VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA STROJNÍHO INŽENÝRSTVÍ
ÚSTAV MATEMATIKY

FACULTY OF MECHANICAL ENGINEERING
INSTITUTE OF MATHEMATICS

APLIKACE LINEÁRNÍ ALGEBRY A OPTIMALIZACE VE ZPRACOVÁNÍ OBRAZŮ

APPLICATIONS OF LINEAR ALGEBRA AND OPTIMIZATION IN IMAGE PROCESSING

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

MARIE DAŇKOVÁ

VEDOUCÍ PRÁCE
SUPERVISOR

Mgr. PAVEL RAJMIC, Ph.D.

BRNO 2012

Vysoké učení technické v Brně, Fakulta strojního inženýrství

Ústav matematiky
Akademický rok: 2011/2012

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

student(ka): Marie Daňková

který/která studuje v **bakalářském studijním programu**

obor: **Matematické inženýrství (3901R021)**

Ředitel ústavu Vám v souladu se zákonem č.111/1998 o vysokých školách a se Studijním a zkušebním řádem VUT v Brně určuje následující téma bakalářské práce:

Aplikace lineární algebry a optimalizace ve zpracování obrazů

v anglickém jazyce:

Applications of linear algebra and optimization in image processing

Stručná charakteristika problematiky úkolu:

Moderním odvětvím zpracování signálu je využívání tzv. řídké reprezentace obrazů, tzn. zjištění, že obrazy z reálného světa se skládají z relativně malého počtu stavebních bloků vůči celkovému počtu pixelů. Metodami lineární algebry a optimalizace lze nejen toto vyjádření získat, ale také vyžít k praktickým úkolům jako jsou potlačování šumu, efektivní komprese, změna rozlišení apod.

Smyslem práce je seznámit se se současným stavem problematiky, nastudovat algoritmy pro hledání řídkých reprezentací a vše empiricky ověřit na simulovaných i reálných obrazových datech.

Cíle bakalářské práce:

Nastudovat základní principy a algoritmy hledání tzv. řídkých reprezentací.
Zaměřit se na typické situace v obrazových signálech.
V software MATLAB nebo MAPLE empiricky ověřit teorii na konkrétních datech.
Shrnout a interpretovat výsledky.

Seznam odborné literatury:

ELAD, M., Sparse and Redundant Representations: From Theory to Applications in Signal and Image Processing, Springer, 2010.

HRBÁČEK, R.; RAJMIC, P.; VESELÝ, V.; ŠPIŘÍK, J. Řídké reprezentace signálů: Úvod do problematiky. Elektrorevue - Internetový časopis, 2011, roč. 2011, č. 50, s. 1-10. ISSN: 1213-1539.

MAIRAL, J., ELAD, M., SAPIRO, G., Sparse learned representations for image restoration, IASC2008.

STARCK, J.L., MURTAGH, F., FADILI, J.M., Sparse Image and Signal Processing: Wavelets, Curvelets, Morphological Diversity, Cambridge University Press, 2010, 336 pages. ISBN 0521119138

Vedoucí bakalářské práce: Mgr. Pavel Rajmic, Ph.D.

Termín odevzdání bakalářské práce je stanoven časovým plánem akademického roku 2011/2012.

V Brně, dne 12.11.2011

L.S.

_____	_____
prof. RNDr. Josef Šlapal, CSc.	prof. RNDr. Miroslav Doupovec, CSc., dr. h. c.
Ředitel ústavu	Děkan fakulty

Abstrakt
Tato bakalářská práce se zabývá řídkou reprezentací obrazů, stručně uvádí tuto problematiku a popisuje základní algoritmy pro hledání řídkých reprezentací. Poté jsou tyto metody empiricky ověřeny na simulovaných i reálných datech pomocí softwaru Matlab.

Summary
This bachelor's thesis deals with sparse representation of images, briefly introduces this problems and describes basic algorithms for searching sparse representations. Then this methods are verified experimentally on simulated and real data by software Matlab.

Klíčová slova
řídkost, zpracování obrazů, řídká reprezentace obrazů

Keywords
sparsity, image processing, sparse representation of images

DAŇKOVÁ, M.*Aplikace lineární algebry a optimalizace ve zpracování obrazů.* Brno: Vysoké učení technické v Brně, Fakulta strojního inženýrství, 2012. 35 s. Vedoucí Mgr. Pavel Rajmic, Ph.D.

Prohlašuji, že jsem bakalářskou práci Aplikace lineární algebry a optimalizace ve zpracování obrazů vypracovala samostatně pod vedením Mgr. Pavla Rajmice, Ph.D. s použitím materiálů uvedených v seznamu literatury.

Marie Daňková

Děkuji vedoucímu bakalářské práce Mgr. Pavlu Rajmicovi, Ph.D. za odbornou pomoc a cenné rady při psaní práce.

Marie Daňková

Obsah

1. Úvod

V poslední době se začíná využívat pro zpracování obrazu jejich tzv. řídká reprezentace. Zjistilo se totiž, že obrazy z reálného světa se skládají z relativně malého počtu stavebních bloků vůči celkovému počtu pixelů, což lze chápat jako nedourčený systém lineárních rovnic, kde existuje řešení, které má velmi málo nenulových proměnných. Metodami lineární algebry a optimalizace lze nejen toto řešení získat, ale také využít k praktickým úkolům jako je potlačování šumu, odstraňování rozmazání, efektivní komprese, změna rozlišení a jiné.

Tato bakalářská práce pojednává o řídké reprezentaci obrazů a algoritmech pro jejich hledání. Důležitou součástí práce jsou programy, které empiricky ověřují tyto metody na simulovaných i reálných obrazových datech.

Kapitola 2. nastiňuje problém řídké reprezentace signálů – obrazů – pomocí nedourčeného systému lineárních rovnic, ve kterém se využívá tzv. slovník. Kapitola 2. udává, za jakých podmínek existuje řídké řešení, které je jednoznačné, a kapitola 3.2. se zamýšlí nad správnou volbou slovníku.

5. kapitola se věnuje popisům základních algoritmů pro hledání řídkého řešení soustavy lineárních rovnic, a to hladovým a relaxačním algoritmům. Podobně 3.2. kapitola uvádí dvě základní metody pro hledání optimálního slovníku (MOD a K-SVD). V poslední 7. kapitole jsou popsány programy, které demonstrují využití popsaných algoritmů. Rovněž jsou zde uvedeny výsledky z těchto programů pro různá vstupní data a tyto výsledky jsou navzájem porovnány.

2. Řídká reprezentace signálů – obrazů

Signály (zvuk, obraz, video) můžeme reprezentovat jako systém lineárních rovnic $\mathbf{Ax} = \mathbf{y}$, kde $\mathbf{A}$ je matice koeficientů, $\mathbf{x}$ vektor neznámých a $\mathbf{y}$ známý vektor (může to být pozorování, měření, signál). Tento systém je nedourčený, tj. máme více neznámých než kolik je rovnic. Předpokládáme, že matice $\mathbf{A}$ je plné hodnosti, a proto existuje nekonečně mnoho řešení. Nyní budeme potřebovat pro další výklad zavést pojem řídkost.

Definice 2.1. [4] Vektor $\mathbf{x}$ nazveme *k-řídkým*, pokud platí:

$$\|\mathbf{x}\|_0 \leq k.$$

(Definice normy viz definice C.1).

Nás budou zajímat pouze řídká řešení – řešení s co nejvíce nulovými složkami. Jinak řečeno, chceme, aby $\mathbf{x}$ mělo minimální nultou normu. Matematicky zapsáno jde o tento problém:

$$\text{(P0)} \qquad \min_{\mathbf{x}} \|\mathbf{x}\|_0 \text{ vzhledem k } \mathbf{Ax} = \mathbf{y}.$$

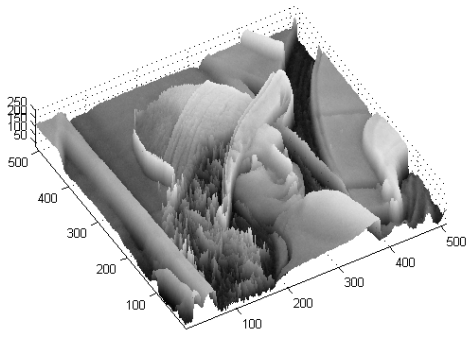
Známe vektor $\mathbf{y} \in \mathbb{R}^m$ a matici $\mathbf{A} \in \mathbb{R}^{m \times n}$, $m < n$. Matice $\mathbf{A}$ se nazývá slovník a její sloupce atomy.

Protože obraz počítač reprezentuje jako matici, jejíž prvky jsou hodnoty určité škály barev (viz obr. 2.1), vyvstává zde otázka, jak ekvivalentně sestavit systém lineárních rovnic. Vektor $\mathbf{y}$ dostaneme tak, že obraz (označme ho $\mathbf{Y}$) „naskládáme“ po sloupcích za sebe do jednoho vektoru (zvektorizujeme jej). Slovník $\mathbf{A}$ musíme také ekvivalentně upravit pomocí Kroneckerova součinu (viz definice C.2): nový slovník je $\mathbf{I} \otimes \mathbf{A}$, kde $\mathbf{I}$ je jednotková matice.

Obraz také může být zašuměný a přesné řešení neexistuje. Proto v praxi uvažujeme odchylku od přesného řešení neboli:

$$\min_{\mathbf{x}} \|\mathbf{x}\|_0 \text{ vzhledem k } \quad \|\mathbf{Ax} - \mathbf{y}\|_2 \leq \epsilon,$$

kde ϵ je povolená odchylka od přesného řešení. [4]



Obr 2.1: Reprezentace obrazu v počítači

3. Postačující podmínky pro jednoznačnost řešení

3.1. Spark matice

Pro jednoznačnost řešení je nutné zavést pojem *spark* (doslova „jiskra“).

Definice 3.1. [4] Číslo $spark(\mathbf{A})$ definujeme jako nejmenší počet sloupců matice $\mathbf{A}$, které jsou lineárně závislé. Formálně:

$$spark(\mathbf{A}) = \min_{\mathbf{z} \in ker\mathbf{A}, \mathbf{z} \neq 0} \|\mathbf{z}\|_0 ,$$

kde $ker\mathbf{A}$ značí jádro lineárního zobrazení určeného maticí $\mathbf{A}$.

Pro nenulovou matici $\mathbf{A} \in \mathbb{R}^{m \times n}$, kde $m < n$, platí, že spark může nabývat hodnot $spark(\mathbf{A}) \in \{2, \ldots, m+1\}$, přičemž hodnoty 2 je dosaženo, když je jeden sloupec přímo násobkem jiného.

Nyní následuje tvrzení, které poskytuje postačující podmínku pro jednoznačnost řešení.

Tvrzení 3.1. [4] *Pokud má soustava* $\mathbf{Ax} = \mathbf{y}$ *řešení* $\mathbf{x}$ *splňující*

$$\|\mathbf{x}\|_0 < \frac{spark(\mathbf{A})}{2},$$

pak $\mathbf{x}$ *je nutně nejřidší možné řešení a žádné jiné řešení se stejnou řídkostí neexistuje.*

Nalezení $spark(\mathbf{A})$ je bohužel výpočtově velmi náročné, proto se snažíme nalézt jiné podmínky zaručující jednoznačnost řešení.

3.2. Vzájemná koherence

Definice 3.2. [4] *Vzájemná koherence (mutual coherence)* matice $\mathbf{A}$ je definována jako největší absolutní normovaný skalární součin dvou různých sloupců matice $\mathbf{A}$:

$$\mu(\mathbf{A}) = \max_{1 \leq j,k \leq n, j \neq k} \frac{\left|\mathbf{a}_j^T \mathbf{a}_k\right|}{\|\mathbf{a}_j\|_2 \cdot \|\mathbf{a}_k\|_2},$$

kde $\mathbf{a}_j$ označuje j-tý sloupec matice $\mathbf{A}$.

Jedině ortogonální matice mají nulovou koherenci, protože její sloupce jsou po dvou ortogonální – tedy čitatel je roven nule, zatímco nedourčená soustava bude mít vždy nenulovou koherenci.

Tvrzení 3.2. [4] *Pro libovolnou matici* $\mathbf{A}$ *platí:*

$$spark(\mathbf{A}) \geq 1 + \frac{1}{\mu(\mathbf{A})}.$$

Tato podmínka nám umožňuje zdola ohraničit $spark(\mathbf{A})$ a přitom není zdaleka tak výpočtově náročná. Z toho plyne:

Tvrzení 3.3. [4] *Pokud má soustava* $\boldsymbol{Ax} = \boldsymbol{y}$ *řešení* $\boldsymbol{x}$ *splňující*

$$\|\mathbf{x}\|_0 < \frac{1}{2}\left(1 + \frac{1}{\mu(\mathbf{A})}\right),$$

pak $\mathbf{x}$ *je nutně nejřidší možné a jediné takové. Navíc tohoto řešení lze dosáhnout pomocí* l_1*-minimalizace (vysvětleno dále viz 5.2).*

Naší snahou je tedy používání maximálně nekoherentních slovníků, které se nejvíce přibližují ortogonální matici. [4][2]

4. Volba slovníku

Nyní se zamyslíme nad tím, jak zvolit slovník, aby řídké řešení existovalo a bylo co nejřidší. Často se využívá spojení dvou ortobází – slovník má tedy $2n$ sloupců a n řádků. Jako báze se bere například jednotková matice (tj. Diracova báze v časové oblasti), matice diskrétní Fourierovy transformace (můžeme použít i reálnou variantu – matici diskrétní kosinové transformace) nebo matice diskrétní vlnkové transformace. Také se využívají tzv. framy nebo se využívá součinu několika matic. [4]

4.1. Diskrétní kosinová transformace a její matice

Jak již bylo řečeno, diskrétní kosinová transformace (DCT) je podobná diskrétní Fourierově transformaci, ale produkuje pouze reálné koeficienty. Označíme-li $\mathbf{Y}$ původní signál a $\mathbf{X}$ transformovaný signál, pak koeficienty transformace můžeme přímo spočítat:

- v jednorozměrném případě:

$$\mathbf{X}_i = \sum_{j=0}^{n-1} \mathbf{Y}_j \cos\left(\frac{\pi\,(2j+1)\,i}{2n}\right),$$

  kde n značí délku vektoru

- ve dvourozměrném případě můžeme transformaci počítat jako sérii dvou po sobě jdoucích jednorozměrných transformací (nejprve provedeme transformaci po řádcích, poté po sloupcích) nebo přímo:

$$\mathbf{X}_{i_1,i_2} = \sum_{j_1=0}^{n_1-1} \sum_{j_2=0}^{n_2-1} \mathbf{Y}_{j_1,j_2} \cos\left(\frac{\pi\,(2j_1+1)\,i_1}{2n_1}\right) \cos\left(\frac{\pi\,(2j_2+1)\,i_2}{2n_2}\right),$$

  kde n_1 a n_2 značí velikost transformované matice (obrazu).

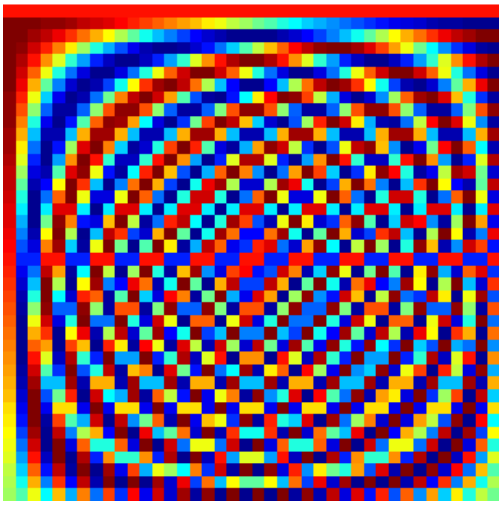
Diskrétní kosinovou transformaci lze ale také spočítat pomocí matice DCT – označíme ji $\mathbf{D}$. Pak transformovaný signál $\mathbf{X} = \mathbf{DY}$. Tato matice má v řádcích „naskládané kosiny“ s různou frekvencí (od nulové po $n-1$) v intervalu $\langle 0, \pi \rangle$, hodnoty těchto kosinů jsou pouze diskrétní. Formálně tedy prvky DCT matice $\mathbf{D}$ vypadají takto:

$$\mathbf{D}_{i,j} = \cos\left(\frac{\pi\,(2j+1)\,i}{2n}\right).$$

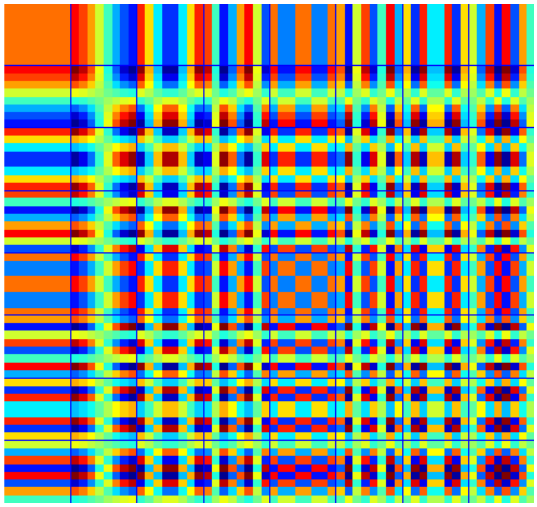
Na obrázku 3.2 máme graficky zobrazenou DCT matici pomocí programu Matlab. Tato matice má tu výhodu, že je ortonormální.

Analogicky pro dvourozměrný případ můžeme sestavit dvourozměrnou bázi DCT (viz obr. 3.2, tato matice byla vykreslena pomocí vytvořeného programu *vykresleni_DCT2*), kde kosiny jsou v matici „naskládány“ ve dvou směrech – vodorovně i svisle. Nejčastěji se ale používají 2 jednorozměrné DCT matice – výsledný transformovaný obraz dostaneme pomocí [8]:

$$\mathbf{X} = \mathbf{DYD}^T.$$



Obr 4.1: Báze DCT pro vektory délky 40



Obr 4.2: Dvourozměrná báze DCT pro obrazy velikosti 8×8

Poznámka. Existuje také diskrétní vlnková transformace, která je podobná DCT. Nepoužívá však k výpočtu kosinus, ale tzv. vlnky (wavelety). Příkladem těchto vlnek by mohl být haar wawelet nebo třída vlnek Daubechies. Matice diskrétní vlnkové transformace má v řádcích „naskládané“ vlnky (jednoho typu), které jsou různě posunuté a roztažené. [6]

4.2. Framy

Frame je množina generátorů vektorového prostoru, přičemž počet generátorů prostoru je větší než je jeho dimenze. Generátory jsou tak nutně lineárně závislé, což je ale výhodné pro reprezentaci signálů. Framy jsou méně omezené než báze a jsou více flexibilní. Nevýhodou jsou náročnější výpočtové metody, protože při počítání s framy hrozí velké riziko numerické nestability. Nyní následuje definice framu:

Definice 4.1. [7] Nechť $\{\mathbf{u}_k\}_{k\in\mathbb{I}}$ je nejvýše spočetná množina vektorů vektorového prostoru $\mathbb{V}$, $\mathbb{I}$ je množina indexů. Pak tato množina tvoří *frame*, pokud existují konstanty $0 < A \leq B < \infty$ takové, že platí:

$$A \left\|\mathbf{x}\right\|_2^2 \leq \sum_{k\in\mathbb{I}} \left|\mathbf{x}^T \mathbf{u}_k\right|^2 \leq B \left\|\mathbf{x}\right\|_2^2, \qquad \forall \mathbf{x} \in \mathbb{V}.$$

Konstanty A a B se obvykle nazývají meze framu. Optimální dolní mez pak definujeme jako supremum všech dolních mezí a optimální horní mez jako infimum všech horních mezí. Framy, pro které platí $A = B$, nazýváme těsné. Pro reprezentaci signálů se využívají právě těsné framy. [7]

5. Metody řešení

Předpokládejme, že $spark(\mathbf{A}) > 2k_0$ a existuje k_0-řídké řešení soustavy. Pak je toto řešení nejřidší možné a jednoznačné. Pro nalezení přesného řešení bychom museli projít všech $\binom{m}{k_0}$ kombinací podmnožin atomů matice, což je obvykle časově velmi náročné. Proto se vyvinulo několik aproximačních metod řešení tohoto problému, které nejsou tolik přesné, ale zato jsou rychlejší. Tyto metody se dělí do dvou hlavních kategorií – hladové algoritmy a relaxační algoritmy. [4]

5.1. Hladové algoritmy

5.1.1. Hlavní myšlenka

Předpokládejme, že slovník A má $spark(\mathbf{A}) > 2$ a existuje řídké řešení soustavy lineárních rovnic $\mathbf{Ax} = \mathbf{y}$ s řídkostí $k = 1$. To znamená, že $\mathbf{y}$ je skalárním násobkem některého sloupce matice $\mathbf{A}$. Tento atom můžeme nalézt pomocí m kroků. V j-tém kroku minimalizujeme chybu $\epsilon(j) = \|\mathbf{a}_j z_j - \mathbf{y}\|_2$. Takto dostaneme optimální volbu konstanty $z_j^* = \mathbf{a}_j^T \mathbf{y} / \|\mathbf{a}_j\|_2^2$. Pokud se chyba rovná nule, našli jsme hledané řešení.

Ve většině případů ovšem nemáme jenom řešení s řídkostí 1, ale s řídkostí $k = k_0$. Předpokládejme tedy, že slovík $\mathbf{A}$ má $spark(\mathbf{A}) = 2k_0$. Pak hledané řešení je nejřidší možné a je zaručena jeho jednoznačnost. Jak bylo uvedeno výše, hledat přesné řešení je výpočtově velmi náročné, proto se pokouší hladové algoritmy iterativně nalézt v každém kroku jeden atom, který má největší podíl na řešení (tj. má nejmenší reziduální ℓ_2 chybu při aproximaci $\mathbf{y}$). Na začátku položíme $\mathbf{x}^0 = 0$ a v každém kroku aproximujeme řídké řešení tak, aby nenulové prvky $\mathbf{x}$ příslušely množině aktivních atomů (na začátku je tato množina prázdná). V každé iteraci přidáme do této množiny vybraný atom. Tento postup není přesný, proto musíme nastavit maximální odchylku od přesného řešení, se kterou jsme spokojeni. [1]

5.1.2. Ortogonální sdružovací metoda (OMP)

Ortogonální sdružovací metoda (Orthogonal Matching Pursuit) je jedna z nejpoužívanějších hladových algoritmů, proto ji uvedeme jako první. Jak bylo předesláno výše, v tomto algoritmu položíme prvotní aproximaci řídkého řešení $\mathbf{x}^0 = 0$, spočítáme počáteční reziduum $\mathbf{r}^0 = \mathbf{y} - \mathbf{Ax}^0 = \mathbf{y}$. Samozřejmě na začátku také množina všech nenulových koeficientů S^0 je prázdná (této množině budeme říkat nosič). Potom můžeme přistoupit k jednotlivým krokům metody. Nejprve se vyjádří reziduální chyba při aproximaci pro všechna $j = 1, 2, \ldots, n$:

$$\epsilon(j) = \min_{z_j} \left\|\mathbf{a}_j z_j - \mathbf{r}^{k-1}\right\|_2^2 = \left\| \frac{\mathbf{a}_j^T \mathbf{r}^{k-1}}{\|\mathbf{a}_j\|_2^2} \mathbf{a}_j - \mathbf{r}^{k-1} \right\|_2^2 =$$

$$= \left\|\mathbf{r}^{k-1}\right\|_2^2 - 2\frac{(\mathbf{a}_j^T \mathbf{r}^{k-1})^2}{\|\mathbf{a}_j\|_2^2} + \frac{(\mathbf{a}_j^T \mathbf{r}^{k-1})^2}{\|\mathbf{a}_j\|_2^2} =$$

$$= \left\|\mathbf{r}^{k-1}\right\|_2^2 - \frac{(\mathbf{a}_j^T \mathbf{r}^{k-1})^2}{\|\mathbf{a}_j\|_2^2}.$$

Z těchto vypočítaných chyb vybereme nejmenší z nich. Označme index nejmenší chyby j_0, formálně zapsáno: $\forall j \notin S^{k-1} : \epsilon(j_0) \leq \epsilon(j)$. Poté přidáme do nosiče S^k nalezený index j_0. Dále spočítáme x^k pomocí minimalizace $\|\mathbf{Ax} - \mathbf{y}\|_2^2$, ale nesmíme zapomenout, že x^k je nenulové pouze na pozicích S^k. To nás vede k tomu, že $\mathbf{x}_{S^k} = \mathbf{A}_{S^k}^{+}\mathbf{y}$, kde $\mathbf{x}_{S^k}$ je nenulová část $\mathbf{x}$ a $\mathbf{A}_{S^k}$ je matice, která obsahuje pouze sloupečky matice $\mathbf{A}$ odpovídající nosiči S^k (znak + nad maticí označuje pseudoinverzi – viz C.3). Nakonec spočítáme nové reziduum – pokud je menší než požadovaná odchylka, máme hledané řešení při dané míře nepřesnosti, jinak provedeme další iteraci. [1]

5.1.3. Sdružovací metoda (MP)

Další metoda zvaná sdružovací metoda (Matching Pursuit) je velice podobná OMP, ale je o něco rychlejší za cenu menší přesnosti. Tato metoda také nejprve spočítá chyby $\epsilon(j)$ a nalezne index j_0. Řídké řešení $\mathbf{x}$ ale nepočítá pomocí metody nejmenších čtverců, jako tomu bylo u OMP, ale nechává nezměněné všechny jeho nenulové složky z předchozí iterace, pouze přidá nový koeficient do nově nalezené nenulové složky vektoru. Za tento koeficient volíme optimální konstantu $z_{j_0}^*$. [1]

5.1.4. Normování

Oba již dříve uvedené hladové algoritmy (OMP a MP) byly popsány pro obecnou matici $\mathbf{A}$, která nemá normované sloupce. Můžeme ale normovat sloupce pomocí $\tilde{\mathbf{A}} = \mathbf{AW}$, kde $\mathbf{W}$ je diagonální matice, která má na diagonále $1/\|\mathbf{a}_i\|_2$. Potom můžeme v předchozích algoritmech použít $\tilde{\mathbf{A}}$ místo $\mathbf{A}$. Je to výhodné zejména proto, že místo hledání největší chyby v každé iteraci $\epsilon(j)$ stačí nalézt největší (v absolutní hodnotě) skalární součin rezidua $\mathbf{r}^{k-1}$ a normovaného vektoru matice $\tilde{\mathbf{A}}$. To značně urychlí výpočet. Když tedy použijeme k výpočtu $\tilde{\mathbf{A}}$, nalezneme řídký vektor $\tilde{\mathbf{x}}$. Nás obvykle ale zajímá vektor původní matice – k získání původního řídkého vektoru $\mathbf{x}$ musíme $\tilde{\mathbf{x}}$ zleva vynásobit maticí $\mathbf{W}$ [1]:

$$\mathbf{y} = \mathbf{A}\tilde{\mathbf{x}} = \mathbf{AW}\tilde{\mathbf{x}} = \mathbf{Ax} \Longrightarrow \mathbf{x} = \mathbf{W}\tilde{\mathbf{x}}.$$

5.2. Relaxace

Norma ℓ_0 není konvexní funkce, tudíž není možné na náš problém použít žádnou z řady metod a algoritmů konvexní optimalizace, které jsou dnes k dispozici. Protože ale ℓ_p normy jsou konvexní pro $p \geq 1$, nabízí se nám otázka, zda by se nedala nultá norma aproximovat nejbližší konvexní normou – tedy ℓ_1. Tím by se náš problém převedl na úlohu:

$$\text{(P1)} \qquad \min_{\mathbf{x}} \|\mathbf{x}\|_1 \text{ vzhledem k } \mathbf{Ax} = \mathbf{y}.$$

Jak ukážeme za chvíli, za určitých podmínek lze použít ℓ_1 normu místo ℓ_0. Navíc se často řešení obou úloh shodují. [4]

5.2.1. Podmínky ekvivalence řešení pomocí ℓ_0- a ℓ_1-minimalizace

Vlastnost nulového prostoru

Pro $T \subset \{1, \dots, n\}$ označíme $\mathbf{x}_T \in \mathbb{R}^n$ vektor odvozený z $\mathbf{x} \in \mathbb{R}^n$ tak, že prvky na pozicích patřících do množiny T zachováme a ostatní vynulujeme. Komplement T označíme $T^c = \{1, \dots, n\} \setminus T$. Nyní následuje definice vlastnosti nulového prostoru:

Definice 5.1. [4]
Řekneme, že matice $\mathbf{A} \in \mathbb{R}^{m \times n}$ má *vlastnost nulového prostoru* řádu k s konstantou $\gamma \in (0, 1)$, pokud platí:

$$\|\mathbf{u}_T\|_1 \leq \gamma \|\mathbf{u}_{T^c}\|_1$$

pro všechny množiny $T \subset \{1, \dots, n\}$, $|T| \leq k$ a pro všechny vektory $\mathbf{u} \in ker\mathbf{A}$.

Vlastnost nulového prostoru umožňuje nalezení k-řídkého řešení pomocí ℓ_1 minimalizace a zaručuje jeho jednoznačnost. Ověření této podmínky ale není úplně jednoduché. Protože reálné signály nejsou řídké v pravém slova smyslu, ale mají místo nulových složek malé nenulové hodnoty, budeme potřebovat definovat chybu aproximace řídkým vektorem:

Definice 5.2. [4] *Chyba nejlepší aproximace* vektoru $\mathbf{x} \in \mathbb{R}^n$ k-řídkým vektorem $\mathbf{z}$ v normě ℓ_p je definována jako

$$\sigma_k(\mathbf{x})_p = \inf_{\mathbf{z} \in \Sigma_k^n} \|\mathbf{x} - \mathbf{z}\|_p ,$$

kde $\Sigma_k^n = \{\mathbf{x} \in \mathbb{R}^n : \|\mathbf{x}\|_0 \leq k\}$.

Následující tvrzení udává horní odhad chyby i v ostatních případech:

Tvrzení 5.1. [4] *Nechť matice $\mathbf{A} \in \mathbb{R}^{m \times n}$ má vlastnost nulového prostoru řádu k s konstantou $\gamma \in (0, 1)$. Nechť $\mathbf{x} \in \mathbb{R}^n$, $\mathbf{y} = \mathbf{A}\mathbf{x}$ a $\mathbf{x}^* \in \mathbb{R}^n$ je řešení ℓ_1 minimalizace. Potom*

$$\|\mathbf{x} - \mathbf{x}^*\|_1 \leq \frac{2(1+\gamma)}{1-\gamma} \sigma_k(\mathbf{x})_1.$$

Pokud tedy existuje nějaké nejvýše k-řídké řešení soustavy $\mathbf{y} = \mathbf{A}\mathbf{x}$ za splnění předpokladů tvrzení, pak $\sigma_k(\mathbf{x})_1 = 0$ a nulovost pravé strany vynutí $\mathbf{x} = \mathbf{x}^*$ (díky tomu, že $\|\mathbf{x} - \mathbf{x}^*\|_1 = 0$). To znamená, že hledané řídké řešení nalezneme i ℓ_1 optimalizací.

Zároveň platí obrácená implikace – pokud je možné ze soustavy $\mathbf{y} = \mathbf{A}\mathbf{x}$ rekonstruovat všechny k-řídké vektory $\mathbf{x}$ pomocí ℓ_1-minimalizace, pak matice $\mathbf{A}$ má vlastnost nulového prostoru řádu k s nějakou konstantou $\gamma \in (0, 1)$. [4]

Vlastnost zeslabené izometrie

Další možností, jak zajistit ekvivalenci řešení pomocí ℓ_0- a ℓ_1-minimalizace, je definovat vlastnost zeslabené izometrie, která je oproti vlastnosti nulového prostoru výpočtově přijatelnější a navíc je stabilní i pod vlivem šumu.

Definice 5.3. [4] *Konstanta zeslabené izometrie* δ_k matice $\mathbf{A} \in \mathbb{R}^{m\times n}$ je nejmenší číslo takové, že platí:

$$(1-\delta_k) \leq \frac{\|\mathbf{Az}\|_2^2}{\|\mathbf{z}\|_2^2} \leq (1+\delta_k)$$

pro všechny vektory $\mathbf{z} \in \Sigma_k^n$. Řekneme, že matice $\mathbf{A}$ má vlastost zeslabené izometrie řádu k s konstantou δ_k, pokud $\delta_k \in (0,1)$.

Zeslabení izometrie (= lineární zobrazení záchovávající délku vektorů) spočívá v omezení se jen na všechny podmatice $\mathbf{A}$ o k sloupcích a dále nepožaduje přesnou izometrii, ale povoluje malou odchylku δ_k. To znamená, že všechny matice musejí být přibližně ortogonální.

Do definice vlastnosti zeslabené izometrie je nutné obecně zahrnout všechny podmatice o maximálně k sloupcích, protože dopředu nevíme, které prvky vektoru $\mathbf{x}$ budou nenulové, a proto také nevíme, které atomy matice $\mathbf{A}$ se budou podílet na reprezentaci signálu $\mathbf{y}$.

Konstantu zeslabené izometrie můžeme spočítat přímo:

$$\delta_k = \max_{T\subset\{1,\dots,n\},|T|\leq k} \left\|\mathbf{A}_T^T\mathbf{A}_T - \mathbf{I}\right\|_{2\to 2},$$

kde norma matice $\|\cdot\|_{2\to 2}$ je definována jako:

$$\|\mathbf{B}\|_{2\to 2} = \max_{\mathbf{x}} \frac{\|\mathbf{Bx}\|_2}{\|\mathbf{x}\|_2}.$$

Podobně jako u vlastnosti nulového prostoru lze i u vlastnosti zeslabené izometrie určit chybu aproximace ℓ_1-minimalizací. Také lze nalézt vzájemný vztah mezi vlastností zeslabené izometrie a vlastností nulového prostoru.

Tvrzení 5.2. [4] *Nechť* $\mathbf{A} \in \mathbb{R}^{m\times n}$ *má vlastnost zeslabené izometrie řádu* $K = k + h$ *s konstantou* $\delta_K \in (0,1)$. *Pak* $\mathbf{A}$ *má vlastnost nulového prostoru řádu* k *s konstantou*

$$\gamma = \sqrt{\frac{k\,(1+\delta_K)}{h\,(1-\delta_K)}}.$$

Dále můžeme pomocí vlastnosti zeslabené izometrie omezit chybu aproximace řešení ℓ_1-minimalizací.

Tvrzení 5.3. [4] *Nechť* $\mathbf{A} \in \mathbb{R}^{m\times n}$ *má vlastnost zeslabené izometrie řádu* $3k$ *s konstantou* $\delta_{3k} < \frac{1}{3}$. *Nechť* $\mathbf{Ax} = \mathbf{y}$ *pro* $\mathbf{x} \in \mathbb{R}^n$ *a* $\mathbf{x}^* \in \mathbb{R}^n$ *je řešení* ℓ_1*-minimalizace. Pak*

$$\|\mathbf{x} - \mathbf{x}^*\|_2 \leq C\frac{\sigma_k\,(\mathbf{x})_1}{\sqrt{k}},$$

kde C *je konstanta závisející pouze na* δ_{3k}.

Tvrzení 5.4. [4] *Nechť* $\mathbf{A}$ *má vlastnost zeslabené izometrie řádu* $2k$ *s konstantou* δ_{2k}, $\delta_{2k} < \sqrt{2} - 1 \approx 0,4142$. *Pak*

$$\|\mathbf{x} - \mathbf{x}^*\|_2 \leq C\frac{\sigma_k\,(\mathbf{x})_1}{\sqrt{k}}$$

a

$$\|\mathbf{x} - \mathbf{x}^*\|_1 \leq C\sigma_k\,(\mathbf{x})_1$$

pro nějakou konstantu C.

Další tvrzení specifikuje vlastnosti v případě bílého šumu (Gaussovského).

Tvrzení 5.5. [4] *Předpokládejme, že matice* $\mathbf{A} \in \mathbb{R}^{m\times n}$ *má vlastnost zeslabené izometrie řádu* $2k$ *s konstantou*

$$\delta_{2k} < \frac{2}{3+\sqrt{\frac{7}{4}}} \approx 0,4627.$$

Následující vztahy platí pro všechna $\mathbf{x} \in \mathbb{R}^n$*. Nechť signál je naměřen se šumem* $\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{e}$*,* $\|\mathbf{e}\|_2 \leq \epsilon$*,* $\mathbf{x}^*$ *je řešením úlohy:*

$$\min_{\mathbf{z}} \|\mathbf{z}\|_1 \;\; \textit{vzhledem k} \quad \|\mathbf{A}\mathbf{z} - \mathbf{y}\|_2 \leq \epsilon.$$

Potom

$$\|\mathbf{x} - \mathbf{x}^*\|_2 \leq C_1 \frac{\sigma_k(\mathbf{x})_1}{\sqrt{k}} + C_2\epsilon$$

pro kladné konstanty C_1 *a* C_2 *závisející pouze na* δ_{2k}*.*

Jak je vidět, celková chyba má 2 části: část, která závisí pouze na řídkosti, a část, která závisí na velikosti šumu.

Je zajímavé, že se zatím nepodařilo deterministicky sestavit matice, které by měly vlastnost zeslabené izomerie s předem danými parametry. Zatím se podařilo nalézt pouze matice, které s vysokou pravděpodobností tuto vlastnost splňují. To ale nevadí – tvrzení neříkají nic o tom, jestli je ℓ_1-minimalizace ekvivalentní s ℓ_0-minimalizací v případě, kdy matice $\mathbf{A}$ nemá vlastnost zeslabené izometrie. Experimentálně bylo zjištěno, že lze použít i některé matice, které tuto vlastnost nemají. [4]

5.2.2. Iterativně převažovaná metoda nejmenších čtverců (IRLS)

Jak již bylo řečeno, další možností, jak nalézt řídké řešení soustavy $\mathbf{A}\mathbf{x} = \mathbf{y}$, je použít relaxaci ℓ_0 normy. Speciálně pro tyto účely byla vyvinuta iterativně převažovaná metoda nejmenších čtverců (Iterative Reweighed Least Squares), která reprezentuje ℓ_p normu (pro pevně dané $p \in (0,1\rangle$) pomocí vážené ℓ_2 normy. Předpokládejme, že máme určené přibližné řešení $\mathbf{x}_{k-1}$ a položme $\mathbf{X}_{k-1} = diag\left(|\mathbf{x}_{k-1}|^q\right)$. Uvažujme, že tato matice je invertibilní a $\left\|\mathbf{X}_{k-1}^{-1}\mathbf{x}\right\|_2^2$ je rovno $\|\mathbf{x}\|_{2-2q}^{2-2q}$. Jestliže zvolíme $q = 1 - p/2$, uvedený výraz nahrazuje ℓ_p normu, $\|\mathbf{x}\|_p^p$. Nyní použijeme místo inverze $\mathbf{X}_{k-1}$ pseudoinverzi, $\left\|\mathbf{X}_{k-1}^{+}\mathbf{x}\right\|_2^2$. Pokusíme se nyní řešit tento problém:

$$\min_{\mathbf{x}} \left\|\mathbf{X}_{k-1}^{+}\mathbf{x}\right\|_2^2 \text{ vzhledem k } \mathbf{A}\mathbf{x} = \mathbf{y}.$$

Tento problém má tu výhodu, že jej lze řešit pomocí lineární algebry a Lagrangeových multiplikátorů. Jako účelovou funkci využijeme klasickou ℓ_2 normu. Tedy máme

$$L(\mathbf{x}) = \left\|\mathbf{X}_{k-1}^{+}\mathbf{x}\right\|_2^2 + \lambda^T(\mathbf{y} - \mathbf{A}\mathbf{x})$$

$$\Rightarrow \frac{\partial L(\mathbf{x})}{\partial \mathbf{x}} = \mathbf{0} = 2\left(\mathbf{X}_{k-1}^{+}\right)^2\mathbf{x} - \mathbf{A}^T\lambda.$$

Pokud budeme uvažovat, že $\mathbf{X}_{k-1}^{+}$ je invertibilní, znamená to, že má nenulové prvky na hlavní diagonále. [1] Můžeme proto položit $\left(\mathbf{X}_{k-1}^{+}\right)^{-1} = \mathbf{X}_{k-1}$. V tomto případě můžeme psát řešení ve tvaru:

$$\mathbf{x}_k = 0{,}5\mathbf{X}_{k-1}^2\mathbf{A}^T\lambda.$$

V obecném případě, kdy jsou některé prvky na hlavní diagonále $\mathbf{X}_{k-1}$ nulové, řešení dané těmito nulovými prvky odpovídá složkám v $\mathbf{x}_k$. Za předpokladu, že nulové složky $\mathbf{x}_k$ mohou zůstat v dalších iteracích (kvůli řídkosti řešení), tento vzorec na výpočet $\mathbf{x}_k$ je smysluplný a dává správný výsledek. Nyní se vrátíme k rovnici $\mathbf{Ax} = \mathbf{y}$, do které dosadíme právě spočítané $\mathbf{x}_k$. Dostaneme:

$$0{,}5\mathbf{A}\mathbf{X}_{k-1}^2\mathbf{A}^T\lambda = \mathbf{y} \qquad \Rightarrow \qquad \lambda = 2\left(\mathbf{A}\mathbf{X}_{k-1}^2\mathbf{A}^T\right)^{-1}\mathbf{y}.$$

Zase musíme být opatrní při používání inverze – proto raději použijeme pseudoinverzi. Celkově tedy máme:

$$\mathbf{x}_k = \mathbf{X}_{k-1}^2\mathbf{A}^T\left(\mathbf{A}\mathbf{X}_{k-1}^2\mathbf{A}^T\right)^{+}\mathbf{y}.$$

Přibližné řešení $\mathbf{x}_k$ použijeme pro vytvoření nové diagonální matice $\mathbf{X}_k$ a můžeme začít novou iteraci.

V praxi se ukázalo, že jakmile je nějaká složka vektoru $\mathbf{x}_k$ nulová, v algoritmu nikdy nedojde k tomu, že by byla nahrazena nenulovou složkou. Proto se za startovací vektor nebere vektor nulový, ale vektor, jehož složky jsou rovny jedné. Na rozdíl od předchozích algoritmů výpočet řešení končí, jakmile $\|\mathbf{x}_k - \mathbf{x}_{k-1}\|_2$ je menší než zadaná požadovaná přesnost. Nejčastěji volíme $p = 1$.

Nesmíme ovšem zapomenout, že zatímco ℓ_0 norma nerozlišuje velikost nenulových složek $\mathbf{x}_k$, ℓ_p norma na velikosti složek přímo závisí. Proto algoritmus vybírá za nenulové složky vektoru $\mathbf{x}_k$ ty, které násobí v ℓ_p normě největší atomy matice $\mathbf{A}$, a tím je přesné řešení ovlivněno. Z tohoto důvodu se matice $\mathbf{A}$ před vlastním výpočtem normuje. Aby bylo nalezené řídké řešení normované matice $\mathbf{A}$ ekvivalentní s řešením původní matice, na konci výpočtu (stejně jako tomu bylo u hladových algoritmů) se musí po složkách vydělit příslušnými normami matice $\mathbf{A}$. [1]

5.2.3. Modifikované IRLS – s uvažováním odchylky od přesného řešení

Jak již bylo řečeno dříve, signál může být zašuměný a přesné řešení nemusí existovat. V případě relaxace se tedy pokoušíme řešit tento problém:

$$\min_{\mathbf{x}} \|\mathbf{x}\|_1 \text{ vzhledem k } \quad \|\mathbf{Ax} - \mathbf{y}\|_2 \leq \epsilon.$$

Využitím příslušných Lagrangeových multiplikátorů dostaneme:

$$\min_{\mathbf{x}} \lambda \|\mathbf{x}\|_1 + \frac{1}{2}\|\mathbf{y} - \mathbf{Ax}\|_2^2,$$

kde Langrangeův multiplikátor λ je funkcí $\mathbf{A}$, $\mathbf{y}$ a ϵ. V případě IRLS položíme $\mathbf{X} = diag(|\mathbf{x}|)$ a dostaneme $\|\mathbf{x}\|_1 = \mathbf{x}^T\mathbf{X}^{-1}\mathbf{x}$. Jestliže máme danou aproximaci $\mathbf{x}_{k-1}$, nastavíme $\mathbf{X}_{k-1} = diag(|\mathbf{x}_{k-1}|)$ a pokusíme se řešit:

$$\min_{\mathbf{x}} \lambda\mathbf{x}^T\mathbf{X}^{-1}\mathbf{x} + \frac{1}{2}\|\mathbf{y} - \mathbf{Ax}\|_2^2,$$

což je kvadratický optimalizační problém řešitelný pomocí klasické lineární algebry.

Největším problémem je zvolení λ tak, aby nalezené řešení co nejvíce odpovídalo přesnému řešení. Volba λ závisí na směrodatné odchylce i řídkosti řešení. Jak toto λ zvolit jsem se pro určité případy pokusila simulovat v kapitole 7.4. Někdy se také λ aproximuje a postupně se iterativně zpřesňuje. Vzniklý systém lineárních rovnic se často řeší pomocí metody sdružených gradientů. [1]

5.3. Další algoritmy

Existují také další algoritmy: algoritmy založené na prahování a hybridní algoritmy, které využívají jednotlivých předností z obou předchozích skupin. Tyto algoritmy však podrobněji nebudeme popisovat, protože je dále nevyužíváme. [4]

6. Metody pro nalezení slovníku k řídkému vyjádření signálů

Všechny doposud uvedené algoritmy pracovaly s předem daným slovníkem. Nabízí se otázka, zda by se nedal nalézt pro určitou známou kategorii signálů $\{\mathbf{y}_i\}_{i=1}^M$ (= trénovací databáze) slovník, ve kterém by tyto signály byly co nejřidší.

Abychom mohli vytvořit tento slovník, předpokládejme tedy, že ϵ – odchylka modelu – je známá a naším cílem je odhad slovníku $\mathbf{A}$. Uvažujme následující optimalizační problém:

$$\min_{\mathbf{A},\{\mathbf{x}_i\}_{i=1}^M} \sum_{i=1}^{M} \|\mathbf{x}_i\|_0 \quad \text{vzhledem k} \quad \|\mathbf{y}_i - \mathbf{A}\mathbf{x}_i\|_2 \leq \epsilon, \qquad 1 \leq i \leq M.$$

Tento problém popisuje každý signál $\mathbf{y}_i$ pomocí $\mathbf{x}_i$, což je nejřidší možná reprezentace tohoto signálu v neznámém slovníku $\mathbf{A}$. Nejvhodnější slovník pak nalezneme vyřešením tohoto problému. Pokud dostaneme řešení, které má pro všechny složky řídkost minimálně k_o, našli jsme i hledaný slovník.

Můžeme však problém otočit:

$$\min_{\mathbf{A},\{\mathbf{x}_i\}_{i=1}^M} \sum_{i=1}^{M} \|\mathbf{y}_i - \mathbf{A}\mathbf{x}_i\|_2^2 \quad \text{vzhledem k} \quad \|\mathbf{x}_i\|_0 \leq \epsilon, \qquad 1 \leq i \leq M.$$

Dále zavedeme následující označení: jako $\mathbf{Y}$ označíme matici, ve které jsou naskládány všechny vektory $\mathbf{y}_i$, obdobně označíme $\mathbf{X}$ matici příslušných $\mathbf{x}_i$, $\mathbf{X}$ má řídké sloupečky.

Pro hledání nejvhodnějšího slovníku existuje několik algoritmů. Stručně si uvedeme dva základní algoritmy. [1]

6.1. Metoda optimálních směrů (MOD)

Metoda optimálních směrů (Method of Optimal Directions) se dívá na uvedený problém jako na složený minimalizační problém: vnitřní minimalizace se týká počtu nenulových prvků jednotlivých reprezentací $\mathbf{x}_i$ s pevně daným slovníkem $\mathbf{A}$ a vnější minimalizace se týká vlastního slovníku $\mathbf{A}$. Vlastní postup je tedy jednoduchý – v k-tém kroku použijeme slovník $\mathbf{A}_{k-1}$. Tak dostaneme matici $\mathbf{X}_k$ a následně hledáme $\mathbf{A}_k$ pomocí metody nejmenších čtverců:

$$\mathbf{A}_k = \arg\min_{\mathbf{A}} \|\mathbf{Y} - \mathbf{A}\mathbf{X}_k\|_F^2 = \mathbf{Y}\mathbf{X}_k^T \left(\mathbf{X}_k\mathbf{X}_k^T\right)^{-1} = \mathbf{Y}\mathbf{X}_k^+,$$

kde počítáme chybu pomocí Frobeniovy normy. Tento postup opakujeme, dokud nejsme spokojeni s přesností. [1]

6.2. K-SVD

Metoda K-SVD oproti MOD hledá slovník $\mathbf{A}$ ne jako celek, ale postupně určuje každý atom slovníku zvlášť. Necháme-li všechny atomy stejné kromě j_0-tého ($\mathbf{a}_{j_0}$), můžeme tento sloupec aktualizovat pomocí koeficientů z $\mathbf{X}$, které ho násobí. Ve výrazu $\|\mathbf{Y} - \mathbf{A}\mathbf{X}_k\|_F^2$ osamostatníme $\mathbf{a}_{j_0}$ a dostaneme:

$$(*) \qquad \|\mathbf{Y} - \mathbf{A}\mathbf{X}_k\|_F^2 = \left\|\mathbf{Y} - \sum_{j=1}^{m} \mathbf{a}_j \mathbf{x}_j^T\right\|_F^2 = \left\|\left(\mathbf{Y} - \sum_{j \neq j_0} \mathbf{a}_j \mathbf{x}_j^T\right) - \mathbf{a}_{j_0}\mathbf{x}_{j_0}^T\right\|_F^2,$$

kde $\mathbf{x}_j^T$ znamená j-tý řádek $\mathbf{X}$. Naším cílem je získat nové vyjádření jak $\mathbf{a}_{j_0}$, tak $\mathbf{x}_{j_0}^T$, které se vztahuje k výrazu v závorce – k chybové matici:

$$\mathbf{E}_{j_0} = \mathbf{Y} - \sum_{j \neq j_0} \mathbf{a}_j \mathbf{x}_j^T.$$

Optimální $\mathbf{a}_{j_0}$ a $\mathbf{x}_{j_0}^T$ minimalizující (*) můžeme dostat pomocí SVD rozkladu chybové matice $\mathbf{E}_{j_0}$, což ale většinou vede k získání hustého vektoru $\mathbf{x}_{j_0}^T$, a tím i k vzrůstu počtu nenulových prvků v $\mathbf{X}$.

Abychom ponechali počet nenulových prvků ve všech $\mathbf{x}_j^T$ kromě j_0-tého při minimalizaci (*), použijeme pouze podmnožinu sloupců $\mathbf{E}_{j_0}$, které odpovídají těm signálům z trénovací databáze, které využívají j_0-tý atom – totiž ty sloupečky, ve kterých jsou složky řádku $\mathbf{x}_j^T$ nenulové. Tímto omezením dovolíme pouze změny nenulových složek $\mathbf{x}_j^T$.

Proto použijeme omezující operátor $\mathbf{P}_{j_0}$, kterým vynásobíme $\mathbf{E}_{j_0}$ zprava, abychom ponechali pouze vybrané sloupce. Matice $\mathbf{P}_{j_0}$ má M řádků (počet signálů v trénovací databázi) a M_{j_0} sloupců (počet signálů využívajících j_0-tý atom). Definujme $(\mathbf{x}_{j_0}^R)^T$, $(\mathbf{x}_{j_0}^R)^T = \mathbf{x}_{j_0}^T\mathbf{P}_{j_0}$ jako zúžení řádku $\mathbf{x}_{j_0}^T$ vybráním pouze nenulových složek. K získání $\mathbf{a}_{j_0}$ a $\mathbf{x}_{j_0}^T$ můžeme použít SVD rozklad podmatice $\mathbf{E}_{j_0}\mathbf{P}_{j_0}$.

Na začátku algoritmu slovník $\mathbf{A}$ náhodně vygenerujeme a jeho atomy znormujeme. K aproximaci řešení použijeme některý z dříve uvedených algoritmů (např. MP, OMP). Po provedení SVD rozkladu ($\mathbf{E}_{j_0}^R = \mathbf{U}\mathbf{\Delta}\mathbf{V}^\mathbf{T}$) nastavíme $\mathbf{a}_{j_0} = \mathbf{u}_1$ a $\mathbf{x}_{j_0}^R = \mathbf{\Delta}\,[1,1]\,\mathbf{v}_1$.

Existuje i další možnost jak spočítat $\mathbf{a}_{j_0}$ a $\mathbf{x}_{j_0}^T$: pro dané $\mathbf{a}_{j_0}$ nalezneme $\mathbf{x}_{j_0}^T$ pomocí metody nejmenších čtverců:

$$\min_{\mathbf{x}_{j_0}^R} \left\|\mathbf{E}_{j_0}\mathbf{P}_{j_0} - \mathbf{a}_{j_0}(\mathbf{x}_{j_0}^R)^T\right\|_F^2 \qquad \Rightarrow \qquad \mathbf{x}_{j_0}^R = \frac{\mathbf{P}_{j_0}^T\mathbf{E}_{j_0}^T\mathbf{a}_{j_0}}{\|\mathbf{a}_{j_0}\|_2^2}.$$

Potom pro spočítané $\mathbf{x}_{j_0}^T$ nalezneme $\mathbf{a}_{j_0}$:

$$\min_{\mathbf{x}_{j_0}^R} \left\|\mathbf{E}_{j_0}\mathbf{P}_{j_0} - \mathbf{a}_{j_0}(\mathbf{x}_{j_0}^R)^T\right\|_F^2 \qquad \Rightarrow \qquad \mathbf{a}_{j_0} = \frac{\mathbf{P}_{j_0}^T\mathbf{E}_{j_0}^T\mathbf{x}_{j_0}^R}{\left\|\mathbf{x}_{j_0}^R\right\|_2^2}.$$

Takto dostaneme postupně celý hledaný slovník $\mathbf{A}$. [1]

Poznámka. Když máme barevný obraz, používáme uvedené postupy pro každou ze tří základních barev zvlášť. Pomocí K-SVD lze odšumovat obrazy – v tomto případě se ukazuje, že je výhodnější odšumovat všechny základní barvy dohromady. [5]

7. Empirické ověření teorie v programu Matlab

V této kapitole si názorně ukážeme některé dříve uvedené postupy pomocí softwaru Matlab (ve verzi R2010b). Ve své bakalářské práci jsem pro tento účel vytvořila programy *test, rekonstrukce_zasumeneho_signalu, hledani_optimalniho_lambda* a *K_SVD*, které pracují se simulovanými daty, a program *rekonstrukce_obrazu* pro práci s reálnými obrazy. Tyto programy také využívají pomocné funkce. Zdrojové kódy všech programů a funkcí jsou uvedeny na přiloženém CD, jejich seznam naleznete v příloze B.

7.1. Vygenerování řídkého obrazu v bázi a následná zpětná rekonstrukce pomocí algoritmů MP, OMP a IRLS

Pro vygenerování náhodného řídkého obrazu a následnou zpětnou rekonstrukci pomocí algoritmů MP, OMP a IRLS jsem vytvořila program *test*. V něm nejdříve nastavíme vstupní parametry: řídkost, rozměry obrazu a bázi, ve které bude tento obraz řídký. Jako bázi můžeme použít DCT nebo vlnky (haar, db1 – db10). Dále nastavíme toleranci, s jakou má být obraz následně rekonstruován.

Pomocí funkce *generovani_ridkeho_signalu* vygenerujeme k-řídký obraz ve zvolené bázi (viz obr. 7.1). V této funkci se nejdříve vygenerují řídké koeficienty, poté se sestaví dvě bázové matice – každá řádu jednoho rozměru matice koeficientů. K tomuto účelu jsem vytvořila funkci *vytvoreni_baze*, která využívá k sestavení vlnkových matic funkce *w_filters* a *dtwtmatrix* od autorů P. Rajmice a Z. Průšy. Následně se koeficienty těmito maticemi transformují a tím získáme obraz řídký ve zvolené bázi.

```
function [koeficienty,ridkyobraz,baze_m,baze_n] = generovani_ridkeho_signalu(k, m ,n,baze)

%% generování koeficientů

koeficienty_vektor = zeros(m*n,1);
generovani_nenulovych_souradnic = randperm(m*n);
nenulove_souradnice = sort(generovani_nenulovych_souradnic(1:k));
koeficienty_vektor(nenulove_souradnice) = randi(255,k,1);
koeficienty = reshape(koeficienty_vektor,m,n);

%% generování signálu

[baze_m,baze_n] =  vytvoreni_baze(baze,m,n);
ridkyobraz = baze_m' * koeficienty * baze_n;

end
```

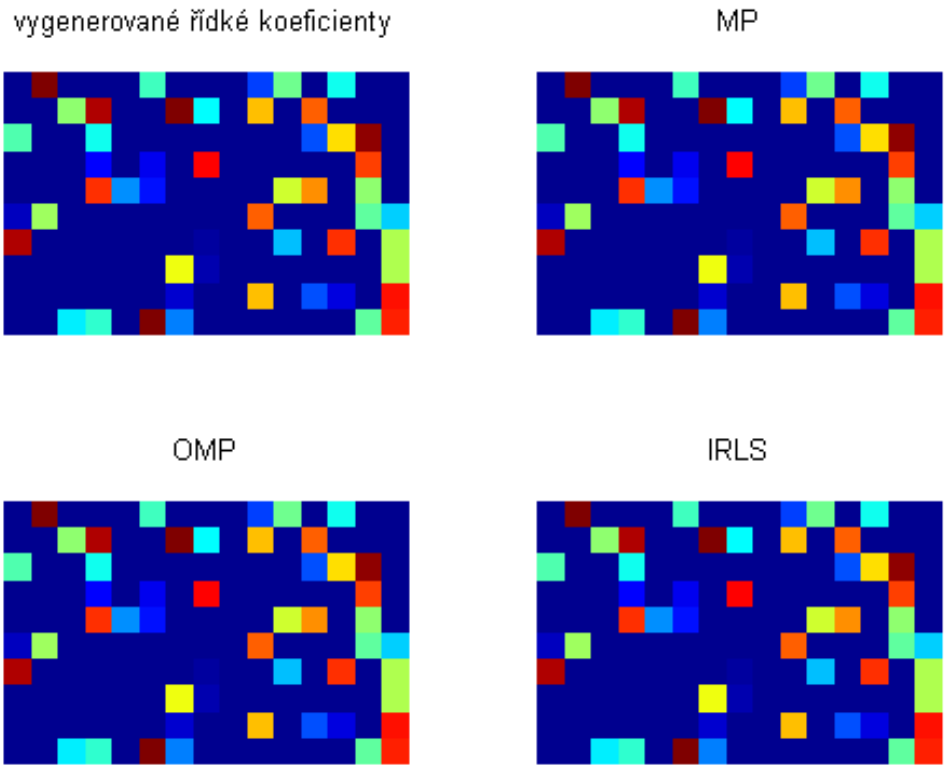
Obr 7.1: Funkce vygenerování obrazu řídkého v zadané bázi

Nyní máme vygenerovaný řídký obraz v některé bázi. Jak již bylo dříve uvedeno, tento obraz kvůli výpočtům zvektorizujeme. Ale pro vytvoření tohoto obrazu jsme přitom použili dvě matice. Jak z nich tedy můžeme získat slovník? Odpověď se mi podařilo

nalézt v článku [3] – slovník dostaneme pomocí Kroneckerova součinu: označíme-li $\mathbf{B_m}$ první bázovou matici a druhou matici $\mathbf{B}_n$, pak slovník $\mathbf{A} = \mathbf{B}_n^T \otimes \mathbf{B}_m^T$.

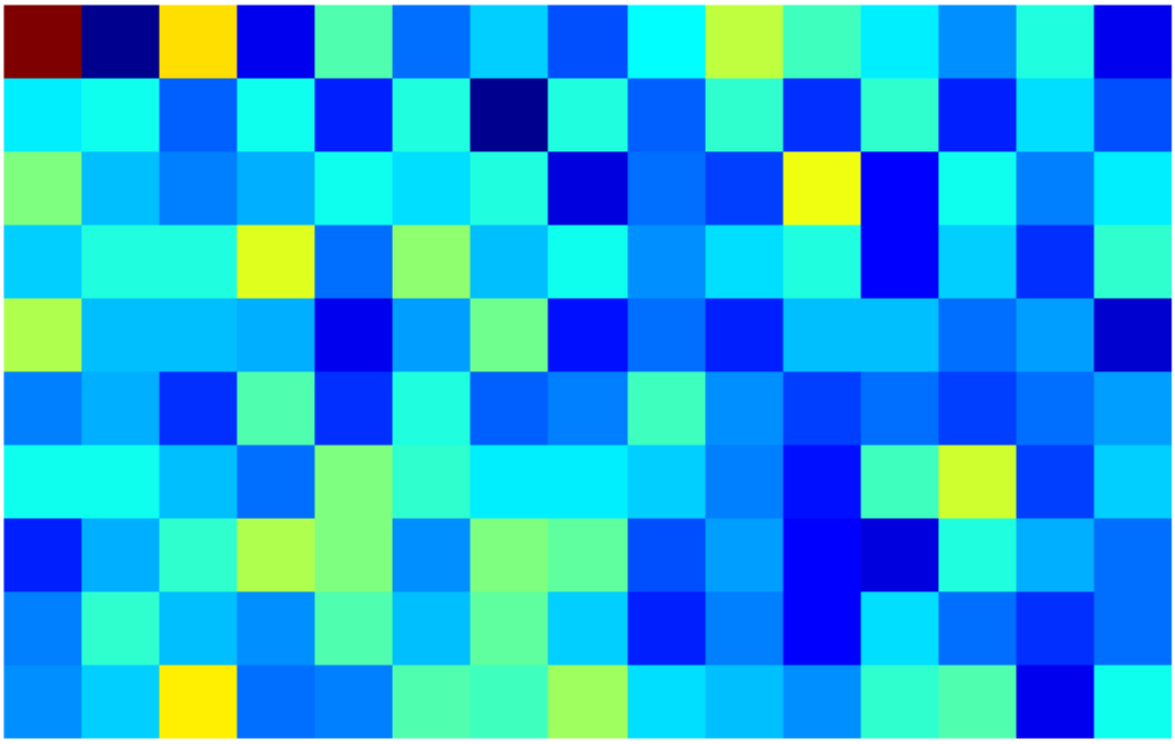
Teď již můžeme přistoupit k samotnému hledání řídké reprezentace obrazu – program postupně hledá původní koeficienty třemi již dříve uváděnými algoritmy, konkrétně pomocí MP, OMP a IRLS (tyto funkce jsem rovněž naprogramovala, všechny před vlastním výpočtem kontrolují normovanost slovníku a popřípadě ho normují pomocí funkce *normovani_slovniku*). Po výpočtu se výsledná řídká reprezentace obrazu opět přeskládá z vektoru do matice původních rozměrů. Na konci programu se počítá úspěšnost rekonstrukce, kterou program měří pomocí relativní chyby rekonstrukce. Dále se také měří úspěšnost pomocí PSNR (rovněž vytvořená funkce).

Výsledky pro konkrétní nastavení počátečních parametrů (řídkost $k = 50$, rozměry matice 10×15, báze DCT, tolerance 0,001) můžete vidět na obrázku 7.2. Pro lepší názornost jsou na tomto obrázku vykresleny pouze koeficienty. Samotný vygenerovaný obraz můžete spatřit na obr. 7.3.



Obr 7.2: Vlevo nahoře jsou znázorněny vygenerované řídké koeficienty, které byly transformovány bázemi DCT. Následující obrazy znázorňují koeficienty získané zpětně pomocí jednotlivých algoritmů MP, OMP a IRLS

Relativní chyby rekonstrukce jednotlivými metodami jsou uvedeny v tabulce 7.1. V této tabulce jsou také uvedeny PSNR pro jednotlivé případy. Jak je vidět, algoritmy mají přibližně stejnou přesnost a dávají dobré výsledky.



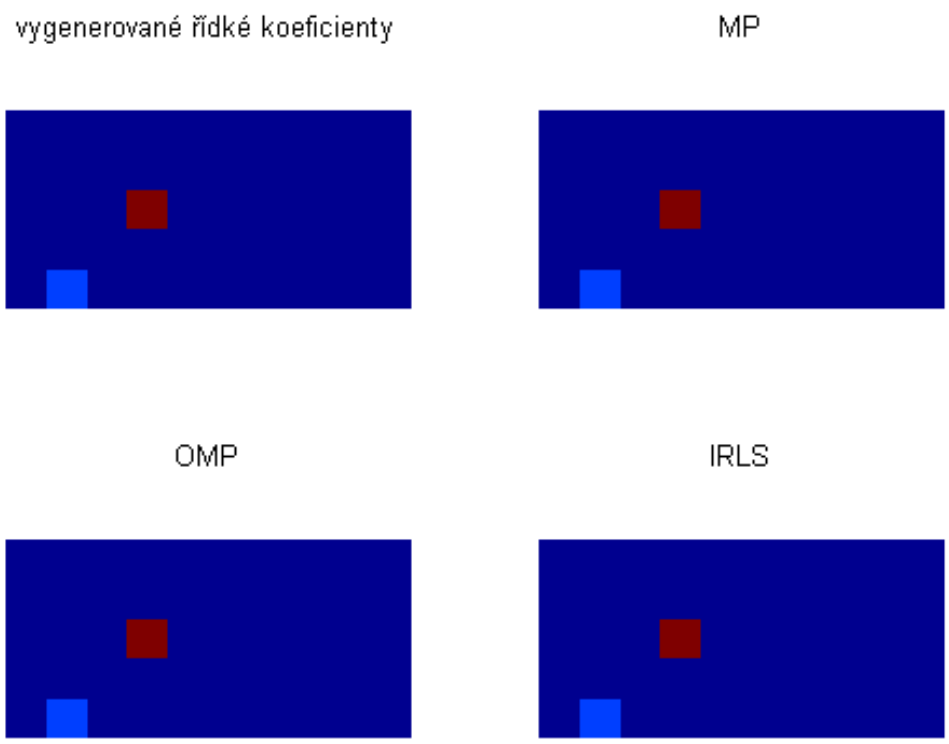
Obr 7.3: Vygenerovaný obraz řídký v bázi DCT

	MP	OMP	IRLS
relativní chyba rekonstrukce	$4{,}3356 \cdot 10^{-13}$	$8{,}5622 \cdot 10^{-13}$	$9{,}9723 \cdot 10^{-13}$
PSNR	313,0588 dB	306,343 dB	304,6791 dB

Tab 7.1: Tabulka přesnosti řešení jednotlivých metod při konkrétním nastavení

Další možností tohoto programu je simulovat měření obrazu pomocí nějakého přístroje. Měření se provádí vynásobením obrazu (který je řídký v zadané bázi) zleva a zprava tzv. měřicími maticemi (označíme je $\mathbf{M}_m$ a $\mathbf{M}_n$). Tyto matice mají náhodné prvky z normovaného normálního rozdělení. $\mathbf{M}_m$ má méně řádků než sloupců a $\mathbf{M}_n$ naopak více řádků nežli sloupců. Pro simulaci měření uživatel v programu nastaví pomocnou proměnnou *mereni* na hodnotu *true* a dále zadá poměr menšího rozměru měřicí matice ku většímu (pro obě matice používáme stejný poměr). Pak slovník $\mathbf{A}$ dostaneme jako: $\mathbf{A} = \left(\mathbf{M}_n^T \mathbf{B}_n^T\right) \otimes \left(\mathbf{M}_m \mathbf{B}_m^T\right)$. Dále program funguje stejně.

Dále jsem zkoušela úspěšnost rekonstrukce obrazu (zvoleného rozměru 5×10, báze DCT a povolené maximální odchylky 0,001) pro tři různá nastavení. Nejprve jsem zvolila větší řídkost ($k = 2$) a nastavila jsem poměr stran měřicí matice na 3/4. Protože obraz byl dostatečně řídký a bylo provedeno velké množství měření, rekonstrukce se většinou povedla všem třem algoritmům. Konkrétní výsledek můžete vidět na obr. 7.4.



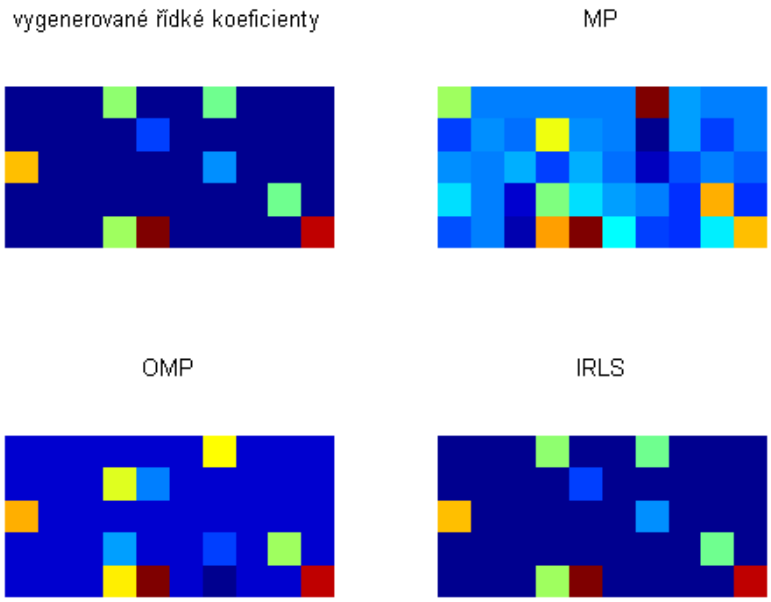
Obr 7.4: Simulace měření: při dostatečné řídkosti a velkém počtu měření se obraz většinou podaří rekonstruovat (postupně PSNR pro MP, OMP a IRLS = 159, 322 a 133 dB)

Ve druhém případě jsem ponechala rozměry měřicí matice a zvedla jsem počet koeficientů obrazu na $k = 10$. Tentokrát obraz není tolik řídký, proto se rekonstrukce ne vždy povede. Výsledky jedné z provedených rekonstrukcí můžete vidět na obr. 7.5.

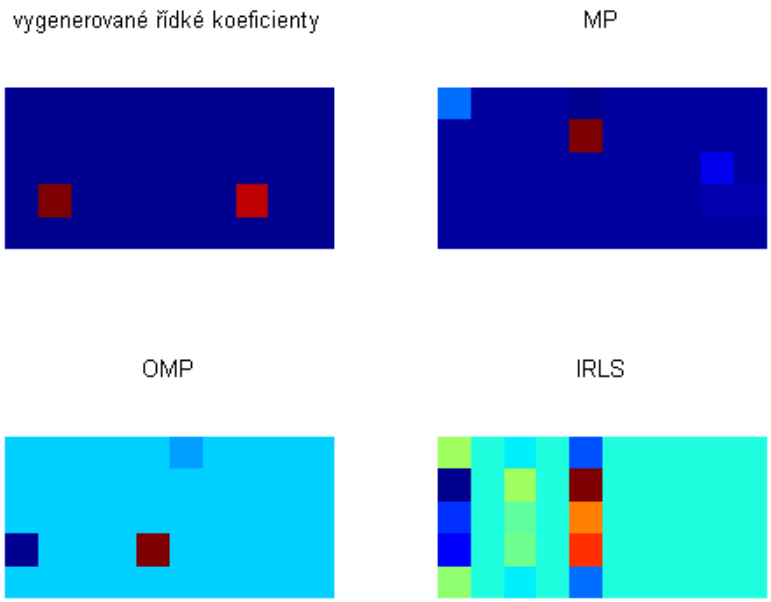
Nakonec jsem nastavila řídkost zpátky na $k = 2$, ale zmenšila jsem měřicí matici na poměr stran 1/3. Také při tomto nastavení se už nepodaří obrazy téměř vůbec rekontruovat. Výsledky jedné z provedených rekonstrukcí můžete vidět na obr. 7.6.

Abychom mohli objektivně porovnat, při jaké volbě řídkosti a poměru stran měřicích matic je rekonstrukce úspěšná, vytvořila jsem program *uspesnost_rekonstrukce_obrazu*, který pro zadané parametry měření postupně počítá úspěšnost rekonstrukce (několikrát ji zopakuje a zprůměruje) a vykreslí závislost úspěšnosti rekonstrukce na řídkosti a poměru stran měřicí matice. Výsledný graf pro obraz rozměru 15×15 můžete vidět na obrázku 7.7. Byla přitom použita metoda OMP, báze DCT, 10 opakování, povolená maximální nepřesnost 0,1. Rekonstrukci jsme považovali za úspěšnou, když relativní chyba byla menší než 0,1.

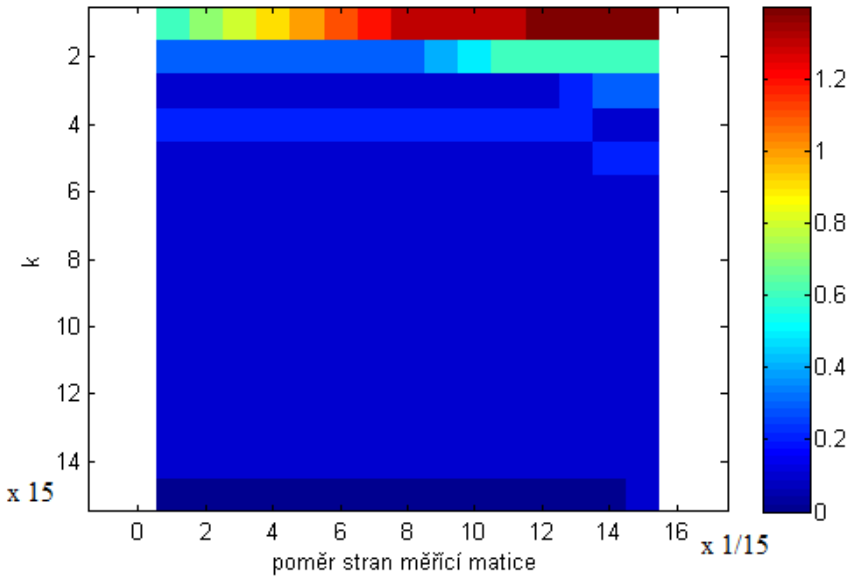
Čím máme tedy řidší signál a větší měřicí matici, tím je úspěšnost rekonstrukce vyšší ve více případech. Se zmenšující se měřicí maticí nebo se zvětšujícím se počtem nenulových koeficientů klesá počet případů, kdy byla rekonstrukce úspěšně provedena, dokonce může být nemožná. Můžeme vidět, že při simulaci měření dávají tyto tři algoritmy rozdílné výsledky.



Obr 7.5: Simulace měření: při velkém počtu měření a menší řídkosti se obraz už nepodaří vždy rekonstruovat (postupně PSNR pro MP, OMP a IRLS = 17, 23 a 101 dB)



Obr 7.6: Simulace měření: při malém počtu měření a velké řídkosti se obraz také nepodaří vždy rekonstruovat (postupně PSNR pro MP, OMP a IRLS = 5, 7 a 13 dB)



Obr 7.7: Závislost úspěšnosti rekonstrukce obrazu na jeho řídkosti a poměru stran měřicích matic (OMP)

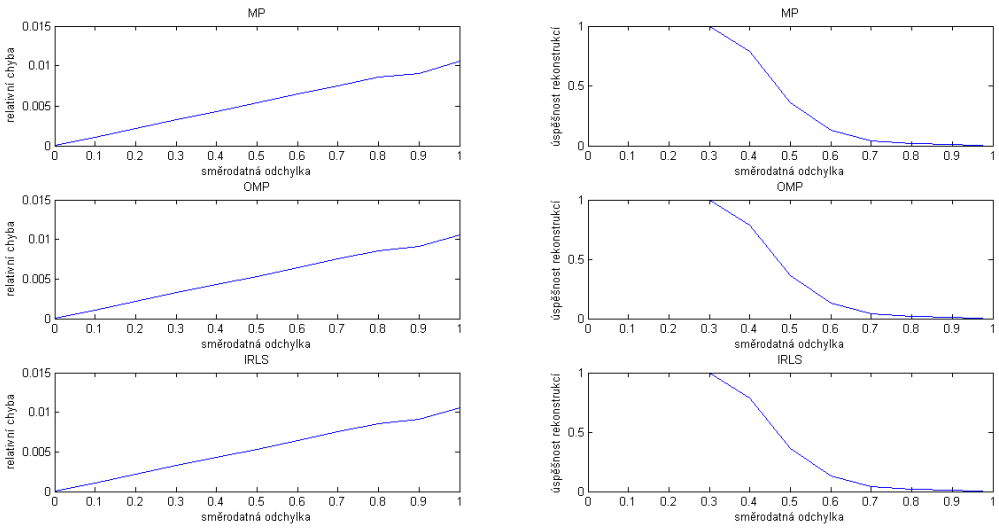
7.2. Vliv šumu na rekonstrukci obrazu

Jako další jsem postupně přidávala do obrazu šum a sledovala jsem, jaký to má vliv na jeho rekonstrukci. K tomuto účelu jsem vytvořila program *rekonstrukce_zasumeneho_signalu*. V tomto případě také nejprve zadáme vstupní parametry – řídkost, rozměry obrazu, bázi, toleranci, s jakou se má hledat původní obraz, a hodnotu odchylky nalezeného obrazu od původního, kdy jsme ještě s řešením spokojeni. Také nastavíme, které hodnoty směrodatné odchylky bílého Gaussovského šumu má program postupně přidat do původního obrazu. Jako poslední nastavení zvolíme, kolikrát chceme s každou odchylkou počítat. Stejně jako v předchozím případě se vytvoří řídký obraz v zadané bázi a sestaví se slovník.

Poté se provede několik cyklů (podle počtu zadaných směrodatných odchylek), ve kterých se nejdříve přidá k transformovanému obrazu bílý šum s jednotlivými směrodatnými odchylkami a následně se provede rekonstrukce pomocí MP, OMP a IRLS. Vždy spočítáme relativní chybu rekonstrukce jednotlivými algoritmy. Aby tento výsledek byl objektivnější, opakujeme tento postup vždy několikrát (na začátku jsme nastavili kolikrát). Potom tuto chybu zprůměrujeme. Zároveň při těchto cyklech počítáme, jestli rekonstrukce proběhla s tolerancí, kterou jsme chtěli (měříme pomocí relativní chyby rekonstrukce). Jestliže proběhla tak, abychom byli spokojeni, přiřadíme rekonstrukci úspěšnost rovnou jedné. Jestliže takto neproběhla, přiřadíme ji úspěšnost nulovou.

Zprůměrujeme úspěšnosti pro jednotlivé směrodatné odchylky. Na konci programu se vykreslí spočítaná závislost relativní chyby na směrodatné odchylce bílého šumu a závislost úspěšnosti rekonstrukce na odchylce pro všechny tři algoritmy.

Na obrázku 7.8 jsem znázornila výstup tohoto programu pro konkrétní parametry – řídkost $k = 7$, rozměry obrazu 3×5, báze DCT, tolerance 0,01. Směrodatná odchylka šumu byla postupně nastavena $0 - 1$ a pro každou z nich se cyklus $100\times$ opakoval. Hranice spokojenosti s výsledkem byla nastavena na 0,005. Z grafu je názorně vidět, že pro všechny algoritmy s rostoucí směrodatnou odchylkou narůstá relativní chyba rekonstrukce. Přibližně do hodnoty 0,3 směrodatné odchylky byla rekonstrukce zcela úspěšná, potom



Obr 7.8: Vliv šumu na úspěšnost rekonstrukce

postupně klesá a při hodnotě 0,9 přestává být zcela úspěšná (při námi zvolené toleranci). Také lze vidět, že všechny tři algoritmy mají stejnou úspěšnost rekonstrukce.

7.3. Algoritmy aplikované na reálné obrazy

Nyní vyzkoušíme rozklad obrazu na řídké koeficienty v bázi na reálném obrazu. V tomto případě ale může nastat (oproti předchozím případům) určitý problém. Reálný obraz je obvykle větších rozměrů a při výpočtu si s ním už neporadí paměť počítače. Proto se obraz rozdělí na menší čtvercové díly (můžeme zadat rozměr), se kterými pak počítáme odděleně. Také kvůli velikosti reálného obrazu jsem pro výpočet použila metodu MP, protože je ze všech tří algoritmů nejrychlejší.

Samotný program (*rekonstrukce_obrazu*) nejprve načte zadaný obraz. Můžeme nastavit, na jak velké čtverce chceme obraz rozdělit, přesnost a bázi, se kterou chceme počítat. Potom nastavíme, zda chceme oříznout koeficienty na požadovanou řídkost (aby byl obraz řídký v zadané bázi) či přidat bílý šum se zadanou směrodatnou odchylkou. Můžeme obě nastavení také kombinovat.

```
% nastavení řídkosti
ridkost = false;
if ridkost
   k = 10;
end

% nastavení zašumění
zasumeni = true;
if zasumeni
   smerodatna_odchylka = 0.1;
end
```

Obr 7.9: Nastavení řídkosti obrazu a jeho šumu (zvolíme-li u zadaných položek true, provede se požadovaná akce, při použití false nikoliv)

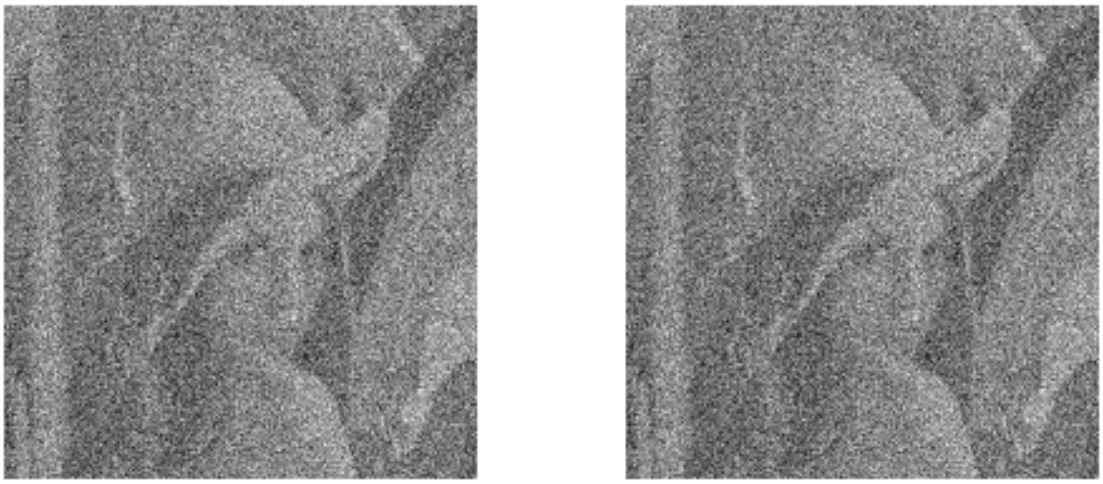
Po úpravě obrazu na zadané parametry dostaneme obraz, se kterým nadále pracujeme. Hledáme jeho řídkou reprezentaci v zadané bázi po částech (využíváme k tomu úměrně menší slovníky). Po skončení výpočtů program vykreslí jak výchozí obraz, tak i nalezenou řídkou reprezentaci (abychom je mohli porovnat, je řídká reprezentace na konci výpočtu vynásobena slovníkem). Také se spočítá relativní chyba a PSNR obou obrazů.

Vyzkoušela jsem v tomto programu rekonstruovat obrázek *Lena* v bázi haar. Dělila jsem tento obrázek na čtverce o straně 8 pixelů s povolenou nepřesností 0,1. Rekonstrukce proběhla dobře – vysledný obraz (viz obr 7.10) má PSNR = 87 dB. Potom jsem ke stejnému obrázku přidala bílý šum se směrodatnou odchylkou 0,1 a poté 0,3. Se zvětšující se odchylkou se chyba rekonstrukce zvětšuje – PSNR vyšlo postupně 68 dB a 58 dB. Výsledky jsou vidět postupně na obrázcích 7.11 a 7.12.

Jako další příklad jsem rekonstruovala při stejné toleranci obrázek *Terezka*. Tento obrázek jsem dělila na díly rozměru 5×5 a k výpočtu jsem použila bázi DCT. Výsledný obraz (viz obr. 7.13) má PSNR 83 dB. Při stejných parametrech jsem pak přidávala šum se směrodatnými odchylkami 0,1 a 0,3 k obrázku *křeslo*. Výsledky dopadly podobně (viz obr. 7.14 a 7.15) jako u obrázku *Leny* – PSNR vyšlo postupně 67 dB a 58 dB.


Obr 7.10: Lena: Vlevo výchozí obraz, vpravo zrekonstruovaný obraz, použitá báze haar, PSNR = 87 dB

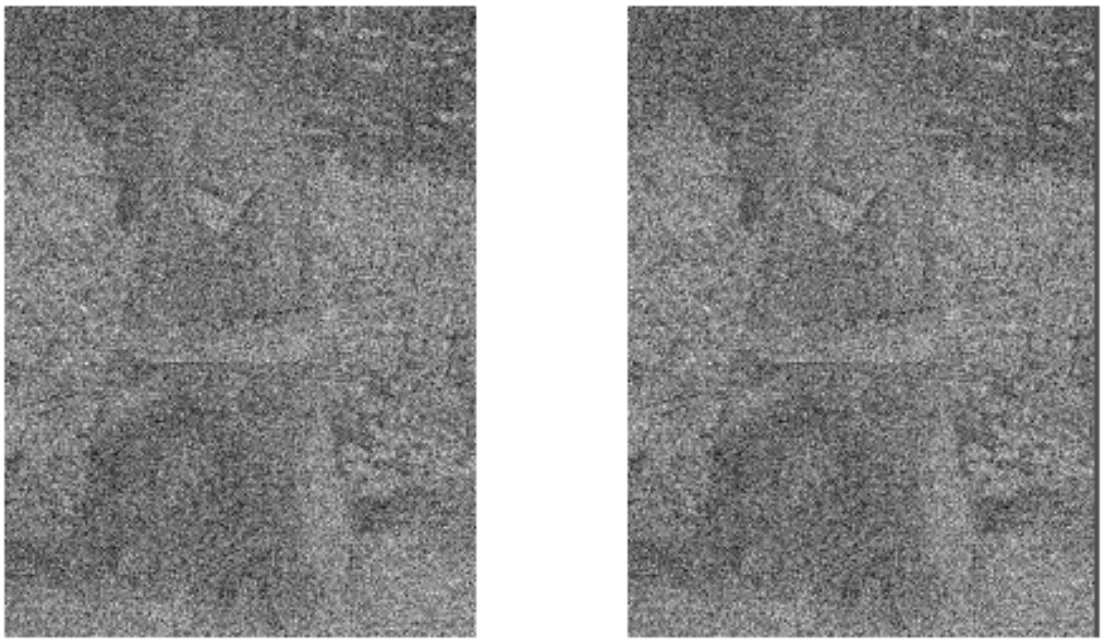

Obr 7.11: Lena: Vlevo výchozí obraz, vpravo zrekonstruovaný obraz, použitá báze haar, směrodatná odchylka bílého šumu = 0,1,
PSNR = 68 dB



Obr 7.12: Lena: Vlevo výchozí obraz, vpravo zrekonstruovaný obraz, použitá báze haar, směrodatná odchylka bílého šumu = 0,3,
PSNR = 58 dB

Obr 7.13: Terezka: Vlevo výchozí obraz, vpravo zrekonstruovaný obraz, použitá báze DCT, PSNR = 83 dB

Obr 7.14: Křeslo: Vlevo výchozí obraz, vpravo zrekonstruovaný obraz, použitá báze DCT, směrodatná odchylka bílého šumu = 0,1,
PSNR = 67 dB

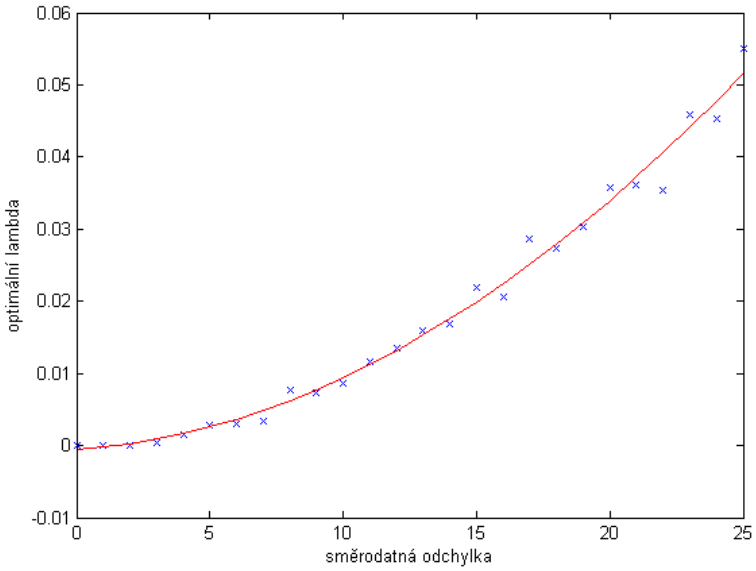


Obr 7.15: Křeslo: Vlevo výchozí obraz, vpravo zrekonstruovaný obraz, použitá báze DCT, směrodatná odchylka bílého šumu = 0,3,
PSNR = 58 dB

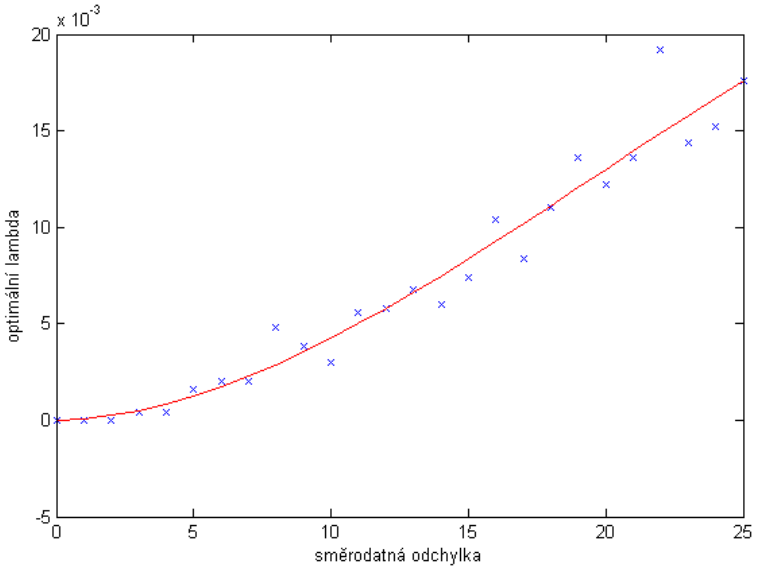
7.4. Hledání optimálního parametru λ v modifikované metodě IRLS, která uvažuje odchylku od přesného řešení

Jako další funkci jsem naprogramovala modifikovanou metodu IRLS (*IRLSe*), která uvažuje odchylku od přesného řešení. Program *hledani_optimalniho_lambda* se snaží nalézt optimální parametr, se kterým dosáhne řešení nejmenší relativní chyby. Na začátku programu nastavíme požadovanou řídkost obrazu, jeho rozměry, bázi, ve které má být řídký, povolenou maximální nepřesnost, rozsah směrodatných odchylek, se kterými chceme, aby program počítal, počet opakování a velikost kroku při výpočtu λ. Potom se vygeneruje řídký obraz (pomocí funkce *generovani_ridkeho_signalu*). Postupně potom přidáváme daný šum a hledáme řídké řešení pomocí *IRLSe* s různě nastavenými $\lambda \in \langle 0, 1\rangle$ (počítáme po nastaveném kroku). Přitom počítáme relativní chybu rekonstrukce. Nalezneme λ příslušné minimální relativní chybě a po proběhnutí nastaveného počtu opakování je zprůměrujeme. Program toto nalezené optimální λ vypíše na obrazovku. Na konci programu se vykreslí závislost optimálního λ na směrodatné odchylce šumu. Tyto hodnoty jsou pak následně proloženy regresní kubickou parabolou (funkce *prolozeni_krivkou*), protože nejsou přesné kvůli stochastické povaze problému.

Pro konkrétní nastavení rozměrů obrazu 5×10, kdy byla řídkost nastavena na $k = 10$, můžete výsledný graf spatřit na obr. 7.16. Počet opakování pro každou směrodatnou odchylku byl 50, báze DCT, povolená nepřesnost 0,001 a λ se počítalo po kroku 0,01. Je názorně vidět, že se zvyšující se odchylkou je optimální λ vyšší. Dále jsem nastavila všechny koeficienty nenulové při zachování ostatních parametrů stejných. Tento případ je znázorněn na obr. 7.17. Při porovnání obou grafů lze pozorovat, že při vyšší řídkosti je optimální λ vyšší.



Obr 7.16: Závislost volby optimálního lambda na směrodatné odchylce bílého šumu (spočítané hodnoty (znázorněny modrými křížky) jsou proloženy regresní kubickou parabolou (znázorněna červeně))



Obr 7.17: Závislost volby optimálního lambda na směrodatné odchylce bílého šumu (spočítané hodnoty (znázorněny modrými křížky) jsou proloženy regresní kubickou parabolou (znázorněna červeně))

7.5. Simulace použití K-SVD

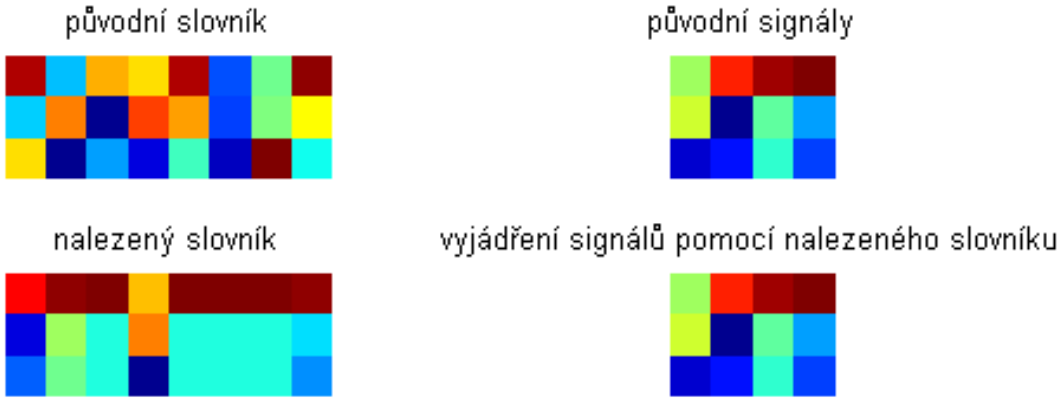
Dále jsem vytvořila program *K_SVD*. Tento program nejprve vygeneruje náhodný slovník (zadaných rozměrů), jehož prvky jsou z normovaného normálního rozdělení. Potom se tento slovník znormuje a následně se vygeneruje zadaný počet jednorozměrných signálů, které jsou lineární kombinací maximálně čtvrtiny atomů slovníku. Tato lineární kombinace je také náhodná. S těmito vygenerovanými signály se pracuje pomocí matice, jejíž sloupečky obsahují ony signály (viz obr. 7.18). Dále se provede samotné K-SVD, které využívá pro svůj výpočet MP kvůli své rychlosti. K-SVD počítá s SVD rozkladem. Na konci programu se vykreslí původní slovník a nově nalezený slovník. Také se zobrazí původní signály a signály vyjádřené pomocí nově nalezeného slovníku. Nově nalezený slovník vypadá vždy jinak než původní (je to způsobeno nejspíše náhodnou tvorbou slovníku), ale nově nalezené signály jsou většinou docela dobrou aproximací signálů původních, ale ne vždy. Také se spočítá norma rozdílu obou vyjádření signálu a jejich PSNR.

```
% generování řídkých signálů vzniklých lineární kombinací několika sloupců slovníku
Y = zeros(m,l);

pocet = randi(n/4);
for i=1:l
   cisla = randperm(n);
   pozice = sort(cisla(1:pocet));
   koef = randi(n/2,pocet,1);
   Y(:,i) = a(:,pozice)*koef;
end
```

Obr 7.18: Generování řídkých signálů

Na obr. 7.19 můžete vidět jeden z výsledků simulace. Slovník má rozměry 3×8 a počet signálů je 4. Rekonstrukce proběhla s nastavením maximální povolené odchylky v MP 0,5. K-SVD bylo v tomto případě úspěšné – PSNR obou vyjádření signálů je 358 dB.



Obr 7.19: Simulace použití K-SVD

8. Závěr

Cílem mé bakalářské práce bylo seznámit se se současným stavem problematiky tzv. řídkých reprezentací obrazů, nastudovat algoritmy pro jejich hledání a vše empiricky ověřit na simulovaných i reálných obrazových datech v softwaru Matlab. Protože je tato problematika velmi obsáhlá, snažila jsem se do této práce vybrat pouze nejdůležitější informace a pojmy, které byly potřebné pro popis řídké reprezentace obrazů.

Ve své práci jsem nejprve stručně uvedla základní problematiku hledání řídkého řešení soustavy lineárních rovnic (tj. problematiku řídké reprezentace obrazů). Poté jsem uvedla podmínky, za kterých toto řešení existuje. Tyto podmínky především kladou nároky na matici soustavy (tzv. slovník), proto jsem se také zmínila o vhodné volbě slovníku. Dále jsem popsala základní algoritmy pro hledání řídké reprezentace a metody pro hledání optimálního slovníku.

Nakonec jsem využití těchto algoritmů prakticky demonstrovala pomocí vlastních programů v softwaru Matlab. Ověřila jsem, že tyto metody fungují jednak pro simulovaná data, tak i pro reálné obrazy. Tyto algoritmy však pracují s jednorozměrným signálem, takže je nutné v průběhu výpočtu obraz převádět na vektor a na konci výpočtu vektor převést zpátky do původní velikosti. Protože obraz obvykle obsahuje velké množství dat, jsou algoritmy výpočtově náročné. Zrychlení algoritmů by se mohlo provést pomocí nově objeveného algoritmu 2D-OMP, který počítá přímo s dvourozměrným signálem.

Tato problematika má velké využití v praxi – můžeme potlačovat šum, odstraňovat rozmazání, efektivně komprimovat data, měnit rozlišení a jiné. Práci by šlo tedy volně rozšířit a věnovat se některé konkrétní aplikaci.

Použité zdroje

[1] ELAD, M.: *Sparse and redundant representations: from theory to applications in signal and image processing.* New York: Springer, 2010. 517 p. 376 s. ISBN 978-1-4419-7010-7.

[2] HRBÁČEK, R.: *Compressive sampling a simulace one-pixel camera:* bakalářská práce. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologi í, Ústav telekomunikací, 2011. 48 s. Vedoucí práce byl Mgr. Pavel Rajmic, Ph.D.

[3] HRBÁČEK, R.; RAJMIC, P.; VESELÝ, V.; aj.: Řídké reprezentace signálů: komprimované snímání. *Elektrorevue – Internetový časopis, 2011, roč. 2011, č. 67, s. 1-8. ISSN: 1213- 1539. URL http://www.elektrorevue.cz/cz/download/ridke-reprezentace-signalu–komprimovane-snimani/*

[4] HRBÁČEK, R.; RAJMIC, P.; VESELÝ, V.; ŠPIŘÍK, J.: Řídké reprezentace signálů: úvod do problematiky. *Elektrorevue - Internetový časopis, 2011, roč. 2011, č. 50, s. 1–10. ISSN: 1213- 1539. URL www.elektrorevue.cz/files/200000751-638ac6484b*

[5] MAIRAL, J.; ELAD, M.; SAPIRO, G.: *Sparse learned representations for image restoration, IASC 2008.*

[6] STARCK, J. L.; MURTAGH, F.; FADILI, J. M.: *Sparse image and signal processing: wavelets, curvelets, morphological diversity. Cambridge: Cambridge University Press, 2010, 316 s. ISBN 978-0-521-11913-9.*

[7] ŠPIŘÍK, J.; RAJMIC, P.; VESELÝ, V.: Reprezentace signálů: od bází k framům. *Elektrorevue – Internetový časopis, 2011, roč. 2011, č. 111, s. 1-10. ISSN: 1213- 1539. URL http://elektrorevue.cz/cz/download/reprezentace-signalu–od-bazi-k-framum/*

[8] *Wikipedie – internetová encyklopedie*: Diskrétní kosinová transformace [online]. 2012 [cit. 2012-05-18]. Dostupné z: http://cs.wikipedia.org/wiki/DCT

[9] *Wikipedie – internetová encyklopedie*: Kronecker product [online]. 2012 [cit. 2012-05-17]. Dostupné z: http://en.wikipedia.org/wiki/Kronecker_product

A. Seznam použitých zkratek a symbolů

A.1. Použité symboly

$\mathbf{A}$	slovník - matice ze soustavy systému lineárních rovnic
$\mathbf{x}$	neznámé řídké koeficienty, které jsou vektorizované
$\mathbf{y}$	pravá strana systému lineárních rovnic
$\mathbf{Y}$	obraz (v kap. 2), původní obraz (v kap. 3.2), matice vektorů z trénovací databáze (v kap. 3.2)
$\mathbf{I}$	jednotková matice
k	řídkost vektoru
m, n	rozměry slovníku $\mathbf{A}$
$\mathbb{R}^{(m \times n)}$	($m \times n$ rozměrná) množina reálných čísel
$\otimes$	operace Kroneckerova součinu
$\|\mathbf{x}\|_p$	p norma vektoru $\mathbf{x}$
ϵ	povolená odchylka od přesného řešení
$\mu(\mathbf{A})$	vzájemná koherence
$spark(\mathbf{A})$	spark matice $\mathbf{A}$
$ker\mathbf{A}$	jádro lineárního zobrazení určeného maticí $\mathbf{A}$
$\mathbf{a}_j$	j-tý sloupec (atom) matice $\mathbf{A}$
$\mathbf{X}$	transformovaný signál (v kap. 3.2), pomocná matice v metodě IRLS (v kap. 5), matice, jejíž sloupce jsou vektory příslušné vektorům z trénovací databáze (v kap. 3.2)
$\mathbf{D}$	matice DCT báze
$\epsilon(j)$	chyba (při výpočtu hladových algoritmů)
$z_j \quad (z_j^*)$	konstanta (optimální) úměrnosti $\mathbf{r}$ k některému sloupci matice $\mathbf{A}$
$\mathbf{r}$	reziduum (rozdíl pravé a levé strany soustavy lineárních rovnic)
S	nosič vektoru $\mathbf{x}$
$\mathbf{A}^T$	transpozice matice $\mathbf{A}$
$\mathbf{A}^+$	pseudoinverze matice $\mathbf{A}$

γ	konstanta nulového prostoru
$\sigma_k(\mathbf{x})_p$	chyba nejlepší aproximace vektoru $\mathbf{x}$ k-řídkým vektorem
δ_k	konstanta zeslabené izometrie

A.2. Použité zkratky

OMP	ortogonální sdružovací metoda
MP	sdružovací metoda
IRLS	iterativně převažovaná metoda nejmenších čtverců
MOD	metoda optimálních směrů

B. Obsah CD

B.1. Hlavní programy

test	· hledání řídké reprezentace pro simulované obrazy ve zvolené bázi i simulace měření (pomocí *MP, OMP a IRLS*)
uspesnost_rekonstrukce_obrazu	· vykreslí závislost úspěšnosti rekonstrukce obrazu na řídkosti a poměru stran měřicích matic (při simulaci měření)
rekonstrukce_zasumeneho_signalu	· vykreslí závislost relativní chyby a úspěšnosti rekonstrukce obrazu na směrodatné odchylce přidaného bílého šumu pro M*P, OMP* a *IRLS*
rekonstrukce_obrazu	· hledání řídké reprezentace ve zvolené bázi pro reálné obrazy (pomocí *MP, OMP a IRLS*), možnost oříznutí koeficientů báze na zadanou řídkost či přidání šumu k obrazu
hledani_optimalniho_lambda	· hledání optimálního parametru pro modifikovanou metodu *IRLSe* – vykreslí závislost optimálního parametru (λ) na směrodatné odchylce přidaného bílého šumu
K_SVD	· hledání optimálního slovníku pro simulované jednorozměrné signály

B.2. Pomocné funkce

dtwtmatrix	· vytvoří bázovou matici konkrétní vlnkové transformace (autoři: P. Rajmic a Z. Průša)
generovani_ridkeho_signalu	· vygeneruje řídký obraz v zadané bázi
IRLS	· iterativně převažovaná metoda nejmenších čtverců
IRLSe	· modifikovaná metoda iterativně převažovaných nejmenších čtverců
MP	· sdružovací metoda
normovani_slovniku	· provede normování slovníku (autor: P. Rajmic, upraveno)
OMP	· ortogonální sdružovací metoda
prolozeni_krivkou	· proloží body v rovině regresní kubickou parabolou

psnr	· špičkový poměr signálu k šumu – vyjadřuje poměr mezi maximální možnou energií obrazu (255) a energií šumu v logaritmickém měřítku
vykresleni_DCT2	· pomocný program, který vykreslí dvourozměrnou bázi DCT zadaných rozměrů
vytvoreni_baze	· vytvoří zadanou bázovou matici zadaných rozměrů
w_filters	· spočítá určité parametry nutné pro vytvoření matice zadané vlnkové transformace (autoři: P. Rajmic a Z. Průša)

B.3. Použité obrazy

kreslo

lena-orig

Terezka

C. Vybrané definice

Definice C.1. [4] Nechť $\mathbf{x} \in \mathbb{R}^n$, $\mathbf{x} = (x_1, ..., x_n)^T$. Pak l_p *norma vektoru* $\mathbf{x}$ je definována takto:

$$\|\mathbf{x}\|_p = \left(\sum_{i=1}^{n} |x_i|^p \right)^{1/p} \quad \text{pro} \quad 1 \leq p < \infty$$

$$\|\mathbf{x}\|_p = \sum_{i=1}^{n} |x_i|^p \quad \text{pro} \quad 0 < p < 1$$

$$\|\mathbf{x}\|_\infty = \max_i |x_i|$$

$$\|\mathbf{x}\|_0 = |supp\,(\mathbf{x})|\,.$$

Poznámka. O normu v pravém slova smyslu se jedná pouze v případě $1 \leq p \leq \infty$ (protože ostatní normy nejsou konvexní). Výrazem $supp(\mathbf{x})$ je myšlen nosič vektoru $\mathbf{x}$.

Definice C.2. [9] Nechť $\mathbf{A}$ je matice typu $m \times n$ a $\mathbf{B}$ je matice typu $p \times q$. Pak *Kroneckerův součin* $\mathbf{A} \otimes \mathbf{B}$ je bloková matice typu $mp \times nq$:

$$\mathbf{A} \otimes \mathbf{B} = \begin{pmatrix} a_{11}\mathbf{B} & \cdots & a_{1n}\mathbf{B} \\ \vdots & \ddots & \vdots \\ a_{m1}\mathbf{B} & \cdots & a_{mn}\mathbf{B} \end{pmatrix},$$

kde a_{ij} značí prvek matice $\mathbf{A}$ na pozici $[i, j]$.

Definice C.3. [4] Matice *pseudoinverzní* k $\mathbf{A}$ je $\mathbf{A}^+$, pro kterou platí:

$$\mathbf{A}\mathbf{A}^+\mathbf{A} = \mathbf{A}$$

$$\mathbf{A}^+\mathbf{A}\mathbf{A}^+ = \mathbf{A}^+$$

$$(\mathbf{A}\mathbf{A}^+)^T = \mathbf{A}\mathbf{A}^+$$

$$(\mathbf{A}^+\mathbf{A})^T = \mathbf{A}^+\mathbf{A}.$$

Také lze ukázat, že $\mathbf{A}^+ = \mathbf{A}^T(\mathbf{A}\mathbf{A}^T)^+$, což při plné řádkové hodnosti A přechází na $\mathbf{A}^+ = \mathbf{A}^T(\mathbf{A}\mathbf{A}^T)^{-1}$.