# Joint Assignment 02 Report

Anton Kudryavtsev CS-04 a.kudryavtsev@innopolis.university

April 25, 2023

## Least Square Approximation

**Equation** Least Square Approximation is done using left pseudoinverse of matrix A;

$$A * x = b; \tag{1}$$

$$A^T A * x = A^T b; \tag{2}$$

$$x \approx (A^T A)^{-1} A^T b; \tag{3}$$

**Code.** The code to find $x$ is the following:

```
int main() {
    // Specify Output Format
    std::cout.setf(std::ios::fixed, std::ios::floatfield);
    std::cout.precision(4);

    // Read Samples
    size_t m;
    std::cin >> m;
    Matrix<double> *ts = new ColumnVector<double>(m);
    Matrix<double> *b = new ColumnVector<double>(m);
    for (int i = 0; i < m; ++i) {
        double ti, bi;
        std::cin >> ti >> bi;
        ts->Put(i, 0, ti);
        b->Put(i, 0, bi);
    }
    // Read Polynomial Degree
    size_t n;
    std::cin >> n;

    // Generate Matrix A
    auto *A = new Matrix<double>(m, n+1, [ts](auto i, auto j) {
        return pow(ts->Get(i, 0), j);
    });

    // Find Model
    auto At = A->Transpose();
    auto AtA = *At * A;
    auto AtAi = AtA->Inverse();
    auto Atb = *At * b;
```

```
    auto x = *AtAi * Atb;

    // Report Steps
    std::cout << "A:\n" << A;
    std::cout << "A_T*A:\n" << AtA;
    std::cout << "(A_T*A)^-1:\n" << AtAi;
    std::cout << "A_T*b:\n" << Atb;
    std::cout << "x~:\n" << x;

    // Free Memory
    free({ts, A, b, At, AtA, AtAi, Atb, x});
    return 0;
}
```

**Proof.** Let us introduce function $4x^2 - 8x + 3$ and generate samples for it with noise. Python code:

```
with open("plot.dat", "w") as f:
    for i in range(1000):
        print(i, (4*i**2 -8*i +3)+(random()-0.5)*10, file=f)
```

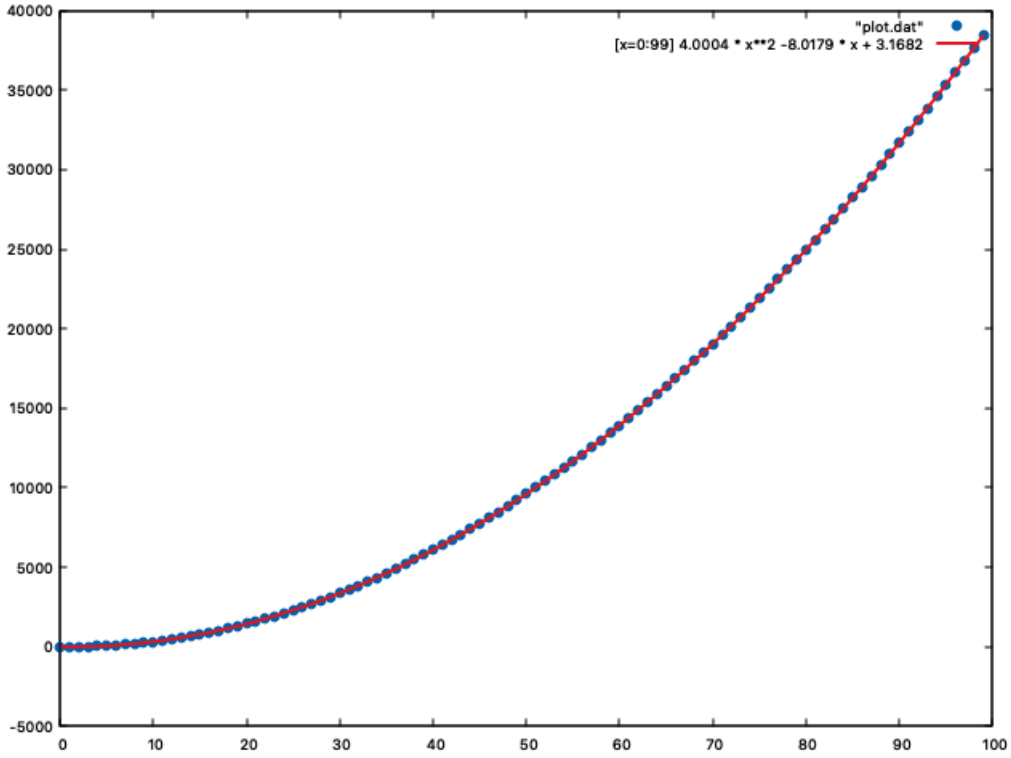
Then, forward data from "plot.dat" file to the c++ program and get x. We get following results:

```
(A_T*A)^-1:
0.0865 -0.0035 0.0000
-0.0035 0.0002 0.0000
0.0000 0.0000 0.0000
A_T*b:
1274149.9855
95402091.6468
7606638251.9895
x~:
3.1682
-8.0179
4.0004
```

Plot data using *gnuplot*:

```
> set style line 2 linecolor rgb '#dd181f' linetype 1 linewidth 2
> plot "plot.dat" with lines linestyle 1, [x=0:99] 4.0004 * x**2
    -8.0179 * x + 3.1682 with lines linestyle 2
```

We get following picture:


"plot.dat"
[x=0:99] 4.0004 * x**2 -8.0179 * x + 3.1682
40000
35000
30000
25000
20000
15000
10000
5000
0
-5000
0
10
20
30
40
50
60
70
80
90
100