Finding the rank of a matrix

Table of Contents

The rank of a matrix is the largest number of linearly independent rows/columns of the matrix. The rank is not only defined for square matrices.

The rank of a matrix can also be defined as the largest order of any non-zero minor in the matrix.

Let the matrix be rectangular and have size $N \times M$. Note that if the matrix is square and its determinant is non-zero, then the rank is N ($= M$); otherwise it will be less. Generally, the rank of a matrix does not exceed $\min(N, M)$.

Algorithm

You can search for the rank using Gaussian elimination. We will perform the same operations as when solving the system or finding its determinant. But if at any step in

the i-th column there are no rows with an non-empty entry among those that we didn't selected already, then we skip this step and decrease the rank by one (initially the rank is set equal to $\max(N, M)$). Otherwise, if we have found a row with a non-zero element in the i-th column during the i-th step, then we mark this row as a selected one and perform the usual operations of taking this row away from the rest.

Complexity

This algorithm runs in $\mathcal{O}(n^3)$.

Implementation

```
const double EPS = 1E-9;

int compute_rank(vector<vector<int>> A) {
    int n = A.size();
    int m = A[0].size();

    int rank = max(n, m);
    vector<bool> row_selected(n, false);
    for (int i = 0; i < m; ++i) {
        int j;
        for (j = 0; j < n; ++j) {
            if (!row_selected[j] && abs(A[j][i
                break;
        }

        if (j == n) {
```

```
            --rank;
        } else {
            row_selected[j] = true;
            for (int p = i + 1; p < m; ++p)
                A[j][p] /= A[j][i];
            for (int k = 0; k < n; ++k) {
                if (k != j && abs(A[k][i]) > E
                    for (int p = i + 1; p < m;
                        A[k][p] -= A[j][p] * A
                }
            }
        }
    }
    return rank;
}
```