



POLITECHNIKA WARSZAWSKA
WYDZIAŁ MATEMATYKI I NAUK INFORMACYJNYCH



PRACA DYPLOMOWA MAGISTERSKA
NA KIERUNKU MATEMATYKA
SPECJALNOŚĆ STATYSTYKA MATEMATYCZNA I ANALIZA DANYCH

**ESTYMACJA W MODELU COXA
METODĄ STOCHASTYCZNEGO SPADKU GRADIENTU
Z PRZYKŁADAMI ZASTOSOWAŃ W ANALIZIE DANYCH
Z THE CANCER GENOME ATLAS**

AUTOR:
MARCIN PIOTR KOSIŃSKI

PROMOTOR:
DR HAB. INŻ. PRZEMYSŁAW BIECEK, PROF. NADZW.

WARSZAWA, GRUDZIEŃ 2015

.....

podpis promotora

.....

podpis autora

Streszczenie

Praca przedstawia zastosowanie algorytmu numerycznej optymalizacji metodą stochastycznego spadku gradientu do wyznaczenia współczynników w modelu proporcjonalnych hazardów Coxa, opartych na analitycznej metodzie wyznaczania estymatorów metodą największej wiarygodności. Zaprezentowano zastosowanie tego algorytmu do napływających danych, wyliczając kolejne kroki optymalizacji w algorytmie w oparciu o częściową funkcję log-wiarodności, konstruowaną na bazie obecnie zaobserwowanego podzbioru obserwacji. Zaprezentowane podejście jest nowatorskie.

Opisano matematyczne własności estymatorów największej wiarodności, przedstawiono model Coxa wraz z najważniejszymi jego cechami, scharakteryzowano algorytm stochastycznego spadku gradientu oraz jego różnice w stosunku do algorytmów spadku gradientu, zaimplementowano algorytm numerycznej optymalizacji metodą stochastycznego spadku gradientu do wyznaczenia współczynników w modelu proporcjonalnych hazardów Coxa oraz przeprowadzono analizę rzeczywistych danych genetycznych, wykorzystując stworzony algorytm. Prace wzbogacono o symulacje zbieżności procesu optymalizacji funkcji log-wiarodności w modelu regresji logistycznej, które przygotowane zostały dla algorytmów spadku gradientu rzędu I, spadku gradientu rzędu II oraz stochastycznego spadku gradientu rzędu I. Dzięki temu możliwe było zobrazowanie różnic w opisywanych metodach numerycznej optymalizacji. Dodatkowo przeprowadzono symulacje przy sztucznie wygenerowanych danych do analizy przeżycia z rozkładu Weibulla, pozwalające zobrazować proces zbieżności nowo wprowadzonego algorytmu numerycznej optymalizacji metodą stochastycznego spadku gradientu do wyznaczenia współczynników w modelu proporcjonalnych hazardów Coxa.

Proces estymacji w modelu proporcjonalnych hazardów Coxa, z wykorzystaniem metody stochastycznego spadku gradientu, w sytuacji napływających danych wygląda na stabilny i zbiega w okolice bliskie do teoretycznego optimum, niekiedy nawet już po jednej epoce. Przy dobrze dobranych parametrach optymalizacji, proces może być z powodzeniem wykorzystywany do znajdowania współczynników modelu. Kluczowym momentem, mającym wpływ na powodzenie optymalizacji, jest wybór ciągu odpowiedzialnego za dobór długości kroków, więc zaleca się symulacje badające różne ciągi oraz wybranie tego ciągu, który w większości przypadków daje stabilne, jednorodne współczynniki, maksymalizujące częściową funkcję log-wiarodności, konstruowaną w oparciu o zbiór testowy.

Do analizy genetycznej wykorzystano środowisko statystyczne \mathcal{R} , a wykorzystane funkcje, zaimplementowane algorytmy i przygotowaną dokumentację do stworzonego w trakcie pracy pakietu `coxphSGD` zamieszczono w Dodatku A. W analizie badano wpływ występowania mutacji w poszczególnych genach na czas przeżycia pacjentów, dla których dane kliniczne i dane o występowaniu mutacji zaczerpnięto z badania *The Cancer Genome Atlas*. Zaprezentowano skuteczność badanego modelu oraz wskazano geny odpowiedzialne za zwiększenie ryzyka zgonu w trakcie choroby nowotworowej.

Słowa kluczowe

model proporcjonalnych hazardów Coxa, stochastyczny spadek gradientu, estymacja metodą największej wiarygodności, częściowa funkcja wiarygodności, analiza przeżycia, pakiet statystyczny \mathcal{R} , napływające dane, nowotworzenie, *The Cancer Genome Atlas*

Tytuł pracy w języku angielskim

Stochastic gradient descent method in Cox models with applications to
The Cancer Genome Atlas data.

Spis treści

Wprowadzenie	7
Podstawy modelu statystycznego	11
1. Estymacja metodą największej wiarogodności	13
1.1. Estymacja	13
1.2. Metoda największej wiarogodności	15
1.3. Asymptotyczne własności estymatora największej wiarogodności	15
2. Model Coxa	21
2.1. Wprowadzenie do modelu Coxa i terminologia	21
2.2. Założenia modelu proporcjonalnego ryzyka Coxa	22
2.3. Estymacja w modelu Coxa	24
2.4. Generowanie danych dla modelu Coxa	26
3. Numeryczne metody estymacji	29
3.1. Algorytmy spadku wzduż gradientu	30
3.2. Algorytm stochastycznego spadku wzduż gradientu I	31
3.3. Porównanie algorytmów spadku wzduż gradientu	32
4. Estymacja w modelu Coxa metodą stochastycznego spadku gradientu	41
4.1. Założenia i obserwacje	42
4.2. Implementacja	44
4.3. Symulacje estymacji w modelu Coxa	46
5. Analiza danych genomicznych	53
5.1. Genetyczne podstawy nowotworzenia	54
5.2. Projekt The Cancer Genome Atlas	55
5.3. Analiza wpływu mutacji genów na czas życia	56
Podsumowanie	63
A. Wykorzystane narzędzia, dokumentacja i kody pakietu \mathcal{R} użyte w pracy	65
A.1. Model proporcjonalnych hazardów Coxa	65
A.2. Dokumentacja pakietu <code>coxphSGD</code>	68
A.3. Analiza wpływu mutacji genów na czas życia	72
A.4. Implementacje optymalizacji w regresji logistycznej	74
Literatura	77

Wprowadzenie

W ciągu ostatniej dekady rozmiary danych rosły szybciej niż prędkość procesorów. W tej sytuacji możliwości statystycznych metod uczenia maszynowego stały się ograniczone bardziej przez czas obliczeń niż przez rozmiary zbiorów danych. Rozwiązania kompromisowe w przypadku dużej skali danych związane są ze złożonością obliczeniową zasadniczych algorytmów optymalizacyjnych, których należy dokonywać w nietrywialny sposób. Jednym z takich rozwiązań są algorytmy optymalizacyjne oparte o stochastyczny spadek gradientu ([Bottou, 2010, 2012; Widrow and Stearns, 1985](#)), które wykazują dużą wydajność w trakcie pracy z danymi wielkiej skali.

W niniejszej pracy przedstawiono algorytm estymacji współczynników w modelu Coxa metodą stochastycznego spadku gradientu. Opisany algorytm może z powodzeniem być stosowany w analizach czasu do wystąpienia zdarzenia, w których liczba zmiennych znacząco przekracza liczbę obserwacji. Przygotowana metoda estymacji współczynników z wykorzystaniem algorytmu optymalizacji metodą stochastycznego spadku gradientu może być stosowana w analizach przeżycia z dziedziny biologii molekularnej, bioinformatycznych badań przesiewowych dotyczących ekspresji genów czy w analizach opartych o mikromacierze DNA, które są szeroko stosowane w diagnostyce, leczeniu i badaniach medycznych. Zaprezentowana schemat estymacji współczynników w modelu Coxa z wykorzystaniem algorytmu optymalizacji metodą stochastycznego spadku gradientu jest podejściem nowym i nie spotkanym do tej pory w literaturze, jest odporna na problem współliniowości zmiennych oraz sprawdza się w sytuacji ciągłej poprawy współczynników dla napływających danych (*ang. streaming data*).

Analiza przeżycia jest starą, jednak wciąż aktywną dziedziną badań znajdująca nowe zastosowania w wielu dziedzinach, takich jak: biostatystyka, socjologia, ekonomia, demografia czy w naukach inżynierijnych ([Box-Steffensmeier and Jones, 2004; Collett, 1994; Heckman and Singer, 1985; Hosmer et al., 2008](#)). Najbardziej charakterystyczną cechą typowych danych, jakimi posługuje się w analizie przeżycia, jest obecność obiektów, w których końcowe zdarzenie nastąpiło (wówczas ma się do czynienia z obserwacjami *kompletnymi*) oraz obiektów, w których to zdarzenie (jeszcze) nie nastąpiło (obserwacja *ucięta*). Ta specyficzna postać danych statystycznych doprowadziła do powstania specjalnych metod stosowanych tylko w analizie czasu trwania zjawisk. Analiza przeżycia charakteryzuje relacje pomiędzy czasem do wystąpienia zdarzenia a zmiennymi objaśniającymi ([Kalbfleisch and Prentice, 1980; Oakes, 2001](#)) i do niedawna ograniczona była jedynie do zastosowań oraz analiz opartych na garści zmiennych objaśniających przy maksymalnie kilku tysiącach obserwacji. Ostatnie dokonania w obszarach rozwoju technik pozyskiwania danych i ułatwiony wraz z postępem cywilizacyjnym dostęp do większej mocy obliczeniowej spowodowały wzmożenie zainteresowania danymi z potencjalnie setkami tysięcy, a nawet milionami zmiennych. Przykładem mogą być nowe technologie w genomice produkujące wielowymiarowe mikromacierze ekspresji genów, w których liczba zmiennych prognozujących przeżycie może sięgać rozmiarom rzędu 10^5 lub nawet większym. Inne przykłady zastosowań analizy przeżycia dla danych wielkiej skali to badania monitorowania medycznych zdarzeń niepożądanych, wielopomiarowe i rozłożone w czasie badania kliniczne czy analizy eksploracyjne (*ang. data mining*) danych biznesowych.

Praca przedstawia podejście do analizy przeżycia z wykorzystaniem modelu proporcjonalnych hazardów Coxa (Cox, 1972). Jak podaje Asselain and Mould (2010), omawiany model jest jednym z najszerzej stosowanych modeli w onkologicznych publikacjach naukowych, ale także jedną z najmniej rozumianych metod statystycznych. Wynika to z łatwego dostępu do pakietów statystycznych zawierających programy do analizy przeżyć, modeli regresji i analiz wielowariantowych, ale prawie nigdy nie zawierających dobrego opisu podstawowych zasad działania modelu Coxa. Dostarczają one wyłącznie instrukcje, jak wprowadzić dane i uruchomić odpowiednie procedury w celu uzyskania wyniku. Poniższa praca zawiera pełny opis metodologii modelu proporcjonalnych hazardów Coxa, w tym wyjaśnienie najważniejszych pojęć. W odróżnieniu do modeli parametrycznych analizy czasu do wystąpienia zdarzenia (Collett, 1994; Hosmer et al., 2008; Klein and Moeschberger, 2003), model Coxa oferuje większą elastyczność, dzięki swojej semi-parametrycznej naturze, dając jednocześnie współczynniki regresji, które są łatwo interpretowalne. Zazwyczaj współczynniki w modelu Coxa otrzymuje się dzięki maksymalizacji częściowej funkcji log-wiarogodności modelu. Jednak w sytuacji przekleństwa wymiarowości standardowe metody estymacji metodami spadku gradientu, takimi jak metoda Cauchy'ego czy Raphsona-Newtona zawodzą z racji na zbyt złożone i długotrwałe obliczenia. Dodatkowy problem stanowi współliniowość zmiennych, co utrudnia utrzymanie numerycznej stabilności algorytmów optymalizacyjnych.

W literaturze zaproponowano wiele rozwiązań problemu współliniowości poprzez dodanie do funkcji częściowej log-wiarogodności dodatkowego parametru związanego z karą dla modelu za zbyt dużą liczbę zmiennych w trakcie estymacji (Goeman, 2010; Park and Hastie, 2007; Sohn et al., 2009). Metoda ta, zwana regularyzacją, prowadzi do zminimalizowania liczb zmiennych w modelu poprzez wyzerowanie współczynników dla nieistotnych statystycznie zmiennych. Jednak powyższe rozwiązania znalazły zastosowania jedynie dla zbiorów danych małej skali. Takie podejście nie przeniosło się na zbiory dużej skali ze względu na występowanie w algorytmie spadku gradientu Raphsona-Newtona kosztownych kroków obliczeniowych algorytmu wymagających odwracania wielkich macierzy, co numerycznie nie jest proste. Ponadto, jak opisuje Mittal and Madigan (2014) możliwe obejścia i przybliżenia często prowadzą do dużych różnic współczynników, złego dopasowania modelu czy słabego oszacowania predykcyjnej dokładności. Rozwój badań nad danym problemem (Kim and Kim, 2004) doprowadził do powstania metod wykorzystujących jedynie pierwszą pochodną częściowej penalizowanej (z parametrem regularyzacji) funkcji log-wiarogodności w modelu Coxa (Kim et al., 2008; Mittal and Madigan, 2014; Sohn et al., 2009). Zabiegi wykorzystujące jedynie pierwszą pochodną są odporne na problemy danych dużej skali, gdyż nie wymagają obliczania i odwracania macierzy Hesjanu.

Podejścia te trudno wykorzystać w sytuacjach napływu danych, gdy nie dysponuje się wszystkimi obserwacjami jednocześnie, a zachodzi potrzeba wykorzystania tych już zaobserwowanych do estymacji współczynników oraz rodzi się okazja do poprawy ich estymatorów z każdą nową obserwacją (Bottou, 1998). Dlatego w warunkach napływu danych, możliwym rozwiązaniem jest zastosowanie algorytmu optymalizacji metodą stochastycznego spadku gradientu, który w wielu modelach do estymacji współczynników wymaga jednej obserwacji. W modelu proporcjonalnych hazardów Coxa wykorzystanie metody stochastycznego spadku gradientu możliwe jest dla zaobserwowanych kilku obserwacji, z racji na postać częściowej funkcji log-wiarogodności, której kolejne składniki są zależne od innych obserwacji. Szerokie badania w celu wykorzystania metody stochastycznego spadku gradientu do estymacji w modelu Coxa prowadzą pracownicy *Harvard Laboratory for Applied Statistical Methodology & Data Science*, którzy zaproponowali podejście (*ang. implicit*) uwiklanego stochastycznego spadku gradientu (Toulis and Aioldi, 2015).

W niniejszej pracy zaprezentowano prostsze podejście wykorzystania algorytmu stochastycznego spadku gradientu do estymacji współczynników rozważanego modelu proporcjonalnych hazardów Coxa. Wprowadzone rozwiązanie opiera się na optymalizacji częściowej funkcji log-wiarogodności konstruowanej jedynie do obecnie zaobserwowanego podzbioru obserwacji. Ta strategia pozwala na efektywne osiągnięcie zbieżności algorytmu optymalizacji w sytuacji napływających danych, o dużym rozmiarze zmiennych objaśniających, przy jednoczesnej odporności na ich współliniowość oraz umożliwia otrzymanie oszacowań współczynników po każdym zaobserwowanym podzbiorze obserwacji.

Rozdział 1 pracy opisuje metodę największej wiarogodności i jej matematyczne podstawy, dzięki którym możliwe jest wyznaczanie estymatorów największej wiarogodności. Uzasadniona również poprawność ich wykorzystywania poprzez udowodnienie ich własności, takich jak: asymptotyczna normalność, asymptotyczna nieobciążoność oraz zgodność.

Rozdział 2 poświęcony jest modelowi proporcjonalnych hazardów Coxa. Przedstawione są podstawowe terminy i definicje analizy przeżycia. Rozdział opisuje niezbędne założenia modelu Coxa oraz prezentuje metodę analitycznej estymacji w oparciu o szeroko opisaną częściową funkcję log-wiarogodności modelu i estymatory największej wiarogodności. Ta część pracy kończy się opisaniem metody generowania danych w modelu Coxa oraz implementacją funkcji generującej dane do analizy przeżycia pochodzące z rozkładu Weibulla.

W rozdziale 3 scharakteryzowany jest algorytm numerycznej optymalizacji metodą stochastycznego spadku gradientu. Ukażane zostały jego wady i zalety oraz przedstawiono różnice między tym algorytmem a algorytmami spadku gradientu rzędu I (Cauchy'ego) oraz spadku gradientu rzędu II (Raphsona-Newtona). Ten fragment pracy podsumowany jest implementacją trzech omówionych algorytmów numerycznej optymalizacji dla modelu regresji logistycznej. Implementacja posłużyła do przeprowadzenia symulacji, dzięki którym możliwe było graficzne ukazanie różnic w ścieżkach zbiegania do optimum między algorytmami spadku gradientu i ich stochastycznym odpowiednikiem.

Rozdział 4 przedstawia matematyczne podstawy zastosowania algorytmu numerycznej optymalizacji metodą stochastycznego spadku gradientu do estymacji współczynników w modelu proporcjonalnych hazardów Coxa w oparciu o analityczną metodę estymacji największej wiarogodności zastosowaną do częściowej funkcji log-wiarogodności skonstruowaną jedynie dla obecnie zaobserwowanego podzbioru danych w sytuacji napływu danych. Opisane są w nim warunki konieczne, jakie musi spełniać zaobserwowany podzióbior danych, aby nie przerwać procesu optymalizacji częściowej funkcji log-wiarogodności. Dodatkowo zaprezentowano implementację opisanej metody w pseudo kodzie oraz podano implementację wyrażoną w języku *R* ([Biecek, 2011](#); [R Core Team, 2013](#)). Rozdział 4 kończy się przeprowadzeniem symulacji mających na celu wygenerowanie danych z modelu Coxa, dzięki wykorzystaniu implementacji z rozdziału 2, które wykorzystano w przygotowanym algorytmie do wyestymowania współczynników w modelu Coxa. Wyniki symulacji przedstawiono na wykresach i porównano ścieżki optymalizacji dla różnych wielkości zaobserwowanych podzbiorów obserwacji wykorzystanych do jednego kroku algorytmu.

Praca kończy się rozdziałem 5, w którym metoda estymacji w modelu Coxa proporcjonalnych hazardów z wykorzystaniem stochastycznego spadku gradientu zastosowana jest do analizy na prawdziwych danych pochodzących z badania *The Cancer Genome Atlas* (TCGA). Rozdział referuje genetyczne podstawy nowotworzenia oraz opisuje dane z TCGA wraz z procesem ich pozyskania i przetwarzania ([Kosiński, 2015b,c](#); [Kosiński and Biecek, 2015](#)). Analiza ma na celu sprawdzenie wpływu występowania mutacji w danym genie na czas do wystąpienia zdarzenia niepożądanego, jakim jest śmierć w wyniku choroby nowotworowej.

Zestawienie i podsumowanie kodów pakietu \mathcal{R} użytych do symulacji oraz analizy przeżycia można znaleźć w Dodatku A. Implementacja estymacji w modelu Coxa proporcjonalnych hazardów metodą stochastycznego spadku gradientu oraz funkcje symulujące dane i generujące wykresy porównujące trajektoria zbieżności algorytmu zostały opakowane w pakiet do \mathcal{R} o nazwie `coxSGD` (Kosiński, 2015a) oraz umieszczone w internecie pod adresem <https://github.com/MarcinKosinski/coxphSGD>. Dokumentacja pakietu w języku angielskim została przedstawiona w Dodatku A.

Podstawy modelu statystycznego

W pracy zakłada się znajomość podstaw statystyki matematycznej. Aby ujednolicić oznaczenia, wprowadzono klasyczną terminologię w oparciu o [Niemiro \(2011\)](#).

Definicja 0.1. *Model statystyczny* określamy przez podanie rodziny $\{\mathbb{P}_\theta : \theta \in \Theta\}$ rozkładów prawdopodobieństwa na przestrzeni próbkowej Ω oraz zmiennej losowej $X : \Omega \rightarrow \mathcal{X}$, którą traktujemy jako obserwację. Zbiór \mathcal{X} nazywamy przestrzenią obserwacji, zaś Θ nazywamy przestrzenią parametrów.

Symbol θ jest nieznanym parametrem opisującym rozkład badanego zjawiska. Może być jednowymiarowy lub wielowymiarowy. Determinując opis zjawiska poprzez podanie parametru θ , jednoznacznie wyznaczany jest rozkład rozważanego zjawiska spośród całej rodziny rozkładów prawdopodobieństwa $\{\mathbb{P}_\theta : \theta \in \Theta\}$, co umożliwia określenie prawdziwości tezy.

Zakłada się, że przestrzeń próbkowa Ω jest wyposażona w σ -ciało \mathcal{F} . Wtedy:

Definicja 0.2. *Przestrzeń statystyczna* nazywa się trójką $(\mathcal{X}, \mathcal{F}, \{\mathbb{P}_\theta : \theta \in \Theta\})$.

Wprowadzenie σ -ciała \mathcal{F} sprawia, że przestrzeń statystyczna staje się przestrzenią mieralną, a więc można na niej określić rodzinę $\{\mathbb{P}_\theta : \theta \in \Theta\}$, dzięki której da się ustalić prawdopodobieństwa zajścia wszystkich zjawisk w rozważanej teorii.

W celu budowania niezbędnych pojęć potrzebna jest również definicja losowej próby statystycznej, zazwyczaj nazywanej *próbką*.

Definicja 0.3. *Losową próbą statystyczną* nazywamy zbiór obserwacji statystycznych wylosowanych z populacji, które są realizacjami ciągu zmiennych losowych o rozkładzie takim jak rozkład populacji.

Rozdział 1

Estymacja metodą największej wiarogodności

*The making of maximum likelihood was one of
the most important developments in 20th century statistics.
It was the work of one man but it was no simple process (...).
John Aldrich o R. A. Fisher'ze*

1.1. Estymacja

Estymacja to dział wnioskowania statystycznego, będący zbiorem metod pozwalających na uogólnianie wyników badania próby losowej na nieznaną postać i parametry rozkładu zmiennej losowej całej populacji oraz szacowanie błędów, wynikających z tego uogólnienia.

W statystyce parametrycznej zakłada się, że rozkład prawdopodobieństwa opisujący doświadczenie należy do rodziny $\{\mathbb{P}_\theta : \theta \in \Theta\}$, ale nieznany jest parametr θ . Można go jednak szacować dzięki estymatorom opartym na statystykach.

Definicja 1.1. *Statystyka, dla $X = (X_1, \dots, X_n)$, to odwzorowanie mierzalne $T : \mathcal{X} \rightarrow \mathcal{R}$.*

Definicja 1.2. *Estymatorem parametru θ nazywamy dowolną statystykę $T = T(X)$, gdzie X to próba z badanego rozkładu, o wartościach w zbiorze Θ .*

Interpretuje się T jako przybliżenie θ i często estymator θ oznacza symbolem $\hat{\theta}$. Niekiedy potrzebna jest również estymacja $g(\theta)$, gdzie g to ustalona funkcja.

Pewne estymatory mające odpowiednie własności są preferowane nad inne ze względu na większą precyzję bądź ufność oszacowania danego estymatora. Poniżej przedstawione są dwie ważne definicje związane z jakością estymatorów (Niemiro, 2011), gdy rozmiar próbki X_1, \dots, X_n jest duży. Mówią się wtedy o własnościach asymptotycznych estymatorów, które z matematycznego punktu widzenia są twierdzeniami granicznymi, w których n dąży do nieskończoności. Dzięki tym twierdzeniom możliwe jest opisanie przybliżonych zachowań estymatorów dla dostatecznie dużych próbek. Niestety, teoria asymptotyczna nie dostarcza informacji o tym, jak duża powinna być próbka, żeby przybliżenie było dostatecznie dobre.

Definicja 1.3. Estymator $\hat{g}(X_1, \dots, X_n)$ wielkości $g(\theta)$ jest **nieobciążony**, jeśli dla każdego n

$$\mathbb{E}\hat{g}(X_1, \dots, X_n) = g(\theta).$$

Definicja 1.4. Estymator $\hat{g}(X_1, \dots, X_n)$ wielkości $g(\theta)$ jest **zgodny**, jeśli dla każdego $\theta \in \Theta$

$$\lim_{n \rightarrow \infty} \mathbb{P}_\theta(|\hat{g}(X_1, \dots, X_n) - g(\theta)| \leq \varepsilon) = 1,$$

dla każdego $\varepsilon > 0$.

Definicja 1.5. Estymator $\hat{g}(X_1, \dots, X_n)$ wielkości $g(\theta)$ jest **mocno zgodny**, jeśli

$$\mathbb{P}_\theta\left(\lim_{n \rightarrow \infty} \hat{g}(X_1, \dots, X_n) = g(\theta)\right) = 1.$$

Zgodność (mocna zgodność) znaczy tyle, że

$$\hat{g}(X_1, \dots, X_n) \rightarrow g(\theta), \quad (n \rightarrow \infty)$$

według prawdopodobieństwa (prawie na pewno). Interpretacja jest taka: estymator jest uznanym za zgodny, jeśli zmierza do estymowanej wielkości przy nieograniczonym powiększaniu badanej próbki.

Jednak zgodność (nawet w mocnym sensie) nie jest satysfakcjonującą własnością estymatora, a zaledwie minimalnym żądaniem, które powinien spełniać każdy przyzwoity estymator. Dlatego od niektórych estymatorów wymaga się silniejszych właściwości, takich jak asymptotyczna normalność.

Definicja 1.6. Estymator $\hat{g}(X_1, \dots, X_n)$ wielkości $g(\theta)$ jest **asymptotycznie normalny**, jeśli dla każdego $\theta \in \Theta$ istnieje funkcja $\sigma^2(\theta)$, zwana **asymptotyczną wariancją**, taka że

$$\sqrt{n}(\hat{g}(X_1, \dots, X_n) - g(\theta)) \xrightarrow{d} \mathcal{N}(0, \sigma^2(\theta)), \quad (n \rightarrow \infty).$$

wg

rozkładu.

Oznacza to, że rozkład prawdopodobieństwa statystyki $\hat{g}(X_1, \dots, X_n)$ jest dla dużych n zbliżony do rozkładu normalnego o średniej $g(\theta)$ i wariancji $\frac{\sigma^2(\theta)}{n}$ oznaczanego jako

$$\mathcal{N}\left(g(\theta), \frac{\sigma^2(\theta)}{n}\right).$$

Inaczej mówiąc, estymator jest asymptotycznie normalny, gdy:

$$\lim_{n \rightarrow \infty} \mathbb{P}_\theta\left(\frac{\sqrt{n}}{\sigma(\theta)}(\hat{g}(X_1, \dots, X_n) - g(\theta)) \leq a\right) = \Phi(a),$$

gdzie $\Phi(x)$ to dystrybuanta standardowego rozkładu normalnego $\mathcal{N}(0, 1)$.

Asymptotyczna normalność mówi, że estymator nie tylko zbiega do nieznanego parametru, lecz także że zbiega wystarczająco szybko, jak $\frac{1}{\sqrt{n}}$, czyli, że

$$\mathbb{P}_\theta\left(\frac{\sqrt{n}}{\sigma(\theta)}(\hat{g}(X_1, \dots, X_n) - g(\theta)) \leq a\right) - \Phi(a) = f(n) \in o\left(\frac{1}{\sqrt{n}}\right).$$

Każdy asymptotycznie normalny estymator jest zgodny, choć nie musi być *mocno zgodny*.

W dalszej części rozdziału zostanie wprowadzone pojęcie estymatora największej wiarogodności oraz zostaną udowodnione jego właściwości, co pokaże, że metoda największej wiarogodności, przy odpowiednich założeniach, to metoda konstrukcji rozsądnych estymatorów.

1.2. Metoda największej wiarogodności

Metodę największej wiarogodności wprowadził Fisher (1922), dla której po raz pierwszy heurystyczną procedurę numeryczną zaproponował już w Fisher (1912). O burzliwym procesie powstawania metody, zmianach w jej uzasadnieniu, o koncepcjach które powstały w obrębie tej metody, takich jak parametr, statystyka, wiarogodność, dostateczność czy efektywność oraz o podejściach, które Fisher odrzucił tworząc podstawy pod nową teorię, można przeczytać w obszernej pracy dokumentalnej Aldrich (1997).

Pomysł Fishera, alternatywy do metody najmniejszych kwadratów (Gauss, 1809; Legendre, 1804) był rozwijany i szeroko stosowany później przez wielu statystyków i wciąż znajduje obszerne zastosowania w wielu obszarach estymacji statystycznej (Hutchinson, 1928; Kenward et al., 1994; Millar, 2011).

Aby zdefiniować estymator oparty o metodę największej wiarogodności, należy najpierw wprowadzić pojęcie funkcji wiarogodności.

Definicja 1.7. *Funkcję wiarogodności nazywamy funkcję $L : \Theta \rightarrow \mathbb{R}$ daną wzorem*

$$L(\theta) = L(\theta; x_1, \dots, x_n) = f(\theta; x_1, \dots, x_n),$$

która rozważamy jako funkcję parametru θ przy ustalonych wartościach obserwacji x_1, \dots, x_n , gdzie

$$f(\theta; x_1, \dots, x_n) = \begin{cases} \mathbb{P}_\theta(X_1 = x_1, \dots, X_n = x_n), & \text{jeśli } \mathbb{P}_\theta \text{ ma rozkład dyskretny,} \\ f_\theta(x_1, \dots, x_n), & \text{jeśli } \mathbb{P}_\theta \text{ ma rozkład absolutnie ciągły.} \end{cases}$$

Oznacza to, że wiarogodność jest właściwie tym samym, co gęstość prawdopodobieństwa, ale rozważana jako funkcja parametru θ , przy ustalonych wartościach obserwacji $x = X(\omega)$.

Definicja 1.8. *Estymatorem największej wiarogodności parametru θ , oznaczanym $ENW(\theta)$, nazywamy wartość parametru, w której funkcja wiarogodności przyjmuje supremum*

$$L(\hat{\theta}) = \sup_{\theta \in \Theta} L(\theta).$$

Takie supremum może nie istnieć, dlatego niektóre pozycje w literaturze, w definicji estymatora największej wiarogodności, supremum zastępują wartością największą (Panchenko, 2006a; Rydlewski, 2009; Woodcock, 2014).

1.3. Asymptotyczne własności estymatora największej wiarogodności

W tym podrozdziale zostanie wykazane, że estymator największej wiarygodności jest

- i) zgodny,
- ii) asymptotycznie normalny,
- iii) asymptotycznie nieobciążony.

Asymptotyczna nieobciążoność wynika z asymptotycznej normalności. Przedstawione dowody są znane i opierają się o Panchenko (2006b) oraz Woodcock (2014).

Zgodność estymatora największej wiarogodności

Chcąc wykazać zgodność estymatora największej wiarogodności *przy pewnych warunkach regularności* przydatna będzie poniższa definicja i następujący Lemat.

Definicja 1.9. *Funkcja log-wiarogodności to funkcja spełniająca równanie*

$$\ell(\theta) = \log(L(\theta)),$$

gdzie przyjmuje się $\ell(\theta) = -\infty$ jeśli $L(\theta) = 0$.

Lemat 1.1. *Gdy θ_0 to maksimum funkcji wiarogodności, to dla każdego $\theta \in \Theta$*

\mathbb{E}_{θ_0} oznacza
wartość
oczekiwana
względem
rozkładu \mathbb{P}_{θ_0}

$$\mathbb{E}_{\theta_0} \ell(\theta) \leq \mathbb{E}_{\theta_0} \ell(\theta_0).$$

Dowód. Rozważając różnicę, przy założeniu ciągłości rozkładu:

$$\begin{aligned} \mathbb{E}_{\theta_0} \ell(\theta) - \mathbb{E}_{\theta_0} \ell(\theta_0) &= \mathbb{E}_{\theta_0} (\ell(\theta) - \ell(\theta_0)) = \mathbb{E}_{\theta_0} (\log f(\theta; X) - \log f(\theta_0; X)) \\ &= \mathbb{E}_{\theta_0} \log \frac{f(\theta; X)}{f(\theta_0; X)}, \end{aligned}$$

i pamiętając o tym, że $\log t \leq t - 1$, można dojść do

$$\begin{aligned} \mathbb{E}_{\theta_0} \log \frac{f(\theta; X)}{f(\theta_0; X)} &\leq \mathbb{E}_{\theta_0} \left(\frac{f(\theta; X)}{f(\theta_0; X)} - 1 \right) = \int_{\mathbb{R}} \left(\frac{f(\theta; x)}{f(\theta_0; x)} - 1 \right) f(\theta_0; x) dx \\ &= \int_{\mathbb{R}} f(\theta; x) dx - \int_{\mathbb{R}} f(\theta_0; x) dx = 1 - 1 = 0. \end{aligned}$$

Obie całki równeją się 1 jako, że są całkami z funkcji gęstości, zaś równość w nierówności zachodzi tylko wtedy, gdy $\mathbb{P}_\theta = \mathbb{P}_{\theta_0}$. ■

Dzięki temu wynikowi możliwe jest udowodnienie poniższego Twierdzenia (1.2).

Twierdzenie 1.2. *Pod pewnymi warunkami regularności nałożonymi na rodzinę rozkładów prawdopodobieństwa, estymator największej wiarogodności $ENW(\theta)$ jest zgodny, tzn.*

$$ENW(\theta) \rightarrow \theta \quad \text{dla } n \rightarrow \infty.$$

Dowód.

1) Z definicji w $ENW(\theta)$ przyjmowana jest wartość największa funkcji $L(\theta)$, a więc tym bardziej funkcji $\ell(\theta) = \log L(\theta)$ oraz funkcji

$$\ell_n(\theta) = \frac{1}{n} \ell(\theta) = \frac{1}{n} \log L(\theta) = \frac{1}{n} \sum_{i=1}^n \log f(\theta; x_i)$$

(zakładając ciągłość rozkładu i niezależność X_1, \dots, X_n), gdyż ekstremum jest niezmiennicze ze względu na monotoniczną transformację i liniowe przekształcenie, jakim jest dzielenie.

2) Z Lematu 1.1 wynika, że θ_0 maksymalizuje $\mathbb{E}_{\theta_0} \ell(\theta)$.

3) Z Prawa Wielkich Liczb, które jest spełnione gdy założy się, że x_i to realizacje ciągu zmiennych losowych o skończonych wartościach oczekiwanych, wynika że

$$\ell_n(\theta) = \frac{1}{n} \sum_{i=1}^n \log f(\theta; x_i) \rightarrow \mathbb{E}_{\theta_0} \ell(\theta),$$

co ostatecznie oznacza, że $ENW(\theta)$ jest zgodny. ■

Asymptotyczna normalność estymatora największej wiarygodności

Fisher w swojej karierze wprowadził wiele pozytycznych pojęć stosowanych do dziś. Jednym z nich jest Informacja Fishera, która zostanie wykorzystana w dowodzie asymptotycznej normalności estymatora największej wiarogodności.

Definicja 1.10. Niech X będzie zmienną losową o gęstości f_θ , zależną od jednowymiarowego parametru $\theta \in \Theta \subset \mathbb{R}$. **Informacją Fishera** zawartą w obserwacji X nazywa się funkcję

$$\mathcal{I}(\theta) = \mathbb{E}_\theta(\ell'(\theta; X))^2 = \mathbb{E}_\theta\left(\frac{\partial}{\partial\theta} \log f_\theta(X)\right)^2, \quad (1.1)$$

gdzie odpowiednio

$$\begin{aligned} \mathcal{I}(\theta) &= \int \left(\frac{\partial}{\partial\theta} \log f_\theta(x)\right)^2 f_\theta(x) dx \quad \text{dla zmiennej ciągkiej;} \\ \mathcal{I}(\theta) &= \sum_x \left(\frac{\partial}{\partial\theta} \log f_\theta(x)\right)^2 \mathbb{P}_\theta(X = x) \quad \text{dla zmiennej dyskretnej.} \end{aligned}$$

W dowodzie asymptotycznej normalności estymatora największej wiarogodności kluczowymi założeniami są poniższe warunki regularności. Rodzina gęstości musi być dostatecznie regularna aby pewne kroki rachunkowe w dalszych rozumowaniach były poprawne.

Definicja 1.11. Warunki regularności.

- i) Informacja Fishera jest dobrze określona. Zakłada się, że Θ jest przedziałem otwartym, istnieje pochodna $\frac{\partial}{\partial\theta} \log f_\theta$, całka/suma we wzorze (1.1) jest bezwzględnie zbieżna (po obłożeniu funkcji podcalkowej modułem całka istnieje i jest skończona) i $0 < \mathcal{I}(\theta) < \infty$.
- ii) Wszystkie gęstości f_θ mają jeden nośnik, tzn. zbiór $\{x \in X : f_\theta(x) > 0\}$ nie zależy od θ .
- iii) Można przenosić pochodną przed znak całki, czyli zamienić kolejność operacji różniczkowania $\frac{\partial}{\partial\theta}$ i całkowania $\int \dots dx$.

Wprowadzając takie założenia, otrzymano przydatne właściwości Informacji Fishera.

Stwierdzenie 1.3. Jeśli spełnione są warunki regularności (Definicja 1.11) to:

- (i) $\mathbb{E}_\theta \frac{\partial}{\partial\theta} \log f_\theta(X) = 0$,
- (ii) $\mathcal{I}(\theta) = \text{Var}_\theta\left(\frac{\partial}{\partial\theta} \log f_\theta(X)\right)$,
- (iii) $\mathcal{I}(\theta) = -\mathbb{E}_\theta\left(\frac{\partial^2}{\partial\theta^2} \log f_\theta(X)\right)$.

Dowód tego stwierdzenia można znaleźć w [Niemiro \(2011\)](#).

Patrząc na postać pochodnej funkcji log-wiarogodności

$$\ell'(\theta_0; x) = (\log f(\theta_0; x))' = \frac{f'(\theta_0; x)}{f(\theta_0; x)},$$

można wywnioskować, że nieformalnie interpretacja Informacji Fishera jest miarą tego, jak szybko zmieni się funkcja gęstości, jeśli delikatnie zmieni się parametr θ w okolicach θ_0 . Biorąc kwadrat i wartość oczekiwana, czyli uśredniając po X , otrzymuje się uśrednioną wersję tej miary. Jeżeli Informacja Fishera jest duża, oznacza to, że gęstość zmieni się szybko, gdyby poruszyć parametr θ_0 , innymi słowy - gęstość z parametrem θ_0 może zostać łatwo odróżniona od gęstości z parametrami nie tak bliskimi θ_0 . Stąd wiemy, że możliwa estymacja θ_0 oparta o takie dane jest dobra. Z drugiej strony, jeżeli Informacja Fishera jest mała, oznacza to, że gęstość dla θ_0 jest bardzo podobna do gęstości z parametrami nie tak bliskimi do θ_0 , a co za tym idzie, dużo ciężej będzie odróżnić tę gęstość, czyli estymacja będzie słabsza.

Informacja Fishera i warunki regularności umożliwiają udowodnienie Twierdzenia 1.4, z którego wynika że im większa Informacja Fishera tym mniejsza asymptotyczna wariancja estymatora prawdziwego parametru θ_0 .

Twierdzenie 1.4. *Pod pewnymi warunkami regularności nałożonymi na rodzinę rozkładów prawdopodobieństwa, estymator największej wiarogodności jest asymptotycznie normalny,*

$$\sqrt{n}(ENW(\theta) - \theta_0) \xrightarrow{d} \mathcal{N}\left(0, \frac{1}{\mathcal{I}(\theta_0)}\right).$$

Dowód. Ponieważ $ENW(\theta)$ maksymalizuje $\ell_n(\theta) = \frac{1}{n} \sum_{i=1}^n \log f(\theta; X)$, to $\ell'_n(\theta) = 0$.

Dalej, korzystając z Twierdzenia o Wartości Średniej:

$$\frac{g(a) - g(b)}{a - b} = g'(c) \text{ albo } g(a) = g(b) + g'(c)(a - b), \text{ dla pewnego } c \in [a, b],$$

gdzie $g(\theta) = \ell'_n(\theta)$, $a = ENW(\theta)$, $b = \theta_0$, można zapisać równość

$$0 = \ell'_n(ENW(\theta)) = \ell'_n(\theta_0) + \ell''_n(\theta_1)(ENW(\theta) - \theta_0), \text{ dla } \theta_1 \in [ENW(\theta), \theta_0],$$

a z niej przejść do postaci

$$\sqrt{n}(ENW(\theta) - \theta_0) = -\frac{\sqrt{n}\ell'_n(\theta_0)}{\ell''_n(\theta_1)}. \quad (1.2)$$

Z Lematu (1.1) wynika, że θ_0 maksymalizuje $\mathbb{E}_{\theta_0} \ell(\theta_0)$ czyli

$$\mathbb{E}_{\theta_0} \ell'(\theta_0) = 0, \quad (1.3)$$

a to można wstawić do licznika w równaniu (1.2)

$$\begin{aligned} \sqrt{n}\ell'_n(\theta_0) &= \sqrt{n} \left(\frac{1}{n} \sum_{i=1}^n \ell'(\theta_0) - 0 \right) \\ &= \sqrt{n} \left(\frac{1}{n} \sum_{i=1}^n \ell'(\theta_0) - \mathbb{E}_{\theta_0} \ell'(\theta_0) \right) \rightarrow \mathcal{N}\left(0, \text{Var}_{\theta_0}(\ell'(\theta_0))\right), \end{aligned} \quad (1.4)$$

gdzie zbieżność wynika z Centralnego Twierdzenia Granicznego.

Następnie można rozważyć mianownik w równaniu (1.2). Dla wszystkich θ wynika

$$\ell''(\theta) = \frac{1}{n} \sum_{i=1}^n \ell''(\theta) \rightarrow \mathbb{E}_{\theta_0} \ell''(\theta)$$

z Prawa Wielkich Liczb.

Dodatkowo, ponieważ $\theta_1 \in [ENW(\theta), \theta_0]$ a $ENW(\theta)$ jest zgodny (poprzedni podrozdział), to ponieważ $ENW(\theta) \rightarrow \theta_0$, to też $\theta_1 \rightarrow \theta_0$, a wtedy

$$\ell''_n(\theta_1) \rightarrow \mathbb{E}_{\theta_0} \ell''(\theta_0) = -\mathcal{I}(\theta_0)$$

z punktu (iii) ze Stwierdzenia (1.3).

Wtedy prawa strona równania (1.2), dzięki (1.4)

$$-\frac{\sqrt{n}\ell'_n(\theta_0)}{\ell''_n(\theta_1)} \xrightarrow{d} \mathcal{N}\left(0, \frac{\mathbb{V}ar_{\theta_0}(\ell'(\theta_0))}{(\mathcal{I}(\theta_0))^2}\right).$$

Ostatecznie wariancja

$$\mathbb{V}ar_{\theta_0}(\ell'(\theta_0)) = \mathbb{E}_{\theta_0}(\ell'(\theta_0))^2 - (\mathbb{E}_{\theta_0} \ell'(\theta_0))^2 = \mathcal{I}(\theta_0) - 0,$$

co wynika z definicji Informacji Fishera i wzoru (1.3).

■

Rozdział 2

Model Coxa

W tym rozdziale zostanie przedstawiony model proporcjonalnych hazardów Cox'a. Głównym celem pracy jest wykorzystanie numerycznej metody do estymacji współczynników w rozważanym modelu. Więcej o estymacji metodą stochastycznego spadku gradientu napisane jest w rozdziale 3.2. Wykorzystane definicje i twierdzenia oparte są o [Cox \(1972\)](#), [Therneau and Grambsch \(2000\)](#), [Asselain and Mould \(2010\)](#) i [Burzykowski \(2015\)](#).

2.1. Wprowadzenie do modelu Coxa i terminologia

Analiza przeżycia, w której model Coxa znalazł największe zastosowania, polega na modelowaniu wpływu czynników na czas do wystąpienia pewnego zdarzenia. Zdarzeniem może być np. śmierć pacjenta, awaria urządzenia, zerwanie umowy przez klienta, odejście z pracy pracownika lub deaktywacja pewnej usługi. Analizując czasy do wystąpienia zdarzenia wykorzystuje się funkcję przeżycia bądź niosącą równoważną informację funkcję hazardu.

W poniższych definicjach $T_i^*, i = 1, \dots, n$ to dodatnie, ciągłe zmienne losowe, które odpowiadają czasom do wystąpienia zdarzenia dla obserwacji X_i . Zakłada się, że czasy T_i^* to niezależne zmienne losowe o gęstościach $f_i(t)$.

Definicja 2.1. *Funkcja przeżycia, to funkcja, która spełnia*

$$S_i(t) = \mathbb{P}(T_i^* \geq t) = 1 - F_i(t), t \in \mathbb{R} \quad (2.1)$$

gdzie $F_i(t)$ to dystrybuanta rozkładu zadanego gęstością $f_i(t)$.

Definicja 2.2. *Funkcja hazardu to funkcja, która wyraża się wzorem*

$$\begin{aligned} \lambda_i(t) &= \lim_{h \rightarrow 0} \frac{\mathbb{P}(t \leq T_i^* \leq t + h | T_i^* \geq t)}{h} \\ &= \lim_{h \rightarrow 0} \frac{\mathbb{P}(t \leq T_i^* \leq t + h)}{h} \cdot \frac{1}{\mathbb{P}(T_i^* \geq t)} \\ &= \lim_{h \rightarrow 0} \frac{F_i(t + h) - F_i(t)}{h} \cdot \frac{1}{S_i(t)} \\ &= \frac{f_i(t)}{S_i(t)}. \end{aligned}$$

Wartość funkcji hazardu w momencie t traktuje się jako chwilowy potencjał pojawiającego się zdarzenia (np. śmierci lub choroby), pod warunkiem że osoba dożyła czasu t . Funkcja hazardu nazywana jest również funkcją ryzyka, intensywnością umieralności (*force of mortality*), umieralnością chwilową (*instantaneous death rate*) lub chwilową częstością niepowodzeń (awarii) (*failure rate*). Ostatniego określenia używa się w teorii odnowy (Cox, 1962), w której analizuje się awaryjność elementów przemysłowych.

Model proporcjonalnych hazardów Coxa (Cox, 1972) jest obecnie najczęściej stosowaną procedurą do modelowania relacji pomiędzy zmiennymi objaśniającymi a przeżyciem lub innym cenzurowanym zdarzeniem. Model ten umożliwia analizę wpływu czynników prognostycznych na przeżycie. Sir David Cox opracował model tego typu dla tabeli przeżyć i zilustrował jego zastosowanie w przypadku białaczki, ale może być on stosowany do obliczania przeżyć w odniesieniu do innych chorób, jak w przypadku przeżyć w chorobach nowotworowych lub kardiologicznych po transplantacji serca (Norwegian Multicentre Study Group, 1981).

Definicja 2.3. *Model Coxa* zakłada postać funkcji hazardu dla i -tej obserwacji X_i jako

$$\lambda_i(t) = \lambda_0(t)e^{X_i(t)'\beta}, \quad (2.2)$$

gdzie λ_0 to niesprecyzowana nieujemna funkcja nazywana bazowym hazardem, a β to wektor współczynników rozmiaru p , co odpowiada liczbie zmiennych objaśniających w modelu Coxa.

Takie sformułowanie modelu gwarantuje, że funkcja hazardu jest nieujemna.

Model Coxa, dla zmiennych stałych w czasie, nazywany jest **modelem proporcjonalnych hazardów**, gdyż proporcja hazardów dwóch obserwacji X_i i X_j jest stała w czasie:

$$\frac{\lambda_i(t)}{\lambda_j(t)} = \frac{\lambda_0(t)e^{X_i\beta}}{\lambda_0(t)e^{X_j\beta}} = \frac{e^{X_i\beta}}{e^{X_j\beta}} = e^{(X_i - X_j)\beta}.$$

Oznacza to, że hazard dla jednej obserwacji można uzyskać poprzez przemnożenie hazardu dla innej obserwacji przez pewną stałą c_{ij} :

$$\lambda_i(t) = \frac{e^{X_i'\beta}}{e^{X_j'\beta}} \cdot \lambda_j(t) = c_{ij} \cdot \lambda_j(t).$$

W modelu proporcjonalnych hazardów istotnym elementem jest estymacja stałych c_{ij} .

2.2. Założenia modelu proporcjonalnego ryzyka Coxa

Model Coxa znalazł szerokie zastosowanie w sytuacjach, gdy analiza wymaga wykorzystania cenzurowanych danych. Model Coxa jest w stanie wykorzystać je do estymacji współczynników w modelu, przekładających się na proporcje hazardów. Z uwagi na aspekt praktyczny podyktowany warunkami technicznymi prób klinicznych i badań biologicznych, zbiory danych klinicznych zawierają cenzurowane czasy zdarzeń. Oznacza to, że w wielu przypadkach niemożliwe jest obserwowanie czasu zdarzeń dla wszystkich pacjentów. Niekiedy jest to uwarunkowane zbyt długim terminem do wystąpienia zdarzenia. Nierzadko jest to związane z założonym okresem próby klinicznej, który jest krótszy niż czas do zdarzenia dla pacjentów, którzy mogli zostać włączeni do próby klinicznej pod koniec jej trwania i nie udało się zaobserwować ich zdarzeń. W wielu przypadkach pacjenci, traktowani jako obserwacje w zbiorze,

znikają z pola widzenia w momencie, gdy np. przestają pojawiać się na wizytach kontrolnych. Może być to spowodowane negatywnymi relacjami z lekarzem prowadzącym lub przeprowadzką. W takich sytuacjach wykorzystuje się daną obserwację do momentu jej ostatniej kontroli. Nie rezygnuje się z niej w analizie lecz wykorzystuje się o niej informacje w pełni dla czasu, w którym przebywała pod obserwacją. Jest to ogromna zaleta modelu Coxa.

Z przyczyny cenzurowanych danych potrzebne są założenia modelu dotyczące cenzurowania czasów, które opierają się o następujące definicje.

Definicja 2.4. *Cenzurowanie prawostronne polega na zaobserwowaniu czasu*

$$T = \min(T^*, C),$$

gdzie T^* to prawdziwy czas zdarzenia, zaś C jest nieujemną zmienną losową.

Definicja 2.5. *Cenzurowanie jest niezależne jeśli zachodzi*

$$\lim_{h \rightarrow 0} \frac{\mathbb{P}(t \leq T^* \leq t+h | T^* \geq t)}{h} = \lim_{h \rightarrow 0} \frac{\mathbb{P}(t \leq T^* \leq t+h | T^* \geq t, Y(t) = 1)}{h},$$

gdzie $Y(t) = 1$ jeśli do chwili t nie wystąpiło zdarzenie ani cenzurowanie, czyli jednostka pozostaje narażona na ryzyko zdarzenia oraz $Y(t) = 0$ w przeciwnym wypadku.

Interpretacja tej definicji jest następująca: jednostka cenzurowana w chwili t jest reprezentatywna dla wszystkich innych narażonych na ryzyko zdarzenia w chwili t . Innymi słowy cenzurowanie nie wybiera z populacji osobników bardziej albo mniej narażonych na zdarzenie. Cenzurowanie działa niezależnie od mechanizmu występowania zdarzenia.

Definicja 2.6. *Cenzurowanie jest nie-informatywne jeśli zachodzi*

$$g(t; \theta, \phi) \equiv g(t; \phi), \quad (2.3)$$

gdzie $g(t; \theta, \phi)$ jest funkcją gęstości dla cenzurowań C_i wyrażonych jako niezależne zmienne losowe o jednakowym rozkładzie, zaś prawdzie czasy T_i^* są interpretowane jako niezależne zmienne losowe o gęstościach $f_i(t; \theta)$, czyli θ parametryzuje jedynie rozkłady czasów zdarzeń.

Oznacza to, że cenzurowanie nie daje informacji o parametrach rozkładu czasów zdarzeń, ponieważ nie zależy od parametrów od których zależny jest hazard.

Model proporcjonalnych hazardów Coxa oparty jest na założeniach:

- i) Współczynniki modelu $\beta_k, k = 1, \dots, p$ są stałe w czasie, co przekłada się na to, że stosunek hazardów dla dwóch obserwacji jest stały w czasie.
- ii) Postać funkcyjonalna efektu zmiennej niezależnej - postać modelu $\lambda_i(t) = \lambda_0(t)e^{X_i(t)'\beta}$.
- iii) Obserwacje są niezależne.
- iv) Cenzurowanie czasów jest nie-informatywne.
- v) Cenzurowanie czasów jest niezależne (od mechanizmu występowania zdarzenia).

2.3. Estymacja w modelu Coxa

Funkcja hazardu jest wykładniczą funkcją zmiennych objaśniających, nieznana jest natomiast postać bazowej funkcji hazardu (Definicja 2.3), co bez dalszych założeń uniemożliwia estymację standardową metodą największej wiarygodności. Rozwiązaniem Cox'a jest maksymalizacja tylko tego fragmentu funkcji wiarygodności, który zależy jedynie od estymowanych parametrów. W modelu Coxa proporcjonalnych hazardów estymacja współczynników β oparta jest o częściową funkcję wiarogodności.

Dla konkretnego czasu zdarzenia t_i , gdzie w zbiorze obserwowanych jest K czasów zdarzeń, prawdopodobieństwo warunkowe ze względu na licznosć zbioru ryzyka w czasie t_i , że czas zdarzenia dotyczy i -tej jednostki spośród wciąż obserwowanych jest równe

$$\frac{e^{X'_i \beta}}{\sum_{l \in \mathcal{R}(t_i)} e^{X'_l \beta}}, \quad (2.4)$$

gdzie *zbior ryzyka* $\mathcal{R}(t_i)$, w chwili t_i , rozumiany jest jako zbiór indeksów obserwacji, które są w danym czasie t_i pod obserwacją.

Chcąc estymować współczynniki metodą największej wiarogodności należy rozważyć funkcję wiarogodności, która dla niezależnego cenzurowania prawostronnego ma postać:

$$L(\beta, \varphi) = {}_p L(\beta) \cdot L^*(\beta, \varphi), \quad (2.5)$$

gdzie, dla $\lambda(t)$, $f(t)$, $S(t)$ wprowadzonych w Definicjach 2.2 i 2.1

$${}_p L(\beta) = \prod_{i=1}^n f(t_i; \beta)^{\delta_i} S(t_i; \beta)^{1-\delta_i} = \prod_{i=1}^n \lambda(t_i; \beta)^{\delta_i} S(t_i; \beta) \quad (2.6)$$

to częściowa funkcja wiarogodności, a $L^*(\beta, \varphi)$ zależy od cenzurowania (parametr φ).

Wtedy dla niezależnego cenzurowania i dla czasów zdarzeń, które nie zaszły jednocześnie **częściowa funkcja wiarogodności w modelu Coxa** ma postać:

$${}_p L(\beta) = \prod_{i=1}^K \frac{e^{X'_i \beta}}{\sum_{l=1}^n Y_l(t_i) e^{X'_l \beta}}, \quad (2.7)$$

gdzie $Y_l(t_i) = 1$, gdy obserwacja X_l jest w zbiorze ryzyka w czasie t_i , i $Y_l(t_i) = 0$ w przeciwnym przypadku, n to liczba obserwacji w zbiorze, a K to wspomniana wyżej liczba zaobserwowań czasów zdarzeń. Zaletą takiej postaci funkcji częściowej wiarogodności jest to, że w jej wzorze nie występuje funkcja bazowego hazardu, zatem estymacja współczynników może odbywać się bez znajomości jej postaci.

Jeśli dodatkowo cenzurowanie jest nie-informatywne, to ${}_p L(\beta)$ jest pełną funkcją wiarogodności, bowiem wówczas

$$L^*(\beta, \varphi) \propto L^*(\varphi)$$

co bierze się z definicji cenzurowania nie-informatycznego (2.3)

$$g(t; \theta, \phi) \equiv g(t; \phi).$$

Ponieważ model proporcjonalnych hazardów Coxa zakłada niezależność i nie-informatywność cenzurowania, można uważać, że częściowa funkcja wiarogodności daje pełną informację o współczynnikach. Zatem wnioskowanie w oparciu o nią jest uzasadnione i poprawne.

W sytuacjach, gdy nie jest spełnione założenie nie-informatywności cenzurowania i częściowa funkcja wiarodnościi nie jest funkcją wiarodności w sensie bycia proporcjonalną do prawdopodobieństwa obserwowanego zbioru, można ją traktować jako funkcję wiarodności w celu asymptotycznego wnioskowania (Therneau and Grambsch, 2000).

Analityczna estymacja współczynników

Standardowo w celu znalezienia maximum, aby ułatwić obliczenia, można rozważaną funkcję obłożyć monotoniczną transformacją, np. logarytmem, tak aby w konsekwencji otrzymać **częściową funkcję log-wiarodności**

$$p\ell(\beta) = \sum_{i=1}^K X'_i \beta - \sum_{i=1}^K \log \left(\sum_{l \in \mathcal{R}(t_i)} e^{X'_l \beta} \right). \quad (2.8)$$

Analityczne obliczenia dają p -wymiarowy wektor pochodnych, dla $k = 1, \dots, p$

$$U_k(\beta) = \frac{\partial_p \ell_k(\beta)}{\partial \beta_k} = \sum_{i=1}^K (X_{ik} - A_{ik}), \quad (2.9)$$

X_{ik} to i ta obserwacja i k ta zmien- na.

gdzie czynnik

$$A_{ik} = \frac{\sum_{l \in \mathcal{R}(t_i)} X_{lk} e^{X'_l \beta}}{\sum_{l \in \mathcal{R}(t_i)} e^{X'_l \beta}} \quad (2.10)$$

to średnia z $X_{.k}$ (k -tych zmiennych) po skończonej populacji $\mathcal{R}(t_i)$, z wykorzystaniem *ważonej eksponencjalnie* formy próbkowania.

Z kolei drugie pochodne cząstkowe (Cox, 1972) mają postać dla $k_1, k_2 = 1, \dots, p$

$$\mathcal{I}_{k_1 k_2}(\beta) = -\frac{\partial^2 p L(\beta)}{\partial \beta_{k_1} \partial \beta_{k_2}} = \sum_{i=1}^K C_{ik_1 k_2}(\beta), \quad (2.11)$$

gdzie

$$C_{ik_1 k_2}(\beta) = \frac{\sum_{l \in \mathcal{R}(t_i)} X_{lk_1} X_{lk_2} e^{X'_l \beta}}{\sum_{l \in \mathcal{R}(t_i)} e^{X'_l \beta}} - A_{ik_1}(\beta) A_{ik_2}(\beta) \quad (2.12)$$

to kowariancja pomiędzy $X_{.k_1}$ (k_1 -tymi zmiennymi) a $X_{.k_2}$ (k_2 -tymi zmiennymi) przy tej formie ważonego próbkowania.

Estymator największej wiarodności β można uzyskać poprzez przyrównanie (2.9) do 0, a numerycznie poprzez iteracyjne wykorzystanie (2.9) oraz (2.11) w algorytmie spadku gradientu rzędu II nazywanego również algorytmem Raphsona-Newtona, który jest opisany w podrozdziale 3.1. Jest to tradycyjne i szeroko stosowane podejście do estymacji współczynników w modelu proporcjonalnych hazardów Coxa. Niniejsza praca skupia się na wykorzystaniu metody estymacji współczynników jaką jest metoda stochastycznego spadku wzduż gradientu, która jest szerzej opisana w następującym rozdziale 3.

2.4. Generowanie danych dla modelu Coxa

Poniższy podrozdział przedstawia metodę odwrotnych prawdopodobieństw opisaną szerzej w Bender et al. (2005), dzięki której można wygenerować czasy zdarzeń dla zadanej z góry funkcji hazardu i zmiennych objaśniających. Ta metoda posłuży w rozdziale 4 do wygenerowania danych w celu weryfikacji jakości procesu numerycznej estymacji współczynników modelu, w sytuacji gdy wykorzystywany jest algorytm stochastycznego spadku gradientu.

Mówiąc o funkcji przeżycia warto wprowadzić skumulowaną funkcję hazardu.

Definicja 2.7. Skumulowaną funkcję hazardu nazywa się funkcję spełniającą zależność

$$H(t) = \int_0^t \lambda(u) du, \quad (2.13)$$

gdzie $\lambda(u)$ to pewna funkcja hazardu, o której mówi Definicja (2.2).

Wtedy, dla zdefiniowanej w Definicji 2.2 funkcji hazardu, funkcja przeżycia i jej dopełnienie (dystrybuanta) dla modelu Coxa proporcjonalnych hazardów wygląda następująco

$$S(t|x) = e^{-H_0(t) \cdot e^{x' \beta}}, \quad (2.14)$$

$$F(t|x) = 1 - e^{-H_0(t) \cdot e^{x' \beta}}, \quad (2.15)$$

gdzie $H_0(t)$ to bazowa skumulowana funkcja hazardu, czyli $H_0(t) = \int_0^t \lambda_0(u) du$.

Niech Y będzie zmienną losową o dystrybuancie zadanej w (2.15), wtedy zmienne losowe $U = F(Y)$, $1 - U$ mają rozkład jednostajny $U \sim \mathcal{U}([0, 1])$. Dodatkowo, niech T będzie czasem przeżycia w modelu Coxa, wtedy z (2.15) wynika

$$U = e^{-H_0(T) \cdot e^{x' \beta}} \sim \mathcal{U}([0, 1]). \quad (2.16)$$

Jeżeli $\lambda_0(t) > 0$ dla każdego t , to $H_0(t)$ jest niemalejąca i ma poprawnie zdefiniowaną uogólnioną odwrotność $H_0^{-1}(t)$, zaś czas przeżycia T dla modelu Coxa może być wyrażony przez

$$T = H_0^{-1}(-\log(U) \cdot e^{-x' \beta}), \quad (2.17)$$

gdzie $U \sim \mathcal{U}([0, 1])$.

Przykład symulacji dla rozkładu Weibulla

Równanie (2.17) jest odpowiednie do generowania czasów do zdarzenia w modelu Coxa, gdy potrafi się odpowiednio generować zmienne z rozkładu $\mathcal{U}([0, 1])$. Jest to możliwe w większości pakietów statystycznych. Poniżej przedstawiony jest kod w języku R (R Core Team, 2013), dzięki któremu możliwe jest generowanie czasów zdarzeń pochodzących z rozkładu Weibulla (Collett, 1994), dla którego funkcja bazowego hazardu ma postać

$$\lambda_0(t) = \lambda \rho t^{\rho-1}, \quad (2.18)$$

gdzie $\lambda > 0$ to parametr skali, zaś $\rho > 0$ to parametr kształtu.

Z (2.13) wynika, że bazowa skumulowana funkcja hazardu dla rozkładu Weibulla wynosi

$$H_0(t) = \int_0^t \lambda \rho u^{\rho-1} du = \lambda t^\rho, \quad (2.19)$$

zaś jej funkcja przeciwna to

$$H_0^{-1}(t) = (\lambda^{-1} t)^{\frac{1}{\rho}}. \quad (2.20)$$

Wtedy podstawiając (2.20) do (2.17) można otrzymać

$$T = (\lambda^{-1} \cdot (-\log(U)) \cdot e^{-x'\beta})^{\frac{1}{\rho}} = \left(-\frac{\log(U)}{\lambda e^{x'\beta}} \right)^{\frac{1}{\rho}}. \quad (2.21)$$

Dzięki tym rozważaniom, możliwe było stworzenie kodu generującego czasy i indykatory zdarzeń dla zadanych z góry współczynników modelu i zmiennych objaśniających. Poniższy kod i jego rezultat posłużą w rozdziale 4.2 do diagnostyki procesu estymacji w modelu Coxa w przypadku wykorzystania algorytmu stochastycznego spadku gradientu.

```
dataCox <- function(N, lambda, rho, x, beta, censRate){

  # real Weibull times
  u <- runif(N)
  Treal <- (- log(u) / (lambda * exp(x %*% beta)))^(1 / rho)

  # censoring times
  Censoring <- rexp(N, censRate)

  # follow-up times and event indicators
  time <- pmin(Treal, Censoring)
  status <- as.numeric(Treal <= Censoring)

  # data set
  data.frame(id=1:N, time=time, status=status, x=x)
}

x <- matrix(sample(0:1, size = 40, replace = TRUE), ncol = 2)

head(dataCox(20, 3, 2, x, beta = c(2,3), 5))

  id      time status x.1 x.2
1 1 0.01193626     0   0   1
2 2 0.03567485     0   1   1
3 3 0.13330012     1   0   1
4 4 0.04358821     1   1   1
5 5 0.03825366     1   1   1
6 6 0.29355955     1   1   0
```


Rozdział 3

Numeryczne metody estymacji

Odkąd zjawiska przyrodnicze zaczęto opisywać przy użyciu formalizmu matematycznego, pojawiła się potrzeba rozwiązywania zadań analizy matematycznej czy algebry. Dopóki były one nieskomplikowane, dawały się rozwiązywać analitycznie, tzn. z użyciem pewnych przekształceń algebraicznych prowadzących do otrzymywania rozwiązań. Z czasem jednak, przy powstawaniu coraz to bardziej skomplikowanych teorii opisujących zjawiska, problemy te stały się na tyle złożone, iż ich rozwiązywanie ścisłe było albo bardzo czasochłonne albo też zgoła niemożliwe. Przez numerykę rozumie się dziedzinę matematyki zajmującą się przybliżonym rozwiązywaniem zagadnień algebraicznych. Pozwalała ona znajdować ich przybliżone rozwiązania z żądaną dokładnością. Podstawową zaletą wykorzystywania przybliżonych rozwiązań była ogólność formułowanych algorytmów, tzn. w ramach danego zagadnienia nie miało znaczenia czy było ono proste czy też bardzo skomplikowane (najwyżej wiązało się z większym nakładem pracy obliczeniowej), natomiast wadą była czasochłonność. Dlatego prawdziwy renesans metod numerycznych nastąpił wraz z powszechnym użyciem w pracy naukowej maszyn cyfrowych oraz komputerów ([Milewski, 2006](#)).

Dziś dziesiątki żmudnych dla człowieka operacji arytmetycznych wykonuje komputer, jednak złożoność obliczeniowa algorytmów uczenia maszynowego stała się krytycznym czynnikiem ograniczającym pracę w sytuacjach, gdy rozważane są duże zbiory danych. Te ograniczenia spowodowały, że w modelowaniu statystycznym wielkiej skali zaczęto wykorzystywać algorytmy **stochastycznego spadku gradientu**.

W poniższym rozdziale przedstawione są klasyczne algorytmy spadku wzduż gradientu Cauchy'ego oraz Raphsona-Newtona. Następnie omówiony jest algorytm stochastycznego spadku wzduż gradientu, którego zastosowanie do estymacji współczynników w modelu Coxa jest kluczowym celem pracy. Algorytm stochastycznego spadku gradientu to metoda optymalizacji używana w sytuacjach, gdy rozważaną funkcję można zapisać jako sumę różniczkowalnych składników. Ponieważ popularne metody statystycznej estymacji, takie jak *Fisher scoring*, algorytm *Expectation–maximization* czy iteracyjna ważona metoda najmniejszych kwadratów ([Dempster et al., 1977](#); [Fisher, 1925](#); [Green, 1984](#)) nie zawsze przenoszą się na zastosowania do danych dużej skali bądź danych napływających (*ang. streaming data*), niekiedy algorytm stochastycznego spadku gradientu jest jedyną dostępną metodą optymalizacji numerycznej. Ponadto przedstawiono również zalety algorytmów stochastycznego spadku gradientu, które przemawiają za atrakcyjnością i popularnością tego typu rozwiązania. Omawiane pojęcia oparto o [Bottou \(2010, 2012\)](#), [Kotłowski \(2012\)](#) oraz [Fortuna et al. \(2006\)](#).

3.1. Algorytmy spadku wzdłuż gradientu

Poniższy rozdział przedstawia popularne iteracyjne algorytmy wyznaczania przybliżonej wartości miejsca zerowego funkcji oraz rozważaną w pracy metodę stochastycznego spadku gradientu. Szukanie miejsc zerowych funkcji jest przydatne w problemach optymalizacyjnych, gdy celem jest znalezienie pierwiastka pochodnych badanej funkcji. Dodatkowo takie algorytmy wykorzystywane są do rozwiązywania (nielinijowych) układów równań. Metody iteracyjne składają się zazwyczaj z k kroków bądź są zatrzymywane, gdy osiągnięty zostanie warunek stopu, czyli gdy odległość pomiędzy kolejnymi przybliżeniami jest dość mała $\|w_{k+1} - w_k\| < \varepsilon$ lub wartość gradientu funkcji w wyznaczonym punkcie jest bliska wektorowymi zerowemu $\|\nabla_Q(\mathbf{w}_k)\| \leq \varepsilon$ (test stacjonarności), gdzie ε to zadana z góry precyzja. Metoda stochastycznego spadku wzdłuż gradientu zakłada, że minimalizowaną funkcję $Q(w)$ można przedstawić jako różniczkowalną sumę jej składników $Q(w) = \sum_{i=1}^n Q_i(w)$. W poniższych algorytmach α_k oznacza długość kroku algorytmu.

Metoda spadku wzdłuż gradientu I (Cauchy'ego)

Minimalizacja funkcji $Q(w)$:

- Zaczynamy od wybranego rozwiązania startowego, np. $w_0 = 0$.
- Dla $k = 1, 2, \dots$ aż do zbieżności
 - Wyznaczamy gradient w punkcie $w_{k-1}, \nabla_Q(w_{k-1})$.
 - Robimy krok wzdłuż negatywnego gradientu:

$$w_k = w_{k-1} - \alpha_k \nabla_Q(w_{k-1}).$$

Metoda spadku wzdłuż gradientu II (Newtona-Raphsona)

Minimalizacja funkcji $Q(w)$:

- Zaczynamy od wybranego rozwiązania startowego, np. $w_0 = 0$.
- Dla $k = 1, 2, \dots$ aż do zbieżności
 - Wyznaczamy gradient w punkcie $w_{k-1}, \nabla_Q(w_{k-1})$ i odwrotność $(D_Q^2(w_{k-1}))^{-1}$.
 - Robimy krok wzdłuż negatywnego gradientu z zadanym krokiem przez Hesjan:

$$w_k = w_{k-1} - (D_Q^2(w_{k-1}))^{-1} \nabla_Q(w_{k-1}). \quad (3.1)$$

Metoda stochastycznego spadku wzdłuż gradientu I

Minimalizacja funkcji $Q(w)$:

- Zaczynamy od wybranego rozwiązania startowego, np. $w_0 = 0$.
- Dla $k = 1, 2, \dots$ aż do zbieżności
 - Wylosuj $i \in \{1, \dots, n\}$
 - Wyznaczamy gradient funkcji Q_i w punkcie $w_{k-1}, \nabla_{Q_i}(w_{k-1})$.
 - Robimy krok wzdłuż negatywnego gradientu:

$$w_k = w_{k-1} - \alpha_k \nabla_{Q_i}(w_{k-1}). \quad (3.2)$$

3.2. Algorytm stochastycznego spadku wzdłuż gradientu I

Stochastyczny spadek gradientu jest często stosowany do estymacji współczynników w szeregowej gamie modeli uczenia maszynowego, takich jak maszyny wektorów podpierających (*ang. Support Vector Machines*), regresja logistyczna czy modele graficzne (Finkel et al., 2008). W połączeniu z algorytmem propagacji wstecznej jest standardową metodą w procesie tworzenia sztucznych sieci neuronowych. Algorytm stochastycznego spadku gradientu był używany już od 1960 przy estymacji współczynników w modelu regresji liniowej (Widrow, 1960) oraz był wykorzystywany w algorytmie filtrów adaptacyjnych najmniejszych średnich kwadratów (Widrow and Stearns, 1985) (*ang. least mean squares (LMS) adaptive filter*).

Idea algorytmu stochastycznego spadku gradientu jest następująca: zamiast obliczać gradient na całej funkcji $Q(w)$, w danym kroku oblicz gradient tylko na pojedynczym elemencie $Q_i(w)$. Nazwa *stochastyczny* bierze się stąd, iż oryginalnie wybiera się $Q_i(w)$ losowo. W praktyce zwykle przechodzi się po całym zbiorze danych w losowej kolejności.

Właściwości stochastycznego spadku wzdłuż gradientu

Zbieżność algorytmu stochastycznego spadku gradientu była szeroko badana w literaturze aproksymacji stochastycznych. Aby uzyskać zbieżność zazwyczaj wymaga się, aby ciąg kroków α_k był malejący i spełniał warunki $\sum_k \alpha_k = \infty$ oraz $\sum_k \alpha_k^2 < \infty$ (Bottou (2010)). Twierdzenie Robbinsa-Siegmunda (Robbins and Siegmund, 1971) przy łagodnych warunkach zapewnia zbieżność prawie na pewno (Bottou, 1998), nawet gdy optymalizowana funkcja nie jest wszędzie różniczkowalna.

Predkość zbieżności stochastycznego spadku gradientu jest w rzeczywistości ograniczana przez zgrubną (*ang. noisy*) aproksymację prawdziwego gradientu. Gdy długości kroków algorytmu maleją zbyt wolno, wariancja estymatorów parametrów w_k maleje również wolno. Gdy kroki maleją zbyt szybko, wartości oczekiwane estymatorów w_k potrzebują więcej czasu by osiągnąć optimum (Bottou, 2010). Pod pewnymi warunkami regularności (Mittal and Madigan, 1998), najlepsza predkość zbieżności jest uzyskana dla ciągu kroków $\alpha_k \sim k^{-1}$.

Jak wykazano w Dennis and Schnabel (1983) pod pewnymi odpowiednimi warunkami regularności, gdy zainicjowany współczynnik początkowy w_0 jest wystarczająco blisko optimum i krok algorytmu jest odpowiednio mały, algorytm stochastycznego spadku gradientu osiąga liniową zbieżność. Oznacza to, iż przy spełnieniu założeń metody, odległości pomiędzy kolejnymi przybliżeniami a minimum funkcji \mathbf{w}^* maleją liniowo: $\| \mathbf{w}^* - \mathbf{w}_{k+1} \| \leq c \| \mathbf{w}^* - \mathbf{w}_k \|$. Zbieżność wymaga często przejścia parokrotnie po całym zbiorze danych. Wady i zalety algorytmu wymienione są poniżej. Zalety zdecydowanie przewyższają wady.

Zalety

- **Szybkość kroku:** obliczenie gradientu wymaga wzięcia tylko jednej obserwacji.
- **Skalowalność:** cały zbiór danych nie musi nawet znajdować się w pamięci operacyjnej.
- **Prostota:** gradient funkcji Q_i daje bardzo prosty wzór na modyfikacje wag.

Wady

- **Wolna zbieżność:** czasem gradient stochastyczny do zbieżności wymaga wielu iteracji po zbiorze uczącym.
- **Problem z ustaleniem długości kroku k :** wyznaczenie k przez przeszukiwanie liniowe nie przynosi dobrych rezultatów, ponieważ nie optymalizujemy oryginalnej funkcji Q tylko jej jeden składnik Q_i .

3.3. Porównanie algorytmów spadku wzdłuż gradientu

W niniejszym podrozdziale przedstawiono graficznie różnice w wyborze kolejnych punktów w trakcie optymalizacji między omawianymi w poprzedniej części pracy algorytmami spadku wzdłuż gradientu I (Cauchy'ego), spadku wzdłuż gradientu II (Newtona-Raphsona) oraz stochastycznego spadku wzdłuż gradientu I. W celu zobrazowania przykładu na dwuwykresowym wykresie, postanowiono ograniczyć się do modelu z jedną zmienną wyjaśniającą i wyrazem wolnym. Do przykładu wybrano model regresji logistycznej, z racji na prostotę przedstawienia funkcji log-wiarogodności jako sumy różniczkowalnych składników.

Funkcja log-wiarogodności dla regresji logistycznej ([Czepiel, 2002](#); [Dobson, 2002](#)) to

$$\ell(\beta_1, \beta_2) = \sum_{i=1}^N Q_i(\beta_1, \beta_2), \quad (3.3)$$

$$Q_i(\beta_1, \beta_2) = y_i(\beta_1 + \beta_2 x_i) - \log(1 + \exp(\beta_1 + \beta_2 x_i)). \quad (3.4)$$

Wtedy współrzędne gradientu funkcji wiarogodności to odpowiednio

$$\frac{\partial \ell(\beta)}{\partial \beta_1} = \sum_{i=1}^N (y_i - \pi_i(\beta)), \quad \frac{\partial \ell(\beta)}{\partial \beta_2} = \sum_{i=1}^N x_i(y_i - \pi_i(\beta)), \quad \pi_i(\beta) = \frac{\exp(\beta_1 + \beta_2 x_i)}{1 + \exp(\beta_1 + \beta_2 x_i)},$$

zaś macierz informacji wyraża się jak następuje

$$\mathcal{I}(\beta) = \begin{bmatrix} \sum_{i=1}^N \pi_i(\beta)(1 - \pi_i(\beta)) & \sum_{i=1}^N x_i \pi_i(\beta)(1 - \pi_i(\beta)) \\ \sum_{i=1}^N x_i \pi_i(\beta)(1 - \pi_i(\beta)) & \sum_{i=1}^N x_i^2 \pi_i(\beta)(1 - \pi_i(\beta)) \end{bmatrix}, \quad (3.5)$$

gdzie N to liczba obserwacji.

Aktualizacja kandydata na miejsce zerowe w k -tym kroku algorytmu optymalizacyjnego wśród kolejnych metod omówionych w rozdziale (3.1) wyraża się poniższymi wzorami

Metoda spadku wzdłuż gradientu I (Cauchy'ego)

$$\beta_k = \beta_{k-1} + \alpha_k \cdot \left(\sum_{i=1}^N (y_i - \pi_i(\beta_{k-1})), \sum_{i=1}^N x_i(y_i - \pi_i(\beta_{k-1})) \right).$$

Metoda spadku wzdłuż gradientu II (Newtona-Raphsona)

$$\beta_k = \beta_{k-1} + \mathcal{I}(\beta_{k-1})^{-1} \cdot \left(\sum_{i=1}^N (y_i - \pi_i(\beta_{k-1})), \sum_{i=1}^N x_i(y_i - \pi_i(\beta_{k-1})) \right),$$

gdzie $\mathcal{I}(\beta_{k-1})$ zdefiniowane jest we wzorze (3.5).

Metoda stochastycznego spadku wzdłuż gradientu I

$$\beta_k = \beta_{k-1} + \alpha_k \cdot \left(y_i - \pi_i(\beta_{k-1}), x_i(y_i - \pi_i(\beta_{k-1})) \right),$$

gdzie i to indeks wylosowanej obserwacji w danym kroku.

Ponieważ algorytmy te znajdują minimum funkcji, a docelowo szukane jest maksimum, stąd wykorzystano przeciwnieństwo funkcji log-wiarogodności. Dlatego w wyżej wymienionych wzorach zmieniono znaki przed pochodnymi na przeciwnie.

Symulacje trajektorii zbieżności algorytmów

Poniższymi wywołaniami kodów z pakietu \mathcal{R} można wygenerować 10000 obserwacji z rozkładu jednostajnego i na ich podstawie wygenerować 10000 obserwacji z rozkładu dwupunktowego o takim rozkładzie prawdopodobieństwa sukcesu, by rzeczywiste współczynniki w modelu regresji logistycznej dla tych zmiennych wynosiły odpowiednio: 2 dla wyrazu wolnego oraz 3 dla zmiennej objaśniającej z rozkładu jednostajnego.

```
x <- runif(10000)
z <- 2 + 3*x
pr <- 1/(1+exp(-z))
y <- rbinom(10000, 1, pr)
```

Tak zasymulowane dane wprowadzono do przygotowanej funkcji `logitGD()`, dzięki której można śledzić wartości ekstremum w kolejnych krokach omawianych algorytmów. Kody funkcji `logitGD()` oraz `graphSGD()` tworzącej wykresy porównujące trajektorie zbieżności różnych algorytmów spadku gradientu dostępne są w Dodatku A.4. Wyniki wywołań funkcji `graphSGD()` zostały umieszczone na Rysunkach 3.1 - 3.5.

Na każdym z wykresów na osi OX pokazano wartości współczynnika dla zmiennej objaśniającej w kolejnych krokach poszczególnych algorytmów. Na osi OY zaznaczono współczynnik wyrazu wolnego. Punkt startowy oznaczono na wykresie czarnym trójkątem, a czarnym kwadratem zaznaczono ekstremum funkcji log-wiarogodności, wyliczone dzięki funkcji `glm()` (Hastie and Pregibon, 1992), która do estymacji używa algorytmu iteracyjnej ważonej metody najmniejszych kwadratów. Ta metoda w regresji logistycznej jest równoważna algorytmowi *Fisher's Scoring* (Jennrich and Sampson, 1976; Longford, 1987), który używa obserwowanej macierzy Informacji Fishera (Definicja 1.10) w miejscu Hesjanu w algorytmie Newtona-Raphsona (3.1)). Trajektorie zbieżności do minimum w odrębnych algorytmach zaznaczono oddzielnymi kolorami. Dodatkowo wpisano liczbę kroków wymaganą przez algorytm do zbieżności. Przez **GDI** oznaczono algorytmu spadku gradientu rzędu I, zaś przez **GDII** rzędu II. Oznaczenie **SGD.i** odpowiada 3 algorytmom stochastycznego spadku gradientu o różnych ciągach odpowiedzialnych za długość kroku. Indeks i odpowiada ciągowi wybranemu do wyznaczania długości kroku algorytmu na zasadzie $\alpha_{k,i} = i/\sqrt{k}$.

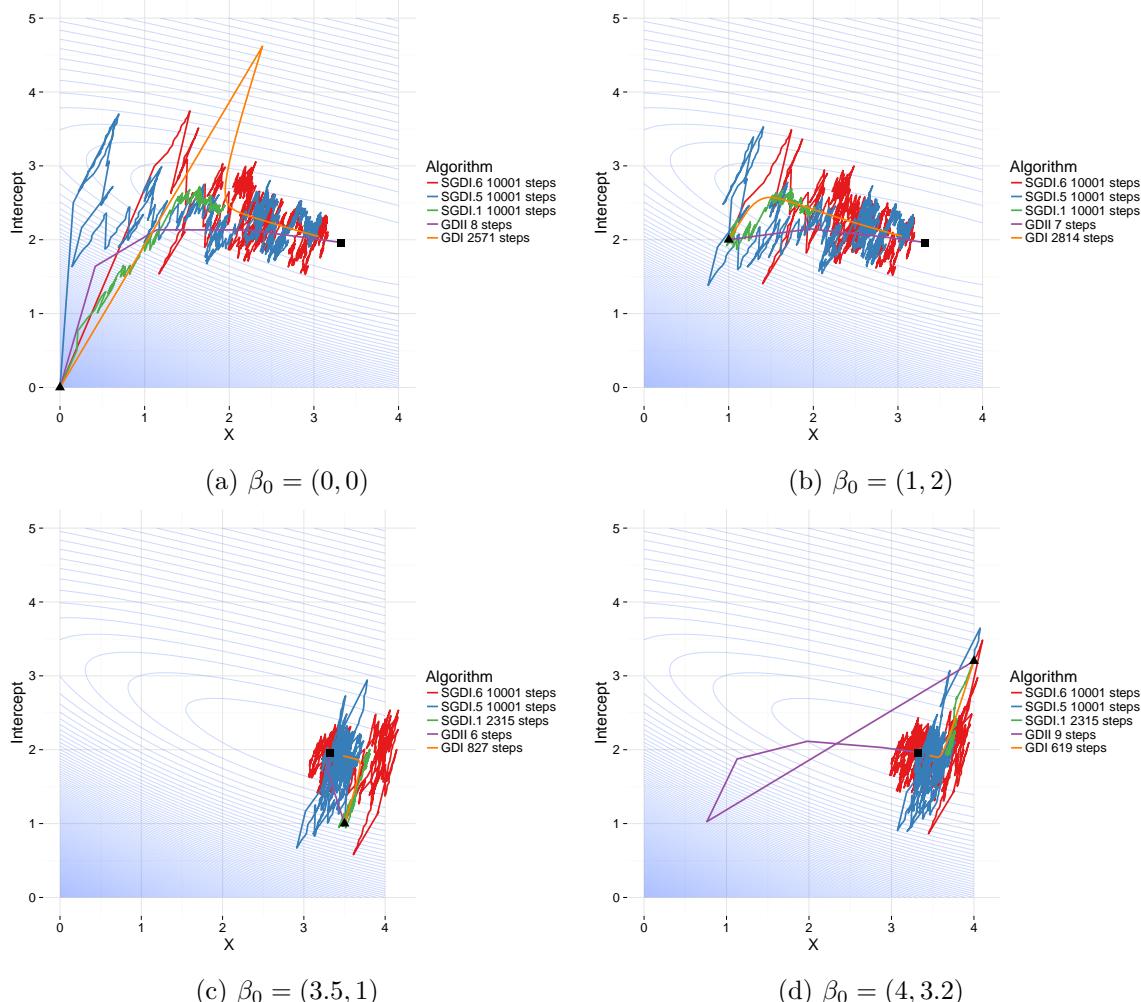
Warunki zbieżności

W trakcie każdej symulacji postawiono pewne warunki konieczne do zbieżności. Ustalono maksymalną liczbę iteracji na 10001, gdzie przez pierwszy krok rozumiano start z punktu startowego. Dodatkowo warunek stopu ustalono na $\varepsilon = 10^{-5}$ oraz ustalono ciąg odpowiadający długościom kroków w algorytmie spadku gradientu rzędu I **GDI** na $\alpha_k = 1/1000\sqrt{k}$. Symulacje powtórzono czterokrotnie, biorąc różne punkty startu algorytmów

$$\beta_0 = (0, 0), (1, 2), (3.5, 1), (4, 3.2).$$

Ponieważ każda trajektoria w procesie estymacji metodą stochastycznego gradientu przy tych samych parametrów zbieżności jest inna, z racji na losową kolejność obserwacji, toteż dla każdego z czterech ustalonych punktów startowych zdecydowano się zobrazować symulację dwukrotnie, aby móc przedstawić stochastyczny aspekt tej metody. Symulacja nr 1 bazowała na losowym wzięciu punktów do algorytmów stochastycznego spadku gradientu, gdy ziarno losowania było ustawione na 4561, zaś w symulacji nr 2 ziarno ustawiono na 456.

Ponieważ na Rysunkach 3.2 - 3.5 każda z symulacji ma inny zakres osi na wykresie, postanowiono na Rysunku 3.1 przygotować zestawienie podobnych symulacji przedstawiając trajektorie zbieżności na wspólnym zakresie osi. Miało to na celu uwypuklić skalę różnic w długościach kroków algorytmów w zależności od odległości punktu startowego od rzeczywistego ekstremum. Dodatkowo na osiach zaznaczono warstwice funkcji log-wiarogodności w modelu regresji logistycznej (równanie (3.3)).



Rysunek 3.1: Porównanie algorytmów spadku gradientu w modelu regresji logistycznej. Wykresy przedstawiają ścieżki zbieżności w kolejnych krokach algorytmów spadku gradientu. Ustalono maksymalną liczbę iteracji na 10001, zaś warunek stopu ustalono na $\varepsilon = 10^{-5}$. Trójkątem zaznaczono punkt startowy, a kwadratem wyestymowane rozwiązanie przy pomocy funkcji `glm()`. Przez GDI oznaczono algorytmu spadku gradientu rzędu I, zaś przez GDII rzędu II. Oznaczenie SGD.i odpowiada 3 algorytmom stochastycznego spadku gradientu o różnych ciągach odpowiedzialnych za długość kroku. Indeks i odpowiada ciągowi wybranemu na zasadzie $\alpha_{k,i} = i/\sqrt{k}$. Ziarno losowania ustalono na 4561.

Na wykresach na Rysunkach 3.2 - 3.5 osie dopasowane są do obecnej symulacji.

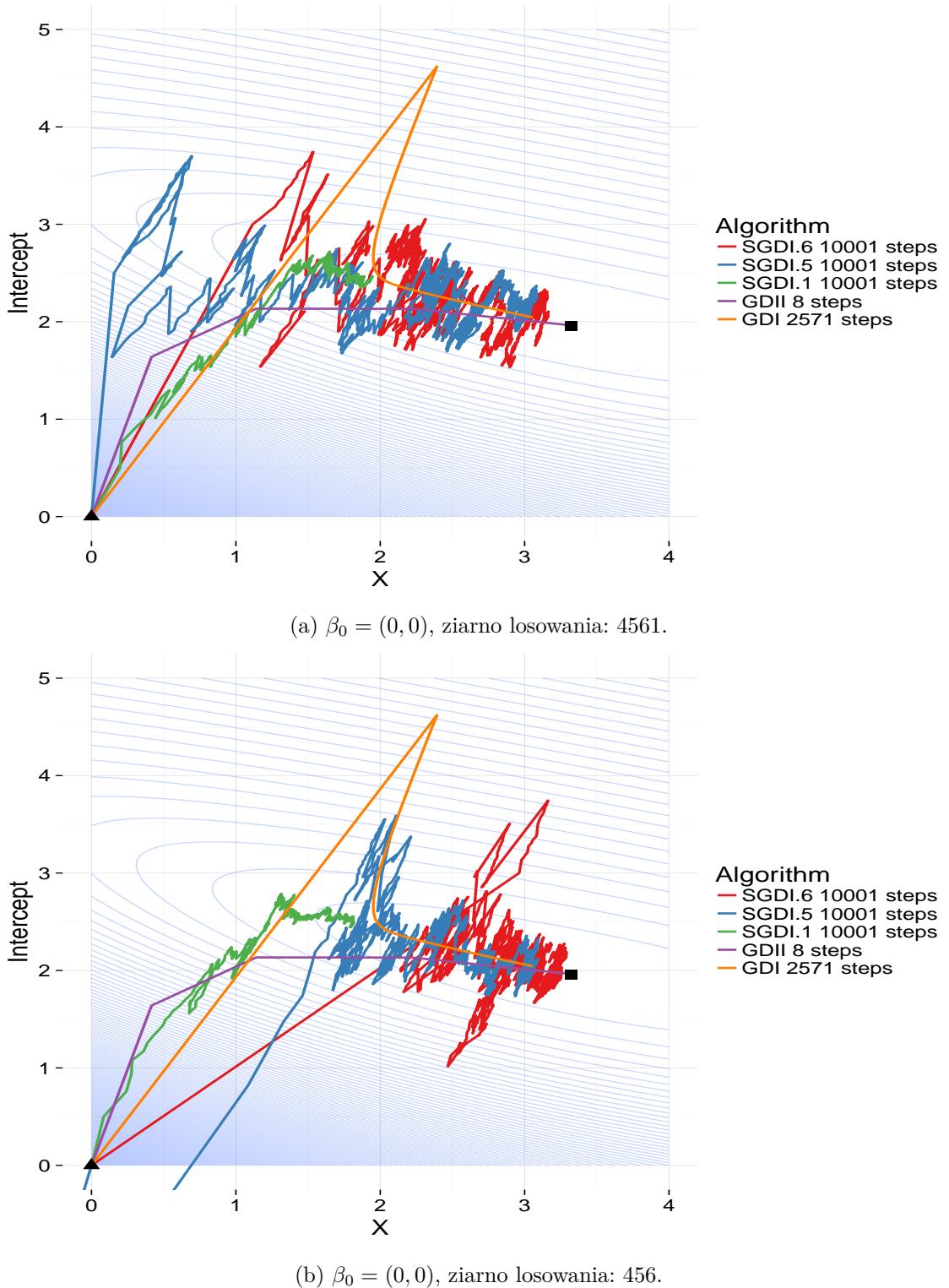
Podsumowanie symulacji

W trakcie symulacji badano zachowanie trajektorii zbieżności w kolejnych krokach algorytmów spadku gradientu. Algorytm spadku gradientu rzędu II zbiegał do rozwiązania wyznaczonego przez funkcję `glm()` i osiągał zbieżność po najmniejszej liczbie kroków, jednak należy pamiętać o kosztownych obliczeniowo operacjach odwracania macierzy Hesjanu wykonywanych w trakcie optymalizacji z wykorzystaniem tej metody.

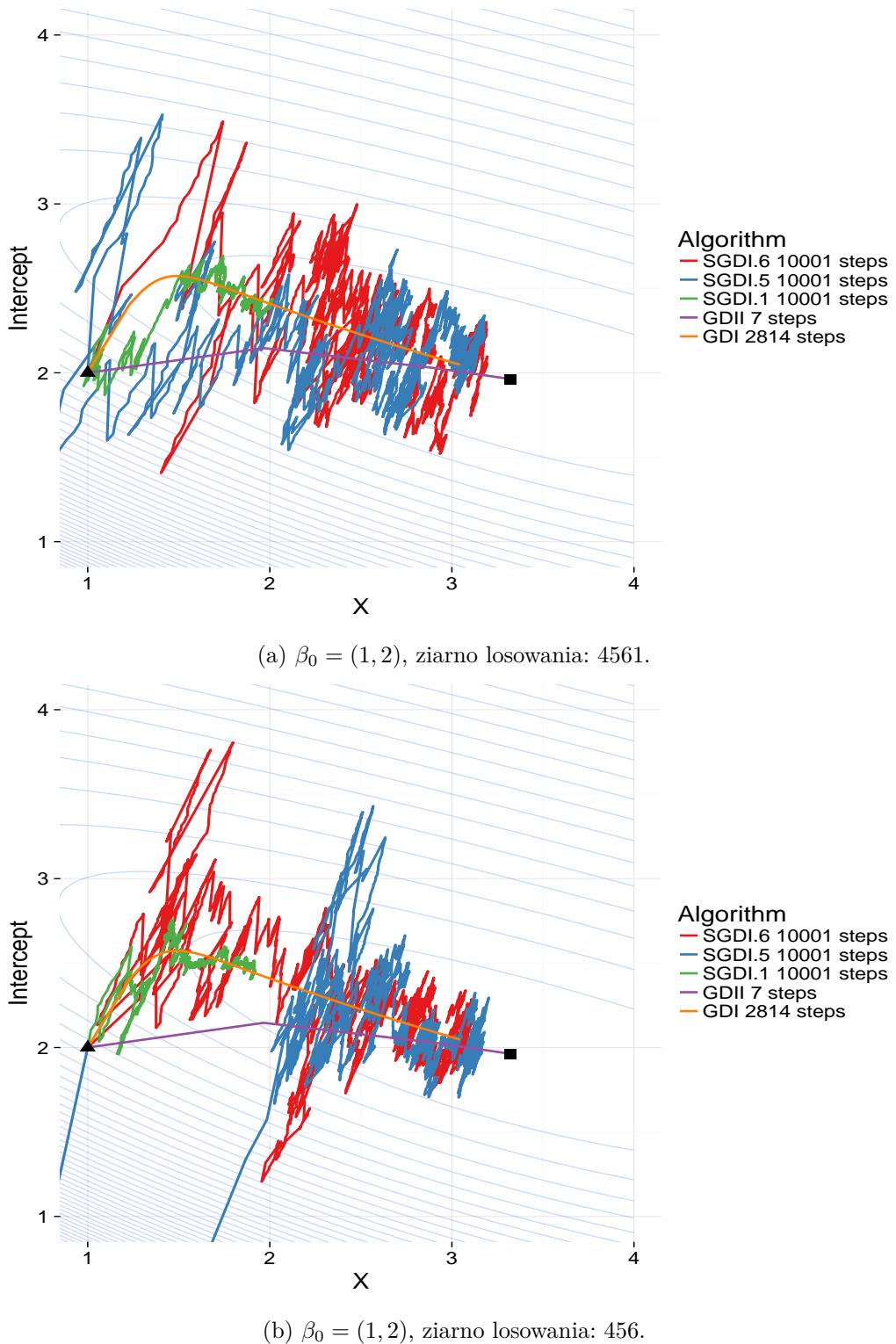
Algorytm spadku gradientu rzędu I zbiegał do rozwiązania wyznaczonego przez funkcję `glm()` w sytuacjach, gdy warunek stopu był dostatecznie mały ($\varepsilon = 10^{-5}$, Rysunek 3.2). Był również na drodze do rozwiązania w przypadku warunek stopu $\varepsilon = 10^{-4}$ (Rysunki 3.3 i 3.5), jednak nie osiągnął równie bliskich punktów co algorytm spadku gradientu rzędu II. Gdy punkt startowy był bliski rozwiązaniu pochodząemu z funkcji `glm()` oraz warunek stopu był duży, algorytm spadku gradientu rzędu I miał problem z obraniem właściwego toru zbieżności i ostatecznie wypadł najgorzej ze wszystkich sprawdzanych metod. Estymacja w tym wypadku w algorytmie spadku gradientu rzędu I miała najmniej kroków, co może wynikać z mało różnowartościowych wartości optymalizowanej funkcji log-wiarogodności w badanym obszarze, będącym stosunkowo blisko rozwiązania znalezionej przez funkcję `glm()`.

W przypadku stochastycznego spadku gradientu widać duże wahania w trajektoriach zbieżności algorytmu. W ramach symulacji badano wiele ciągów odpowiadających za długość kroku w tym algorytmie i ostatecznie zdecydowano na ukazanie w pracy tych, dzięki którym osiągana zbieżność dotyczyła największej liczby punktów startowych. Algorytm SGDI.1 w każdym z możliwych przypadków miał zbyt małe wartości ciągu odpowiadającego za długości kroków, przez co nie osiągał punktu wyliczonego przez funkcję `glm()`. SGDI.1 ukazano aby uświadomić jak ważnym aspektem jest odpowiednie dobranie długości kroków. Dla odpowiednio dobranego ciągu długości kroków o dużych wartościach, jakim był ciąg $\alpha_{ki} = i/\sqrt{k}$, $i = 5, 6$, w wielu sytuacjach doszło do zbieżności w tym samym punkcie, w którym zbiegły algorytm zaimplementowany w `glm()`. Najlepiej widać to na Rysunkach 3.2 ($\varepsilon = 10^{-5}$) i 3.3 ($\varepsilon = 10^{-4}$), gdzie w zbieżności algorytmu przeszkodził zbyt duży warunek stopu. Ciekawy rezultat osiągnięto na Rysunku 3.4, gdzie przy odmiennych początkowych ziarnach losowości osiągnięto zbieżność do punktu wyliczonego przez funkcję `glm()`, za każdym razem w innym z dwóch poważnie rozważanych przypadków SGDI.5, SGDI.6. Istota losowości odgrywa kluczową rolę w algorytmie stochastycznego spadku gradientu i należy mieć świadomość, że przy różnych ziarnach losowości, a co za tym idzie przy innej kolejności obserwacji, algorytm może zachowywać się odmiennie. Doskonale widać to na Rysunku 3.5, gdzie na górnym wykresie jeden przypadek algorytmu stochastycznego spadku gradientu osiągnął lepsze wyniki niż spadek gradientu rzędu I, zaś na dolnym wykresie żadna wersja stochastyczna nie miała trajektorii zgodnych z prawdziwym kierunkiem zbieżności.

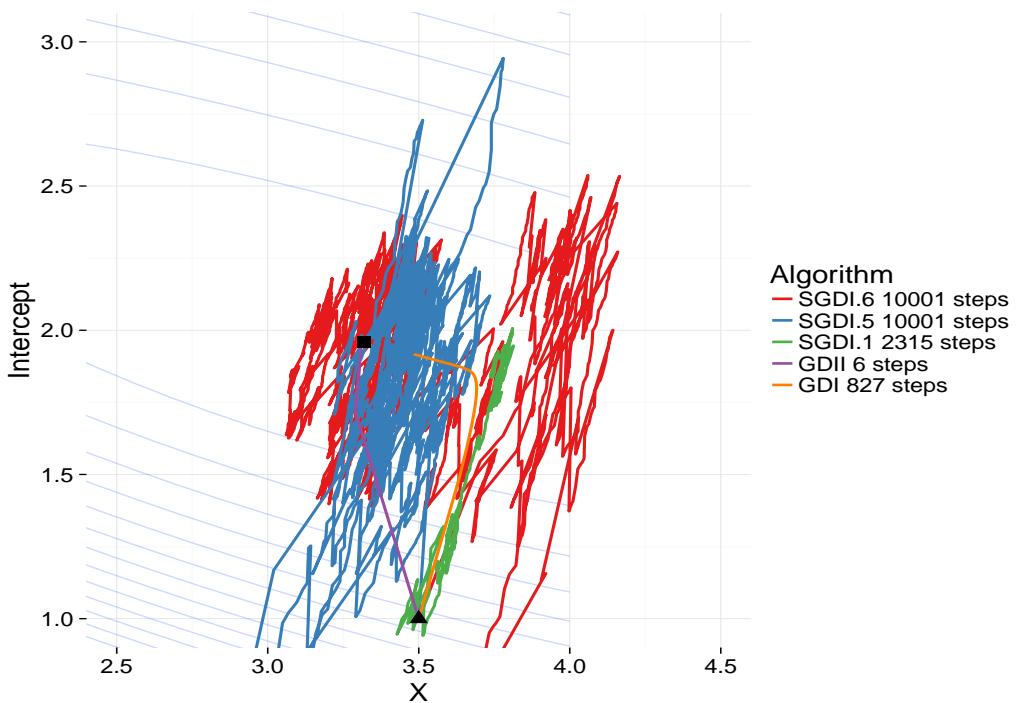
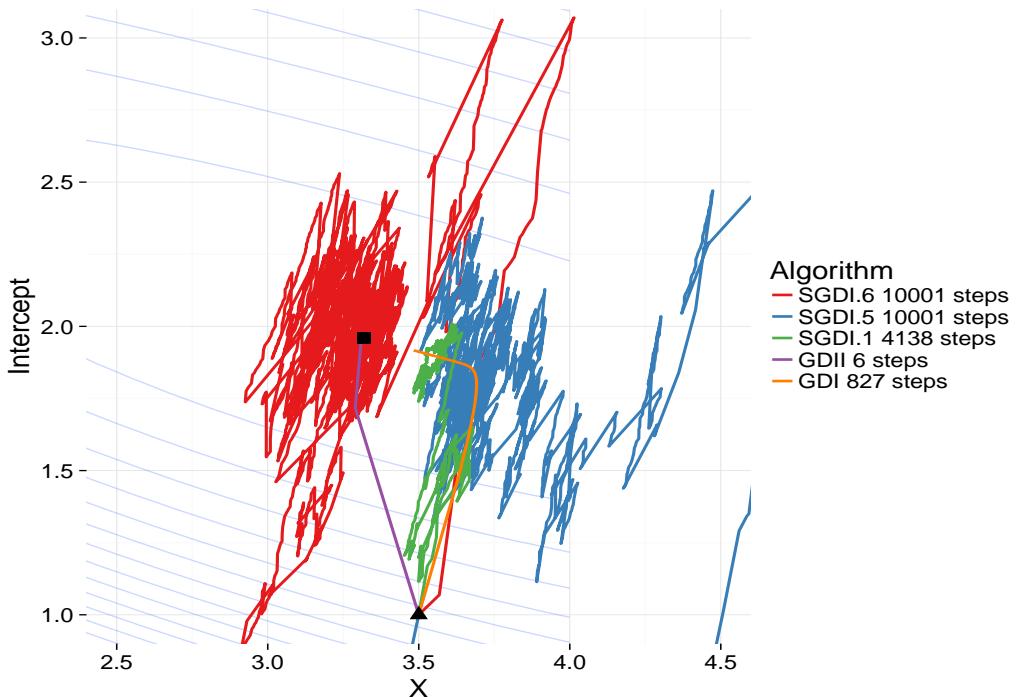
Ostatecznie można stwierdzić, że w wielu przypadkach stochastyczny wariant optymalizacji funkcji log-wiarogodności w modelu regresji logistycznej daje zadowalające rezultaty, które są niekiedy podobne do tych osiągniętych dzięki algorytmom spadku gradientu. Dzieje się to jednak za sprawą właściwego wyboru ciągu odpowiadającego za długości poszczególnych kroków oraz przy odpowiednim warunku stopu algorytmu. Należy pamiętać, że jest to jednak algorytm stochastyczny i istnieją nieliczne sytuacje, w których osiągnięte dzięki niemu współczynniki są dalekie od prawdziwych.



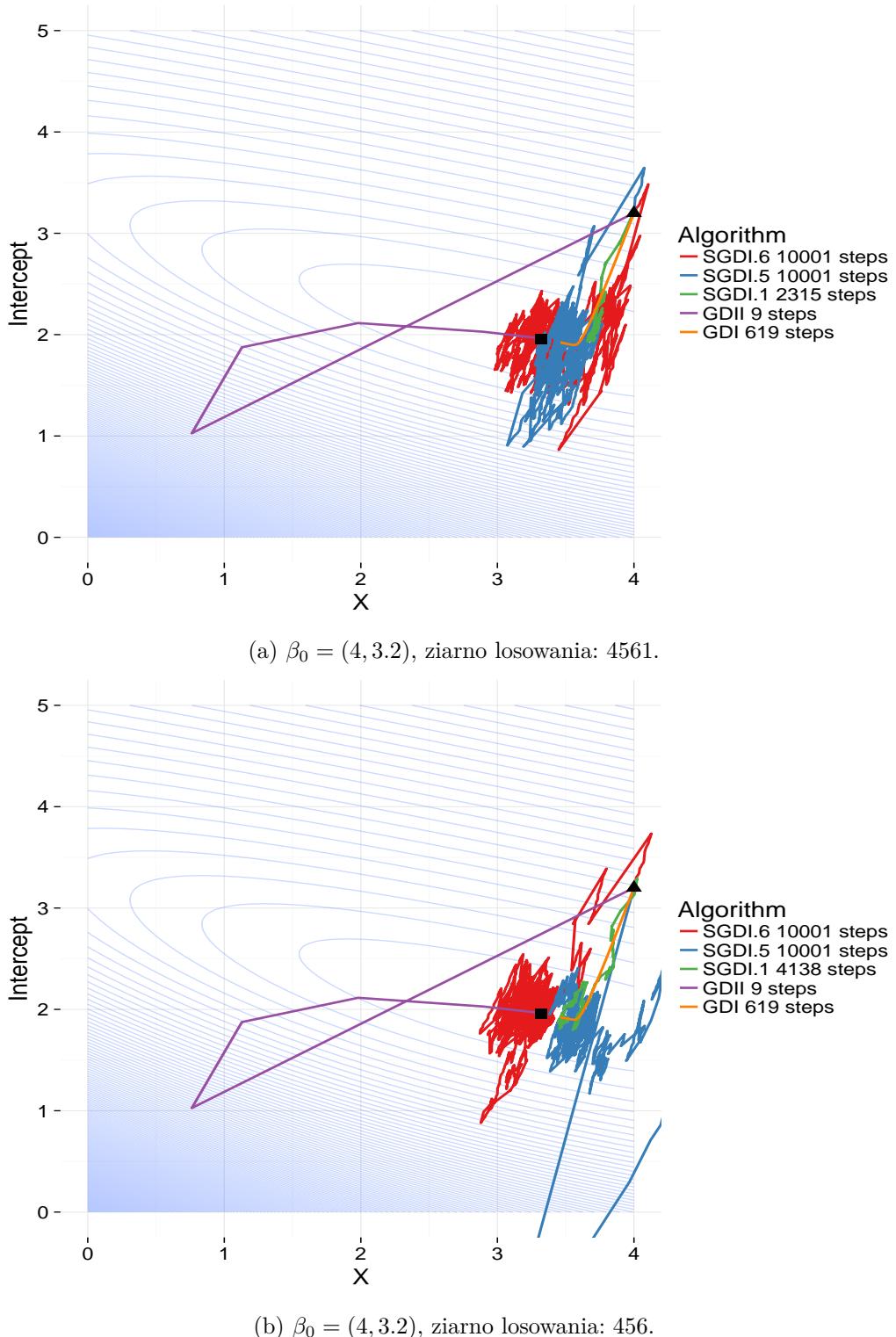
Rysunek 3.2: Porównanie algorytmów spadku gradientu w modelu regresji logistycznej. Wykresy przedstawiają ścieżki zbieżności w kolejnych krokach algorytmów spadku gradientu. Ustalono maksymalną liczbę iteracji na 10001, zaś warunek stopu na $\varepsilon = 10^{-5}$. Trójkątem zaznaczono punkt startowy, a kwadratem wyestymowane rozwiązanie przy pomocy funkcji `glm()`. Przez GDI oznaczono algorytmu spadku gradientu rzędu I, zaś przez GDII rzędu II. Oznaczenie SGD.i odpowiada 3 algorytmom stochastycznego spadku gradientu o różnych ciągach odpowiedzialnych za długość kroku. Indeks i odpowiada ciągowi wybranemu na zasadzie $\alpha_{k,i} = i/\sqrt{k}$. Punkt startowy $\beta_0 = (0, 0)$.



Rysunek 3.3: Porównanie algorytmów spadku gradientu w modelu regresji logistycznej. Wykresy przedstawiają ścieżki zbieżności w kolejnych krokach algorytmów spadku gradientu. Ustalono maksymalną liczbę iteracji na 10001, zaś warunek stopu na $\varepsilon = 10^{-5}$. Trójkątem zaznaczono punkt startowy, a kwadratem wyestymowane rozwiązanie przy pomocy funkcji `glm()`. Przez GDI oznaczono algorytmu spadku gradientu rzędu I, zaś przez GDII rzędu II. Oznaczenie SGD.i odpowiada 3 algorytmom stochastycznego spadku gradientu o różnych ciągach odpowiedzialnych za długość kroku. Indeks i odpowiada ciągowi wybranemu na zasadzie $\alpha_{k,i} = i/\sqrt{k}$. Punkt startowy $\beta_0 = (1, 2)$.

(a) $\beta_0 = (3.5, 1)$, ziarno losowania: 4561.(b) $\beta_0 = (3.5, 1)$, ziarno losowania: 456.

Rysunek 3.4: Porównanie algorytmów spadku gradientu w modelu regresji logistycznej. Wykresy przedstawiają ścieżki zbieżności w kolejnych krokach algorytmów spadku gradientu. Ustalono maksymalną liczbę iteracji na 10001, zaś warunek stopu na $\varepsilon = 10^{-5}$. Trójkątem zaznaczono punkt startowy, a kwadratem wyestymowane rozwiązanie przy pomocy funkcji `glm()`. Przez GDI oznaczono algorytmu spadku gradientu rzędu I, zaś przez GDII rzędu II. Oznaczenie SGD.i odpowiada 3 algorytmom stochastycznego spadku gradientu o różnych ciągach odpowiedzialnych za długość kroku. Indeks i odpowiada ciągowi wybranemu na zasadzie $\alpha_{k,i} = i/\sqrt{k}$. Punkt startowy $\beta_0 = (3.5, 1)$.



Rysunek 3.5: Porównanie algorytmów spadku gradientu w modelu regresji logistycznej. Wykresy przedstawiają ścieżki zbieżności w kolejnych krokach algorytmów spadku gradientu. Ustalono maksymalną liczbę iteracji na 10001, zaś warunek stopu na $\varepsilon = 10^{-5}$. Trójkątem zaznaczono punkt startowy, a kwadratem wyestymowane rozwiązanie przy pomocy funkcji `glm()`. Przez GDI oznaczono algorytmu spadku gradientu rzędu I, zaś przez GDII rzędu II. Oznaczenie SGD.i odpowiada 3 algorytmom stochastycznego spadku gradientu o różnych ciągach odpowiedzialnych za długość kroku. Indeks i odpowiada ciągowi wybranemu na zasadzie $\alpha_{k,i} = i/\sqrt{k}$. Punkt startowy $\beta_0 = (4, 3.2)$.

Rozdział 4

Estymacja w modelu Coxa metodą stochastycznego spadku gradientu

Poniższy rozdział przedstawia implementację oraz zastosowanie metody stochastycznego spadku gradientu do estymacji współczynników w modelu proporcjonalnych hazardów Coxa. Jest to główny cel pracy. Przedstawione rozważania odnośnie omawianego podejścia są nowatorskie, dlatego nie mogą być poparte literaturą naukową.

W czasie powstawania pracy nawiązano jedynie krótką wymianę informacji z pracownikami *Harvard Laboratory for Applied Statistical Methodology & Data Science*, dzięki której dowiedziano się, że podjęte zostały kroki w kierunku stworzenia podwalin pod teorię do omawianego zagadnienia, jednak ewentualne publikacje nie zostały jeszcze dokończone. Przedsmak zaimplementowanego podobnego algorytmu przez pracowników wyżej wymienionego laboratorium można znaleźć w [Tran et al. \(2015\)](#).

W procesie estymacji współczynników w omawianym modelu wykorzystano pochodne częstotliwej funkcji log-wiarogodności (2.9)

$$U_k(\beta) = \frac{\partial_p \ell_k(\beta)}{\partial \beta_k} = \sum_{i=1}^K \left(X_{ik} - \frac{\sum_{l \in \mathcal{R}(t_i)} X_{lk} e^{X_l' \beta}}{\sum_{l \in \mathcal{R}(t_i)} e^{X_l' \beta}} \right) = \sum_{i=1}^n Y_i \left(X_{ik} - \frac{\sum_{l \in \mathcal{R}(t_i)} X_{lk} e^{X_l' \beta}}{\sum_{l \in \mathcal{R}(t_i)} e^{X_l' \beta}} \right) = \sum_{i=1}^n U_{k,i}(\beta),$$

($Y_i = 1$, gdy obserwacja nie była cenzurowana i $Y_i = 0$, gdy obserwacja była cenzurowana; K to liczba zdarzeń; n to liczba obserwacji) oraz wzór (3.2), którego wersja z adekwatnymi oznaczeniami do powyższej funkcji wygląda następująco

$$\beta_{k,j+1} = \beta_{k,j} - \alpha_j U_{k,i}(\beta_{k,j}), \quad (4.1)$$

gdzie j oznacza krok algorytmu, i iteruje składniki U_k , α_j to długość j -tego kroku algorytmu, zaś U_k to k -ta pochodna częstotliwej funkcji log-wiarogodności U oraz $\beta_{k,j}$ to (dla j -tego kroku) k -ta współrzędna estymowanego wektora współczynników β o rozmiarze p , czyli $k = 1, \dots, p$.

Własna implementacja algorytmu w języku \mathcal{R} znajduje się w podrozdziale (4.2).

4.1. Założenia i obserwacje

Skupiąc się na informatycznym aspekcie algorytmu można stwierdzić, że idea stochastycznego spadku gradientu polega na losowaniu składnika optymalizowanej funkcji. Jednak ze statystycznego punktu widzenia, metoda stochastycznego spadku gradientu opiera się o losowanie indeksu obserwacji ze zbioru, z którego uczyony jest algorytm, zanim postanowi się w jakikolwiek sposób przedstawić konkretną funkcję wiarogodności. Zatem w celu estymacji, w oparciu o stochastyczny spadek gradientu w modelu Coxa, konstruując metodę należy najpierw losować obserwacje, a następnie dopiero wyznaczać formę optymalizowanej funkcji częściowej log-wiarogodności.

Dla wielu modeli opierających się o funkcje wiarogodności te dwa punkty widzenia są równoważne, jednak nie w przypadku modelu Coxa nie, gdzie niektóre składniki w funkcji log-wiarogodności są zależne od poprzednich obserwacji. W przypadku modelu ADALINE sformułowanego jak w [Widrow and Ho \(1960\)](#), opartego na minimalizowaniu funkcji kosztu w postaci błędu najmniejszych kwadratów, w [Bottou \(2012\)](#) podano postać funkcji straty oraz równanie algorytmu stochastycznego spadku gradientu jak poniżej

$$Q(w) = \frac{1}{2}(y - w'\Phi(x))^2, \quad \Phi(x) \in \mathbb{R}^d, y \in \{-1, 1\},$$

$$w \leftarrow w + \alpha_k(y_k - w'\Phi(x_t))\Phi(x_t),$$

przy których widać, że w kolejnych krokach algorytmu t wystarczy tylko jedna obserwacja $z_t = (x_t, y_t)$ aby poprawić oszacowanie parametru w .

W modelu proporcjonalnych hazardów Coxa jest to bardziej skomplikowane. Mając z góry zadaną funkcję hazardu, częściowa funkcja wiarogodności odpowiada prawdopodobieństwu tego, że obserwowane zdarzenia zdarzyłyby się dokładnie w tej kolejności w jakiej się pojawiły. To prawdopodobieństwo zależy od wszystkich obserwacji w zbiorze. Niemożliwe jest wyliczenie tego poprzez obliczenie wartości funkcji częściowej wiarogodności oddzielnie dla obserwacji o numerach od 1 do 5 i oddzielnie dla obserwacji od numerach 6 do 10, a następnie przemnożeniu wyników przez siebie. Z tej przyczyny niemożliwe jest losowanie czynników optymalizowanej funkcji przy użyciu metody stochastycznego spadku gradientu do estymacji współczynników w tym modelu. Alternatywnym podejściem do tego problemu mogłoby być pamiętanie wartości licznika i mianownika w składnikach częściowej funkcji log-wiarogodności, dla wszystkich zaobserwowanych czasów zdarzeń i poprawianie odpowiednich składników, z wykorzystaniem świeżych obserwacji. Taki zabieg jest pamięciowo oszczędniejszy niż wykorzystywanie całego zbioru danych. **Innym rozwiązaniem jest losowanie podzbioru obserwacji, a następnie konstruowanie funkcji wiarogodności dla zaobserwowanego zredukowanego zbioru. Właśnie ta metoda zostanie opisana w dalszej części pracy.** Taki sposób wprowadzania obserwacji do estymacji można wykorzystać w sytuacjach, gdy ma się do czynienia z nieskończonym napływem nowych obserwacji, a potrzebne jest oszacowanie estymowanych parametrów modelu $\beta_k, k = 1, \dots, p$ dla obecnie zaobserwowanych i wykorzystanych obserwacji. Proces ten ma dwie zalety: nie dość, że dla nowych obserwacji model jest w stanie na bazie obecnych oszacowań parametrów dokonać predykcji proporcji hazardów, to dodatkowo po każdej porcji obserwacji aktualizuje parametry modelu.

Ponieważ omawiane algorytmy rozwiązujeją problem minimalizacji badanej funkcji, zaś celem estymacji w modelu Coxa jest znalezienie parametrów modelu maksymalizujących funkcję częściowej log-wiarogodności, zatem wzięcie do minimalizacji funkcji z przeciwnym znakiem doprowadzi do wykorzystania metod znajdujących minimum do znalezienia maksimum.

Zakładając, że w j -tym kroku algorytmu i przy k -tej pochodnej cząstkowej dysponuje się zaobserwowanym podzbiorem \mathcal{B} , częściową funkcję wiarogodności dla zaobserwowanego podzbioru obserwacji \mathcal{B} wykorzystywaną do minimalizacji można zapisać następująco

$$-U_k^{\mathcal{B}}(\beta_{\cdot,j}) = -\sum_{i \in \mathcal{B}_{\text{ind}}} U_{k,i}^{\mathcal{B}}(\beta_{\cdot,j}) = -\sum_{i \in \mathcal{B}_{\text{ind}}} Y_i \left(X_{ik} - \frac{\sum_{l \in \mathcal{R}_{\mathcal{B}}(t_i)} X_{lk} e^{X_l' \beta_{\cdot,j}}}{\sum_{l \in \mathcal{R}_{\mathcal{B}}(t_i)} e^{X_l' \beta_{\cdot,j}}} \right), \quad (4.2)$$

gdzie indeksy obserwacji należące do \mathcal{B} definiuje się jako $\mathcal{B}_{\text{ind}} = \{i : X_i \in \mathcal{B}\}$, zaś $\mathcal{R}_{\mathcal{B}}(t_i)$ to zbiór ryzyka dla zbioru \mathcal{B} w czasie t_i , a $\beta_{\cdot,j}$ to wektor współczynników w j -tym kroku.

Postać powyższej funkcji nasuwa pewne obserwacje. W danym kroku algorytmu, obecnie wykorzystywany zaobserwowany podzbiór obserwacji \mathcal{B}

- nie powinien składać się jedynie z obserwacji cenzurowanych

Dla podzbioru zawierającego jedynie takie obserwacje wartość pochodnej cząstkowej funkcji log-wiarogodności jest równa zero, ze względu na czynniki Y_i , które dla obserwacji cenzurowanych są równe zero. Zerowa wartość pochodnej cząstkowej funkcji log-wiarogodności doprowadzi do niezmienienia się optymalizowanych parametrów we wzorze (4.1), a co za tym idzie, doprowadzi do przerwania optymalizacji.

- nie powinien składać się jedynie z jednej obserwacji

W takim przypadku wartość pochodnej cząstkowej funkcji log-wiarogodności jest również równa zero, bez względu na to czy obserwacja była cenzurowana czy nie, gdyż

$$-U_k^{\mathcal{B}}(\beta_{\cdot,j}) = -Y_m \left(X_{mk} - \frac{X_{mk} e^{X_m' \beta_{\cdot,j}}}{e^{X_m' \beta_{\cdot,j}}} \right) = -Y_m (X_{mk} - X_{mk}) = -Y_m \cdot 0 = 0,$$

dla dowolnego m oznaczającego indeks jedynej obserwacji w \mathcal{B} , czyli $\mathcal{B}_{\text{ind}} = \{m\}$.

- nie powinien zawierać tylko jednej obserwacji niecenzurowanej w zbiorze, która dodatkowo ma największy czas bycia pod obserwacją

W tej sytuacji jedyny niezerowy czynnik w całej pochodnej cząstkowej funkcji log-wiarogodności zajdzie tylko dla obserwacji niecenzurowanej, dla której zbiór ryzyka będzie zawierał tylko ją samą, co da ostatecznie zerową wartość pochodnej cząstkowej optymalizowanej funkcji log-wiarogodności i doprowadzi do przerwania optymalizacji.

Dodatkowo nie dochodzi do wykorzystania obserwacji cenzurowanych, gdy:

- zaobserwowany zbiór zawiera obserwacje cenzurowane, które wszystkie mają czasy obserwacji krótsze niż najmniejszy czas obserwacji dla obserwacji niecenzurowanej

Obserwacje cenzurowane niosą ze sobą mniej informacji, jednak teoria analizy przeżycia stara się je maksymalnie wykorzystać. Stąd uwzględnia się ich udział w zbiorze ryzyka przy estymacji fragmentów funkcji log-wiarogodności dla obserwacji niecenzurowanych. W sytuacji, gdy wszystkie obserwacje cenzurowane mają czas obserwacji krótszy niż najmniejszy czas obserwacji dla obserwacji niecenzurowanej w podzbiorze, nie dochodzi do wykorzystania obserwacji cenzurowanych w jakikolwiek sposób przy estymacji współczynników w danym kroku algorytmu.

Mając na względzie te uwagi, w sytuacji gdy zaobserwuje się podzbiór, który spełnia jeden z wyżej wymienionych warunków, zamiast wykorzystywać ten zbiór do estymacji można go dołączyć do kolejnego podzbioru, który ma być zaobserwowany.

4.2. Implementacja

Omówiona w tym rozdziale metoda estymacji metodą stochastycznego spadku gradientu w modelu proporcjonalnych hazardów Coxa została zaimplementowana w języku \mathcal{R} i jest dostępna w specjalnie przygotowanym pakiecie o nazwie `coxphSGD()`, który można jednocześnie pobrać z internetu i zainstalować polecienniem

```
devtools::install_github("MarcinKosinski/coxphSGD")
```

Dokumentacja wraz z opisem argumentów w języku angielskim funkcji `coxphSGD()`, która estymuje współczynniki w modelu proporcjonalnych hazardów Coxa metodą stochastycznego spadku gradientu, dostępna jest w Dodatku A. Starano się zachować jednorodność kolejności i nazewnictwa parametrów z funkcją `coxph()` z pakietu `survival` (Therneau, 2015; Therneau and Grambsch, 2000).

Implementacja algorytmu estymacji w modelu Coxa metodą stochastycznego spadku gradientu opiera się na poniższym pseudo-kodzie i zakłada, że kolejne podzbiory \mathcal{B} dostarczane są jako kolejne elementy listy.

Estymacja w modelu Coxa metodą stochastycznego spadku gradientu

```
# wstępna inicjalizacja parametrów
eps = 1e-5                                # warunek stopu.

n = length(data)                            # data jest listą ramek danych.

diff = eps + 1                             # różnice w oszacowaniach parametrów
                                             # między kolejnymi krokami.

learningRates = function(x) 1/x            # długości kroku algorytmu.

beta_old = numeric(0, length = k)          # punkt startowy długości  $k$ ,
                                             # gdzie  $k$  to liczba zmiennych
                                             # objaśniających w modelu.

max.iter = 500                             # maksymalna liczba kroków.

# estymacja
i = 1                                      # iterator kroku algorytmu.

while(i <= max.iter & diff < eps) do
  iter = ifelse(i mod n == 0, n, i mod n) # wybierz kolejny podzbiór batch.
  batch = data[[iter]]
  beta_new = beta_old - learningRates(i) * U_Batch(batch)
                                             #  $U_Batch$  to gradient częściowej funkcji
                                             # log-wiarogdności dla zaobserwowanego
                                             # zbioru 'batch'.

  diff = euclidean_dist(beta_new, beta_old) # odległość euklidesowa
  beta_old = beta_new
  i = i + 1
end while
return beta_new
```

Docelowa implementacja w języku \mathcal{R} znajduje się poniżej.

```

coxphSGD <- function(formula, data, learningRates = function(x){1/x},
                      beta_0 = 0, epsilon = 1e-5, max.iter = 500 ) {
  # check arguments
  checkArguments(formula, data, learningRates,
                 beta_0, epsilon) -> beta_start
  n <- length(data)
  diff <- epsilon + 1
  i <- 1
  beta_new <- list()      # steps are saved in a list so that they can
  beta_old <- beta_start # be traced in the future
  # estimate
  while(i <= max.iter & diff > epsilon) {
    beta_new[[i]] <- coxphSGD_batch(formula = formula, beta = beta_old,
                                      learningRate = learningRates(i), data = data[[ifelse(i%%n==0,n,i%%n)]]) %>%
      unlist
    diff <- sqrt(sum((beta_new[[i]] - beta_old)^2))
    beta_old <- beta_new[[i]]
    i <- i + 1
    cat("\r iteration: ", i, "\r")
  }
  # return results
  list(Call = match.call(), epsilon = epsilon,
       learningRates = learningRates, steps = i,
       coefficients = c(list(beta_start), beta_new))
}

coxphSGD_batch <- function(formula, data, learningRate, beta){
  # collect times, status, variables and reorder samples
  # to make the algorithm more clear to read and track
  batchData <- prepareBatch(formula = formula, data = data)
  # calculate the log-likelihood for this batch sample
  batchData <- batchData %>% arrange(-times)
  # scores occure in nominator and denominator
  scores <- apply(batchData[, -c(1, 2)], 1,
                  function(element) exp(element %*% beta) )
  nominator <- apply(batchData[, -c(1, 2)], 2,
                      function(element) cumsum(scores*element) )
  denominator <- cumsum(scores)
  # sum over non-censored observations
  partial_sum <- (batchData[, -c(1, 2)] - nominator/denominator)*batchData[, "event"]
  # each column indicates one explanatory variable
  U_batch <- partial_sum %>% colSums()
  return(beta + learningRate * U_batch)
}

```

4.3. Symulacje estymacji w modelu Coxa

Dzięki przygotowanej w rozdziale 2.4 funkcji `dataCox()` możliwe będzie symulacyjne zbadanie zachowania się współczynników w modelu Coxa w trakcie estymacji metodą stochastycznego spadku gradientu, której algorytm został zaimplementowany w rozdziale 4.2. Poniższe wywołanie zwraca ramkę danych o 10 tysiącach wierszy, wraz ze sztucznie wygenerowanymi czasami z rozkładu Weibulla o parametrach $\lambda = 3$ oraz $\rho = 2$. Czasy cenzurowania generowane są z rozkładu wykładniczego o parametrze 0.2, a ostateczne czasy bycia pod obserwacją odpowiadają współczynnikom modelu $\beta = (1, 3)$. Porównując czas życia oraz czas cenzurowania wylosowany dla obserwacji, w rezultacie otrzymać można czasy przeżycia oraz status zdarzenia bądź cenzurowania.

```
x <- matrix(sample(0:1, size = 20000, replace = TRUE), ncol = 2)
dCox <- dataCox(10^4, lambda = 3, rho = 2, x, beta = c(1,3), censRate = 0.2)
```

Funkcja `simulateCoxSGD()` opisana w Dodatku A.1 pięciokrotnie dzieli wygenerowany zbiór danych odpowiednio na 30, 60, 90, 120 i 200 podzbiorów losowych długości ustawiając obserwacje w losowej kolejności, a następnie wykorzystuje funkcję `coxphSGD()` do wyznaczenia wartości współczynników w kolejnych krokach estymacji funkcji log-wiarogodności w modelu Coxa przy pomocy metody stochastycznego spadku gradientu. Dzięki tak otrzymanym współczynnikom modelu w kolejnych krokach algorytmu przy różnych podziałów zbiorów, możliwe było graficzne przedstawienie trajektorii zbieżności algorytmu w zależności od punktu startowego, warunku stopu oraz wartości ciągu odpowiadającego długościom kroków.

Implementacja umożliwia ponowne wykorzystanie wszystkich zaobserwowanych podzbiorów danych w kolejnych krokach algorytmu, jeżeli nie pozostały już do wykorzystania żadne nowe podzbiory, zaś zbieżność algorytmu, poza warunkiem stopu, zależna jest także od maksymalnej liczby iteracji, która jest większa od liczby zaobserwowanych podzbiorów. Jednorazowe wykorzystanie wszystkich obserwacji w trakcie procesu optymalizacji nazywane jest *epoką*.

Przykładowe wywołanie funkcji `simulateCoxSGD()` znajduje się poniżej

```
simulateCoxSGD(dCox, learningRates = function(x){1/(100*sqrt(x))},
                max.iter = 10, epsilon = 1e-5, beta_0 = c(2,2))
```

gdzie w tym wypadku parametr `max.iter` odpowiada za liczbę epok do wykorzystania, `learningRates` to funkcja odpowiedzialna za wyznaczenie długości kolejnych kroków algorytmu, `dCox` to dane do analizy przygotowane w postaci jaką zwraca funkcja `dataCox()`, `epsilon` to warunek stopu, zaś `beta_0` to punkt startowy algorytmu.

Na Rysunkach 4.1 - 4.4 przedstawiono symulacje zbieżności procesu optymalizacji częściowej funkcji log-wiarogodności w modelu Coxa proporcjonalnych hazardów, konstruowanej jedynie dla zaobserwowanego w danym kroku podzbioru obserwacji, z wykorzystaniem algorytmu stochastycznego spadku gradientu. Trajektorie zbieżności zależne są od ustawionego punktu startowego, warunku stopu oraz wartości ciągu odpowiadającego długościom kroków w algorytmie. Dodatkowo elipsami przedstawiono warstwice częściowej funkcji log-wiarogodności wyliczone mając wszystkie zaobserwowane podzbiory połączone w jeden. Trójkątem zaznaczono punkt startowy algorytmu, kwadratem teoretyczne maksimum, zaś gwiazdką współczynniki uzyskane za pomocą funkcji `coxph()`, która znajduje rozwiązania przy pomocy algorytmu spadku gradientu rzędu II (Raphsona-Newtona).

Podsumowanie symulacji w modelu proporcjonalnych hazardów Coxa

W trakcie symulacji badano jak podział całego zbioru obserwacji na podzbiory wpływa na trajektorie optymalizacji częściowej funkcji log-wiarogodności w modelu Coxa proporcjonalnych hazardów, konstruowanej na bazie zaobserwowanego podzbioru. Badano również wpływ punktu początkowego, warunku stopu, liczby epok oraz konstrukcji ciągu odpowiedzialnego za długości kroków na proces optymalizacji w tym modelu.

Na Rysunkach 4.1 - 4.4 ukazano trajektorie powstałe przy punktach początkowych

$$\beta_0 = (0, 0), (2, 2), (-1, 4),$$

gdzie pod uwagę brano ciągi odpowiadające długościom kroków takie jak

$$\alpha_t = \frac{1}{20\sqrt{t}}, \quad \alpha_t = \frac{1}{50\sqrt{t}}, \quad \alpha_t = \frac{1}{100\sqrt{t}}.$$

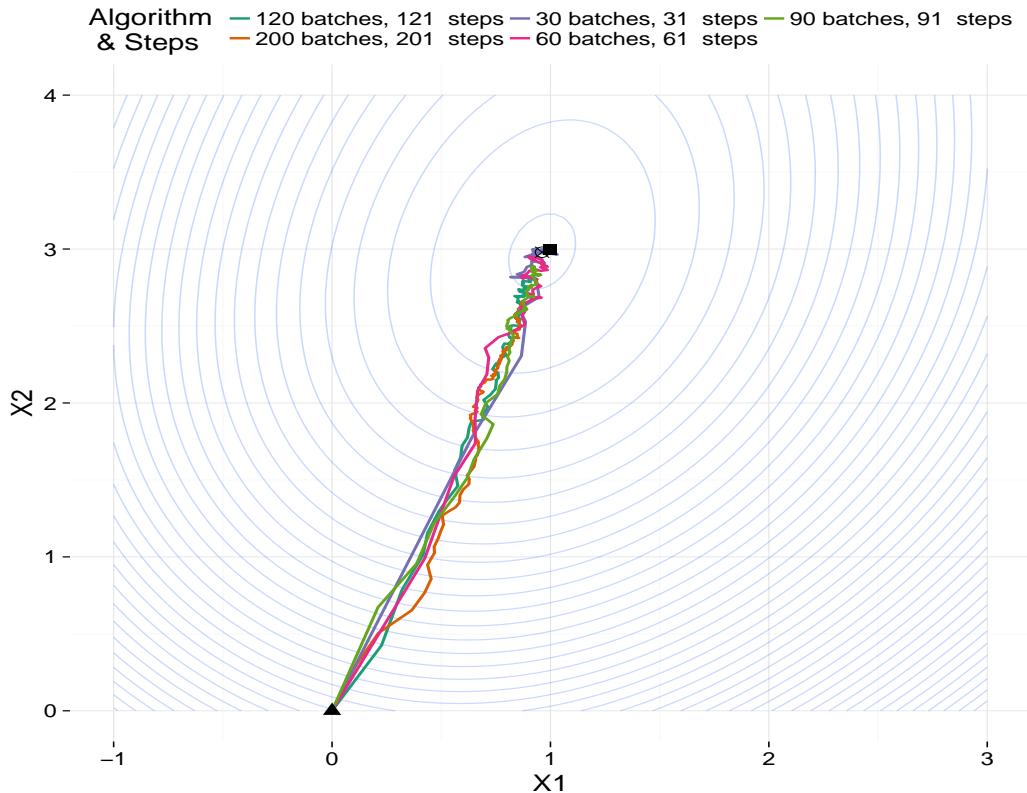
Eksperymentowano z liczbą epok ustalaną na 1, 5, 10 oraz sprawdzano jak zachowają się trajektorie z warunkami stopu $\varepsilon = 10^{-6}$ lub $\varepsilon = 10^{-5}$. Przy ciągach odpowiadających za długości kroków ustalonych na

$$\alpha_t = \frac{1}{20t}, \quad \alpha_t = \frac{1}{50t}, \quad \alpha_t = \frac{1}{100t}$$

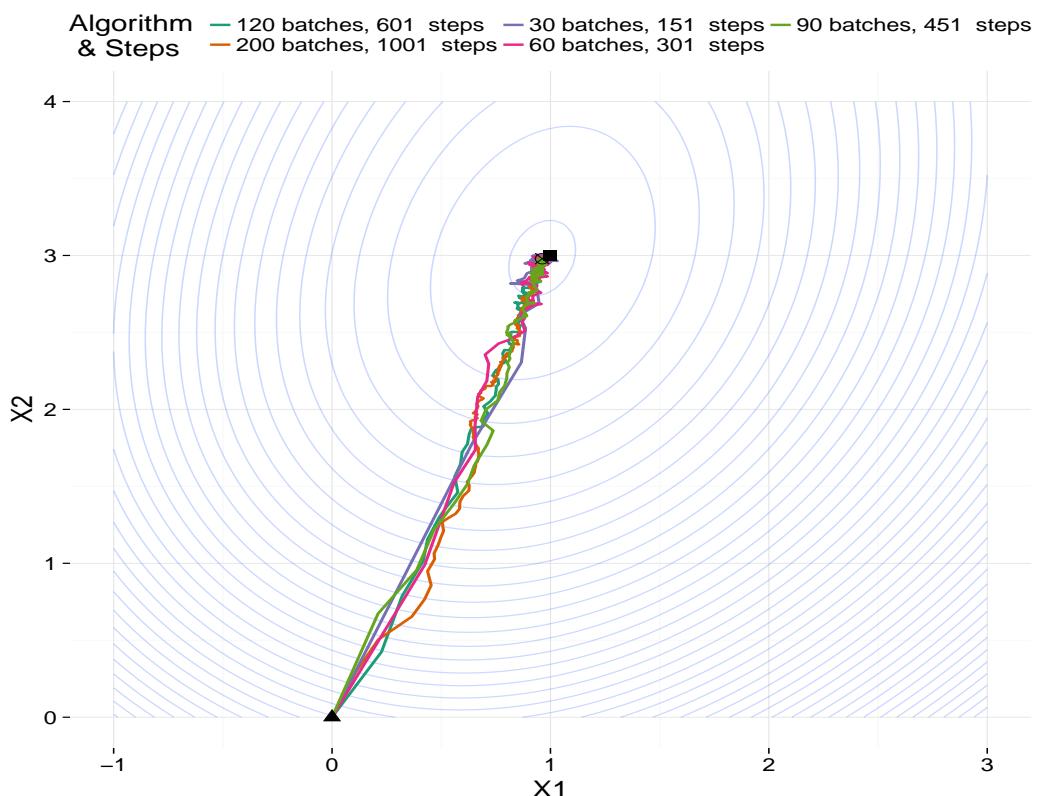
algorytm nie osiągał zbieżności i zatrzymywał się po maksymalnej liczbie iteracji w losowym punkcie znacznie odległym od optimum.

W każdej z symulacji algorytm zbiegł do punktu, w którym powinno być teoretyczne optimum. W wielu wypadkach już po jednej epoce. Może to oznaczać, że opisana metoda dobrze sprawdza się w sytuacji napływających danych. Należy zaznaczyć, że przy odmiennych sprawdzanych wariantach procesu optymalizacji, za każdym razem metoda zbiegała prostopadle do ukazanych warstwic częściowej funkcji log-wiarogodności konstruowanej na bazie **całych** danych. Jest to jasny sygnał świadczący za poprawnością wyliczania wartości gradientu częściowej funkcji log-wiarogodności, konstruowanej na bazie zaobserwowanego podzbioru. Co więcej, stochastyczny spadek gradientu nie wykorzystuje wszystkich obserwacji jednocześnie, przez co trudniej jest mu zbiec do teoretycznego optimum, a jednak w opisanej sytuacji algorytm osiągnął cel, dodatkowo zyskując na szybkości zbieżności i złożoności obliczeniowej. Warta podkreślenia jest stabilność procesu optymalizacji wśród ciągów odpowiadających za długość kroków, które ustawione zostały na mniejsze wartości (Rysunki 4.1 - 4.3). Ostatecznie należy zwrócić uwagę na fakt, że większa liczba epok i mniejszy warunek stopu powodują jedynie poprawienie optymalizacji dla wariantów, w których cały zbiór obserwacji był podzielony na największą liczbę podzbiorów, a więc każdy krok algorytmu budowany był na najmniej licznych podzbiorach. W pozostałych przypadkach nie poprawia to optymalizacji. Można sądzić, że im liczniejszy zaobserwowany podzielony tym skuteczniejsza estymacja.

Proces estymacji w modelu proporcjonalnych hazardów Coxa z wykorzystaniem metody stochastycznego spadku gradientu w sytuacji napływających danych jest stabilny i zbiega w okolice bliskie teoretycznemu optimum, niekiedy nawet już po jednej epoce. Przy dobrze dobranych parametrach optymalizacji, proces może być z powodzeniem wykorzystywany do znajdowania współczynników modelu. Kluczowym momentem mającym wpływ na powodzenie optymalizacji jest wybór ciągu odpowiedzialnego za dobór długości kroków w algorytmie, więc zaleca się symulacje badające odmienne ciągi oraz wybranie tego ciągu, który w większości przypadków daje stabilne, jednorodne współczynniki, maksymalizujące częściową funkcję wiarogodności konstruowaną na bazie oddzielnego zbioru testowego.

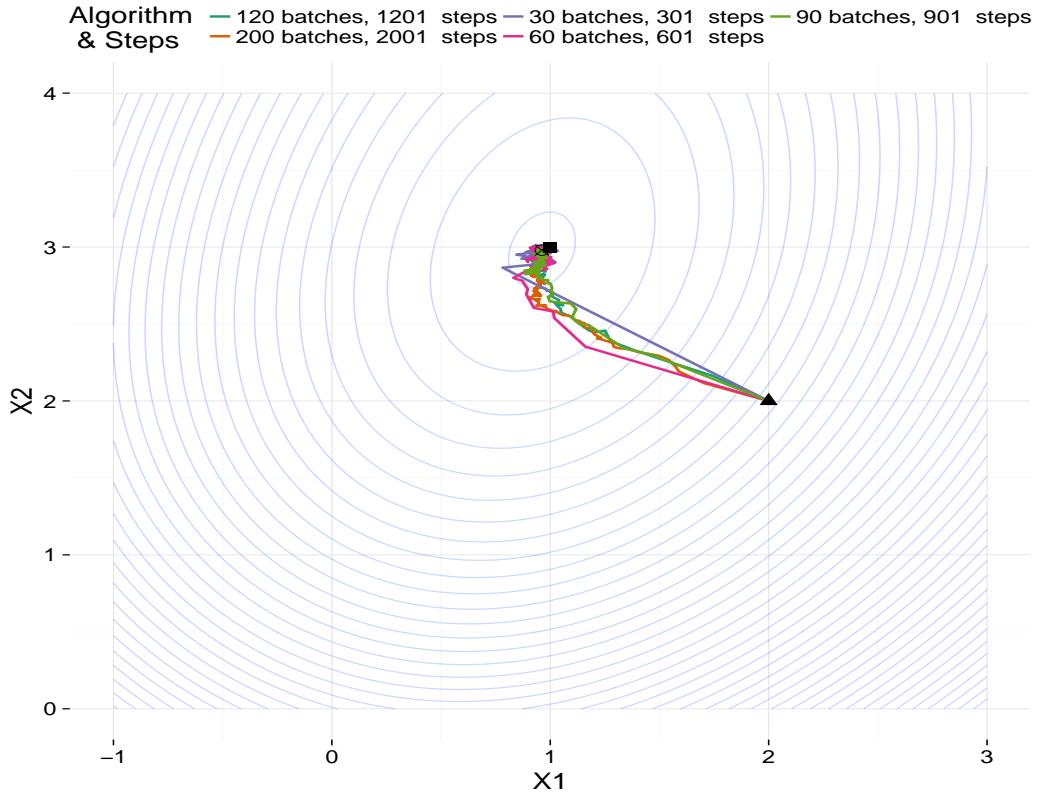


(a) $\beta_0 = (0, 0)$, liczba epok 1, warunek stopu: $\varepsilon = 10^{-6}$, długości kroków = $\frac{1}{50\sqrt{t}}$.

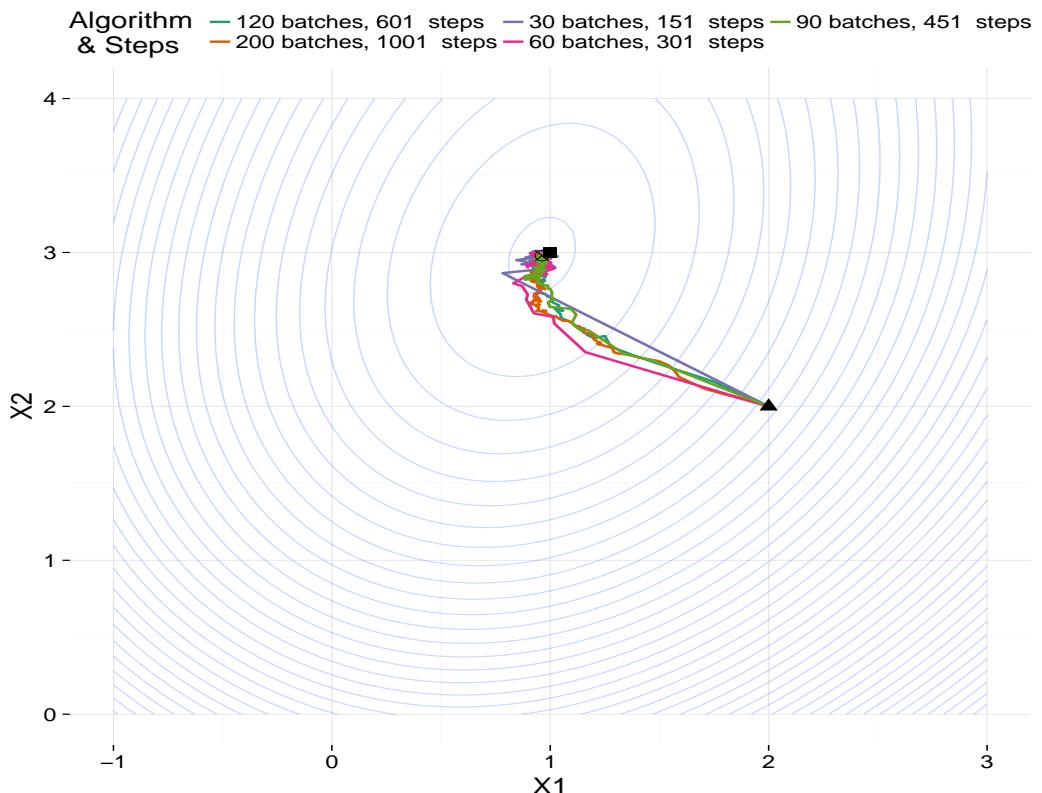


(b) $\beta_0 = (0, 0)$, liczba epok 5, warunek stopu: $\varepsilon = 10^{-6}$, długości kroków = $\frac{1}{50\sqrt{t}}$.

Rysunek 4.1: Porównanie trajektorii estymacji w modelu Coxa, wykorzystującym metodę stochastycznego spadku gradientu, w zależności od różnych podziałów zbioru początkowego na podzbiory. Elipsami przedstawiono warstwice częściowej funkcji log-wiarogodności wyliczone na bazie wszystkich obserwacji. Trójkątem zaznaczono punkt startowy algorytmu, kwadratem teoretyczne maksimum, zaś gwiazdką współczynniki uzyskane za pomocą funkcji `coxph()`.

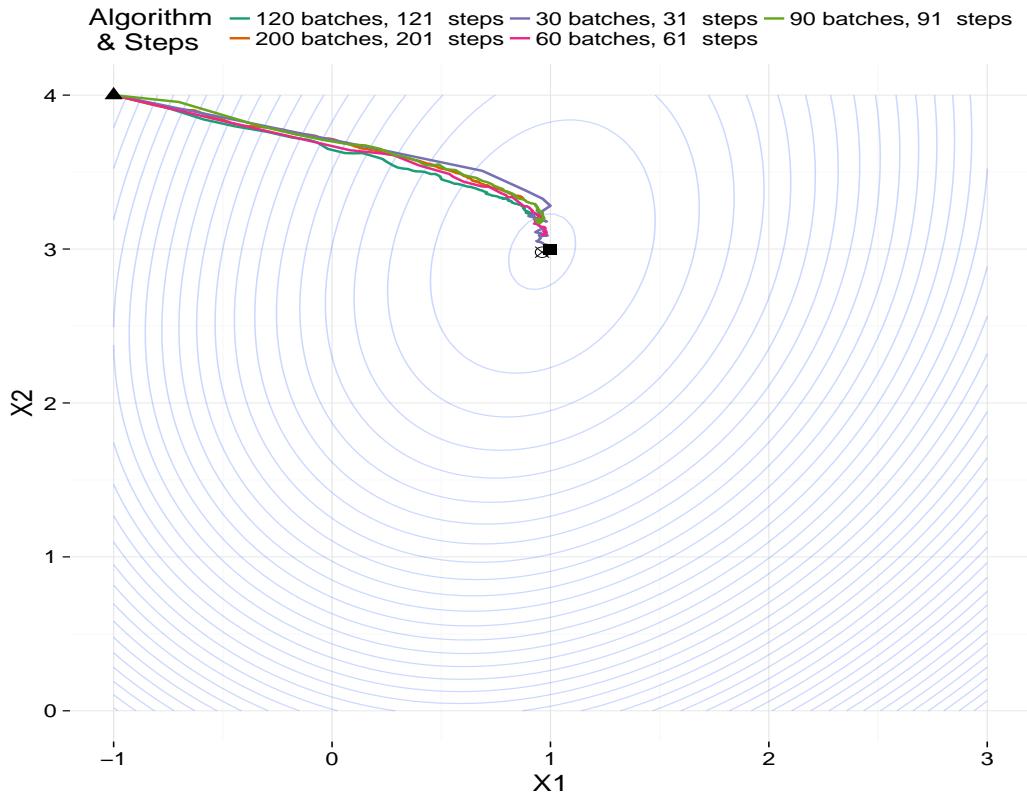


(a) $\beta_0 = (2, 2)$, liczba epok 10, warunek stopu: $\varepsilon = 10^{-6}$, długości kroków = $\frac{1}{50\sqrt{t}}$.

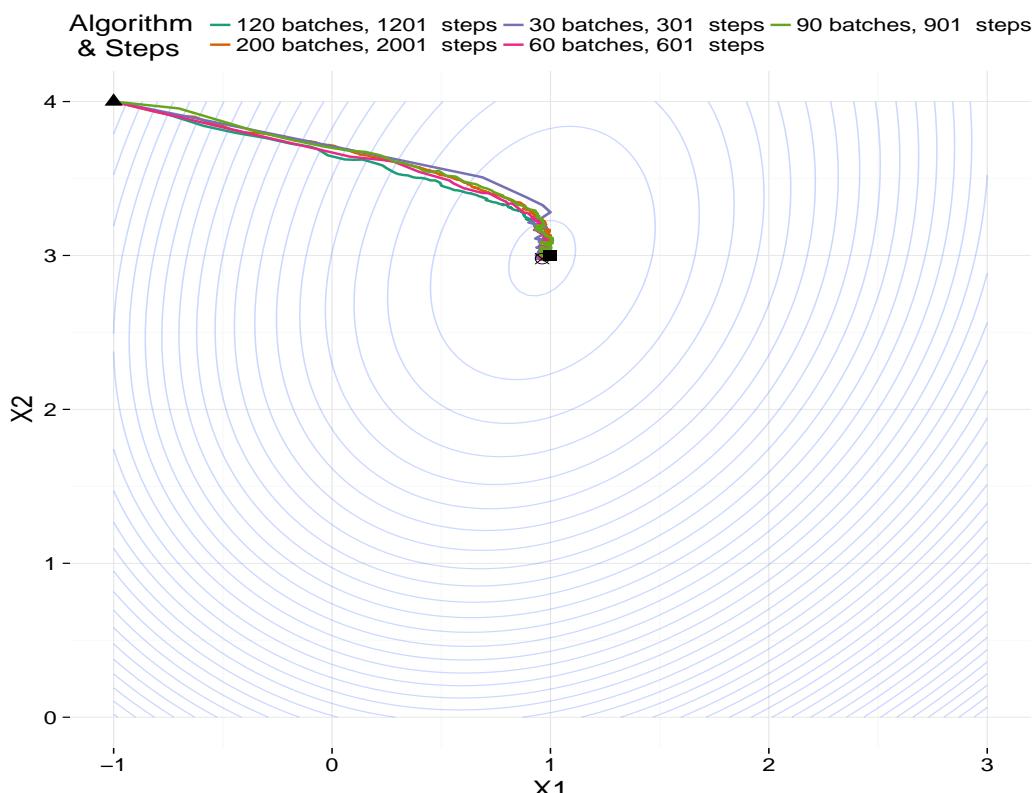


(b) $\beta_0 = (2, 2)$, liczba epok 5, warunek stopu: $\varepsilon = 10^{-5}$, długości kroków = $\frac{1}{50\sqrt{t}}$.

Rysunek 4.2: Porównanie trajektorii estymacji w modelu Coxa, wykorzystującym metodę stochastycznego spadku gradientu, w zależności od różnych podziałów zbioru początkowego na podzbiory. Elipsami przedstawiono warstwice częściowej funkcji log-wiarogodności wyliczone na bazie wszystkich obserwacji. Trójkątem zaznaczono punkt startowy algorytmu, kwadratem teoretyczne maksimum, zaś gwiazdką współczynniki uzyskane za pomocą funkcji `coxph()`.

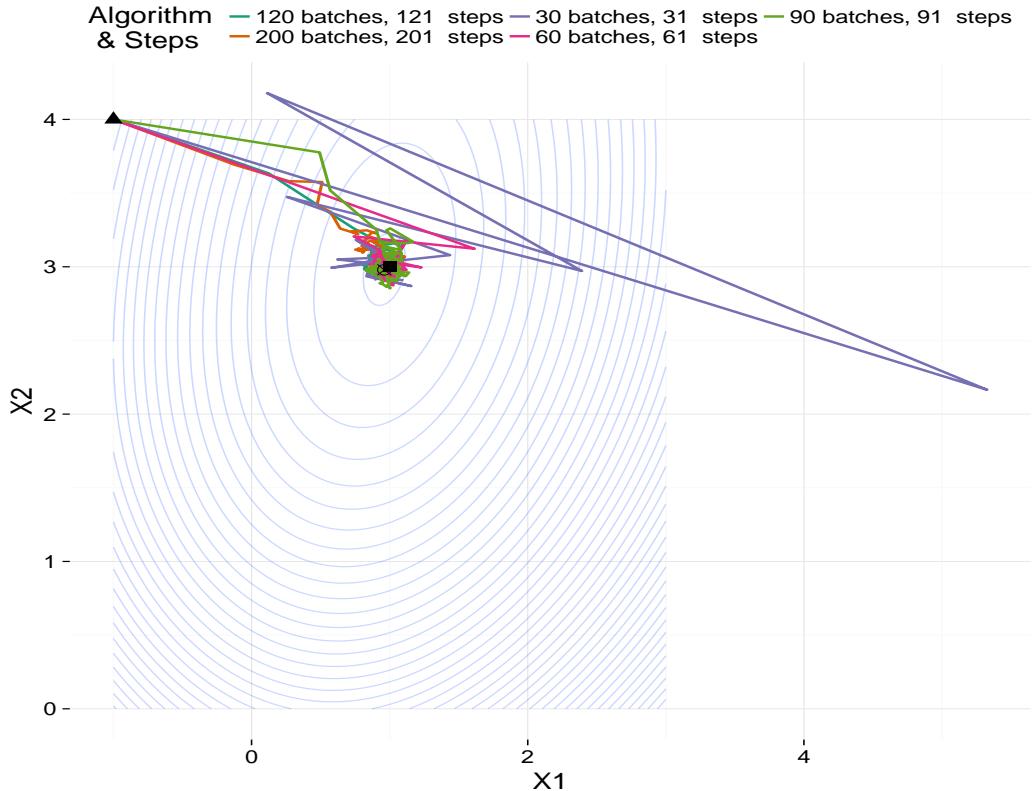


(a) $\beta_0 = (-1, 4)$, liczba epok 1, warunek stopu: $\varepsilon = 10^{-6}$, długości kroków = $\frac{1}{100\sqrt{t}}$.

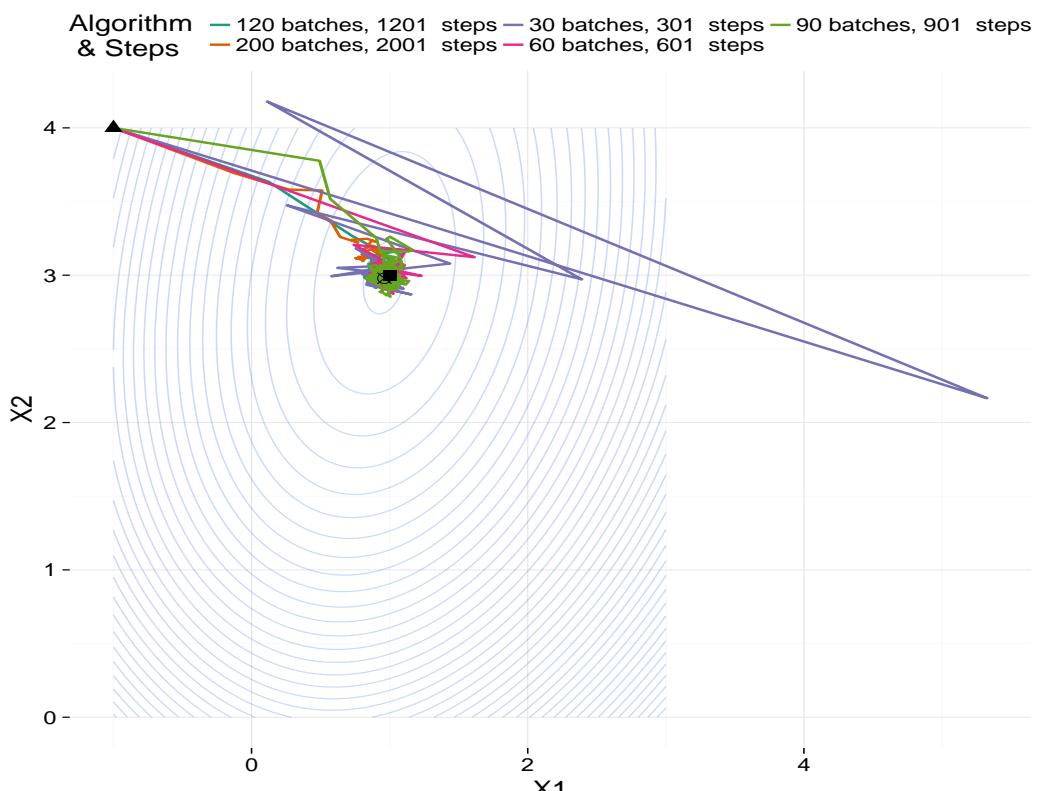


(b) $\beta_0 = (-1, 4)$, liczba epok 10, warunek stopu: $\varepsilon = 10^{-5}$, długości kroków = $\frac{1}{100\sqrt{t}}$.

Rysunek 4.3: Porównanie trajektorii estymacji w modelu Coxa, wykorzystującym metodę stochastycznego spadku gradientu, w zależności od różnych podziałów zbioru początkowego na podzbiory. Elipsami przedstawiono warstwice częściowej funkcji log-wiarogodności wyliczone na bazie wszystkich obserwacji. Trójkątem zaznaczono punkt startowy algorytmu, kwadratem teoretyczne maksimum, zaś gwiazdką współczynniki uzyskane za pomocą funkcji `coxph()`.



(a) $\beta_0 = (-1, 4)$, liczba epok 1, warunek stopu: $\varepsilon = 10^{-6}$, długości kroków = $\frac{1}{20\sqrt{t}}$.



(b) $\beta_0 = (-1, 4)$, liczba epok 10, warunek stopu: $\varepsilon = 10^{-5}$, długości kroków = $\frac{1}{20\sqrt{t}}$.

Rysunek 4.4: Porównanie trajektorii estymacji w modelu Coxa, wykorzystującym metodę stochastycznego spadku gradientu, w zależności od różnych podziałów zbioru początkowego na podzbiory. Elipsami przedstawiono warstwice częściowej funkcji log-wiarogodności wyliczone na bazie wszystkich obserwacji. Trójkątem zaznaczono punkt startowy algorytmu, kwadratem teoretyczne maksimum, zaś gwiazdką współczynniki uzyskane za pomocą funkcji `coxph()`.

Rozdział 5

Analiza danych genomicznych

*The key is to understand genomics
to improve cancer care.
The Cancer Genome Atlas Study*

Niniejszy rozdział poświęcony jest analizie danych genomicznych. W podrozdziale 5.1 przedstawiono pokrótkie genetyczne podstawy nowotworzenia, które szerzej opisane są w cytowanej literaturze. W podrozdziale 5.2 opisano schemat *The Cancer Genome Atlas* (TCGA), badania z którego zaczerpnięto dane do analizy biostatystycznej. Analiza przeżycia polegająca na zastosowaniu modelu Coxa, w którym estymacja współczynników odbywa się za pomocą metody stochastycznego spadku gradientu (opisana w rozdziale 4), zaprezentowana została wraz z komentarzami w podrozdziale 5.3.

W ramach analizy starano się ocenić wpływ mutacji poszczególnych genów na przeżywalność pacjentów. Celem było uzyskanie odpowiedzi na pytanie czy istnieją geny, których mutacje powodują, że ryzyko wystąpienia zdarzenia jakim jest zgon w wyniku choroby nowotworowej jest większe niż u pacjentów bez mutacji w danym genie. Aby zweryfikować hipotezę na tak obszernych danych zdecydowano się zastosować metodę strumieniowej estymacji współczynników w modelu Coxa przy pomocy stochastycznego spadku gradientu. Jej stworzenie było głównym założeniem pracy. Tradycyjna metoda zaimplementowana w funkcji `coxph()` w pakiecie `survival` (Therneau, 2015) nie była wystarczająca, gdyż nie jest przystosowana do tak dużych zbiorów danych jak te z TCGA. Z racji zbyt dużej liczby genów (przekraczającą kilkanaście tysięcy), które mogłyby być zmutowane i wzięte do analizy jako zmienna objaśniająca czas do zdarzenia, niemożliwe byłoby wyestymowanie współczynników modelu standardową metodą.

Wielką zaletą modelu Coxa z estymacją przy użyciu metody stochastycznego spadku gradientu jest możliwość zastosowania go do zbiorów danych o rozmiarze zmiennych znacznie przekraczającym wymiar obserwacji. Zaprezentowana w rozdziale 4.2 implementacja pozwala w skończonym czasie znaleźć współczynniki modelu odpowiadające za wpływ danych genów na śmiertelność pacjentów w chorobach nowotworowych. Jest to ważne narzędzie w analizach biostatystycznych danych dużej skali, które z powodzeniem może zostać wykorzystywane do analizy ludzkiego genomu, by w przyszłości poszerzać zrozumienie procesów mutacji zachodzących w ludzkim organizmie, dzięki któremu możliwe byłoby skuteczniejsze leczenie chorób nowotworowych.

5.1. Genetyczne podstawy nowotworzenia

Choroby nowotworowe stanowią drugą, po chorobach serca, przyczynę zachorowań i zgonów na świecie. Opracowania Polskiej Unii Onkologicznej wskazują na tendencję wzrostową liczby zachorowań na nowotwory i rokują na utrzymanie się jej do 2020 ([Podsiadły, 2011](#)).

Nowotwór jest *chorobą cyklu komórkowego* i oznacza ([Domagała, 2007](#))

nieprawidłową tkankę, która powstała z jednej komórki i rośnie jako następstwo zaburzeń dynamizmu i prawidłowego przebiegu cyklu komórkowego oraz zaburzeń różnicowania się komórki i komunikacji wewnątrzkomórkowej, międzykomórkowej i pozakomórkowej jej klonalnego potomstwa.

Badania nad transformacją nowotworową wykazały, że nowotwory powstają jako wynik mutacji w DNA komórki somatycznej, które poprzez akumulację wywołują utratę kontroli nad proliferacją, wzrostem i różnicowaniem ([Kozłowska and Łaczmańska, 2010](#)).

Proces tworzenia nowotworu (**kancerogeneza**) jest wielostopniowy, a zmiany w nim nasilają się w miarę pogłębiania się niestabilności genetycznej ([Domagała, 2007](#)). Transformacja nowotworowa następuje w wyniku zmian powstałych w obrębie czterech różnych kategorii genów, które wpływają na proliferację i różnicowanie komórek ([Podsiadły, 2011](#)). Czynniki genetyczne zaangażowane w systemy naprawy DNA, a także w proliferację i różnicowanie komórek ([Janik-Papis and Błasiak, 2010b](#)) dzielą się na:

- *geny regulujące naprawę uszkodzonego DNA* - mechanizmy szybkiej naprawy DNA zapobiegają przed mutacjami genów odpowiedzialnych m. in. za proliferację i różnicowanie się komórek. Geny biorące udział w naprawie DNA nie są onkogenne, natomiast mutacje w ich obrębie mogą ułatwić transformację nowotworową oraz podwyższają ryzyko utrwalenia się zmian w pozostałych grupach i dlatego mają podstawowe znaczenie dla integracji genomu.
- *geny supresorowe* (antyonkogeny) - kontrolują cykl komórkowy. Są punktami kontrolnymi pomiędzy fazami cyklu komórkowego, czynnikami negatywnej kontroli podziałów oraz zapobiegają niekontrolowanym podziałom komórkowym.
- *protoonkogeny* - potencjalnie zdolne do wyzwolenia procesu transformacji nowotworowej. Uwarunkowana mutacją zmiana ich ekspresji sprawia, że przekształcają się w onkogeny, czyli geny bezpośrednio aktywujące transformację nowotworową.
- *geny regulujące apoptozę* (naturalny proces zaprogramowanej śmierci komórki w organizmach wielokomórkowych) - zahamowanie procesu apoptozy wskutek mutacji czynników indukujących wydłuża okres życia komórek, zwiększając tym samym populację komórek narażonych na działanie karcynogenów i prawdopodobieństwo wystąpienia mutacji w komórce.

Zmiany genetyczne w komórkach zachodzą pod wpływem działania czynników mutagennych, do których można zaliczyć: promieniowanie UV (czynniki fizyczne), substancje obecne w dymie papierosowym i spalinach samochodowych, azbest, promieniowanie jonizujące, temperatura i niektóre metale ciężkie (czynniki chemiczne), wirusy, toksyny bakteryjne i pasożytnicze czy pośrednie produkty przemiany materii (czynniki biologiczne) ([Janik-Papis and Błasiak, 2010a](#)).

5.2. Projekt The Cancer Genome Atlas

Do analizy wykorzystano dane z badania *The Cancer Genome Atlas Broad Institute of MIT and Harvard (2014)*. The Cancer Genome Atlas (TCGA) to kompleksowy projekt o skoordinowanych wysiłkach, mający na celu przyspieszenie zrozumienia molekularnych podstaw nowotworu. Ma się to odbywać poprzez stosowanie technologii analizy danych dużej skali do udostępnianych danych genetycznych i zsekwencjonowanego genomu tkanek nowotworowych. TCGA to inicjatywa *National Cancer Institute (NCI)* oraz *National Human Genome Research Institute (NHGRI)*, czyli 2 z pośród 27 instytutów i centrów Narodowego Instytutu Zdrowia w Departamencie Zdrowia i Opieki Społecznej Stanów Zjednoczonych.

Jak podaje [Tomczak et al. \(2015\)](#), w biologii jest znane ponad 200 różnych form nowotworu i jeszcze więcej ich podtypów. Każdy z nich wywołany jest nieprawidłowościami w DNA, które sprawiają, że komórki namnażają się w sposób niekontrolowany. Identyfikując zmiany w kompletnym zbiorze DNA (genomie) dla każdego nowotworu i wiedząc jak te zmiany wpływają na jego rozwój, możliwe będzie skuteczne zapobieganie nowotworom, wczesne ich wykrywanie i leczenie.

Nadrzędnym celem TCGA jest poprawienie naszej zdolności do diagnozowania, leczenia i profilaktyki nowotworów. TCGA aby osiągnąć ten cel w sposób naukowo rygorystyczny, opracowało i przetestowało strukturę badawczą konieczną do systematycznego badania całego spektrum zmian genomu, zawartych w ponad 20 rodzajach nowotworów.

Dzięki projektowi TCGA społeczność naukowców walczących z rakiem może korzystać z danych uzyskanych z sekwencjonowania komórek pochodzących z tkanek nowotworowych zebranych przez *Cancer Genomics Hub (CGHub)* i *Genome Data Analysis Centers (GDACs)*. Te i wiele więcej instytucji opisanych szerzej jest w [Tomczak et al. \(2015\)](#). TCGA *Genome Data Analysis Centers* składają się z 7 instytucji, a jedna z nich *Broad Institute, Cambridge* udostępnia dane oraz wyniki swoich analiz poprzez portal *Firehose Broad GDAC* (<http://gdac.broadinstitute.org/>). Portal udostępnia dane dla 38 kohort związanych z występującym typem nowotworu. Więcej na temat kohort i danych z TCGA można znaleźć w [Chin et al. \(2011b\)](#), [Chin et al. \(2011a\)](#), [Medicine \(2015\)](#) czy [Tomczak et al. \(2015\)](#).

Dane z TCGA pobrano przy użyciu pakietu RTCGA i umieszczono w pakietach ([Kosiński, 2015b,c](#); [Kosiński and Biecek, 2015](#))

- `RTCGA.clinical`, zawierającym dane kliniczne o pacjentach,
- `RTCGA.mutations`, zawierającym dane o występujących w genach mutacjach

oraz zostały one wykorzystane w następnym rozdziale do sprawdzenia, czy występowanie mutacji w danym genie ma wpływ na przeżywalność pacjentów dotkniętych nowotworem.

Stworzone pakiety można pobrać z repozytorium pakietów bioinformatycznych Bioconductor oraz wyświetlić ich samouczek poleceniami

```
source("https://bioconductor.org/biocLite.R")
biocLite("RTCGA")
biocLite("RTCGA.clinical")
biocLite("RTCGA.mutations")
browseVignettes("RTCGA")
```

Pakiety zawierające inne dane z tego badania można zainstalować w \mathcal{R} polecienniem

```
RTCGA::installTCGA()
```

5.3. Analiza wpływu mutacji genów na czas życia

Do analizy badającej wpływ występowania mutacji genów na czas przeżycia wykorzystano dane kliniczne i dane o występujących u pacjentów mutacjach genetycznych. Starano się wykorzystać dane ze wszystkich 38 dostępnych kohort nowotworowych z badania *The Cancer Genome Atlas* (TCGA), jednak nie dla wszystkich kohort umieszczone w badaniu dane o mutacjach. Część wspólną nazw kohort zawierających komplet informacji wygenerowano dzięki wywołaniu (kody źródłowe funkcji z analizy zawarte są w Dodatku A.3)

```
(extractCohortIntersection() -> cohorts)
```

```
[1] "ACC"      "BLCA"     "BRCA"     "CESC"     "CHOL"     "COAD"
[7] "COADREAD" "DLBC"     "ESCA"     "GBM"      "GBMLGG"   "HNSC"
[13] "KICH"     "KIPAN"    "KIRC"     "KIRP"     "LAML"     "LGG"
[19] "LIHC"     "LUAD"     "LUSC"     "OV"       "PAAD"     "PCPG"
[25] "PRAD"     "READ"     "SARC"     "SKCM"     "STAD"     "STES"
[31] "TGCT"     "THCA"     "UCEC"     "UCS"      "UVM"
```

Następnie, mając tak otrzymanych 35 kohort nowotworowych uzyskano dane o statusie pacjenta (śmierć bądź cenzurowanie) oraz czasie spędzonym pod obserwacją dzięki funkcji

```
head(extractSurvival(cohorts) -> survivalData)
```

	bcr_patient_barcode	patient.vital_status	times
ACC.1	TCGA-OR-A5J1	1	1355
ACC.2	TCGA-OR-A5J2	1	1677
ACC.3	TCGA-OR-A5J3	0	1942
ACC.4	TCGA-OR-A5J4	1	423
ACC.5	TCGA-OR-A5J5	1	365
ACC.6	TCGA-OR-A5J6	0	2428

Dane o mutacjach występujących wśród tkanek nowotworowych kolejnych pacjentów

```
extractMutations(cohorts, 0.02) -> mutationsData
```

```
mutationsData[c(1,2,5,6), c(1,4,56,100,207,801)]
```

	bcr_patient_barcode	A2ML1	ALMS1	ATP2B2	CNTNAP4	PLEC
1	TCGA-02-0003-01	0	1	0	0	0
2	TCGA-02-0033-01	0	0	0	0	0
5	TCGA-02-2470-01	0	0	0	0	0
6	TCGA-02-2483-01	0	0	1	0	0

gdzie wybrano jedynie te geny, których mutacja dotyczyła co najmniej 2 % pacjentów mających zarówno dane kliniczne jak i dane o występujących mutacjach w genach.

Mając tak otrzymane dwa zbiory danych, połączono informacje kliniczne z informacjami o mutacjach dzięki przypisaniem do pacjentów i ich próbek kodów `bcr_patient_barcode`, by ostatecznie podzielić zbiór pacjentów na 100 losowo utworzonych grup.

```
set.seed(4561)
```

```
prepareCoxDataSplit(mutationsData, survivalData, groups = 100) -> coxData_split
head(coxData_split[[1]][c(1,10), c(210,302,356,898,911,1092:1093)])
```

	COL14A1	DOCK9	FASN	SEMA5A	SHPRH	patient.vital_status	times
81	0	0	0	0	0	1	7
1068	1	0	0	1	0	1	1171

Niezbędną formułę modelu potrzebną do sprecyzowania, które geny (a pozostało ich 1091) należy uwzględnić w modelu uzyskano dzięki pomocniczej funkcji

```
prepareForumlaSGD(coxData_split) -> formulaSGD
```

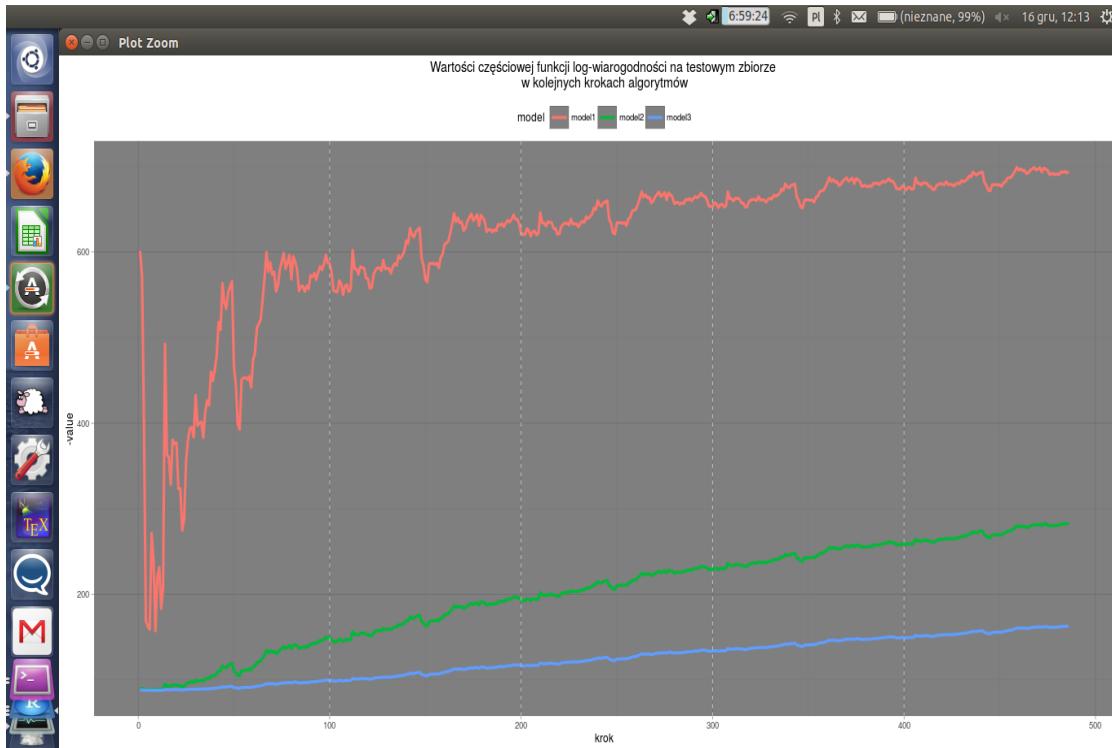
Ostatecznie dla 6085 pacjentów, którzy posiadali informacje o występujących mutacjach, oraz wśród których odnotowano komplet i poprawność danych klinicznych dotyczących statusu i obserwowanego czasu przeżycia, wyliczono współczynniki modelu proporcjonalnych hazardów Coxa, z wykorzystaniem stochastycznego spadku gradientu do estymacji. Model dopasowano wielokrotnie z różnymi ciągami odpowiadającymi za długość kroku algorytmu. Dodatkowo badano różną ilość epok w algorytmie. Mając tak powstały kilka modeli wybrano ten, którego współczynniki dawały największą wartość funkcji częściowej log-wiarogodności, skonstruowaną w oparciu o niewykorzystane do uczenia obserwacje, pochodzące z 2 ostatnich zaobserwowanych podzbiorów. Wśród każdego z ciągów $1/t, 1/50\sqrt{t}, 1/100\sqrt{t}$, odpowiadających długościom kroków, wyznaczono współczynniki modelu po 5 epok, dzięki czemu możliwe było rozważanie postępu danego wariantu algorytmu również po 1, 2, 3 czy 4 epokach.

```
coxData_split[99:100] -> testCox
coxData_split[1:98] -> trainCox
coxphSGD(formulaSGD, data = trainCox, max.iter = 490) -> model_1_over_t
coxphSGD(formulaSGD, data = trainCox, max.iter = 490,
          learningRates = function(t){1/(50*sqrt(t))}) -> model_1_over_50sqrt_t
coxphSGD(formulaSGD, data = trainCox, max.iter = 490,
          learningRates = function(t){1/(100*sqrt(t))}) -> model_1_over_100sqrt_t
```

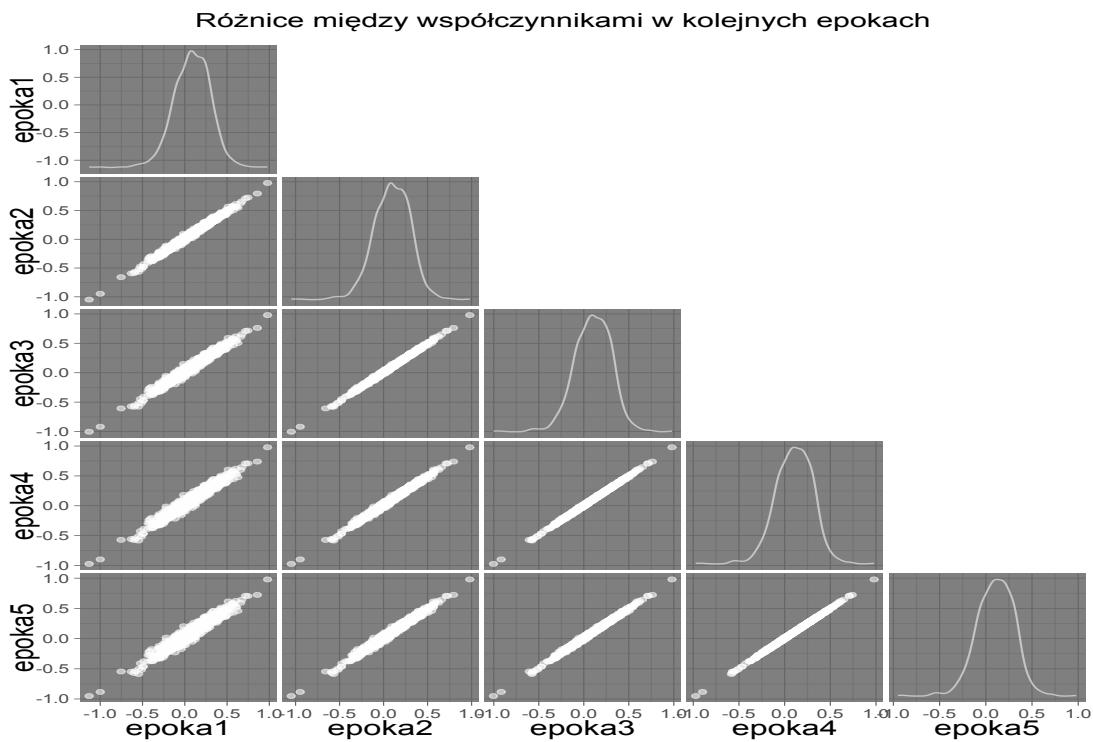
Po dopasowaniu modeli, sporządzono wykresy przedstawiające histogramy wartości współczynników modelu po każdej epoce, w rozróżnieniu na trzy typy ciągów odpowiedzialnych za długość kroku. Histogramy przedstawiono na Rysunkach 5.4 - 5.6, zaś na Rysunku 5.7 przedstawiono ich porównanie na wspólnych osiach. Największe zróżnicowanie wśród współczynników modelu można zaobserwować w modelu `model_1_over_t`, w którym ciąg odpowiadający za długość kroku algorytmu to $1/t$. W pozostałych modelach współczynniki grupują się bliżej 0 i rzadko co do modułu są większe od 0.5. W przypadku modelu `model_1_over_t` wartości współczynników bywają co do modułu niekiedy większe od 1. Oznaczać to może, że przy takim ciągu długości kroków, algorytm jest w stanie uwypuklić rolę genów mających istotny wpływ na czas przeżycia pacjentów poprzez wyestymowanie większych wartości wśród odpowiednich współczynników. Różnice wśród wartości współczynników pomiędzy modelami najwyraźniej widać na Rysunku 5.7, gdzie w przypadku modelu `model_1_over_100sqrt_t` ($\alpha_{model3} = 1/100\sqrt{t}$) pik w okolicach zera wskazuje na to, że mało genów można uznać za istotnie wpływające na czas przeżycia z powodu odpowiadających im współczynnikom bliskim 0. Nie oznacza to jednak, że w tym przypadku istnieją jakiekolwiek podstawy by uznać którykolwiek z modeli za gorszy.

Zdecydowano się uznać za najlepszy ten model, który da największą wartość częściowej funkcji log-wiarogodności na testowym zbiorze wyznaczonym przez dwa ostatnie zaobserwowane podzbiory. Do obliczenia częściowej funkcji log-wiarogodności wykorzystano istniejącą implementację z pakietu `survival`

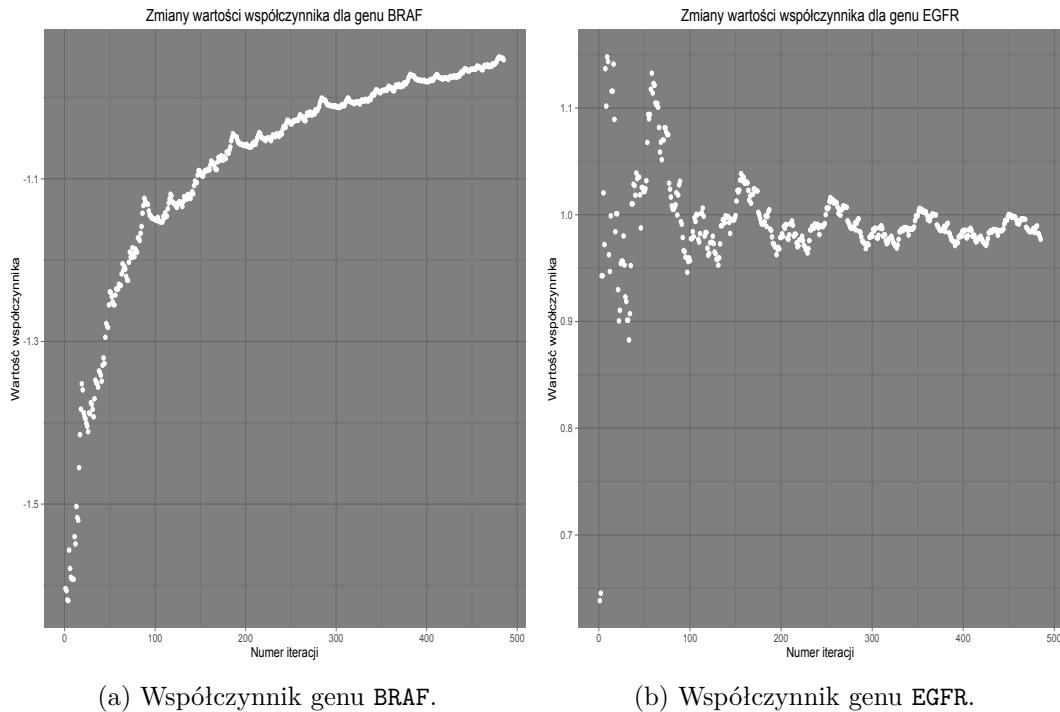
```
coxph_loglik <- function(beta, formula, data) {
  coxph(formula, init=beta, control=list('iter.max'=0), data =data)$loglik[2]
}
```



Rysunek 5.1: Zmiany wartości częściowej funkcji log-wiarododności, konstruowanej w oparciu o zbiór testowy, w kolejnych krokach trzech rozważanych algorytmów. Za najlepszy, maksymalizujący częściową funkcję log-wiarododności, uznano model z ciągiem $\alpha_{model1} = 1/t$.



Rysunek 5.2: Różnice wartości współczynników pomiędzy epokami.

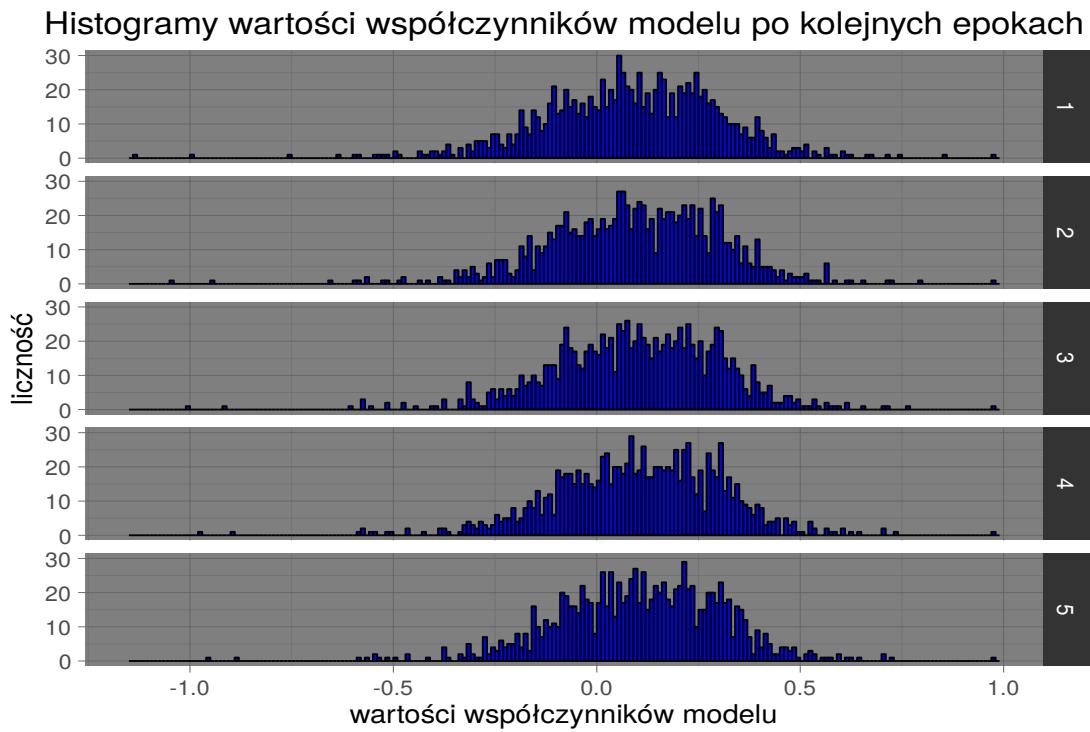


Rysunek 5.3: Zmiany wartości współczynników, w zależności od kroku iteracji wśród dwóch genów o największym i najmniejszym współczynniku w najlepszym modelu ($\alpha_{model1} = 1/t$).

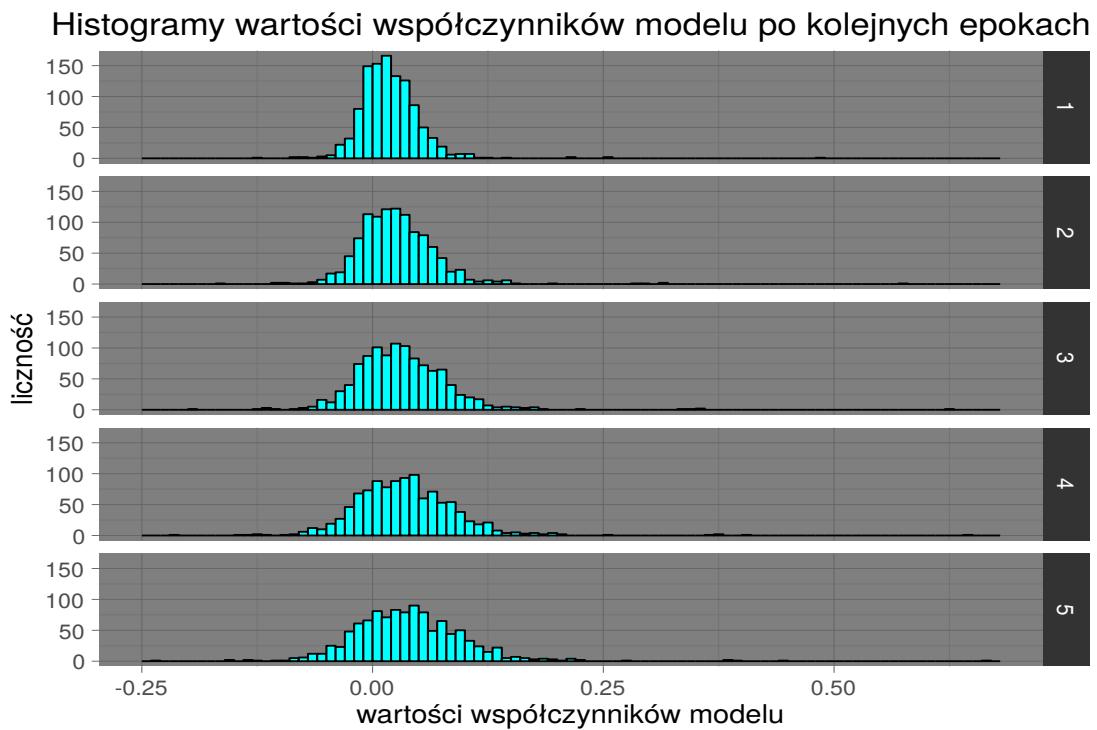
Rezultaty analizy przeżycia

Niemogliwym było sprawdzenie założeń modelu dotyczących proporcjonalności hazardu, gdyż zakładano napływaną postać danych (stąd podział danych na 100 grup). Przy takiej postaci pojawiania się danych ciężko także mówić o jakiekolwiek diagnostyce poprawności dopasowania modelu i dokładności otrzymanych współczynników. Nie stworzono teorii pozwalającej badać istotność statystyczną otrzymanych współczynników w modelu, jednak założono, że współczynniki dostatecznie odległe od 0 dotyczą genów, które można uznać za istotnie wpływające na czas życia pacjenta. Współczynniki dodatnie oznaczają zwiększenie hazardu pacjenta posiadającego mutację w danym genie, w stosunku do pacjentów nie posiadających tej mutacji. Analogicznie, współczynniki ujemne oznaczają zmniejszenie hazardu pacjenta posiadającego mutację w danym genie, w stosunku do pacjentów nie posiadających tej mutacji. Wzrost proporcji hazardu można otrzymać dla danego genu poprzez obłożenie współczynnika funkcją wykładniczą o wykładniku e .

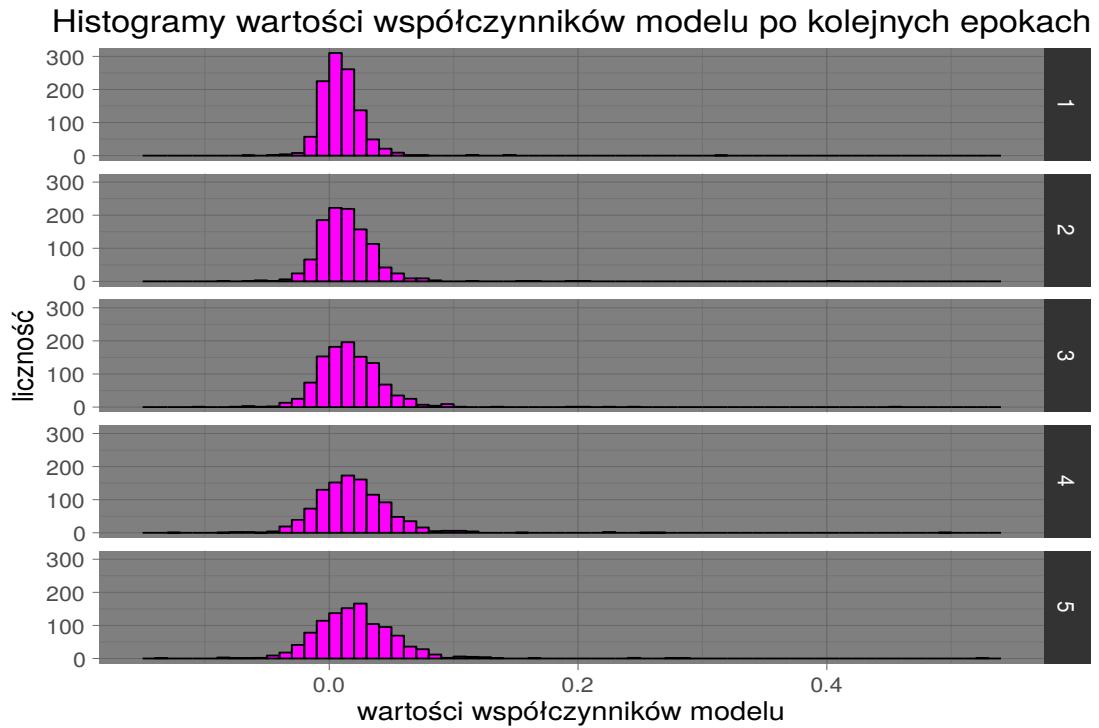
Wyniki estymacji wśród genów zawierających 40 największych, co do modułu, współczynników (pochodzących z modelu maksymalizującego częściową funkcję log-wiarogodności, konstruowaną na bazie dwóch ostatnich zaobserwowanych zbiorów) zawarto w Tabeli 5.1. Na Rysunku 5.1 ukazano jak zmieniały się wartości częściowej funkcji log-wiarogodności skonstruowanej w oparciu o zbiór testowy, w kolejnych krokach trzech rozważanych algorytmów. Za najlepszy, dający największe wartości częściowej funkcji log-wiarogodności wybrano model z ciągiem $\alpha_{model1} = 1/t$ (`model_1_over_t`). Sporządzono również wykres pokazujący zmiany wartości współczynników, w zależności od kroku iteracji wśród dwóch genów o największym i najmniejszym współczynniku (Rysunek 5.3). Ogólne różnice wartości współczynników pomiędzy epokami ukazano na Rysunku 5.2.



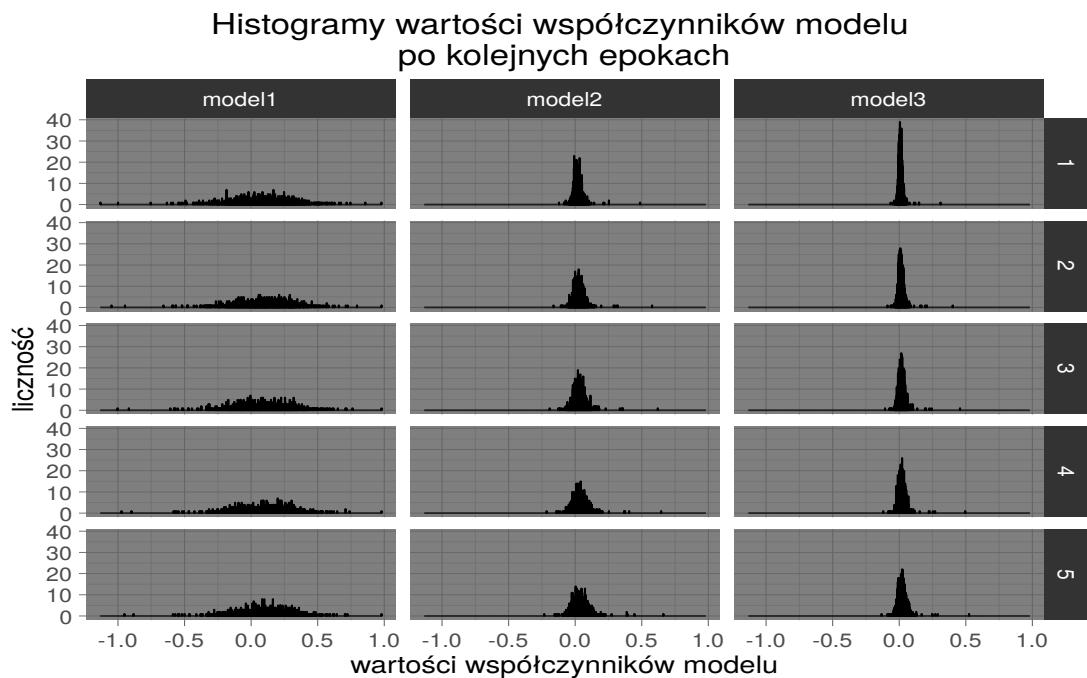
Rysunek 5.4: Histogram wartości współczynników w modelu Coxa proporcjonalnych hazardów po kolejnych epokach. Współczynniki uzyskano dzięki wykorzystaniu algorytmu stochastycznego spadku gradientu, w którym ciąg odpowiadający za długość kroku to $\alpha_t = 1/t$.



Rysunek 5.5: Histogram wartości współczynników w modelu Coxa proporcjonalnych hazardów po kolejnych epokach. Współczynniki uzyskano dzięki wykorzystaniu algorytmu stochastycznego spadku gradientu, w którym ciąg odpowiadający za długość kroku to $\alpha_t = 1/50\sqrt{t}$.



Rysunek 5.6: Histogramy wartości współczynników w modelu Coxa proporcjonalnych hazardów po kolejnych epokach. Współczynniki uzyskano dzięki wykorzystaniu algorytmu stochastycznego spadku gradientu, w którym ciąg odpowiadający za długość kroku to $\alpha_t = 1/100\sqrt{t}$.



Rysunek 5.7: Porównanie histogramów wartości współczynników w modelu Coxa proporcjonalnych hazardów po kolejnych epokach. Współczynniki uzyskano dzięki wykorzystaniu algorytmu stochastycznego spadku gradientu, w którym kolejne ciągi odpowiadający za długość kroku to odpowiednio: $\alpha_{model1} = 1/t$, $\alpha_{model2} = 1/50\sqrt{t}$, $\alpha_{model3} = 1/100\sqrt{t}$.

	β	e^β
EGFR	0.98	2.66
ATP2B2	0.72	2.06
TP53	0.71	2.03
NDST4	0.70	2.02
CHRM2	0.64	1.90
LRRC4C	0.62	1.87
CDKN2A	0.62	1.85
C3	0.59	1.81
GLI2	0.59	1.81
KIAA1409	0.58	1.78
KEAP1	0.57	1.76
Unknown	0.55	1.74
TMEM132D	0.54	1.71
ZNF804B	0.53	1.71
C15orf2	0.53	1.70
LTBP2	0.53	1.69
NTRK3	0.52	1.69
PCDH9	0.51	1.67
PRKD1	0.51	1.67
SMAD4	0.49	1.64
ZFAT	0.49	1.63
GRIA4	0.48	1.62
GPR133	0.48	1.62
FAM5C	0.48	1.62
TRHDE	0.48	1.61
PCDH7	0.47	1.60
PDGFRA	0.47	1.60
LRRIQ1	0.47	1.59
MUC16	0.46	1.59
ITGA8	0.46	1.59
ZMYM3	-0.46	0.63
IDH1	-0.50	0.61
CTCF	-0.52	0.60
DNAH17	-0.54	0.58
DLGAP2	-0.54	0.58
BCORL1	-0.55	0.58
PLXNC1	-0.57	0.57
CDH1	-0.59	0.56
FLRT2	-0.88	0.41
BRAF	-0.95	0.39

Tabela 5.1: Współczynniki oraz stosunki hazardów (e^β) w modelu proporcjonalnych hazardów Coxa w modelu maksymalizującym częściową funkcję log-wiarodności na zbiorze testowym.

Podsumowanie

Praca przedstawia zastosowanie algorytmu numerycznej optymalizacji metodą stochastycznego spadku gradientu do wyznaczenia współczynników w modelu proporcjonalnych hazardów Coxa, opartych na analitycznej metodzie wyznaczania estymatorów metodą największej wiarygodności. Zaprezentowano zastosowanie tego algorytmu do napływających danych, wyliczając kolejne kroki optymalizacji w algorytmie w oparciu o częściową funkcję log-wiarogodności, konstruowaną na bazie obecnie zaobserwowanego podzbioru obserwacji. Zaprezentowane podejście jest nowatorskie.

Opisano matematyczne własności estymatorów największej wiarygodności, przedstawiono model Coxa wraz z najważniejszymi jego cechami, scharakteryzowano algorytm stochastycznego spadku gradientu oraz jego różnice w stosunku do algorytmów spadku gradientu, zaimplementowano algorytm numerycznej optymalizacji metodą stochastycznego spadku gradientu do wyznaczenia współczynników w modelu proporcjonalnych hazardów Coxa oraz przeprowadzono analizę rzeczywistych danych genomicznych, wykorzystując stworzony algorytm. Pracę wzbogacono o symulacje zbieżności procesu optymalizacji funkcji log-wiarogodności w modelu regresji logistycznej, które przygotowane zostały dla algorytmów spadku gradientu rzędu I, spadku gradientu rzędu II oraz stochastycznego spadku gradientu rzędu I. Dzięki temu możliwe było zobrazowanie różnic w opisywanych metodach numerycznej optymalizacji. Dodatkowo przeprowadzono symulacje przy sztucznie wygenerowanych danych do analizy przeżycia z rozkładu Weibulla, pozwalające zobrazować proces zbieżności nowo wprowadzonego algorytmu numerycznej optymalizacji metodą stochastycznego spadku gradientu do wyznaczenia współczynników w modelu proporcjonalnych hazardów Coxa.

Proces estymacji w modelu proporcjonalnych hazardów Coxa, z wykorzystaniem metody stochastycznego spadku gradientu, w sytuacji napływających danych wygląda na stabilny i zbiega w okolice bliskie do teoretycznego optimum, niekiedy nawet już po jednej epoce. Przy dobrze dobranych parametrach optymalizacji, proces może być z powodzeniem wykorzystywany do znajdowania współczynników modelu. Kluczowym momentem, mającym wpływ na powodzenie optymalizacji, jest wybór ciągu odpowiedzialnego za dobór długości kroków, więc zaleca się symulacje badające różne ciągi oraz wybranie tego ciągu, który w większości przypadków daje stabilne, jednorodne współczynniki, maksymalizujące częściową funkcję log-wiarogodności, konstruowaną w oparciu o zbiór testowy.

Do analizy genomicznej wykorzystano środowisko statystyczne \mathcal{R} , a wykorzystane funkcje, zaimplementowane algorytmy i przygotowaną dokumentację do stworzonego w trakcie pracy pakietu `coxphSGD` zamieszczono w Dodatku A. W analizie badano wpływ występowania mutacji w poszczególnych genach na czas przeżycia pacjentów, dla których dane kliniczne i dane o występowaniu mutacji zaczerpnięto z badania *The Cancer Genome Atlas*. Zaprezentowano skuteczność badanego modelu oraz wyniki 40 genów o największych, co do modułu, współczynnikach. Dzięki temu wskazano geny odpowiedzialne za zwiększenie ryzyka zgonu w trakcie choroby nowotworowej.

Dodatek A

Wykorzystane narzędzia, dokumentacja i kody pakietu \mathcal{R} użyte w pracy

W pracy wykorzystano środowisko statystyczne \mathcal{R} (Biecek, 2011; Gągolewski, 2014; R Core Team, 2013) do implementacji algorytmów, przeprowadzenia symulacji oraz wykonania analizy danych genetycznych. Do przetwarzania danych wykorzystano pakiety `dplyr` (Wickham and Francois, 2015), `tidyr` (Wickham, 2015), `reshape2` (Wickham, 2007) oraz `data.table` (Dowle et al., 2015). Do tworzenia dokumentacji oraz struktury pakietu `coxphSGD` (Kosiński, 2015a) użyto pakietów `roxygen2` (Wickham et al., 2015), `rmarkdown` (Allaire et al., 2015), `knitr` (Xie, 2014a,b, 2015), `assertthat` (Wickham, 2013) oraz pakietu `rlp` (Cheng and Xie, 2014). Przejrzystość kodu wspierał pakiet `magrittr` (Bache and Wickham, 2014), zaś do tworzenia wizualizacji wykorzystano pakiet `ggplot2` (Wickham, 2009). Dane do analizy pobrano dzięki pakietowi `RTCGA` (Kosiński and Biecek, 2015)) oraz umieszczone je w pakietach `RTCGA.clinical` (Kosiński, 2015b) oraz `RTCGA.mutations` (Kosiński, 2015c).

A.1. Model proporcjonalnych hazardów Coxa

```
checkArguments <- function(formula, data, learningRates,
                           beta_0, epsilon) {
  assert_that(is.list(data) & length(data) > 0)
  assert_that(length(unique(unlist(lapply(data, ncol)))) == 1)
  assert_that(is.function(learningRates))
  assert_that(is.numeric(epsilon, beta_0))
  if (length(beta_0) == 1) {
    beta_0 <- rep(beta_0, as.character(formula)[3] %>%
      strsplit("\\"+) %>%
      unlist %>%
      length)
  }
  return(beta_0)
}
```

```

prepareBatch <- function(formula, data) {
  # Parameter identification as in 'survival::coxph()'.
  Call <- match.call()
  indx <- match(c("formula", "data"),
                 names(Call), nomatch = 0)
  if (indx[1] == 0)
    stop("A formula argument is required")
  temp <- Call[c(1, indx)]
  temp[[1]] <- as.name("model.frame")

  mf <- eval(temp, parent.frame())
  Y <- model.extract(mf, "response")

  if (!inherits(Y, "Surv"))
    stop("Response must be a survival object")
  type <- attr(Y, "type")

  if (type != "right" && type != "counting")
    stop(paste("Cox model doesn't support \\"", type, "\\" survival data",
              sep = ""))
  # collect times, status, variables and reorder samples
  # to make the algorithm more clear to read and track
  cbind(event = unclass(Y)[,2], # 1 indicates event, 0 indicates cens
        times = unclass(Y)[,1],
        mf[, -1]) %>%
  arrange(times)
}

coxph_loglik <- function(beta, formula, data) {
  coxph(formula, init=beta, control=list('iter.max'=0), data =data)$loglik[2]
}
calculate_outer_cox_3 <- function(dCox){
  ## contours
  outer_res <- outer(seq(-1,3, length = 100),
                      seq(0,4, length = 100),
                      Vectorize( function(beta1,beta2){
                        coxph_loglik(beta=c(beta1,beta2), Surv(time, status)~x.1+x.2-1, dCox)
                      } )
  )
  outer_res_melted <- melt(outer_res)
  outer_res_melted$Var1 <- as.factor(outer_res_melted$Var1)
  levels(outer_res_melted$Var1) <- as.character(seq(-1,3, length = 100))
  outer_res_melted$Var2 <- as.factor(outer_res_melted$Var2)
  levels(outer_res_melted$Var2) <- as.character(seq(0,4, length = 100))
  outer_res_melted$Var1 <- as.numeric(as.character(outer_res_melted$Var1))
  outer_res_melted$Var2 <- as.numeric(as.character(outer_res_melted$Var2))
  return(outer_res_melted)
}

```

```

calculate_outer_cox_3(dCox) -> outerCox
simulateCoxSGD <- function(dCox = dCox, learningRates = function(x){1/x},
                           epsilon = 1e-03, beta_0 = c(0,0), max.iter = 100){

  sample(1:90, size = 10^4, replace = TRUE) -> group
  split(dCox, group) -> dCox_splitted
  coxphSGD(Surv(time, status)~x.1+x.2, data = dCox_splitted,
            epsilon = epsilon, learningRates = learningRates,
            beta_0 = beta_0, max.iter = max.iter*90) -> estimates

  sample(1:60, size = 10^4, replace = TRUE) -> group
  split(dCox, group) -> dCox_splitted
  coxphSGD(Surv(time, status)~x.1+x.2, data = dCox_splitted,
            epsilon = epsilon, learningRates = learningRates,
            beta_0 = beta_0, max.iter = max.iter*60) -> estimates2

  sample(1:120, size = 10^4, replace = TRUE) -> group
  split(dCox, group) -> dCox_splitted
  coxphSGD(Surv(time, status)~x.1+x.2, data = dCox_splitted,
            epsilon = epsilon, learningRates = learningRates,
            beta_0 = beta_0, max.iter = max.iter*120) -> estimates3

  sample(1:200, size = 10^4, replace = TRUE) -> group
  split(dCox, group) -> dCox_splitted
  coxphSGD(Surv(time, status)~x.1+x.2, data = dCox_splitted,
            epsilon = epsilon, learningRates = learningRates,
            beta_0 = beta_0, max.iter = max.iter*200) -> estimates4

  sample(1:30, size = 10^4, replace = TRUE) -> group
  split(dCox, group) -> dCox_splitted
  coxphSGD(Surv(time, status)~x.1+x.2, data = dCox_splitted,
            epsilon = epsilon, learningRates = learningRates,
            beta_0 = beta_0, max.iter = max.iter*30) -> estimates5

  t(simplify2array(estimates$coefficients)) %>%
    as.data.frame() -> df1
  t(simplify2array(estimates2$coefficients)) %>%
    as.data.frame() -> df2
  t(simplify2array(estimates3$coefficients)) %>%
    as.data.frame() -> df3
  t(simplify2array(estimates4$coefficients)) %>%
    as.data.frame() -> df4
  t(simplify2array(estimates5$coefficients)) %>%
    as.data.frame() -> df5

  df1 %>%
  mutate(version = paste("90 batches,", nrow(df1), " steps")) %>%

```

```

bind_rows(df2 %>%
            mutate(version = paste("60 batches,", nrow(df2), " steps"))) %>%
bind_rows(df3 %>%
            mutate(version = paste("120 batches,", nrow(df3), " steps"))) %>%
bind_rows(df4 %>%
            mutate(version = paste("200 batches,", nrow(df4), " steps"))) %>%
bind_rows(df5 %>%
            mutate(version = paste("30 batches,", nrow(df5), " steps))) -> d2ggplot

return(list(d2ggplot = d2ggplot, est1 = estimates, est2 = estimates2,
           est3 = estimates3, est4 = estimates4, est5 = estimates5))

}

simulateCoxSGD(dCox, learningRates = function(x){1/(100*sqrt(x))},
                max.iter = 10, epsilon = 1e-5) -> d2ggplot
d2ggplot -> backpack
d2ggplot <- d2ggplot$d2ggplot
beta_0 = c(0,0)
solution = c(1,3)

pdf(file = "b_0_0_iter_10_e-5_100sqrt_878.pdf", width = 10, height = 10)

ggplot() +
  stat_contour(aes(x=outerCox$Var1,
                    y=outerCox$Var2,
                    z=outerCox$value),
               bins = 40, alpha = 0.25) +
  geom_path(aes(d2ggplot$V1, d2ggplot$V2, group = d2ggplot$version,
                colour = d2ggplot$version), size = 1) +
  theme_bw(base_size = 20) +
  theme(panel.border = element_blank(),
        legend.key = element_blank(), legend.position = "top") +
  scale_colour_brewer(palette="Dark2", name = 'Algorithm \n & Steps') +
  geom_point(aes(x = beta_0[1], y = beta_0[2]), col = "black", size = 4, shape = 17) +
  geom_point(aes(x = solution[1], y = solution[2]), col = "black", size = 4, shape = 15) +
  geom_point(aes(x = summary(coxph(Surv(time, status)~x.1+x.2, data = dCox))$coeff[1,1],
                 y = summary(coxph(Surv(time, status)~x.1+x.2, data = dCox))$coeff[2,1]),
             col = "black", size = 4, shape = 13) +
  xlab("X1") + ylab("X2") +
  guides(col = guide_legend(ncol = 3))

dev.off()

```

A.2. Dokumentacja pakietu coxphSGD

coxphSGD

December 9, 2015

coxphSGD

Stochastic Gradient Descent log-likelihood estimation in Cox proportional hazards model

Description

Function coxphSGD estimates coefficients using stochastic gradient descent algorithm in Cox proportional hazards model.

Usage

```
coxphSGD(formula, data, learningRates = function(x) {  
  1/x  
, beta_0 = 0, epsilon = 1e-05, max.iter = 500)
```

Arguments

formula	a formula object, with the response on the left of a <code>~</code> operator, and the terms on the right. The response must be a survival object as returned by the <code>Surv</code> function.
data	a list of batch data.frames in which to interpret the variables named in the <code>formula</code> . See Details.
learningRates	a function specifying how to define learning rates in steps of the algorithm. By default the $f(t)=1/t$ is used, where t is the number of algorithm's step.
beta_0	a numeric vector (if of length 1 then will be replicated) of length equal to the number of variables after using <code>formula</code> in the <code>model.matrix</code> function
epsilon	a numeric value with the stop condition of the estimation algorithm.

Details

A `data` argument should be a list of data.frames, where in every batch data.frame there is the same structure and naming convention for explanatory and survival (times, censoring) variables. See Examples.

Note

If one of the conditions is fullfiled (j denotes the step number)

- $\|\beta_{j+1} - \beta_j\| < \text{epsilon}$ parameter for any j
- $j > \text{max.iter}$

the estimation process is stopped.

Author(s)

Marcin Kosinski, <m.p.kosinski@gmail.com>

Examples

```
library(survival)
## Not run:
coxphSGD(Surv(time, status) ~ ph.ecog + age, data = split(lung, sample(1:4,
  size = 228, replace = TRUE)))

## End(Not run)
```

dataCox

Cox Proportional Hazards Model Data Generation From Weibull Distribution

Description

Function dataCox generates random survival data from Weibull distribution (with parameters lambda and rho for given input x data, model coefficients beta and censoring rate for censoring that comes from exponential distribution with parameter censRate).

Usage

```
dataCox(N, lambda, rho, x, beta, censRate)
```

Arguments

N	Number of observations to generate.
lambda	lambda parameter for Weibull distribution.
rho	rho parameter for Weibull distribution.
x	A data.frame with input data to generate the survival times for.
beta	True model coefficients.
censRate	Parameter for exponential distribution, which is responsible for censoring.

Details

For each observation true survival time is generated and a censoring time. If censoring time is less than survival time, then the survival time is returned and a status of observations is set to 0 which means the observation had censored time. If the survival time is less than censoring time, then for this observation the true survival time is returned and the status of this observation is set to 1 which means that the event has been noticed.

References

<http://onlinelibrary.wiley.com/doi/10.1002/sim.2059/abstract>
 Generating survival times to simulate Cox proportional hazards models, 2005 by Ralf Bender, Thomas Augustin, Maria Blettner.

Examples

```
## Not run:
x <- matrix(sample(0:1, size = 20000, replace = TRUE), ncol = 2)
dCox <- dataCox(10^4, lambda = 3, rho = 2, x, beta = c(1, 3), censRate = 5)

## End(Not run)
```

simulateCoxSGD

Optimize Partial Log-Likelihood Function For Cox Proportional Hazards Model Using Stochastic Gradient Descent

Description

Function `simulateCoxSGD` splits input `dCox` data on 10, 30, 60, 90, 120 and 200 groups and for each split it uses `coxphSGD` function to generate estimates for Cox Proportional Hazards Model with stochastic gradient descent optimization.

Usage

```
simulateCoxSGD(dCox = dCox, learningRates = function(x) {
  1/x
}, epsilon = 0.001, beta_0 = c(0, 0), max.iter = 100)
```

Arguments

<code>dCox</code>	Input data.frame containing survival times (columnd should be named <code>time</code>) and status (columnd should be named <code>status</code>). So far this function only supports 2 explanatory variable (that should be named <code>x1</code> and <code>x2</code>).
<code>learningRates</code>	Parameter passed to <code>coxphSGD</code> .
<code>epsilon</code>	Parameter passed to <code>coxphSGD</code> .
<code>beta_0</code>	Parameter passed to <code>coxphSGD</code> .
<code>max.iter</code>	Parameter passed to <code>coxphSGD</code> to multiple the maximal iterations by the rows number.

Examples

```
## Not run:
x <- matrix(sample(0:1, size = 20000, replace = TRUE), ncol = 2)
dCox <- dataCox(10^4, lambda = 3, rho = 2, x, beta = c(1, 3), censRate = 5)
d2ggplot <- simulateCoxSGD(dCox, learningRates = function(x) {
  1/(100 * sqrt(x))
}, max.iter = 10, epsilon = 1e-05)

## End(Not run)
```

A.3. Analiza wpływu mutacji genów na czas życia

```

extractSurvival <- function(cohorts){

survivalData <- list()
for(i in cohorts){
  get(paste0(i, ".clinical"), envir = .GlobalEnv) %>%
    select(patient.bcr_patient_barcode,
           patient.vital_status,
           patient.days_to_last_followup,
           patient.days_to_death ) %>%
    mutate(bcr_patient_barcode = toupper(patient.bcr_patient_barcode),
           patient.vital_status = ifelse(patient.vital_status %>%
                                         as.character() == "dead", 1, 0),
           barcode = patient.bcr_patient_barcode %>%
             as.character(),
           times = ifelse( !is.na(patient.days_to_last_followup),
                          patient.days_to_last_followup %>%
                            as.character() %>%
                            as.numeric(),
                          patient.days_to_death %>%
                            as.character() %>%
                            as.numeric()
                         ) %>%
filter(!is.na(times)) -> survivalData[[i]]

}
do.call(rbind,survivalData) %>%
  select(bcr_patient_barcode, patient.vital_status, times) %>%
  unique
}

extractCohortIntersection <- function(){

data(package = "RTCGA.mutations")$results[,3] %>%
  gsub(".mutations", "", x = .) -> mutations_data
data(package = "RTCGA.clinical")$results[,3] %>%
  gsub(".clinical", "", x = .) -> clinical_data

intersect(mutations_data, clinical_data)
}

prepareForumlaSGD <- function(coxData){
  as.formula(paste("Surv(times, patient.vital_status) ~ ",
                  paste(names(coxData)[-c(1092, 1093)], collapse = "+"),
                  collapse = ""))
}

```

```

extractMutations <- function(cohorts, prc){
  mutationsData <- list()
  for(i in cohorts){
    get(paste0(i, ".mutations"), envir = .GlobalEnv) %>%
      select(Hugo_Symbol, bcr_patient_barcode) %>%
      filter(nchar(bcr_patient_barcode)==15) %>%
      filter(substr(bcr_patient_barcode, 14, 15)=="01") %>%
      unique -> mutationsData[[i]]
  }
  do.call(rbind, mutationsData) %>% unique -> mutationsData
  mutationsData %>%
    group_by(Hugo_Symbol) %>%
    summarise(count = n()) %>%
    arrange(desc(count)) %>%
    mutate(count_prc = count/length(unique(mutationsData$bcr_patient_barcode))) %>%
    filter_(paste0("count_prc > ",prc)) %>%
    select(Hugo_Symbol) %>%
    unlist -> topGenes
  mutationsData %>%
    filter(Hugo_Symbol %in% topGenes) -> mutationsData_top
  mutationsData_top %>%
    dplyr::group_by(bcr_patient_barcode) %>%
    dplyr::summarise(count = n()) %>%
    group_by(count) %>%
    summarise(total = n()) %>%
    arrange(desc(count))
  as.data.table(mutationsData_top) -> mutationsData_top_DT
  dcast.data.table(mutationsData_top_DT, bcr_patient_barcode ~ Hugo_Symbol, fill = 0) %>%
    as.data.frame -> mutationsData_top_dcasted
  mutationsData_top_dcasted[,-1][mutationsData_top_dcasted[,-1] != "0"] <- 1
  mutationsData_top_dcasted -> result
  names(result) <- gsub(names(result), pattern = "-", replacement = "")
  result
}

prepareCoxDataSplit <- function(mutationsData, survivalData, groups, seed = 4561){
  mutationsData %>%
    mutate(bcr_patient_barcode = substr(bcr_patient_barcode,1,12)) %>%
    left_join(survivalData,
              by = "bcr_patient_barcode") -> coxDATA
  coxDATA <- coxDATA[, -c(1,2)]
  apply(coxDATA[,-c(1092, 1093)], MARGIN = 2, function(x){
    as.numeric(as.character(x))
  }) -> coxDATA[,-c(1092, 1093)]
  set.seed(seed)
  sample(groups, replace = TRUE, size = 6459) -> groups
  split(coxDATA, groups) coxDATA_split
}

```

A.4. Implementacje optymalizacji w regresji logistycznej

```

logitGD <- function(y, x, optim.method = "GDI", eps = 10e-4,
                      max.iter = 100, alpha = function(t){1/t}, beta_0 = c(0,0)){
  stopifnot(length(y) == length(x) & optim.method %in% c("GDI", "GDII", "SGDI") &
            is.numeric(c(max.iter, eps, x)) & all(c(eps, max.iter) > 0) &
            is.function(alpha))
  iter <- 0
  err <- list()
  err[[iter+1]] <- eps+1
  w_old <- beta_0

  res <-list()
  while(iter < max.iter && (abs(err[[ifelse(iter==0,1,iter)]]) > eps)){

    iter <- iter + 1
    if (optim.method == "GDI"){
      w_new <- w_old + alpha(iter)*updateWeightsGDI(y, x, w_old)
    }
    if (optim.method == "GDII"){
      w_new <- w_old + as.vector(inverseHessianGDII(x, w_old)%*%
                                    updateWeightsGDI(y, x, w_old))
    }
    if (optim.method == "SGDI"){
      w_new <- w_old + alpha(iter)*updateWeightsSGDI(y[iter], x[iter], w_old)
    }
    res[[iter]] <- w_new
    err[[iter]] <- sqrt(sum((w_new - w_old)^2))

    w_old <- w_new
  }
  return(list(steps = c(list(beta_0),res), errors = c(list(c(0,0)),err)))
}

updateWeightsGDI <- function(y, x, w_old){
  #(1/length(y))*c(sum(y-p(w_old, x)), sum(x*(y-p(w_old, x))))
  c(sum(y-p(w_old, x)), sum(x*(y-p(w_old, x))))
}

updateWeightsSGDI <- function(y_i, x_i, w_old){
  c(y_i-p(w_old, x_i), x_i*(y_i-p(w_old, x_i)))
}

p <- function(w_old, x_i){
  1/(1+exp(-w_old[1]-w_old[2]*x_i))
}

```

```

inverseHessianGDII <- function(x, w_old){
  solve(
    matrix(c(
      sum(p(w_old, x)*(1-p(w_old, x))),
      sum(x*p(w_old, x)*(1-p(w_old, x))),
      sum(x*p(w_old, x)*(1-p(w_old, x))),
      sum(x*x*p(w_old, x)*(1-p(w_old, x)))
    ),
    nrow =2 )
  )
}

set.seed(1283)
x <- runif(10000)
z <- 2 + 3*x
pr <- 1/(1+exp(-z))
y <- rbinom(10000,1,pr)

global_loglog <- function(beta1, beta2, xX, yY){
  sum(yY*(beta2+beta1*xX)-log(1+exp(beta2+beta1*xX)))
}

calculate_outer <- function(x, y){
  ## contours
  outer_res <- outer(seq(0,4, length = 100),
                      seq(0,5, length = 100),
                      Vectorize( function(beta1,beta2){
                        global_loglog(beta1, beta2, xX = x, yY = y)
                      } ))
}

outer_res_melted <- melt(outer_res)

outer_res_melted$Var1 <- as.factor(outer_res_melted$Var1)
levels(outer_res_melted$Var1) <- as.character(seq(0,4, length = 100))
outer_res_melted$Var2 <- as.factor(outer_res_melted$Var2)
levels(outer_res_melted$Var2) <- as.character(seq(0,5, length = 100))
outer_res_melted$Var1 <- as.numeric(as.character(outer_res_melted$Var1))
outer_res_melted$Var2 <- as.numeric(as.character(outer_res_melted$Var2))
return(outer_res_melted)
}

library(ggplot2); library(ggthemes); library(reshape2)

```

```

graphSGD <- function(beta, y, x, seed = 4561, outerBounds = calculate_outer(x,y)){
  set.seed(seed); beta <- rev(beta); beta[2] -> XX; beta[1] -> YY
  logitGD(y, x, optim.method = "GDI", beta_0 = beta,
            eps = 10e-4, max.iter = 10000,
            alpha = function(t){1/(1000*sqrt(t))})$steps -> GDI.S
  logitGD(y, x, optim.method = "GDII", beta_0 = beta,
            eps = 10e-4, max.iter = 5000)$steps -> GDII
  ind2 <- sample(length(y))
  logitGD(y[ind2], x[ind2], optim.method = "SGDI", beta_0 = beta,
            max.iter = 10000, eps = 10e-4,
            alpha = function(t){1/sqrt(t)})$steps -> SGDI.1.S
  ind3 <- sample(length(y))
  logitGD(y[ind3], x[ind3], optim.method = "SGDI", beta_0 = beta,
            max.iter = 10000, eps = 10e-4,
            alpha = function(t){5/sqrt(t)})$steps -> SGDI.5.S
  ind4 <- sample(length(y))
  logitGD(y[ind4], x[ind4], optim.method = "SGDI", beta_0 = beta,
            max.iter = 10000, eps = 10e-4,
            alpha = function(t){6/sqrt(t)})$steps -> SGDI.6.S
  do.call(rbind, c(GDI.S, GDII, SGDI.1.S, SGDI.5.S, SGDI.6.S)) -> coeffs
  unlist(lapply(list(GDI.S, GDII, SGDI.1.S, SGDI.5.S, SGDI.6.S),
                length)) -> algorithm
  data2viz <- cbind(as.data.frame(coeffs),
                     algorithm = unlist(mapply(rep,
                                               c(paste("GDI", length(GDI.S), "steps"),
                                                 paste("GDII", length(GDII), "steps"),
                                                 paste("SGDI.1", length(SGDI.1.S), "steps"),
                                                 paste("SGDI.5", length(SGDI.5.S), "steps"),
                                                 paste("SGDI.6", length(SGDI.6.S), "steps")),
                                               algorithm)))
  names(data2viz)[1:2] <- c("Intercept", "X")
  data2viz$algorithm <- factor(data2viz$algorithm, levels = rev(levels(data2viz$algorithm)))
  ggplot()+
    geom_path(aes(x = data2viz$X,
                  y = data2viz$Intercept,
                  col = data2viz$algorithm,
                  group = data2viz$algorithm), size = 1) +
    geom_point(aes(as.vector(round(coefficients(glm(y~x,
                                                family = 'binomial'))), 2)[2]),
               as.vector(round(coefficients(glm(y~x,
                                                family = 'binomial'))), 2)[1])),
    col = "black", size = 4, shape = 15) +
    geom_point(aes(x=XX, y=YY),
               col = "black", size = 4, shape = 17) +
    theme_bw(base_size = 20) +
    theme(panel.border = element_blank(),
          legend.key = element_blank()) +
    scale_colour_brewer(palette="Set1", name = 'Algorithm') +
    xlab('X') +ylab('Intercept')
}

```

Literatura

- Aldrich, J. (1997). R. A. Fisher and the making of maximum likelihood 1912 – 1922. *Statistical Science*, 12(3):162–176.
- Allaire, J., Cheng, J., Xie, Y., McPherson, J., Chang, W., Allen, J., Wickham, H., Atkins, A., and Hyndman, R. (2015). *rmarkdown: Dynamic Documents for R*. R package version 0.8.1, <http://CRAN.R-project.org/package=rmarkdown>.
- Asselain, B. and Mould, R. F. (2010). Methodology of the Cox proportional hazards model. *Journal of Oncology*, 60(5):403–409.
- Bache, S. M. and Wickham, H. (2014). *magrittr: A Forward-Pipe Operator for R*. R package version 1.5, <http://CRAN.R-project.org/package=magrittr>.
- Bender, R., Augustin, T., and Blettner, M. (2005). Generating survival times to simulate Cox proportional hazards models. *Statistics in Medicine*, 24(11):1713–1723.
- Biecek, P. (2011). *Przewodnik po pakiecie R*. Oficyna Wydawnicza GiS.
- Bottou, L. (1998). Online Learning and Stochastic Approximations.
- Bottou, L. (2010). Large-Scale Machine Learning with Stochastic Gradient Descent.
- Bottou, L. (2012). Stochastic Gradient Descent Tricks.
- Box-Steffensmeier, J. M. and Jones, B. S. (2004). *Event History Modeling: A Guide for Social Scientists*. Cambridge University Press.
- Burzykowski, T. (2015). *Notatki do przedmiotu Biostatystyka*. Politechnika Warszawska, <https://e.mini.pw.edu.pl/sites/default/files/biostatystyka.pdf>.
- Cheng, J. and Xie, Y. (2014). *rlp: An Example of Using Literate Programming for R Package Development*. R package version 0.0.2, <https://github.com/yihui/rlp>.
- Chin, L., Andersen, J., and Futreal, P. (2011a). Cancer genomics: from discovery science to personalized medicine. *Nature Medicine*, 17(2):297–303.
- Chin, L., Hahn, W., Getz, G., and Meyerson, M. (2011b). Making sense of cancer genomic data. *Genes and Development*, 25(6):534–555.
- Collett, D. (1994). *Modelling Survival Data in Medical Research*. Chapman and Hall.
- Cox, D. R. (1962). *Renewal Theory. Methuen Monograph on Applied Probability and Statistics*. Methuen.
- Cox, D. R. (1972). Regression models and life-tables (with discussion). *Journal of the Royal Statistical Society, B*(34):187–220.
- Czepiel, S. A. (2002). *Maximum Likelihood Estimation of Logistic Regression Models: Theory and Implementation*. <http://czep.net/stat/mlelr.pdf>.
- Dempster, A., Laird, N., and Rubin, D. (1977). Maximum Likelihood from Incomplete Data via the EM Algorithm. *Journal of the Royal Statistical Society, B*(39):1–38.
- Dennis, J. E. J. and Schnabel, R. B. (1983). *Numerical Methods For Unconstrained Optimization and Nonlinear Equations*. Prentice-Hall.

- Dobson, A. J. (2002). *An Introduction to Generalized Linear Models*. Chapman & Hall/CRC, ii edition.
- Domagała, W. (2007). Molekularne podstawy karcynogenezy i ścieżki sygnałowe niektórych nowotworów ośrodkowego układu nerwowego. *Polski Przegląd Neurologiczny*, 3:127–141.
- Dowle, M., Srinivasan, A., Short, T., Lianoglou, S., and contributions from Saporta, R. and Antonyan, .E (2015). *data.table: Extension of Data.frame*. R package version 1.9.6, <http://CRAN.R-project.org/package=data.table>.
- Finkel, J. R., Kleeman, A., and Manning, C. D. (2008). *Efficient, Feature-based, Conditional Random Field Parsing*. Proc. Annual Meeting of the ACL.
- Fisher, R. A. (1912). *An absolute criterion for fitting frequency curves*.
- Fisher, R. A. (1922). On the mathematical foundations of theoretical statistics. *Philos. Trans. Roy. Soc. London*, A(222):309–368.
- Fisher, R. A. (1925). *Statistical Methods for Research Workers*. Oliver and Boyd.
- Fortuna, Z., Macukow, B., and J., W. (2006). *Metody Numeryczne*. Wydawnictwa Naukowo-Techniczne.
- Gauss, C. F. (1809). *Theoria Motus Corporum Coelestium*.
- Gągolewski, M. (2014). *Programowanie w języku R*. Wydawnictwo Naukowe PWN.
- Goeman, J. J. (2010). L1 penalized estimation in the cox proportional hazards model. *Biom J.*, 52(1):70–84.
- Green, P. J. (1984). Iteratively Reweighted Least Squares for Maximum Likelihood Estimation, and Some Robust and Resistant Alternatives. *Journal of the Royal Statistical Society, B*:149–192.
- Hastie, T. J. and Pregibon, D. (1992). *Generalized linear models. Chapter 6 of Statistical Models in S*. Wadsworth & Brooks/Cole.
- Heckman, J. J. and Singer, B. (1985). *Longitudinal Analysis of Labor Market Data*. Cambridge University Press.
- Hosmer, D. W., Lemeshow, S., and May, S. (2008). *Applied Survival Analysis: Regression Modeling of Time to Event Data*. Wiley Series in Probability and Statistics, Wiley-Interscience, ii edition.
- Hutchinson, J. B. (1928). The Application of the Method of Maximum Likelihood to the Estimation of Linkage. *Genetics*, 14(6):519—537.
- Janik-Papis, K. and Błasiak, J. (2010a). Molekularne wyznaczniki raka piersi. inicjacja i promocja - część i. *Nowotwory - Journal of Oncology*, 60(3):236–247.
- Janik-Papis, K. and Błasiak, J. (2010b). Molekularne wyznaczniki raka piersi. progresja i nowi kandydaci - część ii. *Nowotwory - Journal of Oncology*, 60(4):341–350.
- Jennrich, R. I. and Sampson, P. F. (1976). Newton-raphson and related algorithms for maximum likelihood variance component estimation. *Technometrics*, 18:11–17.
- Kalbfleisch, J. D. and Prentice, R. L. (1980). *The Statistical Analysis of Failure Time Data*. New York: John Wiley and Sons.
- Kenward, M. G., Lesaffre, E., and Molenberghs, G. (1994). An Application of Maximum Likelihood and Generalized Estimating Equations to the Analysis of Ordinal Data from a Longitudinal Study with Cases Missing at Random. *Biometrics*, 50(4):945–953.
- Kim, J. and Kim, Y. (2004). *Gradient lasso for feature selection*. In Proceedings of the 21th International Conference on Machine LearningProc. Morgan Kaufmann, ACM, New York.
- Kim, J., Kim, Y., and Kim, Y. (2008). A gradient-based optimization algorithm for lasso. *Journal of Computational and Graphical Statistics*, 17:994–1009.

- Klein, J. P. and Moeschberger, M. L. (2003). *Survival Analysis: Techniques for Censored and Truncated Data*. New York: John Wiley and Sons.
- Kosiński, M. (2015a). *coxphSGD: Stochastic Gradient Descent log-likelihood estimation in Cox proportional hazards model*. R package version 0.1.0, <https://github.com/MarcinKosinski/coxphSGD>.
- Kosiński, M. (2015b). *RTCGA.clinical: Clinical datasets from The Cancer Genome Atlas Project*. R package version 20151101.0.0, <http://bioconductor.org/packages/RTCGA.clinical/>.
- Kosiński, M. (2015c). *RTCGA.mutations: Mutations datasets from The Cancer Genome Atlas Project*. R package version 20151101.0.0, <http://bioconductor.org/packages/RTCGA.mutations/>.
- Kosiński, M. and Biecek, P. (2015). *RTCGA: The Cancer Genome Atlas Data Integration*. R package version 1.2.0, <https://github.com/RTCGA>.
- Kotłowski, W. (2012). *Notatki do przedmiotu Techniki Optymalizacji*. Politechnika Poznańska, <http://www.cs.put.poznan.pl/wkotlowski/teaching/wyklad3b.pdf>.
- Kozłowska, J. and Łaczmańska, I. (2010). Niestabilność genetyczna - jej znaczenie w procesie powstawania nowotworów oraz diagnostyka laboratoryjna. *Nowotwory - Journal of Oncology*, 60(6):548–553.
- Legendre, A. M. (1804). *Nouvelles méthodes pour la détermination des orbites des comètes*.
- Longford, N. T. (1987). A fast scoring algorithm for maximum likelihood estimation in unbalanced mixed models with nested random effects. *Biometrika*, 74(4):817–827.
- Medicine, N. (2015). The future of cancer genomics. *Nature Medicine*, 21(2):99.
- Milewski, S. (2006). *Konspekt do przedmiotu Metody Numeryczne*. Politechnika Krakowska, http://l5.pk.edu.pl/images/skrypty/Metody_numeryczne_1.
- Millar, R. B. (2011). *Maximum Likelihood Estimation and Inference: With Examples in R, SAS and ADMB. Chapter 6. Some Widely Used Applications of Maximum Likelihood*. John Wiley & Sons, Ltd.
- Mittal, S. and Madigan, D. (1998). *A Statistical Study of On-line Learning. In Online Learning and Neural Networks*. Cambridge University Press.
- Mittal, S. and Madigan, D. (2014). High-dimensional, massive sample-size Cox proportional hazards regression for survival analysis. *Biostatistics*, 15(2):207–221.
- Niemiro, W. (2011). *Skrypt do przedmiotu Statystyka*. Uniwersytet Warszawski, <http://www-users.mat.umk.pl/~wniem/Statystyka/Statystyka.pdf>.
- Oakes, D. (2001). Biometrika centenary: survival analysis. *Biometrika*, 88(1):99–142.
- Panchenko, D. (2006a). *Notatki do otwartego kursu Statistics for Applications, Lecture 2: Maximum Likelihood Estimators*. MIT, <http://ocw.mit.edu/courses/mathematics/18-443-statistics-for-applications-fall-2006/>.
- Panchenko, D. (2006b). *Notatki do otwartego kursu Statistics for Applications, Lecture 3: Properties of MLE: consistency, asymptotic normality. Fisher information*. MIT, <http://ocw.mit.edu/courses/mathematics/18-443-statistics-for-applications-fall-2006/>.
- Park, M. Y. and Hastie, T. (2007). L1-regularization path algorithm for generalized linear models. *Journal of the Royal Statistical Society*, 69(4):659–677.
- Podsiadły, K. (2011). *Genetyczne podstawy nowotworzenia*. www.e-biotechnologia.pl/Artykuly/Genetyczne-podstawy-nowotworzenia.
- Robbins, H. E. and Siegmund, D. O. (1971). A convergence theorem for non negative almost supermartingales and some applications. *Academic Press, New York*. In Proc. Sympos. Optimizing Methods in Statistics, Ohio State University.

- Rydlewski, J. (2009). *Estymatory Największej Wiarogodności w Uogólnionych Modelach Regresji Nieliniowej*. PhD thesis.
- Sohn, I., Kim, J., Jung, S. H., and Park, C. (2009). Gradient lasso for Cox proportional hazards model. *Bioinformatics*, 25(14):1775–1781.
- Broad Institute of MIT and Harvard (2014). *Broad Institute TCGA Genome Data Analysis Center: Firehose stddata 2015 06 01 run*. DOI:10.7908/C1251HBG.
- Norwegian Multicentre Study Group (1981). Timolol-induced reduction in mortality and reinfarction. *The New England Journal of Medicine*, 304(14):801–807.
- R Core Team (2013). *R: A language and environment for statistical computing*. R Foundation for Statistical Computing, ISBN 3-900051-07-0, <http://www.R-project.org/>.
- Therneau, T. M. (2015). *A Package for Survival Analysis in S*. R package version 2.38, <http://CRAN.R-project.org/package=survival>.
- Therneau, T. M. and Grambsch, P. M. (2000). *Modeling Survival Data: Extending the Cox Model*. Springer.
- Tomczak, K., Czerwińska, P., and Wiznerowicz, M. (2015). The Cancer Genome Atlas (TCGA): an immeasurable source of knowledge. *Contemporary Oncology*, 19(1A):A68–A77.
- Toulis, P. and Airoldi, E. M. (2015). Implicit stochastic gradient descent. arXiv:1408.2923v5.
- Tran, D., Lan, T., and P., T. (2015). *sgd: Stochastic Gradient Descent for Scalable Estimation*. R package version 0.1, <https://github.com/aioldilab/sgd>.
- Wickha, H., Danenberg, P., and Eugster, M. (2015). *roxygen2: In-Source Documentation for R*. R package version 5.0.1, <http://CRAN.R-project.org/package=roxygen2>.
- Wickham, H. (2007). Reshaping Data with the reshape Package. *Journal of Statistical Software*, 21(12):1–20.
- Wickham, H. (2009). *ggplot2: elegant graphics for data analysis*. Springer New York.
- Wickham, H. (2013). *assertthat: Easy pre and post assertions*. R package version 0.1, <http://CRAN.R-project.org/package=assertthat>.
- Wickham, H. (2015). *tidyR: Easily Tidy Data with spread() and gather() Functions*. R package version 0.3.1, <http://CRAN.R-project.org/package=tidyr>.
- Wickham, H. and Francois, R. (2015). *dplyr: A Grammar of Data Manipulation*. R package version 0.4.3, <http://CRAN.R-project.org/package=dplyr>.
- Widrow, B. (1960). *An adaptive ADALINE neuron using chemical memistors*. Technical Report No. 1553-2, Stanford University.
- Widrow, B. and Ho, M. (1960). *Adaptive switching circuits*. In: IRE WESCON Conv. Record, Part 4.
- Widrow, B. and Stearns, S. D. (1985). *Adaptive Signal Processing*. Prentice Hall.
- Woodcock, S. (2014). *Notatki do otwartego kursu ECON 837, Lecture 11 Asymptotic Properties of Maximum Likelihood Estimators*. Uniwersytet Simona Frasera, <http://www.sfu.ca/~swoodcoc/teaching/sp2014/econ837/11.mle.pdf>.
- Xie, Y. (2014a). *Dynamic Documents with R and knitr*. Chapman and Hall/CRC, ii edition. ISBN 978-1498716963.
- Xie, Y. (2014b). *knitr: A Comprehensive Tool for Reproducible Research in R*. Chapman and Hall/CRC, ii edition. ISBN 978-1466561595.
- Xie, Y. (2015). *knitr: A General-Purpose Package for Dynamic Report Generation in R*. R package version 1.11, <http://CRAN.R-project.org/package=knitr>.

Marcin Piotr Kosiński
Nr albumu 265361

Warszawa, 10 stycznia 2016

Oświadczenie

Oświadczam, że pracę magisterską pod tytułem „Estymacja w modelu Coxa metodą stochastycznego spadku gradientu z przykładami zastosowań w analizie danych z The Cancer Genome Atlas”, której promotorem jest dr hab. inż Przemysław Biecek, prof. nadzw. wykonałem samodzielnie, co poświadczam własnoręcznym podpisem.

.....
Marcin Piotr Kosiński