Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение

высшего образования

**Пермский национальный исследовательский**

**политехнический университет**

Факультет прикладной математики и механики

Кафедра «Математическое моделирование систем и процессов»

Направление 01.03.02 «Прикладная математика и информатика»

Профиль программы бакалавриата «Математическое моделирование»

**КУРСОВАЯ РАБОТА**

по дисциплине **«ЧИСЛЕННЫЕ МЕТОДЫ 2»**

на тему

**«Способы Рунге-Кутты построения одношаговых методов для решения ОДУ»**

Выполнил студент группы ММ-19-2б

Мельников Демид Леонидович

(Фамилия, Имя, Отчество полностью)

Проверил:

канд. Ф.-м. наук, доц. кафедры ММСП

\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (Няшина Н.Д.)

(подпись) (ФИО)

Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

«\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2021г.

**Пермь 2021**

Содержание

[**Введение** 3](#_Toc93148050)

[**1.Общие понятия** 4](#_Toc93148051)

[**2.Общий вид семейства методов Рунге-Кутты** 7](#_Toc93148052)

[**3.Определение коэффициентов таблицы Батчера** 10](#_Toc93148053)

[**4.Численное решение ОДУ различными одношаговыми методами Рунге-Кутты** 13](#_Toc93148054)

[**Заключение** 17](#_Toc93148055)

[**Список использованных источников** 18](#_Toc93148056)

[**Приложения** 19](#_Toc93148057)

# **Введение**

При исследовании различных явлений и аспектов окружающего мира, так или иначе необходимо будет столкнуться с функциональной зависимостью между величинами, в которых присутствуют производные от исходной функции, другими словами, столкнуться с дифференциальными уравнениями или с их системами.

Для решения таких уравнений и их систем существует множество различных способов. В этой работе будут рассмотрены явные одношаговые способы Рунге-Кутты.

К классу методов Рунге-Кутты относятся явный метод Эйлера первого порядка точности (и него модификация второго порядка точности). Наиболее часто применяется классический метод Рунге-Кутты, имеющий четвёртый порядок точности.

Суть работы состоит в том, чтобы изучить то, как задаются различные методы Рунге-Кутты и их общий вид, решить с их помощью ОДУ, получить графики интегральной кривой, проверить уравнение на устойчивость, а также точность методов.

# **1.Общие понятия**

Обыкновенными дифференциальными уравнениями называют уравнения, которые содержат независимую переменную, искомую функцию и одну или несколько производных от неё. Они принимают вид:

 (1.0)

Здесь *x* – независимая переменная;  - n-ая производная неизвестной функции *y(x)*; F – какая-то функция от переменных . Порядок дифференциального уравнения буде определяться наивысшим порядком *n* производной входящей в уравнение.

Решением дифференциального уравнения называется функция , которая будет удовлетворять данному уравнению (то есть уравнение, после подстановки в него эту функцию, уравнение превращается в тождество). Например:

 (1.1)

Общее решение ОДУ n-ого порядка содержит n произвольных констант  и имеет вид:

 (1.2)

Частное решение будет получатся из общего, если задать определённые значения этим произвольным постоянным. Так, например для уравнения первого порядка частное решение будет получаться из общего решения путём задания произвольной константы  для некоторого значения :

 (1.3)

Для того, чтобы найти частное решение так же необходимо учесть дополнительные условия при определённых *х*. В зависимости от способа задания таких условий выделяют два типа задачи: задача Коши и краевые задачи. Если дополнительные условия задаются в одной точке *х*, то задача называется задачей Коши, а дополнительные условия – начальными условиями. В этой работе будет рассматриваться только решение задачи Коши.[1]

Перед тем, как решать задачу Коши, необходимо проверить её устойчивость, для этого наряду с уравнением  необходимо рассмотреть задачу:

 (1.4)

где - возмущения, вносимые в правую часть уравнения и начальные условия.

Задача Коши будет абсолютно устойчивой, если 

Или же по-другому, решение устойчиво, если возмущённое решение не слишком сильно отличается от исходного.

Для выделения частного решения из общего для уравнений высшего порядка нужно будет задать столько дополнительных условий, сколько произвольных постоянных содержится в общем решении. Численное решение ОДУ выше первого порядка будет сводится к решению системы ОДУ первого порядка. Поэтому в данной работе будут разобраны только ОДУ первого порядка, так как дифференциальное уравнение n-ого порядка можно будет свести к системе из n уравнений первого порядка. [2]

Численное решение уравнения – приближённое определение корней этого уравнений. Применяется, когда точный метод решения неизвестен или трудоёмок. Существует множество численных методов решения обыкновенных дифференциальных уравнений, в данной работе будут рассматриваться явные одношаговые методы Рунге-Кутты для решения ОДУ.

Одношаговые методы решения – это методы, которые дают приближение для  к значениям точного решения  в каждом узле сетки на основе приближения  к решению в точке  . В общем виде их можно представить:

 (1.5)

Такой метод называется неявным. Рассматриваемые нами одношаговые методы не имеют в своей правой части искомое значение . Отсутствие искомого значения в правой части представления метода в виде такой зависимости от искомой функции делает его явным вида:

 (1.6)

Теперь, когда мы определились с определяющими понятиями перейдём к определению семейства методов Рунге-Кутты.

# 

# **2.Общий вид семейства методов Рунге-Кутты**

Пусть нам задано обыкновенное дифференциальное уравнение вида:

 (2.0)

тогда согласно теореме о существования и единственности для любой точки  найдётся такое решение , определённое на некотором интервале, удовлетворяющее условию , такое, что это решение единственное. Тогда мы сможем задать задачу коши для уравнения (2.0) с начальным условием , то есть задача будет состоять в нахождении функции *y(x)*, которое будет обращать изначальное уравнение в тождество [2]

Запишем задачу Коши:

 (2.1)

где независимая переменная *х* принадлежит интервалу.Разобьём этот интервал на N частей, так что . Назовём точки узлами сетки, а множество таких точек – самой сеткой, где *h* – шаг по этой сетке. Если в каждой такой точке , принадлежащей сетке поставлено в соответствие некоторое число , то множество таких значений будем называть сеточной функцией, определённой на этой сетке.

Приведённой способ построения сеточной функции называется методом Рунге-Кутты.

Предположим, что величина сеточной функции в точке 

Ровняется начальному условию из (2.1), а её значения в следующих узлах сетки  будем находить последовательно по формуле:

 (2.2)

где

 (2.3)

Теперь переопределим величины:

 (2.4)

где  - константы, про определение которых будет сказано позже.­­­

Предполагается, что  уже известно и имеет конкретное значение. При нахождении  в качестве  берётся  и т.д.

Величины *K* можно рассматривать как приближённые значения функции , умноженные на *h*, если опять же соответствующим образом подобрать параметры .

Отметим, что соотношение (2.2) является определяющим для семейств методов Рунге-Кутты. Все методы, которые так или иначе входят в это семейство определяются этой формулой и различаются коэффициентами . В зависимости от определения всех констант мы будем получать различные методы. Эти методы будут отличатся скоростью подсчёта, а также точностью(сходимостью).[3]

Определим порядок точности методов Рунге-Кутты, для этого рассмотрим функцию из соображений, что эта функция достаточно гладкая, мы можем разложить её в ряд Маклaрена:

 (2.5)

то есть, если параметры подобраны так, что справедливо:

 (2.6)

Тогда разложение функции примет вид остаточного члена

 (2.7)

То погрешность численного решения будет величиной порядка не ниже, чем , a *s* – порядок точности метода.

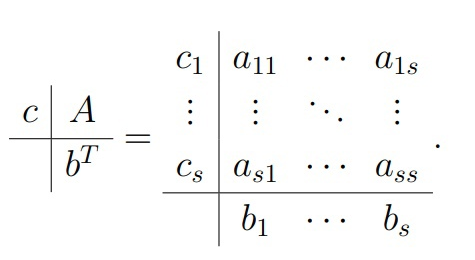
Стоит отметить, что для сходимости явного метода Рунге-Кутты будет необходимо и достаточно, чтобы задача удовлетворяла условиям гладкости и ограниченности, тогда найденное решение будет стремится к точному при  и справедлива оценка:[4]

 (2.8)

Теперь, когда нам известен вид разложения функции (2.7) мы можем определить коэффициенты для различных случаев.

# **3.Определение коэффициентов таблицы Батчера**

Зафиксируем некоторое число этапов(стадий) sметода и зададим матрицу , векторы *b* и *c* размерности *s*, эти коэффициенты, определяющий конкретный метод могут быть представлены в виде таблица Батчера.

 (3.0)

Для вычисления численного решения необходимо сначала найти *s* уже знакомых нам вспомогательных величин  на сетке:

 (3.1)

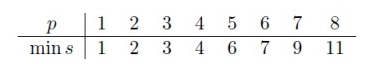
После чего искомое нами значение вычисляется как:

 (3.2)

Из (3.1) и (3.2) получаем:

 (3.3)

Теперь выведем коэффициенты для случая *s*=2, (число стадий) *h*=2. Для многих случаев, коэффициенты, либо =0, либо переопределены, либо вычисления слишком затратны. Ниже приведена таблица зависимости точности, от числа стадий вычисления



Как из неё видно, методы до 4 порядка самые малозатратные для вычисления, их мы и разберём на практике.

Случай *s*=2, *h*=2:

(3.4)

Введём обозначения:



Получим следующие выражения для производных погрешности:



 (3.5)







Подставляя (3.5) в следующие равенства:

 (3.6)

Получим по (2.7):

 (3.7)

Получаем систему уравнений относительно наших коэффициентов:

 (3.8)

Таким образом получаем три алгебраических уравнения и четыре параметра. Система переопределена. Один из способов решения таких систем – метод подбора.

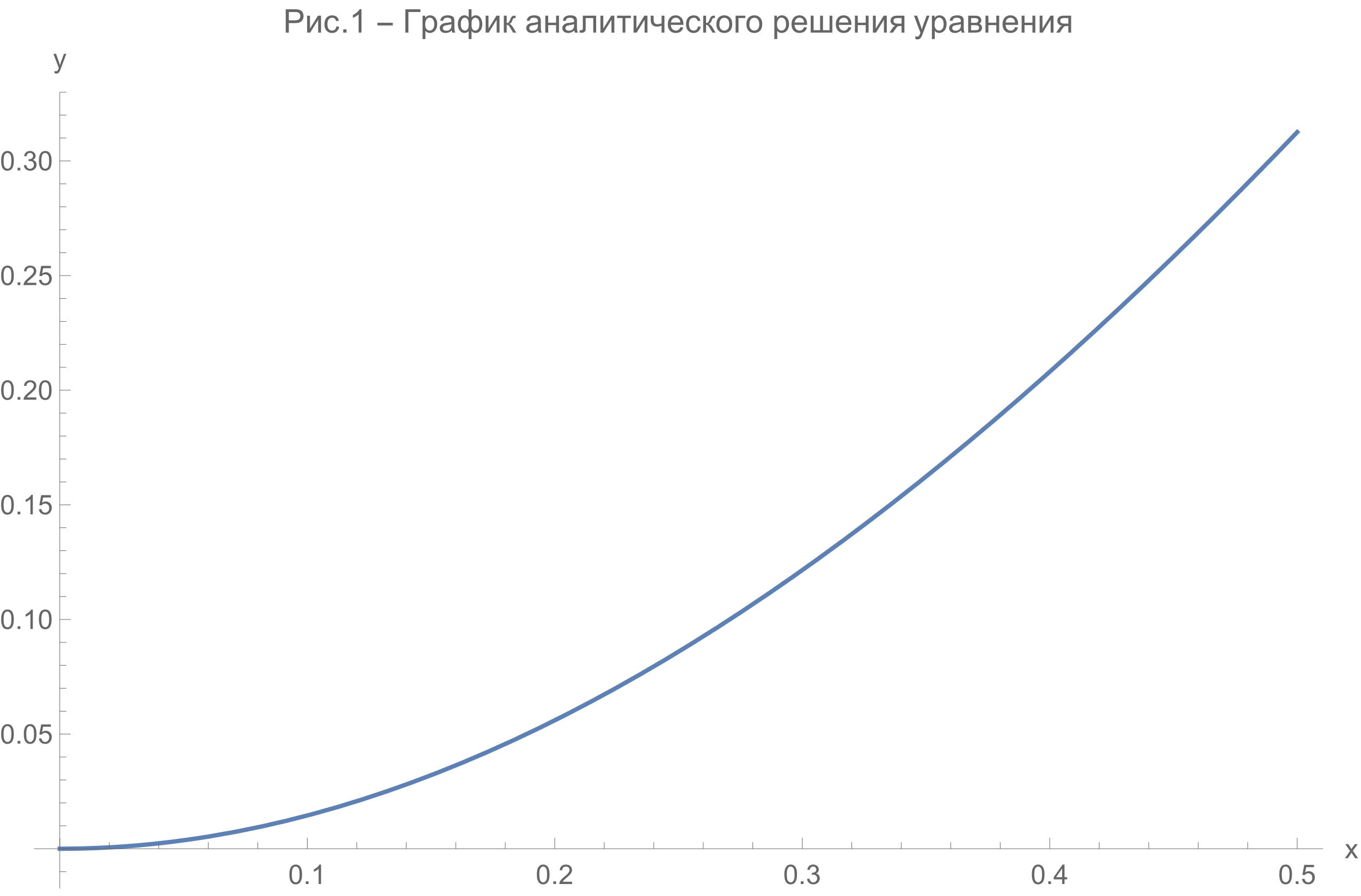
Система (3.8) будет определять семейство методов Рунге-Кутты со 2-ым порядком аппроксимации (т.к наш случай s=2 b h=2).Задавая один из параметров можно получать различные методы.[5]

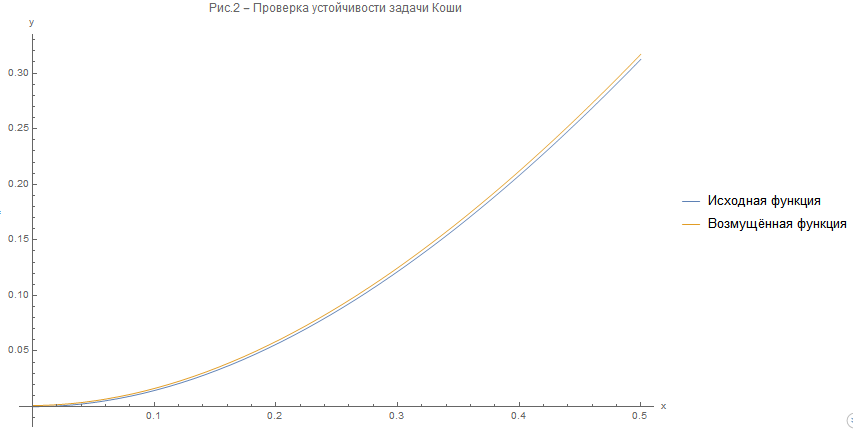
# **4.Численное решение ОДУ различными одношаговыми методами Рунге-Кутты**

Была рассмотрена задача Коши:

 (4.1)

Для начала проверим устойчивость самой задачи Коши, для этого построим график аналитического решения и график возмущённого решения и сверим их. Построение графиков и анализ полученных данных производился в мат. пакете Wolfram Mathematica 12.1.





Как мы видим из Рис.2 различие между решениями уменьшается с увеличением значения аргумента, а значит решение исходной и возмущённой функций сходится, и задача Коши является устойчивой.

Теперь решим численно уравнение (4.1) с помощью методов Рунге-Кутты различного порядка. Программный код был реализован на языке программирования С++. С кодом можно будет ознакомиться в приложении. Начнём с метода 1-ого порядка точности. Для всех приведённых ниже методов возьмём шаг равный 0,025.

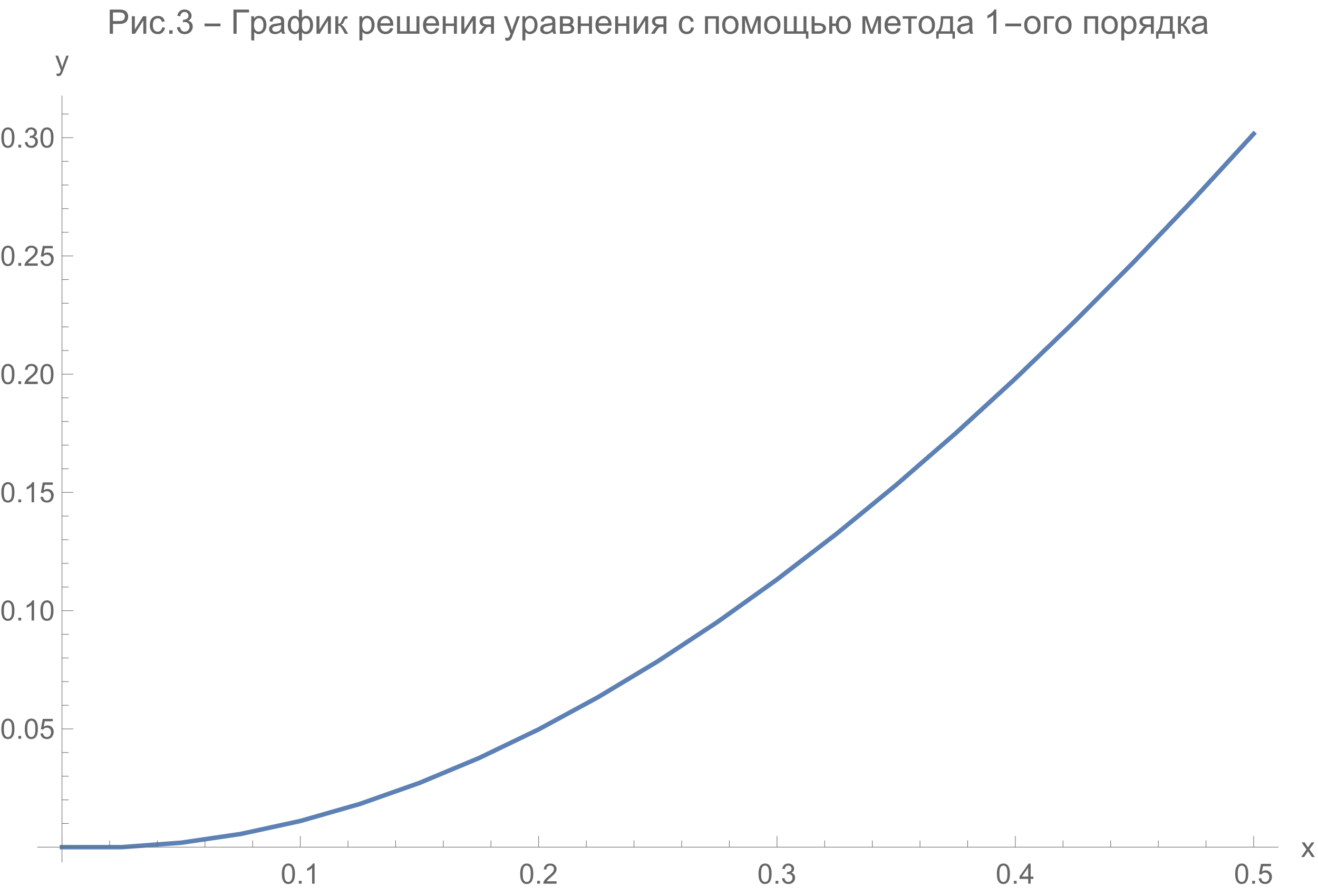
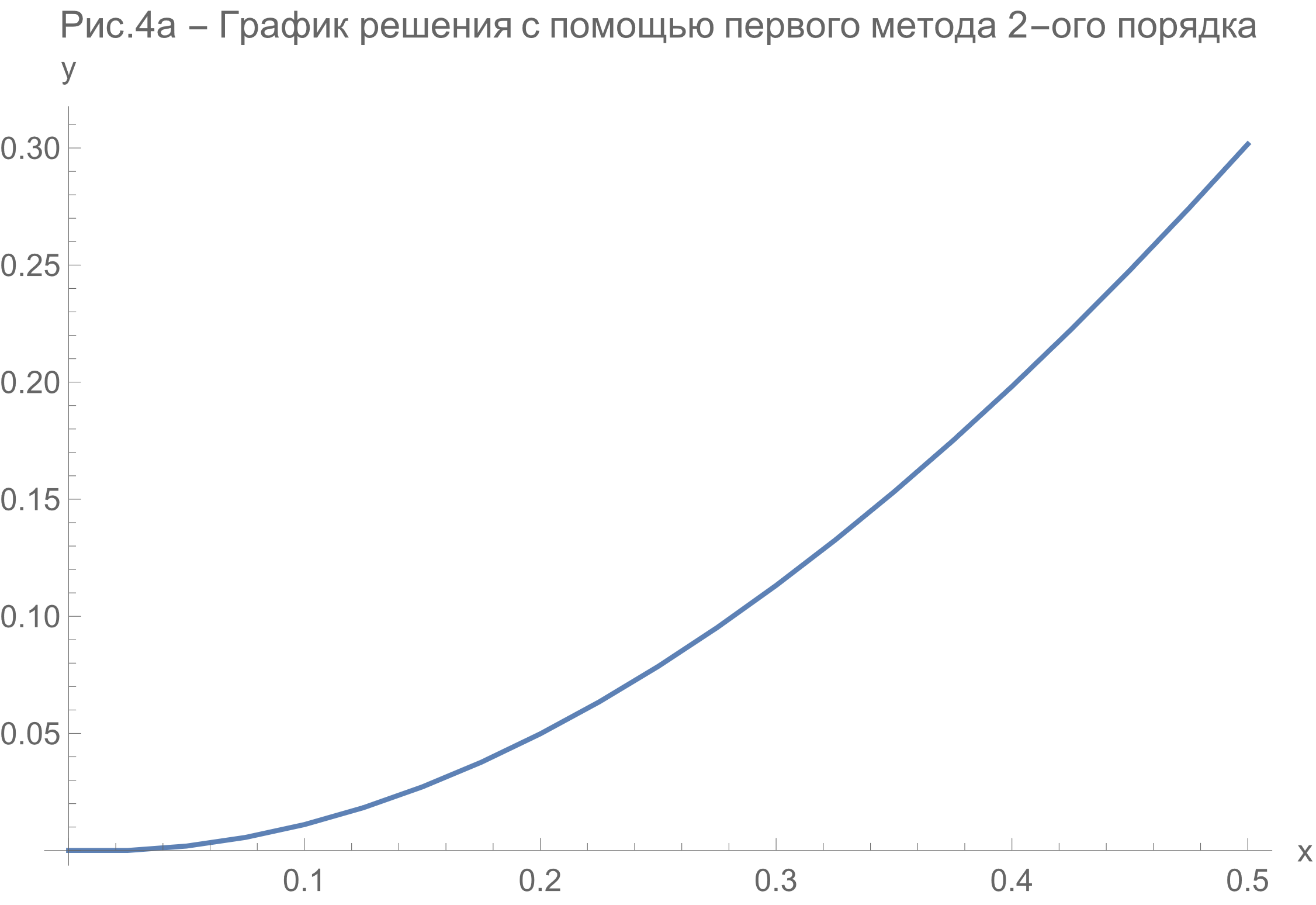
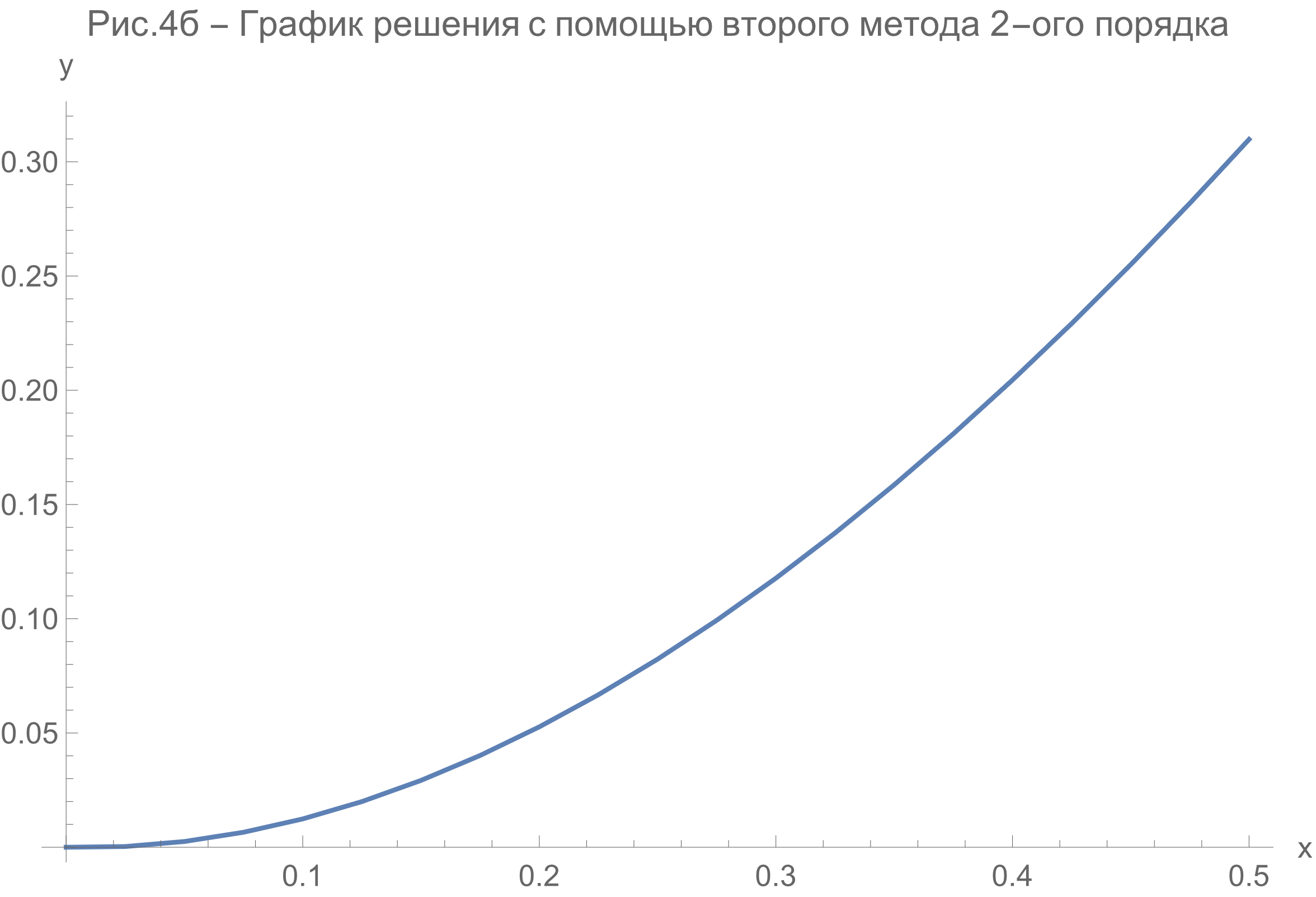
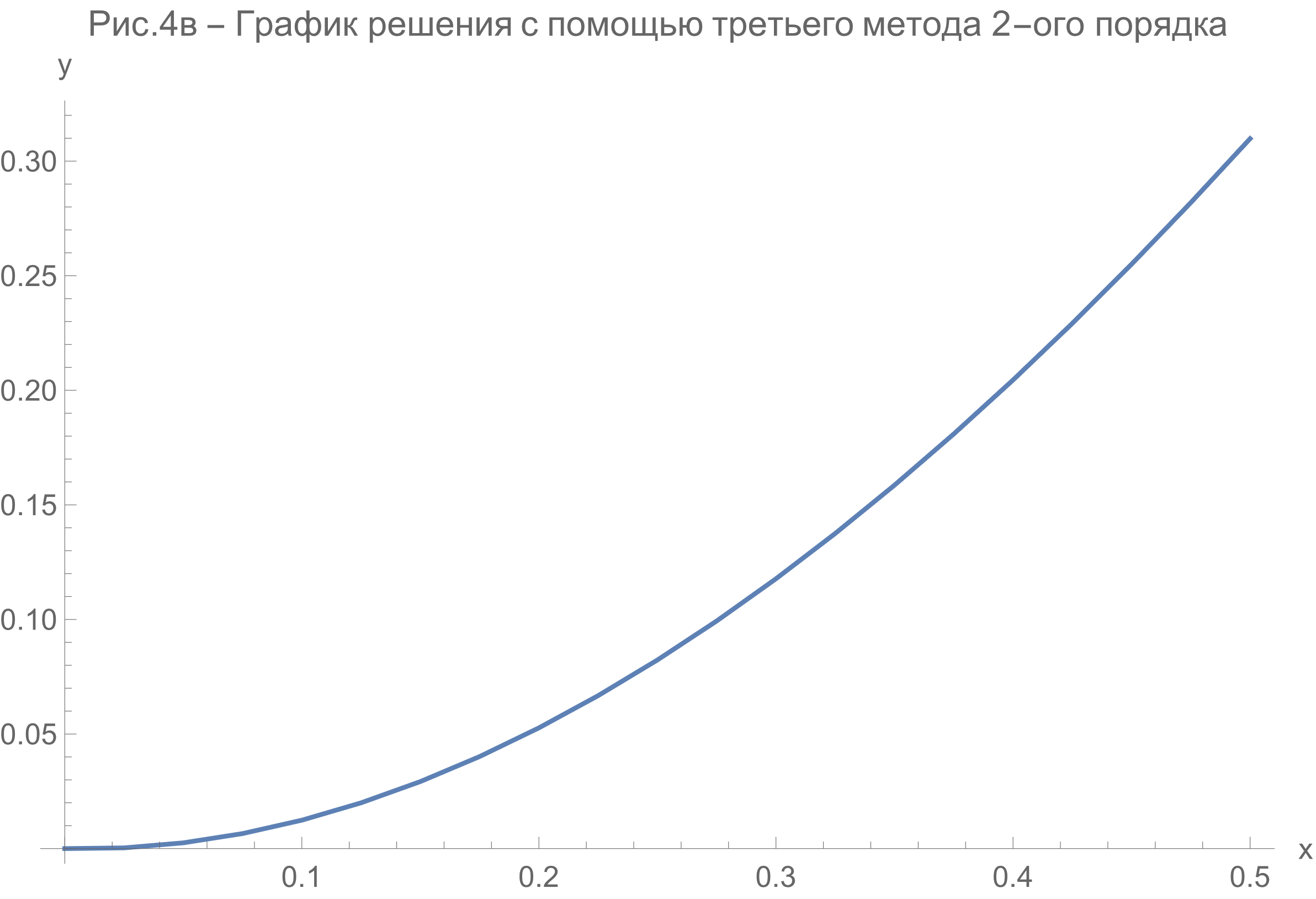


График метода 1-ого порядка нам понадобится позже.

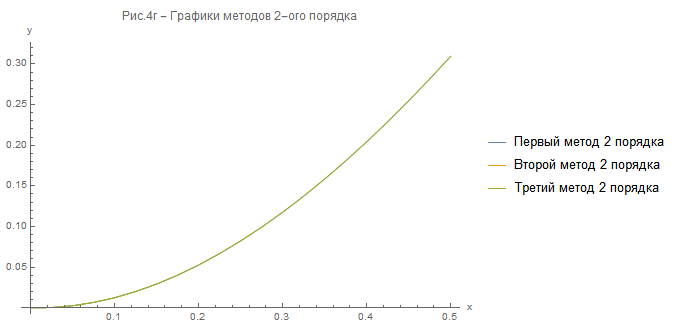
Перейдём к методам 2-ого порядка точности.





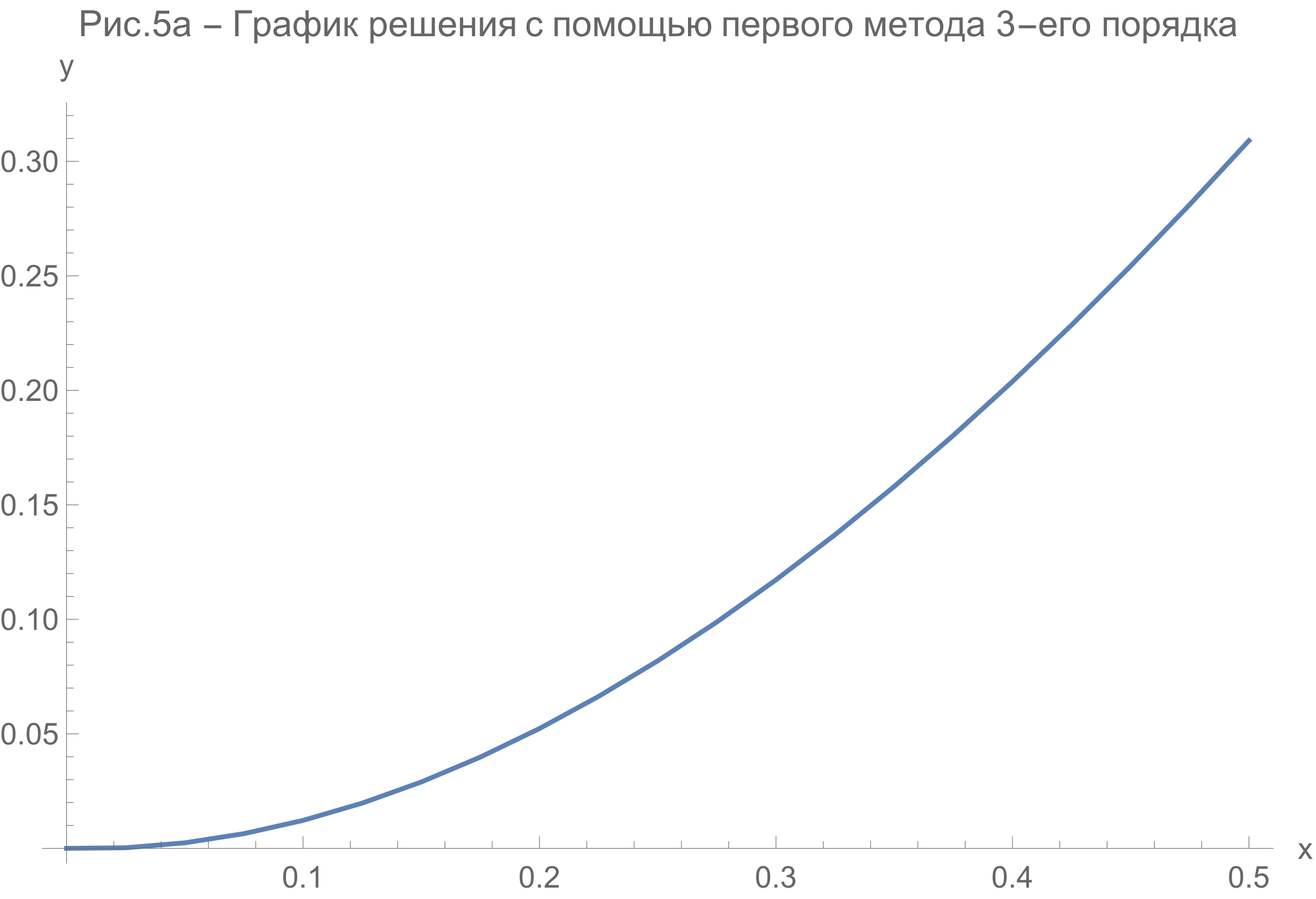


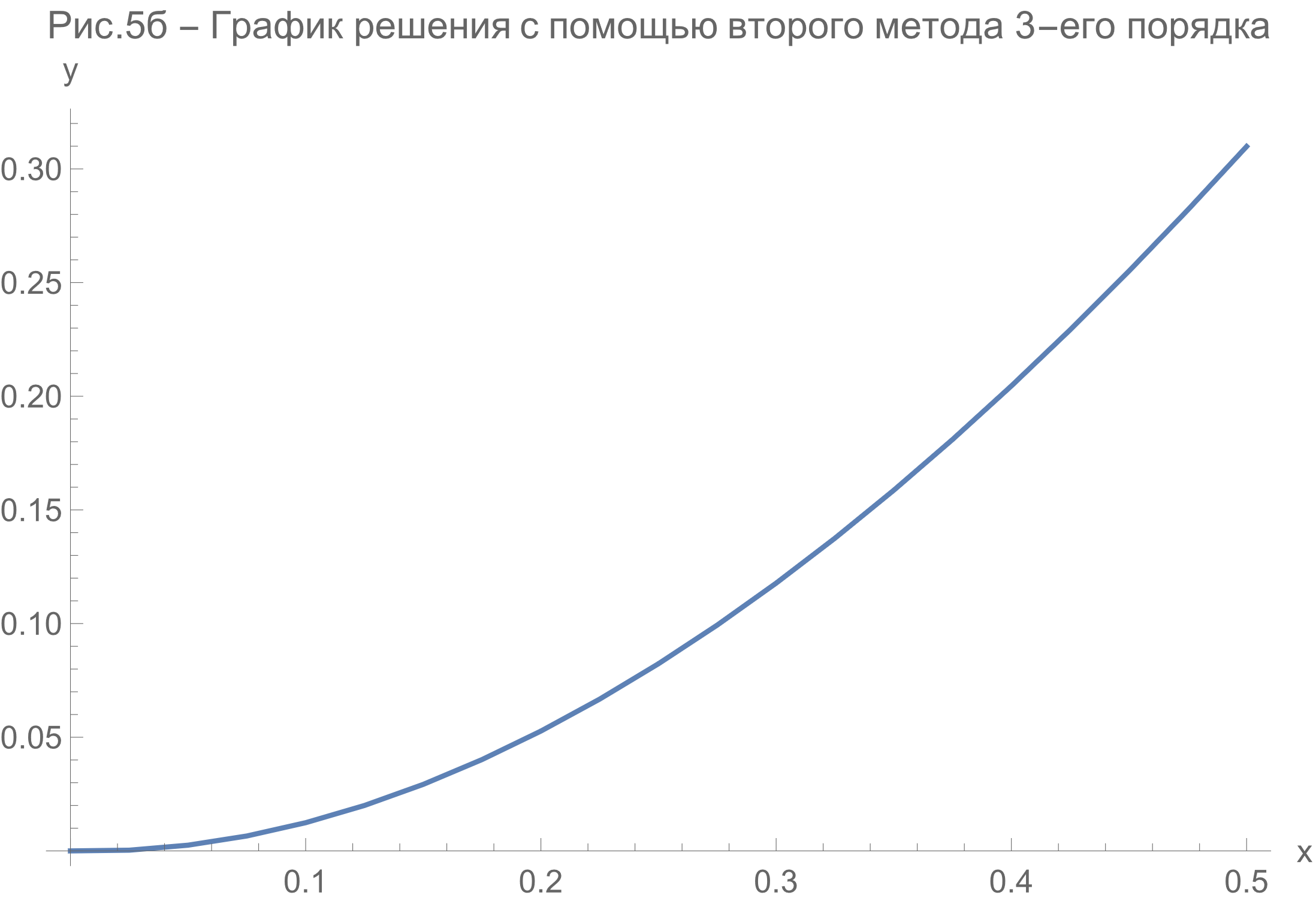
А теперь проверим сходимость этих методов друг к другу. Для того построим графики методов в одной системе координату.

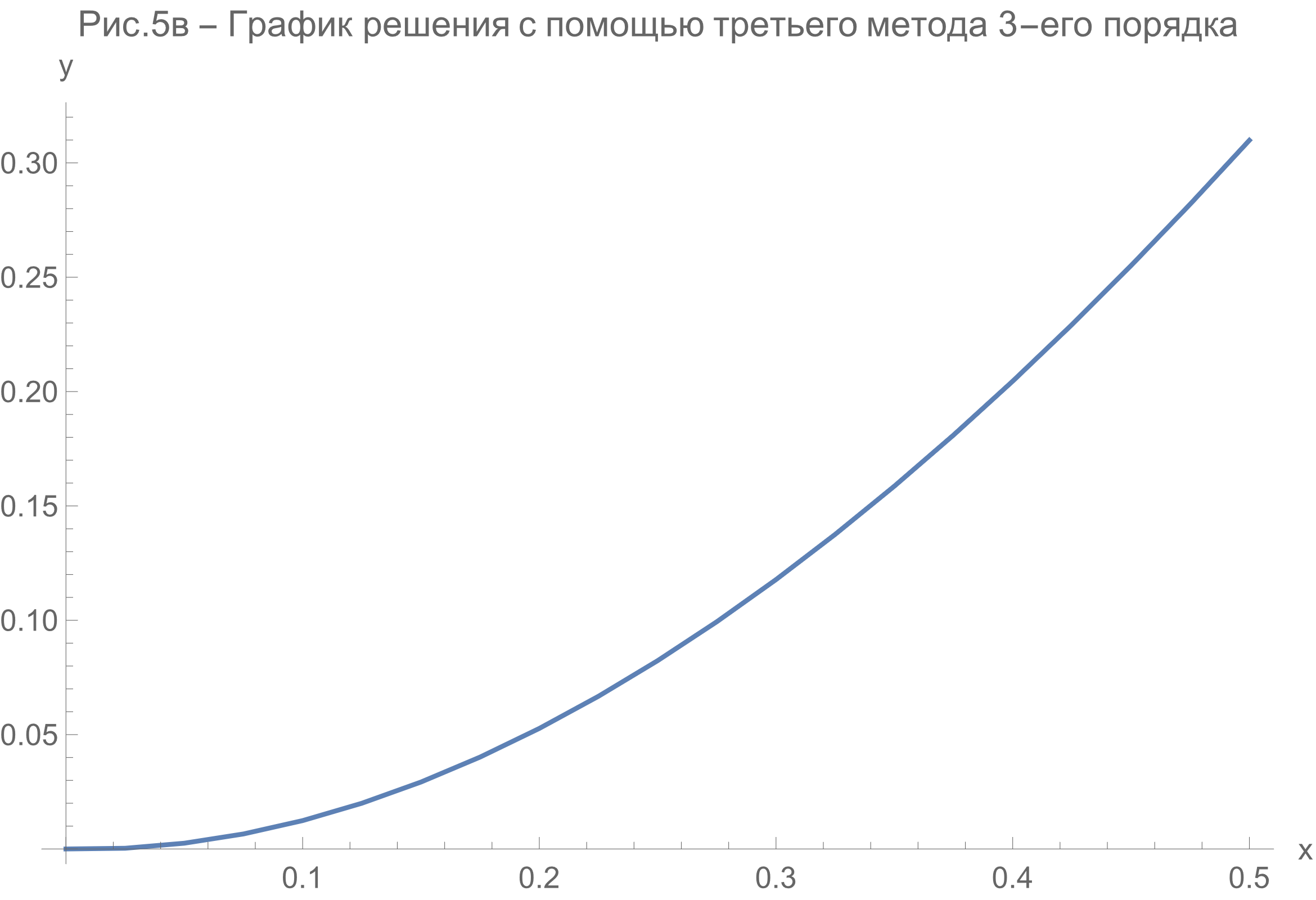


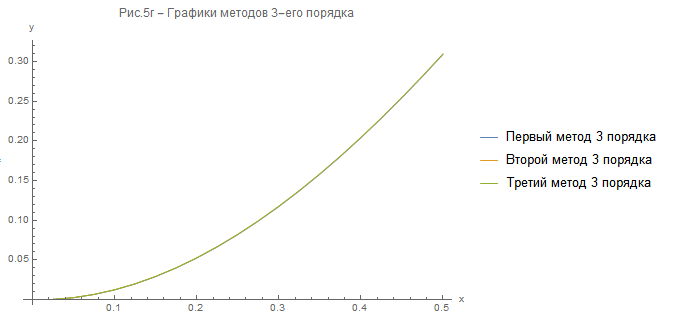
Как мы видим, они точно сходятся друг к другу, а это значит то, что мы, если хотим получить решения со 2-ым порядком точности, можем использовать любой из этих методов. В качестве представителя от методов 2-ого порядка выберем, например, первый метод.

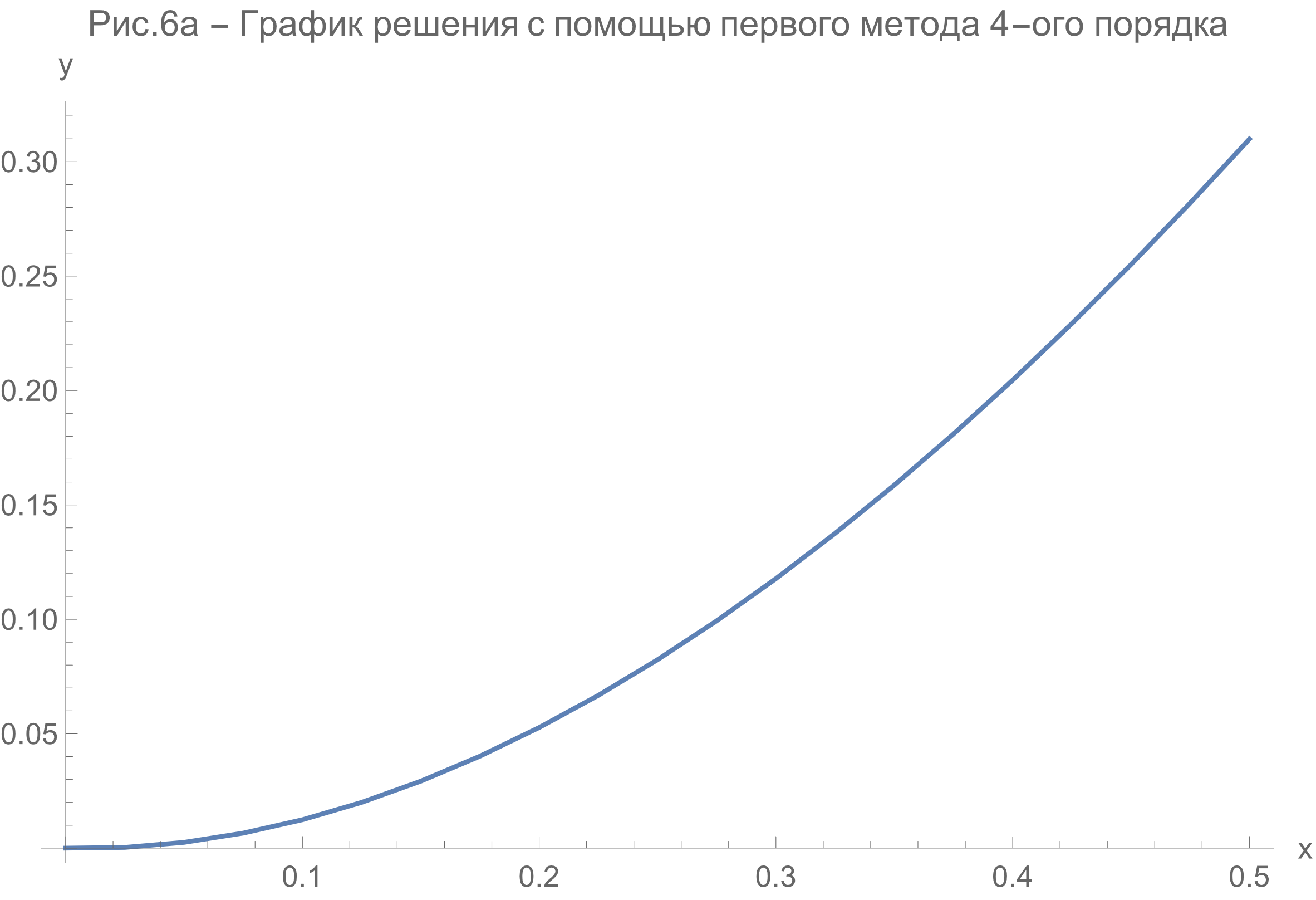
Проделаем всё тоже самое для методов 3-его и 4-ого порядка точности.

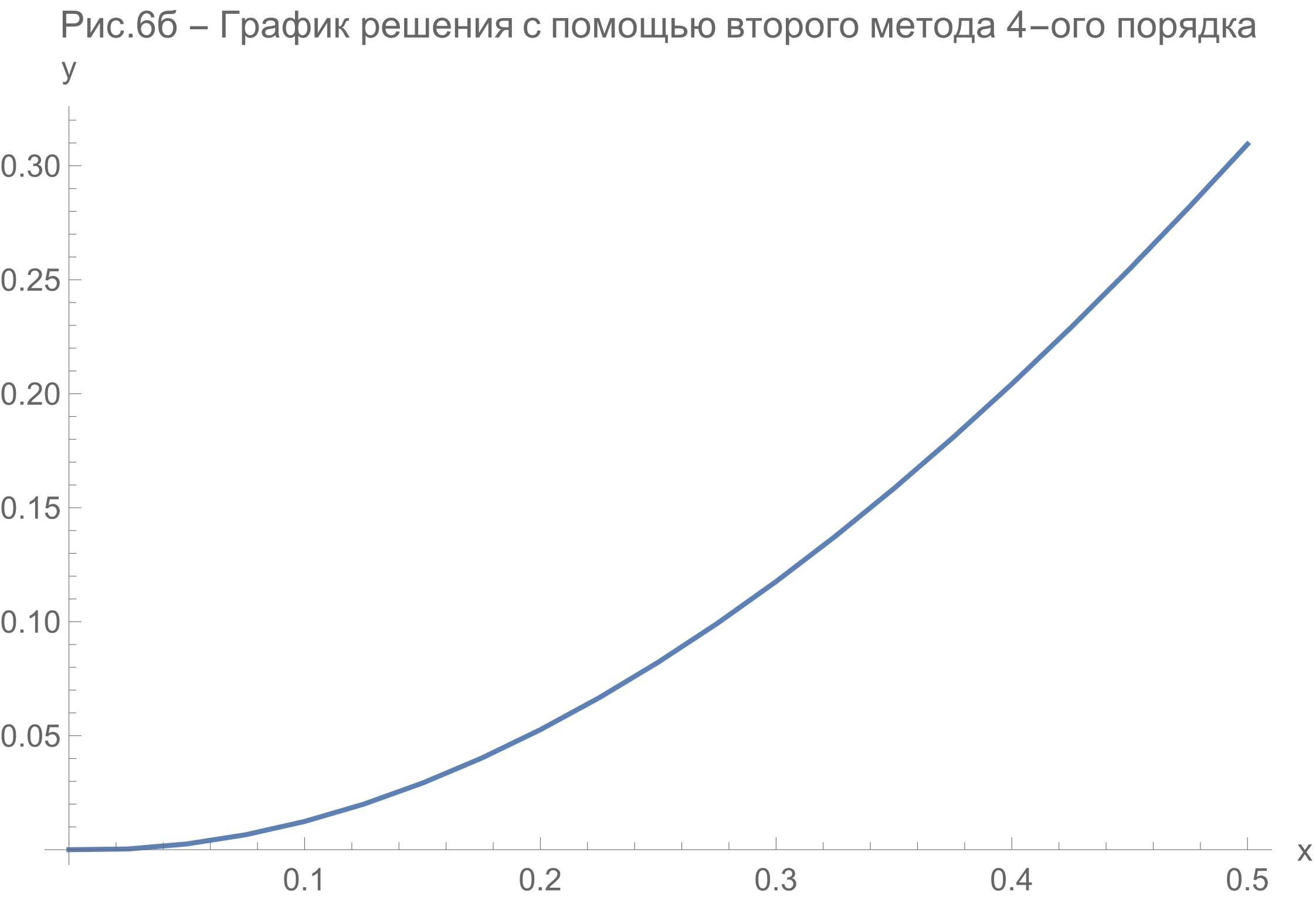


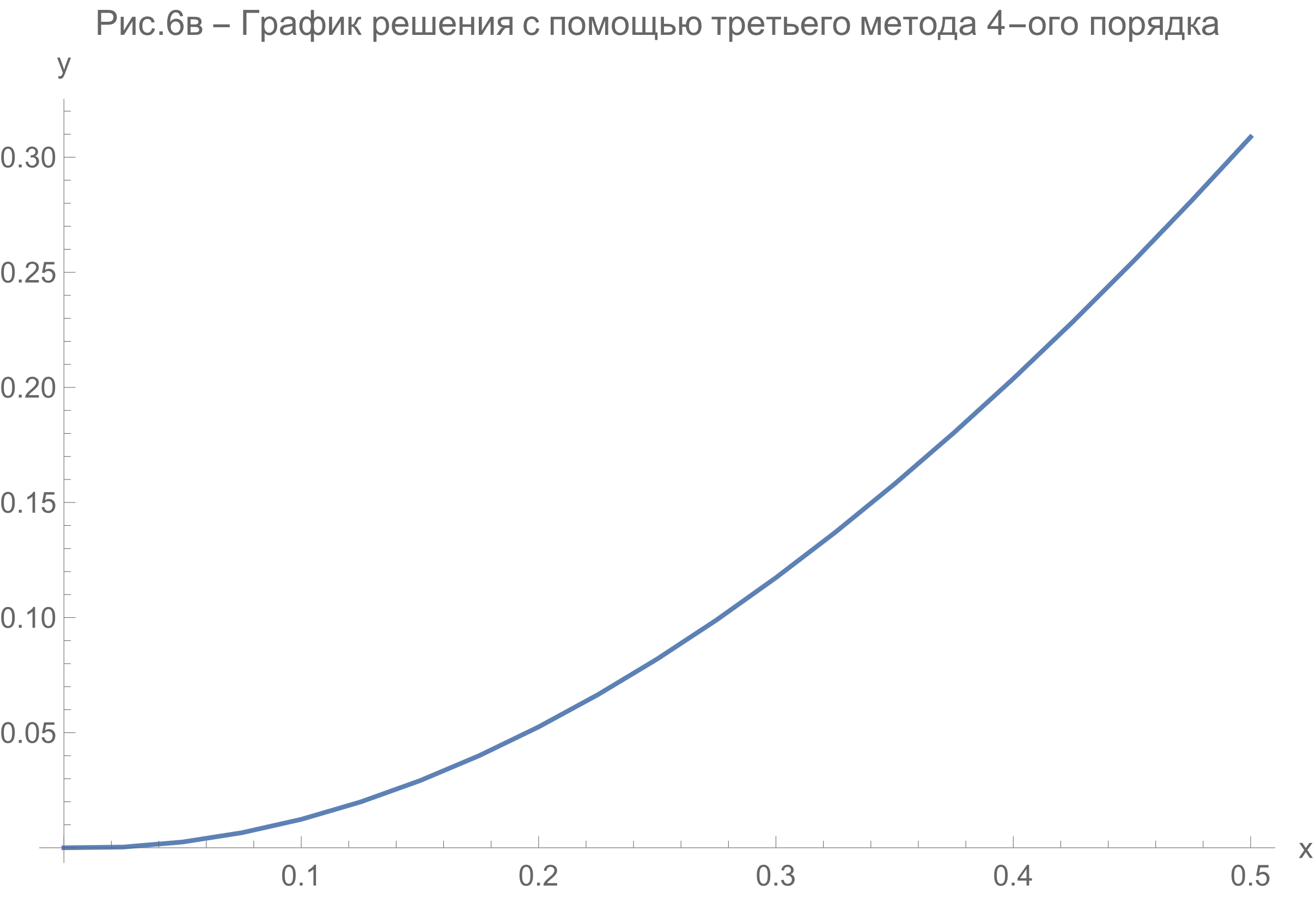


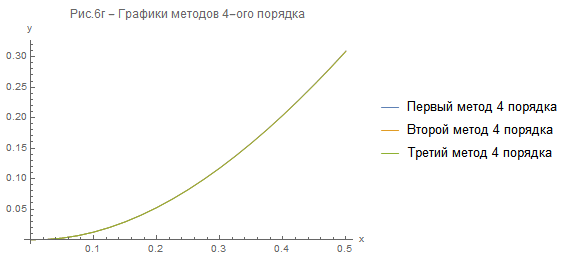




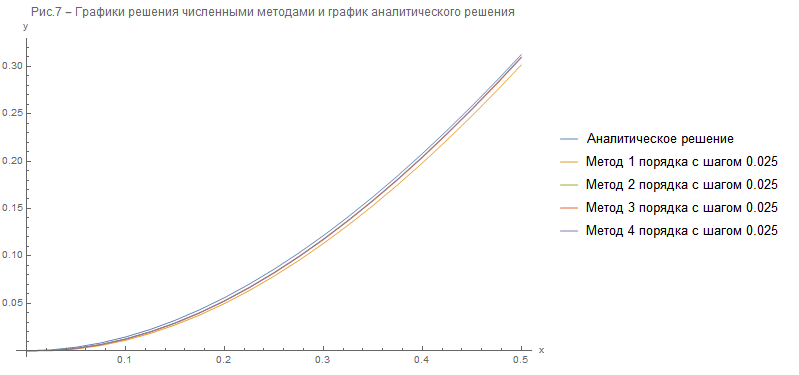






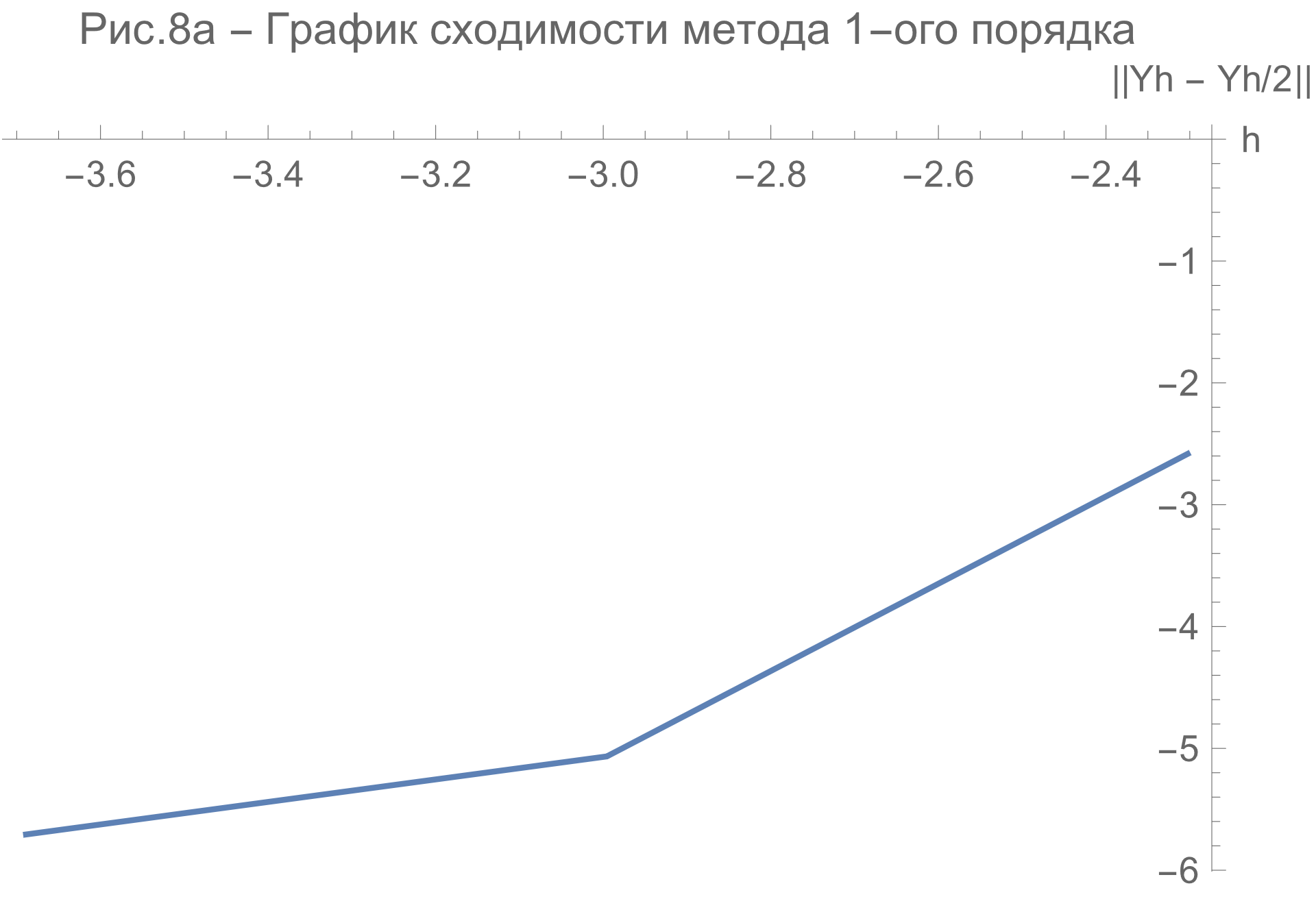


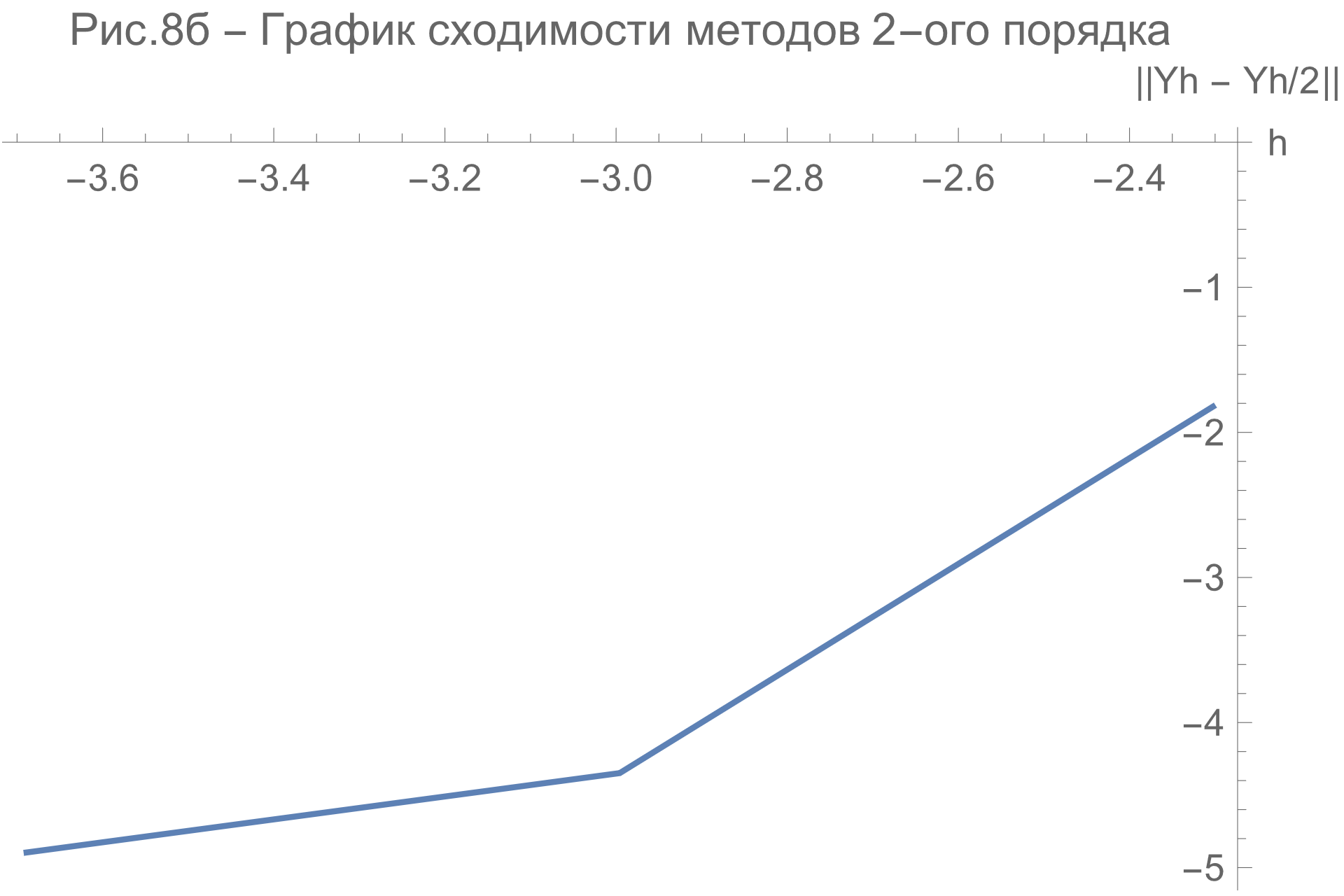
Как мы видим, методы 3-его и 4-ого порядка так же сходятся друг к другу соответственно. Возьмём, например, также 1-ые методы из каждой группы в качестве представителей своих методов и сверим графики представителей своих методов, полученные численно и график аналитического решения.

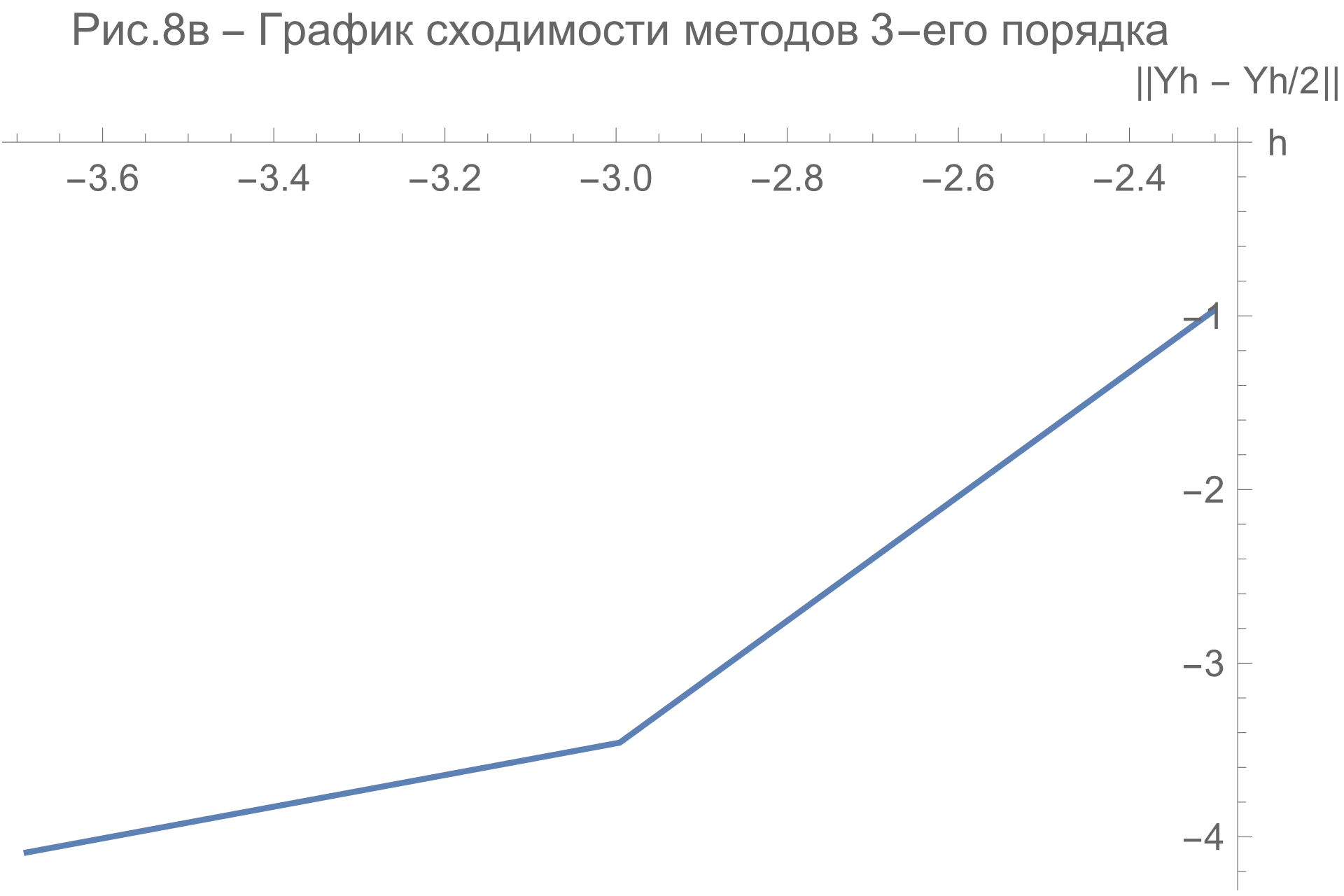


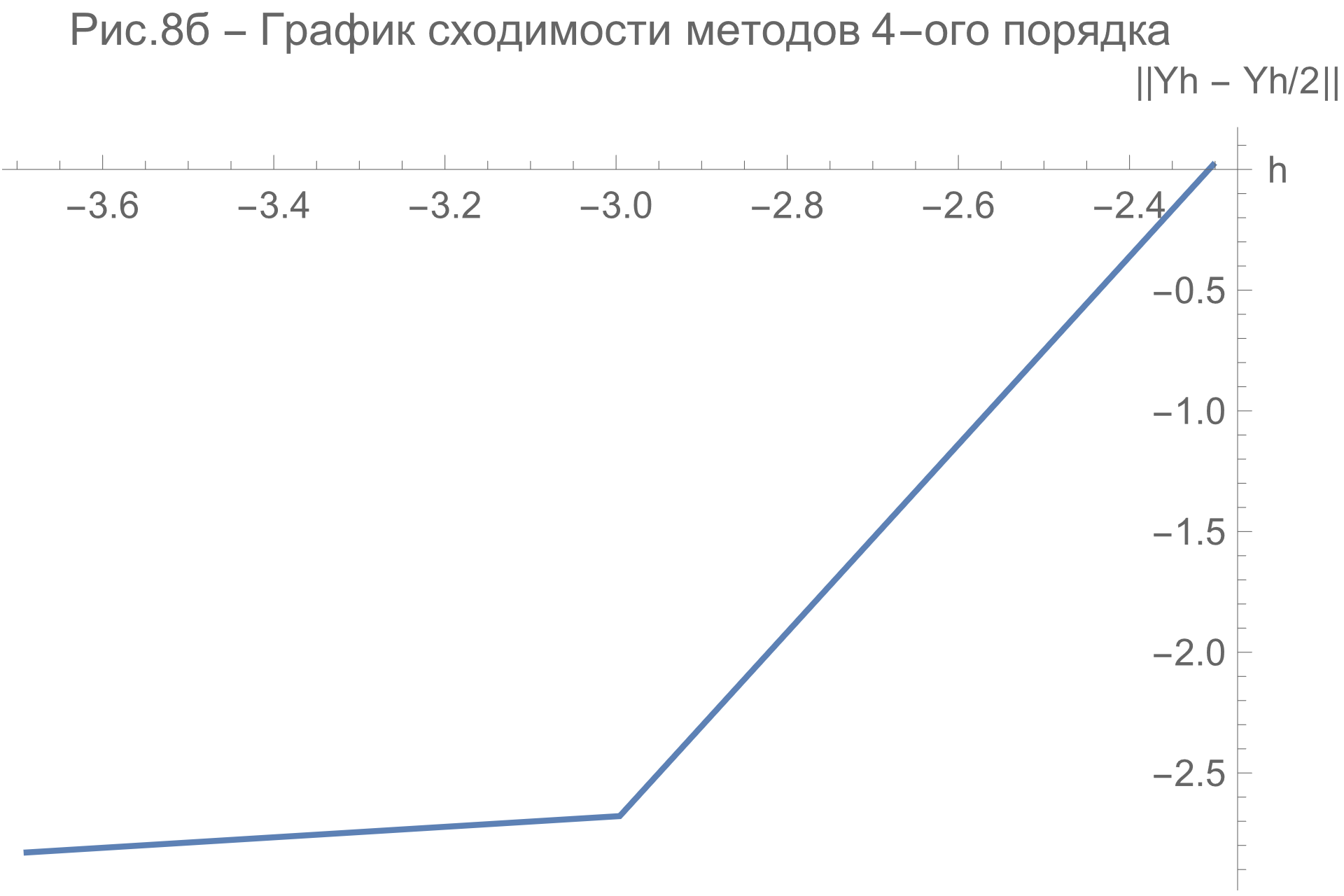
Из графиков очевидно, что каждый из методов отличается от аналитического решения на свою точность, а значит решение получено верно.

Осталось сверить лишь сходимости методов, для этого будем сверять аналитическое и численное решение, полученное разными методами. Как мы выяснили ранее методы соответствующих порядков сходятся к друг другу, поэтому проверим их сходимость для представителей этих методов.

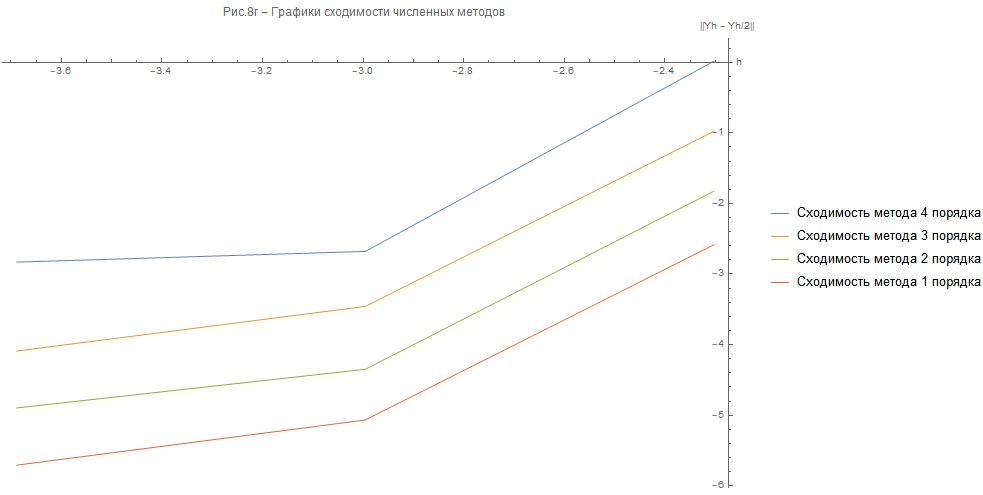








Теперь рассмотрим их сходимость между друг другом.



Графики так же отличаются из-за точности. Наиболее резко изменяющийся график обладает наилучшей сходимостью – это график 4-ых методов.

# **Заключение**

Сегодня Методы Рунге-Кутты играют важную роль в численном решении ОДУ. Они часто используются для решения сложных систем и уравнений. Привлекательны они тем, что является достаточно надёжными по сравнению с другими методами.

Я уверен, что в дальнейшем процессе обучения, ещё ни раз придётся с ними столкнуться

# **Список использованных источников**

1. В.И. Мышенков, Е.В. Мышенков, Учебное пособие для студентов “Численные методы часть вторая”, Издательство “МГУЛ”, 2005 – 109с.
2. М.Г. Бояршинов, Учебное пособие для студентов “ Численные методы 2”, Издательство “ПНИПУ”, 1999 - 200с.
3. В.В.Демченко, Учебно-методическое пособие ”Метод Рунге-Кутты для решения задачи Коши для ОДУ первого порядка”, Издательство “МФТИ”, 2004 – 20с.
4. Г.Ю.Куликов, Журнал Вычислительной Математики и физики Том36 – 1996 – 89с.
5. Б.Ф.Фалейчик ,Учебно-методическое пособие “Одношаговые методы численного решения задачи Коши”, Издательство “БГУ”, 2010 – 42с.

# **Приложения**

Реализация явных методов Рунге-Кутты на языке программирования С++.

double\* w = new double[m];//Равномерная сетка

for (int i = 0; i <= m; i++)

{

w[i] = a + i \* h;

}

for (int i = 1; i <= m + 1; i++)

{

double DeltaY, K0, K1, K2, K3;

switch (v1)

{

case 4://Методы 4 порядка точности

switch (v2)

{

case 1:

if (j == 0 && i == 1)

{

out.open("Solution4.txt");

pres.open("SolutionPres4.txt");

}

K0 = h \* (3 \* sin(w[i - 1]) - Y[i - 1]);

K1 = h \* (3 \* sin(w[i - 1]) + (h / 2) - Y[i - 1] + (K0 / 2));

K2 = h \* (3 \* sin(w[i - 1]) + (h / 2) - Y[i - 1] + (K1 / 2));

K3 = h \* (3 \* sin(w[i - 1]) + h - Y[i - 1] + K2);

DeltaY = (K0 + 2 \* K1 + 2 \* K2 + K3) \* 1 / 6;

break;

case 2:

if (j == 0 && i == 1)

{

out.open("Solution4.txt");

pres.open("SolutionPres4.txt");

}

K0 = h \* (3 \* sin(w[i - 1]) - Y[i - 1]);

K1 = h \* (3 \* sin(w[i - 1]) + (h / 3) - Y[i - 1] + (K0 / 3));

K2 = h \* (3 \* sin(w[i - 1]) + (2 \* h / 3) - Y[i - 1] - (K0 / 2) + K1);

K3 = h \* (3 \* sin(w[i - 1]) + h - Y[i - 1] + K0 - K1 + K2);

DeltaY = (K0 + 3 \* K1 + 3 \* K2 + K3) \* 1 / 8;

break;

case 3:

if (j == 0 && i == 1)

{

out.open("Solution4.txt");

pres.open("SolutionPres4.txt");

}

K0 = h \* (3 \* sin(w[i - 1]) - Y[i - 1]);

K1 = h \* (3 \* sin(w[i - 1]) + (h / 4) - Y[i - 1] + (K0 / 4));

K2 = h \* (3 \* sin(w[i - 1]) + (h / 2) - Y[i - 1] + (K1 / 2));

K3 = h \* (3 \* sin(w[i - 1]) + h - Y[i - 1] + K0 - 2 \* K1 + K2);

DeltaY = (K0 + 4 \* K2 + K3) \* 1 / 6;

break;

}

break;

case 3://Методы 3-его порядка точности

switch (v2)

{

case 1:

if (j == 0 && i == 1)

{

out.open("Solution3.txt");

pres.open("SolutionPres3.txt");

}

K0 = h \* (3 \* sin(w[i - 1]) - Y[i - 1]);

K1 = h \* (3 \* sin(w[i - 1]) + (h / 2) - Y[i - 1] + (K0 / 2));

K2 = h \* (3 \* sin(w[i - 1]) + (h / 2) - Y[i - 1] - K0 + 2 \* K1);

DeltaY = (K0 + 4 \* K1 + K2) \* 1 / 6;

break;

case 2:

if (j == 0 && i == 1)

{

out.open("Solution3.txt");

pres.open("SolutionPres3.txt");

}

K0 = h \* (3 \* sin(w[i - 1]) - Y[i - 1]);

K1 = h \* (3 \* sin(w[i - 1]) + (h / 3) - Y[i - 1] + (K0 / 3));

K2 = h \* (3 \* sin(w[i - 1]) + (2 \* h / 3) - Y[i - 1] + (2 \* K1 / 3));

DeltaY = (K0 + 3 \* K2) \* 1 / 4;

break;

case 3:

if (j == 0 && i == 1)

{

out.open("Solution3.txt");

pres.open("SolutionPres3.txt");

}

K0 = h \* (3 \* sin(w[i - 1]) - Y[i - 1]);

K1 = h \* (3 \* sin(w[i - 1]) + (h / 2) - Y[i - 1] + (K0 / 2));

K2 = h \* (3 \* sin(w[i - 1]) + (3 \* h / 4) - Y[i - 1] + (3 \* K1 / 4));

DeltaY = (2 \* K0 + 3 \* K1 + 4 \* K2) \* 1 / 9;

break;

}

break;

case 2://Методы 2-ого порядка точности

switch (v2)

{

case 1:

if (j == 0 && i == 1)

{

out.open("Solution2.txt");

pres.open("SolutionPres2.txt");

}

K0 = h \* (3 \* sin(w[i - 1]) - Y[i - 1]);

K1 = h \* (3 \* sin(w[i - 1]) + h - Y[i - 1] + K0);

DeltaY = (K0 + K1) \* 1 / 2;

break;

case 2:

if (j == 0 && i == 1)

{

out.open("Solution2.txt");

pres.open("SolutionPres2.txt");

}

K0 = h \* (3 \* sin(w[i - 1]) - Y[i - 1]);

K1 = h \* (3 \* sin(w[i - 1]) + (h / 2) - Y[i - 1] + (K0 / 2));

DeltaY = K1;

break;

case 3:

if (j == 0 && i == 1)

{

out.open("Solution2.txt");

pres.open("SolutionPres2.txt");

}

K0 = h \* (3 \* sin(w[i - 1]) - Y[i - 1]);

K1 = h \* (3 \* sin(w[i - 1]) + (2 \* h / 3) - Y[i - 1] + (2 \* K0 / 3));

DeltaY = (K0 + 3 \* K1) \* 1 / 4;

break;

}

break;

case 1://Метод 1-ого порядка точности

if (j == 0 && i == 1)

{

out.open("Solution1.txt");

pres.open("SolutionPres1.txt");

}

K0 = h \* (3 \* sin(w[i - 1]) - Y[i - 1]);

DeltaY = K0;

break;

}

Y[i] = Y[i - 1] + DeltaY;

}

Код из Wolfram Mathematica 12.1 для получения аналитического и возмущённого решения

DSolve[{y'[x]== 3\*Sin[x]-y[x]},y,x]

Sol=DSolve[{y'[x]== 3\*Sin[x]-y[x],y[0] == 0},y,x];

Plot[{y[x]/.Sol},{x,0,0.5},AxesLabel->{"x","y"},PlotLabel->"Рис.1 - График аналитического решения уравнения"]

dSol=DSolve[{z'[x]-10^-2== 3\*Sin[x]-z[x],z[0] == 10^-3},z,x];

Plot[{y[x]/.Sol,z[x]/.dSol},{x,0,0.5},PlotLegends->{"Исходная функция","Возмущённая функция"},AxesLabel->{"x","y"},PlotLabel->"Рис.2 - Проверка устойчивости задачи Коши"]