Uniwersytet Warszawski
Wydział Matematyki, Informatyki i Mechaniki

Piotr Zawiła-Niedźwiecki
Nr albumu: 360589

Analiza składowych głównych jako optymalizacja na rozmaitościach gładkich

Praca magisterska
na kierunku MATEMATYKA
w zakresie MATEMATYKI STOSOWANEJ

Praca wykonana pod kierunkiem
dr. hab. Błażej Miasojedow

Warszawa, Wrzesień 2020

Streszczenie

Praca zawiera przedstawienie zmodyfikowanego algorytmu Manifold Sparse Functional PCA. Wyprowadzony algorytm znajduje rzadkie reprezentacje macierzy kowariancji empirycznej, korzystając z optymalizacji na rozmaitościach gładkich. Algorytm ten prezentuje odmienne podejście do problemu rekonstrukcji macierzy, niż te analizowane we współczesnej literaturze.

Słowa kluczowe

Optymalizacja na rozmaitościach; Rozmaitość Stiefela; Rozmaitość Grassmanna; Proximal Gradient Method; Złożoność Obliczeniowa; Sparse PCA; Manifold Alternating Direction Method of Multipliers;

Dziedzina pracy (kody wg programu Socrates-Erasmus)

11.1 Matematyka

Klasyfikacja tematyczna

49Qxx Manifolds and measure-geometric topics
58Axx General theory of differentiable manifolds
62Axx Foundational topics in statistics
65Dxx Numerical approximation and computational geometry
65Kxx Numerical methods for mathematical programming, optimization and variational techniques
49M27 Decomposition methods
49M29 Numerical methods involving duality

Tytuł pracy w języku angielskim

Principal component analysis as optimization on Riemanian manifolds

Spis treści

Wprowadzenie 5

1. Optymalizacja na rozmaitościach 7
1.1. Proximal operator 7
1.2. Proximal operator dla spenalizowanego problemu 9
1.3. Alternating direction method of multipliers 11
1.4. Rozmaitości Stiefela i Grassmanna 12

2. Sformułowanie algorytmu 15
2.1. PCA 15
2.2. Twierdzenie Eckarta-Younga-Mirsky'ego 16
2.3. Wyprowadzenie postaci analitycznej algorytmu 16

3. Wyniki symulacji 21

4. Podsumowanie i wnioski 25

A. Geometria różniczkowa 27

Bibliografia 31

Wprowadzenie

Analiza składowych głównych (Principal Component Analysis, w pracy skrótowo zapisywane jako PCA) jako sposób wykrywania kierunków największej wariancji w danych i redukcji wymiarowości, została wymyślona w 1901 roku przez Karla Pearsona. Na przestrzeni lat znalazła zastosowania w wielu dziedzinach pod różnymi nazwami: *proper orthogonal decomposition* w inżynierii mechanicznej, *eigenvalue decomposition* w algebrze liniowej, czy też chociażby jako *empirical orthogonal functions* w meteorologii. PCA, jak i wiele pozostałych osiągnieć Pearsona, jest wykładana już na etapie studiów licencjackich w ramach wykładów ze statystyki. Należy ona także do fundamentalnych metod w wachlarzu zasobów analityków danych. Istotną wadą PCA jest brak intepretowalności, gdyż każda składowa główna jest liniową kombinacją wejściowych wartości własnych. Z tego powodu w 2006 roku Hui Zou, Trevor Hastie i Robert Tibshirani [8] zaproponowali przeformułowanie problemu, nazywane Sparse PCA. Sparse PCA startuje od problemu macierzowego i po relaksacji dochodzi do problemu regresyjnego spenalizowanego karą L_1. W efekcie Sparse PCA narzucało rzadkość na kierunkach głównych. W ostatnich latach pojawiło się wiele podejść skupiających się na wymuszeniu rzadkości kierunków głównych i wartości szczególnych, m.in. w pracach [5] z 2019 roku, czy [9] z 2015 roku. Przekrojowa analiza proponowanych na przestrzeni lat podejść do problemu jest dostępna w [18].

PCA sprowadza się do znalezienia rozkładu wartości własnych symetrycznej macierzy kowariancji. Rozkłady macierzy symetrycznych można także przedstawić jako problem optymalizacyjny z narzuconymi dodatkowymi ograniczeniami na składniki rozkładu. Ograniczenia te mogą być ograniczeniami w przestrzeni euklidesowej [1], [8], [9], [18], lub też ograniczeniami na rozmaitościach [3], [5], [7]. Problemy z ograniczeniami na rozmaitościach wymagają własnych algorytmów optymalizacyjnych. Nasze podejście będzie startowało od problemu optymalizacji na rozmaitości Stiefela, dla którego narzucamy dodatkowe kary na składniki rozkładu wymuszające rzadką reprezentację. Dogłębne wyprowadzenie tematyki optymalizacji nieeuklidesowej jest opisane w [10].

W rozdziale pierwszym opiszemy wprowadzenie do algorytmów optymalizacji w $\mathbb{R}^n$, a następnie ich uogólnienie na rozmaitości Stiefela. W tym samym rozdziale również przedstawimy rozmaitości Stiefela i Grassmanna oraz ich własności. Wszystkie definicje i twierdzenia z geometrii różniczkowej potrzebne do zrozumienia pojęć z pierwszego rozdziału, znajdują się w appendixie. W rozdziale drugim wyprowadzimy pojęcie PCA, twierdzenie Eckarta-Younga-Mirsky'ego, a także sformułujemy algorytm wraz z analitycznymi wzorami użytymi w jego implementacji. Wyniki symulacji zostaną opisane w rozdziale trzecim. Pracę zakończymy podsumowaniem, oraz propozycją przyszłych ulepszeń i modyfikacji algorytmu.

Rozdział 1

Optymalizacja na rozmaitościach

Nasze podejście do PCA będzie szczególnym przypadkiem ogólnego problemu postaci:

$$\min_{x\in\mathbb{M}} f(x) + g(x), \tag{1.1}$$

gdzie f jest funkcją wypukłą i różniczkowalną, której gradient spełnia warunek Lipschitza. Funkcja g jest wypukła i półciągła z dołu, ale niekoniecznie różniczkowalna, a $\mathbb{M}$ to rozmaitość gładka zanurzona w $\mathbb{R}^n$. Problemy tej postaci niekoniecznie są różniczkowalne, więc do ich rozwiązania będziemy potrzebowali sięgnąć po uogólnienie standardowych metod gradientowych.

Opis metod optymalizacji rozpoczniemy od prostszego przypadku, czyli dla $\mathbb{M} = \mathbb{R}^n$, a następnie uogólnimy na rozmaitości gładkie.

1.1. Proximal operator

Podstawowym narzędziem stosowanym w optymalizacji problemów postaci (1.1) jest proximal operator [10] rozdział 6.

Definicja 1 *Proximal operator* $\text{prox}_g : \mathbb{R}^n \to \mathbb{R}^n$ *funkcji g jest zdefiniowany jako*

$$\text{prox}_g(v) = \operatorname*{argmin}_x \big(g(x) + \frac{1}{2}\|x - v\|_2^2\big). \tag{1.2}$$

Jezeli g jest wypukła i wlaściwa, to minimalizowana funkcja jest ściśle wypukła, w związku z czym posiada jednoznacznie wyznaczone minimum. W pracy będziemy zazwyczaj rozważali problem postaci λg, gdzie $\lambda > 0$. Wtedy operator przyjmuje postać

$$\text{prox}_{\lambda g}(v) = \operatorname*{argmin}_x \big(g(x) + \frac{1}{2\lambda}\|x - v\|_2^2\big).$$

Podamy teraz kilka własności proximal operatora:

Uwaga 1.1 *Dla szczególnego przypadku, gdzie funkcja g jest indykatorem zbioru wypukłego* C, prox_g *jest rzutem, to znaczy*

$$\begin{aligned}
\text{prox}_g(x) &= \operatorname*{argmin}_y \begin{cases} \frac{1}{2}\|x - y\|_2^2 & \text{if } y \in C \\ +\infty & \text{if } y \notin C \end{cases} \\
&= \operatorname*{argmin}_{y\in C} \frac{1}{2}\|x - y\|_2^2 \\
&= \operatorname*{argmin}_{y\in C} \|x - y\|_2
\end{aligned} \tag{1.3}$$

Wprowadźmy teraz pojęcie subróżniczki,

Definicja 2 *Wektor $\xi \in \mathbb{R}^n$ nazywamy subgradientem funkcji g w punkcie $x_0 \in \mathbb{M}$, jeśli*

$$g(x) \geq g(x_0) + \langle \xi^T, x - x_0 \rangle, \quad x \in \mathbb{M}.$$

Zbiór wszystkich subgradientów g w punkcie x_0 nazywamy subróżniczką i oznaczamy $\partial g(x_0)$

Uwaga 2.1 *Proximal operator funkcji g, dla każdej pary $(x, p) \in \mathbb{R}^n \times \mathbb{R}^n$, spełnia następującą równoważność*

$$p = \text{prox}_g(x) \iff x - p \in \partial g(p).$$

Jeżeli funkcja g jest różniczkowalna, to powyższa równoważność upraszcza się do poniższej postaci

$$p = \text{prox}_g(x) \iff x - p = \nabla g(p).$$

Twierdzenie 2.1 *Punkt x^* minimalizuje g wtedy i tylko wtedy, gdy*

$$x^* = \text{prox}_g(x^*), \tag{1.4}$$

czyli x^* jest punktem stałym prox_g.

Dowód Znajduje się w [11] na stronie 131. □

Twierdzenie 2.2 *Niech $g : \mathbb{M} \to (-\infty, \infty]$ będzie wypukłą funkcją rzeczywistą. Zachodzi wtedy równoważność*

$$x^* \in \underset{x}{\text{argmin}}\ g(x) \iff 0 \in \partial g(x^*). \tag{1.5}$$

Dowód Wprost z definicji subróżniczki otrzymujemy, że $0 \in \partial g(x^*)$ jest spełnione wtedy i tylko wtedy gdy

$$g(x) \geq g(x^*) + \langle 0, x - x^* \rangle \quad \forall x \in \mathbb{M},$$

co jest równoważne (1.5). □

Stwierdzenie 2.1 *Proximal operator jest ściśle nie-rozszerzającym operatorem, czyli $\forall x, y \in \mathbb{R}^n$:*

$$\left\| \text{prox}_g(x) - \text{prox}_g(y) \right\|_2^2 \leq (x - y)^T \left(\text{prox}_g(x) - \text{prox}_g(y) \right).$$

Ściśle nie-rozszerzające operatory są specjalnym przypadkiem nie-rozszerzających operatorów (czyli tych ze stałą Lipschitza równą 1). Iterowanie nie-rozszerzających operatorów nie musi zbiegać do punktów stałych (dla przykładu $f(x) = -x$, lub obroty). Jednak dla nie-rozszerzającego operatora N, operator zdefiniowany jako $T = (1-\alpha)I + \alpha N$, gdzie $\alpha \in (0, 1)$, posiada te same punkty stałe co N. W efekcie iterowanie T zbiegnie do punktu stałego T (a co za tym idzie także N), czyli sekwencja

$$x^{k+1} := (1 - \alpha)x^k + \alpha N(x^k),$$

zbiegnie do punktu stałego N.

Operatory ogólnej postaci $(1 - \alpha)I + \alpha N$, gdzie N jest nie-rozszerzającym operatorem i $\alpha \in (0, 1)$, nazywane są α-uśrednianymi operatorami. W szczególności ściśle nie-rozszerzające

operatory są $\frac{1}{2}$-uśrednianymi operatorami. Uśredniane operatory spełniają własność opisaną przez Twierdzenie Krasnoselskiego-Manna, to znaczy iterowanie tychże operatorów zbiegnie do punktu stałego, jeżeli takowy istnieje. W efekcie dla uśrednianego T, które posiada punkt stały, zdefiniujmy iterację jako

$$x^{k+1} := T(x^k),$$

dla jakiegoś x^0 (ustalonego z góry, lub wylosowanego). Wtedy dla $\|T(x^k) - x^k\| \to 0$, *gdy* $k \to \infty$, x^k zbiega do punktu stałego T. Dowody i wyprowadzenia zbieżności dla uśrednianych operatorów można znaleźć w [12] rozdziale 5.2.

Mając powyższe podwaliny teoretyczne, możemy zdefiniować proximal point algorithm.

Definicja 3 *Dla funkcji wypukłej* $g : \mathbb{R}^n \to \mathbb{R} \cup \infty$*, jej proximal point algorithm to*

$$x^{k+1} := \text{prox}_{\lambda g}(x^k),$$

gdzie k to licznik iteracji, a x^k *oznacza k-tą iterację algorytmu.*

Powyższa konstrukcja jest konstrukcją czysto teoretyczną, praktyczne przykłady zostaną opisane w następnej sekcji.

1.2. Proximal operator dla spenalizowanego problemu

Definicja 4 *Dla problemu postaci*

$$\underset{x}{\text{argmin}} \big(f(x) + g(x)\big), \tag{1.6}$$

gdzie f jest funkcją wypukłą i gładką, a g jest wypukła, algorytm opisany w definicji 3, przyjmuje postać

$$\mathrm{x}^{\mathrm{k}+1} = \text{prox}_{\lambda_\mathrm{k},\mathrm{g}}\left(\mathrm{x}^{\mathrm{k}} - \lambda^{\mathrm{k}}\nabla \mathrm{f}\left(\mathrm{x}^{\mathrm{k}}\right)\right). \tag{1.7}$$

Algorytm ten nazywamy Proximal Gradient Method.

Uwaga 4.1 *Dla każdego kroku optymalizacyjnego Proximal Gradient Method* (1.7)*,* x^{k+1} *jest opisane wzorem*

$$\mathrm{x}^{\mathrm{k}+1} \in \mathrm{x}^{\mathrm{k}} - \lambda^{\mathrm{k}}\nabla \mathrm{f}\left(\mathrm{x}^{\mathrm{k}}\right) - \lambda^{\mathrm{k}}\partial \mathrm{g}\left(\mathrm{x}^{\mathrm{k}+1}\right). \tag{1.8}$$

Z uwagi 4.1 wynika, że punkt stały (1.8) jest rozwiązaniem problemu (1.6). Dla szczególnych funkcji $g(x)$, algorytm (1.7) przyjmuje postać dobrze znanych algorytmów gradientowych. Pokażmy teraz parę takowych przykładów:

Stwierdzenie 4.1 *Jeżeli* $g(x)$ *jest funkcją stale równą zero, to Proximal Gradient Method jest uogólnieniem standardowego algorytmu gradientowego*

$$x^{k+1} = x^k - \lambda_k \nabla f(x^k).$$

Stwierdzenie 4.2 *Jeżeli* $g(x)$ *jest indykatorem zbioru wypukłego* C *jak w uwadze 1.1, to Proximal Gradient Method jest uogólnieniem rzutowanego algorytmu gradientowego*

$$x^{k+1} = \underset{y \in C}{\text{argmin}} \|y - x^k + \lambda_k \nabla f(x^k)\|_2$$

Jeżeli ∇f jest Lipschitzowsko ciągła ze stałą L, można wykazać, że proximal gradient method (1.7), dla kroków gradientowych $\lambda^k = \lambda \in (0, \frac{1}{L})$, jest zbieżny w tempie $O(\frac{1}{k})$, jeżeli ustalimy stały krok gradientowy $\lambda^k = \lambda \in (0, \frac{1}{L})$ [13]. W praktyce, typowo, stała Lipschitza jest nieznana, wówczas można użyć procedury line-search do znalezienia wielkości kroku. Taka modyfikacja nie zmienia tempa zbieżności algorytmu [13].

Żeby przybliżyć algorytm proximal gradient opiszemy go w szczególnym przypadku problemu regresji liniowej z karą LASSO (model ten jest opisany w [18], w rozdziale 2). Dla przypomnienia ogólne sformułowanie problemu Lasso jest postaci

$$\min_{w \in \mathbb{R}^n} \frac{1}{n} \sum_{i=1}^{n} (y_i - \langle w, x_i \rangle)^2 + \|w\|_1. \tag{1.9}$$

Gdzie x_i to i-ta kolumna macierzy obserwacji, y_i to i-ta wartość zmiennej objaśnianej, a w to wektor wag dla modelu liniowego.

Stwierdzenie 4.3 *Dla funkcji $g(x) = \|x\|_1$ proximal operator ma postać*

$$(prox_{\lambda g}(x))_i = \begin{cases} x_i - \lambda, \ dla \ x_i > \lambda \\ 0, \ dla \ |x_i| \leq \lambda \\ x_i + \lambda, \ dla \ x_i < -\lambda. \end{cases} \tag{1.10}$$

Dowód. Zapiszmy wpierw wzór (1.9) w konwencji (1.1), gdzie składniki $\frac{1}{n} \sum_{i=1}^{n} (y_i - \langle w, x_i \rangle))^2$ oraz $\|w\|_1$, reprezentują odpowiednio funkcje $f(x), \ g(x)$. Z racji, że funkcja g nie jest ściśle różniczkowalna, chcąc wyliczyć jej $prox_g(x)$ skorzystamy z uwagi 2.1 i znajdziemy subróżniczkę operatora $prox_g(x)$:

$$\begin{aligned} y = prox_g(x) &\iff 0 \in \partial(g(y) + \frac{1}{2}\|y - x\|_2^2) \\ &\iff 0 \in \partial g(y) + y - x \\ &\iff x - y \in \partial g(y). \end{aligned}$$

Dla naszego szczególnego przypadku $g(x) = \|w\|_1$ subróżniczka $\partial g(w)$ jest łatwa do wyliczenia, i-ty elementy subróżniczki to dokładnie:

$$\partial |w_1| = \begin{cases} 1, dla \ w_i > 0, \\ -1, dla \ w_i < 0, \\ [-1, 1], dla \ w_i = 0. \end{cases}$$

Podstawiając teraz do wzoru na $\text{prox}_g(x)$ z pierwszego równania w dowodzie, dla $g(w) = \|w\|_1$ i $\lambda > 0$, $prox_{\lambda g}(x)$ jest zdefiniowana względem każdego elementu jako

$$(prox_{\lambda g}(x))_i = \begin{cases} x_i - \lambda, \ dla \ x_i > \lambda, \\ 0, \ dla \ |x_i| \leq \lambda, \\ x_i + \lambda, \ dla \ x_i < -\lambda, \end{cases}$$

gdzie podstawiamy $w_i = x_i - \lambda$, a x_i oraz λ są ustalonymi wartościami. $\square$

Powyższa funkcja jest znana jako soft thresholding operator $S_\gamma(x) = prox_{\gamma\|\cdot\|_1}(x)$. W dalszej części pracy będziemy się bezpośrednio odnosić do pojęcia soft-thresholding operator dla przypadku problemów Lasso. Definicja tego operatora zaprezentowana jest poniżej.

Definicja 5 *Soft thresholding operator elementu $x \in \mathbb{R}$ to:*

$$S_\gamma(x) = \text{prox}_{\lambda\|\cdot\|_1}(x) = \text{sign}(x)(|x| - \lambda)_+.$$

Gdzie operatory $\text{sign}(\cdot)$ *oraz* $(\cdot)_+ = max(0, \cdot)$ *są aplikowane po współrzędnych.*

Dogłębna analiza *Proximal Gradient Method* jest dostępna w [10] w rozdziale 10.

1.3. Alternating direction method of multipliers

Powyżej rozważany proximal operator zakładał, że obie funkcje z (1.1) mają te same dziedziny. Alternating direction method of multipliers (ADMM), jest algorytmem należącym do klasy algorytmów Augmented Lagrangian methods [10] rozdział 14, [14]. Dzięki dualności rozwiązanie problemu (1.1) jest równoważne rozwiązaniu

$$\min_{x,z:x=z} f(x) + g(z), \tag{1.11}$$

gdzie $x \in \mathbb{R}^n$, $z \in \mathbb{R}^m$. Metoda mnożników Lagrange'a z narzuconymi ograniczeniami zastosowana do równania (1.11), daje problem minimalizacyjny opisany równaniem:

$$\underset{x,z}{\text{argmin}}\, L_\rho(x, z, y) = f(x) + g(z) + y^T(x - z) + \frac{\rho}{2}\|x - z\|_2^2, \tag{1.12}$$

gdzie zmienna y jest zmienną dualną powiązaną z rozbieżnością pomiędzy zmiennymi x i z. Możemy na y patrzeć jako na zmienną co trzyma informację o tym, jak bardzo odległe od siebie są x i z.

Definicja 6 *Algorytm ADMM dla problemu* (1.12)*, jest algorytmem iteracyjnym, który powtarza poniższe kroki do zbieżności:*

$$\begin{aligned} x^{k+1} &:= \underset{x}{\text{argmin}}\; L_\rho\left(x, z^k, y^k\right) \\ z^{k+1} &:= \underset{z}{\text{argmin}}\; L_\rho\left(x^{k+1}, z, y^k\right) \\ y^{k+1} &:= y^k + \rho\left(x^{k+1} - z^{k+1}\right), \end{aligned} \tag{1.13}$$

gdzie $\rho > 0$.

Dla algorytmu (1.13) zmienne iterowane x^k oraz z^k z czasem zbiegają do siebie i do rozwiązania optymalnego, jednak obie te zmienne mogą mieć inne własności. Dla przykładu, jeżeli $g = \|\cdot\|_1$, to zmienne z^k będą rzadkie (bo wtedy krok na z^{k+1} upraszcza się do soft-thresholdingu), zaś zmienne x^k będą bliskie z^k (czyli prawie rzadkie).

Pokażmy teraz jak przekształcić (1.13) do algorytmu opisanego za pomocą proximal operatorów. Zauważmy wpierw, że (1.13) można zapisać jako

$$\begin{aligned} x^{k+1} &:= \underset{x}{\text{argmin}} \left(f(x) + y^{kT}x + \frac{\rho}{2}\left\|x - z^k\right\|_2^2 \right) \\ z^{k+1} &:= \underset{z}{\text{argmin}} \left(g(z) - y^{kT}z + \frac{\rho}{2}\left\|x^{k+1} - z\right\|_2^2 \right) \\ y^{k+1} &:= y^k + \rho\left(x^{k+1} - z^{k+1}\right), \end{aligned}$$

a wciągając potem czynniki liniowe pod normę otrzymujemy

$$\begin{aligned} x^{k+1} &:= \operatorname*{argmin}_x \left(f(x) + \frac{\rho}{2} \left\| x - z^k + \frac{1}{\rho} y^k \right\|_2^2 \right) \\ z^{k+1} &:= \operatorname*{argmin}_z \left(g(z) + \frac{\rho}{2} \left\| x^{k+1} - z - \frac{1}{\rho} y^k \right\|_2^2 \right) \\ y^{k+1} &:= y^k + \rho \left(x^{k+1} - z^{k+1} \right). \end{aligned}$$

Podstawiając teraz $u^k = \frac{1}{\rho} y^k$ i $\lambda = \frac{1}{\rho}$, możemy zapisać ADMM w formie złożonej z proximal operatorów

$$\begin{aligned} x^{k+1} &:= \operatorname{prox}_{\lambda f} \left(z^k - u^k \right) \\ z^{k+1} &:= \operatorname{prox}_{\lambda g} \left(x^{k+1} + u^k \right) \\ u^{k+1} &:= u^k + x^{k+1} - z^{k+1}. \end{aligned}$$

Zaletą ADMM jest to, że funkcję optymalizowaną można rozbić na mniejsze podproblemy, i narzucić własności na zmienne. Będzie to szczególnie przydatne w dalszej części pracy, gdzie będziemy potrzebowali narzucić, by zmienna należała do rozmaitości Stiefela i jednocześnie była rzadka. Dodatkowo jeżeli wyliczenie proximal operatora dla jednej z funkcji jest prostsze niż ich sum (na przykład posiadają analityczne wzory jak *soft-thresholding*), to ADMM ułatwia rozwiązywanie takich problemów.

1.4. Rozmaitości Stiefela i Grassmanna

Wprowadźmy teraz pojęcia rozmaitości Stiefela i Grassmanna, oraz ich własności, które będą później potrzebne do opisania algorytmów optymalizacyjnych na nich. Pojęcia z geometrii różniczkowej potrzebne do zrozumienia poniższego rozdziału są spisane w appendixie A.

Definicja 7 *Rozmaitością Stiefela, nazywamy Riemannowską zanurzoną rozmaitość macierzy ortonormalnych, to znaczy*

$$St(n,p) = \{X \in \mathbb{R}^{n \times p} : X^T X = I_p\}.$$

Definicja 8 *Rozmaitością Grassmanniana nazywamy rozmaitość*

$$Gr(n,p) = \{\mathcal{U} = col(X) : X \in R_*^{n \times p}\},$$

gdzie $R_*^{n \times p}$ jest zbiorem wszystkich macierzy pełnego rzędu w $\mathbb{R}^{n \times p}$, a $col(X)$ opisuje przestrzeń rozpinaną przez kolumny macierzy X.

Obie powyższe rozmaitości można zdefiniować też jako przestrzenie ilorazowe.

Stwierdzenie 8.1 *Oznaczmy zbiór kwadratowych macierzy ortogonalnych jako*

$$O_n = \{X \in \mathbb{R}^{n \times n} : X^T X = I_m\}.$$

Rozmaitość Stiefela to przestrzeń ilorazowa zdefiniowana jako

$$St(n,p) = O_n / O_{n-p}.$$

Rozmaitość Grassmanna zdefiniujemy jako iloraz obu powyższych rozmaitości, czyli

$$Gr(n,p) = St(n,p) / O_p.$$

Punkt na rozmaitości Stiefela to ortonormalna macierz wymiaru $n \times p$. Dla rozmaitości Grassmanna jeden punkt to cała podprzestrzeń liniowa zdefiniowana przez wybraną ortogonalną bazę zapisaną w postaci macierzy $n \times p$. Warto zauważyć, że o ile wybór punktu na rozmaitości Stiefela jest unikalny, to dla rozmaitości Grassmanna wybór macierzy nie jest (chociażby permutacja wektorów bazowych będzie już inną macierzą, ale opisującą tą samą przestrzeń). Dla szczególnych przypadków, rozmaitość Stiefela jest sferą jednostkową ($p = 1$), lub całym zbiorem macierzy ortogonalnych ($p = n$).

Pokażemy teraz kilka własności rozmaitości Stiefela.

Stwierdzenie 8.2 *Przestrzeń styczna do rozmaitości Stiefela w punkcie X jest zdefiniowana jako*

$$T_X St(p,n) = \{Z \in \mathbb{R}^{n \times p} : X^T Z + Z^T X = 0\}.$$

By wykonać kroki gradientowe na rozmaitości Stiefela będziemy szukali przestrzeni stycznych wzdłuż których będziemy chcieli ten krok wykonać. Kroki te jednak będą zazwyczaj kończyć się poza naszą rozmaitością, więc będziemy potrzebowali wykonać retrakcję z powrotem na rozmaitość.

Stwierdzenie 8.3 *Dla punktu $X \in St(n,p)$, rzutowanie punktu $Z \in R^{n \times p}$ na przestrzeń styczną T_X jest opisane wzorem*

$$P_X(Z) = Z - X(X^T Z + Z^T X)/2.$$

Stwierdzenie 8.4 *Retrakcję na rozmaitość Stiefela punktu $X + \xi$ można wyliczyć poprzez rozkład QR macierzy $X + \xi$, to znaczy*

$$R_X(\xi) := qf(X + \xi).$$

Gdzie $qf(A)$ oznacza składnik Q z dekompozycji macierzy $A \in \mathbb{R}_*^{n \times p}$. Składnik Q rozkładu macierzy $A = QR$, należy do St(n,p), a macierz R jest górno trójkątna.

By lepiej zrozumieć do czego powyższe pojęcia są nam potrzebne, opiszmy algorytm gradientowy na rozmaitości Stiefela.

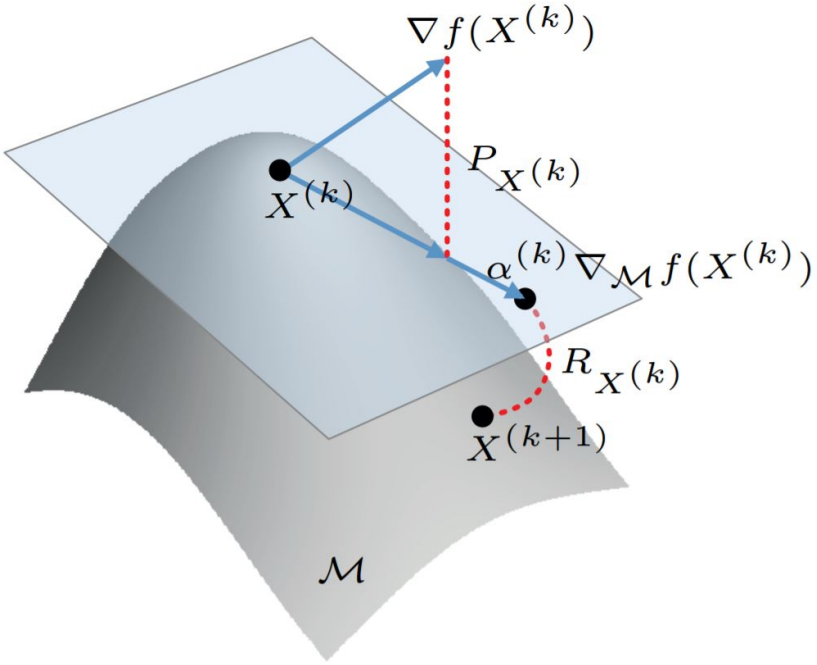
Algorithm 1: Algorytm gradientowy na rozmaitości Stiefela

- Dla funkcji f wypukłej i punktu startowego $X^0 \in St(n,p)$, algorytm gradientowy na rozmaitości Stiefela przyjmuje poniższą postać.
- Powtarzaj do zbiegnięcia:
  1. Wylicz gradient $\nabla f(X^k)$
  2. Zrzutuj gradient na przestrzeń styczną do punktu X^k

     $$\nabla_{St(n,p)} f(X^k) = P_{X^k}(\nabla f(X^k))$$

  3. Wylicz rozmiar kroku optymalizacyjnego α^k wzdłuż przestrzeni stycznej
  4. Wykonaj retrakcję z przestrzeni stycznej na rozmaitość Stiefela

     $$X^{k+1} = R_{X^k}(-\alpha^k \nabla_{St(n,p)} f(X^k))$$



Rysunek 1.1: Algorytm gradientowy na rozmaitości $\mathcal{M}$, rysunek zaczerpnięty z [3].

Powyższy algorytm w ogólniejszym przypadku dla rozmaitości $\mathcal{M}$, a nie $St(n,p)$, jest zobrazowany na rysunku 1.1. Dla konkretnego punktu z rozmaitości wyliczamy gradient funkcji, by potem zrzutować go na przestrzeń styczną do rozmaitości. Następnie wyznaczamy rozmiar kroku optymalizacyjnego. Wiele metod znajdujących ten krok jest opisanych w [2] w rozdziale 4. Znalazłwszy rozmiar kroku gradientowego na przestrzeni stycznej, potrzebujemy tylko wykonać rzutowanie z powrotem na rozmaitość.

Wprowadźmy jeszcze pojęcie odległości geodezyjnej pomiędzy dwoma punktami z rozmaitości Grassmanna, która przyda nam się w rozdziale 3.

Stwierdzenie 8.5 *Dla $\mathcal{X}, \mathcal{Y} \in Gr(n,p)$, oznaczmy ich bazy ortonormalne jako odpowiednio X, Y i niech obie macierze rozpinają tę samą przestrzeń. Wtedy X^TY jest odwracalne. Oznaczmy*

$$P_{X_\perp} Y \left(X^T Y\right)^{-1} = U \Sigma V, \tag{1.14}$$

jako rozkład SVD. Przyjmijmy $\Theta = arctan\Sigma$. Wtedy odległość geodezyjna na $Gr(n,p)$ pomiędzy $\mathcal{X}, \mathcal{Y}$ jest wyznaczona wzorem

$$\text{dist}(\mathcal{X}, \mathcal{Y}) = \sqrt{\theta_1^2 + \ldots + \theta_p^2}.$$

Odległość geodezyjna jest najkrótszą odległością pomiędzy dwoma punktami na rozmaitości. W celu dokładniejszego zrozumienia pojęć z Geometrii różniczkowej użytych w powyższym rozdziale odsyłamy czytelnika do [2], [6], [7], [15].

Rozdział 2

Sformułowanie algorytmu

Nim przejdziemy do opisania zaimplementowanego algorytmu i jego postaci analitycznej, wpierw wprowadzimy pojęcie PCA.

2.1. PCA

Główną ideą stojącą za PCA jest redukcja wymiarowości danych, zatrzymująca jednocześnie jak najwięcej informacji o wariancji w danych. Osiągane jest to poprzez kombinację liniową danych wejściowych. Poprzez tę transformację otrzymujemy tak zwane składowe główne, które są parami nieskorelowane. Pierwsze kilka składowych głównych zawiera większość informacji o wariancji zawartej w całych danych wejściowych. Zacznijmy od wyprowadzenia pojęcia rozkładu spektralnego:

Twierdzenie 8.1 *Dla symetrycznej macierzy Σ o wymiarze $p \times p$ istnieją:*

- *ortonormalna macierz kwadratowa V o wymiarze $p \times p$, oznaczmy $V = [v_1, ..., v_p]$,*
- *diagonalna macierz Λ o wyrazach na przekątnych $(\lambda_1, ..., \lambda_p)$, takich że $\Sigma v_i = \lambda_i v_i$.*

Czyli v_i to wektory własne macierzy Σ, a λ_i to jej wartości własne. Wtedy zachodzi:

$$\begin{aligned} \Sigma[v_1, ..., v_p] &= [\lambda_1 v_1, ..., \lambda_p v_p], \\ \Sigma V &= V\lambda, \\ \Sigma &= V\Lambda V^T. \end{aligned}$$

Gdzie rozkład $\Sigma = V\Lambda V^T$ nazywamy rozkładem spektralnym macierzy Σ, a macierze V i Λ nazywamy odpowiednio macierzami wektorów własnych i wartości własnych.

Niech $X \in \mathbb{R}^n$ będzie p-elementowym wektorem obserwacji z empiryczną macierzą kowariancji Σ. Dla rozkładu spektralnego $\Sigma = V\Lambda V^T$ zauważmy że:

$$V^T \Sigma V = V^T V \Lambda V^T V = \Lambda = \mathrm{diag}(\lambda_1, ..., \lambda_p,)$$

gdzie $\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_p \geq 0$ to wartości własne macierzy Σ. Jeżeli wszystkie wartości własne $\lambda_1, ..., \lambda_p$ są parami różne, to macierz V jest unikalna z dokładnością do znaków kolumn.

Definicja 9 *Dla danego wektora losowego $X \in \mathbb{R}^n$ oraz macierzy kowariancji $Var(x) = \Sigma = V\Lambda V^T$,*

- *składowymi głównymi nazywamy elementy wektora* $[Y_1, ..., Y_p]^T = Y = V^T X$,
- *kierunkami głównymi nazywamy kolumny macierzy* $V = [v_1, ..., v_p]$, *które są także wektorami głównymi* Σ.

Twierdzenie 9.1 *Składowe główne* $y_1, y_2, ..., y_p$ *są parami nieskorelowane oraz zachodzi* $Var(y_i) = \lambda_i$ *dla* $i = 1, ..., p$.

Dowód Zauważmy, że

$$Var(Y) = Var(V^T X) = V^T Var(x) V = V^T \Sigma V = \Lambda,$$

gdzie diagonalność macierzy Λ wykazuje jednocześnie brak korelacji pomiędzy składowymi głównymi jak i to, że $Var(y_i) = \lambda_i$ $\square$.

2.2. Twierdzenie Eckarta-Younga-Mirsky'ego

Idea algorytmu opisanego w następnej sekcji bazuje na twierdzeniu Eckara-Younga-Mirskiego.

Twierdzenie 9.2 *Niech* $X \in \mathbb{R}^{m \times n}$ *będzie rzeczywistą macierzą, a* $X = U\Sigma V^T$ *będzie jej rozkładem SVD. Wtedy problem minimalizacji*

$$\underset{r(M)=k}{\operatorname{argmin}} \|X - M\|_F,$$

ma jawne rozwiązanie postaci $M = U\Sigma_k V^T$. *Gdzie* U *oraz* V^T *to te same macierze wektorów własnych, a* Σ_k *to macierz wartości własnych* Σ *ograniczona do pierwszych* k *wartości własnych.*

Dowód Dowód opisany jest w pracy Mirsky'ego [17]. $\square$

Powyższe twierdzenie porusza temat rozkładów SVD macierzy. W naszym algorytmie będziemy rozważali macierze wejściowe będące symetrycznymi macierzami kwadratowymi. Dla takich macierzy rozkład spektralny i SVD są sobie równoważne. Ograniczenie się do k pierwszych wartości własnych pozwala ograniczyć rząd zrekonstruowanej macierzy wejściowej. W naszym podejściu rzadkość macierzy wartości własnych będziemy chcieli osiągnąć za pomocą kary L_1.

2.3. Wyprowadzenie postaci analitycznej algorytmu

Problem, który będziemy chcieli rozwiązać korzystając z algorytmów optymalizacyjnych z poprzedniego rozdziału, ma następującą postać:

$$\underset{\boldsymbol{V} \in St(p,p), \Lambda \in \mathbb{R}^p_+}{\operatorname{argmin}} \left\| X - V\Lambda V^\top \right\|_F^2 + \lambda_1 \|V\|_1 + \lambda_2 \|\Lambda\|_1. \tag{2.1}$$

Gdzie X to symetryczna macierz wejściowa, której rzadką rekonstrukcję chcemy uzyskać, a macierze V, Λ to odpowiednio wektory i wartości własne rozkładu spektralnego macierzy X. Powyższe sformułowanie problemu wynika z tego, że chcieliśmy osiągnąć rzadkie reprezentacje macierzy wektorów głównych oraz wartości głównych wejściowej macierzy. Metod znajdujących rzadkie reprezentacje jest wiele [5], [8], [9], [18], jednak w naszym podejściu

chcieliśmy osiągnąć także interpretowalność spenalizowanej macierzy. Penalizacja macierzy wartości głównych ma nam zapewnić niski rząd zrekonstruowanej macierzy, zaś penalizacja macierzy wektorów głównych ma nam zapewnić, że wektory główne będą rzadkie, co pozwoli zinterpretować zależności pomiędzy zmiennymi w danych. Rozważanie macierzy wektorów głównych jako macierzy z rozmaitości Stiefela wynika z tego, że jako macierz wektorów głównych ma rozpinać całą przestrzeń. Powyżej zaprezentowana rzadka rekonstrukcja powinna skutecznie rozdzielić podprzestrzenie własne, co zostanie zaprezentowane w rozdziale 3. Problem optymalizacyjny (2.1) będziemy chcieli rozwiązać korzystając z odmiany Manifold Sparse Functional PCA [5]:

Algorithm 2: Manifold SFPCA

1. Zainicjalizuj $\hat{V}$ i $\hat{\Lambda}$ jako odpowiednio wektory szczególne i wartości własne macierzy X.
2. Powtarzaj n razy:
   (a) Podproblem V:
   $$V^{k+1} = \underset{\boldsymbol{V} \in St(p,p)}{\operatorname{argmin}} \left\| X - V\Lambda^k V^\top \right\|_F^2 + \lambda_1 \|V\|_1$$
   (b) Podproblem Λ:
   $$\Lambda^{k+1} = \underset{\Lambda \in \mathbb{R}_+^p}{\operatorname{argmin}} \left\| X - V^{k+1}\Lambda V^{\top k+1} \right\|_F^2 + \lambda_2 \|\Lambda\|_1$$
3. Zwróć $\hat{V}$ i $\hat{\Lambda}$.

Powyższy algorytm jest przykładem algorytmu spadku po współrzędnych [19], [18] str.109-115, na przestrzeni $St(p,p) \times \mathbb{R}^p$, przy czym macierze diagonalne utożsamiamy z jej przekątną. Do rozwiązania każdego z podproblemów, skorzystamy z wcześniej opisanych algorytmów optymalizacyjnych.

Dla podproblemu V będziemy chcieli skorzystać z algorytmu opisanego w definicji 6. Podstawiając do ogólnego wzoru (1.11) ograniczenie $x = V \in St(p,p)$, oraz $z = W \in \mathbb{R}^{p \times p}$ otrzymujemy poniższą postać algorytmu.

Algorithm 3: MADMM dla podproblemu V

1. Zainicjalizuj $V^k = W^k = \hat{V}$ i $Z^k = 0, k = 1$.
2. Powtarzaj m razy:
   (a)
   $$V^{k+1} = \underset{\boldsymbol{V} \in St(p,p)}{\operatorname{argmin}} \left\| X - V\Lambda V^\top \right\|_F^2 + \frac{\rho}{2} \left\| V - W^k + Z^k \right\|_F^2$$
   (b)
   $$W^{k+1} = \underset{W \in \mathbb{R}^{p \times p}}{\operatorname{argmin}} \ \lambda_1 \|W\|_1 + \frac{\rho}{2} \left\| V^{k+1} - W + Z^k \right\|_F^2$$
   (c)
   $$Z^{k+1} = Z^k + V^{k+1} - W^{k+1}$$
3. Zwróć W.

Do rozwiązania podpunktu b) powyższego algorytmu użyjemy proximal gradient opisanego w definicji 4, jako że ten problem upraszcza się do regularyzacji Lasso i wcześniej wspomnianego w definicji 5 soft-thresholdingu. By wyliczyć jego dokładne rozwiązanie, wpierw przeformułujemy jego postać, by lepiej zobrazować optymalizację po każdym elemencie macierzy

$$W^{k+1} = \min_{W \in \mathbb{R}^{p \times p}} \sum_{i,j} \lambda_1 \|W\|_1 + \frac{\rho}{2} (V^{k+1} - W + Z^k)_{ij}^2.$$

Podstawiając $S = V^{k+1} + Z^k$, skorzystajmy z definicji 5 otrzymując względem każdego elementu macierzy rozwiązanie postaci

$$W_{ij} = sign(S_{ij}) \cdot max(0, S_{ij} - \frac{\lambda_1}{\rho})$$

Podproblem V nie ma rozwiązania analitycznego, będziemy korzystać z metody iteracyjnej alg 1, bo problem ten sprowadza się do minimalizacji funkcji na rozmaitości Stiefela.

Rozwiązanie podproblemu Λ z algorytmu 2 jest prostsze i podobnie jak podpunkt b) algorytmu 3, bazuje na soft-thresholdingu 5. Mamy więc wzór

$$\Lambda^{k+1} = \underset{\boldsymbol{V} \in \mathcal{V}_{\boldsymbol{p \times p}}^{\boldsymbol{Su}}}{\operatorname{argmin}} \left\| V^{kT} X V^k - \Lambda \right\|_F^2 + \lambda_2 \|\Lambda\|_1,$$

gdzie skorzystaliśmy, z tego, że macierz V jest ortonormalna, więc możemy wymnożyć oba elementy normy Frobeniusa z obu stron przez odpowiednio V^T i V. Podstawiając teraz $Z^k = V^{kT} X V^k$, otrzymujemy:

$$\underset{\boldsymbol{V} \in \mathcal{V}_{\boldsymbol{p \times p}}^{\boldsymbol{Su}}}{\operatorname{argmin}} \sum_j |Z_{jj}^k - \Lambda_{jj}|^2 + \lambda_2 |\Lambda_{jj}|,$$

co możemy zapisać jako problem względem każdego elementu

$$\min_x f(x) = (x - z)^2 + \lambda_2 \quad x \geq 0,$$

co po zróżniczkowaniu daje analityczne rozwiązanie postaci

$$x = max(0, z - \frac{\lambda_2}{2}).$$

Wszystkie powyższe wyprowadzenia sprowadzają nasz algorytm 2 do postaci bardziej analitycznej:

Algorithm 4: Manifold SFPCA

---

1. Zainicjalizuj $\hat{V}$ i $\hat{\Lambda}$ jako odpowiednio, wektory własne i wartości własne macierzy X.
2. Powtarzaj n razy:
   (a) Podproblem V:
      i. Zainicjalizuj $V^k = W^k = \hat{V}$ i $Z^k = 0, k = 1$.
      ii. Powtarzaj m razy:
         A. Wylicz za pomocą algorytmu 1

$$V^{k+1} = \underset{\boldsymbol{V} \in St(p,p)}{\operatorname{argmin}} \left\| X - V\Lambda V^\top \right\|_F^2 + \frac{\rho}{2} \left\| V - W + Z^k \right\|_F^2$$

         B.

$$W_{ij}^k = sign((V^{k+1} + Z^k)_{ij}) \cdot max(0, (V^{k+1} + Z^k)_{ij} - \frac{\lambda_1}{\rho})$$

         C.

$$Z^{k+1} = Z^k + V^{k+1} - W^{k+1}$$

      iii. Zwróć W
   (b) Podproblem Λ:

$$\Lambda_{ii}^{k+1} = max(0, (V^{kT} X V)_{ii} - \frac{\lambda_2}{2})$$

3. Zwróć $\hat{V}$, $\hat{\Lambda}$ oraz $\hat{V}\hat{\Lambda}\hat{V}^T$

Gdzie powyżej zwrócone macierze, to odpowiednio: macierz wektorów własnych, macierz wartości własnych, oraz rekonstrukcja macierzy wejściowej.

Rozdział 3

Wyniki symulacji

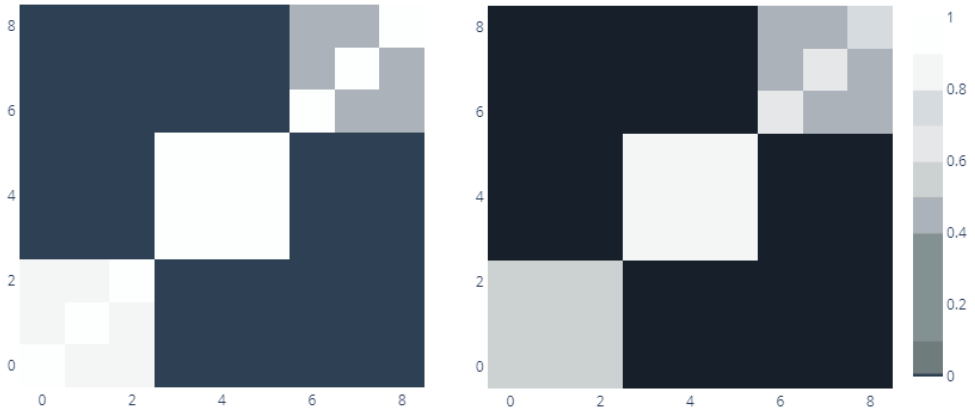
Finalna postać algorytmu zaprezentowana w poprzednim rozdziale, została zaimplementowana w języku programowania Python i jest ona dostępna na repozytorium GitHub pod linkiem `https://github.com/ZawilaP/Magisterka/tree/feature/develop2.0`. W ramach implementacji nie ustawiono kryteriów stopu, a jedynie ograniczenie iteracji algorytmu. Parametr ρ we wstępnej analizie był ustalony na wartość 1. W ramach eksperymentów wyszło, że algorytm jest szybko zbieżny i nieduża ilość iteracji znajduje sensowne rzadkie reprezentacje macierzy wejściowej (wystarczało 5 kroków głównych algorytmu i 3 kroki dla każdego wywołania MADMM). Parametry λ_1 oraz λ_2 miały znaczący wpływ na wyniki. Algorytm był bardzo czuły na małe zmiany w wartościach parametrów λ_i. Dla przykładu zmienienie wartości λ_1 z 0.075 na 0.08, potrafiło powodować diametralnie różne wyniki finalne. Niewielka różnica wartości mogła decydować o tym, czy zwracana rekonstrukcja jest macierzą zer, czy też macierzą o oczekiwanej przez nas rzadkiej postaci.

Zmienianie wartości parametru ρ o rzędy wielkości miało niebagatelny wpływ na postać rekonstrukcji. By zrozumieć czemu, spójrzmy na wyprowadzone wzory w algorytmie 3. Dla bardzo małych wartości ρ (rzędu 10^{-2}), w podpunktach a) i b), kary powiązane z podobieństwem macierzy V oraz W mniej wnoszą do wartości minimalizowanych funkcji. Podobnie dla przypadku gdy ρ przyjmuje bardzo duże wartości (rzędu 10^4 lub więcej), kara za rozbieżność macierzy ma większy wkład, co powoduje, że kara L_1 nie będzie narzucała rzadkości na rozwiązanie. W kontekście całego algorytmu oznacza to tyle, że za małe wartości ρ będą wymuszać, by macierz wektorów własnych była zerowa, zaś dla za dużych parametrów ρ wejściowa macierz wektorów własnych niemalże się nie zmieni.

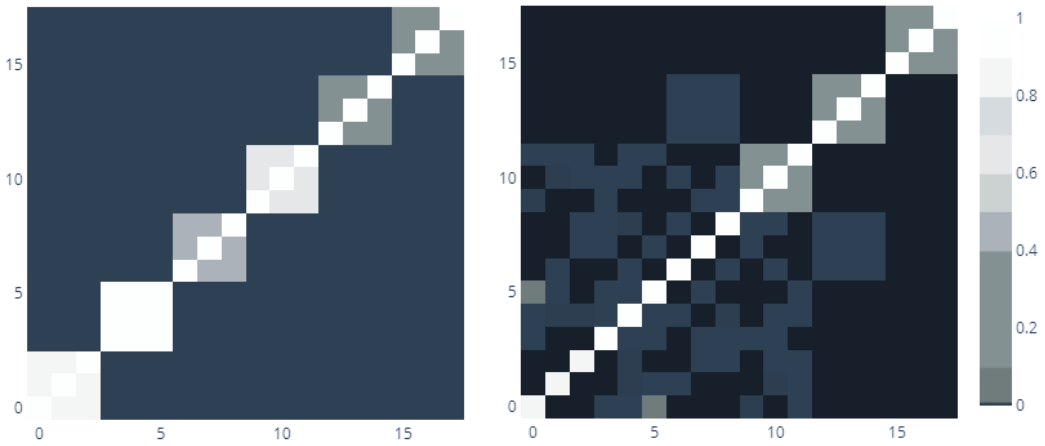
W ramach eksperymentów rozważano dwie miary jakości dopasowania modelu, normę Frobeniusa między macierzą wejściową i wyjściową, oraz odległość na rozmaitości Grassmanna pomiędzy macierzami wektorów głównych macierzy wejściowej i wyjściowej. Żadna z miar jednak nie pozwalała wybrać jawnie najlepszego dopasowania. Odległość na rozmaitości Grassmanna premiowała dopasowania nie posiadające rzadkiej reprezentacji wektorów głównych. Norma Frobeniusa premiowała w ogóle nie spenalizowane macierze i dla wyjątkowo małych parametrów λ_i, preferowała rekonstrukcje stworzone z tymi małymi parametrami.

Macierze wejściowe były postaci blokowej, gdzie wiodąca przekątna złożona z jedynek otoczona była blokami kwadratowymi, gdzie i-ty blok złożony był z wartości ϕ_i. Pozostałe pola macierzy były wypełnione wartościami φ, gdzie $\forall i \ \varphi \ll \phi_i$. Algorytm w idealnym przypadku powinien zauważyć, że każdy blok odpowiada pewnej podprzestrzeni własnej, w efekcie niemalże zerując wartości φ.

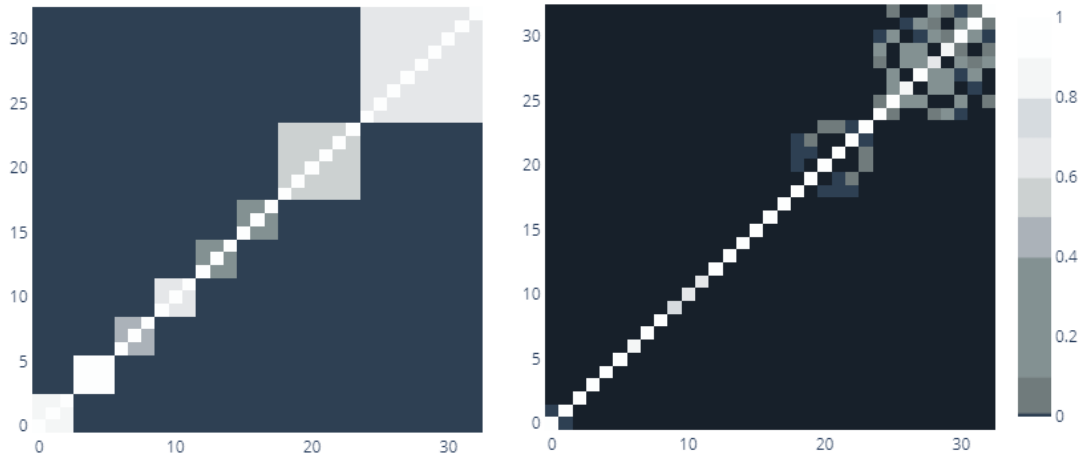
Rysunki 3.1, 3.2, 3.3, 3.4 pokazują reprezentację macierzy wejściowej i wyjściowej w postaci heatmap, gdzie każde pole macierzy jest reprezentowane przez kolory. Wartości bliskie



Rysunek 3.1: Heatmapy macierzy wejściowej 9×9 (lewo) i jej rekonstrukcji (prawo) dla parametrów $\lambda_1 = 0.025,\ \lambda_2 = 0.6,\ \rho = 1$, o normie Frobeniusa 0.43 i odległości na rozmaitości Grassmanna równej 0.89.



Rysunek 3.2: Heatmapy macierzy wejściowej 18×18 (lewo) i jej rekonstrukcji (prawo) dla parametrów $\lambda_1 = 0.015,\ \lambda_2 = 1.0,\ \rho = 1$, o normie Frobeniusa 1.15 i odległości na rozmaitości Grassmanna równej 0.08.



Rysunek 3.3: Heatmapy macierzy wejściowej 33×3 (lewo) i jej rekonstrukcji (prawo) dla parametrów $\lambda_1 = 0.075,\ \lambda_2 = 0.125,\ \rho = 1$, o normie Frobeniusa 5.97 i odległości na rozmaitości Grassmanna równej 1.77.

jedynki przyjmują kolor biały, a wartości bliskie 0, ale mniejsze conajmniej 10-krotnie od φ, przyjmują kolor czarny. Rekonstrukcja potrafi dobrze rozdzielić podprzestrzenie własne jak widać na rysunku 3.1, jednak penalizowane są wszystkie wartości macierzy. Przekątna przyjmuje wartości bliższe pozostałym elementom swojego bloku (które też są penalizowane), gdy wszystkie φ przyjmują wartości 0.

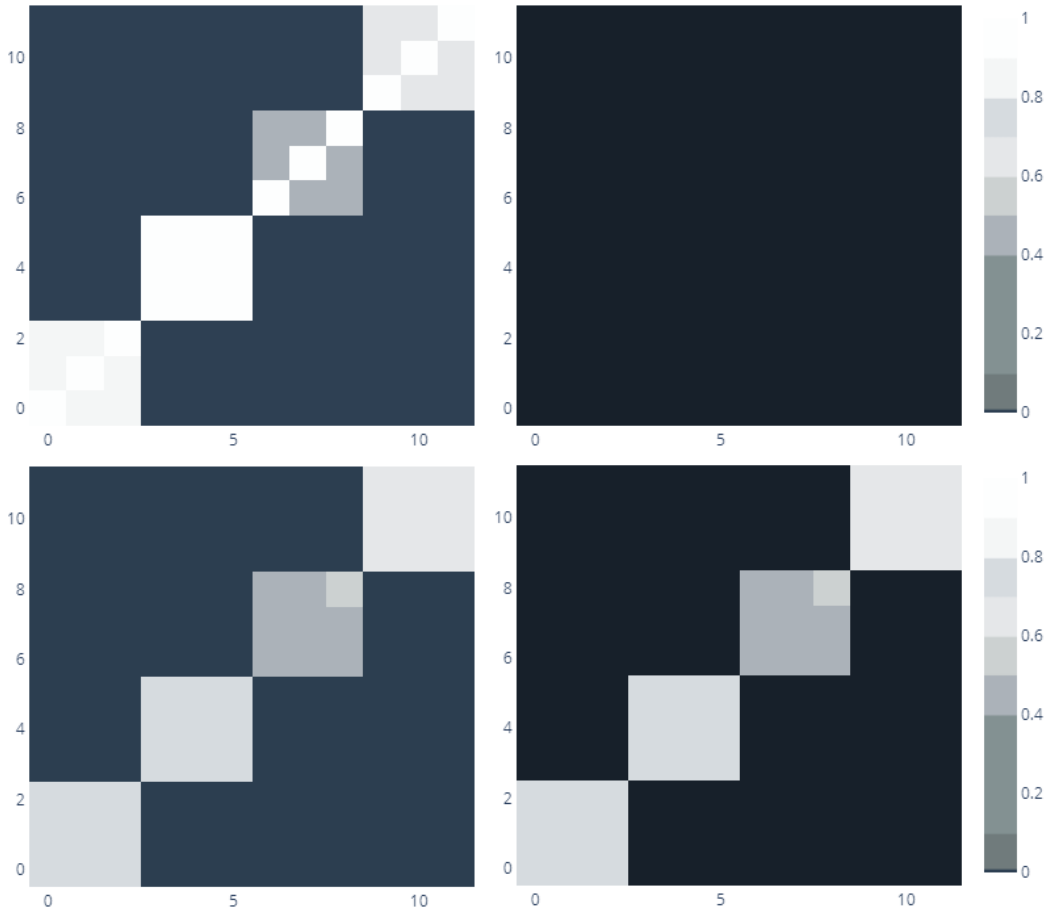
Jak widać na rysunku 3.3, algorytm dla przypadku, gdy podprzestrzenie własne nie są tego rozmiaru, bardziej penalizuje mniejsze bloki, skupiając się na rekonstrukcji większych bloków.

Rysunek 3.2 obrazuje przypadek, gdy algorytm pomimo wyjątkowo niskiej miary odległości na rozmaitości Grassmanna nie rekonstruuje dobrze macierzy. Warto zauważyć, że pomimo podobieństwa wektorów własnych, zrekonstruowana macierz, ma widoczne nie spenalizowane bloki kwadratowe.

Algorytm jeżeli nawet nie znajduje idealnej rekonstrukcji jak w przypadku rysunków 3.2, 3.3, to penalizuje macierze całymi blokami i w dodatku symetrycznie. Na rysunku 3.2 dobrze widać jak symetrycznie spenalizowane są bloki rozmiaru 3×3.

Rysunek 3.4 obrazuje jak różne rzędy wielkości zmiennej ρ potrafią wpłynąć na postać zrekonstruowanej macierzy. Prawa górna heatmapa jest przykładem sytuacji, gdy ρ jest tak małe, że kara L_1 dominuje algorytm, zwracając nam zerową macierz. Lewy dolny wykres obrazuje nam przypadek, gdy bardzo duża wartość ρ powoduje zwracanie niemalże niezmienionej macierzy wektorów własnych. Dla tej wartości penalizowane jedynie są wartości własne, co dobrze widać na wiodących blokach, których wartości zmalały, a wartości poza tymi blokami pozostały takie same. Prawa dolna część rysunku zaś przedstawia nam przypadek dobrze dobranej wartości ρ, zwracającej rzadkie macierze wektorów i wartości własnych, dla których podprzestrzenie własne zostały widocznie rozdzielone. Warto zauważyć także, że żadne z rozważanych kryteriów wyboru modelu, nie wybrałoby tu modelu z prawego dolnego rogu. Oba

kryteria faworyzują model, który nie penalizuje w ogóle wektorów własnych. Wybrany przez te kryteria model w efekcie nie miałby rzadkiej rekonstrukcji, która była naszym celem.



Rysunek 3.4: Heatmapy macierzy wejściowej 12×12 (góra po lewej) i jej rekonstrukcji (pozostałe), dla parametrów wspólnych $\lambda_1 = 0.009$, $\lambda_2 = 1.0$. Wykres na górze po prawej to rekonstrukcja o parametrze $\rho = 0.01$, normie Frobeniusa 4.99 i odległości na rozmaitości Grassmanna równej 5.44. Wykres na dole po lewej to rekonstrukcja o parametrze $\rho = 10000$, normie Frobeniusa 1.34 i odległości na rozmaitości Grassmanna równej $3.74 * 10^{-5}$. Wykres na dole po prawej to rekonstrukcja o parametrze $\rho = 1$, normie Frobeniusa 1.35 i odległości na rozmaitości Grassmanna równej $4.41 * 10^{-4}$.

Rozdział 4

Podsumowanie i wnioski

Wyniki naszej implementacji zmodyfikowanego algorytmu SFPCA są obiecujące. Potrafi osiągnąć zamierzony cel rekonstrukcji, jednak wymaga to bardzo dokładnego dobrania parametrów, by osiągnąć rzadką rekonstrukcję. Uważamy, że można tenże problem rozwiązać poprzez znalezienie lepszych kryteriów wyboru modelu. Rozważane przez nas miary dopasowania algorytmu nie wyznaczały najlepszego dopasowania, a ustalenie, które parametry są najlepsze, wymagało ręcznej analizy. Dobrze sformułowane kryterium, bazujące na zerowaniu wartości poza blokami, oraz premiujące brak penalizacji wartości otaczających przekątną macierzy, mogłoby osiągnąć dobre wyniki. Jednak wymagałoby ono obszernego testowania, by sprawdzić czy dobrze się uogólnia na różne przykłady z życia wzięte.

Kolejną rzeczą wymagającą implementacji są kryteria stopu dla całego algorytmu i dla MADMM. Sugerowalibyśmy standardową różnicę normy Frobeniusa pomiędzy krokami algorytmu, tzn. jeżeli $\|X_{k+1} - X_k\|_F^2 \leq \epsilon$, gdzie ϵ jest stałą rzędu 10^{-7}. Można rozważyć też powyższe kryterium z dodatkową stałą cierpliwości, w razie jeżeli optymalizacja utykałaby na jakichś płaszczyznach przestrzeni parametrów [16].

Warto byłoby też rozważyć w każdym kroku algorytmu progowanie wartości macierzy V oraz Λ. Jeżeli byłyby one odpowiednio małe (rzędu 10^{-5}), byłyby im przypisywane zera. To podejście stosuje się w niektórych bibiliotekach Pythonowych korzystających z penalizacji Lasso, a algorytm ten nazywa się *thresholded Lasso*. Efektywność takiego podejścia dla wielu problemów jest poparta teoretycznymi wynikami [20], [21]. Dzięki *thresholded Lasso* moglibyśmy znaleźć lepsze rzadkie reprezentacje wektorów własnych podczas każdej iteracji głównej części algorytmu.

W samej implementacji algorytmu możnaby zoptymalizować obliczenia używając własnych implementacji algorytmu gradientowe na rozmaitości Stiefela. Zastąpiłoby to algorytm 1 w wyliczaniu kolejnych wartości V^{k+1} wewnątrz algorytmu 3. Aktualna implementacja korzystająca z bibliotek TensorFlow i Pymanopt, jest wymagająca obliczeniowo, a jej uproszczenie mogłoby skrócić czas liczenia algorytmu kilkukrotnie.

Innym sposobem narzucenia rzadkości na macierz wartości własnych, byłoby użycie w trakcie znajdowania rozkładu SVD, rozkładu SVD rzędu k, gdzie $k < n$. Możnaby wtedy usunąć krok podproblemu Λ i wykonywać jedynie kolejne kroki na macierzy V w ramach całego algorytmu. W takim podejściu rekonstrukcja powinna odnaleźć jedynie k podprzestrzeni własnych i osiągnąć przez nas zamierzone wyniki.

Dodatek A

Geometria różniczkowa

Definicja 10 *Niech M będzie zbiorem. Mapą M nazywamy parę (U, ϕ), gdzie $U \subset M$, a przez ϕ oznaczamy bijekcję między U, a zbiorem otwartym z $\mathbb{R}^n$. U nazywamy dziedziną mapy, a przez n oznaczamy jej wymiar. Dla danego $p \in U$, elementy $\phi(p) = (x_1, ..., x_n)$ nazywamy koordynatami p mapy (U, ϕ).*

Definicja 11 *Dla dwóch map (U, ϕ) i (V, φ) zbioru M, wymiarów odpowiednio n i m mówimy, że są gładko spójne, jeśli zachodzi jeden z dwóch przypadków $U \cap V = \emptyset$, bądź $U \cap V \neq \emptyset$, dla którego ponadto*

1. *$\psi(U \cap V)$ to otwarty pozdbiór $\mathbb{R}^n$,*
2. *$\varphi(U \cap V)$ to otwarty podzbiór R^m,*
3. *$\psi \circ \varphi^{-1} : \varphi(U \cap V) \to \psi(U \cap V)$ to gładki dyfeomorfizm.*

Dla przypadku, gdy $U \cap V \neq \emptyset$, ostatni podpunkt implikuje $n = m$.

Definicja 12 *Zbiór $\mathcal{A} = \{(U_i, \varphi_i), i \in I\}$ parami gładko spójnych mapowań takich, że $\cup_{i \in I} U_i = M$, nazywamy gładkim atlasem M.*

Definicja 13 *Gładką rozmaitością nazywamy parę $\mathcal{M} = (M, \mathcal{A}^+)$, gdzie M jest zbiorem, a $\mathcal{A}^+$ jest maksymalnym atlasem M.*

Definicja 14 *Dla danej rozmaitości $\mathcal{M} = (M, \mathcal{A}^+)$, jeżeli wszystkie mapy z $\mathcal{A}^+$ mają ten sam rozmiar n, to wymiar $dim\mathcal{M} := n$ jest wymiarem rozmaitości $\mathcal{M}$.*

Definicja 15 *Niech $\mathcal{M}$ i $\mathcal{N}$ będą dwiema gładkimi rozmaitościami. Mapowanie $f : \mathcal{M} \to \mathcal{N}$ jest gładkie jeżeli, dla każdego $p \in \mathcal{M}$ istnieją mapa (U, ϕ) rozmaitości $\mathcal{M}$ i mapa (V, φ) rozmaitości $\mathcal{N}$ takie, że $p \in U, f(U) \subset V$ i*

$$\psi \circ f \circ \varphi^{-1} : \varphi(U) \to \psi(V),$$

jest klasy C^∞.

Definicja 16 *Niech $\mathcal{M} \subset \mathbb{R}^n$ rozmaitość gładka zanurzona w $\mathbb{R}^n$. Przestrzeń styczna w punkcie $x \in \mathcal{M}$, oznaczana jako $T_x\mathcal{M}$, to podprzestrzeń liniowa $\mathbb{R}^n$ zdefiniowana jako:*

$$T_x\mathcal{M} = \{v \in \mathbb{R}^n : v = c'(0) \text{ dla funkcji gładkiej } c : \mathbb{R} \to \mathcal{M}, \text{ takiej, że } c(0) = x\}.$$

Definicja 17 *Przestrzeń styczna do rozmaitości $\mathcal{M}$ w punkcie p, oznaczona jako $T_p\mathcal{M}$, to przestrzeń ilorazowa*

$$\mathrm{T}_p\mathcal{M} = C_p/\sim = \{[c] : c \in C_p\}.$$

Dla danego $c \in C_p$, klasa równoważności $[c]$, to element $T_p\mathcal{M}$ zwany wektorem stycznym do $\mathcal{M}$ w punkcie p.

Definicja 18 *Pochodna kierunkowa pola skalarnego f na $\mathcal{M}$ w punkcie $p \in \mathcal{M}$, w kierunku $\xi = [c] \in T_p\mathcal{M}$ to skalar:*

$$\mathrm{D}f(p)[\xi] := \frac{\mathrm{d}}{\mathrm{d}t} f(c(t))\Big|_{t=0} = (f \circ c)'(0).$$

Klasa równoważności C_p jest tak zdefiniowana, by definicja nie zależała od wyboru c, reprezentanta klasy równoważności ξ. Nawiasy kwadratowe wokół ξ mają oznaczać, że rozważamy kierunek, a nie klasę równoażności ξ.

Definicja 19 *Wiązkę styczną do gładkiej rozmaitości $\mathcal{M}$ oznaczamy jako $T\mathcal{M}$ i jest ona zbiorem:*

$$T\mathcal{M} = \coprod_{p\in\mathcal{M}} \mathrm{T}_p\mathcal{M},$$

Gdzie $\coprod$ oznacza sumę rozłączną.

Definicja 20 *Pole wektorowe X jest gładkim mapowaniem z rozmaitości $\mathcal{M}$ do $T\mathcal{M}$, które spełnia zależność, że $\pi \circ X = Id$ jest mapowaniem identycznościowym. Wektor w punkcie p jest oznaczany jako $X(p)$ i należy do $T_p\mathcal{M}$. Zbiór pól wektorowych na $\mathcal{M}$ jest oznaczany jako $\mathcal{X}(\mathcal{M})$.*

Definicja 21 *Niech $\mathcal{M}$ będzie gładką rozmaitością, a $p \in \mathcal{M}$ będzie ustalonym punktem z tej rozmaitości. Produkt wewnętrzny $\langle\cdot,\cdot\rangle_p$ zdefiniowany na $T_p\mathcal{M}$ jest symetryczną, dodatnio określoną formą dwuliniową na $T_p\mathcal{M}$. To znaczy, że $\forall\xi,\zeta,\eta \in \mathrm{T}_p\mathcal{M}$, oraz $\forall a, b \in \mathbb{R}$:*

- $\langle a\xi + b\zeta, \eta\rangle_p = a\langle\xi,\eta\rangle_p + b\langle\zeta,\eta\rangle_p$,
- $\langle\xi,\zeta\rangle_p = \langle\zeta,\xi\rangle_p$,
- $\langle\xi,\xi\rangle_p \geq 0$, *gdzie* $\langle\xi,\xi\rangle_p = 0 \Leftrightarrow \xi = 0$.

Definicja 22 *Metryką Riemannowską nazywamy produkt wewnętrzny zdefiniowany na przestrzeni stycznej $\mathcal{M}$, spełniający:*

- $\forall p \in \mathcal{M}$, $g_p(\cdot,\cdot) = \langle\cdot,\cdot,\rangle_p$ *jest produktem wewnętrznym na $T_p\mathcal{M}$,*
- *Dla każdych pól wektorowych $X, Y \in \mathcal{X}(\mathcal{M})$ rozmaitości $\mathcal{M}$, funkcja $p \mapsto g_p(X_p, Y_p)$ jest funkcją gładką z $\mathcal{M}$ do $\mathbb{R}$.*

Definicja 23 *Rozmaitość Riemannowska to para $(\mathcal{M}, g)$, gdzie $\mathcal{M}$ to gładka rozmaitość, a g jest metryką Riemannowską.*

Przestrzeń wektorowa z produktem wewnętrznym to specjalny przypadek rozmaitości Riemannowskiej zwany przestrzenią Euklidesową.

Definicja 24 *Niech f będzie polem skalarnym na rozmaitości Riemannowskiej $\mathcal{M}$. Gradient f w punkcie p, oznaczany przez* $\operatorname{grad} f(p)$*, definiujemy jako element $T_p\mathcal{M}$ spełniający:*

$$\mathrm{D}f(p)[\xi] = \langle \operatorname{grad} f(p), \xi \rangle_p, \quad \forall \xi \in \mathrm{T}_p\mathcal{M}.$$

Co oznacza, że $\operatorname{grad} f : \mathcal{M} \to T\mathcal{M}$ *jest polem wektorowym na $\mathcal{M}$.*

Definicja 25 *Niech $(\overline{\mathcal{M}}, \bar{g})$ będzie rozmaitością Riemannowską. Niech $(\mathcal{M}, g)$ będzie rozmaitością taką, że $\mathcal{M}$ jest podrozmaitością $\overline{\mathcal{M}}$, oraz g jest ograniczeniem $\bar{g}$ do przestrzeni stycznych $\mathcal{M}$. Wtedy $\mathcal{M}$ jest podrozmaitością Riemannowską rozmaitości $\overline{\mathcal{M}}$.*

Definicja 26 *Niech $\mathcal{X}(\mathcal{M})$ oznacza zbiór gładkich pól wektorowych $\mathcal{M}$, a $\mathcal{F}(\mathcal{M})$ oznacza zbiór gładkich pól skalarnych $\mathcal{M}$. Afinicznym połączeniem ∇ na rozmaitości $\mathcal{M}$ nazywamy mapowanie*

$$\nabla : \mathcal{X}(\mathcal{M}) \times \mathcal{X}(\mathcal{M}) \to \mathcal{X}(\mathcal{M}) : (X, Y) \mapsto \nabla_X Y,$$

które spełnia poniższe własności:

- *$\mathcal{F}(\mathcal{M})$-liniowość względem X : $\nabla_{fX+gY} Z = f\nabla_X Z + g\nabla_Y Z$,*
- *$\mathbb{R}$-liniowość względem Y : $\nabla_X(aY + bZ) = a\nabla_X Y + b\nabla_X Z$,*
- *Spełnia prawo Leibnitza: $\nabla_X(fY) = \nabla_X(f)Y + f\nabla_X(Y)$,*

gdzie $X, Y, Z \in \mathcal{X}(\mathcal{M}), f, g \in \mathcal{F}(\mathcal{M}), a, b \in \mathbb{R}$.

Definicja 27 *Niech M będzie gładką rozmaitością z połączeniem ∇. Niech $\gamma : I \to \mathcal{M}$, gdzie I jest otwartym przedziałem w $\mathbb{R}$ będącym krzywą klasy C^2 na $\mathcal{M}$. Przyspieszenie wzdłuż γ jest opisane wzorem:*

$$t \mapsto \nabla_{\dot\gamma(t)} \dot\gamma(t) \in \mathrm{T}_{\gamma(t)}\mathcal{M}.$$

Uwaga 27.1 *Powyższa definicja dla przypadku podrozmaitości przestrzeni Euklidesowej $\mathbb{R}^n$ sprowadza się do wzoru:*

$$\nabla_{\dot\gamma(t)} \dot\gamma(t) = P_{\gamma(t)} \gamma''(t),$$

gdzie $\gamma''(t)$ to klasyczna druga pochodna γ, rozpatrywana jako krzywa w $\mathbb{R}^n$.

Definicja 28 *Krzywa $\gamma : I \to \mathcal{M}$, gdzie I jest otwartym podzbiorem $\mathbb{R}$, jest krzywą geodezyjną wtedy i tylko wtedy, gdy ma zerowe przyspieszenie na całej swojej dziedzinie.*

Definicja 29 *Długość krzywej klasy C^1 $\gamma : [a, b] \to \mathcal{M}$, na rozmaitości Riemannowskiej $(\mathcal{M}, g)$, z produktem wewnętrznym $\langle \xi, \eta \rangle_p = g_p(\xi, \eta)$, definiujemy jako*

$$\operatorname{length}(\gamma) = \int_a^b \sqrt{\langle \dot\gamma(t), \dot\gamma(t) \rangle_{\gamma(t)}} \mathrm{d}t = \int_a^b \|\dot\gamma(t)\|_{\gamma(t)} \mathrm{d}t$$

Definicja 30 *Odległość geodezyjną na rozmaitości $\mathcal{M}$ opisujemy wzorem:*

$$\textit{dist: } \mathcal{M} \times \mathcal{M} \to \mathbb{R}^+ : (p, q) \mapsto \operatorname{dist}(p, q) = \inf_{\gamma \in \Gamma} \operatorname{length}(\gamma),$$

gdzie Γ jest zbiorem wszystkim krzywych klasy C^1 $\gamma : [0, 1] \to \mathcal{M}$ takich, że $\gamma(0) = p$ i $\gamma(1) = q$.

Definicja 31 *Retrakcją na rozmaitość $\mathcal{M}$ nazywamy gładkie mapowanie R z wiązki stycznej $T\mathcal{M}$ na $\mathcal{M}$, spełniające poniższe własności. Dla każdego $x \in \mathcal{M}$, niech R_x oznacza ograniczenie R do $T_x\mathcal{M}$. Zachodzi wtedy*

- $R_x(0) = x$, *gdzie* 0 *jest elementem zerowym* $T_x\mathcal{M}$,
- *Różniczka* $(DR_x)_0 : T_0(T_xM) \equiv T_x\mathcal{M} \to T_x\mathcal{M}$ *jest mapowaniem identycznościowym na* $T_x\mathcal{M}$, *to znaczy* $(DR_x)_0 = \mathrm{Id}$.

Bibliografia

[1] Xiantao Xiao, Yongfeng Li, Zaiwen Wen, Liwei Zhang, *A regularized semi-smooth newton method with projection steps for composite convex programs*, Springer, Dalian, 2016

[2] Nicolas Boumal, *Optimizationan estimation on manifolds*, Universite catholique de Louvain, Louvain, 2014

[3] Artiom Kovnatsky, Klaus Glashoff, Michael M. Bronstein, *MADMM: a generic algorithm for non-smooth optimization on manifolds*, Springer, Lugano, 2015

[4] Shixiang Chen, Shiqian Ma, Anthony Man-Cho So, Tong Zhang, *Proximal Gradient Method for Nonsmooth Optimization over the Stiefel Manifold*, wersja elektroniczna dostępna pod adresem: `https://arxiv.org/abs/1811.00980`, Hong Kong, 2019

[5] Michael Weylandt, *Multi-rank sparse and functional PCA manifold optimization and iterative deflation techniques*, praca dostępna elektronicznie pod: `https://arxiv.org/abs/1907.12012`, Houston, 2019

[6] P.-A. Absil, Robert Mahony, Rodolphe Sepulchre, *Optimization algorithms on matrix manifolds*, Princeton University Press, New Jersey, 2008

[7] Alan Edelman, Tomas A. Arias, Steven T. Smith, *The geometry of algorithms with orthogonality constraints*, SIAM Journal on Matrix Analysis and Applications, Massachusetts, 1998

[8] Hui Zou,Trevor Hastie, Robert Tibshirani, *Sparse principal component analysis*, Journal of Computational and Graphical Statisics, Stanford, 2006

[9] Trevor Hastie, Rahul Mazumder, Jason D. Lee, Reza Zadeh, *Matrix completion and low-rank SVD via fast alternating least squares*, Journal of Machine Learning Research, Brookline, 2015

[10] Amir Beck, *First-order methods in optimization*, Society for Industrial and Applied Mathematics, Tel-Aviv, 2017

[11] Neal Parikh, Stephen Boyd, *Proximal algorithms*, Foundations and Trends in Optimization, Stanford, 2013

[12] Heinz Bauschke, Patrick Combettes, *Convex analysis and monotone operator theory in hilbert spaces*, str. 14-22, Springer, Second edtion, London, 2017

[13] Amir Beck, Marc Teboulle, *Gradient-Based Algorithms with Applications to Signal Recovery Problems*, Cambridge University Press, Cambridge, 2010

[14] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, Jonathan Eckstein, *Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers*, Foundations and Trends in Machine Learning, vol. 3, no. 1, str. 1-112, Stanford, 2011

[15] P.-A. Absil, R. Mahon, R.Sepulchre, *Riemannian geometry of Grassmann manifolds with a view on algorithmic computation*, Acta Applicandae Mathematicae, Volume 80, Issue 2, str. 199-220, Liege, 2004

[16] Lutz Prechelt, *Early Stopping – but when?*, wersja dostępna elektronicznie pod adresem: `https://page.mi.fu-berlin.de/prechelt/Biblio/stop_tricks1997.pdf`, Berlin, 1997

[17] Leon Mirsky, *Symmetric Gauge Functions and Unitarily Invariant Norms*, The quarterly journal of mathematics, 11(1) str. 50-59, Sheffield, 1960.

[18] Trevor Hastie, Robert Tibshirani, Martin Wainwright, *Statistical Learning with Sparsity: The Lasso and Generalizations*, Monographs on Statistics and Applied Probability 143, CRC Press, Stanford, 2016

[19] Stephen Wright, *Coordinate Descent Algorithms*, praca dostępna elektronicznie pod: `https://arxiv.org/abs/1502.04759`, Madison, 2015

[20] Błażej Miasojedow, Wojciech Rejchel, *Sparse Estimation in Ising Model via Penalized Monte Carlo Methods*, Journal Of Machine Learning Research 19(75) str. 1-26, Warszawa, 2018

[21] Shuheng Zhou, *Thresholded Lasso for high dimensional variable selection and statistical estimation*, wersja elektroniczna dostępna pod adresem: `https://arxiv.org/abs/1002.1583`