CS113 - Skill Builder 7

Minerva University

CS113: Theory and Applications of Linear Algebra

October 30, 2022

CS113 Skill Builder 8

1) a) To find the needed basis we can consider the following:

   - To find the basis for Col(A): find a set of linearly independent columns
   - To find the basis for N(A): find a set of solutions to Ax = 0
   - To find the basis for Row(A): find a set of linearly independent rows
   - To find the basis for N(A^T): find a set of solutions to A^Ty = 0
   - dim(N(A)) + dim(Row(A)) = dim(R^n)
   - dim(Col(A)) + dim(N(AT)) = dim(R^m)

We then find the following:

Col(A): {(0,1,2), (-1,0,1)}, Dimension: 2

N(A): {(-2,-2,-2), (1,3,5)}, Dimension: 2

Row(A): {(0,-1,-2,1), (1,0,-2,3)}, Dimension: 2

N(A^T): {(2,1,-2,5)}, Dimension: 1

Col(U): {(1,0,0), (0,1,0)}, Dimension: 2

N(U): {(-2,2,0), (3,-1,0)}, Dimension: 2

Row(U): {(1,0,-2,3), (0,1,2,-1)}, Dimension: 2

N(A^T): {(0,0,0,0)}, Dimension: 1

b) All four fundamental subspaces of A have the same dimensions as the fundamental spaces of U. This is because U is the RREF of A. The RREF is a tool for us to extract the dimensions of the fundamental spaces of A. For us to extract the ones from U we would do its RREF too. Since U is already in RREF it will then have the same characteristics as A.

c) The row has the same ??

2) a) b = eggs hatched

   x = collection of unknowns

   A = times that the eggs hatched

   Parabola equation: $b = 1C + Dt + Et^2$

   System of equations:

   $Ax = b$

   $A = [[1,0.5, 0.5^2], [1, 0.75, 0.75^2], [1, 1, 1^2], [1, 1.25, 1.25^2], [1, 1.5, 1.5^2], [1, 1.75, 1.75^2], [1, 3, 3^2], [1, 3.25, 3.25^2], [1, 3.5, 3.5^2], [1, 3.75, 3.75^2], [1,4, 4^2]]$

   $x = (C, D, E)$

   $b = (1, 3, 2, 2, 1, 5, 6,7, 9, 15, 14)$

   xˆ is the projection of b into the column space of A. x^ is a component of b.

```
# question 2

A = matrix(QQ, [[1,0.5, 0.5^2], [1, 0.75, 0.75^2], [1, 1, 1^2], [1, 1.25, 1.25^2], [1, 1.5, 1.5^2], [1, 1.75, 1.75^2],
b = vector([1, 3, 2, 2, 1,  5, 6,7, 9, 15, 14])


A.augment(b, subdivide=True).rref()
```

```
[1 0 0|0]
[0 1 0|0]
[0 0 1|0]
[0 0 0|1]
[0 0 0|0]
[0 0 0|0]
[0 0 0|0]
[0 0 0|0]
[0 0 0|0]
[0 0 0|0]
[0 0 0|0]
```

```
# To find the closest solution to the real one we can use the following relation: (A^T)Ax = (A^T)b
A_trans = A.transpose()

# left side of the equation
approx = A_trans*A
approx
```

```
[        11       97/4    1129/16]
[      97/4    1129/16   14923/64]
[   1129/16   14923/64 208549/256]
```

```
# right side
b_approx = A_trans*b
b_approx
```

```
(65, 202, 2791/4)
```

```
# combining them on a matrix
result_approx = (approx.augment(b_approx).rref())
n(result_approx)
```

```
[ 1.00000000000000 0.000000000000000 0.000000000000000  3.62142214221422]
[0.000000000000000  1.00000000000000 0.000000000000000 -3.24163354388536]
[0.000000000000000 0.000000000000000  1.00000000000000  1.47066742072437]
```

```
# using the approx numerical values we can find our parabola

var('t')

# Parabola equation: b = 1C + Dt + Et^2
parabola = 3.621*1 - 3.242*t + 1.471*(t^2)
```

```
s = scatter_plot([[0.5,1], [0.75,3],[1,2],[1.25,2],[1.5,1],[1.75,5],[3,6],[3.25,7],[3.5,9],[3.75,15],[4,14]])

prbl = plot(parabola, (0,5))

s + prbl
```

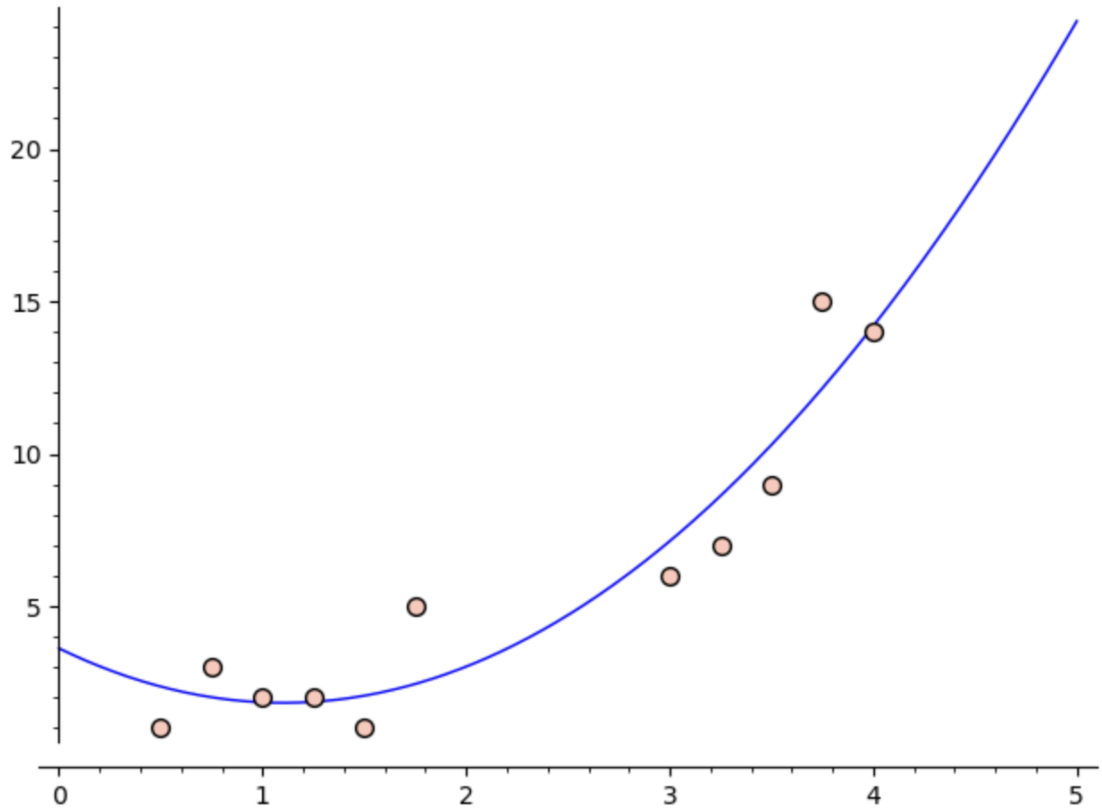

Figure 1: Scatter plot with the closest fitting parabola to the actual solution. Y-axis is the number of cracked eggs and the X-axis is the time.

We follow similar steps to find the regression line so then we can plot it with our parabola.

```
#Solve for x_hat -> regression line

A = matrix(QQ,[[1,0.5],[1, 0.75],[1, 1], [1, 1.25], [1, 1.5], [1, 1.75], [1, 3],[1, 3.25], [1, 3.5], [1, 3.75], [1,4]])

b = vector([1, 3, 2, 2, 1,  5, 6,7, 9, 15, 14])


A.augment(b, subdivide=True).rref()
```

```
[1 0|0]
[0 1|0]
[0 0|1]
[0 0|0]
[0 0|0]
[0 0|0]
[0 0|0]
[0 0|0]
[0 0|0]
[0 0|0]
[0 0|0]
```

```
# To find the closest solution to the real one we can use the following relation: (A^T)Ax = (A^T)b
A_trans = A.transpose()

# left side of the equation
approx = A_trans*A
approx
```

```
[     11    97/4]
[   97/4 1129/16]
```

```
# right side
b_approx = A_trans*b
b_approx
```

```
(65, 202)
```

```
# combining them on a matrix
result_approx = (approx.augment(b_approx).rref())
n(result_approx)
```

```
[ 1.00000000000000 0.000000000000000 -1.65813953488372]
[0.000000000000000  1.00000000000000   3.43255813953488]
```

```
s = scatter_plot([[0.5,1],[0.75,3],[1,2], [1.25, 2], [1.5, 1], [1.75, 5], [3,6],[3.25, 7], [3.5, 9], [3.75, 15], [4,14]
regline = -1.658 + 3.432*t

reg_plot = plot(regline, (0,5), color='red')

s + reg_plot + prbl
```

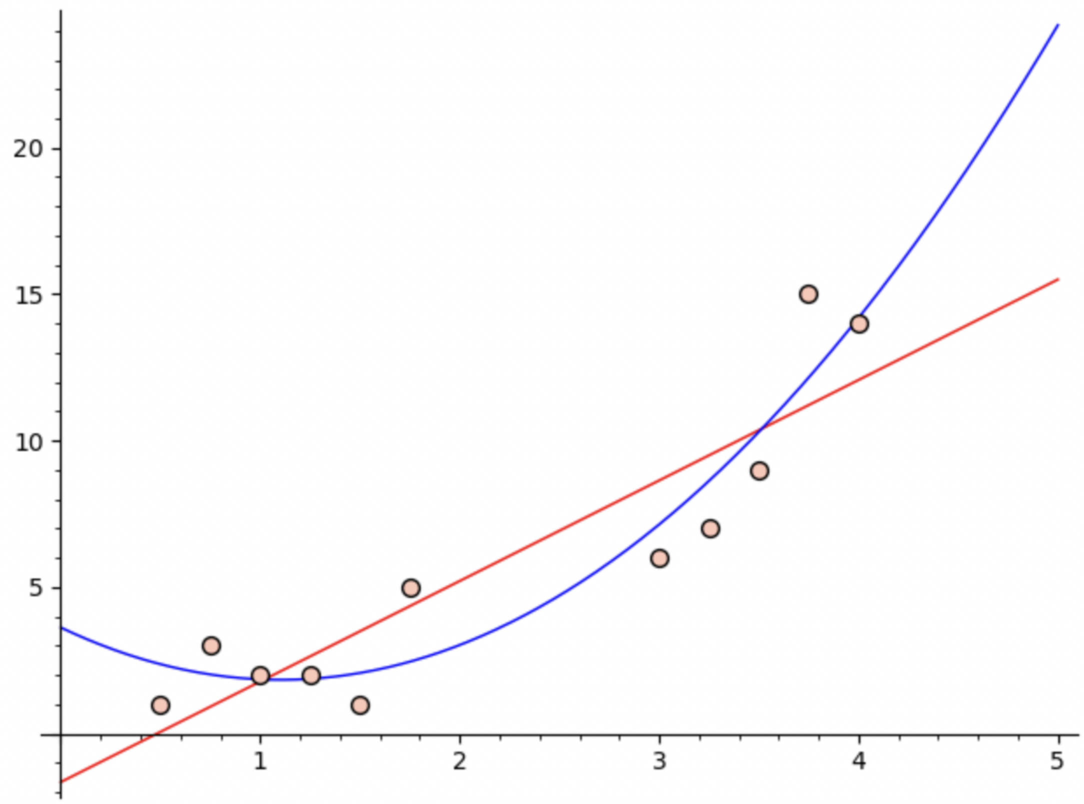

Figure 2: Scatter plot with the closest fitting parabola to the actual solution and the regression line. Y-axis is the number of cracked eggs and the X-axis is the time.

As we can see, the parabola fitted better the scatter plot, coming closer to the actual results.

c) We know we are getting to the closest solution we can using our matrix A. Being the closest solution and not the solution itself we can conclude that the vector b won't be on the column space spanned by A. However, its projection will be on the column space of A (x hat), since b can be geometrically built by the combination of different vectors (components) in other spaces and x hat is one of its possible components.