Aplikační seminář lineární algebry

František Batysta

19. května 2019

Abstrakt

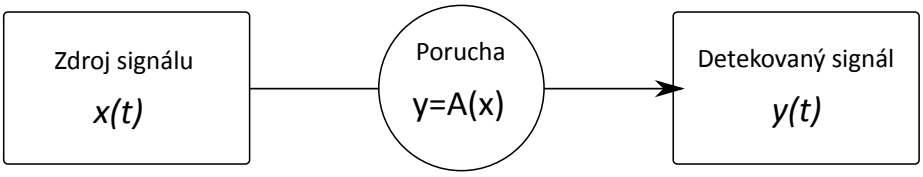
V tomto textu naleznete shrnutí aplikačního semináře k lineární algebře, který se konal v závěru kurzu lineární algebry na FJFI ČVUT. Cílem semináře je ukázat rozličné aplikace lineární algebry a motivovat tím studenty, aby se lineární algebru před zkouškou pořádně a s chutí naučili. Bude se jim to totiž mockrát hodit :).

1 Teoretický úvod

1.1 Motivace: zkreslení přenosu signálu

Uvažujme systém podle schématu 1. Při něm dochází k přenosu signálu ze zdroje k přijímači, přičemž při přenosu dojde k „poruše“ signálu. Předpokládáme, že porucha je

- Lineární.
- Respektuje kauzalitu.



Obrázek 1: Obecné schéma systémů s lineární odezvou.

Signál numericky reprezentujeme tak, že ho „navzorkujeme“ posloupností komplexních čísel $\vec{\mathbf{x}} \in (x_i)_{-\infty}^{+\infty} : \mathbb{Z} \to \mathbb{C}$. Předpokládáme, že zkreslený signál dostaneme v každém okamžiku jako libovolnou (avšak stále stejnou) lineární kombinaci l bodů původního signálu. Abychom splnili podmínku kauzality, předpokládáme pouze „jednostrannou“ lineární kombinaci - tj. kombinujeme pouze z těch vzorků, které již nastaly (nikoliv z těch budoucích)

$$(\exists l \in \mathbb{N})(\forall j \in \mathbb{N}) \left(y_j = \sum_{k=0}^{l} \alpha_k x_{j+k} \right), \tag{1}$$

kde $\alpha_1, \ldots, \alpha_l \in \mathbb{C}$ jsou pevně dané koeficienty lineární kombinace, které charakterizují poruchu přenášeného signálu. Je zřejmé, že takto definovaná odezva systému je lineární a lze si ji představit jako působení lineárního operátoru

$$\vec{\mathbf{y}} = A\vec{\mathbf{x}}, \quad A \in \mathcal{L}((x_i)_{-\infty}^{+\infty}) \tag{2}$$

Příklad 1. *Jednoduchým příkladem odezvy systému je průměrování. Například pro $l = 3$ vypadá odezva následujícím způsobem.*

$$(\forall j \in \mathbb{N}) \left(y_j = \frac{1}{3}(x_j + x_{j+1} + x_{j+2}) \right)$$

1.2 Harmonické posloupnosti jako vlastní vektory odezvy

Co můžeme obecně říci o operátoru A, který je definován vztahem (1)? Můžeme najít jeho vlastní vektory? Ano. Například lze snadno ukázat, že každá harmonická posloupnost $\vec{\mathbf{x}}$, jejíž složky lze psát

$$x_j = \exp(i\omega j), \quad \omega \in \mathbb{R} \tag{3}$$

je jejím vlastním vektorem, neboť

$$y_j = (A\vec{\mathbf{x}})_j = \sum_{k=0}^{l} \alpha_k \exp(i\omega[k+j]) = \underbrace{\exp(i\omega j)}_{x_j} \underbrace{\sum_{k=0}^{l} \alpha_k \exp(i\omega k)}_{=:\lambda \quad \text{(nezávisí na } j\text{)}} \tag{4}$$

Tedy pro každé reálné číslo ω je komplexní harmonická posloupnost (3) vlastním vektorem operátoru A, neboť

$$(\forall j \in \mathbb{N})(y_j = \lambda x_j) \Rightarrow \vec{\mathbf{y}} = A\vec{\mathbf{x}} = \lambda\vec{\mathbf{x}}. \tag{5}$$

Zároveň jsme pro každé $\omega \in \mathbb{R}$ dostali jemu příslušící vlastní číslo

$$\lambda_\omega = \sum_{k=0}^{l} \alpha_k \exp(i\omega k) \tag{6}$$

1.3 Přechod k prostoru konečné dimenze

Praktické počítání (např. na počítači) s nekonečnými posloupnostmi je poněkud obtížné. Omezíme se tedy pouze na podprostor konečné dimenze: budeme uvažovat pouze periodické komplexní posloupnosti s periodou $n \in \mathbb{N}$, tj. budeme pracovat s podprostorem komplexních posloupností

$$P_n = \left\{ (\dots, x_0, x_1, x_2, \dots) \in (x_i)_{-\infty}^{+\infty} | (\forall j \in \mathbb{N})(x_{n+j} = x_n) \right\} \tag{7}$$

Poznámka 1. *Tím, že jsme se omezili na podprostor konečné dimenze, stačí nyní, když budeme reprezentovat periodické posloupnosti vektory z prostoru $\mathbb{C}^n$.*

Poznámka 2. *Oproti zvyklostem z přednášky se nám bude hodit, pokud budeme v následujícím textu číslovat složky všech vektorů z $\mathbb{C}^n$ od nuly. Tím totiž nastavíme fázi všech harmonických vektorů tak, že jejich první složka je rovná jedné, což je v souladu s historickými konvencemi. Budiž tedy například*

$$\vec{\mathbf{x}} = \begin{pmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_{n-1} \end{pmatrix}. \tag{8}$$

Tímto omezením se nám počítání velmi zjednoduší (budeme moci např. najít šikovnou bázi prostoru) a přitom z hlediska aplikací se o žádné podstatné omezení nejedná. Periodu n si můžeme volit libovolně dlouhou. Pokud nás zajímá odezva systému na jakýkoliv časově omezený děj, stačí jej reprezentovat takovou periodickou posloupností, jejíž perioda je delší než doba trvání tohoto děje.

Hledejme nyní šikovnou bázi podprostoru periodických posloupností. Můžeme tuto bázi poskládat z harmonických posloupností? Harmonické posloupnosti v podprostoru P musí splňovat podmínky

$$(\forall j \in \mathbb{N})(\underbrace{x_j = \exp(i\omega j)}_{\text{posl. je harmonická}} \wedge \underbrace{x_{j+n} = x_j}_{\text{má periodu } n}) \tag{9}$$

Z toho nám vyplyne podmínka pro ω. Dosazením první podmínky do druhé dostaneme podmínku

$$\exp(i\omega[j+n]) = \exp(i\omega j) \tag{10}$$

$$\exp(i\omega n) = 1 \tag{11}$$

Rovnici (11) lze splnit, pokud

$$\omega \in \left\{ \frac{2\pi k}{n} \middle| k \in \mathbb{Z} \right\}. \tag{12}$$

Není těžké si rozmyslet, že se zvětšujícím se k dostaneme pouze n různých harmonických posloupností; pak se posloupnosti začnou opakovat. Omezíme-li tedy čísla k na množinu $k \in \{0, 1, \ldots, n-1\}$ a označíme-li si

$$\varphi := \exp\left(i\frac{2\pi}{n}\right), \tag{13}$$

dostaneme následující soubor harmonických vektorů[1], které navíc volíme normované k jedné.

$$\mathcal{X} = (\vec{\mathbf{x}}_0, \vec{\mathbf{x}}_1, \ldots, \vec{\mathbf{x}}_{n-1}) \tag{14}$$

$$= \left(\frac{1}{\sqrt{n}} \begin{pmatrix} 1 \\ 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix}, \frac{1}{\sqrt{n}} \begin{pmatrix} 1 \\ \varphi^1 \\ \varphi^2 \\ \vdots \\ \varphi^{n-1} \end{pmatrix}, \frac{1}{\sqrt{n}} \begin{pmatrix} 1 \\ \varphi^2 \\ \varphi^4 \\ \vdots \\ \varphi^{(n-1)2} \end{pmatrix}, \ldots, \frac{1}{\sqrt{n}} \begin{pmatrix} 1 \\ \varphi^{n-1} \\ \varphi^{2(n-1)} \\ \cdots \\ \varphi^{(n-1)(n-1)} \end{pmatrix} \right), \tag{15}$$

Přičemž j-tá složka k-tého vektoru lze vyčíslit jako

$$(\vec{\mathbf{x}}_k)_j = \frac{1}{\sqrt{n}}\varphi^{jk}, \qquad k, j \in \{0, 1, \ldots, n-1\} \tag{16}$$

Poznámka 3. *Také vektory v souborech číslujeme z historických důvodů od nuly. První (zde nultý) vektor souboru $\mathcal{X}$ totiž odpovídá „nulové“ frekvenci.*

[1] Protože posloupnosti jsou periodické, stačí pracovat pouze s prvními $n-$složkami.

1.4 Je soubor X LN? Vandermondův determinant

Je soubor $\mathcal{X}$ bází podprostoru P? Musíme ověřit, zda je $\mathcal{X}$ LN. Můžeme například zkoumat determinant matice přechodu

$$ {}^{\mathcal{X}}I^{\mathcal{E}} = (\vec{\mathbf{x}}_0, \vec{\mathbf{x}}_1, \ldots, \vec{\mathbf{x}}_{n-1}) = \frac{1}{\sqrt{n}} \begin{pmatrix} 1 & 1 & 1 & \ldots & 1 \\ 1 & \varphi^1 & \varphi^2 & \ldots & \varphi^{1(n-1)} \\ 1 & \varphi^2 & \varphi^4 & \ldots & \varphi^{2(n-1)} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & \varphi^{n-1} & \varphi^{(n-1)n} & \ldots & \varphi^{(n-1)(n-1)} \end{pmatrix}. \tag{17} $$

To je ale speciální případ Vandermondova determinantu, jehož hodnota je se dá vyčíst ve sbírce příkladů vzadu

$$ \det {}^{\mathcal{X}}I^{\mathcal{E}} = \det V = \prod_{0 \le i < j \le n-1} (\varphi^i - \varphi^j) \neq 0. \tag{18} $$

Determinant je nenulový, neboť všechny členy produktu jsou zřejmě nenulové.

1.5 Diskrétní Fourierova transformace

Navíc si můžeme všimnout, že vektory $\vec{\mathbf{x}}_i$ jsou dokonce ON!

$$ \langle \vec{\mathbf{x}}_k | \vec{\mathbf{x}}_j \rangle = \frac{1}{n} \sum_{l=0}^{n-1} \exp\left(i \left[\frac{2\pi kl}{n} - \frac{2\pi jl}{n} \right] \right) = \frac{1}{n} \sum_{l=0}^{n-1} \left(\varphi^{[k-j]} \right)^l = \delta_{k,j}. \tag{19} $$

V posledním kroku jsme sečetli geometrickou řadu. K výpočtu souřadnic vektorů v nové bázi bychom potřebovali znát matici ${}^{\mathcal{E}}I^{\mathcal{X}} = ({}^{\mathcal{X}}I^{\mathcal{E}})^{-1}$. Protože je matice ${}^{\mathcal{X}}I^{\mathcal{E}}$ unitární, matici k ní inverzní dostaneme hermitovským sdružením.

$$ {}^{\mathcal{E}}I^{\mathcal{X}} = \left({}^{\mathcal{X}}I^{\mathcal{E}} \right)^{-1} = \left({}^{\mathcal{X}}I^{\mathcal{E}} \right)^{H} = \frac{1}{\sqrt{n}} \begin{pmatrix} 1 & 1 & 1 & \ldots & 1 \\ 1 & \overline{\varphi^1} & \overline{\varphi^2} & \ldots & \overline{\varphi^{1(n-1)}} \\ 1 & \overline{\varphi^2} & \overline{\varphi^4} & \ldots & \overline{\varphi^{2(n-1)}} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & \overline{\varphi^{(n-1)1}} & \overline{\varphi^{(n-1)2}} & \ldots & \overline{\varphi^{(n-1)(n-1)}} \end{pmatrix} \tag{20} $$

Souřadnice vektorů v nové bázi pak můžeme psát po složkách jako Fourierovy koeficienty

$$ {}^{\mathcal{E}}I^{\mathcal{X}} (\vec{\mathbf{x}})_{\mathcal{E}} = (\vec{\mathbf{x}})_{\mathcal{X}} = \begin{pmatrix} \langle \vec{\mathbf{x}} | \vec{\mathbf{x}}_1 \rangle \\ \langle \vec{\mathbf{x}} | \vec{\mathbf{x}}_2 \rangle \\ \vdots \\ \langle \vec{\mathbf{x}} | \vec{\mathbf{x}}_n \rangle \end{pmatrix} \tag{21} $$

Tento proces, tj. převod vektoru do nové fourierovské báze, není ničím jiným než známou **Fourierovou transformací** na prostoru konečné dimenze a obvykle ji zapisujeme jako

$$ \mathcal{FT}((\vec{\mathbf{x}})_{\mathcal{E}}) := (\vec{\mathbf{x}})_{\mathcal{X}} = {}^{\mathcal{E}}I^{\mathcal{X}} (\vec{\mathbf{x}})_{\mathcal{E}} \tag{22} $$

Zpětný převod z fourierovské báze do standardní pak nazýváme **inverzní Fourierovou transformací.**

$$\mathcal{IFT}((\vec{\mathbf{x}})_{\mathcal{X}}) := (\vec{\mathbf{x}})_{\mathcal{E}} = {}^{\mathcal{X}}I^{\mathcal{E}}(\vec{\mathbf{x}})_{\mathcal{X}} \tag{23}$$

V řadě aplikací se vyplatí pracovat spíše s Fourierovou transformací vektoru než s původním vektorem.[2] Například v úvodu zmíněný operátor A má v bázi $\mathcal{X}$ diagonální tvar

$${}^{\mathcal{X}}A = \begin{pmatrix} \lambda_0 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \lambda_{n-1} \end{pmatrix}, \tag{24}$$

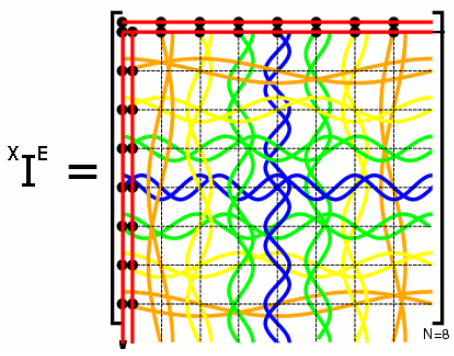
kde vlastní čísla na diagonále lze vyjádřit pomocí inverzní Fourierovy transformace.

$$\lambda_k = \sum_{j=0}^{n-1} \alpha_j \exp\left(\frac{2\pi kj}{n}\right) \tag{25}$$

což lze zapsat vektorově, doplníme-li případné chybějící koeficienty α_j tak, abychom mohli pracovat s vektorem $\vec{\alpha}$

$$\vec{\lambda} = \begin{pmatrix} \lambda_0 \\ \lambda_1 \\ \vdots \\ \lambda_{n-1} \end{pmatrix} = \sqrt{n} \cdot {}^{\mathcal{X}}I^{\mathcal{E}}\vec{\alpha} = \sqrt{n} \cdot \mathcal{IFT}(\vec{\alpha}) \tag{26}$$

Poznámka 4. *Prvky matice přechodu* ${}^{\mathcal{X}}I^{\mathcal{E}}$ *jsou graficky znázorněny na obr. 2. Všimněte si, že harmonické vektory s nejvyšší frekvencí se nacházejí poblíž prostředku matice, kdežto vektory s nízkými frekvencemi jsou na krajích matice. Vektory v posledním sloupečku sice mají numericky nejvyšší frekvenci* $\omega = \exp\left(i\frac{2\pi(n-1)}{n}\right)$*, jenže to se kvůli vzorkování jeví jako mnohem pomalejší záporná frekvence* $\omega = \exp\left(i\frac{2\pi(-1)}{n}\right)$.



Obrázek 2: Vizualizace prvků matice Fourierovy transformace.

[2] Procesu, kdy se přejde od standardní báze k Fourierově bázi, se také někdy říká „přechod do frekvenční domény“

2 Aplikace Fourierovy transformace

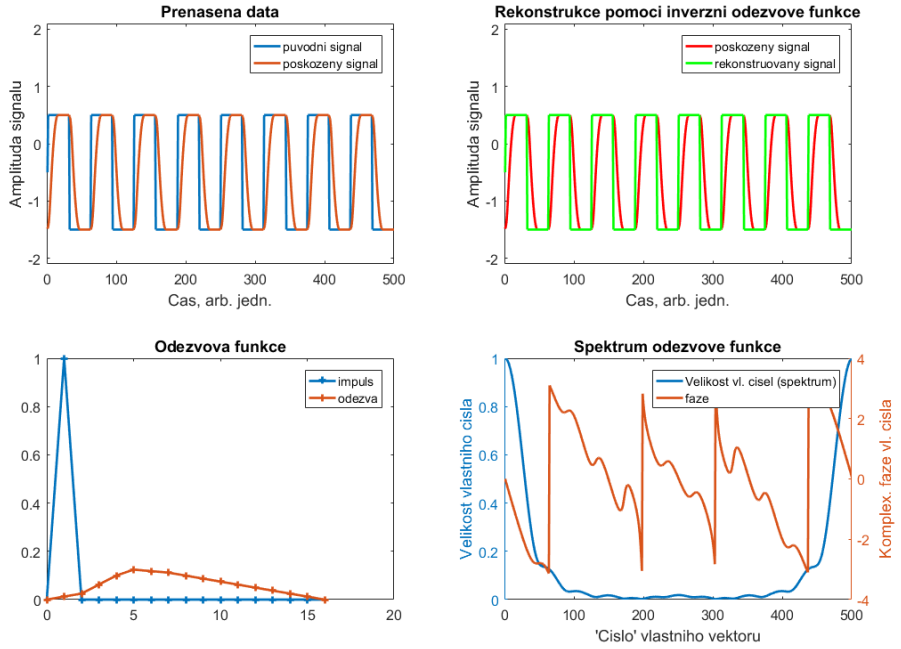
V této sekci jsou stručně popsány vybrané aplikace diskrétní Fourierovy transformace. Ty jsou předvedeny pomocí jednoduchých prográmků v Matlabu. Studenti si mohou, v případě zájmu, prográmky sami vyzkoušet. Všechny prográmky by měly jít spustit tak, jak jsou; některé prográmky mají nepovinné vstupní parametry.

2.1 Systémy s lineární odezvou

Prográmek 1. *odezva.m*

Tento prográmek navazuje přímo na předchozí teoretickou sekci. Typická odezva reálných systémů (např. osciloskop, mikrofon, fotodioda, atd.) je podobná jako na obrázku 3 vlevo dole. Na krátký vstupní impuls systém odpoví např. mnohem delším, jakoby rozmazaným, impulsem.

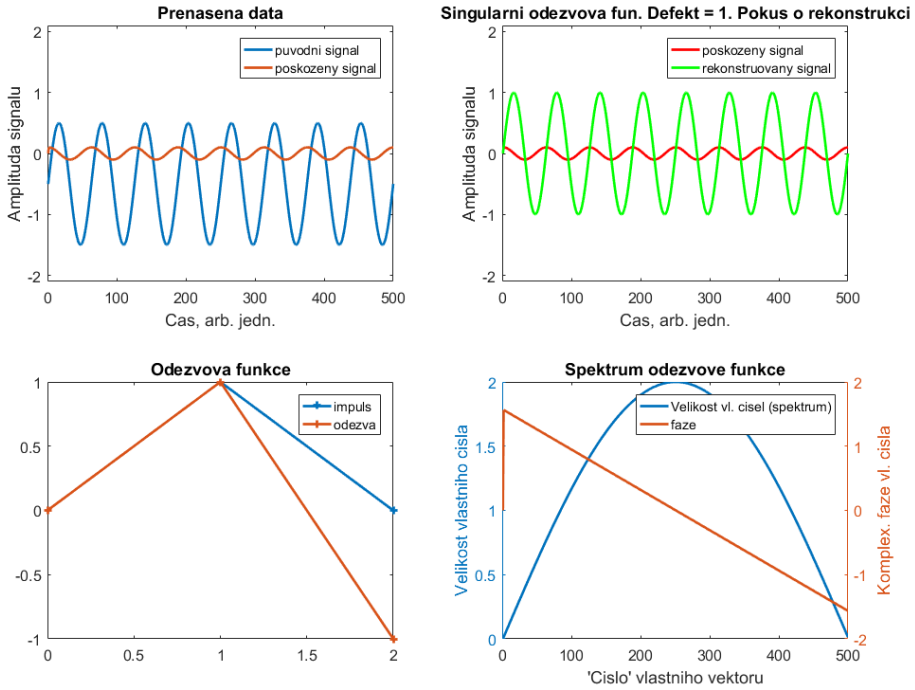
Jednotlivé body tzv. impulzní odezvy odpovídají koeficientům α_l z rovnice (1). Operátor odezvy A pak můžeme snadno reprezentovat pomocí jeho vlastních čísel, viz obr. 3 vpravo dole. Vidíme, že se operátor chová jako frekvenční filtr. Nízké frekvence propustí, zatímco vyšší frekvence utlumí. Pro rekonstrukci původního signálu stačí poškozený vektor ve frekvenční doméně vynásobit diagonální maticí A^{-1} (tím vlastně opravíme chybějící vyšší frekvence) a pak převést zpět do časové domény.



Obrázek 3: Rekonstrukce původního signálu s pomocí spektra odezvové funkce.

Operátory lineární odezvy však mohou nabývat velmi různých podob. Obrázek 4 ukazuje diferenční operátor – příbuzný operátoru derivování. Jeho koeficienty

odezvy $\vec{\alpha}^T = (1, -1, 0, 0, \dots)$. Tato odezva není prostá, jelikož všechny konstantní posloupnosti zobrazuje do nuly. Skutečně, vlastní číslo příslušící vektoru $\vec{\mathbf{x_0}}$ je 0 a inverzní operátor neexistuje. Nicméně, můžeme se pokusit částečně invertovat signál pomocí zbývajících harmonických vektorů, jež přísluší nenulovým vlastním číslům. Na obr. 4 vpravo nahoře je vidět, že se opravdu podařilo původní signál zrekonstruovat, ovšem až na konstantní posun, který přísluší vektoru s nulovou frekvencí $\vec{\mathbf{x_0}}$ a informaci o této složce jsme ztratili.



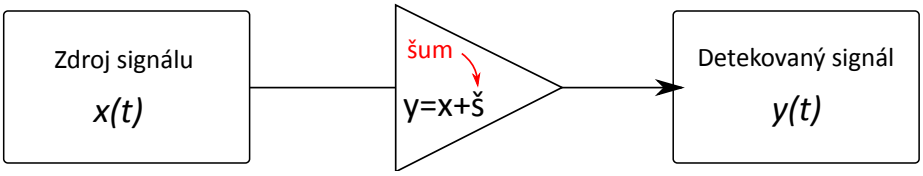
Obrázek 4: Částečná rekonstrukce signálu v případě odezvy s defektem 1.

2.2 Odstranění šumu

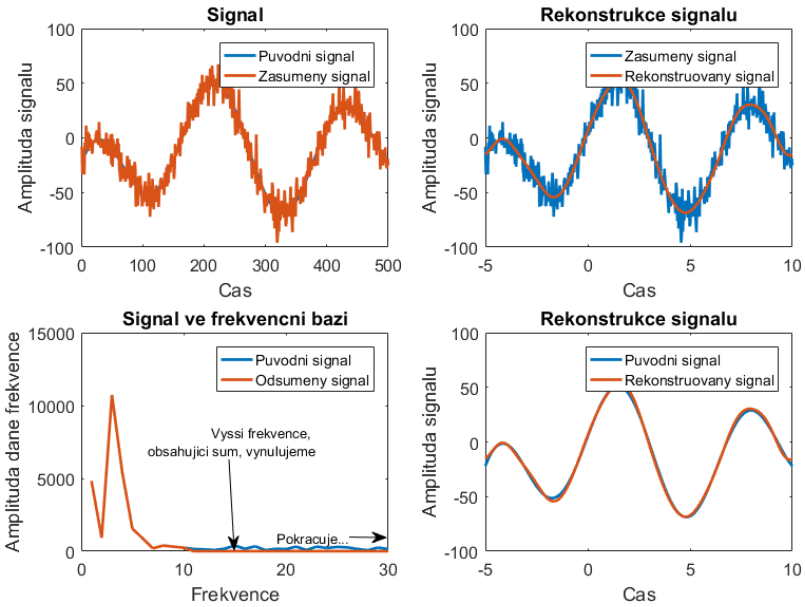
Prográmek 2. *odsumet.m; odsumet2.m*

Fourierova transformace se hodí i k analýze šumu. Uvažujme uspořádání podle obr. 5. K původnímu zdroji signálu přidáme náhodný šum. Tento proces není lineární, a tak už nemůžeme interpretovat fourierovskou bázi jakožto bázi vlastních vektorů. Přesto však může být reprezentace vektoru ve fourierovské doméně užitečná, pokud máme důvod se domnívat, že užitečný signál a náhodný šum je možné od sebe (alespoň částečně) frekvenčně oddělit.

Na obr. 6 je příklad původního hladkého a pomalu se měnícího signálu, který je zatížený náhodným (tzv. „bílým“) šumem. Ve frekvenční doméně vidíme náš signál jako výrazný pík na nízkých frekvencích, kdežto šum je rovnoměrně rozprostřen přes všechny frekvence. Po vynulování složek vektoru ve frekvenční doméně, které neobsahují užitečný signál, provedeme zpětnou Fourierovu transformaci. Výsledná rekonstrukce je sice také trochu ovlivněna šumem (šum je totiž mimo jiné i na těch frekvencích, které nesou náš signál), avšak velice se blíží původnímu signálu, což jsme chtěli.



Obrázek 5: Schéma přenosu signálu zatíženého šumem.



Obrázek 6: Ukázka odstranění šumu.

2.3 Zpracování obrazu

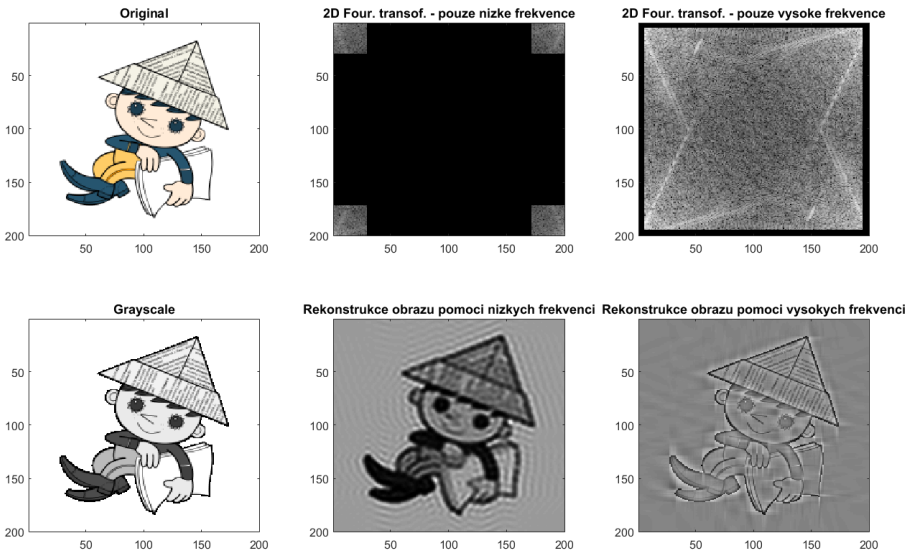
Prográmek 3. *obraz.m*

Fourierova transformace je užitečná také při zpracování obrazu, tj. na prostoru matic. Ve 2D lze Fourierovu transformaci provést tak, že převedeme do frekvenční domény nejprve všechny sloupce matice a takto získanou mezi-matici převedeme do frekvenční domény ještě jednou – tentokrát po řádcích.

Na obrázku 7 jsou znázorněny nejjednoduší aplikace: Komprese obrázku pomocí rozmazání a detekce hran. Pro účely analýzy použijeme obrázek Večerníčka, který pro jednoduchost převedeme do odstínů šedi. Tak vlastně pracujeme s maticemi z prostoru $\mathbb{C}^{m,n}$.

Jednoduchou ztrátovou kompresi obrázku můžeme provést tak, že převedeme obrázek do frekvenční domény, a zde ponecháme jenom ty pixely, které obsahují informace o nízkých prostorových frekvencích obrázku. Nízké prostorové frekvence nesou informace o celkovém vzezření obrázku, ale postrádají informace o detailech, například hranách. Zpětně zrekonstruovaný Večerníček (pouze s pomocí $< 10\%$ původní informace) má zachované všechny hlavní rysy, nicméně je rozmazaný.

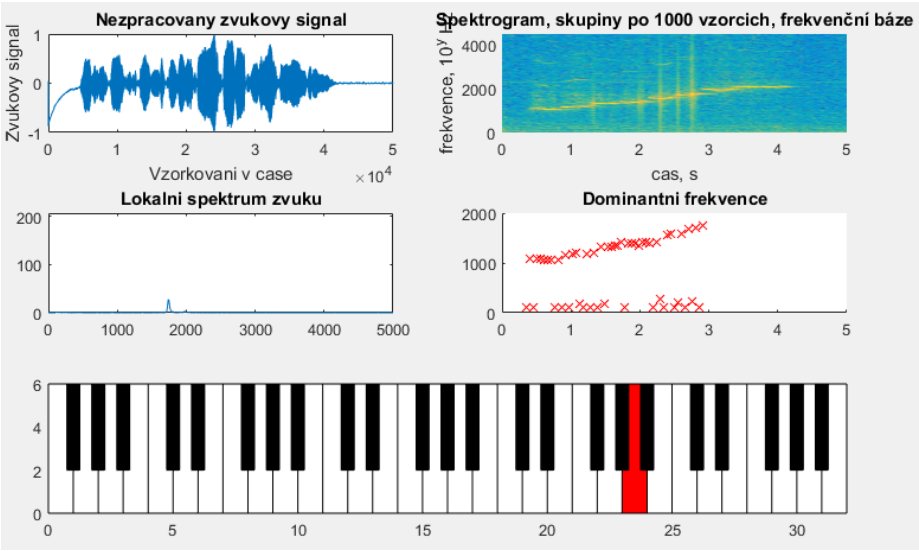
Naopak, potlačíme-li při rekonstrukci nízké frekvence, dostaneme jednoduchý algoritmus pro detekci hran obrázku. Rekonstrukci večerníčku z frekvenční domény, kde jsme ponechali pouze vyšší prostorové frekvence, lze vidět na obrázku 7 vpravo.



Obrázek 7: Zpracování obrazu pomocí 2D - Fourierovy transformace

2.4 Analýza zvuku

Prográmek 4. *nahraj.m; spektrogram.m; rozpoznanitonu.m*



Obrázek 8: Ukázka analýzy zvuku.

Fourierova transformace je také vhodným nástrojem pro analýzu zvuku. Zvukové signály jsou obvykle velmi dlouhé (viz obr. 8 vlevo nahoře), a tak je vhodné je rozdělit na kratší úseky o zvolené délce (např několik milisekund), které převedeme do frekvenční domény každý zvlášť. Tím dostaneme tzv. spektrogram – časový vývoj frekvencí, které jsou ve zvuku zastoupeny.

Prográmek `rozpoznanitonu.m` je jednoduchý nástroj pro analýzu tónů - fourierovsky analyzuje zvukový záznam. Navzorkovaný zvuk rozdělí do skupin po 1000 vzorcích a v každém segmentu určí dominantní frekvenci. Tu pak přiřadí nejbližšímu tónu, který zvýrazní na klávesnici.

Pro nahrání nového zvuku je možné použít funkci `nahraj.m`.