Akademia Górniczo-Hutnicza im. Stanisława Staszica w Krakowie
Wydział Elektrotechniki Automatyki Informatyki i Inżynierii Biomedycznej

Metody stochastyczne w sterowaniu

Sprawozdanie końcowe

Dawid Lisek
Paweł Mańka
Marcin Piątek

Zestaw danych nr 6

Kraków, 2024

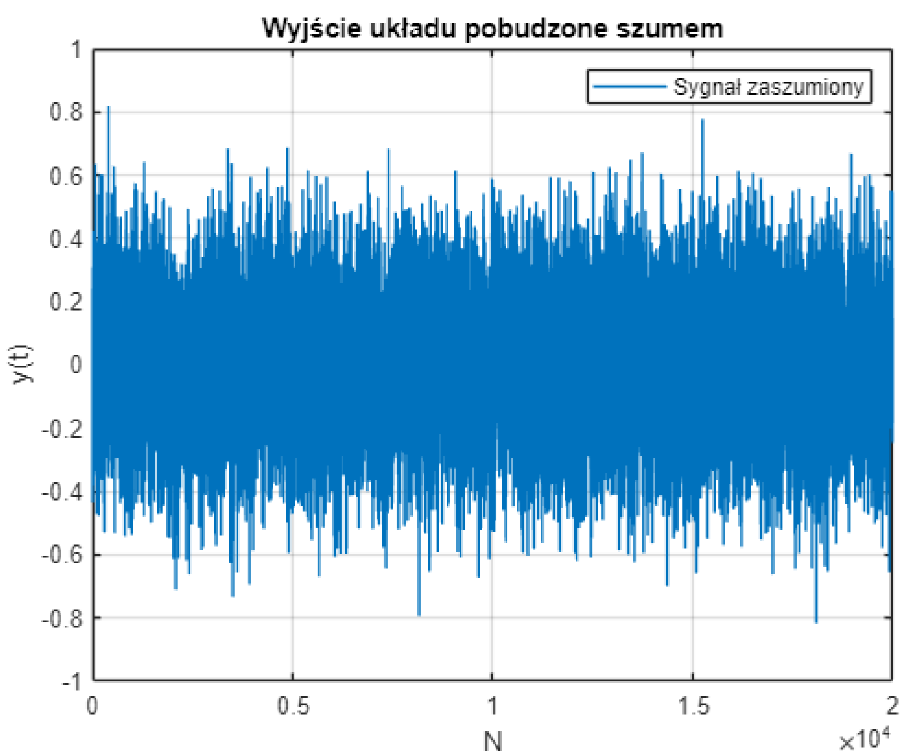
1. Cel ćwiczenia

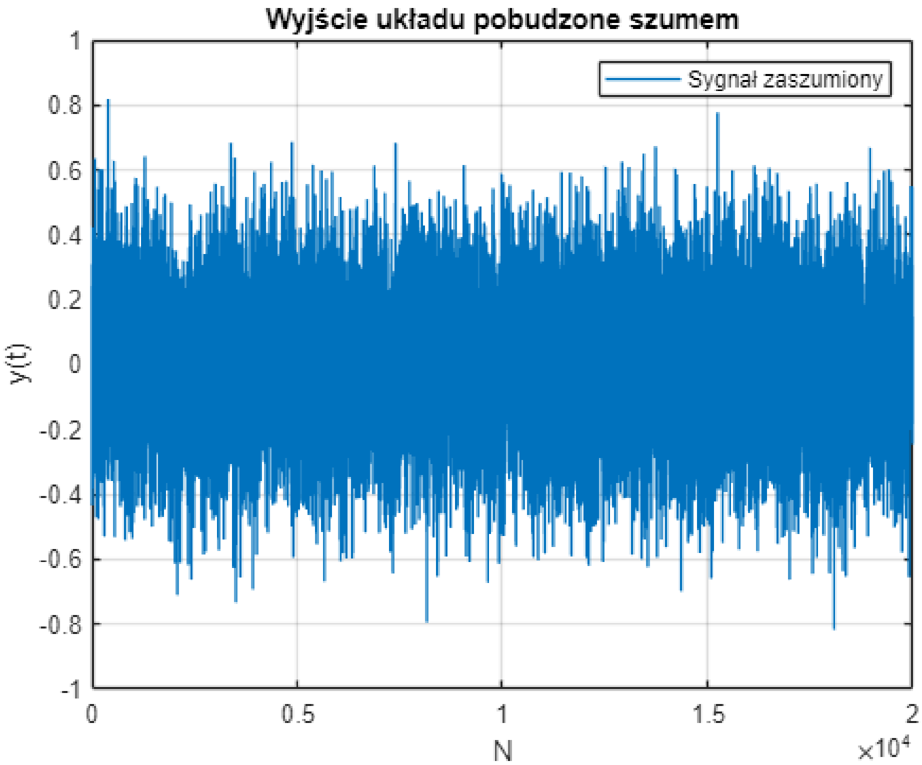
Celem ćwiczenia było oszacowanie parametrów σw, σv, a następnie narysowanie wykresu poziomicowego funkcji L w otoczeniu jej minimum. Dodatkowo, trzeba było obliczyć estymatę błędu estymacji parametrów. Po tym etapie, należało obliczyć estymatę stanu za pomocą już nastrojonego filtru Kalmana.

2. Przebieg ćwiczenia

Pierwszym krokiem był odczyt sygnału wejściowego oraz jego wyświetlenie:

```
clear all
load("src\zadanie_06.mat")
figure
plot(y)
title('Wyjście układu pobudzone szumem')
legend('Sygnał zaszumiony')
xlabel('N')
ylabel('y(t)')
grid on
```





Podane są macierze stanu: A, B, G, C, wyjście y. Za pomocą funkcji Matlab'a fmincon możemy minimalizować funkcję L względem σ oraz wyznaczyć Hesjan w punkcje σ optymalnym:

```
A = 2×2
      0.9754         0
      0.0194    0.9770

B = 2×1
      0.0082
      0.0001

C = 1×2
       0     1

G = 2×2
      0.0082    0.0001
      0.0001    0.0000
```

L – funkcja do minimalizacji:

```
function q=l_fun(th, y, A, B, G, C)
          [nx, ~] = size(A);
          N = length(y);
          th = th.^2;
          q = 0;
          m = zeros(nx, 1);
          S = zeros(nx, nx);
          u = 0;
          for k = 1:N - 1
             m = A*m + B*u;
             S = A*S*A' + (G*G') * th(1);
             W = th(2) + C*S*C';
             e = y(k + 1) - C*m;
             q = (k - 1) * q/k + (e*e/W + log(W))/k;
             S = S - (S*(C' * C)*S)/W;
             m = m+S*C'*e/th(2);
          end
          q = 0.5 * q;
end
```

Znajdowanie minimum L-funkcji:

```
handle = @(th) l_fun(th, y, A, B, G, C)
```

```
handle = function_handle with value:
    @(th)l_fun(th,y,A,B,G,C)
```

```
lb = [0,0];
ub = [3,3];
a = [];
b = [];
Aeq = [];
beq = [];
x0 = [0, 0]
```

```
x0 = 1×2
     0     0
```

```
[theta_min, min_value, ~, ~, ~, ~, H] = fmincon(handle,x0,a,b,Aeq,beq,lb,ub)
```

```
Local minimum found that satisfies the constraints.

Optimization completed because the objective function is non-decreasing in
feasible directions, to within the value of the optimality tolerance,
```

```
and constraints are satisfied to within the value of the constraint tolerance.

<stopping criteria details>
theta_min = 1×2
     1.9955    0.1999
min_value = -1.0998
```

Hesjan oraz oszacowanie błędu:

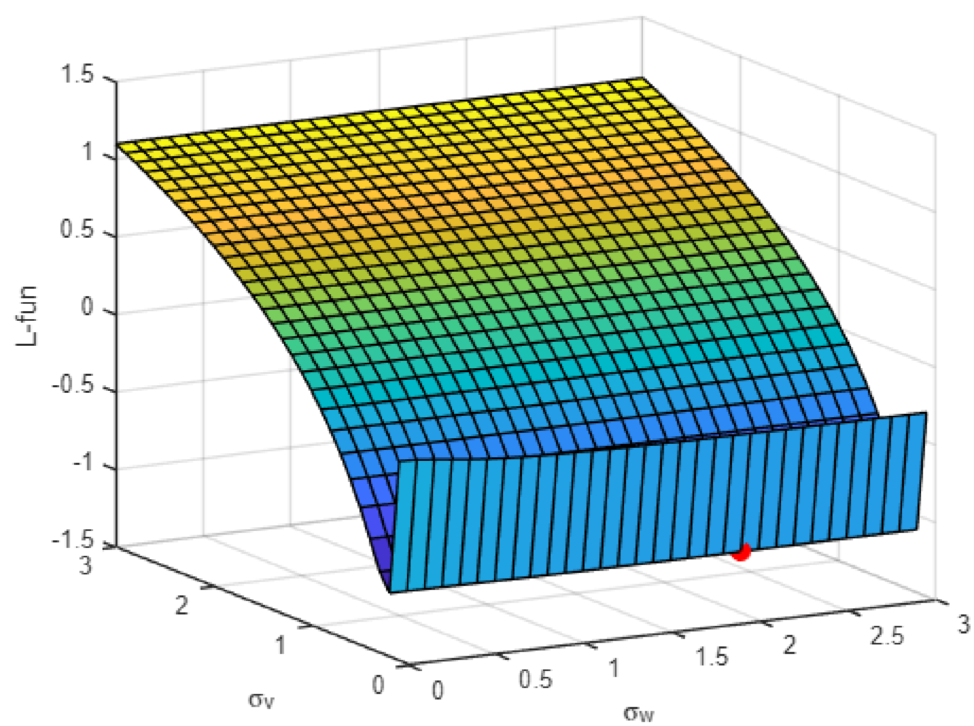
```
H
```

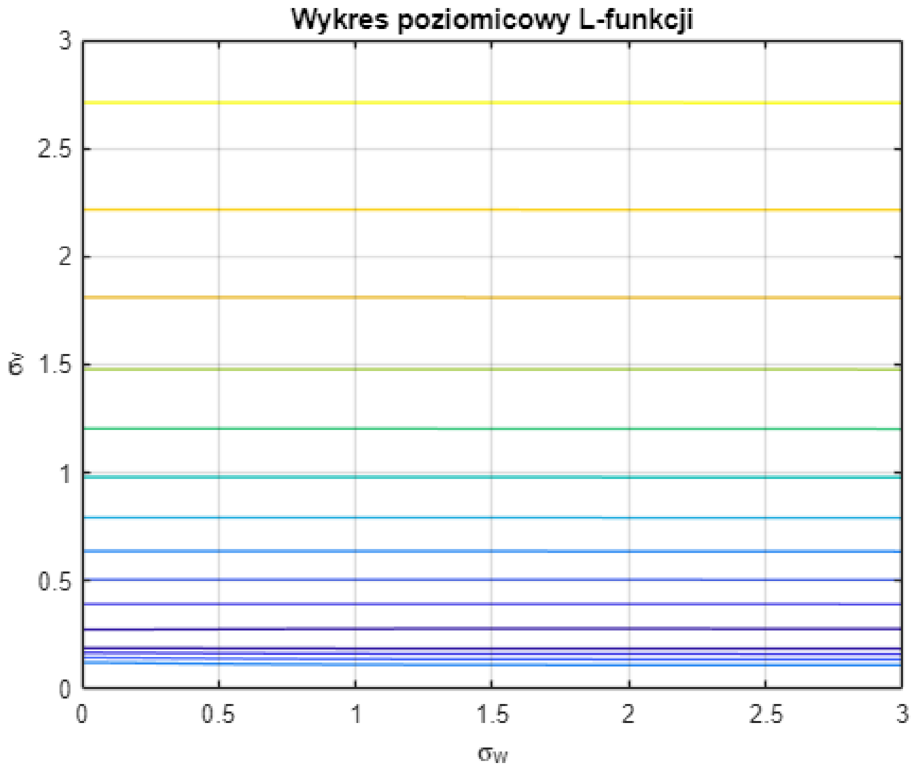
```
H = 2×2
     0.0051    0.0485
     0.0485   48.0129
```

```
S_theta_e = inv(H)
```

```
S_theta_e = 2×2
   197.0604   -0.1989
    -0.1989    0.0210
```

Następnie zostało narysowane otoczenie znalezionego minimum oraz wykres poziomicowy funkcji L:

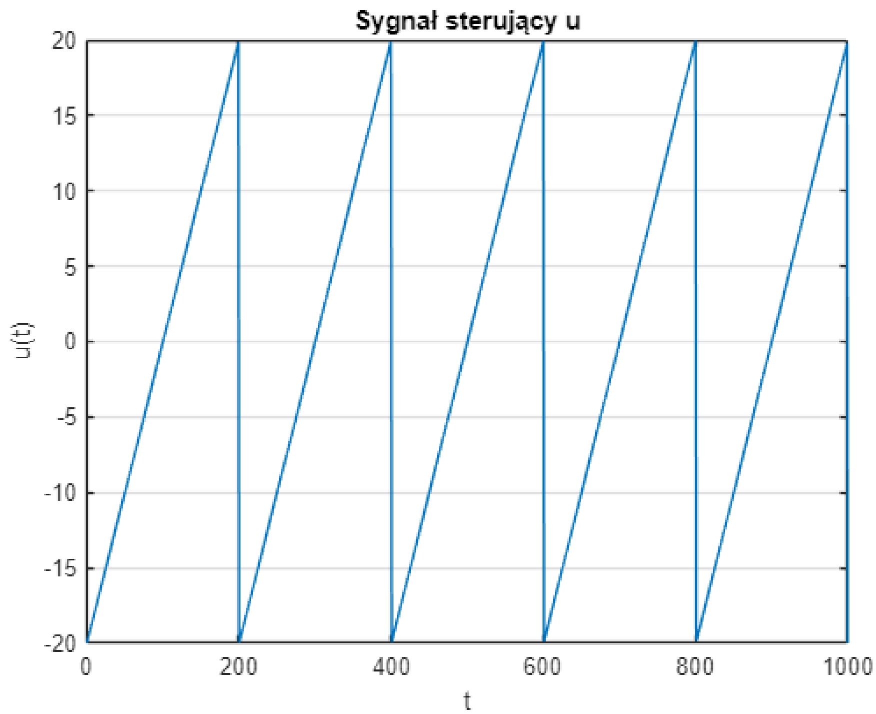




Kolejnym krokiem było wygenerowanie sygnału sterującego. Wybraliśmy sygnał typu piła:

Tworzenie sygnału sterującego:

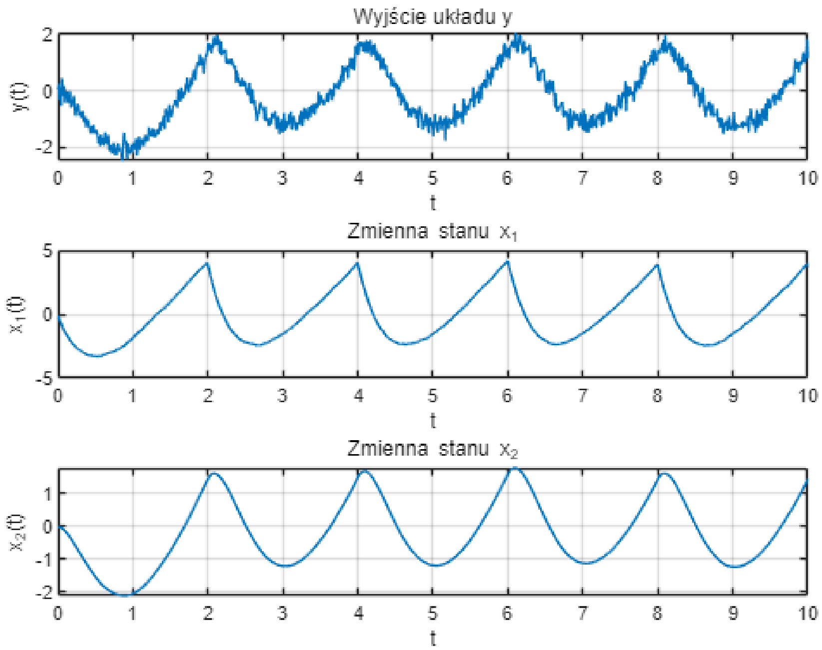
```
t = linspace(0, 10, 1000);
n = length(t);
f = 0.5;
u = 20 * sawtooth(2*pi*f*t);

figure
plot(u)
title('Sygnał sterujący u')
xlabel('t')
ylabel('u(t)')
grid on
```



Tak stworzony sygnał sterujący podano na wejściu układu, nałożono na niego szum pomiarowy oraz procesowy, których odchylenie standardowe było równe znalezionym σ_w oraz σ_v. Następnie zaimplementowano równanie opisujące układ z czasem dyskretnym:

$$x_{k+1} = Ax_k + Bu_k + Gw_k, y_k = Cx_k + v_k$$

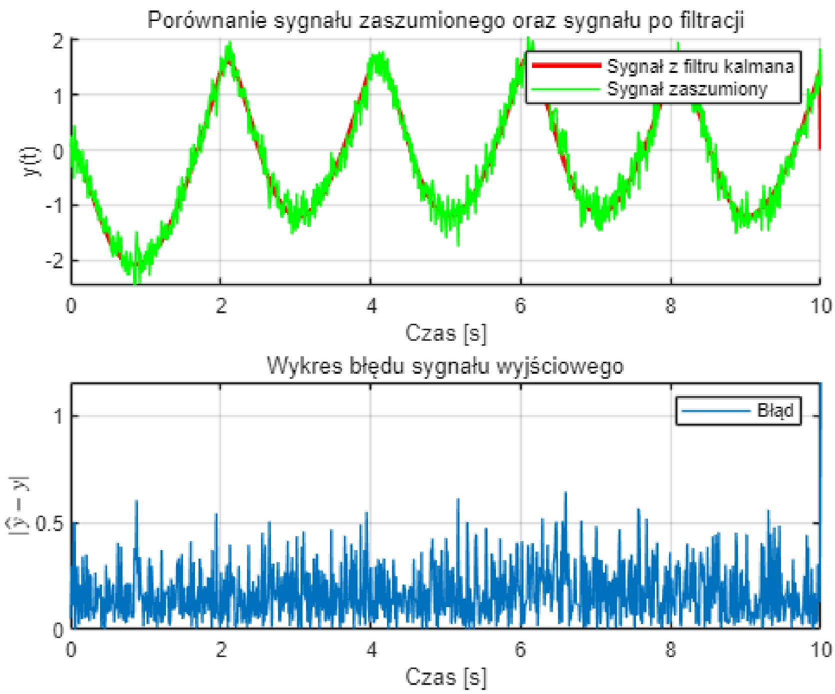
```
x = zeros(2, n);
y2 = zeros(1, n);
w = zeros(2, n);
v = zeros(1, n);

x(1:2, 1) = [0; 0];
for k = 1:n
    w(:, k) = theta_min(1, 1) * randn(2, 1);
    v(k) = theta_min(1, 2) * randn(1, 1);
    y2(k) = C * x(:, k) + v(k);
    x(:, k + 1) = A * x(:, k) + B * u(k) + G*w(:, k);
end
```



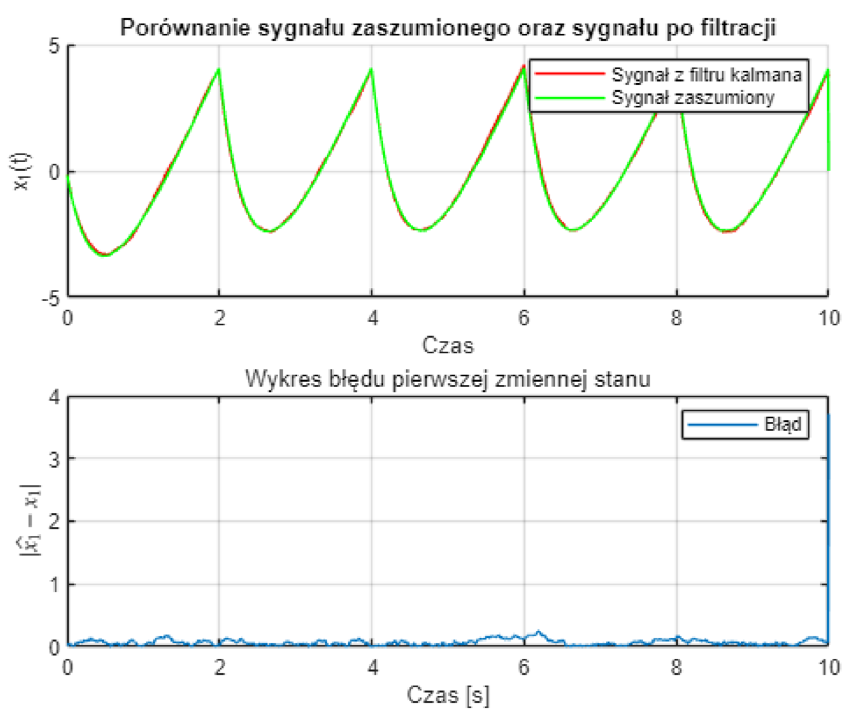
Kolejnym krokiem było zaimplementowanie filtru Kalmana oraz zastosowanie go do estymacji wejść i wyjść naszego układu:

```
function [x_e, y_e] = kalman_filter(th, y, u, A, B, G, C)
  [nx, ~] = size(A);
  N = length(y);
  th = th.^2;
  m = zeros(nx, 1);
  S = zeros(nx, nx);
  x_e = zeros(N, nx);
  y_e = zeros(N, 1);
  for k = 1:N - 1
     m = A*m + B*u(k);
     S = A*S*A' + (G*G') * th(1);
     W = th(2) + C*S*C';
     e = y(k + 1) - C*m;
     S = S - (S*(C' * C)*S)/W;
     m = m+S*C'*e/th(2);
     x_e(k, :) = m;
     y_e(k) = C*m;
  end
end
```

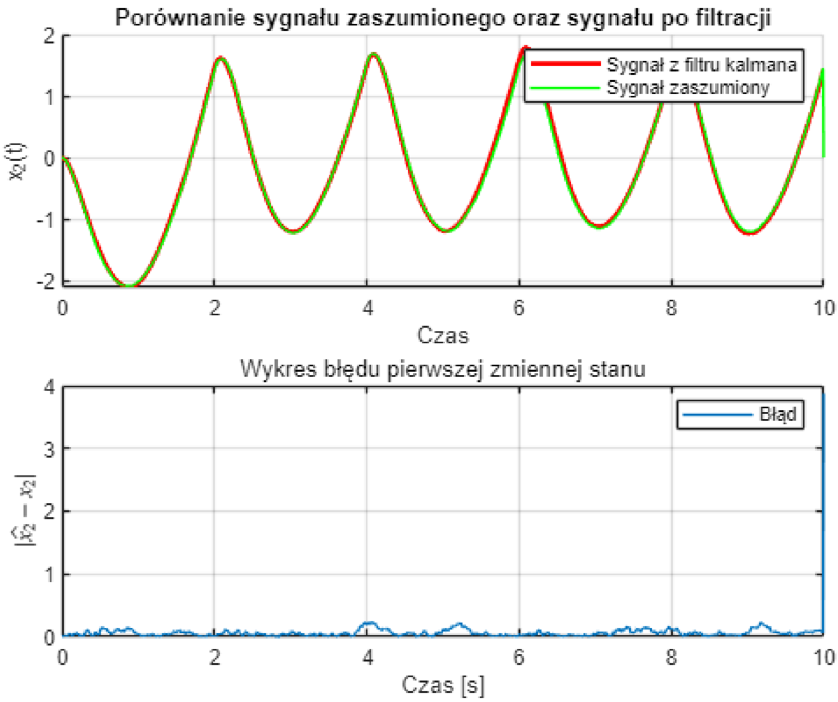
Porównanie zaszumionego wyjścia oraz sygnału uzyskanego przy pomocy filtru Kalmana:



Estymowana pierwsza zmienna stanu oraz wykres błędu:



Estymowana druga zmienna stanu oraz wykres błędu:



Jak widać filtr Kalmana radzi sobie dobrze z usunięciem szumu pomiarowego oraz procesowego z naszych sygnałów wyjściowych i wejściowych. Trajektorie sygnałów estymowanych pokrywają się z sygnałami rzeczywistymi.

3. Wnioski

Filtr Kalmana jest potężnym narzędziem stosowanym w teorii sterowania do estymacji wartości zmiennych stanu w procesie, które zawierają szumy pomiarowe, procesowe lub nie są całkowicie obserwowalne.

Metoda filtru Kalmana opiera się na matematycznym sposobie estymacji Bayesowskiej, która umożliwia optymalną estymację stanu systemu opartą na aktualnych pomiarach oraz wiedzy a priori o charakterystyce systemu. Filtr Kalmana może efektywnie redukować błędy wynikające z zaszumionych danych pomiarowych.

W naszym przypadku dzięki minimalizacji funkcji L względem parametru θ, udało się osiągnąć przybliżoną estymatę stanów x1 oraz x2. Ciekawym zagadnieniem jest wysoka wartość jednego pola w macierzy Hesjanu H(2,2), jednak po interpretacji wykresu 3d, widać że krzywizna funkcji L w kierunku σ_v jest stosunkowo spora w porównaniu do σ_w.