



ФАКУЛЬТЕТ КОМПЬЮТЕРНЫХ НАУК И ТЕЛЕКОММУНИКАЦИЙ

Лабораторная работа №3
По дисциплине
«Численные методы и прикладное программирование»

Тема:
«Методы решения нелинейного уравнения»

Работу выполнили:

*Дзенис Ричард
Кобелев Денис
Якушин Владислав*

Рига
2017 г.

Содержание

1	Формулировка задания	2
2	Поиск корней методом бисекции	3
2.1	Результаты работы метода	4
3	Поиск корней методом парабол	5
3.1	Результаты работы метода	7
4	Выводы	7

1 Формулировка задания

В данной лабораторной работе требуется реализовать два алгоритма решения нелинейного уравнения: метод бисекции и индивидуальный метод. Предусмотрено два способа реализации алгоритмов: с использованием скользящего окна или же с локально заданным интервалом. В зависимости от выбора реализации, будут меняться входные параметры метода. Во всех обоих случаях в качестве входного параметра требуется установить ввод точности решения (эпсилон). В таблице ниже представлены необходимые параметры для запуска алгоритмов:

Использование скользящего окна	Установка локальной области поиска
– Начальная точка всей области поиска	– Начальная точка локального окна
– Конечная точка всей области поиска	– Конечная точка локального окна
– Ширина скользящего окна	

Так же, для некоторых задач будет необходимо предусмотреть смещение функции по оси Y , поэтому для создания более универсальной реализации, требуется ввести и этот параметр.

2 Поиск корней методом бисекции

В данной лабораторной работе была разработана модификация метода половинного деления, которая способна находить абсолютно все корни на заданном интервале (left;right). Это достигается тем, что после нахождения очередного корня, в список интервалов для проверки добавляется два следующих интервала:

- (oleft;middle - ε), и
- (middle + ε ;oright).

где: oleft — исходная левая граница интервала;
oright — исходная правая граница интервала;
middle — найденный корень.

```
1 #include "common.h"
2
3 roots find_roots(
4     std::function<double(double)> f,
5     double left,
6     double right,
7     const double epsilon
8 )
9 {
10     roots results;
11     std::vector<std::pair<double, double>> intervals;
12
13     intervals.emplace_back(std::make_pair(left, right));
14
15     while (!intervals.empty()) {
16         auto [oleft, oright] = intervals.back();
17         intervals.pop_back();
18
19         left = oleft;
20         right = oright;
21
22         size_t iterations = 0;
23         double middle;
24         do {
25             middle = left / 2 + right / 2;
26             if (f(left) * f(middle) <= 0)
27                 right = middle;
28             else
29                 left = middle;
30             iterations++;
31         } while (right - left > epsilon);
32
33         if (middle - oleft > epsilon
34             && oright - middle > epsilon) {
35             intervals.emplace_back(std::make_pair(oleft, middle - epsilon));
36             intervals.emplace_back(std::make_pair(middle + epsilon, oright));
37         } else {
38             continue;
39         }
40
41         if (can_insert(results, middle, epsilon))
42             results.emplace(middle, iterations);
43     }
44
45     return results;
46 }
47
48 #include "main.h"
```

Основная логика метода бисекции располагается на строчках 24-31.

Также можно заметить, что отсутствует начальная проверка на наличие корней в интервале, это делается из расчёта на то, что количество корней может быть чётное на заданном интервале.

Наконец, в случае если метод половинного деления «ушёл» в исходную левую или правую границу, то найденный корень считается «ложным» и отбрасывается.

2.1 Результаты работы метода

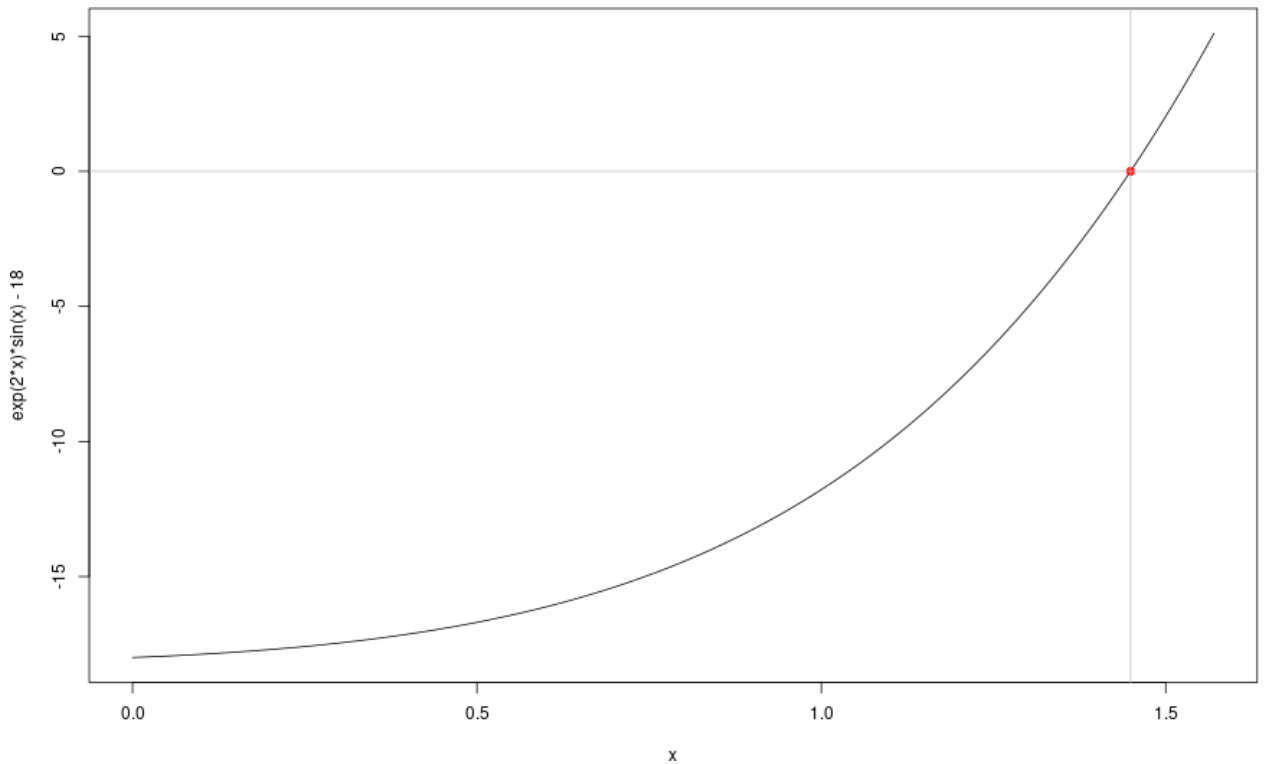


Рис. 1: График функции $e^{2x} \cdot \sin(x) - 18$ на интервале $[0; \pi/2]$.

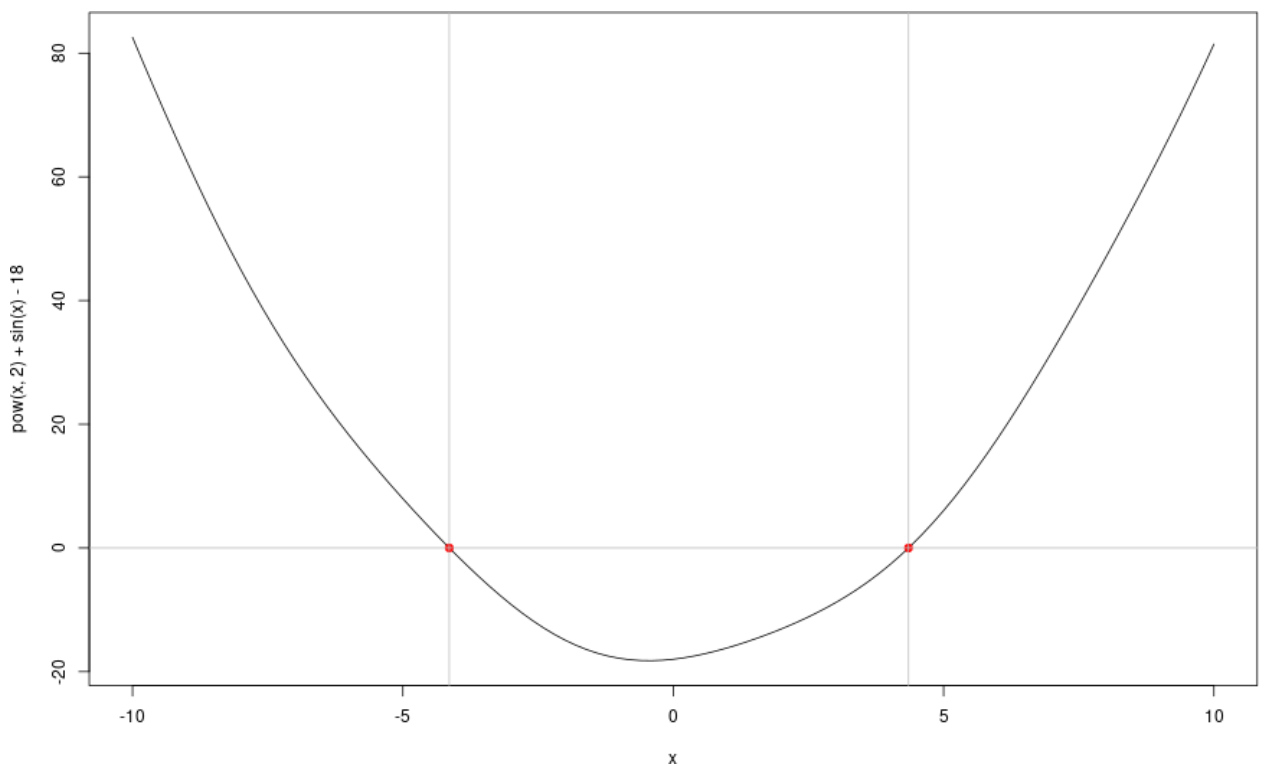


Рис. 2: График функции $x^2 + \sin(x) - 18$ на интервале $[-10; +10]$.

Функция	Точность	Найденные корни	Число итераций	k
$e^{2x} \cdot \sin(x) - 18$	10^{-2}	1.45348	8	7.295352
	10^{-4}	1.44897	14	13.93921
	10^{-6}	1.44891	21	20.58306
	10^{-9}	1.44891	31	30.54885
$x^2 + \sin(x) - 18$	10^{-2}	-4.15039	11	10.96578
		4.34522	11	
	10^{-4}	-4.14223	18	17.60964
		4.35149	18	
	10^{-6}	-4.14224	25	24.2535
		4.3515	24	
	10^{-9}	-4.14224	35	34.21928
		4.3515	34	

Во всех случаях число итераций совпало с теоретическим числом итераций, рассчитанным по формуле:

$$k = \log_2 \frac{|b - a|}{\varepsilon} \quad (1)$$

3 Поиск корней методом парабол

К методу параболу была применена та-же модификация, что и к методу бисекции для нахождения, по возможности, не только одного корня. Основная логика метода располагается на строчках 27-82, основная часть – вычисление коэффициентов квадратного уравнения и его решения. Коэффициенты квадратного уравнения рассчитываются с помощью метода описанного в приложении (версия 2).

В силу того, что в данный метод для поиска корней могут попасть интервалы, на которых нет корней, оба корня квадратного уравнения проходят проверку на принадлежность интервала. В случае если ни один из корней не принадлежит интервалу, то поиск корней на данном интервале сразу-же прекращается.

```

1  #include "common.h"
2
3  roots find_roots(
4      std::function<double(double)> f,
5      double left,
6      double right,
7      const double epsilon
8  )
9  {
10     roots results;
11     std::vector<std::pair<double, double>> intervals;
12
13     intervals.emplace_back(std::make_pair(left, right));
14
15     while (!intervals.empty()) {
16         auto [oleft, oright] = intervals.back();
17         intervals.pop_back();
18
19         left = oleft;
20         right = oright;
21
22         double prev_middle;
23         bool has_root = true;
24         size_t iterations = 0;
25         double middle = left / 2 + right / 2;
26         do {
27             double x_0 = left;
28             double x_1 = middle;
29             double x_2 = right;
30
31             double y_0 = f(x_0);

```

```

32     double y_1 = f(x_1);
33     double y_2 = f(x_2);
34
35     double denom_0 = (x_0 - x_1) * (x_0 - x_2);
36     double a_0 = y_0 / denom_0;
37     double b_0 = -y_0 * (x_1 + x_2) / denom_0;
38     double c_0 = y_0 * x_1 * x_2 / denom_0;
39
40     double denom_1 = (x_1 - x_0) * (x_1 - x_2);
41     double a_1 = y_1 / denom_1;
42     double b_1 = -y_1 * (x_0 + x_2) / denom_1;
43     double c_1 = y_1 * x_0 * x_2 / denom_1;
44
45     double denom_2 = (x_2 - x_0) * (x_2 - x_1);
46     double a_2 = y_2 / denom_2;
47     double b_2 = -y_2 * (x_0 + x_1) / denom_2;
48     double c_2 = y_2 * x_0 * x_1 / denom_2;
49
50     double A = a_0 + a_1 + a_2;
51     double B = b_0 + b_1 + b_2;
52     double C = c_0 + c_1 + c_2;
53
54     double D = sqrt(B * B - 4 * A * C);
55
56     double part1_x = -B / (2 * A);
57     double part2_x = D / (2 * A);
58
59     // Choose X, which belongs [left; right]
60     double X1 = part1_x + part2_x;
61     double X2 = part1_x - part2_x;
62     double X;
63     if (left <= X1 && X1 <= right) {
64         X = X1;
65     } else if (left <= X2 && X2 <= right) {
66         X = X2;
67     } else {
68         // No root can be found!
69         has_root = false;
70         break;
71     }
72
73     // Choose what to replace (left, or right?)
74     if (middle <= X && X <= right) {
75         left = middle;
76     } else {
77         right = middle;
78     }
79     prev_middle = middle;
80     middle = X;
81
82     iterations++;
83 } while (fabs(middle - prev_middle) > epsilon);
84
85 if (has_root
86     && middle - oleft > epsilon
87     && oright - middle > epsilon)
88 {
89     intervals.emplace_back(std::make_pair(oleft, middle - epsilon));
90     intervals.emplace_back(std::make_pair(middle + epsilon, oright));
91 } else {
92     continue;

```

```

93         }
94
95         if (can_insert(results, middle, epsilon))
96             results.emplace(middle, iterations);
97     }
98
99     return results;
100 }
101
102 #include "main.h"

```

3.1 Результаты работы метода

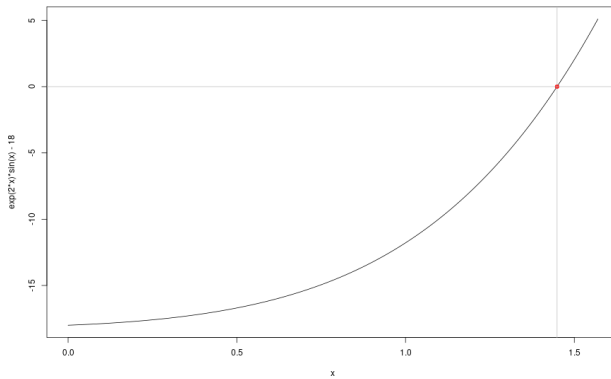


Рис. 3: График функции $e^{2x} \cdot \sin(x) - 18$ на интервале $[0; \pi/2]$.

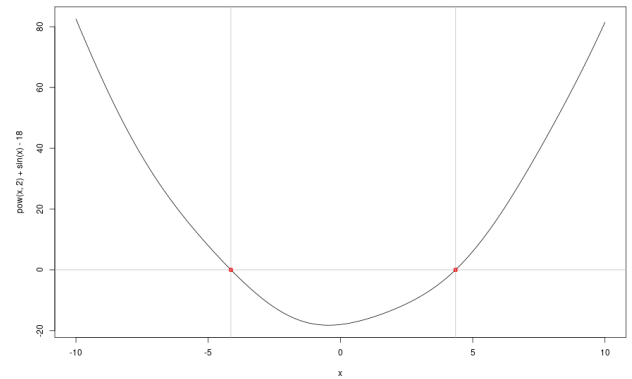


Рис. 4: График функции $x^2 + \sin(x) - 18$ на интервале $[-10; +10]$.

Как можно заметить, графики функций с найденными корнями, как и следовало ожидать, совпадают с теми, которые были получены в методе половинного деления.

Функция	Точность	Найденные корни	Число итераций
$e^{2x} \cdot \sin(x) - 18$	10^{-2}	1.44891	3
	10^{-4}	1.44891	4
	10^{-6}	1.44891	4
	10^{-9}	1.44891	5
$x^2 + \sin(x) - 18$	10^{-2}	-4.14224	3
		4.35151	3
	10^{-4}	-4.14224	4
		4.3515	4
	10^{-6}	-4.14224	5
		4.3515	5
	10^{-9}	-4.14224	5
		4.3515	6

4 Выводы

Приложение

Вывод коэффициентов для метода парабол

$$L_2(x) = y_0 l_0(x) + y_1 l_1(x) + y_2 l_2(x) = 0$$

$$\begin{aligned} & y_0 \cdot \frac{(x - x_1)(x - x_2)}{(x_0 - x_1)(x_0 - x_2)} \\ & + y_1 \cdot \frac{(x - x_0)(x - x_2)}{(x_1 - x_0)(x_1 - x_2)} \\ & + y_2 \cdot \frac{(x - x_0)(x - x_1)}{(x_2 - x_0)(x_2 - x_1)} \\ & = 0 \end{aligned}$$

$$\begin{aligned} & y_0 \cdot \frac{(x - x_1)(x - x_2)}{(x_0 - x_1)(x_0 - x_2)} \cdot \frac{(x_1 - x_0)(x_1 - x_2)(x_2 - x_0)(x_2 - x_1)}{(x_1 - x_0)(x_1 - x_2)(x_2 - x_0)(x_2 - x_1)} \\ & + y_1 \cdot \frac{(x - x_0)(x - x_2)}{(x_1 - x_0)(x_1 - x_2)} \cdot \frac{(x_0 - x_1)(x_0 - x_2)(x_2 - x_0)(x_2 - x_1)}{(x_0 - x_1)(x_0 - x_2)(x_2 - x_0)(x_2 - x_1)} \\ & + y_2 \cdot \frac{(x - x_0)(x - x_1)}{(x_2 - x_0)(x_2 - x_1)} \cdot \frac{(x_0 - x_1)(x_0 - x_2)(x_1 - x_0)(x_1 - x_2)}{(x_0 - x_1)(x_0 - x_2)(x_1 - x_0)(x_1 - x_2)} \\ & = 0 \end{aligned}$$

$$\begin{aligned} & y_0 \cdot (x - x_1)(x - x_2) \cdot (x_1 - x_0)(x_1 - x_2)(x_2 - x_0)(x_2 - x_1) \\ & + y_1 \cdot (x - x_0)(x - x_2) \cdot (x_0 - x_1)(x_0 - x_2)(x_2 - x_0)(x_2 - x_1) \\ & + y_2 \cdot (x - x_0)(x - x_1) \cdot (x_0 - x_1)(x_0 - x_2)(x_1 - x_0)(x_1 - x_2) \\ & = 0 \end{aligned}$$

$$\begin{aligned} & -y_0 \cdot (x - x_1)(x - x_2) \cdot (x_1 - x_0)(x_1 - x_2)^2(x_2 - x_0) \\ & -y_1 \cdot (x - x_0)(x - x_2) \cdot (x_0 - x_1)(x_0 - x_2)^2(x_2 - x_1) \\ & -y_2 \cdot (x - x_0)(x - x_1) \cdot (x_0 - x_1)^2(x_0 - x_2)(x_1 - x_2) \\ & = 0 \end{aligned}$$

$$\begin{aligned} & -y_0 \cdot (x - x_1)(x - x_2) \cdot (x_0 - x_1)(x_1 - x_2)^2(x_0 - x_2) \\ & + y_1 \cdot (x - x_0)(x - x_2) \cdot (x_0 - x_1)(x_0 - x_2)^2(x_1 - x_2) \\ & -y_2 \cdot (x - x_0)(x - x_1) \cdot (x_0 - x_1)^2(x_0 - x_2)(x_1 - x_2) \\ & = 0 \end{aligned}$$

$$\begin{aligned} & (x_0 - x_1)(x_1 - x_2)(x_0 - x_2) \cdot (\\ & -y_0 \cdot (x - x_1)(x - x_2) \cdot (x_1 - x_2) \\ & + y_1 \cdot (x - x_0)(x - x_2) \cdot (x_0 - x_2) \\ & -y_2 \cdot (x - x_0)(x - x_1) \cdot (x_0 - x_1) \\ &) = 0 \end{aligned}$$

Введём следующие обозначения:

$$\begin{aligned} m &= (x_0 - x_1) \\ n &= (x_1 - x_2) \\ k &= (x_0 - x_2) \end{aligned}$$

Тогда:

$$\begin{aligned} & -y_0 \cdot (x - x_1)(x - x_2) \cdot n \\ & + y_1 \cdot (x - x_0)(x - x_2) \cdot k \\ & - y_2 \cdot (x - x_0)(x - x_1) \cdot m \\ & = 0 \end{aligned}$$

$$\begin{aligned} & -y_0 \cdot (x^2 - xx_2 - xx_1 + x_1x_2) \cdot n \\ & + y_1 \cdot (x^2 - xx_2 - xx_0 + x_0x_2) \cdot k \\ & - y_2 \cdot (x^2 - xx_1 - xx_0 + x_0x_1) \cdot m \\ & = 0 \end{aligned}$$

$$\begin{aligned} & -y_0 \cdot (x^2 - x(x_2 + x_1) + x_1x_2) \cdot n \\ & + y_1 \cdot (x^2 - x(x_2 + x_0) + x_0x_2) \cdot k \\ & - y_2 \cdot (x^2 - x(x_1 + x_0) + x_0x_1) \cdot m \\ & = 0 \end{aligned}$$

Введём следующие обозначения:

$$\begin{aligned} e &= x_2 + x_1 & o &= x_1x_2 \\ f &= x_2 + x_0 & p &= x_0x_2 \\ g &= x_1 + x_0 & q &= x_0x_1 \end{aligned}$$

Тогда:

$$\begin{aligned} & -y_0 \cdot (x^2 - xe + o) \cdot n \\ & + y_1 \cdot (x^2 - xf + p) \cdot k \\ & - y_2 \cdot (x^2 - xg + q) \cdot m \\ & = 0 \end{aligned}$$

$$\begin{aligned} & (-y_0n + y_1k - y_2m)x^2 \\ & - (-y_0ne + y_1kf - y_2mg)x \\ & + (-y_0no + y_1kp - y_2mq) \\ & = 0 \end{aligned}$$

Введём следующие обозначения:

$$u = y_0n$$

$$v = y_1k$$

$$w = y_2m$$

$$a = -u + v - w$$

$$b = -ue + vf - wg$$

$$c = -uo + vp - wq$$

Далее решается квадратное уравнение, при условии $a \neq 0$.

$$ax^2 - bx + c = 0$$

$$x_{1,2} = \frac{b \pm \sqrt{b^2 - 4ac}}{2a}$$

В случае $a = 0$, уравнение – линейное, и принимает вид:

$$-bx + c = 0$$

$$x = \frac{c}{b}$$

Вывод коэффициентов для метода парабол (версия 2)

$$L_2(x) = y_0 l_0(x) + y_1 l_1(x) + y_2 l_2(x) = 0$$

$$\begin{aligned} & y_0 \cdot \frac{(x - x_1)(x - x_2)}{(x_0 - x_1)(x_0 - x_2)} \\ & + y_1 \cdot \frac{(x - x_0)(x - x_2)}{(x_1 - x_0)(x_1 - x_2)} \\ & + y_2 \cdot \frac{(x - x_0)(x - x_1)}{(x_2 - x_0)(x_2 - x_1)} \\ & = 0 \end{aligned}$$

Каждую из дробей с коэффициентом y_i можно представить как квадратное уравнение:

$$y_0 \cdot \frac{(x - x_1)(x - x_2)}{(x_0 - x_1)(x_0 - x_2)} = y_0 \cdot \frac{x^2 - x(x_1 + x_2) + x_1 x_2}{(x_0 - x_1)(x_0 - x_2)} = a_0 x^2 + b_0 x + c_0, \text{ где } \begin{cases} a_0 = \frac{y_0}{(x_0 - x_1)(x_0 - x_2)} \\ b_0 = -\frac{y_0(x_1 + x_2)}{(x_0 - x_1)(x_0 - x_2)} \\ c_0 = \frac{y_0 x_1 x_2}{(x_0 - x_1)(x_0 - x_2)} \end{cases}$$

Аналогично можно проделать для дробей при коэффициентах y_1 и y_2 . Тогда исходное уравнение с тремя дробями можно представить в виде следующего квадратного уравнения:

$$Ax^2 + Bx + C = 0$$

где:

$$\begin{aligned} A &= a_0 + a_1 + a_2 \\ B &= b_0 + b_1 + b_2 \\ C &= c_0 + c_1 + c_2 \end{aligned}$$