**1)**

**A)**

```
>> A=[1 1 3;1 2 1;1 1 -1;1 2 2]

A =

     1     1     3
     1     2     1
     1     1    -1
     1     2     2

>> b = [-2;0;1;4]

b =

    -2
     0
     1
     4

>> rref([A b])

ans =

     1     0     0     0
     0     1     0     0
     0     0     1     0
     0     0     0     1
```

**No, the system is inconsistent**

**B)**

```
>> A_pinv = inv(A'*A)*A'

A_pinv =

      15/17          -8/17          19/17          -9/17
     -21/34           9/17         -13/34           8/17
       4/17          -1/17          -4/17           1/17

>> x = A_pinv*b
x =
     -47/17
      93/34
      -8/17
```

**C)**

```
>> b- A*x
ans =
  -0.558823529411768
  -2.235294117647058
   0.558823529411764
   2.235294117647060
>> norm(ans)
ans =
   3.258473117707668
```

**3.2585**

**D)**

```
>> x1 = [1; 0; 0]
x1 =
     1
     0
     0
>> norm(b-A*x1)
ans =
   4.358898943540673
```
**approximately 4.3589**

```
>> x2 = [0;1;0]
x2 =
     0
     1
     0
>> norm(b-A*x2)
ans =
   4.123105625617661
```
**approximately 4.1231**

```
>> x3 = [0;0;1]
x3 =
     0
     0
     1
>> norm(b-A*x3)
ans =
   5.830951894845300
```
**approximately 5.8310**

**The answers overestimate, as they exceed the least square's solution's residual.**

**2)**

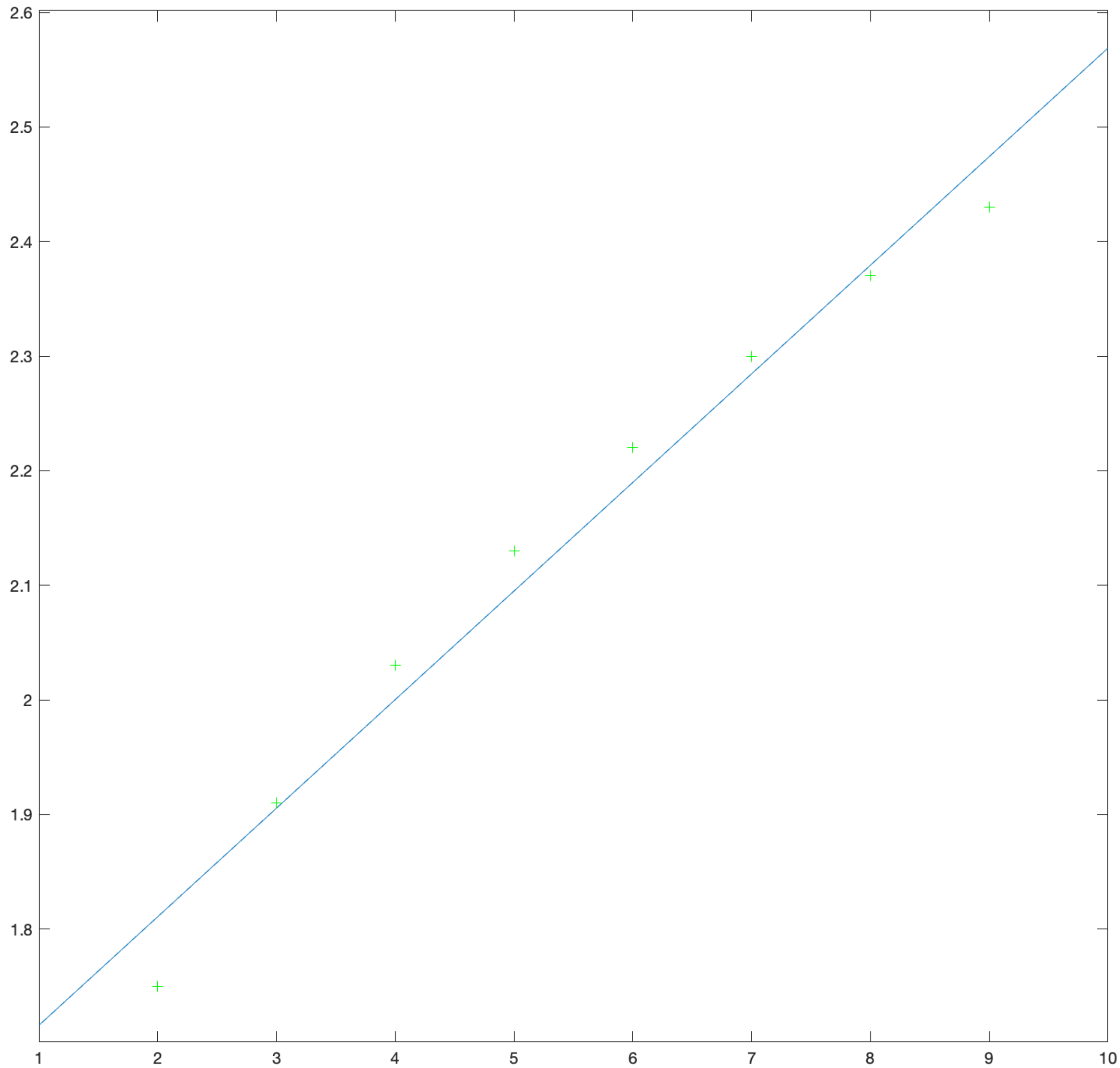
**A)**

```
>> xpoints = [2; 3; 4; 5; 6; 7; 8; 9]
xpoints =
     2
     3
     4
     5
     6
     7
     8
     9
>> A = [xpoints ones(8,1)]
A =
     2     1
     3     1
     4     1
     5     1
     6     1
     7     1
     8     1
     9     1
>> ypoints = [1.75; 1.91; 2.03; 2.13; 2.22; 2.3; 2.37; 2.43]
ypoints =
   1.750000000000000
   1.910000000000000
   2.030000000000000
   2.130000000000000
   2.220000000000000
   2.300000000000000
   2.370000000000000
   2.430000000000000

>> u = A\ypoints
u =
    199/2100
   1263/779
```

**B)**

```
>> x = linspace(1,10);
>> y = (199/2100)*x+1263/779;
>> plot(x,y,xpoints,ypoints,'g+')
```




**C)**

>> norm(ypoints-A*u)

ans =

**approximately 0.0949**

**3)**

A)

>> x = [2; 3; 4; 5; 6; 7; 8; 9]

x =

2
3
4
5
6
7

```
     8
     9
>> A = [(x.^2) x ones(8,1)]
A =
     4     2     1
     9     3     1
    16     4     1
    25     5     1
    36     6     1
    49     7     1
    64     8     1
    81     9     1

>> y = [1.75; 1.91; 2.03; 2.13; 2.22; 2.3; 2.37; 2.43]
y =
   1.750000000000000
   1.910000000000000
   2.030000000000000
   2.130000000000000
   2.220000000000000
   2.300000000000000
   2.370000000000000
   2.430000000000000

>> u = A\y
```

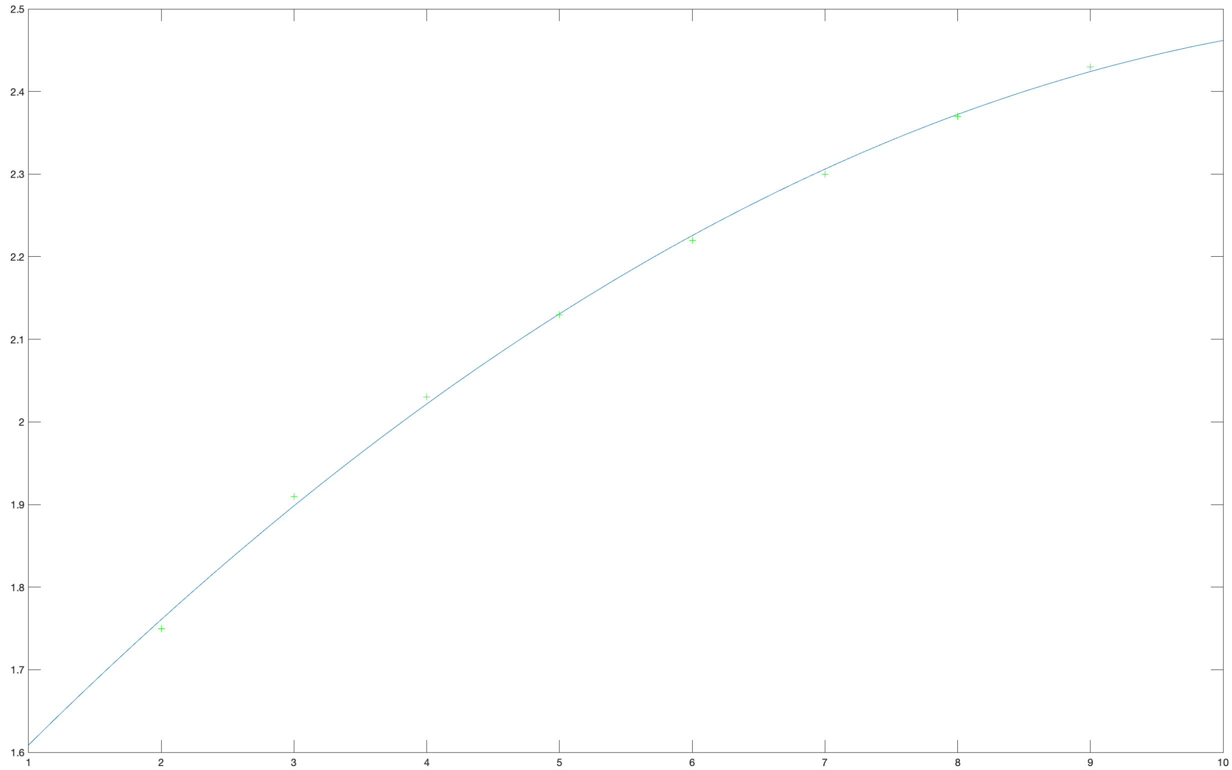
**u =**

```
      -1/140
      13/75
    1033/716
```

```
>> x = linspace(1,10);
>> y = (-1/140)*(x.^2) + (13/75)*x + 1033/716;
>> xpoints = [2; 3; 4; 5; 6; 7; 8; 9]
xpoints =
       2
       3
       4
       5
       6
       7
       8
       9
>> ypoints = [1.75; 1.91; 2.03; 2.13; 2.22; 2.3; 2.37; 2.43]
ypoints =
       7/4
     191/100
     203/100
     213/100
```

111/50
 23/10
237/100
243/100

```
>> plot(x,y,xpoints,ypoints,'g+')
```



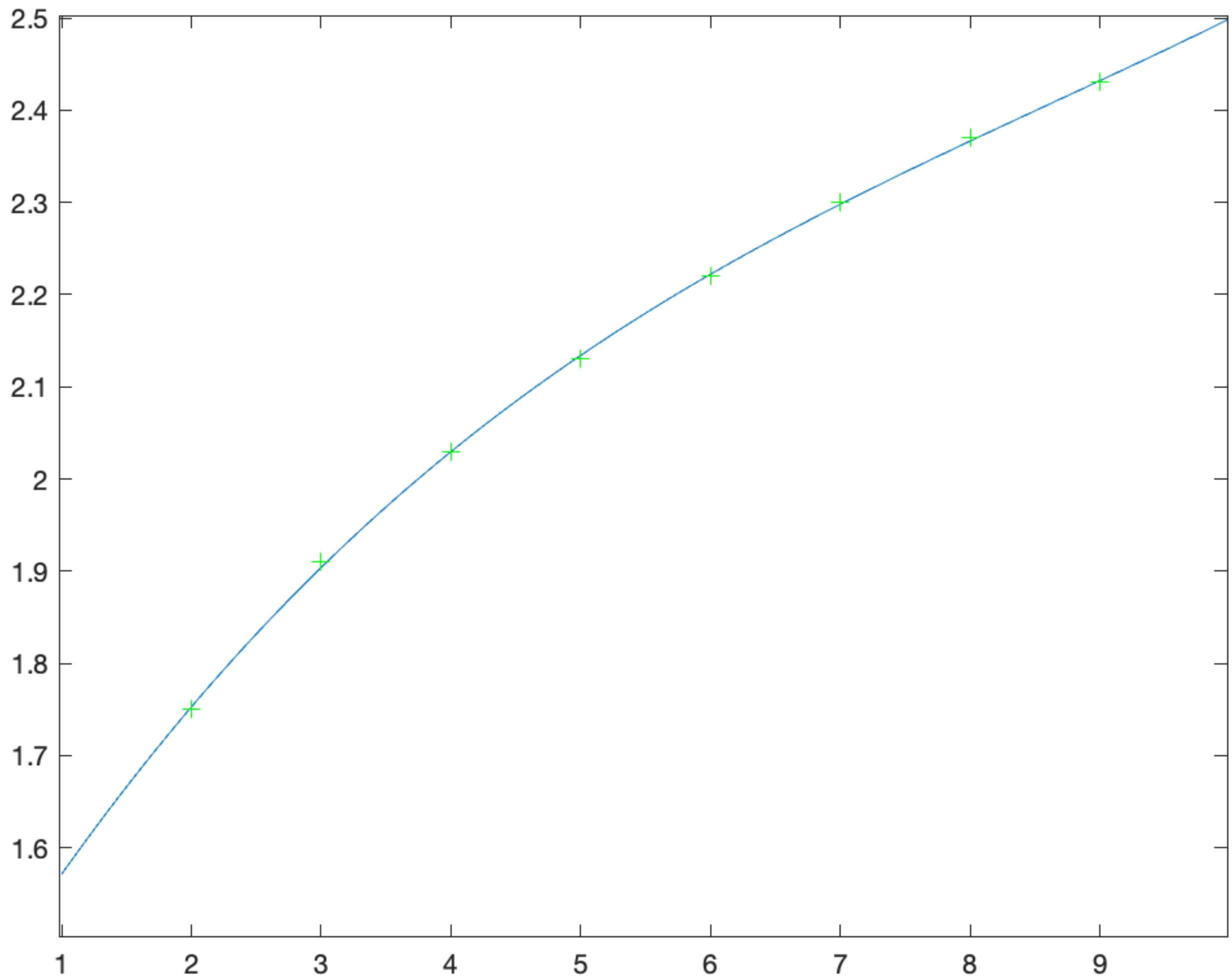
```
>> norm(y-A*u)
ans =
   0.020644381225662
```

**approximately 0.0206**

B)

```
>> x = [2; 3; 4; 5; 6; 7; 8; 9]
x =
     2
     3
     4
     5
     6
     7
     8
     9
>> A = [x.^3 x.^2 x ones(8,1)]
A =
     8     4     2     1
    27     9     3     1
```

```
      64              16              4             1
     125              25              5             1
     216              36              6             1
     343              49              7             1
     512              64              8             1
     729              81              9             1
>> y = [1.75; 1.91; 2.03; 2.13; 2.22; 2.3; 2.37; 2.43]
y =
       7/4
     191/100
     203/100
     213/100
     111/50
      23/10
     237/100
     243/100
>> u=A/y
Error using  /
Matrix dimensions must agree.
>> u=A\y
u =
       1/1320
     -11/560
     760/3233
    1423/1050
>> x = linspace(1,10);
>> y = (1/1320)*(x.^3)+(-11/560)*(x.^2)+(760/3233)*(x)+(1423/1050);
>> plot(x,y,xpoints,ypoints,'g+')
```




>> norm(y-A*u)
ans =
  0.009234792108185 **approximately 0.0092**

C)
**The magnitude of the residuals are becoming smaller as we increase the order of our of equations to represent to the line of best fit. I expected this, as the more unknowns we introduce, the closer we will get to finding a line which goes through exactly all the points.**

**4)**

A)
X = ln(x)
Y = ln(Y)

y = a*x^b
ln(y) = ln(a*x^b)
ln(y) = ln(a) + ln(x^b)
ln(y) = b*ln(x) + ln(a)

Y = b*X + ln(a)

B)

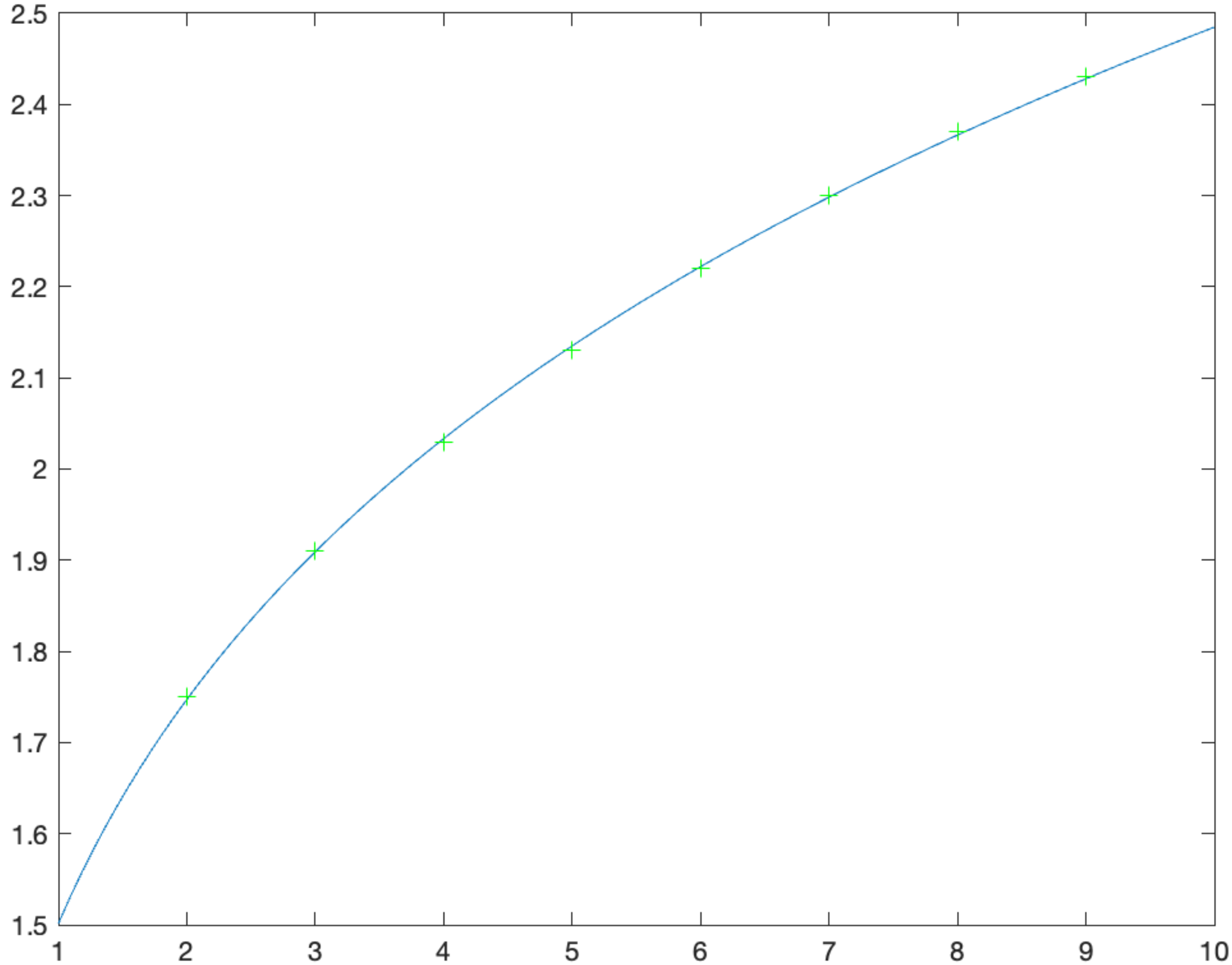
```
>> xpoints = [2; 3; 4; 5; 6; 7; 8; 9]
xpoints =
     2
     3
     4
     5
     6
     7
     8
     9
>> ypoints = [1.75; 1.91; 2.03; 2.13; 2.22; 2.3; 2.37; 2.43]
ypoints =
   1.750000000000000
   1.910000000000000
   2.030000000000000
   2.130000000000000
   2.220000000000000
   2.300000000000000
   2.370000000000000
   2.430000000000000
>> X = log(xpoints)
X =
   0.693147180559945
   1.098612288668110
   1.386294361119891
   1.609437912434100
   1.791759469228055
   1.945910149055313
   2.079441541679836
   2.197224577336220
>> Y = log(ypoints)
Y =
   0.559615787935423
   0.647103242058538
   0.708035793053696
   0.756121979721334
   0.797507195884188
   0.832909122935104
   0.862889955147040
   0.887891257352457
>> A = [X ones(8,1)]
A =
   0.693147180559945   1.000000000000000
   1.098612288668110   1.000000000000000
```

```
   1.386294361119891   1.000000000000000
   1.609437912434100   1.000000000000000
   1.791759469228055   1.000000000000000
   1.945910149055313   1.000000000000000
   2.079441541679836   1.000000000000000
   2.197224577336220   1.000000000000000
>> u = A\Y
u =
   0.218803396165447
   0.406373875540226
>> b = 0.218803396165447
b =
   0.218803396165447
>> a = exp(0.406373875540226)
a =
   1.501363770729448
```

***b*** is approximately **0.218803**
***a*** is approximately **1.501363**

C)

```
>> x = linspace(1,10);
>> y = a*(x.^b);
>> plot(x,y,xpoints,ypoints,'g+')
```

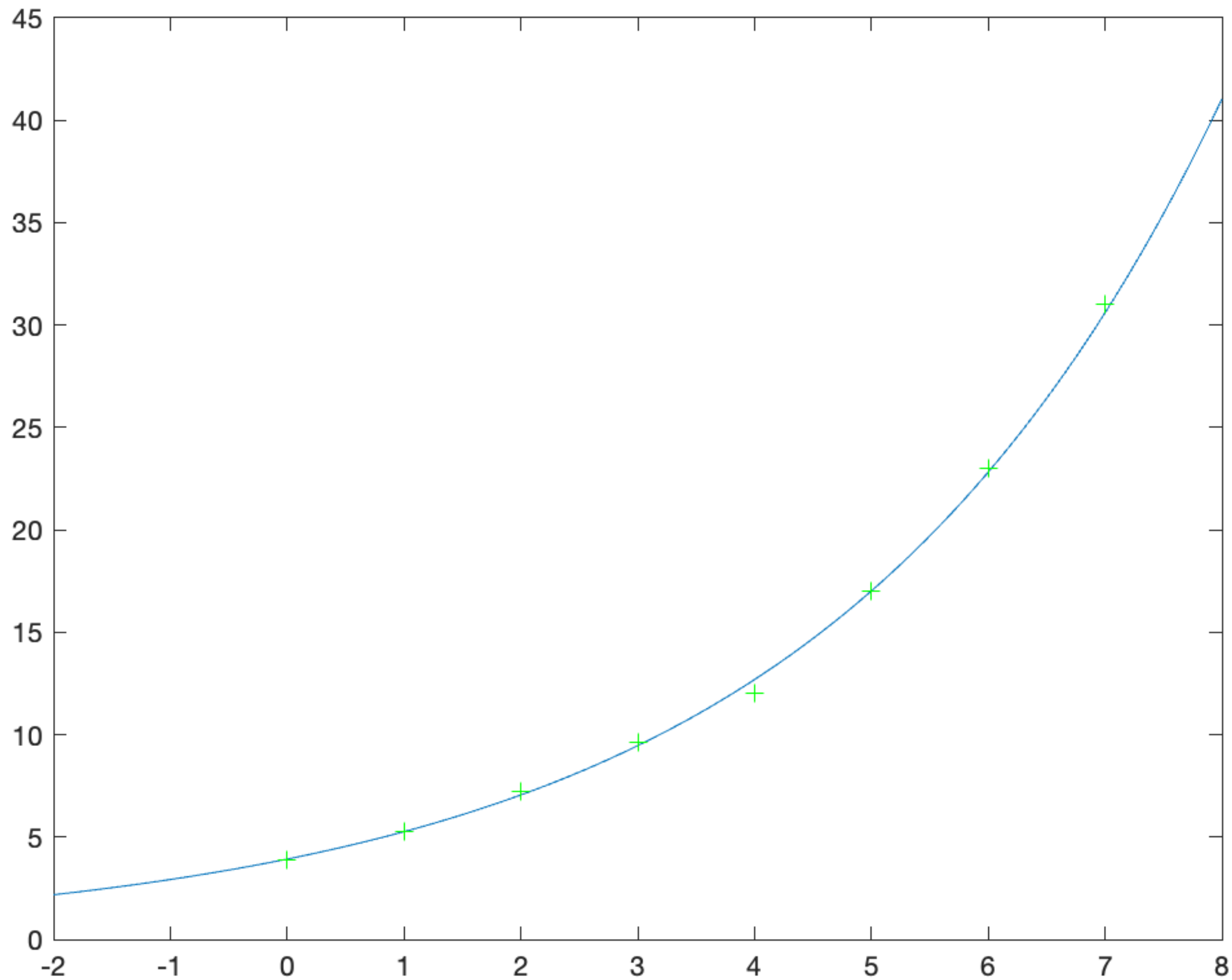



5)
Y = ln y

ln y = bx + ln a
Y = bx + ln a

```
>> xpoints = [0;1;2;3;4;5;6;7]
xpoints =
     0
     1
     2
     3
     4
     5
     6
     7
>> ypoints = [3.9;5.3;7.2;9.6;12;17;23;31]
ypoints =
   3.900000000000000
```

```
   5.300000000000000
   7.200000000000000
   9.600000000000000
  12.000000000000000
  17.000000000000000
  23.000000000000000
  31.000000000000000
>> Y = log(ypoints)
Y =
   1.360976553135601
   1.667706820558076
   1.974081026022010
   2.261763098473791
   2.484906649788000
   2.833213344056216
   3.135494215929150
   3.433987204485146
>> A = [xpoints ones(8,1)]
A =
     0     1
     1     1
     2     1
     3     1
     4     1
     5     1
     6     1
     7     1
>> u = A\Y
u =
   0.293458952877607
   1.366909778984373
>> b = 0.293458952877607
b =
   0.293458952877607
>> a = exp(1.366909778984373)
a =
   3.923208362955756
>> x = linspace(-2,8);
>> y = a*exp(b*x);
>> plot(x,y,xpoints,ypoints,'g+')
```



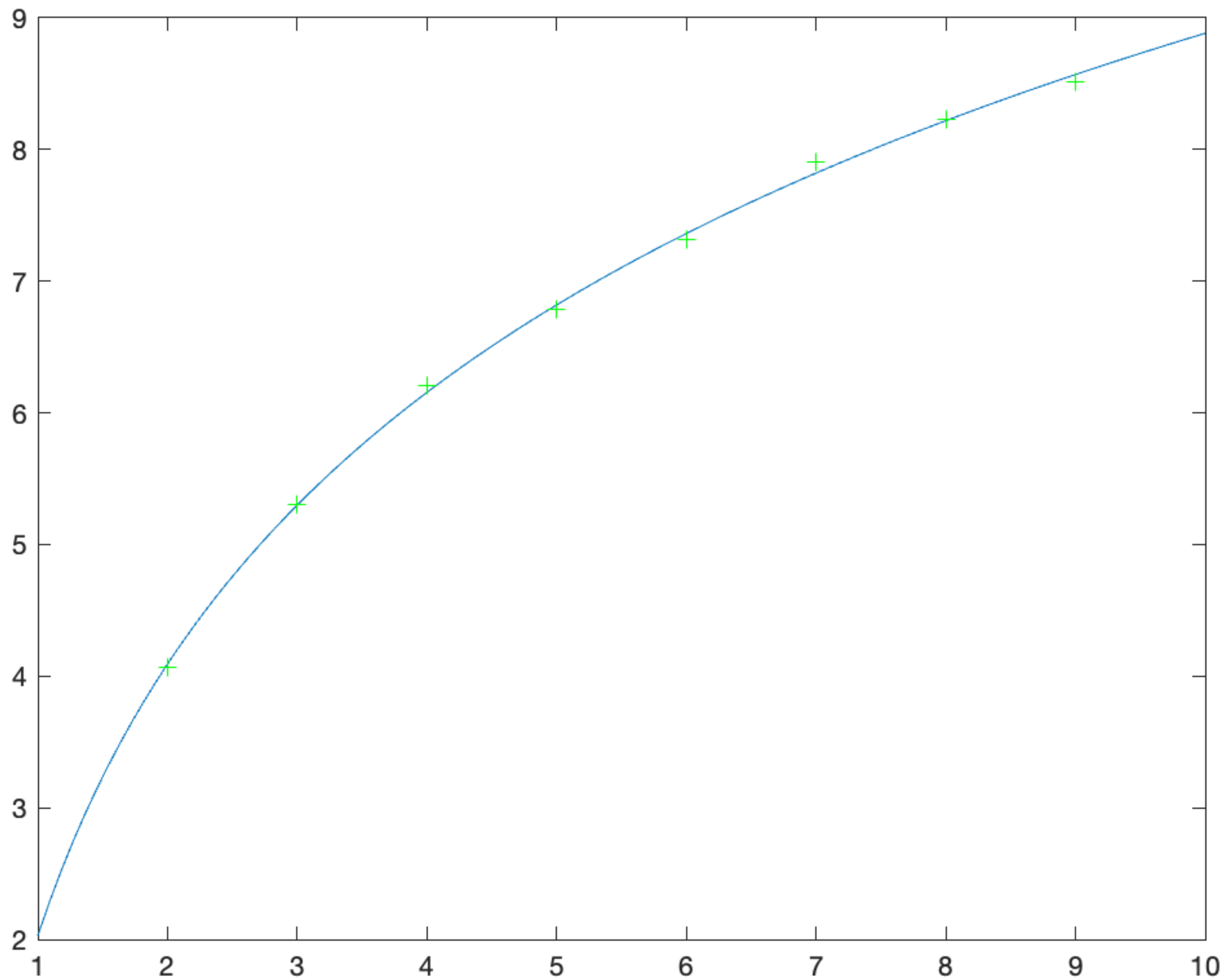

6)

X = ln(x)

y = a + b*ln(x)
y = b*X + a

```
>> xpoints = [2;3;4;5;6;7;8;9]
xpoints =
    2
    3
    4
    5
    6
    7
    8
    9
>> ypoints = [4.07;5.3;6.21;6.79;7.32;7.91;8.23;8.51]
ypoints =
```

```
   4.070000000000000
   5.300000000000000
   6.210000000000000
   6.790000000000000
   7.320000000000000
   7.910000000000000
   8.230000000000000
   8.510000000000000
>> X = log(xpoints)
X =
   0.693147180559945
   1.098612288668110
   1.386294361119891
   1.609437912434100
   1.791759469228055
   1.945910149055313
   2.079441541679836
   2.197224577336220
>> A = [X ones(8,1)]
A =
   0.693147180559945   1.000000000000000
   1.098612288668110   1.000000000000000
   1.386294361119891   1.000000000000000
   1.609437912434100   1.000000000000000
   1.791759469228055   1.000000000000000
   1.945910149055313   1.000000000000000
   2.079441541679836   1.000000000000000
   2.197224577336220   1.000000000000000
>> u = A\ypoints
u =
   2.976823596711755
   2.028902234532560
>> b = 2.976823596711755
b =
   2.976823596711755
>> a = 2.028902234532560
a =
   2.028902234532560
>> x = linspace(1,10);
>> y = a + b*log(x);
>> plot(x,y,xpoints,ypoints,'g+')
```




7)

A)

```
>> A = [1 3 2;3 1 2; 1 3 2; 3 1 2]
A =
     1     3     2
     3     1     2
     1     3     2
     3     1     2
>> b = [1; -1; 3; -2]
b =
     1
    -1
     3
    -2
>> rref([A b])
ans =
     1     0     1/2     0
```

```
     0        1        1/2      0
     0        0        0        1
     0        0        0        0
```

**The system is not consistent**

B)

```
>> x_ls = A\b

x_ls =

   -13/16
    15/16
     0

>> rref(A)
```

**Checking this least squares solution:**

```
>> LHS = A'*A

LHS =

   20   12   16
   12   20   16
   16   16   16

>> RHS = A'*b

RHS =

   -5
    9
    2

>> LHS*x_ls

ans =

  -5.0000
   9.0000
   2.0000
```

**Therefore, x_ls is a least squares solution and an initial solution for the parametric form. Since x3 is a free variable:**

```
>> rref(A)

ans =

       1              0              1/2
       0              1              1/2
       0              0              0
       0              0              0
```

x1 = (-1/2)x3
x2 = (-1/2)x3

x = x3[-1/2; -1/2; 1]

Therefore, all possible least square solutions is:

**Least square solutions = [-13/16; 15/16; 0] + x3[-1/2; -1/2; 1]**

C)

```
>> x_ls+[-1/2;-1/2;1]

ans =

    -21/16
      7/16
      1

>> x_ls+[-1;-1;2]

ans =

    -29/16
     -1/16
      2

>> x_ls+[-2;-2;4]

ans =

    -45/16
    -17/16
      4
```

**D)**

```
>> norm(b-A*(x_ls+[-1/2;-1/2;1]))
```

```
ans =

    1.5811

>> norm(b-A*(x_ls+[-2;-2;4]))

ans =

   1.5811

>> norm(b-A*(x_ls+[-1;-1;2]))

ans =

    1.5811
```

**All residuals are equal. This should be expected, because b-Ax represents the same vector in the Left Null space(Nul(A')). When calculated, least square solution x = particular solution + v(v is an element of the Nul(A)), therefore Distributing A to these parts results in A(particular solution) + 0(the product of A and any vector in its Nullspace is 0). Therefore, the residual calculation b-Ax will always result in the same vector, resulting in equal norm calculations.**