

406_a2

January 30, 2018

Kyle Leeners, 30591119, w0u0b

1 Exercise 1

1.0.1 a)

$$A = \begin{bmatrix} 1 & x_1^1 & x_2^1 \\ 1 & x_1^2 & x_2^2 \\ \vdots & & \\ 1 & x_1^n & x_2^n \end{bmatrix}, x = \begin{bmatrix} c_0 \\ c_1 \\ c_2 \end{bmatrix}, b = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}$$

1.0.2 b)

```
In [4]: data = CSV.read("C:\\\\Users\\\\Kyle\\\\Desktop\\\\lsq_classification.csv"; datarow=1)
      mtx = convert(Array, data)
      n = size(mtx, 1)
      c = mtx[:,1:2]
      A = hcat(ones(n,1), mtx[:,1:2])
      betas = mtx[:,3:3]
```

Out[4]: 115×1 Array{Float64,2}:

-1.0
-1.0
-1.0
-1.0
-1.0
-1.0
-1.0
-1.0
-1.0
-1.0

i)

```
In [5]: # Solution via normal equation A^T Ax = A^T b  
xls = \(*(transpose(A),A), *(transpose(A),betas))
```

Out [5]: 3×1 Array{Float64,2}:

-0.368611
0.576988
0.63353

ii)

```
In [6]: # Solution via QR
```

```

QR = qrfact(A)
Q = QR[:Q]
R = QR[:R]
m = size(R,1)

```

```

y = *(transpose(Q), betas)
y = y[1:m]
xqr = \ (R, y)

```

Out [6]: 3-element Array{Float64,1}:

-0.368611
0.576988
0.63353

1.0.3 c)

```
In [7]: # define coefficients and function
```

```
c0, c1, c2 = reshape(xls[1:1],1)[1], reshape(xls[2:2],1)[1], reshape(xls[3:3],1)[1]
f(x1) = (-c0 -c1*x1) / c2
```

```
# # # (x1,x2) where label = 1
111 = mtx[:,1:1][mtx[:,3:3] == 1]
112 = mtx[:,2:2][mtx[:,3:3] == 1]
```

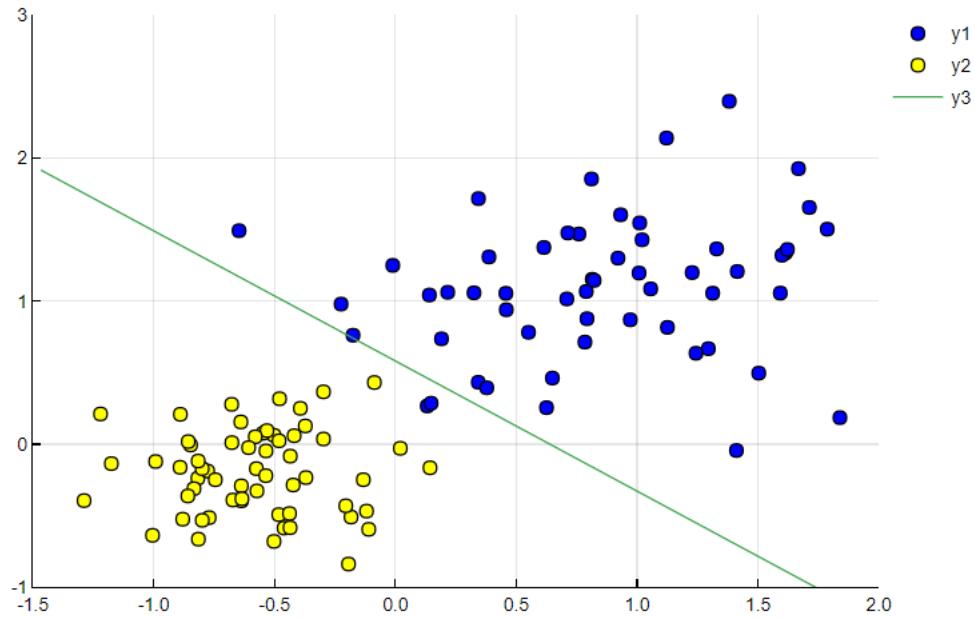
(x_1, x_2) where label = -1

```

121 = mtx[:,1:1] [mtx[:,3:3] .== -1]
122 = mtx[:,2:2] [mtx[:,3:3] .== -1]

# # # plot data
scatter(l11, l12, xlim=(-1.5,2), ylim=(-1, 3), colour="blue")
scatter!(l21, l22, colour="yellow")
plot!(f)

```



2 Exercise 2

1-norm

$$f(\alpha) = \|\alpha e - b\| = \sum_{i=1}^n |\alpha - b_i|$$

This shows that by minimizing $f(\alpha)$ we are really just minimizing the sum of the distances between a and the various values of b . To do this we should let $a = \text{median}(b)$

2-norm

$$\begin{aligned} \min f(\alpha) &= \|\alpha e - b\| \Rightarrow \min f(\alpha)^2 = \min \sum_{i=1}^n (\alpha - b_i)^2 \\ &\Rightarrow 2\alpha n - 2 \sum_{i=1}^n b_i \\ &\Rightarrow \alpha n = \sum_{i=1}^n b_i \\ &\Rightarrow \alpha = \frac{1}{n} \sum_{i=1}^n b_i \\ &\Rightarrow \alpha = \text{mean}(b) \end{aligned}$$

inf-norm

$$\min_\alpha f(\alpha) = \min_\alpha \{ \max_i \{ |\alpha - b_i| \} \} = \min_\alpha \{ \max_i \{ \alpha - b_i, b_i - \alpha \} \}$$

To maximize $\alpha - b_i$ we would pick the smallest b_i . To maximize $b_i - \alpha$ we would pick the largest b_i . Therefore, to minimize both components we should let $\alpha = \frac{\max b_i - \min b_i}{2}$

3 Exercise 3

$$f(x) = \frac{1}{2}(Ax - b)^T W(Ax - b) = \frac{1}{2}(Ax - b)^T QDQ(Ax - b)$$
$$= \frac{1}{2}(Ax - b)^T Q\sqrt{D}(\sqrt{D})^T Q(Ax - b)$$

Let $Q\sqrt{D} = P$

$$\Rightarrow \frac{1}{2}(Ax - b)^T PP^T(Ax - b)$$
$$= \frac{1}{2}(P^T Ax - P^T b)^T (P^T Ax - P^T b)$$
$$= \frac{1}{2}(Bx - d)^T (Bx - d)$$

4 Exercise 4

a) Since A is underdetermined and consistent, should pick x such that

$$\begin{aligned} & \text{minimize } \|x\|^2 \\ & \text{subject to : } Ax = b \end{aligned}$$

$$\begin{aligned} & \text{let } L(x, \mu) = \|x\|^2 - \mu^T(b - Ax) \\ & \frac{\partial}{\partial x} = 2x - u^T A \end{aligned}$$

$$\Rightarrow x = \frac{1}{2}A^T\mu$$

$$\frac{\partial}{\partial \mu} = b - Ax$$

$$\Rightarrow b = Ax$$

$$\Rightarrow b = A(\frac{1}{2}A^T\mu)$$

$$\Rightarrow \mu = 2(AA^T)^{-1}b \Rightarrow x = A^T(AA^T)^{-1}b$$

Notice that A^T is overdetermined so let $A^T = QR$

$$\Rightarrow x = QR(R^T Q^T QR)^{-1}b = QR^{-T}b$$

We know the norm is invariant to orthogonal transformations therefore :

$$\|x_{ls}\| = \|R^{-T}b\|$$

b)

In [11]: m = 5

n = 10

A = randn(5,10)

b = randn(5)

At = A'

QR = qrfact(At)

Q = QR[:,Q]

R = QR[:,R]

xls = (*(*(Q,inv(R')),b)

Out[11]: 10-element Array{Float64,1}:

1.01408

0.3956

0.187194

0.598118

-0.370701

```

-0.00259992
-0.391118
0.245214
0.801711
-0.0776825

```

5 Exercise 5

Proof: RLS has a unique solution, then $\text{null}(A) \cap \text{null}(L) = \{0\}$.

$$\begin{aligned}
f(x) &= x^T A^T A x - 2x^T A^T b + b^T b + \lambda x^T L^T L x \\
\nabla f(x) &= 2A^T A x - 2A^T b + 2\lambda L^T L x \\
\Rightarrow A^T A x + \lambda L^T L x &= A^T b \\
\Rightarrow (A^T A + \lambda L^T L)x &= A^T b
\end{aligned}$$

We know that $(A^T A + \lambda L^T L)^{-1}$ must exist since RLS has a unique solution. Since the inverse exists, $\text{null}(A^T A + \lambda L^T L) = \{0\}$.

Note: If $x \in \text{null}(A) \wedge \text{null}(B)$, $Ax = 0 \wedge Ay = 0$

$$\begin{aligned}
\Rightarrow (A + B)x &= Ax + Bx = 0 + 0 = 0 \rightarrow x \in \text{null}(A + B) \\
\Rightarrow \text{null}(A) \wedge \text{null}(B) &\subseteq \text{null}(A + B)
\end{aligned}$$

This means that since $\text{null}(A^T A + \lambda L^T L) = \{0\}$, $\text{null}(A^T A) \wedge \text{null}(\lambda(L^T L)) = \{0\}$

Furthermore, note that if $x \in \text{null}(A^T A) \rightarrow A^T A x = 0$

$$\Rightarrow x^T A^T A x = 0 \rightarrow \|Ax\| = 0 \rightarrow x \in \text{null}(A)$$

Similarly, if $x \in \text{null}(A) \rightarrow Ax = 0 \rightarrow A^T A x = 0 \rightarrow x \in \text{null}(A^T A)$

Combining these we have, $\text{null}(A) = \text{null}(A^T A)$

$$\Rightarrow \text{null}(A) \wedge \text{null}(L) = \text{null}(A^T A) \wedge \text{null}(L^T L) = \{0\}$$

Proof: If $\text{null}(A) \cap \text{null}(L) = \{0\}$ then, RLS has a unique solution

From the previous proof we know to show RLS has a unique solution, we need to show that $A^T A + \lambda L^T L$ is invertible.

It is sufficient to show that $A^T A + \lambda L^T L$ is positive definite (no 0 eigenvalues)

$$A^T A + \lambda L^T L \rightarrow x^T (A^T A + \lambda L^T L)x = x^T A^T A x + \lambda x^T L^T L x = \|Ax\|^2 + \lambda \|Lx\|^2$$

By our assumption there is no x such that both Ax and Lx are 0, therefore $\|Ax\|^2 + \lambda \|Lx\|^2 > 0$

$\Rightarrow A^T A + \lambda L^T L$ is positive definite and invertible

So there exists a unique solution for $(A^T A + \lambda L^T L)x = A^T b$

6 Exercise 6

```

In [14]: x1 = [0,0.5,1,1,0]
          x2 = [0,0,0,1,1]
          function circle_fit(A)
              A = A'
              new_A = 2*A[:,1:1]
              new_b = A[:,1:1].^2
              for i in 2:size(A,2)
                  new_A = hcat(new_A, 2*A[:,i:i])
                  new_b = new_b + A[:,i:i].^2
              end
              new_A = hcat(new_A, ones(size(A,1)))

```

```
    xls = *(inv(*(new_A',new_A)),*(new_A',new_b))
    return xls[1:size(xls,1)-1], xls[size(xls,1)]
end

x, r = circle_fit(vcat(x1',x2'))
```

Out[14]: ([0.5, 0.541667], -0.0833333333333304)