Netiesinio modeliavimo metodai. Antroji pratybų užduotis

Netiesiniai evoliuciniai modeliai ir jų kompiuterinio sprendimo algoritmai. Tirsime erdvėje (erdvės kintamąjį žymime x) pasklidusio darinio (pavyzdžiui, bangos) judesį (evoliuciją), priklausomai nuo laiko kintamojo t. Tokį erdvinį-laikinį darinį galime modeliuoti kompiuteriu, jei žinome koks yra šio darinio matematinis modelis bei turime algoritmą duotam modeliui išspręsti. Papildomos literatūros sąraše, esančiame šio aprašymo pabaigoje, rasite nuorodas kur galima paskaityti apie taikomąją kai kurių iš nagrinėjamų modelių prasmę.

Tarkime, kad a, c, d, k, α, β, γ, θ yra duotos realios konstantos (programoje numatyti galimybę šiuos parametrus laisvai keisti), o $f = f(x,t)$ duota funkcija. Evoliucinio modelio sprendinys, kurio artinį reikia surasti žymimas $u = u(x,t)$.

Funkcijos $f(x,t) = \operatorname{Re} f(x,t) + i \operatorname{Im} f(x,t)$ ir $u(x,t) = \operatorname{Re} u(x,t) + i \operatorname{Im} u(x,t)$ įgyja kompleksines reikšmes, t. y. turi realiąją ir menamąją dalis. Čia $i = \sqrt{-1}$ yra menamasis kompleksinis vienetas. Taigi programuodami algoritmą naudokite kompleksinių skaičių aritmetiką.

Pasirinkite Jums paskirtą (pagal atitinkamą užduoties numerį, žr. užduočių skyrelį šio aprašymo pabaigoje) netiesinį evoliucinį modelį (lygtį) bei jo algoritmą (pagal kurį ir turėsite parašyti kompiuterio programą).

A Burgers'o lygtis

$$\frac{\partial u}{\partial t} = a^2 \frac{\partial^2 u}{\partial x^2} + u \frac{\partial u}{\partial x} + f(x,t), \qquad 0 < x < 1, \qquad t > 0,$$

A1 Algoritmas

$$\frac{\hat{u}_j - u_j}{\tau} = a^2 \frac{1}{2} \left(\frac{\hat{u}_{j+1} - 2\hat{u}_j + \hat{u}_{j-1}}{h^2} + \frac{u_{j+1} - 2u_j + u_{j-1}}{h^2} \right) +$$

$$\frac{\hat{u}_j + u_j}{2} \frac{1}{2} \left(\frac{\hat{u}_{j+1} - \hat{u}_{j-1}}{2h} + \frac{u_{j+1} - u_{j-1}}{2h} \right) + \frac{\hat{f}_j + f_j}{2}, \qquad j = 1, 2, \ldots, N-1,$$

A2 Algoritmas

$$\frac{\hat{u}_j - u_j}{\tau} = a^2 \frac{1}{2} \left(\frac{\hat{u}_{j+1} - 2\hat{u}_j + \hat{u}_{j-1}}{h^2} + \frac{u_{j+1} - 2u_j + u_{j-1}}{h^2} \right) +$$

$$\frac{1}{2} \left(\hat{u}_j \frac{\hat{u}_{j+1} - \hat{u}_{j-1}}{2h} + u_j \frac{u_{j+1} - u_{j-1}}{2h} \right) + \frac{\hat{f}_j + f_j}{2}, \qquad j = 1, 2, \ldots, N-1,$$

A3 Algoritmas

$$\frac{\hat{u}_j - u_j}{\tau} = a^2 \frac{1}{2} \left(\frac{\hat{u}_{j+1} - 2\hat{u}_j + \hat{u}_{j-1}}{h^2} + \frac{u_{j+1} - 2u_j + u_{j-1}}{h^2} \right) +$$

$$\frac{1}{2} \left[\frac{\hat{u}_j + u_j}{2} \frac{1}{2} \left(\frac{\hat{u}_{j+1} - \hat{u}_{j-1}}{2h} + \frac{u_{j+1} - u_{j-1}}{2h} \right) + \frac{1}{2} \left(\hat{u}_j \frac{\hat{u}_{j+1} - \hat{u}_{j-1}}{2h} + u_j \frac{u_{j+1} - u_{j-1}}{2h} \right) \right] +$$

$$\frac{\hat{f}_j + f_j}{2}, \qquad j = 1, 2, \ldots, N-1.$$

B Netiesinė Šriodingerio lygtis

$$\frac{\partial u}{\partial t} = i\frac{\partial^2 u}{\partial x^2} + icu + id|u|^2 u + f(x,t), \qquad 0 < x < 1, \qquad t > 0,$$

B1 Algoritmas

$$\frac{\hat{u}_j - u_j}{\tau} = i\frac{1}{2}\left(\frac{\hat{u}_{j+1} - 2\hat{u}_j + \hat{u}_{j-1}}{h^2} + \frac{u_{j+1} - 2u_j + u_{j-1}}{h^2}\right) +$$

$$ic\frac{\hat{u}_j + u_j}{2} + id\left|\frac{\hat{u}_j + u_j}{2}\right|^2 \frac{\hat{u}_j + u_j}{2} + \frac{\hat{f}_j + f_j}{2}, \qquad j = 1, 2, \ldots, N-1,$$

B2 Algoritmas

$$\frac{\hat{u}_j - u_j}{\tau} = i\frac{1}{2}\left(\frac{\hat{u}_{j+1} - 2\hat{u}_j + \hat{u}_{j-1}}{h^2} + \frac{u_{j+1} - 2u_j + u_{j-1}}{h^2}\right) +$$

$$ic\frac{\hat{u}_j + u_j}{2} + id\frac{|\hat{u}_j|^2\hat{u}_j + |u_j|^2 u_j}{2} + \frac{\hat{f}_j + f_j}{2}, \qquad j = 1, 2, \ldots, N-1,$$

B3 Algoritmas

$$\frac{\hat{u}_j - u_j}{\tau} = i\frac{1}{2}\left(\frac{\hat{u}_{j+1} - 2\hat{u}_j + \hat{u}_{j-1}}{h^2} + \frac{u_{j+1} - 2u_j + u_{j-1}}{h^2}\right) +$$

$$ic\frac{\hat{u}_j + u_j}{2} + id\frac{|\hat{u}_j|^2 + |u_j|^2}{2}\frac{\hat{u}_j + u_j}{2} + \frac{\hat{f}_j + f_j}{2}, \qquad j = 1, 2, \ldots, N-1.$$

C Netiesinė išvestinių atžvilgiu Šriodingerio lygtis

$$\frac{\partial u}{\partial t} = i\frac{\partial^2 u}{\partial x^2} + \alpha|u|^2\frac{\partial u}{\partial x} + f(x,t), \qquad 0 < x < 1, \qquad t > 0,$$

C1 Algoritmas

$$\frac{\hat{u}_j - u_j}{\tau} = i\frac{1}{2}\left(\frac{\hat{u}_{j+1} - 2\hat{u}_j + \hat{u}_{j-1}}{h^2} + \frac{u_{j+1} - 2u_j + u_{j-1}}{h^2}\right) +$$

$$\alpha\left|\frac{\hat{u}_j + u_j}{2}\right|^2 \frac{1}{2}\left(\frac{\hat{u}_{j+1} - \hat{u}_{j-1}}{2h} + \frac{u_{j+1} - u_{j-1}}{2h}\right) + \frac{\hat{f}_j + f_j}{2}, \qquad j = 1, 2, \ldots, N-1,$$

C2 Algoritmas

$$\frac{\hat{u}_j - u_j}{\tau} = i\frac{1}{2}\left(\frac{\hat{u}_{j+1} - 2\hat{u}_j + \hat{u}_{j-1}}{h^2} + \frac{u_{j+1} - 2u_j + u_{j-1}}{h^2}\right) +$$

$$\alpha\frac{1}{2}\left(|\hat{u}_j|^2\frac{\hat{u}_{j+1} - \hat{u}_{j-1}}{2h} + |u_j|^2\frac{u_{j+1} - u_{j-1}}{2h}\right) + \frac{\hat{f}_j + f_j}{2}, \qquad j = 1, 2, \ldots, N-1,$$

C3 Algoritmas

$$\frac{\hat{u}_j - u_j}{\tau} = i\frac{1}{2}\left(\frac{\hat{u}_{j+1} - 2\hat{u}_j + \hat{u}_{j-1}}{h^2} + \frac{u_{j+1} - 2u_j + u_{j-1}}{h^2}\right) +$$

$$\alpha\frac{|\hat{u}_j|^2 + |u_j|^2}{2}\frac{1}{2}\left(\frac{\hat{u}_{j+1} - \hat{u}_{j-1}}{2h} + \frac{u_{j+1} - u_{j-1}}{2h}\right) + \frac{\hat{f}_j + f_j}{2}, \qquad j = 1, 2, \ldots, N-1.$$

D Netiesinė išvestinių atžvilgiu Šriodingerio lygtis

$$\frac{\partial u}{\partial t} = i\frac{\partial^2 u}{\partial x^2} + \beta\frac{\partial}{\partial x}\left(|u|^2 u\right) + f(x,t), \qquad 0 < x < 1, \qquad t > 0,$$

D1 Algoritmas

$$\frac{\hat{u}_j - u_j}{\tau} = i\frac{1}{2}\left(\frac{\hat{u}_{j+1} - 2\hat{u}_j + \hat{u}_{j-1}}{h^2} + \frac{u_{j+1} - 2u_j + u_{j-1}}{h^2}\right) +$$

$$\beta\frac{\left|\frac{\hat{u}_{j+1} + u_{j+1}}{2}\right|^2 \frac{\hat{u}_{j+1} + u_{j+1}}{2} - \left|\frac{\hat{u}_{j-1} + u_{j-1}}{2}\right|^2 \frac{\hat{u}_{j-1} + u_{j-1}}{2}}{2h} +$$

$$\frac{\hat{f}_j + f_j}{2}, \qquad j = 1, 2, \ldots, N-1,$$

D2 Algoritmas

$$\frac{\hat{u}_j - u_j}{\tau} = i\frac{1}{2}\left(\frac{\hat{u}_{j+1} - 2\hat{u}_j + \hat{u}_{j-1}}{h^2} + \frac{u_{j+1} - 2u_j + u_{j-1}}{h^2}\right) +$$

$$\beta\frac{1}{2}\left(\frac{|\hat{u}_{j+1}|^2\hat{u}_{j+1} - |\hat{u}_{j-1}|^2\hat{u}_{j-1}}{2h} + \frac{|u_{j+1}|^2 u_{j+1} - |u_{j-1}|^2 u_{j-1}}{2h}\right) + \frac{\hat{f}_j + f_j}{2},$$

$$j = 1, 2, \ldots, N-1.$$

E Netiesinė Kuramoto-Cuzuki lygtis

$$\frac{\partial u}{\partial t} = (a^2 + i)\frac{\partial^2 u}{\partial x^2} + i\gamma\ln(1 + |u|^2) + f(x,t), \qquad 0 < x < 1, \qquad t > 0,$$

E1 Algoritmas

$$\frac{\hat{u}_j - u_j}{\tau} = (a^2 + i)\frac{1}{2}\left(\frac{\hat{u}_{j+1} - 2\hat{u}_j + \hat{u}_{j-1}}{h^2} + \frac{u_{j+1} - 2u_j + u_{j-1}}{h^2}\right) +$$

$$i\gamma\ln(1 + \left|\frac{\hat{u}_j + u_j}{2}\right|^2) + \frac{\hat{f}_j + f_j}{2}, \qquad j = 1, 2, \ldots, N-1,$$

E2 Algoritmas

$$\frac{\hat{u}_j - u_j}{\tau} = (a^2+i)\frac{1}{2}\left(\frac{\hat{u}_{j+1} - 2\hat{u}_j + \hat{u}_{j-1}}{h^2} + \frac{u_{j+1} - 2u_j + u_{j-1}}{h^2}\right) +$$

$$i\gamma\frac{\ln(1+|\hat{u}_j|^2) + \ln(1+|u_j|^2)}{2} + \frac{\hat{f}_j + f_j}{2}, \qquad j = 1, 2, \ldots, N-1.$$

F Netiesinė išvestinių atžvilgiu Kuramoto-Cuzuki lygtis

$$\frac{\partial u}{\partial t} = (a^2+i)\frac{\partial^2 u}{\partial x^2} + \theta u\frac{\partial |u|^2}{\partial x} + f(x,t), \qquad 0 < x < 1, \qquad t > 0,$$

F1 Algoritmas

$$\frac{\hat{u}_j - u_j}{\tau} = (a^2+i)\frac{1}{2}\left(\frac{\hat{u}_{j+1} - 2\hat{u}_j + \hat{u}_{j-1}}{h^2} + \frac{u_{j+1} - 2u_j + u_{j-1}}{h^2}\right) +$$

$$\theta\frac{\hat{u}_j + u_j}{2}\frac{\left|\frac{\hat{u}_{j+1} + u_{j+1}}{2}\right|^2 - \left|\frac{\hat{u}_{j-1} + u_{j-1}}{2}\right|^2}{2h} + \frac{\hat{f}_j + f_j}{2}, \qquad j = 1, 2, \ldots, N-1,$$

F2 Algoritmas

$$\frac{\hat{u}_j - u_j}{\tau} = (a^2+i)\frac{1}{2}\left(\frac{\hat{u}_{j+1} - 2\hat{u}_j + \hat{u}_{j-1}}{h^2} + \frac{u_{j+1} - 2u_j + u_{j-1}}{h^2}\right) +$$

$$\theta\frac{1}{2}\left(\hat{u}_j\frac{|\hat{u}_{j+1}|^2 - |\hat{u}_{j-1}|^2}{2h} + u_j\frac{|u_{j+1}|^2 - |u_{j-1}|^2}{2h}\right) + \frac{\hat{f}_j + f_j}{2}, \qquad j = 1, 2, \ldots, N-1,$$

F3 Algoritmas

$$\frac{\hat{u}_j - u_j}{\tau} = (a^2+i)\frac{1}{2}\left(\frac{\hat{u}_{j+1} - 2\hat{u}_j + \hat{u}_{j-1}}{h^2} + \frac{u_{j+1} - 2u_j + u_{j-1}}{h^2}\right) +$$

$$\theta\frac{1}{2}\left[\frac{\hat{u}_j + u_j}{2}\frac{\left|\frac{\hat{u}_{j+1} + u_{j+1}}{2}\right|^2 - \left|\frac{\hat{u}_{j-1} + u_{j-1}}{2}\right|^2}{2h} + \frac{1}{2}\left(\hat{u}_j\frac{|\hat{u}_{j+1}|^2 - |\hat{u}_{j-1}|^2}{2h} + u_j\frac{|u_{j+1}|^2 - |u_{j-1}|^2}{2h}\right)\right] +$$

$$\frac{\hat{f}_j + f_j}{2}, \qquad j = 1, 2, \ldots, N-1.$$

G Netiesinė Kuramoto-Cuzuki lygtis

$$\frac{\partial u}{\partial t} = (a^2+i)\frac{\partial^2 u}{\partial x^2} + icu + id|u|^2 u + f(x,t), \qquad 0 < x < 1, \qquad t > 0,$$

G1 Algoritmas

$$\frac{\hat{u}_j - u_j}{\tau} = (a^2+i)\frac{1}{2}\left(\frac{\hat{u}_{j+1} - 2\hat{u}_j + \hat{u}_{j-1}}{h^2} + \frac{u_{j+1} - 2u_j + u_{j-1}}{h^2}\right) +$$

$$ic\frac{\hat{u}_j + u_j}{2} + id\left|\frac{\hat{u}_j + u_j}{2}\right|^2 \frac{\hat{u}_j + u_j}{2} + \frac{\hat{f}_j + f_j}{2}, \qquad j = 1, 2, \ldots, N-1,$$

G2 Algoritmas

$$\frac{\hat{u}_j - u_j}{\tau} = (a^2+i)\frac{1}{2}\left(\frac{\hat{u}_{j+1} - 2\hat{u}_j + \hat{u}_{j-1}}{h^2} + \frac{u_{j+1} - 2u_j + u_{j-1}}{h^2}\right) +$$

$$ic\frac{\hat{u}_j + u_j}{2} + id\frac{|\hat{u}_j|^2\hat{u}_j + |u_j|^2 u_j}{2} + \frac{\hat{f}_j + f_j}{2}, \qquad j = 1, 2, \ldots, N-1,$$

H Netiesinė išvestinių atžvilgiu Kuramoto-Cuzuki lygtis

$$\frac{\partial u}{\partial t} = (a^2+i)\frac{\partial^2 u}{\partial x^2} + \beta\frac{\partial}{\partial x}\left(|u|^2 u\right) + f(x,t), \qquad 0 < x < 1, \qquad t > 0,$$

H1 Algoritmas

$$\frac{\hat{u}_j - u_j}{\tau} = (a^2+i)\frac{1}{2}\left(\frac{\hat{u}_{j+1} - 2\hat{u}_j + \hat{u}_{j-1}}{h^2} + \frac{u_{j+1} - 2u_j + u_{j-1}}{h^2}\right) +$$

$$\beta\frac{\left|\frac{\hat{u}_{j+1} + u_{j+1}}{2}\right|^2 \frac{\hat{u}_{j+1} + u_{j+1}}{2} - \left|\frac{\hat{u}_{j-1} + u_{j-1}}{2}\right|^2 \frac{\hat{u}_{j-1} + u_{j-1}}{2}}{2h} +$$

$$\frac{\hat{f}_j + f_j}{2}, \qquad j = 1, 2, \ldots, N-1,$$

H2 Algoritmas

$$\frac{\hat{u}_j - u_j}{\tau} = (a^2+i)\frac{1}{2}\left(\frac{\hat{u}_{j+1} - 2\hat{u}_j + \hat{u}_{j-1}}{h^2} + \frac{u_{j+1} - 2u_j + u_{j-1}}{h^2}\right) +$$

$$\beta\frac{1}{2}\left(\frac{|\hat{u}_{j+1}|^2\hat{u}_{j+1} - |\hat{u}_{j-1}|^2\hat{u}_{j-1}}{2h} + \frac{|u_{j+1}|^2 u_{j+1} - |u_{j-1}|^2 u_{j-1}}{2h}\right) + \frac{\hat{f}_j + f_j}{2},$$

$$j = 1, 2, \ldots, N-1.$$

Kraštinės sąlygos. Taip pat atkreipkite dėmesį kokias kraštines sąlygas privalo tenkinti Jūsų modelio sprendinys.

I Nulinės kraštinės sąlygos (angliškoje literatūroje dar vadinamos “Dirichlet boundary conditions”)

$$u(0,t) = u(1,t) = 0, \qquad t > 0.$$

Jų aproksimacija algoritme

$$\hat{u}_0 = 0, \qquad \hat{u}_N = 0.$$

II Nulinių išvestinių kraštinės sąlygos (angliškoje literatūroje dar vadinamos "Neumann boundary conditions")

$$\left.\frac{\partial u}{\partial x}\right|_{x=0} = \left.\frac{\partial u}{\partial x}\right|_{x=1} = 0, \qquad t > 0.$$

Jų aproksimacija algoritme

$$\hat{u}_0 = \hat{u}_1, \qquad \hat{u}_N = \hat{u}_{N-1}.$$

Pradinė sąlyga. Jos prireiks norint apibrėžti pradinį (kai laikas $t = 0$) sprendinio reikšmių masyvą. Pradinė sąlyga visoms užduotims vienoda:

$$u(x,0) = u^{(0)}(x), \qquad 0 \le x \le 1,$$

čia $u^{(0)}(x)$ yra duota kompleksines reikšmes įgyjanti funkcija.

Svarbu tai, kad pradinė funkcija $u^{(0)}(x)$ privalo tenkinti Jums nurodytas kraštines sąlygas intervalo galuose $x = 0$ ir $x = 1$.

Kompiuterinio modeliavimo algoritmas. Programuojant visos funkcijos pakeičiamos jų reikšmių masyvais. Kadangi šios reikšmės yra kompleksiniai skaičiai, tai dirbsime su kompleksinių skaičių masyvais. Pavyzdžiui, vietoje matematinio objekto – funkcijos $u(x,t)$ operuosime jos reikšmėmis

$$u_j = u(jh, t), \qquad j = 0, 1, \ldots, N, \qquad t = 0, \tau, 2\tau, \ldots,$$

kur $h = 1/N$ yra diskretizacijos žingsnis pagal erdvės kintamąjį x, o $\tau > 0$ yra diskretizacijos žingsnis pagal laiko kintamąjį t.

Taip pat naudosime pažymėjimą

$$\hat{u}_j = u(jh, t + \tau), \qquad j = 0, 1, \ldots, N, \qquad t = 0, \tau, 2\tau, \ldots,$$

skirtą sprendinio reikšmių sekančiu (vėlesniu) diskretizuoto laiko momentu masyvui reprezentuoti.

Be to, algoritme naudojami analogiški žymėjimai duotai funkcijai $f(x,t)$:

$$f_j = f(jh, t), \qquad \hat{f}_j = f(jh, t + \tau), \qquad j = 0, 1, \ldots, N, \qquad t = 0, \tau, 2\tau, \ldots.$$

Taigi, vykdydami algoritmą pirmiausiai pasirenkame masyvų ilgį $N + 1$ (sveikąjį skaičių, atitinkantį diskretizacijos tinklo taškų skaičių intervale $0 < x < 1$), pvz. $N = 100$ arba $N = 1000$. Tuomet $h = 1/N$ bus mūsų diskretizacijos žingsnis pagal erdvės kintamąjį x. Taip pat pasirenkame $\tau > 0$ – diskretizacijos žingsnį pagal laiko kintamąjį t, pvz. $\tau = 0.01$, $\tau = 0.005$ ar $\tau = 0.0001$. Matome, kad kuo artimesni nuliui algoritmo žingsniai $h > 0$ ir $\tau > 0$, tuo tiksliau funkcijų reikšmių masyvai atkartos matematines funkcijas. Tačiau mažinant žingsnius, masyvų ilgis didėja, algoritmo vykdymo trukmė taip pat auga.

Kadangi funkcija $f(x,t)$ duota, tai jos reikšmių masyvai f_j ir $\hat{f}_j$ bus lengvai apskaičiuojami bet kokiu diskretaus laiko momentu $t = 0, \tau, 2\tau, \ldots$.

Tuo tarpu modelio sprendinio $u(x,t)$ reikšmių masyvą lengva apskaičiuoti tik pradiniu laiko momentu $t = 0$, iš duotos pradinės sąlygos $u^{(0)}(x)$. Tačiau kaip apskaičiuoti sprendinio reikšmių masyvus kai $t = \tau, 2\tau, \ldots$?

Pagrindinis algoritmo tikslas – turint masyvą u_j rasti sekantį masyvą $\hat{u}_j$. Įvykdę šį žingsnį (kaip tai padaryti paaiškinsime toliau) $\hat{u}_j$ pervardiname į u_j ir ciklišk ai kartojame aptariamąjį algoritmo žingsnį. Tokiu būdu, surasime sprendinio reikšmių masyvus visais diskretizuoto laiko momentais $t = 0, \tau, 2\tau, \ldots$. Algoritmas baigia darbą kai laikas t pasiekia nurodytą (šią konstantą apibrėžia vartotojas) sprendinio evoliucionavimo trukmę T, pvz. $T = 2$ (sekundės).

Kaip surasti masyvą $\hat{u}_j$, kai jau žinome masyvą u_j? Tam reikia išspręsti netiesinių lygčių sistemą, gaunamą iš nurodyto algoritmo, pvz. A1, E2 ar kito (pasirinkti pagal Jūsų užduotį). Dar dvi papildomas lygtis gauname iš kraštinių sąlygų I arba II (pagal Jūsų užduotį) aproksimacijos. Visais atvejais gaunamos lygčių sistemos nežinomieji yra $\hat{u}_0, \hat{u}_1, \ldots, \hat{u}_N$, taigi iš viso $N + 1$ nežinomųjų. Matyti (algoritme indeksas j kinta $j =$

$1,2,\ldots,N-1$, plius dvi lygtys iš kraštinių sąlygų aproksimacijos), kad gaunama lygčių sistema sudaryta iš $N+1$ netiesinių lygčių. Taigi, lygčių yra tiek pat, kiek ir nežinomųjų, būtent $N+1$.

Belieka mokėti išspręsti gautą *netiesinių* lygčių sistemą. Kaip dažnai elgiamasi netiesinio modeliavimo algoritmuose, netiesinių lygčių sistema pakeičiama serija tiesinių lygčių sistemų. Šioje serijoje (cikle) tiesines lygčių sistemas sudarome taip: algoritme prie netiesiškai įeinančių nežinomųjų prirašome "old", o prie tiesiškai įeinančių nežinomųjų – "new".

Pavyzdžiui, A1 aproksimacijai su I kraštinėmis sąlygomis tiesinių lygčių sistema konstruojama taip:

$$\frac{\hat{u}_j^{new}-u_j}{\tau}=a^2\frac{1}{2}\left(\frac{\hat{u}_{j+1}^{new}-2\hat{u}_j^{new}+\hat{u}_{j-1}^{new}}{h^2}+\frac{u_{j+1}-2u_j+u_{j-1}}{h^2}\right)+$$

$$\frac{\hat{u}_j^{old}+u_j}{2}\frac{1}{2}\left(\frac{\hat{u}_{j+1}^{old}-\hat{u}_{j-1}^{old}}{2h}+\frac{u_{j+1}-u_{j-1}}{2h}\right)+\frac{\hat{f}_j+f_j}{2},\qquad j=1,2,\ldots,N-1$$

$$\hat{u}_0^{new}=0,\qquad \hat{u}_N^{new}=0.$$

Čia masyvo reikšmės $\hat{u}_j^{old}$ žymi jau apskaičiuotąją serijos iteraciją, tuo tarpu masyvų reikšmes u_j, $\hat{f}_j$ ir f_j žinome (žr. aukščiau esantį tekstą). Šioje *tiesinių* (todėl nesunkiai išsprendžiamoje) lygčių sistemoje nežinomieji yra $\hat{u}_j^{new}$.

Išsprendę sukonstruotąją tiesinę sistemą (kiekvienos pratybų užduoties atveju ši sistema bus skirtinga) ir pervardinę $\hat{u}_j^{new}$ į $\hat{u}_j^{old}$ ciklišk ai kartojame šią procedūrą. Taigi daug kartų tiksliname sprendinio sekančiu laiko momentu įvertį $\hat{u}_j^{old}$. Ciklą sustabdysime kai šis tikslinimas taps kontroliuojamai mažas:

$$\max_{j=0,1,\ldots,N}\left|\hat{u}_j^{new}-\hat{u}_j^{old}\right|<\delta,$$

kur δ yra vartotojo pasirinktas tikslumas, pvz. $\delta=10^{-5}$. Baigus ciklą, paskutinysis (tiksliausias) $\hat{u}_j^{new}$ ir bus apytikslis (paklaida priklauso nuo δ pasirinkimo) netiesinės lygčių sistemos sprendinys $\hat{u}_j$.

Kaip ir kiekviename cikle, būtina turėti pirmąją iteraciją (patį pirmąjį masyvą $\hat{u}_j^{old}$). Serijos pradžioje pasirenkame

$$\hat{u}_j^{old}=u_j,$$

taigi pradinis grubus įvertis (kurį vėliau tikslinsime) bus tiesiog sprendinio reikšmės praeitu (senu) laiko momentu.

Liko aptarti kaip spręsti tiesinių lygčių sistemas $\hat{u}_j^{new}$ atžvilgiu. Trumpumo tikslu pažymėję $y_j=\hat{u}_j^{new}$ ir sugrupavę narius (po elementarių algebrinių pertvarkymų), sukonstruotąją tiesinių lygčių sistemą užrašome tokiu lakonišku pavidalu:

$$y_{j-1}-Cy_j+y_{j+1}=-F_j,\qquad j=1,2,\ldots,N-1,$$

ir

$$y_0=0,\qquad y_N=0,$$

arba (priklausomai nuo kraštinių sąlygų)

$$y_0=y_1,\qquad y_N=y_{N-1},$$

su žinoma kompleksine konstanta C ir žinomu kompleksinių skaičių masyvu F_j.

Tai yra tiesinių algebrinių lygčių sistema (sudaryta iš $N+1$ lygčių), kurios nežinomieji yra $y_0,y_1,\ldots,y_N$ ($N+1$ nežinomųjų). Šios sistemos matrica yra tridiagonalinė (visi matricos elementai lygūs nuliui, išskyrus tris pagrindines jos įstrižaines). Tokių sistemų sprendimui galime taikyti (taip ir darykite programuodami) paprastą ir efektyvų *perkelties* metodą (angliškoje literatūroje dar vadinamą "Thomas algorithm") – žr. papildomos literatūros sąrašą, jo supaprastintas variantas taip pat bus išdėstytas paskaitose.

<u>*Darbo tikslas*</u>*:*

Sudaryti testinį uždavinį: iš anksto sugalvoti sprendinio funkciją $u_{tikslus}(x,t)$, visiems t tenkinančią kraštines sąlygas. Pavyzdžiui, I kraštinių sąlygų atveju galima pasirinkti

$$u_{tikslus}(x,t) = (1+it)\sin(2\pi x),$$

o II kraštinių sąlygų atveju

$$u_{tikslus}(x,t) = (1+it)\cos(2\pi x).$$

Įsistatę pasirinktąją funkciją $u_{tikslus}(x,t)$ į duotąjį evoliucinį modelį (lygtį) apskaičiuokite kokia turėtų būti funkcija $f(x,t)$, kad $u_{tikslus}(x,t)$ iš tiesų būtų šios lygties sprendinys. Lygtyje esančias išvestines apskaičiuokite rankiniu būdu (ant popieriaus, prisiminę išvestinių skaičiavimo taisykles) arba naudodami simbolinių skaičiavimų paketą (pvz. Mathematica, Maple).

Pradinė sąlyga, suderinta su Jūsų testiniu sprendiniu bus

$$u^{(0)}(x) = u_{tikslus}(x,0).$$

Parašykite kompiuterinę programą, nurodytu algoritmu sprendžiančią paskirtą netiesinį evoliucinį modelį, sudarytą iš lygties, kraštinių sąlygų bei pradinės sąlygos.

Palyginkite kompiuteriu apskaičiuotą sprendinį (masyvus u_j) su žinomu tiksliu sprendiniu $u_{tikslus}(x,t)$. Jei maksimali paklaida

$$\max_{t=0,\tau,\ldots,T}\ \max_{j=0,1,\ldots,N} |u_j - u_{tikslus}(jh,t)|$$

turi tendenciją mažėti mažinant algoritmo žingsnius h ir τ, tai yra geras ženklas, kad galbūt Jūsų programa veikia be klaidų.

Patartina (nors neprivaloma) papildomai testuoti atskirus algoritmo blokus, ypač tiesinių lygčių sistemos sprendimą perkelties metodu. Pagalvokite kaip tai galima padaryti, jei nesugalvosite – klauskite dėstytojo.

Jeigu norite pamatyti (tai daryti neprivaloma) apskaičiuotojo sprendinio evoliuciją grafiškai (animaciją laike), animuokite (prieš tai pavaizduotąjį grafiką nutriname, naujai apskaičiuotojo pavaizduojame) masyvų reikšmių modulius $|u_j|$.

Užduotys

1. Lygtis A, algoritmas A1, kraštinės sąlygos I.
2. Lygtis A, algoritmas A1, kraštinės sąlygos II.
3. Lygtis A, algoritmas A2, kraštinės sąlygos I.
4. Lygtis A, algoritmas A2, kraštinės sąlygos II.
5. Lygtis A, algoritmas A3, kraštinės sąlygos I.
6. Lygtis A, algoritmas A3, kraštinės sąlygos II.
7. Lygtis B, algoritmas B1, kraštinės sąlygos I.
8. Lygtis B, algoritmas B1, kraštinės sąlygos II.
9. Lygtis B, algoritmas B2, kraštinės sąlygos I.
10. Lygtis B, algoritmas B2, kraštinės sąlygos II.
11. Lygtis B, algoritmas B3, kraštinės sąlygos I.
12. Lygtis B, algoritmas B3, kraštinės sąlygos II.
13. Lygtis C, algoritmas C1, kraštinės sąlygos I.
14. Lygtis C, algoritmas C1, kraštinės sąlygos II.
15. Lygtis C, algoritmas C2, kraštinės sąlygos I.
16. Lygtis C, algoritmas C2, kraštinės sąlygos II.
17. Lygtis C, algoritmas C3, kraštinės sąlygos I.
18. Lygtis C, algoritmas C3, kraštinės sąlygos II.
19. Lygtis D, algoritmas D1, kraštinės sąlygos I.
20. Lygtis D, algoritmas D1, kraštinės sąlygos II.
21. Lygtis D, algoritmas D2, kraštinės sąlygos I.
22. Lygtis D, algoritmas D2, kraštinės sąlygos II.
23. Lygtis E, algoritmas E1, kraštinės sąlygos I.
24. Lygtis E, algoritmas E1, kraštinės sąlygos II.
25. Lygtis E, algoritmas E2, kraštinės sąlygos I.
26. Lygtis E, algoritmas E2, kraštinės sąlygos II.
27. Lygtis F, algoritmas F1, kraštinės sąlygos I.
28. Lygtis F, algoritmas F1, kraštinės sąlygos II.
29. Lygtis F, algoritmas F2, kraštinės sąlygos I.
30. Lygtis F, algoritmas F2, kraštinės sąlygos II.
31. Lygtis F, algoritmas F3, kraštinės sąlygos I.

32. Lygtis F, algoritmas F3, kraštinės sąlygos II.
33. Lygtis G, algoritmas G1, kraštinės sąlygos I.
34. Lygtis G, algoritmas G1, kraštinės sąlygos II.
35. Lygtis G, algoritmas G2, kraštinės sąlygos I.
36. Lygtis G, algoritmas G2, kraštinės sąlygos II.
37. Lygtis H, algoritmas H1, kraštinės sąlygos I.
38. Lygtis H, algoritmas H1, kraštinės sąlygos II.
39. Lygtis H, algoritmas H2, kraštinės sąlygos I.
40. Lygtis H, algoritmas H2, kraštinės sąlygos II.

Papildoma literatūra

1. `http://en.wikipedia.org/wiki/Burgers'_equation` – apie vieno iš evoliucinių modelių (Burgers'o lygties) prasmę
2. `http://en.wikipedia.org/wiki/Nonlinear_Schroedinger_equation` – apie vieno iš evoliucinių modelių (netiesinės Šriodingerio lygties) prasmę
3. `http://en.wikipedia.org/wiki/Tridiagonal_matrix_algorithm` – perkelties metodas (Thomas algorithm) tiesinių algebrinių lygčių sistemai su tridiagonaline matrica spręsti