Hleb Shypula
286720
17 czerwca 2020

Sprawozdanie

Pakiet obliczeniowy MATLAB i jego zastosowania

1. Laboratorium 1 Podstawy

prog.m

```
% tworze zapis
prompt = 'Podaj wektor [x(1), . . . , x(i)]: ';
% wyswietlam zapis i pobieram wektor
v = input(prompt);
% tworze zapis
prompt = 'Podaj funkcje - 1 (sin) lub 2 (cos) ';
% wyswietlam zapis i pobieram przel
przel = input(prompt);

% wywoluje funkcje, ktora pryjmuje wektor i przel i zwraca [...]
[maks, minim, parz, niep, niezer, moj_fun] = przetworz(v, przel);

% jezeli przel == 1 lub 2, to wyswietlam wyniki korzystajac z disp()
if przel==1 || przel==2
    disp(' ');

    disp('Wartosc maksymalna elementow wektora: ');
    disp(maks);

    disp('Wartosc minimalna elementow wektora: ');
    disp(minim);

    disp('Liczba elementow parzystych w wektorze: ');
    disp(parz);

    disp('Liczba elementow nieparzystych w wektorze: ');
    disp(niep);

    disp('Liczba elementow niezerowych w wektorze: ');
    disp(niezer);

    if przel==1
        disp('Wektor sin(v): ');
        disp(moj_fun);
    elseif przel==2
        disp('Wektor cos(v): ');
        disp(moj_fun);
    end
end
```

przetworz.m

```
% potrzebna funkcja
function [maks, minim, parz, niep, niezer, moj_fun] = przetworz(v, przel)

% jezeli przel ~= 1 i 2, to wyswietlam komunikat o bledzie
% nadaje wartosci nan naszym elementom na wyjsciu na wszelki wypadek
% i return'em koncze dzialanie programu
    if przel ~= 1 && przel ~= 2
        disp('ERROR - przel ~= 1 lub 2 ');
            maks = nan; minim = nan; parz = nan;
            niep = nan; niezer = nan; moj_fun = nan;
            return;
    end

    % szukam maksa
    maks = max(v);

    % szukam mina
    minim = min(v);

    % licze ilosc parzystych elementow
    parz = 0;
    s = size(v);
    for i=1:s(2)
        if mod(v(i),2)==0
            parz=parz+1;
        end
    end

    % licze ilosc nieparzystych elementow
    niep = 0;
    s = size(v);
    for i=1:s(2)
        if mod(v(i),2)==1
            niep=niep+1;
        end
    end

    % licze niezerowe elementy
    v_niezer = find(v);
    s_niezer = size(v_niezer);
    niezer = s_niezer(2);

    % zaleznie od przel, do moj_fun zapisuje odpowiedni wektor
    if przel==1
        moj_fun = sin(v);
    elseif przel==2
        moj_fun = cos(v);
    end

end
```

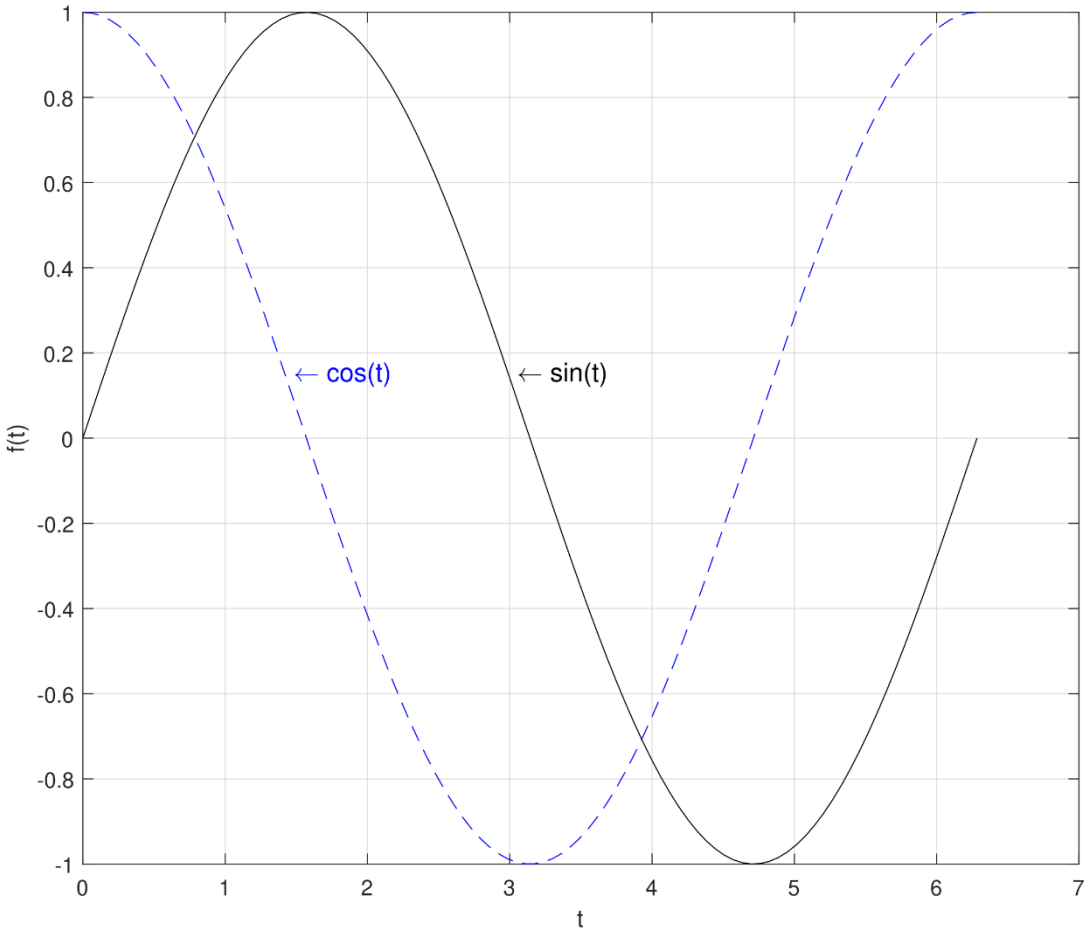
przykładowe wyniki

```
>> prog
Podaj wektor [x(1), . . . , x(i)]: [1 2 3 4 5 6 7]
Podaj funkcje - 1 (sin) lub 2 (cos): 3
ERROR - przel ~= 1 lub 2
>>
>> prog
Podaj wektor [x(1), . . . , x(i)]: [1 2 3 4 5 6 7]
Podaj funkcje - 1 (sin) lub 2 (cos): 2

Wartosc maksymalna elementow wektora:
     7

Wartosc minimalna elementow wektora:
     1

Liczba elementow parzystych w wektorze:
     3

Liczba elementow nieparzystych w wektorze:
     4

Liczba elementow niezerowych w wektorze:
     7

Wektor cos(v):
    0.5403   -0.4161   -0.9900   -0.6536    0.2837    0.9602    0.7539
```

2. Laboratorium 2 Wykresy

Zad 1a

```
% zakres zmiennej t
t = linspace(0,2*pi);
% funkcja sin
y_sin = sin(t);
% funkcja cos
y_cos = cos(t);
% rysuje w jednym wywolaniu funkcji plot
plot(t,y_sin,'-k',t,y_cos,'--b')
% siatka
grid on;
%podpis OX
xlabel('t');
% podpis OY
ylabel('f(t)');
% podpis krzywych
text(3.05,0.16,'\leftarrow sin(t)','Color','black','FontSize',12);
text(1.48,0.16,'\leftarrow cos(t)','Color','blue','FontSize',12);
```



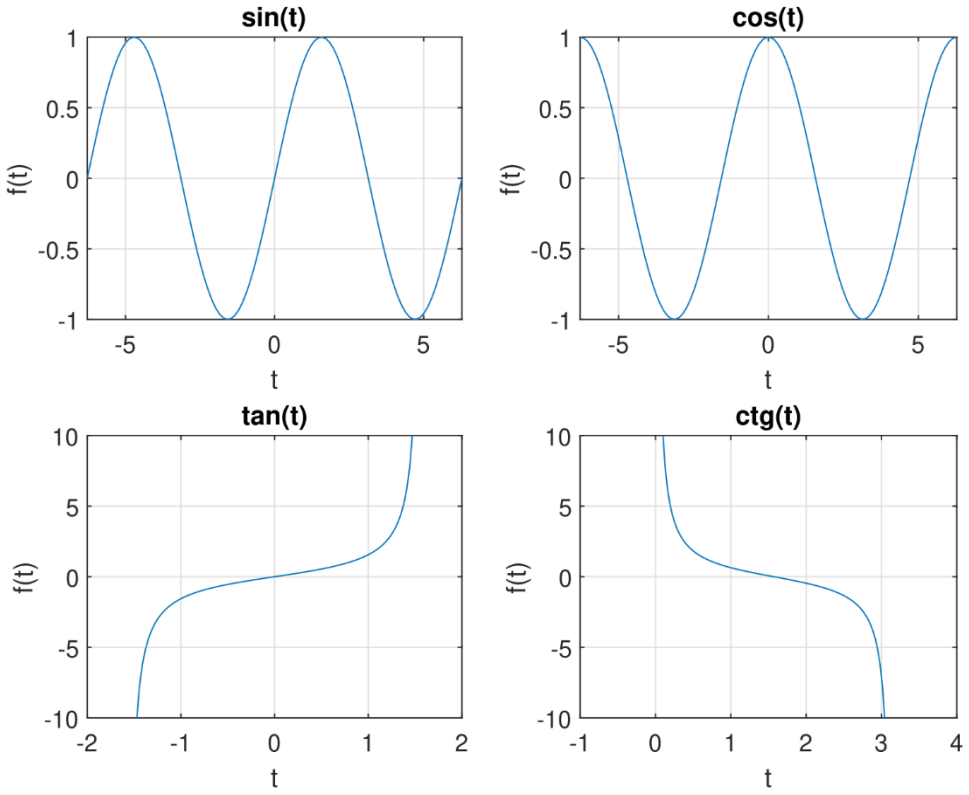
Zad 1b

```
% zakres zmiennej t
t = linspace(0,2*pi);
% funkcja sin
y_sin = sin(t);
% funkcja cos
y_cos = cos(t);
% rysuje w dwoch wywolaniach funkcji plot
plot(t,y_sin,'-k')
hold on
plot(t,y_cos,'--b')
% siatka
grid on;
%podpis OX
xlabel('t');
% podpis OY
ylabel('f(t)');
% podpis krzywych
text(3.05,0.16,'\leftarrow sin(t)','Color','black','FontSize',12);
text(1.48,0.16,'\leftarrow cos(t)','Color','blue','FontSize',12);
```

Zad 2

```
% rysuje dzielac przedzial na 4 podprzedzialy
subplot(2,2,1)
% zakres zmiennej t
t = linspace(-2*pi,2*pi);
y_sin = sin(t);
plot(t,y_sin)
% siatka
grid on;
%podpis OX
xlabel('t');
% podpis OY
ylabel('f(t)');
% podpis wykresu
title('sin(t)')

subplot(2,2,2)
t = linspace(-2*pi,2*pi);
y_cos = cos(t);
plot(t,y_cos)
grid on;
%podpis OX
xlabel('t');
% podpis OY
ylabel('f(t)');
title('cos(t)')

subplot(2,2,3)
t = linspace(-pi/2,pi/2);
y_tan = tan(t);
plot(t,y_tan)
grid on;
% zakres po OY
ylim([-10 10]);
%podpis OX
xlabel('t');
% podpis OY
ylabel('f(t)');
title('tan(t)')
```

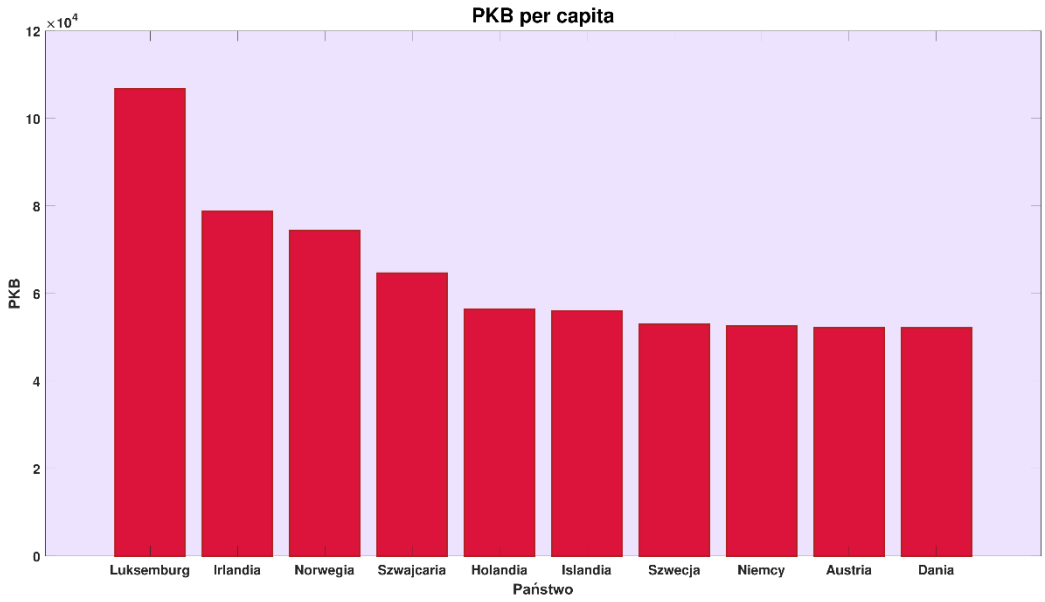
```
subplot(2,2,4)
t = linspace(0,pi);
y_ctg = cot(t);
plot(t,y_ctg)
grid on;
ylim([-10 10]);
% zakres po OX
xlim([-1 4]);
%podpis OX
xlabel('t');
% podpis OY
ylabel('f(t)');
title('ctg(t)')
```



Zad 3

1	'Luksemburg'	106705
2	'Irlandia'	78785
3	'Norwegia'	74356
4	'Szwajcaria'	64649
5	'Holandia'	56383
6	'Islandia'	55917
7	'Szwecja'	52984
8	'Niemcy'	52559
9	'Austria'	52137
10	'Dania'	52121

```
% odczytuje dane
pkb = readtable('PKB.dat');
% rysuje wykres
bar(pkb.Var1, pkb.Var3,'FaceColor',[0.863 0.078 0.235],...
'EdgeColor',[178/255 34/255 34/255],'LineWidth',2);
xlabel('Państwo');
ylabel('PKB');
title('PKB per capita');
% zmieniam indeksy na kraje
set(gca, 'XTick', 1:10, 'XTickLabel', pkb.Var2);
```



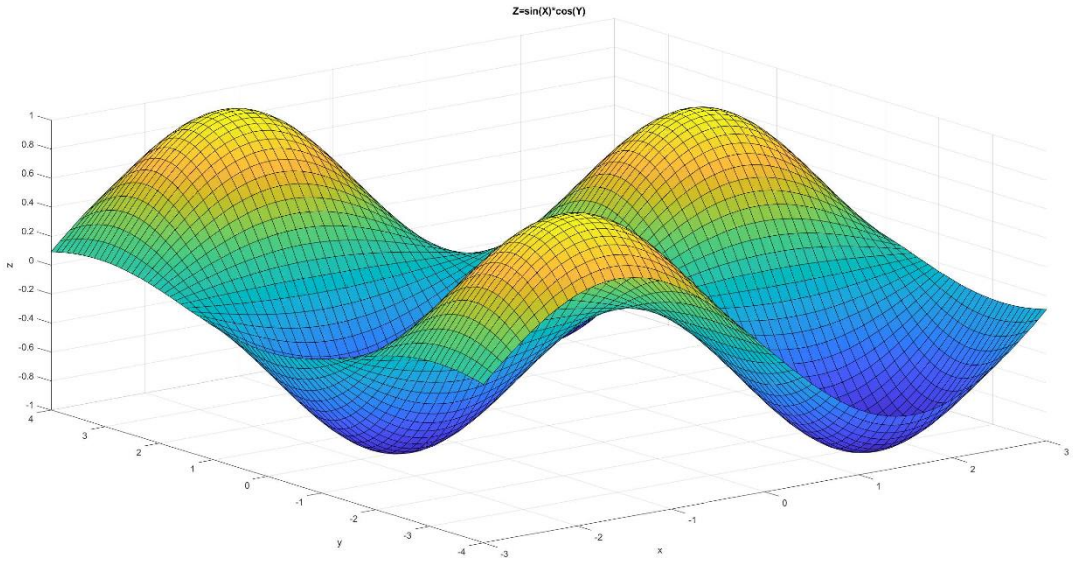
Zad 4

```
% pobieram dane
pol = readtable('Poland.dat');
gre = readtable('Greece.dat');
% tworze animacje
h = figure;
filename = 'zad4.gif';
p = animatedline('Color','b','LineWidth',3);
g = animatedline('Color','r','LineWidth',3);
% ustaliam zeby miec statyczne pole
set(gca,'XLim',[2000 2018],'YLim',[0 35000]);
% siatka
grid on;
% podpisy
xlabel('Rok');
ylabel('PKB, $');
title('PKB per capita animation');
% legenda
legend({'Polska','Grecja'},'Location','northeast','Orientation','horizontal')
% petla do rysowania ramek
for i = 1:length(pol.Var1)
    addpoints(p,pol.Var1(i),pol.Var2(i));
    addpoints(g,gre.Var1(i),gre.Var2(i));
    drawnow
    frame = getframe(h);
    im = frame2im(frame);
    [imind,cm] = rgb2ind(im,256);
    % zapis do gif
    if i == 1
        imwrite(imind,cm,filename,'gif', 'Loopcount',inf,'DelayTime',0.2);
    else
        imwrite(imind,cm,filename,'gif','WriteMode','append','DelayTime',0.2);
    end
end
```

Obrazek .gif z animacją (kliknij)

Zad 5

```
% przedzialy
[X,Y] = meshgrid(-3:0.1:3,-4:0.1:4);
% funkcja
Z = sin(X) .* cos(Y);
% wykres
surf(X,Y,Z)
% podpis OX
xlabel('x');
% podpis OY
ylabel('y');
% podpis OZ
zlabel('z');
% tytul
title('Z=sin(X)*cos(Y)');
```



3. Laboratorium 3 GUI

gui.m

```
function varargout = gui(varargin)
% GUI MATLAB code for gui.fig
%      GUI, by itself, creates a new GUI or raises the existing
%      singleton*.
%
%      H = GUI returns the handle to a new GUI or the handle to
%      the existing singleton*.
%
%      GUI('CALLBACK',hObject,eventData,handles,...) calls the local
%      function named CALLBACK in GUI.M with the given input arguments.
%
%      GUI('Property','Value',...) creates a new GUI or raises the
%      existing singleton*.  Starting from the left, property value pairs are
%      applied to the GUI before gui_OpeningFcn gets called.  An
%      unrecognized property name or invalid value makes property application
%      stop.  All inputs are passed to gui_OpeningFcn via varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help gui

% Last Modified by GUIDE v2.5 23-May-2020 14:53:31

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @gui_OpeningFcn, ...
                   'gui_OutputFcn',  @gui_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT



% --- Executes just before gui is made visible.
function gui_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to gui (see VARARGIN)

% Choose default command line output for gui
handles.output = hObject;
```

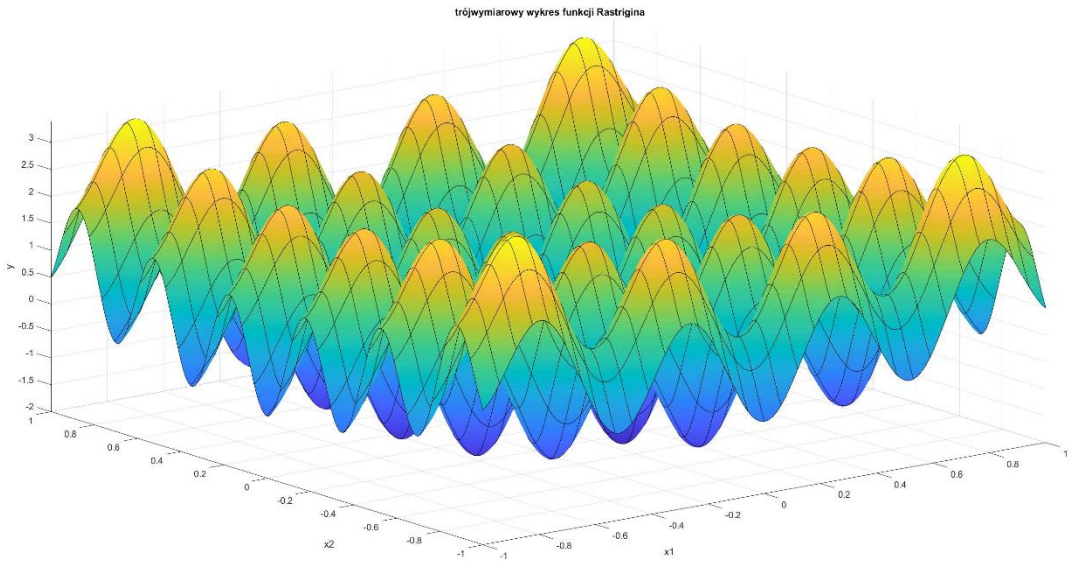
```
% Update handles structure
guidata(hObject, handles);

% UIWAIT makes gui wait for user response (see UIRESUME)
% uiwait(handles.figure1);

% --- Outputs from this function are returned to the command line.
function varargout = gui_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;



% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% wychodze po klikniecie
close;
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)



% --- Executes on button press in pushbutton2.
function pushbutton2_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% pobieram min, max i krok
min = str2double(get(handles.edit1, "String"))
max = str2double(get(handles.edit2, "String"))
krok = str2double(get(handles.edit3, "String"))
% tworze x
x = min:krok:max
% pobieram wybor
contents = cellstr(get(handles.popupmenu1, "String"))
pop_choice = contents{get(handles.popupmenu1, "Value")}
% rysuje w zaleznosci od wyboru
if(strcmp(pop_choice, 'sin(x)'))
    plot(x,sin(x));
    ylabel("sin(x)");
elseif(strcmp(pop_choice, 'cos(x)'))
    plot(x,cos(x));
    ylabel("cos(x)");
elseif(strcmp(pop_choice, 'tan(x)'))
    plot(x,tan(x));
    ylabel("tg(x)");
elseif(strcmp(pop_choice, 'ctg(x)'))
    plot(x,cot(x));
    ylabel("ctg(x)");
end

% --- Executes on selection change in popupmenu1.
function popupmenu1_Callback(hObject, eventdata, handles)
% hObject    handle to popupmenu1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns popupmenu1 contents as cell array
```

```
%        contents{get(hObject,'Value')} returns selected item from popupmenu1


% --- Executes during object creation, after setting all properties.
function popupmenu1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to popupmenu1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end




function edit1_Callback(hObject, eventdata, handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit1 as text
%        str2double(get(hObject,'String')) returns contents of edit1 as a double

% --- Executes during object creation, after setting all properties.
function edit1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end




function edit2_Callback(hObject, eventdata, handles)
% hObject    handle to edit2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit2 as text
%        str2double(get(hObject,'String')) returns contents of edit2 as a double

% --- Executes during object creation, after setting all properties.
function edit2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
function edit3_Callback(hObject, eventdata, handles)
% hObject    handle to edit3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit3 as text
%        str2double(get(hObject,'String')) returns contents of edit3 as a double

% --- Executes during object creation, after setting all properties.
function edit3_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

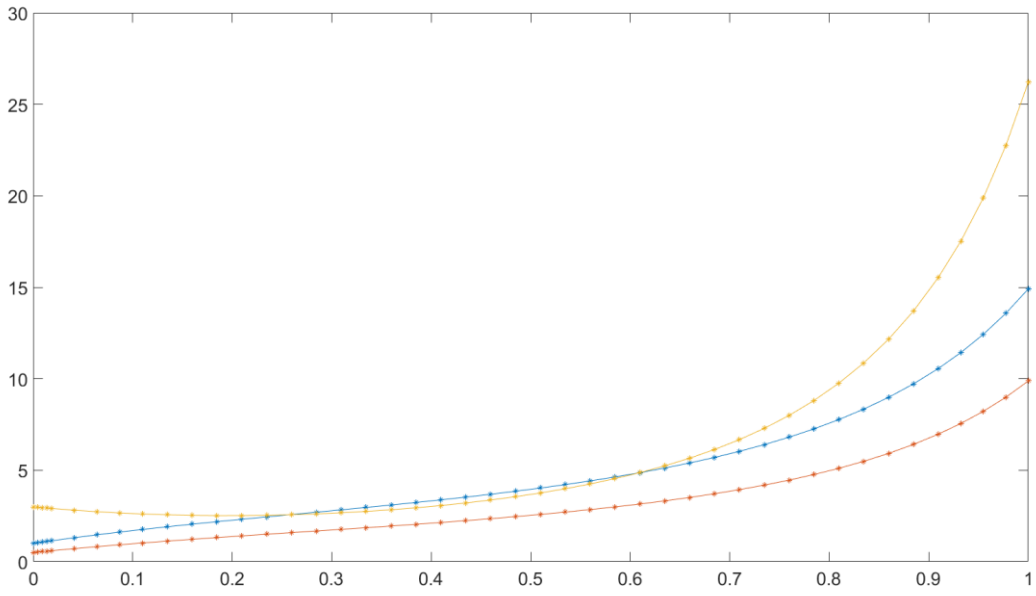
4. Laboratorium 4 Obliczenia numeryczne

```
% zad 1
A = readmatrix('L4_mac_A.txt');
B = readmatrix('L4_mac_B.txt');
C=A*B;
xlswrite ('L4_mac_C.xls', C, 'wynik');

% zad 2
num=quad('sin', 0, pi/2);
syms x;
sym=int(sin(x), 0, pi/2);
sym1=int(sin(x));

% zad 3
syms x;
f=4*x^7+5*x^4+cos(2*x);
wynik=diff(f);

% zad 4
syms x y
eqns = [2*x+2*y==-6, 10*x-5*y==30];
vars = [y x];
[soly, solx] = solve(eqns,vars);

% zad 5
syms y(x);
dsolve(diff(y)==-2*x*y);

% zad 6
syms x1 x2;
y=x1^2+x2^2-cos(12*x1)-cos(18*x2);
fsurf(y, [-1,1]);
```



5. Laboratorium 5 Zastosowania

```
clear;
set(gcf, 'Position', get(0, 'Screensize'));

[t,y] = ode45(@row_1, [0 1], [1 0.5 3]);
plot(t,y,'-*');

function dy = row_1(t,y)
dy = zeros(3,1);
dy(1) = -y(1) + 3*y(3);
dy(2) = -y(2) + 2*y(3);
dy(3) = y(1)*y(1) - 2*y(3);
end

function f = rastrigin(x)
    f = x(1)^2 + x(2)^2 - cos(12*x(1)) - cos(18*x(2))
end
```



Projekt

1. Komentarze

Jako dane do wykresów stosowałem przeważnie dane dotyczące Europy oraz Polski, ponieważ aktualnie znajdujemy się tutaj i otrzymane wyniki będą bardziej interesujące.

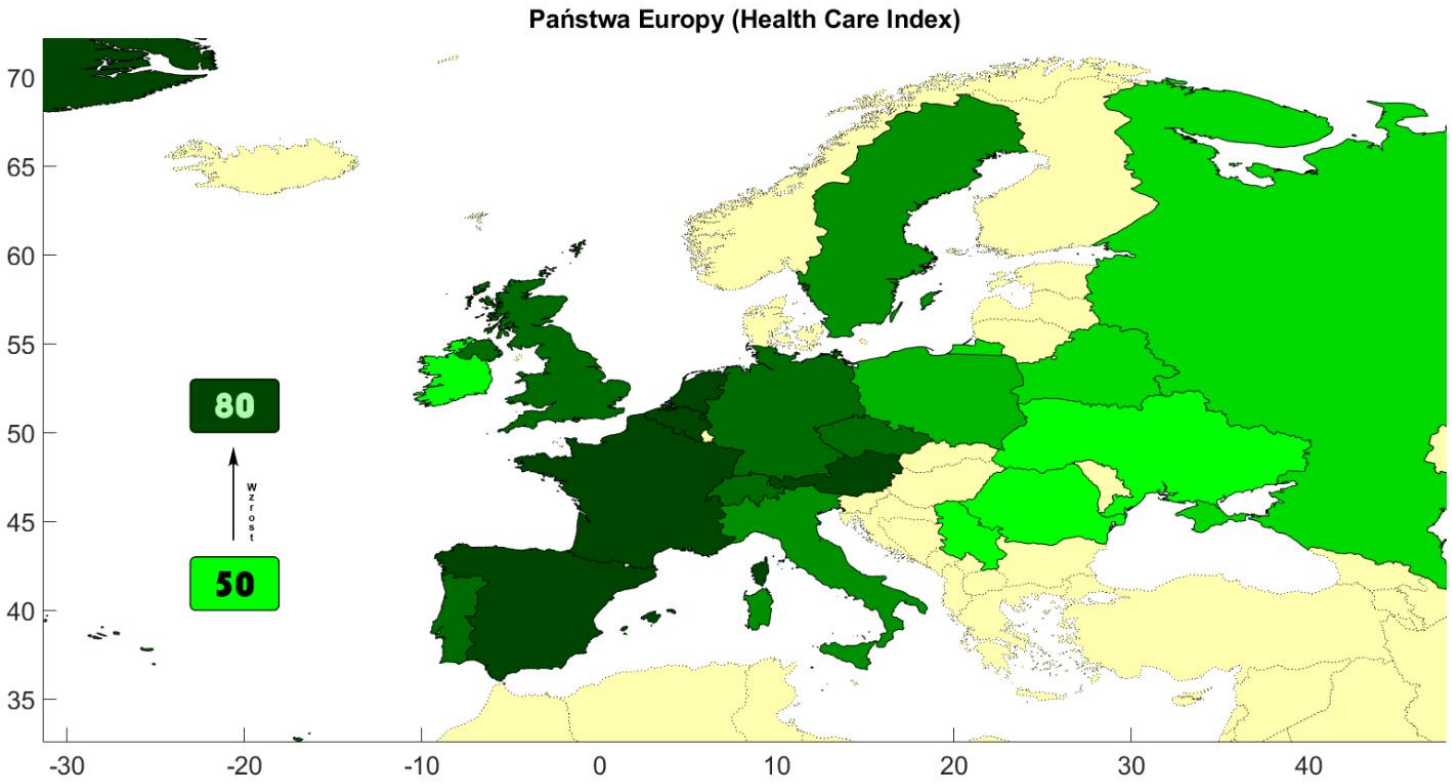
Myśle, że Panu nie chodziło o dokładność i konkretność danych, dlatego podchodziłem twórczo.

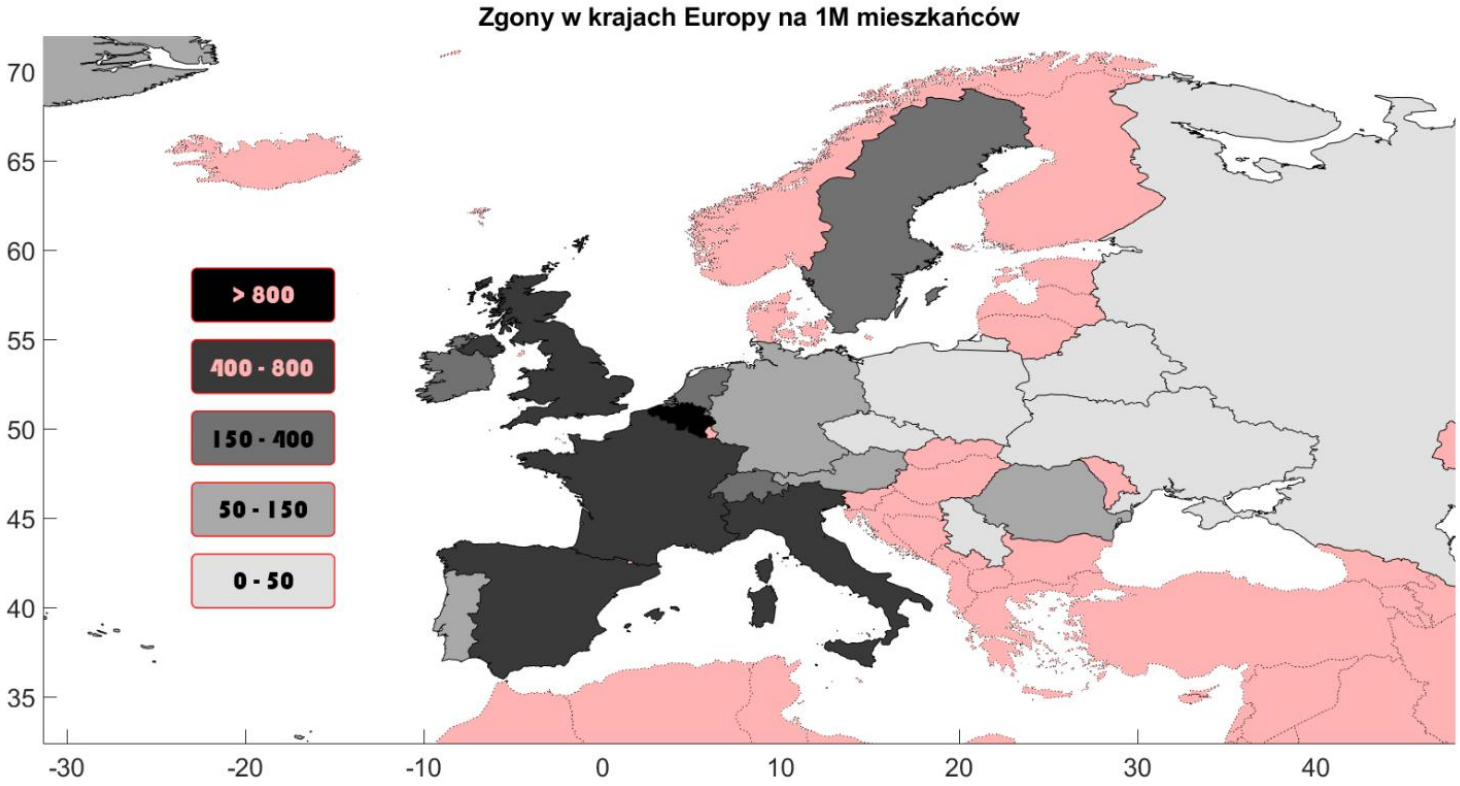
Wszystkie dane pobierałem w dniach 25-30 maja.

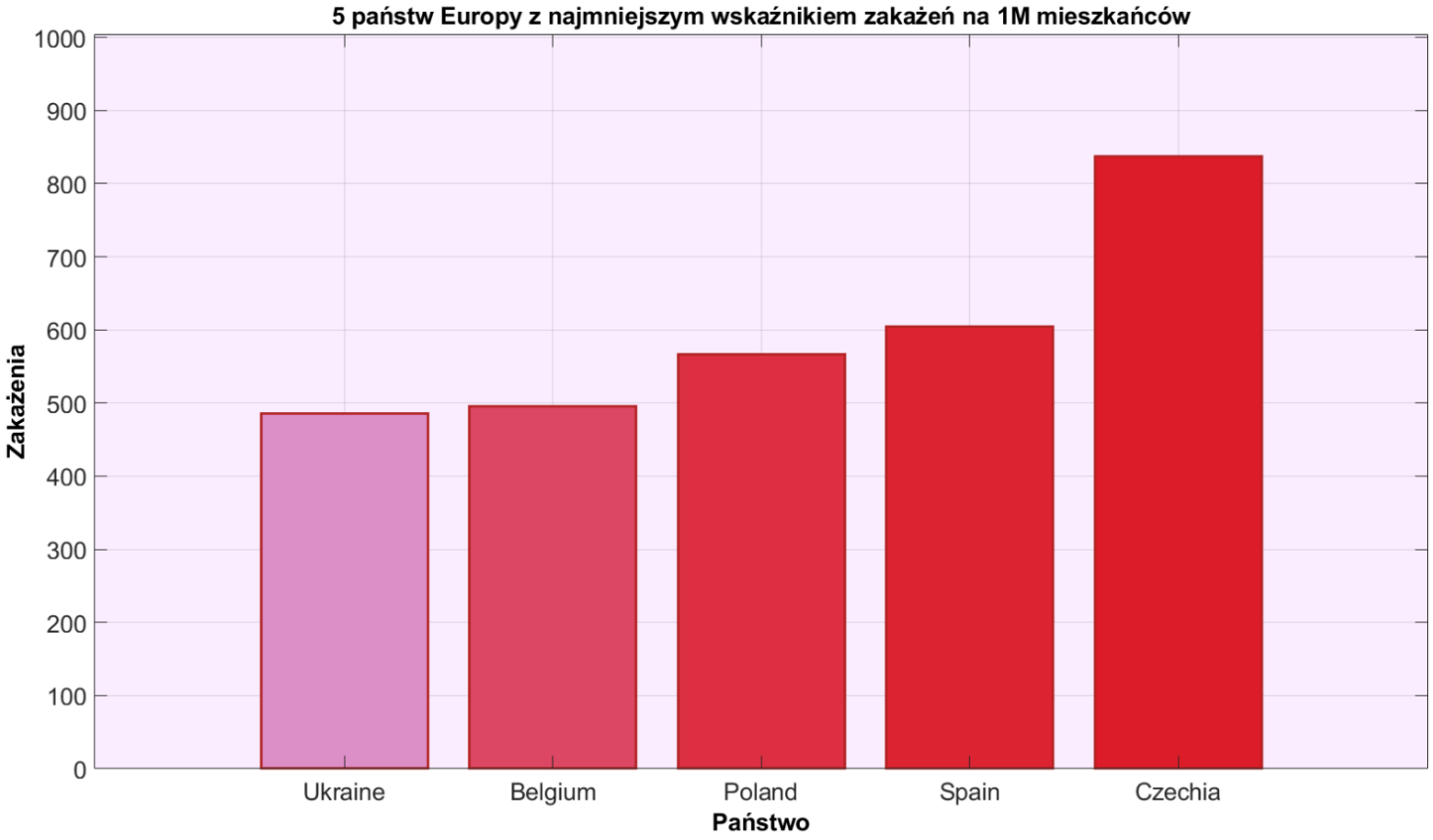
Stosowałem przeliczanie danych na 1 milion mieszkańców, myślę, że taka skala będzie bardziej przyjemna.

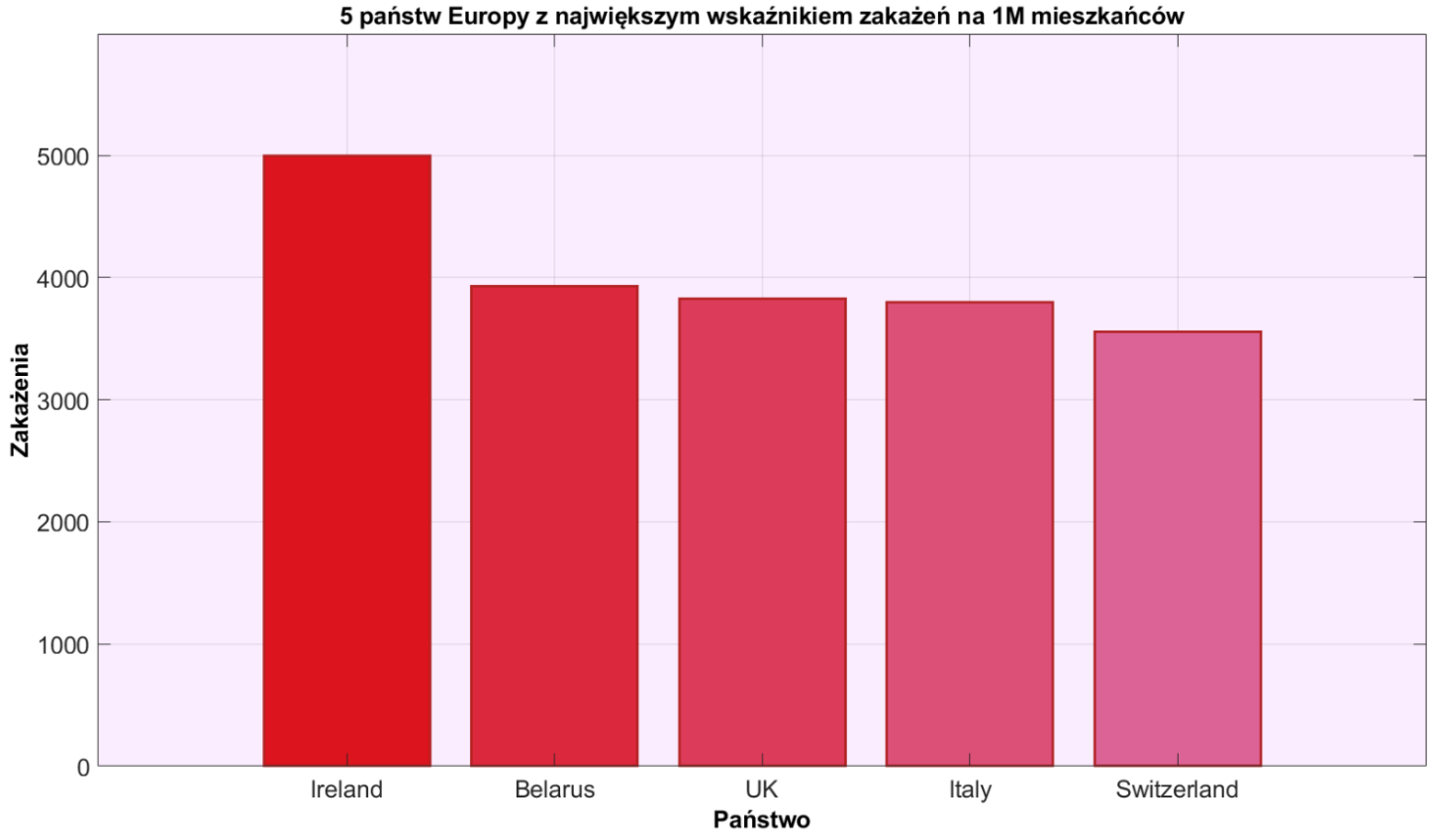
Plik źródłowy można pobrać tutaj.

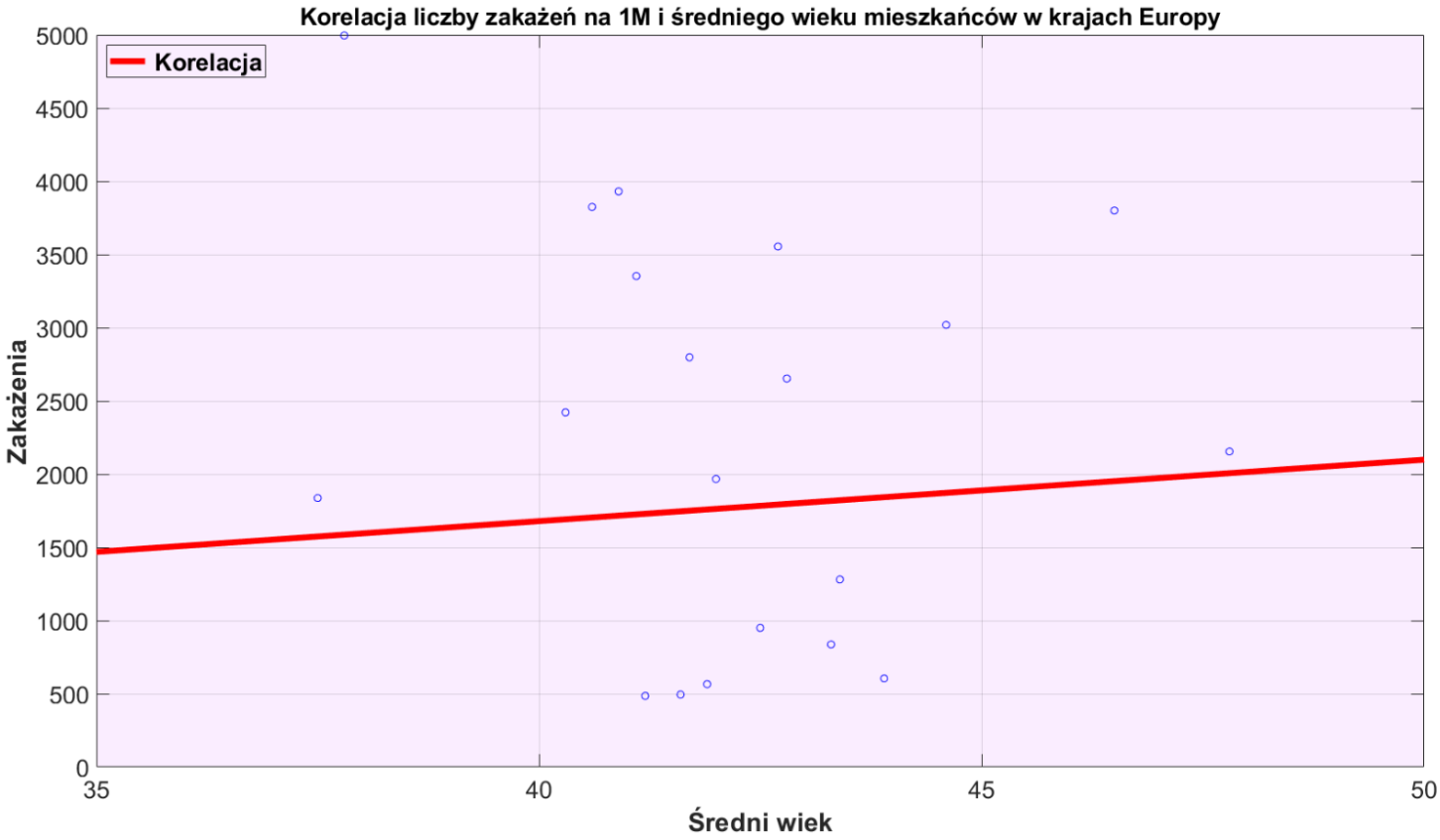
2. Wyniki

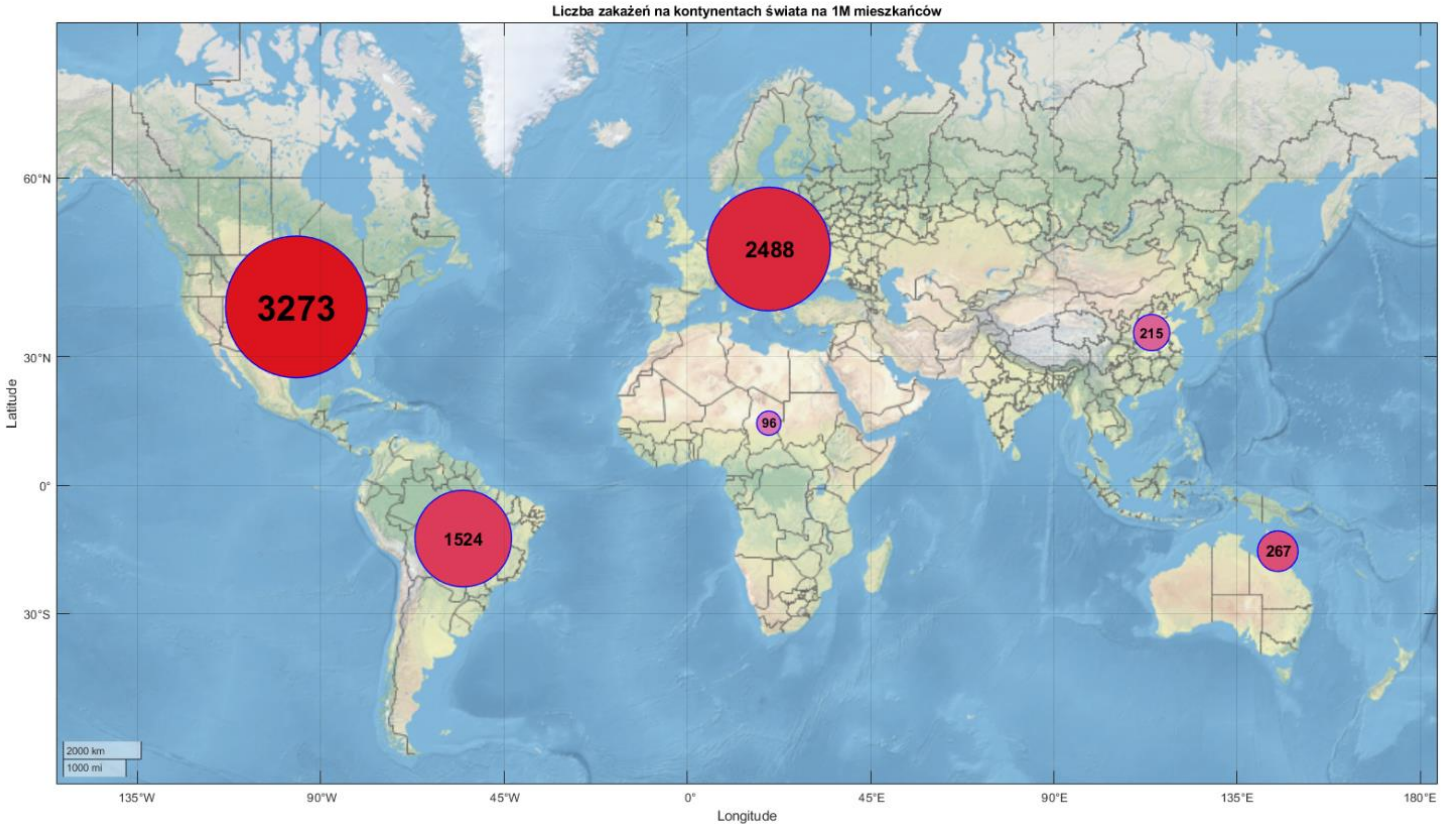

Państwa Europy (Health Care Index)
80
wzrost
50
70
65
60
55
50
45
40
35
-30
-20
-10
0
10
20
30
40

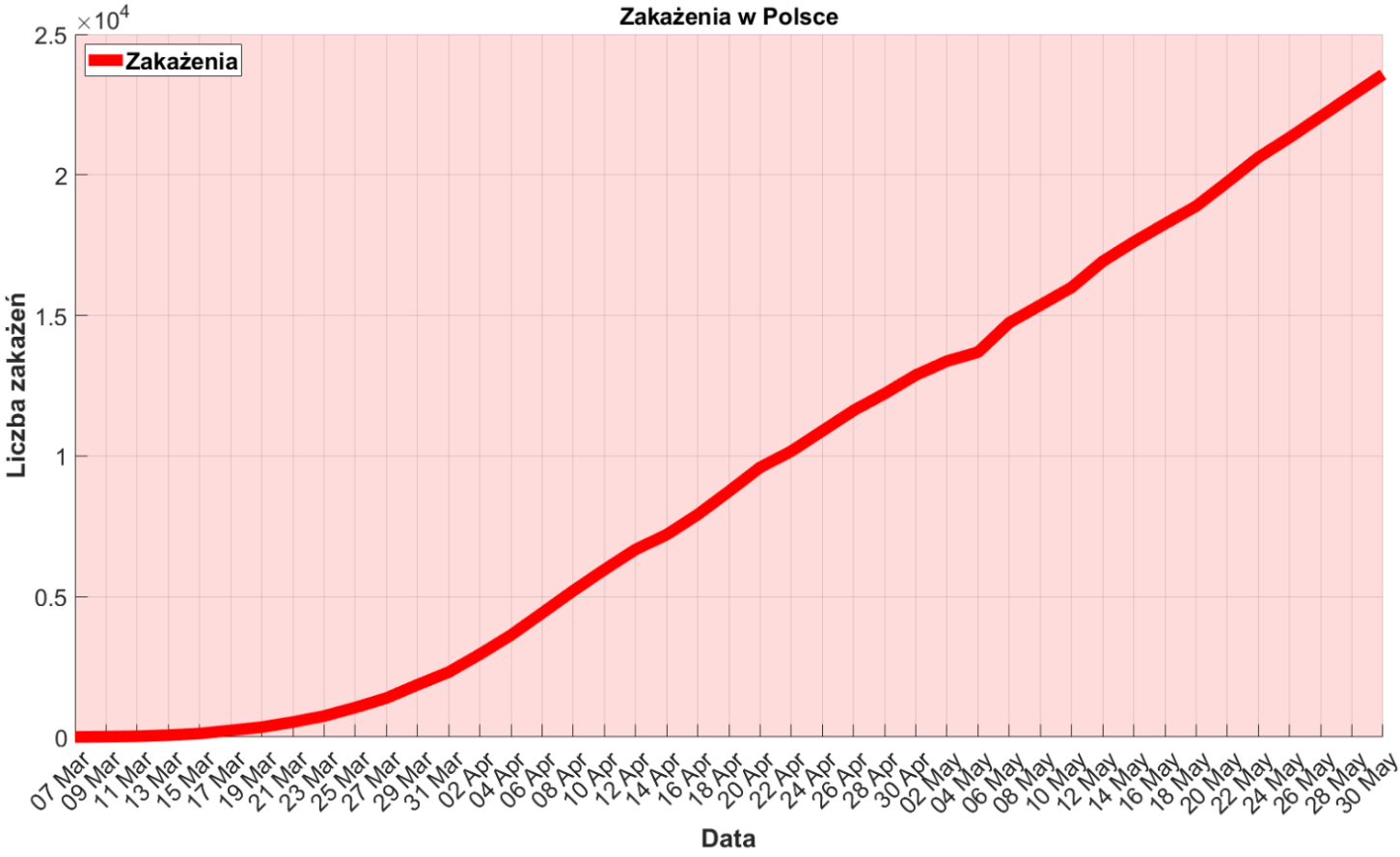

Zgony w krajach Europy na 1M mieszkańców
> 800
400 - 800
150 - 400
50 - 150
0 - 50
70
65
60
55
50
45
40
35
-30
-20
-10
0
10
20
30
40


5 państw Europy z najmniejszym wskaźnikiem zakażeń na 1M mieszkańców
Zakażenia
0
100
200
300
400
500
600
700
800
900
1000
Ukraine
Belgium
Poland
Spain
Czechia
Państwo

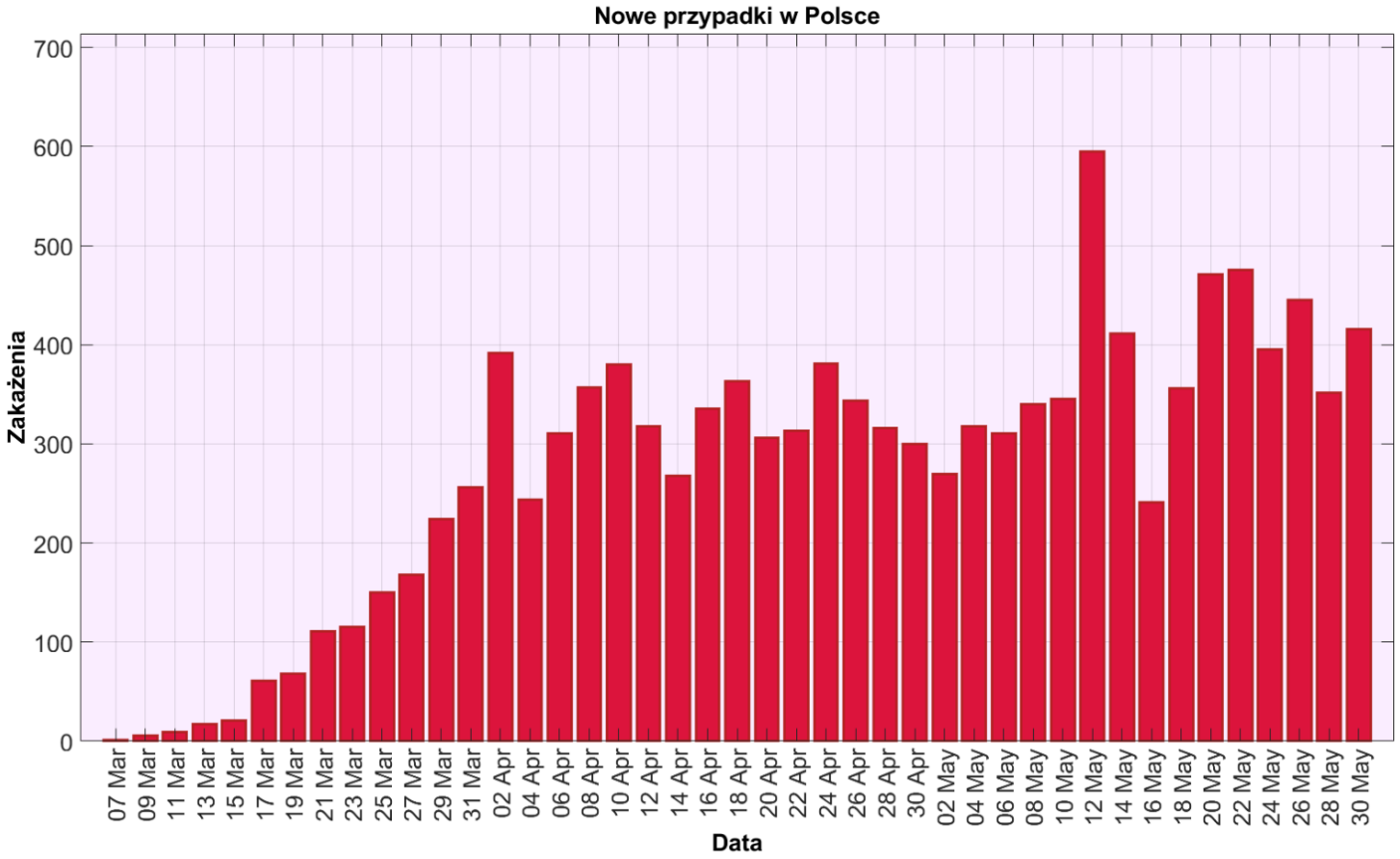

5 państw Europy z największym wskaźnikiem zakażeń na 1M mieszkańców
Zakażenia
0
1000
2000
3000
4000
5000
Ireland
Belarus
UK
Italy
Switzerland
Państwo


Korelacja liczby zakażeń na 1M i średniego wieku mieszkańców w krajach Europy
Korelacja
Zakażenia
Średni wiek
0
500
1000
1500
2000
2500
3000
3500
4000
4500
5000
35
40
45
50

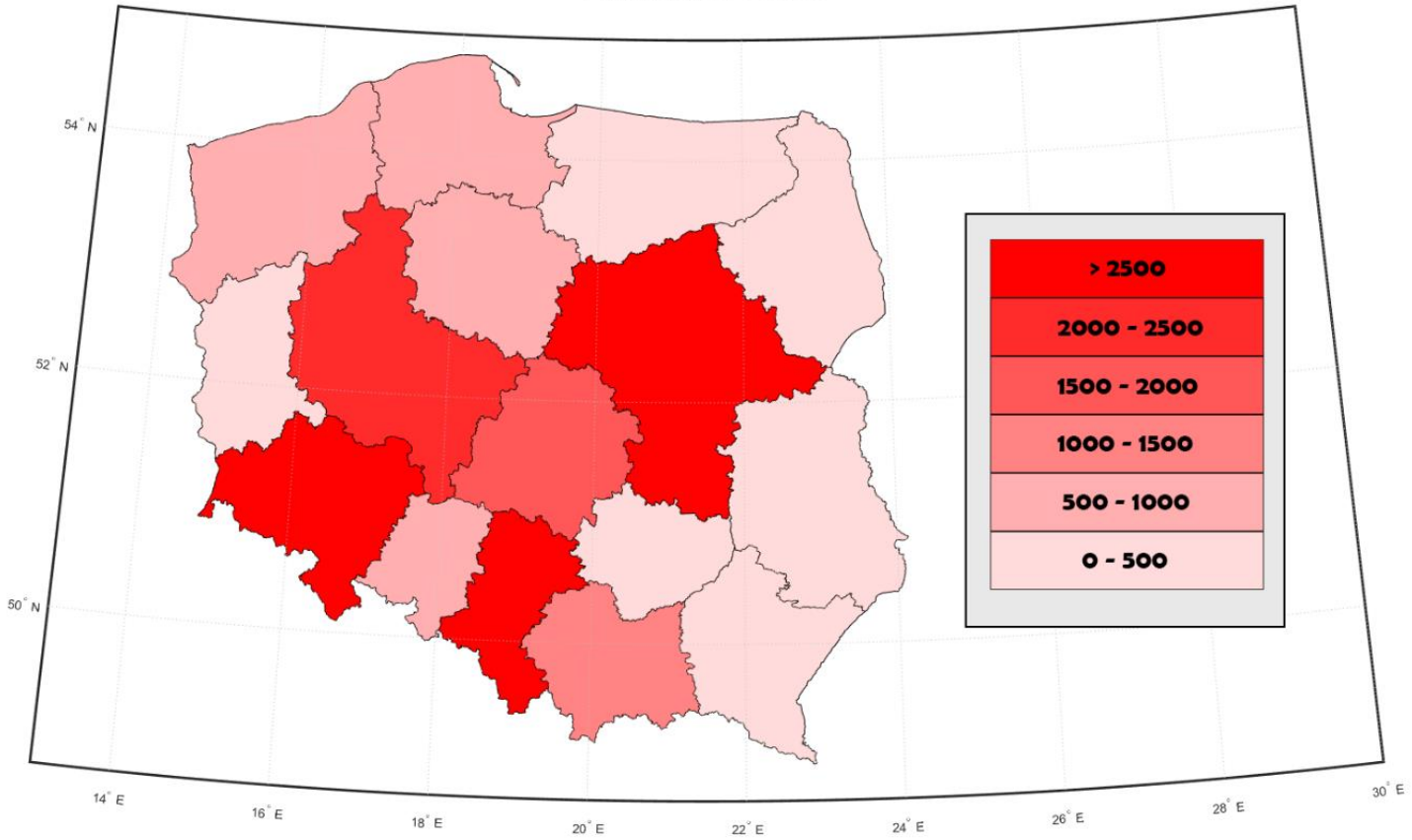

Liczba zakażeń na kontynentach świata na 1M mieszkańców
3273
2488
215
96
1524
267
60°N
30°N
0°
30°S
Latitude
135°W
90°W
45°W
0°
45°E
90°E
135°E
180°E
Longitude
2000 km
1000 mi



ANIMACJA (Obraz .gif)


Nowe przypadki w Polsce
Zakażenia
0
100
200
300
400
500
600
700
07 Mar
09 Mar
11 Mar
13 Mar
15 Mar
17 Mar
19 Mar
21 Mar
23 Mar
25 Mar
27 Mar
29 Mar
31 Mar
02 Apr
04 Apr
06 Apr
08 Apr
10 Apr
12 Apr
14 Apr
16 Apr
18 Apr
20 Apr
22 Apr
24 Apr
26 Apr
28 Apr
30 Apr
02 May
04 May
06 May
08 May
10 May
12 May
14 May
16 May
18 May
20 May
22 May
24 May
26 May
28 May
30 May
Data

Zakażenia w Polsce



Zgony w Polsce