COT 5615 Math for Intelligent Systems Fall 2021 Homework #5

UFID: 96703101
Name: *Vyom Pathak*
Instructor: Professor Kejun Huang
Due Date: November 01, 2021

Problem A12.1

Solving least squares problems in Julia

Solution

a The following code shows the comparision for different methods to find the least square solution:

```
using LinearAlgebra
A = rand(20,10);
b = rand(20);
#a.1
x_cap_1 = A\b;
println(x_cap_1,"\n");
#a.2
x_cap_2 = inv(A'A)A'b;
println(x_cap_2,"\n");
#a.3
x_cap_3 = pinv(A)b;
println(x_cap_3,"\n");

println("Checking if the vectors are equal or not by printing it. One can see
that the vectors are similar upto 13 to 14 decimal places. Thus, they are equivalent.");
```

b The following Julia code checks the inequality for the least squared problem:

```
function verify(A, b, x_cap, delta)
    return norm(A*(x_cap+delta)-b).^2 > norm(A*x_cap-b).^2
end
delta1 = rand(10);
println(verify(A, b, x_cap, delta1));
delta2 = rand(10);
println(verify(A, b, x_cap, delta2));
delta3 = rand(10);
println(verify(A, b, x_cap, delta3));
delta4 = rand(10);
scaled_delta4 = (delta4.-minimum(delta4))./(maximum(delta4)-minimum(delta4));
println(verify(A, b, x_cap, scaled_delta4));
```

Problem A12.2

Julia timing test for least squares

Solution

The following code finds the time for calculating the least squared solution:

```
A = rand(100000,100);
b = rand(100000);
@time x_cap = A\b;
@time x_cap = A\b;
@time x_cap = A\b;
@time x_cap = A\b;
@time x_cap = A\b;
@time x_cap = A\b;
@time x_cap = A\b;
#=
2.766416 seconds (2.49 M allocations: 216.166 MiB, 1.83% gc time, 17.91% compilation time)
2.273690 seconds (633 allocations: 77.364 MiB, 1.37% gc time)
2.187377 seconds (633 allocations: 77.364 MiB, 0.11% gc time)
2.186633 seconds (633 allocations: 77.364 MiB, 0.07% gc time)
2.208185 seconds (633 allocations: 77.364 MiB, 0.07% gc time)
2.199118 seconds (633 allocations: 77.364 MiB, 0.04% gc time)
2.195885 seconds (633 allocations: 77.364 MiB, 0.03% gc time)
=#
```

Here, we can see that on average 2.1 seconds are taken to solve the least squared solution. [M1 Macbook Air]

Problem A12.4

Transit system tomography

Solution

Here the c vector corresponds to the total trip time for each link which can be shown as below:

$$c = (f_1 - s_1, f_2 - s_2, \ldots, f_n - s_n) \; where \; n \; is \; the \; number \; of \; links$$

For the vector R, it can be denoted as follows:

$$R = \begin{cases} 1, & \text{if passenger i with link j where i is the row from 1 to m, j is the column from 1 to n.} \\ 0, & \text{otherwise.} \end{cases}$$

The dimension of c vector is thus $nX1$ where n is the number of links, while the dimension of R vector is mXn where m is the number of passengers.

Problem A12.7

Constructing a portfolio of bonds to approximate a sequence of liabilities

Solution

From the equation $\|l - p\|$, we get the equation as follows:

$$\|l - p\|^2 = \|p - l\|^2$$
$$\|l - p\|^2 = \|C * q - l\|^2 \;\; [\because \; p = q_1 c_1 + q_2 c_2 + \ldots + q_{20} c_{20}]$$

Thus, for the original equation $\|Aq - b\|^2$, $A = C$, $q = q$, and $b = l$. Thus, the dimension of A is $120X20$ [As it is given that there are 20 c values where each c value is having 120-vectors], and that of b is $120X1$.

Problem A13.3

Auto-regressive time series prediction

Solution

a The matrix A, and vector b can be shown as follows:

$$A = (x_M, x_{M+1}, x_{M+2}, \ldots, x_{N-1}; x_{M-1}, x_M, x_{M+1}, , \ldots, x_{N-2};$$
$$x_{M-2}, x_{M-1}, x_M, \ldots, x_{N-3}; \ldots, \ldots, \ldots; x_1, x_2, x_3, \ldots, x_{N-M})$$

$$b = (x_{M+1}, x_{M+2}, \ldots, x_N)$$

The dimension of the matrix A and vector b are $((N-M)XM)$ and $(N-MX1)$ respectively.

b The below Julia code shows all the train and test data Loss values and also how find the minimum value of J [0.01891584139384952] and the corresponding value of M [12]. It also shows the calculations for the sample predictors 1&2 [Sample 1 J: 2.6318895838567373 Sample 2 J: 13.116905507420244].

Train Losses:

[0.030494624025126198, 0.030292696561256324, 0.02449655127211178, 0.02439490352714129, 0.022996314298873437, 0.019842912056501263, 0.019137191183089056, 0.019081546282888485, 0.01899948835215664, 0.01920081297889598, 0.01891584139384952]

Test Losses:

[3.384808634417929, 3.380245845279127, 3.367326758735222, 3.3617479695771517, 3.3564118647685057, 3.293567803934302, 3.3375738507437154, 3.319862796736469, 3.3710871022554083, 3.3827366034025976, 3.4236529933573667]

```julia
include("time_series_data.jl")
J_train = zeros(11);
J_test = zeros(11);
for M in 2:12
    N = size(x_train)[1];
    A = toeplitz(x_train,M)[M:N-1,1:M];
    b = x_train[M+1:N];
    beta = A\b;
    J_train[M-1] = (norm(A*beta-b).^2)./(N-M);

    N_test = size(x_test)[1];
    A = toeplitz(x_test,M)[M:N_test-1,1:M];
    J_test[M-1] = (norm(A*beta-b).^2)./(N_test-M);
end
println("Train Losses");
println(J_train);
println("\nTest Losses: ");
println(J_test);
mJ = minimum(J_train);
print("The mimimum value of J [", mJ,"] is at M = ", findall(x->x==mJ,J_train)[1]+1);

# 2 Sample predictors
M = 2;
N = size(x_train)[1];
A_sam1 = rand(N-M);
A_sam1 .= x_train[M:N-1]; #Previous value
```

```
    J_sam1 = (norm(A_sam1-x_train[M+1:N]).^2)./(N-M);
    println("\nSample 1: ", J_sam1);
    A_sam2 = rand(N-M); #Extrapolating line
    for i in M:N-2
        A_sam2[i] = x_train[i]+(x_train[i]-x_train[i-1]);
    end
    J_sam2 = (norm(A_sam2-x_train[M+1:N]).^2)./(N-M);
    println("\nSample 2: ", J_sam2);
```

Problem A13.5

Nonlinear auto-regressive model

Solution

The matrix A and vector b can be expressed as follows:

$$
\begin{aligned}
A &= (z_2, z_3, \ldots, z_{T-1}; z_1, z_2, \ldots, z_{T-2}; z_2 \cdot z_1, z_3 \cdot z_2, \ldots, z_{T-1} z_{T-2}) \\
b &= (z_2, z_3, \ldots, z_{T-1}) \\
\theta &= (\theta_1, \theta_2, \theta_3)
\end{aligned}
$$