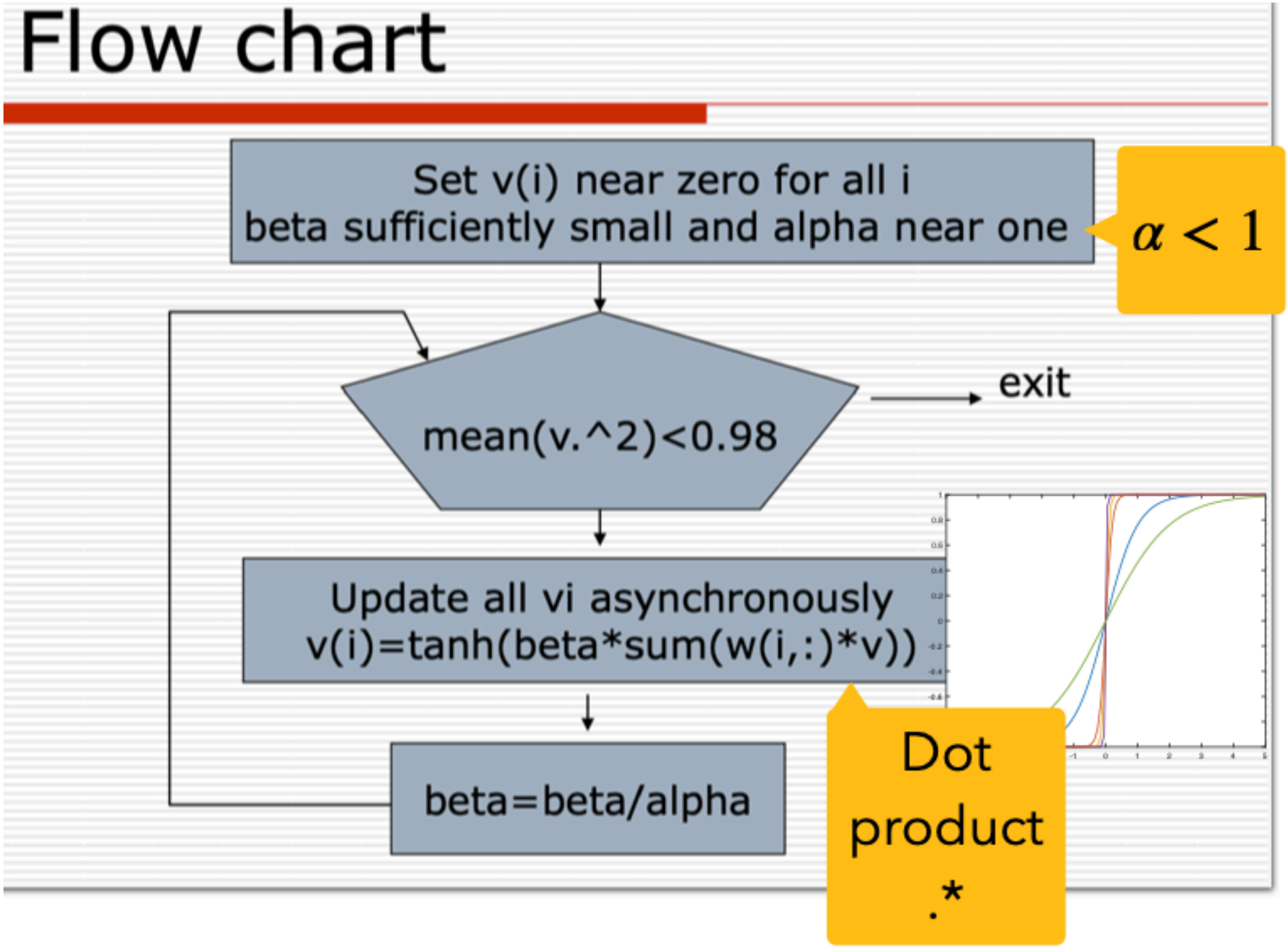
Matlab codes : Mean Field Annealing solving graph bisection

Flow chart



Set v(i) near zero for all i
beta sufficiently small and alpha near one

mean(v.^2)<0.98

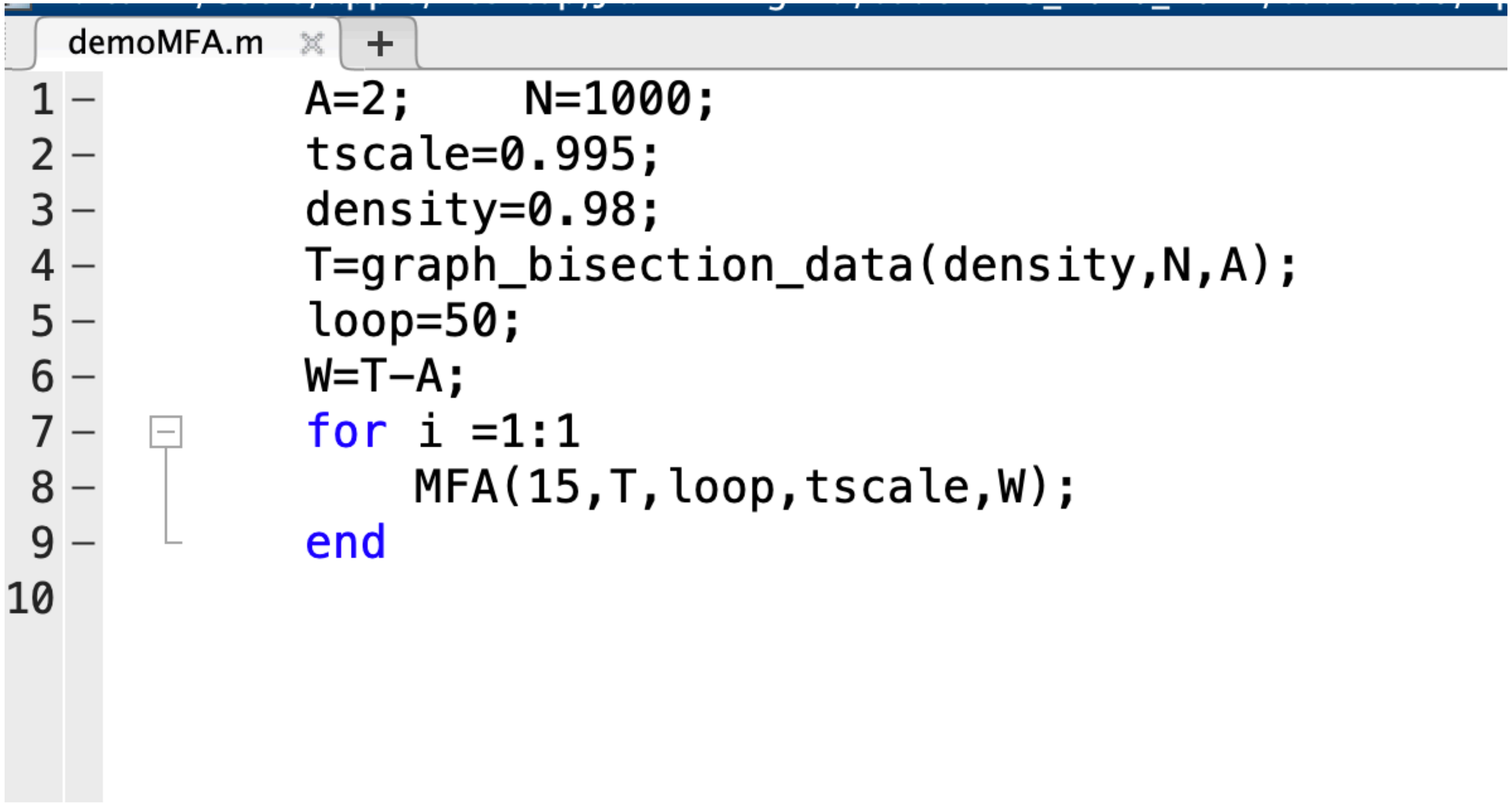
exit

Update all vi asynchronously
v(i)=tanh(beta*sum(w(i,:)*v))

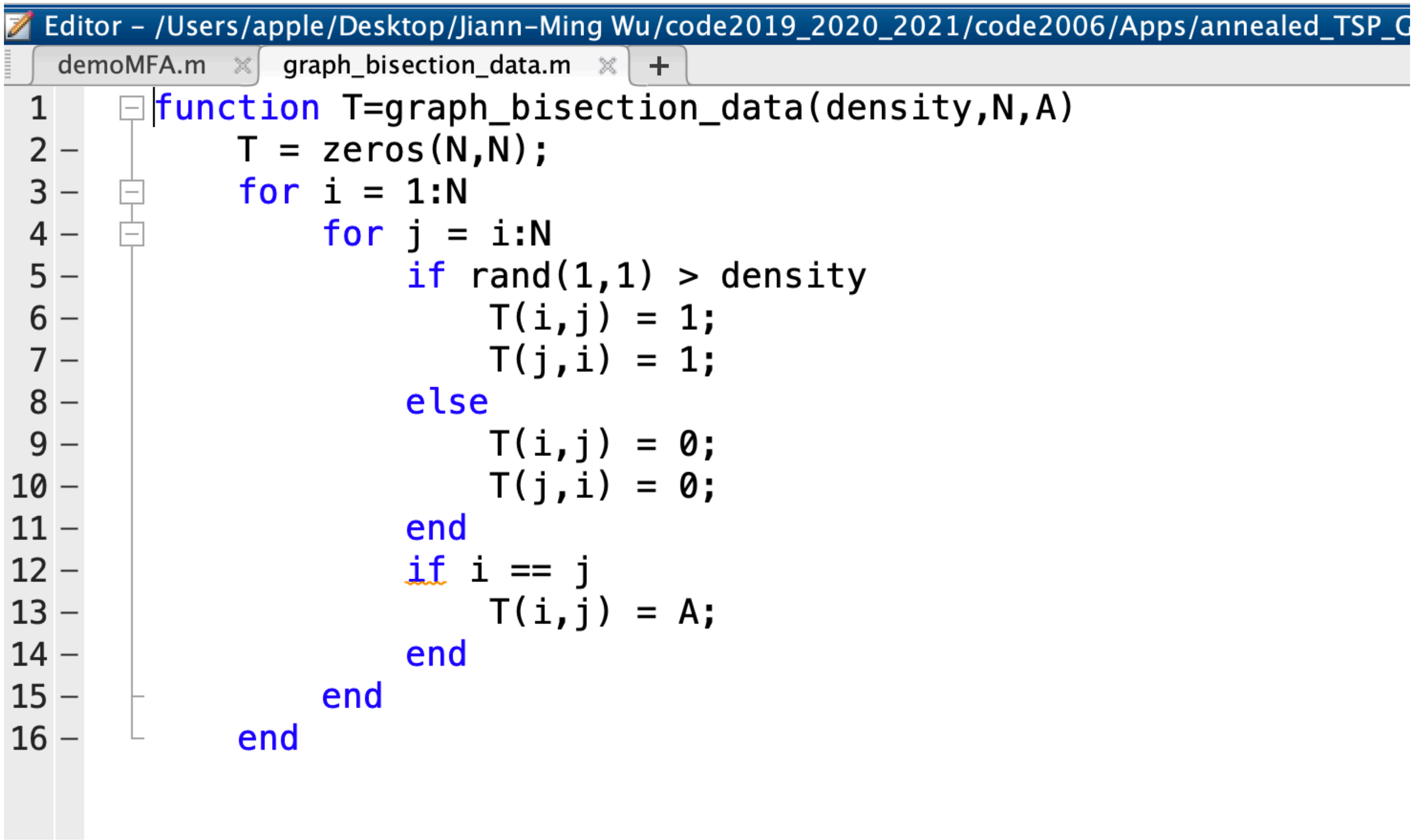
beta=beta/alpha

$\alpha < 1$

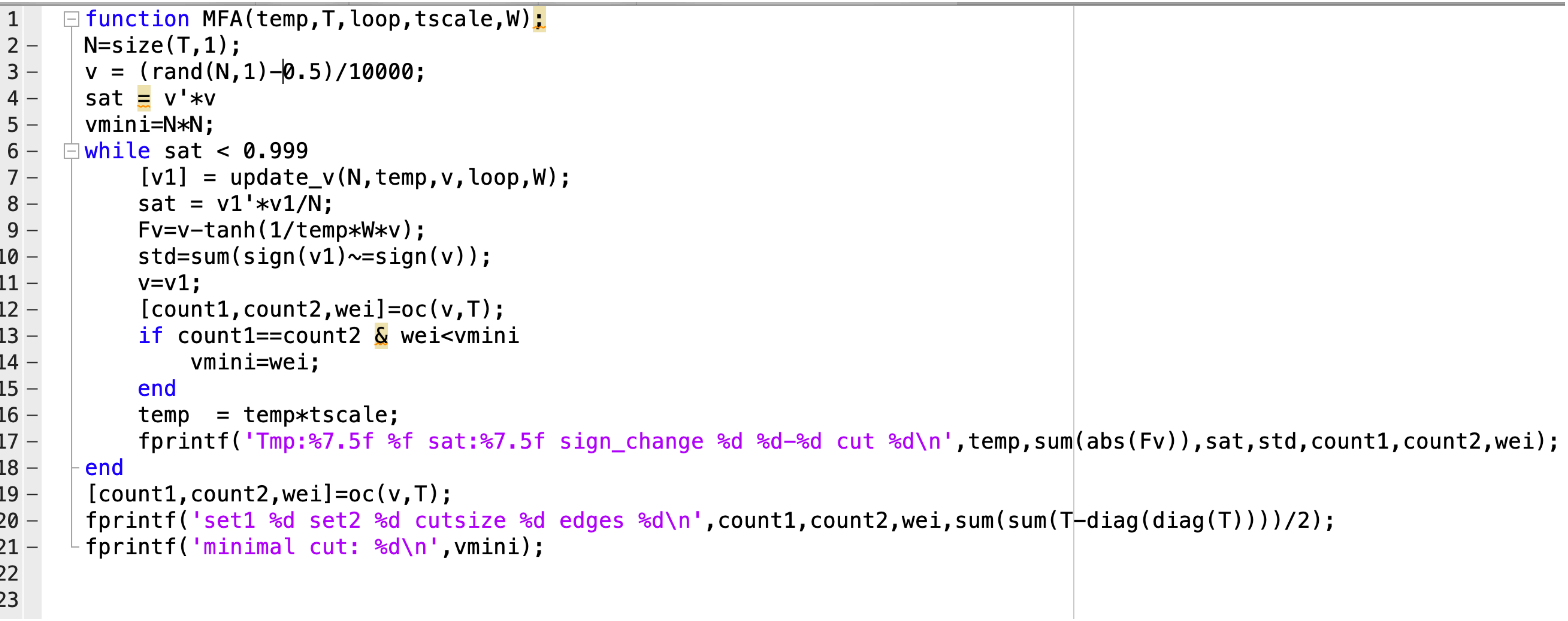
Dot product
.*



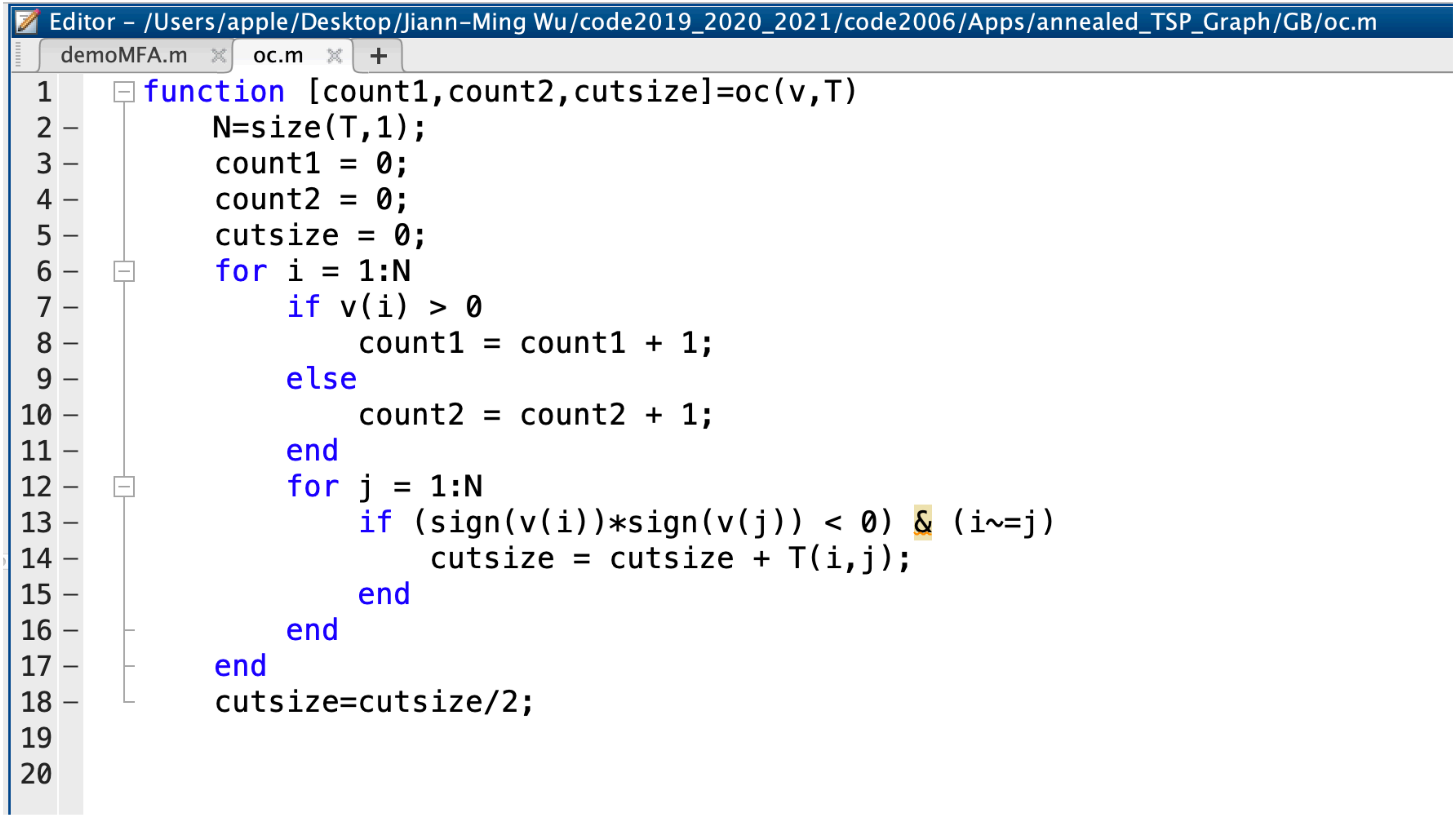
```
A=2;    N=1000;
tscale=0.995;
density=0.98;
T=graph_bisection_data(density,N,A);
loop=50;
W=T-A;
for i =1:1
    MFA(15,T,loop,tscale,W);
end
```


Editor – /Users/apple/Desktop/Jiann-Ming Wu/code2019_2020_2021/code2006/Apps/annealed_TSP_C
demoMFA.m
graph_bisection_data.m
+

```matlab
function T=graph_bisection_data(density,N,A)
    T = zeros(N,N);
    for i = 1:N
        for j = i:N
            if rand(1,1) > density
                T(i,j) = 1;
                T(j,i) = 1;
            else
                T(i,j) = 0;
                T(j,i) = 0;
            end
            if i == j
                T(i,j) = A;
            end
        end
    end
```

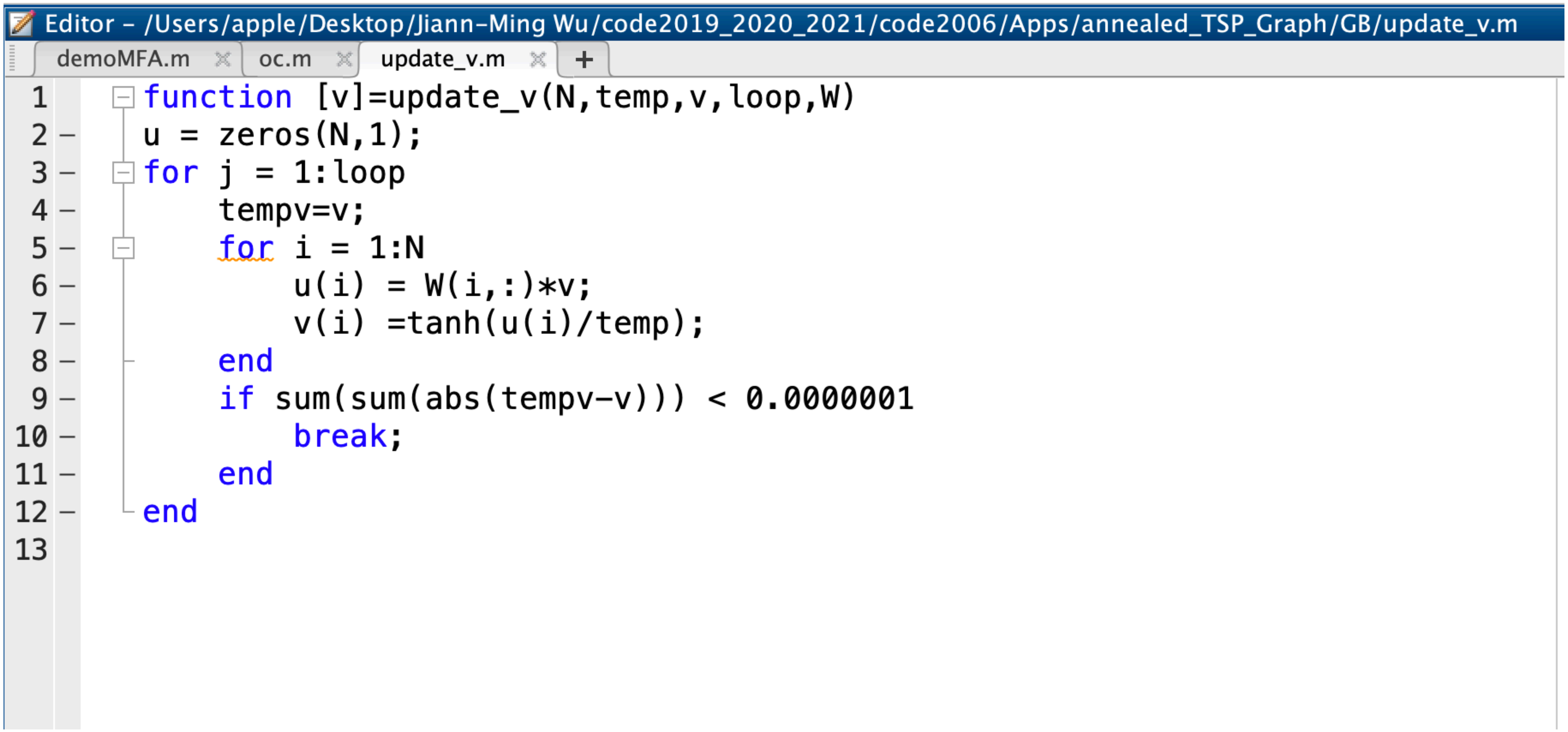


```
function MFA(temp,T,loop,tscale,W);
N=size(T,1);
v = (rand(N,1)-0.5)/10000;
sat = v'*v
vmini=N*N;
while sat < 0.999
    [v1] = update_v(N,temp,v,loop,W);
    sat = v1'*v1/N;
    Fv=v-tanh(1/temp*W*v);
    std=sum(sign(v1)~=sign(v));
    v=v1;
    [count1,count2,wei]=oc(v,T);
    if count1==count2 & wei<vmini
        vmini=wei;
    end
    temp  = temp*tscale;
    fprintf('Tmp:%7.5f %f sat:%7.5f sign_change %d %d-%d cut %d\n',temp,sum(abs(Fv)),sat,std,count1,count2,wei);
end
[count1,count2,wei]=oc(v,T);
fprintf('set1 %d set2 %d cutsize %d edges %d\n',count1,count2,wei,sum(sum(T-diag(diag(T))))/2);
fprintf('minimal cut: %d\n',vmini);
```

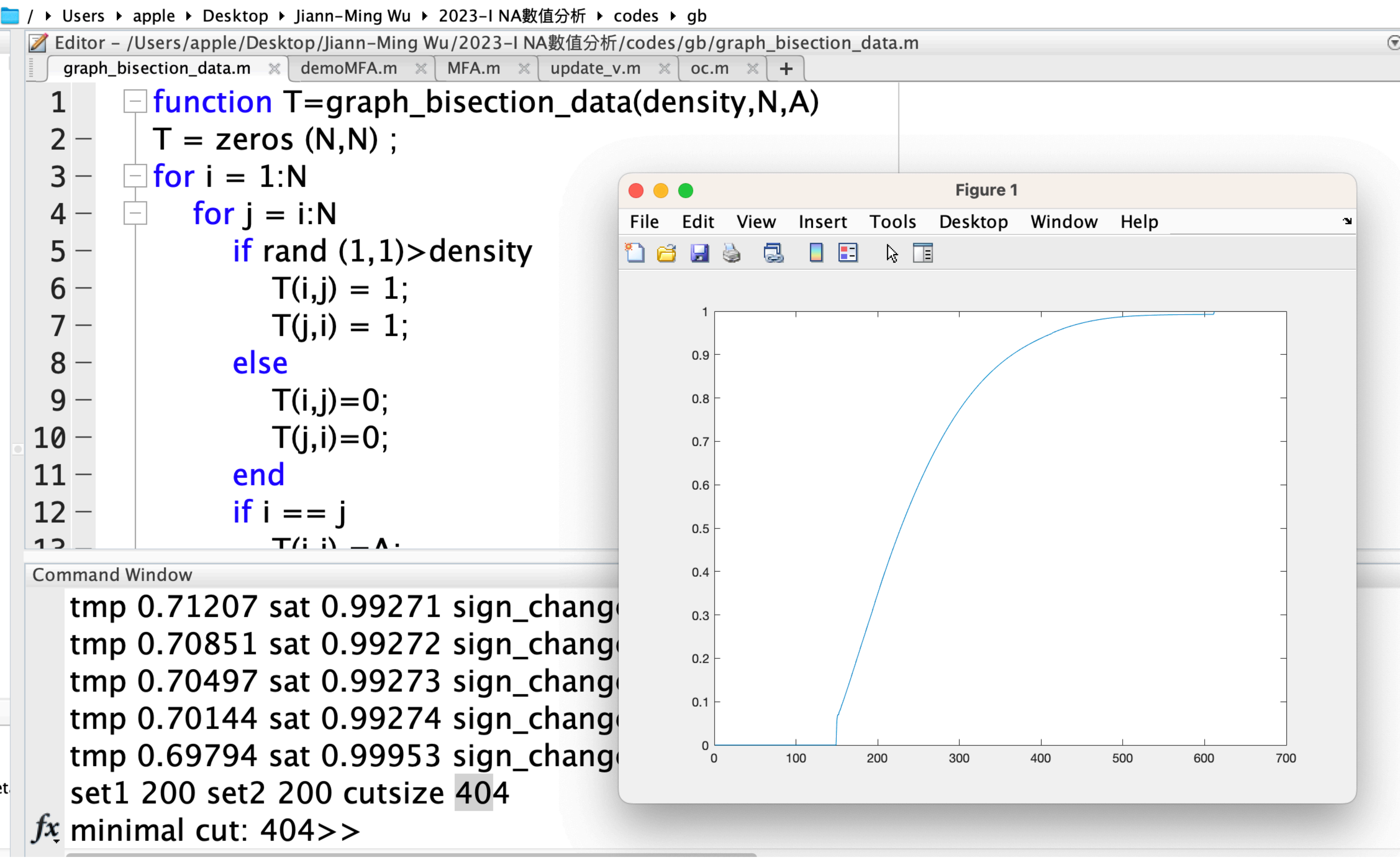

Editor – /Users/apple/Desktop/Jiann-Ming Wu/code2019_2020_2021/code2006/Apps/annealed_TSP_Graph/GB/oc.m
demoMFA.m
oc.m
+

```matlab
function [count1,count2,cutsize]=oc(v,T)
    N=size(T,1);
    count1 = 0;
    count2 = 0;
    cutsize = 0;
    for i = 1:N
        if v(i) > 0
            count1 = count1 + 1;
        else
            count2 = count2 + 1;
        end
        for j = 1:N
            if (sign(v(i))*sign(v(j)) < 0) & (i~=j)
                cutsize = cutsize + T(i,j);
            end
        end
    end
    cutsize=cutsize/2;


```


Editor – /Users/apple/Desktop/Jiann-Ming Wu/code2019_2020_2021/code2006/Apps/annealed_TSP_Graph/GB/update_v.m
demoMFA.m
oc.m
update_v.m
+

```matlab
function [v]=update_v(N,temp,v,loop,W)
u = zeros(N,1);
for j = 1:loop
    tempv=v;
    for i = 1:N
        u(i) = W(i,:)*v;
        v(i) =tanh(u(i)/temp);
    end
    if sum(sum(abs(tempv-v))) < 0.0000001
        break;
    end
end
```

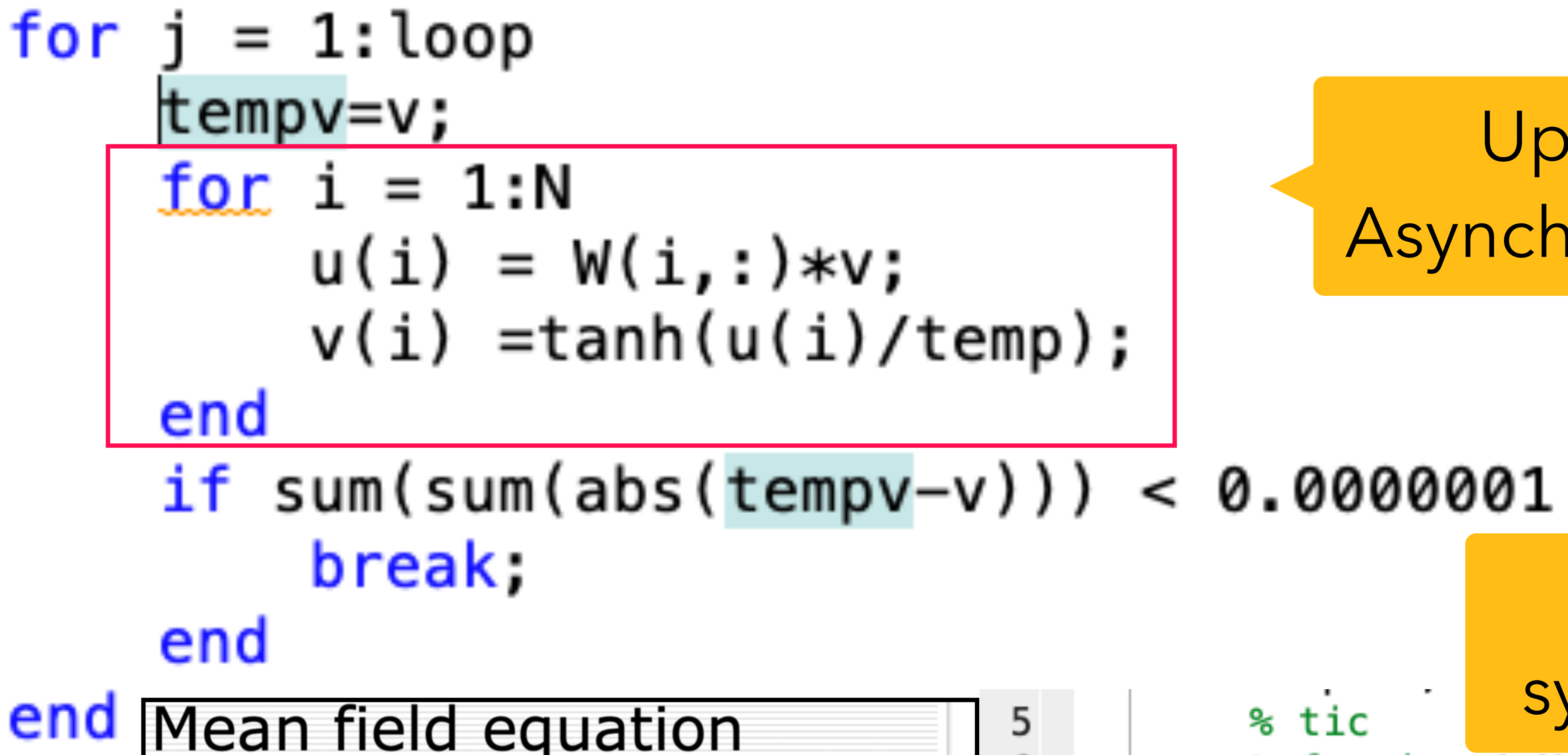


Editor – /Users/apple/Desktop/Jiann-Ming Wu/2023-I NA數值分析/codes/gb/graph_bisection_data.m

```
function T=graph_bisection_data(density,N,A)
T = zeros (N,N) ;
for i = 1:N
    for j = i:N
        if rand (1,1)>density
            T(i,j) = 1;
            T(j,i) = 1;
        else
            T(i,j)=0;
            T(j,i)=0;
        end
        if i == j
            T(i,i) =A;
```

Command Window

```
tmp 0.71207 sat 0.99271 sign_chang
tmp 0.70851 sat 0.99272 sign_chang
tmp 0.70497 sat 0.99273 sign_chang
tmp 0.70144 sat 0.99274 sign_chang
tmp 0.69794 sat 0.99953 sign_chang
set1 200 set2 200 cutsize 404
minimal cut: 404>>
```



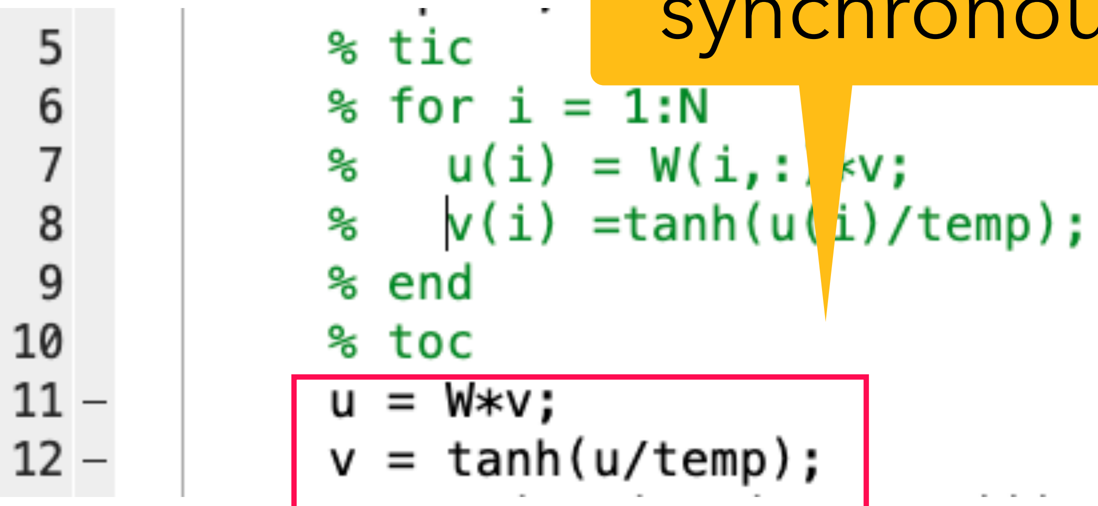
```
for j = 1:loop
    tempv=v;
    for i = 1:N
        u(i) = W(i,:)*v;
        v(i) =tanh(u(i)/temp);
    end
    if sum(sum(abs(tempv-v))) < 0.0000001
        break;
    end
end
```

Update Asynchronously

Mean field equation

$$v_i = \tanh(\beta \sum_{j \neq i}^{N} w_{ij} v_j)$$

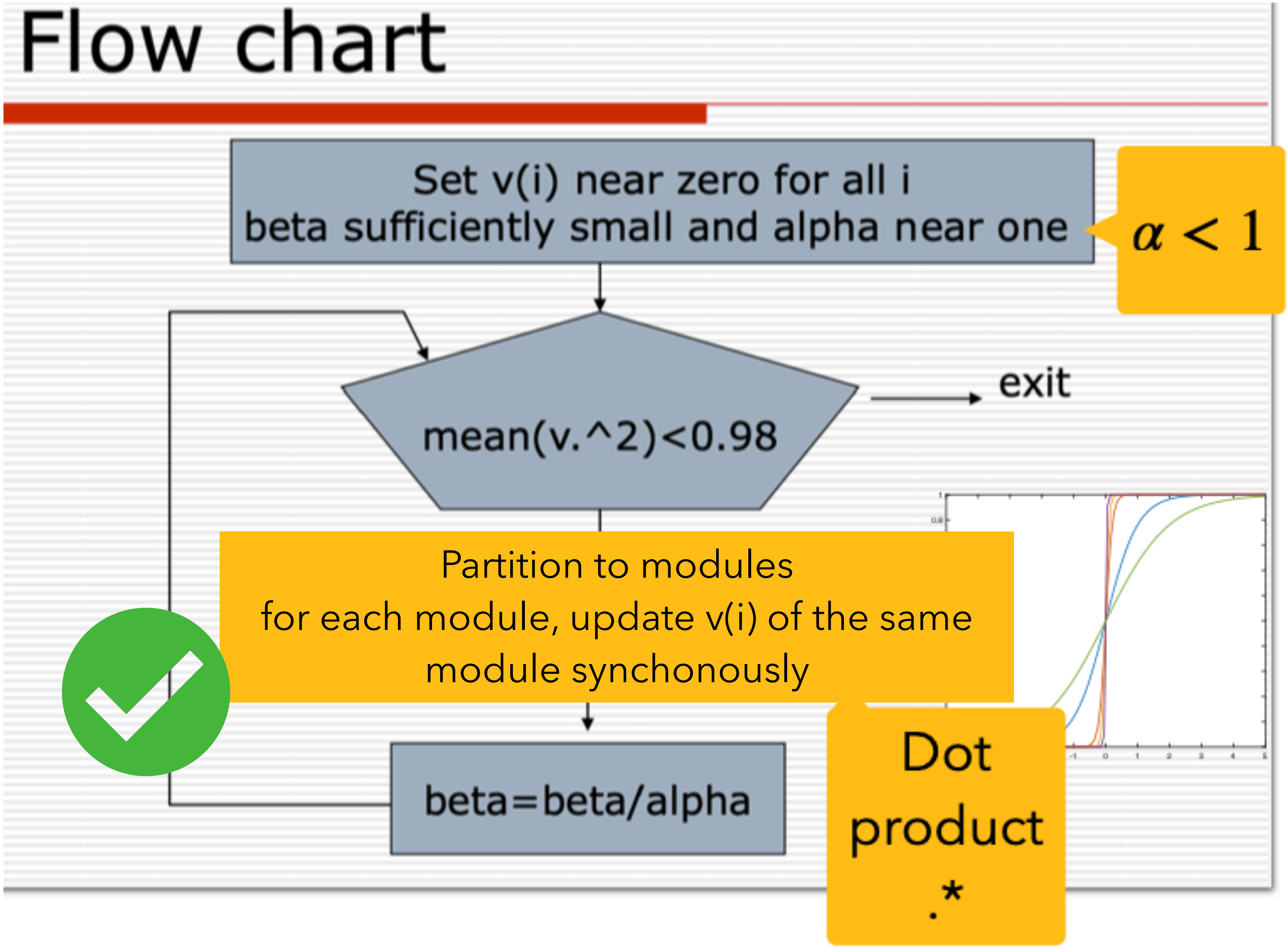
$$u_i = \sum_{j \neq i}^{N} w_{ij} v_j$$



```
% tic
% for i = 1:N
%   u(i) = W(i,:)*v;
%   v(i) =tanh(u(i)/temp);
% end
% toc
u = W*v;
v = tanh(u/temp);
```

Update synchronously

How to speed up the program? How to update neural activations at batch and asynchronously?



Flow chart

Set v(i) near zero for all i
beta sufficiently small and alpha near one

$\alpha < 1$

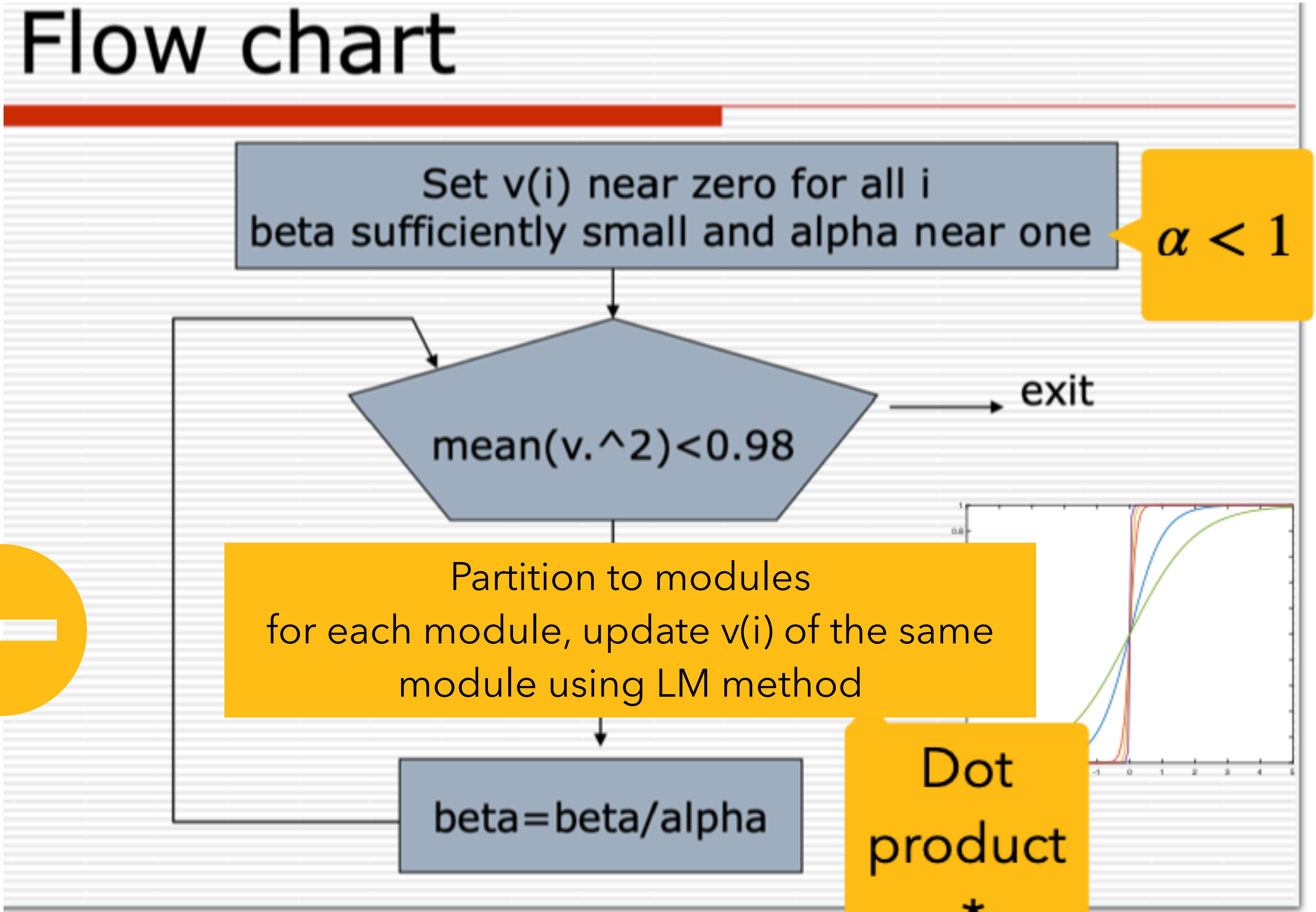
mean(v.^2)<0.98

exit

Partition to modules
for each module, update v(i) of the same
module synchonously

beta=beta/alpha

Dot product
.*



Flow chart

Set v(i) near zero for all i
beta sufficiently small and alpha near one

$\alpha < 1$

mean(v.^2)<0.98

exit

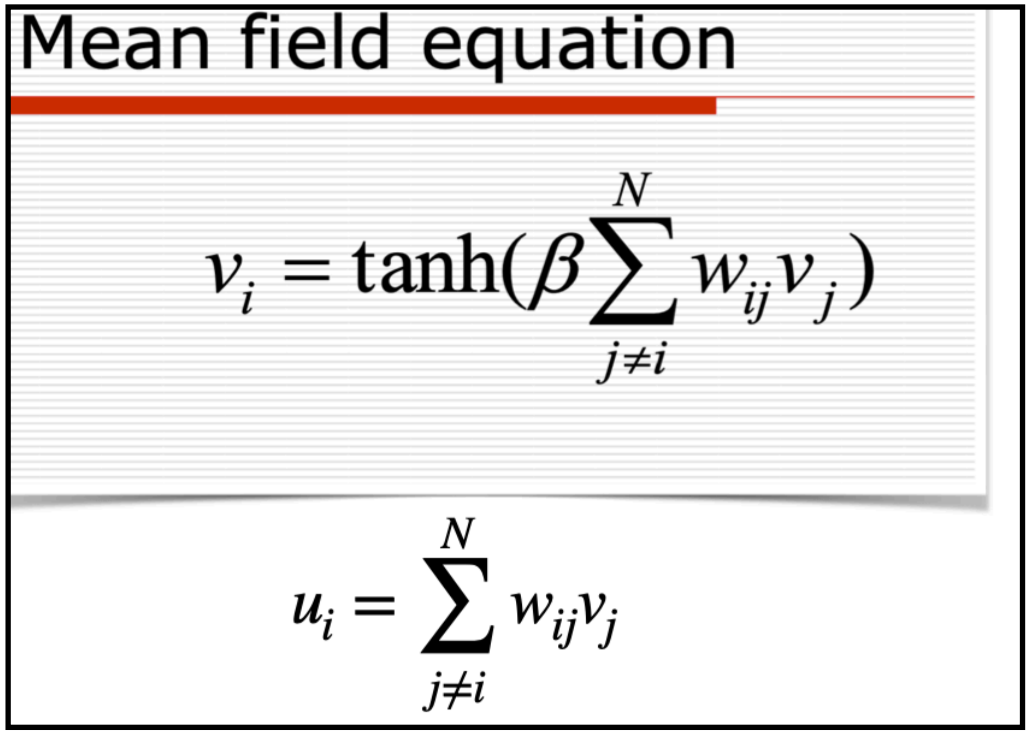
Partition to modules
for each module, update v(i) of the same
module using LM method

beta=beta/alpha

Dot
product
.*

LM coding

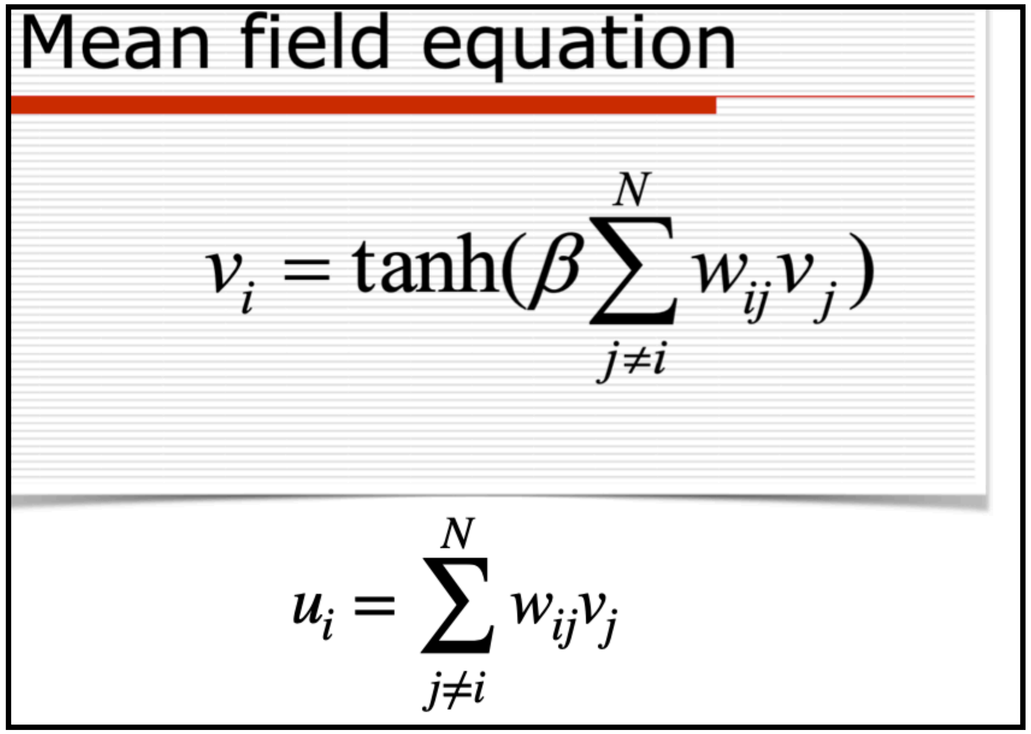
- Review Application of Matlab LM fsolve for learning MLP
- Apply Matlab LM fsolve to synchronously updating v(i) of the same module



Mean field equation

$$v_i = \tanh(\beta \sum_{j \neq i}^{N} w_{ij} v_j)$$

$$u_i = \sum_{j \neq i}^{N} w_{ij} v_j$$

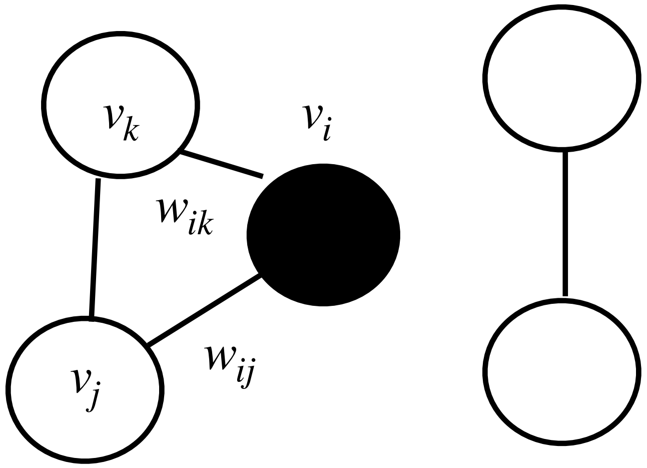


Mean field equation

$$v_i = \tanh(\beta \sum_{j \neq i}^{N} w_{ij} v_j)$$

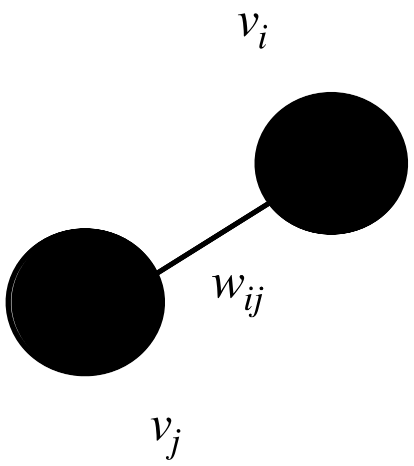
$$u_i = \sum_{j \neq i}^{N} w_{ij} v_j$$

Leave one out



asynchronous update

$u_i = \sum_{j \neq i} w_{ij} v_j$, Pr($s$) is proportional to $exp(\beta u_i)$



2004 IEEE SMC-B

$$u_i = \sum_{j \neq i} (T_{ij} - A) m_j + \frac{k}{\beta} \sum_{j \neq i} (v_{ij} - m_i m_j) m_j \quad (28)$$

$$m_i = \tanh(\beta u_i) \quad (29)$$

$$w_{ij} = \frac{1}{2}(T_{ij} - A) + \frac{k}{2\beta}(m_i m_j - v_{ij}) \quad (30)$$

$$v_{ij} = \tanh(\beta w_{ij}). \quad (31)$$