# ULTIMATE MATH PAPER EPIC STYLE

Adam Aske

20. mars 2022

## Innhold

# 1 Github

Link til min branch : https://github.com/Hedmark-University-College-SPIM/3Dprog22/tree/AdamA

# 2 Kappitel 2 Oppgaver

## 2.1 2.3.1

Finn de stasjonærepunktene til funksjonene

### 2.1.1 A

$$f(x,y) = xy \quad \frac{df}{dy} = y \quad \frac{df}{dx} = x \quad y = 0 \quad x = 0 \quad [0,0]$$

### 2.1.2 B

$$f(x,y) = x^2 + y^2 \quad \frac{df}{dx} = 2x \quad \frac{df}{dy} = 2y \quad 2y = 0 \quad 2x = 0 \quad [0,0]$$

### 2.1.3 C

$$f(x,y) = sin * xy \quad \frac{df}{dx} = ycos(xy) \quad \frac{df}{dy} = xcos(xy) \quad x = 0 \quad y = 0$$

### 2.1.4 D

$$f(x,y) = sinx{*}siny \quad \frac{df}{dx} = cos(x){*}sin(y) \quad \frac{df}{dy} = sin(x){*}cos(y) \quad f = 0 = cosx{*}siny$$

## 2.2 2.3.2

Regn ut $\int_3^6 \int_- 2^1 (x^2 + xy^2)\, dy, dx$

Regn ut $\int_{-2}^{1} \frac{x^3}{3} + y^2 \int_{-2}^{1} x\, dx = \frac{x^3}{3} + \frac{y^2x^2}{2} + C$

$\frac{1^3}{3} + \frac{y^2}{2} = \frac{1}{3} + \frac{y^2}{2}$

$\frac{-2^3}{3} + \frac{y^2x^{-2}}{2} = \frac{-8}{3} + 2y^2$

$\int_{-2}^{1} = (\frac{1}{3} + \frac{y^2}{2})^2 - (\frac{-8}{3} + 2y^2)^2$

$\frac{1}{3} + \frac{8}{3} + \frac{1}{2}y^2 - 2y^2 = \frac{9}{3} + \frac{1}{2}y^2 + 2y^2 = 3 - \frac{3}{2}y^2$

$\int_3^6 (3 - \frac{3}{2}y^2)\, dy = MANGELRNOEHER(3 * 6 - \frac{6^3}{2})^6 - (3 * 3 - \frac{3^3}{2})^3 = (18 - \frac{216}{2}) - (9 - \frac{27}{2}) = -\frac{171}{2}$

### 2.3 2.3.3

# 3 Oblig 1

## 3.1 Del 1

Jeg har valgt funksjonen; f(x, y) = sin(PI*x)*sin(PI*y). Omerådet $0 < x < 1$, $0 < y < 3$ og steg = 0.2. Funkjsonen tar inn en array og en størrelse. Først blir arrayen fylt med tilfeldige tall.

Listing 1: trianglesurface.cpp

```
//Create triangle
float xmin=0.0f, xmax=1.0f, ymin=0.0f, ymax=1.0f, h=0.1f;
for (auto x=xmin; x<xmax; x+=h)
{
        for (auto y=ymin; y<ymax; y+=h)
        {
              float z = func(x, y);
              mVertices.push_back(Vertex{x,y,z,x,y,z});
              z = func(x+h, y);
              mVertices.push_back(Vertex{x+h,y,z,x,y,z});
              z = func(x, y+h);
              mVertices.push_back(Vertex{x,y+h,z,x,y,z});
              mVertices.push_back(Vertex{x,y+h,z,x,y,z});
              z = func(x+h, y);
              mVertices.push_back(Vertex{x+h,y,z,x,y,z});
              z = func(x+h, y+h);
              mVertices.push_back(Vertex{x+h,y+h,z,x,y,z});
        }
}
```

Listing 2: trianglesurface.hh

```
static float func(float x, float y){
       //Matte oblig funksjon
       return pow(x, 3) * y;
   }
```

### 3.1.1 Lese og skrive til fil

Listing 3: trianglesurface.cpp

```
void TriangleSurface::readFile(std::string fileName) {
        std::ifstream inn;
       inn.open(fileName.c_str());
       if (inn.is_open())
       {
                int n;
             Vertex vertex;
             inn >> n;
             mVertices.reserve(n);
             for (int i=0; i<n; i++) {
                inn >> vertex;
                   mVertices.push_back(vertex);
              }
              inn.close();
          }
}
```

```
void TriangleSurface::writeFile(std::string fileName){
        std::ofstream wF;
       wF.open(fileName.c_str())
        if(wF.is_open())
        {
                wF << mVertices.size() << "\n";
            for (int i = 0; i < mVertices.size(); i++)
            {
                wF << mVertices[i] << "\n";
            }
        }
        else
        {
            std::cout << "Failed to write to file.\n";
        }
        wF.close();
}
```

### 3.2 Oblig 1 Del 2

### 3.3 A

Analytisk utregning for volumet av funksjonen. $\int_0^1 \int_0^1 x^3 * y\,dy, dx$
$\int_0^1 x^3 * y\,dy = x^3 \int y\,dy = x^3 * (y^2/2) = x^3y^2/2 = x^3 * 1^2/2 = x^3/2$
$\int x^3/2 = 1/2 \int x^3\,dx = 1/2 * x^4/4 = x^4/8$
$\int_0^1 \int_0^1 x^3 * y\,dy, dx = 1/2$

### 3.4 B

For å regne integralet numerisk lagde jeg en funksjon i trianglesurface.cpp og skriver resultatene til en fil. Funksjoner gjør det 4 ganger og halverer steg lengden for hver iterasjon. Resultatene blir lagret i Numerisk.txt.

Listing 4: trianglesurface.cpp

```
void TriangleSurface::CalculateNumerical(){
        std::ofstream file;
        file.open("Numerisk.txt");
        if(file.is_open())
        {
                float xmin= 0.0f, xmax = 1.0f, ymin = 0.0f, ymax  = 1.0f,h = 0.1f, result = 0
            for(int i = 0; i < 4; i++)
            {
                for(auto x = xmin; x < xmax; x+=h)
                {
                    for(auto y = ymin; y < ymax; y+= h)
                    {
                        float z = func(x, y) * pow(h, 2);
                        result += z;
                    }
                }
            h = h / 2;
            file << result << "\n";
            }
        }
        else
        {
            std::cout << "Failed to write to file.\n";
        }
```

```
        file.close();
}
```

Resultatene ble: h1 = 0.091125, h2 = 0.198297, h3 = 0.332908, h 4 =0.462652

### 3.5 Resultat

Den numeriske utregningen går nærmere og nærmere svaret jeg fikk fra manuell utergning; 1/2.

# 4 Oblig 2

## 4.1 Oppgave 3.4.6

Oppgave 3.4.6 Valgte punkter: (-6, 10), ( -5.9, 6.6), (-3, 4.8), (-3.1, 1.60), (0.1, 0.5), (2.6, 1.1), (3.8, 4.3), (6.7, 5.2)
Wolfram Aplha er brukt til matrise multiplikasjonene.
y = Ax + e

$$\begin{bmatrix} 10 \\ 6.6 \\ 4.8 \\ 1.6 \\ 0.5 \\ 1.1 \\ 4.3 \\ 5.2 \end{bmatrix} = \begin{bmatrix} 36 & -6 & 1 \\ 43.8 & -5.9 & 1 \\ 9 & -3 & 1 \\ 9.6 & -3.1 & 1 \\ 0 & 0.1 & 1 \\ 6.7 & 2.6 & 1 \\ 14.4 & 3.8 & 1 \\ 44.9 & 6.7 & 1 \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} + \begin{bmatrix} e_1 \\ e_2 \\ e_3 \\ e_4 \\ e_5 \\ e_6 \\ e_7 \end{bmatrix}$$

$$B = A^T * A$$

$$= \begin{bmatrix} 36 & 34.8 & 9 & 9.6 & 0 & 6.7 & 14.4 & 44.9 \\ -6 & -5.9 & -3 & -3.1 & 0.1 & 2.6 & 3.8 & 6.7 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} 36 & -6 & 1 \\ 43.8 & -5.9 & 1 \\ 9 & -3 & 1 \\ 9.6 & -3.1 & 1 \\ 0 & 0.1 & 1 \\ 6.7 & 2.6 & 1 \\ 14.4 & 3.8 & 1 \\ 44.9 & 6.7 & 1 \end{bmatrix} = \begin{bmatrix} 4948.5 & -97.07 & 155.4 \\ -105.11 & 158.6 & -4.8 \\ 155.4 & -3.6 & 8 \end{bmatrix}$$

$$C = A^T * y = \begin{bmatrix} 36 & -6 & 1 \\ 43.8 & -5.9 & 1 \\ 9 & -3 & 1 \\ 9.6 & -3.1 & 1 \\ 0 & 0.1 & 1 \\ 6.7 & 2.6 & 1 \\ 14.4 & 3.8 & 1 \\ 44.9 & 6.7 & 1 \end{bmatrix} \begin{bmatrix} 10 \\ 6.6 \\ 4.8 \\ 1.6 \\ 0.5 \\ 1.1 \\ 4.3 \\ 5.2 \end{bmatrix} = \begin{bmatrix} 951 \\ -64.2 \\ 34.1 \end{bmatrix}$$

$$B^{-1} = \begin{bmatrix} 0.0005 & 0 & -0.01 \\ 0 & 0.006 & 0.003 \\ -0.01 & 0.001 & 0.32 \end{bmatrix}$$

$$x = B^{-1} * c = \begin{bmatrix} 00.0005 & 0 & -0.01 \\ 0 & 0.006 & 0.003 \\ -0.01 & 0.001 & 0.32 \end{bmatrix} \begin{bmatrix} 951 \\ -64.2 \\ 34.1 \end{bmatrix} = \begin{bmatrix} 0.145 & \\ -0.268 & 1.309 \end{bmatrix}$$

$y = 0.145x^2 - 0.268x + 1.309$

## 4.2 Beregne punkter og lagre i array

Funksjonen tar inn x som verdi og bruker funksjonen fra utergningen og returnerer y verdien punktet skal ha.

Listing 5: trianglesurface.h

```
static float func2(float x) {
        return 0.174 * x + 1, 743;
    }
```

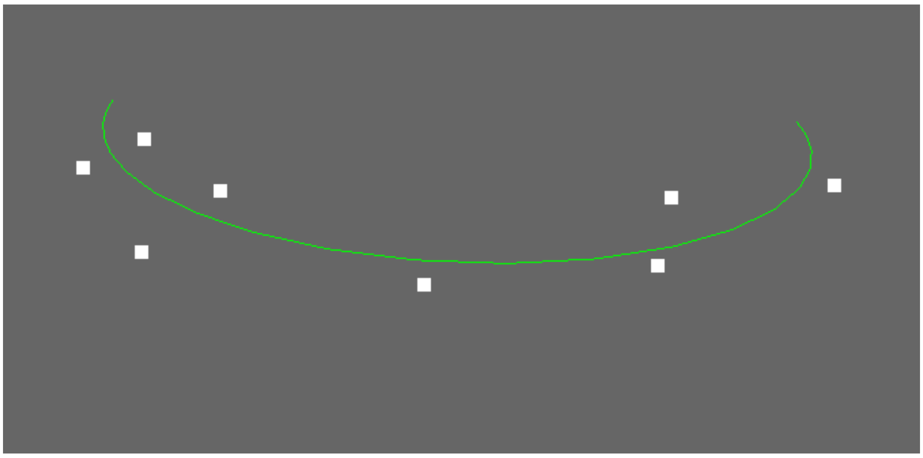
## 4.3 3.4.6 Visualisering

VisualPoint klassen tar inn en vector av Vertex'er, vertexene blir vist som hvite brikker på skjermen. MMap får en QuadraticPolynomial som tar inn 6.9, 1.3 og 3.2 fra minste kvadtraters metode, og blir vist som en grønn kurve på skjermen. De stemmer ikke med hverandre, noe er feil med utregningen.

Listing 6: renderwindow.cpp

```
mMap.insert(std::pair<std::string, VisualObject*>{"QuadtraticPolynomial",
new QuadtraticPolynomial(0.145f, -0.268f, 1.3f, 0.1f)});
     std::vector<Vertex> points;
     points.push_back(Vertex{ -6, 10, 0 });
     points.push_back(Vertex{ -5.9, 6.6, 0 });
     points.push_back(Vertex{ -3, 4.8, 0 });
     points.push_back(Vertex{ -3.1, 1.6, 0 });
     points.push_back(Vertex{ 0.1, 0.5, 0 });
     points.push_back(Vertex{ 2.6, 1.1, 0 });
     points.push_back(Vertex{ 3.8, 4.3, 0 });
     points.push_back(Vertex{ 6.7, 5.2, 0 });

     for (auto i = 0; i < points.size(); i++) {
         mMap.insert(std::pair<std::string, VisualObject*>
{ std::to_string(i) , new VisualPoint(points)});
     }
```

Den ser noe forvrengt ut, men det skyldes kamera sin rotasjon.



## 4.4 Oppgave 4.6.7

Punkter: ( 0.9, 0.6), ( 2.2, 2.6), (4.5, -1), (5.9, 1.6)

$$f(x) = ax^3 + bx^2 + cx + d$$
$$0.6 = a * 0.6^3 + b * 0.6^2 + c * 0.6 + d$$
$$2.6 = a * 2.6^3 + b * 2.6^2 + c * 2.6 + d$$
$$-1 = -a * 1^3 - b * 1^2 - c * 1 + d$$
$$1.6 = a * 1.6^3 + b * 1.6^2 + c * 1.6 + d$$

$$A = \begin{bmatrix} 0.729 & 0.81 & 0.9 & 1 \\ 10.648 & 4.84 & 2.2 & 1 \\ 91.125 & 20.25 & 4.5 & 1 \\ 205.379 & 34.81 & 5.9 & 1 \end{bmatrix} B = \begin{bmatrix} 0.6 \\ 2.6 \\ -1 \\ 1.6 \end{bmatrix}$$

$$A^{-1} = \begin{bmatrix} -0.04 & 0.09 & -0.09 & -0.4 \\ 0.5 & -1.02 & 0.77 & -0.29 \\ -2.11 & 3.25 & -1.7 & 0.6 \\ 2.5 & -2.15 & 1 & -0.34 \end{bmatrix}$$

$$x = A^{-1} * B =$$

$$\begin{bmatrix} -0.04 & 0.09 & -0.09 & -0.4 \\ 0.5 & -1.02 & 0.77 & -0.29 \\ -2.11 & 3.25 & -1.7 & 0.6 \\ 2.5 & -2.15 & 1 & -0.34 \end{bmatrix} \begin{bmatrix} 0.6 \\ 2.6 \\ -1 \\ 1.6 \end{bmatrix} = \begin{bmatrix} -0.34 \\ -3.586 \\ 9.844 \\ -5.634 \end{bmatrix} f(x) = -0.34x^3 - 3.586x^2 + 9.844x - 5.634$$

## 4.5 4.6.7 Visualisering

VisualPoint klassen tar inn en vector av Vertex'er, vertexene blir vist som hvite brikker på skjermen. MMap får en QuadraticPolynomial som tar inn 6.9, 1.3 og 3.2 fra minste kvadtraters metode, og blir vist som en grønn kurve på skjermen. De stemmer ikke med hverandre, noe er feil med utregningen.

Listing 7: renderwindow.cpp

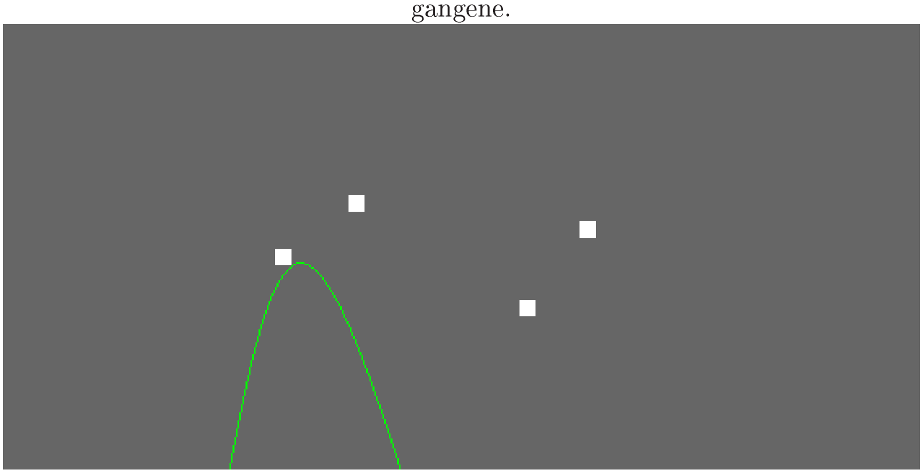
```
//Matte oblig 2 4.6.7
```

```
    mMap.insert(std::pair<std::string, VisualObject*>
        {"CubicPolynomial", new CubicPolynomial(-0.34, -3.586, 9.844, -5.634,0.1f)});
    std::vector<Vertex> points2;
    points2.push_back(Vertex{ 0.9f, 0.6f, 0 });
    points2.push_back(Vertex{ 2.2f, 2.6f, 0 });
    points2.push_back(Vertex{ 4.5f, -1.f, 0 });
    points2.push_back(Vertex{ 5.9f, 1.6, 0 });

    for (auto i = 0; i < points2.size(); i++) {
        mMap.insert(std::pair<std::string, VisualObject*>
        { std::to_string(i*10), new VisualPoint(points2)});
    }
```

Listing 8: cubicpolynomial.cpp

```
CubicPolynomial::CubicPolynomial(double a, double b, double c, double d, float dx)
{

    for (auto x = -10.f; x <= 10; x += 0.1)
    {
        auto y = p(a, b, c, d, x);
        mVertices.push_back(Vertex(x, y, 0, 0, 1, 0));
    }
    mMatrix.setToIdentity();

}

double CubicPolynomial::p(double a, double b, double c, double d, double x)
{
    return a * x * x * x + b * x * x + c * x + d;

}
```

Resultatet ser ikke riktig ut i mine øyne, men har omregnet matrisene flere gangene.



## 4.6 Oppgave 4.11.6

Bezier kurve.

Initialiseringen av Bezier kurven.

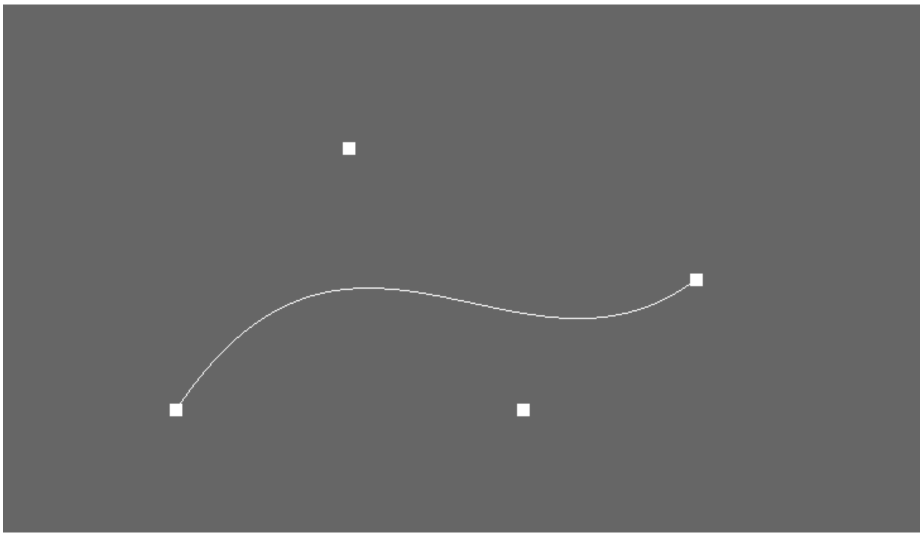
Listing 9: renderwindow.cpp

```
//Bezier curve
```

```
    std::vector<QVector3D> controlPoints;
    controlPoints.push_back(QVector3D(0.f, 0.f, 0.f));
    controlPoints.push_back(QVector3D(2.f, 3.f, 0.f));
    controlPoints.push_back(QVector3D(4.f, -3.f, 0.f));
    controlPoints.push_back(QVector3D(6.f, 3.f, 0.f));
    mMap.insert(std::pair<std::string, VisualObject*>
        {"BezierCurve", new BezierCurve(controlPoints)});
```

Listing 10: beziercurve.cpp

```
BezierCurve::BezierCurve(std::vector<QVector3D> controlPoints) {
    mControlPoints = controlPoints;
    //Create vertexs from control points
    for (auto it : mControlPoints) {
        mControlPointsVertices.push_back
        (Vertex(it.x(), it.y(), it.z(), 1.f, 1.f, 1.f));
    }
    //Visualpoint for displaying control points
    mControlPointVisual = new VisualPoint(mControlPointsVertices);

    for (float t{}; t < 1.00f; t += 0.01f) {
        QVector3D point = EvaluateBezier(t);

        mVertices.push_back(Vertex(point.x(), point.y(), point.z()));
    }
}
QVector3D BezierCurve::EvaluateBezier(float t)
{
    std::vector<QVector3D> temp;

    //Gets the control points
    for (int i = 0; i < mControlPoints.size(); i++) {
        temp.push_back(mControlPoints[i]);
    }
    for(int k = temp.size()-1; k > 0; k--)
    {
        for(int i = 0; i < k; i++)
            //Bezier algoritmen
            temp[i] = temp[i] * (1 - t) + temp[i + 1] * t;
    }
    return temp[0];
}
```



# 5 Kappitel 4

## 5.1 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 og polynomer.

## 5.2 Eksempler på interpolasjonav to, tre og fire punkter

Funksjonens graf skal gå gjennom punktene. Hvis vi har to punkter kan vi finne et førstegradspolynom(linært) som interpolere 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.

## 5.3 Eksempel - interpolere to punkter

Velg to punkter og konstruer et linært polynom sin interpolerer punktene. Her bruker vi punktene nedenfor.
f(x) = ax + b
1 = a * 1 + b
3 = a * 3 + b

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

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

$$x^T = [a, b]ogb^T = [1, 3].Lsningenblirx = A^{-1}b = [1/2, 1/2], altsf(x) = 1/2x + 1/2.$$

## 5.4 Eksempel - interpolere tre punkter

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

$f(x) = ax^2 + bx + c$
$1 = a * 1^2 + b * 1 + c$
$3 = a * 3^2 + b * 3 + c$
$1 = a * 5^2 + b * 5 + c$

Vi kan sette opp dette på matrise form Ax = b med A =

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

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

## 5.5 Oppgave 4.2.3

1. (1/2, 5/4), (2, -1), (4, 3) 2. Bestem interpolasjonspolynomet(andregradsfunksjonen). $f(x) = ax^2 + bx + c$

$5/4 = a * 1/2^2 + b * 1/2 + c$
$-1 = a * 2^2 + b * 2 + c$
$3 = a * 4^2 + b * 4 + c$

$$A = \begin{bmatrix} 0.25 & 1/2 & 1 \\ 4 & 2 & 1 \\ 16 & 4 & 1 \end{bmatrix} A^{-1} = \begin{bmatrix} 4/21 & -1/3 & 1/7 \\ -8/7 & 3/2 & -5/14 \\ 32/21 & -2/3 & 1/7 \end{bmatrix}$$

$$B = \begin{bmatrix} 5/4 \\ -1 \\ 3 \end{bmatrix} B^T = \begin{bmatrix} 5/4 & 1 & 3 \end{bmatrix}$$

$$x = A^{-1} * b$$

$$= \begin{bmatrix} 1 \\ -4 \\ 3 \end{bmatrix}$$

$$f(x) = x^2 - 4b + 3$$

## 5.6 Interpolasjon av to funksjonsverdier og to deriverte

Gitt punktene x0 ( -2, -15/2) og x3(3,0). I stedet for å å bruke interpolasjonsbetingelsene ytterligere i to punkter x1 og x2 for å sette opp et ligningsystem med fire lignner og fire ukjente, kan vi stille betingelser til de deriverte i x0 og x3. La oss kreve at den deriverte til interpolasjonspolynomet p i x0 = -2 og x3 = 3 skal være p'(-2) ) 23/2 og p'(3) ) 4. Vi har da følgende fire interpolasjons betlingelser:

1. p(-2) = -15/2
2. p(3) = 0
3. p'(-2) = 23/2
4. p'(3) = 4

Polynoet vi skal bestemme er på formen $p(x) = ax^3 + bx^2 + xc + d$, og når vi deriverer fir vi $p'(x) = 3ax^2 + 2bx + c$. x0 (-2, -15/2), x1(3,4) x2(-2,23/2) og x3(3, 0) Vi setter in og regner ut og får

-8a + 4b -2c + d = -15/2

27a + 9b + 3c + d = 0

12a - 4b + c =23/2

27a + 6b + c = 4

$Ax = b, x = A^{-1} * b$ Vi kan fortsette utregnigne på vanlig måte.

# 6 Oppgaver uke 6

## 6.1 4.6.1

## 6.2 4.6.2

## 6.3 4.6.4

## 6.4 4.6.6

# 7 Oppgaver uke 7

## 7.1 4.9.3

## 7.2 4.9.4

## 7.3 4.9.5

## 7.4 4.11.4

## 7.5 4.11.5

### 7.5.1 1

# 8 Oppgaver uke 8

## 8.1 1

Gitt kontrollpunkter (5,1), (1,2) og (3,4). Bruk deCasteljau's algoritme til å tegne opp den tilhørende kvadratiske Bezierkurven. Kontroller svaret ved utregning (bruk ligning 4.16).

## 8.2 2

Gitt kontrollpunkter (1,2), (3,4) og (3,1). Bruk deCasteljau's algoritme til å tegne opp den tilhørende kvadratiske Bezierkurven. Kontroller svaret ved utregning (bruk ligning 4.16).

### 8.3 3

Gitt kontrollpunkter (5,1), (1,2), (3,4) og (3,1). Bruk nå deCasteljau's algoritme til å tegne opp den tilhørende kvadratiske Bezierkurven. Kontroller svaret ved utregning (bruk ligning 4.16).

### 8.4 4

Gitt kontrollpunkter (5,1), (1,2) og (3,4). Bruk Neville's algoritme til å tegne opp den kvadratiske kurven som interpolerer kontrollpunktene. Kontroller svaret ved utregning som vist i 5.1.4.

### 8.5 5

Gitt kontrollpunkter (1,2), (3,4) og (3,1). Bruk Neville's algoritme til å tegne opp den kvadratiske kurven som interpolerer kontrollpunktene. Kontroller svaret ved utregning som vist i 5.1.4.

### 8.6 6

Gitt kontrollpunkter (5,1), (1,2), (3,4) og (3,1). Bruk nå Neville's algoritme til å tegne opp den kubiske kurven som interpolerer kontrollpunktene. Kontroller svaret ved utregning som vist i 5.1.4.

## 9 Oppgaver uke 10

### 9.1 1

Gitt en trekant med hjørner P(0,1), Q(1.5, 0) og R(2.5, 1) (som på figur 6.2 eller rett og slett trekant på figur 6.3). Regn ut de barysentriske koordinatene med hensyn på P, Q og R for punktene: $(\frac{1}{2}, \frac{1}{2})$

$(1, \frac{1}{2})$

$(2, \frac{1}{2})$

$(\frac{3}{2}, 2)$

$(\frac{3}{2}, \frac{3}{2})$

### 9.2 2

Oppgave 6.2.14

## 10 Oppgaver uke 11

Øvingsoppgaver til oblig 3

I tillegg til ukeoppgaver som har vært gitt, kan dere gjøre følgende tidligere eksamensoppgaver:

## 10.1 2018

### 10.1.1 1

La x0 = 0, x1 = 1, y0 = 0 og y1 = 1 og gitt funksjonen g(x) = sin $\pi/2 * x.Lap(x)vrepolynometforkubiskHermiteinterpolasjonavgpintervallet[0,1].a)Regnuty'0 = g'(0), y'1 = g'(1)ogsettoppinterpolasjonsporblemetpmatriseformAx = bhvorb^T = [y0, y1, y'0y'1]b)Brukresultatetfraa)tilbestemmeinterpolasjonspolynomettilp(x).$

### 10.1.2 2a-d

Gitt funksjonen f(x, y) = $e^{(x^2+y^2)}$, 0 x 1, 0 y 1. a)Regn ut f(i/2, j/2) hvor i = 0,1,2 og j = 0,1,2 b)Bestem de partiellderiverte f/dx(1/2, 1/2) og f/dy(1/2, 1/2). c)Bruk f(1/2, 1/2) og de partiellderiverte til å finne en normalvektor til f(x,y) i punktet (1/2, 1/2). Vi antar nå at vi kjenner de 9 punktene med funksjonserdier fra a), men at funksjonen er ukjent. d)Forklar hva slags funksjon z=f(x,y) vi da kan lage. Tegn figur.

### 10.1.3 3a

Gitt kontrollpunktene (1,1), (0,1), (0,0) og (1,0). Skisser en kubisk Bezier kurve med disse kontrollpunktene. Forklar og tegn hvordan du bruker deCasteljau algoritmen.

## 10.2 2019

### 10.2.1 2

I en annen spill-scene i xy-planet er fire trofeer/items plassert på posisjonene (0,1), (1,2), (2,0) og (4,2). En NPC skal patruljere langs et tredjegradspolynom som interpolerer disse punktene. a) Sett opp interpolasjonsproblemet på formen Ax=b hvor A er en 4x4 matrise og x og b er 4-dimensjonale vektorer. b) Bestem ligningen for tredjegradspolynomet som interpolerer punktene.

### 10.2.2 3

## 10.3 2020

### 10.3.1 1

### 10.3.2 3