# Analytical Geometry and Linear Algebra II — Least Squares Approximation

Dmitriy Okoneshnikov
B22-DSAI-04
d.okoneshnikov@innopolis.university

April 23, 2023

## 1 Source code

Showing only `main` function. The whole program can be found here.

```
int main(void)
{
    cout << fixed << setprecision(4);

    int m;
    cin >> m;
    double t[m];
    ColumnVector *b = new ColumnVector(m);
    for (int i = 0; i < m; i++)
    {
        double t_i, b_i;
        cin >> t_i >> b_i;
        t[i] = t_i;
        (*b)[i][0] = b_i;
    }
    int n;
    cin >> n;

    Matrix *A = new Matrix(m, n + 1);
    for (int i = 0; i <= n; i++)
        for (int j = 0; j < m; j++)
            (*A)[j][i] = pow(t[j], (double)i);

    cout << "A:\n";
    cout << *A;

    Matrix *A_T = A->T();
    SquareMatrix *A_1 = (SquareMatrix *)((*A_T) * (*A));

    cout << "A_T*A:\n";
    cout << *A_1;

    SquareMatrix *A_2 = A_1->inverse(false);
```

```
    cout << "(A_T*A)^-1:\n";
    cout << *A_2;

    ColumnVector *A_3 = (ColumnVector *)((*A_T) * (*(Matrix *)b));

    cout << "A_T*b:\n";
    cout << *A_3;

    ColumnVector *A_4 = (ColumnVector *)((*(Matrix *)A_2) * (*(Matrix *)A_3));

    cout << "x~:\n";
    cout << *A_4;

#if (defined(WIN32) || defined(_WIN32)) && USE_GNUPLOT
    FILE *pipe = _popen(GNUPLOT_NAME, "w");
#elif USE_GNUPLOT
    FILE *pipe = popen(GNUPLOT_NAME, "w");
#endif
#if USE_GNUPLOT
    fprintf(pipe, "%s\n", "set terminal png");
    fprintf(pipe, "%s\n", "set output 'output.png'");
    fprintf(pipe, "%s\n", "set title \"Least Squares Approximation\"");
    fprintf(pipe, "%s\n", "set key noautotitle");
    fprintf(pipe, "%s\n", "set autoscale xy");
    fprintf(pipe, "%s\n", "set offsets 0.05, 0.05, 0.05, 0.05");
    string func;
    for (int i = 0; i <= n; i++)
    {
        if ((*A_4)[i][0] < 0 and i != 0)
            func = func.substr(0, func.size() - 1);
        func += to_string((*A_4)[i][0]);
        func += '*';
        func += "x**";
        func += to_string(i);
        if (i != n)
            func += '+';
    }
    cout << func << endl;
    fprintf(pipe, "plot %s lw 3, '-' w p pt 7 ps 2\n", func.c_str());
    for (int i = 0; i < m; i++)
        fprintf(pipe, "%lf %lf\n", t[i], (*b)[i][0]);
    fprintf(pipe, "%s\n", "e");
    fflush(pipe);
#endif
#if (defined(WIN32) || defined(_WIN32)) && USE_GNUPLOT
    _pclose(pipe);
#elif USE_GNUPLOT
    pclose(pipe);
#endif

    return 0;
```

```
}
```

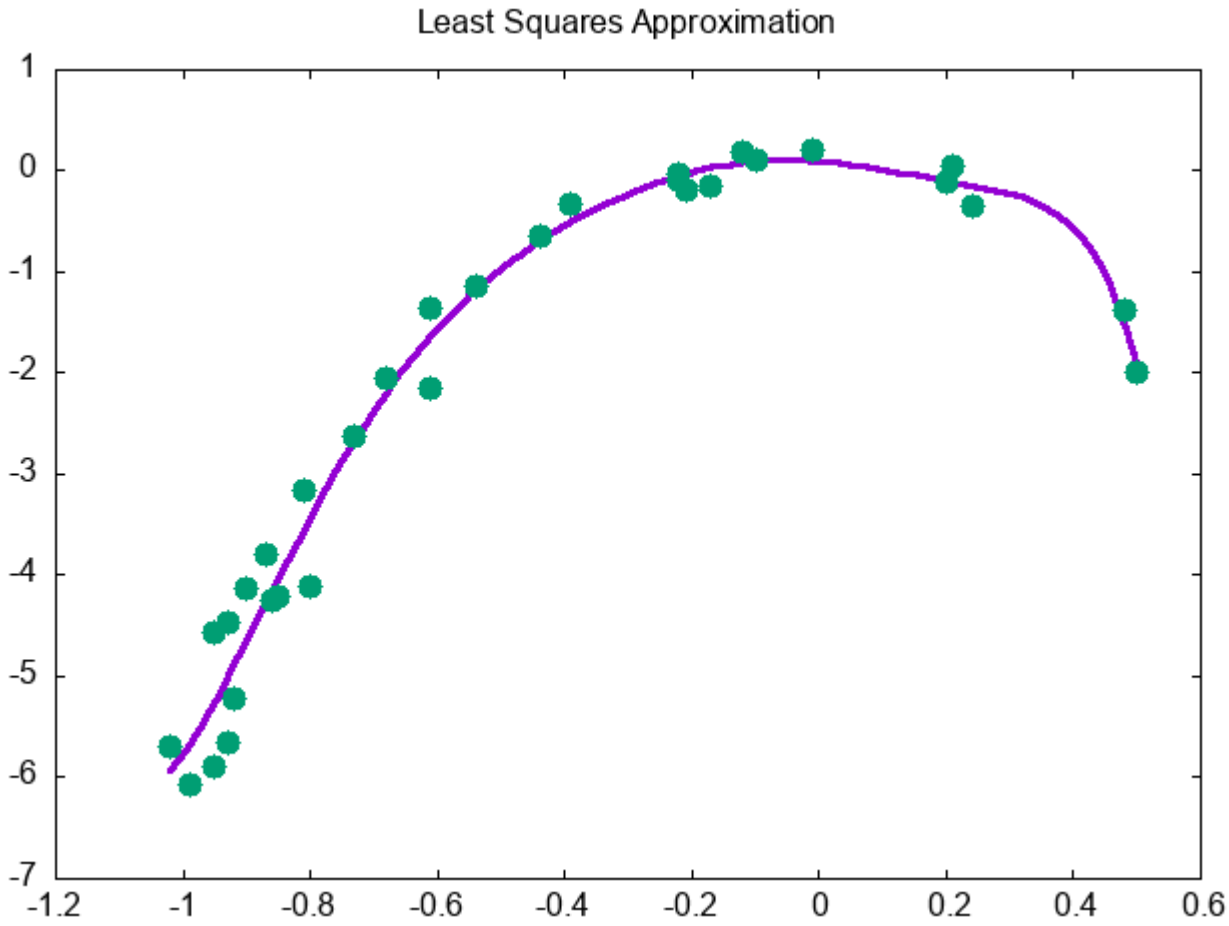
## 2 Points

Generated by using this Python script (written by me).

| x | y |
|---|---|
| -0.61 | -2.15 |
| -0.93 | -4.47 |
| -0.17 | -0.14 |
| -0.95 | -4.56 |
| -0.87 | -3.8 |
| -0.54 | -1.13 |
| -0.22 | -0.08 |
| -0.92 | -5.21 |
| -0.12 | 0.19 |
| -0.1 | 0.1 |
| 0.24 | -0.34 |
| -0.93 | -5.66 |
| 0.2 | -0.1 |
| -0.86 | -4.25 |
| -0.81 | -3.16 |
| -0.9 | -4.13 |
| -0.44 | -0.64 |
| -0.01 | 0.2 |
| 0.21 | 0.04 |
| -1.02 | -5.69 |
| 0.5 | -2.0 |
| 0.48 | -1.38 |
| -0.68 | -2.05 |
| -0.39 | -0.32 |
| -0.73 | -2.63 |
| -0.21 | -0.19 |
| -0.85 | -4.2 |
| -0.61 | -1.36 |
| -0.99 | -6.07 |
| -0.22 | -0.03 |
| -0.8 | -4.1 |
| -0.95 | -5.9 |

## 3 Plot

The program was asked to compose a least square approximation with the degree of the polynomial equal to 2. The polynomial calculated by the program: $0.094434x^0 - 0.486457x^1 - 4.264471x^2 + 7.714597x^3 + 15.651272x^4 - 17.897050x^5 - 87.106170x^6 - 80.904751x^7 - 21.714462x^8$


Least Squares Approximation
1
0
-1
-2
-3
-4
-5
-6
-7
-1.2
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6