University of Waterloo

ECE204 Lab Report

Simulation Assignment #5

Section: 202

Kehan Xing (k5xing)

Jing Liu (j762liu)

Submission Date: 2019-11-26

MATLAB Code:

```
%Loading the test file to X
X = load('test11.txt');
%Determine whether Derivative of Integration
method = input('Select Derivative(1) or Integration(2): ');

if(method == 1)
    %take point p
    p = input('The point you want to perform the derivative: ');

    %check if p exceeds data limit
    if(p >= max(X(:,1)) || p <= min(X(:,1)))
         disp('Error: exceeding the data limit');
    %p1 or p2 found in data set
    elseif(ismember(p,X(:,1)) == 1)
        %find the index of p
        index = find(X(:,1)== p);

        %not even space
        if(X(index, 1) - X(index-1, 1) ~= X(index+1, 1) - X(index, 1))

            %polynomial regression
            [xa,Ypoly,polyrsqr, degree] = polynomial(X);
            deltaX = min(diff(X(:,1)));
            Y1=Plug(p-deltaX,xa,degree); Y2=Plug(p+deltaX,xa,degree);

            %calculate CDD and display
            d=CDD(deltaX,Y1,Y2);
            disp(['Derivative is ',num2str(d)]);
            display(['Degree: ', num2str(degree)]);

            %display coeff of polynomial
            k = 1;
            while k <= degree+1
                display(['a',num2str(k-1),': ', num2str(xa(k,1))]);
                k = k+1;
            end

            plotPoly(X,Ypoly,polyrsqr);

        %the point is from the data set and the spacing between the points is
even
        else
            deltaX =X(index, 1) - X(index-1, 1);
             Y1=X(index-1, 2);
             Y2=X(index+1, 2);
             d=CDD(deltaX,Y1,Y2);
             disp(['Derivative is ',num2str(d)]);

        end

    %p1 or p2 not found in data set
    else
```

```
        %polynomial regression
        [xa,Ypoly,polyrsqr, degree] = polynomial(X);
        deltaX = min(diff(X(:,1)));
        Y1=Plug(p-deltaX,xa,degree);
        Y2=Plug(p+deltaX,xa,degree);

        %calculate CDD and display
        d=CDD(deltaX,Y1,Y2);
        disp(['Derivative is ',num2str(d)]);
        display(['Degree: ', num2str(degree)]);

        %display coeff of polynomial
        k = 1;
        while k <= degree+1
            display(['a',num2str(k-1),': ', num2str(xa(k,1))]);
            k = k+1;
        end
        plotPoly(X,Ypoly,polyrsqr);

    end



elseif(method == 2)

    error = 0;
    p1 = input('The lower limit: '); p2 = input('The upper limit: ');

    %error check
    if(p2 <= p1)
        error = 1;
        disp('Error: incorrect upper and lower limit');
    end

    n = input('Number of segments: ');

    if(n <= 0 )
        error = 1;
        disp('Error: incorrect segments');
    end

    if(error == 0)
        %show error if p1 or p2 exceeds data limit
        if(p2 > max(X(:,1)) || p1 < min(X(:,1)))
             disp('Error: exceeding the data limit');
        %p1 or p2 are within the data limit
        elseif((ismember(p1, X(:,1)) == 1) && (ismember(p2, X(:,1)) == 1))

            %find the index of p
            index1 = find(X(:,1)== p1); index2 = find(X(:,1)== p2);
            h = (p2 - p1)/n;
            k = 0; notFound = 0;

            %check if all segments are from data set
```

```
            %-----------------------------------------
            while(k < n)
                p = p1+k*h;
                if(ismember(p, X(:,1)) == 0)
                    notFound = 1;
                    break;
                end
                k = k+1;
            end
            %-----------------------------------------

            checkSpacing = diff(X(index1:index2, 1));

            %even spacing
            %-----------------------------------------
            if(max(checkSpacing) == min(checkSpacing))
                evenSpacing = 1;
            else
                evenSpacing = 0;
            end
            %-----------------------------------------

            %the limits are from the data set and the spacing between the
points is even
            if(evenSpacing == 1 && notFound == 0)
                i = 1;
                tSum = 0;
                while (i < n)
                    tindex=find(X(:,1)== p1+i*h);
                    tSum = tSum+X(tindex,2);
                    i=i+1;
                end
                integral=(h/2)*(X(index1,2)+2*tSum+X(index2,2));
                disp(['Integral is ',num2str(integral)]);

            %polynomial regression
            else
                [xa,Ypoly,polyrsqr, degree] = polynomial(X);

                i = 1;
                tSum = 0;

                %partial sum btw p1 and p2
                while (i < n)
                    tSum = tSum+Plug(p1+i*h,xa,degree);
                    i=i+1;
                end

                %calculate integral and display

integral=(h/2)*(Plug(p1,xa,degree)+2*tSum+Plug(p2,xa,degree));
                disp(['Integral is ',num2str(integral)]);
                plotPoly(X,Ypoly,polyrsqr);
            end
```

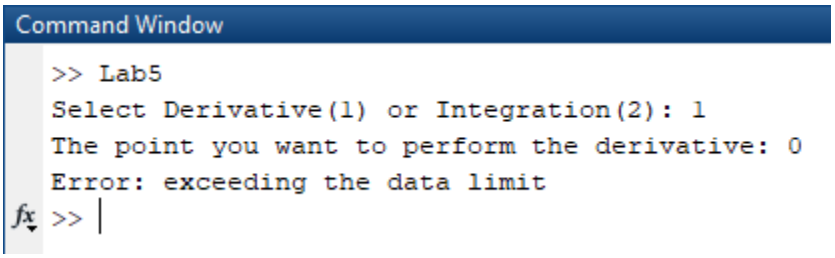
```
        %p1 or p2 not found in data set
        else
                h = (p2 - p1)/n;
                [xa,Ypoly,polyrsqr, degree] = polynomial(X);

                i = 1;
                tSum = 0;

                %partial sum btw p1 and p2
                while (i < n)
                    tSum = tSum+Plug(p1+i*h,xa,degree);
                    i=i+1;
                end

                %calculate integral and display

integral=(h/2)*(Plug(p1,xa,degree)+2*tSum+Plug(p2,xa,degree));
                disp(['Integral is ',num2str(integral)]);
                plotPoly(X,Ypoly,polyrsqr);
        end
    end
end

%CDD calculation
function d = CDD(deltaX,Y1,Y2)
    d=(Y2-Y1)/(deltaX*2);
end

%Polynomial fit
function [xa,Ypoly,polyrsqr, degree] =polynomial(X)
    polyrsqr=2;
    prevrsqr=1;
    degree=0;
    while(abs((polyrsqr-prevrsqr))>0.01)
        degree=degree+1;
        prevrsqr=polyrsqr;
        xpower = zeros(length(X), degree+1);
        j = 1;

        %Matrix which contains the powers of x
        while j <= degree*2
            i = 1;
            while i <= length(X)
                xpower(i,j) = power(X(i,1), j);
                i = i+1;
            end
            j = j+1;
        end

        xleft = zeros(degree+1, degree+1);
        offset = -1;
        column = 1;

        %Matrix on the left side of coefficients
        while column <= degree+1
```

```
            row = 1;
            %set up the first column
            if row == 1 && column == 1
                xleft(row,column) = length(X);
                row = row + 1;

                while row <= degree + 1
                      xleft(row, column) = sum(xpower(:,row-1));
                      row = row + 1;
                end
            %set up the rest of the columns
            else
                while row <= degree + 1
                    xleft(row,column) = sum(xpower(:,row+offset));
                    row = row + 1;
                end
            end
            column = column + 1;
            offset = offset + 1;
        end

        %Matrix on the right side of coefficients
        xright = zeros(degree+1, 1);
        row = 1;
        while row <= degree+1
            %xright(row,1) = sum(xpower(:,row).*X(:, 2));
            xright(row,1) = sum(power(X(:,1),row-1).*X(:, 2));
            row = row+1;
        end

        %Coefficient matrix
        xa =xleft\xright;
        Ypoly = 0;
        d = 1;
        while d <= degree+1
            Ypoly = Ypoly + xa(d,1)*power(X(:,1),d-1);
            d = d +1;
        end

    %Calculate R^2
    Ybar=sum(X(:,2))/length(X);
    Stprep= Ypoly - Ybar;
    St=sum(Stprep.*Stprep);
    Srprep=X(:,2)-Ypoly;
    Sr=sum(Srprep.*Srprep);
    polyrsqr=(St-Sr)/St;

    end
end
function result=Plug(x,xa,degree)
    index=1;
    result=0;
    while(index <= degree+1)
        result = result + xa(index, 1)*power(x, index - 1);
        index=index+1;
    end
```

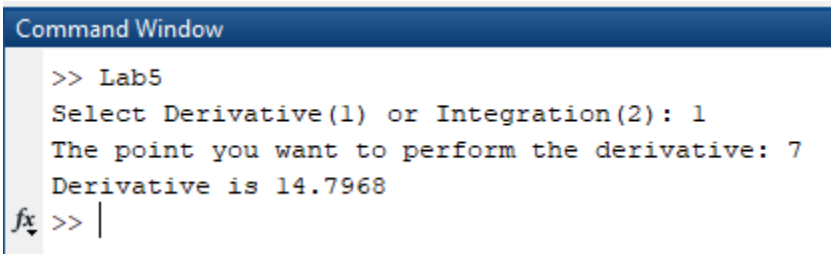
```
end

%plot data and polynomial
function plotPoly(X,Ypoly,polyrsqr)

    plot(X(:,1), X(:,2),  X(:,1), Ypoly);
    grid on;
    xlabel('x'); ylabel('y');
    xt = max(X(:,1))*0.1;
    yt = max(X(:,2))*0.8;
    caption = sprintf('R^2 = %f', polyrsqr);
    text(xt,yt, caption, 'FontSize', 12, 'Color', 'black', 'FontWeight',
'bold');
end
```

Derivative (test11.txt)

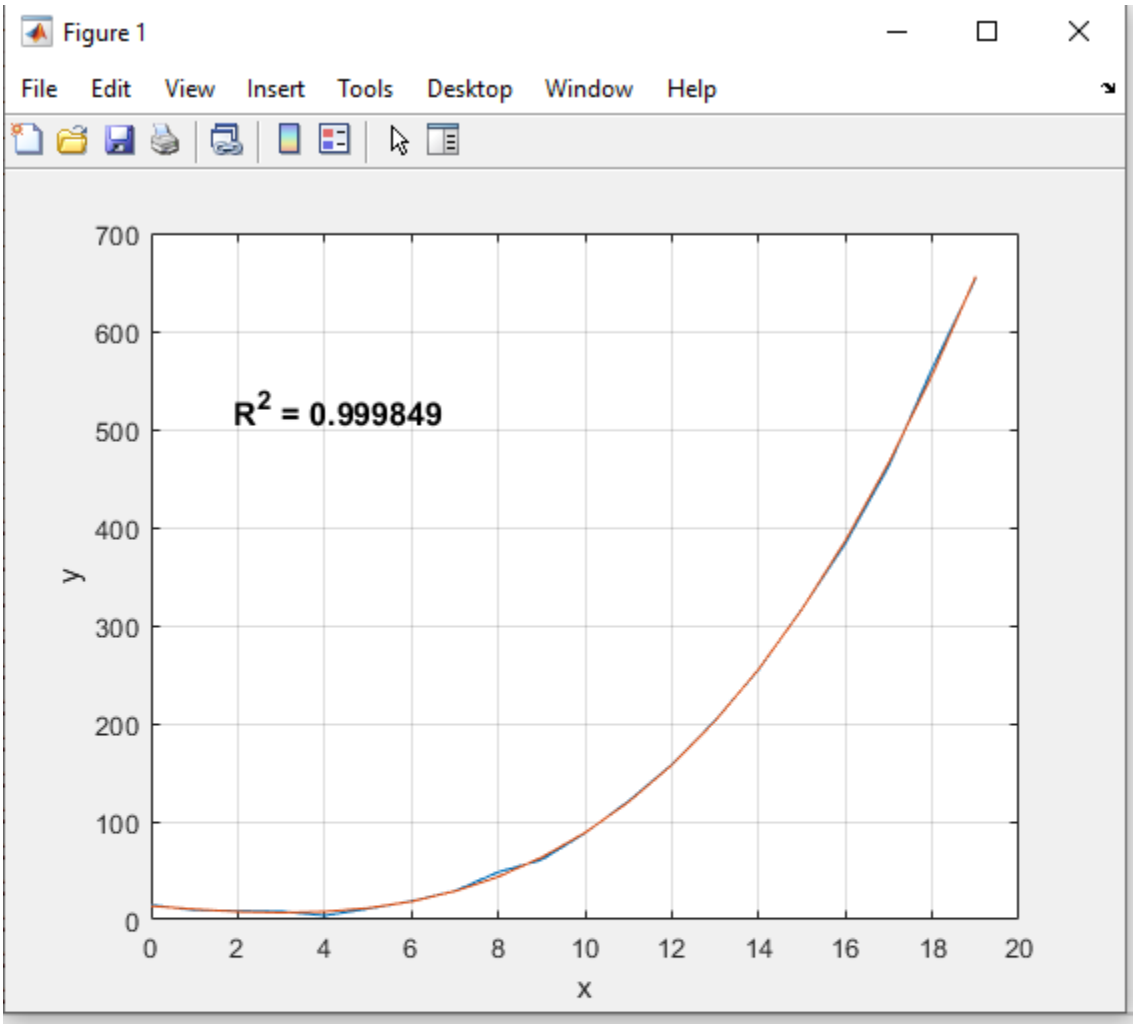
- p = 0



- p = 7



- p = 18.5

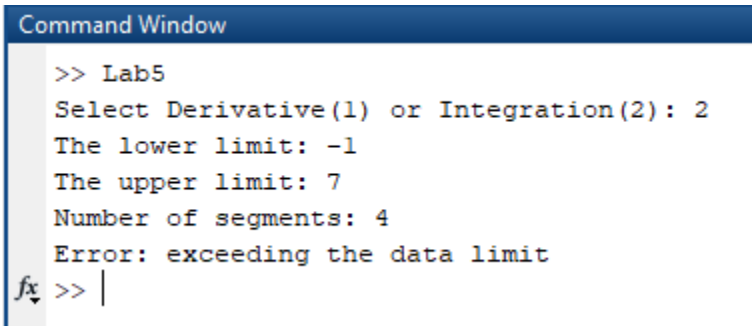


Command Window

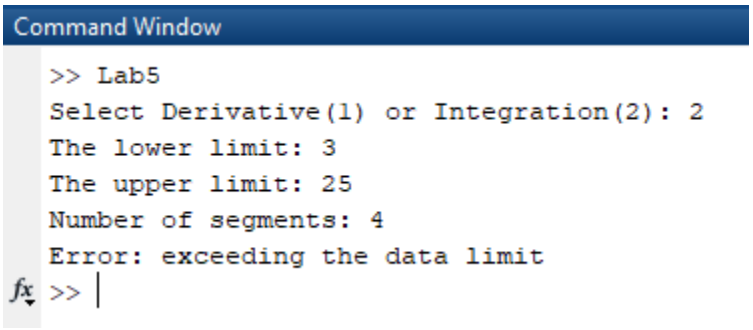
```
>> Lab5
Select Derivative(1) or Integration(2): 1
The point you want to perform the derivative: 18.5
Derivative is 100.2263
Degree: 3
a0: 13.9321
a1: -3.5676
a2: 0.14668
a3: 0.095711
```

Integration (test11.txt)

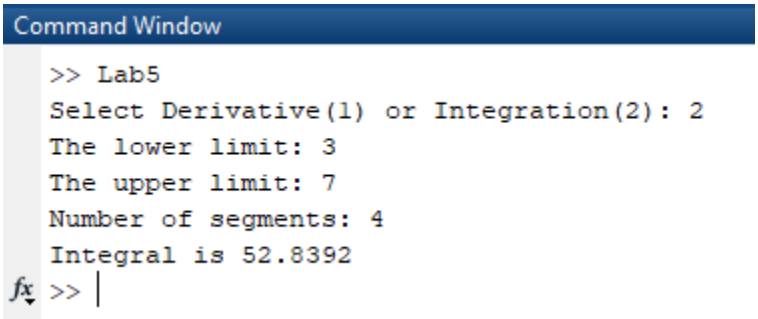
- p1 = -1, p2 = 7, n = 4



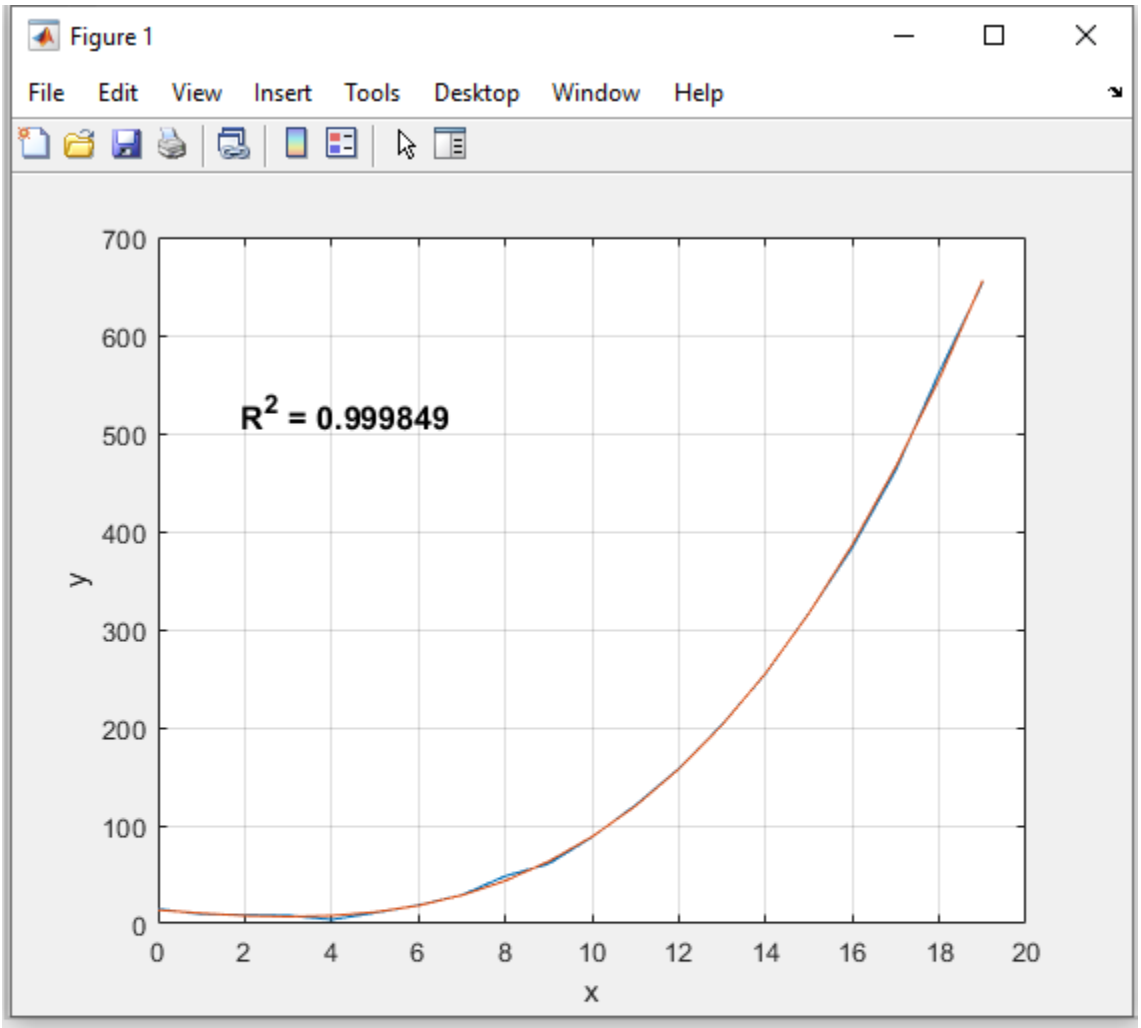
- p1 = 3, p2 = 25, n = 4



- p1 = 3, p2 = 7, n = 4



- p1 = 3, p2 = 7, n = 10



Command Window

```
>> Lab5
Select Derivative(1) or Integration(2): 2
The lower limit: 3
The upper limit: 7
Number of segments: 10
Integral is 55.5082
>>
```