Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет
имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ «Информатика и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

Отчёт по лабораторной работе №1
По дисциплине «Математическое моделирование»
Тема: «Приближенный аналитический метод Пикара в сравнении с численными методами»

Студент: Барсуков Н.М.

Группа: ИУ7-66Б

Преподаватель: Градов В.М

Москва, 2020

Содержание

1. Введение

Класс дифференциальных уравнение, которые интегрируются в явном виде очень мал и найти общее решение в квадратурах удается лишь для узкого класса функций. В связи с этим возникает потребность в приближенных методах интегрирования.

2. Аналитическая часть

Целью работы является получение навыков решения задачи Коши для ОДУ методами Пикара и явными и неявными методами Эйлера.

Исходные данные:

1) ОДУ, не имеющее аналитического решения:
   $u`(x) = u^2 + x^2$;
   $u(0) = 0$

Для выполнения поставленной цели необходимо выполнить следующие задачи:

1) Поставить задачу Коши для ОДУ
2) Понять какие методы существуют
3) Рассмотреть конкретные методы для решения поставленной задачи Коши:
   а) Метод Пикара (Приближения 1 - 4)
   б) Метод ломанных (Явный)
   в) Метод Рунге-Кутта 2 порядка
   г) Метод Рунге-Кутта 4 порядка

2.1. Задача Коши

Задача Коши - одна из основных задачи теории дифференциальных уравнений (обыкновенных и с частными производными); состоит в нохождении решения (интеграла) дифференциального уравнения, удовлетворяющего так называемым начальным условиям

2.2. Методы

Методы решения ДУ можно разбить на точные, приближенные и численные.

К точным относятся методы, позволяющие выразить решение дифференциального уравнения, через элементарные функции. Либо представить его в при помощи квадратур от элементарных функций. Однако классы, для которых разработаны

методы получения точных решенений, сравнительно узки и охватывают только малую часть возникающих на практике задач. **В данной лабораторной работе рассматриваться не будут**

Приближенными будем называть методы, в которых решение получается как предел $u(x)$ некоторой последовательности $y_n(x)$, причем $y_n(x)$ выражается через элементарные функции или при помощи квадратур. Ограничиваясь конечным числом n, получаем приближенное выражение для $u(x)$. Эти методы удобны лишь в том случае, когда большую часть промежуточных выкладок удается сделать точно. Примером может служить метод Пикара. Это выполнимо лишь для сравнительно простых задач, что сильно сужает область применения приближенных методов.

Численные методы - это алгоритмы вычисления приближенных (а иногда - точных) значений искомого решения $u(x)$ на некоторой выбранной сетке значений аргумента x_n. Численные методы не позволяют найти общего решения системы; они могут дать только какое-то частное решение. Зато данные методы применимы к очень широким классам уравнений и всем типам задач для них.

Рассмотрим детально представителей данных методологий

2.2.1. Метод Пикара

Метод Пикара - это приближенный метод решения, являющийся обобщением метода последовательных приближений.

Рассмотрим пример из лекций:

$$\begin{cases} u`(x) = u^2 + x^2 \\ u(0) = 0 \end{cases}$$

Тогда:

$$y^{(1)}(x) = 0 + \int_0^x t^2 dt = \frac{x^3}{3}$$

$$y^{(2)}(x) = 0 + \int_0^x [t^2 + (\frac{t^3}{3})^2]dt = \frac{t^3}{3} + \frac{t^7}{63}|_0^x = \frac{x^3}{3} + \frac{x^7}{63}$$

$$y^{(3)}(x) = 0 + \int_0^x [t^2 + (\frac{x^3}{3} + \frac{x^7}{63})^2]dt = \frac{x^3}{3} + \frac{x^7}{63} + \frac{2x^1 1}{2079} + \frac{x^{15}}{59535}$$

$$y^{(4)}(x) = 0 + \int_0^x [t^2 + (\frac{x^3}{3} + \frac{x^7}{63})^2]dt = \frac{x^3}{3} + \frac{x^7}{63} + \frac{2x^{11}}{2079} + \frac{x^{15}}{59535})^2]dt = \frac{x^3}{3} + \frac{x^7}{63})^2]dt =$$

$$\frac{x^3}{3} + \frac{x^7}{63} + \frac{2x^{11}}{2079} + \frac{x^{15}}{59535} + \frac{2x^{15}}{3393495} + \frac{2x^{19}}{2488563} + \frac{2x^{23}}{86266215} + \frac{x^2 3}{99411543} + \frac{2x^{27}}{3341878155} +$$

$$+ \frac{x^3 1}{109876902975}$$

2.3. Метод ломанных (явная схема)

Это простейший численный метод. В практике вычислений он употребляется очень редко из-за невысокой точности. Но на его примере удобно пояснить способы построения и исследования численных методов.

Рассмотрим задачу Коши и выберем на отрезке $[\Xi, X]$ некоторую сетку $X_n, 0 <= n <= N$ значений аргумента так, чтобы выполнялись соотношения $\xi = x_0 < x_1 < x_2 < ... < x_N$ (сетка может быть неравномерной). Разлагая решение и $u(x)$ по формуле Тейлора на интервале сетки $x_n < x < x_{n+1}$ и обозначая $u(x_n) = u_n$ получим:

$$u_{n+1} = u_n + h_n u'_n + \frac{1}{2} h_n^2 u''_n + ..., hn = x_{n+1} - x_n. \tag{1}$$

Стоящие в правой части производные можно найти, дифференцируя уравнения:

$$u'(x) = f(x, u(x)), f\xi <= x <= X, fu(\xi) = \eta \tag{2}$$

требуемое число раз:

$$u' = f(x, u), u'' = \frac{d}{dx} f(x, u) = f_x + f f_u \tag{3}$$

и т.д.

Однако использовать для расчета формулу 1 с большим числом членов невыгодно. Во-первых, даже при сравнительно простой правой части выражения для производных могут оказаться громоздкими. Во-вторых, если правая часть известна лишь приближенно, то находить ее производные нежелательно. В простейшем случае, подставляя 3 в 1 и ограничиваясь только первым членом разложения, получим схему ломаных

$$y_{n+1} = y_n + h_n f(x_n, y_n), h_n = x_{n+1} - x_n \tag{4}$$

Поскольку при такой замене можно найти только приближенные значения искомой функции в узлах, то будем обозначать эти значения через y_n в отличие от точных значений $u_n = u(x_n)$ Для численного расчета по схеме ломаных достаточно задать начальное значение $y_0 = \eta$. Затем по формуле 4 последовательно вычисляем величины $y_1, y_2, y_3, ..., y_N$

2.4. Неявная схема

$$y_{n+1} = y_n + h[f(x_{n+1}; y_{n+1})] \tag{5}$$

Решая это алгебраическое уравнение, можно определить y_{n+1} которое и будет приближенным значением искомого решения и $u(x_n)$. Схема 5 имеет второй порядок точности, допускает счет неравномерным шагом, не требует специальных приемов для начала счета. Но у этой схемы есть серьезные недостатки. Во-первых, неизвестно, имеет ли уравнение 5 вещественный корень, т. е. разрешима ли задача.

Можно привести пример, когда при большом шаге корня нет. Пусть $f(x,u) = u^2$ и $u(0) = 1$; тогда на первом шаге

$$y_1 = 1 +_{\frac{1}{2}} h(1 + y_1^2) \text{и при } h > \sqrt{1+\sqrt{2}} \text{вещественного корня нет}$$

Во-вторых, даже если корень есть, то как его найти? Метод Ньютона применять нежелательно, так как для этого надо дифференцировать $f(x,u)$. Метод деления пополам не обобщается на системы уравнений. Остается метод последовательных приближений.

2.5. Метод Рунге-Кутта

Можно строить схемы различного порядка точности. Например, схема ломаных 3 есть схема Рунге—Кутта первого порядка точности. Наиболее употребительны схемы четвертого порядка точности, образующие семейство четырехчленных схем. Приведем без вывода ту из них, которая записана в большинстве стандартных программ ЭВМ:

$$y_{n+1} = y_n + (k_1 + 4k_2 + k_3)/6$$
$$k_1 = h_n \phi(x_n, y_n),$$
$$k_2 = h_n \phi(x_n + \frac{h_n}{2}, y_n + \frac{k_1}{2}), k_3 = h_n \phi(x_n + h_n, y_n - k_1 + 2k_2)$$

Формулы более высокого порядка точности практически не употребляются. Пятичленные формулы имеют всего лишь четвертый порядок точности; шестичленные имеют шестой порядок, но слишком громоздки. Кроме того, высокий порядок реализуется лишь при наличии у правой части непрерывных производных соответствующего порядка.

3. Листинг

```
1   from polynomial import Polynomial
2   from prettytable import PrettyTable
3   import numpy as np
4
5
6   def test_func(x, u):
7   return x ** 2 + u ** 2
8
9
10  def picard(ns, x, func, pol=False):
11  """ Cauchy problem:
```

```
12          u'(x) = func(x, u) = x^2 + u^2
13          u(0) = 0
14
15          y(0) = u(0) """
16
17      y = Polynomial(0, [0.0])
18      t = Polynomial(1, [1.0])
19
20      us = []
21
22      for i in range(max(ns)):
23          under_int = func(t, y)
24          y = under_int.integral_variable_up(0.0)
25
26          if i+1 in ns:
27              if pol:
28                  print("Полином {0} итерации: ".format(i+1), y)
29              us.append(round(y.get(x), 7))
30      return us
31
32
33  def broken_evident(x, func):
34      y = 0
35      t = 0
36
37      while t <= x:
38          f = func(t, y) * h
39          y += f
40          t += h
41
42      return y
43
44
45  def runge_kutta_second(x, alpha, h, func):
46
47      # assert(alpha == 0.5 or alpha == 1, "Alpha должна быть равно 1 или
    0.5")
48
49      y = 0
50      t = 0
51
52      while t <= x:
53          f = h * ((1 - alpha) * func(t, y) +
54          alpha * func(t + h / (2 * alpha),
```

```
55          y + (h / (2 * alpha)) * func(t, y)))
56
57          y += f
58          t += h
59
60      return y
61
62
63  def runge_kutta_fourth(x, h, func):
64      y = 0
65      t = 0
66
67      while t <= x:
68          k1 = func(t, y)
69          k2 = func(t + h / 2, y + h / 2 * k1)
70          k3 = func(t + h / 2, y + h / 2 * k2)
71          k4 = func(t + h, y + h * k3)
72
73          f = h / 6 * (k1 + 2 * k2 + 2 * k3 + k4)
74
75          y += f
76          t += h
77
78      return y
79
80
81  def calc(ns, a, b, h, alpha, func):
82      table_picard = PrettyTable()
83      title = ["X"]
84      for it in ns:
85      title.append("Пикар: я{0}- итерация".format(it))
86      title.append("Метод ломанной явный()")
87      title.append("Метод РунгеКутты- ого2- порядка неявный()")
88      title.append("Метод РунгеКутты- ого4- порядка")
89
90      table_picard.field_names = title
91
92      xs = np.arange(a, b, h)
93
94      for x in xs:
95      u = picard(ns, x, func)
96      u.insert(0, round(x, 3))
97      bevi = broken_evident(x, func)
98      u.append(bevi)
```

```
99          rks = runge_kutta_second(x, alpha, h, func)
100         u.append(rks)
101         rkf = runge_kutta_fourth(x, h, func)
102         u.append(rkf)
103         table_picard.add_row(u)
104
105         u = picard(ns, b, func, pol=True)
106         u.insert(0, round(b, 3))
107         bevi = broken_evident(b, func)
108         u.append(bevi)
109         rk = runge_kutta_second(b, alpha, h, func)
110         u.append(rk)
111         rkf = runge_kutta_fourth(b, h, func)
112         u.append(rkf)
113         table_picard.add_row(u)
114
115         print(table_picard)
116
117
118         if __name__ == "__main__":
119         x1 = int(input("Введите X: от "))
120         x2 = int(input(" " * 11 + "до "))
121         h = float(input(" " * 6 + "с шагом "))
122         nl = input("Введите через пробел интересующие итерации для метода
        Пикара: ").split(" ")
123         a = float(input("Введите α для метода РунгеКутты- второго порядка: "
        ))
124         nl = list(map(lambda x: int(x), nl))
125         calc(nl, x1, x2, h, a, test_func)
126
```

4. Результат работы

В данном разделе представлен результат работы программы на рисунках 1 2

X	Пикар: 1-я итерация	Пикар: 2-я итерация	Пикар: 3-я итерация	Пикар: 4-я итерация	Метод ломанной (явный)	Метод Рунге-Кутты 2-ого порядка (неявный)	Метод Рунге-Кутты 4-ого порядка
0.0	0.0	0.0	0.0	0.0	0.0	2.5000000000000004e-07	3.3333333348958334e-07
0.01	3e-07	3e-07	3e-07	3e-07	1.0000000000000002e-06	2.5000000056250003e-06	2.6666666875347228e-06
0.02	2.7e-06	2.7e-06	2.7e-06	2.7e-06	5.000000100000001e-06	8.750000208125005e-06	9.000000349635436e-06
0.03	9e-06	9e-06	9e-06	9e-06	1.4000000260000002e-05	2.1000001963750162e-05	2.133333594062541e-05
0.04	2.13e-05	2.13e-05	2.13e-05	2.13e-05	3.000000222000074e-05	4.1250010373752586-05	4.1666679081341546e-05
0.05	4.17e-05	4.17e-05	4.17e-05	4.17e-05	5.500001122000140de-05	7.150003926439788e-05	7.200004445930588e-05
0.06	7.2e-05	7.2e-05	7.2e-05	7.2e-05	5.500001122000140de-05	7.150003926439788e-05	7.200004445930588e-05
0.07	0.0001143	0.0001143	0.0001143	0.0001143	0.0001400001242800892	0.0001700003104988788	0.0001706669996109663
0.08	0.0001707	0.0001707	0.0001707	0.0001707	0.0002040003202804372	0.0002422507185399861	0.0002430007592944276
0.09	0.000243	0.000243	0.000243	0.000243	0.0002850007364417439	0.0003325015180213375	0.0003333349207695215
0.1	0.0003333	0.0003333	0.0003333	0.0003333	0.0003850015486595416	0.0004427529810996755	0.0004436697600638993
0.11	0.0004437	0.0004437	0.0004437	0.0004437	0.0005060030309578666	0.0005750055137451706	0.0005760056878800406
0.12	0.000576	0.000576	0.000576	0.000576	0.0006500055913485	0.0007312596999279108	0.0007323493293894766
0.13	0.0007323	0.0007323	0.0007323	0.0007323	0.0008190098164212278	0.0009135163546104131	0.0009146833997123825
0.14	0.0009147	0.0009147	0.0009147	0.0009147	0.0010150165241920217	0.001123776586348182	0.001125027128054113
0.15	0.001125	0.001125	0.001125	0.001125	0.0012400268267774655	0.001364041870302288	0.001365375943617648
0.16	0.0013653	0.0013654	0.0013654	0.0013654	0.0014960422033442777	0.001636311132469327	0.001637731803789208
0.17	0.0016377	0.0016377	0.0016377	0.0016377	0.0017850645848655217	0.001942595845938502	0.001944097118473121
0.18	0.001944	0.0019441	0.0019441	0.0019441	0.0017850645848655217	0.001942595845938502	0.001944097118473121
0.19	0.0022863	0.0022865	0.0022865	0.0022865	0.0021090964494212433	0.002284890139966948	0.002286475229846214
0.2	0.0026667	0.0026669	0.0026669	0.0026669	0.002470140932299573	0.002665200922739389	0.002666869862029857
0.21	0.003087	0.0030873	0.0030873	0.0030873	0.002870201948216273	0.003085533018539157	0.003087285921969343
0.22	0.0035493	0.0035497	0.0035497	0.0035497	0.003311284328854658	0.003547892320228687	0.003549729232089596
0.23	0.0040557	0.0040562	0.0040562	0.0040562	0.003795339374893113	0.004054285857837722	0.004056207207945016
0.24	0.004608	0.0046087	0.0046087	0.0046087	0.004324538025047378	0.004606722484057673	0.004608728159595858
0.25	0.0052083	0.0052093	0.0052093	0.0052093	0.004900725041338868	0.005207212077527273	0.005209302376838729
0.26	0.0058587	0.0058599	0.0058599	0.0058599	0.005525965212397988	0.005857766764708376	0.005859941912710343
0.27	0.006561	0.0065627	0.0065627	0.0065627	0.006202270575313275	0.006560400661226499	0.006562660911121384
0.28	0.0073173	0.0073195	0.0073195	0.0073195	0.006931655256916169	0.007317130233522622	0.007319475869106227
0.29	0.0081297	0.0081324	0.0081324	0.0081324	0.007716135735362176	0.00812997458616879	0.008132405919756867
0.3	0.009	0.0090035	0.0090035	0.0090035	0.008557731122869042	0.009009855744360347	0.009003473137220844
0.31	0.0099303	0.0099347	0.0099347	0.0099347	0.009458643470488756	0.009932099026575809	0.009934702864655755
0.32	0.0109227	0.0109281	0.0109281	0.0109281	0.010420358095800983	0.010925433351479964	0.010928124066047203
0.33	0.011979	0.0119858	0.0119858	0.0119858	0.011445443934429431	0.011982991636823153	0.011985769702813
0.34	0.0131013	0.0131097	0.0131097	0.0131097	0.012535753916297993	0.013106811197180285	0.013109677136134066
0.35	0.0142917	0.0143019	0.0143019	0.0143019	0.013693325367560493	0.014298934172852008	0.014301888555972028
0.36	0.015552	0.0155644	0.0155645	0.0155645	0.014920200439156713	0.015561407986431838	0.015564451437755082
0.37	0.0168843	0.0168994	0.0168994	0.0168994	0.016218462656296816	0.016896285828042045	0.016899419027737363
0.38	0.0182907	0.0183088	0.0183089	0.0183089	0.017590056936569943	0.018305627170269876	0.018308850858306314
0.39	0.019773	0.0197948	0.0197948	0.0197948	0.019037151037600262	0.019791498313863405	0.019794813292595616
0.4	0.0213333	0.0213593	0.0213594	0.0213594	0.020561775168796547	0.021355972965277466	0.021359380104601229
0.41	0.0229737	0.0230046	0.0230046	0.0230046	0.022166003034777747	0.023001132847194107	0.023004633087414787
0.42	0.024696	0.0247327	0.0247327	0.0247327	0.023851916351682885	0.024729068343317891	0.024732662269924603
0.43	0.0265023	0.0265455	0.0265456	0.0265456	0.025621605490819333	0.026541879177474713	0.026545568743331874
0.44	0.0283947	0.0284453	0.0284455	0.0284455	0.027477170157498603	0.028441675132577827	0.028445461084677826
0.45	0.030375	0.0304343	0.0304345	0.0304345	0.029420720106297246	0.030430576801689303	0.030434460404682905
0.46	0.0324453	0.0325145	0.0325147	0.0325147	0.031454375894012988	0.032510716384838279	0.032514698994991646
0.47	0.0346077	0.0346881	0.0346883	0.0346883	0.033580269676146418	0.034684238519893364	0.034688321591972672
0.48	0.036864	0.0369572	0.0369575	0.0369575	0.035800546016754003	0.036953330116367598	0.036957486253282934
0.49	0.0392163	0.039324	0.0393244	0.0393244	0.038117362807704986	0.039320076507365846	0.039324365278040093
0.5	0.0416667	0.0417907	0.0417911	0.0417911	0.040532892141179125	0.041786751943926680	0.041791146177220184
0.51	0.044217	0.0443594	0.04436	0.04436	0.043049321294632421	0.044355553107965547	0.044360032660801935

Рис. 1.

0.52	0.0468693	0.0470325	0.0470332	0.0470332	0.045666853735271 7	0.047028634794790 15	0.047033245748820 13
0.53	0.0496257	0.0498121	0.049813	0.049813	0.048393710177286636	0.049808302354553184	0.049813024832490 77
0.54	0.052488	0.0527005	0.0527016	0.0527016	0.051226129689133 87	0.052696792572556 98	0.052701628862969 46
0.55	0.0554883	0.0557	0.0557013	0.0557013	0.054168370852763 15	0.055696385028596286	0.055701337564554 51
0.56	0.0585387	0.0588128	0.0588144	0.0588145	0.057222712976771 58	0.058809381342959 78	0.058814452709069 45
0.57	0.061731	0.0620413	0.0620433	0.0620433	0.060391457365575 81	0.062038106509507474	0.062043299449534656
0.58	0.0650373	0.0653878	0.0653902	0.0653902	0.063676928646803 19	0.065384910289983 58	0.065390022771461 935
0.59	0.0684597	0.0688547	0.0688576	0.0688576	0.067081476159222 1	0.068852168671366 1	0.068857613666904 78
0.6	0.072	0.0724443	0.0724478	0.0724479	0.07060747540365 91	0.072442285390995 62	0.072447861227426 97
0.61	0.0756603	0.0761592	0.0761634	0.0761634	0.074257329559487 88	0.076157693528776 94	0.076163406669319 6
0.62	0.0794427	0.0800017	0.0800067	0.0800067	0.078033471069420 95	0.080000857172909 51	0.080006705283515 07
0.63	0.083349	0.0839742	0.0839802	0.0839803	0.081938363295492 38	0.083974273160175 83	0.083980263111963 502
0.64	0.0873813	0.0880794	0.0880866	0.0880866	0.085974502249287 82	0.088080472894803 11	0.088086608085388 24
0.65	0.0915417	0.0923282	0.0923282	0.0923283	0.090144418399657 95	0.092322024249312 71	0.092328310497992 61
0.66	0.095832	0.0966979	0.0967079	0.096708	0.094450678561344 08	0.096701533551084 96	0.096707974621351 92
0.67	0.1002543	0.1012164	0.1012281	0.1012282	0.098895887868151 07	0.101221647658595 77	0.101228248442946 38
0.68	0.1048107	0.1058778	0.1058917	0.1058918	0.103482691834523 37	0.105885056131528 82	0.105891821744641 37
0.69	0.109503	0.110685	0.1107013	0.1107014	0.108213778509615 67	0.11069449349920 59	0.110701429341880 93
0.7	0.1143333	0.1156405	0.1156596	0.1156599	0.113091880728209 86	0.115652741632142 66	0.115659853406014 38
0.71	0.1193037	0.1207473	0.1207697	0.1207699	0.118119778463076 31	0.120762632221670 41	0.120769925944805 57
0.72	0.124416	0.1260082	0.1260345	0.1260345	0.123303012837175 8	0.126027049373097 92	0.126034531396497 9
0.73	0.1296723	0.1314259	0.1314562	0.1314566	0.128363639326685 6	0.131448932180649	0.131456609343155 03
0.74	0.1350747	0.1370034	0.1370387	0.1370392	0.134130803983027 35	0.137031278252195 55	0.137039157349369 03
0.75	0.140625	0.1427846	0.1427842	0.1427852	0.13978671470879 87	0.142777145304533 92	0.142785233932826 94
0.76	0.1463253	0.14865	0.1486972	0.148698	0.145607117964888 48	0.148689655645675 3	0.148697961673655 68
0.77	0.1521777	0.1547251	0.1547797	0.1547805	0.151595132292909 9	0.154771998741966 76	0.154780530469924 85
0.78	0.158184	0.1609722	0.1610351	0.1610362	0.157753943134258 97	0.161027434763643 98	0.161036200947182 05
0.79	0.1643443	0.1673946	0.167467	0.1674683	0.164086806200003 04	0.167459299155302 72	0.167468308030426 23
0.8	0.1706667	0.1739955	0.1740787	0.1740802	0.170597050999692 22	0.174071001377673 76	0.174080264687496 88
0.81	0.177147	0.1807782	0.1808737	0.1808755	0.177288084537790 13	0.180866038302843 2	0.180875568583470 95
0.82	0.1837893	0.1877462	0.1878555	0.1878578	0.184163395186980 94	0.18784799096525 05	0.187887792546321 7
0.83	0.1905957	0.194903	0.1950279	0.1950306	0.191226556748248 9	0.195020528603156 2	0.195030616184806 66
0.84	0.197568	0.202252	0.2023945	0.2023977	0.198481232708306 8	0.202387417462741 91	0.202397803120319 13
0.85	0.2047083	0.2097969	0.2099593	0.2099631	0.205931180705680 92	0.209952522916953 82	0.209963219395266 88
0.86	0.2120187	0.2175413	0.2177262	0.2177307	0.213582357217549 28	0.217572038587947 92	0.217730835741443 95
0.87	0.219501	0.2254892	0.2256992	0.2257046	0.221432422480280 44	0.225693373601381 53	0.225704732832746 56
0.88	0.2271573	0.2336442	0.2338825	0.233889	0.22949174565753 53	0.233877394097657 98	0.233889106807905 03
0.89	0.2349897	0.2420105	0.2422804	0.2422881	0.237762410270784 74	0.242276193046325 3	0.242288275079469 82
0.9	0.243	0.250592	0.2508974	0.2509065	0.246248719908162 47	0.25089421435178 22	0.250906682447206 6
0.91	0.2511903	0.2593929	0.2597379	0.2597486	0.25495510422872 66	0.259736035687170 74	0.259748907534813 17
0.92	0.2595627	0.2684174	0.2688066	0.2688193	0.263886125280449 4	0.268806375271053 44	0.26881966957057 77
0.93	0.268119	0.2776698	0.2781085	0.2781234	0.273096484516947	0.278112439909791 06	0.278123835534063 17
0.94	0.2768613	0.2871546	0.2876483	0.2876659	0.282441027976680 25	0.287652228029366 47	0.28766642769259 14
0.95	0.2857917	0.2968764	0.2974314	0.297452	0.29207475731952 55	0.297437946622186 7	0.29745263155311 75
0.96	0.294912	0.3068397	0.3074628	0.307487	0.30195283395815 81	0.307472610636479 5	0.30748780425706 45
0.97	0.3042243	0.3170495	0.3177482	0.3177766	0.31208058909751 74	0.317761756337821 2	0.31777748344783 39
0.98	0.3137307	0.3275104	0.3282931	0.3283263	0.32246353203842626	0.328311105774946 1	0.32832739664304 67
0.99	0.323433	0.3382277	0.3391035	0.3391421	0.33310735933337327	0.339126596685023	0.33914347114611 23
1.0	0.3333333	0.3492063	0.3501851	0.3502302	0.34401796446179383	0.350214352970697 9	0.35023184453449 48
1.01	0.3434337	0.3604517	0.3615445	0.3615968	0.35520144806051823	0.361588757650689	0.36159887576506 89
1.02	0.353736	0.3719691	0.3731878	0.3732487	0.36666412874756116	0.373232335378077 93	0.37325115694025 63
1.03	0.3642423	0.3837641	0.3851219	0.3851925	0.37841255458066325	0.385175987371510 7	0.38519552578223 665
1.04	0.3749547	0.3958425	0.3973537	0.3974355	0.39045351519530 59	0.397418786174047 5	0.39743907886466 23
1.05	0.385875	0.4082099	0.4098902	0.4099848	0.40279405467058965	0.409968098214383	0.40998918567004 91
1.06	0.3970053	0.4208725	0.4227389	0.4228483	0.41544148517536 94	0.422831580406502 87	0.42283530495225 94

Рис. 2.

1.07	0.4083477	0.4338363	0.4359075	0.4360337	0.4280340141514166	0.4360171889292006	0.4360399933334228
1.08	0.419904	0.4471076	0.4494039	0.4495494	0.44168769619516807	0.4495332031570662	0.4495569367409818
1.09	0.4316763	0.4606928	0.4632364	0.4634039	0.45530257640487004	0.463388240073233	0.4634129538039353
1.1	0.4436667	0.4745987	0.4774135	0.4776062	0.46925658076567917	0.4775912740961146	0.4776170222760452
1.11	0.455877	0.4888319	0.4919443	0.4921655	0.48355859151159813	0.4921516574623329	0.4921784979818796
1.12	0.4683093	0.5033995	0.5068378	0.5070917	0.4982178873300615	0.5070791419728095	0.5071071365885224
1.13	0.4809657	0.5183086	0.5221038	0.5223947	0.5132440979626173	0.5223839022110527	0.5224131168306761
1.14	0.493848	0.5335666	0.5377521	0.5380851	0.528647293003552	0.538076560352808	0.5381070653820722
1.15	0.5069583	0.5491809	0.5537932	0.554174	0.5444379726075523	0.5541682126973958	0.5542000083381277
1.16	0.5202987	0.5651593	0.5702377	0.5706727	0.5606270996777226	0.5706704580634	0.5707037774877399
1.17	0.533871	0.5815098	0.5870971	0.5875935	0.5772261271165411	0.5875954282052519	0.5876302772789227
1.18	0.5476773	0.5982404	0.6043828	0.6049487	0.5942470271348007	0.6049558204217961	0.6049922940021345
1.19	0.5617197	0.6153596	0.622107	0.6227515	0.6117023224273862	0.6227649325460812	0.6228031295045213
1.2	0.576	0.6328759	0.6402824	0.6410157	0.6296051197400168	0.6410367005227662	0.6410767269108402
1.21	0.5905203	0.6507981	0.6589221	0.6597557	0.6479691458080452	0.6597857388016988	0.6598277084605002
1.22	0.6052827	0.6691352	0.6780398	0.6789864	0.6668087859472374	0.6790273837988582	0.6790714190282567
1.23	0.620289	0.6878966	0.6976498	0.6987238	0.6861391255174016	0.6987777407020204	0.698823972996285
1.24	0.6355413	0.7070916	0.7177669	0.7189844	0.7059759945130595	0.7190537339275258	0.7191023047848482
1.25	0.6510417	0.7267301	0.7384067	0.7397855	0.7263360155613465	0.7398731615670394	0.7399242233814008
1.26	0.666792	0.7468221	0.7595852	0.7611455	0.747236655636619	0.7612547541596333	0.7613084712445435
1.27	0.6827943	0.7673778	0.7813193	0.7830835	0.7666962818316281	0.7832182384854344	0.7832747880003128
1.28	0.6990507	0.7884078	0.8036265	0.8056196	0.7907342215686458	0.8057844060030731	0.8058439793944641
1.29	0.715563	0.8099229	0.8265252	0.828775	0.8133708276602436	0.8289751878449599	0.8290379920164064
1.3	0.7323333	0.8319342	0.8500345	0.8525721	0.836627548693130	0.8528137355428295	0.8528799943691043
1.31	0.7493637	0.854453	0.8741743	0.8770342	0.8605270052454536	0.8773245089619784	0.8773944649255323
1.32	0.766656	0.877491	0.8989654	0.9021861	0.8850930725130207	0.9025333718772903	0.9026072878872664
1.33	0.7842123	0.9010602	0.9244295	0.9280539	0.9103509699831261	0.9284676960287773	0.9285458574458243
1.34	0.8020347	0.9251729	0.9505893	0.9546649	0.9363273558566184	0.9551564745504781	0.9552391914439274
1.35	0.820125	0.9498417	0.9774685	0.9820482	0.9630504480982772	0.9826304457758809	0.9827180554437268
1.36	0.8384853	0.9750794	1.0050917	1.0102343	0.9905501097541002	1.0109222285476513	1.0110150983342747
1.37	0.8571177	1.0008993	1.0334849	1.0392555	1.0188580049534388	1.0400664703017408	1.0401650007535654
1.38	0.876024	1.027315	1.0626749	1.0691458	1.048007721296016	1.0701000093587654	1.0702046377641672
1.39	0.8952063	1.0543404	1.09269	1.0999414	1.0780349231349766	1.101062053042224	1.1011732574091901
1.4	0.9146667	1.0818897	1.1235596	1.1316803	1.1089775160899963	1.132994373457613	1.133112676691074
1.41	0.934407	1.1102776	1.1553145	1.1644029	1.1408758274018937	1.1659415230134724	1.166067459161414
1.42	0.9544293	1.1392191	1.1879869	1.198152	1.1737728039373934	1.199951072050434	1.2000853502186664
1.43	0.9747357	1.1688295	1.2216104	1.2329727	1.2077142298900239	1.2350738712740499	1.2352171432660919
1.44	0.995328	1.1991246	1.2562201	1.2689131	1.2427489665008125	1.271364342069447	1.271517369419827
1.45	1.0162083	1.2301205	1.2918529	1.3060239	1.278929216438201	1.3088807982206307	1.3090444205136
1.46	1.0373787	1.2618337	1.3285471	1.3443593	1.3163108158447931	1.3476858030738204	1.3478609474213283
1.47	1.058841	1.2942813	1.3663431	1.3839764	1.3549535574838931	1.3878456667820128	1.3880342586507253
1.48	1.0805573	1.3274805	1.4052828	1.424936	1.3945215489132758	1.4294538905367722	1.4296367646004903
1.49	1.1026497	1.3614491	1.4454102	1.467303	1.436283610189502	1.4725301533842394	1.472746473706434
1.5	1.125	1.3962054	1.4867713	1.5111459	1.479113716278492	1.5172148802896388	1.5174475476974453

Рис. 3.

5. Ответы на вопросы

1) Укажите интервалы значений аргумента, в которых можно считать решением заданного уравнения каждое из первых 4-х приближений Пикара. Точность результата оценивать до второй цифры после запятой. Объяснить свой ответ.

   а) 1ое приближение: [0 - 0.86] Значение функции совпадает с приближением на данном интервале до 2ух знаков.

   б) 2ое приближение: [0 - 0.18] Значение функции совпадает с приближением на данном интервале до 2ух знаков.

   в) 3ье приближение: [0 - 1.36] Значение функции совпадает с приближением на данном интервале до 2ух знаков.

   г) 4ое приближение: [0 - 1.51] Значение функции совпадает с приближением на данном интервале до 2ух знаков.

2) Пояснить, каким образом можно доказать правильность полученного результата при фиксированном значении аргумента в численных методах.

   а) Численные методы зависят от шага, по этому мы сравниваем значение полученные Эйлером при разных его величинах. На пример: берем шаг равный 0.00001 получаем (для примера) 200, а при 0.0000001 получаем 300. Видим, что слишком большая разница которая нас не устраивает, продолжаем уменьшать шаг. И так до тех пора пока не получаем устраивающую нас точность. Правда есть вероятно того, что мы выйдем за разрядность и будет происходить округление.

3) Из каких соображений выбирался корень уравнения в неявном методе?

1.997	2.6546847	94.17541680571122	159.9532568733473	162.66176119606615
1.998	2.6586747	103.04841394524065	189.7990399802563	194.26385243431724
1.999	2.6626687	113.67138156587032	232.98922448745535	241.10121320580254
2	2.6666667	142.62724969805748	420.28806125158275	465.4404228687651

Рис. 4.

a) Выбирается наименьший из двух корней для сохранения непрерывности функции

4) Каково значение функции при x=2, т.е. привести значение u(2).

a) Около 465

Список литературы

Список литературы

[1] Градов В.М. Курс лекций по Моделированию - 2020

[2] Калитикин Численнные методы URL https://studfile.net/preview/878067/