

Estymacja parametrów modelu

Autor: Wiktor Murawski

Przedmiot: Modelowanie matematyczne

Prowadzący: dr inż. Jakub Wagner

Politechnika Warszawska

Wydział Matematyki i Nauk Informatycznych

Oświadczam, że niniejsza praca, stanowiąca podstawę do uznania osiągnięcia efektów uczenia się z przedmiotu Modelowanie matematyczne, została wykonana przeze mnie samodzielnie.

Warszawa

27 grudnia 2024

Spis treści

1	Lista Symboli i Akronimów	3
2	Wprowadzenie	4
3	Metodyka i Wyniki Doświadczeń	4
3.1	Aproksymacja masy	5
4	Dyskusja Wyników Eksperymentów Numerycznych	5
	Bibliografia	6
	Listing Programów	7

1 Lista Symboli i Akronimów

URRZ	układ równań różniczkowych zwyczajnych
t	czas
$x_k(t)$ i $y_k(t)$	współrzędne położenia k -tego obiektu w chwili t
m	masa obiektu
G	stała grawitacyjna
$r_{jk}(t)$	odległość pomiędzy obiektami j i k dla $j, k = 1, 2, 3$ w chwili t .

2 Wprowadzenie

Dane zawarte w pliku `data_30.csv` reprezentują wyniki pomiaru położenia trzech obiektów o identycznych masach m , przyciągających się grawitacyjnie. Trajektorie ruchu tych obiektów opisane są następującym układem nieliniowych równań różniczkowych zwyczajnych drugiego rzędu:

$$\begin{cases} \frac{d^2 x_1(t)}{dt^2} = Gm \left(\frac{x_2(t) - x_1(t)}{r_{12}^3(t)} + \frac{x_3(t) - x_1(t)}{r_{13}^3(t)} \right) \\ \frac{d^2 y_1(t)}{dt^2} = Gm \left(\frac{y_2(t) - y_1(t)}{r_{12}^3(t)} + \frac{y_3(t) - y_1(t)}{r_{13}^3(t)} \right) \\ \frac{d^2 x_2(t)}{dt^2} = Gm \left(\frac{x_3(t) - x_2(t)}{r_{23}^3(t)} + \frac{x_1(t) - x_2(t)}{r_{12}^3(t)} \right) \\ \frac{d^2 y_2(t)}{dt^2} = Gm \left(\frac{y_3(t) - y_2(t)}{r_{23}^3(t)} + \frac{y_1(t) - y_2(t)}{r_{12}^3(t)} \right) \\ \frac{d^2 x_3(t)}{dt^2} = Gm \left(\frac{x_1(t) - x_3(t)}{r_{13}^3(t)} + \frac{x_2(t) - x_3(t)}{r_{23}^3(t)} \right) \\ \frac{d^2 y_3(t)}{dt^2} = Gm \left(\frac{y_1(t) - y_3(t)}{r_{13}^3(t)} + \frac{y_2(t) - y_3(t)}{r_{23}^3(t)} \right) \end{cases} \quad (1)$$

gdzie:

- t oznacza czas,
- $x_k(t)$ i $y_k(t)$ to współrzędne położenia k -tego obiektu dla $k = 1, 2, 3$,
- m to masa obiektu,
- G to stała grawitacyjna,
- $r_{jk}(t) \equiv \sqrt{[x_k(t) - x_j(t)]^2 + [y_k(t) - y_j(t)]^2}$ dla $j, k = 1, 2, 3$.

Wyniki pomiaru położenia są zaburzone oraz nieznana jest masa m obiektów. Wyznaczono przybliżone współrzędne położenia początkowych obiektów oraz przybliżenie masy m , a następnie rozwiązano układ (1) w celu wyznaczenia współrzędnych położenia obiektów w chwilach t zapisanych w pliku `query_30.csv`.

3 Metodyka i Wyniki Doświadczeń

Z pliku `data_30.csv` odczytano wyniki pomiaru położenia trzech obiektów. Wyznaczono przybliżone wartości pochodnych $x'_k(t_0)$ i $y'_k(t_0)$ dla $k = 1, 2, 3$ korzystając ze wzoru różnicy w przód:

$$x'_0 = \frac{x_1 - x_0}{t_1 - t_0}$$

Uzyskano następujące warunki początkowe położenia i ich pochodnych:

x_1	y_1	x_2	y_2	x_3	y_3
-0.554795098	0.311684711	0.231422269	-0.376185606	0.322675278	0.0579379401
x'_1	y'_1	x'_2	y'_2	x'_3	y'_3
-0.255664946	-0.557361492	0.778963645	0.481299004	-0.399831836	0.250095091

Tabela 1: Wartości x_k i y_k oraz ich pochodne dla $t = t_0$ zaokrąglone do 9 cyfr znaczących

3.1 Aproksymacja masy

W celu znalezienia wartości masy m odpowiednio bliskiej faktycznej masie obiektów przekształcono (1) następująco:

$$Gm = \frac{d^2x_1(t)}{dt^2} \frac{1}{\frac{x_2(t) - x_1(t)}{r_{12}^3(t)} + \frac{x_3(t) - x_1(t)}{r_{13}^3(t)}}$$

Analogicznie dla y_1, x_2, y_2, x_3, y_3 . Wartość drugiej pochodnej wyznaczono korzystając ze wzoru różnicy centralnej:

$$x'_n = \frac{x_{n+1} - x_{n-1}}{t_{n+1} - t_{n-1}}$$
$$x''_n = \frac{x'_{n+1} - x'_{n-1}}{t_{n+1} - t_{n-1}}$$

W ten sposób, dla różnych n , wyznaczono różne wartości mające być przybliżeniem iloczynu Gm . Przez względnie dużą niedokładność danych pomiarowych przybliżenia były błędne (na przykład ujemne bądź bardzo odchyłone od reszty), w szczególności te na końcach przedziału. W celu zminimalizowania wpływu błędów pomiarów dane wygładzono korzystając z wbudowanej w środowisko MATLAB funkcji `movmean`

4 Dyskusja Wyników Eksperymentów Numerycznych

Bibliografia

1. Dokumentacja MATLAB: <https://www.mathworks.com/help/matlab>.

Listing Programów

plik: Projekt2.m

```
1 function result = Projekt2()
2     % Wczytanie danych pomiarowych
3     data = readtable("data_30.csv");
4     t_data = data.t;
5     x1 = data.x1; y1 = data.y1;
6     x2 = data.x2; y2 = data.y2;
7     x3 = data.x3; y3 = data.y3;
8     data = [x1, y1, x2, y2, x3, y3];
9
10    % Wczytanie wartości czasu z zapytania
11    data2 = readtable("query_30.csv");
12    t_query = data2.t;
13
14    % Przybliżenie początkowych wartości pochodnych
15    dx1 = ApproximateDerivative(t_query(1:2), x1(1:2));
16    dy1 = ApproximateDerivative(t_query(1:2), y1(1:2));
17    dx2 = ApproximateDerivative(t_query(1:2), x2(1:2));
18    dy2 = ApproximateDerivative(t_query(1:2), y2(1:2));
19    dx3 = ApproximateDerivative(t_query(1:2), x3(1:2));
20    dy3 = ApproximateDerivative(t_query(1:2), y3(1:2));
21
22    % Przybliżenie początkowe masy
23    Gm = ApproximateMass(t_data, x1, x2, x3, y1, y2, y3);
24    % Gm = 0.361115455784322; % delta = 0.0169
25
26    % Warunki początkowe
27    p0 = [ x1(1); y1(1); x2(1); y2(1); x3(1); y3(1);
28          dx1(1); dy1(1); dx2(1); dy2(1); dx3(1); dy3(1); Gm];
29
30    % Minimalizacja
31    opts = optimset('TolX', 1e-6, 'TolFun', 1e-6, ...
32                  'MaxIter', 1e4, 'MaxFunEvals', 1e4);
33    pmin = fminsearch(@(p) Criterium(p, t_data, data), p0, opts);
34
35    % Zdefiniowanie rozwiązywanego URRZ
36    odefun = @(t, z) ODEFunction(t, z, Gm);
37
38    % Rozwiązanie URRZ za pomocą ode45
39    opts = odeset('RelTol', 1e-12, 'AbsTol', 1e-12);
40    [~, z] = ode45(odefun, t_query, pmin(1:12), opts);
41
42    % Przypisanie estymatów wartości x1, y1, x2, y2, x3, y3 w danych chwilach
43    x1 = z(:, 1); y1 = z(:, 2);
44    x2 = z(:, 3); y2 = z(:, 4);
45    x3 = z(:, 5); y3 = z(:, 6);
46    result = [x1, y1, x2, y2, x3, y3];
47
48    % Wizualizacja orbit
49    Visualize(data, result);
50
51    % Wyświetlenie wyniku testu dokładności
52    test_solution_30(x1, y1, x2, y2, x3, y3);
53
54 end % function
```

plik: ApproximateDerivative.m

```
1 function dy = ApproximateDerivative(t, y)
2     N = length(y);
3     dy = zeros(N, 1);
4
5     dy(1) = (y(2) - y(1)) / (t(2) - t(1)); % Różnica wprzód
6     for n = 2:N-1
7         dy(n) = (y(n+1) - y(n-1)) / (t(n+1) - t(n-1)); % Różnica środkowa
8     end
9     dy(N) = (y(N) - y(N-1)) / (t(N) - t(N-1)); % Różnica wstecz
10
11 end % function
```

plik: ODEFunction.m

```
1 function dzdt = ODEFunction(~, z, Gm)
2     x1 = z(1); y1 = z(2);
3     x2 = z(3); y2 = z(4);
4     x3 = z(5); y3 = z(6);
5     dx1 = z(7); dy1 = z(8);
6     dx2 = z(9); dy2 = z(10);
7     dx3 = z(11); dy3 = z(12);
8
9     r12 = sqrt((x1 - x2)^2 + (y1 - y2)^2);
10    r13 = sqrt((x1 - x3)^2 + (y1 - y3)^2);
11    r23 = sqrt((x2 - x3)^2 + (y2 - y3)^2);
12
13    ddx1 = Gm * ((x2 - x1)/r12^3 + (x3 - x1)/r13^3);
14    ddy1 = Gm * ((y2 - y1)/r12^3 + (y3 - y1)/r13^3);
15    ddx2 = Gm * ((x3 - x2)/r23^3 + (x1 - x2)/r12^3);
16    ddy2 = Gm * ((y3 - y2)/r23^3 + (y1 - y2)/r12^3);
17    ddx3 = Gm * ((x1 - x3)/r13^3 + (x2 - x3)/r23^3);
18    ddy3 = Gm * ((y1 - y3)/r13^3 + (y2 - y3)/r23^3);
19
20    dzdt = [ dx1; dy1; dx2; dy2; dx3; dy3;
21            ddx1; ddy1; ddx2; ddy2; ddx3; ddy3];
22
23 end % function
```

plik: Criterium.m

```
1 function J = Criterium(p, t, data)
2     Gm = p(13);
3
4     % Zdefiniowanie rozwiązywanego URRZ
5     odefun = @(t, z) ODEFunction(t, z, Gm);
6
7     % Rozwiązanie URRZ za pomocą ode45
8     opts = odeset('RelTol', 1e-9, 'AbsTol', 1e-9);
9     [~, z] = ode45(odefun, t, p(1:12), opts);
10
11    % Obliczenie błędu średniokwadratowego
12    J = sum((z(:, 1:6) - data(:, 1:6)).^2, "all");
13
14 end % function
```


plik: ApproximateMass.m

```
1 function m = ApproximateMass(t, x1, x2, x3, y1, y2, y3)
2     N = length(t);
3
4     % 'Wygładzanie' zaburzonych danych
5     window_size = 3;
6     for i = 1:5
7         x1 = movmean(x1, window_size);
8         x2 = movmean(x2, window_size);
9         x3 = movmean(x3, window_size);
10        y1 = movmean(y1, window_size);
11        y2 = movmean(y2, window_size);
12        y3 = movmean(y3, window_size);
13    end
14
15    % Przybliżanie wartości pierwszej pochodnej
16    dx1 = ApproximateDerivative(t, x1);
17    dy1 = ApproximateDerivative(t, y1);
18    dx2 = ApproximateDerivative(t, x2);
19    dy2 = ApproximateDerivative(t, y2);
20    dx3 = ApproximateDerivative(t, x3);
21    dy3 = ApproximateDerivative(t, y3);
22
23    % Przybliżanie wartości drugiej pochodnej
24    ddx1 = ApproximateDerivative(t, dx1);
25    ddy1 = ApproximateDerivative(t, dy1);
26    ddx2 = ApproximateDerivative(t, dx2);
27    ddy2 = ApproximateDerivative(t, dy2);
28    ddx3 = ApproximateDerivative(t, dx3);
29    ddy3 = ApproximateDerivative(t, dy3);
30
31    % Wyznaczenie macierzy przybliżonych wartości masy
32    m_mat = zeros(N, 6);
33    for i = 1:N
34        r12 = sqrt((x1(i)-x2(i))^2 + (y1(i)-y2(i))^2);
35        r13 = sqrt((x1(i)-x3(i))^2 + (y1(i)-y3(i))^2);
36        r23 = sqrt((x2(i)-x3(i))^2 + (y2(i)-y3(i))^2);
37        m_mat(i, 1) = ddx1(i)/((x2(i) - x1(i))/r12^3 + (x3(i) - x1(i))/r13^3);
38        m_mat(i, 2) = ddy1(i)/((y2(i) - y1(i))/r12^3 + (y3(i) - y1(i))/r13^3);
39        m_mat(i, 3) = ddx2(i)/((x3(i) - x2(i))/r23^3 + (x1(i) - x2(i))/r12^3);
40        m_mat(i, 4) = ddy2(i)/((y3(i) - y2(i))/r23^3 + (y1(i) - y2(i))/r12^3);
41        m_mat(i, 5) = ddx3(i)/((x1(i) - x3(i))/r13^3 + (x2(i) - x3(i))/r23^3);
42        m_mat(i, 6) = ddy3(i)/((y1(i) - y3(i))/r13^3 + (y2(i) - y3(i))/r23^3);
43    end
44
45    % Zignorowanie wartości ze skrajów przedziału dla każdej zmiennej
46    m_mat = m_mat(10:N-9, 1:6);
47
48    m_mat(m_mat < 0) = 10;
49
50    % Obliczenie wskaźnika odchylenia od średniej wartości wartości masy
51    % w kolumnie dla każdej kolumny
52    mean_vals = zeros(1, 6);
53    deviation = zeros(1, 6);
54    for i = 1:6
55        mean_vals(i) = median(m_mat(:, i));
56        for j = 1:length(m_mat(:, i))
57            deviation(i) = abs(sum(m_mat(:, i) - mean_vals(i)));
```

```

58     end
59 end
60
61 % Wybranie kolumny z wartościami o najmniejszym odchyleniu
62 [~, best_idx] = min(deviation);
63
64 % Przypisanie do wartości zwracanej średniej arytm. z najlepszej kolumny
65 m = mean(m_mat(:, best_idx));
66
67 end % function

```

plik: Visualize.m

```

1 function [] = Visualize(data, calc)
2     x1_data = data(:, 1); y1_data = data(:, 2);
3     x2_data = data(:, 3); y2_data = data(:, 4);
4     x3_data = data(:, 5); y3_data = data(:, 6);
5     x1_calc = calc(:, 1); y1_calc = calc(:, 2);
6     x2_calc = calc(:, 3); y2_calc = calc(:, 4);
7     x3_calc = calc(:, 5); y3_calc = calc(:, 6);
8
9     c1 = "#0000FF";
10    c2 = "#FF0000";
11    c3 = "#00FF00";
12    figure(1); clf; hold on;
13    title('Problem trzech ciał', FontSize=16);
14    xlabel('x'); ylabel('y');
15    axis equal;
16    maxy = max(max([y1_calc, y2_calc, y3_calc]));
17    miny = min(min([y1_calc, y2_calc, y3_calc]));
18    maxx = max(max([x1_calc, x2_calc, x3_calc]));
19    minx = min(min([x1_calc, x2_calc, x3_calc]));
20    a = 1.05;
21    ylim([a*miny, a*maxy]); xlim([a*minx, a*maxx]);
22    %set(gca, "Color", 'k')
23
24    % Wyświetl pozycje początkowe z danych
25    opts = {'LineStyle', 'none', 'Marker', 'o', 'MarkerSize', 24};
26    plot(x1_data(1), y1_data(1), opts{:}, Color=c1, ...
27         DisplayName='Obiekt 1 - Dane - Start');
28    plot(x2_data(1), y2_data(1), opts{:}, Color=c2, ...
29         DisplayName='Obiekt 2 - Dane - Start');
30    plot(x3_data(1), y3_data(1), opts{:}, Color=c3, ...
31         DisplayName='Obiekt 3 - Dane - Start');
32
33    % Wyświetl pozycje z danych
34    opts = {'LineStyle', 'none', 'Marker', 'o', 'MarkerSize', 6};
35    plot(x1_data, y1_data, opts{:}, Color=c1, ...
36         DisplayName='Obiekt 1 - Dane');
37    plot(x2_data, y2_data, opts{:}, Color=c2, ...
38         DisplayName='Obiekt 2 - Dane');
39    plot(x3_data, y3_data, opts{:}, Color=c3, ...
40         DisplayName='Obiekt 3 - Dane');
41
42    % Wyświetl wyznaczone pozycje początkowe
43    opts = {'LineStyle', 'none', 'Marker', '*', 'MarkerSize', 30};
44    plot(x1_calc(1), y1_calc(1), opts{:}, Color=c1, ...

```

```

45     DisplayName='Obiekt 1 Start');
46 plot(x2_calc(1), y2_calc(1), opts{:}, Color=c2, ...
47     DisplayName='Obiekt 2 Start');
48 plot(x3_calc(1), y3_calc(1), opts{:}, Color=c3, ...
49     DisplayName='Obiekt 3 Start');
50
51 % Wyświetl wyznaczone pozycje dynamicznie
52 opts = {'LineStyle', '-', 'Marker', '.', 'MarkerSize', 16};
53 plot(x1_calc, y1_calc, opts{:}, Color=c1, ...
54     DisplayName='Obiekt 1');
55 plot(x2_calc, y2_calc, opts{:}, Color=c2, ...
56     DisplayName='Obiekt 2');
57 plot(x3_calc, y3_calc, opts{:}, Color=c3, ...
58     DisplayName='Obiekt 3');
59
60 legend('show', Interpreter='latex', Color='#FFFFFF', ...
61     Location='bestoutside', FontSize=16);
62
63 end % function

```