Problem 19.11

The following data was collected for the distance travelled versus time for a rocket:

<table>
<thead>
<tr>
<th>t, sec.</th>
<th>0</th>
<th>25</th>
<th>50</th>
<th>75</th>
<th>100</th>
<th>125</th>
</tr>
</thead>
<tbody>
<tr>
<td>y, km</td>
<td>0</td>
<td>32</td>
<td>58</td>
<td>78</td>
<td>92</td>
<td>100</td>
</tr>
</tbody>
</table>

Use numerical differentiation to calculate velocity and acceleration at each time.

At time \( t=0 \), we can calculate the velocity by forward differences \( v=(32-0)/(25-0)=1.28 \text{ km/s} \). We can also use a three point forward difference equation

\[
f'(x) = \frac{-f(x+2h) + 4f(x+h) - 3f(x)}{2h} = \frac{-58 + 4 \times 32 - 3 \times 0}{50} = 1.4
\]

We can also fit a quintic polynomial to the data and take its derivative at zero

\[
\text{>> } t=\text{[0:25:125]}
\]
\[
t =
\begin{align*}
0 & \quad 25 & \quad 50 & \quad 75 & \quad 100 & \quad 125 \\
\end{align*}
\]

\[
\text{>> } y=\text{[0 32 58 78 92 100]}
\]
\[
y =
\begin{align*}
0 & \quad 32 & \quad 58 & \quad 78 & \quad 92 & \quad 100 \\
\end{align*}
\]

\[
\text{>> } p=\text{polyfit}(t,y,5)
\]

Warning: Polynomial is badly conditioned. Add points with distinct \( X \) values, reduce the degree of the polynomial, or try centering and scaling as described in HELP POLYFIT.

\[
p =
\begin{align*}
-0.0000 & \quad 0.0000 & \quad -0.0000 & \quad -0.0048 & \quad 1.4000 & \quad 0.0000 \\
\end{align*}
\]

\[
\text{>> format short e}
\]
\[
\text{>> } p
\]
\[
p =
\begin{align*}
-3.0478e-022 & \quad 1.0186e-019 & \quad -1.2065e-017 & \quad -4.8000e-003 & \quad 1.4000e+000 & \quad 7.2060e-015 \\
\end{align*}
\]
Which means that the data fits very well a quadratic polynomial

\[ v_0 = p(5) \]

\[ v_0 = 1.4000e+000 \]

We verify that a quadratic polynomial will do

\[ p = \text{polyfit}(t, y, 2) \]

\[ pq = -4.8000e-003 \quad 1.4000e+000 \quad 9.6598e-015 \]

For the velocity at \( t=25 \), we can use two-point forward differences \((58-32)/25=1.04\), two-point backward differences \((32-0)/25=1.28\)

two-point centered differences \((58-0)/50=1.16\) (which is the average of the backward and forward differences), or we can use the three-point forward differences

\[
f'(x) = \frac{-f(x+2h) + 4f(x+h) - 3f(x)}{2h} = \frac{-78 + 4\times58 - 3\times32}{50} = 1.16
\]

The fact that the three-point forward difference gives the same result as the centered differences confirms the fact that the data is derived from a quadratic polynomial. If we use the Matlab function \text{gradient} we can get the velocities everywhere

\[ \text{vel} = \text{gradient}(y, 25) \]

\[ \text{vel} = 1.2800e+000 \quad 1.1600e+000 \quad 9.2000e-001 \quad 6.8000e-001 \quad 4.4000e-001 \quad 3.2000e-001 \]

It uses the centered differences at all points except the ends where it uses two-point forward differences (first point) and
two-point backward differences (last point). We can apply once again for the acceleration

\[ \text{acc} = -4.8000e-003 \quad -7.2000e-003 \quad -9.6000e-003 \quad -9.6000e-003 \quad -7.2000e-003 \quad -4.8000e-003 \]

We see that the gradient function is inaccurate for the two extreme points for the velocity, but for the acceleration, two points on each ends are less accurate.
To see the effect of noise we contaminate the distance measurements with normally distributed errors with standard deviation of 1 km.

```matlab
>> yerr=randn(1,6)
yerr =
   -4.3359e-001   3.4262e-001   3.5784e+000   2.7694e+000  -1.3499e+000   3.0349e+000
>> ytot=y+yerr
ytot =
   -4.3359e-001   3.2343e+001   6.1578e+001   8.0769e+001   9.0650e+001   1.0303e+002
```

We calculate the velocity again from using the gradient function

```matlab
>> vtot=gradient(ytot,25)
vtot =
   1.3110e+000   1.2402e+000   9.6854e-001   5.8143e-001   4.4531e-001   4.9539e-001
```

Compare to the original calculation

```matlab
vel =
   1.2800e+000   1.1600e+000   9.2000e-001   6.8000e-001   4.4000e-001   3.2000e-001
```

With the maximum difference of 0.18 km/sec.

We next calculate acceleration with the gradient again

```matlab
>> acctot=gradient(veltot,25)
acctot =
   -2.8324e-003  -6.8502e-003  -1.3176e-002  -1.0465e-002  -1.7208e-003  2.0033e-003
```

Compared to the original acceleration

```matlab
acc =
```

Now the differences are large. Recall that the true acceleration without noise or differentiation errors is -0.96e-003.

We can filter out noise and reduce the finite-difference approximations by fitting the data with a quadratic polynomial
>> ptot=polyfit(t,ytot,2)

ptot =
-5.1251e-003  1.4537e+000 -1.7187e-001

Compare with the noise-free fit

pq =
-4.8000e-003  1.4000e+000  9.6598e-015

The error in the coefficients are much smaller than the errors in the data.

Ptot(3)=-0.17187 is the position at t=0. With the noise we measured -0.43359, but the fit reduced the error by more than a factor of two.

Ptot(2)=1.4537 is the speed at t=0. The true value is pq(2)=1.4, finite difference from the noise-free data gave us 1.28, and with the noise we got 1.311. So again the fitted result is more accurate.

2*ptot(3)=-1.025e-002 is the acceleration throughout the trajectory. The true value is -9.6e-003, so that the error is about 7%. However, the gradient value range from 2e-003 (wrong sign) to -1.3176e-002, which 35% too high.