UNIVERZA V LJUBLJANI
FAKULTETA ZA MATEMATIKO IN FIZIKO

Matematika – 2. stopnja

Jure Slak

TBA

Magistrsko delo

Mentor: doc. dr. George Mejak
Somentor: dr. Gregor Kosec

Ljubljana, 2017

Kazalo

TBA

Povzetek

TBA

TBA

Abstract

TBA

Math. Subj. Class. (2010):
Ključne besede: ??
Keywords: ??

1 Uvod

Uporabili bomo [1].

2 Numerična metoda

V 20. stoletju je skupaj z razvojem računalnikov začel svojo pot razvoj numeričnih metod za reševaje parcialnih diferencialnih enačb. Do danes je bilo razvitih veliko metod za numerično reševanje parcialnih diferencialnih enačb. Dva pomembna razreda metod se ločita glede na obliko v kateri rešujemo parcialno diferencialno enačbo: šibki (*angl.* weak form) ali močni (*angl.* strong form). Najznamenitejši in zelo uspešen predstavnik prve skupine je metoda končnih elementov (MKE) (*angl.* finite element method (FEM), kjer problem najprej prevedemo v šibko obliko, nato pa rešitev poiščemo kot linearno kombinacijo baznih funkcij iz izbranega prostora. Najbolj poznan, predstavnik druge pa je metoda končnih diferenc (MKD) (*angl.* finite diference method (FDM)), pri kateri direktno diskretiziramo operator, ki nastopa v enačbi.

Poleg tega se metode delijo tudi, glede na tip diskretizacije domene, ki ga potrebujejo. Metoda končnih elementov potrebuje *mrežo*, nad katero deluje, tj. triangulacijo notranjosti domene, ki inducira tudi mrežo na robu. Metoda robnih elementov potrebuje samo mrežo na robu domene. Metoda končnih diferenc je običajno formulirana na pravokotni mreži. Obstajajo pa tudi metode, ki mreže ne potrebujejo, imenujemo jih *brezmrežne metode* (*angl.* meshfree methods). Predstavljena metoda v tem razdelku, bo reševala enačbo v močni obliki in bo brezmrežna.

2.1 Izpeljava

Izpeljavo začnimo z osvežitvijo spomina na metodo končnih diferenc, ki nam bo služila kot motivacija.

2.1.1 Ideja in motivacija

Primer 2.1. Rešujemo enodimenzionalno Poissonovo enačbo. Izpeljava metode končnih diferenc ne bo povsem običajna in tudi ne najkrajša možna, ampak bo narejena tako, da jo bomo lahko posplošili v brezmrežno metodo.

Rešujemo problem z mešanimi robnimi pogoji

$$\begin{aligned} u''(x) &= f(x) \quad \text{na } (a,b) \\ u(a) &= A \\ u'(b) &= B, \end{aligned} \tag{1}$$

katerega rešitev poznamo v kvadraturah

$$u(x) = \int_a^x \left(\int_b^\eta f(\xi) d\xi \right) d\eta + B(x-a) + A.$$

Numeričnega reševanja se lotimo tako, da interval $[a,b]$ diskretiziramo na N enakih delov dolžine $h = \frac{b-a}{N}$ z delilnimi točkami $x_i = a + ih$, za $i = 0, \ldots, N$. Za vsako od teh točk uvedemo neznanko u_i, ki predstavlja neznano funkcijsko vrednost v točki x_i. S pomočjo vrednosti u_{i-1}, u_i in u_{i+1} želimo sedaj aproksimirati $u''(x_i)$. To nam bo dalo zvezo med spremenljivkami in ko jo uporabimo za vse notranje točke ter upoštevamo še robne pogoje, bomo dobili sistem linearnih enačb, katerega rešitev nam bo dala dobro aproksimacijo funkcije u.

Funkcijo u v okolici x_i aproksimiramo z interpolacijskim polinomom, njene odvode pa z odvodi interpolacijskega polinoma. Da najdemo interpolacijski polinom $\hat{u}$, zapišimo

$$\hat{u}(x) = \alpha_0 + \alpha_1 x + \alpha_2 x^2 = \begin{bmatrix} 1 & x & x^2 \end{bmatrix} \begin{bmatrix} \alpha_0 \\ \alpha_1 \\ \alpha_2 \end{bmatrix} = \boldsymbol{b}(x)^\mathsf{T} \boldsymbol{\alpha}$$

in poiščimo koeficiente $\boldsymbol{\alpha}$, da bo veljalo

$$\begin{aligned} \hat{u}(x_{i-1}) &= u_{i-1} \\ \hat{u}(x_i) &= u_i \\ \hat{u}(x_{i+1}) &= u_{i+1}. \end{aligned}$$

Če sistem enačb razpišemo, dobimo

$$\begin{aligned} \alpha_0 + \alpha_1(x_i - h) + \alpha_2(x_i - h)^2 &= u_{i-1} \\ \alpha_0 + \alpha_1 x_i + \alpha_2 x_i^2 &= u_i \\ \alpha_0 + \alpha_1(x_i + h) + \alpha_2(x_i + h)^2 &= u_{i+1} \end{aligned}$$

oziroma v matrični obliki

$$\begin{bmatrix} 1 & x_i - h & (x_i - h)^2 \\ 1 & x_i & x_i^2 \\ 1 & x_i + h & (x_i + h)^2 \end{bmatrix} \begin{bmatrix} \alpha_0 \\ \alpha_1 \\ \alpha_2 \end{bmatrix} = \begin{bmatrix} u_{i-1} \\ u_i \\ u_{i+1} \end{bmatrix}.$$

Krajše ga zapišemo kar kot $B\boldsymbol{\alpha} = \boldsymbol{u}$. Sistem rešimo in dobimo

$$\begin{aligned} \alpha_0 &= \frac{2h^2 u_i + h(u_{i-1} - u_{i+1})x_i + (u_{i-1} - 2u_i + u_{i+1})x_i^2}{2h^2} \\ \alpha_1 &= \frac{h(u_{i+1} - u_{i-1}) - 2(u_{i-1} - 2u_i + u_{i+1})x_i}{2h^2} \\ \alpha_2 &= \frac{u_{i-1} - 2u_i + u_{i+1}}{2h^2}. \end{aligned}$$

Interpolacijski polinom skozi točke (x_i, u_i) lahko sedaj zapišemo kot

$$\begin{aligned} \hat{u}(x) &= \begin{bmatrix} 1 & x & x^2 \end{bmatrix} \begin{bmatrix} \alpha_0 \\ \alpha_1 \\ \alpha_2 \end{bmatrix} = \\ &= u_i + \frac{u_{i+1} - u_{i-1}}{2h}(x - x_i) + \frac{u_{i-1} - 2u_i + u_{i+1}}{2h^2}(x - x_i)^2 \end{aligned}$$

Toda, ker u_j nastopajo linearno, lahko napišemo tudi v obliki

$$\hat{u}(x) = \begin{bmatrix} \frac{(x_i - x)(h + x_i - x)}{2h^2} & \frac{(h + x - x_i)(h + x_i - x)}{h^2} & \frac{(x - x_i)(h + x - x_i)}{2h^2} \end{bmatrix} \begin{bmatrix} u_{i-1} \\ u_i \\ u_{i+1} \end{bmatrix} = \boldsymbol{\varphi}(x)^\mathsf{T} \boldsymbol{u}.$$

S tem smo ločili podatke, ki se nanašajo na vrednost funkcije, od podatkov, ki se nanašajo na pozicije točk. Če na primer vemo, da bomo večkrat potrebovali vrednost

interpolacijskega polinoma v neki točki x^* za različne nabore funkcijskih vrednosti (vendar še vedno izmerjene v istih točkah) $\boldsymbol{u}$, potem se nam splača poračunati $\boldsymbol{\varphi}(x^*)$ vnaprej in vrednosti interpolacijskega polinoma dobimo vsakič znova le s skalarnim produktom $\hat{u}(x^*) = \boldsymbol{\varphi}(x^*)^\mathsf{T}\boldsymbol{u}$.

Za aproksimacijo u' in u'' bomo vzeli kar odvode $\hat{u}$. Izračunajmo jih v točki x_i in dobimo znane formule

$$\hat{u}'(x_i) = \boldsymbol{\varphi}'(x_i)^\mathsf{T}\boldsymbol{u} = \begin{bmatrix} -\frac{1}{2h} & 0 & \frac{1}{2h} \end{bmatrix} \begin{bmatrix} u_{i-1} \\ u_i \\ u_{i+1} \end{bmatrix}$$
$$\hat{u}''(x_i) = \boldsymbol{\varphi}''(x_i)^\mathsf{T}\boldsymbol{u} = \begin{bmatrix} \frac{1}{h^2} & -\frac{2}{h^2} & \frac{1}{h^2} \end{bmatrix} \begin{bmatrix} u_{i-1} \\ u_i \\ u_{i+1} \end{bmatrix}.$$

To lahko uporabimo za reševanje našega problema (1). Namesto enakosti

$$u''(x_i) = f(x_i)$$

za vsako točko x_i v notranjosti naredimo zapišemo podobno enakost

$$\begin{bmatrix} \frac{1}{h^2} & -\frac{2}{h^2} & \frac{1}{h^2} \end{bmatrix} \begin{bmatrix} u_{i-1} \\ u_i \\ u_{i+1} \end{bmatrix} = f(x_i).$$

Dirichletov pogoj na levem robu zapišemo preprosto kot $u_0 = A$, za Neumannovega na desnem robu pa lahko uporabimo npr. enostransko diferenco na treh točkah

$$\begin{bmatrix} \frac{1}{2h} & \frac{-2}{h} & \frac{3}{2h} \end{bmatrix} \begin{bmatrix} u_{N-2} \\ u_{N-1} \\ u_N \end{bmatrix} = B,$$

ki bi jo izpeljali na enak način.

Vse te enakosti zložimo sistem enačb in ga zapišimo v matrični obliki

$$\begin{bmatrix} 1 & & & & & & \\ -1 & 2 & 1 & & & & \\ & -1 & 2 & 1 & & & \\ & & \ddots & \ddots & \ddots & & \\ & & & -1 & 2 & 1 & \\ & & & 1/2h & -2/h & 3/2h \end{bmatrix} \begin{bmatrix} u_0 \\ u_1 \\ u_2 \\ \vdots \\ u_{N-1} \\ u_N \end{bmatrix} = \begin{bmatrix} f(x_0) \\ f(x_1) \\ f(x_2) \\ \vdots \\ f(x_{N-1}) \\ f(x_N) \end{bmatrix}.$$

Rešitev tega sistema nam dobro aproksimira neznano funkcijo u v izbranih točkah x_i.

2.1.2 Splošna izpeljava

Postavimo se sedaj v splošnejši okvir. Rešujmo parcialno diferencialno enačbo

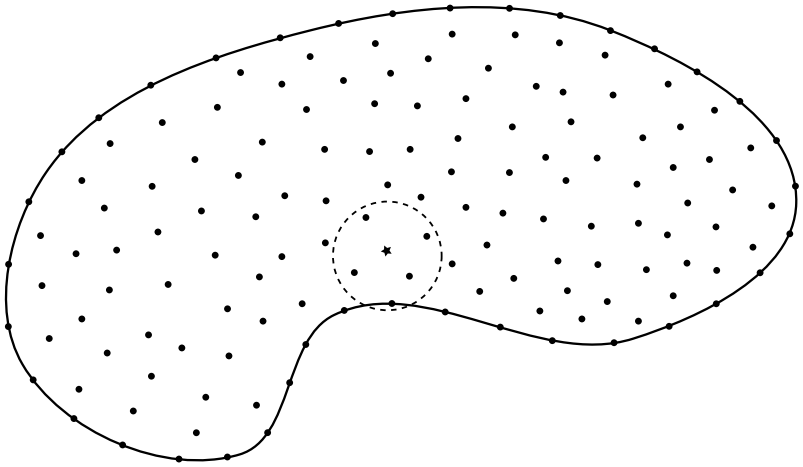
$$\begin{aligned} \mathcal{L}u &= f \text{ na } \Omega, \\ \mathcal{R}u &= g \text{ na } \partial\Omega, \end{aligned} \tag{2}$$

kjer je $\Omega \subseteq \mathbb{R}^d$ omejena domena, torej, omejena povezana odprta množica z odsekoma gladkim robom, $u \in C^r(\mathbb{R}^d)$ funkcija, $\mathcal{L}\colon C^r(\mathbb{R}^d) \to C(\mathbb{R})$ linearen parcialni diferencialni operator reda r in $\mathcal{R}u$ robni pogoji, pri katerih je problem enolično rešljiv.

Domeno in njen rob sedaj diskretiziramo, tako da izberemo N točk v zaprtju domene, $x_1, \ldots, x_N \in \overline{\Omega}$. Podobno kot pri končnih diferencah bomo v teh točkah aproksimirali vrednost funkcije u. Izberimo fiksno točko $p \in \overline{\Omega}$ in n izmed točk $\{x_1, \ldots, x_N\}$, ki bodo sestavljali *soseščino* (*angl.* support) točke p. Število n imenujemo velikost soseščine. Označimo z $\mathcal{N}(p)$ soseščino točke p in z $\mathcal{I}(p) = \{i_1, \ldots, i_n\}$ množico indeksov, za katere so izbrani x_{i_j} v soseščini p. Velja torej

$$\mathcal{N}(p) = \bigcup_{i \in \mathcal{I}(p)} x_i.$$

Običajno bo $n \ll N$, npr. $n = 9$ in $N = 10^6$. Primer domene Ω, točke p in njene soseščine je prikazan na sliki 1.



Slika 1: Primer domene z diskretizirano notranjostjo in robom skupaj z izbrano točko in njeno soseščino.

V okolici točke p aproksimirajmo u z elementi iz nekega končno dimenzionalnega prostora funkcij $\mathcal{B} = \mathcal{L}in\{b_1, \ldots, b_m\}$. Funkcijam $b_i\colon \mathbb{R}^d \to \mathbb{R}$ pravimo *bazne funkcije*, številu m pa moč baze. Aproksimacijo za $\hat{u}$ za u lahko torej zapišemo kot

$$u \approx \hat{u} = \sum_{i=1}^{m} \alpha_i b_i = \boldsymbol{b}^{\mathsf{T}} \boldsymbol{\alpha},$$

pri čemer smo z $\boldsymbol{\alpha} = (\alpha_i)_{i=1}^m$ označili vektor neznanih koeficientov in $\boldsymbol{b} = (b_i)_{i=1}^m$ vektor baznih funkcij.

Če bi poznali vrednosti $u(x_i)$ za $i \in I(p)$, potem bi lahko aproksimiranko $\hat{u}$ izračunali po metodi najmanjših kvadratov. Ker pa teh vrednosti ne poznamo, uvedimo spremenljivke u_i za vsako točko v domeni, ki nam bodo predstavljale neznane prave vrednosti in nadaljujmo s simbolnim računanjem. Za funkcijo $\hat{u}$ zahtevamo, da aproksimira u v smislu utežene diskretne 2-norme, torej da minimizira

$$\|u - \hat{u}\|_{2,N(p),\boldsymbol{w}} = \sum_{i \in I(p)} w(p - x_i)(u_i - \hat{u}(x_i))^2$$

kar je utežena vsota kvadratov odstopanj od pravih vrednosti. Pri tem je $w\colon \mathbb{R}^d \to \mathbb{R}$ nenegativna funkcija, ki jo imenujemo *utež*, $\boldsymbol{w}$ pa je vektor sestavljen iz vrednosti te funkcije v točkah v soseščini.

Matrično lahko zapišemo sistem enačb, ki po vrsticah postavlja našo zahtevo $\hat{u}(x_j) = u_j$, za vsak $j \in I(p)$:

$$\begin{bmatrix} b_1(x_{i_1}) & \cdots & b_m(x_{i_1}) \\ \vdots & \ddots & \vdots \\ b_1(x_{i_n}) & \cdots & b_m(x_{i_n}) \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \vdots \\ \alpha_m \end{bmatrix} = \begin{bmatrix} u_{i_1} \\ \vdots \\ u_{i_n} \end{bmatrix}.$$

Na krajše sistem zapišemo kot $B\boldsymbol{\alpha} = \tilde{\boldsymbol{u}}$. Odvisno od n, m, b_i in uteži w je ta sistem lahko poddoločen, predoločen ali običajen. V vsakem primeru lahko minimiziranje utežene napake prevedemo na minimiziranje diskretne 2-norme $\|WB\boldsymbol{\alpha} - W\tilde{\boldsymbol{u}}\|_{2,N(p)}$, kjer je W diagonalna matrika iz korenov uteži za posamezne točke, $W = \operatorname{diag}(\sqrt{w(x_{i_1} - p)}, \ldots, \sqrt{w(x_{i_n} - p)})$. Tak sistem pa lahko ne glede na njegovo določenost v rešimo s pomočjo Moore-Penroseovega psevdoinverza, ki ga lahko izračunamo s pomočjo singularnega razcepa matrike WB. Tako lahko izrazimo

$$\boldsymbol{\alpha} = (WB)^{+} W\tilde{\boldsymbol{u}},$$

kjer $+$ označuje Moore-Penroseov psevdoinverz.

To lahko vstavimo nazaj v izraz za $\hat{u}$ in dobimo

$$\hat{u} = \boldsymbol{b}^{\mathsf{T}}\boldsymbol{\alpha} = \boldsymbol{b}^{\mathsf{T}}(WB)^{+} W\tilde{\boldsymbol{u}}.$$

Sedaj lahko za izbrano točko p izračunamo

$$\hat{u}(p) = \underbrace{\boldsymbol{b}(p)^{\mathsf{T}}(WB)^{+} W}_{\boldsymbol{\varphi}_p}\tilde{\boldsymbol{u}}.$$

Izračunljivi kos $\boldsymbol{\varphi}_p$ je v praksi vrstica velikosti n, matematično pa je linearen funkcional $\boldsymbol{\varphi}_p \in (\mathbb{R}^n)^*$, ki naboru funkcijskih vrednosti v soseščini $N(p)$ priredi aproksimacijo za funkcijsko vrednost v točki p.

Podobno kot pri deljenih diferencah odvode funkcije u aproksimiramo z odvodi aproksimacijskega polinoma skozi točke v soseščini, bomo tudi v našem primeru aproksimirali odvode funkcije u z odvodi $\hat{u}$,

$$(\mathcal{L}u)(p) \approx (\mathcal{L}\hat{u})(p) = (\mathcal{L}\boldsymbol{b})(p)^{\mathsf{T}}(WB)^{+} W\tilde{\boldsymbol{u}}$$

od koder kot prej definiramo

$$\boldsymbol{\varphi}_{\mathcal{L},p} = (\mathcal{L}\boldsymbol{b})(p)^{\mathsf{T}}(WB)^{+} W \tag{3}$$

Funkcional $\boldsymbol{\varphi}_{\mathcal{L},p}$ je aproksimacija operatorja $\mathcal{L}$ v točki p. Pogosto se ga imenuje tudi *funkcija oblike* (*angl.* shape function), saj v sebi nosi podatke o lokalni obliki domene in izboru okoliških točk, ter seveda o obnašanju $\mathcal{L}$ v tej okolici. Tudi če funkcijskih vrednosti $\tilde{\boldsymbol{u}}$ v okolici p ne poznamo, lahko $\boldsymbol{\varphi}_{\mathcal{L},p}$ izračunamo in kasneje samo s skalarnim produktom dobimo aproksimacijo za $(\mathcal{L}u)(p)$. Lahko pa to izkoristimo za zapis linearne enačbe

$$\boldsymbol{\varphi}_{\mathcal{L},p} \cdot \tilde{\boldsymbol{u}} = f(p),$$

ki je direktna aproksimacija diferencialne enačbe (2) v točki p,

$$(\mathcal{L}u)(p) = f(p).$$

To lahko sedaj storimo za vsako diskretizacijsko točko x_i v domeni in dobimo sistem enačb

$$\begin{aligned}\boldsymbol{\varphi}_{\mathcal{L},x_i} \cdot \tilde{\boldsymbol{u}} &= f(x_i), \text{ za vsak } i, \text{ tak da je } x_i \in \Omega \\ \boldsymbol{\varphi}_{\mathcal{R},x_i} \cdot \tilde{\boldsymbol{u}} &= g(x_i), \text{ za vsak } i, \text{ tak da je } x_i \in \partial\Omega.\end{aligned}$$

Te enačbe lahko zapišemo v matrični sistem

$$A\boldsymbol{u} = \boldsymbol{f}, \tag{4}$$

kjer ima matrika A v vrsticah zapisane funkcionale $\boldsymbol{\varphi}_{\mathcal{L},x_i}$, tako da so neničelni elementi na tistih mestih, ki se pomnožijo z neznankami, ki ustrezajo sosedom x_i. Natančneje, elementi matrike A so

$$\begin{aligned}A(k, i_j) &= \boldsymbol{\varphi}_{\mathcal{L},p}(j), \text{ za vsak } k, \text{ tak da je } x_k \in \Omega \text{ in za vsak } i_j \in \mathcal{I}(x_k), \\ A(k, i_j) &= \boldsymbol{\varphi}_{\mathcal{R},p}(j), \text{ za vsak } k, \text{ tak da je } x_k \in \partial\Omega \text{ in za vsak } i_j \in \mathcal{I}(x_k).\end{aligned}$$

Razumljivejša je morda kar Matlab-ova notacija

$$A(k, I(x_k)) = \begin{cases}\boldsymbol{\varphi}_{\mathcal{L},p} & x_k \in \Omega \\ \boldsymbol{\varphi}_{\mathcal{R},p} & x_k \in \partial\Omega\end{cases}, \text{ za } k = 1, \ldots, N.$$

Vektor $\boldsymbol{u} = (u_i)_{i=1}^N$ je vektor neznanih funkcijskih vrednosti, ki ga iščemo, v vektorju $\boldsymbol{f}$ pa so zapisani robni pogoji

$$\boldsymbol{f}(k) = \begin{cases}f(x_k) & x_k \in \Omega \\ g(x_k) & x_k \in \partial\Omega\end{cases}, \text{ za } k = 1, \ldots, N.$$

Vidimo, da je matrika A razpršena. Sama je dimenzij $N \times N$, v vsaki vrstici pa ima največ n neničelnih elementov, torej je skupno število neničelnih elementov

$$\operatorname{nnz}(A) \leq nN.$$

Enakost je lahko dosežena, lahko pa je tudi stroga, saj so kakšni koeficienti v $\varphi_{\mathcal{L},x_i}$ lahko tudi 0, kot se to zgodi pri Dirichletovih robnih pogojih.

Sistem (4) nato rešimo in za aproksimacijo $u(x_i)$ vzamemo $u(x_i) \approx u_i$.

2.2 Posebni primeri

2.3 Algoritem

V izpeljavi metode v razdelku 2.1.2 je veliko detajlov ostalo neizdelanih, kot na primer, kako diskretiziramo domeno ali kako poiščemo sosede. Ti detajli so opisani v podrazdelkih, celotno metodo pa podajamo v psevdokodi kot algoritem 1.

Algoritem 1 Brezmrežna metoda za reševanje PDE iz razdelka 2.1.2.

Vhod: Parcialna diferencialna enačba, kot opisana v (2). Parametri metode:

- N ... celotno število diskretizacijskih točk
- Q ... število diskretizacijskih točk v notranjosti Ω
- n ... število sosedov, ki jih ima vsaka točka
- m ... število baznih funkcij
- b ... seznam baznih funkcij dolžine m
- w ... utež

Izhod: skalarno polje u, ki aproksimira rešitev enačbe (2).

function SOLVE($\Omega, \mathcal{L}, f, \mathcal{R}, g, N, Q, n, m, b, w$)
  $x \leftarrow$ DISKRETIZIRAJ(Ω, N, Q) ▷ x postane seznam N točk, brez škode za splošnost naj leži prvih Q točk v Ω in preostalih $N - Q$ na $\partial\Omega$.
  $s \leftarrow$ SOSEDI(x, n) ▷ s je seznam dolžine N, pri čemer je $s[i]$ seznam indeksov elementov v x, ki so sosedi $x[i]$, vključno z i.
  $\varphi \leftarrow$ prazen seznam dolžine N.
  for $i \leftarrow 1$ **to** Q **do** ▷ Izračunamo funkcije oblik v notranjosti.
    $\varphi[i] \leftarrow$ FUNKCIJAOBLIKE($\mathcal{L}, x[i], x, s[i], n, m, b, w$) ▷ Glej algoritem 2.
  end for
  for $i \leftarrow Q + 1$ **to** N **do** ▷ Izračunamo funkcije oblik na robu.
    $\varphi[i] \leftarrow$ FUNKCIJAOBLIKE($\mathcal{R}, x[i], x, s[i], n, m, b, w$) ▷ Glej algoritem 2.
  end for
  $A \leftarrow$ prazna razpršena $N \times N$ matrika
  for $i \leftarrow 1$ **to** N **do** ▷ Aproksimiramo enačbo.
    for $j \leftarrow 1$ **to** n **do**
      $A[i, s[j]] \leftarrow \varphi[i][j]$
    end for
  end for
  $r \leftarrow$ prazen vektor dolžine N
  for $i \leftarrow 1$ **to** Q **do** ▷ Izračunamo desno stran v notranjosti.
    $r[i] \leftarrow f(x[i])$
  end for
  for $i \leftarrow Q + 1$ **to** N **do** ▷ Izračunamo robne pogoje.
    $r[i] \leftarrow g(x[i])$
  end for
  $u \leftarrow$ REŠIRAZPRŠENSISTEM(A, r)
  return u
end function

2.3.1 Diskretizacija

Splošno d-dimenzionalno domeno Ω je težko dobro diskretizirati. Želimo si, da bi bile točke v domeni čim bolj enakomerno razporejene, saj to pomeni, da smo dobro popisali celotno področje in upamo, da s tem tudi obnašanje funkcije u, hrakti pa si zaradi numerične stabilnosti ne želimo, da bi bile točke preveč skupaj. Uvedimo dve količini, ki nam merita ti dve lastnosti. Za dano domeno Ω in množico točk X

Algoritem 2 Izračun funkcije oblike.

Vhod:

- $\mathcal{L}$ ... parcialen diferencialen operator
- p ... točka, v kateri aproksimiramo operator
- x ... seznam diskretizacijskih točk
- I ... množica indeksov točk v soseščini p
- n ... število sosedov, ki jih ima vsaka točka
- m ... število baznih funkcij
- b ... seznam baznih funkcij dolžine m
- w ... utež

Izhod: Funkcional, ki aproksimira operator $\mathcal{L}$ v točki p.

function FUNKCIJAOBLIKE($\mathcal{L}, p, x, I, n, m, b, w$)
  $W \leftarrow$ prazen vektor dolžine n
  for $i \leftarrow 1$ **to** n **do**
    $W[i] \leftarrow \sqrt{w(x[I[i]] - p)}$
  end for
  $B \leftarrow$ prazna matrika velikosti $n \times m$
  for $i \leftarrow 1$ **to** n **do**
    for $j \leftarrow 1$ **to** m **do**
      $B[i, j] \leftarrow W[i] \cdot b[j](x[I[i]])$
    end for
  end for
  $\ell \leftarrow$ prazen vektor dolžine m
  for $j \leftarrow 1$ **to** m **do**
    $\ell[j] \leftarrow (\mathcal{L}(b[j]))(p)$
  end for
  $\varphi \leftarrow (\ell \cdot \text{PINV}(B)) \cdot W$ ▷ Direktna analogija enačbe (3).
  return φ
end function

definirajmo

$$h_\Omega(X) = \max_{p \in \Omega} \min_{x \in X} \|p - x\|$$
$$S(X) = \min_{\substack{x,y \in \Omega \\ x \neq y}} \|x - y\|$$

Količina h pove, da ne glede na to, kje v domeni smo, imamo na razdalji manj ali enako h vsaj eno diskretizacijsko točko. Količina S pa pove, kako blizu so si diskretizacijske točke med seboj. Dobra diskretizacija želi maksimizirati S in minimizirati h.

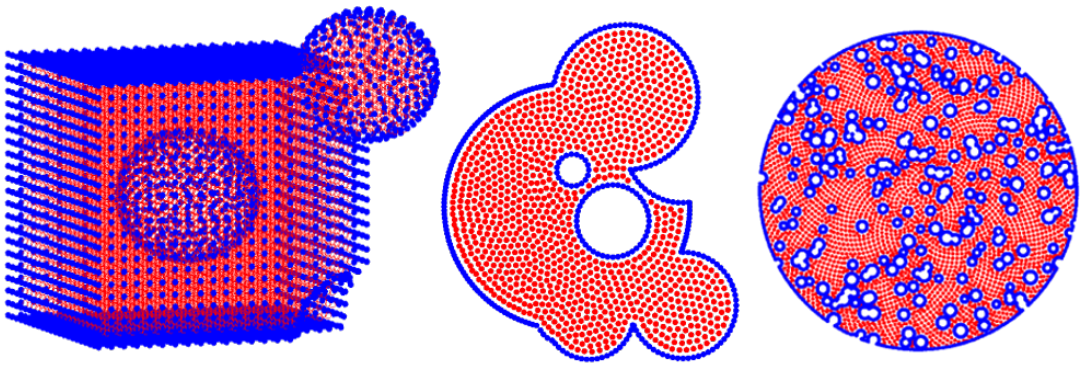
Algoritmi za diskretizacijo odvisni od načina, kako podamo domeno. V našem se izkaže, da je za trenutne potrebe dovolj, da podpiramo kvadre in krogle do vključno treh dimenzij, kot tudi unije in razlike osnovnih oblik.

Tako lahko za diskretizacijo kvadra uporabimo kar enakomerno diskretizacijo. Prav tako ni težko ugotoviti, kdaj so točke na robu. Za čimbolj enakomerno diskretizacijo notranjosti kroga ali površine sfere lahko uporabimo npr. Fibonaccijevo mrežo, kot predlagano v [2] ali v [3]. Pri razlikah domen preprosto izbrišemo točke,

ki so padle izven domene, pri unijah pa naredimo tudi unijo diskretizacij. Pri tem lahko potem naredimo še en korak in pobrišemo ven kakšno izmed točk, ki so si preblizu skupaj.

Pri enakomerni diskretizaciji smo močno uporabili naše znanje o domeni in njeni obliki. Lahko pa, da tega nimamo na voljo in želimo splošnejši algoritem, ki potrebuje npr. samo karakteristično funkcijo domene in njene meje, torej, kakšen kvader, ki našo domeno vsebuje. V tem primeru lahko vzamemo enakomerno diskretizacijo kvadra in odstranimo vse točke, ki niso v domeni, vendar je tu težko kontrolirati število točk. Druga metoda je, da naključno izbiramo točke v kvadru, in sprejmemo tiste, ki pristanejo v notranjosti. Še boljšo diskretizacijo dobimo, če namesto psevdonaključnih števil uporabimo kvazinaključna števila, ki imajo manjšo diskrepanco. Več o tem si bralec lahko prebere v [4].

Primeri domen in njihovih diskretizacij so prikazani na sliki 2.



Slika 2: Primeri domen in njihovih diskretizacij.

2.3.2 Iskanje najbližjih sosedov

2.3.3 Časovna zahtevnost

Literatura

[1] L.P. Lebedev and M.J. Cloud. *Introduction to Mathematical Elasticity*. World Scientific, 2009.

[2] JH Hannay and JF Nye. Fibonacci numerical integration on a sphere. *Journal of Physics A: Mathematical and General*, 37(48):11591, 2004.

[3] Álvaro González. Measurement of areas on a sphere using fibonacci and latitude–longitude lattices. *Mathematical Geosciences*, 42(1):49–64, 2010.

[4] William J Morokoff and Russel E Caflisch. Quasi-random sequences and their discrepancies. *SIAM Journal on Scientific Computing*, 15(6):1251–1279, 1994.