Министерство науки и высшего образования Российской Федерации

Новосибирский государственный технический университет

Кафедра автоматизированных систем управления



**Лабораторная работа №5**

**«Расчет переходной функции численными методами»**

|  |  |
| --- | --- |
| Группа: АВТ-813  Студент:  Пайхаев А. Е. | Преподаватель:  Достовалов Дмитрий Николаевич,  Заведующий кафедрой АСУ, доцент  кафедры Автоматизированных систем управления |

Новосибирск

2020

**Цель работы:** построить переходную характеристику системы с заданной передаточной функцией. Сравнить результаты расчета разными численными методами.

1. **Передаточная функция, в соответствии с вариантом задания.**

Вариант 36.

Согласно варианту передаточная функция выглядит следующим образом:

1. **Структурная схема в Matlab.**

На рисунке 1 представлена структурная схема заданной передаточной функции.

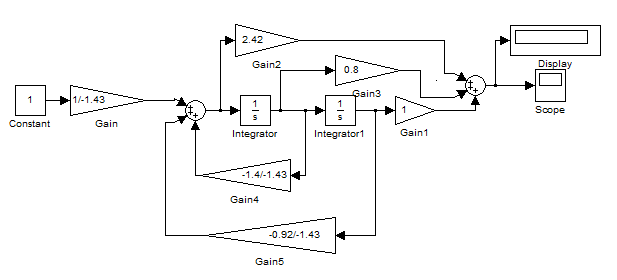


Рис. 1 – Структурная схема передаточной функции.

1. **График переходной характеристики из Matlab.**

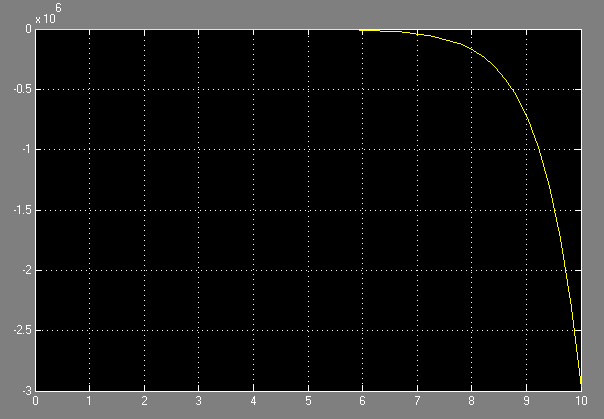
****

Рис. 2 – график переходной характеристики из Matlab.

1. **Значения и , полученные в Matlab.**

Проверим систему на устойчивость. Характеристическое уравнение системы имеет вид:

Корни:

По корневому портрету получается седло, значит система неустойчива.

Так как система неустойчива, мы берем любым. Пусть будет

Значение , его видно на рисунке 3 в блоке Display.

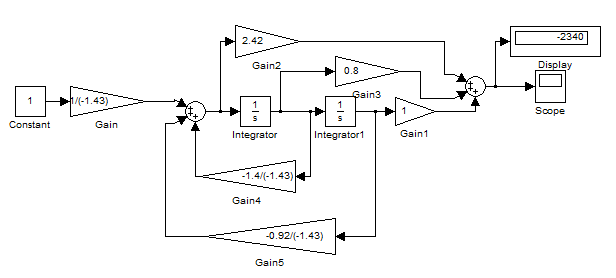


Рис. 3 – Структурная схема со значением y.

1. **Система уравнений, используемая для расчета переходного процесса.**
2. **Значения y, полученные с помощью наших программ.** **Выводы об оценке точности расчетов.**

Пусть шаг h = 0,001.

Полученное значение в нашей программе по методу Эйлера:

Полученное значение в нашей программе по методу Рунге-Кутты 4 порядка:

Полученное значение в Matlab:

**Абсолютная погрешность по методу Эйлера**:

**Относительная погрешность по методу Эйлера**:

**Абсолютная погрешность по методу Рунге-Кутты 4 порядка**:

**Относительная погрешность по методу Рунге-Кутты 4 порядка**:

Можем сделать вывод о том, что при относительной ошибке в 0,512%, которая является незначительной, результат работы программы достаточно точен для метода Эйлера. Относительная ошибка по методу Рунге-Кутты 4 порядка составляет всего 0,299%, что меньше результата по методу Эйлера. Т. е. метод Рунге-Кутты обладает более высокой точностью по отношению к методу Эйлера, что соответствует теории.

1. **Скриншоты из программы**

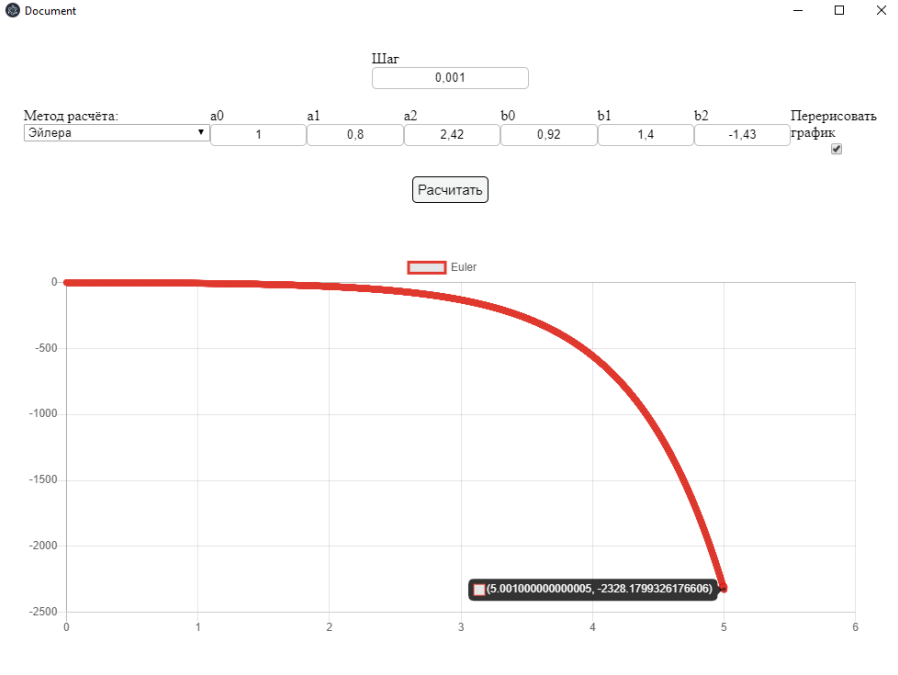
****

Рис.4 – Пример работы программы для метода Эйлера

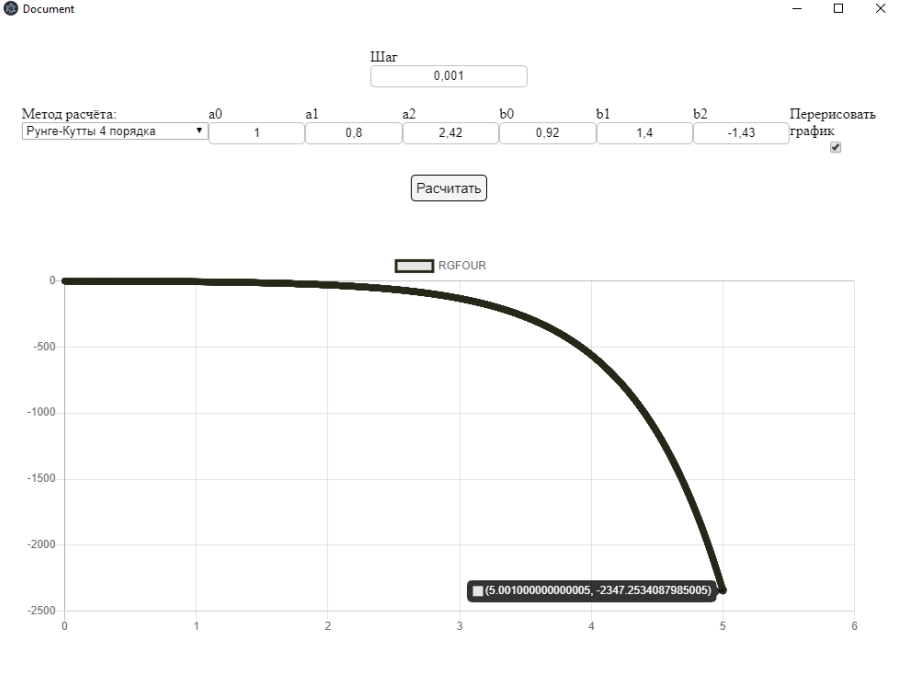


Рис.5 – Пример работы программы для метода Рунге-Кутты 4 порядка

1. **Материалы по дополнительным заданиям**

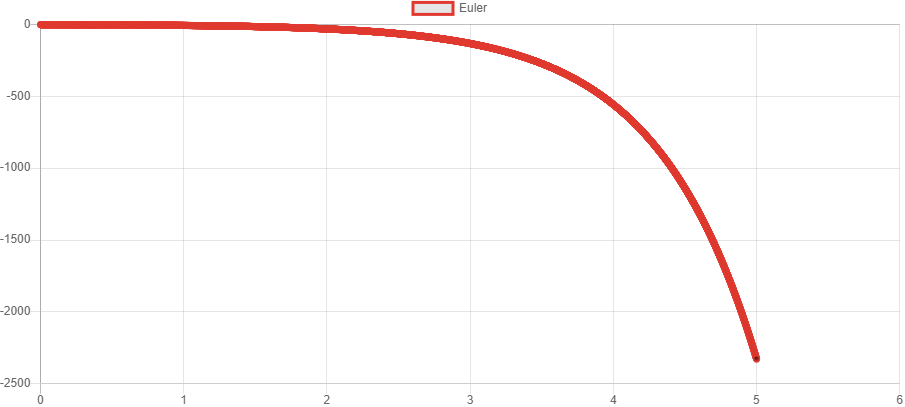


Рис.6 – График для метода Эйлера, построенный нашей программой

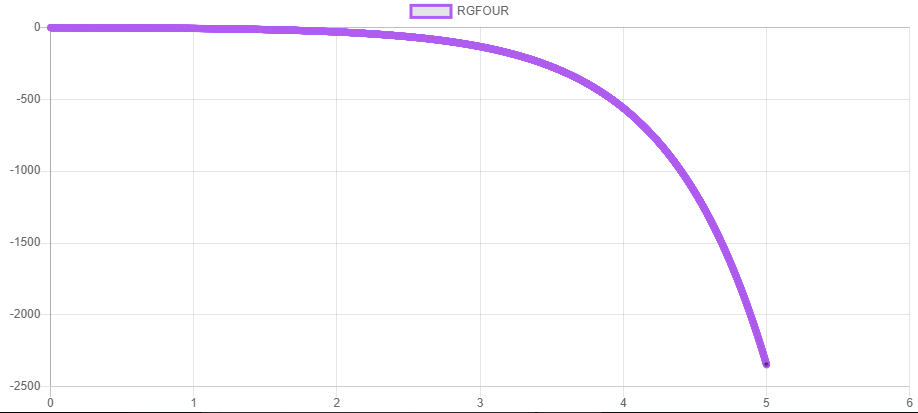


Рис.7 – График для метода Рунге-Кутты, построенный нашей программой

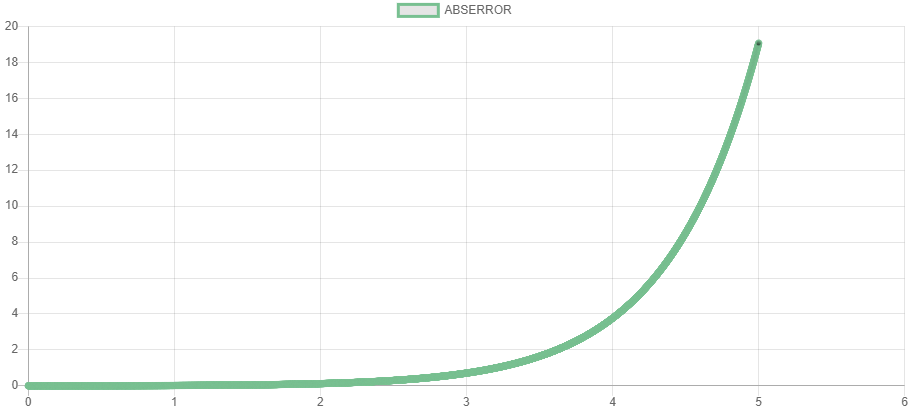


Рис.8 – График абсолютной погрешности для методов Эйлера и Рунге-Кутты, построенный нашей программой (Рунге-Кутта – считаем, как точный)

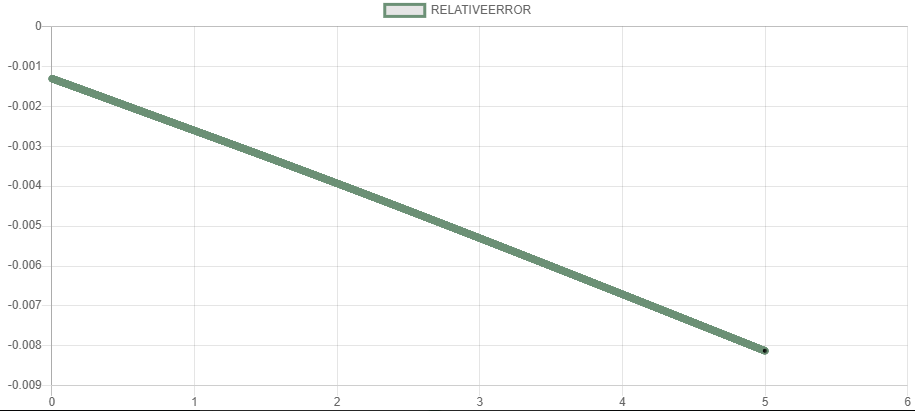


Рис.9 – График относительной погрешности для методов Эйлера и Рунге-Кутты, построенный нашей программой (Рунге-Кутта – считаем, как точный)

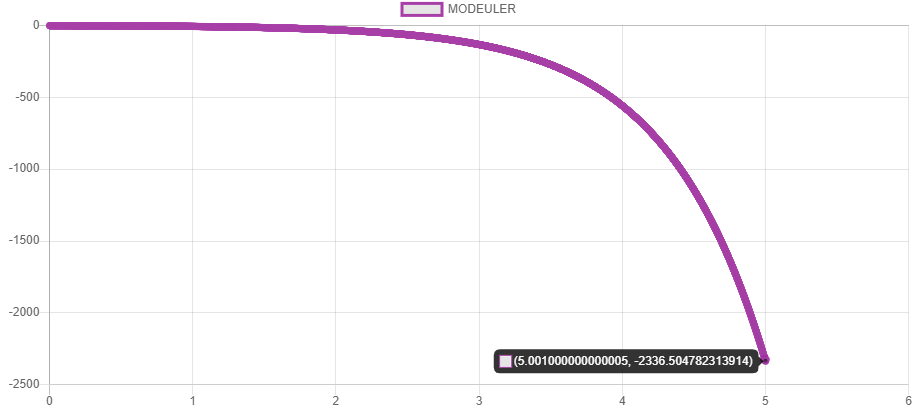


Рис.10 – График для дополнительного метода (модифицированный метод Эйлера) построенный нашей программой

Полученное значение в нашей программе по модифицированному методу Эйлера:

Полученное значение в Matlab:

**Абсолютная погрешность по модифицированному методу Эйлера**:

**Относительная погрешность по модифицированному методу Эйлера**:

1. **Приложение – исходные коды программы**
2. export class DiffEquations
3. {
4. constructor( coefs )
5. {
6. this.coefs = coefs
7. }
8. absoluteError( step ) {
9. let firstResults = this.euler( step )
10. let secondResults = this.rgFour( step )
11. let results = []
12. for( const [ index, result ] of firstResults.entries() ) {
13. results.push( { x: result.x, y: result.y - secondResults[ index ].y } )
14. }
15. return results
16. }
17. relativeError( step ) {
18. let absoluteErrorResults = this.absoluteError( step )
19. let secondResults = this.rgFour( step )
20. let results = []
21. for( const [ index, result ] of absoluteErrorResults.entries() ) {
22. results.push( { x: result.x, y: result.y / secondResults[ index ].y } )
23. }
24. return results
25. }
26. euler( step ) {
27. let x1 = 0, x2 = 0, x3 = 0, y = 0, x = 0
28. let results = []
29. for( let i = 0; i < 5001; i++ ) {
30. x1 += step \* x2
31. x2 += step \* x3
32. x3 = 1 / this.coefs.b2 - this.coefs.b1 / this.coefs.b2 \* x2 - this.coefs.b0 / this.coefs.b2 \* x1
34. y = this.coefs.a0 \* x1 + this.coefs.a1 \* x2 + this.coefs.a2 \* x3
35. x += step
36. results.push( { x, y } )
37. }
38. return results
39. }
40. modEuler( step ) {
41. let x1 = 0, x2 = 0, x3 = 0, y = 0, x = 0
42. let results = []
43. for( let i = 0; i < 5001; i++ ) {
44. x1 += step \* ( x2 + step / 2 \* x2 )
45. x2 += step \* ( x3 + step / 2 \* x3 )
46. x3 = 1 / this.coefs.b2 - this.coefs.b1 / this.coefs.b2 \* x2 - this.coefs.b0 / this.coefs.b2 \* x1
47. y = this.coefs.a0 \* x1 + this.coefs.a1 \* x2 + this.coefs.a2 \* x3
48. x += step
49. results.push( { x, y } )
50. }
51. return results
52. }
53. rgFour( step ) {
54. let x1 = 0, x2 = 0, y = 0, x = 0
55. let k1 = 0, k2 = 0, k3 = 0, k4 = 0, m1 =0, m2 = 0, m3 = 0, m4 =0
56. let results = []
57. for ( let i = 0; i < 5001; i++ ) {
58. m1 = x2
59. m2 = x2 + step / 2 \* m1
60. m3 = x2 + step / 2 \* m2
61. m4 = x2 + step \* m3
62. x1 += ( m1 + 2 \* m2 + 2 \* m3 + m4 ) \* step / 6
63. k1 = ( -this.coefs.b0 \* x1 - this.coefs.b1 \* x2 + 1 ) / this.coefs.b2
64. k2 =( -this.coefs.b0 \* ( x1 + step / 2 \* m1 ) -this.coefs.b1 \* ( x2 + step \* k1 / 2 ) + 1 ) / this.coefs.b2
65. k3 = ( -this.coefs.b0 \* ( x1 + step / 2 \* m2 ) -this.coefs.b1 \* ( x2 + step \* k2 / 2 ) + 1 ) / this.coefs.b2
66. k4 = ( -this.coefs.b0 \* ( x1 + step \* m3 ) -this.coefs.b1 \* ( x2 + step \* k3 ) + 1 ) / this.coefs.b2
67. x2 += ( k1 + 2 \* k2 + 2 \* k3 + k4 ) \* step / 6
68. y = this.coefs.a0 \* x1 + this.coefs.a1 \* x2 + this.coefs.a2 \* ( -this.coefs.b0 \* x1 - this.coefs.b1 \* x2 + 1 ) / this.coefs.b2
69. x += step
70. results.push( { x, y } )
71. }
72. return results
73. }
74. }