BEBERAPA M-FILE MATLAB KOMPUTASI PENELITIAN INTERNAL.

1. Metode backpropagation standar dengan kriteria stopping batas atas toleransi

```
%network berukuran 1-2-1
%inisialisasi bobot dan bias:
N0 = 1; N1 = 2; N2 = 1;
W1 = zeros(N1,N0); W2 = zeros(N2,N1);b1 = zeros(N1,1); b2 = zeros(N2,1);
%inisialiasi bobot dan bias, biasanya diambil random
%W1 = [-0.27;-0.41];W2 = [0.09 -0.17];b1 = [-0.48;-0.13];  b2 = [0.48];
%W1 = rand(N1,N0); W2 = rand(N2,N1);b1 = rand(N1,1); b2 = rand(N2,1);
W1 = [0.7112;0.4011]; W2 = [0.1874 0.5455]; b1 = [0.9245;0.5629]; b2 = [0.9657];
%Initial
W10 = W1; W20 = W2; b10 = b1; b20 = b2;

%data training
input = -2:0.2:2; t = 1+sin(pi/4*input); %pasangan input-target
L =length(input);
%N = 300;%banyak iterasi
r = randi([1 L],N,1);%indeks sampel random
%load best_order.mat;
%load best_order1.mat;
%s1 = r(3); s2 = r(1); r(3) = s2;
E = []; %penyiapan penyimpanan error setiap iterasi
Et = 1; k = 0; tol = 0.0676;
while Et > tol
    k = k+1;
    p1 = input(r(k)); t1 = t(r(k));
    %output network pada setiap layer
    [n1,a1,n2,a2] = output(W1,b1,W2,b2,p1);
    e = t1 - a2;%error padalayer terakhir (output)

    %mulai menjalankan algoritma backpropagation
    s2 = -2*feval('dfun2',n2)*e; %sensitivity pada layer 2
    v = [feval('dfun1',n1(1)),feval('dfun1',n1(2))];
    F = diag(v);
    s1 = F*W2'*s2; %sensitivity pada layer 1
    %update bobot & bias
    grad2 = s2*a1';grad1 = s1*p1';
    sens2 = s2; sens1 = s1;
    alpha = 0.35;%ini learning rate optimal by trial and error
    W2 = W2 - alpha*grad2; b2 = b2 - alpha*sens2; %pada layer 2
    W1 = W1 - alpha*grad1; b1 = b1 - alpha*sens1; %pada layer 1
    W10 = [W10 W1]; W20 = [W20; W2]; b10 = [b10 b1]; b20 = [b20; b2];
    [n1,a1,n2,a2] = output(W1,b1,W2,b2,input);
    E0 = norm(a2-t); E = [E;E0];
    Et = E(end);
end

%forward
function [n1,a1,n2, a2] = output(W1,b1,W2,b2,p)
n1 = W1*p+b1; a1 = feval('fun1',n1);
n2 = W2*a1+b2; a2 = feval('fun2',n2);
end

%fungsi aktivasi & derivatif
function y = fun1(x)
y = logsig(x);
end

function dy = dfun1(x)
a = 1./(1+exp(-x)); dy = (1-a)*a;
end

function y = fun2(x)
y = purelin(x);
end

function dy = dfun2(x)
%a = 1./(1+exp(-x)); dy = (1-a)*a;
dy = 1;
end
```

2. Metode backpropagation standar dengan kriteria stopping banyak iterasi

```
%network berukuran 1-2-1
%inisialisasi bobot dan bias:
N0 = 1; N1 = 2; N2 = 1;
W1 = zeros(N1,N0); W2 = zeros(N2,N1);b1 = zeros(N1,1); b2 = zeros(N2,1);
%inisialiasi bobot dan bias, biasanya diambil random
%W1 = [-0.27;-0.41];W2 = [0.09 -0.17];b1 = [-0.48;-0.13];  b2 = [0.48];
W1 = rand(N1,N0); W2 = rand(N2,N1);b1 = rand(N1,1); b2 = rand(N2,1);
%Initial
W10 = W1; W20 = W2; b10 = b1; b20 = b2;

%data training
input = -2:0.2:2; t = 1+sin(pi/4*input); %pasangan input-target
L =length(input);
N = 200;%banyak iterasi
r = randi([1 L],N,1);%indeks sampel random
E = []; %penyiapan penyimpanan error setiap iterasi

for k = 1:N
    p1 = input(r(k)); t1 = t(r(k));
    %output network pada setiap layer
    [n1,a1,n2,a2] = output(W1,b1,W2,b2,p1);
    e = t1 - a2;%error padalayer terakhir (output)

    %mulai menjalankan algoritma backpropagation
    s2 = -2*feval('dfun2',n2)*e; %sensitivity pada layer 2
    v = [feval('dfun1',n1(1)),feval('dfun1',n1(2))];
    F = diag(v);
    s1 = F*W2'*s2; %sensitivity pada layer 1
    %update bobot & bias
    grad2 = s2*a1';grad1 = s1*p1';
    sens2 = s2; sens1 = s1;
    alpha = 0.35;
    W2 = W2 - alpha*grad2; b2 = b2 - alpha*sens2; %pada layer 2
    W1 = W1 - alpha*grad1; b1 = b1 - alpha*sens1; %pada layer 1
    [n1,a1,n2,a2] = output(W1,b1,W2,b2,input);
    E0 = norm(a2-t); E = [E;E0];

end


%forward
function [n1,a1,n2, a2] = output(W1,b1,W2,b2,p)
n1 = W1*p+b1; a1 = feval('fun1',n1);
n2 = W2*a1+b2; a2 = feval('fun2',n2);
end

%fungsi aktivasi & derivatif
function y = fun1(x)
y = logsig(x);
end

function dy = dfun1(x)
a = 1./(1+exp(-x)); dy = (1-a)*a;
end

function y = fun2(x)
y = purelin(x);
end

function dy = dfun2(x)
%a = 1./(1+exp(-x)); dy = (1-a)*a;
dy = 1;
end
```

3. Metode backpropagation dengan momentum (BPMO/MOBP), kriteria stopping banyak iterasi.

```
%network berukuran 1-2-1
%inisialisasi bobot dan bias:
N0 = 1; N1 = 2; N2 = 1;
%W1 = zeros(N1,N0); W2 = zeros(N2,N1);b1 = zeros(N1,1); b2 = zeros(N2,1);
%inisialiasi bobot dan bias, biasanya diambil random
%W1 = [-0.27;-0.41];W2 = [0.09 -0.17];b1 = [-0.48;-0.13];  b2 = [0.48];
%W1 = rand(N1,N0); W2 = rand(N2,N1);b1 = rand(N1,1); b2 = rand(N2,1);
W1 = [0.7112;0.4011]; W2 = [0.1874 0.5455]; b1 = [0.9245;0.5629]; b2 = [0.9657];
%Initial
W10 = W1; W20 = W2; b10 = b1; b20 = b2;

%data training
input = -2:0.2:2; t = 1+sin(pi/4*input); %pasangan input-target
L =length(input);
N = 200;%banyak iterasi
%r = randi([1 L],N,1);%indeks sampel random
load best_order.mat;
%load best_order1.mat;
E = [];
Wdelta1 = zeros(size(W1)); Wdelta2 = zeros(size(W2));
bdelta1 = zeros(size(b1)); bdelta2 = zeros(size(b2));
for k = 1:N
    p1 = input(r(k)); t1 = t(r(k));
    %output network pada setiap layer
    [n1,a1,n2,a2] = output(W1,b1,W2,b2,p1);
    e = t1 - a2;%error padalayer terakhir (output)

    %mulai menjalankan algoritma backpropagation
    s2 = -2*feval('dfun2',n2)*e; %sensitivity pada layer 2
    v = [feval('dfun1',n1(1)),feval('dfun1',n1(2))];
    F = diag(v);
    s1 = F*W2'*s2; %sensitivity pada layer 1
    %update bobot & bias
    grad2 = s2*a1';grad1 = s1*p1';
    sens2 = s2; sens1 = s1;
    %alpha = 0.5; gamma = 0.81;
    alpha = 0.35;
    Wdelta1 = gamma*Wdelta1 - (1-gamma)*alpha*grad1; Wdelta2 = gamma*Wdelta2 - (1-gamma)*alpha*grad2;
    bdelta1 = gamma*bdelta1 - (1-gamma)*alpha*sens1; bdelta2 = gamma*bdelta2 - (1-gamma)*alpha*sens2;
    W2 = W2 + Wdelta2; b2 = b2 + bdelta2; %pada layer 2
    W1 = W1 + Wdelta1; b1 = b1 + bdelta1; %pada layer 1
    [n1v,a1v,n2v,a2v] = output(W1,b1,W2,b2,input);
    E0 = norm(a2v-t); E = [E;E0];
end


%forward
function [n1,a1,n2, a2] = output(W1,b1,W2,b2,p)
n1 = W1*p+b1; a1 = feval('fun1',n1);
n2 = W2*a1+b2; a2 = feval('fun2',n2);
end

%fungsi aktivasi & derivatif
function y = fun1(x)
y = logsig(x);
end

function dy = dfun1(x)
a = 1./(1+exp(-x)); dy = (1-a)*a;
end

function y = fun2(x)
y = purelin(x);
end

function dy = dfun2(x)
%a = 1./(1+exp(-x)); dy = (1-a)*a;
dy = 1;
end
```

4. Metode backpropagation dengan momentum (BPMO/MOBP), menggunakan model batch

```
%inisialisasi bobot dan bias:
N0 = 1; N1 = 2; N2 = 1;
W1 = zeros(N1,N0); W2 = zeros(N2,N1);b1 = zeros(N1,1); b2 = zeros(N2,1);
%inisialiasi bobot dan bias, biasanya diambil random
%W1 = [-0.27;-0.41];W2 = [0.09 -0.17];b1 = [-0.48;-0.13];  b2 = [0.48];
W1 = rand(N1,N0); W2 = rand(N2,N1);b1 = rand(N1,1); b2 = rand(N2,1);
%data training
input = -2:0.2:2; t = 1+sin(pi/4*input); %pasangan input-target
%[n1,a1,n2,a2] = output(W1,b1,W2,b2,input);
%E0 = norm(a2-t);
%MSE = E0; %memory MSE untuk tiap iterasi
L =length(input);
N = 200;%banyak iterasi
E = [];
for k = 1:N
    grad1 = zeros(size(W1)); grad2 = zeros(size(W2));
    sens1 = zeros(size(b1)); sens2 = zeros(size(b2));
    G2 = zeros(L,2); G1 = zeros(2,L);
    S2 = zeros(1,L); S1 = zeros(2,L);
    for j = 1:L
        p1 = input(j); t1 = t(j);
        %output network pada setiap layer
        [n1,a1,n2,a2] = output(W1,b1,W2,b2,p1);
        e = t1 - a2;%error padalayer terakhir (output)
        %mulai menjalankan algoritma backpropagation
        s2 = -2*feval('dfun2',n2)*e; %sensitivity pada layer 2
        v = [feval('dfun1',n1(1)),feval('dfun1',n1(2))];
        F = diag(v);
        s1 = F*W2'*s2; %sensitivity pada layer 1
        %menghitung total gradien
        grad20 = s2*a1';grad10 = s1*p1';
        G2(j,:) = grad20; G1(:,j) = grad10;
        S2(j) = s2; S1(:,j) = s1;
        grad2 = grad2+grad20; grad1 = grad1+grad10;
        sens20 = s2; sens10 = s1;
        sens2 = sens2+s2; sens1 = sens1+s1;
    end
    %update bobot & bias
    grad2 = grad2/L; grad1 = grad1/L;
    sens2 = sens2/L; sens1 = sens1/L;
    alpha = 0.35;
    W2 = W2 - alpha*grad2; b2 = b2 - alpha*sens2; %pada layer 2
    W1 = W1 - alpha*grad1; b1 = b1 - alpha*sens1; %pada layer 1
    %menghitung MSE
    [n1,a1,n2,a2] = output(W1,b1,W2,b2,input);
    E0 = norm(a2-t); E = [E;E0];
end

%forward
function [n1,a1,n2, a2] = output(W1,b1,W2,b2,p)
n1 = W1*p+b1; a1 = feval('fun1',n1);
n2 = W2*a1+b2; a2 = feval('fun2',n2);
end

%fungsi aktivasi & derivatif
function y = fun1(x)
y = logsig(x);
end

function dy = dfun1(x)
a = 1./(1+exp(-x)); dy = (1-a)*a;
end

function y = fun2(x)
y = purelin(x);
end

function dy = dfun2(x)
%a = 1./(1+exp(-x)); dy = (1-a)*a;
dy = 1;
end
```

5. Metode backpropagation dengan Variable Learning Rate (VLBP)

```
%network berukuran 1-2-1 metode momentum dengan variabel learning rate
%inisialisasi bobot dan bias:
N0 = 1; N1 = 2; N2 = 1;
%W1 = zeros(N1,N0); W2 = zeros(N2,N1);b1 = zeros(N1,1); b2 = zeros(N2,1);
%inisialiasi bobot dan bias, biasanya diambil random
%W1 = [-0.27;-0.41];W2 = [0.09 -0.17];b1 = [-0.48;-0.13];  b2 = [0.48];
%W1 = rand(N1,N0); W2 = rand(N2,N1);b1 = rand(N1,1); b2 = rand(N2,1);
W1 = [0.7112;0.4011]; W2 = [0.1874 0.5455]; b1 = [0.9245;0.5629]; b2 = 0.9657;
%Initial
W10 = W1; W20 = W2; b10 = b1; b20 = b2;

%data training
input = -2:0.2:2; t = 1+sin(pi/4*input); %pasangan input-target
L =length(input);
N = 300;%banyak iterasi
r = randi([1 L],N,1);%indeks sampel random
%load best_order.mat;
%load best_order1.mat;
E = [];
Wdelta1 = zeros(size(W1)); Wdelta2 = zeros(size(W2));
bdelta1 = zeros(size(b1)); bdelta2 = zeros(size(b2));
%parameter learning for variable learning
nu = 1.05; %menaikkan rate
nd = 0.8; %menurunkan rate
xi = 0.04; %toleransi fluktuasi SE

%iterasi pertama dan kedua
for k = 1:2
    p1 = input(r(k)); t1 = t(r(k));
    %output network pada setiap layer
    [n1,a1,n2,a2] = output(W1,b1,W2,b2,p1);
    e = t1 - a2;%error padalayer terakhir (output)
    %mulai menjalankan algoritma backpropagation
    s2 = -2*feval('dfun2',n2)*e; %sensitivity pada layer 2
    v = [feval('dfun1',n1(1)),feval('dfun1',n1(2))];
    F = diag(v);
    s1 = F*W2'*s2; %sensitivity pada layer 1
    %update bobot & bias
    grad2 = s2*a1';grad1 = s1*p1';
    sens2 = s2; sens1 = s1;
    alpha0 = 0.5; gamma = 0.81;
    %alpha = 0.35;
    Wdelta1 = gamma*Wdelta1 - (1-gamma)*alpha0*grad1; Wdelta2 = gamma*Wdelta2 - (1-gamma)*alpha0*grad2;
    bdelta1 = gamma*bdelta1 - (1-gamma)*alpha0*sens1; bdelta2 = gamma*bdelta2 - (1-gamma)*alpha0*sens2;
    W2 = W2 + Wdelta2; b2 = b2 + bdelta2; %pada layer 2
    W1 = W1 + Wdelta1; b1 = b1 + bdelta1; %pada layer 1
    [n1v,a1v,n2v,a2v] = output(W1,b1,W2,b2,input);
    E0 = norm(a2v-t); E = [E;E0];
end
%%
e0 = E(1); e1 = E(2);
if e1 > e0
    if e1 > e0 + xi
        alpha = nd*alpha0;
        gamma = 0;
        E(end) = []; lr(end)=[];
        p1 = input(r(k-1)); t1 = t(r(k-1));
    else
        alpha = 0.5;
        gamma = 0.81;
        p1 = input(r(k)); t1 = t(r(k));
    end
else
    alpha = nu*alpha0;
    gamma = 0.81;
    p1 = input(r(k)); t1 = t(r(k));
end
lr = [alpha0;alpha];
[n1v,a1v,n2v,a2v] = output(W1,b1,W2,b2,input);
E0 = norm(a2v-t); E = [E;E0];
e0 = E(1); e1 = E(2);
```

```
%iterasi ke-3 dan seterusnya
for k = 3:N
    if e1 > e0
        if e1 > e0 + xi
            alpha = nd*alpha0;
            gamma = 0;
            E(end) = []; lr(end)=[];
            p1 = input(r(k-1)); t1 = t(r(k-1));
        else
            alpha = 0.5;
            gamma = 0.81;
            p1 = input(r(k)); t1 = t(r(k));
        end
    else
        alpha = nu*alpha0;
        gamma = 0.81;
        p1 = input(r(k)); t1 = t(r(k));
    end
    lr = [lr;alpha];
    %p1 = input(r(k)); t1 = t(r(k));
    %output network pada setiap layer
    [n1,a1,n2,a2] = output(W1,b1,W2,b2,p1);
    e = t1 - a2;%error padalayer terakhir (output)

    %mulai menjalankan algoritma backpropagation
    s2 = -2*feval('dfun2',n2)*e; %sensitivity pada layer 2
    v = [feval('dfun1',n1(1)),feval('dfun1',n1(2))];
    F = diag(v);
    s1 = F*W2'*s2; %sensitivity pada layer 1
    %update bobot & bias
    grad2 = s2*a1';grad1 = s1*p1';
    sens2 = s2; sens1 = s1;
    Wdelta1 = gamma*Wdelta1 - (1-gamma)*alpha*grad1; Wdelta2 = gamma*Wdelta2 - (1-gamma)*alpha*grad2;
    bdelta1 = gamma*bdelta1 - (1-gamma)*alpha*sens1; bdelta2 = gamma*bdelta2 - (1-gamma)*alpha*sens2;
    W2 = W2 + Wdelta2; b2 = b2 + bdelta2; %pada layer 2
    W1 = W1 + Wdelta1; b1 = b1 + bdelta1; %pada layer 1
    [n1v,a1v,n2v,a2v] = output(W1,b1,W2,b2,input);
    E0 = norm(a2v-t);
    E = [E;E0];
    e0 = E(end-1); e1 = E(end); %dua error terakhir
    alpha0 = alpha;
end


%forward
function [n1,a1,n2, a2] = output(W1,b1,W2,b2,p)
n1 = W1*p+b1; a1 = feval('fun1',n1);
n2 = W2*a1+b2; a2 = feval('fun2',n2);
end

%fungsi aktivasi & derivatif
function y = fun1(x)
y = logsig(x);
end

function dy = dfun1(x)
a = 1./(1+exp(-x)); dy = (1-a)*a;
end

function y = fun2(x)
y = purelin(x);
end

function dy = dfun2(x)
%a = 1./(1+exp(-x)); dy = (1-a)*a;
dy = 1;
end
```