årsakene til dette, men vi bør være klar over det, bl.a. fordi datamaskiner oftest har en funksjon $exp(x) = e^x$ som vi kan benytte oss av. Det rare tallet e er forøvrig omtrent 2.71, og er like viktig som π for matematikerne.

Funksjonen $f(x) = e^x$ stiger mer og mer, og brattere og brattere. Før eller senere, for store nok x, blir den faktisk større enn alle tenkelige polynomer. Grafen er vist i figur 2.6 (legg merke til at $e^0 = 1$):

Noen regneregler for funksjoner med potenser kan være nyttige:

$$x^{a+b} = x^a x^b,$$



Figur 2.6: Eksponensialfunksjonen e^x

$$x^{-a} = \frac{1}{a},$$
$$x^{a-b} = \frac{x^a}{x^b},$$
$$x^{ab} = x^{a^b}.$$

De omvendte funksjoner av eksponensialfunksjoner heter *logaritmer*. Det finnes forskjellige typer logaritmer, avhengig av hvilket grunntall vi snakker om. Logaritmen til x, med grunntall 10, skriver vi $f(x) = log_{10}x$, og vi mener da at $f(x)$ er det tallet vi må opphøye 10 i for å få x (det der krever kanskje litt fordøying). Vi har dermed f.eks. at $log_{10}100 = 2$ og $log_{10}1000 = 3$.

Logaritmen med grunntall e kalles den *naturlige* logaritmen, og skrives $f(x) = lnx$. Vi har dermed at $ln(e) = 1$. Den logaritmefunksjonen som oftest finnes på datamaskiner, og som gjerne heter *log* er den naturlige logaritmen.

En viktig regneregel for logaritmer er at $ln(x^a) = aln(x)$. Siden $ln(x)$ er den motsatte (inverse) funksjonen av e^x, har vi også at $e^{ln(x)} = x$; hvis vi tar e og opphøyer i den naturlige logaritmen til x, får vi simpelthen x tilbake. Disse to regnereglene kan vi bruke til å lage oss potenser dersom programmeringsspråket vårt ikke har dette tilgjengelig (dette er faktisk svært vanlig):

$$x^a = e^{ln(x^a)} = e^{aln(x)}.$$

Eksempel 2: Beregning av frekvens fra notenummer

Et av de vanligste spørsmålene som dukker opp i denne bransjen er følgende: Vi har et MIDI notenummer (60 for C, 61 for Ciss etc.). Hvilken frekvens i Hz tilsvarer dette?

Vi skal ikke gå igjennom teorien om skalaer her, men anta at vi vil bruke "equal temperment". Vi går ut fra at kammertonen A har frekvensen 440 Hz. Nå er det slik at vi vil få neste halvtone ved å multiplisere frekvensen med en konstant k. Når vi har multiplisert 12 ganger skal vi ha gått opp en oktav, dvs. at frekvensen skal være doblet. For å oppnå dette setter vi $k = \sqrt[12]{2} = 1.0594631$. Hvis MIDI-noten til kammertonen er 69, får vi frekvensen

$$f = 440 \cdot 1.0594631^{n-69}.$$

for MIDI-note n. Dette er en eksponensialfunksjon, og dette viser at øret ikke oppfatter frekvens lineært. Bemerk også at n ikke behøver å være et helt tall, så mikrotonalitet håndteres problemfritt med denne formelen.

En datamaskin vil kanskje ha formelen på denne måten:

```
f=440*pow(1.0594631,(n-69));
```

eller i verste fall slik:

```
f=440*exp((n-69)*log(1.0594631));
```

Eksempel 3: Beregning av notenummer fra frekvens

Det å finne et MIDI-notenummer fra en frekvens i Hz er også ofte nyttig. Da tar vi simpelthen formelen fra forrige eksempel og løser likningen med hensyn på n. Dette gjøres ved å ta den naturlige logaritmen på hver side av likhetstegnet, og deretter anvende regnereglene som er gitt over:

$$n = \frac{ln(f) - ln(440)}{ln(1.0594631)} + 69.$$

Dette er et logaritme-funksjon. På en datamaskin skriver man noe slikt:

```
n=(log(f)-log(440))/log(1.0594631)+69;
```

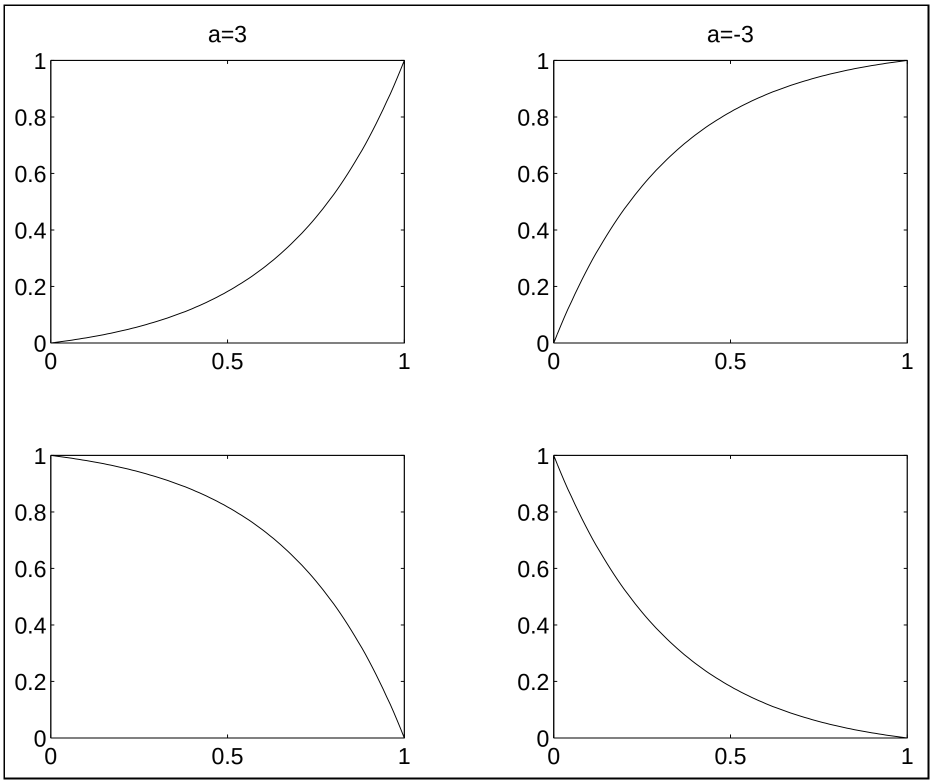
Eksempel 4: Eksponensielle kontrollfunksjoner

Som vi har sett, så er eksponensialfunksjoner fornuftige å bruke dersom vi vil at øret skal oppfatte en jevnt glidende tonehøyde. Amplityder bør også styres med slike funksjoner, fordi øret oppfatter lydstyrke på en liknende måte. Eksponensialfunksjoner er dermed mye brukt i envelopes og andre kontrollfunksjoner. Spesielt velkjent er envelopes av ADSR-typen (Attack-Decay-Sustain-Release). Slike envelopes består av tre eksponensialfunksjoner som er satt sammen, samt en konstant Sustain-fase.

I C-Sound og mange andre systemer finnes en funksjonsgenerator som lar brukeren spesifisere start- og endepunkter for slike eksponensialfunksjoner. Hvis vi programmerer f.eks. i C, kan vi bruke noe slikt:

$$f(x) = startverdi + (sluttverdi - startverdi)\frac{1 - e^{ax}}{1 - e^{a}}.$$

Hvis vi lar x løpe fra 0 til 1, vil $f(x)$ løpe fra *startverdi* til *sluttverdi*, med en eksponensiell form som bestemmes av a (positiv eller negativ). Figur 2.7 viser noen eksempler.



Figur 2.7: Eksponensielle kontrollfunksjoner

Kapittel 3

Trigonometri

Trigonometri betyr trekantmåling. Når vi studerer forholdene mellom sidene i rettvinklede trekanter (trekanter hvor ett hjørne er rettvinklet), dukker det opp såkalte trigonometriske funksjoner: *Sinus*, *cosinus*, *tangens* etc. Disse funksjonene er morsomme i seg selv, men det underlige er at de samme funksjonene dukker opp overalt i fysikken. Trigonometriske funksjoner brukes til å beskrive f.eks. en pendels bevegelser og bølger på havet. Trigonometriske funksjoner er sentrale også opp i forbindelse med akustikk og vibrasjon i musikkinstrumenter, og er derfor av spesiell interesse for oss. Summetonen i telefonen er en temmelig ren sinusfunksjon.

3.1 Rettvinklede trekanter

En rettvinklet trekant består av de to sidene som ligger inntil den rette vinkelen (de to *katetene*) og den siste, "skrå" siden (*hypotenusen*). Vi kaller lengden til katetene for a og b, og lengden til hypotenusen for c. En av vinklene kaller vi ϕ. Det der var en gresk bokstav som heter *phi*. Den ser flott ut på papiret, og er "standardbokstaven" for vinkler.

Forholdene mellom a, b og c er gitt ved *Pythagoras' teorem*:

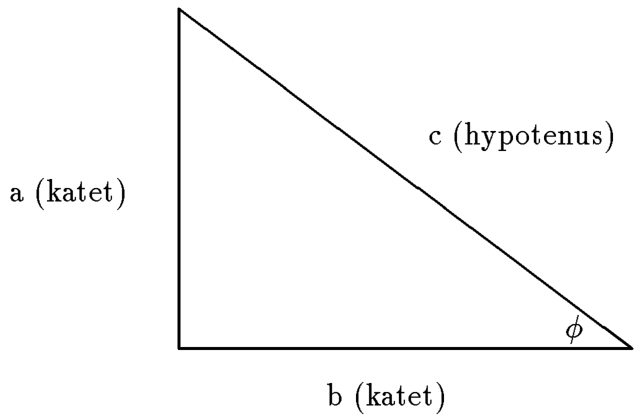
$$c^2 = a^2 + b^2,$$

som vi uttaler "kvadratet på hypotenusen er summen av kvadratene på katetene".

Eksempel 5: Avstanden til en lydkilde

Anta at en lytter er plassert i origo i et koordinatsystem, altså i punktet (0,0). Vi ønsker å simulere at en lydkilde beveger seg i planet på en eller annen måte, og vi har allerede beregnet (x,y)-koordinatene til lydkilden.

Som en del av simuleringen må vi kontrollere tidforsinkelsen fra lyden sendes ut til den mottas av lytteren. Dette vil gi en stilig Doppler-effekt når lydkilden beveger seg. Denne tidsforsinkelsen avhenger direkte av avstanden til lydkilden. Vi må også kunne kontrollere



Figur 3.1: Rettvinklet trekant

amplityden, som faller som en direkte hyperbelfunksjon av avstanden (lydstyrken vil da falle med kvadratet av avstanden).

Alt dette koker dermed ned til at vi må finne avstanden s fra punktet (0,0) til punktet (x,y). Ved å tegne opp situasjonen i et koordinatsystem, ser vi at vi får en rettvinklet trekant, og fra Pythagoras har vi

$$s = \sqrt{x^2 + y^2}.$$

3.2 De trigonometriske funksjonene

Du er sikkert vant til at vinkler måles i *grader*. Når vi arbeider med matematikk og datamaskiner måler vi derimot nesten alltid vinkler i *radianer*.

180 grader tilsvarer π radianer, slik at 1 radian er $180/\pi$, ca 57 grader (husk at $\pi = 3.141592654...$). Eksempelvis er da 30 grader $\pi/6$ radianer, 45 grader $\pi/4$ radianer, 90 grader $\pi/2$ radianer. Vi kutter som oftest ut enheten radianer, og sier f.eks. at vinkelen $\phi = 1.28$.

La oss gå tilbake til tegningen av den rettvinklede trekanten. Vi *definerer* nå en funksjon $f(x) = sin(\phi)$ på denne måten:

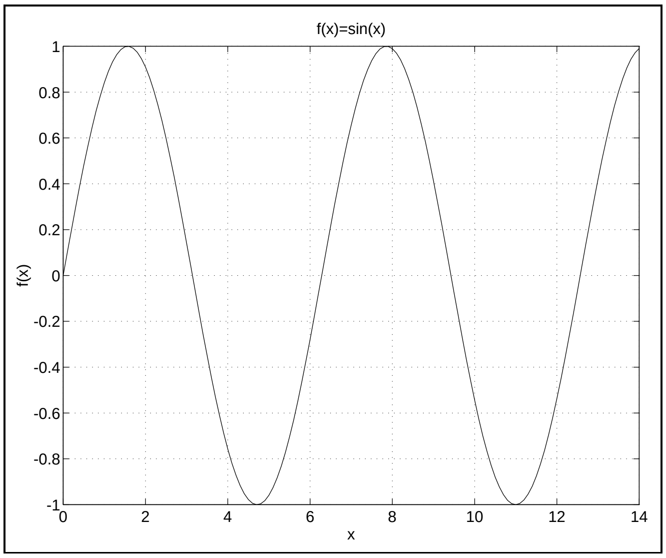
$$sin(\phi) = \frac{a}{c},$$

som vi uttaler "sinus til en vinkel er forholdet mellom den motstående katet og hypotenusen". Ettersom ϕ varierer, må lengdene til sidene variere for at vi fortsatt skal ha en rettvinklet trekant. Forholdet mellom motstående katet og hypotenusen vil alltid følge den samme funksjonen $sin(\phi)$, uansett hvor stor trekanten er.

I trekanten over gir det ikke mening å snakke om ϕ større enn $\pi/2$. Vi utvider derfor definisjonsområdet til å omfatte alle mulige tall, positive og negative, og definerer hvordan

$sin(x)$ skal variere her (geometrisk gjøres dette ved å studere den såkalte enhetssirkelen, men det hopper vi over nå).

Sinusfunksjonen er vist i figur 3.2.



Figur 3.2: Sinusfunksjonen

Det viktigste å legge merke til her er at sinusfunksjonen er *periodisk*, med periode $2\pi = 6.28$. Dette betyr at funksjonen gjentar seg bortover i det uendelige. Hvis vi fester en blyant på en svingende pendel, og lar blyanten tegne på et langt papir som vi trekker forbi pendelen, så vil vi få en graf som denne. En pendel beskrives altså ved en sinusfunksjon, og i naturen er det svært mange prosesser som oppfører seg som pendler.

Vi kan styre *frekvensen*, dvs. hvor mange ganger sinuskurven gjentar seg hvert sekund, ved å lage oss funksjoner av typen

$$F(t) = sin(2\pi ft),$$

der f er frekvensen i svingninger pr. sekund og t er tiden i sekunder. Enheten "svingninger pr. sekund" har fått et eget navn, *Hertz*, forkortet Hz. Hvis frekvensen f.eks. er 5 Hz, ser funksjonen ut som vist i figur 3.3.

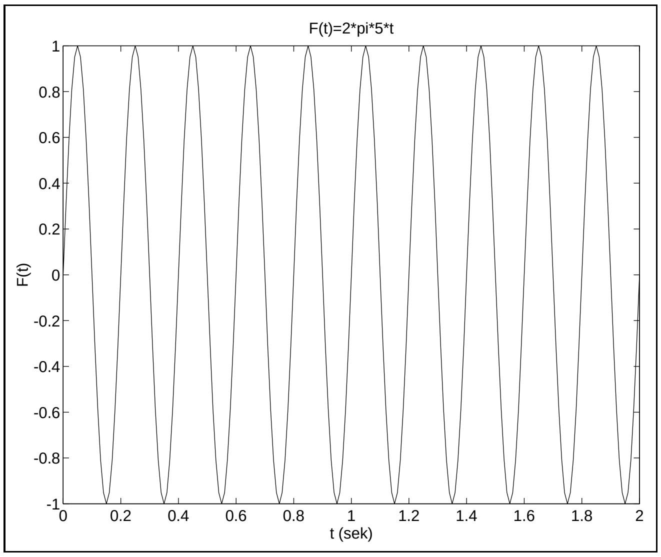
Tilbake til den rettvinklede trekanten igjen. Vi definerer nå en ny funksjon $f(x) = cos(\phi)$ på denne måten:

$$cos(\phi) = \frac{b}{c},$$

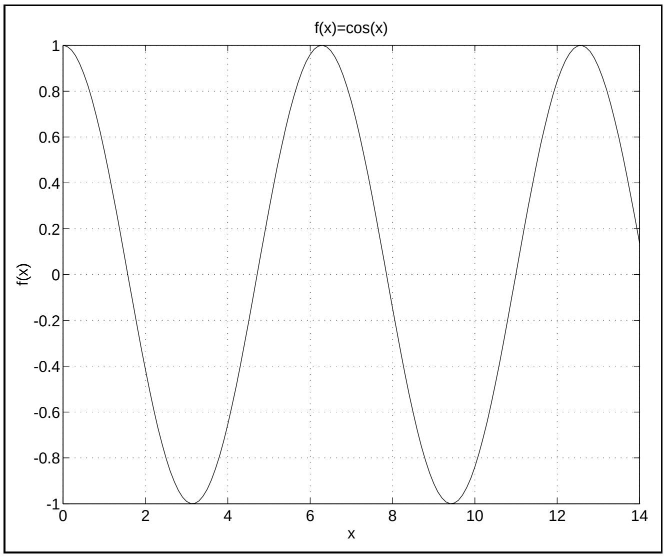
som vi uttaler "cosinus til en vinkel er forholdet mellom den hosliggende katet og hypotenusen".

Vi utvider igjen definisjonsområdet, og får funksjonen vist i figur 3.4.

Denne grafen er identisk med grafen for sinus, bortsett fra at den er forskjøvet et stykke i horisontal retning. Vi sier at sinus og cosinus, uansett frekvens, begge er *sinusoider*, men at de har forskjellig *fase*. Hvis vi spiller en sinusoid ut gjennom en høyttaler, vil vi høre



Figur 3.3: Sinusoid funksjon



Figur 3.4: Cosinusfunksjonen

en ren, myk tone. Tonehøyden vil avhenge av frekvensen, men fasen vil ikke ha noen betydning for hvordan lyden oppfattes.

Det må bemerkes at man ofte bruker ordet "sinus" om alle sinusoider, uavhengig av frekvens og fase. Dette er ikke helt korrekt, fordi sinus er en strengt definert funksjon med en bestemt periode og fase.

3.3 Fourier-rekker

Vi har nå definert sinus og cosinus, og vi har sett at begge er bølgeformede, periodiske funksjoner.

Enhver periodisk funksjon kan konstrueres ved å summere et antall sinus- og cosinus-funksjoner, hver med en frekvens som er et heltallsmultiplum av frekvensen til den periodiske funksjonen. Dette er *Fourier's teorem*, og en sum av sinuser og cosinuser som bygger opp en funksjon heter en *Fourier-rekke*. Fourier's teorem er et av de dypeste og vakreste resultater i matematikken.

Dette er veldig viktig når vi driver med lyd, for øret oppfatter som hovedregel en lyd i *frekvensdomenet*, dvs. som sinusoider. Dette er nærmere forklart i heftet om akustikk.

Bemerk at vi *enten* kan spesifisere en Fourier-rekke som en sum av ekte sinus- og cosinus-funksjoner, *eller* som en sum av sinusoider der fasen er oppgitt. Dette er likegyldige betraktningsmåter, fordi vi kan lage en sinusoid med hvilken som helst fase ved å addere en sinus og en cosinus med de riktige amplityder.

Dersom funksjonen vi ønsker å konstruere har sprang (diskontinuiteter), må vi i prinsippet addere et uendelig antall sinusoider. Dette er imidlertid ikke av interesse for oss. Alle digitale lyder er nemlig *båndbegrensede*, og inneholder bare frekvenser opp til halve samplingfrekvensen.

Fourier's teorem omhandler periodiske funksjoner. Men hva så med ikke-periodiske funksjoner? De kan også håndteres, på en av to måter. Enten sier vi at en funksjon bare er definert i et avgrenset område (en lyd varer f.eks. bare en viss tid). Vi utvider så funksjonen periodisk, dvs. at vi *later som om* funksjonen gjentar seg periodisk utenfor det opprinnelige definisjonsområdet. Funksjonen er så klar for Fourier's teorem, og vi kan etterpå glemme funksjonsverdiene utenfor det opprinnelige området.

Den andre angrepsmåten er å si at funksjonen faktisk *er* periodisk, men med uendelig lang periode. Grunnfrekvensen blir da uendelig liten, og de øvrige sinusoidene kommer uendelig nær hverandre. Fourier-rekka går da over til et *Fourier-integral*.

Disse tingene blir forhåpentlig mer intuitivt forståelige når vi ser på anvendelsene innen akustikken.