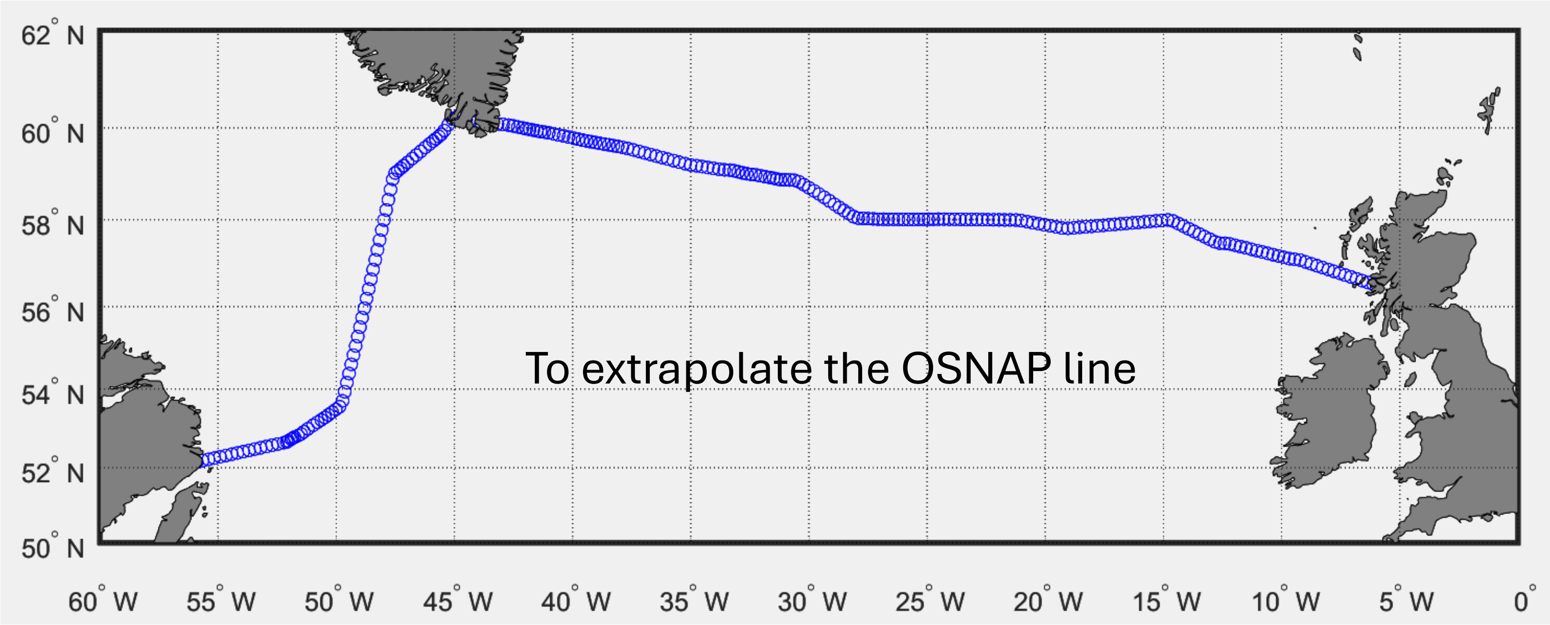
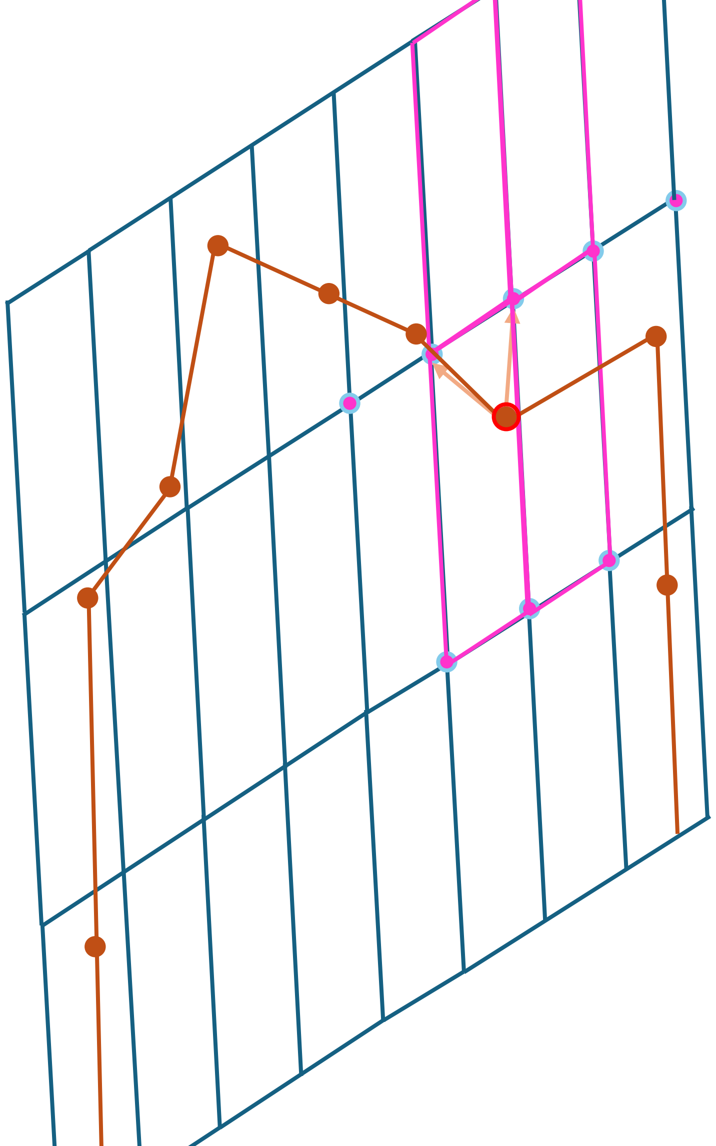
Defining OSNAP transect in POP coordinates (1/10)

OSNAP line

- POP grid is not regular
- Size of the cells varies





- find all the cells crossed by the OSNAP line

```
% loop to select cells in which point of the crossection is located
cellN=[];lonC=[];latC=[];
for n1=1:length(rlat)
    dy=111303;%length of a degree in meters
    dx=dy*cosd(rlat(n1));%length of a degree in meters
```

Points around

```
    % slliding window to define coordinate points around the target point
    ind=find((ulat>rlat(n1)-0.2&ulat<rlat(n1)+0.2)&(ulon>rlon(n1)-0.2...
        &ulon<rlon(n1)+0.2));
    %Find minimum distances between selected points and
    % target point of the crossection
```

The shortest distance

```
    dlon=ulon(ind)-rlon(n1);
    dlat=ulat(ind)-rlat(n1);
    dlon=dlon*dx; dlat=dlat*dy;
    D=sqrt(dlon.^2+dlat.^2);
    [val,n]=min(D); % location of the nearest point
```

The nearest cells

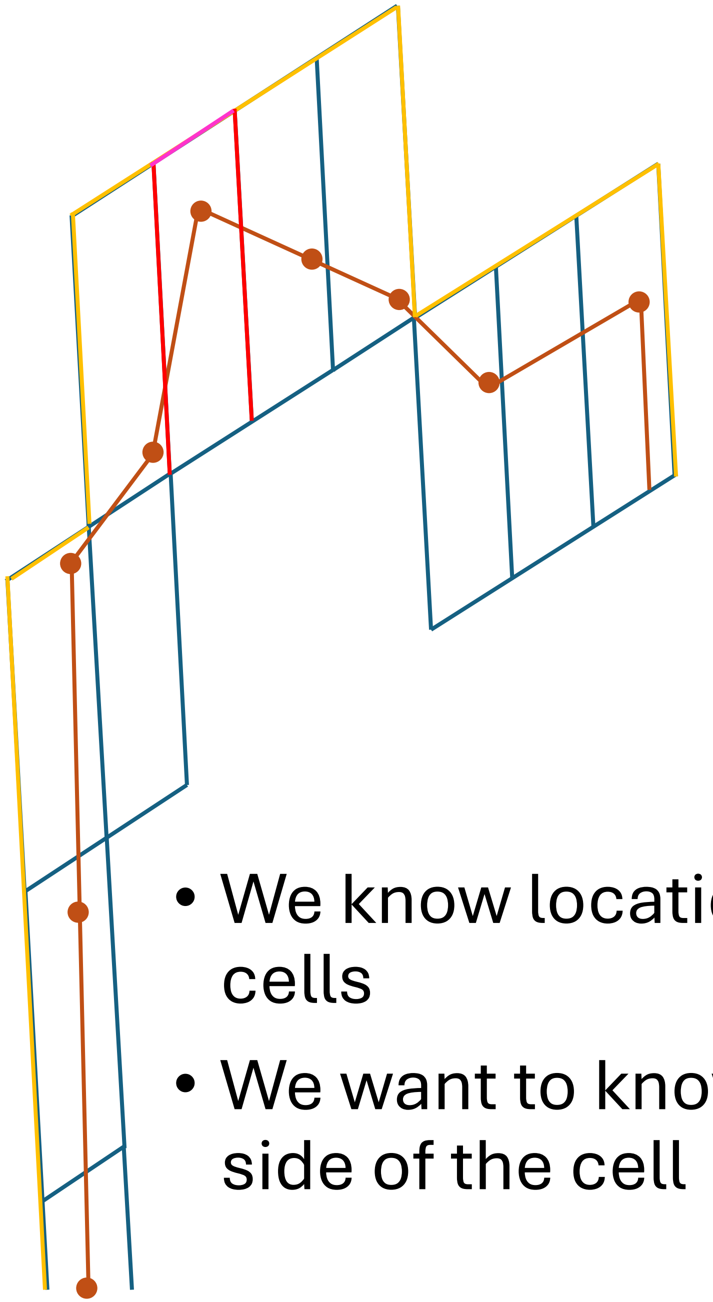
```
    [r,c]=ind2sub(size(ulat),ind(n)); % find the grid points around
    cell1=[[r-1,c-1];[r-1,c];[r,c];[r,c-1]]; % set of cells around the nearest point
    cell2=[[r-1,c];[r-1,c+1];[r,c+1];[r,c]];
    cell3=[[r,c-1];[r,c];[r+1,c];[r+1,c-1]];
    cell4=[[r,c];[r,c+1];[r+1,c+1];[r+1,c]];
    celln=cat(3,cell1,cell2,cell3,cell4);% cell indexes of 4 nearest cells
```

The exact cell where the point is

```
    for n3=1:size(celln,3)% the cell  where exactly the point is located
        for n4=1:size(celln,1)
            latt(n4)=ulat(celln(n4,1,n3),celln(n4,2,n3));
            lont(n4)=ulon(celln(n4,1,n3),celln(n4,2,n3));
        end
        a=inpolygon(rlon(n1),rlat(n1),lont,latt);
        if a==1
            break %when the point is found inside of any grid cell
        end
    end
```

output

```
    cellN=cat(3,cellN,celln(:,:,n3));% indexes of corners of the cell
    latC=cat(1,latC,latt);% geocoordinatess of the cell corners
    lonC=cat(1,lonC,lont);
end
```



- We know location of the cells
- We want to know open side of the cell

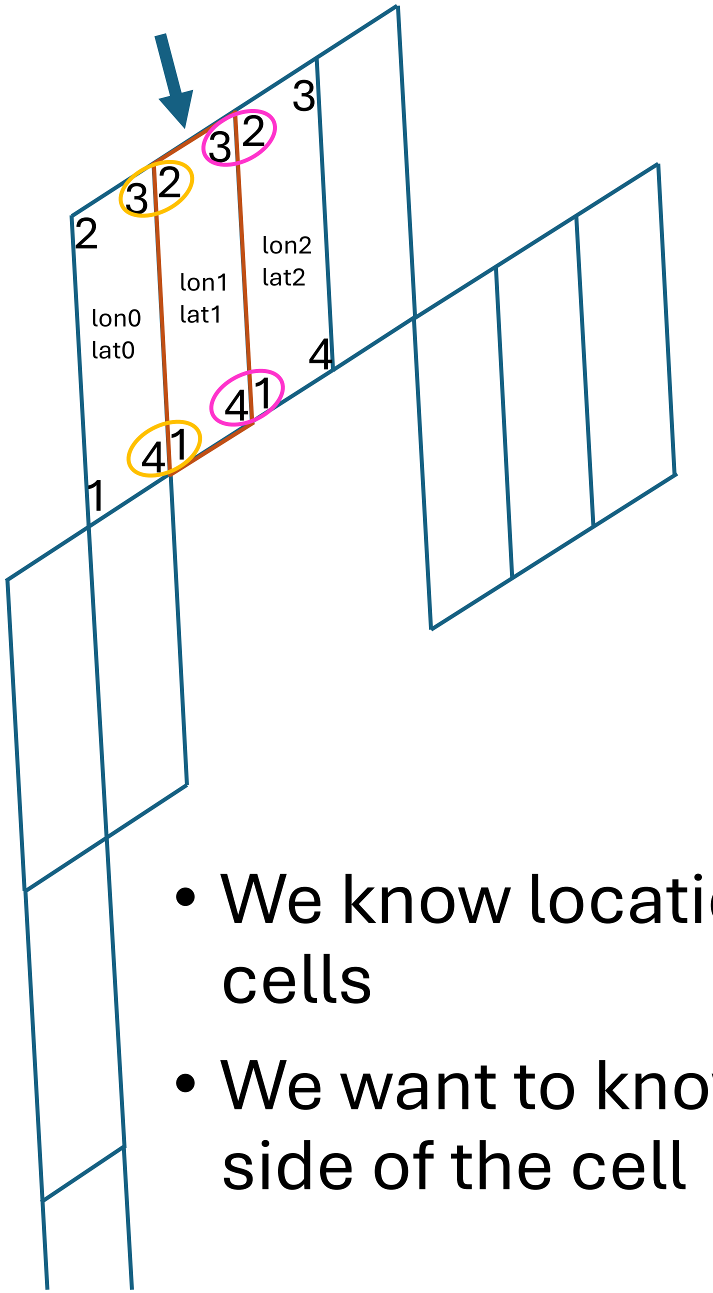
```
Atr=[];
for n1=1:length(LonC)-1
    lon1=LonC(n1,:);
    lat1=LatC(n1,:);
    lon2=LonC(n1+1,:);
    lat2=LatC(n1+1,:);
    if n1~=1
        lon0=LonC(n1-1,:);
        lat0=LatC(n1-1,:);
    end
    ind1=[]; % current and next
    ind2=[]; % current and previous
    for n2=1:length(lon1)
        if n1~=1
            ind1=[ind1,find((lon1==lon2(n2))&(lat1==lat2(n2)))];
            ind2=[ind2,find((lon1==lon0(n2))&(lat1==lat0(n2)))];
        else
            ind1=[ind1,find((lon1==lon2(n2))&(lat1==lat2(n2)))];
        end
    end
```

Step n

Step n+1

Step n-1

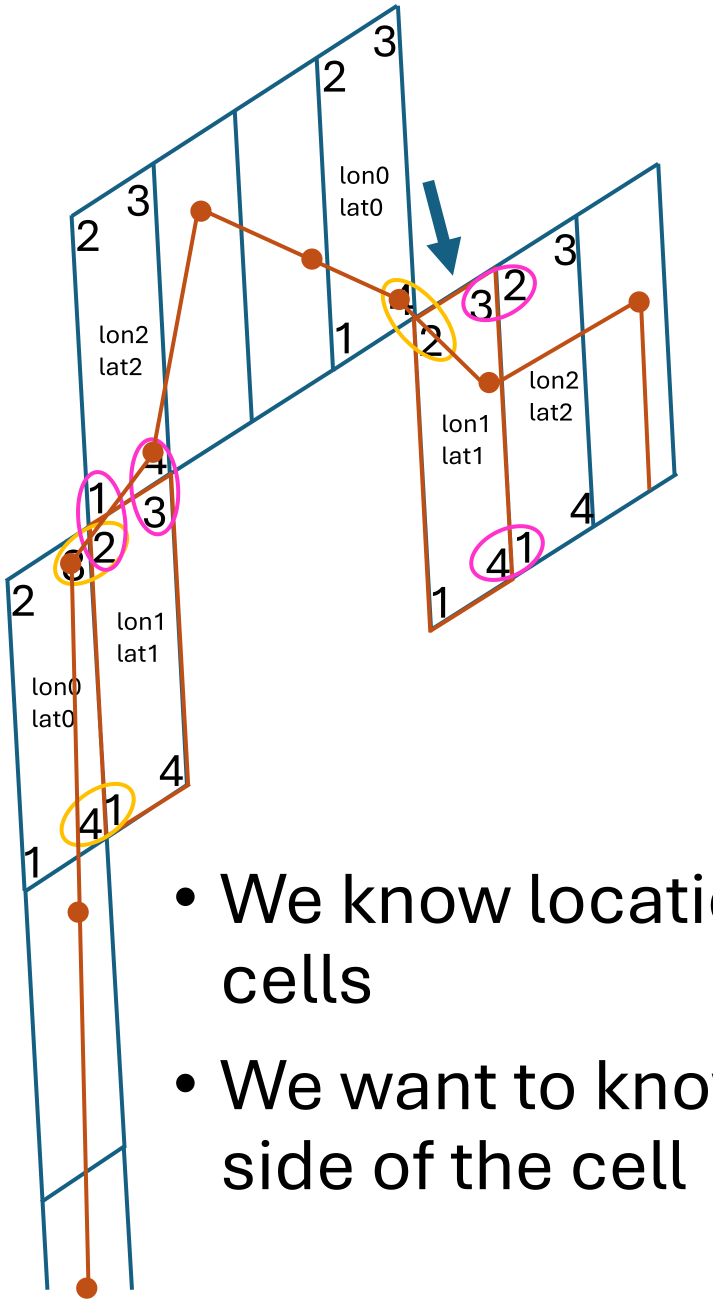
Here we define what points of the cell are on the borders, literally what face of the cell is open




```
for n2=1:length(lon1)
    if n1~=1
        ind1=[ind1,find((lon1==lon2(n2))&(lat1==lat2(n2)))];
        ind2=[ind2,find((lon1==lon0(n2))&(lat1==lat0(n2)))];
    else
        ind1=[ind1,find((lon1==lon2(n2))&(lat1==lat2(n2)))];
    end
end

% atribute 1=north
if (isempty(ind2)& ind1==[4,3])|(ind2==[2,1]&ind1==[4,3])|...
        (ind2==[2,1]&ind1==[3])|(ind2==[2]&ind1==[4,3])
    Atr(n1)=1;
    %          n1
end
```

- We know location of the cells
- We want to know open side of the cell

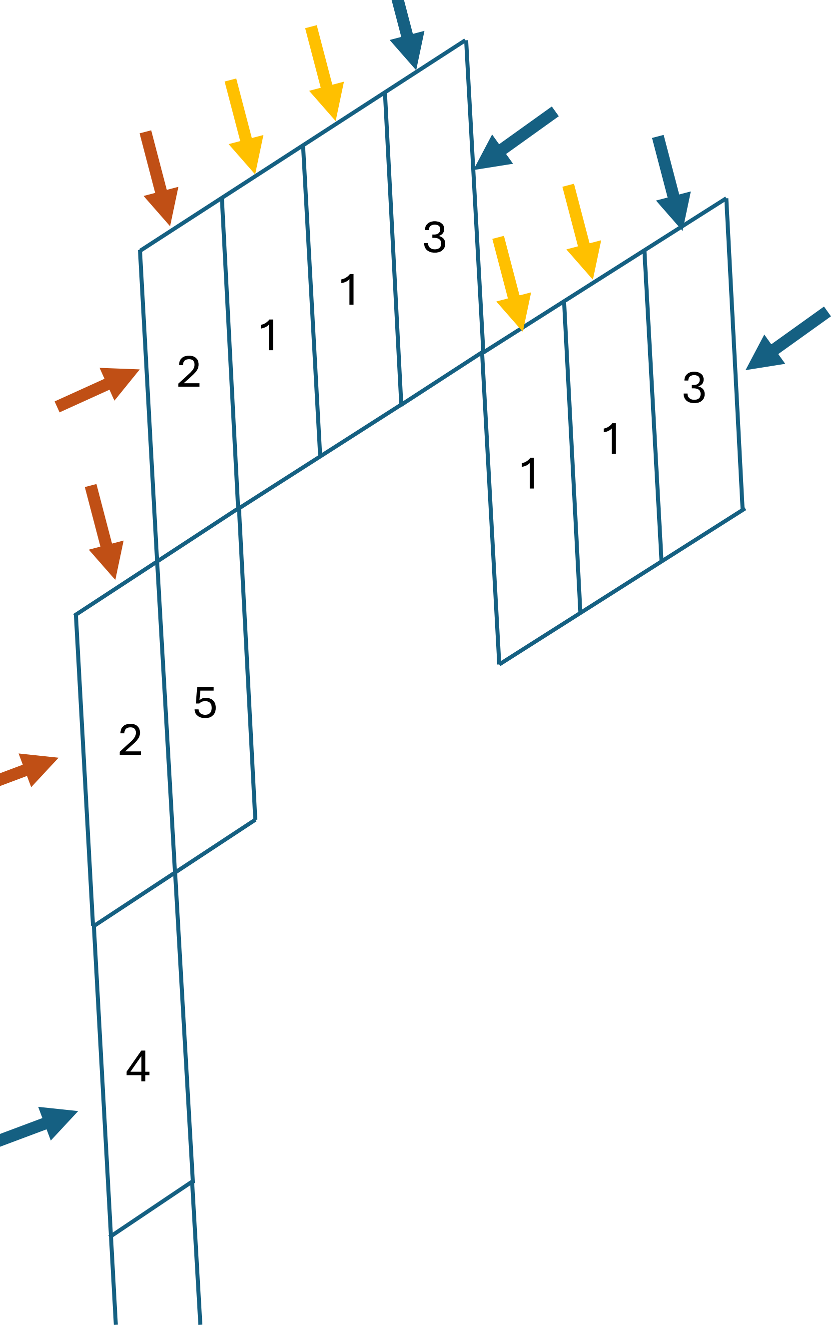



```
for n2=1:length(lon1)
    if n1~=1
        ind1=[ind1,find((lon1==lon2(n2))&(lat1==lat2(n2)))];
        ind2=[ind2,find((lon1==lon0(n2))&(lat1==lat0(n2)))];
    else
        ind1=[ind1,find((lon1==lon2(n2))&(lat1==lat2(n2)))];
    end
end

% atribute 1=north
if (isempty(ind2)& ind1==[4,3])|(ind2==[2,1]&ind1==[4,3])|...
        (ind2==[2,1]&ind1==[3])|(ind2==[2]&ind1==[4,3])
    Atr(n1)=1;
    %          n1
end

 % atribute 5=nothing
 if (ind2==[2,1]&ind1==[2,3])|(ind2==[2,3]&ind1==[4,3])
     Atr(n1)=5;
 end
```

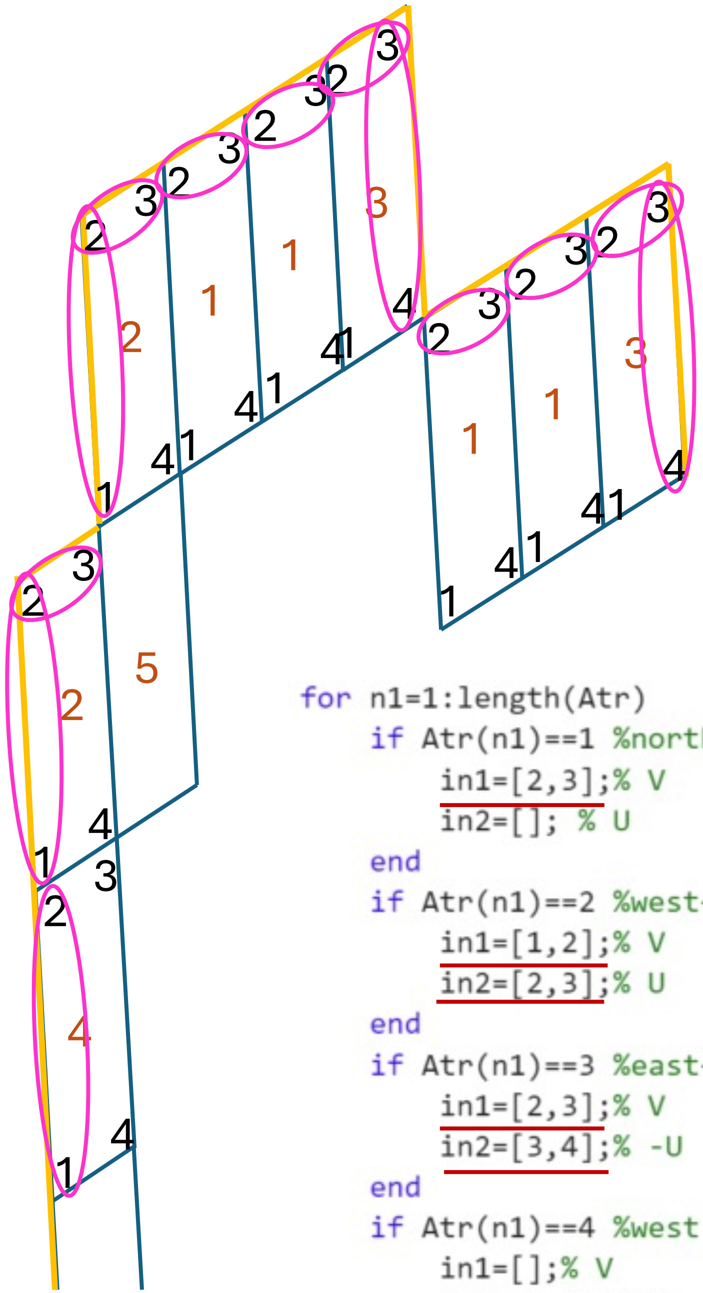
- We know location of the cells
- We want to know open side of the cell




```
% atribute 1=north
if (isempty(ind2)& ind1==[4,3])|(ind2==[2,1]&ind1==[4,3])|...
        (ind2==[2,1]&ind1==[3])|(ind2==[2]&ind1==[4,3])
    Atr(n1)=1;
    %          n1
end
% atribute 2=north+west
if ~isempty(ind2)
    if (ind2==[1]&ind1==[4,3])|(ind2==[1,4]&ind1==[4,3])|...
            (ind2==[1,4]&ind1==[3])|(ind2==[1]&ind1==[3])
        %                n1
        Atr(n1)=2;
    end
    % atribute 3=north+east
    if (ind2==[2,1]&ind1==[1,4])|(ind2==[2,1]&ind1==[4])
        Atr(n1)=3;
    end

    % atribute 4=west
    if (ind2==[1]&ind1==[2,3])|(ind2==[1,4]&ind1==[2,3])
        Atr(n1)=4;
    end


    % atribute 5=nothing
    if (ind2==[2,1]&ind1==[2,3])|(ind2==[2,3]&ind1==[4,3])
        Atr(n1)=5;
    end
end
```




```
for n1=1:length(Atr)
    if Atr(n1)==1 %north [2,3] - north;
        in1=[2,3];% V
        in2=[]; % U
    end
    if Atr(n1)==2 %west+north [2,3] - north; [1,2] - west;
        in1=[1,2];% V
        in2=[2,3];% U
    end
    if Atr(n1)==3 %east+north [2,3] - north; [3,4] - east;
        in1=[2,3];% V
        in2=[3,4];% -U
    end
    if Atr(n1)==4 %west [1,2] - west;
        in1=[];% V
        in2=[1,2];% U
    end
```

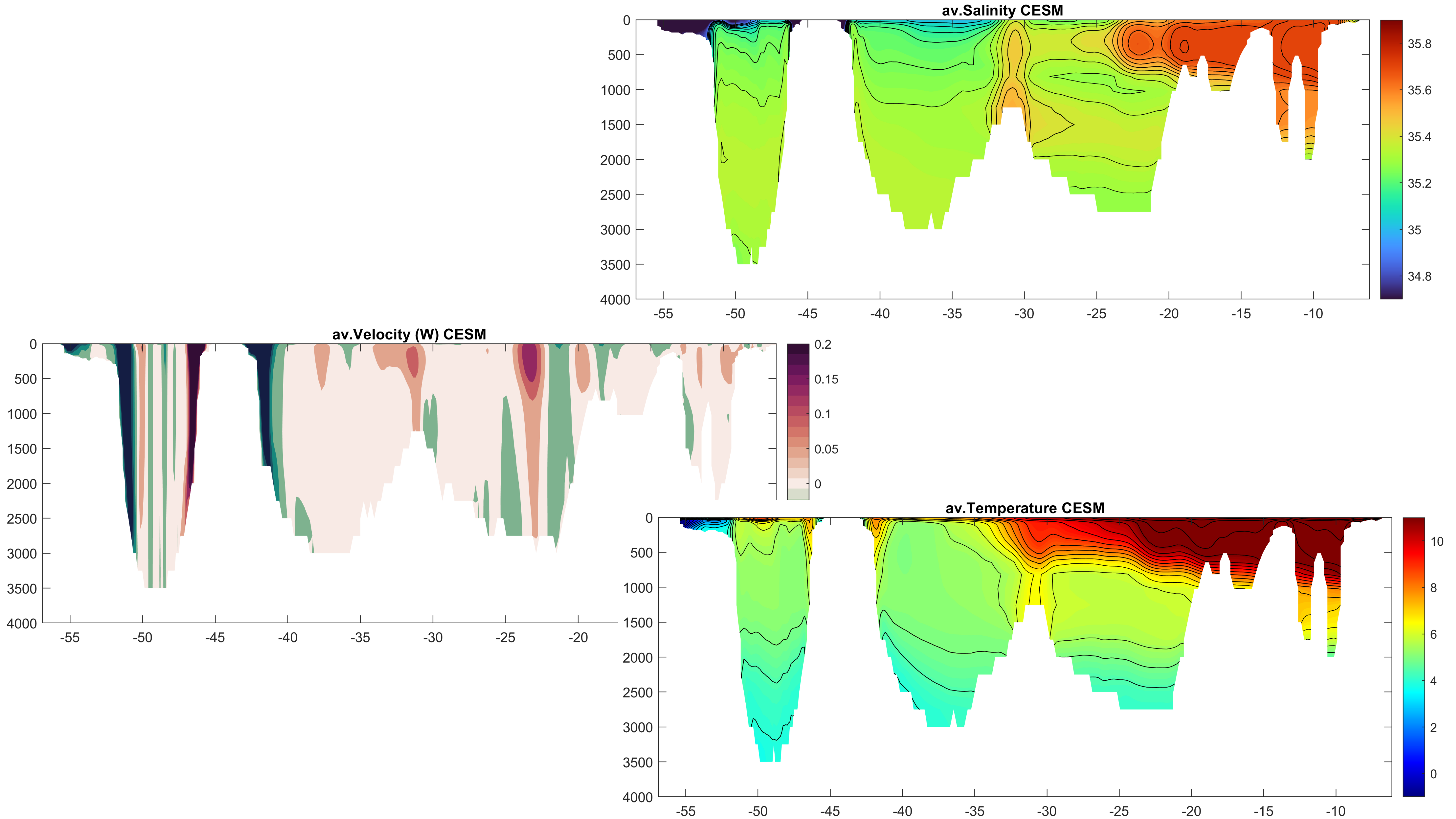
```
if ~isempty(in1)& ~isempty (in2)& in2==[2,3] % west+north
    indc=CellN([in1,in2],:,n1);
    Lat0=[Lat0,LatC(n1,[in1,in2])];
    Lon0=[Lon0,LonC(n1,[in1,in2])];
    Vel=cat(2,Vel,f2(-U,V,indc));%%%%%% minus U here %%%%%
    Tem=cat(2,Tem,f2(temp,temp,indc));
    Sal=cat(2,Sal,f2(salt,salt,indc));

elseif ~isempty(in1)& ~isempty (in2)& in2==[3,4]% east+north
    indc=CellN([in1,in2],:,n1);
    Lat0=[Lat0,LatC(n1,[in1,in2])];
    Lon0=[Lon0,LonC(n1,[in1,in2])];
    Vel=cat(2,Vel,f2(V,U,indc));
    Tem=cat(2,Tem,f2(temp,temp,indc));
    Sal=cat(2,Sal,f2(salt,salt,indc));

elseif ~isempty(in1) & isempty(in2) % north
    indc=CellN([in1,in2],:,n1);
    Lat0=[Lat0,LatC(n1,[in1])];
    Lon0=[Lon0,LonC(n1,[in1])];
    Vel=cat(2,Vel,f1(V,indc));
    Tem=cat(2,Tem,f1(temp,indc));
    Sal=cat(2,Sal,f1(salt,indc));

elseif isempty(in1) & ~isempty(in2)%west
    indc=CellN([in1,in2],:,n1);
    Lat0=[Lat0,LatC(n1,[in2])];
    Lon0=[Lon0,LonC(n1,[in2])];
    Vel=cat(2,Vel,f1(-U,indc));%%%%%% minus U here %%%%%
    Tem=cat(2,Tem,f1(temp,indc));
    Sal=cat(2,Sal,f1(salt,indc));

end
```


av.Salinity CESM
35.8
35.6
35.4
35.2
35
34.8
av.Velocity (W) CESM
0.2
0.15
0.1
0.05
0
av.Temperature CESM
10
8
6
4
2
0

Estimation of the volume transport using OSNAP line in CESM coordinates

- Preparation steps:
  - Open NC file -> cutting off the OSNAP region
  - Linear interpolation of T S from t-grid to u-grid
- Density estimation
- Integration of Heat, Salt and volume
- Stream function – accumulative sum

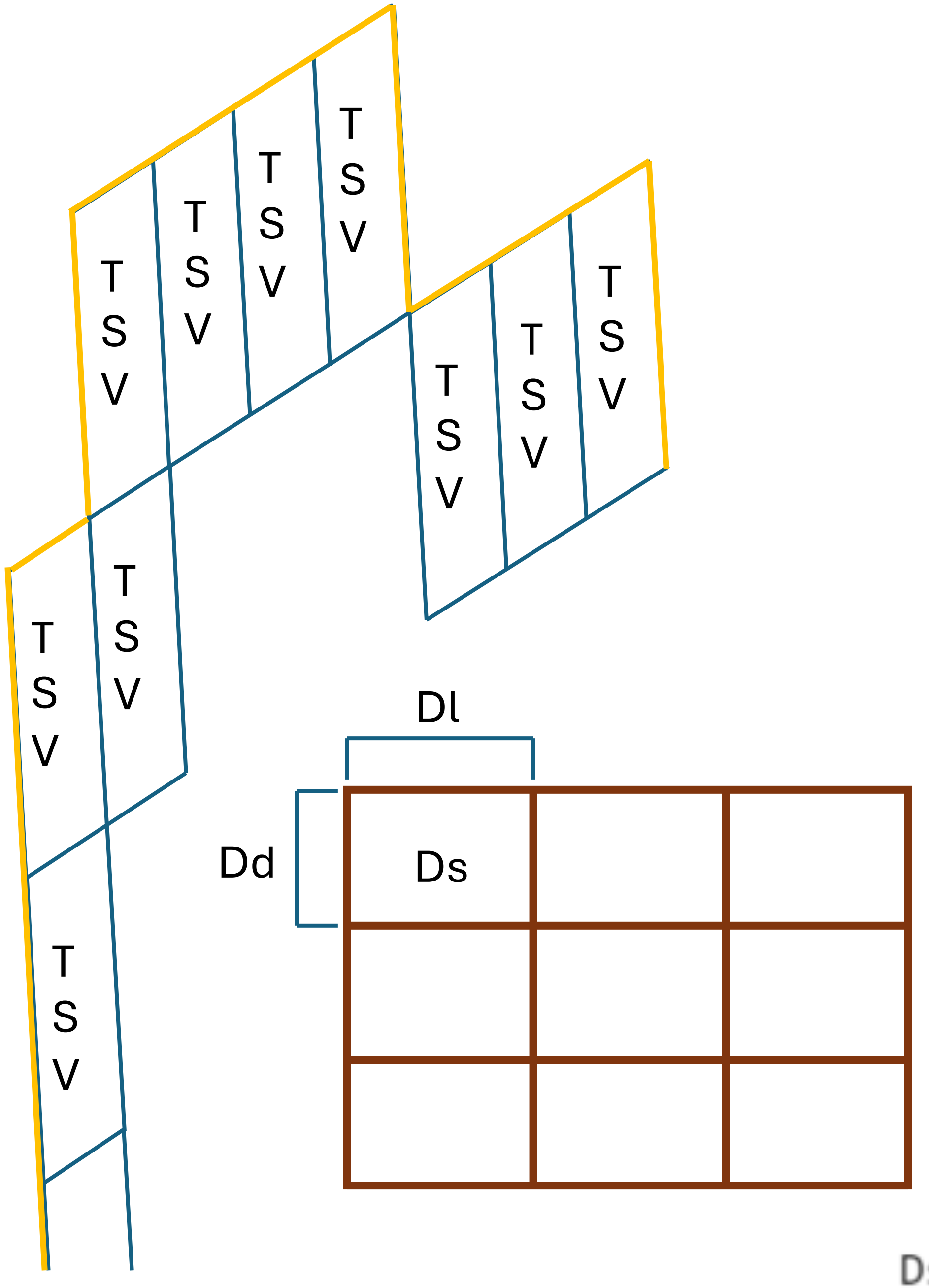
Integral transport through the OSNAP line

$$MOC(t) = max[\Psi(\sigma,t)] = \max\left[\int_{\sigma_{min}}^{\sigma}\int_{x_w}^{x_e} v(x,\sigma,t)\,dx\,d\sigma\right], [\mathrm{Sv}]$$

$$MHT(t) = \rho C_p \int_{z_{min}}^{z_{max}}\int_{x_w}^{x_e} v(\sigma,t)\theta(\sigma,t)\,dx\,dz, [W]$$

$$MFT(t) = -\int_{z_{min}}^{z_{max}}\int_{x_w}^{x_e} v(\sigma,t)\frac{S(\sigma,t)-\bar{S}}{\bar{S}}\,dx\,dz, [Sv]$$

- define the area of every cell on the cross-section:
  - Step by depth
  - Step by distance
- define the density coordinates
  - Find density field, decide on density intervals



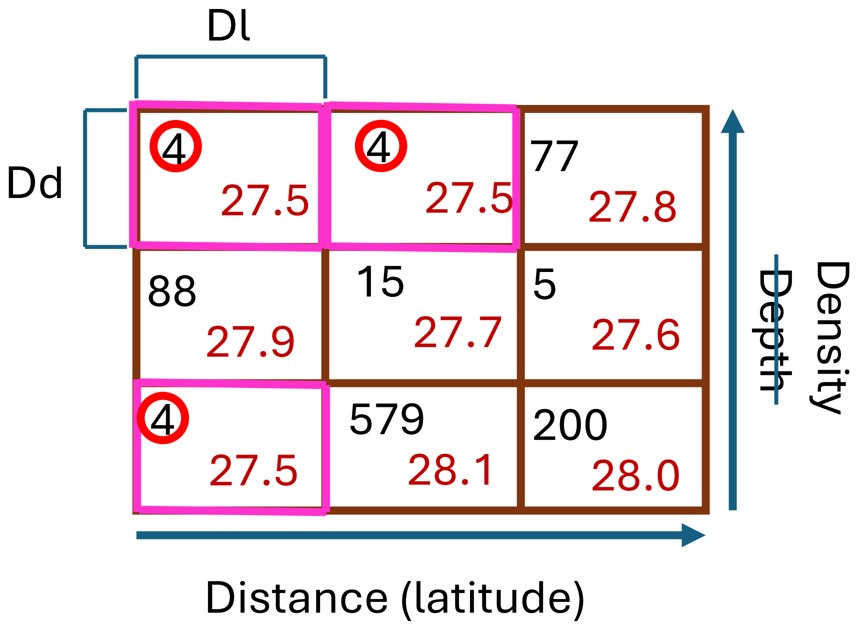

```
Dl=[];
for n1=1:length(lat)-1
    dy=111303;
    dx=dy*cosd(lat(n1));
    dlon=lon(n1)-lon(n1+1); dlat=lat(n1)-lat(n1+1);
    dlon=dlon*dx;
    dlat=dlat*dy;
    Dl(n1)=sqrt(dlon.^2+dlat.^2);
end
```

```
D=ones(size(vel,1),length(lat));
D=D.*depth';
Dd=[];
for n2=1:length(depth)-1
    Dd(n2,:)=D(n2+1,:)-D(n2,:);
end
```

```
Ds=(zeros(size(Dd,1),size(Dl,2))+Dl).*Dd;% cell size in m squared
```

Density coordinates

```
Ro=gsw_sigma0(salt,temp);% get potential dens
Rolay=min(Ro(:)):0.01:max(Ro(:))+0.1; %density intervals
```




```
IND=ones(size(Ro,1)*size(Ro,2),size(Ro,3))*nan;% empty
IND1=IND;% empty
for n2=1:size(Ro,3)
    ind=[];ind11=[];% empty
    for n1=1:length(Rolay)-1

        in=find(Ro(:,:,n2)>=Rolay(n1)&Ro(:,:,n2)<Rolay(n1+1));
        in11=n1*ones(size(in));% index for layer to attribute
        if ~isempty (in)
            ind=cat(1,ind,in);
            ind11=cat(1,ind11,in11);
        else
            ind=cat(1,ind,0);% accum values
            ind11=cat(1,ind11,0);
        end
    end
    IND(1:length(ind),n2)=ind; % matrix with number of cells
    IND1(1:length(ind11),n2)=ind11;% matrix of density layer attributes
end
```

cells that are in particular density level

Density layer attribute (No) for integration

output for the full array

Integration

$$MOC(t) = max[\Psi(\sigma,t)] = \max\left[\int_{\sigma_{min}}^{\sigma}\boxed{\int_{x_w}^{x_e}} v(x,\sigma,t)dxd\sigma\right], [\mathrm{Sv}]$$

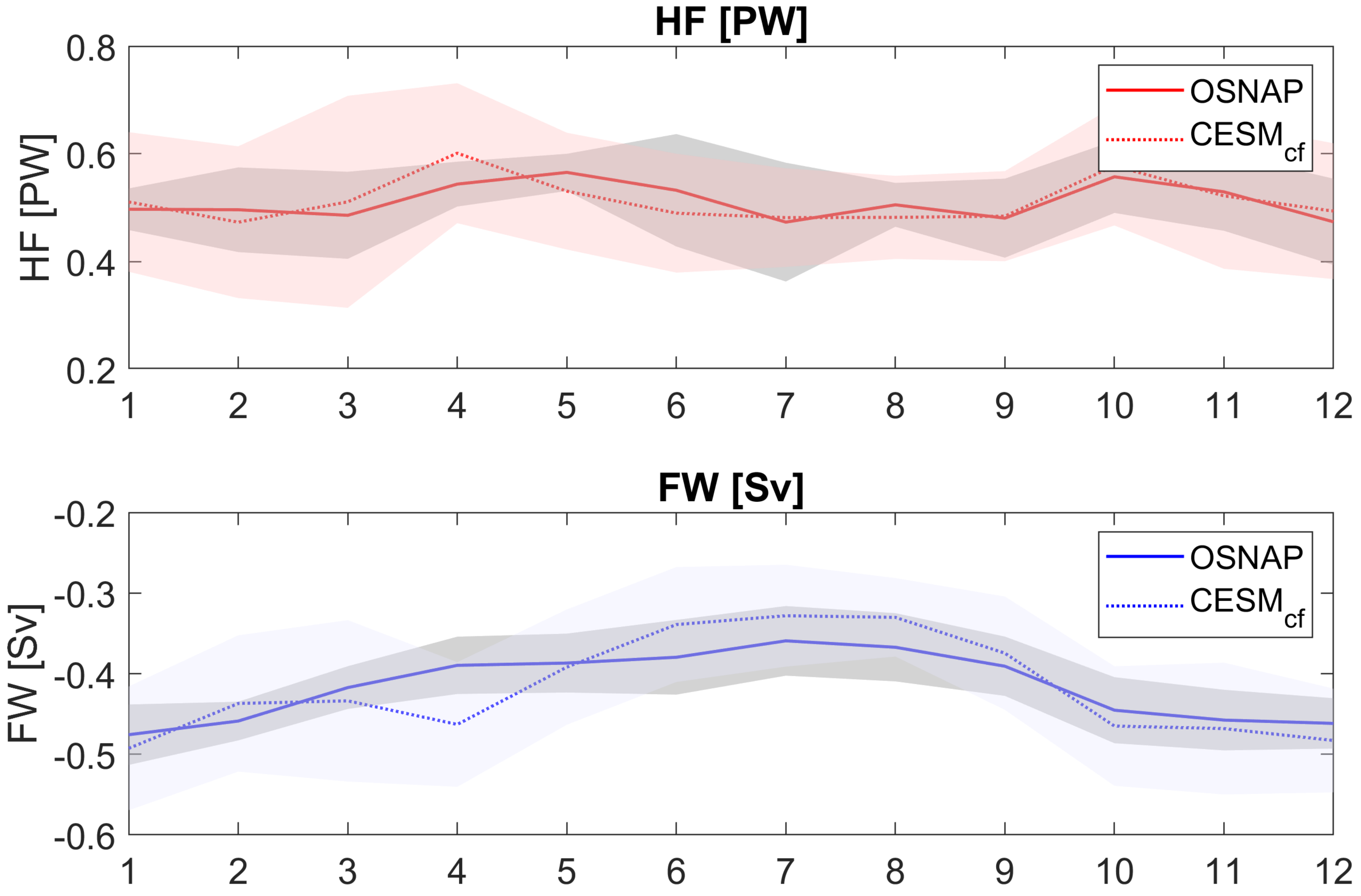
$$MHT(t) = \boxed{\rho C_p}\boxed{\int_{z_{min}}^{z_{max}}}\boxed{\int_{x_w}^{x_e}} v(\sigma,t)\theta(\sigma,t)dxdz, [W]$$

$$MFT(t) = -\boxed{\int_{z_{min}}^{z_{max}}}\boxed{\int_{x_w}^{x_e}} v(\sigma,t)\frac{S(\sigma,t) - \boxed{\bar{S}}}{\bar{S}}dxdz, [Sv]$$

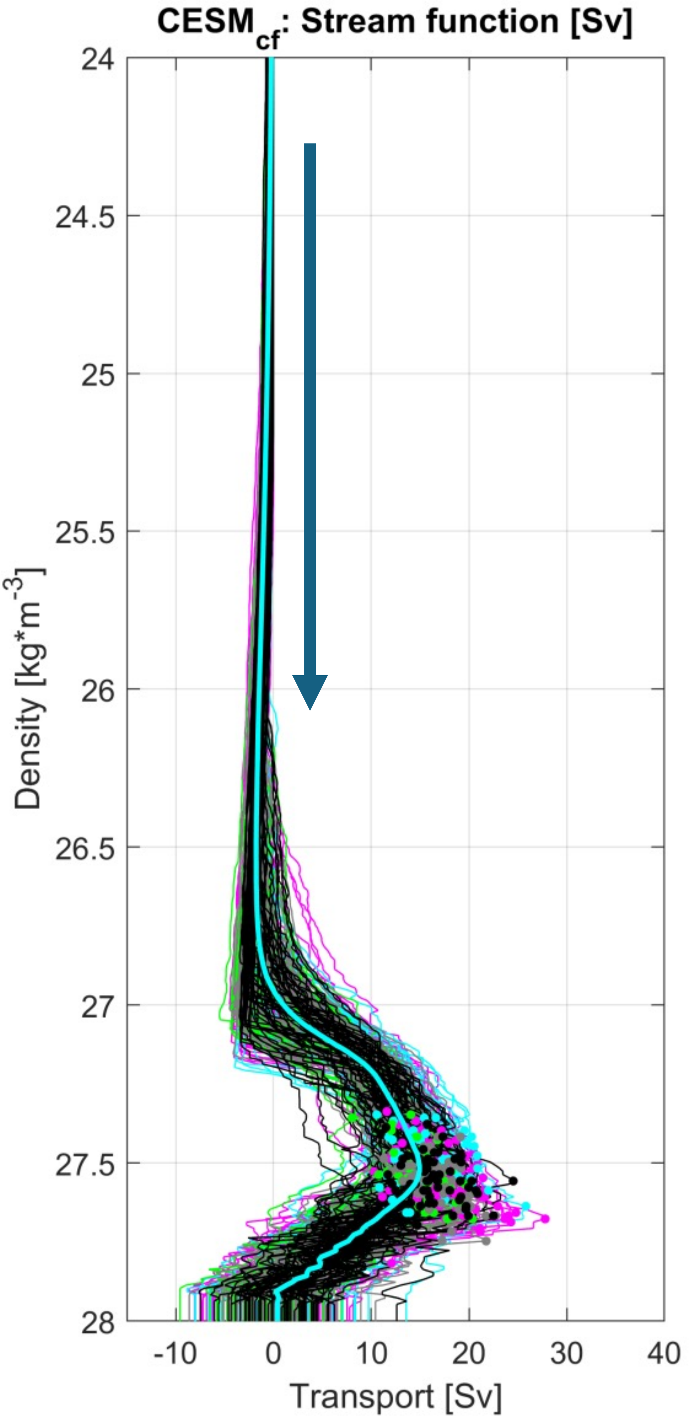
```
Roref=gsw_sigma0(Sref, 0)+1000;
Cp=gsw_cp0;
Sref=34.92
VOL=[];
HF=[];
FW=[];
for n2=1:size(Ro,3)
    vol=[];
    hit=[];
    fwa=[];
    for n1=1:length(Rolay)
        ind=IND(IND1(:,n2)==n1,n2);% define location of cell attributed to
        v1=vel(:,:,n2);
        if ~isempty (ind)
            vol=cat(1,vol,sum(v1(ind).*Dl(ind)));% integration inside dens
            hit=cat(1,hit,sum(v1(ind).*temp(ind).*Dl(ind)));% integration
            fwa=cat(1,fwa,sum(v1(ind).*((salt(ind)-Sref)/Sref).*Dl(ind)));
        else
            vol=cat(1,vol,nan);
            hit=cat(1,hit,nan);
            fwa=cat(1,fwa,nan);
        end

    end
    VOL=cat(2,VOL,vol);% write VOLUME
    HF=cat(2,HF,hit);% write HEAT flux
    FW=cat(2,FW,fwa);% write FRESHWATER flux

end
HF=Cp*Roref*sum(HF,1,'omitnan')/10^15;% heat flux in dens coordinates
FW=-sum(FW,1,'omitnan')/10^6;% freshwater (salt) flux in dens coordinates
```

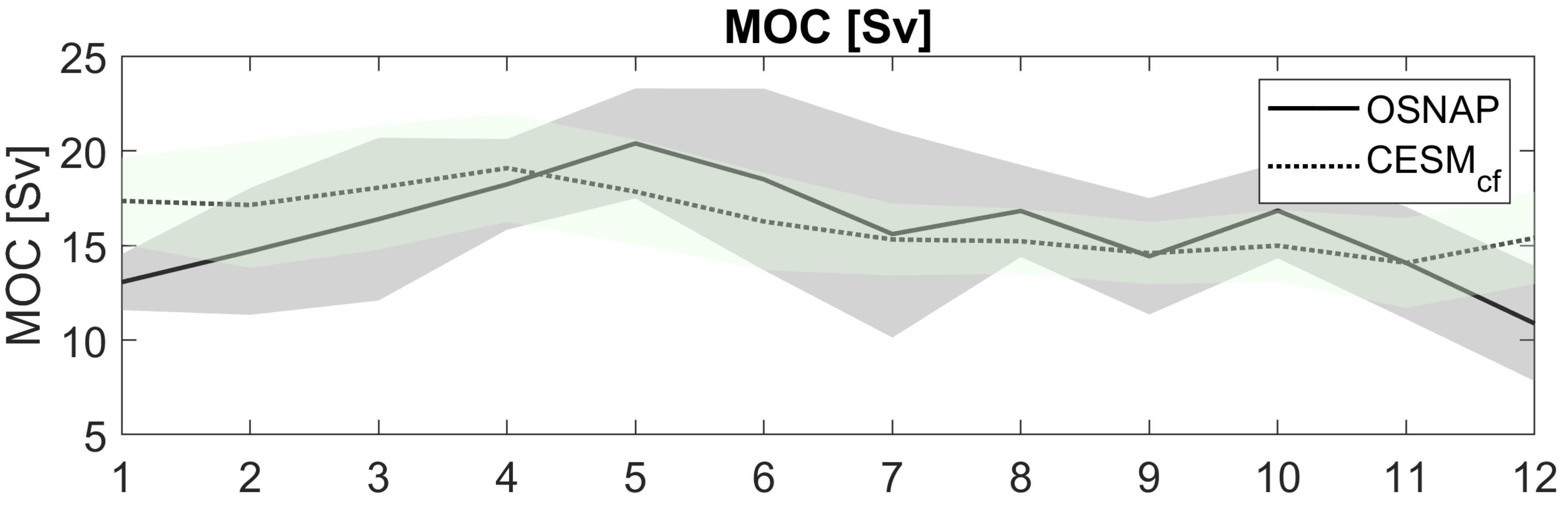


HF [PW]
0.8
0.6
0.4
0.2
1
2
3
4
5
6
7
8
9
10
11
12
OSNAP
CESM$_{cf}$
FW [Sv]
-0.2
-0.3
-0.4
-0.5
-0.6
1
2
3
4
5
6
7
8
9
10
11
12
OSNAP
CESM$_{cf}$



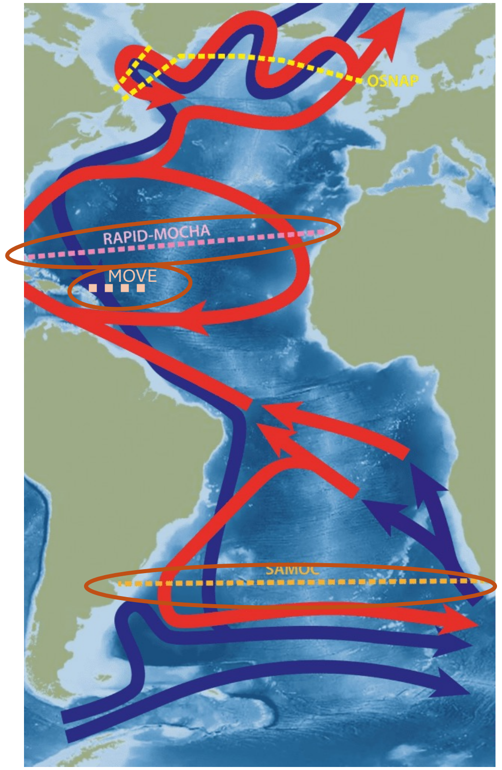
$$MOC(t) = \underline{max[\Psi(\sigma,t)] = } \max\left[\int_{\sigma_{min}}^{\sigma}\int_{x_w}^{x_e} v(x,\sigma,t)dxd\sigma\right], [\mathrm{Sv}]$$

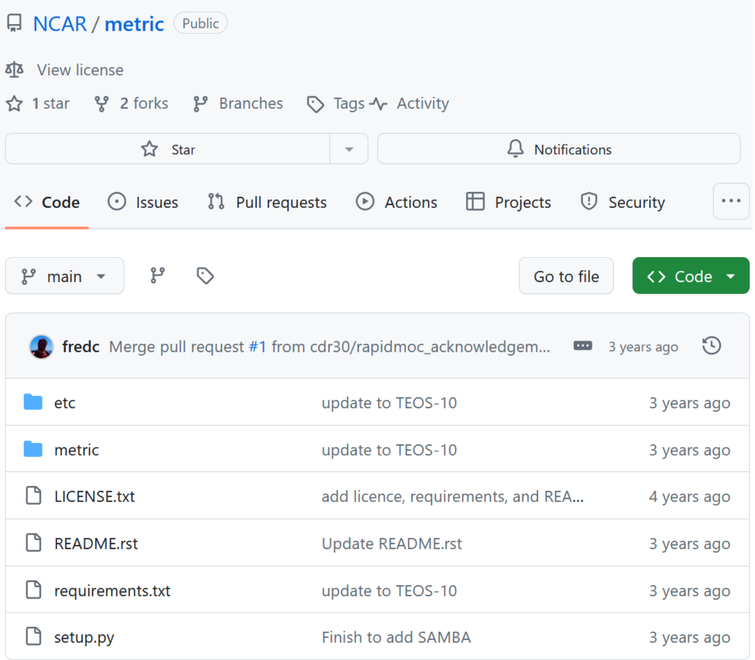
```
Ve=[];
for n2=1:size(VOL,2)
    W=0;
    for n=1:size(VOL,1)
        W=sum(cat(2,W,VOL(n,n2)),'omitnan');% accumulative sum
        Ve(n,n2)=W;
    end
end
Ve=Ve/10^6;
[MOC,np]=max(Ve,[],1);% MOC in dens coordinates
```



AMOC metric

- https://github.com/NCAR/metric





Meridional ovErTurning ciRculation diagnostIC (METRIC)

METRIC is a fork of the RapidMoc package, which extends the calculation of observation-style transports to other observing arrays.

The METRIC python module enables consistent calculations of Atlantic meridional overturning circulation (AMOC) estimates at various observational arrays from ocean general circulation models. To make the most appropriate comparisons, the package evaluates the model meridional overturning circulation using analogous observation-style methods. The current version allows AMOC estimates at the RAPID (26.5N) site, the MOVE (16N) site, and the SAMBA (34.5S) site. METRIC also includes a few additional, alternative approaches to calculate these transports.