Санкт-Петербургский политехнический университет Петра Великого
Институт компьютерных наук и технологий
Высшая школа интеллектуальных систем и суперкомпьютерных технологий

Отчёт по лабораторной работе №3

Дисциплина: Вычислительная математика

Вариант №2

Выполнил студент гр. 3530901/90003 _____ Бехтольд Ек.В.
(подпись)

Принял старший преподаватель _____ Цыган В.Н.
(подпись)

“___”_____ 2021 г.

Санкт-Петербург
2021

Задание:

Привести дифференциальное уравнение: $ty''-(t+1)y'-2(t-1)y = 0$ к системе двух дифференциальных уравнений первого порядка.

Начальные условия: $y(t=1) = e^2$; $y'(t=1) = 2e^2$

Точное решение: $y(t) = e^{2t}$

Решить на интервале 1<= t <= 2

1) используя программу RKF45 с шагом печати h_{print} = 0.1 и выбранной погрешностью EPS в диапазоне 0.001 – 0.00001;

2) Используя метод Адамса 2-й степени точности;

3) используя метод Рунге-Кутты 3-й степени точности;

4) используя метод Рунге-Кутты 4-й степени точности.

Исследовать влияние величины шага интегрирования h_{int} на величины локальной и глобальной погрешностей решения заданного уравнения, для этого взять шаг вычисления h_{int}=(0.05, 0.025, 0.0125) .

Цель работы

Использовать в рабочих целях реализацию базовых подпрограмм, таких как RKF45, а также рассмотреть методы Адамса 2-й степени точности, Рунге-Кутты 3-й и 4-й степени точности.

Используемая среда разработки и язык программирования

Для решения поставленной задачи в качестве языка программирования был выбран язык Python, а среда разработки Pycharm Community Edition.

Ход выполнения работы

Сначала приведём дифференциальное уравнение: $ty''-(t+1)y'-2(t-1)y = 0$ к системе двух диффернциальных уравнений первого порядка:

пусть $y_0 = y$, тогда $y_1 = y'$. После замены получим:

$$\begin{cases} y_0 = y_1 \\ ty'_1-(t+1)y_1-2(t-1)y_0=0 \end{cases} \Rightarrow \begin{cases} y_0 = y_1 \\ y'_1=\dfrac{((t+1)y_1+2(t-1)y_0)}{t} \end{cases}$$

После получения системы двух дифференциальных уравнений первого порядка необходимо составить программу, которая решает полученные функции указанными в задании методами.

С начала проведем вычисления, используя программу RKF45, используя библиотеку SciPy. Далее посчитаем то же самое остальными методами, которые были написаны самостоятельно. Метод Адамса для старта использует метод Рунге-Кутты 4 степени точности. Так как метод Адамса не самостартующий, для преодоления больших погрешностей, перед началом работы мы, с помощью функции RK4 просчитываем значения 2 точек левее точки старта и саму точку старта. Затем уже, на основании этих данных, вычисляются с большей точностью дальнейшие точки.

Расчетные соотношения для метода Адамса 2й степени точности:

$$x_n = x_{n-1} + \frac{h}{2}\left(3f(t_{n-1}, x_{n-1}) - f(t_{n-2}, x_{n-2})\right)$$

Для метода Рунге-Кутты 3-й степени точности :

$$k_1 = hf(t_n, x_n);\ k_2 = hf\left(t_n + \frac{h}{2}, x_n + \frac{k_1}{2}\right);\quad k_3 = hf\left(t_n + \frac{3h}{4}, x_n + \frac{3k_2}{4}\right);\quad x_{n+1} = x_n + \left(\frac{2k_1 + 3k_2 + 4k_3}{9}\right)$$

Для метода Рунге-Кутты 4-й степени точности:

$$k_1 = hf(t_n, x_n);\ k_2 = hf\left(t_n + \frac{h}{2}, x_n + \frac{k_1}{2}\right);\quad k_3 = hf\left(t_n + \frac{h}{2}, x_n + \frac{k_2}{2}\right);\ k_4 = hf(t_n + h, x_n + k_3);\quad x_{n+1} = x_n + \left(\frac{k_1 + 2k_2 + 2k_3 + k_4}{6}\right)$$

Составленная программа состоит из шести файлов: main.py, rkf45.py, Runge_Kutta3.py, Runge_Kutta4.py, Adams2.py и ExactSolution.py. Листинги программы приведен в конце отчета для подробного ознакомления. Вкратце о них можно сказать следующее.

В файле main.py расположены исходные условия (интервал интегрирования, начальные условия, шаг, приведенная система дифференциальных уравнений первого порядка) и функция вывода графиков и результатов в консоль.

В файле rkf45.py вычисляются значения функции с помощью библиотечной

реализации программы rkf45. Программа rkf45 с шагом печати h_print = 0.1 и погрешностью atol = 0.0001. Интегрирование осуществляется при помощи функции integrate.ode(), которая возвращает настраиваемый объект, при помощи которого можно решать произвольные системы вида y'= f(t,y). Этот объект можно настроить на использование методов Рунге-Кутта при помощи параметра "dopri5".

В файлах Runge_Kutta3.py, Runge_Kutta4.py представлены реализации соответствующих методов. Данные реализации имеют методы которые рассчитывают значения функции в соответствии с приведенными выше формулами. Эти методы на вход принимают систему ДУ, массив точек интегрирования и начальные условия.

В файле Adams2.py находится реализация метода Адамса степени точности. Суть методов Адамса в пошаговом вычислении значений решения y=y(t) дифференциального уравнения вида y’=f (t, y). Использование трёх точек $\{t_n, t_{n-1}, t_{n-2}\}$ и полинома 2-й степени приведёт к формуле $x_n = x_{n-1} + \frac{h}{2}\left(3f(t_{n-1}, x_{n-1}) - f(t_{n-2}, x_{n-2})\right)$. Данный метод имеет 2-ю степень точности и является явным. Методы Адамса не являются самостартующими, то есть они требуют для начала интегрирования специальных стартовых алгоритмов для расчета дополнительных начальных условий.

В двумерном массиве y хранятся значения функции y и производные y' . Чтобы вычислить начальные условия для «старта» метода Адамса, был использован метод Рунге-Кутты 4 степени. С его помощью были получены значения функции f в точках t = 0.8 и t = 0.9. Они были сохранены на нулевую и первую позицию массива y, на второй позиции - начальное условие для точки t = 1.0. Затем была применена сама формула для вычисления интеграла. Значения в точках t = 0.8 и t = 0.9 не относятся к нужному промежутку решений [1;2], поэтому возвращается массив значений функции y со второго индекса.

В файле ExactSolution.py расположено точное решение данной системы дифференциальных уравнений для вычисления погрешности и возможности оценки качества решения каждого метода.

Результаты выполнения программы

При шаге интегрирования h = 0.1 наиболее точный результат, как и ожидалось дает метод Рунге-Кутты 4-5 порядка точности. Наихудший результат даёт метод Адамса 2 порядка.

Table of Values:

t	Exact	RKF45	RK4	RK3	ADAMS
1.00	7.38905609893065	7.38905609893065	7.38905609893065	7.38905609893065	7.369260124712355
1.10	9.025013499434122	9.025013516869686	9.024993119233898	9.024500515493969	8.975071509119614
1.20	11.023176380641605	11.023176538875457	11.023126595832284	11.021923296256636	10.930666949384266
1.30	13.463738035001697	13.463738395525422	13.463646824149555	13.46144231916144	13.312359883287588
1.40	16.444646771097062	16.444647415729236	16.44449823101627	16.440908219135842	16.213001153335433
1.50	20.085536923187686	20.085537960060527	20.085310139363273	20.07982923830458	19.745665511007317
1.60	24.532530197109374	24.532531768309873	24.532197804218306	24.524164776382666	24.048065048975966
1.70	29.96410004739705	29.96410233870179	29.963626398072243	29.95217991355537	29.287918012568033
1.80	36.59823444367804	36.59823769693419	36.597573282605445	36.581595734422294	35.669486911440855
1.90	44.70118449330089	44.70118896468737	44.7002760073743	44.67832225697444	43.441541183616317
2.00	54.598150033144336	54.598155598718776	54.59691711540698	54.56712424985146	52.907054847557127

Table of Errors:

	RKF45	RK4	RK3	ADAMS
Local Error:	1.743556410360725e-08	2.038020022432363e-05	0.0005129839401529779	0.049941990314508
Global Error:	1.9370420464071003e-05	0.0041453067193089055	0.1043203044531662	5.925275787917668

Рис. 1. Вычисленные значения функции с шагом печать h = 0.1.

1.86	41.26439410861086	41.26439410867181	41.264394015541626	41.26437082576027	41.2525769468987
1.87	42.097990164996965	42.09799016505987	42.097990068943545	42.09796613560172	42.0857957120088
1.88	42.94842597876309	42.948425978828006	42.9484258796429	42.94840118216239	42.9358438139001
1.89	43.81604173557404	43.816041735641015	43.81604163330236	43.816016150576985	43.8030611711970
1.90	44.70118449330089	44.70118449336999	44.70118438779086	44.70115809817351	44.6877945681939
1.91	45.60420832084881	45.60420832092009	45.604208212011315	45.604181093300745	45.5903977935277
1.92	46.525474439789285	46.5254744398628	46.52547432753295	46.52544635696021	46.5112317816516
1.93	47.465351368853604	47.46535136892942	47.46535125308471	47.46532240729928	47.4506647571659
1.94	48.42421507134526	48.42421507142343	48.424214951967706	48.42418520702327	48.4090723820644
1.95	49.40244910553026	49.40244910561086	49.40244898244549	49.40241831378542	49.3868379059547
1.96	50.40044477806558	50.400444778148675	50.40044465117253	50.40041303361497	50.3843523193127
1.97	51.41860130052701	51.41860130061267	51.418601169722045	51.4185685774447	51.4020145098325
1.98	52.45732594909914	52.457325949187435	52.457325814276	52.457292220800525	52.4402314219343
1.99	53.51703422749126	53.51703422758225	53.51703408854097	53.51699946671699	53.4994182194940
2.00	54.598150033144336	54.598150033238106	54.598149889955174	54.598114211943965	54.5799984518595

Table of Errors:

	RKF45	RK4	RK3	ADAMS
Local Error:	1.2967404927621828e-13	1.9769963444105088e-10	4.945807408063274e-08	4.93991249097547
Global Error:	2.708514301730247e-09	4.1361288012353725e-06	0.001034723008584315	0.5269978607

Рис. 2. Вычисленные значения функции с шагом печать h = 0.01.

С уменьшением шага интегрирования мы можем наблюдать как повышается точность вычислений и уменьшается погрешность. (рис. 2-5).

1.30	13.463738035001697	13.463738050827361	13.463731840766492	13.463427281404847	13.427300697330
1.35	14.879731724872844	14.87973174583232	14.87972373823644	14.879331050499259	14.833682919727
1.40	16.444646771097062	16.44464679809516	16.44463668355655	16.4441406993101	16.38737032281993
1.45	18.174145369443075	18.17414540351814	18.174132827430096	18.173516162854213	18.10379172525655
1.50	20.085536923187686	20.085536965529673	20.08552152200161	20.08476427931438	19.99999196790404
1.55	22.197951281441654	22.197951333412544	22.19793255840512	22.197011989355612	22.09480117682682
1.60	24.532530197109374	24.532530260266185	24.53250762384972	24.53139775023618	24.40902175495564
1.65	27.11263892065792	27.11263899677867	27.112611894406353	27.111283080302687	26.96563495935765
1.70	29.96410004739705	29.96410013851003	29.964067881184317	29.962486350914524	29.79002911551352
1.75	33.11545195869236	33.115452067109096	33.11541387030504	33.11354116548571	32.91025173487267
1.80	36.59823444367804	36.59823457203044	36.59818954322325	36.59598191138929	36.3572880393279
1.85	40.44730436006745	40.447304511349635	40.44725163597533	40.44465934240373	40.16536865848345
1.90	44.70118449330089	44.701184670916255	44.701122796573884	44.6980893499132	44.37230955528957
1.95	49.40244910553026	49.40244931334445	49.40237713202523	49.3988384132124	49.01988755565884
2.00	54.598150033144336	54.59815027554484	54.59806630364794	54.59394958633524	54.154255211243197

Table of Errors:

	RKF45	RK4	RK3	ADAMS
Local Error:	7.847766880786367e-11	6.261657148343147e-07	3.1413899460375205e-05	0.00622325631841
Global Error:	1.464755396085593e-06	0.0005181692291778006	0.025995142229247037	2.8110436727

Рис. 3. Вычисленные значения функции с шагом печать h = 0.05.

1.65	27.112638920657762	27.112638923463532	27.11263715978044	27.112462538238873	27.07576678120663
1.67	28.50273364376714	28.502733646830215	28.502731721409273	28.502541086293157	28.46253515603194
1.70	29.964100047396865	29.96410005073625	29.96409795162874	29.963890119904974	29.92033112320416
1.72	31.50039230874777	31.50039231238375	31.500390026840876	31.50016373626093	31.45279257854367
1.75	33.115451958692134	33.11545196264635	33.115449477068594	33.11520338115547	33.06374374385434
1.77	34.81331748760183	34.813317491897344	34.81331479178076	34.81304745451012	34.75720471017981
1.80	36.59823444367778	36.5982344483392	36.598231518218405	36.597941408375725	36.53740146984693
1.82	38.474666049031896	38.47466605408545	38.47466287747318	38.47434836266765	38.40877646233095
1.85	40.44730436006714	40.447304365540795	40.44730092486375	40.44696026517858	40.37599966025959
1.87	42.52108200006252	42.52108200598605	42.52107828251689	42.5207096237745	42.44398022322078
1.90	44.70118449330054	44.7011844997057	44.70118047349089	44.700781840110096	44.61787874845584
1.92	46.99306323157897	46.99306323849957	46.9930588882829	46.99262817570406	46.90312014900949
1.95	49.40244910552984	49.4024491130019	49.40244441614332	49.40197938279588	49.30540719147380
1.97	51.93536683483107	51.93536684289295	51.93536177528256	51.93486003406797	51.83073472710909
2.00	54.59815003314385	54.59815004183638	54.598144577803225	54.59760358706465	54.48540465185543

Table of Errors:

	RKF45	RK4	RK3	ADAMS
Local Error:	3.091837896818106e-11	1.940383942411472e-08	1.9436371978542866e-06	0.00077460303909
Global Error:	1.0308739195608041e-07	6.469663356600819e-05	0.006480486576228017	1.3535401412

Рис. 4. Вычисленные значения функции с шагом печать h = 0.025.

1.85	40.44730436006714	40.4473043602458	40.447304140851166	40.44726048025181	40.429421110343
1.86	41.4712327448305	41.47123274501637	41.47123251675966	41.471187092565685	41.452631053057
1.87	42.52108200006252	42.52108200025586	42.52108176282899	42.521034513729305	42.501736953667
1.89	43.59750831572322	43.59750831592429	43.59750806900925	43.59745893171878	43.577394201267
1.90	44.70118449330054	44.7011844935096	44.70118423677817	44.70113314597721	44.680274771893
1.91	45.83280036633317	45.8328003665505	45.8328000996639	45.83274698793566	45.811067648323
1.92	46.99306323157897	46.99306323180486	46.99306295441346	46.99300775217972	46.970479250487
1.94	48.1826982910985	48.18269829133323	48.18269800307621	48.18264063853112	48.159233876776
1.95	49.40244910552984	49.40244910577373	49.40244880627873	49.402389205320596	49.378074156524
1.96	50.65307805883828	50.653078059091634	50.6530777479744	50.65301583413549	50.627761513934
1.97	51.93536683483107	51.93536683509421	51.935366511958236	51.93530220633244	51.909076643750
1.99	53.25011690573524	53.25011690600849	53.25011657044465	53.25004979161305	53.222819998967
2.00	54.59815003314385	54.598150033427565	54.598149685013716	54.59808034896792	54.569812290881
2.01	55.98030878164376	55.98030878193829	55.98030842023885	55.98023644030207	55.950895001802

Table of Errors:

	RKF45	RK4	RK3	ADAMS
Local Error:	4.920508445138694e-13	6.038369804173271e-10	1.2086842193781422e-07	9.654500323463
Global Error:	6.879879776988673e-09	8.441397461922406e-06	0.0016896896782769844	0.691293582

Рис. 5. Вычисленные значения функции с шагом печать h = 0.0125.

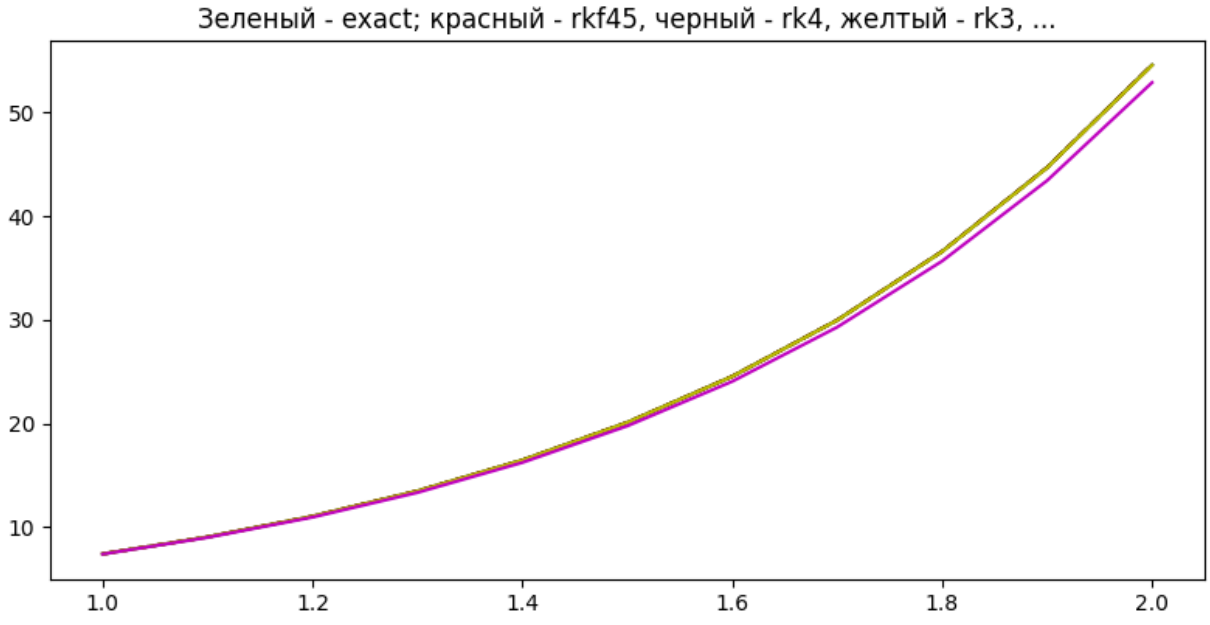

Рис. 6. График функции при h = 0.1.

Рис. 7. График функции при h = 0.01.

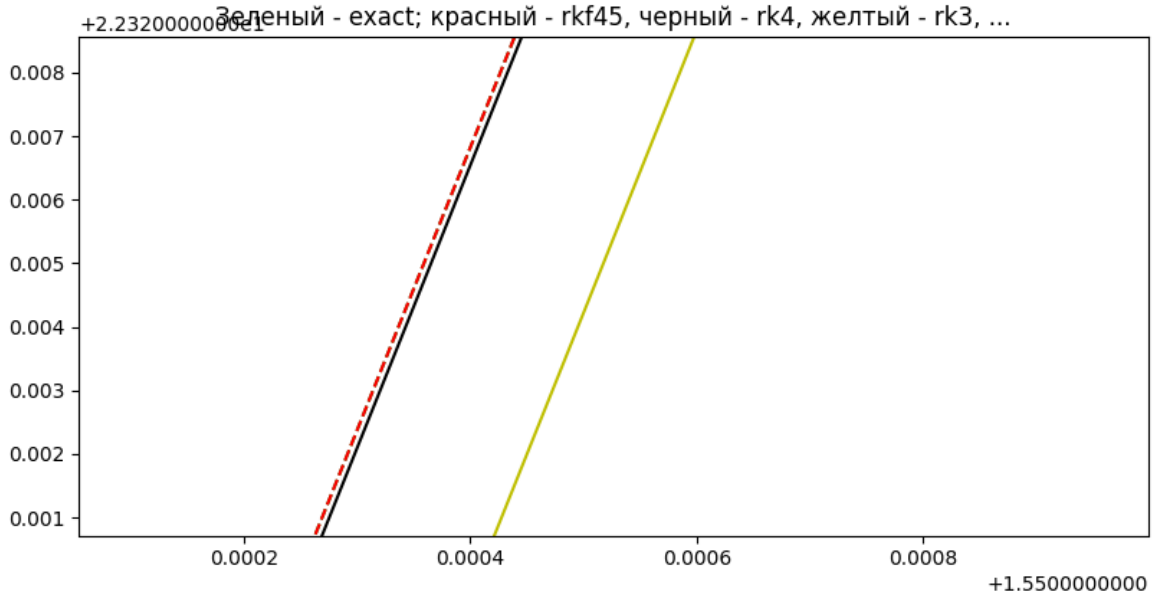

Рис. 8. Увеличенный фрагмент графика на рис. 6.

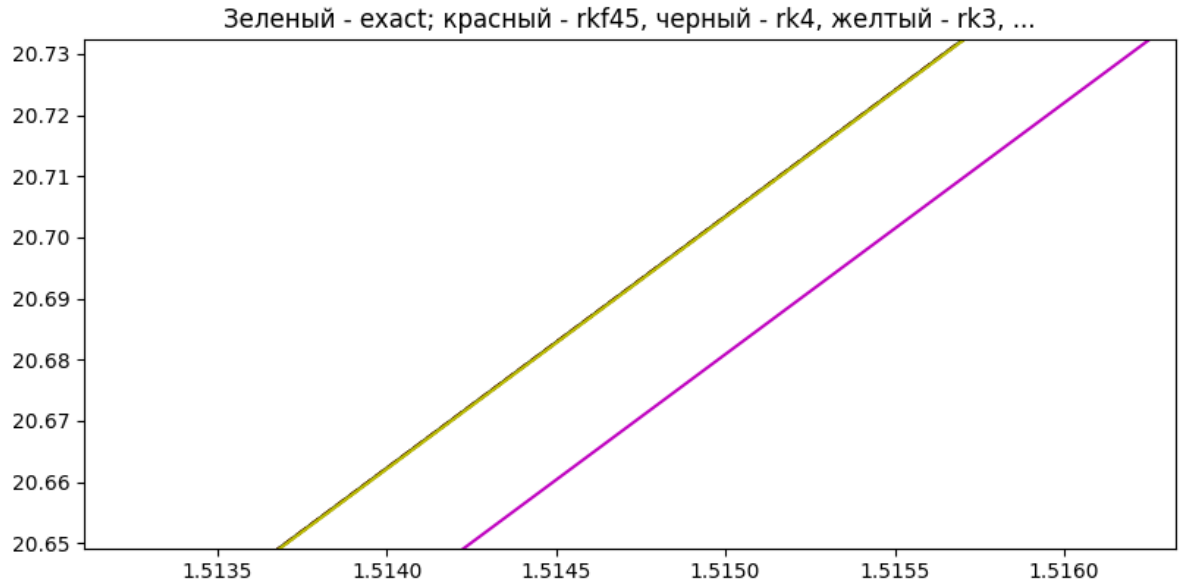

Рис. 9. Увеличенный фрагмент графика на рис. 7.

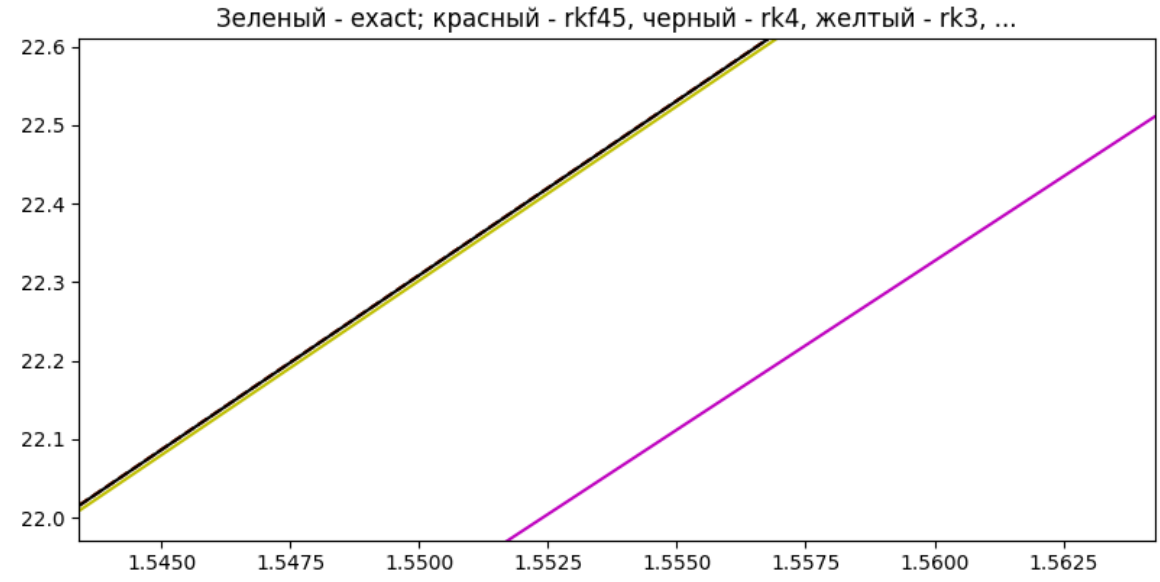

Рис. 10. Увеличенный фрагмент графика на рис. 6.

Из графиков видно что при большем шаге интегрирования мы ещё можем различить разные кривые, но с уменьшение шага кривые разных методов сливаются в одну и только при многократном увеличении масштаба видны различия.

Для исследования зависимости изменения локальной погрешности при изменении шага интегрирования, были построены таблицы. В таблице 1 представлены глобальные и локальные погрешности первого шага (для h_{int}={0.1,0.05,0.025,0.0125}) для каждого из рассмотренных методов. В таблице 2

представлены соотношения локальных погрешностей первого шага каждого метода $h_i / h_{i+1.}$

Таблица 1. Погрешности исследуемых методов.

h =	Погрешность	RKF45	RK4	RK3	ADAMS
0.1	Локальная	4.4943556e-08	2.038020e-05	5.129839e-04	0.04994199
	Глобальная	1.937042e-05	4.145306e-03	0.104320304	5.925275787
0.05	Локальная	6.847766e-10	6.261657e-07	3.141389e-05	0.006223256
	Глобальная	1.464755e-06	5.181692e-04	0.025995142	2.811043672
0.025	Локальная	1.091837e-11	1.940383e-08	1.943637e-06	7.746030e-04
	Глобальная	1.030873e-07	6.469663e-05	6.480486e-03	1.353540141
0.0125	Локальная	1.650508e-13	6.038369e-10	1.208684e-07	9.654500e-05
	Глобальная	6.879879e-09	8.441396e-06	1.689689e-03	0.691293582

Таблица 2. Соотношение локальных погрешностей на первом шаге.

	RKF45	RK4	RK3	ADAMS
$h_{0.1} / h_{0.05} =$	6.55E+01	3.25E+01	1.63E+01	8.03E+00
$h_{0.05} / h_{0.025} =$	6.27E+01	3.23E+01	1.62E+01	8.03E+00
$h_{0.25} / h_{0.0125} =$	6.62E+01	3.21E+01	1.61E+01	8.02E+00

RKF45 – метод пятого порядка точности, поэтому величина локальной погрешности пропорциональна h^6, то есть если шаг уменьшается в 2 раза, то локальная погрешность уменьшается в $2^6 = 64$ раза. Из таблицы 2 следует, что при уменьшении шага интегрирования локальная погрешность становится примерно в 64 раза меньше.

Также стоит отметить, что если шаг интегрирования уменьшиться в 8 раз, то погрешность уменьшиться в $8^6 = 262144$ раза. При шаге интегрирования $h_{int} = 0.1$ локальная погрешность первого шага = 4.4943556e-08, при $h_{int} = 0.0125$ локальная погрешность первого шага = 1.650508e-13. Разделив погрешность шага 0.1 на погрешность шага 0.0125, получим: 272 301, примерно такой результат был ожидаем.

Runge Kutta 3 – метод третьего порядка точности, поэтому величина локальной погрешности пропорциональна h^4, то есть если шаг уменьшается в 2 раза, то локальная погрешность уменьшается в $2^4 = 16$ раз. Из таблицы 2 следует, что при уменьшении шага интегрирования локальная погрешность становится примерно в 16

раз меньше.

Если шаг интегрирования уменьшиться в 8 раз, то погрешность уменьшиться в 8^4 = 4096 раз. При шаге интегрирования h_{int} = 0.1 локальная погрешность первого шага = 5.129839e-04, при h_{int} = 0.0125 локальная погрешность первого шага = 1.208684e-07. Разделив погрешность шага 0.1 на погрешность шага 0.0125, получим: 4 244, примерно такой результат был ожидаем.

Продолжая рассуждения находим что метод Адамса второго порядка точность поэтому величина локальной погрешности пропорциональна h^3, то есть если шаг уменьшается в 2 раза, то локальная погрешность уменьшается в 2^3 = 8 раз. Метод Рунге-Кутты 4 - метод четвертого порядка точности, поэтому величина локальной погрешности пропорциональна h^5, то есть если шаг уменьшается в 2 раза, то локальная погрешность уменьшается в 2^5 = 32 раза.

Выводы

В ходе выполнения работы было проведенно исследование работы программы RKF45(в интерпретации библиотеки `scipy` для языка программирования Python), а также реализованы и исследованы методы Адамса 2 порядка и Рунге-Кутты 3 и 4 порядка точности.

С помощью указанных методов были вычислены значения заданного дифференциального уравнения на промежутке [1,2] и произведено сравнение полученных решений с точным.

Из исследования влияния шага интегрирования на

1. локальную погрешность, была получена пропорциональная зависимость локальной погрешности от шага интегрирования. Для программы:

•RKF45 — это h^6

•RK4 - h^5

•RK3 - h^4

•ADAMS2 - h^3

2. глобальную погрешность, можно сделать вывод, что при уменьшении шага интегрирования в два раза, погрешность уменьшается, но при этом растет количество вычислений.

Листинг 1. main.py

```
from math import exp
import numpy
import matplotlib
matplotlib.use('Qt5Agg')
import matplotlib.pyplot as plt
import Adams2
import Runge_Kutta3
import Runge_Kutta4
import rkf45
import ExactSolution
# исходные данные
a = 1
b = 2
h = 0.1
initialConditions = numpy.array([exp(2), exp(2) * 2])

# система двух дифференциальных уравнений первого порядка.
def originalFunction(t, y):
    dy = numpy.zeros(y.shape)
    dy[0] = y[1]
    dy[1] = ((t + 1) * y[1] + 2 * (t - 1) * y[0]) / t
    return dy

# вывод результатов на экран
def printResult():
    plt.title('Зеленый - exact; красный - rkf45, черный - rk4, желтый - rk3, ...')
    t, y_exact = ExactSolution.pick_step(a, b + h, step=h)
    plt.plot(t, y_exact, 'g--')
    y_rkf45 = rkf45.rkf45(originalFunction, t, initialConditions)
    plt.plot(t, y_rkf45, 'r--')
    y_rk4 = Runge_Kutta4.rk4Values(originalFunction, t, initialConditions)
    plt.plot(t, y_rk4, 'k')
    y_rk3 = Runge_Kutta3.rk3Values(originalFunction, t, initialConditions)
    plt.plot(t, y_rk3, 'y')
    y_adams = Adams2.adams2(originalFunction, t, initialConditions)
    plt.plot(t, y_adams, 'm')
    plt.show()
    error_local_rkf45 = numpy.abs(y_rkf45 - y_exact)
    error_local_rk4 = numpy.abs(y_rk4 - y_exact)
    error_local_rk3 = numpy.abs(y_rk3 - y_exact)
    error_local_adams = numpy.abs(y_adams - y_exact)
    print('\tTable of Values: ')
    print('t\t\t\t\tExact\t\t\t\tRKF45\t\t\t\tRK4\t\t\t\t\tRK3\t\t\t\tADAMS')
    for it in range(0, len(t)):
        print('{:0.2f} \t {}\t {}\t {}\t {}\t {}'.format(t[it],
                                                        y_exact[it], y_rkf45[it], y_rk4[it], y_rk3[it], y_adams[it]))
    print('\tTable of Errors: ')
    print('\t\t\t\t\tRKF45\t\t\t\t\tRK4\t\t\t\t\t\tRK3\t\t\t\t\tADAMS')
    print('Local Error: \t {}\t {}\t {}\t {}'.format(error_local_rkf45[1], error_local_rk4[1], error_local_rk3[1],
                                                    error_local_adams[1]))
    print('Global Error: \t {}\t {}\t {}\t\t {}'.format(error_local_rkf45.sum(), error_local_rk4.sum(),
                                                       error_local_rk3.sum(), error_local_adams.sum()))
if __name__ == '__main__':
```

```
    printResult()
```

Листинг 2. rkf45.py

```
import numpy
from scipy import integrate
'''
    Программа  rkf45  с шагом печати  h_print = 0.1  и погрешностью atol =  0.0001.
    Интегрирование осуществляется при помощи функции integrate.ode(), которая возвращает настраиваемый объект,
    при помощи которого можно решать произвольные системы вида y'= f(t,y). Этот объект можно настроить на использование
    методов Рунге-Кутта при помощи параметра "dopri5".
'''
def rkf45(function, arguments_T, initialConditions):
    r = integrate.ode(function)\
        .set_integrator('dopri5', atol=0.0001)\
        .set_initial_value(initialConditions, arguments_T[0])
    y = numpy.zeros((len(arguments_T), len(initialConditions)))
    y[0] = initialConditions

    for it in range(1, len(arguments_T)):
        y[it] = r.integrate(arguments_T[it])

        if not r.successful():
            raise RuntimeError('Нельзя!!!')
    return y[:, 0]
```

Листинг 3. Runge_Kutta3.py

```
import numpy

def rk3(function, arguments_T, initialConditions):
    y = numpy.zeros((len(arguments_T), len(initialConditions)))
    y[0] = initialConditions
    h = arguments_T[1] - arguments_T[0]

    for it in range(1, len(arguments_T)):
        k1 = function(arguments_T[it - 1], y[it - 1])
        k2 = function(arguments_T[it - 1] + 0.5 * h, y[it - 1] + 0.5 * h * k1)
        k3 = function(arguments_T[it - 1] + 3.0 * h / 4, y[it - 1] + 3.0 * h * k2 / 4)
        y[it] = y[it - 1] + h * (2 * k1 + 3 * k2 + 4 * k3) / 9.0
    return y

def rk3Values(function, arguments_T, initialConditions):
    y = rk3(function, arguments_T, initialConditions)
    return y[:, 0]
```

Листинг 4. Runge_Kutta4.py

```
import numpy

def rk4(function, arguments_T, initialConditions):
    y = numpy.zeros((len(arguments_T), len(initialConditions)))
    y[0] = initialConditions
    h = arguments_T[1] - arguments_T[0]

    for it in range(1, len(arguments_T)):
        k1 = function(arguments_T[it - 1], y[it - 1])
        k2 = function(arguments_T[it - 1] + 0.5 * h, y[it - 1] + 0.5 * h * k1)
        k3 = function(arguments_T[it - 1] + 0.5 * h, y[it - 1] + 0.5 * h * k2)
        k4 = function(arguments_T[it - 1] + h, y[it - 1] + h * k3)
        y[it] = y[it - 1] + h * (k1 + 2 * k2 + 2 * k3 + k4) / 6.0
    return y

def rk4Values(function, arguments_T, initialConditions):
    y = rk4(function, arguments_T, initialConditions)
    return y[:, 0]
```

Листинг 5. Adams2.py

```
import numpy
import Runge_Kutta4
'''
    Метод Адамса 2 степени точности.
    Суть методов Адамса в пошаговом вычислении значений решения y = y(t)
    дифференциального уравнения вида y’ = f (t, y). Использование трёх точек {t_n, t_n-1, t_n-2}
    и полинома 2-й степени приведёт к формуле y[n] = y[n-1] + (h / 2) * (3 * f(t[n-1], y[n-1]) - f(t[n-2], y[n-2])).
    Данный метод имеет 2-ю степень точности и является явным. Методы Адамса не являются
самостартующими,
    т. е. они требуют для начала интегрирования специальных стартовых алгоритмов для расчета
дополнительных начальных условий.
    В двумерном массиве Y хранятся значения функции y и производные y' . Чтобы вычислить начальные
условия для «старта»
    метода Адамса, был использован метод Рунге-Кутты 4 степени. С его помощью были получены
значения функции f
    в точках 0.8 и 0.9. Они были сохранены на нулевую и первую позицию массива Y, на второй позиции -
начальное условие
    для точки 1.0. Затем была применена сама формула для вычисления интеграла. Значения в точках 0.8 и
0.9 не относятся
    к нужному промежутку решений [1;2], поэтому возвращается массив значений функции y со второго
индекса.
'''
def adams2(function, arguments_T, initialConditions):
    length = len(arguments_T) + 2
    y = numpy.zeros((length, len(initialConditions)))

    h = arguments_T[1] - arguments_T[0]
    startPoints = Runge_Kutta4.rk4(function, [arguments_T[0], arguments_T[0] - h, arguments_T[0] - 2 * h],
                                   initialConditions)
    y[0] = startPoints[2]
    y[1] = startPoints[1]
    y[2] = initialConditions
```

```
    arguments_T = numpy.append(arguments_T, [2.1, 2.2])

    for it in range(2, length):
        y[it] = y[it - 1] + (h / 2) * (3 * function(arguments_T[it - 1], y[it - 1]) - function(arguments_T[it - 2],
                                                                                               y[it - 2]))
    return y[2:, 0]
```

Листинг 6. ExactSolution.py

```
import math
import numpy

''' Точное решение: y(t) = e^2t '''
def exact(t):
    y = numpy.zeros(len(t))
    for it in range(0, len(t)):
        y[it] = math.exp(2.0 * t[it])
    return y

# Функция pick_step возвращает массив точек t и точных значений в этих точках y(t)
def pick_step(a, b, step):
    t = numpy.arange(a, b, step)
    y = exact(t)
    return t, y
```