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

Spis treści

Wprowadzenie

W przeciągu 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. Jak podaje [7], bardziej szczegółowa analiza wykazuje jakościowo różne kompromisy w przypadkach problemów uczenia maszynowego na małą i na dużą skalę. 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 ([7], [9], [87]), które wykazują niesamowitą wydajność dla problemów wielkiej skali danych.

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 metoda estymacji współczynników w modelu Coxa z wykorzystaniem algorytmu optymalizacji metodą stochastycznego spadku gradientu jest metodą nową i nie spotkaną do tej pory w literaturze, jest odporna na problem współliniowości zmiennych oraz efektywnie sprawdza się w sytuacji ciągłej poprawy estymacji współczynników dla napływających danych (*ang. streaming data*).

Analiza przeżycia jest starą, jednak wciąż aktywną dziedziną badań znajdującą nowe zastosowania w wielu dziedzinach, chociażby takich jak: biostatystyka, socjologia, ekonomia, demografia czy w naukach inżynieryjnych [36], [18], [10], [37]. 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 ([42], [62]) i do niedawna ograniczona była jedynie do zastosowań oraz analiz opartych na garści zmiennych objaśniających przy maksymalnie kilku tysiącach obserwacji. Jednak ostatnie dokonania i postępy w obszarach rozwoju technik pozyskiwania danych i ułatwiony, wraz z postępem cywilizacyjnym, dostęp do większej mocy obliczeniowej spowodowały wzmożenie zainteresowania danych 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 [17]. Jak podaje [3], model proporcjonalnych hazardów Coxa 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 ([46], [18], [37]), 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-wiargodności dodatkowego parametru związanego z karą dla modelu za zbyt dużą liczbę zmiennych w trakcie estymacji [63], [70], [32]. Taka metoda, zwana regularyzacją, prowadzi do zminimalizowana liczby 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 [58] możliwe obejścia i przybliżenia często prowadzą do dużych różnic współczynników i słabego oszacowania predykcyjnej dokładności czy złego dopasowania modelu. Rozwój badań nad danym problemem ([44]) doprowadził do powstania metod wykorzystujących jedynie pierwszą pochodną częściowej penalizowanej (z parametrem regularyzacji) funkcji log-wiarogodności w modelu Coxa ([70] w opraciu o [45], rozszerzenie podejścia w [58]), które byłyby odporne na problemy danych dużej skali, gdyż nie wymagały obliczania i odwracania macierzy Hesjanu.

Podejścia te trudno wykorzystać w sytuacjach napływu danych, gdy nie dysponuje się wszystkimi obserwacji jednocześnie, a zachodzi potrzeba wykorzystania tych już zaobserwowanych do estymacji współczynników oraz rodzi się okazja do poprawy współczynników z każdą nową obserwacją [8]. Dlatego w warunkach napływu danych możliwym rozwiązaniem jest zastosowanie algorytmu optymalizacji metodą stochastycznego spadku gradientu, który w wielu modelach do estymacji współczynników wymaga jedynie jednej obserwacji. W modelu proporcjonalnych hazardów Coxa wykorzystanie metody stochastycznego spadku gradientu jest możliwe 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 proporcjonalnych hazardów Coxa prowadzą pracownicy *Harvard Laboratory for Applied Statistical Methodology & Data Science*, którzy zaproponowali podejście uwikłanego (*ang.*

implicit) stochastycznego spadku gradientu, który dla danego kroku optymalizacji wykorzystuje w równaniu na nowe współczynniki jeszcze nie wyliczone nowe współczynniki ([78] w publikacji).

W owej pracy zaprezentowano prostsze podejście wykorzystania w estymacji współczynników w modelu proporcjonalnych hazardów Coxa estymacji opartej o stochastyczny spadek gradientu do zaobserwowanych podzbiorów obserwacji. Takie podejście pozwala na efektywne osiągnięcie zbieżności algorytmu optymalizacji, w sytuacji napływających danych o dużym rozmiarze zmiennych objaśniających, przy jednoczesnej odporności na współliniowość zmiennych oraz z profitem poprawy oszacowań współczynników modelu po każdym zaobserwowanym podzbiorze obserwacji.

Rozdział 1 pracy opisuje metodę największej wiarogodności i jej matematyczne własności, dzięki której możliwe jest wyznaczanie estymatorów największej wiarogodności. Rozdział uzasadnia również poprawność wykorzystywania estymatorów uzyskanych dzięki metodzie największej wiarogodności 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 oraz sformułowany jest model Coxa. Rozdział opisuje niezbędne założenia w modelu Coxa oraz prezentuje metodę analitycznej estymacji w oparciu o szeroko opisaną częściową funkcję log-wiarogodności modelu i estymatory największej wiarogodności. Rozdział 2 kończy się opisaniem metody generowania danych dla 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. Rozdział ukazuje wady i zalety stochastycznego spadku gradientu oraz przedstawia różnice między tym algorytmem a algorytmami spadku gradientu rzędu I (Cauchy'ego) oraz spadku gradientu rzędu II (Raphsona-Newtona). Rozdział 3 zwieńczony jest implementacją trzech omówionych algorytmów numerycznej optymalizacji dla problemu estymacji przy pomocy metody największej wiarogodności 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 algorytmem stochastycznego spadku gradientu.

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 warunki konieczne, jakie musi spełniać zaobserwowany podzbiór danych, aby nie przerwać procesu optymalizacji częściowej funkcji log-wiarogodności. Rozdział prezentuje implementację opisanej metody w pseudo kodzie oraz podaje implementację wyrażoną w języku $\mathcal{R}$ [67], [6]. Rozdział 4 kończy się przeprowadzeniem symulacji mających na celu wygenerowanie danych z modelu Coxa, dzięki wykorzystaniu implementacji z rozdziału 2, oraz na zastosowaniu przygotowanego algorytmu do wyestymowania współczynników w modelu Coxa dzięki użyciu metody stochastycznego spadku gradientu. Wyniki symulacji przedstawiono na wykresach i porównano kroki algorytmów dla różnych wielkości zaobserwowanych podzbiorów obserwacji wykorzystanych do jednego kroku algorytmu optymalizacji.

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 ([48], [49], [50]). W analizie w rozdziale 5 sprawdzany jest wpływ wystąpienia mutacji w danym genie na czas do wystąpienia zdarzenia niepożądanego jakim jest śmierć pacjenta w wyniku choroby nowotworowej.

W pracy zostały przedstawione kody pakietu $\mathcal{R}$, użyte do symulacji oraz analizy przeżycia, których zestawienie i podsumowanie 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` ([47]) 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, w niniejszym rozdziale wprowadzona została klasyczna terminologia oparta o [60].

Definicja 0.1. ***Model statystyczny*** *określamy przez podanie rodziny* $\{\mathbb{P}_\theta : \theta \in \Theta\}$ *rozkładów prawdopodobieństwa na przestrzeni próbkowej* Ω *oraz zmiennej losowej* $X : \Omega \to \mathcal{X}$*, 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. ***Przestrzenią statystyczną*** *nazywa się trójkę* $(\mathcal{X}, \mathcal{F}, \{\mathbb{P}_\theta : \theta \in \Theta\})$.

Wprowadzenie σ-ciała $\mathcal{F}$ sprawia, że przestrzeń statystyczna staje się przestrzenią mierzalną, 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 próba 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 nie zna się parametru θ. Można go jednak szacować dzięki estymatorom opartym na statystykach.

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

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 w kręgu zainteresowań 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 2 ważne definicje związane z jakością estymatorów [60], gdy rozmiar próbki $X_1, \ldots, X_n$ jest duży. Mówi 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 opisane w przybliżeniu zachowania 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 $\hat{g}(X_1, \ldots, X_n)$ wielkości $g(\theta)$ jest* ***nieobciążony****, jeśli dla każdego n*

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

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

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

dla każdego $\varepsilon > 0$.

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

$$\mathbb{P}_\theta\Big(\lim_{n\to\infty} \hat{g}(X_1, \ldots, X_n) = g(\theta)\Big) = 1.$$

Zgodność (mocna zgodność) znaczy tyle, że

$$\hat{g}(X_1, \ldots, X_n) \to g(\theta), \qquad (n \to \infty)$$

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

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

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

$\xrightarrow{d}$ *oznacza zbieżność wg rozkładu.*

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

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

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

Inaczej mówiąc, estymator jest asymptotycznie normalny, gdy:

$$\lim_{n\to\infty} \mathbb{P}_\theta\Big(\frac{\sqrt{n}}{\sigma(\theta)}(\hat{g}(X_1, \ldots, X_n) - g(\theta) \leq a\Big) = \Phi(a),$$

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

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

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

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

W dalszej części tego rozdziału zostanie wprowadzone pojęcie estymatora największej wiarogodności oraz zostaną udowodnione dla niego jego właściwości, co utwierdzi w przekonaniu, że metoda największej wiarogodności, przy odpowiednich założeniach, jest metodą konstrukcji rozsądnych estymatorów.

1.2. Metoda największej wiarogodności

Metodę największej wiarogodności wprowadził R. A. Fisher w 1922 r. [27], dla której po raz pierwszy heurystyczną procedurę numeryczną zaproponował już w 1912 r. [26]. O burzliwym procesie powstawania metody, o 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 [1].

Metoda ta, jako alternatywa dla metody najmniejszych kwadratów [53], [30], była rozwijana i szeroko stosowana później przez wielu statystyków, i wciąż znajduje obszerne zastosowania w wielu obszarach estymacji statystycznej, np. [38], [43], [56].

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

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

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

którą rozważamy jako funkcję parametru θ *przy ustalonych wartościach obserwacji* $x_1, \ldots, x_n$*, gdzie*

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

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(*θ*), 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 [69], [88], [64].

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.
Dowody w tym rozdziale są znane w literaturze i opierają się o [65] i [88].

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* $\theta \in \Theta$

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

$\mathbb{E}_{\theta_0}$ *oznacza wartość oczekiwaną względem rozkładu* $\mathbb{P}_{\theta_0}$

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

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

Obie całki równają 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.

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

$$ENW(\theta) \to \theta \quad dla \quad n \to \infty.$$

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 $X_1, \ldots, X_n$), 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

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

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

Asymptotyczna normalność estymatora największej wiarygodności

Fisher w swojej karierze wprowadził wiele pożytecznych 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, zależnej od jednowymiarowego parametru $\theta \in \Theta \subset \mathbb{R}$.* ***Informacją Fishera*** *zawartą w obserwacji X nazywa się funkcję*

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

gdzie odpowiednio

$$\mathcal{I}(\theta) = \int \Big(\frac{\partial}{\partial\theta}\log f_\theta(x)\Big)^2 f_\theta(x)dx \qquad \textit{dla zmiennej ciągłej;}$$
$$\mathcal{I}(\theta) = \sum_x \Big(\frac{\partial}{\partial\theta}\log f_\theta(x)\Big)^2 \mathbb{P}_\theta(X = x) \;\; \textit{dla zmiennej dyskretnej.}$$

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 podcałkowej 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 $\int \ldots dx$.

Wprowadzając takie założenia, otrzymano przydatne właściwości Informacji Fishera.

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

$$(i)\;\; \mathbb{E}_\theta\frac{\partial}{\partial\theta}\log f_\theta(X) = 0,$$
$$(ii)\;\; \mathcal{I}(\theta) = \mathbb{V}ar_\theta\Big(\frac{\partial}{\partial\theta}\log f_\theta(X)\Big),$$
$$(iii)\;\; \mathcal{I}(\theta) = -\mathbb{E}_\theta\Big(\frac{\partial^2}{\partial\theta^2}\log f_\theta(X)\Big).$$

Dowód tego stwierdzenia można znaleźć w [60].

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 oczekiwaną, innymi słowy 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 jest znacząco inna i 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 bliski do θ_0, a co za tym idzie, dużo ciężej będzie odróżnić tę gęstość, czyli estymacja będzie słabsza.

Dzięki pojęciu Informacji Fishera i warunkom regularności, możliwe jest udowodnienie poniższego Twierdzenia.

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

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

Z Twierdzenia widać, że im większa Informacja Fishera tym mniejsza asymptotyczna wariancja estymatora prawdziwego parametru θ_0.

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

$$\sqrt{n}(ENW(\theta) - \theta_0) = -\frac{\sqrt{n}\ell_n'(\theta_0)}{\ell_n''(\theta_1)}. \tag{1.2}$$

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

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

a to można wstawić do licznika w równaniu (1.2)

$$\begin{aligned}\sqrt{n}\ell_n'(\theta_0) &= \sqrt{n}\Big(\frac{1}{n}\sum_{i=1}^{n}\ell'(\theta_0) - 0\Big) \\ &= \sqrt{n}\Big(\frac{1}{n}\sum_{i=1}^{n}\ell'(\theta_0) - \mathbb{E}_{\theta_0}\ell'(\theta_0)\Big) \to \mathcal{N}\Big(0, \mathbb{V}ar_{\theta_0}(\ell'(\theta_0))\Big),\end{aligned} \tag{1.4}$$

gdzie zbieżność wynika z Centralnego Twierdzenia Granicznego.

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

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

z Prawa Wielkich Liczb.

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

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

z punktu (iii) ze Stwierdzenia (1.3).

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

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

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 (1.3).

■

Rozdział 2

Model Coxa

W tym rozdziale zostanie przedstawiony model proporcjonalnych hazardów Coxa. Głównym celem tej pracy jest wykorzystanie numerycznej metody estymacji współczynników metodą stochastycznego spadku gradientu w omawianym modelu. Więcej o estymacji metodą stochastycznego spadku gradientu napisane jest w rozdziale 3.2. Definicje i twierdzenia w tym rozdziale oparte są o [17], [75], [3] i [12].

2.1. Wprowadzenie do modelu Coxa i nomenklatura

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^* to czas do wystąpienia zdarzenia. Zakłada się, że wewnątrz każdej grupy $j = 1, 2, \ldots, m$ wyznaczonej przez poziomy zmiennych objaśniających, czasy $T^*_{j,i}$ dla $i = 1, \ldots, n_j$ to niezależne zmienne losowe z tego samego rozkładu o gęstości $f_j(t)$.

Definicja 2.1. ***Funkcja przeżycia*** *w grupie j, to funkcja, która spełnia*

$$S_j(t) = \mathbb{P}(T^*_{j,i} \geq t) = 1 - F_j(t), t \in \mathbb{R} \tag{2.1}$$

gdzie $F_j(t)$ to dystrybuanta rozkładu zadanego gęstością $f_j(t)$.

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

$$\begin{aligned} \lambda_j(t) &= \lim_{h \to 0} \frac{\mathbb{P}(t \leq T^* \leq t + h | T^* \geq t)}{h} \\ &= \lim_{h \to 0} \frac{\mathbb{P}(t \leq T^* \leq t + h)}{h} \cdot \frac{1}{\mathbb{P}(T^* \geq t)} \\ &= \lim_{h \to 0} \frac{F_j(t+h) - F_j(t)}{h} \cdot \frac{1}{S_j(t)} \qquad = \frac{f_j(t)}{S_j(t)}. \end{aligned} \tag{2.2}$$

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 [16], w której analizuje się awaryjność elementów przemysłowych.

Model proporcjonalnych hazardów Coxa [17] 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 tego typu model dla tabeli przeżyć i zilustrował zastosowanie modelu dla przypadku białaczki, ale model może być stosowany do obliczania przeżyć w odniesieniu do innych chorób, jak w przypadku przeżyć w chorobach nowotworowych lub kardiologicznych po transplantacji serca lub zawałach serca [61].

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

$$\lambda_i(t) = \lambda_0(t)e^{X_i(t)'\beta}, \tag{2.3}$$

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 wersji kiedy zmienne są stałe w czasie, nazywany jest **modelem proporcjonalnych hazardów**, gdyż stosunek (proporcja) hazardów dla dwóch obserwacji X_i oraz X_j jest stały 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 obserwacji w zbiorze. Niekiedy jest to uwarunkowane zbyt długim czasem do wystąpienia zdarzenia. Czasem jest to związane z zaplanowanym 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 dla nich zaobserwować czasów 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 tej obserwacji w analizie i 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. ***Cenzuorwanie 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*

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

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*

$$g(t; \theta, \phi) \equiv g(t; \phi), \tag{2.4}$$

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 jednakowym rozkładzie i funkcji gęstości $f(t; \theta)$, czyli θ parametryzuje jedynie rozkład czasów zdarzeń.

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

Model proporcjonalnych hazardów Coxa oparty jest na założeniach:

i) Współczynniki modelu $\beta_k, k = 1, \cdots, p$ są stałe w czasie, co przekłada się na to, że stosunek hazardów dla dwóch obserwacji jest stały w czasie.

ii) Postać funkcjonalna 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, 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, którą wprowadził Cox w 1972 r. [17].

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

$$\frac{e^{X_i'\beta}}{\sum\limits_{l\in\mathscr{R}(t_i)} e^{X_l'\beta}}, \tag{2.5}$$

gdzie *zbiór ryzyka* $\mathscr{R}(t_i)$, 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:

$$L(\beta,\varphi) = L_p(\beta)\cdot L^*(\beta,\varphi), \tag{2.6}$$

gdzie, dla $\lambda(t)$ wprowadzonego w definicji (2.2)

$$L_p(\beta) = \prod_{i=1}^{n} f(t_i;\beta)^{\delta_i} S(t_i;\beta)^{1-\delta_i} = \prod_{i=1}^{n} \lambda(t_i;\beta)^{\delta_i} S(t_i;\beta) \tag{2.7}$$

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:

$$L_p(\beta) = \prod_{i=1}^{K} \frac{e^{X_i'\beta}}{\sum\limits_{l=1}^{n} Y_l(t_i) e^{X_l'\beta}}, \tag{2.8}$$

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 zaobserwowanych 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 $L_p(\beta)$ jest **pełną** funkcją wiarogodności, bowiem wówczas

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

co bierze się z definicji cenzurowania nie-informatywnego (2.4)

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

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

W sytuacjach, gdy nie jest spełnione założenie nie-informatywności cenzurowania i częściowa funkcja wiarogodności nie jest funkcją wiarogodności w sensie bycia proporcjonalną do prawdopodobieństwa obserwowanego zbioru, można ją traktować jako funkcję wiarogodności dla celów asymptotycznego wnioskowania o współczynnikach modelu, zobacz [75].

Analityczna estymacja współczynników

Standardowo w celu znalezienia maximum, aby ułatwić obliczenia, można rozważaną funkcję obłożyć monotoniczną transformacją jaką jest logarytm, tak aby w konsekwencji otrzymać **częściową funkcję log-wiarogodności**

$$\ell_p(\beta) = \sum_{i=1}^{K} X_i'\beta - \sum_{i=1}^{K} \log\Big(\sum_{l\in\mathscr{R}(t_i)} e^{X_l'\beta}\Big). \tag{2.9}$$

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

$$U_k(\beta) = \frac{\partial \ell_k(\beta)}{\partial \beta_k} = \sum_{i=1}^{K} \Big(X_{ik} - A_{ik}\Big), \tag{2.10}$$

X_{ik} to i ta obserwacja i k ta zmienna.

gdzie czynnik

$$A_{ik} = \frac{\sum\limits_{l\in\mathscr{R}(t_i)} X_{lk} e^{X_l'\beta}}{\sum\limits_{l\in\mathscr{R}(t_i)} e^{X_l'\beta}} \tag{2.11}$$

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

Z kolei drugie pochodne cząstkowe, jak podaje [17], mają postać dla $k_1, k_2 = 1, \ldots, p$

$$\mathscr{I}_{k_1k_2}(\beta) = -\frac{\partial^2 L_p(\beta)}{\partial\beta_{k_1}\partial\beta_{k_2}} = \sum_{i=1}^{K} C_{ik_1k_2}(\beta), \tag{2.12}$$

gdzie

$$C_{ik_1k_2}(\beta) = \frac{\sum\limits_{l\in\mathscr{R}(t_i)} X_{lk_1}X_{lk_2} e^{X_l'\beta}}{\sum\limits_{l\in\mathscr{R}(t_i)} e^{X_l'\beta}} - A_{ik_1}(\beta)A_{ik_2}(\beta) \tag{2.13}$$

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

Estymator największej wiarogodności β można uzyskać poprzez przyrównanie (2.10) do 0, a numerycznie poprzez iteracyjne wykorzystanie (2.10) oraz (2.12) 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 wzdłuż gradientu, która jest szerzej opisana w następnym rozdziale w podrozdziale 3.2.

2.4. Generowanie danych dla modelu Coxa

W celu skutecznej diagnostyki procesu estymacji w modelu Coxa, należy wiedzieć jak dane dla tego modelu można generować, aby ów model symulować dla znanych współczynników.

Poniższy rozdział przedstawia metodę odwrotnych prawdopodobieństw opisaną szerzej w [5], 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, który jest opisany w rozdziale 3.

Mówiąc o funkcji przeżycia warto wprowadzić skumulowaną funkcję hazardu.

Definicja 2.7. ***Skumulowaną funkcją hazardu*** *nazywa się funkcję spełniającą zależność*

$$H(t) = \int_0^t \lambda(u)du, \tag{2.14}$$

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

Wtedy dla zdefiniowanej w (2.3) funkcji hazardu funkcja przeżycia i jej dopełnienie (dystrybuanta) dla modelu Coxa proporcjonalnych hazardów wygląda następująco

$$S(t|x) = e^{-H_0(t)\cdot e^{x'\beta}} \tag{2.15}$$

$$F(t|x) = 1 - e^{-H_0(t)\cdot e^{x'\beta}} \tag{2.16}$$

gdzie $H_0(t)$ to bazowa skumulowana funkcja hazardu.

Niech Y będzie zmienną losową o dystrybuancie zadanej w (2.16), wtedy jak wykazano w [57] zmienna losowa $U = F(Y)$ pochodzi z rozkładu jednostajnego $U \sim \mathcal{U}([0,1])$. Zachodzi to również dla zmiennej losowej $1-U \sim \mathcal{U}([0,1])$. Dodatkowo, niech T będzie czasem przeżycia w modelu Coxa (definicja 2.3), wtedy z (2.16) wynika

$$U = e^{-H_0(T)\cdot e^{x'\beta}} \sim \mathcal{U}([0,1]). \tag{2.17}$$

Jeżeli $\lambda_0(t) > 0$ dla każdego t, to $H_0(t)$ jest dodatnia i można mówić o jej odwrotności, zaś czas przeżycia T dla modelu Coxa może być wyrażony przez

$$T = H_0^{-1}(-\log(U)\cdot e^{-x'\beta}), \tag{2.18}$$

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

Przykład symulacji dla rozkładu Weibulla

Równanie (2.18) 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 $\mathcal{R}$ [67], dzięki któremu możliwe jest generowanie czasów zdarzeń pochodzących z rozkładu Weibulla [18], dla którego funkcja hazardu ma postać

$$\lambda_0(t) = \lambda\rho t^{\rho-1}, \tag{2.19}$$

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

Z (2.14) wynika, że bazowa skumulowana funkcja hazardu dla rozkładu Weibulla wynosi

$$H_0(t) = \int_0^t \lambda \rho u^{\rho-1} du = \lambda t^{\rho}, \tag{2.20}$$

zaś jej funkcja przeciwna to

$$H_0^{-1}(t) = (\lambda^{-1} t)^{\frac{1}{\rho}}. \tag{2.21}$$

Wtedy podstawiając (2.21) do (2.18) można otrzymać

$$T = (\lambda^{-1} \cdot (-\log(U)) \cdot e^{-x'\beta})^{\frac{1}{\rho}} = \Big(-\frac{\log(U)}{\lambda e^{x'\beta}} \Big)^{\frac{1}{\rho}}. \tag{2.22}$$

Wynik ten oznacza, że T czyli odpowiadające czasy zdarzeń pochodzą z warunkowego rozkładu Weibulla (warunkowanego przez x) o parametrach kształtu ρ i skali $\lambda e^{x'\beta}$.

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

Przez numerykę rozumie się dziedzinę matematyki zajmującą się rozwiązywaniem przybliżonych zagadnień algebraicznych. 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 ścisłych danych problemów. Z czasem jednak, przy powstawaniu coraz to bardziej skomplikowanych teorii opisujących zjawiska, problemy te stawały się na tyle złożone, iż ich rozwiązywanie ścisłe było albo bardzo czasochłonne albo też zgoła niemożliwe. Numeryka pozwalała znajdywać przybliżone rozwiązania z żądaną dokładnością. Ich podstawową zaletą była ogólność tak 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. Stąd prawdziwy renesans metod numerycznych nastąpił wraz z powszechnym użyciem w pracy naukowej maszyn cyfrowych, a w szczególności mikrokomputerów [55].

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

W poniższym rozdziale przedstawione są klasyczne algorytmy spadku wzdłuż gradientu Cauchy'ego oraz Raphsona-Newtona. Następnie omówiony jest algorytm stochastycznego spadku wzdłuż gradientu, którego wykorzystanie do estymacji współczynników w modelu Coxa jest kluczowym celem tej pracy. Algorytm stochastycznego spadku gradientu to metoda optymalizacji wzdłuż spadku gradientu wykorzystywana w sytuacjach, gdy rozważaną funkcję można zapisać jako sumę różniczkowalnych składników. Ponieważ popularne metody statystycznej estymacji takie jak algorytm Fishera (*ang. Fisher scoring*, [28]), algorytm EM (*ang. Expectation–maximization algorithm*, [20]) czy iteracyjna ważona metoda najmniejszych kwadratów ([33]) nie zawsze przenoszą się na zastosowania do danych dużej skali bądź danych napływowych (*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.

Definicje i pojęcia w tym rozdziale pochodzą z [7], [9], [51] i [29].

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 (nieliniowych) 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 $\| w_{k+1} - w_k \| < \epsilon$ lub wartość gradientu funkcji w wyznaczonym punkcie jest bliska wektorowymi zerowemu $\| \nabla_Q(\mathbf{w_k}) \| \leqslant \epsilon$ (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 $k = 1, 2, \ldots$ 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 $k = 1, 2, \ldots$ aż do zbieżności
  - Wyznaczamy gradient w punkcie $w_{k-1}, \nabla_Q(w_{k-1})$ i odwrotność Hesjanu $(D^2_Q(w_{k-1}))^{-1}$.
  - Robimy krok wzdłuż negatywnego gradientu z zadanym krokiem przez Hesjan:

$$w_k = w_{k-1} - (D^2_Q(w_{k-1}))^{-1} \nabla_Q(w_{k-1}). \tag{3.1}$$

Metoda stochastycznego spadku wzdłuż gradientu I

Minimalizacja funkcji $Q(w)$:

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

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

3.2. Algorytm stochastycznego spadku wzdłuż gradientu I

Stochastyczny spadek gradientu to popularny algorytm wykorzystywany do estymacji współczynników w szerokiej gamie modeli uczenia maszynowego takich jak maszyny wektorów podpierających (*ang. Support Vector Machines*), regresja logistyczna czy modele graficzne [25]. W połączeniu z algorytmem propagacji wstecznej jest standardowym algorytmem w trenowaniu sztucznych sieci neuronowych. Algorytm stochastycznego spadku gradientu był używany już od 1960 przy estymacji współczynników w modelu regresji liniowej, oryginalnie znanym jako *ADALINE* [85]. Kolejnym algorytmem wykorzystującym stochastyczny spadek gradientu jest filtr adaptacyjny najmniejszych średnich kwadratów [87] (*ang. least mean squares (LMS) adaptive filter*), który został wynaleziony przez Bernarda Widrowa, twórcę *ADALINE*.

Idea algorytmu stochastycznego spadku gradientu jest następująca: zamiast obliczać gradient na całej funkcji L, w danym kroku oblicz gradient tylko na pojedynczym elemencie ℓ_i. Nazwa *stochastyczny* bierze się stąd, iż oryginalnie wybiera się element ℓ_i 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 algorytmu α_k był malejący i spełniał poniższe warunki $\sum_k \alpha_k = \infty$ oraz $\sum_k \alpha_k^2 < \infty$ [7]. Twierdzenie Robbinsa-Siegmunda [68] przy łagodnych warunkach zapewnia zbieżność prawie na pewno [8], nawet gdy optymalizowana funkcja nie jest wszędzie różniczkowalna.

Prędkość zbieżności stochastycznego spadku gradientu jest w rzeczywistości ograniczana przez zgrubną (*ang. noisy*) aproksymacje prawdziwego gradientu. Gdy długości kroków algorytmu maleją zbyt wolno, wariancja estymatorów parametrów w_k maleje równie wolno. Gdy kroki algorytmu maleją zbyt szybko, oczekiwane estymatory parametrów w_k potrzebują więcej czasu by osiągnąć optimum [7]. Pod pewnymi warunkami regularności [59], najlepsza prędkość zbieżności jest uzyskana dla kroków algorytmu $\alpha_k \sim k^{-1}$.

Jak wykazano w [21] 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: $\| \mathbf{w}^* - \mathbf{w_{k+1}} \| \leqslant c \| \mathbf{w}^* - \mathbf{w_k} \|$. 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 zbiega wolno i 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 dwuwymiarowym wykresie, postanowiono ograniczyć się do modelu z jedną zmienną objaśniającą 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 modelu regresji logistycznej, za [19] i [22], ma postać

$\beta = (\beta_1, \beta_2)$

$$\ell(\beta) = \sum_{i=1}^{N} \Big(y_i(\beta_1 + \beta_2 x_i) - \log(1 + \exp(\beta_1 + \beta_2 x_i)) \Big) = \sum_{i=1}^{N} Q_i(\beta_1, \beta_2), \tag{3.3}$$

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

Dla tak skonstruowanej funkcji wiarogodności, współrzędne gradientu to odpowiednio

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

zaś macierz informacji wyraża się jak następuje

$$\mathscr{I}(\beta) = \begin{bmatrix} \sum\limits_{i=1}^{N} \pi_i(\beta)(1 - \pi_i(\beta)) & \sum\limits_{i=1}^{N} x_i \pi_i(\beta)(1 - \pi_i(\beta)) \\ \sum\limits_{i=1}^{N} x_i \pi_i(\beta)(1 - \pi_i(\beta)) & \sum\limits_{i=1}^{N} x_i^2 \pi_i(\beta)(1 - \pi_i(\beta)) \end{bmatrix}, \tag{3.5}$$

gdzie $\pi_i(\beta) = \frac{\exp(\beta_1 + \beta_2 x_i)}{1 + \exp(\beta_1 + \beta_2 x_i)}$, a N to liczba obserwacji.

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

Metoda spadku wzdłuż gradientu I (Cauchy'ego)

$\alpha_k \in \mathbb{R}$

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

Metoda spadku wzdłuż gradientu II (Newtona-Raphsona)

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

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

Metoda stochastycznego spadku wzdłuż gradientu I

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

dla wylosowanego w danym kroku i.

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

Symulacje trajektorii zbieżności algorytmów

Poniższymi wywołaniami kodów z pakietu $\mathcal{R}$ [67] 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 normalnego.

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

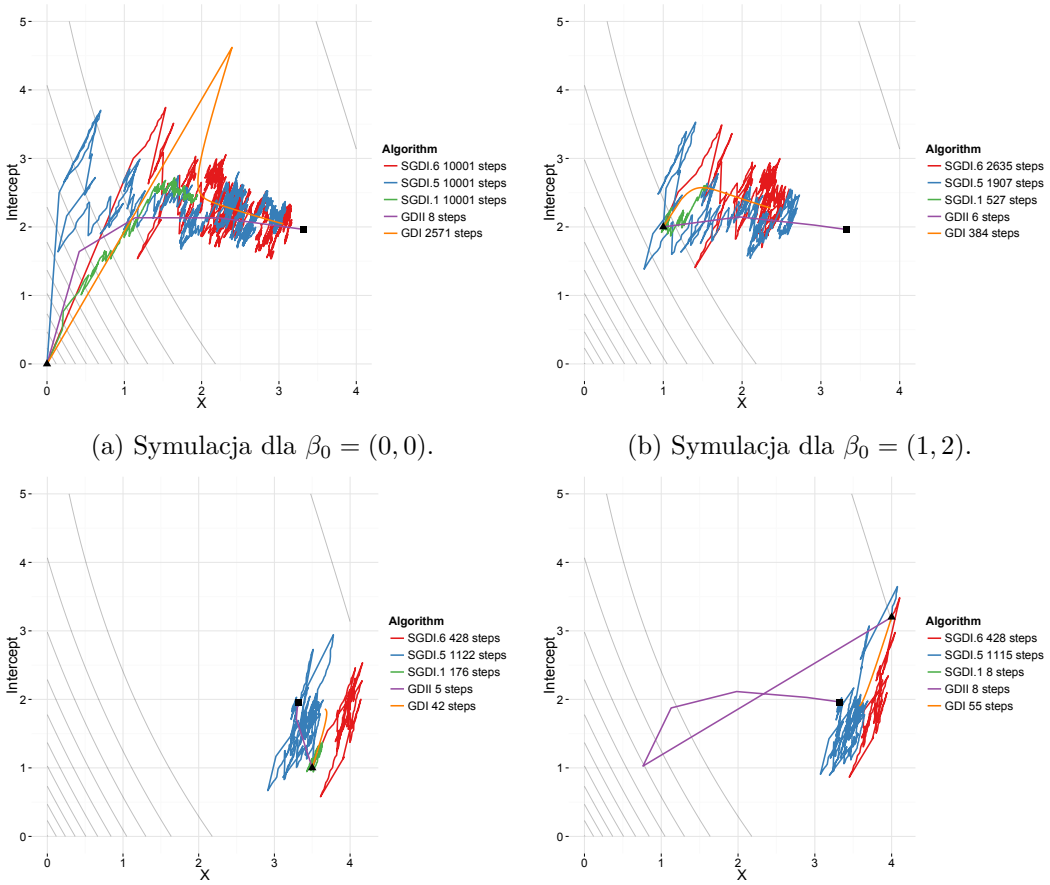
Dla tak sztucznie zasymulowanych danych przygotowano funkcję `logitGD()`, dzięki której można śledzić wartości ekstremum w danym kroku kolejnych omawianych algorytmów. Kody funkcji `logitGD()` oraz funkcji `graphSGD` tworzącej wykresy porównujące trajektorie zbieżności dla różnych algorytmów spadku gradientu dostępne są w Dodatku A.2. Wyniki wywołań funkcji `graphSGD` zostały umieszczone na Rysunkach (3.2) - (3.5).

Na każdym z wykresów na osi OX ukazano współczynnik dla zmiennej objaśniającej w prostym modelu regresji logistycznej z jedną zmienną objaśniającą w kolejnych krokach poszczególnych algorytmów. Na osi OY zaznaczono współczynnik wyrazu wolnego w tym modelu. Punkt startowy zaznaczono na wykresie czarnym trójkątem. Czarnym kwadratem zaznaczono ekstremum funkcji log-wiarogodności w tym modelu, wyliczone dzięki funkcji `glm` [35], która do estymacji używa algorytmu iteracyjnej ważonej metody najmniejszych kwadratów, która dla regresji logistycznej jest równoważna algorytmowi Fishera (*ang. Fisher's Scoring algorithm* [54], [41]) używającego obserwowanej macierzy Informacji Fishera (definicja (1.10)) w miejscu Hesjanu w algorytmie Newtona-Raphsona (równanie (3.1)). Trajektoria zbieżności do minimum dla odrębnych algorytmów zaznaczono oddzielnymi kolorami, które są wyjaśnione na legendzie wykresu, na której dodatkowo wpisano liczbę kroków wymaganą przez algorytm do zbieżności. Przez `GDI` oznaczono trajektoria dla algorytmu spadku gradientu rzędu I, przez `GDII` oznaczono trajektoria dla algorytmu spadku gradientu rzędu II, zaś przez `SGD.i` oznaczono 3 różne trajektorie dla algorytmów stochastycznego spadku gradientu dla różnych ciągów odpowiadających długości kroku algorytmu. Indeks i odpowiada ciągowi wybranemu do wyznaczania długości kroku algorytmu na zasadzie $\alpha_{ki} = \frac{i}{\sqrt{k}}$.

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 $\epsilon = 10^{-4}$ ($\epsilon = 10^{-5}$ dla punktu startowego $\beta_0 = (0, 0)$) oraz ustalono ciąg odpowiadający długościom kroków w algorytmie spadku gradientu rzędu I `GDI` na $\alpha_k = \frac{1}{1000\sqrt{k}}$. Symulacje powtórzono czterokrotnie dla czterech różnych punktów startu algorytmów $\beta_0 = (0, 0), (1, 2), (3.5, 1), (4, 3.2)$.

Ponieważ każda trajektoria w procesie estymacji metodą stochastycznego gradientu dla 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 dla 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 dla modelu regresji logistycznej (równanie (3.3)).



(a) Symulacja dla $\beta_0 = (0, 0)$.

(b) Symulacja dla $\beta_0 = (1, 2)$.

(c) Symulacja dla $\beta_0 = (3.5, 1)$.

(d) Symulacja dla $\beta_0 = (4, 3.2)$.

Rysunek 3.1: Wykresy o wspólnym zakresie osi z zaznaczeniem warstwic funkcji log-wiarogodności dla modelu regresji logistycznej (równanie (3.3)). Wykresy przedstawiają porównanie algorytmów spadku gradientu. Początkowe dane losowano z innego ziarna losowania ustalone na 4561. Wykresy przedstawiają ścieżki zbieżności w kolejnych krokach algorytmów spadku gradientu. Ustalono maksymalną liczbę iteracji na 10001, zaś warunek stopu ustalono na $\epsilon = 10^{-4}$ ($\epsilon = 10^{-5}$ dla punktu startowego $\beta_0 = (0, 0)$). Trójkątem zaznaczono punkt startowy, a kwadratem wyestymowane rozwiązanie przy pomocy funkcji `glm` [35]. Przez `GDI` oznaczono trajektoria dla algorytmu spadku gradientu rzędu I, przez `GDII` oznaczono trajektoria dla algorytmu spadku gradientu rzędu II, zaś przez `SGD.i` oznaczono 3 różne trajektorie dla algorytmów stochastycznego spadku gradientu dla różnych ciągów odpowiadających długości kroku algorytmu. Indeks i odpowiada ciągowi wybranemu do wyznaczania długości kroku algorytmu na zasadzie $\alpha_{ki} = \frac{i}{\sqrt{k}}$.

Na kolejnych wykresach na Rysunkach (3.2) - (3.5) osie dopasowane są do obecnej symulacji. Ze względu na pomniejszenie zakresu osi, nie zdecydowano się na ukazanie warstwic.

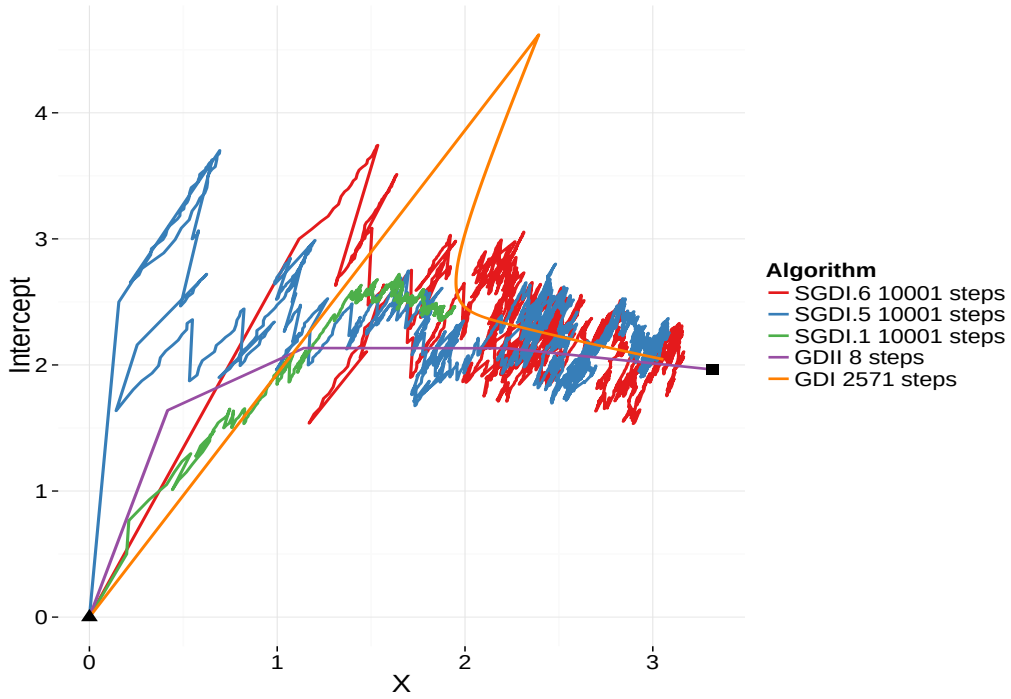
Podsumowanie symulacji

W trackie symulacji badano zachowanie trajektorii zbieżności w kolejnych krokach algorytmów spadku gradientu. Algorytm spadku gradientu rzędu II zbiegał do rozwiązania wyznaczonego przez funkcję `glm` [35] 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.

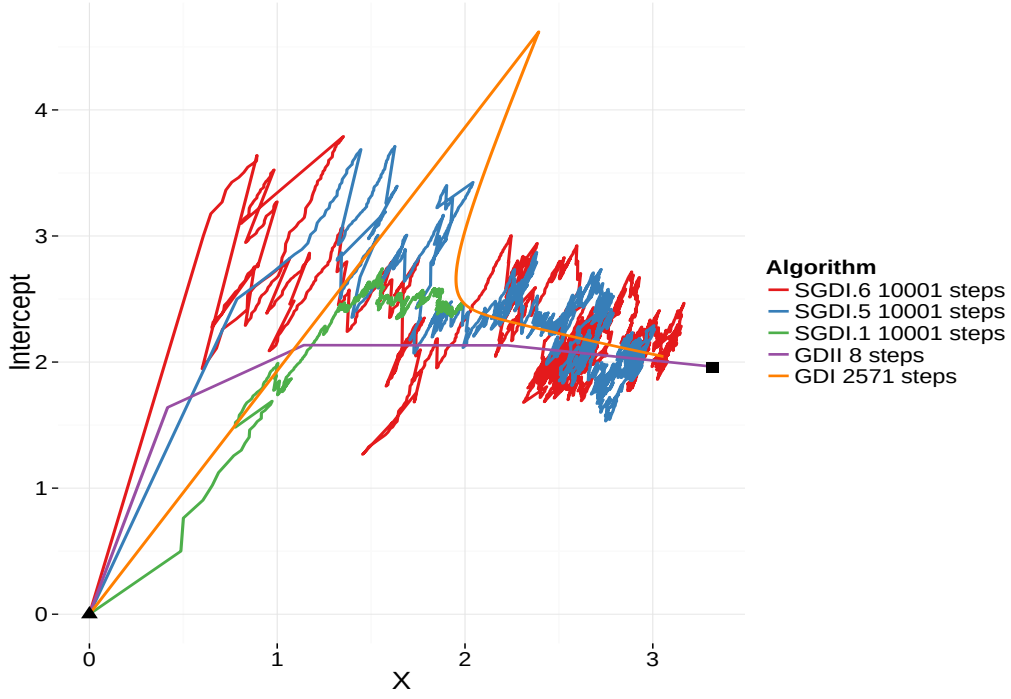
Algorytm spadku gradientu rzędu I zbiegał do rozwiązania wyznaczonego przez funkcję `glm` w sytuacjach, gdy warunek stopu algorytmu był dostatecznie mały ($epsilon = 10^{-5}$, Rysunek 3.2) oraz był na dobrej drodze do rozwiązania jednak nie dotarł do tych samych punktów co algorytm spadku gradientu rzędu II z racji na zbyt duży warunek stopu ($epsilon = 10^{-4}$, Rysunki 3.3 i 3.5). Dla punktu startowego bliskiego rozwiązaniu pochodzącemu z wyliczeń z funkcji `glm` i dużego warunku stopu algorytmu algorytm spadku gradientu rzędu I miał problem z obraniem właściwego toru zbieżności do rozwiązania i ostatecznie wypadł najgorzej z wszystkich sprawdzanych algorytmów. Estymacja w tym wypadku dla algorytmu 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 znalezionego 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 ukazać w pracy ciągi, dla których osiągana była zbieżność dla największej liczby punktów startowych. Algorytm `SGDI.1` w każdym z możliwych przypadków miał zbyt małe wartości ciągu odpowiadającego za długości kroków algorytmu przez co nie dochodził do punktu wyliczonego przez funkcję `glm`, a ukazany został aby uświadomić jak ważnym aspektem jest odpowiednie dobranie długości kroków. Dla odpowiednio dobranego ciągu długości kroków o dużych wartościach jakim był ciąg $\alpha_{ki} = \frac{i}{\sqrt{k}}, k = 5, 6$ w wielu sytuacjach doszło do zbieżności w tym samym punkcie, w którym zbiegł algorytm zaimplementowany w funkcji `glm`. Najlepiej widać to na Rysunku 3.2 ($epsilon = 10^{-5}$) oraz Rysunku 3.3 ($epsilon = 10^{-4}$), gdzie w zbieżności algorytmu przeszkodził zbyt duży warunek stopu. Ciekawy rezultat osiągnięto na Rysunku 3.4, gdzie dla różnych początkowych ziaren losowości osiągnięto zbieżność do punktu wyliczonego przez funkcję `glm` za każdym razem dla innego z dwóch poważnie rozważanych przypadków `SGDI.5`, `SGDI.6`. Istota losowości odgrywa kluczową rolę w algorytmie stochastycznego spadku gradientu i należy mieć świadomość, że dla różnych ziaren losowości, a co za tymi idzie dla innej kolejności obserwacji uwzględnianych w kolejnych krokach optymalizacji, algorytm może zachowywać się różnie. Doskonale widać to na Rysunku 3.5, gdzie dla Symulacji nr 1 jeden przypadek algorytmu stochastycznego spadku gradientu osiągnął lepsze wyniki niż spadek gradientu rzędu I, zaś dla Symulacji nr 2 żadna wersja stochastyczna nie miała trajektorii zgodnych z prawdziwym kierunkiem zbieżności.

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

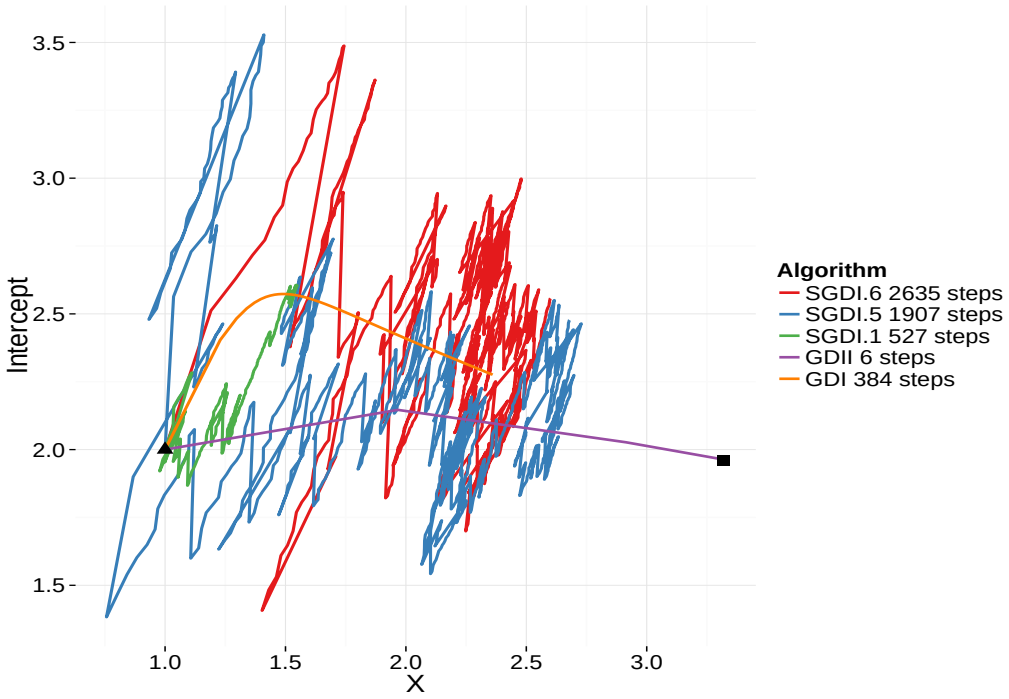


(a) Symulacja dla $\beta_0 = (0, 0)$ nr 1 dla ziarna losowania ustalonego na 4561.

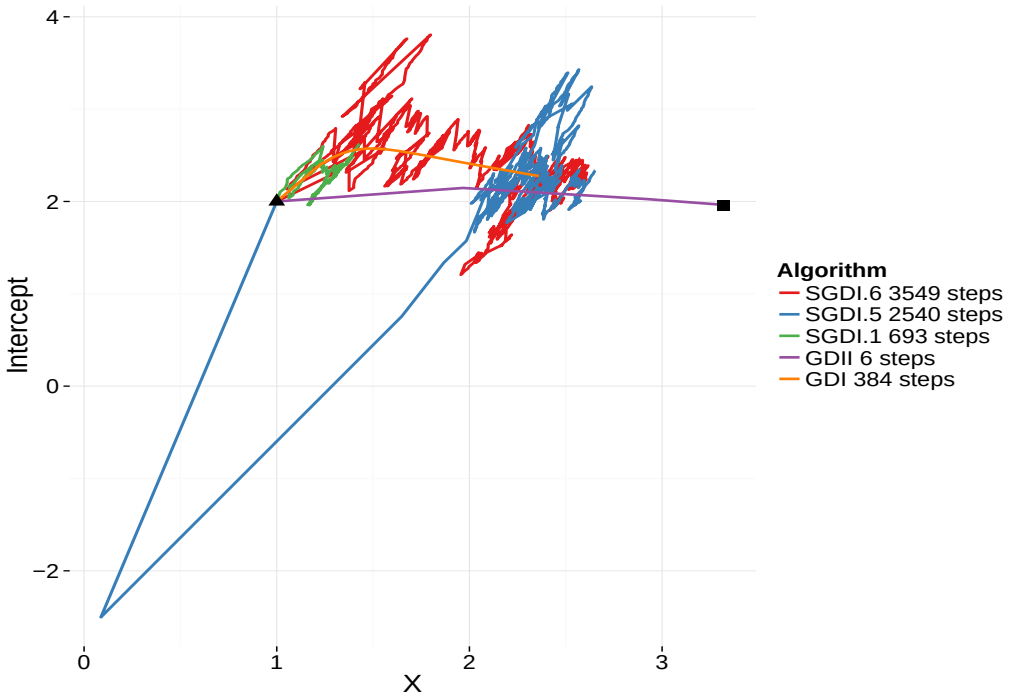


(b) Symulacja dla $\beta_0 = (0, 0)$ nr 2 dla ziarna losowania ustalonego na 456.

Rysunek 3.2: Porównanie algorytmów spadku gradientu dla punktu startowego $\beta_0 = (0, 0)$. Wykresy przedstawiają ścieżki zbieżności w kolejnych krokach algorytmów spadku gradientu. Ustalono maksymalną liczbę iteracji na 10001, zaś warunek stopu ustalono na $\epsilon = 10^{-5}$. Trójkątem zaznaczono punkt startowy, a kwadratem wyestymowane rozwiązanie przy pomocy funkcji `glm` [35]. Przez `GDI` oznaczono trajektoria dla algorytmu spadku gradientu rzędu I, przez `GDII` oznaczono trajektoria dla algorytmu spadku gradientu rzędu II, zaś przez `SGD.i` oznaczono 3 różne trajektorie dla algorytmów stochastycznego spadku gradientu dla różnych ciągów odpowiadających długości kroku algorytmu. Indeks i odpowiada ciągowi wybranemu do wyznaczania długości kroku algorytmu na zasadzie $\alpha_{ki} = \frac{i}{\sqrt{k}}$.

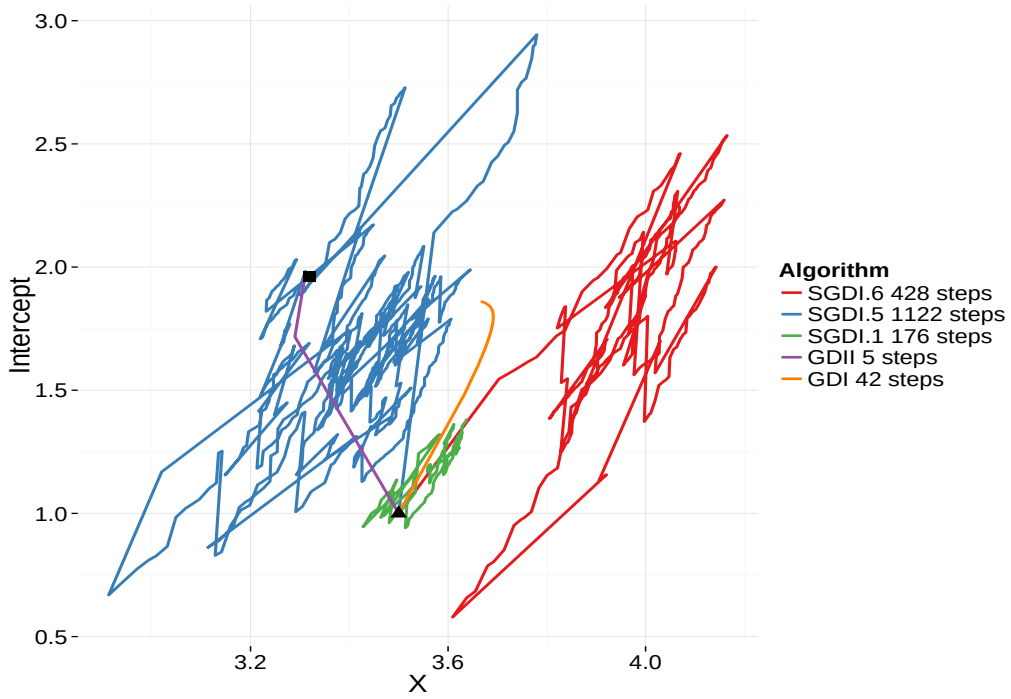


(a) Symulacja dla $\beta_0 = (1, 2)$ nr 1 dla ziarna losowania ustalonego na 4561.

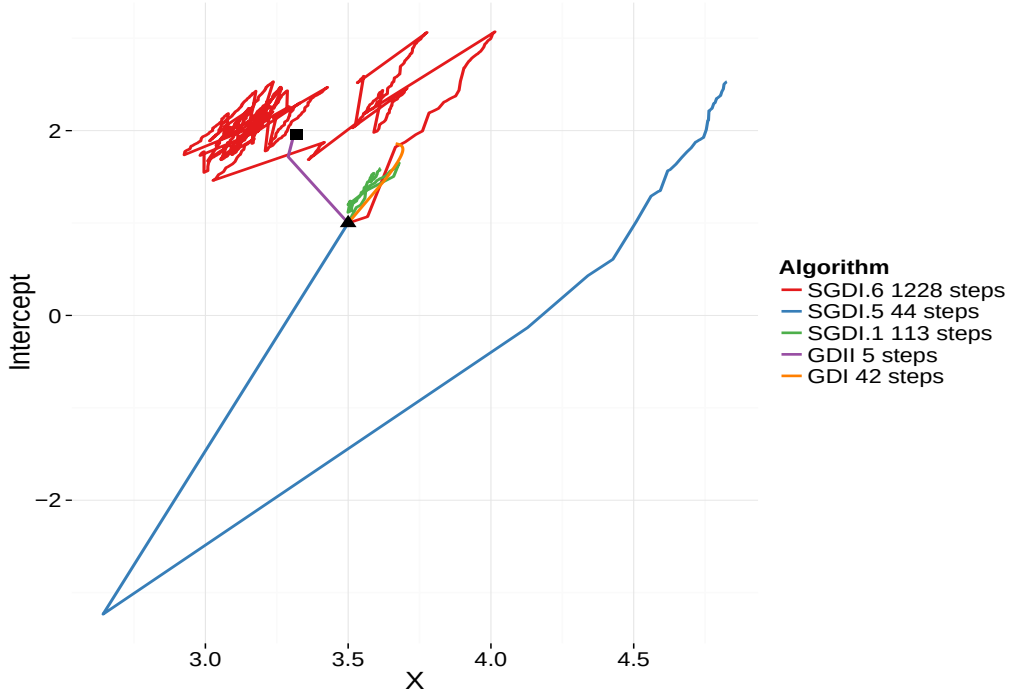


(b) Symulacja dla $\beta_0 = (1, 2)$ nr 2 dla ziarna losowania ustalonego na 456.

Rysunek 3.3: Porównanie algorytmów spadku gradientu dla punktu startowego $\beta_0 = (1, 2)$. Wykresy przedstawiają ścieżki zbieżności w kolejnych krokach algorytmów spadku gradientu. Ustalono maksymalną liczbę iteracji na 10001, zaś warunek stopu ustalono na $\epsilon = 10^{-4}$. Trójkątem zaznaczono punkt startowy, a kwadratem wyestymowane rozwiązanie przy pomocy funkcji `glm` [35]. Przez `GDI` oznaczono trajektoria dla algorytmu spadku gradientu rzędu I, przez `GDII` oznaczono trajektoria dla algorytmu spadku gradientu rzędu II, zaś przez `SGD.i` oznaczono 3 różne trajektorie dla algorytmów stochastycznego spadku gradientu dla różnych ciągów odpowiadających długości kroku algorytmu. Indeks i odpowiada ciągowi wybranemu do wyznaczania długości kroku algorytmu na zasadzie $\alpha_{ki} = \frac{i}{\sqrt{k}}$.

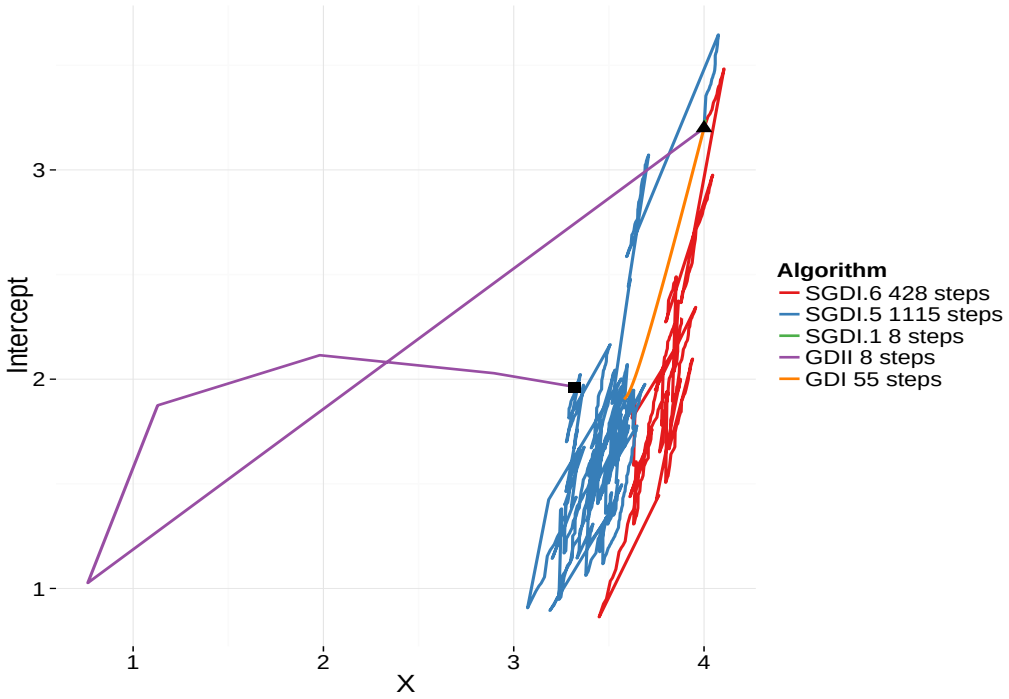


(a) Symulacja dla $\beta_0 = (3.5, 1)$ nr 1 dla ziarna losowania ustalonego na 4561.

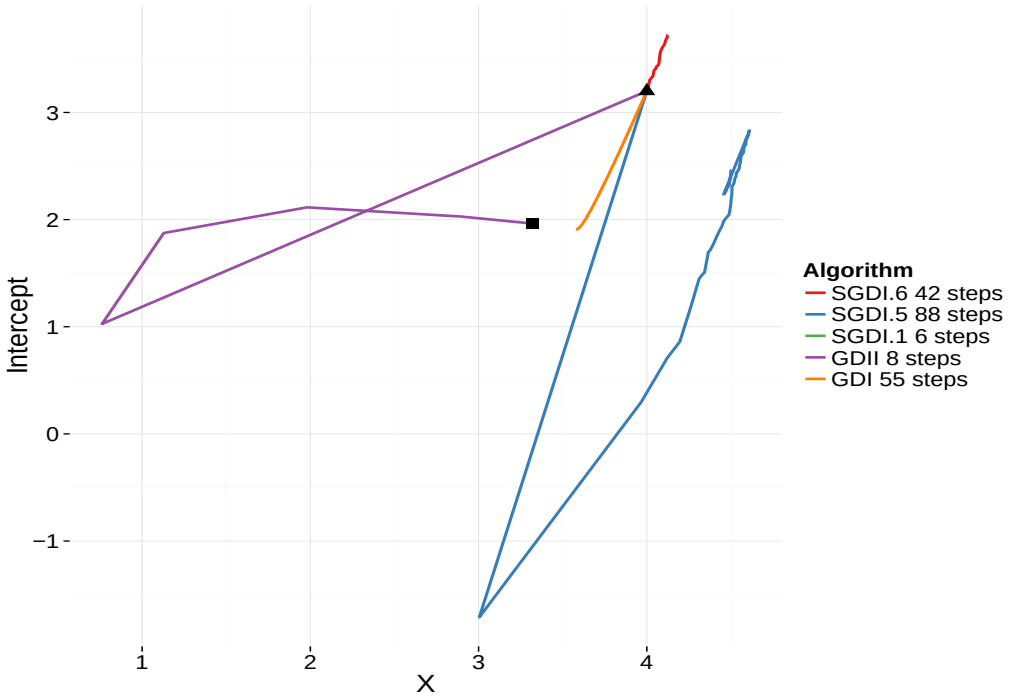


(b) Symulacja dla $\beta_0 = (3.5, 1)$ nr 2 dla ziarna losowania ustalonego na 456.

Rysunek 3.4: Porównanie algorytmów spadku gradientu dla punktu startowego $\beta_0 = (3.5, 1)$. Wykresy przedstawiają ścieżki zbieżności w kolejnych krokach algorytmów spadku gradientu. Ustalono maksymalną liczbę iteracji na 10001, zaś warunek stopu ustalono na $\epsilon = 10^{-4}$. Trójkątem zaznaczono punkt startowy, a kwadratem wyestymowane rozwiązanie przy pomocy funkcji `glm` [35]. Przez `GDI` oznaczono trajektoria dla algorytmu spadku gradientu rzędu I, przez `GDII` oznaczono trajektoria dla algorytmu spadku gradientu rzędu II, zaś przez `SGD.i` oznaczono 3 różne trajektorie dla algorytmów stochastycznego spadku gradientu dla różnych ciągów odpowiadających długości kroku algorytmu. Indeks i odpowiada ciągowi wybranemu do wyznaczania długości kroku algorytmu na zasadzie $\alpha_{ki} = \frac{i}{\sqrt{k}}$.



(a) Symulacja dla $\beta_0 = (4, 3.2)$ nr 1 dla ziarna losowania ustalonego na 4561.



(b) Symulacja dla $\beta_0 = (4, 3.2)$ nr 2 dla ziarna losowania ustalonego na 456.

Rysunek 3.5: Porównanie algorytmów spadku gradientu dla punktu startowego $\beta_0 = (4, 3.2)$. Wykresy przedstawiają ścieżki zbieżności w kolejnych krokach algorytmów spadku gradientu. Ustalono maksymalną liczbę iteracji na 10001, zaś warunek stopu ustalono na $\epsilon = 10^{-4}$. Trójkątem zaznaczono punkt startowy, a kwadratem wyestymowane rozwiązanie przy pomocy funkcji `glm` [35]. Przez `GDI` oznaczono trajektoria dla algorytmu spadku gradientu rzędu I, przez `GDII` oznaczono trajektoria dla algorytmu spadku gradientu rzędu II, zaś przez `SGD.i` oznaczono 3 różne trajektorie dla algorytmów stochastycznego spadku gradientu dla różnych ciągów odpowiadających długości kroku algorytmu. Indeks i odpowiada ciągowi wybranemu do wyznaczania długości kroku algorytmu na zasadzie $\alpha_{ki} = \frac{i}{\sqrt{k}}$.

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. Poniższe rozważania odnośnie podejścia do stosowania tej metody w tym konkretnym modelu nie są oparte na żadnej literaturze ze względu na jej brak.

W czasie powstawania pracy nawiązano jedynie nieznaczną 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 niedokończonej implementacji algorytmu przez pracowników wyżej wymienionego laboratorium można znaleźć w [73].

W procesie estymacji współczynników w omawianym modelu wykorzystano pochodne cząstkowe częściowej funkcji log-wiarogodności (2.10)

$$U_k(\beta) = \frac{\partial \ell_k(\beta)}{\partial \beta_k} = \sum_{i=1}^{K} \Big(X_{ik} - \frac{\sum\limits_{l \in \mathscr{R}(t_i)} X_{lk} e^{X_l' \beta}}{\sum\limits_{l \in \mathscr{R}(t_i)} e^{X_l' \beta}} \Big) = \sum_{i=1}^{n} Y_i \Big(X_{ik} - \frac{\sum\limits_{l \in \mathscr{R}(t_i)} X_{lk} e^{X_l' \beta}}{\sum\limits_{l \in \mathscr{R}(t_i)} e^{X_l' \beta}} \Big) = \sum_{i=1}^{n} U_{k_i}(\beta),$$

(dla $Y_i = 1$, 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 dla powyższej funkcji wygląda następująco

$$\beta_{k_{j+1}} = \beta_{k_j} - \alpha_j U_{k_i}(\beta_{k_j}), \tag{4.1}$$

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 cząstkowa gradientu U oraz β_k to k-ta współrzędna estymowanego wektora współczynników β o rozmiarze p, czyli $k = 1, \ldots, p$.

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

4.1. Założenia i obserwacje

Skupiając 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 uczony 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 dla modelu Coxa nie, gdyż niektóre składniki w funkcji log-wiarogodności są zależne od poprzednich obserwacji. W przypadku modelu ADALINE sformułowanego jak w [86], opartego na minimalizowaniu funkcji kosztu w postaci błędu najmniejszych kwadratów, w [9] podano postać funkcji straty oraz równanie algorytmu stochastycznego spadku gradientu jak poniżej

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

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

dla 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. Dla zadanej z góry funkcji 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 dla składników częściowej funkcji log-wiarogodności dla wszystkich zaobserwowanych czasów zdarzeń i poprawianie odpowiednich składników z wykorzystaniem świeżo zaobserwowanych obserwacji. Taki zabieg jest pamięciowo oszczędniejszy niż pamiętanie wszystkich obserwacji. **Możliwe jest też 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 mamy do czynienia z nieskończonym napływem nowych obserwacji a interesują nas oszacowania estymowanych parametrów modelu $\beta_k, k = 1, \ldots, p$ 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 rozwiązują 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 dla j-ego kroku algorytmu i 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

$$-U_k^{\mathcal{B}}(\beta_j) = -\sum_{i \in \mathcal{B}_{\text{ind}}} U_{k_i}^{\mathcal{B}}(\beta_j) = -\sum_{i \in \mathcal{B}_{\text{ind}}} Y_i \Big(X_{ik} - \frac{\sum\limits_{l \in \mathscr{R}_{\mathcal{B}}(t_i)} X_{lk} e^{X_l' \beta_j}}{\sum\limits_{l \in \mathscr{R}_{\mathcal{B}}(t_i)} e^{X_l' \beta_j}} \Big), \tag{4.2}$$

gdzie indeksy obserwacji należące do zbioru $\mathcal{B}$ definiuje się jako $\mathcal{B}_{\text{ind}} = \{i : X_i \in \mathcal{B}\}$, zaś $\mathscr{R}_{\mathcal{B}}(t_i)$ to zbiór ryzyka dla podzbioru $\mathcal{B}$ w czasie t_i.

Postać powyższej funkcji nasuwa pewne obserwacje. Dla danego 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. 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 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, co da ostatecznie zerową wartość pochodnej cząstkowej optymalizowanej funkcji log-wiarogodności, co 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, można 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 dla modelu proporcjonalnych hazardów Coxa została zaimplementowana w języku $\mathcal{R}$ [67] i jest dostępna w specjalnie przygotowanym pakiecie o nazwie `coxphSGD()`, który można pobrać z internetu i zainstalować poleceniem

```
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` [75], [76].

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 dlugosci 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 częściowa funkcja
                                        # 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 ) {
  checkArguments(formula, data, learningRates,
                 beta_0, epsilon) -> beta_start # check arguments
  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 tracked 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)]])
    diff <- sqrt(sum((beta_new[[i]] - beta_old)^2))
    beta_old <- beta_new[[i]]
    i <- i + 1
  }
  # 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
  partial_sum <- list()
  for(k in 1:nrow(batchData)) {
    # risk set for current time/observation
    risk_set <- batchData %>% filter(times >= batchData$times[k])

    nominator <- apply(risk_set[, -c(1,2)], MARGIN = 1, function(element){
      element * exp(element * beta)
    }) %>% rowSums()

    denominator <- apply(risk_set[, -c(1,2)], MARGIN = 1, function(element){
      exp(element * beta)
    }) %>% rowSums()

    partial_sum[[k]] <-
      batchData[k, "event"] * (batchData[k, -c(1,2)] - nominator/denominator)
  }
  do.call(rbind, partial_sum) %>%
    colSums() -> U_batch

  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 z ze sztucznie wygenerowanymi czasami z rozkładu Weibulla o parametrach $\lambda = 3$ oraz $\rho = 2$. Czasy cenzurowania generowane są z rozkładu wykładniczego o parametrze 5 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 = 5)
```

Funkcja `simulateCoxSGD()` opisana w Dodatku A.1 sześciokrotnie dzieli wygenerowany zbiór danych odpowiednio na 10, 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 dla modelu Coxa przy pomocy metody stochastycznego spadku gradientu. Dzięki tak otrzymany współczynnikom modelu w kolejnych krokach algorytmu dla różnych podziałów zbiorów, możliwe było graficzne przedstawienie trajektorii zbieżności algorytmu w zależności od ustawionego punktu startowego, warunku stopu oraz wartości ciągu odpowiadającego długościom kroków w algorytmie.

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 gdy jest większa od liczby zaobserwowanych podzbiorów powoduje, że w trakcie procesu optymalizacji algorytm przechodzi do nowej epoki.

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 dla zaobserwowanego podzbioru obserwacji w modelu Coxa proporcjonalnych hazardów, 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 przedstawiono warstwice częściowej funkcji log-wiarogodności wyliczone dla wszystkich zaobserwowanych podzbiorów połączonych w jeden. Trójkątem zaznaczono punkt startowy algorytmu, kwadratem zaś teoretyczne maksimum.

Podsumowanie symulacji dla modelu proporcjonalnych hazardów Coxa

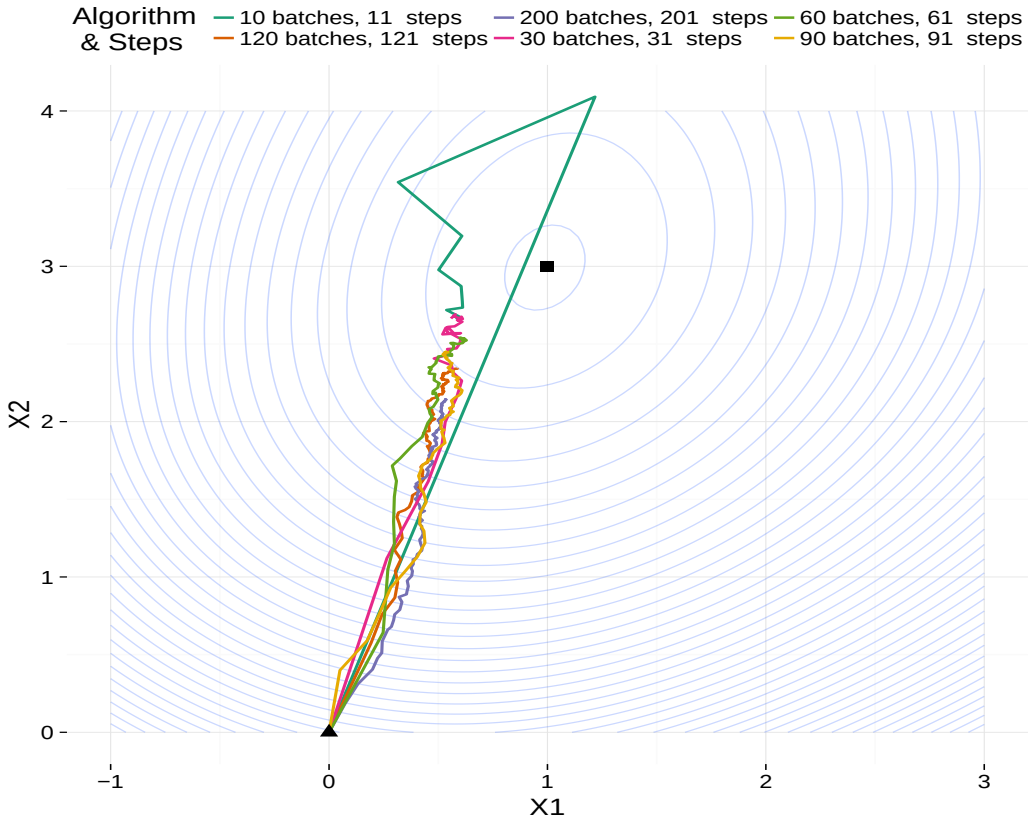
W trakcie symulacji badano jak podział całego zbioru obserwacji na podzbiory wpływa na trajektoria optymalizacji częściowej funkcji log-wiarogodności w modelu Coxa proporcjonalnych hazardów dla zaobserwowanego podzbioru. Badano również wpływ punktu początkowego, warunku stopu, liczby epok oraz konstrukcji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

Na Rysunkach (4.1) - (4.4) zaprezentowano trajektoria dla punktów początkowych: $\beta_0 = (0,0), (2,2), (-1,4)$. Brano pod uwagę ciągi odpowiadające długościom kroków równe $\frac{1}{20\sqrt{t}}, \frac{1}{50\sqrt{t}}, \frac{1}{100\sqrt{t}}$, eksperymentowano z liczbą epok ustalaną na $1, 5, 10$ oraz sprawdzano jak zachowają się trajektoria dla warunków stopu $\epsilon = 10^{-6}, \epsilon = 10^{-5}$. Dla ciągów odpowiadającym długościom kroków ustalanym na $\frac{1}{20t}, \frac{1}{50t}, \frac{1}{100t}$ algorytm nie osiągał zbieżności i zatrzymywał się po maksymalnej liczbie iteracji w losowym punkcie.

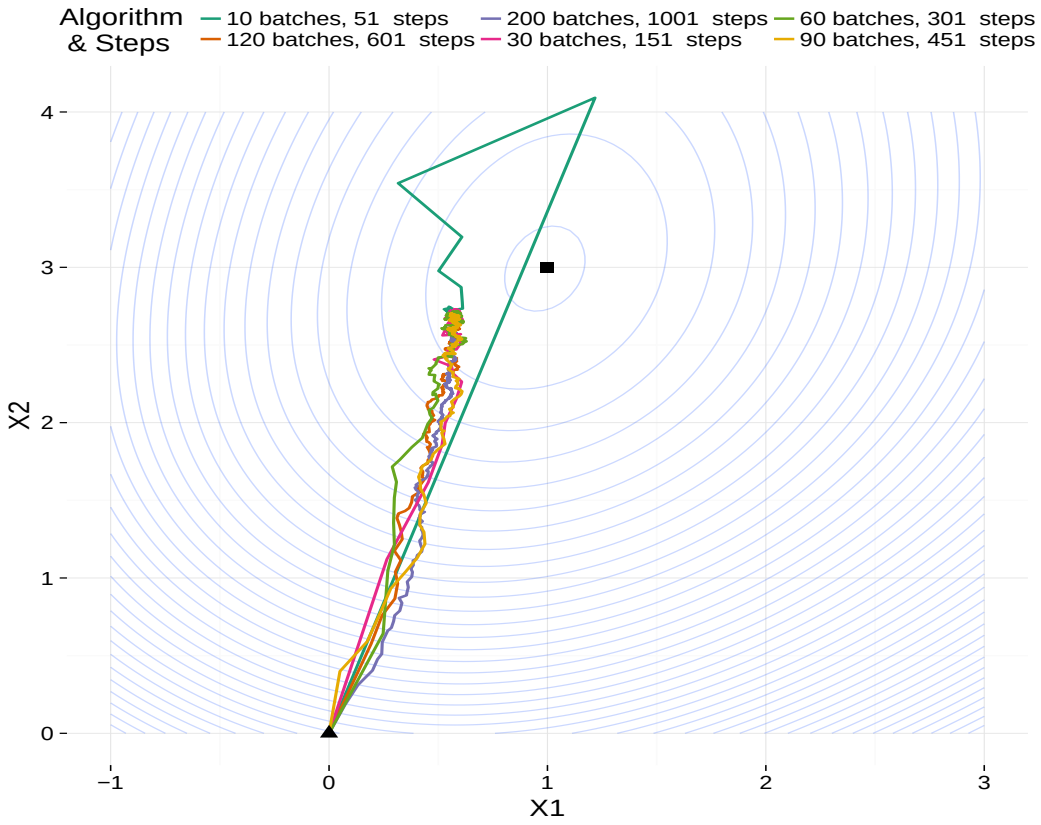
Dla żadnej z symulacji algorytm nie zbiegł do punktu, w którym powinno być teoretyczne optimum. Może to wynikać z dużej liczby obserwacji cenzurowanych bądź z samej istoty losowości przy procesie generowania danych. Należy podkreślić, że dla różnych sprawdzanych wariantów procesu optymalizacji za każdym razem metoda zbiegała do tego samego punktu. Może być to optimum wynikające z postaci danych. Stochastyczny spadek gradientu nie wykorzystuje wszystkich obserwacji jednocześnie przez co trudniej zbiec do teoretycznego optimum. Warto zauważyć stabilność procesu optymalizacji dla ciągów odpowiadających za długość kroków w algorytmie, które ustawione zostały na mniejsze wartości $\frac{1}{50\sqrt{t}}, \frac{1}{100\sqrt{t}}$ (Rysunki (4.1) - (4.3)) - w przypadku ciągu o zbyt dużych wartościach początkowych ($\frac{1}{20\sqrt{t}}$) trajektoria algorytmu we wczesnej fazie procesu były dalekie od optimum (Rysunek (4.4)). Dzięki ukazanym warstwicom funkcji częściowej log-wiarogodności skonstruowanym na całych danych, widać, że poczynając od punktu startowego algorytm w różnych wydaniach postępuje we właściwym kierunku zbieżności i kieruje się w okolice teoretycznego optimum. Algorytm zatrzymuje się niedaleko teoretycznego optimum, jednak patrząc na rozrzedzone zagęszczenie warstwic w okolicach teoretycznego optimum, można stwierdzić, że wartości funkcji częściowej log-wiarogodności nie różnią się już na tyle, aby proces optymalizacji mógł być kontynuowany.

Warto podkreślić, że większa liczba epok i mniejszy warunek stopu powodują jedynie poprawienie optymalizacji dla wariantów, w których cały zbiór obserwacji był podzielony na największą liczbę obserwacji, a więc każdy krok algorytmu budowany był na najmniej licznych podzbiorach.

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 dla tylko jednej epoki. Dla dobrze dobranych parametrów 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 różne ciągi oraz wybranie tego ciągu, który w większości przypadków daje stabilne, jednorodne współczynniki.

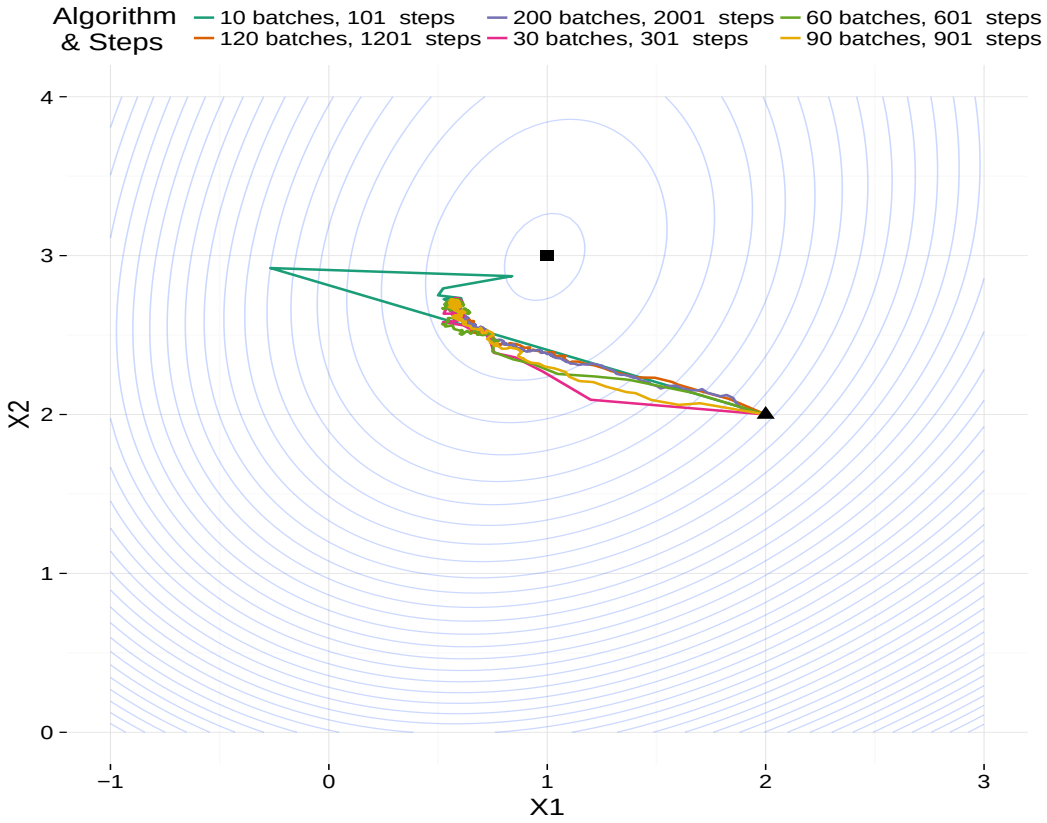


(a) $\beta_0 = (0, 0)$, liczba epok 1, warunek stopu: $\epsilon = 10^{-6}$, długości kroków $= \frac{1}{50\sqrt{t}}$.

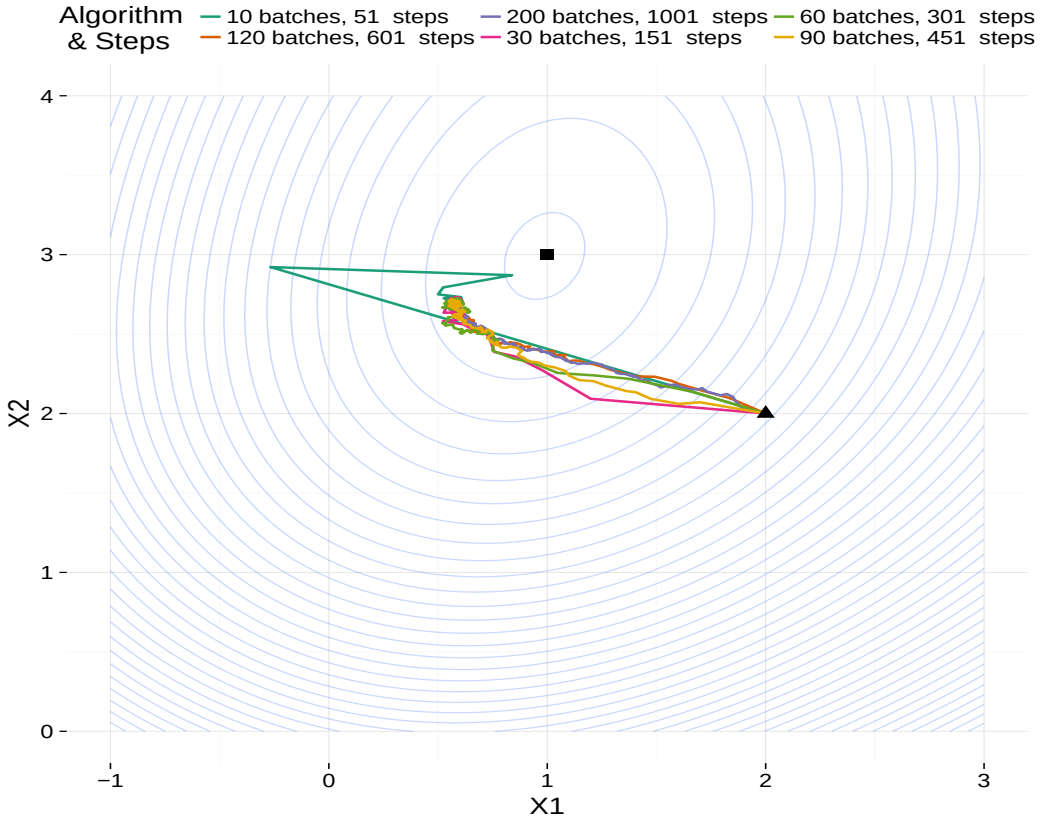


(b) $\beta_0 = (0, 0)$, liczba epok 5, warunek stopu: $\epsilon = 10^{-6}$, długości kroków $= \frac{1}{50\sqrt{t}}$.

Rysunek 4.1: Porównanie estymacji w modelu Coxa metodą stochastycznego spadku gradientu dla różnych podziałów zbioru początkowego na podzbiory.

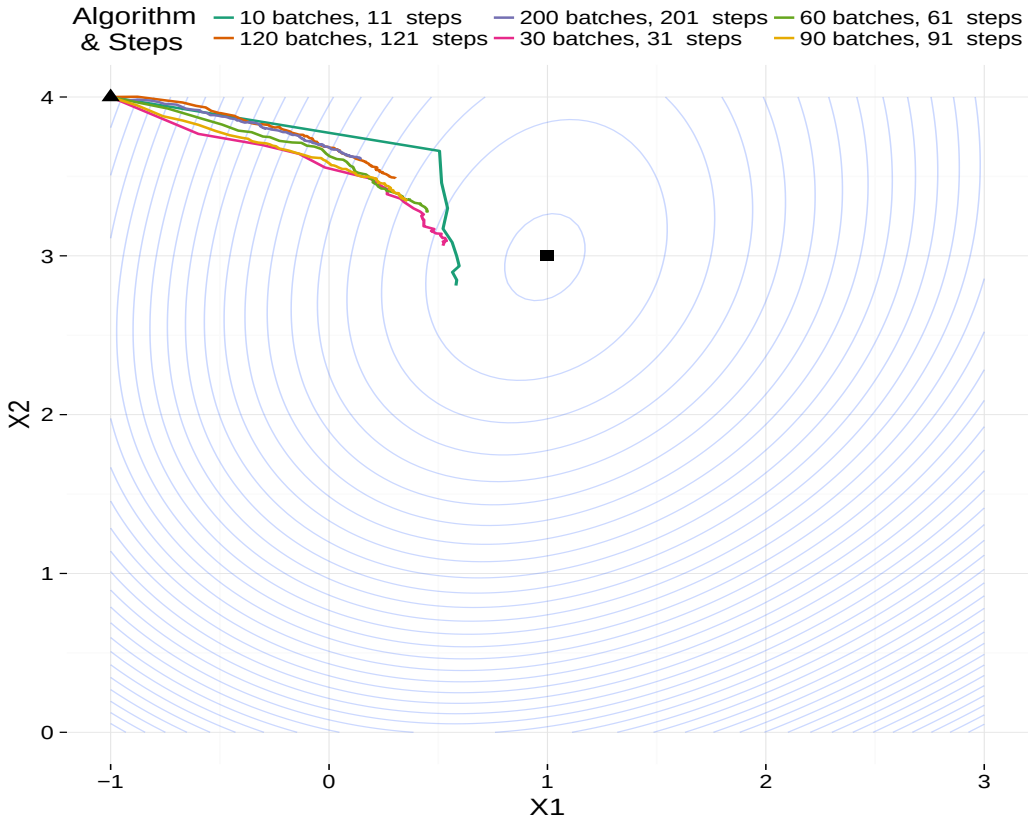


(a) $\beta_0 = (2, 2)$, liczba epok 10, warunek stopu: $\epsilon = 10^{-6}$, długości kroków $= \frac{1}{50\sqrt{t}}$.

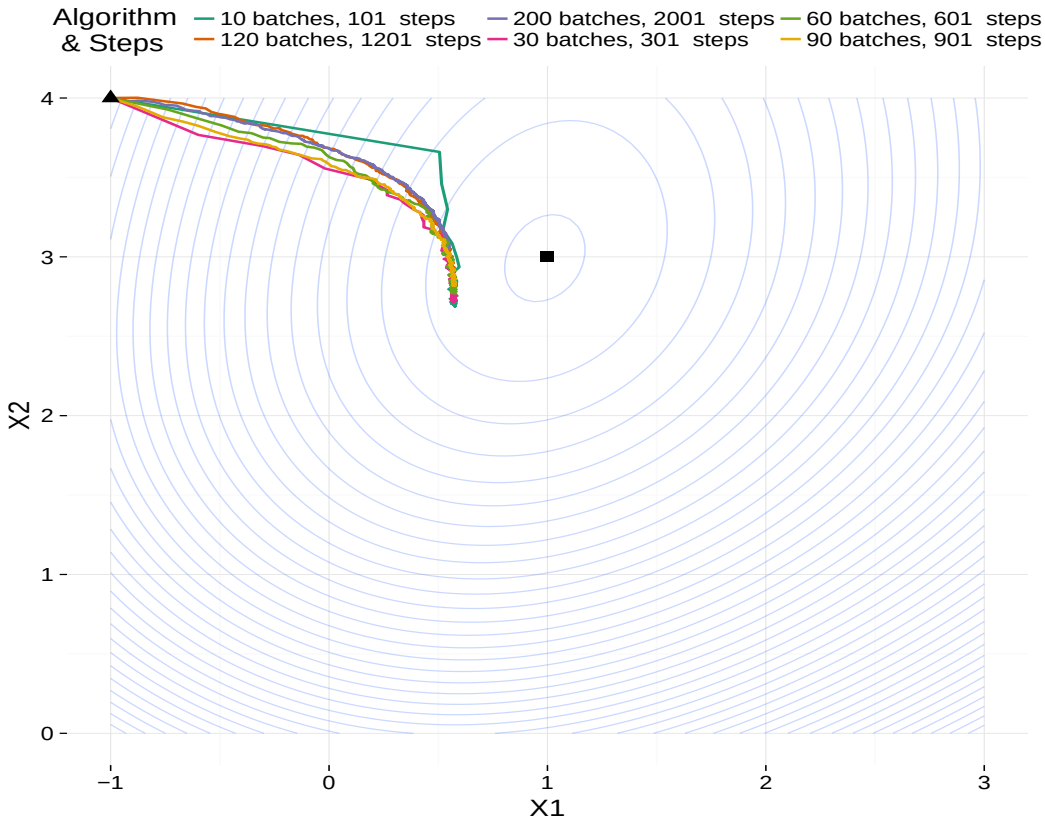


(b) $\beta_0 = (2, 2)$, liczba epok 5, warunek stopu: $\epsilon = 10^{-5}$, długości kroków $= \frac{1}{50\sqrt{t}}$.

Rysunek 4.2: Porównanie estymacji w modelu Coxa metodą stochastycznego spadku gradientu dla różnych podziałów zbioru początkowego na podzbiory.

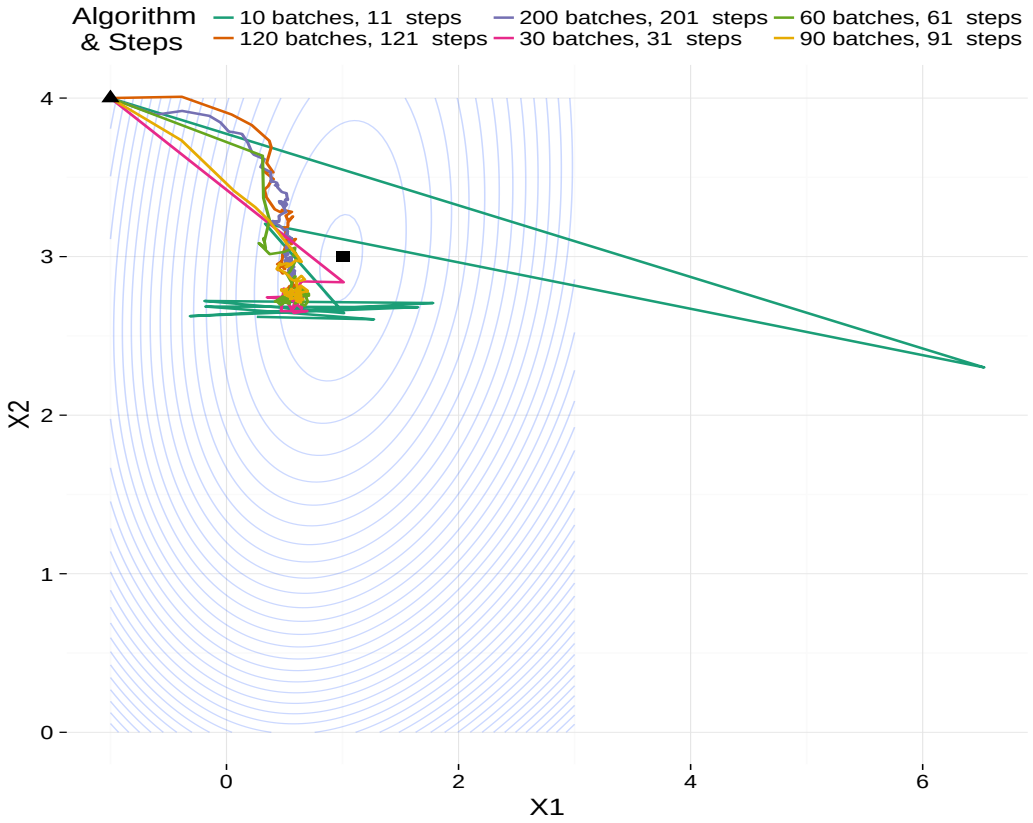


(a) $\beta_0 = (-1, 4)$, liczba epok 1, warunek stopu: $\epsilon = 10^{-6}$, długości kroków $= \frac{1}{100\sqrt{t}}$.

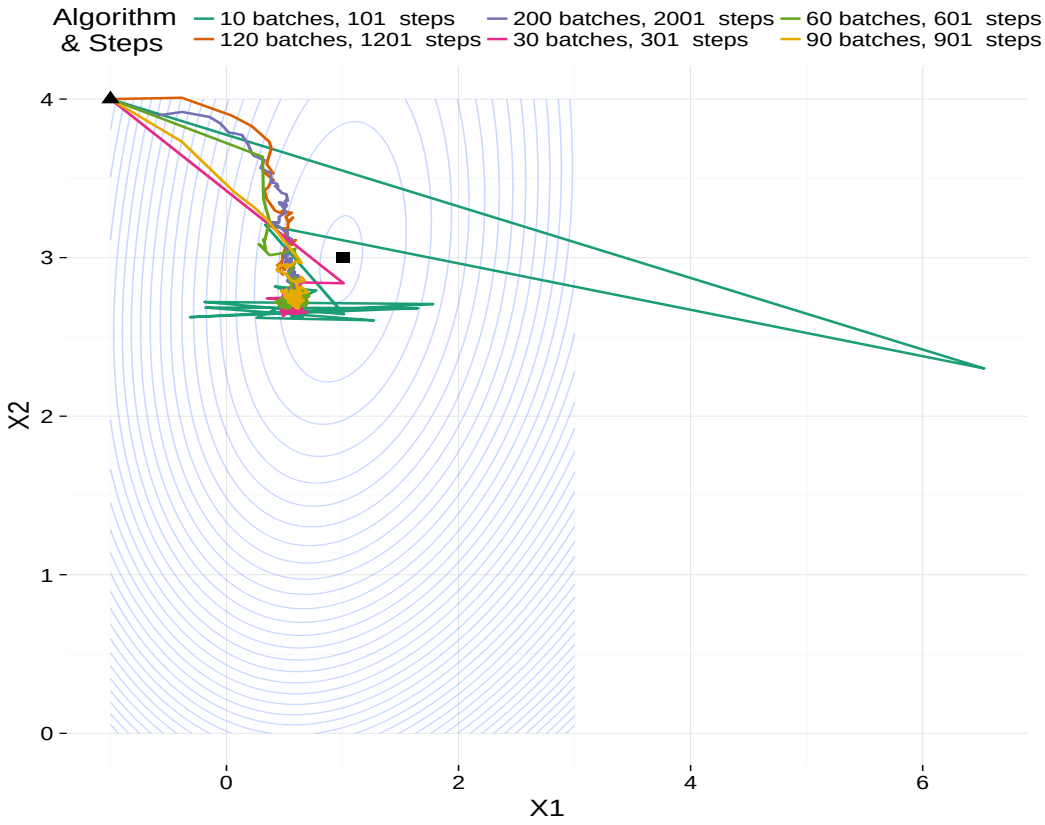


(b) $\beta_0 = (-1, 4)$, liczba epok 10, warunek stopu: $\epsilon = 10^{-5}$, długości kroków $= \frac{1}{100\sqrt{t}}$.

Rysunek 4.3: Porównanie estymacji w modelu Coxa metodą stochastycznego spadku gradientu dla różnych podziałów zbioru początkowego na podzbiory.



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



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

Rysunek 4.4: Porównanie estymacji w modelu Coxa metodą stochastycznego spadku gradientu dla różnych podziałów zbioru początkowego na podzbiory.

Rozdział 5

Analiza danych genomicznych

Tke 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 pokrótce 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, dla którego estymacja współczynników odbywa się metodą stochastycznego spadku gradientu (opisaną 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 analizy było uzyskanie odpowiedzi na pytanie czy istnieją geny, które zmutowane powodują, że ryzyko wystąpienia zdarzenia jakim jest zgon w wyniku choroby nowotworowej jest większe w grupach pacjentów bez mutacji w danym genie. Aby zweryfikować hipotezę na tak obszernych danych zdecydowano się zastosować metodę strumieniowej estymacj współczynników w modelu Coxa przy pomocy stochastycznego spadku gradientu, której stworzenie było głównym założeniem pracy, gdyż tradycyjna metoda zaimplementowana w funkcji `coxph` w pakiecie `survival` [76] nie jest przystosowana do tak dużych zbiorów danych jak te z TCGA. Z racji na zbyt dużą ilość 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 nie tylko zdobycz obliczeniowa ale także ważne narzędzie w analizach biostatystycznych danych dużej skali, które z powodzeniem może zostać wykorzystywane do analizy ludzkiego genomu, tak by w przyszłości poszerzać zrozumienie procesów mutacji zachodzących w ludzkim organizmie, dzięki którym możliwe byłoby skuteczniejsze leczenie chorób nowotworowych.

5.1. Genetyczne podstawy nowotworzenia

Choroby nowotworowe stanowią drugą, po chorobach serca, przyczynę zachorowań i zgonów na całym świecie. Wyniki opracowane przez Polską Unię Onkologiczną wskazują na tendencję wzrostową liczby zachorowań na nowotwory i rokują na utrzymanie się jej do 2020 [66].

Według współczesnej definicji **nowotwór** jest *chorobą cyklu komórkowego* i oznacza [23]

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.

Wyniki badań nad transformacją nowotworową wykazały, że nowotwory powstają jako wynik wielu mutacji w DNA komórki somatycznej, które poprzez akumulację wywołują utratę kontroli nad proliferacją (rozwojem), wzrostem i różnicowaniem [52].

Proces tworzenia nowotworu (**kancerogeneza**) jest wieloczynnikowy i wielostopniowy, a zmiany w nim nasilają się w miarę pogłębiania się niestabilności genetycznej [23]. Transformacja nowotworowa następuje w wyniku zmian powstałych w obrębie czterech różnych kategorii genów, które wpływają na proliferację i różnicowanie komórek [66]. Czynniki genetyczne zaangażowane w systemy naprawy DNA, a także w procesy proliferacji i różnicowania komórek [40] można podzielić na:

- *geny regulujące naprawę uszkodzonego DNA* - mechanizmy szybkiej naprawy DNA zapobiegają przed mutacjami genów odpowiedzialnyh 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, zapobiegają niekontrolowanym podziałom komórkowym oraz są czynnikami negatywnej kontroli podziałów.

- *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) [39].

5.2. Projekt The Cancer Genome Atlas

Do analizy wykorzystano dane udostępniane w ramach *The Cancer Genome Atlas* [11]. The Cancer Genome Atlas (TCGA) to kompleksowy projekt o skoordynowanych wysiłkach, mający na celu przyśpieszenie zrozumienia molekularnych podstaw nowotworu. Ma się to odbywać poprzez stosowanie technologii analizy danych dużej skali do udostępnianych danych genomicznych 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 [77], 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, początkowo jako projekt pilotażowy, opracowało i przetestowało strukturę badawczą konieczną do systematycznego badania całego spektrum zmian genomu zawartych w ponad 20 rodzajach raka.

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 [77]. 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 zawartych w TCGA można znaleźć w [14], [15], [74] czy [77].

Dane z TCGA pobrano przy użyciu pakietu `RTCGA` [48] i umieszczono w pakietach

- `RTCGA.clinical` [49], zawierającym dane kliniczne o pacjentach,
- `RTCGA.mutations` [50], 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.

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 umieszczono w badaniu dane o mutacjach. Część wspólną nazw dla kohort zawierających zarówno dane kliniczne oraz dane o mutacjach wygenerowaną dzięki wywołaniu

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

Następnie dla tak otrzymanych 35 kohort nowotworowych uzyskano dane o statusie pacjenta (śmierć bądź cenzurowanie) oraz jego czasie spędzonym pod obseracją dzięki funkcji

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

Dane o mutacjach występujących wśród tkanek nowotworowych kolejnych pacjentów uzyskano za pomocą

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

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

Dla tak otrzymanych dwóch zbiorów danych połączono dla pacjentów informacje kliniczne z informacjami o mutacjach dzięki przypisanym do pacjentów i ich próbek kodów `bcr_patient_barcode`, by ostatecznie podzielić zbiór pacjentów na 100 losowo utworzonych grup.

```
prepareCoxDataSplit(mutationsData,survivalData, groups = 100) -> coxData_split
```

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

```
prepareForumlaSGD(coxData) -> formulaSGD
```

Ostatecznie dla 6459 pacjentów, którzy posiadali informacje o występujących mutacjach, oraz dla 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.

```
coxphSGD(formulaSGD, data = coxData_split, max.iter = 100)
```

Niemożliwe było sprawdzenie założeń modelu dotyczących proporcjonalności hazardu, gdyż zakładano napływającą postać danych (stąd podział danych na 100 grup). Dla takiej postaci pojawia się danych ciężko także mówić o jakiejkolwiek diagnostyce poprawności dopasowania modelu i dokładności otrzymanych wpsółczynników. Nie stworzono teorii pozwalającej badać istotność statystyczna otrzymanych współczynników w modelu, jednak założono, że współczynniki dostatecznie odległe od 0 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 mutacji w danym genie. Współczynniki ujemne oznaczają zmniejszenie hazardu pacjenta posiadającego mutację w danym genie w stosunku do pacjentów nie posiadających mutacji w danym genie. Wzrost proporcji hazardu można otrzymać dla danego genu poprzez obłożenie współczynnika funkcją wykładniczą o wykładniku e.

Wyniki estymacji dla genów zawierających największe co do modułu współczynniki można znaleźć w Tabeli 1

Podsumowanie

W pracy przedstawiono zastosowanie algorytmu numerycznej optymalizacji metodą stochastycznego spadku gradientu do wyznaczenia współczynników w modelu proporcjonalnych Coxa opartych na analitycznej metodzie wyznaczania estymatorów metodą największej wiarogodności. Zaprezentowano zastosowanie tego algorytmu do napływających danych wyliczając kolejne kroki optymalizacji w algorytmie w oparciu o częściową funkcję log-wiarogodności konstruowaną dla obecnie zaobserwowanego podzbioru obserwacji.

W pracy opisano matematyczne własności estymatorów największej wiarogodności, przedstawiono model Coxa wraz z najważniejszymi jego własnościami, scharakteryzowano algorytm stochastycznego spadku gradientu oraz jego różnice w stosunku go algorytmów spadku gradientu, zaimplementowano algorytm numerycznej optymalizacji metodą stochastycznego spadku gradientu do wyznaczenia współczynników w modelu proporcjonalnych Coxa opartych na analitycznej metodzie wyznaczania estymatorów metodą największej wiarogodności oraz przeprowadzono analizę rzeczywistych danych genomicznych z wykorzystaniem tego algorytmu. Pracę wzbogacono o symulacje zbieżności procesu optymalizacji funkcji log-wiarogodności dla modelu regresji logistycznej przygotowane dla algorytmów spadku gradientu rzędu I, spadku gradientu rzędu II oraz stochastycznego spadku gradientu rzędu I, dzięki którym możliwe było zobrazowanie różnic w tych metodach. Dodatkowo przeprowadzono symulacje dla sztucznie wygenerowanych danych do analizy przeżycia z rozkładu Weibulla obrazujące proces zbieżności dla nowo wprowadzonego algorytmu numerycznej optymalizacji metodą stochastycznego spadku gradientu do wyznaczenia współczynników w modelu proporcjonalnych Coxa opartych na analitycznej metodzie wyznaczania estymatorów metodą największej wiarogodności.

Do analizy genomicznej wykorzystano środowisko statystyczne $\mathcal{R}$, a wykorzystane funkcje, zaimplementowane algorytmy i przygotowaną dokumentację do stworzonego pakietu `coxphSGD` można znaleźć w Dodatku A.

Dodatek A

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

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)
  # + check names and types for every variables
  assert_that(is.function(learningRates))
  assert_that(is.numeric(epsilon))
  assert_that(is.numeric(beta_0))

    # check length of the start parameter
  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)
}

full_cox_loglik <- function(beta1, beta2, x1, x2, censored){
  sum(rev(censored)*(beta1*rev(x1) + beta2*rev(x2) -
                       log(cumsum(exp(beta1*rev(x1) + beta2*rev(x2))))))
}


calculate_outer_cox <- function(x1, x2, censored){
  ## contours
  outer_res <- outer(seq(-1,3, length = 100),
           seq(0,4, length = 100),
           Vectorize( function(beta1,beta2){
             full_cox_loglik(beta1, beta2, x1 = x1, x2 = x2, censored = censored)
           } )
  )
  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)
}
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


sample(1:10, 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*10) -> estimates6

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$coefficients)) %>%
  as.data.frame() -> df5
t(simplify2array(estimates6$coefficients)) %>%
  as.data.frame() -> df6

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"))) %>%
    bind_rows(df6 %>%
                mutate(version = paste("10 batches,", nrow(df6), " 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) +
  xlab("X1") + ylab("X2") +
  guides(col = guide_legend(ncol = 3))
dev.off()
```

A.2. 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*(beta1+beta2*xX)-log(1+exp(beta1+beta2*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)

  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$algorithm
  beta[2] -> XX
  beta[1] -> YY
  ggplot()+
    geom_path(aes(x = data2viz$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')  -> pl_g

  return(pl_g)
}
```

A.3. 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 ~ operator, and the terms on the right. The response must be a survival object as returned by the Surv function.
data	a list of batch data.frames in which to interpret the variables named in the formula. See Details.
learningRates	a function specifing 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 formula in the model.matrix 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)

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

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` generaters random survivaldata 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 censroing time. If censoring time is less then 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 Propotional 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

`dCox`	Input data.frame containing survival times (columnd should be named `time`) and status (columnd should be named `status`). So far this function only supports 2 explanatory variable (that should be named `x1` and `x2`.
`learningRates`	Parameter passed to coxphSGD.
`epsilon`	Parameter passed to coxphSGD.
`beta_0`	Parameter passed to coxphSGD.
`max.iter`	Parameter passed to coxphSGD 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.4. Analiza wpływu mutacji genów na czas życia

W pracy wykorzystano środowisko statystyczne $\mathcal{R}$ ([67], [6], [31]) do implementacji algorytmów, przeprowadzenia symulacji oraz wykonania analizy danych genomicznych. Do przetwarazania danych wykorzystano pakiety `dplyr` ([80]), `tidyr` ([84]), `reshape2` ([82]) oraz `data.table` ([24]). Do tworzenia dokumentacji oraz struktury pakietu `coxphSGD` ([47]) użyto pakietów `roxygen2` ([83]),`rmarkdown` ([2]), `knitr` ([89], [90], [91]), `assertthat` ([79]) oraz pakietu `rlp` ([13]). Przejrzystość kodu wspierał pakiety `magrittr` ([4]), zaś do wizualizacji wykorzystano pakiet `ggplot2` ([81]). Dane do analizy pobrane dzięki pakietowi `RTCGA` ([48]) oraz umieszczono je w pakietach `RTCGA.clinical` ([49]) oraz `RTCGA.mutations` ([50]). Funkcje wykorzystane w trakcie analizy danych znajdują sie poniżej.

```
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

}


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))
#
#   mutationsData_top %>%
#     spread(Hugo_Symbol, bcr_patient_barcode) -> mutationsData_top_sp

  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
}


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)
}
```

```
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
}

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

Literatura

[1] Aldrich J., (1997), *R. A. Fisher and the Making of Maximum Likelihood 1912 – 1922*, Statistical Science 1997, Vol. 12, No. 3, 162-176.

[2] Allaire J.J., Cheng J., Yihui Xie, McPherson J., Chang W., Allen J., Wickham H., Atkins A., Hyndman R., (2015), *rmarkdown: Dynamic Documents for R. R package version 0.8.1.*
`http://CRAN.R-project.org/package=rmarkdown`

[3] Asselain B., Mould R. F., (2010), *Methodology of the Cox proportional hazards model*, Journal of Oncology 2010, volume 60, Number 5, 403–409.

[4] Bache S. M., Wickham H., (2014), *magrittr: A Forward-Pipe Operator for R*, R package version 1.5.
`http://CRAN.R-project.org/package=magrittr`

[5] Bender R., Augustin T., Blettner Maria, (2005) *Generating survival times to simulate Cox proportional hazards models*, Statistics in Medicine, Volume 24, Issue 11, 1713–1723.

[6] Biecek P., (2011), *Przewodnik po pakiecie R*, Rozprawa doktorska, Oficyna Wydawnicza GiS, wydanie II.

[7] Bottou L., (2010), *Large-Scale Machine Learning with Stochastic Gradient Descent.*

[8] Bottou L., (1998), *Online Learning and Stochastic Approximations.*

[9] Bottou L., (2012), *Stochastic Gradient Descent Tricks.*

[10] Box-Steffensmeier J. M., Jones B. S, (2004), *Event History Modeling: A Guide for Social Scientists*, Cambridge University Press.

[11] Broad Institute TCGA Genome Data Analysis Center (2014): Firehose stddata 2015 06 01 run. Broad Institute of MIT and Harvard. DOI:10.7908/C1251HBG

[12] Burzykowski T., (2015?), *Notatki do przedmiotu Biostatystyka*,
`https://e.mini.pw.edu.pl/sites/default/files/biostatystyka.pdf`.

[13] Cheng J., Yihui Xie, (2014), *rlp: An Example of Using Literate Programming for R Package Development*, R package version 0.0.2.
`https://github.com/yihui/rlp`

[14] Chin L., Hahn W.C., Getz G., Meyerson M., (2011), *Making sense of cancer genomic data. Genes and Development*, 25(6): 534-555.

[15] Chin L., Andersen J.N., Futreal P.A., (2011), *Cancer genomics: from discovery science to personalized medicine*, Nature Medicine, 17(3): 297-303.

[16] Cox D. R. (1962), *Renewal Theory. Methuen Monograph on Applied Probability & Statistics*, London: Methuen.

[17] Cox D. R., (1972), *Regression models and life-tables (with discussion)*, Journal of the Royal Statistical Society Series B 34:187-220.

[18] Collett D., (1994), *Modelling Survival Data in Medical Research*, Chapman and Hall: London.

[19] Czepiel S. A., *Maximum Likelihood Estimation of Logistic Regression Models: Theory and Implementation*, `http://czep.net/stat/mlelr.pdf`.

[20] Dempster A., Laird N., Rubin D., (1977), *Maximum Likelihood from Incomplete Data via the EM Algorithm*, Journal of the Royal Statistical Society, Series B, 39, 1-38.

[21] Dennis J. E. Jr., Schnabel R. B., (1983), *Numerical Methods For Un- constrained Optimization and Nonlinear Equations*, Prentice-Hall.

[22] Dobson A. J., (2002), *An Introduction to Generalized Linear Models*, Wydanie II, Chapman & Hall/CRC.

[23] Domagała W., (2007), *Molekularne podstawy karcynogenezy i ścieżki sygnałowe niektórych nowotworów ośrodkowego układu nerwowego*, Polski Przegląd Neurologiczny, tom 3: 127-141.

[24] Dowle M., Srinivasan A.,Short T.,Lianoglou S. and contributions from R Saporta and E Antonyan, (2015), *data.table: Extension of Data.frame*, R package version 1.9.6. `http://CRAN.R-project.org/package=data.table`

[25] Finkel J. R., Kleeman A., Manning C. D., (2008), *Efficient, Feature-based, Conditional Random Field Parsing*, Proc. Annual Meeting of the ACL.

[26] Fisher R. A., (1912) *An absolute criterion for fitting frequency curves.*

[27] Fisher R. A., (1922) *On the mathematical foundations of theoretical statistics*, Philos. Trans. Roy. Soc. London Ser. A 222 309-368.

[28] Fisher R., A., (1925), *Statistical Methods for Research Workers*, Oliver and Boyd, Edinburgh.

[29] Fortuna Z., Macukow B., Wąsowski J., (2006), *Metody Numeryczne*, Wydawnictwa Naukowo-Techniczne.

[30] Gauss C. F., (1809,) *Theoria Motus Corporum Coelestium.*

[31] Gągolewski M., (2014), *Programowanie w języku R*, Wydawnictwo Naukowe PWN.

[32] Goeman J. J., (2010), *L1 penalized estimation in the Cox proportional hazards model*, Biom J. Feb; 52(1):70-84.

[33] Green P. J., (1984), *Iteratively Reweighted Least Squares for Maximum Likelihood Estimation, and Some Robust and Resistant Alternatives*, Journal of the Royal Statistical Society, Series B (Methodological), pp. 149-192.

[34] Hald A., (1949), *Maximum likelihood estimation of the parameters of a normal distribution which is truncated at a known point*, Skandinavisk Aktuarietidskrift, 119-134.

[35] Hastie T. J., Pregibon D., (1992), *Generalized linear models. Chapter 6 of Statistical Models in S*, eds J. M. Chambers and T. J. Hastie, Wadsworth & Brooks/Cole.

[36] Heckman J. J., Singer B, (1985), *Longitudinal Analysis of Labor Market Data*, Cambridge University Press.

[37] Hosmer D. W., Lemeshow S., May S, (2008), *Applied Survival Analysis: Regression Modeling of Time to Event Data*, Wiley Series in Probability and Statistics 2nd edition, Wiley-Interscience.

[38] Hutchinson J. B., (1928), *The Application of the "Method of Maximum Likelihood" to the Estimation of Linkage*, Genetics. 1929 Nov; 14(6): 519–537.

[39] Janik-Papis K., Błasiak J., (2010), *Molekularne wyznaczniki raka piersi. Inicjacja i promocja - część I*, Nowotwory - Journal of Oncology, 60 (3): 236-247.

[40] Janik-Papis K., Błasiak J., (2010), *Molekularne wyznaczniki raka piersi. Progresja i nowi kandydaci - część II*, Nowotwory - Journal of Oncology, 60 (4): 341-350.

[41] Jennrich R. I., Sampson P. F., (1976), *Newton-Raphson and related algorithms for maximum likelihood variance component estimation*, Technometrics, 18, 11-17.

[42] Kalbfleisch J. D., Prentice R. L, (1980), *The Statistical Analysis of Failure Time Data*, New York: John Wiley and Sons.

[43] 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 Vol. 50, No. 4 (Dec., 1994), pp. 945-953.

[44] Kim J., Kim Y., (2004), *Gradient lasso for feature selection. In Proceedings of the 21th International Conference on Machine Learning*, Morgan Kaufmann, ACM, New York, pp. 473-480.

[45] Kim J. Kim Y., Kim Y., (2008), *A gradient-based optimization algorithm for lasso*, Journal of Computational and Graphical Statistics 17, 994-1009.

[46] Klein J. P., Moeschberger M. L, (2003), *Survival Analysis: Techniques For Censored and Truncated Data, 2nd edition*, New York: John Willey and Sons.

[47] Kosiński M., (2015), *coxphSGD: Stochastic Gradient Descent log-likelihood estimation in Cox proportional hazards model*, R package version 0.0.5, `https://github.com/MarcinKosinski/coxphSGD`.

[48] Kosiński M., Biecek P., (2015), *RTCGA: The Cancer Genome Atlas Data Integration*, R package version 1.1.10, `https://github.com/RTCGA`.

[49] Kosiński M., (2015), *RTCGA.clinical: Clinical datasets from The Cancer Genome Atlas Project*, R package version 20151101.0.0, `https://bioconductor.org/packages/release/data/experiment/html/RTCGA.clinical.html`

[50] Kosiński M., (2015), *RTCGA.mutations: Clinical datasets from The Cancer Genome Atlas Project*, R package version 20151101.0.0, `https://bioconductor.org/packages/release/data/experiment/html/RTCGA.mutations.html`

[51] Kotłowski W., (2012), Notatki do przedmiotu *Techniki Optymalizacji* prowadzonego na Politechnice Poznańskiej, `http://www.cs.put.poznan.pl/wkotlowski/teaching/wyklad3b.pdf`

[52] Kozłowska J., Łaczmańska I., (2010), *Niestabilność genetyczna - jej znaczenie w procesie powstawania nowotworów oraz diagnostyka laboratoryjna*, Nowotwory - Journal of Oncology, 60 (6): 548-553.

[53] Legendre A. M., (1804), *Nouvelles mèthods pour la dètermination des orbites des comètes.*

[54] 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.

[55] Milewski S., (2006), Konspekt do przedmiotu *Metody Numeryczne* prowadzonego na Politechnice Krakowskiej,
http://l5.pk.edu.pl/images/skrypty/Metody_numeryczne_1

[56] 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.

[57] Mood A. M., Graybill F. A., Boes D. C., (1974), *Introduction to the Theory of Statistics*, McGraw-Hill: New York.

[58] Mittal S., Madigan D., (2014), *High-dimensional, massive sample-size Cox proportional hazards regression for survival analysis*, Biostatistics, 2014 Apr; 15(2): 207-221.

[59] Murata N., (1998), *A Statistical Study of On-line Learning. In Online Learning and Neural Networks*, Cambridge University Press.

[60] Niemiro W., (2011), Skrypt do przedmiotu *Statystyka* prowadzonego na Uniwersytecie Warszawskim,
http://www-users.mat.umk.pl/∼wniem/Statystyka/Statystyka.pdf

[61] Norwegian Multicentre Study Group, (1981), *Timolol-induced reduction in mortality and reinfarction*, The New England Journal of Medicine; 304: 801-7.

[62] Oakes D., (2001), *Biometrika centenary: survival analysis*, Biometrika, 88(1):99-142.

[63] Park M. Y., Hastie T., (2007), *L1-regularization path algorithm for generalized linear models*, Journal of the Royal Statistical Society, 69(4):659-677.

[64] Panchenko D., (2006), Notatki do otwartego kursu MIT *Statistics for Applications, Lecture 2: Maximum Likelihood Estimators.*,
http://ocw.mit.edu/courses/mathematics/18-443-statistics-for-applications-fall-2006/

[65] Panchenko D., (2006), Notatki do otwartego kursu MIT *Statistics for Applications, Lecture 3: Properties of MLE: consistency, asymptotic normality. Fisher information.*,
http://ocw.mit.edu/courses/mathematics/18-443-statistics-for-applications-fall-2006/

[66] Podsiadły K., (2011), *Genetyczne podstawy nowotworzenia*,
www.e-biotechnologia.pl/Artykuly/Genetyczne-podstawy-nowotworzenia.

[67] R Core Team, (2013) *R: A language and environment for statistical computing.* R Foundation for Statistical Computing, Wiedeń , ISBN 3-900051-07-0,
http://www.R-project.org/.

[68] Robbins H. E., Siegmund. D. O., (1971), *A convergence theorem for non negative almost supermartingales and some applications*, In Proc. Sympos. Optimizing Methods in Statistics, pages 233–257, Ohio State University. Academic Press, New York.

[69] Rydlewski J., (2009), *Estymatory Największej Wiarogodności w Uogólnionych Modelach Regresji Nieliniowej*, Rozprawa doktorska.

[70] Sohn I., Kim J., Jung S. H., Park C., (2009) *Gradient lasso for Cox proportional hazards model*, Bioinformatics, Jul 15; 25(14):1775-81.

[71] Sokołowski A., (2010), *Jak rozumieć i wykonywać analizę przeżycia*
http://www.statsoft.pl/Portals/0/Downloads/Jak_rozumiec_i_wykonac_analize_przezycia.pdf

[72] Statistics Views, (2014), *"I would like to think of myself as a scientist, who happens largely to specialise in the use of statistics"– An interview with Sir David Cox.*

[73] Tran D., Lan T., Toulis P., (2015), *sgd: Stochastic Gradient Descent for Scalable Estimation. R package version 0.1*, `https://github.com/airoldilab/sgd`.

[74] *The future of cancer genomics*, Nature Medicine, (2015), 21(2): 99.

[75] Therneau T. M., Grambsch P. M., (2000), *Modeling Survival Data: Extending the Cox Model*, Springer.

[76] Therneau T. M., (2015), *A Package for Survival Analysis in S. version 2.38*, `http://CRAN.R-project.org/package=survival`.

[77] Tomczak K., Czerwińska P., Wiznerowicz M., (2015), *The Cancer Genome Atlas (TCGA): an immeasurable source of knowledge*, Contemporary Oncology. 19(1A): A68-A77.

[78] Toulis P.,Airoldi E., M., (2015), *Implicit stochastic gradient descent*, arXiv:1408.2923v5 [stat.ME] 4 Oct 2015.

[79] Wickham H., (2013), *assertthat: Easy pre and post assertions*, R package version 0.1. `http://CRAN.R-project.org/package=assertthat`

[80] Wickham H., Francois R., (2015), *dplyr: A Grammar of Data Manipulation*, R package version 0.4.3.
`http://CRAN.R-project.org/package=dplyr`

[81] Wickham H., *ggplot2: elegant graphics for data analysis*, Springer New York, 2009.

[82] Wickham H., (2007), *Reshaping Data with the reshape Package*, Journal of Statistical Software, 21(12), 1-20.
`http://www.jstatsoft.org/v21/i12/`

[83] Wickham H., Danenberg P.,Eugster M., (2015), *roxygen2: In-Source Documentation for R*, R package version 5.0.1.
`http://CRAN.R-project.org/package=roxygen2`

[84] 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

[85] Widrow B., (1960), *An adaptive "ADALINE" neuron using chemical "memistors"*, Technical Report No. 1553-2, Stanford University.

[86] Widrow B., Ho M.E., (1960), *Adaptive switching circuits*, In: IRE WESCON Conv. Record, Part 4. pp. 96-104.

[87] Widrow B., Stearns S. D., (1985), *Adaptive Signal Processing, Prentice Hall.*

[88] Woodcock S., (2014), Notatki do otwartego kursu Uniwersytetu Simona Frasera *ECON 837, Lecture 11 Asymptotic Properties of Maximum Likelihood Estimators*,
`http://www.sfu.ca/ swoodcoc/teaching/sp2014/econ837/11.mle.pdf`

[89] Yihui Xie, (2015), *knitr: A General-Purpose Package for Dynamic Report Generation in R*, R package version 1.11.

[90] Yihui Xie, (2015), *Dynamic Documents with R and knitr*, 2nd edition. Chapman and Hall/CRC. ISBN 978-1498716963.

[91] Yihui Xie, (2014), *knitr: A Comprehensive Tool for Reproducible Research in R. In Victoria Stodden, Friedrich Leisch and Roger D. Peng, editors, Implementing Reproducible Computational Research*, Chapman and Hall/CRC. ISBN 978-1466561595.

[92] Zieliński R., (1990), *Siedem wykładów wprowadzających do statystyki matematycznej*, Warszawa, Wydawnictwo Naukowe PWN.

Marcin Piotr Kosiński
Nr albumu 265361

Warszawa, 9 grudnia 2015

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