Снежана Георгиева Гочева-Илиева

КОМПЮТЪРНИ ЧИСЛЕНИ МЕТОДИ

http://fmi.uni-plovdiv.bg/gocheva/KChM-lekcii.pdf

Пловдив, 2013

УНИВЕРСИТЕТСКО ИЗДАТЕЛСТВО
„ПАИСИЙ ХИЛЕНДАРСКИ“

Рецензент: проф. д-р Николай Кюркчиев

ISBN 978-954-423-848-3

Пловдивски университет „Паисий Хилендарски”
Факултет по математика и информатика
Катедра “Приложна математика и моделиране”

“Компютърни числени методи”
ЛЕКЦИЯ 1

Проф. д-р Снежана Гочева-Илиева, snow@uni-plovdiv.bg

Он-лайн обучение: www.fmi-plovdiv.org/evlm
www.fmi-plovdiv.org/evlm/DBbg - числени методи

Литература:

1. Бояджиев Д., Гочева С., Макрелов И., Попова Л. – Ръководство по числени методи – част 1, Издания: 2003, 2006, 2010.
2. Семерджиев Х., Боянов Б., Числени методи, ПУ.
3. Гочева-Илиева С., Въведение в система Mathematica, ЕксПрес, Габрово, 2009.

Съдържание:

Основни понятия в компютърните числени методи

1. Основни характеристики на компютърното смятане

- Повишена скорост – около 10 млн. аритметични операции в секунда за среден клас персонален компютър
- Практически неограничена степен на точност
- Улеснен обмен на данните между различни програмни сегменти, програми, компютри и компютърни мрежи
- Високоефективни паралелни изчисления
- Символни изчисления
- Качествена машинна графика за представяне на резултатите
- Използуване на супермощни специализирани интерактивни софтуерни системи за научни пресмятания като *Mathematica*, *MatLab*, *Maple*, *Reduce* и др.
- Он-лайн изчисления на сайта: **http://www.wolframalpha.com/**

1.1 Пример 1. - Някои задачи изискват стотици и милиони години компютърни изчисления ...

<u>Правило:</u> Дори и с най-бърз компютър не трябва наивно да прилагаме математическите рецепти и дефиниции.

Преговор - Пресмятане на детерминанти.

А) Детерминанта от 2-ри ред (n=2):

$$A_2 = \begin{vmatrix} 2 & 1 \\ 3 & 2 \end{vmatrix} = 2.2 - 3.1 = 4 - 3 = 1$$

Б) Детерминанта от 3-ти ред (n=3):

$$A_3 = \begin{vmatrix} 2 & 1 & 0 \\ 3 & 4 & 1 \\ 2 & 0 & 5 \end{vmatrix} = 2.4.5 + 1.1.2 + 0.0.3 - 2.4.0 - 2.0.1 - 3.1.5$$

$$= 40 + 2 + 0 - 0 - 0 - 15 = 27$$

Пример 1-1. Като се приложи директно дефиницията, да се пресметне стойността на детерминанта от n -ти ред при n=20.

Решение:

$$A=\begin{vmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \cdots & & & \\ a_{n1} & a_{n1} & \cdots & a_{nn} \end{vmatrix} = \sum (-1)^{[\alpha_1\alpha_2\ldots\alpha_n]} a_{1\alpha_1} a_{2\alpha_2} \ldots a_{1\alpha_n} \quad , \qquad (0.1)$$

равно на сумата от $n!$ произведения, където сумирането се отнася до всичките $n!$ на брой пермутации $\alpha_1\alpha_2\ldots\alpha_n$ от вторите индекси.

Във всяко събираемо има n-1 произведения, събираемите са $n!$, т.е. общият брой аритметични действия е приблизително

$Брой = n.\ n!$

При $n = 20$ - $Брой = 20.20! = 20.20.19.18...1 \approx 4{,}8.10^{19}$.

В 1 година има средно $Сек\_\ година = 365.\ 24.\ 60.\ 60 = 3{,}1.10^{7}$ секунди и за всяка секунда се извършват $Опер\_секунда = 1.10^{7}$.

$$\frac{Брой}{Сек\_година.Опер\_секунда} = \frac{4{,}8.10^{19}}{3{,}1.10^{7}.10^{7}} \approx 1{,}5.10^{5} = 150\,000$$ години!

За щастие съществуват много по-икономични числени методи, които за същата задача изискват едва около n^3 операции, т.е. при $n = 20$ приблизително $20.20.20 = 8000$ и резултатът ще се появи на същия компютър за ... една хилядна част от секундата!

1.2 Пример 2. Взрив на грешката от закръгляне

Да се пресметне сумата: $\sin(x) = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \ldots + (-1)^k \frac{x^{2k+1}}{(2k+1)!} + \ldots$

Смятаме докато пореден член $< \varepsilon = 10^{-8}$, напр. за $x = \frac{\pi}{6} + 2k\pi$

k	x	**Sin(x) -** приближено решение
0	$\pi/6$	0,5000000
1	$\pi/6+2\pi$	0,5000052
2	$\pi/6+4\pi$	0,5000144
3	$\pi/6+6\pi$	0,6488953
4	$\pi/6+8\pi$	-448,0291
5	$\pi/6+10\pi$	66527,500
...	...	...

Взрив на грешката!

Точното решение за всяко k. $30^0 = 0,5$

ОСНОВНИ ПОНЯТИЯ

2. Числени методи и математическо моделиране

За решаване на математически задачи съществуват три основни групи методи: графични, аналитични и числени. И трите метода могат да се реализират на компютър, като за целта се използва специализиран научен софтуер: Mathematica, Maple, Matlab и др., или който и да е от стандартните езици за програмиране: C++, Pascal и др.

Графичните методи могат да помогнат за визуално определяне на редица характеристики на решението.

Аналитичните методи са тези, чрез които решението може да се получи точно, във вид на формула.

Числените методи получават резултата приближено във вид на числа.

Определение. **Числените методи** свеждат даден клас математически задачи до ефективни алгоритми, с помощта на които те могат да се решат с краен брой аритметични действия и за краен интервал от време, а грешката на резултата може да се оцени предварително. Те са изцяло ориентирани към пресмятания с компютър.

За да се реши дадена реална задача с някой от изброените методи, тя трябва предварително да бъде зададена в подходящ вид, наречен математически модел.

Определение. **Математическият модел** е приближено описание на даден клас реални явления с помощта на математическа символика.

Например: Изчисляването на обема на бензин в цистерна се свежда до тримерен интеграл, движението на спътник - до система диференциални уравнения с променливи коефициенти и т.н.

Изключително важно за всеки модел е той да бъде адекватен, т.е. достатъчно точно да отразява същността на моделираното явление или процес. Адекватността на модела се установява чрез проверката му в практиката, а най-добрият критерий за адекватност е, когато моделът може да “предсказва” достатъчно точно бъдещите състояния на процеса или явлението.

Когато моделът не дава достатъчно удовлетворително решение на поставената задача неговите параметри могат многократно да се коригират, а самият модел може да се променя. Тази последователност от промени в модела се нарича математическо моделиране.

Определение. **Математическото моделиране** е цикличен процес на създаване и уточняване на математически модели. Целта на този процес е получаването на достатъчно удовлетворителни резултати, в рамките на зададена допустима грешка.

Математическото моделиране преминава през следните основни етапи:

1. **Постановка на задачата.**
2. **Построяване на математически модел.**
3. **Избор или разработване на числен метод.**
4. **Съставяне на алгоритъм.**
5. **Програмиране.**
6. **Получаване на резултати от модела.**
7. **Цикъл на моделирането до достигане на удовлетворителни резултати.**
8. **Внедряване.**

3. Видове грешки

Основен въпрос при числените решаване на задачи е анализ на възможните грешки.

Различават се следните видове грешки:

- неотстранима грешка (грешка от експериментални данни)
- грешка от изчисления (от закръгляне и от аритметични действия)
- грешка от числения метод

Сумата от тези три грешки формира т.н. пълна грешка, която характеризира крайния резултат.

3.1 Приближаване на числа, абсолютна и относителна грешка

Нека x е дадено точно число, а $\tilde{x}$ е негово приближено.

Определение 1. Реалното число $\alpha(\tilde{x})$ се нарича абсолютна грешка на приближението $\tilde{x}$ на x, ако е горна граница на модула на тяхната разлика, т.е. $\alpha(\tilde{x}) \geq |x - \tilde{x}|$. (1)

Последното се записва и като равенство $\alpha(\tilde{x}) = |x - \tilde{x}|$.

Ще отбележим, че абсолютната грешка зависи от мерната единица, например килограм, метър, унция и т.н. Ако разрешим неравенството (1) спрямо x ще получим: $\tilde{x} - \alpha(\tilde{x}) \leq x \leq \tilde{x} + \alpha(\tilde{x})$, което определя интервала $[\tilde{x} - \alpha(\tilde{x});\ \tilde{x} + \alpha(\tilde{x})]$, в който се намира точното число.

Определение 2. Реалното число $\Delta(\tilde{x})$ се нарича относителна грешка на приближението $\tilde{x}$ на x, ако удовлетворява условието

$$\Delta(\tilde{x}) \geq \frac{\alpha(\tilde{x})}{|\tilde{x}|}, \qquad \tilde{x} \neq 0. \tag{2}$$

Записва се и като:

$$\Delta(\tilde{x}) = \frac{\alpha(\tilde{x})}{|\tilde{x}|}$$

Относителната грешка е безмерна величина и се изразява с проценти.

Пример 1. Да се намерят относителните и абсолютните грешки на числото $x = \frac{2}{7}$ и две негови приближения $\tilde{x}_1 = 0,286$ и $\tilde{x}_2 = 0,2857$.

Решение. Имаме $x = \frac{2}{7} = 0,285714285...$.

За абсолютните грешки намираме:

$$\left|x - \tilde{x}_1\right| = \left|-0,00028571...\right| \le \alpha(\tilde{x}_1),$$

$$\left|x - \tilde{x}_2\right| = \left|0,00001428...\right| \le \alpha(\tilde{x}_2).$$

Можем да изберем например: или $\alpha(\tilde{x}_1) = 0,0003$.

Съответно за $\tilde{x}_2$: $\alpha(\tilde{x}_2) = 0,00002$, $\alpha(\tilde{x}_2) = 0,00005$ или $\alpha(\tilde{x}_2) = 0,0001$.

За относителните грешки намираме:

$$\Delta(\tilde{x}_1) = \frac{\alpha(\tilde{x}_1)}{|\tilde{x}_1|} = \frac{0,0003}{0,286} = 0,001048...,$$

$\Delta(\tilde{x}_1) = 0,002$, т.е. $\Delta(\tilde{x}_1) = 0,2\%$.

$$\Delta(\tilde{x}_2) = \frac{\alpha(\tilde{x}_2)}{|\tilde{x}_2|} = \frac{0,00005}{0,2857} = 0,000175...,$$

$\Delta(\tilde{x}_2) = 0,0002$ или $\Delta(\tilde{x}_2) = 0,02\%$.

Пример 2. Да се определи абсолютната и относителната грешка на числото $b = 81,002$.

Решение. В случая не знаем дали b е точно число или приближено. Затова формално считаме, че последната цифра се явява резултат от правилата на закръгляне до третия знак, т.е. абсолютната грешка е $\alpha(b) = 0,0005$.

Интервалът, в който се намира числото е:
(81,0015; 81,0025).

Относителната му грешка е:

$$\Delta(b) = \frac{\alpha(b)}{|b|} = \frac{0,0005}{81,002} = 0,000006172...,$$

т.е. $\Delta(b) = 0,0007\%$.

Пример 3. Дадено е приближеното число $\tilde{z} = 0,00123$. Да се определят абсолютната и относителната му грешка.

Решение. За абсолютната грешка имаме: $\alpha(\tilde{z}) = 0,000005$,

а за относителната $$\Delta(\tilde{z}) = \frac{\alpha(\tilde{z})}{|\tilde{z}|} = \frac{0,000005}{0,00123} = 0,004065...$$, т.е.

$\Delta(\tilde{z}) = 0,41\%$.

Забележка. Обърнете особено внимание на последния резултат! Той показва, че относителните грешки на числа, близки до нулата, са големи! Следователно, следва да се избягва деление на малки числа.

Задача. Да се определи кое претегляне е по-точно: (а) на жп вагон с тегло 45 т и грешка $\alpha_1 = 50$ кг, или (б) на лекарство с тегло 0,5 грама и грешка $\alpha_2 = 0,001$ грама.

3.2 Грешки от изчисления

Нека x и y са точни, а $\tilde{x}$ и $\tilde{y}$ са съответни на тях приближени числа. Означаваме $z = x + y$ и $\tilde{z} = \tilde{x} + \tilde{y}$. За абсолютната грешка на сумата намираме:

$$\alpha(\tilde{z}) = |z - \tilde{z}| = |x + y - \tilde{x} - \tilde{y}| \le |x - \tilde{x}| + |y - \tilde{y}| \le \alpha(\tilde{x}) + \alpha(\tilde{y}),$$

или

$$\alpha(\tilde{x} + \tilde{y}) \le \alpha(\tilde{x}) + \alpha(\tilde{y}).$$

Това показва, че при събиране абсолютната грешка на сумата е равна на сумата от абсолютните грешки на събираемите.

Аналогично се доказват подобни оценки за всички аритметични действия. Валидна е следната по-обща

Теорема. Нека $u = f(x)$ е произволна диференцируема функция на x, където $f(x)$ се разглежда като последователност от аритметични действия над x. Ако $\tilde{x}$ е приближение на x, и $\tilde{u} = f(\tilde{x})$ е съответното приближение към u. Тогава абсолютната грешка на резултата е непрекъсната функция на грешката $\tilde{x}$.

Доказателство. Нека предположим, че f е диференцируема функция в околност на x. Тогава по теоремата за средните стойности:

$$\alpha(\tilde{u}) = |u - \tilde{u}| = |f(x) - f(\tilde{x})| = |f'(\xi)(x - \tilde{x})| \le |f'(\xi)||x - \tilde{x}| \le |f'(\xi)|\alpha(\tilde{x})$$

При ограничена първа производна получаваме:

$$\alpha(\tilde{u}) \le M\alpha(\tilde{x}), \quad M = const,$$

с което теоремата е доказана.

От тази теорема в частност следва, че с точност до константа за основните аритметични действия, са изпълнени оценките:

$$\alpha(\tilde{x}-\tilde{y})=\alpha(\tilde{x})+\alpha(\tilde{y}),\ \alpha(\tilde{x}.\tilde{y})=\alpha(\tilde{x})+\alpha(\tilde{y}),\ \alpha(\tilde{x}/\tilde{y})=\alpha(\tilde{x})+\alpha(\tilde{y}),\ \tilde{y}\neq 0.$$

Реално, при извършване на огромен брой изчисления поради различните знаци на приближенията (с недостиг или с излишък) абсолютните грешки се компенсират и сумарната грешка е малка. Не са рядко изключение обаче и случаите, при които грешките само се натрупват и могат да доведат до резултати, клонящи към безкрайност. Този феномен се нарича “взрив на грешката” или изчислителна неустойчивост. Той може да се прояви независимо

от точността (броят на знаците след десетичната запетая) на участващите числа.

3.3 Неустойчивост на грешката

Определение 3. Неустойчивост на грешката от изчисления (абсолютна или относителна) се нарича случаят, когато “малки” грешки във входните данни видят до “големи” грешки в резултата.

В увода бе приведен пример за взрив на грешката при сумиране на функцията $\sin(x)$. Ще приведем още един характерен пример, свързан с неустойчивостта на изчисленията.

Пример 4. Неустойчивост на относителната грешка при изваждане на близки числа.

Нека разгледаме близките числа $x = 2{,}123$ и $y = 1{,}985$. Разликата им е $z = x - y = 0{,}138$.

Имаме $\alpha(x) = \alpha(y) = 0{,}0005$, $\alpha(z) = \alpha(x) + \alpha(y) = 0{,}001$,

$$\Delta(x) = \frac{\alpha(x)}{|x|} = \frac{0{,}0005}{2.123} = 0{,}0002355..., \ \Delta(x) = 0{,}025\%,$$

$$\Delta(y) = \frac{\alpha(y)}{|y|} = \frac{0{,}0005}{1.985} = 0{,}0002518..., \ \Delta(y) = 0{,}026\%,$$

$$\Delta(z) = \frac{\alpha(z)}{|z|} = \frac{0{,}001}{0{,}138} = 0{,}00724..., \ \Delta(z) = 0{,}73\%.$$

За отношението на относителните грешки получаваме $0,73/0,025 \approx 29$, т.е. относителната грешка е на разликата е 29 пъти по-голяма спрямо относителната грешка на x и y.

Забележка. Един подход за избягване изваждането на близки числа е тяхното "раздалечаване" чрез рационализиране.

Например: $\sqrt{101}-10 = \dfrac{(\sqrt{101}-10).(\sqrt{101}+10)}{\sqrt{101}+10} = \dfrac{1}{\sqrt{101}+10}$.

3.4 Правила при действия с приближени числа

Редът на грешката зависи както от точността на началните данни, така и от този на междинните резултати.

- Намира се приближеното число от началните данни с най-голяма грешка, която определя и грешката на резултата. Нека тя е например в k -тия знак след десетичната запетая.
- Останалите начални данни могат да се закръглят с 1-2 знака повече от k .
- Всички междинни резултати се извършват с 1-2 знака след k .
- Избягва се по възможност изваждане на близки числа и деление на числа, близки до 0.

- Резултатът закръгляме до k -ти знак след десетичната запетая, ако няма и друг тип грешка – например от числения метод.

Забележка. Много да се внимава с математически, физични и други константи, които следва да се задават с възможно по-голяма точност. Например заместването $\pi = 3{,}14$ реално може да влоши точността на резултата до два знака след десетичната запетая, дори ако се работи с двойна точност над другите данни.

3.5 Грешки от числения метод

Всяка решавана задача може да се запише формално в явен вид:

$$y = A(x), \quad x \in R_1, \quad y \in R_2, \tag{3}$$

или в неявен вид:

$$A(x, y) = 0, \quad A : R_1 \to R_2 \tag{4}$$

където се търси y по дадено x, или x по дадено y (обратна задача). Тук са използвани означенията $R_{1,2}$ за съответните пространства, към които принадлежат x и y, а A е оператор (последователност от действия), преобразуващ x в y.

За да оценяваме "близостта" на елементите, ще считаме, че съответните пространства са нормирани и ще означаваме нормите им съответно с $\|\cdot\|_{R_1}$,$\|\cdot\|_{R_2}$.

При прилагането на даден числен метод, той от своя страна работи обикновено в подпространства на дадените, например $\overline{R_1} \subset R_1$, $\overline{R_2} \subset R_2$, а операторът A се заменя с приближен $\tilde{A}$. Така получаваме задачата:

$$\tilde{y} = \tilde{A}(\tilde{x}) , \qquad (5)$$

където $\tilde{x}$ е някакво приближение до началните данни (получено чрез измерване, при закръгляне и т.н.), а $\tilde{y}$ е приближение, получено от избрания ЧМ.

Решаването на конкретна задача от класа (5) със зададени значения на x ще доведе до допускане на изчислителни грешки,т.е. до приближена задача от вида:

$$\tilde{\tilde{y}} = \tilde{\tilde{A}}(\tilde{\tilde{x}}) \quad .$$

За разликата от точното и приближеното решение имаме:

$$y - \tilde{\tilde{y}} = (y - \tilde{y}) + (\tilde{y} - \tilde{\tilde{y}})$$

и използувайки свойствата на нормите, получаваме оценката:

$$\| y - \tilde{\tilde{y}} \|_{\overline{R_2}} \leq \| y - \tilde{y} \|_{\overline{R_2}} + \| \tilde{y} - \tilde{\tilde{y}} \|_{\overline{R_2}} \; . \qquad (6)$$

Това показва, че пълната грешка на резултата не надминава сумата от грешката на избрания числен метод и изчислителната грешка. Стои проблемът за оценяване на тези две грешки.

Един пример за задача от вида (3-4) е изчисляването на определен риманов интеграл от реална функция $I = \int_a^b f(t)dt$. Тук начални данни са a, b и функцията $f(t)$, операторът A е действието интегриране, а търсеният резултат е числото I. Пространствата са съответно: $R_1 = C_{[a,b]}$, $R_2 = R$.

При използването на числен метод за пресмятане на интеграла, например, чрез някоя квадратурна формула, се работи в пространствата $\overline{R_1} = R$ и $\overline{R_2} = R$, а операторът $\overline{A}$ е крайна сума, зависеща от параметър, например стъпка h. Приближеното значението на интеграла се получава с някаква грешка, пропорционална на степен на стъпката.

Определение. Казваме, че $\psi = O(\varphi)$, ако съществува константа $M \geq 0$, такава, че за функциите $\varphi(x)$ и $\psi(x)$ е вярно неравенството $|\psi(x)| \leq M\,|\varphi(x)|$ в определена област. Символът O се нарича символ на Ландау и се чете "о-голямо".

Чрез този символ например точността на даден метод може да се представи във вида $O(h^s)$, където h е малък параметър.

Друг изключително важен проблем, освен оценка на грешката в (6), касае поведението на решението на изходната задача (3-4), както и това на числения метод.

Определение 4. Решението на задача (3) е устойчиво по начални данни, когато е вярно неравенството:

$$\| y - \tilde{y} \|_{R_2} \leq C \| x - \tilde{x} \|_{R_1} . \qquad (7)$$

Това неравенство означава, че решението y зависи непрекъснато от x, т.е. малки грешки в изходните данни водят до малки грешки в резултата.

При типично неустойчива задача, незначителни промени в данните (след закръгляне) водят до големи грешки, а понякога и до съвсем неверни решения.

Определение 5. Ще казваме, че задачата (3) е **коректна**, когато са изпълнени следните условия:

1^0) съществува решение y,

2^0) решението е единствено в подобласт на R_2,

3^0) решението е устойчиво по начални данни.

Освен неустойчивост по начални данни има и други типове неустойчивост, например неустойчивост относно коефициентите на задачата, относно дясната част и пр.

В други случаи, макар и изходната задача да е коректно поставена, може да се окаже неустойчив избрания числен метод. Аналогично на горното определение един ЧМ ще наричаме коректен, когато съществува единствено и

устойчиво по начални данни решение на задача (5) в някаква подобласт на $\overline{R_1}$. Има много случаи на неустойчиви, т.е. некоректни ЧМ, например численото диференциране, решаване на диференциални задачи и други.

Определение 6. Ще казваме, че един числен метод, зависещ от параметър h е сходящ, когато полученото по метода приближено решение $\tilde{y}_h$ е сходящо по норма към точното решение $\tilde{y}_h$ на задача (3), т.е.

$$\left\| y - \tilde{y}_h \right\| \underset{h \to 0}{\to} 0 .$$

Следователно основните изисквания за правилно прилагане на даден числен метод са: изходната задача да е коректна, а числениият метод да е устойчив и сходящ.

Пловдивски университет „Паисий Хилендарски”
Факултет по математика и информатика
Катедра “Приложна математика и моделиране”

“Компютърни числени методи”
ЛЕКЦИЯ 2

Проф. д-р Снежана Гочева-Илиева, snow@uni-plovdiv.bg

Он-лайн обучение: www.fmi-plovdiv.org/evlm
www.fmi-plovdiv.org/evlm/DBbg - числени методи

Литература:

1. Бояджиев Д., Гочева С., Макрелов И., Попова Л. – Ръководство по числени методи – част 1, Издания: 2003, 2006, 2010.
2. Семерджиев Х., Боянов Б., Числени методи, ПУ.
3. Гочева-Илиева С., Въведение в система Mathematica, ЕксПрес, Габрово, 2009.

Съдържание:

2.1. Числено решаване на уравнения с едно неизвестно

Постановка на задачата: Дадено е уравнение от вида $f(x) = 0$, на което търсим корените.

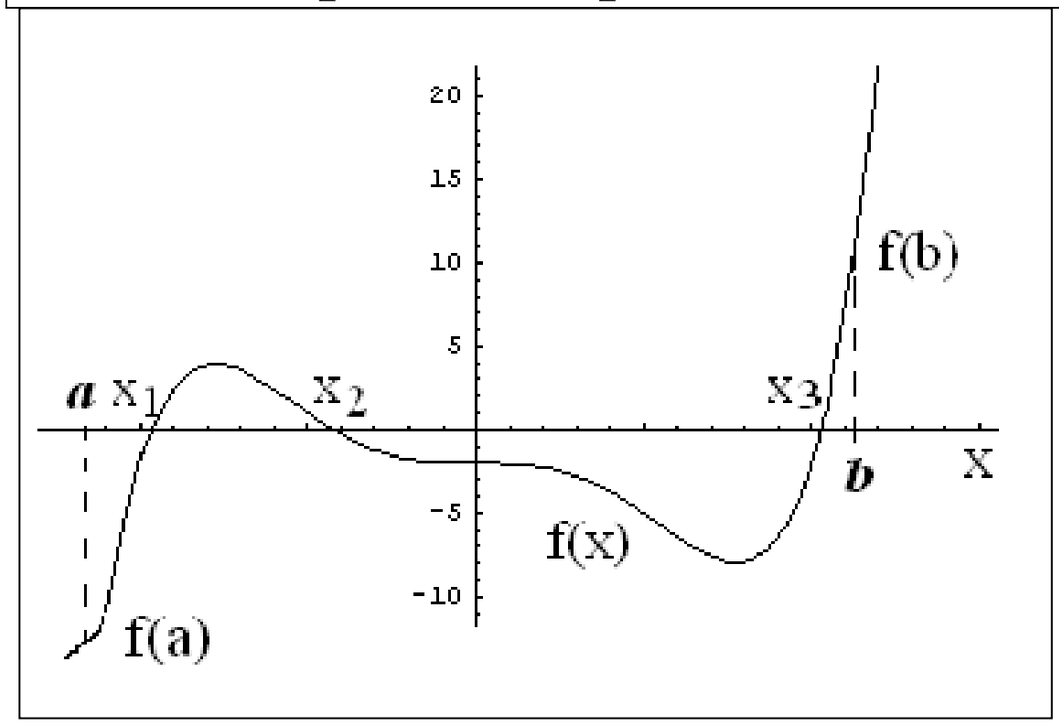
Въпроси	*Методи за решаване*
1. Съществуват ли корени, брой, вид (реални, комплексни), кратност (прости, двойни ...)?	Графични, аналитични емпирични
2. Местоположение на корените (локализация)	Графични, аналитични емпирични
3. Уточняване на корените с дадена точност	Числени методи

Целта ни е да намерим методи, които да работят за възможно по-голям клас от уравнения, т.е. функции $f(x)$. Различните методи са приложими при различни условия.

В нашия курс ще търсим предимно **реални** корени на уравнението $f(x) = 0$.

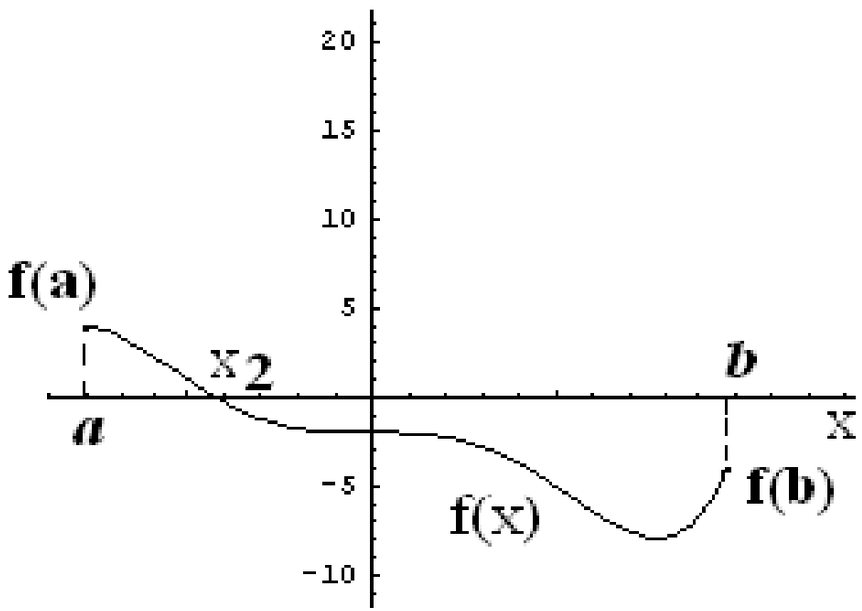
Основен факт от математическия анализ:

Нека $f(x)$ е реална функция, дефинирана и непрекъсната в някакъв краен затворен интервал [*a*,*b*]. Ако в краищата *a*, *b* функцията има различни знаци, т.е. произведеннието $f(a)f(b)<0$, то в интервала има поне един реален корен.	(*)



Фиг. 2.1.а

Очевидно функцията $f(x)$ е непрекъсната. Освен това тук $f(a)<0$, а $f(b)>0$, т.е. $f(a)f(b)<0$. В случая уравнението $f(x)=0$ има нечетен брой реални корени: x_1, x_2, x_3 .



Фиг. 2.1.б
От горната графика 2.1.а е намален първоначалният интервал $[a,b]$ така, че е спазено условието $f(a)f(b)<0$ и има локализиран само един реален корен (в случая x_2).

Локализацията на корен е намирането на $[a,b]$, в който съществува (поне) един реален корен. Локализирането е по-нестандартна процедура, но можем да посочим следните начини:

- **графичен**: построяваме графиката на функцията $f(x)$ и намираме приблизителната ѝ пресечна точка с оста Ox и интервал около нея,
- **аналитичен**: като използваме някои теореми от алгебрата, например в случая за корени на полиноми,
- **емпиричен**: с използване на априорна информация за физическия смисъл на решаваното уравнение и корените, на базата на експерименти.

Уточняването на корена представлява следното: При предварително намерен интервал $[a,b]$, съдържащ корена x^* и зададено положително число ε, наричано точност, да се намери точка $\tilde{x} \in [a,b]$, за която е в сила $\left|x^* - \tilde{x}\right| \le \varepsilon$. Тази точка $\tilde{x}$ се приема за приближение на корена x^*.

2.2. Метод на разполовяването за уравнението $f(x)=0$

Условия за прилагане на метода:

а) $f(x)$ е реална функция, дефинирана и непрекъсната в интервал $[a,b]$,

б) $f(a)f(b)<0$, т.е. функцията има различни знаци в a и b.

Съгласно факт (*) съществува поне един реален корен $x^* \in [a,b]$: $f(x^*)=0$.

Идея на метода на разполовяването (лов на лъвове):

Разделяме интервала на два равни подинтервала и за нов интервал избираме този подинтервал, в краищата на който функцията има различни знаци, т.е. е изпълнено условието б) $f(a)f(b)<0$. След това пак разделяме на два равни подинтервала и т.н. докато поредният интервал стане по-малък от зададената грешка.

Оценка на грешката и скорост на сходимост на разполовяването

Нека означим началния интервал с $[a_0,b_0]=[a,b]$ и работим без закръгляне. При първото разделяне на подинтервали, очевидно

получеиният подинтервал $[a_1, b_1]$ има два пъти по-малка дължина $\varepsilon_1 = (b-a)/2$ и $a_1 \le x^* \le b_1$, при второто разполовяване аналогично $\varepsilon_2 = (b-a)/2^2$ и т.н. След n итерации $\varepsilon_n = (b-a)/2^n$ и $a_n \le x^* \le b_n$.

За оценка на грешката получаваме: $|x^* - a_n| \le |b_n - a_n| = \varepsilon_n \le \frac{b-a}{2^n}$. (1)

Очевидно при $n \to \infty$ дясната част на (1) клони към нула и методът е сходящ със скорост на геометрична прогресия с частно $q = \frac{1}{2}$.

От (1) можем предварително да определим броя стъпки за достигане на желана точност ε:

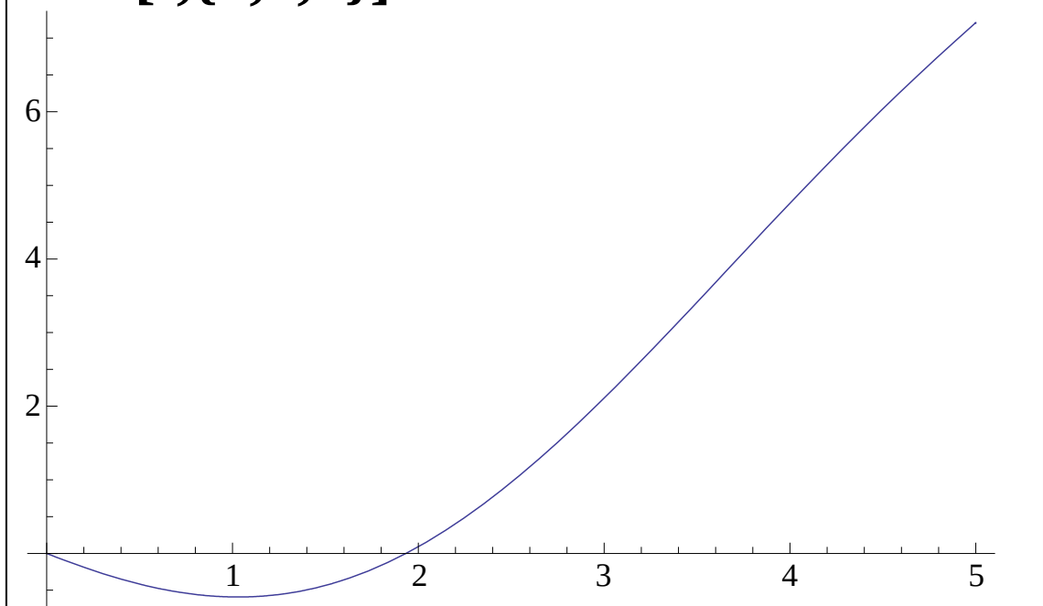
$$\frac{b-a}{2^n} \le \varepsilon \quad \Rightarrow \quad 2^n \ge \frac{b-a}{\varepsilon} \quad \Rightarrow \quad n \ge \log_2 \frac{b-a}{\varepsilon}. \qquad (2)$$

Например, при дължина на начален интервал $b-a=1$ и $\varepsilon = 0.001$ получаваме n =10, т.к. $2^{10} = 1024 \approx 1000$. При $\varepsilon = 0.000001 \Rightarrow n = 100$ и т.н.

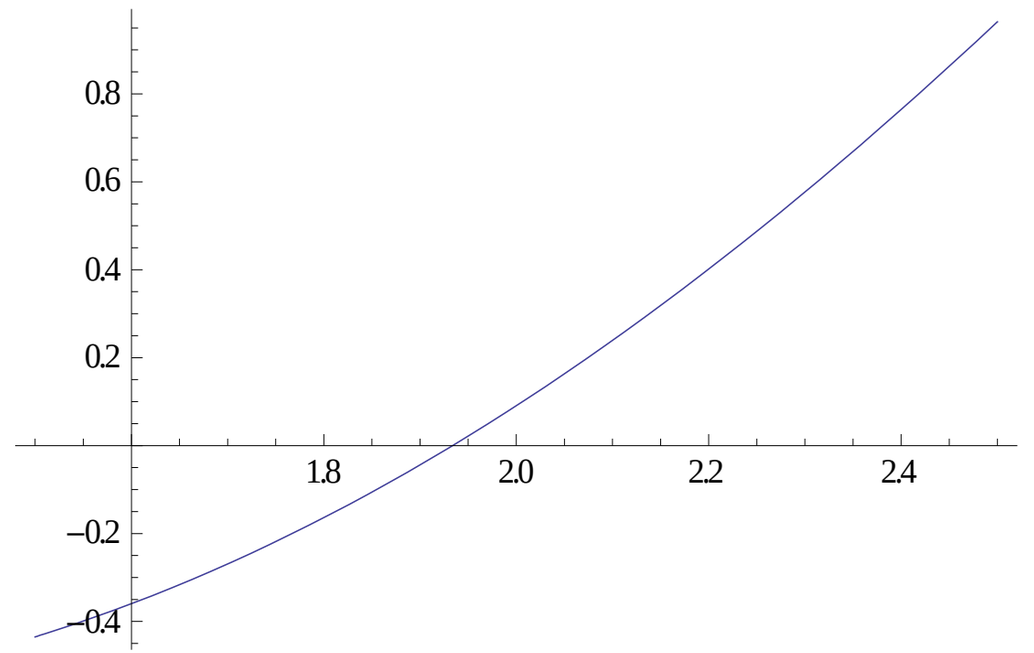
Пример. Да се реши уравнението: $x^2/4-\sin(x)=0$.

Решение. Започваме с графиката, за да локализираме корен.

f=x^2/4-Sin[x];
Plot[f,{x,0,5}]



Plot[f,{x,1.5,2.5}]



Виждаме, че функцията се анулира близо до x =2. Избираме например начален интервал $[a_0,b_0]=[1.8;2]$.

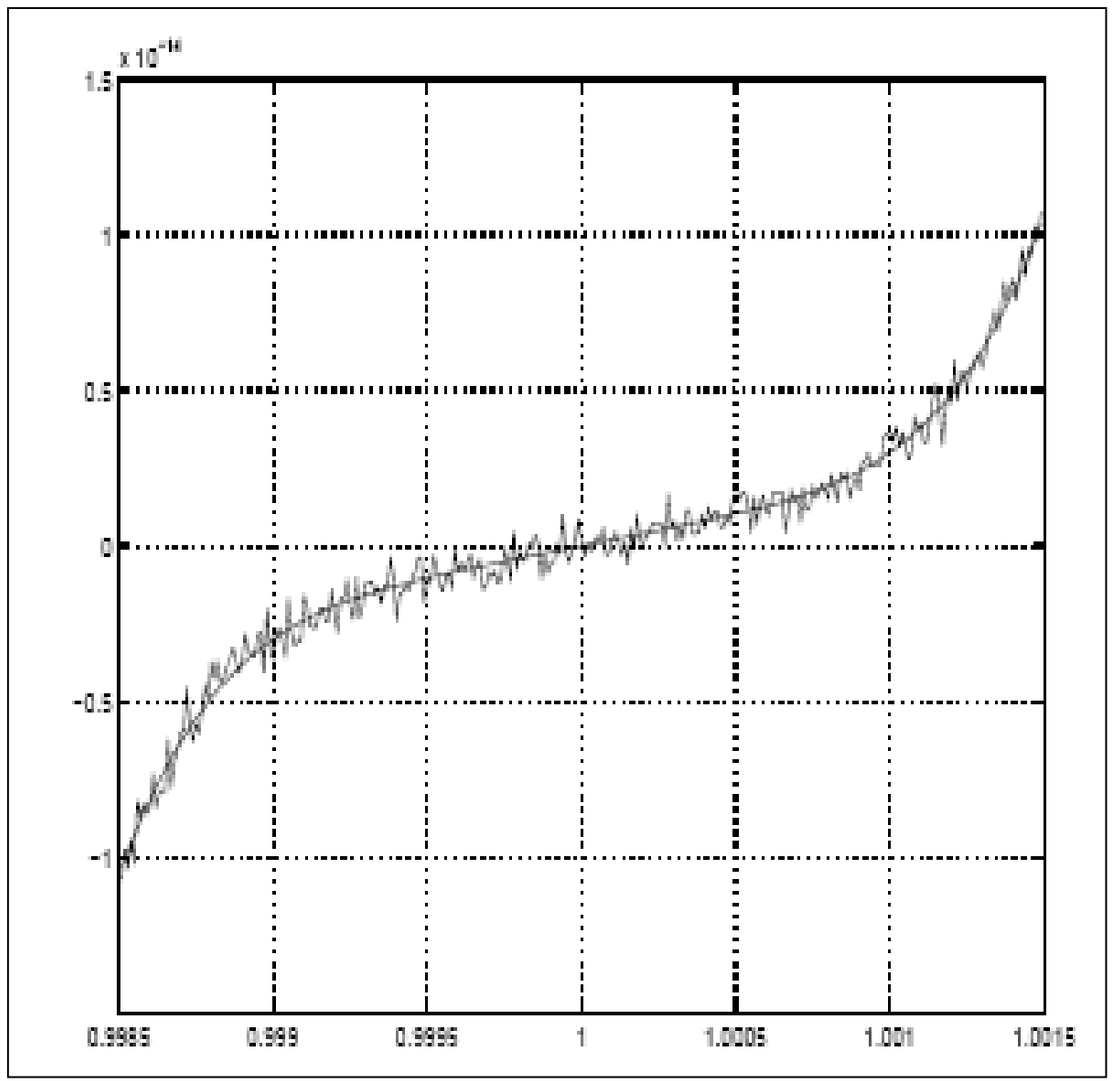
Последователно разделяме интервалите $[a_k, b_k]$ и проверяваме знака на функцията в средата m_k на последния избран интервал:

k	a_k	b_k	m_k **- среда**	$f(m_k)$
0	**1.8**	**2**	**1.9**	**<0**
1	**1.9**	**2**	**1.95**	**>0**
2	**1.9**	**1.95**	**1.925**	**<0**
3	**1.925**	**1.95**	**1.9375**	**>0**
4	**1.925**	**1.9375**	**1.93125**	**<0**
5	**1.93125**	**1.9375**	**1.934375**	**>0**

Така например след 5 итерации имаме $x^* \in [1,93125;\ 1,934375]$. Дължината на последния интервал е: (2-1.8) * 2^{-6}=0. 2 * 2^{-6} = 0. 003125. Или точността е 0. 003125. Приближената стойност на корена е: $\tilde{x} \in 1,932$.

2.3. Някои забележки по прилагане на метода на разполовяването

Трябва винаги да се отчитат и грешките от закръгляне при пресмятане стойностите на $f(x)$. Реалната графика е вярна с точността, с която закръгляме. Това е показано на съседната фигура, съдържаща графика на полином от 5-та степен с корен =1, като е закръгляно с точност 10^{-14}.



Затова методът на разполовяване се използва за грубо намиране на корена, който след това се уточнява с други методи.

2.4. Метод на хордите

Условия за прилагане на методите на хордите и на допирателните:

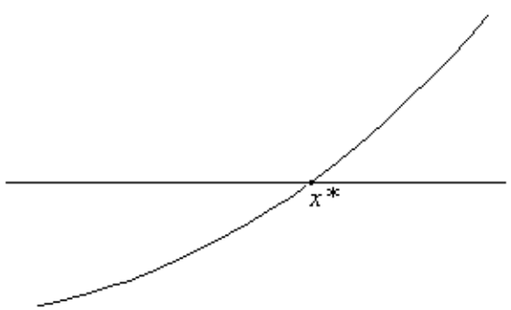
а) $f(x),\ f'(x),\ f''(x)$ **са дефинирани и непрекъснати в интервал [*a, b*],**

б) $f(a)f(b) < 0$ **- функцията има различни знаци в *a* и *b*,**

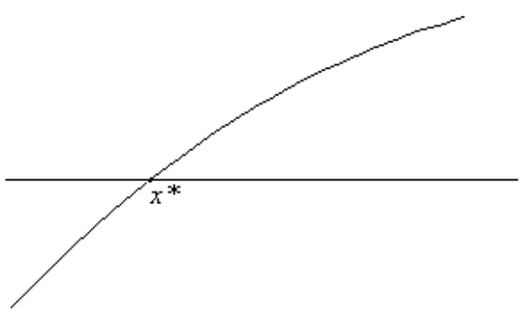
в) $f'(x),\ f''(x)$ **са с постоянни знаци в [*a, b*].**

Не е трудно да се съобрази, че тези условия гарантират съществуването на точно един реален корен x^* **в [*a,b*].**

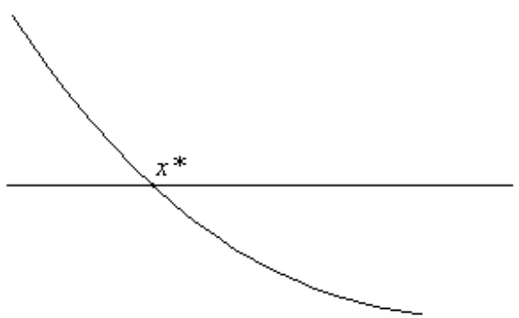
От в) производните не трябва да стават нула, т.е. имаме следните четири случая:



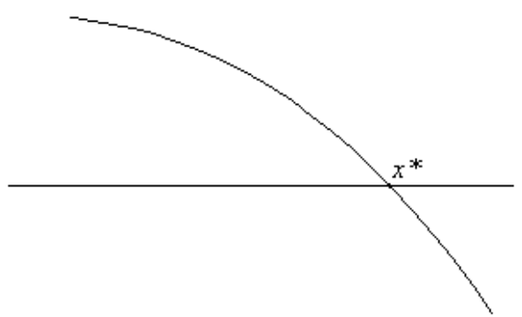
I) $f'(x) > 0,\ f''(x) > 0$



II) $f'(x) > 0,\ f''(x) < 0$

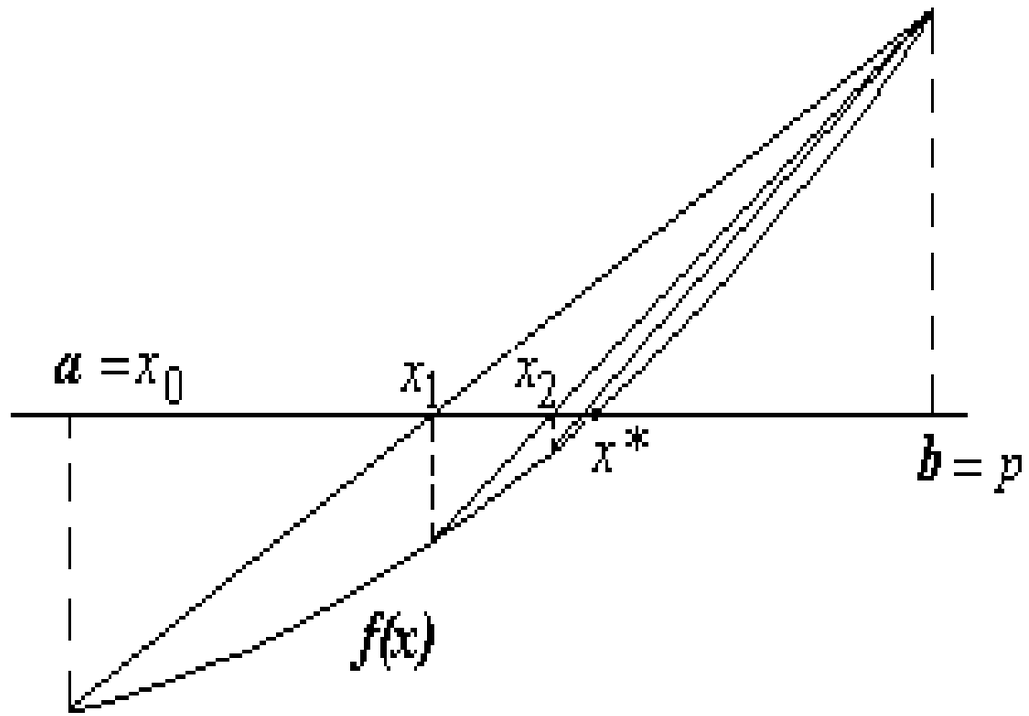


III) $f'(x) < 0,\ f''(x) > 0$



IV) $f'(x) < 0,\ f''(x) < 0$

<u>Идея на метода на хордите</u> (на примера от случай I): Построява се хордата през точките $(a, f(a))$ и $(b, f(b))$. Намира се пресечната точка с Ox, която се взима за първо приближение x_1 към търсения корен x^*. След това се построява хордата през $(x_1, f(x_1))$ и $(b, f(b))$ и се намира второто приближение x_2 и т.н. Вижда се, че ако се започне с начално приближение $x_0 = a$, то се приближаваме към търсения корен x^* отляво, а десният край на хордите е неподвижен. В избрания от нас случай на чертежа – неподвижната точка е $p = b$.



Избор на начално приближение x_0 **за метода на хордите – определя се например визуално за всеки от четирите случая, а именно:**

за случаите I) и IV) - $x_0 = a$, $p = b$**, а за случаите II) и III)-** $x_0 = b$, $p = a$**.**

Еквивалентното условие е:

да се избере x_0 **в онзи край на интервала [*a, b*], за който е изпълнено** $f(x_0).f'' < 0$**, другият край става неподвижна точка** p **.**

Извод на итерационната формула на метода на хордите

Построяваме хордата през поредната точка $(x_k, f(x_k))$ **и неподвижната точка** $(p, f(p))$ **като уравнение на права през две точки:** $\frac{y - f(x_k)}{f(p) - f(x_k)} = \frac{x - x_k}{p - x_k}$ **. За да намерим следващото приближение** x_{k+1} **като пресечна точка на хордата с** Ox**, полагаме тук** $y = 0,\ x = x_{k+1}$**.**

Получената итерационна формула на метода на хордите е:

$$x_{k+1} = x_k - \frac{f(x_k)}{f(x_k) - f(p)}(x_k - p); \qquad k = 0, 1, \ \ldots \tag{3}$$

Тази формула може да се запише в по-общ вид итерационна формула

$$x_{k+1} = \varphi(x_k), \ k = 0,1,\ldots. \tag{**}$$

Формула (**) по същество е явен едностъпков итерационен процес, тъй като при известна дясна част се изчислява директно лявата и се използва само предишното *k*-то приближение x_k при изчисляване на *k+1* –вото x_{k+1}.

2.5. Критерии за прекъсване на итерационния процес (достигане на точност)

Нека коренът се търси със зададена точност $\varepsilon > 0$. Прекратяването на изчисленията по формула (7) може да стане при изпълнение на поне едно от следните условия:

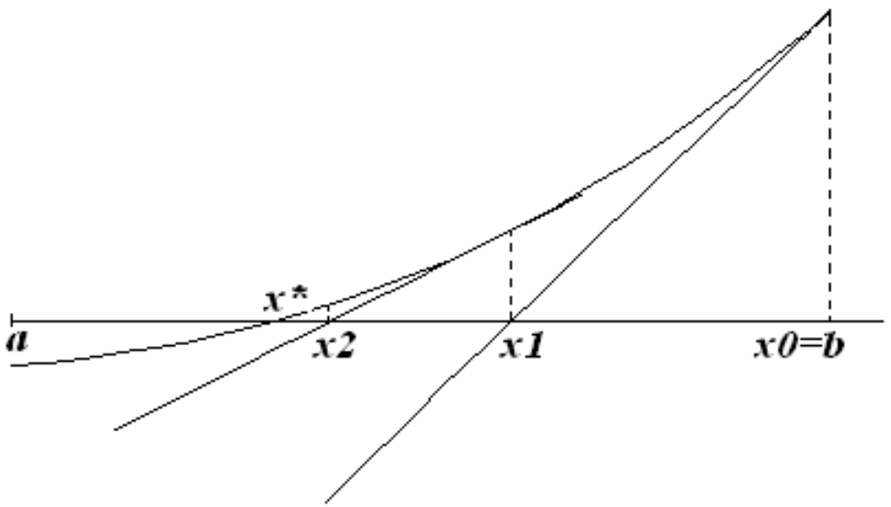
- **Когато $|x_{k+1} - x_k| < \varepsilon$ - по абсолютна грешка,**
- **Когато $\dfrac{|x_{k+1} - x_k|}{|x_k|} < \varepsilon$ - по относителна грешка,**
- **Когато $|f(x_k)| < \varepsilon$ - по стойност на функцията.**

Тези критерии са приложими и при другите итерационни методи за решаване на уравнения.

2.6. Метод на Нютон (метод на допирателните)

<u>Идея на метода на Нютон</u> (на примера от случай I):

От единия край на интервала ($x_0 = b$) се построява допирателната към $f(x)$ и за първо приближение x_1 към корена x^* се взима пресечната точка на допирателната с оста Ox. На следващата стъпка (итерация) се построява допирателната към $f(x)$ в точката x_1 и се получава x_2 като пресечна точка на допирателната с оста Ox и т.н.



Така получаваме безкрайната редица от приближения: $x_0, x_1, \ldots, x_k, \ldots$.

По-нататък с помощта на теоремата за неподвижната точка ще покажем, че:

а) редицата е сходяща и клони към корена на уравнението $f(x) = 0$**,**

б) скоростта на сходимост е квадратична.

Важен момент от метода на Нютон е изборът на началното приближение x_0. Тъй като $x_1, x_2, \cdots$ трябва да остават вътре в интервала [*a*, *b*], то от геометрични съображения не трудно да се съобрази, че:

за случаите I) и IV) $x_0 = b$, а за случаите II) и III) $x_0 = a$.

Еквивалентно условие е:

да се избере x_0 в онзи край на интервала [*a*, *b*], за който е изпълнено $f(x_0).f'' > 0$.

Извод на итерационната формула на Нютон

Вместо да търсим уравнението на допирателната, ще получим формулата по-елегантно. Достатъчно е да приведем изходното уравнение $f(x) = 0$ в т. нар. нормален вид $x = x - \frac{f(x)}{f'(x)}$, или по-общо

$$x = \varphi(x). \qquad \textbf{(4)}$$

Очевидно всяко решение на уравнението $f(x)=0$ е решение и на уравнение (4) и обратно (Защо?). От (4) заместваики вдясно началното приближение x_0 веднага изчисляваме x_1

$$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)}.$$

От полученото x_1 определяме аналогично $x_2 = x_1 - \frac{f(x_1)}{f'(x_1)}$ и т.н.

Ако сме намерили x_k, следващото приближение x_{k+1} се пресмята от формулата:

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}, \qquad (5)$$

или записано като общ итерационен явен едностъпков метод (виж и метод на хордите):

$$x_{k+1} = \varphi(x_k), \; k = 0,1,\ldots. \qquad (**)$$

Втори извод на формулата на Нютон – с уравнение на допирателната:

Уравнението на допирателната d към кривата $f(x)$ в точката $(x_0, f(x_0))$ е:

$$d: y = k.x + b, \quad (6)$$

където $k = f'(x_0) = tg(\alpha)$ е наклонът на правата d, която сключва ъгъл α с оста Ox. Тъй като искаме правата да минава през $(x_0, f(x_0))$, заместваме тази точка в общия вид (6) и получаваме $f(x_0) = k.x_0 + b$, откъдето $b = f(x_0) - k.x_0$. Така допирателната е: $d: y = f'(x_0).x + f(x_0) - f'(x_0).x_0$. В метода на Нютон следващото приближение x_1 е пресечната точка на допирателната с Ox. Намираме го като положим $y = 0,\ x = x_1$ и намираме

$$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)}. \quad (7)$$

В общия случай при известно x_k следващото приближение x_{k+1} е:

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}. \quad k = 0,1,\ldots. \text{ (Сравни с формула (5)).} \quad (8)$$

За да изследваме сходимостта на разгледаните методи на Нютон и хордите, както и за следващи методи от този тип ще докажем по-общата теорема за неподвижната точка. Тази теорема е валидна за много по-широки класове от математически задачи и е една от централните теореми в математиката.

2.7. Някои предварителни понятия от функционалния анализ

Определение 1. Едно множество X с елементи (точки) $x, y, z, \ldots$ се нарича **метрично пространство**, ако за всяка наредена двойка елементи $x, y \in X$ е съпоставено реално число $d(x, y)$ (разстояние между x, y, или метрика), такова че за всички $x, y, z \in X$:

1^0) $d(x, y) \geq 0$ и $d(x, y) = 0$ тогава и само тогава, когато $x = y$,

2^0) $d(x, y) = d(y, x)$ - симетричност,

3^0) $d(x, y) \leq d(x, z) + d(y, z)$ (неравенство на триъгълника).

Определение 2. Казваме, че една редица от точки $x^{(0)}, x^{(1)}, \ldots, x^{(k)}, \ldots \in X$ е **сходяща по метрика** в X, ако съществува точка $a \in X$, такава че числовата редица от разстоянията им до a е сходяща, т.е. $\lim\limits_{k \to \infty} d(x^{(k)}, a) = 0$.

Определение 3. Казваме, че една редица от точки $x^{(0)}, x^{(1)}, \ldots, x^{(k)}, \ldots \in X$ е **фундаментална редица (или редица на Коши)**, ако съществува границата $\lim\limits_{k,m \to \infty} d(x^{(m)}, x^{(k)}) = 0$.

Определение 4. Казваме, че X е **пълно метрично пространство**, ако всяка фундаментална редица е сходяща в X (границата й е точка от X).

Определение 5. Нека операторът (изображението) $\varphi : X \to X$ съпоставя на всяка точка $x \in X$ нова точка $\varphi(x) \in X$.

Казваме, че е φ е **свиващ оператор в X**, ако съществува реално число q, $0 < q < 1$, такова че за всеки $x, y \in X$ е изпълнено $d\left(\varphi(x), \varphi(y)\right) \le q.d(x, y)$.

Определение 6. Точката $x^* \in X$ се нарича **неподвижна точка** на свиващия оператор φ, ако съвпада с изображението си:

$$x^* = \varphi(x^*). \qquad (9)$$

Т.е. неподвижната точка е решение на уравнението (9).

Примери. 1) Реалните числа и комплексните числа са метрични пространства с метрика $d(x, y) = |x - y|$. 2) Векторните пространства са метрични, напр. пространството на n-мерните вектори $\vec{a} = (a_1, a_2, ..., a_n)$ е метрично с метрика $d(\vec{a}, \vec{b}) = \sqrt{(a_1 - b_1)^2 + ... + (a_n - b_n)^2}$ - евклидовото разстояние.

2.8. Теорема за неподвижната точка (н. т.)

Нека X е пълно метрично пространство и $\varphi(x)$ е свиващ оператор в X. Тогава φ притежава единствена неподвижна точка $x^* \in X$, която се получава като граница на итерационния процес:

$$x^{(k+1)} = \varphi\left(x^{(k)}\right),\ k = 0,1,\ldots, \qquad (**)$$

при $k \to \infty$ за произволен елемент $x^{(0)} \in X$, наричан начално приближение.

Доказателство: Ще покажем, че за итерационния процес (**) $\exists x^* \in X$ за което $\lim\limits_{k\to\infty} d(x^{(k)}, x^*) = 0$ и $x^* = \varphi(x^*)$.

(а) Нека $x^{(0)} \in X$ е произволно начално приближение. Щом φ по условие преобразува X в себе си, то от (**) всяко $x^{(k)}$ също принадлежи на X. От това, че φ е свиващ оператор съществува q, $0 < q < 1$, за което:

$$d(x^{(k+1)}, x^{(k)}) = d\left(\varphi(x^{(k)}), \varphi(x^{(k-1)})\right) \le q.d(x^{(k)}, x^{(k-1)}) \le q^2.d(x^{(k-1)}, x^{(k-2)}) \le \ldots$$

$$\le q^k.d(x^{(1)}, x^{(0)}) = \alpha q^k \qquad . \qquad (10)$$

Тук $\alpha = d(x^{(1)}, x^{(0)})$ е константа.

(б) Ще покажем, че (**) е фундаментална редица. Наистина, нека $m > k$. От неравенството на триъгълника в пълното метрично пространство X:

$$d(x^{(m)}, x^{(k)}) \le d(x^{(m)}, x^{(m-1)}) + d(x^{(m-1)}, x^{(m-2)}) + \ldots + d(x^{(k+1)}, x^{(k)}) \le ,$$

$$\le \alpha(q^{m-1} + q^{m-2} + \ldots + q^k) \le \alpha q^k (1 + q + q^2 + \ldots) \le q^k \frac{\alpha}{1-q} , \qquad (11)$$

където всяко d от първия ред на (11) е оценено с (10) и накрая е използвана формулата за сума от безкрайна геометрична прогресия с частно $q,\ 0 < q < 1$. Като направим граничен преход в (11) при $m, k \to \infty$ имаме $d(x^{(m)}, x^{(k)}) \to 0$, т.е. редицата (**) е фундаментална. Но X е пълно метрично пространство, значи всяка фунд. редица е сходяща в X. Следователно, съществува граница на (**), означаваме я с x^*, $x^* \in X$. Или:

$$\lim_{k \to \infty} d(x^{(k)}, x^*) = 0 . \qquad (12)$$

Освен това след граничен преход по m от (11) получаваме

$$d(x^{(k)}, x^*) \le \frac{\alpha}{1-q} q^k \ . \qquad (13)$$

(в) Ще покажем, че x^* е неподвижна точка. От неравенството на триъгълника, свойствата на φ и (13) имаме

$$d(x^*, \varphi(x^*)) \le d(x^*, x^{(k+1)}) + d(x^{(k+1)}, \varphi(x^*)) = d(x^*, x^{(k+1)}) + d(\varphi(x^{(k)}), \varphi(x^*)) \le$$

$$d(x^*, x^{(k+1)}) + q.d(x^{(k)}, x^*) \le q^{k+1} \frac{\alpha}{1-q} + q.q^k \frac{\alpha}{1-q} = q^{k+1} \frac{2\alpha}{1-q} \cdot$$

Но k е произволно, следователно след граничен преход $k \to \infty$ следва $d(x^*, \varphi(x^*)) = 0$, от което по свойство 1^0) намираме $x^* = \varphi(x^*)$.

(г) Единственост на неподвижната точка. Допускаме, че има две н.т. x^* и y^* и $x^* \ne y^*$. Тогава от $x^* = \varphi(x^*)$ и $y^* = \varphi(y^*)$

$$d(x^*, y^*) = d(\varphi(x^*), \varphi(y^*)) \le q.d(x^*, y^*) \ .$$

Последното е невъзможно, т.к. $0 < q < 1$. Следователно н.т. е единствена.

Забележка. Ние допълнително доказахме в (13), че скоростта на сходимост на итерационния процес (**) в най-лошия случай е равна на скоростта на сходимост на геометрична прогресия с частно q.

2.9. Теорема 2 за неподвижната точка (усилена формулировка)

При k=0 от (11) за всяко m имаме

$$d(x^{(m)}, x^{(0)}) \le \frac{\alpha}{1-q},$$

т.е. всички $x^{(m)}$, $m \to \infty$ се намират в околност на $x^{(0)}$, означаваме околността с $\Omega_\alpha = \{\forall x \in X : d(x, x^{(0)}) \le \frac{\alpha}{1-q}\}$.

От хода на доказателството на теоремата за неподвижната точка се вижда, че на практика всички резултати остават валидни в Ω_α. Следователно, във формулировката може да се уточни, че

$\varphi(x)$ е достатъчно да бъде свиващ оператор само в някаква околност Ω_α на избраното начално приближение $x^{(0)}$.

2.10. Следствие. Сходимост на итерационния процес за уравнението $f(x)=0$

Нека уравнението $f(x)=0$ е приведено по някакъв начин в нормален вид (3), т.е. $x=\varphi(x)$.

Както бе споменато по-горе, лесно се проверява, че множеството на реалните числа е метрично пространство с метрика $d(x,y)=|x-y|$ (всички свойства на метриката следват веднага от свойствата на модула).

Нека съществува производната $\varphi'(x)$ и е нека непрекъсната в околност на корена x^*. Тогава от теоремата на средните стойности

$$d(\varphi(x),\varphi(y))=|\varphi(x)-\varphi(y)|=|\varphi'(\xi)||x-y|\le \max|\varphi'(\xi)|.d(x,y).$$

Ако тук означим $q=\max|\varphi'(\xi)|$, то очевидно е вярно твърдението:

За сходимост на итерационния процес (*) съгласно теорема 2 за неподвижната точка е достатъчно в някаква околност на корена x^* да е изпълнено: $q=\max|\varphi'(\xi)|<1$. (14)

2.11. Сходимост на метода на Нютон (следствие от Теорема за н.т.)

Теорема. Нека в достатъчно малка околност на корена x^* са изпълнени условията за метода на Нютон, т.е. съществува единствен прост корен. Тогава методът (5) е сходящ при произволно начално приближение x_0 от тази област и скоростта на сходимост е квадратична.

Доказателство. Лесно се проверява, че $\varphi(x)$ от формулата на Нютон е свиващо изображение. Наистина имаме формулата: $x = x - \frac{f(x)}{f'(x)}$, т.е. $\varphi(x) = x - \frac{f(x)}{f'(x)}$. Изчисляваме $\varphi'(x) = \left(x - \frac{f(x)}{f'(x)} \right)' = 1 - \frac{f' \cdot f' - f \cdot f''}{(f')^2} = \frac{f \cdot f''}{(f')^2}$.

Тук последният член съдържа $f(x)$, което е нула при $x = x^*$, т.е.

$$\varphi'(x^*) = 0. \qquad (15)$$

Но $\varphi'(x)$ е непрекъсната, следователно в достатъчно малка околност на корена, $|\varphi'(x)| \approx 0$ и тогава съществува $q : 0 < q < 1$, така че да е в сила (14).

Освен това като развием в ред на Тейлър $\varphi(x_k)$ около точката x^* имаме:

$$x_{k+1} - x^* = \varphi(x_k) - \varphi(x^*) = (x_k - x^*)\varphi'(x^*) + \frac{1}{2}(x_k - x^*)^2 \varphi''(\xi),\ \xi \in (x_k, x^*).$$

Но $\varphi'(x^*) = 0$, откъдето

$$x_{k+1} - x^* = \frac{1}{2}(x_k - x^*)^2 \varphi''(\xi),$$

т.е. грешката от поредното *k+1*-во приближение е приблизително равна на квадрата на предишното *k* -то приближение. (Предполага се, че $\varphi''(x)$ не е много голяма в областта.). Това се нарича квадратична сходимост.

Например, ако за някое *k* имаме достигната точност 10^{-2}, на следващото приближение *k+1* грешката става 10^{-4}, а на *k+2*- 10^{-8} и т.н.

Поради горните качества на метода на Нютон, той е сред най-предпочитаните за решаване на уравнения. Лесно се обобщава и за случая от системи нелинейни уравнения.

Забележка. За грешката от метода на Нютон може да се докаже още, че е валидна оценката:

$$\left|x^* - x_n\right| \le \frac{M_2}{2m_1}\left|x_n - x_{n-1}\right|^2,$$

където $M_2 = \max\limits_{[a,b]}\left|f''(x)\right|$, $m_1 = \min\limits_{[a,b]}\left|f'(x)\right|$. Следователно, за достигане на зададена точност ε на n- то приближение, достатъчно е да е изпълнено

$$\frac{M_2}{2m_1}\left|x_n - x_{n-1}\right|^2 < \varepsilon.$$

2.12. Обобщение: Метод на Нютон за системи нелинейни уравнения:

Постановка на задачата:

Търсят се решенията на система от n нелинейни уравнения n неизвестни:

$$\begin{cases} f_1(x_1, x_2, ..., x_n) = 0 \\ f_2(x_1, x_2, ..., x_n) = 0 \\ \dots \\ f_n(x_1, x_2, ..., x_n) = 0 \end{cases}.$$

Извод на метода на Нютон.

Ако въведем векторен запис, ще получим: $\vec{F}(\vec{x}) = \vec{0}$,

където $\vec{F} = \begin{pmatrix} f_1 \\ f_2 \\ \dots \\ f_n \end{pmatrix}$, $\vec{x} = \begin{pmatrix} x_1 \\ x_2 \\ \dots \\ x_n \end{pmatrix}$ и $\vec{0}$ е нулевият вектор.

Въвеждаме якобиана:

$$jac(\vec{x}) = \begin{pmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \dots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \dots & \frac{\partial f_2}{\partial x_n} \\ \dots & & & \\ \frac{\partial f_n}{\partial x_1} & \frac{\partial f_n}{\partial x_2} & & \frac{\partial f_n}{\partial x_n} \end{pmatrix}$$

и нека в околност на някакъв корен съществува обратната матрица

$$J(\vec{x}) = jac^{-1}(\vec{x}) .$$

Тогава веднага получаваме следното обобщение на метода на Нютон:

$$\vec{x} = \vec{x} - J(\vec{x})\vec{F}(\vec{x})$$

и съответния му итерационен процес:

$$\vec{x}^{(k+1)} = \vec{x}^{(k)} - J(\vec{x}^{(k)})\vec{F}(\vec{x}^{(k)}), \; k = 0,1,2, \dots \qquad (16)$$

За прилагане на формула (16) е необходимо да се решават следните трудни подзадачи:

- локализиране на корен,
- намиране на подходящо начално приближение $\vec{x}^{(0)}$,
- обръщане на матрицата *jac* на всяка итерация *k*.

Програмирани методи за решаване на уравнения
Примери с Mathematica:
http://www.fmi-plovdiv.org/evlm/DBbg/numanmenu/programs_list.htm

Допълнителна литература по числени методи за решаване на уравнения:

Дж. Форсайт, М. Малкълм, К. Молър, *Компютърни методи за математически пресмятания*, Наука и изкуство, С., 1986.

Пловдивски университет „Паисий Хилендарски”
Факултет по математика и информатика
Катедра “Приложна математика и моделиране”

“Компютърни числени методи”
ЛЕКЦИЯ 3

Проф. д-р Снежана Гочева-Илиева, snow@uni-plovdiv.bg

Он-лайн обучение: www.fmi-plovdiv.org/evlm
www.fmi-plovdiv.org/evlm/DBbg - числени методи

Литература:

1. Бояджиев Д., Гочева С., Макрелов И., Попова Л. – Ръководство по числени методи – част 1, Издания: 2003, 2006, 2010.
2. Семерджиев Х., Боянов Б., Числени методи, ПУ.
3. Гочева-Илиева С., Въведение в система Mathematica, ЕксПрес, Габрово, 2009.

Съдържание

Числени методи за решаване на системи уравнения

1. Системи линейни уравнения, детерминанти и обратни матрици

Постановка на задачата. Търси се решението на системата линейни алгебрични уравнения

$$Ax = b,$$

където A е матрица с реални коефициенти с размерност $n \times n$, x – вектор на неизвестните, b - дясна част:

$$A = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & & a_{2n} \\ & & \cdots & \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{pmatrix}, \quad x = \begin{pmatrix} x_1 \\ x_2 \\ \dots \\ x_n \end{pmatrix}, \quad b = \begin{pmatrix} b_1 \\ b_2 \\ \dots \\ b_n \end{pmatrix} \neq \vec{0} \; .$$

В разгънат вид системата е:

$$
\left|\begin{array}{llll}
a_{11}x_1 & +a_{12}x_2 & \cdots & +a_{1n}x_n = b_1 \\
a_{21}x_1 & +a_{22}x_2 & & +a_{2n}x_n = b_2 \\
 & & \cdots & \\
 & & & \\
 & & & \\
a_{n1}x_1 & +a_{n2}x_2 & \cdots & +a_{nn}x_n = b_n
\end{array}\right. .
$$

Без ограничения на общността ще считаме, че задачата е коректна, т.е. съществува единствено решение на тази система. Както е добре известно от линейната алгебра, необходимо и достатъчно условие за това е детерминантата $\det A \neq 0$. Условието за устойчивост също подлежи на разглеждане.

В курса по Линейна алгебра са изучавани методът на Гаус и методът на Гаус-Жордан. Затова вместо тези методи в нашия курс ще покажем как се изчислява решението на задачата с помощта на система *Mathematica*.

Пример. Да се реши системата уравнения, да се намери детерминантата и обратната матрица.

$$\begin{cases} 2x_1 + x_2 + 2x_3 + 3x_4 = -1 \\ -2x_1 + 3x_2 + 2x_3 - 3x_4 = 2 \\ \quad + 4x_2 + 2x_3 + 3x_4 = 4 \\ x_1 + x_2 + x_3 + x_4 = 5 \end{cases}.$$

Записваме и изпълняваме следните няколко клетки в *Mathematica*:

```
Решаване на системи уравнения от вида A.x = b ,
намиране на детерминантата и обратната матрица на A.

a = {{2, 1, 2, 3}, {-2, 3, 2, -3}, {0, 4, 2, 3}, {1, 1, 1, 1}};
MatrixForm[a]
b = {-1, 2, 4, 5}; MatrixForm[b]
```

$$\begin{pmatrix} 2 & 1 & 2 & 3 \\ -2 & 3 & 2 & -3 \\ 0 & 4 & 2 & 3 \\ 1 & 1 & 1 & 1 \end{pmatrix}$$

$$\begin{pmatrix} -1 \\ 2 \\ 4 \\ 5 \end{pmatrix}$$

```
xx = LinearSolve[a, b]
```

$$\left\{\frac{169}{20}, \frac{73}{10}, -\frac{141}{20}, -\frac{37}{10}\right\}$$

```
(*  Проверка на решението със заместване *)
a.xx - b
```

```
{0, 0, 0, 0}
```

```
(* Детерминанта на матрицата A *)
Det[a]
```

```
20
```

```
(*  Обратна матрица *)
obr = Inverse[a]; MatrixForm[obr]
```

$$\begin{pmatrix} -\frac{11}{20} & -\frac{3}{20} & -\frac{1}{5} & \frac{9}{5} \\ -\frac{7}{10} & -\frac{1}{10} & \frac{1}{5} & \frac{6}{5} \\ \frac{19}{20} & \frac{7}{20} & -\frac{1}{5} & -\frac{6}{5} \\ \frac{3}{10} & -\frac{1}{10} & \frac{1}{5} & -\frac{4}{5} \end{pmatrix}$$

Отг.: Точното решение е: $x_1 = \frac{169}{20}$, $x_2 = \frac{73}{10}$, $x_3 = -\frac{141}{20}$, $x_2 = -\frac{37}{10}$,
детерминантата е =20, обратната матрица – получената по-горе.

2. Метод на прогонването за тридиагонални системи линейни уравнения

Постановка на задачата. Търси се решението на системата линейни алгебрични уравнения

$$Ax = d\,, \tag{1}$$

където A е тридиагонална матрица, x – вектор на неизвестните, d - дясна част:

$$A = \begin{pmatrix} b_1 & c_1 & 0 & \dots & & & 0 \\ a_2 & b_2 & c_2 & 0 & \dots & & 0 \\ & & \dots & & & & \\ 0 & \dots & a_k & b_k & c_k & \dots & 0 \\ & & & & \dots & & \\ 0 & \dots & & 0 & a_{n-1} & b_{n-1} & c_{n-1} \\ 0 & \dots & & & o & a_n & b_n \end{pmatrix}, \quad x = \begin{pmatrix} x_1 \\ x_2 \\ \\ x_k \\ \\ \dots \\ x_n \end{pmatrix}, \quad d = \begin{pmatrix} d_1 \\ d_2 \\ \\ d_k \\ \\ \dots \\ d_n \end{pmatrix} \neq 0 \quad . \tag{2}$$

В разгънат вид системата е:

$$
\left|\begin{array}{ccccccccc}
b_1x_1 & +c_1x_2 & & & & & & = & d_1 \\
a_2x_1 & +b_2x_2 & +c_2x_3 & & & & & = & d_2 \\
 & a_3x_2 & +b_3x_3 & +c_3x_4 & & & & = & d_3 \\
 & & & ..... & & & & & ..... \\
 & & & a_kx_{k-1} & +b_kx_k & +c_kx_{k+1} & & = & d_k \\
 & & & & ..... & & & & ..... \\
 & & & & a_{n-1}x_{n-2} & +b_{n-1}x_{n-1} & +c_{n-1}x_n & = & d_{n-1} \\
 & & & & & a_nx_{n-1} & +b_nx_n & = & d_n
\end{array}\right., \qquad (3)
$$

Тази система може да се реши по който и да е от съществуващите методи за системи линейни алгебрични уравнения: метод на Гаус, метод на Гаус-Жордан, проста итерация и др. Но тъй като тя е от специален вид, за нейното

решаване се прилагат специални ефективни методи, между които е методът на прогонването. Методът спада към групата на точните методи, т.е. теоретично, при работа с точни числа (без закръгляне), след краен брой аритметични операции се получава точното решение. В случая на реални системи обаче, като правило броят на уравненията n е много голям и неминуемо се работи със закръгляне на междинните резултати до определен знак след десетичната запетая. Това може да доведе до натрупване на грешка, независимо, че методът е точен.

3. Изчислителна схема на метода на прогонването

Целта е тридиагоналната система (3) да се преобразува в двудиагонална. Полагаме:

$$x_k = \alpha_k x_{k+1} + \beta_k \ , \quad k = \overline{1,n-1}, \tag{4}$$

където α_i, β_i са търсени прогонъчни коефициенти.

От първото уравнение на (3) имаме

$k=1$: $b_1 x_1 + c_1 x_2 = d_1$, откъдето сравнено с (4) намираме $x_1 = -\frac{c_1}{b_1} x_2 + \frac{d_1}{b_1}$, т.е. $\alpha_1 = -\frac{c_1}{b_1}$, $\beta_1 = \frac{d_1}{b_1}$.

По-нататък, ако сме получили формулата за някое *k-1*:

$x_{k-1} = \alpha_{k-1} x_k + \beta_{k-1}$, заместваме го в *k*-тото уравнение на (3)

$a_k x_{k-1} + b_k x_k + c_k x_{k+1} = d_k$ и получаваме

$a_k(\alpha_{k-1}x_k+\beta_{k-1})+b_kx_k+c_kx_{k+1}=d_k$, откъдето изразяваме x_k:

$x_k(a_k\alpha_{k-1}+b_k)+c_kx_{k+1}=d_k-a_k\beta_{k-1}$.

Ако множителят пред x_k не е нула, то:

$$x_k=-\frac{c_k}{b_k+a_k\alpha_{k-1}}x_{k+1}+\frac{d_k-a_k\beta_{k-1}}{b_k+a_k\alpha_{k-1}},$$ което е във вида (4), където

$$\alpha_k=-\frac{c_k}{b_k+a_k\alpha_{k-1}},\qquad \beta_k=\frac{d_k-a_k\beta_{k-1}}{b_k+a_k\alpha_{k-1}}.$$

Накрая за $k=n$ от това представяне при $x_{n+1}=0$ формално намираме: $x_n=\beta_n$.

Окончателно изчислителната схема на прогонването е:

Прав ход: (изчисляване на прогонъчните коефициенти) по формулите:

$$\alpha_1 = -\frac{c_1}{b_1},\quad \beta_1 = \frac{d_1}{b_1},\quad \alpha_k = -\frac{c_k}{b_k + a_k\alpha_{k-1}},\qquad \beta_k = \frac{d_k - a_k\beta_{k-1}}{b_k + a_k\alpha_{k-1}},\qquad k = \overline{2,n} \tag{5}$$

Обратен ход: $x_n = \beta_n \quad x_k = \alpha_k x_{k+1} + \beta_k \quad k = n-1, n-2, \ldots, 1.$ (6)

Определение. Прогонъчните формули (5) (методът на прогонването) ще наричаме устойчиви, ако

$$|\alpha_k| \le 1, \ k = \overline{1,n} \qquad (7)$$

Забележка. Очевидно е, че ако това не е изпълнено, то от формула (6) напр. при някаква малка грешка ε да речем за $\tilde{x}_n = \beta_n + \varepsilon$ ще получаваме:

$$x_k = \alpha_k x_{k+1} + \beta_k = \alpha_k \alpha_{k+1} x_{k+2} + \ldots + \beta_k = \ldots = \alpha_k \alpha_{k+1} \ldots \alpha_n \tilde{x}_n + \ldots .$$

Вижда се, че водещият член $\alpha_k \alpha_{k+1} \ldots \alpha_n$ ще се умножи по грешката и ако не е по-малък от 1, то грешката на резултата е неуправляема.

Сега ще намерим условията, които могат да гарантират избягването на тази ситуация.

4. Теорема: Достатъчно условие за устойчивост на метода на прогонването

Нека за коефициентите на системата е изпълнено:

$|b_k| \geq |a_k| + |c_k|,\ c_k \neq 0,\ k = \overline{1,n}$. (8)

Тогава съществува единствено решение на системата (1), което е и устойчиво, т.е. малки грешки в началните данни водят до малки грешки в резултата.

Доказателство. Става по метода на непълната математическа индукция. Очевидно условието за устойчивост (7) е вярно при $k=1$:

$$|\alpha_1| = \left|-\frac{c_1}{b_1}\right| \leq 1 .$$

Допускаме, че твърдението е вярно за *k-1*. За следващото *k* имаме:

Знаменател от (5): $|b_k + a_k \alpha_{k-1}| \geq |b_k| - |a_k|.|\alpha_{k-1}| \geq |b_k| - |a_k| \geq |c_k| > 0$. (9)

Тогава: $|\alpha_k| = \left| -\frac{c_k}{b_k + a_k \alpha_{k-1}} \right| \le \frac{|-c_k|}{|b_k + a_k \alpha_{k-1}|} \le \frac{|c_k|}{|c_k|} = 1$, или $|\alpha_k| \le 1$. Доказахме условието за устойчивост и при k. Следователно по индукция условието е изпълнено за всяко k. В частност тъй като знаменателят от (9) е ненулев, то винаги е възможно делението, а оттам - винаги съществува решение по формулите на прогонването при поставените условия на теоремата.

Забележка 1. Условието (8) може да се прецизира. Не е трудно да се съобрази, че всъщност то означава преобладаващ главен диагонал на матрицата *A*.

Забележка 2. Възможни са и случаи, когато методът на прогонването работи и без да е изпълнено условието за устойчивост, т.к. то е само достатъчно, но не и необходимо.

5. Преимущества на метода на прогонването. Пример

- Бърз: Лесно се пресмята от формули (5), (6), че са необходими $3n$ операции събиране/изваждане и $5n$ операции умножение/деление (знаменателят се изчислява само веднъж). Или всичко $8n$ аритметични операции - $O(n)$.
- Малко памет: Ясно е, че са достатъчни само едномерни масиви за $a_k, b_k, c_k, d_k, \alpha_k, \beta_k, x_k$. При повторно ползване е нужна памет само $4n$.
- Лесно се програмира.
- Условията за устойчивост са прости за проверка.

Реално методът се използва за решаване на системи с n=1000000 и повече уравнения, т.е. изисква 1 секунда машинно време. Дори и за по-големи n работи в реално време.

Пример. Проверете дали метода на прогонването е устойчив (не натрупва грешки) за системата:

$$\left|\begin{array}{rrrrcr} -4x_1 & +2x_2 & & & = & 1 \\ x_1 & +3x_2 & -x_3 & & = & 3 \\ & x_2 & -7x_3 & -2x_4 & = & -1 \\ & & -9x_3 & +10x_4 & = & 0 \end{array}\right. .$$

Решение: Имаме по редове: $|-4| \geq |2|$, $|3| \geq |1| + |-1|$, $|-7| \geq |1| + |-2|$ и $|10| \geq |-9|$.

Етапи за решаване на тридиагонална система алгебрични уравнения по метода на прогонването:

Провежда се на три етапа:

1. Проверка на условието за устойчивост (8),
2. Изчисляване на прогонъчните коефициенти по формули (5),
3. Изчисляване на неизвестните в обратен ред по формули (6).

За да решим горната тридиагонална система, съставяме и изпълняваме следния код на *Mathematica*:

Зареждаме векторите: a, b, c, d, α, β и някаква стойност на x (за да има списък, всеки вектор с дължина n):

```
a = {0, 1, 1, -9};
b = {-4, 3, -7, 10};
c = {2, -1, -2, 0};
d = {1, 3, -1, 0};
α = a;
β = a;
x = a;
n = Length[a];
```

$$\alpha_{[1]} = -\frac{c_{[1]}}{b_{[1]}};\ \beta_{[1]} = \frac{d_{[1]}}{b_{[1]}};$$

$$\text{For}\left[i = 2,\ i \le n,\ i{+}{+},\ z = a_{[i]} * \alpha_{[i-1]} + b_{[i]};\right.$$

$$\left.\alpha_{[i]} = -\frac{c_{[i]}}{z};\ \beta_{[i]} = \frac{d_{[i]} - a_{[i]} * \beta_{[i-1]}}{z}\right]$$

```
Print["    α= ", α, "    β= ", β]
(*  Край прав ход *)
```

$$\alpha = \left\{\frac{1}{2}, \frac{2}{7}, -\frac{14}{47}, 0\right\} \quad \beta = \left\{-\frac{1}{4}, \frac{13}{14}, \frac{27}{94}, \frac{243}{1192}\right\}$$

$$x_{[n]} = \beta_{[n]};$$

$$\text{For}[i = n - 1,\ i \ge 1,\ i{-}{-},$$

$$x_{[i]} = \alpha_{[i]} * x_{[i+1]} + \beta_{[i]}]$$

```
Print["    x= ", x]    (*  Край обратен ход  *)
```

Отг.: $x = \left\{\frac{147}{596}, \frac{148}{149}, \frac{135}{596}, \frac{243}{1192}\right\}$.

6. Итерационни методи: Метод на простата итерация

Постановка на задачата. Търси се приближеното решение на системата линейни алгебрични уравнения

$$Ax = b, \tag{10}$$

където A е матрица с реални коефициенти с размерност $n \times n$, x – вектор на неизвестните, b - дясна част:

$$A = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & & a_{2n} \\ & & \cdots & \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{pmatrix}, \quad x = \begin{pmatrix} x_1 \\ x_2 \\ \cdots \\ x_n \end{pmatrix}, \quad b = \begin{pmatrix} b_1 \\ b_2 \\ \cdots \\ b_n \end{pmatrix} \neq \vec{0} \quad . \tag{11}$$

В разгънат вид системата е:

$$\left|\begin{array}{llll} a_{11}x_1 & +a_{12}x_2 & \cdots & +a_{1n}x_n = b_1 \\ a_{21}x_1 & +a_{22}x_2 & & +a_{2n}x_n = b_2 \\ & & \cdots & \\ & & & \\ a_{n1}x_1 & +a_{n2}x_2 & \cdots & +a_{nn}x_n = b_n \end{array}\right. . \qquad (12)$$

За да построим процеса на проста итерация, аналогично на итерационните методи за едно уравнение, изходният вид на системата трябва да е от тип $x = \varphi(x)$ с процес $x^{(k+1)} = \varphi(x^{(k)})$, $k = 0,1,...$

Най-често привеждането към такъв вид става така: първото уравнение се дели на a_{11} и всички останали членове се прехвърлят отдясно, второто уравнение се дели на a_{22} и т.н. Така от всяко уравнение се изразява неизвестното x_i от i-тия ред на системата.

Стигаме до вида:

$$x = Cx + d \tag{13}$$

и итерационен процес, наречен проста итерация или метод на Якоби:

$$x^{(k+1)} = Cx^{(k)} + d\,,\ k = 0,1,\ldots, \tag{14}$$

където $x^{(0)}$ е произволно начално приближение.

От (4) последователно изчисляваме редицата от приближения:

$$x^{(0)}, x^{(1)}, \ldots, x^{(k)}, \ldots \tag{15}$$

Както при итерационните методи за едно уравнение, формулата (14) определя явен процес, т.е. заместваме известна стойност $x^{(k)}$ в дясната част и изчисляваме следващото приближение $x^{(k+1)}$. Повтаряме процеса и получаваме последователно редицата (15).

7. Критерии за прекъсване на итерационния процес

за получаване на решение с желана точност ε (поне един от следните критерии):

1) $\left|x_i^{(k+1)} - x_i^{(k)}\right| < \varepsilon$, $i = 1, \ldots, n$ - по абсолютна грешка,

2) $\dfrac{\left|x_i^{(k+1)} - x_i^{(k)}\right|}{\left|x_i^{(k)}\right|} < \varepsilon$, $i = 1, \ldots, n$ - по относителна грешка.

Може и със заместване в уравненията.

8. Пример за метода на проста итерация

Пример. Да се реши по метода на проста итерация системата:

$$\left|\begin{array}{llll} 10x_1 & +x_2 & -3x_3 & =3 \\ x_1 & +5x_2 & -2x_3 & =5 \\ -x_1 & +x_2 & -5x_3 & =-14 \end{array}\right. . \qquad (16)$$

С директна проверка се вижда, че точното решение е: $x = (1,2,3)$.

Решение. 1) Привеждане към вид, удобен за итерация:

Делим първото уравнение на 10, второто на 5 и третото на (-5) и изразяваме неизвестните от главния диагонал:

$$\left|\begin{array}{llll} x_1 = & & -0.1x_2 & +0.3x_3 & +0.3 \\ x_2 = & -0.2x_1 & & +0.4x_3 & +1 \\ x_3 = & -0.2x_1 & +0.2x_2 & & +2.8 \end{array}\right. .$$

Тук матрицата C и дясната част в съответствие с (13)-(14) са:

$$C = \begin{pmatrix} 0 & -0.1 & 0.3 \\ -0.2 & 0 & 0.4 \\ -0.2 & 0.2 & 0 \end{pmatrix}, \quad d = \begin{pmatrix} 0.3 \\ 1 \\ 2.8 \end{pmatrix}. \tag{17}$$

2) Записваме формулите за пресмятане по проста итерация:

$$\begin{cases} x_1^{(k+1)} = & -0.1x_2^{(k)} & +0.3x_3^{(k)} & +0.3 \\ x_2^{(k+1)} = & -0.2x_1^{(k)} & +0.4x_3^{(k)} & +1 \\ x_3^{(k+1)} = & -0.2x_1^{(k)} & +0.2x_2^{(k)} & +2.8 \end{cases}. \tag{18}$$

3) Избираме начално приближение $x^{(0)} = (0,0,0)^T$. Заместваме го отдясно (при k=0) и получаваме 1-во приближение:

$x_1^{(1)} = (-0.1).0 + (0.3).0 + 0.3 = 0.3$, $x_2^{(1)} = (-0.2).0 + (0.4).0 + 1 = 1$,

$x_3^{(1)} = (-0.2).0 + (0.2).0 + 2.8 = 2.8$, т.е. $x^{(1)} = (0.3,1,2.8)^T$.

За второ приближение заместваме полученото $x^{(1)}$ вдясно на (18):

$x_1^{(2)} = (-0.1).1 + (0.3).2.8 + 0.3 = 1.04$, $x_2^{(2)} = (-0.2).(0.3) + (0.4).(2.8) + 1 = 2.06$,

$x_3^{(2)} = (-0.2).(0.3) + (0.2).1 + 2.8 = 2.94$, т.е. $x^{(2)} = (1.04, 2.06, 2.94)^T$.

На следващата итерация заместваме тези стойности отдясно и т.н. Получаваме следната таблица от приближения:

k	$x_1^{(k)}$	$x_2^{(k)}$	$x_3^{(k)}$	$\left\|x_1^{(k+1)} - x_1^{(k)}\right\|$	$\left\|x_2^{(k+1)} - x_2^{(k)}\right\|$	$\left\|x_3^{(k+1)} - x_3^{(k)}\right\|$
0	0	0	0			
1	0.3	1	2.8	0.3	1	2.8
2	1.04	2.06	2.94	0.74	1.06	0.14
3	0.976	1.968	3.004	0.064	0.092	0.064
4	1.0044	2.0064	2.9984	0.0284	0.0384	0.0056
5	0.99888	1.99848	3.0004	0.00552	0.00792	0.002
6	1.00027	2.00038	2.99992	0.00139	0.00190	0.00048
7	0.99994	1.99991	3.00002	0.000334	0.000108	0.000102

Отговор с точност $\varepsilon = 0.001$, k=7 (закръглен): $x \approx (1.000, 2.000, 3.000)^T$.

Същата задача с *Mathematica* може да се реши така:

А) Зареждане на приведената матрица и дясна част по формула (17):

```
(* Проста итерация *)
c = { {0., -0.1, 0.3}, {-0.2, 0., 0.4}, {-0.2, 0.2, 0.} };
d = {0.3, 1., 2.8}; MatrixForm[c]
MatrixForm[d]
```

$$\begin{pmatrix} 0. & -0.1 & 0.3 \\ -0.2 & 0. & 0.4 \\ -0.2 & 0.2 & 0. \end{pmatrix}$$

$$\begin{pmatrix} 0.3 \\ 1. \\ 2.8 \end{pmatrix}$$

Б) Зареждаме клетка за точността 0.001 и начално приближение $x^{(0)}$, (стандартно =0), след което в цикъл уточняваме по формула (14):

```
eps = 0.001;
x = {0., 0., 0.}
n = Length[x]; r = ∞; k = 0;
While[r ≥ eps,
 {k++; xnew = c.x + d; r = Max[Abs[xnew - x]]; x = xnew;
  Print[" итерация k=", k, "  x= ", x, "  текуща грешка= ", r]}]
```

```
{0., 0., 0.}
итерация k=1  x= {0.3, 1., 2.8}  текуща грешка= 2.8
итерация k=2  x= {1.04, 2.06, 2.94}  текуща грешка= 1.06
итерация k=3  x= {0.976, 1.968, 3.004}  текуща грешка= 0.092
итерация k=4  x= {1.0044, 2.0064, 2.9984}  текуща грешка= 0.0384
итерация k=5  x= {0.99888, 1.99848, 3.0004}  текуща грешка= 0.00792
итерация k=6  x= {1.00027, 2.00038, 2.99992}  текуща грешка= 0.001904
итерация k=7  x= {0.999938, 1.99991, 3.00002}  текуща грешка= 0.0004704
```

Отг.: $x_1 \approx 0.999938 \approx 1$, $x_2 \approx 0.99991 \approx 2$, $x_2 \approx 3.00002 \approx 3$ с точност 0.001.

9. Метод на Гаус-Зайдел. Пример

Постановка на задачата – както в метода на простата итерация.

Аналогично системата трябва да бъде предварително приведена във вида (13) - $x = Cx + d$.

Дотук приликата между методите завършва, тъй като схемата на итерациите при метода на Гаус-Зайдел е друга. Сега за изчисляване на всяко следващо приближение се използват най-новите, току-що получени приближения. По-точно формулите са

$$
\begin{array}{llllll}
x_1^{(k+1)} = & c_{11}x_1^{(k)} & +c_{12}x_2^{(k)} \ldots & +c_{1n}x_n^{(k)} & +d_1 \\
x_2^{(k+1)} = & c_{21}x_1^{(k+1)} & +c_{22}x_2^{(k)} \ldots & +c_{2n}x_n^{(k)} & +d_2 \\
x_3^{(k+1)} = & c_{31}x_1^{(k+1)} & +c_{32}x_2^{(k+1)} \ldots & +c_{3n}x_n^{(k)} & +d_3 \\
 & & & \ldots & \\
x_n^{(k+1)} = & c_{n2}x_1^{(k+1)} & +c_{n2}x_2^{(k+1)} \ldots & +c_{3n}x_n^{(k)} & +d_n
\end{array}
.
$$

Пример. Да се реши системата (16) по метода на Зайдел.

Решение. 1) е същото.

2) Записваме формулите по Зайдел:

$$\left|\begin{array}{llll} x_1^{(k+1)} = & & -0.1x_2^{(k)} & +0.3x_3^{(k)} & +0.3 \\ x_2^{(k+1)} = & -0.2x_1^{(k+1)} & & +0.4x_3^{(k)} & +1 \\ x_3^{(k+1)} = & -0.2x_1^{(k+1)} & +0.2x_2^{(k+1)} & & +2.8 \end{array}\right. . \qquad (19)$$

3) Избираме $x^{(0)} = (0,0,0)^T$. Заместваме го отдясно (при k=0) и получаваме 1-во приближение: $x_1^{(1)} = (-0.1).0 + (0.3).0 + 0.3 = 0.3$. За $x_2^{(1)}$ на мястото на x_1 заместваме вече получената стойноср 0.3 и намираме $x_2^{(1)} = (-0.2).(0.3) + (0.4).0 + 1 = 0.94$. Тези две нови стойности веднага заместваме в третото уравнение и получаваме $x_3^{(1)} = (-0.2).(0.3) + (0.2).(0.94) + 2.8 = 2.929$. Така $x^{(1)} = (0.3, 0.94, 2.928)^T$. Аналогично работим нататък по (19).

Резултатите нанасяме в следната таблица от приближения:

k	$x_1^{(k)}$	$x_2^{(k)}$	$x_3^{(k)}$	$\left\|x_1^{(k+1)} - x_1^{(k)}\right\|$	$\left\|x_1^{(k+1)} - x_1^{(k)}\right\|$	$\left\|x_1^{(k+1)} - x_1^{(k)}\right\|$
0	0	0	0			
1	0.3	0.94	2.928	0.3	0.94	2.928
2	1.0844	1.95432	2.97398	0.7844	1.01432	0.04598
3	0.99968	1.99902	2.99870	0.08472	0.04470	0.02472
4	1.00058	1.99936	2.99976	0.00090	0.00034	0.00106

Отговор за $\varepsilon = 0.001$, k=4, (със закръгляне): $x \approx (1.000, 1.999, 3.000)^T$.

Решени примери може да намерите на:
http://www.fmi-plovdiv.org/evlm/DBbg/numanmenu/programs_list.htm

Пловдивски университет „Паисий Хилендарски”
Факултет по математика и информатика
Катедра “Приложна математика и моделиране”

“Компютърни числени методи”
ЛЕКЦИЯ 4

Проф. д-р Снежана Гочева-Илиева, snow@uni-plovdiv.bg

Он-лайн обучение: www.fmi-plovdiv.org/evlm
www.fmi-plovdiv.org/evlm/DBbg - числени методи

Литература:

1. Бояджиев Д., Гочева С., Макрелов И., Попова Л. – Ръководство по числени методи – част 1, Издания: 2003, 2006, 2010.
2. Семерджиев Х., Боянов Б., Числени методи, ПУ.
3. Гочева-Илиева С., Въведение в система Mathematica, ЕксПрес, Габрово, 2009.

Съдържание

Числени методи за решаване на системи уравнения-продължение

1. Норми на вектори и матрици

Разглеждаме n-мерното векторно пространство R^n, т.е. $x \in R^n$: $x = (x_1, x_2, ..., x_n)$.

Определение 1. Казваме, че в R^n е дефинирана норма $\|\cdot\|$, ако на всяко $x \in R^n$ е съпоставено реално число $\|x\|$, такова че:

1^0) $\|x\| \geq 0$ и $\|x\| = 0 \Leftrightarrow x = 0$;

2^0) $\|\lambda x\| = |\lambda| . \|x\|$, за всяко $x \in R^n$, λ - произволно реално число;

3^0) $\|x + y\| \leq \|x\| + \|y\|$, $\forall x, y \in R^n$ - неравенство на триъгълника.

Забележка. Едно нормирано пространство лесно става метрично, ако се въведе разстояние (метрика) $d(x, y) = \|x - y\|$.

Определение 2. Две норми $\|\cdot\|_1$ и $\|\cdot\|_2$ в R^n са еквивалентни, ако съществуват числата $m > 0, M > 0$, така че за всяко $x \in R^n$:

$$m\|x\|_1 \leq \|x\|_2 \leq M\|x\|_1 . \qquad (20)$$

Теорема. Всеки две норми в R^n са еквивалентни.

Най-разпространените **норми на вектор** в R^n са:

$\| x \|_1 = \max\limits_{i=1,n} |x_i|$, (кубична или чебишева норма),

$\| x \|_2 = \sum\limits_{i=1}^{n} |x_i|$, $\| x \|_3 = \sqrt{\sum\limits_{i=1}^{n} x_i^2} = \sqrt{(x,x)}$ (евклидова). (21)

Задача 1. Да се докаже, че $\| x \|_1$ е норма.

Решение: 1^0) От свойствата на модула $|x_i| \geq 0$ и от $|x_i| = 0$ следва $x_i = 0$, $i = 1,...,n$. Тогава и за максимума е изпълнено същото. 2^0) Нека λ е произволно реално число. От определението (21) и свойствата на модул $\| \lambda x \|_1 = \max\limits_{i=1,n} |\lambda x_i| = |\lambda| \max\limits_{i=1,n} |x_i| = |\lambda| \| x \|_1$. 3^0) Пак от свойствата на модула и неравенството за сума на модули имаме:

$\| x + y \|_1 = \max\limits_{i=1,n} |x_i + y_i| \leq \max\limits_{i=1,n} |x_i| + \max\limits_{i=1,n} |y_i| = \| x \|_1 + \| y \|_1$.

Задача 2. За упражнение докажете, че $\| x \|_2$ е норма.

Задача 3. Намерете трите норми на вектора $a = (-2, 3, 1, 0, -5)$.

Решение: Тук $n = 5$. По определението:

$$\| a \|_1 = \max_{i=1,5} |a_i| = \max \{|-2|, |3|, |1|, |0|, |-5|\} = 5,$$

$$\| a \|_2 = \sum_{i}^{5} |a_i| = |-2| + |3| + |1| + |0| + |-5| = 2 + 3 + 1 + 0 + 5 = 11,$$

$$\| a \|_3 = \sqrt{\sum_{i=1}^{5} a_i^2} = \sqrt{(-2)^2 + 3^2 + 1^2 + 0 + (-5)^2} = \sqrt{4 + 9 + 1 + 25} = \sqrt{39} \approx 6.25.$$

Задача 4. Покажете, че втора норма е съгласувана с първа.

Решение: Да означим индекса на максималния по модул компонент на вектора x с i^*. Тогава очевидно:

$\|x\|_1 = \max_{i=1,n} |x_i| = |x_{i^*}| \le \sum_{i=1}^{n} |x_i| = \|x\|_2 \le \sum_{i=1}^{n} |x_{i^*}| = n|x_{i^*}| = n\|x\|_1$. За $m = 1, M = n$ имаме (20).

Определение 3. Казваме, че е дефинирана норма на матрица $\|\cdot\|$, ако на всяка матрица от числа A с размерност $n\times n$ е съпоставено реално число $\|A\|$, такова че:

1^0) $\|A\|\geq 0$ и $\|A\|=0 \Leftrightarrow A=0$;

2^0) $\|\lambda A\|=|\lambda|.\|A\|$, λ - произволно число;

3^0) $\|A+B\|\leq\|A\|+\|B\|$, $\forall\ A,B$ - неравенство на триъгълника,

4^0) $\|A.B\|\leq\|A\|.\|B\|$ за $\forall\ A,B$.

Определение 4. Казваме, че нормата на вектор и нормата на матрица за дадено n са съгласувани, ако е в сила $\|Ax\|\leq\|A\|.\|x\|$.

Съгласуваната норма на матрица се определя от равенството: $\|A\|=\sup\limits_{\|x\|=1}\|Ax\|$.

Най-разпространените норми на матрици са:

$$\|A\|_1 = \max_{i=1,n} \sum_{j=1}^{n} |a_{ij}|, \quad \|A\|_2 = \max_{j=1,n} \sum_{i=1}^{n} |a_{ij}|, \quad \|A\|_3 = \sqrt{\sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij}^2} \ . \qquad (22)$$

Задача 1. Да се изчислят нормите на матрицата

$$A = \begin{pmatrix} 5 & -2 & 0 & 2 \\ -2 & -4 & 4 & 0 \\ 0 & 1 & 2 & -3 \\ 1 & 0 & -1 & 6 \end{pmatrix}.$$

Решение. За първа норма $\|A\|_1$ събираме модулите на елементите по редове: $i=1: \sum_{j=1}^{4} |a_{1j}| = |5| + |-2| + |0| + |2| = 9$,

$i=2: \sum_{j=1}^{4} |a_{2j}| = |-2| + |-4| + |4| + |0| = 10$, $i=3: \sum_{j=1}^{4} |a_{3j}| = 6$ и $i=4: \sum_{j=1}^{4} |a_{4j}| = 8$. Така

$\|A\|_1 = \max\{9, 10, 6, 8\} = 10$.

За втора норма аналогично по стълбове:

$j=1:\sum_{i=1}^{4}|a_{i1}|=|5|+|-2|+|0|+|1|=8$, $j=2:\sum_{i=1}^{4}|a_{i2}|=7$, $j=3:\sum_{i=1}^{4}|a_{i3}|=7$, $j=4:\sum_{i=1}^{4}|a_{i4}|=11$.

Тогава $\|A\|_2=\max\{8,7,7,11\}=11$.

За $\|A\|_3$ по формулата $\|A\|_3=\sqrt{5^2+(-2)^2+0^2+2^2+\ldots+6^2}=\sqrt{129}\approx 11.358$.

Отговор: $\|A\|_1=10$, $\|A\|_2=11$, $\|A\|_3\approx 11.36$.

2. Условия за сходимост на метода на простата итерация за системи линейни алгебрични уравнения

Както бе казано по-горе, въвеждането на метрика (разстояние) в R^n от вида $d(x,y)=\|x-y\|$ го прави пълно метрично пространство. Валидна е следната важна

Теорема. Достатъчното условие за сходимост на итерационния процес (13)-(14) при произволно начално приближение $x^{(0)}$ е: поне една норма на матрицата C да е по-малка от 1, т.е. да е изпълнено поне едно от следните неравенства:

$$\|C\|_1=\max_{i=1,n}\sum_{j=1}^{n}|c_{ij}|<1,\quad \|C\|_2=\max_{j=1,n}\sum_{i=1}^{n}|c_{ij}|<1,\quad \|C\|_3=\sqrt{\sum_{i=1}^{n}\sum_{j=1}^{n}c_{ij}^2}<1.$$

Доказателство. Основава се на теоремата за неподвижната точка. Остава да покажем, кога изображението $x=Cx+d$ е

свиващо. Нека $x, y \in R^n$ са два произволни вектора. Тогава

$$d(x,y) = \|x-y\| = \|Cx+d-Cy-d\| = \|Cx-Cy\| = \|C(x-y)\| \leq$$

$$\leq \|C\|\|x-y\| = \|C\| d(x,y) = q.d(x,y).$$

Тук сме означили $q = \|C\|$. Очевидно, изображението е свиващо за $0<q<1$. Твърдението е вярно за всяко начално приближение $x^{(0)}$.

Забележка. За добра сходимост на простата итерация се препоръчва $q<0.5$.

Пример. За системата (16) ще проверим условието за сходимост. Имаме матрицата

$$C = \begin{pmatrix} 0 & -0.1 & 0.3 \\ -0.2 & 0 & 0.4 \\ -0.2 & 0.2 & 0 \end{pmatrix}.$$

Пресмятаме $\|C\|_1 = \max\limits_{i=\overline{1,3}} \sum\limits_{j=1}^{3} |c_{ij}| = \max\{0.4,\ 0.6,\ 0.4\} = 0.6 < 1$

$$\|C\|_2 = \max_{j=\overline{1,3}} \sum_{i=1}^{3} |c_{ij}| = \max\{0.4,\ 0.3,\ 0.7\} = 0.7 < 1,$$

$$\|C\|_3 = \sqrt{\sum_{i=1}^{3}\sum_{j=1}^{3} c_{ij}^2} = \sqrt{0.01+0.09+0.04+0.04+0.04+0.16} = \sqrt{0.38} = 0.62 < 1.$$

И трите норми са по-малки от 1. Следователно методът на простата итерация е сходящ. По принцип е достатъчно поне една от нормите да е < 1.

3. Обобщение на метода на простата итерация за системи нелинейни уравнения (СНУ).

Вече дадохме в предишната лекция обобщение на метода на Нютон за СНУ. Да разгледаме обобщение и за проста итерация.

Постановка на задачата. Нека е дадена СНУ $F(x)=0$, във вида:

$$\left|\begin{array}{l} f_1(x_1,x_2,\ldots,x_n)=0 \\ f_1(x_1,x_2,\ldots,x_n)=0 \\ \quad\ldots \\ f_1(x_1,x_2,\ldots,x_n)=0 \end{array}\right. \text{, или във вида: } x=\Phi(x): \left|\begin{array}{l} x_1=\varphi_1(x_1,x_2,\ldots,x_n) \\ x_2=\varphi_2(x_1,x_2,\ldots,x_n) \\ \quad\ldots \\ x_n=\varphi_n(x_1,x_2,\ldots,x_n) \end{array}\right. \quad (23a,b)$$

Последното представяне е готово за прилагане на метода на простата итерация $x^{(k+1)}=\Phi(x^{(k)})$, $k=0,1,2,\ldots$. Остава да решим въпроса за началното приближение и за условията за сходимост на метода. В частност тези условия са валидни и за системи линейни алгебрични уравнения.

4. Условия за сходимост на метода на простата итерация за системи нелинейни алгебрични уравнения

Записваме подробно итерационния процес $x^{(k+1)} = \Phi(x^{(k)})$, от (23b):

$$\left|\begin{array}{l} x_1^{(k+1)} = \varphi_1(x_1^{(k)}, x_2^{(k)}, \ldots, x_n^{(k)}) \\ x_2^{(k+1)} = \varphi_2(x_1^{(k)}, x_2^{(k)}, \ldots, x_n^{(k)}) \\ \qquad\qquad \ldots \\ x_n^{(k+1)} = \varphi_n(x_1^{(k)}, x_2^{(k)}, \ldots, x_n^{(k)}) \end{array}\right., \quad k = 0,1,2,\ldots \tag{24}$$

За да използваме теоремата за неподвижната точка, ще въведем следната метрика (разстояние) между два вектора $x, y \in R^n$:

$$d(x,y) = \|x - y\|_1 = \max_{1 \le i \le n} |x_i - y_i| \tag{25}$$

Относно тази метрика R^n е пълно метрично пространство (виж също задачите към Лекция 2 – теорема за н.т.)

Определение. Нека $x^{(0)} \in R^n$, $x^{(0)} = (x_1^{(0)}, x_2^{(0)}, \ldots, x_n^{(0)})$ е начален вектор. За избраното разстояние, околност δ на $x^{(0)}$ е n-мерен куб с център $x^{(0)}$ и страна δ: $\left\| x - x^{(0)} \right\|_1 = \max\limits_{1 \le i \le n} \left| x_i - x_i^{(0)} \right|$. (26)

Теорема. Нека в δ-околност на $x^{(0)}$ функциите φ_i, $i = 1, \ldots, n$ притежават непрекъснати частни производни $\frac{\partial \varphi_i}{\partial x_j}$, $j = 1, \ldots n$. Тогава достатъчно условие за сходимост на метода на простата итерация (24) е в тази δ-околност да бъде изпълнено неравенството:

$$q = \max_{1 \le i \le n} \left\{ \max_{x \in \delta} \sum_{j=1}^{n} \left| \frac{\partial \varphi_i(x)}{\partial x_j} \right| \right\} < 1 \quad . \qquad (27)$$

Доказателство. Следва от теоремата за неподвижната точка.

Трябва да докажем, че изображението $\Phi = \left(\varphi_1(x_1,\ldots,x_n), \varphi_2(x_1,\ldots,x_n), \ldots \varphi_n(x_1,\ldots,x_n)\right)$ е свиващо. За целта с помощта на теоремата за средните стойности (Т за ср. ст.) и свойствата на нормите имаме:

$$d\left(\Phi(x),\Phi(y)\right) = \left\|\Phi(x)-\Phi(y)\right\|_1 = \max_{1\le i\le n}\left|\varphi_i(x_1,\ldots,x_n)-\varphi_i(y_1,\ldots,y_n)\right| = \text{ (по Т за ср. ст.)}$$

$$= \max_{1\le i\le n}\left\{\sum_{j=1}^{n}\left|\frac{\partial\varphi_i(\xi_i)}{\partial x_j}(x_j-y_j)\right|\right\} \le \max_j\left|x_j-y_j\right|.\max_i\sum_{j=1}^{n}\left|\frac{\partial\varphi_i(\xi_i)}{\partial x_j}\right| \le \qquad (28)$$

$$\le \left\|x-y\right\|_1.\max_i\sum_{j=1}^{n}\left|\frac{\partial\varphi_i(\xi_i)}{\partial x_j}\right| = d(x,y).\max_i\sum_{j=1}^{n}\left|\frac{\partial\varphi_i(\xi_i)}{\partial x_j}\right|.$$

От условието за непрекъснатост на частните производни следва, че те са ограничени в δ-околността на $x^{(0)}$. Да означим

техните максимуми с M_{ij}, т.е.

$$\left|\frac{\partial \varphi_i(x)}{\partial x_j}\right| \le \max_{x\in\delta}\left|\frac{\partial \varphi_i(x)}{\partial x_j}\right| = M_{ij}, \ \forall x \in \delta, \quad i, j, = 1, \ldots, n .$$

Тогава за да имаме свиващо изображение е достатъчно да изберем горната граница на (28) да бъде $q = \max_i \sum_{j=1}^{n} M_{ij} < 1$.

Теоремата е доказана.

Забележка. За добра сходимост на простата итерация се препоръчва q<0.5.

Пример:

Да се изследва за сходимост методът на проста итерация за нелинейната система:

$$\begin{cases} x = \frac{1}{2}\sin\left(\frac{x-y}{2}\right) \\ y = \frac{1}{2}\cos\left(\frac{x+y}{2}\right) \end{cases}.$$

Решение. Тук $\varphi_1(x,y) = \frac{1}{2}\sin\left(\frac{x-y}{2}\right)$, $\varphi_2(x,y) = \frac{1}{2}\cos\left(\frac{x+y}{2}\right)$. Намираме частните производни:

$$\frac{\partial\varphi_1(x,y)}{\partial x} = \frac{1}{4}\cos\left(\frac{x-y}{2}\right), \qquad \frac{\partial\varphi_1(x,y)}{\partial y} = -\frac{1}{4}\cos\left(\frac{x-y}{2}\right)$$

$$\frac{\partial\varphi_2(x,y)}{\partial x} = -\frac{1}{4}\sin\left(\frac{x+y}{2}\right), \qquad \frac{\partial\varphi_2(x,y)}{\partial y} = -\frac{1}{4}\sin\left(\frac{x+y}{2}\right).$$

Очевидно $\left|\frac{\partial \varphi_1}{\partial x}\right| \le \frac{1}{4}, \left|\frac{\partial \varphi_1}{\partial y}\right| \le \frac{1}{4}$ и $\left|\frac{\partial \varphi_1}{\partial x}\right| + \left|\frac{\partial \varphi_1}{\partial y}\right| \le \frac{1}{4} + \frac{1}{4} \le \frac{1}{2}$. Същото е и за втората функция $\varphi_2(x, y)$. Следователно $q = \frac{1}{2}$ и методът на простата итерация ще е сходящ.

Остава намирането на начално приближение, което обикновено се постига с помощта на графични методи. Виж темата за нелинейни уравнения: он-лайн обучение: www.fmi-plovdiv.org/evlm - числени методи, програми със система *Mathematica*.

Пловдивски университет „Паисий Хилендарски”
Факултет по математика и информатика
Катедра “Приложна математика и моделиране”

“Компютърни числени методи”
ЛЕКЦИЯ 5

Проф. д-р Снежана Гочева-Илиева, snow@uni-plovdiv.bg

Он-лайн обучение: www.fmi-plovdiv.org/evlm
www.fmi-plovdiv.org/evlm/DBbg - числени методи

Литература:

1. Бояджиев Д., Гочева С., Макрелов И., Попова Л. – Ръководство по числени методи – част 1, Издания: 2003, 2006, 2010.
2. Семерджиев Х., Боянов Б., Числени методи, ПУ.
3. Гочева-Илиева С., Въведение в система Mathematica, ЕксПрес, Габрово, 2009.

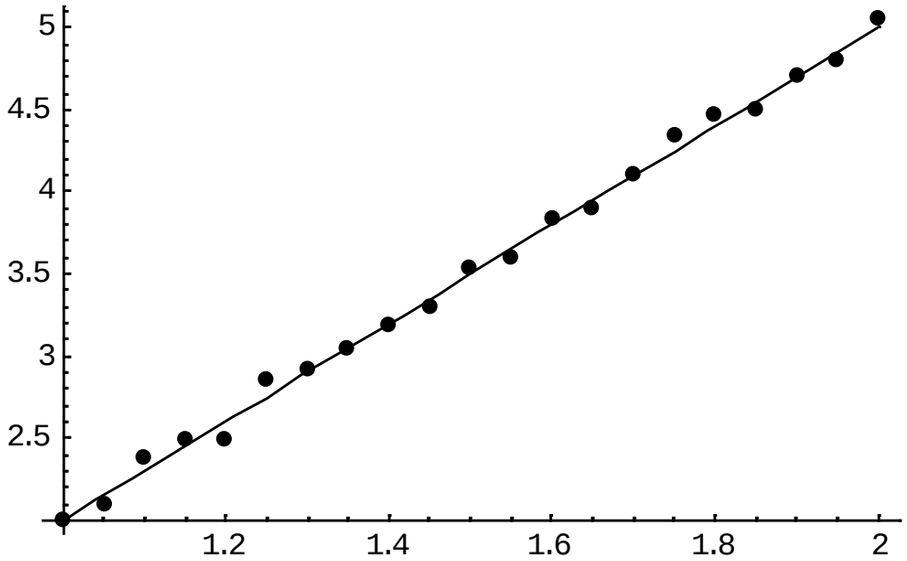
Съдържание

Интерполационен полином на Лагранж
Крайни разлики. Интерполационен полином на Нютон

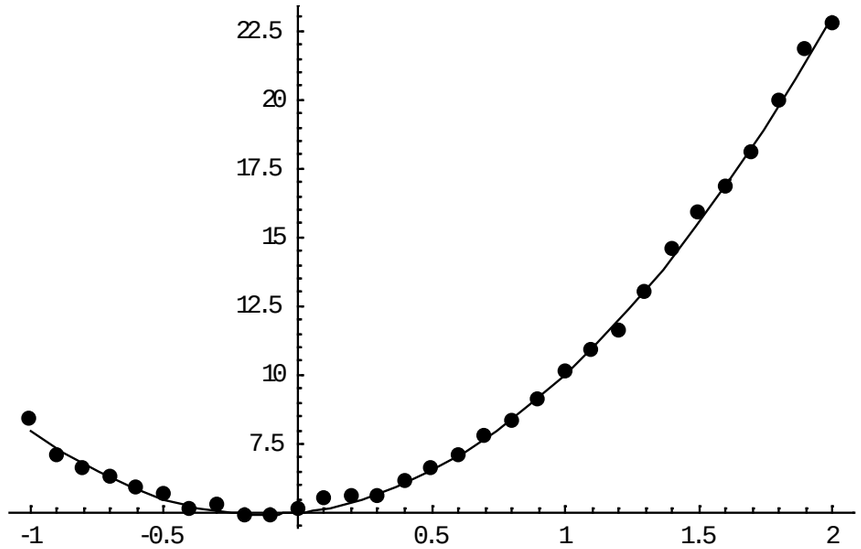
1. Основни етапи при приближаване на функции, зададени в таблична форма

Приближаването на функции може да се нарече “получаване на нещо, вместо нищо”. В общия случай функцията е известна в дискретно (крайно) множество от точки и търсим нейна стойност в някаква друга точка. Естествено, ние не я знаем. Затова стандартният подход при приближаване на финкции е:

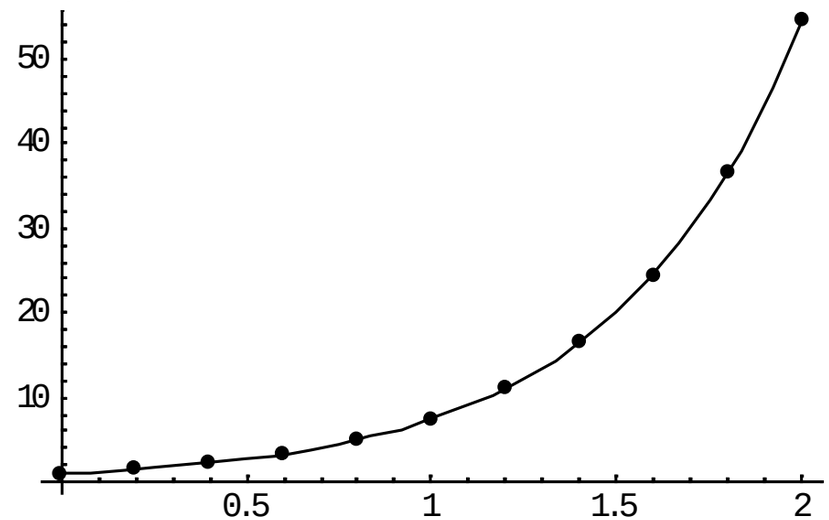
1. Начертаваме графиката на функцията по известните точки $(x_i, y_i), i = 0,1,...,n$.
2. Опитваме се да “познаем” вида (класа) на търсената функция по графиката; тя може да прилича, например на: полином от някаква степен (права линия – полином от 1ва степен, парабола – полином от 2ра степен и т.н.), тригонометрична функция, експоненциална, логаритмична и т.н. Примери:



$$y \approx ax + b$$



$$y \approx ax^2 + bx + c$$



$$y \approx e^{\lambda x}$$

3. Според броя на точките и класа на функцията и други характеристики на приближаваната функция използваме различен метод за нейното приближаване.
4. Получаваме някаква формула, “близка” до данните.
5. Използваме получената приближаваща формула, за да пресметнем стойности на функцията в точки, в които не разполагаме с данни.

Методи за приближаване:
Интерполиране,
метод на най-малките квадрати,
интерполиране със сплайни и др.

2. Задача на интерполирането

Постановка на задачата. Нека функцията $y = f(x)$ е дефинирана в някакъв интервал и е известна таблица от стойностите ѝ

x_i	x_0	x_1	...	x_n
y_i	y_0	y_1	...	y_n

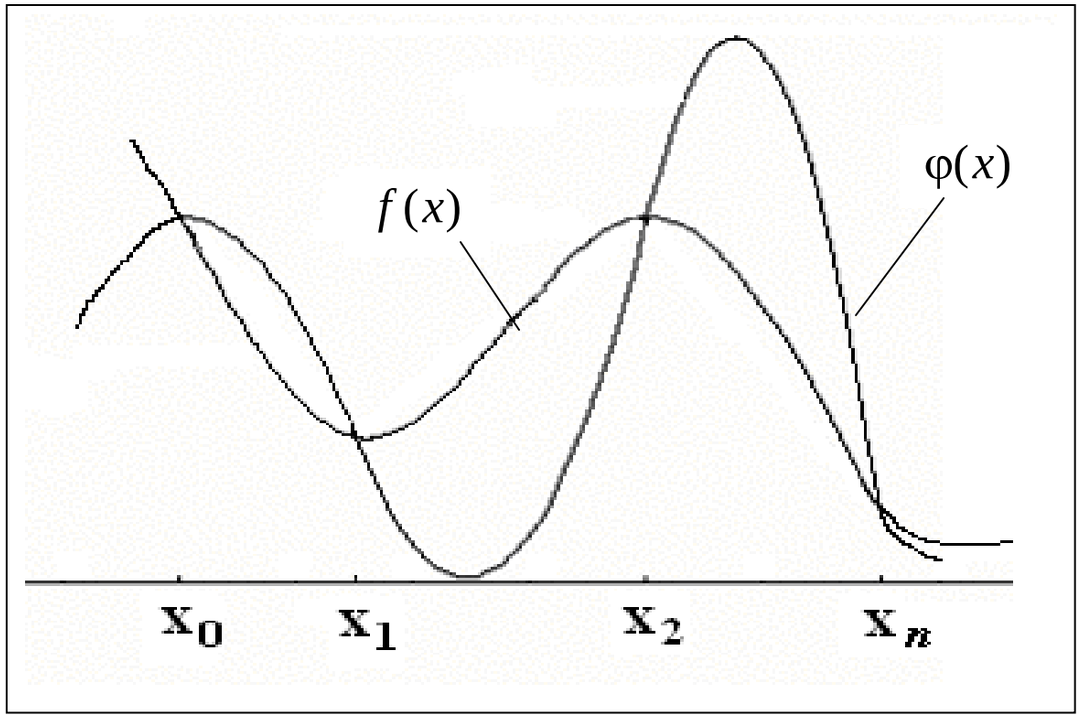
Търси се приближаваща фунция $\varphi(x)$, такава че

$$\varphi(x_i) = y_i, \ i = 0,1,\ldots,n. \qquad (1)$$

Тази задача се решава при задаване класа на функциите $\varphi(x)$.

Определение. Точките $x_0, x_1, \ldots, x_n$ се наричат възли на интерполирането.

Графично условието (1) означава, че приближаващата функция $\varphi(x)$ минава през точките (x_i, y_i), тъй като има в x_i същите стойности като $f(x)$ - виж следващата фиг.



Примери.

1) Ако изберем системата от функции $1, x, x^2, \ldots, x^n$ и разгледаме всички линейни комбинации по тази система, ще получим класа на полиномите от n-та степен $P_n(x) = a_0 + a_1 x + a_2 x^2 + \ldots + a_n x^n$. Интерполационният полином ще се определя от условието

$$P_n(x_i) = y_i, \ i = 0,1,\ldots,n. \qquad (2)$$

2) Ако изберем системата функции

$$\frac{1}{2}, \cos(x), \sin(x), \cos(2x), \sin(2x), \ldots, \cos(nx), \sin(nx), \ldots$$

ще получим класа на тригонометричните полиноми от n-та степен $T_n(x) = a_0 + a_1 \cos(x) + b_1 \sin(x) + \ldots + a_n \cos(nx) + b_n \sin(nx)$ и т.н.

Според вида на всяка конкретна функция $f(x)$ най-напред се определя класа приближаващи функции, а след това и съответната интерполираща функция (полином) $\varphi(x)$ по условието (1).

3. Интерполационен полином на Лагранж

Дадена е таблицата със стойности на функцията $y = f(x)$:

x_i	x_0	x_1	...	x_n
y_i	y_0	y_1	...	y_n

Търси се приближаваща функция във вид на полином

$P_n(x) = a_0 + a_1 x + a_2 x^2 + \ldots + a_n x^n$, така че

$$P_n(x_i) = y_i, \ i = 0,1,\ldots,n. \qquad (2)$$

Без ограничение по-нанатък ще считаме, че възлите са подредени във възходящ ред, т.е. $x_0 < x_1 < \ldots < x_n$.

Съществуването на такъв полином се осигурява със следния интерполационен полином на Лагранж, който има вида:

$$L_n(x) = \sum_{i=0}^{n} y_i F_i(x), \qquad (3)$$

където $F_i(x) = \prod_{\substack{j=0 \\ j \neq i}}^{n} \frac{(x - x_j)}{(x_i - x_j)}$ са коефициентите на Лагранж. Те имат свойството: $F_i(x_i) = 1$, $F_i(x_j) = 0,\ j \neq i$.

Формулата става: $$L_n(x) = \sum_{i=0}^{n} y_i \prod_{\substack{j=0 \\ j \neq i}}^{n} \frac{(x - x_j)}{(x_i - x_j)} \qquad (4)$$

Формулата (3) или (4) се записва подробно така:

$$L_n(x) = y_0 \frac{(x - x_1)...(x - x_n)}{(x_0 - x_1)...(x_0 - x_n)} + y_1 \frac{(x - x_0)(x - x_2)...(x - x_n)}{(x_1 - x_0)(x_1 - x_2)...(x_1 - x_n)} + ...$$
$$+ y_k \frac{(x - x_0)...(x - x_{k-1})(x - x_{k+1})...(x - x_n)}{(x_k - x_0)...(x_k - x_{k-1})(x_k - x_{k+1})...(x_k - x_n)} + ... + y_n \frac{(x - x_0)...(x - x_{n-1})}{(x_n - x_0)...(x_n - x_{n-1})} \qquad (5)$$

Очевидно този полином удовлетворява условията (2). Проверка:

При $x = x_0$,

$$L_n(x_0) = y_0 \frac{(x_0 - x_1)...(x_0 - x_n)}{(x_0 - x_1)...(x_0 - x_n)} + y_1 \frac{(x_0 - x_0)(x_0 - x_2)...(x_0 - x_n)}{(x_1 - x_0)(x_1 - x_2)...(x_1 - x_n)} + ...$$

$$+ y_n \frac{(x_0 - x_0)...(x_0 - x_{n-1})}{(x_n - x_0)...(x_n - x_{n-1})} = y_0.1 + 0 + ... + 0 = y_0 \quad .$$

При $x = x_1$,

$$L_n(x_1) = y_0 \frac{(x_1 - x_1)...(x_1 - x_n)}{(x_0 - x_1)...(x_0 - x_n)} + y_1 \frac{(x_1 - x_0)(x_1 - x_2)...(x_0 - x_n)}{(x_1 - x_0)(x_1 - x_2)...(x_1 - x_n)} + ...$$

$$+ y_n \frac{(x_1 - x_0)(x_1 - x_1)...(x_1 - x_{n-1})}{(x_n - x_0)...(x_n - x_{n-1})} = y_0.0 + y_1.1 + ... + 0 = y_1$$

и т.н. за всички възли x_i.

В сила е следната

4. Теорема за съществуване и единственост на интерполационния полином

Ако всички интерполационни възли са различни, т.е. $x_i \neq x_j,\ i \neq j$, то интерполационният полином на Лагранж е единствен.

Доказателство. Съществуването бе показано по-горе с проверката на условията за интерполиране. Но твърдението може да се докаже и независимо от този факт.

Заместваме всеки от възлите в условията (2):

$$
\begin{aligned}
&x = x_0: \quad P_n(x_0) = a_0 + a_1 x_0 + a_2 x_0^2 + \ldots + a_n x_0^n = y_0 \\
&x = x_1: \quad P_n(x_1) = a_0 + a_1 x_1 + a_2 x_1^2 + \ldots + a_n x_1^n = y_1 \\
&\ldots\ldots\ldots \\
&x = x_n: \quad P_n(x_n) = a_0 + a_1 x_n + a_2 x_n^2 + \ldots + a_n x_n^n = y_n
\end{aligned}
\qquad (6)
$$

Тук коефициентите на полинома са n+1 неизвестни $a_0, a_1, \ldots, a_n$.

От линейната алгебра е известно, че необходимото и достатъчно условие за съществуване и единственост на решение на линейната система с ненулева дясна част (което е общия за нас случай) е детерминантата на системата да е различна от нула. Така получаваме следната детерминанта, известна като детерминанта на Вандермонд

$$\det = \begin{vmatrix} 1 & x_0 & x_0^2 & \ldots & x_0^n \\ 1 & x_1 & x_1^2 & \ldots & x_1^n \\ \ldots & \ldots & & & \\ & & & & \\ 1 & x_n & x_n^2 & \ldots & x_n^n \end{vmatrix} = \prod_{i>j,\, j=0}^{n} (x_i - x_j) \neq 0 \quad .$$

Очевидно, когато всички възли са ралични,т.е. $x_i \neq x_j,\ i \neq j$, стойността на тази детерминанта е различна от нула, с което теоремата е доказана.

Без доказателство ще приведем следната

5. Теорема за оценка на грешката от интерполиране

Нека функцията $f(x)$ е дефинирана и непрекъсната в интервала $[a,b]$ и освен това съществуват и са непрекъснати в $[a,b]$ и производните й до n+1-ви ред: $f'(x), f''(x),..., f^{(n+1)}(x)$. Нека по стойностите на функцията във възлите $x_0 < x_1 < ... < x_n,\ x_i \in [a,b]$ е построен интерполационният полином на Лагранж. Тогава за всяка точка x от дефиниционната област е в сила следната оценка за грешката в тази точка:

$$f(x) - L_n(x) = \frac{f^{(n+1)}(\xi)}{(n+1)!}(x-x_0)(x-x_1)...(x-x_n), \quad \xi \in (a,b) \tag{7}$$

или като оценка на абсолютната грешка:

$$\left| f(x) - L_n(x) \right| \le \frac{M_{n+1}}{(n+1)!}\left|(x-x_0)(x-x_1)...(x-x_n)\right|, \quad M_{n+1} = \max_{[a,b]}\left| f^{(n+1)}(\xi)\right| \tag{8}$$

6. Особености при интерполиране с алгебрични полиноми

1. От оценката на грешката (7)-(8) лесно се съобразява, че грешката е малка, когато възлите са близко един до друг, така, че произведението $|(x-x_0)(x-x_1)...(x-x_n)|$ да бъде число близко до нула.
2. По същите съображения е добре така да избираме възлите, че точката x, в която търсим стойност да бъде по възможност между възлите на интерполиране. Когато x е извън интервала на възлите, казваме, че се прави екстраполиране. В някои случаи е възможна само екстраполация, но грешката е по-голяма.
3. Теоретично е доказано, че при степен $n > 6$ интерполационният полином обикновено все по-силно скача около функцията. Затова се интерполира с неголеми n.

Пример. Дадена е следната таблица на фунцията $y = f(x) = \sqrt{x+3}$:

x_i	1.0	1.2	1.4	1.6	1.8	2.0
y_i	2.	2.049	2.098	2.145	2.191	2.236

Да се намери приближена стойност в точката x'=1.65 с полином на Лагранж от втора степен и да се оцени грешката .

Решение. За да построим полинома от 2-ра степен (n=2) са необходими 3 възела. Съгласно горните забележки, като знаем, че x'=1.65, да изберем възли на интерполирането $x_0 = 1.6, x_1 = 1.8, x_2 = 2$, така че x' да е вътре в интервала $[1.6, 2]$. Това ще намали грешката. Формулата на полинома от 2-ра степен, съгласно (5) е:

$$L_2(x) = y_0 \frac{(x-x_1)(x-x_2)}{(x_0-x_1)(x_0-x_2)} + y_1 \frac{(x-x_0)(x-x_2)}{(x_1-x_0)(x_1-x_2)} + y_2 \frac{(x-x_0)(x-x_1)}{(x_2-x_0)(x_2-x_1)}.$$

Като заместим x’ приближената стойност е:

$$L_2(x') = y_0 \frac{(x'-x_1)(x'-x_2)}{(x_0-x_1)(x_0-x_2)} + y_1 \frac{(x'-x_0)(x'-x_2)}{(x_1-x_0)(x_1-x_2)} + y_2 \frac{(x'-x_0)(x'-x_1)}{(x_2-x_0)(x_2-x_1)}.$$

Заместваме тук нашите данни и получаваме:

$$L_2(1.65) = 2.145\frac{(1.65-1.8)(1.65-2)}{(1.6-1.8)(1.6-2)} + 2.191\frac{(1.65-1.6)(1.65-2)}{(1.8-1.6)(1.8-2)} + 2.236\frac{(1.65-1.6)(1.65-1.8)}{(2-1.6)(2-1.8)}$$

$$L_2(1.65) = 2.145\frac{(-0.15)(-0.35)}{(-0.2)(-0.4)} + 2.191\frac{(0.05)(-0.35)}{(0.2)(-0.2)} + 2.236\frac{(0.05)(-0.15)}{(0.4)(0.2)}$$

$$L_2(1.65) = 2.145*0.6562 + 2.191*0.4375 + 2.236*(-0.09375) \approx 2.15659.$$

Нашите данни са с 3 знака след десетичната точка (неотстранима грешка = 0.001), затова закръгляме засега резултата на 2.157.

Код на *Mathematica* за изчисляване по формулата (5):

```
f[x_] := Sqrt[x + 3]; h = 0.2; (* дефинираме функцията *)
x =.;
xi = Table[x, {x, 1., 2., h}]
y = Table[f[x], {x, 1., 2., h}]
```

```
1.65
{1., 1.2, 1.4, 1.6, 1.8, 2.}
{2., 2.04939, 2.09762, 2.14476, 2.19089, 2.23607}
```

```
x =.;

xx = 1.65; (* xx е точката, в която ще търсим приближението*)
xi = {1.6, 1.8, 2.};   y = { 2.145, 2.191,  2.236};
(* Подбор на възлите на интерполиране *)
```

$$L2[x\_] = y[[1]]\ \frac{(x - xi[[2]]) * (x - xi[[3]])}{(xi[[1]] - xi[[2]]) * (xi[[1]] - xi[[3]])} +$$

$$y[[2]]\ \frac{(x - xi[[1]]) * (x - xi[[3]])}{(xi[[2]] - xi[[1]]) * (xi[[2]] - xi[[3]])} +$$

$$y[[3]]\ \frac{(x - xi[[2]]) * (x - xi[[1]])}{(xi[[3]] - xi[[2]]) * (xi[[3]] - xi[[1]])};$$

```
Expand[%] (* Получаване развит полинома на Лагранж *)
 Print["Приближена стойност L2(xx) = ",  L2[xx]  ]
  Print["Точна стойност=  f[xx]=",  f[xx]]
```

$$1.741 + 0.2725\,x - 0.0125\,x^2$$

Приближена стойност $L2(xx) = 2.15659$

Точна стойност= $f[xx]=2.15639$

```
(* Проверка за правилност на получения полином *)
L2[1.6]
L2[1.8]
L2[2]
```

```
2.145

2.191

2.236
```

Наистина като заместим 1.6, 1.8, 2 получаваме точно първоначалните данни от таблицата на функцията. Значи това е търсеният полином на Лагранж.

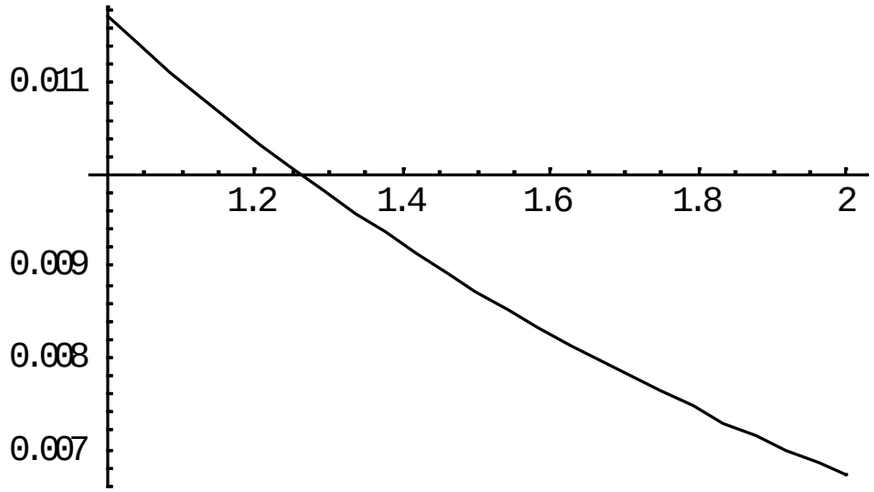
<u>Оценка на грешката:</u> В случая полиномът е от 2-ра степен, затова по формула (8) пресмятаме третата производна и правим графика на третата производна, за да видим къде е максимумът по абсолютна стойност:

$\partial_{x,x,x} f[x]$ (* Пресмятаме третата производна *)

$$\frac{3}{8\,(3+x)^{5/2}}$$

$$f3[x\_] := \text{Abs}\left[\frac{3}{8\,(3+x)^{5/2}}\right]$$

```
(* дефинираме абс. стойност на третата производна
   – (тази формула се копира от по–горе) *)
Plot[f3[x], {x, 1, 2}]
(* Правим графика на третата производна,
за да видим max*)
```




```
m3 = f3[1.];
Print["Max на третата производна в случая е при x=1: M3= ", m3]
gr = m3 * Abs[ (xx - 1.6) * (xx - 1.8) * (xx - 2)] / 3!;
Print[" Теоретична грешка= ", gr]
```

Max на третата производна в случая е при x=1: M3= 0.0117188

Теоретична грешка= 5.12695×10^{-6} ----------

Получихме теоретична грешка на метода от порядъка 0.0000052. Но тъй като неотстранимата ни грешка беше ε= 0.001, то се взима по-лошата грешка, т.е. 0.001. Така крайният резултат на приближената стойност с интерполационния полином на Лагранж е вярна само с три знака, или:

$f(1.65) = \sqrt{1.65+3} \approx 2.157$. Като сравним с точната стойност, която в случая знаем и е 2.15639, виждаме, че това отговаря на истината.

Освен това, като прегледаме първоначалните данни виждаме, че точката x'=1.65 е между точките 1.6 и 1.8, в които стойностите на функцията са съответно :

за 1.6 → 2.145, за 1.8 → 2.191,

а ние получихме приближено за 1.65→ 2.157, т.е. между тях. Което прилича на истина, дори и да не знаехме точната стойност! Това е един от критериите за правилност на резултата в общия случай.

В този пример успяхме да намерим грешката, като използвахме формула (8). В реалния случай обаче, формула на функцията не е известна, а само нейни значения в определен брой точки. Дори обикновено не се знае дали съществуват някакви производни и до какъв ред. Затова теоретичното оценяване на грешката има ограничено приложение и е съществено при прилагане на формулите, избор на точките на интерполиране и др. (Виж особености при интерполирането по-горе).

За упражнение:

1. Решете същата задача с полином от 3-та степен.

2. Постройте полином от 2-та степен с възли 1.4, 1.6, 1.8 и намерете стойността в същата точка 1.65.

7. Крайни разлики. Таблица на крайните разлики

Нека интерполационните възли $x_0, x_1, \ldots, x_n$ са равноотдалечени със стъпка $h > 0$, т.е. $\boxed{x_k = x_0 + kh,\ k = 0,1,\ldots,n}$. И нека е дадена таблицата от стойности на функцията $y = f(x)$:

x_i	x_0	$x_1 = x_0 + h$	...	$x_n = x_0 + nh$
y_i	y_0	y_1	...	y_n

Определение 1. Крайни разлики от първи ред наричаме:

- в x_0: $\Delta_0 = y_1 - y_0$; в x_1: $\Delta_1 = y_2 - y_1$, ..., в x_{n-1}: $\Delta_{n-1} = y_n - y_{n-1}$ или в общия случай: x_i: $\Delta_i = y_{i+1} - y_i$, $i = 0,1,...,n-1$.

Определение 2. Крайни разлики от втори ред наричаме:

- в x_0: $\Delta_0^2 = \Delta_1 - \Delta_0$; в x_1: $\Delta_1^2 = \Delta_2 - \Delta_1$, ..., в x_{n-2}: $\Delta_{n-2}^2 = \Delta_{n-1} - \Delta_{n-2}$ или в общия случай: x_i: $\Delta_i^2 = \Delta_{i+1} - \Delta_i$, $i = 0,1,...,n-2$.

Определение 3. Крайни разлики от k-ти ред наричаме числата:

$$\Delta_i^k = \Delta_{i+1}^{k-1} - \Delta_i^{k-1},\ i = 0,1,...,n-k,\ k = 1,...,n. \qquad (11)$$

Получаваме следната триъгълна таблица на крайните разлики:

i	x_i	y_i	Δ_i	Δ_i^2		Δ_i^{n-1}	Δ_i^n
0	x_0	y_0	Δ_0	Δ_0^2		Δ_0^{n-1}	Δ_0^n
1	x_1	y_1	Δ_1	Δ_1^2		Δ_1^{n-1}	
2	x_2	y_2	Δ_2	Δ_2^2			
…							
n-2	x_{n-2}	y_{n-2}	Δ_{n-2}	Δ_{n-2}^2			
n-1	x_{n-1}	y_{n-1}	Δ_{n-1}				
n	x_n	y_n					

Пример. Да се състави таблицата на кр. р. по стойностите на функцията $y = e^x$:

x_i	3.60	3.65	3.70	3.75	3.80
y_i	36.598	38.475	40.447	42.521	44.701

(*)

Определяме $h = 0.05$, равномерна стъпка – можем да ползваме кр.р. Таблицата е:

i	x_i	y_i	Δ_i	Δ_i^2	Δ_i^3	Δ_i^4
0	3.60	36.598	1.877	0.095	0.007	-0.003
1	3.65	38.475	1.972	0.102	0.004	
2	3.70	40.447	2.074	0.106		
3	3.75	42.521	2.180			
4	3.80	44.701				

Забелязваме, че крайните разлики намаляват по стойност.

8. Свойства на крайните разлики

Апаратът на кр.р. е добре изследван, тъй като той се използва в един от най-разпространените методи за числено решаване на диференциални задачи – метод на кр.р. или диференчните схеми.

Тук ще приведем само две техни основни свойства:

1) Формула за изразяване на кр.р. със стойностите на функцията:

$$\Delta_i^2 = \Delta_{i+1} - \Delta_i = (y_{i+2} - y_{i+1}) - (y_{i+1} - y_i) = y_{i+2} - 2y_{i+1} + y_i,$$

$$\Delta_i^3 = \Delta_{i+1}^2 - \Delta_i^2 = (y_{i+3} - 2y_{i+2} + y_{i+1}) - (y_{i+2} - 2y_{i+1} - y_i) = y_{i+3} - 3y_{i+2} + 3y_{i+1} - y_i,$$

т.е. с участието на коефициентите на нютоновия бином, или

$$\Delta_i^k = \Delta_{i+1}^{k-1} - \Delta_i^{k-1} = y_{i+k} - \frac{k}{1!} y_{i+k-1} + \frac{k(k-1)}{2!} y_{i+k-2} - \frac{k(k-1)(k-2)}{3!} y_{i+k-3} + \ldots + (-1)^k y_i \quad (12)$$

2) Приближение на производните с кр.р.:

От определението на производна имаме:

$$y'(x_i) = \lim_{h\to 0} \frac{y_{i+1} - y_i}{h} = \lim_{h\to 0} \frac{\Delta_i}{h} \approx \frac{\Delta_i}{h}\ ,\quad y''(x_i) = \lim_{h\to 0} \frac{y'_{i+1} - y'_i}{h} \approx \frac{\Delta_{i+1} - \Delta_i}{h.h} \approx \frac{\Delta_i^2}{h^2}\ , \ldots$$

В общия случай, ако съществуват производните:

$$y^{(k)}(x_i) \approx \frac{\Delta_i^k}{h^k}\ . \qquad (13)$$

Разпространение на грешката в таблицата на крайните разлики

Нека допуснем, че има някаква допусната грешка ε (от закръгляне, от входни данни и др.). Съгласно свойство 1) грешката ще се рапространява с множители - коефициентите на Нютоновия бином. Директно по примерна таблица с изчисляване на кр.р. се показва разрастването на грешката с реда на кр.р.

i	x_i	y_i	Δ_i	Δ_i^2	Δ_i^3	Δ_i^4	Δ_i^5	Δ_i^6
0	x_0	0	0	0	ε	-4ε	10ε	-20ε
1	x_1	0	0	ε	-3ε	6ε	-10ε	
2	x_2	0	ε	-2ε	3ε	-4ε		
3	x_3	ε	-ε	ε	-ε			
4	x_4	0	0	0				
5	x_5	0	0					
6	x_6	0						

Освен това при близки стойности на функцията и относителната грешка от изваждането е неустойчива (виж Лекция 1 – пример за неустойчивост на относителната грешка).

Извод. Да се избягва ползването на кр.р. от висок ред!

9. Интерполационен полином на Нютон. Формули на Нютон за интерполиране напред и назад

Както знаем интерполационният полином на Лагранж е единствен. Но неговият вид не е много ефективен за пресмятане, особено многократно. Затова в някои случаи е по-удобно да използваме друг вид на полинома. Съществуват много еквивалентни представяния – формили на Нютон, Ейткин, Гаус...

Постановка. Нека възлите на интерполиране са равноотдалечени със стъпка $h > 0$, т.е. $x_k = x_0 + kh,\ k = 0,1,\ldots,n$. И нека е дадена таблицата от стойности на функцията $y = f(x)$:

x_i	x_0	$x_1 = x_0 + h$	...	$x_n = x_0 + nh$
y_i	y_0	y_1	...	y_n

Търсим интерполационния полином на Нютон във вида:

$$P_n(x) = a_0 + a_1(x - x_0) + a_2(x - x_0)(x - x_1) + \ldots + a_n(x - x_0)(x - x_1)\ldots(x - x_{n-1}), \quad (14)$$

за който $P_n(x_i) = y_i, \ i = 0,1,\ldots,n$. (15)

Извод на полинома на Нютон за интерполиране напред

Заместваме $x = x_0$ в (14) и от условие (15) веднага получаваме

$$P_n(x_0) = a_0 + a_1(x_0 - x_0) + a_2(x_0 - x_0)(x_0 - x_1) + \ldots + a_n(x_0 - x_0)\ldots = a_0 = y_0 \ \rightarrow \ \boxed{a_0 = y_0}.$$

При $x = x_1$ аналогично

$$P_n(x_1) = a_0 + a_1(x_1 - x_0) + a_2(x_1 - x_0)(x_1 - x_1) + \ldots = y_0 + a_1 h = y_1 \ \rightarrow \ \boxed{a_1 = \frac{y_1 - y_0}{h} = \frac{\Delta_0}{h}}$$

По-нататък (по принципа на непълната математическа индукция) намираме за $x = x_k$, че $\boxed{a_k = \frac{\Delta_0^k}{k!h^k}}$, $k = 0,1,\ldots,n$. Или

$$P_n(x) = y_0 + \frac{\Delta_0}{h}(x - x_0) + \frac{\Delta_0^2}{2!h^2}(x - x_0)(x - x_1) + \ldots + \frac{\Delta_0^n}{n!h^n}(x - x_0)(x - x_1)\ldots(x - x_{n-1}).$$

Накрая полагаме тук $t = \frac{x - x_0}{h}$ и след елементарни преобразования

получаваме формула на Нютон за интерполиране напред

$$P_n(t) = y_0 + t\Delta_0 + \frac{t(t-1)}{2!}\Delta_0^2 + \frac{t(t-1)(t-2)}{3!}\Delta_0^3 + \ldots + \frac{t(t-1)\ldots(t-n+1)}{n!}\Delta_0^n, \quad t = \frac{x-x_0}{h} \tag{16}$$

Тази формула се нарича формула на Нютон за интерполиране напред, защото в нея участват само кр.р. в точката x_0 - в случая това са стойностите на кр.р. от първия ред на таблицата, а именно y_0, Δ_0, Δ_0^2 и т.н.

Аналогично, ако търсим формулата във вида

$$P_n(x) = b_0 + b_1(x-x_n) + b_2(x-x_n)(x-x_{n-1}) + \ldots + b_n(x-x_n)(x-x_{n-1})\ldots(x-x_1)$$

се получава формула на Нютон за интерполиране назад:

$$P_n(t) = y_n + t\Delta_{n-1} + \frac{t(t+1)}{2!}\Delta_{n-2}^2 + \frac{t(t+1)(t+2)}{3!}\Delta_{n-3}^3 \ldots + \frac{t(t+1)\ldots(t+n-1)}{n!}\Delta_0^n, t = \frac{x-x_n}{h} \tag{17}$$

Оценка на грешката във формулата на Лагранж

Като се използват равноотдалечени възли, грешката е:

$$R_n(x) = h^{n+1} \frac{t(t-1)\ldots(t-n)}{(n+1)!} f^{(n+1)}(\xi), \quad \xi \in (x_0, x_n) \tag{18}$$

10. Преимущества на формулите на Нютон

- Очевидно формулите на Нютон (16) или (17) изискват по-малък брой аритметични действия от формулата на Лагранж
- За (16) ако x е близо до x_0, то t е малко. За достатъчно гладка функция и кр.р. намаляват, така че обикновено от формулата се пресмятат само няколко събираеми, а останалите се отхвърлят, щом станат по-малки по модул от зададената точност ε. Следователно сметките са реално още по-малко.
- Аналогично (17) е удобна, когато x е близо до x_n.
- Ако x е близо до някое друго x_j от таблицата, то правим преномерация с начало (или край) x_j и избираме (16) или (17).

Пример. Да се намери приближена стойност на $e^{3.62}$ с точност ε=0.001 по таблица (*), за която изчислихме таблицата на кр.р.:

i	x_i	y_i	Δ_i	Δ_i^2	Δ_i^3	Δ_i^4
0	3.60	36.598	1.877	0.095	0.007	-0.003
1	3.65	38.475	1.972	0.102	0.004	
2	3.70	40.447	2.074	0.106		
3	3.75	42.521	2.180			
4	3.80	44.701				

Решение: Очевидно x=3.62 се намира най-близо до x_0, h=0.05 и е удобна формулата на Нютон напред. Трябва да използваме кр.р. от първия ред – подчертани с права черта.

Изчисляваме $t = \frac{x - x_0}{h} = \frac{3.62 - 3.60}{0.05} = 0.4$. Тогава от (16):

$$P_n(t) = y_0 + t\Delta_0 + \frac{t(t-1)}{2!}\Delta_0^2 + \frac{t(t-1)(t-2)}{3!}\Delta_0^3 + \ldots$$

$$= 36.598 + 0.4(1.877) + \frac{0.4(-0.6)}{2!}0.095 + \frac{0.4(-0.6)(-1.6)}{3!}0.007 + \ldots =$$

$$= 36.598 + 0.7508 - 0.0114 + 0.000448 \approx 37.3374 \approx 37.337 \quad ,$$

Виждаме, че добавянето на четвъртия член 0.000448 не влияе, защото е по-малък от ε=0.001. Затова прекъсваме изчисленията дефакто след намирането на третия член. Закръгленият резултат според даденото ε е

$$f(x) = e^{3.62} \approx P_2(3.62) \approx 37.337 \; . \qquad (19)$$

Забележка. С прекъсването по точност освен това определяме и каква степен има приближаващият полином, в случая n=2.

Пример. Оценете грешката на приближението за предния пример
Решение: а) Неотстранима грешка – данните са с три знака след десетичната точка, т.е. 0.001.

б) Грешка на метода. Пресмятаме производните $f' = f'' = f''' = e^x$. Тъй като $f(x) = e^x$ е растяща за всяко x, то очевидно $|f'''(\xi)| \le \max\limits_{[3.6,3.8]} |e^x| = e^{3.8} \le 45$ (от таблицата с данни). От формула (18)

$R_n(x) = h^{n+1} \dfrac{t(t-1)\ldots(t-n)}{(n+1)!} f^{(n+1)}(\xi)$ при за n=2 оценяваме абсолютната грешка на приближението по метода на Нютон с кр.р.:

$$|R_n(x=3.62)| = 0.05^3 \frac{0.4(-0.6)(-1.6)}{3!} \left| f^{(n+1)}(\xi) \right| \le 0.000125 . \frac{0.384}{3!} .45 = 0.00036 \approx 0.0005 .$$

Заключение. От а) и б) следва, че грешката на приближението е 0.001. Следователно всички знаци в (19) са верни, т.е. $e^{3.62} \approx 37.337$.

Домашно. Намерете приближено а) $e^{3.77}$ и б) $e^{3.67}$.

Упътване. а) Използвайте формулата Нютон назад (17) и почертаните с къдрава линия последни по стълбовете кр.р.

б) Преномерирайте възлите с начало $x_0 = 3.65$ и игнорирайте първия ред. След това приложете формулата Нютон напред, в която ще участват горните кр.р. от втория ред на таблицата.

Пловдивски университет „Паисий Хилендарски”
Факултет по математика и информатика
Катедра “Приложна математика и моделиране”

“Компютърни числени методи”
ЛЕКЦИЯ 6

Проф. д-р Снежана Гочева-Илиева, snow@uni-plovdiv.bg

Он-лайн обучение: www.fmi-plovdiv.org/evlm
www.fmi-plovdiv.org/evlm/DBbg - числени методи

Литература:

1. Бояджиев Д., Гочева С., Макрелов И., Попова Л. – Ръководство по числени методи – част 1, Издания: 2003, 2006, 2010.
2. Семерджиев Х., Боянов Б., Числени методи, ПУ.
3. Гочева-Илиева С., Въведение в система Mathematica, ЕксПрес, Габрово, 2009.

Съдържание

Метод на най-малките квадрати
Интерполационни сплайни – линеен, квадратичен

1. Приближаване на експериментални данни

Често при провеждане на експерименти се натрупва голямо количество таблици от данни, за която експериментаторът желае да установи аналитичен закон (формула), която възможно най-добре да описва данните. Обикновено по характер тези данни са получени с голяма неотстранима грешка, най-вече зависеща от точността на прибора, методиката на измерване и други фактори. Ще разгледаме случая, когато търсим да приближим реална функция на една реална променлива.

Изходна постановка. Нека функцията $y = f(x)$ е дефинирана в някакъв интервал, в който е известна таблица от стойностите ѝ

x_i	x_1	x_2	...	x_N
y_i	y_1	y_2	...	y_N

Тук точките $x_1, x_2, \ldots, x_N$ не е задължително да са различни.

Да предположим, че е избрана система от базисни функции $\varphi_0(x), \varphi_1(x), \ldots, \varphi_m(x)$ и се търси приближение от вида

$$f(x) \approx c_0\varphi_0(x) + c_1\varphi_1(x) + \ldots + c_m\varphi_m(x). \quad (1)$$

Този тип приближение е линейна комбинация на избраните m+1 на брой базисни функции $\varphi_j(x), j = 0,1,\ldots m$ и се нарича обобщен полином. Очевидно (1) е линеен математически модел спрямо базисните функции. Коефициентите $c_j, j = 0,1,\ldots m$ са неизвестни.

<u>Примери за базисни функции:</u>

1) Най-често това са едночлените $\varphi_j(x) = x^j,\ j = 0,1,\ldots m$.

Така приближаващата функция като тяхна линейна комбинация е алгебричен полином $P_m(x)$ от степен m, т.е.

$$f(x) \approx P_m(x) = c_0 + c_1x + \ldots + c_mx^m \quad (2)$$

2) Ако данните наподобяват тригонометрична функция, можем да изберем базисни полиноми $\varphi_j(x) = \sin(jx),\ j = 0,1,\ldots m$. Тогава

$$f(x) \approx T_m(x) = c_0 + c_1 \sin(x) + c_2 \sin(2x) + \ldots + c_m \sin(mx)\,. \qquad (3)$$

3) Друг вариант е експоненциалната апроксимация

$$f(x) \approx E_m(x) = c_0 e^{\lambda_0 x} + c_1 e^{\lambda_1 x} + \ldots + c_m e^{\lambda_m x}\,, \qquad (4)$$

където λ_j са зададени (а ако не са – става нелинеен модел).

2. Метод на най-малките квадрати (МНМК) с алгебрични полиноми

Основно предположение: нека $m \ll N$, т.е. броят на данните N е много по-голям от степента m на обобщения полином.

Означаваме разликата (остатъка) между стойностите на функцията y_i и полинома $P_m(x)$ за всяка точка x_i с

$$r_i = P_m(x_i) - y_i = c_0 + c_1 x_i + c_2 x_i^2 + \ldots c_m x_i^m - y_i, \; i = 1, \ldots, N. \quad (5)$$

Критерият за намиране на коефициентите c_j в (1) по МНМК е сумата от квадратите на всички остатъци да бъде възможно най-малка. Стигаме до задачата: Измежду всички полиноми от дадена степен m да се намери този, за който

$$\sum_{i=1}^{N} r_i^2 = \min. \quad (6)$$

Минимумът тук зависи от $c_0, c_1, \ldots c_m$. По-подробно от (5) и (6) трябва да намерим минимума на функцията с променливи c_j:

$$\Phi(c_0, c_1, \ldots, c_m) = \sum_{i=1}^{N} r_i^2 = \sum_{i=1}^{N} (c_0 + c_1 x_i + c_2 x_i^2 + \ldots c_m x_i^m - y_i)^2 \quad . \qquad (7)$$

От математическия анализ е известно, че необходимото условие за екстремум на диференцируема функция на много променливи е всички частни производни да са равни на нула, т.е.

$$\left|\begin{array}{l} \dfrac{\partial \Phi(c_0, c_1, \ldots, c_m)}{\partial c_0} = 0 \\ \dfrac{\partial \Phi(c_0, c_1, \ldots, c_m)}{\partial c_1} = 0 \\ \ldots \\ \dfrac{\partial \Phi(c_0, c_1, \ldots, c_m)}{\partial c_m} = 0 \end{array}\right. . \qquad (8)$$

За първото уравнение от (7) диференцираме спрямо c_0 при фиксирани останали коефициенти c_j (тук x_i, y_i са дадени). Имаме:

$$\frac{\partial \Phi}{\partial c_0} = 2\sum_{i=1}^{N}(c_0 + c_1 x_i + c_2 x_i^2 + \ldots c_m x_i^m - y_i) = 0 .$$

За второто уравнение аналогично:

$$\frac{\partial \Phi}{\partial c_1} = 2\sum_{i=1}^{N}(c_0 + c_1 x_i + c_2 x_i^2 + \ldots c_m x_i^m - y_i).x_i = 0, \ldots$$

$$\frac{\partial \Phi}{\partial c_m} = 2\sum_{i=1}^{N}(c_0 + c_1 x_i + c_2 x_i^2 + \ldots c_m x_i^m - y_i).x_i^m = 0.$$

След приведение пред неизвестните стигаме до системата

$$\left|\begin{array}{l} Nc_0 + \left(\sum_{i=1}^{N} x_i\right)c_1 + \left(\sum_{i=1}^{N} x_i^2\right)c_2 + \ldots + \left(\sum_{i=1}^{N} x_i^m\right)c_m = \sum_{i=1}^{N} y_i \\ \left(\sum_{i=1}^{N} x_i\right)c_0 + \left(\sum_{i=1}^{N} x_i^2\right)c_1 + \ldots + \left(\sum_{i=1}^{N} x_i^{m+1}\right)c_m = \sum_{i=1}^{N} x_i y_i \\ \ldots \\ \left(\sum_{i=1}^{N} x_i^m\right)c_0 + \left(\sum_{i=1}^{N} x_i^{m+1}\right)c_1 + \ldots + \left(\sum_{i=1}^{N} x_i^{2m}\right)c_m = \sum_{i=1}^{N} x_i^m y_i \end{array}\right. . \quad (9)$$

Очевидно това е линейна система за намиране на $c_0, c_1, \dots c_m$.

Може да се покаже, че тази система има единствено решение. При не-много големи m тя може да се решава по метода на квадратния корен или по метода на Гаус. При голяма степен m на полинома е известно, че (9) е лошо обусловена, т.к. детерминантата е малко число. Затова МНМК с алгебрични полиноми от вида (2) се прилага за $m \le 5$. За по-големи m се избират система ортогонални полиноми или друга система базисни функции.

Определение. Полиномът $P_m^*(x) = c_0^* + c_1^* x + \dots + c_m^* x^m$, чиито коефициенти $c_0^*, c_1^*, \dots, c_m^*$ са решения на (9) се нарича полином на най-добро приближение по МНМК към функцията $f(x)$.

Определение. Грешка на приближението по МНМК или среднoквадратична грешка се нарича отклонението на полинома на най-добро приближение по МНМК

$$\delta_m = \sqrt{\sum_{i=1}^{N} r_i^2} = \sqrt{\sum_{i=1}^{N} (c_0^* + c_1^* x_i + c_2^* x_i^2 + \ldots c_m^* x_i^m - y_i)^2} \quad . \qquad (10)$$

Как се избира най-добрата степен m за полинома по МНМК?

Това в общия случай е труден проблем. Обикновено постъпват така:

Нека данните имат неотстранима грешка ε. Започва се с $m=1$, изчислява се δ_1. Ако $\delta_1 \approx \varepsilon$, значи решението е добро. Ако ср.кв. грешка не е приблизително равна на ε, се продължава с $m=2$ и δ_2 и т.н., докато се намери оптималното m, за което $\delta_m \approx \varepsilon$. Ако има резки скоци и не може да се опредли m, то се избират други базисни функции и т.н.

2.1 Пример. Приближение на данни с МНМК с Mathematica

http://www.fmi-plovdiv.org/evlm/DBbg/database/numan/leastsquares_BG/index.html

Задача 1. Дадена е функцията $f(t) = t + \sin(t^2)$ в интервала $[1, 2]$.

а) Да се табулира функцията в 11 точки в дадения интервал.

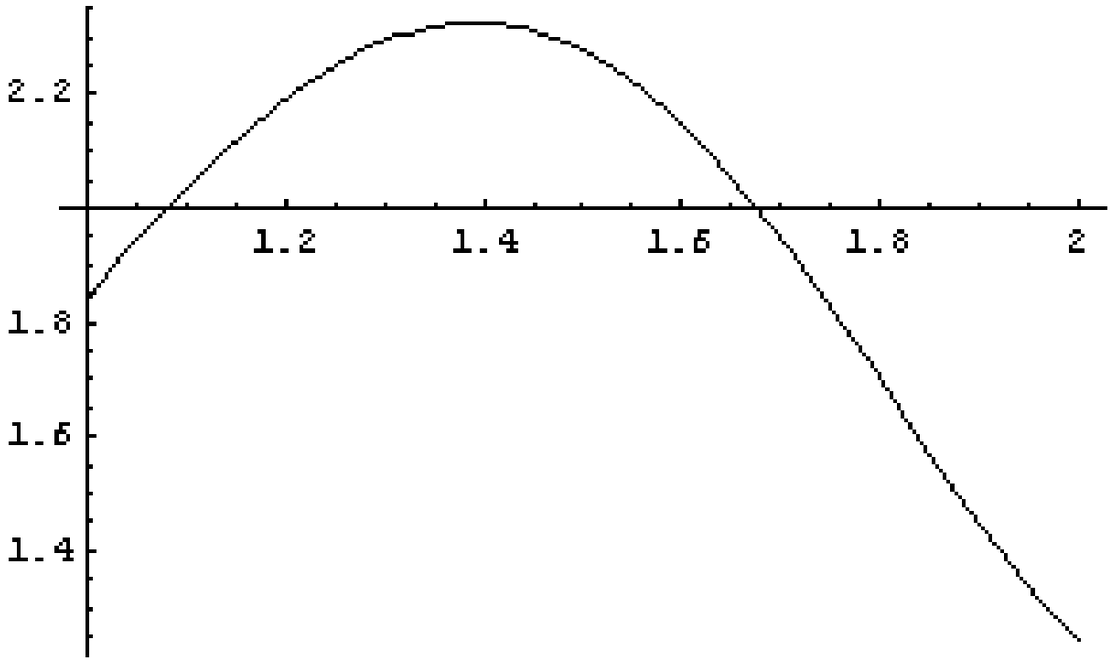
б) Да се намери полиномът на най – добро приближение по МНМК от първа степен.

в) Да се оцени грешката на приближението.

Решение :

Дефинираме функцията и начертаваме графиката ѝ в интервала [1, 2] :

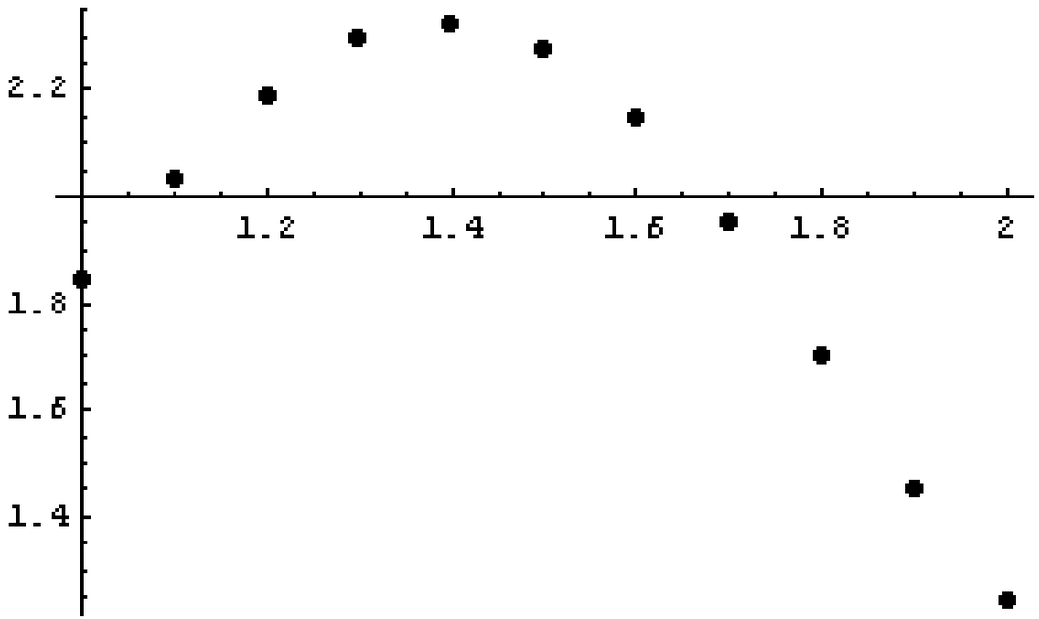
```
f[t_] := t + Sin[t^2]
a = 1.; b = 2.;
gf = Plot[f[t], {t, a, b}]
```



а) Пресмятаме стойностите на масив от точки x_i, $y_i = f[x_i]$, $i = 1, 2, \ldots n1$,
като избираме n1 = 11 равноотдалечени точки със стъпка h в интервала [a, b]:

```
n1 = 11;  h = (b - a)/(n1 - 1); Print["Step h= ", h]
x = Table[a + h * (i - 1), {i, 1, n1}]; Print["x=", x]
y = Table[f[x[[i]]], {i, 1, n1}]; Print["y=", y]
xy = Table[{x[[i]], y[[i]]}, {i, 1, n1}];
gy = ListPlot[xy, PlotStyle → PointSize[0.02]]
```

```
Step h= 0.1
x={1., 1.1, 1.2, 1.3, 1.4, 1.5, 1.6, 1.7, 1.8, 1.9, 2.}
y={1.84147, 2.03562, 2.19146, 2.2929, 2.32521, 2.27807, 2.14936, 1.94895, 1.70175, 1.44853, 1.2432}
```



б) Пресмятаме всички суми, необходими за МНМК от първа степен, решаваме получената система с LinearSolve и извличаме c0 и c1 :

$$s0 = n1;\ s1 = \sum_{i=1}^{n1} x_{[i]}\ ;\ s2 = \sum_{i=1}^{n1} x_{[i]}^2;\ d1 = \sum_{i=1}^{n1} y_{[i]};\ d2 = \sum_{i=1}^{n1} x_{[i]} * y_{[i]};$$

$$s = \begin{pmatrix} s0 & s1 \\ s1 & s2 \end{pmatrix}$$

$$d = \begin{pmatrix} d1 \\ d2 \end{pmatrix}$$

$$c = \text{LinearSolve}[s, d]$$

$$c0 = c_{[1]_{[1]}}$$

$$c1 = c_{[2]_{[1]}}$$

```
{{11, 16.5}, {16.5, 25.85}}
```

```
{{21.4565}, {31.4175}}
```

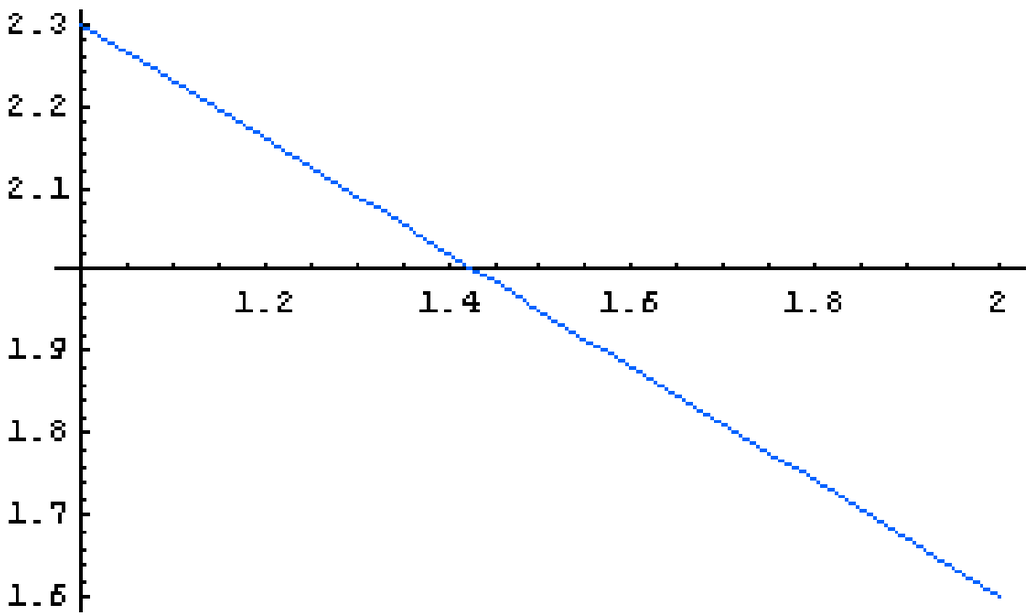
```
{{2.99685}, {-0.697508}}
```

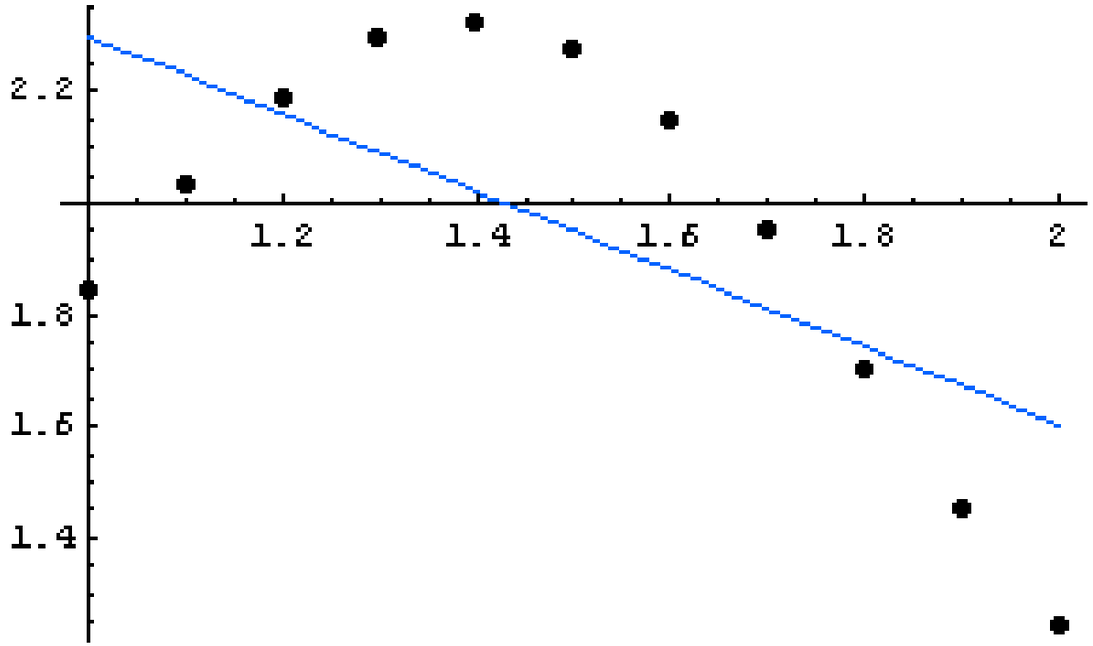
```
2.99685
```

```
-0.697508
```

Дефинираме като функция полинома на най –
добро приближение от 1 – ва степен по МНМК и рисуваме графиката му.
Накрая показваме с Show едновременно графиките
на таблицата на f[t] и приближението му p1[t] :

```
p1[t_] := c0 + c1 * t
gp1 = Plot[p1[t], {t, a, b}, PlotStyle → Hue[.6]]
Show[gy, gp1]
```





в) Изчисляваме средноквадратичната грешка на приближението :

$$\delta_1 = \sqrt{\sum_{i=1}^{n1} (f[x_{[i]}] - p1[x_{[i]}])^2}$$

0.87215

Заключение: **Получената грешка очевидно е голяма, както се вижда и от графиката на двете функции. Може да се потърси приближение с полином от по-висока степен.**

Задача 2. За функцията от задача 1 :

а) Да се намери полиномът на най – добро приближение по МНМК от втора степен.

в) Да се оцени грешката на приближението.

Решение :

а) Пресмятаме всички суми, необходими за МНМК от втора степен, решаваме получената система с LinearSolve и извличаме коефициентите на полинома, означаваме ги с v0, v1 и v2 :

$$s0 = n1;\ s1 = \sum_{i=1}^{n1} x_{[i]}\ ;\ s2 = \sum_{i=1}^{n1} x_{[i]}^2;\ s3 = \sum_{i=1}^{n1} x_{[i]}^3;\ s4 = \sum_{i=1}^{n1} x_{[i]}^4;$$

$$d1 = \sum_{i=1}^{n1} y_{[i]};\ d2 = \sum_{i=1}^{n1} x_{[i]} * y_{[i]};\ d3 = \sum_{i=1}^{n1} x_{[i]}^2 * y_{[i]};$$

$$s = \begin{pmatrix} s0 & s1 & s2 \\ s1 & s2 & s3 \\ s2 & s3 & s4 \end{pmatrix}$$

$$d = \begin{pmatrix} d1 \\ d2 \\ d3 \end{pmatrix}$$

```
c = LinearSolve[s, d]
v0 = c[[1]][[1]]
v1 = c[[2]][[1]]
v2 = c[[3]][[1]]
```

```
{{11, 16.5, 25.85}, {16.5, 25.85, 42.075}, {25.85, 42.075, 70.7333}}
{{21.4565}, {31.4175}, {47.8688}}
{{-3.32315}, {8.1211}, {-2.93954}}
-3.32315        8.1211        -2.93954
```

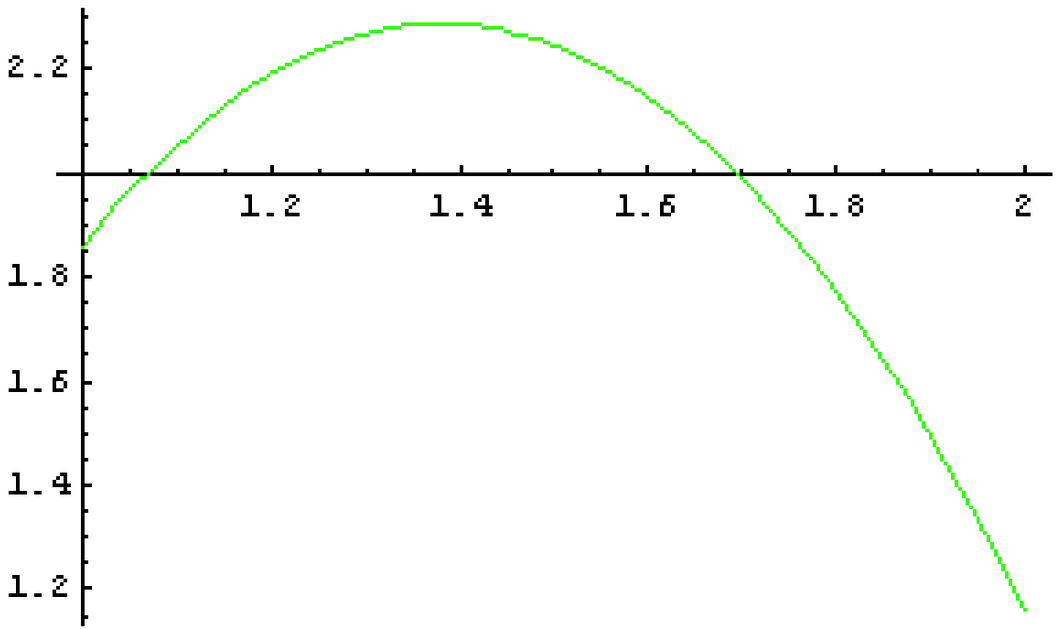
Дефинираме като функция полинома на най –

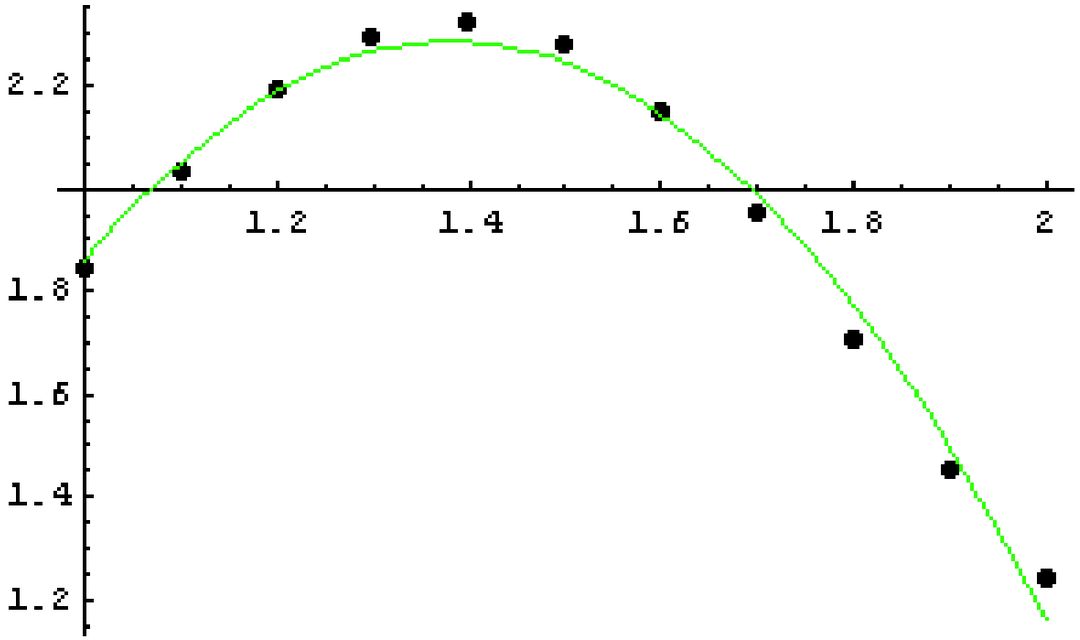
добро приближение от 2 – ра степен по МНМК и рисуваме графиката му.

Накрая показваме с Show едновременно графиките

на таблицата на f[t] и приближението му – p2[t] :

```
p2[t_] := v0 + v1 * t + v2 * t^2
gp2 = Plot[p2[t], {t, a, b}, PlotStyle → Hue[.3]]
Show[gy, gp2]
```





в) Изчисляваме средноквадратичната грешка на приближението :

$$\delta_2 = \sqrt{\sum_{i=1}^{n1} (f[x_{[i]}] - p2[x_{[i]}])^2}$$

```
0.138776
```

<u>Заключение:</u> Получената грешка очевидно е все още голяма, но по-приемлива от предишното приближение (сравни графиките на двете функции). Може да се потърси приближение с полином от още по-висока степен. -→ прегледайте уебстраницата на примера.

2.2 Пример. Приближение с МНМК с Mathematica – данни за лазер с пàри на меден бромид

http://www.fmi-plovdiv.org/evlm/DBbg/database/numan/MNMK_CuBr_laser/2leastsquares_CuBr.html

3. Интерполационни сплайни

3.1 Постановка на задачата

Нека $y = f(x)$ е функция, дефинирана в интервала $[a,b]$ и е известна таблица от стойности $y_i = f(x_i)$ в точките (възлите) $a \le x_0 < x_1 < x_2 < \ldots < x_n \le b$. Стъпките между x_{i-1} и x_i ще означаваме с $h_i = x_i - x_{i-1}$. Нека таблицата е:

x_i	x_0	x_1	...	x_n
y_i	y_0	y_1	...	y_n

Определение. Интерполационният сплайн $S_k(f,x)$ от ред k е функция със следните свойства:

(1) $S_k(f,x)$ е полином $f_i(x)$ от степен k във всеки подинтервал $[x_{i-1}, x_i]$, $i = \overline{1,n}$.

(2) $S_k(f,x)$ интерполира функцията, т.е. $S_k(f,x_i)=y_i,\ \ i=\overline{0,n}$.
(3) $S_k(f,x)$ и производните му до ред (k-1) са непрекъснати в $[a,b]$.

В общия случай сплайнът от ред k не е единствен. За единственост се налагат допълнителни условия. Най-използвани са линеен, квадратичен и кубичен сплайн.

3.2 Линеен сплайн

Тук $k = 1$ и във всеки подинтервал $[x_{i-1}, x_i], i = \overline{1,n}$, сплайнът $S_1(f,x)$ е полином от първа степен (отсечка). Тъй като през две точки минава само една отсечка, линейният сплайн е единствен. Като се интерполира таблицата са всеки подинтервал, получаваме коефициентите на линейния сплайн по изходна таблица с данни. Графиката на сплайна е начупена линия.

Как се използва сплайнът за приближение в произволна точка z от $[a,b]$?

а) определяме в кой подинтервал $[x_{i-1}, x_i]$ се намира z.

б) заместваме $x = z$ в съответния ред f_i на $S_1(f,x)$.

Извод на формулите за линеен сплайн $S_1(f,x)$

Във всеки подинтервал $[x_{i-1}, x_i]$ търсим сплайна като интерполационен полином от 1-ва степен по x във вида:

$$\boxed{f_i(x) = a_i + b_i(x - x_{i-1})}. \qquad (11)$$

От условие 2) трябва $f_i(x_{i-1}) = y_{i-1}$ и $f_i(x_i) = y_i$. Т.е.

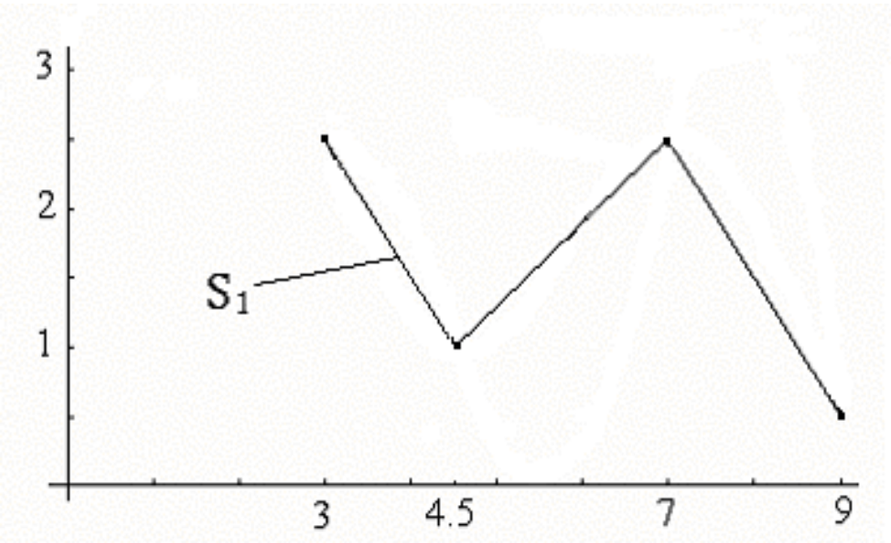
$f_i(x_{i-1}) = a_i + b_i(x_{i-1} - x_{i-1}) = y_{i-1}$, откъдето $a_i = y_{i-1}$ и

$f_i(x_i) = a_i + b_i(x_i - x_{i-1}) = y_i$ или $b_i = \dfrac{y_i - y_{i-1}}{h_i}$.

Обща формула на линеен сплайн	*Коефициенти на сплайна*
$S_1(f,x) = \begin{cases} f_1 = a_1 + b_1(x - x_0), & x \in [x_0, x_1] \\ \dots \\ f_i = a_i + b_i(x - x_{i-1}), & x \in [x_{i-1}, x_i] \\ \dots \\ f_n = a_n + b_n(x - x_{n-1}), & x \in [x_{n-1}, x_n] \end{cases}$	$a_i = y_{i-1},$ $b_i = \dfrac{y_i - y_{i-1}}{h_i}, \quad i = \overline{1,n}$

Пример. Дадена е следната таблица на функцията $y = f(x)$:

x_i	3	4.5	7	9
y_i	2.5	1	2.5	0.5



Да се построи линеен сплайн и с негова помощ да се намерят приближените стойности на функцията в точката $z = 5$.

Решение:

Изчисляваме стъпките: $h_1 = 4.5 - 3 = 1.5$; $h_2 = 7 - 4.5 = 2.5$; $h_3 = 9 - 7 = 2$.

По формулите пресмятаме последователно коефициентите a_i, b_i:

при $i = 1$, интервал $[3, 4.5]$: $a_1 = y_0 = 2.5$, $b_1 = \frac{y_1 - y_0}{h_1} = \frac{1 - 2.5}{1.5} = -1$;

при $i = 2$, интервал [4.5, 7]: $a_2 = y_1 = 1$, $b_2 = \frac{y_2 - y_1}{h_2} = \frac{2.5-1}{2.5} = 0.6$;

при $i = 3$, интервал [7, 9]: $a_3 = y_2 = 2{,}5$,

$$b_3 = \frac{y_3 - y_2}{h_3} = \frac{0.5-2.5}{2} = -1.$$

Нанасяме коефициентите и получаваме следната таблица:

i	a_i	b_i	Линеен сплайн	
1	2.5	-1.0	$f_1 = 2.5-(x-3)$	при $x \in [3,\ 4.5]$
2	1.0	0.6	$f_2 = 1+0.6(x-4.5)$	при $x \in [4.5,\ 7]$
3	2.5	-1.0	$f_3 = 2.5-(x-7)$	при $x \in [7,9]$

За да изчислим приближената стойност на функцията в точката $z = 5$ виждаме, че тя се намира във втория подинтервал и ще се апроксимира по формулата за f_2. Тогава

$$f(5) \approx f_2(5) = a_2 + b_2(z - x_1) = 1 + 0.6(5 - 4.5) = 1.3 .$$

3.3 Квадратичен сплайн

При $k = 2$ съгласно свойство 1) търсеният сплайн във всеки подинтервал $[x_{i-1}, x_i]$ е полином от втора степен (част от парабола) и коефициентите му са $3n$ на брой. Всъщност търсим $3n$ неизвестни a_i, b_i, c_i – коефициенти на сплайна.

Извод на формулите за квадратичен сплайн $S_2(f, x)$

От определението за сплайн имаме условията:

1) $\boxed{S_2(x) = f_i(x) = a_i + b_i(x - x_{i-1}) + c_i(x - x_{i-1})^2}$ за $x \in [x_{i-1}, x_i]$, i=1,2,…,n (12)

2) $f_i(x_{i-1}) = y_{i-1}$, $f_i(x_i) = y_i$ i=1,…,n - интерполиране, ($2n$ условия)(13)

3) $S_2'(x)$ е непрекъсната във вътрешните точки $x_1, x_2, \ldots, x_{n-1}$

$$f_i'(x_i) = f_{i+1}'(x_i) \qquad (n\text{-1 условия}). \qquad (14)$$

Така получихме общо $3n$-1 условия за определяне на $3n$ неизвестни. Едно условие остава свободно, т.е. квадратичният

сплайн не е единствен и се определя със задаване на едно допълнително условие. Ако се зададе $b_1 = 0$ сплайнът се нарича **естествен.** Изчисляването на коефициентите му е рекурентно.

От условие (13а) получаваме със заместване на $x = x_{i-1}$ в (12)- $f_i(x_{i-1}) = y_{i-1} = a_i$, или $\boxed{a_i = y_{i-1}}$ (15)

От (13б) $f_i(x_i) = a_i + b_i(x_i - x_{i-1}) + c_i(x_i - x_{i-1})^2 = a_i + b_i h_i + c_i h_i^2 = y_i$. Оттук

$$b_i + c_i h_i = \frac{y_i - y_{i-1}}{h_i} \quad . \tag{16}$$

Производните са: $f_i^{'}(x) = b_i + 2c_i(x - x_{i-1})$ и $f_{i+1}^{'}(x) = b_{i+1} + 2c_{i+1}(x - x_i)$.
Оттук за $x = x_i$ от (14) намираме: $b_{i+1} = b_i + 2c_i h_i$. Изразяваме

$$\boxed{c_i = \frac{b_{i+1} - b_i}{2h_i}} \quad . \tag{17}$$

Последното можем да заместим в (16) и излючвайки $c_i h_i$,

$$\boxed{b_{i+1} = -b_i + 2\frac{y_i - y_{i-1}}{h_i}}. \qquad (18)$$

Така стигаме до следната система уравнения (15),(17) и (18) за изчисляване коефициентите на сплайна S_2:

$$a_i = y_{i-1},$$

$$b_1 = \gamma_1,$$

$$b_{i+1} = -b_i + 2\frac{y_i - y_{i-1}}{h_i},$$

$$c_i = \frac{b_{i+1} - b_i}{2h_i}, \quad i = \overline{1,n} \ ,$$

където γ_1 е някакво начално значение на b_1. Ако $\gamma_1 = 0$ сплайнът е естествен.

Формулите са:

Обща формула на квадратичен сплайн	*Коефициенти на сплайна*
$S_2 = \begin{cases} f_1 = a_1 + b_1(x - x_0) + c_1(x - x_0)^2, & x \in [x_0, x_1] \\ \dots \\ f_i = a_i + b_i(x - x_{i-1}) + c_i(x - x_{i-1})^2, & x \in [x_{i-1}, x_i] \\ \dots \\ f_n = a_n + b_n(x - x_{n-1}) + c_n(x - x_{n-1})^2, & x \in [x_{n-1}, x_n] \end{cases}$	$a_i = y_{i-1}$, $b_1 = \gamma_1$, $b_{i+1} = -b_i + 2\frac{y_i - y_{i-1}}{h_i}$, $c_i = \frac{b_{i+1} - b_i}{2h_i}$, $i = \overline{1,n}$

Пловдивски университет „Паисий Хилендарски”
Факултет по математика и информатика
Катедра “Приложна математика и моделиране”

“Компютърни числени методи”
ЛЕКЦИЯ 7

Проф. д-р Снежана Гочева-Илиева, snow@uni-plovdiv.bg

Он-лайн обучение: www.fmi-plovdiv.org/evlm
www.fmi-plovdiv.org/evlm/DBbg - числени методи

Литература:

1. Бояджиев Д., Гочева С., Макрелов И., Попова Л. – Ръководство по числени методи – част 1, Издания: 2003, 2006, 2010.
2. Семерджиев Х., Боянов Б., Числени методи, ПУ.
3. Гочева-Илиева С., Въведение в система Mathematica, ЕксПрес, Габрово, 2009.

Съдържание

Интерполационен кубичен сплайн

1. Интерполационни сплайни – кубичен сплайн

Разгледахме случая на линеен и квадратичен интерполационен сплайн. Аналогично поставяме и задачата за кубичен сплайн.

Нека $y = f(x)$ е функция, дефинирана в интервала $[a,b]$ и е известна таблица от стойности $y_i = f(x_i)$ в точките (възлите) $a \le x_0 < x_1 < x_2 < \ldots < x_n \le b$. Стъпките между x_{i-1} и x_i ще означаваме с $h_i = x_i - x_{i-1}$. Нека таблицата е:

x_i	x_0	x_1	...	x_n
y_i	y_0	y_1	...	y_n

Дадохме следното *Определение.*

Интерполационният сплайн $S_k(f,x)$ от ред k е функция, за която:

(1) $S_k(f,x)$ е полином $f_i(x)$ от степен k във всеки подинтервал $[x_{i-1}, x_i]$, $i = \overline{1,n}$.

(2) $S_k(f,x)$ интерполира функцията, т.е. $S_k(f,x_i) = y_i$, $i = \overline{0,n}$.

(3) $S_k(f,x)$ и производните му до ред (k-1) са непрекъснати в $[a,b]$.

Кубичен сплайн

Тук $k = 3$ и във всеки подинтервал $[x_{i-1}, x_i], i = \overline{1,n}$, сплайнът $S_3(f,x)$ е полином от трета степен. Така търсим коефициентите a_i, b_i, c_i, d_i, които определят сплайна, т.е. $4n$ неизвестни.

Извод на формулите за кубичен сплайн $S_3(x)$

Аналогично на линеен и квадратичен сплайн търсим S_3 във вида:

$$S_3(x) = f_i(x) = a_i + b_i(x - x_{i-1}) + c_i(x - x_{i-1})^2 + d_i(x - x_{i-1})^3$$ за $x \in [x_{i-1}, x_i]$, i=1,... n,

при условията:

$f_i(x_{i-1}) = y_{i-1}$, $f_i(x_i) = y_i$ i=1,…,n - интерполиране, ($2n$ условия) (2)

$S_3'(x)$, $S_3''(x)$ са непрекъснати във вътрешните точки $x_1, x_2, ..., x_{n-1}$:

$$f_i'(x_i) = f_{i+1}'(x_i)$$ (n-1 условия), (3*)

$$f_i''(x_i) = f_{i+1}''(x_i)$$ (n-1 условия). (3**)

Или имаме общо $4n$-2 условия за намиране на $4n$ неизвестни. Две условия остават свободни. Следователно кубичният сплайн е единствен при задаване на две допълнителни условия. В таблицата по-долу те са означени с γ_1 и γ_2. По-точно тук разглеждаме условията: $S_3''(a) = 2c_1 = l_1 = \gamma_1$ **и** $S_3''(b) = 2c_n + 6d_n h_n = l_{n+1} = \gamma_2$.

Ако $\gamma_1 = \gamma_2 = 0$ сплайнът се нарича **естествен** кубичен сплайн.

Могат да се задават и редица други типове допълнителни условия, например $S_3'(a) = \gamma_1$ и $S_3'(b) = \gamma_2$, както и различни комбинации с по-горните условия, периодични условия и др.

Обща формула на кубичен сплайн

$$S_3(x)=\begin{cases} f_1(x)=a_1+b_1(x-x_0)+c_1(x-x_0)^2+d_1(x-x_0)^3, & x\in[x_0,x_1] \\ \dots \\ f_i(x)=a_i+b_i(x-x_{i-1})+c_i(x-x_{i-1})^2+d_i(x-x_{i-1})^3, & x\in[x_{i-1},x_i] \\ \dots \\ f_n(x)=a_n+b_n(x-x_{n-1})+c_n(x-x_{n-1})^2+d_n(x-x_{n-1})^3, & x\in[x_{n-1},x_n] \end{cases}$$

За намиране на коефициентите a_i, b_i, c_i, d_i:

А) От условия (2) при $i=\overline{1,n}$:

$$f_i(x_{i-1})=y_{i-1}=a_i \Rightarrow \boxed{a_i=y_{i-1}} \quad (4)$$

$$f_i(x_i)=y_i \Rightarrow a_i+b_i(x_i-x_{i-1})+c_i(x_i-x_{i-1})^2+d_i(x_i-x_{i-1})^3=\boxed{a_i+b_ih_i+c_ih_i^2+d_ih_i^3=y_i} \quad (5)$$

Б) За условие (3*) изчисляваме първите производни

отляво на x_i: $f_i'(x) = b_i + 2c_i(x - x_{i-1}) + 3d_i(x - x_{i-1})^2 \Rightarrow f_i'(x_i) = b_i + 2c_i h_i + 3d_i h_i^2$;

и отдясно: $f_{i+1}'(x) = b_{i+1} + 2c_{i+1}(x - x_i) + 3d_{i+1}(x - x_i)^2 \Rightarrow f_{i+1}''(x_i) = b_{i+1}$.

За да има непрекъснатост на $S_3'(x)$ във всяка вътрешна точка $x = x_i$ искаме лявата и дясната производна да са равни

$$\boxed{b_{i+1} = b_i + 2c_i h_i + 3d_i h_i^2} \quad (6)$$ при $i = \overline{1, n-1}$.

С) Аналогично за условие (3**) изчисляваме вторите производни

отляво на $x_i \Rightarrow f_i''(x) = 2c_i + 6d_i(x - x_{i-1}) \Rightarrow f_i''(x_i) = 2c_i + 6d_i h_i$

и отдясно на $x_i \Rightarrow f_{i+1}''(x) = 2c_{i+1} + 6d_{i+1}(x - x_i) \Rightarrow f_{i+1}''(x_i) = 2c_{i+1}$.

За да има непрекъснатост на $S_3''(x)$ двете трябва да са равни в x_i:

$$\boxed{2c_{i+1} = 2c_i + 6d_i h_i} \quad (7)$$ при $i = \overline{1, n-1}$.

Така получаваме системата от $4n$-2 уравнения (4)-(7) за a_i, b_i, c_i, d_i.

По-нататък означаваме за удобство $\boxed{l_i = 2c_i}$ и ще изразим всички коефициенти чрез тези неизвестни l_i.

Веднага $\boxed{c_i = \frac{l_i}{2}}$, при $i = \overline{1, n-1}$. (8)

От (7) изразяваме $d_i = \frac{2c_{i+1} - 2c_i}{6h_i}$ или

$\boxed{d_i = \frac{l_{i+1} - l_i}{6h_i}}$ при $i = \overline{1, n-1}$. (9)

Като заместим в (5) тези c_i, d_i и a_i от (4), намираме за b_i

$$b_i = \frac{y_i - a_i - c_i h_i^2 - d_i h_i^3}{h_i} = \frac{y_i - y_{i-1}}{h_i} - \frac{l_i h_i}{2} - \frac{(l_{i+1} - l_l)h_i}{6} \Rightarrow$$

$$\boxed{b_i = \frac{y_i - y_{i-1}}{h_i} - \frac{h_i(l_{i+1} + 2l_i)}{6}}. \quad (10)$$

Накрая замествайки в (6) изразите за a_i, c_i, b_i, d_i от (4),(8),(9),(10):

$$\frac{y_{i+1}-y_i}{h_{i+1}}-\frac{h_{i+1}(l_{i+2}+2l_{i+1})}{6}=\frac{y_i-y_{i-1}}{h_i}-\frac{h_i(l_{i+1}+2l_i)}{6}+l_ih_i+\frac{h_i(l_{i+1}-l_i)}{2}.$$

След елементарни преобразования намираме:

$$h_il_i+2(h_i+h_{i+1})l_{i+1}+h_{i+1}l_{i+2}=6\left(\frac{y_{i+1}-y_i}{h_{i+1}}-\frac{y_i-y_{i-1}}{h_i}\right),\ \text{при}\ i=\overline{1,n-1}.$$

Като добавим двете допълнителни условия при $i=0$ и $i=n$ получаваме напълно определена система относно спомагателните неизвестни l_i.

Разписана тя има тридиагонален вид с преобладаващ главен диагонал – виж (11). Може да се решава с метода на прогонването, който е устойчив (Защо?).

$$
\begin{aligned}
& l_1 && && && = \gamma_1 \\
& h_1 l_1 && +2(h_1+h_2) l_2 && +h_2 l_3 && = 6\left(\frac{y_2-y_1}{h_2} - \frac{y_1-y_0}{h_1}\right) \\
& && \ldots \\
& && h_{i-1} l_{i-1} && +2(h_{i-1}+h_i) l_i && +h_i l_{i+1} && = 6\left(\frac{y_i-y_{i-1}}{h_i} - \frac{y_{i-1}-y_{i-2}}{h_{i-1}}\right) \\
& && && \ldots \\
& && && && l_{n+1} && = \gamma_2
\end{aligned}
\qquad (11)
$$

Коефициенти на кубичния сплайн

$$
a_i = y_{i-1}, \quad b_i = \frac{y_i - y_{i-1}}{h_i} - \frac{h_i}{6}(l_{i+1} + 2l_i), \quad i = \overline{1,n}
$$

$$
c_i = \frac{l_i}{2}, \qquad d_i = \frac{l_{i+1} - l_i}{6\,h_i}, \qquad i = \overline{1,n-1}
\qquad (12)
$$

2. Алгоритъм за намиране на кубичен сплайн

и приближаване на стойност на функцията $y = f(x)$ в точка $x = \tilde{x}$:

1. Изчисляват се стъпките h_i.
2. Решава се тридиагоналната система (11).
3. Изчисляват се коефициентите на сплайна по (12).
4. Намира се в кой подинтервал се намира точката $\tilde{x}$, нека напр. това е интервалът $[x_{p-1}, x_p]$.
5. Изчислява се приближението по формулата в този подинтервал

$$\tilde{y} = f(\tilde{x}) = a_p + b_p(\tilde{x} - x_{p-1}) + c_p(\tilde{x} - x_{p-1})^2 + d_p(\tilde{x} - x_{p-1})^3.$$

Кубичен сплайн при равноотстоящи възли:

$$h = x_i - x_{i-1}, \quad i = \overline{1,n}$$

$$a_i = y_{i-1}, \quad b_i = \frac{y_i - y_{i-1}}{h} - \frac{h}{6}(l_{i+1} + 2l_i), \quad i = \overline{1,n}$$

$$c_i = \frac{l_i}{2}, \quad d_i = \frac{l_{i+1} - l_i}{6h}, \quad i = \overline{1,n-1}$$

$$\left|\begin{array}{llll}
l_1 & & & = \gamma_1 \\
l_1 & +4l_2 & +l_3 & = \frac{6}{h^2}\left(y_2 - 2y_1 + y_0\right) \\
 & \ldots & & \\
 & l_{i-1} & +4l_i \quad +l_{i+1} & = \frac{6}{h^2}\left(y_i - 2y_{i-1} + y_{i-2}\right) \\
 & & \ldots & \\
 & & l_{n+1} & = \gamma_2
\end{array}\right. , \qquad (13)$$

3. Теорема за оценка на грешката при интерполиране с кубичен сплайн (без доказателство).

Нека функцията $f(x) \in C_{q+1}[a,b]$, $0 \le q \le 3$.

То интерполационният кубичен сплайн $S_3(x)$, получен по таблица на функцията във възлите $x_0 < x_1 < \ldots < x_n$ удовлетворява оценките:

$$\max_{[x_k, x_{k+1}]} \left| f^{(i)}(x) - S_3^{(i)}(x) \right| \le C h^{q+1-i} \max_{x \in [a,b]} \left| f^{(q+1)}(x) \right|, \qquad (14)$$

където $k = 0,1,\ldots,n-1$, $i = 0,\ldots,q$, $h = \max_k h_k$, C е константа, независеща от k, h и f.

Пловдивски университет „Паисий Хилендарски”
Факултет по математика и информатика
Катедра “Приложна математика и моделиране”

“Компютърни числени методи”
ЛЕКЦИЯ 8

Проф. д-р Снежана Гочева-Илиева, snow@uni-plovdiv.bg

Он-лайн обучение: www.fmi-plovdiv.org/evlm
www.fmi-plovdiv.org/evlm/DBbg - числени методи

Литература:

1. Бояджиев Д., Гочева С., Макрелов И., Попова Л. – Ръководство по числени методи – част 1, Издания: 2003, 2006, 2010.
2. Семерджиев Х., Боянов Б., Числени методи, ПУ.
3. Гочева-Илиева С., Въведение в система Mathematica, ЕксПрес, Габрово, 2009.

Съдържание

Числено интегриране и оценка на грешката

1. Постановка на задачата за числено интегриране

Нека $f(x)$ е функция, дефинирана и непрекъсната в интервала $[a,b]$. Търси се стойността на определения интеграл $\boxed{I = \int_a^b f(x)dx}$. Определените интеграли служат за намиране на лица и обеми.

Числените методи за интегриране се налага да се използват:

- Когато не съществува примитивна функция (интегралът не се изразява с елементарни функции), напр. $\int_1^2 \frac{1}{\ln(x)} dx$, $\int_1^2 \frac{\sin(x)}{x} dx$ и др.
- когато примитивната е много сложен израз
- когато $f(x)$ може да е известна само от таблици.

<u>Идея на численото интегриране.</u> Функцията $f(x)$ да се приближи с подходяща функция $\varphi(x)$, която по-лесно може да се интегрира. Най-често $\varphi(x)$ е интерполационен полином.

Общ случай. Нека е известна таблица от стойности на y_i на функцията $f(x)$ в точките (възлите) $a \le x_0 < x_1 < x_2 < \ldots < x_n \le b$: (1)

x_i	x_0	x_1	...	x_n
y_i	y_0	y_1	...	y_n

И нека приближаващата функция е интерполационният полином на Лагранж, ($\varphi(x) = L_n(x)$), построена по тази таблица, т.е.

$$L_n(x) = \sum_{i=0}^{n} y_i \frac{(x-x_0)\ldots(x-x_{i-1})(x-x_{i+1})\ldots(x-x_n)}{(x_i-x_0)\ldots(x_i-x_{i-1})(x_i-x_{i+1})\ldots(x_i-x_n)} = \sum_{i=0}^{n} y_i\, l_i(x). \quad (2)$$

Тук $l_i(x)$ са функциите на Лагранж, независещи от $f(x)$.

Като включим и грешката на приближението $R_n(x)$, имаме:

$$f(x) = L_n(x) + R_n(x), \quad (3)$$

$$R_n(x) = \frac{M_{n+1}}{(n+1)!}\left|(x-x_0)(x-x_1)\ldots(x-x_n)\right|, \quad M_{n+1}(x) = \max_{[a,b]}\left|f^{(n+1)}(\xi)\right|. \quad (4)$$

Интегрираме двете страни на (2):

$$\int_a^b f(x)\,dx = \int_a^b \left[L_n(x) + R_n(x)\right]dx = \int_a^b L_n(x)dx + \int_a^b R_n(x)\,dx .$$

Ако грешката $R_n(x)$ е малка, то и интеграл от грешката $r_n = \int_a^b R_n(x)dx$ ще бъде малка. Тогава приемаме, че интегралът е приближено равен на интеграл от L_n:

$$I = \int_a^b f(x)\,dx \approx \int_a^b L_n(x)dx .$$

Сега да заместим L_n от (1):

$$I = \int_a^b f(x)dx \approx \int_a^b L_n(x)\,dx = \int_a^b \left[\sum_{i=0}^{n} y_i l_i(x)\right]dx = \sum_{i=0}^{n} y_i \int_a^b l_i(x)\,dx = \sum_{i=0}^{n} y_i A_i \qquad (5)$$

Получихме, че интегралът е една сума:

$$\boxed{I \approx \sum_{i=0}^{n} y_i A_i}, \qquad (6)$$

която се нарича ***квадратурна формула***. Може да бъде получена

и независимо от използването на L_n. Важно е грешката r да е малка.

2. Формули на Нютон-Коутс за числено интегриране

Когато полиномът на Лагранж се използва с равноотдалечени възли, то квадратурните формули се наричат формули на Нютон-Коутс.

2.1. Частен случай n=0: формули на правоъгълниците

Имаме само 1 възел:

x_0
y_0

Записваме интерполационния полином $L_0 = y_0$.

Тук грешката на приближението е $|R_0(x)| \le M_1 |x - x_0|$, където $M_1 = \max_{[a,b]} |f'(\xi)|$. Интегрираме в някакъв интервал, напр. $[x_0, x_1]$:

$$I = \int_{x_0}^{x_1} f(x)\,dx = \int_{x_0}^{x_1} L_0 dx + \int_{x_0}^{x_1} R_0 dx$$. Или $$I \approx \int_{x_0}^{x_1} L_0 dx = \int_{x_0}^{x_1} y_0 dx = y_0 \int_{x_0}^{x_1} dx = y_0(x_1 - x_0) = y_0.h$$.

Означаваме $h = x_1 - x_0$. Така получаваме формулата на левия правоъгълник: $\boxed{I \approx y_0.h}$, (7)

с грешка от численото интегриране: $r_{0,ляв} = \int_{x_0}^{x_1} R_0 dx$. Оценяваме я:

$$|r_0| \le \left| \int_{x_0}^{x_1} R_0 dx \right| \le M_1 \left| \int_{x_0}^{x_1} (x - x_0) dx \right| = M_1 \frac{(x - x_0)^2}{2!} \Bigg|_{x_0}^{x_1} = M_1 \frac{h^2}{2!} = O(h^2)$$. (8)

Този тип формули се наричат от отворен тип, защото интервалът не е фиксиран. Напр. ако интерполираме в т. x_1 и интегрираме в $[x_0, x_1]$ формулата се нарича формула на десния правоъгълник и ще има вида $I \approx y_1.h$, със същата оценка на грешката. Реално можем да интерполираме в която и да е точка от $[x_0, x_1]$ и получената формула ще има същата грешка и подобен вид.

Сумарна формула на правоъгълниците:

Получава се, като разделим интервала $[a,b]$ на n подинтервала с равни дължини, озн. с h, и приложим някоя от току-що изведените формули на правоъгълника във всеки подинтервал. Нека възлите са изчислени по формулата

$$x_i = a + ih, \quad i = 0,1,\ldots,n, \qquad h = \frac{b-a}{n}. \tag{9}$$

Имаме: $$I \approx \int_a^b = \int_{x_0}^{x_1} + \int_{x_1}^{x_2} + \ldots + \int_{x_{n-1}}^{x_n} = y_0 h + y_1 h + \ldots + y_{n-1} h = h \sum_{i=0}^{n-1} y_i \,. \tag{10}$$

Грешката не надминава сумата от грешките, т.е. от (8)

$$\left| r_{0\text{общ}a1} \right| \le n.M_1 \frac{h^2}{2} = h_1 \frac{(b-a)}{2} h = O(h), \tag{11}$$

тъй като $n.h = b - a$.

Пример. Да се пресметне по формулата на десните

правоъгълници $\int_2^3 \frac{\ln(x)}{x} dx$ за n=10.

<u>Решение.</u> Тук $[a,b]=[2,3]$, n=10, $h=\frac{b-a}{n}=\frac{3-2}{10}=\frac{1}{10}=0.1$.

Съгласно формула (10) трябва да изчислим сумата

$$I \approx \int_a^b = y_0 h + y_1 h + \ldots + y_{n-1} h = h \sum_{i=0}^{n-1} y_i .$$

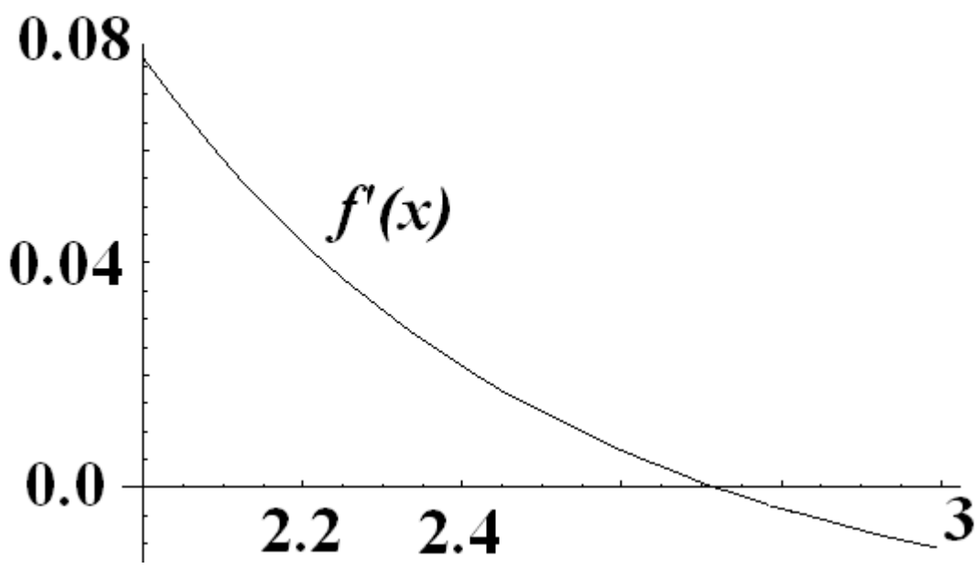
В нашия случай таблицата за интегриране при е:

x= {2., 2.1, 2.2, 2.3, 2.4, 2.5, 2.6, 2.7, 2.8, 2.9, 3.}

y= {0.346574, 0.353303, 0.35839, 0.362134, 0.364779, 0.366516, 0.367504, 0.367871, 0.367721, 0.367142, 0.366204}.

Събираме всички y_i, $i=0,1,\ldots,9$ (без последното!) и заместваме в

сумата: $I \approx h\sum_{i=0}^{n-1} y_i = 0.1(3.62193) = 0.362193$. Но не всички цифри тук са верни. Трябва да се направи оценка на грешката по формула (11). Търсим $M_1 = \max_{[2,3]} |f''(\xi)|$. За $f = \frac{\ln(x)}{x}$ изчисляваме $f' = \frac{1-\ln(x)}{x^2}$. Вместо да търсим максимум на тази функция, начертаваме графиката ѝ:



Виждаме, че по абс. стойност $|f'| \le 0.08$, следователно:

$$|r_{\text{обща}1}| \le h_1 \frac{(b-a)}{2} M_1 = (0.08)\frac{1}{2}(0.1) = 0.004.$$

Следователно не повече от 3 знака ще останат, закръгляме:

Отговор с правоъгълн.: $I \approx 0.362$.

2.2. Частен случай n=1: формула на трапците

Сега имаме 2 възела:

x_0	x_1
y_0	y_1

$h = x_1 - x_0$. Интерполационният полином е $L_1(x) = y_0 \frac{(x - x_1)}{(x_0 - x_1)} + y_1 \frac{(x - x_0)}{(x_1 - x_0)}$

или $\boxed{L_1(x) = y_0 \frac{(x - x_1)}{-h} + y_1 \frac{(x - x_0)}{h}}$.

Тук грешката на приближението е $R_1(x) = \frac{f''(\xi)}{2!}(x - x_0)(x - x_1)$, с оценка $\boxed{|R_1(x)| \le \frac{M_2}{2!}|(x - x_0)(x - x_1)|}$, където $M_2 = \max_{[a,b]} |f''(\xi)|$.

Представянето на функцията е: $f(x) = L_1(x) + R_1(x)$.

Интегрираме това равенство в $[x_0, x_1]$: $\boxed{I = \int_{x_0}^{x_1} f(x) = \int_{x_0}^{x_1} L_1 dx + \int_{x_0}^{x_1} R_1 dx}$.

Приближението за интеграла е интеграл от интерп. полином:

$$I \approx \int_{x_0}^{x_1} L_1(x)dx = \int_{x_0}^{x_1} (y_0 \frac{(x-x_1)}{-h} + y_1 \frac{(x-x_0)}{h})dx = \int_{x_0}^{x_1} (y_0 \frac{(x-x_1)}{-h})dx + \int_{x_0}^{x_1} (y_1 \frac{(x-x_0)}{h})dx =$$

$$\frac{y_0}{-h} \int_{x_0}^{x_1} (x-x_1)dx + \frac{y_1}{h} \int_{x_0}^{x_1} (x-x_0)dx = -\frac{y_0}{h} \frac{(x-x_1)^2}{2}\Big|_{x_0}^{x_1} + \frac{y_1}{h} \frac{(x-x_0)^2}{2}\Big|_{x_0}^{x_1} =$$

$$\frac{y_0}{h} \frac{(x_0-x_1)^2}{2} + \frac{y_1}{h} \frac{(x_0-x_1)^2}{2} = \frac{y_0}{h} \frac{h^2}{2} + \frac{y_1}{h} \frac{h^2}{2} = h \frac{y_0+y_1}{2}.$$

Така получихме формулата на трапеца:

$I \approx h \frac{y_0+y_1}{2}$, y_0, y_1 и височина h (12)

която геометрично вместо точния интеграл пресмята лицето на трапеца с основи

Грешката от численото интегриране с тази формула е интеграл от грешката на приближението:

$r_1 = \int_{x_0}^{x_1} R_1(x)dx$. Оценяваме я:

$$|r_1| \le \left| \int_{x_0}^{x_1} R_1 d \right| \le \left| \int_{x_0}^{x_1} \frac{M_2}{2}(x-x_0)(x-x_1)d \right| \le \frac{M_2}{2} \left| \int_{x_0}^{x_1} (x-x_0)(x-x_1)d \right| =$$

$$\frac{M_2}{2} \left| \int_{x_0}^{x_1} (x-x_0)^2 - h(x-x_0)d \right| = \ldots = M_2 \frac{h^3}{12} = O(h^3). \qquad (13)$$

Сумарна формула на трапците:

Получава се, като разделим интервала $[a,b]$ на n подинтервала с равни дължини, озн. с h, и приложим формула (12) във всеки подинтервал. Нека възлите са изчислени по формулата (9)

$x_i = a + ih, \quad i = 0,1,\ldots,n$, където стъпката е $h = \frac{b-a}{n}$.

Имаме:

$$I \approx \int_a^b = \int_{x_0}^{x_1} + \int_{x_1}^{x_2} + \ldots + \int_{x_{n-1}}^{x_n} = h\frac{y_0+y_1}{2} + h\frac{y_1+y_2}{2} + \ldots + h\frac{y_{n-1}+y_n}{2}$$, или

$$\boxed{I \approx \frac{h}{2}(y_0 + 2\sum_{i=1}^{n-1} y_i + y_n)}. \qquad (14)$$

Грешката не надминава сумата от грешките, т.е. от (13)

$$\boxed{\left|r_{1\text{обща}}\right| \le n.M_2\frac{h^3}{12} = \frac{h}{2}\frac{(b-a)}{12}h^2 = O(h^2)}, \qquad (15)$$

тъй като $n.h = b - a$.

<u>Пример.</u> Да се пресметне по формулата на трапеците същия интеграл $\int_2^3 \frac{\ln(x)}{x}dx$ за n=10.

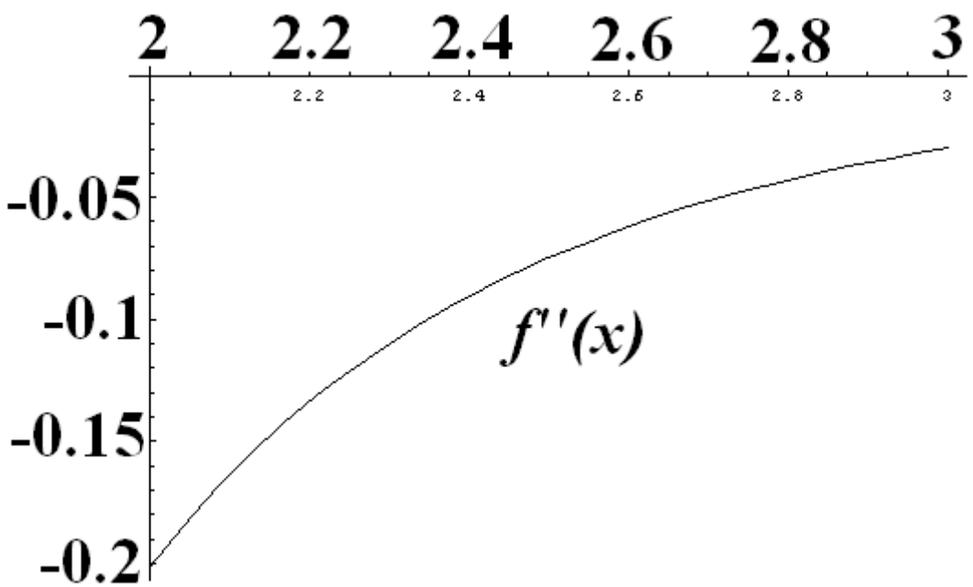
<u>Решение.</u> Интервалът е $[a,b] = [2, 3]$, n=10, $h = \frac{b-a}{n} = \frac{3-2}{10} = \frac{1}{10} = 0.1$.

Съгласно формула (14) трябва да изчислим сумата от значенията

на функцията във вътрешните точки, която се умножава по 2, да добавим стойностите от краищата и цялото да умножим по $h/2$.

$$I \approx \frac{h}{2}(y_0 + 2\sum_{i=1}^{n-1} y_i + y_n) = \frac{0.1}{2}(0.346574+2.3.27536+0.366204) = \frac{0.1}{2}7.2635=0.363175.$$

Оценяваме грешката по формула (15). Търсим $M_2 = \max_{[2,3]} |f''(\xi)|$. За



$f = \frac{\ln(x)}{x}$ изчисляваме $f' = \frac{1-\ln(x)}{x^2}$ и $f'' = \frac{-3+2\ln(x)}{x^3}$.

Вместо да търсим максимум на f'', начертаваме графиката ѝ:
Виждаме, че по абс. стойност $|f''| \le 0.2$, следователно:

$$\left|r_{1\text{общ}}\right| \le n.M_2 \frac{h^3}{12} = \frac{h}{2}\frac{(b-a)}{12}^2 = 0.2\frac{1}{12}(0.1)^2 = \frac{0.002}{12} \approx 0.0002$$

Следователно не повече от 4 знака ще останат, закръгляме:

Отговор с метода на трапеците: $I \approx 0.3632$.

2.3. Частен случай n=2: формула на Симпсън

Вече имаме 3 възела:

x_0	x_1	x_2
y_0	y_1	y_2

$h = x_1 - x_0 = x_2 - x_1$. Интерполационният полином е

$$L_2(x) = y_0 \frac{(x-x_1)(x-x_2)}{(x_0-x_1)(x_0-x_2)} + y_1 \frac{(x-x_0)(x-x_2)}{(x_1-x_0)(x_1-x_2)} + y_2 \frac{(x-x_0)(x-x_1)}{(x_2-x_0)(x_2-x_1)}$$ или

$$L_2(x) = y_0 \frac{(x-x_1)(x-x_2)}{h^2} + y_1 \frac{(x-x_0)(x-x_2)}{-h^2} + y_2 \frac{(x-x_0)(x-x_1)}{h^2}.$$

Тук грешката на приближението е $R_2(x) = \frac{f'''(\xi)}{3!}(x-x_0)(x-x_1)(x-x_2)$, с оценка $\boxed{|R_2(x)| \le \frac{M_3}{3!}|(x-x_0)(x-x_1)(x-x_2)|}$, където $M_3 = \max_{[a,b]} |f'''(\xi)|$.

Аналогично представянето на функцията е: $f(x) = L_2(x) + R_2(x)$.

Интегрираме това равенство в $[x_0, x_2]$: $\boxed{I = \int_{x_0}^{x_2} f(x) = \int_{x_0}^{x_2} L_2 dx + \int_{x_0}^{x_2} R_2 dx}$.

Приближението за интеграла е интеграл от интерп. полином:

$$I \approx \int_{x_0}^{x_2} L_2(x)dx.$$

Аналогично на предните два случая като се интегрира в интервала $[x_0, x_2]$ с дължина $2h$, след доста сметки се получава формулата на Симпсън:

$$\boxed{I \approx \int_{x_0}^{x_2} L_2(x)dx = \frac{h}{3}\left(y_0 + 4y_1 + y_2\right)}. \qquad (16)$$

Грешката от численото интегриране с тази формула е интеграл от грешката на приближението

$r_2 = \int_{x_0}^{x_2} R_2(x)dx$. След интегрирането се получава следната оценка:

$$\boxed{|r_2| \le M_4 \left|-\frac{h^5}{90}\right| = M_4 \frac{h^5}{90} = O(h^5)}, \qquad (17)$$

където $M_4 = \max_{[a,b]} \left| f^{(IV)}(\xi) \right|$. Тук се получава по-фина оценка, поради факта, че се използва симетрия относно средната точка x_1.

Сумарна формула на Симпсън:

Отново делим интервала $[a,b]$ на n подинтервала с равни дължини, озн. с h, но вече задължително имаме четен брой подинтервали $n = 2m$. Възлите са изчислени пак по формула (9)

$x_i = a + ih, \;\; i = 0,1,...,n$, където стъпката е $h = \frac{b-a}{n}$.

Като приложим формула (16) във всеки подинтервал имаме

$$I \approx \int_a^b = \int_{x_0}^{x_1} + \int_{x_1}^{x_2} + \ldots + \int_{x_{n-1}}^{x_n} =$$

$$\frac{h}{3}\left(y_0 + 4y_1 + y_2\right) + \frac{h}{3}\left(y_2 + 4y_3 + y_4\right) + \ldots + \frac{h}{3}\left(y_{2m-2} + 4y_{2m-1} + y_{2m}\right) =$$

$$\frac{h}{3}(y_0 + 4\sum_{i=1}^{m} y_{2i+1} + 2\sum_{i=1}^{m-1} y_{2i} + y_{2m}) \quad ,$$

$$\boxed{I \approx \frac{h}{3}(y_0 + 4\sum_{i=1}^{m} y_{2i+1} + 2\sum_{i=1}^{m-1} y_{2i} + y_{2m})}. \qquad (18)$$

Грешката не надминава сумата от грешките, т.е. от (17)

$$\boxed{\left|r_{1\text{общ}}\right| \le m.M_4 \frac{h^5}{90} = \frac{h}{4}\frac{(b-a)}{180}O(h^4) = O(h^4)}, \qquad (19)$$

тъй като $m.h = (b-a)/2$.

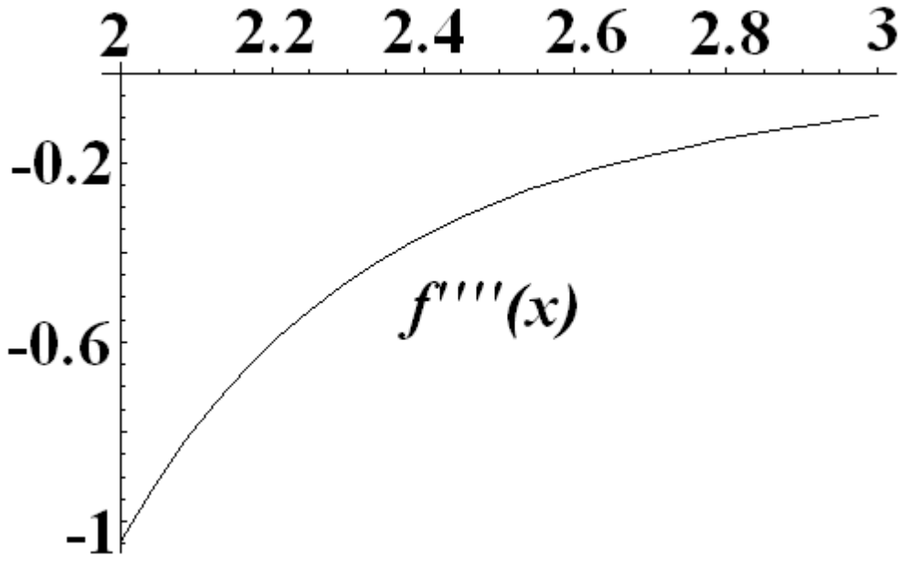
Пример. Да се пресметне по формулата на Симпсън същия интеграл $\int_2^3 \frac{\ln(x)}{x} dx$ за n=10 и да се оцени грешката.

Решение. Интервалът е $[a,b]=[2,3]$, n=10, m=5, $h=\frac{b-a}{n}=\frac{3-2}{10}=\frac{1}{10}=0.1$. Съгласно формула (18) трябва да изчислим две суми, като се внимава за различния брой събираеми!

$$I \approx \frac{h}{3}(y_0 + 4\sum_{i=1}^{m} y_{2i+1} + 2\sum_{i=1}^{m-1} y_{2i} + y_{2m}) = \frac{0.1}{3}(0.346574+4.1.81697+2.1.45839+0.366204) =$$

$$\frac{0.1}{3}10.8974 = 0.363248.$$

Оценяваме грешката по формула (19). Търсим $M_4 = \max_{[2,3]} |f''''(\xi)|$. Тя е $f'''' = \frac{-50+25\ln(x)}{x^5}$. Вместо да търсим максимума на тази функция, отново начертаваме графиката ѝ.



Виждаме, че по абс. стойност $|f''''| \le 1$, следователно:

$$|r_{1\text{общ}}| \le m.\ {}_4\frac{h^5}{90} = 1\frac{1}{180}(0.1)^4 = \frac{0.0001}{180} = 0.000006$$

Следователно до 5-ти знак закръгляме:

Отговор с метода на Симпсън: $I \approx 0.36325$.

Пловдивски университет „Паисий Хилендарски”
Факултет по математика и информатика
Катедра “Приложна математика и моделиране”

“Компютърни числени методи”
ЛЕКЦИЯ 9

Проф. д-р Снежана Гочева-Илиева, snow@uni-plovdiv.bg

Он-лайн обучение: www.fmi-plovdiv.org/evlm
www.fmi-plovdiv.org/evlm/DBbg - числени методи

Литература:

1. Бояджиев Д., Гочева С., Макрелов И., Попова Л. – Ръководство по числени методи – част 1, Издания: 2003, 2006, 2010.
2. Семерджиев Х., Боянов Б., Числени методи, ПУ.
3. Гочева-Илиева С., Въведение в система Mathematica, ЕксПрес, Габрово, 2009.

Съдържание

Числени методи за обикновени диференциални уравнения (ОДУ)

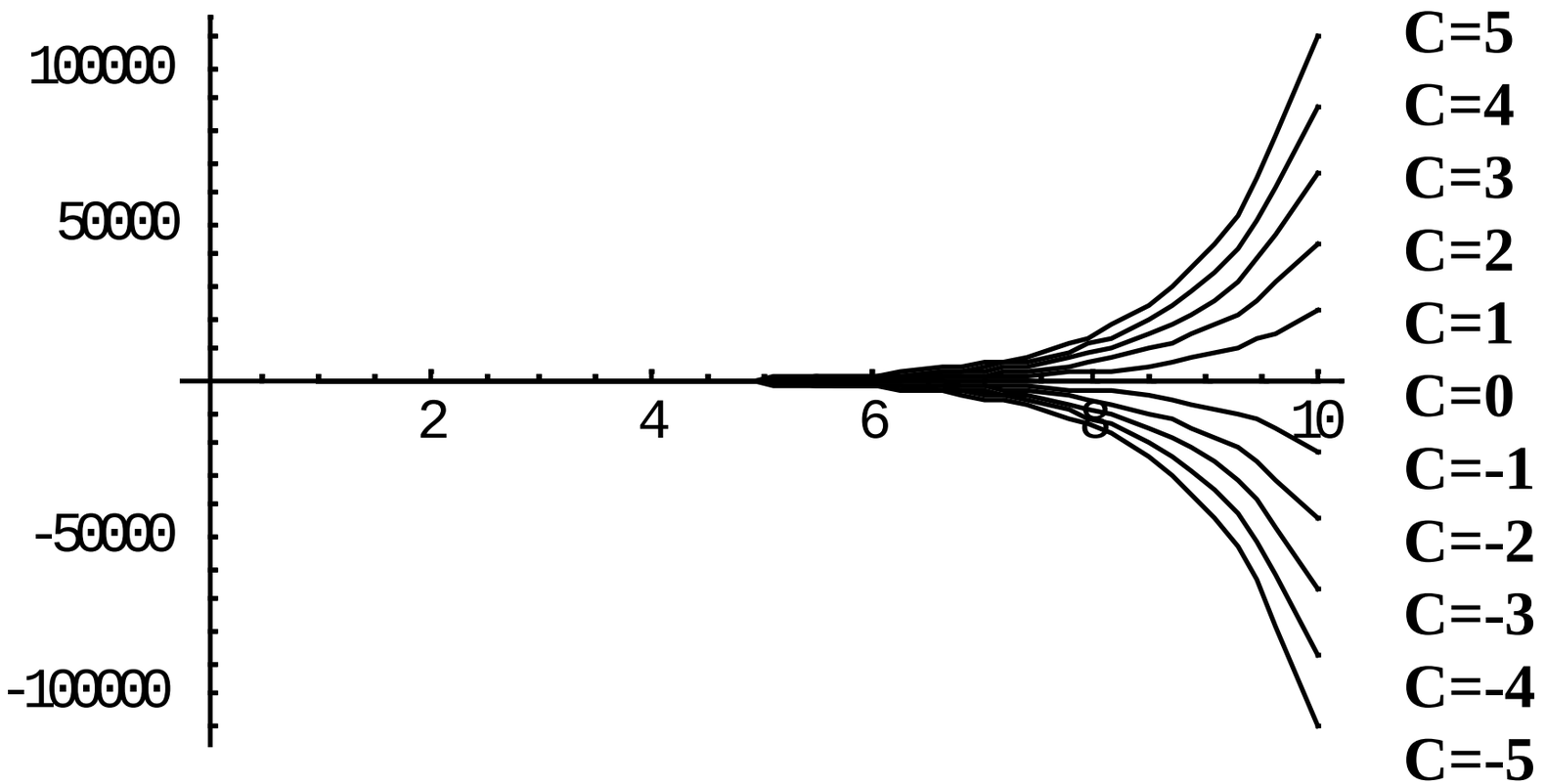
1. Преговор по ОДУ

Определение 1. Диференциално уравнение е уравнение, в което неизвестното е функция, заедно с производните й до някакъв ред. *Определение 2.* Ако неизвестната функция зависи само от една променлива, уравнението се нарича обикновено диференциално уравнение (ОДУ). Ако независимите променливи са две или повече, говорим за частно диференциално уравнение (ЧДУ).

Пример 1. ОДУ от първи ред е например уравнението:

$$y'(x) = y(x), \qquad (1)$$

където неизвестната функция $y(x)$ се търси в някакъв интервал $[x_0, x_0 + a]$. Очевидно едно решение на това уравнение е функцията $y(x) = e^x$. Очевидно е също, че и за всяка константа C, $y(x) = Ce^x$ също е решение, т.е. има безброй решения на уравнение (1).



Фиг. 1. Графика на част от общото решение на (1): $y(x) = Ce^x$ за C=-5, -4, ..., 5 в интервала $[1,10]$.

Определение 3. Множеството от всички решения на едно ОДУ се нарича ***общо решение.*** От теорията на ОДУ е известно, че общото решение на ОДУ от 1-ви ред зависи от една произволна константа, общото решение на ОДУ от 2-ри ред зависи от две произволни константи, от трети ред- от три константи и т.н.

Определение 4. Всяко отделно решение, което се получава от общото при конкретна стойност на константите, се нарича ***частно решение*** на даденото ОДУ.

Пример 2. ОДУ от 2-ри ред е уравнението: $y''(x) - \frac{x.y'(x) - y(x)}{x^2 + \sin(y(x))} = 0$.

Пример 3. Частно диференциално уравнение от втори ред:

$\frac{\partial U}{\partial t} = \frac{\partial^2 U}{\partial x^2} + \frac{\partial^2 U}{\partial y^2}$ с неизвестна ф-я $U = U(x, y, t)$ (параболично ЧДУ).

2. Постановка на задачата на Коши (начална задача за ОДУ)

Търси се решението $y = y(x)$ на задачата:

$$\begin{cases} y' = f(x,y) \\ y(x_0) = y_0 \end{cases}, \quad x \in [x_0, b]. \tag{2}$$

Вторият ред на задачата се нарича начално условие, тъй като от общото решение се търси онова частно решение, което в точката x_0 има дадена стойност $y(x_0) = y_0$.

Теорема на Коши. Ако дясната част $f(x,y)$ на уравнението (2) и частната й производна $\dfrac{\partial f(x,y)}{\partial y}$ са дефинирани и непрекъснати в област $G(x,y) \in \mathbb{R}^2$, то за всяка вътрешна точка $(x_0, y_0) \in G$ съществува единствено решение $y(x)$ на задача (2).

По-нататък ще считаме, че решението на (2) съществува.

3. Класификация на задачите за ОДУ

Могат да се формулират следните основни задачи:

А) задача на Коши за ОДУ от 1-ви ред

Б) задача на Коши за ОДУ от p-ти ред

В) задача на Коши за система ОДУ от първи ред

Г) гранична задача за ОДУ от 2-ри ред

Първата задача вече дефинирахме с (2). Тя може да се зададе и в т.н. неявен вид:

$$\begin{cases} f(x, y, y') = 0 \\ \varphi(x_0, y(x_0)) = 0, \quad x \in [x_0, b]. \end{cases} \tag{3}$$

Задача Б) може също да е в явен или неявен вид.

Напр.: Търси се решението на ОДУ (явен вид)

$$\left| \begin{array}{l} y^{(p)} = f(x, y, y', \ldots, y^{(p-1)}) \\ y(x_0) = y_0,\ y'(x_0) = y'_0,\ \ldots,\ y^{(p-1)}(x_0) = y_0^{(p-1)} \end{array} \right., \quad x \in [x_0, b]. \qquad (4)$$

В) - задача на Коши за система ОДУ от първи ред е:

Търсят се неизвестните функции $y_1(x), y_2(x), \ldots, y_p(x)$, които удовлетворяват системата уравнения:

$$\left| \begin{array}{l} y'_1 = f_1(x, y_1, y_2, \ldots, y_p) \\ y'_2 = f_2(x, y_1, y_2, \ldots, y_p) \\ \ldots \\ y'_p = f_p(x, y_1, y_2, \ldots, y_p) \end{array} \right., \quad x \in [x_0, b]. \qquad (5a)$$

и граничните условия:

$$\left|\begin{array}{l} y_1(x_0)=\alpha_1 \\ y_2(x_0)=\alpha_2 \\ \ldots \\ y_p(x_0)=\alpha_p \end{array}\right. . \qquad (5б)$$

По-нататък задачата (5а,б) ще представяме във векорен вид, аналогичен на (2):

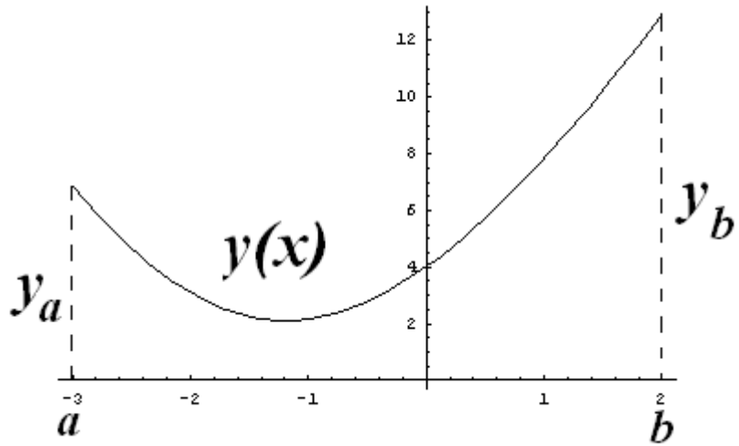
$$Y'=F(x,Y)$$
$$Y(x_0)=\mathrm{A} \quad , \quad x\in[x_0,b], \qquad (6)$$

където сме означили векторите с

$$Y=\begin{pmatrix} y_1 \\ y_2 \\ \ldots \\ y_p \end{pmatrix}, \quad F=\begin{pmatrix} f_1(x,y_1,y_2,\ldots,y_p) \\ f_2(x,y_1,y_2,\ldots,y_p) \\ \ldots \\ f_p(x,y_1,y_2,\ldots,y_p) \end{pmatrix}, \quad A=\begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \ldots \\ \alpha_p \end{pmatrix}.$$

Идея за задача Г) - гранична задача за ОДУ от 2-ри ред:

$$\left| \begin{array}{l} y'' = f(x, y, y'), \qquad x \in (a, b) \\ y(a) = y_a, \quad y(b) = y_b \end{array} \right. .$$



4. Свеждане на задача за уравнение от по-висок ред към система ОДУ.

За простота ще разгледаме случая за уравнение от 2-ри ред:

$$\begin{cases} y'' = f(x, y, y'), \quad x \in [x_0, b] \\ y(x_0) = \alpha_1 \\ y'(x_0) = \alpha_2 \end{cases}.$$

Полагаме: $\boxed{y'(x) = z(x)}$. Тогава $y''(x) = z'(x)$ и получаваме веднага начална задача за системата ОДУ:

$$\begin{cases} y' = z, \\ z' = f(x, y, z), \quad x \in [x_0, b] \\ y(x_0) = \alpha_1 \\ z(x_0) = \alpha_2 \end{cases}.$$

или във векторен вид:

$$\left|\begin{array}{l} \begin{pmatrix} y' \\ z' \end{pmatrix} = \begin{pmatrix} z \\ f(x,y,z) \end{pmatrix}, \qquad [x_0, b] \\ \begin{pmatrix} y(x_0) \\ z(x_0) \end{pmatrix} = \begin{pmatrix} \alpha_1 \\ \alpha_2 \end{pmatrix} \end{array}\right. .$$

5. Едностъпкови методи. Метод на Ойлер. Оценка на локалната грешка. Пример.

Числените методи решават началната задача на Коши като търсят **таблица на неизвестната функция в избрано множество (мрежа) от точки в зададения интервал**. При необходимост по-нататък се интерполира или се прави друг тип приближение за намиране на приближена формула на решението.

Метод на Ойлер (1768 г.)- явен едностъпков метод

Решава се задачата на Коши:

$$\begin{cases} y' = f(x,y) \\ y(x_0) = y_0 \end{cases}, \quad x \in [x_0, b]. \tag{2}$$

В интервала $[x_0, b]$ се и избират n точки: $x_0, x_1, ..., x_{n-1}, x_n = b$. Най-често това става равномерно, като се изчислява стъпка $h = (b - x_0)/n$ и точките се намират по формулата: $x_i = x_0 + ih, \ \ i = 0, 1, 2, ..., n$. Тогава $x_1 - x_0 = h, \ \ x_2 - x_1 = h, ...$

Извод на формулата за метода на Ойлер:

Развиваме функцията $y(x)$ в $x_1 = x_0 + h$ по формулата на Тейлър около точката x_0. Имаме:

$$y(x_1) = y(x_0 + h) = y(x_0) + h\, y'(x_0) + \frac{h^2}{2!} y''(\xi), \quad \xi \in (x_0, x_1). \tag{8}$$

Тъй като търсим функция, която удовлетворява уравнението и началното условие в (2), то $y'(x_0) = f(x_0, y_0)$ и $y(x_0) = y_0$. Получаваме:

$$y(x_1) = y_0 + hf(x_0, y_0) + \frac{h^2}{2!} y''(\xi), \quad \xi \in (x_0, x_1).$$

Ако тук последното събираемо разглеждаме като локална

грешка за точката x_0, намираме приближена формула за $y(x_1)$:

$$\boxed{y(x_1) = y_0 + hf(x_0, y_0)}. \qquad (9)$$

Следователно като знаем решението в (x_0, y_0) от (9) намираме решението в съседната точка (x_1, y_1), като сме означили $y_1 = y(x_1)$. Грешката на този етап при ограничена втора производна е:

$$R(x) = \frac{h^2}{2!} y''(\xi), \quad \text{или оценена отгоре е:} \quad |R(x)| \le \frac{h^2}{2!} \max_{[x_0, x_1]} |y''(\xi)| = O(h^2) \qquad .(10)$$

Обща формула за метода на Ойлер (стъпка след стъпка):

$$\boxed{y(x_{i+1}) = y_i + hf(x_i, y_i)}, \quad i = 0, 1, \ldots, n-1. \qquad (11)$$

Пример 4.

Задача. Да се реши по метода на Ойлер началната задача : $y' = \sqrt{x + y} + y \cos[x y],\ y(1) = 1$ **в интервала [1, 2] и стъпка h = 0.05 .**

Решение : Дефинираме функцията по правилата на система *Mathematica* и програмираме :

```
f[x_, y_] := Sqrt[x + y] + y Cos[x y]
n = 20; a = 1.; b = 2; h = (b - a)/n;
```

```
x = a; y = 1.;
res = {{"i", "x", "y", "f(x,y)"}, {0, x, y, f[x, y]}};
For[i = 0, i < n, i++, fn = f[x, y];
    x = x + h; y = y + h * fn;                    (* Формула на Ойлер *)
    res = Append[res, {i + 1, x, y, fn}]]
TableForm[res]
```

i	x	**y**	f(x,y)
0	1.	**1.**	1.95452
1	1.05	**1.09773**	1.95452
2	1.1	**1.19329**	1.9113
3	1.15	**1.28424**	1.81903
4	1.2	**1.36827**	1.68064
5	1.25	**1.44354**	1.50534
6	1.3	**1.50889**	1.30701
7	1.35	**1.56395**	1.10125
8	1.4	**1.60906**	0.90222
9	1.45	**1.64509**	0.720528
10	1.5	**1.67322**	0.562579
11	1.55	**1.69477**	0.431085

12	1.6	**1.71108**	0.326119
13	1.65	**1.72339**	0.246214
14	1.7	**1.73285**	0.189244
15	1.75	**1.74051**	0.153034
16	1.8	**1.74729**	0.13575
17	1.85	**1.7541**	0.136144
18	1.9	**1.76179**	0.153732
19	1.95	**1.77123**	0.188971
20	2.	**1.78341**	0.243499

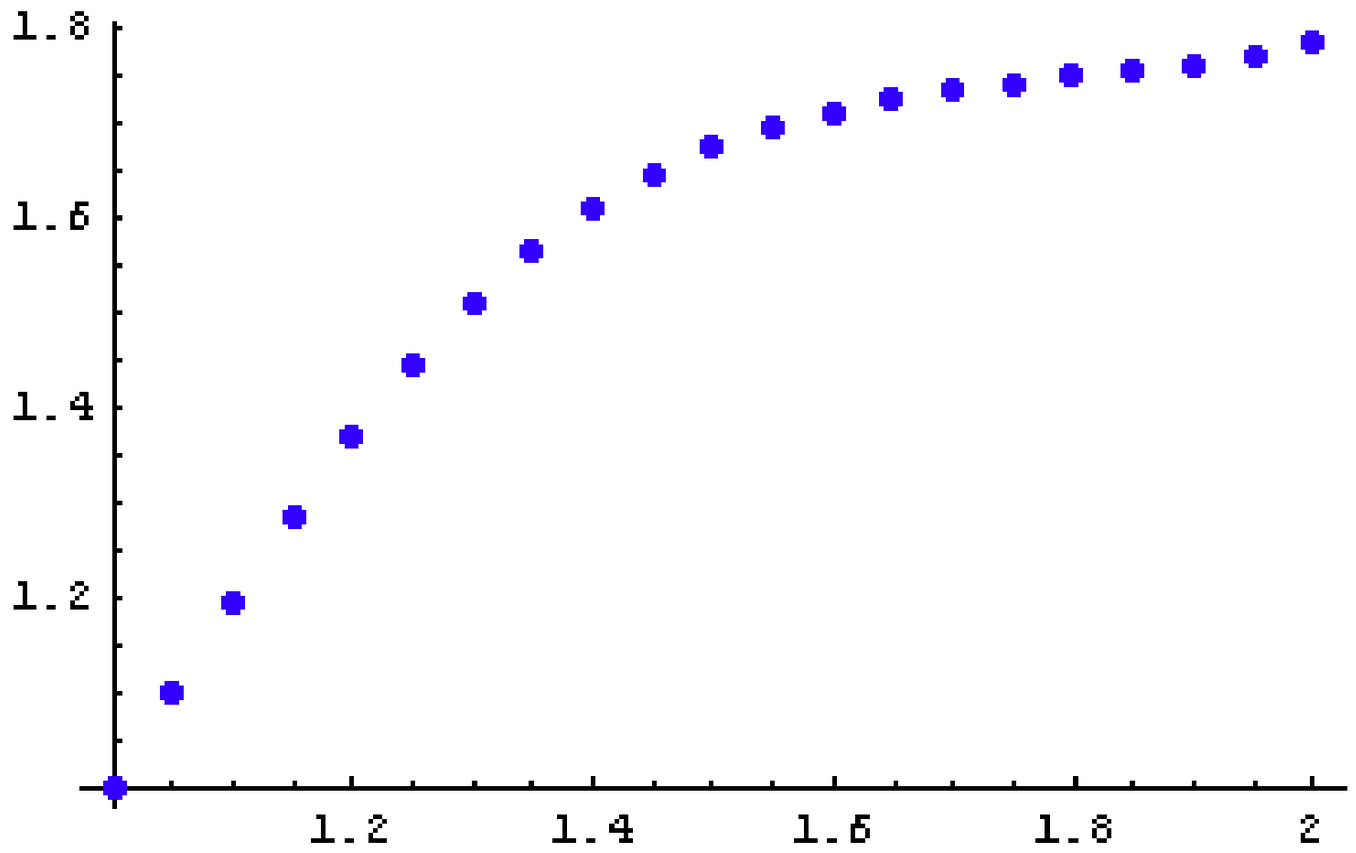
<u>Заключение:</u> Решението по метода на Ойлер е във вид на таблица на неизвестната функция y(x) - съответно втора колона от горната таблица. Тъй като теоретичната локална грешка на метода е O(h^2) и тук h=0.05, то тези решения трябва да се вземат закръглени само с три знака след десетичната точка. За уточняване може да се използва принципа на Рунге.

Графика на решението

```
res2 = Drop[res, 1]; res2 = res2[[All, {2, 3}]]; TableForm[res2]
  (* Извличат се само вторите и третите компоненти от всички елементи
   на списъка, т.е. {x,y}*)
gr1 = ListPlot[res2, PlotStyle → {Hue[0.7], PointSize[0.02]}]
  (* Чертае се точковата графика на решението y(x) *)
```

1.	1.
1.05	1.09773
1.1	1.19329
1.15	1.28424
1.2	1.36827
1.25	1.44354
1.3	1.50889
1.35	1.56395

1.4	1.60906
1.45	1.64509
1.5	1.67322
1.55	1.69477
1.6	1.71108
1.65	1.72339
1.7	1.73285
1.75	1.74051
1.8	1.74729
1.85	1.7541
1.9	1.76179
1.95	1.77123
2.	1.78341


1.8
1.6
1.4
1.2
1.2
1.4
1.6
1.8
2

Пример 5.

Дадено : Начална задача за система от обикновени диференциални уравнения от вида :

$y' = f(x, y, z),\ z' = g(x, y, z),\ x \in [a, b],\ y(a) = y_0,\ z(a) = z_0.$

Цел : Да се намери приближено решение

$\{y(x),\ z(x)\}$ **на задачата по метода на Ойлер при зададена стъпка h.**

Теория : Това е едностъпков метод,
при който от решението $\{y_i,\ z_i\}$ **в една точка** x_i **се изчислява решението** $\{y_{i+1},\ z_{i+1}\}$
в съседната точка $x_{i+1} = x_i + h$ **по формулите**

$y_{i+1} = y_i + h.f(x_i, y_i, z_i),\ z_{i+1} = z_i + h.g(x_i, y_i, z_i),\ i = 0, 1, \ldots$ **и** $\{y_0,\ z_0\}$ **са дадените начални стойности.**

Задача. Да се реши системата : $y' = x + y + z^2,\ z' = \frac{y+z}{1+x^2},\ y(1) = 1,\ z(1) = -1$ **в интервала [1, 2] и стъпка h = 0.1**

Решение :

Дефинираме функциите по правилата на система *Mathematica* :

```
f[x_, y_, z_] := x + y + z^2;   g[x_, y_, z_] := (y + z)/(1. + x^2)
h = 0.1; n = 10; a = 1.; b = 2;
```

Задачата програмираме като запомняме всички решения в списък :

```
x = a; y = 1.; z = -1.;
res = {{x, y, z}};  (*Начално зареждане на списъка за резултата- res *)
For[i = 0, i < n, i++,              (* i брояч *)
   fn = f[x, y, z]; gn = g[x, y, z];
   x = x + h; y = y + h * fn; z = z + h * gn; (* Формули за пресмятане на x,y,z *)
   res = Append[res, {x, y, z, fn, gn}](* Добавяне към края на списъка на поредните x,
   y,z,fn,gn *) ]
 TableForm[res]
```

1.	1.	-1.	fn	gn
1.1	1.3	-1.	3.	0.
1.2	1.64	-0.986425	3.4	0.135747
1.3	2.0213	-0.959639	3.81303	0.267858
1.4	2.44552	-0.920172	4.24221	0.394671
1.5	2.91475	-0.86864	4.69224	0.515322
1.6	3.43168	-0.805683	5.16928	0.629572
1.7	3.99976	-0.731919	5.6808	0.737639
1.8	4.6233	-0.647913	6.23546	0.840061
1.9	5.30761	-0.554154	6.8431	0.937592
2.	6.05908	-0.451042	7.5147	1.03112

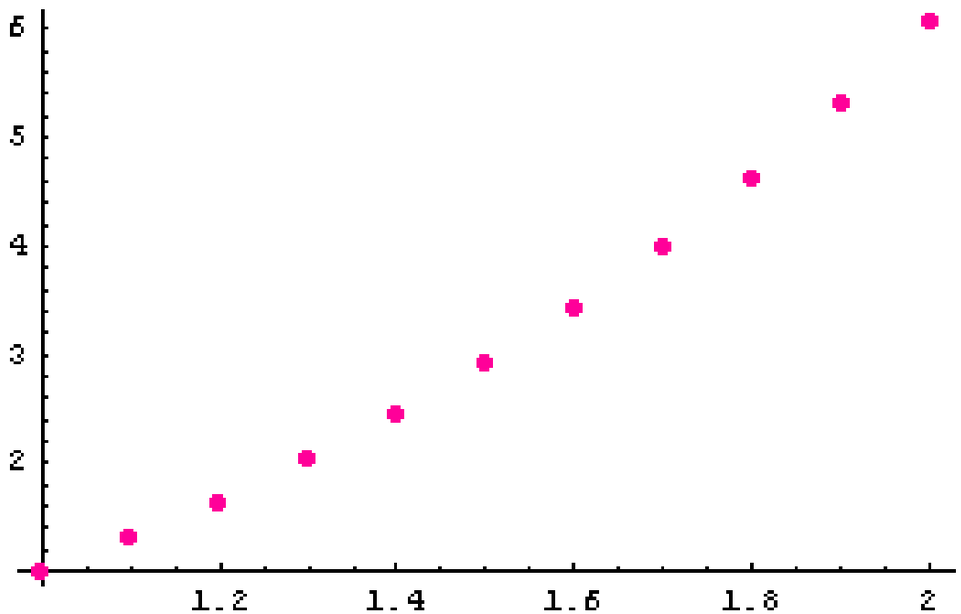
Заключение: Решението по метода на Ойлер е във вид на таблица на неизвестните функции y(x), z(x) - съответно втора и трета колона от горната таблица. Тъй като априорната грешка на метода е $O(h^2)$ и тук h=0.1, то тези решения трябва да се вземат закръглени само с два знака след десетичната точка.

Графика на решението

```
py = res[[All, {1, 2}]]
(* Извличат се само първите две компоненти от всички елементи
  на списъка, т.е. {x,y}*)
gr1 = ListPlot[py, PlotStyle → {Hue[0.9], PointSize[0.02]}]
(* Чертае се точковата графика на решението y(x) *)
```

Out[142]=

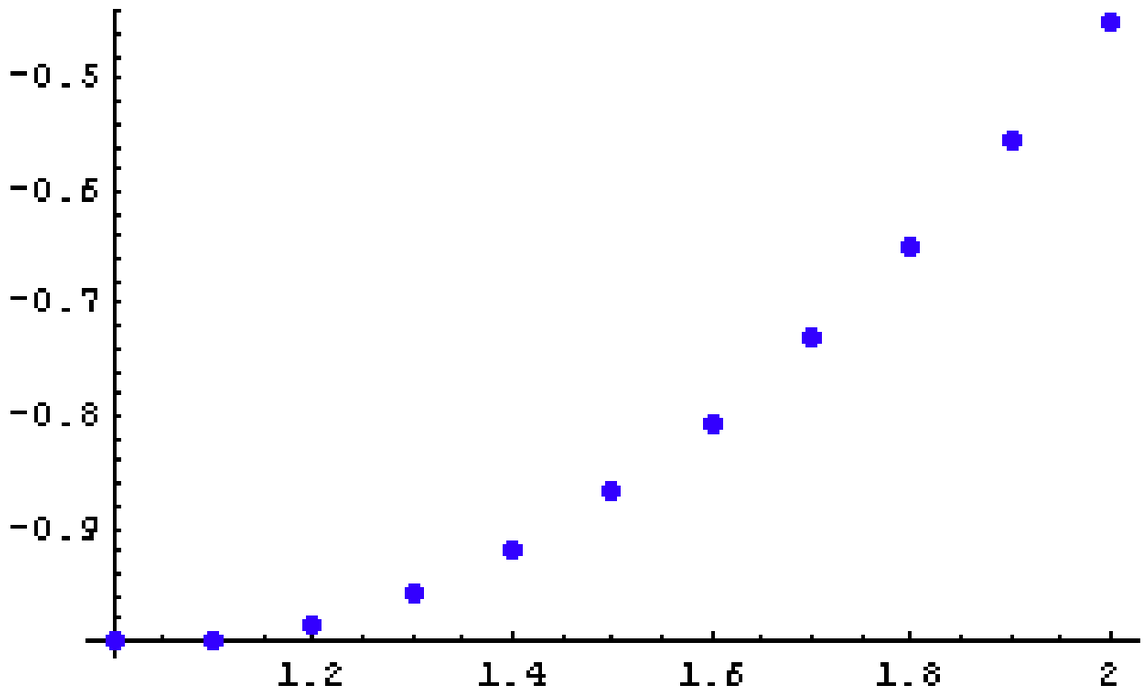
```
{{1., 1.}, {1.1, 1.3}, {1.2, 1.64}, {1.3, 2.0213}, {1.4, 2.44552},
 {1.5, 2.91475}, {1.6, 3.43168}, {1.7, 3.99976}, {1.8, 4.6233}, {1.9, 5.30761}, {2., 6.05908}}
```



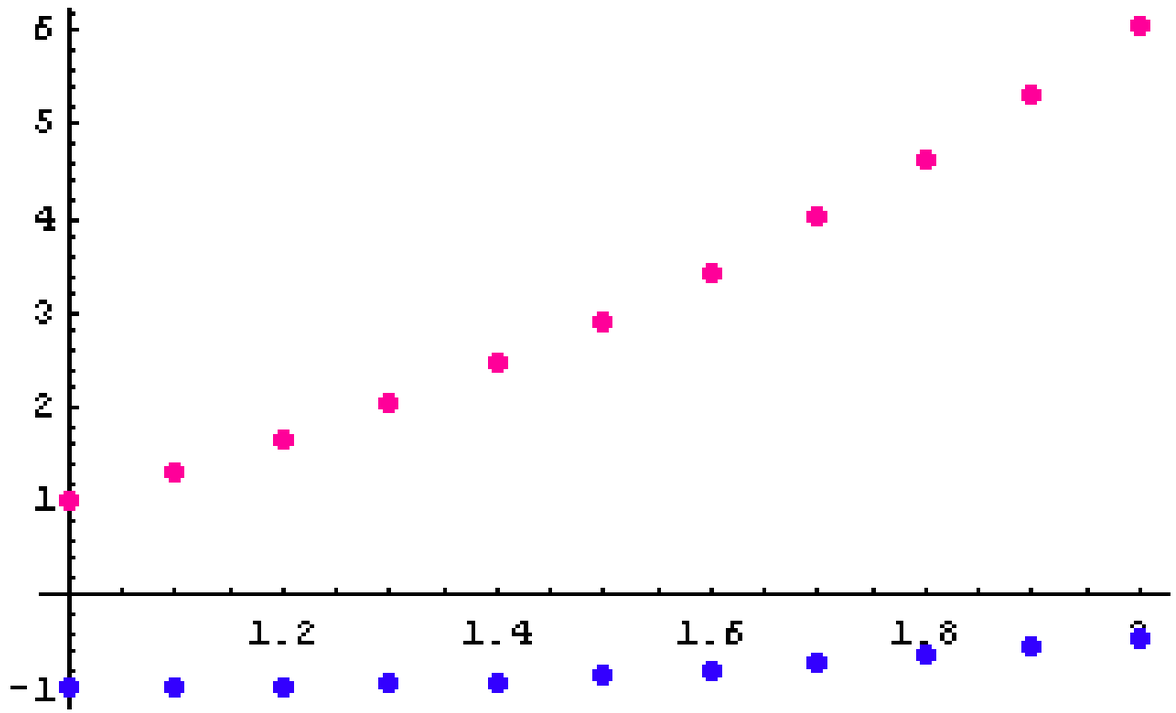
```
pz = res[[All, {1, 3}]]  (* Извличат се x и z компонентите от всички елементи на списъка *)
gr2 = ListPlot[pz, PlotStyle → {Hue[0.7], PointSize[0.02]}]
(* Чертае се точковата графика на решението z (x) *)
```

Out[143]=

```
{{1., -1.}, {1.1, -1.}, {1.2, -0.986425}, {1.3, -0.959639}, {1.4, -0.920172},
 {1.5, -0.86864}, {1.6, -0.805683}, {1.7, -0.731919}, {1.8, -0.647913}, {1.9, -0.554154}, {2., -0.451042}}
```



```
Show[gr1, gr2] (* Едновременно показва двете графики за y и z*)
```



6. Модифициран метод на Ойлер – явен едностъпков метод

Решава се отново същата задача на Коши:

$$\left|\begin{array}{l} y' = f(x,y) \\ y(x_0) = y_0 \end{array}\right. , \quad x \in [x_0, b]. \qquad (2)$$

Аналогично се избират n точки: $x_0, x_1, ..., x_{n-1}, x_n = b$ в интервала $[x_0, b]$.

Извод на формулата за модифицирания метод на Ойлер:

Използва се междинната точка $x_{i+\frac{1}{2}} = x_i + \frac{h}{2}$. Изчисленията са на два етапа:

1) в междинната точка $x_{i+\frac{1}{2}}$ се изчислява $\boxed{y_{i+\frac{1}{2}} = y_i + \frac{h}{2} f(x_i, y_i)}$ по формулата на Ойлер и функцията от $\boxed{f_{i+\frac{1}{2}} = f(x_{i+\frac{1}{2}}, y_{i+\frac{1}{2}})}$.

2) намира се търсеното $\boxed{y_{i+1} = y_i + hf_{i+\frac{1}{2}}}$. (12)

Оценка на локалната грешка в точката x_i. Отново развиваме функцията $y(x)$ в $x_{i+1} = x_i + h$ по формулата на Тейлър, но вече около точката $x_{i+\frac{1}{2}}$ (с половин стъпка $\frac{h}{2}$ напред от x_i). Имаме:

$$y_{i+1} = y(x_{i+1}) = y(x_{i+\frac{1}{2}} + \frac{h}{2}) = y_{i+\frac{1}{2}} + \frac{h}{2} y'_{i+\frac{1}{2}} + \frac{h^2}{4.2!} y''_{i+\frac{1}{2}} + \frac{h^3}{8.3!} y'''(\xi_1). \quad (13)$$

Също развиваме и с половин стъпка назад:

$$y_i = y(x_i) = y(x_{i+\frac{1}{2}} - \frac{h}{2}) = y_{i+\frac{1}{2}} - \frac{h}{2} y'_{i+\frac{1}{2}} + \frac{h^2}{4.2!} y''_{i+\frac{1}{2}} - \frac{h^3}{8.3!} y'''(\xi_2). \quad (14)$$

От (13) изваждаме (14) и намираме:

$$y_{i+1} - y_i = hy'_{i+\frac{1}{2}} + \frac{h^3}{8.3!}(y'''(\xi_1) + y'''(\xi_2)) = hf_{i+\frac{1}{2}} + O(h^3). \quad (15)$$

Като сравним с формулата на модифицирания метод на Ойлер (12) получаваме грешката между точното решение (15) и приближеното (12) във вида:

$$\left|R_{\mathrm{mod}\,Euler}(x_i)\right| \le \frac{h^3}{24} \max_{[x_i, x_{i+1}]} \left|y'''(\xi)\right| = O(h^3) \quad . \tag{16}$$

Забележка. Модифицираният метод на Ойлер има с един порядък по-добра грешка от обикновения метод на Ойлер, като сравним (16) с (10). Това обаче е в сила, ако търсената функция притежава ограничена трета производна, което е доста по-силно изискване.

7. Метод на Ойлер-Коши – двустъпков неявен метод

Отново за същата задача на Коши

$$\left| \begin{array}{l} y' = f(x,y) \\ y(x_0) = y_0 \end{array} \right. , \quad x \in [x_0, b]. \qquad (2)$$

се избират n точки: $x_0, x_1, \ldots, x_{n-1}, x_n = b$ в интервала $[x_0, b]$. Означаваме стандартно стъпката с h.

Пресмятанията се извършват на два етапа:

1) $\boxed{y_{i+1} = y_{i-1} + 2hf(x_i, y_i)}$ (17)

формула предиктор – явна двустъпкова, защото използва известни значения на y с 2 стъпки назад за точките x_{i-1}, x_i;

2) $\boxed{y_{i+1} = y_i + \frac{h}{2}\left(f(x_i, y_i) + f(x_{i+1}, y_{i+1}) \right)}$, (формула на трапеца) (18)

или формула коректор – неявна едностъпкова, защото неизвестното y_{i+1} участва отляво и отдясно във формулата

(18).

Така за всяка точка x_{i+1} отначало се прави приближение за търсеното y_{i+1} по формула предиктор. След което един или повече пъти това y_{i+1} се уточнява чрез итериране по формулата коректор (18), например така: $\boxed{y_{i+1}^{(0)} = y_{i-1} + 2hf(x_i, y_i)}$ - нулева итерация, и 1 път итерация по формулата $\boxed{y_{i+1}^{(1)} = y_i + \frac{h}{2}\left(f(x_i, y_i) + f(x_{i+1}, y_{i+1}^{(0)}) \right)}$. Допуска се и евентуално още една втора итерация по формулата коректор.

Забележка. Формулите са приложими, ако в началния момент знаем поне 2 стойности, т.е. y_0 и y_1. Първото е дадено по условие, а второто можем да изчислим еднократно например с модифициран метод на Ойлер.

Оценка на локалната грешка за Ойлер-Коши в точката x_i.

Отново се използва формулата на Тейлър.

Развиваме функцията $y(x)$ в $x_{i+1} = x_i + h$ и после в $x_{i-1} = x_i - h$ около точката x_i. Имаме:

$$y_{i+1} = y_i + hy'_i + \frac{h^2}{2!} y''_i + \frac{h^3}{3!} y'''(\eta_1)$$

$$y_{i-1} = y_i - hy'_i + \frac{h^2}{2!} y''_i - \frac{h^3}{3!} y'''(\eta_2) . \qquad (19)$$

Като ги извадим почленно получаваме:

$$y_{i+1} - y_{i-1} = 2hy'_i + \frac{h^3}{3!}\left(y'''(\eta_1) + y'''(\eta_2)\right) . \qquad (20)$$

Понеже $y(x)$ е решение на уравнението (2), то $y'_i = f_i = f(x_i, y_i)$. За локалната грешка на предиктора имаме оценка на последното събираемо

$$r_{prediktor} = y_{i+1} - y_{i-1} - 2hf_i = \frac{h^3}{3!}\left(y'''(\eta_1) + y'''(\eta_2)\right) = O(h^3), \quad \eta_1, \eta_2 \in (x_{i-1}, x_{i+1}) \ . \qquad (21)$$

За формулата коректор имаме:

$$y_{i+1} = y_i + hy'_i + \frac{h^2}{2!}y''_i + \frac{h^3}{3!}y'''(\zeta_1) \qquad (22)$$

Развиваме и производната около точката x_i, т.е.

$y'_{i+1} = y'_i + hy''_i + \frac{h^2}{2!}y'''(\zeta_2)$. Оттук изразяваме втората производна

$y''_i = \frac{y'_{i+1} - y'_i}{h} + \frac{h}{2!}y'''(\zeta_2)$ и я заместваме в (22). Веднага получаваме формулата коректор, заедно с локалната грешка:

$$y_{i+1} = y_i + \frac{h}{2}\left(f(x_i, y_i) + f(x_{i+1}, y_{i+1})\right) + O(h^3). \qquad (23)$$

Общо за целия метод

$$\left|R_{Euler-Cauchy}(x_i)\right| = O(h^3) \quad . \qquad (24)$$

8. Понятие за твърди диференциални уравнения

Много често се натъкваме на ОДУ, за които разгледаните дотук, както и други методи за числено интегриране не вървят. Започва да се наблюдава рязко скачане в решението и клонене към безкрайности, т.е. – неустойчивост. По принцип това свойство е поведение на самата задача, а не се дължи на числения метод.

Такива задачи се наричат твърди (stiff equations).

Решават се общо казано по следните начини:

А) чрез използване на стандартните числени методи, но със силно намалена стъпка h, например h =0.000001, докато решението започне да се стабилизира;

Б) чрез специални числени методи, наречени обратни или неявни методи.

9. Неявен метод на Ойлер

Когато дадена задача не се решава лесно и спада към твърдите диференциални задачи, т.е. има неустойчиво решение, се прилагат различни неявни методи. Най-простият такъв метод е т.н. неявен или обратен метод на Ойлер. Той се отличава от обикновения метод на Ойлер по това, че в дясната част на рекурентната формула (11) функцията $f(x,y)$ се изчислява в точката (x_{i+1}, y_{i+1}):

$$\boxed{y(x_{i+1}) = y_i + hf(x_{i+1}, y_{i+1})}, \quad \boxed{i = 0,1,\ldots,n-1}. \tag{25}$$

От тук се вижда, че на всяка стъпка i трябва да се решава нелинейното уравнение (25) относно неизвестната стойност на решението y_{i+1}.

Локалната грешка на неявния метод на Ойлер обаче,

независимо по-добрите му изчислителни качества (устойчивост на изчислителния процес) е от същия порядък, както обикновения метод на Ойлер, т.е. $O(h^2)$ - локална грешка на метода и $O(h)$ - глобална грешка.

Пловдивски университет „Паисий Хилендарски”
Факултет по математика и информатика
Катедра “Приложна математика и моделиране”

“Компютърни числени методи”
ЛЕКЦИЯ 10

Проф. д-р Снежана Гочева-Илиева, snow@uni-plovdiv.bg

Он-лайн обучение: www.fmi-plovdiv.org/evlm
www.fmi-plovdiv.org/evlm/DBbg - числени методи

Литература:

1. Бояджиев Д., Гочева С., Макрелов И., Попова Л. – Ръководство по числени методи – част 1, Издания: 2003, 2006, 2010.
2. Семерджиев Х., Боянов Б., Числени методи, ПУ.
3. Гочева-Илиева С., Въведение в система Mathematica, ЕксПрес, Габрово, 2009.

Съдържание

Числени методи за обикновени диференциални уравнения (ОДУ) - продължение

1. Методи на Рунге-Кута

Тези методи също са едностъпкови, могат да са явни или неявни. Решава се задачата на Коши:

$$\begin{cases} y' = f(x,y) \\ y(x_0) = y_0 \end{cases}, \quad x \in [x_0, b]. \tag{1}$$

Отново в интервала $[x_0, b]$ се и избират n точки: $x_0, x_1, ..., x_{n-1}, x_n = b$. Най-често точките са равномерно отдалечени, т.е. изчислява се стъпка $h = (b - x_0)/n$ и точките се намират по формулата: $x_i = x_0 + ih, \ i = 0,1,2,...,n$. Тогава $x_1 - x_0 = h, \ x_2 - x_1 = h, ...$

Новата, разширена идея спрямо методите на Ойлер е, че подобно на модифицирания метод на Ойлер, се избират няколко междинни точки във всеки подинтервал $[x_i, x_{i+1}]$. Тези точки не е задължително да са равноотдалечени, а се избират така, че при оценка на грешката с помощта на формулите на Тейлър да се

получава възможно най-висока степен s по h, т.е. локална грешка от вида $R(x_i) = O(h^s)$.

Най-известни формули на Рунге-Кута:

Формула (1,1):

$$k_1 = hf(x_i, y_i)$$
$$k_2 = hf(x_i + h, y_i + k_1)$$
$$y_{i+1} = y_i + \frac{1}{2}(k_1 + k_2), \quad i = 0,1,\ldots,n-1 \qquad (1,1)$$

Може да се покаже, че локалната грешка е $R(x_i) = O(h^3)$, а глобалната за целия интервал - $r = O(h^2)$.

Формула $(\frac{1}{2},\frac{1}{2})$: (подобна на модифициран метод на Ойлер)

$$k_1 = hf(x_i, y_i)$$

$$k_2 = hf(x_i + \frac{h}{2}, y_i + \frac{k_1}{2})$$

$$y_{i+1} = y_i + k_2$$

, $i = 0,1,...,n-1$ $(\frac{1}{2},\frac{1}{2})$

Локалната грешка $R(x_i) = O(h^3)$, глобална - $r = O(h^2)$.

Формула $(\frac{2}{3},\frac{2}{3})$:

$$k_1 = hf(x_i, y_i)$$

$$k_2 = hf(x_i + \frac{2}{3}h, y_i + \frac{2}{3}k_1)$$

$$y_{i+1} = y_i + \frac{1}{4}k_1 + \frac{3}{4}k_2$$

, $i = 0,1,...,n-1$ $(\frac{2}{3},\frac{2}{3})$

Локалната грешка $R(x_i) = O(h^3)$, глобална - $r = O(h^2)$.

Най-известен е следният метод на Рунге-Кута:

Формула с четири междинни точки:

$$k_1 = hf(x_i, y_i)$$

$$k_2 = hf\left(x_i + \frac{1}{2}h, y_i + \frac{1}{2}k_1\right)$$

$$k_3 = hf\left(x_i + \frac{1}{2}h, y_i + \frac{1}{2}k_2\right)$$

$$k_4 = hf(x_i + h, y_i + k_3)$$

$$y_{i+1} = y_i + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4), \quad i = 0,1,\ldots,n-1 \qquad (2)$$

Локалната грешка $R(x_i) = O(h^5)$, глобална - $r = O(h^4)$.

Забележка. Не трябва да се забравя, че грешка от типа $R(x_i) = O(h^s)$ е валидна, ако съществува и е ограничена производна $y^{(s)}(x)$.

2. Метод на Рунге-Кута за системи ОДУ

Силата на числените методи е във възможността по един и същи алгоритъм да се намира приближеното решение на големи класове от задачи, например при произволен вид на десните части на уравненията или редът им.

Ще запишем как се трансформират формулите на Р-К за система ОДУ от 2 уравнения. Обобщението за произволен брой уравнения е аналогично.

Нека се решава началната задача за системата ОДУ:

$$\begin{cases} y' = f(x, y, z), \\ z' = g(x, y, z), \quad x \in (x_0, b) \\ y(x_0) = y_0, \\ z(x_0) = z_0 \end{cases} . \tag{3}$$

Формула (1,1) за случая на системата (3):

$$k_1 = hf(x_i, y_i, z_i), \qquad l_1 = hg(x_i, y_i, z_i),$$

$$k_2 = hf(x_i + h, y_i + k_1, z_i + l_1) \qquad l_2 = hg(x_i + h, y_i + k_1, z_i + l_1)$$

$$y_{i+1} = y_i + \frac{1}{2}(k_1 + k_2), \qquad z_{i+1} = z_i + \frac{1}{2}(l_1 + l_2) \quad , \quad i = 0,1,\ldots,n-1 \qquad (4)$$

Локалната грешка е $R(x_i) = O(h^3)$, а глобалната за целия интервал е $r = O(h^2)$.

Пример: Да се реши по формула (4) уравнението:

$$\begin{cases} y'' = xy' + \sin(x.y^2 + y'), & x \in (1,2) \\ y(1) = 2, \\ y'(1) = 1. \end{cases}$$

.

Това уравнение се привежда към система с полагането (виж лекция 9): $\boxed{y'(x) = z(x)}$:

Получаваме началната задача за система ОДУ:

$$\left|\begin{array}{l} y' = z, \\ z' = x.z + \sin(x.y^2 + z), \qquad x \in (1,2] \\ y(x_0) = y_0, \quad x_0 = 1 \\ z(x_0) = z_0 \end{array}\right. .$$

За да приложим (4) е достатъчно тук да означим десните части така:

$$f(x, y, z) = z,$$

$$g(x, y, z) = x.z + \sin(x.y^2 + z) .$$

Формула с четири междинни точки за система (3):

$$
\begin{aligned}
k_1 &= hf(x_i, y_i, z_i), & l_1 &= hg(x_i, y_i, z_i),\\
k_2 &= hf\left(x_i + \frac{1}{2}h, y_i + \frac{1}{2}k_1, z_i + \frac{1}{2}l_1\right), & l_2 &= hg\left(x_i + \frac{1}{2}h, y_i + \frac{1}{2}k_1, z_i + \frac{1}{2}l_1\right),\\
k_3 &= hf\left(x_i + \frac{1}{2}h, y_i + \frac{1}{2}k_2, z_i + \frac{1}{2}l_2\right), & l_3 &= hg\left(x_i + \frac{1}{2}h, y_i + \frac{1}{2}k_2, z_i + \frac{1}{2}l_2\right),\\
k_4 &= hf(x_i + h, y_i + k_3, z_i + l_3), & l_4 &= hg(x_i + h, y_i + k_3, z_i + l_3),\\
y_{i+1} &= y_i + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4), & z_{i+1} &= z_i + \frac{1}{6}(l_1 + 2l_2 + 2l_3 + l_4).
\end{aligned}
\quad , \quad i = 0,1,\ldots,n-1 \qquad (5)
$$

3. Теорема за глобалната грешка при едностъпковите явни методи за ОДУ

Нека началната задача за ОДУ

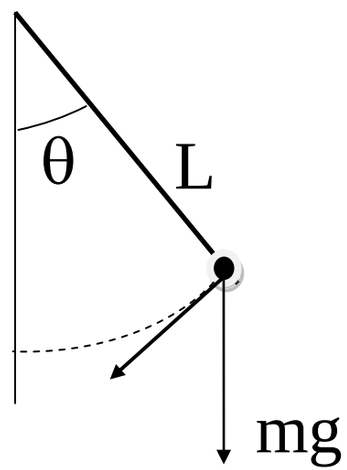
$$\left| \begin{array}{l} y' = f(x, y) \\ y(x_0) = y_0 \end{array} \right. , \quad x \in [x_0, b] . \tag{6}$$

се решава с някакъв едностъпков явен метод (Ойлер, модифициран метод на Ойлер, Ойлер-Коши, Рунге-Кута или друг). Нека точното решение в точката x_i е означено с $y(x_i)$, а приближеното, получено по избрания метод - с y_i. Нека локалната грешка на метода в точката x_i с $r_i = |y(x_i) - y_i| = O(h^{p+1})$

Може да се покаже, че при изчисленията на $y_1, y_2, \ldots, y_n$ локалната грешка се натрупва и глобалната грешка е:

$$\sum_{i=1}^{n} |r|_i \le Ch^p (x_n - x_0) = O(h^p)$$.

Задача: Да се пресметне свободното колебание на махало в съпротивителна среда.



Махалото се състои от материална точка с маса m, окачена на безмасова неразтеглива нишка с дължина L.

Нека $\theta = \theta(t)$ е ъгълът на отклонението от вертикалното положение, t – времето, a, b – константи, зависещи от L и теглото P=mg.

Уравнението, описващо $\theta(t)$ в зависимост от времето има вида:

$$\frac{d^2\theta}{dt^2} + a\frac{d\theta}{dt} + b\sin\theta = 0, \qquad t \in [t_0, t_0 + A],$$ където

$\theta(t_0) = \theta_0,$

$\theta'(t_0) = \theta'_0$, θ_0 - начален ъгъл на отклонение, θ'_0 - начална скорост.

Да се реши задачата при a=0.2, b=10, t_0=0, A=0.3 min, θ_0=0.5, θ'_0=0.

Това уравнение се привежда към система с полагането (виж лекция 9): $\boxed{\theta'(t) = u(t)}$. Получаваме системата ОДУ от вида (3):

$$\left| \begin{array}{l} \theta' = u \\ u' = -0.2u - 10\sin\theta, \quad t \in [0, 0.3] \end{array} \right. ,$$

с начални условия: $\theta(0) = 0.5,\ u(0) = 0$

Тази система може да се реши числено по метода на Рунге-Кута, напр. по формули (4) или (5).

4. Мрежов метод за граничната задача за ОДУ

4.1 Опростена линейна гранична задача от втори ред:

Да се намери функцията $u(x)$, която удовлетворява обикновеното диференциално уравнение

$$u''(x) - q(x).u(x) = f(x), \qquad a \le x \le b \tag{7}$$

с гранични условия

$$u(a) = \alpha, \qquad u(b) = \beta \tag{8'}$$

(гранични условия от първи род)

или

$$u'(a) - \alpha_1 u(a) = \beta_1,$$

$$u'(b) + \alpha_2 u(b) = \beta_2, \quad \alpha_1 \ge 0,\ \alpha_2 \ge 0 \tag{8''}$$

(гр. условия от втори род)

Могат да се налагат и смесен тип гранични условия.

В сила е следната теорема за съществуване и единственост на

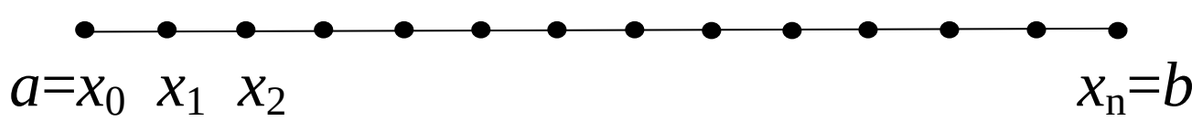
решението на опростената гранична задача:

Теорема: Ако функциите удовлетворяват условията: $q(x) \geq 0,\ q(x) \neq 0$ и $f(x)$ е интегруема в $[a,b]$, то съществува единствено решение на задача (7)- (8') или (7)- (8'').

Идея на мрежовия метод:

4.2 Апроксимация на интервала [a, b] с мрежа от точки.

За простота ще вземем равномерна мрежа от n на брой точки в интервала, със стъпка $h = \frac{b-a}{n}$. Получаваме мрежата: $\omega_h = \{x_k = a + kh,\ k = 0,1,...,n\}$.



Целта е да намерим приближените значения на решението $u(x)$ в тези точки. Ще означим търсените стойности с $u_k = u(x_k)$.

4.3 Апроксимация на уравнението (7).

Производната $u''(x)$ във всяка вътрешна точка x_k от мрежата ω_h представяме приближено с помощта на централната крайна разлика от втори ред:

$$\boxed{u''(x_k) \approx \frac{u_{k-1} - 2u_k + u_{k+1}}{h^2}, \quad k = 1, 2, \ldots, n-1}. \tag{9}$$

Това се получава от точното развитие

$$u''(x_k) = \frac{u_{k-1} - 2u_k + u_{k+1}}{h^2} - \frac{h^2}{12} u^{(\mathrm{IV})}(\xi), \quad \xi \in (x_{k-1}, x_{k+1}) \tag{10}$$

където последният член е пренебрегнат и последното събираемо е грешката от апроксимация. В случая грешката е от порядъка на $O(h^2)$, при условия, че съществува четвъртата производна на $u(x)$ и е ограничена в интервала $[a, b]$.

Заместваме точките x_k и приближението (9) в уравнение (7) и получаваме търсената апроксимация на уравнението във вида:

$$\frac{u_{k-1} - 2u_k + u_{k+1}}{h^2} - q(x_k)u_k = f(x_k), \quad k = 1, 2, \ldots, n-1 \qquad (11)$$

с грешка на приближението: $O(h^2)$.

Извод на формула (10):

Развиваме в ред на Тейлър $u(x)$ около т. x_{k-1}, x_{k+1}:

$$u_{k-1} = u(x_k - h) = u(x_k) - h.u'(x_k) + \frac{h^2}{2!}u''(x_k) - \frac{h^3}{3!}u'''(x_k) + \frac{h^4}{4!}u^{(\mathrm{IV})}(\xi_1)$$

$$u_{k+1} = u(x_k + h) = u(x_k) + h.u'(x_k) + \frac{h^2}{2!}u''(x_k) + \frac{h^3}{3!}u'''(x_k) + \frac{h^4}{4!}u^{(\mathrm{IV})}(\xi_2)$$

където $\xi_1 \in (x_{k-1}, x_k), \quad \xi_2 \in (x_k, x_{k+1})$.

Като съберем двете равенства и изразим $u''(x_k)$, получаваме (10), с усреднено формално $\xi \in (x_{k-1}, x_{k+1})$.

4.4 Апроксимация на граничните условия (8') или (8'').

В случая на гранични условия от първи род (8'), виждаме, че при k=0 и $k=n$ имаме съответно:

$$u_0 = \alpha, \quad u_n = \beta.$$

За приближаване на първата производна в гранични условия от втори род от типа (8'' използваме представянията:

при $k=0$: $$u'(x_0) = \frac{-3u_0 + 4u_1 - u_2}{2h} + O(h^2),$$

при $k=n$: $$u'(x_n) = \frac{u_{n-2} - 4u_{n-1} + 3u_n}{2h} + O(h^2)$$

Тези равенства се доказват подобно на извода на формула (10).

Като пренебрегнем грешките от порядъка на $O(h^2)$ и използваме (8') или (8"), получаваме окончателно:

За гр. усл. от първи ред:

$$u_0 = \alpha\,,\quad u_n = \beta\,. \tag{12'}$$

За гр.усл. от втори ред:

$$\frac{-3u_0 + 4u_1 - u_2}{2h} - \alpha_1 u_0 = \beta_1\,, \tag{12"}$$

$$\frac{u_{n-2} - 4u_{n-1} + 3u_n}{2h} + \alpha_2 u_n = \beta_2$$

Диференчна схема за граничната задача (7)- (8') или (7)- (8'')

Вместо изходната задача, за всяка точка от избраната мрежа $\omega_h = \{x_k = a + kh,\ k = 0,1,...,n\}$ след заместваме с приближенията (11)-(12') или (11)- (12'') и получаваме система алгебрични уравнения:

За задача (7)- (8'):

$$
\begin{cases}
u_0 = \alpha, & k = 0 \\
\dfrac{u_{k-1} - 2u_k + u_{k+1}}{h^2} - q(x_k)u_k = f(x_k), & k = 1,2,...,n-1 \\
u_n = \beta & k = n
\end{cases}
\qquad (11)\text{-}(12')
$$

За задача (7)- (8''):

$$\frac{-3u_0+4u_1-u_2}{2h}-\alpha_1 u_0=\beta_1, \qquad k=0$$

$$\frac{u_{k-1}-2u_k+u_{k+1}}{h^2}-q(x_k)u_k=f(x_k), \qquad k=1,2,\ldots,n-1$$

$$\frac{u_{n-2}-4u_{n-1}+3u_n}{2h}+\alpha_2 u_n=\beta_2, \qquad k=n$$

(11)-(12'')

с грешка на приближението: $O(h^2)$.

Получените системи алгебрични уравнения се свеждат към системи с тридиагонални матрици и могат да се решават например с метода на прогонването. Получените решения са приближения за търсените стойности на решението $u(x)$ в т. $x_k \in \omega_h$.

Например диференчната схема (11)-(12') се представя във вида:

$$\begin{cases} u_0 = \alpha, & k = 0 \\ u_{k-1} - (2 + h^2 q(x_k))u_k + u_{k+1} = h^2 f(x_k), & k = 1, 2, \ldots, n-1 \\ u_n = \beta & k = n \end{cases}$$

или по-подробно, с ясно видим преобладаващ главен диагонал:

$$\begin{cases} u_0 = \alpha \\ u_0 - (2 + h^2 q(x_1))u_1 + u_2 = h^2 f(x_1) \\ \quad \ldots \\ u_{k-1} - (2 + h^2 q(x_k))u_k + u_{k+1} = h^2 f(x_k) \\ \quad \ldots \\ u_n = \beta \end{cases} \qquad (13)$$

Теорема. Може да се докаже, че при условията от апроксимацията на производните с грешка $O(h^2)$, получените стойности $\tilde{u}_k$ след решаване на диференчните схеми е също с грешка $O(h^2)$ спрямо точното решение на граничната задача $u(x_k)$ в избраните точки от мрежата, т.е.

$$\max_k \left| u(x_k) - \tilde{u}_k \right| \leq Mh^2$$

където $M > 0$ е константа.

На практика, избирайки достатъчно гъста мрежа от точки в интервала [*a*, *b*], съответно – малка стъпка *h*, търсеното решение може да се намери с произволна зададена точност.

Към темата „Гранична задача от втори ред” е подготвен и пример със система Mathematica, за по-общ случай линейна гранична задача със смесени гранични условия от вида:

$$u''(x) + p(x)u'(x) + q(x)u(x) = f(x), \qquad a \le x \le b \tag{14}$$

$$u'(a) = \alpha u(a) + \beta, \quad u(b) = \gamma \quad \text{(смесени гр. условия)} \tag{15}$$

Решеният пример може да се намери в сайта на Европейската виртуална лаборатория, ФМИ:

http://www.fmi-plovdiv.org/evlm/DBbg/database/numan/ODU/mesh_method%20ODE/mesh_methodmixte.html