Tomáš Oberhuber

Faculty of Nuclear Sciences and Physical Engineering
Czech Technical University in Prague

Video na Youtube

Samoadjungovaná úloha

Budeme řešit rovnici tvaru:

$$
\begin{aligned}
-(p(x)u'(x))' + q(x)u(x) &= f(x) \text{ na } (0,1) \\
u(0) &= \gamma_1 \\
u(1) &= \gamma_2
\end{aligned}
$$

- úlohu bychom mohli pomocí **metodou střelby převést na řešení ODR**
- tento postup ale nelze zobecnit na úlohy s více prostorovými proměnnými, tj. na **eliptické parciální diferenciální rovnice**
- druhá možnost je převedení úlohy na systém **algebraických rovnic pomocí diskretizace diferenciálního operátoru**

Samoadjungovaná úloha - diskretizace

Interval $(0, 1)$ pokryjeme ekvidistantní sítí uzlů

$$\omega_h = \{x_i = ih \mid i = 0, \ldots N\} ,$$

kde $h = 1/N$ a budeme používat značení $u_i = u(x_i), f_i = f(x_i)$ apod.

Samoadjungovaná úloha - diskretizace

Pak lze psát:

$$
\begin{aligned}
-\left(p\left(x\right)u'\left(x\right)\right)'\,|_{x_i} &\approx -\frac{p(x)u'(x)\,|_{x_{i+1}} - p(x)u'(x)\,|_{x_i}}{h},\\
p(x)u'(x)\,|_{x_{i+1}} &\approx p_{i+1}\frac{u_{i+1}-u_i}{h}\\
p(x)u'(x)\,|_{x_i} &\approx p_i\frac{u_i-u_{i-1}}{h}
\end{aligned}
$$

a tedy

$$
-\left(p\left(x\right)u'\left(x\right)\right)'\,|_{x_i} \approx -\frac{p_i u_{i-1} - (p_i + p_{i+1})u_i + p_{i+1}u_{i+1}}{h^2}
$$

Samoadjungovaná úloha - diskretizace

$$-\frac{p_i u_{i-1} - (p_i + p_{i+1})u_i + p_{i+1} u_{i+1}}{h^2} + q_i u_i = f_i,$$

pro $i = 1, \dots N-1$. Např. pro $q(x) \equiv 0$ dostáváme:

$$\begin{array}{rcl}
u_0 & = & \gamma_1 \\
-p_1 u_0 + (p_1 + p_2)u_1 - p_2 u_2 & = & f_1 h^2 \\
-p_2 u_1 + (p_2 + p_3)u_2 - p_3 u_3 & = & f_2 h^2 \\
-p_3 u_2 + (p_3 + p_4)u_3 - p_4 u_4 & = & f_3 h^2 \\
& \vdots & \\
-p_{N-1} u_{N-2} + (p_{N-1} + p_N)u_{N-1} - p_N u_N & = & f_{N-1} h^2 \\
u_N & = & \gamma_2
\end{array}$$

Samoadjungovaná úloha - diskretizace

To lze přepsat do maticového tvaru jako:

$$\mathbb{A}\mathbf{u} = \mathbf{b},$$

kde

$$\mathbb{A} = \begin{pmatrix} 1 & & & & & & \\ -p_1 & p_1 + p_2 & -p_2 & & & & \\ & -p_2 & p_2 + p_3 & -p_4 & & & \\ & & -p_3 & p_4 + p_5 & -p_5 & & \\ & & & \ddots & \ddots & \ddots & \\ & & & & -p_{N-1} & p_{N-1} + p_N & -p_N \\ & & & & & & 1 \end{pmatrix},$$

$$\mathbf{u} = \begin{pmatrix} u_0 \\ u_1 \\ \vdots \\ u_{n-1} \\ u_n \end{pmatrix}, \qquad \mathbf{b} = \begin{pmatrix} \gamma_1 \\ h^2 f_1 \\ \vdots \\ h^2 f_{N-1} \\ \gamma_2 \end{pmatrix}$$

Řešení **u** lze tak získat řešením soustavy lineárních rovnic.

Poissonova rovnice

A je-li $p(x) \equiv 1$, dostáváme aproximaci tzv. **Poissonovy rovnice**:

$$-u''(x) = f(x) \text{ na } (0,1)$$

ve tvaru:

$$\begin{array}{rcl}
u_0 & = & \gamma_1 \\
-u_0 + 2u_1 - u_2 & = & f_1 h^2 \\
-u_1 + 2u_2 - u_3 & = & f_2 h^2 \\
-u_2 + 2u_3 - u_4 & = & f_3 h^2 \\
 & \vdots & \\
-u_{N-2} + 2u_{N-1} - u_N & = & f_{N-1} h^2 \\
u_N & = & \gamma_2
\end{array}$$

Poissonova rovnice

Matice soustavy pak má tvar:

$$\mathbb{A} = \begin{pmatrix} 1 & & & & & & \\ -1 & 2 & -1 & & & & \\ & -1 & 2 & -1 & & & \\ & & -1 & 2 & -1 & & \\ & & & \ddots & \ddots & \ddots & \\ & & & & -1 & 2 & -1 \\ & & & & & & 1 \end{pmatrix}.$$

Jde pak tedy o tridiagonální soustavu lineárních rovnic, kterou lze dobře řešit pomocí **Thomasova algoritmu**.
Úloha je implementována v `PDE/poisson-1d.cpp`.

Eliptické PDR ve 2D

Budeme řešit úlohu:

$$
\begin{aligned}
-\frac{\partial}{\partial x}\left(p(x,y)\frac{\partial u(x,y)}{\partial x}\right)-\frac{\partial}{\partial y}\left(p(x,y)\frac{\partial u(x,y)}{\partial y}\right) &= f(x,y) \text{ na } \Omega,\\
u(x,y) &= g(x,y) \text{ na } \partial\Omega,
\end{aligned}
$$

kde

- $\Omega \equiv (0,1)\times(0,1)$
- $u : \langle 0,1\rangle \times \langle 0,1\rangle \rightarrow \mathbb{R}$
- $p : \langle 0,1\rangle \times \langle 0,1\rangle \rightarrow \mathbb{R}$
- $g : \langle 0,1\rangle \times \langle 0,1\rangle \rightarrow \mathbb{R}$

První rovnici lze také zapsat takto:

$$
-\nabla\cdot\left(p\left(x,y\right)\nabla u\left(x,y\right)\right) = f(x,y) \text{ na } (0,1)\times(0,1).
$$

Eliptické PDR ve 2D

Uzávěr oblasti $\overline{\Omega}$ nyní diskretizujeme takto

$$\omega_h \equiv \{(ih_x, jh_y) \mid i = 0, \ldots, N_x \wedge j = 0, \ldots, N_y\},$$

kde

- N_x a N_y udává rozlišení diskretizace podle jednotlivých os
- $h_x = 1/N_x$
- $h_y = 1/N_y$
- pro jednoduchost budeme předpokládat $N_x = N_y = N$ a $h_x = h_y = h$, tj.

$$\omega_h \equiv \{(ih, jh) \mid i = 0, \ldots, N \wedge j = 0, \ldots, N\}.$$

Eliptické PDR ve 2D

Pro libovolný vnitřní uzel ij, tj. $i > 0 \wedge j > 0 \wedge i < N \wedge j < N$ lze použitím dopředné diference na vnější derivaci a zpětné diference na vnitřní derivaci psát:

$$
\begin{aligned}
&-\frac{\partial}{\partial x}\left(p(x,y)\frac{\partial u(x,y)}{\partial x}\right)\Big|_{ij} \approx \\
&-\frac{p(x,y)\frac{\partial u(x,y)}{\partial x}\;|_{i+1,j} - p(x,y)\frac{\partial u(x,y)}{\partial x}\;|_{i,j}}{h} \approx \\
&-\frac{1}{h}\left(p_{i+1,j}\frac{u_{i+1,j}-u_{ij}}{h} - p_{i,j}\frac{u_{i,j}-u_{i-1,j}}{h}\right) = \\
&-\frac{p_{i+1,j}u_{i+1,j} - (p_{i+1,j}+p_{ij})u_{ij} + p_{ij}u_{i-1,j}}{h^2}.
\end{aligned}
$$

A podobně lze odvodit

$$
\begin{aligned}
&-\frac{\partial}{\partial y}\left(p(x,y)\frac{\partial u(x,y)}{\partial y}\right)\Big|_{ij} \approx \\
&-\frac{p_{i,j+1}u_{i,j+1} - (p_{i,j+1}+p_{ij})u_{ij} + p_{ij}u_{i,j-1}}{h^2}.
\end{aligned}
$$

Poissonova rovnice ve 2D

Předpokládejme opět, že $p(x, y) \equiv 1$. Pak dostáváme:

$$
\begin{aligned}
&-\nabla \cdot (p\,(x,y)\,\nabla u\,(x,y)) \approx \\
&-\frac{u_{i+1,j} - 2u_{ij} + u_{i-1,j}}{h^2} - \frac{u_{i,j+1} - 2u_{ij} + u_{i,j-1}}{h^2} = \\
&-\frac{u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1} - 4u_{ij}}{h^2}
\end{aligned}
$$

Dostáváme tak soustavu lineárních algebraických rovnic:

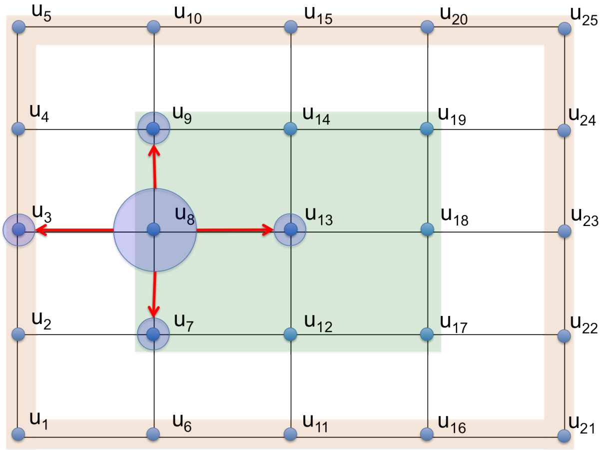
$$
\begin{aligned}
-u_{i+1,j} - u_{i-1,j} - u_{i,j+1} - u_{i,j-1} + 4u_{ij} &= h^2 f_{ij} \\
\text{pro } i, j = 1, \ldots, N-1, & \\
u_{ij} &= g_{ij} \\
\text{pro } i = 0 \vee j = 0 \vee i = N \vee j = N. &
\end{aligned}
$$

Poissonova rovnice ve 2D

- předchozí soustava není vhodná pro přepis do maticového tvaru, neboť neznámé u_{ij} závisí na dvojici indexů i, j
- provedeme transformaci $(i, j) \rightarrow I$ očíslováním uzlů sítě ω_h po řádcích

$$I(i, j) = jN + i.$$

Poissonova rovnice ve 2D



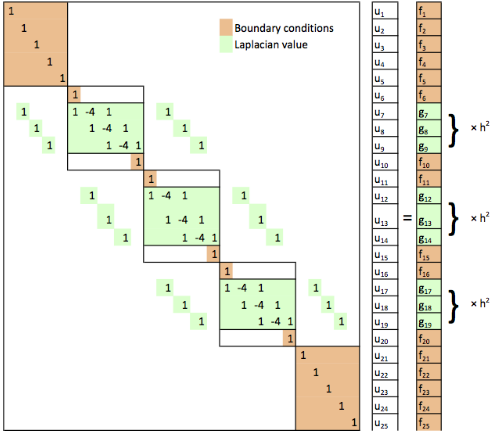
Příklad s indexováním po sloupcích od 1 do N^2.

Poissonova rovnice ve 2D

Pak dostáváme rovnice tvaru:

$$
\begin{aligned}
u_1 &= g_1 \\
u_2 &= g_2 \\
u_3 &= g_3 \\
u_4 &= g_4 \\
u_5 &= g_5 \\
u_6 &= g_6 \\
-u_2 - u_6 - u_8 - u_{12} + 4u_7 &= f_7 \\
-u_3 - u_7 - u_9 - u_{13} + 4u_8 &= f_8 \\
-u_4 - u_8 - u_{10} - u_{14} + 4u_9 &= f_9 \\
u_{10} &= g_{10} \\
u_{11} &= g_{11} \\
-u_7 - u_{11} - u_{13} - u_{17} + 4u_{12} &= f_{12} \\
&\vdots \\
-u_{14} - u_{18} - u_{20} - u_{24} + 4u_{19} &= f_{19} \\
u_{20} &= g_{20} \\
&\vdots \\
u_{25} &= g_{25}
\end{aligned}
$$

Poissonova rovnice ve 2D



Pozn.: Na obrázku je prohozen význam funkcí *f* a *g* a zelené rovnice jsou vynásobené -1.

Ellpack formát

- výsledná matice je řídká, ale ne tridiagonální
- k jejímu uložení použijeme **Ellpack formát**

5		2					
		1					
	3						
5							
4							
	2					9	
		2			5		3
				1			7

values[]		
5	2	0
1	0	0
3	0	0
5	0	0
4	0	0
2	9	0
2	5	3
1	7	0

columns[]		
0	2	*
2	*	*
1	*	*
0	*	*
0	*	*
1	6	*
2	5	7
5	7	*

Ellpack formát

Formát Ellpack lze použít vložením souboru `matrices/EllpackMatrix.h`

- nejprve je potřeba nastavit rozměry matice voláním `setDimensions( rows, columns)` nebo pomocí konstruktoru
- dále je potřeba zadat maximální počet nenulových prvků v řádku (pro všechny řádky je stejný) voláním `setRowLength(nonzeros)`
- následně je možné nastavovat jednotlivé prvky matice voláním `setElement(row,column,value)`

Poissonova rovnice ve 2D

Domácí úkol:

- do souboru `PDE/Poisson2D.py` doplňte potřebný kód pro vyřešení Poissonovy úlohy ve 2D
- proveďte výpočetní studii pro různé okrajové podmínky a pravé strany f
- porovnejte rychlost konvergence různých stacionárních metod
- najděte nejvhodnější parametr ω pro SOR metodu
- zmenšujte h a sledujte počet iterací nutných pro vyřešení lineární soustavy rovnic
- napište report a odevzdejte spolu se zdrojovým kódem

Poissonova rovnice ve 2D

- přesnost aproximace je dána použitými konečnými diferencemi
- bude-li řešení u dostatečně hladké, tj. $u \in C^4(\Omega)$ a tudíž $f \in C^2(\Omega)$, pak jde o náhrady druhé derivace s přesností druhého řádu
- zmenšení $h \to h/2$ způsobí 4x menší chybu aproximace
- zmenšení $h \to h/2$ způsobí 4x více uzlů sítě ω_h
- podmíněnost matice soustavy je úměrná $1/h$, tj. roste počet iterací nutných k vyřešení lineární soustavy
- schémata s vyšším řádem přesnosti tak mohou vést na přesnější řešení s menším počtem uzlů sítě a tedy kratším výpočetním časem

Nelineární úlohy

Mějme úlohu tvaru:

$$
\begin{aligned}
-\nabla \cdot \left(p\left(x, y, u(x, y)\right) \nabla u\left(x, y\right)\right) &= f(x, y) \text{ na } \Omega, \\
u(x, y) &= g(x, y) \text{ na } \partial\Omega.
\end{aligned}
$$

Čím se bude lišit diskretizace této úlohy?

Nelineární úlohy

$$
-\frac{\partial}{\partial x}\left(p(x,y,u)\frac{\partial u(x,y)}{\partial x}\right)\Big|_{ij} \approx
$$

$$
-\frac{p(x,y,u)\frac{\partial u(x,y)}{\partial x}\,|_{i+1,j} - p(x,y,u)\frac{\partial u(x,y)}{\partial x}\,|_{i,j}}{h} \approx
$$

$$
-\frac{1}{h}\left(p(u)_{i+1,j}\frac{u_{i+1,j}-u_{ij}}{h} - p(u)_{i,j}\frac{u_{i,j}-u_{i-1.j}}{h}\right) =
$$

$$
-\frac{p(u)_{i+1,j}u_{i+1,j} - (p(u)_{i+1,j}+p(u)_{ij})u_{ij} + p(u)_{ij}u_{i-1,j}}{h^2}.
$$

Nelineární úlohy

$$
\begin{aligned}
&-\frac{\partial}{\partial x}\left(p(x,y,u)\frac{\partial u(x,y)}{\partial x}\right)\Big|_{ij} \approx \\
&-\frac{p(x,y,u)\frac{\partial u(x,y)}{\partial x}\,\big|_{i+1,j} - p(x,y,u)\frac{\partial u(x,y)}{\partial x}\,\big|_{i,j}}{h} \approx \\
&-\frac{1}{h}\left(p(u)_{i+1,j}\frac{u_{i+1,j}-u_{ij}}{h} - p(u)_{i,j}\frac{u_{i,j}-u_{i-1.j}}{h}\right) = \\
&-\frac{p(u)_{i+1,j}u_{i+1,j} - (p(u)_{i+1,j} + p(u)_{ij})u_{ij} + p(u)_{ij}u_{i-1,j}}{h^2}.
\end{aligned}
$$

- závislost na u už není lineární, ale nelineární
- ve výsledku tak dostaneme soustavu nelineárních algebraických rovnic
- na jejich řešení lze použít Newtonovu metodu