Homework 1

[redacted]

1.1 ① $\hat{L}(w) = \frac{1}{3} \sum_{i=1}^{3} \frac{1}{2} (w x_i - y_i)^2$

$\hat{L}'(w) = \frac{1}{3} \sum_{i=1}^{3} x_i (w x_i - y_i)$

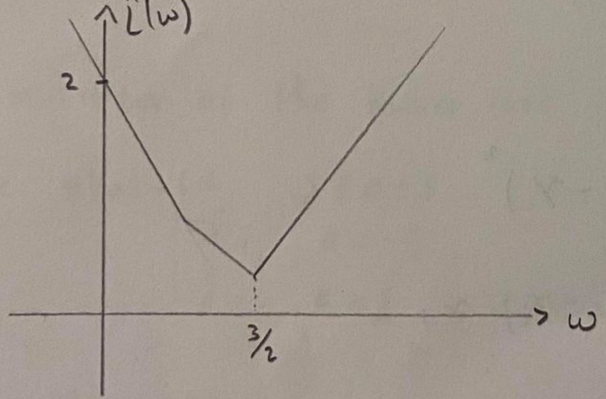
$\hat{L}'(w) = 0 \implies 0 = \sum_{i=1}^{3} w x_i^2 - x_i y_i$

$$w = \frac{\sum x_i y_i}{\sum x_i^2}$$

For $X = (1,2,3)^T$, $Y = (2,4,5)^T$, we get $w = \frac{25}{14}$

② $\hat{L}(w) = \frac{1}{2} \sum_{i=1}^{2} |w x_i - y_i|$

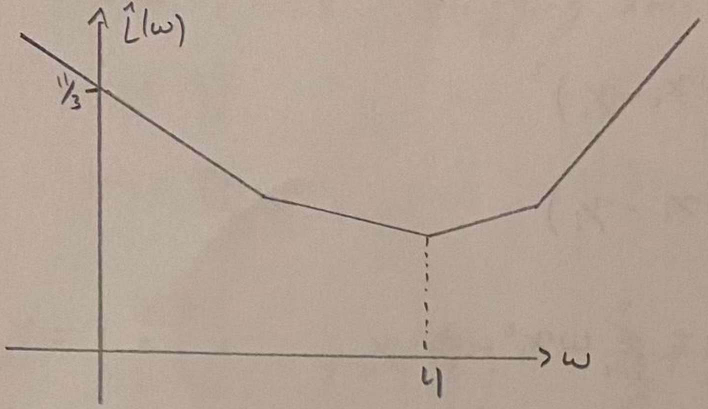
$= \frac{1}{2} |w - 1| + \frac{1}{2} |2w - 3|$



$w = \frac{3}{2}$

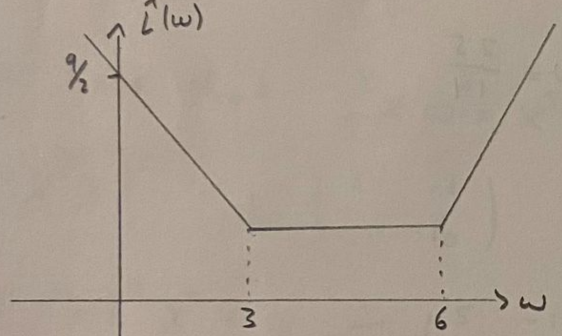
③ $\hat{L}(w) = \frac{1}{m} \sum_{i=1}^{m} |w - y_i|$

- For $y = (2,4,5)^T$: $\hat{L}(w) = \frac{1}{3}|w-2| + \frac{1}{3}|w-4| + \frac{1}{3}|w-5|$



$w = 4$

- For $y = (3,6)$: $\hat{L}(w) = \frac{1}{2}|w-3| + \frac{1}{2}|w-6|$



$w^* \in [3,6]$

(any $w \in [3,6]$ is a minimizer)

(4) See code at the end

(5) i) $\hat{L}(w) = \frac{1}{3} \sum_{i=1}^{3} \frac{1}{2}(w_1 x_{i1} + w_2 x_{i2} - y_i)^2$

$$\begin{cases} 0 = \dfrac{\partial \hat{L}(w)}{\partial w_1} = \frac{1}{3} \sum_{i=1}^{3} (w_1 x_{i1} + w_2 x_{i2} - y_i) \cdot x_{i1} \\ 0 = \dfrac{\partial \hat{L}(w)}{\partial w_2} = \frac{1}{3} \sum_{i=1}^{3} (w_1 x_{i1} + w_2 x_{i2} - y_i) \cdot x_{i2} \end{cases}$$

$$\Rightarrow \begin{cases} 0 = 3 \cdot (3w_1 - 6) + \cancel{0 \cdot (2w_2 - 2)} + 1 \cdot (w_1 + w_2 - 5) = 10w_1 + w_2 - 23 & (1) \\ 0 = \cancel{0 \cdot (3w_1 - 6)} + 2 \cdot (2w_2 - 2) + 1 \cdot (w_1 + w_2 - 5) = w_1 + 5w_2 - 9 & (2) \end{cases}$$

$\Rightarrow \quad w_1 = 9 - 5w_2 \quad$ from (2)

$90 - 49w_2 = 23 \quad$ plugging into (1)

$$\begin{cases} w_2 = \frac{67}{49} \\ w_1 = \frac{106}{49} \end{cases}$$

ii) $X = \begin{pmatrix} 3 & 0 \\ 0 & 2 \\ 1 & 1 \end{pmatrix}, \quad Y = \begin{pmatrix} 6 \\ 2 \\ 5 \end{pmatrix}, \quad X^T = \begin{pmatrix} 3 & 0 & 1 \\ 0 & 2 & 1 \end{pmatrix}$

$$X^T X = \begin{pmatrix} 10 & 1 \\ 1 & 5 \end{pmatrix}, \quad X^T Y = \begin{pmatrix} 23 \\ 9 \end{pmatrix}$$

we have that $X^T X w = X^T Y \Rightarrow w = (X^T X)^{-1} X^T Y$

$$(X^T X)^{-1} = \frac{1}{49} \begin{pmatrix} 5 & -1 \\ -1 & 10 \end{pmatrix}$$

$$w = (X^T X)^{-1} X^T Y = \frac{1}{49} \begin{pmatrix} 5 & -1 \\ -1 & 10 \end{pmatrix} \begin{pmatrix} 23 \\ 9 \end{pmatrix} = \begin{pmatrix} \frac{106}{49} \\ \frac{67}{49} \end{pmatrix} \quad \square$$

6) The minimizer of the huber loss is: $\frac{1}{10} \sum_{i=1}^{10} q'(e)$

where $q'(e) = \begin{cases} e, & -\delta \leq e \leq \delta \\ -\delta, & e < -\delta \\ \delta, & e > \delta \end{cases}$

so for the numbers $e \in \{-2, -1, 0, 0, 0, 0, 0, 0, 1, 10\}$,

$$\frac{1}{10} \sum_{i=1}^{10} q'(e) = (-1 - 1 + 0 + 0 + 0 + 0 + 0 + 0 + 1 + 1) \cdot \frac{1}{10} = 0 \quad (\delta = 1)$$

And the 10% windzorized mean is exactly the same (the lowest and largest values become the second lowest and second largest respectively, giving the same sum).

The main difference between the two is that is that the minimizer of the huber loss says values outside $[-\delta, \delta]$ are outliers, while the windsorized mean says tha $x\%$ of the values are outliers for some x.

So for the huber loss the intlier region is <u>always</u> $[-\delta, \delta]$ while for the windsorized mean the inlier region is determined by the data itself.

7) $\hat{L}(w) = \frac{1}{10} \sum_{i=1}^{10} l(w, y_i)$

- For l_1: $\hat{L}(w) = \frac{1}{10}(9|w| + |w-y|)$

For $y \geq 0$, $\hat{L}'(w) = \begin{cases} 10, & w > y \geq 0 \\ 8, & 0 < w < y \\ -10, & w < 0 \end{cases}$ For $y < 0$, $\hat{L}'(w) = \begin{cases} -10, & w < y < 0 \\ -8, & y < w < 0 \\ 10, & w > 0 \end{cases}$

So in both cases the derivative changes direction at $w = 0$, making $\boxed{w=0}$ our minimizer.

- For l_2: $\hat{L}(w) = \frac{1}{10}\left(\frac{9}{2}w^2 + \frac{1}{2}(w-y)^2\right)$

$$\hat{L}'(w) = \frac{9}{10}w + \frac{1}{10}(w-y) = w - \frac{y}{10}$$

$$\hat{L}'(w) = 0 \Rightarrow \boxed{w = y/10} \text{ is our minimizer}$$

- For L_{huber}: $\hat{L}(w) = \frac{1}{10}(9q(w) + q(w-y))$ where $q(e) = \begin{cases} \frac{1}{2}e^2, & |e| \leq \delta \\ -e\delta - \frac{1}{2}\delta^2, & e < -\delta \\ e\delta - \frac{1}{2}\delta^2, & e > \delta \end{cases}$

we have four cases:

1) Both 0 and y are outliers. But this makes it just the L_1 error, which we know minimizes at 0, and this contradicts 0 being an outlier.

2) Both 0 and y are inliers. This is just the l_2 error and minimizes at $w^* = \frac{y}{10}$. But we need y to be an inlier so we must have

$|\frac{y}{10} - y| \leq 1 \Rightarrow |y| < \frac{10}{9}$

3) 0 is an outlier, y is an inlier. In this case $\hat{L}(w) = \frac{9}{10}|w| + \frac{1}{20}(w-y)^2$

$\Rightarrow \hat{L}'(w) = \frac{9}{10} + \frac{1}{10}(w-y)$ (for $y > w \geq 0$, the reverse case is symmetric)

$\hat{L}'(w) = 0 \Rightarrow w = y - 9$ but this contradicts y being an inlier.

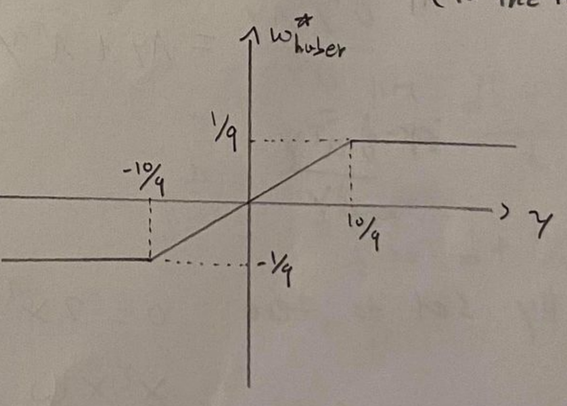
4) y is an outlier, 0 is an inlier. In this case $\hat{L}(w) = \frac{9}{20}w^2 + \frac{1}{10}|w-y|$

$\Rightarrow \hat{L}'(w) = \frac{9}{10}w \pm \frac{1}{10}$ (for $0 \leq w \leq y$, the reverse case is symmetric).

$\hat{L}'(w) = 0 \Rightarrow w = \frac{1}{9}$

Since y is an outlier we must have $y > \frac{1}{9} + 1 = \frac{10}{9}$ (and $y < \frac{-10}{9}$ in the reverse case)

so $w^*_{huber} = \begin{cases} -\frac{1}{9}, & y < \frac{-10}{9} \\ \frac{y}{10}, & |y| \leq \frac{10}{9} \\ \frac{1}{9}, & y > \frac{10}{9} \end{cases}$



$w^*_{huber}(y) = \text{Median}\left(-\frac{1}{9}, \frac{y}{10}, \frac{1}{9}\right)$

1.2) ① $X^T X w = X^T Y$ (★)

If $d=1$, the X is $m \times 1$, so $X^T X = \|X\|^2$

and $X^T Y = X \cdot Y$

Then ★ becomes $w\|X\|^2 = X \cdot Y \Rightarrow w = \frac{X \cdot Y}{\|X\|^2}$

② Linear model: $\hat{y} = Xw$, so $e = Xw - y$ and the L_2 loss is $\frac{1}{M}\|Xw - y\|^2$

we now minimize $\cancel{\frac{1}{M}}\|Xw - y\|^2$ (removing the $\frac{1}{M}$ for simplicity).

$$\|Xw-y\|^2 = (Xw-y)^T(Xw-y)$$
$$= (w^TX^T - y^T)(Xw-y)$$
$$= w^TX^TXw - w^TX^Ty - y^TXw - y^Ty$$
$$= w^TX^TXw - 2y^TXw - y^Ty \quad \left(\begin{array}{l}\text{Since } w^TX^Ty \text{ is a scalar} \\ \text{then } w^TX^Ty = (w^TX^Ty)^T = y^TXw\end{array}\right)$$

Then
$$\frac{\partial \|Xw-y\|^2}{\partial w} = X^TXw + (X^TX)^Tw - 2X^Ty \quad (*)$$
$$= 2X^TXw - 2X^Ty$$

wher here (in $*$) we cite the matrix theory facts that:

1) $\dfrac{\partial y^TAy}{\partial y} = Ay + A^Ty$

2) $\dfrac{\partial a^Ty}{\partial y} = a$

we finally set to zero: $0 = 2X^TXw - 2X^Ty$

$$X^TXw = X^Ty$$
$$w = (X^TX)^{-1}Xy \quad \text{as expected.}$$

③ $L_{huber}(e) = \begin{cases} \frac{1}{2}e^2, & |e| \leq \delta \\ \delta(e - \frac{1}{2}\delta), & e > \delta \\ \delta(-e - \frac{1}{2}\delta), & e < -\delta \end{cases}$

Clearly this is continuous and differentiable everywhere except possibly at $\pm\delta$, which we manually check:

$$\lim_{e\to\delta^+} L_\delta(e) = \lim_{e\to\delta^+} \delta(e - \tfrac{1}{2}\delta) = \tfrac{1}{2}\delta^2$$

$$\lim_{e\to\delta^-} L_\delta(e) = \lim_{e\to\delta^-} \tfrac{1}{2}e^2 = \tfrac{1}{2}\delta^2$$

So $\lim_{e\to\delta^+} L_\delta(e) = \lim_{e\to\delta^-} L_\delta(e) = L_\delta(\delta) = \tfrac{1}{2}\delta^2$ so $L_\delta(e)$ is continuous at δ.

$$\lim_{e\to-\delta^+} L_\delta(e) = \lim_{e\to-\delta^+} \tfrac{1}{2}e^2 = \tfrac{1}{2}\delta^2$$

$$\lim_{e\to-\delta^-} L_\delta(e) = \lim_{e\to-\delta^-} \delta(-e - \tfrac{1}{2}\delta) = \tfrac{1}{2}\delta^2$$

So $\lim_{e\to-\delta^-} L_\delta(e) = \lim_{e\to-\delta^+} L_\delta(e) = L_\delta(-\delta) = \tfrac{1}{2}\delta^2$ so $L_\delta(e)$ is continuous at $-\delta$.

Now $$\lim_{h\to0^+} \frac{L_\delta(\delta+h) - L_\delta(\delta)}{h} = \lim_{h\to0^+} \frac{\delta(\delta+h-\tfrac{1}{2}\delta) - \tfrac{1}{2}\delta^2}{h} = \lim_{h\to0^+} \frac{\delta h}{h} = \delta$$

$$\lim_{h\to0^-} \frac{L_\delta(\delta+h) - L_\delta(\delta)}{h} = \lim_{h\to0^-} \frac{\tfrac{1}{2}(\delta+h)^2 - \tfrac{1}{2}\delta^2}{h} = \lim_{h\to0^-} \frac{\delta h + \frac{h^2}{2}}{h}$$

$$= \lim_{h\to0^-} \delta + h/2$$

so $L_\delta(e)$ is differentiable at δ.

$$= \delta$$

$$\lim_{h\to0^+} \frac{L_\delta(-\delta+h) - L_\delta(-\delta)}{h} = \lim_{h\to0^+} \frac{\tfrac{1}{2}(-\delta+h)^2 - \tfrac{1}{2}\delta^2}{h} = \lim_{h\to0^+} \frac{-\delta h + h^2/2}{h}$$

$$= \lim_{h\to0^+} -\delta + h/2$$

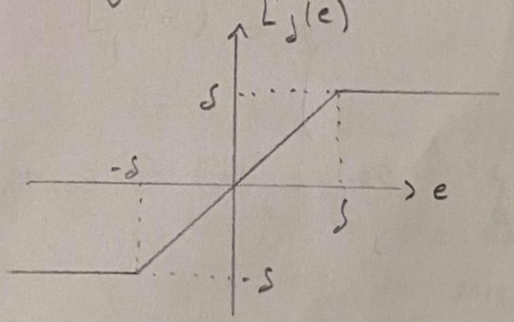
$$= -\delta$$

$$\lim_{h\to0^-} \frac{L_\delta(-\delta+h) - L_\delta(-\delta)}{h} = \lim_{h\to0^-} \frac{\delta((\delta-h)-\tfrac{1}{2}\delta) - \tfrac{1}{2}\delta^2}{h} = \lim_{h\to0^-} \frac{-\delta h}{h} = -\delta$$

so $L_\delta(e)$ is differentiable at $-\delta$.

$$L'_\delta(e) = \begin{cases} e, & |e| \le \delta \\ \delta, & e > \delta \\ -\delta, & e < -\delta \end{cases} \quad \text{so} \quad L''_\delta(e) = \begin{cases} 1, & |e| < \delta \\ 0, & |e| > \delta \\ \text{undefined}, & |e| = \delta \end{cases}$$

$\Downarrow$



(☆)

⑤ For $L_2(e) = \frac{1}{2}e^2$: $L_2''(e) = 1 \ \forall e$ so L_2 is convex.

For $L_1(e) = |e|$:

$$\begin{aligned} L_1(te_1 + (1-t)e_2) &= |te_1 + (1-t)e_2| \\ &\le |te_1| + |(1-t)e_2| \quad \text{by the triangle inequality} \\ &= t|e_1| + (1-t)|e_2| \quad \text{for any } t \in [0,1] \\ &= tL_1(e_1) + (1-t)L_1(e_2) \end{aligned}$$

so by definition L_1 is convex $\square$

For $L_{huber}(e) = \begin{cases} \frac{1}{2}e^2, & |e| \le \delta \\ \delta(e - \frac{1}{2}\delta), & e > \delta \\ \delta(-e - \frac{1}{2}\delta), & e < -\delta \end{cases}$: $L'_{huber}(e)$ is monotonically non-decreasing (see ☆ above) so by definition $L_{huber}(e)$ is also convex.

1.3

① i) Let the grades be $g_1 \le g_2 \le g_3 \le g_4 \le g_5$, then we could make the final grade fg:

$$fg = \max\left(\frac{g_1+g_2+g_3+g_4+g_5}{5}, \frac{g_2+g_3+g_4+g_5-\delta}{4}\right)$$

so in other words, the lowest grade will have an effect of at most δ on the average (where we can chose δ as desired)

ii) Solving CVIL (in the example above) with $0.9, 0.9, 0.9, 0.9, \gamma$ as our scores will give:

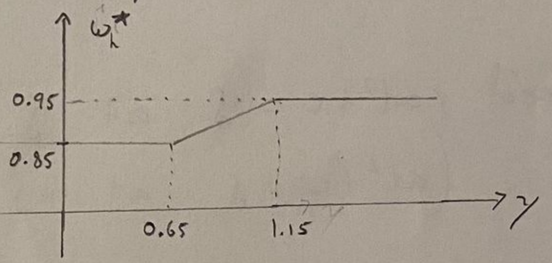
$\rightarrow \frac{3.6+\gamma}{5}$, when $\left|\frac{3.6+\gamma}{5} - \gamma\right| < 0.2$

$0.72 + \gamma/5$, when $\gamma > 0.65$ or $\gamma < 1.15$

$\rightarrow 0.85$, when $\gamma \leq 0.65$ and 0.95, when $\gamma \geq 1.15$

Note that solving is the same as in problem 1.1-7 where we skipped calculations for simplicity, but we will get the same result: the average up until γ becomes an outlier, and then a flat value:

$$w_h^* = \text{median}\left(0.85, \frac{3.6+\gamma}{5}, 0.95\right)$$



So in our example if $\gamma = 0$ (missed assignment) then $w^* = 0.85$ which is the same as $\frac{0.9+0.9+0.9+0.9-0.2}{4} = 0.85$ so an effect of only $\delta = 0.2$ in the average, as required.

Note that the solution $w^*(\gamma) = \text{median}\left(0.85, \frac{3.6+\gamma}{5}, 0.95\right)$ is almost the same as our max function in part (i), just that it is capped above aswell (in practice we wouldn't want this for grading!)

(2) $L_{flipped}(e) = \begin{cases} |t|, & |t| < \delta \\ at^2 + bt + c, & |t| \geq \delta \end{cases}$

since $|t|$ is symmetric around zero, we want at^2+bt+c to also be symmetric around zero, so let $b = 0$.

Then we get $L_{flipped}(e) = \begin{cases} |t|, & |t| < \delta \\ at^2 + c, & |t| \geq \delta \end{cases}$

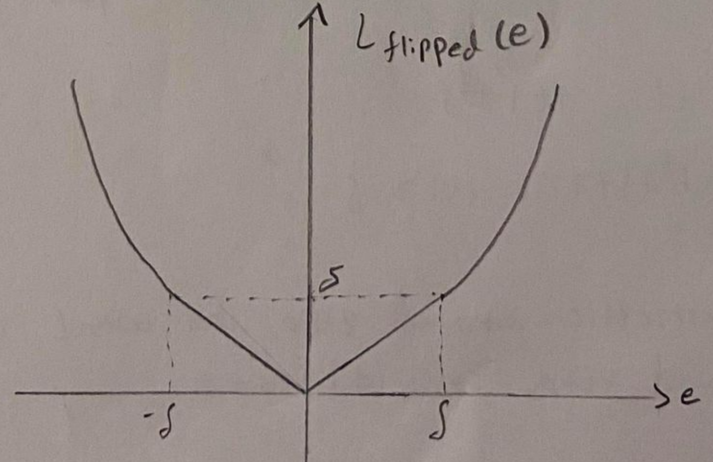
$$L'_{flipped}(e) = \begin{cases} 2at, & t \leq -\delta \\ -1, & -\delta < t < 0 \\ 1, & 0 < t < \delta \\ 2at, & t \geq \delta \end{cases}$$

For differentiability we need $\begin{cases} 2at = -1 & \text{at } -\delta \\ 2at = 1 & \text{at } \delta \end{cases}$ so $a = \frac{1}{2\delta}$

For continuity we need $\begin{cases} at^2 + c = \delta & \text{at } \delta \\ at^2 + c = \delta & \text{at } -\delta \end{cases}$

$$\Rightarrow \begin{cases} \frac{1}{2\delta}(\delta^2) + c = \delta \\ \frac{1}{2\delta}(-\delta)^2 + c = \delta \end{cases} \Rightarrow c = \delta/2$$

So $L_{flipped}(e) = \begin{cases} |t|, & |t| < \delta \\ t^2/2\delta + \delta/2, & |t| \geq \delta \end{cases}$



1.1 Q4)

Import libraries as needed

In [1]:
```
import numpy as np
import math
import matplotlib.pyplot as plt
```

Create our X variable and our random noise

In [2]:
```
X=np.linspace(0,1,11)
e=np.random.uniform(0,1,11)
```

Define our Y variable

In [3]:
```
def g(m):
    return np.sin(2*np.pi*m)

Y=g(X)+0.1*e
```

Now we will use numpy polyfit method to fit to the polynomial models given. Since this implicitly defines the matrices, I will answer the dimension questions first below:

For the first case f(X)=(1,X): we have that F will be 11x2 , F(t)F will be 2x2 , and w will be 2x1

For the second case: f(X)=(1,X,X^2,X^3) we have that F will be 11x4 , F(t)F will be 4x4 , and w will be 4x1

The matrices are shown below for illustration, but as mentioned np.polyfit will be used for the regression

In [4]:
```
one=np.ones(11)
FT1=np.matrix([X,one])
FT3=np.matrix([X**3,X**2,X,one])
```

In [5]:
```
F1=np.transpose(FT1)
F3=np.transpose(FT3)
```

In [6]:
```
print(F1)
print(F3)
```

```
[[0.  1. ]
 [0.1 1. ]
 [0.2 1. ]
 [0.3 1. ]
 [0.4 1. ]
 [0.5 1. ]
 [0.6 1. ]
 [0.7 1. ]
 [0.8 1. ]
 [0.9 1. ]
 [1.  1. ]]
[[0.    0.    0.    1.   ]
 [0.001 0.01  0.1   1.   ]
 [0.008 0.04  0.2   1.   ]
 [0.027 0.09  0.3   1.   ]
 [0.064 0.16  0.4   1.   ]
```

```
 [0.125 0.25  0.5   1.    ]
 [0.216 0.36  0.6   1.    ]
 [0.343 0.49  0.7   1.    ]
 [0.512 0.64  0.8   1.    ]
 [0.729 0.81  0.9   1.    ]
 [1.    1.    1.    1.    ]]
```

We now fit our data F, to our variable Y via least squares

In [7]:
```
model1=np.polyfit(X,Y,1)
model1
```

Out[7]:
```
array([-1.420743  ,  0.76067946])
```

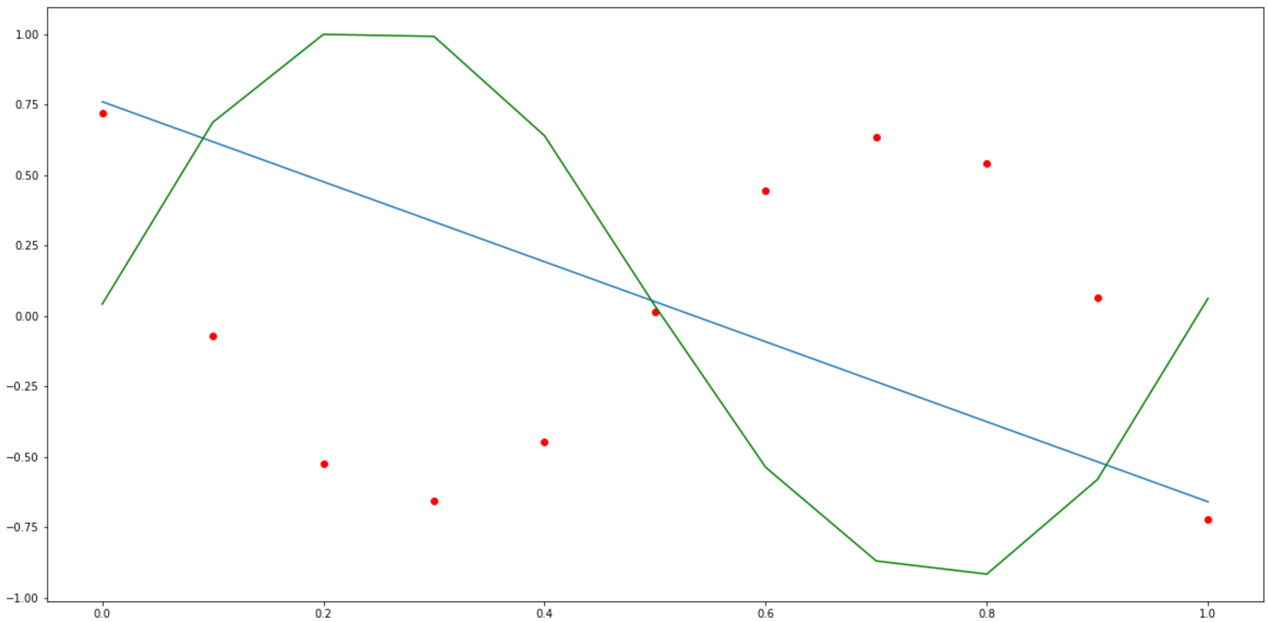
Note that the output above is our solution, w (the transpose of it).

Now we plot the solution (blue), with the actual data (green), and the errors (red)

In [8]:
```
e1=np.polyval(model1,X)-Y
```

In [9]:
```
plt.figure(figsize=(20,10))
plt.plot(X, np.polyval(model1,X))
plt.plot(X, Y,color='green')
plt.scatter(X, e1,color='red')
```

Out[9]:
```
<matplotlib.collections.PathCollection at 0x7f9c7fcb7400>
```



The fit is not great and we can also see that the residuals (our L_2 loss) is pretty high (second output).

In [10]:
```
np.polyfit(X,Y,1,full=True)
```

Out[10]:
```
(array([-1.420743  ,  0.76067946]),
 array([2.85190917]),
 2,
 array([1.35836455, 0.39350444]),
 2.4424906541753444e-15)
```

We now repeat with f(X)=(1,X,X^2,X^3)

In [11]:
```
model3=np.polyfit(X,Y,3)
model3
```

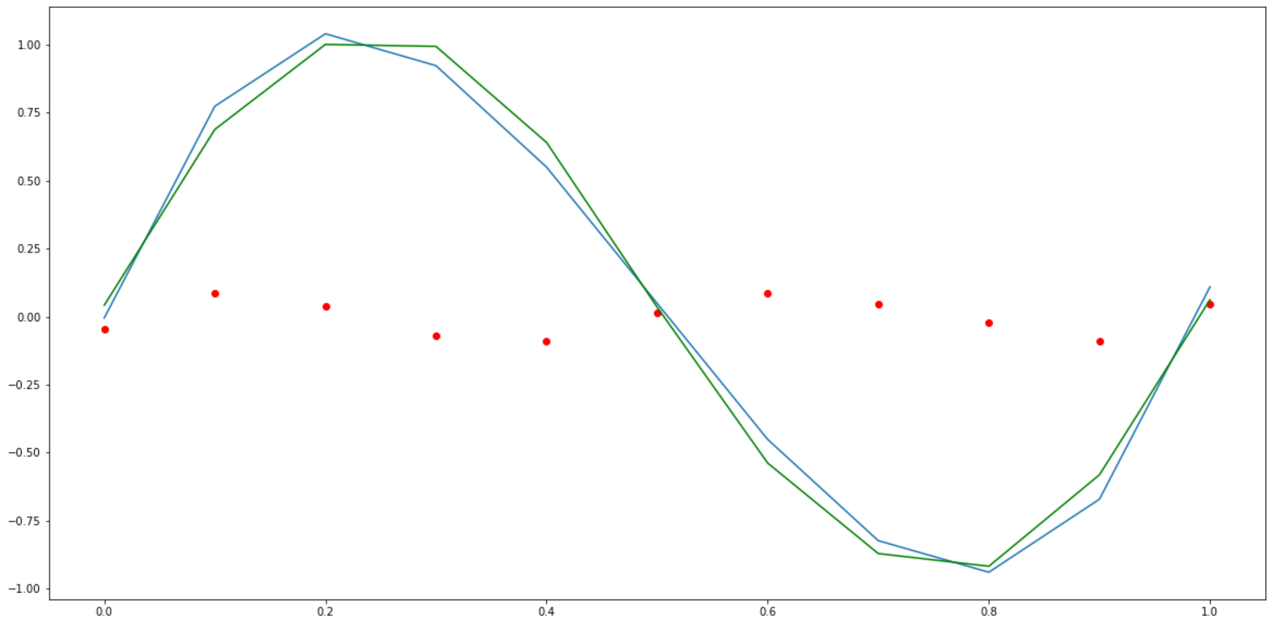
Out[11]:
```
array([ 2.13163290e+01, -3.19603750e+01,  1.07580786e+01, -4.59060260e-03])
```

Again, the transpose of the above is our solution, w.

In [12]:
```
e3=np.polyval(model3,X)-Y
```

In [13]:
```
plt.figure(figsize=(20,10))
plt.plot(X, np.polyval(model3,X))
plt.plot(X, Y,color='green')
plt.scatter(X, e3,color='red')
```

Out[13]:
```
<matplotlib.collections.PathCollection at 0x7f9c806c2fd0>
```



We can see that the fit is much much better and our residuals (L_2 loss) is also a lot lower:

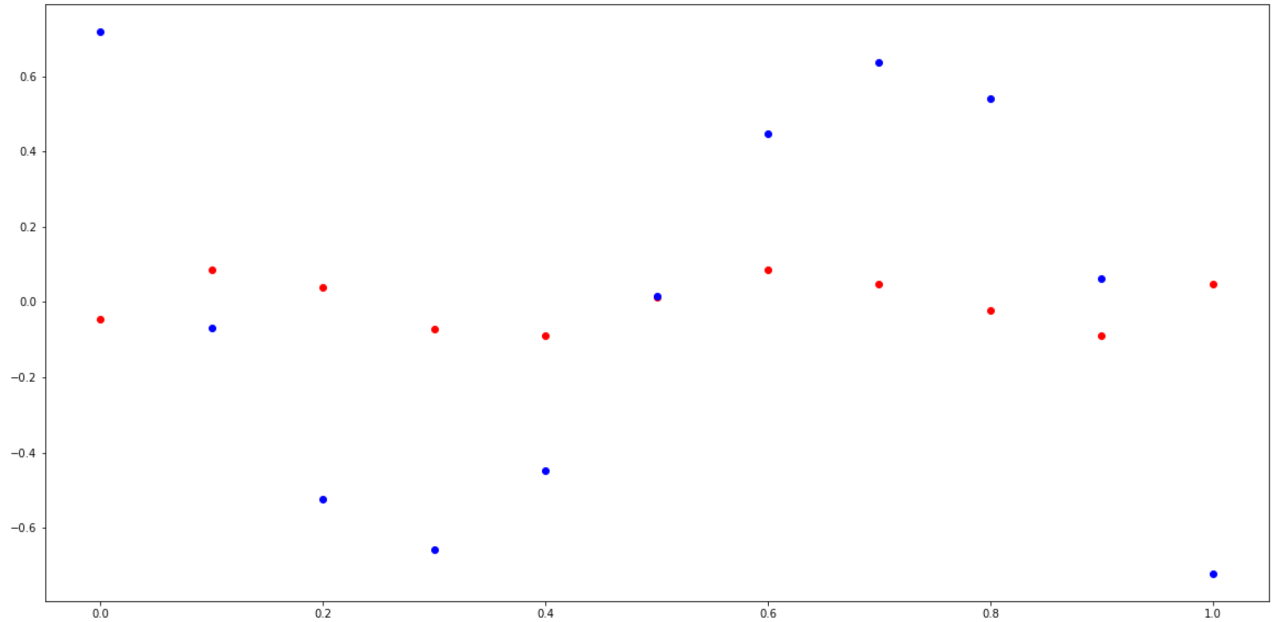
In [14]:
```
np.polyfit(X,Y,3,full=True)
```

Out[14]:
```
(array([ 2.13163290e+01, -3.19603750e+01,  1.07580786e+01, -4.59060260e-03]),
 array([0.04487783]),
 4,
 array([1.88678101, 0.6389302 , 0.17651721, 0.02583192]),
 2.4424906541753444e-15)
```

To further see this we compare the errors in both models (blue for degree 1 and red for degree 3)

In [15]:
```
plt.figure(figsize=(20,10))
plt.scatter(X, e3,color='red')
plt.scatter(X,e1,color='blue')
```

Out[15]:
```
<matplotlib.collections.PathCollection at 0x7f9c809e0070>
```


0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6
0.0
0.2
0.4
0.6
0.8
1.0