9. *Paprastųjų diferencialinių lygčių sprendimas*

Lygtys, kurių išraiškoje yra viena arba kelios išvestinės, vadinamos diferencialinėmis lygtimis. Pagal kintamųjų skaičių ir išvestinių tipą diferencialinės lygtys skirstomos į ***paprastąsias diferencialines lygtis*** (turinčias vieną kintamąjį ir išvestines šio kintamojo atžvilgiu) ir ***diferencialines lygtis su dalinėmis išvestinėmis*** (turinčias keletą kintamųjų ir dalines išvestines tų kintamųjų atžvilgiu).

Šiame skyriuje nagrinėsime paprastųjų diferencialinių lygčių (toliau — diferencialinių lygčių), plačiai taikomų fizikoje, technikoje ir matematikoje, sprendimo metodus.

Pirmuosius skaitinius diferencialinių lygčių sprendimo metodus sukūrė I. Niutonas (I. Newton) ir L. Oileris (L. Euler). XX amžiaus pradžioje jau buvo žinomi dabar tapę klasikiniais Rungės ir Kutos bei Adamso metodai. Sparčiai tobulėjant skaičiavimo technikai, apibendrinami senieji diferencialinių lygčių sprendimo metodai ir kuriami naujieji, orientuoti į lygtis, turinčias specialių savybių, pavyzdžiui, į standžias diferencialines lygtis.

Koši uždavinys ir kraštinis uždavinys. Norint skaitiniu būdu išspręsti diferencialinę lygtį, reikia žinoti ieškomos funkcijos ir jos išvestinės reikšmes. *Tos reikšmės, nusakytos viename taške, vadinamos pradinėmis sąlygomis, o pati lygtis su pradinėmis sąlygomis —* ***Koši uždaviniu***.

Ieškomosios funkcijos ir jos išvestinės reikšmės, nusakytos skirtinguose taškuose, vadinamos kraštinėmis sąlygomis, o pati lygtis — ***diferencialine lygtimi su kraštinėmis sąlygomis.***

Toliau nagrinėsime Koši uždavinį. Nesiaurindami jo, spręsime diferencialinę lygtį, kurios išraiška yra:

$$\begin{aligned} &y' = f(x,y), \\ &y(x_0) = y_0. \end{aligned} \qquad (9.1)$$

Taigi reikia rasti lygties sprendinį, t. y. funkciją $y = y(x)$*, tenkinančią tiek pačią lygtį, tiek pradines sąlygas.*

(9.1) diferencialinės lygties sprendimo metodai apibendrinami diferencialinių lygčių sistemai

$$\begin{cases} y_1' = f_1(x, y_1, y_2, \ldots, y_n), \\ y_2' = f_2(x, y_1, y_2, \ldots, y_n), \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ y_n' = f_n(x, y_1, y_2, \ldots, y_n), \end{cases} \qquad (9.2)$$

$$y_i(x_0) = y_{i0}, \ i = \overline{0, n}.$$

Pažymėkime: $y' = (y_1', y_2', \ldots, y_n')$, $y = (y_1, y_2, \ldots, y_n)$ ir $f = (f_1, f_2, \ldots, f_n)$. Tada (9.2) lygties vektorinė išraiška sutaps su (9.1) lygties išraiška. Vadinasi, y', y ir f traktuojant kaip vektorius, (9.1) diferencialinės lygties sprendimo metodai visiškai tiks (9.2) lygčių sistemai.

Aukštesnės eilės diferencialinę lygtį

$$y^{(n)} = f\left(x, y', \ldots, y^{(n-1)}\right),$$
$$y(x_0) = y_0,\ y'(x_0) = y'_0,\ \ldots,\ y^{(n-1)}(x_0) = y_0^{(n-1)}$$

galima paversti (9.2) lygčių sistema, vartojant tokį keitinį:

$$\begin{cases} y' = z_1, \\ z'_1 = z_2, \\ \ldots\ldots\ldots \\ z'_{n-2} = z_{n-1}, \\ z'_{n-1} = f(x, y, z_1, \ldots, z_{n-1}), \end{cases} \qquad (9.3)$$
$$y(x_0) = y_0,\ z(x_0) = y'_0,\ z_2(x_0) = y''_0,\ \ldots,\ z_{n-1}(x_0) = y_0^{(n-1)}.$$

(9.1) diferencialinės lygties sprendimo metodus galima suskirstyti į ***vienažingsnius*** ir ***daugiažingsnius***. Sprendžiant skaitiniu būdu (9.1) diferencialinę lygtį, apskaičiuojama sprendinio $y = y(x)$ reikšmių lentelė, kai $x_0 \le x \le x_{end}$.

Tarkime, kad žinome $y = y(x)$ reikšmes $(x_i; y_i)$; čia $i = \overline{1, m}$. Reikia rasti tašką $(x_{m+1}; y_{m+1})$; čia $x_{m+1} = x_m + h$, o h — integravimo žingsnis.

Vienažingsniams metodams būdinga tai, kad, norint apskaičiuoti y_{m+1} reikšmę, reikia žinoti tik x_m ir y_m reikšmes. Rungės ir Kutos metodai yra tipiškas vienažingsnių metodų pavyzdys.

Daugiažingsniams metodams būdinga tai, kad, norint apskaičiuoti y_{m+1} reikšmę, reikia žinoti taškus $(x_{m-j}; y_{m-j})$; čia $j = \overline{0, k}$. Tipiški šios klasės metodų pavyzdžiai yra Milno, Adamso ir Bašforto bei Hemingo metodai. Paprastai jie būna poriniai. Remiantis taškais $(x_{m-j}; y_{m-j})$ $(\text{čia } j = \overline{0, k})$, pirmiausia prognozuojama y_{m+1} reikšmė, paskui, taikant iteracinę procedūrą, ta reikšmė tikslinama. Todėl dažnai daugiažingsniai metodai dar vadinami ***prognozės ir korekcijos*** metodais.

Prieš pradėdami nagrinėti konkrečius diferencialinių lygčių sprendimo metodus, aptarkime skaičiavimo paklaidas. Skaičiuojant dėl metodo ir apvalinimo paklaidų susidaro ***lokalioji*** ir ***globalioji*** sprendinio ***paklaida.***

Lokaliąja vadinama ***paklaida***, kuri atsiranda dėl metodo ir apvalinimo paklaidų apskaičiuojant tašką $(x_{m+1}; y_{m+1})$ ir laikant, kad taškas $(x_m; y_m)$ yra tikslus, t. y. neturi paklaidos.

Globalioji paklaida — tai taško $(x_{m+1}; y_{m+1})$ suminė paklaida, susikaupusi nuo sprendimo pradžios, t. y. tikslios sprendinio reikšmės y_{m+1}^t ir apytikslės reikšmės y_{m+1} skirtumas.

9.1. Vienažingsniai metodai

9.1.1. Teiloro eilučių metodas

Spręsime (9.1) diferencialinę lygtį, t. y. lygtį

$$y' = f(x, y),$$
$$y(x_0) = y_0.$$

Teiloro eilučių metodas turi daugiau teorinę vertę ir yra etalonas, leidžiantis palyginti skirtingus diferencialinių lygčių sprendimo metodus.

Tarkime, kad $y = y(x)$ yra (9.1) diferencialinės lygties sprendinys. Funkciją $y = y(x)$ išskleiskime Teiloro eilute taško $x = x_m$ aplinkoje:

$$y = y(x_m) + y'(x_m)(x - x_m) + \frac{y''(x_m)}{2!}(x - x_m)^2 + \dots .$$

Pažymėkime: $y_m = y(x_m)$, $y'_m = y'(x_m)$, $y''_m = y''(x_m)$ ir t. t. Imkime $x = x_{m+1} = x_m + h$; čia h — integravimo žingsnis. Tada

$$y_{m+1} = y_m + y'_m h + y''_m \frac{h^2}{2!} + y'''_m \frac{h^3}{3!} + \dots . \qquad (9.4)$$

Jei imsime narius iki p-tosios eilės išvestinės imtinai, tai (9.4) formulė apibūdins p-tosios eilės vienažingsnį metodą.

Taikant (9.4) formulę, sunkiau apskaičiuoti aukštesnės eilės išvestines.

Aišku, kad $y'_m = f(x_m, y_m)$. Tada $y''_m = \frac{\partial f}{\partial x} + \frac{\partial f}{\partial y} y'_m$. Supaprastinus simboliką, galima parašyti: $y'' = f'_x + f'_y f$. Lengva pastebėti, kad aukštesnės eilės y išvestinės bus sudėtingesnės, todėl praktiškai jas stengiamasi skaičiuoti kitokiais būdais.

9.1.2. Rungės ir Kutos metodai

Taikant Rungės ir Kutos metodus, aukštesnės eilės y išvestinės įvertinamos apskaičiuojant $f(x, y)$ reikšmes keliuose tarpiniuose taškuose. Iš pradžių panagrinėkime atskirus šio metodo atvejus, paskui aptarkime bendrą Rungės ir Kutos metodų schemą.

Pirmieji Rungės ir Kutos metodai buvo paskelbti pačioje XX amžiaus pradžioje, o apibendrinti antrojoje jo pusėje. Tačiau istoriškai kai kurie diferencialinių lygčių sprendimo metodai, priklausantys Rungės ir Kutos metodų klasei, buvo išnagrinėti kur kas anksčiau ir vadinami kitais vardais. Pirmiausia tai Oilerio, pataisytasis Oilerio ir modifikuotasis Oilerio metodas. Toliau šie metodai bus įtraukti į bendrą Rungės ir Kutos metodų sistemą.

Pirmiausia aptarkime diferencialinės lygties sprendimo metodo charakteristikas: ***pateikimo formą, metodo eilę*** ir ***stabilumą***.

Metodo pateikimo forma gali būti ***išreikštinė*** arba ***neišreikštinė***.

Išreikštiniu (*explicit*) vadinamas metodas, kai funkcijos reikšmė y_{m+1} išreiškiama reikšmėmis y_m ir $f(x_m, y_m)$. Tai reiškia, kad kiekviename žingsnyje y_{m+1} reikšmė apskaičiuojama pagal išreikštinę formulę ir nereikia spręsti netiesinės lygties arba jų sistemos.

Neišreikštiniu (*implicit*) vadinamas metodas, kai funkcijos reikšmė y_{m+1} išreiškiama reikšmėmis y_m, $f(x_m, y_m)$ ir y_{m+1}. Tai reiškia, kad kiekviename žingsnyje reikės išspręsti netiesinę lygtį (jų sistemą), tam, kad apskaičiuotume y_{m+1} reikšmę.

Metodo tikslumo eilę nusako kiek Teiloro eilutės narių įvertina formulė, apskaičiuojanti y_{m+1} reikšmę.

Metodo stabilumas nusakomas tiriant metodo savybę kaupti paklaidas sprendimo eigoje. Jei pradinės informacijos paklaida sprendimo eigoje nesikaupia, tai metodas vadinamas ***absoliučiai stabiliu***. Jei paklaidos ir sprendinio reikšmių santykis sprendimo eigoje yra pastovus dydis, tai sakome, kad metodas yra ***santykinai stabilus***. Jei pradinės informacijos paklaida sprendimo eigoje neribotai auga, tai metodas yra ***nestabilus***.

Pirmosios eilės Rungės ir Kutos metodas. Šis metodas priklauso Oileriui ir dar vadinamas ***Oilerio metodu***. Aišku, kad jis turi įvertinti (9.4) formulės narius iki pirmosios eilės išvestinės imtinai ir yra apibūdinamas formulėmis

$$\begin{aligned} x_{m+1} &= x_m + h, \\ y_{m+1} &= y_m + y'_m h. \end{aligned} \qquad (9.5)$$

Metodo paklaida yra $O\left(h^2\right)$.

Oilerio metodas taikomas ribotai, nes, daug kartų remiantis (9.5) formule, labai padidėja globalioji metodo paklaida.

Aptarkime Oilerio metodo stabilumo klausimą. Pirmiausia stabilumo klausimą aptarkime, kai Oilerio metodas taikomas paprastai diferencialiniai lygčiai $y' = ky, y(x_0) = y_0$, o po to, kaip šis rezultatas gali būti išplėstas bendrai diferencialinei lygčiai.

Oilerio metodas lygčiai $y' = ky, y(x_0) = y_0$ aprašomas formule: $y_{m+1} = y_m + khy_m, m = 0,1,2,\cdots$. Laikydami, kad y_0 yra tiksli ir kiekviename žingsnyje skaičiavimai atliekami absoliučiai tiksliai, turėsime: $y_{m+1} = \left(1+kh\right)^{m+1} y_0, m = 0,1,2,\cdots$. Jei žingsnis h yra pakankamai mažas, tai ši formulė gerai aproksimuoja lygties tikslų sprendinį: $y = e^{kx}$.

Tarkime, kad pradinė reikšmė y_0 turi paklaidą: $y_0^a = y_0 + e_0$. Tada nagrinėjamai lygčiai Oilerio metodas bus aprašomas formule: $y_{m+1}^a = \left(1+kh\right)^{m+1} y_0 + \left(1+kh\right)^{m+1} e_0$.

Aišku, kad pradinė paklaida neribotai augs, jei $\left|1+kh\right| > 1$ ir mažės, jei $\left|1+kh\right| < 1$. Vadinasi, metodas bus absoliučiai stabilus, jei $-2 < kh < 0$.

Šis reikalavimas gali būti per daug griežtas. Dažnai pakanka, kad paklaida neaugtų greičiau nei sprendinys, t.y. metodas būtų santykinai stabilus. Aišku, kad Oilerio metodas bus santykinai stabilus, jei $k > 0$.

Oilerio metodo absoliutaus stabilumo sąlyga gali būti apibendrinta bendrai diferencialinei lygčiai (9.1). Galima įrodyti, kad absoliutaus stabilumo sąlyga nusakoma nelygybe: $-2 < h\dfrac{\partial f}{\partial y} < 0$. Kadangi $h > 0$, tai dalinė išvestinė $\dfrac{\partial f}{\partial y}$ turi buti neigiama.

Antrosios eilės Rungės ir Kutos metodai. Taikant šiuos metodus, reikia įvertinti (9.4) formulės narius iki antrosios eilės išvestinės imtinai. Plačiausiai praktikuojami antrosios eilės Rungės ir Kutos metodai yra ***pataisytasis Oilerio metodas*** ir ***modifikuotasis Oilerio metodas***. Apskritai antrosios eilės Rungės ir Kutos metodų, kaip ir aukštesnės eilės metodų, yra be galo daug.

Ieškosime antrosios eilės Rungės ir Kutos metodo formulės pavidalu

$$y_{m+1} = y_m + h\left(b_1 f\left(x_m, y_m\right) + b_2 f\left(x_m + c_2 h,\, y_m + a_{21} h y'_m\right)\right). \qquad (9.6)$$

Koeficientus b_1, b_2, c_2 ir a_{21} parinkime taip, kad (9.6) formulė įvertintų (9.4) formulės narius iki antrosios eilės išvestinės imtinai.

Funkciją $y' = f(x, y)$ išskleiskime Teiloro eilute taško $(x_m; y_m)$ aplinkoje:

$$f(x, y) = f(x_m, y_m) + \frac{\partial f}{\partial x}(x - x_m) + \frac{\partial f}{\partial y}(y - y_m) + \dots .$$

Tada

$$f(x_m + c_2 h,\ y_m + a_{21} h y'_m) = f + f'_x c_2 h + f'_y a_{21} h y'_m + \dots . \qquad (9.7)$$

(9.7) įrašę į (9.6), gauname:

$$\begin{aligned} y_{m+1} &= y_m + h b_1 y'_m + b_2 h\left(f + f'_x c_2 h + f'_y a_{21} h y'_m + O\left(h^2\right)\right) = \\ &= y_m + h(b_1 + b_2) y'_m + f'_x h^2 b_2 c_2 + f'_y y'_m h^2 a_{21} b_2 + O\left(h^3\right), \end{aligned} \qquad (9.8)$$

čia $f = f(x_m, y_m) = y'_m$.

Atsižvelgdami į tai, kad $y''_m = f'_x + f'_y y'_m$, perrašykime (9.4) formulę:

$$y_{m+1} = y_m + y'_m h + \frac{h^2}{2} f'_x + \frac{h^2}{2} f'_y y'_m + O\left(h^3\right). \qquad (9.9)$$

(9.8) ir (9.9) formulė sutaps, jei

$$\begin{cases} b_1 + b_2 = 1, \\ b_2 c_2 = 1/2, \\ b_2 a_{21} = 1/2. \end{cases} \qquad (9.10)$$

Kadangi keturi parametrai turi tenkinti tris sąlygas, tai vieną iš jų galime pasirinkti laisvai. Tarę, kad $b_2 = \omega \neq 0$, turėsime:

$$\begin{aligned} b_1 &= 1 - \omega, \\ a_{21} &= c_2 = \frac{1}{2\omega}. \end{aligned} \qquad (9.11)$$

(9.6) ir (9.11) formulė apibūdina antrosios eilės Rungės ir Kutos metodų šeimą. Jeigu keisime $b_2 = \omega$ ir pagal (9.11) formulę apskaičiuosime b_1, c_2 ir a_{21}, tai (9.6) formulė aprašys skirtingus antrosios eilės metodus.

Kai $\omega = 1/2$, turime ***pataisytąjį Oilerio*** metodą, kurį nusako formulė

$$y_{m+1} = y_m + \frac{1}{2} h\left(f(x_m, y_m) + f\left(x_m + h, y_m + h y'_m\right)\right). \qquad (9.12)$$

Kai $\omega = 1$, turime ***modifikuotąjį Oilerio*** metodą, apibūdinamą formule

$$y_{m+1} = y_m + h f\left(x_m + \frac{h}{2}, y_m + \frac{h}{2} y'_m\right). \qquad (9.13)$$

Kai $\omega = 3/4$, turime ***Heuno*** metodą, kurį nusako formulė

$$y_{m+1} = y_m + \frac{1}{4} h\left(f(x_m, y_m) + 3 f\left(x_m + \frac{2}{3} h, y_m + \frac{2}{3} h y'_m\right)\right). \qquad (9.14)$$

Antrosios eilės Rungės ir Kutos metodų paklaida yra $O\left(h^3\right)$, t. y. proporcinga h^3. Kaip nurodyta literatūroje, to metodo paklaida yra mažiausia, kai $\omega = 2/3$.

Bendra Rungės ir Kutos metodų schema. Aptarkime bendrą Rungės ir Kutos metodų schemą.

1 apibrėžimas. Tarkime, kad teigiamasis sveikasis skaičius s žymi stadijų arba etapų skaičių; $a_{21}, a_{31}, a_{32}, \ldots, a_{s1}, a_{s2}, \ldots, a_{s,\,s-1}$, $b_1, b_2, \ldots, b_s$, $c_2, \ldots, c_s$ — realieji skaičiai. Tada metodas, nusakomas formulėmis

$$\begin{cases} k_1 = f\left(x_m,\, y_m\right), \\ k_2 = f\left(x_m + c_2 h,\, y_m + h\, a_{21} k_1\right), \\ k_3 = f\left(x_m + c_3 h,\, y_m + h\left(a_{31} k_1 + a_{32} k_2\right)\right), \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ k_s = f\left(x_m + c_s h,\, y_m + h\left(a_{s1} k_1 + \ldots + a_{s,s-1} k_{s-1}\right)\right), \end{cases} \qquad (9.15)$$

$$y_{m+1} = y_m + h\left(b_1 k_1 + b_2 k_2 + \ldots + b_s k_s\right),$$

vadinamas s stadijų išreikštiniu Rungės ir Kutos metodu.

Koeficientai c_i paprastai tenkina sąlygą

$$c_2 = a_{21},\ c_3 = a_{31} + a_{32},\ \ldots,\ c_s = a_{s1} + \ldots + a_{s,\,s-1}. \qquad (9.16)$$

Šią sąlygą be jokių komentarų pirmasis pradėjo taikyti V. Kuta (W. Kutta). Ji labai palengvina aukštesnės eilės Rungės ir Kutos formulių koeficientų skaičiavimą, tačiau nėra būtina taikant žemesnės eilės metodus.

2 apibrėžimas. Formulė

$$y_{m+1} = y_m + h\left(b_1 k_1 + b_2 k_2 + \ldots + b_s k_s\right) \qquad (9.17)$$

apibūdina p-tosios eilės Rungės ir Kutos metodą, jeigu ji sutampa su (9.4) formule (su Teiloro eilute) iki p-tosios eilės išvestinės imtinai.

1964 m. Dž. K. Bučeris (J. C. Butcher) pasiūlė (9.15) formulę vaizduoti 9.1 lentele.

9.1 lentelė. ***s*** *stadijų išreikštinis Rungės ir Kutos metodas*

0					
c_2	a_{21}				
c_3	a_{31}	a_{32}			
$\vdots$					
c_s	a_{s1}	a_{s2}	$\ldots$	$a_{s,s-1}$	
	b_1	b_2	$\ldots$	b_{s-1}	b_s

Pavyzdžiui, pataisytąjį Oilerio metodą vaizduosime 9.2 lentele.

9.2 lentelė. Pataisytasis Oilerio metodas

0		
1	1	
	1/2	1/2

Kad (9.17) formulė apibūdintų p-tosios eilės Rungės ir Kutos metodą, (9.15) formulės koeficientus reikia apskaičiuoti taip, kad (9.17) formulė sutaptų su Teiloro eilute iki p-tosios eilės išvestinės imtinai. Tam tikslui suvienodinami (9.17) formulės ir Teiloro eilutės ((9.4) formulė) koeficientai prie tų pačių išvestinių. Gaunamos lygtys, kurias turi tenkinti (9.15) formulės koeficientai. Šis darbas trivialus, tačiau, didėjant p eilei, darosi

labai sudėtingas. 9.3 lentelėje pateikta sąlygų skaičiaus priklausomybė nuo metodo eilės p.

9.3 lentelė. p-tosios eilės Rungės ir Kutos metodo sąlygų skaičius

Eilė p	1	2	3	4	5	6	7	8	9	10
Sąlygų skaičius	1	2	4	8	17	37	85	200	486	1205

Pavyzdžiui, kaip nurodyta literatūroje, keturių stadijų ir 4-osios eilės Rungės ir Kutos metodo koeficientai, atsižvelgiant į (9.16) sąlygą, turi tenkinti lygčių sistemą

$$\begin{cases} b_1 + b_2 + b_3 + b_4 = 1, \\ b_2 c_2 + b_3 c_3 + b_4 c_4 = 1/2, \\ b_2 c_2^2 + b_3 c_3^2 + b_4 c_4^2 = 1/3, \\ b_2 c_2^3 + b_3 c_3^3 + b_4 c_4^3 = 1/4, \\ b_3 c_3 a_{32} c_2 + b_4 c_4 \left(a_{42} c_2 + a_{43} c_3 \right) = 1/8, \\ b_3 a_{32} + b_4 a_{42} = b_2 \left(1 - c_2 \right), \\ b_4 a_{43} = b_3 \left(1 - c_3 \right), \\ 0 = b_4 \left(1 - c_4 \right). \end{cases} \qquad (9.18)$$

Ši sistema turi be galo daug sprendinių. Populiariausi šios sistemos sprendiniai pateikti 9.4 ir 9.5 lentelėje.

9.4 lentelė. Klasikinis ketvirtosios eilės Rungės ir Kutos metodas, pasiūlytas Kutos 1901 m.

0				
1/2	1/2			
1/2	0	1/2		
1	0	0	1	
	1/6	2/6	2/6	1/6

9.5 lentelė. 4-osios eilės Gilo(Gill) metodas

0				
1/2	1/2			
1/2	$\left(\sqrt{2}-1\right)/2$	$\left(2-\sqrt{2}\right)/2$		
1	0	$-\sqrt{2}/2$	$\left(1+\sqrt{2}/2\right)$	
	1/6	$\left(2-\sqrt{2}\right)/6$	$\left(2+\sqrt{2}\right)/6$	1/6

Kalbant apie ketvirtos eilės Rungės ir Kutos metodo stabilumą, yra žinoma, kad metodas yra absoliučiai stabilus, jei tenkinama sąlygą: $-2.78 < h\partial f / \partial y < 0$. Praktiškai $\partial f / \partial y$ reikšmę galima aproksimuoti, panaudojant sprendimo eigoje apskaičiuotas dvi gretimas f ir y reikšmes.

Konstruojant Rungės ir Kutos metodus, reikia žinoti, kad metodo eilė ir stadijų skaičius sutampa tik metodams iki ketvirtos eilės imtinai. Todėl kyla klausimas: „Koks

turi būti mažiausias stadijų skaičius *s,* kad egzistuotų *p*-tosios eilės Rungės ir Kutos metodas?“ Šis stadijų skaičius vadinamas Bučerio barjeru.

Pirmiausia buvo įrodyta teorema, kad, kai $p \geq 5$, nėra išreikštinių Rungės ir Kutos metodų, kurių $s = p$. Paskui Bučeris įrodė teoremą ir sukūrė algoritmą, leidžiantį konstruoti šešių stadijų 5-osios eilės Rungės ir Kutos metodus.

1964 m. Dž. K. Bučeris įrodė, kad neegzistuoja $s = p+1$ stadijų Rungės ir Kutos metodai, kai $p \geq 7$, o 1984 metais — kad neegzistuoja $s = p+2$ stadijų Rungės ir Kutos metodai, kai $p \geq 8$.

Iki šiol aukščiausios eilės sukonstruotas Rungės ir Kutos metodas yra 10-osios eilės. Pirmasis 1975 m. jį sukonstravo A. R. Kurtis (A. R. Curtis). Šis metodas turi 18 stadijų ir yra įtrauktas į Gineso rekordų knygą. Vėliau, 1978 metais, E. Haireris (E. Hairer) sukonstravo 17 stadijų 10-osios eilės Rungės ir Kutos metodą, kuris pateiktas literatūroje „*E.Hairer, S.P.Norsett, G.Wanner, Solving Ordinary Differential Equations I, Nonstiff Problems, Springer-Verlag, Berlin, Heidelberg, 1987*“. Šis metodas gali būti naudingas, kai reikia apskaičiuoti sprendinį 10^{-15} tikslumu.

1 pavyzdys. *Skaitiniu būdu išspręskime diferencialinę lygtį* $y' = x + y$ *su duotomis pradinėmis sąlygomis:* $y(0) = 1$, *t.y. išspręskime Koši uždavinį, kai* $0 \leq x \leq 1$ *ir integravimo žingsnis h=0.1.*

Pradžioje Oilerio metodu, pataisytuoju Oilerio metodu, modifikuotuoju Oilerio metodu ir ketvirtosios eilės Rungės ir Kutos metodu apskaičiuosime sprendinio tašką (x_1, y_1), o po to, grafiškai pavaizduosime tikslų sprendinį ir skaitiniais metodais gautus sprendinius.

Sprendimas. Kadangi $y' = x + y$ yra tiesinė diferencialinė lygtis, tai jos sprendinį galima apskaičiuoti tiksliai. Jos sprendinys, tenkinantis duotas pradines sąlygas, yra: $y = 2e^x - x - 1$. Tada tiksli sprendinio reikšmė kai $x = 0.1$ yra: $y_1 = 2e^{0.1} - 0.1 - 1 = 1.1103418$.

Reikšmės $y_1 = y(x_1)$ ***apskaičiavimas Oilerio metodu***

Darbo formulės:

$$x_{m+1} = x_m + h,$$
$$k1 = y'(x_m, y_m),$$
$$y_{m+1} = y_m + k1h.$$

Pradinės sąlygos: $x_0 = 0; y_0 = 1$.

Taško (x_1, y_1) *apskaičiavimas:*

$k1 = x_0 + y_0 = 0+1 = 1;$

$x_1 = x_0 + h = 0 + 0.1 = 0.1;$

$y_1 = y_0 + k1 * h = 1 + 1 * 0.1 = 1.1.$

Reikšmės $y_1 = y(x_1)$ ***apskaičiavimas pataisytuoju Oilerio metodu***

Darbo formulės:

$$x_{m+1} = x_m + h,$$
$$k1 = f(x_m, y_m),$$
$$k2 = f(x_m + h, y_m + k1 * h),$$
$$y_{m+1} = y_m + h(k1 + k2)/2.$$

Pradinės sąlygos: $x_0 = 0; y_0 = 1$.

Taško (x_1, y_1) *apskaičiavimas:*

$k1 = x_0 + y_0 = 0+1 = 1;$

$k2 = x_0 + h + y_0 + k1 * h = 0+0.1+1+1*0.1 = 1.2;$

$x_1 = x_0 + h = 0 + 0.1 = 0.1;$

$y_1 = y_0 + h * (k1 + k2)/2 = 1 + 0.1 * (1 + 1.2)/2 = 1.11.$

Reikšmės $y_1 = y(x_1)$ ***apskaičiavimas modifikuotuoju Oilerio metodu***

Darbo formulės:

$$x_{m+1} = x_m + h,$$
$$k1 = f(x_m, y_m),$$
$$k2 = f(x_m + h/2, y_m + k1 * h/2),$$
$$y_{m+1} = y_m + h * k2.$$

Pradinės sąlygos: $x_0 = 0; y_0 = 1$.

Taško (x_1, y_1) *apskaičiavimas:*

$k1 = x_0 + y_0 = 0+1 = 1;$

$k2 = x_0 + h/2 + y_0 + k1 * h/2 = 0+0.1/2+1+1*0.1/2 = 1.1;$

$x_1 = x_0 + h = 0 + 0.1 = 0.1;$

$y_1 = y_0 + h * k2/2 = 1 + 0.1 * 1.1 = 1.11.$

Reikšmės $y_1 = y(x_1)$ ***apskaičiavimas 4-osios eilės Rungės ir Kutos metodu***

Darbo formulės:

$$x_{m+1} = x_m + h,$$
$$k1 = f(x_m, y_m),$$
$$k2 = f(x_m + h/2, y_m + k1 * h/2),$$
$$k3 = f(x_m + h/2, y_m + k2 * h/2),$$
$$k4 = f(x_m + h, y_m + k3 * h),$$
$$y_{m+1} = y_m + h * (k1 + 2 * k2 + 2 * k3 + k4)/6.$$

Pradinės sąlygos: $x_0 = 0; y_0 = 1$.

Taško (x_1, y_1) *apskaičiavimas:*

$k1 = x_0 + y_0 = 0+1 = 1;$

$k2 = x_0 + h/2 + y_0 + k1 * h/2 = 0+0.1/2+1+1*0.1/2 = 1.1;$

$k3 = x_0 + h/2 + y_0 + k2 * h/2 = 0 + 0.1/2 + 1 + 1.1 * 0.1/2 = 1.1050;$

$k4 = x_0 + h + y_0 + k3 * h = 0+0.1+1+1.1050*0.1=1.2105;$

$x_1 = x_0 + h = 0 + 0.1 = 0.1;$

$y_1 = y_0 + h * (k1 + 2 * k2 + 2 * k3 + k4) / 6 = 1+0.1*(1+2*1.1+2*1.1050+1.2105)/6=$
$=1.1103417.$

Kaip matome, kuo metodo eilė yra aukštesnė, tuo apskaičiuota y_1 reikšmė yra tikslesnė. Šį faktą vaizdžiai galima pamatyti 9.1, 9.2 ir 9.3 paveikslėliuose, kuriuose pavaizduoti sprendžiamos diferencialinės lygties $y' = x + y$ tikslus sprendinys ir skaitiniu būdu gauti sprendiniai, taikant Oilerio metodą (9.1 pav.), pataisytą Oilerio metodą (9.2 pav.) ir ketvirtos eilės Rungės ir Kutos metodą (9.3 pav.), kai integravimo žingsnis $h = 0.25$.

2 pavyzdys. *Išspręskime diferencialinių lygčių sistemą*

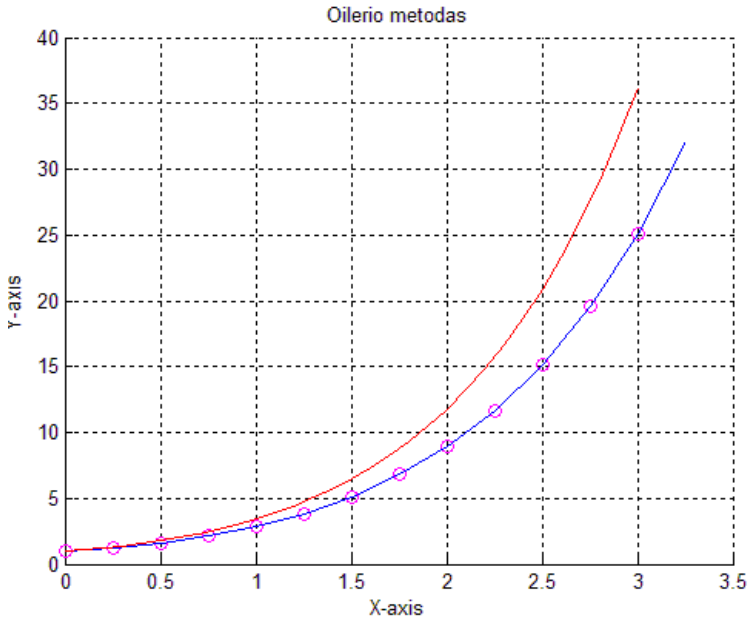
$$\begin{cases} y_1' = \cos x - e^x - 3y_2, \\ y_2' = 2e^x - \cos x + 4y_2, \end{cases} \quad \textit{su pradinėmis sąlygomis:} \quad \begin{cases} y_1(0) = 1, \\ y_2(0) = -\dfrac{2}{3}. \end{cases}$$

Sistemą spręsime Oilerio metodu ir pataisytuoju Oilerio metodu. Pradžioje abiem metodais apskaičiuosime tašką $(x_1, y_1(x_1), y_2(x_1))$, kai integravimo žingsnis $h = 0.1$, o po to grafiškai pavaizduosime tikslų sprendinį ir skaitiniais metodais gautus sprendinius.

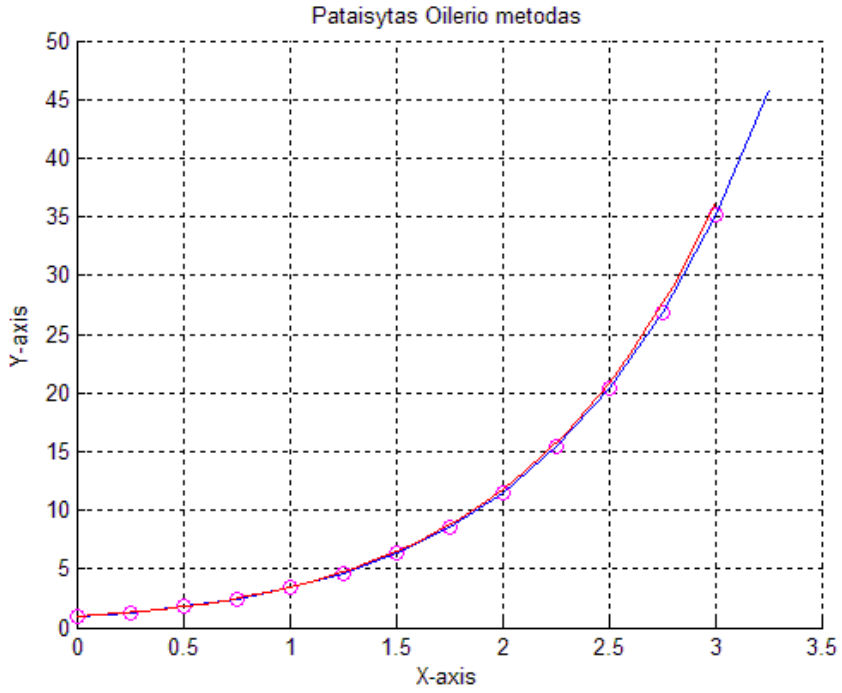
Sprendimas

Yra žinomas tikslus šios diferencialinių lygčių sistemos sprendinys:

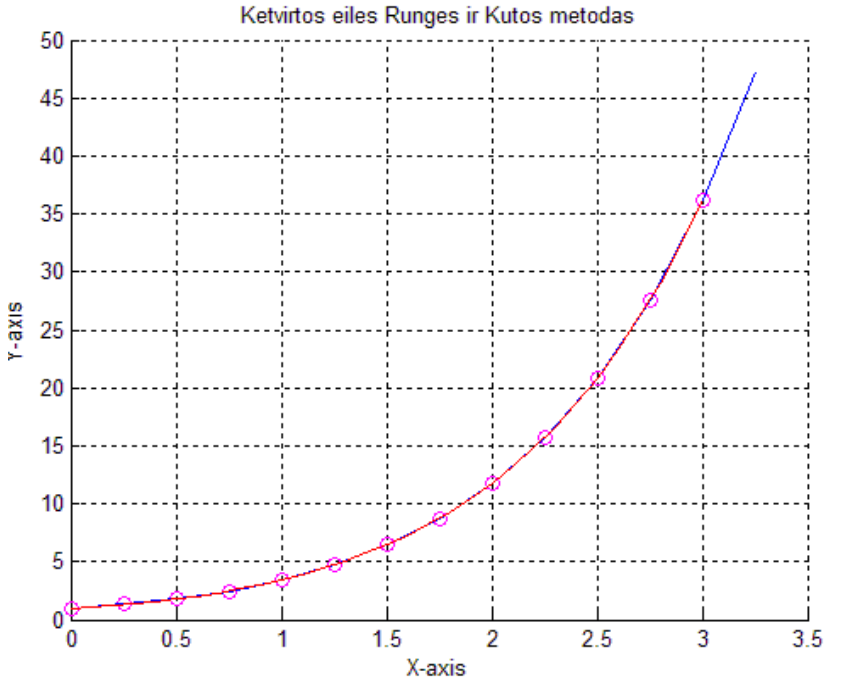
$$\begin{cases} y_1 = e^x + (5\sin x - 3\cos x + 3e^{4x})/17, \\ y_2 = -2e^x/3 - (\sin x - 4\cos x + 4e^{4x})/17. \end{cases}$$



9.1 pav. Diferencialinės lygties $y' = x + y$ tikslus sprendinys (raudonas grafikas) ir sprendinys, apskaičiuotas Oilerio metodu (mėlynas grafikas).



9.2 pav. Diferencialinės lygties $y' = x + y$ tikslus sprendinys (raudonas grafikas) ir sprendinys, apskaičiuotas pataisytu Oilerio metodu (mėlynas grafikas).



9.3 pav. Diferencialinės lygties $y' = x + y$ tikslus sprendinys (raudonas grafikas) ir sprendinys, apskaičiuotas pataisytu Oilerio metodu (mėlynas grafikas).

Reikšmių $\begin{cases} y_1(x_1) \\ y_2(x_2) \end{cases}$ ***apskaičiavimas Oilerio metodu***

Darbo formulės:

$$x_1 = x_0 + h,$$
$$k1_1 = y_1'(x_0, y_1(x_0), y_2(x_0)),$$
$$k1_2 = y_2'(x_0, y_1(x_0), y_2(x_0)),$$
$$y_1(x_1) = y_1(x_0) + k1_1 h,$$
$$y_2(x_1) = y_2(x_0) + k1_2 h.$$

Pradinės sąlygos: $x_0 = 0; y_1 = 1, y_2 = -\frac{2}{3}$.

Taško (x_1, y_1, y_2) *apskaičiavimas:*

$$k1_1 = y_1'(x_0, y_1(x_0), y_2(x_0)) = \cos(x_0) - e^{x_0} - 3y_2(x_0) = \cos(0) - e^0 + 3 * \frac{2}{3} = 2,$$

$$k1_2 = y_2'(x_0, y_1(x_0), y_2(x_0)) = 2e^{x_0} - \cos(x_0) + 4y_2(x_0) = 2e^0 - \cos(0) - 8/3 = -5/3.$$

$$x_1 = x_0 + h = 0 + 0.1 = 0.1;$$

$$y_1(x_1) = y_1(x_0) + k1_1 * h = 1 + 2 * 0.1 = 1.2,$$

$$y_2(x_1) = y_2(x_0) + k1_2 * h = -2/3 - \frac{5}{3} * 0.1 = -0.8333.$$

Reikšmių $\begin{cases} y_1(x_1) \\ y_2(x_2) \end{cases}$ ***apskaičiavimas pataisytuoju Oilerio metodu***

Darbo formulės:

$$x_1 = x_0 + h,$$
$$k1_1 = y_1'(x_0, y_1(x_0), y_2(x_0)),$$
$$k1_2 = y_2'(x_0, y_1(x_0), y_2(x_0)),$$
$$k2_1 = y_1'(x_0 + h, y_1(x_0) + k1_1 h, y_2(x_0) + k1_2 h),$$
$$k2_2 = y_2'(x_0 + h, y_1(x_0) + k1_1 h, y_2(x_0) + k1_2 h),$$
$$y_1(x_1) = y_1(x_0) + h(k1_1 + k2_1)/2,$$
$$y_2(x_1) = y_2(x_0) + h(k1_2 + k2_2)/2.$$

Pradinės sąlygos: $x_0 = 0; y_1 = 1, y_2 = -\frac{2}{3}$.

Taško (x_1, y_1, y_2) *apskaičiavimas:*

$$k1_1 = y_1'(x_0, y_1(x_0), y_2(x_0)) = \cos(x_0) - e^{x_0} - 3y_2(x_0) = \cos(0) - e^0 + \frac{2}{3} = 2,$$

$$k1_2 = y_2'(x_0, y_1(x_0), y_2(x_0)) = 2e^{x_0} - \cos(x_0) + 4y_2(x_0) = 2e^0 - \cos(0) - 8/3 = -5/3,$$

$$k2_1 = y_1'(x_0 + h, y_1(x_0) + k1_1 h, y_2(x_0) + k1_2 h) =$$

$$= \cos(x_0 + h) - e^{x_0 + h} - 3(y_2(x_0) + k1_2 h) = \cos(0.1) - e^{0.1} - 3 * (-\frac{2}{3} - \frac{5}{3} * 0.1) =$$

$= 2.38983,$

$k2_2 = y_2'(x_0 + h, y_1(x_0) + k1_1 h, y_2(x_0) + k1_2 h) =$

$= 2e^{x_0+h} - \cos(x_0 + h) + 4(y_2(x_0) + k1_2 h) = 2e^{0.1} - \cos(0.1) + 4 * (-\frac{2}{3} - \frac{5}{3} * 0.1) =$

$= -2.11800,$

$x_1 = x_0 + h = 0 + 0.1 = 0.1;$

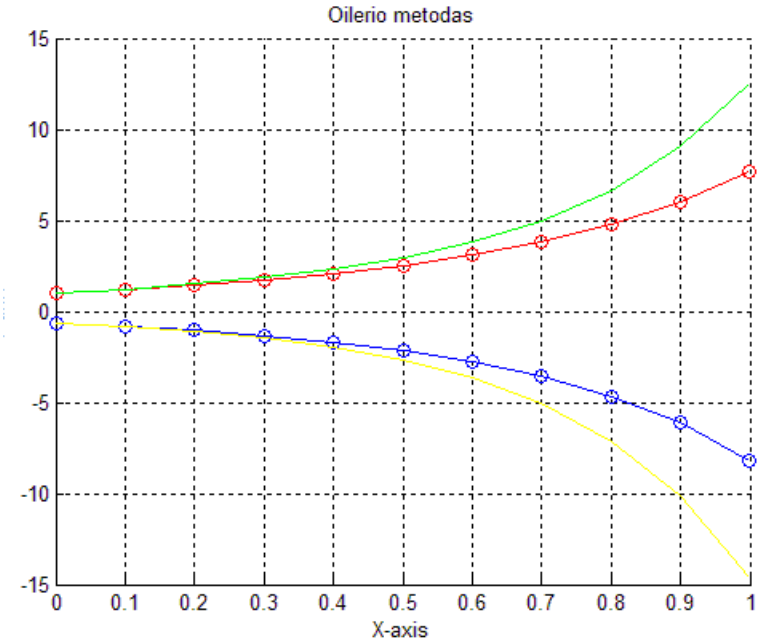
$y_1(x_1) = y_1(x_0) + h(k1_1 + k2_1)/2 = 1 + 0.1 * (2 + 2.38983)/2 = 1.2194915,$

$y_2(x_1) = y_2(x_0) + h(k1_2 + k2_2)/2 = -2/3 + 0.1 * (-5/3 - 2.118)/2 = -0.8559.$

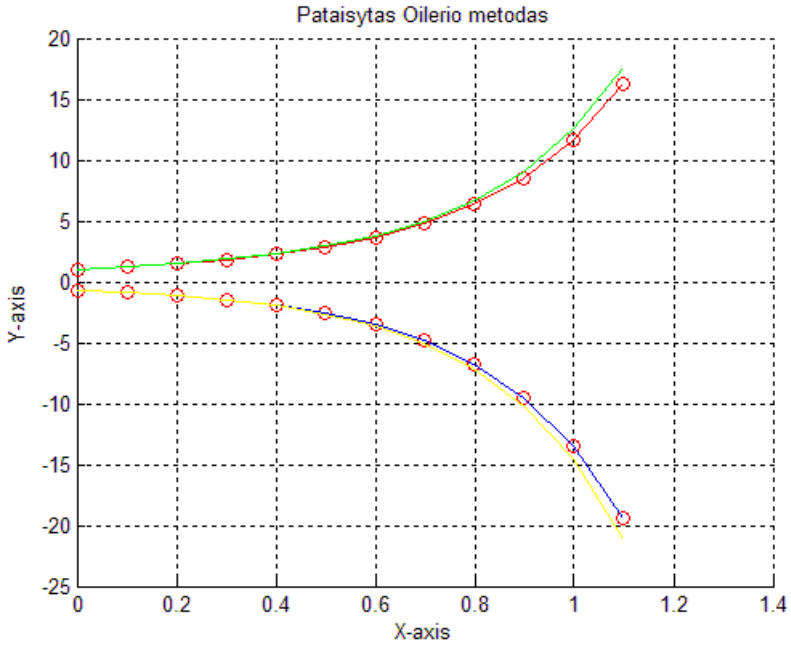
Tiksli sprendinio reikšmė: $\begin{matrix} y_1 = 1.22220789927117 \\ y_2 = -0.85955211476219 \end{matrix}$.

Kaip ir reikėjo tikėtis, pataisytas Oilerio metodas įgalino tiksliau apskaičiuoti $\begin{cases} y_1(x_1) \\ y_2(x_2) \end{cases}$ reikšmes, nei Oilerio metodas.

Tikslus sprendžiamos diferencialinių lygčių sistemos sprendinys ir sprendiniai, apskaičiuoti Oilerio metodu ir pataisytu Oilerio metodu, kai $h = 0.1$, pavaizduoti 9.4 ir 9.5 paveikslėliuose.



9.4 pav. 2 pavyzdžio diferencialinių lygčių sistemos tikslus sprendinys (y_1 - žalia kreivė; y_2 - geltona kreivė) ir Oilerio metodu gautas sprendinys (y_1 - raudona kreivė; y_2 - mėlyna kreivė).



9.5 pav. 2 pavyzdžio diferencialinių lygčių sistemos tikslus sprendinys (y_1 - žalia kreivė; y_2 - geltona kreivė) ir pataisytu Oilerio metodu gautas sprendinys (y_1 - raudona kreivė; y_2 - mėlyna kreivė).

9.1.3. Integravimo žingsnio parinkimas ir paklaidos įvertinimas.

Integravimo žingsnio parinkimas. Sprendžiant diferencialines lygtis, tenka pasirinkti tokį integravimo žingsnį h, kad metodo lokalioji paklaida būtų mažesnė už reikiamą tikslumą ε.

Tarkime, kad taikome p-tosios eilės Rungės ir Kutos metodą, taigi lokalioji metodo paklaida proporcinga h^{p+1}. Tada galima rašyti, kad metodo paklaida

$$err = Ch^{p+1}. \qquad (9.19)$$

Iš (9.19) formulės apskaičiuojame:

$$C = \frac{err}{h^{p+1}}. \qquad (9.20)$$

Tarkime, kad h_{opt} yra optimalus žingsnis, garantuojantis norimo dydžio lokaliąją paklaidą ε, t. y. laikydami, kad sprendinio $(p+1)$-osios eilės išvestinė integravimo intervale yra pastovi, galime užrašyti:

$$Ch_{opt}^{p+1} = \varepsilon. \qquad (9.21)$$

Tada iš (9.20) ir (9.21) formulės išplaukia, jog

$$h_{opt} = \left(\frac{\varepsilon}{err}\right)^{\frac{1}{p+1}} h. \qquad (9.22)$$

Vadinasi, iš (9.22) formulės galima apskaičiuoti optimalų integravimo žingsnį, jei žinoma metodo paklaida *err*, apskaičiuota esant integravimo žingsniui h.

Metodo paklaidos įvertinimas. Sprendinio lokaliąją paklaidą galime apskaičiuoti dviem metodais: 1) Ričardsono metodu; 2) įdėtųjų formulių metodu.

Ričardsono metodas. Tarkime, kad diferencialinę lygtį $y' = f(x, y)$, tenkinančią pradinę sąlygą $y(x_0) = y_0$, sprendžiame p-tosios eilės Rungės ir Kutos metodu. Pagal tašką $(x_m; y_m)$, naudodami integravimo žingsnį h, apskaičiuojame y_{m+1}^h. Taip pat, naudodami integravimo žingsnį $h/2$ ir du kartus pritaikę metodą, apskaičiuojame $y_{m+1}^{h/2}$. Tada, laikydami, kad sprendinio $(p+1)$-osios eilės išvestinė yra pastovi, galime parašyti:

$$\begin{aligned} y_{m+1} &= y_{m+1}^h + Ch^{p+1}, \\ y_{m+1} &= y_{m+1}^{h/2} + 2C(h/2)^{p+1}; \end{aligned} \qquad (9.23)$$

čia y_{m+1} — tiksli sprendinio reikšmė.

Iš (9.23) sistemos išplaukia, kad metodo paklaida

$$err = 2C(h/2)^{p+1} = \frac{y_{m+1}^{h/2} - y_{m+1}^h}{2^p - 1}. \qquad (9.24)$$

Aišku, kad, remdamiesi (9.24) formule, gausime tikslią metodo paklaidą, jei $y^{(p+1)}(x)$ bus pastovi; priešingu atveju $\left| \frac{y_{m+1}^{h/2} - y_{m+1}^h}{2^p - 1} \right|$ vertins metodo paklaidą.

Reikia pabrėžti, kad nors

$$y_{m+1} = y_{m+1}^{h/2} + \frac{y_{m+1}^{h/2} - y_{m+1}^h}{2^p - 1} \qquad (9.25)$$

yra $(p+1)$-osios eilės formulė, tačiau praktiškai ja naudotis nerekomenduojama. Geriau vietoj jos taikyti $(p+1)$-osios eilės Rungės ir Kutos metodą.

Norint pagal (9.24) formulę apskaičiuoti metodo paklaidą, kiekviename sprendinio taške reikia 3 kartus taikyti p-tosios eilės metodą. Todėl pastaruoju metu plačiai naudojamas kitas, racionalesnis, metodo paklaidos įvertinimo būdas — ***įdėtosios formulės*** (embedded formulas).

Įdėtųjų formulių metodas. Įdėtųjų formulių idėja labai paprasta. Paimkime daugiau stadijų negu jų reikia konstruojant norimos eilės Rungės ir Kutos metodą. Tada, pasinaudodami tomis pačiomis stadijomis, sukonstruokime dvi p-tosios ir q-tosios eilės formules. Šių formulių rezultatų skirtumo modulis ir įvertins metodo paklaidą. Kitaip tariant, reikia sudaryti tokią koeficientų lentelę (žr. 9.6 lentelę), kad formulė

$$y_{m+1} = y_m + h(b_1k_1 + b_2k_2 + \ldots + b_sk_s)$$

būtų p-tosios, o formulė

$$\hat{y}_{m+1} = y_m + h(\hat{b}_1k_1 + \hat{b}_2k_2 + \ldots + \hat{b}_sk_s)$$

— q-tosios eilės.

Įdėtosios formulės paprastai žymimos $p(q)$; čia p nurodo eilę metodo, pagal kurį apskaičiuojamas sprendinys, o q — eilę metodo, kuris padeda įvertinti paklaidą err ir kartu valdyti integravimo žingsnį. Aišku, kad tikslesnės formulės yra tos, kurių $p > q$. Plačiausiai šiuo metu naudojamos Rungės, Kutos ir Felbergo šešių stadijų 5-osios (4-osios) eilės, Dormano ir Prinso septynių stadijų 5-osios (4-osios) eilės bei Dormano ir Prinso trylikos stadijų 8-osios (7-osios) eilės įdėtosios formulės. Jos pateiktos atitinkamai

9.7, 9.8 ir 9.9 lentelėje. 9.9 lentelės parametrų skaitinės reikšmės nurodytos programoje ***dopri8***.

9.6 lentelė. p(q) eilės Rungės ir Kutos metodas

0					
c_2	a_{21}				
c_3	a_{31}	a_{32}			
$\vdots$	$\vdots$				
c_s	a_{s1}	a_{s2}	$\cdots$	$a_{s,s-1}$	
	b_1	b_2	$\cdots$	b_{s-1}	b_s
	$\hat{b}_1$	$\hat{b}_2$	$\cdots$	$\hat{b}_{s-1}$	$\hat{b}_s$

Dažniausiai $|p-q|=1$. Tada

$err \approx |y_{m+1} - \hat{y}_{m+1}|$.

Toliau pateikiama funkcija ***dopri8***, realizuojanti 8-osios (7-osios) eilės Dormano ir Prinso metodą diferencialinių lygčių sistemai. Funkcija ***dopri8*** skirtingai nuo MATLAB'o funkcijos ***ode45***, kuri realizuoja 5(4) Dormano ir Prinso metodą bei funkcijos ***ode23***, kuri realizuoja 3(2) Bogackio ir Šampani (Bogacki, Shampine) yra žymiai tikslesnė ir tinka spręsti nestandžias diferencialines lygtis 10^{-12} tikslumu. Reikia pažymėti, kad funkcija ***ode45*** geriausiai tinka pirmam bandymui, o ***ode23***, kuri taip pat priklauso Rungės ir Kutos metodų klasei, yra pranašesnė nei ***ode45***, jei reikalaujamas sprendinio tikslumas nėra didelis, o diferencialinė lygtis yra vidutiniškai standi.

9.7 lentelė. Rungės, Kutos ir Felbergo 5-osios (4-osios) eilės metodas

0						
$\frac{1}{4}$	$\frac{1}{4}$					
$\frac{3}{8}$	$\frac{3}{32}$	$\frac{9}{32}$				
$\frac{12}{13}$	$\frac{1932}{2197}$	$-\frac{7200}{2197}$	$\frac{7296}{2197}$			
1	$\frac{439}{216}$	-8	$\frac{3680}{513}$	$-\frac{845}{4104}$		
$\frac{1}{2}$	$-\frac{8}{27}$	2	$-\frac{3544}{2565}$	$\frac{1859}{4104}$	$-\frac{11}{40}$	
y_{m+1}	$\frac{16}{135}$	0	$\frac{6656}{12825}$	$\frac{28561}{56430}$	$-\frac{9}{50}$	$\frac{2}{55}$
$\hat{y}_{m+1}$	$\frac{25}{216}$	0	$\frac{1408}{2565}$	$\frac{2197}{4104}$	$-\frac{1}{5}$	0

9.8 lentelė. Dormano ir Prinso 5-osios (4-osios) eilės metodas

0							
$\frac{1}{5}$	$\frac{1}{5}$						
$\frac{3}{10}$	$\frac{3}{40}$	$\frac{9}{40}$					
$\frac{4}{5}$	$\frac{44}{45}$	$-\frac{56}{15}$	$\frac{32}{9}$				
$\frac{8}{9}$	$\frac{19372}{6561}$	$-\frac{25360}{2187}$	$\frac{64448}{6561}$	$-\frac{212}{729}$			
1	$\frac{9017}{3168}$	$-\frac{355}{33}$	$\frac{46732}{5247}$	$\frac{49}{176}$	$-\frac{5103}{18656}$		
1	$\frac{35}{384}$	0	$\frac{500}{1113}$	$\frac{125}{192}$	$-\frac{2187}{6784}$	$\frac{11}{84}$	
y_{m+1}	$\frac{35}{384}$	0	$\frac{500}{1113}$	$\frac{125}{192}$	$-\frac{2187}{6784}$	$\frac{11}{84}$	0
$\hat{y}_{m+1}$	$\frac{5179}{57600}$	0	$\frac{7571}{16695}$	$\frac{393}{640}$	$-\frac{92097}{339200}$	$\frac{187}{2100}$	$\frac{1}{40}$

9.9 lentelė. Dormano ir Prinso 8-osios (7-osios) eilės metodas

0												
c_2	a_{21}											
c_3	a_{31}	a_{32}										
c_4	a_{41}	0	a_{43}									
c_5	a_{51}	0	a_{53}	a_{54}								
c_6	a_{61}	0	a_{63}	a_{64}	a_{65}							
c_7	a_{71}	0	0	a_{74}	a_{75}	a_{76}						
c_8	a_{81}	0	0	a_{84}	a_{85}	a_{86}	a_{87}					
c_9	a_{91}	0	0	a_{94}	a_{95}	a_{96}	a_{97}	a_{98}				
c_{10}	$a_{10,1}$	0	0	$a_{10,4}$	$a_{10,5}$	$a_{10,6}$	$a_{10,7}$	$a_{10,8}$	$a_{10,9}$			
c_{11}	$a_{11,1}$	0	0	$a_{11,4}$	$a_{11,5}$	$a_{11,6}$	$a_{11,7}$	$a_{11,8}$	$a_{11,9}$	$a_{11,10}$		
c_{12}	$a_{12,1}$	0	0	$a_{12,4}$	$a_{12,5}$	$a_{12,6}$	$a_{12,7}$	$a_{12,8}$	$a_{12,9}$	$a_{12,10}$	$a_{12,11}$	
c_{13}	$a_{13,1}$	0	0	$a_{13,4}$	$a_{13,5}$	$a_{13,6}$	$a_{13,7}$	$a_{13,8}$	$a_{13,9}$	$a_{13,10}$	$a_{13,11}$	0
y_{m+1}	b_1	0	0	0	0	b_6	b_7	b_8	b_9	b_{10}	b_{11}	b_{12}	b_{13}
$\hat{y}_{m+1}$	bq_1	0	0	0	0	bq_6	bq_7	bq_8	bq_9	bq_{10}	bq_{11}	bq_{12}	bq_{13}

```
function [xspr,yspr]=dopri8(fun,x,y,xend,ep,hmax,h,stat)
% DOPRI8 sprendžia n (n>=1) diferencialinių lygčių sistemą
% Normano ir Prinso 8(7) eilės metodu.
% Lit. E.Hairer, S.P.Norsett, G.Wanner, Solving Ordinary Differential
% equations I, Nonstiff Problems, Springer-Verlag, Berlin, Heidelberg, 1987.
% Įėjimo parametrai:
% fun  - funkcija, apskaičiuojanti sistemos reikšmes (vektorių-
%        stulpelį) ,
% x    - pradine x-o reikšmė,
% y    - pradiniu sąlygų vektorius,
% xend - paskutine x-o reikšmė,
% ep   - lokaliosios paklaidos dydis,
% hmax - maksimalus leistinas integravimo žingsnis,
% h    - pradinis (spėjamas) integravimo žingsnis,
% stat - statistikos požymis.
%       Jei stat=1,tai atspausdinama informacija kiek buvo skaičiuota
%       funkcijos reikšmių, o taip pat bendras, priimtų ir atmestų
%       žingsnių skaičius.
% Išėjimo parametrai:
% xspr    - sprendinio x-o reikšmių vektorius,
% yspr    - sprendinio y-o reikšmių formato (n,numel(x)) matrica,

n=numel(y);
a=zeros(13); aa=zeros(n,12); bt=zeros(n,13);
% Vektorius c
c1=[0 1/18 1/12 1/8 5/16 3/8 59/400 93/200 5490023248/9719169821
    13/20];
c2=[1201146811/1299019798 1 1];
c=[c1 c2];
% Vektorius b
t1=[14005451/335480064 0 0 0 0 -59238493/1068277825
    181606767/758867731];
t2=[561292985/797845732 -1041891430/1371343529 760417239/1151165299];
t3=[118820643/751138087 -528747749/2220607170 1/4];
b=[t1 t2 t3];
% Vektorius bq
t1=[13451932/455176623 0 0 0 0 -808719846/976000145
    1757004468/5645159321];
t2=[656045339/265891186 -3867574721/1518517206 465885868/322736535];
t3=[53011238/667516719 2/45 0];
bq=[t1 t2 t3];
% Matrica a
a(2,1)=1/18;
a(3,1:2)=[1/48 1/16];
a(4,1:3)=[1/32 0 3/32];
a(5,1:4)=[5/16 0 -75/64 75/64 ];
a(6,1:5)=[3/80 0 0 3/16 3/20];
a(7,1:6)=[29443841/614563906 0 0 77736538/692538347 -28693883/1125e6
          23124283/18e8];
t1=[16016141/946692911 0 0 61564180/158732637 22789713/633445777];
t2=[545815736/2771057229 -180193667/1043307555];
a(8,1:7)=[t1 t2];
t1=[39632708/573591083 0 0 -433636366/683701615
   -421739975/2616292301];
t2=[100302831/723423059 790204164/839813087 800635310/3783071287];
a(9,1:8)=[t1 t2];
```

```
t1=[246121993/1340847787 0 0 -37695042795/15268766246
   -309121744/1061227803];
t2=[-12992083/490766935 6005943493/2108947869 393006217/1396673457];
t3=123872331/1001029789;
a(10,1:9)=[t1 t2 t3];
t1=[-1028468189/846180014 0 0 8478235783/508512852
     1311729495/1432422823];
t2=[-10304129995/1701304382 -48777925059/3047939560
     15336726248/1032824649];
t3=[-45442868181/3398467696 3065993473/597172653];
a(11,1:10)=[t1 t2 t3];
t1=[185892177/718116043 0 0 -3185094517/667107341
   -477755414/1098053517];
t2=[-703635378/230739211 5731566787/1027545527 5232866602/850066563];
t3=[-4093664535/808688257 3962137247/1805957418 65686358/487910083];
a(12,1:11)=[t1 t2 t3];
t1=[403863854/491063109 0 0 -5068492393/434740067
   -411421997/543043805];
t2=[652783627/914296604 11173962825/925320556
   -13158990841/6184727034];
t3=[3936647629/1978049680 -160528059/685178525 248638103/1413531060
    0];
a(13,1:12)=[t1 t2 t3];
nmax=2000;
% Pradiniai pasiruošimai
[m1,n1]=size(y);
if m1 == 1
    y=y';
end;
if x == xend
     return
end
if x < xend
    posneg=1;
else
    posneg=-1;
end
hmax=abs(hmax);
if abs(h) < 1e-10
    h=1e-10;
end
if h > hmax
    h=hmax;
end
h=posneg*h;
if ep < 13*eps
    ep=13*eps;
end
reject=false;
naccept=0; % priimtų žingsnių skaičius
nreject=0; % atmestų žingsnių skaičius
nstep=0;   % bendras žingsnių skaičius
i=1;
xspr(i)=x; yspr(:,i)=y;
% Pirmoji stadija
k(:,1) = feval(fun,x,y);
nfcn=1; % skaičiuotų funkcijos reikšmių skaičius
```

```
%  Pagrindiniai skaičiavimai
while   ((x-xend)*posneg+eps) <= 0
    if  (nstep > nmax)|(x+0.03*h == x)
        disp(['darbas nutraukiamas, kai x=',num2str(x)])
        disp('per daug mažas žingsnis')
        return
    end
    if  (x+h-xend)*posneg > 0
        h=xend-x;
    end
    nstep=nstep+1;
%  12-ka stadijų: nuo 2-os iki 13-os
    for j=2:13
       aa=repmat(a(j,1:j-1),n,1);
       if j==2
           y1=y+h*aa.*k(:,1);
       else
           y1=y+h*(sum((aa.*k(:,1:j-1))'))';
       end
       k(:,j) = feval(fun,x+c(j)*h,y1);
    end
    nfcn=nfcn+12;
    bt=repmat(b,n,1);
    y1=y+h*(sum((bt.*k)'))';% 8-os eiles sprendinys
    bt=repmat(bq,n,1);
    yh1=y+h*(sum((bt.*k)'))';% 7-os eiles sprendinys
    % Lokaliosios paklaidos apskaičiavimas
    err=abs(y1-yh1);
    for j=1:n
        d(j)=max([1e-5,abs(y(j)),abs(y1(j)),1e-7/ep]);
    end
    lokerr=err./d';
    errn=norm(lokerr/n);
%  Naujo žingsnio apskaičiavimas. Norime, kad 0.333<hnew/h<=6.
    fac=max([1/6,min([3,((errn/ep)^(1/8))/0.9])]);
    hnew=h/fac;
    if errn <= ep
  % Žingsnis priimamas
       i=i+1;
       xspr(i)=x+h; yspr(:,i)=y1;
       naccept=naccept+1;
       x=x+h; y=y1;
       k(:,1) = feval(fun,x,y);
       nfcn=nfcn+1;
       if abs(hnew) >= hmax
           hnew=posneg*hmax;
       end
       if reject
           hnew=posneg*min(abs(hnew),abs(h));
       else
           hnew=posneg*abs(hnew);
       end
       reject=false;
    else
       % Žingsnis atmetamas
       reject=true;
       if naccept >=1
```

```
            nreject=nreject+1;
        end
    end
    h=hnew;
end
if  stat == 1
    disp(['apskaičiuotų funkcijos reikšmių skaičius
           =',num2str(nfcn)])
    disp(['apskaičiuotų žingsnių skaičius =',num2str(nstep)])
    disp(['priimtų žingsnių skaičius =',num2str(naccept)])
    disp(['atmestų žingsnių skaičius =',num2str(nreject)])
end
```

Pastaba. *Visi skaičių masyvai, esantys tarp laužtinių skliaustų, o taip pat komandos* ***disp*** *tekstai turi būti vienoje eilutėje. Funkcijos* ***dopri8*** *tekste dėl lapo formato šis reikalavimas ne visur tenkinamas.*

9.2. Daugiažingsniai (prognozės ir korekcijos) metodai

Aptarti Teiloro eilučių bei Rungės ir Kutos metodai yra vienažingsniai metodai: sprendinio $y(x)$ aproksimacija y_{m+1} apskaičiuojama remiantis tik y_m reikšme ir keliomis funkcijos $f(x,y)$ reikšmėmis tarpiniuose taškuose, kurios tolesniuose skaičiavimuose nenaudojamos, kai tik apskaičiuojama y_{m+1} reikšmė. Kadangi skirtumas $|y(x_m)-y_m|$ auga didėjant m reikšmei, tai galima tikėtis geresnės y_{m+1} aproksimacijos, jei, apskaičiuodami y_{m+1}, pasinaudosime keliomis prieš tai buvusiomis sprendinio aproksimacijomis. Tokie metodai vadinami ***daugiažingsniais metodais***.

Kaip jau buvo minėta, daugiažingsniams metodams būdinga tai, kad, apskaičiuojant y_{m+1} reikšmę, remiamasi taškais $(x_{m-j}; y_{m-j})$; čia $j=\overline{0,k}$.

Tarkime, kad sprendžiame Koši uždavinį $y'=f(x,y)$, tenkinantį pradinę sąlygą $y(x_0)=y_0$, ir jau žinome šios diferencialinės lygties sprendinio taškus $(x_0; y_0), \ldots, (x_m; y_m)$. Tada

$$y_{m+1}=y_{m-j}+\int_{x_{m-j}}^{x_{m+1}} f(x,y(x))\,dx. \qquad (9.26)$$

(9.26) formulėje pointegralinę funkciją $f(x,y)$ pakeiskime interpoliaciniu polinomu, einančiu per taškus $(x_{m-i}; f(x_{m-i}, y(x_{m-i})))$. Kai $i=\overline{0,j}$, turime *ekstrapoliacines (išreikštines) formules*, o kai $i=\overline{-1,j}$ — *interpoliacines (neišreikštines) formules.*

Daugiažingsnius metodus paprastai sudaro metodų pora: prognozės metodas ir korekcijos metodas. Taigi (9.26) formulė yra prognozės formulė. Prognozuojamąją y_{m+1} reikšmę toliau galime koreguoti panaudodami tokios pat eilės korekcijos formulę, kaip ir prognozės formulė.

Korekcija yra būtina, nes prognozės metodas paprastai nėra stabilus. Be to, skirtumas tarp prognozuotos ir pakoreguotos reikšmės leidžia įvertinti lokaliąją metodo paklaidą.

Prognozės ir korekcijos metodai nėra savaime prasidedantys. Tai didžiausias jų trūkumas. Norint pradėti skaičiuoti arba keisti integravimo žingsnį, keletą sprendinio taškų reikia apskaičiuoti kokiu nors vienažingsniu metodu.

Yra įvairių korekcijos ir prognozės formulių išvedimo metodų. Originalų šių formulių išvedimo metodą, susijusį su kvadratūrinių formulių (žr. 8 skyrių) išvedimu, pasiūlė R. V. Hemingas.

Šį metodą paaiškinsime nagrinėdami 4-osios eilės prognozės ir korekcijos metodus.

9.2.1. Korekcijos metodas

Pati bendriausia korekcijos formulė, naudojanti sprendinio ir jo išvestinės reikšmes paskutiniuose trijuose taškuose bei išvestinės reikšmę taške, kuriame sprendinys turi būti apskaičiuotas, yra tokia:

$$y_{n+1} = a_0 y_n + a_1 y_{n-1} + a_2 y_{n-2} + h\big(b_{-1} y'_{n+1} + \\ + b_0 y'_n + b_1 y'_{n-1} + b_2 y'_{n-2}\big) + E_5 \frac{h^5 y^{(5)}(x)}{5!}. \qquad (9.27)$$

Galima naudoti ir kitas tiesines išraiškas, tačiau pastaroji apima daugelį žinomų metodų. Joje yra 7 parametrai. Penkis iš jų pasirinksime taip, kad (9.27) formulė būtų tiksli, kai diferencialinės lygties sprendinys $y(x) = 1, x, x^2, x^3$ ir x^4, o du — taip, kad metodas būtų stabilus. E_5 reikšmei apskaičiuoti panaudosime funkciją $y(x) = x^5$.

Tarę, kad $h = 1$, o $x = -2, -1, 0, 1$, turime sąlygas, kuriomis (9.27) formulė yra ketvirtosios eilės:

$$\begin{cases} a_0 = 1 - a_1 - a_2, & b_0 = \frac{1}{24}\big(19 + 13a_1 + 8a_2\big), \\ a_1 = a_1, & b_1 = \frac{1}{24}\big(-5 + 13a_1 + 32a_2\big), \\ a_2 = a_2, & b_2 = \frac{1}{24}\big(1 - a_1 + 8a_2\big), \\ b_{-1} = \frac{1}{24}\big(9 - a_1\big), & E_5 = \frac{1}{6}\big(-19 + 11a_1 - 8a_2\big). \end{cases} \qquad (9.28)$$

Parametrai a_1 ir a_2 parenkami taip, kad (9.27) formulė būtų stabili. Stabilumo klausimas nagrinėjamas literatūroje [**Hemming**],o stabilumo sritis priklauso nuo išraiškos Ah; čia $A = \frac{\partial f}{\partial y}$. Stabilumo sritis plokštumoje $a_1 \times a_2$ didžiausia yra tada, kai $Ah = 0$. Dauguma taikomų korekcijos metodų patenka į sritį, kai $|Ah| < 0{,}4$. Pavyzdžiui, plačiai žinomas Adamso ir Bašforto metodas patenka į šią sritį. Šis metodas gaunamas iš (9.28) sąlygų, kai $a_1 = a_2 = 0$. Tada (9.27) formulė įgyja išraišką

$$y_{n+1} = y_n + \frac{1}{24} h \big(9y'_{n+1} + 19y'_n - \\ - 5y'_{n-1} + y'_{n-2}\big) - \frac{19}{720} h^5 y^{(5)}(x). \qquad (9.29)$$

Jei $a_1 = 1$, $a_2 = 0$, tai (9.28) formulė apibūdins Milno korekcijos formulę

$$y_{n+1} = y_{n-1} + \frac{1}{3}h\left(y'_{n+1} + 4y'_n + y'_{n-1}\right) - \frac{1}{90}h^5 y^{(5)}(x).$$

Jei (9.28) formulėje parametrų a_1 ir a_2 yra: $a_1 = 0$, $a_2 = 1$, tai turėsime 3/8 metodą.

9.2.2. Prognozės metodas

Prognozė pagal tris taškus. Prognozės formulė, sudaryta remiantis trimis paskutiniais taškais, turės tokią išraišką:

$$y_{n+1} = A_0 y_n + A_1 y_{n-1} + A_2 y_{n-2} + \\ + h\left(B_0 y'_n + B_1 y'_{n-1} + B_2 y'_{n-2}\right) + \overline{E}_5 \frac{h^5 y^{(5)}}{5!}. \qquad (9.30)$$

Koeficientus A_0, A_1, B_0, B_1 ir B_2 parinkime taip, kad (9.30) formulė būtų tiksli, kai diferencialinės lygties sprendinys $y(x) = 1, x, x^2, x^3$ ir x^4. Tada turėsime sąlygas

$$\begin{cases} A_0 = -8 - A_2, & B_0 = \dfrac{17 + A_2}{3}, \\ A_1 = 9, & B_1 = \dfrac{14 + 4A_2}{3}, \\ A_2 = A_2, & B_2 = \dfrac{-1 + A_2}{3}, \\ & \overline{E}_5 = \dfrac{40 - 4A_2}{3}. \end{cases} \qquad (9.31)$$

Kadangi koeficientas $A_1 = 9$, tai, skaičiuojant pagal (9.31) formulę, labai greitai kaupsis apvalinimo paklaida. Todėl praktiškai imama 3-iosios eilės prognozės pagal tris taškus formulė. Kad (9.30) formulė būtų tiksli, kai $y(x) = 1, x, x^2$ ir x^3, koeficientai turi tenkinti tokias sąlygas:

$$\begin{cases} A_0 = 1 - A_1 - A_2, & B_1 = \dfrac{-16 + 8A_1 + 16A_2}{12}, \\ A_1 = A_1, & \\ A_2 = A_2, & B_2 = \dfrac{5 - A_1 + 4A_2}{12}, \\ B_0 = \dfrac{23 + 5A_1 + 4A_2}{12}, & \overline{E}_4 = 9 - A_1. \end{cases} \qquad (9.32)$$

Jei norime prognozės formulėje turėti 7 parametrus, tai į (9.30) formulę įrašome arba y_{n-3}, arba y'_{n-3}. Pirmuoju atveju turėsime Milno, o antruoju atveju — Adamso ir Bašforto prognozės formules.

Milno prognozės formulė. Ji turi tokią išraišką:

$$y_{n+1} = A_0 y_n + A_1 y_{n-1} + A_2 y_{n-2} + A_3 y_{n-3} + \\ + h\left(B_0 y'_n + B_1 y'_{n-1} + B_2 y'_{n-2}\right) + \overline{E}_5 \frac{h^5 y^{(5)}}{5!}. \qquad (9.33)$$

Kad (9.33) formulė būtų ketvirtosios eilės, koeficientai turi tenkinti šias sąlygas:

$$\begin{cases} A_0 = -8 - A_2 + 8A_3,\ B_0 = \dfrac{17 + A_2 - 9A_3}{3}, \\ A_1 = 9 - 9A_3, \qquad B_1 = \dfrac{14 + 4A_2 - 18A_3}{3}, \\ A_2 = A_2, \qquad B_2 = \dfrac{-1 + A_2 + 9A_3}{3}, \\ A_3 = A_3, \qquad \overline{E}_5 = \dfrac{40 - 4A_2 + 72A_3}{3}. \end{cases} \tag{9.34}$$

(9.33) formulė sutaps su Milno prognozės formule, jei bus $A_2 = 0$, $A_3 = 1$. Tada $A_0 = A_1 = 0$, $B_0 = 8/3$, $B_1 = -4/3$, $B_2 = 8/3$. Vadinasi,

$$y_{n+1} = y_{n-3} + \frac{4}{3}h\left(2y'_n - y'_{n-1} + 2y'_{n-2}\right) + \frac{14}{45}h^5 y^{(5)}(x) \tag{9.35}$$

yra Milno prognozės formulė.

Adamso ir Bašforto prognozės formulė. Ši formulė yra tokia:

$$\begin{aligned} y_{n+1} &= A_0 y_n + A_1 y_{n-1} + A_2 y_{n-2} + h\left(B_0 y'_n + B_1 y'_{n-1} + \right. \\ &\left. + B_2 y'_{n-2} + B_3 y'_{n-3}\right) + \overline{E}_5 \frac{h^5 y^{(5)}}{5!}. \end{aligned} \tag{9.36}$$

Sąlygas, kurias turi tenkinti koeficientai, galime užrašyti taip:

$$\begin{cases} A_0 = 1 - A_1 - A_2, \qquad B_1 = \dfrac{-59 + 19A_1 + 32A_2}{24}, \\ A_1 = A_1, \qquad B_2 = \dfrac{37 - 5A_1 + 8A_2}{24}, \\ A_2 = A_2, \qquad B_3 = \dfrac{-9 + A_1}{24}, \\ B_0 = \dfrac{55 + 9A_1 + 8A_2}{24}, \quad \overline{E}_5 = \dfrac{251 - 19A_1 - 8A_2}{6}. \end{cases} \tag{9.37}$$

Kai $A_1 = A_2 = 0$, iš (9.36) formulės apskaičiuojame Adamso ir Bašforto prognozės formulę

$$\begin{aligned} y_{n+1} &= y_n + \frac{1}{24}h\left(55y'_n - 59y'_{n-1} + \right. \\ &\left. + 37y'_{n-2} - 9y'_{n-3}\right) + \frac{251}{720}h^5 y^{(5)}(x). \end{aligned} \tag{9.38}$$

Trumpai apibūdinsime dažniausiai taikomus prognozės ir korekcijos metodus .

Milno metodas

Prognozės formulė: (9.35) formulė.

Korekcijos formulė:

$$y_{n+1} = y_{n-1} + \frac{1}{3}h\left(y'_{n+1} + 4y'_n + y'_{n-1}\right) - \frac{1}{90}h^5 y^{(5)}(x). \tag{9.39}$$

Adamso ir Bašforto metodas

Prognozės formulė: (9.38) formulė.
Korekcijos formulė: (9.29) formulė.

Hemingo metodas

Prognozės formulė: (9.35) formulė.
Korekcijos formulė:

$$y_{n+1} = y_{n-1} + \frac{1}{8}\left(9y_n - y_{n-2} + 3h\left(y'_{n+1} + \right.\right. \\ \left.\left. + 2y'_n - y'_{n-1}\right)\right) - \frac{1}{40}h^5 y^{(5)}(x). \qquad (9.40)$$

Sprendžiant paprastas diferencialines lygtis, prognozės ir korekcijos metodai plačiai taikomi dėl šių priežasčių:

1) prognozuotos ir sukoreguotos reikšmių skirtumas įvertina sprendimo metodo lokaliąją paklaidą ir gali būti naudojamas integravimo žingsniui valdyti;

2) kiekviename žingsnyje, skirtingai nei taikant vienažingsnius metodus, išvestinę (dešiniąją diferencialinės lygties pusę $f(x, y)$) reikia apskaičiuoti vieną arba du kartus, kas įgalina sutrumpinti sprendimo laiką.

Pagrindinis šių metodų trūkumas tas, kad, apskaičiuojant sprendinį eiliniame taške, reikia žinoti sprendinius keliuose prieš tai buvusiuose vienodai nutolusiuose vienas nuo kito taškuose. Vadinasi, šie metodai nėra savaime prasidedantys. Pradedant spręsti diferencialinę lygtį arba keičiant integravimo žingsnį, reikia keletą vienodai nutolusių taškų apskaičiuoti taikant kurį nors vienažingsnį metodą.

Pastaba. XX a. septintajame dešimtmetyje buvo sukurti kintamo žingsnio daugiažingsniai metodai. Detalus jų aprašymas pateiktas literatūroje [**HNW87**].

Kaip apskaičiuoti lokaliąją paklaidą? Į šį klausimą pamėginkime atsakyti imdami konkretų, pavyzdžiui, Milno, metodą:

$$y_{n+1} = y^{(p)}_{n+1} + \frac{28}{90}h^5 y^{(5)}(x),$$

$$y_{n+1} = y^{(k)}_{n+1} - \frac{1}{90}h^5 y^{(5)}(x);$$

čia indeksai *p* ir *k* žymi atitinkamai prognozuotą ir pakoreguotą reikšmę.

Iš šių formulių randame:

$$\frac{1}{90}h^5 y^{(5)}(x) = \frac{1}{29}\left(y^{(k)}_{n+1} - y^{(p)}_{n+1}\right).$$

Tada

$$y_{n+1} = y^{(k)}_{n+1} + \frac{1}{29}\left(y^{(k)}_{n+1} - y^{(p)}_{n+1}\right).$$

Prognozės ir korekcijos metodų algoritmų loginės schemos praktiškai yra tos pačios, o algoritmai skiriasi tik formulėmis. Apibendrintą šių metodų algoritmų loginę schemą galime parašyti taip.

begin
1. Taikant tinkamos eilės (ne žemesnės kaip prognozės ir korekcijos metodų eilė) vienažingsnį metodą, apskaičiuoti reikiamą kiekį pradinių taškų.

2. While not „integravimo intervalo pabaiga“ do
begin

2.1. Pagal prognozės formulę apskaičiuoti prognozuotą reikšmę y_{n+1}^{0}.

2.2. Apskaičiuoti išvestinės reikšmę, t. y. diferencialinės lygties dešiniąją pusę $y_{n+1}^{(0)\prime} = f\left(x_{n+1}, y_{n+1}^{(0)}\right)$.

2.3. i:=0; z:=1+ε; / y' pokytis*/*

2.4. while z > ε do
begin

2.4.1. Pagal korekcijos formulę apskaičiuoti $y_{n+1}^{(i+1)}$.

2.4.2. Apskaičiuoti $y_{n+1}^{(i+1)\prime} = f\left(x_{n+1}, y_{n+1}^{(i+1)}\right)$.

2.4.3. Apskaičiuoti išvestinės pokytį z;

2.4.4. $i := i + 1$;

end;

2.5. Remiantis $y_{n+1}^{(0)}$ *ir* $y_{n+1}^{(i)}$ *reikšmėmis, apskaičiuoti* y_{n+1} *reikšmę.*

end;

end;

9.3. Standžių diferencialinių lygčių sprendimo metodai

Visi skaitiniai diferencialinių lygčių sprendimo metodai turi paklaidas, į kurias kaip daugiklis įeina diferencialinės lygties sprendinio aukštesnės eilės išvestinė. Jei tos išvestinės reikšmė yra aprėžta, tai metodas turės prognozuojamo dydžio paklaidą, t.y. ją galima įvertinti ir panaudoti parenkant integravimo žingsnį, kuris leistų apskaičiuoti norimo tikslumo sprendinį. Net ir tada, kai didėjant argumento reikšmei šios išvestinės reikšmė auga, sprendinio santykinę paklaidą galima valdyti, jei tik sprendinio reikšmė taip pat auga. Problemos dažniausiai atsiranda tada, kai išvestinės reikšmė auga, o sprendinio reikšmė nedidėja. Šiuo atveju sprendinio paklaida gali augti taip sparčiai, kad paklaidos reikšmė taps dominuojanti. Diferencialinės lygtys, turinčios šią savybę vadinamos ***standžiomis diferencialinėmis lygtimis.*** Tokios diferencialinės lygtys paprastai aprašo vibracijas, chemines reakcijas bei elektrines grandines. Jų pavadinimas siejamas su judėjimu sistemos, sudarytos iš standžios spyruoklės ir masės.

Standžių diferencialinių lygčių tikslūs sprendiniai dažniausiai turi e^{-cx} formos dėmenį, čia c didelė teigiama konstanta. Šis dėmuo apibrėžia sprendinio dalį, vadinamą ***pereinamuoju sprendiniu***. Žymiai svarbesni yra sprendinio dėmenys, aprašantys ***nusistovėjusį sprendinį***. Pereinamojo sprendinio dėmens reikšmė, augant argumento reikšmei x, sparčiai mažėja. Kadangi šio dėmens *n*-os eilės išvestinės modulis yra $c^n e^{-cx}$, tai, išvestinės reikšmė mažėja žymiai lėčiau nei sprendinio reikšmė. Be to, išvestinės reikšmė paklaidos išraiškoje apskaičiuojama imant ne *x* reikšmę, bet skaičių tarp x_0 ir *x*. Vadinasi, didėjant *x* reikšmei, išvestinės reikšmė auga ir labai sparčiai.

Standžios diferencialinės lygties sprendimo specifiką panagrinėkime Oilerio metodu spręsdami paprastą testinį standžios diferencialinės lygties pavyzdį:

$y' = \lambda y$, $y(0) = \alpha$, čia λ - didelio modulio neigiamas skaičius.

Šios lygties pereinamąjį sprendinį nusako išraiška: $e^{\lambda x}$, o nusistovėjęs sprendinys yra $y = 0$.

Spręsdami šią lygtį Oilerio metodu, naudokime pastovų integravimo žingsnį h. Tada skaitinis sprendinys bus nusakomas formulėmis:

$$x_i = ih,$$
$$y_0 = \alpha,$$
$$y_{i+1} = y_i + h(\lambda y_i) = (1+h\lambda)y_i = (1+h\lambda)^{i+1}\alpha, i = 1,2,3,\ldots.$$

Kadangi tikslus lygties sprendinys yra $y(x) = e^{\lambda x}$, tai skaitinio sprendinio absoliutinė paklaida apskaičiuojama pagal formulę: $|y(x_i) - y_i| = |e^{\lambda hi} - (1+h\lambda)^i||\alpha|$. Tuo būdu tikslumas priklauso nuo to, kaip gerai dvinaris $1+h\lambda$ aproksimuoja $e^{h\lambda} = 1 + h\lambda + \frac{1}{2!}(h\lambda)^2 + \frac{1}{3!}(h\lambda)^3 + \ldots$. Skirtumas tarp $1+h\lambda$ ir $e^{h\lambda}$ yra $O((h\lambda)^2)$, t.y. augant $|\lambda|$ reikšmei, skirtumas didėja kvadratiniu greičiu.

Kai λ - didelio modulio neigiamas skaičius, tai tikslus sprendinys sparčiai artėja prie nulio, tačiau Oilerio metodo skaitinis sprendinys artės prie nulio tik esant sąlygai: $|1+h\lambda| < 1$. Ši sąlyga labai apriboja integravimo žingsnio dydį: $h < 2/|\lambda|$.

Tarkime, kad pradinė sąlyga turi apvalinimo paklaidą: $y_0 = \alpha + \delta_0$. Tada *i*-me žingsnyje apvalinimo paklaida bus apskaičiuojama pagal formulę: $\delta_i = (1+h\lambda)^i \delta_o$, t.y. turės tokį pat pobūdį, kaip ir metodo paklaida. Vadinasi, norint, kad jinai neaugtų , žingsnis *h* turi tenkinti tą pačią sąlygą: $h < 2/|\lambda|$.

Panaši situacija yra ir su kitais išreikštinais vienažingsniais metodais, taikant kuriuos skaitinis sprendinys apskaičiuojamas pagal formulę: $y_{i+1} = Q(h\lambda)y_i$, čia funkcija Q priklauso nuo konkretaus metodo. Vadinasi, skaitinio sprendinio tikslumas priklauso nuo to, kaip tiksliai funkcija $Q(h\lambda)$ aproksimuoja eksponentę $e^{h\lambda}$.

Sprendžiant standžias lygtis išreištiniais daugiažingsniais (prognozės - korekcijos) metodais gaunama žymiai sudėtingesnė situacija, nes koreguojant sprendinį kelis kartus naudojama prieš tai buvusi aproksimacija. Todėl pagrindiniai standžių diferencialinių lygčių sprendimo metodai yra neišreikštinai metodai, kuriuos taikant kiekviename žingsnyje reikia išspręsti netiesinę lygtį arba netiesinių lygčių sistemą.

9.3.1 Trapecijų metodas

Vienas iš paprasčiausių neišreikštinių metodų, skirtų standžioms diferencialinėms lygtims spręsti, yra ***trapecijų*** metodas. Šis metodas aprašomas formulėmis:

$$x_{m+1} = x_m + h,$$
$$y_{m+1} = y_m + h(f(x_m, y_m) + f(x_{m+1}, y_{m+1}))/2.$$

Kaip matome, kiekviename trapecijų metodo žingsnyje, kad apskaičiuotume y_{m+1} reikšmę, reikia išspręsti netiesinę lygtį arba jų sistemą.

Panagrinėkime, kaip kaupiasi skaičiavimo paklaida sprendžiant lygtį

$$y' = ky, y(x_0) = y_0.$$

Aišku, kad šiuo atveju $y_{m+1} = y_m + h(ky_m + ky_{m+1})/2$.

Iš šios lygties išreiškime y_{m+1}:

$$y_{m+1} = \big((1 + kh/2)/(1 - kh/2)\big) y_m .$$

Šią formulę taikydami pakartotinai, turėsime:

$$y_{m+1} = \big((1 + kh/2)/(1 - kh/2)\big)^{m+1} y_0 .$$

Tarkime, kad pradinė y-o reikšmė turi paklaidą, t.y. $y_0^a = y_0 + e_0$. Tada diferencialinės lygties sprendinys bus apskaičiuojamas pagal formulę:

$$y_{m+1}^a = \big((1 + kh/2)/(1 - kh/2)\big)^{m+1} \big(y_0 + e_0\big).$$

Iš šios formulės turime:

$$y_{m+1}^a = y_{m+1} + \big((1 + kh/2)/(1 - kh/2)\big)^{m+1} e_0 .$$

Vadinasi, jei diferencialinė lygtis tenkina sąlygą:

$$\big|(1 + kh/2)/(1 - kh/2)\big| < 1,$$

tai trapecijų metodas yra absoliučiai stabilus. Aišku, kad ši sąlyga galioja, jei $k < 0$; jei $k > 0$, metodas yra santykinai stabilus.

Galima parodyti, kad trapecijų metodas yra antros eilės metodas, kurio lokalioji paklaida yra $O\big(h^3\big)$.

9.3.2 Neišreikštinis 5-os eilės Rado (Radau) metodas

Panagrinėkime neišreikštinį 5-os eilės Rado (Radau) metodą, kuris yra tikslesnis nei trapecijų metodas.

Pirmiausia apibrėšime neišreikštinį ***s***–stadijų ***p***-os eilės metodą. Neišreikštinai Rungės ir Kutos metodai, analogiškai išreikštiniems metodams, apibrėžiami Bučerio pasiūlyta lentele (žr. 9.1.2 paragrafą).

9.10 lentelė. ***s*** *stadijų neišreikštinis Rungės ir Kutos metodas*

c_1	a_{11}	a_{12}	$\cdots$	$a_{1,s-1}$	$a_{1,s}$
c_2	a_{21}	a_{22}	$\cdots$	$a_{2,s-1}$	$a_{2,s}$
c_3	a_{31}	a_{32}	$\cdots$	$a_{3,s-1}$	$a_{3,s}$
$\vdots$			$\cdot\ \cdot\ \cdot$		
c_s	a_{s1}	a_{s2}	$\cdots$	$a_{s,s-1}$	$a_{s,s}$
	b_1	b_2	$\cdots$	b_{s-1}	b_s

9.10 lentelė nusako ***s****-stadijų neišreikštinį Rungės ir Kutos metodą, kuris aprašomas formulėmis:*

$$g_i = y_0 + h\sum_{j=1}^{s} a_{ij} f\big(x_0 + c_j h, g_j\big), \qquad i = 1,2,\ldots,s, \tag{9.40}$$

$$y_1 = y_0 + h\sum_{j=1}^{s} b_j f\big(x_0 + c_j h, g_j\big). \tag{9.41}$$

s*-stadijų neišreikštinis Rungės ir Kutos metodas yra* ***p****-os eilės, jei* (*9.41*) *formulė įvertina sprendinio Teiloro eilutės narius iki* ***p****-os eilės išvestinės imtinai.*

Praleisdami neišreikštinių Rungės ir Kutos metodų išvedimo klausimus, (žr. *R. Čiegis, Diferencialinių lygčių skaitiniai sprendimo metodai, Vilnius, Technika, 2003*) pateiksime trijų stadijų penktos eilės Rado metodą (žr. *E.Hairer, G.Wanner, Solving Ordinary Differential Equations II, Stiff and Differential-Algebraic Problems, Springer-Verlag, Berlin, Heidelberg, 1987*), kurį nusako 9.11 lentelė.

9.11 lentelė. Trijų stadijų penktos eilės Rado metodas

$\frac{4-\sqrt{6}}{10}$	$\frac{88-7\sqrt{6}}{360}$	$\frac{296-169\sqrt{6}}{1800}$	$\frac{-2+3\sqrt{6}}{225}$
$\frac{4+\sqrt{6}}{10}$	$\frac{296+169\sqrt{6}}{1800}$	$\frac{88+7\sqrt{6}}{360}$	$\frac{-2-3\sqrt{6}}{225}$
1	$\frac{16-\sqrt{6}}{36}$	$\frac{16+\sqrt{6}}{36}$	$\frac{1}{9}$
	$\frac{16-\sqrt{6}}{36}$	$\frac{16+\sqrt{6}}{36}$	$\frac{1}{9}$

Rado metodo realizacija. Aptarsime Rado metodo programinę realizaciją. Pradžioje aptarsime teorinius klausimus, o po to pateiksime Rado metodą realizuojančią matlabinę funkciją ir standžių lygčių sprendimo pavyzdžius.

Skaičiavimo paklaidų sumažinimas. Norėdami sumažinti apvalinimo paklaidų įtaką, skaičiavimus atliksime su galimai mažesniais skaičiais. Tam tikslui įveskime pakeitimą:

$$z_i = g_i - y_0 \ .$$

Tada (9.40) formulė taps

$$z_i = h\sum_{j=1}^{s} a_{ij} f\left(x_0 + c_j h, y_0 + z_j\right), \ \ i = 1,2,\ldots,s \ . \qquad (9.42)$$

Kai apskaičiuosime (9.42) sistemos sprendinį $\left(z_1, z_2, \ldots, z_s\right)$, pagal išreikštinę (9.41) formulę bus galima apskaičiuoti y_1 reikšmę. Tiesiogiai pagal formulę skaičiuojant y_1 reikšmę, reikės papildomai apskaičiuoti s funkcijos $f\left(x, y\right)$ reikšmių. Šių skaičiavimų galime išvengti, jei neišreikštinį Rungės ir Kutos metodą nusakanti matrica (lentelė) A yra neišsigimusi.

Remiantis matrica A, (9.42) formulę galima užrašyti taip:

$$\begin{pmatrix} z_1 \\ \vdots \\ z_s \end{pmatrix} = A \begin{pmatrix} hf\left(x_0 + c_1 h, y_0 + z_1\right) \\ \vdots \\ hf\left(x_0 + c_s h, y_0 + z_s\right) \end{pmatrix}.$$

Tada (9.41) formulę bus galima perrašyti taip:

$$y_1 = y_0 + \sum_{i=1}^{s} b_i h f(x_0 + c_i h, g_i) = y_0 + (b_1, b_2, \ldots, b_s) A^{-1} (z_1, z_2, \ldots, z_s)^t = y_0 + \sum_{i=1}^{s} d_i z_i ,$$ čia $(d_1, d_2, \ldots, d_s) = (b_1, b_2, \ldots, b_s) A^{-1}$. (9.43)

Kadangi Rado metodo trečiosios stadijos koeficientai yra lygūs svoriniams koeficientams *b,* t.y. $b_i = a_{3i}, i = 1,2,3$, tai vektorius $d = (0,0,1)$.

(9.43) formulė turi dar vieną privalumą. Kadangi (9.42) sistemos sprendinys $(z_1, z_2, \ldots, z_s)$ apskaičiuojamas apytiksliai, tai, apskaičiuojant funkcijų $f(x_0 + c_i h, y_0 + z_i), i = \overline{1,s}$ reikšmes, esant didelei funkcijos $f(x, y)$ Lipšico konstantai, ši paklaida gali dar daugiau išaugti. Vadinasi, (9.43) formulė padeda išvengti šios blogybės.

Supaprastintas Niutono metodas. Sprendžiant *n* netiesinių diferencialinių lygčių sistemą neišreikštiniu *s* stadijų Rungės ir Kutos metodu, kiekviename žingsnyje reikės išspręsti $n \times s$ netiesinių lygčių (9.42) sistemą. Šią sistemą geriausia spręsti Niutono metodu, nes paprastųjų iteracijų metodas pažeidžia geras neišreikštinio Rungės ir Kutos metodo savybes.

Sprendžiant $n \times s$ netiesinių lygčių (9.42) sistemą Niutono metodu, kiekvienoje šio metodo iteracijoje reikės išspręsti tiesinę lygčių sistemą, kurios matrica yra:

$$\begin{pmatrix} I_n - h a_{11} \dfrac{\partial f}{\partial y}(x_0 + c_1 h, y_0 + z_1) \cdots - h a_{1s} \dfrac{\partial f}{\partial y}(x_0 + c_s h, y_0 + z_s) \\ \vdots \\ - h a_{s1} \dfrac{\partial f}{\partial y}(x_0 + c_1 h, y_0 + z_1) \cdots I_n - h a_{ss} \dfrac{\partial f}{\partial y}(x_0 + c_s h, y_0 + z_s) \end{pmatrix},$$

čia I_n *n-os* eilės vienetinė matrica.

Niutono metodą supaprastinsime visas tikslias Jakobianas $J = \dfrac{\partial f}{\partial y}(x_0 + c_i h, y_0 + z_i)$ pakeisdami Jakobiana $\dfrac{\partial f}{\partial y}(x_0, y_0)$, t.y. $J \approx \dfrac{\partial f}{\partial y}(x_0, y_0)$. Tada supaprastinta *k-oji* Niutono iteracija bus aprašoma formulėmis:

$$(I_{n \times s} - hA \otimes J) \Delta Z^k = -Z^k + h(A \otimes I_n) F(Z^k), \qquad (9.44)$$

$$Z^{k+1} = Z^k + \Delta Z^k .$$

čia $Z^k = (z_1, z_2, \ldots, z_s)$ yra (9.42) sistemos *k-oji* aproksimacija, $\Delta Z^k = (\Delta z_1^k, \Delta z_2^k, \ldots, \Delta z_s^k)^t$ yra *k-os* aproksimacijos pokytis, simbolis „$\otimes$“ žymi matricų Kronekerio tenzorinę sandaugą (žr. 2.3.3 paragrafą), o sutrumpinimas $F(Z^k)$ žymi vektorių $F(Z^k) = \left(f(x_0 + c_1 h, y_0 + z_1^k), \ldots, f(x_0 + c_s h, y_0 + z_s^k)\right)^t$.

Vadinasi, kiekvienoje iteracijoje reikia *s* kartų apskaičiuoti funkcijas *f* ir išspręsti $n \times s$ formato tiesinių lygčių sistemą , kurios koeficientų matrica yra pastovi: $(I_{n \times s} - hA \otimes J)$.

Pastaba. Literatūroje [**30**] nagrinėjamas Bučerio ir Bikarto (J.C. Butcher, ir T.A. Bickart) sukurtas metodas, kaip pertvarkyti (9.44) sistemą, kad skaičiavimo apimtis būtų galimai mažiausia. Šis metodas duoda apčiuopiamą naudą, jei Rungės ir Kutos neišreikštinio metodo stadijų skaičius yra didelis (žymiai didesnis už tris). Todėl Bučerio ir Bikaro modifikacijos čia nenagrinėsime.

Pradinio sprendinio parinkimas. Sprendžiant (9.42) sistemą Niutono metodu, kiekviename žingsnyje reikia parinkti pradinio sprendinio reikšmes.

Kadangi tikslus (9.42) sistemos sprendinys tenkina sąlygą: $z_i = O(h)$, tai natūralu, kad pradinio sprendinio reikšmės nusakomos formule:

$$z_i^0 = 0, i = 1,2,\ldots,s\,. \qquad (9.45)$$

Tačiau yra ir geresnis šių reikšmių parinkimo būdas. Literatūroje [**30**] parodyta, kad, jei neišreikštinis Rungės ir Kutos metodas kažkokiam $\eta \le s$ tenkina sąlygą $C(\eta)$:

$$C(\eta)\colon \sum_{j=1}^{s} a_{ij} c_j^{q-1} = \frac{c_i^q}{q},\ i = 1,2,\ldots,s,\ q = 1,2,\ldots,\eta,$$

tai $z_i = y(x_0 + c_i h) - y_0 + O(h^{\eta+1})$.

Kadangi Rado metodas šią sąlygą tenkina, tai z_i^0 reikšmes apskaičiuosime taip.

Panagrinėkime *s* laipsnio interpoliacinį polinomą $q(x)$, kurį apibrėžia taškai:

$$q(0) = 0, q(c_i) = z_i, i = 1,2,\ldots,s\,,$$

čia z_i - reikšmės, apskaičiuotos prieš tai buvusiame žingsnyje.

Kadangi interpolacinio polinomo $q(x)$ paklaida yra $O(h^{s+1})$, tai drauge su (9.45) formule turėsime: $y(x_0 + c_i h) - y_0 - q(x) = O(h^{\eta+1})$. Parenkant z_i^0 reikšmes, pasinaudosime polinomo $q(x)$ reikšmėmis už intervalo $[0;1]$ ribų ir z_i^0 reikšmes apskaičiuosime pagal formules:

$z_i^0 = q(1 + wc_i) + y_0 - y_1$, $i = 1,2,\ldots,s$, h_{new} / h_{old}, čia h_{old}, h_{new} - atitinkamai prieš ir dabartinio žingsnių ilgiai, o y_0 ir y_1 atitinkamai dviejų paskutinių žingsnių *y*-ų reikšmės.

Apskaičiavus z_i^0 reikšmes pagal pastarąją formulę, konvergavimo greitis dažniausiai yra didesnis, nei z_i^0 reikšmes parinkus pagal (9.45) formulę. Aišku, kad pirmajame žingsnyje z_i^0 reikia parinkti pagal (9.45) formulę.

Supaprastinto Niutono metodo pabaigos sąlyga. Supaprastinto Niutono metodo konvergavimo greitis yra tiesinis, todėl

$\|\Delta Z^{k+1}\| \le \Theta \|\Delta Z^k\|$, čia (tikėtina, kad) $\Theta < 1$.

Tarkime, kad Z^* yra tikslus (9.44) sistemos sprendinys. Kadangi

$Z^{k+1} - Z^* = (Z^{k+1} - Z^{k+2}) + (Z^{k+2} - Z^{k+3}) + \ldots$, tai $\|Z^{k+1} - Z^*\| \le \frac{\Theta}{1-\Theta} \|\Delta Z^k\|$.

Konvergavimo greitį apibūdinanti konstanta Θ gali būti įvertinta pagal formulę:

$\Theta_k = \|\Delta Z^k\| / \|\Delta Z^{k-1}\|, k \ge 1$.

Aišku, kad iteracijos paklaida turi būti mažesnė už norimą paklaidą *tol*. Todėl supaprastinto Niutono metodo iteracijas nutrauksime, kai

$\eta_k \left\| \Delta Z^k \right\| \le \kappa * tol$, čia $\eta_k = \dfrac{\Theta_k}{1-\Theta_k}$ ir laikysime, kad Z^{k+1} yra norimo tikslumo (9.42) sistemos sprendinys.

Ši strategija gali būti taikoma mažiausiai po dviejų iteracijų. Tam, kad būtų galimybė patikrinti, ar po vienos iteracijos gautas norimo tikslumo sprendinys, koeficientą η_0 apskaičiuosime pagal formule: $\eta_0 = \left(\max(\eta_{old}, eps)\right)^{0.8}$, čia η_{old} prieš tai buvusiame žingsnyje apskaičiuotas koeficientas η, o *eps* - mažiausias skaičius tenkinantis nelygybę 1+*eps*>1 (žr.2.2 paragrafą).

Aptarkime koeficiento κ parinkimo klausimą. Eksperimentiškai nustatyta, kad geriausia κ reikšmė priklauso intervalui $\left[10^{-1};10^{-2}\right]$. Žemiau pateiktoje funkcijoje ***sprniut*** $\kappa = 10^{-2}$.

Kadangi supaprastinto Niutono metodo konvergavimo greitis yra tiesinis, tai maksimalų iteracijų skaičių $k_{\max}$ siūloma imti iš intervalo $[7;10]$ (žemiau pateiktoje funkcijoje ***sprniut*** $k_{\max} = 7$). Skaičiavimo eigoje, iteracijos nutraukiamos nepasiekus $k_{\max}$ ir skaičiavimas atliekamas su mažesniu integravimo žingsniu (pvz. su $h := h/2$), jei *k*-oje iteracijoje atsitinka viena iš žemiau išvardintų situacijų:

1. konvergavimo greitį apibūdinanti konstanta $\Theta_k \ge 1$ (metodas diverguoja);
2. $\dfrac{\Theta_k^{k_{\max}-k}}{1-\Theta_k}\left\| \Delta Z^k \right\| > \kappa * tol$ (kairės pusės išraiška yra grubus paklaidos įvertis po $k_{\max}-1$ iteracijų.

Metodo paklaidos įvertinimas ir integravimo žingsnio ilgio valdymas. Kiekviename metodo žingsnyje paklaidą įvertinsime pasinaudodami įdėtosiomis formulėmis. Kadangi trijų stadijų Rado metodas eilės požiūriu yra optimalus, t.y. naudojant tas pačias stadijas neįmanoma sukonstruoti aukštesnės eilės metodo, tai metodo paklaidą įvertinsime naudodami žemesnės eilės metodą, kurį aprašo formulė [30]:

$$y_1^* = y_0 + h\left(b_0^* f(x_0, y_0) + \sum_{i=1}^{3} b_i^* f(x_0 + c_i h, g_i) \right), \qquad (9.46)$$

čia g_1, g_2, g_3 reikšmės, apskaičiuotos taikant Rado metodą ir $b_0^* \ne 0$. Literatūroje [30] rekomenduojama imti $b_0^* = \gamma_0 = 0.27488882959568$, čia γ_0 Rado metodą aprašančios matricos A realioji tikrinė reikšmė. Tada skirtumas $\left| y_1^* - y_1 \right|$ gali būti traktuojamas kaip Rado metodo paklaidos įvertis.

Skirtumą $y_1^* - y_1 = \gamma_0 h f(x_0, y_0) + \sum_{i=1}^{3} (b_i^* - b_i) h f(x_0 + c_i h, g_i)$, galima perrašyti taip: $y_1^* - y_1 = \gamma_0 h f(x_0, y_0) + e_1 z_1 + e_2 z_2 + e_3 z_3$. Tam, kad skirtumas $\left| y_1^* - y_1 \right|$ būtų $O(h^4)$ eilės, koeficientai e_1, e_2, e_3 turi tenkinti sąlygą [30]:

$$(e_1, e_2, e_3) = \frac{\gamma_0}{3}\left(-13 - 7\sqrt{6}, -13 + 7\sqrt{6}, -1\right).$$

Tačiau lygčiai $y'=\lambda y$, kai $h\lambda \to \infty$, skirtumas $\left|y_1^* - y_1\right| \approx \left|\gamma_0 h\lambda y_0\right|$ yra neaprėžtas. Todėl paklaidą siūloma skaičiuoti pagal formulę:

$$err = \left(I_n - h\gamma_0 J\right)^{-1}\left(y_1^* - y_1\right).$$

Kai $h \to 0$, paklaida yra $O\left(h^4\right)$ eilės; kai $h\lambda \to \infty$ (jei $y'=\lambda y$, tai $J=\lambda$),tai $err \to -1$.

Kai turime metodo įvertį *err*, tai integravimo žingsnio valdymo formulė yra:

$h_{new} = fac * h_{old} * \left(\frac{tol}{\|err\|}\right)^{0.25}$, čia fac saugos daugiklis, kuris priklauso nuo atliktų Niutono iteracijų skaičiaus $Newt$, sprendžiant (9.44) sistemą, ir apskaičiuojamas pagal formulę: $fac = 0.9 * (2k_{\max}+1)/(2k_{\max} + Newt)$, čia $k_{\max}$ didžiausias leistinas Niutono iteracijų skaičius (parastai $k_{\max} \in [7;10]$); h_{old} ir h_{new} - atitinkamai paskutinio ir naujo Rado metodo žingsnio ilgis; tol - norimas y_1 tikslumas.

Žemiau pateiktoje ***rado5*** funkcijoje realizuotos tokios strategijos:

- pirmame žingsnyje ir po kiekvieno atmesto žingsnio metodo paklaida apskaičiuojama pagal kitą formulę:

  $$err_1 = \left(I_n - h\gamma_0 J\right)^{-1}\left(\gamma_0 hf\left(x_0, y_0 + err\right) + e_1 z_1 + e_2 z_2 + e_3 z_3\right),$$

  kuri naudojama naujo integravimo žingsnio prognozei. Be to, $err_1 \to 0$, kai $h\lambda \to \infty$.
- kiekviename žingsnyje paklaidos vektorius (err arba err_1) yra normuojamas pagal formulę:

```
for j=1:n
    d(j)=max([nd,abs(y(j)),abs(y1(j)));
end
lokerr=err./d';
errn=norm(lokerr/n);
```

Šis normavimas padeda tiksliau įvertinti paklaidą esant labai mažoms ir labai didelėms diferencialinių lygčių sistemos sprendinio vektoriaus komponentėms. Pavyzdžiui, žemiau pateiktiems pirmam ir antram pavyzdžiams daugiklis $nd=1$, tuo tarpu trečiam pavyzdžiui, - daugiklis $nd = 1e-4$.

Reikia dar kartą pabrėžti, kad realizuojant neišreikštinus Rungės ir Kutos metodus, skirtingai nei išreikštinus, kiekviename žingsnyje reikia spręsti netiesinių lygčių sistemą. Kaip buvo pažymėta anksčiau, jei duotam integravimo žingsniui Niutono metodas diverguoja, tai žingsnis mažinamas pusiau. Tam, kad žingsnis nebūtų per daug smulkinamas, nereikėtų imti labai mažos parametro *tol* reikšmės. Praktinė patirtis rodo, kad optimali *tol* reikšmė yra 10^{-2}. Kadangi parametras $\kappa = 10^{-2}$, tai netiesinių lygčių sprendimas baigiamas, kai Niutono metodo paklaida tampa mažesnė nei 10^{-4}. Toks sistemos sprendinys leidžia pasiekti gerą diferencialinių lygčių sistemos sprendinio tikslumą, kuris, kaip rodo praktika, paprastai priklauso $[10^{-4};10^{-6}]$ intervalui.

Žemiau pateikta matlabinė funkcija ***rado5***, realizuojanti aprašytą Rado metodą.

```
function [xspr,yspr]=rado5(ff,x,y,xend,tol,hmax,h,nd,stat,Jp,Jak)
% RADO5 sprendžia n standžių diferencialinių lygčių sistemą
% Rado IIA 5 eiles metodu.
% Lit. E.Hairer, G.Wanner, Solving Ordinary Differential
% equations II, Stiff and Differential-Algebraic Problems,
% Springer-Verlag, Berlin, Heidelberg, 1991.
%
% Įėjimo parametrai:
% ff   - funkcija, apskaičiuojanti sistemos dešiniąją pusę,
% x    - pradinė x-o reikšmė,
% y    - pradinių y-ko reikšmių vektorius,
% xend - paskutinė x-o reikšmė,
% tol  - lokaliosios paklaidos dydis,
% hmax - maksimalus leistinas integravimo žingsnis,
% h    - pradinis (spėjamas) integravimo žingsnis,
% nd   - paklaidą normuojančio daugiklio apatinė riba,
% stat - statistikos požymis. Jei stat=1,tai atspausdinama
%      informacija apie bendrą, priimtų ir atmestų žingsnių skaičių,
% Jp   - Jp=1, jei jakobiana apskaičiuojama tiksliai,
%        Jp=0, jei jakobiana apskaičiuojama skaitiniu būdu,
% Jak  - Jakobianos apskaičiavimo funkcija; naudojama, kai Jp=1.
% Išėjimo parametrai:
% xspr - sprendinio x-o reikšmių vektorius,
% yspr - sprendinio y-o reikšmių formato (n,numel(x)) matrica.

n=numel(y);
% Rado5 metodą nusakantys koeficientai
s=3;
c=[(4-sqrt(6))/10, (4+sqrt(6))/10, 1];
a1=[(88-7*sqrt(6))/360 (296-169*sqrt(6))/1800 (-2+3*sqrt(6))/225];
a2=[(296+169*sqrt(6))/1800 (88+7*sqrt(6))/360 (-2-3*sqrt(6))/225];
a3=[(16-sqrt(6))/36 (16+sqrt(6))/36 1/9];
a=[a1; a2; a3];
b=a3;
% Metodo paklaidą apibrėžiantys koeficientai
gama=0.27488882959568;
e=[-13-7*sqrt(6),-13+7*sqrt(6),1]*gama/3;
ematr=repmat(e,n,1); In=eye(n);
maxf=30000; %didžiausias leistinas sprendinio taškų skaičius
posneg=1;
if xend < x
    posneg=-1;
end;

% Pradiniai pasiruošimai
s=3; etaold=1; kmax=7;
p0=0; % p0=0, jei atliekamas metodo pirmasis žingsnis
yold=zeros(n,1); zmatr=zeros(n,s); hold=h;
[m1,n1]=size(y);
if m1 == 1
    y=y';
end;
hmax=abs(hmax); h=min(max(1e-4,abs(h)),hmax); h=posneg*h;
tol=max(tol,10*eps);
reject=false;
naccept=0; % priimtų žingsnių skaičius
nreject=0; % atmestų žingsnių skaičius
```

```
i=1;
nstep=0;   % bendras žingsnių skaičius
xspr(i)=x; yspr(:,i)=y;
% Pagrindiniai skaičiavimai
while (((x-xend)*posneg+eps) <= 0)&(i<=maxf)
    if x+0.1*h == x
        disp('programos darbas nutraukiamas');
        disp('per daug mažas integravimo žingsnis');
        return;
    end
    if (x+h-xend)*posneg >0
        h=xend-x;
    end
    nstep=nstep+1;
  % Taško (x(i+1);y(i+1)) apskaičiavimas
  % Jakobio matricos apskaičiavimas
if Jp==0 % Jakobio matrica apskaičiuojama skaitiniu būdu
    delta=1.0e-8;
    for j=1:n
        yd=y; yd(j)=yd(j)+delta;
        J(:,j)=(feval(ff,x,yd)-feval(ff,x,y))/delta;
    end
else    % Jakobio matrica apskaičiuojama tiksliai
    J=feval(Jak,x,y);
end
% Netiesinių lygčių sistemos sprendimas
p=false;
while ~p
    [z,p,eta,Newt]=sprniut(ff,c,a,x,y,h,tol,etaold,...
                           kmax,p0,zmatr,yold,hold,J);
    if ~p
        h=h/2;
    end %endif
end % endwhile
zmatr=reshape(z,n,s); % vektorius transformuojamas į matricą
y1=y+zmatr(:,s);% y(i+1) reikšmė
yold=y; hold=h; etaold=eta;
    % Lokaliosios paklaidos apskaičiavimas
Invmat=inv(In-h*gama*J);
err=Invmat*(gama*h*feval(ff,x,y)+sum((ematr.*zmatr)'));
if (p0==0) | (norm(err/n)>tol)
    p0=1;% antrame ir kituose metodo žingsniuose p0=1
    err=Invmat*(gama*h*feval(ff,x,y+err)+sum((ematr.*zmatr)'));
end

% Paklaidos normavimas
for j=1:n
    d(j)=max([nd,abs(y(j)),abs(y1(j))]);
end
lokerr=err./d';
errn=norm(lokerr/n);
% Naujo žingsnio hnew apskaičiavimas
    fac=0.9*(2*kmax+1)/(2*kmax+Newt);
    hnew=fac*h*((tol/errn)^0.25);
    if errn <= tol % žingsnis priimamas
        i=i+1; xspr(i)=x+h; yspr(:,i)=y1; naccept=naccept+1;
        x=x+h; y=y1;
```

```
        if abs(hnew) >= hmax
            hnew=posneg*hmax;
        end
        if reject
            hnew=posneg*min(abs(hnew),abs(h));
        else
            hnew=posneg*abs(hnew);
        end
        reject=false;
    else % žingsnis atmetamas
        reject=true;
        nreject=nreject+1;
    end
    h=hnew;
end
if i>maxf
     disp('sprendinio taškų skaičius viršijo leistiną ribą')
end
if  stat == 1
    disp(['apskaičiuotų žingsnių skaičius =',num2str(nstep)])
    disp(['priimtų žingsnių skaičius =',num2str(naccept)])
    disp(['atmestų žingsnių skaičius =',num2str(nreject)])
end

function [z,p,eta,Newt]=sprniut(ff,c,a,x,y,h,tol,etaold,...
                                kmax,p0,zmatr,yold,hold,J);
%SPRNIUT realizuoja supaprastintą Niutono metodą, kuriuo sprendžiama
%netiesinių lygčių sistemą.
%  Įėjimo parametrai
% ff      - funkcija, apskaičiuojanti sistemos dešiniąją pusę,
% c, a    - Rado5 metodą nusakantys koeficientai,
% x, y    - paskutinio žingsnio sprendinio taškas,
% h       - dabartinio momento integravimo žingsnio ilgis,
% tol     - sprendinio lokaliosios paklaidos dydis,
% etaold  - paskutinio žingsnio konvergavimo greičio konstanta,
% kmax    - maksimalus Niutono iteracijų skaičius,
% p0      - jei p0=0, tai z=0; priešingu atveju z apskaičiuojamas,
% zmatr   - paskutinio žingsnio sistemos sprendinys,
% yold    - priešpaskutinio žingsnio y-o reikšmė,
% hold    - paskutinio žingsnio h reikšmė,
% J       - funkcija, apskaičiuojanti Jakobio matricą.
%  Išėjimo parametrai
% z       - netiesinių lygčių sistemos sprendinys,
% p       - p=true, jei rastas norimo tikslumo sprendinys,
%           p=false, priešingu atveju,
% eta     - nagrinėjamo žingsnio konvergavimo greičio konstanta,
% Newt    - atliktų Niutono iteracijų skaičius.

n=numel(y);
s=3; ns=n*s; it=0; p=true;
Ins=eye(ns); In=eye(n); c0=[0,c];
kapa=1e-2; eta=(max(etaold,eps))^0.8;
% Pradinio sprendinio apskaiciavimas
if p0 == 0
    z=zeros(ns,1);% pradinis sprendinys - nuliai
else % Pradinis sprendinys - apskaičiuojamas
    w=h/hold;
```

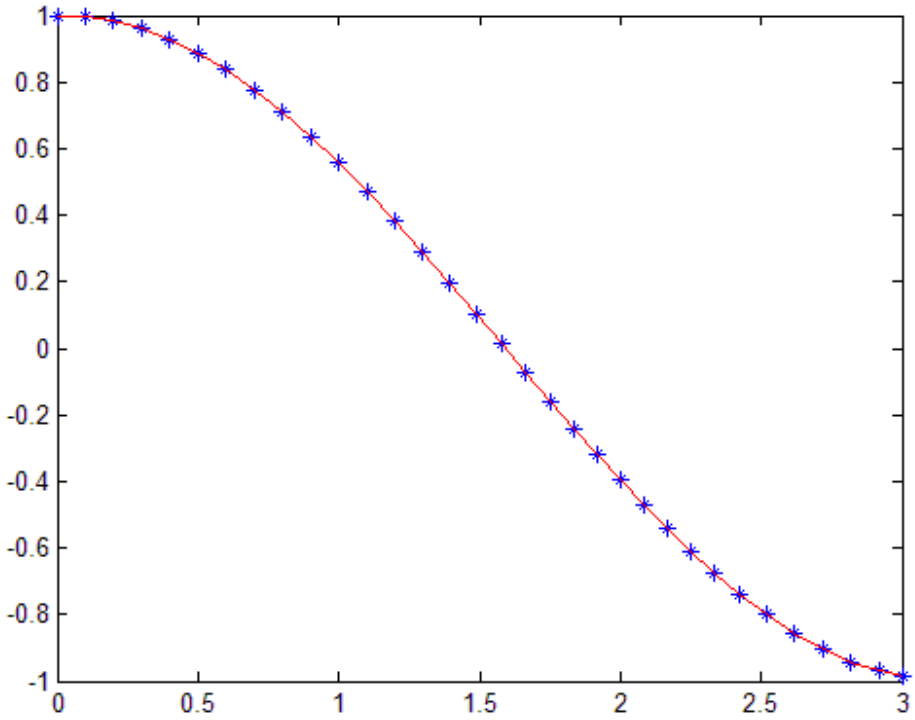
```
    for i=1:n
        fv(i,:)=aitken(c0,[0,zmatr(i,:)],1+w*c)+(yold(i)-y(i));
    end
    z=fv(:);
end
%  Tiesines lygčių sistemos formavimas
pkl=1+kapa*tol; dz1=zeros(ns,1);
A=Ins-h*kron(a,J); % Sistemos matrica
while p & (it <= kmax) & (pkl >= kapa*tol)
    it=it+1;
    for i=1:s
        f(:,i)=feval(ff,x+c(i)*h,y+z((i-1)*n+1:i*n));
    end
    F=f(:); % matrica transformuojama į vektorių
    b=h*kron(a,In)*F-z; % Sistemos laisvųjų narių stulpelis
% Tiesines lygčių sistemos sprendimas
    dz=A\b;
    normdz=norm(dz); z=z+dz;
    if it > 1
        teta=normdz/normdz1;
        eta=teta/(1-teta);
        if (teta >1) | (teta^(kmax-it)*normdz/(1-teta) > (kapa*tol))
            p=false;
        end
    end
    pkl=eta*normdz;% Sprendinio paklaida po it iteracijų
    dz1=dz; normdz1=normdz;
end;
Newt=it;
if it > kmax
    p=false;
end
```

Pastaba. Funkcijos ***sprniut*** tekste yra kreipinys į funkciją ***aitken***. Ši funkcija (žr.5.1.1.2 paragrafą) apskaičiuoja interpoliacinio polinomo reikšmes pagal Aitkeno schemą .

1-as pavyzdys. *Raskime diferencialinės lygties* $y' = -k(y - \cos x)$, *čia k>>0 pavyzdžiui, k=50, sprendinį, kai* $x_0 = 0$, $y_0 = 2500/2501$, *o xend=3.*

Kadangi ši lygtis yra tiesinė, tai galime rasti tikslų lygties sprendinį, kuris yra $y(x) = \frac{k}{k^2+1}(\sin x + k\cos x) + Ce^{-kx}$. Įvertinus pradines sąlygas ir tai, kad *k=50,* turėsime: $y(x) = (50\sin x + 2500\cos x)/2501$. Kitaip tariant, $y(x) \approx \cos x$.

9.6 paveiksle pavaizduotas lygties skaitinis sprendinys (mėlynos žvaigždutės), kurį apskaičiuoja funkcija *rado5*, ir tikslus sprendinys (raudonas grafikas).



9.6 pav. Lygties $y' = -k(y - \cos x)$, *k=50, sprendinys.*

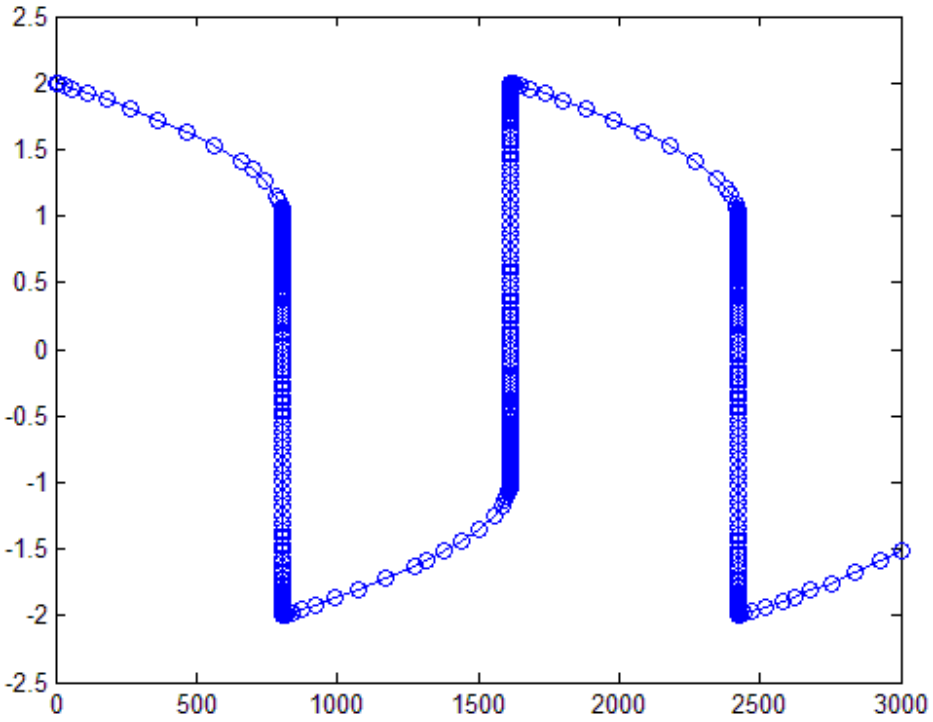
2-as pavyzdys. *Raskime Van-der-Polio lygties*[*] $y''+\varepsilon\left(y^2-1\right)y'+y=0$, $\varepsilon>0$ *skaitinį sprendinį. Šią antros eilės diferencialinę lygtį galima pakeisti lygčių sistema:*

$$y_1' = y_2,$$
$$y_2' = \varepsilon\left(1-y_1^2\right)y_2 - y_1.$$

Apskaičiuokime sistemos sprendinį, kai $\varepsilon = 1000$, $x \in [0;3000]$, *t.y.,* $x_0 = 0$, $x_{end} = 3000$ *ir* $y_0 = (2;0)^t$.

9.7 paveiksle pavaizduotas Van-der –Polio lygties sprendinys, apskaičiuotas taikant funkciją ***rado5*** ir imant *h*=0.01; *hmax*=100; *tol*=1e-2.

- Aš turiu teoriją: jeigu jus norite sukompromituoti kokį nors metodą,- ieškokite Van-der –Polio lygties sprendinio (P. E. Zadunaiskis) [30]



9.7 pav. Van-der –Polio lygties sprendinys, kai $\varepsilon = 1000$.

Pastaba. Kaip sudaryti Jakobio matricos funkciją, galima spręsti iš funkcijos, apskaičiuojančios Van-der –Polio sistemos Jakobio matricą.

```
function J=Jak(x,y);
% JAK apskaičiuoja dif. lygčių sistemos Jakobio matricą

J(1,1)=0; J(1,2)=1;
J(2,1)=-1000*2*y(1)*y(2)-1; J(2,2)=1000*(1-y(1)^2);
```

3-ias pavyzdys *(Cheminės kinetikos lygtis* [*30,31*]***).*** *Žemiau pateikta diferencialinių lygčių sistema aprašo chemines reakcijas, iš kurių pirmoji yra lėta, antroji – labai greita, o trečioji – greita.*

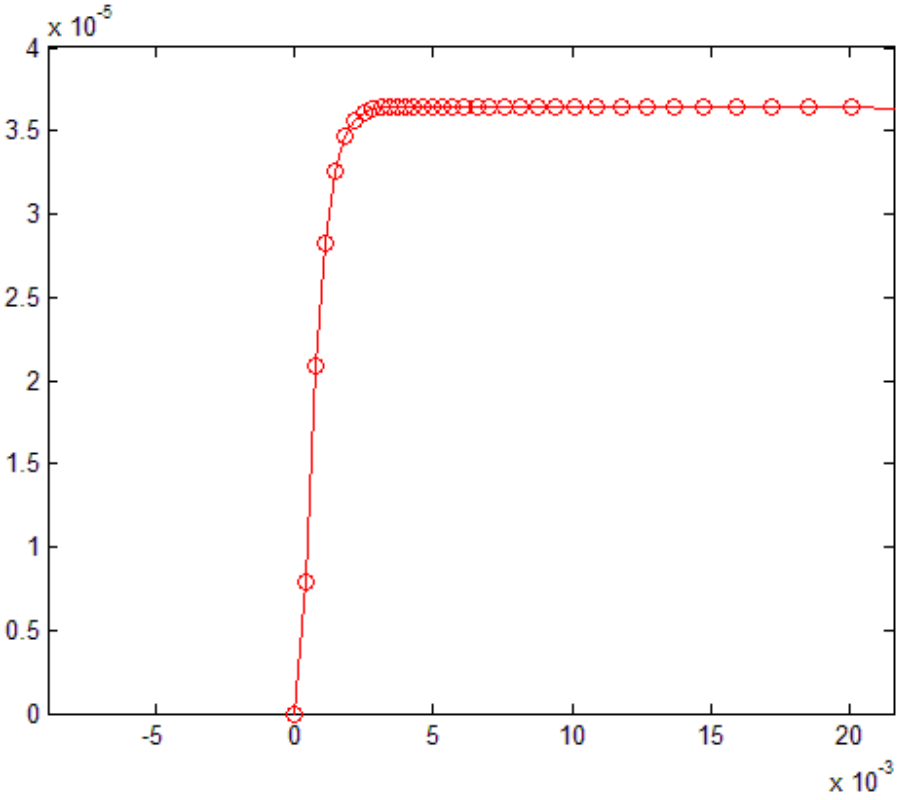
$$y_1' = -0.04 y_1 + 10^4 y_2 y_3,$$

$$y_2' = 0.04 y_1 - 10^4 y_2 y_3 - 3*10^7 y_2^2, \quad y_1(0) = 1, y_2(0) = 0, y_3(0) = 0, t \in [0;3].$$

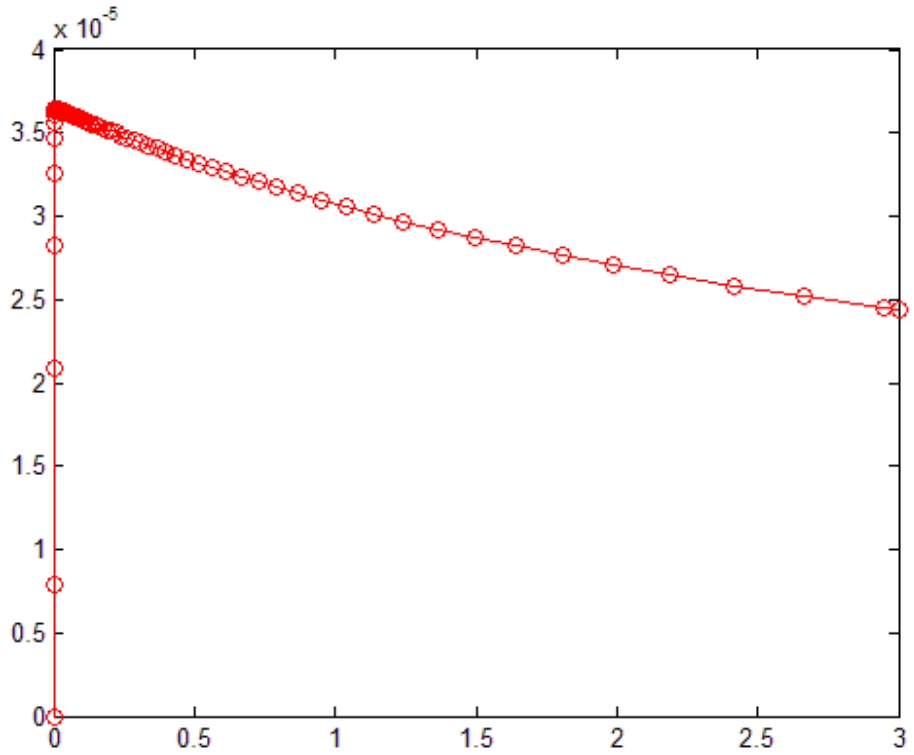
$$y_3' = 3*10^7 y_2^2,$$

Žemiau pateikti sistemos sprendimo rezultatai, gauti ją sprendžiant su funkcija ***rado5***.

9.8 paveiksle parodytas funkcijos $y_2(t)$ grafiko pradinis fragmentas, 9.9 paveiksle – visas funkcijos $y_2(t)$ grafikas. Matome, kad $y_2(t)$ labai greitai pasiekia kvazi-stacionarią poziciją: $y_2(t) = 3.6486\text{e-}5$, o po to $y_2(t)$ lėtai artėja prie nulio.



9.8 pav. Funkcijos $y_2(t)$ grafiko pradinis fragmentas



9.9 pav. Funkcijos $y_2(t)$ grafikas

Neišreikštinių Rungės ir Kutos metodų privalumai yra jų stabilumas ir aukšta aproksimuoto sprendinio tikslumo eilė, esant mažam stadijų skaičiui. Tačiau pagrindinis jų trūkumas – daugybė skaičiavimų kiekviename metodo žingsnyje. Apie tai kalbėjome šiame paragrafe aptardami supaprastintą Niutono metodą. Tam, kad būtų sumažinta skaičiavimų apimtis, yra pasiūlyta daug neišreikštinių Rungės ir Kutos metodo modifikacijų. Populiariausia modifikacija yra ***įstrižaininis neišreištinis Rungės ir Kutos metodas*** (angl. *Singly diagonally implicit Runge-Kuta* sutrumpintai SDIRK). Šį metodą nusakančios lentelės (žr. 9.10 lentelę) visi koeficientai virš pagrindinės įstrižainės yra lygūs nuliui. Tada, realizuodami ***s*** stadijų įstrižainį neišreištinį metodą, kiekviename jo žingsnyje spręsime ***s*** netiesinių lygčių arba jų sistemų. Skaičiavimų apimtį dar labiau sumažina kita Rungės ir Kutos metodo modifikacija, kuri vadinama Rozenbroko metodu. Šių modifikacijų čia nenagrinėsime, juolab, kad šie metodai detaliai išnagrinėti [30,31] literatūroje.

9.3.3 MATLAB‘o funkcijos diferencialinėms lygtims spręsti

MATLAB‘as turi eilę funkcijų, skirtų spręsti pradinių reikšmių diferencialinų lygčių uždavinį. Tai funkcijos: ***ode45,ode23,ode113, ode15s, ode23s, ode23t ir ode23tb.***

Funkcija ***ode45*** realizuoja 5(4) išreikštinį Rungės ir Kutos metodą (Dormando-Prinso (Dormand-Prince)porą).

Funkcija ***ode23*** realizuoja 3(2) išreikštinį Rungės ir Kutos metodą (Bogackio-Shampini (Bogacki-Shampine) porą).

Funkcija ***ode113*** realizuoja kintamos eilės Adamso-Bašforto-Moultono metodą.

Šios funkcijos skirtos spręsti nestandžias diferencialines lygtis ir tiksliausios iš jų yra ***ode113*** ir ***ode45*** funkcijos

Funkcijos ***ode15s ir ode23s*** realizuoja atitinkamai ekstrapoliacinį ir Rozenbroko metodus, o funkcijos ***ode23t ir ode23tb*** - atitinkamai trapecijų ir modifikuotą trapecijų metodus. Visos šios funkcijos yra skirtos spręsti standžias diferencialines lygtis.

Funkcija ***ode23t*** skirta spręsti vidutinio standumo lygtis. Kitoms funkcijoms standumo laipsnis neribojamas.

Į diferencialinių lygčių sprendimo funkcijas galima kreiptis keliais būdais. Dažniausiai naudojami du kreipiniai:

[*x,y*]=*solver*(*odefun*,[*x0, xend*],*y0*),

[*x,y*]=*solver*(*odefun*,[*x0, xend*],*y0,options*),

čia „*solver*“ – viena iš funkcijų:*ode45,ode23,ode113, ode15s, ode23s, ode23t ode23tb*, *odefun,*– vardas funkcijos, apskaičiuojančios diferencialinių lygčių sistemos dešinę pusę (vektorių – stulpelį), [*x0, xend*], – lygties sprendinio intervalas, *y0,*– pradinių sąlygų vektorius (stulpelis). Parametras „*options*“ yra specialios struktūros masyvas, sukuriamas funkcijos „***odeset***“ pagalba, kuriame galima nurodyti santykinę, absoliutinę paklaidas, paklaidos normavimą ir kitokius parametrus. Kreipinys į *odeset* užrašomas taip:

```
options = odeset('vardas1',reikšmė1,'vardas2',reikšmė2,...).
```

Pavyzdžiui, spręsdami trijų diferencialinių lygčių sistemą ir norėdami nurodyti santykinę ir kiekvienos komponentės absoliutinę paklaidą, į funkciją *odeset* kreipsimės taip: `options = odeset('RelTol',1e-4,'AbsTol',[1e-4,1e-4,1e-5]),` čia R*elTol* yra santykinės paklaidos vardas, o A*bsTol* - atskirų sprendinio komponenčių absoliutinių paklaidų masyvo vardas. Plačiau apie funkciją *odeset* galima sužinoti MATLAB‘o komandų lauke surinkus komandą: *help odeset*.

Pavyzdys. *Išspręskime sistemą trijų diferencialinių lygčių, aprašančių kieto kūno, kurio neveikia išorinės jėgos, judėjimą.*

Kūno judesį aprašo tokia diferencialinių lygčių sistema:

$y_1' = y_2 y_3,$

$y_2' = -y_1 y_3, \quad , y_1(0) = 0, y_2(0) = 1, y_3(0) = 1, t \in [0;12].$

$y_3' = -0.51 y_1 y_2.$

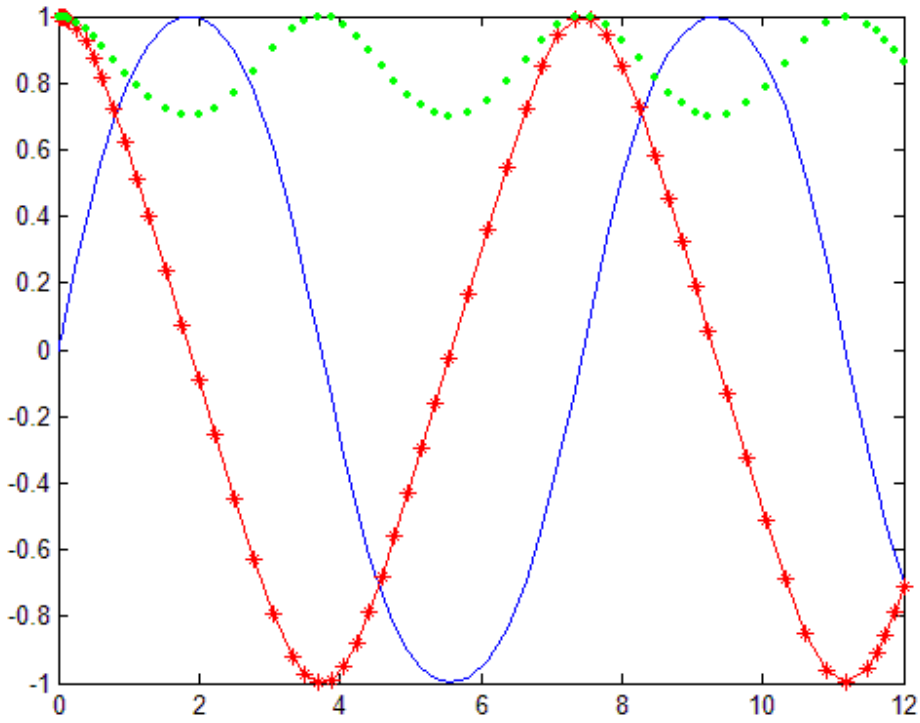
Sudarę funkciją „***kunas***“, kuri apskaičiuoja diferencialinių lygčių sistemos dešinę pusę:

```
function dy=kunas(t,y);
% KUNAS - funkcija, apskaičiuojanti diferencialinių lygčių sistemos
%           dešiniąją pusę.
n=numel(y);
dy=zeros(n,1); % vektorius - stulpelis
dy(1)=y(2)*y(3);
dy(2)=-y(1)*y(3);
dy(3)=-0.51*y(1)*y(2);
```

, o taip pat pagrindinę programą:

```
options=odeset('RelTol',1e-4,'AbsTol',[1e-4,1e-4,1e-5]);
[t,y]=ode45('kunas',[0,12],[0;1;1]);
plot(t,y(:,1),'-',t,y(:,2),'r-*',t,y(:,3),'g.'),
```

turėsime sprendinio grafikus (žr. 9.10 paveikslą).



9.10 pav. Kieto kūno judėjimą nusakantis sprendinys: y_1 - mėlynas grafikas, y_2 - raudonas grafikas, y_3 - žalias grafikas.