MAT301 Matematikk III
forelesningsnotater og oppgaver

Dag Nylund

20. januar 2022

Innhold

Kapittel 1

Introduksjon

Dette notatet er forelesningsnotater og oppgaver som dekker de fleste av læringsmålene i Matematikk 3, Spillskolen, Høgskolen i Innlandet. Temaer Quaternioner og Spillteori dekkes av annen litteratur.

Forelesningsnotater er ikke en bok, og mye kunne vært skrevet i tillegg til det som står her. Kom gjerne med tilbakemeldinger på skrive- og regnefeil og innspill til neste revisjon. Planen for semesteret er som oversikten nedenfor (og i Canvas) viser:

1.1 Uke 2-3

Vi starter med kapittel 2 Funksjoner av to variable. Vi trenger funksjoner av to variable til 3d scener. Dette kapitlet er delvis et komprimert sammendrag av [5], boka fra matematikk 1-2. Boka anbefales brukt som tilleggslitteratur. Programmering av en datastruktur for en flate til bruk i 3d-programmering. Senere skal vi se at vi ikke alltid har en kjent funksjon, men måledata.

1.2 Uke 4

kapittel 3 - Matriser, affint rom og homogene koordinater. Noe repetisjon fra matematikk 2 (skalering, translasjon, rotasjon). Projeksjonsmatrise/perspektivdivisjon. Litt om projeksjoner og projektive rom, og affine transformasjoner og rom.

1.3 Uke 5

kapittel 3 minste kvadraters metode.

1.4 Uke 6

kapittel 4 interpolasjon og funksjonsbasiser.

1.5 Uke 7

kapittel 4 Hermite interpolasjon, Bezier kurver, deCasteljau algoritmen.

1.6 Uke 8

kapittel 5 Affint rom, parametriske kurver og Neville's algoritme.

1.7 Uke 9

Studieuke.

1.8 Uke 10-11

kapittel 6 Flater og trianguleringer, barysentriske koordinater.

1.9 Uke 12

Quaternioner.

1.10 Uke 13-14

kapittel 7 Introduksjon til B-splines.

1.11 Uke 15

Påske.

1.12 Uke 16

Introduksjon til spillteori.

1.13 Uke 17

Oppsummering/sammendrag/eksamensforberedelser.

Kapittel 2

Funksjoner av to variable $z = f(x, y)$

2.1 Funksjoner av to variable f(x,y)

- Vi trenger funksjoner av to variable til 3d scener.
- Analyse for funksjoner av en variabel kan lett utvides til funksjoner av to variable.
- Kontinuitet, grenseverdier, deriverbarhet, maks og min.
- Senere skal vi se at vi ikke alltid har en kjent funksjon, men måledata.
- Visualisering: Graf, nivåkurver, snitt.

2.1.1 Grenseverdier, kontinuitet og deriverbarhet

For en funksjon $f(x)$ gjelder at f er kontinuerlig for x=a hvis og bare hvis $lim_{x\to a} f(x) = f(a)$. Husk at vi har ensidige grenseverdier som begge skal være like. Når vi har en funksjon av to variable må dette gjelde samme hvilken retning vi nærmer oss et punkt (a,b) og f(x,y) er kontinuerlig for (x,y)=(a,b) hvis og bare hvis

$$\lim_{(x,y)\to(a,b)} f(x, y) = f(a, b)$$

En funksjon f(x) er deriverbar i x=a hvis og bare hvis høyresiden nedenfor eksisterer, og den deriverte er

$$f'(a) = \lim_{\Delta x\to 0} \frac{f(a + \Delta x) - f(a)}{\Delta x}$$

De ensidige grenseverdiene må være like, og noen ganger brukes skrivemåten $f'(x) = \frac{df}{dx}$ for å vise hva vi deriverer med hensyn på. Når vi har en funksjon av to variable, kan vi derivere med hensyn på enten den ene eller den andre. Dette angir vi med å erstatte $f'(x) = \frac{df}{dx}$ med $f_x = \frac{\partial f}{\partial x}$ og tilsvarende for y.

2.1.2 Partiellderiverte

Uttrykkene for de partiellderiverte blir da $\frac{\partial f}{\partial x}(x,y)$ og $\frac{\partial f}{\partial y}(x,y)$. Når vi partiellderiverer med hensyn på en variabel betraktes den andre som konstant (fordi det skjer ingen endring av y i x-retning og omvendt).

Eksempel

$f(x,y) = 2x^2y$. Da blir de partiellderiverte $\frac{\partial f}{\partial x} = 4xy$ og $\frac{\partial f}{\partial y} = 2x^2$.

Eksempel

$f(x,y) = \frac{2x}{y}$. Da blir de partiellderiverte $\frac{\partial f}{\partial x} = \frac{2}{y}$ og $\frac{\partial f}{\partial y} = -\frac{2x}{y^2}$.

2.1.3 Tangentplan og linearisering

Ligningen for tangentplanet til en funksjon kalles lineariseringen. Lineariseringen til en funksjon av to variable $f(x,y)$ om punktet (a,b) er gitt ved

$$L(x,y) = f(a,b) + f_x(a,b)(x-a) + f_y(a,b)(y-b) \tag{2.1}$$

En forutsetning er at grenseverdien nedenfor eksisterer og er 0 (se [5], 11.2.4).

$$\lim_{(x,y)\to(a,b)} \frac{f(x,y) - L(x,y)}{\sqrt{(x-a)^2 + (y-b)^2}} = 0$$

Eksempel

Finn tangentplanet til funksjonen $f(x,y) = xy$ i punktet (1,1).
$f_x(x,y) = y$
$f_y(x,y) = x$
$f(1,1) = 1$
$f_x(1,1) = 1$
$f_y(1,1) = 1$
$L(x,y) = f(1,1)+f_x(1,1)(x-1)+f_y(1,1)(y-1) = 1+(x-1)+(y-1) = x+y-1$

Vi skal se nærmere på tangentplan og normalvektor i kapittel 6.

2.1.4 Gradient

Gradienten til en funksjon f(x,y) er gitt ved

$$\nabla f = [\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}] \tag{2.2}$$

Gradienten er en vektor. Den kan regnes ut i et hvilket som helst punkt. Gradienten er normalvektor til nivåkurven for en funksjon av to variable (se [5] teorem 11.2.19).

2.1.5 Andrederiverte og Hessematrisen

Vi husker en del metoder for å analysere funksjoner av en variabel

- Derivere og sette f'(x)=0 for å finne maks-eller min-punkt.
- Finne ligningen til tangenten i et punkt - tilsvarer å finne linearisering/tangentplan
- Den deriverte har stigningstall 0 i ekstremalpunkter (her: stasjonære punkter, de partiellderiverte er 0)
- Den andrederiverte sier noe om krumningen (f.eks x^2 og $-x^2$) - vi skal nå gjøre noe tilsvarende andrederiverttesten for en funksjon av to variable.

Hvis funksjonen er to ganger deriverbar, skriver vi de partiellderiverte slik:

$$\frac{\partial^2 f}{\partial x^2} \qquad \frac{\partial^2 f}{\partial y \partial x} \qquad \frac{\partial^2 f}{\partial x \partial y} \qquad \frac{\partial^2 f}{\partial y^2}$$

I første tilfelle deriverer vi med hensyn på x to ganger. I det andre tilfellet deriverer vi med hensyn på y og deretter med hensyn på x. I det tredje tilfellet deriverer vi med hensyn på x først og deretter med hensyn på y. Dette gir samme resultat. I siste tilfelle deriverer vi med hensyn på y to ganger.

De andreordens partiellderiverte kan brukes til å klassifisere stasjonære punkter (vi husker at de stasjonære punktene er der hvor de partiellderiverte begge er null). Vi setter da opp den såkalte Hessematrisen og regner ut determinanten i de aktuelle punktene.

$$H = \begin{bmatrix} \frac{\partial^2 f}{\partial x^2} & \frac{\partial^2 f}{\partial y \partial x} \\ \frac{\partial^2 f}{\partial x \partial y} & \frac{\partial^2 f}{\partial y^2} \end{bmatrix} = \begin{bmatrix} f_{xx} & f_{yx} \\ f_{xy} & f_{yy} \end{bmatrix} \tag{2.3}$$

$$\Delta = det|H| = f_{xx} \cdot f_{yy} - f_{xy} \cdot f_{yx} \tag{2.4}$$

1. $\Delta > 0$ og $f_{xx} < 0 \Rightarrow$ lokalt maksimum.
2. $\Delta > 0$ og $f_{xx} > 0 \Rightarrow$ lokalt minimum.
3. $\Delta < 0 \Rightarrow$ sadelpunkt.
4. $\Delta = 0$ - Metoden gir ingen konklusjon.

2.1.6 Eksempel

Vi fortsetter med eksempel 2.1.2. $f_x = 0 \Leftrightarrow x = 0 \vee y = 0$ og $f_y = 0 \Leftrightarrow x = 0$. Når x=0 er både $f_x = f_y = 0$ og vi får stasjonære punkter altså for x=0.

De andreordens partiellderiverte blir $f_{xx} = 4y,\ f_{xy} = 4x,\ f_{yx} = 4x,\ f_{yy} = 0$. Vi prøver med å klassifisere det stasjonære punktet ved hjelp av Hessematrisen. Vi får $\Delta = 0$ og ingen konklusjon fra metoden.

2.1.7 Eksempel

Gitt $f(x,y) = x^2y^2 - 2xy^2 + y^2 - 1$. Finn de stasjonære punktene og klassifiser dem.

$f_x = 2xy^2 - 2y^2 = 2y^2(x-1)$ og $f_y = 2x^2y - 4xy + 2y = 2y(x^2 - 2x + 1)$
$f_x = 0 \Rightarrow y = 0 \vee x = 1$
$f_y = 0 \Rightarrow y = 0 \vee x = 1$
Dette gir stasjonære punkter for x=1 (samme hva y er) og for y=0 (samme hva x er).

$f_{xx} = 2y^2,\ f_{xy} = 4y(x-1),\ f_{yy} = 2x^2 - 4x + 2 = 2(x-1)^2$

Vi setter inn (1,0) og får $\Delta = 0$, så heller ikke her gir metoden noen konklusjon.

2.1.8 Franke's testfunksjon

Franke's testfunksjon er en mye brukt funksjon i numerisk analyse:

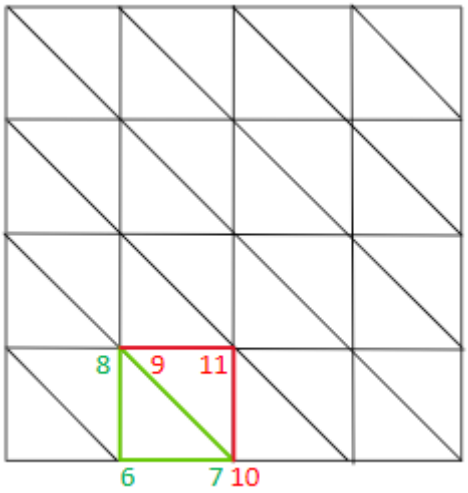
$$\begin{aligned} f(x,y) &= \frac{3}{4}e^{-((9x-2)^2+(9y-2)^2)^2)/4} \\ &+ \frac{3}{4}e^{-((9x+1)^2/49-(9y+1))/10} \\ &+ \frac{1}{2}e^{-((9x-7)^2+(9y-3)^2)/4} \\ &- \frac{1}{5}e^{-((9x-4)^2+(9y-7)^2)} \end{aligned} \tag{2.5}$$

2.1.9 Programmering - tegne grafen til f(x,y)

Vi skal se på et eksempel på hvordan vi kan tegne grafen til en funksjon f(x,y). Dette eksemplet er også en del av notatene fra 3d-programmering 18/1-19. Det er typisk et tema som vi berører i begge kursene.

Vi skal konstruere og tegne en 3d flate. I eksemplet nedenfor er området [0, 1] x [0, 1] delt opp i 4x4 kvadrater. Det gir 32 trekanter, og 32 * 3 = 96 vertices. Funksjonen $z = f(x,y) = \sin \pi x \cdot \sin \pi y$ beregnes i hvert vertex. Disse legges inn i en std::vector etter hverandre slik at tre og tre vertices definerer en trekant for glDrawArrays(). Den grønne og den røde trekanten på figur 2.2.3 viser indekseringen for de gitte vertexene i en std::vector mVertices for $x = x_{min}+h$, $y = y_{min}$ i indre løkke.

```
float xmin=0.0f, xmax=1.0f, ymin=0.0f, ymax=1.0f, h=0.25f;
for (auto x=xmin; x<xmax; x+=h)
    for (auto y=ymin; y<ymax; y+=h)
    {
        float z = sin(M_PI*x)*sin(M_PI*y);
        mVertices.push_back(Vertex{x,y,z,x,y,z});
        z = sin(M_PI*(x+h))*sin(M_PI*y);
        mVertices.push_back(Vertex{x+h,y,z,x,y,z});
        z = sin(M_PI*x)*sin(M_PI*(y+h));
```



Figur 2.1: Regulær triangulering av [0,1]x[0,1]

```
        mVertices.push_back(Vertex{x,y+h,z,x,y,z});
        mVertices.push_back(Vertex{x,y+h,z,x,y,z});
        z = sin(M_PI*(x+h))*sin(M_PI*y);
        mVertices.push_back(Vertex{x+h,y,z,x,y,z});
        z = sin(M_PI*(x+h))*sin(M_PI*(y+h));
        mVertices.push_back(Vertex{x+h,y+h,z,x,y,z});
    }
```

Funksjonen kan lett byttes ut. Metoden kan også brukes til å lage en plan flate med z = f(x,y)=0.

2.2 Dobbeltintegral

Dobbeltintegraler kan tolkes som volumet under en flate, regnet over et avgrenset område i xy-planet. Som et eksempel skal vi se hvordan vi regner ut volumet under $f(x,y) = \sin x \sin y$ over området $[0, \frac{\pi}{2}] \times [0, \frac{\pi}{2}]$. Når vi setter opp dobbeltintegralet, får vi dette uttrykket:

$$\int_0^{\frac{\pi}{2}} \int_0^{\frac{\pi}{2}} \sin x \sin y \; dxdy$$

Det innerste integralet blir
$\int_0^{\frac{\pi}{2}} \sin x \sin y \; dx = [-\cos x \sin y]_0^{\frac{\pi}{2}} = 0 - (-1 \cdot \sin y) = \sin y$, slik at

$$\int_0^{\frac{\pi}{2}} \int_0^{\frac{\pi}{2}} \sin x \sin y \; dxdy = \int_0^{\frac{\pi}{2}} \sin y \; dy = [-\cos y]_0^{\frac{\pi}{2}} = 0 - (-1) = 1.$$

2.2.1 Eksempel

$$\int_0^1 \int_0^2 x^2 y dx dy = \int_0^1 \left[\frac{x^3}{3} y\right]_0^2 dy = \int_0^1 \left[\frac{8}{3} y\right]_0^2 dy = \left[\frac{8}{3} \frac{y^2}{2}\right]_0^1 = \frac{4}{3}$$

2.2.2 Eksempel

Integralet $\int_0^2 \int_0^1 yx^2 dydx$ gir samme resultat. Det er fordi vi kan bytte om dx og dy hvis vi samtidig bytter om integralene med integrasjonsgrenser.

Det er litt vanskeligere når vi ikke har et rektangulært område å integrere over, se [5], 11.4.2-11.4.3.

2.2.3 Eksempel: Numerisk integrasjon

Integraler må ofte regnes ut numerisk. Når lengden $dx = dy = h$ på hver kant på er liten, blir arealet $dA = dx \cdot dy$ lite. kan vi regne ut volumet over ett kvadrat ved $h^2 \cdot f(x, y)$ for $(x, y)P=$ punkt i kvadratet. Vi regner da ut volumet av en veldig liten søyle. Når vi gjentar dette og summerer alle søylene, får vi en approksimasjon til et bestemt dobbeltintegral. Se oppgave 2.3.10.

2.3 Oppgaver

2.3.1

Finn de stasjonære punktene til funksjonene nedenfor:

1. $f_1(x, y) = xy$
2. $f_2(x, y) = x^2 + y^2$
3. $f_3(x, y) = \sin xy$
4. $f_4(x, y) = \sin x \cdot \sin y$

2.3.2

Finn tangentplanet til funksjonen $f(x, y) = x^2 + y^2$ i punktet $(\frac{1}{2}, \frac{1}{2})$.

2.3.3 (11.4.2)

Regn ut

$$\int_3^6 \int_{-2}^1 \left(x^2 + xy^2\right) dx\, dy$$

2.3.4 (11.4.2-3)

Gitt $f(x, y) = e^{x+y}$. Regn ut volumet avgrenset av grafen til f og de rette linjene $x \geq 0,\ y \geq 0,\ x + 2y \leq 3$.

2.3.5

Regn ut volumet avgrenset av grafen til funksjonen $f(x, y) = xy$, x-aksen, y-aksen og den rette linjen x+y=1.

2.3.6 (m1-h2016-4)

Gitt funksjonen h(x,y)= xy-x-y+1.s

a) Finn $\frac{\partial h}{\partial x}$ og $\frac{\partial h}{\partial y}$ og bestem det stasjonære punktet.

b) Finn de andreordens partiellderiverte, og bruk disse til å klassifisere det stasjonære punktet.

c) Regn ut volumet avgrenset av grafen til h, x-aksen, y-aksen og den rette linjen $y = 1 - x$.

2.3.7 m3-v2016-1

Gitt funksjonen $f(x, y) = 3x - x^3 - 3xy^2$.

a) Finn lineariseringen til funksjonen f i punktet (-1, 1).

b) Regn ut de andreordens partiellderiverte til $f(x, y)$.

c) Finn og klassifiser de kritiske punktene til funksjonen f.

2.3.8 (m3-v2016-2)

Gitt funksjonen $g(x, y) = 1 - x - y$ og la A være området i xy-planet avgrenset av x-aksen, y-aksen og den rette linjen $x + y = 1$.

Regn ut $\int\int_A g(x, y)\, dA$.

2.3.9

1. Velg en egen funksjon av to variable, et område i xy-planet (eller xz-planet), og en oppdeling. Bruk algoritmen i 2.1.9 og lag en tekstfil med xyz (eventuelt også rgb og uv-koordinater) linjevis. Antall linjer skal stå øverst i fila. Fila skal kunne leses inn av et TriangleSurface objekt i Qt/3d-programmering.

2. Velg parametre til en Lissajous kurve og lag en fil med datapunkter for kurven, på samme måte som for flaten ovenfor.

2.3.10 Numerisk integrasjon oppgave

Regn ut volumet for funksjonen i 2.3.9. Bestem integrasjonsgrenser sjøl.

Bruk 2.3.8 til å regne ut det samme integralet numerisk: Regn ut og summer integralet over alle trekantene du har laget i planet.

Gjenta forrige oppgave med halvert avstand mellom alle vertices.

Skriv en kort rapport hvor du sammenligner svarene over. Utregninger skal være med, og kildekode skal være med.

2.4 Fasit

Det skal være samsvar mellom nummereringen i oppgaver og fasit.

2.4.1

Vi må først finne de partiellderiverte, og deretter løse ligningene vi får når vi setter disse lik 0. $f_1(x,y) = xy$ $\frac{\partial f}{\partial x}(x,y) = y$ og $\frac{\partial f}{\partial y}(x,y) = x$
$\frac{\partial f}{\partial x} = 0 \Leftrightarrow y = 0$ og $\frac{\partial f}{\partial y} = 0 \Leftrightarrow x = 0$. Funksjonen har ett stasjonært punkt (0,0).

Finn tangentplanet til funksjonen $f(x,y) = x^2 + y^2$ i punktet $(\frac{1}{2}, \frac{1}{2})$.
$f_x(x,y) = 2x$
$f_y(x,y) = 2y$
$f(\frac{1}{2}, \frac{1}{2}) = \frac{1}{2}$
$f_x(\frac{1}{2}, \frac{1}{2}) = 1$
$f_y(\frac{1}{2}, \frac{1}{2}) = 1$

$$L(x,y) = \tfrac{1}{2} + 1 \cdot (x - \tfrac{1}{2}) + 1 \cdot (y - \tfrac{1}{2}) = x + y - \tfrac{1}{2}$$

$f_2(x,y) = x^2 + y^2$. De partiellderiverte blir: $\frac{\partial f}{\partial x}(x,y) = 2x$ og $\frac{\partial f}{\partial y}(x,y) = 2y$
$\frac{\partial f}{\partial x} = 0 \Leftrightarrow x = 0$ og $\frac{\partial f}{\partial y} = 0 \Leftrightarrow y = 0$ Funksjonen har ett stasjonært punkt (0,0).

$f_3(x,y) = \sin xy$. Vi finner her de partiellderiverte ved å bruke kjerneregelen: $\frac{\partial f}{\partial x}(x,y) = y \cos xy$ og $\frac{\partial f}{\partial y}(x,y) = x \cos xy$. $\frac{\partial f}{\partial x} = 0 \Leftrightarrow y = 0 \ \vee \cos xy = 0 \Leftrightarrow x = 0 \vee y = 0$. Vi får de samme løsningene fra ligningen $\frac{\partial f}{\partial y} = 0$.

Dette gir

1. (0,0)
2. $\cos xy = 0 \Leftrightarrow xy = \frac{\pi}{2} \Leftrightarrow y = \frac{\pi}{2x}$ fra begge ligninger gir stasjonære punkter over en hyperbel i xy-planet.
3. $\cos xy = 0 \Leftrightarrow xy = -\frac{\pi}{2} \Leftrightarrow y = -\frac{\pi}{2x}$ fra begge ligninger gir stasjonære punkter over en hyperbel i xy-planet.
4. Flere løsninger har vi ikke. $x = 0$ fra ligning 1 sammen med $\cos xy = 0 \Leftrightarrow xy = 1 \Leftrightarrow y = \frac{1}{x}$ fra ligning 2 er en umulig kombinasjon (gir null i nevner).

$f_4(x,y) = \sin x \cdot \sin y$.
Partiellderiverte: $f_x = \frac{\partial f}{\partial x}(x,y) = \cos x \sin y$ og $f_y = \frac{\partial f}{\partial y}(x,y) = \sin x \cos y$.
$f_x = 0 \Leftrightarrow \cos x = 0 \vee \sin y = 0 \Leftrightarrow x = \frac{\pi}{2} \vee x = \frac{3\pi}{2} \vee y = 0 \vee y = \pi$.
$f_y = 0 \Leftrightarrow \sin x = 0 \vee \cos y = 0 \Leftrightarrow x = 0 \vee x = \pi \vee y = \frac{\pi}{2} \vee y = \frac{3\pi}{2}$.
For hver x-verdi i ligning 1 får vi en y-verdi fra ligning 2:
$(\frac{\pi}{2}, \frac{\pi}{2}), (\frac{\pi}{2}, \frac{3\pi}{2}), (\frac{3\pi}{2}, \frac{\pi}{2}), (\frac{3\pi}{2}, \frac{3\pi}{2})$.
Tilsvarende, for hver x-verdi fra ligning fra ligning 2 en y-verdi fra ligning 1:
$(0,0), (0,\pi), (\pi,0), (\pi,\pi)$

2.4.2 (11.4.2)

$$\int_3^6 \int_{-2}^1 \left(x^2 + xy^2\right) dx\, dy = \int_3^6 \left[\frac{1}{3}x^3 + \frac{1}{2}x^2y^2\right]_{-2}^1 dy = \int_3^6 \left[(\frac{1}{3} + \frac{1}{2}y^2) - (-\frac{8}{3} + 2y^2)\right] dy =$$
$$\int_3^6 \left(3 - \frac{3}{2}y^2\right) dy = \left[3y - \frac{y^3}{2}\right]_3^6 = -\frac{171}{2}$$

Integralet er negativt, og et volum må være positivt.

2.4.3 (11.4.2-3)

$$\int_0^3 \int_0^{\frac{3-x}{2}} e^{x+y} dy\, dx = \int_0^3 \left[e^{x+y}\right]_0^{\frac{3-x}{2}} dx = \int_0^3 \left(e^{x+\frac{3-x}{2}} - e^{x+0}\right) dx =$$
$$= \int_0^3 \left(e^{\frac{x+3}{2}} - e^x\right) dx = \left[2e^{\frac{x+3}{2}} - e^x\right]_0^3 = (2e^3 - e^3) - (2e^{\frac{3}{2}} - e^0) =$$
$$e^3 - 2e^{\frac{3}{2}} + 1 = (e^{\frac{3}{2}} - 1)^2.$$

2.4.4

$$\int_0^1 \int_0^{1-x} xy\, dy\, dx = \int_0^1 [\frac{1}{2}xy^2]_0^{1-x}\, dx$$
$$= \int_0^1 \frac{1}{2}x(1-x)^2\, dx$$
$$= \int_0^1 (\frac{1}{2}x^3 - x^2 + \frac{1}{2}x)\, dx$$
$$= [\frac{1}{8}x^4 - \frac{1}{3}x^3 + \frac{1}{4}x^2]_0^1$$
$$= \frac{1}{8} - \frac{1}{3} + \frac{1}{4} = \frac{3-8+6}{24} = \frac{1}{24}.$$

2.4.5 (m1-h2016-4)

a) $\frac{\partial h}{\partial x} = y - 1$ og $\frac{\partial h}{\partial y} = x - 1$. Det stasjonære punktet blir enkelt (1, 1).

b) $\frac{\partial^2 h}{\partial x^2} = 0$, $\frac{\partial^2 h}{\partial x \partial y} = 1$ og $\frac{\partial^2 h}{\partial y^2} = 0$.
$\Delta = \frac{\partial^2 h}{\partial x^2} \cdot \frac{\partial^2 h}{\partial y^2} - (\frac{\partial^2 h}{\partial x \partial y})^2 = -1$.
Det betyr at (1, 1) er et sadelpunkt.

c) Regn ut volumet avgrenset av grafen til h, x-aksen, y-aksen og den rette linjen $y = 1 - x$.

$$\begin{aligned}\int_0^1 \int_0^{1-x} (xy - x - y + 1)\, dy\, dx &= \int_0^1 \left[\frac{1}{2}xy^2 - xy - \frac{1}{2}y^2 + y\right]_0^{1-x} dx \\ &= \int_0^1 \left(\frac{1}{2}x(1-x)^2 - x(1-x) - \frac{1}{2}(1-x)^2 + (1-x)\right) dx \\ &= \int_0^1 (\frac{1}{2}x^3 - \frac{1}{2}x^2 - \frac{1}{2}x + \frac{1}{2})\, dx \\ &= \left[\frac{1}{8}x^4 - \frac{1}{6}x^3 - \frac{1}{4}x^2 + \frac{1}{2}x\right]_0^1 \\ &= \frac{1}{8} - \frac{1}{6} - \frac{1}{4} + \frac{1}{2} = \frac{3-4-6+12}{24} = \frac{5}{24}.\end{aligned}$$

2.4.6 m3-v2016-1

a) (oppgaven er hentet fra [3], 14.10)

$$\frac{\partial f}{\partial x} = 3 - 3x^2 - 3y^2 = 3(1 - (x^2 + y^2)) \textbf{ og } \frac{\partial f}{\partial y} = -6xy$$

Vi har $f(-1,1) = 1$, $\frac{\partial f}{\partial x}(-1,1) = -3$ og $\frac{\partial f}{\partial y}(-1,1) = 6$. Ligningen for tangentplanet blir da

$$L(x,y) = 1 - 3(x+1) + 6(y-1) = -3x + 6y - 8$$

b)

$$\frac{\partial^2 f}{\partial x^2} = -6x,\ \frac{\partial^2 f}{\partial x \partial y} = -6y,\ \textbf{og}\ \frac{\partial f^2}{\partial y^2} = -6x$$

c) $\frac{\partial f}{\partial x} = 0 \Leftrightarrow x^2 + y^2 = 1$, altså ligger de kritiske punktene på enhetsssirkelen. I tillegg $\frac{\partial f}{\partial y} = 0 \Leftrightarrow x = 0 \vee y = 0$.
Dette gir følgende kritiske/stasjonære punkter: (1,0), (0,1), (-1,0) og (0,-1). Vi må da bruke determinanten til Hessematrisen for hvert av disse punktene. Vi regner ut $\Delta = \det(\mathrm{H})$ hvor

$$H = \begin{bmatrix} \frac{\partial^2 f}{\partial x^2} & \frac{\partial^2 f}{\partial y \partial x} \\ \frac{\partial^2 f}{\partial x \partial y} & \frac{\partial^2 f}{\partial y^2} \end{bmatrix} = \begin{bmatrix} f_{xx} & f_{yx} \\ f_{xy} & f_{yy} \end{bmatrix} = 36(x^2 - y^2)$$

1. (1,0) gir $f_{xx} = -6, f_{yx} = 0, f_{xy} = -6, f_{yy} = 0$. $\Delta = 36$ *og* $f_{xx} < 0 \Rightarrow$ lokalt maksimum.
2. (0,1) gir $f_{xx} = 0, f_{yx} = -6, f_{xy} = -6, f_{yy} = -0$. $\Delta = -36 \Rightarrow$ sadelpunkt.
3. (-1,0) gir $f_{xx} = 6, f_{yx} = 0, f_{xy} = 0, f_{yy} = 6$. $\Delta = 36$ *og* $f_{xx} > 0 \Rightarrow$ lokalt minimum.
4. (0, -1) gir $f_{xx} = 0, f_{yx} = 6, f_{xy} = 6, f_{yy} = 0$. $\Delta = -36$ *og* $f_{xx} < 0 \Rightarrow$ sadelpunkt.

2.4.7 (m3-v2016-2)

Vi finner grensene for integralet og setter opp (vi kan bytte grensene og dy med dx hvis vi vil, det kan føre til enklere utregninger):

$$\int_0^{1-x}\int_0^1 (1-x-y)\,dx\,dy = \int_0^1\int_0^{1-x}(1-x-y)\,dy\,dx$$

Vi regner først ut det indre integralet.

$$\int_0^{1-x}(1-x-y)\,dy = \left[y - xy - \frac{1}{2}y^2\right]_0^{1-x} = \frac{1}{2} - x + \frac{1}{2}x^2$$

slik at

$$\begin{aligned}\int_0^1\int_0^{1-x}(1-x-y)\,dy\,dx &= \int_0^1\left(\frac{1}{2}-x+\frac{1}{2}x^2\right)dx \\ &= \left[\frac{x}{2} - \frac{x^2}{2} + \frac{x^3}{6}\right]_0^1 = \frac{1}{6}\end{aligned}$$

Kapittel 3

Matrix4x4

3.1 Numerisk lineær algebra

- Repetisjon av lineære transformasjoner
- Rotasjon, skalering, translasjon
- Repetisjon av affine rom
- Homogene koordinater
- Projektive rom
- Løse lineære ligningssystemer, Gausseliminasjon
- LU-faktorisering
- Vi har fra Matematikk II og lineær algebra metoder som vi kan bruke til 3d grafikk.
- Vi har fra Matematikk I en hel del forskjellige funksjoner som vi hittil ikke har hatt altfor mye praktisk nytte av. Det får vi nå.
- Minste kvadraters metode

3.2 Lineære transformasjoner

Dette er kjent stoff fra [5] (kapittel 9.5, side 486-488), med figurer som viser de aller fleste aktuelle transformasjoner vi får bruk for. Vi skal nå anvende dette og programmere noe av det sjøl. Senere skal vi studere teorien bak *litt* nærmere.

3.2.1 Rotasjoner

Vi husker at når et punkt $(x, y) \in \mathbf{R}^2$ roteres en vinkel θ om origo, så kan vi finne de nye koordinatene til punktet ved å multiplisere matrisen

$$\mathbf{R}(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$

med $[x \;\; y]^T$, slik at de nye koordinatene blir $x' = x\cos\theta - y\sin\theta$ og $y' = x\sin\theta + y\cos\theta$. Hvis for eksempel $\theta = \frac{\pi}{4}$, blir rotasjonsmatrisen

$$\mathbf{R}(\frac{\pi}{4}) = \begin{bmatrix} \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \end{bmatrix}$$

Øving

Eller husker vi faktisk det? Kan vi regne ut noe? Hva blir koordinatene (1,1), (2,1), (2,2) og (1,2) rotert 45 grader om origo?

Rotasjon om origo i planet er det samme som rotasjon om z-aksen. Z-aksen har normalvektor $[0 \;\; 0 \;\; 1]^T$. Hvis vi tilføyer en dimensjon, blir rotasjonsmatrisen for rotasjon en vinkel θ om z-aksen

$$\mathbf{R_z}(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Vi ser at normalvektoren som det roteres om står i tredje kolonne. Tilsvarende matriser for rotasjon om x-aksen og y-aksen blir

$$\mathbf{R_x}(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{bmatrix}$$

og

$$\mathbf{R_y}(\theta) = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix}$$

3.2.2 Homogene koordinater

I [5] kapittel 9.5 står det litt om homogene koordinater i 2d. Ved å bruke homogene koordinater kan translasjon uttrykkes som en matrise multiplisert med en vektor. Når vi bruker homogene koordinater i 3d, bruker vi 4x4 matriser

til transformasjoner i 3d. Med homogene koordinater blir rotasjonsmatrisene ovenfor

$$\mathbf{R_x}(\theta) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta & 0 \\ 0 & \sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.1}$$

$$\mathbf{R_y}(\theta) = \begin{bmatrix} \cos\theta & 0 & \sin\theta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\theta & 0 & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.2}$$

$$\mathbf{R_z}(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.3}$$

Vi gjenkjenner de opprinnelige rotasjonsmatrisene som øvre venstre 3×3 submatriser. Vi må passe på at den endelige transformasjonsmatrisen må sendes til shaderprogrammet med kolonnevise data, se 3.2.8.

3.2.3 Skalering

4×4 matrisen for skalering med faktorer s_x, s_y og s_z i henholdsvis x-retning, y-retning og z-retning er

$$\mathbf{S} = \begin{bmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.4}$$

3.2.4 Translasjon

Den homogene 4×4 matrisen for å representere translasjonen

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} + \begin{bmatrix} dx \\ dy \\ dz \end{bmatrix} = \begin{bmatrix} x + dx \\ y + dy \\ z + dz \end{bmatrix}$$

er

$$\mathbf{T} = \begin{bmatrix} 1 & 0 & 0 & dx \\ 0 & 1 & 0 & dy \\ 0 & 0 & 1 & dz \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.5}$$

Vi skal kort repetere hvordan vi bruker denne matrisen.

Øving

1. Vektoren $[2\ 2\ 1]^T$ skal transleres med $[1\ 2\ 3]^T$. Hva blir translasjonsmatrisen i homogene koordinater? Kontroller ved å regne ut.

2. Vektoren $[4\ 5\ 6]^T$ skal transleres med $[-1\ 1\ 1]^T$. Hva blir translasjonsmatrisen i homogene koordinater? Kontroller ved å regne ut.

Løsning

I homogene koordinater er siste koordinaten til en vektor 0 og siste koordinaten til et punkt 1. Vektoren $[2\ 2\ 1]^T$ må utvides til $[2\ 2\ 1\ 0]^T$ i homogene koordinater. Translasjonsmatrisen multiplisert med vektoren blir

$$\begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 2 \\ 0 & 0 & 1 & 3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 2 \\ 2 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 2 \\ 2 \\ 1 \\ 0 \end{bmatrix}$$

Dette er samme vektoren som før, ingenting er endret etter multiplikasjonen. Men det er jo slik det skal være. En vektor har lengde og retning, og den skal ikke endres selv om vi prøver å flytte posisjonen.

Skal vi derimot translere punktet $(2, 2, 1) \in \mathbf{R}^3$, må vi føye til 1 som siste koordinat. Vi får da

$$\begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 2 \\ 0 & 0 & 1 & 3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 2 \\ 2 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 3 \\ 4 \\ 4 \\ 1 \end{bmatrix}$$

3.2.5 Refleksjon, projeksjon og forskyvning

Se [5].

3.2.6 Sammensatte transformasjoner

Vi kan gjøre både translasjon, rotasjon og skalering, men rekkefølgen (av rotasjon og translasjon) spiller en rolle for resultatet. Det er for eksempel en vanlig erfaring i 3d programmering at man sender et objekt i bane når man egentlig har tenkt å rotere det.

Rotasjon om et punkt

Angel and Schreiner ([1], 4.10.1) beskriver hvordan man kan rotere om et annet punkt enn origo, om for eksempel z-aksen. Dette kan gjøres ved en sammensatt transformasjon: Translater til origo, roter om z-aksen og translater tilbake.

Generell rotasjon

I Angel and Schreiner ([1], 4.10.2) beskrives hvordan en generell rotasjon kan kombineres ved rotasjon om de tre aksene,

$$\mathbf{R} = \mathbf{R_x R_y R_z}$$

Se også [?] om Euler-rotasjoner.

3.2.7 Programmering

Disse tre transformasjonene skal implementeres i en 4×4 matriseklasse, tilsvarende QMatrix4x4 med funksjoner **rotate, translate** og **scale**. Parametrene skal være de samme som i QMatrix4x4 funksjonene. Vi trenger ikke overlaste disse funksjonene.

3.2.8 C++ matriserepresentasjon

Todimensjonale arrayer i C++ ligger lagret radvis, som eksemplet nedenfor viser:

```
#include <iostream>
struct Matrix4x4 {
    float A[4][4];
    Matrix4x4() {
        int k=0;
        // legger inn data radvis
        for (int i=0;i<4;i++)
            for (int j=0;j<4;j++)
                A[i][j] = ++k;
    }
    void print() {
        // utskrift viser at i en todimensjonal
        // array ligger data radvis
        float* tmp=A[0];
        for (int i=0;i<16;i++) {
                std::cout << (*tmp)++ << " ";
        }
    }
};
int main(){
    Matrix4x4 m;
    m.print();
}
```

Vi må huske på dette når vi skal sende data til shaderprogrammet - da skal vi sende kolonnevis data. (Klassen ovenfor er forøvrig en veldig forenklet versjon i forhold til oblig 1 klassen).

3.2.9 Andre funksjoner i matriseklasse

Vi får også bruk for å

- angi perspektivprojeksjoner eller frustum
- sette en matrise til identitetsmatrisen
- finne den transponerte matrisen
- finne den inverse matrisen
- multiplisere to matriser

-
-

Vi trenger altså 4x4 matriser til transformasjoner, og vi får også bruk for å løse 4x4 ligningssystemer. Når vi implementerer en metode for å løse 4x4 ligningssystemer, kan vi egentlig løse større ligningssystemer med tilnærmet uendret kode. Siden vi trenger 4x4 matriser til lineære transformasjoner, er det logisk for oss å slå alt sammen til en Matrix4x4 klasse.

3.3 Gausseliminasjon

Vi får bruk for Gausseliminasjon til løsing av 4x4 ligningssystemer:

$$\mathbf{Ax} = \mathbf{b} \Leftrightarrow \mathbf{x} = \mathbf{A}^{-1}\mathbf{b} \tag{3.6}$$

Skrevet på koordinatform:

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{23} & a_{33} & a_{34} \\ a_{41} & a_{24} & a_{34} & a_{44} \end{bmatrix} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{bmatrix}$$

Funksjonene nedenfor er hentet fra Matriseklassen i oblig1. Funksjonen LU() lager en øvre triangulær matrise, men diagonalelementene er ikke nødvendigvis 1 som i matrisen på trappeform i [5], kapittel 9.

3.3.1 Rader, kolonner og b-vektor

Nedenfor brukes m om rader og n om kolonner, som vanlig. Det som ikke framgår av algoritmen nedenfor er radoperasjoner på høyresiden (b-vektor). Den kan legges til som en ekstra kolonne i matrisen, slik at vi får m=4 og n=5. Da er første del av Gausseliminasjonen nesten i samsvar med [5] bortsett fra at vi ikke får 1-ere på diagonalen.

3.3.2 LU-faktorisering

Dette blir litt ulogisk siden klassen heter Matrix4x4. Dessuten er det en fordel med å ikke bruke en utvidet matrise og radoperasjoner på denne: Det er veldig ofte slik at man får bruk for å løse ligningen 3.6 for flere ulike vektorer $\mathbf{b}$. Derfor er det bedre å gjøre radoperasjoner på $\mathbf{b}$ i del to av algoritmen.

LU-faktoriseringen tar utgangspunkt i elementære radoperasjoner kjent fra [5]:

1. Multiplisere en rad med et tall.
2. Addere et multiplum av en rad til en annen rad.
3. Bytte rekkefølgen av to rader (nedenfor kalt pivotering).

3.3.3 Eksempel

Matrisen nedenfor er den utvidede 3x4 matrisen til et ligningssystem **Ax**=**b**. Radoperasjoner på denne i henhold til [5], kapittel 9 gir

$$\begin{bmatrix} 1 & 2 & 3 & 1 \\ 2 & 3 & 4 & 1 \\ 3 & 4 & 2 & 1 \end{bmatrix} \begin{matrix} \longrightarrow \\ R_2 - 2R_1 \\ R_3 - 3R_1 \end{matrix} \begin{bmatrix} 1 & 2 & 3 & 1 \\ 0 & -1 & -2 & -1 \\ 0 & -2 & -7 & -2 \end{bmatrix} \begin{matrix} \longrightarrow \\ R_3 - 2R_2 \end{matrix} \begin{bmatrix} 1 & 2 & 3 & 1 \\ 0 & -1 & -2 & -1 \\ 0 & 0 & -3 & 0 \end{bmatrix}$$

Vi ser her at vi ikke har 1-ere på diagonalen, men det er likevel enkelt å løse ligningssystemet når vi har det på denne formen. Fra siste ligning finner vi z=0, og når vi setter inn det i andre ligning finner vi y=1. Vi gjentar for øverste ligning og finner x=-1. Vi gjør en ekstra divisjon i hvert steg her sammenlignet med å ha 1-ere på diagonalen. Denne prosessen svarer til solve() nedenfor.

Når vi nuller ut elementene i nedre triangulære del, blir de ikke tilordnet 0 - vi bruker plassen til å lagre koeffisienter i stedet. Så hvis man studerer utskriften av en matrise etter LU-faktoreisering, kan man bli forvirret over at det ikke er 0 i nedre triangulære del.

3.3.4 LU-faktorisering algoritme

Algoritmen nedenfor lager en øvre triangulær matrise *uten* 1-ere på diagonalen og *uten* å tilordne 0 under diagonalelementene. I stedet brukes denne plassen til å lagre faktorene som vi multipliserer en og en rad med for å nulle ut elementene under diagonalelementet. Implementering av pivotering blir ikke gjennomgått her.

```
void Matrix4x4::LU()
{
   for (int k=0; k<m-1; k++)
   {
        // pivot(k);

        // Ved radoperasjoner skal vi oppnå 0-ere under diagonalelementet
        // i alle rader nedenfor (altså kolonne k
        // Vi subtraherer et multiplum av k-te
        // rad fra radene nedenfor, nuller ut kolonner fra venstre

        for (int i=k+1; i<m; i++) {
            // Skal mult med denne og deretter trekke fra rad i
            // denne skal bli null og vi kan lagre faktoren her
            A[i][k] = A[i][k]/A[k][k];

            for (int j=k+1; j<n; j++) {
                // kolonnene til høyre for den som blir nullet ut
                A[i][j] = A[i][j] - A[i][k]*A[k][j];
            }
        }
    }
}
```

Algoritmen nedenfor løser deretter ligningssystemet ved å bestemme verdien for en variabel om gangen, med start nedenfra.

```
// Løser Ax=b, syntaks x = A.solve(b)
// Forutsetter at LU er gjort på A
Vec4 Matrix4x4::solve(Vec4 &b) const
{
    Vec4 x;

    for (int k=0; k<m; k++)
        for (int i=k+1; i<n; i++)
            b[i] = b[i]-A[i][k]*b[k];
    for (int i=n-1; i>=0; i--) {
        x[i] = b[i];
        for (int j=i+1; j<n; j++)
            x[i] = x[i] - A[i][j]*x[j];
        x[i] = x[i]/A[i][i];
    }
    return x;
}
```

Første doble for-løkke gjør radoperasjoner på høyresiden. Den er lettest å forstå dersom man (midlertidig) flytter setningen til slutten av andre for-løkke i LU-faktoriseringsalgoritmen. Da ser man at b-vektoren behandles som en femte kolonne, som kjent fra manuell Gausseliminasjon.

```
            b[i] = b[i]-A[i][k]*b[k];
```

3.3.5 Eksempel

En 4x4 matrise med dens inverse, eksempel til bruk for å teste implementering. Rekkefølgen er byttet om fra opprinnelig notat! Matrisen som nedenfor kalles A^{-1} har $a_{11} = 0$ og vil ikke fungere bra til testing uten pivotering.

$$A = \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \quad A^{-1} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 3 & 2 & 1 & 0 \end{bmatrix}$$

3.3.6 Invers matrise

Gitt en matrise $\mathbf{A} = [a_{ij}]$ og anta at dens inverse er $\mathbf{B} = [b_{ij}]$. Da gjelder

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix} \begin{bmatrix} b_{11} & b_{12} & b_{13} & b_{14} \\ b_{21} & b_{22} & b_{23} & b_{24} \\ b_{31} & b_{32} & b_{33} & b_{34} \\ b_{41} & b_{42} & b_{43} & b_{44} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Vi benevner kolonnevektor j i B ved $\mathbf{b}_j$ og enhetsvektorene $\mathbf{e}_j$ hvor j-te element er 1 og de andre er 0. Vi kan bruke LU-faktorisering og løse

$$\mathbf{Ab_j} = \mathbf{e_j} \Leftrightarrow \mathbf{b} = \mathbf{A^{-1}e_j}, \qquad j = 1, \ldots, 4$$

for å finne den inverse matrisen. Hver gang vi bruker solve(), finner vi en kolonne i den inverse matrisen. Vi kan da bruke de fire løsningsvektorene til å danne den inverse matrisen.

3.4 Minste kvadraters metode (se[5], 9.6)

Minste kvadraters metode kan brukes til å beregne en funksjon som passer til måledata. Vi skal først se på et eksempel hvor vi prøver å finne en rett linje som passer best mulig til måledata.

Dersom vi kun hadde to punkter, kunne vi bruke disse punktene til å bestemme ligningen for den rette linjen gjennom punktene. Vi har $y_1 = ax_1 + b$ og $y_2 = ax_2 + b$, og vi kan finne konstantene a og b. Generelt for m punkter er $y_i = ax_i + b,\ i = 1, 2, \ldots m$. Hvis ikke alle punktene ligger eksakt på linjen, kan vi skrive

$$y_i = ax_i + b + e_i,\ i = 1, 2, \ldots m$$

Med m punkter får vi da følgende ligningssystem:

$$\begin{aligned} y_1 &= ax_1 + b + e_1 \\ y_2 &= ax_2 + b + e_2 \\ &\vdots \\ y_m &= ax_m + b + e_m \end{aligned} \tag{3.7}$$

hvor e_i er et lite feilledd som er positiv hvis y_i ligger over den rette linjen og negativ hvis y_i ligger under den rette linjen. Minste kvadraters metode går ut på å minimere summen av alle avvikene kvadrert:

$$\min\ \Sigma_{i=1}^{m} e_i^2 = \min\ \mathbf{e}^T\mathbf{e}$$

hvor $\mathbf{e}^T = [e_1\ e_2 \ldots e_m]$. Ligning 3.7 blir på matriseform

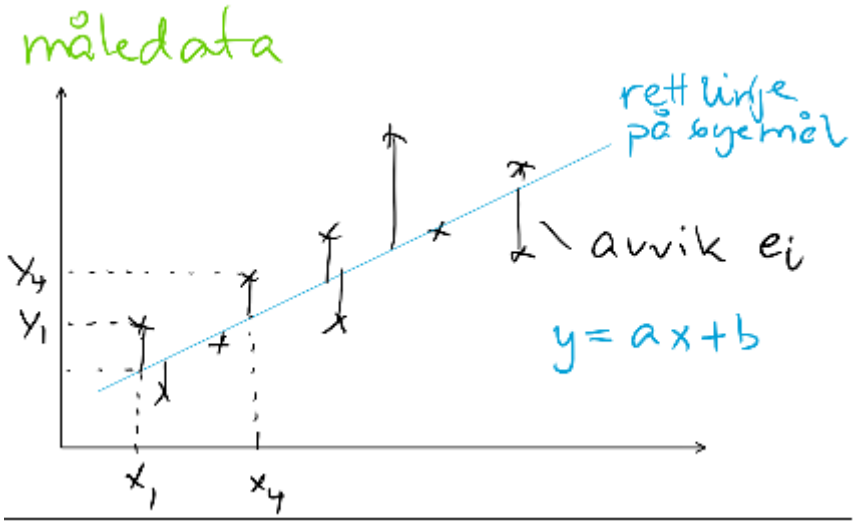
$$\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{e} \tag{3.8}$$

eller

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix} = \begin{bmatrix} x_1 & 1 \\ x_2 & 1 \\ \vdots & 1 \\ x_m & 1 \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} + \begin{bmatrix} e_1 \\ e_2 \\ \vdots \\ e_m \end{bmatrix}$$

Fra 3.7 følger (rettet -$y^T y$ til + $y^T y$)

$$\begin{aligned} \mathbf{e} &= \mathbf{A}\mathbf{x} - \mathbf{y} \\ \mathbf{e}^T\mathbf{e} &= (\mathbf{y} - \mathbf{A}\mathbf{x})^T (\mathbf{y} - \mathbf{A}\mathbf{x}) \\ &= (\mathbf{A}\mathbf{x} - \mathbf{y})^T (\mathbf{A}\mathbf{x} - \mathbf{y}) \\ &= (\mathbf{A}\mathbf{x})^\mathbf{T}(\mathbf{A}\mathbf{x} - \mathbf{y}) - \mathbf{y}^\mathbf{T}(\mathbf{A}\mathbf{x} - \mathbf{y}) \\ &= \mathbf{x}^\mathbf{T}\mathbf{A}^\mathbf{T}\mathbf{A}\mathbf{x} - \mathbf{x}^\mathbf{T}\mathbf{A}^\mathbf{T}\mathbf{y} - \mathbf{y}^\mathbf{T}\mathbf{A}\mathbf{x} + \mathbf{y}^\mathbf{T}\mathbf{y} \\ &= \mathbf{x}^\mathbf{T}\mathbf{A}^\mathbf{T}\mathbf{A}\mathbf{x} - \mathbf{2}\mathbf{y}^\mathbf{T}\mathbf{A}\mathbf{x} + \mathbf{y}^\mathbf{T}\mathbf{y} \\ &= \mathbf{x}^\mathbf{T}\mathbf{B}\mathbf{x} - \mathbf{2}\mathbf{c}^\mathbf{T}\mathbf{x} + \mathbf{y}^\mathbf{T}\mathbf{y} \end{aligned} \tag{3.9}$$



Figur 3.1: Minste kvadraters metode.

hvor $\mathbf{B} = \mathbf{A^T A}$ og $\mathbf{c} = \mathbf{A^T y}$. Minste kvadraters metode er et kvadratisk minimeringsproblem, og det kan vises at løsningen er

$$\mathbf{Bx} = \mathbf{c} \Leftrightarrow \mathbf{x} = \mathbf{B^{-1}c}$$

3.4.1 Eksempel

Gitt punktene (1,2), (2,1) og $(3, \frac{5}{2})$. Vi får $\mathbf{x}^T = [\, a \; b \,]$ og

$$\mathbf{y} = \begin{bmatrix} 2 \\ 1 \\ \frac{5}{2} \end{bmatrix} \quad \mathbf{A^T} = \begin{bmatrix} 1 & 2 & 3 \\ 1 & 1 & 1 \end{bmatrix} \quad \mathbf{A} = \begin{bmatrix} 1 & 1 \\ 2 & 1 \\ 3 & 1 \end{bmatrix} \quad \mathbf{A^T A} = \begin{bmatrix} 14 & 6 \\ 6 & 3 \end{bmatrix} \mathbf{A^T y} = \begin{bmatrix} \frac{23}{2} \\ \frac{11}{2} \end{bmatrix}$$

Løsning: For en 2x2 matrise $\mathbf{B} = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$ er $\mathbf{B^{-1}} = \frac{1}{ad-bc} \begin{bmatrix} d & -b \\ -c & a \end{bmatrix}$, slik at her er $\mathbf{B^{-1}} = \frac{1}{6} \begin{bmatrix} 3 & -6 \\ -6 & 14 \end{bmatrix}$ og vi får $\mathbf{x} = B^{-1}\mathbf{c} = \left[\frac{1}{4}\frac{4}{3}\right]$ $a = \frac{1}{4}$ og $b = \frac{4}{3}$.

3.4.2 Eksempel

Gitt punktene $(1, \frac{1}{2}), (\frac{3}{2}, \frac{3}{2}), (2, 1)$ og $(3, \frac{5}{2})$. Vi skal bruke minste kvadraters metode til å bestemme ligningen for den rette linjen som passer best mulig til punktene.

$$\mathbf{y} = \begin{bmatrix} \frac{1}{2} \\ \frac{3}{2} \\ 1 \\ \frac{5}{2} \end{bmatrix} \quad \mathbf{A^T} = \begin{bmatrix} 1 & \frac{3}{2} & 2 & 3 \\ 1 & 1 & 1 & 1 \end{bmatrix} \quad \mathbf{A} = \begin{bmatrix} 1 & 1 \\ \frac{3}{2} & 1 \\ 2 & 1 \\ 3 & 1 \end{bmatrix} \quad \mathbf{A^T A} = \begin{bmatrix} \frac{65}{4} & \frac{15}{2} \\ \frac{15}{2} & 4 \end{bmatrix} \mathbf{A^T y} = \begin{bmatrix} \frac{49}{4} \\ \frac{11}{2} \end{bmatrix}$$

Videre, $\mathbf{B^{-1}} = \frac{4}{35} \begin{bmatrix} 4 & -\frac{15}{2} \\ -\frac{15}{2} & \frac{65}{4} \end{bmatrix}$ og $\mathbf{x} = B^{-1}\mathbf{c} = \frac{4}{35} \begin{bmatrix} 49 - \frac{165}{4} \\ -\frac{735}{8} + \frac{715}{8} \end{bmatrix} = \begin{bmatrix} \frac{31}{35} \\ -\frac{2}{7} \end{bmatrix}$

Altså er $a = \frac{31}{35}$ og $b = -\frac{2}{7}$.

Vi kontrollerer at dette stemmer med punktene, og konkluderer med at vi trenger et verktøy slik at vi slipper å gjøre så mye brøkregning for hånd.

3.4.3

I utledningen av ligning 3.10 bruker regneregler for matriser og at $(AB)^T = B^T A^T$.

3.4.4 Minste kvadraters metode og andregradsfunksjon

. Ligningssystemet 3.7 blir på formen

$$\begin{aligned} y_1 &= ax_1^2 + bx_1 + c + e_1 \\ y_2 &= ax_2^2 + bx_2 + c + e_2 \\ &\vdots \\ y_m &= ax_m^2 + bx_m + c + e_m \end{aligned} \tag{3.10}$$

Vi ser at vi da får en ekstra kolonne i A matrisen og at $\mathbf{x}^T = [\ a\ b\ c\ ]$. Antall ligninger er det samme som antall målepunkter. $\mathbf{B} = \mathbf{A^T A}$ blir en 3×3 matrise.

3.4.5 Eksempel

Gitt m=5 punkter (-2.0, 4.1), (-1.0, 0.9), (0,0, 0.1), (1.0, 1.1) og (2.0, 3.9). Vi skal bruke minste kvadraters metode til å bestemme en andregradsfunksjon som passer best mulig til punktene. (Vi trenger jo bare tre punkter for å bestemme en andregradsfunksjon, så vi har to punkter ekstra).

$$A = \begin{bmatrix} 4.00 & -2.00 & 1.00 \\ 1.00 & -1.00 & 1.00 \\ 0.00 & 0.00 & 1.00 \\ 1.00 & 1.00 & 1.00 \\ 4.00 & 2.00 & 1.00 \end{bmatrix} \quad A^T = \begin{bmatrix} 4.00 & 1.00 & 0.00 & 1.00 & 4.00 \\ -2.00 & -1.00 & 0.00 & 1.00 & 2.00 \\ 1.00 & 1.00 & 1.00 & 1.00 & 1.00 \end{bmatrix}$$

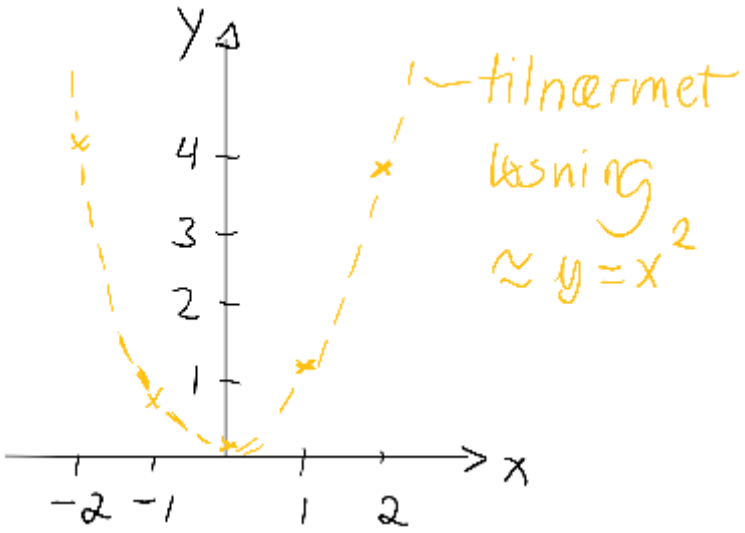
$$\mathbf{B} = \mathbf{A^T A} = \begin{bmatrix} 34 & 0 & 10 \\ 0 & 10 & 0 \\ 10 & 0 & 5 \end{bmatrix} \quad A^T = \begin{bmatrix} 4.00 & 1.00 & 0.00 & 1.00 & 4.00 \\ -2.00 & -1.00 & 0.00 & 1.00 & 2.00 \\ 1.00 & 1.00 & 1.00 & 1.00 & 1.00 \end{bmatrix}$$

Løsning:

$$B^{-1} = \begin{bmatrix} 0.0714 & 0 & -0.1429 \\ 0 & 0.1000 & 0 \\ -0.1429 & 0 & 0.4857 \end{bmatrix}$$

$$\mathbf{c} = \mathbf{A^T y} = \begin{bmatrix} 34.0 \\ -0.2 \\ 10.1 \end{bmatrix}$$

$$\mathbf{x} = \begin{bmatrix} 0.9858 \\ -0.0190 \\ 0.0484 \end{bmatrix}$$



Figur 3.2: Minste kvadraters metode og andregradsfunksjon.

3.4.6 Oppgave

Bruk av minste kvadraters metode i spill.

1. Bestem koordinater til 7-8 punkter i planet for en scene. Tegn gjerne en skisse. Et forslag er figur 4.3
2. Vi skal lage en parabelbane som passer best mulig til disse punktene.
3. Sett opp matrisene som trengs for å bruke minste kvadraters metode, og vis utregningen. Dokumenter i rapport.
4. Bruk et egnet verktøy til å regne ut løsningen (for eksempel Wolfram Alpha)
5. Beregn punkter (x,y) og lagre i en array eller på fil.
6. Dere kan bruke løsning på oblig 1 til å tegne kurven.
7. Punktene og grafen til funksjonen skal visualiseres.

Kapittel 4

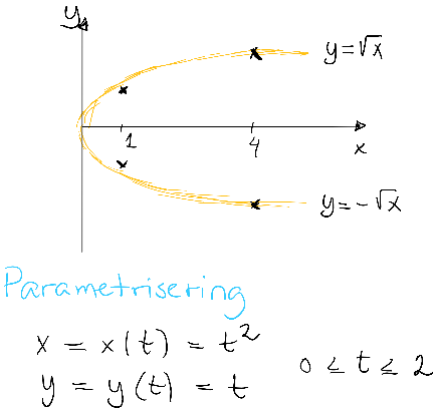
Interpolasjon

4.1 Introduksjon

Vi har inntil nå sett på flere eksempler hvor vi bruker en kjent funksjon eller kurve til å produsere kurver og flater i 2D og 3D.

- Vi kan
  - Analysere en kjent funksjon f(x), regne ut punkter på grafen til en kjent funksjon f(x) og tegne grafen.
  - Analysere en kjent funksjon f(x,y), regne ut punkter på flaten til f og tegne flaten.
  - Regne ut punkter på en kjent parametrisk kurve og tegne kurven.
  - (snart) konstruere en funksjon med minste kvadraters metode og tegne grafen.
- Minste kvadraters metode kan brukes til å bestemme en funksjon som beskriver et datasett best mulig, eller passer best mulig til dataene. Vi har sett eksempler på at en lineær funksjon og en andregradsfunksjon kan brukes.
- Det er ikke slik at alle datapunktene ligger på grafen til funksjonen. Vi sier at funksjonen approksimerer dataene.
- Når datapunktene ligger på grafen til funksjonen, eller på en parametrisk kurve, sier vi at funksjonen eller kurven interpolerer punktene.
- Polynomer er en klasse spesielt viktige funksjoner som er mye brukt til interpolasjon og approksimasjon.
- Vi får også bruk for Gausseliminasjon til løsing av 4x4 ligningssystemer.
- Vi trenger også paramatriske kurver.

Vi skal nå lære hvordan vi kan velge punkter og selv konstruere funksjoner, kurver og flater som passer til punktene. Vi har altså startet med dette, ved minste kvadraters metode.

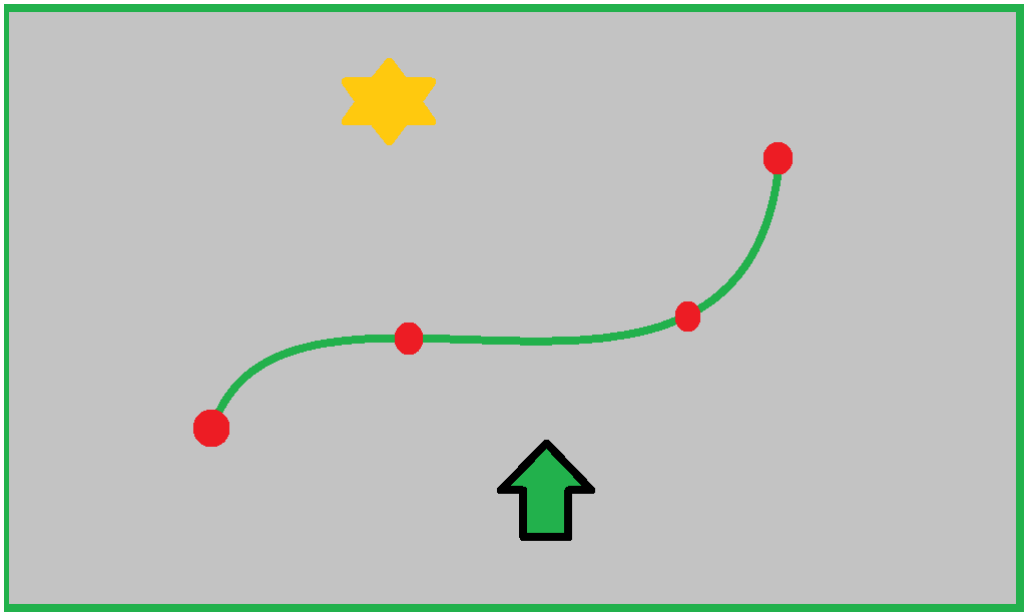


Figur 4.1: Vi trenger parametriske kurver!

4.1.1 Eksempel, parametrisk kurve

Når vi roterer grafen til funksjonen, er det plutselig nødvendig med to funksjoner $f(x) = \sqrt{x}$ og $g(x) = -\sqrt{x}$ for å beskrive kurven som grafen til en funksjon av variabel x. Vi har tilsvarende situasjon for en sirkel. Dette er to eksempler på at det kan være mer hensiktsmessig å benytte parametriske kurver. Ligningen for enhetssirkelen er $x^2 + y^2 = 1$, dette gir funksjonene $f(x) = \sqrt{1 - x^2}$ og $g(x) = -\sqrt{1 - x^2}$. En kjent parametrisering er $x(t) = \cos t,\ y(t) = \sin t,\ 0 \leq t\ < 2\pi$.

På samme måte kan vi parametrisere kurven på figur 4.1, $x(t) = t^2,\ y(t) = t$.



Figur 4.2: Bane til en npc som interpolerer fire punkter kan representeres med 3. gradsfunksjon.

4.2 Interpolasjon

Polynomer er en klasse funksjoner som er spesielt mye brukt til konstruksjon av kurver og flater, interpolasjon og approksimasjon. Vi skal først starte med noen eksempler på interpolasjon av punkter med polynomer. Senere skal vi løse problemer med parametriske kurver, som på figur 4.1.

4.2.1 Eksempler på interpolasjon av to, tre og fire punkter

Vi skal bestemme en funksjon som interpolerer henholdsvis to, tre og fire punkter, for eksempel punktene nedenfor. Funksjonens graf skal gå gjennom punktene. Hvis vi har to punkter, kan vi finne et førstegradspolynom (lineært) som interpolerer punktene. Hvis vi har tre punkter, kan vi finne et andregradspolynom (kvadratisk) som går gjennom punktene. Og hvis vi har fire punkter, kan vi bestemme et tredjegradspolynom (kubisk) som går gjennom punktene. Noen eksempler:

1. (1,1) og (3,2)
2. (1,1), (3,3) og (5,1) - se avsnitt 4.2.2.
3. (0,0), (1,1), (2,0) og (3,1) - se avsnitt 4.3.1.

Eksempel - interpolere to punkter

Velg to punkter og konstruer et lineært polynom som interpolerer punktene. Her bruker vi punktene ovenfor.

$$\begin{aligned} f(x) &= ax + b \\ 1 &= a \cdot 1 + b \\ 3 &= a \cdot 2 + b \end{aligned}$$

Vi kan sette opp dette på matriseform Ax=b (her er b en kolonnevektor), med

$$A = \begin{bmatrix} 1 & 1 \\ 3 & 1 \end{bmatrix}$$

$x^T = [a\ b\ ]$ og $b^T = [1\ 3]$. Løsningen blir $x = A^{-1}b = [\frac{1}{2}\ \frac{1}{2}]$, altså $f(x) = \frac{1}{2}x + \frac{1}{2}$.

4.2.2 Interpolasjon med 2. gradspolynom

Et andregradspolynom kan som kjent skrives på formen

$$p(x) = ax^2 + bx + c$$

. Vi husker at vi kan sette $p(x) = 0$ og finne nullpunktene/røttene av formelen

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

Men nå er problemstillingen snudd. Vi har ikke lenger et kjent funksjonsuttrykk å jobbe med. Derimot har vi punkter (her: tre) og vi skal konstruere et polynom.

Vi vet jo fra før at det er mulig å konstruere et førstegradspolynom (rett linje) som går gjennom to punkter. Tilsvarende kan vi konstruere et andregradspolynom som går gjennom tre punkter, og et tredjegradspolynom som går gjennom fire punkter. Og slik kan vi fortsette. Imidlertid kommer vi sjelden til å arbeide med polynomer av grad høyere enn 3. Hvis vi trenger å interpolere flere punkter enn fire, skal vi heller skjøte sammen flere polynomer av grad 3 eller lavere.

Punktene uten å angi koordinater er (x_0, y_0), (x_1, y_1) og (x_2, y_2), og som alltid er det et valg man må gjøre om nummereringen skal starte på 0 eller 1.

Så hva slags problem får vi å løse? Bruker vi den kjente notasjonen vår for et andregradspolynom, får vi

$$\begin{aligned} ax_0^2 + bx_0 + c &= y_0 \\ ax_1^2 + bx_1 + c &= y_1 \\ ax_2^2 + bx_2 + c &= y_2 \end{aligned}$$

På matriseform blir dette $\mathbf{Ax} = \mathbf{b}$ som har løsningen $\mathbf{x} = A^{-1}\mathbf{b}$.

Eksempel - interpolere tre punkter

Velg tre punkter som ikke ligger på en rett linje, (1,1), (3,3) og (5,1). Konstruer et andregradspolynom som interpolerer punktene.

$$\begin{aligned} f(x) &= ax^2 + bx + c \\ 1 &= a \cdot 1^2 + b \cdot 1 + c \\ 3 &= a \cdot 3^2 + b \cdot 3 + c \\ 1 &= a \cdot 5^2 + b \cdot 5 + c \end{aligned}$$

Vi kan sette opp dette på matriseform Ax=b, med

$$A = \begin{bmatrix} 1 & 1 & 1 \\ 9 & 3 & 1 \\ 25 & 5 & 1 \end{bmatrix}$$

$x^T = [a\ b\ c]$ og $b^T = [1\ 3\ 1]$. Løsningen blir $x = A^{-1}b = [-\frac{1}{2}\ 3\ -\frac{3}{2}]$, altså $f(x) = -\frac{1}{2}x^2 + 3x - -\frac{3}{2}$.

Eksempel (uten matriseformen)

Gitt punktene $(0, \frac{3}{2}), (1, 0) og (2, -\frac{1}{2})$. Bestem andregradspolynomet som interpolerer punktene.

Vi får

$$\begin{aligned} a \cdot 0 + b \cdot 0 + c &= \frac{3}{2} \\ a \cdot 1 + b \cdot 1 + c &= 0 \\ a \cdot 4 + b \cdot 2 + c &= -\frac{1}{2} \end{aligned}$$

Ligning 1 gir $c = \frac{3}{2}$. Innsatt i ligning 2 får vi $a + b = -\frac{3}{2} \Leftrightarrow b = -\frac{3}{2} - a$. Setter vi dette inn i ligning 3, får vi $4a + 2(-\frac{3}{2} - a) + \frac{3}{2} = -\frac{1}{2} \Leftrightarrow 2a - \frac{3}{2} = -\frac{1}{2} \Leftrightarrow a = \frac{1}{2}$. Dette gir $b = -2$ og $p(x) = \frac{1}{2}x^2 - 2x + \frac{3}{2}$.

4.2.3 Oppgave

1. La $(x_0, y_0) = (\frac{1}{2}, \frac{5}{4})$, $(x_1, y_1) = (2, -1)$ og $(x_2, y_2) = (4, 3)$. Du skal finne et andregradspolynom som interpolerer disse punktene. Sett opp ligningene på matriseform.

2. Bestem interpolasjonspolynomet (andregradsfunksjonen).

3. Tegn/skisser funksjonen.

4.3 Interpolasjon med 3. gradspolynom

Når vi skal bestemme et polynom som interpolerer 4 punkter, trenger vi et polynom av grad ≤ 3. Vi skal se to måter å bruke tredjegradspolynom til interpolasjon.

4.3.1 Interpolasjon av 4 punkter

Gitt fire punkter (x_0, y_0), $(x_1, y_1), (x_2, y_2)$ og (x_3, y_3). Vi skal bestemme et kubisk polynom (3. grads) $p_3(x)$ som interpolerer punktene. Det vil si at $p_3(x_i) = y_i,\ i = 0, 1, 2, 3$. Vi kan skrive det kubiske polynomet på formen $p_3(x) = ax^3 + bx^2 + cx + d$ hvor $a, b, c, d \in \mathbf{R}$, altså reelle konstanter som i det kvadratiske tilfellet.

Vi skriver opp ligningene vi får av interpolasjonsbetingelsene:

$$\begin{aligned} ax_0^3 + bx_0^2 + cx_0 + d &= y_0 \\ ax_1^3 + bx_1^2 + cx_1 + d &= y_1 \\ ax_2^3 + bx_2^2 + cx_2 + d &= y_2 \\ ax_3^3 + bx_3^2 + cx_3 + d &= y_3 \end{aligned}$$

Vi velger nå punktene (0,0), (1,1), (2,0) og (3,1) og skal konstruere et tredjegradspolynom som interpolerer punktene.

$$\begin{aligned} f(x) &= ax^3 + bx^2 + cx + d \\ 0 &= a \cdot 0^3 + b \cdot 0^2 + c \cdot 0 + d \\ 1 &= a \cdot 1^3 + b \cdot 1^2 + c \cdot 1 + d \\ 0 &= a \cdot 2^3 + b \cdot 2^2 + c \cdot 2 + d \\ 1 &= a \cdot 3^3 + b \cdot 3^2 + c \cdot 3 + d \end{aligned}$$

Vi kan sette opp dette på matriseform Ax=b, med

$$A = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 8 & 4 & 2 & 1 \\ 27 & 9 & 3 & 1 \end{bmatrix}$$

$x^T = [a\ b\ c\ d]$ og $b^T = [0\ 1\ 0\ 1]$. Løsningen blir $x = A^{-1}b = [\frac{2}{3}\ \ -3\ \frac{10}{3}\ 0]$, altså $f(x) = \frac{2}{3}x^3 - 3x^2 + \frac{10}{3}x$.

4.4 Notasjon og matematisk formulering*

Det generelle interpolasjonsproblemet kan formuleres som følger:

Gitt $n+1$ distinkte punkter $x_0, x_1, \ldots x_n$ med verdier $y_0, y_1, \ldots y_n$. Da eksisterer et entydig polynom $p(x) \in \mathcal{P}_n$ (betyr et entydig polynom av grad n) slik at $p(x_i) = y_i,\ i = 0, \ldots n$.
Et polynom $p(x) \in \mathcal{P}_n$ er et polynom av grad $\leq n$ og kan skrives

$$p(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3 + \ldots + a_n x^n = \sum_{i=0}^{n} a_i\ x^i \tag{4.1}$$

hvor a_i-ene er konstanter (reelle tall). Denne skrivemåten kalles potensformen eller monomialformen til p og funksjonene $p_i(x) = x^i$ kalles monomialer.

Det fins andre måter å skrive opp et polynom på som er mer effektive med tanke på evaluering, og vi kommer til det.

Hvis vi skriver opp interpolasjonsbetingelsene $p(x_i) = y_i$ i likning (4.1), får vi n+1 ligninger med n+1 ukjente:

$$\begin{aligned} a_0 + a_1x_0 + a_2x_0^2 + a_3x_0^3 + \ldots + a_nx_0^n &= y_0 \\ a_0 + a_1x_1 + a_2x_1^2 + a_3x_1^3 + \ldots + a_nx_1^n &= y_1 \\ a_0 + a_1x_2 + a_2x_2^2 + a_3x_2^3 + \ldots + a_nx_2^n &= y_2 \\ &\vdots \\ a_0 + a_1x_n + a_2x_n^2 + a_3x_n^3 + \ldots + a_nx_n^n &= y_n \end{aligned}$$

På matriseform $\mathbf{Ax}$=$\mathbf{b}$ blir dette

$$\begin{bmatrix} 1 & x_0 & x_0^2 & x_0^3 & \ldots & x_0^n \\ 1 & x_1 & x_1^2 & x_1^3 & \ldots & x_1^n \\ 1 & x_2 & x_2^2 & x_2^3 & \ldots & x_2^n \\ \vdots & & \ldots & & & \vdots \\ 1 & x_n & x_n^2 & x_n^3 & \ldots & x_n^n \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ \vdots \\ a_n \end{bmatrix} = \begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

og på mer kompakt form blir dette

$$\sum_{i=0}^{n} a_i \, x_j^i \; = y_i, \; j = 0, \ldots, n \tag{4.2}$$

Vi ser at vi bruker indeksen i til å multiplisere respektive rader i A med koeffisient-vektoren I [7] introduseres Lagrange funksjonene

$$L_i(t_j) = \delta_{ij} = \begin{cases} 1, & i = j \\ 0, & i \neq j, \quad j = 0, \ldots, n. \end{cases} \tag{4.3}$$

som er unike i $\mathcal{P}_n$

4.5 Interpolasjon av to funksjonsverdier og to deriverte

Du kan gjøre oppgave 4.6.1 før du går videre med denne seksjonen.

Gitt punktene $p_0(x_0, y_0)(-2, -\frac{15}{2})$ og $p_3(3, 0)$. I stedet for å bruke interpolasjonsbetingelsene i ytterligere to punkter p_2 og p_3 for å sette opp et ligningssystem med fire ligninger og fire ukjente, kan vi stille betingelser til de deriverte i p_0 og p_3. (p_0 og p_3 brukes for å være konsistent med indekseringen i oppgave 4.6.1).

La oss kreve at den deriverte til interpolasjonspolynomet p i $x_0 = -2$ og $x_3 = 3$ skal være
$p'(-2) = \frac{23}{2}$ og $p'(3) = 4$.
Vi har da følgende fire interpolasjonsbetingelser:

1. $p(-2) = -\frac{15}{2}$

2. $p(3) = 0$
3. $p'(-2) = \frac{23}{2}$
4. $p'(3) = 4$

Polynomet vi skal bestemme er på formen $p(x) = ax^3 + bx^2 + cx + d$, og når vi deriverer får vi $p'(x) = 3ax^2 + 2bx + c$. Vi setter inn og regner ut og får

$$\begin{aligned} -8a + 4b - 2c + d &= -\frac{15}{2} \\ 27a + 9b + 3c + d &= 0 \\ 12a - 4b + c &= \frac{23}{2} \\ 27a + 6b + c &= 4 \end{aligned}$$

Vi kan nå sette opp dette på matriseform $\mathbf{Ax}=\mathbf{b}$ og løse ligningssystemet med matriseklassens funksjoner. Løsningen blir, skrevet på potensform
$p(x) = \frac{1}{2}x^3 - \frac{3}{2}x^2 - \frac{1}{2}x + \frac{3}{2}$.

Denne formen for interpolasjon kalles kubisk **Hermite** interpolasjon. Vi kan skjøte sammen flere kubiske polynomer for å interpolere mer enn fire punkter.

4.6 Oppgaver

I oppgavene nedenfor er det meningen at du skal tegne eller skissere polynomene du regner ut i tillegg til å sette opp og løse ligningssystemene. Du kan gjerne bruke GeoGebra eller et tilsvarende program. Senere skal vi programmere tegning av slike kurver sjøl.

4.6.1

1. Sett opp matrisen $\mathbf{A}$ og vektorene $\mathbf{x}$ og $\mathbf{b}$ for det generelle tilfellet.
2. La $(x_0, y_0) = (-2, -\frac{15}{2})$, $(x_1, y_1) = (\frac{1}{2}, \frac{15}{16})$, $(x_2, y_2) = (2, -\frac{3}{2})$ og $(x_3, y_3) = (3, 0)$. Du skal finne et tredjegradspolynom som interpolerer disse punktene. Sett opp ligningene på matriseform.
3. Bestem interpolasjonspolynomet ved å bruke matrisefunksjoner fra oblig1.
4. Tegn/skisser funksjonen.

I oppgavene nedenfor er problemstillingen ikke like detaljert, men framgangsmåten blir den samme.

4.6.2

Bestem et polynom av grad ≤ 3 som interpolerer punktene
$(0, 1)$, $(\frac{1}{2}, \frac{9}{16})$, $(\frac{3}{2}, -\frac{5}{16})$ og $(3, 4)$

4.6.3

1. Bestem et polynom av grad ≤ 2 som interpolerer punktene $(-1,0),\ (0,1),\ (1,0)$
2. Bestem et polynom av grad ≤ 3 som interpolerer punktene
   $(-1,0),\ (-\frac{1}{2},\ \frac{\sqrt{3}}{2}),(\frac{1}{2},\ \frac{\sqrt{3}}{2}),\ (1,0)$
3. Tegn begge polynomene sammen med øvre halvdel av enhetssirkelen.

4.6.4

1. Bestem et polynom av grad ≤ 2 som interpolerer punktene $(0,0),\ (\frac{\pi}{2},1),\ (\pi,0)$
2. Bestem et polynom av grad ≤ 2 som interpolerer punktene $(\pi,0),\ (\frac{3\pi}{2},-1),\ (2\pi,0)$
3. Tegn begge polynomene sammen med funksjonen $\sin x, 0 \leq x \leq 2\pi$

4.6.5

1. Bestem et polynom av grad ≤ 3 som interpolerer punktene $p_0(1,0)$ og $p_1(\pi,0)$ og hvor de deriverte er henholdsvis 1 og -1 (vi skriver $p'(x_0) = 1$ og $p'(x_1) = -1$
2. Tegn polynomet på samme figur som i forrige oppgave.

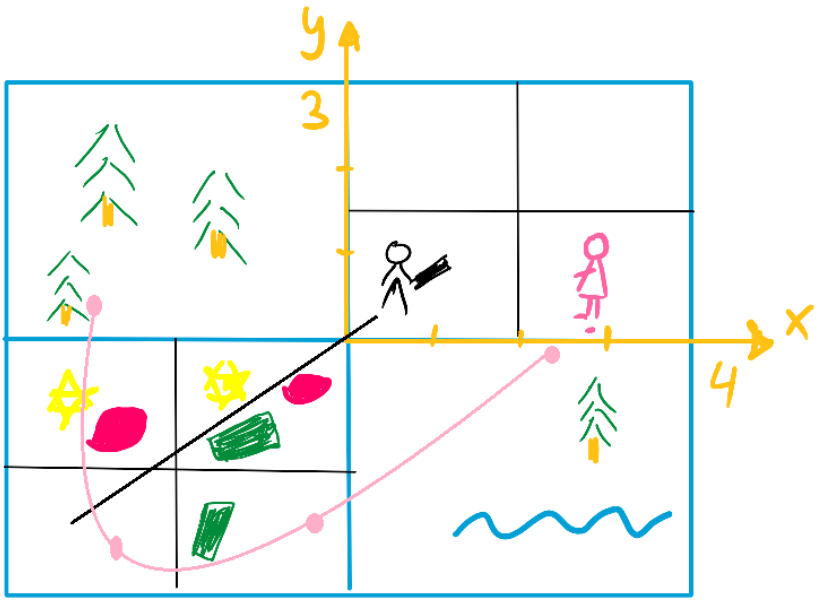
4.6.6

1. Bestem et polynom av grad ≤ 3 som interpolerer punktene
   $(-\frac{1}{2},-\frac{\sqrt{3}}{3}),\ (-\frac{\sqrt{3}}{2},-\sqrt{3}),\ (\frac{1}{2},\frac{\sqrt{3}}{3}),\ (\frac{\sqrt{3}}{2},\sqrt{3})$
   Forklar resultatet.
2. Bestem et polynom av grad ≤ 3 som interpolerer punktene
   $(-\frac{1}{2},-\frac{\sqrt{3}}{3}),\ (-\frac{\sqrt{2}}{2},-1),(\frac{1}{2},\frac{\sqrt{3}}{3}),\ (\frac{\sqrt{3}}{2},\sqrt{3})$
   Tegn opp polynomet sammen med $f(x) = \tan x,\ -\frac{\pi}{2} \leq x \leq \frac{\pi}{2}$.

4.6.7

Bruk av interpolasjon i spill. Et forslag til scene er figur 4.3

1. Bestem koordinater til 4 punkter for interpolasjon med kubisk polynom. Sett opp matriser og vis utregningen i rapport.
2. Beregn punkter (x,y) og lagre i en array eller på fil.
3. Dere kan bruke løsning på oblig 1 til å tegne kurven.
4. Punktene og grafen til funksjonen skal visualiseres.



Figur 4.3: Eksempel på scene. Kurven er ikke en funksjon.

4.7 Kubisk Hermite interpolasjon

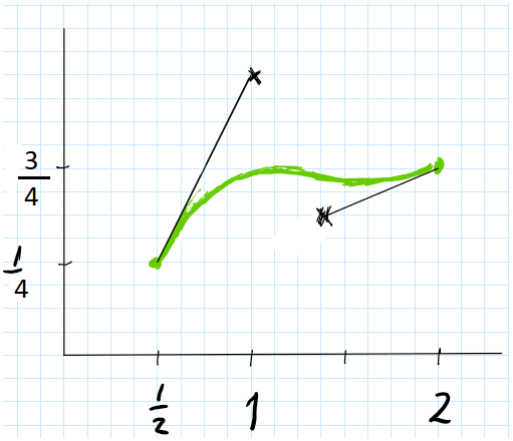
Splines er stykkevise polynomer. Vi kan for eksempel ha stykkevis lineære polynomer, kvadratiske, kubiske. Vi kan ha polynomer i flere variable. En Hermite spline er et stykkevis interpolerende polynom hvor både en verdi og en derivert er gitt i kontrollpunktene/skjøtene. Hver polynombit avhenger kun av sine endepunktsbetingelser og kan justeres lokalt uten at hele det sammensatte polynomet må rekalkuleres.

Kubisk spline interpolasjon ligner på kubisk Hermite interpolasjon, men det er ikke noe krav om å interpolere den deriverte i skjøtene. I stedet krever man at venstre og høyre polynombit har samme derivert i skjøtene.

Vi har sett at vi kan velge å interpolere fire punkter med et kubisk polynom, og at vi i stedet kan interpolere endepunktene og de deriverte i endepunktene. I begge tilfeller får vi et 4x4 ligningssystem som bestemmer konstantene til det kubiske polynomet.

Vi arbeider nå med funksjoner av en variabel. Men vi har kjennskap til parametriseringer og affine rom. Metodene som vi her går gjennom for funksjoner av en variabel vil i mange tilfeller kunne generaliseres til parametriske kurver.

Kubisk Hermite interpolasjon er en metode som er viktig å forstå når vi kommer til Bezier kurver og splines. Vi starter med litt repetisjon, og skal gå videre til å interpolere sammensatte polynomer.



Figur 4.4: Hermite interpolasjon, eksempel.

4.7.1 Eksempel

Vi kan sette opp en tredjegradsfunksjon på potensform som tidligere, med uttrykket for den deriverte:

$$\begin{aligned} p(x) &= ax^3 + bx^2 + cx + d \\ p'(x) &= 3ax^2 + 2bx + c \end{aligned} \tag{4.4}$$

La oss se på et eksempel hvor vi interpolerer en funksjon i punktene (x_0, y_0) og (x_1, y_1) hvor $x_0 < x_1$. Vi tegner figur og setter inn noen tall:

- Endepunkter til intervallet: $x_0 = \frac{1}{2},\ y_0 = \frac{1}{4},\ x_1 = 2,\ y_1 = \frac{3}{4}$
- De deriverte i endepunktene: $y_0' = 2$ og $y_1' = \frac{1}{2}$

Interpolasjonsbetingelser

Ved kubisk Hermite interpolasjon på intervallet $[x_0,\ x_1]$ krever vi

$$\begin{aligned} p(x_0) = y_0,\ p(x_1) = y_1 \\ p'(x_0) = y_0',\ p'(x_1) = y_1' \end{aligned} \tag{4.5}$$

$p(x_0) = y_0 \Rightarrow ax_0^3 + bx_0^2 + cx_0 + d = y_0$
$p(x_1) = y_1 \Rightarrow ax_1^3 + bx_1^2 + cx_1 + d = y_1$

Når vi setter inn tall, får vi

$$\begin{aligned} \frac{1}{8}a + \frac{1}{4}b + \frac{1}{2}c + d &= \frac{1}{4} \\ 8a + 4b + 2c + d &= \frac{3}{4} \end{aligned}$$

Vi har så langt to ligninger med fire ukjente, og vi trenger to ligninger til for å bestemme konstantene a, b, c og d. De siste to ligningene får vi av de deriverte

i endepunktene. Vi får av ligning 4.4 $p'(x) = 3ax^2 + 2bx + c$ og videre

$p'(x_0) = y'_0 \Rightarrow p'(\frac{1}{2}) = 2 \Rightarrow \frac{3}{4}a + b + c = 2$
$p'(x_1) = y_1 \Rightarrow p'(2) = \frac{1}{2} \Rightarrow 12a + 4b + c = \frac{1}{2}$

På matriseform blir dette **Ax**=**b** med

$$A = \begin{bmatrix} \frac{1}{8} & \frac{1}{4} & \frac{1}{2} & 1 \\ 8 & 4 & 2 & 1 \\ \frac{3}{4} & 1 & 1 & 0 \\ 12 & 4 & 1 & 0 \end{bmatrix}$$

med $\mathbf{x}^T = [a\ b\ c\ d]$ og $\mathbf{b}^T = [\frac{1}{4}\ \frac{3}{4}\ 2\ \frac{1}{2}\ ]$

4.7.2 Hermite interpolasjon setning 1

Gitt reelle tall $x_0 < x_1, y_0, y_1, y'_0, y'_1$ og la $h = x_1 - x_0$. Polynomet p gitt ved

$$p(x) = y_0 + y'_0(x - x_0) + q(x - x_0)^2 + c(x - x_0)^3 \tag{4.6}$$

hvor

$$q = \frac{3\frac{y_1 - y_0}{h} - 2y'_0 - y'_1}{h} \tag{4.7}$$

$$c = \frac{-2\frac{y_1 - y_0}{h} + y'_0 + y'_1}{h^2} \tag{4.8}$$

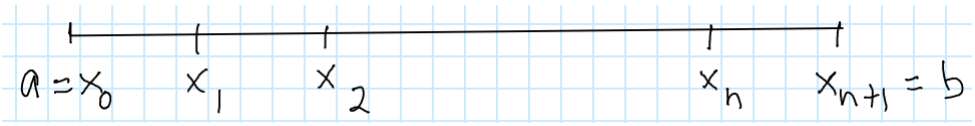
er det entydige kubisk polynomet slik at $p(x_0) = y_0,\ p(x_1) = y_1,$
$p'(x_0) = y'_0$ og $p'(x_1) = y'_1$

Eksempel

Vi skal finne kubisk Hermite interpolant til x^4 på [0,1] ved å bruke resultatet ovenfor. Vi får
$x_0 = 0,\ x_1 = 1,\ h = 1,\ y_0 = 0,\ y_1 = 1,\ y'_0 = 0,\ y'_1 = 4$
Videre q=-1 og c=2 (sett inn i formlene og regn ut dette) og til slutt $p(x) = -x^2 + 2x^3$

Kontroll: $p(0) = 0,\ p(1) = 1,\ p'(x) = -2x + 6x^2 \Rightarrow p'(0) = 0,\ p'(1) = 4.$ Alle de fire interpolasjonsbetingelsene stemmer, og da må resultatet stemme. Tegn figur.

Vi kommer ikke til å bevise denne setningen her, selv om det ikke er veldig vanskelig. Vi kommer heller ikke nærmere inn på hvordan disse formlene framkommer. Setningen er hentet fra [8] og er tatt med for å kunne regne ut Hermite interpolasjonspolynomer på en litt enklere måte enn ved å løse 4x4 ligningssystem. Metoden gir litt raskere beregninger (bedre matriser) enn ved å bruke potensbasisen direkte.



Figur 4.5: Stykkevis Hermite interpolasjon, intervaller. Punktene hvor lokale polynomer møtes kalles gjerne kontrollpunkter eller skjøter.

4.7.3 Stykkevis Hermite interpolasjon

Før vi går videre, repeterer vi ideen med å bruke kubiske polynomer til interpolasjon. Hvis vi skal interpolere $n+1$ punkter, kan vi gjøre det med et polynom av grad $\leq n$. Når n blir stor, leder dette til store ligningssystemer som er tunge å løse. Hvilke flere ulemper husker vi med polynomer av høy grad?

Vi skal nå bruke kubisk Hermite interpolasjon med flere polynombiter/intervaller, se 4.5. Vi har n+2 punkter $a = x_0 < x_1 < \ldots < x_n < x_{n+1} = b$ og skal konstruere en approksimasjon F til en gitt funksjon f ved å sette sammen biter av kubiske Hermite interpolanter.

Et nytt begrep her er approksimasjon. Vi kommer tilbake til det senere.

Anta at $s_i = f'(x_i),\ i = \ \ldots\ , n+1$ eksisterer. På hvert intervall $[x_i,\ x_{i+1}]$ har vi et kubisk Hermite polynom:

$$p_i(x_j) = f(x_j), \quad p_i'(x_j) = s_j, \quad j = i,\ i+1 \tag{4.9}$$

Den sammensatte funksjonen

$$F(x) = \begin{cases} p_0(x) & x_0 \leq x < x_1 \\ p_1(x) & x_1 \leq x < x_2 \\ \vdots & \\ p_{n-1}(x) & x_{n-1} \leq x < x_n \\ p_n(x) & x_n \leq x < x_{n-1} \end{cases} \tag{4.10}$$

er konstruert slik at

$$\begin{aligned} p_{i-1}(x_i) &= f(x_i) = p_i(x) \\ p_{i-1}'(x_i) &= f(x_i) = p_i'(x_i) \quad x_{n-1} \leq x < x_n \end{aligned} \tag{4.11}$$

Siden F er et polynom på hvert intervall og deriverbar i skjøtene, er F deriverbar på [a, b]. Vi kan nå bruke setning 4.7.2 på hvert intervall, ved å re-indeksere: $x_0 := x_i,\ x_1 := x_{i+1},\ y_0 := f(x_i),\ y_1 := f(x_{i+1}),\ y_0' := s_i,\ y_1' := s_{i+1}$. Dette gir for hvert polynom p_i

$$p_i(x) = f(x_i) + s_i(x - x_i) + q_i(x - x_i)^2 + c_i(x - x_i)^3 \tag{4.12}$$

Her er

$$\begin{aligned} q_i &= \ \ \frac{3d_i - 2s_i - s_{i+1}}{h_i} \\ c_i &= \ \ \frac{-2d_i + s_i + s_i + 1}{h_i^2} \\ d_i &= \ \ \frac{f(x_{i+1}) - f(x_i)}{x_{i+1} - x_i} \\ h_i &= \ \ x_{i+1} - x_i \end{aligned} \tag{4.13}$$

4.7.4 Eksempel

La oss se på et eksempel på stykkevis kubisk Hermite interpolasjon med to intervaller. Vi skal finne den stykkevise kubiske Hermite interpolanten til $f(x) = x^4$. Vi velger a=0 og b=2. Med to intervaller blir $n = 1$ og
$\{x_0, x_1, x_2\} = \{0,\ 1,\ 2\}$.

Vi finner først p_1 ved å sette i=1 i ligningene 4.12 og 4.13. Dette gir
$f(x_1) = 1,\ f(x_2) = 16,\ s_1 = 4,\ s_2 = 32,\ h_1 = 1,\ d_1 = 15,\ q = 5,\ c = 6.$

Innsatt i 4.13 får vi
$p_1(x) = 1 + 4(x-1) + 5(x-1)^2 + 6(x-1)^3,\ 1 \leq x \leq 2.$

Oppgave

Vis på samme måte at $p_0(x) = 2x^3 - x^2,\ 0 \leq x \leq 1$ (samme resultat som i eksempel 4.7.2).

Resultat

$$F = \begin{cases} p_0(x) = 2x^3 - x^2, & 0 \leq x \leq 1 \\ p_1(x) = 1 + 4(x-1) + 5(x-1)^2 + 6(x-1)^3, & 1 \leq x \leq 2 \end{cases}$$

4.8 Dividerte differenser

Når vi ikke har noen underliggende funksjon å interpolere, så har vi heller ikke noe eksplisitt uttrykk for de deriverte (y_i'). I det generelle tilfellet så har vi jo et sett med punkter som vi skal interpolere eller approksimere, og ingen kjent funksjon.

Hva gjør man da?

Man estimerer den deriverte. Koeffisienten d i ligning 4.13 er et eksempel på en dividert differens. Dividerte differenser har en egen notasjon:

$$f[x_i, x_{i+1}] = \frac{f(x_{i+1}) - f(x_i)}{x_{i+1} - x_i}$$

Dette er absolutt ingen dårlig tilnærming. Hvis vi husker tilbake på den deriverte fra Matematikk 1 kurset, får vi den deriverte som en grenseverdi når $x_{i+1} \to x_i$. På figur 4.4 er endepunktene markert med grønne prikker, og de deriverte med tynne linjer. Vi ser at dersom vi hadde hatt et ekstra interpolasjonspunkt der de tynne strekene slutter, så kunne vi ha regnet ut den deriverte eksakt ved en subtraksjon.

4.9 Oppgaver

4.9.1

Finn stykkevis kubisk Hermite interpolant med a=0, b=2 og n=1 for
1) $f(x) = \sin \pi x$
2) $g(x) = \sqrt{2x - x^2}$

4.9.2

Gitt $f(x) = x\cos(\pi x), n = 1, x_k = \frac{k}{2}, y_k = f(x_k), k = 0, 1, 2$
La $s_k = f'(x_k), k = 0, 1, 2$. Bestem en kubisk Hermite interpolant til f på $[a, b]$

4.9.3

Kubiske Hermite interpolasjon av x^4 på intervallet [0, 1]. Bruk ligningene 4.4 og 4.5 og sett opp først de to ligningene for interpolasjon i endepunktene, deretter de to ligningene for interpolasjon av de deriverte i endepunktene.

Sett så opp ligningene på matriseform $\mathbf{Ax}$=$\mathbf{b}$ hvor $\mathbf{x}^T = [a\ b\ c\ d]$ og $\mathbf{b}^T = [y_0\ y_1\ y_0'\ y_1']$. Vis at

$$A = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 3 & 2 & 1 & 0 \end{bmatrix}$$

og $\mathbf{b}^T = [0\ 1\ 0\ 4]$. Bestem til slutt $\mathbf{A}^{-1}$ (kalles Hermite matrisen) og $\mathbf{x}$ og sett opp interpolasjonspolynomet.

4.9.4

Vi har nå to metoder for å gjøre kubisk Hermite interpolasjon. La oss for anledningen kalle den uavhengige variabelen t i stedet for x. Bestem en kubisk Hermite interpolant for følgende fire sett av interpolasjonsbetingelser:

t_0	y_0	y_0'	t_1	y_1	y_1'
0	1	-3	1	0	0
0	0	3	1	0	0
0	0	0	1	0	-3
0	0	0	1	1	3

4.9.5 Horner's regel*

Oppgave 2.14 fra [6] beskriver en algoritme kalt Horner's regel for evaluering av polynomet $f(x) = \Sigma_{i=0}^{n} a_i x^i$:

```
double f = 0;
for (int i=n; i>=0; --i)
   f = x * f + a[i];
```

a) Vis hvordan stegene utføres med denne algoritmen for $x = 3, f(x) = 4x^4 + 8x^2 + x + 2$.

b) Forklar hvorfor algoritmen fungerer.

c) Hva blir kjøretiden?

4.10 Bezier kurver

Vi skal nå bruke interpolasjonskunnskapene våre på kurver, ikke bare funksjoner. Vi har allerede lært litt om affine rom og barysentriske koordinater. Vi tar en kort repetisjon. Se figur 5.1.

4.10.1 Parameterframstilling av en rett linje

Gitt to punkter A og B, kan vi skrive en parameterframstilling av linja gjennom A og B som $x(t) = (1-t)\ A + tB$. Alternativt med vektoren $\mathbf{v} = B - A$ får vi $x(t) = A + t\mathbf{v} = A + t(B-a)$. 1-t og t kalles de barysentriske koordinatene til A og B, og når $0 \leq t \leq 1$ får vi et punkt på linjestykket mellom A og B.

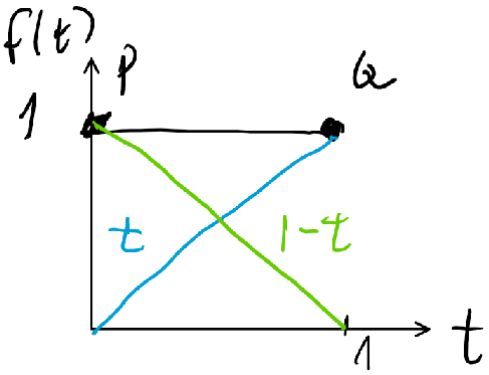
Vi kan finne igjen et hvilket som helst punkt på linja slik: Vi starter i punktet A. Deretter går vi et tall, en konstant c multiplisert med retningsvektoren. Vi får en affin sum som er slik: $A + c\mathbf{v}$. Alternativt kan du tenke deg vektoren $\mathbf{a} = \vec{OA}$ og starte i origo.

- $c = 0$ gir punktet A.
- $c = \frac{1}{2}$ gir punktet midt mellom A og B.
- $c = 1$ gir punktet B.
- $c = 2$ gir et punkt like langt unna B som A er, på motsatt side.

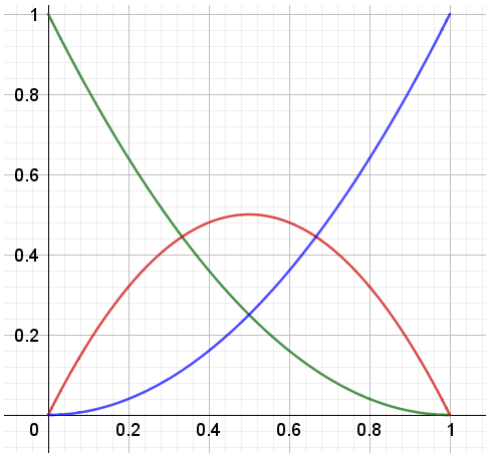
Dette hittil anonyme punktet på linja som vi har omtalt, har koordinater (x, y). Når vi har med en parametrisering å gjøre, er ikke y en funksjon av x, men både x og y er parametriske med t som parameter. Vi skriver gjerne

$$x = x(t) \text{ og } y = y(t)$$

Vi kaller gjerne punktet $\mathbf{x}(t) = (x(t), y(t))$, og i tre dimensjoner har vi i tillegg z(t). Da skrives $\mathbf{x}(t) = (x(t), y(t), z(t))$ La oss nå bytte ut navnet på x-aksen til t-aksen og tegne opp de rette linjene $y = t$ og $y = 1 - t$ for $0 \leq t \leq 1$. I tillegg plasserer vi punktene P og Q fra figur 5.1 som vist på figur 4.6. Vi ser at



Figur 4.6: Lineære basisfunksjoner.



Figur 4.7: Kvadratiske Bezierkurver kan lages ved å kombinere disse Bernstein-basisfunksjonene av grad d=2.

- t og $1-t$ summerer seg til 1 (opplagt).
- t er 1 der hvor $1-t$ er 0 og omvendt (også opplagt).
- t og $1-t$ er lineært uavhengige.
- t og $1-t$ er lineære basisfunksjoner.

Det er klart at $1-t+t=1$, vi har en affin kombinasjon av punktene. Hvis vi kvadrerer hver side av dette uttrykket, og betrakter det som en ligning, må venstresiden fortsatt være 1. Vi får

$$((1-t)+t)^2 = (1-t)^2 + 2t(1-t) + t^2$$

De tre uttrykkene på høyre side er alle kvadratiske og summerer seg opp til 1 uavhangig av hva t er. De kalles Bernstein basisfunksjoner av grad 2.

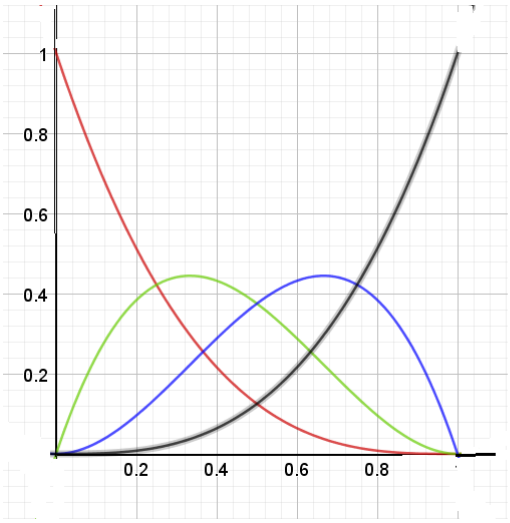
4.10.2 Basisfunksjoner

Hva hvis vi fortsetter?

$$\begin{aligned} ((1-t)+t)^3 &= ((1-t)+t)((1-t)+t)^2 \\ &= ((1-t)+t)[(1-t)^2 + 2t(1-t) + t^2] \\ = (1-t)[(1-t)^2 + 2t(1-t) + t^2] + t[(1-t)^2 + 2t(1-t) + t^2] &\qquad (4.14) \\ &= (1-t)^3 + 3t(1-t)^2 + 3t^2(1-t) + t^3 \end{aligned}$$

Dette er Bernstein basisfunksjonene av grad 3. Det er som vi ser fire av dem, og vi så ovenfor at vi har tre i det kvadratiske tilfellet. Med binomialkoeffisientene

$$\binom{n}{r} = \frac{n!}{r!(n-r)!} = \frac{1\cdot 2\ldots\cdot n}{1\cdot 2\ldots\cdot r\cdot 1\cdot 2\ldots\cdot (n-r)} = \frac{(n-r+1)\ldots\cdot n}{1\cdot 2\ldots\cdot r}$$



Figur 4.8: Kubiske Bezierkurver kan lages ved å kombinere disse Bernsteinbasisfunksjonene av grad d=3.

kan hver og en av dem skrives

$$B_i^d(t) = \binom{d}{i} t^i (1-t)^{d-i} \tag{4.15}$$

Vi har sett at vi har 3 basisfunksjoner av grad d=2 og 4 basisfunksjoner av grad d=3. Antall basisfunksjoner for en bestemt grad d er altså $d+1$. Ved å blande basisfunksjoner av grad d på forskjellige måter, kan vi uttrykke et hvilket som helst polynom av grad d. Det høres kanskje ikke veldig matematisk ut å blande funksjoner, men disse basisfunksjonene brukes ofte slik og de kalles gjerne **blending functions**. For et polynom p av grad d har vi

$$p(t) = \Sigma_{i=0}^d c_i B_i^d(t) \tag{4.16}$$

hvor c_i er reelle tall. Dette gjelder for en dimensjon, vi kan tenke oss t som uavhengig variabel og $p(t)$ som en funksjon av t. Når vi får med kurver i $\mathbf{R}^2$ å gjøre, gjelder resultatet for $\mathbf{p}(t) = (x(t), y(t))$. For kurver i $\mathbf{R}^3$ skriver vi $\mathbf{p}(t) = (x(t), y(t), z(t))$. Vi har altså

$$\mathbf{p}(t) = \Sigma_{i=0}^d c_i \mathbf{B}_i^d(t) = \mathbf{c}^T \mathbf{B}(t) \tag{4.17}$$

hvor c_i er reelle tall og hver $\mathbf{B}(t) = (x(t), y(t), z(t))$. Skrivemåten

$$\mathbf{p}(t) = \mathbf{c}^T \mathbf{B}(t), \ \mathbf{c}^T = [c_0 \ \ldots c_n] \in \mathbf{R}^n$$

er mye brukt.

4.10.3 Matriseform

Vi kan bruke ligning (4.15) til å bestemme Bernstein basis funksjonene av grad 3 i ligning (4.14), $(1-t)^3$, $3t(1-t)^2$, $3t^2(1-t)$, t^3. Når vi skriver polynomene på standardform (potensform) får vi

$$\begin{aligned}(1-t)^3 &= -t^3+3t^2-3t+1\\ 3t(1-t)^2 &= \ \ 3t^3-6t^2+3t\\ 3t^2(1-t) &= -3t^3+3t^2\\ t^3 &\end{aligned}$$

Vi kan sette opp dette på matriseform $\mathbf{Bx}$ med

$$B = \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

med $\mathbf{x}^T = [t^3\ t^2\ t\ 1]$ Vi kan da skrive ligning 4.17 på matriseform

$$\mathbf{c}^T\mathbf{Bx}$$

Sammenligne med oppgave 4.9.4.

4.10.4 Basisfunksjoner

Basisfunksjoner kalles også blending funksjoner. Vi har polynomer av grad 0, 1, 2, 3 etc og matematiske rom som består av slike funksjoner. Ved hjelp av basisfunksjoner til et gitt matematisk rom kan vi uttrykke alle andre funksjoner i dette rommet.

Et matematisk rom kan være et rom som består av alle mulige polynomer av grad mindre eller lik 3. Dette rommet kan ha betegnelsen $\mathcal{P}_3$. Funksjonene i ligning 4.14 danner en basis for $\mathcal{P}_3$. Det betyr at et hvilket som helst polynom i dette rommet, $p \in \mathcal{P}_3$, kan uttrykkes som en lineærkombinasjon av basisfunksjonene. Og hva betyr så det?

En annen basis for $\mathcal{P}_3$ er monomialbasisen $1,\ x,\ x^2,\ x^3$.

4.11 deCasteljau algoritmen

Vi skal ikke bruke matriseformen av Bezierpolynomer til beregninger. Vi skal i stedet benytte oss av deCasteljau algoritmen. Figur 4.9 viser et eksempel på hvordan algoritmen fungerer. Men aller øverst på figuren har vi en helt vanlig kurve i Paint. En kurve i paint lages ved først å velge endepunkter og trekke en linje mellom endepunktene. Deretter kan man klikke på de to siste kontrollpunktene, og kurven tegnes opp. På figuren er kontrollpunktene P, Q, R, S og kontrollpolygonet markert.

Vi skal nå finne punktet som svarer til parameterverdien $t = \frac{1}{2}$ i en rekursiv prosess.

1. Først må **kontrollpunktene** tegnes opp. Endepunktene her svarer til endepunktene i kubisk Hermite interpolasjon. De to siste kontrollpunktene blir ikke interpolert, men brukes til å styre tangenten i endepunktene.

2. Deretter tegner vi opp **kontrollpolygonet** som består av rette linjer mellom kontrollpunktene.

3. For hvert sett av kontrollpunkter har vi en rett linje som kan uttrykkes ved en affinkombinasjon av punktene. Hvis vi er ute etter et punktet på linjen mellom P og Q velger vi t og finner en affin kombinasjon $tP + (1-t)Q$. Midtpunktet på linjen svarer som kjent til $t = \frac{1}{2}$.

4. Dette gjentas for to de gjenstående delene av kontrollpolygonet. Vi får tre nye punkter $\mathbf{b}_0^1,\ \mathbf{b}_1^1,\ \mathbf{b}_2^1$.

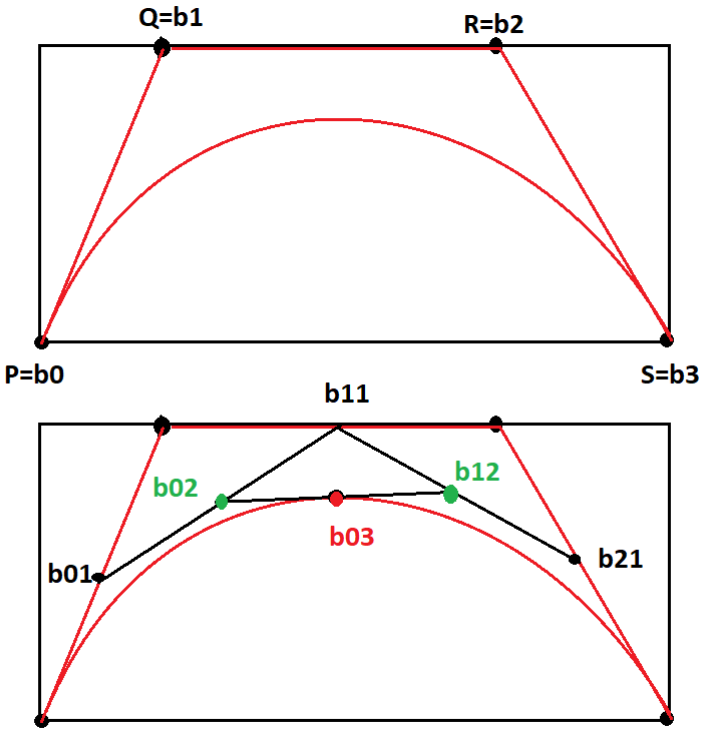
5. Så gjør vi samme prosessen med de tre punktene $\mathbf{b}_0^1,\ \mathbf{b}_1^1,\ \mathbf{b}_2^1$ og får to nye punkter $\mathbf{b}_0^2,\ \mathbf{b}_1^2$

6. Til slutt danner vi $\mathbf{b}_0^3$ av $\mathbf{b}_0^2,\ \mathbf{b}_1^2$.

Indekseringen her gjenspeiler rekkefølgen av kontrollpunktene og hvor mange ganger vi har gjentatt prosessen.

4.11.1 Egenskaper ved Bezierkurver og deCasteljau algoritmen

- Kontrollpunkter.
- Kontrollpolygon.
- Partisjonering av 1.
- Konveksitet.
- Lokal kontroll.
- Corner-cutting.
- Blending functions.



Figur 4.9: Illustrasjon av deCasteljau algoritmen for grad d=3 og $t = \frac{1}{2}$.

4.11.2 C++ implementering av de Casteljau algoritmen

Gitt en egen C++ klasse **BezierCurve** med grad d og en objektvariabel `Vector3D c[4];` for å lagre kontrollpunktene. Anta at kontrollpunktene inneholder noen ekte verdier. deCasteljau algoritmen blir da som nedenfor.

Listing 4.1: deCasteljau algoritmen

```
Vector3D BezierCurve::evaluateBezier(float t)
{
    Vector3D a[4]; // 4=d+1 for kubisk Bezier
    for (int i=0; i<4; i++)
        a[i] = c[i];

    for (int k=d; k>0; k--) //for (int k=1; k<=d; k++)
    {
        for (int i=0; i<k; i++) //for (int i=0; i<=d-k; i++)
            a[i] = a[i] * (1-t) + a[i+1] * t;
    }
    return a[0];
}
```

Sammenligne denne algoritmen og spesielt k-ene med figur 4.9.

4.11.3 Sammensatte Bezierkurver

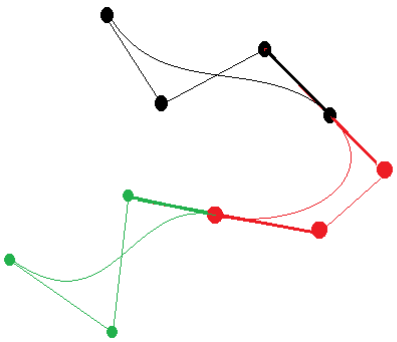
Hva gjør vi når vi har flere enn fire kontrollpunkter, som vi vanligvis har? Vi må sette sammen Bezierkurver på en spesiell måte for å få kontinuerlig derivert i skjøtene. Vi må velge kontrollpunktene slik at tre og tre punkter ligger på en rett linje. Figur 4.10 illustrerer dette. De svarte kontrollpunktene tilhører en kubisk Bezierkurve, B_s. Denne skjøter vi sammen med den røde B_r på en slik måte at to svarte kontrollpunkter og det ene røde ligger på en rett linje. Det samme gjør vi med den røde og den grønn kurven B_g. Vi trenger altså 3 kubiske Bezierkurver til 10 kontrollpunkter: Hver kurve har 4 kontrollpunkter, og kurvene har parvis ett felles kontrollpunkt.

Det er viktig å være klar over at vi nå ikke kan endre på en av kurvene (for eksempel flytte ett kontrollpunkt og samtidig opprettholde C^1 kontinuitet i skjøtene.

Det er mer vanlig å bruke B-splines til slike oppgaver.

4.11.4 Oppgaver

1. Bruk ligning 4.15 til å bestemme Bernstein basis funksjonene av grad 2 i ligning 4.14.
2. Skriv polynomene på standardform (potensform) og sett opp koeffisientmatrisen. Sammenligne med 4.9.4.
3. Hva er de lineære Bernstein basis funksjonene?



Figur 4.10: Tre kubiske Bezierkurver satt sammen med kontinuerlig derivert i skjøtene.

Vis at de deriverte til Bernstein basis funksjonene av grad 3 i ligning 4.14 kan skrives

$$\begin{aligned} -3(1-t)^2 \\ 3(1-t)^2 - 6t(1-t) \\ -3t^2 + 6t(1-t) \\ 3t^2 \end{aligned}$$

Hint: Bruk produktregelen og kjerneregelen. Sett opp på matriseform.

Hva skal vi egentlig med de deriverte til Bezierkurver?

4.11.5

1. Tegn en skisse av hvordan deCasteljau algoritmen fungerer ved å velge $t = \frac{1}{4},\ \frac{1}{3},\ \frac{2}{3},\ \frac{3}{4}$
2. Skriv en egen klasse BezierCurve tilpasset Qt-prosjektet i 3D-programmering. Klassen skal arve baseklassen din.
3. Implementer nødvendige konstruktører og funksjoner slik at du kan tegne en kubisk Bezierkurve med kontrollpolygon i valgt farge.

4.11.6

Skriv en C++ klasse BezierCurve som arver VisualObject. Et BezierCurve objekt skal representere en kubisk Bezierkurve. Funksjonene init() og draw() skal implementeres slik at både kontrollpolygonet og den kubiske Bezierkurven kan rendres med en valgt farge. Lever skjermbilde av Bezierkurve med kontrollpolygon.

4.12 Fasit

4.2.3
$p(x) = x^2 - 4x + 3$

4.6.1
$p(x) = \frac{1}{2}x^3 - \frac{3}{2}x^2 - \frac{1}{2}x + \frac{3}{2}$.

4.6.2
Vi kan sette opp fire ligninger med fire ukjente for å finne konstantene $\mathbf{x}^T = [a\ b\ c\ d]$. Koeffisientmatrisen blir

$$A = \begin{bmatrix} 0 & 0 & 0 & 1 \\ \frac{1}{8} & \frac{1}{4} & \frac{1}{2} & 1 \\ \frac{27}{8} & \frac{9}{4} & \frac{3}{2} & 1 \\ 27 & 9 & 3 & 1 \end{bmatrix} \quad \mathbf{b} = \begin{bmatrix} 1 \\ \frac{9}{16} \\ -\frac{5}{16} \\ 4 \end{bmatrix}$$

Løsningen blir $p(x) = \frac{1}{2}x^3 - x^2 - \frac{1}{2}x + 1$.

4.6.4
Her får vi tre ligninger med tre ukjente a, b og c:
$a \cdot 0 + b \cdot 0 + c = 0$
$a\frac{\pi^2}{4} + b\frac{\pi}{2} + c = 1$
$a \cdot \pi^2 + b \cdot \pi + c = 0$
På matriseform med $\mathbf{x}^T = [a\ b\ c]$ og

$$A = \begin{bmatrix} 0 & 0 & 1 \\ \frac{\pi^2}{4} & \frac{\pi}{2} & 1 \\ \pi^2 & \pi & 1 \end{bmatrix} \quad \mathbf{b} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$$

Vi finner $p_1(x) = -\frac{4}{\pi^2}x^2 + \frac{4}{\pi}x$.

Vi kan regne ut $p_2(x)$ på tilsvarende måte, men også ved et annet resonnement: Vi vet at polynomet er av grad 2 og vi kjenner to nullpunkter, slik at polynomet må være på formen $p_2(x) = a(x - \pi)(x - 2\pi)$. Vi kan da bestemme konstanten a ved å sette inn verdiene for det siste (midterste) interpolasjonspunktet: $a(\frac{9\pi^2}{4} - \pi)(\frac{3\pi}{2} - 2\pi) = -1 \Leftrightarrow a = \frac{4}{\pi^2}$ og vi får
$p_2(x) = \frac{4}{\pi^2}(x - \pi)(x - 2\pi) = \frac{4}{\pi^2}(x^2 - 3\pi x + 2\pi^2) = \frac{4}{\pi^2}x^2 - \frac{12}{\pi}x + 8$.

4.6.5

$$A = \begin{bmatrix} 1 & 1 & 1 & 1 \\ \pi^3 & \pi^2 & \pi & 1 \\ 3 & 2 & 1 & 0 \\ 3\pi^2 & 2\pi & 1 & 0 \end{bmatrix}$$

og $\mathbf{b}^t = [0\ 0\ 1\ \ -1]$ gir $p(x) = -0.46x^2 + 1.93x - 1.46$.
Eksakt utregnet: $\frac{x^2+(\pi+1)x-\pi}{\pi-1}$

4.9.3

$$A = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 3 & 2 & 1 & 0 \end{bmatrix} \quad A^{-1} = \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

Hermite $x^4, x \in [0,1]$ $(x^4)' = 4x^3,\ x_0 = 0,\ x_1 = 1,\ y_0 = 0,\ y_1 = 1,\ y_0' = 0, y_1' = 4 \Rightarrow q = -1,\ c = 2,\ p(x) = 2x^3 - x^2$.

4.9.4 Ligningene 4.4 gir i hvert tilfelle samme matrise

$$A = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 3 & 2 & 1 & 0 \end{bmatrix} \quad A^{-1} = \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

med

$$\mathbf{b}_1 = [1\ 0\ -3\ 0]^T\ \mathbf{b}_2 = [0\ 0\ 3\ 0]^T\ \mathbf{b}_3 = [0\ 0\ -3\ 0]^T\ \mathbf{b}_4 = [0\ 1\ 0\ 3]^T$$

$$\mathbf{x}_1 = [-1\ 3\ -3\ 1]^T\ \mathbf{x}_2 = [3\ -6\ 3\ 0]^T\ \mathbf{x}_3 = [-3\ 3\ 0\ 0]^T\ \mathbf{x}_4 = [1\ 0\ 0\ 0]^T$$

Dette gir

$$\begin{aligned} b_0(t) &= -t^3 + 3t^2 - 3t + 1 \\ b_1(t) &= 3t^3 - 6t^2 + 3t \\ b_2(t) &= -3t^3 + 3t^2 \\ b_3(t) &= t^3 \end{aligned}$$

4.9.5
Vi får 4 addisjoner og 4 multiplikasjoner, altså O(n).

$$\begin{aligned} a = 4: &\quad f = 3 \cdot 0 + 4 &&= 4 \\ a = 3: &\quad f = 3 \cdot 4 + 0 &&= 12 \\ a = 2: &\quad f = 3 \cdot 12 + 8 &&= 44 \\ a = 1: &\quad f = 3 \cdot 44 + 1 &&= 133 \\ a = 0: &\quad f = 3 \cdot 133 + 2 &&= 401 \end{aligned}$$

Vi kan skrive polynomet på nøstet form og ser da hvordan utregningen foregår:

$$\begin{aligned} 2 + x(1 + x(8 + x(0 + x(4)))) \\ 0 + 3 \cdot 4 \\ 8 + 3 \cdot 12 \\ 1 + 3 \cdot 44 \\ 2 + 3 \cdot 133 \end{aligned}$$

Kapittel 5

Affint rom

5.1 Affint rom

Hvorfor trenger vi punkter og kunnskap om punkter, linjer og kunnskap om linjer, trekanter og kunnskap om trekanter? Målet med dette kapitlet er at dere skal få en liten forståelse av hva et affint rom er, hva en parametrisk kurve er, fordelen med parametriske kurver og hvordan man kan implementere en parametrisk kurve.

- Koordinatfri geometri
- Hva er et punkt?
- Konveksitet
- Linje
- Trekant
- Barysentriske koordinater
- Implementering
- Anvendelser6.2.4

5.1.1 Punkt og vektor

Hva er et punkt? For å svare på spørsmålet, er det kanskje en ide å sammenligne med noe vi kan, en vektor. En vektor har størrelse og retning, men ingen posisjon. Et punkt har posisjon, men ingen størrelse eller retning. Vektoraddisjon er definert, og differansen mellom to punkter P og Q er en vektor

$$\mathbf{v} = P - Q$$

Figur 5.1: Barysentriske koordinater for et punkt på en linje gjennom to punkter.

fra Q til P. Men hva med summen P+Q? Kun i enkelte tilfeller gir en sum av punkter mening. Midtpunktet på linjen mellom P og Q kan skrives som

$$\frac{P+Q}{2} = P + \frac{P-Q}{2} = P + \mathbf{v} \tag{5.1}$$

Her ser vi at et punkt kan uttrykkes som et punkt + en vektor $\mathbf{v} = P - Q$. Lar vi Q være et fast punkt og P variere, blir $\mathbf{v}_i = P - Q_i$

Vektor

En vektor $\mathbf{v}$ kan vi skrives som en lineærkombinasjon av vektorer $v_0, \ldots, v_n$ og konstanter $c_0, \ldots, c_n$

$$\mathbf{v} = \sum_{i=0}^{n} c_i \mathbf{v}_i \tag{5.2}$$

Affin kombinasjon

Vi skal nå vise at det fins en lignende måte å skrive et punkt som en kombinasjon av andre punkter på. Vi har sett i likning (5.1) at det kan gi mening for to punkter. Hva skal til for at det gir mening for flere enn to punkter? Altså, for punkter $P_0, \ldots, P_n$ og konstanter $c_0, \ldots, c_n$ skal vi kunne skrive et punkt som en kombinasjon av andre punkter,

$$P = \sum_{i=0}^{n} c_i P_i$$

Vi tar utgangspunkt i likning (5.1) og observerer først at vi kan skrive dette som $\frac{P}{2} = \frac{Q}{2} - \mathbf{v}$. Videre multipliserer vi med 2 og setter inn uttrykket for $\mathbf{v}$ fra likning (5.2), med $\mathbf{v}_i = Q - P_i$ og notasjonen $\tilde{c}_i$ for konstantene.

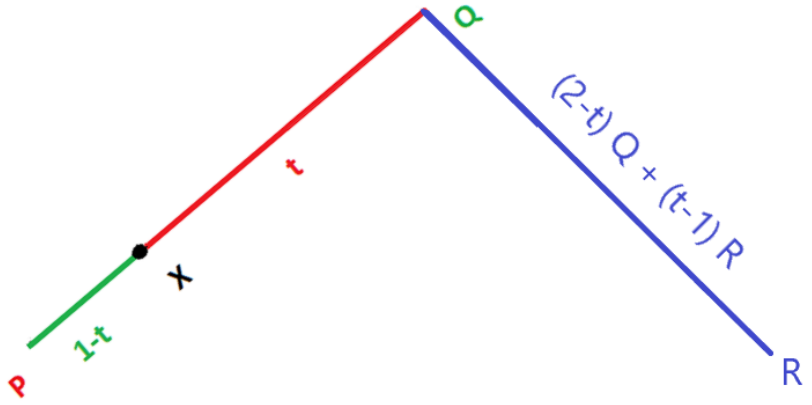
$$\begin{aligned} P &= Q - 2\sum_{i=0}^{n} \tilde{c}_i \mathbf{v}_i \\ &= Q + \sum_{i=0}^{n} c_i \mathbf{v}_i \\ &= Q + \sum_{i=0}^{n} c_i P_i - \sum_{i=0}^{n} c_i Q \\ &= Q + \sum_{i=0}^{n} c_i P_i - Q(\sum_{i=0}^{n} c_i) \end{aligned}$$

hvor $c_i = -2 \cdot \tilde{c}_i$.

Vi ser fra siste linje at vi kan skrive

$$P = \sum_{i=0}^{n} c_i P_i, \qquad \text{hvis } \sum_{i=0}^{n} c_i = 1 \tag{5.3}$$

Dette kalles en affin kombinasjon eller en barysentrisk kombinasjon.



Figur 5.2: Neville's algoritme, ide.

5.1.2 Linje

Linjen som går gjennom to punkter P og Q kan uttrykkes ved en parametrisering.

$$(1-t)P + tQ = P + t\mathbf{v} \tag{5.4}$$

hvor $\mathbf{v} = P - Q$ som ovenfor og $t \in \mathbf{R}$. Mellom P og Q er $0 \leq t \leq 1$. Til venstre for P er t<0 og til høyre for Q er t>1.

5.1.3 Konveksitet

Et rom (eller en mengde) er konvekst dersom den rette linjen mellom to vilkårlige punkter også ligger i rommet (mengden). Linjen mellom P og Q er dermed en konveks mengde.

Barysentriske koordinater for en trekant, se kapittel 6.

5.1.4 Neville's algoritme

Vi skal nå utvide figur 5.1 med et punkt og en linje, denne gangen uten å ha fokus på trekanter. Framstillingen her er delvis hentet fra [4]. Vi har sett at vi kan kombinere punktene P og Q til en linje mellom P og Q ved å la parameter t gå fra 0 til 1. Tilsvarende kan vi kombinere Q og R ved å la parameter u gå fra 0 til 1, eller t gå fra 1 til 2 når $t = 1 + u$. Vi kan altså finne et punkt X mellom P og Q og et punkt Y mellom Q og R på samme måte,

$$\begin{aligned} X &= (1-t)P + tQ,\ 0 \leq t \leq 1 \\ &\text{og} \\ Y &= (2-t)Q + (t-1)R,\ 1 \leq t \leq 2 \end{aligned}$$

Vi ser at t=0 i første linje gir punktet P, t=1 i både første og andre linje gir punktet Q, og t=2 i andre linje gir punktet R. Legg merke til at summen av vektene for hvert punkt er 1 i begge tilfeller.

La oss nå gjenta dette, altså lage en affin kombinasjon (eller en barysentrisk kombinasjon, eller en konveks kombinasjon) av punkt X og Y. Siden X ligger på linjen PQ og Y ligger på linjen QR, blir dette som en kombinasjon av linjene PQ og QR. Vi får da

$$\begin{aligned} P_{PQR}(t) &= X \cdot \frac{2-t}{2} + Y \cdot \frac{t-0}{2} \\ &= ((1-t)P + tQ)\frac{2-t}{2} + ((2-t)Q + (t-1)R) \cdot \frac{t-0}{2} \\ &= \frac{1}{2}(1-t)(2-t)P + t(2-t)Q + \frac{1}{2}t(2-t)R \end{aligned} \tag{5.5}$$

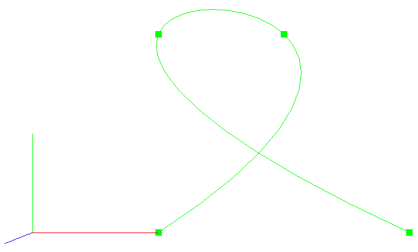
hvor $0 \leq t \leq 2$. Vi må dividere på 2 her for at vektene for X og Y skal summeres til 1.

Hvis vi absolutt vil regne ut de kvadratiske uttrykkene, får vi $P_{PQR} = (\frac{1}{2}t^2 - \frac{3}{2}t + 1)P + (2t - t^2)Q + (\frac{1}{2}t^2 - \frac{1}{2}t)R = p(t) \cdot P + q(t) \cdot Q + r(t) \cdot R$. Vi kan sette inn $t = 0, 1, 2$ i dette uttrykket eller ligning (5.5) og vi får da

	$p(t)$	$q(t)$	$r(t)$
$t = 0$	1	0	0
$t = 1$	0	1	0
$t = 2$	0	0	1
punkt	P	Q	R

Dette viser at $P_{PQR}(t)$ interpolerer alle punktene P, Q og R. Hvis vi legger til enda et punkt og gjentar prosessen kan vi konstruere en kubisk kurve som interpolerer fire punkter.

Punktene P, Q, R ovenfor kalles **kontrollpunkter**, siden man kan kontrollere kurven ved valg og variasjon av disse punktene. For en kubisk kurve har vi fire kontrollpunkter. Her trenger vi ikke tenke på å finne en funksjon som interpolerer punktene, algoritmen tar seg av den saken.



Figur 5.3: Eksempler på kontrollpunkter $(1, 0), (2, 2), (1, 2)$ og $(3, 0)$ og interpolerende kurve.

Implementering av Neville's algoritme

Listing 5.1: Neville's algoritme for kubisk polynom

```
// Neville's algoritme for grad d=3,
// interpolasjon i parameterverdier 0, 1, 2, 3
//        P0123
//     P012 - P123
//   P01 - P12 - P13
//  P0 - P1 - P2 - P3
QVector3D NevilleCurve::nevilleSimple(double t)
{
    // mP0123 initieres med kontrollpunktene
    std::array<QVector3D,4> P=mP0123;

    // Hvilket nivaa i pyramiden er vi paa
    int level=3;

    for (int i=level; i>=0; i--)
    {
        // Hvilken grad er det paa polynomene vi regner ut
        int d = level-i;

        // Vi kombinerer to og to punkter, start til venstre
        // og legger resultatet tilbake i arrayen
        for (int j=0; j<i; j++)
            P[j] = P[j]*(mt[j+1+d]-t)/(d+1) + P[j+1]*(t-mt[j])/(d+1);
    }
    // Den aktuelle verdien p(t) ligger naa i P[0]
    return P[0];
}
QVector3D NevilleCurve::neville(double t) // Normalisert
{
    std::array<QVector3D,4> P=mP0123;
    int level=3;
    for (int i=level; i>=0; i--)
    {
        int d = level-i;
        for (int j=0; j<i; j++)
            P[j] = P[j]*(mt[j+1+d]-t)/(mt[j+1+d]-mt[j])
                 + P[j+1]*(t-mt[j])/(mt[j+1+d]-mt[j]);
    }
    return P[0];
}
```

Kapittel 6

Flater og trianguleringer

6.1 Introduksjon

Hvorfor trenger vi trekanter og kunnskap om trekanter?

- Objekter basert på triangler i OpenGL
- Må kunne lage en topologi for rendring
- Enkleste: Gjentatt opplisting av vertices og bruk av glDrawArrays(). Eksempler på dette er OktaederBall og TriangleSurface klassene fra 3d-programmering. (Egentlig ingen topologi)
- glDrawElements() krever topologi (indeksering av vertices)
- Behov for å navigere på en flate og i 3d. Quadtre og Okttre er datastrukturer som er mye brukt
- Når flaten består av triangler, kan vi navigere direkte på denne

6.1.1 3d flater

Det er klart at vi trenger 3d flater i mange slags spill og simuleringer. Vi må skille mellom parametriske flater som for eksempel en ball og flater som kan uttrykkes som en funksjon $z = f(x, y)$. Vi husker fra Matematikk 1 at en funksjon av to variable $z = f(x, y)$ kan ha partiellderiverte $\frac{\partial f}{\partial x}$ og $\frac{\partial f}{\partial y}$. Vi kan bestemme et tangentplan til flaten i et punkt (x_0, y_0) ved

$$z = f(x_0, y_0) + \frac{\partial f}{\partial x}(x_0, y_0)(x - x_0) + \frac{\partial f}{\partial y}(x_0, y_0)(y - y_0)$$

Normalvektor

Når vi har funnet ligningen for tangentplanet til flaten $f(x, y)$, er det enkelt å regne ut en normalvektor til flaten. I tangeringspunktet blir jo det det samme som normalen til tangentplanet.

1. La $P_0 = (x_0, y_0, z_0)$. Her er $z_0 = f(x_0, y_0)$.

2. Velg to punkter P_1 og P_2 og bruk ligningen for tangentplanet til å regne ut verdiene.

3. Regn ut vektorene $\mathbf{u} = P_1 - P_0$ og $\mathbf{v} = P_2 - P_0$.

4. Regn ut vektoren $\mathbf{n} = \mathbf{u} \times \mathbf{v}$ ved å bruke determinanter:

$$\begin{vmatrix} \mathbf{i} & \mathbf{j} & \mathbf{k} \\ u_x & u_y & u_z \\ v_x & v_y & v_z \end{vmatrix} = [u_y v_z - v_y u_z,\ -(u_x v_z - v_x u_z),\ u_x v_y - v_x u_y] \tag{6.1}$$

5. Normaliser normalvektoren slik at den får lengde 1.

Eksempel

La $f(x, y) = 1 - xy$ og $(x_0, y_0) = (\frac{1}{2}, \frac{1}{4})$. Vi finner $z_0 = 1 - \frac{1}{2} \cdot \frac{1}{4} = \frac{7}{8}$.

De partiellderiverte blir $\frac{\partial f}{\partial x} = -y$ og $\frac{\partial f}{\partial y} = -x$. Innsatt for punktet $(\frac{1}{2}, \frac{1}{4})$ får vi $\frac{\partial f}{\partial x}(\frac{1}{2}, \frac{1}{4}) = -\frac{1}{4}$ og $\frac{\partial f}{\partial y}(\frac{1}{2}, \frac{1}{4}) = -\frac{1}{2}$.

Vi har altså et punkt $P_0 = (\frac{1}{2}, \frac{1}{4}, \frac{7}{8})$ hvor vi skal finne tangentplan og normal.

Ligningen for tangentplanet i $P_0 = (\frac{1}{2}, \frac{1}{4}, \frac{7}{8})$ blir
$z = \frac{7}{8} - \frac{1}{4}(x - \frac{1}{2}) - \frac{1}{2}(y - \frac{1}{4}) = \frac{9}{8} - \frac{1}{4}x - \frac{1}{2}y$.

Vi skal nå regne ut en normalvektor til flaten.

1. Vi kan velge oss to punkter $P_1 = (1, 0, z_1)$, og $P_2 = (0, 1, z_2)$ og nå skal vi bruke ligningen for tangentplanet til å beregne z-verdiene.
   $P_1 = (1, 0, z_1) \Rightarrow z_1 = \frac{9}{8} - \frac{1}{4} - 0 = \frac{7}{8}$ og $P_2 = (0, 1, z_2) \Rightarrow z_2 = \frac{9}{8} - 0 - \frac{1}{2} = \frac{5}{8}$

2. Vektorene $\mathbf{u} = P_1 - P_0 = (1, 0, \frac{7}{8}) - (\frac{1}{2}, \frac{1}{4}, \frac{7}{8}) = [\frac{1}{2}, -\frac{1}{4}, 0]$
   og $\mathbf{v} = P_2 - P_0 = (0, 1, \frac{5}{8}) - (\frac{1}{2}, \frac{1}{4}, \frac{7}{8}) = [-\frac{1}{2}, \frac{3}{4}, -\frac{1}{4}]$.

3. Normalvektoren $\mathbf{n} = \mathbf{u} \times \mathbf{v}$ blir

$$\begin{vmatrix} \mathbf{i} & \mathbf{j} & \mathbf{k} \\ \frac{1}{2} & -\frac{1}{4} & 0 \\ -\frac{1}{2} & \frac{3}{4} & -\frac{1}{4} \end{vmatrix} = = [\tfrac{1}{16}\ \ \tfrac{1}{8}\ \ \tfrac{1}{4}]^T$$

4. Med desimaltall blir $\mathbf{n} = [0.0625, 0.125, 0.25]$.

5. Det som gjenstår nå er å normalisere normalvektoren slik at den får lengde 1. Vi finner $|\mathbf{n}|^2 = \frac{21}{256} \approx 0.0820 \Rightarrow n = 0.286$.

   Enhetsnormalvektoren blir da $[0.218, 0.436, 0.873]$.

Kontroller utregningene!

6.1.2 Normalvektor II

Vi har i eksemplet ovenfor sett hvordan vi kan bruke partiellderiverte til å lage et tangentplan, og deretter hvordan vi lager en normalvektor ved hjelp av tangentplanet.

Vi skal evaluere en kjent funksjon $f(x, y)$ i en del punkter, og vi vil trenge normalvektor i hvert punkt. Må vi da finne en ny ligning for et nytt tangentplan i hvert punkt? Og hvordan skal vi velge ut punktene i tangentplanet?

La oss studere eksemplet ovenfor litt nærmere. Det som er klart er at uttrykket for de partiellderiverte er $\frac{\partial f}{\partial x} = -y$ og $\frac{\partial f}{\partial y} = -x$. Videre, for hvert punkt (x_0, y_0) kan vi regne ut verdien til de partiellderiverte ved innsetting. Ovenfor fikk vi $\frac{\partial f}{\partial x}(\frac{1}{2}, \frac{1}{4}) = -\frac{1}{4}$ og $\frac{\partial f}{\partial y}(\frac{1}{2}, \frac{1}{4}) = -\frac{1}{2}$.

Det betyr:

1. I x-retning endrer z-verdien seg med $-\frac{1}{4}$ når x øker med 1 (og y er konstant).
   En vektor i tangentplanet i x-retning er da $\mathbf{u} = [1, 0, -\frac{1}{4}]$.
2. I y-retning endrer z-verdien seg med $-\frac{1}{2}$ når y øker med 1 (og y er konstant).
   En vektor i tangentplanet i x-retning er da $\mathbf{v} = [0, 1, -\frac{1}{2}]$.

Vi kan da finne en normalvektor av disse to: $\mathbf{n} = \mathbf{u} \times \mathbf{v}$

$$\begin{vmatrix} \mathbf{i} & \mathbf{j} & \mathbf{k} \\ 1 & 0 & -\frac{1}{4} \\ 0 & 1 & -\frac{1}{2} \end{vmatrix} = [\tfrac{1}{4}\ \tfrac{1}{2}\ 1]^T \quad = 4\ [\tfrac{1}{16}\ \tfrac{1}{8}\ \tfrac{1}{4}]^T$$

Vi ser at vi her har funnet samme normalvektor som i eksemplet ovenfor, bortsett fra lengden. Når vi normaliserer, får vi samme vektor. Og vi har ikke lenger noe behov for å regne ut ligningen for tangentplanet.

6.1.3 Konstruksjon av 3d flater

Vi skiller mellom det å lage en 3d flate og selve rendringen. Noen har kanskje vært borti enkleste form for en 3d flate - et heightmap, hvor vi strengt tatt kan angi høyden i hvert punkt (x,y) for et rektangulært område.

6.1.4 Regulært grid (rektangulært mesh)

En generell metode for å generere en 3d flate er å evaluere en funksjon $f(x, y)$ i en dobbel for-løkke. Man får da et rektangulært område i xy-planet (eller xz-planet) med z-verdier i tillegg til x- og y-verdier. Siden vi skal tegne trekanter, må hvert rektangel (quad) deles i to. Vi har tidligere gått gjennom et eksempel på dette (figur 2.2.3). Med normaler i vertexene blir dette som i oppgave 6.2.14.

6.1.5 Triangulering basert på irregulære datapunkter

GPU-en trenger normaler. Normalene beregnes ulikt avhengig av hvilken shadermodell som skal benyttes. Det er ikke nødvendig å regne ut tangentplan og normaler i hvert vertex som i eksemplet ovenfor. Vi kan gjøre noen forenklinger og likevel få et akseptabelt resultat.

Ettersom vi jobber tett på GPU med trekanter, skal vi fokusere på å bruke flater sammensatt av trekanter, men ikke nødvendigvis trekanter som ligger ordnet. Slike flater går gjerne under betegnelsen *mesh* når det kommer til implementering. Vi starter med å fokusere på *en* trekant.

6.1.6 Stykkevis bilineær interpolant

En flate som er satt sammen av trekanter er lineær i x og y og kontinuerlig, men generelt ikke deriverbar over kantene og hjørnene. En slik flate kalles en stykkevis bilineær interpolant. Flaten er definert over et område $\Omega \subset \mathbf{R}^2$ og interpolerer et gitt datasett. Vi har studert kurver, deriverbarhet og kontinuitet og skjønner at det blir mye jobb å lage glatte flater fra bunnen av. Vi skal derfor nøye oss med å bruke trekantbaserte flater. De er såpass viktige at de har en egen betegnelse i matematikken: Splinerommet $\mathbf{S}_0^1(\Omega)$.

6.1.7 Konveksitet

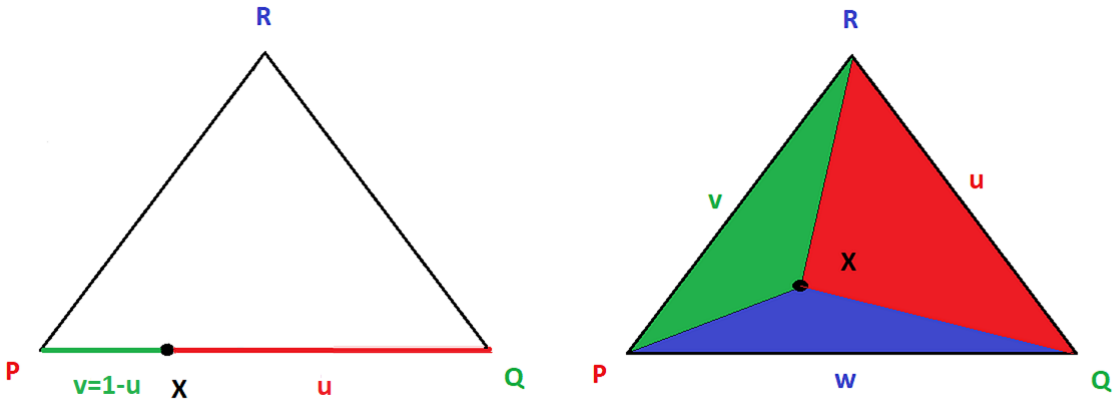
Området Ω som funksjonen er definert over skal være et konvekst område. Når vi deler opp dette området i triangler, får vi en samling triangler som vi kaller Δ. Dette skal også være et konvekst område.

6.2 Barysentriske koordinater

Vi skal nå innføre et eget, lokalt koordinatsystem for *en* trekant. Vi har tidligere lært om barysentriske koordinater for en linje, se figur 5.1. Vi kan legge til ett ekstra punkt, og få en trekant. Se figur 6.1. Her har de barysentriske koordinatene for linjen gjennom P og Q fått nye navn. Summen av dem er fortsatt 1. Det er ganske opplagt at det tredje punktet R ikke kan ha noen innvirkning på et punkt på linjen gjennom P og Q.

Hvis vi nå flytter punktet X inn i trekanten, kan vi (forhåpentligvis) forestille oss intuitivt at X kan skrives som en kombinasjon av P, Q og R. De barysentriske koordinatene til punktet X med hensyn på P, Q og R er henholdsvis u, v og w. Vi har $u + v + w = 1$ og størrelsen på disse barysentriske koordinatene er indikert med en farget trekant. Vi ser at når X ligger på linjen gjennom P og Q, forsvinner den blå trekanten og $w = 0$. Tilsvarende gjelder for de to andre barysentriske koordinatene. Et punkt X kan altså skrives som en barysentrisk kombinasjon av hjørnene P, Q og R til en trekant T, og denne trekanten T er valgt vilkårlig.

La Boundary(T) betegne kantene som omslutter trekanten T og la Int(T) be-



Figur 6.1: Barysentriske koordinater til et punkt i en trekant. Se også figur 5.1.

tegne det indre området til T. Vi har

$$\begin{aligned} u+v+w &= 1 \\ u \geq 0, v \geq 0, w \geq 0, & \quad X \in T \\ u > 0, v > 0, w > 0, & \quad X \in Int(T) \\ u = 0 \,\|v = 0 \,\|w = 0 & \quad X \in Boundary(T) \\ u < 0 \,\|v < 0 \,\|w < 0 & \quad X \notin T \end{aligned} \tag{6.2}$$

6.2.1 Formler for utregning

Sett $\mathbf{x}_1 = Q - P$ og $\mathbf{x}_2 = R - P$ på figur 6.1. Vektorene $\mathbf{x}_1$ og $\mathbf{x}_2$ utspenner da et parallellogram med areal $\mathbf{x}_1 \times \mathbf{x}_2$. Kryssproduktet har som vi kanskje husker to geometriske tolkninger: Det er en vektor vinkelrett på $\mathbf{x}_1$ og $\mathbf{x}_2$, med et fortegn som angir hvilken side den er på. Arealet av parallellogrammet utspent av $\mathbf{x}_1$ og $\mathbf{x}_2$ og selvsagt det dobbelte til arealet av trekanten T.

Videre, sett $\mathbf{u}_1 = Q - X$ og $\mathbf{u}_2 = R - X$ på figuren. Vektorene $\mathbf{u}_1$ og $\mathbf{u}_2$ utspenner da et parallellogram med areal $\mathbf{u}_1 \times \mathbf{u}_2$. Lengden av denne vektoren er arealet av parallellogrammet utspent av $\mathbf{u}_1$ og $\mathbf{u}_2$ og altså det dobbelte til arealet av den røde trekanten. Fortegnet har også her betydning!

Vi gjør det samme med de to siste fargede trekantene.

$\mathbf{v}_1 = R - X$ og $\mathbf{v}_2 = P - X$. Den grønne trekantens areal er halvparten av arealet $\mathbf{v}_1 \times \mathbf{v}_2$ (og husk på fortegnet).

$\mathbf{w}_1 = P - X$ og $\mathbf{w}_2 = Q - X$. Den blå trekantens areal er halvparten av arealet $\mathbf{w}_1 \times \mathbf{w}_2$ (og husk på fortegnet).

Vi har da følgende formler for utregning av de barysentriske koordinatene:

$$\begin{aligned} u &= \frac{\mathbf{u}_1 \times \mathbf{u}_2}{\mathbf{x}_1 \times \mathbf{x}_2} \\ v &= \frac{\mathbf{v}_1 \times \mathbf{v}_2}{\mathbf{x}_1 \times \mathbf{x}_2} \\ w &= \frac{\mathbf{w}_1 \times \mathbf{w}_2}{\mathbf{x}_1 \times \mathbf{x}_2} \end{aligned} \tag{6.3}$$

Utregning av determinater er vist i ligning 6.1.

6.2.2 $\pm$

Fortegnet til kryssproduktet, eller retningen på vektoren om man vil, spiller en viktig rolle i anvendelse av barysentriske koordinater. Hva skjer hvis vi for eksempel flytter punktet X på undersiden av linjen gjennom P og Q? Eller ut over en av de to andre kantene til T?

6.2.3 Implementering

Gitt en funksjon $z = f(x, y)$ eller en stykkevis bilineær funksjon, altså en samling trekanter - som vi er vant til nå. Vi kan regne ut de barysentriske koordinatene med 3-dimensjonale vektorer. Men det gir unødvendig mange operasjoner (multiplikasjoner/divisjoner - kostbar utregning) sammenlignet med å regne ut de barysentriske koordinatene til punkter i planet. Projeksjonen av en tredimensjonal triangulær flate i planet danner en todimensjonal triangulær flate og de barysentriske koordinatene kan regnes ut på denne, det blir samme resultat.

Funksjonen nedenfor ligger i en klasse **Vec2** med overlastede operatorer for subtraksjon og kryssprodukt (hattetegnet brukes vanligvis til kryssprodukt operator). Funksjonen har tre parametre inn: (x,y) koordinatene til den aktuelle trekanten, og kalles opp av punktet som vi skal ha de barysentriske koordinatene til. De barysentriske koordinatene returneres i en Vec3.

```
Vec3 barycentricCoordinates(const Vec2& p1, const Vec2& p2, const Vec2& p3)
{
    Vec2 p12 = p2-p1;
    Vec2 p13 = p3-p1;

    Vec3 n = p12^p13;
    float areal_123 = n.length(); // dobbelt areal

    Vec3 baryc; // til retur. Husk
    // u
    Vec2 p = p2 - *this;
    Vec2 q = p3 - *this;
    n = p^q;
    baryc.x = n.z/areal_123;
    // v
    p = p3 - *this;
    q = p1 - *this;
```

```
    n = p^q;
    baryc.y = n.z/areal_123;
    // w
    p = p1 - *this;
    q = p2 - *this;
    n = p^q;
    baryc.z = n.z/areal_123;

    return baryc;
}
```

6.2.4 Anvendelser

Barysentriske koordinater har mange praktiske anvendelser. Blant annet kan en fragmentshader bruke barysentriske koordinater til å interpolere farger i vertices.

Beregne høyde $z = f(x, y)$

Anta at vi har et spillerobjekt eller et annet interaktivt objekt som skal bevege seg på et tresdimensjonalt underlag. Vi kan styre spilleren i xy-planet for eksempel med tastene wasd. Men hvordan beregne riktig høyde? Vi kan bruke barysentriske koordinater til å beregne høyden. Når vi har funnet ut hvilken trekant i xy-planet spilleren er i, kan vi beregne høyden som en kombinasjon av høyden i hvert vertex. Terrenget vårt er jo generert ved å regne ut høyden i punkter med jevn avstand og deretter generere en regulær triangulering. Hvis spilleren har posisjon (x,y) og er i trekant PQR, regner vi ut de barysentriske koordinatene u, v, w med hensyn på (x,y) og regner ut høyden

$$f(x, y) = u \cdot f(P_x, P_y) + v \cdot f(Q_x, Q_y) + w \cdot f(R_x, R_y)$$

Men hvordan finne igjen riktig trekant? I TriangleSurface klassen ligger vertexdata sekvensielt i en vektor. Gitt spillerens (x,y) koordinater kan vi da søke gjennom alle vertexer helt til vi finner en trekant hvor de barysentriske koordinatene $u, v, w \in [0, 1]$. Dette er O(n) søk, men det fungerer, også for irregulære trianguleringer.

Søking

Barysentriske koordinater kan også brukes til søking. La oss se på et eksempel. Nedenfor er en fil med vertex data for 12 punkter gjengitt, riktignok uten farge/normal og teksturkoordinater. Dette er altså rene posisjonsdata. Hver linje representerer en trekant (og det går fint an å lese inn filen slik, selv om vi vanligvis ville legge ett vertex på hver linje.

```
42
0.0 0.0 2.0    1.5 0.0 2.0    0.0 1.0 2.0
0.0 1.0 2.0    1.5 0.0 2.0    2.5 1.0 1.0
0.0 1.0 2.0    2.5 1.0 1.0    2.0 2.0 2.0
0.0 1.0 2.0    2.0 2.0 2.0    1.0 3.0 1.0
0.0 1.0 2.0    1.0 3.0 1.0    0.0 4.0 2.0
1.5 0.0 2.0    4.0 0.5 2.0    2.5 1.0 1.0
```

Figur 6.2: Datasett for triangulering.

```
1.5 0.0 2.0    4.0 0.0 2.0    4.0 0.5 2.0
3.5 2.5 1.0    4.0 0.5 2.0    4.0 4.0 2.0
2.0 2.0 2.0    2.5 1.0 1.0    4.0 0.5 2.0
2.0 2.0 2.0    4.0 0.5 2.0    3.5 2.5 1.0
2.5 4.0 2.0    3.5 2.5 1.0    4.0 4.0 2.0
2.0 2.0 2.0    3.5 2.5 1.0    2.5 4.0 2.0
1.0 3.0 1.0    2.0 2.0 2.0    2.5 4.0 2.0
0.0 4.0 2.0    1.0 3.0 1.0    2.5 4.0 2.0
```

Det er tilsammen 14 trekanter. Uten noen form for topologi eller indeksering må vi duplisere disse punktene for å tegne en flate med glDrawArrays(), og vi får da $14 \cdot 3 = 42$ vertices. Vi trenger i tillegg som nevnt farge/normal og teksturkoordinater.

6.2.5 Lineært søk

Selv med denne enkle (eller kanskje heller tungvinte) representasjonen av en triangulær flate, har barysentriske koordinater interessante anvendelser. Hvis vi skal finne igjen hvilket triangel T_i et gitt punkt P tilhører, kan vi gjennomløpe arrayen sekvensielt, lese inn tre punkter om gangen og regne ut de barysentriske koordinatene til P med respekt på T_i. Hvis alle *u, v, w* er positive, har vi funnet det riktige trianglet. Hvis ikke, fortsetter søket.

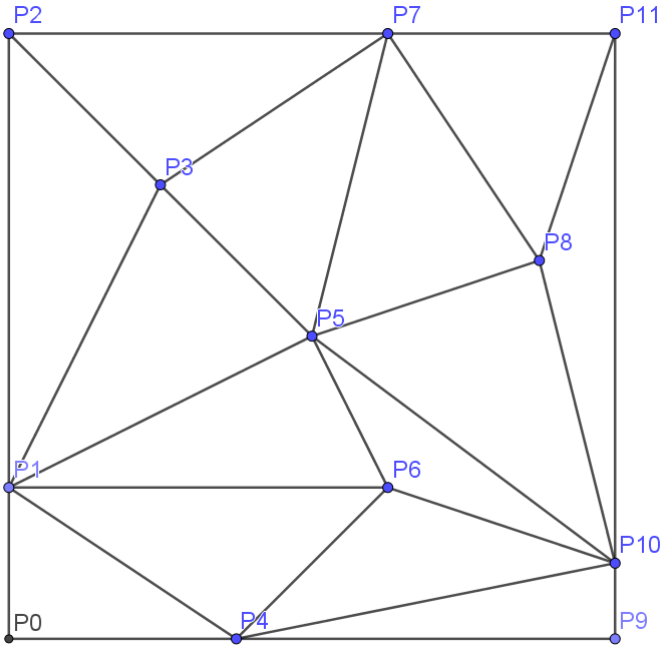
Her blir søketiden $O(n)$ og vi bruker unødig mye lagringsplass. Men det virker, og noen ganger trenger vi også glDrawArrays()

6.2.6 Triangulering med indeksering

Gitt n punkter trenger vi bare å lagre n forekomster av vertex data. For å kunne definere trekanter og rendre noe, må vi da i tillegg angi en topologi/indeksering og bruke glDrawElements(). For eksemplet i figur 6.2.4 ovenfor blir da geometrien 12 linjer med vertex data:

```
12
0.0 0.0 2.0
0.0 1.0 2.0
0.0 4.0 2.0
1.0 3.0 1.0
1.5 0.0 2.0
2.0 2.0 2.0
2.5 1.0 1.0
2.5 4.0 2.0
3.5 2.5 1.0
4.0 0.0 2.0
4.0 0.5 2.0
4.0 4.0 2.0
```

og topologien 14 linjer med indekser:



Figur 6.3: En triangulering for datasett 6.2.4.

```
14
0 4 1
1 4 6
1 6 5
1 5 3
1 3 2
4 10 6
4 9 10
8 10 11
5 6 10
5 10 8
7 8 11
5 8 7
3 5 7
2 3 7
```

Sammenlign med figur 6.3. Merk at indekseringen starter på 0 og at alle vertices er angitt i positiv omløpsretning. Konsekvens er viktig her!

Vi oppnår her et mindre datasett, noe som er generelt er ønskelig. Men søking etter et triangel som inneholder et gitt punkt, $P \in T_i$ må fortsatt foregå lineært selv om vi bruker topologien. Søketiden er dermed fortsatt $O(n)$.

6.2.7 Triangulering med ekte topologi

Vi kan utvide indekseringen ovenfor til å inkludere naboinformasjon. Vi har her

- Geometri lagret unikt og separat.
- Indeksering av trekanter. Hver trekant har implisitt en egen indeks/nummer som starter på 0 og svarer til posisjon i array/vektor.
- Vi tilføyer informasjon om nabotrekanter på hver linje.

En trekant har like mange naboer som kanter, bortsett fra trekanter langs randen til det triangulerte området. To trekanter er **naboer** hvis de har en felles kant. Vi skal se på et eksempel hvordan naboene kan angis for hver trekant.

Med samme datasettet som i 6.2.4 kan vi angi en komplett triangulering med naboinformasjon på følgende vis:

```
14
0 4 1     1 -1 -1
1 4 6     5 2 0
1 6 5     8 3 1
1 5 3     12 4 2
1 3 2     13 -1 3
4 10 6    8 1 6
4 9 10     -1 5 -1
8 10 11   -1 10 9
5 6 10    5 9 2
5 10 8    7 11 8
7 8 11    7 -1 11
5 8 7     10 12 9
3 5 7     11 13 3
2 3 7     12 -1 4
```

Vi ser at dersom en trekant ikke har noen nabo over en gitt kant, angis dette med -1. Sammenlign med figur 6.4

6.2.8 Topologisk søk algoritme

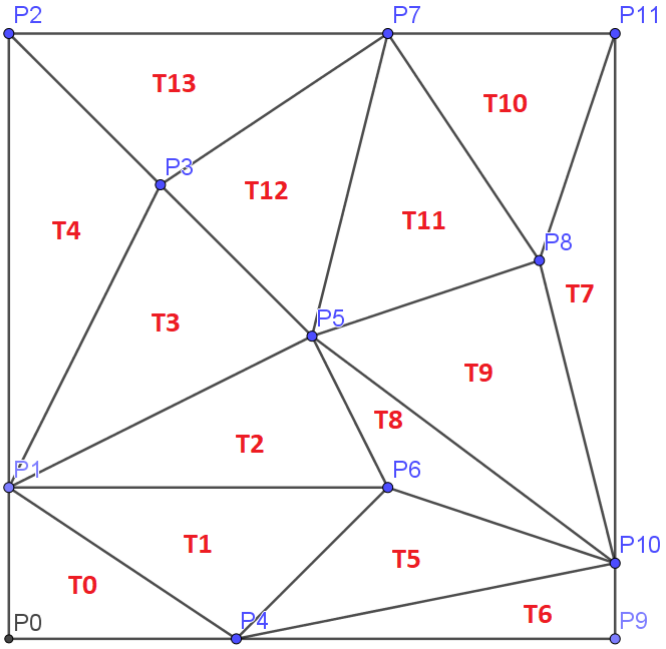
Vi kan nå utnytte naboinformasjonen for hvert triangel til en søkealgoritme. Problemet er som nevnt tidligere:

Gitt et punkt P og en triangulering

$$\Delta = \{T_i\}_{i=0}^{n-1}$$

1. Velg en initiell trekant og beregn barysentriske kordinater til P med hensyn på denne.
2. Hvis $u, v, w \geq 0$ er trekanten funnet.
3. Ellers fortsett søket på den trekanten som svarer til den minste (mest negative) barysentriske koordinaten

Dette kan lett omskrives til en løkke, men krever en grundigere forklaring. Igjen bruker vi eksempeldata som ovenfor.



Figur 6.4: En triangulering med naboinformasjon.

6.2.9 Eksempel

Ta deg tid til å regne ut dette eksemplet manuelt!

- La $P = (3.5, 3.5)$
- Start søket i trekant i=0.
- **do** {
  Beregn barysentriske koordinater for trekant i
  if $u, v, w \geq 0$ funnet = true;
  else i = nabo som svarer til minste barysentriske koordinaten
  } **while** (!funnet)

6.2.10 Oppgave

1. Lag manuelt en tekstfil av datasettet i avsnitt 6.2.4.
2. Skriv en funksjon som leser tekstfilen i forrige punkt, beregner normaler for hvert vertex, og skriver en tekstfil med Vertexdata. Forklar tydelig hvordan normaler beregnes. Bruk av teksturkoordinater er valgfritt.
3. Opprett et TriangleSurface objekt. Implementer initialize() og draw() funksjonene til å tegne flaten med glDrawArrays(). Bruk Phong shadermodell (fra 3D-programmering).
4. Implementer eksemplet i avsnitt 6.2.5:
   - Start søket i midtpunktet (tyngdepunktet) til trekant 0.
   - Lagre tyngdepunktet for hver trekant som testes i løkka.
   - De lagrede punktene utgjør en stykkevis lineær kurve. Lag en simulering av hvordan dette søket foregår: La et selvvalgt objekt starte i trekant 0 og bevege seg med konstant tidssteg langs den stykkevis lineære kurven.
   - Objektet skal følge flaten. Bruk barysentriske koordinater (for hvert tidssteg) til å beregne høyden til objektet. Siden objektet beveger seg på en rett linje mellom to punkter, er det tilstrekkelig å bruke barysentriske koordinater for en linje om gangen.
5. Beskriv detaljene i rapport og forklar på videopresentasjon.

6.2.11 Oppgave

Denne oppgaven er ligner på den forrige, men vi skal her bruke en triangulering med indekser.

1. Lag manuelt en tekstfil av datasettet i avsnitt 6.2.6 og en indeksfil av datasettet(indeksene) i avsnitt 6.2.7
2. Skriv en klasse i 3DProg21 prosjektet som arver VisualObject og kan lese inn og lagre i minne datasett og indekser for en triangulering med naboinformasjon, som forklart i avsnitt 6.2.7. En struct eller lignende bør brukes til å lagre indeksene og nabotrekantene. Navn på klassen skal være noe annet enn TriangleSurface.
3. Implementer oppgave 6.2.10 punkt 2 i den nye klassen. Kommenter/forklar eventuelle endringer, for eksempel i forhold til beregning av normaler.
4. Implementer initialize() og draw() funksjonene til å tegne flaten med glDrawElements().
5. Gjenta oppgave 6.2.10 punkt 4 med bruk av indekseringen/topologien.
6. Gjenta oppgave 6.2.10 punkt 5, men la nå objektet bevege seg langs en Bezierkurve (eller flere/en sammensatt Bezierkurve) som har den stykkevis lineære funksjonen som kontrollpolygon.
7. Beskriv detaljene i rapport og forklar på videopresentasjon.

6.2.12 Oppgave

En høydekurve eller konturlinje er en sammenhengende kurve/linje som går gjennom punkter på en overflate med lik høyde. Skriv kode som finner alle høydekurver for en gitt høydeverdi på området fra forrige oppgaver. Tegn høydekurvene som stykkevis lineære kurver. Test for z=1.5. Beskriv detaljene i rapport og forklar på videopresentasjon.

6.2.13 Oppgave

Se avsnitt 2.1.9. Implementer normaler i hvert vertex for flaten. For eksempel skal normalen til den grønne trekanten lagres i vertex 6, 7 og 8. Normaler for den røde trekanten skal lagres i vertex 9, 10 og 11.

- Kryssprodukt av vektor fra 6-7 og 6-8 gir en normal til den grønne trekanten
- 6-7: $\mathbf{u}$ = [h, 0, z(7)-z(6)];
- 6-8: $\mathbf{v}$ = [0, h, z(8)-z(6)];
- $\mathbf{n} = \mathbf{u} \times \mathbf{v}$;
- n.normalize();
- Oppdater vertex lagring (noen endringer i koden må gjøres):
- mVertices.push_back(Vertex{x,y,z,n.x(),n.y(),n.z()});
- Gjenta dette for vertex 6, 7 og 8
- Gjenta dette for rødtrekant

6.2.14 Oppgave

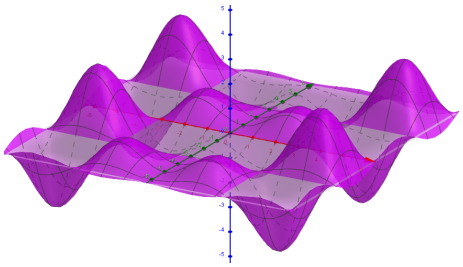
Gitt $f(x,y) = \frac{x}{2}\sin x \cdot \sin y,\ -\pi \leq x \leq \pi, -\pi \leq y \leq \pi$.

a) Bestem $\frac{\partial f}{\partial x}$ og $\frac{\partial f}{\partial y}$

b) Finn et uttrykk for normalvektoren i et punkt (x_0, y_0)

c) Sett opp en algoritme/funksjon for å beregne f og f' i et rutenett med avstand h

d) Sett opp en algoritme/funksjon for å lage triangler av flaten, for bruk med glDrawArrays()



Figur 6.5: $f(x,y) = \frac{x}{2}\sin x \cdot \sin y,\ -\pi \leq x \leq \pi, -\pi \leq y \leq \pi$

Kapittel 7

B-spline kurver

7.1 Introduksjon

- B-splines er stykkevise polynomer
- lære litt om B-splines
- Her: funksjoner av en variabel $f(x)$
- Generelt: Parametriske kurver $\vec{\mathbf{f}}(t) = (x(t), y(t), z(t)$
- definisjon
- grad d
- skjøt (eng: knot), skjøtvektor
- multiple skjøter
- kontinuitet
- Sammenheng med Bezierkurver
- Kunne regne ut kvadratiske B-spline basisfunksjoner på en gitt skjøtvektor

Senere (i Visualisering og Simulering) skal vi studere en algoritme for B-splines som ligner på deCasteljau algoritmen.

Figur 7.1: Normalfordelingen.

7.1.1 Litt motivasjon

Normalfordelingsfunksjonen (Gaussfordelingsfunksjonen) er gitt ved

$$\frac{1}{\sqrt{2\pi}\sigma}e^{-\frac{(x-\mu)^2}{2\sigma^2}}, -\infty < x < \infty$$

Med $\mu = 0$ og $\sigma = 1$ blir funksjonen

$$\frac{1}{\sqrt{2\pi}}e^{-x^2}$$

Dette er en av de mest berømte funksjonene i matematikken. Integralet $\int_{-\infty}^{\infty} f(x)dx = 1$, men det er umulig å regne ut analytisk. Vi skal se på den enklere funksjonen

$$f(x) = e^{-x^2}$$

Heller ikke denne er det mulig å integrere analytisk. Grafen til begge funksjonene er klokkeformet. Hvis vi klarer å approksimere disse eksponentialfunksjonene med polynomer, blir det i det minste mulig å regne ut et integral.

7.2 Splines er stykkevise polynomer

Funksjonen $f(x) = x^2$ er den enkleste andregradsfunksjonen vi kjenner. (Et annet ord for andregradsfunksjon er kvadratisk polynom). Polynomet $p_0(x) = \frac{1}{2}x^2$ er bare $f(x)$ skalert, og er enkelt å tegne opp, og det er enkelt å regne ut $p_0(1) = \frac{1}{2}$. Notasjonen p_0 her er valgt fordi vi skal ha flere polynomer, og her ikke noe med graden (som er 2) å gjøre. Men man kan noen ganger bli forvirret av slikt.

Det er ofte nyttig å skrive polynomer på en annen måte enn på standardform/potensform. Hvis vi skriver $p_2(x) = \frac{1}{2}(x-3)^2$, så er det ganske enkelt å se at et variabelskifte u=x-3 gir et uttrykk $p_2(u) = \frac{1}{2}u^2$, det er altså $p_0(x)$ flyttet tre enheter i positiv x-retning. Det er enkelt å regne ut $p_2(2) = \frac{1}{2}$. Videre ser vi at $p_0(0) = p_2(3) = 0$. Disse to polynomene har hver sin symmetrilinje, og de er innbyrdes symmetriske om $x = \frac{3}{2}$.

La nå $p_0(x)$ være definert for $0 \leq x \leq 1$ og la $p_2(x)$ være definert for $2 \leq x \leq 3$.

Hvordan kan vi finne et kvadratisk polynom $p_1(x), 1 \leq x \leq 2$ som skjøter sammen disse to andre polynombitene? Vi skal ha et polynom på formen $q(x) = ax^2 + bx + c$, da er $q'(x) = 2ax + b$. Vi krever at den deriverte er kontinuerlig i skjøtene, noe som gir får betingelsene $p_0'(1) = 1 = q'(1)$ og $q'(2) = p_2'(2) = -1$. Av dette får vi to ligninger med to ukjente a og b, med løsning a= -1 og b=3. Altså er $q(x) = -x^2 + 3x + c$. Vi kan bestemme c av betingelsen $p_1(\frac{1}{2}) = q(\frac{1}{2})$ og får $c = -\frac{3}{2}$

Vi har nå bestemt en polynombit som passer mellom $p_0(x)$ og $p_2(x)$,

$$p_1(x) = -x^2 + 3x - -\frac{3}{2} = -(x - \frac{3}{2})^2 + \frac{3}{4}$$

Hvis vi nå tegner opp disse tre polynombitene i et egnet verktøy, ser vi at de passer finst sammen, de er kontinuerlige og faktisk deriverbare i skjøtene x=1 og x=2.

Øving

Vis ved regning at $p_0^{'}(1) = p_1^{'}(1)$ og $p_1^{'}(2) = p_2^{'}(2)$.

De tre polynombitene danner sammen en kvadratisk splinefunksjon, faktisk det som kaller en B-spline basisfunksjon, som vi kommer tilbake til i avsnitt 7.3.3 nedenfor. Her er også funksjonene tegnet opp og regnet ut ved rekursjonsformelen for B-splines.

Vi studerte Bezier kurver i avsnitt 4.10 og så i figur 4.10 at vi får noen tilleggsbetingelser hvis vi skal skjøte sammen Bezierkurver og ha deriverbarhet i skjøtene.

7.3 B-spline definisjon

Vi skal først definere B-splines og se på noen regneeksempler.

7.3.1 B-spline definisjon

B-splines kan defineres rekursivt (se [2]). For $d \geq 1$ gjelder

$$B_{i,d}(x) = w_{i,d}(x) \cdot B_{i,d-1}(x) + (1 - w_{i+1,d}(x)) \cdot B_{i+1,d-1}(x)$$
$$\textbf{hvor } w_{i,d}(x) = \begin{cases} \frac{x-t_i}{t_{i+d}-t_i} & t_i < t_{i+d} \\ 0 & ellers \end{cases} \tag{7.1}$$
$$\textbf{og } B_{i,0} = \begin{cases} 1 & t_i \leq x < t_{i+1} \\ 0 & ellers \end{cases}$$

t_i kalles skjøter, og $\mathbf{t} = \{t_0, t_1, t_2, \ldots\}$ kalles en skjøtvektor. Skjøtene er markert med grønt på figur 7.2.

7.3.2 Lineær B-spline, d=1

Vi får

$$B_{i,1}(x) = w_{i,1}(x) \cdot B_{i,0}(x) + (1 - w_{i+1,1}(x)) \cdot B_{i+1,0}(x)$$
$$= \frac{x - t_i}{t_{i+1} - t_i} \cdot B_{i,0}(x) + \left(1 - \frac{x - t_{i+1}}{t_{i+2} - t_{i+1}}(x)\right) \cdot B_{i+1,0}(x)$$

La oss sette $t_0 = 0, t_1 = 1, t_2 = 2, \ldots$ og $i = 0$ og regne ut den første lineære B-spline basisfunksjonen

$$B_{0,1}(x) = w_{0,1}(x) \cdot B_{0,0}(x) + (1 - w_{1,1}(x)) \cdot B_{1,0}(x)$$
$$= \frac{x - t_0}{t_1 - t_0} \cdot B_{0,0}(x) + \left(1 - \frac{x - t_1}{t_2 - t_1}\right) \cdot B_{1,0}(x)$$
$$= x \cdot B_{0,0}(x) + (1 - (x - 1)) \cdot B_{1,0}(x)$$
$$= x \cdot B_{0,0}(x) + (2 - x) \cdot B_{1,0}(x)$$

Hva slags funksjon er dette? Her må vi bruke siste linje i B-spline definisjonen (7.1). Vi ser at $B_{0,0}$ er definert for $0 \leq x < 1$ og at $B_{1,0}$ er definert for $1 \leq x < 2$. Det betyr at funksjonen vi har funnet, er en funksjon satt sammen av $y = x$ for $0 \leq x < 1$ og $y = 2 - x$ for $1 \leq x < 2$. figur 7.2 viser noen lineære B-splines på en uniform skjøtvektor.

Øving

Bestem uttrykkene for $B_{i,1}(x),\ i = 1, \ldots, 4$ på figur 7.2.

7.3.3 Kvadratisk B-spline, d=2

Vi har

$$B_{1,1}(x) = (x-1)B_{1,0}(x) + (3-x)B_{2,0}(x)$$

og kan nå regne ut $B_{0,2}$ ved hjelp av definisjonen.

$$\begin{aligned}
B_{0,2}(x) &= w_{0,2}(x) \cdot B_{0,1}(x) + (1 - w_{1,2}(x)) \cdot B_{1,1}(x) \\
&= \frac{x - t_0}{t_2 - t_0} B_{0,1}(x) + \left(1 - \frac{x - t_1}{t_3 - t_1}\right) B_{1,1}(x) \\
&= \frac{x}{2} B_{0,1}(x) + \frac{3-x}{2} B_{1,1}(x) \\
&= \frac{x}{2} \left(x \cdot B_{0,0}(x) + (2-x) \cdot B_{1,0}(x)\right) + \frac{3-x}{2} \left((x-1)B_{1,0}(x) + (3-x)\right) B_{2,0}(x) \\
&= \frac{x^2}{2} B_{0,0} + \frac{x(2-x)}{2} B_{1,0} + \frac{(3-x)(x-1)}{2} B_{1,0} + \frac{(3-x)^2}{2} B_{2,0} \\
&= \frac{x^2}{2} B_{0,0} + (-x^2 + 3x - \frac{3}{2}) B_{1,0} + \frac{(3-x)^2}{2} B_{2,0}
\end{aligned}$$

Hva har vi egentlig her? Når vi bruker definisjonsområdet for $B_{i,0}$, ser vi at vi har en andregradsfunksjon på intervallet $[0, 1)$, en annen andregradsfunksjon på $[1, 2)$ og en tredje på $[2, 3)$. Se figur 7.3.

7.3.4 B-splines og Bezier

Bezierkurver er et spesialtilfelle av B-splines. Se avsnitt 4.10 og spesielt figur 4.8. Hvordan kan vi lage tilsvarende funksjoner med B-splines? La oss starte med å lage lineære B-spline basisfunksjoner som på figur 4.6.

Vi velger skjøter $t_0 = 0, t_1 = 0, t_2 = 1, t_3 = 1$. Hvordan finner vi $B_{0,0}$? Vi ser av definisjonen (7.1) at

$$B_{0,0} = \begin{cases} 1 & t_0 \leq x < t_1 \\ 0 & ellers \end{cases}$$

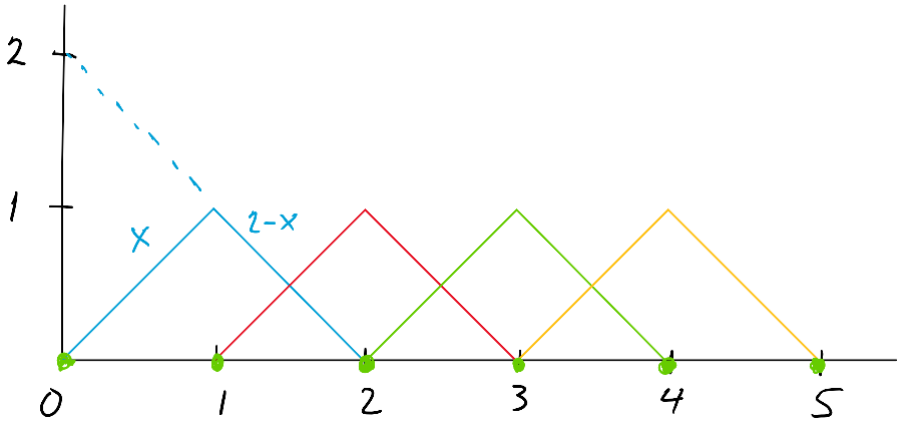
Første del (øverste linje) blir umulig å oppfylle når vi har multiple skjøter $t_0 = t_1$, derfor blir $B_{0,0} = 0$. Videre får vi

$$B_{0,1} = w_{0,1}B_{0,0} + (1 - w_{1,1})B_{1,0} = 0 + (1 - x) \cdot 1 = 1 - x$$

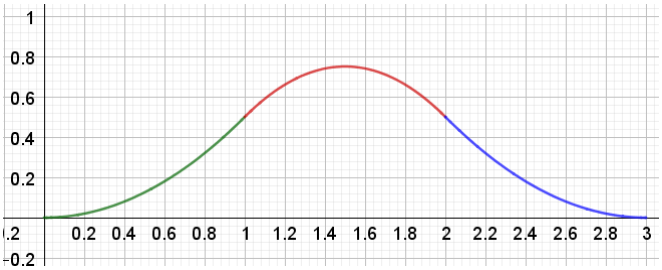
og

$$B_{1,1} = w_{1,1}B_{1,0} + (1 - w_{2,1})B_{2,0} = \frac{x - t_1}{t_2 - t_1} \cdot 1 + 0 = x$$

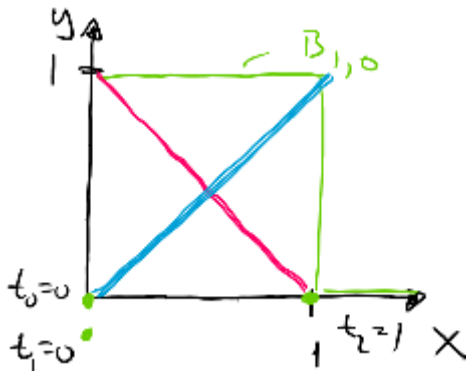
$B_{2,0} = 0$ av samme grunn som at $B_{0,0} = 0$.



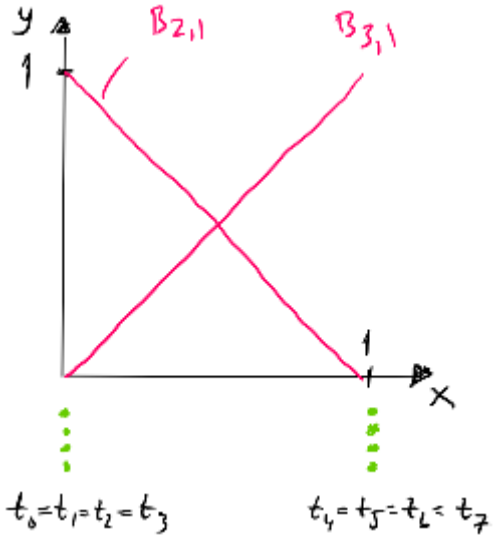
Figur 7.2: Lineære B-spline basisfunksjoner på uniform skjøtvektor.



Figur 7.3: Kvadratisk B-spline $B_{0,2}(x)$ med $t_0 = 0,\ t_1 = 1,\ t_2 = 2,\ t_3 = 3$.



Figur 7.4: $B_{0,1}$ og $B_{1,1}$ med $t_0 = 0,\ t_1 = 0,\ t_2 = 1,\ t_3 = 1$.



Figur 7.5: $B_{2,1}$ og $B_{3,1}$ med $t_0 = t_1 = t_2 = t_3 = 0,\ t_4 = t_5 = t_6 = t_7 = 1$.

7.3.5 Øving

La $t_0 = t_1 = t_2 = t_3 = 0$ og $t_4 = t_5 = t_6 = t_7 = 1$. Bruk rekursjonsformelen (7.1) til å regne ut

1. $B_i, 0,\ i = 0, \ldots, 7$
2. $B_i, 1,\ i = 0, \ldots$
3. $B_i, 2,\ i = 0, \ldots$
4. $B_i, 3,\ i = 0, \ldots$

På figur 7.5 er skjøtene markert med prikker, og vi finner ved samme argument som ovenfor at $B_{0,0} = B_{1,0} = B_{2,0} = 0$ og $B_{4,0} = B_{5,0} = B_{6,0} = 0$, slik at kun $B_{3,0} = 1,\ 0 \leq x < 1$.

Lineære B-spline basisfunksjoner

Dette forplanter seg til de lineære basisfunksjonene. Kun to av dem blir noe annet enn 0:

$$B_{2,1} = w_{2,1}B_{2,0} + (1 - w_{3,1})B_{3,0} = \frac{x - 0}{1} \cdot 0 + (1 - \frac{x - t_3}{t_5 - t_3}) \cdot 1 = 1 - x$$

$$B_{3,1} = w_{3,1} \cdot B_{3,0} + 0 = x$$

$B_{1,2}$ går over t_1, t_2, t_3, t_4:

$$B_{1,2} = w_{1,2}B_{1,1} + (1 - w_{2,2})B_{2,1} = 0 + 1 - \frac{x - t_2}{t_4 - t_2}(1 - x) = (1 - x)^2$$

Kvadratiske B-spline basisfunksjoner

Videre

$$\begin{aligned} B_{2,2} &= w_{2,2}B_{2,1} + (1 - w_{3,2})B_{3,1} \\ &= \frac{x-0}{1}(1-x) + \left(1 - \frac{x-0}{1}\right)x \\ &= x(1-x) + x(1-x) = 2x(1-x) \end{aligned}$$

$B_{3,2}$ går over t_3, t_4, t_5, t_6:

$$B_{3,2} = w_{3,2}B_{3,1} + (1 - w_{4,2})B_{4,1} = \frac{x-0}{1} \cdot x + 0 = x^2$$

Vi har her tre 2.gradsfunksjoner, definert på hvert sitt intervall. Disse intervallene er bestemt av $B_{i,0}$.

En kvadratisk B-spline basisfunksjon er C^1 kontinuerlig

En kvadratisk B-spline basisfunksjon er C^1 kontinuerlig. Ved å derivere ser vi at denne funksjonen har kontinuerlig derivert, siden den er deriverbar i skjøtene.

$$\begin{aligned} \frac{d}{dx}(\frac{x^2}{2}) &= x \\ \frac{d}{dx}(-x^2 + 3x - \frac{3}{2}) &= -2x + 3 \\ \frac{d}{dx}\left(\frac{(3-x)^2}{2}\right) &= x - 3 \end{aligned}$$

Når vi setter inn x=1 i de to første uttrykkene, får vi at den deriverte er 1. Når vi setter inn x=2 i de to siste uttrykkene, får vi at den deriverte er -1.

Kubiske B-spline basisfunksjoner

Vi fortsetter med skjøtvektoren på figur 7.5:

$$B_{0,3}(x) = w_{0,3}B_{0,2} + (1 - w_{1,3})B_{1,2} = \frac{x}{1} \cdot 0 + \left(1 - \frac{x}{1}\right)(1-x)^2 = (1-x)^3$$

$$\begin{aligned} B_{1,3}(x) &= w_{1,3}B_{1,2} + (1 - w_{2,3})B_{2,2} \\ &= x(1-x)^2 + (1-x) \cdot 2x(1-x) \\ &= 3x(1-x)^2 \end{aligned}$$

$$\begin{aligned} B_{2,3}(x) &= w_{2,3}B_{2,2} + (1 - w_{3,3})B_{3,2} \\ &= x \cdot 2x(1-x) + (1-x) \cdot x^2 \\ &= 3x^2(1-x) \end{aligned}$$

$$B_{3,3}(x) = w_{3,3}B_{3,2} + (1 - w_{4,3})B_{4,2} = x \cdot x^2 + 0 = x^3$$

7.4 Bezier/B-splines

På intervallet [0, 1] har vi regnet ut tredjegrads basisfunksjoner på tre forskjellige måter:

- Ved kubisk Hermite interpolasjon i avsnitt 4.9.4
- Ved å ekspandere $(1-t+t)^3$ som på figur 4.8
- Ved å bruke Cox-deBoor rekursjonsformel, (7.1) på skjøtvektor $\{0,0,0,0,1,1,1,1\}$
- Bezier: [0, 1]
- B-splines: Ingen slik restriksjon.

7.5 Spline funksjon

En splinefunksjon av *en* variabel x lages ved å kombinere B-spline basisfunksjoner. Gitt en grad d, n kontrollpunkter og en skjøtvektor $\mathbf{t} = \{t_0, \ldots t_{n+d}\}$. Da er

$$f(x) = \Sigma_{i=0}^{n-1} c_i B_{i,d}(x) \tag{7.2}$$

Det er et begrenset antall B-spline basisfunksjoner som er større enn 0 for en gitt x. Når vi har funnet hvilket intervall på skjøtvektoren x ligger i, $t_\mu \leq x < t_{\mu+1}$, reduseres summen ovenfor til

$$f(x) = \Sigma_{i=\mu-d}^{\mu} c_i B_{i,d}(x) \tag{7.3}$$

Vi kan lage en kvadratisk splinefunksjon som interpolerer endepunktene med grad $d = 2$ og $d+1 = 3$ multiple skjøter i hver ende. Vi kan regne ut funksjonsverdier med denne formelen. Vanligvis foretrekker man en annen algoritme, men det er nyttig for forståelsen å bruke denne formelen et par ganger. Gjør først oppgave 7.8.3.

7.5.1 Eksempel, kontrollpolygon

Vi har punktene (0,1), (1,2), (2,1), (3,2) og (4,1). Tegn opp, og trekk linjer mellom to og to punkter. Dette er opplagt en stykkevis lineær funksjon. Hvordan blir B-spline representasjonen?

Andrekoordinatene (1, 2, 1, 2, 1) svarer til c_i-ene i likning (7.2). Når vi skal representere dette ved B-splines, blir den en kombinasjon av lineære B-splines. Vi oppnår interpolasjon i endene ved å bruke d+1 like skjøter, så vi ha to like skjøter i hver ende. Skjøtvektoren blir da $\{0,0,1,2,3,4,4\}$. Hvordan blir sammenhengen mellom skjøtvektoren og førstekoordinaten til punktene? Formelen (7.4) gir denne sammenhengen, slik at de førstekoordinatene $x_i = t_i^*$ med d=1: $t_0^* = t_1 = 0,\ t_1^* = t_2 = 1,\ \ldots,\ t_4^* = t_5 = 4$.

7.5.2 Eksempel

La fortsatt $d = 2$ og $\mathbf{t} = \{0, 0, 0, 1, 2, 2, 2\}$. La $(0, 1)$, $(\frac{1}{2}, 2)$ $(\frac{3}{2}, 1)$ og $(2, 2)$ være $n = 4$ kontrollpunkter. Da blir verdiene i likning (7.2) $\mathbf{c} = \{1, 2, 1, 2\}$. Førstekoordinatene t_i^* er beregnet ved

$$t_i^* = \frac{t_{i+1} + \ldots + t_{i+d}}{d} \tag{7.4}$$

Her blir disse verdiene $t_0^* = \frac{t_1+t_2}{2} = \frac{0+0}{2} = 0$, $t_1^* = \frac{t_2+t_3}{2} = \frac{0+1}{2} = \frac{1}{2}$

$t_2^* = \frac{t_3+t_4}{2} = \frac{1+2}{2} = \frac{3}{2}$ og $t_3^* = \frac{t_4+t_5}{2} = \frac{2+2}{2} = 2$.

7.5.3 t_i^* for d=3

Gitt en skjøtvektor $\mathbf{t} = \{0, 0, 0, 0, 1, 1, 1, 1\}$, grad d=3 og n=4 kontrollpunkter. Førstekoordinatene som svarer til hvert kontrollpunkt blir da

$t_0^* = \frac{t_1+t_2+t_3}{3} = 0,\ t_1^* = \frac{t_2+t_3+t_4}{3} = \frac{1}{3},\ t_2^* = \frac{1}{3},\ t_3^* = 1.$

Vi kan sammenligne med de kubiske B-spline basisfunksjonene for denne skjøtvektoren, utregnet i avsnitt 7.3.5. Vi ser at hver t_i^* svarer til den parameterverdi hvor den tilsvarende basisfunksjonen B_i har sin største verdi.
($B_2'(t) = 0 \Leftrightarrow t = \frac{1}{3}$ og $B_3'(t) = 0 \Leftrightarrow t = \frac{2}{3}$).

7.5.4 Kvadratisk splinefunksjon, eksempel

La oss nå bruke likning (7.3) og regne ut noen funksjonsverdier. Vi har allerede regnet ut B-spline basisfunksjonene i oppgave 7.8.3

$$\begin{aligned}
f(0) &= 1 \\
f(\frac{1}{2}) &= 1 \cdot B_{0,2}(\frac{1}{2}) + 2 \cdot B_{1,2}(\frac{1}{2}) + 1 \cdot B_{2,2}(\frac{1}{2}) = 1 \cdot \frac{2}{8} + 2 \cdot \frac{5}{8} + 1 \cdot \frac{1}{8} = \frac{13}{8} \\
f(1) &= 2 \cdot B_{1,2}(1) + 1 \cdot B_{2,2}(1) + 2 \cdot B_{3,2}(1) = 2 \cdot \frac{1}{2} + 1 \cdot \frac{1}{2} + 0 = \frac{3}{2} \\
f(\frac{3}{2}) &= 2 \cdot B_{1,2}(\frac{3}{2}) + 1 \cdot B_{2,2}(\frac{3}{2}) + 2 \cdot B_{3,2}(\frac{3}{2}) = 2 \cdot \frac{1}{8} + 1 \cdot \frac{5}{8} + 2 \cdot \frac{2}{8} = \frac{11}{8} \\
f(2) &= 1
\end{aligned}$$

Dette er tilstrekkelig til å tegne en skisse av kontrollpolygonet og kurven. Du kan eventuelt regne ut noen flere punkter. Noen detaljer i utregningene er utelatt.

7.5.5 Kvadratisk splinefunksjon på komponentform

Vi har fra fasit 7.8.2

$$\begin{aligned}
B_{0,2}(x) &= (1-x)^2 B_{2,0} \\
B_{1,2}(x) &= (2x - \frac{3}{2}x^2)B_{2,0} + (2 - 2x + \frac{x^2}{2})B_{3,0} \\
B_{2,2}(x) &= \frac{x^2}{2}B_{2,0} + (-\frac{3}{2}x^2 + 4x - 2)B_{3,0} \\
B_{3,2}(x) &= (x-1)^2 B_{3,0}
\end{aligned}$$

Vi har sett at hver B-spline basisfunksjon er satt sammen av flere deler (se for eksempel figur 7.7). Vi skal nå skrive splinefunksjonen på komponentform - i stedet for å regne ut verdiene som ovenfor. Vi får i dette tilfellet

$$\begin{aligned} f(x) &= 1 \cdot (1-x)^2 B_{2,0} \\ &+ 2 \cdot \left((2x - \frac{3}{2}x^2) B_{2,0} + (2 - 2x + \frac{x^2}{2}) B_{3,0} \right) \\ &+ 1 \cdot \left(\frac{x^2}{2} B_{2,0} + (-\frac{3}{2}x^2 + 4x - 2) B_{3,0} \right) \\ &+ 2 \cdot (x-1)^2 B_{3,0} \\ &= (-\frac{3}{2}x^2 + 2x + 1) B_{2,0} + (\frac{3}{2}x^2 - 4x + 4) B_{3,0} \end{aligned}$$

Siste linje her viser *f(x)* som en sum, og skal tolkes som en delt funksjonsforskrift: Bruk første ledd for x<1 og andre ledd for x>1.

7.6 Spline kurve

Anta at c_i-ene i likning (7.2) er punkter $\mathbf{c_i} \in \mathbf{R}^2$ eller $\mathbf{c_i} \in \mathbf{R}^3$. Funksjonen $\mathbf{f}(t)$ har da to eller tre koordinater for hver parameterverdi t. Den kalles en vektorfunksjon og definerer en parametrisk kurve. En spline kurve er både en generalisering fra spline funksjon, og en generalisering fra Bezier kurve.

7.7 Oppgaver

7.7.1

La $t_0 = 0,\ t_1 = \frac{1}{2},\ t_2 = 1,\ t_3 = \frac{3}{2},\ t_4 = 2$. Bestem alle lineære og kvadratiske B-splines.

7.7.2

La $\mathbf{t} = \{0, 0, 0, 1, 2, 2, 2\}$. Bestem alle lineære og kvadratiske B-splines.

7.7.3

a) La $\mathbf{t} = \{0, 0, 0, 1, 2, 3, 3, 3\}$. Bestem alle lineære og kvadratiske B-splines.

b) La $\mathbf{c} \in \mathbf{R}^2$ være gitt ved $\mathbf{c} = \{(0, 1),\ (1, 0),\ (2, 1),\ (3, 0),\ (4, 1)\}$. Tegn splinekurven.

7.7.4

a) La $\mathbf{t} = \{0, 0, 0, 1, 1, 1\}$. Bestem alle lineære og kvadratiske B-splines.

b) La $\mathbf{c} \in \mathbf{R}^2$ være gitt ved $\mathbf{c} = \{(0, 1),\ (0, 0),\ (1, 0)$. Bestem den kvadratiske splinekurven med kontrollpunkter $\mathbf{c}$.

7.7.5

Gitt

- Lineære B-spline basisfunksjoner på uniform skjøtvektor som i figur 7.2
- $t_i = i$
- B-spline definisjon ligning (7.1)

Bestem alle B-spline basisfunksjoner av grad $d \leq 3$.

7.8 Fasit

7.8.1

Vi har 5 skjøter, og hver B-spline basisfunksjon av grad d går over $d+1$ skjøter. Vi får tre lineære, to kvadratiske og en kubisk B-spline basisfunksjon. De lineære basisfunksjonene kan vi finne ved å bruke rekursjonsformelen eller ved å tegne opp og lese av grafisk. Vi får

Lineære basisfunksjoner

$B_{0,1} = 2xB_{0,0} + (2-2x)B_{1,0}$
$B_{1,1} = (2x-1)B_{1,0} + (3-2x)B_{2,0}$
$B_{2,1} = (2x-2)B_{2,0} + (4-2x)B_{3,0}$

Notasjonen med de konstante b-spline basisfunksjonene brukes her til å angi hvilket intervall funksjonens første og andre del er definert over. $B_{i,0}$ notasjonen betyr at funksjonen er derfinert på det halvåpne intervallet $[t_i, t_{i+1})$. Vi ser for eksempel at $B_{1,0}$ inngår i både første og andre lineære funksjon. På intervallet $[t_1, t_2) = [\frac{1}{2}, 1)$ fins to lineære b-spline basisfunksjoner.

Kvadratiske basisfunksjoner

$B_{0,2} = w_{0,2}B_{0,1} + (1-w_{1,2})B_{1,1} =$
$x\ (2xB_{0,0} + (2-2x)B_{1,0})\ + (\frac{3}{2} - x)\ ((2x-1)B_{1,0} + (3-2x)B_{2,0})\ =$
$2x^2B_{0,0} + (2x-2x^2)B_{1,0} + (-2x^2+4x-\frac{3}{2})B_{1,0} + (2x^2-6x+\frac{9}{2})B_{2,0} =$
$2x^2B_{0,0} + (-4x^2+6x-\frac{3}{2})B_{1,0} + (2x^2-6x+\frac{9}{2})B_{2,0}.$

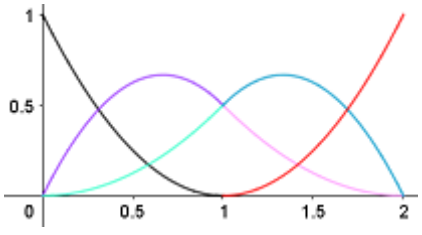
$B_{1,2} = w_{1,2}B_{1,1} + (1-w_{2,2})B_{2,1} =$
$(x-\frac{1}{2})\ ((2x-1)B_{1,0} + (3-2x)B_{2,0})\ + (2-x)\ ((2x-2)B_{2,0} + (4-2x)B_{3,0}\ ) =$
$(2x^2 - 2x + \frac{1}{2})B_{1,0} + (-4x^2+10x-\frac{11}{2})B_{2,0} + (2x^2-8x+8)B_{3,0}.$

7.8.2

$B_{0,0} = B_{1,0} = 0$
$B_{2,0} = 1,\ 0 \leq t < 1$
$B_{3,0} = 1,\ 1 \leq t < 2$
$B_{4,0} = B_{5,0} = 0$

$B_{0,1} = 0$
$B_{1,1} = 0 + (1-x)B_{2,0}$
$B_{2,1} = xB_{2,0} + (2-x)B_{3,0}$
$B_{3,1} = (x-1)B_{3,0} + 0$

$$
\begin{aligned}
B_{0,2}(x) &= 0 + (1 - w_{1,2})B_{1,1} = (1 - \frac{x-0}{1-0})(1-x)B_{2,0} = (1-x)^2 B_{2,0} \\
\\
B_{1,2}(x) &= w_{1,2}B_{1,1} + (1 - w_{2,2})B_{2,1} \\
&= \frac{x-t_1}{t_3-t_1}B_{1,1} + (1 - \frac{x-t_2}{t_4-t_2})B_{2,1} \\
&= \frac{x-0}{1-0}B_{1,1} + (1 - \frac{x-0}{2-0})B_{2,1} \\
&= xB_{1,1} + (1 - \frac{x}{2})B_{2,1} \\
&= x(1-x)B_{2,0} + (1 - \frac{x}{2})\,(xB_{2,0} + (2-x)B_{3,0}) \\
&= (x - x^2)B_{2,0} + (x - \frac{x^2}{2})B_{2,0} + (1 - \frac{x}{2})(2-x)B_{3,0} \\
&= (x - x^2)B_{2,0} + (x - \frac{x^2}{2})B_{2,0} + (1 - \frac{x}{2})(2-x)B_{3,0} \\
&= (2x - \frac{3}{2}x^2)B_{2,0} + (2 - 2x + \frac{x^2}{2})B_{3,0} \\
\\
B_{2,2}(x) &= w_{2,2}B_{2,1} + (1 - w_{3,2})B_{3,1} \\
&= \frac{x-t_2}{t_4-t_2}B_{2,1} + (1 - \frac{x-t_3}{t_5-t_3})B_{3,1} \\
&= \frac{x}{2}B_{2,1} + (2-x)B_{3,1} \\
&= \frac{x}{2}(xB_{2,0} + (2-x)B_{3,0}) + (2-x)(x-1)B_{3,0} \\
&= \frac{x^2}{2}B_{2,0} + (x - \frac{x^2}{2})B_{3,0} + (-x^2 + 3x - 2)B_{3,0} \\
&= \frac{x^2}{2}B_{2,0} + (-\frac{3}{2}x^2 + 4x - 2)B_{3,0} \\
\\
B_{3,2}(x) &= w_{3,2}B_{3,1} = \frac{x-t_3}{t_5-t_3}B_{3,1} = (x-1)^2 B_{3,0}
\end{aligned}
$$



Figur 7.6: Kvadratiske B-spline basisfunksjoner $B_{i,2}(x)$ med $\mathbf{t} = \{0, 0, 0, 1, 2, 2, 2\}$.

7.8.3

Vi har d=2, $\mathbf{t} = \{0, 0, 0, 1, 2, 3, 3, 3\}$ og $\mathbf{c} = \{(0, 1),\ (1, 0),\ (2, 1),\ (3, 0),\ (4, 1)\}$.

Vi har grad 2 og 3 skjøter i hver ende, så de første B-spline basisfunksjonene blir som i .

$B_{0,1}(x) = 0$
$B_{1,1}(x) = 0 + (1 - x)B_{2,0}$
$B_{2,1}(x) = xB_{2,0} + (2 - x)B_{3,0}$
$B_{3,1}(x) = (x - 1)B_{3,0} + (3 - t)B_{4,0}$
$B_{4,1}(x) = (x - 2)B_{4,0} + 0$
$B_{5,1}(x) = 0$

Videre trenger vi for å regne ut de kvadratiske B-spline basisfunksjonene

$w_{1,2}(x) = \frac{x-t_1}{t_3-t_1} = \frac{x-0}{1} = x$
$w_{2,2}(x) = \frac{x-t_2}{t_4-t_2} = \frac{x-0}{2} = \frac{x}{2}$
$w_{3,2}(x) = \frac{x-t_3}{t_5-t_3} = \frac{x-1}{2}$
$w_{4,2}(x) = \frac{x-t_4}{t_6-t_4} = x - 2$

Siden de fem første skjøtene er som i , blir de to første kvadratiske basisfunksjonene de samme:

$$\begin{aligned} B_{0,2}(x) &= (1 - x)^2 B_{2,0} \\ B_{1,2}(x) &= (2x - \frac{3}{2}x^2)B_{2,0} + (2 - 2x + \frac{x^2}{2})B_{3,0} \end{aligned}$$

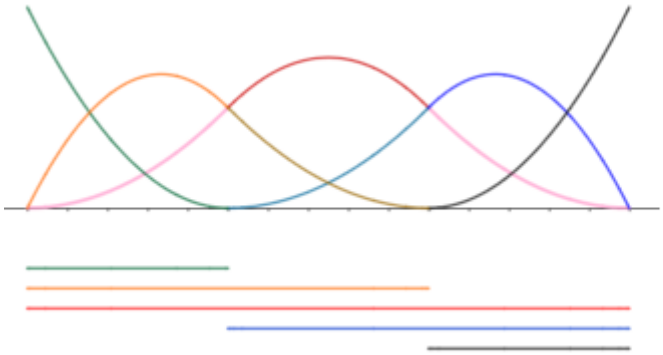
Videre

$$\begin{aligned} B_{2,2}(x) &= (2x - \frac{3}{2}x^2)B_{2,0} + (2x - \frac{x^2}{2} - \frac{3}{2})B_{3,0} \\ B_{3,2}(x) &= \frac{x-1}{2}B_{3,1} + (1 - \frac{x-2}{1})B_{4,1} \\ &= \frac{(x-1)^2}{2}B_{3,0} + (2x - \frac{x^2}{2} - \frac{3}{2})B_{4,0} + (5x - x^2 - 6)B_{4,0} \\ &= \frac{(x-1)^2}{2}B_{3,0} + (7x - \frac{3}{2}x^2 - \frac{15}{2})B_{4,0} \\ B_{4,2}(x) &= (x - 2)^2 B_{4,0} \end{aligned}$$

7.8.4

a) La $\mathbf{t} = \{0, 0, 0, 1, 1, 1\}$. Bestem alle lineære og kvadratiske B-splines.

b) La $\mathbf{c} \in \mathbf{R}^2$ være gitt ved $\mathbf{c} = \{(0, 1),\ (0, 0),\ (1, 0)$. Tegn splinekurven.



Figur 7.7: Kvadratiske B-spline basisfunksjoner $B_{0,2}(x)$ med $\mathbf{t} = \{0, 0, 0, 1, 2, 3, 3, 3\}$.

$$
\begin{aligned}
B_{0,1}(t) &= 0 \\
B_{1,1}(t) &= 1 - t \\
B_{2,1}(t) &= t \\
B_{3,1}(t) &= 0 \\
B_{0,2}(t) &= (1-t)^2 \\
B_{1,2}(t) &= 2t(1-t) \\
B_{2,2}(t) &= t^2 \\
\mathbf{p}(t) &= \mathbf{c}_0 B_{0,2}(t) + \mathbf{c}_1 B_{1,2}(t) + \mathbf{c}_2 B_{2,2}(t) \\
\mathbf{p}(t) &= \mathbf{c}_0 (1-t)^2 + \mathbf{c}_1 2t(1-t) + \mathbf{c}_2 t^2
\end{aligned}
$$

Dette er en parametrisk kurve. For hver t-verdi regner vi ut både x(t) og y(t) ved å bruke x-koordinatene og y-koordinatene i $\mathbf{c}$.
For eksempel er for $t = \frac{1}{2}$ $x = 0 \cdot (1 - \frac{1}{2})^2 + 0 \cdot 2 \cdot \frac{1}{2} \cdot (1 - \frac{1}{2}) + 1 \cdot (\frac{1}{2})^2 = \frac{1}{4}$
Her får vi

$$
\begin{aligned}
x(t) &= 0 \cdot (1-t)^2 + 0 \cdot 2 \cdot t \cdot (1-t) + 1 \cdot t^2 = t^2 \\
y(t) &= 1 \cdot (1-t)^2 + 0 \cdot 2 \cdot t \cdot (1-t) + 0 \cdot t^2 = (1-t)^2
\end{aligned}
$$

7.8.5

Vi starter med å sette opp uttrykkene for $w_{i,d}(x)$ og $1 - w_{i,d}(x)$:

$$
\begin{aligned}
w_{i,d}(x) &= \frac{x - t_i}{t_{i+d} - t_i} = \frac{x - i}{d} \\
1 - w_{i,d}(x) &= 1 - \frac{x - t_i}{t_{i+d} - t_i} = \frac{i + d - x}{d}
\end{aligned}
$$

Med $\mathbf{t} = \{0, 1, 2, 3, 4, 5\}$ får vi

$$\begin{aligned} B_{0,1} &= (x-0)B_{0,0} + (2-x)B_{1,0} \\ B_{1,1} &= (x-1)B_{1,0} + (3-x)B_{2,0} \\ B_{2,1} &= (x-2)B_{2,0} + (4-x)B_{3,0} \\ B_{3,1} &= (x-3)B_{3,0} + (5-x)B_{4,0} \end{aligned}$$

Videre

$$\begin{aligned} B_{0,2} &= \frac{x-0}{2}B_{0,1} + \frac{3-x}{2}B_{1,1} \\ &= \frac{1}{2}(x-0)^2 B_{0,0} + \frac{1}{2}(x-0)(2-x)B_{1,0} + \frac{1}{2}(3-x)(x-1)B_{1,0} + \frac{1}{2}(3-x)^2 B_{2,0} \\ &= \frac{1}{2}(x-0)^2 B_{0,0} + (-x^2 + 3x - \frac{3}{2})B_{1,0} + \frac{1}{2}(3-x)^2 B_{2,0} \\ B_{1,2} &= \frac{x-1}{2}B_{1,1} + \frac{4-x}{2}B_{2,1} \\ &= \frac{1}{2}(x-1)^2 B_{1,0} + \frac{1}{2}(x-1)(3-x)B_{2,0} + \frac{1}{2}(4-x)(x-2)B_{2,0} + \frac{1}{2}(4-x)^2 B_{3,0} \\ &= \frac{1}{2}(x-1)^2 B_{1,0} + (-x^2 + 5x - \frac{11}{2})B_{2,0} + \frac{1}{2}(4-x)^2 B_{3,0} \\ B_{2,2} &= \frac{x-2}{2}B_{2,1} + \frac{5-x}{2}B_{3,1} \\ &= \frac{1}{2}(x-2)^2 B_{2,0} + \frac{1}{2}(x-2)(4-x)B_{3,0} + \frac{1}{2}(5-x)(x-3)B_{3,0} + \frac{1}{2}(5-x)^2 B_{4,0} \\ &= \frac{1}{2}(x-2)^2 B_{2,0} + (-x^2 + 7x - \frac{23}{2})B_{3,0} + \frac{1}{2}(5-x)^2 B_{4,0} \end{aligned}$$

og

$$\begin{aligned} B_{0,3} &= \frac{x-0}{3}B_{0,2} + \frac{4-x}{3}B_{1,2} \\ B_{1,3} &= \frac{x-1}{3}B_{1,2} + \frac{5-x}{3}B_{2,2} \end{aligned}$$

Dette gir

$$\begin{aligned} B_{0,3} &= \frac{x-0}{3}B_{0,2} + \frac{4-x}{3}B_{1,2} \\ &= \frac{1}{3}(x-0)\left(\frac{1}{2}(x-0)^2 B_{0,0} + \frac{1}{2}(x-0)(2-x)B_{1,0} + \frac{1}{2}(3-x)(x-1)B_{1,0} + \frac{1}{2}(3-x)^2 B_{2,0}\right) \\ &+ \frac{1}{3}(4-x)\left(\frac{1}{2}(x-2)^2 B_{2,0} + \frac{1}{2}(x-2)(4-x)B_{3,0} + \frac{1}{2}(5-x)(x-3)B_{3,0} + \frac{1}{2}(5-x)^2 B_{4,0}\right) \\ &= \frac{x^3}{6}B_{0,0} + (-\frac{x^3}{2} + 2x^2 - 2x + \frac{2}{3})B_{1,0} \\ &+ (\frac{x^3}{2} - 4x^2 + 10x - \frac{22}{3})B_{2,0} + (-\frac{1}{6}x^3 + 2x^2 - 8x + \frac{32}{3})B_{3,0} \end{aligned}$$

Tillegg A

Kode

A.1 Matrix4x4/Matrix4x4.h

Listing A.1: Matrix4x4/Matrix4x4.h

```
#ifndef MATRIX4x4_H
#define MATRIX4x4_H

#include <iostream>
#include <fstream>
#include <cmath>
#include "vector3d.h"

namespace gsml {

typedef std::pair<int, int> Dimension;
typedef Vector4d Vec4;

class Matrix4x4
{
public:
    Matrix4x4();
    void setToIdentity();
    void read(std::string filnavn);
    void print() const;
    void set(int i, int j, float x);
    void set_dim(int rader, int kolonner);
    Dimension get_dim() const;
    void operator =(const Matrix4x4& M);
    Matrix4x4 operator * (const Matrix4x4& M) const;
    float& operator () (int i, int j) {return A[i][j]; }

    Matrix4x4 inverse() const;
    Matrix4x4 transpose() const;
    void LU();
    float determinant() const;
    // Disse tre endres/flyttes
    Vec4 solve(Vec4& b) const;
    void set(int j, Vec4 &v);
    Vec4 operator * (Vec4 &v) const;

    void translate(float tx, float ty, float tz);
    void rotate(float degrees, float rx, float ry, float rz);
    void scale(float sx, float sy, float sz);
```

```
    void frustum(float left, float right, float bottom, float top, float near, float far);

    void perspective(float fovy, float aspectRatio, float nearPlane, float farPlane);

    void lookAt(const Vector3d& eye, const Vector3d& at, const Vector3d& up);
    const float * constData() const;

protected:
    static const int M=4;
    static const int N=4;
    static const float EPS;
    float A[M][N];
    int m; // forelopig ingen sjekk
    int n; //
    bool lu_faktorisert;
    int permutasjon[M];

    void mult(const Matrix4x4 &M);
    void pivot(int k);
};

}
```

Bibliografi

[1] D. Angel, E. Shreiner. *Interactive Computer Graphics A Top-Down Approach with WebGL Seventh edition.* Pearson, 2015.

[2] C deBoor. *A Practical Guide to Splines.* Springer, 1978.

[3] D.E. Edwards, C.H. Penney. *Calculus and analytic geometry.* Prentice-Hall, 1982.

[4] R Goldman. *Pyramid Algorrithms.* Morgen Kaufman Elsevier, 2003.

[5] M.G.Gulbrandsen, J.Kleppe, T.A.Kro, J.E.Vatne. *Matematikk for ingeniørfag med numeriske beregninger.* Gyldendal akademisk, 2013.

[6] M.A.Weiss. *Data Structures and Algorithm Analysis i C++ Fourth Edition.* Pearson, 2014.

[7] L.L Schumaker. *Spline Functions: Basic Theory Third Edition.* Cambridge Mathematical Library, 2007.

[8] O. Dahl, T. Lyche, R. Winter. *Construction and analysis of numerical methods.* Universitetet i Oslo, 1985.