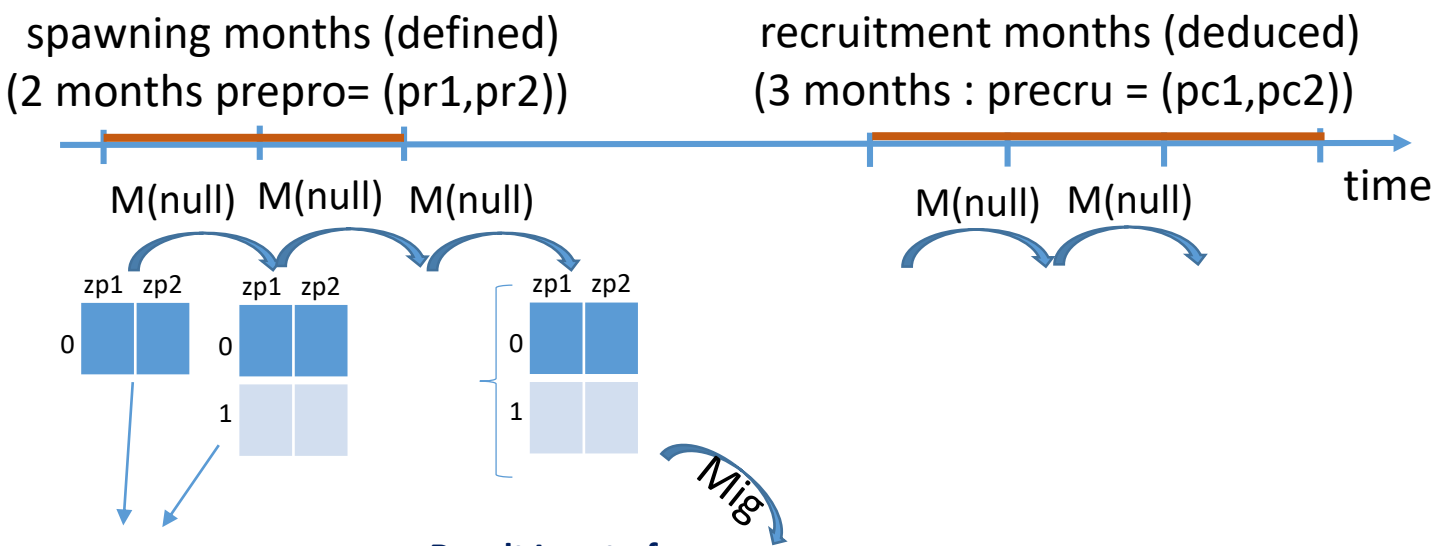
Default coding

Reproduction Equation **Recruitement Equation**



Illustration

2 reproduction zones
2 recruitment zones
Migration Repro-Recru : Mig =

	zc1	zc2
zp1	1	0
zp2	0	1

Reproduction season : 2 months
prepro= (pr1,pr2), pr1+pr2 can be !=0
Recruitment season : 3 months
precru = (pc1,pc2)), pc1+pc2=1

Result of Reproduction

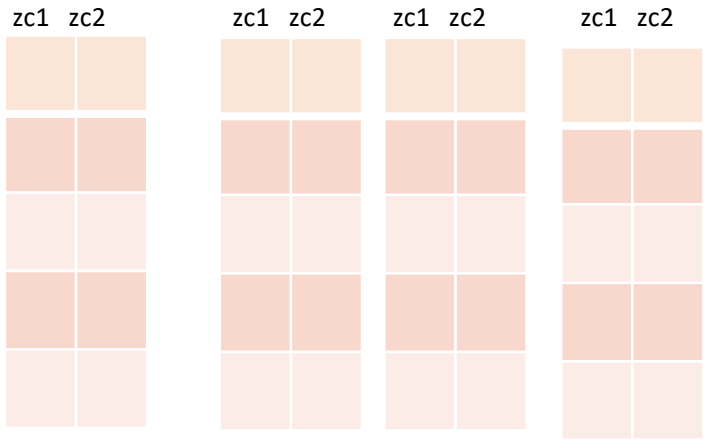
(vecteur de dimension nb zones repro) *at each month of reproduction season, stored in reproductions of PopulationMonitor*

No equation by default

Need to be coded by user

Result Input of Recruitment (matrix of dimension nb groupxnbzonesRecrutement)

at each month of the recruitment season, result stores the nb of recruits computed by ISIS in each groupxzone Recru



Result of Recruitment

(matrix of dimension nb groupxnbzonesRecrutement) *at each month of the recruitment season, stored in N(group,zc) the group is computed by isis according to the nb of month between repro and recruitment and teh number of months of repro and recuit. Seasons*

Default coding

can be modified by user in Recruitment equation

Code de Reproduction : https://gitlab.nuiton.org/ifremer/isis-fish/blob/master/src/main/java/fr/ifremer/isisfish/entities/PopulationSeasonInfoImpl.java#L797

Code de la recrutement : https://gitlab.nuiton.org/ifremer/isis-fish/blob/master/src/main/java/fr/ifremer/isisfish/simulator/PopulationMonitor.java#L252

Reproduction

Paramètres de Compute reproduction equation:
@param context simulation context
@param N effectif courant de la population
@param pop population pour lequel on souhaite la matrice de reproduction
@param month le mois pour lequel on souhaite la matrice de reproduction
@param prepro le coefficient de reproduction de la population pour ce mois
@param zoneRepro la liste des zones de reproduction (dimension 0 de result)
@param groups la liste des groups de la population (dimension 0 de N)
@param zones la liste des zones de la population (dimension 1 de N)
@param result la matrice resultat que l'equation doit remplir, il s'agit d'un vecteur avec comme semantique la liste des zones de reproduction
@return la valeur retournée n'est pas utilisée, par exemple 'return 0;' convient.

Exemples de codage

Si on veut une reproduction, il faut la coder
En sortie : result est de taille le nb de zones de repro

ex1

```
Return 0;
```

Result[zp1]=0 et Result[zp2]=0 à chaque pas de temps de la repro

ex2

```
double Neggs=0;
int step = context.getSimulationControl().getStep().getStep();
for (Zone zone:zoneRepro){
   Neggs=0;
   for (PopulationGroup cr : groups)
                Neggs += N.getValue(cr,zone)*cr.getMaturityOgive()* cr.getReproductionRate()*prepro;
  System.out.println("zone repro"+ zone.getName() +"date "+ step+":"+ Neggs);
  result.setValue(zone,Neggs);
}
return 0;
```

result[zp1] contient Neggs = Na[zp1] * Pm *tx Repro* Prepro au pas de temps t de la repro
result[zp2] contient Neggs = Na[zp1] * Pm *tx Repro* Prepro au pas de temps t de la repro

Recruitment

Paramètres de Compute recruitment equation:
@param context simulation context
@param step current time step
@param pop population pour lequel on souhaite la matrice de recrutement
@param recruitmentInputs les abundances, biomasses et reproduction correspondant au recrutement courant
@param result la matrice resultat que l'equation doit remplir, il s'agit d'une matrice groupes/zones Recrutement
@return la valeur retournée n'est pas utilisée, par exemple 'return 0;' convient.

Result en entrée de recrutement contient la valeur par défaut : pour chaque zone de recrutement et group (calculé par isis en utilisant M(null) et Mig(repro,recru))

au 1er pas de recrutement :
Result[group,zc1]= recruitmentInputs.get(0).getReproduction(zc1)*precru(0)
au 2eme pas de recrutement :
Result[group,zc1]= recruitmentInputs.get(0).getReproduction (zc1)*precru(1)
+ recruitmentInputs.get(1).getReproduction (zc1)* precru(0)
au 3eme pas de recrutement :
Result[group,zc1]= recruitmentInputs.get(1).getReproduction (zc1)*precru(1)

Exemples de codage

Ex1

Relation de repro/recru par défaut dans ISIS

```
Return 0;
```

Renvoie la valeur par default de Result

Ex2

Relation stock/recrutement classique

```
double k = 10000;
double biom = recruitmentInputs.get(0).getBiomass().sumAll(); // zone x group
Double p = recruitmentInputs.get(0).getRecruitementContribution();
double res=0;
PopulationGroup g0  = pop.getPopulationGroup().get(0);
if(p != null) res = k*biom*p; // test necessaire car p est null si le mois de repro demandé ne contribue pas au mois de recru en cours
for(Zone z : pop.getRecruitmentZone() ){     result.setValue(g0,z,res / pop.getRecruitmentZone().size()); }
return 0;
```

Ne se soucie pas de ce qu'à produit la repro et Renvoie eq stock recrutement simple R = k * biomass, la biomasse est celle présente au pas de temps de la repro qui a contribue à ce pas de temps du recrutement

Recruitment

Exemples de codage

Ex2bis

Relation stock/recrutement classique

```
double k = 10000;  //recutement estimé par l'eval (1 seul mois de repro, 1 seul mois de recrut et 1 seule zone)
PopulationGroup g0  = pop.getPopulationGroup().get(0);
if(p != null) res = k; // test necessaire car p est null si le mois de repro demandé ne contribue pas au mois de recru en cours
for(Zone z : pop.getRecruitmentZone() ){     result.setValue(g0,z,res ); }
return 0;
```

Ne se soucie pas de ce qu'à produit la repro et Renvoie eq stock recrutement simple R = recutement estimé par l'eval (1 seul mois de repro, 1 seul mois de recrut et 1 seule zone)

Ex3

Si on connait le recrutement, on peut programmer Reproduction en mettant M(null)=0 (au lieu de programmer Recruitment)