Inna Williams

##

1.

(25 pts) Consider the function f(x) = $3x^5 + 2x - 1$ and the equation f(x) = 0.

##

****

(a)

Show (by hand) that there is a unique simple root r in the interval [0; 1].

****

$f(x) = 3x^5 + 2x - 1$

$f(0) = 0 + 0 - 1 = -1 < 0$

$f(1) = 3 + 2 - 1 = 4 > 0$

On interval [0,1] function changing value from negative to positive.

According intermideate Value Theorem the root will be somewhere in between 0 and 1

```
> bisect:=proc(A,B,TOL,N)
            local i,a,b;a:=A;b:=B;

>
            i:=0;
            while i<=N do
                 c:=(a+b)/2:
                 if f(c)=0 or (b-a)/2<TOL then
                    printf("c=%.8f, f(c)=%.8f\n",c,f(c));
                    printf("Number of iterations needed: %d",i);
  return();
                 break;
                 end if;
                 if signum(f(a))*signum(f(c))<0 then
                      b:=c;
                 else
                      a:=c;
                 end if;
                 i:=i+1;
            end do;
>
            printf("The method failed after %d iterations.\n",N);
            printf("c=%.8f, f(c)=%.8f\n",c,f(c));
            return();
         end proc:
```

****

(b)

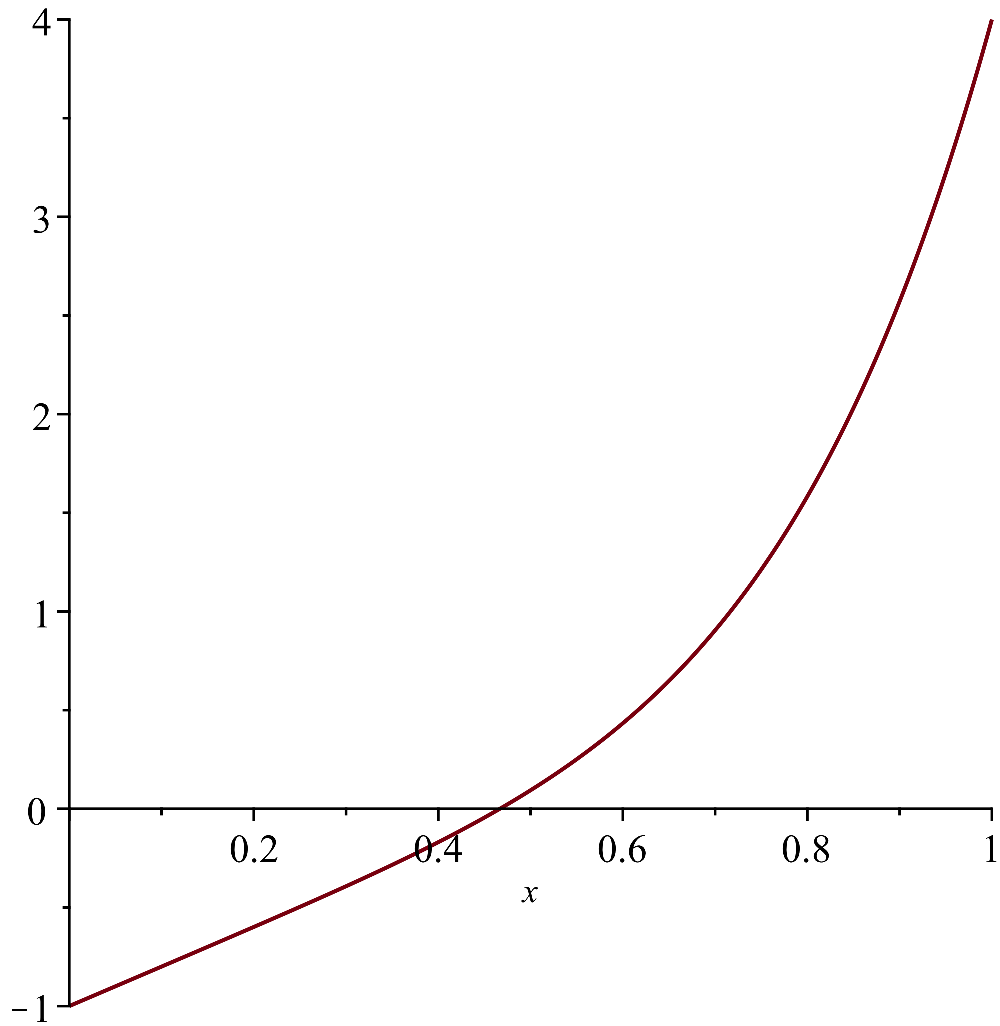
Use the bisection procedure (Maple, Matlab, Python) to approximate r to eight

correct decimal places. Report the approximation xc, number of steps needed, and the backward error |f(xc)|
****

> $f := x \rightarrow 3 \cdot x^5 + 2 \cdot x - 1;$

$$f := x \mapsto 3\,x^5 + 2\,x - 1 \qquad \textbf{(1)}$$

> $plot(f(x), x = 0..1);$



> $\text{bisect}(0, 0.5, 0.5 \cdot 10^{-8}, 1000);$

```
c=0.46676574,  f(c)=0.00000000
Number of iterations needed: 26
```

> $BackwardError := \text{abs}(f(0.46676574))$

$$BackwardError := 1.6\ 10^{-8} \qquad \textbf{(2)}$$

>

****
Answer: Xc = 0.46676574 approximate to 8 decimal places
found in 26 iterations on interval [0.0, 1.0] with Bisection Method

BackwardError $:= \mathbf{1.6 \cdot 10^{-8}}$
****

```
>
fixedpoint:=proc(g,X0,TOL,N)
          local i,x0;x0:=X0;
          i:=1;
          while i<=N do
               r:=g(x0);
               printf("Iteration %d: %.8g\n",i,r);
               if abs(r-x0)<TOL then
                  printf("Fixed point r=%.8g, g(r)=%.8g\n",r,g(r));
                  printf("Number of iterations needed: %d",i);return
();
               break;
               end if;
               i:=i+1;x0:=r;
          end do;
          printf("The method failed after %d iterations.\n",N);
          printf("r=%.8g, g(r)=%.8g\n",r,g(r));
          return();
       end proc:
```

\>

****

(c)

Build a fixed point iteration procedure g(x) = x (not Newton's) to find the root of f(x), starting from x0 = 0.5. Report the approximation xc, number of steps needed, and the backward error |f(xc)|

$\mathbf{3 \cdot x^5 + 2 \cdot x - 1 = 0}$

$\mathbf{2 \cdot x = 1 - 3 \cdot x^5} \Rightarrow \mathbf{x} = \dfrac{\mathbf{(1 - 3 \cdot x^5)}}{\mathbf{2}} = \boldsymbol{g(x)} \rightarrow FPI$

****

\> $g(x) := x \rightarrow \dfrac{(1 - 3 \cdot x^5)}{2}$

$$g(x) := x \mapsto \frac{1}{2} - \frac{3\,x^5}{2} \qquad \textbf{(3)}$$

\> $x0 := 0.5$

$$x0 := 0.5 \qquad \textbf{(4)}$$

\> $fixedpoint(g(x), x0, 0.5 \cdot 10^{-8}, 50)$

```
Iteration 1: 0.453125
Iteration 2: 0.47134626
Iteration 3: 0.46510272
Iteration 4: 0.46735357
Iteration 5: 0.46655593
Iteration 6: 0.46684036
Iteration 7: 0.46673916
Iteration 8: 0.46677519
Iteration 9: 0.46676237
Iteration 10: 0.46676693
Iteration 11: 0.46676531
Iteration 12: 0.46676589
```

```
Iteration 13: 0.46676568
Iteration 14: 0.46676575
Iteration 15: 0.46676573
Iteration 16: 0.46676574
Iteration 17: 0.46676573
Fixed point r=0.46676573, g(r)=0.46676573
Number of iterations needed: 17
```

> BackwardError := abs(f(0.46676573))

$$BackwardError := 1.10\ 10^{-8} \tag{5}$$

****

Answer: Xc = 0.46676573 approximate to 8 decimal places
found in 17 iterations with initial guess x0=0.5 with FPI Method

BackwardError $:= \mathbf{1.10 \cdot 10^{-8}}$

****

$$BackwardError := 1.100000000\ 10^{-8} \tag{6}$$

```
newton:=proc(f,X0,TOL,N)
           local i,x0;x0:=X0;
           i:=1;
           while i<=N do
                if D(f)(p0)=0 then
                   printf("Division by 0. Method failed");break;
                else
                   r:=x0-f(x0)/D(f)(x0);
                   printf("Iteration %d: %.8g\n",i,r);
                   if abs(r-x0)<TOL then
                      printf("r=%.8g, f(r)=%.8g\n",r,f(r));
                      printf("Number of iterations needed: %d",i);
return();
                   break;
                   end if;
                   i:=i+1;x0:=r;
                end if;
           end do;
           printf("The method failed after %d iterations.\n",N);
           printf("r=%.8g, f(r)=%.8g\n",r,f(r));
           return();
        end proc:
```

>

****

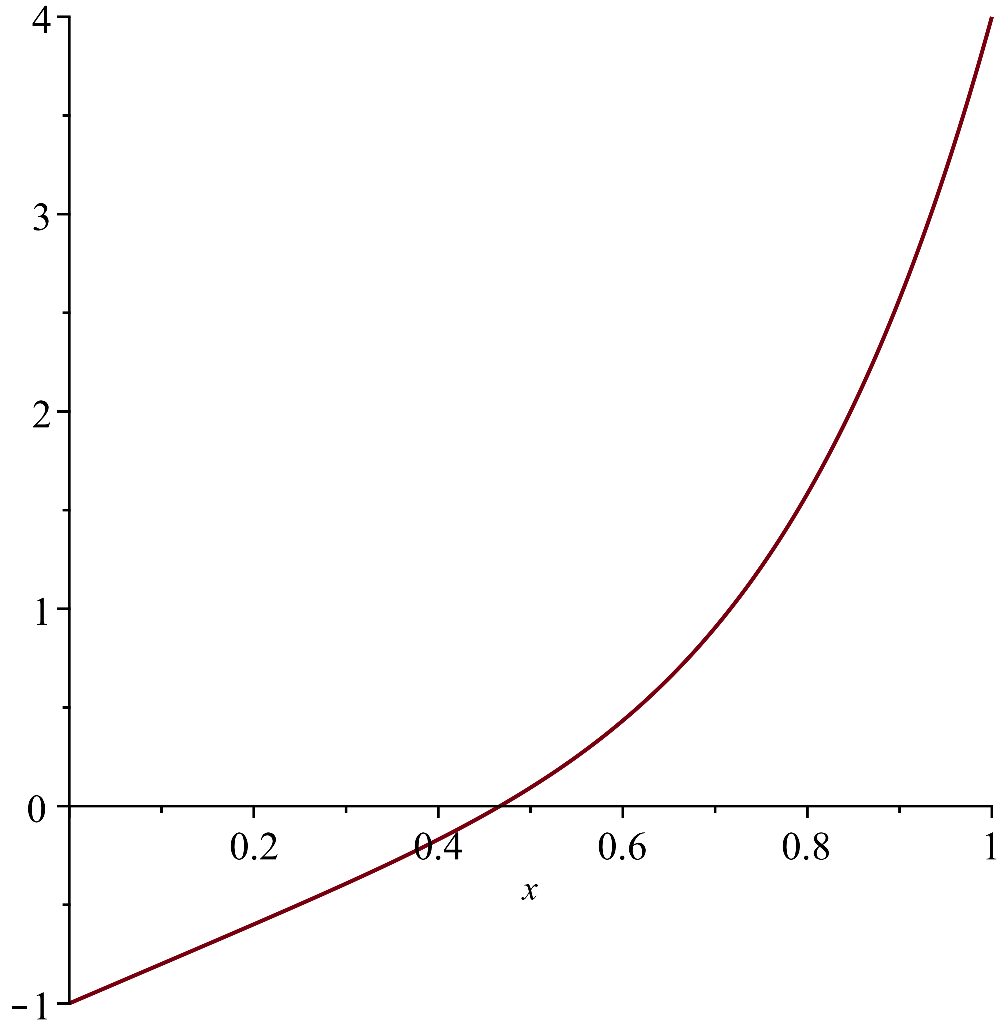
(d)

Use Newton's method to approximate r to eight correct decimal places (starting with x0 = 0:5). Report the approximation xc, number of steps needed, and the backward error |f(xc)|

****

> $f(x); plot(f(x), x = 0..1);$

$$3x^5 + 2x - 1$$



> $newton\left(f, 0.5, 0.5 \cdot 10^{-8}, 50\right)$

```
Iteration 1: 0.46808511
Iteration 2: 0.4667677
Iteration 3: 0.46676573
Iteration 4: 0.46676573
r=0.46676573, f(r)=-1e-10
Number of iterations needed: 4
```

> BackwardError := abs(f(0.46676573))

$$BackwardError := 1.10\ 10^{-8} \tag{7}$$

****

Answer: Xc = 0.46676573 approximate to 8 decimal places
found in 4 iterations with initial guess x0=0.5 with Newton Method

BackwardError := $1.10 \cdot 10^{-8}$

****

$$BackwardError := 1.100000000\ 10^{-8} \tag{8}$$

```
Halley:=proc(f,X0,TOL,N)
          local i,x0;x0:=X0;
          i:=1;
          while i<=N do
               if D(f)(p0)=0 then
```

```
                    printf("Division by 0. Method failed");break;
                 else
                    r:=x0-( (2*f(x0)*(D(f))(x0) )/(2*((D(f))(x0))^2-f
(x0)*(D(D(f)))(x0)));
                    printf("Iteration %d: %.8g\n",i,r);
                    if abs(r-x0)<TOL then
                       printf("r=%.8g, f(r)=%.8g\n",r,f(r));
                       printf("Number of iterations needed: %d",i);
return();
                    break;
                    end if;
                    i:=i+1;x0:=r;
                 end if;
            end do;
            printf("The method failed after %d iterations.\n",N);
            printf("r=%.8g, f(r)=%.8g\n",r,f(r));
            return();
         end proc:
>
```

****

(e)

Modify the code on Newton's method in order to solve the equation using Halley's method, an iterative process given by

$$Xk+1 = H(X) = Xk - \frac{2 \cdot f(Xk) \cdot f'(Xk)}{2 \cdot (f'(Xk))^2 - f(Xk) \cdot f''(Xk)}$$

Approximate r to eight correct decimal places (starting with x0 = 0:5). Report the approximation xc, number of steps needed, and the backward error |f(xc)|

****

false **(9)**

> $Halley(f, x0, 0.5 \cdot 10^{-8}, 50)$

```
Iteration 1: 0.46672959
Iteration 2: 0.46676573
Iteration 3: 0.46676573
r=0.46676573, f(r)=0
Number of iterations needed: 3
```

> BackwardError := abs(f(0.46676573))

$BackwardError := 1.10\ 10^{-8}$ **(10)**

****

Answer: Xc = 0.46676573 approximate to 8 correct decimal places found in 3 iterations with initial guess x0=0.5 with Halley's Method

BackwardError := $1.10 \cdot 10^{-8}$

$BackwardError := 1.100000000\ 10^{-8}$ **(11)**

###

2.

(25 pts) The total cost of a federal government program in millions of dollars is shown in the table below. (Notice that the year 2010 data is missing.)

****

(c)

Perform a least-squares analysis and estimate the 2010 cost and compare to the actual value of $1,049 (in mil). (Choose an appropriate least squares linear or polynomial model.)

****

Linear
model : y=a+b(x-2005)
Polynomial 10th degree
model: y=

$a + b\cdot(x - 2005) + c\cdot(x-2005)^2 + d\cdot(x-2005)^3 + e\cdot(x-2005)^4 + f\cdot(x-2005)^5 + g\cdot(x - 2005)^6 + h\cdot(x-2005)^7 + i\cdot(x-2005)^8 + j\cdot(x-2005)^9 + j\cdot(x-2005)^{10}$

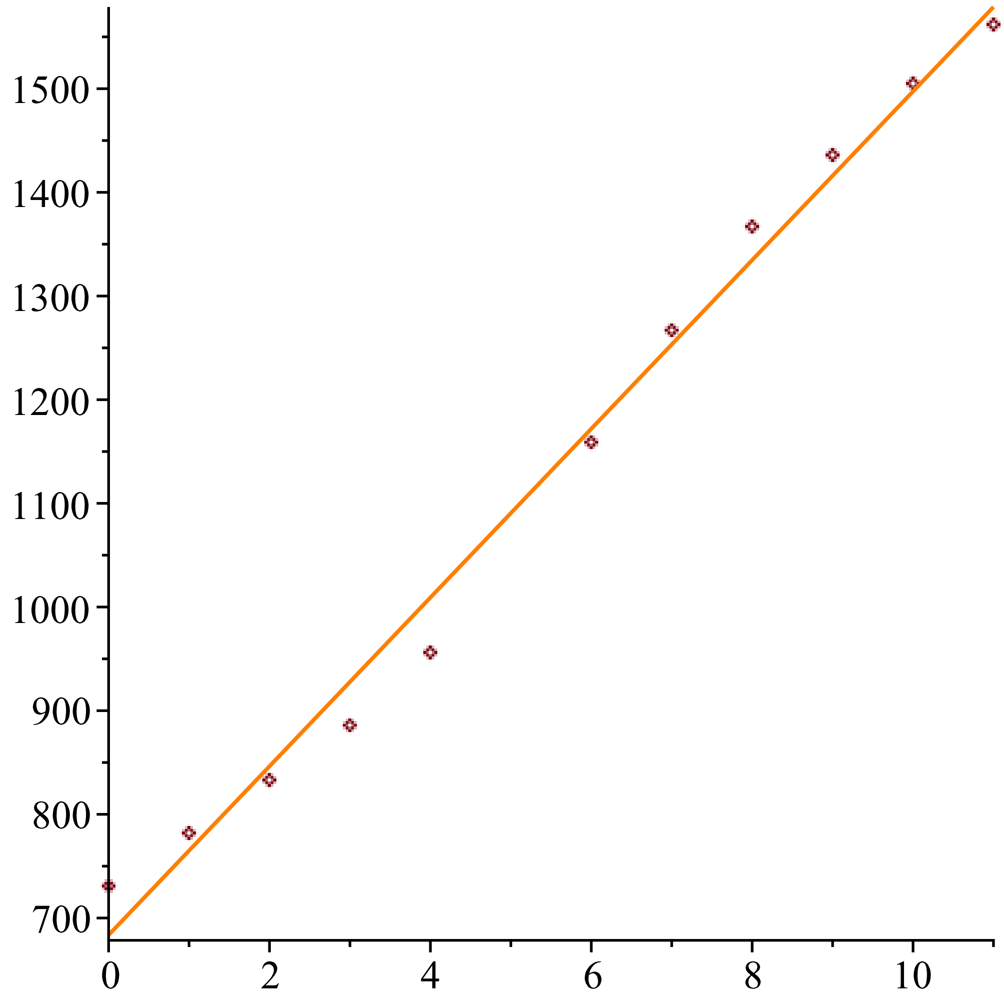
```
> restart; with(plots) : with(CurveFitting) : with(LinearAlgebra) : actual_cost_2010 := 1049;
```

$$actual\_cost\_2010 := 1049 \quad (12)$$

```
> xvalues := ⟨0, 1, 2, 3, 4, 6, 7, 8, 9, 10, 11⟩ : yvalues := ⟨731, 782, 833, 886, 956, 1159, 1267, 1367, 1436, 1505, 1562⟩ :
  p1 := plot(xvalues, yvalues, style = point) :
> ls_line := CurveFitting[LeastSquares](xvalues, yvalues, x)
```

$$ls\_line := \frac{1073259}{1570} + \frac{127771\,x}{1570} \quad (13)$$

```
>
> p2_ls_line := plot(ls_line, x = 0 ..11, color = coral) : display(p1, p2_ls_line)
```



>

>

>

> $interface(rtablesize = 12); yout := map\left(x \rightarrow \frac{1073259}{1570} + \frac{127771\, x}{1570}, xvalues\right)$

10

(14)

$$yout := \begin{bmatrix} \frac{1073259}{1570} \\ \frac{120103}{157} \\ \frac{1328801}{1570} \\ \frac{728286}{785} \\ \frac{1584343}{1570} \\ \frac{367977}{314} \\ \frac{983828}{785} \\ \frac{2095427}{1570} \\ \frac{1111599}{785} \\ \frac{2350969}{1570} \\ \frac{247874}{157} \end{bmatrix} \tag{14}$$

>

> $\mathrm{Calculated\_LeastSquare\_Line\_2010} := \mathrm{evalf}(\mathrm{subs}(x = 2010 - 2005, \mathrm{ls\_line}))$

$$\mathit{Calculated\_LeastSquare\_Line\_2010} := 1090.518471 \tag{15}$$

```
> absolute_error := abs(actual_cost_2010-
  Calculated_LeastSquare_Line_2010)
```

$$\mathit{absolute\_error} := 41.518471 \tag{16}$$

> $\mathrm{relative\_error} := 100 \cdot \left(\frac{\mathrm{absolute\_error}}{\mathrm{actual\_cost\_2010}} \right)$

$$\mathit{relative\_error} := 3.957909533 \tag{17}$$

> $\mathrm{RMSE\_ls\_line} := \mathrm{evalf}\left(\frac{\mathrm{Norm}(\mathrm{yvalues} - \mathrm{yout}, 2)}{\sqrt{\mathrm{Dimension}(\mathrm{xvalues})}} \right)$

$$\mathit{RMSE\_ls\_line} := 29.26255267 \tag{18}$$

****

Answer:

Calculated_LeastSquare_Line_2010 ≔ 1090.518
absolute_error ≔ 41.518
relative_error ≔ 3.958 %

RMSE_ls_line ≔ 29.263

****

\>

\> $unassign('a','b','c','x') : interface(rtablesize = 12);$

$$12 \tag{19}$$

\>

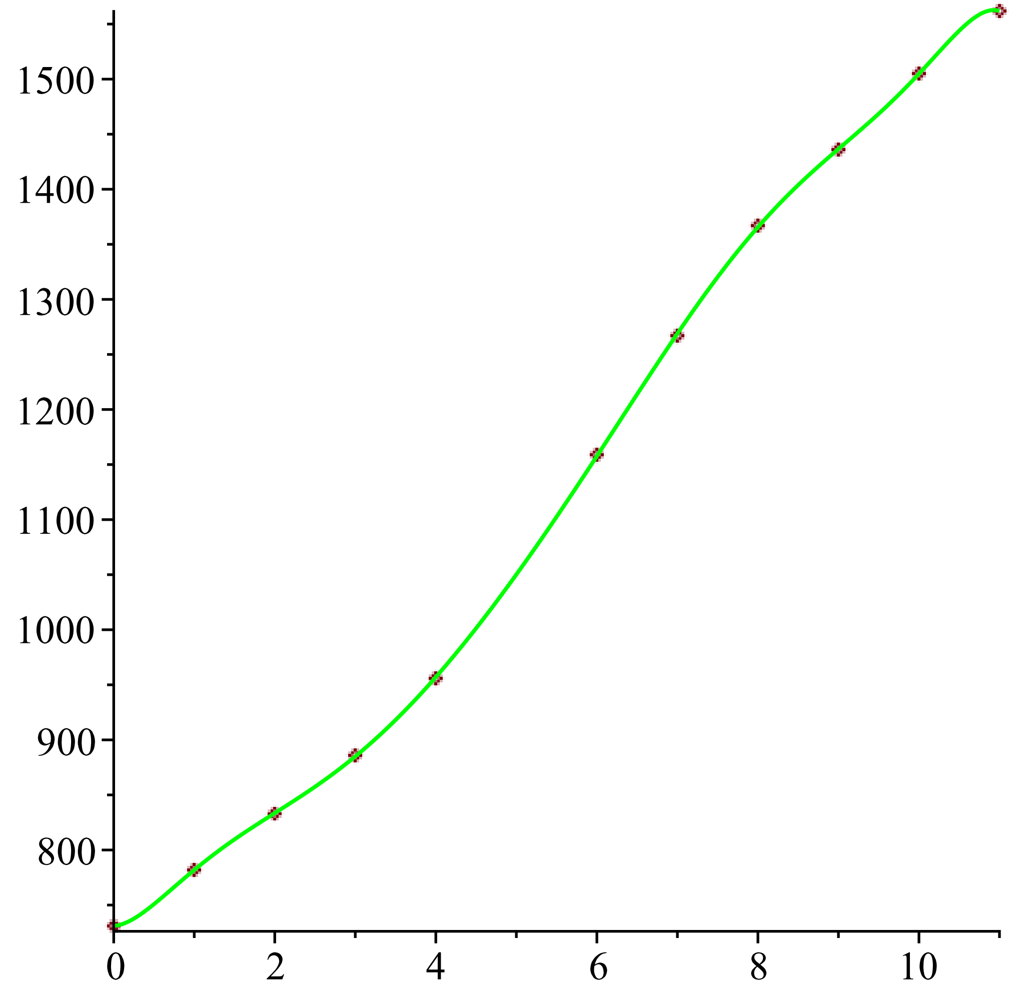
$ls\_poly := CurveFitting[LeastSquares](xvalues, yvalues, x, curve = a + b\cdot x + c\cdot x^2 + d\cdot x^3 + e\cdot x^4 + f\cdot x^5 + g\cdot x^6 + h\cdot x^7 + i\cdot x^8 + j\cdot x^9 + j\cdot x^{10})$

$$ls\_poly := \frac{168887723756}{231031411} + \frac{36482432347}{6930942330}x + \frac{30553404678421}{291099577860}x^2 - \frac{62620127019091}{698638986864}x^3 + \frac{10259709153157}{268707302640}x^4 - \frac{362768476349}{39808489280}x^5 + \frac{10204101847}{7961697856}x^6 - \frac{3666157279}{35827640352}x^7 + \frac{40115941}{10537541280}x^8 - \frac{170963}{63225247680}x^{10} - \frac{170963}{63225247680}x^9 \tag{20}$$

\>

\> $p3\_ls\_poly := plot(ls\_poly, x = 0..11, color = green) :$

\> $display(p1, p3\_ls\_poly)$



$$> yout1 := map\left(x \to \frac{168887723756}{231031411} + \frac{36482432347}{6930942330}\,x + \frac{30553404678421}{291099577860}\,x^2 - \frac{62620127019091}{698638986864}\,x^3 + \frac{10259709153157}{268707302640}\,x^4 - \frac{362768476349}{39808489280}\,x^5 + \frac{10204101847}{7961697856}\,x^6 - \frac{3666157279}{35827640352}\,x^7 + \frac{40115941}{10537541280}\,x^8 - \frac{170963}{63225247680}\,x^{10} - \frac{170963}{63225247680}\,x^9, xvalues\right)$$

$$yout1 := \begin{bmatrix} \frac{168887723756}{231031411} \\ \frac{180633989036}{231031411} \\ \frac{192569316713}{231031411} \\ \frac{204457532491}{231031411} \\ \frac{221098321526}{231031411} \\ \frac{267451409821}{231031411} \\ \frac{293157352687}{231031411} \\ \frac{315495530192}{231031411} \\ \frac{331902617786}{231031411} \\ \frac{26743658705}{17771647} \\ \frac{21227928103}{13590083} \end{bmatrix} \tag{21}$$

\>

\> **Calculated_LeastSquare_Poly_2010 ≔ evalf(subs(x = 2010 − 2005, ls_poly))**

$$Calculated\_LeastSquare\_Poly\_2010 := 1050.122300 \tag{22}$$

\> **absolute_error ≔ abs(actual_cost_2010 −Calculated_LeastSquare_Poly_2010)**

$$absolute\_error := 1.122300 \tag{23}$$

```
> relative_error := 100*(absolute_error/actual_cost_2010)
```

$$relative\_error := 0.1069876072 \tag{24}$$

\> $\mathbf{RMSE := evalf\left(\frac{Norm(yvalues - yout1, 2)}{\sqrt{Dimension(xvalues)}}\right)}$

$$RMSE := 0.9629895927 \tag{25}$$

****

Answer:
Calculated Least Square Polynomial 10th degree:
Calculated_LeastSquare_Poly_2010 ≔ 1050.122
absolute_error ≔ 1.122
relative_error ≔ 0.107 %

RMSE ≔ 0.963

****

(a)

Determine and plot the degree 10 Lagrange polynomial through the data. Use it to estimate the 2010 cost and compare to the actual value of $1,049 (in mil.).

****

Lagrange Polinomial:

$$
\begin{aligned}
f := x \rightarrow\; & (731\cdot((x-1)\cdot(x-2)\cdot(x-3)\cdot(x-4)\cdot(x-6)\cdot(x-7)\cdot(x-8)\cdot(x-9)\cdot(x-10)\cdot(x-11)))/((0-1)\cdot(0-2)\cdot(0-3)\cdot(0-4)\cdot(0-6)\cdot(0-7)\cdot(0-8)\cdot(0-9)\cdot(0-10)\cdot(0-11)) \\
&+ (782\cdot((x-0)\cdot(x-2)\cdot(x-3)\cdot(x-4)\cdot(x-6)\cdot(x-7)\cdot(x-8)\cdot(x-9)\cdot(x-10)\cdot(x-11)))/((1-0)\cdot(1-2)\cdot(1-3)\cdot(1-4)\cdot(1-6)\cdot(1-7)\cdot(1-8)\cdot(1-9)\cdot(1-10)\cdot(1-11)) \\
&+ (833\cdot((x-0)\cdot(x-1)\cdot(x-3)\cdot(x-4)\cdot(x-6)\cdot(x-7)\cdot(x-8)\cdot(x-9)\cdot(x-10)\cdot(x-11)))/((2-0)\cdot(2-1)\cdot(2-3)\cdot(2-4)\cdot(2-6)\cdot(2-7)\cdot(2-8)\cdot(2-9)\cdot(2-10)\cdot(2-11)) \\
&+ (886\cdot((x-0)\cdot(x-1)\cdot(x-2)\cdot(x-4)\cdot(x-6)\cdot(x-7)\cdot(x-8)\cdot(x-9)\cdot(x-10)\cdot(x-11)))/((3-0)\cdot(3-1)\cdot(3-2)\cdot(3-4)\cdot(3-6)\cdot(3-7)\cdot(3-8)\cdot(3-9)\cdot(3-10)\cdot(3-11)) \\
&+ (956\cdot((x-0)\cdot(x-1)\cdot(x-2)\cdot(x-3)\cdot(x-6)\cdot(x-7)\cdot(x-8)\cdot(x-9)\cdot(x-10)\cdot(x-11)))/((4-0)\cdot(4-1)\cdot(4-2)\cdot(4-3)\cdot(4-6)\cdot(4-7)\cdot(4-8)\cdot(4-9)\cdot(4-10)\cdot(4-11)) \\
&+ (1159\cdot((x-0)\cdot(x-1)\cdot(x-2)\cdot(x-3)\cdot(x-4)\cdot(x-7)\cdot(x-8)\cdot(x-9)\cdot(x-10)\cdot(x-11)))/((6-0)\cdot(6-1)\cdot(6-2)\cdot(6-3)\cdot(6-4)\cdot(6-7)\cdot(6-8)\cdot(6-9)\cdot(6-10)\cdot(6-11)) \\
&+ (1267\cdot((x-1)\cdot(x-2)\cdot(x-3)\cdot(x-4)\cdot(x-6)\cdot(x-7)\cdot(x-8)\cdot(x-9)\cdot(x-10)\cdot(x-11)))/((7-0)\cdot(7-1)\cdot(7-2)\cdot(7-3)\cdot(7-4)\cdot(7-6)\cdot(7-8)\cdot(7-9)\cdot(7-10)\cdot(7-11)) \\
&+ (1367\cdot((x-0)\cdot(x-1)\cdot(x-2)\cdot(x-3)\cdot(x-4)\cdot(x-6)\cdot(x-7)\cdot(x-9)\cdot(x-10)\cdot(x-11)))/((8-0)\cdot(8-1)\cdot(8-2)\cdot(8-3)\cdot(8-4)\cdot(8-6)\cdot(8-7)\cdot(8-9)\cdot(8-10)\cdot(8-11)) \\
&+ (1436\cdot((x-0)\cdot(x-1)\cdot(x-2)\cdot(x-3)\cdot(x-4)\cdot(x-6)\cdot(x-7)\cdot(x-8)\cdot(x-10)\cdot(x-11)))/((9-0)\cdot(9-1)\cdot(9-2)\cdot(9-3)\cdot(9-4)\cdot(9-6)\cdot 7\cdot(9-8)\cdot(9-10)\cdot(10-11)) \\
&+ (1505\cdot((x-0)\cdot(x-1)\cdot(x-2)\cdot(x-3)\cdot(x-4)\cdot(x-6)\cdot(x-7)\cdot(x-8)\cdot(x-9)\cdot(x-11)))/((10-0)\cdot(10-1)\cdot(10-2)\cdot(10-3)\cdot(10-4)\cdot(10-6)\cdot(10-7)\cdot(10-8)\cdot(10-9)\cdot(0-11)) \\
&+ (1562\cdot((x-0)\cdot(x-1)\cdot(x-2)\cdot(x-3)\cdot(x-4)\cdot(x-6)\cdot(x-7)\cdot(x-8)\cdot(x-9)\cdot(x-10)))/((11-0)\cdot(11-1)\cdot(11-2)\cdot(11-3)\cdot(11-4)\cdot(11-6)\cdot(11-7)\cdot(11-8)\cdot(11-9)\cdot(11-10))
\end{aligned}
$$

```
> with(plots):with(CurveFitting);with(LinearAlgebra);
  with(ArrayTools):interface(rtablesize=15);
```

[ArrayInterpolation, BSpline, BSplineCurve, Interactive, LeastSquares, Lowess, PolynomialInterpolation, RationalInterpolation, Spline, ThieleInterpolation]

Typesetting:-mparsed(with(LinearAlgebra); with(ArrayTools) : interface(rtablesize = 15); , with(LinearAlgebra); **(26)**

with(ArrayTools) :

interface(rtablesize = 15))

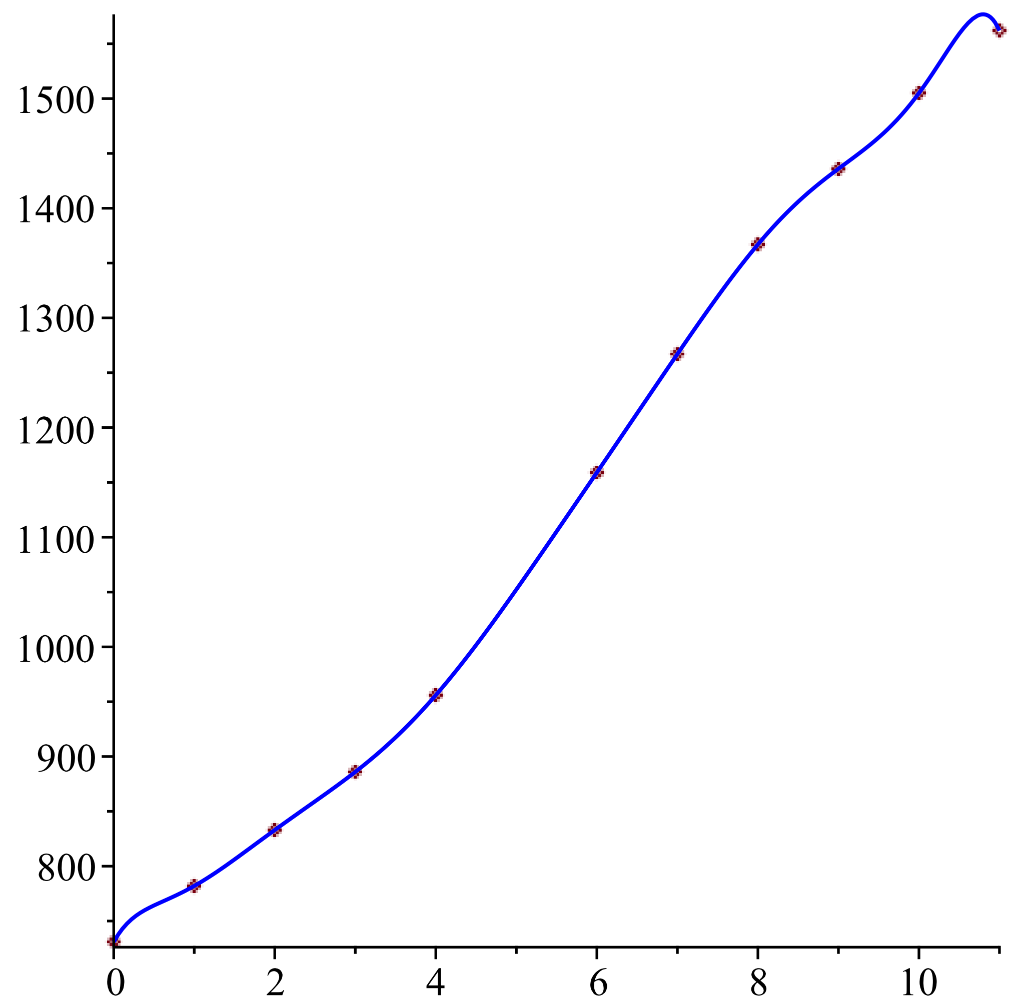
>

```
> data1 := [[0, 731], [1, 782], [2, 833], [3, 886], [4, 956], [6, 1159], [7,
    1267], [8, 1367], [9, 1436], [10, 1505], [11, 1562]]:
```

```
> poly := PolynomialInterpolation(data1, x);
```

$$poly := -\frac{599}{6652800}x^{10}+\frac{6353}{1330560}x^{9}-\frac{299}{2772}x^{8}+\frac{300259}{221760}x^{7}-\frac{363303}{35200}x^{6}+\frac{68993}{1408}x^{5}-\frac{19032107}{133056}x^{4}+\frac{81659899}{332640}x^{3}-\frac{61824641}{277200}x^{2}+\frac{152056}{1155}x+731 \qquad \textbf{(27)}$$

```
> p2Lagrange := plot(poly, x = 0..11, color = blue) :
> display(p1, p2Lagrange)
```



$$> \quad yout := map\left(x \to -\frac{599}{6652800}x^{10}+\frac{6353}{1330560}x^{9}-\frac{299}{2772}x^{8}+\frac{300259}{221760}x^{7}-\frac{363303}{35200}x^{6}+\frac{68993}{1408}x^{5}-\frac{19032107}{133056}x^{4}+\frac{81659899}{332640}x^{3}-\frac{61824641}{277200}x^{2}+\frac{152056}{1155}x+731,\; xvalues\right)$$

$$yout := \begin{bmatrix} 731 \\ 782 \\ 833 \\ 886 \\ 956 \\ 1159 \\ 1267 \\ 1367 \\ 1436 \\ 1505 \\ 1562 \end{bmatrix} \quad (28)$$

> **calculated_2010_polynomial ≔ evalf(subs(x = 2010 − 2005, poly))**

$$calculated\_2010\_polynomial := 1052.272727 \quad (29)$$

> **absolute_error ≔ abs(actual_cost_2010 − calculated_2010_polynomial)**

$$absolute\_error := 3.272727 \quad (30)$$

> $\mathbf{relative\_error := 100 \cdot \left(\frac{absolute\_error}{actual\_cost\_2010} \right)}$

$$relative\_error := 0.3119854147 \quad (31)$$

> $\mathbf{RMS\_lagrange\_poly := evalf\left(\frac{Norm(yvalues - yout, 2)}{\sqrt{Dimension(xvalues)}} \right)}$

$$RMS\_lagrange\_poly := 0. \quad (32)$$

>

****

Answer: Lagrange 10th Degree Polynomial
calculated_2010_polynomial ≔ 1052.273
absolute_error ≔ 3.273
relative_error ≔ 0.312%

****

(b)

Determine and plot a clamped cubic spline for the data (using the appropriate slopes at 2005 and 2016). Use it to estimate the 2010 cost and compare to the actual value of $1,049 (in mil.)

****

> $slope\_0 := \frac{(data1[2][2] - data1[1][2])}{data1[2][1] - data1[1][1]}$

$$slope\_0 := 51 \quad (33)$$

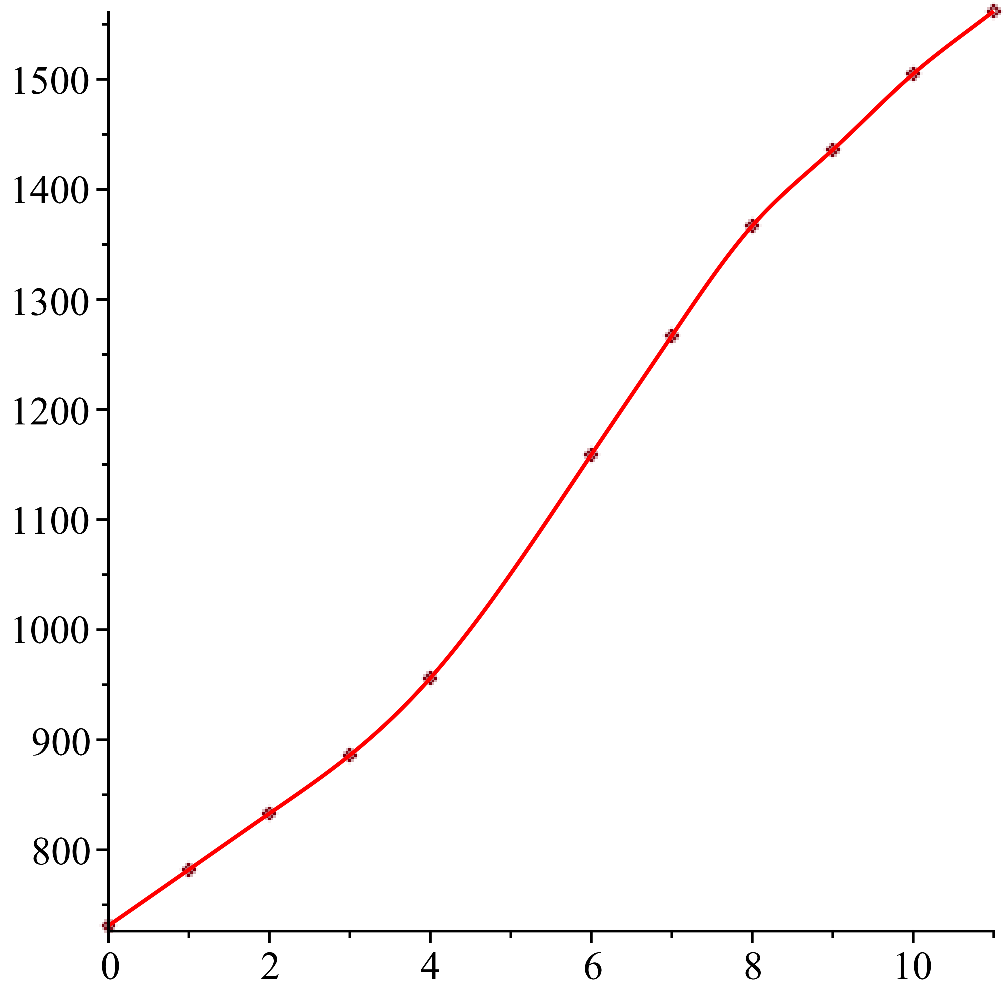
> $slope\_1 := \frac{(data1[11][2] - data1[10][2])}{data1[11][1] - data1[10][1]}$

$$slope\_1 := 57 \quad (34)$$

```
> p1_clamped_spline := plot(data1, style=point, color=black) :
> clamped_spline := evalf(Spline(data1, x, endpoints=[slope_0, slope_1]))
```

$$clamped\_spline := \begin{cases} 0.1255477457\,x^3 - 0.1255477457\,x^2 + 51.\,x + 731. & x < 1. \\ -0.3766432370\,x^3 + 1.381025202\,x^2 + 49.49342705\,x + 731.5021910 & x < 2. \\ 3.381025202\,x^3 - 21.16498543\,x^2 + 94.58544832\,x + 701.4408435 & x < 3. \\ 1.852542428\,x^3 - 7.408640466\,x^2 + 53.31641342\,x + 742.7098784 & x < 4. \\ -2.778265896\,x^3 + 48.16105943\,x^2 - 168.9623862\,x + 1039.081611 & x < 6. \\ 0.9301165366\,x^3 - 18.58982437\,x^2 + 231.5429166\,x + 238.0710056 & x < 7. \\ -8.955129268\,x^3 + 189.0003375\,x^2 - 1221.588217\,x + 3628.710317 & x < 8. \\ 11.89040054\,x^3 - 311.2923777\,x^2 + 2780.753506\,x - 7044.200943 & x < 9. \\ -7.606472873\,x^3 + 215.1232043\,x^2 - 1956.986733\,x + 7169.019772 & x < 10. \\ 6.535490958\,x^3 - 209.1357106\,x^2 + 2285.602417\,x - 6972.944059 & otherwise \end{cases} \qquad (35)$$

```
> p2_clamped_spline := plot(clamped_spline, x=0..11, color=red) :
> display(p1, p2_clamped_spline)
```



> $f := x \rightarrow piecewise(x < 1, 0.1255477457\,x^3 - 0.1255477457\,x^2 + 51.\,x + 731., x < 2, -0.3766432370\,x^3 + 1.381025202\,x^2 + 49.49342705\,x + 731.5021910, x < 3, 3.381025202\,x^3 - 21.16498543\,x^2 + 94.58544832\,x + 701.4408435, x < 4, 1.852542428\,x^3 - 7.408640466\,x^2 + 53.31641342\,x + 742.7098784, x < 6, -2.778265896\,x^3 + 48.16105943\,x^2 - 168.9623862\,x + 1039.081611, x < 7, 0.9301165366\,x^3 - 18.58982437\,x^2 + 231.5429166\,x + 238.0710056, x < 8, -8.955129268\,x^3 + 189.0003375\,x^2 - 1221.588217\,x + 3628.710317, x < 9, 11.89040054\,x^3 - 311.2923777\,x^2 + 2780.753506\,x - 7044.200943, x < 10, -7.606472873\,x^3 + 215.1232043\,x^2 - 1956.986733\,x + 7169.019772, x \leq 11, 6.535490958\,x^3 - 209.1357106\,x^2 + 2285.602417\,x - 6972.944059)$

$$f := x \tag{36}$$

$$\mapsto \begin{cases} 0.1255477457\,x^3 - 0.1255477457\,x^2 + 51.\,x + 731. & x < 1 \\ -0.3766432370\,x^3 + 1.381025202\,x^2 + 49.49342705\,x + 731.5021910 & x < 2 \\ 3.381025202\,x^3 - 21.16498543\,x^2 + 94.58544832\,x + 701.4408435 & x < 3 \\ 1.852542428\,x^3 - 7.408640466\,x^2 + 53.31641342\,x + 742.7098784 & x < 4 \\ -2.778265896\,x^3 + 48.16105943\,x^2 - 168.9623862\,x + 1039.081611 & x < 6 \\ 0.9301165366\,x^3 - 18.58982437\,x^2 + 231.5429166\,x + 238.0710056 & x < 7 \\ -8.955129268\,x^3 + 189.0003375\,x^2 - 1221.588217\,x + 3628.710317 & x < 8 \\ 11.89040054\,x^3 - 311.2923777\,x^2 + 2780.753506\,x - 7044.200943 & x < 9 \\ -7.606472873\,x^3 + 215.1232043\,x^2 - 1956.986733\,x + 7169.019772 & x < 10 \\ 6.535490958\,x^3 - 209.1357106\,x^2 + 2285.602417\,x - 6972.944059 & x \le 11 \end{cases}$$

> $yout := map(f, xvalues)$

$$yout := \begin{bmatrix} 731. \\ 782.0000000 \\ 833.0000000 \\ 886.0000001 \\ 955.9999998 \\ 1159.000000 \\ 1266.999997 \\ 1367.000017 \\ 1436.000002 \\ 1505.000011 \\ 1562.000011 \end{bmatrix} \quad (37)$$

> **calculated_clamped_spline_2010 := eval(clamped_spline, x = 2010 − 2005)**

$$calculated\_clamped\_spline\_2010 := 1051.012929 \quad (38)$$

```
> absolute_error := abs(actual_cost_2010-
calculated_clamped_spline_2010)
```

$$absolute\_error := 2.012929 \quad (39)$$

> **relative_error := $100 \cdot \left(\frac{\text{absolute\_error}}{\text{actual\_cost\_2010}}\right)$**

$$relative\_error := 0.1918902765 \quad (40)$$

>

> $RMSE\_clamed\_spline := \text{evalf}\left(\frac{\text{Norm}(yvalues - yout, 2)}{\sqrt{\text{Dimension}(xvalues)}}\right)$

$$RMSE\_clamed\_spline := 7.032715800\ 10^{-6} \quad (41)$$

>

>

****

Answer for Clamped spline:

calculated_clamped_spline_2010 ≔ 1051.013

absolute_error ≔ 2.0123

relative_error ≔ 0.192 %

RMSE_clamed_spline ≔ $7.033 \cdot 10^{-6}$

****

(d)

On one plot show the Lagrange polynomial, the clamped cubic spline and the least-squares \014t. Which method gives a better estimate of the 2010 cost?

****

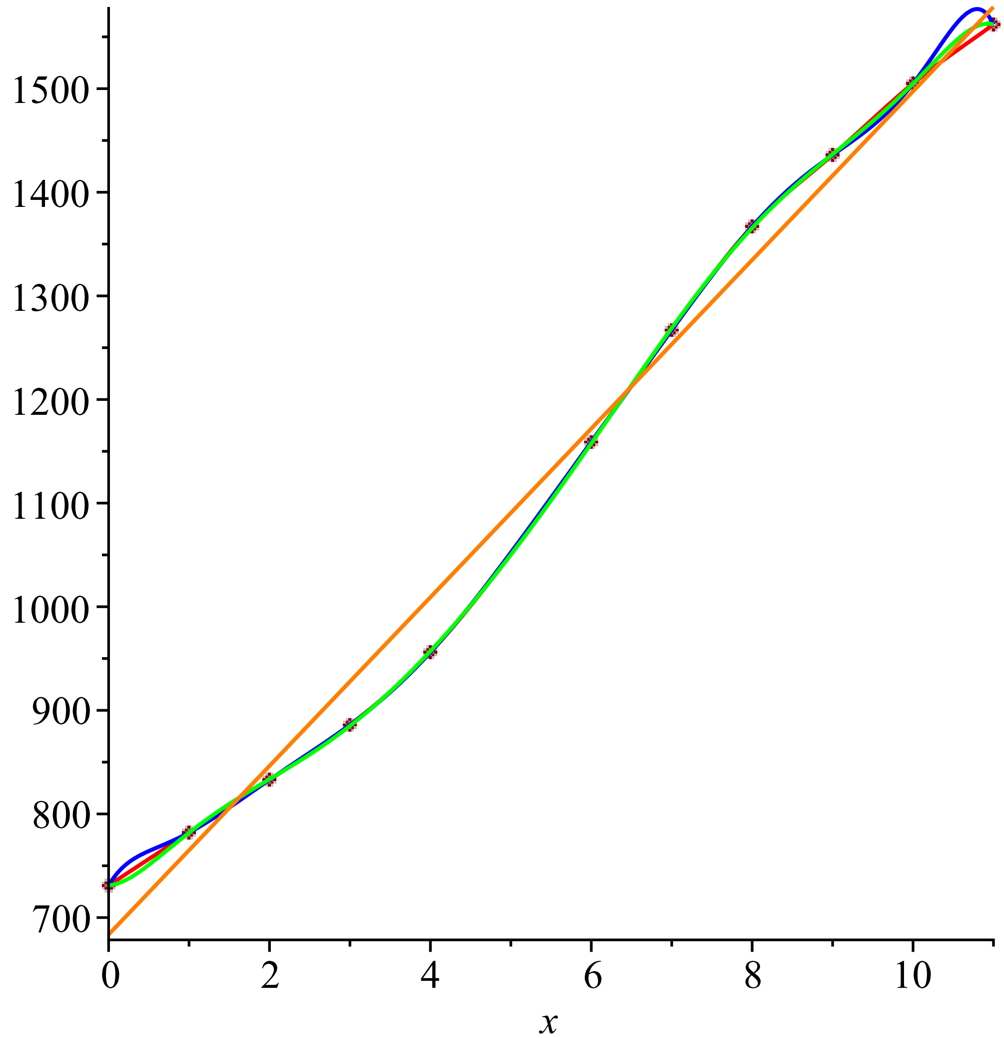
Clamped_spline -> red

Lagrange polynomial->blue

Least square polynomial->green

Least square line-> coral

```
> display(p1, p2_clamped_spline, p2Lagrange, p3_ls_poly, p2_ls_line)
```



On the graph we can see that

`Clamped_spline` -> red

`Lagrange` polynomial->blue

Least square polynomial->green

almost on the same line

Compare absolute errors:

Least Square Line : absolute_error := 41.52

Least square polynomial: absolute_error := 1.12

`Lagrange` polynomial: absolute_error := 3.27

`Clamped_spline` : absolute_error := 2.01

Compare relative errors:

Least Square Line : relative_error := 3.96 %

Least Square polynomial: relative_error := 0.11 %

`Lagrange` polynomial: relative_error := 0.31 %

`Clamped_spline` : relative_error := 0.19 %

Compare RMSE:

Least Square Line : RMSE := 29.26

Least Square polynomial: RMSE := 0.96

`Lagrange` polynomial: RMSE := 0.000000 *gives exact fit trhough the points*
`Clamped_spline`: RMSE := 0.000007

Answer: Using absolute or relative error for the fit we can determine that the better estimate for 2010 cost gives :

Least Square Polynomial Model :

with absolute_error := 1.12
with relative_error := 0.11 %

****

(e)

Which method would you choose to estimate the 2019 cost? Explain.

****

I would use Least Square polynomial to approximate 2019 cost.
There is no reason to think that the relations between the cost can be exactly expressed by a polynomial or spline.
But regression Linear Or Polynomial might do the trick, since the cost might have some slope and a
regression function might be a good aproximation, at least locally.
Regression is definately a better idea than interpolation in this case.
Between Linear or Polinomial I choose Least Square **Polinomial regression line because it does have smaller RMSE.**

Least Square Line : RMSE := 29.26
Least Square polynomial: RMSE := 0.96 < 29.26

##

3.

(15 pts) Integrals of functions like f(x) = exp(-x2) are hard to evaluate by elementary integration techniques. Use the minimal linear congruential generator with seed x0 = 3 and n = 10000 random values to approximate the integral $\int_0^2 \exp(-x^2)\,dx$

****

(a)

Type 1 Monte Carlo method: view the integral $\int_0^2 \exp(-x^2)\,dx$ **as twice the average value of the function f(x) = exp(x2) on the interval [0; 2] which you can approximate with the Monte Carlo method**

****

$LCG := \mathbf{proc}\ (m, a, b, x0, N, scale, shift)$
local s, i;
$s[0] := x0$;
for i **to** N **do** $s[i] :=$ \`$mod$\`$(a * s[i-1] + b, m)$
end do;
return $seq(scale * s[i] - shift, i = 1\ ..\ N)$
end proc:

> $\boldsymbol{n := 10000;\quad x0 := 3;\quad m := 2\hat{}31 - 1;\quad a := 7^5;\quad b := 0;}$

> $A := (m, a, b, x0, n, scale, shift) \to evalf\left(\frac{\boldsymbol{LCG}(m, a, b, x0, n, scale, shift)}{m}\right)$

$$A := (m, a, b, x0, n, scale, shift) \mapsto evalf\left(\frac{LCG(m, a, b, x0, n, scale, shift)}{m}\right) \quad \textbf{(42)}$$

> $\boldsymbol{theoretical\_value := evalf\left(\int_0^2 \exp(-x^2)\,\mathrm{d}x\right)}$

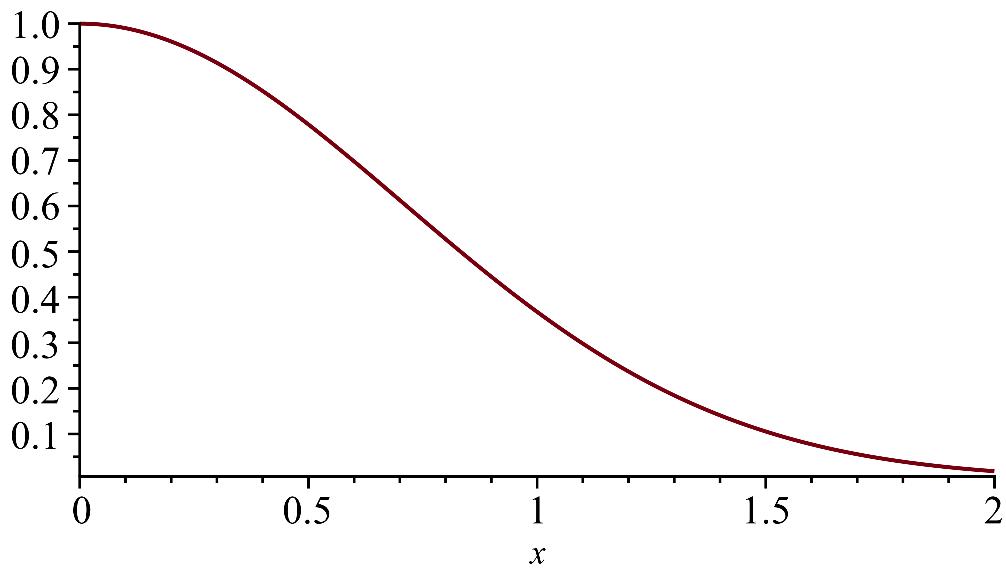
$$theoretical\_value := 0.8820813910 \quad \textbf{(43)}$$

> $f := x \to \exp(-x^2)$

$$f := x \mapsto \mathrm{e}^{-x^2} \quad \textbf{(44)}$$

> $plotf := plot(f(x), x = 0.0\ ..2.0)$:

> $display(plotf, scaling = constrained)$



> $xvalues\_mc\_type1 := A(m, a, b, x0, n, 2, 0)$:

> $yvalues\_mc\_type1 := map(f, [xvalues\_mc\_type1])$:

> $Stat\_mc\_type1 := 2 \cdot Statistics[Mean](yvalues\_mc\_type1)$

$$Stat\_mc\_type1 := 0.874263292637839 \tag{45}$$

****

(*b*) ***Type* 2 *Monte Carlo method* : *enclose the area under the graph* in *the rectangular region* [0; 2] *x* [0; 1], *generate random pairs* · (*x*; *y*) in *that region* and *count how many land under the graph of f* (*x*) = exp(−*x2*); then *estimate the area under the graph, hence the integral.***

```
> arearatio := proc(l, m)
      local i, k, n;
       i := 1; k := 0; n := nops(l);
       while i ≤ n − 1 do
          if m[i] ≤ f(l[i]) and m[i] ≥ g(l[i])
          then k := k + 1;
       end if;
       i := i + 1;
       end do;
```

return$\left(\frac{k}{\text{n}}\right)$;
end proc:

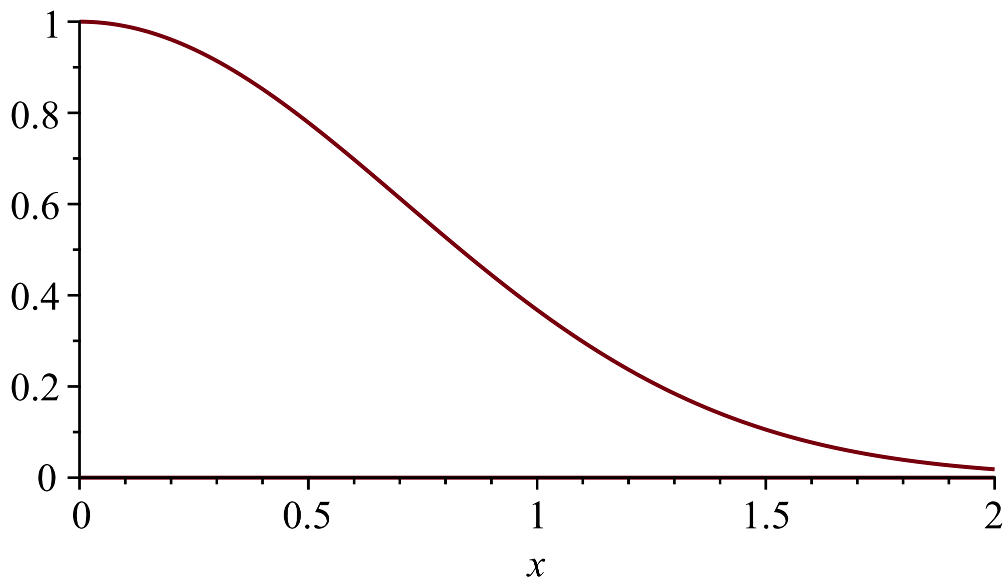
> $g := x \rightarrow 0$

$$g := x \mapsto 0 \tag{46}$$

>

> $plotg := plot(g(x), x = 0.0 ..2.0)$:

> $display(plotf, plotg, scaling = constrained)$



> $\boldsymbol{xvalues\_mc\_type2 := A(m, a, b, x0, n, 2, 0)}$:

> $\boldsymbol{yvalues\_mc\_type2 := A(m, a, b, x0, n, 1, 0)}$:

> $\boldsymbol{Stat\_mc\_type2 := 2 \cdot evalf(arearatio([xvalues\_mc\_type2], [yvalues\_mc\_type2]))}$

$$\boldsymbol{Stat\_mc\_type2 := 0.8834000000} \tag{47}$$

****

> **absolute_error_type_1 := abs(theoretical_value − Stat_mc_type1)**

$$\textbf{absolute\_error\_type\_1 := 0.00781809836216141} \tag{48}$$

> absolute_error_type_2 := abs(theoretical_value - Stat_mc_type2)

absolute_error_type_2 := 0.0013186090 **(49)**

Answer:

theoretical_value := 0.8821   absolute_error_type_1 := 0.007818

a) For Type 1 Monte Carlo area under the graph:

Stat_mc_type1 := 0.8743   absolute_error_type_2 := 0.001319

b) For Type 2 Monte Carlo area under the graph:

Stat_mc_type2 := 0.8834

##

4

(25 pts) Consider the nxn matrix with entries Aij = 1=(i+2j-1), set x = [1; : : : ; 1]T , and b = Ax. You act as the \quality control" person trying to find out what kind of job Maple's LinearSolve command does for this matrix (using double-oating precision computations).

****

(a)

Find ||A||∞ (by hand) when n = 5 and check your work with Maple/Python.

****

> $with(LinearAlgebra) : with(plots) :$

> $n := [5, 10, 15]$

$$n := [5, 10, 15] \quad \textbf{(50)}$$

> $M := x \rightarrow Matrix\left(x, x, (i, j) \rightarrow \left(\frac{1}{i+2j-1}\right)\right)$

$$M := x \mapsto Matrix\left(x, x, (i, j) \mapsto \frac{1}{i+2j-1}\right) \quad \textbf{(51)}$$

> $X := x \rightarrow Matrix(x, 1, (i, j) \rightarrow (1))$

$$X := x \mapsto Matrix(x, 1, (i, j) \mapsto 1) \quad \textbf{(52)}$$

> $A := M(n[1])$

$$A := \begin{bmatrix} \frac{1}{2} & \frac{1}{4} & \frac{1}{6} & \frac{1}{8} & \frac{1}{10} \\ \frac{1}{3} & \frac{1}{5} & \frac{1}{7} & \frac{1}{9} & \frac{1}{11} \\ \frac{1}{4} & \frac{1}{6} & \frac{1}{8} & \frac{1}{10} & \frac{1}{12} \\ \frac{1}{5} & \frac{1}{7} & \frac{1}{9} & \frac{1}{11} & \frac{1}{13} \\ \frac{1}{6} & \frac{1}{8} & \frac{1}{10} & \frac{1}{12} & \frac{1}{14} \end{bmatrix} \quad \textbf{(53)}$$

> $AA := Norm(A, \text{infinity})$

$$AA := \frac{137}{120} \quad \textbf{(54)}$$

> $\boldsymbol{AAdec := evalf(AA)}$

$$AAdec := 1.141666667 \quad \textbf{(55)}$$

Answer:

||A||∞ (by hand) = $\mathbf{\frac{137}{120}}$ **= 1.1417 (attached witten page)**

****

(b)

Use the LinearSolve(A,b) command in Maple (or linalg.solve in Python) to compute the solution xc in double-precision for n = 5; 10; 15.

****

$$\frac{137}{120} \quad (56)$$

> $with(MTM):$

> $interface(rtablesize = 15);$

$$12 \quad (57)$$

> $n[1]$

$$5 \quad (58)$$

> $A\_5 := M(n[1]);$

$$A\_5 := \begin{bmatrix} \frac{1}{2} & \frac{1}{4} & \frac{1}{6} & \frac{1}{8} & \frac{1}{10} \\ \frac{1}{3} & \frac{1}{5} & \frac{1}{7} & \frac{1}{9} & \frac{1}{11} \\ \frac{1}{4} & \frac{1}{6} & \frac{1}{8} & \frac{1}{10} & \frac{1}{12} \\ \frac{1}{5} & \frac{1}{7} & \frac{1}{9} & \frac{1}{11} & \frac{1}{13} \\ \frac{1}{6} & \frac{1}{8} & \frac{1}{10} & \frac{1}{12} & \frac{1}{14} \end{bmatrix} \quad (59)$$

> $x\_5 := X(n[1])$

$$x\_5 := \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \quad (60)$$

>

> $b\_5 := double(A\_5 \cdot x\_5)$

(61)

$$b\_5 := \begin{bmatrix} 1.14166666666667 \\ 0.878210678210678 \\ 0.725000000000000 \\ 0.621800421800422 \\ 0.546428571428571 \end{bmatrix} \quad (61)$$

>

> $Xc\_5 := double(LinearSolve(Transpose(A\_5).A\_5, Transpose(A\_5).b\_5))$

$$Xc\_5 := \begin{bmatrix} 1.00000011133266 \\ 0.999997123469983 \\ 1.00001487507692 \\ 0.999974724640299 \\ 1.00001343707213 \end{bmatrix} \quad (62)$$

>

```
> n[2]
```

$$10 \quad (63)$$

> $A\_10 := M(n[2]);$

$$A\_10 := \begin{bmatrix} \frac{1}{2} & \frac{1}{4} & \frac{1}{6} & \frac{1}{8} & \frac{1}{10} & \frac{1}{12} & \frac{1}{14} & \frac{1}{16} & \frac{1}{18} & \frac{1}{20} \\ \frac{1}{3} & \frac{1}{5} & \frac{1}{7} & \frac{1}{9} & \frac{1}{11} & \frac{1}{13} & \frac{1}{15} & \frac{1}{17} & \frac{1}{19} & \frac{1}{21} \\ \frac{1}{4} & \frac{1}{6} & \frac{1}{8} & \frac{1}{10} & \frac{1}{12} & \frac{1}{14} & \frac{1}{16} & \frac{1}{18} & \frac{1}{20} & \frac{1}{22} \\ \frac{1}{5} & \frac{1}{7} & \frac{1}{9} & \frac{1}{11} & \frac{1}{13} & \frac{1}{15} & \frac{1}{17} & \frac{1}{19} & \frac{1}{21} & \frac{1}{23} \\ \frac{1}{6} & \frac{1}{8} & \frac{1}{10} & \frac{1}{12} & \frac{1}{14} & \frac{1}{16} & \frac{1}{18} & \frac{1}{20} & \frac{1}{22} & \frac{1}{24} \\ \frac{1}{7} & \frac{1}{9} & \frac{1}{11} & \frac{1}{13} & \frac{1}{15} & \frac{1}{17} & \frac{1}{19} & \frac{1}{21} & \frac{1}{23} & \frac{1}{25} \\ \frac{1}{8} & \frac{1}{10} & \frac{1}{12} & \frac{1}{14} & \frac{1}{16} & \frac{1}{18} & \frac{1}{20} & \frac{1}{22} & \frac{1}{24} & \frac{1}{26} \\ \frac{1}{9} & \frac{1}{11} & \frac{1}{13} & \frac{1}{15} & \frac{1}{17} & \frac{1}{19} & \frac{1}{21} & \frac{1}{23} & \frac{1}{25} & \frac{1}{27} \\ \frac{1}{10} & \frac{1}{12} & \frac{1}{14} & \frac{1}{16} & \frac{1}{18} & \frac{1}{20} & \frac{1}{22} & \frac{1}{24} & \frac{1}{26} & \frac{1}{28} \\ \frac{1}{11} & \frac{1}{13} & \frac{1}{15} & \frac{1}{17} & \frac{1}{19} & \frac{1}{21} & \frac{1}{23} & \frac{1}{25} & \frac{1}{27} & \frac{1}{29} \end{bmatrix} \quad (64)$$

> $x\_10 := X(n[2])$

$$x\_10 := \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \quad (65)$$

> $b\_10 := double(A\_10 \cdot x\_10)$

$$b\_10 := \begin{bmatrix} 1.46448412698413 \\ 1.18087457777860 \\ 1.00993867243867 \\ 0.891019505314834 \\ 0.801605339105339 \\ 0.731019505314834 \\ 0.673400210900211 \\ 0.625199399494729 \\ 0.584114496614497 \\ 0.548571047004307 \end{bmatrix} \quad (66)$$

> $Xc\_10 := double(LinearSolve(Transpose(A\_10).A\_10, Transpose(A\_10).b\_10))$

$$Xc\_10 := \begin{bmatrix} 1.00423747408443 \\ 0.659778324466989 \\ 6.25612644809576 \\ -27.5676101516914 \\ 64.4963052615087 \\ -47.1722599531985 \\ -4.64896429150035 \\ -18.4267525991919 \\ 71.7527161688275 \\ -36.4089124180204 \end{bmatrix} \quad (67)$$

> **n[3]**

$$15 \quad (68)$$

> $A\_15 := M(n[3]);$

$A\_15 :=$ **(69)**

$$\begin{bmatrix}
\frac{1}{2} & \frac{1}{4} & \frac{1}{6} & \frac{1}{8} & \frac{1}{10} & \frac{1}{12} & \frac{1}{14} & \frac{1}{16} & \frac{1}{18} & \frac{1}{20} & \frac{1}{22} & \frac{1}{24} & \frac{1}{26} & \frac{1}{28} & \frac{1}{30} \\
\frac{1}{3} & \frac{1}{5} & \frac{1}{7} & \frac{1}{9} & \frac{1}{11} & \frac{1}{13} & \frac{1}{15} & \frac{1}{17} & \frac{1}{19} & \frac{1}{21} & \frac{1}{23} & \frac{1}{25} & \frac{1}{27} & \frac{1}{29} & \frac{1}{31} \\
\frac{1}{4} & \frac{1}{6} & \frac{1}{8} & \frac{1}{10} & \frac{1}{12} & \frac{1}{14} & \frac{1}{16} & \frac{1}{18} & \frac{1}{20} & \frac{1}{22} & \frac{1}{24} & \frac{1}{26} & \frac{1}{28} & \frac{1}{30} & \frac{1}{32} \\
\frac{1}{5} & \frac{1}{7} & \frac{1}{9} & \frac{1}{11} & \frac{1}{13} & \frac{1}{15} & \frac{1}{17} & \frac{1}{19} & \frac{1}{21} & \frac{1}{23} & \frac{1}{25} & \frac{1}{27} & \frac{1}{29} & \frac{1}{31} & \frac{1}{33} \\
\frac{1}{6} & \frac{1}{8} & \frac{1}{10} & \frac{1}{12} & \frac{1}{14} & \frac{1}{16} & \frac{1}{18} & \frac{1}{20} & \frac{1}{22} & \frac{1}{24} & \frac{1}{26} & \frac{1}{28} & \frac{1}{30} & \frac{1}{32} & \frac{1}{34} \\
\frac{1}{7} & \frac{1}{9} & \frac{1}{11} & \frac{1}{13} & \frac{1}{15} & \frac{1}{17} & \frac{1}{19} & \frac{1}{21} & \frac{1}{23} & \frac{1}{25} & \frac{1}{27} & \frac{1}{29} & \frac{1}{31} & \frac{1}{33} & \frac{1}{35} \\
\frac{1}{8} & \frac{1}{10} & \frac{1}{12} & \frac{1}{14} & \frac{1}{16} & \frac{1}{18} & \frac{1}{20} & \frac{1}{22} & \frac{1}{24} & \frac{1}{26} & \frac{1}{28} & \frac{1}{30} & \frac{1}{32} & \frac{1}{34} & \frac{1}{36} \\
\frac{1}{9} & \frac{1}{11} & \frac{1}{13} & \frac{1}{15} & \frac{1}{17} & \frac{1}{19} & \frac{1}{21} & \frac{1}{23} & \frac{1}{25} & \frac{1}{27} & \frac{1}{29} & \frac{1}{31} & \frac{1}{33} & \frac{1}{35} & \frac{1}{37} \\
\frac{1}{10} & \frac{1}{12} & \frac{1}{14} & \frac{1}{16} & \frac{1}{18} & \frac{1}{20} & \frac{1}{22} & \frac{1}{24} & \frac{1}{26} & \frac{1}{28} & \frac{1}{30} & \frac{1}{32} & \frac{1}{34} & \frac{1}{36} & \frac{1}{38} \\
\frac{1}{11} & \frac{1}{13} & \frac{1}{15} & \frac{1}{17} & \frac{1}{19} & \frac{1}{21} & \frac{1}{23} & \frac{1}{25} & \frac{1}{27} & \frac{1}{29} & \frac{1}{31} & \frac{1}{33} & \frac{1}{35} & \frac{1}{37} & \frac{1}{39} \\
\frac{1}{12} & \frac{1}{14} & \frac{1}{16} & \frac{1}{18} & \frac{1}{20} & \frac{1}{22} & \frac{1}{24} & \frac{1}{26} & \frac{1}{28} & \frac{1}{30} & \frac{1}{32} & \frac{1}{34} & \frac{1}{36} & \frac{1}{38} & \frac{1}{40} \\
\frac{1}{13} & \frac{1}{15} & \frac{1}{17} & \frac{1}{19} & \frac{1}{21} & \frac{1}{23} & \frac{1}{25} & \frac{1}{27} & \frac{1}{29} & \frac{1}{31} & \frac{1}{33} & \frac{1}{35} & \frac{1}{37} & \frac{1}{39} & \frac{1}{41} \\
\frac{1}{14} & \frac{1}{16} & \frac{1}{18} & \frac{1}{20} & \frac{1}{22} & \frac{1}{24} & \frac{1}{26} & \frac{1}{28} & \frac{1}{30} & \frac{1}{32} & \frac{1}{34} & \frac{1}{36} & \frac{1}{38} & \frac{1}{40} & \frac{1}{42} \\
\frac{1}{15} & \frac{1}{17} & \frac{1}{19} & \frac{1}{21} & \frac{1}{23} & \frac{1}{25} & \frac{1}{27} & \frac{1}{29} & \frac{1}{31} & \frac{1}{33} & \frac{1}{35} & \frac{1}{37} & \frac{1}{39} & \frac{1}{41} & \frac{1}{43} \\
\frac{1}{16} & \frac{1}{18} & \frac{1}{20} & \frac{1}{22} & \frac{1}{24} & \frac{1}{26} & \frac{1}{28} & \frac{1}{30} & \frac{1}{32} & \frac{1}{34} & \frac{1}{36} & \frac{1}{38} & \frac{1}{40} & \frac{1}{42} & \frac{1}{44}
\end{bmatrix}$$

> $x\_15 := X(n[3])$

$$x\_15 := \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \tag{70}$$

> $b\_15 := double(A\_15 \cdot x\_15)$

$$b\_15 := \begin{bmatrix} 1.65911449661450 \\ 1.36813069882202 \\ 1.19036449661450 \\ 1.06510039579172 \\ 0.969776261320379 \\ 0.893671824363149 \\ 0.830887372431490 \\ 0.777841708533033 \\ 0.732203161905174 \\ 0.692371623062948 \\ 0.657203161905174 \\ 0.625852776056296 \\ 0.597679352381365 \\ 0.572185513086707 \\ 0.548978053680066 \end{bmatrix} \tag{71}$$

> $Xc\_15 := double(LinearSolve(Transpose(A\_15).A\_15, Transpose(A\_15).b\_15))$

$$Xc\_15 := \begin{bmatrix} 0.998731519878267 \\ 1.10886847155932 \\ -0.784935890655999 \\ 10.9580411415985 \\ -19.1138712586410 \\ 2.44522998431929 \\ 41.7914612958163 \\ -42.2779372012657 \\ 17.6877502518298 \\ 21.3585181191549 \\ -75.2593893385501 \\ 60.9398133357623 \\ -17.2268431580972 \\ 38.1198501657362 \\ -25.7767440239996 \end{bmatrix} \quad (72)$$

****

(c)

Find the infnity norm of the forward error ||x- xc||∞, the backward error ||b -Axc||∞, and the error magnifcation factor for n = 5; 10; 15.

****

> $ForwardError\_5\_Infinity = Norm(x\_5 - Xc\_5, \text{infinity})$

$$ForwardError\_5\_Infinity = 0.000025275359700538402 3 \quad (73)$$

> $BackwardError\_5\_infinity := Norm(b\_5 - A\_5 \cdot Xc\_5, \text{infinity})$

$$BackwardError\_5\_infinity := 1.90433224744879226\ 10^{-11} \quad (74)$$

> $ErrorMagnificationFactor\_5 := \dfrac{\dfrac{Norm(x\_5 - Xc\_5, \text{infinity})}{Norm(x\_5, \text{infinity})}}{\dfrac{Norm(b\_5 - A\_5 \cdot Xc\_5, \text{infinity})}{Norm(b\_5, \text{infinity})}}$

$$ErrorMagnificationFactor\_5 := 1.515283676\ 10^{6} \quad (75)$$

****

> $ForwardError\_10\_Infinity = Norm(x\_10 - Xc\_10, \text{infinity})$

$$ForwardError\_10\_Infinity = 70.7527161688275186 \quad (76)$$

> $BackwardError\_10\_infinity := Norm(b\_10 - A\_10 \cdot Xc\_10, \text{infinity})$

$$BackwardError\_10\_infinity := 2.56394261377579369\ 10^{-8} \quad (77)$$

> $ErrorMagnificationFactor\_10 := \dfrac{\dfrac{Norm(x\_10 - Xc\_10, \text{infinity})}{Norm(x\_10, \text{infinity})}}{\dfrac{Norm(b\_10 - A\_10 \cdot Xc\_10, \text{infinity})}{Norm(b\_10, \text{infinity})}}$

$$ErrorMagnificationFactor\_10 := 4.041285058\ 10^{9} \tag{78}$$

****

> $ForwardError\_15\_Infinity = Norm(x\_15 - Xc\_15, \text{infinity})$

$$ForwardError\_15\_Infinity = 76.2593893385501360 \tag{79}$$

> $BackwardError\_15\_infinity := Norm(b\_15 - A\_15 \cdot Xc\_15, \text{infinity})$

$$BackwardError\_15\_infinity := 6.32045549231463610\ 10^{-9} \tag{80}$$

> $ErrorMagnificationFactor\_15 := \dfrac{\dfrac{Norm(x\_15 - Xc\_15, \text{infinity})}{Norm(x\_15, \text{infinity})}}{\dfrac{Norm(b\_15 - A\_15 \cdot Xc\_15, \text{infinity})}{Norm(b\_15, \text{infinity})}}$

$$ErrorMagnificationFactor\_15 := 2.001802853\ 10^{10} \tag{81}$$

****

(d)

Find the condition number of A for n = 5; 10; 15 and compare it with error magnifcation factor obtained in part (c).

****

> $\boldsymbol{CondititionNumber\_5 := double((ConditionNumber(A\_5)))}$

$$ConditionNumber\_5 := 1.97906775000000000\ 10^{6} \tag{82}$$

> $\boldsymbol{ConditionNumber\_10} := double((\boldsymbol{ConditionNumber(A\_10)}))$

$$ConditionNumber\_10 := 1.31334652072236797\ 10^{14} \tag{83}$$

> $\boldsymbol{ConditionNumber\_15 := double(ConditionNumber(A\_15))}$

$$ConditionNumber\_15 := 9.28569976203256739\ 10^{21} \tag{84}$$

> $\boldsymbol{\dfrac{ConditionNumber\_5}{ErrorMagnificationFactor\_5}}$

$$1.306070791 \tag{85}$$

> $\boldsymbol{\dfrac{ConditionNumber\_10}{ErrorMagnificationFactor\_10}}$

$$32498.24009 \tag{86}$$

> $\boldsymbol{\dfrac{ConditionNumber\_15}{ErrorMagnificationFactor\_15}}$

$$4.638668462\ 10^{11} \tag{87}$$

****

Answer:

The condition Number is the maximum possible error magnification factor
As we can see in all this 3 examples EMF < Condition number as expected.

n=5 :

$\mathbf{\mathit{ConditionNumber\_5}} := \mathbf{1.9791 \cdot 10^{6}}$ $\mathit{ErrorMagnificationFactor\_5} := \mathbf{1.5153 \cdot 10^{6}}$

for n=5 EMF and Condition Number have the same order O(1)

n=10 :

$\mathbf{\mathit{ConditionNumber\_10}} := \mathbf{1.3133 \cdot 10^{14}} >> \mathit{ErrorMagnificationFactor\_10} :=$
$\mathbf{4.041285058\ 10^{9}}$

for n=10 Condition Number vs EMF = O(10^5)

n=15 :

$\mathbf{\mathit{ConditionNumber\_15}} := \mathbf{9.2857 \cdot 10^{21}} >> \mathit{ErrorMagnificationFactor\_15} :=$
$\mathbf{2.001802853\ 10^{10}}$

for n=15 Condition Number vs EMF = O(10^9)

with n-=>∞ the condition number =>∞.This matrix is ill conditioned. almost singular, and the computation of its inverse, or solution of a linear system of equations is prone to large numerical errors

(e)

For what value of n does the solution Xc of the linear system start to have no correct signifcant digits?

****

Answer: The answer is different for different algorithms. For this solution :

For value of n=5 condidion number = $2*10^{6}$

For double presision we expect to loose **6** significant digits, we will be left with **16-6=10** significant digits

For value of n=10 condidion number = $1.3*10^{14}$

For double presision we expect to loose **14** significant digits, we will be left with **16-14=2** significant digits

> $\mathit{ConditionNumber}(\mathit{double}(M(11)))$

$$\mathbf{4.804517656\ 10^{15}} \tag{88}$$

> $\mathit{ConditionNumber}(\mathit{double}(M(12)))$

$$\mathbf{1.202196980\ 10^{17}} \tag{89}$$

For value of n=11 condidion number = $4.8*10^{15}$

For double presision we expect to loose **15** significant digits, we wil have left with **16-15=1** significant digits

For value of n=12 condidion number = $1.8*10^{17}$

For double presision we expect to loose **17** significant digits, we wil have left with **16-17= -1** significant digits

Start with n=12 the solution Xc of the linear system start to have no correct signifcant digits

>

##

5.

(10 pts) Design a fancy letter symbol representing your last name initial, using at least four Beezier curves joined together.

##

```
> with(plots) :
> bezier :=proc(param)
            local x1, x2, x3, x4, y1, y2, y3, y4, bx, cx, bvy, cy, fx, fy, dx, dy;
            x1 := param[1, 1]; y1 := param[1, 2];
            x2 := param[2, 1]; y2 := param[2, 2];
            x3 := param[3, 1]; y3 := param[3, 2];
            x4 := param[4, 1]; y4 := param[4, 2];
            bx := 3 * (x2-x1); bvy := 3 * (y2-y1);
            cx := 3 * (x3-x2)-bx; cy := 3 * (y3-y2) - bvy;
            dx := x4 - x1 - bx - cx; dy := y4 - y1 - bvy - cy;
            fx := proc (t) options operator, arrow; x1 + bx * t + cx * t^2 + dx * t
    ^3 end proc;
            fy := proc (t) options operator, arrow; y1 + bvy * t + cy * t^2 + dy * t
    ^3 end proc;
            plot([fx(t), fy(t), t = 0 ..1]);
        end proc:
```

> $data1 := [[-4.0, 3.0], [-4.0, 2.0], [-4.0, 4.0], [-3.5, 3.5]];$

$$data1 := [[-4.0, 3.0], [-4.0, 2.0], [-4.0, 4.0], [-3.5, 3.5]] \quad \textbf{(90)}$$

> $data2 := [[-3.5, 3.5], [-3.0, -3.5], [-1.55, 1.83], [-1.5, 3.25]]$

$$data2 := [[-3.5, 3.5], [-3.0, -3.5], [-1.55, 1.83], [-1.5, 3.25]] \quad \textbf{(91)}$$

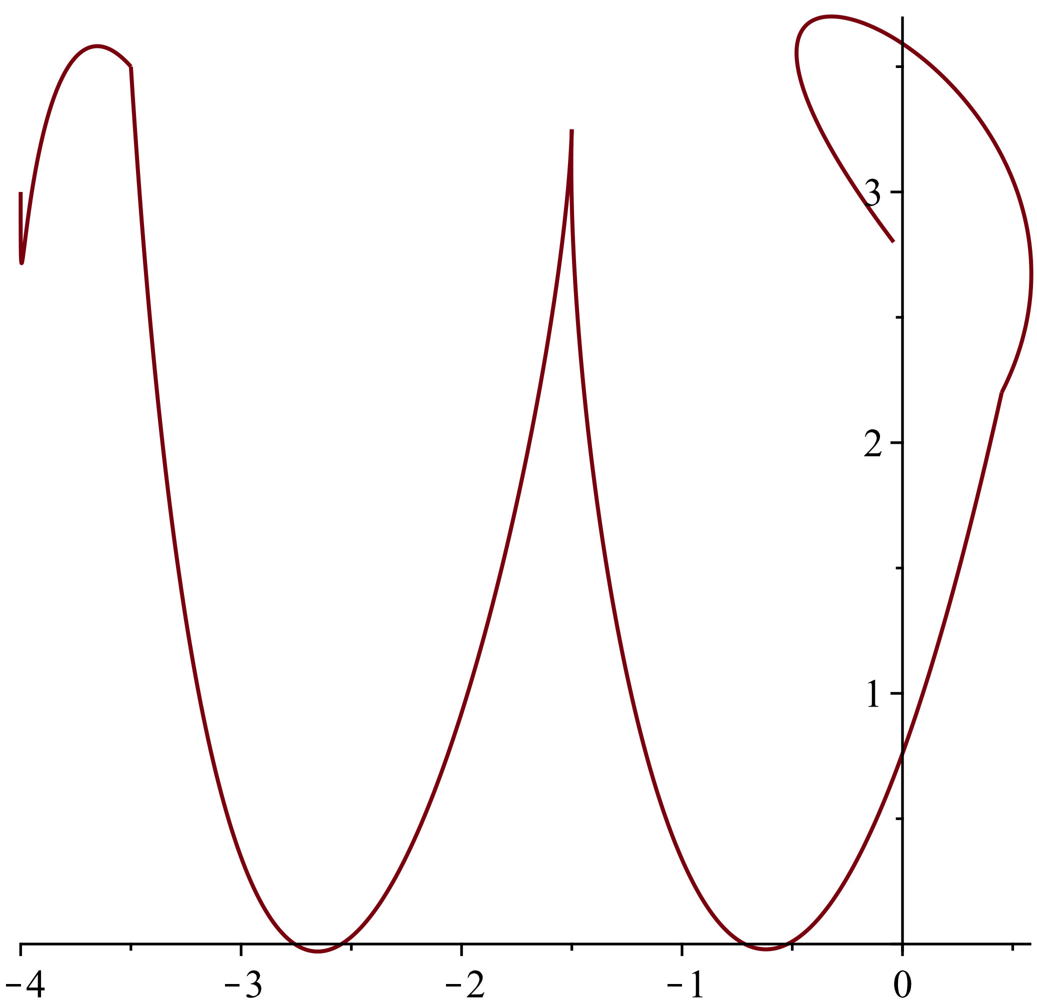
> $data3 := [[-1.5, 3.25], [-1.55, 1.83], [-0.79, -2.7], [0.45, 2.2]];$

$$data3 := [[-1.5, 3.25], [-1.55, 1.83], [-0.79, -2.7], [0.45, 2.2]] \quad \textbf{(92)}$$

> $data4 := [[0.45, 2.2], [1.2, 3.5], [-1.5, 4.5], [-0.04, 2.80]]$

$$data4 := [[0.45, 2.2], [1.2, 3.5], [-1.5, 4.5], [-0.04, 2.80]] \quad \textbf{(93)}$$

> $display(bezier(\mathrm{data1}), bezier(\mathrm{data2}), (bezier(\mathrm{data3}), bezier(\mathrm{data4})))$



****
Curve 1
****
data1 := [[-4.0, 3.0], [-4.0, 2.0], [-4.0, 4.0], [-3.5, 3.5]];

bx=3*(x2-x1) = 3*(-4+4)= 0
cx=3*(x3-x2) = 3*(-4+4)= 0
dx = x4-x1-bx-cx= -3.5+4 - 0 - 0= 0.5

by=3*(y2-y1)=3*(2-3)= -3
cy=3*(y3-y2)=3*(4-2)= 6

****
X1(t) = -4 + $0.5t^3$
Y1(t) = 3 - 3t + $6t^2$ $-2.5\,t^3$
****
Curve 2
****
data2 := [[-3.5, 3.5], [-3.0, -3.5], [-1.55, 1.83], [-1.5, 3.25]]

bx=3*(x2-x1) = 3*(-3.0+3.5) = 1.5

cx=3*(x3-x2) = 3*(-1.55+3)= 4.35
dx = x4-x1-bx-cx= -1.5+3.5-4.35-1.5= -3.85

by=3*(y2-y1)=3*(-3.5-3.5)= -21
cy=3*(y3-y2)=3*(1.83+3.5)= 15.99

****

X1(t) = -3.5 + 1.5t + **4.35** t^2 **- 3.85** t^3
Y1(t) = 3.5 - 21t + 15.99 t^2 **- 4.76** t^3

****

Curve 3

****

data3 := [[-1.5, 3.25], [-1.55, 1.83], [-0.79, -2.7], [0.45, 2.2]]

bx=3*(x2-x1) = 3*(-1.55+1.5) = -0.15
cx=3*(x3-x2) = 3*(-0.79+1.55) = 2.28
dx = x4-x1-bx-cx= 0.45 +1.5 - 2.28 + 0.15= -0.18

by=3*(y2-y1)=3*(1.83-3.25) = -4.26
cy=3*(y3-y2)=3*(-2.7-1.83) = -13.59
dy= y4-y1-by-cy = 2.2 + 1.5 + 4.26 +13.59 = 16.8

X1(t) = -1.5 - 0.15t + **2.28** t^2 **- 0.18** t^3
Y1(t) = 3.25 - 4.26t - 13.59 t^2 **+ 16.8** t^3

****

Curve 4

****

data4 := [[0.45, 2.2], [1.2, 3.5], [-1.5, 4.5], [-0.04, 2.80]]

bx=3*(x2-x1) = 3*(1.2-0.45) = 2.25
cx=3*(x3-x2) = 3*(-1.5-1.2) = -8.1
dx = x4-x1-bx-cx= -0.04-0.45-2.25+8.1= 5.36

by=3*(y2-y1)=3*(3.5-2.2) = 3.9
cy=3*(y3-y2)=3*(4.5-3.5) = 3
dy= y4-y1-by-cy = 2.8-2.2-3.9-3= -6.3

X1(t) = 0.45 +2.25t -8.1 t^2 **+ 5.36** t^3
Y1(t) = 2.2 + 3.9t + 3 t^2 **-6.3** t^3

****

>

>