VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV TELEKOMUNIKACÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF TELECOMMUNICATIONS

COMPRESSIVE SAMPLING A SIMULACE ONE-PIXEL CAMERA

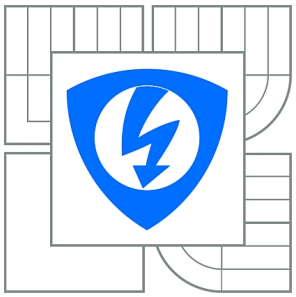
BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE RADEK HRBÁČEK
AUTHOR

BRNO 2011

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

ÚSTAV TELEKOMUNIKACÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

DEPARTMENT OF TELECOMMUNICATIONS

COMPRESSIVE SAMPLING A SIMULACE ONE-PIXEL CAMERA

COMPRESSIVE SAMPLING AND SIMULATION OF ONE-PIXEL CAMERA

BAKALÁŘSKÁ PRÁCE

BACHELOR'S THESIS

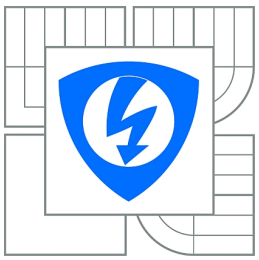
AUTOR PRÁCE — RADEK HRBÁČEK

AUTHOR

VEDOUCÍ PRÁCE — Mgr. PAVEL RAJMIC, Ph.D.

SUPERVISOR

BRNO 2011



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav telekomunikací

Bakalářská práce

bakalářský studijní obor
Teleinformatika

Student: Radek Hrbáček ***ID:*** 119440
Ročník: 3 ***Akademický rok:*** 2010/2011

NÁZEV TÉMATU:

Compressive sampling a simulace one-pixel camera

POKYNY PRO VYPRACOVÁNÍ:

Prostudujte momentální stav vědění v oblasti tzv. compressive sampling, která spočívá v úsporném procesu již během pořizování lineárních vzorků signálu, přičemž rekonstrukce spočívá v náročných nelineárních výpočetních postupech (sparse recovery). Proveďte simulace v MATLABu, které prokáží nebo vyvrátí funkčnost metody při na signálech jednak umělých a jednak reálných. Nasimulujte průběh snímání fotografií pomocí tzv. one-pixel camera a postup jejich rekonstrukce pomocí minimalizace totální variace.

DOPORUČENÁ LITERATURA:

[1] BRUCKSTEIN, A.M., DONOHO, D.L., ELAD, M. From Sparse Solutions of Systems of Equations to Sparse Modeling of Signals and Images. SIAM Review, Vol. 51, No. 1, pp. 34-81. 2009

[2] CHEN, S.S., DONOHO, D.L., SAUNDERS, M.A. Atomic Decomposition by Basis Pursuit. SIAM Rewiev, Vo. 43, No. 1, pp. 129-159. 2001.

[3] FORNASIER, M., RAUHUT, H. Compressive Sensing. Kapitola v knize Handbook of Mathematical Methods in Imaging, Springer, 2011. ISBN 978-0-387-92920-0
<http://www.ricam.oeaw.ac.at/people/page/fornasier/CSFornasierRauhut.pdf>

Termín zadání: 7.2.2011 ***Termín odevzdání:*** 2.6.2011

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

prof. Ing. Kamil Vrba, CSc.
Předseda oborové rady

ABSTRAKT

V klasickém pojetí zpracování číslicových signálů je základním pilířem Nyquistův teorém, podle něhož je možné spojitý signál rekonstruovat z jeho vzorků tehdy, pokud byl vzorkován s frekvencí alespoň dvakrát vyšší, než je nejvyšší frekvence signálu. Kvůli úspoře dat však v praxi signál ihned po jeho navzorkování komprimujeme. Compressive sampling se neomezuje pouze na frekvenční oblast, umožňuje signál vnímat v libovolné bázi. Jestliže najdeme takovou bázi, ve které je signál řídký, můžeme provést poměrně malý počet měření, ze kterých jsme schopni signál zrekonstruovat. One-pixel camera je jednou z praktických aplikací, tvoří ji pole zrcátek, které odrážejí světlo do jediného senzoru. Pomocí matematických metod je pak možné původní signál zrekonstruovat. Tato práce se zabývá simulací této kamery.

KLÍČOVÁ SLOVA

zpracování signálů a obrazů, Nyquistův teorém, compressive sampling, one-pixel camera, lineární programování, ℓ_1-minimalizace

ABSTRACT

The Nyquist theorem is the main pillar of the traditional digital signal processing approach. It states that the sampling rate must be at least twice the maximum frequency present in the signal to guarantee perfect signal reconstruction from the sequence of its samples. In practice, we often compress the signal right after the sampling process to reduce the data size. The compressive sampling approach is not limited to the frequency domain, it provides a new look at the signal by using an arbitrary basis. If we find a basis in which the signal is sparse, it is possible to take a small number of samples and reconstruct the signal successfully. One-pixel camera is one of real applications, it's formed by digital micromirror array reflexing the light into single sensor. Mathematical methods are then used to reconstruct the signal. This thesis deals with the simulation of the camera.

KEYWORDS

signal and image processing, Nyquist theorem, compressive sampling, one-pixel camera, linear programming, ℓ_1-optimization

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

PROHLÁŠENÍ

Prohlašuji, že svou bakalářskou práci na téma „Compressive sampling a simulace one-pixel camera“ jsem vypracoval samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené bakalářské práce dále prohlašuji, že v souvislosti s vytvořením této bakalářské práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení § 152 trestního zákona č. 140/1961 Sb.

Brno ....

(podpis autora)

PODĚKOVÁNÍ

Děkuji vedoucímu bakalářské práce Mgr. Pavlu Rajmicovi, Ph.D. a doc. RNDr. Vítězslavu Veselému, CSc. za odbornou pomoc a cenné rady při psaní práce.

OBSAH

SEZNAM OBRÁZKŮ

1 ÚVOD

Až donedávna bylo v oblasti zpracování signálů považováno za samozřejmé, že je nutné signál vzorkovat s frekvencí nejméně dvakrát vyšší, než je nejvyšší frekvence v signálu obsažená. Toto pravidlo známe pod názvem Nyquistův (nebo také Shannon-Kotělnikovův) teorém. Takto navzorkovaný signál ale klade vysoké nároky na kapacitu paměti a proto je často ihned komprimován. Při ztrátové komprimaci se obvykle vychází z vlastností lidských smyslů (zraku a sluchu) a ze signálu je odstraněna část informace tak, abychom nepoznali rozdíl mezi komprimovaným a originálním signálem. Například u formátu JPEG se provádí nad částmi obrázku diskrétní kosinová transformace [10] (resp. vlnková u JPEG 2000 [11]) a získané koeficienty se kvantují nebo vůbec neuchovávají. Formát MP3 pak na základě psychoakustického modelu za využití principů časového a frekvenčního maskování odebírá informace, které člověk buď vůbec neslyší, nebo je nevnímá.

Tento model je sice funkční, ale plýtvá někdy poměrně drahými prostředky. Výrazně odlišný přístup je zvolen v případě *compressive sampling* (komprimující vzorkování). Princip spočívá v rozšíření pohledu na signál o další oblasti, neomezuje se tedy pouze na časovou a frekvenční oblast. Komprimace signálu je již integrována v samotném snímání. Tím jsou ušetřeny prostředky při získávání signálu, ale na druhou stranu je potřeba využít matematických metod k jeho rekonstrukci.

Jednou z mnoha aplikací pak je *one-pixel camera*. Jak název napovídá, je tvořena pouze jediným snímačem (pixelem), na který dopadá světlo přes pole pohyblivých zrcátek. Konfigurace tohoto pole se s časem mění a tím je získáno více vzorků. Počet těchto vzorků je pak výrazně nižší, než je rozlišení pole zrcátek a dojde tak k redukci informace již při snímání. Při rekonstrukci se využívá poznatku, že reálné signály jsou řídké v nějaké bázi.

1.1 Definice základních pojmů

Pro účely této práce se omezíme pouze na vektorové (nebo též lineární) prostory konečné dimenze.

Definice 1.1 [12]**.** *Vektorový prostor nad tělesem $\mathcal{F}$ je množina V společně s operacemi:*

- *sčítání vektorů: zobrazení $V \times V \to V$, značeno $\mathbf{x} + \mathbf{y}$, kde $\mathbf{x}, \mathbf{y} \in V$*
- *násobení skalárem: zobrazení $\mathcal{F} \times V \to V$, značeno $\alpha\mathbf{x}$, kde $\alpha \in \mathcal{F}$, $\mathbf{x} \in V$*

splňující axiomy:

1. $\mathbf{x} + \mathbf{y} = \mathbf{y} + \mathbf{x}$ *pro každé dva prvky* $\mathbf{x}, \mathbf{y} \in V$
2. $\mathbf{x} + (\mathbf{y} + \mathbf{z}) = (\mathbf{x} + \mathbf{y}) + \mathbf{z}$ *pro každou trojici prvků* $\mathbf{x}, \mathbf{y}, \mathbf{z} \in V$

3. *v V existuje prvek $\mathbf{0}$ takový, že pro všechna $\mathbf{x} \in V$ platí $\mathbf{x} + \mathbf{0} = \mathbf{x}$, prvek $\mathbf{0}$ nazýváme nulovým prvkem množiny V*
4. *ke každému $\mathbf{x} \in V$ existuje prvek $-\mathbf{x} \in V$ takový, že $\mathbf{x} + (-\mathbf{x}) = \mathbf{0}$*
5. *$\alpha(\beta\mathbf{x}) = (\alpha\beta)\mathbf{x}$ pro každou trojici $\mathbf{x} \in V$ a $\alpha, \beta \in \mathcal{F}$*
6. *$(\alpha + \beta)\mathbf{x} = \alpha\mathbf{x} + \beta\mathbf{x}$ pro každou trojici $\mathbf{x} \in V$ a $\alpha, \beta \in \mathcal{F}$*
7. *$\alpha(\mathbf{x} + \mathbf{y}) = \alpha\mathbf{x} + \alpha\mathbf{y}$ pro každou trojici $\mathbf{x}, \mathbf{y} \in V$ a $\alpha \in \mathcal{F}$*
8. *$1\mathbf{x} = \mathbf{x}$, kde $\mathbf{x} \in \mathcal{F}$ je jednotkový prvek tělesa $\mathcal{F}$.*

Definice 1.2. *Dimenzí vektorového prostoru nazýváme počet prvků libovolné báze tohoto prostoru.*

Tělesem bývá obvykle množina reálných ($\mathbb{R}$) nebo komplexních ($\mathbb{C}$) čísel, pak podle dimenze prostoru značíme vektorové prostory $\mathbb{R}^2, \mathbb{R}^3, \ldots, \mathbb{R}^n$ resp. $\mathbb{C}^2, \mathbb{C}^3, \ldots, \mathbb{C}^n$, kde $n \in \mathbb{N}$. Prvky vektorového prostoru se nazývají vektory.

Definice 1.3. *ℓ_p-norma vektoru $\mathbf{x} \in \mathbb{C}^N$ je definována jako*

$$\begin{aligned} \|\mathbf{x}\|_p &:= \left(\sum_{i=1}^{N} |x_i|^p \right)^{1/p}, \qquad 0 < p < \infty, \\ \|\mathbf{x}\|_\infty &:= \max_{i=1,\ldots,N} |x_i|, \\ \|\mathbf{x}\|_0 &:= |\mathrm{supp}(\mathbf{x})|. \end{aligned} \tag{1.1}$$

O normu se ale jedná pouze v případě $1 \leq p \leq \infty$. Pro $0 < p < 1$ neplatí trojúhelníková nerovnost, takže se jedná o *kvazinormu*, a ℓ_0 není pozitivně homogenní, tudíž se také ve skutečnosti o normu nejedná. Pro zjednodušení budu v této práci tuto skutečnost zanedbávat a pro všechna $0 \leq p \leq \infty$ budu používat označení ℓ_p-norma.

Definice 1.4. *Součet čtverců všech prvků matice $\mathbf{A}$ nazveme normou matice $\mathbf{A}$*

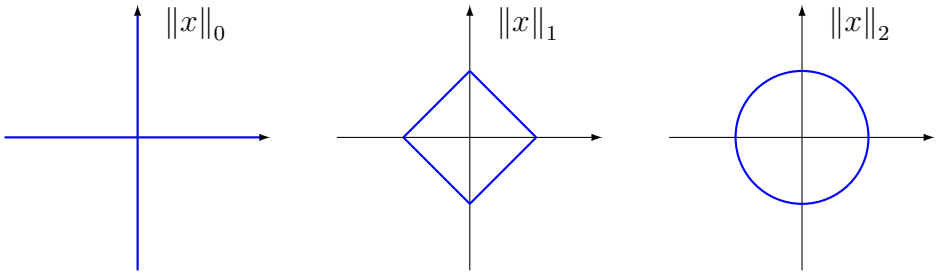
$$\|\mathbf{A}\| := \sqrt{\sum_i \sum_j a_{ij}^2} \tag{1.2}$$

Abychom si dokázali normy lépe představit, zobrazujeme jednotkové koule v jednotlivých normách.

Definice 1.5. *Jednotková koule B_p^N v normě ℓ_p je definována jako*

$$B_p^N := \left\{ \mathbf{x} \in \mathbb{C}^N, \|\mathbf{x}\|_p \leq 1 \right\}. \tag{1.3}$$

Nás budou zajímat především normy ℓ_0 (počet nenulových prvků), ℓ_1 a ℓ_2 (energie signálu). Na obrázku 1.1 je ilustrace jednotkových koulí v těchto normách. Jak je vidět z ilustrace, jednotková koule v normě ℓ_0 kopíruje osy souřadného systému kromě počátku.



Obr. 1.1: Ilustrace jednotkových koulí B_0^2, B_1^2 a B_2^2 v různých normách

Definice 1.6. *Systém* $\{\mathbf{v}_i\}_{i\in\{1,\dots,n\}} \subseteq V$ *se nazývá (Schauderova) báze* n*-rozměrného vektorového prostoru* V*, pokud platí*

$$\forall \mathbf{x} \in V : \exists! \, \{\alpha_i\}_{i\in\{1,\dots,n\}} \text{ tak, že platí } \mathbf{x} = \sum_{i=1}^{n} \alpha_i \mathbf{v}_i. \tag{1.4}$$

Báze vektorového prostoru V je tedy množina lineárně nezávislých vektorů, pro které platí, že uzávěr jejich lineárního obalu je celým prostorem V. U prostorů konečné dimenze n je bází každá množina n lineárně nezávislých vektorů. V případě prostorů nekonečné dimenze je situace složitější a je třeba uvažovat uzávěr lineárního obalu. V této práci se setkáme pouze s prostory konečné dimenze.

Z praktického hlediska mají největší význam báze ortogonální a ortonormální.

Definice 1.7. *Bázi nazveme ortogonální, jestliže pro libovolné dva různé vektory* $\mathbf{v}_i, \mathbf{v}_j$ *báze platí*

$$(\mathbf{v}_i, \mathbf{v}_j) = 0 \tag{1.5}$$

Definice 1.8. *Bázi nazveme ortonormální, jestliže je ortogonální a navíc pro každý vektor* $\mathbf{v}_i$ *báze platí*

$$(\mathbf{v}_i, \mathbf{v}_i) = 1 \tag{1.6}$$

Při rekonstrukci signálů nás velmi zajímá numerická stabilita. Pokud řešíme soustavu lineárních rovnic, vyžadujeme, aby byla dobře podmíněná.

Definice 1.9. *Nechť* $\mathbf{Ax} = \mathbf{b}$ *je systém lineárních rovnic. Podmíněnost soustavy definujeme jako*

$$\kappa(\mathbf{A}) := \left\|\mathbf{A}\right\| \cdot \left\|\mathbf{A}^{-1}\right\| \tag{1.7}$$

Pokud je $\kappa(\mathbf{A}) \gg 1$, soustava je špatně podmíněná a malé odchylky ve vstupních datech nebo při numerickém výpočtu způsobí velké odchylky v řešení.

Nyní již máme definovány základní pojmy, se kterými se budeme v práci setkávat. Zbývá jen doplnit několik označení [8]:

- Pro $T \subset \{1, \ldots, N\}$ označíme $\mathbf{x}_T \in \mathbb{C}^N$ vektor odvozený z $\mathbf{x} \in \mathbb{C}^N$ tak, že prvky na pozicích patřících do T zachováme a ostatní vynulujeme.
- Komplement T označíme $T^c = \{1, \ldots, N\} \setminus T$.
- Kardinalitu množiny T, tedy počet prvků, označíme $|T|$.
- Jádro lineárního zobrazení definovaného maticí $\mathbf{A}$ budeme značit $\ker A$.

1.2 Řídkost

Řídkost vektoru je klíčovým předpokladem v oblasti compressive sampling. Z pozorování totiž vyplývá, že mnohé reálné signály často mají řídkou reprezentaci v určité bázi (např. vlnkové transformaci), což znamená, že se dají velmi dobře (jen s malou chybou) aproximovat vektorem, který má jen několik nenulových prvků [8].

Definice 1.10. *Vektor* $\mathbf{x}$ *nazveme* k*-řídkým (*k*-sparse), pokud platí*

$$\|\mathbf{x}\|_0 \leq k \tag{1.8}$$

Relativní řídkostí vektoru $\mathbf{x}$ pak budeme rozumět poměr $\frac{k}{N} \in \mathbb{Q}$, kde $k \in \mathbb{Z}_0^+$ je řídkost vektoru $\mathbf{x}$ a $N \in \mathbb{N}$ je jeho délka.

Dále označíme $\Sigma_k := \left\{\mathbf{x} \in \mathbb{C}^N : \|\mathbf{x}\|_0 \leq k\right\}$ množinu všech k-řídkých vektorů. Reálné signály ovšem nebudou striktně řídké, tak jak jsme si řídký vektor definovali výše, ale budou obsahovat malé nenulové hodnoty. Proto je vhodné definovat chybu aproximace:

Definice 1.11. *Chyba nejlepší aproximace* k*-řídkého vektoru* $\mathbf{x} \in \mathbb{C}^N$ *v normě* ℓ_p *je definována jako*

$$\sigma_k(\mathbf{x})_p := \inf_{\mathbf{z} \in \Sigma_k} \|\mathbf{x} - \mathbf{z}\|_p . \tag{1.9}$$

1.3 Formulace problému

Nyní je možné formálně popsat problém, kterým se budeme zabývat. Předpokládejme, že vektor $\mathbf{x} \in \mathbb{C}^N$ je k-řídký, vektor naměřených hodnot označme $\mathbf{y} \in \mathbb{C}^m$. Matici $\mathbf{A} \in \mathbb{C}^{m \times N}$ nazveme měřicí (*measurement matrix*), přičemž zajímá nás především případ, kdy $m \ll N$ (výrazné podvzorkování). V následujícím textu se budeme zabývat optimalizační úlohou

$$\min \|\mathbf{x}\|_0 \quad \text{vzhledem k} \quad \mathbf{A}\mathbf{x} = \mathbf{y} \tag{P0}$$

a také požadavky kladenými na $\mathbf{A}$, které zaručí možnost rekonstrukce $\mathbf{x}$ z $\mathbf{y}$.

1.4 Lineární programování

Lineární programování je jednou z optimalizačních úloh. Jejím principem je nalezení minima (resp. maxima v duální úloze) účelové funkce na dané množině přípustných řešení. Formálně se tedy jedná o úlohu

$$\min_{\mathbf{x}\in M} \mathbf{c}^T\mathbf{x}, \tag{1.10}$$

kde $\mathbf{x} \in \mathbb{R}^n$ je proměnný vektor, $\mathbf{c} \in \mathbb{R}^n$ jsou koeficienty účelové funkce, M je množina přípustných řešení a je popsána soustavou lineárních rovnic

$$\mathbf{A}\mathbf{x} = \mathbf{b}, \quad \mathbf{x} \geq 0,$$

kde $\mathbf{A} \in \mathbb{R}^{m\times n}$.

Výhodou lineárního programování je především fakt, že na něj lze převést řadu reálných problémů a také existence velmi efektivních algoritmů k jeho řešení – nejznámější metodou je tzv. *simplexový algoritmus*, ale existují i rychlejší algoritmy, např. *elipsoidová metoda* nebo metoda *vnitřních bodů*. Jak uvidíme dále, lze ho využít i v compressive sampling [3].

2 ŘEŠENÍ STUDENTSKÉ PRÁCE

V minulé kapitole jsme si formálně definovali rekonstrukční úlohu. Slovně by se dala popsat jako hledání nejřidšího řešení soustavy $\mathbf{y} = \mathbf{A}\mathbf{x}$. Násobení vektoru $\mathbf{x} \in \mathbb{C}^N$ maticí $\mathbf{A} \in \mathbb{C}^{m \times N}$ představuje sejmutí m „vzorků“ signálu $\mathbf{x}$. Přitom nás zajímá případ $m \ll N$, kdy je signál silně podvzorkován. Soustava obsahuje více proměnných než rovnic, možných řešení je tedy nekonečně mnoho. Pokud ale víme, že vektor $\mathbf{x}$ je k-řídký, je možné za určitých podmínek rekonstrukci provést.

2.1 Od ℓ_0- k ℓ_1-minimalizaci

Vzhledem k vlastnostem ℓ_0-normy je základní problém (P0) NP-těžký a tedy neřešitelný pro úlohy většího rozsahu. Lze ale ukázat, že pro vhodně zvolené měřicí matice $\mathbf{A}$ se dá problém převést na

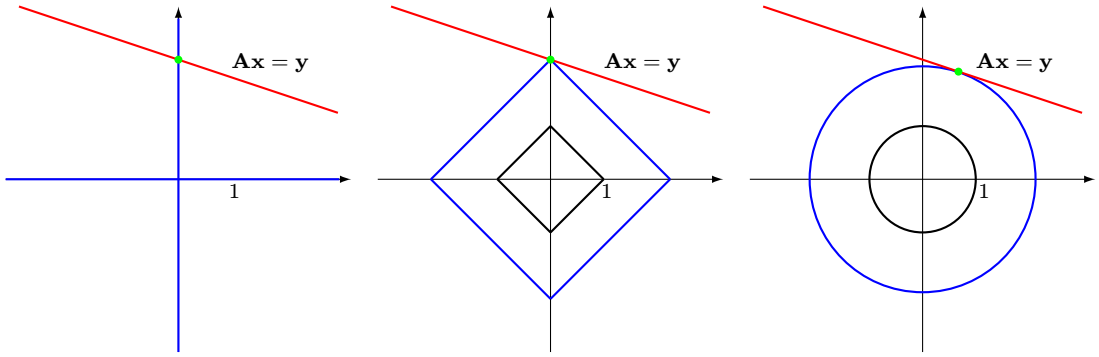
$$\min \|\mathbf{x}\|_1 \quad \text{vzhledem k} \quad \mathbf{A}\mathbf{x} = \mathbf{y} \tag{P1}$$

Tato úloha již je konvexní a řešitelná řadou efektivních algoritmů. Musíme ovšem zajistit, aby výsledek minimalizace podle úlohy (P1) dával stejné výsledky jako podle úlohy (P0) a to závisí na volbě měřicí matice $\mathbf{A}$. Na obrázku 2.1 je znázorněna minimalizace ve třech normách – zleva ℓ_0, ℓ_1 a ℓ_2. Je na něm vidět, že ℓ_0-minimalizace dává stejný „řídký“ výsledek jako ℓ_1-minimalizace, zatímco ℓ_2-minimalizace již řídké řešení nedává. Důvod je zřejmý – řídké řešení leží (v tomto případě dvourozměrného prostoru) na osách souřadného systému. Když si představíme ℓ_1-kouli, jak se nafukuje, zřejmě se nejprve dotkne nadroviny $\mathbf{A}\mathbf{x} = \mathbf{y}$[1] vrcholem (tedy v ose souřadného systému). Oproti tomu ℓ_2-koule, která odpovídá řešení s minimální energií, se dotkne mimo osu a řešení pak není řídké.

Ale i v případě ℓ_1-koule se může vyskytnout komplikace, když bude nadrovina $\mathbf{A}\mathbf{x} = \mathbf{y}$ rovnoběžná s hranou (resp. nadplochou) ℓ_1-koule. Potom bude řešení nekonečně mnoho a bude záležet na zvolené numerické metodě, ke kterému řešení se přikloní. Řešitelnost ℓ_1-minimalizace tedy není v rozporu s vysokou náročností úlohy ℓ_0-minimalizace. Shodnost výsledků je totiž zaručena jen pro omezenou podmnožinu matic $\mathbf{A}$ a vektorů y. Na obrázku 2.1 je také názorně vidět, proč ℓ_2 minimalizace dává řešení s nejnižší energií – vzdálenost řešení od počátku souřadného systému je minimální.

Mohlo by nás napadnout použít libovolnou normu ℓ_p, $0 < p < 1$, zřejmě by vždy bylo řešení řídké. Taková úloha by však vedla na nekonvexní optimalizaci,

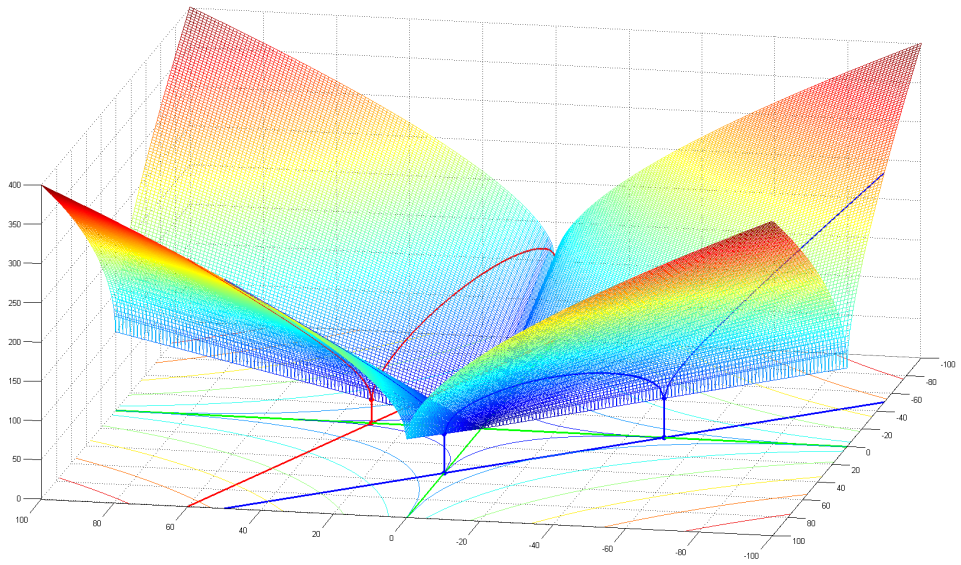
[1] Nadrovina $\mathbf{A}\mathbf{x} = \mathbf{y}$ je jinak také jádro lineárního zobrazení definovaného maticí $\mathbf{A}$ posunuté o libovolné $\mathbf{x} : \mathbf{A}\mathbf{x} = \mathbf{y}$



Obr. 2.1: Nafukující se koule v normách ℓ_0, ℓ_1 a ℓ_2 a jejich dotyk s nadrovinou $\mathbf{Ax} = \mathbf{y}$

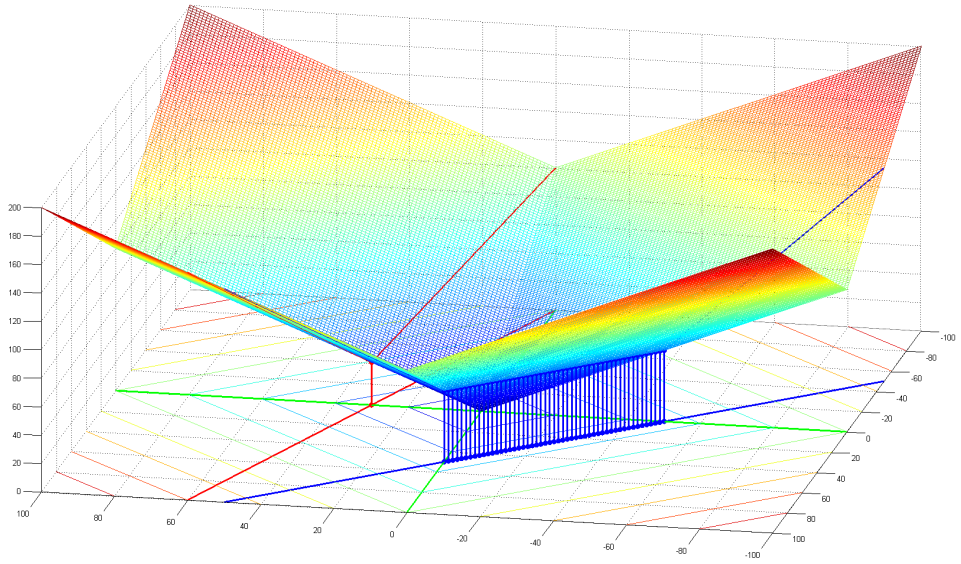
která nemusí vést ke globálnímu extrému [2]. Nalezené řešení by tedy nemuselo být nejřidší možné.

Na následujících 3 vizualizacích jsou vidět dvě stejné úlohy řešené postupně $\ell_{0,5}$-, ℓ_1- a ℓ_2-minimalizací. Normy jsou zde vykresleny jako plochy, množina všech řešení červenou resp. modrou čárou.



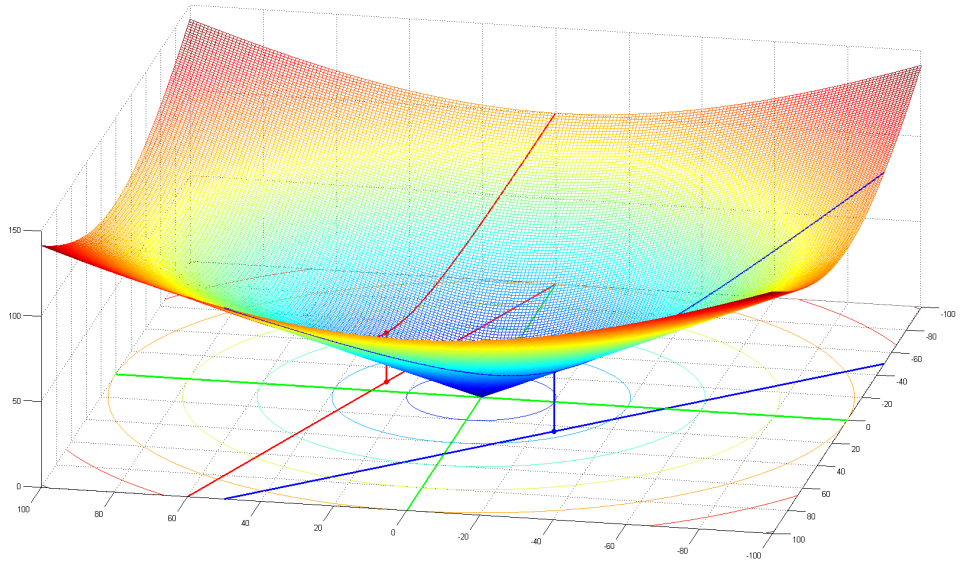
Obr. 2.2: Vizualizace normy $\ell_{0,5}$ a řešení dvou odlišných úloh

Obrázek 2.3 vysvětluje, proč ℓ_1-minimalizace může selhat v hledání řídkého řešení. Modrá čára je zde rovnoběžná s plochou tvořenou normou ℓ_1 a v množině všech řešení (modrá čára) tedy existuje nekonečně mnoho stejně kvalitních řešení.



Obr. 2.3: Vizualizace normy ℓ_1 a řešení dvou odlišných úloh

Poslední vizualizace 2.4 pak ilustruje normu ℓ_2 a stejné dvě úlohy. Nalezené řešení nebude téměř nikdy řídké, ale bude mít nejnižší možnou energii. Proto se ℓ_2-minimalizace nehodí pro rekonstrukci řídkých signálů.



Obr. 2.4: Vizualizace normy ℓ_2 a řešení dvou odlišných úloh

Nyní je třeba stanovit takové podmínky pro matici $\mathbf{A}$, aby byla zaručena ekvi-

valence řešení ℓ_0- a ℓ_1-minimalizace. Dále nás bude zajímat, jestli nalezené řešení je nejřidší možné.

2.1.1 Spark

Do češtiny těžko přeložitelný pojem *spark* je důležitou vlastností pro rozhodování, zda dané řešení je nejřidší.

Definice 2.1 [4]**.** *spark*$(\mathbf{A})$ *je nejmenší počet sloupců matice* $\mathbf{A}$*, které jsou lineárně závislé.*

Pro matici $\mathbf{A} \in \mathbb{C}^{m \times N}$, kde $m \ll N$, může *spark* nabývat hodnot $spark(\mathbf{A}) \in \{2, \ldots, m+1\}$. Čím menší *spark* je, tím řidší musí vektor $\mathbf{x}$ být, aby bylo možné zajistit jedinečnost tohoto řešení.

Věta 2.2 [2]**.** *Pokud má soustava* $\mathbf{Ax} = \mathbf{b}$ *řešení* $\mathbf{x}$ *splňující*

$$\|\mathbf{x}\|_0 < \frac{spark(\mathbf{A})}{2}, \tag{2.1}$$

pak $\mathbf{x}$ *je nutně nejřidší možné.*

Bohužel nalezení *spark*$(\mathbf{A})$ je srovnatelně výpočetně náročné jako řešení problému (P0), proto je nutné hledat jednodušší způsob ověření jedinečnosti řešení.

2.1.2 Vzájemná koherence

Definice 2.3 [2]**.** *Vzájemná koherence (mutual coherence) matice* $\mathbf{A}$ *je největší absolutní normalizovaný skalární součin dvou různých sloupců matice* $\mathbf{A}$*.*

$$\mu(\mathbf{A}) = \max_{1 \leq j,k \leq N,\, j \neq k} \frac{\left|\mathbf{a}_j^T \mathbf{a}_k\right|}{\|\mathbf{a}_j\| \cdot \|\mathbf{a}_k\|}, \tag{2.2}$$

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

Pomocí vzájemné koherence můžeme zjistit závislost mezi sloupci matice. Unitární matice bude mít vždy koherenci nulovou, nedourčená soustava bude mít naopak vždy nenulovou koherenci. Naším cílem je získat matici $\mathbf{A}$ s co nejnižší koherencí, aby se její chování alespoň přiblížilo chování unitární matice. Zároveň požadujeme, aby byla soustava co nejvíce nedourčená a prováděla tak komprimaci signálu.

Spočtení vzájemné koherence není příliš výpočetně náročné (oproti *spark*u je třeba testovat jen všechny dvojice sloupců, nikoliv všechny n-tice, kde $n \leq k$) a umožňuje nám zdola ohraničit *spark*, který výpočetně náročný je.

Věta 2.4 [2]. *Pro libovolnou matici* $\mathbf{A}$ *platí*

$$spark(\mathbf{A}) \geq 1 + \frac{1}{\mu(\mathbf{A})}. \tag{2.3}$$

Ohraničením $spark(\mathbf{A})$ dostaneme větu podobnou (2.2).

Věta 2.5 [2]. *Pokud má soustava* $\mathbf{Ax} = \mathbf{b}$ *řešení* $\mathbf{x}$ *splňující*

$$\|\mathbf{x}\|_0 < \frac{1}{2}\left(1 + \frac{1}{\mu(\mathbf{A})}\right), \tag{2.4}$$

pak $\mathbf{x}$ *je nutně nejřidší možné.*

2.1.3 Null Space Property

Další možností, jak ověřit rekonstrukční schopnosti matice, je *Null Space Property* (vlastnost nulového prostoru).

Definice 2.6 [8]. *Matice* $\mathbf{A} \in \mathbb{C}^{m\times N}$ *splňuje null space property (NSP) řádu* k *s konstantou* $\gamma \in (0,1)$*, pokud*

$$\|\eta_T\|_1 \leq \gamma \|\eta_{T^c}\|_1, \tag{2.5}$$

pro všechny množiny $T \subset \{1,\ldots,N\}$*,* $|T| \leq k$ *a pro všechny* $\eta \in \ker \mathbf{A}$*.*

Splnění NSP zajišťuje pro libovolný vektor z $\ker(\mathbf{A})$, že v něm nebude energie koncentrována v malém počtu prvků. Dále můžeme pomocí NSP omezit chybu aproximace řešení ℓ_1-minimalizací.

Věta 2.7 [8]. *Nechť matice* $\mathbf{A} \in \mathbb{C}^{m\times N}$ *splňuje NSP řádu* k *s konstantou* $\gamma \in (0,1)$*. Nechť* $\mathbf{x} \in \mathbb{C}^N$*,* $\mathbf{y} = \mathbf{Ax}$ *a* $\mathbf{x}^*$ *je řešení* ℓ_1*-minimalizace. Potom*

$$\|\mathbf{x} - \mathbf{x}^*\|_1 \leq \frac{2(1+\gamma)}{1-\gamma}\sigma_k(\mathbf{x}). \tag{2.6}$$

2.1.4 Restricted Isometry Property

Na první pohled je vidět, že ověření NSP není jednoduché. *Restricted Isometry Property* (RIP) oproti tomu nabízí výpočetně přijatelnější alternativu, která je navíc stabilní i pod vlivem šumu.

Definice 2.8 [8]. *Restricted isometry constant* δ_k *matice* $\mathbf{A} \in \mathbb{C}^{m\times N}$ *je nejmenší číslo takové, že*

$$(1-\delta_k) \leq \frac{\|\mathbf{Az}\|_2^2}{\|\mathbf{z}\|_2^2} \leq (1+\delta_k), \tag{2.7}$$

kde $\mathbf{z} \in \Sigma_k$ *lib.*

Matice splňuje RIP řádu k s konstantou δ_k, pokud $\delta \in (0,1)$. RIP je také možné spočítat přímo.

$$\delta_k = \max_{T \subset \{1,\dots,N\}, \|T\| \leq k} \|\mathbf{A}_T^* \mathbf{A}_T - \mathbf{I}\| \tag{2.8}$$

Pro matici splňující RIP platí, že libovolná podmatice o maximálně k sloupcích je dobře podmíněná. To je důležité zejména proto, že dopředu nevíme, které prvky vektoru $\mathbf{x}$ budou nenulové a tedy nevíme, které sloupce matice $\mathbf{A}$ budou při snímání obrazu $\mathbf{y}$ použity. Proto musíme zajistit, aby se libovolná kombinace nejvýše k sloupců co nejvíce blížila ortonormálnímu systému.

Podobně jako u NSP lze určit i u RIP chybu aproximace ℓ_1-minimalizací. Stejně tak lze nalézt vztah mezi RIP a NSP.

Věta 2.9 [8]**.** *$\mathbf{A} \in \mathbb{C}^{m \times N}$ splňuje RIP řádu $K = k + h$ s konstantou $\delta_K \in (0,1)$. Pak $\mathbf{A}$ splňuje NSP řádu k s konstantou*

$$\gamma = \sqrt{\frac{k}{h} \frac{1+\delta_K}{1-\delta_K}}$$

Věta 2.10 [8]**.** *$\mathbf{A} \in \mathbb{C}^{m \times N}$ splňuje RIP řádu $3k$ s konstantou $\delta_{3k} < \frac{1}{3}$. Pro $\mathbf{x} \in \mathbb{C}^N$, nechť $\mathbf{y} = \mathbf{A}\mathbf{x}$ a $\mathbf{x}^*$ je řešení ℓ_1-minimalizace. Pak*

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

Věta 2.11 [8]**.** *Předpokládejme, že matice $\mathbf{A} \in \mathbb{C}^{m \times N}$ vyhovuje RIP řádu $2k$ s konstantou*

$$\delta_{2k} < \frac{2}{3 + \sqrt{7/4}} \approx 0.4627.$$

Následující platí pro $\forall \mathbf{x} \in \mathbb{C}^N$. Nechť měření jsou zatížená šumem $\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{e}$, $\|\mathbf{e}\|_2 \leq \eta$, $\mathbf{x}^*$ je řešení

$$\min \|\mathbf{z}\|_1 \quad \textit{vzhledem k} \quad \|\mathbf{A}\mathbf{z} - \mathbf{y}\|_2 \leq \eta$$

Potom

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

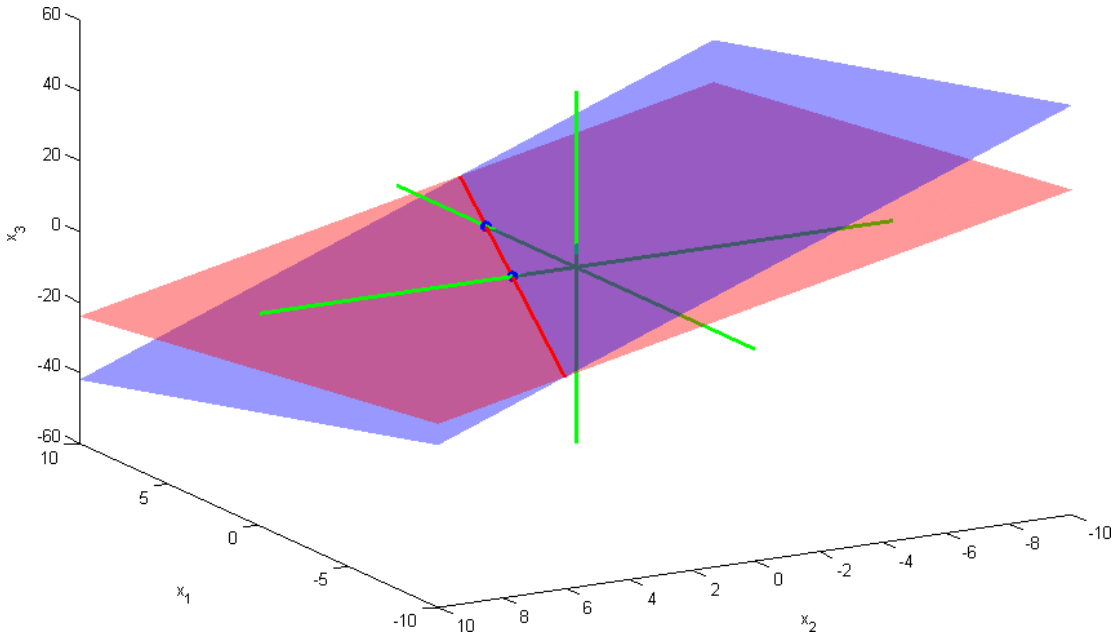
pro nějaké konstanty C_1, C_2, které jsou závislé pouze na δ_{2k}.

Uvedené vlastnosti měřicí matice jsou bohužel pouze teoretické hranice. Experimenty ukazují, že i matice, které tato kritéria vůbec nesplňují, jsou v praxi použitelné.

2.1.5 Příklady soustav rovnic

Na obrázku 2.5 je vizualizována soustava dvou lineárních rovnic (modrá a červená plocha):

$$\begin{aligned} \frac{4}{5}x_1 + 2x_2 + x_3 &= 4 \\ \frac{7}{5}x_1 + \frac{7}{2}x_2 + x_3 &= 7 \end{aligned} \tag{2.9}$$



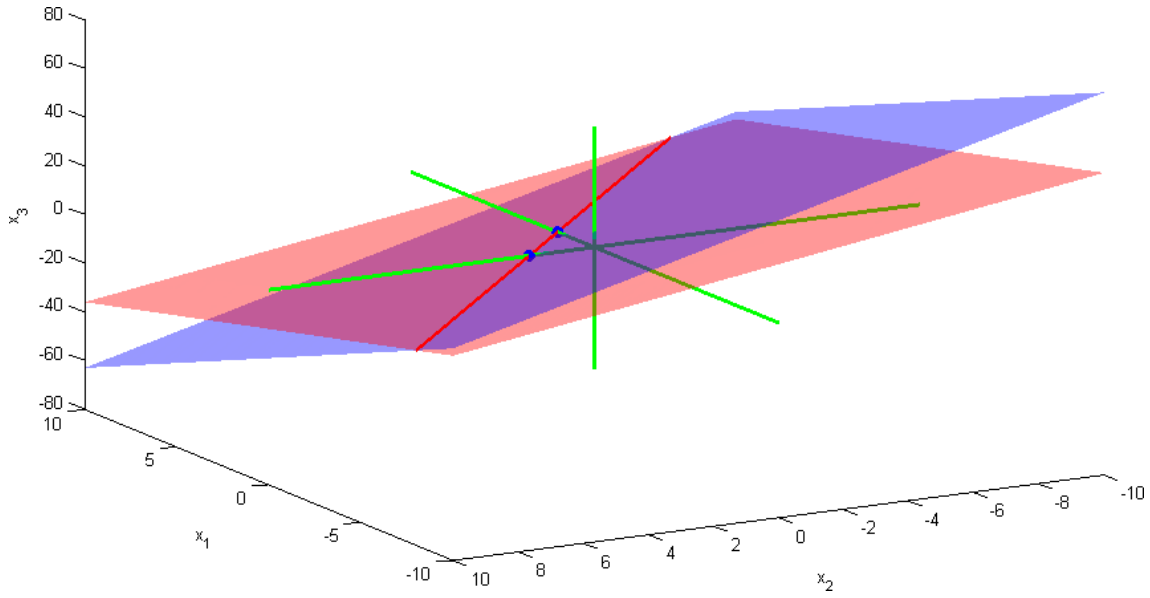
Obr. 2.5: Vizualizace soustavy dvou lineárních rovnic (2.9), množiny řešení a nalezených řídkých řešení.

Řešením této soustavy je přímka určená rovnicí $x_2 = 2 - \frac{2}{5}x_1$ (červená čára). Soustava má dvě řešení o řídkosti $k = 1$, která jsou vyznačena modrými body ($[0, 2, 0]$ a $[5, 0, 0]$).

Z ilustrace je patrné, že červená přímka (tedy množina řešení soustavy) není rovnoběžná se stranou ℓ_1-koule. Proto numerické metody založené na problému (P1) uspějí a naleznou řídké řešení. Mírnou úpravou koeficientů v soustavě získáme odlišnou soustavu zobrazenou na obrázku 2.7:

$$\begin{aligned} 2x_1 + 2x_2 + x_3 &= 4 \\ \frac{7}{2}x + \frac{7}{2}x_2 + x_3 &= 7 \end{aligned} \tag{2.10}$$

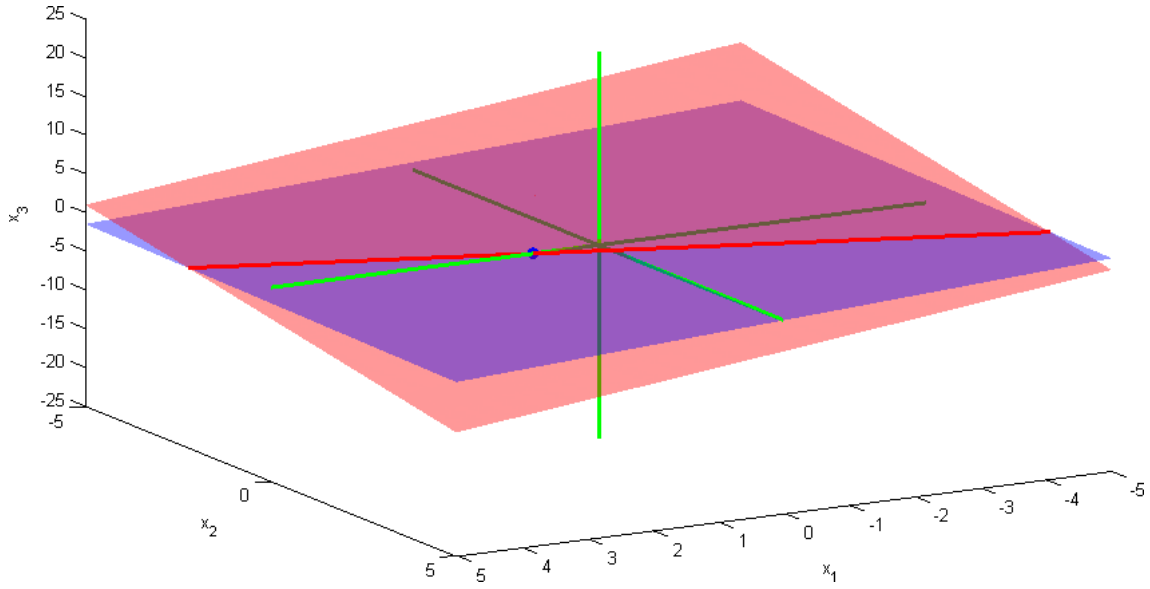
Tato soustava má opět nekonečně mnoho řešení (přímka $x_2 = 2 - x_1$), dvě z nich jsou 1-řídká ($[0, 2, 0]$ a $[2, 0, 0]$). V tomto případě však ℓ_1-minimalizací nemusíme získat jedno ze dvou 1-řídkých řešení.



Obr. 2.6: Vizualizace modifikované soustavy dvou lineárních rovnic (2.10), množiny řešení a nalezených řídkých řešení.

Poslední příklad soustavy rovnic 2.11 má pouze jediné 1-řídké řešení $[1, 0, 0]$.

$$\begin{aligned} x_1 + x_2 + x_3 &= 1 \\ \frac{1}{2}x + \frac{1}{10}x_2 + x_3 &= \frac{1}{1} \end{aligned} \tag{2.11}$$



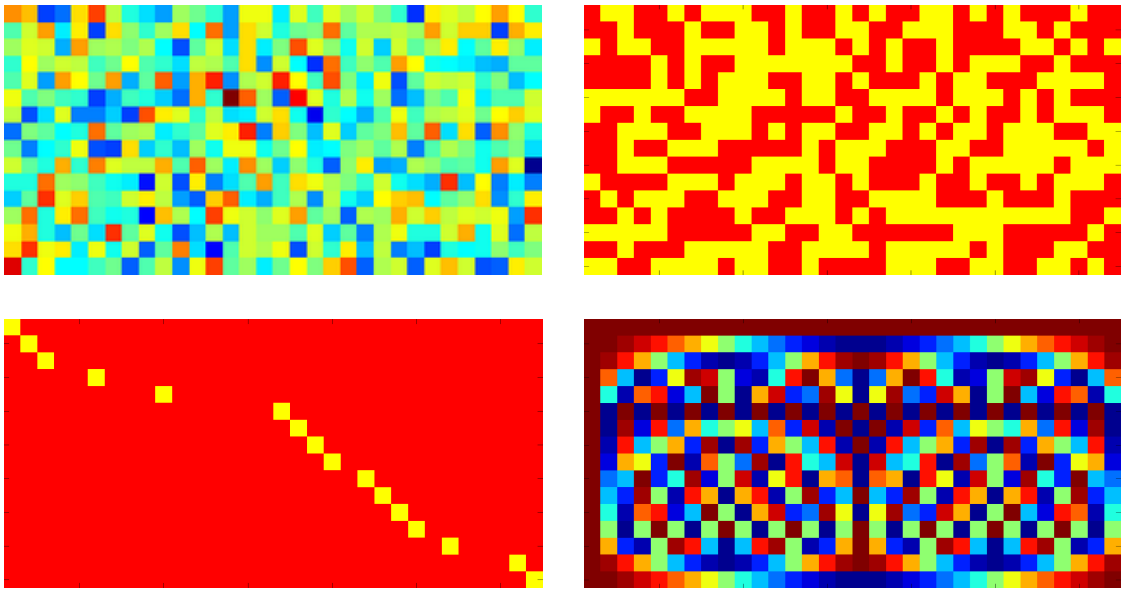
Obr. 2.7: Vizualizace soustavy dvou lineárních rovnic (2.11) s jediným 1-řídkým řešením, množiny řešení a nalezených řídkých řešení.

2.1.6 Konstrukce měřicí matice

Nyní již víme, jaké požadavky jsou kladeny na měřicí matici **A**, aby bylo možné z naměřených dat zrekonstruovat původní signál. Zbývá tedy takové matice nalézt a v praxi lze využít dvou odlišných přístupů, jak takové matice konstruovat:

- Složením několika ortogonálních bází, např. identity (tj. Diracovy báze v časové oblasti) a matice Fourierovy transformace ($\mathbf{A} = [\mathbf{I}\ \mathbf{F}]$)
- Použitím matic s náhodným rozdělením
    1. *Gaussovo rozdělení* – hodnoty v matici jsou generovány s normálním rozdělením se střední hodnotou 0 a rozptylem $1/m$.
    2. *Bernoulliho rozdělení* – zde jsou hodnoty rovny $\pm 1/\sqrt{m}$ se stejnou pravděpodobností
    3. *náhodné podvzorkování* – náhodně vybereme m z N vzorků
    4. *částečné náhodné Fourierovy matice* – náhodně vybereme některé řádky matice Fourierovy transformace

Vizualizace některých případů měřicích matic jsou vidět na obrázku 3.6. Matice s Bernoulliho rozdělením je využita v případě one-pixel camery, jak bude dále v textu uvedeno. Částečné náhodné Fourierovy matice (*Random Partial Fourier Matrix*) jsou tvořeny náhodným výběrem řádků z úplné matice Fourierovy transformace, rekonstrukční vlastnosti takové matice jsou zachovány díky invarianci vůči unitární transformaci. I z tohoto důvodu bude použití náhodných matic výhodné, jak uvidíme dále.



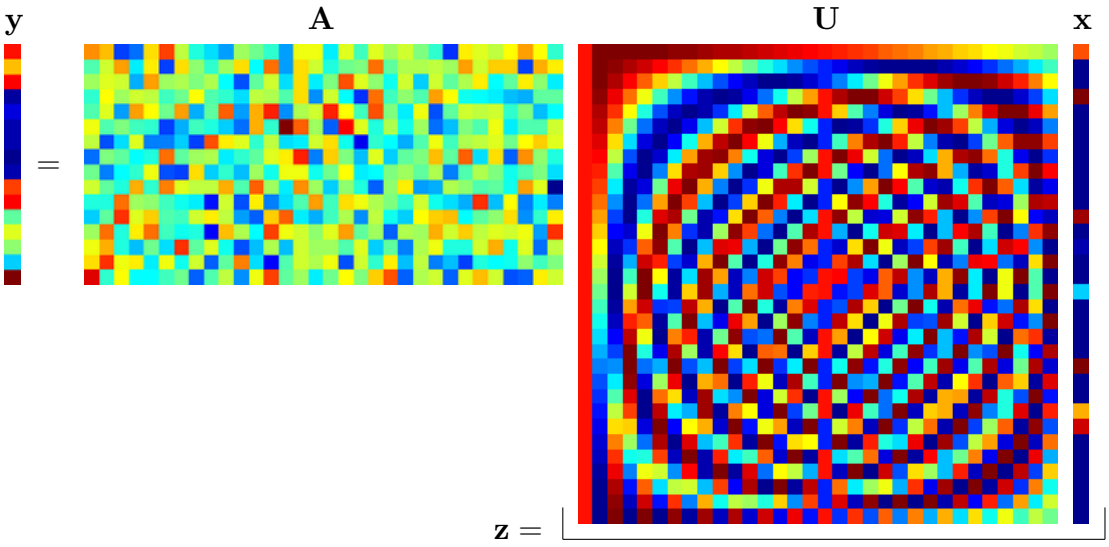
Obr. 2.8: Nahoře: Matice s Gaussovským a Bernoulliho rozdělením, dole: náhodné podvzorkování a Random Partial Fourier Matrix

2.1.7 Invariance vůči unitární transformaci

Lze ukázat, že rekonstrukční schopnost gaussovských nebo bernoulliovských matic je invariantní vůči unitární transformaci. To je velmi výhodné, protože signál, který se snažíme navzorkovat, je velmi často řídký v jiné bázi, než ve které ho snímáme. Snímaný signál, který *není* řídký, označme $\mathbf{z}$. Víme, že je řídký v ortogonální bázi $\mathbf{U}^{-1}$, tedy $\mathbf{z} = \mathbf{U}\mathbf{x}$, kde $\mathbf{x}$ je řídké (viz obrázek 2.10). Po navzorkování tedy máme $\mathbf{y} = \mathbf{A}\mathbf{z} = \mathbf{A}\mathbf{U}\mathbf{x}$ (viz obrázek 2.9). Potom můžeme problém (P1) upravit na

$$\min \|\mathbf{x}\|_1 \quad \text{vzhledem k} \quad \mathbf{A}\mathbf{U}\mathbf{x} = \mathbf{y} \tag{P1U}$$

a signál $\mathbf{z}$ poté zpětně dopočítat jako $\mathbf{z} = \mathbf{U}\mathbf{x}$. Důležité je, že matici $\mathbf{U}$ vůbec nemusíme znát při snímání, ale až při rekonstrukci.

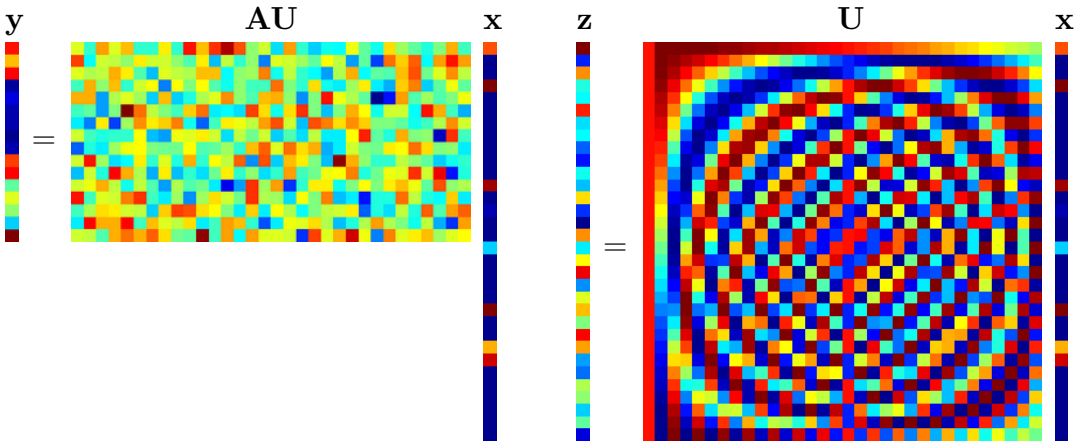


Obr. 2.9: Ilustrace procesu vzorkování: naměřený vektor $\mathbf{y}$ je roven součinu měřicí matice $\mathbf{A}$, unitární matice $\mathbf{U}$ a řídkého vektoru $\mathbf{x}$. Ve skutečnosti do procesu měření vstupuje vektor $\mathbf{z}$, který je pozorovatelný a není přímo řídký, ale je řídký v bázi $\mathbf{U}^{-1}$. Zde $\mathbf{U}$ je matice zpětné DCT.

Uvedená vlastnost náhodných měřicích matic je někdy nazývána jako univerzalita těchto matic.

2.2 Numerické metody

Úlohu (P0) je možné řešit různými algoritmy, které lze rozdělit na tři základní skupiny:



Obr. 2.10: Vlevo: při rekonstrukci se použije matice $\mathbf{AU}$, řešením je vektor $\mathbf{x}$. Vpravo: pozorovaný vektor $\mathbf{z}$ není přímo řídký, ale víme, že je řídký v bázi $\mathbf{U}^{-1}$.

- relaxační – založeny na „relaxaci“ ℓ_0-normy, tedy nahrazení diskrétní ℓ_0-normy spojitou funkcí (nebo jinou normou, např. ℓ_1).
- žravé (*greedy*) – iterační algoritmy hledající řídká řešení. Každá iterace je složena ze dvou fází: selekce (*greedy selection*) a aktualizace (*greedy update*). Selekce vybere atom (nebo skupinu atomů), který nejlépe aproximuje vektor v dané iteraci. V aktualizaci je atom doplněn do aktuální aproximace celého vektoru, ta je spočtena znovu, tentokrát i s nově přidaným atomem
- hybridní – kombinují vlastnosti obou předchozích skupin.

2.2.1 Relaxační algoritmy

Jedním z nejjednodušších postupů, jak řešit úlohu ℓ_0-minimalizace relaxací, je převedení ℓ_1-minimalizace na úlohu lineárního programování.

Abychom mohli řešit ℓ_1-minimalizaci pomocí lineárního programování, musíme se zbavit absolutních hodnot, které jsou obsaženy v ℓ_1-normě. Lze tak učinit nahrazením vektoru $\mathbf{x} \in \mathbb{C}^N$ vektorem $\mathbf{v} \in \mathbb{C}^{2N}$ a rozšířením rekonstrukční matice na $\mathbf{A}' = (\mathbf{A}| - \mathbf{A})$. Úloha (P1) je pak ekvivalentní s úlohou lineárního programování, pokud se omezíme na reálné signály.

$$\min \sum_{j=1}^{2N} v_j \quad \text{vzhledem k} \quad \mathbf{v} \geq 0,\ (\mathbf{A}| - \mathbf{A})\,\mathbf{v} = \mathbf{y}. \tag{2.12}$$

Řešení $\mathbf{x}$ pak získáme jednoduše jako

$$\mathbf{x} = (\mathbf{I}| - \mathbf{I})\,\mathbf{v}. \tag{2.13}$$

Tato úloha bývá také označována jako *Basis Pursuit* (BP). Lineární programování se přes všechny své výhody (např. hotové funkční algoritmy) kvůli své časové a paměťové náročnosti na řešení rozsáhlejších problémů (větších N) nehodí. Jako vhodnější se jeví použití specializovaných metod, jako např. *IRLS* nebo *LARS* [5].

2.2.2 Žravé algoritmy

Mezi žravé algoritmy patří především *Matching Pursuit* (MP), *Orthogonal Matching Pursuit* (OMP) a *Weak Matching Pursuit* (WMP). První jmenovaný je jedním z prvních algoritmů, které byly použity k hledání řídkých řešení. Výhodou algoritmu je, že vždy konverguje, nevýhodou, že to trvá příliš dlouho. *Orthogonal Matching Pursuit* je speciální variantou MP přidávající ortogonalizaci reziduí v každé iteraci a *Weak Matching Pursuit* je zjednodušením MP. Podrobný popis těchto algoritmů je možné nalézt např. v [15].

Dalším žravým algoritmem je *Tresholding* algoritmus, který již nepatří do skupiny MP, je o hodně jednodušší. Tento algoritmus dává dobré výsledky pouze v jednoduchých případech.

Nejpoužívanějším algoritmem z této skupiny je OMP – nabízí rychlou konvergenci za poměrně nízké výpočetní náročnosti.

2.2.3 Hybridní algoritmy

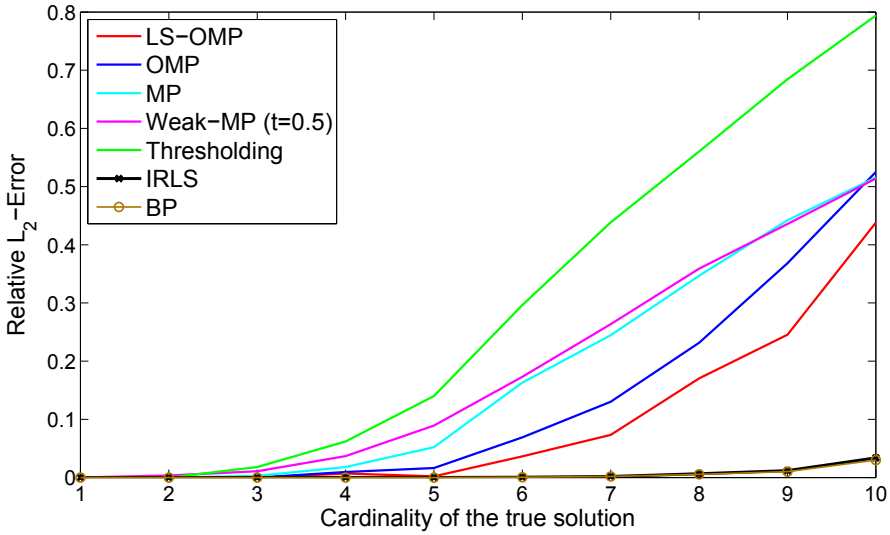
Poslední skupina algoritmů jsou hybridní algoritmy. Ty kombinují vlastnosti relaxačních i žravých algoritmů. Můžeme mezi ně zařadit speciální algoritmy, jako je např. CoSaMP (*Compressive Sampling Matching Pursuit*) [14] nebo skupinu *Iterative-Shrinkage* algoritmů [6].

2.2.4 Srovnání algoritmů

Srovnání jednotlivých algoritmů jsem převzal z článku [15]. Simulace byly prováděny v prostředí MATLAB na počítači s procesorem Intel(R) Core(TM) i5 CPU 680 @ 3,60 GHz a 4 GB paměti RAM.

Z hlediska relativní chyby rekonstrukce v normě ℓ_2 jsou relaxační algoritmy lepší než žravé algoritmy, což je patrné z grafu na obrázku 2.11. Nejlepší výsledky tedy dávají algoritmy BP a IRLS, ze žravých algoritmů jsou na tom nejlépe LS-OMP a OMP.

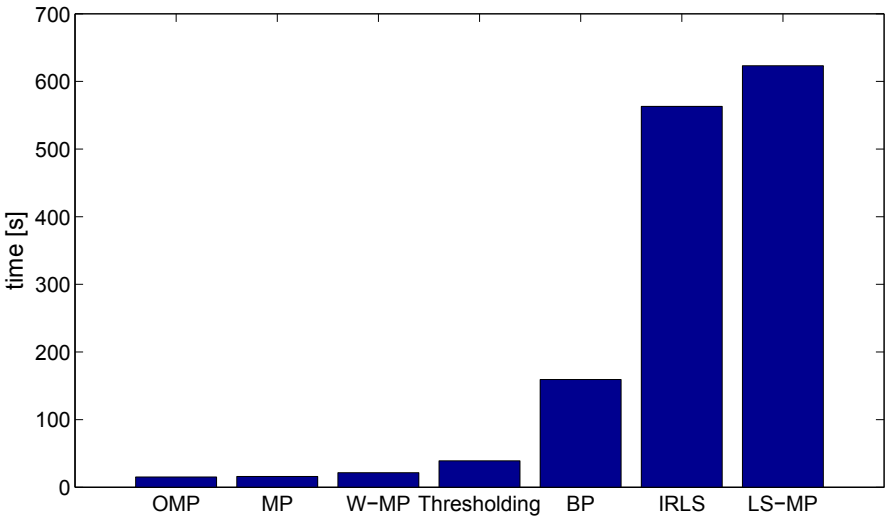
Co se týče výpočetní náročnosti, jsou na tom naopak výrazně lépe žravé algoritmy. Srovnání časové náročnosti je uvedeno v grafu na obrázku 2.12. Relaxační algoritmy jsou několikanásobně pomalejší než žravé algoritmy.



Obr. 2.11: Výkonnost algoritmů podle relativní chyby v normě ℓ_2 v závislosti na kardinalitě řešení [15]

Vezmeme-li obě uvedená kritéria v úvahu, dobrým kompromisem je použití algoritmu OMP – při své velmi malé výpočetní náročnosti dává zároveň poměrně dobré výsledky.

Schopnosti jednotlivých algoritmů jsem ověřil jednoduchým úkolem nalézt řešení soustavy (2.11). Výsledky jsou uvedeny v tabulce 2.1. Tučně jsou vyznačeny algoritmy, které se přiblížily ke správnému řešení $[1, 0, 0]$. Zde je názorně vidět, že při nevhodné volbě měřicí matice relaxační algoritmy nemají šanci nalézt řídké řešení, zatímco žravé algoritmy stále mohou k řídkému řešení dospět. Nemusí tomu tak



Obr. 2.12: Výpočetní náročnost algoritmů [15]

Algoritmus	x_1	x_2	x_3
MP	1	0	0
OMP	0	$0,5556$	$0,4444$
Tresholding	1	0	$2,2 \cdot 10^{-16}$
BP	$0,3346$	$0,3697$	$0,2958$
IRLS	$0,3361$	$0,3689$	$0,2951$

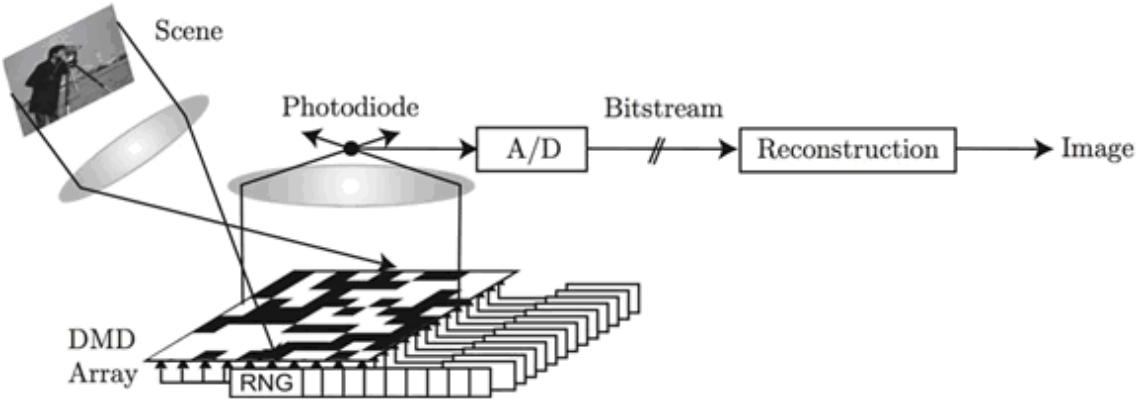
Tab. 2.1: Srovnání výsledků řešení soustavy rovnic (2.11) pomocí různých algoritmů

být ovšem vždy, ostatně výše vyzdvihovaný algoritmus OMP řídké řešení v tomto případě nenašel, zatímco jednodušší Tresholding ano.

3 VÝSLEDKY STUDENTSKÉ PRÁCE

Po teoretické části práce se dostáváme k praktickým výsledkům, zejména k simulaci one-pixel camery. Všechny zdrojové texty předvedené v této kapitole jsou také k dispozici v příloze bez podrobnějšího komentáře.

One-pixel camera je zařízení složené z pole zrcátek, které soustřeďují obraz do jediného senzoru. Každé zrcátko buď odráží obraz do senzoru nebo mimo senzor. Nastavení těchto zrcátek se s časem mění, v každém okamžiku odpovídá konfigurace všech zrcátek jednomu řádku matice $\mathbf{A}$. Na senzor tedy dopadá vždy součet několika prvků vektoru $\mathbf{x}$ (resp. $\mathbf{z}$, pokud je signál řídký v jiné bázi). Je zřejmé, že konstrukce one-pixel camery značně omezuje možnosti měřicí matice $\mathbf{A}$. Ta se blíží bernoulliovské náhodné matici s tím rozdílem, že neobsahuje kladné a záporné hodnoty, ale jen nuly a jedničky.



Obr. 3.1: Schéma one-pixel camery, která byla postavena na Rice University [1]

3.1 Simulace one-pixel camery v Matlabu

Simulovaná one-pixel camera generuje náhodný řídký vektor $\mathbf{x}$, který pak „pozoruje“ přes měřicí matici $\mathbf{A}$ (náhodná bernoulliovská matice). Ze získaného měření $\mathbf{y}$ se pak snaží zpětně rekonstruovat původní signál $\mathbf{x}$.

Vlastní implementace tedy začíná nastavením parametrů. Nejprve zvolíme velikost vektoru $\mathbf{x} \in \mathbb{R}^N$, podvzorkování (zde 0.4) a řídkost.

```
%% Parameters
N = 100;
m = round(N*0.4);
k = round(m*0.3);
```

Následně vygenerujeme měřicí matici $\mathbf{A} \in \mathbb{R}^{m \times N}$. Použijeme náhodnou bernoulliovskou matici, kde hodnoty prvků matice nabývají hodnot $\pm 1/\sqrt{m}$ se stejnou pravděpodobností. Funkce *bernmtx* je uvedena v příloze A.

```
%% Measurement matrix
A = bernmtx(m, N);
```

Vygenerování náhodného řídkého vektoru $\mathbf{x}$ můžeme provést dvěma způsoby. Buď bude signál řídký přímo v časové (prostorové) doméně a pak bude unitární matice $\mathbf{U} = \mathbf{I}$ identita:

```
%% Original vector
U = eye(N);
perm = randperm(N);
x = zeros(N, 1);
for ii = 1:k
    x(perm(ii)) = rand();
end
```

nebo bude signál $\mathbf{x}$ řídký v libovolné bázi (zde $\mathbf{U}$ je matice zpětné DCT) a bude vygenerován odlišně:

```
%% Original vector
U = dctmtx(N)';
x = zeros(N, 1);
perm = randperm(N);
for kk = 1:k
    x(perm(kk)) = rand();
end
z = idct(x_dct);
```

Nyní máme vygenerovanou měřicí matici i signál $\mathbf{x}$, můžeme proto provést měření.

```
%% Encoding
y = A*x;
```

Vektor $\mathbf{y} \in \mathbb{R}^m$ jsou data, která získáme z one-pixel camery. Tato data budeme chtít následně rekonstruovat a získat původní vektor $\mathbf{x}$. Použijeme lineární programování k řešení úlohy (P1). Účelová funkce f vrací součet všech prvků vektoru, vektor *lb* tvoří dolní hranici řešení, v našem případě musí být řešení kladné. Výsledkem úlohy lineárního programování je vektor $\mathbf{v}$, ze kterého ještě musíme získat řešení $\mathbf{x}$ podle (2.13). Z následujícího kódu je také vidět, jak je použita při rekonstrukci unitární matice $\mathbf{U}$.

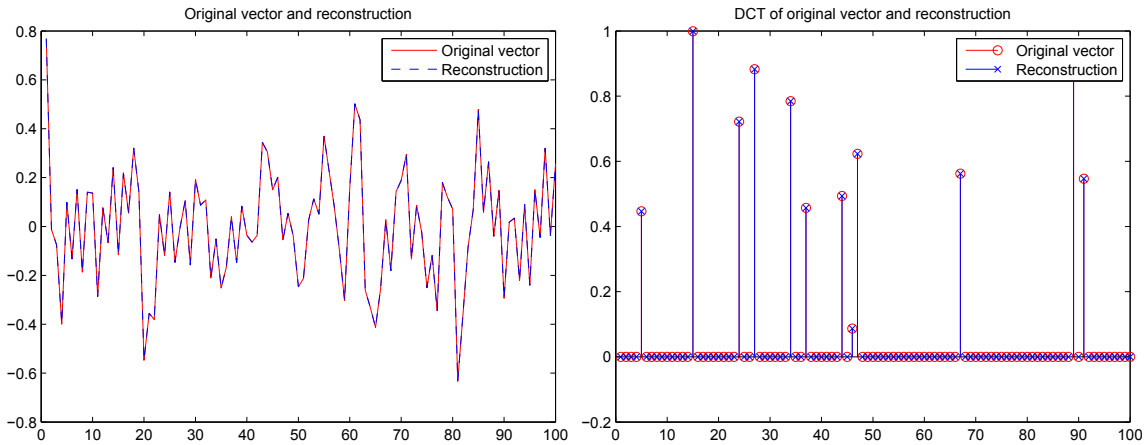
```
%% Decoding
f = ones(2*N, 1);
lb = zeros(2*N, 1);
v = linprog(f, [], [], [A*U -A*U], y, lb);
x_ = U*[eye(N) -eye(N)]*v;
```

Po provedení rekonstrukce můžeme porovnat původní a rekonstruovaný signál.

```
%% Results
figure(1);
subplot(1, 2, 1);
plot(x, '-r');
hold('on');
plot(x_, '--b');
hold('off');
title('Original vector and reconstruction');
legend('Original vector', 'Reconstruction');

subplot(1, 2, 2);
stem(dct(x), 'r');
hold('on');
stem(dct(x_), 'xb');
hold('off');
title('DCT of original vector and reconstruction')
legend('Original vector', 'Reconstruction');
```

Výsledek jedné simulace je vidět na obrázku 3.2. Zde jsem ukázal situaci, kdy se sice signály lišily, ale nijak výrazně. To, jestli se podaří signál obnovit zcela bez chyby nebo se budou signály lišit, záleží na zvolení parametrů m a k. Pro vhodně zvolené parametry může být rekonstrukce ale téměř bezchybná. Současně jsem ověřil, že rekonstrukční vlastnosti matice $\mathbf{A}$ jsou invariantní vůči unitární transformaci.



Obr. 3.2: Výsledek simulace one-pixel camery. Zde $N = 100$, $m = 40$ a $k = 12$, signál je řídký v DCT. Vlevo: originální signál a jeho rekonstrukce z měření $\mathbf{y}$, vpravo: DCT obou signálů.

3.1.1 Vliv šumu na rekonstrukci

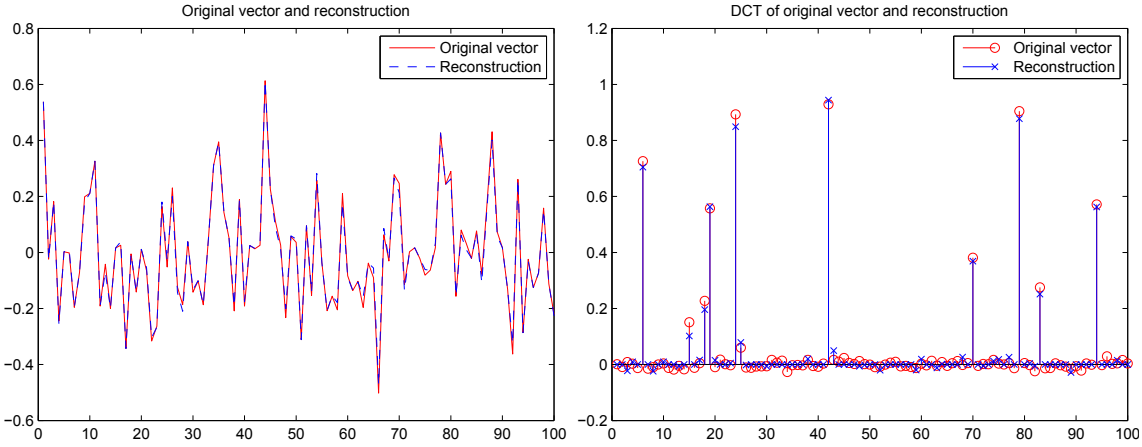
V praxi je potřeba počítat ještě s jedním faktorem, kterým je šum. Šum se do naměřených dat může dostat mnoha způsoby – nelinearitou senzoru, kvantováním apod. V simulaci one-pixel camery je možné šum přidat na několika úrovních. Přidáním šumu do vstupního signálu $\mathbf{x}$ simulujeme, že signál není zcela řídký.

```
% Additive noise
x = x + 0.01*randn(N, 1);
```

Naopak přidáním šumu k měření $\mathbf{y}$ simulujeme neideální vlastnosti senzoru.

```
% Additive noise
y = y + 0.01*randn(m, 1);
```

V našem případě přidáváme gaussovský bílý šum k signálu $\mathbf{x}$. Rekonstrukce je opět úspěšná, viz obrázek 3.3, DCT koeficienty se ovšem začínají mírně lišit a průběh signálu v časové oblasti již nekopíruje původní signál zcela dokonale.



Obr. 3.3: Výsledek simulace one-pixel camery s přidáním šumu k signálu $\mathbf{x}$.

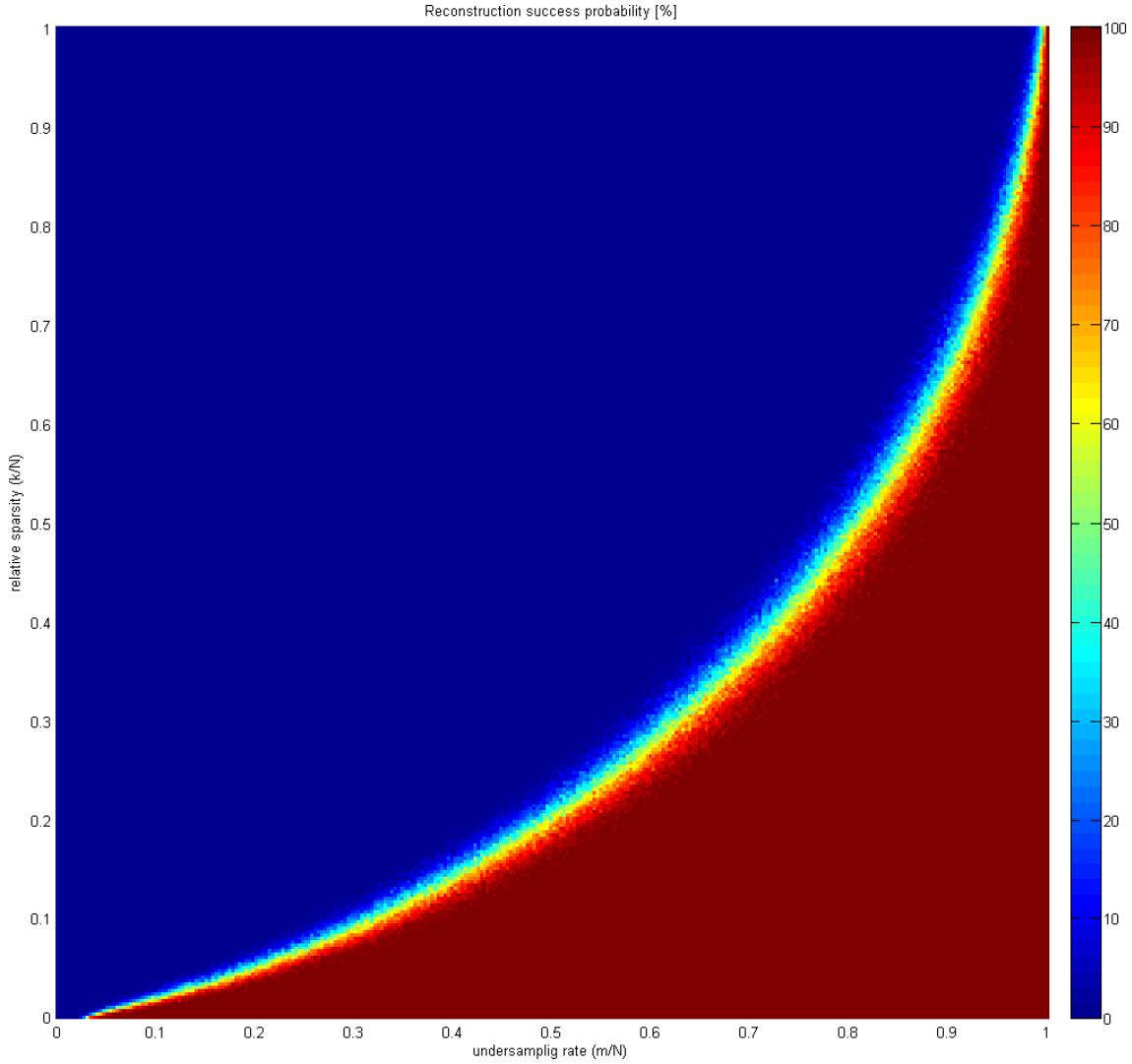
3.1.2 Úspěšnost rekonstrukce v závislosti na podvzorkování a řídkosti

Funkčnost one-pixel camery jsme již ověřili v praxi. Velmi důležité je také vědět, jaká je závislost úspěšnosti rekonstrukce na parametrech m a k, tedy na míře podvzorkování a řídkosti signálu. Pro tento účel jsem napsal skript v MATLABu, který postupně zkouší všechny kombinace $m \in \{1, \ldots, N\}$ a $k \in \{1, \ldots, m\}$ a pro každou kombinaci provede několik simulací. Relativní chyba rekonstrukce

$$\sigma = \frac{\|\mathbf{x} - \mathbf{x}^*\|_2}{\|\mathbf{x}\|_2} \tag{3.1}$$

je porovnávána s prahem ϵ a relativní četnost úspěšné rekonstrukce je pak zaznamenána do grafu (viz obrázek 3.4).

Z grafu je patrné, že pro úspěšnou rekonstrukci podvzorkovaného signálu je třeba, aby byl dostatečně řídký. Čím vyšší je míra podvzorkování, tím řidší musí být signál, aby ho bylo možné následně rekonstruovat.



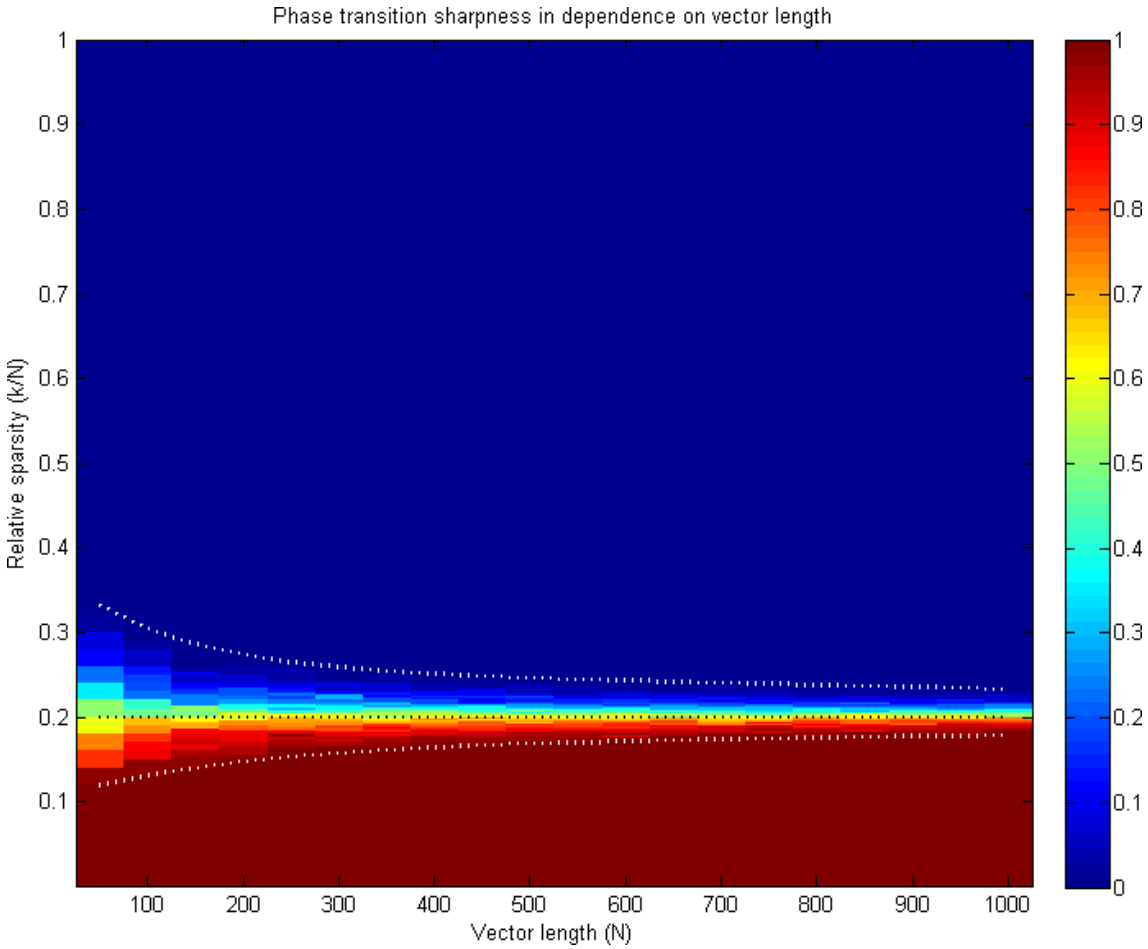
Obr. 3.4: Relativní četnost úspěšné rekonstrukce v závislosti na míře podvzorkování a řídkosti signálu ($N = 300$)

3.1.3 Fázový přechod

Oblast mezi jistou a nemožnou rekonstrukcí signálu se nazývá fázový přechod (*Phase Transition*). Tvar tohoto přechodu v závislosti na míře podvzorkování a řídkosti vektoru je vidět na obrázku 3.4.

Mě zajímalo, jestli se mění šířka tohoto přechodu s měnící se délkou vektoru. Pro tento účel jsem napsal skript v prostředí MATLAB (viz příloha A.5), který pro konstantní míru podvzorkování $\frac{m}{N} = 0,5$ prováděl opakovaně pokusy a postupně zvyšoval délku vektoru od $N = 50$ po $N = 1000$. Výsledek této simulace je uveden na obrázku 3.5.

Je evidentní, že šířka fázového přechodu závisí na délce vektoru $\mathbf{x}$, s rostoucí délkou N klesá šířka fázového přechodu. Z praktického hlediska je důležité, že s rostoucí



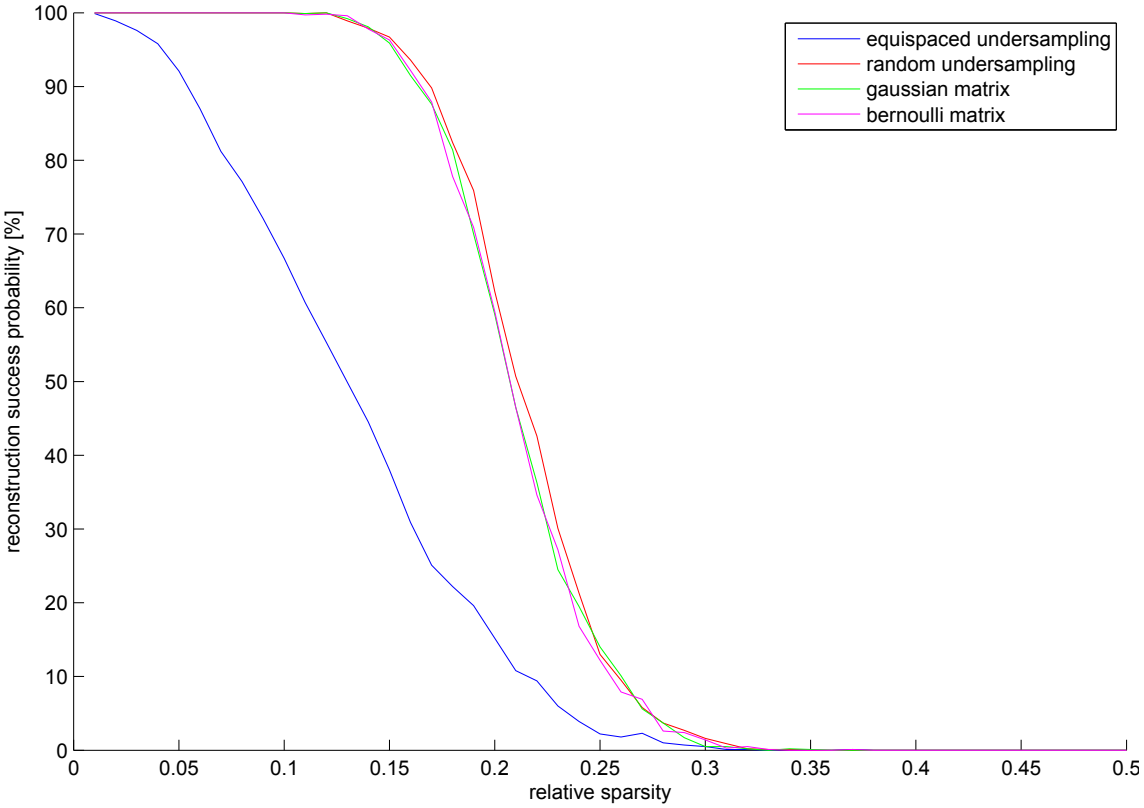
Obr. 3.5: Ostrost fázového přechodu v závislosti na délce vektoru

délkou vektoru **x** roste i maximální relativní řídkost vektoru, při kterém můžeme zaručit úspěšnou rekonstrukci.

3.1.4 Vliv použité měřicí matice na úspěšnost rekonstrukce

Volba vhodné měřicí matice je důležitým předpokladem pro úspěšnou rekonstrukci signálu. Několik příkladů takovýchto matic jsem uvedl v předchozích kapitolách, pro úplnost jsem pomocí simulace provedl srovnání úspěšnosti rekonstrukce při použití různých měřicích matic. Výsledky tohoto experimentu jsou vyneseny do grafu na obrázku 3.6.

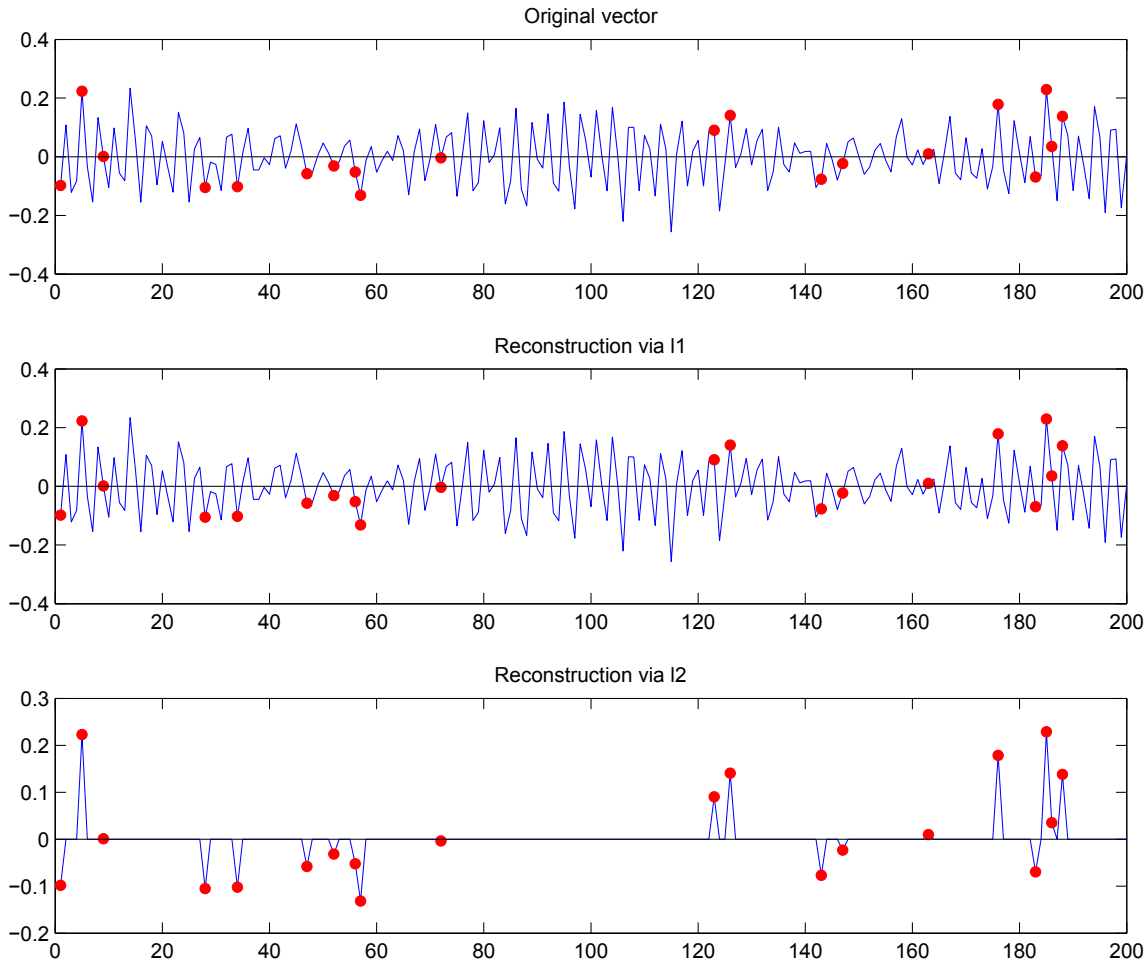
Srovnával jsem 4 typy matic – rovnoměrné podvzorkování, náhodné podvzorkování, gaussovskou a bernoulliovskou náhodnou matici. Překvapivě nejlepšího výsledku bylo dosaženo náhodným podvzorkováním signálu, naopak při rovnoměrném podvzorkování byla úspěšnost rekonstrukce velmi malá, což se dalo očekávat. Gaussovské a bernoulliovské náhodné matice mají velmi podobné vlastnosti, jsou jen



Obr. 3.6: Vliv použité měřicí matice na úspěšnost rekonstrukce

o velmi málo horší než náhodné podvzorkování. Pokus byl proveden pro délku vektoru $N = 100$ a dvojnásobné podvzorkování (tedy $m = 50$). Signál byl řídký ve frekvenční oblasti. Celý zdrojový kód experimentu je uveden v příloze A.6.

Příklad náhodného podvzorkování signálu je uveden na obrázku 3.7. Nejprve je vygenerován signál řídký ve frekvenční oblasti (zde délka vektoru $N = 200$ a relativní řídkost $\frac{k}{N} = \frac{1}{30}$), poté je vybráno náhodně $m = 20$ vzorků v časové oblasti (červené body v grafu). Rekonstrukce je nejprve provedena ℓ_1-minimalizací a poté pro srovnání ℓ_2-minimalizací. Srovnáním rekonstruovaných signálů s původním zjistíme, že rekonstrukce pomocí ℓ_1-minimalizace je téměř dokonalá, zatímco pomocí ℓ_2-minimalizace získáme signál s nejnižší energií.



Obr. 3.7: Náhodné podvzorkování signálu a jeho následná rekonstrukce pomocí ℓ_1- a ℓ_2-minimalizace.

4 ZÁVĚR

Cílem této práce bylo seznámit se s aktuálním stavem problematiky compressive sampling a implementovat simulaci one-pixel camery v prostředí MATLAB.

Přesto, že je compressive sampling novinka v oblasti zpracování signálů, ve světě se jím již seriózně zabývá velké množství vědeckých týmů a tento počet stále roste. Začínají se objevovat i praktické aplikace, mezi nimi např. rekonstrukce obrazu z nukleární magnetické rezonance (NMR) [13] nebo výpočetní tomografie (zde je důležitý především nižší počet měření, tedy kratší procedura a nižší úroveň ozáření), rekonstrukce obrazu z ultrazvuku, *inpainting* (obnova poškozených snímků) [7], levnější kamery pracující v neviditelné oblasti spektra a další [16]. Jiná je situace v České republice – tato práce je teprve jedna z prvních.

V řešení práce jsem uvedl řadu teoretických podmínek zaručujících rekonstrukci signálu. Tyto podmínky se však ukazují být příliš přísné (a výpočetně náročné) a nedají se tak prakticky využít k posouzení rekonstrukčních vlastností měřicích matic.

Podařilo se mi vytvořit simulaci one-pixel camery a ověřit její funkčnost. Využil jsem při tom lineárního programování jako nástroje pro minimalizaci. Ukazuje se ale, že pro rekonstrukci signálů o vyšším počtu vzorků je třeba hledat specializované algoritmy. Ve své práci proto uvádím přehled algoritmů různých typů umožňujících rekonstrukci řídkých signálů. Dále jsem provedl řadu experimentů v prostředí MATLAB a zjistil jsem tak např. vlastnosti tzv. fázového přechodu nebo rekonstrukční vlastnosti různých měřicích matic.

LITERATURA

[1] BARANIUK, R.G. Compressive Sensing. *IEEE Signal Processing Magazine* 2007, 24, s. 118-120,124

[2] BRUCKSTEIN, A.M; DONOHO, D.L; ELAD, M. From Sparse Solutions of System of Equations to Sparse Modeling of Signals and Images. *SIAM Review.* 2009, 51, 1, s. 34-81.

[3] CANDES, E.J., Tao, T. Decoding by Linear Programming. *IEEE Trans. Inf. Th.* 2005, 51, 12, s. 4203-4215

[4] DONOHO, D.L; ELAD, M. Optimally Sparse Representation in General (non-Orthogonal) Dictionaries via l^1 Minimization *Proc. Natl Acad. Sci.* 2003, 100, s. 2197-2202

[5] DRORI, I.; DONOHO, D.L. Solution of l_1 Minimization Problems by LARS/Homotopy Methods *IEEE International Conference on Acoustics, Speech, and Signal Processing* 2006

[6] ELAD, M. *Sparse and Redundant Representations: From Theory to Applications in Signal and Image Processing.* Springer, 2010, ISBN 9781441970107.

[7] FADILI, M.; STARCK, J.L.; MURTAGH, F. Inpainting and zooming using sparse representations. *The Computer Journal* 2006

[8] FORNASIER, M; RAUHUT, H. *Handbook of Mathematical Methods in Imaging*, kapitola Compressive Sensing. Springer, 2011, ISBN 978-0-387-92920-0.

[9] CHEN, S.S.; DONOHO, D.L.; SAUNDERS, M.L. Atomic Decomposition by Basic Pursuit. *SIAM Review.* 2001, 43, 1, s. 129-159.

[10] JPEG. In *Wikipedia : the free encyclopedia* [online]. St. Petersburg (Florida) : Wikipedia Foundation, 5. 9. 2005, last modified on 28. 8. 2010 [cit. 2010-12-15]. Dostupné z WWW: `http://cs.wikipedia.org/wiki/JPEG`.

[11] JPEG 2000. In *Wikipedia : the free encyclopedia* [online]. St. Petersburg (Florida) : Wikipedia Foundation, 29. 7. 2006, last modified on 24. 6. 2010 [cit. 2010-12-15]. Dostupné z WWW: `http://cs.wikipedia.org/wiki/JPEG_2000`.

[12] KUFNER, A. *Geometrie Hilbertova prostoru.* 2. vydání. Praha : SNTL, 1975. 248 s.

[13] LUSTIG M.; DONOHO D.; PAULY J.M. Sparse MRI: The application of compressed sensing for rapid MR imaging. *Magnetic Resonance in Medicine* 2007, 58, 6, s. 1182-1195

[14] NEEDEL, D; TROPP, J. A. CoSaMP: Iterative signal recovery from incomplete and inaccurate samples. Technická zpráva, California Institute of Technology, Pasadena, 2008.

[15] ŠPIŘÍK, J. Algorithms for Computing Sparse Solutions. In *Proceedings of the 17th Conference STUDENT EEICT 2011 Volume 3*. Brno: NOVPRESS s.r.o., 2011. s. 123-127. ISBN: 978-80-214-4273- 3.

[16] *Compressed Sensing Hardware* [online]. 2010 [cit. 2010-12-15]. Dostupné z WWW: `http://sites.google.com/site/igorcarron2/compressedsensinghardware`.

SEZNAM SYMBOLŮ, VELIČIN A ZKRATEK

ONB	Ortonormální báze
RIP	Restricted Isometry Property
NSP	Null Space Property
LP	lineární programování
$\|x\|_p$	l_p-norma vektoru x
$\|\mathbf{A}\|$	norma matice $\mathbf{A}$
$\kappa(\mathbf{A})$	podmíněnost matice $\mathbf{A}$
$\|T\|$	kardinalita (počet prvků) množiny T
$\ker \mathbf{A}$	jádro matice $\mathbf{A}$
$\sigma_k(x)_p$	chyba nejlepší aproximace k-řídkého vektoru x v normě l_p
$spark(\mathbf{A})$	nejmenší počet sloupců matice $\mathbf{A}$, které jsou lineárně závislé
$\mu(\mathbf{A})$	vzájemná koherence matice $\mathbf{A}$
BP	Basis Pursuit
IRLS	Iteratively Reweighted Least Squares
MP	Matching Pursuit
OMP	Orthogonal Matching Pursuit
WMP	Weak Matching Pursuit
CoSaMP	Compressive Sampling Matching Pursuit

SEZNAM PŘÍLOH

A MATLAB SKRIPTY

A.1 Simulace one pixel camery

```
%% Parameters
N = 100;
m = round(N*0.4);
k = round(m*0.3);

%% Measurement matrix
A = bernmtx(m, N);
%A = gaussmtx(m, N);

%% Original vector (sparse)
%U = eye(N);
%perm = randperm(N);
%x = zeros(N, 1);
%for ii = 1:k
%    x(perm(ii)) = rand();
%end
% Additive Noise
%x = x + 0.01*randn(N, 1);

%% Original vector (sparse in frequency)
U = dctmtx(N)';
x_dct = zeros(N, 1);
perm = randperm(N);
for kk = 1:k
    x_dct(perm(kk)) = rand();
end
% Additive Noise
x_dct = x_dct + 0.01*randn(N, 1);

x = idct(x_dct);

%% Encoding
y = A*x;

%% Decoding
f = ones(2*N, 1);
lb = zeros(2*N, 1);
v = linprog(f, [], [], [A*U -A*U], y, lb);
x_ = U*[eye(N) -eye(N)]*v;

%% Results
figure(1);
plot(x, 'r');
hold('on');
plot(x_, 'b');
hold('off');
title('Original vector and reconstruction');
legend('Original vector', 'Reconstruction');

figure(2);
stem(dct(x), 'r');
hold('on');
```

```
stem(dct(x_), 'b');
hold('off');
title('DCT of original vector and reconstruction');
legend('Original vector', 'Reconstruction');
```

A.2 Generátor Bernoulliovských matic

```
function [ A ] = bernmtx( m, n )
%BERNOULLI Generates matrix of size m x n with bernoulli random variables.

    A = (double(rand(m, n) <= 0.5) - 0.5) * 2 / sqrt(m);

end
```

A.3 Generátor Gaussovských matic

```
function [ A ] = gaussmtx( m, n )
%GAUSSMTX Generates matrix of size n x m with gaussian random variables.

    A = randn(m, n) / m;

end
```

A.4 Restricted Isometry Property

Následující funkce vrací RIP konstantu pro danou matici $\mathbf{A}$ a řídkosti $(2, \ldots, k)$. Časová složitost je vysoká, neboť je potřeba vyzkoušet veškeré kombinace sloupců matice $\mathbf{A}$ o počtu nejvýše k. Proto pro větší matice nemá smysl RIP počítat.

```
function [ deltas ] = rip( A, k )
%RIP Returns Restricted Isometry Property of order k of matrix A

    deltas = zeros(k-1, 1);
    for kk = 2:k
        T = nchoosek(1:size(A, 2), kk);
        for ii = 1:size(T, 1)
            AT = A(:, T(ii, :));
            d = norm(AT'*AT - eye(kk));
            if d > max(deltas)
                deltas(kk) = d;
            end
        end
    end

end
```

A.5 Fázový přechod

```
%% Parameters
mtx = 'bern'; % bern | gauss
Ns = [50 100 150 200 250 300 350 400 450 500 550 600 650 700 750 800 850 900 ↩
    950 1000];
epsilon = 1e-1;
num = 100;
options = optimset('Display', 'off');
noise = 0;

prob = cell(2, length(Ns));
load('phase_trans_test.mat');

%% Test
for Ni = 1:2:length(Ns)
    N = Ns(Ni);
    m = N/2;
    disp(['N = ' num2str(N)]);
    disp(['m = ' num2str(m)]);
    f = ones(2*N, 1);
    lb = zeros(2*N, 1);

    prob{1, Ni} = 1/N:1/N:0.5;
    prob{2, Ni} = zeros(1, m);
    tmp_prob = zeros(1, m);

    parfor k = 1:m
        disp(['k/m = ' num2str(k) '/' num2str(m) ' = ' num2str(k/m)]);

        for ii = 1:num
            %% Measurement matrix
            switch mtx
                case 'bern'
                    A = bernmtx(m, N);
                case 'gauss'
                    A = gaussmtx(m, N);
                otherwise
                    disp('Unknown mtx type!');
            end

            %% Original vector
            perm = randperm(N);
            x = zeros(N, 1);
            for jj = 1:k
                x(perm(jj)) = rand();
            end

            %% Noise
            x = x + noise*randn(N, 1);

            %% Encoding
            y = A*x;

            %% Decoding
            x_ = l1eq_pd(A'*y, A, [], y, 1e-3);

            %% Results
```

```
            if norm(x - x_)/norm(x) < epsilon
                tmp_prob(1, k) = tmp_prob(1, k) + 1/num;
            end
        end
    end

    prob{2, Ni} = tmp_prob;
    save('phase_trans_test.mat', 'prob', 'Ns');
end

%% Visualization
sz = [1000 size(prob, 2)];
hist = zeros(sz);

for ii = 1:sz(2)
    for jj = 1:sz(1)
        prob_sz = size(prob{2,ii});
        hist(sz(1)-jj+1,ii) = mean(prob{2,ii}(floor((jj-1)*prob_sz(2)/sz(1)+1)↩
            :ceil(jj*prob_sz(2)/sz(1))));
    end
end

imagesc(flipud(hist));
title('Phase transition sharpness in dependence on vector length');
xlabel('Vector length (N)');
ylabel('Relative sparsity (\\kappa)');
colorbar;
xTicks = get(gca, ['x' 'Tick']);
yTicks = get(gca, ['y' 'Tick']);
set(gca, ['x' 'TickLabel'], Ns(xTicks));
set(gca, ['y' 'TickLabel'], 0.1:0.1:1);
set(gca, 'ydir', 'normal');

hold('on');
plot(1:length(Ns), (0.168*exp(6.232e-5*Ns)-0.06354*exp(-0.005076*Ns))*max(↩
    yTicks), ':w', 'LineWidth', 2);
plot(1:length(Ns), (0.1158*exp(-0.008537*Ns)+0.2572*exp(-9.954e-5*Ns))*max(↩
    yTicks), ':w', 'LineWidth', 2);
plot(1:length(Ns), ones(1,length(Ns))*0.2*max(yTicks), ':k', 'LineWidth', 2);
hold('off');
```

A.6 Srovnání měřicích matic

```
%% Parameters
N = 100;
m = N/2;
U = dctmtx(N)';
epsilon = 1e-1;

%% Test - equispaced sampling
prob_equi = zeros(1, m);
% Measurement matrix
A = zeros(m, N);
samples = 1:2:N;
for ii = 1:m
```

```
        A(ii, samples(ii)) = 1;
    end
    parfor kk = 1:m
        for ii = 1:1000
            % Original vector
            x_dct = zeros(N, 1);
            perm = randperm(N);
            for jj = 1:kk
                x_dct(perm(jj)) = 2*rand()-1;
            end
            x = idct(x_dct);

            % Encoding
            y = A*x;

            % Decoding
            x_ = U*l1eq_pd(U'*A'*y, A*U, [], y, 1e-3);

            % Evaluation
            if norm(x - x_) / norm(x) < epsilon
                prob_equi(kk) = prob_equi(kk) + 1;
            end
        end
    end
    prob_equi = prob_equi ./ 10;

    %% Test - random sampling
    prob_rand = zeros(1, m);
    parfor kk = 1:m
        for ii = 1:1000
            % Measurement matrix
            A = zeros(m, N);
            samples = randperm(N);
            samples = sort(samples(1:m));
            for ii = 1:m
                A(ii, samples(ii)) = 1;
            end

            % Original vector
            x_dct = zeros(N, 1);
            perm = randperm(N);
            for jj = 1:kk
                x_dct(perm(jj)) = 2*rand()-1;
            end
            x = idct(x_dct);

            % Encoding
            y = A*x;

            % Decoding
            x_ = U*l1eq_pd(U'*A'*y, A*U, [], y, 1e-3);

            % Evaluation
            if norm(x - x_) / norm(x) < epsilon
                prob_rand(kk) = prob_rand(kk) + 1;
            end
        end
    end
    prob_rand = prob_rand ./ 10;
```

```
%% Test - Gaussian measurement matrix
prob_gauss = zeros(1, m);
parfor kk = 1:m
    for ii = 1:1000
        % Measurement matrix
        A = gaussmtx(m, N);

        % Original vector
        x_dct = zeros(N, 1);
        perm = randperm(N);
        for jj = 1:kk
            x_dct(perm(jj)) = 2*rand()-1;
        end
        x = idct(x_dct);

        % Encoding
        y = A*x;

        % Decoding
        x_ = U*l1eq_pd(U'*A'*y, A*U, [], y, 1e-3);

        % Evaluation
        if norm(x - x_) / norm(x) < epsilon
            prob_gauss(kk) = prob_gauss(kk) + 1;
        end
    end
end
prob_gauss = prob_gauss ./ 10;

%% Test - Bernoulli measurement matrix
prob_bern = zeros(1, m);
parfor kk = 1:m
    for ii = 1:1000
        % Measurement matrix
        A = bernmtx(m, N);

        % Original vector
        x_dct = zeros(N, 1);
        perm = randperm(N);
        for jj = 1:kk
            x_dct(perm(jj)) = 2*rand()-1;
        end
        x = idct(x_dct);

        % Encoding
        y = A*x;

        % Decoding
        x_ = U*l1eq_pd(U'*A'*y, A*U, [], y, 1e-3);

        % Evaluation
        if norm(x - x_) / norm(x) < epsilon
            prob_bern(kk) = prob_bern(kk) + 1;
        end
    end
end
prob_bern = prob_bern ./ 10;
```

```
%% Visualization
figure(1);
hold('on');
plot(1/N:1/N:m/N, prob_equi, '-b');
plot(1/N:1/N:m/N, prob_rand, '-r');
plot(1/N:1/N:m/N, prob_gauss, '-g');
plot(1/N:1/N:m/N, prob_bern, '-m');
hold('off');
legend('equispaced undersampling', 'random undersampling', 'gaussian matrix', ↩
    'bernoulli matrix');
xlabel('relative sparsity');
ylabel('reconstruction success probability [%]');
xTicks = get(gca, ['x' 'Tick']);
set(gca, ['x' 'TickLabel'], xTicks/N);
```