ALGORITMUS PRO VÝPOČET DEPLANAČNÍ FUNKCE A MOMENTU TUHOSTI VE VOLNÉM KROUCENÍ OBDÉLNÍKOVÉHO PRŮŘEZU METODOU SÍTÍ

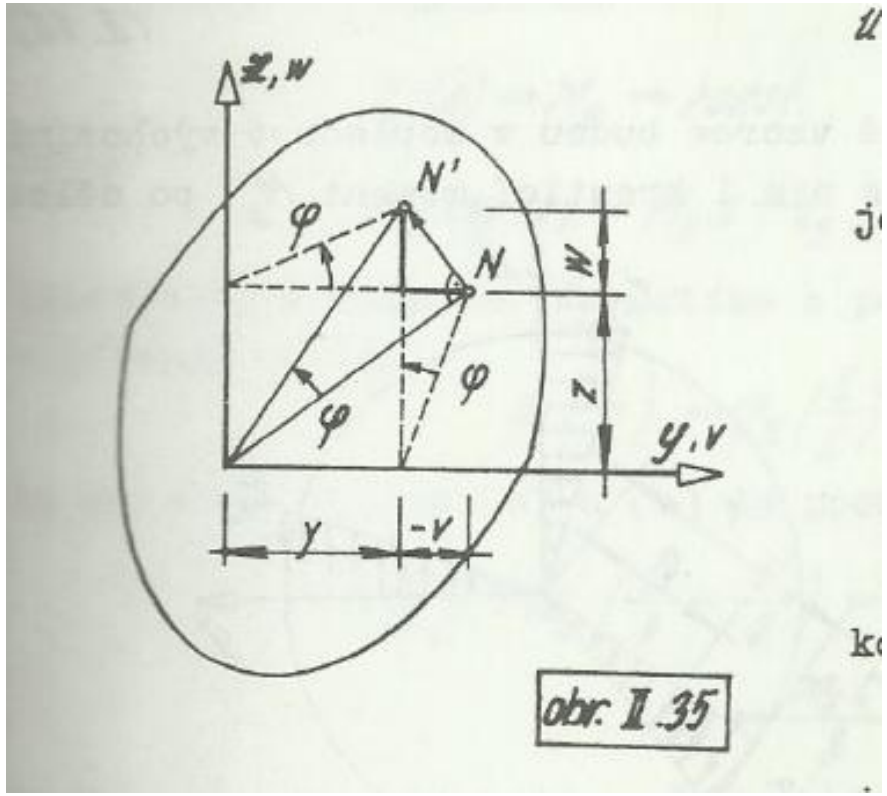
Ondřej Faltus, ZS 2015/16

VOLNÉ KROUCENÍ

Předpoklady

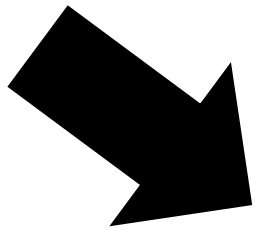
- nosník zatížený pouze kroutícím momentem => $M_X = const.$
- $\varepsilon_x = \varepsilon_y = \varepsilon_z = \gamma_{yz} = 0$
- Posunutí (viz obr.):
- $v = -\varphi(x)z$
- $w = \varphi(x)y$
- $u = \theta(x)\psi(y,z)$
- relativní úhel zkroucení $\theta = \frac{d\varphi(x)}{dx}$

deplanační funkce



Vztahy pro napětí

- $\gamma_{xy} = \frac{\partial u}{\partial y} + \frac{\partial v}{\partial x} = \frac{\partial u}{\partial y} - \theta z$
- $\gamma_{xz} = \frac{\partial u}{\partial z} + \frac{\partial w}{\partial x} = \frac{\partial u}{\partial z} + \theta y$



- $\tau_{xy} = G\gamma_{xy} = G\theta\left(\frac{\partial \psi}{\partial y} - z\right)$
- $\tau_{xz} = G\gamma_{xz} = G\theta\left(\frac{\partial \psi}{\partial z} + y\right)$

Laplaceova rovnice deplanační funkce

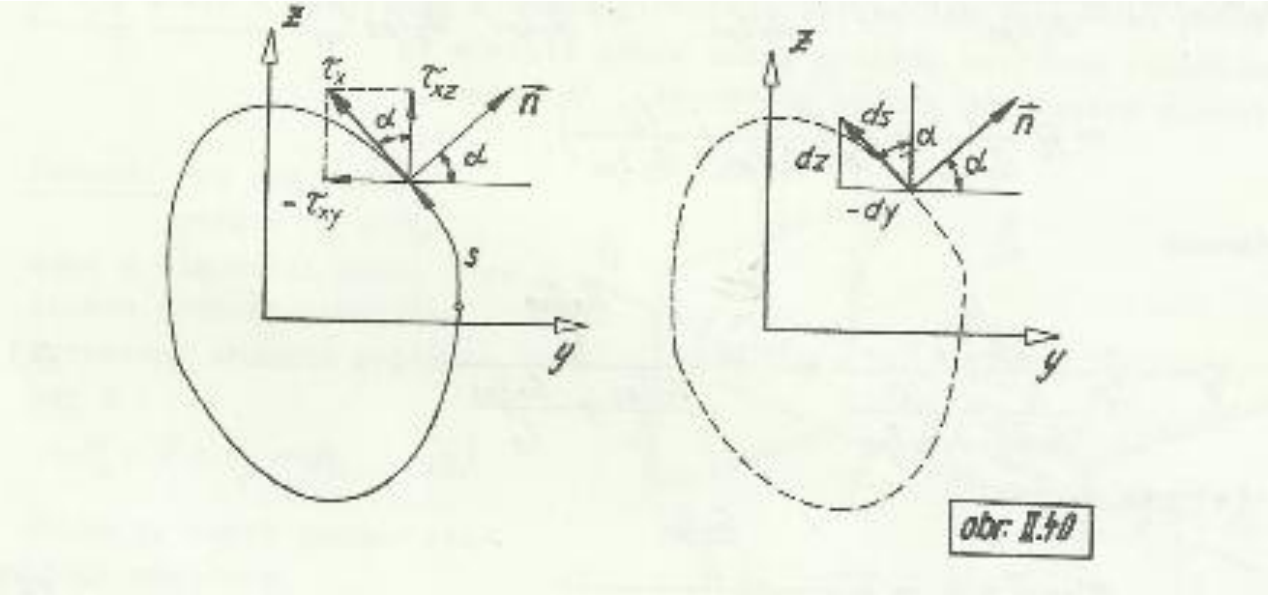
- podmínka rovnováhy:

$$\frac{\partial \tau_{xy}}{\partial y} + \frac{\partial \tau_{xz}}{\partial z} = 0$$

$$\frac{\partial}{\partial y}\left(G\theta\left(\frac{\partial \psi}{\partial y} - z\right)\right) + \frac{\partial}{\partial z}\left(G\theta\left(\frac{\partial \psi}{\partial z} + y\right)\right) = 0$$

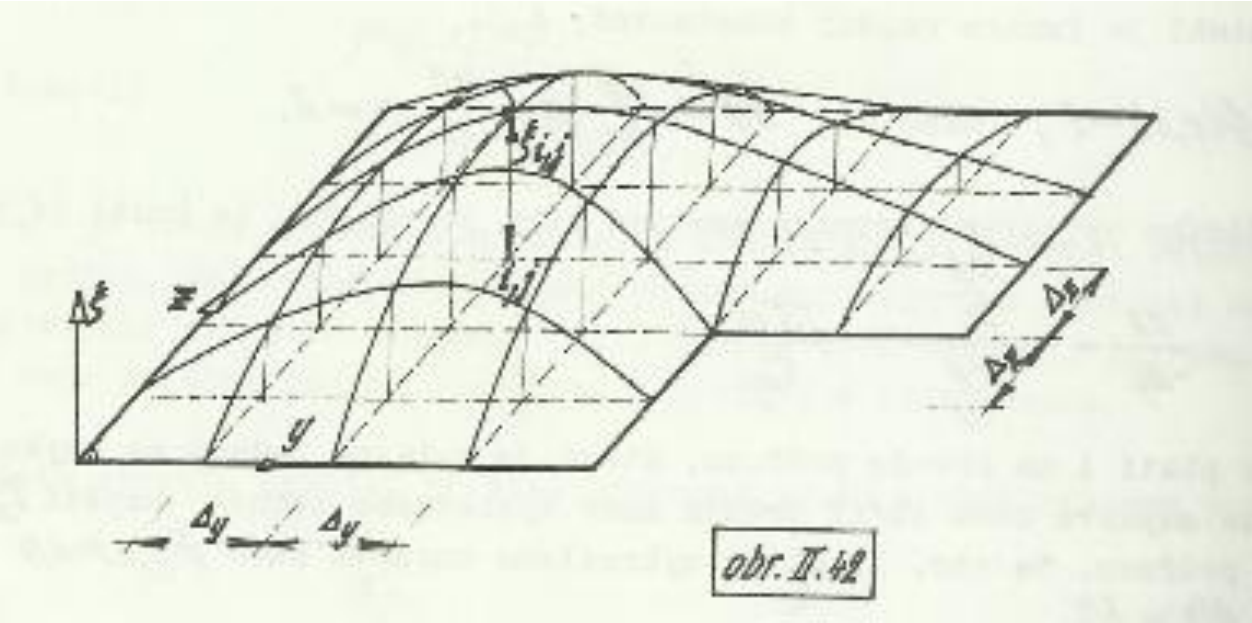
$$\frac{\partial^2 \psi}{\partial y^2} + \frac{\partial^2 \psi}{\partial z^2} = 0$$

Okrajová podmínka



$$\frac{\partial \psi}{\partial \vec{n}} = zn_y - yn_z$$

Metoda sítí



- Rozdělení oblasti mřížkou na obdélníčky o stranách konečné délky
- Pro každý bod (včetně okrajových) náhrada Laplaceovy rovnice pomocí vztahu z Taylorova rozvoje (druhého řádu)

$$\frac{\psi_{i+1,j} - 2\psi_{i,j} + \psi_{i-1,j}}{(\Delta y)^2} + \frac{\psi_{i,j+1} - 2\psi_{i,j} + \psi_{i,j-1}}{(\Delta z)^2} = 0$$

- fiktivní body po stranách – navážou se rovnicí vzatou z okrajové podmínky

$$\frac{\psi_{ext} - \psi_{int}}{2\Delta n} = zn_y - yn_z$$

Soustava lineárních rovnic = práce pro algoritmický řešič. Čím více rovnic, tím přesnější odhad výsledku.

PROGRAM

Specifikace

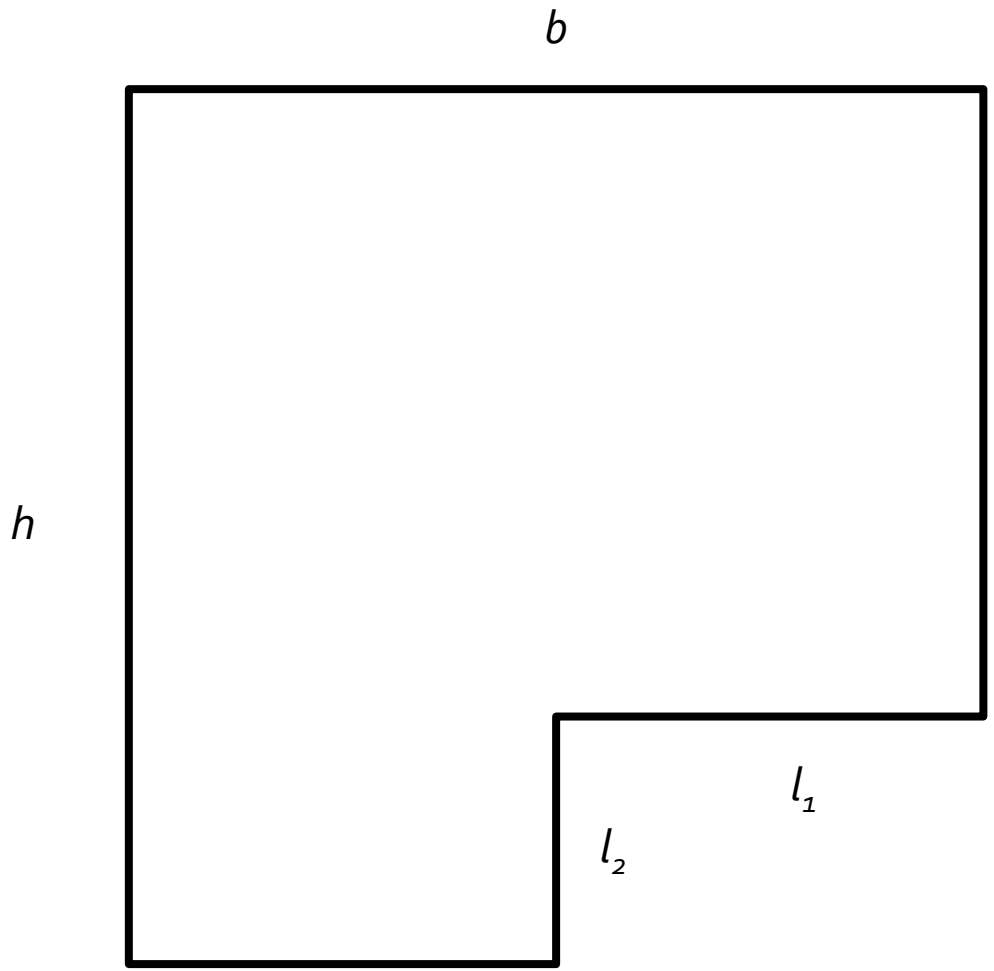
- desktopová aplikace v jazyce Java
- počítá deplanační funkci, hodnoty napětí a moment tuhosti v kroucení
- obdélníkové průřezy (s výkusem – tvar L)
- zobrazení grafického výstupu

Problém

1. přijetí vstupních dat
2. sestavení soustavy rovnic
3. řešení rovnic pro získání hodnot deplanační funkce
4. výpočet zbylých proměnných
5. grafická interpretace výsledku

Vstupní data

- modul pružnosti ve smyku G
- relativní zkroucení průřezu ϑ
- rozměry průřezu b, h, l_1, l_2
- přesnosti rozdělení



Sestavení rovnic

- pole bodů
- každý bod zná svůj typ (vnitřní, západní, severní etc.) – viz ukázka
- každý bod zná své pořadí v rovnicích (index neznámé)
- bod nejblíže těžišti – speciální typ, jeho rce je jen $\psi=0$

- bod vytvoří podle typu rovnici, předá ji poli, to je seřadí do matice a předá řešiči

Sestavení rovnic – ukázka kódu

```
else if (type == INTERNAL) {
  rightSide = 0.0;
  double coefHere = (-2 / (field.getDeltaB() * field.getDeltaB())) + (-2 / (field.getDeltaH() * field.getDeltaH()));

  int indexWest = field.getEquationIndexAt(gridPosY + 1, gridPosZ);
  int indexEast = field.getEquationIndexAt(gridPosY - 1, gridPosZ);
  int indexNorth = field.getEquationIndexAt(gridPosY, gridPosZ - 1);
  int indexSouth = field.getEquationIndexAt(gridPosY, gridPosZ + 1);

  double coefWest = 1 / (field.getDeltaB() * field.getDeltaB());
  double coefEast = coefWest;
  double coefNorth = 1 / (field.getDeltaH() * field.getDeltaH());
  double coefSouth = coefNorth;

  variables.set(0, equationIndex, coefHere);
  variables.set(0, indexWest, coefWest);
  variables.set(0, indexEast, coefEast);
  variables.set(0, indexNorth, coefNorth);
  variables.set(0, indexSouth, coefSouth);
```

Sestavení rovnic – ukázka kódu

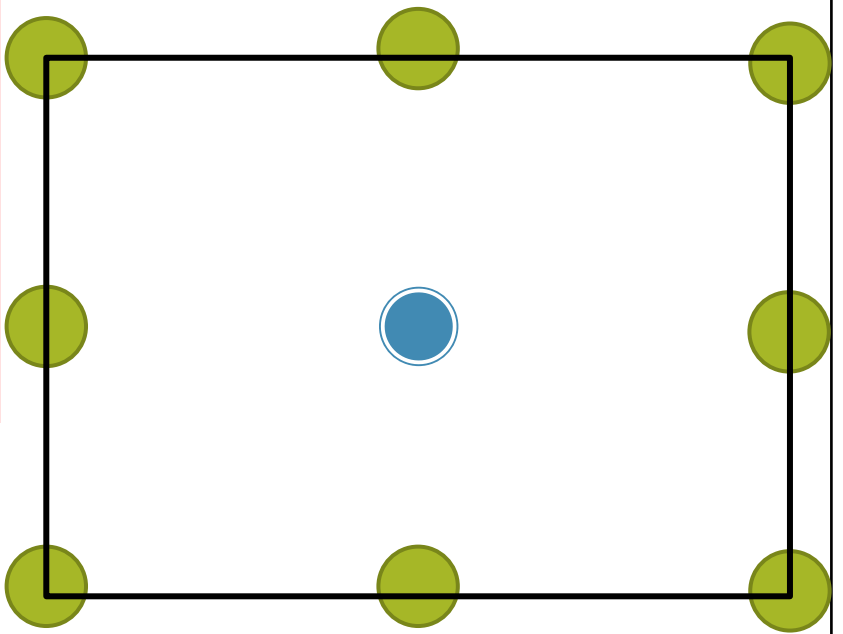
```
} else if (type == BORDER_WEST) {
    double relativeZ = coordinates.getZ() - field.getShape().getCentroid().getZ();

    int indexInternal = field.getEquationIndexAt(gridPosY - 2, gridPosZ);
    int indexExternal = equationIndex;

    double coefExternal = +1 / (2 * field.getDeltaB());
    double coefInternal = -1 / (2 * field.getDeltaB());

    rightSide = relativeZ;

    variables.set(0, indexInternal, coefInternal);
    variables.set(0, indexExternal, coefExternal);
```



Řešící algoritmus

- Cramerovo pravidlo

-velká složitost: *(n+1)!* iterací funkce determinantu pro soustavu o *n* neznámých

-hodí se tak do 20 neznámých, potom jsou časy a nároky astronomické

- Gaussova eliminace

-používám neúplnou (eliminuje jen pod diagonálou, pak zpětně dopočítává neznámé odzadu)

-problémy okolo 100 000 neznámých (paměť)

Dopočet hodnot napětí

- napětí se spočte pro každý bod podle vztahů dole - derivace nahrazena vztahem:

$$\frac{\partial\psi}{\partial y} = \frac{\psi_{west} - \psi_{east}}{2\Delta y}, \frac{\partial\psi}{\partial z} = \frac{\psi_{south} - \psi_{north}}{2\Delta z}$$

- $\tau_{xy} = G\gamma_{xy} = G\theta\left(\frac{\partial\psi}{\partial y} - z\right)$
- $\tau_{xz} = G\gamma_{xz} = G\theta\left(\frac{\partial\psi}{\partial z} + y\right)$

```
public void computeTensions() {
    if (!isAssignedEquation()) {
        return;
    }
    if (type == INTERNAL || type == ZERO_STATED) {
        double g = field.getG();
        double theta = field.getTheta();
        double relativeY = coordinates.getY() - field.getShape().getCentroid().getY();
        double relativeZ = coordinates.getZ() - field.getShape().getCentroid().getZ();

        double psiSouth = field.getPointAt(gridPosY, gridPosZ + 1).getPsiValue();
        double psiNorth = field.getPointAt(gridPosY, gridPosZ - 1).getPsiValue();
        double psiWest = field.getPointAt(gridPosY + 1, gridPosZ).getPsiValue();
        double psiEast = field.getPointAt(gridPosY - 1, gridPosZ).getPsiValue();

        double dPsiSlashdY = (psiWest - psiEast) / (2 * field.getDeltaB());
        double dPsiSlashdZ = (psiSouth - psiNorth) / (2 * field.getDeltaH());

        tauYValue = g * theta * (dPsiSlashdY - relativeZ);//melo by byt -z
        tauZValue = g * theta * (dPsiSlashdZ + relativeY);//melo by byt +y
    } else {
        tauYValue = 0.0;
        tauZValue = 0.0;
    }
}
```

Dopočet momentu tuhosti v kroucení

- Moment tuhosti v kroucení je dán vztahem:

$$I_K = \iint_A \left(y^2 + z^2 + \frac{\partial \psi}{\partial z} y - \frac{\partial \psi}{\partial y} z \right) dA$$

- Integrál přejde na sumu konečného počtu bodů
- Každý bod vypočte svůj příspěvek k I_K a bodové pole je všechny sečte (opět se použijí náhrady derivací viz výše)
- rohové a postranní body započítají pouze příslušnou část elementárního obdélníku

Vizualizace výsledku

- velikosti funkcí dvou proměnných jsou reprezentovány barvami (maximální, minimální, nulová)
- každý bod pole si pamatuji svou pozici
- v místech mezi jednotlivými body určuji funkci lineární interpolací
- konkrétní hodnoty zobrazeny v „tooltipu“

PŘÍKLAD

Zadání a výsledek

- mějme obdélník se stranami $b = 1\mathrm{m}, h = 0{,}5\mathrm{m}$
- zajímá nás jeho moment tuhosti v kroucení

- výstup programu pro přesnost mřížky 3*3 dílky: $I_K = 0{,}027850\ \mathrm{m}^4$
- výstup programu pro přesnost mřížky 10*10 dílků: $I_K = 0{,}028655\ \mathrm{m}^4$
- výstup programu pro přesnost mřížky 25*25 dílků: $I_K = 0{,}028600\ \mathrm{m}^4$

Ověření výsledku

Program:

$I_{K3} = 0{,}027850\ \mathrm{m}^4$

$I_{K10} = 0{,}028655\ \mathrm{m}^4$

$I_{K25} = 0{,}028600\ \mathrm{m}^4$

- vzorec pro masivní průřez obecného tvaru:

$$I_K = \frac{A^4}{40 I_P} = \frac{(bh)^4}{40\,\frac{bh^3 + b^3 h}{12}} = \frac{(0{,}5)^4}{40\,\frac{0{,}5^3 + 0{,}5}{12}} = 0{,}03\ \mathrm{m}^4$$

- přibližný vzorec pro obdélník (h<<b):

$$I_K = \frac{bh^3}{3}\left(1 - 0{,}63\,\frac{h}{b}\right) = \frac{0{,}5^3}{3}\,(1 - 0{,}63 \cdot 0{,}5) = 0{,}028542\ \mathrm{m}^4$$

- přesný vzorec pro obdélník (propočteno na prvních 11 členů):

$$I_K = \frac{bh^3}{3}\left(1 - \frac{192}{\pi^5} \cdot \frac{h}{b} \sum_{n=1;3;5\ldots}^{\infty} \left(\frac{1}{n^5} \tanh \frac{n\pi b}{8h}\right)\right) = 0{,}0330366\ \mathrm{m}^4$$