# Решение СЛУ с циркулянтной матрицей

**Былинкин Дмитрий, Б05-005**

## Теоретическая мотивация

Циркулянтная матрица - это матрица вида

$$A = [a_{kl}], \text{ где } a_{k^1 l^1} = a_{k^2 l^2} \text{ при } k^1 - l^1 = k^2 - l^2 (modN), \ N \text{ - порядок матрицы}$$

СЛУ с циркулянтной матрицей возникают при дискретизации интегрального уравнения на окружности, если его ядро $K(x, y)$ зависит лишь от расстояния между точками. Более того, на циркулянтах основаны методы построения предобусловливателей для более общего случая систем с теплицевой матрицей.
Рассматриваемый тип уравнения интересен еще и тем, что для него есть возможность строить быстрые и супербыстрые алгоритмы решения.

## Небходимый теоретический минимум

Заранее оговорим, что интерес представляет только случай невырожденных матриц. Рассмотрим уравнение вида:

$$Cx = f \tag{1}$$

Очевидно, что его можно заменить эквивалентной системой с матрицей преобразования Фурье:

$$F^* C F y = F^* f \tag{2}$$

$$x = Fy \tag{3}$$

**Теорема 1:** Пусть $c = (c_0, ..., c_{N-1})^T$ - первый столбец циркулянта C. Тогда $\lambda_m = \phi(\omega_m) = \Sigma_{k=0}^{N-1} c_k \omega_m^k$ и $F^* C F = diag(\lambda_0, ..., \lambda_{N-1})$, где $\omega$ - комплексный корень степени $N$ из единицы
Другими словами, нет необходимости перемножать матрицы в левой части (2), поскольку $F^* C F$ имеет диагональный вид, откуда легко следует выражение для $k$-й координаты $y$:

$$y_k = \frac{[F^* f]_k}{\phi(\omega_k)}, k = \overline{0, N-1} \tag{4}$$

Далее, аналогичным образом вычисляя значение еще одного многочлена, получим координаты вектора $x$.
Для вычисления (4) и координат $x$ помимо некоторого количества операций умножения требуется только $N$ операций деления. Таким образом, реализуя алгоритм БПФ для вычисления значения многочлена, получим алгоритм решения СЛУ (1) за $O(NlogN)$.

## Практическая реализация

Алгоритм решения реализован на языке **C++** и рассматривает только случай матриц порядка степени двойки - в остальных случаях СЛУ (1) решается без использования БПФ за квадратичное время.
В тестах генерировалась случайная циркулянтная матрица заранее заданного размера, а также вектор свободных членов. Выход программы - вектор решений и время работы. Сгенерированные значения также выводились в терминал для проверки корректности работы.

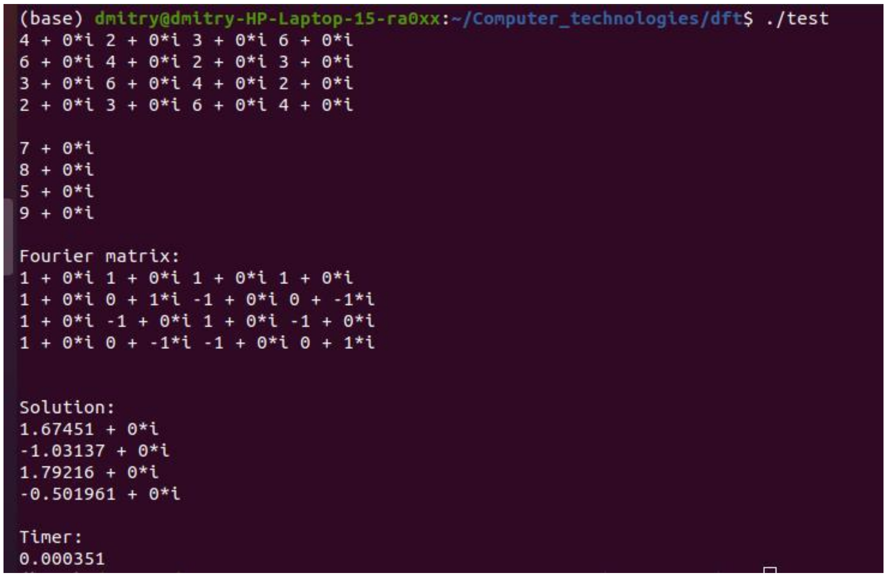


Figure 1: Пример работы программы

На любой совместной системе с циркулянтной матрицей программа дает правильный (с точностью до округления) ответ. На входе из примера аналитически методом Гаусса был получен ответ:

$$x_0 = 1.675; x_1 = -1.031; x_2 = 1.792; x_3 = -0.502$$

Тесты проводились на размерах матриц вплоть до $256 \times 256$, при этом было получено неплохое время работы:

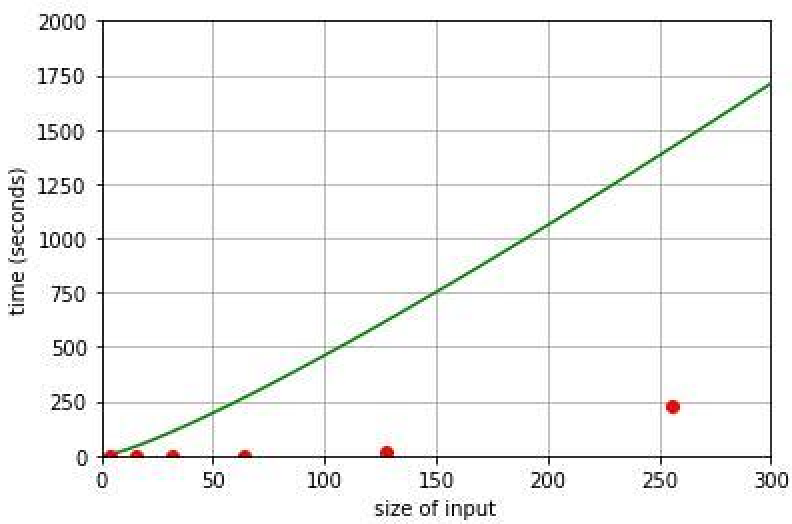


Figure 2: Зависимость времени работы программы от размера входа

## Дальнейшая работа

Время работы программы может быть существенно улучшено, если для хранения матриц вместо контейнеров STL использовать динамические массивы. Также в программу можно включить построение оптимальных обусловливателей для теплицевых матриц, что позволит использовать ее для решения более широкого класса задач.