University of Zagreb

Faculty of Science
Department of Mathematics

Marko Filipović

Sparse representations of signals for information recovery from incomplete data

Doctoral thesis

Zagreb, 2013.

Sveučilište u Zagrebu

Prirodoslovno-matematički fakultet
Matematički odjel

Marko Filipović

Rijetke reprezentacije signala s primjenom u obnavljanju informacija iz nepotpunih podataka

Doktorski rad

Zagreb, 2013.

University of Zagreb

Faculty of Science
Department of Mathematics

Marko Filipović

Sparse representations of signals for information recovery from incomplete data

Doctoral thesis

Supervisors:

- **dr. sc. Ivica Kopriva, senior scientist**
  Ruđer Bošković Institute, Zagreb
- **prof. dr. sc. Zlatko Drmač**
  Department of Mathematics, Faculty of Science, University of Zagreb

Zagreb, 2013.

Sveučilište u Zagrebu

Prirodoslovno-matematički fakultet
Matematički odjel

Marko Filipović

Rijetke reprezentacije signala s primjenom u obnavljanju informacija iz nepotpunih podataka

Doktorski rad

Mentori:

- **dr. sc. Ivica Kopriva, znanstveni savjetnik**
  Institut Ruđer Bošković, Zagreb

- **prof. dr. sc. Zlatko Drmač**
  Prirodoslovno-matematički fakultet, Matematički odjel, Sveučilište u Zagrebu

Zagreb, 2013.

Acknowledgements

I would like to thank my supervisor, dr. sc. Ivica Kopriva, for ideas that shaped this thesis, help and financial support through the scientific project funded by the Ministry of Science, Education and Sports of Republic of Croatia. I am also thankful to my co-supervisor, professor dr. sc. Zlatko Drmač, for the letter of recommendation when I applied for a position of research assistant at the Rudjer Bošković Institute, as well as for useful advices during my writing of this thesis. I am also thankful to other members of the thesis evaluation committee: professor Tapani Ristaniemi for accepting to be the member of the committee and for his positive critics; professor dr. sc. Miljenko Huzak for pointing out some inaccuracies in the early version of the thesis; and to professor dr. sc. Sven Lončarić, whose participation in the committee, as a recognized expert in the field of image processing, emphasizes the inter-disciplinary character of this thesis.

Zahvale

Zahvaljujem mentoru, dr. sc. Ivici Koprivi, bez čijih ideja, pomoći i financijske potpore kroz znanstveni projekt Ministarstva znanosti, obrazovanja i sporta RH ne bi bilo ove doktorske radnje. Zahvaljujem i su-voditelju, profesoru dr. sc. Zlatku Drmaču, na preporuci prilikom mog zapošljavanja na Institutu Ruđer Bošković, i korisnim savjetima prilikom pisanja doktorata. Zahvaljujem i ostalim članovima povjerenstva za ocjenu doktorske radnje: profesoru Tapaniju Ristaniemiju na prihvaćanju da bude član povjerenstva i na pozitivnim kritikama; profesoru dr. sc. Miljenku Huzaku što je ukazao na neke nepreciznosti u ranoj verziji radnje; i profesoru dr. sc. Svenu Lončariću, čije sudjelovanje u povjerenstvu, kao priznatog stručnjaka u području obrade slike, naglašava inter-disciplinarni karakter ove radnje.

Abstract

Mathematical modeling of inverse problems in imaging, such as inpainting, deblurring and denoising, results in ill-posed, i.e. underdetermined linear systems. Sparseness constraint is used often to regularize these problems. That is because many classes of discrete signals (e.g. natural images), when expressed as a vectors in a high-dimensional space, are sparse in some predefined basis or a frame (fixed or learned). An efficient approach to basis/frame learning is formulated using the independent component analysis (ICA) and biologically inspired linear model of sparse coding. In the learned basis, the inverse problem of data recovery and removal of impulsive noise is reduced to solving sparseness constrained underdetermined linear system of equations. The same situation occurs in bioinformatics data analysis when novel type of linear mixture model with a reference sample is employed for feature extraction. Extracted features can be used for disease prediction and biomarker identification.

Keywords

Independent component analysis, Source separation, Sparsity, Sparse component analysis, Sparse representation, Sparse signal reconstruction, Underdetermined linear system, Dictionary learning, K-SVD, Incomplete data, Missing data, Image inpainting, Salt-and-pepper noise, Nonlinear filtering, Feature extraction, Linear mixture model, Bioinformatics

Contents

Sažetak

0.1 Uvod

Rijetkost vektora ili matrice po definiciji znači da je 'mnogo' elemenata (vektora, odn. matrice) jednako nuli. Kompresibilnost ili približna rijetkost podrazumijeva da je mnogo elemenata približno jednako nuli. Mnogi signali koji se susreću u praksi su ili rijetki/približno rijetki ili postoji transformacija koja ih čini rijetkima. To je analitički model rijetkosti. Mnogi standardi kompresije (JPEG, audio kompresija) koriste neke poznate matematičke transformacije (diskretna kosinusna transformacija, valići) koje rezultiraju rijetkim prikazom signala. Iz analitičkog modela slijedi sintetički model: često je realno pretpostaviti da postoji baza (ili okvir) vektorskog prostora takva da se signali iz dane klase mogu prikazati kao linearna kombinacija od nekoliko vektora te baze/okvira. Kažemo da signal ima rijetku reprezentaciju u danoj bazi/okviru. U terminologiji obrade signala, takva baza/okvir se često naziva *rječnik*, a vektori te baze/okvira atomi. Rijetkost ili kompresibilnost su vrlo važni za praktične inverzne probleme u obradi slike, kao što su rekonstrukcija nedostajućih dijelova slike, uklanjanje šuma ili super-rezolucija. Navedeni problemi su općenito loše postavljeni ili vrlo loše uvjetovani, a pretpostavka rijetkosti rješenja ih regularizira i omogućava najsuvremenije rezultate. Rijetkost je vrlo korisna i za rješavanje problema razdvajanja signala [22], posebno u slučaju kad je broj mjerenja manji od broja izvornih signala. Uobičajeni naziv za razdvajanje signala uz pretpostavku rijetkosti je analiza rijetkih komponenata. Važna primjena analize rijetkih komponenata je u bioinformatici. Rijetkost omogućava izdvajanje značajki (primjer su geni u genomici) na osnovu danih uzoraka za koje su poznate labele (odn. uzorci su dobiveni od zdravih ili bolesnih pacijenata).

Gore navedeni pristupi rješavanju praktičnih problema u obradi slike i bioinformatici bazirani su na fundamentalnim rezultatima iz područja sažetog uzorkovanja. Naime, u nekoliko začetnih radova (dobar pregled dan je u [16]) dokazano je da se signali koji su dovoljno rijetki mogu rekonstruirati iz malog broja mjerenja (manjeg od Nyquist-ove granice, koja je odavno poznata u području obrade signala) rješavanjem problema konveksne optimizacije. Drugim riječima, dovoljno rijedak signal je jedinstveno rješenje linearnog sustava (pri čemu matrica sustava ima više stupaca nego redaka, odn. problem je pododređen) s ograničenjem rijetkosti. To rješenje se može dobiti brzim algoritmima optimizacije. Dovoljno rijetki signali se mogu prikazati i kao jedinstveno rješenje nekonveksnih problema, uz još manji broj mjerenja. Mnogi algoritmi nekonveksne optimizacije u praksi se pokazuju korisnima za rješavanje ovih problema. Također, u praksi je neophodno da algoritam bude prilagođen formulaciji problema koja uzima u obzir ne-egzaktnost modela, odn. grešku. Izvor te greške mogu biti i približna rijetkost i greška modela, odn. aditivni šum. Navedeni rezultati, iako motivirani u području sažetog uzorkovanja, vrlo su korisni i za rješavanje inverznih problema u obradi signala i slike.

Iako su se mnoge poznate, fiksne, transformacije (diskretna kosinusna transformacija, valići) pokazale korisnima u gore navedenim problemima, pokazano je da se bolji rezultati mogu dobiti učenjem transformacije, odn. rječnika, na specijalnoj klasi signala. U mnogim publikacijama prezentirani su poboljšani rezultati dobiveni korištenjem naučenih rječnika u odnosu na fiksne. U ovom radu, prezentirat ćemo korištenje metode analize nezavisnih komponenata (ANK) za učenje rječnika u nekim problemima obrade slike (rekonstrukcija nedostajućih dijelova, uklanjanje impulsnog šuma). Korištenje ANK ima biološko opravdanje. Naime, pokazano je da atomi naučeni pomoću ANK imaju mnogo sličnosti s eksperimentalno opaženim 'receptivnim poljima' (više detalja dano je u Odjeljku 0.3) u mozgu nekih sisavaca.

U nastavku navodimo glavne doprinose ovog rada:

1. ANK, koja je po sebi metoda za rješavanje problema razdvajanja signala, korištena je kao metoda za učenje rječnika za rijetku reprezentaciju slika prirodnih scena. Naučeni rječnik primijenjen je na problemu rekonstrukcije nedostajućih dijelova slike i uklanjanja specijalne vrste impulsnog šuma. Dobiveni rezultati su bolji od ili usporedivi s najsuvremenijim metodama u slučaju uniformne raspodjele nedostajućih elemenata u slici ili nekih strukturiranih raspodjela (linije, tekst).

2. Problem uklanjanja specijalne vrste impulsnog šuma formuliran je kao problem rekonstrukcije nedostajućih dijelova. Nakon toga, primijenjen je pristup opisan u točki (1). Dobiveni rezultati su značajno bolji od onih dobivenih najsuvremenijim metodama nelinearnog filtriranja (myriad filtri, modificirani medijan filtri). Predložene metode za uklanjanje specijalne vrste impulsnog šuma i rekonstrukcije nedostajućih dijelova u slici su efikasne i za sive i za slike u boji (RGB). Metode su opisane u radovima [34, 35].

3. Predložena je nova metoda za izdvajanje značajki u bioinformatici (proteomici i genomici). Bazirana je na novom tipu linearnog modela miješanja s referentnim uzorkom. Kroz matričnu faktorizaciju s ograničenjem rijetkosti, omogućeno je automatsko izdvajanje značajki na nivou svakog uzorka. Pri tome, labele uzoraka se ne koriste. Ova činjenica omogućava korištenje izdvojenih značajki za učenje klasifikatora, dok postojeće metode koje koriste faktorizaciju matrica koriste cijeli skup uzoraka za izvlačenje značajki i pri tome koriste labele uzoraka. Zbog toga se izdvojene značajke ne mogu koristiti za učenje klasifikatora. Predložena metoda je opisana u radu [60].

Ovdje dajemo i pregled sadržaja rada po poglavljima.

U Odjeljku 0.2 dan je pregled metoda za određeno i pododređeno razdvajanje signala. Naglasak je na teoriji i metodama za ANK baziranim na teoriji informacija. Sav materijal u ovom poglavlju je dobro poznat i preuzet iz literature.

U Odjeljku 0.3 opisana je motivacija za korištenje ANK kao metode za učenje rječnika za rijetku reprezentaciju.

U Odjeljku 0.4 dan je kratak pregled algoritama za rekonstrukciju signala uz ograničenje rijetkosti. Opisana su tri bitna pristupa, s posebnim naglaskom na algoritam koji koristi glatku aproksimaciju ℓ_0 kvazi-norme, koja je jedna od mjera rijetkosti. Također, naglasak je na varijantama algoritama prilagođenim za formulaciju modela koja uzima o obzir grešku (ili zbog približne rijetkosti, ili aditivnog šuma). Ovaj odjeljak je bitan jer se navedeni algoritmi koriste u eksperimentima opisanim u Odjeljku 0.7.

U Odjeljku 0.5 opisana su dva važna predstavnika metoda za učenje rječnika za rijetke reprezentacije signala/slika, osim ANK. Opisane metode su korištene u usporedbama opisanim u Odjeljku 0.7.

Odjeljak 0.6 predstavlja pregled metoda za nelinearno filtriranje (myriad i medijan filtri). Ove metode se uobičajeno koriste za uklanjanje impulsnog šuma u slici. U Odjeljku 0.7 demonstrirana je njihova inferiornost prema predloženom pristupu.

U Odjeljku 0.7 izloženi su rezultati usporedbi predložene i kompetitivnih metoda na problemima rekonstrukcije nedostajućih dijelova u slici (ponovno, korištene su uniformne i neke strukturirane raspodjele nedostajućih elemenata u slici). Ukratko su opisani i razlozi neuspješnosti metoda nelinearnog filtriranja za uklanjanje specijalne vrste impulsnog šuma. Također, u ovom poglavlju opisana je i primjena nove metode za izdvajanje značajki u proteomici i genomici. Metoda je primijenjena na realnim, javno dostupnim uzorcima, i uspješno uspoređena s kompetitivnim metodama.

Zaključak je dan u Odjeljku 0.8.

0.2 Linearno slijepo razdvajanje signala

Općeniti model *linearnog bezmemorijskog slijepog razdavanja signala* je oblika

$$\mathbf{X} = \mathbf{A}\mathbf{S}, \tag{0.2.1}$$

gdje je $\mathbf{X} \in \mathbb{R}^{M\times T}$ matrica čiji retci predstavljaju *miješane signale*, $\mathbf{A} \in \mathbb{R}^{M\times N}$ je *matrica miješanja*, a retci matrice $\mathbf{S} \in \mathbb{R}^{N\times T}$ predstavljaju *izvorne signale*. T označava veličinu uzorka. Zbog neizbježnih grešaka, bilo zbog grešaka mjerenja, slučajnog šuma ili ne-egzaktnosti modela, praktičniji je oblik

$$\mathbf{X} = \mathbf{A}\mathbf{S} + \mathbf{E}, \tag{0.2.2}$$

gdje je $\mathbf{E} \in \mathbb{R}^{M \times T}$ greška.

Model (0.2.1) se često interpretira na sljedeći način. Svaki stupac matrice $\mathbf{S}$ je *realizacija* slučajnog vektora $\mathbf{s}$, pa je prema tome i svaki stupac matrice $\mathbf{X}$ realizacija slučajnog vektora

$$\mathbf{x} = \mathbf{A}\mathbf{s}. \tag{0.2.3}$$

Razlikujemo slučajeve $N \leq M$ (*određeni* slučaj) i $N > M$ (*pododređeni* slučaj). Razmotrimo prvo lakši slučaj, $N \leq M$. Problem rekonstrukcije nepoznatog (slučajnog) vektora izvornih signala $\mathbf{s}$, čije su komponente međusobno *statistički nezavisne*, *samo* iz poznatog vektora miješanih signala $\mathbf{x}$ naziva se analiza nezavisnih komponenata (ANK). Formalna definicija je sljedeća [21].

Definicija 0.1. Analiza nezavisnih komponenata slučajnog vektora $\mathbf{x} \in \mathbb{R}^M$ je uređeni par $(\mathbf{F}, \Lambda)$ matrica takvih da vrijedi:

1. kovarijacijska matrica $\mathbf{C_x} = \mathbb{E}\left((\mathbf{x} - \mathbb{E}(\mathbf{x}))(\mathbf{x} - \mathbb{E}(\mathbf{x}))^T\right)$ vektora $\mathbf{x}$ može se faktorizirati u obliku $\mathbf{C_x} = \mathbf{F}\Lambda^2\mathbf{F}^T$, gdje je Λ dijagonalna i pozitivna, a $\mathbf{F}$ je $M \times \rho$ matrica punog ranga po stupcima, i $\rho \leq M$;

2. $\mathbf{x}$ se može zapisati kao $\mathbf{x} = \mathbf{F}\mathbf{z}$, gdje je $\mathbf{z}$ $\rho \times 1$ slučajni vektor s kovarijacijskom matricom Λ^2, i komponente od $\mathbf{z}$ su maksimalno statistički nezavisne u smislu maksimizacije dane *kontrastne funkcije*.

ANK je određena do na skaliranje i permutaciju. Uobičajeno se uzima da stupci matrice $\mathbf{F}$ iz gornje definicije imaju jediničnu normu, a dijagonalni elementi od Λ su silazno sortirani.

Analiza glavnih komponenata (AGK) se definira slično kao ANK. Naime, kažemo da je par $(\mathbf{F}, \Lambda)$ AGK vektora $\mathbf{x}$ ako je $\mathbf{C_x} = \mathbf{F}\Lambda^2\mathbf{F}^T$, gdje je Λ dijagonalna i pozitivna, a stupci od $\mathbf{F}$ su ortogonalni. Ova definicija povlači da vektor $\mathbf{y} = \mathbf{F}^T\mathbf{x}$ ima nekorelirane komponente (kovarijacijska matrica od $\mathbf{y}$ je dijagonalna). AGK se često koristi u svrhu *standardizacije*. Cilj standardizacije je tranformacija danog slučajnog vektora $\mathbf{x}$ u slučajni vektor $\mathbf{z}$ s jediničnom kovarijacijskom matricom. Standardizacija je čest prvi korak u računanju ANK.

Definicija 0.1 zahtijeva odabir mjere statističke nezavisnosti. Često korištene mjere nezavisnosti su međusobna informacija i negentropija. *Međusobna informacija* slučajnog vektora $\mathbf{x}$, $I(\mathbf{x})$, definirana je s

$$I(\mathbf{x}) = D_{\mathrm{KL}}\left(p_{\mathbf{x}}, \prod_{i=1}^{M} p_{x_i}\right),$$

gdje je $p_{\mathbf{x}}(\cdot)$ funkcija gustoće slučajnog vektora $\mathbf{x}$, a $D_{\mathrm{KL}}(\cdot)$ označava Kullback-Leiblerovu divergenciju, definiranu s $D_{\mathrm{KL}}\left(p_{\mathbf{x}}, \prod_{i=1}^{M} p_{x_i}\right) = \int p_{\mathbf{x}}(\mathbf{x}) \log \frac{p_x(\mathbf{x})}{\prod_{i=1}^{M} p_{x_i}(x_i)} d\mathbf{x}$. Vrijedi $I(\mathbf{x}) \geq 0$ i $I(\mathbf{x}) = 0$ ako i samo ako su komponente od $\mathbf{x}$ međusobno statistički nezavisne. Međusobna informacija može se izraziti i pomoću entropije. Naime, iz definicije međusobne informacije slijedi $I(\mathbf{x}) = \sum_i H(x_i) - H(\mathbf{x})$, gdje $H(\mathbf{x})$ označava entropiju slučajnog vektora $\mathbf{x}$, definiranu s $H(\mathbf{x}) = -\int p_{\mathbf{x}}(\mathbf{x}) \log p_{\mathbf{x}}(\mathbf{x}) d\mathbf{x}$. *Negentropija* slučajnog vektora $\mathbf{x}$, u oznaci $J(\mathbf{x})$, definirana je s

$$J(\mathbf{x}) = H(\mathbf{x}_{\text{Gauss}}) - H(\mathbf{x}),$$

gdje $\mathbf{x}_{\text{Gauss}}$ označava normalni slučajni vektor s istim očekivanjem i kovarijacijskom matricom kao $\mathbf{x}$. Negentropija ima svojstvo nenegativnosti, kao i poništavanja za normalno distribuirane slučajne vektore. Veza međusobne informacije i negentropije izražena je s

$$I(\mathbf{x}) = J(\mathbf{x}) - \sum_i J(x_i) + \frac{1}{2} \log \frac{\prod_i \Sigma_{ii}}{\det \Sigma},$$

gdje je $\Sigma = \mathbf{C}_{\mathbf{x}}$. Za standardizirani $\mathbf{x}$, krajnji desni član u gornjem izrazu je jednak nuli.

Nakon uvođenja gornjih nekoliko definicija, možemo definirati neke kontrastne funkcije iz Definicije 0.1. Kontrastna funkcija je, po definiciji, preslikavanje Ψ sa skupa vjerojatnosnih funkcija gustoće u $\mathbb{R}$ koje zadovoljava:

1. $\Psi(p_{\mathbf{Px}}) = \Psi(p_{\mathbf{x}})$, za svaku permutaciju $\mathbf{P}$;
2. $\Psi(p_{\Gamma\mathbf{x}}) = \Psi(p_{\mathbf{x}})$, za svaku invertibilnu dijagonalnu matricu Γ;
3. ako $\mathbf{x}$ ima nezavisne komponente, onda je $\Psi(p_{\mathbf{Ax}}) \leq \Psi(p_{\mathbf{x}})$, za svaku invertibilnu matricu $\mathbf{A}$.

Kažemo da je kontrastna funkcija (*kontrast*) Ψ diskriminirajuća ako jednakost u točki (3) vrijedi samo ako je $\mathbf{A}$ oblika $\mathbf{A} = \Gamma\mathbf{P}$, gdje je Γ invertibilna dijagonalna, a $\mathbf{P}$ permutacijska matrica. Diskriminirajuća kontrastna funkcija osigurava jedinstvenost ANK do na skaliranje i permutaciju. Može se pokazati [21] da je funkcija

$$\Psi(p_{\mathbf{x}}) = -I(\mathbf{z})$$

kontrastna funkcija za ICA, gdje je $\mathbf{z}$ standardizirani vektor pridružen vektoru $\mathbf{x}$. Ψ je i diskriminirajuća na skupu slučajnih vektora s najviše jednom normalno distribuiranom komponentom. Prema tome, međusobna informacija (ili preciznije, *negativna* međusobna informacija) je dobar izbor za kontrast u definiciji ANK.

U praksi, međusobnu informaciju je potrebno aproksimirati. U tu svrhu koriste se aproksimacije entropije. Jedan način za aproksimiranje entropije je korištenjem statistika višeg reda. Glavni nedostatak ovog pristupa je osjetljivost statistika višeg reda (kao što su kumulanti) na greške u podacima. Naime, ako su u uzorku iz kojeg se kumulanti aproksimiraju prisutni podaci koji značajno odskaču od ostalih, oni mogu potpuno odrediti procjene kumulanata, što te procjene čini neupotrebljivima u praksi.

Iz navedenih razloga, često se koriste neke druge aproksimacije entropije. U [52] pokazano je da se entropija slučajne varijable x može aproksimirati kao

$$H(x) \approx H(\nu) - \mathbb{E}\left(G(x)\right), \tag{0.2.4}$$

gdje je ν standardna normalna varijabla, a G neka funkcija. G se može odabrati ovisno o pretpostavljenoj vjerojatnosnoj razdiobi od x. Često su korištene funkcije

$$G_1(x) = \frac{1}{a_1} \log\cosh\left(a_1 x\right) \tag{0.2.5}$$

i

$$G_2(x) = -\frac{1}{a_2} \exp\left(-\frac{a_2 x^2}{2}\right), \tag{0.2.6}$$

koje pretpostavljaju *sub-Gaussovske* (približno rijetke) razdiobe dane slučajne varijable. Gornja aproksimacija entropije (0.2.4) povlači aproksimaciju negentropije oblika $J(x) \approx \frac{1}{2}\mathbb{E}\left(G(x)\right)$. FastICA ('brza ANK') [53] algoritam polazi od ove aproksimacije. U *deflacijskom* načinu optimizacije procjenjuje se jedan po jedan redak $\mathbf{w}_{G,k}$ inverzne matrice miješenja $\mathbf{W} = \mathbf{A}^{-1}$. Dakle, u svakom koraku se rješava problem oblika

$$\begin{array}{rcl} \mathbf{w}_{G,k} & = & \arg\max_{\mathbb{E}\left\{(\mathbf{w}^T\mathbf{x})^2\right\}=1} \mathbb{E}\left(G\left(\mathbf{w}^\mathbf{T}\mathbf{x}\right)\right) \\ & & \text{uz uvjet} \quad \mathbf{w}^T\mathbf{w}_{G,i} = 0,\ i = 1,\ldots,k-1 \end{array}. \tag{0.2.7}$$

Gornji uvjet ortogonalnosti pretpostavlja da je vektor $\mathbf{x}$ *standardiziran*. Očekivanje se procjenjuje iz danog uzorka, odn. skupa realizacija slučajnog vektora $\mathbf{x}$, a to je po našoj pretpostavci matrica $\mathbf{X}$. Slična je formulacija i za *simultani* način optimizacije, gdje se svi retci matrice $\mathbf{W}$ procjenjuju odjednom. FastICA algoritam koristi aproksimativnu Newtonovu metodu za rješavanje problema (0.2.7), što ga općenito čini vrlo brzim. `MATLAB` kod FastICA algoritma je javno dostupan, dobro dokumentiran i jednostavan za korištenje. Uz brzinu, ovo su razlozi iz kojih je FastICA algoritam korišten u eksperimentima opisanim u Odjeljku 0.7.

Teži slučaj razdvajanja signala javlja se kad je $N > M$. U ovom slučaju, statistička nezavisnost općenito nije dovoljno jak uvjet da osigura jedinstvenost vektora $\mathbf{s}$, odn. matrice realizacija $\mathbf{S}$, uz dani $\mathbf{x}$, respektivno matricu $\mathbf{X}$. Ovdje ulazi u igru *rijetkost*. Naime, ukoliko već sami izvorni signali nisu (približno) rijetki, često postoji transformacija Ψ takva da su transformirani signali, odn. retci matrice $\mathbf{S}\Psi$ (približno) rijetki. U tom slučaju, iz modela $\mathbf{X} = \mathbf{AS}$ slijedi $\mathbf{X}\Psi = \mathbf{AS}\Psi$, pa se problem svodi na *razdvajanje rijetkih signala*. Jedna od metoda rješavanja ovog problema je pristup koji se sastoji od dva koraka: prvo se aproksimira matrica miješanja $\mathbf{A}$, a zatim izvorni signali uz poznatu aproksimaciju matrice miješanja $\hat{\mathbf{A}}$. Uočimo da se iz dobivene aproksimacije $\hat{\mathbf{S}}\Psi$ transformirane matrice izvornih signala $\mathbf{S}\Psi$, aproksimacija $\hat{\mathbf{S}}$ dobiva invertiranjem transformacije Ψ.

Mnoge metode za aproksimaciju matrice miješanja pretpostavljaju da postoje stupci od $\mathbf{S}$ ili $\mathbf{S}\Psi$ u kojima je samo jedna komponenta dominantna. U slučaju kompleksnih signala, to povlači (približnu) kolinearnost realnog i imaginarnog dijelova danog stupca. Na ovoj činjenici se zasniva jedan kriterij nalaženja takvih stupaca (s jednom dominantnom komponentom). Metoda koja koristi ovaj kriterij korištena je u eksperimentima opisanim u Odjeljku 0.7. Uz poznatu matricu miješanja, izvorni signali se, ako su dovoljno rijetki, mogu dobiti kao jedinstveno rješenje optimizacijskih problema s ograničenjem rijetkosti. Pri tome se kao osnovna mjera rijetkosti koristi ℓ_0 funkcija, u oznaci $\|\cdot\|_0$, koja je definirana kao broj komponenata različitih od nule u danom vektoru. Osnovni rezultat je sljedeći [16].

Propozicija 0.1. *Ako linearni sustav* $\mathbf{x} = \mathbf{A}\mathbf{s}$*, pri čemu je* $\mathbf{A} \in \mathbb{R}^{M\times N}$ *i* $N > M$*, ima rješenje* $\mathbf{s}$ *koje zadovoljava* $\|\mathbf{s}\|_0 = |\{i : s_i \neq 0\}| < \frac{spark(\mathbf{A})}{2}$*, gdje* $spark(\mathbf{A})$ *označava broj elemenata najmanjeg linearno zavisnog podskupa skupa stupaca od* $\mathbf{A}$*, onda je* $\mathbf{s}$ *jedinstveno najrjeđe rješenje sustava.*

Optimizacijski algoritmi za nalaženje rijetkih rješenja opisani su u Odjeljku 0.4.

U literaturi postoje i rezultati vezani uz rješivost problema analize *nezavisnih* komponenata u slučaju $N > M$. Ali, ono što oni garantiraju je samo jedinstvenost *distribucije* vektora $\mathbf{s}$. Čini se da u literaturi nije predloženo mnogo algoritama za pod-određenu ANK. Kao aproksimativna metoda može se koristiti modifikacija FastICA algoritma. Naime, umjesto inzistiranja na ortogonalnosti (vidi (0.2.7)), može se zahtijevati samo približna ortogonalnost matrice $\mathbf{W}$ (ovdje je bitno da je vektor $\mathbf{x}$ standardiziran). Ovaj pristup korišten je u Odjeljku 0.7 za učenje *redundantnog* rječnika (s više atoma od dimenzije prostora) za rijetku reprezentaciju slika prirodnih scena.

0.3 Linearan model rijetkog kodiranja

Pretpostavimo sada da su izvorni signali u modelu (0.2.3) i statistički nezavisni, i rijetki u smislu da su im distribucije *rijetke*. Rijetke distribucije su neformalno definirane kao distribucije s visokim vrhom u nuli (višim od normalne razdiobe) i teškim repovima. Dakle, pretpostavljamo nezavisnost komponenata, ali također zahtijevamo da distribucije komponenata pripadaju točno određenoj klasi distribucija (rijetkima). Ova ideja je bitna u kontekstu učenja rječnika za rijetku reprezentaciju. Naime, ako imamo uzorak signala iz neke klase, smješten u matricu $\mathbf{X}$ (po stupcima), onda navedenu ideju možemo primijeniti korištenjem ANK na matrici $\mathbf{X}$ (koju smatramo realizacijom slučajnog vektora $\mathbf{x}$), uz *zadane distribucije* nezavisnih komponenata. Distribucije komponenata, ili barem njihova svojstva, se u FastICA algoritmu mogu implicitno zadati kroz funkciju G (vidi (0.2.7) u prethodnom odjeljku). Ukoliko $\mathbf{X}$ približno zadovoljava linearan model $\mathbf{X} = \mathbf{A}\mathbf{S}$ s nezavisnim i rijetkim komponentama, problem je dobro postavljen. Više detalja o učenju rječnika pomoću ANK dano je u Odjeljku 0.5. Dakle, *ANK se može koristiti kao metoda za učenje rječnika za rijetku reprezentaciju signala.*

Gore opisana ideja je bitna za rješavanje *inverznih problema* u obradi signala i slike. Općeniti inverzni problem se može zapisati kao

$$\mathbf{y} = \mathbf{H}\mathbf{x} + \mathbf{n}, \tag{0.3.1}$$

gdje je $\mathbf{y} \in \mathbb{R}^m$ vektor opažanja, $\mathbf{H} \in \mathbb{R}^{m\times M}$, $M \geq m$, predstavlja operator degradacije, $\mathbf{x} \in \mathbb{R}^M$ je originalni signal kojeg želimo rekonstruirati, a $\mathbf{n} \in \mathbb{R}^m$ predstavlja grešku. Problem (0.3.1) je općenito loše postavljen zbog ne-invertibilnosti operatora $\mathbf{H}$ i zbog greške $\mathbf{n}$. Pretpostavka da $\mathbf{x}$ ima rijetku reprezentaciju u danom rječniku $\mathbf{A} \in \mathbb{R}^{M\times N}$ može regularizirati ovaj problem. Naime, ako pretpostavimo da vrijedi $\mathbf{x} = \mathbf{A}\mathbf{s} + \mathbf{e}$, gdje je $\mathbf{s} \in \mathbb{R}^N$ rijedak, imamo

$$\mathbf{y} = \mathbf{H}\mathbf{A}\mathbf{s} + \tilde{\mathbf{e}},$$

gdje je $\tilde{\mathbf{e}} = \mathbf{H}\mathbf{e} + \mathbf{n}$ greška. Dakle, ako je $\mathbf{s}$ dovoljno rijedak (vidi prethodni odjeljak), $\mathbf{x}$ se može dobiti rijetkošću ograničenom optimizacijom iz opažanja $\mathbf{y}$.

Korištenje ANK za učenje rječnika za rijetku reprezentaciju slika prirodnih scena ima i biološko opravdanje. Naime, u ranim radovima [84, 10] pokazano je da se učenjem rječnika za rijetku reprezentaciju slika prirodnih scena (preciznije, malih komadića slika prirodnih scena), respektivno korištenjem ANK uz implicitno zadavanje rijetkih distribucija na izvorne signale, dobivaju vrlo slični bazni vektori (atomi), bez obzira na različitost metoda kojima su dobiveni. Navedeni bazni vektori, nakon matricizacije, sliče eksperimentalno opaženim receptivnim poljima u dijelu mozga nekih sisavaca zaduženom za procesiranje vizualnih informacija. To je u skladu s pretpostavkom efikasnog procesiranja informacija [90], jer bazni vektori predstavljaju upravo nezavisne komponente čijim se kombiniranjem mogu prikazati signali iz dane klase. U našem slučaju, to su slike prirodnih scena. U navedenom radu [10] korišten je infomax algoritam [9] za ANK, ali može se koristiti i bilo koji drugi algoritam za ANK. U Odjeljku 0.7 ovog rada korišten je FastICA.

0.4 Algoritmi za rješavanje pododređenih linearnih sustava s ograničenjem rijetkosti

U Odjeljku 0.2 vidjeli smo da se razdvajanje rijetkih signala nakon procjene matrice miješanja svodi na rješavanje problema oblika

$$\min_{\mathbf{s}} \|\mathbf{s}\|_0 \quad \text{uz uvjet} \quad \mathbf{x} = \mathbf{A}\mathbf{s}, \tag{0.4.1}$$

ili

$$\min_{\mathbf{s}} \|\mathbf{s}\|_0 \quad \text{uz uvjet} \quad \|\mathbf{x} - \mathbf{A}\mathbf{s}\|_2 \leq \varepsilon, \tag{0.4.2}$$

gdje je $\mathbf{A} \in \mathbb{R}^{M\times N}$, a ε označava dozvoljenu pogrešku aproksimacije. Nekoliko je glavnih pristupa rješavanju ovih problema. Direktne metode, kao što su pohlepne metode (Orthogonal

Matching Pursuit (OMP) [104] i varijante OMP-a) ili IHT [11], su namijenjene gornjim formulacijama problema. Kod indirektnih metoda, diskretna funkcija $\|\cdot\|_0$ se zamjenjuje neprekidnom ili glatkom aproksimacijom, što olakšava optimizaciju. ℓ_1 norma je jedan predstavnik neprekidnih, a ujedno i *konveksnih*, aproksimacija ℓ_0 funkcije. Ipak, u praksi se bolji rezultati mogu dobiti korištenjem preciznijih aproksimacija ℓ_0 funkcije, kao što je

$$F_\sigma(\mathbf{x}) = \sum_i \left(1 - \exp\left(-\frac{x_i^2}{2\sigma^2}\right)\right),$$

gdje je $\sigma > 0$ parametar kojim se kontrolira glatkoća aproksimacije. Ovaj pristup predložen je u [79].

Uvjeti pod kojima navedeni algoritmi uspijevaju naći najrjeđe rješenje sustava se najčešće izražavaju pomoću koherencije ili tzv. RIP (Restricted Isometry Property) [18] svojstva matrice **A**. Koherencija je definirana kao maksimalna (po apsolutnoj vrijednosti) korelacija između različitih stupaca matrice. Za **A** kažemo da ima RIP svojstvo reda k s konstantom $\delta_k \in (0,1)$ ako za svaki $\mathbf{s} \in \mathbb{R}^N$ takav da je $\|\mathbf{s}\|_0 \leq k$ vrijedi

$$(1-\delta_k)\,\|\mathbf{s}\|_2^2 \leq \|\mathbf{A}\mathbf{s}\|_2^2 \leq (1+\delta_k)\,\|\mathbf{s}\|_2^2.$$

Koherencija se lako računa, ali uvjeti za rješivost pododređenih sustava izraženi pomoću koherencije su previše pesimistični. S druge strane, RIP svojstvo daje oštre uvjete rješivosti *i stabilnosti*, ali se (jako) teško računa pa nije praktično. Zato ovdje navodimo samo rezultate koji koriste koherenciju.

OMP algoritam

OMP algoritam u svakom koraku traži *lokalno najbolju* aproksimaciju rijetkog rješenja. OMP uobičajeno kreće od aproksimacije $\hat{\mathbf{s}} = \mathbf{0}$. Trenutni ostatak (*rezidual*) $\mathbf{r}$ definiran je kao $\mathbf{r} = \mathbf{x} - \mathbf{A}\hat{\mathbf{s}}$. U svakoj sljedećoj iteraciji određuje se novi indeks stupca matrice **A** koji je maksimalno koreliran s trenutnim ostatkom:

$$i^* \in \arg\max_i \left|\left(\mathbf{A}^T\mathbf{r}\right)_i\right|.$$

Indeks i^* dodajemo u skup indeksa $\hat{I}$, i sljedeća aproksimacija rješenja se dobiva rješavanjem problema najmanjih kvadrata

$$\hat{\mathbf{s}}_{\hat{I}} = \arg\min_{\mathbf{s}} \left\|\mathbf{x} - \mathbf{A}_{\hat{I}}\mathbf{s}\right\|_2^2,$$

gdje $\hat{\mathbf{s}}_{\hat{I}}$ označava vektor elemenata od $\hat{\mathbf{s}}$ s indeksima u $\hat{I}$, a $\mathbf{A}_{\hat{I}}$ označava podmatricu od **A** koja se sastoji od stupaca matrice **A** s indeksima u $\hat{I}$. Postupak staje ili kad norma trenutnog ostatka padne ispod nekog zadanog praga ili kada broj elemenata u $\hat{I}$ dostigne zadanu gornju granicu. U literaturi su predložene i razne varijante gore opisanog OMP algoritma, koje su ili brže ili imaju bolje teorijske garancije. Primjeri uključuju [12, 81, 13], a neke od ostalih referenci mogu se naći u [16, 105]. Ovdje se koncentriramo na klasični OMP algoritam. Osnovni rezultat je sljedeći.

Teorem 0.1. *Pretpostavimo da je $\mathbf{s}^*$ rješenje problema (0.4.1), i definirajmo $I = \{i : s_i^* \neq 0\}$. Tada, dovoljan uvjet da OMP algoritam nađe rješenje $\mathbf{s}^*$ je*

$$\max_{i \in \{1,\dots,N\} \setminus I} \left\| \mathbf{A}_I^\dagger \mathbf{a}_i \right\|_1 < 1,$$

gdje $\mathbf{a}_i$ označava i-ti stupac od $\mathbf{A}$, a $\mathbf{A}_I^\dagger$ je pseudoinverz podmatrice od $\mathbf{A}$ sa stupcima indeksiranim u I.

Dokaz ovog rezultata može se naći u [103]. Dovoljan uvjet da OMP nađe stvarno rješenje može se izraziti i pomoću koherencije $\mu(\mathbf{A})$ od $\mathbf{A}$ kao (dokaz u [16])

$$\|\mathbf{s}^*\|_0 < \frac{1}{2}\left(1 + \frac{1}{\mu(\mathbf{A})}\right).$$

U slučaju problema (0.4.2) vrijedi sljedeći rezultat.

Teorem 0.2. *Pretpostavimo da rješenje $\mathbf{s}^*$ problema (0.4.2) zadovoljava*

$$\|\mathbf{s}^*\|_0 \leq \frac{1+\mu(\mathbf{A})}{2\mu(\mathbf{A})} - \frac{1}{\mu(\mathbf{A})} \frac{\varepsilon^*}{\min_{i \in I} |s_i^*|},$$

gdje je I definiran kao u Teoremu 0.1, a $\varepsilon^* = \|\mathbf{x} - \mathbf{A}\mathbf{s}^*\|_2$. Označimo s $\hat{\mathbf{s}}_{\varepsilon^*}$ aproksimaciju rješenja dobivenu OMP algoritmom, uz zaustavljanje kad je norma ostatka $\leq \varepsilon^*$. Tada vrijedi:

1. *$supp(\hat{\mathbf{s}}_{\varepsilon^*}) = supp(\mathbf{s}^*)$;*
2. *$\|\hat{\mathbf{s}}_{\varepsilon^*} - \mathbf{s}^*\|_2^2 \leq \frac{(\varepsilon^*)^2}{1-\mu(\mathbf{A})(\|\mathbf{s}^*\|_0 - 1)}$.*

Računska složenost OMP algoritma je reda $O(|I|MN)$. OMP je jedan od najčešće korištenih algoritama za rješavanje problema (0.4.1) i (0.4.2).

ℓ_1 minimizacija

Drugi, indirektan, pristup rješavanju (0.4.1) i (0.4.2) je zamjena ℓ_0 funkcije neprekidnom i *konveksnom* aproksimacijom, ℓ_1 normom, što rezultira problemima oblika

$$\min_{\mathbf{s}} \|\mathbf{s}\|_1 \quad \text{uz uvjet} \quad \mathbf{x} = \mathbf{A}\mathbf{s} \tag{0.4.3}$$

i

$$\min_{\mathbf{s}} \|\mathbf{s}\|_1 \quad \text{uz uvjet} \quad \|\mathbf{x} - \mathbf{A}\mathbf{s}\|_2 \leq \varepsilon. \tag{0.4.4}$$

Oba gornja problema se svode na poznate probleme konveksne optimizacije: linearno, odn. kvadratno programiranje, što je i glavni razlog za korištenje ℓ_1 norme. Glavno je pitanje pod kojim se uvjetima rješenja problema (0.4.1) i (0.4.3), odn. (0.4.2) i (0.4.4), podudaraju. Vrijedi sljedeći rezultat [16].

Teorem 0.3. *Ako rješenje* $\mathbf{s}^*$ *problema (0.4.1) zadovoljava*

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

onda je $\mathbf{s}^*$ *jedinstveno rješenje* oba *problema (0.4.1) i (0.4.3).*

Vidimo da je uvjet u gornjem teoremu isti kao i za OMP algoritam (Teorem 0.1). U slučaju problema (0.4.4), vrijedi sljedeći teorem [24].

Teorem 0.4. *Ako rješenje* $\mathbf{s}^*$ *problema (0.4.2) zadovoljava*

$$\|\mathbf{s}^*\|_0 < \frac{1}{4}\left(1 + \frac{1}{\mu(\mathbf{A})}\right),$$

onda rješenje $\hat{\mathbf{s}}$ *problema*

$$\min_{\mathbf{s}} \|\mathbf{s}\|_1 \quad \textit{uz uvjet} \quad \|\mathbf{x} - \mathbf{A}\mathbf{s}\|_2 \leq \delta,$$

gdje je $\delta \geq \varepsilon$*, zadovoljava*

$$\|\hat{\mathbf{s}} - \mathbf{s}^*\|_2^2 \leq \frac{(\varepsilon + \delta)^2}{1 - \mu(\mathbf{A})\,(4\|\mathbf{s}^*\|_0 - 1)}.$$

Ni OMP niti ℓ_1 minimizacija (formulacije (0.4.3) i (0.4.4)) nisu uniformno bolji od drugog. U nekim situacijama OMP može dati ispravan, a ℓ_1 minimizacija krivi rezultat, i obratno. Naprimjer, pokazuje se da ℓ_1 minimizacija općenito češće daje ispravan rezultat kad su sve ne-nul (različite od nule) komponente rijetkog rješenja iste (ili slične) apsolutne vrijednosti [74].

SL0 metoda

ℓ_1 norma je gruba aproksimacija ℓ_0 funkcije. Zato se, u praksi, bolji rezultati mogu dobiti korištenjem preciznijih aproksimacija. U [79] predloženo je korištenje aproksimacije

$$\begin{aligned} \|\mathbf{x}\|_0 \quad &\approx \quad F_\sigma(\mathbf{x}) = \sum_i (1 - f_\sigma(x_i)) = \\ &= \sum_i \left(1 - \exp\left(-\frac{x_i^2}{2\sigma^2}\right)\right), \end{aligned} \tag{0.4.5}$$

gdje je $\sigma > 0$ parametar koji kontrolira glatkoću aproksimacije. $F_\sigma(\cdot)$ može proizvoljno dobro aproksimirati ℓ_0 funkciju. Međutim, za veći σ aproksimacija je glađa, pa ima i manje lokalnih minimuma (na danom skupu definiranom ograničenjima), a za σ vrlo blizu nuli ima mnogo lokalnih minimuma na danom skupu. Zato je u [79] predložen sljedeći postupak: $F_\sigma(\cdot)$ se minimizira za opadajući niz vrijednosti parametra σ, nadajući se da dobiveni niz točaka minimuma konvergira k rješenju $\mathbf{s}^*$ originalnog problema (0.4.1) (odn. (0.4.2)).

Osnovni teorijski rezultati o konvergenciji gore predloženog algoritma su sljedeći. Pretpostavljat ćemo da vrijedi $\text{spark}(\mathbf{A}) = M + 1$ (odn. svaki skup od M različitih stupaca od $\mathbf{A}$ je linearno nezavisan). Sljedeći teorem [79] vrijedi i za općenitije funkcije od F_σ.

Teorem 0.5. *Označimo $\mathscr{S} = \{\mathbf{s} : \mathbf{As} = \mathbf{x}\}$. Pretpostavimo da familija funkcija f_σ, indeksirana parametrom $\sigma \in \mathbb{R}_+$, zadovoljava:*

1. $\lim_{\sigma \downarrow 0} f_\sigma(s) = 0$*, za sve* $s \neq 0$*;*

2. $f_\sigma(0) = 1$*, za sve* $\sigma \in \mathbb{R}_+$*;*

3. $0 \leq f_\sigma(s) \leq 1$*, za sve* $\sigma \in \mathbb{R}_+$ *i* $s \in \mathbb{R}$*;*

4. *za sve* $\nu > 0$ *i* $\alpha > 0$ *postoji* $\sigma_0 \in \mathbb{R}_+$ *takav da vrijedi*

$$|s| > \alpha \Rightarrow f_\sigma(s) < \nu, \text{ za sve } \sigma < \sigma_0.$$

Označimo sa $\mathbf{s}^\sigma$ točku globalnog minimuma od F_σ na $\mathscr{S}$ (pri čemu je F_σ definirana pomoću f_σ kao u (0.4)). Vrijedi:

$$\lim_{\sigma \downarrow 0} \mathbf{s}^\sigma = \mathbf{s}^0.$$

Važna pretpostavka u gornjem teoremu bila je da se za dani σ može naći *globalni* minimum of F_σ na $\mathscr{S}$. Pokazuje se da je u praksi to često slučaj (vidi sljedeći odjeljak).

U slučaju prisutnosti greške u modelu, imamo sljedeći rezultat [79].

Teorem 0.6. *Označimo $\mathscr{S}_\varepsilon = \{\mathbf{s} : \|\mathbf{As} - \mathbf{x}\|_2 \leq \varepsilon\}$, za $\varepsilon > 0$. Pretpostavimo da $\mathbf{A}$ i f_σ zadovoljavaju uvjete iz Teorema 0.5. Neka je $\mathbf{s^0} \in \mathscr{S}_\varepsilon$ rijetko rješenje takvo da je $k = \left\|\mathbf{s^0}\right\|_0 < \frac{M}{2}$. Pretpostavimo sljedeće dodatne uvjete na f_σ:*

1. *postoji* $\gamma > 0$ *takav da je* $\left|f_\sigma'(s)\right| < \frac{\gamma}{\sigma}$*, za sve* $\sigma > 0$ *i za sve* s*;*

2. *za sve* $\nu > 0$ *i* $\sigma_0 > 0$ *postoji* $\alpha > 0$ *takav da vrijedi:*

$$|s| > \alpha \Rightarrow f_\sigma(s) < \nu, \text{ za sve } \sigma < \sigma_0.$$

Definirajmo $\sigma_0 = \frac{N\gamma\varepsilon\left\|\mathbf{A}^\dagger\right\|_2}{M-2k}$. Tada, optimizirajući F_{σ_0}, rijetko rješenje $\mathbf{s^0}$ se može aproksimirati s greškom manjom od

$$(M_\mathbf{A} + 1)(N\alpha + \varepsilon),$$

gdje je $M_\mathbf{A} = \max\left\{\left\|\mathbf{A}_I^{-1}\right\|_2; |I| \leq M\right\}$, a α je takav da uvjet (2) vrijedi za σ_0 i $\nu = \frac{1}{N}$.

Navedeni teorijski rezultati ilustrirani su primjerom u sljedećem odjeljku.

Usporedba metoda

Ovdje navodimo jedan citat iz rada [74]:

> Princip empirijske provjere ima veliko značenje za područje rijetkih reprezentacija i sažetog uzorkovanja. U mnogim ključnim radovima u području diskutiraju se rigorozni teorijski rezultati, dobiveni korištenjem matematičke analize. Potrebna je prava matematička zrelost da bi se razumjelo što se tvrdi, koja je interpretacija dokazanih teorema, i kako usporediti rezultate u različitim radovima. Često je slučaj da su dokazane tvrdnje nejasne (koriste neodređene konstante) ili vrlo slabe (korištene su nerealno jake pretpostavke, koje gotovo nikad nisu ispunjene u stvarnim primjenama). Za praktične inženjerske primjene važnije je znati što se stvarno događa nego što se može dokazati. Empirijske studije i usporedbe predstavljaju direktnu metodu koja inženjerima daje korisne informacije o tome što se u praksi zaista događa.

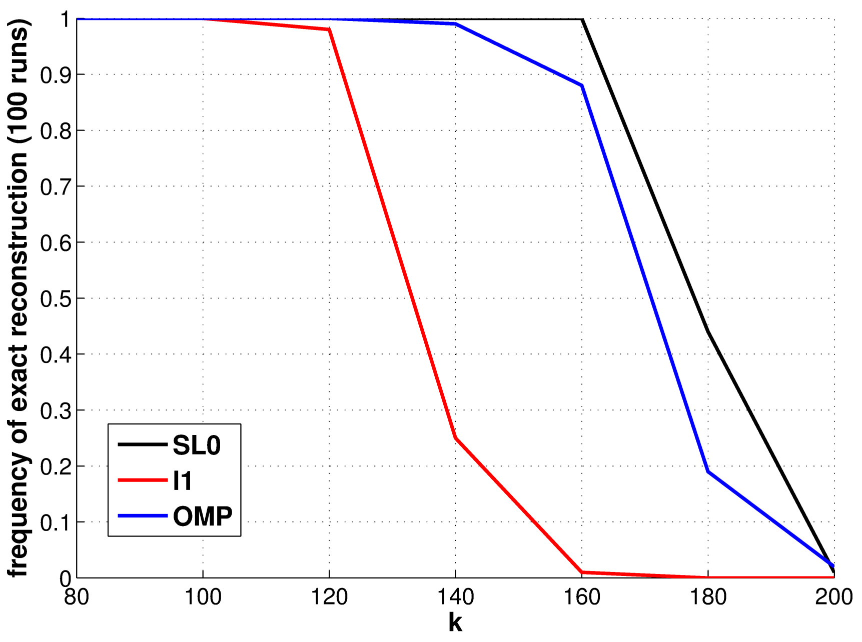
Gore navedene tvrdnje, izrečene od strane priznatih stručnjaka u (ili bliskih) području rijetkih reprezentacija, lijepo sažimaju ideju empirijske provjere algoritama, za razliku od rigoroznih teorijskih rezultata navedenih u prethodnim odjeljcima.

Graf na Slici 0.4.1 dobiven je na sljedeći način. Neka je $M = 400, N = 1000$. Matricu $\mathbf{A} \in \mathbb{R}^{M \times N}$ generiramo slučajno (naredba $\texttt{randn}\,(M,N)$ u `MATLAB`-u). Stupce od $\mathbf{A}$ normaliziramo. ℓ_0 funkcija rijetkog rješenja $\mathbf{s}^* \in \mathbb{R}^N$ ($k = \|\mathbf{s}^*\|_0$) varirana je u skupu $[80, 100, 120, 140, 160, 180, 200]$. k indeksa ne-nul elemenata od $\mathbf{s}^*$ generirano je slučajno iz $\{1, \dots, N\}$. Ne-nul elementi od $\mathbf{s}^*$ generirani su iz $\mathcal{N}(0,1)$. Vektor mjerenja $\mathbf{x}$ dobiven je kao $\mathbf{x} = \mathbf{A}\mathbf{s}^*$. Na ovaj način generirano je 100 sustava s rijetkim rješenjem *za svaki k* i *za svaki algoritam*. Smatramo da je algoritam našao rješenje ako aproksimacija $\hat{\mathbf{s}}$ dobivena danim algoritmom zadovoljava $\|\hat{\mathbf{s}} - \mathbf{s}^*\|_2 < 10^{-5}$. Slika 0.4.1 pokazuje udio slučajeva u kojima je algoritam vratio točno rješenje u ovom smislu, za svaku vrijednost od k. Pokazuje se da SL0 metoda daje točno rješenje *s najvećom vjerojatnošću* (u najvećem broju slučajeva). Sličan se rezultat dobiva i ako se mjerenjima $\mathbf{x}$ doda šum.

SL0 algoritam korišten je u eksperimentima opisanim u Odjeljku 0.7. OMP algoritam korišten je kao dio K-SVD algoritma za učenje rječnika za rijetke reprezentacije (vidi sljedeći odjeljak).

0.5 Algoritmi za učenje rječnika za rijetke reprezentacije

U Odjeljku 0.2 diskutirane su metode za rješavanje pododređenog problema razdvajanja signala. Navedene metode su bazirane na pretpostavci da su izvorni signali ili rijetki ili da postoji transformacija koja ih čini rijetkima. U Odjeljku 0.3 opisano je korištenje rječnika za rijetku reprezentaciju signala u rješavanju inverznih problema u obradi signala i slike. Ovdje opisujemo



Slika 0.4.1: Usporedba opisanih algoritama na sintetički generiranim primjerima (vidi tekst).

nekoliko poznatih i često korištenih metoda za *učenje* rječnika za rijetke reprezentacije *dane klase signala*. Pri tome, naglasak je na korištenju ANK za učenje rječnika za rijetke reprezentacije, iako je ANK sama po sebi metoda za rješavanje problema razdvajanja signala.

Pretpostavimo da je dana matrica podataka $\mathbf{X} \in \mathbb{R}^{n \times T}$, pri čemu stupci od $\mathbf{X}$ čine reprezentativni *trening skup* za danu klasu signala. Trening skup se sastoji od tipičnih signala iz dane klase. Općenita formulacija problema učenja rječnika za rijetke reprezentacije je sljedeća:

$$\arg\min_{\mathbf{D},\mathbf{C}} \|\mathbf{X}-\mathbf{DC}\|_F^2 \quad \text{uz uvjet} \quad \|\mathbf{c}_i\|_0 \le K, \tag{0.5.1}$$

gdje $\mathbf{c}_i$ označava i-ti stupac *matrice koeficijenata* $\mathbf{C}$, a K je ograda na rijetkost reprezentacije. Uobičajeni način rješavanja gornjeg problema je alternirajuća minimizacija: prvo, $\mathbf{D}$ fiksiramo pa minimiziramo po $\mathbf{C}$ (ovaj postupak se uobičajeno naziva *rijetko kodiranje*), a nakon toga slijedi minimizacija po $\mathbf{D}$ za fiksni $\mathbf{C}$. K-SVD algoritam, koji opisujemo u sljedećem odjeljku, je takvog oblika.

K-SVD algoritam

K-SVD algoritam je generalizacija K-means algoritma [38] za grupiranje podataka u slučaju kad je broj grupa veći od 1. Izraz koji se minimizira u (0.5.1) može se zapisati kao

$$\|\mathbf{X}-\mathbf{DC}\|_F^2 = \left\| \left(\mathbf{X} - \sum_{j\neq k} \mathbf{d}_j \left(\mathbf{c}^j\right)^T\right) - \mathbf{d}_k \left(\mathbf{c}^k\right)^T \right\|_F^2,$$

pri čemu $\mathbf{d}_j$ označava j-ti stupac od $\mathbf{D}$, a $\left(\mathbf{c}^j\right)^T$ j-ti redak od $\mathbf{C}$. Označimo s ω_k skup indeksa stupaca od $\mathbf{X}$ koji koriste stupac $\mathbf{d}_k$ u trenutnoj reprezentaciji, odn. one indekse $i \in \{1,\dots,T\}$ za koje je $\mathbf{c}_i^k \neq 0$. S Ω_k označimo $T \times |\omega_k|$ matricu s jedinicama na indeksima $(\omega_k(i), i)$ i nulama na preostalim indeksima (ovdje smo s $\omega_k(i)$ označili i-ti indeks u skupu ω_k po veličini, od najmanjeg prema najvećem). Ako označimo $\mathbf{E}_k = \mathbf{X} - \sum_{j \neq k} \mathbf{d}_j \left(\mathbf{c}^j\right)^T$, ideja K-SVD algoritma je minimizacija od $\left\|\mathbf{E}_k \Omega_k - \mathbf{d}_k \left(\mathbf{c}^k\right)^T \Omega_k\right\|_F^2$ po $\mathbf{d}_k$ i $\mathbf{c}^k$, što se postiže krnjim SVD rastavom. Naime, na ovaj način nova vrijednost od $\mathbf{c}^k$ zadržava (ili smanjuje) skup elemenata različitih od nule. Ukoliko bi rijetko kodiranje bilo egzaktno (u smislu da je greška $\mathbf{X} - \mathbf{DC}$ za dane $\mathbf{D}$ i $\mathbf{C}$ jednaka nuli), imali bi sigurnu konvergenciju prema lokalnom optimumu (ili, moguće, sedlastoj točki) problema (0.5.1). Budući da je u praksi dobivena reprezentacija vrlo rijetko egzaktna, nema garancije za konvergenciju. Ipak, čini se da u praksi algoritam gotovo uvijek konvergira (k lokalnom optimumu). Uobičajeno korišten algoritam za rijetko kodiranje (optimizacija po $\mathbf{C}$ za fiksni $\mathbf{D}$) je OMP. Naime, OMP po svojo strukturi lako nalazi aproksimaciju rijetkog rješenja s danim brojem komponenata različitih od nule. Također, vrlo je brz.

Korištenje ANK za učenje rječnika za rijetke reprezentacije

Kao što je već napomenuto, ANK je metoda za rješavanje problema razdvajanja nezavisnih signala. Ali, može se indirektno koristiti i za učenje rječnika za rijetke reprezentacije signala. Naime, implicitno se može zadati distribucija nepoznatih nezavisnih komponenata (izvornih signala). U FastICA algoritmu, opisanom u Odjeljku 0.2, to se postiže odabirom nelinearne funkcije korištene u aproksimaciji negentropije (vidi opis u Odjeljku 0.2). Naravno, ograničenja u formulaciji (0.5.1) općenito ne mogu biti zadovoljena korištenjem ANK. Ali, ANK možemo koristiti kao *aproksimativnu* metodu za rješavanje (0.5.1). Biološko opravdanje korištenja ANK za učenje rijetkih reprezentacija slika prirodnih scena diskutirano je u Odjeljku 0.3.

Interpretacija modela $\mathbf{X} = \mathbf{DC}$ u ovom slučaju je sljedeća. Budući da se u ovom radu bavimo primjenom ANK u nekim problemima obrade slika, pretpostavljamo da se trening skup (stupci od $\mathbf{X}$) sastoji od vektoriziranih *komadića slika prirodnih scena*. Naime, svaki stupac $\mathbf{x_i} \in \mathbb{R}^n$ predstavlja vektorizirani dio $\tilde{\mathbf{X}}_i \in \mathbb{R}^{\sqrt{n} \times \sqrt{n}}$ slike. U kontekstu ANK, svaki redak matrice podataka $\mathbf{X}$ predstavlja miješani signal. Retke od $\mathbf{C}$ interpretiramo kao nezavisne komponente. Stupci matrice (rječnika) $\mathbf{D}$ su vektori miješanja. Kroz odabir prikladne nelinearne funkcije u FastICA algoritmu, implicitno pretpostavljamo *rijetkost* nezavisnih komponenata, odn. redaka matrice $\mathbf{C}$. Rijetkost redaka od $\mathbf{C}$ povlači da su i *stupci* od $\mathbf{C}$ rijetki, s velikom vjerojatnošću. Drugim riječima, ograničenje u formulaciji (0.5.1) je približno zadovoljeno. Dakle, svaki stupac od $\mathbf{X}$ može se aproksimirati linearnom kombinacijom malog broja vektora miješanja (stupaca matrice $\mathbf{D}$). Prema tome, matrica miješanja $\mathbf{D}$ odgovara *rječniku* za rijetku reprezentaciju stupaca od $\mathbf{X}$. Na ovaj način, rječnik za rijetku reprezentaciju dobiva se kao nusprodukt ANK.

U Odjeljku 0.7, u svrhu učenja rječnika pomoću ANK, korišten je FastICA algoritam, uz implicitno forsiranje rijetkosti nezavisnh komponenata korištenjem funkcija (0.2.5) i (0.2.6).

Polja eksperata

U ovom odjeljku $\mathbf{X}$ će nam označavati sliku, odn. matricu čiji su elementi elementi slike (pikseli). U radu [92] predložen je sljedeći pristup modeliranju slike. Sliku $\mathbf{X} \in \mathbb{R}^{m \times n}$ modeliramo kao graf $G = (V, E)$, gdje elementi matrice (slike) $\mathbf{X}$, X_{ij}, čine skup vrhova V. X_{ij} interpretiramo kao *slučajne varijable*. Skup bridova E definiran je pomoću zadane strukture na slici. Osnovna pretpostavka metode predložene u [92] je da X_{ij} čine *Markovljevo slučajno polje* (MSP). Neformalno, to znači da je svaki X_{ij} uvjetno nezavisan s $X_{i'j'}$, $(i', j') \neq (i, j)$, uz dane *susjedne elemente* od X_{ij}. Specijalnu klasu MSP-a čine MSP-i čija se vjerojatnosna funkcija gustoće može faktorizirati po svim *potpunim podgrafovima* (potpuni podgraf grafa je podgraf u kojem su svaka dva vrha povezana, odn. postoji brid koji ih povezuje). U spomenutom radu [92], korištena je upravo ova klasa MSP-a za modeliranje povezanosti elemenata slike. Za više detalja vidi [92] ili Poglavlje 5 u punoj verziji ovog rada. Kao spomenuti potpuni podgrafovi korišteni su svi kvadratni komadići slike. Susjedni elementi iz gornje definicije MSP-a, za dani X_{ij}, su preostali elementi potpunog podgrafa koji X_{ij} definira (a to je komadić slike u kojem je X_{ij} centralni element). Parametri modela optimizirani su prema danom trening skupu slika prirodnih scena.

Ovaj pristup modeliranju slike se značajno razlikuje od pristupa korištenjem rječnika za rijetku reprezentaciju, ali je tipičan predstavnik vjerojatnosnih pristupa modeliranju slike. Korišten je u usporedbi metoda u Odjeljku 0.7.

0.6 Metode za uklanjanje impulsnog šuma u slici

Uklanjanje šuma je jedan od fundamentalnih problema u obradi slike. Uobičajeno je pretpostaviti da je šum Gaussov ili barem da ima konačnu (malu) varijancu. Napomenimo da ovdje mislimo na *aditivni* šum. U praksi je, ipak, nerijetko slučaj da je šum *impulsni*, odn. da ima veliku ili čak beskonačnu varijancu. U ovom odjeljku dajemo kratak pregled osnovnih pristupa uklanjanju impulsnog šuma.

Označimo sa $\mathbf{s} \in \mathbb{R}^n$ vektorizirani oblik slike (slika je matrica čiji su elementi elementi slike (pikseli); slično, slika u boji je 3D tenzor). $\tilde{\mathbf{s}} = \mathbf{s} + \mathbf{n}$ označava sliku oštećenu šumom $\mathbf{n}$. U ovom odjeljku razmatramo samo sive slike. Označimo $d_{\min} = \min_i s_i$, $d_{\max} = \max_i s_i$. Dva značajna modela impulsnog šuma u slici su sljedeća:

- 'salt-and-pepper' šum:

$$\tilde{s}_i = \begin{cases} d_{\min} & \text{, s vjerojatnošću } \frac{p}{2} \\ d_{\max} & \text{, s vjerojatnošću } \frac{p}{2} \\ s_i & \text{, s vjerojatnošću } 1-p \end{cases},$$

  gdje p označava udio impulsnog šuma;

- impulsni šum sa slučajnim vrijednostima:

$$\tilde{s}_i = \begin{cases} d_i \text{, s vjerojatnošću } p \\ s_i \text{, s vjerojatnošću } 1-p \end{cases},$$

gdje je $d_i \sim \mathscr{U}\,(d_{\min}, d_{\max})$, a p označava udio šuma.

U eksperimentima u Odjeljku 0.7 razmatran je problem uklanjanja ‘salt-and-pepper’ šuma. Problem uklanjanja ove vrste šuma može se svesti na problem popunjavanja nedostajućih elemenata slike. Puno je teži slučaj ako impulsni šum ima slučajne vrijednosti. Time se u ovom radu ne bavimo.

Nelinearno filtriranje

Metode nelinearnog filtriranja uvedene su da bi se uzela u obzir ne-Gaussovska priroda mnogih signala koji se pojavljuju u praksi. Konkretan primjer je upravo uklanjanje *ne-Gaussovskog šuma* (najčešće, tu se misli na impulsni/super-Gaussovski šum). Klasični primjeri nelinearnih filtara u obradi signala i slike su *medijan* i *myriad filtri*. Glavna referenca je knjiga [5].

Medijan filtri

Medijan vektora $\mathbf{x} \in \mathbb{R}^N$ je definiran na sljedeći način. Označimo s $x_{[i]}$ i-ti element od $\mathbf{x}$ po veličini (dakle, vrijedi $x_{[1]} \leq x_{[2]} \leq \ldots \leq x_{[N]}$). Za neparni N, medijan od $\mathbf{x}$ je definiran kao $x_{[(N+1)/2]}$. Za parni N, medijan je definiran kao $\frac{1}{2}\left(x_{[N/2]} + x_{[N/2+1]}\right)$.
Definicija medijan filtra je sljedeća.

Definicija. Neka je $\mathbf{s} \in \mathbb{R}^N$ diskretan signal. Izlaz $\mathbf{y} \in \mathbb{R}^N$ medijan filtra s prozorom veličine $N_L + N_R + 1$, gdje je $N_L + N_R + 1 \leq N$, primijenjenog na $\mathbf{s}$, je definiran s

$$y_n = \text{medijan}\left(s_{n-N_L}, \ldots, s_n, \ldots, s_{n+N_R}\right),$$

gdje je $1 \leq n \leq N$. Za $n \leq N_L$ ili $n \geq N - N_R + 1$, y_n je definiran proširivanjem vrijednosti od $\mathbf{s}$ izvan rubova, prema danim *rubnim uvjetima*.

y_n iz gornje definicije se dobiva kao *procjenitelj maksimalne vjerodostojnosti (MV procjenitelj)* srednje vrijednosti skupa vrijednosti elemenata od $\mathbf{s}$ unutar prozora oko s_n, uz pretpostavku da elementi od $\mathbf{s}$ čine *slučajni uzorak iz Laplaceove distribucije*. Za usporedbu, uzoračka sredina se dobiva kao MV procjenitelj srednje vrijednosti, uz pretpostavku Gaussovosti uzorka. Prema tome, medijan filter zamjenjuje vrijednost danog elementa signala (na danom indeksu) robusnom procjenom srednje vrijednosti susjedstva danog elementa. Ideja ove procedure je postići otpornost na šum koji možda nije male varijance.

U slučaju 2D signala, odn. slika, definicija medijan filtra se jednostavno proširuje korištenjem 2D prozora. U tom slučaju, element slike se zamjenjuje MV procjeniteljem srednje vrijednosti

2D susjedstva elementa slike. Uobičajeni rubni uvjet u slučaju slika je *simetrični* rubni uvjet (vrijednosti izvan rubova se dobiju *zrcaljenjem* vrijednosti elemenata slike).

Težinski medijan filter se dobije množenjem svakog elementa susjedstva od s_i iz gornje definicije $(s_{n-N_L},\ldots,s_n,\ldots,s_{n+N_R})$ pripadnim težinskim faktorom. Statistička interpretacija je slična kao gore: izlaz težinskog medijana je MV procjenitelj srednje vrijednosti *otežanog* uzorka. U obradi slike često se koristi medijan filter s centralnom težinom, pri čemu je otežan samo centralni element uzorka (s_n iz gornje definicije), a težine ostalih uzoraka su jednake jedinici. Takvi medijan filtri korišteni su u eksperimentima opisanim u Odjeljku 0.7.

Medijan filtri su optimalni (u smislu gornje statističke interpretacije) u slučaju Laplaceove distribucije uzorka. U obradi slike, odn. uklanjanju šuma u slici, daju dobre rezultate ako distribucija šuma nije jako impulsna. Laplaceova distribucija je primjer takve distribucije jer je super-Gaussova, ali ima konačnu varijancu.

Bolji rezultati u uklanjanju impulsnog (konkretno, 'salt-and-pepper') šuma mogu se dobiti nekim modifikacijama medijan filtara. Neki primjeri dani su u Odjeljku 0.7. Ipak, u Odjeljku 0.7 pokazat ćemo da je pristup uklanjanju 'salt-and-pepper' šuma korištenjem rječnika za rijetke reprezentacije naučenog pomoću ANK puno uspješniji i od navedenih modifikacija medijan filtara.

Myriad filtri

Prije formalne definicije myriad filtra, razmotrimo opći oblik robusnog procjenitelja srednje vrijednosti distribucije (odnosno *lokacijskog parametra*, u slučaju da očekivanje nije konačno).

Definicija. Neka je $x_1,\ldots,x_N$ dani slučajni uzorak iz neke distribucije. M-procjenitelj srednje vrijednosti (lokacijskog parametra) distribucije, $\hat{\beta}$, *za danu funkciju* $\rho(\cdot)$, definiran je s

$$\hat{\beta} = \arg\min_{\beta} \sum_{i=1}^{N} \rho(x_i - \beta). \tag{0.6.1}$$

Za $\rho(x) = x^2$ dobiva se uobičajena uzoračka sredina, a za $\rho(x) = |x|$ medijan uzorka. Za $\rho(x) \sim -\log f(x)$, gdje je $f(\cdot)$ vjerojatnosna funkcija gustoće, $\hat{\beta}$ je MV procjenitelj srednje vrijednosti distribucije $f(\cdot)$. Nas zanimaju *impulsne distribucije*, odn. preciznije impulsne *α-stabilne* distribucije. Karakteristična funkcija α-stabilne distribucije je oblika $\phi(t) = \exp\left(-k|t|^{\alpha}\right)$, gdje je $k > 0$ *parametar raspršenosti* i $\alpha \in (0,2]$ *karakteristični eksponent*. Za α blizu nuli dobivaju se vrlo impulsne distribucije. Za $\alpha = 1$ dobiva se Cauchy-jeva distribucija, čija je vjerojatnosna funkcija gustoće

$$f(x) = \frac{k}{\pi}\frac{1}{k^2 + x^2}.$$

Cauchy-jeva distribucija nema konačnu varijancu i dobar je model *vrlo impulsne* distribucije. Sada navodimo definiciju myriad filtra.

Definicija. Za dani uzorak $x_1,\ldots,x_N$ i parametar raspršenosti $k > 0$, *uzorački myriad reda k* definiran je s

$$\hat{\beta}_k = \text{myriad}\{k; x_1,\ldots,x_N\} = \arg\min_{\beta} \sum_{i=1}^{N} \log\left[k^2 + (x_i - \beta)^2\right].$$

Dakle, uzorački myriad je robustna procjena lokacijskog parametra, uz pretpostavku da je uzorak iz Cauchy-jeve distribucije. *Myriad filter* definiran je po analogiji s medijan filtrom. Naime, myriad filer zamjenjuje vrijednost danog elementa signala (slike) uzoračkim myriad-om vrijednosti susjednih elemenata.

Vrijedi sljedeći teorem [43].

Teorem. *Označimo s $T_{\alpha,k}(x_1,\ldots,N)$ MV procjenitelj srednje vrijednosti (0.6.1), odnosno lokacijskog parametra, za danu funkciju ρ izvedenu iz α-stabilne distribucije s karakterističnim eksponentom α i parametrom raspršenosti k. Tada vrijedi*

$$\lim_{\alpha\downarrow 0} T_{\alpha,k}(x_1,\ldots,x_N) = myriad\{0; x_1,\ldots,x_N\}.$$

Dakle, uzorački myriad je optimalni procjenitelj lokacijskog parametra za vrlo impulsne (α blizu nuli) distribucije. Sličan rezultat vrijedi i za *težinski* uzorački myriad, definiran s

$$\hat{\beta}_{k,\mathbf{w}} = \arg\min_{\beta} \sum_{i=1}^{N} \log\left[k^2 + w_i(x_i - \beta)^2\right].$$

Vrlo je važan dobar izbor parametra k. On ovisi o udjelu šuma u podacima. Naime, vrijede sljedeće dvije tvrdnje [43]. Za $k \to \infty$, uzorački myriad je jednak uzoračkoj sredini. U tom slučaju implicitno je pretpostavljeno da su svi elementi u uzorku pouzdani, odn. niti jedan nije oštećen velikim vrijednostima impulsnog šuma. Za $k \to 0$, vrijedi sljedeće.

Propozicija. *Neka je $\mathscr{M}$ skup vrijednosti u uzorku $x_1,\ldots,x_N$ koje se najčešće ponavljaju. Tada je*

$$\hat{\beta}_0 = \arg\min_{x_j\in\mathscr{M}} \prod_{i=1,x_i\neq x_j}^{N} \left|x_i - x_j\right|.$$

U ovom slučaju najpouzdanije su vrijednosti koje se ponavljaju (udio šuma je velik). Dakle, kad je udio šuma velik, preporuka je da k bude reda $\sim \min_{i\neq j}\left|x_i - x_j\right|$.

Myriad filtri se pokazuju korisnima u uklanjanju impulsnog šuma u slici ako je udio šuma mali. U Odjeljku 0.7 demonstriramo da, kada je udio impulsnog (odn. 'salt-and-pepper') šuma velik, pristup koji koristi rječnik za rijetke reprezentacije naučen pomoću ANK daje značajno bolje rezultate od myriad filtara.

Ostale metode nelinearnog filtriranja

Vrlo efikasan pristup nelinarnom filtriranju predložen je u [99]. Krećemo od modela $\tilde{\mathbf{s}} = \mathbf{s} + \mathbf{n}$. s_i možemo označiti i s $\mathbf{s}(\mathbf{x}_i)$, gdje $\mathbf{x}_i$ označava poziciju elementa od $\mathbf{s}$. U slučaju slika (2D signala), $\mathbf{x}_i$ označava koordinate elementa slike. Aproksimacija $\hat{\mathbf{s}}$ od $\mathbf{s}$ dobiva se iz $\tilde{\mathbf{s}}$ kao

$$\hat{\mathbf{s}}\left(\mathbf{x}_j\right) = \arg\min_{\mathbf{s}(\mathbf{x}_j)} \sum_{i=1}^{N} \left[\tilde{s}_i - \mathbf{s}\left(\mathbf{x}_j\right)\right]^2 K\left(\mathbf{x}_i, \mathbf{x}_j, \tilde{s}_i, \tilde{s}_j\right),$$

gdje je K jezgra koja mjeri sličnost uzoraka $\tilde{s}_i$ i $\tilde{s}_j$ na pozicijama $\mathbf{x}_i$ i $\mathbf{x}_j$. Gornji oblik aproksimacije $\hat{\mathbf{s}}$ obuhvaća mnoge metode, ovisno o korištenoj jezgri K. U Odjeljku 0.7 dajemo usporedbu metode ovog tipa i predložene metode u ovom radu, koja koristi rječnik za rijetke reprezentacije naučen pomoću ANK, na primjeru rekonstrukcije nedostajućih dijelova slike. Pokazuje se da korištenje naučenog rječnika daje usporedive ili bolje rezultate.

Maximum-a-posteriori pristup uklanjanju impulsnog šuma

Prisjetimo se općeg oblika inverznih problema (0.3.1):

$$\mathbf{y} = \mathbf{H}\mathbf{x} + \mathbf{n},$$

gdje je $\mathbf{y} \in \mathbb{R}^m$ i $H \in \mathbb{R}^{m\times M}$, pri čemu je $M \geq m$. Ovdje nas zanima problem uklanjanja šuma, pa je $\mathbf{H} = \mathbf{I}$, gdje $\mathbf{I}$ označava jediničnu matricu. Pretpostavimo da je distribucija šuma Cauchyjeva. Želimo aproksimirati *čist* (neoštećen šumom) signal, odn. sliku, $\mathbf{x}$. *Maximum-a-posteriori* pristup procjeni nepoznatog čistog signala zasniva se na maksimizaciji *posteriorne vjerojatnosti* $p(\mathbf{x}\,|\,\mathbf{y})$. Prema Bayesovoj formuli, posteriorna funkcija gustoće vjerojatnosti je proporcionalna produktu funkcije vjerodostojnosti i apriorne funkcije gustoće vjerojatnosti nepoznatog parametra, odn. $p(\mathbf{x}\,|\,\mathbf{y}) \sim p(\mathbf{y}\,|\,\mathbf{x})\,p(\mathbf{x})$. Maksimizacija $p(\mathbf{x}\,|\,\mathbf{y})$ je ekvivalentna minimizaciji $-\log p(\mathbf{x}\,|\,\mathbf{y})$. Za $p(\mathbf{x})$ se uobičajeno pretpostavlja da je oblika $p(\mathbf{x}) \sim \exp(-\phi(\mathbf{x}))$, pri čemu je član $\phi(\mathbf{x})$ obično nekonveksan. Budući da je šum po pretpostavci Cauchy-jev, vrijedi $-\log p(\mathbf{y}\,|\,\mathbf{x}) \sim \sum_i \log\left(1 + (\mathbf{x}-\mathbf{y})_i^2\right)$. Prema tome, imamo

$$-\log p(\mathbf{x}\,|\,\mathbf{y}) \sim \sum_i \log\left(1 + (\mathbf{x}-\mathbf{y})_i^2\right) + \phi(\mathbf{x}).$$

Oba člana u gornjem izrazu su nekonveksna, što čini njegovu minimizaciju vrlo nepraktičnom. Zato ovaj pristup uklanjanju impulsnog šuma nije uobičajen.

0.7 Primjene

U ovom poglavlju prezentiramo rezultate usporedbi metoda za učenje rječnika za rijetke reprezentacije na problemima rekonstrukcije nedostajućih elemenata slike i uklanjanja 'salt-and-pepper' šuma. Nakon toga, opisujemo i jednu primjenu analize rijetkih komponenata za izdvajanje značajki u bioinformatici.

Rekonstrukcija nedostajućih elemenata slike

Kao trening skup korišteno je 6 slika prirodnih scena iz javno dostupne baze[1]. Iz svake slike izvučeno je 3000 kvadratnih, 16×16, komadića slike, koji su zatim spremljeni kao stupci u 256×18000 matricu podataka $\mathbf{X}$. Svakom stupcu od $\mathbf{X}$, odn. svakom komadiću slike, prije učenja rječnika nekim od algoritama, oduzeta je njegova srednja vrijednost. Nakon toga, primijenjeni su K-SVD i FastICA algoritmi za učenje rječnika. Korišteni parametri su sljedeći. Nelinearna funkcija koja je korištena u FastICA algoritmu je (0.2.5), uz $a_1 = 5$ (ovaj parametar odabran je empirijskom usporedbom rezultata za nekoliko različitih vrijednosti parametra). Za rijetko kodiranje u K-SVD-u korišten je OMP algoritam, uz $K = 40$ (vidi (0.5.1)). Ovaj izbor je malo usporio K-SVD algoritam (ukupno je 100 iteracija algoritma trajalo oko 5 sati). Učenje rječnika FastICA-om trajalo je oko 3 sata.

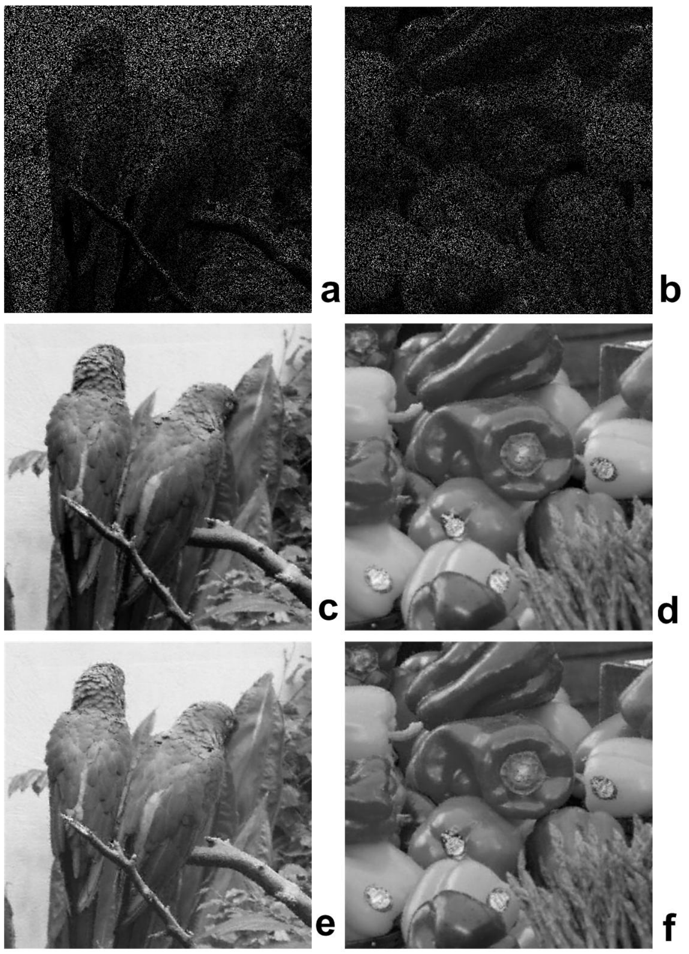
Metode su uspoređene na drugih 6 slika (testni skup), različitih od slika u trening skupu. Sve korištene slike su 8-bitne, odn. raspon vrijednosti elemenata je $[0, 255]$, gdje 0 označava crnu, a 255 bijelu boju. U svakoj slici je zadržano samo 20% elemenata (slučajnim odabirom), nakon čega su korišteni rječnici za rijetku reprezentaciju kako bi se dobila aproksimacija originalnih (neoštećenih) slika. Postupak je ponovljen 10 puta za svaku sliku. Na svakom 16×16 komadiću problem rekonstrukcije koji je rješavan je sljedeći:

$$\arg\min_{\mathbf{c}} \|\mathbf{c}\|_0 \text{ uz uvjet } \mathbf{MDc} = \tilde{\mathbf{x}} - \mathbb{E}(\tilde{\mathbf{x}}),$$

pri čemu $\mathbf{D}$ označava $256 \times N$ naučeni rječnik (K-SVD-om ili FastICA algoritmom), $N \geq 256$, $\mathbf{M}$ označava $m \times 256$ matricu projekcije na indekse elemenata u danom komadiću slike koji su poznati (neoštećeni), pri čemu je $m \approx 0.2 \cdot 256$, a $\tilde{\mathbf{x}}$ označava $m \times 1$ vektor poznatih elemenata u danom komadiću. $\mathbb{E}(\tilde{\mathbf{x}})$ označava srednju vrijednost vektora $\tilde{\mathbf{x}}$. Rekonstruirani komadić $\hat{\mathbf{x}}$ se dobiva kao $\hat{\mathbf{x}} = \mathbf{Dc} + \mathbb{E}(\tilde{\mathbf{x}})$. Komadići slike su odabrani tako da se susjedni komadići preklapaju u 2 stupca, odn. retka. Bolji rezultati se mogu dobiti ako se rekonstrukcija radi na *svim* kvadratnim komadićima slike, ali proces rekonstrukcije u tom je slučaju puno sporiji. Za rješavanje gornjeg problema, za svaki komadić slike, korišten je SL0 algoritam, opisan u Odjeljku 0.4. Naime, SL0 algoritam se pokazao najefikasnijim (u smislu kvalitete dobivenih rekonstrukcija), a i najbržim. U svim primjerima parametri SL0 algoritma optimizirani su empirijskom provjerom. Slika 0.7.1 pokazuje primjer dviju slika iz testnog skupa i njihovih rekonstrukcija korištenjem rječnika naučenih pomoću K-SVD i FastICA algoritama.

Rekonstrukcije dobivene korištenjem naučenih rječnika uspoređene su i s fiksnima. Konkretno, kao fiksni rječnici korištene su inverzna diskretna kosinusna transformacija i inverzna transformacija valićima. Za usporedbu, prikazujemo i rezultate dobivene metodom *analize morfoloških komponenata* (AMK) [29]. AMK koristi dva fiksna rječnika: jedna se koristi za efikasnu (rijetku) reprezentaciju po dijelovima konstantnog dijela slike, a druga za efikasnu reprezentaciju

[1] A. Olmos, F. A. A. Kingdom, McGill calibrated color image database, 2004., http://pirsquared.org/research/mcgilldb/



Slika 0.7.1: Dvije slike iz testnog skupa sa 80% nepoznatih (oštećenih) elemenata: a) i b). Rekonstrukcije dobivene korištenjem rječnika naučenog pomoću FastICA algoritma: c) i d). Rekonstrukcije dobivene korištenjem rječnika naučenog pomoću K-SVD algoritma: e) i f).

ostatka (‘teksture’). Budući da koristi fiksne rječnike, ova metoda očekivano daje slabije rezultate. Za usporedbu dajemo i rezultate dobivene metodom polja eksperata, ukatko opisanom u Odjeljku 0.5.

Dobiveni rezultati na svih 6 slika u testnom skupu prikazani su u Tablici 0.7.1. Ovdje ne prikazujemo sve slike iz testnog skupa radi prostora (vidi punu verziju rada). Vrijednosti u tablici su izražene vrijednostima *indeksa strukturne sličnosti (ISS)* [108, 107] dobivene rekonstrukcije i originalne slike. ISS postiže vrijednosti između -1 i 1, pri čemu veća vrijednost znači ‘bolju’ rekonstrukciju (ISS je jednak 1 ako se slike podudaraju). Vrijednosti u tablici odnose se na srednju vrijednost i standardnu devijaciju vrijednosti ISS indeksa nakon 10 ponavljanja (za različite distribucije oštećenih elemenata slike).

Metoda rekonstrukcije nedostajućih elemenata slika koja koristi naučeni rječnik može se koristiti i za neke strukturirane raspodjele nedostajućih elemenata. Naprimjer, vidi Sliku 0.7.2. Ipak, u slučaju kada na slici nedostaju veći dijelovi, kao naprimjer blokovi, ova metoda se ne može uspoređivati sa metodama specijalno dizajniranim za takve slučajeve, kao što su [46, 92].

Tablica 0.7.1: Rezultati rekonstrukcije u terminima ISS indeksa, pri čemu su naučeni rječnici kvadratni, 256 × 256, naučeni na 16 × 16 komadićima slika u trening skupu.

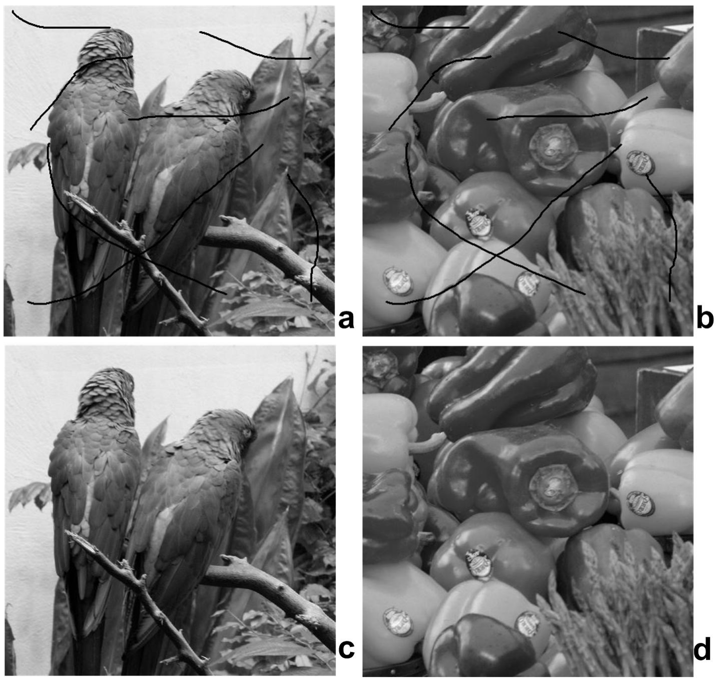
	ICA	**K-SVD**	**DCT**	**Symmlet 4 wavelet**	**MCA**	**FoE**
Sl. 1	0.907 ± 0.0008	0.905 ± 0.001	0.75 ± 0.0008	0.736 ± 0.0022	0.789	0.92
Sl. 2	0.76 ± 0.0016	0.749 ± 0.0012	0.55 ± 0.0015	0.503 ± 0.0015	0.682	0.77
Sl. 3	0.773 ± 0.0007	0.766 ± 0.0011	0.617 ± 0.0011	0.562 ± 0.003	0.644	0.78
Sl. 4	0.944 ± 0.0005	0.94 ± 0.0004	0.81 ± 0.0007	0.81 ± 0.0017	0.854	0.95
Sl. 5	0.6 ± 0.002	0.577 ± 0.0015	0.434 ± 0.0015	0.35 ± 0.0022	0.491	0.6
Sl. 6	0.919 ± 0.0006	0.917 ± 0.0005	0.84 ± 0.0003	0.812 ± 0.0009	0.852	0.92
Sredina	**0.817 ± 0.001**	**0.809 ± 0.0009**	**0.666 ± 0.0008**	**0.63 ± 0.002**	**0.719**	**0.824**

Opisana metoda se može koristiti i za slike u boji, kao što je opisano u [35]. Dobivaju se rezultati usporedivi sa specijaliziranim metodama, kao što je [73], ali jednostavnijim pristupom. Budući da je slika u boji 3D tenzor, u literaturi je predložen i pristup rekonstrukciji nedostajućih dijelova minimizacijom nuklearne norme tenzora, što je čest pristup rekonstrukciji tenzora. Međutim, taj pristup pretpostavlja da je tenzor niskog ranga, što je u slučaju općenitih slika u boji nerealna pretpostavka.

Predložena metoda se pokazuje uspješnom i u usporedbi s metodom opisanom u Odjeljku 0.6 i [99]. Ovdje ne ulazimo u sve detalje odabira parametara te metode. Za detalje vidi punu verziju rada ili [99]. Na primjerima dvije slike sa Slike 0.7.3, dobiveni rezultati su sljedeći. Ovdje predložena metoda primijenjena je na rekonstrukciju Slike 0.7.3a iz 30% poznatih elemenata, a dobiveni rezultat je, u terminima mjere PSNR (koja efektivno mjeri Frobeniusovu normu pogreške dobivene rekonstrukcije; vidi kraticu u Tablici 1.3.2), 31.89 dB (decibela; veća vrijednost obično znači rekonstrukciju bolje kvalitete, iako se upravo zbog boljeg slaganja s vizualnim dojmom ponekad koristi i navedeni ISS indeks). Metoda iz [99], uz vrijednosti parametara predložene od strane autora, na ovoj slici postiže 31.74 dB. Na Slici 0.7.3b ovdje predložena metoda postiže 29.59 dB, a metoda iz [99] 28.685 dB.

Uklanjanje 'salt-and-pepper' šuma

Problem uklanjanja 'salt-and-pepper' šuma (vidi Poglavlje 0.6) u slici može se svesti na problem rekonstrukcije nedostajućih elemenata slike. Naime, ideja je proglasiti sve elemente slike maksimalnog (255) ili minimalnog (0) intenziteta nepoznatima (oštećenima). Pri tome je mo-



Slika 0.7.2: Dvije slike iz testnog skupa s linijskom strukturom nedostajućih elemenata: a) i b). Rekonstrukcije dobivene rječnikom naučenim pomoću FastICA algoritma: c) i d).

Slika 0.7.3: Slike korištene za usporedbu s nekim metodama iz literature. a) Lena. b) Brod.

guće da se oštećenima proglase i elementi koji to možda nisu, ali zbog efikasnosti metode čak i u slučaju kad je značajan udio elemenata oštećen (u prethodnom odjeljku vidjeli smo primjere s 80% oštećenih elemenata) to ne utječe značajno na kvalitetu rezultata. U Odjeljku 0.6 vidjeli smo kratak pregled metoda nelinearnog filtriranja (medijan i myriad filtri), koje su u određenom smislu teorijski optimalne u danoj klasi metoda. Ovdje dajemo usporedbu s nekim modifikacijama osnovnih medijan filtara [32, 70], koje se pokazuju značajno uspješnijima od klasičnih medijan, *ali i* myriad filtara. Na primjeru sa Slike 0.7.3a, uz 70% oštećenih elemenata, metoda iz [32] postiže 24.3 dB, a metoda iz [70] 29.72 dB. U prethodnom odjeljku vidjeli smo da metoda predložena u ovom radu, bazirana na korištenju rječnika naučenog pomoću FastICA algoritma, postiže 31.89 dB, što je značajno bolje. Slična je razlika među metodama i na Slici 0.7.3b. Za više primjera vidi punu verziju rada. Vidimo da se filteri medijan i myriad tipa ne mogu uspoređivati s ovdje predloženom metodom na problemu uklanjanja 'salt-and-pepper' šuma.

Izdvajanje značajki u bioinformatici

Analiza podataka u bioinformatici se često bazira na linearnom modelu miješanja signala, pri čemu je broj izvornih signala nepoznat. Izvorni signali se mogu aproksimirati metodama razdvajanja signala. Neki od izdvojenih izvornih signala mogu se dalje koristiti za predikciju bolesti ili identifikaciju *biomarkera* (kemijskih tvari koje mogu služiti kao indikatori stanja bolesti). Ovdje razmatramo biološke uzorke proteinskih ili genskih profila. Proteinski profili predstavljaju spektre masa uzoraka, a genski profili izraženost svakog pojedinog gena u uzorku. Pojedinačni elementi profila (omjer masa/naboj u slučaju proteinskih, odn. geni u slučaju genskih profila) se uobičajeno nazivaju *značajke*.

U literaturi su predložene brojne metode izdvajanja značajki ili klasifikacije uzoraka. Osnovne karakteristike ovdje opisane metode, koja je predložena u [60], su sljedeće. Linearni model miješanja je formuliran posebno *za svaki uzorak*. Izdvojene komponente (izvorni signali) se automatski klasificiraju u komponentu tipičnu za stanje bolesti, u komponentu tipičnu za zdrav organizam i, moguće, jednu ili više komponenti koje sadrže značajke koje nisu posebno izražene ni u 'zdravoj' niti u 'bolesnoj' komponenti. Konkretno, za svaki *testni* uzorak $\mathbf{x} \in \mathbb{R}^n$ pretpostavljamo dva linearna modela

$$\begin{bmatrix} \mathbf{x}_\mathrm{z}^T \\ \mathbf{x}^T \end{bmatrix} = \mathbf{A}_\mathrm{z}\mathbf{S}_\mathrm{z} \tag{0.7.1}$$

i

$$\begin{bmatrix} \mathbf{x}_\mathrm{b}^T \\ \mathbf{x}^T \end{bmatrix} = \mathbf{A}_\mathrm{b}\mathbf{S}_\mathrm{b}, \tag{0.7.2}$$

gdje $\mathbf{x}_\mathrm{z}$ predstavlja referentni zdravi, $\mathbf{x}_\mathrm{b}$ referentni bolesni uzorak, a $\mathbf{A}_\mathrm{z} \in \mathbb{R}^{2\times M}$ i $\mathbf{A}_\mathrm{b} \in \mathbb{R}^{2\times M}$, $M \geq 2$, označavaju matrice miješanja, čije elemente interpretiramo kao relativne udjele koncentracije redaka od $\mathbf{S}_\mathrm{z}$, odn. $\mathbf{S}_\mathrm{b}$ (*komponenata*), u miješanim uzorcima $\mathbf{x}$, $\mathbf{x}_\mathrm{z}$ i $\mathbf{x}_\mathrm{b}$. Retke od $\mathbf{S}_\mathrm{z}$ i $\mathbf{S}_\mathrm{b}$ interpretiramo kao komponente koje sadrže značajke tipične za zdrav organizam, tipične za bolesno stanje ili značajke koje nisu posebno izražene ni u jednom niti u drugom slučaju.

Matrice miješanja u gornja dva modela mogu se aproksimirati metodom opisanom u Odjeljku 0.2. Uz pretpostavku da su u zdravim uzorcima dominante komponente koje sadrže značajke tipične za zdrav organizam, i analogno za bolesne uzorke, možemo pretpostaviti približnu rijetkost matrica $\mathbf{S}_\mathrm{z}$ i $\mathbf{S}_\mathrm{b}$. Prema tome, u [60] predložena je aproksimacija matrica $\mathbf{S}_\mathrm{z}$ i $\mathbf{S}_\mathrm{b}$, za dane aproksimacije matrica miješanja, minimizacijom ℓ_1 norme.

Komponente sa značajkama tipičnim za zdrav, odn. bolestan organizam, identificiraju se na osnovu kuta kojeg pripadni vektor miješanja (stupac matrice miješanja) zatvara s osima koordinatnog sustava. Promatrajmo model (0.7.1). Kao komponentu koja sadrži značajke tipične za zdrav organizam proglašavamo onu čiji pripadni vektor miješanja (odn. stupac matrice miješanja čiji je indeks jednak indeksu retka pripadne komponente) zatvara najmanji kut s x-osi. Za komponentu sa značajkama tipičnim za stanje bolesti proglašavamo onu koja zatvara najveći kut s x-osi. Analogan je postupak za model (0.7.2). Na ovaj način, za svaki uzorak dobivamo 4 komponente s pripadnim labelama (zdrava/bolesna) dobivenim na gore opisan način. Dakle, dobivamo 4 skupa izdvojenih komponenata, $\left\{\mathbf{s}^{\mathrm{z}}_{\text{zdrava ref.};i}, y_i\right\}_{i=1}^{N}$, $\left\{\mathbf{s}^{\mathrm{b}}_{\text{zdrava ref.};i}, y_i\right\}_{i=1}^{N}$, $\left\{\mathbf{s}^{\mathrm{b}}_{\text{bolesna ref.};i}, y_i\right\}_{i=1}^{N}$ i $\left\{\mathbf{s}^{\mathrm{z}}_{\text{bolesna ref.};i}, y_i\right\}_{i=1}^{N}$, gdje je N ukupan broj uzoraka. Na svakom od ova 4 skupa komponenata možemo trenirati klasifikator, i zadržati onaj s najvećom točnošću utvrđenom pomoću kros-validacije.

Gore opisana metoda uspoređena je s kompetitivnim metodama iz literature na 3 javno dostupna skupa uzoraka preuzeta sa[2]. Treba napomenuti da je u radu [60] korištena dvostruka kros-validacija, koja daje realnije procjene točnosti klasifikatora. Za razliku od toga, u većini kompetitivnih radova iz literature korištena je trostruka ili desetorostruka kros-validacija, koje daju preoptimistične rezultate. Dobiveni rezultati pokazuju da predložena metoda daje usporedivu (posebno kada se uzme u obzir spomenuti odnos dvostruke i višestruke kros-validacije) ili bolju točnost od kompetitivnih metoda, i to na sva 3 spomenuta skupa uzoraka.

0.8 Zaključak

Prvi doprinos ove radnje je korištenje analize nezavisnih komponenata za učenje rječnika za rijetke reprezentacije slika prirodnih scena, s primjenom u problemima popunjavanja nedostajućih dijelova slike i uklanjanja posebne vrste impulsnog šuma. ***Drugi doprinos*** je upravo formulacija problema uklanjanja posebne vrste impulsnog šuma kao problema popunjavanja nedostajućih dijelova slike. Korištenje analize nezavisnih komponenata za navedene probleme ima biološko opravdanje, kao što je opisano u Odjeljku 0.3.

Predložene metode su uspoređene s nekoliko reprezentativnih kompetitivnih metoda iz literature. Rezultati dobiveni predloženim metodama su usporedivi ili bolji od rezultata dobivenih kompetitivnim metodama. Na problemu uklanjanja posebne vrste impulsnog šuma, predložena metoda daje značajno bolje rezultate od uobičajeno korištenih metoda za ovaj problem (medijan i myriad filtri). Na problemu popunjavanja nedostajućih dijelova slike samo je metoda ‘polja eksperata’, bazirana na vjerojatnosnom modelu slike, opisana u Odjeljku 0.5, dala bolji rezultat, ali uz značajno veću računsku složenost. Opisane metode su korištene i na sivim i slikama u boji. Na problemu popunjavanja nedostajućih dijelova slike, najbolji rezultati se postižu za uniformnu raspodjelu nedostajućih elemenata slike. Ali, kao što je demonstrirano u Odjeljku 0.7, dobri rezultati se dobivaju i za neke strukturirane raspodjele, kao što su (tanke) linije ili tekst.

[2]Program proteomike američkog Nacionalnog instituta za rak, http://home.ccr.cancer.gov/ncifdaproteomics/ppatterns.asp

Predložene metode su opisane u radovima [34, 35]. Također, kod za reprodukciju opisanih rezultata je dostupan na stranici autora [3].

Treći doprinos ove radnje je nova metoda za izdvajanje značajki u bioinformatici. Bazirana je na linearnom modelu miješanja s referentnim uzorkom. Metoda omogućava automatsko izdvajanje značajki na nivou svakog uzorka, pri čemu se labele uzoraka ne koriste. Metoda je demonstrirana na nekoliko javno dostupnih skupova uzoraka, i dobiveni rezultati su usporedivi ili bolji od rezultata dobivenih kompetitivnim metodama iz literature. Metoda je opisana u radu [60].

[3]http://www.lair.irb.hr/ikopriva/marko-filipovi.html

Ključne riječi

Analiza nezavisnih komponenata, Razdvajanje signala, Rijetkost, Analiza rijetkih komponenata, Rijetka reprezentacija, Rekonstrukcija rijetkih signala, Pod-određeni linearni sustav, Učenje rječnika za rijetke reprezentacije, K-SVD algoritam, Nepotpuni podaci, Popunjavanje nedostajućih dijelova slike, Impulsni šum, Nelinearno filtriranje, Izdvajanje značajki, Linearan model miješanja, Bioinformatika

Chapter 1

Introduction

In recent years, the concept of sparsity has become ubiquitous in signal and image processing. Sparsity of a vector or a matrix means that many of its coefficients are zero (or close to zero in the case of *approximate sparsity*, or *compressibility*). Sparseness constraint is used to regularize many ill-posed inverse problems. Examples in imaging include inpainting, deblurring, denoising and super-resolution. It is known that many classes of signals can be compressed using transformations which result in many signal's coefficients being close to zero. This is known as the *analysis model* of sparsity. For example, JPEG still image compression standard uses sparsity of image coefficients in the wavelet domain. Digital audio encoding formats are based on the modified discrete cosine transform (MDCT). Another view on sparsity follows from the analysis model. Namely, if the transformation coefficients of a signal are mostly zero, this means that a signal can be represented as an inverse transformation of a sparse vector of coefficients. Therefore, it is often realistic to suppose that a signal can be represented as a linear combination of a small number of vectors from a pre-specified basis or a frame of the vector space (i.e., a signal has a *sparse representation* with respect to the basis or the frame). In signal processing, such a basis or a frame is usually referred to as the *dictionary*, and its elements, i.e. columns, are referred to as the dictionary vectors or *atoms*. This is known as the *synthesis model* of sparsity. It is the more often used model. Since many inverse problems, like image inpainting or impulse noise removal, are extremely ill-posed (because of the small number of available measurements), sparsity is a powerful tool for regularization. It has enabled state-of-the-art results in many image and signal processing problems.

Another area where sparsity proved to be fundamental is blind or semi-blind source separation (Chapter 2). Many algorithms proved to perform well in overdetermined (the number of mixtures greater than the number of sources; see Chapter 2) or determined case (the number of mixtures equal to the number of sources), but sparsity-based regularization enabled efficient solving of underdetermined (the number of mixtures less than the number of sources) source separation problems. Sparsity-constrained underdetermined source separation is known as sparse component analysis. In bioinformatics applications, like protein or gene expression analysis, sparsity enables selection of disease-specific features in proteomics (m/z ratios) and genomics (genes) that are used for disease prediction and biomarker identification. Many of these applications are based on several fundamental results in compressed sensing area. Basically, several seminal papers (some references are given in Chapter 4) have shown that sparse enough solutions of underdetermined linear systems of equations can be reconstructed by convex optimization methods under some regularity assumptions on the system matrix (in the terminology of compressed sensing, sparse signals can be reconstructed from fewer non-adaptive measurements than was previously known). Also, sparse solutions can in theory be reconstructed from even fewer measurements, but the resulting problems are non-convex. Nevertheless, there are many approximate algorithms available that perform well in practice. These results are theoretically well founded in approximation theory [37, 7]. Although fixed transforms, like wavelets, wavelet-like transforms or discrete cosine transform (DCT), have proven useful in the above mentioned applications, it was shown that better results can be obtained by *learning* transformations directly from the data of interest. Since fixed transforms are not adapted to a specific signal subclass of interest, for example natural images, it is logical that improved results can be obtained by exploiting specific signal structure. Many papers presented improved results using learned dictionaries compared to fixed ones.

One motivation for the use of independent component analysis (ICA, Section 2.2) in image restoration problems (like inpainting, removal of impulse noise and super-resolution) is the fact that it has been shown that the dictionary vectors learned by ICA have similarities with experimentally observed receptive fields of neurons in the brain of mammals (Section 3.1). Therefore, the use of ICA in this context has a biological justification. Although this is a known result, there haven't been, at least to the author's knowledge, results in the literature that used ICA in some realistic image restoration problems and compared results obtained with the ICA to the state-of-the-art methods. In this thesis, ICA was used for dictionary learning, and was extensively compared to state-of-the-art methods in *image inpainting* and *impulse noise removal* problems. This is the main contribution of this thesis (see Section 1.1). Independent component analysis is by itself a method for solving source separation problems. However, it can also be used for learning the dictionaries for sparse representations of signals. Namely, ICA decomposes mixed signals into mutually independent sources, with some assumptions on their probability distribution. In the context of dictionary learning for sparse representations, these a-priori unknown distributions can be chosen in such a way to enforce desired properties of source signals, such as sparsity. Sparsity of source signals implies sparsity of signal representations with respect to the inverse of the transformation learned by ICA.

Of course, signals that are encountered in applications are often not exactly sparse. Instead, their coefficients in the chosen dictionary decay slowly. This introduces error in the sparsity-constrained linear model. Namely, the contribution of small coefficients can be interpreted as an error in the linear model. Also, there is often a random noise with small variance that is present in the model. Therefore, to be useful in applications, algorithms for reconstruction of sparse signals from incomplete data need to be robust to model-caused additive errors. Robust algorithms for sparse reconstruction are reviewed in Chapter 4.

Before describing the main contributions of this thesis, we cite one paragraph from the introduction of the paper [74]:

> The empirical tuning approach has a larger significance for the field of sparse representations and compressed sensing. Many of the better known papers in this field discuss what can be proved rigorously, using mathematical analysis. It requires real mathematical maturity to understand what is being claimed and what the interpretation must be, and to compare claims in competing papers. Often, what can be proved is vague (with unspecified constants) or very weak (unrealistically strong conditions are assumed, far from what can be met in applications). For practical engineering applications it is important to know what really happens rather than what can be proved. Empirical studies provide a direct method to give engineers useful guidelines about what really does happen.

These few sentences, made by distinguished experts in or closely related to the field of sparse representations, are relevant for this thesis since the thesis deals with *practical* performance of proposed methods, and not their formal theoretical guarantees of performance. These words provide nice justification and explanation of the approach used in this thesis.

1.1 Contributions of the thesis

In the following we list the main contributions of this thesis:

1. Independent component analysis (ICA), which is a method for solving source separation problems, is used as a dictionary learning method for (approximately) sparse representation of signals. More concretely, approximately sparse representation of natural images' patches can be learned by applying ICA to a given training set of image patches. The learned dictionary is used for inpainting images, and also for removal of salt-and-pepper noise in natural images. The method compares favourably with state-of-the-art methods for unstructured (uniform distribution of missing pixels) inpainting. Also, the method performs well for some structured patterns of missing pixels (lines, text).

2. The problem of removal of salt-and-pepper noise in images is posed as an image inpainting problem, and solved by using the learned dictionary method. This concept compares favorably against state-of-the-art nonlinear filtering methods such as myriad and modified median filters.

The proposed methods for image inpainting and removal of salt-and-pepper noise work well both for grayscale and color images. They have been described in publications [34, 35].

3. A novel method for feature extraction in bioinformatics (more precisely, proteomics and genomics) is presented. It is based on a novel type of linear mixture model with a reference sample that, through sparsity constrained factorization, enables automatic feature extraction on a sample-by-sample basis. Therein, sample label information is not used. This allows the use of extracted features for training of the classifier. As opposed to that, existing matrix factorization methods use the whole dataset to extract disease specific features by using label information. This prevents the use of extracted features for training of the classifier.

   The method was described in the paper [60].

This thesis author's personal contributions in the above cited works are as follows. In [34], the author extracted relevant methods for comparison with the proposed method from the literature on dictionary learning and sparse recovery algorithms, and performed all numerical experiments. The idea to test the ICA for dictionary learning with aplication in image inpainting was suggested by the supervisor (I. Kopriva), as well as the interpretation of salt-and-pepper noise removal problem as the inpainting problem (by declaring all noise-corrupted pixels as missing). In [35], the author performed all experiments. The idea to compare the proposed method for color image inpainting with tensor completion methods was suggested by the supervisor (I. Kopriva). In [60], the author implemented the sparse recovery method and performed numerical experiments (cross-validation based component extraction and classification).

1.2 Chapter-by-chapter overview

We review methods for both (over)determined and underdetermined source separation in Chapter 2. The emphasis is on the information-theoretic approaches to independent component analysis: both the theory and algorithms. All the material presented in this chapter is well known and is taken from the literature.

In Chapter 3 a motivation for using the ICA for dictionary learning is described. This chapter is short but important, since it presents a background for the use of ICA in inpainting and removal of impulse noise problems.

We review some of the approximate algorithms for solving sparsity-constrained, generally nonconvex, inverse problems in Chapter 4. There, we also present and discuss several algorithms for sparse reconstruction that are robust to the presence of small errors. These algorithms are compared on simple synthetic data sets. Again, all the theoretical part of this chapter consists of known results from the literature. This chapter is important since robust sparse recovery algorithms are used in experiments presented in Chapter 7.

Dictionary learning methods are briefly described in Chapter 5. Since there are many dictionary learning methods presented in the literature, only two representatives of state-of-the-art methods, which are used in experimental comparison in Chapter 7, are reviewed here.

Chapter 6 reviewes state-of-the-art non-linear filtering methods for removal of impulse noise in images. Also, this chapter includes a section about maximum-a-posteriori (MAP) approach to removal of impulse noise. There, the reasons for the inefficiency of MAP approach for removal of impulse noise are discussed. Nonlinear filtering methods are used in experimental comparison presented in Chapter 7.

In Chapter 7 we present a comparison of dictionary-based approaches to image inpainting and removal of impulse noise (more precisely, a special kind of impulse noise - salt-and-pepper noise), with a special emphasis on the dictionaries learned by independent component analysis (ICA) (described in Section 2.2). The reasons for inefficiency of non-linear filtering methods for removal of salt-and-pepper noise are also discussed there. In Section 7.3, we describe an application of sparsity constrained matrix factorization (also known as sparse component analysis) for feature extraction on protein/gene expression data.

Summary is presented in Chapter 8.

1.3 Notation and abbreviations

The notation and abbreviations used throughout the thesis are listed in the following tables. Several basic remarks are the following. Scalars are denoted by regular lowercase or uppercase letters. Sets are also denoted by uppercase letters, but it will be emphasized or clear from context whether a symbol (uppercase letter) refers to a scalar or a set. Vectors are denoted by bold lowercase letters. Matrices are denoted by bold uppercase letters.

Table 1.3.1: Notation

$a, b, c, \ldots, M, N, \ldots$	scalars
$f(\cdot), p(\cdot), F(\cdot), \ldots$	functions
$\mathbf{a}, \mathbf{b}, \mathbf{c}, \ldots$	*column* vectors
$\mathbf{A}, \mathbf{B}, \mathbf{C}, \ldots$	matrices
$\Theta, \ldots$	sets
$p_{\mathbf{x}}(\cdot)$	probability *density* function of a random vector $\mathbf{x}$
$\mathbb{E}(\cdot)$	mathematical expectation
$\mathbb{R}$	the set of real numbers
$\mathbb{C}$	the set of complex numbers
$\times$	Cartesian product of sets
$\mathbb{R}^n$	n-dimensional Euclidean space $\mathbb{R} \underbrace{\times \ldots \times}_{(n-1)\text{times}\times} \mathbb{R}$
$\mathscr{N}(\mu, \sigma^2)$	normal (Gaussian) distribution with mean μ and variance σ^2
$\mathscr{U}(a, b)$	uniform distribution in $[a, b]$
I^C	the complement of the set I (in the set which should be clear from context)
(a, b)	open interval of real numbers, $(a, b) = \{x \in \mathbb{R} : a < x < b\}$
$[a, b]$	closed interval of real numbers, $[a, b] = \{x \in \mathbb{R} : a \leq x \leq b\}$
$[a, b)$	half-open interval of real numbers, $[a, b) = \{x \in \mathbb{R} : a \leq x < b\}$
x_i, $(\mathbf{x})_i$	i-th element of vector $\mathbf{x}$
$\lvert\cdot\rvert$	cardinality of a set, or the absolute value of a number
$\lVert\mathbf{x}\rVert_0$	ℓ_0 quasi-norm (not really a norm) of vector $\mathbf{x}$, $\lVert\mathbf{x}\rVert_0 = \lvert\{i : \mathbf{x}_i \neq 0\}\rvert$
$\lVert\mathbf{x}\rVert_1$	ℓ_1 norm of $\mathbf{x}$, $\lVert\mathbf{x}\rVert_1 = \sum_i \lvert\mathbf{x}_i\rvert$
$\lVert\mathbf{x}\rVert_2$	ℓ_2 norm (2-norm) of $\mathbf{x}$, $\lVert\mathbf{x}\rVert_2 = \sqrt{\sum_i \mathbf{x}_i^2}$
$\mathbf{a}^T$, $\mathbf{A}^T$	transpose of vector $\mathbf{a}$ (therefore, $\mathbf{a}^T$ is a *row* vector), respectively matrix $\mathbf{A}$
$\mathbf{0}$	the null-vector, i.e. the vector whose all elements are equal to zero
$\text{null}(\mathbf{A})$	the nullspace (kernel) of matrix $\mathbf{A}$, $\text{null}(\mathbf{A}) = \{\mathbf{x} : \mathbf{A}\mathbf{x} = \mathbf{0}\}$

Table 1.3.2: Abbreviations

ICA	Independent Component Analysis
PCA	Principal Component Analysis
SVD	Singular Value Decomposition
SCA	Sparse Component Analysis
SSP	Single Source Point
NMF	Nonnegative Matrix Factorization
OMP	Orthogonal Matching Pursuit
SL0	Smoothed ℓ_0
DCT	Discrete Cosine Transform
MOD	Method of Optimal Directions
FoE	Fields of Experts
MCA	Morphological Component Analysis
SSIM	Structural Similarity
PSNR	Peak Signal-to-Noise Ratio
dB	deciBel
MSE	Mean Squared Error
IST	Iterative Soft Thresholding
SVM	Support Vector Machine
RBF	Radial Basis Function
ML	Maximum Likelihood
MAP	Maximum A Posteriori
CV	Cross-Validation
p.d.f.	probability density function
i.i.d.	independent and identically distributed
i.e.	that is (*id est*)

Chapter 2

Linear instantaneous blind source separation

2.1 Problem formulation

The general model for the *linear instantaneous blind source separation* is presented as follows: a set of T *observations* of M *sensors*

$$\mathbf{X} = [\mathbf{x}_1 | \cdots | \mathbf{x}_T] = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1T} \\ x_{21} & x_{22} & \cdots & x_{2T} \\ \vdots & \vdots & \ddots & \vdots \\ x_{M1} & x_{M2} & \cdots & x_{MT} \end{bmatrix}$$

is modeled as a linear mixture of N source signals

$$\mathbf{S} = [\mathbf{s}_1 | \cdots | \mathbf{s}_T] = \begin{bmatrix} s_{11} & s_{12} & \cdots & s_{1T} \\ s_{21} & s_{22} & \cdots & s_{2T} \\ \vdots & \vdots & \ddots & \vdots \\ s_{N1} & s_{N2} & \cdots & s_{NT} \end{bmatrix},$$

where the linear mixing is represented by the *unknown* $M \times N$ mixing matrix

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1N} \\ a_{21} & a_{22} & \cdots & a_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ a_{M1} & a_{M2} & \cdots & a_{MN} \end{bmatrix},$$

i.e. the mixing model is written as

$$\mathbf{X} = \mathbf{AS}. \tag{2.1.1}$$

In practice, the model (2.1.1) is not exact since there are always some errors due to imperfect measurements, noise, or even errors due to model imperfection. Therefore, practically more relevant model formulation is

$$\mathbf{X} = \mathbf{AS} + \mathbf{E}, \tag{2.1.2}$$

where $\mathbf{E}$ is the error matrix of small norm. Here, every row of the matrix $\mathbf{S}$ represents a source *signal*, and hence every row of the matrix $\mathbf{X}$ is a mixed signal. Formally, a signal is a discretization of the continuous function on some interval. For signals defined on a subset of $\mathbb{R}$, like audio/speech, independent variable usually refers to *time*, and an observation is a value of the signal at the given time instant. For discrete signals defined on a subset of $\mathbb{R}^2$, like digital images, the independent variables refer to the *spatial* location. By sensors we will mean distinct mixtures of the unknown source signals. The problem of blind source separation consists in recovering (approximating) the source signals (which also requires the mixing matrix to be recovered/approximated) given the mixed signals only.

Now, first of all, linear mixture models can be divided into two main classes. If $M \geq N$, i.e. the number of mixed signals (sensors) is greater than or equal to the number of source signals, the mixing model is said to be *(over)determined.* If $M < N$, the model is said to be *underdetermined.* Since in most blind source separation methods the mixing matrix is approximated first, the main distinction between the overdetermined and underdetermined cases is that in the (over)determined case, once the mixing matrix is known (approximated), the matrix of source signals is obtained by inversion. On the contrary, approximating the mixing matrix is only half of the problem in the underdetermined case. Since both the mixing matrix $\mathbf{A}$ and the matrix of source signals $\mathbf{S}$ are unknown, the problem (2.1.1) as described above, is, of course, ill-posed in both the (over)determined and especially underdetermined case, i.e. there are infinitely many solutions. Namely, if $\hat{\mathbf{S}}$ is one solution/approximation of the source matrix (where $\hat{\mathbf{A}}$ is the corresponding approximation of the mixing matrix), any $\tilde{\mathbf{S}}$ of the form $\tilde{\mathbf{S}} = \mathbf{B}\hat{\mathbf{S}}$, where $\mathbf{B} \in \mathbb{R}^{N \times N}$ is an invertible matrix, is also a solution/approximation (corresponding approximation of the mixing matrix is $\hat{\mathbf{A}}\mathbf{B}^{-1}$). That is because representations $\mathbf{X} = \hat{\mathbf{A}}\hat{\mathbf{S}}$ and $\mathbf{X} = \hat{\mathbf{A}}\mathbf{B}^{-\mathbf{1}}\mathbf{B}\hat{\mathbf{S}}$ are equivalent from the data point of view. Therefore, some constraints (regularizations) need to be introduced to make the problem well-defined. Therefore, it should be mentioned that the name *blind* source separation should only indicate that very little is known a-priori about the sources, i.e. the source matrix $\mathbf{S}$. Without some assumptions/knowledge about the structure of the source signals, the problem is ill-posed, as explained above. Therefore, the problem of blind source separation is often referred to as semi-blind source separation or source separation. We will often use the name source separation. Two of the most often used assumptions/constraints on the source signals are statistical independence (measured through some contrast function, explained in Section 2.2.3) in the (over)determined case, and sparsity in the underdetermined case. These are explained in the following sections of this chapter.

Before we go into more detailed description of the (over)determined and underdetermined cases, we briefly mention the statistical interpretation of the linear model in (2.1.1) (this is presented in more detail in the following section). It is supposed that the linear model

$$\mathbf{x} = \mathbf{A}\mathbf{s} \tag{2.1.3}$$

is valid, where $\mathbf{x}$ is a *random vector* in $\mathbb{R}^M$, $\mathbf{A} \in \mathbb{R}^{M \times N}$ is a deterministic mixing matrix, as before, and $\mathbf{s}$ is a *random vector* in $\mathbb{R}^N$. The linear model in (2.1.1) is then viewed as follows: every column of the source matrix $\mathbf{S}$ is a realization of the random vector $\mathbf{s}$, and likewise every column of $\mathbf{X}$ is a realization of the random vector $\mathbf{x}$.

The rest of the chapter is organized as follows. In Section 2.2 we review basic methods for the solution of the (over)determined blind source separation problem, with an emphasis on the *independent component analysis (ICA)*. Section 2.3 reviews methods for under-determined blind source separation, with special emphasis on the methods that assume sparsity of the source signals. In that section, we concentrate on *sparse component analysis* methods. We review some basic results on over-complete ICA in Section 2.3.3. For other approaches to both the (over)determined and underdetermined problems the main reference is the handbook [22].

2.2 Over-determined case and the independent component analysis

The simpler cases of the source separation problems (2.1.1) and (2.1.2) are the over-determined case, when $M > N$, and determined case, when $M = N$. Since these cases are conceptually very similar, we will use the term over-determined case both when $M > N$ and $M = N$. The technical assumption that will be used is the invertibility of the mixing matrix $\mathbf{A}$. This is equivalent to the full rank assumption on $\mathbf{A}$. The often used assumption on the sources in the over-determined case is the *statistical independence of the sources*. The method for source separation using the assumption of statistical independence of the sources is called independent component analysis (ICA) [55]. We start with the model (2.1.3). The covariance matrix of a random vector $\mathbf{x} \in \mathbb{R}^M$ is defined as $\mathbf{C}_\mathbf{x} = \mathbb{E}\left((\mathbf{x} - \mathbb{E}(\mathbf{x}))(\mathbf{x} - \mathbb{E}(\mathbf{x}))^T\right)$. The covariance matrix is finite if all *correlations* $\mathbb{E}\left(x_i x_j\right)$, $i, j = 1, \ldots, M$ (*second moments* if $i = j$), are finite. The following definitions and results are taken from the seminal paper on ICA [21]. First of all, we present a formal definition of the ICA.

Definition 2.1. ICA of a random vector $\mathbf{x} \in \mathbb{R}^M$ with finite covariance matrix $\mathbf{C}_\mathbf{x}$ is a pair $(\mathbf{F},\ \Lambda)$ of matrices such that

1. the covariance matrix $\mathbf{C}_\mathbf{x}$ can be factorized as $\mathbf{C}_\mathbf{x} = \mathbf{F}\Lambda^2\mathbf{F}^T$, where Λ is diagonal matrix with positive elements, and $\mathbf{F}$ is $M \times \rho$ full column rank matrix, where $\rho \leq M$;
2. $\mathbf{x}$ can be written as $\mathbf{x} = \mathbf{F}\mathbf{z}$, where $\mathbf{z}$ is a $\rho \times 1$ random vector with the covariance matrix Λ^2 and the components of $\mathbf{z}$ are *maximally independent* in the sense of maximization of a given *contrast function*.

Before discussing relevant contrast functions from the definition, we discuss the ambiguities of the above definition and therefore introduce some constraints which make the definition of ICA good.

Remark 2.1. If a pair $(\mathbf{F},\, \Lambda)$ is an ICA of a random vector $\mathbf{y}$, then the pair $(\mathbf{F}',\, \Lambda')$ is also an ICA of $\mathbf{y}$, where

$$\mathbf{F}' = \mathbf{F}\bar{\Lambda}\mathbf{P}, \quad \Lambda' = \mathbf{P}^T\bar{\Lambda}^{-1}\Lambda\mathbf{P} \tag{2.2.1}$$

and $\bar{\Lambda}$ is a $\rho \times \rho$ invertible diagonal matrix, while $\mathbf{P}$ is a $\rho \times \rho$ permutation matrix.

Remark 2.1 highlights the *scaling and permutation ambiguities of the ICA*. We consider the two ICA-s in the above remark equivalent, and therefore all ICA-s of a random vector form one equivalence class. Usually, the following two constraints are introduced to define the unique representative of ICA equivalence class:

1. the columns of $\mathbf{F}$ have unit norm;
2. the entries of Λ are sorted in decreasing order.

It should be noted that the definition of a contrast function from (2) in Definition 2.1 should take into account the scaling and permutation ambiguities. Some contrast functions for ICA are introduced in Section 2.2.3.

2.2.1 Standardization and the principal component analysis (PCA)

The goal of standardization is to transform a random vector $\mathbf{x} \in \mathbb{R}^M$ into another, $\mathbf{z}$, that has a unit covariance matrix (i.e., identity). Therefore, the elements of $\mathbf{z}$ are *uncorrelated*. It is possible that the covariance matrix $\mathbf{C}_\mathbf{x}$ of $\mathbf{x}$ is not invertible, which requires, apart from decorrelation, also a projection of $\mathbf{x}$ onto the range space of $\mathbf{C}_\mathbf{x}$. Both of these tasks are accomplished by PCA.

Definition 2.2. The principal component analysis (PCA) of a random vector $\mathbf{y} \in \mathbb{R}^M$ with finite covariance matrix is a pair $(\mathbf{F},\, \Lambda)$ of matrices such that the covariance matrix $\mathbf{C}_\mathbf{y}$ of $\mathbf{y}$ can be factorized as $\mathbf{C}_\mathbf{y} = \mathbf{F}\Lambda^2\mathbf{F}^T$, where Λ is diagonal matrix with positive elements and $\mathbf{F} \in \mathbb{R}^{M\times\rho}$ is full column rank matrix, with orthogonal columns.

PCA shares the same ambiguities as ICA, and therefore PCA-s of a random vector also form an equivalence class. Therefore, the constraints introduced at the end of the previous section are also used for PCA.

We briefly describe the computation of the PCA. Let us denote the (truncated) eigenvalue decomposition of the covariance $\mathbf{C}_\mathbf{x}$ of $\mathbf{x}$ as $\mathbf{C}_\mathbf{x} = \mathbf{U}\Gamma^2\mathbf{U}^T$, where Γ is full rank and $\mathbf{U}$ possibly rectangular. From the definition of the PCA, the pair $(\mathbf{U},\, \Gamma)$ is the PCA of $\mathbf{x}$. Then, the standardized vector associated to $\mathbf{x}$ is defined as $\mathbf{z} = \Gamma^{-1}\mathbf{U}^T\mathbf{x}$. It is also possible to use the singular value decomposition (SVD) in the calculation of the PCA to avoid explicit calculation of the covariance matrix $(\mathbf{C}_\mathbf{x})$.

2.2.2 Measures of statistical independence

In the Definition 2.1, it is required that the components of $\mathbf{z}$ are 'maximally statistically independent'. This definition requires a valid measure of statistical (in)dependence. In this section, we review the most important ones.

2.2.2.1 Mutual information

We denote by $p_{\mathbf{x}}(\cdot)$ the probability density function (p.d.f.) of vector $\mathbf{x}$. $\mathbf{x}$ has (mutually) independent components if (by definition)

$$p_{\mathbf{x}}(\mathbf{u}) = \prod_{i=1}^{M} p_{x_i}(u_i), \tag{2.2.2}$$

where $\mathbf{x} = [x_1, \ldots, x_M]^T$ and $\mathbf{u} = [u_1, \ldots, u_M]^T$. A natural way to measure independence is to measure a distance between sides in (2.2.2), $\delta\left(p_{\mathbf{x}}, \prod_{i=1}^{M} p_{x_i}\right)$, where $\delta(\cdot)$ is some measure of distance. One class of measures of distance of two probability density functions $p_{\mathbf{x}}$ and $p_{\mathbf{y}}$ on $\mathbb{R}^M$ are *f-divergences*, denoted and defined as

$$D_f(p_{\mathbf{x}}, p_{\mathbf{y}}) = \int_{\mathbb{R}^M} f\left(\frac{p_{\mathbf{x}}(\mathbf{u})}{p_{\mathbf{y}}(\mathbf{u})}\right) p_{\mathbf{x}}(\mathbf{u})\, d\mathbf{u},$$

where f is a convex function such that $f(1) = 0$. *f-divergences* are non-negative and positive definite in the sense $D_f(p_{\mathbf{x}}, p_{\mathbf{y}}) = 0 \Leftrightarrow p_{\mathbf{x}} = p_{\mathbf{y}}$ a.e. (almost everywhere). Therefore, $D_f\left(p_{\mathbf{x}}, \prod_{i=1}^{M} p_{x_i}\right)$ are 'good' measures of independence of vector $\mathbf{x}$. *Kullback-Leibler (KL) divergence* is obtained for $f(t) = -\ln t$. KL divergence of random variables is defined as the KL divergence of their p.d.f.-s. *Mutual information of a vector* $\mathbf{x}$, denoted as $I(p_{\mathbf{x}})$ or $I(\mathbf{x})$, is defined as the Kullback-Leibler divergence between the sides in (2.2.2): $I(p_{\mathbf{x}}) = D_{KL}\left(p_{\mathbf{x}}, \prod_{i=1}^{M} p_{x_i}\right)$. This is a good candidate for the contrast function for ICA. Two important properties of mutual information that follow from the properties of Kullback-Leibler divergence are stated in the following proposition.

Proposition 2.1. *Mutual information has the following properties:*

1. $I(\mathbf{x}) \geq 0$, *for every random vector* $\mathbf{x}$*;*
2. $I(\mathbf{x}) = 0 \Leftrightarrow$ *the components* $x_1, \ldots, x_M$ *of* $\mathbf{x}$ *are mutually statistically independent.*

Mutual information can also be defined/expressed using *entropy*. (Differential) entropy of a random vector $\mathbf{x}$, denoted as $H(\mathbf{x})$, $H_{\mathbf{x}}$ or $H_{p_{\mathbf{x}}}$, where $p_{\mathbf{x}}(\cdot)$ is the p.d.f. of $\mathbf{x}$, is defined as

$$H(\mathbf{x}) = -\int p_{\mathbf{x}}(\mathbf{u}) \log p_{\mathbf{x}}(\mathbf{u})\, d\mathbf{u}. \tag{2.2.3}$$

It follows from the definition of mutual information that

$$I(\mathbf{x}) = \sum_{i=1}^{M} H(x_i) - H(\mathbf{x}).$$

.

2.2.2.2 Negentropy

The well known result of information theory (see [23], for example) states that, under some regularity conditions, Gaussian density has maximum entropy among densities with equal variance. A simple proof can be presented as follows: it is known that the Kullback-Leibler divergence is nonnegative, $D_{KL}(p,\, q) \geq 0$, for every probability density functions p and q (this is a consequence of Jensen's inequality). Let $g(x)$ be the Gaussian p.d.f. with mean μ and variance σ^2, and let $f(x)$ be an arbitrary p.d.f. with the same mean and variance. It follows that

$$0 \leq D_{KL}(f,\, g) = -H_{f(x)} - \int_{\mathbb{R}} f(x) \log g(x)\, dx$$

where $H_{f(x)}$ denotes the entropy of $f(x)$, which is defined in the same way as the entropy of a random variable. The second term above can be written as

$$\begin{aligned}
\int_{\mathbb{R}} f(x) \log g(x)\, dx &= \int_{\mathbb{R}} f(x) \log\left(\frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)\right) dx \\
&= \int_{\mathbb{R}} f(x) \log\left(\frac{1}{\sqrt{2\pi\sigma^2}}\right) dx + \\
&\quad + \int_{\mathbb{R}} f(x) \left(-\frac{(x-\mu)^2}{2\sigma^2}\right) dx \\
&= -\frac{1}{2}\log\left(2\pi\sigma^2\right) - \frac{\sigma^2}{2\sigma^2} \\
&= -\frac{1}{2}\left(\log\left(2\pi\sigma^2\right) + 1\right) \\
&= -H_{g(x)}
\end{aligned}$$

which is the entropy of the Gaussian p.d.f. Therefore, $H_{g(x)} \geq H_{f(x)}$, for an arbitrary p.d.f. $f(x)$. It follows from the property of the Kullback-Leibler divergence (i.e. from the property of Jensen's inequality) that an equality is true if and only if $f(x) = g(x)$ a.e. The above proof of maximality of entropy of a Gaussian p.d.f. also extends to random vectors.

Negentropy of a random vector $\mathbf{x}$ is defined as $J(\mathbf{x}) = H(\mathbf{x}_{Gauss}) - H(\mathbf{x})$, where $\mathbf{x}_{Gauss}$ is a random vector with the same mean and covariance matrix as x. It follows from the above results that negentropy of a random vector is non-negative, and equal to zero if and only if the vector is Gaussian. Therefore, negentropy is a measure of distance of a random vector from the Gaussian random vector with the same mean and covariance matrix, i.e. it is a measure of 'non-Gaussianity'.

Here, we are interested in measures of *statistical independence*. Therefore, we provide an intuition for using non-Gaussianity as a measure of statistical independence (see chapter 8 in [55]). From the linear model of ICA $\mathbf{x} = \mathbf{A}\mathbf{s}$ it follows $\mathbf{s} = \mathbf{A}^{-1}\mathbf{x}$, i.e. independent components s_i are obtained by linearly combining mixed components $\mathbf{x}_i$. Let us denote by $y = \mathbf{b}^T\mathbf{x}$ one such linear combination. It follows $y = \mathbf{b}^T\mathbf{A}\mathbf{s} = \mathbf{q}^T\mathbf{s}$, for $\mathbf{q} = \mathbf{A}^T\mathbf{b}$. Roughly, by central limit theorem it follows that y, as a linear combination of independent components, is 'more Gaussian' than individual components s_i. Since $y = \mathbf{b}^T\mathbf{x} = \mathbf{q}^T\mathbf{s}$, the intuition is that by *maximizing non-Gaussianity* of y (one of the) independent components s_i can be obtained.

The following connection provides a rigorous justification for the use of negentropy as a measure of statistical independence. Namely, mutual information can be expressed using negentropy. Using the expressions for the entropy of a Gaussian random vector $\mathbf{x}_{Gauss}$ and the entropies of its components $(\mathbf{x}_{Gauss})_i$ it follows that

$$I(\mathbf{x}) = J(\mathbf{x}) - \sum_i J(x_i) + \frac{1}{2} \log \frac{\prod_i \Sigma_{ii}}{\det \Sigma}, \tag{2.2.4}$$

where $\Sigma = \mathbf{C}_\mathbf{x}$. Note that, for standardized random vector $\mathbf{x}$, the term $\frac{1}{2}\log\frac{\prod_i \Sigma_{ii}}{\det\Sigma}$ is equal to zero. Here we also mention one useful property of negentropy. Namely, negentropy is invariant to invertible linear transformations, i.e. for invertible matrix $\mathbf{M}$, $J(\mathbf{Mx}) = J(\mathbf{x})$. See [55] for a proof of this invariance property.

2.2.3 Contrast functions for ICA

To be a valid contrast function for ICA, we will require of a measure of statistical independence the properties listed in the following definition.

Definition 2.3. A *contrast* function for ICA is a mapping Ψ from the set of probability densities $\{p_\mathbf{x},\ \mathbf{x} \in \mathbb{R}^M\}$ to $\mathbb{R}$ that satisfies the following:

1. $\Psi(p_{\mathbf{Px}}) = \Psi(p_\mathbf{x})$, for every permutation matrix $\mathbf{P}$ (Ψ is invariant to permutations);
2. $\Psi(p_{\Gamma\mathbf{x}}) = \Psi(p_\mathbf{x})$, for every invertible diagonal matrix Γ (Ψ is invariant to scaling);
3. if $\mathbf{x}$ has independent components, then $\Psi(p_{\mathbf{Ax}}) \le \Psi(p_\mathbf{x})$, for every invertible matrix $\mathbf{A}$.

The following definition introduces one important property that will be required of the contrast function in the definition of ICA.

Definition 2.4. A contrast is *discriminating* if the equality in (3) holds only when $\mathbf{A}$ is of the form $\mathbf{A} = \Gamma\mathbf{P}$, where Γ is invertible diagonal and $\mathbf{P}$ is a permutation matrix.

Now, the following theorem from [21] states the conditions under which ICA, if it exists, is unique up to permutation and scaling ambiguities.

Theorem 2.1. *Assume that* $\mathbf{x}$ *is a random vector with independent components of which at most one is Gaussian, and whose densities are not point-like masses. Let* $\mathbf{C}$ *be an orthogonal matrix, and vector* $\mathbf{z}$ *defined as* $\mathbf{z} = \mathbf{Cx}$. *Then the following statements are equivalent:*

1. *the components* z_i *of* $\mathbf{z}$ *are pairwise independent;*
2. *the components* z_i *are mutually independent;*
3. $\mathbf{C} = \Gamma\mathbf{P}$, *where* Γ *is invertible diagonal and* $\mathbf{P}$ *a permutation matrix.*

This theorem is a consequence of the following one.

Theorem 2.2. *Let* $\mathbf{x}$ *and* $\mathbf{z}$ *be two random vectors such that* $\mathbf{z} = \mathbf{B}\mathbf{x}$, *where* $\mathbf{B}$ *is a rectangular matrix. Suppose that* $\mathbf{x}$ *has independent components and that* $\mathbf{z}$ *has pairwise independent components. If* $\mathbf{B}$ *has two non-zero entries in the same column* j*, then* x_j *is either Gaussian or deterministic.*

This theorem can be proven using several known results in probability theory, see [21] for details. Using these results, it follows that the negative mutual information is a valid contrast for ICA.

Theorem 2.3. *The mapping*

$$\Psi(p_{\mathbf{x}}) = -I(p_{\mathbf{z}})$$

is a contrast for ICA, where $\mathbf{z}$ *is a standardized random vector associated to* $\mathbf{x}$. Ψ *is also discriminating over the set of random vectors having at most one Gaussian component.*

Proof. That Ψ is a contrast follows from the properties of mutual information (see Proposition 2.1). To prove that Ψ is discriminating, we suppose $\Psi(p_{\mathbf{Ax}}) = \Psi(p_{\mathbf{x}})$, where $\mathbf{x}$ has independent components of which at most one is Gaussian. It follows that $\Psi(p_{\mathbf{x}}) = 0$, and therefore also $\Psi(p_{\mathbf{Ax}}) = 0$. Again, from the Proposition 2.1 it follows that $\mathbf{Ax}$ has independent components. Since $\mathbf{x}$ has at most one Gaussian component, from the Theorem 2.1 it follows that $\mathbf{A} = \Gamma\mathbf{P}$. $\square$

From the connection between mutual information and negentropy (see Section 2.2.2.2) it follows that negentropy can also be used as a contrast for ICA.

Mutual information and negentropy are computationally very complicated to use because their definitions involve generally unknown probability density functions of the sources. Therefore, in practice some approximations are used. Since both mutual information and negentropy are expressed using entropy, in the following subsections we introduce some approximations of entropy.

2.2.3.1 Approximations of entropy by cumulants

We briefly recall the definitions of the first and second characteristic functions.

Definition 2.5. The first characteristic function of a random vector $\mathbf{x}$ in $\mathbb{R}^M$ with p.d.f. $p_{\mathbf{x}}$, denoted as $\phi_{p_{\mathbf{x}}}$ or $\phi_{\mathbf{x}}$, is defined as a function $\phi_{\mathbf{x}} : \mathbb{R}^M \to \mathbb{C}$,

$$\phi_{\mathbf{x}}(\mathbf{t}) = \int_{\mathbb{R}^M} \exp\left(j\mathbf{t}^T\mathbf{u}\right) p_{\mathbf{x}}(\mathbf{u})\, d\mathbf{u}.$$

Here, j denotes imaginary unit. k-th moment of a random variable x is defined as $\mathbb{E}\left(x^k\right)$. We denote it by $m_x^{(k)}$. In the case of random vectors, definition of moments is more complicated. First moment of a random vector is the mean of a vector, while the second moment of a vector is its correlation matrix (covariance matrix in the case of a zero-mean vector). Higher-order moments are defined as follows.

Definition 2.6. The k-th order moment of a random vector $\mathbf{x}$ in $\mathbb{R}^M$ is a k-th order *tensor* $\mathscr{M}_{\mathbf{x}}^{(k)} \in \mathbb{R}^{M\times M\times\cdots\times M}$ (k-dimensional array) with elements

$$\left(\mathscr{M}_{\mathbf{x}}^{(k)}\right)_{i_1 i_2 \ldots i_k} = \mathbb{E}\left(x_{i_1} x_{i_2} \cdots x_{i_k}\right).$$

In the case of a random *variable* x, if k-th absolute moment of x is finite, k-th moment can be obtained as the k-th coefficient in the Taylor expansion of the first characteristic function about 0, up to constant. Similarly, k-th order moment of a random *vector* $\mathbf{x}$ can be obtained from the coefficients in the Taylor expansion of its first characteristic function about the origin, again up to constant.

Now we define cumulants. Before that, we need to define the second characteristic function of a random vector.

Definition 2.7. The second characteristic function of a random vector $\mathbf{x}$ in $\mathbb{R}^M$, $\Omega_{\mathbf{x}} : \mathbb{R}^M \to \mathbb{C}$, is defined as the (complex) logarithm of $\phi_{\mathbf{x}}$:

$$\Omega_{\mathbf{x}}(\mathbf{t}) = \ln \phi_{\mathbf{x}}(\mathbf{t}).$$

Cumulants of a random vector x are now defined as the coefficients (up to constants) in the Taylor expansion of the second characteristic function $\Omega_{\mathbf{x}}$ about the origin, similarly to moments. Namely, we have the following definition.

Definition 2.8. k-th order cumulants of a random vector $\mathbf{x}$ in $\mathbb{R}^M$, $\operatorname{cum}\left(x_{i_1},\ldots,x_{i_k}\right)$, $1 \le i_1,\ldots,i_k \le M$, are defined as

$$\operatorname{cum}\left(x_{i_1},\ldots,x_{i_k}\right) = (-j)^k \left.\frac{\partial^k \Omega_{\mathbf{x}}(t)}{\partial t_{i_1} \partial t_{i_2} \cdots \partial t_{i_k}}\right|_{t=0}.$$

k-th order cumulant tensor $\mathscr{C}_{\mathbf{x}}^{(k)} \in \mathbb{R}^{M\times M\times\cdots\times M}$ of $\mathbf{x}$ is a tensor of order k with elements $\operatorname{cum}\left(x_{i_1},\ldots,x_{i_k}\right)$. $\mathscr{C}_{\mathbf{x}}^{(k)}$ is symmetric (see properties of cumulants in Appendix A). k-th order cumulant of a random *variable* x is denoted as $c_x^{(k)}$.

Cumulants of a random vector that involve different components of a vector are often called *cross-cumulants*. Cumulants can be expressed as functions of moments from the definition of Ψ. First-order cumulant is equal to the first moment (expectation), while the second-order cumulants are the components of the covariance matrix. Cumulants have some very important properties which make them more useful than moments in applications. Some of the properties of cumulants are listed in Appendix A.

Cumulants are important regarding an approximations of entropy because they appear as coefficients in the *Gram-Charlier expansion* of the p.d.f. of a random variable about the Gaussian variable. The Gram-Charlier expansion of the p.d.f. p_x of a random variable x with mean $\mu = 0$ and variance $\sigma^2 = 1$ about the standard normal (Gaussian) p.d.f. ϕ is given by

$$p_x(\zeta) = \phi(\zeta)\left(1 + c_x^{(3)}\frac{H_3(\zeta)}{3!} + c_x^{(4)}\frac{H_4(\zeta)}{4!} + \dots\right), \tag{2.2.5}$$

where H_i represents i-th Hermite polynomial (defined by the i-th derivative of the standard normal p.d.f.). An approximation of p_x can be obtained by keeping only several terms in the above expansion. By using this truncated expansion as an approximation of p_x and plugging it into the expression for entropy (2.2.3), simple approximations of entropy can be obtained (for more details, see [55]). See Appendix A for the formal derivation of the expression (2.2.5) and some references.

An important and often used higher-order cumulant is kurtosis. Kurtosis of a random variable x is denoted as $\kappa(x)$ and defined as $\kappa(x) = \frac{c_x^{(4)}}{\left(c_x^{(2)}\right)^2}$. For a standardized random variable z, $\kappa(z) = \mathbb{E}\left(z^4\right) - 3$, therefore it is a normalized fourth-order moment. Kurtosis of a Gaussian random variable is zero. For many non-Gaussian random variables, kurtosis is nonzero. Random variables with negative kurtosis are called sub-Gaussian (or platykurtic), while those with positive kurtosis are called super-Gaussian (or leptokurtic). Super-Gaussian random variables have typically 'spiky' p.d.f., with heavy tails and relatively large at zero (compared to the Gaussian random variable), a representative example being the Laplacian distribution (with unit variance), with p.d.f. $p(x) = \frac{1}{\sqrt{2}}\exp\left(-\sqrt{2}\,|x|\right)$. Sub-Gaussian random variables typically have a flat p.d.f., rather constant near zero and very small for larger values of the variable, typical example being the uniform distribution. Since the kurtosis is zero for the Gaussian random variable and nonzero for most of the interesting non-Gaussian distributions, it can be used as a measure of non-Gaussianity.

However, there are several drawbacks of using cumulants in practice. Namely, finite-sample estimators of moments, and therefore also cumulants, are very sensitive to outliers (possibly erroneous observations with large values). This means that large erroneous values can completely determine the estimates of cumulants, which makes these estimates useless. Therefore, in the following subsections we review other methods for approximating entropy.

2.2.3.2 Approximation of entropy by non-polynomial functions

Assume that the information about the density f is available through the expectations

$$c_i = \mathbb{E}\left(G_i(x)\right) = \int G_i(u) f(u) du, \; i = 1, \dots, n. \tag{2.2.6}$$

The estimation of f based on these measurements only is ill-posed without additional assumptions on f. The idea of *maximum entropy method* is to find the p.d.f. with maximal entropy satisfying the measurements (2.2.6). A result from information theory ([23]) states that, under some regularity assumptions (more precisely, it is assumed that the constraints (2.2.6) can be satisfied), the density $f_0(x)$ that satisfies the constraints (2.2.6) and has maximum entropy among all such densities, has the form

$$f_0(x) = A\exp\left(\sum_i a_i G_i(x)\right) \tag{2.2.7}$$

where A and a_i are constants that *can* be determined from (2.2.6) and the constraint $\int f_0(x)dx = 1$. However, determining these constants requires solving a system of non-linear equations. Therefore, in [52] the following approximate approach was suggested. We assume that f_0 is *not very far from* standardized *Gaussian* density $\phi(x) = \exp\left(-x^2/2\right)/\sqrt{2\pi}$. The standardization assumption introduces additional two constraints in (2.2.6) defined by $G_{n+1} = x$, $c_{n+1} = 0$, $G_{n+2} = x^2$, $c_{n+2} = 1$. We also assume that the functions G_i form an orthonormal system with respect to ϕ and are orthogonal to all polynomials of degree ≤ 2. Then, we can express f_0 as

$$f_0(x) = A\exp\left(-\frac{x^2}{2} + a_{n+1}x + \left(a_{n+2} + \frac{1}{2}\right)x^2 + \sum_{i=1}^{n} a_i G_i(x)\right)$$

where in the exponential all other terms are *small* compared to the first one. Now, first-order approximation of exponential, $\exp\varepsilon \approx 1 + \varepsilon$, can be used to get

$$f_0(x) \approx \tilde{A}\phi(x)\left(1 + a_{n+1}x + \left(a_{n+2} + \frac{1}{2}\right)x^2 + \sum_{i=1}^{n} a_i G_i(x)\right)$$

where $\tilde{A} = \sqrt{2\pi}A$. Using the orthogonality constraints for G_i we get

$$\begin{array}{rcccl} \int f_0(x)dx & \approx & \tilde{A}\left(1 + \left(a_{n+2} + \frac{1}{2}\right)\right) & = & 1 \\ \int f_0(x)xdx & \approx & \tilde{A}a_{n+1} & = & 0 \\ \int f_0(x)x^2dx & \approx & \tilde{A}\left(1 + 3\left(a_{n+2} + \frac{1}{2}\right)\right) & = & 1 \\ \int f_0(x)G_i(x) & \approx & \tilde{A}a_i & = & c_i,\ i = 1,\ldots,n \end{array}$$

which yields $\tilde{A} = 1$, $a_{n+1} = 0$, $a_{n+2} = -\frac{1}{2}$ and $a_i = c_i$, $i = 1,\ldots,n$. Finaly, we obtain an approximative density $\hat{f}$,

$$\hat{f}(x) = \phi(x)\left(1 + \sum_{i=1}^{n} c_i G_i(x)\right). \tag{2.2.8}$$

This approximation can be used in the definition of entropy to obtain an approximation of entropy. Namely, we have (see [52] for a derivation)

$$H_{\hat{f}} = -\int \hat{f}(x)\log\hat{f}(x)dx \approx H(\nu) - \frac{1}{2}\sum_{i=1}^{n} c_i^2, \tag{2.2.9}$$

where $H(\nu) = \frac{1}{2}(1 + \log(2\pi))$ is the entropy of a standardized Gaussian variable, and $c_i = \mathbb{E}(G_i(X))$, as above. Notice that the functions G_i can be chosen to obtain 'good' approximations of entropy. Namely, some properties that G_i should have are the following. First, estimation of expectation(s) $\mathbb{E}(G_i)$ should be robust to outliers, contrary to cumulants. Second, the function f_0 in (2.2.7) should be integrable. Third, notice that if the true p.d.f. f were known, optimal choice would be $G = -\log f$ (the approximation $H_{\hat{f}}$ is *exact* in this case). This fact motivates the choice of G_i-s as log-densities of some important distributions. A simple special case of an approximation of the form (2.2.8) is obtained when $n = 1$. The FastICA algorithm, which we review in Section 2.2.4.3, and which is used in the experiments presented in Chapter 7, is based on the approximation of this form.

2.2.3.3 Other approximations of entropy

Entropy can be estimated directly from the definition, using kernel estimates of unknown p.d.f. First reference using this approach was [87] (or chapter 2 in [22]). See also [14]. The main problem with these methods is their high computational complexity, which makes them impractical when the number of sources N is large (when $N \gtrapprox 100$ the problem is usually classified as large). Some other methods estimate the unknown p.d.f.- s indirectly, for example through sample estimates of quantiles [88, 62]. Again, their problem is high computational complexity, which is impractical when N is large. Therefore, we don't discuss these methods in more detail here.

2.2.4 Information-theoretic approaches to ICA

2.2.4.1 Maximum likelihood

In this subsection we make connection between the mutual information minimization and the maximum likelihood principle as the methods to perform ICA. We start from the ICA model (2.1.3). We briefly recall the definition of the likelihood and the maximum likelihood principle.

Definition 2.9. Let X be a random variable with p.d.f. $p_{X,\theta}$ that depends on a parameter $\theta \in \Theta$. The likelihood function of X, $L_\theta(\cdot)$, is defined as

$$L_\theta(x) = p_{X,\theta}(x).$$

The likelihood is defined in the same way for a random *vector* $\mathbf{x}$. The likelihood of a *sample* $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T$ of independent observations of a random vector $\mathbf{x}$ with p.d.f. $p_{\mathbf{x},\theta}$ is defined as

$$L_\theta(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T) = \prod_{i=1}^{T} p_{\mathbf{x},\theta}(\mathbf{x}_i).$$

It is often more convenient to work with the logarithm of the likelihood (log-likelihood), $\ln L_\theta$, or its scaled version, the average log-likelihood, $\hat{l}_\theta = \frac{1}{T}\ln L_\theta$. Now, recall the definition of the maximum likelihood estimator.

Definition 2.10. Suppose that $x_1, x_2, \ldots, x_T$ is a sample of a random variable X with p.d.f. $p_{X,\theta}$ that depends on a parameter $\theta \in \Theta$ and that θ_0 is the true value of θ (i.e., $x_1, x_2, \ldots, x_T$ are generated from p_{X,θ_0}). The maximum likelihood estimator (MLE) $\hat{\theta}_{MLE}$ of θ is defined as

$$\hat{\theta}_{MLE} = \arg\max_{\theta\in\Theta} L_\theta \left(x_1, x_2, \ldots, x_T\right).$$

In the context of the ICA models (2.1.3) and (2.1.1), the parameter that we want to estimate is the vector $\mathbf{s}$ of the independent components, i.e. a matrix of its realizations $\mathbf{S}$ (see (2.1.1)), or (which is the same in the over-determined case) the mixing matrix $\mathbf{A}$ or its inverse $\mathbf{W} = \mathbf{A}^{-1}$. The matrix $\mathbf{X}$ in (2.1.1) represents exactly a random sample of observations of $\mathbf{x}$. Therefore, the likelihood function is (using the notation from Section 2.1)

$$L_{\mathbf{W}}(\mathbf{X}) = L_{\mathbf{W}}\left(\mathbf{x}_1, \ldots, \mathbf{x}_T\right) = \prod_{i=1}^{T} p_{\mathbf{x}}\left(\mathbf{x}_i\right).$$

P.d.f. $p_{\mathbf{x}}$ of a mixed vector $\mathbf{x}$ is obtained from the linear model $\mathbf{x} = \mathbf{A}\mathbf{s}$, using the independence of the components of $\mathbf{s}$, as

$$p_{\mathbf{x}}(\mathbf{x}) = \left|\det \mathbf{A}^{-1}\right| p_{\mathbf{s}}(\mathbf{s}) = \left|\det \mathbf{A}^{-1}\right| \prod_{i=1}^{N} p_{s_i}\left(s_i\right).$$

Therefore, by denoting the rows of $\mathbf{W} = \mathbf{A}^{-1}$ as $\mathbf{w}_i^T$, we have the expression for the likelihood of a sample $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_T]$

$$L_{\mathbf{W}}(\mathbf{X}) = \prod_{i=1}^{T}\prod_{j=1}^{N} p_{s_j}\left(\mathbf{w}_j^T \mathbf{x}_i\right) \left|\det \mathbf{W}\right|.$$

The average log-likelihood is

$$\hat{l}_{\mathbf{W}}(\mathbf{X}) = \frac{1}{T}\sum_{i=1}^{T}\sum_{j=1}^{N} \log p_{s_j}\left(\mathbf{w}_j^T \mathbf{x}_i\right) + \log\left|\det \mathbf{W}\right|. \tag{2.2.10}$$

The first term on the right in the expresion above is an estimate of $\mathbb{E}\left(\sum_{j=1}^{N} \log p_{s_j}\left(\mathbf{w}_j^T \mathbf{x}\right)\right)$ (a sample mean). Further, this expectation is exactly equal to $-\sum_{j=1}^{N} H_{p_{s_j}}\left(\mathbf{w}_j^T \mathbf{x}\right)$. On the other hand, from the definition of mutual information $I(\mathbf{W}\mathbf{x})$ we have $I(\mathbf{W}\mathbf{x}) = \sum_{j=1}^{N} H_{s_j}\left(\mathbf{w}_j^T \mathbf{x}\right) - H(\mathbf{x}) - \log\left|\det \mathbf{W}\right|$. Therefore, up to an additive constant (given by the entropy of the observation vector $\mathbf{x}$), the average log-likelihood is equal to the negative of the mutual information. This means that, *in theory*, ICA by the maximum likelihood principle (i.e. maximization of the likelihood) is equivalent to the minimization of mutual information.

2.2.4.2 Infomax

An estimation method for the ICA very similar to the maximum likelihood is the information maximization ('infomax') [9]. In the context of source separation and ICA, it consists in maximizing the entropy of the transformed variables $\mathbf{z} = g(\mathbf{y}) = g(\mathbf{W}\mathbf{x})$, where $g : \mathbb{R}^M \to \mathbb{R}^M$ is a componentwise ($(g(\mathbf{y}))_i = g_i(y_i)$) non-linear continuous function and $\mathbf{W}$ is an estimate of $\mathbf{A}^{-1}$. The functions g_i are chosen as the probability distribution functions of some p.d.f.-s, $g_i(s) = \int_{-\infty}^{s} q_i(\zeta)d\zeta$ (hence, g_i are invertible). Therefore, the contrast function for the infomax is $\Psi_I(\mathbf{W}) = H(g(\mathbf{Wx}))$, wherein the entropy is meant with respect to the *unknown* density of $g(\mathbf{Wx})$. This contrast can be written in another way, by using the following result.

Fact 2.1. *If a random vector* $\mathbf{x}$ *in* $\mathbb{R}^M$ *has distribution* $p_\mathbf{x}$ *with support* $[0,\,1]^M$*, then the entropy of* $\mathbf{x}$ *can be expressed as*

$$H(\mathbf{x}) = -\int_{\mathbb{R}^M} p_\mathbf{x}(\zeta)\log\frac{p_\mathbf{x}(\zeta)}{\prod_{i=1}^{M}\mathbf{1}_{[0,\,1]}(\zeta_i)}d\zeta = -D_{KL}(\mathbf{x},\mathbf{u}) \tag{2.2.11}$$

where $\mathbf{u} \sim \mathscr{U}\left([0,\,1]^M\right)$ *(*$\mathbf{u}$ *is distributed uniformly on* $[0,\,1]^M$*).*

Let us denote by $\tilde{\mathbf{s}}$ a random vector with probability distribution function g. The random vector $g(\tilde{\mathbf{s}})$ is distributed uniformly on $[0,\,1]^M$. It follows that

$$\begin{aligned}\Psi_I(\mathbf{W}) &= H(g(\mathbf{Wx})) \\ &= -D_{KL}(g(\mathbf{Wx}),\mathbf{u}) \\ &= -D_{KL}(g(\mathbf{Wx}),g(\tilde{\mathbf{s}})).\end{aligned}$$

Using the invariance of the Kullback-Leibler divergence to invertible transformations of its arguments (which follows by using the expressions for the p.d.f. of a transformation of a random variable), it follows that $\Psi_I(\mathbf{W}) = -D_{KL}(\mathbf{Wx},\tilde{\mathbf{s}})$ [19]. Therefore, the information maximization approach consists in estimating the transformation $\mathbf{Wx}$ of $\mathbf{x}$ whose probability distribution function is as close as possible to the pre-defined probability distribution function g. In the expression for Ψ_I above, the p.d.f. of $\mathbf{Wx}$ is unknown. On the contrary, in the maximum likelihood approach, the p.d.f. of $\mathbf{Wx}$ is considered known, equal to the true p.d.f. of the sources. It follows that, if g is chosen as the true probability distribution function of the sources, and if the sample mean in the definition of the average log-likelihood is replaced with the theoretical expectation, the maximum likelihood (in the sense of the maximization of this modified likelihood) and infomax contrast functions coincide.

2.2.4.3 FastICA algorithm

The special case of the entropy approximation of the type (2.2.9) is obtained for $n = 1$: $H_f \approx H(\nu) - \frac{1}{2}\mathbb{E}(G(X))$. This yields an approximation of the negentropy of the form

$$J_X \approx \mathbb{E}(G(X)). \tag{2.2.12}$$

Here, we have supposed that X has been *standardized* (however, the following derivations can also be adapted to the case when no standardization is performed). As already noted in the

Section 2.2.3.2, the optimal choice for G would be $-\log p_X$ if p_X were known. In practice, however, p_X is unknown. Note also that G must be orthogonal to all polynomials of degree 2 or less (see Section 2.2.3.2). For an odd or even G, after orthogonalizing it with respect to all polynomials of degree ≤ 2 and normalizing, the negentropy approximation (2.2.12) can be written as [53]

$$J_x \approx c\left[\mathbb{E}\left(G(X)\right) - \mathbb{E}\left(G(\nu)\right)\right]^2, \tag{2.2.13}$$

where c is an irrelevant constant and ν, as before, a gaussian variable with the same mean and variance as X. Using (2.2.4), it follows that the sum $\sum_{i=1}^{N} J(y_i) = \sum_{i=1}^{N} J\left((\mathbf{W}\mathbf{x})_i\right) = \sum_{i=1}^{N} J\left(\mathbf{w}_i^T\mathbf{x}\right)$ is a contrast function for the ICA (recall, the ICA mixing model is $\mathbf{x} = \mathbf{A}\mathbf{s}$). The two main approaches to estimating the ICA demixing matrix $\mathbf{W} = \mathbf{A}^{-1}$ are the deflationary and symmetric approaches. Firstly we focuss on the deflationary approach, which is based on estimating one independent component, i.e. one row of $\mathbf{W}$, at a time. We denote $J_G(\mathbf{w}) = \left[\mathbb{E}\left(G\left(\mathbf{w}^T\mathbf{x}\right)\right) - \mathbb{E}\left(G(\nu)\right)\right]^2$. The *one-unit FastICA functional* $\mathbf{w}_{G,1}$ is defined as

$$\mathbf{w}_{G,1} = \arg \max_{\mathbb{E}\left\{\left(\mathbf{w}^T\mathbf{x}\right)^2\right\}=1} J_G(\mathbf{w}). \tag{2.2.14}$$

Note that in the case of standardized x, the condition $\mathbb{E}\left\{\left(\mathbf{w}^T\mathbf{x}\right)^2\right\} = 1$ is equivalent to $\|\mathbf{w}\|_2 = 1$. The *k-th FastICA functional* $\mathbf{w}_{G,k}$ is defined as

$$\begin{array}{rcl} \mathbf{w}_{G,k} & = & \arg\max_{\mathbb{E}\left\{\left(\mathbf{w}^T\mathbf{x}\right)^2\right\}=1} J_G(\mathbf{w}) \\ & & \text{subject to} \quad \mathbf{w}^T\mathbf{w}_{G,i} = 0,\ i = 1,\ldots,k-1 \end{array}. \tag{2.2.15}$$

Namely, after estimating $k-1$ rows of the demixing matrix $\mathbf{W}$, the next row is constrained to be orthogonal to all previously estimated rows. Such a constraint is natural since independence implies uncorrelatedness (it is needed to avoid estimating the same row of $\mathbf{W}$ multiple times). Since the estimator of $\mathbf{W}$ is based on an approximation of negentropy, it is unclear whether it is a consistent estimator. The following theorem holds [53, 56].

Theorem 2.4. *Assume that the data follows the ICA model (2.1.3) and that G is a sufficiently smooth* even *function. Let us denote* $\mathbf{A}^{-1} = \mathbf{W}$ *and*
$J_G(\mathbf{w}) = \left[\mathbb{E}\left(G\left(\mathbf{w}^T\mathbf{x}\right)\right) - \mathbb{E}\left(G(\nu)\right)\right]^2$ *(where* $\mathbf{w}^T$ *denotes a row of* $\mathbf{W}$*). Then, the set of local maxima of* $J_G(\mathbf{w})$ *under the constraint* $\mathbb{E}\left(\left(\mathbf{w}^T\mathbf{x}\right)^2\right) = 1$ *includes the i-th row of the inverse of the mixing matrix* $\mathbf{A}$ *such that the corresponding independent component* s_i *satisfies*

$$\mathbb{E}\left\{s_i g(s_i) - g'(s_i)\right\}\left[\mathbb{E}\left(G(s_i)\right) - \mathbb{E}\left(G(\nu)\right)\right] > 0, \tag{2.2.16}$$

where g denotes the derivative of G and ν *is a standardized Gaussian variable.*

This theorem states the conditions under which the estimate of $\mathbf{W}$ based on (2.2.14) and (2.2.15) is consistent. For a sketch of the proof see (appendix A in [56]). The inequality (2.2.16) divides the space of probability distributions into two half-spaces, depending on whether the non-polynomial moments on the left side are (both) positive or negative. For example, if $G(x) = x^4$, the first term on the left is proportional to the kurtosis $\kappa(s_i)$ of s_i,

$$\mathbb{E}\left\{s_i g(s_i) - g_i'(s_i)\right\} = 4\left(\mathbb{E}\left(s_i^4\right) - 3\mathbb{E}\left(s_i^2\right)\right) = 4\left(\mathbb{E}\left(s_i^4\right) - 3\right),$$

the last equality following since we have supposed $\mathbb{E}\left(s_i^2\right) = 1$. In this case, the inequality (2.2.16) holds for all random variables of non-vanishing kurtosis. Therefore, in this case the FastICA estimator is consistent for most interesting distributions of the sources. We describe some other (good) choices for G later in this section.

Regarding the symmetric approach, analogous procedure can be used. Namely, we define a functional $\tilde{J}_{\tilde{G}}$ as

$$\tilde{J}_{\tilde{G}}(\mathbf{W}) = \mathbb{E}\left(\tilde{G}(\mathbf{Wx})\right),$$

wherein $\tilde{G}$ operates elementwise by applying G to every element of a vector. The negentropy maximization problem is now formulated as

$$W_G = \arg\max_{\mathbf{W}} \tilde{J}_{\tilde{G}}(\mathbf{W}) \quad \text{subject to} \quad \mathbf{W^T W} = \mathbf{I}.$$

The same consistency theorem (2.2.16) holds also in this case. See Paragraph 2.2.4.3.2 for details regarding the optimization method for this problem used in FastICA algorithm.

Some other properties of the estimator that we would like are the *efficiency* and *robustness*. The efficiency is related to the asymptotic variance of the estimator. Robustness means that a single, highly erroneous observation doesn't have large influence on the estimator. We review basic results related to the efficiency and (non)robustness of the FastICA estimator based on (2.2.14) and (2.2.15) in the following subsections. The theorems are listed in Appendix B. Paragraph 2.2.4.3.2 and Paragraph 2.2.4.3.3 are related to practical algorithms for computing the FastICA estimator.

2.2.4.3.1 Statistical properties of the FastICA estimator As already mentioned above, robustness of the estimator implies that a single erroneous observation (*outlier*) doesn't have a large influence on the estimator. A useful measure of robustness of an estimator is the influence function of the estimator, defined as follows (for the special case of the FastICA estimator).

Definition 2.11. The influence function of the FastICA estimator $\mathbf{w}_{G,k}$ in (2.2.15) at the distribution F, $IF_{\mathbf{w}_{G,k},F}$, is defined as

$$IF_{\mathbf{w}_{G,k},F}(z) = \lim_{\varepsilon\downarrow 0} \frac{\mathbf{w}_{G,k}(F_\varepsilon) - \mathbf{w}_{G,k}(F)}{\varepsilon} = \frac{\partial}{\partial\varepsilon}\mathbf{w}_{G,k}(F_\varepsilon)\,|_{\varepsilon=0}\,, \tag{2.2.17}$$

where $F_\varepsilon = (1-\varepsilon)F + \varepsilon\Delta_z$ denotes the *ε-contaminated distribution* F, and Δ_z denotes the degenerate distribution at z.

A robustness of the FastICA estimator can now be more formally defined as follows: an estimator is robust if it has a bounded and continuous influence function. Boundedness implies that a small amount of contamination of the distribution does not have an arbitrarily large influence on the estimator. Continuity implies that the small change in the data set results in the small change of the estimator. The expression for the influence function of the FastICA estimator is given in Theorem B.1 in Appendix B (this result is taken from [82]). It follows from this expression that the FastICA estimator is not robust, regardless of the choice of nonlinear function G. However, robustness can be controlled to some level through the choice of G, see Paragraph 2.2.4.3.3 for details.

Regarding the variance of the FastICA estimator and its theoretical lower bound, the theorems are given in Appendix B.2 (these results are taken from [83, 82]). The conclusions are the following. The accuracy of the k-th demixing vector $\hat{\mathbf{w}}_{G,k}$ depends on the distribution of the sources extracted previously. Also, the existence of kurtosis is required for all sources to ensure the existence of the asymptotic covariance matrix. Depending on the chosen non-linearity, the existence of even higher order moments might be required. Also, the case $c_{g,j} \approx \rho_{g,j}$, where $c_{g,j} = \operatorname{cov}\left(g(s_j), s_j\right) = \mathbb{E}\left(g(s_j)s_j\right)$ and $\rho_{g,j} = \mathbb{E}\left(g'(s_j)\right)$, for any $j \in \{1, \ldots, k\}$, leads to very high asymptotic variances of the estimator.

A modification of the algorithm for practical computation of the FastICA estimator (FastICA algorithm), reviewed in the following subsection, was presented in [59, 100]. It is asymptotically efficient in some cases. However, FastICA method, as presented in the following subsection and implemented in publicly available software (see Chapter 7), is good enough for the purposes of numerical experiments presented in Chapter 7. Therefore, we don't review this modification in more detail here.

2.2.4.3.2 FastICA optimization method After reviewing the theoretical properties of the estimators (2.2.14) and (2.2.15), we now consider the optimization method for finding these estimators in practice [53]. The deflationary mode of the FastICA algorithm uses the following approach. First we consider (2.2.14). KKT conditions imply

$$\mathbb{E}\left\{\mathbf{x}g\left(\mathbf{w}^T\mathbf{x}\right)\right\} - \beta\mathbf{w} = 0, \tag{2.2.18}$$

where β is a Lagrangian multiplier. The optimal multiplier is given by $\beta^* = \mathbb{E}\left\{\mathbf{w}^{*T}\mathbf{x}g\left(\mathbf{w}^{*T}\mathbf{x}\right)\right\}$, where $\mathbf{w}^*$ is the value of $\mathbf{w}$ at the optimum. FastICA uses the Newton method for solving (2.2.18). The Jacobian matrix of the left-hand side in (2.2.18) is $\mathbb{E}\left\{\mathbf{x}\mathbf{x}^T g'\left(\mathbf{w}^T\mathbf{x}\right)\right\} - \beta\mathbf{I}$. Now, the idea is to approximate this Jacobian by $\mathbb{E}\left\{\mathbf{x}\mathbf{x}^T\right\}\mathbb{E}\left\{g'\left(\mathbf{w}^T\mathbf{x}\right)\right\} - \beta\mathbf{I} = \left(\mathbb{E}\left\{g'\left(\mathbf{w}^T\mathbf{x}\right)\right\} - \beta\right)\mathbf{I}$ (here, $\mathbf{x}$ is standardized by assumption), which makes it diagonal. The approximate Newton iteration is

$$\mathbf{w} \leftarrow \mathbf{w} - \frac{\mathbb{E}\left\{\mathbf{x}g\left(\mathbf{w}^T\mathbf{x}\right)\right\} - \beta\mathbf{w}}{\mathbb{E}\{g'(\mathbf{w}^T\mathbf{x})\} - \beta}, \tag{2.2.19}$$

which can be rewritten as

$$\tilde{\mathbf{w}} \leftarrow \mathbb{E}\left\{\mathbf{x}g\left(\mathbf{w}^T\mathbf{x}\right)\right\} - \mathbb{E}\left\{g'\left(\mathbf{w}^T\mathbf{x}\right)\right\}\mathbf{w} \tag{2.2.20}$$

followed by the normalization $\mathbf{w} \leftarrow \tilde{\mathbf{w}}/\|\tilde{\mathbf{w}}\|_2$. To enable line search, which stabilizes the Newton iteration, the form (2.2.19) can be used, wherein β can be calculated from the current approximation $\mathbf{w}$ as $\beta = \mathbb{E}\left\{\mathbf{w}^T\mathbf{x}g\left(\mathbf{w}^T\mathbf{x}\right)\right\}$, again followed by the normalization $\mathbf{w} \leftarrow \mathbf{w}/\|\mathbf{w}\|_2$. Newton method can be used in the similar way for non-standardized data $\mathbf{x}$, see [53]. After estimating the first row of $\mathbf{W}$, i.e. the first independent component, the subsequent rows (solutions of (2.2.15)) can be estimated in a similar way. Namely, FastICA uses the same iteration (2.2.20) for all independent components, followed by the orthogonalization with respect to previously estimated rows of $\mathbf{W}$, and normalization. Namely, if we denote the current approximation of $\mathbf{w}_{G,k}$, for $k > 1$, by $\hat{\mathbf{w}}_{G,k}$, after every iteration

$$\tilde{\mathbf{w}}_{G,k} \leftarrow \mathbb{E}\left\{\mathbf{x}g\left(\hat{\mathbf{w}}_{G,k}^T\mathbf{x}\right)\right\} - \mathbb{E}\left\{g'\left(\hat{\mathbf{w}}_{G,k}^T\mathbf{x}\right)\right\}\hat{\mathbf{w}}_{G,k},$$

new approximation $\hat{\mathbf{w}}_{G,k}$ is obtained by

$$\begin{array}{rcl} \tilde{\mathbf{w}}_{G,k} & \leftarrow & \tilde{\mathbf{w}}_{G,k} - \sum_{i=1}^{k-1} \tilde{\mathbf{w}}_{G,k}^T \mathbf{w}_{G,i}\mathbf{w}_{G,i}, \\ \hat{\mathbf{w}}_{G,k} & \leftarrow & \frac{\tilde{\mathbf{w}}_{G,k}}{\|\tilde{\mathbf{w}}_{G,k}\|_2}. \end{array} \tag{2.2.21}$$

In the symmetric mode, the analogue of the iteration (2.2.20) for the whole matrix $\mathbf{W}$ can be used, followed by the *symmetric orthogonalization*. Namely, for a given approximation $\hat{\mathbf{W}}$ of the mixing matrix, $\hat{\mathbf{W}}$ can be orthogonalized by

$$\hat{\mathbf{W}} \leftarrow \left(\hat{\mathbf{W}}\hat{\mathbf{W}}^T\right)^{-\frac{1}{2}}\hat{\mathbf{W}}.$$

2.2.4.3.3 Choice of the nonlinear function in FastICA The non-linear function G used in the approximation of negentropy (2.2.12) should be chosen such that it approximates the negative logarithm of the true pdf(s) of the sources and/or it yields a robust estimate of the entropy. Namely, if robustness is important, G should not grow too fast, so that possible (large) outliers don't influence the negentropy approximation too much. Two good general choices for G are

$$G_1(x) = \frac{1}{a_1}\log\cosh(a_1x) \tag{2.2.22}$$

and

$$G_2(x) = -\frac{1}{a_2}\exp\left(-\frac{a_2x^2}{2}\right) \tag{2.2.23}$$

which correspond to

$$g_1(x) = \tanh(a_1x) \tag{2.2.24}$$

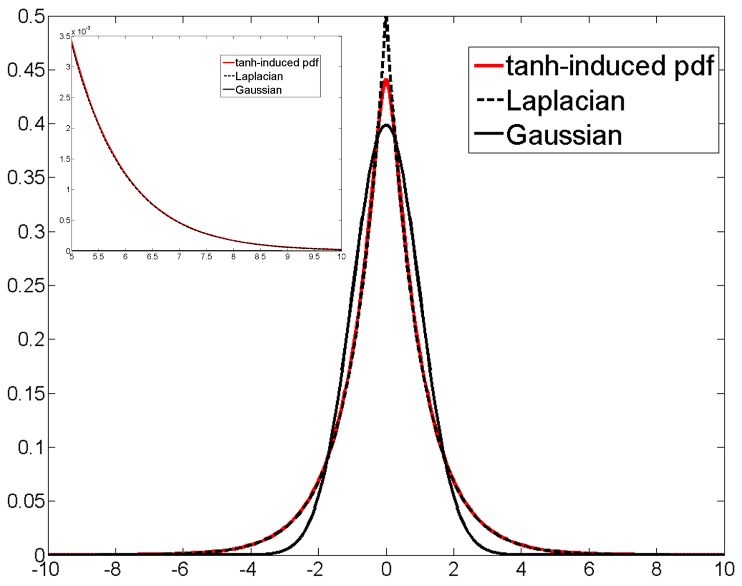


Figure 2.2.1: Comparison of the p.d.f. induced by using the nonlinearity (2.2.22) with the scale parameter $a_1 = 5$ in the FastICA algorithm, and generalized Gaussian p.d.f. with parameter $\beta = 1$ (see [34]), which models Laplacian p.d.f. $p(x) \sim \exp(-\lambda |x|)$. Also shown is Gaussian p.d.f. The zoomed part illustrates heavy tails of Laplacian and tanh-induced p.d.f.-s.

and

$$g_2(x) = x \exp\left(-\frac{a_2 x^2}{2}\right) \tag{2.2.25}$$

respectively.

Choice of G_1, i.e., g_1, implies that the sources are approximately distributed according to the Laplacian distribution, see Figure 2.2.1. Therefore, this choice implicitly assumes super-gaussian distributions for the sources. G_2, i.e. g_2, is also a good choice for super-gaussian distributed sources, and is faster to compute than g_1. Regarding the robustness, g_2 is also more robust than g_1 in the sense that the empirical influence function grows more slowly for g_2, see [82] for an example. In the numerical experiments in Chapter 7, we have used g_1 because empirically better results were obtained.

2.3 Under-determined case and the sparse component analysis (SCA)

When the number of source signals N exceeds the number of sensors M in the source separation mixture model (2.1.1), the assumption of independence alone is not enough to uniquely separate the signals (however, see Section 2.3.3 for some results on over-complete, i.e. under-determined ICA). As already discussed in the introduction of this chapter, the stronger assumption that we will consider here is *sparsity*. We will assume that the matrix $\mathbf{S}$ of source signals is sparse. Sparsity of a vector or a matrix is usually measured by the number of its nonzero elements, often referred to as the ℓ_0-quasi-norm. More precisely, the ℓ_0-quasi-norm of a vector $\mathbf{x}$, $\|\mathbf{x}\|_0$, is defined as

$$\|\mathbf{x}\|_0 = |\{i : x_i \neq 0\}|.$$

In practice, the assumption of sparsity of $\mathbf{S}$ is often satisfied directly or indirectly; i.e., if $\mathbf{S}$ is not already sparse, often one of the following two assumptions is realistic. Either there is a frame $\Phi \in \mathbb{R}^{K\times T}$ (often called the *dictionary*), where $K \geq T$, such that every row $\mathbf{s}^T \in \mathbb{R}^{1\times T}$ of $\mathbf{S}$ can be written as $\mathbf{s}^T = \mathbf{c_s}^T\Phi$, where $\mathbf{c_s} \in \mathbb{R}^{1\times K}$ is sparse (this is the *synthesis* view of sparsity of $\mathbf{S}$), i.e. $\mathbf{S}$ can be written as

$$\mathbf{S} = \mathbf{C_S}\Phi, \tag{2.3.1}$$

where the matrix $\mathbf{C_S}$ of coefficients is sparse; or, there is a linear transform $\Psi \in \mathbb{R}^{T\times K}$, $K \geq T$, such that the matrix of *analysis* coefficients

$$\mathbf{C}_{\mathbf{S},\Psi} = \mathbf{S}\Psi \tag{2.3.2}$$

is sparse (this is the analysis view of sparsity of $\mathbf{S}$). The problem with the synthesis formulation (2.3.1) is that, if the dictionary Φ is redundant, i.e. $K > T$, there is no analysis operator $\tilde{\Psi}$ such that $\Phi\tilde{\Psi} = \mathbf{I}$, i.e. the source signals generally can not be transformed so that the resulting coefficients coincide with the synthesis coefficients $\mathbf{C_S}$. Therefore, the model $\mathbf{C_X}\Phi = \mathbf{X} = \mathbf{AC_S}\Phi$ is generally not equivalent to $\mathbf{C_X} = \mathbf{AC_S}$, i.e. the original source separation problem is not easily transformed to a problem with sparse sources. On the contrary, if there is an analysis operator Ψ such that the analysis model (2.3.2) is valid, we have

$$\mathbf{C}_{\mathbf{X},\Psi} = \mathbf{X}\Psi = \mathbf{AS}\Psi = \mathbf{AC}_{\mathbf{S},\Psi}. \tag{2.3.3}$$

Therefore, by transforming the mixtures using the analysis operator Ψ, we obtain a mixture model with sparse sources. The same is true in the case when the dictionary Φ is a basis or a full rank matrix in the case $K < T$, since we can use $\Phi^\dagger$ as the analysis operator ($\Phi^\dagger$ denotes the pseudoinverse of Φ). Here we suppose that the analysis model is valid. Note that the sources $\mathbf{S}$ can be recovered from their analysis coefficients $\mathbf{C}_{\mathbf{S},\Psi}$ as $\mathbf{S} = \mathbf{C}_{\mathbf{S},\Psi}\Psi^\dagger$.

Among various methods for under-determined source separation, we concentrate on sparse component analysis methods. Namely, the approach for solving/approximating the solution of the problem (2.3.3) is two-stage: firstly, mixing matrix is estimated using the geometric methods based on *clustering*; secondly, the sources (or their analysis coefficients) are estimated using sparse recovery methods with *known* mixing matrix. See chapter 10 in [22] for some references to other (non-geometric) methods for SCA.

The rest of this section is organized as follows. In Section 2.3.1 and Section 2.3.2, we review mixing matrix estimation and basics of sparse recovery, respectively. Methods for sparse recovery are reviewed in Chapter 4. In Section 2.3.3, methods for under-determined ICA are briefly discussed, with an emphasis on the extension of FastICA algorithm to under-determined case. We will use the basic notation $\mathbf{X} = \mathbf{AS}$ and, without loss of generality (based on the discussion above), assume that the sources $\mathbf{S}$ are sparse.

2.3.1 Mixing matrix estimation

We assume that the data satisfy the model (2.1.1), wherein the matrix of source signals $\mathbf{S}$ is sparse. More precisely, we assume that every column of $\mathbf{S}$ has at most $k < M$ non-zero elements. Therefore, every column of $\mathbf{X}$ can be expressed as a linear combination of k mixing vectors (columns of $\mathbf{A}$), which means that every column of $\mathbf{X}$ belongs to a linear subspace spanned by some k mixing vectors. In the case $k = 1$, every column of $\mathbf{X}$ is proportional to one of the mixing vectors. This is the much simpler case, since mixing vectors can be estimated using *clustering* methods [38], like k-means or hierarchical clustering. Namely, if the distance between points (vectors) is defined as the angle between them, the problem reduces to classical clustering problem. Although many clustering methods are very sensitive to initialization because of the non-convex nature of the clustering problem, this is the simpler case for mixing matrix estimation. More realistic and harder case is when $k > 1$. In this case, subspace clustering methods [38, 106] can be used. Namely, linear subspaces play the role of cluster centers (points/vectors in the $k = 1$ case). Here, we will only briefly review one practical method, which was used in the experiments presented in the Chapter 7, Section 7.3.

Many methods [67, 91] for mixing matrix estimation in under-determined source separation are based on the assumption that there are points (columns of $\mathbf{X}$) such that the corresponding column of $\mathbf{S}$ has only one nonzero element. If such points were known, mixing vectors corresponding to those nonzero elements can easily be identified (they are proportional to these *single-source points* (SSP-s)). If there is at least one such point for every mixing vector, the whole mixing matrix (as before, up to scaling and permutation of its columns) can be estimated. Therefore, the whole problem is to find these single-source points. The method in [91] is based on the following idea. Suppose that the sources are complex. Let $\mathbf{x}_t$ be a single-source point (column of $\mathbf{X}$), i.e. suppose that $\mathbf{x}_t = S_{i,t}\mathbf{a}_i$, where $S_{i,t} \in \mathbb{C}$ and $\mathbf{a}_i$ is a (real) mixing vector. Then, the real and imaginary part of $\mathbf{x}_t$ point in the same direction, $\mathbf{a}_i$. On the other hand, if a point $\mathbf{x}_{t'}$ can be expressed as a linear combination of two or more mixing vectors, say $\mathbf{x}_{t'} = S_{i_1t'}\mathbf{a}_{i_1} + S_{i_2t'}\mathbf{a}_{i_2}$, then the real and imaginary parts of $\mathbf{x}_{t'}$ would point in the same direction if and only if

$$\frac{\operatorname{real}\left(S_{i_1t'}\right)}{\operatorname{imag}\left(S_{i_1t'}\right)} = \frac{\operatorname{real}\left(S_{i_2t'}\right)}{\operatorname{imag}\left(S_{i_2t'}\right)}, \tag{2.3.4}$$

where $\operatorname{real}(\cdot)$ and $\operatorname{imag}(\cdot)$ denote the real and imaginary part of a complex number. Generally, we can consider the probability that (2.3.4) holds for a randomly chosen column of $\mathbf{S}$ (i.e., $\mathbf{X}$) negligible (even more when $\mathbf{x}_{t'}$ is a linear combination of more mixing vectors). Therefore, in the noiseless case ($\mathbf{X} = \mathbf{AS}$), we can, with high probability, detect single-source points as those for which the angle between the real and imaginary part is equal to zero. After selecting the single-source points, mixing vectors can be found by clustering the set of these points. This set is expected to contain much less points than the whole matrix $\mathbf{X}$, which makes classical clustering methods applicable. If the data are not complex, as we have supposed, they can be transformed to the complex domain (for example, using the Fourier transformation). Namely, for a linear transformation $\mathbf{F}$ (a regular matrix), $\mathbf{X} = \mathbf{AS}$ is equivalent to $\mathbf{XF} = \mathbf{ASF}$ ($\mathbf{F}$ operates from the right since it transforms the rows of $\mathbf{S}$ and $\mathbf{X}$). Now, $\mathbf{A}$ can be estimated using the above idea *if enough SSP-s can be found*. Of course, the transformed signals might not be sparse, especially if the original signals are. Nevertheless, due to the simplicity of the condition (2.3.4), this method can always be tried and used if, as said already, enough SSP-s in the transformed domain can be found. In the case when some noise is present and/or $\mathbf{S}$ is only approximately sparse, there are no exact single-source points. However, approximate single-source points, i.e. those at which some coefficient is significant while others are close to zero, can be selected by constraining the angle between real and imaginary part of the point to be smaller than some pre-specified tolerance. The tolerance depends on the assumed level of sparsity of $\mathbf{S}$ and the level of noise. *Due to its simplicity*, this method was used in the experiments presented in Chapter 7.

Regarding the theoretical conditions under which the mixing matrix can be uniquely determined from the mixtures, we mention the paper [42]. However, condition presented there requires that the number T of observations grows exponentially with k and also doesn't present a practical algorithm for estimating the mixing matrix. Therefore, it remains mainly theoretical. Some more useful results are presented in more recent papers [44, 41].

2.3.2 Sparse recovery

After estimating the mixing matrix, the source separation problem reduces to the problem of finding sparse $\hat{\mathbf{S}}$ such that $\mathbf{X} = \hat{\mathbf{A}}\hat{\mathbf{S}}$, where $\hat{\mathbf{A}}$ denotes the estimated mixing matrix. If the matrix of true source signals $\mathbf{S}$ is sparse enough, it can be uniquely identified from the mixtures if the true mixing matrix $\mathbf{A}$ has been correctly estimated. More precisely, the equation $\mathbf{X} = \hat{\mathbf{A}}\hat{\mathbf{S}}$ consists of T linear systems $\mathbf{x}_t = \hat{\mathbf{A}}\hat{\mathbf{s}}_t$. The basic assumption on $\mathbf{A}$ will be that every $M \times M$ sub-matrix of $\mathbf{A}$ is regular. This is a realistic assumption that is satisfied with probability one for random $\mathbf{A}$. This condition can be expressed in terms of the *spark* of the matrix, defined as follows (most of the following results can be found for example in [16]).

Definition 2.12. The spark of a $M \times N$ matrix $\mathbf{A}$, $\operatorname{spark}(\mathbf{A})$, where $N \geq M$, is the smallest number k such that there are k columns of $\mathbf{A}$ that are linearly dependent.

Therefore, we assume that the spark of $\mathbf{A}$ is equal to $M+1$. The following proposition holds regarding the uniqueness of the sparse solution of the under-determined linear system.

Proposition 2.2. *If a system of linear equations* $\mathbf{As} = \mathbf{x}$*, where* $\mathbf{A} \in \mathbb{R}^{M \times N}$ *and* $N \geq M$*, has a solution* $\mathbf{s}$ *that satisfies* $\|\mathbf{s}\|_0 < \frac{spark(\mathbf{A})}{2}$*, where* $\|\mathbf{s}\|_0 = |\{i : s_i \neq 0\}|$*, then this solution is the sparsest among all solutions of the system.*

Therefore, based on this proposition, $\mathbf{S}$ can be uniquely identified from the mixtures $\mathbf{X} = \mathbf{AS}$ if $\|\mathbf{s}_t\|_0 < \frac{\text{spark}(\mathbf{A})}{2}$, for every $t,\ 1 \leq t \leq T$. If $\text{spark}(\mathbf{A}) = M+1$, as assumed, then every column $\mathbf{s}_t$ of $\mathbf{S}$ should be $\lfloor \frac{M}{2} \rfloor$-sparse, i.e. $\|\mathbf{s}_t\|_0 < \frac{M+1}{2}$, to guarantee the uniqueness of the solution. In other words, to ensure the uniqueness, the number of mixtures M should generally be at least twice the maximal sparsity (measured by the ℓ_0-quasi-norm) of any column of $\mathbf{S}$.

Uniqueness condition can also be expressed in terms of the *coherence* of a matrix, defined as follows.

Definition 2.13. The mutual coherence, or coherence, of a matrix $\mathbf{A} \in \mathbb{R}^{M \times N}$, $N \geq M$, denoted as $\mu(\mathbf{A})$, is defined as the maximal absolute correlation between its columns, i.e.

$$\mu(\mathbf{A}) = \max_{i \neq j} \left| \frac{\mathbf{a}_i^T \mathbf{a}_j}{\|\mathbf{a}_i\|_2 \|\mathbf{a}_j\|_2} \right|,$$

where $\mathbf{a}_i$ denotes i-th column of $\mathbf{A}$.

The conection between coherence and spark is detailed in the following theorem [16].

Theorem 2.5. *For* $\mathbf{A} \in \mathbb{R}^{M \times N}$, $N \geq M$*, the following holds:*

$$spark(\mathbf{A}) \geq 1 + \frac{1}{\mu(\mathbf{A})}. \tag{2.3.5}$$

The proof can be found in [16].

The connection between the spark and coherence can be used to obtain new condition on sparsity of $\mathbf{s}$ in the Proposition 2.2, $\|\mathbf{s}\|_0 < \frac{1}{2}\left(1 + \frac{1}{\mu}\right)$. However, since (2.3.5) only gives a (not very sharp) lower bound on the spark, the condition using coherence is much more restrictive, which makes it not very useful in practice. Nevertheless, coherence is easily computed, while the spark is very hard to compute. Coherence can give a rough prediction of degree of sparsity sufficient to guarantee the uniqueness of sparse solution.

The problem with ℓ_0-quasi-norm, $\|\cdot\|_0$, is that it results in discrete optimization problems since it is not a continuous function. Namely, we can formally formulate the problem of source separation with estimated mixing matrix $\hat{A}$ as an optimization problem

$$\min_{\mathbf{S}} \|\mathbf{S}\|_0 \quad \text{subject to} \quad \mathbf{X} = \hat{\mathbf{A}}\mathbf{S}, \tag{2.3.6}$$

where ℓ_0-quasi-norm of a matrix is defined analogously to vectors. (2.3.6) can also be written as simultaneous T linear systems with the same matrix

$$\min_{\mathbf{s}_t} \|\mathbf{s}_t\|_0 \quad \text{subject to} \quad \mathbf{x}_t = \hat{\mathbf{A}}\mathbf{s}_t,\ t = 1, \ldots, T. \tag{2.3.7}$$

Every sub-problem in (2.3.7), for fixed t, is a combinatorial optimization problem, which makes it impractical to solve. There are basically two approaches to solve (2.3.7) (or its noisy version): direct and indirect. Direct methods include greedy algorithms and iterative hard thresholding, see Chapter 4 for more details. Indirect methods consider continuous and/or smooth approximations of $\|\cdot\|_0$. We discuss them in Chapter 4. There, we also discuss conditions under which the solution of (2.3.7) can be found using these algorithms.

2.3.3 Overcomplete ICA

The overcomplete ICA model in the noiseless case is formulated as follows: we assume that the observation vector $\mathbf{x} \in \mathbb{R}^M$ can be expressed as

$$\mathbf{x} = \mathbf{A}\mathbf{s}, \tag{2.3.8}$$

where $\mathbf{A}$ is a $M \times N$ mixing matrix with $N > M$, and $\mathbf{s} \in \mathbb{R}^N$ is a random vector with independent components. In the noisy case, the model is the following:

$$\mathbf{x} = \mathbf{A}\mathbf{s} + \mathbf{b}, \tag{2.3.9}$$

where $\mathbf{b}$ is an $M \times 1$ random vector that has either Gaussian components, independent components or is of small variance. We assume that $\text{spark}(\mathbf{A}) \geq 3$, i.e. no two columns of $\mathbf{A}$ are colinear. $\mathbf{A}$ is obviously not an invertible matrix, so in this case $\mathbf{s}$ can not be obtained by linear transformation of observations $\mathbf{x}$. We introduce the following definition (chapter 9 in [22]).

Definition 2.14. The (noiseless) representation of $\mathbf{x}$ is a pair $(\mathbf{A}, \mathbf{s})$ such that the model (2.3.8) is valid. Assume that $\mathbf{x}$ admits two noiseless representations

$$\mathbf{x} = \mathbf{A}\mathbf{s} \text{ and}$$
$$\mathbf{x} = \mathbf{B}\mathbf{z},$$

where $\mathbf{s}$ and $\mathbf{z}$ have independent components. The above two representations of $\mathbf{x}$ are equivalent if every column of $\mathbf{A}$ is proportional to some column of $\mathbf{B}$, and vice versa.

If all representations of $\mathbf{x}$ are equivalent, they are said to be essentially unique, i.e. equal up to permutation and scaling.

We are interested in conditions under which all representations of $\mathbf{x}$ are essentially unique. The following theorem holds [22].

Theorem 2.6. *Let $\mathbf{x}$ be of the form $\mathbf{x} = \mathbf{A}\mathbf{s}$, where $\mathbf{s}$ has independent components and $\mathbf{A}$ has no colinear columns. Then $\mathbf{s}$ can be represented as*

$$\mathbf{x} = \mathbf{A}_1\mathbf{s}_1 + \mathbf{A}_2\mathbf{s}_2,$$

where $\mathbf{s}_1$ is non-Gaussian, $\mathbf{s}_2$ is Gaussian and independent of $\mathbf{s}_1$, and $\mathbf{A}_1$ is essentially unique.

This theorem is quite general, and doesn't say anything about the uniqueness of $\mathbf{s}_1$ and $\mathbf{s}_2$. Namely, the distribution of $\mathbf{s}_1$ is generally not unique. Now we introduce the following important definitions [31].

Definition 2.15. The ICA model (2.3.8) is

1. *identifiable*, if all representations of $\mathbf{x}$ are essentially unique,
2. *unique*, if it is identifiable and the distributions of source vectors $\mathbf{s}$ and $\mathbf{r}$ in two different representations of $\mathbf{x}$ are the same for some permutation, up to changes in location and scale, and
3. *separable*, if for every full row rank matrix $\mathbf{W}$ such that $\mathbf{Wx}$ has independent components, $\Lambda\mathbf{Ps} = \mathbf{Wx}$, for some diagonal matrix Λ and permutation matrix $\mathbf{P}$.

Of course, an overcomplete model can not be separable. Namely, the following theorem holds [31].

Theorem 2.7. *The linear ICA model* $\mathbf{x} = \mathbf{As}$ *is separable if and only if the mixing matrix* $\mathbf{A}$ *is of full column rank and at most one source variable is Gaussian.*

The proof is presented in [31]. However, identifiability and uniqueness hold under some (mild) conditions. Namely, the following theorem, which follows from the above Theorem 2.6, holds.

Theorem 2.8. *The model (2.3.8) is identifiable if all source variables are non-Gaussian.*

This theorem shows that if the number of sources is greater than the number of mixtures ($N > M$), it can still be possible to identify the mixing matrix from mixtures alone. However, nothing is said about the recovery of the sources, i.e. about the *uniqueness* of the model. The conditions for uniqueness are stated in the following theorem [31].

Theorem 2.9. *The over-complete ICA model (2.3.8), with the assumption that no two columns of* $\mathbf{A}$ *are colinear, is unique if any of the following properties hold:*

1. *All characteristic functions (c.f.-s) of source variables are analytic (or all c.f.-s are non-vanishing), and none of the c.f.-s has an exponential factor with a polynomial of degree at least* 2*.*
2. *All source variables are non-Gaussian with non-vanishing c.f.-s, and* $\mathrm{rank}(\mathbf{A} \odot \mathbf{A}) = N$*, where* $\odot$ *is defined as* $\mathbf{A} \odot \mathbf{B} = [\mathbf{a}_1 \otimes \mathbf{b}_1 \ \ldots \ \mathbf{a}_N \otimes \mathbf{b}_N]$*, where* $\mathbf{a}_i$ *and* $\mathbf{b}_i$ *are i-th columns of* $\mathbf{A}$ *and* $\mathbf{B}$*, respectively, and* $\otimes$ *denotes Kronecker product.*
3. *All source variables have non-vanishing c.f.-s without exponential factors with a polynomial of degree* n*,* $1 < n \leq q$*, and* $\mathrm{rank}((\mathbf{A}\odot)^q \mathbf{A}) = N > \mathrm{rank}\left((\mathbf{A}\odot)^{q-1}\mathbf{A}\right)$*, where* $(\mathbf{A}\odot)^q \mathbf{A} = \mathbf{A} \odot \cdots \odot \mathbf{A}$ *(includes* q *times* $\odot$*).*

The sketch of the proof and references can be found in [31]. Analytic characteristic functions are exactly those for which moment-generating function exists. The second condition in (1) is equivalent to the assumption that none of the source variables has a Gaussian component. Note that the number of sources in (1) is *unlimited*. The number of sources is limited by the maximal number of linearly independent rows of the matrix $\mathbf{A} \odot \mathbf{A}$ in (2). However, (2) has weaker assumptions on source distributions, as well as (3). (3) has more stringent assumptions on source distributions compared to (2), however maximal number of sources for which uniqueness can still hold is larger.

The above theorem shows that even the overcomplete ICA is a well posed problem under some assumptions on source distributions (and mixing matrix). However, algorithms for source separation in over-complete (i.e. under-determined) case seem rare in the literature. There are algorithms for mixing matrix *identification* (see chapter 9 in [22]). Some algorithms for over-complete ICA have been presented, for example a Bayesian approach in [65]. In the experiments in Chapter 7 we have used a simple modification of the FastICA algorithm presented in [54] that can be viewed as an approximative method for solving the overcomplete ICA problem. Namely, the only difference compared to the FastICA algorithm as presented in Section 2.2.4.3 is in the step (2.2.21). There, instead of orthogonalization with respect to the previously estimated rows of the separating matrix $\mathbf{W}$, *quasi-orthogonalization* is performed. Namely, since in the over-complete case the separating matrix $\mathbf{W}$ has more rows than columns, its rows can be made only *approximately orthogonal*. Consequently, the step (2.2.21) is replaced with

$$\begin{aligned} \tilde{\mathbf{w}}_{G,k} &\leftarrow \tilde{\mathbf{w}}_{G,k} - \alpha \textstyle\sum_{i=1}^{k-1} \tilde{\mathbf{w}}_{G,k}^T \mathbf{w}_{G,i} \mathbf{w}_{G,i}, \\ \hat{\mathbf{w}}_{G,k} &\leftarrow \frac{\tilde{\mathbf{w}}_{G,k}}{\|\tilde{\mathbf{w}}_{G,k}\|_2}, \end{aligned} \tag{2.3.10}$$

where $0 < \alpha < 1$. α should generally depend on the dimension of $\tilde{\mathbf{w}}_{G,k}$, since for very large dimensions it is possible to have large number of vectors (more than the dimension of vector space) with angles between them arbitrarily close to 90 degrees. On the contrary, in low-dimensional vector spaces one can not have many mutually approximately orthogonal vectors. It was shown (see references in [16]) that, for m vectors in $\mathbb{R}^n$, $m > n$, maximal angle γ between them satisfies

$$\cos\gamma \geq \sqrt{\frac{m-n}{n(m-1)}}.$$

Therefore, α could be selected from n and m, such that it takes into account maximal possible angle between vectors. However, we have chosen α heuristically, which can already give satisfactory practical results, see Chapter 7 for details. It should be noted that this method is only a very rough approximation of quasi-orthogonalization.

2.4 Summary

In this chapter we have presented basic theoretical results on independent component analysis as a method for solving the source separation problem. We have reviewed the FastICA algorithm as a practical algorithm for the independent component analysis. Both the theoretical properties and the numerical implementation of FastICA have been presented. All these are well known results from the literature. FastICA was used in the experiments in image inpainting and denoising presented in Chapter 7. The connection between the independent component analysis and dictionary learning concept that is important for the experiments in Chapter 7 is discussed in the next chapter.

Chapter 3

Linear model of sparse coding

Most of the previous chapter was concentrated around the concept of statistical independence of source signals in the linear mixture models (2.1.1) and (2.1.3). However, as discussed in Section 2.3, the assumption of independence alone is not enough to recover the sources in the underdetermined case. Stronger assumption on the sources is their *sparsity*. We have introduced the notion of sparsity, as well as a measure of sparsity, the ℓ_0 function, in Section 2.3. Both the independence and sparsity were used in the context of source separation, as the assumptions on the unknown sources from which the mixtures were obtained. Here, we consider the somehow different problem. Namely, we suppose that we are given a sample of some data of interest. We call this sample the *training set*. As discussed in the introduction of the Chapter 2, we suppose we are given the data matrix $\mathbf{X} \in \mathbb{R}^{M \times T}$, $T \gg M$, in which every column $\mathbf{x}_t$ is viewed as a realization of a random vector $\mathbf{x}$ in $\mathbb{R}^M$. Again we will suppose that the linear model (2.1.2) holds, but we will assume that the unknown sources (viewed as random vectors) are both mutually independent and have 'sparse distributions' (we will define sparse distributions shortly). Therefore, we *prescribe desired distributions* (and also mutual independence) *on the sources* and search for the mixing matrix $\mathbf{A} \in \mathbb{R}^{M \times M}$, and matrix of source realizations $\mathbf{S} \in \mathbb{R}^{M \times T}$ such that the mixing model (2.1.2) holds, where the error matrix $\mathbf{E}$ is of as small norm as possible. Note that we have supposed that $M = N$, i.e. $\mathbf{A}$ is a square matrix. Sparse distributions are informally defined as distributions that have *high peaks at zero* (higher than Gaussian distribution) and *heavy tails* (heavier than Gaussian). Therefore, a vector of realizations of a random variable with sparse distribution will have many elements close to zero and few large elements. This means that sparse distributions can be used to model approximately sparse vectors in the sense of ℓ_0. If the original data in $\mathbf{X}$ approximately satisfy the linear model $\mathbf{X} = \mathbf{AS}$ with mutually independent and sparsely-distributed sources, the resulting ICA problem with sparse distributions imposed on the sources is well posed.

The idea of expressing the data in the matrix $\mathbf{X}$ as linear mixtures of sparsely-distributed sources is motivated by inverse problems in signal and image processing. The general inverse problem in signal or image processing can be presented as follows: an observation vector $\mathbf{y} \in \mathbb{R}^m$ is obtained as

$$\mathbf{y} = \mathbf{H}\mathbf{x} + \mathbf{n}, \tag{3.0.1}$$

where $\mathbf{H} \in \mathbb{R}^{m \times M}$, $M \geq m$, is a non-invertible or ill-conditioned operator (‘degradation operator’), $\mathbf{x} \in \mathbb{R}^M$ is the original signal or image that we want to recover, and $\mathbf{n} \in \mathbb{R}^m$ represents an error. We restrict ourselves to the case $M > m$. Since $\mathbf{H}$ is not invertible, the problem (3.0.1) is ill-posed, and some regularization is needed to make it well-posed. An effective regularization is introduced by assuming that $\mathbf{x}$ can be written as a linear mixture $\mathbf{x} = \mathbf{A}\mathbf{s} + \mathbf{e}$, where $\mathbf{A} \in \mathbb{R}^{M \times N}$, $N \geq M$, and $\mathbf{s} \in \mathbb{R}^N$ is sparse [16]. The problem (3.0.1) can then be written as

$$\mathbf{y} = \mathbf{H}\mathbf{A}\mathbf{s} + (\mathbf{H}\mathbf{e} + \mathbf{n}). \tag{3.0.2}$$

We can assume that the term $\mathbf{H}\mathbf{e} + \mathbf{n}$, where $\mathbf{e}$ represents an error in the representation $\mathbf{x} = \mathbf{A}\mathbf{s} + \mathbf{e}$ of $\mathbf{x}$, and $\mathbf{n}$ represents some noise in the formulation (3.0.1), is small, which is true if $\mathbf{H}\mathbf{e}$ is small. Therefore, the measurement $\mathbf{y}$ can be represented as a linear mixture of sparse ‘sources’ $\mathbf{s}$. The resulting model (3.0.2) reduces to a sparsity-constrained recovery problem. Many inverse problems in signal and image processing, like denoising, deconvolution (deblurring) or inpainting, have been successfully approached using the sparsity-based regularization. Since the main application interest of this thesis is in some image processing problems (more precisely, inpainting and removal of salt-and-pepper noise), in Section 3.1 we describe additional motivation for the use of sparsity-based regularization for image processing.

From the above discussion it is seen that ICA can be used as a tool for designing the matrix $\mathbf{A}$ that induces a sparse representation of a given clas of signals, whose representative set (training set) is stored in matrix $\mathbf{X}$. The obtained matrix $\mathbf{A}$ can then be used in inverse problems of the form (3.0.1) for signals in the given class in the way described in the previous paragraph. The use of independent component analysis in this context is discussed in more detail in Section 3.2.

3.1 Biological foundation of sparse coding of images

In this section we briefly review main results on computational models presented in the literature that tried to explain the experimentally observed behaviour of the part of the brain of mammals that is responsible for processing of visual information (primary visual cortex).

It has been shown in the early studies [51] that the neurons in the primary visual cortex respond to localised oriented edges in visual scenes. Several computational models of visual cortex have been presented that try to explain the observed behaviour. The basic underlying principle for these models is the principle of efficient coding (see introduction of the paper [90]). Namely, it is natural to assume that the organism is adapted in order to maximize the efficiency of information processing. It was suggested in the literature that efficient visual neuronal coding means that neurons are sensitive to *independent elements that constitute an image* (again, see introduction of the paper [90] for some references). If the number of independent elements in an image or its part is small, this would be reflected in the small number of active neurons, which means that the information processing is efficient. The first computational model in the literature that (*at least partially*) explained the experimentally observed data was presented in [84]. This model constrains the representations of images to be sparse, and is able to learn the so called receptive fields that *resemble* the ones observed experimentally. The term receptive field refers to a part of the visual scene that activates the given neuron. The model presented in that paper is referred to as *Sparsenet*. The second approach suggested in the literature to reproduce the experimentally observed receptive fields uses ICA [10] (more precisely, the infomax algorithm, see Section 2.2.4.2 in Chapter 2). This result agrees with the intuition that the individual neurons are sensitive to independent elements (components) in an image. It was implicitly assumed in [10], through the choice of the non-linear function used in the infomax algorithm, that the distributions of underlying independent components are super-Gaussian (highly peaked at zero with heavy tails, i.e. sparse). Our approach to dictionary learning for inpainting and removal of impulsive noise, presented in Chapter 5 and Chapter 7, is based on this result. The approach in [10] yields receptive fields with very similar shapes to those obtained with Sparsenet, despite the different notion of sparseness used. Namely, the sparseness was assumed implicitly in [10] through the choice of super-Gaussian distributions, while in [84] explicit measures of sparsity were used. More precisely, they used functions of the form $F(x) = -\sum_i f\left(\frac{x_i}{\sigma}\right)$, where $f(u)$ is defined as $-\exp\left(-u^2\right)$, $\log\left(1+u^2\right)$ or $|u|$, all of which are measures of sparsity.

It should be noted that the above computational models produce receptive fields that have *some* similarities with the experimentally observed ones. Other models have been presented in the literature since the publication of the above mentioned seminal papers, that produce receptive fields that *better* predict diverse shapes of receptive fields that occur in nature (see [90]). However, the model that uses ICA already enables very good results in image inpainting and (impulse-noise-) denoising applications, see Chapter 7. It should also be mentioned that several other sparseness-based methods yielded receptive fields similar to those obtained by ICA and Sparsenet. For example, non-negative matrix factorization (NMF) with sparseness constraints was used in [49]. The results on image inpainting experiments reported in Chapter 7 using this method were inferior to those obtained with ICA. Therefore, we don't review this method in more detail here.

The training sets used in both seminal papers [84] and [10] consisted of many small *patches* extracted from images of natural scenes. See Figure 3.1.1.

Obtained dictionary elements ('receptive fields') look similar to those shown in Figure 7.1.2 in Chapter 7. It is important to note that the obtained receptive fields using any of the two methods are qualitatively similar regardless of the specific training set of natural images and the specific patches extracted. We review details regarding the use of ICA for sparse coding in the following section.

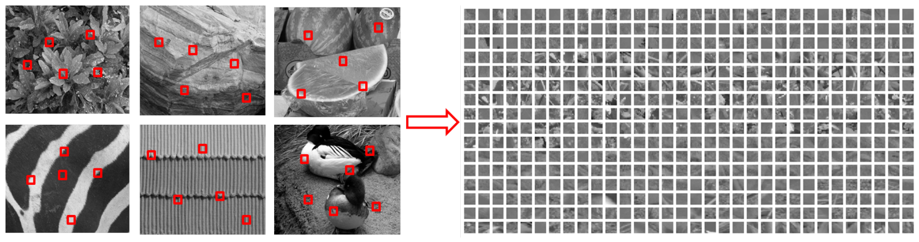


Figure 3.1.1: Extraction of small, quadratic parts of images (image patches)

3.2 The use of ICA in the context of sparse coding

The principle that underlies computational models of vision is the principle of efficient coding. Namely, it is assumed that individual neurons are sensitive to independent elements that constitute an image, and that the number of independent elements in a scene is usually small. This should therefore be reflected in a small number of active neurons. These assumptions suggest that ICA can be used to extract individual independent components from a large database of natural images. This was the approach used in the seminal paper [10]. Specifically, in that paper the infomax algorithm was used with the *sigmoid function* (or *logistic* function), i.e. $g(x) = (1+\exp(-x))^{-1}$ (see Section 2.2.4.2 in Chapter 2). The logistic function is a distribution function that has *heavier tails (*higher kurtosis*)* than the normal distribution, therefore it is a good model of sparse distribution. By using this function in the infomax algorithm, the independent components (sources) are implicitly assumed to be sparse (again, see the discussion on infomax algorithm in Section 2.2.4.2 in Chapter 2). In the context of the linear model $\mathbf{X} = \mathbf{AS}$, the matrix $\mathbf{X}$ represents a training set with individual image patches (after column-wise vectorization) as columns, and $\mathbf{S}$ represents the independent components of the mixtures $\mathbf{X}$, wherein rows of $\mathbf{X}$ are viewed as linear mixtures. The columns of the mixing matrix $\mathbf{A}$ represent vectorized receptive fields. See Figure 7.1.2 in Chapter 7. In the context of sparse coding, the mixing matrix $\mathbf{A}$ is referred to as the dictionary. The assumption of sparsity of the independent components in $\mathbf{S}$, introduced implicitly through the choice of the non-linear function in the ICA algorithm, means that there are only few significant receptive fields (columns of $\mathbf{A}$) for a given image patch (column of $\mathbf{X}$), i.e. only few coefficients in the corresponding column of $\mathbf{S}$ are significant. Although in the mentioned seminal paper [10] the infomax algorithm was used, any ICA algorithm can be used, for example FastICA. In Paragraph 2.2.4.3.3 two choices for the nonlinear function in the FastICA algorithm were presented that induce super-gaussian sources and therefore can be used for sparse coding.

3.3 Summary

In this chapter we have discussed the motivation for using ICA as a method for learning dictionaries for sparse representations of (patches of) natural images. More details related to dictionary learning procedure using ICA are presented in Section 5.1.2 and Section 7.1.

Chapter 4

Algorithms for under-determined systems of linear equations with sparsity constraints

In Section 2.3.2 basic results were presented concerning the uniqueness of the solution of the problem

$$\min_{\mathbf{s}} \|\mathbf{s}\|_0 \quad \text{subject to} \quad \mathbf{x} = \mathbf{A}\mathbf{s}, \tag{4.0.1}$$

where $\mathbf{A} \in \mathbb{R}^{M \times N}$ and $N > M$. Practically more relevant formulation of the problem, that allows error (coming from noise or model error), is

$$\min_{\mathbf{s}} \|\mathbf{s}\|_0 \quad \text{subject to} \quad \|\mathbf{x} - \mathbf{A}\mathbf{s}\|_2 \leq \varepsilon, \tag{4.0.2}$$

where ε denotes error tolerance. Problems (4.0.1) and (4.0.2) are referred to as *sparse recovery* problems. Methods for solving (4.0.1) and (4.0.2) can be classified as either direct or indirect (relaxation) methods. Main representatives of direct methods are greedy algorithms, like matching pursuit (MP), orthogonal matching pursuit (OMP) and its variants, and iterative hard thresholding methods [11], which we don't review here. We formulate greedy approaches in Section 4.1, with an emphasis to OMP algorithm. Indirect, or relaxation, methods replace the discrete $\|\cdot\|_0$ function by the continuous or smooth approximation, hoping that, under appropriate conditions, the solution of the original problem (4.0.1) or (4.0.2) can be obtained by solving the *relaxed* problem (the one obtained by replacing $\|\cdot\|_0$ with its approximation). ℓ_1 norm, defined for vector $\mathbf{x} \in \mathbb{R}^N$ as $\|\mathbf{x}\|_1 = \sum_{i=1}^N |x_i|$, and generally ℓ_p functions for $0 < p \leq 1$, defined as $\|\mathbf{x}\|_p^p = \sum_{i=1}^N |x_i|^p$, are the main representatives of this approach. ℓ_1 norm can be seen as a convex function 'closest' to ℓ_0. Namely, we have $\|\mathbf{x}\|_0 = \lim_{p \downarrow 0} \|\mathbf{x}\|_p^p$. Also, ℓ_1 norm is the convex envelope of ℓ_0 on the set $\|\mathbf{x}\|_\infty \leq 1$. In Section 4.1 we briefly present conditions under which the solution of (4.0.1) or (4.0.2) can be obtained by solving the relaxed problem, in which $\|\cdot\|_0$ is replaced by $\|\cdot\|_1$. The relaxed problem is a convex optimization problem, which can be solved by one of many (fast) specialized algorithms. In the case where ℓ_p function, for $0 < p < 1$, is used as an approximation of ℓ_0, we obtain a non-convex optimization problem. Empirically, improved results can be obtained compared to the ℓ_1 case, although in theory the best we can expect is finding the local minimum of the relaxed problem.

We review some basic results on the connection between the original and ℓ_1-relaxed problem from the literature in Section 4.1. There are also other approaches for solving (4.0.1) and (4.0.2), like Bayesian methods, which have shown excellent performance in practice [94]. See [105] for a review of various approaches to sparse recovery. In Section 4.2, one effective method for sparse recovery ('Smoothed ℓ_0 method'), belonging to the class of indirect (relaxation) methods, is presented. Although there exist many more methods, which also *sometimes* perform better than the methods we have reviewed here, we have concentrated on three simple approaches that are good enough for the experiments we will present in Chapter 7. The results presented there were obtained using the Smoothed ℓ_0 method presented in Section 4.2.1, since it yielded the best results when compared to OMP and ℓ_1-minimization. In combination with ICA-based dictionary learning method reviewed in Section 3.2, Chapter 3, it yielded very good results in image inpainting and removal of impulse noise experiments.

In Section 4.3 we present some numerical simulations comparing the OMP, ℓ_1-minimization and Smoothed ℓ_0 method on synthetic data. Before reviewing the theoretical results and empirical performance in Section 4.1, Section 4.2 and Section 4.3, in the following paragraph we briefly discuss another property of the matrix $\mathbf{A}$ that is often used to state theoretical conditions for the success of algorithms.

Restricted isometry property In Section 2.3.2, two properties of the general matrix $\mathbf{A} \in \mathbb{R}^{M\times N}$ were reviewed: mutual coherence and spark. Both of these properties can be used to theoretically determine the degree of sparsity of the solution which guarantees unique reconstruction. However, coherence and spark are very different regarding the trade-off between computational complexity and sharpness of the resulting theoretical conditions. Coherence is easy to compute, but results in very pessimistic conditions. On the contrary, spark is very hard to compute, but gives sharp conditions for uniqueness. For example, the condition in the proposition Proposition 2.2 is even too sharp in the sense that it can not guarantee *stable* reconstruction. The following definition introduces another property of the matrix, *the restricted isometry property (RIP),* that is useful at least from the theoretical point of view.

Definition 4.1. The matrix $\mathbf{A} \in \mathbb{R}^{M\times N}$, where $N \geq M$, is said to satisfy the (symmetric) restricted isometry property (RIP) of order k with constant $\delta_k \in (0,1)$, where $k \leq M$, if

$$(1-\delta_k)\,||\mathbf{s}||_2^2 \leq ||\mathbf{A}\mathbf{s}||_2^2 \leq (1+\delta_k)\,||\mathbf{s}||_2^2, \tag{4.0.3}$$

for every $\mathbf{s} \in \mathbb{R}^N$ such that $||\mathbf{s}||_0 \leq k$. δ_k in (4.0.3) is called the restricted isometry constant (RIC) of order k of the matrix $\mathbf{A}$.

This definition was introduced in [18]. It follows from the above definition that the condition $\delta_{2k} < 1$ guarantees the uniqueness of the solution $\mathbf{s}$ for which $\|\mathbf{s}\|_0 < k$. However, small values of RIC-s would guarantee, apart from uniqueness, the stability of the reconstruction, i.e. robustness to noise. Many results were presented in the literature regarding the values of RIC-s that guarantee stable reconstruction, with many papers presenting sharper and sharper results compared to previous ones (see, for example, [36] for RIP-based guarantees for ℓ_1 minimization). These theoretical results are important since they state (sharp) conditions under which *stable* sparse recovery is guaranteed. It is important to note that these results are *uniform*, i.e. valid for all $\mathbf{A}$ and $\mathbf{s}$ that satisfy the conditions. This has to be contrasted with empirical evaluation of algorithms (see Section 4.3.1) which often indicates better performance. This is so because empirical evaluation mostly relies on *random* realizations of under-determined linear systems with sparse solutions, which avoid the worst-case instances (since those appear very rarely). See also Section 4.3.1 for related discussion.

However, these results are mainly theoretical, since it is very hard to calculate the precise RIC-s for a given matrix. Therefore, in the following sections, in which theoretical conditions for sparse recovery with OMP, ℓ_1 minimization and SL0 method are reviewed, we don't mention results that use the RIP, but only coherence.

4.1 Direct vs. relaxation methods

4.1.1 Greedy methods

The main representative of direct methods for solving (4.0.1) and (4.0.2) is orthogonal matching pursuit (OMP) algorithm [104]. It is a classic greedy algorithm, based on iteratively updating the current approximation of the solution by *locally optimal* steps. More precisely, the structure of the OMP algorithm for the noiseless problem (4.0.1) is as follows. It starts with the initial approximation $\hat{\mathbf{s}}$ of the true solution $\mathbf{s}^*$, $\hat{\mathbf{s}} = 0$. An index set of selected dictionary vectors $\hat{I}$ is initialized as $\hat{I} = \emptyset$. The current residual is defined as $\mathbf{r} = \mathbf{x} - \mathbf{A}\hat{\mathbf{s}}$. Since the solution is a linear combination of a small number (compared to the dimension of $\mathbf{s}^*$) of dictionary vectors, it makes sense to select the dictionary vector that is maximally correlated with the current residual. Namely, in every iteration an index $i^* \in \{1, \ldots, N\}$ is selected such that

$$i^* \in \arg\max_i \left| \left(\mathbf{A}^T \mathbf{r}\right)_i \right|. \tag{4.1.1}$$

Now, i^* is added to the index set $\hat{I}$, and the next approximation of the solution is found using the solution of the least squares problem

$$\hat{\mathbf{s}}_{\hat{I}} \leftarrow \arg\min_{\mathbf{s}} \|\mathbf{x} - \mathbf{A}_{\hat{I}}\mathbf{s}\|_2^2, \tag{4.1.2}$$

where $\mathbf{A}_{\hat{I}}$ denotes the submatrix of $\mathbf{A}$ consisting of columns of $\mathbf{A}$ indexed by $\hat{I}$, and $\hat{\mathbf{s}}_{\hat{I}}$ denotes the subvector of $\hat{\mathbf{s}}$ indexed by $\hat{I}$. The other elements of $\hat{\mathbf{s}}$ are kept at value 0. This procedure is iterated until the norm of the residual drops below some tolerance or until the maximal number of dictionary vectors has been selected. OMP is based on matching pursuit (MP) algorithm, which differs from OMP in the step (4.1.2). Namely, after selecting the new index i^*, MP simply sets the corresponding element of $\hat{\mathbf{s}}$ to the correlation of the current residual with $\mathbf{a}_{i^*}$. It is much faster than OMP but has weaker convergence properties. There are also many variants of the basic OMP algorithm, like orthogonal least squares (OLS) [12], compressive sampling matching pursuit (CoSaMP) [81], gradient pursuits [13] and others (see [105, 16] for other references). Here we review some basic properties of the OMP algorithm. The following theorem holds.

Theorem 4.1. *Suppose that* $\mathbf{s}^*$ *is the solution of the problem (4.0.1) and* I *is the support of* $\mathbf{s}^*$*, i.e.* $I = \{i : s_i^* \neq 0\}$*. Then, a sufficient condition for orthogonal matching pursuit algorithm to recover* $\mathbf{s}^*$*is*

$$\max_{i \in \{1,\dots,N\} \setminus I} \|\mathbf{A}_I^\dagger \mathbf{a}_i\|_1 < 1. \tag{4.1.3}$$

The proof of this theorem can be found in [103]. The condition (4.1.3) guarantees that OMP selects an index $i^* \in I$ in every iteration, so that it recovers the (exact) solution $\mathbf{s}^*$ after $|I|$ iterations. The following result states the condition for exact recovery using the OMP in terms of coherence of $\mathbf{A}$.

Theorem 4.2. *OMP recovers* $\mathbf{s}^*$ *if*

$$\|\mathbf{s}^*\|_0 < \frac{1}{2}\left(1 + \frac{1}{\mu(\mathbf{A})}\right).$$

The proof of this result can be found in [16]. We compare these results with exact recovery results for ℓ_1-relaxed problem in Section 4.1.2.

Regarding the formulation (4.0.2), that takes into account additive noise or error, the following result holds [24].

Theorem 4.3. *Suppose that the optimal solution* $\mathbf{s}^*$ *of the problem (4.0.2) satisfies*

$$\|\mathbf{s}^*\|_0 \leq \frac{1+\mu(\mathbf{A})}{2\mu(\mathbf{A})} - \frac{1}{\mu(\mathbf{A})} \frac{\varepsilon^*}{\min_{i \in I} |s_i^*|}, \tag{4.1.4}$$

where $I = supp(\mathbf{s}^*) = \{i : s_i^* \neq 0\}$ *is the support of* $\mathbf{s}^*$ *and* $\varepsilon^* = \|\mathbf{x} - \mathbf{A}\mathbf{s}^*\|_2$ *is the noise level. Let us denote by* $\hat{\mathbf{s}}_{\varepsilon^*}$ *the result of the OMP algorithm which stops when the representation error* $\leq \varepsilon^*$*. Then,*

1. $\hat{\mathbf{s}}_\varepsilon$ has the correct support,

$$\operatorname{supp}(\hat{\mathbf{s}}_{\varepsilon^*}) = \operatorname{supp}(\mathbf{s}^*);$$

2. $\hat{\mathbf{s}}_\varepsilon$ approximates $\mathbf{s}^*$,

$$||\hat{\mathbf{s}}_{\varepsilon^*} - \mathbf{s}^*||_2^2 \leq \frac{(\varepsilon^*)^2}{1-\mu(\mathbf{A})\left(||\mathbf{s}^*||_0 - 1\right)}.$$

Note that this result assumes that the noise level ε^* is known and provided to OMP algorithm. Also note (from (4.1.4)) that the smallest nonzero element of $\mathbf{s}^*$ should be sufficiently large compared to the noise level.

Computationally, the most demanding step in OMP is solving the least squares sub-problem (4.1.2). Since in every iteration only one index is added to a current index set $\hat{I}$, this step can be done by updating the Cholesky or QR factorization of the current sub-matrix $\mathbf{A}_{\hat{I}}$, which can be done by solving one $|\hat{I}| \times |\hat{I}|$ linear system. The least squares step can then be solved efficiently by solving two triangular systems. Overall, computational complexity of OMP is $O(|I|MN)$. OMP was used in the experiments presented in Chapter 7 as a part of the K-SVD algorithm for dictionary learning (see Chapter 5).

One problem with OMP algorithm is that, if it selects a wrong index i^* in one step, it stays in the index set $\hat{I}$ and the error introduced through adding this wrong index propagates through subsequent iterations. Therefore, it was shown both theoretically and empirically that better performance can be obtained by allowing that some indices can be *removed* from the current approximation of the true support of the solution. Of course, this degrades the computational efficiency of the algorithm. We don't review these methods in detail here. See, for example, [81, 98].

4.1.2 Relaxation methods - ℓ_1 minimization

Instead of directly trying to solve hard problem (4.0.1), relaxation methods are based on the idea of replacing the $||\cdot||_0$ quasi-norm with continuous approximations. As already discussed above, a natural idea is to use the $||\cdot||_1$ *norm* (therefore, a *convex* function) instead of $||\cdot||_0$. This results in a convex optimization problem. Namely, we have the following problem:

$$\min_{\mathbf{s}} ||\mathbf{s}||_1 \quad \text{subject to} \quad \mathbf{x} = \mathbf{A}\mathbf{s}, \tag{4.1.5}$$

and the formulation that allows additive noise or error,

$$\min_{\mathbf{s}} ||\mathbf{s}||_1 \quad \text{subject to} \quad ||\mathbf{x} - \mathbf{A}\mathbf{s}||_2 \leq \varepsilon. \tag{4.1.6}$$

The main question is, under what conditions do the *global* solutions of the original and relaxed problems coincide. Note that the global solution of the relaxed problem can be obtained by practical polynomial-time algorithms, since we are dealing with the convex optimization problem. In terms of coherence of $\mathbf{A}$, the sufficient condition can be stated as follows (Theorem 7 in [16] and references therein).

Theorem 4.4. *If the solution* $\mathbf{s}^*$ *of (4.0.1) satisfies*

$$\|\mathbf{s}^*\|_0 < \frac{1}{2}\left(1 + \frac{1}{\mu(\mathbf{A})}\right), \tag{4.1.7}$$

then $\mathbf{s}^*$ *is both the unique solution of (4.0.1) and (4.1.5).*

Therefore, we have the complete analogue of the recovery Theorem 4.2 for OMP. In fact, the condition (4.1.3) also holds for ℓ_1 minimization. The proof can be found in [103]. The condition in the theorem is pretty restricting and pessimistic, since in practice the solution of (4.0.1) can often be obtained even for significantly less sparse $\mathbf{s}^*$ than the upper bound (4.1.7) suggests, by solving the relaxed problem (4.1.5). Regarding the stability of ℓ_1 minimization when noise is present, a result similar to the Theorem 4.3 holds.

Theorem 4.5. *If the solution* $\mathbf{s}^*$ *of (4.0.2) satisfies*

$$\|\mathbf{s}^*\|_0 < \frac{1}{4}\left(1 + \frac{1}{\mu(\mathbf{A})}\right), \tag{4.1.8}$$

then the solution $\hat{\mathbf{s}}$ *of the relaxed problem*

$$\min_{\mathbf{s}} \|\mathbf{s}\|_1 \quad \textit{subject to} \quad \|\mathbf{x} - \mathbf{A}\mathbf{s}\|_2 \leq \delta, \tag{4.1.9}$$

where $\delta \geq \varepsilon$*, satisfies*

$$\|\hat{\mathbf{s}} - \mathbf{s}^*\|_2^2 \leq \frac{(\varepsilon + \delta)^2}{1 - \mu(\mathbf{A})\,(4\|\mathbf{s}^*\|_0 - 1)}.$$

The proof can be found in [24].

The problems (4.1.5) and (4.1.6) can be rewritten as linear and quadratic programs, respectively, and solved using standard convex optimization methods (like interior point methods) [15]. There are also many specialized methods that are much faster for large problems, see [105] for some references.

Neither OMP nor ℓ_1 relaxation is uniformly better than the other, meaning that neither condition (4.1.4) nor (4.1.8) is stronger than the other. For example, it is known that ℓ_1 minimization performs better than OMP when all nonzero elements of the true solution $\mathbf{s}^*$ have similar magnitude (ℓ_1 minimization is not sensitive to the distribution of magnitudes of the nonzero elements in $\mathbf{s}^*$) [74].

In the next section, we review one very effective method for sparse recovery, that has some advantages over both greedy methods like OMP, and convex relaxation methods (ℓ_1 minimization). It was used to obtain the experimental results presented in Chapter 7.

4.2 Smoothed ℓ_0 method

4.2.1 The algorithm

ℓ_1 norm is useful as a relaxation of ℓ_0 function since, as a norm, it is a convex function. However, it is a crude approximation of $\|\cdot\|_0$. Therefore, ℓ_p pseudo-norms ($p < 1$) can be used as a better approximations of ℓ_0. Of course, they are non-convex functions with many local minima. Namely, for $\mathbf{A} \in \mathbb{R}^{M\times N}$, $N > M$, with maximal spark (spark($\mathbf{A}$) $= M+1$), there are at least as many local minima as $M \times M$ sub-matrices of $\mathbf{A}$ formed by taking M of its columns, i.e. at least $\binom{N}{M}$ local minima, which can be a huge number. This follows since every quadratic sub-matrix of $\mathbf{A}$ yields one solution $\hat{\mathbf{s}}$ of the system $\mathbf{As} = \mathbf{x}$ with $\|\hat{\mathbf{s}}\|_0 = M$. This fact explains why algorithms minimizing ℓ_p pseudo-norms are very sensitive to initialization. Although even this approach often outperforms ℓ_1 minimization on randomly generated under-determined systems (see [80], for example), here we consider another approach. Namely, ℓ_0 function can be approximated as

$$\|\mathbf{x}\|_0 \approx F_\sigma(\mathbf{x}) = \sum_i \left(1 - \exp\left(-\frac{x_i^2}{2\sigma^2}\right)\right) = \sum_i \left(1 - f_\sigma(x_i)\right), \tag{4.2.1}$$

where σ is a parameter. For every $\varepsilon > 0$, there is small enough σ such that $|\|\mathbf{x}\|_0 - F_\sigma(\mathbf{x})| < \varepsilon$. To see the effect of σ, let us consider our basic sparse recovery problem (4.0.1). Let us denote by $\mathbf{s_p}$ one solution of the under-determined system

$$\mathbf{As} = \mathbf{x}, \tag{4.2.2}$$

for example $\mathbf{s_p} = \mathbf{A}^\dagger\mathbf{x}$, and by $\mathbf{V} \in \mathbb{R}^{N\times(N-M)}$ full-column-rank matrix such that $\mathbf{AV} = 0$. Namely, columns of $\mathbf{V}$ form a basis for the nullspace of $\mathbf{A}$. $\mathbf{V}$ can be obtained from the SVD of $\mathbf{A}$. Then, every solution $\hat{\mathbf{s}}$ of the system (4.2.2) is of the form $\hat{\mathbf{s}} = \mathbf{s_p} + \mathbf{Vc}$, where $\mathbf{c} \in \mathbb{R}^{N-M}$. By choosing $N - M = 2$, we can visualize the function

$$\tilde{F}_\sigma(\mathbf{c}) = F_\sigma(\mathbf{s_p} + \mathbf{Vc}),$$

see Figure 4.2.1.

The figure illustrates that for large values of σ, local minima of the function $\tilde{F}_\sigma$ are smoothed out. Also, $\tilde{F}_\sigma$ is easier to optimize for larger σ, since for small σ it becomes almost flat except in the small region around every local minimum. However, it is not clear whether the global minimum of $\tilde{F}_\sigma$ for larger values of σ corresponds to or is even close to the global minimum of $\|\mathbf{s_p} + \mathbf{Vc}\|_0$. The basic idea of the Smoothed ℓ_0 (SL0) [79] algorithm is the following: by performing local minimizations of $\tilde{F}_\sigma$ for decreasing sequence of σ, we hope that the obtained sequence of points will converge to $\mathbf{s}^*$, the global optimum of the initial problem. We review formal conditions under which the convergence can be obtained in the next subsection. The algorithm steps are outlined in Algorithm 4.1.

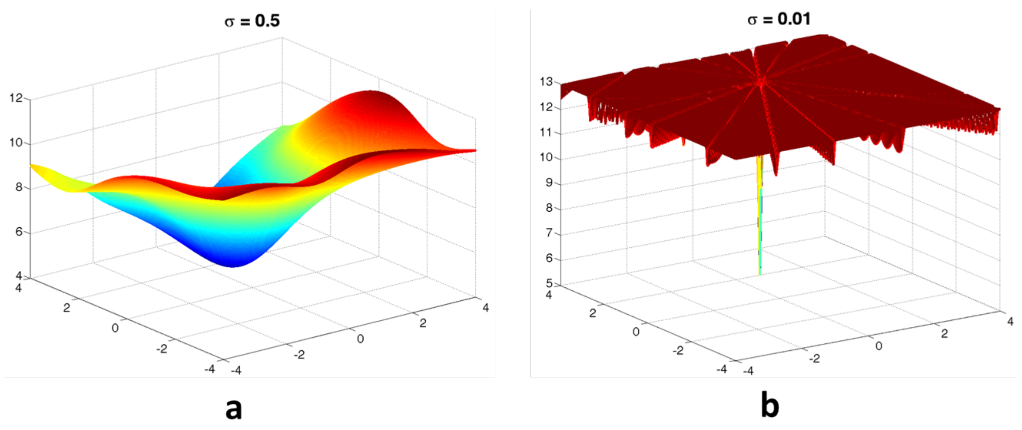


Figure 4.2.1: (a) Graph of F for large value of σ. (b) Graph of F for smaller σ. F is almost discontinuous and has many local minima.

Algorithm 4.1 Smoothed ℓ_0 (SL0) algorithm for the noiseless sparse recovery problem (4.0.1)

```
• Initialization:
    – let $\hat{\mathbf{s}}_0$ be one solution of the system $\mathbf{As} = \mathbf{x}$ (for example, minimum $\ell_2$-norm solution,
      $\hat{\mathbf{s}}_0 = \mathbf{A}^{\dagger}\mathbf{x}$)
    – choose a suitable decreasing sequence $\sigma_1, \dots, \sigma_J$, a small step size $\mu > 0$, number
      of iterations $J$ and the number of inner iterations $L$
• for $j = 1, \dots, J$
    1. let $\sigma = \sigma_j$
    2. approximately minimize the function $F_\sigma$ defined in (4.2.1) on the set $S =
       \{\mathbf{s} : \mathbf{As} = \mathbf{x}\}$ using $L$ iterations of gradient descent followed by projection onto $S$:
        – initialization: $\mathbf{s} = \hat{\mathbf{s}}_{j-1}$
        – for $l = 1, \dots, L$
            a) let $\delta = \nabla F_\sigma(\mathbf{s})$
            b) let $\mathbf{s} \leftarrow \mathbf{s} - \mu\delta$
            c) project onto the feasible set $S$: $\mathbf{s} \leftarrow \mathbf{s} - \mathbf{A}^{\dagger}(\mathbf{As} - \mathbf{x})$
    3. set $\hat{\mathbf{s}}_j = \mathbf{s}$
• final approximation is $\hat{\mathbf{s}} = \hat{\mathbf{s}}_J$
```

The parameters of the algorithm: J, suitable decreasing sequence of σ_j-s, L and μ, depend on the application. In Section 4.3 we present some details regarding the parameters selection and some experiments on synthetic data. Both these synthetic experiments and the inpainting experiments in Chapter 7 indicate that the algorithm is not too sensitive to the choice of the parameters.

4.2.1.1 Noisy version

The Algorithm 4.1 is designed to solve the noiseless problem (4.0.1). However, formulation (4.0.2) is more realistic. The noiseless version of the algorithm, outlined in Algorithm 4.1, can also be used in the noisy case, see Theorem 4.7 in the next section for theoretical results. However, better results can be achieved by using noise-aware version of the algorithm. Namely, the step (2c) in Algorithm 4.1 can be modified to perform the projection onto $S = \{\mathbf{s} : \mathbf{A}\mathbf{s} = \mathbf{x}\}$ only if $\left\|\mathbf{A}\hat{\mathbf{s}}_j - \mathbf{x}\right\|_2 > \varepsilon$, where $\hat{\mathbf{s}}_j$ denotes the current approximation of the solution, and ε is an estimate of the noise level. This is the only difference between the algorithm outlined in Algorithm 4.1 and the modification, presented in [26].

4.2.2 Theoretical analysis

Here we review some theoretical conditions for the convergence of SL0 algorithm to the global optimum of the problems (4.0.1) and (4.0.2). These results are taken from [79]. The basic assumption on $\mathbf{A}$ will be that spark$(\mathbf{A}) = M + 1$ (this condition is called *unique representation property* (URP) in [79]). Without loss of generality, we can assume that the columns $\mathbf{a}_i$ of $\mathbf{A}$ are normalized, $\|\mathbf{a}_i\|_2 = 1$, for all i. It is also assumed that the minimization of F_σ is done perfectly for every σ, i.e. a global solution is found for every σ. This is not very unrealistic since by gradually decreasing σ it can be assumed that local minima of ℓ_0 are avoided and the approximation is guided towards the true solution. The basic theorem is stated as follows [79].

Theorem 4.6. *Let us denote $S = \{\mathbf{s} : \mathbf{A}\mathbf{s} = \mathbf{x}\}$. Let us consider a family of functions f_σ, indexed by $\sigma \in \mathbb{R}_+$, that satisfy:*

1. $\lim_{\sigma\downarrow 0} f_\sigma(s) = 0$, *for all* $s \neq 0$;
2. $f_\sigma(0) = 1$, *for all* $\sigma \in \mathbb{R}_+$;
3. $0 \leq f_\sigma(s) \leq 1$, *for all* $\sigma \in \mathbb{R}_+$, $s \in \mathbb{R}$;
4. *for every* $\nu > 0$, $\alpha > 0$, *there is* $\sigma_0 \in \mathbb{R}_+$ *such that*

$$|s| > \alpha \Rightarrow f_\sigma(s) < \nu, \text{ for all } \sigma < \sigma_0. \tag{4.2.3}$$

Let $\mathbf{s}^\sigma$ be the (global) minimizer of F_σ on S. Then,

$$\lim_{\sigma\downarrow 0} \mathbf{s}^\sigma = \mathbf{s}^\mathbf{0}.$$

This theorem is based on the following lemmas.

Lemma 4.1. *Let spark$(\mathbf{A}) = M + 1$. If $N - M$ elements of $\mathbf{s} \in$ null$(\mathbf{A})$ converge to zero, than all elements of $\mathbf{s}$ converge to zero.*

Proof. Let $\beta > 0$. Let us denote by I_α the set of indices i such that $|s_i| > \alpha$. Since $\mathbf{s} \in \mathrm{null}(\mathbf{A})$, it follows

$$\sum_{i \in I_\alpha} s_i \mathbf{a}_i + \sum_{i \notin I_\alpha} s_i \mathbf{a}_i = 0,$$

and therefore

$$\begin{aligned}
\left\| \sum_{i \in I_\alpha} s_i \mathbf{a}_i \right\|_2 &= \left\| \sum_{i \notin I_\alpha} s_i \mathbf{a}_i \right\|_2 \leq \sum_{i \notin I_\alpha} \|s_i \mathbf{a}_i\| \\
&\leq \sum_{i \notin I_\alpha} |s_i| \\
&\leq (N - |I_\alpha|)\,\alpha \leq N\alpha.
\end{aligned}$$

The sub-matrix $\mathbf{A}_{I_\alpha}$, consisting of columns $\mathbf{a}_i, i \in I_\alpha$, is regular (more precisely, full rank) since $|I_\alpha| \leq M$ can be assumed ($N - M$ elements of $\mathbf{s}$ are arbitrarily small). Then,

$$\begin{aligned}
\|\mathbf{s}_{I_\alpha}\|_2 &= \left\| \mathbf{A}_{I_\alpha}^{-1} \left(\sum_{i \in I_\alpha} s_i \mathbf{a}_i \right) \right\|_2 \\
&\leq \left\| \mathbf{A}_{I_\alpha}^{-1} \right\|_2 \left\| \sum_{i \in I_\alpha} s_i \mathbf{a}_i \right\|_2 \\
&\leq \left\| \mathbf{A}_{I_\alpha}^{-1} \right\|_2 N\alpha.
\end{aligned}$$

It follows that

$$\|\mathbf{s}\|_2 \leq \|\mathbf{s}_{I_\alpha}\|_2 + \left\| \mathbf{s}_{I_\alpha^C} \right\|_2 \leq \left(\left\| \mathbf{A}_{I_\alpha}^{-1} \right\|_2 + 1 \right) N\alpha.$$

If we denote

$$M_{\mathbf{A}} = \max \left\{ \|\mathbf{A}_I^{-1}\|_2 ; |I| \leq M \right\},$$

it follows

$$\|\mathbf{s}\|_2 \leq (M_{\mathbf{A}} + 1) N\alpha.$$

□

Lemma 4.2. *Let a function f_σ have the properties (2) and (3) from Theorem 4.6, and let F_σ be defined as in (4.2.1). Assume that there exists a sparse solution $\mathbf{s^0} \in S$ such that $\left\|\mathbf{s^0}\right\|_0 = k \leq \frac{M}{2}$ (therefore, $\mathbf{s^0}$ is the unique solution of (4.0.1)). Then, if a solution $\hat{\mathbf{s}} \in S$ satisfies*

$$F_\sigma(\hat{\mathbf{s}}) \leq M - k, \tag{4.2.4}$$

and if $\alpha > 0$ is such that the elements $\hat{s}_i$ of $\hat{\mathbf{s}}$ with $|\hat{s}_i| > \alpha$ satisfy $f_\sigma(\hat{s}_i) < \frac{1}{N}$, then

$$\left\| \hat{\mathbf{s}} - \mathbf{s^0} \right\|_2 \leq (M_{\mathbf{A}} + 1) N\alpha.$$

Proof. Let I_α be as before. We have

$$\begin{aligned} F_\sigma(\hat{\mathbf{s}}) &= N - \left(\sum_{i \in I_\alpha} f_\sigma(\hat{s}_i) + \sum_{i \notin I_\alpha} f_\sigma(\hat{s}_i) \right) \\ &> N - ((N - |I_\alpha|) + 1). \end{aligned}$$

Combining this with (4.2.4) yields $|I_\alpha| - 1 < F_\sigma(\hat{\mathbf{s}}) \le M - k$, i.e. $|I_\alpha| \le M - k$. Now we have that $\hat{\mathbf{s}} - \mathbf{s^0}$ has at most $k + |I_\alpha| \le M$ elements with absolute values greater than α, and of course $\hat{\mathbf{s}} - \mathbf{s^0} \in \text{null}(\mathbf{A})$. Lemma 4.1 implies the statement. □

Lemma 4.2 implies the following corollary.

Corollary 4.1. *Let f_σ, F_σ, S, $\mathbf{s^0}$ be as in Lemma 4.2, and let $\mathbf{s}^\sigma$ be the minimizer of F_σ on S. Then $\mathbf{s}^\sigma$ satisfies (4.2.4).*

Proof. We have

$$\begin{aligned} F_\sigma(\mathbf{s}^\sigma) &\le F_\sigma\left(\mathbf{s^0}\right) \\ &\le k \le M - k. \end{aligned}$$

The second inequality above follows since $\mathbf{s^0}$ has $M - k$ zeros. The last inequality is true since $k \le \frac{M}{2}$. □

Now we have the proof of the main theorem (in the noiseless case).

Proof of Theorem 4.6. We need to show

$$\forall \beta > 0 \; \exists \sigma_0 > 0 \text{ such that } \forall \sigma < \sigma_0 \; \left\| \mathbf{s}^\sigma - \mathbf{s^0} \right\| < \beta.$$

For $\beta > 0$, we define $\alpha = \frac{\beta}{N}(M_\mathbf{A} + 1)$. For $\nu = \frac{1}{N}$, condition (4) of the theorem gives σ_0 for which (4.2.3) holds. For $\sigma < \sigma_0$, this condition states that for elements s_i^σ of $\mathbf{s}^\sigma$ for which $|s_i^\sigma| > \alpha$, we have $f_\sigma(s_i^\sigma) < \frac{1}{N}$. By Corollary 4.1, the condition (4.2.4) of Lemma 4.2 follows. Therefore, by Lemma 4.2 and by definition of α, the statement of the theorem follows. □

In the noisy case (4.0.2), a similar result holds. We state the theorem from [79]. The proof is similar to the proof of Theorem 4.6, see [79] for details.

Theorem 4.7. *Let $S_\varepsilon = \{\mathbf{s} : \|\mathbf{A}\mathbf{s} - \mathbf{x}\|_2 < \varepsilon\}$, $\varepsilon > 0$. Assume that $\mathbf{A}$ and f_σ satisfy the conditions of Theorem 4.6. Let $\mathbf{s^0} \in S_\varepsilon$ be a sparse solution with $k = \|\mathbf{s^0}\|_0 < \frac{M}{2}$. Assume that f_σ satisfies the following additional conditions:*

1. *there exists $\gamma > 0$ such that*

$$\left| f_\sigma'(s) \right| < \frac{\gamma}{\sigma}, \textit{ for all } \sigma > 0 \textit{ and all } s;$$

2. *for every* $\nu > 0$ *and* $\sigma_0 > 0$ *there exists* $\alpha > 0$ *such that*

$$|s| > \alpha \Rightarrow f_\sigma(s) < \nu \text{ for all } \sigma < \sigma_0.$$

Let

$$\sigma_0 = \frac{N\gamma\varepsilon \left\|\mathbf{A}^\dagger\right\|_2}{M - 2k}.$$

Then, optimizing F_{σ_0}*, the sparse solution* $\mathbf{s^0}$ *can be estimated with an error smaller than*

$$(M_\mathbf{A} + 1)(N\alpha + \varepsilon),$$

where α *is such that the above condition (2) holds for* σ_0 *and* $\nu = \frac{1}{N}$.

The function $f_\sigma(s) = \exp\left(-\frac{s^2}{2\sigma^2}\right)$ satisfies all the conditions of both Theorem 4.6 and Theorem 4.7. It is used in the implementation of Algorithm 4.1.

4.3 Comparison of methods on synthetic data

In this section, we generate synthetic sparse recovery problems to compare the performance of three methods reviewed in previous sections: OMP as a representative of greedy (direct) methods, ℓ_1-minimization as a convex relaxation of the problem, and Smoothed ℓ_0 method. First of all, it should be stressed that the level of sparsity of true solution $\mathbf{s}^*$ of (4.0.1) and the properties of $\mathbf{A}$ (spark, coherence...) are not the only factors that affect the empirical performance of algorithms. Namely, the *distribution of the* amplitudes *of non-zero elements* of $\mathbf{s}^*$ also affects the performance. For example, in the papers [74, 97] an extensive performance analysis was done by comparing the recovery success of sparse recovery algorithms for various distributions of amplitudes of non-zero elements of a sparse vector. Here, we illustrate the performance of OMP, ℓ_1-minimization and Smoothed ℓ_0 method on the synthetic example with normal-distributed sparse vectors. Namely, it is a realistic scenario, since for real-world signals like images, their sorted coefficients in some dictionary usually have slowly decaying magnitude, which is well modelled by the normal p.d.f. See Chapter 7, Figure 7.1.3 for an illustration. Before the presentation of the results obtained on synthetic data, we briefly review some results from the literature on the empirical evaluation of algorithms.

4.3.1 Empirical evaluation of the performance of algorithms

Since it is relevant for this section, here we repeat a paragraph from the introduction (it is taken from the paper [74]):

> The empirical tuning approach has a larger significance for the field of sparse representations and compressed sensing. Many of the better known papers in this field discuss what can be proved rigorously, using mathematical analysis. It requires real mathematical maturity to understand what is being claimed and what the interpretation must be, and to compare claims in competing papers. Often, what can be proved is vague (with unspecified constants) or very weak (unrealistically strong conditions are assumed, far from what can be met in applications). For practical engineering applications it is important to know what really happens rather than what can be proved. Empirical studies provide a direct method to give engineers useful guidelines about what really does happen.

These few claims sum up the problems regarding the interpretation of theoretical conditions for the performance of algorithms presented in previous sections. As already noted, the theoretical conditions are mostly too pessimistic, since they result from *the worst case analysis*. On the contrary, synthetic experiments indicate that the algorithms *usually* perform better or much better. The term usually here refers to the fact that synthetic experiments use Monte Carlo simulations (i.e., problem instances are generated *randomly*), which don't reflect the worst-case behaviour. Namely, Monte Carlo simulations indicate what happens *in most of the cases* (or, *with high probability*). Therefore, both the theoretical conditions and empirical evaluation have advantages and disadvantages. An approach presented in [74] consists in generating many instances of the sparse recovery problem for various levels of sparsity and underdeterminacy of the system. Namely, if we denote by $\rho = \frac{k}{M}$ the level of sparsity of the solution, where k denotes the ℓ_0 function of the true solution, and by $\delta = \frac{M}{N}$ the level of underdeterminacy of the system, for every point on some grid of (δ, ρ) value pairs in $[0,1]^2$ a certain number of sparse recovery problems was randomly generated (100 in [74]), and the number of successful recoveries was counted. An important result used in [74] for empirical assessment of the performance of algorithms roughly states the following (see section III in [74]). For fixed $\varepsilon > 0$, the probability that the sparsest solution of the linear system $\mathbf{As} = \mathbf{x}$, where $\mathbf{A} \in \mathbb{R}^{M \times N}$, can be recovered by solving the ℓ_1-relaxed problem (4.1.5) tends to 0 or 1 with increasing system size as $k \sim M \cdot (\rho_{\ell_1}(\delta) \pm \varepsilon)$, where ρ and δ are defined as above. In other words, for large system sizes ($N \to \infty$) there is a curve ρ_{ℓ_1} in the (δ, ρ) plane under which the recovery is successful, and above which the recovery is unsuccessful (both with overwhelming probability). This curve is called (ℓ_1-) phase transition curve, since it indicates the sharp transition in the recovery performance. This result was obtained under the assumption that $\mathbf{A}$ is generated randomly from Gaussian distribution and $\mathbf{s}$ is k-sparse ($\|\mathbf{s}\|_0 \leq k$). However, similar phase transitions have been proved or empirically observed for other types of distributions of $\mathbf{A}$ and $\mathbf{s}$ (see references in [74]). The phase transition curve was estimated for several iterative sparse recovery algorithms in [74]: iterative soft thresholding, iterative hard thresholding and two-stage thresholding (see paper for details) by optimal tuning (i.e. optimal parameters for the algorithms were selected by extensive computational experiments). In the following subsections, we present simpler experiments, by generating random instances of the problem only for several levels of sparsity and for fixed M and N. We compare only the OMP algorithm, ℓ_1 minimization and SL0 method, since it was shown in [74] that ℓ_1 minimization offers the best performance compared to other algorithms used in that paper. The purpose of these comparisons is to show good performance of SL0 method compared to OMP and ℓ_1 minimization, which are the most often used sparse recovery methods. These comparisons justify the use of SL0 in numerical experiments presented in Chapter 7.

The numerical simulations reported in the following subsections were done in `MATLAB` 7.9 on a 3.4 GHz Quad-Core 64-bit Windows 7 PC with 12 GB memory.

4.3.2 Synthetic examples in the noiseless case

We generate synthetic examples as follows. We set $M = 400$, $N = 1000$, and generate $\mathbf{A}$ by command `randn` (M, N) in `MATLAB`, i.e. the elements of $\mathbf{A}$ are i.i.d., generated from $\mathcal{N}(0, 1)$. The columns of $\mathbf{A}$ are normalized after $\mathbf{A}$ is generated in this way. We vary the ℓ_0 function of the sparse solution $\mathbf{s}^* \in \mathbb{R}^N$, denoted by $k = \|\mathbf{s}^*\|_0$, in the range $[80, 100, 120, 140, 160, 180, 200]$. We generate k indexes of non-zero elements of $\mathbf{s}^*$ randomly from $\{1, \ldots, N\}$. The non-zero elements are generated as i.i.d sequence from $\mathcal{N}(0, 1)$. Measurement vector $\mathbf{x}$ is generated as $\mathbf{x} = \mathbf{A}\mathbf{s}^*$. For every sparsity level k, we generate 100 instances of under-determined linear systems, and count the number of successful recoveries of the true sparse solution for every algorithm. The recovery is considered successful if $\|\hat{\mathbf{s}} - \mathbf{s}^*\|_\infty < 10^{-5}$, where $\hat{\mathbf{s}}$ denotes the result of the algorithm.

The parameters for the algorithms were set as follows. For Smoothed ℓ_0 method, the initial value σ_1 of parameter σ was set to $2\|\hat{\mathbf{s}}_\mathbf{0}\|_\infty$, where $\hat{\mathbf{s}}_\mathbf{0}$ is the initial approximation of the solution. The sequence $\sigma_1, \ldots, \sigma_J$ was generated by $\sigma_{i+1} = c\sigma_i$, where $c = 0.5$, wherein $\sigma_J \leq 10^{-6}$. The parameters L and μ in Algorithm 4.1 were set to $L = 10$, $\mu = 2$. This choice yielded good results, while the computational efficiency of the algorithm was preserved (for example, the choice $c \approx 1$ can significantly slow down the algorithm). For OMP, the stopping criterion was selected in the following way: the algorithm stops when the number of selected indexes of the approximation is twice the ℓ_0 function of the true solution $\mathbf{s}^*$. This choice enables better results than when the number of selected indexes is closer to the ℓ_0 function of the true solution $\mathbf{s}^*$. ℓ_1 minimization requires no parameter tuning. Figure 4.3.1 shows the frequency of successful recovery (proportion of successful recoveries over 100 realizations) for various levels of sparsity k and for all three algorithms.

Regarding the computational complexity, Figure 4.3.2 shows average running time for the three algorithms.

Obviously, Smoothed ℓ_0 method achieves the best performance and is also very fast. The OMP

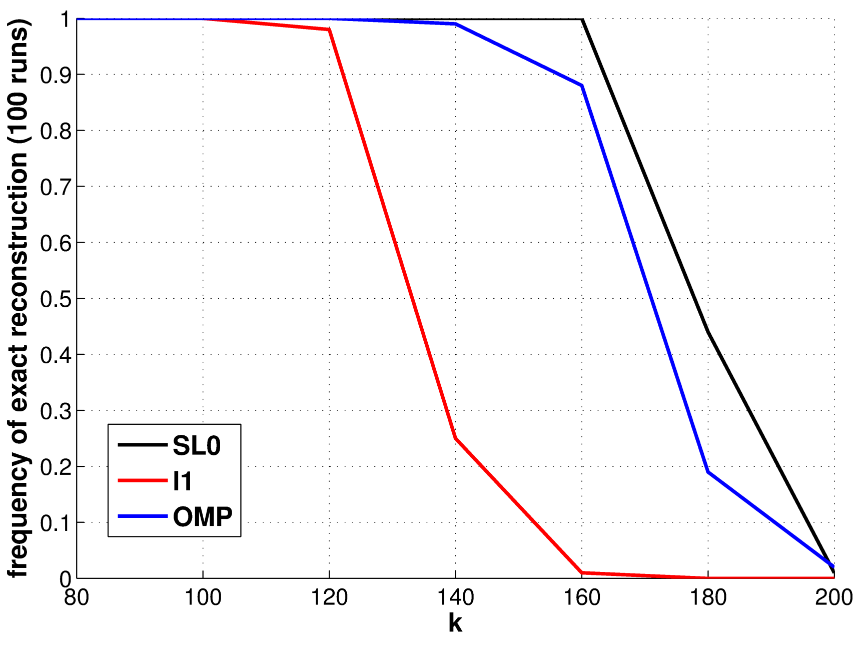


Figure 4.3.1: Comparison of performance of different algorithms.

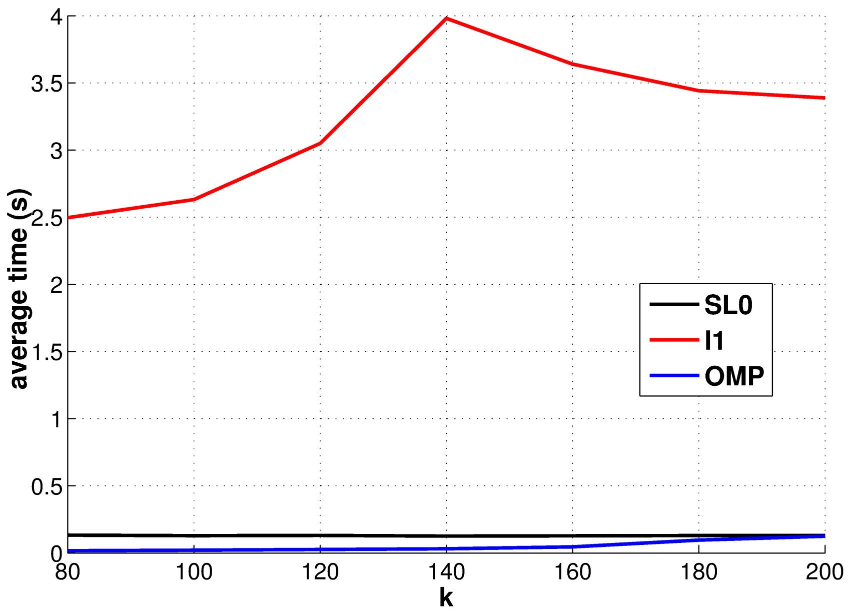


Figure 4.3.2: Average running time for every algorithm.

algorithm exhibited a little worse performance with about the same running time (computational complexity). The problem with OMP and related greedy algorithms, as already noted above, is that it can select an index of a column of $\mathbf{A}$ which is not in the support of the true solution, and this 'greediness' introduces error in the approximation.

It should be noted that some other distribution of the amplitudes of nonzero elements of the true solution would yield different results. Namely, it is known that the worst case distribution for most sparse recovery algorithms, except ℓ_1 minimization, is the symmetric Bernoulli and similar distributions [74]. ℓ_1-minimization approach, however, is not affected by the choice of the distribution. Again, thorough examination of various algorithms' performance for different distributions of amplitudes of nonzero elements of the true solution was presented in [74, 97].

4.3.3 Noisy case

Synthetic examples were generated as in the previous subsection, except that the noise was added to measurements: $\mathbf{x} = \mathbf{A}\mathbf{s}^* + \mathbf{n}$, where $\mathbf{n} \in \mathbb{R}^M$ is the noise vector. The noise level is controlled through the ratio $\|\mathbf{A}\mathbf{s}^*\|_2 / \|\mathbf{n}\|_2$, i.e. through signal-to-noise ratio, snr, defined as $\mathrm{snr}(\mathbf{A}\mathbf{s}^*, \mathbf{n}) = 20\log_{10}\frac{\|\mathbf{A}\mathbf{s}^*\|_2}{\|\mathbf{n}\|_2}$. Signal-to-noise ratio was varied in the range [5, 10, 15, 20, 25]. Here, the level of sparsity of the solution, k, was set to 100, since all algorithms perform well in the noiseless case for this sparsity level (see Figure 4.3.1 in the previous subsection). The performance of algorithms was again measured through signal-to-noise ratio $\mathrm{snr}(\mathbf{s}^*, \mathbf{s}^* - \hat{\mathbf{s}}) = 20\log_{10}\frac{\|\mathbf{s}^*\|_2}{\|\mathbf{s}^* - \hat{\mathbf{s}}\|_2}$, where $\hat{\mathbf{s}}$ denotes the result of the algorithm.

The noise-robust version of Smoothed ℓ_0 method was used in this case, as explained in Section 4.2.1.1. The projection in the step ((2c)) is done if $\|\mathbf{A}\hat{\mathbf{s}} - \mathbf{x}\|_2^2 > 1.1\,\|\mathbf{n}\|_2^2$, where $\mathbf{n}$ is the true noise vector, and $\hat{\mathbf{s}}$ denotes the current approximation of the solution. The other parameters were the same as in the noiseless case. For OMP, the stopping criterion was the following: the algorithm stops when the representation error falls below the small over-estimation of the noise level, $\|\mathbf{A}\hat{\mathbf{s}} - \mathbf{x}\|_2^2 \leq 1.1\,\|\mathbf{n}\|_2^2$. For ℓ_1, the constrained formulation

$$\min \|\mathbf{s}\|_1 \text{ subject to } \|\mathbf{A}\mathbf{s} - \mathbf{x}\|_2^2 \leq \varepsilon$$

of the problem was solved, wherein ε was set to $1.1\,\|\mathbf{n}\|_2^2$. Figure 4.3.3 shows the results.

4.4 Summary

In this chapter we have presented several simple and often used methods for sparse recovery problems (4.0.1) and (4.0.2). Although there are many more methods, which also sometimes perform better than the methods we have reviewed here, we have concentrated on three simple approaches that are good enough for the experiments we will present in Chapter 7. The results presented there were obtained using the Smoothed ℓ_0 method presented in Section 4.2.1, since it yielded the best results when compared to OMP and ℓ_1-minimization. In combination with ICA-based dictionary learning method reviewed in Section 3.2, Chapter 3, it yielded very good results in image inpainting and removal of impulse noise experiments.

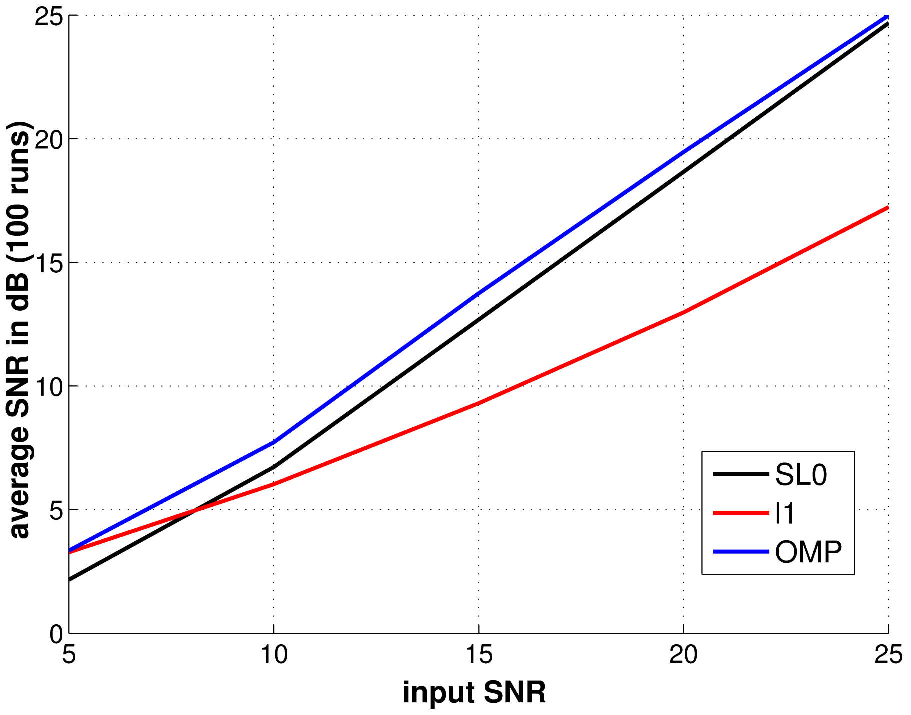


Figure 4.3.3: Comparison of performance of different algorithms in the noisy case.

Chapter 5

Algorithms for dictionary learning

In Chapter 2, Section 2.3, we have discussed methods for the sparse component analysis problem. The main assumption on the sources was *sparsity*. We saw that sparsity enables the solving of underdetermined source separation problems, similarly as independence in the (over)determined case. However, in practice, source signals are often not sparse themselves. Instead, often there is a transformation such that the *transformed* signals *are* sparse or approximately sparse. We have already discussed this problem in the introduction of the section Section 2.3 on sparse component analysis. However, nothing was said about the choice of the *sparsifying* transformation. It can be chosen as one of many available *fixed* mathematical transforms, like *discrete cosine transform* (DCT) or *wavelets* [75]. However, better results can be obtained by using an *adaptive* transforms or *dictionaries*, specially designed to maximize sparsity for a given class of signals. The procedure of designing dictionaries that provide sparse representations of signals in a given class is referred to as *dictionary learning*.

We now review important work done in the dictionary learning domain. In [84], the method has been derived that yields an overcomplete dictionary trained on the patches of natural images by maximizing sparseness based cost function. Similar results were presented in [10] using ICA, and in [64, 65] using probabilistic methods. In a deterministic setting, with data matrix $\mathbf{X}$ of size $n \times T$ with columns that represent the *training set* for the signals of interest, the dictionary learning problem is stated as

$$\arg\min_{\mathbf{D},\mathbf{C}} \|\mathbf{X}-\mathbf{D}\mathbf{C}\|_F^2 \quad \text{subject to} \quad \|\mathbf{c}_i\|_0 \le K, \forall i \tag{5.0.1}$$

where $\mathbf{c}_i$ denotes i-th column of the coefficients matrix $\mathbf{C}$, and K is a bound on the sparsity of representation. The usual method for solving (5.0.1) is alternating minimization: first, $\mathbf{D}$ is fixed and $\mathbf{C}$ is approximated by some of the algorithms reviewed in the previous section. This stage is called sparse coding. In the next step, $\mathbf{C}$ is fixed and $\mathbf{D}$ is updated. This alternating minimization is repeated until convergence to a local minimum (it is also possible that algorithm gets stuck in a saddle point of the problem). The MOD (Method of Optimal Directions) algorithm [30] follows this approach. The dictionary update step in MOD is calculated as $\mathbf{D}_{k+1} = \mathbf{X}\mathbf{C}_k^{\dagger}$, where subscript denotes iteration number and $\mathbf{C}^{\dagger}$ is pseudoinverse of $\mathbf{C}$. Calculating the pseudoinverse in each iteration of the algorithm makes MOD slow. K-SVD algorithm [1], reviewed in Section 5.1.1, fixed this partially. It should be noted that the dictionary learning problem could be stated more generally than in (5.0.1), by using some other measures of representation error and sparsity of coefficients. In the following two subsections, we discuss ICA- and K-SVD-based approaches for approximating the solution of the problem (5.0.1). In Section 5.1.3, another method for dictionary learning is described which is used in experimental comparison of dictionary learning methods in Chapter 7.

5.1 Dictionary learning methods

5.1.1 K-SVD

The K-SVD algorithm [1] is a generalization of the K-means clustering approach to dictionary learning with sparseness constraints when the number of dictionary elements that form one cluster (that corresponds to the level of sparseness K) is greater than 1. Hence, philosophy in dictionary learning by the K-SVD algorithm reflects the knowledge that clustering yields the most efficient signal representation, i.e. representing the signal by one coefficient only. Its development was inspired in part by computational inefficiency of the MOD dictionary learning algorithm [30]. The K-SVD algorithm starts with the formulation (5.0.1). The improvement upon the MOD is in the dictionary update stage. Contrary to the MOD, the K-SVD updates $\mathbf{D}$ column by column. The objective function in (5.0.1) can be written as

$$\|\mathbf{X}-\mathbf{DC}\|_F^2 = \left\| \left(\mathbf{X} - \sum_{j\neq k} \mathbf{d}_j \left(\mathbf{c}^j\right)^T \right) - \mathbf{d}_k \left(\mathbf{c}^k\right)^T \right\|_F^2$$

where $\mathbf{d}_j$ denotes j-th column of $\mathbf{D}$, and $\left(\mathbf{c}^j\right)^T$ denotes j-th row of $\mathbf{C}$. We denote by ω_k a set of indices i for which $\mathbf{c}_i^k \neq 0$, and by Ω_k a matrix of size $T \times |\omega_k|$ with ones at positions $(\omega_k(i), i)$ and zeros elsewhere. Here, we have denoted by $\omega_k(i)$ the i-th element of the set ω_k according to value, from smallest to largest. If we define the error matrix $\mathbf{E}_k$ by $\mathbf{E}_k = \mathbf{X} - \sum_{j\neq k} \mathbf{d}_j \left(\mathbf{c}^j\right)^T$, then K-SVD minimizes $\left\| \mathbf{E}_k\Omega_k - \mathbf{d}_k \left(\mathbf{c}^k\right)^T \Omega_k \right\|_F^2$ with respect to $\mathbf{d}_k$ and $\mathbf{c}^k$. Therefore, updated $\mathbf{d}_k$ and $\mathbf{c}^k$ are scaled first left and right singular vectors of the restricted matrix $\mathbf{E}_k\Omega_k$, respectively. By restricting $\mathbf{E}_k$ and $\mathbf{c}^k$ in this way, updated $\mathbf{c}^k$ is forced to have the same or smaller support than the previous one. Provided that sparse coding stage is solved exactly, this guarantees convergence to a local minimum (or, possibly, a saddle point) of the objective function in (5.0.1) on the constraints set. Since it is solved only approximately, convergence is not guaranteed. However, it seems that convergence occurs in practice. Usually, the algorithm used in sparse coding stage of K-SVD is OMP, for the following reasons: first, it naturally finds an approximation with fixed number of nonzero coefficients, which fits into the dictionary learning framework (5.0.1), and second, it is fast, especially when compared to convex relaxation methods. We discuss details related to the K-SVD algorithm, such as the choice of parameter K, in Chapter 7. Since it has been demonstrated in [1] that the K-SVD outperforms the MOD algorithm, the MOD is not included in the comparative performance analysis between dictionary learning algorithms in Chapter 7.

5.1.2 ICA-based dictionary learning

As already mentioned, ICA is by itself a method for solving source separation problems that uses the assumption of *independence* of the sources. The additional assumption of *approximate* sparsity of the independent components is introduced *implicitly* through the choice of the nonlinear function in the ICA algorithm. Therefore, ICA is not a *direct* method for solving (5.0.1) like K-SVD. Namely, the constraints in the problem formulation (5.0.1) generally can not be satisfied when ICA is used. However, it can be used as an approximation. This is especially the case in dictionary learning for some image restoration problems that we are interested in in this thesis, namely inpainting and removal of impulse noise. See Section 3.1 and Section 3.2 in Chapter 3 for the explanation of the biological background and justification of the use of ICA for dictionary learning.

Here we explain the interpretation of the linear model $\mathbf{X} = \mathbf{DC}$ in the context of ICA-based dictionary learning. Since this thesis deals with applications in image processing, training set (the columns of $\mathbf{X}$) is composed of small parts of images, i.e. *image patches* (see Figure 3.1.1 in Section 3.1). Every row of $\mathbf{X}$ represents one mixed component. Rows of $\mathbf{S}$ represents independent (and approximately sparse) components. Columns of $\mathbf{D}$ are mixing vectors. Since every row of $\mathbf{C}$ is approximately sparse, every column of $\mathbf{C}$ is also approximately sparse with high probability. This means that every image patch (i.e. column of $\mathbf{X}$) can be approximated by a linear combination of a small number of mixing vectors. That is because the coefficients in the linear combination are the elements of the corresponding column of $\mathbf{C}$. Mixing matrix $\mathbf{D}$, therefore, corresponds to the *dictionary* matrix, since by combining its columns (atoms), every image patch can be approximated. In this way, the dictionary for approximately sparse representation of image patches is obtained as a byproduct of ICA.

In the experiments reported in Chapter 7, FastICA was used for the dictionary learning purpose. In Paragraph 2.2.4.3.3, two choices for the nonlinear function in the FastICA algorithm were presented that induce super-Gaussian sources and therefore can be used for dictionary learning. Both these functions yield similar results. It seems, as presented in the experiments in Chapter 7, that the function (2.2.22) yields slightly better results, but function (2.2.23) can also be used since it is faster to compute. The choice of other parameters for FastICA is described in Chapter 7.

5.1.3 Fields of experts image model

Since our experiments in Chapter 7 are concerned with image processing applications, here we briefly review one special class of methods for learning the so-called *image priors*. Namely, problems with incomplete data require some a-priori (prior) knowledge/assumptions on the structure of data of interest, in our case images. In the case of dictionaries, this a-priori knowledge/assumptions means sparsity of images (or image patches) in predefined dictionaries (see the introduction of Chapter 3 for more details). The method that we review in this section assumes some *probabilistic* prior knowledge of image structure. Namely, the spatial structure of image(s) is formulated as Markov random field (MRF). Formally, an image $\mathbf{X} \in \mathbb{R}^{m \times n}$ is viewed as a graph $G = (V, E)$, where image pixels (measurements) X_{ij} form the set V of nodes of the graph (or, more precisely, X_{ij} are indexed by the nodes $v \in V$ of the graph, i.e. pairs (i, j) and nodes $v \in V$ are in bijection). Every node, i.e. image pixel, is viewed as a *random variable*. E is the set of edges of the graph, which are chosen according to some regular neighbourhood structure (the nodes are neighbours if there is an edge $e \in E$ connecting them). Now we have the following definition.

Definition 5.1. X_{ij} form a Markov random field (MRF) with respect to graph G if every X_{ij} is conditionally independent to all other $X_{i'j'}$, $(i', j') \neq (i, j)$, given the neighbours of X_{ij}, i.e.

$$\begin{aligned} P\left(X_{ij} = x_{ij}, X_{i'j'} = x_{i'j'} \left| X_{ne((i,j))} = x_{ne((i,j))}\right.\right) &= P\left(X_{ij} = x_{ij} \left| X_{ne((i,j))} = x_{ne((i,j))}\right.\right) \cdot \\ &\quad \cdot P\left(X_{i'j'} = x_{i'j'} \left| X_{ne((i,j))} = x_{ne((i,j))}\right.\right) \end{aligned}, \tag{5.1.1}$$

where $X_{ne((i,j))}$ denotes the set of neighbours of X_{ij}.

In the above definition, $P(A\,|\,B)$ denotes the conditional probability of event A given B, and is defined as $P(A\,|\,B) = P(A \cap B)/P(B)$.

A clique in a graph is a set of pairwise connected nodes. A special class of MRF-s are those whose probability density can be facorized according to the cliques of a graph. More precisely,

$$p(\mathbf{X}) = \prod_{C \in cl(G)} \Phi_C(\mathbf{X}_C),$$

where $cl(G)$ is the set of cliques of G, $\mathbf{X}_C = \left\{x_{ij} \in C\right\}$, and Φ_C is the joint density of $\mathbf{X}_C$. In [92] a high-order MRF of the form

$$p(\mathbf{X}) = \frac{1}{Z} \exp\left(-\sum_k U_k(\mathbf{X}_k)\right) \tag{5.1.2}$$

was used to model images, where $U_k(\mathbf{X}_k)$ is the so-called potential function for the clique $\mathbf{X}_k$. U_k was chosen as

$$U_k(\mathbf{X}_k) = \prod_{i=1}^{n} \phi_i\left(\mathbf{J}_i^T \mathbf{X}_k; \alpha_i\right),$$

where functions $\phi_i(\cdot;\cdot)$ have the form

$$\phi_i\left(\mathbf{J}_i^T\mathbf{x};\alpha_i\right) = \exp\left(-\alpha_i\sqrt{1+\left(\mathbf{J}_i^T\mathbf{x}\right)^2}\right),$$

ϕ_i are referred to as 'experts'. Consequently, the model (5.1.2) of images is called *'fields of experts'* (FoE) model. $\mathbf{J}_i$ are linear filters (pre-defined or learned vectors) and α_i are the parameters. Here, $\mathbf{J}_i^T\mathbf{x}$ denotes scalar product $\mathbf{J}_i \cdot \mathbf{x}$. $\mathbf{J}_i$ are modelling *directions* in the vector space of pixel values in $\mathbf{X}_C$. In [92], neighbourhoods C, i.e. cliques of the image graph, were chosen as *all* $m \times m$ overlapping image patches (small rectangular regions of an image). The approach taken in [92] is to *learn* all parameters $\Theta = \{\mathbf{J}_i, \alpha_i : i = 1,\ldots,n\}$ of the model on a training set, i.e. to optimize model parameters so that the model error is minimized. Parameter n (the number of experts in the model) can be chosen based on the quality or computational complexity of the model, depending on the application. We don't go into details of (complex) optimization used in [92] because it is not relevant for this thesis, see [92] for details. See Section 7.1 for parameters used when comparing the ICA-based and the above method.

5.1.4 Other dictionary learning methods

Many other dictionary learning methods have been proposed in the literature. Mostly they use the ℓ_1 regularization term for the coefficients matrix $\mathbf{C}$, which makes both sub-problems (optimization over dictionary $\mathbf{D}$ with fixed $\mathbf{C}$ and optimization over $\mathbf{C}$ with fixed dictionary) convex. We don't go into extensive review of the available literature, since the methods presented in this chapter are good representatives of state of the art dictionary learning methods. For more details and a (partial) review of the literature on dictionary learning, see the review paper [102].

5.2 Summary

In this chapter we have briefly reviewed several state of the art dictionary learning methods and pointed to some references where more details on these and other dictionary learning methods can be found. In Chapter 7 we present results of extensive experiments where we have compared these and ICA-learned dictionary based method. These results illustrate comparable or better performance of ICA-based method for inpainting with random pattern of missing values and, perhaps practically more important, removal of salt-and-pepper noise.

Chapter 6

Removal of salt-and-pepper and impulse noise

Denoising is one of the fundamental problems in signal and image processing. It is important both from theoretical and practical point of view. Many algorithms for denoising has been presented in the literature. Mostly they deal with the case of additive white *Gaussian (normal-distributed) noise*. This is justified by the central limit theorem, which roughly states that the superposition (addition) of many individual independent processes (random variables) is Gaussian or close to Gaussian. Since, usually, noise can be assumed to be the result of many individual and independent interferences, it is natural to assume its Gaussianity. The Gaussianity assumption also implies finitness of the noise variance. More precisely, the noise is assumed to be of known, *small variance.* Even in cases where Gaussianity is not assumed, the noise is mostly assumed to be of small variance, i.e. its realization is assumed to be of *small norm.* However, in some cases the noise is *impulsive*, characterized by very high or even infinite variance.

Since our main applications are in image processing, here we define two main models for impulse noise in images. A digital image is viewed as a matrix of grayscale or color values of picture elements (pixels). In this chapter, we will deal only with grayscale images. We will denote by $\mathbf{s} \in \mathbb{R}^N$ a column-vectorized form of an image, and by $\tilde{\mathbf{s}} = \mathbf{s} + \mathbf{n}$ the noise-degraded image $\mathbf{s}$. Here, it is important to note that we consider only *additive* noise. Let us denote by $[d_{min}, d_{max}]$ the dynamic range of pixel values in an image $\mathbf{s}$. In other words, $d_{min} = \min_{i \in \{1,\dots,N\}} s_i$, $d_{max} = \max_{i \in \{1,\dots,N\}} s_i$. The two impulse noise models are the following:

- *salt-and-pepper noise*:

$$\tilde{s}_i = \begin{cases} d_{min} & \text{with probability } \frac{p}{2} \\ d_{max} & \text{with probability } \frac{p}{2} \\ s_i & \text{with probability } 1-p \end{cases},$$

  where p determines the density of salt-and-pepper noise;

- *random-valued impulse noise*:

  $$\tilde{s}_i = \begin{cases} d_i & \text{with probability } p \\ s_i & \text{with probability } 1-p \end{cases},$$

  where d_i is uniformly distributed random number in $[d_{min}, d_{max}]$, and p determines the density of noise.

In Chapter 7 we will reduce the problem of salt-and-pepper noise removal to the inpainting problem by declaring all noise-corrupted pixels as missing. See Section 7.2 in Chapter 7 for details.

The rest of this chapter is organized as follows. In Section 6.1 we describe median and myriad filters, classical methods for removal of impulse noise in signals and images that are theoretically optimal from the *filtering* point of view (however, see Section 6.1.2 for precise results). In Section 6.2, we discuss the inefficiency of the approach motivated from a Bayesian interpretation of the denoising problem.

6.1 Nonlinear filtering

The concept of nonlinear filtering was introduced to enable processing of non-Gaussian random processes (signals). The connection between the nonlinear filtering and the classical, well developed, linear filtering is similar to the connection between principal and independent component analysis. Namely, since the majority of processes (signals) encountered in practice are in fact non-Gaussian, better results in many practical signal processing problems were obtained when the non-Gaussianity was taken into account. In this section we briefly review main results in nonlinear filtering theory. The main reference for this section is the book [5].

6.1.1 Median filters

Firstly, we introduce several definitions. Let us denote by $\mathbf{s} \in \mathbb{R}^N$ a discrete signal. For now, we restrict ourselves to one-dimensional signals. Therefore, the element indices of $\mathbf{s}$ are referred to as time indices. Later we will extend the following definitions to 2-D signals, like images. We have the following basic definition.

Definition 6.1. The output $\mathbf{y} \in \mathbb{R}^N$ of a *running median smoother* (or *filter*) with *window size* $N_L + N_R + 1$, where $N_L + N_R + 1 \leq N$, operating on a discrete sequence (discrete signal, vector) $\mathbf{s} \in \mathbb{R}^N$ is defined at time index $1 \leq n \leq N$ as

$$y_n = \text{median}\left(s_{n-N_L}, \ldots, s_n, \ldots, s_{n+N_R}\right). \tag{6.1.1}$$

When $n \leq N_L$ or $n \geq N - N_R + 1$, y_n is defined by extending the values of $\mathbf{s}$ according to some boundary conditions (see later text for details).

We will suppose that $N_L = N_R$ in the above definition, so that the *window* is symmetric. According to the definition, the median smoother replaces every element of **s** by the sample median of all values of elements of **s** with indexes in the neighbourhood of the index of the selected element. The statistical foundation for this definition is as follows. Suppose that the elements of **s** represent a random sample from some distribution with *location parameter* (mean, if it is finite) β (notice that it is assumed that the elements of **s** are independent!). If the underlying distribution is Gaussian (with mean β and variance σ_i, where σ_i, $i = 1, \ldots, N$, can be different for every element of **s**), the maximum likelihood (ML) estimate $\hat{\beta}$ of the location parameter β given the sample **s** is given as

$$\hat{\beta} = \arg\min_{\beta} \sum_{i=1}^{N} \frac{1}{\sigma_i^2} (s_i - \beta)^2 = \frac{\sum_{i=1}^{N} w_i s_i}{\sum_{i=1}^{N} w_i},$$

where $w_i = \frac{1}{\sigma_i^2}$. In other words, the ML estimate of the location parameter β is given as a weighted sample mean. When the underlying distribution of samples s_i is Laplacian, the ML estimate $\hat{\beta}$ of location is given as

$$\hat{\beta} = \arg\min_{\beta} \sum_{i=1}^{N} \frac{1}{\sigma_i} |s_i - \beta| , \tag{6.1.2}$$

where σ_i is again the variance of the corresponding element of **s** (more precisely, of the underlying random variable). If we suppose that $\sigma_i = \sigma_j$, for all i, j, the above $\hat{\beta}$ is exactly the sample median of **s**. Therefore, median smoother replaces every element of a sequence **s** by the robust estimate of mean of neighbouring elements of **s**. Correspondingly, *weighted* median smoother replaces elements by the *weighted* sample median of their neighbourhood. Notice that the weighted sample median (of a sample **s**) is defined as $\hat{\beta}$ in 6.1.2, and it is equal to one of the samples s_i. See also Section 6.1.2 for generalization.

In the case of two-dimensional signals like images, the definition of weighted median smoother is naturally extended by using 2-D windows in the Definition 6.1. Namely, every image pixel is replaced by the (weighted) sample median of its 2-D neighbourhood.

The elements on the boundary are processed by extending the signal **s** beyond its boundaries according to some boundary conditions. Some often used boundary conditions are symmetric and zero boundary conditions. Symmetric conditions mean that a signal is extended beyond boundaries by mirroring. Zero boundary conditions imply that extended samples are zero. However, zero boundary conditions are ineffective in image processing since artifacts appear on the boundaries of an image.

There are many important variants of the weighted median smoother that were introduced in the literature. For example, center-weighted median smoother allows only the center sample to be weighted. Center-weighted median smoothers are especially suited for image denoising in the presence of impulsive noise, since by simply tuning the center weight, the desired degree of smoothing, i.e. outlier (noise) rejection can be obtained. See also Section 6.1.2. This type of filter was used in the experiments presented in Chapter 7, Section 7.2.

Usually, the term smoother is reserved for the case when the weights w_i above are nonnegative. The above statistical interpretation of median smoothers naturally implies nonnegativity constraints on the weights. However, it was demonstrated in practice that, in some applications, improved results can be obtained if negative weights are also allowed. See chapter 6 in [5] for more details.

Notice that, in Definition 6.1, every sample is replaced by the (weighted) median of its neighbourhood, wherein neighbourhood consists of neighbouring samples before they were processed by median smoother. This kind of median smoother (filter) is referred to as non-recursive. The output of a *recursive* median smoother is defined as (the notation is the same as in definition (6.1.1))

$$y_n = \text{median}\left(y_{n-N_L}, \dots, y_{n-1}, s_n, s_{n+1}, \dots, s_{n+N_R}\right).$$

Therefore, “previous” samples are replaced by median-smoothed versions. Recursive weighted median smoothers are defined analogously.

Median filters work well when the noise is not highly impulsive. When highly impulsive noise is present, median filters need to be modified to improve robustness. In Chapter 7, Section 7.2, we have compared some modified versions of median filters with the method based on inpainting using the ICA-learned dictionary. Although these modified median filters (significantly) improve upon basic median filters, the approach based on inpainting using the ICA-learned dictionary still outperforms it by large margin. See Section 7.2.1 for the results of the experiments that confirm this.

Median filters are theoretically optimal for the Laplace-distributed noise. In some applications, the noise distribution is much more impulsive than the Laplace distribution, and median filters don’t perform well (unless some modifications are introduced, see previous paragraph). In the next subsection, we review myriad filters, that are specifically designed to perform well in the presence of highly impulsive noise.

6.1.2 Myriad filters

The idea of myriad filters is based on some concepts from the field of robust statistics. Therefore, before formally defining the myriad filter, we introduce some definitions and notation. The results in this subsection are taken from [43]. The following definition introduces *robust estimates of the location parameter of a distribution.*

Definition 6.2. Let $x_1, \dots, x_N$ be a random sample from some distribution. An M-estimator of the location parameter of the underlying distribution is defined as

$$\hat{\beta} = \arg\min_{\beta} \sum_{i=1}^{N} \rho\left(x_i - \beta\right),$$

where ρ is a cost function associated with $\hat{\beta}$.

The cost function ρ characterizes an M-estimator. For example, when $\rho(x) = x^2$, the M-estimator corresponds to the sample mean. When $\rho(x) = |x|$, the M-estimator corresponds to the sample median. When $\rho(x) = -\log f(x)$, where f is a p.d.f., $\hat{\beta}$ is a maximum likelihood estimator associated with f. Here, we are concerned with *impulsive distributions f*. More precisely, we are interested in a specific sub-class of *α-stable distributions*. α-stable distributions are those which appear as limits of sums of independent and identically distributed (i.i.d.) random variables (distributions) in central limit theorem of the general form. It is known (see, for example, [25]) that the characteristic function ϕ of a α-stable distribution is of the form

$$\phi(t) = \exp\left(-\gamma |t|^{\alpha}\right), \tag{6.1.3}$$

where $\gamma > 0$ and $\alpha \in (0, 2]$. The parameter α determines the heaviness of the tail of the distribution. It can be shown (exercise 3.7.5 in chapter 3 in [25]) that the tail behaviour of the α-stable distribution is

$$P(|X| > x) \geq c x^{-\alpha},$$

for a constant c (here, X denotes a random variable with α-stable distribution with specific parameter α). The case $\alpha = 1$ corresponds to the Cauchy distribution, which has density

$$f(x) = \frac{\gamma}{\pi} \frac{1}{\gamma^2 + x^2}.$$

Apart from the Gaussian, the Cauchy distribution is the only α-stable distribution for which the closed form expression for its p.d.f. exists. It can also be used as a good model for impulsive distributions. For these reasons, it is used in the definition of the robust estimator of the location parameter in the presence of impulsive noise. Note that the negative logarithm of f is associated to

$$\rho(x) = \log\left(k^2 + x^2\right),$$

where k (that corresponds to γ in (6.1.3)) is called the *dispersion* parameter. We have the following definition [43].

Definition 6.3. Given a set of samples $x_1, \ldots, x_N$ and a parameter $k > 0$, the *sample myriad* of order k is defined as

$$\hat{\beta}_k = \text{myriad}\{k; x_1, \ldots, x_N\} = \arg\min_{\beta} \sum_{i=1}^{N} \log\left[k^2 + (x_i - \beta)^2\right].$$

The impulsiveness of the underlying Cauchy distribution in the sample myriad estimator is controlled through the parameter k. We have the following results [43].

Proposition 6.1. *Given a set of samples $x_1,\dots,x_N$, the sample myriad $\hat{\beta}_k$ converges to the sample average as $k \to \infty$.*

When k is small, the estimator becomes more resistant to the presence of impulsive noise. Before looking at the behaviour of the estimator when $k \to 0$, we state the following definition.

Definition 6.4. Let $x_1,\dots,x_N$ be a set of samples. The mode-myriad estimator, $\hat{\beta}_0$, is defined as

$$\hat{\beta}_0 = \lim_{k \downarrow 0} \hat{\beta}_k.$$

The following result holds.

Proposition 6.2. *Let $\mathscr{M}$ be the set of the most repeated values in the sample $x_1,\dots,x_N$. The mode-myriad can be expressed as*

$$\hat{\beta}_0 = \arg \min_{x_j \in \mathscr{M}} \prod_{i=1, x_i \neq x_j}^{N} \left| x_i - x_j \right|.$$

In other words, mode-myriad is one of the repeating values of the sample. Therefore, this estimator's output is *equal to one of the samples*.

Given a data sample $x_1,\dots,x_N$, in practice one needs to choose the dispersion parameter k to reflect the properties of the noise present in the data. When the noise is impulsive, the estimator should interpret many of data samples as outliers. Therefore, the underlying Cauchy distribution should be highly localized, i.e. k should be 'small'. Geometrically, k is equivalent to half of the interquartile range. Since in the extreme when every sample is a possible outlier, the estimator should be equal to (one of) the sample modes (repeating values), the distribution should be localized so that it is close to an impulse around the sample mode. Therefore, k should be of the order

$$k \sim \min_{i \neq j} \left| x_i - x_j \right|.$$

We now review basic theoretical facts about the sample myriad estimator. The following proposition holds [43].

Proposition 6.3. *Let $T_{\alpha,\gamma}(x_1,\dots,x_N)$ denote the maximum likelihood estimator of the location (mean) derived from a symmetric α-stable distribution with characteristic exponent α and dispersion γ. Then,*

$$\lim_{\alpha \downarrow 0} T_{\alpha,\gamma}(x_1,\dots,x_N) = \mathit{myriad}\{0; x_1,\dots,x_N\}.$$

Therefore, the sample myriad is *theoretically optimal for highly impulsive* (α close to zero) *distributions*.

Usually, a modified form of a sample myriad estimator is used. It is introduced in the following definition.

Definition 6.5. Let $w = [w_1, \ldots, w_N]$ be a vector of non-negative weights. The *weighted myriad* of order k, for $k > 0$, for the data sample $x_1, \ldots, x_N$ is defined as

$$\begin{aligned}\hat{\beta}_{k,w} &= \text{myriad}\{k; w_1 \circ x_1, \ldots, w_N \circ x_N\} \\ &= \arg\min_{\beta} \sum_{i=1}^{N} \log\left[k^2 + w_i (x_i - \beta)^2\right] \end{aligned} \tag{6.1.4}$$

where $\circ$ denotes the weighting operation in Definition 6.5.

The weighted myriad estimator has properties similar to those of a myriad estimator. Namely, we have the following results.

Proposition 6.4. *Let the weights $w_1, \ldots, w_N$ be assigned to samples $x_1, \ldots, x_N$. Then,*

$$\lim_{k \to \infty} \hat{\beta}_{k,w} = \frac{\sum_{i=1}^{N} w_i x_i}{\sum_{i=1}^{N} w_i}.$$

Therefore, for large k, weighted myriad estimator corresponds to the weighted sample mean. A generalization of the mode property also holds.

Proposition 6.5. *Let the weights $w_1, \ldots, w_N$ be assigned to samples $x_1, \ldots, x_N$. Let $\mathscr{M}$ denote the set of the most repeated values in the sample, and let r_j be the number of repetitions associated with an element x_j of $\mathscr{M}$. Then,*

$$\begin{aligned}\hat{\beta}_{0,w} &= \lim_{k \downarrow 0} \hat{\beta}_{k,w}(x_1, \ldots, x_N) \\ &= \arg\min_{x_j \in \mathscr{M}} \left(\frac{1}{w_j}\right)^{\frac{r_j}{2}} \prod_{i=1, x_i \neq x_j}^{N} \left|x_i - x_j\right|.\end{aligned}$$

In Chapter 7, Section 7.2, we have compared the weighted myriad estimator and the method based on inpainting using the ICA-learned dictionary for removal of salt-and-pepper noise in images. See Section 7.2 for details regarding the setting of parameters for the myriad filter. Although the myriad-filter-based denoising can perform well when the density of noise is low or moderate with appropriate parameter tuning, we demonstrate there that the method based on inpainting using the ICA-learned dictionary yields significantly better results when the density of salt-and-pepper noise is high.

6.1.3 Other nonlinear filtering methods

Apart from median and myriad filters and their modifications, many other nonlinear filtering methods have been proposed in the literature. One notable approach that yielded very good results in practice was presented in [99] (see also review paper [78]). They considered the measurement model

$$y_i = s_i + n_i, \tag{6.1.5}$$

where s_i represents the value of the signal of interest (for example, an image) at the position $\mathbf{x}_i, i = 1, \ldots, n$ ($\mathbf{x}_i$ is a two-dimensional vector, $\mathbf{x}_i = [x_{i,1}, x_{i,2}]^T$, in the case of images), y_i is the noisy measured signal and n_i represents noise. The general *kernel regression* framework consists in estimating the underlying $\mathbf{s}$ by using the estimates of the form

$$\hat{\mathbf{s}}\left(\mathbf{x}_j\right) = \arg\min_{\mathbf{s}\left(\mathbf{x}_j\right)} \sum_{i=1}^{n} \left[y_i - \mathbf{s}\left(\mathbf{x}_j\right)\right]^2 K\left(\mathbf{x}_i, \mathbf{x}_j, y_i, y_j\right), \tag{6.1.6}$$

where $\mathbf{s}\left(\mathbf{x}_j\right)$ denotes the value of $\mathbf{s}$ at $\mathbf{x}_j$, and K is a weight (or *kernel*) function that is required to be symmetric, positive and unimodal. Its role is to measure "similarity" between the samples y_i and y_j at respective positions $\mathbf{x}_i$ and $\mathbf{x}_j$ [78]. If K depends only on the spatial locations $\mathbf{x}_i$ and $\mathbf{x}_j$, the resulting estimator is known as classical, non-adaptive kernel regression estimator. Adaptive estimators use kernel K that also takes into account *values* y_i and y_j, and not just spatial locations. Another important question is the range in the sum in (6.1.6). This range is what separates *local* and *non-local* estimators.

The name kernel regression is justified by the following. The measurement model (6.1.5) is represented in a slightly different form:

$$y_i = z\left(x_i\right) + n_i,$$

where $z\left(\cdot\right)$ represents some (a-priori unspecified) *regression function*. It is assumed that z can be approximated by its Taylor expansion:

$$z\left(x_i\right) \approx \beta_0 + \beta_1\left(x_i - x\right) + \beta_2\left(x_i - x\right)^2 + \cdots + \beta_N\left(x_i - x\right)^N,$$

where $\beta_i = z^{(i)}\left(x\right)$ is the i-th derivative of z at x, and x is close to x_i. This Taylor representation can be seen as a local representation of the regression function, and the parameter β_0 would yield an estimate of the data at x_i. The idea is to estimate the parameters β_i from the data, while giving the nearby samples (much) higher weights than samples far away (since the Taylor approximation is local). Namely, in the most basic form of the kernel regression estimation, β_i are estimated by solving

$$\begin{array}{ll} \min_{\{\beta_n\}} & \sum_{i=1}^{n} \Big[y_i - \beta_0 - \beta_1\left(x_i - x\right) - \beta_2\left(x_i - x\right)^2 - \\ & - \cdots - \beta_N\left(x_i - x\right)^N\Big]^2 \frac{1}{h} K\left(\frac{x_i - x}{h}\right) \end{array}, \tag{6.1.7}$$

where K is the kernel function that penalizes the distance from the local position x_i. Other, more advanced kernel regression methods, use kernel K that depends both on spatial distances between samples and their values. For more details, see the review paper [78].

In Chapter 7, Section 7.1.4, we present the results of the experimental comparison between the kernel-regression method and the method based on inpainting using the ICA-learned dictionary applied to the image inpainting problem. It shows that, although kernel-regression is a very effective method, it does not outperform the method based on inpainting using the ICA-learned dictionary on the inpainting problem.

6.2 Maximum-a-posteriori (MAP) approach

Another possible approach to removal of impulse noise is based on the Bayesian interpretation of the problem. Namely, let us recall the formulation (3.0.1) of the problem

$$\mathbf{y} = \mathbf{H}\mathbf{x} + \mathbf{n},$$

where $\mathbf{y} \in \mathbb{R}^m$ represents observed data, $\mathbf{H} \in \mathbb{R}^{m\times M}$ is the degradation operator (wherein $M \geq m$), $\mathbf{x} \in \mathbb{R}^M$ is the vector of clean data that we want to recover, and $\mathbf{n} \in \mathbb{R}^m$ represents noise. More precisely, $\mathbf{n}$ represents a *realization* of the random noise vector $\tilde{\mathbf{n}}$. In the case of denoising, $\mathbf{H} = \mathbf{I}$, where $\mathbf{I}$ denotes the identity matrix. The likelihood function $L(\mathbf{y}) = p(\mathbf{y}\,|\,\mathbf{x})$ is defined as the likelihood of the data given the true (unknown) solution $\mathbf{x}$. It depends on the noise statistics. The most common assumption about noise is that it is Gaussian and has independent elements, i.e. $\tilde{\mathbf{n}} \sim \mathscr{N}\left(0, \sigma^2\mathbf{I}\right)$, where $\mathbf{I}$ is the $m \times m$ identity matrix. In this case, the likelihood of the data is given as the probability density function $p(\mathbf{y}\,|\,\mathbf{x}) \sim \mathscr{N}\left(\mathbf{x}, \sigma^2\mathbf{I}\right)$. In Bayesian statistics, *prior distribution* of the unknown parameters that are of interest refers to the user's assumptions on the parameters. In our case, it means that we assume that the true vector $\mathbf{x}$ is a realization of some underlying random vector with density function $p(\mathbf{x})$. The maximum-a-posteriori (MAP) approach to estimation of the vector of parameters, in our case $\mathbf{x}$, consists of the maximization of the *posterior distribution* $p(\mathbf{x}\,|\,\mathbf{y})$. This posterior distribution is, by Bayes theorem, proportional to $p(\mathbf{y}\,|\,\mathbf{x})\,p(\mathbf{x})$. Equivalently, MAP estimate of $\mathbf{x}$ is obtained by minimizing the negative logarithm of the posterior. If we suppose $p(\mathbf{x}) \sim \exp(-\phi(\mathbf{x}))$, the negative logarithm of the posterior (under the i.i.d Gaussian noise assumption) can be expressed as

$$-\ln p(\mathbf{x}\,|\,\mathbf{y}) \sim \frac{1}{2}\|\mathbf{x}-\mathbf{y}\|_2^2 + \phi(\mathbf{x}). \tag{6.2.1}$$

The assumption of sparsity on $\mathbf{x}$ usually results in non-convex *regularization* term $\phi(\mathbf{x})$. For example, generalized Gaussian distribution, whose pdf is given by

$$p_{\mu;\alpha,\beta}(\mathbf{x}) = \frac{\beta}{2\alpha\Gamma\left(\frac{1}{\beta}\right)}\exp\left(-\left(\frac{|\mathbf{x}-\mu|}{\alpha}\right)^{\beta}\right),$$

where μ, α and β are parameters, can be used as a model of sparse distribution if $\beta \leq 1$. The choice $\beta < 1$ results in ϕ that is proportional to ℓ_p quasi-norms, $p < 1$. The non-convex regularization term leads to non-convex optimization problems, which have, however, proven useful in sparse recovery problems. However, if we assume that the noise is impulsive, the first term in (6.2.1) also becomes non-convex. Namely, we have seen in the previous section (on myriad filters) that impulsive noise can be well modeled by the Cauchy distribution. This noise distribution results in the first term in (6.2.1) of the form

$$\sum_i \ln\left(1+(\mathbf{x}-\mathbf{y})_i^2\right),$$

which is (highly) non-convex in $\mathbf{x}$. Therefore, the resulting optimization problem is extremely hard to solve in practice. Because of this, the MAP approach to removal of impulse noise is usually avoided.

6.3 Summary

In this chapter we have reviewed some classical methods for removal of impulse noise, with an emphasis to median and myriad filters. These are well known and theoretically founded methods. However, they don't perform well when the density of impulse noise is high. We demonstrate in Section 7.2 that the method for removal of salt-and-pepper noise based on using the learned dictionaries for sparse representations outperforms median, myriad, and specialized modified median filters, on the removal of salt-and-pepper noise problem, by large margin.

Chapter 7

Applications

This chapter presents the results of the comparative performance analysis of dictionary learning methods on the problems of inpainting and denoising of natural images. The reported numerical simulations were done in MATLAB 7.9 on a 3.4 GHz Quad-Core 64-bit Windows 7 PC with 12 GB memory. The chapter is organized as follows. In Section 7.1.1 we describe the dictionary learning procedure. Section 7.1.2 describes the parameter selection of sparse recovery algorithms used for the inpainting experiments. There, inpainting methods used in the comparative performance analysis are described as well. Section 7.2 presents the results of the experiments related to the denoising of natural images corrupted by the salt and pepper noise. The results presented in Section 7.1 and Section 7.2 were published in [34, 35]. Section 7.3 presents one application of under-determined source separation methods in bioinformatics. It is a short version of the paper [60].

7.1 Image inpainting

In Section 7.1.1, the used dataset and parameters for all dictionary learning methods are described. In Section 7.1.2, the results of extensive inpainting experiments are presented. Section 7.1.3 presents an extension of the inpainting/denoising concept to color images. In Section 7.1.4, additional comparison with kernel regression method for inpainting is presented.

7.1.1 Dictionary learning

Six images of natural scenes, shown in Figure 7.1.1, were used as the training set for learning of the dictionary matrix. Images were taken from the publicly available database[1] and converted to grayscale. Training images were of the size 576×768 pixels. We randomly extracted 18000 patches of the size 16×16 pixels from six training images (3000 patches per training image) and organized them as columns of the 256×18000 data matrix $\mathbf{X}$. Mean value was subtracted

[1]A. Olmos, F. A. A. Kingdom, McGill calibrated color image database, 2004., http://pirsquared.org/research/mcgilldb/

from every patch: this is a very important preprocessing step. Then, the ICA and the K-SVD were used for dictionary learning. In many papers in the literature, smaller patches were used, and hence also smaller number of atoms in the K-SVD training stage. For example, in [28] the authors used patches of the size 8×8 pixels, and the number of atoms in the training stage was 6, whereas in the inpainting experiments in [73] (although they worked with color images) patches of the size 9×9 pixels were used and the number of atoms in the training stage was 20. Therefore, for comparison, below we also present results obtained using smaller patches, of the size 8×8 pixels, and smaller number of atoms in the K-SVD training stage, namely 4. It should be noted that one of the seminal papers [84] also used patches of size 16×16 pixels, as well as recent paper [72] (in which few patch sizes were used for comparison). Another important point is that the same patch sizes are used for both dictionaries, which allows fair comparison between them.

Figure 7.1.1: Six images from the training set used for bases learning. Images were randomly selected from[1].

`MATLAB` implementations of the FastICA algorithm, available at[2], and the K-SVD, available at[3], were used. As discussed in Paragraph 2.2.4.3.3, tanh nonlinearity (function g_1 from (2.2.24)) in the FastICA algorithm has been used because it yields components with sparse (super-Gaussian) distributions. The parameter a_1 from (2.2.24) was set to 5 by empirical tuning. Sparse coding stage in the K-SVD was done using the OMP algorithm with 40 nonzero coefficients of a solution. Smaller number of nonzero coefficients would speed up the K-SVD algorithm and possibly find sparser representation, but simulations showed that this doesn't yield a better performance. Namely, if the training set doesn't allow such a sparse representation, which seems to be the case with real signals such as natural images, then this model is not appropriate. Therefore, this number was chosen heuristically to obtain sparsity, but only at a reasonable, realistic level. For comparison, we have also tried running the K-SVD algorithm with 4 nonzero coefficients of a solution, see later text. It is expected that better performance can be obtained by using some other reconstruction algorithm, but the reason for choosing the OMP is its speed. Dictionary

[2]The FastICA package for `MATLAB`, http://www.cis.hut.fi/projects/ica/fastica/index.shtml
[3]The KSVD-Box `MATLAB` toolbox, http://www.cs.technion.ac.il/∼ronrubin/software.html

learning with the K-SVD algorithm took around 5 hours for 100 iterations, while dictionary learning for the complete case with the FastICA took around 3 hours. It should be noted that the K-SVD is much faster when using smaller number of atoms, but with larger number of atoms better performance was obtained. Figure 7.1.2 shows basis vectors (i.e. dictionary atoms) learned by the FastICA and K-SVD algorithms with the patches of the size 16×16 pixels. The FastICA and K-SVD were also used for learning of the overcomplete dictionary. Sequential version of the FastICA algorithm was modified to perform quasi-orthogonalization after every step, as explained in Section 2.3.3. Parameter α in (2.3.10) was set to 0.5. The K-SVD takes the number of atoms directly as a parameter.

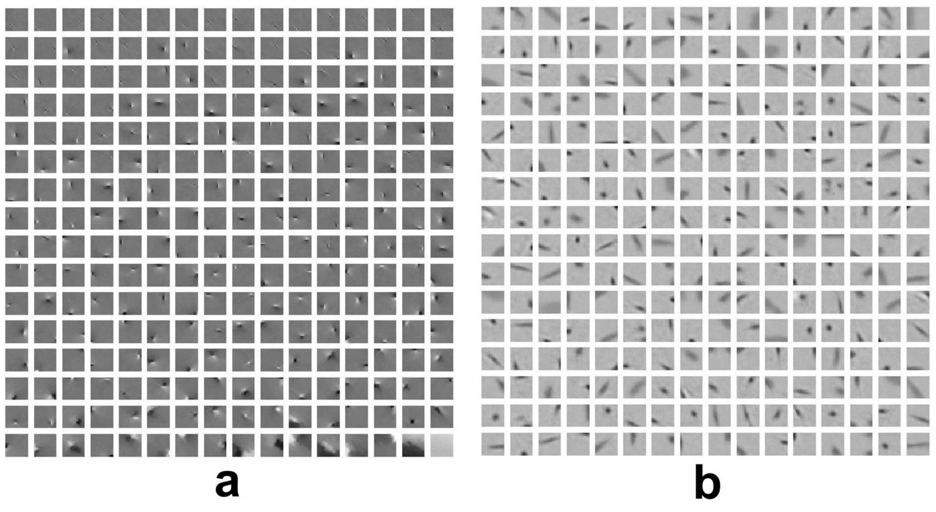


Figure 7.1.2: 256 matricized atoms of the size 16×16 pixels learned by FastICA, (a), and K-SVD, (b), algorithms. The atoms are the columns of the learned dictionary.

7.1.2 Inpainting results

In all examples in this section with random pattern of missing pixels, 80 percent of pixels were removed. We have used freely available `MATLAB` implementation of the Smoothed ℓ_0 (SL0) algorithm[4] for image reconstruction, i.e. inpainting. Justification of this choice was already discussed in Chapter 4. The parameter σ in (4.2.1) was chosen as suggested by the authors: we randomly selected an image patch and computed its coefficients in the learned dictionary by applying a direct transformation. Then the absolute values of these coefficients were sorted in descending order and the smallest 80 percent of them were interpreted as noise. Parameter σ was selected to be few times larger than the standard deviation of the vector of these smallest coefficients. In this way, since natural image patches are not exactly sparse signals (see Figure 7.1.3), smallest coefficients in selected dictionary were interpreted as noise/error. We have also experimented with other values of σ and found that this choice yields the best results. In our simulations the SL0 algorithm worked better and was much faster than other approaches,

[4]http://ee.sharif.ir/∼SLzero/

especially those using ℓ_1 minimization. For example, we have also tested the $\ell_1$ls algorithm for the problem (4.1.6) (`MATLAB` implementation is available at[5]). On average, reconstruction using the $\ell_1$ls took 10 to 15 minutes per image, while the SL0 took about 30 seconds per image only. The OMP performed worse (in terms of the measures of the quality of reconstructed images, see below) than the SL0 and the $\ell_1$ls, with a computational complexity of the order of few minutes. For every patch, before the reconstruction, mean value of the observed pixels in the patch was subtracted from the vector of the observed pixels and added back after the reconstruction. Thus, DC component was artificially added in the reconstruction, yielding better results than when the DC component was a part of the dictionary. To prevent border effects, reconstruction was done with two rows, i.e. columns, of adjacent patches overlapping. After reconstruction, overlapping regions were averaged.

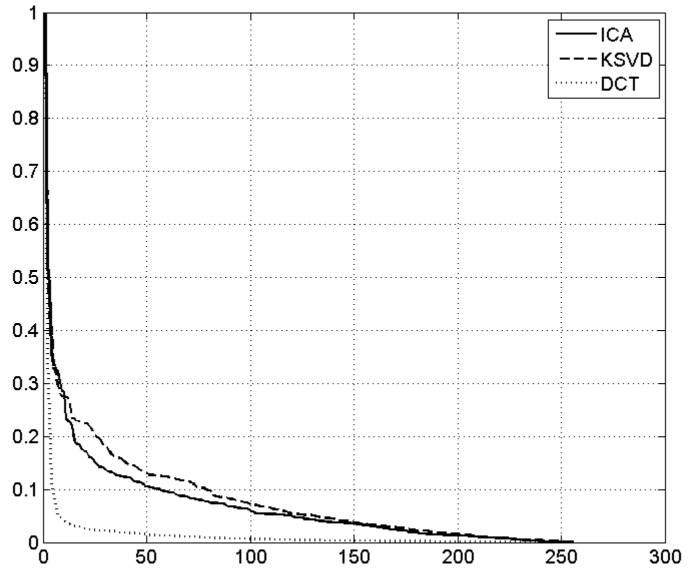


Figure 7.1.3: Coefficients in different bases of a patch chosen randomly from a natural image. Coefficients were normalized for comparison purpose. Without normalization, coefficients in K-SVD and DCT dictionaries are much larger in absolute value (up to three orders of magnitude) than those in ICA dictionary.

For measuring the quality of the reconstructed images, we have used structural similarity index (SSIM) [107], [108] and peak signal-to-noise ratio (PSNR). We have noted that PSNR can give higher values (that should correspond to higher image quality) despite obvious visual quality degradation. That is in line with the objection already pointed out in [107] that high PSNR value does not always correspond with the high quality of visual perception. It was demonstrated that the SSIM is the metric that better corresponds to subjective quality of visual perception. The SSIM index is computed locally on image patches, within a sliding window that moves pixel-by-pixel across the image; local SSIM measures the similarity of local patch brightness values, contrasts and structures. For more details, we refer the interested reader to the paper

[5]http://www.stanford.edu/~boyd/l1_ls/

[108]. Global SSIM is computed as the average of SSIM values across the image. It has values between -1 and 1, achieving maximum value 1 if and only if the images being compared are equal. `MATLAB` code for computing the SSIM index is available at[6].

We have also compared our results with those obtained with morphological component analysis (MCA) method, explained in [29]. The MCA approach models an image as a combination of piecewise smooth ('cartoon') and 'texture' parts. The image decomposition part and the inpainting (the authors in [29] used the term filling-in) part are integrated using a union of fixed bases consisting of one adapted for cartoon and the other adapted to texture part. Inpainting is performed by combining sparse representations of each part. More precisely, it is assumed that a column-vectorized image, $\mathbf{s} \in \mathbb{R}^N$, can be represented as

$$\mathbf{s} = \mathbf{D_t}\alpha_t + \mathbf{D_c}\alpha_c, \tag{7.1.1}$$

where $\mathbf{D_t} \in \mathbb{R}^{N \times L_t}$, $L_t \geq N$, is a dictionary that allows sparse representation of textures in an image, while $\mathbf{D_c} \in \mathbb{R}^{N \times L_c}$, $L_c \geq N$, is a dictionary that enables sparse representation of a cartoon part of an image. The dictionaries $\mathbf{D_t}$ and $\mathbf{D_c}$ are supposed to be incoherent; in other words, it is assumed that $\mathbf{D_t}$ can not sparsely represent the cartoon part of an image, and that $\mathbf{D_c}$ can not sparsely represent the texture part. Let us denote by $\mathbf{M} \in \mathbb{R}^{n \times N}$ a projection matrix that selects given n elements of an image (image 'pixels') and discards the others. Then, [29] proposed to solve the following problem

$$\begin{aligned}\arg\min_{\{\alpha_t,\alpha_c\}} \quad & \{\|\alpha_t\|_1 + \|\alpha_c\|_1 + \\ & +\lambda \left\|\mathbf{M}\left(\mathbf{s} - \mathbf{D_t}\alpha_t - \mathbf{D_c}\alpha_c\right)\right\|_2^2 + \gamma \mathrm{TV}\left(\mathbf{D_c}\alpha_c\right)\Big\}\end{aligned}$$

to obtain the sparse representation coefficients. Therefore, they used the ℓ_1 norm as a measure of sparsity. The additional regularization term $\mathrm{TV}(\mathbf{D_c}\alpha_c)$ denotes the total variation of the reconstructed cartoon part of an image. The total variation is basically the ℓ_1 norm of the image gradient, see paper [29] for details. This term forces the reconstructed cartoon part to have sparser gradient, and hence be closer to a piecewise smooth image. It is important to note that the cartoon part was modelled globally, using wavelet-like dictionaries (*curvelets*), contrary to patch-based approaches. Also, both dictionaries $\mathbf{D_t}$ and $\mathbf{D_c}$ are fixed operators, contrary to learned dictionaries. This is one of the reasons that MCA performs worse than learned-dictionary-based approaches. `MATLAB` implementation of the MCA is available as a part of the MCALab package[7]. Dictionaries used in the MCA were curvelets for the cartoon part and two-dimensional cosine packets for the texture part. Parameters of the MCA were the same as in the Barbara image inpainting example that is available as a part of the MCALab package. Namely, window width for cosine packets was 32 pixels, and the coarsest scale for curvelets was 2. Hard thresholding was used with a linear decrease schedule, and the number of iterations was 300.

[6]http://www.ece.uwaterloo.ca/∼z70wang/research/ssim/

[7]MCALAB webpage

We have also used the FoE method [92] for comparison. `MATLAB` implementation of the FoE method is available at[8]. Default values of the parameters were used.

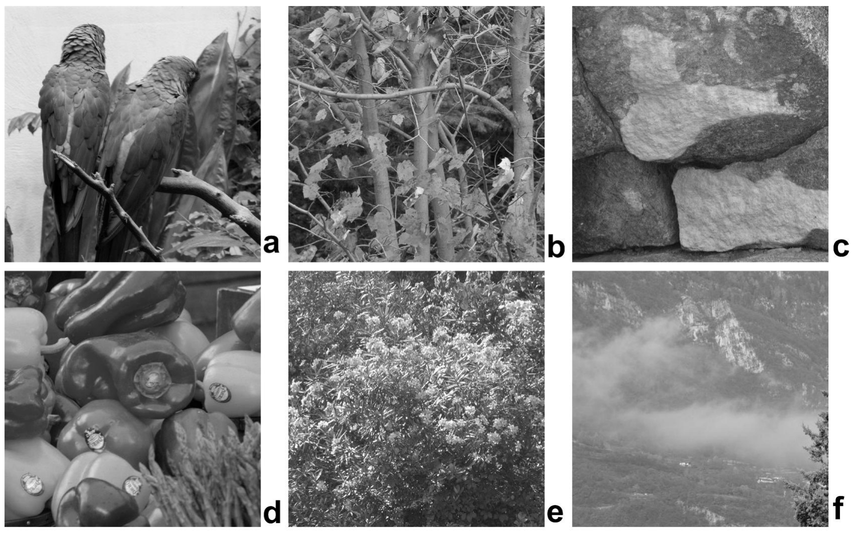


Figure 7.1.4: Images used for validation purpose. Images were randomly selected from[1].

Six images shown in Figure 7.1.4, also taken from[1], were used as the validation set. These images were reshaped to the size of 512×512 pixels. Since the distribution of missing pixels was generated randomly, we repeated the inpainting experiment 10 times for every image in the validation set, every time randomly generating missing pixels distribution. We did not repeat inpainting experiments with the MCA and FoE 10 times because reconstruction was slow (the MCA took about 50 minutes for one image, while the FoE took about 5 hours for one image). Table 7.1.1 shows detailed results of inpainting of six natural images in the validation set using: the ICA and K-SVD learned complete bases and DCT and symmlet 4 wavelet fixed bases. Also presented are results using the MCA and the FoE. Numbers in the table stand for mean values and standard deviations of the SSIM metric of the reconstructed images after 10 runs.

It is clear that the learned bases greatly outperformed fixed bases. It is also clear that ICA outperformed the K-SVD, although not by large margin. One reason for better performance of the ICA-learned dictionary is its smaller coherence, see Figure 7.1.5. Average coherence (or t-coherence [27], see caption of the Figure 7.1.5) better describes columns' correlations than the coherence alone.

Table 7.1.2 shows the corresponding results in terms of the PSNR metric. It can be seen that the results in terms of both metrics are consistent. For comparison, in Table 7.1.3 we also show results obtained by the ICA and K-SVD bases learned on patches of the size of 8×8 pixels, wherein the number of atoms used in the K-SVD training phase was 4. It can be seen that the comparative performance of the two bases is similar. The K-SVD dictionary performed even worse than when using larger patches and larger number of atoms in the training phase.

[8]FoE webpage

Table 7.1.1: Inpainting results in terms of the SSIM metric for the complete bases learned on patches of size 16×16 pixels.

	ICA	**K-SVD**	**DCT**	**Symmlet 4 wavelet**	**MCA**	**FoE**
Fig. 7.1.4a	0.907 ± 0.0008	0.905 ± 0.001	0.75 ± 0.0008	0.736 ± 0.0022	0.789	0.92
Fig. 7.1.4b	0.76 ± 0.0016	0.749 ± 0.0012	0.55 ± 0.0015	0.503 ± 0.0015	0.682	0.77
Fig. 7.1.4c	0.773 ± 0.0007	0.766 ± 0.0011	0.617 ± 0.0011	0.562 ± 0.003	0.644	0.78
Fig. 7.1.4d	0.944 ± 0.0005	0.94 ± 0.0004	0.81 ± 0.0007	0.81 ± 0.0017	0.854	0.95
Fig. 7.1.4e	0.6 ± 0.002	0.577 ± 0.0015	0.434 ± 0.0015	0.35 ± 0.0022	0.491	0.6
Fig. 7.1.4f	0.919 ± 0.0006	0.917 ± 0.0005	0.84 ± 0.0003	0.812 ± 0.0009	0.852	0.92
Mean	**0.817 ± 0.001**	**0.809 ± 0.0009**	**0.666 ± 0.0008**	**0.63 ± 0.002**	**0.719**	**0.824**

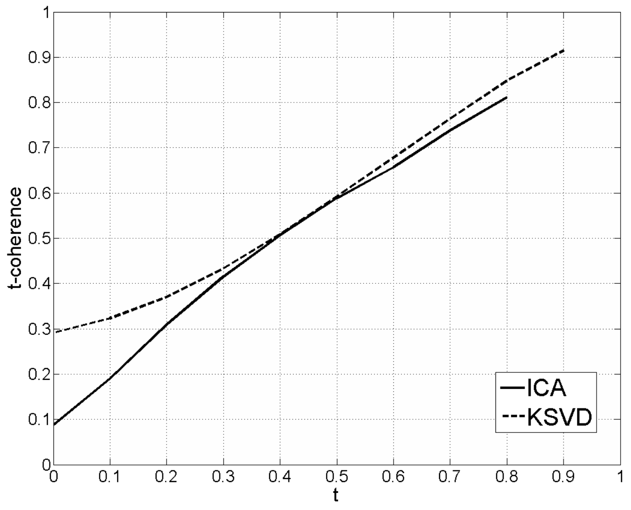


Figure 7.1.5: Comparison of t-coherence of ICA and K-SVD learned bases as a function of t. For a given t, $0 \le t < 1$, t-coherence of a matrix is defined as the mean value of all absolute normalized inner products between different columns of the matrix that are above or equal to t. For $t \to 1$, t-coherence approaches a mutual coherence measure which is defined as maximal absolute normalized inner product between the columns of a matrix.

Figure 7.1.6 shows degraded and reconstructed versions of two images from the validation set, whereupon 80 percent of pixels were removed randomly from each image. Images reconstructed using fixed dictionary are not shown because they are inferior. Table 7.1.4 shows detailed results of inpainting for learned overcomplete bases in term of the SSIM values, while Table 7.1.5 shows corresponding PSNR values. Both bases were twice overcomplete, i.e. of the size 256×512. Again, for comparison, in Table 7.1.6 we also show results obtained with

Table 7.1.2: Inpainting results in terms of the PSNR metric for the complete bases learned on patches of size 16×16 pixels. The values are in dB.

	ICA	**K-SVD**	**DCT**	**Symmlet 4 wavelet**	**MCA**	**FoE**
Fig. 7.1.4a	30.2 ± 0.07	30.2 ± 0.08	25.4 ± 0.02	23.4 ± 0.1	27.1	30.9
Fig. 7.1.4b	24.4 ± 0.04	24.1 ± 0.04	21.6 ± 0.02	19.8 ± 0.03	23.6	24.7
Fig. 7.1.4c	29.7 ± 0.01	29.3 ± 0.02	27.1 ± 0.02	25.6 ± 0.05	27.6	29.8
Fig. 7.1.4d	34.3 ± 0.08	34.4 ± 0.06	28.4 ± 0.05	27 ± 0.1	30.3	35.5
Fig. 7.1.4e	19.5 ± 0.03	18.8 ± 0.03	18.2 ± 0.01	16 ± 0.01	18.4	19.5
Fig. 7.1.4f	33.4 ± 0.09	32.9 ± 0.07	30.5 ± 0.02	28.6 ± 0.09	30.7	33.1
Mean	$\mathbf{28.6 \pm 0.05}$	$\mathbf{28.3 \pm 0.05}$	$\mathbf{25.2 \pm 0.03}$	$\mathbf{23.4 \pm 0.07}$	**26.3**	**28.9**

Table 7.1.3: Inpainting results in terms of the SSIM metric for the complete bases learned on patches of size 8×8 pixels.

	ICA	**K-SVD**
Fig. 7.1.4a	0.9 ± 0.002	0.9 ± 0.0007
Fig. 7.1.4b	0.75 ± 0.003	0.74 ± 0.0016
Fig. 7.1.4c	0.77 ± 0.001	0.76 ± 0.0012
Fig. 7.1.4d	0.936 ± 0.0004	0.935 ± 0.0007
Fig. 7.1.4e	0.58 ± 0.004	0.575 ± 0.0015
Fig. 7.1.4f	0.91 ± 0.001	0.914 ± 0.0008
Mean	$\mathbf{0.809 \pm 0.002}$	$\mathbf{0.804 \pm 0.0011}$

bases learned on patches with the size of 8×8 pixels, wherein number of atoms used in K-SVD training was 4. It can be seen that smaller patches and smaller number of atoms in the K-SVD training phase did not bring performance improvement. It can be seen from the Table 7.1.4 and Table 7.1.5 that the ICA dictionary again outperformed the K-SVD learned dictionary. It is also clear that the use of overcomplete bases did not make significant performance improvement with respect to the complete case, which seems to be consistent with the conclusion already presented in [65] in the speech coding problem.

It should be said that the K-SVD algorithm is designed to minimize the mean squared error (MSE) of the representation, and thus does not necessarily give good performance in terms of the SSIM. The authors in [73] themselves noted that using some metric other than the MSE in the K-SVD could be an interesting direction to study. However, such extensions of the K-SVD are not straightforward.

Table 7.1.4: Inpainting results in terms of the SSIM metric for the two times overcomplete bases (256×512) learned on patches of size 16×16 pixels.

	ICA	**K-SVD**
Fig. 7.1.4a	0.907 ± 0.0005	0.903 ± 0.0011
Fig. 7.1.4b	0.76 ± 0.0008	0.754 ± 0.0011
Fig. 7.1.4c	0.773 ± 0.0011	0.772 ± 0.001
Fig. 7.1.4d	0.944 ± 0.0004	0.94 ± 0.0004
Fig. 7.1.4e	0.6 ± 0.0009	0.596 ± 0.0014
Fig. 7.1.4f	0.919 ± 0.0005	0.918 ± 0.0005
Mean	$\mathbf{0.817 \pm 0.0007}$	$\mathbf{0.814 \pm 0.0009}$

Table 7.1.5: Inpainting results in terms of the PSNR metric for the two times overcomplete bases (256×512) learned on patches of size 16×16 pixels. The values are in dB.

	ICA	**K-SVD**
Fig. 7.1.4a	30.2 ± 0.06	30 ± 0.1
Fig. 7.1.4b	24.5 ± 0.03	24.3 ± 0.03
Fig. 7.1.4c	29.7 ± 0.03	29.6 ± 0.02
Fig. 7.1.4d	34.4 ± 0.06	34.1 ± 0.06
Fig. 7.1.4e	19.5 ± 0.03	19.3 ± 0.02
Fig. 7.1.4f	33.3 ± 0.09	33.2 ± 0.05
Mean	$\mathbf{28.6 \pm 0.05}$	$\mathbf{28.4 \pm 0.05}$

Table 7.1.6: Inpainting results in terms of the SSIM metric for the two times overcomplete bases (256×512) learned on patches of size 8×8 pixels.

	ICA	**K-SVD**
Fig. 7.1.4a	0.899 ± 0.0006	0.896 ± 0.0013
Fig. 7.1.4b	0.745 ± 0.0013	0.74 ± 0.0011
Fig. 7.1.4c	0.765 ± 0.0007	0.765 ± 0.0011
Fig. 7.1.4d	0.936 ± 0.0006	0.931 ± 0.0004
Fig. 7.1.4e	0.593 ± 0.0016	0.59 ± 0.0017
Fig. 7.1.4f	0.915 ± 0.0006	0.915 ± 0.0004
Mean	$\mathbf{0.809 \pm 0.0009}$	$\mathbf{0.807 \pm 0.001}$

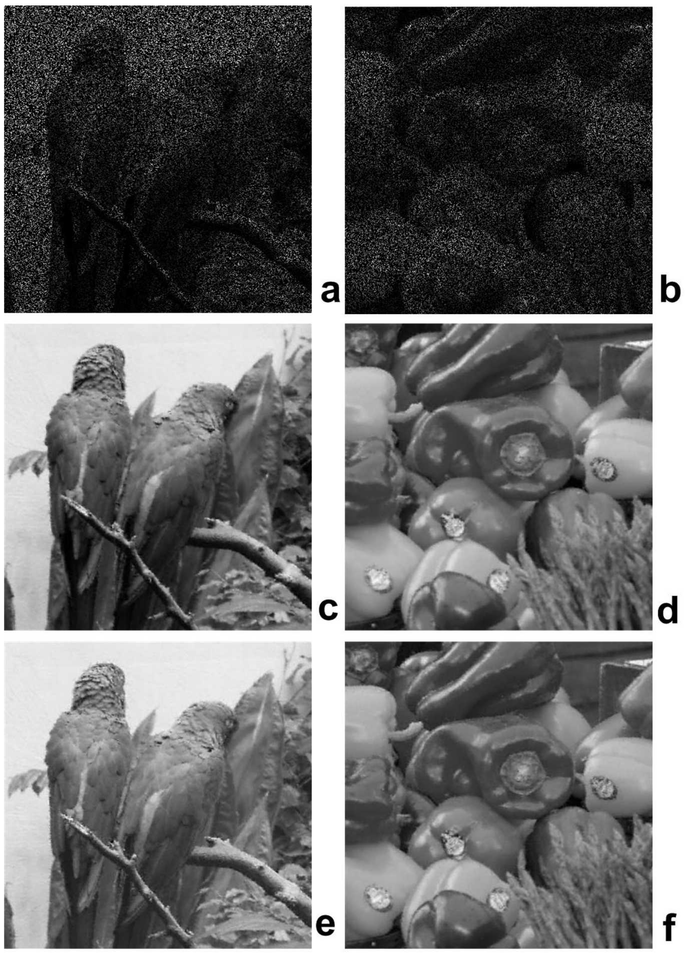


Figure 7.1.6: Examples of two images from the validation set with 80 percent of missing pixels: a) and b). Inpainting of two degraded images using ICA learned dictionary: c) and d). Inpainting of two degraded images using K-SVD learned dictionary: e) and f).

We also include the results of the inpainting experiments with stronger structure of missing pixels, namely lines, blocks and text. We have modified the patch based algorithm presented above for the case when missing regions are larger than the used patch size: only patches that overlap with the missing region and have ratio of known pixels greater than the predefined threshold are inpainted; after the whole image is processed, this procedure is repeated. Similar iterative approaches were used in [45, 46]. Threshold of 0.8 was used in our experiments. Table 7.1.7 shows the results of inpainting, using the ICA and K-SVD bases, where missing pixels have block structure. The ICA dictionary again performed better compared to the K-SVD. Figure 7.1.7 shows two degraded and reconstructed images from the validation set, whereupon images are corrupted by the block pattern of missing pixels. It can be seen that the inpainted regions are blurry, but this is a known effect when large missing regions are being inpainted (for an example, see [29]). Figure 7.1.8 shows another, often used example. It should be noted that our heuristic approach in this case can not compete with more specialized methods like [46, 92]. Table 7.1.8 shows the results of inpainting of images corrupted by the line structures. Figure 7.1.9 shows two degraded and reconstructed images from the validation set, whereupon images are corrupted by the lines pattern of missing pixels. Figure 7.1.10 shows another often used example. We also show the results of text inpainting in Figure 7.1.11 (the images and the corresponding masks were taken from[9]). These examples should illustrate that ICA-based approach to dictionary learning performs reasonably well on realistic inpainting problems.

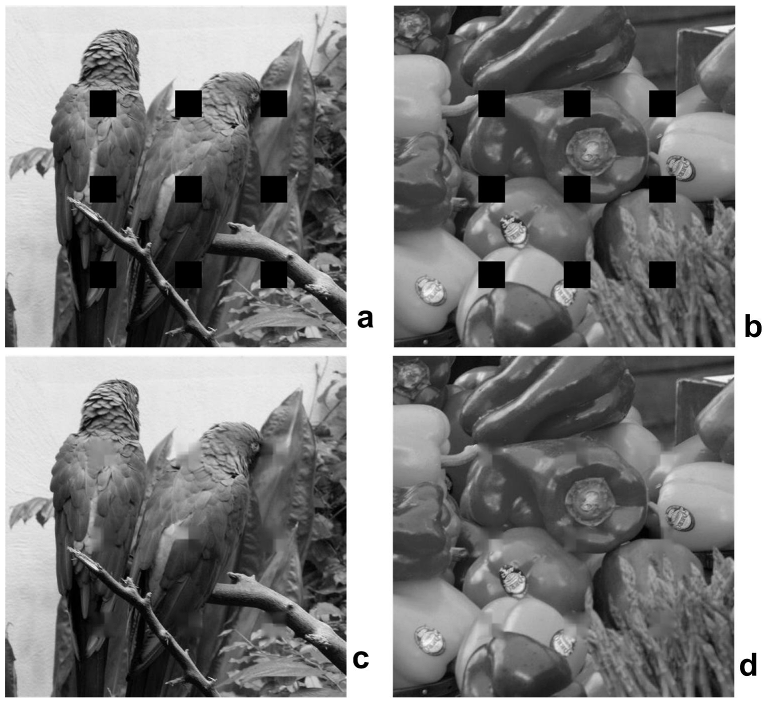


Figure 7.1.7: Examples of two images from the validation set with the block pattern of missing pixels: a) and b). Inpainting of two degraded images using ICA learned dictionary: c) and d).

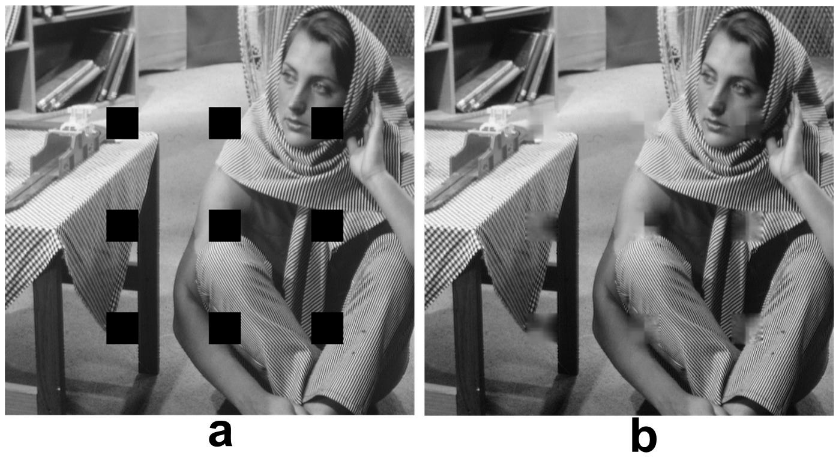


Figure 7.1.8: a) The Barbara image corrupted by the block structure of missing pixels. b) Inpainting using ICA learned dictionary.

[9]http://www.dtic.upf.edu/∼mbertalmio/restoration0.html

Table 7.1.7: Inpainting results in terms of the SSIM metric for the block pattern of missing pixels.

	initial	**ICA**	**K-SVD**
Fig. 7.1.4a	0.9284	0.9746	0.963
Fig. 7.1.4b	0.9303	0.9658	0.9488
Fig. 7.1.4c	0.9254	0.9689	0.9554
Fig. 7.1.4d	0.9265	0.9775	0.97
Fig. 7.1.4e	0.9337	0.9545	0.942
Fig. 7.1.4f	0.9205	0.9885	0.98
Mean	**0.927**	**0.9716**	**0.96**

Table 7.1.8: Inpainting results in terms of the SSIM metric for the lines pattern of missing pixels.

	initial	**ICA**	**K-SVD**
Fig. 7.1.4a	0.939	0.997	0.9965
Fig. 7.1.4b	0.9439	0.9935	0.9887
Fig. 7.1.4c	0.935	0.9943	0.9913
Fig. 7.1.4d	0.935	0.9982	0.9983
Fig. 7.1.4e	0.952	0.9895	0.9825
Fig. 7.1.4f	0.924	0.9975	0.9964
Mean	**0.938**	**0.995**	**0.9923**

7.1.3 Extension to color images

The presented approach to image inpainting can be extended to color images. An (RGB) color image is a 3*D-tensor*, with every pixel being a vector whose components specify the color of the pixel. The RGB color space is the most often used color space for color images. However, some other color space (like YCbCr) could also be used. Here we restrict ourselves to RGB images. The simplest approach to inpainting of color images would be to process every color channel (R, G and B in the case of RGB images) separately. However, we show later that it leads to (severe) color artifacts in the reconstructed image, which degrades its visual quality. Therefore, here we present a method in which every 3D-patch of a color image is vectorized (that is, color channels are not processed separately) and processed in a 'grayscale-like' way. This approach yields results comparable to state-of-the-art, as presented in this subsection.

Before presenting the results, we describe another possible approach to color image inpainting. An RGB image is a 3D tensor. Therefore, color image inpainting is basically a *tensor completion* problem [39]. One approach to solve it is based on minimization of the trace norm of the matricized tensor [39, 101, 69]. Nuclear norm, which is defined as the sum of the singular values of a matrix, is the tightest convex lower bound of the matrix rank on the set of matrices $\{\mathbf{Y} : \|\mathbf{Y}\|_2 \leq 1\}$. Because of its convexity, it is often used as an approximation of the

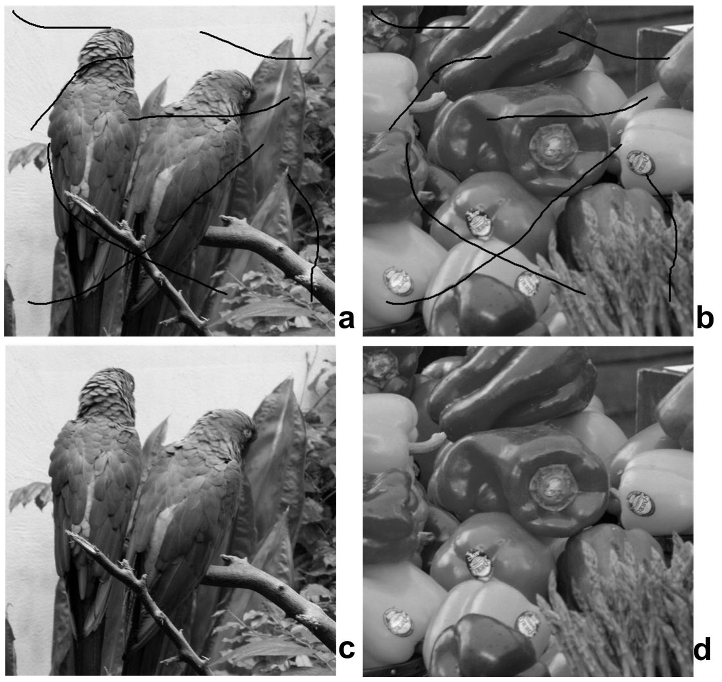


Figure 7.1.9: Examples of two images from the validation set with the lines pattern of missing pixels: a) and b). Inpainting of two degraded images using ICA learned dictionary: c) and d).

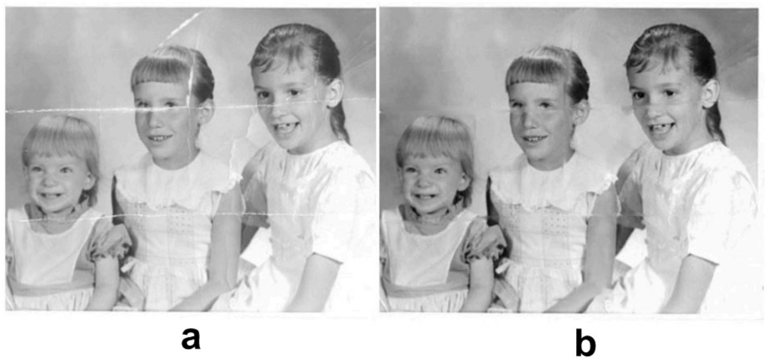


Figure 7.1.10: a) The Girls image. b) Inpainting using ICA learned dictionary.

matrix rank. Hence, the nuclear norm heuristic for tensor completion assumes that a tensor unfolded (matricized) in the selected mode (‘dimension’) is of low-rank. However, fulfillment of the low-rank asumption is data dependent and fails in some applications. When it comes to RGB images, experimental checking demonstrates that the rank of a matricized tensor in each of the three modes mostly equals tensor dimension in that mode. Thus, for RGB color images, low-rank assumption is rarely satisfied. Moreover, it has been demonstrated [95] that the nuclear norm minimization problem can have multiple solutions (i.e., the solution is generally not unique). As shown in [95], the nuclear norm minimization fails to recover a color image damaged by a thick line pattern of missing pixels. Therefore, the nuclear norm heuristic is generally not applicable to the color image inpainting problem.

The dictionary for color image patches was learned on a generic database of natural images. The

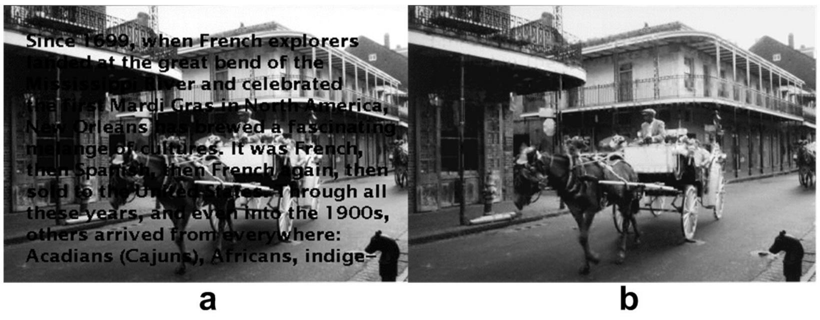

Since 1690, when French explorers
landed at the great bend of the
Mississippi River and celebrated
the first Mardi Gras in North America,
New Orleans has brewed a fascinating
melange of cultures. It was French,
then Spanish, then French again, then
sold to the United States. Through all
these years, and even into the 1900s,
others arrived from everywhere:
Acadians (Cajuns), Africans, indige-
a
b

Figure 7.1.11: Inpainting for text removal. a) Image with text. b) Inpainting using ICA learned dictionary.

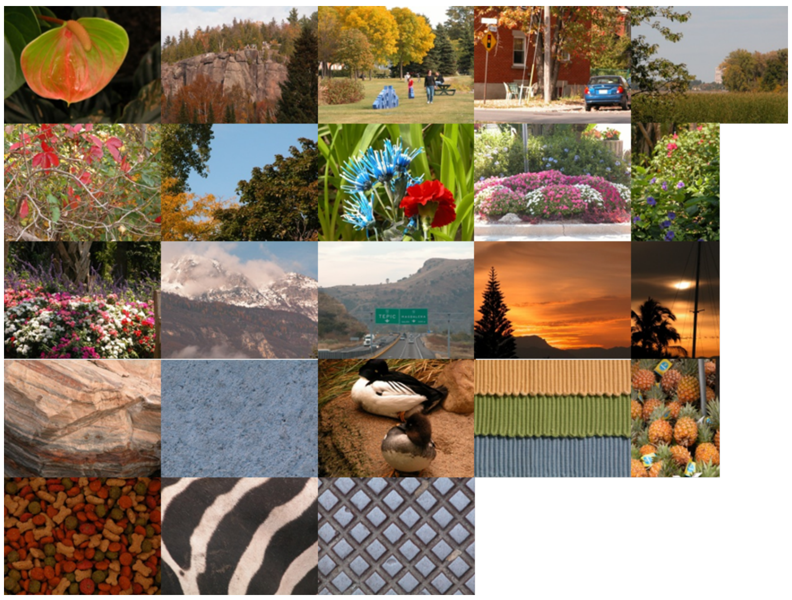


Figure 7.1.12: Training set of color images

23 images used for training are shown in Figure 7.1.12. The patch size was $\sqrt{l} \times \sqrt{l} \times 3$, where $\sqrt{l} = 8$, i.e. $l = 64$. Around 3000 patches were extracted from every image in the training set, wherein patches with small variance were not used for training. Vectorized 3-D patches were stacked as columns in a matrix $\mathbf{S} \in \mathbb{R}^{3l \times T}$, where T is the number of extracted color patches. The color patches need to be pre-processed in a different way than for grayscale image patches. Namely, it is important that the *average colors* are taken into account. More precisely, for color image patches it is important that the patches are *not* centered before dictionary learning. Otherwise, color artifacts appear in the reconstructed image. The rest of the dictionary learning process is performed in the same way as for grayscale images. Namely, we used ICA for dictionary learning, with the same parameters as for grayscale images. We note that *gauss* nonlinearity (2.2.25) can also be used, yielding similar (almost the same) results *faster*. Learned dictionary atoms are shown in Figure 7.1.13.

When inpainting images with missing (color) pixels, again it is important *not* to center the patches with missing pixels, since centering results in color artifacts. This simple approach

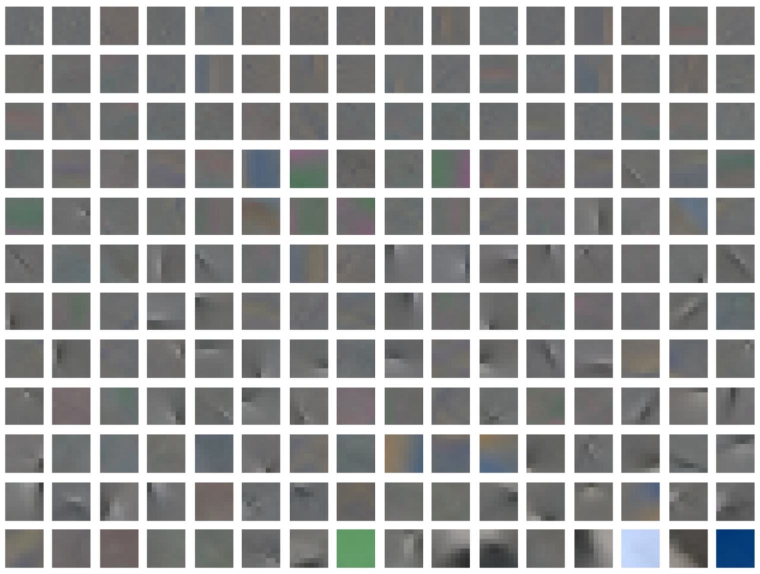


Figure 7.1.13: Learned color dictionary atoms

works (surprisingly) well for color images. This is in contrast to (complex) modifications introduced in K-SVD-like color image inpainting/denoising procedure described in [73]. In Figure 7.1.14 an example is presented. This image was also used in the paper [73] in an example of color image inpainting performance of their method. They obtained 29.65 dB of PSNR, while the method described here achieves performance of 29.36 dB (average over 5 runs, see Figure 7.1.14c for a representative inpainting result). Therefore, it can be seen that the (simple) method described here achieves a result comparable to the one obtained with the more specialized method of [73].

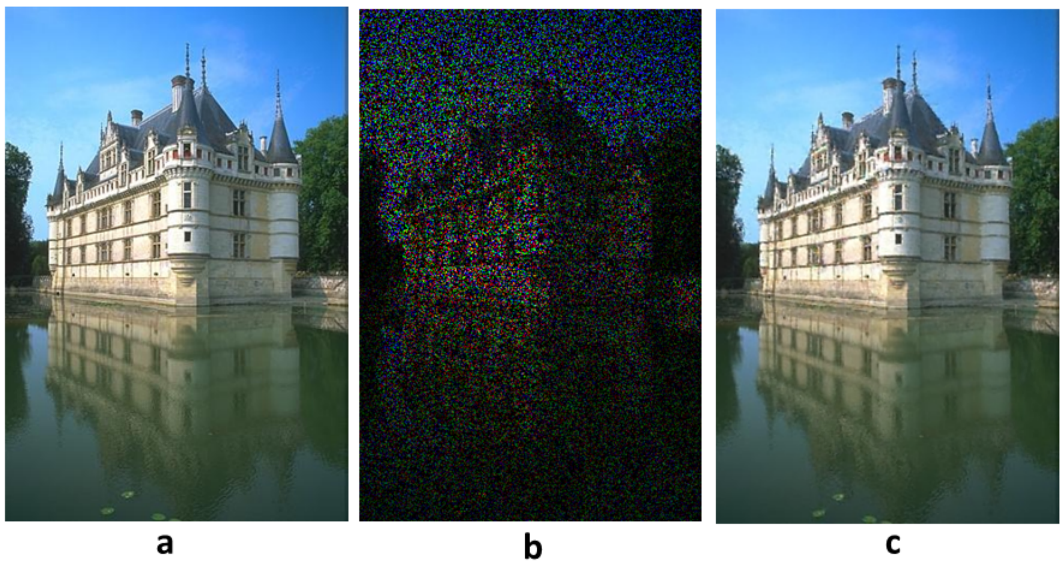


Figure 7.1.14: Color image inpainting example. a) Original image. b) Image with 80 percent pixels missing. c) Inpainting result.

Figure 7.1.15 shows the result achieved by inpainting every channel of the image separately (in grayscale-like manner). It is clear that the resulting image is of bad quality because of color artifacts.

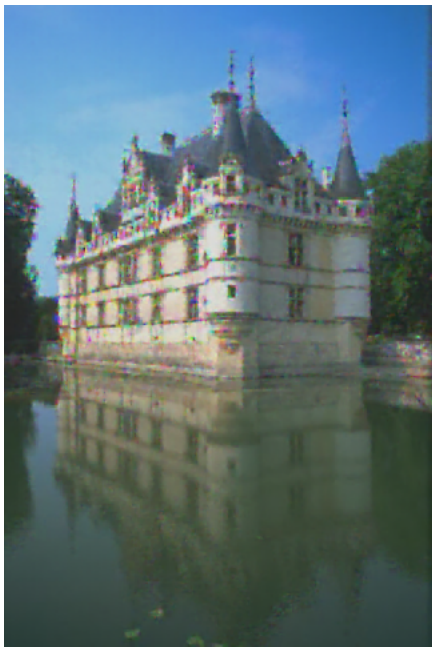


Figure 7.1.15: The result of inpainting every channel separately. Obtained image is obviously of poor quality because of color artifacts.

The results presented in this subsection were published in [35].

7.1.4 Comparison with kernel regression method(s)

In Section 6.1.3 in Chapter 6 an effective nonlinear filtering method which uses *kernel regression* was reviewed. Here we present inpainting results obtained by using one variant of kernel regression, and compare them to the results obtained using the learned dictionary-based inpainting, described in Section 7.1.2.

Firstly, we describe kernel-regression-based inpainting in more detail (again, see [99]). The kernel used (see (6.1.7)) is of the form

$$K_{\text{adapt}}\left(\mathbf{x}_i - \mathbf{x}, y_i - y\right) = K_{\mathbf{H}_i^{\text{steer}}}\left(\mathbf{x}_i - \mathbf{x}\right),$$

wherein $\mathbf{H}_i$ are data-dependent (so called *steering*) matrices. In other words, the kernel K is local since it does not take into account the pixel values y_i, y, but only spatial distance from the center point $\mathbf{x}_i$. The steering matrices $\mathbf{H}_i$ are of the form

$$\mathbf{H}_i^{\text{steer}} = h\mu_i\mathbf{C}_i^{-\frac{1}{2}},$$

where $\mathbf{C}_i$ are covariance matrices estimated based on the differences in local pixels' values, μ_i is the scalar that captures the local density of data samples and can be set to 1, and h is the global smoothing parameter. The steering kernel $K_{\mathbf{H}_i^{\text{steer}}}$ is selected as the Gaussian kernel centered at $\mathbf{x}_i$ and with $\mathbf{C}_i$ as the covariance matrix. $\mathbf{C}_i$ are estimated as follows. The local gradient matrix $\mathbf{G}_i$ is defined as

$$\mathbf{G}_i = \begin{bmatrix} \vdots & \vdots \\ z_1(\mathbf{x}_j) & z_2(\mathbf{x}_j) \\ \vdots & \vdots \end{bmatrix} = \mathbf{U}_i \mathbf{S}_i \mathbf{V}_i^T,$$

where $\mathbf{x}_j \in w_i$ are coordinates of a pixel in the local analysis window w_i (in other words, w_i is the set of coordinates of pixels neighbouring $\mathbf{x}_i$), while $z_1(\cdot)$ and $z_2(\cdot)$ are the first derivatives along horizontal and vertical directions. $\mathbf{U}_i \mathbf{S}_i \mathbf{V}_i^T$ is the truncated singular value decomposition of $\mathbf{G}_i$, wherein $\mathbf{S}_i$ is the 2×2 diagonal matrix representing the energy in dominant directions. $\mathbf{G}_i$ is estimated from the data. The right singular vector corresponding to the smaller singular value should point in the dominant direction of the local gradient field. If we define $\theta_i = \arctan\left(\frac{v_1}{v_2}\right)$, where $v_2 = [v_1, v_2]^T$ is the second column of $\mathbf{V}_i$, and $\sigma_i = \frac{s_1+\lambda'}{s_2+\lambda'}$, wherein s_1, s_2 are diagonal elements of $\mathbf{S}_i$ and $\lambda' \geq 0$ is a regularization parameter, the covariance matrix $\mathbf{C}_i$ should be set to

$$\begin{aligned} \mathbf{C}_i &= \gamma_i \mathbf{U}_{\theta_i} \Lambda_i \mathbf{U}_{\theta_i}^T, \\ \mathbf{U}_{\theta_\mathbf{i}} &= \begin{bmatrix} \cos\theta_i & \sin\theta_i \\ -\sin\theta_i & \cos\theta_i \end{bmatrix}, \\ \Lambda_i &= \begin{bmatrix} \sigma_i & 0 \\ 0 & \sigma_i^{-1} \end{bmatrix}. \end{aligned}$$

γ_i is the scaling parameter of the kernel. In [99] it was suggested to set γ_i to

$$\gamma_i = \left(\frac{s_1 s_2 + \lambda''}{M}\right)^{\frac{1}{2}},$$

where λ'' is a regularization parameter (which prevents γ_i from becoming zero), and M is the number of samples (pixels) in the local analysis window w_i. This choice of the scaling parameter makes the kernel area larger in flat areas and smaller in textured areas (areas with edges). Notice that the procedure described above to construct the covariance matrix from the data is equivalent to applying PCA to the set of local image gradients.

It was suggested in [99] to apply the procedure of the previous paragraph iteratively. They demonstrate that the method is most effective when the output of one iteration (less noisy image) is used to estimate the new parameters of the kernel.

We tested this method on two widely used images, shown in Figure 7.1.16: Lena and Boat. These two images were also used to compare the inpainting result with the results obtained using a modified median filters, presented in Section 7.2.1. Although these filters are used for salt-and-pepper denoising, the results can be compared with inpainting since these two problems (inpainting and removal of salt-and-pepper noise) are very similar. The code for kernel-regression-based image reconstruction was downloaded from[10].

Figure 7.1.16: Images used for comparison. a) Lena image. b) Boat image.

All reported results in terms of the PSNR (in dB) are obtained as the mean over 3 realizations of spatial distribution of missing pixels, generated randomly. Results are very similar for different random masks (the difference in peak SNR is less than 0.5 dB). More realizations of missing pixels' spatial distribution can be used, but it turns out empirically that it is not necessary. The images were damaged by removing 70 and/or 90 percent of pixels randomly. Firstly, Lena image was damaged by removing 70 percent of pixels. The inpainting result using the ICA-learned dictionary was 31.89 dB and 31.74 dB for steering kernel regression described previously. The results of removal of salt-and-pepper noise using two versions of modified median filters were 24.3 and 29.72 dB, see Section 7.2.1 for more details. For 90 percent of missing pixels, the inpainting method that uses ICA-learned dictionary achieved 27.164 dB and kernel regression 27.168 dB. Median-filter-based method, reviewed in Section 7.2.1, performed much worse. For Boat image, with 70 percent of missing pixels, ICA-learned dictionary-based method achieved 29.59 dB, and kernel regression 28.685 dB . The parameters used for kernel regression in all examples were as follows. The size of the local analysis window was 9×9 pixels, the regularization parameter λ' above was set to 1, and the global smoothing parameter h was set to 2.3 (which is the default value in the toolbox[10]). These few examples show that ICA-learned-dictionary-based inpainting performs comparably or little better than the kernel regression method.

7.2 Removal of salt-and-pepper noise

Presented approach to image inpainting can also be used for the image denoising purpose when image is corrupted by additive impulsive noise such as salt and pepper noise. All pixels with

[10]kernel regression toolbox

maximal intensity in the given resolution (when salt noise is considered) or with zero intensity (when pepper noise is considered) are declared as missing. Due to the high efficiency of nonlinear image reconstruction methods and the learned dictionary that provides sparse representation of an image, the small amount of correct pixels that is possibly mistakenly declared as corrupted will not significantly influence the quality of denoising. Yet, such noise corrupted pixel detection scheme is very simple. Impulsive noise belongs to the class of alpha stable processes, has infinite variance and is, in some sense (see Chapter 6), optimally filtered out by means of myriad filters, [5], [43]. In the comparative performance analysis presented below we have used two-dimensional myriad filter with the sliding window of size 5×5 pixels. Larger window would filter out impulsive noise better, but would also cause a loss of details in filtered image. We refer the interested reader to ref. [5], page 337, for other details related to parameter settings for myriad filtering based denoising of grayscale images. Figure 7.2.1a and Figure 7.2.1b show an image chosen from the validation set and corrupted with impulsive noise. Corrupted pixels were selected randomly to respectively occupy 5 and 20 percent of the image. Denoising results obtained by myriad filtering are respectively shown in Figure 7.2.1c and Figure 7.2.1d, while denoising results obtained by inpainting in ICA learned dictionary are shown in Figure 7.2.1e and Figure 7.2.1f. It can be seen that myriad filtering failed to denoise the image well, especially when 20 percent of the image pixels were corrupted. On the contrary, quality of visual perception of the image denoised through inpainting approach with the ICA learned dictionary is very good. Numerical results for all six images from the validation set are shown in Table 7.2.1 and Table 7.2.2 for the corruption level of 5 and 20 percent respectively. It is evident that denoising using myriad filtering becomes very poor when higher percentage of the pixels is corrupted. On the contrary, denoising based on inpainting approach with learned dictionary remains robust even when as much as 80 percent of the pixels are corrupted, see Figure 7.2.1. Obtained results are explained by the fact that filtering is trying to smoothen the image corrupted by additive impulsive noise, while, on the other hand, inpainting is based on nonlinear signal reconstruction approach whereupon pixels corrupted by additive noise are treated as missing pixels. Hence, provided that the learned dictionary yields sparse representation of the image, good reconstruction is possible even when a large number of pixels is corrupted. In the filtering approach, this requires a smoothing window with a large support that causes severe loss of details in the image.

7.2.1 Comparison with new, modified versions of median-filter based approaches to salt-and-pepper noise removal

Several modifications of basic median filters for image denoising were proposed in recent papers [32, 70]. The modified decision-based unsymmetric trimmed median filter proposed in [32] proceeds roughly as follows. An image is processed as usually, in raster-scan order. We assume that values of image pixels are in the range $[0, 255]$ (8-bit image). If the pixel value is different

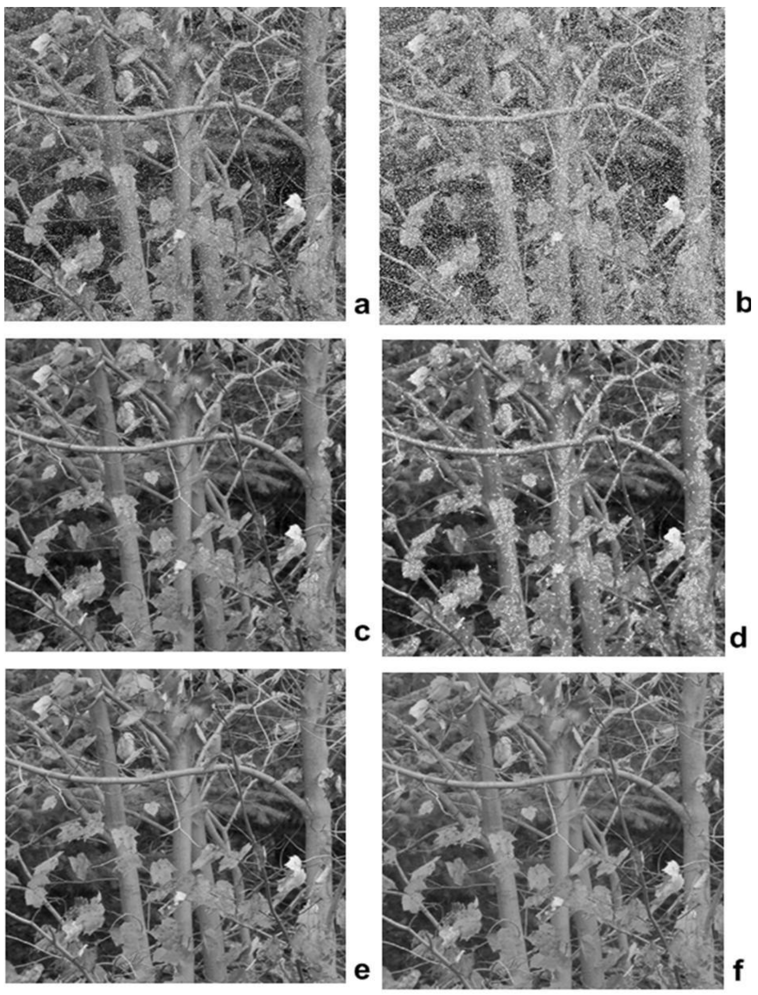


Figure 7.2.1: Examples of noisy image from the validation set with 5 and 20 percent of corrupted pixels, respectively: a) and b). Denoising using two-dimensional myriad filtering with the sliding window of size 5×5 pixels: c) and d). Denoising using inpainting and ICA learned dictionary: e) and f).

from 0 and 255, the pixel is classified as clean. Otherwise, a window is selected around the current pixel (usually, 3×3, 5×5 or 7×7 window). Two cases are possible. If all values in the current window are 0-s and 255-s, the pixel value is replaced with the *mean* value of all pixels in the window. Otherwise, 0-s and 255-s are eliminated from the window, and pixel value is set to the median of the remaining values. This modified median filter performs much better than ordinary medians when the level of noise is high. However, it is still inferior to learned-dictionary-based inpainting. See the end of this subsection for the results.

Another modification was introduced in paper [70]. Namely, the authors proposed to introduce a directional-weighting into the median filtering. Their approach proceeds as follows. In every local window around the given pixel, absolute differences between the center pixel $x_{i,j}$ and its neighbours $x_{i+\Delta i,j+\Delta j}$ are computed:

$$d_{i,j}^{(k)} = \sum_{\Delta i} \sum_{\Delta j} w_{\Delta i,\Delta j} \left| x_{i+\Delta i,j+\Delta j} - x_{i,j} \right| ,$$

Table 7.2.1: Denoising results in terms of the SSIM metric when 5 percent of pixels were corrupted by impulsive noise.

	ICA	**Myriad filtering**
Fig. 7.1.4a	0.998	0.873
Fig. 7.1.4b	0.983	0.93
Fig. 7.1.4c	0.986	0.865
Fig. 7.1.4d	0.999	0.909
Fig. 7.1.4e	0.871	0.854
Fig. 7.1.4f	0.998	0.771
Mean	**0.973**	**0.867**

Table 7.2.2: Denoising results in terms of SSIM metric when 20 percent of pixels were corrupted by impulsive noise.

	ICA	**Myriad filtering**
Fig. 7.1.4a	0.991	0.643
Fig. 7.1.4b	0.977	0.691
Fig. 7.1.4c	0.974	0.569
Fig. 7.1.4d	0.997	0.674
Fig. 7.1.4e	0.916	0.743
Fig. 7.1.4f	0.993	0.287
Mean	**0.975**	**0.601**

where k denotes the direction index, and $1 \leq k \leq 12$ (i.e., there are 12 directions). The sets of horizontal (Δi) and vertical (Δj) offsets in the above sums depend on the given direction index (see paper for details). In order to detect an edge in an image, the direction with minimum sum is selected:

$$k^* = \arg\min_k \left\{ d_{i,j}^{(k)} \,:\, 1 \leq k \leq 12 \right\}.$$

The pixel can be declared as noisy if $d_{i,j}^{(k^*)} > T$, for a given threshold T (this is true both for edges and flat regions). If the pixel is declared as noisy, it is replaced by the trimmed median of the values in the current window (again, values 0 and 255 are excluded when calculating the median). The procedure can be repeated after the whole image is processed, by decreasing the value of the threshold T. T is set to large value initially (the authors suggest initial value $T \approx 500$ for 8-bit images) and decreased by $T \leftarrow cT$, where $c < 1$ (for example, $c = 0.8$). This approach yields better results than [32] since the edges are better preserved. However, learned-dictionary-based approach still performs (much) better. Namely, again we use the two images from Figure 7.1.16 for comparison. For Lena image (Figure 7.1.16a) with 70 percent pixels missing, the method [32] achieves 24.3 dB, while the method [70] achieves 29.72 dB. Still, we have seen in Section 7.1.4 that ICA-learned-dictionary-based approach performs around 31.8 dB, which is significantly better than median-filter-based approach [70]. For Boat image with 70 percent missing pixels, the method [70] achieves 26.58 dB, while we have seen that learned-dictionary-based approach performs about 29.5 dB, which is much better. It is obvious that median filtering approach is inferior to learned-dictionary-based approaches and even more advanced nonlinear filtering approaches like kernel regression (see Section 7.1.4).

7.2.2 Discussion

The inherent trade-off when using median and myriad filters is between better removal of outliers (noisy pixels) and image blurring. Some degree of blurring is unavoidable because of the low-pass nature of median and myriad filters, which is needed to remove high-frequency impulsive noise. On the other hand, the concept of (approximate) sparsity in a dictionary is based on the underlying *structure* of images, i.e. image patches. This enables recovery even when large number of image pixels is corrupted, i.e. missing. The modified decision-based unsymmetric trimmed median filter proposed in [32] is good at discarding outliers (impulse noise-corrupted pixels), however it estimates pixel value based on the (uncorrupted) values in the neighbourhood, which can be only few. Dictionary atoms contain basic structure learned from many patches from training images. Therefore, dictionary-based inpainting/denoising is in this sense non-local, which enables excellent reconstruction from only small number of samples (pixels). The locality of median filters is one of the reasons for their inferior performance. Median filters are better tailored to problems with some other type of impulse noise, different from salt-and-pepper. Namely, it is then very hard to determine which pixels are noise-corrupted, and which are not. Therefore, the inpainting-based approach to denoising would have problems in that case. Here, we restricted ourselves only to inpainting and removal of *salt-and-pepper* noise.

7.3 Feature extraction in proteomics and genomics

In this section we describe one application of sparsity-regularized under-determined source separation methods in bioinformatics. Namely, bioinformatics data analysis is often based on the use of a linear mixture model of a sample, wherein the mixture is composed of components that are generated by the unknown number of sources. For example, components can be generated during disease progression that causes cancerous cells to produce proteins and/or other molecules that can serve as an early indicators (‘biomarkers’) representing disease-related chemical entities. Source separation methods enable extraction of individual components (or groups of individual components) from their mixtures. However, source separation is generally an *unsupervised* technique, meaning that it is not clear which of the extracted components needs to be retained for further analysis. Namely, the goal of source separation in bioinfomatics is the *extraction of features*/components that can later be used for disease prediction or retained for biomarker identification. Many approaches to feature extraction in bioinformatics have been proposed in the literature. We review some of them in Section 7.3.1. In Section 7.3.2, the description of the method published in [60] is presented. Results are presented in Section 7.3.3.

7.3.1 Overview of the literature

In [48], a matrix factorization approach to the classification of infrared spectra is proposed that takes into account class labels. In other words, feature extraction and classification stages are unified. Thus, this concept is classifier specific. It is formulated as the multiclass assignment problem where the number of components equals the number of classes and must be smaller than the number of samples available. In [2], gene expression profile is modelled as a linear superposition of three components comprised of up-regulated, down-regulated and differentially not expressed genes, whereas existence of two fixed thresholds is assumed to enable a decision to which of the three components the particular gene belongs. The thresholds are defined heuristically and in each specific case the optimal value must be obtained by cross-validation. Moreover, the upper threshold c_{u} and the lower one c_{l} are mutually related through $c_{\mathrm{u}} = \frac{1}{c_{\mathrm{l}}}$. As opposed to that, the method proposed in [60] and reviewed in Section 7.3.2 decomposes each sample (experiment) into components comprised of up-regulated, down-regulated and not differentially expressed features using *data adaptive* thresholds. They are based on mixing angles of an innovative linear mixture model of a sample. The method proposed in [93] uses available sample labels (the clinical diagnosis of the experiments) to select component(s), extracted by ICA or nonnegative matrix factorization (NMF), for further analysis. ICA or NMF are used to factorize the whole dataset simultaneously and one selected component (gene expression mode for ICA and metagene for NMF) is used for further analysis related to gene marker extraction. This component cannot be used for classification. Alternatively, basis matrix with labelled column vectors (for ICA) or row vectors (for NMF) can be used for classification in which case the test sample needs to be projected onto the space spanned by the column/row vectors, respectively. However, in this case no feature extraction can be performed. As opposed to ICA/NMF method proposed in [93], the method proposed in [60] extracts disease and control specific component from each sample separately. Since no label information is used in the selection process, extracted components can be used for classification and that was the goal in [60]. The disease specific component can, however, also be retained for further biomarker related analysis as in [93]. The important difference is that by the method proposed in [60] such component can be obtained from each sample separately while the method in [93], as well as in [68, 71, 63], needs the whole dataset. The method [68] uses again ICA (the FastICA algorithm) to factorize the microarray dataset. Extracted components (gene expression modes) were analyzed to discriminate between those with biological significance and those representing noise. However, biologically significant components can be used for further gene marker related analysis but not for classification. The reason is that, as in [93], the whole dataset composed of case and control samples is reduced to several biologically interesting components only. In the extreme case it can happen that there is only one such component. In [71] ICA is used to decompose the whole dataset into components (gene expression modes). As in [93, 68], these components cannot be used for classification. They are used for further decomposition into submodes to identify a regulating network in the problem considered there. We want to emphasize that the component selected as disease specific by the method proposed in [60] can also be interpreted as a sub-mode and used for the similar type of analysis. However, since it is extracted from an individual and labelled sample, it can be used for classification as well. The method in [63] again uses ICA to extract components (gene expression modes). Similarly, as in [93, 68, 71], these components are not used for classification. Instead, they are further analyzed by data clustering to determine biological relevance and extract gene markers. Similar types of comments as those discussed in relation to [93, 68, 71, 63] can also be raised to other methods that use either ICA or NMF to extract components from the whole dataset, [96, 20, 17, 40, 58]. Hence, although related to component selection methods [48, 93, 68, 71, 63], the method proposed in [60] is dissimilar to all of them by the fact that it extracts the most interesting components on a sample (experiment)-by-sample basis. To achieve this, the linear mixture model (LMM) used for components extraction is composed of a test sample and a reference sample representing control and/or case group. Hence, a test sample is, in principle, associated with two LMMs. Each LMM describes a sample as an additive mixture of two or more components. Two of them are selected automatically (no thresholds need to be predefined) as case (disease) and control specific, while the rest are considered neutral i.e. not differentially expressed. Decomposition of each LMM is enabled by enforcing sparseness constraint on the components to be extracted. This implies that each feature (m/z ratio or a gene) belongs to two components at most (disease and neutral, or control and neutral). The model formally presumes that disease specific features are present in the prevailing concentration in disease samples as well as that control specific features are present in prevailing concentration in control samples. However, the features do not have to be expressed equally strong across the whole dataset in order to be selected as a part of disease or case specific components. It is this way due to the fact that decomposition is performed locally (on a sample-by-sample basis). This should prevent losing some important features for classification. Accordingly, the level of expression of indifferent features can also vary between the samples. Thus, postulating one or more components with indifferent features enables their removal that is sample adaptive. As opposed to that, existing methods try to optimize a single threshold for the whole dataset. Geometric interpretation of the LMM based on a reference sample enables automatic selection of disease and control specific components, without using label information. Hence, the selected components can be further used for disease prediction. By postulating existence of one or more components with differentially not expressed features, the complexity of the selected components can be controlled, whereas the overall number of components is selected by cross-validation. Although the feature selection is the main goal of the presented method, component extracted from the sample as disease specific can also be interpreted as a sub-mode as in [93, 68]. Thus, it can be used for further biomarker identification related analysis. We see the linearity of the model used to describe a sample as a potential limitation of a proposed method. Although linear models dominate in bioinformatics, it has been discussed in [63] that nonlinear models might be more accurate description of biological processes. Assumption of an availability of a reference sample might also be seen as a potential weakness. Yet, we have demonstrated that, in the absence of expert information, the reference sample can be obtained by a simple average of all the samples within the same class. The proposed method is demonstrated in Section 7.3.3 on the experimental datasets related to a prediction of ovarian, prostate and colon cancers from protein and gene expression profiles.

7.3.2 Description of the method

Let us suppose that $\mathbf{x}_{\text{control}} \in \mathbb{R}^n$ and $\mathbf{x}_{\text{disease}} \in \mathbb{R}^n$ are two protein or gene expression levels of given samples from a healthy and ill patient, respectively. The elements of a sample are referred to as features (mass/charge (m/z) ratios in the case of protein expression profiles, i.e. genes in the case of gene expression profiles/samples). Figure 7.3.1 shows mass spectrum (a) and gene expression levels (b) of typical samples.

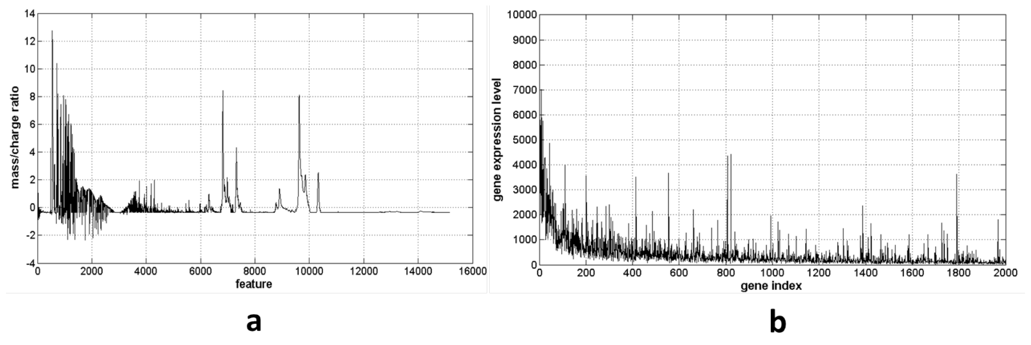


Figure 7.3.1: (a) Protein mass spectrum of a typical sample from the ovarian cancer dataset (see Section 7.3.3) and (b) gene expression levels of a typical sample from the colon cancer dataset (again, see Section 7.3.3).

For a sample $\mathbf{x} \in \mathbb{R}^n$ under consideration, belonging to a *test* group of samples, we assume the following two linear models:

$$\begin{bmatrix} \mathbf{x}_{\text{control}}^T \\ \mathbf{x}^T \end{bmatrix} = \mathbf{A}_{\text{control}}\mathbf{S}_{\text{control}} \tag{7.3.1}$$

and

$$\begin{bmatrix} \mathbf{x}_{\text{disease}}^T \\ \mathbf{x}^T \end{bmatrix} = \mathbf{A}_{\text{disease}}\mathbf{S}_{\text{disease}}, \tag{7.3.2}$$

where $\mathbf{A}_{\text{control}} \in \mathbb{R}^{2\times M}$ and $\mathbf{A}_{\text{disease}} \in \mathbb{R}^{2\times M}$ denote the mixing matrices whose coefficients represent relative amounts of concentration (therefore, the columns of $\mathbf{A}$ are referred to as relative concentration profiles, or concentration vectors) at which related rows of $\mathbf{S}_{\text{control}}$ and $\mathbf{S}_{\text{disease}}$, which represent underlying ‘components’, are present in the mixture samples $\mathbf{x}$, $\mathbf{x}_{\text{control}}$ and $\mathbf{x}_{\text{disease}}$. The number of components M is supposed to be ≥ 2. The rows of the source matrices $\mathbf{S}_{\text{control}}$ and $\mathbf{S}_{\text{disease}}$ are interpreted as components comprised of *disease-specific*, *control*-(healthy) *specific* and differentially not expressed (*indifferent*) features. Depending on the postulated number of components M, different number of indifferent components is assumed. Importance of postulating components with indifferent features is to obtain less complex disease and control specific components used for classification. Components with indifferent features absorb features that do not vary substantially across the sample population. These features are removed automatically from components comprised of disease- and control-specific features. Figure 7.3.2 nicely illustrates that an increased number of indifferent components yields less complex disease-specific components than when using smaller M. Therefore, by the principle of parsimony, larger M should be preferred since it yields less complex components.

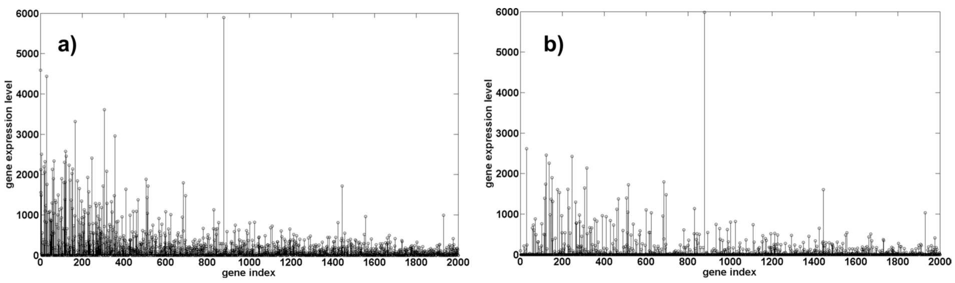


Figure 7.3.2: Colon cancer feature vectors. Components containing disease-specific genes extracted from a cancerous sample with respect to a control reference sample using LMM (7.3.1): a) assumed number of components $M = 2$; b) assumed number of components $M = 4$.

The underlying assumption in the above models is that disease-specific, i.e. control-specific components are present in prevailing concentration in control- and disease-specific samples $\mathbf{x}_{\text{control}}$ and $\mathbf{x}_{\text{disease}}$, respectively. Likewise, it is assumed that control-, i.e. disease-specific components are present in minor concentrations in disease-, i.e. control-specific samples, respectively. Features that are not differentially expressed are assumed to be present in similar concentrations in both disease-specific and control-specific samples. The next crucial assumption in models (7.3.1) and (7.3.2) is that at most two components of $\mathbf{S}$ are active at the same time at the specific feature point (i.e., column of S). Therefore, the source matrices $\mathbf{S}_{\text{control}}$ and $\mathbf{S}_{\text{disease}}$ are (relatively) sparse, and sparse component analysis methods can be used for the estimation of the above mixing models.

Mixing matrix estimation technique described in Chapter 2, Section 2.3.1 can be used to estimate $\mathbf{A}_{\text{control}}$ and $\mathbf{A}_{\text{disease}}$. Namely, it is supposed that there exists a complex transformation of rows of matrices $\mathbf{S}_{\text{control}}$ and $\mathbf{S}_{\text{disease}}$ such that there is at least one point (corresponding to a column of the transformed source matrix), “single-source point”, at which only one (transformed) component is active, i.e. different from zero, and that is true for all components. In the experiments reported in Section 7.3.3, the *analytic continuation* was used to obtain complex representation of real data. The analytic continuation of a vector

$\mathbf{x} \in \mathbb{R}^M$, denoted as $\mathbf{x}_\mathrm{a}$, is defined as $\mathbf{x}_\mathrm{a} = \mathbf{x} + \sqrt{-1}\mathbf{x}_\mathbf{i}$, where $\mathbf{x}_\mathbf{i}$ denotes *Hilbert transform* of $\mathbf{x}$. Hilbert transform of a vector $\mathbf{x}$ is defined as an inverse transformation of a discrete Fourier transform $\mathbf{x}_\mathbf{f}$ of $\mathbf{x}$, wherein coefficients of $\mathbf{x}_\mathbf{f}$ that correspond to negative frequencies are set to zero. Finding the single-source points enables simple estimation of mixing matrix by clustering, as explained in Section 2.3.1. Of course, there are no exact single-source points because of the non-exactness of the model, but it is enough to find *single-dominant* points to enable approximate estimation of the mixing matrix. Single-dominant points (columns of the transformed source matrix) are defined as those whose real and imaginary parts (which are vectors) are within some predefined angle (let us denote it by $\Delta\theta$). $\Delta\theta$ can be set to 1%, 3% or even larger, depending on the number of points that satisfy this criterion of closeness of real and imaginary parts. In real-world datasets, $\Delta\theta$ often has to be set to a value larger than zero because of the non-exactness of the linear model and non-existence of exact single-source points. It should be emphasized that the *transformation to complex domain is only for the mixing matrix estimation purpose*. After the mixing matrix has been estimated, source matrices $\mathbf{S}_\mathrm{control}$ and $\mathbf{S}_\mathrm{disease}$ are found from the original (not transformed) samples.

After estimating the mixing matrix, source matrices $\mathbf{S}_\mathrm{control}$ and $\mathbf{S}_\mathrm{disease}$ are found by sparse recovery methods. Let us denote by $\hat{\mathbf{A}}_\mathrm{control}$ and $\hat{\mathbf{A}}_\mathrm{disease}$ the approximations of the mixing matrices. The approximations of the source matrices are found by

$$\begin{aligned} \hat{\mathbf{S}}_\mathrm{control} &= \arg\min_\mathbf{S} \left\{ \frac{1}{2} \left\| \hat{\mathbf{A}}_\mathrm{control}\mathbf{S} - \left[\begin{array}{c} \mathbf{x}_\mathrm{control}^T \\ \mathbf{x}^T \end{array} \right] \right\|_F^2 + \right. \\ & +\lambda \left\|\mathbf{S}\right\|_1 \} \end{aligned} \tag{7.3.3}$$

and

$$\begin{aligned} \hat{\mathbf{S}}_\mathrm{disease} &= \arg\min_\mathbf{S} \left\{ \frac{1}{2} \left\| \hat{\mathbf{A}}_\mathrm{disease}\mathbf{S} - \left[\begin{array}{c} \mathbf{x}_\mathrm{disease}^T \\ \mathbf{x}^T \end{array} \right] \right\|_F^2 + \right. \\ & +\lambda \left\|\mathbf{S}\right\|_1 \} \end{aligned} \tag{7.3.4}$$

where $\|\mathbf{S}\|_1 = \sum_i \|\mathbf{s}_\mathbf{i}\|_1$, where $\mathbf{s}_\mathbf{i}$ is i-th column of $\mathbf{S}$, and λ is a regularization parameter. The above minimization problems (7.3.3) and (7.3.4) are convex optimization problems (more precisely, ℓ_1-regularized least squares problems) and can be solved by one of many specialized methods. In [60], the iterative soft thresholding (IST) algorithm described in [8] was used, for the following reasons. Although the above minimization problems can be written as a sequence of independent vector optimization problems, it is impractical to solve them one by one since the number of features (columns of $\mathbf{S}$) is usually (very) large. Namely, in bioinformatics problems the number of features is usually of the order of thousands or tens of thousands. The IST algorithm is the first-order algorithm for sparse reconstruction, specifically tailored for the problems of the form (7.3.3) and (7.3.4), that can be easily implemented in 'batch' mode. In other words, instead of solving many independent vector optimization problems one by one, they can be solved simultaneously, using matrix-matrix operations in MATLAB. This, together with fast convergence of the fast IST algorithm presented in [8], makes solving the above optimization problems (7.3.3) and (7.3.4) very fast.

After estimating the factors of the mixing models (7.3.1) and (7.3.2), it remains to select the component(s) that will be retained for classification purpose. The following approach was suggested in [60]. Let us look at the model (7.3.1), where the reference sample is control (healthy) sample. If a current sample $\mathbf{x}$ belongs to the control group, it is expected that it contains similar concentration of control-specific features, whereas related component is postulated to be one of the rows of $\hat{\mathbf{S}}_{\text{control}}$. Therefore, the control-specific (healthy) component is postulated to be the one for which the corresponding mixing vector closes the minimal angle with $\begin{bmatrix} 1 & 0 \end{bmatrix}^T$. If we assume that the model is a good approximation of reality, this mixing vector (closest to $\begin{bmatrix} 1 & 0 \end{bmatrix}^T$) should in fact be close to $\begin{bmatrix} 1 & 1 \end{bmatrix}^T$ (in the case of a healthy reference and healthy test sample). Due to biological variability of the samples within the same group (there are no two equally 'healthy' individuals), the similarity (measured by the angle between the mixing vector and $\begin{bmatrix} 1 & 0 \end{bmatrix}^T$) between the test sample from control group and control reference sample will vary on a sample-by-sample basis. This is especially obvious if we take into account that, in our experiments, reference samples are defined as the mean of the healthy, i.e. cancerous group of samples. The main assumption in our model is that the control-specific (respectively, disease-specific) component (feature) should be significantly present in the control (respectively, disease) reference sample. The criterium for the selection of components comprised of disease- and control-specific features stems directly from this assumption. The extreme (or close to extreme) case could happen when the vector $\begin{bmatrix} 1 & 0 \end{bmatrix}^T$ (or a vector close to it) is one of the mixing vectors. That would mean that the corresponding component (with row index the same as column index of the vector $\begin{bmatrix} 1 & 0 \end{bmatrix}^T$) is chosen as the control-specific. However, the case when one of the mixing vectors closes small angle with $\begin{bmatrix} 1 & 0 \end{bmatrix}^T$ happens rarely (in the case when both the reference and test sample are of the same type (healthy or cancerous)). This justifies the use of models (7.3.1) and (7.3.2) in practice. Similarly to the previous discussion, the component whose corresponding mixing vector closes the maximal angle with $\begin{bmatrix} 1 & 0 \end{bmatrix}^T$ is postulated to be disease-specific.

In complete analogy, the above discussion can be applied when the reference sample is cancerous. The disease-specific component is the one whose corresponding mixing vector closes the minimal angle with $\begin{bmatrix} 1 & 0 \end{bmatrix}^T$, while the control-specific is the one whose corresponding mixing vector closes the maximal angle with $\begin{bmatrix} 1 & 0 \end{bmatrix}^T$. In this way, for every sample, four components are selected, which together form four sets of labelled feature vectors $\left\{\mathbf{s}^{\text{control}}_{\text{control ref.};i}, y_i\right\}_{i=1}^N$, $\left\{\mathbf{s}^{\text{disease}}_{\text{control ref.};i}, y_i\right\}_{i=1}^N$, $\left\{\mathbf{s}^{\text{disease}}_{\text{disease ref.};i}, y_i\right\}_{i=1}^N$ and $\left\{\mathbf{s}^{\text{control}}_{\text{disease ref.};i}, y_i\right\}_{i=1}^N$. Here, y_i, for $i = 1, \ldots, N$, denote *sample labels* (healthy or cancerous). Classifiers can be trained on every set of labelled feature vectors, and the classifier with highest accuracy achieved through cross-validation is retained for disease diagnosis. The component comprised of disease-specific features can also be retained for further biomarker identification procedure.

The classifier training and cross-validation (CV, see [47]) proceed as follows. Two-fold cross validation is used. The set of all available samples is divided into two sets, training and test, with the same number of samples (in the case of even number of samples). Classifier is trained on the samples from the training set, and is then tested on the test set. This procedure is repeated certain number of times (100, in our case). The classifier with the highest *accuracy* over 100 realizations is retained for classification purpose. Accuracy is defined as the average of mean values of sensitivity and specificity over 100 realizations. *Sensitivity* of the classifier is defined as the proportion of diseased samples that are classified as such (diseased), while *specficity* is defined as the proportion of healthy samples that are correctly classified as such. The results obtained with two-fold cross validation are more realistic than with, for example, three- or ten-fold CV. This is the reason for using two-fold CV in [60]. Three classifiers were tested: linear support vector machine (SVM, [57]) classifier and nonlinear SVM with gaussian and polynomial kernels. We refer the reader to the paper [60] for more details which are not directly connected to the topic of the thesis. In the following, we list the results obtained using the described method on three experimental datasets.

The presented two-stage SCA-based approach for feature extraction/ component selection is presented in concise form in Algorithm 7.1.

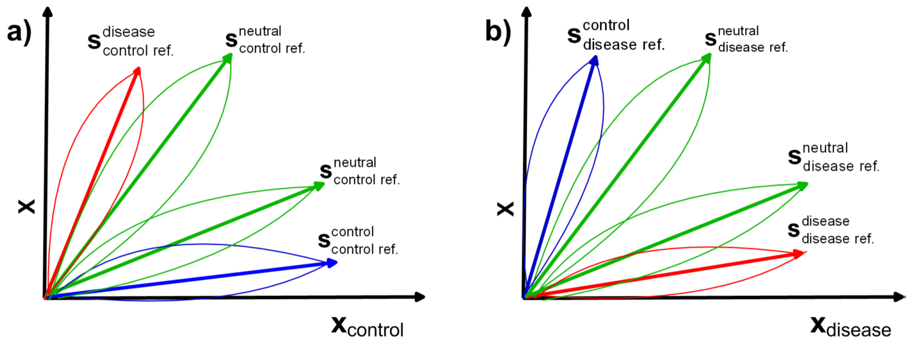

a)
$s^{disease}_{control\ ref.}$
$s^{neutral}_{control\ ref.}$
$s^{neutral}_{control\ ref.}$
$s^{control}_{control\ ref.}$
x
$x_{control}$
b)
$s^{control}_{disease\ ref.}$
$s^{neutral}_{disease\ ref.}$
$s^{neutral}_{disease\ ref.}$
$s^{disease}_{disease\ ref.}$
x
$x_{disease}$

Figure 7.3.3: Geometrical interpretation of the linear mixture model. The figure shows mixing vectors (*concentration vectors*) of the linear mixture model composed of control reference sample and a test sample, (7.3.1) and a), i.e. disease reference sample and a test sample, (7.3.2) and b). Features (m/z ratios or genes) with prevailing concentration in the disease sample are supposed to be highly expressed in the component *associated to* the red color mixing vector. Likewise, features with prevailing concentration in the control sample are suposed to be highly expressed in the component associated to the blue color mixing vector. Features that are not differentially expressed are linearly combined into one or more components associated to green color mixing vectors. The x-axis in a) and b) is associated to the reference sample, while the y-axis is associated to the test sample.

Algorithm 7.1 The algorithm for feature extraction/component selection based on the linear mixture model with a reference

```
Inputs. $\{\mathbf{x_i} \in \mathbb{R}^n, y_i \in \{-1,1\}\}_{i=1}^N$ pairs of samples and sample labels, where $n$ represents the
number of feature points ($m/z$ ratios or genes); $\mathbf{x}_{\text{control}}$ and $\mathbf{x}_{\text{disease}}$, representing control and
disease reference samples, respectively.

• Nested two-fold cross-validation (CV). Parameters: single component points (SCP-s) selection threshold
  $\Delta\theta \in \left\{\frac{\pi}{180}, \frac{3\pi}{180}, \frac{5\pi}{180}\right\}$; regularization constant
  $\lambda \in \left\{10^{-2}\lambda_{\max}, 10^{-4}\lambda_{\max}, 10^{-6}\lambda_{\max}\right\}$, where $\lambda_{\max}$ denotes the smallest value of the
  regularization parameter for which the solution of the problem (7.3.3) or (7.3.4) is equal to zero;
  the number of components $M \in \{2,3,4,5\}$; parameters of a selected classifier.
    – Components selection from a mixture samples:
        1. for every $\mathbf{x} \in \{\mathbf{x_i} \in \mathbb{R}^n\}_{i=1}^N$ form a linear mixture models (LMM-s) (7.3.1) and (7.3.2).
        2. For LMM-s (7.3.1) and (7.3.2) select a set of single component points (SCP-s) for given $\Delta\theta$.
        3. Use clustering on the sets of SCP-s to estimate the mixing matrices $\mathbf{A}_{\text{control}}$ and
           $\mathbf{A}_{\text{disease}}$ for given $M$.
        4. Estimate source matrices $\mathbf{S}_{\text{control}}$ and $\mathbf{S}_{\text{disease}}$ by solving (7.3.3) and (7.3.4),
           respectively, for given regularization parameter $\lambda$.
        5. Using minimal and maximal mixing angles estimated from the obtained approximations of mixing
           matrices $\mathbf{A}_{\text{control}}$ and $\mathbf{A}_{\text{disease}}$, following the logic illustrated in Figure 7.3.3,
           select disease- and control-specific components:
           $\mathbf{s}^{\text{disease}}_{\text{control ref.};i}$, $\mathbf{s}^{\text{control}}_{\text{control ref.};i}$, $\mathbf{s}^{\text{control}}_{\text{disease ref.};i}$, $\mathbf{s}^{\text{disease}}_{\text{disease ref.};i}$.
    – end of components selection.
• end of nested two-fold CV.
```

7.3.3 Results

7.3.3.1 Ovarian cancer prediction from protein mass spectra

Low resolution surface-enhanced laser desorption ionization time-of-flight (SELDI-TOF) mass spectra of 100 controls and 100 cases have been used for ovarian cancer prediction study [86]. See also the website of the clinical proteomics program of the National Cancer Institute (NCI)[11], where the used dataset is labelled as "Ovarian 4-3-02". All spectra were baseline corrected. Thus, some intensities have negative values. Table 7.3.1 presents the best result obtained by the proposed SCA based component selection method, together with results obtained for the same dataset by competing methods reported in cited references as well as by predictive factorization method proposed in [48].

Described SCA method has been used to extract four sets of components with the overall num-

[11]National Cancer Institute clinical proteomics program, http://home.ccr.cancer.gov/ncifdaproteomics/ppatterns.asp

Table 7.3.1: Comparative performance results in ovarian cancer prediction. Sensitivities and specificities were estimated by 100 two-fold cross validations (standard deviations are in brackets)

Method	**Sensitivity/specificity**
Proposed method, $M = 3$, $\Delta\theta = 5^\circ$, $\lambda = 10^{-4}\lambda_{\max}$, linear SVM	Sensitivity: 96.2 (2.7) %; specificity: 93.6 (4.1) %; accuracy: 94.9%. Control specific component extracted with respect to a cancer reference sample.
Proposed method, $M = 4$, $\Delta\theta = 3^\circ$, $\lambda = 10^{-6}\lambda_{\max}$, linear SVM	Sensitivity: 95.4 (3) %; specificity: 94 (3.7) %; accuracy: 94.7%. Control specific component extracted with respect to a cancer reference sample.
[48]	Sensitivity: 81.4 (7.1) %; specificity: 71.7 (6.6) %
[86]	Sensitivity: 100%; specificity: 95% (*one partition only*: 50/50 training; 66/50 test)
[6]	Accuracy averaged over 10 ten-fold partitions: 98 − 99% (std: 0.3 − 0.8)
[66]	Sensitivity: 98%; specificity: 95%, two-fold CV with 100 partitions
[110]	Average error rate of 4.1%, three-fold CV

ber of components M assumed to be 2, 3, 4 and 5. Figure 7.3.4 shows sensitivities and specificities estimated by 100 independent two-fold cross-validations using linear SVM classifier which yielded the best results compared against nonlinear SVM classifiers based on polynomial and RBF kernels. Performance improvement is visible when assumed number of components is increased from 2 to 3, 4 or 5. The error bars are dictated by the sample size and would decrease with a larger sample. Thus, the mean values should be looked at to observe the trend in performance as a function of M.

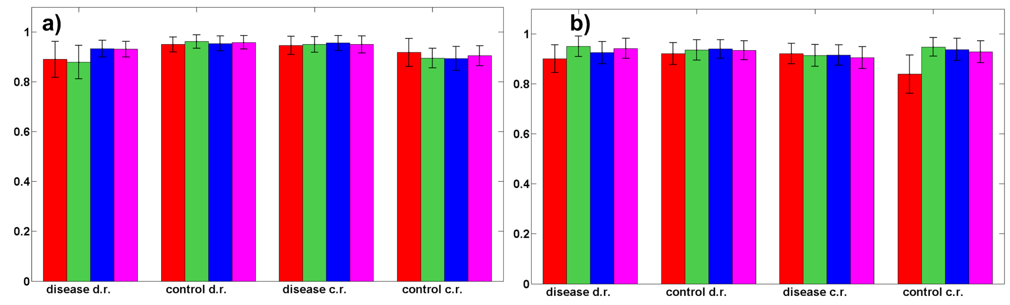


Figure 7.3.4: Ovarian cancer prediction. Sensitivities (a) and specificities (b) (with standard deviations as error bars) estimated in ovarian cancer prediction from protein expression levels using 100 independent two-fold cross-validations and linear SVM classifier. Four sets of selected components were extracted by SCA-based factorization using LMM-s (7.3.1) and (7.3.2) with control reference (c.r.) and disease reference (d.r.) samples respectively, where the overall number of components M has been set to 2 (red bars), 3 (green bars), 4 (blue bars) and 5 (magenta bars). Optimal values of parameters λ and $\Delta\theta$, obtained through nested CV, were used for each M. Performance improvement is visible when the number of components is increased from 2 to 3, 4 or 5.

The best result (shown in Table 7.3.1) has been obtained with the linear SVM classifier for $M = 3$ with sensitivity of 96.2% and specificity of 93.6%, but very similar results have been obtained for several combinations of parameters M, $\Delta\theta$ and λ, see Figure 7.3.4, most notably $M = 4$ (see second row in Table 7.3.1). As seen in Table 7.3.1, only [66] reported better result for a two-fold cross-validation with the same number of partitions. There, a combination of genetic algorithm and k-nearest neighbours method, originally developed for mining of high-dimensional microarray gene expression data, has been used for analysis of proteomics data. However, the method [66] is tested on proteomic ovarian cancer dataset only, while the method presented here exhibited excellent performance in prediction of prostate cancer from proteomic data (reported in Section 7.3.3.2), as well as on colon cancer from genomic data (presented in Section 7.3.3.3). The method shown in [86] used 50 samples from the control group and 50 samples from the ovarian cancer group to discover a pattern that discriminated cancer from non-cancer group. This pattern has then been used to classify an independent set of 50 samples with ovarian cancer and 66 samples unaffected by ovarian cancer. In [6], a fuzzy rule based classifier fusion is proposed for feature selection and classification (diagnosis) of protein mass spectra based ovarian cancer. Demonstrated accuracy of $98-99\%$ has been estimated through 10 ten-fold cross-validations (as opposed to 100 two-fold cross-validations used here). Moreover, as demonstrated in Section 7.3.3.2 and Section 7.3.3.3, the method presented here exhibited good performance on diagnosis of prostate and colon cancers from proteomic and gene expression levels, respectively. In [110], a clustering based method for feature selection from mass spectrometry data is derived by combining k-means clustering and genetic algorithm. The method exhibited an accuracy of 95.8% (error rate 4.1%), but this has been assessed through three-fold cross-validations (as opposed to two-fold cross-validations used here).

7.3.3.2 Prostate cancer

Low resolution SELDI-TOF mass spectra of 63 controls: no evidence of cancer with prostate-specific antigen (PSA)< 1, and 69 cases (prostate cancers): 26 with $4 < \mathrm{PSA} < 10$ and 43 with PSA> 10, have been used for prostate cancer prediction study [85]. There are additional 190 control samples with benign cancer ($4 < \mathrm{PSA} < 10$) available as well (see the website of the clinical proteomics program of the NCI, see footnote 11), in dataset labelled as "JNCI_Data_7-3-02". However, in the two-class comparative performance analysis problem reported here these samples were not used. Proposed SCA-based method has been used to extract four sets of components with the overall number of components M assumed to be 2, 3, 4 and 5. The best result has been achieved for $M = 5$ with sensitivity of 97.6% and specificity of 99%, but very similar results have been obtained for several combinations of parameters M, $\Delta\theta$ and λ, (see Figure 7.3.5).

Table 7.3.2 presents two best results achieved by the presented SCA-based approach to component selection together with the results obtained by competing methods reported in cited

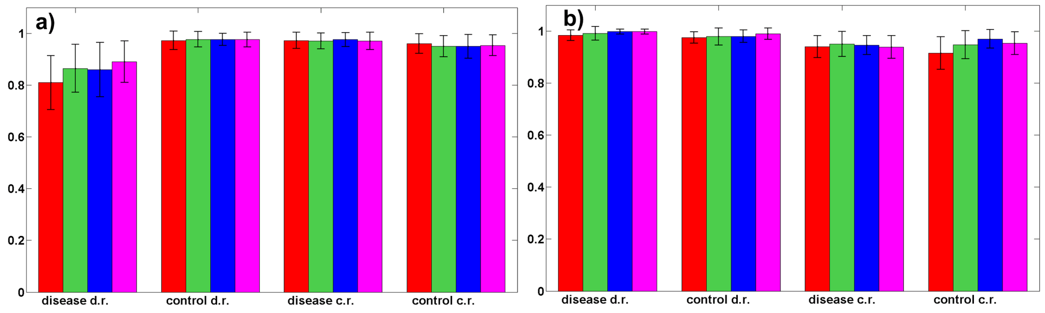


Figure 7.3.5: Prostate cancer prediction. Sensitivities (a) and specificities (b) (with standard deviations as error bars) estimated in prostate cancer prediction from protein expression levels using 100 independent two-fold cross-validations and linear SVM classifier. Four sets of selected components were extracted by SCA-based factorization using LMM-s (7.3.1) and (7.3.2) with control reference (c.r.) and disease reference (d.r.) samples respectively, where the overall number of components M has been set to 2 (red bars), 3 (green bars), 4 (blue bars) and 5 (magenta bars). Optimal values of the parameters λ and $\Delta\theta$, obtained through nested CV, were used for each M. Performance improvement is visible when the number of components is increased from 2 to 3, 4 or 5.

references.

Table 7.3.2: Comparative performance results in prostate cancer prediction. Sensitivities and specificities were estimated by 100 two-fold CV-s (standard deviations are in brackets).

Method	**Sensitivity/specificity**
Proposed method, $M = 5$, $\Delta\theta = 1^\circ$, $\lambda = 10^{-4}\lambda_{\max}$, linear SVM	Sensitivity: 97.6 (2.8) %; specificity: 99 (2.2) %; accuracy: 98.3%. Control specific component extracted with respect to a cancer reference sample.
Proposed method, $M = 4$, $\Delta\theta = 1^\circ$, $\lambda = 10^{-4}\lambda_{\max}$, linear SVM	Sensitivity: 97.7 (2.3) %; specificity: 99.8 (2.4) %; accuracy: 97.9%. Control specific component extracted with respect to a cancer reference sample.
[48]	Sensitivity: 86 (6.6) %; specificity: 67.8 (12.9) %; accuracy: 76.9%
[85]	Sensitivity: 94.7%; specificity: 75.9%; accuracy: 85.3%. 253 benign and 69 cancers. Results were obtained on independent test set comprised of 38 cancers and 228 benign samples.
[109]	Sensitivity: 97.1%; specificity: 96.8%; accuracy: 97%. 253 benign and 69 cancers. Cross validation details not reported
[110]	Average error rate of 28.97 on four class problem with three-fold cross validation.

Linear SVM classifier yielded the best results when compared against nonlinear SVM classifiers based on polynomial and RBF kernels. According to Table 7.3.2, comparable result (although slightly worse) is in the reference [109] only. The method [109] is proposed for analysis of mass spectra for screening of prostate cancer. The system is composed of three stages: a feature selection using statistical significance test, a classification by radial basis function and probabilistic neural networks and an optimization of the results through the receiver-operating-characteristic analysis. The method achieved sensitivity 97.1% and specificity 96.8%, but the cross-validation setting has not been described in details. In [85], the training group has been used to discover a pattern that discriminated cancer from non-cancer group. This pattern has then been used to classify an independent set of 38 patients with the prostate cancer and 228 patients with benign conditions. The obtained specificity is low. The predictive matrix factorization method [48] yielded significantly worse result than the method proposed here. In [110] a clustering based method for feature selection from mass spectrometry data is derived combining k-means clustering and a genetic algorithm. Despite the three-fold cross-validation, the reported error was 28.97%. Figure 7.3.5 shows sensitivities and specificities estimated by 100 independent two-fold cross-validations using linear SVM classifier on components selected by the method presented here. For each M the optimal values of the parameters λ and $\Delta\theta$ (obtained by cross-validation) have been used to obtain results shown in Figure 7.3.5. Increasing a postulated number of components from 2 to 5 increased accuracy from 97.4% to 98.3%. Thus, better accuracy is achieved with the smaller number of features (m/z ratios) contained in selected components.

7.3.3.3 Colon cancer

Gene expression profiles of 40 colon cancer and 22 normal colon tissue samples obtained by an Affymetrix oligonucleotide array [3], have also been used for validation and comparative performance analysis of proposed feature extraction method. Gene expression profiles have been downloaded from[12]. Original data produced by oligonucleotide array contained more than 6500 genes but only 2000 high-intensity genes have been used for cluster analysis in [3] and are provided for download on the cited website. The proposed SCA-based approach to feature extraction/component selection has been used to extract four sets of components with up- and down-regulated genes and with the overall number of components M assumed to be 2, 3, 4 and 5. The linear SVM classifier has been applied to groups of four sets of selected components extracted from gene expression levels for specific combinations of parameters $\Delta\theta$, λ and M. The best result in terms of sensitivity and specificity for each M has been selected and shown in Figure 7.3.6.

An increased number of postulated components M did not decrease the accuracy but it yielded components selected for classification with reduced number of genes. This is verified in Figure 7.3.2, which shows component with up-regulated genes extracted from a cancer labelled sample w.r.t. the control reference for assumed number of components $M = 2$ and $M = 4$. Thus, it is confirmed again that an increased M yields less complex components that (following the principle of parsimony), should be preferred over the more complex ones obtained by smaller M. In order

[12]Data pertaining to the article [3],
http://genomics-pubs.princeton.edu/oncology/affydata/index.html

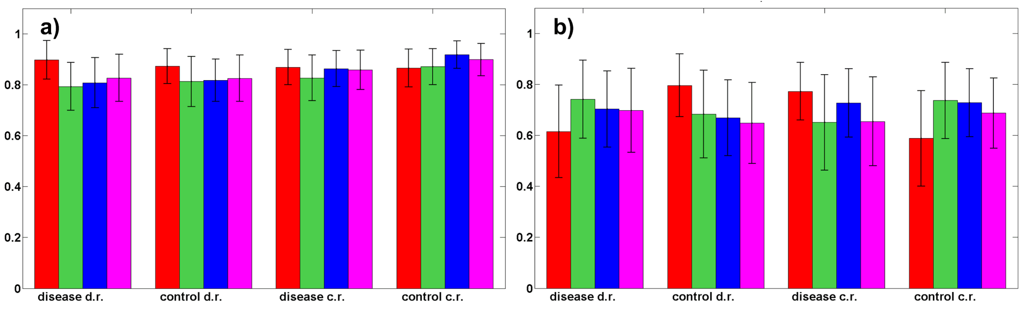


Figure 7.3.6: Colon cancer prediction. Sensitivities (a) and specificities (b) (with standard deviations as error bars) estimated in colon cancer prediction from gene expression levels using 100 independent two-fold cross-validations and linear SVM classifier. Four sets of selected components were extracted by SCA-based factorization using LMM-s (7.3.1) and (7.3.2) with control reference (c.r.) and disease reference (d.r.) samples respectively, where the overall number of components M has been set to 2 (red bars), 3 (green bars), 4 (blue bars) and 5 (magenta bars). Optimal values of the parameters λ and $\Delta\theta$, obtained through nested CV, were used for each M. Increasing the number of components M did not decrease prediction accuracy , but did reduce the number of features (genes) in components used for classification (see Figure 7.3.2).

to (possibly) increase the prediction accuracy, we have applied nonlinear, polynomial and RBF SVM classifiers to two groups of four sets of components that yielded the best results with the linear SVM classifier: $M = 2$ ($\Delta\theta = 10$) and $M = 4$ ($\lambda = 10^{-2}\lambda_{\text{max}}$ and $\Delta\theta = 50$). The polynomial SVM classifier has been cross-validated for degree of the polynomial equal to $d = 2$, 3 and 4. The RBF SVM classifier $\kappa(x,y) = \exp\left(-\|x-y\|_2^2/\left(2\sigma^2\right)\right)$ has been cross-validated for the variance σ^2 in the range $5\cdot 10^2$ to $1.5\cdot 10^3$ in steps of 10^2. The best result has been obtained with $\sigma^2 = 1.2\cdot 10^3$ for $M = 2$ and with $\sigma^2 = 1.0\cdot 10^3$ for $M = 4$. Achieved accuracy is comparable with the accuracy obtained by other state-of-the-art results reported. That is shown in Table 7.3.3.

Predictive matrix factorization method [48] yielded slightly better results here, but it has shown significantly worse result in the cases of ovarian (see Table 7.3.1) and prostate (see Table 7.3.2) cancers. Gene discovery method [2] has been applied for three values of the threshold $c_{\text{u}} \in \{2, 2.5, 3\}$ used to select up-regulated genes. Maximum a posteriori probability has been used for an assignment of genes to each of the three components containing up-, down-regulated and differentially not expressed genes. Thus, for each threshold value, two components were obtained for training of a classifier. The logarithm with base 10 has been applied to gene folding values prior to gene discovery/selection. The best result reported in Table 7.3.3 has been obtained for component containing up-regulated genes with $c_{\text{u}} = 2.0$ and an RBF SVM classifier, whereas σ^2 has been cross-validated in the range 10^2 to 10^3 in steps of 10^2. The best result has been obtained for $\sigma^2 = 5\cdot 10^2$. The gene discovery method [2] slightly outperformed the method proposed here. However, as opposed to the proposed method, the gene discovery method [2] is not applicable to the analysis of mass spectra. The gene selection method in [89]

Table 7.3.3: Comparative performance results in colon cancer prediction. Sensitivities and specificities were estimated by 100 two-fold CV-s (standard deviations are in brackets).

Method	**Sensitivity/specificity**
Proposed method, $M = 2$, $\Delta\theta = 1^\circ$, RBF SVM ($\sigma^2 = 1200$, $C = 1$)	Sensitivity: 90.8(5.5)%; specificity: 79.4(9.8)%; accuracy: 85.1%. Control specific component extracted with respect to a cancer reference sample.
Proposed method, $M = 4$, $\Delta\theta = 5^\circ$, $\lambda = 10^{-2}\lambda_{\max}$, RBF SVM ($\sigma^2 = 1000$, $C = 1$)	Sensitivity: 89.8(6.2)%; specificity: 78.6(12.8)%; accuracy: 84.2%. Control specific component extracted with respect to a control reference sample.
[48]	Sensitivity: 89.7 (6.4) %; specificity: 84.3 (8.4) %; accuracy: 87%. 100 two-fold cross-validations.
[2]	Sensitivity: 92.1 (4.7) %; specificity: 85 (10.1) %; accuracy: 88.55%. 100 two-fold cross-validations. $c_u = 2$
[3]	Sensitivity: 92 – 95%, calculated from Figure 5. Specificity not reported.
[89]	Accuracy: 85%. Cross-validation details not reported.
[4]	Accuracy: 82.5%, ten-fold cross-validation (RFE with linear SVM).
[50]	Accuracy: 88.84%, two-fold cross-validation (RFE with linear SVM and optimized penalty parameter C).

is model driven, i.e. trying to take into account genes' group behaviours and interactions by developing an ensemble dependence model (EDM). The microarray dataset is clustered first. The EDM is based on modelling dependencies that represent inter-cluster relationships. Intercluster dependence matrix is the basis for discrimination between cancerous and non-cancerous samples. Classification accuracy of 85% reported in [89] is very close to the one obtained by the SCA-based method presented here. However, while SCA-based performance has been assessed through two-fold cross-validation, no cross-validation details were reported in [89]. Similarly, sensitivity had to be estimated indirectly from Figure 5 in [3]. The method in [4] combines a recursive feature extraction and the linear SVM to yield accuracy of 82.5%. This is also less accurate than what has been achieved by the method proposed here. Moreover, the accuracy reported in [4] has been assessed by tenfold cross-validation only, and that is known to yield too optimistic performance assessment. In this regard, accuracy reported in [50] can be taken as more realistic since it has been assessed by two-fold cross-validation. This method, as [4], again combines recursive feature elimination with SVM, but it is additionally taking into account the parameter C. Reported accuracy of 88.84% is slightly better than the one obtained by the method presented here. However, the proposed method is classifier independent and, as demonstrated in Section 7.3.3.1 and Section 7.3.3.2, it yields good results on cancer diagnosis from proteomic datasets as well.

7.3.3.4 Conclusion

The results presented in this section indicate that the proposed model and the sparse component analysis method used to estimate its parameters, despite its simplicity, yield very good results, comparable or better than other competing methods from the literature.

7.4 Summary

In this chapter we have presented results of the comparative performance analysis of the dictionary learning methods on the problems of inpainting and removal of salt-and-pepper noise in natural images. Some classical methods for image inpaiting and removal of salt-and-pepper noise were also included in the comparison. Salt-and-pepper noise removal problem was reduced to the inpainting problem. The inpainting method using the dictionary for sparse representation of natural images' patches learned by ICA, in combination with the robust Smoothed ℓ_0 method for sparse recovery, yielded excellent results on problems with random and some structured (thin lines) patterns of missing pixels. Both grayscale and color images were considered. Reported results are better than or comparable to the results obtained with competing methods from the literature.

Also, in this chapter we have presented a novel method for feature extraction in bioinformatics, more precisely proteomics and genomics. It is based on a novel type of linear mixture model with a reference sample that, through sparsity constrained factorization, enables automatic feature extraction on a sample-by-sample basis. Therein, sample label information is not used. This allows the use of extracted features for the training of a classifier. The proposed method is demonstrated on publicly available experimental datasets related to the prediction of ovarian, prostate and colon cancers from protein and gene expression profiles. Obtained results are better than or comparable to those obtained with competing methods.

Chapter 8

Summary

This chapter presents a summary of contributions of the thesis.

The first contribution is the use of independent component analysis (ICA) for dictionary learning for natural images, and the application of the learned dictionary in image inpainting and removal of salt-and-pepper noise problems. The salt-and-pepper noise removal problem was *reduced to the inpainting problem* by declaring all noise-corrupted pixels as missing, as described in Section 7.2. This is ***the second contribution*** of the thesis.

The dictionary is learned on *patches* (small rectangular parts) of natural images, which is a common approach for dictionary learning for natural images. This approach has a biological background and motivation, as described in Section 3.1. The proposed method was compared to several representatives of state-of-the-art methods for image inpainting and salt-and-pepper noise removal in Chapter 7. The results obtained with the proposed method are comparable or better than the competing methods used in this comparative performance analysis. This is especially the case in removal of salt-and-pepper noise problems. Namely, the method that uses the dictionary learned by ICA performs *much better* than commonly used methods for removal of impulse noise based on median and myriad filters. It also compares favorably with kernel regression-based method. In inpainting experiments with random pattern of missing pixels, only Fields of Experts (FoE) method, reviewed in Section 5.1.3, yielded better results than the proposed method, but with much higher computational complexity.

The proposed method for inpainting and removal of salt-and-pepper noise was used both on grayscale and color images. The extension to color images is straightforward and was described in Section 7.1.3. Again, obtained results are comparable to those obtained with a state-of-the-art method for color image inpainting.

In inpainting experiments, both random and some structured patterns of missing pixels were used. Since the method is patch-based, it is better suited to inpainting of random pattern of missing pixels, and removal of salt-and-pepper noise. However, as demonstrated in Section 7.1.2, it can also be used for inpainting of thin lines of missing pixels, or text inpainting. The method is limited by patch size in the case when larger missing regions are present. In this case, there are more specialized methods in the literature that perform much better.

The above described results were presented in publications [34, 35]. Also, the `MATLAB` codes for reproducing the results presented in these publications are available on author's webpage[1].

The third contribution of the thesis is related to the novel method for feature extraction in bioinformatics. It is based on a novel type of linear mixture model with a reference sample that, through sparsity constrained factorization, enables automatic feature extraction on a sample-by-sample basis. Therein, sample label information is not used. This allows the use of extracted features for training of a classifier. As opposed to that, existing matrix factorization methods use the whole dataset to extract disease specific features by using label information. This prevents the use of extracted features for the training of a classifier. The proposed method is demonstrated on publicly available experimental datasets related to the prediction of ovarian, prostate and colon cancers from protein and gene expression profiles. Obtained results are better than or comparable to those obtained with competing methods. The method was described in the paper [60].

[1]http://www.lair.irb.hr/ikopriva/marko-filipovi.html

Bibliography

[1] M. Aharon, M. Elad, and A. Bruckstein, *K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation*, IEEE Trans. Signal Process., 54 (2006), pp. 4311 – 4322.

[2] M. Alfo, A. Farcomeni, and L. Tardella, *A three component latent class model for semiparametric gene discovery*, Statistical Applications in Genetics and Molecular Biology, 10 (2011).

[3] U. Alon, N. Barkai, D. A. Notterman, K. Gish, S. Ybarra, D. Mack, and A. J. Levine, *Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays*, Proceedings of the National Academy of Sciences USA, 96 (1999), pp. 6745 – 6750.

[4] C. Ambroise and G. J. MacLachlan, *Selection bias in gene extraction on the basis of microarray gene-expression data*, Proceedings of the National Academy of Sciences USA, 99 (2002), pp. 6562 – 6566.

[5] G. R. Arce, *Nonlinear signal processing: A statistical approach*, John Wiley and & Sons, 1st ed., 2005.

[6] A. Assareh and L. G. Volkert, *Fuzzy rule based classifier fusion for protein mass spectra based ovarian cancer diagnosis*, in Proc. IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB), 2009, pp. 193 – 199.

[7] R. Baraniuk, M. Davenport, R. DeVore, and M. Wakin, *A simple proof of the restricted isometry property for random matrices*, Constructive Approximation, 28 (2008), pp. 253–263.

[8] A. Beck and M. Teboulle, *A fast iterative shrinkage-thresholding algorithm for linear inverse problems*, SIAM J. Imag. Sci., 2 (2009), pp. 183 – 202.

[9] A. J. Bell and T. J. Sejnowski, *An information-maximization approach to blind separation and blind deconvolution*, Neural Computation, 7 (1995), pp. 1129–1159.

[10] ——, *The "independent components" of natural scenes are edge filters*, Vision research, 37 (1997), pp. 3327 – 3338.

[11] T. BLUMENSATH AND M. DAVIES, *Iterative thresholding for sparse approximations*, Journal of Fourier Analysis and Applications, 14 (2008), pp. 629–654. 10.1007/s00041-008-9035-z.

[12] T. BLUMENSATH AND M. E. DAVIES, *On the difference between Orthogonal Matching Pursuit and Orthogonal Least Squares*. Unpublished, 2007.

[13] ——, *Gradient pursuits*, IEEE Trans. Signal Process., 56 (2008), pp. 2370 – 2382.

[14] R. BOSCOLO, H. PAN, AND V. P. ROYCHOWDHURY, *Independent component analysis based on nonparametric density estimation*, IEEE Trans. on Neural Netw., 15 (2004), pp. 55 – 65.

[15] S. BOYD AND L. VANDENBERGHE, *Convex optimization*, Cambridge University Press, 1st ed., 2004.

[16] A. M. BRUCKSTEIN, M. ELAD, AND D. L. DONOHO, *From sparse solutions of systems of equations to sparse modelling of signals and images*, SIAM Review, 51 (2009), pp. 34 – 81.

[17] J. P. BRUNET, P. TAMAYO, T. R. GOLUB, AND J. P. MESIROV, *Metagenes and molecular pattern discovery using matrix factorization*, PNAS USA, 101 (2004), pp. 4164 – 4169.

[18] E. J. CANDES AND T. TAO, *Decoding by linear programming*, IEEE Trans. Inform. Theory, 51 (2004), pp. 4203 – 4215.

[19] J.-F. CARDOSO, *Infomax and maximum likelihood for blind source separation*, IEEE Signal Processing Letters, 4 (1997), pp. 112 – 114.

[20] P. CARMONA-SAEZ, R. D. PASCUAL-MARQUI, F. TIRADO, J. M. CARAZO, AND A. PASCUAL-MONTANO, *Biclustering of gene expression data by non-smooth non-negative matrix factorization*, BMC Bioinformatics, 7 (2006), p. 78.

[21] P. COMON, *Independent component analysis, a new concept?*, Signal processing, 36 (1994), pp. 287 – 314.

[22] P. COMON AND C. JUTTEN, eds., *Handbook of blind source separation: Independent component analysis and applications*, Elsevier, 1st ed., 2010.

[23] T. M. COVER AND M. J. THOMAS, *Elements of information theory*, Wiley, 1st ed., 1991.

[24] D. L. Donoho, M. Elad, and V. N. Temlyakov, *Stable recovery of sparse overcomplete representations in the presence of noise*, IEEE Trans. Inf. Theory, 52 (2006), pp. 6 – 18.

[25] R. Durrett, *Probability: Theory and examples*, Wadsworth & Brooks, 1st ed., 1991.

[26] A. Eftekhari, M. Babaie-Zadeh, C. Jutten, and H. A. Moghaddam, *Robust-SL0 for stable sparse representation in noisy setting*, in IEEE International Conference on Acoustics, Speech and Signal Processing, 2009.

[27] M. Elad, *Optimized projections in compressed sensing*, IEEE Trans. Signal Process., 55 (2007), pp. 5695 – 5702.

[28] M. Elad and M. Aharon, *Image denoising via sparse and redundant representations over learned dictionaries*, IEEE Transactions on Image Processing, 15 (2006), pp. 3736 – 3745.

[29] M. Elad, J.-L. Starck, P. Querre, and D. L. Donoho, *Simultaneous cartoon and texture image inpainting using morphological component analysis (MCA)*, Appl. Comput. Harmon. Anal., 19 (2005), pp. 340 – 358.

[30] K. Engan, S. O. Aase, and J. H. Husoy, *Multi-frame compression: Theory and design*, Signal Process., 80 (2000), pp. 2121 – 2140.

[31] J. Eriksson and V. Koivunen, *Identifiability, separability and uniqueness of linear ICA models*, IEEE Signal Process. Lett., 11 (2004).

[32] S. Esakkirajan, T. Veerakumar, A. N. Subramanyam, and C. H. PremChand, *Removal of high-density salt and pepper noise through modified decision based unsymmetric trimmed median filter*, IEEE Signal Process. Lett., 18 (2011), pp. 287 – 290.

[33] T. Ferguson, *A course in large sample theory*, Chapman & Hall, 1st ed., 1996.

[34] M. Filipović and I. Kopriva, *A comparison of dictionary based approaches to inpainting and denoising with an emphasis to independent component analysis dictionaries*, Inverse Problems and Imaging, 5 (2011), pp. 815 – 841.

[35] M. Filipović, I. Kopriva, and A. Cichocki, *Inpainting color images in learned dictionary*, in European Signal Processing Conference (EUSIPCO), Special session on tensor decompositions and source separation, 2012.

[36] S. Foucart, *A note on guaranteed sparse recovery via ℓ_1 minimization*, Appl. and Comput. Harm. Analysis, 29 (2010), pp. 97 – 103.

[37] S. Foucart, A. Pajor, H. Rauhut, and T. Ullrich, *The gelfand widths of ℓ_p-balls for* $0 < p \leq 1$, Journal of Complexity, 26 (2010), pp. 629 – 640.

[38] G. Gan, C. Ma, and J. Wu, *Data clustering: Theory, algorithms and applications*, ASA-SIAM, 1st ed., 2007.

[39] S. Gandy, B. Recht, and I. Yamada, *Tensor completion and low-n-rank tensor recovery via convex optimization*, Inverse Problems, 27 (2011).

[40] Y. Gao and G. Church, *Improving molecular cancer class discovery through sparse non-negative matrix factorization*, Bioinformatics, 21 (2005), pp. 3970 – 3975.

[41] Q. Geng, H. Wang, and J. Wright, *On the local correctness of ℓ_1-minimization for dictionary learning*, CoRR, abs/1101.5672 (2011).

[42] P. Georgiev, F. Theis, and A. Cichocki, *Sparse component analysis and blind source separation of underdetermined mixtures*, IEEE Trans. Neural Netw., 16 (2005), pp. 992 – 996.

[43] J. G. Gonzalez and G. R. Arce, *Statistically-efficient fitering in impulsive environments: Weighted myriad filters*, EURASIP J. Adv. Signal Process., (2002), pp. 4 – 20.

[44] R. Gribonval and K. Schnass, *Dictionary identification - Sparse matrix factorization via ℓ_1-minimization*, IEEE Trans. Information Theory, 56 (2010), pp. 3523 – 3539.

[45] O. Guleyruz, *Nonlinear approximation based image recovery using adaptive sparse reconstructions and iterated denoising: Part I - Theory*, IEEE Trans. Image Process., 15 (2006), pp. 539 – 554.

[46] ——, *Nonlinear approximation based image recovery using adaptive sparse reconstructions and iterated denoising: Part II - Adaptive algorithms*, IEEE Trans. Image Process., 15 (2006), pp. 555 – 571.

[47] T. Hastie, R. Tibshirani, and J. Friedman, *The elements of statistical learning: Data mining, inference and prediction*, Springer, 2nd ed., 2001.

[48] C. Henneges, P. Laskov, E. Darmawan, J. Backhaus, B. Kammerer, and A. Zell, *A factorization method for the classification of infrared spectra*, BMC Bioinformatics, 11 (2010), p. 561.

[49] P. O. Hoyer, *Non-negative matrix factorization with sparseness constraints*, The Journal of Machine Learning Research, 5 (2004), pp. 1457 – 1469.

[50] T. M. Huang and V. Kecman, *Gene extraction for cancer diagnosis using suport vector machines*, Artificial intelligence in medicine, 35 (2005), pp. 185 – 194.

[51] D. H. Hubel and T. N. Wiesel, *Receptive fields and functional architecture of monkey striate cortex*, The Journal of Physiology, (1968), pp. 215 – 243.

[52] A. HYVÄRINEN, *New approximations of differential entropy for independent component analysis and projection pursuit*, in NIPS, 1997.

[53] A. HYVARINEN, *Fast and robust fixed-point algorithms for independent component analysis*, IEEE Trans. on Neural Netw., 10 (1999), pp. 626 – 634.

[54] A. HYVÄRINEN, R. CRISTESCU, AND E. OJA, *A fast algorithm for estimating over-complete ICA bases for image windows*, in Proc. of the International Joint Conference on Neural Networks, 1999.

[55] A. HYVÄRINEN, J. KARHUNEN, AND E. OJA, *Independent component anaylsis*, John Wiley & Sons, Inc., 1st ed., 2001.

[56] A. HYVARINEN AND E. OJA, *Independent component analysis by general nonlinear hebbian-like learning rules*, Signal Processing, 64 (1998), pp. 301 – 313.

[57] V. KECMAN, *Learning and soft computing - Support Vector Machines, Neural Networks and Fuzzy Logic Models*, The MIT Press, 2001.

[58] H. KIM AND H. PARK, *Sparse non-negative matrix factorizations via alternating non-negativity-constrained least squares for microarray data analysis*, Bioinformatics, 23 (2007), pp. 1495 – 1502.

[59] Z. KOLDOVSKY, P. TICHAVSKY, AND E. OJA, *Efficient variant of algorithm FastICA for independent component analysis attaining the Cramer-Rao lower bound*, IEEE Trans. Neural Netw., 17 (2006), pp. 1265 – 1277.

[60] I. KOPRIVA AND M. FILIPOVIĆ, *A mixture model with a reference-based automatic selection of components for disease classification from protein and/or gene expression levels*, BMC Bioinformatics, 12 (2011).

[61] L. D. LATHAUWER, *Signal processing based on multilinear algebra*, PhD thesis, Faculty of Engineering, K. U. Leuven, 1997.

[62] E. G. LEARNED-MILLER AND J. W. FISHER III, *ICA using spacings estimates of entropy*, The Journal of Machine Learning Research, 4 (2003), pp. 1271 – 1295.

[63] S. I. LEE AND S. BATZOGLOU, *Application of independent component analysis to microarrays*, Genome Biology, 4 (2003).

[64] M. S. LEWICKI AND B. A. OLSHAUSEN, *Probabilistic framework for the adaptation and comparison of image codes*, Journal of the Optical Society of America A, 16 (1999), pp. 1587 – 1601.

[65] M. S. LEWICKI AND T. J. SEJNOWSKI, *Learning overcomplete representations*, Neural computation, 12 (2000), pp. 337 – 365.

[66] L. Li, D. M. Umbach, P. Terry, and J. A. Taylor, *Application of the GA/KNN method to SELDI proteomics data*, Bioinformatics, 20 (2004), pp. 1638 – 1640.

[67] Y. Li, S. i. Amari, A. Cichocki, D. W. C. Ho, and S. Xie, *Underdetermined blind source separation based on sparse representation*, IEEE Trans. Signal Process., 54 (2006), pp. 423 – 437.

[68] W. Liebermeister, *Linear modes of gene expression determined by independent component analysis*, Bioinformatics, 18 (2002), pp. 51 – 60.

[69] J. Liu, P. Musialski, P. Wonka, and J. Ye, *Tensor completion for estimating missing values in visual data*, in IEEE International Conference on Computer ision, 2009, pp. 2114 – 2121.

[70] C.-T. Lu and T.-C. Chou, *Denoising of salt-and -pepper noise corrupted image using modified directional-weighted-median filter*, Pattern Rec. Lett., (2012).

[71] D. Lutter, P. Ugocsai, M. Grandl, E. Orso, F. Theis, E. W. Lang, and G. Schmitz, *Analyzing m-csf dependent monocyte/macrophage differentiation: Expression modes and meta-modes derived from an independent component analysis*, BMC Bioinformatics, 9 (2008), p. 100.

[72] J. Mairal, F. Bach, J. Ponce, and G. Sapiro, *Online learning for matrix factorization and sparse coding*, The Journal of Machine Learning Research, 11 (2010), pp. 9 – 60.

[73] J. Mairal, G. Sapiro, and M. Elad, *Sparse representation for color image restoration*, IEEE Trans. Image Process., 17 (2008), pp. 53 – 69.

[74] A. Maleki and D. L. Donoho, *Optimally tuned iterative reconstruction algorithms for compressed sensing*, IEEE J. Sel. Top. Signal Process., 4 (2010), pp. 330 – 341.

[75] S. Mallat, *A wavelet tour of signal processing*, Academic Press, 3rd ed., 2008.

[76] P. McCullagh, *Tensor methods in statistics*, Chapman and Hall, 1st ed., 1987.

[77] J. M. Mendel, *Tutorial on higher-order statistics (spectra) in signal processing and system theory: Theoretical results and some applications*, Proc. of the IEEE, 79 (1991), pp. 278 – 305.

[78] P. Milanfar, *A tour of modern image filtering*, IEEE Signal Process. Mag., (2012).

[79] G. H. Mohimani, M. Babaie-Zadeh, and C. Jutten, *A fast approach for overcomplete sparse decomposition based on smoothed ℓ_0 norm*, IEEE Trans. Signal Process., 57 (2009).

[80] N. Mourad and J. P. Reilly, *Minimizing nonconvex functions for sparse vector reconstruction*, IEEE Trans. Signal Process., 58 (2010), pp. 3485 – 3496.

[81] D. Needell and J. A. Tropp, *CoSaMP: Iterative signal recovery from incomplete and inaccurate samples*, Appl. and Comput. Harmon. Analysis, 26 (2009), pp. 301 – 321.

[82] E. Ollila, *The deflation-based FastICA estimator: Statistical analysis revisited*, IEEE Trans. Signal Process., 58 (2010), pp. 1527 – 1541.

[83] E. Ollila, H.-J. Kim, and V. Koivunen, *Compact Cramer-Rao bound expression for independent component analysis*, IEEE Trans. Signal Process., 56 (2008), pp. 1421 – 1428.

[84] B. A. Olshausen and D. J. Field, *Emergence of simple-cell receptive field properties by learning a sparse code for natural images*, Nature, 381 (1996), pp. 607 – 609.

[85] E. F. P. D. K. Ornstein, C. P. Paweletz, A. M. Ardekani, P. S. Hackett, B. A. Hitt, A. Velassco, C. Trucco, L. Wiegand, K. Wood, C. B. Simone, P. J. Levine, W. M. Linehan, M. R. Emmert-Buck, S. M. Steinberg, E. C. Kohn, and L. A. Liotta, *Serum proteomic patterns for detection of prostate cancer*, Journal of the National Cancer Institute, 94 (2002), pp. 1576 – 1578.

[86] E. F. Petricoin, A. M. Ardekani, B. A. Hitt, P. J. Levine, V. A. F. andS. M. Steinberg, G. B. Mills, C. Simone, D. A. Fishman, E. C. Kohn, and L. A. Liotta, *Use of proteomic patterns in serum to identify ovarian cancer*, The Lancet, 359 (2002), pp. 572 – 577.

[87] D. T. Pham, *Blind separation of instantaneous mixture of sources via an independent component analysis*, IEEE Trans. Signal Process., 44 (1996), pp. 2768 – 2779.

[88] ——, *Blind separation of instantaneous mixture of sources based on order statistics*, IEEE Trans. Signal Process., 48 (2000), pp. 363 – 375.

[89] P. Qiu, Z. J. Wang, and R. K. J. Liu, *Ensemble dependence model for classification and prediction of cancer and normal gene expression data*, Bioinformatics, 21 (2005), pp. 3114 – 3121.

[90] M. Rehn and F. T. Sommer, *A network that uses few active neurons to code visual input predicts the diverse shapes of cortical receptive fields*, J. Comput. Neurosci., (2007), pp. 135 – 146.

[91] V. G. Reju, S. N. Koh, and I. Y. Soon, *An algorithm for mixing matrix estimation in instantaneous blind source separation*, Signal Processing, 89 (2009), pp. 1762 – 1773.

[92] S. Roth and M. J. Black, *Fields of experts*, Int. Journal of Computer Vision, 82 (2009), pp. 205 – 229.

[93] R. Schachtner, D. Lutter, P. Knollmueller, A. M. Tomé, F. J. Theis, G. Schmitz, M. Stetter, P. G. Vilda, and E. W. Lang, *Knowledge-based gene expression classification via matrix factorization*, Bioinformatics, 24 (2008), pp. 1688 – 1697.

[94] M. W. Seeger and D. P. Wipf, *Variational Bayesian inference techniques*, IEEE Signal Process. Mag., 81 (2010).

[95] X. Shi and P. S. Yu, *Limitations of matrix completion via trace norm minimization*, ACM SIGKD Explorations, 12 (2011), pp. 16 – 20.

[96] K. Stadtlthanner, F. J. Theis, A. M. Tomé, C. G. Puntonet, and J. M. Górriz, *Hybridizing sparse component analysis with genetic algorithms for microarray analysis*, Neurocomputing, 71 (2008), pp. 2356 – 2376.

[97] B. L. Sturm, *A study on sparse vector distributions and recovery from compressed sensing*, CoRR, abs/1103.6246 (2011).

[98] B. L. Sturm and M. G. Christensen, *Cyclic pure greedy algorithms for recovering compressively sampled sparse signals*, in Asilomar Conf. on Signals, Systems and Computers, 2011.

[99] H. Takeda, S. Farsiu, and P. Milanfar, *Kernel regression for image processing and reconstruction*, IEEE Trans. Image Process., 16 (2007), pp. 349 – 366.

[100] P. Tichavsky, Z. Koldovsky, and E. Oja, *Performance analysis of the FastICA algorithm and Cramer-Rao bounds for linear independent component analysis*, IEEE Trans. Signal Process., 54 (2006), pp. 1189 – 1203.

[101] R. Tomioka, K. Hayashi, and H. Kashima, *Estimation of low-rank tensors via convex optimization*. arXiv.

[102] I. Tošić and P. Frossard, *Dictionary learning: What is the right representation for my signal?*, IEEE Signal Process. Mag., 28 (2011), pp. 27 – 38.

[103] J. A. Tropp, *Greed is good: Algorithmic results for sparse approximation*, IEEE Trans. Inf. Theory, 50 (2004), pp. 2231 – 2242.

[104] J. A. Tropp and A. C. Gilbert, *Signal recovery from random measurements via Orthogonal matching pursuit*, IEEE Trans. Inf. Theory, 53 (2007), pp. 4655 – 4666.

[105] J. A. Tropp and S. J. Wright, *Computational methods for sparse solution of linear inverse problems*, Proc. of the IEEE, 98 (2010), pp. 948 – 958.

[106] R. VIDAL, *Subspace clustering*, IEEE Signal Process. Mag., 28 (2011), pp. 52 – 68.

[107] Z. WANG AND A. BOVIK, *Mean squared error: love it or leave it? A new look at signal fidelity measures*, IEEE Signal Process. Mag., 13 (2004), pp. 600 – 612.

[108] Z. WANG, A. BOVIK, H. R. SHEIKH, AND E. P. SIMONCELLI, *Image quality assessment: from error visibility to structural similarity*, IEEE Trans. Image Process., 26 (2009), pp. 98 – 117.

[109] Q. XU, S. S. MOHAMED, M. M. A. SALAMA, AND M. KARNEL, *Mass spectrometry-based proteomic pattern analysis for prostete cancer detection using neural networks with statistical significance test-based feature selection*, in Proceedings of the IEEE Conference on Science and Technology for Humanity (TIC-STH), 2009, pp. 837 – 842.

[110] P. YANG, Z. ZHANG, B. B. ZHOU, AND A. Y. ZORNAYA, *A clustering based hybrid system for biomarker selection and sample classification of mass spectrometry*, Neurocomputing, 73 (2010), pp. 2317 – 2331.

Curriculum vitae

Marko Filipović was born on October 8th, 1982, in Osijek, Croatia. In 2001 he started his undergraduate studies at the Department of Mathematics, University of Zagreb. He received the BSc degree in mathematics (with major in applied mathematics) from the Department of Mathematics, University of Zagreb, in 2007. After graduation, he started to work as an assistant on the scientific project Multispectral Data Analysis, funded by the Ministry of Science, Education and Sports, Republic of Croatia, under the leadership of dr. sc. Ivica Kopriva, in the Division of Laser and Atomic Research and Development, Ruđer Bošković Institute, Zagreb. In the same year, 2007, he started his graduate studies at the Department of Mathematics, University of Zagreb, under the supervision of dr. sc. Ivica Kopriva and formal co-supervision from prof. dr. sc. Zlatko Drmač from the Department of Mathematics, University of Zagreb.

List of publications

Journals

1. Ivica Kopriva and Marko Filipović. *A mixture model with a reference-based automatic selection of components for disease classification form protein and/or gene expression levels*. BMC Bioinformatics, 12 (2011), 496.
2. Marko Filipović and Ivica Kopriva. *A comparison of dictionary based approaches to inpainting and denoising with an emphasis to independent component analysis learned dictionaries*. Inverse Problems and Imaging, vol. 5 (2011), nr. 4, pp. 815-841.

Conferences

3. Marko Filipović, Ivica Kopriva and Andrzej Cichocki. *Inpainting color images in learned dictionary*. Proceedings of the 20th European Signal Processing Conference (EUSIPCO 2012), August 27-31, 2012, Bucharest, Romania.

Appendix A

Approximation of entropy by cumulants

A.1 Properties of cumulants

Some of the properties of cumulants are listed in the following proposition [77, 61].

Proposition A.1. *Let $x_1, \ldots, x_M$ be random variables.*

1. $cum(a_1x_1, \ldots, a_Mx_M) = \left(\prod_{i=1}^{M} a_i\right) cum(x_1, \ldots, x_M)$.
2. $cum(x_1 + y_1, \ldots, x_M) = cum(x_1, \ldots, x_M) + cum(y_1, \ldots, x_M)$*, for random variable* y_1.
3. $cum(x_1, \ldots, x_M) = cum(x_{p_1}, \ldots, x_{p_M})$*, where* $(p_1, \ldots, p_M)$ *is any permutation of* $\{1, \ldots, M\}$.
4. *If a random variable y has an even p.d.f., the cumulants of y of odd order vanish.*
5. $cum(a + x_1, \ldots, x_M) = cum(x_1, \ldots, x_M)$*, for any* $a \in \mathbb{R}$.
6. *If a proper subset of variables* $x_1, \ldots, x_M$ *is independent of the others, then* $cum(x_1, \ldots, x_M) = 0$.
7. *If random variables* $x_1, \ldots, x_M$ *are mutually independent with* $y_1, \ldots, y_M$*, then* $cum(x_1 + y_1, \ldots, x_M + y_M) = cum(x_1, \ldots, x_M) + cum(y_1, \ldots, y_M)$.
8. *If y is a Gaussian random variable with the same mean and variance as a given random variable x, then for* $k \geq 3$ *it holds* $c_x^{(k)} = m_x^{(k)} - m_y^{(k)}$*. Therefore, higher-order* ($k > 2$) *cumulants of a Gaussian variable are zero.*

A.2 Expansions of probability density functions

The expansion (2.2.5) (and the related, and often more useful, *Edgeworth expansion*) can be, informally, derived as follows [76]. A *moment generating function* of a random *vector* **x** with p.d.f. f, $M_{\mathbf{x}}:\mathbb{R}^N\to\mathbb{R}$, is defined as

$$M_{\mathbf{x}}(\mathbf{t})=\mathbb{E}\left(\exp\left(\mathbf{t}^T\mathbf{x}\right)\right)=\int_{\mathbb{R}^N}\exp\left(\mathbf{t}^T\mathbf{x}\right)f(\mathbf{x})\,d\mathbf{x},$$

whenever the expectation exists. Moments can be obtained from the coefficients in the Taylor expansion of $M_{\mathbf{x}}$. It is possible that the moment generating function does not exist, in which case the characteristic function, which is always well defined, can be used. Cumulant generating function is then defined as $C_{\mathbf{x}}(\mathbf{t})=\log M_{\mathbf{x}}(\mathbf{t})$. To derive the expansion of p.d.f. f, an initial approximating p.d.f. f_0 is chosen. Let us denote the cumulants of f as κ_i, $\kappa_{i,j}$, $\kappa_{i,j,k}$ and so on, and cumulants of f_0 as λ_i, $\lambda_{i,j}, \lambda_{i,j,k}$ and so on. Let

$$C_{\mathbf{x}}(\zeta)=\sum_i\kappa_i\zeta_i+\sum_{i,j}\kappa_{i,j}\frac{\zeta_i\zeta_j}{2!}+\sum_{i,j,k}\kappa_{i,j,k}\frac{\zeta_i\zeta_j\zeta_k}{3!}+\cdots$$

and

$$C_0(\zeta)=\sum_i\lambda_i\zeta_i+\sum_{i,j}\lambda_{i,j}\frac{\zeta_i\zeta_j}{2!}+\sum_{i,j,k}\lambda_{i,j,k}\frac{\zeta_i\zeta_j\zeta_k}{3!}+\cdots$$

be Taylor expansions of cumulant generating functions of f and f_0, respectively. By *formal* subtraction and exponentiation we get

$$\begin{aligned}M_{\mathbf{x}}(\zeta) &= M_0(\zeta)\Big(1+\textstyle\sum_i\eta_i\zeta_i+\sum_{i,j}\eta_{ij}\frac{\zeta_i\zeta_j}{2!}+\\ &\quad+\textstyle\sum_{i,j,k}\eta_{ijk}\frac{\zeta_i\zeta_j\zeta_k}{3!}+\cdots\Big)\end{aligned}\tag{A.2.1}$$

where we have denoted $\eta_i=\kappa_i-\lambda_i$, $\eta_{i,j}=\kappa_{i,j}-\lambda_{i,j}$ and so on, and

$$\begin{aligned}\eta_{ij} &= \sum_{i,j}\left(\eta_{i,j}+\eta_i\eta_j\right),\\ \eta_{ijk} &= \sum_{i,j,k}\left(\eta_{i,j,k}+3\eta_i\eta_{j,k}+\eta_i\eta_j\eta_k\right),\\ \eta_{ijkl} &= \sum_{i,j,k,l}\left(\eta_{i,j,k,l}+4\eta_i\eta_{j,k,l}+3\eta_{i,j}\eta_{k,l}+6\eta_i\eta_j\eta_{k,l}+\eta_i\eta_j\eta_k\eta_l\right)\end{aligned}$$

and so on. Here, $\eta_i,\eta_{i,j},\eta_{i,j,k},\ldots$ are formal cumulants, while $\eta_{ij},\eta_{ijk},\ldots$ are formal moments. (A.2.1) can be formally inverted term by term. Namely, $M_0(\zeta)$ transforms to $f_0(\mathbf{x})$, while the second term, $\zeta_i M_0(\zeta)$, transforms to $f_i(\mathbf{x})=-\partial f_0(\mathbf{x})/\partial x_i$, see [76] for details. The other terms transform similarly, and we get

$$f_{\mathbf{x}}(\mathbf{x})=f_0(\mathbf{x})+\sum_i\eta_i f_i(\mathbf{x})+\sum_{i,j}\frac{f_{ij}(\mathbf{x})}{2}+\sum_{i,j,k}\frac{f_{ijk}(\mathbf{x})}{3!}+\ldots.\tag{A.2.2}$$

(A.2.2) can be expressed as

$$\begin{aligned}f_{\mathbf{x}}(\mathbf{x}) &= f_0(\mathbf{x})\Big\{1+\textstyle\sum_i\eta_i h_i(\mathbf{x})+\sum_{i,j}\eta_{ij}\frac{h_{ij}(\mathbf{x})}{2}+\\ &\quad+\textstyle\sum_{i,j,k}\eta_{ijk}\frac{h_{ijk}(\mathbf{x})}{3!}+\ldots\Big\}\end{aligned},$$

where

$$\begin{aligned} h_i(\mathbf{x}) &= \frac{f_i(\mathbf{x})}{f_0(\mathbf{x})}, \\ h_{ij}(\mathbf{x}) &= \frac{f_{ij}(\mathbf{x})}{f_0(\mathbf{x})} \end{aligned}$$

and so on. If f_0 is chosen as the density of the multivariate normal distribution, i.e.

$$f_0(\mathbf{x}) = \frac{1}{(2\pi)^{\frac{N}{2}}\sqrt{\det\mathbf{M}}} \exp\left(-\frac{1}{2}(\mathbf{x}-\mathbf{m})^T \mathbf{M}^{-1}(\mathbf{x}-\mathbf{m})\right),$$

where we have denoted by $\mathbf{m}$ the mean vector and by $\mathbf{M}$ the covariance matrix of f_0, the above functions h_i and h_{ij} are Hermite tensors, which in the univariate case $N = 1$ reduce to Hermite polynomials (see [76] for details). We can take $\lambda_i = \kappa_i$ and $\lambda_{ij} = \kappa_{ij}$ and we obtain the expression (2.2.5) (or, more precisely, its equivalent for the multivariate case). However, if $\mathbf{x}$ is taken as a standardized sum of n independent variables, the terms in this expansion are not monotonically decreasing in n. The re-grouped expansion, formed by collecting together the terms that are of equal order in n, is called Edgeworth expansion. However, the above derivation of this series was only formal, without taking into account conditions for the convergence of series. See [76] for details about conditions under which the Edgeworth expansion is valid.

Appendix B

Statistical properties of the FastICA estimator

B.1 Non-robustness

The following results are taken from [82]. Here we assume that, in the basic ICA model (2.1.3), the sources $\mathbf{s}$ satisfy $\mathbb{E}(s_i) = 0$ and $\mathbb{E}(s_i^2) = 1$ for all i. We also assume that at most one source has a Gaussian distribution, so that the sources $\mathbf{s}$ can be uniquely determined using the discriminating ICA contrast. The following theorem then holds [82].

Theorem B.1. *Assume that the moments* $c_{g,j} = cov\left(g(s_j), s_j\right) = \mathbb{E}\left(g(s_j)s_j\right)$ *and* $\rho_{g,j} = \mathbb{E}\left(g'(s_j)\right)$, *where cov denotes the cross-covariance of two random variables, defined as* $cov(x,y) = \mathbb{E}(xy) - \mathbb{E}(x)\mathbb{E}(y)$, *satisfy*

If $k < M$, $c_{g,j}, \rho_{g,j}$ *exist and* $c_{g,j} \neq \rho_{g,j}$ *for* $j = 1, \dots, k$.
If $k = M$, $c_{g,j}, \rho_{g,j}$ *exist and* $c_{g,j} \neq \rho_{g,j}$ *for* $j = 1, \dots, M-1$.

Then the influence function of the FastICA estimator $\mathbf{w}_{G,k}$, $k \in \{1, \dots, M\}$, *at the distribution* $F_\mathbf{x}$ *of the mixture* $\mathbf{x}$, *can be expressed as*

$$\begin{aligned} IF_{\mathbf{w}_{G,k},F_\mathbf{x}}(\mathbf{z}) &= u_k\left(\bar{s}_k\right)\sum_{j=k+1}^{M} \bar{s}_j \mathbf{w}_j \\ &\quad -\bar{s}_k \sum_{j=1}^{k-1}\left(u_j\left(\bar{s}_j\right) + \bar{s}_j\right)\mathbf{w}_j - \frac{(\bar{s}_k^2 - 1)}{2}\mathbf{w}_k \end{aligned}, \tag{B.1.1}$$

where

$$u_j(\bar{s}_j) = \frac{g(\bar{s}_j) - \mathbb{E}\left(g\left(s_j\right)\right) - c_{g,j}\bar{s}_j}{c_{g,j} - \rho_{g,j}}$$

and $\bar{s}_j = \mathbf{w}_j^T\left(\mathbf{z} - \nu\right)$, $\nu = \mathbb{E}_{F_\mathbf{x}}(\mathbf{x})$, *is the projection of the centered contamination point* $\mathbf{z}$ *in the direction* $\mathbf{w}_j$.

It can be seen from the expression (B.1.1) that the weights next to demixing vectors $\mathbf{w}_j$ are unbounded functions of $\mathbf{z}$, which shows that the estimators $\mathbf{w}_{G,k}$ are generally not robust, regardless of the choice of g.

B.2 Asymptotic variance and non-efficiency

We recall some definitions first.

Definition B.1. Let x be a random variable with p.d.f. $f_\theta(\cdot)$, where $\theta \in \Theta$ is a parameter vector. The Fisher information matrix $\mathscr{F}(\theta)$ of θ is defined as

$$\mathscr{F}(\theta) = \operatorname{cov}_{f_\theta}\left[\nabla_\theta \log f_\theta(x)\right],$$

where $\operatorname{cov}(\cdot)$ denotes the covariance matrix.

Under some regularity conditions (see, for example, [33]), the inverse of the Fisher information matrix is a lower bound on the variance of any unbiased estimator of θ. More precisely, if we denote the covariance matrix of an unbiased estimator $\hat{\theta}$ of θ as $\operatorname{cov}\left(\hat{\theta}\right)$, we have

$$\operatorname{cov}\left(\hat{\theta}\right) \geq \mathscr{F}(\theta)^{-1}, \tag{B.2.1}$$

where the notation $\mathbf{B} \geq \mathbf{C}$ for matrices means that $\mathbf{B} - \mathbf{C}$ is positive semi-definite. This lower bound is referred to as the Cramer-Rao (CR) lower bound. More precisely, let $x_1, \ldots, x_T$ be a random sample from $f_\theta(\cdot)$. The Fisher information matrix is computed from the likelihood $\prod_{i=1}^{T} f_\theta(x_i)$ as

$$\mathscr{F}_n(\theta) = \sum_{i=1}^{T} \operatorname{cov}_\theta\left[\nabla_\theta \log f_\theta(x_i)\right] = TF(\theta).$$

In this case, the inequality (B.2.1) is written as

$$\operatorname{cov}\left(\hat{\theta}\right) \geq \frac{1}{T}\mathscr{F}(\theta)^{-1}.$$

In the case of the (linear) ICA model (2.1.3), the unknown parameter vector that we wish to estimate is the inverse of the mixing matrix, $\mathbf{W} = \mathbf{A}^{-1}$. By denoting $\kappa_i = \operatorname{var}(\phi_i(s_i))$, where $\phi_i(s) = -f_i'(s)/f_i(s)$ is the *score function* of the i-th independent component, and $\lambda_i = \operatorname{var}(\phi_i(s_i)s_i)$ (assuming the existence and finitness of these variances), the inverse of the Fisher information matrix for $\mathbf{W}$ can be expressed as [83]

$$\left(\mathscr{F}(\theta)^{-1}\right)[i,\ j] = \begin{cases} \frac{1}{\lambda_i}\mathbf{w}_i\mathbf{w}_i^T + \sum_{l=1, l\neq i}^{M} \frac{\kappa_l}{\kappa_i\kappa_l - 1}\mathbf{w}_l\mathbf{w}_l^T & i = j \\ -\frac{1}{\kappa_i\kappa_j - 1}\mathbf{w}_j\mathbf{w}_i^T & i \neq j \end{cases}, \tag{B.2.2}$$

where we have denoted $\theta = \operatorname{vec}\left(\mathbf{W}^T\right)$, and the notation $[i,\ j]$ denotes the (i, j)-block of size $M \times M$ of the $M^2 \times M^2$ matrix $\mathscr{F}(\theta)^{-1}$. In ICA, the performance of the separation is often investigated via the matrix $\hat{\mathbf{G}} = \hat{\mathbf{W}}\mathbf{A}$, where $\hat{\mathbf{W}}$ is the estimate of $\mathbf{W} = \mathbf{A}^{-1}$. Using the above result (B.2.2), the lower bounds on the elements of $\hat{\mathbf{G}} = \left(\hat{g}_{ij}\right)$ are as follows (again, see [83])

$$\begin{array}{rcl} \delta_i & = & \sum_{j=1, j\neq i}^{N} \operatorname{var}\left(\hat{g}_{ij}\right) \geq \frac{1}{T}\sum_{j=1, j\neq i}^{N} \frac{\kappa_j}{\kappa_i\kappa_j - 1}, \\ \operatorname{var}\left(\hat{g}_{ii}\right) & \geq & \frac{1}{T}\frac{1}{\lambda_i}. \end{array}$$

Note that we have used the notation N for the number of sources; however, we have supposed that $M = N$, so that the above notation for the size of $\mathscr{F}(\theta)^{-1}$ agrees with this one. Using the expression (B.1.1) for the influence function of the FastICA estimator, the asymptotic covariance matrix of the FastICA estimator can be obtained as [82]

$$\mathrm{ASV}\left(\hat{\mathbf{w}}_{G,k}, F_{\mathbf{x}}\right) = \mathbb{E}\left[IF_{\mathbf{w}_{G,k},F_{\mathbf{x}}}(\mathbf{z}) \cdot IF_{\mathbf{w}_{G,k},F_{\mathbf{x}}}(\mathbf{z})^T\right].$$

Therefore, the following theorem holds [82].

Theorem B.2. *Under the assumptions of Theorem B.1, and the additional assumptions:*

1. *kurtosis* $\gamma_k = \mathbb{E}\left(s_k^4\right) - 3$ *of the k-th source exists;*
2. $\begin{cases} \sigma_{g,j}^2 \text{ exists } \forall j = 1,\dots,k & k < M \\ \sigma_{g,j}^2 \text{ exists } \forall j = 1,\dots,M-1 & k = M \end{cases},$

where $\sigma_{g,j}^2 = var\left(g\left(s_j\right)\right)$ *denotes the variance of* $g\left(s_j\right)$*, the asymptotic covariance matrix of the FastICA estimator* $\hat{\mathbf{w}}_{G,k}$*,* $k \in \{1,\dots,M\}$ *is*

$$\begin{aligned} ASV\left(\hat{\mathbf{w}}_{G,k}, F_{\mathbf{x}}\right) &= \sum_{j=1}^{k-1}\left(\alpha_{g,j}+1\right)\mathbf{w}_j\mathbf{w}_j^T + \beta_{g,k}\mathbf{w}_k\mathbf{w}_k^T \\ &\quad + \alpha_{g,k}\sum_{j=k+1}^{M}\mathbf{w}_j\mathbf{w}_j^T, \end{aligned} \tag{B.2.3}$$

where constants $\alpha_{g,j}$ *and* $\beta_{g,j}$ *are defined as*

$$\begin{aligned} \alpha_{g,j} &= var\left(u_j\left(s_j\right)\right) = \mathbb{E}\left[u_j\left(s_j\right)^2\right] = \frac{\sigma_{g,j}^2 - c_{g,j}^2}{\left(c_{g,j} - \rho_{g,j}\right)^2}, \\ \beta_{g,j} &= \frac{1}{4} var\left(s_j^2\right) = \frac{1}{4}\left(\gamma_j + 2\right), \end{aligned}$$

and $c_{g,j}$ *and* $\rho_{g,j}$ *are as before.*

It can be noted that the accuracy of k-th demixing vector $\hat{\mathbf{w}}_{G,k}$ depends on the distribution of sources extracted previously. Also, the existence of kurtosis is required for all sources to ensure the existence of the covariance matrix. Depending on the chosen non-linearity, the existence of even higher order moments might be required. Also, the case $c_{g,j} \approx \rho_{g,j}$ for any $j \in \{1,\dots,k\}$ leads to very high asymptotic variances of the estimator $\hat{\mathbf{w}}_{G,k}$.