ECE 133A HW 4

Lawrence Liu

November 9, 2022

Exercise T13.3

(a)

With the following julia code we get:

```
using MAT
using LinearAlgebra
using PyPlot
# using Statistics

include("mooreslaw.m")
# println(T)
Years,Transistors=T[:,1],T[:,2]
# println(Years)
# println(Transistors)
A=transpose([reshape(ones(size(Years)),1,:);reshape(Years.-1970,1,:)])
# println(A)
log_Transistors=log10.(Transistors)
theta=A\log_Transistors
println("theta_1=",theta[1])
println("theta_2=",theta[2])
#plot out
plot(Years,log_Transistors,"o")
plot(Years,A*theta)
xlabel("Years")
ylabel("Transistors (log10)")
title("Moore's Law")
legend(["Data","Fit"])
savefig("Moore's Law.png")
close()
```

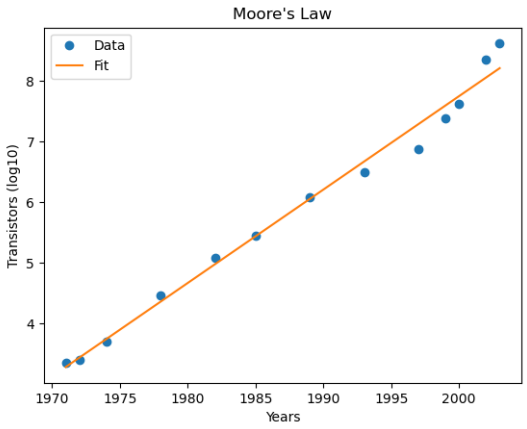
that

$$\theta_1 = 3.125592633829346$$

and

$$\theta_2 = 0.1540181798438225$$

which results in the following fit:



(b)

From our fit we expect the number of transistors to be:

$$10^{\theta_1 + \theta_2(2015-1970)} \approx 10^{10}$$

Which is more than the acutally number of $4 \cdot 10^9$ transistors:

(c)

This is in line with Moore's law since $2\theta_2 = 0.30803635968$ which is close to $\log_{10}(2) = 0.30102999566$

Exercise T12.12

(a)

Exercise A8.12

(a)

$$f(y) = ||Ay - b||^2 + (c^T y - d)^2$$

To minimize we take the derivative of it with respect to y_i for all $1 \leq n \leq N$ and set it to zero have

$$\frac{\partial}{\partial y_i} f(y) = 2(A^T(Ay - b))_i + 2(c^T y - d)c_i = 0$$

Thus we have

$$\nabla f(y) = 2(A^T(Ay - b) + c(c^T y - d)) = 0$$

which gives us

$$\begin{aligned} \nabla f(y) &= 0 \\ 2(A^T(Ay - b) + c(c^T y - d)) &= 0 \\ A^T(Ay - b) + c(c^T y - d) &= 0 \end{aligned}$$

if $\hat{y}$ is a solution then we must have that

$$A^T(A\hat{y} - b) + c(c^T \hat{y} - d) = 0$$

we can confirm this, since

$$\hat{y} = \hat{x} + \frac{d - c^T \hat{x}}{1 + c^T (A^T A)^{-1} c} (A^T A)^{-1} c$$

we have:

$$\begin{aligned}
A^T(A\hat{y}-b)+c(c^T\hat{y}-d) &= 0\\
A^TA\frac{d-c^T\hat{x}}{1+c^T(A^TA)^{-1}c}(A^TA)^{-1}c+cc^T\hat{x}+c(c^T\frac{d-c^T\hat{x}}{1+c^T(A^TA)^{-1}c}(A^TA)^{-1}c-d) &= 0\\
dc-c^T\hat{x}c+cc^T(d-c^T\hat{x})(A^TA)^{-1}c-c(d-c^T\hat{x})(1+c^T(A^TA)^{-1}c) &= 0\\
cc^T(d-c^T\hat{x})(A^TA)^{-1}c-c(d-c^T\hat{x})(c^T(A^TA)^{-1}c) &= 0\\
cc^T(d-c^T\hat{x})(A^TA)^{-1}c-cc^T(d-c^T\hat{x})(A^TA)^{-1}c &= 0
\end{aligned}$$

(b)

We first compute the QR factorization fo A, which will cost us $2mn^2$ flops, then we can compute $\hat{x}$ with an additonal $2mn+n^2$ flops. Likewise, since we can rewrite $(A^TA)^{-1}c$ as $(R^TQ^TQR)^{-1}c=(R^TR)^{-1}c$, which we can solve in $2n^2$ flops. then computing $c^T\hat{x}$ and $c^T(A^TA)^{-1}c$ will each cost us an additional $2n-1$ flops, then computing $\frac{d-c^T\hat{x}}{1+c^T(A^TA)^{-1}c}$ will cost us 3 flops.Then computing $\hat{x}+\frac{d-c^T\hat{x}}{1+c^T(A^TA)^{-1}c}(A^TA)^{-1}c$ will cost us $2n$ flops, so in total this algorithm will cost us $\boxed{2mn^2+2mn+3n^2+6n-1}$ flops.