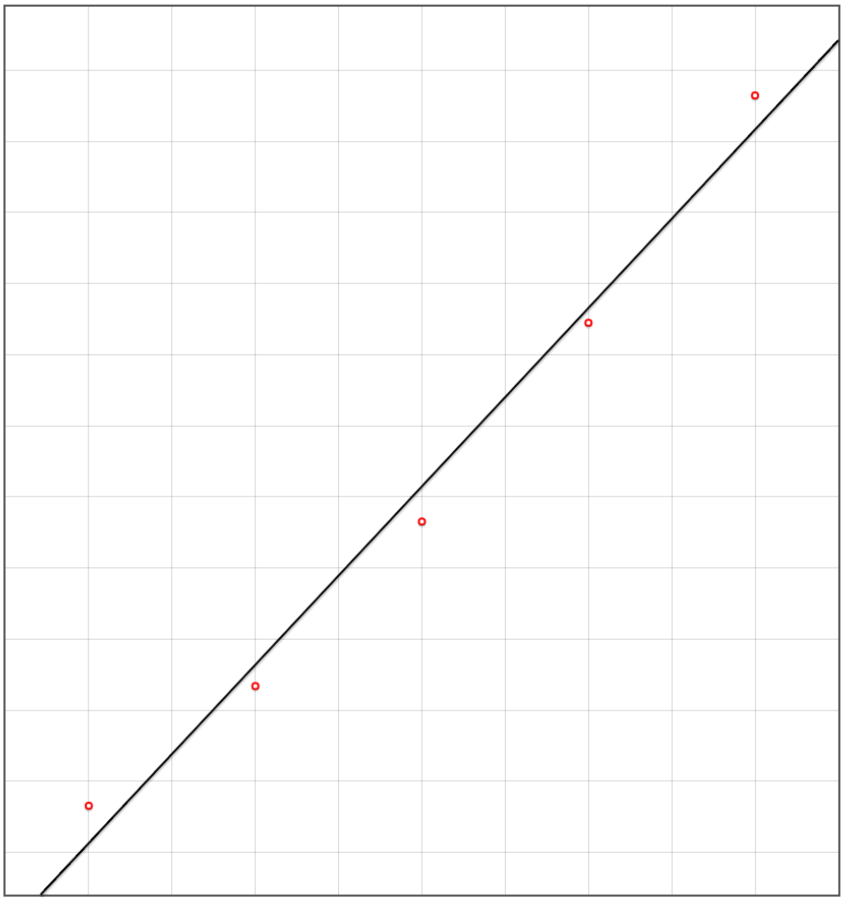
CSE3521 SP21
John Choi.1655
Austin Schall.37

1.



Function type: ◉ Line ○ Polynomial ○ Non-linear

Function: y = a*x + b

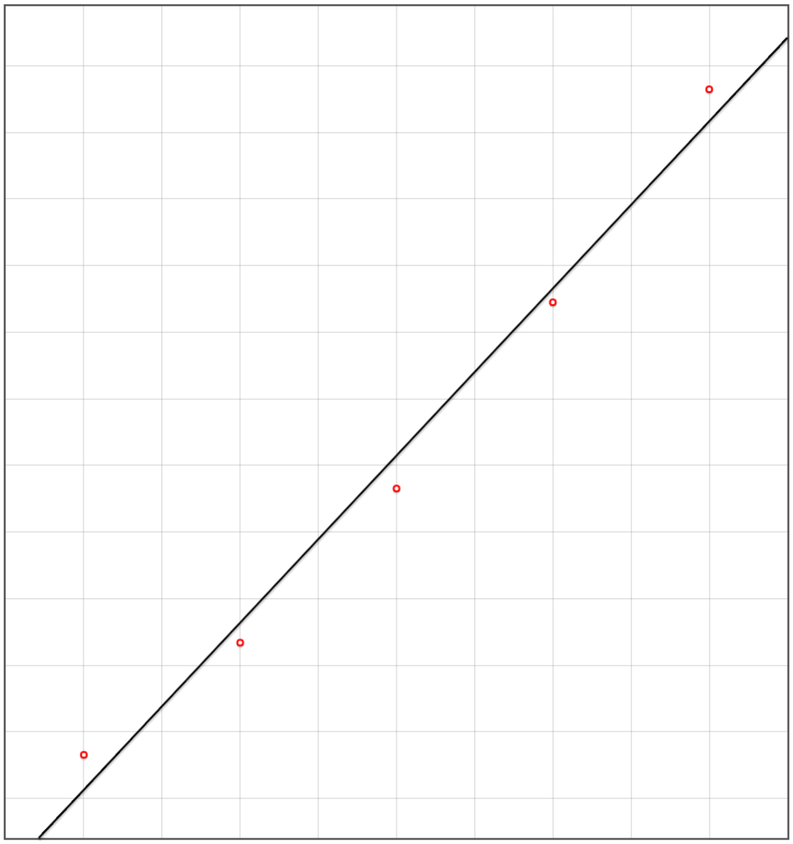
a 1.2547000000000001

b -0.5629500000000007

Fit From Data

SSE=0.22131790000000018

2.



Function type: ○ Line ◉ Polynomial ○ Non-linear

Order: 1

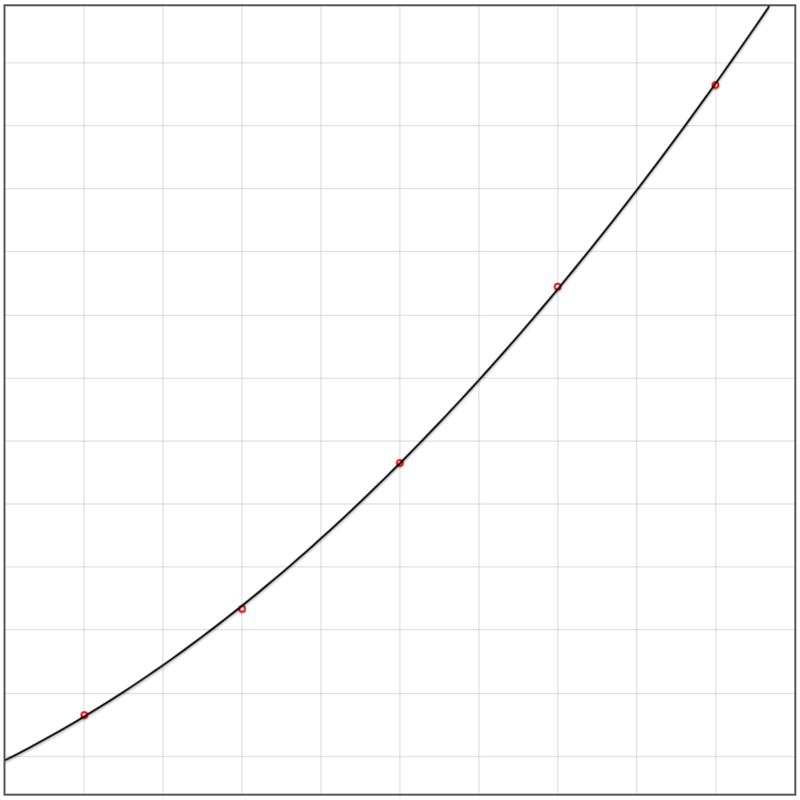
Function: a*x + b

a 1.2547000000000001

b -0.5629500000000007

Fit From Data

SSE=0.22131790000000018



Function type: ◯ Line ◉ Polynomial ◯ Non-linear

Order: 2

Function: $a*x^2 + b*x + c$

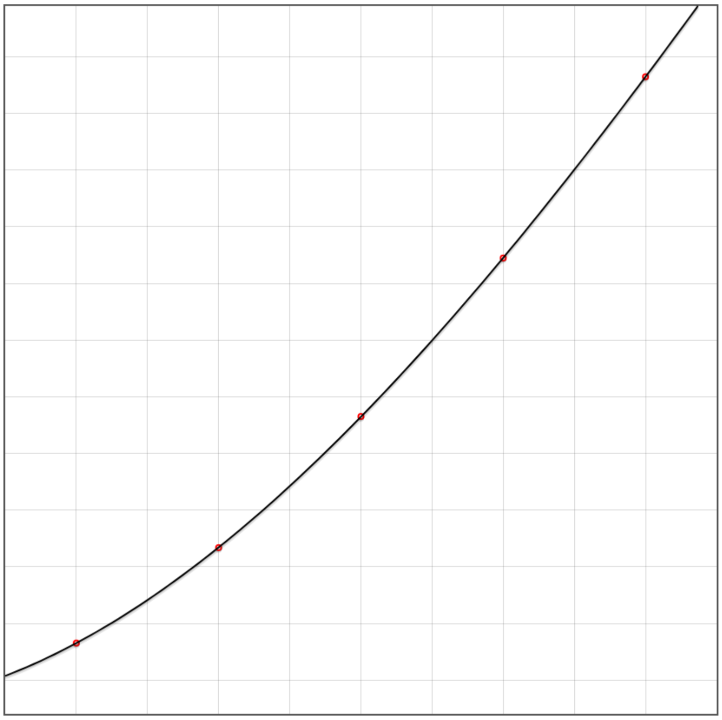
a 0.12535714285714228

b 0.6279142857142901

c -0.03018214285715004

Fit From Data

SSE=0.0013161142857142745



Function type: ○ Line ◉ Polynomial ○ Non-linear

Order: 3

Function: $a*x^3 + b*x^2 + c*x + d$

a -0.009499999999992959

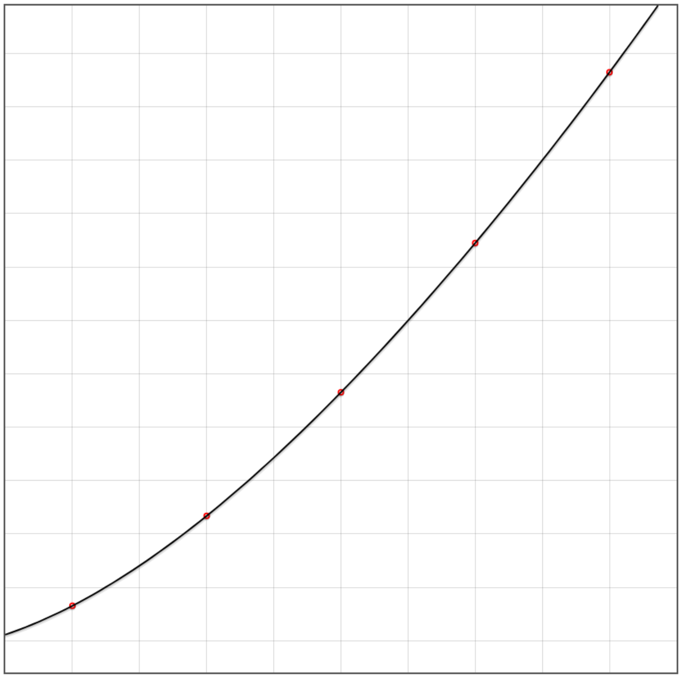
b 0.19660714285711833

c 0.4820892857144088

d 0.03750535714277703

Fit From Data

SSE=0.000016514285714284396



Function type: ○ Line ◉ Polynomial ○ Non-linear

Order: 4

Function: $a*x^4 + b*x^3 + c*x^2 + d*x + e$

a 0.001416666666528954

b -0.02366666666375749(

c 0.24345833333063638

d 0.42491666667592654

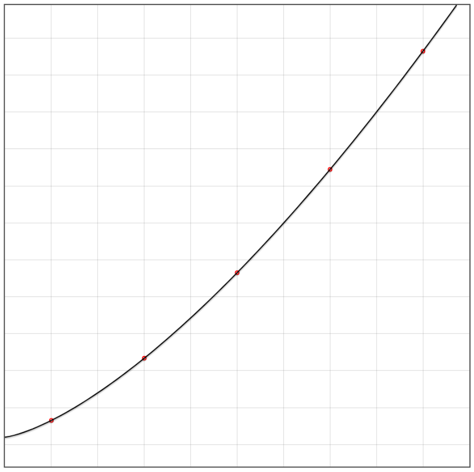
e 0.0565468749973661

Fit From Data

SSE=4.954960433513401e-20

f(x, a, b, c, d) is the graph that best fits the data because it has the lowest SSE value out of all 4 plots.

4. .



Function type: ○ Line ○ Polynomial ◉ Non-linear

Function: $y = a*x^b + c*x + d$

a 0.4938455104742545

b 1.5037635080613851

c 0.10730333930639983

d 0.09923420477446322

Gauss-Newton:

Fit From Data Iterations 10 ▶

Gradient Descent:

Fit From Data Iterations 5000

Learning rate 0.001

Iteration 0: SSE=86.91664599999999

Iteration 1: SSE=1.2257484260468634

Iteration 2: SSE=0.002951589411944793

Iteration 3: SSE=0.04503184217908566

Iteration 4: SSE=0.000004780982186876069

Iteration 5: SSE=2.2090802391677188e-7

Iteration 6: SSE=2.209078328567685e-7

Iteration 7: SSE=2.2090783285694642e-7

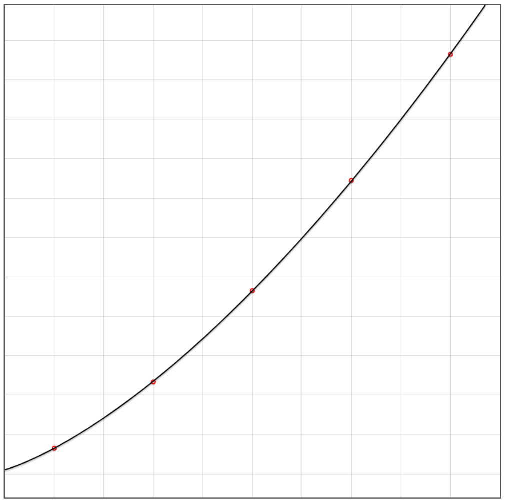
Iteration 8: SSE=2.209078328565876e-7

Iteration 9: SSE=2.209078328568952e-7

Iteration 10: SSE=2.2090783285673085e-7

This model clearly fits the data better than the previous ones because the SSE value after 10 iterations is significantly smaller. This indicates that the graph is more accurate.

5. J



Function type: ◯ Line ◯ Polynomial ◉ Non-linear

Function: $y = a*x^b + c*x + d$

a 0.2869086366665941

b 1.6822635934297665

c 0.37200221825057966

d 0.052247386352797826

Gauss-Newton:

Fit From Data Iterations 10 ▶

Gradient Descent:

Fit From Data Iterations 5000

Learning rate 0.001

Iteration 0: SSE=86.91664599999999

Iteration 1: SSE=9.823254279495549

Iteration 2: SSE=1.9886924028316724

Iteration 3: SSE=0.6616441968742724

Iteration 4: SSE=0.4614402093746779

Iteration 5: SSE=0.4304949859231547

Iteration 6: SSE=0.4217574968881253

Iteration 7: SSE=0.4157848284023885

Iteration 8: SSE=0.4101990138398179

Iteration 9: SSE=0.4047189191749914

Iteration 10: SSE=0.39931134666846885

Iteration 11: SSE=0.393971894688973

Iteration 12: SSE=0.388699485064214

Iteration 4990: SSE=0.00020958159394776103

Iteration 4991: SSE=0.00020952622362921898

Iteration 4992: SSE=0.00020947092333323868

Iteration 4993: SSE=0.00020941569296893233

Iteration 4994: SSE=0.00020936053244554747

Iteration 4995: SSE=0.00020930544167243313

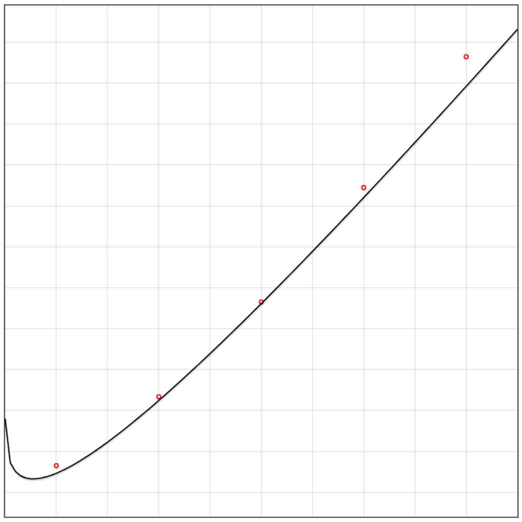
Iteration 4996: SSE=0.00020925042055904762

Iteration 4997: SSE=0.00020919546901501445

Iteration 4998: SSE=0.00020914058695000246

Iteration 4999: SSE=0.00020908577427386632

Iteration 5000: SSE=0.00020903103089654364



Function type: ○ Line ○ Polynomial ◉ Non-linear

Function: $y = a*x^b + c*x + d$

a -1.969627173604088

b 0.38339370208118523

c 1.6808523207730635

d 0.9028580953723963

Gauss-Newton:

Fit From Data Iterations 10 ▶

Gradient Descent:

Fit From Data Iterations 5000

Learning rate 0.01

Iteration 0: SSE=86.91664599999999

Iteration 1: SSE=267.4071557770459

Iteration 2: SSE=19.951018043894518

Iteration 3: SSE=9.826605940838439

Iteration 4: SSE=6.1133148695126565

Iteration 5: SSE=4.163624565634833

Iteration 6: SSE=2.970914926677272

Iteration 7: SSE=2.1803543655625495

Iteration 8: SSE=1.6318343950624976

Iteration 9: SSE=1.2383181948305062

Iteration 10: SSE=0.9502028893332372

Iteration 4990: SSE=0.15756033460934982

Iteration 4991: SSE=0.16706502020153474

Iteration 4992: SSE=0.15755042450067047

Iteration 4993: SSE=0.16705396629675734

Iteration 4994: SSE=0.15754052416667694

Iteration 4995: SSE=0.16704292329263734

Iteration 4996: SSE=0.1575306335967123

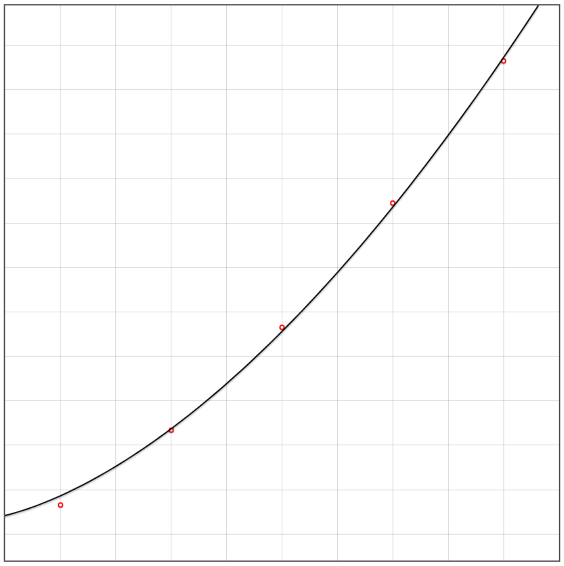
Iteration 4997: SSE=0.16703189117727776

Iteration 4998: SSE=0.15752075278013405

Iteration 4999: SSE=0.1670208699388045

Iteration 5000: SSE=0.1575108817063168

Keeping the same iterations but with higher learning rate caused the convergence rate to be higher as well. This resulted in a graph with a more inaccurate line of best fit.



Function type: ○ Line ○ Polynomial ◉ Non-linear

Function: $y = a*x^b + c*x + d$

a 0.25616477549893796

b 1.7955950433925965

c 0.29767394272532177

d 0.208385785314082

Gauss-Newton:

Fit From Data Iterations 10 ▶

Gradient Descent:

Fit From Data Iterations 500

Learning rate 0.001

Iteration 0: SSE=86.91664599999999

Iteration 1: SSE=9.823254279495549

Iteration 2: SSE=1.9886924028316724

Iteration 3: SSE=0.6616441968742724

Iteration 4: SSE=0.4614402093746779

Iteration 5: SSE=0.4304949859231547

Iteration 6: SSE=0.4217574968881253

Iteration 7: SSE=0.4157848284023885

Iteration 8: SSE=0.4101990138398179

Iteration 9: SSE=0.4047189191749914

Iteration 10: SSE=0.39931134666846885

Iteration 11: SSE=0.393971894688973

Iteration 488: SSE=0.017035307424790272

Iteration 489: SSE=0.017009498455388655

Iteration 490: SSE=0.016983756925682162

Iteration 491: SSE=0.016958082227549343

Iteration 492: SSE=0.016932473762659803

Iteration 493: SSE=0.016906930942303645

Iteration 494: SSE=0.016881453187223828

Iteration 495: SSE=0.016856039927450474

Iteration 496: SSE=0.01683069060213973

Iteration 497: SSE=0.016805404659413763

Iteration 498: SSE=0.01678018155620513

Iteration 499: SSE=0.016755020758102777

Iteration 500: SSE=0.016729921739201292

Keeping the same learning rate but with lower iteration number also caused the graph to result in a more inaccurate line of best fit with a higher SSE value.