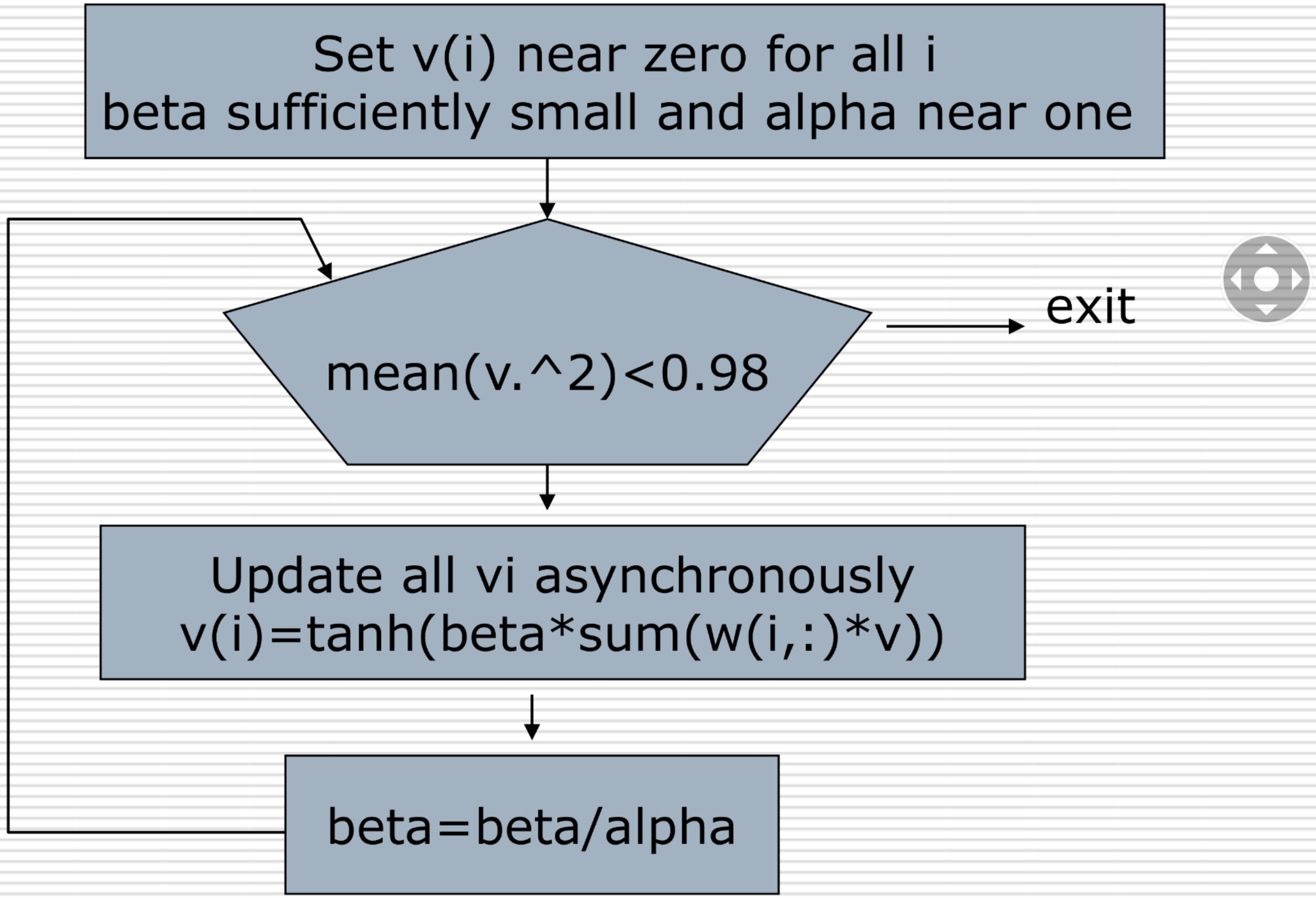
Synchronous recurrent neural networks for tracking mean field dynamics

Mean field equation

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

Flow chart


Set v(i) near zero for all i
beta sufficiently small and alpha near one
exit
mean(v.^2)<0.98
Update all vi asynchronously
v(i)=tanh(beta*sum(w(i,:)*v))
beta=beta/alpha

Guess x(0)
n=0

Not HC

EXIT

T

$$\mathbf{x}_{n+1} = \mathbf{x}_n - \left[J(\mathbf{x}_n)\right]^{-1} F(\mathbf{x}_n)$$

n= n+1

Asynchronous updating

http://134.208.26.59/AdvancedNA/MethodII/CombinatorialOptimization/methodII-1.pdf

- Data generation
- Demo script
- Mean field annealing
- Asynchronous updating
- Count

```
A=2;    N=1000;
tscale=0.9;
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
```

```
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
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;
    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 sat:%7.5f sign_change %d %d-%d cut %d\n',temp,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);
```

```
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
```

```
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;
```

Idea

- Consider mean field equations as nonlinear equations
- Apply the Newton method for seeking a solution
- Initial value
- Synchronous updating

$$\mathbf{x}_{n+1} = \mathbf{x}_n - \left[J(\mathbf{x})\right]^{-1} F(\mathbf{x}_n)$$

$$J(\mathbf{x}) = \begin{bmatrix} \dfrac{\partial f_1(\mathbf{x})}{\partial x_1} & \dfrac{\partial f_1(\mathbf{x})}{\partial x_2} & \cdots & \dfrac{\partial f_1(\mathbf{x})}{\partial x_n} \\ \dfrac{\partial f_2(\mathbf{x})}{\partial x_1} & \dfrac{\partial f_2(\mathbf{x})}{\partial x_2} & \cdots & \dfrac{\partial f_2(\mathbf{x})}{\partial x_n} \\ \vdots & \vdots & \vdots & \vdots \\ \dfrac{\partial f_n(\mathbf{x})}{\partial x_1} & \dfrac{\partial f_n(\mathbf{x})}{\partial x_2} & \cdots & \dfrac{\partial f_n(\mathbf{x})}{\partial x_n} \end{bmatrix}$$

Mean field equation

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

Symbolic differentiation

```
s=sym('tanh(x+y)+tanh(x-y)')
y=sym('y')
x=sym('x')
dfdx=diff(s,x);
dfdy=diff(s,y);
```

Numerical differentiation

Richardson extrapolation

Richardson extrapolation is with $O(h^3)$

Start at the formula that is with $O(h^2)$

$$f'(x) = \underbrace{\frac{f(x+h) - f(x-h)}{2h}}_{\equiv \phi(h)} + a_2 h^2 + a_4 h^4 + a_6 h^6 + \cdots$$

Strategy: elimination of h^2 term

Richardson extrapolation

- Divide by 3 and write $f'(x)$

$$
\begin{aligned}
f'(x) &= \frac{4}{3}\phi\left(\frac{h}{2}\right) - \frac{1}{3}\phi(h) - \frac{1}{4}a_4 h^4 - \frac{5}{16}a_6 h^6 - \cdots \\
&= \phi\left(\frac{h}{2}\right) + \underbrace{\frac{1}{3}\left[\phi\left(\frac{h}{2}\right) - \phi(h)\right]}_{\equiv(*)} + O\left(h^4\right)
\end{aligned}
$$

```
function value=fx(x,b,w,v)
value=x-tanh(b*(w'*v))
function d=phi(h,x,b,w,v)
function didj
function didi
```

$$f_{\hat{i}} = v_{\hat{i}} - \tanh(\beta w_{\hat{i}}^{T} v)$$

```
>> dsdu=diff(s,u)

dsdu =

2*b*(tanh(2*b*u)^2 - 1)
```

$$\frac{df_{\hat{i}}}{dv_{\hat{j}}} = 1$$

$$\frac{df_{\hat{i}}}{dv_{\hat{j}}} = \beta w_{\hat{i}\hat{j}} \left(\tanh\left(\beta w_{\hat{i}}^{T} v\right)^2 - 1\right)$$