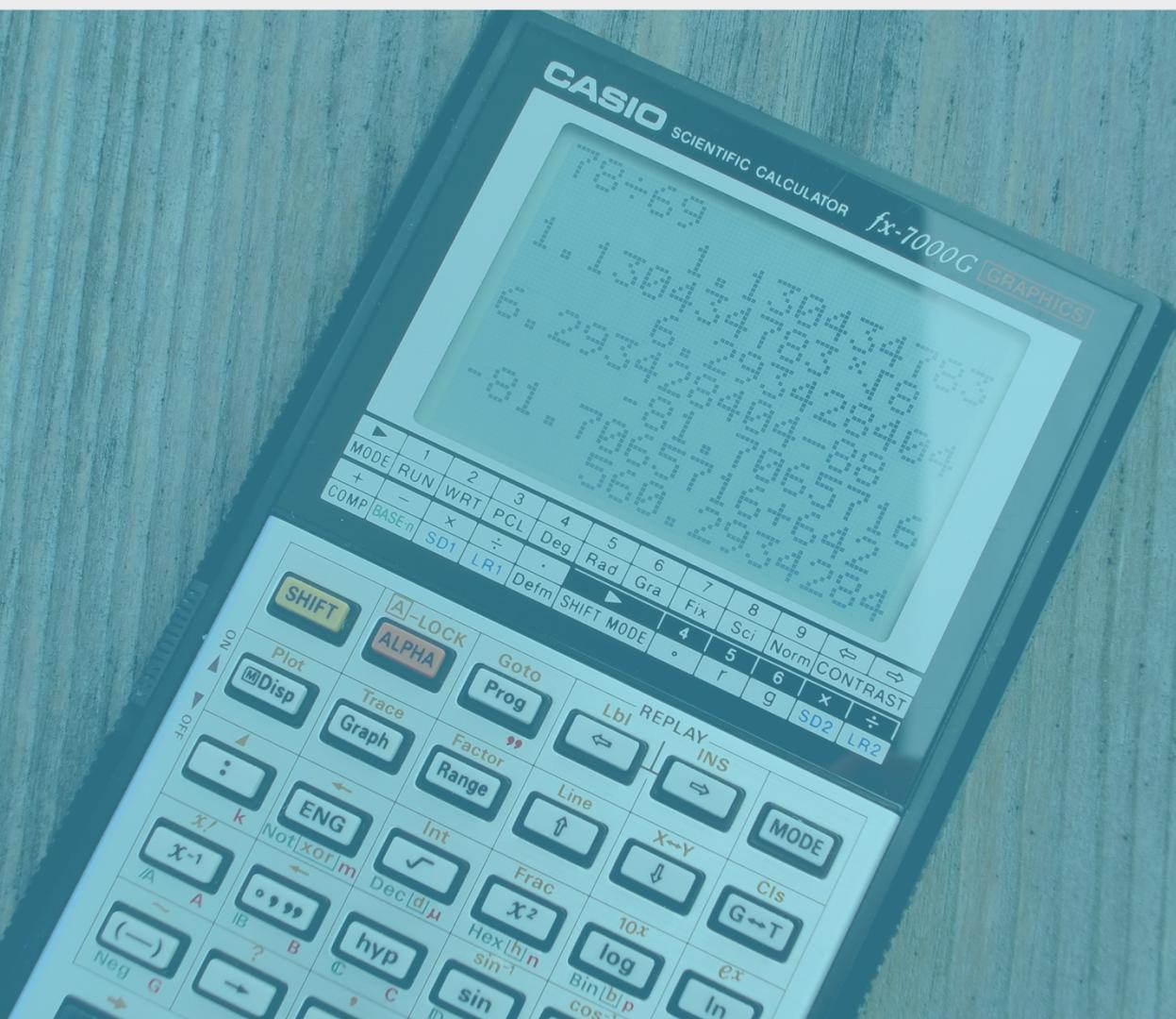




تمرین کامپیوتری دوم

محاسبات عددی دکتر آریانیان



۱۴۰۲ بهار

مجید صادقی نژاد

سوال اول

توضیحات:

تئوری:

مقدمه (خطوط ۱ تا ۱۳) :

خط ۱: تمامی پنجره های قبلی بسته شوند تمامی داده های قبلی پاک شوند و صفحه command نیز پاک شود.

خط ۵: p به صورت متغیر جبری تعریف می شود تا از آن به عنوان متغیر مستقل مسئله یعنی همان قیمت استفاده شود همچین (۱) و (۲) q به صورت توابع جبری تعریف می شوند تا در ادامه منحنی های برآشش شده داخل آن ها قرار داده شود.

خط ۶: متغیر های a,b,c نیز به صورت جبری تعریف می شوند زیرا برای یافتن ضرایب منحنی برآشش شده به حل دستگاه معادلات نیز داریم و طبیعتنا نیاز به متغیر جبری داریم.

خط ۸: جدول صورت سوال در یک ماتریس دو ردیفه تعریف شده است بدین معنی که هر یک از ستون های جدول سوال به یک ردیف ماتریس تبدیل شده اند که ردیف اول ماتریس مقادیر قیمت و ردیف دوم ماتریس مقادیر تقاضا است.

```
priceDemand=[ 140, 190, 220, 260, 280, 310, 330, 400, 450;
              1200, 970, 845, 650, 550, 480, 380, 80, 0 ]
```

تقاضا مربوط به آن قیمت

قیمت

خط ۱۱: تعداد داده های جدول صورت سوال (تعداد ستون های ماتریس) توسط فانکشن سایز متلب ، مشخص می شود.

خط ۱۲: بزرگترین مقدار قیمت که همان متغیر مستقل مسئله است پیدا می شود تا برای رسم نمودار تقریبی از این که نمودار تا کدام نقطه باید رسم شود داشته باشیم.

```
1 clc, clear, close all
2
3 %--- Information ---
4
5 syms p q1(p) q2(p)
6 syms a b c
7
8 priceDemand=[ 140, 190, 220, 260, 280, 310, 330, 400, 450;
9             1200, 970, 845, 650, 550, 480, 380, 80, 0 ]
10
11 tableSize= size(priceDemand,2);
12 range=max(priceDemand(1,1:tableSize));
```

بخش الف (خطوط ۱۴ تا ۳۴) :

خط ۱۶: با توجه به معادله های مورد نیاز برای برازش منحنی خطی که در بخش تئوری توضیح داده شد اولین معادله را می نویسم در این معادله (eq1) از متغیر های جبری ای که پیش از این تعریف کردیم استفاده شده همچنین از فانکشن sum مطلب به عنوان سیگما استفاده می شود همچنان تعداد داده ها که در بخش مقدمه به دست آوردم نیز در این بخش مورد نیاز است.

خط ۱۷: همانند معادله اول ، معادله دوم را نیز برأساس تئوری و مقادیر جبری که قبلا تعریف کردیم به دست می آوریم

خط ۱۸: با استفاده از فانکشن solve مطلب ، دو معادله دو مجهول جبری ای که در بالا تعریف شد را حل می کنیم به این صورت که دو معادله جبری به همراه متغیر های جبری آن ها را به فانکشن داده و جواب عددی متغیر های a,b را به ترتیب در A,B ذخیره می کنیم.

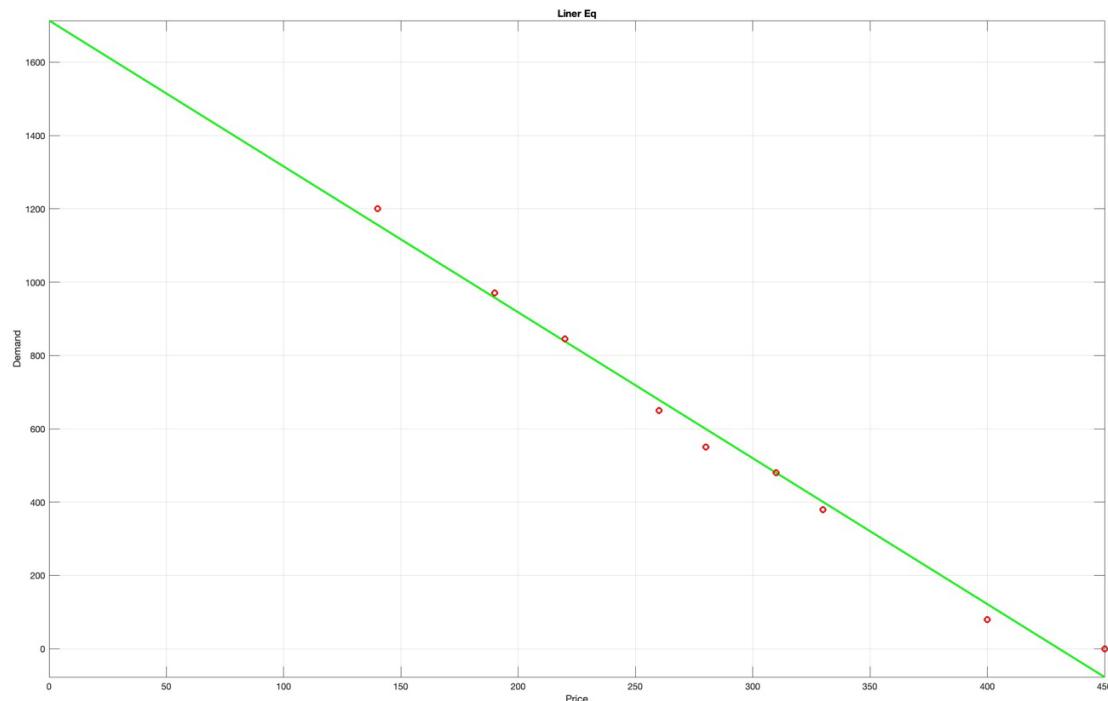
خطوط ۱۹ تا ۲۰: دو متغیر عددی به دست آمده را با استفاده از فانکشن vpa مطلب ، تبدیل به مقادیر اعشاری با دقت بالایی از اعشار می کنیم.

خط ۲۲: تابع جبری q1 را که پیش تر تعریف کردیم با استفاده از مجهولات عددی ای که حساب کردیم و بر حسب متغیر جبری p می نویسم و نمایش می دهیم.

$$q1(p) = 1713.441595 - 3.979059829 p$$

خط برازش شده برای داده های جدول

خطوط ۲۴ تا ۳۱: ابتدا یک پنجره‌ی جدید برای نمایش نمودار خطی و نقاط جدولی باز می کنیم در این پنجره ابتدا با دستور fplot تابع جبری ای که به دست آوردم را رسم می کنیم سپس پنجره را باز نگه می داریم نقاط جدولی را نیز در آن نمایش می دهیم در آخر عنوان نمودار و محور های افقی و عمودی آن را مشخص کرده و نمایش خط کشی نمودار را فعال می کنیم.



خط ۳۳: در نهایت مجموع مجذور خط را حساب کرده و در متغیر error1 نمایش می دهیم همچنین با فانکشن vpa آن را با دقت بالایی از اعشار نمایش می دهیم.

$$\text{error1} = 13488.019943$$

مجموع مجذورات خط برازش شده

```

14 %--- Part A - Liner Eq ---
15
16 eq1 = a*tableSize+b*sum(priceDemand(1,1:tableSize)) == sum(priceDemand(2,1:tableSize));
17 eq2 = a*sum(priceDemand(1,1:tableSize)) + b*sum(priceDemand(1,1:tableSize)).^2 ==
        sum(priceDemand(2,1:tableSize).*priceDemand(1,1:tableSize));
18 [A,B] = solve([eq1,eq2],[a,b]);
19 A=vpa(A);
20 B=vpa(B);
21
22 q1(p)= A+B*p
23
24 f1=figure;
25 fplot(q1,[@,range],'g','LineWidth',2);
26 hold on
27 plot(priceDemand(1,1:tableSize),priceDemand(2,1:tableSize),'o','color','red','LineWidth',2)
28 title('Liner Eq')
29 xlabel('Price')
30 ylabel('Demand')
31 grid on
32
33 error1 = vpa(sum((priceDemand(2,1:tableSize)-q1(priceDemand(1,1:tableSize))).^2),12)

```

بخش ب (خطوط ۳۵ تا ۵۷) :

خطوط ۳۷ تا ۳۹: با توجه به بخش تئوری برای برازش سهمی سه معادله سه مجهول لازم را با توجه به متغیر های جبری ای که پیش از این تعریف کردیم تشکیل می دهیم.

خط ۴۰: با استفاده از فانکشن solve متلب همانند بخش الف این دستگاه معادله را حل می کنیم تا ضرایب سهمی (A,B,C) مشخص شوند.

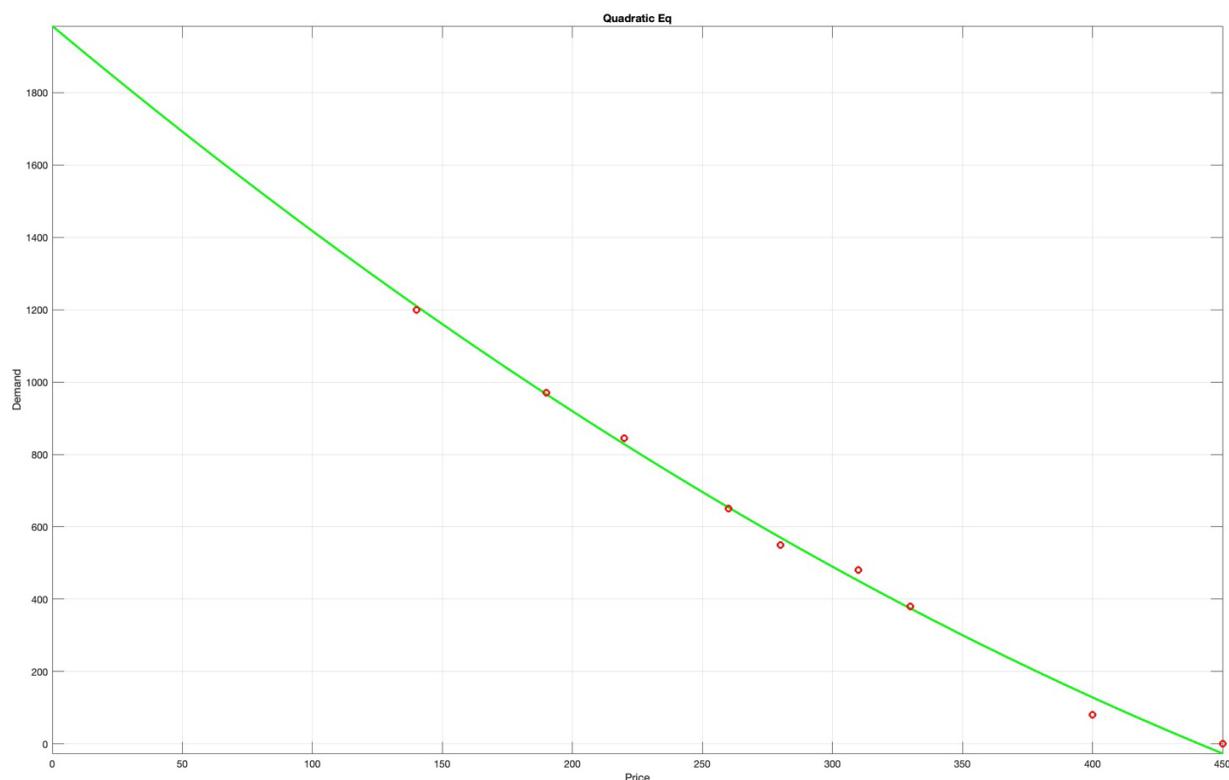
خطوط ۴۱ تا ۴۳: ضرایب به دست آمده را به شکل اعشاری با دقت اعشار بالا تبدیل می کنیم

خط ۴۵: معادله ی جبری سهمی برازش شده را نمایش می دهیم.

$$q_2(p) = 0.003419859126 p^2 - 6.007007061 p + 1984.111266$$

معادله ی سهمی برازش شده

خطوط ۴۷ تا ۵۴: دقیقا همانند بخش الف این بار تابع q_2 و نقاط جدولی را در یک پنجره ی جدید رسم می کنیم لازم به ذکر است که محدوده رسم نمودار از صفر تا بزرگترین قیمت جدول صورت سوال است.



خط ۵۶: مجددا همانند بخش الف مجموع مجذور خطها محاسبه شده و به شکل اعشاری نمایش داده می شود.

$$\text{erorr2} = 4778.76895922$$

مجموع مجذورات خط

```

35 %--- Part B - Quadratic Eq ---
36
37 eq1 = a*tableSize+b*sum(priceDemand(1,1:tableSize))+ c*sum(priceDemand(1,1:tableSize).^2) ==
        sum(priceDemand(2,1:tableSize)) ;
38 eq2 = a*sum(priceDemand(1,1:tableSize)) + b*sum(priceDemand(1,1:tableSize).^2)+c*
        c*sum(priceDemand(1,1:tableSize).^3) == sum(priceDemand(2,1:tableSize).*priceDemand(1,1:tableSize));
39 eq3 = a*sum(priceDemand(1,1:tableSize).^2) + b*sum(priceDemand(1,1:tableSize).^3)+c*
        c*sum(priceDemand(1,1:tableSize).^4) == sum(priceDemand(2,1:tableSize).*(priceDemand(1,1:tableSize).^2));
40 [A,B,C] = solve([eq1,eq2,eq3],[a,b,c]);
41 A=vpa(A);
42 B=vpa(B);
43 C=vpa(C);
44
45 q2(p)=A+B*p+C*(p^2)
46
47 f2=figure;
48 fplot(q2(p),[0,range],'g','LineWidth',2);
49 hold on
50 plot(priceDemand(1,1:tableSize),priceDemand(2,1:tableSize),'o','color','red','LineWidth',2)
51 title('Quadratic Eq')
52 xlabel('Price')
53 ylabel('Demand')
54 grid on
55
56 error2 = vpa(sum((priceDemand(2,1:tableSize)-q2(priceDemand(1,1:tableSize))).^2),12)

```

بخش ج (خطوط ۵۸ تا ۶۹) :

خط ۶۰: متغیر maxError جهت مشخص کردن حداقل مقدار خطای منحنی نسبت به جدول بر حسب کیلوگرم است که طبق اصلاحیه سوال ۵۰ در نظر گرفته شد.

خطوط ۶۲ تا ۶۳: مقادیر خطا با استفاده از فانکشن قدر مطلق متلب حساب شده و درون دو ماتریس خطوط ۶۴ تا ۶۵: با استفاده از فانکشن find مطلب ، اندیس هایی از دو ماتریس ارور بخش قبل که دارای مقادیر کمتر از حداقل مقدار مجاز خطا هستند یافت می شوند و این اندیس های برای خط در ماتریس اندیس ۱ و برای سهمی در ماتریس اندیس ۲ ذخیره می شوند.

خطوط ۶۶ تا ۶۷: با استفاده از فانکشن find مطلب ، اندیس هایی از دو ماتریس ارور بخش قبل که دارای مقادیر کمتر از حداقل مقدار مجاز خطا هستند یافت می شوند و این اندیس های برای خط در ماتریس اندیس ۱ و برای سهمی در ماتریس اندیس ۲ ذخیره می شوند.

خطوط ۶۸ تا ۶۹: در نهایت از ماتریس اصلی سوال که همان جدول صورت سوال است مقادیر قیمتی که منجر به خطای کمتر از مقدار ۰.۵ شده اند، که در بخش قبل پیدا شدند برای خط و سهمی به صورت جداگانه در ماتریس های price1 برای خط و price2 برای سهمی نمایش داده می شوند.

```

price1 = 310
price2 =

```

1×0 empty **double** row vector

مقادیر قیمت (برای خط یک قیمت ۳۱۰ و برای سهمی قیمتی که خطای کمتر از مقدار مورد نظر ما تولید کند وجود ندارد)



```

58 %--- Part C - Question ---
59
60 maxError=0.5;
61
62 totalError1= abs(priceDemand(2,1:tableSize)-q1(priceDemand(1,1:tableSize)));
63 totalError2= abs(priceDemand(2,1:tableSize)-q2(priceDemand(1,1:tableSize)));
64
65 indeces1= find(maxError > totalError1);
66 indeces2= find(maxError > totalError2);
67
68 price1= priceDemand(1,indeces1)
69 price2= priceDemand(1,indeces2)

```

سوال دو

تئوری:

توضیح کد :

خط ۱ : تمامی پنجره های قبلی بسته شوند تمامی داده های قبلی پاک شوند پنجره ی دستورات نیز پاک شود.

خطوط ۳ تا ۵ : هر ستون از جدول صورت سوال در یک ماتریس یک ردیفه تعریف شده است ماتریس A داده های زمان هستند که از صفر تا یک با طول ۵۰.۱ هستند ماتریس X ماتریس موقعیت افقی و ماتریس Y موقعیت عمودی را نشان می دهند.

خط ۶: تعداد نقاط جدولی با استفاده از فانکشن متلب برای محاسبه های بعدی محاسبه می شوند.

خطوط ۸ تا ۱۱: ماتریس جواب نهایی را ماتریس $speed$ می نامیم این ماتریس از چهار ردیف تشکیل شده که به ترتیب ردیف (زمان ها) ردیف (سرعت در مولفه ای افقی) ردیف (سرعت در مولفه عمودی) و ردیف (مقدار سرعت) هستند و به تعداد نقاط جدولی ستون دارد در ابتدا زمان ها را به ترتیب در ردیف اول این ماتریس می قرار می دهیم سپس با استفاده از تابع $numDiff$ که تابع مشتق گیری عددی ای است که خودمان تعریف کرده ایم مشتقات مولفه ای افقی بر حسب زمان را می یابیم و در ردیف دوم قرار می دهیم در انتهای این بخش تابع مشتق گیری عددی به طور کامل توضیح داده شده سپس همین عمل را برای مولفه ای عمودی و ردیف سوم تکرار می کنیم در نهایت مقدار سرعت را با توجه به دو ردیف قبلی حساب کرده و در ردیف چهارم قرار می دهیم

خط ۱۳ : ماتریس جواب را نمایش می دهیم.

	1	2	3	4	5	6	7	8	9	10	11
1	0	0.1000	0.2000	0.3000	0.4000	0.5000	0.6000	0.7000	0.8000	0.9000	1.0000
2	0.3900	0.8900	1.2500	1.3250	1.1200	0.6300	-0.1450	-1.1950	-2.5300	-4.1500	-5.9100
3	0.3500	0.3700	0.3100	0.0850	-0.3000	-0.8500	-1.5650	-2.4350	-3.4700	-4.6700	-5.9500
4	0.5240	0.9638	1.2879	1.3277	1.1595	1.0580	1.5717	2.7124	4.2944	6.2475	8.3863

0.2000	زمان
1.2500	سرعت در راستای مولفه‌ی افقی
0.3100	سرعت در راستای مولفه‌ی عمودی
1.2879	اندازه‌ی سرعت

```

● ● ●
1 clc, clear , close all
2
3 T= 0:0.1:1
4 X=[ 0.5, 0.564, 0.678, 0.814, 0.943, 1.038, 1.069, 1.009, 0.830, 0.503, 0]
5 Y=[ 1.5, 1.536, 1.574, 1.598, 1.591, 1.538, 1.421, 1.225, 0.934, 0.531, 0]
6 pointSize= size(T,2);
7
8 speed(1,1:pointSize)= T;
9 speed(2,1:pointSize)= numDiff(X,T);
10 speed(3,1:pointSize)= numDiff(Y,T);
11 speed(4,1:pointSize)= ((speed(2,1:pointSize).^2)+(speed(3,1:pointSize).^2)).^(1/2);
12
13 speed

```

تابع مشتق گیر عددی

خط ۱۴: تابع تعریف می شود ورودی آن دو ماتریس نقاط جدولی یک تابع و مقادیر آن ها است و خروجی این تابع یک ماتریس شامل مشتق در همان نقاط جدولی است.

خط ۱۵: برای محاسبات بعدی تعداد نقاط جدولی محاسبه می شوند.

خطوط ۱۸ تا ۲۳: از آنجایی که برای مشتق گیری عددی حداقل نیازمند ۲ نقطه هستیم و برای بیش از ۲ نقطه فاصله‌ی نقاط باید ثابت باشد این مواد را با استفاده از دو شرط چک می کنیم.

خط ۲۵: مقدار h که همان فاصله‌ی بین دو نقطه‌ی متوالی است را می یابیم.

خطوط ۲۷ تا ۳۳: بر اساس فرمول‌های مشتق گیری عددی که در بخش تئوری آمده برای نقطه اول و نقطه‌های وسط و نقطه‌ی انتهای از فرمول‌های جداگانه‌ای استفاده کرد. پس این فرمول‌ها به صورت جداگانه نوشته شده اند همچنین برای محاسبه مشتق تمام خانه‌های وسط از یک حلقه استفاده شده که به ترتیب نقاط وسطی را پیمایش می کند و در فرمول مناسب قرار می دهد و همگی مقادیر مشتق تولید شده به ترتیب در ماتریس نتیجه قرار می گیرند.

```
14 function result = numDiff(values,points)
15
16     pointSize= size(points,2);
17
18     if pointSize < 2
19         disp('we must have at least 2 point!')
20     elseif pointSize == 2 && points(2)-points(1) ~= points(3)-points(2)
21         disp('distances between the points must be equal!')
22         return
23     end
24
25     h=points(2)-points(1);
26
27     result(1,1)= (-values(1,3)+4*values(1,2)-3*values(1,1))/(2*h);
28
29     for i=2:pointSize-1
30         result(1,i)= (values(1,i+1)-values(1,i-1))/(2*h) ;
31     end
32
33     result(1,pointSize)= (3*values(1,pointSize)-4*values(1,pointSize-1)+values(1,pointSize-2))/(2*h) ;
34
35 end
```