확률적 두뇌의 이해

**Rust’s Experiment Implementation with MATLAB**

2014-22449

**Dong Hyun Kwak**

**Method**

1. **데이터 로딩 및 Spike Triggered Ensemble (n\*256 행렬)생성**

load('C:\Users\user\Documents\MATLAB\DATA\_set1.mat')

STE = zeros(sum(Data.vectSpike), 256);

indices = find(Data.vectSpike == 1)';

for data\_index=1:length(indices)

 i = indices(data\_index);

 STE(data\_index,:) = reshape(Data.matST(i:(i+15), :)', 1, []);

end

STA=mean(STE,1)

for i=1:length(indices)

 PROJECTION(i,:)=(dot(STE(i,:),STA)/norm(STA))\*(STA/norm(STA));

 SHIFT(i,:)=STE(i,:)-PROJECTION(i,:);

end

COVar=cov(SHIFT);

[EigVec, EigVal] = eig(COVar)

1. **임의의 Stimulus를 500회 생성하고 maximum eigenvalue와 minimum eigenvalue의 분포를 구하여, 99% 이상의 값을 excitatory threshold와 suppressive threshold를 구함.**

iteration = 500;

minEigVal=zeros(iteration,1);

maxEigVal=zeros(iteration,1);

for count=1:iteration

 Shuff\_vectSpike=Data.vectSpike(randperm(length(Data.vectSpike)));

 STE\_rand = zeros(sum(Shuff\_vectSpike), 256);

 indices = find(Shuff\_vectSpike == 1)';

 for data\_index=1:length(indices)

 i = indices(data\_index);

 STE\_rand(data\_index,:) = reshape(Data.matST(i:(i+15), :)', 1, []);

 end

 STA\_rand=mean(STE\_rand,1);

 for i=1:length(indices)

 PROJECTION\_rand(i,:)=(dot(STE\_rand(i,:),STA\_rand)/norm(STA\_rand))\*(STA\_rand/norm(STA\_rand));

 SHIFT\_rand(i,:)=STE\_rand(i,:)-PROJECTION\_rand(i,:);

 end

 COVar\_rand=cov(SHIFT\_rand);

 [EigVec\_rand, EigVal\_rand] = eig(COVar\_rand);

 minEigVal(count)=EigVal\_rand(2,2);

 maxEigVal(count)=EigVal\_rand(256,256);

 count

end

Exit\_thres=icdf('Normal',[0.99],mean(maxEigVal),std(maxEigVal));

Supp\_thres=icdf('Normal',[0.01],mean(minEigVal),std(minEigVal));

1. **Threshold를 이용하여 excitatory filter와 suppressive filter를 구한 뒤, STA와 함께 이들을 visualization 함.**

x=find(diag(EigVal)>Exit\_thres);

Exit\_filter = EigVec(:,x);

x=find(diag(EigVal)<Supp\_thres);

Supp\_filter = EigVec(:,x(x>1));

for visual=1:size(Exit\_filter,2)

 figure;

 imagesc(reshape(Exit\_filter(:,visual),16,16)');

 colormap(gray);

 title(['Excitatory\_filter\_' num2str(visual)]);

end

for visual=1:size(Supp\_filter,2)

 figure;

 imagesc(reshape(Supp\_filter(:,visual),16,16)');

 colormap(gray);

 title(['Suppressive\_filter\_' num2str(visual)]);

end

figure;

imagesc(reshape(STA,16,16))

title('STA');

colormap(gray);

**Results**

1. **DATA\_set1.mat의 실험 결과**

(왼쪽 위부터 순서대로 STA / excitatory filter 1, 2, 3 / suppressive filter 1 ,2, 3 )



1. **DATA\_set2.mat의 실험 결과**

( 왼쪽 위부터 순서대로 STA / excitatory filter 1, 2 / suppressive filter 1 ,2, 3 )



**Discussion**

데이터 1번은 STA와 STC에서 모두 유의미한 방향성을 가진 filter가 발견되었다. 그러나 데이터 2번에서는 STA분석을 통해서는 어떠한 정보도 알아낼 수 없었으나, STC를 통해서는 filter를 발견할 수 있었다. 따라서 1번 데이터는 simple cell로 보이고, 2번 데이터는 complex cell로 보인다.

다만, 실험 결과에서 1번 데이터의 excitatory filter 1, suppressive filter 3 그리고 2번 데이터의 suppressive filter 3은 제대로 된 필터가 분석되지 않았다. 어떠한 이유에서인지 필터를 제대로 캐치하지 못하였는데 이에 대한 원인이 코드상의 버그인지, 데이터 자체의 원인인 지 불분명하다.