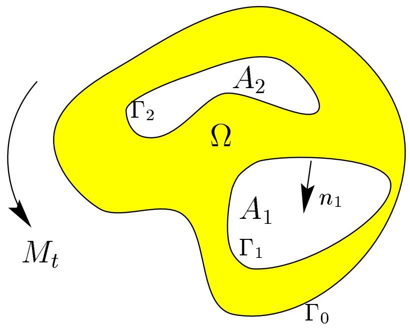
Laboratorijske vaje pri predmetu Računalniško podprto konstruiranje

Leon Kos, Laboratorij za CAD - LECAD

Naloga

Izdelajte program za izračun torzijskega vztrajnostnega momenta poljubnega polnega prereza. Program mora avtomatsko genererirati mrežo poljubne natančnosti. Reševanje parcialne diferencialne enačbe izvedite z metodo končnih diferenc ali metodo končnih elementov. Napetostno funkcijo U prikažite kot 3D model z uporabo jezika GKS in homogenih transformacij v prostoru.

1 Torzija linijskih nosilcev



Slika 1: Splošen prerez linijskega nosilca obremenjenega s torzijskim momentom M_t

Za izračun torzijskega vztrajnostnega momenta je potrebno izračunati Prandtlovo napetostno funkcijo $U(y,z)$ podano s Poissonovo parcialno diferencialno enačbo $\Delta U = 1$ ali

$$\frac{\partial^2 U}{\partial y^2} + \frac{\partial^2 U}{\partial z^2} = 1 \qquad (1)$$

z robnimi pogoji po konturi Γ_k

$$U_k = C_k \qquad k = 0, 1, 2, .., n \qquad (2)$$

Če imamo v notranjosti še luknje (slika 1) moramo zadostiti še dodatnim robnim pogojem

$$\oint_{\Gamma_k} \frac{\partial U_k}{\partial n_k} ds = -A_k \qquad k = 1, 2, ..., n \qquad (3)$$

Kjer je A_k površina znotraj in n_k normala. Za polne prereze si lahko na ograji Γ izberemo poljubno konstanto vendar se izračun pri numeričnem računanju poenostavi če izberemo $C_0 = U_\Gamma = 0$. Ob poznavanju funkcije $U(y,z)$ lahko izračunamo torzijski vztrajnostni moment

$$I_t = -4 \int_\Omega U d\Omega \qquad (4)$$

in s tem tudi strižni napetosti

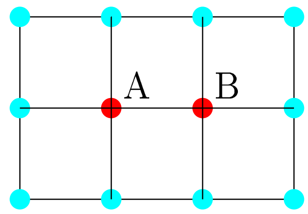
$$\tau_{xy} = +\tfrac{2M_t}{I_t} \tfrac{\partial U}{\partial z} \qquad (5)$$

$$\tau_{xz} = -\tfrac{2M_t}{I_t} \tfrac{\partial U}{\partial y} \qquad (6)$$

S poznavanjem I_t lahko tako izračunamo tudi maksimalne strižne napetosti kakor tudi specifičen zasuk.

2 Metoda končnih diferenc

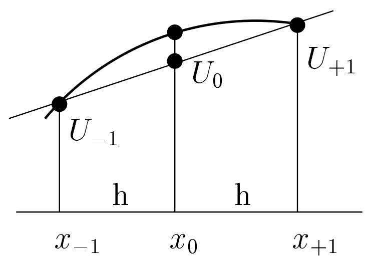
Enačbo (1) lahko v splošnem rešimo le numerično; Najenostavneje z uporabo ekvidistantne mreže. Na enostavnem primeru na sliki 2 je diskretiziran pravokotni prerez z ekvidistantno mrežo. Mreža vsebuje dve notranji in devet točk na robu. Iz pogoja (2) že vnaprej vemo, da je funkcija v teh točkah enaka nič in jih tako ni potrebno upoštevati pri izračunu. Diferencialno enačbo (1) je potrebno napisati le za notranji točki A in B.



Slika 2: Mreža z dvema notranjima točkama

2.1 Diskretizacija

Ker smo pri numeričnem izračunu najprej diskretizirali področje, smo dobili končno število točk, za katere mora veljati diferencialna enačba (1). Ker ne moremo zapisati točnih odvodov v točki se zadovoljimo s približki. Splošno enačbo za funkcijo drugega reda, ki gre skozi tri aproksimacijske točke zapišemo kot



Slika 3: Diskretizacija funkcije U v točki $x = 0$

$$U(x) = ax^2 + bx + c \qquad (7)$$

Koeficiente a, b in c določimo tako, da gre funkcija $u(x)$ v točkah $x = -h$, $x = 0$ in $x = -h$ zavzame vrednosti $U(-h) = U_{-1}$, $U(0) = U_0$ in $U(h) = U_{+1}$. Tako imamo tri enačbe iz katerih izračunamo koeficiente

$$a = \frac{U_{-1} - 2U_0 + U_{+1}}{2h^2} \qquad (8)$$

$$b = \frac{U_{+1} - U_{-1}}{2h} \qquad (9)$$

$$c = U_0 \qquad (10)$$

Če želimo izračunati odvod v točko $x = 0$ odvajamo enačbo (7)

$$\frac{dU(0)}{dx} = b = \frac{U_{+1} - U_{-1}}{2h} \tag{11}$$

$$\frac{d^2U(0)}{dx^2} = 2a = \frac{U_{-1} - 2U_0 + U_{+1}}{h^2} \tag{12}$$

Enako pot uberemo pri integraciji enačbe (7)

$$\begin{aligned} I &= \int_{-h}^{+h} (ax^2 + bx + c)dx \\ &= \left[\frac{ax^3}{3} + \frac{bx^2}{2} + cx\right]_{-h}^{+h} \\ &= \frac{h}{3}(U_{-1} + 4U_0 + U_{+1}) \end{aligned} \tag{13}$$

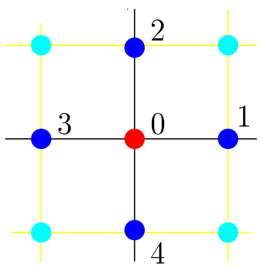
kar je znano kot *Simpson-ovo pravilo* za numerično integracijo.

2.2 Aproksimacija funkcije U

Pri dvodimenzionalnelnem numeričnem računu se v kartezijevem koordinatnem sistemu odvodi enačbe (1) računajo v posamezni smeri

$$\frac{\partial U}{\partial x} \doteq \frac{U_1 - U_{-1}}{2h} \tag{14}$$

$$\frac{\partial^2 U}{\partial x^2} \doteq \frac{U_1 - 2U_{-1} + U_{-1}}{h^2} \tag{15}$$



Slika 4: Aproksimacija $\Delta U = 1$ v točki

Za točko tako lahko aproksimiramo enačbo (1) kot

$$\Delta U = \frac{1}{h^2}(U_1 + U_2 + U_3 + U_4 - 4U_0) \tag{16}$$

Za primer na sliki 2 je potrebno tako napisati dve linearni enačbi za dve notranji točki

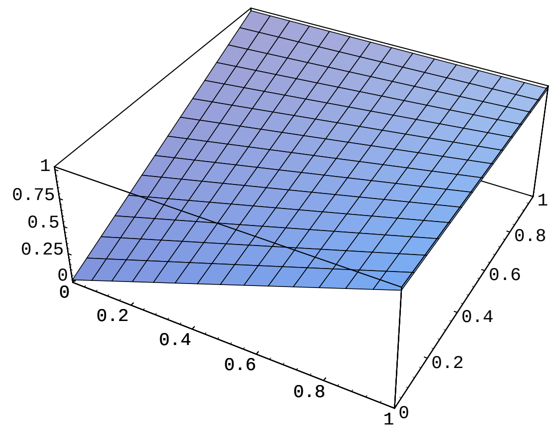
$$\begin{aligned} -4U_A + U_B &= h^2 \\ U_A - 4U_B &= h^2 \end{aligned}$$

kar da rešitev $U_A = U_B = -h^2/3$. Za večjo natančnost funkcije $U(y, z)$ bi bilo seveda potrebno zgostiti mrežo.

2.3 Integracija napetostne funkcije U

Integracija funkcije $U(y, z)$ je možna na več načinov. Iz enačbe (4) je razvidno, da je potrebno izračunati prostornino, ki jo ploskev U obdaja. Ker so naše osnovne ploskvice kvadrati si izberemo bilinearno funkcijo, ki je križni produkt dveh linearnih funkcij

$$\begin{aligned} U(s,t) = a_{11}st &+ a_{12}s \\ +a_{21}t &+ a_{22} \qquad t = 0..1,\ s = 0..1 \end{aligned} \tag{17}$$



Slika 5: Bilinearna funkcija s štirimi točkami

Za podane vrednosti $U(s, t)$ v robnih točkah lahko izračunamo splošne koeficiente a_{ij} kot sistem linearnih enačb

$$\begin{bmatrix} U(0,0) \\ U(0,1) \\ U(1,0) \\ U(1,1) \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix} \cdot \begin{bmatrix} a_{11} \\ a_{12} \\ a_{21} \\ a_{22} \end{bmatrix} \tag{18}$$

Matriko sistema invertiramo in napišemo enačbo

$$\begin{bmatrix} a_{11} \\ a_{12} \\ a_{21} \\ a_{22} \end{bmatrix} = \begin{bmatrix} 1 & -1 & -1 & 1 \\ -1 & 0 & 1 & 0 \\ -1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} U(0,0) \\ U(0,1) \\ U(1,0) \\ U(1,1) \end{bmatrix} \tag{19}$$

Za praktičen primer vzemimo funkcijo na sliki 5 s točkami $U(0,0) = 0$, $U(0,1) = 1$, $U(1,0) = 1$, $U(1,1) = 1$. Z izračunom enačbe (19) dobimo koeficiente $a_{11} = -1$, $a_{12} = 1$, $a_{21} = 1$, $a_{22} = 0$. Funkcija (17) je tako

$$U(s,t) = -st + s + t \qquad s = [0..1],\ t = [0..1] \tag{20}$$

Če želimo izračunati prostornino, ki jo ploskev U obdaja integriramo enačbo (17)

$$\begin{aligned} V &= \int_0^1 \int_0^1 U(s,t)dsdt \\ &= \left[\frac{a_{11}}{4}st + \frac{a_{12}}{2}s + \frac{a_{21}}{2}t + a_{22}\right]_0^1 \\ &= \frac{a_{11}}{4} + \frac{a_{12}}{2}s + \frac{a_{21}}{2} + a_{22} \end{aligned} \tag{21}$$

Iz enačbe (21) lahko razberemo to, da znamo izračunati volumen, ki ga obdaja ploskev, že če poznamo vrednosti v ogliščih. Potrebno je le zmnožiti enačbe (19), kar je v primeru naše bilinerarne funkcije enostavno

$$V = \frac{1}{4}[U(0,0) + U(0,1) + U(1,0) + U(1,1)] \tag{22}$$

Torzijski vztrajnosti moment za primer mreže na sliki 2 po enačbi (4) je z uporabo bilinearne funkcije enak

$$I_t = -4\left(4\frac{-h^2}{3\cdot 4} + 2\frac{-2h^2}{3\cdot 4}\right)h^2 = \frac{8}{3}h^4 = 2.7h^4 \tag{23}$$

Iz literature [6] razberemo, da je za razmerje stranic $3/2$ torzijski moment $I_t = 0.196ab^3 = 4.7h^4 \quad a > b$, kar nam daje slutiti, da bo za večje natančnosti potrebno zgostiti mrežo.

Seveda je natančneje, če za interpolacijsko funkcjo izberemo ploskev višjega reda, ki se razteza preko več točk. Podobno kot pri bilinerani funkciji naredimo križni produkt dveh paraboličnih funkcij z istim razponom parametrov s in t.

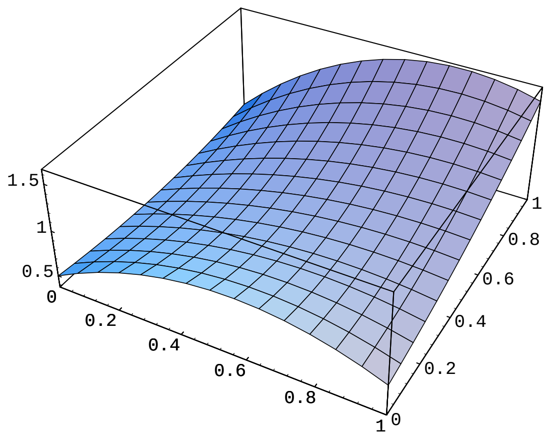
$$\begin{aligned} U(s,t) &= a_{11}s^2t^2 + a_{12}s^2t + a_{13}s^2 \\ &\quad + a_{21}st^2 + a_{22}st + a_{23}s \\ &\quad + a_{31}t^2 + a_{32}t + a_{33} \end{aligned} \tag{24}$$

ali zapisano v matrični obliki

$$U(s,t) = \mathbf{S}\mathbf{C_u}\mathbf{T}^T \tag{25}$$

kjer so $S = [s^2\, s\, 1]$, $T = [t^2\, t\, 1]$ in

$$C_u = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$$



Slika 6: Bikvadratična funkcija z devetimi točkami

Podobno kot v prejšnem primeru je potrebno izračunati koeficiente matrike C_u. Imamo devet točk $U(s,t)$; Zapišemo devet linearnih enačb $\mathbf{U} = \mathbf{M}\cdot\mathbf{a}$

$$\begin{bmatrix} U(0,0) \\ U(0,\frac{1}{2}) \\ U(0,1) \\ U(\frac{1}{2},0) \\ U(\frac{1}{2},\frac{1}{2}) \\ U(\frac{1}{2},1) \\ U(1,0) \\ U(1,\frac{1}{2}) \\ U(1,1) \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{4} & \frac{1}{2} & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & \frac{1}{4} & 0 & 0 & \frac{1}{2} & 0 & 0 & 1 \\ \frac{1}{16} & \frac{1}{8} & \frac{1}{4} & \frac{1}{8} & \frac{1}{4} & \frac{1}{2} & \frac{1}{4} & \frac{1}{2} & 1 \\ \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & \frac{1}{2} & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ \frac{1}{4} & \frac{1}{2} & 1 & \frac{1}{4} & \frac{1}{2} & 1 & \frac{1}{4} & \frac{1}{2} & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \cdot \begin{bmatrix} a_{11} \\ a_{12} \\ a_{13} \\ a_{21} \\ a_{22} \\ a_{23} \\ a_{31} \\ a_{32} \\ a_{33} \end{bmatrix} \tag{26}$$

Sistem enačb $\mathbf{a} = \mathbf{M}^{-1}\mathbf{U}$ rešimo z invertiranjem matrike $\mathbf{M}$

$$M^{-1} = \tag{27}$$
$$\begin{bmatrix} 4 & -8 & 4 & -8 & 16 & -8 & 4 & -8 & 4 \\ -6 & 8 & -2 & 12 & -16 & 4 & -6 & 8 & -2 \\ 2 & 0 & 0 & -4 & 0 & 0 & 2 & 0 & 0 \\ -6 & 12 & -6 & 8 & -16 & 8 & -2 & 4 & -2 \\ 9 & -12 & 3 & -12 & 16 & -4 & 3 & -4 & 1 \\ -3 & 0 & 0 & 4 & 0 & 0 & -1 & 0 & 0 \\ 2 & -4 & 2 & 0 & 0 & 0 & 0 & 0 & 0 \\ -3 & 4 & -2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Prostornino, ki jo obdaja ploskev izračunamo z integriranjem enačbe (24)

$$\begin{aligned} V &= \int_0^1\int_0^1 U(s,t)dsdt \\ &= \frac{a_{11}}{9} + \frac{a_{12}}{6} + \frac{a_{13}}{3} \\ &\quad + \frac{a_{21}}{6} + \frac{a_{22}}{4} + \frac{a_{23}}{2} \\ &\quad + \frac{a_{31}}{3} + \frac{a_{32}}{2} + a_{33} \end{aligned} \tag{28}$$

Za robne točke $U(s,t)$ na sliki 6 lahko izračunamo sledeče koeficiente a_{ij} in z njimi prostornino $V = 0.9222$

$$\begin{bmatrix} U(0,0) \\ U(0,\frac{1}{2}) \\ U(0,1) \\ U(\frac{1}{2},0) \\ U(\frac{1}{2},\frac{1}{2}) \\ U(\frac{1}{2},1) \\ U(1,0) \\ U(1,\frac{1}{2}) \\ U(1,1) \end{bmatrix} = \begin{bmatrix} 0.5 \\ 0.4 \\ 0.5 \\ 1.0 \\ 0.9 \\ 1.5 \\ 0.7 \\ 1.0 \\ 1.5 \end{bmatrix} \qquad \begin{bmatrix} a_{11} \\ a_{12} \\ a_{13} \\ a_{21} \\ a_{22} \\ a_{23} \\ a_{31} \\ a_{32} \\ a_{33} \end{bmatrix} = \begin{bmatrix} -4 \\ 3.6 \\ -1.6 \\ 4 \\ -2.8 \\ 1.8 \\ 0.4 \\ -0.4 \\ 0.5 \end{bmatrix} \tag{29}$$

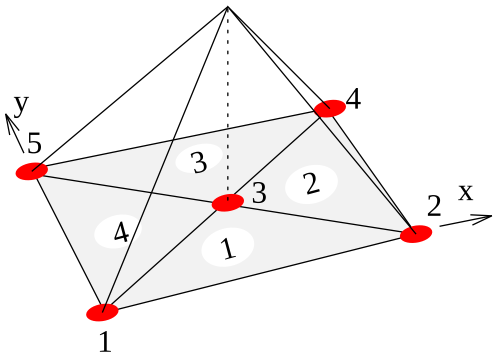
Pri aproksimaciji funkcije z bikvadratično funkcijo je potrebno paziti, da lahko celotno področje, ki je sestavljeno iz osnovnih ploskvic, popišemo z osnovno ploskvico. To konkretno pomeni, da moramo imeti v obeh smereh sodo število intervalov. V primeru na sliki 2 to ne velja, lahko pa si pomagamo tako, da večino ploskvic izračunamo z bikvadratično aproksimacijo, za preostale pa uporabimo bilinearno.

3 Izračun I_t z metodo MKE

Pri primerih izračuna prostornine, ki jo obdaja ploskev smo videli, da je možno točno izračunati interpolirano prostornino, že če smo poznali vrednosti funkcije U v vozliščih. Problem pri MKD je prav ta, da je potrebno funkcijo $\Delta U = 1$ zadostiti z diskretizacijo, ki se napenja preko večih točk in povezuje posamezne osnovne elemente (ploskvice).

Pri metodi MKE so kot osnovne neznanke vrednosti napetostne funkcije U_e v vozliščih. Zveza med $U(x,y)$ v elementu in vrednostmi funkcije v vozliščih pa se vzpostavlja z interpolacijskimi funkcijami $\mathbf{N}(\mathbf{x},\mathbf{y})$.

$$U(x,y) = \mathbf{N}(x,y)\mathbf{U}_e \tag{30}$$



Slika 7: Mreža štirih trikotnih elementov in funkcija U

Za osnovo in zgled MKE izberimo pravokotnik dimenzije 3×2 na sliki 7 in najenostavnejši dvodimenzionalni trikotni element CST (*Constant Strain Triangle*) za katerega so interpolacijske funkcije

$$N_i = \frac{1}{2\Delta}(a_i + b_i + c_i) \qquad i = 1,2,3 \tag{31}$$

Kjer 2Δ predstavlja dvakratno površino trikotnika

$$2\Delta = \begin{vmatrix} 1 & x_1 & y_1 \\ 1 & x_2 & y_2 \\ 1 & x_3 & y_3 \end{vmatrix} = \begin{matrix} x_1y_2 - x_2y_1 \\ +x_2y_3 - y_2x_3 \\ +x3_y1 - x_1y_3 \end{matrix} \tag{32}$$

izrazi a_i, b_i in c_i pa so za prvi element enaki

$$\begin{aligned} a_1 &= x_2y_3 - x_3y_2 = 3 \cdot 1 + 1.5 \cdot 0 = 3 \\ b_1 &= y_2 - y_3 = 0 - 1 = -1 \\ c_1 &= x_3 - x_2 = 1.5 - 3 = -1.5 \\ a_2 &= x_3y_1 - x_1y_3 = 1.5 \cdot 0 - 0 \cdot 1.5 = 0 \\ b_2 &= y_3 - y_1 = 1 - 0 = 1 \\ c_2 &= x_1 - x_3 = 0 - 1.5 = -1.5 \\ a_3 &= x_1y_2 - x_2y_1 = 0 \cdot 3 - 3 \cdot 0 = 0 \\ b_3 &= y_1 - y_2 = 0 - 0 = 0 \\ c_3 &= x_2 - x_1 = 3 - 0 = 3 \end{aligned} \tag{33}$$

Enačbo elementa CST lahko zapišemo kot

$$\begin{bmatrix} b_1^2 + c_1^2 & b_1b_2 + c_1c_2 & b_1b_3 + c_1c_3 \\ b_1b_2 + c_1c_2 & b_2^2 + c_2^2 & b_2b_3 + c_2c_3 \\ b_1b_3 + c_1c_3 & b_2b_3 + c_2c_3 & b_3^2 + c_3^2 \end{bmatrix} \begin{bmatrix} U_1 \\ U_2 \\ U_3 \end{bmatrix} = -\frac{4\Delta^2}{3} \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \tag{34}$$

Za prvi element je $U_1 = U_2 = 0$ in enačba elementa

$$\begin{bmatrix} 3.25 & 1.25 & -4.5 \\ 1.25 & 3.25 & -4.5 \\ -4.5 & -4.5 & 9 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ U_3 \end{bmatrix} = -\frac{4 \cdot 1.5^2}{3} \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

Podobno je za drugi element $U_2 = U_4 = 0$ in enačba elementa

$$\begin{bmatrix} 3.25 & -1.25 & -2 \\ -1.25 & 3.25 & -2 \\ -2 & -2 & 4 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ U_3 \end{bmatrix} = -\frac{4 \cdot 1.5^2}{3} \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

Za štiri končne elemente smo dobili štiri enačbe za vrednost funkcije U_3, kar pomeni, da je sistem predoločen. Če bi bili vsi elementi enake oblike, bi bile enačbe za točko 3 enake in bi uporaba ene enačbe zadovoljila tudi ostale tri. Žal v našem primeru to ni tako, zato se odločimo za izračun U_3 tako, da bo izračunana vrednost U_3 v povprečju zadovoljila vse enačbe. To naredimo tako, da vse enačbe za točko 3 seštejemo in rešimo. S tem dobimo povprečno vrednost U_3. Iz navedenega lahko sklepamo, da lahko po izračunu vrednosti v točkah enostavno ocenimo napako v vozliščih in primerno ukrepamo z zgostitvijo mreže na kritičnih mestih. Glavna slabost elementov CST je občutljivost na deformacije. Priljubljeni pa so zaradi enostavnosti in prilagodljivosti na modelirano obliko.

Ob upoštevanju enačb vseh štirih elementov dobimo skupno enačbo za točko 3, ki je edina neznanka sistema

$$26U_3 = -4\frac{4 \cdot 1.5^2}{3}$$

Za izračun I_t je potrebno izračunati prostornino vseh tristranih prizem, ki sestavljajo mrežo končnih elementov. Prostornina poljubno prisekane tristrane prizme z vrednostmi v ogljiščih a, b in c in površino osnovne ploskve Δ se izračuna kot

$$V = \frac{\Delta}{3}(a + b + c) \tag{35}$$

Torzijski moment za primer na sliki 7 je tako

$$I_t = -4(4\frac{1.5}{3}[-0.46 + 0 + 0]) = 3.69 \tag{36}$$

kar je v primerjavi z metodo MKD, kjer smo za isti prerez uporabili dve notranji točki, veliko bolje vendar še vedno ne dovolj natančno. Daje pa razultat slutiti, da metoda MKE veliko hitreje konvergira k točni rešitvi.

4 Konstrukcija prereza

S programom Autocad bomo narisali prereze, ki jih bomo kasneje izračunali. V Autocadu bomo uporabljali le lomljenke `POLYLINE`. Narisani lik mora biti zaključen z ukazom `Close`. Krivulja narisana s `PLINE` je lahko tudi tipa *Spline*, ne pa tudi tipa *Fit* ali vključenimi loki `ARC`.

S programom v Autolispu konturo prereza zapišemo v datoteko `points.txt`, ki jo uporabimo pri generiranju mreže.

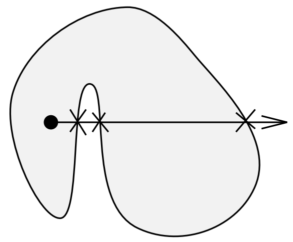
Datoteka ima v prvi vrstici zapisano število točk, nato pa sledijo v vsaki vrstici koordinata točke x in y.

Program v lispu se naloži v Autocad z ukazom `(load "torzija")` ter ga izvedemo z ukazom `torzija`. Če program preimenujemo v `acad.lsp` se bo ob vsakem zagonu Autocad-a avtomatsko naložil.

5 Generiranje mreže

Glavno delo seveda ni generiranje mreže, ampak sestavljanje sistema linearnih enačb. Ker je prerez mnogokotnik poljubne oblike je za generiranje mreže pomembno, da najprej ugotovimo meje v katerih se prerez nahaja. Za vnaprej izbrano število delitev ene izmed osi določimo velikost osnovne delitve in s osnovno mrežo točk. Sistem enačb zapišemo le za notranje točke, vse ostale pa so znane in imajo vrednost nič.

5.1 Kontrola notranjosti lika



Slika 8: Ugotavljanje notranjosti lika

Nortranjost lika ugotovimo tako, da za izbrano točko (x, y) štejemo koliko daljic od vseh možnih prereže poltrak desno od točke. Če je število takih prerezov liho potem je točka v notranjosti lika, sicer je v zunanjosti.

6 Reševanje sistema enačb

Sistem linearnih enačb lahko rešujemo na različne načine. Klasična Gaussova eliminacijska metota se pri konkretnih problemih pokaže kot počasna (N^3 operacij). Predlagana metoda, ki se v praksi tudi največ uporablja za reševanje sistema enačb je Lower/Upper dekompozicija, ki ima časovno zahtevnost $1/3N^3$. Reševanje sistema po tej metodi se sestoji iz dveh korakov:

1. **decomposition** razdeli matriko **M** na dve matriki (zgornja / spodnja), katerih produkt je **M**.) Obe matriki sta shranjeni v matriki M, le da je zgornji del matrike **M** matrika **U**, spodnji pa matrika **L**.
2. **backsubstitution** množi desno stran enačbe z zgornjo matriko in pri tem izračuna neznane linearne spremenljivke.

6.1 Lower/Upper dekompozicija

Rešujemo sistem enačb velikosti N. Matrika A predstavlja levo stran sistema enačb. Pri fortranu so lahko matrike le statične in zato parameter NP opisuje velikost matrike A, ki je običajno večja kot pa sistem enačb ($N < NP$). INDX je celoštevilčni vektor permutacji v matriki A in se prenaša naprej tako kot parameter D v podprogram *LUBKSB*, kateri zahteva še desno stran sistema enačb v vektorju B. Po izračunu se rezultat nahaja v vektorju B.

Potek izračuna sistema linearnih enačb bi potekal nekako takole:

```
parameter (NP=100)
real a(NP,NP), dQ(NP), d
integer indx(NP)
...
...
n = pipes
...
...
call ludcmp(a, n, NP, indx, d)
call lubksb(a, n, NP, indx, dQ)
...
```

7 Grafični prikaz enačbe U

Za grafičen prikaz rezultatov v programu uporabite grafični jezik GKS [3]. V ta namen uporabite homogene matrike in homogene transformacije.

Omogočati mora tudi ogled sistema s poljubnega zornega kota (rotacija, translacija in skaliranje v homogenih koordinatah).

Osnovni program, ki demonstrira uporabo jezika GKS je:

```
      program GKS skeleton
      implicit none
      real x(3)  /-200.0, 350.0, 250.0/
      real y(3)  /-200.0, 300.0,-250.0/
      real kx(2) /0.0, 100.0/
      real ky(2) /50.0, 120.0/
      character*80 datrec(1)

c     Odpre GKS (Gks OPen gKS)
      call gopks()

c     Nastavitev zaslonskih koordinat
c     (Gks Set WorKstation ViewPort)
      call gswkvp(1, 20.0, 50.0, 600.0, 450.0)

c     Nastavitev uporabniskega koord. sistema
c     (Gks Set WiNdow)
      call gswn(1, -300.0, -300.0, 400.0, 400.0)

c     Izris lika z dvema crtama
c     (GksSetPolyLineColorIndex, GksPolyLine)
c     Indeksi:1 modra, 2 zelena, 3 cyan, 4 rdeca,
c              5 violicna, 6 rjava, 7 bela, 8 siva,
c              9 svetlomodra, ..., 14 rumena, 15
      call gsplci(3)
      call gpl(3, x, y)

c     Izris teksta (GksSetCharacterHeight,
c     ...TextColorIndex, GksTeXt)
      call gstxci(14)
      call gschh(20)
      call gtx(-250.0, 300.0, 'GKS skeleton')

c     Izris kroga
c     (Generalized Drawing Primitive)
      call gdp(2, kx, ky, -1, 1, datrec)

c     Zapre GKS (Gks CLose gKS)
      call gclks()
      end
```

8 Poročilo

Poročilo se oddaja v dveh delih, prvi za numerični program, drugega za grafični del. V poročilu opišite problem, ki ga rešujete (zahteve), podajte teoretične osnove in bistvene dele rešitve, ki ste jih uporabili v programu.

Izpisa programa ni potrebno prilagati poročilu, potrebno pa je predstaviti pomembne dele programa na način, ki čimbolj jasno predstavlja rešitev. To je lahko z besednim opisom, diagramom poteka ali z delom programske kode. Priložite tudi rezultate izračuna prerezov, ki ste jih izračunali in analizo konvergence izračuna

Celotno poročilo naj ima klasično obliko (uvod, zahteve, teoretične osnove, problemi, rezultati, možne izboljšave, zaključek, literatura, priloge). Izberite si primer in na njem pokažite reševanje problema. Diskutirajte izboljšave programa, kot tudi omejitve, ki so posledica vgrajenih algoritmov (velikost matrik, stabilnost, ...). Torej: poročilo mora biti napisano tako, da je možno iz njega rekonstrurirati program.

Pišite čitljivo; priporočen je izpis s tiskalnikom. Ovojnico za poročilo dobite pri vratarju (Računalniško podprto konstruiranje).

9 Delo na računalniku

Na računalniku PC je instalirano programsko okolje s fortranskim prevajalnikom po standardu F77. Okolje je integrirano in se imenuje *Programmers WorkBench*. Integrirano okolje z požene z ukazom pwb ime.for v katerem lahko program popravljamo, prevajamo, zaganjamo in iščemo napake. Osnovni ukazi so dosegljiv s pritiskom na tipko skupaj s tipko **ALT**. Tako je izhod iz **PWB ALT+F,X**, Prevajanje in zaganjanje programa z ukazom **Alt+R,E,B,R**. Za pomoč lahko uporabite tipko **F1**, ki je kontekstno občutljiva. Poleg okolja PWB je možno uporabiti tudi zmogljivejši 32-bitni prevajalnik *Fortran Power Station*, ki nima spominskih omejitev pri matrikah, žal pa deluje le v sistemu *Windows*, zato se priporoča najprej uporaba PWB za majhne sisteme, kasneje pa prevajanje s FPS za večje probleme in hitrejše izvajanje.

10 Fortran

Tu bom prikazal standardne posebnosti v Fortranu, ki jih boste s pridom uporabili pri programiranju. Priporoča se tipkanje programa z malimi črkami, saj je tako kodiran program bolj čitljiv.

Formatne stavke vključene lahko vključimo v ukaz `read` ali `write`:

```
character ime*65
write(*, '('' Ime vhodne datoteke: '', $)' )
read(*,'(A)') ime
```

Datoteke delimo na tekstovne in binarne datokeke. Direktna datoteka je poseben primer binarne datoteke, ki ima fiksno dolžino zapisa (record). Fortran omogoča branje/pisanje v prostem formatu (*), uporabniškem formatu (format()) in brez formata (samo v binarne datoteke). Neformatirano direktno datoteko z dolžino enega zapisa 58 bytov se odpre z ukazom

```
  open(1, 'results.bin', status='NEW',
* access='DIRECT', form='UNFORMATTED',
* recl=58)
```

Dolžina enega zapisa RECL se določa z maksimalno velikostjo zapisa v datoteki. Zapisi imajo lahko različne vsebine podatkov, dolžina zapisa pa je fiksna zaradi hitrejšega določanja pozicije datotečnega kazalca. Določi se jo s seštevkom dolžine vseh spremenljivk in upoštevanjem, da zavzame realna spremenljivka 4 byte, celoštevilčna 2, znakovna pa toliko, kolikor znakov ima rezerviranih.

Za branje in pisanje enote se uporablja standardni read/write stavek brez podatka o formatu.

```
write(1, rec=2) (x(i), i = 1, n/2)
read(1, rec=2)  (x(i), i = 1, n/2)
```

Fortran omogoča pisanje in branje v svoje interne enote v obliki `character` polja. To nam omogoča pretvarjanje numeričnih spremenljivk v znakovne. Največkrat se to uporablja za podprograme, ki zahtevajo kot parameter znakovno spremenljivko, na voljo pa imamo le numerično spremenljivko. Tipičen primer uporabe je izpis teksta v jeziku GKS. Primer:

```
character cnode*3
integer node
...
write(cnode, '(I3)') node
call gtx(x, y, cnode)
end
```

11 Splošna priporočila za delo

Noben program ne deluje na mah. Ker je problem kompleksen, se ga je potrebno lotiti sistematično. Posamezne dele program moramo sproti kontrolirati. Pomembno je tudi dokazovanje pravilnosti delovanja, kjer si logično razlagamo obnašanje programa za različne vhodne parametre. Švejkova metoda pri problemih, kot je naš, nikoli ni bila uspešna.

Fortran ima to (ne)ugodno lastnost, da so vse premenljivke ob začetku postavljene na nič. Po končanem delu brišite z (`del`) vse nepotrebne datoteke, ki jih imate na svojem področju.

Literatura

[1] Dragan Domazet. *Programiranje računarske grafike pomoću GKS sistema.* Naučna knjiga, Beograd, 1989.

[2] Divjak J. Dovedan Z. *Fortran 77 s tehnikama programiranja.* Zveza organizacij za tehnično kulturo Slovenije, Ljubljana, 1988.

[3] Kansy K. Plaff G. Enderle G. *Computer Graphics Programming.* Springer-Verlag, New York, 1984.

[4] William H. et.al. *Numerical Recipes in Fortran; The Art of Scientific programing.* Cambridge university press, Cambridge, 1989.

[5] Miodrag Sekulović. *Metod konačnih elemenata.* IRO, Beograd, 1988.

[6] Marko ŠKerlj. *Mehanika - Trdnost.* Fakulteta za strojništvo, Ljubljana, 1984.

Laboratorij za CAD, Soba N-17. Govorilne ure za študente so od ponedeljka do četrtka od 13^{00} do 14^{00}. Prosimo, da se držite razporeda urnika. Nadomeščanje zamujenih laboratorijskih vaj načeloma ni možno zaradi zasedenosti računalniške učilnice. Pred vajami na računalniku si pripravite gradivo za delo, da ne boste gledali v zrak (k sosedu).

Ljubljana 9. oktober 1995

Typeset in LaTeX