TOMÁŠ KALVODA, KATEDRA APLIKOVANÉ MATEMATIKY, FIT ČVUT, 2019

XXX. CVIČENÍ BI-LIN
PRO PARALELKY 14 A 15

DOMÁCÍ ÚKOL[1]

Úvod

Tento domácí úkol je možné vypracovat do konce výukové části semestru (**19. května 2019**). Za správné řešení všech úloh lze získat až dva body. Částečné nebo „částečně správné“ řešení bude hodnoceno odpovídajícím počtem bodů.

Příklady jsou zvoleny tak, aby ukazovaly různorodá využití lineární algebry a zároveň bylo možné je vyřešit „ručním počítáním“ (ano, i příklad na lineární kódování lze s trochou systematičnosti takto zvládnout). Zadání příkladů je dále občas doplněno komentáři k demonstrované problematice. Zvolený styl s částečným výkladem není náhodný, takto může třeba v budoucnu vypadat popis nějaké metody, kterou budete chtít použít.

Úlohy

1.1 Lineární regrese

Tento příklad demonstruje využití lineární algebry při konstrukci tzv. regrese (lidově „prokládání dat křivkou“; speciálně přímkou, viz např. wiki). V následujícím odstavci metodu popíšeme a níže si ji vyzkoušíme na jednoduchém příkladě.

Mějme sadu bodů[2] $\{(x_i, y_i)\}_{i=1}^n \subset \mathbb{R}^2$. *Tyto body představují „přibližnou“ závislost jisté veličiny y na veličině x, tj. $y = y(x)$. Naším cílem je co nejlépe tuto závislost vystihnout v následujícím tvaru*

$$y(x) = F_c(x) := \sum_{j=1}^{k} c_j f_j(x), \tag{1}$$

kde $f_1, \ldots, f_k$ jsou námi zvolené[3] *funkce*[4] *a $c_1, \ldots, c_k$ neznámé konstanty. Která z funkcí F_c tvaru (1) nejlépe vystihuje naše data? Nejjednodušší a asi nejznámější možností jak tuto funkci*

[1] Dokument byl vytvořen 25. února 2021, zdrojové kódy a Jupyter notebooky naleznete na Gitlabu.

[2] Zobecnění do vyšších dimenzí je samozřejmě možné.

[3] Typicky polynomy, ale není to nutné.

[4] Typicky samozřejmě máme $n \gg k$, tedy podstatně více dat (n) než funkcí (k), chceme „jednoduchou“ závislost vystihující například nějaký trend v datech.

určit je tzv. ***Metoda nejmenších čtverců****: najdeme konstanty $c_1, \dots, c_k$ takové, že funkce*

$$(c_1, \dots, c_k) \mapsto \sum_{i=1}^{n} \big(y_i - F_c(x_i)\big)^2$$

nabývá minimální hodnoty[5]*. Detailní řešení tohoto optimalizačního problému je mimo rámec BI-LIN (viz např. BI-VMM nebo MI-MPI). Zde se spokojíme s prozrazením výsledku: optimální $c = (c_1, \dots, c_k)$ jsou* ***řešením soustavy lineárních rovnic***

$$\mathbb{X}^T \mathbb{X} c^T = \mathbb{X}^T \mathbb{Y}, \tag{2}$$

kde $\mathbb{X} \in \mathbb{R}^{n,k}$ a $\mathbb{Y} \in \mathbb{R}^n$ jsou zadány předpisy

$$\mathbb{X}_{ij} = f_j(x_i), \quad \mathbb{Y}_i = y_i, \quad i \in \hat{n},\ j \in \hat{k}.$$

Aplikujte výše popsanou metodu na velmi, *velmi*, jednoduchém příkladě. Mějme tři body

$$(x_1, y_1) = (1,0), \quad (x_2, y_2) = (2,2), \quad (x_3, y_3) = (3,3),$$

a pokusme se je proložit přímkou (lineární regrese). Volíme tedy $f_1(x) = 1$ a $f_2(x) = x$.

1. V popisu metody výše jsme se nezabývali podmínkami, za kterých ji lze aplikovat. Jednou z těchto podmínek je lineární nezávislost souboru funkcí $(f_1, \dots, f_k)$. **Ukažte, že námi zvolený soubor** $(1, x)$ **je lineárně nezávislý v prostoru všech polynomů.**

2. **Nalezněte konstanty** $c = (c_1, c_2)$ **vyřešením soustavy** (2)**.**

3. **Nakreslete graf v kterém znázorníte tři výše zadané body a získanou přímku** $y = c_1 + c_2 x$**.**

1.2 Kubická interpolace ?

Věděli jste, že každým použitím příkazu `Plot` *v Mathematica nutíte Váš počítat řešit poměrně rozsáhlou soustavu lineárních rovnic?*

Mathematica totiž vypočte funkční hodnoty vykreslované funkce pouze v několika relativně málo bodech a poté je „hezkým způsobem spojí“. Ve výchozím nastavení k tomu použije kubické polynomy (kubický polynom je první polynom (co se týče stupně), který může mít inflexní bod).

Máme-li body $\{(x_i, y_i)\}_{i=1}^n \subset \mathbb{R}^2$ pak hledáme polynomy

$$f_j(x) = a_j + b_j x + c_j x^2 + d_j x^3, \quad j = 1, \dots, n-1,$$

[5] Jinak řečeno, pro kterou je součet **kvadrátů** odchylek funkčních hodnot $F_c(x_i)$ od y_i **nejmenší**.

tedy $4 \times (n-1)$ *neznámých konstant, splňující podmínky*

$$f_j(x_j) = y_j, \quad f_j(x_{j+1}) = y_{j+1}, \quad j = 1, \ldots, n-1, \tag{3}$$
$$f_j'(x_{j+1}) = f_{j+1}'(x_{j+1}), \quad j = 1, \ldots, n-2, \tag{4}$$
$$f_j''(x_{j+1}) = f_{j+1}''(x_{j+1}), \quad j = 1, \ldots, n-2, \tag{5}$$
$$f_1'(x_1) = 0, \quad f_{n-1}'(x_n) = 0. \tag{6}$$

Podmínky výše (je jich $4 \times (n-1)$*) popořadě představují následující omezení:*

- (3)*: chceme, aby graf polynomu* f_j *spojoval body* (x_j, y_j) *a* (x_{j+1}, y_{j+1}), $j = 1, 2, \ldots, n-1$,
- (4)*: chceme, aby „švy" byly hezké a polynomy na sebe navazovaly se stejnou derivací,*
- (5) *chceme, aby „švy" byly ještě hezčí a polynomy na sebe navazovaly se stejnou druhou derivací,*
- (6)*: doposud zmíněné požadavky by nestačily na jednoznačné určení neznámých konstant (více neznámých než rovnic), v této jednoduché ukázce proto volíme požadavek na nulovost derivací na krajích interpolovaného intervalu (kdybychom vykreslovali nějakou uživatelem zadanou funkci, pak bychom tyto derivace mohli položit rovné derivacím této funkce na krajích).*

Mějme opět tři body

$$(x_1, y_1) = (0, 4), \quad (x_2, y_2) = (1, 2), \quad (x_3, y_3) = (2, 1),$$

Aplikujte na ně postup popsaný výše a nalezněte jejich kubickou interpolaci (tj. hledáme dvě funkce $f_1(x) = a_1 + b_1 x + c_1 x^2 + d_1 x^3$ a $f_2(x) = a_2 + b_2 x + c_2 x^2 + d_2 x^3$ splňující rovnice (3) až (6), $n = 2$). Svůj výsledek nakreslete pomocí svého oblíbeného počítačového systému a do řešení tento obrázek přibližně překreslete.

1.3 Lineární kód ?

Lineární kódy (resp. samoopravovací kódy obecně) využíváme dnes a denně při přenosu nebo čtení dat (ani jedno nejsou bezchybné aktivity, jak se může na první pohled zdát).

V této úloze jsem studentům zakódoval a odeslal (bohužel, s chybami) jednoduchou textovou zprávu obsahující moudrý a velmi pravdivý citát jednoho z největších matematiků. K tomu jsem použil ASCII kódování (viz např. web). Na decimální hodnotu každého ze znaků jsem použil osm bitů. Například tedy

$$\text{'A'} \mapsto 65 \mapsto 01000001.$$

Zpráva tak představuje prosté osmice nul a jedniček jdoucí za sebou. Tuto posloupnost dále přečteme po čtveřicích a každou čtveřici zakódujeme pomocí následujícího kódování.

Použijeme lineární $(7,4)$-kód K s generující maticí

$$G_K = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix} \in \mathbb{Z}_2^{4,7}.$$

To znamená, že příklad písmene A uvedený výše je kódován do dvou kódových slov

$$01000001 \mapsto 01001010001111$$

Následně jsou při kopírování do tohoto dokumentu kódová slova drobně poškozena. Dekódujte následující zprávu:

1100101001111101101100010000011101111011100111011000110101011000010010011101
0111110011001001011100000110000010001111010000001001001100010110101100001111 0
0111101011010110101101000000001001111111100010011010001100101000010001110011
1101000001001101011111111100000010111100111010011001001100010100110010010100 1
0111011000101101101000100100111001100011000110001001100001111010111001001011 1
0010000001110001110000110001000001101100100001100011011110110010010011100011 1
1010110010000001101110110110011001001000100111011001101101100010111111011001 0
1101101010001101010111111100001100000001101000000111110001010000101100000100 1
0010011000101001100100110010011010001010000111011110111001101100000010010001 1
0100001111001111010100110100111001101101110001010011101111001000111011110111 1
1111100000001001110111101010011101101001001010110001000001111010111010001101 1
1001101011010001001110110001000001110101100000010001001100011010111011000011 0
1111100001101101100010111111011000110001100111111001000100101100011010100100 1 1
0100010011101111001110111111010010001010110000001001110111111000011000111101 1 1
0110110000000100100111100101011000101000100111110000100101101111000100111110 0
0010111011011101110100010010000100001110000110101011101111011110000011000001 0 0
0110001000011000100111101010001111001000000111000010111001111010011000011101 1
0111010011001011100110110100010101101100011111011000101001101110001010000000 1 0
0111011001111101110111111000100100110000101001010010000000010011100100001000 1 1
1100000000011000001000010011000101101101100010000011100000110000010011100111 0
0111000110010101111010101011001001100011101111100010011101110111011000111011
0111111111001111100100011110000010001111100110100011001010000010100011010101 1
0111110001001001100101101010001011100000011110011011000000100011111110101000 1 1
0000110001001101111110110001010010001111011000100001001000000010110010001101 1
0110010011111101011000111010001001111101100110001000001010000110001000011111 0 0
0000001011000111101110110100001101101111010011011011101110001001110011101011 0
0110001100011001111101101100001111000000101110011101001001100101100000011011 1
0111111001110001110000000110000000011110011101110000100110110000000101110000 1 0 0
0111000001110001100011100011001001100011011111100010110100111100100000011011 1
0000111011111000001100110111000101000100111001111101011001000000110111000110 0

01110001100101001011100010000101100010110001100011101111011111011010100010011
11001110111011010011100111100000010011001000100100110001011101001000110011101
01101111011111011001011011011110011010001000101000010000011110101000101111100
00111010110010010001000100110010011010110001010110110010110111000110001011000
00101000001000001001100010111011011101100000111011010011100111100010000011110 1
01001000110001100101101100100101011011100000100100111011000101010101100010000
01011000110100011000101010110111110001111001100100001010001111000000010110111
10100001110011011111100101110010000001001101110000111011011101001100101111000
01100100101110111110010101100010011000111001101110100110001011100100001110011
10011100110001111100001101111100101010001110011010110111101100111100110001110
01101000100100111001110011100111001100101111011011001011000000011100000100100001 10
01000000111110010011101110111100011010001101011110011100010011100101000001000
00100111001101011110101000010010100000001001000110001010011111001000111110011
01011100111110011011000101000010000001110001110000110111111110110111110000101 0
01110111100100011001001001110011111000000100111011001011101100100101000011011 0
01000001111100000100101000111011111101011001100000

1.4 Vlastní vektory a vlastní čísla ?

Ukázky použití vlastních vektorů a vlastních čísel by vydaly na větší příklady (viz studijní text). Zde se proto soustředíme na jednoduché geometrické transformace v rovině (tj. operátory na $\mathbb{R}^2$) a prozkoumáme jejich vlastní čísla a vlastní vektory.

Nalezněte vlastní čísla a vlastní vektory následujících lineárních operátorů na $\mathbb{R}^2$:

a) **Projektor A na podprostor $Q = \langle(1,2)\rangle$**: buď $\mathscr{X} = (x_1, x_2) = \big((1,2),(-2,1)\big)$ báze $\mathbb{R}^2$, je-li $x_{\mathscr{X}} = (\alpha_1, \alpha_2)$ pak $Ax := \alpha_1 x_1$.

b) **Rotace okolo počátku proti směru hodinových ručiček o úhel $\frac{\pi}{4}$**: tj. operátor B, který na vektoru $x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \in \mathbb{R}^2$ působí předpisem

$$Bx := \begin{pmatrix} \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \end{pmatrix} x.$$

c) **Zrcadlení C vůči přímce $P = \langle(1,-1)\rangle$**: buď $\mathscr{Y} = \big(y_1, y_2\big) = \big((1,-1),(1,1)\big)$ báze $\mathbb{R}^2$, je-li $x_{\mathscr{Y}} = (\alpha_1, \alpha_2)$ pak $(Cx)_{\mathscr{Y}} := (\alpha_1, -\alpha_2)$.

d) **Škálování D_γ parametrem γ ve směru vektoru** $(2,1)$: buď $\mathscr{Z} = \big(z_1, z_2\big) = \big((2,1),(1,-2)\big)$ báze $\mathbb{R}^2$, je-li $x_{\mathscr{Z}} = (\alpha_1, \alpha_2)$ pak $(D_\gamma x)_{\mathscr{Y}} := (\gamma\alpha_1, \alpha_2)$.

Pokuste se výsledky (vlastní čísla a vlastní vektory) interpretovat ve vztahu k danému operátoru. Co nám vlastní čísla a vlastní vektory v těchto konkrétních příkladech o původních operátorech říkají?