Uniwersytet Warszawski
Wydział Matematyki, Informatyki i Mechaniki

Joanna Skrzeszewska
Nr albumu: 277648

Algorytmy optymalizacji, estymacji i redukcji wariancji

Praca licencjacka
na kierunku MATEMATYKA

Praca wykonana pod kierunkiem
dra inż. Przemysława Biecka
Instytut Matematyki Stosowanej i Mechaniki

Sierpień 2012

Oświadczenie kierującego pracą

Potwierdzam, że niniejsza praca została przygotowana pod moim kierunkiem i kwalifikuje się do przedstawienia jej w postępowaniu o nadanie tytułu zawodowego.

Data Podpis kierującego pracą

Oświadczenie autora (autorów) pracy

Świadom odpowiedzialności prawnej oświadczam, że niniejsza praca dyplomowa została napisana przeze mnie samodzielnie i nie zawiera treści uzyskanych w sposób niezgodny z obowiązującymi przepisami.

Oświadczam również, że przedstawiona praca nie była wcześniej przedmiotem procedur związanych z uzyskaniem tytułu zawodowego w wyższej uczelni.

Oświadczam ponadto, że niniejsza wersja pracy jest identyczna z załączoną wersją elektroniczną.

Data Podpis autora (autorów) pracy

Streszczenie

W pracy omówiono algorytmy optymalizacji, estymacji i redukcji wariancji. Praca została podzielona na trzy części. W pierwszej części przedstawiono wybrane metody optymalizacji dla funkcji jedno- i wielowymiarowych, w drugiej opisano podstawowe algorytmy estymacji punktowej i przedziałowej, a w ostatniej wybrane algorytmy redukcji wariancji.

Słowa kluczowe

optymalizacja, estymacja, estymator, przedział ufności, redukcja warancji

Dziedzina pracy (kody wg programu Socrates-Erasmus)

11.2 Statystyka

Klasyfikacja tematyczna

62H12, 65C20

Tytuł pracy w języku angielskim

Algorithms for optimisation, estimation and variance reduction

Spis treści

Wprowadzenie

Bardzo ważnym, praktycznym zadaniem w modelowaniu statystycznym jest estymacja nieznanych parametrów rozkładu tak, aby reprezentował on dobrze dane empiryczne. Estymacji można dokonywać zarówno standardowymi metodami – takimi jak metoda momentów, metoda kwantyli czy metoda największej wiarogodności, jak i zmodyfikowanymi – np. wykorzystując problem nadokreślonych układów równań liniowych do metody kwantyli czy metody momentów. We wszystkich wymienionych metodach bardzo pomocne są algorytmy optymalizacji, zwłaszcza wielowymiarowej, które służą do znajdowania lokalnych ekstremów funkcji.

Podczas wielokrotnych symulacji wartości estymowanych parametrów różnią się – jest to skutkiem losowości. Pożądane jest, aby zmienność wyników, którą można wyrazić poprzez wariancję, była jak najmniejsza. W tym celu można zwiększyć liczebność próby lub posłużyć się metodami redukcji wariancji, które są mniej czasochłonne.

W poniższej pracy przedstawię algorytmy optymalizacji, estymacji i redukcji wariancji. Praca składać się będzie z trzech części. W pierwszej części omówię wybrane metody optymalizacji dla funkcji jednowymiarowej i funkcji wielu zmiennych oraz pokażę ich przykładowe zastosowania. W drugiej części przedstawię algorytmy estymacji punktowej i przedziałowej. Pokażę także wykorzystanie algorytmów optymalizacji w metodach estymacji. W obu rozdziałach użyję danych `apartments` pochodzących z pakietu `PBImisc`, które zawierają informacje na temat cen mieszkań w Warszawie w latach 2007 – 2009. W rozdziale trzecim omówię algorytmy redukcji wariancji: *antithetic sampling*, *importance sampling* i *control variance* i na podstawie przykładu prześledzę jak zmniejszyła się wariancja po wykorzystaniu wymienionych algorytmów.

Rozdział 1

Algorytmy optymalizacji

W poniższym rozdziale opiszę zagadnienie optymalizacji, rozumiane jako znajdowanie lokalnych ekstremów funkcji. Problem ten przedstawię w jednym i w wielu wymiarach na wybranych kilku, spośród bardzo wielu istniejących, algorytmach. Rozważę tylko maksima lokalne, gdyż minima lokalne, w każdym z poniższych przypadków, można uzyskać mnożąc funkcję przez -1. Do omówienia zagadnienia optymalizacji funkcji jednej zmiennej wykorzystam *metodę Newtona* i *metodę złotego podziału*, natomiast w przypadku funkcji wielu zmiennych użyję *metody najszybszego wzrostu* oraz *wielowymiarowej metody Newtona*. Pokażę też przykładowe zastosowania metod optymalizacji. Źródłem informacji były [1], [2], [3] i [7].

1.1. Optymalizacja funkcji jednej zmiennej

Definicja 1.1.1 *Mówimy, że funkcja $f : X \to \mathbb{R}$ osiąga* maksimum lokalne *w punkcie $x_0 \in X$, jeśli istnieje pewne otoczenie punktu x_0, w którym wartości funkcji f są niewiększe od wartości funkcji f w punkcie x_0, to znaczy:*

$$\exists \delta > 0 : \forall x \in X : d(x, x_0) < \delta \Longrightarrow f(x) \leqslant f(x_0).$$

Jeśli ponadto $f(x_0) = supf(X)$ - to znaczy: jeśli w punkcie x_0 funkcja f osiąga kres górny wartości w zbiorze X, to mówimy, że funkcja f osiąga w punkcie x_0 maksimum globalne.

Wszystkie metody, które będą omówione w tej pracy są algorytmami wykorzystującymi technikę lokalnego wyszukiwania, polegającą na generowaniu ciągu punktów $x(0), x(1), x(2), ...$, które mają zbiegać do lokalnego ekstremum. Nie będzie badana cała przestrzeń możliwych rozwiązań, a jedynie w sąsiedztwie punktu $x(n)$ wyszukiwany będzie następny punkt $x(n+1)$.

Niech x^* będzie lokalnym ekstremum funkcji f oraz niech $x(n) \to x^*$ (dla $n \to \infty$). Pożądanym *kryterium stopu*, kończącym wyszukiwanie jest warunek $|x(n) - x^*| \leqslant \epsilon$, dla zadanej wartości ϵ. Niestety w ogólności uzyskanie takiego kryterium stopu nie jest możliwe i zamiast niego używane są następujące kryteria:

- $|x(n) - x(n-1)| \leqslant \epsilon$;
- $|f(x(n)) - f(x(n-1))| \leqslant \epsilon$;
- $|f'(x(n))| \leqslant \epsilon$.

Jeżeli ciąg $x(n)_{n=1}^{\infty}$ zbiega do lokalnego maksimum, to powyższe trzy kryteria są spełnione, jednak implikacja odwrotna nie jest prawdziwa. Poniższe metody mogą wcale nie być zbieżne na przykład, gdy f jest funkcją nieograniczoną i $x(n) \to \infty$. Z tego powodu zwykle przyjmuje się maksymalną liczbę iteracji n_{max} i zatrzymuje wyszukiwanie gdy $n \leqslant n_{max}$.

1.1.1. Metoda Newtona

Metoda Newtona jest algorytmem mającym zastosowanie w wielu sytuacjach. Jej wariant używany do szukania miejsc zerowych nazywa się *metodą Newtona-Raphsona*.

Niech f będzie funkcją, której zera będą szukane numerycznie, x^* będzie zerem funkcji f, a x jego przybliżeniem. Jeżeli f'' istnieje, to na mocy twierdzenia Taylora:

$$0 = f(x^*) = f(x+h) = f(x) + hf'(x) + \mathcal{O}(h^2), \tag{1.1}$$

gdzie $h = x^* - x$. Dla małych h (czyli dla x bliskiego x^*) składnik $\mathcal{O}(h^2)$ można pominąć i wtedy:

$$h = -\frac{f(x)}{f'(x)}, \tag{1.2}$$

a lepsze niż x przybliżenie x^* jest równe:

$$x + h = x - \frac{f(x)}{f'(x)}. \tag{1.3}$$

Działanie algorytmu optymalizacji Newtona rozpoczyna się przybliżeniem zera x^* za pomocą $x(0)$, a następnie stosuje się rekurencyjny wzór:

$$x(n+1) = x(n) - \frac{f(x(n))}{f'(x(n))}, \qquad n \geqslant 0. \tag{1.4}$$

Interpretacja geometryczna metody Newtona

Metoda Newtona opiera się na linearyzacji funkcji f. Przybliżając, w dowolnym punkcie x, funkcję $f(x)$ funkcją liniową $l(x)$, powstałą z rozwinięcia funkcji $f(x)$ w szereg Taylora otrzymujemy:

$$l(x) = f(x) = f(c) + f'(c)(x - c). \tag{1.5}$$

Powyższa funkcja liniowa przybliża dobrze f w pobliżu c. Zachodzi $l(c) = f(c)$ oraz $l'(c) = f'(c)$. Funkcja liniowa $l(x)$ ma więc w punkcie c tą samą wartość i to samo nachylenie co funkcja f. Z rozważań tych wynika, że w metodzie Newtona konstruowana jest styczna do wykresu w punkcie bliskim zeru funkcji f i znajdowany jest punkt, w którym ta styczna przecina oś x. Łatwo więc wyobrazić sobie funkcję oraz punkt startowy, dla której metoda Newtona zawodzi. Aby metoda Newtona była zbieżna,[1] badana funkcja nie może mieć dowolnego kształtu oraz punkt startowy musi znajdować się dostatecznie blisko zera funkcji. Interpretację geometryczną metody Newtona oraz przykład funkcji, dla której metoda Newtona nie jest zbieżna przedstawiłam odpowiednio na rysunkach 1.1 i 1.2.

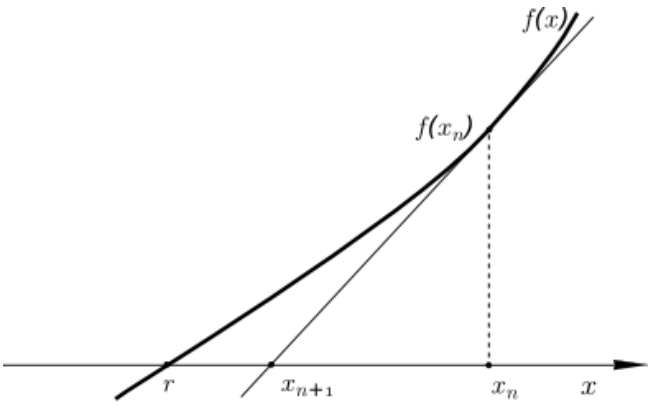
Szukanie lokalnych ekstremów funkcji przy pomocy metody Newtona

Jeżeli funkcja $f : [a, b] \to \mathbb{R}$ ma ciągłą pierwszą pochodną f', to problem znalezienia maksimum funkcji f jest równoważny znalezieniu maksimum $f(a), f(b), f(x_1), ..., f(x_n)$, gdzie $x_1, ..., x_n$ są zerami funkcji f'. Po zastosowaniu metody Newtona-Raphsona do znajdowania miejsc zerowych funkcji f' uzyskujemy metodę Newtona do optymalizacji funkcji f:

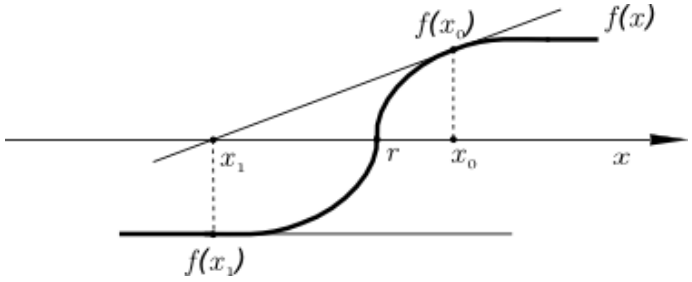
$$x(n+1) = x(n) - \frac{f'(x(n))}{f''(x(n))}. \tag{1.6}$$

Implementację metody Newtona przedstawiłam w dodatku A. Więcej informacji na temat metody Newtona można znaleźć w [2] i [3].

[1] Metoda Newtona jest zbieżna kwadratowo, źródło [3].



Rysunek 1.1: Interpretacja geometryczna metody Newtona. Rysunek opracowany został na podstawie [3], przy użyciu programu Inkscape.



Rysunek 1.2: Przykład funkcji, dla której metoda Newtona jest rozbieżna. Rysunek opracowany został na podstawie [3], przy użyciu programu Inkscape.

1.1.2. Metoda złotego podziału

Metodę złotego podziału, w przeciwieństwie do metody Newtona, można stosować tylko w jednym wymiarze, ale nie jest za to konieczne istnienie pochodnej funkcji. Metoda ta jest podobna do metody bisekcji.

Niech $f : \mathbb{R} \to \mathbb{R}$ będzie funkcją ciągłą. Jeżeli istnieją dwa punkty $a < b$ takie że $f(a)f(b) \leqslant 0$ wiadomo, że w przedziale $[a, b]$ znajduje się zero funkcji f. Do ustalenia czy w tym przedziale istnieje lokalne maksimum potrzebne są trzy punkty: a, b, c. Jeżeli $a < c < b$ oraz $f(a) \leqslant f(c)$ i $f(b) \leqslant f(c)$ w przedziale $[a, b]$ musi istnieć lokalne maksimum. Powyższa obserwacja prowadzi do następującego algorytmu:

Rozpocznij od $x_l < x_m < x_r$, takich, że $f(x_l) \leqslant f(x_m)$ oraz $f(x_r) \leqslant f(x_m)$
1. Jeżeli $x_r - x_l \leqslant \epsilon$ STOP.
2. Jeżeli $x_r - x_m > x_m - x_l$ idź do 2a.
W przeciwnym przypadku idź do 2b.
    2a. Wybierz punkt y z przedziału (x_m, x_r).
        Jeżeli $f(y) \geqslant f(x_m)$ to: $x_l = x_m$ oraz $x_m = y$.
        W przeciwnym przypadku: $x_r = y$.
    2b. Wybierz punkt y z przedziału (x_l, x_m).
        Jeżeli $f(y) \geqslant f(x_m)$ to: $x_r = x_m$ oraz $x_m = y$.
        W przeciwnym przypadku: $x_l = y$.
3. Wróć do kroku 1.

Stosując ten algorytm można dowolnie zawężać przedział, w którym znajduje się szukane maksimum lokalne. Jak dotąd nie zostało powiedziane jeszcze w jaki sposób dokonywać wyboru punktu y. Wiadomo, że ma on należeć do większego spośród dwóch przedziałów: (x_l, x_m) oraz (x_m, x_r). Naśladując metodę bisekcji, algorytm złotego podziału zakłada taki wybór punktu y, aby stosunek długości dłuższego do krótszego przedziału pozostawał w każdym kroku taki sam.

Niech $a = x_m - x_l$, $b = x_r - x_m$ oraz $c = y - x_m$. Niech (x_m, x_r) będzie dłuższym przedziałem, w którym wybierany będzie punkt y.

Jeżeli $f(y) \leqslant f(x_m)$ to nowym przedziałem jest (x_l, y). Musi wtedy zachodzić:

$$\frac{a}{c} = \frac{b}{a}. \tag{1.7}$$

Jeżeli $f(y) \geqslant f(x_m)$ to nowym przedziałem jest (x_m, x_r). Musi wtedy zachodzić:

$$\frac{b-c}{c} = \frac{b}{a}. \tag{1.8}$$

Ponieważ w metodzie złotego podziału stosunek dłuższego do krótszego przedziału za każdym razem ma być taki sam, niech $\rho = \frac{b}{a}$. Zachodzi wtedy:

$$\rho^2 - \rho - 1 = 0 \tag{1.9}$$

oraz:

$$\rho = \frac{1+\sqrt{5}}{2}. \tag{1.10}$$

Z odpowiedniego przekształcenia powyższych równości, zachodzi:

$$y = x_m + c = x_m + (x_r - x_m)/(1+\rho), \tag{1.11}$$

co jednoznacznie wskazuje jak dokonywać wyboru punktu y. Analogiczne rozumowanie można przedstawić, gdy (x_l, x_m) jest dłuższym przedziałem. Zachodzi wtedy:

$$y = x_m + c = x_m - (x_r - x_m)/(1+\rho). \tag{1.12}$$

Metoda złotego podziału jest optymalna w tym sensie, że jeżeli punkt y będzie wybierany w inny sposób, w najgorszym przypadku algorytm będzie wolniejszy od algorytmu złotego podziału[2]. W dodatku B przedstawiłam implementację metody złotego podziału. Więcej informacji na temat metody złotego podziału można znaleźć w [2].

1.2. Optymalizacja funkcji wielu zmiennych

Problem szukania ekstremum lokalnego funkcji wielu zmiennych jest bardziej przydatny od problemu optymalizacji funkcji jednej zmiennej, ale jest on zarazem o wiele trudniejszy.

Niech $f : \mathbb{R}^d \to \mathbb{R}$ i wszystkie pochodne cząstkowe pierwszego i drugiego rzędu funkcji f istnieją i są wszędzie ciągłe. Niech $\mathbf{x} = (x_1, ..., x_d)^T$ będzie punktem przestrzeni $\mathbb{R}^d$, natomiast $\mathbf{e_i}$ i-tym wektorem bazowym. Zachodzi wówczas $\mathbf{x} = x_1 e_1 + ... + x_d e_d$. Dla każdego i $f_i(\mathbf{x}) = \frac{\partial f(\mathbf{x})}{\partial x_i}$ oznacza pochodną cząstkową.

Definicja 1.2.1 *Przy powyższych oznaczeniach,* gradientem *funkcji f jest wektor:*

$$\nabla f(\mathbf{x}) = (f_1(\mathbf{x}), ..., f_d(\mathbf{x}))^T. \tag{1.13}$$

[2]Metoda złotego podziału jest zbieżna liniowo

Definicja 1.2.2 Macierzą Hessego *(hesjanem) funkcji f jest macierz:*

$$H(\mathbf{x}) = \begin{pmatrix} \frac{\partial^2 f(\mathbf{x})}{\partial x_1 \partial x_1} & \cdots & \frac{\partial^2 f(\mathbf{x})}{\partial x_1 \partial x_d} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 f(\mathbf{x})}{\partial x_d \partial x_1} & \cdots & \frac{\partial^2 f(\mathbf{x})}{\partial x_d \partial x_d} \end{pmatrix}. \tag{1.14}$$

Definicja 1.2.3 *Funkcja f posiada* lokalne maksimum *w punkcie* $\mathbf{x}^*$*, jeżeli istnieje liczba* $\delta > 0$ *taka, że dla każdego* $\mathbf{x}$ *zachodzi poniższa implikacja:*

$$\|\mathbf{x} - \mathbf{x}^*\| < \delta \Rightarrow f(\mathbf{x}) \leqslant f(\mathbf{x}^*),$$

gdzie $\|\mathbf{x}\| = \sqrt{x_1^2 + ... + x_n^2}$ *oznacza normę euklidesową.*

Definicja 1.2.4 *Dla każdego wektora* $\mathbf{v} \neq 0$ nachylenie *(ang. slope) w punkcie* $\mathbf{x}$ *w kierunku* $\mathbf{v}$ *zadane jest przez:*

$$\frac{v^T \nabla f(\mathbf{x})}{\|\mathbf{v}\|}, \tag{1.15}$$

Definicja 1.2.5 Krzywizna *w punkcie* $\mathbf{x}$ *w kierunku wektora* $\mathbf{v}$ *jest zadana przez wzór:*

$$\frac{\mathbf{v}^T H(\mathbf{x}) \mathbf{v}}{\|\mathbf{v}\|^2}. \tag{1.16}$$

Koniecznym, ale niewystarczającym warunkiem istnienia maksimum w punkcie $\mathbf{x}$ jest $\nabla f(\mathbf{x}) = \mathbf{0} = (0, ..., 0)^T$ oraz nachylenie w punkcie $\mathbf{x}$ w kierunku wektora $\mathbf{v}$ musi być nieujemne (odpowiednio macierz Hessego musi być półujemnie określona).

Wystarczającym, ale niekoniecznym warunkiem do istnienia lokalnego maksimum w punkcie $\mathbf{x}$ jest $\nabla f(\mathbf{x}) = \mathbf{0}$ oraz krzywizna w każdym kierunku musi być mniejsza od zera (odpowiednio macierz Hessego ujemnie określona).

Tak samo jak w przypadku jednego wymiaru, maksima lokalne będą wyszukiwane metodami iteracyjnymi. Warunki stopu są kombinacjami poniższych warunków:

- $\|\mathbf{x}(n) - \mathbf{x}(n-1)\|_\infty \leqslant \epsilon$;
- $\|f(\mathbf{x}(n)) - f(\mathbf{x}(n-1))\|_\infty \leqslant \epsilon$;
- $\|\nabla f(\mathbf{x}(n))\|_\infty \leqslant \epsilon$,

gdzie $\|\mathbf{x}\|_\infty = max_i |x_i|$ (norma L_∞). Analogicznie do przypadku jednowymiarowego zachodzi potrzeba przyjęcia maksymalnej liczby iteracji n_{max}.

1.2.1. Metoda najszybszego wzrostu

Niech $f : \mathbb{R}^d \to \mathbb{R}$ będzie funkcją z ciągłymi pochodnymi cząstkowymi w całej dziedzinie. W *metodzie najszybszego wzrostu* lokalne maksimum jest wyszukiwane w pobliżu punktu startowego $x(0)$, a $n+1$-wszy punkt obliczany jest według wzoru:

$$\mathbf{x}(n+1) = \mathbf{x}(n) + \alpha \mathbf{v}, \tag{1.17}$$

gdzie α jest dodatnim skalarem, a wektor $\mathbf{v}$ jest wektorem w kierunku największego nachylenia, tzn. $\mathbf{v}$ maksymalizuje $\frac{\mathbf{v}^T \nabla f(\mathbf{x})}{\|\mathbf{v}\|}$ (def. 1.2.4):

$$\frac{\partial}{\partial v_i} \frac{\mathbf{v}^T \nabla f(\mathbf{x})}{\|\mathbf{v}\|} = \frac{f_i(\mathbf{x})}{\|\mathbf{v}\|} - \frac{(\mathbf{v}^T \nabla f(\mathbf{x})) v_i}{\|\mathbf{v}\|^3}. \tag{1.18}$$

Po przyrównaniu prawej strony powyższej równości do zera zachodzi $v_i \propto f_i(\mathbf{x})$, z czego wynika, że kierunkiem o największym nachyleniu jest gradient. Po podstawieniu gradientu do równania (1.17) otrzymujemy:

$$\mathbf{x}(n+1) = \mathbf{x}(n) + \alpha \nabla f(\mathbf{x}(n)), \tag{1.19}$$

dla pewnego $\alpha \geqslant 0$ wybranego tak, żeby zmaksymalizować funkcję:

$$g(\alpha) = f(\mathbf{x}(n+1)) = f(\mathbf{x}(n) + \alpha \nabla f(\mathbf{x}(n))). \tag{1.20}$$

Jeżeli $\alpha = 0$ to $\mathbf{x}(n)$ jest lokalnym maksimum, natomiast jeśli $\alpha > 0$ to $f(\mathbf{x}(n+1)) > f(\mathbf{x}(n))$. Do maksymalizacji funkcji (1.20) często używana jest metoda złotego podziału. Implementację metody najszybszego wzrostu[3] przedstawiłam w dodatku C. Więcej informacji na temat tej metody można znaleźć w [2].

1.2.2. Wielowymiarowa metoda Newtona

W metodzie najszybszego wzrostu wykorzystywane są informacje na temat gradientu. Używając hesjanu można stworzyć metody, które zbiegają korzystając z mniejszej ilości kroków. Najprostszą z nich jest *wielowymiarowa metoda Newtona*, która powstaje poprzez uogólnienie jednowymiarowej metody Newtona[4].

Wielowymiarowa metoda Newtona polega na wyszukaniu punktu $\mathbf{x}$ takiego, że $\nabla f(\mathbf{x}) = \mathbf{0}$. Podstawą tej metody jest rozwinięcie Taylora funkcji f. Dla $\mathbf{y}$ w pobliżu $\mathbf{x}$ zachodzi:

$$f(\mathbf{y}) \approx f(\mathbf{x}) + (\mathbf{y}-\mathbf{x})^T \nabla f(\mathbf{x}) + \frac{1}{2}(\mathbf{y}-\mathbf{x})^T \mathbf{H}(\mathbf{x})(\mathbf{y}-\mathbf{x}). \tag{1.21}$$

Wielowymiarowe przybliżenie za pomocą rozwinięcia Taylora można wyprowadzić z rozwinięcia w jednym wymiarze. Podstawiając: $\mathbf{v} = \mathbf{y} - \mathbf{x}$ i definiując $g(\alpha) = f(\mathbf{x} + \alpha\mathbf{v})$ zachodzi:

$$\begin{aligned} g'(0) &= \lim_{\alpha \to 0} \frac{g(\alpha) - g(0)}{\alpha} \\ &= \lim_{\alpha \to 0} \Bigg(\frac{f(\mathbf{x} + \alpha v_1 \mathbf{e}_1 + ... + \alpha v_d \mathbf{e}_d) - f(\mathbf{x} + \alpha v_2 \mathbf{e}_2 + ... + \alpha v_d \mathbf{e}_d)}{\alpha} \\ &+ \frac{f(\mathbf{x} + \alpha v_2 \mathbf{e}_2 + ... + \alpha v_d \mathbf{e}_d) - f(\mathbf{x} + \alpha v_3 \mathbf{e}_3 + ... + \alpha v_d \mathbf{e}_d)}{\alpha} \\ &+ ... + \frac{f(\mathbf{x} + \alpha v_d \mathbf{e}_d) - f(\mathbf{x})}{\alpha} \Bigg) \\ &= v_1 f_1(\mathbf{x}) + v_2 f_2(\mathbf{x}) + ... + v_d f_d(\mathbf{x}) = \mathbf{v}^T \nabla f(\mathbf{x}). \end{aligned} \tag{1.22}$$

Korzystając z tej samej zasady:

$$g''(0) = \frac{1}{2}\mathbf{v}^T \mathbf{H}(\mathbf{x})\mathbf{v}. \tag{1.23}$$

Z rozwinięcia funkcji g w szereg Taylora:

$$g(\alpha) \approx g(0) + \alpha g(0) + \frac{1}{2}\alpha^2 g''(0) = f(\mathbf{x}) + \alpha \mathbf{v}^T \nabla f(\mathbf{x}) + \frac{1}{2}\alpha^2 \mathbf{v}^T \mathbf{H}(\mathbf{x})\mathbf{v}. \tag{1.24}$$

Po podstawieniu $\alpha = 1$ i obustronnym zróżniczkowaniu zachodzi:

$$\nabla f(\mathbf{y}) \approx \nabla f(\mathbf{x}) + \mathbf{H}(\mathbf{x})(\mathbf{y}-\mathbf{x}). \tag{1.25}$$

[3]Metoda najszybszego wzrostu jest zbieżna liniowo

[4]Wielowymiarowa metoda Newtona jest zbieżna kwadratowo.

Jeżeli $\mathbf{y}$ jest lokalnym maksimum, to $\nabla f(\mathbf{y}) = \mathbf{0}$ i zachodzi:

$$\mathbf{y} = \mathbf{x} - \mathbf{H}(\mathbf{x})^{-1}\nabla f(\mathbf{x}). \tag{1.26}$$

Powyższą własność wykorzystuje się w metodzie Newtona, podstawiając $\mathbf{x} = \mathbf{x}(n)$ oraz $\mathbf{y} = \mathbf{x}(n+1)$:

$$\mathbf{x}(n+1) = \mathbf{x}(n) - \mathbf{H}(\mathbf{x}(n))^{-1}\nabla f(\mathbf{x}(n)). \tag{1.27}$$

Jeżeli $\mathbf{H}(\mathbf{x}(n))$ jest osobliwa, metoda Newtona nie jest zbieżna. Analogicznie do przypadku jednowymiarowego, metoda Newtona może nie być zbieżna także w przypadku, gdy hesjan jest macierzą nieosobliwą. Implementację wielowymiarowej metody Newtona wraz z numerycznym obliczaniem hesjanu przedstawiłam w dodatku D. Więcej informacji na temat tej metody można znaleźć w [2].

1.3. Przykłady zastosowań metod optymalizacji

Zastosowanie wielowymiarowej metody Newtona do optymalizacji funkcji wiarogodności

Definicja 1.3.1 *Niech rozkład zmiennej losowej X zależy od parametrów $\theta = (\theta_1, ..., \theta_m)$ oraz niech $(x_1, ..., x_n)$ będzie zaobserwowaną wartością próby prostej $(X_1, ...X_n)$ z populacji mającej cechę X.* Funkcję wiarogodności *określa się wzorem:*

$$L(\theta) = L(x_1, ..., x_n, \theta) = p(x_1, \theta)...p(x_n, \theta),$$

gdzie $p(x_i, \theta) = \mathbb{P}(X = x_i)$ w przypadku dyskretnym oraz:

$$L(\theta) = L(x_1, ..., x_n, \theta) = f(x_1, \theta)...f(x_n, \theta),$$

dla cechy typu ciągłego o gęstości $f(x, \theta)$.

Funkcję wiarogodności wykorzystuje się do estymacji nieznanych parametrów metodą największej wiarogodności, o której będzie mowa w rozdziale 2. Aby dokonać estymacji nieznanego parametru tą metodą, wyszukuje się taką wartość estymowanego parametru, przy której wartość funkcji wiarogodności (lub równoważnie jej logarytmu) jest największa. Poniżej przedstawię doświadczenie, w którym dokonam optymalizacji funkcji wiarogodności, przy użyciu wielowymiarowej metody Newtona.

Rozważmy próbkę pochodzącą z rozkładu normalnego. Logarytm funkcji wiarogodności, dla próbki pochodzącej z tego rozkładu jest postaci:

$$-\frac{n}{2}\log(2\pi) - n\log\sigma - \frac{1}{2\sigma^2}\sum_{i=1}^{n}(x_i - \mu)^2. \tag{1.28}$$

W doświadczeniu wylosuję $n = 100$ próbek pochodzących ze standardowego rozkładu normalnego $N(0, 1)$, a następnie, korzystając z wielowymiarowej metody Newtona, wyszukam współrzędne punktu (μ_1, σ_1), w którym logarytm funkcji wiarogodności, zadany wzorem (1.28), osiąga największą wartość. Oczywiście spodziewam się wyniku zbliżonego do prawdziwych parametrów rozpatrywanego rozkładu normalnego $N(0, 1)$. Wartości gradientu i hesjanu, potrzebne do wielowymiarowego algorytmu Newtona, będą przybliżone, obliczone na podstawie metod przedstawionych w dodatku D.

```
n <- 100
y <- rnorm(n, 0, 1)
# l.w to funkcja zwracająca wartość logarytmu wiarogodności próbki, pochodzącej
    z rozkładu normalnego
l.w <- function(x){
  s <- sum((y - x[1])^2)
  return(-50 * log(2 * pi) - 100 * log(x[2]) - s / (2 * (x[2])^2))
}

# funkcja log.wiarog zwraca wektor, którego współrzędne to oodpowiednio:
    wartość logarytmu funkcji wiarogodności, przybliżona wartość gradientu
    logarytmu funkcji wiarogodności oraz przybliżona wartość hesjanu logarytmu
    funkcji wiarogodności
log.wiarog <- function(x){
  eps <- 0.000001
  e1 <- c(1, 0)
  e2 <- c(0, 1)
  l0 <- l.w(x)
  l1 <- (l.w(x + eps * e1) - l.w(x)) / eps
  l2 <- (l.w(x + eps * e2) - l.w(x)) / eps
  l11 <- (l.w(x + 2 * eps * e1) - 2 * l.w(x + eps * e1) + l.w(x)) / eps^2
  l12 <- (l.w(x + eps * e1 + eps * e2) - l.w(x + eps * e1) - l.w(x + eps * e2)
        + l.w(x)) / eps^2
  l22 <- (l.w(x + 2 * eps * e2) - 2 * l.w(x + eps * e2) + l.w(x)) / eps^2
  return(list(l.w, c(l1, l2), matrix(c(l11, l12, l12, l22), 2, 2)))
}
```

Rozpoczynając wyszukiwanie odpowiednio blisko punktu, w którym spodziewałam się, że występuje maksimum lokalne otrzymałam wyniki zgodne z oczekiwaniami – zbliżone do punktu o współrzędnych $(0, 1)$:

```
newton(Wiarogodnosc, c(0.1, 0.9))
[1] -0.03881049  0.97085314
newton(Wiarogodnosc, c(-1, 0.1))
[1] -0.03881049  0.97085314
```

Zastosowanie wielowymiarowych metod optymalizacji do *curve fitting*

Niech dane będą obserwacje $(x_1, y_1), ..., (x_n, y_n)$. Proces szukania takiej funkcji f, że $y_i \approx f(x_i)$ dla $i = 1, ...n$ nosi nazwę dopasowywania krzywej (ang. *curve fitting*). Zakłada się, że f jest zależna od wektora parametrów $\theta = (\theta_1, ...\theta_n)^T$ i tak dobiera θ^*, aby dopasowane punkty $\hat{y}_i = f(x_i; \theta^*)$ były jak najbliższe obserwacjom.

Na przykład, przy zawężeniu poszukiwania funkcji f tylko do funkcji kwadratowych można zapisać f jako funkcję $f(x) = ax^2 + bx + c$ i w tym przypadku wektorem parametrów jest wektor $\theta = (a, b, c)^T$. Do zmierzenia dopasowania funkcji do obserwacji używana jest *funkcja straty*, za pomocą której mierzona jest odległość pomiędzy $\mathbf{y} = (y_1, ..., y_n)^T$ a $\hat{\mathbf{y}} = (\hat{y}_1, ..., \hat{y}_n)^T$. Dwie, często używane funkcje straty to *metoda najmniejszych kwadratów*:

$$L_2(\theta) = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \tag{1.29}$$

oraz *suma wartości bezwzględnych różnic*:

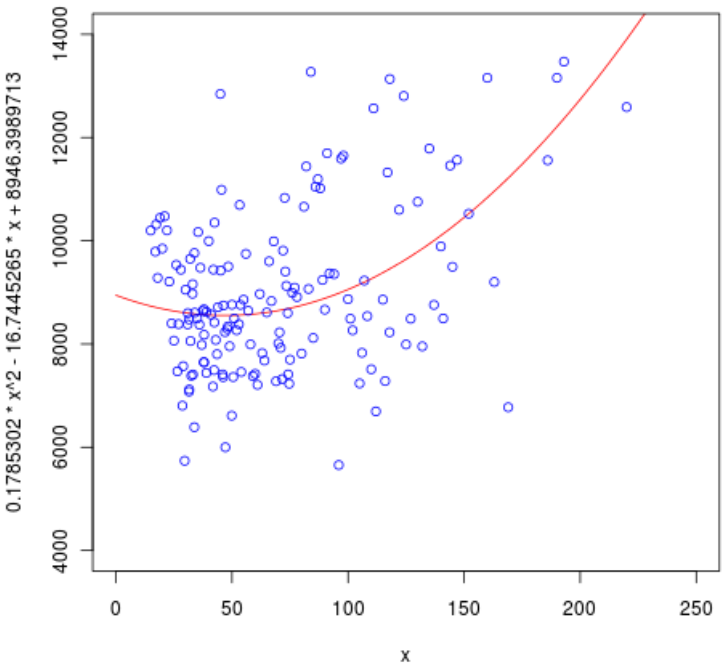
$$L_1(\theta) = \sum_{i=1}^{n} |y_i - \hat{y}_i|. \tag{1.30}$$

Przy użyciu metod optymalizacji, wybierany jest taki parametr θ^*, aby wartość funkcji straty była minimalna. Funkcja (1.30) jest nieróżniczkowalna, przy jej minimalizacji można więc skorzystać z metod numerycznych opisanych w dodatku D.

Przykład dopasowywania krzywej do danych przedstawię przy pomocy danych `apartments` pochodzących z pakietu `PBImisc` (źródło [8]). Zbadam zależność ceny za m^2 od całkowitej powierzchni dla 973 mieszkań. W przypadku powtarzających się powierzchni mieszkań, cenę za m^2 uśredniłam i uzyskane dane zapisałam w pliku `dane.txt`. Przy pomocy funkcji `optim` dostępnej w pakiecie R oraz funkcji straty sumy wartości bezwzględnych różnic (zadanej wzorem (1.30)) wyszukam funkcję kwadratową pasującą do zaobserwowanych danych. Prześledzę też jak zmieniać się będzie wykres szukanej funkcji, przy zwiększaniu maksymalnej liczby iteracji. Wyniki zobrazuję wykresami.

```
install.packages("PBImisc", dependencies = TRUE)
library(PBImisc)
data(apartments)
library(lattice)
# dane[,1] zawierać będzie powierzchnie całkowite mieszkań posortowane rosnąco
# dane[,2] zawierać będzie uśrednione ceny mieszkań za m^2 odpowiadające
    posortowanym cenom mieszkań
dane <- read.table("dane.txt")
# funkcja f2 to kwadratowa funkcja, której parametry będą wyszukiwane
f2 <- function(t, theta){
  return(theta[1] * t^2 + theta[2] * t + theta[3])
}
# loss.L1 to funkcja straty suma modułów, która będzie minimalizowana przy
    pomocy funkcji optim
loss.L1 <- function(theta){
  return(sum(abs(dane[ ,2] - f2(dane[ ,1], theta))))
}
# za punkty startowe funkcji optim przyjęte będą punkty (i,j,k) dla i,j,k
    całkowitych, należących do przedziału (-10,10)
# w wektorze min zachowane będą takie współczynniki funkcji kwadratowej, przy
    której wartość funkcji straty jest najmniejsza
# w wektorze pam, zachowany będzie punkt startowy, przy którym wartość funkcji
    straty jest najmniejsza
min <- optim(c(0, 0, 0), loss.L1)
for(i in -10:10){
  for(j in -10:10){
    for(k in -10:10){
      w <- optim(c(i, j, k), loss.L1)
      if(w[[2]] < min[[2]]){
        min <- w
        pam <- c(i,j,k)
      }
    }
  }
}
# zawartość wektora min
min
$par
[1]     0.1785302   -16.7445265   8946.3989713
# wartość funkcji wiarogodności
$value
[1] 226910.8
$convergence
# wartość zero oznacza, że metoda wyszukiwania użyta w funkcji optim była
    zbieżna do lokalnego minimum
[1] 0
```

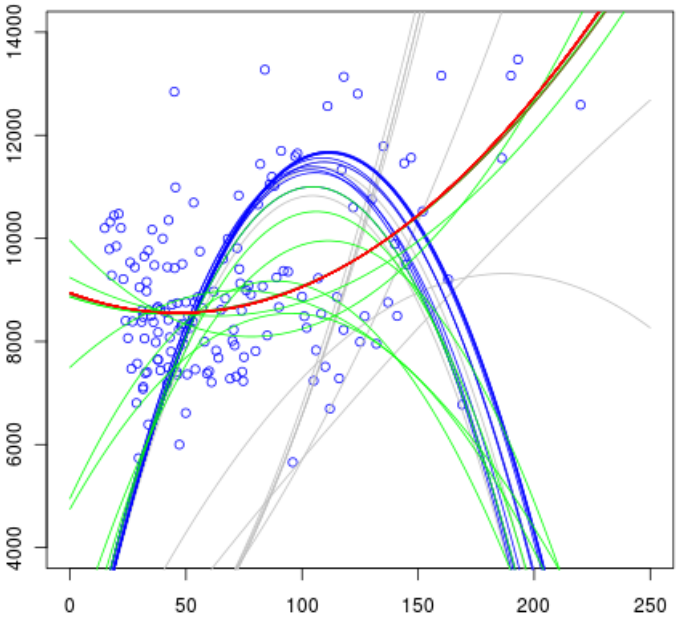
```
$message
NULL
# zawartość wektora pam
pam
[1]   1   6  10
```



Rysunek 1.3: Zależność ceny mieszkań za m^2 (w złotówkach) od ich całkowitej powierzchni.

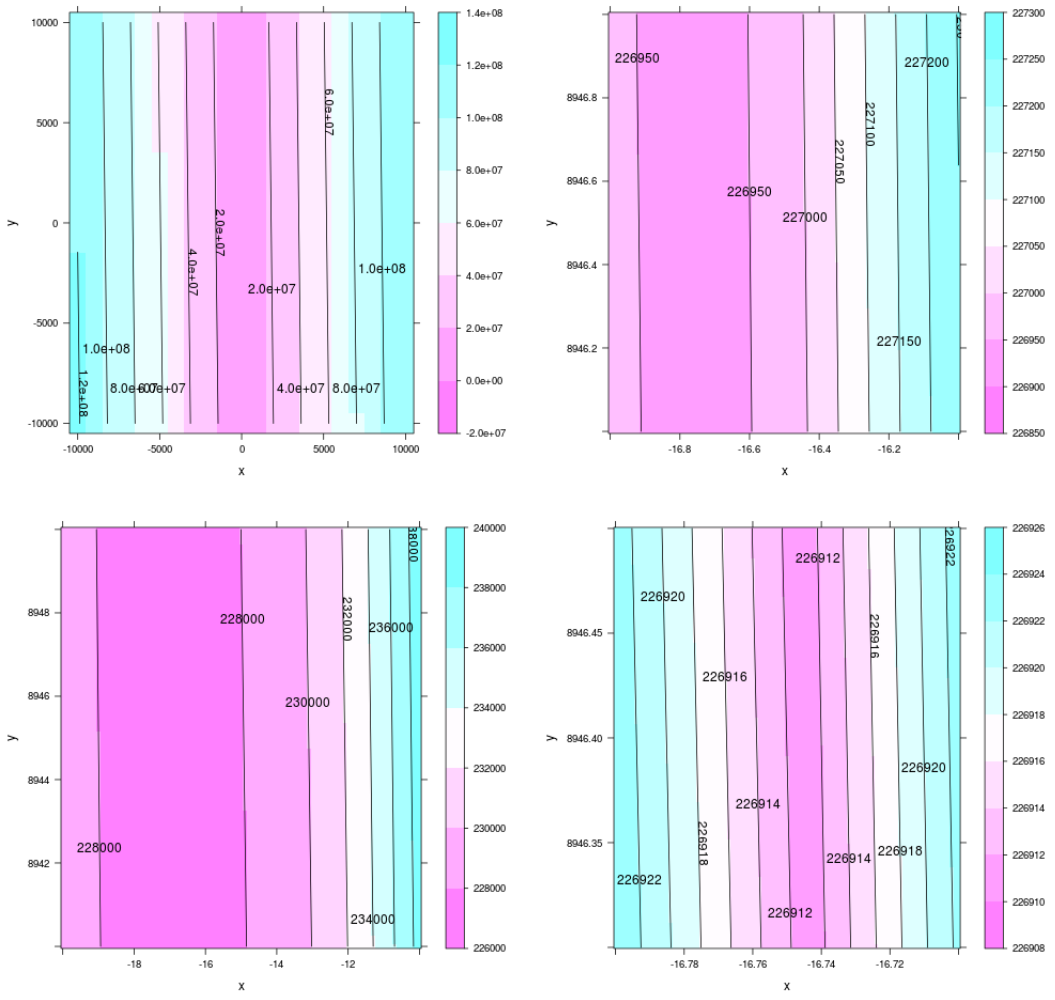
Prześledzę teraz zachowanie funkcji straty przy redukcji parametrów do dwóch, ustalając po kolei pierwszy, drugi i trzeci parametr (przyjmując w tym celu za stałe parametry wartości wyliczone wcześniej przez funkcję `optim`). Wyniki przedstawię przy pomocy wykresów konturowych, używając różnych przybliżeń.

```
# Przykładowe obliczenia, dla jednego z 12 wykresów konturowych zamieszczonych
    poniżej:
x <- seq(0.1, 0.2, 0.001)
y <- seq(-16.8, -16.7, 0.001)
xyz <- data.frame(matrix(0, length(x)*length(y), 3))
names(xyz) <- c('x', 'y', 'z')
n <- 0
for (i in 1:length(x))
{
  for (j in 1:length(y))
  {

    n <- n + 1
    xyz[n,] <- c(x[i], y[j], loss.L1(c(x[i], y[j], 8946.3989713)))
  }
}
library(lattice)
```

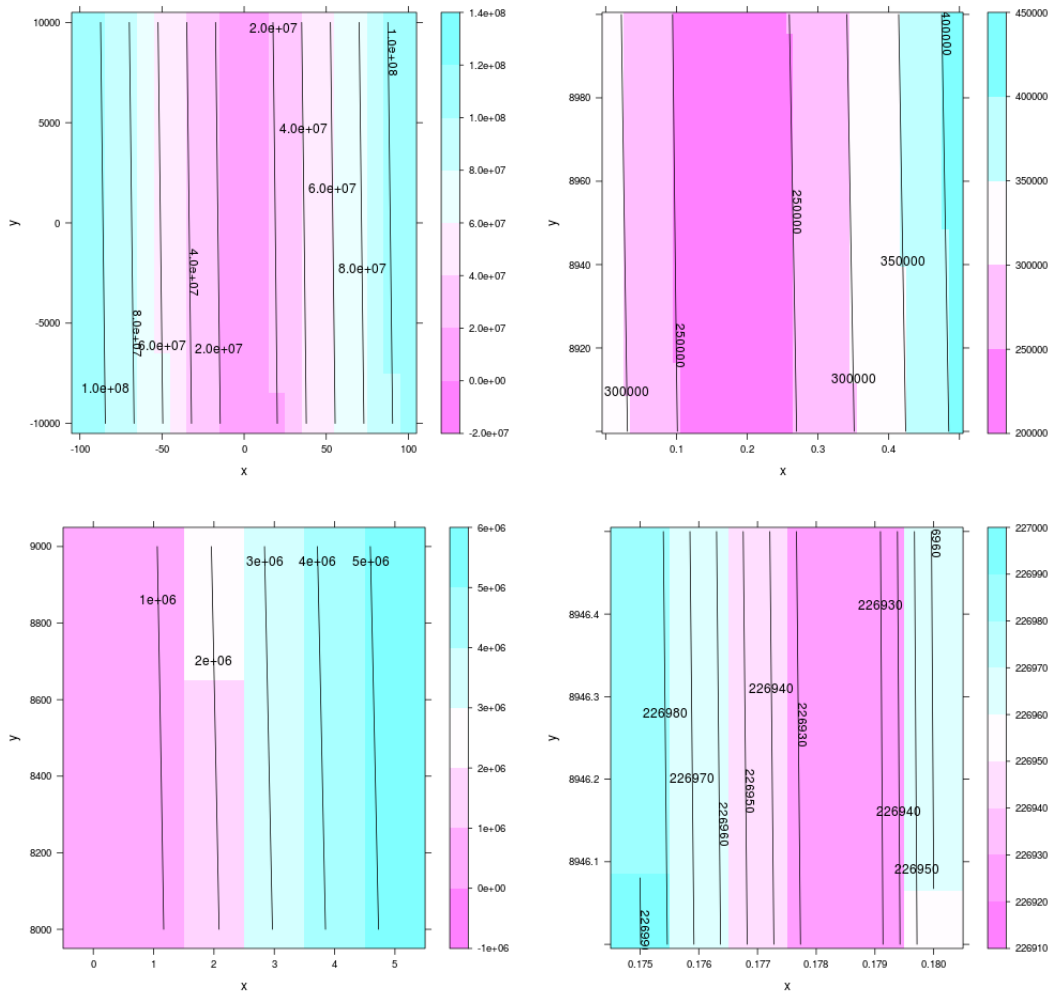


Rysunek 1.4: Zależność ceny mieszkań za m^2 (w złotówkach) od ich całkowitej powierzchni, podczas zwiększania liczby iteracji – n od 10 do 500, co 10. Dla n=500 algorytm optymalizacji był już zbieżny. Wykresy dla n należących do przedziałów: (10, 100) zaznaczone są kolorem szarym, (100, 200) kolorem niebieskim, (200, 300) kolorem zielonym, (300, 400) brązowym, zaś (400, 500) czerwonym.

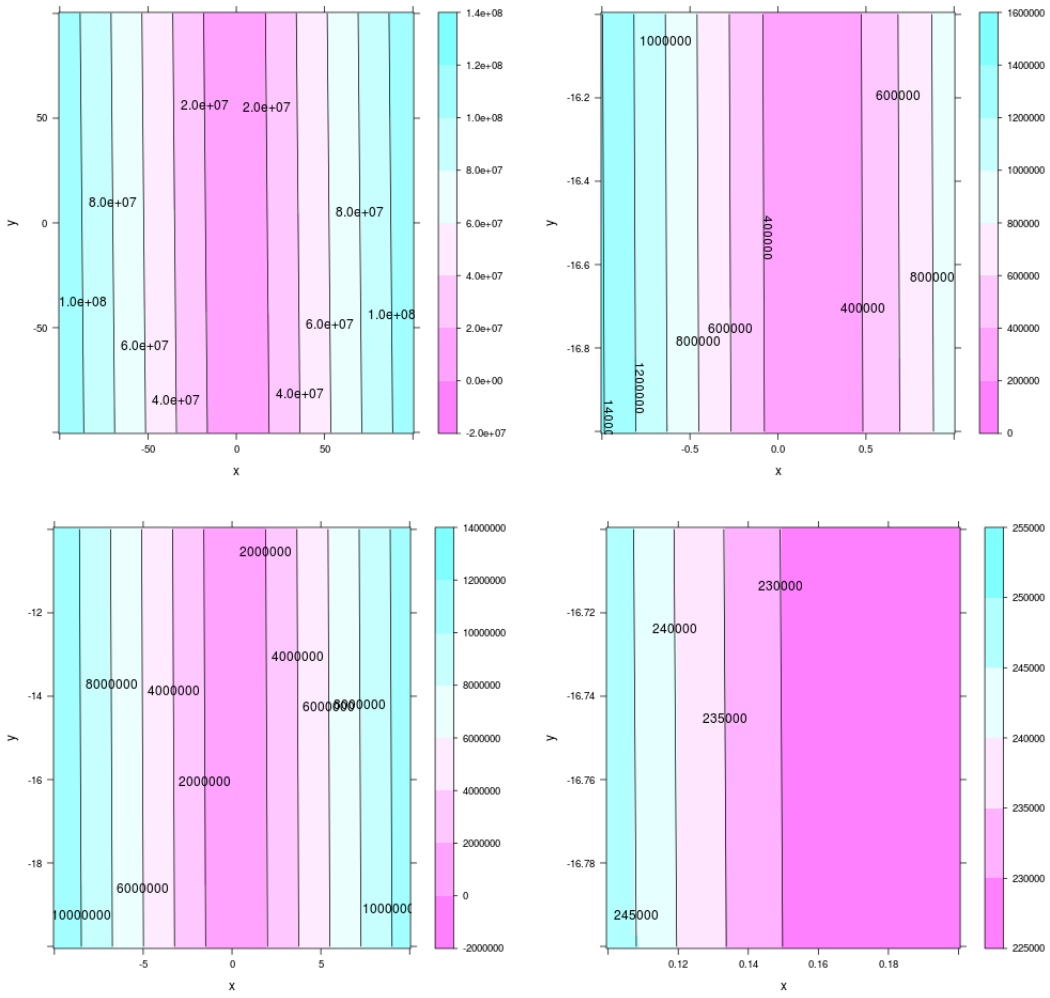
```
17 print (contourplot(z~x*y, data = xyz, region=TRUE))
```



Rysunek 1.5: Wykresy konturowe przedstawiające wartości funkcji straty po ustaleniu pierwszego parametru (przyjmując wartość tego parametru równą 0.1785302), przedstawione w coraz większych przybliżeniach.



Rysunek 1.6: Wykresy konturowe przedstawiające wartości funkcji straty po ustaleniu drugiego parametru (przyjmując wartość tego parametru równą -16.7445265), przedstawione w coraz większych przybliżeniach.



Rysunek 1.7: Wykresy konturowe przedstawiające wartości funkcji straty po ustaleniu trzeciego parametru (przyjmując wartość tego parametru równą 8946.3989713), przedstawiane w coraz większych przybliżeniach.

Rozdział 2

Algorytmy estymacji

W rozdziale tym omówię podstawowe algorytmy estymacji punktowej i przedziałowej. W przypadku estymacji punktowej posłużę się *metodą momentów*, *metodą kwantyli* oraz *metodą największej wiarogodności*. Dla zobrazowania estymacji przedziałowej opiszę zaś *przedziały ufności*. Źródłem informacji były [1], [2], [4] i [5].

2.1. Estymacja punktowa

Definicja 2.1.1 Estymatorem *nazywamy dowolną statystykę Z służącą do oszacowania nieznanej wartości parametru θ populacji statystycznej lub nieznanego rozkładu populacji.*

Zwykle istnieje wiele estymatorów dla danego parametru. Który z nich jest lepszy decydują ich własności. Przedstawię trzy, często używane estymatory powstałe przy pomocy metody momentów, metody największej wiarogodności oraz metody kwantyli.

2.1.1. Metoda momentów

Niech wektor θ będzie estymowanym parametrem. W metodzie momentów przyrównywane są momenty nieznanego rozkładu parametru θ do ich empirycznych odpowiedników. Układanych jest tyle równań, ile jest współrzędnych estymowanego wektora θ:

$$\begin{aligned} \mu(\theta) &= \hat{\mu} \\ m_k(\theta) &= \hat{m}_k, \end{aligned} \tag{2.1}$$

gdzie μ i $\hat{\mu}$ oznaczają odpowiednio wartość oczekiwaną i wartość oczekiwaną empiryczną:

$$\begin{aligned} \mu(\theta) &= \int x f_\theta(x) dx \\ \hat{\mu} &= \overline{X}, \end{aligned} \tag{2.2}$$

natomiast m_k i $\hat{m}_k$ oznaczają odpowiednio k-ty moment i k-ty moment empiryczny:

$$\begin{aligned} m_k(\theta) &= \int (x - \mu(\theta))^k f_\theta(x) \\ \hat{m}_k &= \frac{1}{n} \sum (X_i - \overline{X})^k. \end{aligned} \tag{2.3}$$

Przykład zastosowania metody momentów

Załóżmy, że mamy próbkę z rozkładu gamma o nieznanych parametrach α i λ. Chcemy estymować dwuwymiarowy wektor nieznanych parametrów $\theta = (\alpha, \lambda)$. Musimy zatem wykorzystać dwa pierwsze momenty rozkładu gamma: wartość oczekiwaną i wariancję. Dostajemy układ równań:

$$\begin{aligned} \frac{\alpha}{\lambda} &= \overline{X} \\ \frac{\alpha}{\lambda^2} &= \tilde{S}^2, \end{aligned} \tag{2.4}$$

gdzie $\tilde{S}^2$ to wariancja z próbki. Po rozwiązaniu układu równań otrzymujemy szukane estymatory:

$$\begin{aligned} \hat{\lambda} &= \frac{\overline{X}}{\tilde{S}^2} \\ \hat{\alpha} &= \frac{\overline{X}^2}{\tilde{S}^2}. \end{aligned} \tag{2.5}$$

Powyższy przykład zobrazuję przy pomocy danych `apartments` pochodzących z pakietu `PBImisc`. Przeanalizuję ceny mieszkań (w tysiącach złotych) za m^2. Założę, że pochodzą one z rozkładu gamma o nieznanych parametrach α, λ. Parametry te wyestymuję przy pomocy metody momentów. Wyniki przedstawię na rysunku 2.1.

```
#cena1000 zawiera ceny mieszkań za m2 w tys. zł.
lambda <- mean(cena1000) / var(cena1000) * (n - 1 / n)
alfa <- mean(cena1000) * lambda
# wartości estymowanych parametrów:
lambda
[1] 2.06173
alfa
[1] 18.01493
```

2.1.2. Metoda kwantyli

Metoda kwantyli stosowana jest, gdy momenty są trudne do obliczenia lub prowadzą do skomplikowanych układów równań. W metodzie tej przyrównywane są kwantyle teoretyczne nieznanego parametru θ do kwantyli obliczanych empirycznie, za pomocą próbki. Wybieramy tyle różnych kwantyli, ile mamy niewiadomych i dostajemy równania postaci:

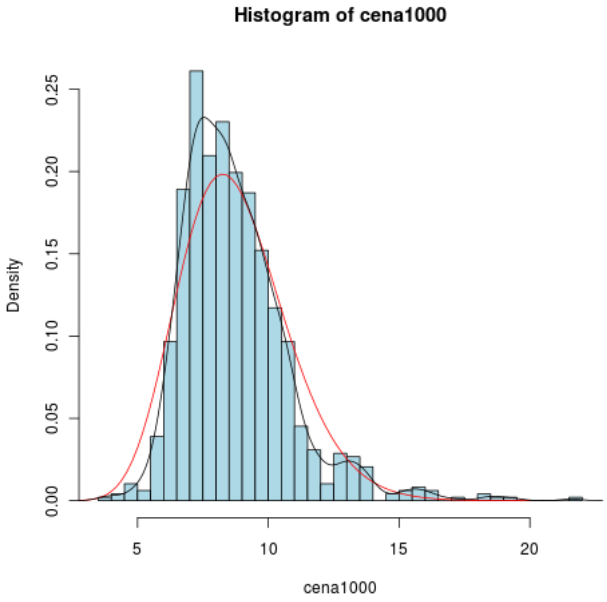
$$\xi_q(\theta) = \hat{\xi}_q. \tag{2.6}$$

Przykład zastosowania metody kwantyli

Rozważmy rozkład Weibulla o nieznanych parametrach $a > 0$ (parametr kształtu) i $b > 0$ (parametr skali) o dystrybuancie:

$$F_\theta = 1 - exp\left[-\left(\frac{x}{b}\right)^a\right] \ (x > 0). \tag{2.7}$$

Momenty tego rozkładu są trudne do obliczenia, ale przy estymowaniu parametru $\theta = (a, b)$ łatwo można skorzystać z równań dla kwantyli. Potrzebujemy dwóch równań, bo tyle jest



Rysunek 2.1: Histogram dla analizowanych cen mieszkań (w tysiącach złotych) za m^2. Na czerwono zaznaczony jest wykres gęstości gamma(2,06173; 18,01493), na czarno wykres gęstości powstały przy pomocy funkcji `density`.

niewiadomych parametrów. Wybieramy kwantyle rzędu 1/4 i 3/4 i dostajemy układ równań:

$$\begin{aligned} 1 - exp\left[-\left(\frac{\hat{\xi}_{1/4}}{b}\right)^a\right] &= 0.25 \\ 1 - exp\left[-\left(\frac{\hat{\xi}_{3/4}}{b}\right)^a\right] &= 0.75 \end{aligned} \tag{2.8}$$

po rozwiązaniu, którego otrzymujemy estymatory:

$$\begin{aligned} \hat{a} &= log\left(\frac{log(0.25)}{log(0.75)}\right) / (log\hat{\xi}_{3/4} - log\hat{\xi}_{1/4}) \\ \hat{b} &= \hat{\xi}_{1/4}/(-log(0.75))^{\frac{1}{\hat{a}}}. \end{aligned} \tag{2.9}$$

Można także porównywać więcej kwantyli empirycznych i teoretycznych niż jest niewiadomych parametrów. Mamy wtedy do czynienia z problemem nadokreślonych układów równań liniowych, gdy równań jest więcej niż niewiadomych. W takim przypadku nie można liczyć na to, że uda się znaleźć rozwiązanie spełniające wszystkie równania, gdyż jest ich za dużo. Dlatego konieczne jest znalezienie rozwiązania, które minimalizuje resztę.

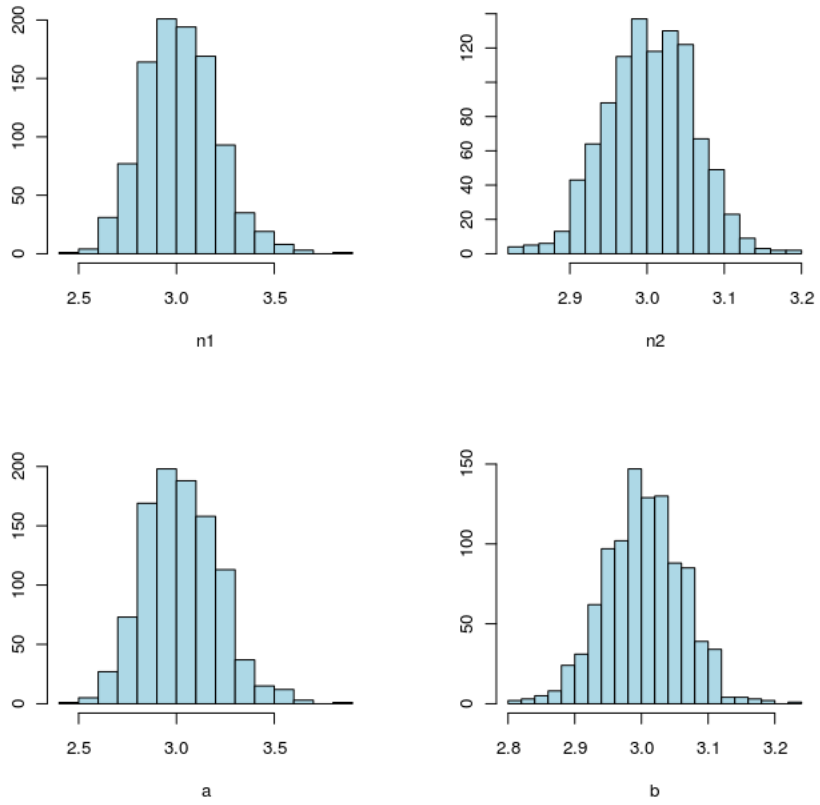
Przykładowe doświadczenie przeprowadzę na $n = 100$ próbkach losowanych $N = 1000$ razy z rozkładu Weibulla(3,3) i korzystając z obu przedstawionych wyżej metod (gdy równań jest tyle samo i więcej niż niewiadomych) dokonam estymacji parametrów skali i kształtu. Wyniki zobrazuję histogramami i wykresami pudełkowymi. W celu minimalizacji reszty, użyję funkcji straty najmniejszych kwadratów, a minimalizację funkcji straty przeprowadzę przy pomocy wielowymiarowego algorytmu Newtona.

```
# wektor wykorzystywanych kwantyli:
y <- c(0.25, 0.5, 0.75)
```

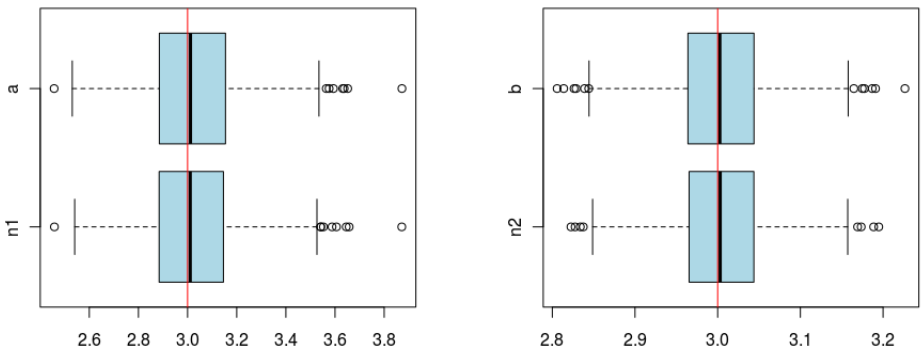
```
# wektor wylosowanych próbek
x <- rweibull(n, shape = 3, scale = 3)
q <- quantile(x)
q2 <- q[2]
q3 <- q[3]
q4 <- q[4]
# funckja Weibull za argumenty przyjmuje wartości parametrów rozkładu Weibulla
# funkcja Weibull zwraca wektor wartości dystrybuanty rozkładu Weibulla w
    punktach będących odpowiednio kwantylami rzędu 0.25, 0.5 i 0.75
Weibull <- function(theta){
  g1 <- pweibull(q2, theta[1], theta[2])
  g2 <- pweibull(q3, theta[1], theta[2])
  g3 <- pweibull(q4, theta[1], theta[2])
  return(c(g1, g2, g3))
}
# loss.L2 to funkcja straty najmniejszych kwadratów
loss.L2 <- function(theta){
  return(sum((y - W(theta))^2))
}

#funkcja loss2.L2 zwraca wartość funkcji straty jej gradientu i hesjanu
loss2.L2 <- function(x) {
  eps <- 0.0000001
  e1 <- c(1,0)
  e2 <- c(0,1)
  g1 <- (loss.L2(x + eps * e1) - loss.L2(x)) / eps
  g2 <- (loss.L2(x + eps * e2) - loss.L2(x)) / eps
  g11 <- (loss.L2(x + 2 * eps * e1) - 2 * loss.L2(x + eps * e1) + loss.L2(x)) /
      eps^2
  g12 <- (loss.L2(x + eps * e1 + eps * e2) - loss.L2(x + eps * e1) -
          loss.L2(x + eps * e2) + loss.L2(x)) / eps^2
  g22 <- (loss.L2(x + 2 * eps * e2) - 2 * loss.L2(x + eps * e2) + loss.L2(x)) /
      eps^2
  return(list(g, c(g1, g2), matrix(c(g11, g12, g12, g22), 2, 2)))
}

for(i in 1 : N){
  x <- rweibull(n, shape = 3, scale = 3)
  q <- quantile(x)
  q2 <- q[2]
  q3 <- q[3]
  q4 <- q[4]
# wartości współczynników rozkładu Weibulla, przy których wartość funkcji
    straty jest najmniejsza
  n1[i] <- newton(loss2.L2, c(3, 3))[1]
  n2[i] <- newton(loss2.L2, c(3,3))[2]
# wartości współczynników rozkłady Weibulla, gdy równań jest tyle samo co
    niewiadomych
  a[i] <- log((log(0.25) / log(0.75))) / (log(q4) - log(q2))
  b[i] <- q2 / ((-log(0.75))^(1 / a[i]))
}
```

2.1.3. Metoda największej wiarogodności

Główną ideą metody największej wiarogodności jest oszacowanie nieznanych parametrów tak, aby empiryczne dane były najbardziej prawdopodobne. Aby znaleźć taki estymator konstruuje się funkcję wiarogodności (def.1.3.1).



Rysunek 2.2: Histogramy estymowanych parametrów rozkładu Weibulla: po lewej parametru kształtu, po prawej parametru skali. W pierwszym rzędzie przedstawione są wyniki estymacji metodą kwantyli, gdy liczba równań jest większa niż liczba parametrów, w drugim rzędzie, gdy liczba parametrów jest równa liczbie równań.



Rysunek 2.3: Wykresy pudełkowe przedstawiające wartości estymowanych parametrów. Wykres po lewej stronie dotyczy estymacji parametru kształtu, po prawej zaś parametru skali.

Definicja 2.1.2 Estymatorem największej wiarogodności (ENW) *parametru θ jest ta jego wartość, przy której funkcja wiarogodności jest największa.*

Gdy funkcja wiarogodności L jest różniczkowalna jej maksimum można znaleźć szukając w miejscach zerowania się pochodnych. Maksimum można też szukać używając metod optymalizacji (tak jak w rozdziale 1.3). Ze względu na to, że funkcja L jest złożona z iloczynów często wygodniej jest badać pochodne funkcji $\ln L$.

Przykład zastosowania metody największej wiarogodności

Rozpatrzmy rozkład wykładniczy z parametrem λ. Funkcja wiarogodności przyjmuje postać:

$$L(\lambda) = \lambda^n e^{-\lambda(x_1+...+x_n)}.$$

Po zlogarytmowaniu i zróżniczkowaniu otrzymujemy:

$$\frac{n}{\lambda} - (x_1 + ... + x_n) = 0,$$

czyli estymatorem ENW jest:

$$\hat{\lambda} = \frac{1}{\bar{X}}.$$

2.1.4. Porównanie metody momentów, metody kwantyli i metody największej wiarogodności

Załóżmy, że mamy próbkę z rozkładu $Exp(\lambda) = Gamma(1, \lambda)$ o nieznanym parametrze λ. Aby porównać estymatory otrzymane metodą momentów, metodą kwantyli i metodą największej wiarogodności będę estymowała parametr λ tymi trzema metodami. W tym celu wygeneruję próbki rozmiaru $n = 30$ z rozkładu $Gamma(1, 3)$. Doświadczenie powtórzę $N = 1000$ razy, a wyniki zapiszę w tablicach `EMM`, `ENW` i `EMK`.

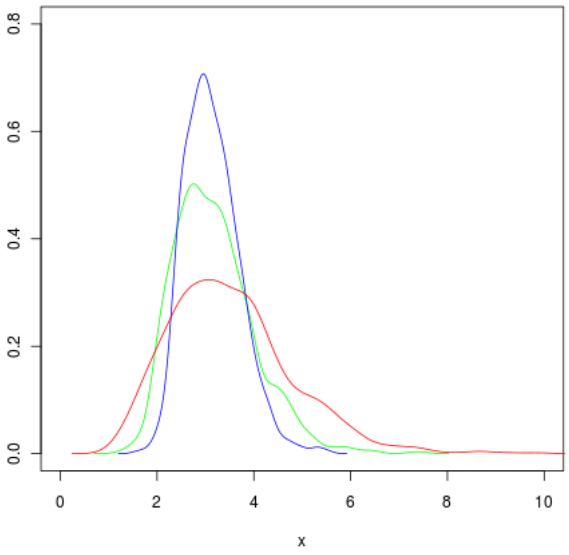
```
for(i in 1:N)
{
  x <- rgamma(n,1,3)
  EMM[i] <- mean(x) / var(x) * (n - 1 / n)
  ENW[i] <- 1 / mean(x)
  EMK[i] <- log(0.5)/-quantile(x)[3]
}
```

Uzyskane wykresy gęstości i wykresy pudełkowe przedstawiające obliczone wartości estymatorów (przedstawione na rysunkach 2.4 i 2.5) pokazują, że estymatory uzyskane metodą największej wiarogodności najczęściej (spośród analizowanych estymatorów) przyjmują wartości bliskie 3. Wartości uzyskane tą metodą charakteryzują się też najmniejszą rozbieżnością wyników.

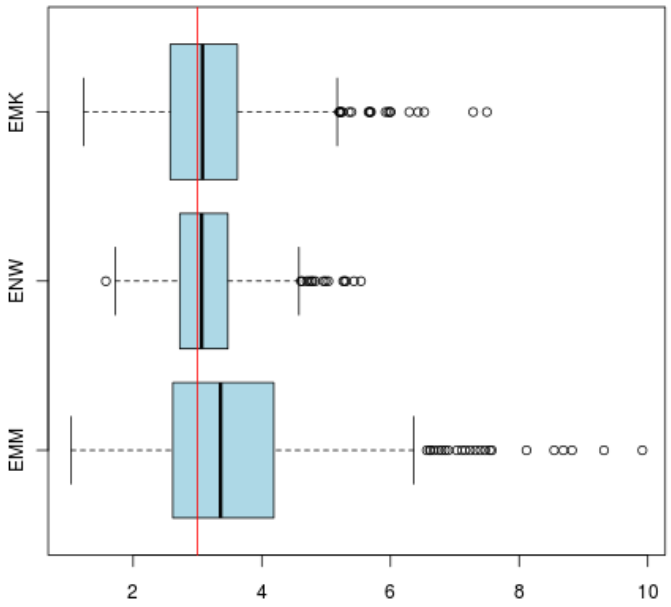
Więcej na temat metody momentów, metody największej wiarogodności oraz metody kwantyli można znaleźć w [2] i [5].

2.2. Przedziały ufności

Ideą estymacji przedziałowej jest znalezienie przedziału (Z_1, Z_2), w którym nieznany parametr θ znajdzie się z zadowalającym, zbliżonym do jedynki prawdopodobieństwem. Końce przedziału są statystykami $Z_1 = g_1(X_1, ..., X_n)$, $Z_2 = g_2(X_1, ..., X_n)$.



Rysunek 2.4: Wykresy gęstości estymatorów uzyskane przy pomocy funkcji `density`. Na niebiesko zaznaczono wykres gęstości dla estymatorów powstałych metodą największej wiarogodności, na zielono dla estymatorów powstałych metodą kwantyli, na czerwono dla estymatorów powstałych metodą momentów.



Rysunek 2.5: Wykresy pudełkowe otrzymane w wyniku szacowania parametru λ metodą momentów, metodą największej wiarogodności i metodą kwantyli.

Definicja 2.2.1 *Niech $g(\theta)$ będzie funkcją nieznanego parametru. Rozważmy dwie statystyki $Z_1 = g_1(X_1, ..., X_n)$ i $Z_2 = g_2(X_1, ...X_n)$. Mówimy, że $[Z_1, Z_2]$ jest przedziałem ufności dla $g(\theta)$ na poziomie ufności $1 - \alpha$ jeśli dla każdego θ:*

$$\mathbb{P}(Z_1 \leqslant g(\theta) \leqslant Z_2) \geqslant 1 - \alpha.$$

Zazwyczaj α jest mała liczbą, np. $\alpha = 0.1$, $\alpha = 0.05$, $\alpha = 0.01$. Przedziały ufności można interpretować następująco: przy wielokrotnym losowaniu prób n–elementowych i obliczaniu przy ich pomocy granic przedziału, parametr $g(\theta)$ należy do obliczonego przedziału średnio w $(1-\alpha)100\%$ przypadkach. Podstawą konstrukcji przedziału dla zadanego parametru $g(\theta)$ jest dobry estymator tego parametru. Sposób określenia przedziału ufności zależy od rozkładu, w którym występuje nieznany parametr, od tego czy znamy pozostałe parametry w tym rozkładzie oraz od liczebności próby. Więcej na temat przedziałów ufności można znaleźć w [4] i [5].

2.2.1. Przykłady zastosowań przedziałów ufności

Poniżej omówię przykładowe tworzenie przedziałów ufności dla średniej i wariancji, dla rozkładu normalnego oraz przedstawię symulację przedziałów ufności dla średniej.

Przykłady przedziałów ufności dla średniej

Załóżmy, że mamy model, w którym populacja generalna ma rozkład $N(m, \sigma^2)$. Nieznany jest parametr m, dla którego szukamy przedziału ufności.

Załóżmy najpierw, że wariancja jest znana. Ponieważ:

$$\frac{\overline{X} - m}{\sigma/\sqrt{n}} \sim N(0, 1) \tag{2.10}$$

to jeżeli z jest kwantylem rzędu $1 - \alpha/2$ rozkładu $N(0, 1)$ to zachodzi:

$$\mathbb{P}_\mu(|\sqrt{n}(\overline{X} - m)/\sigma| \leqslant z) = \Phi(z) - \Phi(-z) = 1 - \alpha. \tag{2.11}$$

Czyli przedział ufności dla m ma postać:

$$[\overline{X} - \sigma z/\sqrt{n}, \overline{X} + \sigma z/\sqrt{n}]. \tag{2.12}$$

Niech teraz wariancja σ^2 będzie nieznana. Zastosujemy nieobciążony estymator wariancji:

$$S^2 = \sum(X_i - \overline{X})^2/(n - 1). \tag{2.13}$$

Niech $S = \sqrt{S^2}$. Zachodzi:

$$\frac{\overline{X} - m}{S/\sqrt{n}} \sim t(n - 1), \tag{2.14}$$

gdzie $t(n-1)$ jest rozkładem t-Studenta z $n-1$ stopniami swobody. Jeżeli z jest kwantylem rzędu $1 - \alpha/2$ to:

$$\mathbb{P}_{m,\sigma}(|\sqrt{n}(\overline{X} - m)/S| \leqslant z) = 1 - \alpha. \tag{2.15}$$

Przedział ufności na poziomie istotności $1 - \alpha$ jest więc postaci:

$$[\overline{X} - St/\sqrt{n}, \overline{X} + St/\sqrt{n}], \tag{2.16}$$

gdzie $t = t_{1-\frac{\alpha}{2}}(n - 1)$.

Symulacja przedziałów ufności dla średniej

Przedstawię symulację przedziałów ufności dla średniej, w przypadku, gdy wariancja jest znana. Wygeneruję próbki rozmiaru $n = 15$ z rozkładu $N(\mu, \sigma^2)$. Prawdziwe wartości parametrów użyte w symulacjach to $\mu = 4$ i $\sigma = 2$. Obliczę przedziały ufności dla μ. Wykonam $N = 20$ doświadczeń. Poziom istotności wynosił będzie 0.95. Krańce lewych i prawych przedziałów zapiszę w tablicach `L` i `P`:

```
for(i in 1:N){
  k <- rnorm(n, 4, 2)
  q <- qnorm(0.975, 0, 1)
  L[i] <- mean(k) - (2 * q / sqrt(15))
  P[i] <- mean(k) + (2 * q / sqrt(15))
}
```

Otrzymałam następujące wyniki:

nr próby	L	P	czy $\mu \in$?
1	3.730729	5.754971	TAK
2	2.875956	4.900198	TAK
3	2.901103	4.925345	TAK
4	2.602048	4.626290	TAK
5	2.867086	4.891328	TAK
6	3.048413	5.072655	TAK
7	3.561889	5.586131	TAK
8	3.234790	5.259032	TAK
9	2.827525	4.851767	TAK
10	3.026169	5.050411	TAK
11	2.110822	4.135065	TAK
12	3.177976	5.202218	TAK
13	3.581776	5.606018	TAK
14	2.810548	4.834791	TAK
15	3.095057	5.119299	TAK
16	2.219202	4.243444	TAK
17	2.723671	4.747913	TAK
18	2.794172	4.818414	TAK
19	3.503764	5.528006	TAK
20	2.810442	4.834684	TAK

Wszystkie obliczone przedziały ufności zawierały μ. Wynik ten jest zgodny z oczekiwaniami i z definicją (2.2.1).

Przykład przedziału ufności dla wariancji

Rozważmy ten sam model co przy estymacji wartości oczekiwanej. Nieznanymi parametrami są m i σ^2. Zachodzi:

$$\frac{(n-1)S^2}{\sigma^2} \sim \chi^2(n-1). \tag{2.17}$$

Niech c_1 i c_2 będą kwantylami rozkładu chi-kwadrat z $n-1$ stopniami swobody rzędów $\alpha/2$ i $1-\alpha/2$ odpowiednio. Zachodzi:

$$\mathbb{P}_{m,\sigma}(c_1 \leqslant (n-1)S^2/\sigma^2 \leqslant c_2) = 1-\alpha. \tag{2.18}$$

Szukanym przedziałem ufności jest zatem:

$$[(n-1)S^2/c_2, (n-1)S^2/c_1]. \tag{2.19}$$

Rozdział 3

Algorytmy redukcji wariancji

Algorytmy redukcji wariancji są procedurami pozwalającymi na zwiększenie precyzji estymacji, poprzez odpowiedni dobór próby (ang. *sampling*). Używając symulacji do estymowania nieznanych parametrów, wartości estymatorów dla każdej symulacji będą różne. Jest to skutkiem losowości. Oczywiście pożądane jest, aby zmienność wyników była jak najmniejsza. Rezultat taki można osiągnąć przez zwiększenie liczebności próby, ale jest to zbyt czasochłonne i często lepszy efekt można uzyskać korzystając z poniżej przedstawionych metod redukcji wariancji. Źródłem informacji były [2] i [6].

3.1. Metoda *antithetic sampling*

Niech θ będzie estymowanym parametrem oraz niech X i Y będą jego nieobciążonymi estymatorami o takich samych, skończonych wariancjach $\sigma_X^2 = \sigma_Y^2 = \sigma^2$. Estymator $Z = (X+Y)/2$ jest też nieobciążony. Estymatory X, Y, Z można porównać badając ich wariancję. Zachodzi:

$$VarZ = \frac{1}{4}VarX + \frac{1}{4}VarY + \frac{1}{2}Cov(X,Y) = \frac{1}{2}(\sigma^2 + Cov(X,Y)). \tag{3.1}$$

Jeżeli X i Y są estymatorami niezależnymi to wariancja estymatora Z jest mniejsza dwukrotnie. Wariancję estymatora Z można zmniejszyć jeszcze bardziej, jeżeli kowariancja estymatorów X i Y będzie ujemna. Dzieje się tak wtedy, gdy zmienne X i Y są ujemnie skorelowane. Powyższa obserwacja pokazuje, że bardziej można zredukować wariancję uśredniając estymatory, które są ujemnie skorelowane niż takie, które są niezależne. Jest to ideą metody *antithetic sampling*.

3.1.1. Ogólna technika metody *antithetic sampling*

Niech Z będzie zmienną losową, której wartość oczekiwaną chcemy estymować. Niech $\theta = \mathbb{E}(Z)$ oraz $Var(Z) = \sigma^2$. Parametr θ można estymować używając $2n$ niezależnych obserwacji wartości zmiennej Z, używając estymatora:

$$\hat{\theta}_{2n} = \sum_{i=1}^{2n} Z_i/2n, \tag{3.2}$$

którego wariancja wynosi:

$$Var(\hat{\theta}_{2n}) = \sigma^2/2n. \tag{3.3}$$

Załóżmy teraz, że możemy wygenerować $2n$ niezależnych par obserwacji (X_i, Y_i), gdzie X_i oraz Y_i mają tę samą dystrybuantę co Z i są ujemnie skorelowane. Można wtedy estymować parametr θ używając nieobciążonego estymatora:

$$\hat{\theta}_{2n}^{ant} = (\overline{X} + \overline{Y})/2, \tag{3.4}$$

gdzie $\overline{X}$, $\overline{Y}$ oznaczają średnią empiryczną. Zachodzi wtedy:

$$\begin{aligned} Var(\hat{\theta}_{2n}^{ant}) &= \frac{1}{4}(Var(\overline{X}) + Var(\overline{Y}) + 2Cov(\overline{X}, \overline{Y})) \\ &= \frac{\sigma^2}{2n} + \frac{1}{2n}Cov(X_1, Y_1) \\ &= \frac{\sigma^2}{2n}(1 + \rho(X_1, Y_1)), \end{aligned} \tag{3.5}$$

gdzie $\rho(X_1, Y_1)$ jest korelacją X_1 i Y_1. Estymator $\hat{\theta}_{2n}^{ant}$ skonstruowany metodą zmiennych antytetycznych charakteryzuje się mniejszą wariancją o $100\rho\%$ w porównaniu do estymatora $\hat{\theta}_{2n}$.

Generacja zmiennych antytetycznych

Z poprzedniego rozdziału widać, że wariancja estymatora $\hat{\theta}_n^{ant}$ jest tym mniejsza im ρ bliższe -1. Standardowym sposobem konstruowania zmiennych losowych o ujemnej korelacji jest stosowanie funkcji odwrotnej do dystrybuanty i zastosowanie „odwróconych liczb losowych".

Stwierdzenie 1 *Jeśli* $g : [0,1] \to \mathbb{R}$ *jest funkcją monotoniczną różną od stałej,* $\int_0^1 g(u)^2 du < \infty$ *oraz* $U \sim U(0,1)$, *to*

$$Cov(g(U), g(1-U)) < 0.$$

Dowód. Załóżmy bez straty ogólności, że g jest funkcją rosnącą. Można wybrać takie u^*, że $g(1-u) > \theta$ jeżeli $u < u^*$ oraz $g(1-u) < \theta$ jeżeli $u > u^*$.

Z tego wynika:

$$\begin{aligned} Cov(g(U), g(1-U)) &= \mathbb{E}(g(U) - \theta)(g(1-U) - \theta) \\ &= \mathbb{E}g(U)(g(1-U) - \theta) \\ &< g(u^*)\int_1^{u^*}(g(1-u) - \theta)du + g(u^*)\int_{u^*}^1 (g(1-u) - \theta)du = 0 \end{aligned} \tag{3.6}$$

Przez symetrię można to samo pokazać dla funkcji g, która jest malejąca.

Definicja 3.1.1 *Dla dowolnej dystrybuanty* F *określamy* uogólnioną funkcję odwrotną F^{-1} *następującym wzorem:*

$$F^{-1}(u) = inf\{x : F(x) \geqslant u\}.$$

Niech $U_i \sim U(0,1)$, $X_i = F^{-1}(U_i)$ oraz $Y_i = F^{-1}(1 - U_i)$. X_i i Y_i mają tą samą dystrybuantę F i są ujemnie skorelowane (co można dostać od razu podstawiając $g = F^{-1}$ do Stwierdzenia 1). Z tej zależności można skorzystać, aby wyprodukować parę obserwacji ujemnie skorelowanych (X_i, Y_i) o tej samej zadanej dystrybuancie F.

3.1.2. Zastosowanie metody *antithetic sampling*

Przedstawię zastosowanie metody *antithetic sampling* do obliczania wartości całki $\theta = \int_0^1 g(u)du$, gdzie g jest funkcją rosnąca na przedziale $[0,1]$. Przybliżoną wartość parametru θ można obliczyć korzystając z metody Monte Carlo, opartej na $2n$ obserwacjach:

$$\hat{\theta}_{2n} = \frac{1}{2n}\sum_{i=1}^{2n} g(U_i), \tag{3.7}$$

gdzie $U_1, ..., U_n \sim_{iid} U(0,1)$. Można też skorzystać z metody *antithetic sampling*.

Niech $(X_i, Y_i) = (g(U_i), g(1-U_i))$. $1-U_i \sim U(0,1)$, zatem X_i oraz Y_i pochodzą z rozkładu o tej samej dystrybuancie. Przybliżoną wartość parametru θ można więc obliczyć korzystając ze wzoru:

$$\hat{\theta}_{2n}^{ant} = \frac{1}{4n}(\sum_{i=1}^{2n} g(U_i) + \sum_{i=1}^{2n} g(1-U_i)). \tag{3.8}$$

Zachodzi $\mathbb{E}(X_i) = \mathbb{E}(Y_i) = \theta$ oraz $\hat{\theta}_{2n}^{ant}$ ma mniejszą wariancję, gdyż kowariancja zmiennych (X_i, Y_i) jest ujemna.

Powyższe rozważania zastosuję do obliczenia przybliżonej wartości całki na przedziale $(0,1)$ funkcji:

$$g(x) = e^{-x^2/2}.$$

W tym celu wylosuję $n = 100$ próbek z rozkładu jednostajnego $U(0,1)$, wyprodukuję pary obserwacji ujemnie skorelowanych i dokonam obliczenia parametru θ dwoma przedstawionymi wyżej sposobami: metodą Monte Carlo i za pomocą metody *antithetic sampling*. Doświadczenie powtórzę $N = 1000$ razy i sprawdzę jak zmieniła się wariancja po zastosowaniu metody *antithetic sampling*.

```
# g to funkcja, której całka na przedziale (0,1) będzie przybliżana
g <- function(x){
  return(exp(-x^2 / 2))
}
# w tablicy MC znajdują się obliczone metodą Monte Carlo wartości całki
# w tablicy Ant znajdują się przybliżone wartości całki wyliczone metodą
    antithetic sampling
for(i in 1:N){
  u <- runif(n, min = 0, max = 1)
  MC[i] <- mean(g(u))
  Ant[i] <- (mean(g(u)) + mean(g(1 - u))) / 2
}
# wariancja po skorzystaniu z metody Monte Carlo
var(MC)
[1] 0.0001449233
# wariancja po skorzystaniu z metody antithetic sampling
var(Ant)
[1] 2.157682e-05
# obliczenie redukcji wariancji
red <- (var(MC) - var(Ant)) / var(MC)
red
[1] 0.8511156
```

Redukcja wariancji uzyskana metodą *antithetic sampling* wyniosła około $85,11\%$.

3.2. Metoda emphimportance sampling

Metoda *importance sampling* (losowanie istotne) znajduje zastosowanie głównie przy obliczaniu wartości całek lub sum. Ideą losowania istotnego jest kompensacja wagami szybkiej zmienności estymowanej funkcji podcałkowej. Losowanie punktów nie jest równomierne – punkty losowane są tym gęściej, im funkcja jest bardziej zmienna.

3.2.1. Ogólna technika metody importance sampling

Niech X będzie zmienną losową o gęstości f, a estymowanym parametrem $\theta = \mathbb{E}\phi(X) = \int \phi(x)f(x)dx$. Niech Y będzie zmienną losową o gęstości g, zbliżonej do ϕf. Zachodzi wtedy:

$$\mathbb{E}\phi(X) = \int \phi(x)f(x)dx = \int \frac{\phi(x)f(x)}{g(x)}g(x)dx = \int \psi(x)g(x)dx = \mathbb{E}\psi(Y), \tag{3.9}$$

gdzie $\psi(x) = \phi(x)f(x)/g(x) = w(x)\phi(x)$.

Niech $Y_1, ..., Y_n$ będą niezależnymi zmiennymi losowymi o gęstości g. Wtedy estymator *importance sampling* jest postaci:

$$\hat{\theta}_n^{IS} = \frac{1}{n}\sum_{i=1}^{n}\psi(Y_i) = \frac{1}{n}\sum_{i=1}^{n}w(Y_i)\phi(Y_i), \tag{3.10}$$

gdzie $w(x) = f(x)/g(x)$. $\hat{\theta}_n^{IS}$ jest estymatorem nieobciążonym z wariancją:

$$Var(\hat{\theta}_n^{IS}) = \frac{1}{n}Var\psi(Y_1). \tag{3.11}$$

Oczywiście im bardziej g jest zbliżona do ϕf, tym bardziej ψ zbliżone jest do stałej i tym większą można osiągnąć redukcję wariancji.

3.2.2. Zastosowanie metody *importance sampling*

Rozważmy tak jak poprzednio problem obliczenia całki:

$$\theta = \textstyle\int_0^1 e^{-x^2/2}dx,$$

Stosując metodę *importance sampling* szukamy funkcji $h(x)$, która (analogicznie do funkcji $g(x)$ w rozdziale 3.2.1) jest zbliżona do funkcji $e^{-x^2/2}$. Szukana funkcja $h(x)$ musi być gęstością pewnego rozkładu. Aby uzyskać funkcję $h(x)$ rozpatrujemy rozwinięcie w szereg Taylora. Po zastosowaniu rozwinięcia funkcji $e^{x^2/2}$ w szereg Taylora wokół zera otrzymujemy:

$$h_1(x) = e^{-x^2/2} = \frac{1}{e^{x^2/2}} \approx \frac{1}{1 + x^2/2}. \tag{3.12}$$

Modyfikujemy funkcję h_1 tak żeby była gęstością pewnego rozkładu. Ponieważ:

$$\int_0^x \frac{1}{1 + u^2/2}du = \sqrt{2}arctan\left(\frac{x}{\sqrt{2}}\right), \tag{3.13}$$

a gęstość musi całkować się do jedynki, otrzymujemy dystrybuantę będącą całką szukanej gęstości $h(x)$:

$$H(x) = \frac{arctan\left(\frac{x}{\sqrt{2}}\right)}{arctan\left(\frac{1}{\sqrt{2}}\right)}\ x \in (0, 1). \tag{3.14}$$

Możemy wygenerować obserwacje pochodzące z rozkładu o gęstości h korzystając z funkcji $H^{-1}(u) = \sqrt{2}tan\left(u\ arctan\left(1/\sqrt{2}\right)\right)$. Korzystając ze wzoru (3.10) oraz licząc pochodną funkcji $H(x)$ (i dostając w ten sposób gęstość $h(x)$) uzyskujemy estymator importance sampling parametru θ:

$$\hat{\theta}_n^{IS} = \frac{1}{n}\sum_{i=1}^{n} e^{-X_i^2/2}\sqrt{2}arctan\left(\frac{1}{\sqrt{2}}\right)\left(1+\frac{X_i^2}{2}\right), \tag{3.15}$$

gdzie $X_i = H^{-1}(U_i)$ oraz $U_1, ..., U_n$ są niezależnymi zmiennymi losowymi pochodzącymi z rozkładu $U(0,1)$.

Przedstawię teraz obliczenia redukcji wariancji, jaką można uzyskać stosując estymator importance sampling. Obliczenia wykonam dla $n = 100$ próbek losowanych z rozkładu jednostajnego. Doświadczenie powtórzę $N = 1000$ razy. Wariancję porównam z wariancją otrzymaną przy obliczaniu wartości całki metodą Monte Carlo.

```
# g to funkcja, której wartość całki na przedziale (0,1) będzie przybliżana
# w tablicy MC znajdować się będą obliczone metodą Monte Carlo przybliżone
    wartości całki
# w tablicy IS znajdować się będą obliczone metodą importance sampling
    przybliżone wartości całki
for(i in 1:N){
  u <- runif(n, min = 0, max = 1)antithetic
  MC[i] <- mean(g(u))
  IS[i] <- mean(g(u) * sqrt(2) * atan(1 / sqrt(2)) * (1 + u^2 / 2))
}
# wariancja po wykorzystaniu metody Monte Carlo
var(MC)
[1] 4.121681e-06
# wariancja wartości obliczonych metodą importance sampling
var(IS)
[1] 1.26943e-07
# obliczenie redukcji wariancji
red <- (var(MC) - var(IS)) / var(MC)
red
[1] 0.9692012
```

Redukcja wariancji po zastosowaniu metody *importance sampling* wyniosła około $96,92\%$ i była większa niż uzyskana metodą *antithetic sampling*.

3.3. Metoda control variates

Podobnie jak w metodzie zmiennych antytetycznych, w metodzie *control variates* redukcję wariancji można uzyskać dzięki kowariancji, ale w tym przypadku dodatniej. Główną ideą tej metody jest użycie zmiennej Y, ze znaną wartością oczekiwaną μ, do kontrolowania innej zmiennej X, z nieznaną wartością oczekiwaną θ, którą chcemy estymować.

Załóżmy, że $Cov(X,Y) > 0$. Kontrolowaną wersją zmiennej X jest:

$$X^* = X - \alpha\left(Y - \mu\right), \tag{3.16}$$

gdzie $\alpha > 0$ jest pewną stałą. Oczywiście $\mathbb{E}(X^*) = \theta$, zatem X^* jest nieobciążonym estymatorem θ.

$$\begin{aligned} Var(X^*) &= Var(X) + \alpha^2 Var(Y) - 2\alpha Cov(X, Y-\mu) \\ &= Var(X) - \alpha(2Cov(X,Y) - \alpha Var(Y)). \end{aligned} \tag{3.17}$$

Zachodzi:

$$Var(X^*) < Var(X) \iff 0 < \alpha < 2Cov(X,Y)/Var(Y). \tag{3.18}$$

$f(\alpha) = Var(X^*)$ jest funkcją kwadratową, która osiąga minimum w punkcie α^* zerowania się pochodnej:

$$\alpha^* = \frac{Cov(X,Y)}{Var(Y)} \tag{3.19}$$

Podstawiając wzór (3.19) do wzoru (3.17) dostajemy minimalną wariancję:

$$f(\alpha^*) = Var(X) - \frac{Cov(X,Y)^2}{Var(X)Var(Y)}Var(X) = Var(X)(1-\rho^2), \tag{3.20}$$

gdzie $\rho = \frac{Cov(X,Y)}{\sqrt{Var(X)Var(Y)}}$ jest współczynnikiem korelacji. Redukcja wariancji wynosi $100\rho^2\%$. Im większa korelacja zmiennych X i Y tym bardziej można zredukować wariancję. Dlatego zmienną Y, o znanej wartości oczekiwanej, najlepiej wybrać taką, aby jej wartość oczekiwana była zbliżona do pewnego estymatora nieznanej wartości oczekiwanej zmiennej X.

3.3.1. Zastosowanie metody control variates

Niech ponownie nieznanym parametrem, którego wartość chcemy obliczyć będzie:

$$\theta = \int_0^1 e^{-x^2/2}dx$$

oraz niech $X = \hat{\theta}_n^{IS}$ będzie estymatorem importance sampling z przykładu z poprzedniego rozdziału:

$$\hat{\theta}_n^{IS} = \frac{1}{n}\sum_{i=1}^{n} e^{-T_i^2/2}\sqrt{2}arctan\left(\frac{1}{\sqrt{2}}\right)\left(1+\frac{T_i^2}{2}\right), \tag{3.21}$$

gdzie T_i są próbkami pochodzącymi z rozkładu o gęstości h, tak jak wcześniej.

Rozwinięcie w szereg Taylora funkcji $e^{-x^2/2}$ w pobliżu zera daje:

$$e^{-x^2/2} \approx 1 - x^2/2. \tag{3.22}$$

Definiujemy, więc:

$$\mu = \int_0^1 \left(1 - \frac{x^2}{2}\right)dx = \frac{5}{6} \tag{3.23}$$

i jako zmienną kontrolną używamy zmiennej Y, mającej taką samą gęstość h jaką użyto do estymowania parametru θ (we wzorze 3.21):

$$Y = \hat{\mu} = \frac{1}{n}\sum_{i=1}^{n}\left(1-\frac{T_i^2}{2}\right)\sqrt{2}arctan\left(\frac{1}{\sqrt{2}}\right)\left(1+\frac{T_i^2}{2}\right) = \frac{1}{n}\sum_{i=1}^{n}\psi_2(T_i). \tag{3.24}$$

Oczywiście $\mathbb{E}Y = \mu$, a pozytywną korelację pomiędzy zmiennymi X i Y można zapewnić używając tych samych T_i do generowania X i Y. Ponieważ zachodzi (3.22), wartości oczekiwane Y i X są zbliżone.

Dla $\alpha > 0$ formułujemy X^*, kontrolowaną wersję X, jako:

$$X^* = \hat{\theta}_n^{CV} = X - \alpha\left(Y - \frac{5}{6}\right) = \hat{\theta}_n^{IS} - \alpha\left(\hat{\mu} - \frac{5}{6}\right). \tag{3.25}$$

Powyższe równanie wyjaśnia, że estymator *control variate* to skorygowany estymator pierwotny. Na przykład jeżeli wartości zmiennej kontrolnej $\hat{\mu}$ przewyższają swoją znaną średnią, pozytywna korelacja sugeruje że wartości $\hat{\theta}_n^{IS}$ mogą także być za wysokie, więc estymacja jest korygowana w dół. Optymalny wybór α nie jest znany, gdyż wartość wyrażenia:

$$Cov\,(X,Y)\,/VarY = Cov(\psi_1(T_1),\psi_2(T_1))/Var\psi_2(T_1) \tag{3.26}$$

nie jest znana. Można ją jednak przybliżać używając wariancji i kowariancji próby opartej na estymacji.

Przedstawię obliczenia redukcji wariancji, jaką można uzyskać korzystając z metody *control variates*. Obliczenia wykonam dla $n = 100$ próbek losowanych z rozkładu jednostajnego. Doświadczenie powtórzę $N = 1000$ razy. Wariancję porównam z wariancją otrzymaną przy wykorzystaniu metody Monte Carlo.

```
# g to funkcja, której wartość całki na przedziale (0,1) będzie przybliżana
# w tablicy MC znajdują się będą obliczone metodą Monte Carlo przybliżone wartości całki
# w tablicy IS znajdują się będą obliczone metodą importance sampling przybliżone wartości całki
# importance sampling (wzór (3.15))
# w tablicy EY znajdować się będą wartości estymowanego parametru mi (wzór 3.24)
for(i in 1:1000){
  u <- runif(100, min = 0, max = 1)
  MC[i] <- mean(g(u))
  IS[i] <- mean(g(u) * sqrt(2) * atan(1 / sqrt(2)) * (1 + u^2 / 2))
  EY[i] <- mean((1 - (u^2 / 2)) * sqrt(2) * atan(1 / sqrt(2)) * (1 + u^2 / 2))
}
# obliczenie parametru a - wzór (3.19)
a <- cov(IS, EY) / var(EY)
# CV to tablica zawierająca wartości estymatorów obliczonych metodą control variates (wzór 3.25)
CV <- IS - a * (EY - (5 / 6))
# wariancja po wykorzystaniu metody Monte Carlo
var(MC)
[1] 2.054094e-06
# wariancja wartości obliczonych metodą importance sampling
var(IS)
[1] 1.117196e-07
# wariancja wartości obliczonych metodą control variates
var(CV)
[1] 4.278394e-10
# obliczenie redkcji wariancji
red <- (var(MC) - var(CV)) / var(MC)
red
[1] 0.9997917
```

Redukcja wariancji po zastosowaniu metody *control variates* wynosi około $99,98\%$ i jest największa spośród uzyskanych redukcji.

Podsumowanie

W pracy przedstawiłam wybrane algorytmy optymalizacji, estymacji i redukcji wariancji oraz ich zastosowania.

W rozdziale pierwszym omówiłam metodę Newtona i metodę złotego podziału – jako przykłady metod optymalizacji jednowymiarowej oraz metodę najszybszego wzrostu i wielowymiarową metodę Newtona – jako przykłady algorytmów optymalizacji dla funkcji wielowymiarowej. Dokonałam optymalizacji funkcji wiarogodności, wykorzystując do tego celu wielowymiarową metodę Newtona. Wyniki, które otrzymałam były bardzo bliskie spodziewanemu. Pokazałam także wykorzystanie metod optymalizacji przy *curve fitting*. W tym celu przeanalizowałam dane `apartments` pochodzące z pakietu `PBImisc` i próbowałam znaleźć najlepszą kwadratową zależność pomiędzy ceną za m^2 mieszkań a ich całkowitą powierzchnią. Prześledziłam też zmianę uzyskiwanych parametrów funkcji kwadratowej przy zwiększaniu maksymalnej liczby iteracji.

W rozdziale drugim omówiłam metodę momentów, metodę kwantyli i metodę największej wiarogodności, jako przykłady algorytmów estymacji punktowej. Zaprezentowałam przykład zastosowania metody momentów, wykorzystując ponownie dane `apartments`. Przeanalizowałam ceny za m^2 mieszkań i założyłam, że pochodzą one z rozkładu gamma. Przy użyciu metody momentów znalazłam parametry tego rozkładu, które najlepiej pasowały do danych. Wykres gęstości uzyskanego rozkładu gamma naniosłam na histogram częstości, obrazujący dane empiryczne. Otrzymana gęstość powstała w wyniku estymacji była zbliżona do kształtu histogramu. W rozdziale tym porównałam także parametry rozkładu Weibulla powstałe w wyniku standardowej metody kwantyli, gdy równań jest tyle samo co estymowanych parametrów, oraz zmodyfikowanej metody kwantyli, gdy równań jest więcej niż parametrów. W drugim przypadku, w celu optymalizacji funkcji straty użytej w zagadnieniu nadokreślonych układów równań liniowych, wykorzystałam wielowymiarową metodę Newtona. Po zobrazowaniu wyników wykresami zauważyłam, że wartości parametrów rozkładu Weibulla obliczone tymi dwoma metodami różniły się nieznacznie. Na końcu omawiania estymacji punktowej porównałam estymatory parametru λ rozkładu wykładniczego, powstałe w wyniku metody momentów, metody największej wiarogodności i metody kwantyli, a wyniki przedstawiłam na wykresach, z których odczytałam, że estymowane parametry metodą największej wiarogodności były najczęściej bliskie prawdziwej wartości parametru oraz charakteryzowały się najmniejszą rozbieżnością wyników. W rozdziale drugim pokazałam także przykład estymacji przedziałowej, na podstawie przedziałów ufności. Wykonałam symulację przedziałów ufności dla średniej w przypadku, gdy wariancja jest znana, używając do tego celu próbek pochodzących z rozkładu normalnego o parametrach $(4, 2)$. Uzyskałam wyniki zgodne z oczekiwaniami.

W rozdziale trzecim omówiłam trzy metody redukcji wariancji: *antithetic sampling*, *importance sampling* i *control variates*. Przedstawiłam działanie każdej z metod na jednym przykładzie, dotyczącym obliczenia przybliżonej wartości całki. Redukcja wariancji otrzymana metodą *control variates* była największa.

Dodatek A

Implementacja metody Newtona

W implementacji metody Newtona funkcja `f3` pobiera argument x i zwraca wektor $(f(x), f'(x), f''(x))$. W rzeczywistości pochodne funkcji f często obliczane są numerycznie (dodatek D). Ponieważ szukany jest taki punkt x^*, że $f'(x^*) = 0$ jako warunek stopu przyjęłam $|f'(x(n))| \leqslant \epsilon$.

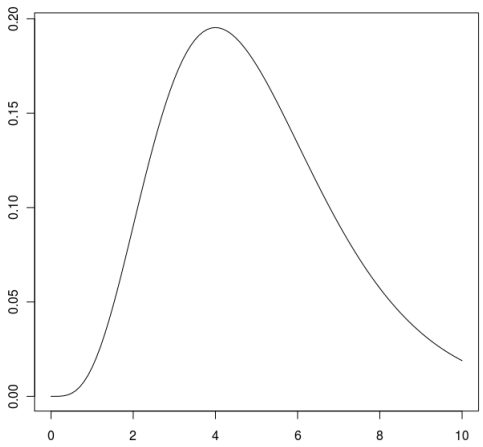
```
newton <- function(f3, x0, tol = 1e-20, n.max = 100){
  x <- x0
  f3.x <- f3(x)
  n <- 0
  while ((abs(f3.x[2]) > tol) & (n < n.max)){
    x <- x - f3.x[2]/f3.x[3]
    f3.x <- f3(x)
    n <- n + 1
  }
  if (n == n.max){
    cat('Metoda Newtona nie jest zbieżna\n')
  }
  else {
    return(x)
  }
}
```

Metodę Newtona zastosowałam do funkcji będącej gęstością rozkładu gamma(5,1).

```
gamma.5.1 <- function(x){
  if (x < 0) return(c(0, 0, 0))
  if (x == 0) return(c(0, 0, NaN))
  y<-exp(-x)
  return(c(y*x^4/24, -y*x^3*(x-4)/24, y*x^2*(x^2-8*x+12)/24))
}
```

```
newton(gamma.5.1, 3)
[1] 4
newton(gamma.5.1, 1.3)
[1] 3.102875e-07
```

Powyższy przykład pokazuje, że metoda Newtona może zbiegać zarówno do punktu, w którym istnieje maksimum jak i do punktu, w którym nie ma ekstremum, ale pochodna w tym punkcie jest dostatecznie mała.



Rysunek A.1: Funkcja $f(x) = \frac{1}{24}x^4e^{-x}$ - gęstość rozkładu gamma, do której zastosowałam metodę Newtona.

Dodatek B

Implementacja metody złotego podziału

W implementacji metody złotego podziału maksymalizowana jest funkcja jednej zmiennej f_{gs}. Punkty x_l, x_m, x_r są wybrane tak, że $x_l < x_m < x_r$ oraz $f_{gs}(x_l) \leqslant f_{gs}(x_m)$, $f_{gs}(x_r) \leqslant f_{gs}(x_m)$. Algorytm kończy swoją pracę, gdy $x_r - x_l \leqslant tol$ i zwraca x_m.

```
gsection <- function(fgs, x.l, x.r, x.m, tol = 1e-9){
  gr1 <- 1 + (1 + sqrt(5)) / 2
  f.l <- fgs(x.l)
  f.r <- fgs(x.r)
  f.m <- fgs(x.m)
  while ((x.r - x.l) > tol){
    if ((x.r - x.m) > (x.m - x.l)){
      y <- x.m + (x.r - x.m) / gr1
      f.y <- fgs(y)
      if (f.y >= f.m){
        x.l <- x.m
        f.l <- f.m
        x.m <- y
        f.m <- f.y
      }
      else{
        x.r <- y
        f.r <- f.y
      }
    }
    else{
      y <- x.m - (x.m - x.l) / gr1
      f.y <- fgs(y)
      if (f.y >= f.m){
        x.r <- x.m
        f.r <- f.m
        x.m <- y
        f.m <- f.y
      }
      else{
        x.l <- y
        f.l <- f.y
      }
    }
  }
  return(x.m)
```

```
37 }
```

Do powyższego algorytmu zastosowałam (tak jak w przypadku algorytmu Newtona w dodatku A) gęstość rozkładu gamma(5,1).

```
gamma.5.1 <- function(x){
  if (x <= 0) return(0)
  y <- exp(-x)
  return(y * x^4 / 24)
}
```

```
gsection(gamma.5.1, 0, 8, 5)
[1] 4
gsection(gamma.5.1, 3, 8, 100)
[1] 4
```

Algorytm jest zbieżny do maksimum funkcji jeżeli wybrane punkty początkowe x_l, x_m, x_r spełniają warunki przedstawione na wstępie.

Jeżeli wyszukiwanie zostanie rozpoczęte od punktu x_m wybranego tak, że stosunek $(x_r - x_m)/(x_m - x_l) = \rho$ lub $1/\rho$, to w każdej iteracji długość przedziału zmniejsza się $\rho/(1+\rho)$ razy i w końcu zbiega do zera.

Dodatek C

Implementacja metody najszybszego wzrostu

W implementacji metody najszybszego wzrostu funkcja `ascent` przyjmuje za argumenty funkcje f oraz ∇f. W algorytmie wykorzystywana jest też funkcja `line.search`, która pobiera: $f, \mathbf{x}(n), \nabla f \mathbf{x}(n)$ i zwraca $\mathbf{x}(n) + \alpha_m \nabla f(\mathbf{x}(n))$, gdzie $\alpha_m \geqslant 0$ maksymalizuje wartość funkcji $g(\alpha) = f(\mathbf{x}(n) + \alpha \nabla f(\mathbf{x}(n)))$. W celu obliczenia tego maksimum użyty został algorytm złotego podziału. Algorytm złotego podziału rozpoczyna swoje działanie od wyboru punktów: $\alpha_l < \alpha_m < \alpha_r$ takich, że $g(\alpha_m) \geqslant g(\alpha_l)$ oraz $g(\alpha_m) \geqslant g(\alpha_r)$. Przyjmowane jest $\alpha_l = 0$. Jeżeli $\|\nabla f(\mathbf{x}(n))\| > 0$ to $g'(0) > 0$ i istnieje $\epsilon > 0$ taki, że $g(\epsilon) > g(0)$, więc można przyjąć $\alpha_m = \epsilon$. W praktyce jednak, jeżeli $g'(0)$ jest bardzo małe to $g(\epsilon) - g(0) \approx g'(0)\epsilon$ jest bardzo małe i trudno jest numerycznie oddzielić $g(0)$ od $g(\epsilon)$. Nie ma także gwarancji, że odpowiednie α_r istnieje (jest tak np. wtedy gdy funkcja jest rosnąca na całym przedziale $[0, \infty)$). W tym celu przyjęto więc α_{max}, które oznacza największy rozmiar kroku. Jeżeli nie da się znaleźć takiego α_r, że $g(\alpha_r) \leqslant g(\alpha_m)$ to zwracane jest α_{max}.

Jeśli funkcja f jest ograniczona to ciąg $\{f(\mathbf{x}(n))\}_{n=1}^{\infty}$ musi być zbieżny. Obserwacja ta sugeruje, aby za warunek stopu przyjąć $f(\mathbf{x}(n)) - f(\mathbf{x}(n-1)) \leqslant \epsilon$, dla pewnego ustalonego ϵ. Zbieżność ciągu $\{f(\mathbf{x}(n))\}_{n=1}^{\infty}$ nie implikuje zbieżności ciągu $\{\mathbf{x}(n)\}_{n=0}^{\infty}$. Można pokazać jednak, że jeżeli f jest ciągła, a ∇f jest wektorem funkcji jednostajnie ciągłych (w rozpatrywanym obszarze) to ciąg $\{\mathbf{x}(n)\}_{n=0}^{\infty}$ jest zbieżny.

```
ascent <- function(f, grad, x0, tol = 1e-9, n.max = 100000){
  x <- x0
  x.old <- x
  x <- line.search(f, x, grad(x))
  n <- 1
  while ((f(x) - f(x.old) > tol) && (n < n.max)){
    x.old <- x
    x <- line.search(f, x, grad(x))
    n <- n + 1
  }
  return(x)
}
```

```
line.search <- function(f, x, y, tol = 1e-9, a.max = 2^5){
  if (sum(abs(y)) == 0) return(x)
  g <- function(a) return(f(x + a * y))
  a.l <- 0
  g.l <- g(a.l)
```

```
  a.m <- 1
  g.m <- g(a.m)
  while ((g.m < g.l) & (a.m > tol)){
    a.m <- a.m/2
    g.m <- g(a.m)
  }
  if ((a.m <= tol) & (g.m < g.l)) return(x)
  a.r <- 2*a.m
  g.r <- g(a.r)
  while ((g.m < g.r) & (a.r < a.max)){
    a.m <- a.r
    g.m <- g.r
    a.r <- 2*a.m
    g.r <- g(a.r)
  }
  if ((a.r >= a.max) & (g.m < g.r)) return(x - a.max * y)
  a <- gsection(g, a.l, a.r, a.m)
  return(x + a * y)
}
```

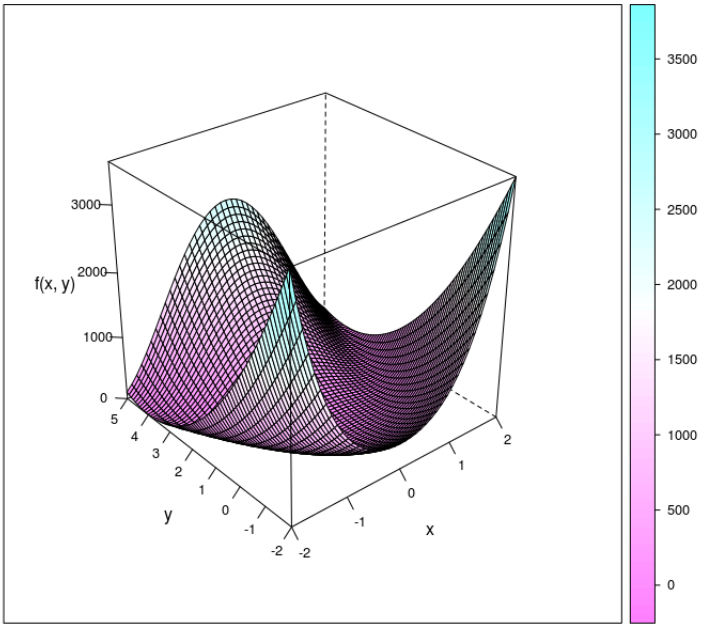
Metodę najszybszego wzrostu zastosowałam do funkcji Rosenbrocka. Funkcja Rosenbrocka używana jest często do testowania algorytmów optymalizacji. Przyjmuje ona swoje minimum globalne w punkcie $(-1, 1)$ i wyraża się wzorem:

$$f(x, y) = (1 - x)^2 + 100(y - x^2)^2. \tag{C.1}$$

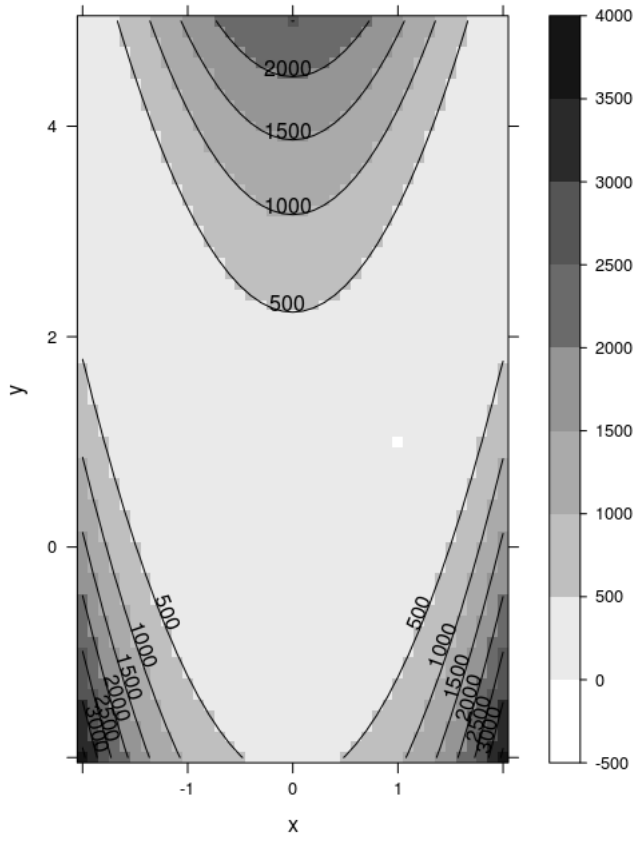
```
Rosenbrock <- function(x){
  g <- (1 - x[1])^2 + 100 * (x[2] - x[1]^2)^2
  g1 <- -2 * (1 - x[1]) - 400 * (x[2] - x[1]^2) * x[1]
  g2 <- 200 * (x[2] - x[1]^2)
  g11 <- 2 - 400*x[2] + 1200 * x[1]^2
  g12 <- -400 * x[1]
  g22 <- 200
  return(list(g, grad=c(g1, g2), matrix(c(g11, g12, g12, g22), 2, 2)))
}
```

```
x <- seq(-2, 2, .1)
y <- seq(-2,5, .1)
xyz <- data.frame(matrix(0, length(x)*length(y), 3))
names(xyz) <- c('x', 'y', 'z')
n <- 0
for (i in 1:length(x)){
  for (j in 1:length(y)){
    n <- n + 1
    xyz[n,] <- c(x[i], y[j], Rosenbrock(c(x[i], y[j]))[[1]])
  }
}
library(lattice)
print(wireframe(z ~ x*y, data = xyz, scales = list(arrows = FALSE),
zlab = 'f(x, y)', drape = T))
```

```
print (contourplot(z~x*y, data = xyz, region=TRUE,
aspect="iso",col.regions=gray((12:1)/12)))
```



Rysunek C.1: Przestrzenny wykres funkcji Rosenbrocka.



Rysunek C.2: Poziomicowy wykres funkcji Rosenbrocka.

Poniżej zastosowałam algorytm najszybszego wzrostu do funkcji Rosenbrocka. Ponieważ algorytm najszybszego wzrostu wyszukuje maksimum, a funkcja Rosenbrocka ma jedno minimum, przy wyszukiwaniu ekstremum tej funkcji należy ją pomnożyć przez -1.

```
g <- function(x){
  return(-((1 - x[1])^2 + 100 * (x[2] - x[1]^2)^2))
}
grad <- function(x){
  g1 <- -2 * (1 - x[1]) - 400 * (x[2] - x[1]^2) * x[1]
  g2 <- 200 * (x[2] - x[1]^2)
  return(c(-g1, -g2))
}
ascent(g, grad, c(0,3))
[1] 0.9993649 0.9987301
```

Algorytm najszybszego wzrostu wskazuje przybliżone współrzędne ekstremum funkcji Rosenbrocka.

Dodatek D

Implementacja wielowymiarowej metody Newtona

W implementacji wielowymiarowej metody Newtona założyłam, że funkcja `f3` pobiera argument x i zwraca wektor $(f(x), \nabla f(x), \mathbf{H}(x))$. Jako warunek stopu przyjęłam: $\|\nabla f(\mathbf{x}(n))\|_\infty \leqslant \epsilon$.

```
newton <- function(f3, x0, tol = 1e-9, n.max = 100000) {
  x <- x0
  f3.x <- f3(x)
  n <- 0
  while ((max(abs(f3.x[[2]])) > tol) & (n < n.max)) {
    x <- x - solve(f3.x[[3]], f3.x[[2]])
    f3.x <- f3(x)
    n <- n + 1
  }
  if (n == n.max){
    cat('Metoda Newtona nie jest zbieżna\n')
  }
  else {
    return(x)
  }
}
```

Po zastosowaniu funkcji Rosenbrocka do metody Newtona dostajemy współrzędne minimum funkcji Rosenbrocka:

```
newton(Rosenbrock,c(0,3))
[1] 1 1
```

Częstą trudnością w metodzie Newtona jest obliczenie gradientu i hesjanu. W metodzie najszybszego wzrostu obliczany jest tylko gradient, ale czasami też może być to kłopotliwe. Dla wielomianów i prostych funkcji obliczenia te są stosunkowo łatwe, ale nie zawsze tak jest. Istnieje wiele sytuacji, w których f jest znaną funkcją, a ∇f nie jest. Jest tak na przykład wtedy, gdy f jest rezultatem numerycznych procedur albo aproksymacji powstałej z symulacji. W tej sytuacji można poradzić sobie na dwa sposoby. Pierwszy z nich polega na wykorzytsaniu metod, które nie potrzebują gradientu i pochodnych. W jednym wymiarze można skorzystać z metody złotego podziału, zaś w przypadku wielowymiarowym istnieje algorytm Neldera-Meada, nie używający pochodnych. Druga metoda polega na numerycznym oblczeniu gradientu i Hesjanu.

Niech $f : \mathbb{R}^d \to \mathbb{R}$. Ponieważ:

$$\frac{\partial f(\mathbf{x})}{\partial x_i} = \lim_{\epsilon \to 0} \frac{f(\mathbf{x} + \epsilon \mathbf{e_i})}{\epsilon}, \tag{D.1}$$

to dla małego ϵ można zapisać:

$$\frac{\partial f(\mathbf{x})}{\partial x_i} \approx \frac{f(\mathbf{x} + \epsilon \mathbf{e}_i) - f(\mathbf{x})}{\epsilon}. \tag{D.2}$$

Podobnie można zapisać dla $i \neq j$

$$\frac{\partial^2 f(\mathbf{x})}{\partial x_i \partial x_j} \approx \frac{f(\mathbf{x} + \epsilon \mathbf{e}_i + \epsilon \mathbf{e}_j) - f(\mathbf{x} + \epsilon \mathbf{e}_i) - f(\mathbf{x} + \epsilon \mathbf{e}_j) + f(\mathbf{x})}{\epsilon^2}. \tag{D.3}$$

oraz:

$$\frac{\partial^2 f(\mathbf{x})}{\partial x_i^2} \approx \frac{f(\mathbf{x} + 2\epsilon \mathbf{e}_i) - 2f(\mathbf{x} + \epsilon \mathbf{e}_i) + f(\mathbf{x})}{\epsilon^2}. \tag{D.4}$$

Poniżej zastosowałam algorytm Newtona do funkcji Rosenbrocka korzystając z gradientu i hesjanu obliczonego numerycznie.

```
e1 = c(1, 0)
e2 = c(0, 1)
eps = 10^-7
g <- function(x){
  return((1 - x[1])^2 + 100 * (x[2] - x[1]^2)^2)
}
Rosenbrock <- function(x){
  g  <- g(x)
  g1 <- (g(x + eps * e1) - g(x)) / eps
  g2 <- (g(x + eps * e2) - g(x)) / eps
  g11 <- (g(x + 2 * eps * e1) - 2 * g(x + eps * e1) + g(x)) / eps^2
  g12 <-(g(x + eps * e1 + eps * e2) - g(x + eps * e1) - g(x + eps * e2) + g(x))
       / eps^2
  g22 <- (g(x + 2 * eps * e2) - 2 * g(x + eps * e2) + g(x)) / eps^2
  return(list(g, c(g1, g2), matrix(c(g11, g12, g12, g22), 2, 2)))
}
newton(Rosenbrock, c(0, 3))
[1] 0.9999700 0.9999399
```

Algorytm Newtona wskazał punkt będący przybliżeniem minimum funkcji Rosenbrocka.

Bibliografia

[1] Przemysław Biecek, *Przewodnik po pakiecie R*, 2008.

[2] Owen Jones, Robert Maillardet, Andrew Robinson, *Introduction to Scientific Programming and Simulation Using R*, 2009.

[3] David Kincaid, Ward Cheney *Analiza numeryczna*, 2006.

[4] Wojciech Kordecki, *Rachunek prawdopodobieństwa i statystyka matematyczna*, 2010.

[5] Wojciech Niemiro, *Statystyka*, 2011.

[6] Wojciech Niemiro, *Symulacje stochastyczne i metody Monte Carlo*, 2011.

[7] Rafał Czyż, Leszek Gasiński, Marta Kosek, Jerzy Szczepański, Halszka Tutaj-Gasińska, *Ekstremum funkcji jednej zmiennej,*
http://wazniak.mimuw.edu.pl/index.php?title=Analiza_matematyczna_1/Wykład_9:_Pochodna_funkcji_jednej_zmiennej

[8] Piotr Krzyżanowski, Leszek Plaskota, *Nadokreślone układy równań liniowych,*
http://osilek.mimuw.edu.pl/index.php?title=MN12

[9] Przemysław Biecek, *Pakiet danych PBImisc,*
http://cran.r-project.org/web/packages/PBImisc/