Rozdział 4: Metody znajdowania minimum sumy kwadratów reszt i maksimum funkcji wiarygodności

W modelowaniu ekonometrycznym stosuje się wiele metod optymalizacyjnych do wyznaczenia oszacowań parametrów. W przypadku NMNK i MNW najbardziej popularne są metody gradientowe oraz metoda przeszukiwania po kratownicy (jako najprostsza metoda). Nieco rzadziej niż metody gradientowe wykorzystywane są metody nie wymagające wyliczania pochodnych funkcji celu, na przykład metoda symulowanego wyżarzania lub metoda Neldera i Meada (1965). Poniżej przedstawiono najpopularniejsze metody stosowane w ekonometrii.

4.1 Metoda przeszukiwania po kratownicy

Metoda przeszukiwania po kratownicy (grid search) służy do wyznaczania takich wartości parametrów, dla których funkcja celu osiąga poszukiwane optimum. Zazwyczaj wykorzystywana jest ona w przypadku kiedy liczba parametrów do oszacowania nie jest duża lub kiedy nie jest wymagana duża precyzja oszacowań. Metodę tą stosuje się często jako element bardziej złożonej procedury optymalizacyjnej, na przykład w celu wyznaczenia dobrych wartości startowych parametrów dla algorytmu gradientowego.

Metoda przeszukiwania po kratownicy składa się z kilku kroków. Wyznacza się przedziały dopuszczalnych wartości dla każdego szacowanego parametru. Następnie na każdym przedziale – między najmniejszą i największą wartością tego przedziału – ustawia się równomiernie pewną liczbę P punktów. Zwykle P równe jest 100 lub 1000. Wartość każdego z tych punktów może stanowić przybliżenie oszacowania odpowiadającego mu parametru. Ponieważ parametrów w modelu jest k, a punktów w każdym przedziale jest P, to liczba wszystkich kombinacji wartości dla wszystkich analizowanych parametrów równa jest P^k, na przykład $1000^3 = 1000000000$. Oznacza to, że metoda sprawdzania wszystkich kombinacji wartości parametrów jest bardzo czasochłonna przy dużej liczbie parametrów i przy dużej liczbie sprawdzanych wartości dla pojedynczego parametru.

Przykład 4.1

W tym przykładzie porównujemy oszacowanie analityczne MNK parametrów modelu regresji liniowej z oszacowaniami MNK uzyskanymi metodą przeszukiwania po kratownicy. Mimo sprawdzenia miliona różnych wartości parametrów przy pomocy metody numerycznej dokładniejsza okazuje się metoda analityczna. W pierwszym kroku

losujemy dane do modelu.

```
X =[ones(100,1) randn(100,1)];  % stała i zmienna generowana z N(0,1)
e = randn(100,1);               % wektor składników losowych z N(0,1)
alfa = [1; 2];                  % prawdziwe parametry modelu
y = X*alfa+e;                   % wartości y wygenerowane z modelu
alfa_MNK = X\y                  % oszacowania MNK
```

Oszacowanie uzyskane w wektorze `alfa_MNK` wykorzystuje analityczny wzór $\widehat{\boldsymbol{\alpha}} = (\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{X}'\boldsymbol{y}$. Dla metody numerycznej wybieramy przedziały, w których będziemy szukali optymalnych wartości parametrów.

```
beta_min = [0; 0];              % dolne ograniczenie wartości parametrów
beta_max = [5; 5];              % górne ograniczenie wartości parametrów
```

Sprawdzamy 1001 różnych wartości dla każdego z dwóch parametrów i wybieramy optymalne wartości, które minimalizują funkcję celu, czyli sumę kwadratów reszt modelu.

```
Q_min = -1;                     % wartość startowa funkcji celu
beta_opt = [0; 0];              % wartości startowe parametrów
for i=0:1000                    % sprawdzamy po 1000 wartości...
    for j=0:1000                % dla każdego parametru
        delta = [i/1000; j/1000]; % nowe wybrane wartości...
        beta = beta_min + (beta_max - beta_min).*delta; % wstawione do beta
        reszty = y-X*beta;      % liczymy reszty dla nowych parametrów
        Q = reszty'*reszty;     % liczymy sumę kwadratów reszt
        if (Q_min < 0) || (Q_min > Q)   % czy nowy wynik najlepszy?
            Q_min = Q;                  % tak, najlepszy
            beta_opt = beta;            % nowe optymalne parametry
        end                             % ...ale szukamy dalej
    end
end

table(alfa_MNK, beta_opt)       % porównanie wyniku analitycznego i numerycznego
```

Ostatnie polecenie pozwala zaprezentować oba oszacowania i porównać ich wartości.

4.2 Metoda najszybszego wzrostu

Metoda najszybszego wzrostu (steepest ascent), to najprostsza metoda gradientowa służąca do znajdowania maksimum funkcji celu. Polega ona na wyznaczeniu kierunku w którym funkcja celu najszybciej rośnie, podążaniu w tym kierunku i przybliżaniu się w kolejnych krokach do (lokalnego) maksimum tej funkcji (Hamilton, 1994, str. 133-137; Qunadt, 1983, str. 707). W analogiczny sposób działa metoda najszybszego spadku (steepest descent) i jej nie będziemy oddzielnie omawiać.

Niech $\boldsymbol{\theta}_0$ oznacza wartość startową wektora parametrów $\boldsymbol{\theta}$. Poszukujemy takich wartości parametrów wektora $\boldsymbol{\theta}$, które maksymalizują funkcję celu $L(\boldsymbol{\theta})$. Niech d oznacza długość

kroku mierzoną jako $d = (\boldsymbol{\theta}_1 - \boldsymbol{\theta}_0)'(\boldsymbol{\theta}_1 - \boldsymbol{\theta}_0)$. Możemy poszukać maksimum następującego Lagrange'anu:

$$J(\boldsymbol{\theta}_1) = L(\boldsymbol{\theta}_1) + \lambda(d - (\boldsymbol{\theta}_1 - \boldsymbol{\theta}_0)'(\boldsymbol{\theta}_1 - \boldsymbol{\theta}_0)). \tag{4.1}$$

Z warunku pierwszego rzędu wynika

$$\frac{\partial L(\boldsymbol{\theta}_1)}{\partial \boldsymbol{\theta}}\Big|_{\boldsymbol{\theta}=\boldsymbol{\theta}_1} - 2\lambda(\boldsymbol{\theta}_1 - \boldsymbol{\theta}_0) = \mathbf{0}. \tag{4.2}$$

Niech $g(\boldsymbol{\theta})$ oznacza gradient funkcji celu $L(\boldsymbol{\theta})$, czyli wektor pochodnych cząstkowych funkcji $L(\boldsymbol{\theta})$ po parametrach, $\frac{\partial L(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}$. Wtedy możemy zapisać przesunięcie z punktu $\boldsymbol{\theta}_0$ do $\boldsymbol{\theta}_1$ jako:

$$\boldsymbol{\theta}_1 - \boldsymbol{\theta}_0 = g(\boldsymbol{\theta}_1) \cdot \frac{1}{2\lambda}. \tag{4.3}$$

Ze względów praktycznych przyjmuje się, że dla dostatecznie małego kroku $d \to 0$, gradienty w obu punktach $\boldsymbol{\theta}_0$ i $\boldsymbol{\theta}_1$ są w przybliżeniu równe (czyli, że pochodne cząstkowe na odcinku między $\boldsymbol{\theta}_0$ i $\boldsymbol{\theta}_1$ są mniej więcej stałe) i możemy zapisać $\boldsymbol{\theta}_1 - \boldsymbol{\theta}_0 \approx g(\boldsymbol{\theta}_0) \cdot s$, gdzie $s = 1/(2\lambda)$. W kolejnych krokach definiujemy

$$\boldsymbol{\theta}_{m+1} = \boldsymbol{\theta}_m + g(\boldsymbol{\theta}_m) \cdot s. \tag{4.4}$$

Można też wyprowadzić optymalną długość kroku s i wtedy krok $m+1$ algorytmu zdefiniowany jest następująco

$$\boldsymbol{\theta}_{m+1} = \boldsymbol{\theta}_m + \frac{g(\boldsymbol{\theta}_m)'g(\boldsymbol{\theta}_m)}{g(\boldsymbol{\theta}_m)'H(\boldsymbol{\theta}_m)g(\boldsymbol{\theta}_m)} g(\boldsymbol{\theta}_m), \tag{4.5}$$

gdzie $H(\boldsymbol{\theta}_m)$ oznacza hesjan (macierz drugich pochodnych cząstkowych) funkcji celu $L(\boldsymbol{\theta}_m)$. Jednak stosowanie tego wzoru jest kłopotliwe ze względu na potrzebę liczenia drugich pochodnych funkcji $L(\boldsymbol{\theta})$ w każdym kroku. Ponadto jeśli hesjan H nie jest dodatnio określony, to algorytm może wskazywać przybliżenia maksimum funkcji celu w złym kierunku. Dlatego w praktyce ustala się sztywną wartość parametru s i utrzymuje się ją stałą w kolejnych krokach.

Przykład 4.1

W tym przykładzie porównujemy oszacowanie analityczne MNK parametrów modelu regresji liniowej z oszacowaniami MNK uzyskanymi metodą najszybszego spadku. W pierwszym kroku losujemy dane do modelu.

```
X =[ones(100,1) randn(100,1)];  % stała i zmienne generowana z N(0,1)
e = randn(100,1);               % wektor składników losowych z N(0,1)
alfa = [1; 2];                  % prawdziwe parametry modelu
y = X*alfa+e;                   % wartości y wygenerowane z modelu
```

```
alfa_MNK = X\y                    % oszacowania MNK
```

Oszacowanie uzyskane w wektorze `alfa_MNK` wykorzystuje analityczny wzór $\widehat{\boldsymbol{\alpha}} = (\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{X}'\boldsymbol{y}$. Dla metody numerycznej wybieramy początkową wartość parametrów oraz metodę optymalizacji funkcji celu, czyli sumy kwadratów reszt modelu.

```
beta_opt = [0; 0];                % wartości startowe parametrów
options = optimoptions('fminunc',...
    'Display','iter',...
    'Algorithm','quasi-newton',...
    'HessUpdate','steepdesc',...
    'MaxFunctionEvaluations',600);  % wybór opcji dla funkcji fminunc
```

Następnie uruchamiamy procedurę poszukiwania minimum funkcji celu metodą gradientową. Zastosowanie funkcji `fminunc` wymaga posiadania zainstalowanej biblioteki Optimization Toolbox w programie MATLAB.

```
fun = @(beta) (y-X*beta)'*(y-X*beta); % funkcja licząca sumy kwadratów reszt
[beta_opt,fval,eflag,output] = fminunc(fun,beta_opt,options); % szuka beta_opt

table(alfa_MNK, beta_opt)         % porównanie wyniku analitycznego i numerycznego
```

Ostatnie polecenie pozwala zaprezentować oba oszacowania `alfa_MNK` i `beta_opt` oraz porównać ich wartości.

4.3 Metoda Newtona

Jeśli da się policzyć drugie pochodne funkcji $L(\boldsymbol{\theta})$ i funkcja ta jest wypukła na całej przestrzeni parametrów, to znaczy hesjan $H(\boldsymbol{\theta})$ jest macierzą dodatnio określoną, to metoda Newtona zwana także metodą Newtona-Raphsona (algorytm Newtona-Raphsona) pozwala z reguły odnaleźć lokalne maksimum funkcji celu w mniejszej liczbie kroków niż metoda najszybszego wzrostu (Amemiya, 1983, str. 342-345; Hamilton, 1994, str. 138-139; Quandt, 1983, str. 717).

Niech $H(\boldsymbol{\theta}_0) = -\frac{\partial^2 L(\boldsymbol{\theta})}{\partial\boldsymbol{\theta}\partial\boldsymbol{\theta}'}\Big|_{\boldsymbol{\theta}=\boldsymbol{\theta}_0}$. Możemy przybliżać funkcję celu przy pomocy szeregu Taylora w następujący sposób

$$L(\boldsymbol{\theta}) \approx L(\boldsymbol{\theta}_0) + g(\boldsymbol{\theta}_0)'(\boldsymbol{\theta}_1 - \boldsymbol{\theta}_0) - \frac{1}{2}(\boldsymbol{\theta}_1 - \boldsymbol{\theta}_0)'H(\boldsymbol{\theta}_0)(\boldsymbol{\theta}_1 - \boldsymbol{\theta}_0). \qquad (4.6)$$

Z warunku pierwszego rzędu wynika przybliżona równość

$$g(\boldsymbol{\theta}_0) - H(\boldsymbol{\theta}_0)(\boldsymbol{\theta}_1 - \boldsymbol{\theta}_0) \approx \mathbf{0},$$

a następnie

$$g(\boldsymbol{\theta}_0) \approx H(\boldsymbol{\theta}_0)(\boldsymbol{\theta}_1 - \boldsymbol{\theta}_0) \qquad \text{oraz} \qquad \boldsymbol{\theta}_1 - \boldsymbol{\theta}_0 \approx H(\boldsymbol{\theta}_0)^{-1} g(\boldsymbol{\theta}_0)\,.$$

Krok $m+1$ algorytmu jest zdefiniowany jako

$$\boldsymbol{\theta}_{m+1} - \boldsymbol{\theta}_m = H(\boldsymbol{\theta_m})^{-1} g(\boldsymbol{\theta}_m)\,. \tag{4.7}$$

Popularna modyfikacja algorytmu polega na tym, że optymalizowana jest długość kroku w metodzie Newtona. Zdefiniujmy wektor $\boldsymbol{d}_m = H(\boldsymbol{\theta_m})^{-1} g(\boldsymbol{\theta}_m)$. Krok $m+1$ algorytmu jest zdefiniowany jako

$$\boldsymbol{\theta}_{m+1} = \boldsymbol{\theta}_m + \alpha_m \cdot \boldsymbol{d}_m, \tag{4.8}$$

gdzie parametr α_m jest wybierany w taki sposób by optymalizować (w tym przypadku maksymalizować) funkcję $L(\boldsymbol{\theta}_m + \alpha_m \cdot \boldsymbol{d}_m)$. Ta jednowymiarowa optymalizacja może wykorzystywać dowolną numeryczną metodę, np. metodę złotego podziału (por. także Berndt, Hall, Hall i Hausman, 1974).

4.4 Metody quasi-Newtona

W metodach quasi-Newtona hessian $H(\boldsymbol{\theta_m})$ nie jest obliczany w każdym kroku m, a jedynie wykorzystywana jest jego odpowiednia aproksymacja B_m (Quandt, 1983, str. 722-724; Hamilton, 1994, str. 139-142). Dzięki temu, że łatwiej lub szybciej jest policzyć B_m^{-1} niż $H(\boldsymbol{\theta_m})^{-1}$, to cały algorytm powinien pozwolić odnaleźć optimum funkcji szybciej niż oryginalny algorytm Newtona. Krok $m+1$ algorytmu jest zdefiniowany jako

$$\Delta\boldsymbol{\theta}_{m+1} = \boldsymbol{\theta}_{m+1} - \boldsymbol{\theta}_m = \alpha_m B_m^{-1} g(\boldsymbol{\theta}_m)\,, \tag{4.9}$$

gdzie α_m jest odpowiednio dobranym parametrem regulującym długość kroku.

Niech $\boldsymbol{\delta}_m = g(\boldsymbol{\theta}_m) - g(\boldsymbol{\theta}_{m-1})$. Wtedy zgodnie z metodą BFGS (Broyden-Fletcher-Goldfarb-Shanno) macierze B_{m+1} i B_{m+1}^{-1} są wyliczane według wzorów (np. Fletcher, 1987, str. 55-57)

$$B_{m+1} = B_m + \frac{\Delta\boldsymbol{\theta}_m \boldsymbol{\delta}'_m}{\boldsymbol{\delta}'_m \Delta\boldsymbol{\theta}_m} - \frac{B_m \Delta\boldsymbol{\theta}_m (B_m \Delta\boldsymbol{\theta}_m)'}{\Delta\boldsymbol{\theta}'_m B_m \Delta\boldsymbol{\theta}_m} \tag{4.10}$$

oraz

$$B_{m+1}^{-1} = \left(\boldsymbol{I} - \frac{\Delta\boldsymbol{\theta}_m \boldsymbol{\delta}'_m}{\boldsymbol{\delta}'_m \Delta\boldsymbol{\theta}_m}\right) B_m^{-1} \left(\boldsymbol{I} - \frac{\Delta\boldsymbol{\theta}_m \boldsymbol{\delta}'_m}{\boldsymbol{\delta}'_m \Delta\boldsymbol{\theta}_m}\right) + \frac{\Delta\boldsymbol{\theta}_m \Delta\boldsymbol{\theta}'_m}{\boldsymbol{\delta}'_m \Delta\boldsymbol{\theta}_m}. \tag{4.11}$$

Natomiast zgodnie z metodą DFP (Davidon-Fletcher-Powell; Davidon, 1959; Fletcher i Powell, 1963) macierze B_{m+1} i B_{m+1}^{-1} są wyliczane według wzorów

$$B_{m+1} = \left(\boldsymbol{I} - \frac{\boldsymbol{\delta}_m \Delta\boldsymbol{\theta}_m'}{\boldsymbol{\delta}_m' \Delta\boldsymbol{\theta}_m}\right) B_m \left(\boldsymbol{I} - \frac{\boldsymbol{\delta}_m \Delta\boldsymbol{\theta}_m'}{\boldsymbol{\delta}_m' \Delta\boldsymbol{\theta}_m}\right) + \frac{\boldsymbol{\delta}_m \boldsymbol{\delta}_m'}{\boldsymbol{\delta}_m' \Delta\boldsymbol{\theta}_m} \tag{4.12}$$

oraz

$$B_{m+1}^{-1} = B_m^{-1} + \frac{(\Delta\boldsymbol{\theta}_m - B_m^{-1}\boldsymbol{\delta}_m)(\Delta\boldsymbol{\theta}_m - B_m^{-1}\boldsymbol{\delta}_m)'}{(\Delta\boldsymbol{\theta}_m - B_m^{-1}\boldsymbol{\delta}_m)'\boldsymbol{\delta}_m}. \tag{4.13}$$

Przykład 4.3

W tym przykładzie porównujemy oszacowania MNK parametrów modelu regresji liniowej uzyskane metodami BFGS i DFP. W pierwszym kroku losujemy dane do modelu.

```
X =[ones(100,1) randn(100,1)];  % stała i zmienne generowana z N(0,1)
e = randn(100,1);               % wektor składników losowych z N(0,1)
alfa = [1; 2];                  % prawdziwe parametry modelu
y = X*alfa+e;                   % wartości y wygenerowane z modelu
alfa_MNK = X\y                  % oszacowania MNK
```

Oszacowanie uzyskane w wektorze `alfa_MNK` wykorzystuje analityczny wzór $\hat{\boldsymbol{\alpha}} = (\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{X}'\boldsymbol{y}$. Wybieramy początkową wartość parametrów `beta_opt` oraz metodę BFGS optymalizacji funkcji celu, czyli sumy kwadratów reszt modelu.

```
beta_opt = [0; 0];              % wartości startowe parametrów
options1 = optimoptions('fminunc',...
    'Display','iter-detailed',...
    'Algorithm','quasi-newton',...
    'HessUpdate','bfgs',...
    'MaxFunctionEvaluations',600);  % wybór metody BFGS dla funkcji fminunc
```

Wybieramy początkową wartość parametrów `gamma_opt` oraz metodę DFP optymalizacji funkcji celu, czyli sumy kwadratów reszt modelu.

```
gamma_opt = [0; 0];
options2 = optimoptions('fminunc',...
    'Display','iter-detailed',...
    'Algorithm','quasi-newton',...
    'HessUpdate','dfp',...
    'MaxFunctionEvaluations',600);  % wybór metody DFP dla funkcji fminunc
```

Definiujemy funkcję celu `fun`, czyli sumę kwadratów reszt modelu znajdujemy minimum tej funkcji dwiema metodami. Zastosowanie funkcji `fminunc` wymaga posiadania zainstalowanej biblioteki Optimization Toolbox w programie MATLAB.

```
fun = @(beta) (y-X*beta)'*(y-X*beta);   % funkcja licząca sumy kwadratów reszt

[beta_opt,fval,eflag,output] = fminunc(fun,beta_opt,options1); % szuka beta_opt
[gamma_opt,fval,eflag,output]= fminunc(fun,gamma_opt,options2);% szuka gamma_opt

table(alfa_MNK, beta_opt, gamma_opt)    % porównanie oszacowań parametrów
```

Ostatnie polecenie pozwala zaprezentować oba oszacowania `alfa_MNK`, `beta_opt` i `gamma_opt` oraz porównać ich wartości.

4.5 Metoda Gaussa-Newtona

W przypadku stosowania NMNK można wykorzystać do wyznaczenia oszacowań parametrów metodę Gaussa-Newtona (np. Amemiya, 1983, str. 345-347), która bardzo przypomina oryginalną metodę MNK. Metoda ta też wykorzystuje fakt, że model $y = h(\boldsymbol{x}; \boldsymbol{\theta}) + \varepsilon$ jest różniczkowalny względem parametrów $\boldsymbol{\theta}$. Wylicza się tutaj gradient $\nabla_{\boldsymbol{\theta}} h(\boldsymbol{x}; \boldsymbol{\theta})$ funkcji $h(\boldsymbol{x}; \boldsymbol{\theta})$ objaśniającej zmienną y.

Krok $m + 1$ w tej metodzie ma wzór

$$\boldsymbol{\theta}_{m+1} = \boldsymbol{\theta}_m + \left[\left(\nabla_{\boldsymbol{\theta}} \boldsymbol{h}(\boldsymbol{\theta}_m)\right)'\left(\nabla_{\boldsymbol{\theta}} \boldsymbol{h}(\boldsymbol{\theta}_m)\right)\right]^{-1}\left(\nabla_{\boldsymbol{\theta}} \boldsymbol{h}(\boldsymbol{\theta}_m)\right)'\left(\boldsymbol{y} - \boldsymbol{h}(\boldsymbol{\theta}_m)\right), \qquad (4.14)$$

gdzie $\boldsymbol{y}$ to wektor obserwacji zmiennej y, $\boldsymbol{h}(\boldsymbol{\theta}_m)$ to wektor kolejnych obserwacji wartości prognozowanych przez model $h(\boldsymbol{x}; \boldsymbol{\theta})$ z parametrami $\boldsymbol{\theta}_m$

$$\boldsymbol{h}(\boldsymbol{\theta}_m) = [h(\boldsymbol{x}_1; \boldsymbol{\theta}_m) \quad h(\boldsymbol{x}_2; \boldsymbol{\theta}_m) \quad \dots \quad h(\boldsymbol{x}_n; \boldsymbol{\theta}_m)]',$$

natomiast $\nabla_{\boldsymbol{\theta}} \boldsymbol{h}(\boldsymbol{\theta}_m)$ jest macierzą pochodnych cząstkowych funkcji $f(\boldsymbol{x}; \boldsymbol{\theta})$ po wektorze parametrów $\boldsymbol{\theta}$ dla $\boldsymbol{\theta} = \boldsymbol{\theta}_m$ dla kolejnych obserwacji wektora $\boldsymbol{x}_i$, $i = 1, \dots, n$,

$$\nabla_{\boldsymbol{\theta}} \boldsymbol{h}(\boldsymbol{\theta}_m) = [\nabla_{\boldsymbol{\theta}} h(\boldsymbol{x}_1; \boldsymbol{\theta}_m) \quad \nabla_{\boldsymbol{\theta}} h(\boldsymbol{x}_2; \boldsymbol{\theta}_m) \quad \dots \quad \nabla_{\boldsymbol{\theta}} h(\boldsymbol{x}_n; \boldsymbol{\theta}_m)]_{k \times n}.$$

We wzorze (4.14) zmiana wartości punktu $\boldsymbol{\theta}_{m+1} - \boldsymbol{\theta}_m$ jest szacowana jak parametry regresji reszt, $\boldsymbol{y} - \boldsymbol{h}(\boldsymbol{\theta}_m)$, na $\nabla_{\boldsymbol{\theta}} \boldsymbol{h}(\boldsymbol{\theta}_m)$ przy pomocy MNK.

4.6 Algorytm Levenberga-Marquardta

Algorytm Levenberga-Marquardta polega na zastąpieniu w metodzie Gaussa-Newtona macierzy $\left[\left(\nabla_{\boldsymbol{\theta}} \boldsymbol{h}(\boldsymbol{\theta}_m)\right)'\left(\nabla_{\boldsymbol{\theta}} \boldsymbol{h}(\boldsymbol{\theta}_m)\right)\right]$ przez $\left[\left(\nabla_{\boldsymbol{\theta}} \boldsymbol{h}(\boldsymbol{\theta}_m)\right)'\left(\nabla_{\boldsymbol{\theta}} \boldsymbol{h}(\boldsymbol{\theta}_m)\right) + \lambda_m \cdot \boldsymbol{I}\right]$ lub przez $\left[\left(\nabla_{\boldsymbol{\theta}} \boldsymbol{h}(\boldsymbol{\theta}_m)\right)'\left(\nabla_{\boldsymbol{\theta}} \boldsymbol{h}(\boldsymbol{\theta}_m)\right) + \lambda_m \cdot \mathrm{diag}\left(\left(\nabla_{\boldsymbol{\theta}} \boldsymbol{h}(\boldsymbol{\theta}_m)\right)'\left(\nabla_{\boldsymbol{\theta}} \boldsymbol{h}(\boldsymbol{\theta}_m)\right)\right)\right]$ (jeszcze lepiej dla zachowania skali), gdzie $\mathrm{diag}(Q)$ oznacza macierz diagonalną złożoną z elementów leżących na głównej przekątnej macierzy Q (Marquardt, 1963; Amemiya, 1983, str. 346). Skalar λ_m może być ustalany w każdym kroku, tak by przyspieszyć znalezienie lokalnego optimum (minimum w przypadku metody Gaussa-Newtona). W analogiczny sposób może być

modyfikowana macierz $H(\boldsymbol{\theta_m})$ w metodzie Newtona w celu zapewnienia jej dodatniej określoności (por. np. Goldfeld, Quandt i Trotter, 1966).

Przykład 4.4

W tym przykładzie poszukujemy oszacowania MNK parametrów modelu regresji liniowej metodą Levenberga-Marquardta. W pierwszym kroku losujemy dane do modelu.

```
X =[ones(100,1) randn(100,1)];  % stała i zmienne generowana z N(0,1)
e = randn(100,1);               % wektor składników losowych z N(0,1)
alfa = [1; 2];                  % prawdziwe parametry modelu
y = X*alfa+e;                   % wartości y wygenerowane z modelu
alfa_MNK = X\y                  % oszacowania MNK
```

Oszacowanie uzyskane w wektorze `alfa_MNK` wykorzystuje analityczny wzór $\hat{\boldsymbol{\alpha}} = (\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{X}'\boldsymbol{y}$. Wybieramy początkową wartość parametrów `beta_opt` oraz metodę Levenberga-Marquardta optymalizacji funkcji celu, czyli sumy kwadratów reszt modelu.

```
beta_opt = [0; 0];              % wartości startowe parametrów
options  = optimoptions('lsqnonlin',...
    'Display','iter-detailed',...
    'Algorithm','levenberg-marquardt',...
    'MaxFunctionEvaluations',600);  % wybór opcji dla funkcji lsqnonlin
```

Nie musimy definiować funkcji celu, ale zamiast niej definiujemy funkcję `res` liczącą reszty modelu. Funkcja `lsqnonlin` minimalizuje sumę kwadratów reszt wyliczoną przez funkcję `res`. Zastosowanie funkcji `lsqnonlin` wymaga posiadania zainstalowanej biblioteki Optimization Toolbox w programie MATLAB.

```
res = @(beta) (y-X*beta);       % funkcja licząca reszty

[beta_opt,suma_kw_reszt,reszty,exitflag,output,lambda,jacobian] = ...
    lsqnonlin(res,beta_opt,[],[],options); % szuka beta_opt

table(alfa_MNK, beta_opt)    % porównanie oszacowań parametrów
```

Ostatnie polecenie pozwala zaprezentować oba oszacowania `alfa_MNK` i `beta_opt` oraz porównać ich wartości.

Należy pamiętać, że metody gradientowe polegają na poszukiwaniu lokalnego optimum funkcji celu i globalne optimum może znajdować się gdzie indziej. W praktyce ważne jest rozpoczęcie algorytmu gradientowego odpowiednio blisko potencjalnego optimum globalnego, tak by zwiększyć szanse jego odnalezienia. Ponadto warto wybrać różne punkty startowe dla wybranego algorytmu i sprawdzić, czy dla każdego z nich algorytm odnajduje to samo optimum funkcji celu. Punkty startowe mogą zostać wybrane w sposób losowy, na

przykład poprzez wylosowanie poszczególnych wartości parametrów z odpowiednich rozkładów jednostajnych (lub innych rozkładów) na wybranych przedziałach.

4.7 Metoda Neldera-Meada

Metoda Neldera i Meada (1965), nazywana także sympleksową metodą spadku (downhill simplex method), jest algorytmem wyznaczania minimum funkcji celu bez korzystania z pochodnych tej funkcji. Algorytm polega na konstruowaniu figury sympleksowej, przekształcaniu jej i przesuwaniu w kierunku minimum funkcji celu, tak by jeden z wierzchołków tej figury na końcu procedury wyznaczał minimum funkcji celu.

Niech minimalizowana będzie funkcja $Q(\boldsymbol{\theta})$, gdzie $\boldsymbol{\theta} \in \mathbb{R}^k$. Szukamy takiego wektora $\boldsymbol{\theta}_{min}$, który minimalizuje $Q(\boldsymbol{\theta})$. Rozpoczęcie algorytmu wymaga wyboru pierwszych $k+1$ wektorów wartości parametrów $\boldsymbol{\theta}_1$, …, $\boldsymbol{\theta}_{k+1}$. Te pierwsze zgadywane wektory, powinny być różne i możliwie bliskie poszukiwanej wartości $\boldsymbol{\theta}_{min}$. Algorytm składa się z następujących kroków.

1) Posortuj wektory w taki sposób by spełnione były nierówności $Q(\boldsymbol{\theta}_1) \leq Q(\boldsymbol{\theta}_2) \leq \cdots \leq Q(\boldsymbol{\theta_{k+1}})$. Sprawdź, zgodnie z ustalonymi kryteriami, czy punkt $\boldsymbol{\theta}_1$ jest odpowiednio blisko punktu wyznaczającego minimum $Q(\boldsymbol{\theta})$. Jeśli tak jest, to zatrzymaj procedurę.
2) Znajdź punkt $\boldsymbol{\theta}_0$ będący centroidem figury wyznaczonej przez wierzchołki $\boldsymbol{\theta}_1$, …, $\boldsymbol{\theta}_k$ (bez punktu $\boldsymbol{\theta}_{k+1}$). Współrzędne tego punktu oblicza się jako średnie z odpowiednich współrzędnych ze wszystkich punktów $\boldsymbol{\theta}_1$, …, $\boldsymbol{\theta}_k$.
3) Krok odbicia (reflection): wyznacz punkt „odbicia” $\boldsymbol{\theta_r} = \boldsymbol{\theta}_0 + \alpha(\boldsymbol{\theta}_0 - \boldsymbol{\theta}_{k+1})$, gdzie $\alpha > 0$. Jeśli $Q(\boldsymbol{\theta}_1) \leq Q(\boldsymbol{\theta_r}) < Q(\boldsymbol{\theta}_k)$, to podmień $\boldsymbol{\theta}_{k+1}$ na $\boldsymbol{\theta_r}$ i powróć do kroku (1).
4) Krok ekspansji (expansion): jeżeli punkt odbicia $\boldsymbol{\theta_r}$ jest najlepszym z punktów wybranych do tej pory, czyli $Q(\boldsymbol{\theta_r}) < Q(\boldsymbol{\theta}_1)$, to wyznacz punkt „ekspansji” $\boldsymbol{\theta_e} = \boldsymbol{\theta}_0 + \gamma(\boldsymbol{\theta_r} - \boldsymbol{\theta}_0)$, gdzie $\gamma > 1$. Jeśli tak utworzony punkt $\boldsymbol{\theta_e}$ jest lepszy niż punkt odbicia, czyli $Q(\boldsymbol{\theta_e}) < Q(\boldsymbol{\theta_r})$, to podmień $\boldsymbol{\theta}_{k+1}$ na $\boldsymbol{\theta_e}$ i powróć do kroku (1). Jeśli jednak $Q(\boldsymbol{\theta_e}) \geq Q(\boldsymbol{\theta_r})$, to podmień $\boldsymbol{\theta}_{k+1}$ na $\boldsymbol{\theta_r}$ i powróć do kroku (1).
5) Krok kurczenia (contraction): wyznacz punkt „kurczenia” $\boldsymbol{\theta_c} = \boldsymbol{\theta}_0 + \rho(\boldsymbol{\theta}_{n+1} - \boldsymbol{\theta}_0)$, gdzie $0 < \rho \leq 0{,}5$. Jeżeli punkt kurczenia nie jest najgorszy, czyli $Q(\boldsymbol{\theta_c}) < Q(\boldsymbol{\theta}_{k+1})$, to podmień $\boldsymbol{\theta}_{k+1}$ na $\boldsymbol{\theta_e}$ i powróć do kroku (1).

6) Krok zmniejszania (shrink): podmień wszystkie punkty z wyjątkiem najlepszego $\boldsymbol{\theta}_1$ wykorzystując następującą formułę $\boldsymbol{\theta}_i = \boldsymbol{\theta}_1 + \sigma(\boldsymbol{\theta}_i - \boldsymbol{\theta}_1)$, gdzie $0 < \sigma < 1$. Następnie powróć do kroku (1).

Standardowo przyjmuje się, że $\alpha = 1$, $\gamma = 2$, $\rho = 0{,}5$ oraz $\sigma = 0{,}5$.

Przykład 4.5

W tym przykładzie poszukujemy oszacowania MNK parametrów modelu regresji liniowej metodą Neldera-Meada. W pierwszym kroku losujemy dane do modelu.

```
X =[ones(100,1) randn(100,1)];  % stała i zmienne generowana z N(0,1)
e = randn(100,1);               % wektor składników losowych z N(0,1)
alfa = [1; 2];                  % prawdziwe parametry modelu
y = X*alfa+e;                   % wartości y wygenerowane z modelu
alfa_MNK = X\y                  % oszacowania MNK
```

Oszacowanie uzyskane w wektorze `alfa_MNK` wykorzystuje analityczny wzór $\widehat{\boldsymbol{\alpha}} = (\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{X}'\boldsymbol{y}$. Wybieramy początkową wartość parametrów `beta_opt` oraz metodę Neldera-Meada optymalizacji funkcji celu, czyli sumy kwadratów reszt modelu.

```
beta_opt = [0; 0];              % wartości startowe parametrów
options  = optimset(...
    'Display','iter',...
    'MaxFunEvals',600);  % wybór opcji dla funkcji lsqnonlin
```

Definiujemy funkcję celu `fun`, czyli sumę kwadratów reszt modelu znajdujemy minimum tej funkcji dwiema metodami. Funkcja `fminsearch` minimalizuje sumę kwadratów reszt wyliczoną przez funkcję `fun`.

```
fun = @(beta) (y-X*beta)'*(y-X*beta);   % funkcja licząca sumy kwadratów reszt

[beta_opt,fval,eflag,output] = fminsearch(fun,beta_opt,options);% szuka beta_opt

table(alfa_MNK, beta_opt)    % porównanie oszacowań parametrów
```

Ostatnie polecenie pozwala zaprezentować oba oszacowania `alfa_MNK` i `beta_opt` oraz porównać ich wartości.

4.8 Metoda symulowanego wyżarzania

Metoda symulowanego wyżarzania (simulated annealing) służy do wyznaczania maksimum funkcji celu bez potrzeby wyliczania pochodnych tej funkcji. Charakteryzuje się tym, że wartości parametrów w kolejnych krokach są wybierane losowo (Kirkpatrick, Gelatt Jr i Vecchi, 1983). Złe wartości parametrów (powiązane z małą wartością funkcji celu) są odrzucane z pewnym prawdopodobieństwem, natomiast dobre wartości parametrów

(powiązane z coraz wyższymi wartościami funkcji celu) są zawsze akceptowane. W odwrotny sposób przebiega procedura minimalizacji funkcji celu.

Niech maksymalizowana będzie funkcja $L(\boldsymbol{\theta})$, gdzie $\boldsymbol{\theta} \in \mathbb{R}^k$. Szukamy takiego wektora $\boldsymbol{\theta}_{max}$, który maksymalizuje $L(\boldsymbol{\theta})$. W poniższym opisie operator „:=" w wyrażeniach typu „$X := Y$" oznacza przypisanie zmiennej X wartości Y. Algorytm symulowanego wyżarzania składa się z następujących kroków (Goffe, Ferrier i Rogers, 1994).

1) Wybierz wartości początkowe $\boldsymbol{\theta}_0$ parametrów $\boldsymbol{\theta}$. W tym kroku podstaw $\boldsymbol{\theta}_{max} := \boldsymbol{\theta}_0$, $L := L(\boldsymbol{\theta}_0)$ oraz $L_{max} := L$.
2) Powtarzaj następujące działania dla $i = 1, \ldots, k$, czyli dla kolejnych elementów wektora parametrów $\boldsymbol{\theta}$. Niech $\theta_{(i)}$ oznacza i-ty element wektora parametrów $\boldsymbol{\theta}$. Niech $\boldsymbol{v}$ oznacza k-elementowy wektor długości kroku. Wyznacz $\theta^*_{(i)} = \theta_{(i)} + r \cdot v_{(i)}$, gdzie r jest wartością niezależnej zmiennej losowej wygenerowanej z rozkładu jednostajnego na przedziale $[-1, 1]$, $r \sim U([-1,1])$, a $v_{(i)}$ oznacza i-ty element wektora $\boldsymbol{v}$. Elementy $\theta^*_{(i)}$, gdzie $i = 1, \ldots, k$, tworzą k-elementowy wektor $\boldsymbol{\theta}^*$.
    a. Jeżeli $L(\boldsymbol{\theta}^*) \leq L$, to policz $p^* = \exp\left(\frac{L(\boldsymbol{\theta}^*) - L}{T}\right)$ i porównaj z p, gdzie p jest losowo wygenerowane z rozkładu jednostajnego $U([0,1])$. Jeżeli $p^* > p$, to nowy punkt jest akceptowany. Wtedy podstaw $\theta_{(i)} := \theta^*_{(i)}$, $\boldsymbol{\theta} := \boldsymbol{\theta}^*$ i $L := L(\boldsymbol{\theta}^*)$.
    b. Jeżeli $L(\boldsymbol{\theta}^*) > L$, to nowy punkt jest akceptowany. Wtedy podstaw $\theta_{(i)} := \theta^*_{(i)}$, $\boldsymbol{\theta} = \boldsymbol{\theta}^*$ i $L := L(\boldsymbol{\theta}^*_{(i)})$.
    c. Jeżeli $L(\boldsymbol{\theta}^*) > L_{max}$, to $\boldsymbol{\theta}^*$ odpowiada najwyższej wartości $L(\boldsymbol{\theta})$ wyznaczonej do tej pory w tym algorytmie. Podstaw zatem $L_{max} := L(\boldsymbol{\theta}^*)$ i $\boldsymbol{\theta}_{max} := \boldsymbol{\theta}^*$.
3) Powtarzaj krok (2) N_s razy. W ten sposób każdy element $\theta_{(i)}$ wektora parametrów $\boldsymbol{\theta}$ będzie losowany N_s razy.
4) Powtarzaj krok (3) N_T razy, a za każdym razem dostosuj długość kroku $\boldsymbol{v}$ tak, by około połowy nowych punktów było akceptowanych.
5) Powtarzaj krok (4) aż do zatrzymania procedury, a po każdej iteracji sprawdź następujące warunki:
    a. Jeżeli w ostatnich N_c iteracjach zmiana L_{max} była mniejsza od ε oraz $|\mathrm{L} - L(\boldsymbol{\theta}^*_m)| < \varepsilon$, to zatrzymaj procedurę i raportuj $\boldsymbol{\theta}_{max}$ i L_{max}.

b. W przeciwnym przypadku, podstaw $\boldsymbol{\theta}_0 := \boldsymbol{\theta}_{max}$ oraz $T := r_T \cdot T$ i wykonuj dalej powtórzenia.

Goffe, Ferrier i Rogers (1994) w jednym z problemów optymalizacyjnych rozwiązywanych przy użyciu algorytmu wyżarzania przyjęli $N_c = 4$, $N_s = 20$, $N_T = 100$, $r_T = 0{,}85$, $T_0 = 5000000$, $t = 10^{-8}$, $v_{(i)} = 100$ dla każdego $i = 1, \ldots, k$, $\varepsilon = 10^{-11}$.

Przykład 4.5

W tym przykładzie poszukujemy oszacowania MNK parametrów modelu regresji liniowej metodą symulowanego wyżarzania. W pierwszym kroku losujemy dane do modelu.

```
X =[ones(100,1) randn(100,1)];  % stała i zmienne generowana z N(0,1)
e = randn(100,1);               % wektor składników losowych z N(0,1)
alfa = [1; 2];                  % prawdziwe parametry modelu
y = X*alfa+e;                   % wartości y wygenerowane z modelu
alfa_MNK = X\y                  % oszacowania MNK
```

Oszacowanie uzyskane w wektorze `alfa_MNK` wykorzystuje analityczny wzór $\widehat{\boldsymbol{\alpha}} = (\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{X}'\boldsymbol{y}$. Wybieramy początkową wartość parametrów `beta_opt` oraz metodę symulowanego wyżarzania optymalizacji funkcji celu, czyli sumy kwadratów reszt modelu.

```
beta_opt = [0; 0];              % wartości startowe parametrów
options = optimoptions(@simulannealbnd,...
    'Display','iter',...
    'MaxFunctionEvaluations',60000);  % wybór opcji dla funkcji simulannealbnd
```

Zwróćmy uwagę na dużą dozwoloną liczbę wyliczeń funkcji celu. Definiujemy funkcję celu `fun`, czyli sumę kwadratów reszt modelu znajdujemy minimum tej funkcji dwiema metodami. Funkcja `simulannealbnd` minimalizuje sumę kwadratów reszt wyliczoną przez funkcję `fun`. Zastosowanie funkcji `simulannealbnd` wymaga posiadania zainstalowanej biblioteki Global Optimization Toolbox w programie MATLAB.

```
fun = @(beta) (y-X*beta)'*(y-X*beta);   % funkcja licząca sumy kwadratów reszt

[beta_opt,fval,eflag,output] = simulannealbnd(fun,beta_opt,[],[],options); %
szuka beta_opt

table(alfa_MNK, beta_opt)    % porównanie oszacowań parametrów
```

Ostatnie polecenie pozwala zaprezentować oba oszacowania `alfa_MNK` i `beta_opt` oraz porównać ich wartości.

Literatura

Amemiya, T. (1983) Non-linear regression models, w: Griliches Z., Intriligator M. D. (red.), Handbook of Econometrics, 1, North Holland, rozdz. 6, str. 333-389.

Berndt, E. R., Hall, B. H., Hall, R. E., Hausman J. A. (1974) Estimation and Inference in Nonlinear Structural Models, Annals of Econometric and Social Measurement, 3, 653-666.

Davidon, W. C. (1959) Variable Metric Method for Minimization, AEC Research and Development Report ANL-5990, Argonne National Laboratory; opublikowany także jako: Davidon, W. C. (1991) Variable metric method for minimization, SIAM Journal on Optimization, 1(1), 1-17.

Nelder, J. A., Mead R. (1965) A simplex method for function minimization, Computer Journal, **7**(4), 308-313.

Fletcher, R. (1987) Practical methods of optimization, John Wiley and Sons.

Fletcher, R., Powell, M. J. D. (1963) A Rapidly Convergent Descent Method for Minimization, Computer Journal, 6, 163-168.

Goffe, W. G., Ferrier G. D., Rogers J. (1994) Global optimization of statistical functions with simulated annealing, Journal of Econometrics, 60, 65-99.

Goldfeld, S. M., Quandt, R. E., Trotter, H. F. (1966) Maximization by Quadratic Hill-Climbing, Econometrica, 34, 541-551.

Hamilton, J. D. (1994) Time Series Analysis, Princeton University Press.

Kirkpatrick, S., Gelatt Jr, C. D., Vecchi, M. P. (1983) Optimization by Simulated Annealing, Science, 220(4598), 671-680.

Marquardt, D. W. (1963) An Algorithm for Least Squares Estimation of Nonlinear Parameters, Journal of the Society for Industrial and Applied Mathematics, 11(43), l-441.

Nelder, J. A., Mead R. (1965) A Simplex Method for Function Minimization, Computer Journal, 7, 308-313.

Quandt, R. (1983) Computational problems and methods, w: Griliches Z., Intriligator M. D. (red.), Handbook of Econometrics, 1, North Holland, rozdz. 12, str. 701-764.