SEAPODYM Source Code Documentation

4.01

Generated by Doxygen 1.8.17

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 2

Class Index

2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 3

Class Documentation

3.1 CBord Class Reference

Class handling the type of the borders of a grid cell.

`#include <Map.h>`

Public Member Functions

- int **cotex** ()
- int **cotey** ()

Public Attributes

- union {
  unsigned short int **b**
  struct {
  char **x**
  char **y**
  } **cote**
  };

3.1.1 Detailed Description

Class handling the type of the borders of a grid cell.

For a given pair of indices (i,j) structure cote stores the type of cell's borders, two in x and two in y direction - left-closed (G_FERME), right-closed (D_FERME) or open (SANS) for ocean cells, and land (TERRE) if the land is next to the land cell.

The documentation for this class was generated from the following file:

- src/Map.h

3.2 CCalpop Class Reference

This is main computational class: all functions and variables to solve ADR equations are here.

`#include <calpop.h>`

Public Member Functions

- void **InitCalPop** (CParam &param, const PMap &map)
- void **precaldia** (const CParam &param, const PMap &map, CMatrices &mat)
- void **precaldia_comp** (const PMap &map, CParam &param, CMatrices &mat, const dmatrix &habitat, const dmatrix &total_pop, double MSS, double MSS_size_slope, double sigma_species, double c_diff_fish, const int sp, const int age, const int jday)
- void Precaldia_Caldia (const PMap &map, VarParamCoupled &param, VarMatrices &mat, dvar_matrix &habitat, dvar_matrix &total_pop, const int sp, const int age, const int t_count, const int jday)
- void **caldia** (const PMap &map, const CParam &param, const DMATRIX &diffusion_x, const DMATRIX &advection_x, const DMATRIX &diffusion_y, const DMATRIX &advection_y)
- void **caldia_GO** (const PMap &map, const CParam &param, const DMATRIX &diffusion_x, const DMATRIX &advection_x, const DMATRIX &diffusion_y, const DMATRIX &advection_y)
- void **starvation_penalty** (const PMap &map, VarParamCoupled &param, VarMatrices &mat, dvar_matrix &mortality, dvar_matrix &total_pop, dvar3_array &nF_ratio, dvar_matrix &uu, const int sp, const int age)
- void **precalrec** (PMap &map, const dmatrix &mortality)
- void Precalrec_juv (const PMap &map, CMatrices &mat, dvar_matrix &mortality, const int t_count)
- void precalrec_juv_comp (const PMap &map, dmatrix &bm, const dmatrix &mortality)
- void Precalrec_total_mortality_comp (const PMap &map, VarParamCoupled &param, VarMatrices &mat, CReadWrite &rw, dvar_matrix &mortality, const int age, const int sp, const int t_count, const int year, const int month, const int step_count)
- void **Recomp_total_mortality_comp** (const PMap &map, CParam &param, CMatrices &mat, CReadWrite &rw, dmatrix &mortality, const int age, const int sp, const int year, const int month, const int step_count)
- void precalrec_total_mortality_comp (const imatrix carte, const dmatrix effort, dvar_matrix &mortality, const double sq, const dvector lat_correction)
- void Precalrec_Calrec_adult (const PMap &map, VarMatrices &mat, VarParamCoupled &param, CReadWrite &rw, dvar_matrix &uu, dvar_matrix &mortality, const int t_count, const bool fishing, const int age, const int sp, const int year, const int month, const int jday, const int step_count, const int no_mortality)
- void **calrec** (const PMap &map, dmatrix &uu, const dmatrix &mortality)
- void **calrec1** (const PMap &map, dvar_matrix &uu, const dmatrix &mortality)
- void **calrec_with_catch** (const PMap &map, CParam &param, dvar_matrix &uu, const dmatrix &C_obs, dvar_matrix &C_est)
- void **calrec_GO** (const PMap &map, dvar_matrix &uu)
- void **calrec_GO_with_catch** (const PMap &map, CParam &param, dvar_matrix &uu, const dmatrix &C_obs, dvar_matrix &C_est)
- void Calrec_juv (const PMap &map, CMatrices &mat, dvar_matrix &uu, dvar_matrix &mortality, const int t_count)
- void **Calrec_adult** (const PMap &map, dvar_matrix &uu, dvar_matrix &mortality)
- void **Recomp_abc_coef** (const PMap &map, CMatrices &mat, const int t_count, const dmatrix &mortality, dmatrix &aa, dmatrix &bbm, dmatrix &cc)
- void **Recomp_DEF_coef** (const PMap &map, CParam &param, CMatrices &mat, const int t_count, const int jday, const dmatrix &habitat, dmatrix &dd, dmatrix &ee, dmatrix &ff, dmatrix &advection_x, dmatrix &advection_y, const int sp, const int age, const double MSS, const double c_diff_fish, const double sigma_species)
- void **Recomp_DEF_UV_coef** (const PMap &map, CParam &param, CMatrices &mat, dmatrix &u, dmatrix &v, const dmatrix &habitat, dmatrix &dd, dmatrix &ee, dmatrix &ff, dmatrix &advection_x, dmatrix &advection_y, const int sp, const int age, const double MSS, const double c_diff_fish, const double sigma_species, const int jday)

- void **RecompDiagCoef_juv** (const PMap &map, CMatrices &mat, const int t_count, const dmatrix mortality, dmatrix &a, dmatrix &bm, dmatrix &c, dmatrix &d, dmatrix &e, dmatrix &f)
- void **RecompDiagCoef_adult** (const PMap &map, CParam &param, CMatrices &mat, const int t_count, const int jday, const dmatrix &mortality, const dmatrix &habitat, dmatrix &aa, dmatrix &bbm, dmatrix &cc, dmatrix &dd, dmatrix &ee, dmatrix &ff, const int sp, const int age, const double MSS, const double c_diff_fish, const double sigma_species)
- void **RecompDiagCoef_UV_adult** (const PMap &map, CParam &param, CMatrices &mat, const int t_count, const int jday, const dmatrix &mortality, const dmatrix &habitat, dmatrix &aa, dmatrix &bbm, dmatrix &cc, dmatrix &dd, dmatrix &ee, dmatrix &ff, const int sp, const int age, const double MSS, const double c_diff_fish, const double sigma_species)
- void **RecompM_sp** (const PMap &map, const CParam &param, dmatrix &M, const dmatrix &H, const double age, const int sp)
- void Predicted_Catch_Fishery (const PMap &map, VarParamCoupled &param, VarMatrices &mat, CReadWrite &rw, const int sp, const int f, const int k, const int year, const int month, const int t_count, const int step_count)
- void **predicted_catch_fishery_comp** (const PMap &map, CParam &param, VarMatrices &mat, const int f, const int k, const int sp, const int age, const dmatrix &uu, const int step_count)
- void Total_obs_catch_age_comp (const PMap &map, VarParamCoupled &param, VarMatrices &mat, CReadWrite &rw, const int age, const int sp, const int year, const int month, const int t_count)
- void Ctot_proportion_fishery_comp (const PMap &map, CParam &param, CMatrices &mat, CReadWrite &rw, const int year, const int month, const int sp)
- void **Recomp_C_fishery_proportion_in_Ctot** (const PMap &map, CParam &param, CReadWrite &rw, dmatrix &Ctot_proportion_fishery, const int year, const int month, const int sp, const int k)
- void Total_exploited_biomass_comp (const PMap &map, VarParamCoupled &param, VarMatrices &mat, const int sp, const int t_count)
- void **Selectivity_comp** (CParam &param, const int nb_fishery, const int a0, const int nb_ages, const int sp)
- void Predicted_Catch_Fishery_no_effort (const PMap &map, VarParamCoupled &param, VarMatrices &mat, CReadWrite &rw, const int sp, const int year, const int month)
- void **predicted_catch_fishery_no_effort_comp** (const PMap &map, CParam &param, VarMatrices &mat, const int f, const int k, const int sp, const int age)
- void total_exploited_biomass_comp (const imatrix carte, const dmatrix &uu, const dmatrix &Cobs, const int f, const int fne, const int age, const int sp)
- void **Recomp_total_exploited_biomass** (const PMap &map, CParam &param, CMatrices &mat, dmatrix &EB, const dmatrix &Cobs, const dvector &selectivity, const int f, const int sp, const int t_count)
- void total_obs_catch_age_comp (const PMap &map, const CParam &param, CMatrices &mat, const dmatrix &uu, const dmatrix &Cobs, dvar_matrix &Ctot_age_obs, const int f, const int fne, const int k, const int age, const int sp, const double C2Dunits)
- void **Recomp_total_obs_catch_age** (const PMap &map, CParam &param, CMatrices &mat, CReadWrite &rw, dmatrix &Ctot_age_obs, const int age, const int sp, const int year, const int month, const int t_count)
- int **get_iterationN** ()
- int **get_maxn** ()
- int **get_Vinf** ()
- void Xbet_comp1 (const PMap &map, int dt)
- void xbet_comp (const PMap &map, dmatrix &xbet, dmatrix &a, dmatrix &bm, dmatrix &c, int dt)
- void **ybet_comp** (const PMap &map, dmatrix &ybet, dmatrix &d, dmatrix &e, dmatrix &f, int dt)
- void **time_reading_init** ()

Public Attributes

- dvar_matrix **dvarsA**
- dvar_matrix **dvarsB**
- dvar_matrix **dvarsBM**
- dvar_matrix **dvarsC**
- dvar_matrix **dvarsD**
- dvar_matrix **dvarsE**

- dvar_matrix **dvarsF**
- dvar_matrix **Xbet**
- dvar_matrix **Ybet**
- dvar3_array **dvarsSNsum**
- d3_array **Selectivity**
- DMATRIX **uuint**
- double **elapsed_time_reading**

3.2.1 Detailed Description

This is main computational class: all functions and variables to solve ADR equations are here.

3.2.2 Member Function Documentation

3.2.2.1 Calrec_juv()

```
void CCalpop::Calrec_juv (
            const PMap & map,
            CMatrices & mat,
            dvar_matrix & uu,
            dvar_matrix & mortality,
            const int t_count )
```

Forward main function called in simulation mode only for: calrec for larval and juvenile life stages, i.e. with passive drift only. See calrec_adre.cpp

3.2.2.2 Ctot_proportion_fishery_comp()

```
void CCalpop::Ctot_proportion_fishery_comp (
            const PMap & map,
            CParam & param,
            CMatrices & mat,
            CReadWrite & rw,
            const int year,
            const int month,
            const int sp )
```

Forward functions for: predicting catch by fishery without using the effort data. They compute local proportions of catch by fishery in the total catch over 'no effort' fisheries. Note, the catches being used need to have the same units.

3.2.2.3 Precaldia_Caldia()

```
void CCalpop::Precaldia_Caldia (
            const PMap & map,
            VarParamCoupled & param,
            VarMatrices & mat,
            dvar_matrix & habitat,
            dvar_matrix & total_pop,
            const int sp,
            const int age,
            const int t_count,
            const int jday )
```

Forward main function called in simulation mode only for: precaldia and caldia functions. See caldia.cpp

3.2.2.4 Precalrec_Calrec_adult()

```
void CCalpop::Precalrec_Calrec_adult (
            const PMap & map,
            VarMatrices & mat,
            VarParamCoupled & param,
            CReadWrite & rw,
            dvar_matrix & uu,
            dvar_matrix & mortality,
            const int t_count,
            const bool fishing,
            const int age,
            const int sp,
            const int year,
            const int month,
            const int jday,
            const int step_count,
            const int no_mortality )
```

Forward main function called in simulation mode only for: precalrec and calrec for adults functions. See calrec_↩ precalrec.cpp

3.2.2.5 Precalrec_juv()

```
void CCalpop::Precalrec_juv (
            const PMap & map,
            CMatrices & mat,
            dvar_matrix & mortality,
            const int t_count )
```

Forward main function called in simulation mode only for: precalrec for larval and juvenile life stages. See precalrec_juv.cpp

3.2.2.6 precalrec_juv_comp()

```
void CCalpop::precalrec_juv_comp (
            const PMap & map,
            dmatrix & bm,
            const dmatrix & mortality )
```

Forward function for: precalrec for larval and juvenile life stages. This routine precomputes diagonal coefficient for calrec_adre

3.2.2.7 precalrec_total_mortality_comp()

```
void CCalpop::precalrec_total_mortality_comp (
            const imatrix carte,
            const dmatrix effort,
            dvar_matrix & mortality,
            const double sq,
            const dvector lat_correction )
```

Forward functions for: computing the sum of natural and fishing mortalities

3.2.2.8 Precalrec_total_mortality_comp()

```
void CCalpop::Precalrec_total_mortality_comp (
            const PMap & map,
            VarParamCoupled & param,
            VarMatrices & mat,
            CReadWrite & rw,
            dvar_matrix & mortality,
            const int age,
            const int sp,
            const int t_count,
            const int year,
            const int month,
            const int step_count )
```

Forward main function called in simulation mode only for: computing the sum of natural and fishing mortalities See total_mortality_comp.cpp

3.2.2.9 Predicted_Catch_Fishery()

```
void CCalpop::Predicted_Catch_Fishery (
            const PMap & map,
            VarParamCoupled & param,
            VarMatrices & mat,
            CReadWrite & rw,
            const int sp,
            const int f,
            const int k,
            const int year,
            const int month,
            const int t_count,
            const int step_count )
```

Forward main function called in simulation mode only for: predicting catch by fishery based on fishing effort. See predicted_catch.cpp

3.2.2.10 Predicted_Catch_Fishery_no_effort()

```
void CCalpop::Predicted_Catch_Fishery_no_effort (
            const PMap & map,
            VarParamCoupled & param,
            VarMatrices & mat,
            CReadWrite & rw,
            const int sp,
            const int year,
            const int month )
```

Forward main function called in simulation mode only for: predicting catch by fishery without using the effort data. See predicted_catch_without_effort.cpp

3.2.2.11 total_exploited_biomass_comp()

```
void CCalpop::total_exploited_biomass_comp (
            const imatrix carte,
            const dmatrix & uu,
            const dmatrix & Cobs,
            const int f,
            const int fne,
            const int age,
            const int sp )
```

Forward functions for: computing total exploited biomass at age, which is used to split the observed catch of fisheries without effort data among age classes, and then used in computation of predicted catch without effort

3.2.2.12 Total_exploited_biomass_comp()

```
void CCalpop::Total_exploited_biomass_comp (
            const PMap & map,
            VarParamCoupled & param,
            VarMatrices & mat,
            const int sp,
            const int t_count )
```

Forward main function called in simulation mode only for: computing total exploited biomass at age, which is used to split the observed catch of fisheries without effort data among age classes, and then used in computation of predicted catch without effort See total_exploited_biomass.cpp

3.2.2.13 total_obs_catch_age_comp()

```
void CCalpop::total_obs_catch_age_comp (
            const PMap & map,
            const CParam & param,
            CMatrices & mat,
            const dmatrix & uu,
            const dmatrix & Cobs,
            dvar_matrix & Ctot_age_obs,
            const int f,
            const int fne,
            const int k,
            const int age,
            const int sp,
            const double C2Dunits )
```

Forward functions for: computing the total (sum over fisheries without effort) observed catch at age.

3.2.2.14 Total_obs_catch_age_comp()

```
void CCalpop::Total_obs_catch_age_comp (
            const PMap & map,
            VarParamCoupled & param,
            VarMatrices & mat,
            CReadWrite & rw,
            const int age,
            const int sp,
            const int year,
            const int month,
            const int t_count )
```

Forward main function called in simulation mode only for: computing the total (sum over fisheries without effort) observed catch at age. See total_obs_catch_age.cpp

3.2.2.15 xbet_comp()

```
void CCalpop::xbet_comp (
            const PMap & map,
            dmatrix & xbet,
            dmatrix & a,
            dmatrix & bm,
            dmatrix & c,
            int dt )
```

Forward functions for: tridag_bet function. This routine precomputes an operator in the Gaussian solver of the tridiagonal linear system, which does not change during iterations.

3.2.2.16 Xbet_comp1()

```
void CCalpop::Xbet_comp1 (
            const PMap & map,
            int dt )
```

Forward main function called in simulation mode only for: tridag_bet function. See tridag_bet.cpp

The documentation for this class was generated from the following files:

- src/calpop.h
- src/caldia.cpp
- src/Calpop_caldia.cpp
- src/Calpop_calrec.cpp
- src/Calpop_InitCalPop.cpp
- src/Calpop_precaldia.cpp
- src/Calpop_precalrec.cpp
- src/Calpop_recompute_coefs.cpp
- src/Calpop_tridag.cpp
- src/calrec_adre.cpp
- src/calrec_precalrec.cpp
- src/dv_caldia.cpp
- src/dv_calrec_adre.cpp

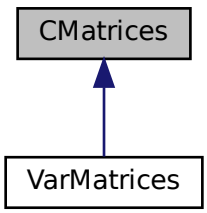
- src/dv_calrec_precalrec.cpp
- src/dv_precalrec_juv.cpp
- src/dv_predicted_catch.cpp
- src/dv_predicted_catch_without_effort.cpp
- src/dv_total_exploited_biomass.cpp
- src/dv_total_mortality_comp.cpp
- src/dv_total_obs_catch_age.cpp
- src/dv_tridag_bet.cpp
- src/fd_caldia.cpp
- src/fd_calrec_adre.cpp
- src/fd_calrec_precalrec.cpp
- src/fd_precalrec_juv.cpp
- src/fd_predicted_catch.cpp
- src/fd_predicted_catch_without_effort.cpp
- src/fd_total_exploited_biomass.cpp
- src/fd_total_mortality_comp.cpp
- src/fd_total_obs_catch_age.cpp
- src/fd_tridag_bet.cpp
- src/precalrec_juv.cpp
- src/predicted_catch.cpp
- src/predicted_catch_without_effort.cpp
- src/total_exploited_biomass.cpp
- src/total_mortality_comp.cpp
- src/total_obs_catch_age.cpp
- src/tridag_bet.cpp
- src/VarCalpop_caldia.cpp
- src/VarCalpop_calrec.cpp

3.3 CMatrices Class Reference

Seapodym matrices class.

`#include <Matrices.h>`

Inheritance diagram for CMatrices:



Public Member Functions

- void **createMatHeader** (const CParam &param)
- void **createMatOcean** (const PMap &map, int t0, int nbt, int nbi, int nbj, int nb_layer, int dt)
- void **createMatTransport** (const PMap &map)
- void **createMatFluxes** (const int nb_region, const int nb_cohort)
- void **createMatSource** (int nforage, int ntr, int nbi, int nbj)
- void **createMatNoBorder** (int nbi, int nbj)
- void **createMatForage** (const PMap &map, int nforage, int t0, int nbt, int nbi, int nbj)
- void **createMatHabitat** (const PMap &map, const int nb_forage, const int nb_species, int t0, int nbt, const ivector sp_adult_age0, const ivector sp_nb_age_class, const imatrix age_compute_habitat)
- void **createMatHabitat_input** (const PMap &map, const int nb_ages, const int nbt_total)
- void **createMatSpecies** (const PMap &map, int t0, int nbt, int nbi, int nbj, int nb_species, const ivector a0_adult, const ivector sp_nb_age_class)
- void **createMatEffort** (const PMap &map, int nbi, int nbj, int nb_fleet)
- void createMatCatch (const PMap &map, int nbi, int nbj, int nb_species, const IVECTOR &nb_fleet, const ivector a0_adult, const IVECTOR &nb_cohorts, const IVECTOR &nb_region)
- void **createMatMortality** (int nforage, int nbi, int nbj)
- void **MeanVarMovement** (const PMap &map, const dmatrix &Adv_x, const dmatrix &Adv_y, const dmatrix &Diff, const double mss, const double sigma_species, const double length_age, const double length_age_↩ max, const int dT, const int sp, const int age)
- void **MeanVarMortality** (const PMap &map, const dmatrix &M, const double Mp_max, const double Ms_↩ max, const double Mp_exp, const double Ms_slope, const double mean_age_in_month, const int sp, const int age)
- void **MeanVarTemperature** (const PMap &map, const int sp, const int sp_nb_cohort_lv, const int a0_adult, const int t_count)
- double **comp_waverage** (const PMap &map, const dmatrix &var, const int sp, const int age)
- double **comp_waverage2** (const PMap &map, const dmatrix &var1, const dmatrix &var2, const int sp, const int age)

Public Attributes

- DMATRIX **xlon**
- DMATRIX **ylat**
- DVECTOR **zlevel**
- IMATRIX **mask**
- DMATRIX **u**
- DMATRIX **v**
- DMATRIX **diffusion_x**
- DMATRIX **advection_x**
- DMATRIX **diffusion_y**
- DMATRIX **advection_y**
- DMATRIX **speed**
- DVECTOR **lastlat**
- DVECTOR **lat_correction**
- dmatrix **daylength**
- D3_ARRAY **np1**
- D3_ARRAY **sst**
- D3_ARRAY **ph1**
- D3_ARRAY **vld**
- D4_ARRAY **un**
- D4_ARRAY **vn**
- D4_ARRAY **tempn**
- D4_ARRAY **oxygen**

- D4_ARRAY **forage**
- D3_ARRAY **season_switch**
- D3_ARRAY **sigma_season**
- d3_array **fluxes_region**
- D3_ARRAY **mats**
- DMATRIX **Hs**
- DMATRIX **Hj**
- DMATRIX **Ha**
- DMATRIX **mat2d_NoBorder**
- D3_ARRAY **mortality**
- ivector **nb_age_built**
- dmatrix **mean_speed**
- dmatrix **mean_diffusion**
- dmatrix **mean_mortality**
- dmatrix **mean_temperature**
- D3_ARRAY **larvae**
- D3_ARRAY **juvenile**
- D3_ARRAY **young**
- D3_ARRAY **recruit**
- D3_ARRAY **adult**
- D3_ARRAY **total_pop**
- D3_ARRAY **PEB**
- d4_array **habitat_input**
- d3_array **density_input**
- D3_ARRAY **total_obs_catch**
- D3_ARRAY **total_pred_catch**
- d4_array **Ctot_proportion_fishery**
- D4_ARRAY **init_density_species**
- D5_ARRAY **F_access_sum_age**
- D5_ARRAY **density_before**
- D5_ARRAY **adult_habitat**
- D4_ARRAY **density_after**
- DVECTOR **sum_B_larvae**
- DVECTOR **sum_B_juv**
- DVECTOR **sum_B_young**
- DVECTOR **sum_B_recruit**
- DVECTOR **sum_B_adult**
- DVECTOR **sum_total_pop**
- D3_ARRAY **effort**
- D3_ARRAY **efflon**
- D3_ARRAY **efflat**
- D4_ARRAY **catch_obs**
- D4_ARRAY **catch_est**
- D4_ARRAY **C_N_sp_age_fishery**
- D4_ARRAY **C_tot_no_effort_sp_age**
- D4_ARRAY **LF_qtr_obs**
- D4_ARRAY **C_N_sp_age_fishery_qtr**
- D5_ARRAY **Sum_C_N_sp_age_fishery_area**

Friends

- class **dim**

3.3.1 Detailed Description

Seapodym matrices class.

3.3.2 Member Function Documentation

3.3.2.1 createMatCatch()

```
void CMatrices::createMatCatch (
            const PMap & map,
            int nbi,
            int nbj,
            int nb_species,
            const IVECTOR & nb_fleet,
            const ivector a0_adult,
            const IVECTOR & nb_cohorts,
            const IVECTOR & nb_region )
```

const int agemax = nb_age_class(sp);

The documentation for this class was generated from the following files:

- src/Matrices.h
- src/Matrices.cpp

3.4 CNumfunc Class Reference

Class for computing various mathematical functions.

`#include <Numfunc.h>`

Public Member Functions

- void **corcatch** (DMATRIX &xx, DMATRIX &yy, const int imin, const int imax, const ivector jinf, const ivector jsup, int &nn, double &cor, double &z, double &prob, const IMATRIX &mask, double missval)
- void **corcpue** (DMATRIX &xx, DMATRIX &yy, dmatrix &eff, const int imin, const int imax, const ivector jinf, const ivector jsup, int &nn, double &cor, double &z, double &prob, const IMATRIX &mask, double missval)
- void **corlin** (const DMATRIX &xx, const DMATRIX &yy, const int imin, const int imax, const ivector jinf, const ivector jsup, int &nn, double &cor, double &z, double &prob, double missval)
- void **summat** (const DMATRIX &xx, double &sx, const int imin, const int imax, const ivector jinf, const ivector jsup, int &nn, const double missval)
- void **sumdif** (const DMATRIX &xx, const DMATRIX &yy, const int imin, const int imax, const ivector jinf, const ivector jsup, const int nn, double sx, double sy, double &sxx, double &syy, double &sxy, const double missval)
- double **gammln** (const double xx)
- double **betacf** (double a, double b, double x)
- double **betai** (double a, double b, double x)
- void **pearsn** (const double sxx, const double syy, const double sxy, const int n, double &r, double &prob, double &z)
- double **deplete** (double fish, double f, double m)

Public Attributes

- double **sx**
- double **sy**
- double **sxx**
- double **syy**
- double **sxy**
- double **nn**
- double **missval**

3.4.1 Detailed Description

Class for computing various mathematical functions.

The documentation for this class was generated from the following files:

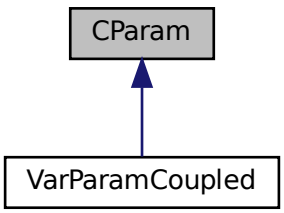
- src/Numfunc.h
- src/Numfunc.cpp

3.5 CParam Class Reference

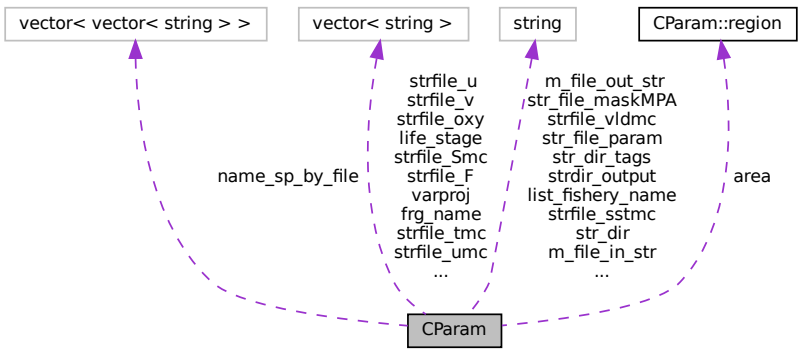
Seapodym parameter class.

```
#include <Param.h>
```

Inheritance diagram for CParam:



Collaboration diagram for CParam:



Classes

- struct region

  Structure defining the regional ID and boundaries.

Public Member Functions

- void **init_param** ()
- void **init_param_dym** ()
- void **delete_param** (bool flag)
- void **read_param** (bool &file_found)
- void **write_param** (char runtype)
- void **rbin_input2d** (string file_in, const imatrix &carte, DMATRIX &mat2d, int nbi, int nbj, int nbytetoskip)
- void **rbin_input2d** (string file_in, DMATRIX &mat2d, int nbi, int nbj, int nbytetoskip)
- void **rbin_mat2d** (string file_in, const imatrix &carte, DMATRIX &mat2d, int nlat, int nlong, int nbytetoskip)
- void **rbin_mat2d** (string file_in, DMATRIX &mat2d, int nlat, int nlong, int nbytetoskip)
- double **correction_lat** (double lat)
- double **lastlat** (int j)
- double **cell_surface_area** (int j)
- double **jtolat** (int j)
- double **itolon** (int i)
- int **lattoj** (double lat)
- int **lontoi** (double lon)
- double **func_limit_one** (const double m)
- double **dffunc_limit_one** (const double x, const double dfy)
- void **afcoef** (const double lon, const double lat, dmatrix &a, int &ki, int &kj, const int reso)
- double **selectivity_comp** (const int sp, const int age, const int f, const int k)
- void dfselectivity (double &dfslope, double &dflength, double &dfasympt, const int sp, const int age, const int f, const int k)
- void **define_regions** ()
- float **fdate** (float year, float month)
- int **get_month** (double fdate)
- int **get_year** (double fdate)
- int **get_nbi** () const

- int **get_nbj** () const
- void **set_nbt** (int nbt)
- int **get_nbt** ()
- int **get_nbspecies** ()
- int **get_nbfishery** () const
- int **get_nbforage** () const
- void **time_reading_init** ()

Public Attributes

- bool **flag_coupling**
- bool **build_forage**
- bool **flag_twin**
- bool **connectivity_comp**
- int **tuna_spinup**
- int **wbin_flag**
- int **mpa_simulation**
- int **nb_mpa**
- int **type_oxy**
- int **use_sst**
- int **use_vld**
- int **use_ph1**
- int **maxfn**
- double **crit**
- ivector **vert_movement**
- ivector **food_requirement_in_mortality**
- ivector **uncouple_sst_larvae**
- ivector **gaussian_thermal_function**
- ivector **cannibalism**
- string **idformat**
- int **idfunc**
- imatrix **like_types**
- bool **cpue**
- dmatrix **like_param**
- dmatrix **prob_zero**
- ivector **tag_like**
- ivector **stock_like**
- dvector **mean_stock_obs**
- dvector **stock_lonmin**
- dvector **stock_lonmax**
- dvector **stock_latmin**
- dvector **stock_latmax**
- ivector **frq_like**
- dvector **eff_units_converter**
- dvector **cpue_mult**
- double **total_like**
- int **fdata_rm**
- int **use_lf_regstruc**
- int **use_mask_catch**
- string $*$ **parfile_names**
- int **nb_varproj**
- ivector **varproj_nsteps**
- vector$<$ string $>$ **varproj**
- dvector **statpars**

- int **_nstatpars**
- adstring_array **statpar_names**
- double **longitudeMin**
- double **longitudeMax**
- double **latitudeMin**
- double **latitudeMax**
- double **deltaX**
- double **deltaY**
- int **deltaT**
- int **nlevel**
- double **startdate**
- double **enddate**
- int **ndatini**
- int **ndatfin**
- int **date_mode**
- ivector **rundates**
- int **nbytetoskip**
- double **save_first_yr**
- double **save_last_yr**
- int **first_recruitment_date**
- int **nb_yr_forecast**
- int **nbsteptoskip**
- int **nlong**
- int **nlat**
- int **iterationNumber**
- int **nb_layer**
- DVECTOR **source_frg**
- IVECTOR **day_layer**
- IVECTOR **night_layer**
- double **lambda**
- double **E**
- double **c_pp**
- double **pp_transform**
- double **sigma_fcte**
- int **inv_lambda_max**
- double **inv_lambda_curv**
- int **Tr_max**
- double **Tr_exp**
- string **str_file_mask**
- string **str_file_topo**
- string **str_file_maskEEZ**
- string **str_file_maskMPA**
- string **str_file_param**
- string **str_dir**
- string **str_dir_forage**
- string **str_dir_init**
- string **str_dir_fisheries**
- string **str_dir_tags**
- string **strfile_pp**
- string **strfile_sst**
- string **strfile_vld**
- string **strfile_ph1**
- vector< string > **frg_name**
- vector< string > **sp_name**
- vector< string > **strfile_F**

- vector< string > **strfile_Fmc**
- vector< string > **strfile_S**
- vector< string > **strfile_Smc**
- string **strfile_ppmc**
- string **strfile_sstmc**
- string **strfile_vldmc**
- vector< string > **strfile_u**
- vector< string > **strfile_v**
- vector< string > **strfile_t**
- vector< string > **strfile_oxy**
- vector< string > **strfile_umc**
- vector< string > **strfile_vmc**
- vector< string > **strfile_tmc**
- vector< string > **strfile_oxymc**
- string **strdir_output**
- int **write_all_cohorts_dym**
- int **write_all_fisheries_dym**
- vector< string > life_stage
- ivector **sp_nb_cohort_life_stage**
- ivector **sp_nb_cohorts**
- ivector **sp_nb_cohort_lv**
- ivector **sp_nb_cohort_jv**
- ivector **sp_nb_cohort_ad**
- ivector **sp_a0_adult**
- imatrix **sp_unit_cohort**
- DMATRIX length
- DMATRIX **length_bins**
- DMATRIX **weight**
- DVECTOR **M_inc_ph_a**
- DVECTOR **M_inc_ph_b**
- DVECTOR **Mp_mean_max**
- DVECTOR **Mp_mean_exp**
- DVECTOR **Ms_mean_slope**
- DVECTOR **Ms_mean_max**
- DVECTOR **M_mean_range**
- dvector **residual_competition**
- int **habitat_run_type**
- int **nb_habitat_run_age**
- ivector **habitat_run_age**
- int **migrations_by_maturity_flag**
- IVECTOR **age_mature**
- DMATRIX **maturity_age**
- IVECTOR **age_autonomous**
- IVECTOR **age_recruit**
- imatrix **age_compute_habitat**
- DVECTOR **nb_recruitment**
- DVECTOR **a_adults_spawning**
- ivector **seasonal_migrations**
- dvector **spawning_season_peak**
- dvector **spawning_season_start**
- DVECTOR **a_sst_spawning**
- DVECTOR **b_sst_spawning**
- DVECTOR **a_sst_larvae**
- DVECTOR **b_sst_larvae**
- DVECTOR **alpha_hsp_prey**

- DVECTOR **alpha_hsp_predator**
- DVECTOR **beta_hsp_predator**
- DVECTOR **a_sst_habitat**
- DVECTOR **b_sst_habitat**
- DVECTOR **T_age_size_slope**
- dmatrix **thermal_func_delta**
- DVECTOR **a_oxy_habitat**
- DVECTOR **b_oxy_habitat**
- dmatrix **eF_habitat**
- DVECTOR **hp_cannibalism**
- DVECTOR **forage_ration**
- DVECTOR **sigma_species**
- DVECTOR **MSS_species**
- DVECTOR **MSS_size_slope**
- DVECTOR **c_diff_fish**
- dmatrix **sigma_ha**
- dmatrix **temp_age**
- string ∗ **list_fishery_name**
- dvector **fishery_reso**
- float **catch_reso**
- ivector **fishery_catch_units**
- IVECTOR **nb_fishery_by_sp**
- IMATRIX **mask_fishery_sp**
- IMATRIX **mask_fishery_sp_no_effort**
- IMATRIX **mask_fishery_sp_like**
- int **nb_fishery_type**
- ivector **fisheries_no_effort_exist**
- int **actual_eff**
- ivector **mpa_scenario**
- ivector **mpa_ID**
- ivector **mpa_S1_X**
- ivector **mpa_fishery**
- IVECTOR **type_each_fishery**
- IVECTOR **list_fishery_type**
- IVECTOR **nb_fishery_type_sp**
- IMATRIX **list_fishery_type_sp**
- DMATRIX **q_sp_fishery**
- dvector **q_dyn_fishery**
- ivector **s_func_type**
- DMATRIX **s_slope_sp_fishery**
- DMATRIX **s_length_sp_fishery**
- DMATRIX **s_asympt_sp_fishery**
- D3_ARRAY **selectivity_sp_fishery_age**
- vector< vector< string > > **name_sp_by_file**
- vector< string > **file_catch_data**
- vector< string > **file_frq_data**
- vector< string > **file_tag_data**
- int **nb_catch_files**
- int **nb_frq_files**
- int **nb_tag_files**
- int **tag_gauss_kernel_on**
- int **dx_tags**
- int **dy_tags**
- float **lonmin_tags**
- float **lonmax_tags**

- float **latmin_tags**
- float **latmax_tags**
- bool **tags_only**
- string **m_file_in_str**
- string **m_file_out_str**
- double **m_f**
- int **nb_region**
- IVECTOR **nb_region_sp_B**
- IVECTOR **nb_region_fishery**
- IMATRIX **area_sp_B**
- region ∗∗ **area**
- int **nb_EEZ**
- IVECTOR **EEZ_ID**
- string ∗ **EEZ_name**
- double **elapsed_time_reading**

Protected Attributes

- int **nbt_total**
- int **nbi**
- int **nbj**
- int **nb_species**
- int **nb_forage**
- int **nb_fishery**

3.5.1 Detailed Description

Seapodym parameter class.

All static SEAPODYM parameters are defined and described here. For the DVAR parameters see class VarParamCoupled

3.5.2 Member Function Documentation

3.5.2.1 dfselectivity()

```
void CParam::dfselectivity (
            double & dfslope,
            double & dflength,
            double & dfasympt,
            const int sp,
            const int age,
            const int f,
            const int k )
```

Adjoint code for selectivity functions (Param class) Forward functions are in Param.cpp

3.5.3 Member Data Documentation

3.5.3.1 length

`DMATRIX CParam::length`

DMATRIX juv_length; // length by age for each species (cm) for the first three months of live DMATRIX juv_weight; // weight by age for each species (kg) for the first three months of live

3.5.3.2 life_stage

`vector<string> CParam::life_stage`

IVECTOR sp_nb_age_class_ad; // number of age classes for each species [sp] IVECTOR sp_unit_age_class_ad; // time step used for the population of the species [sp] (0= pas de calcul de pop; 1=month;2=quarter) IMATRIX sp_unit_age_class; // time step (in days) used for the population of the species [sp] and cohort [a] IVECTOR sp_↩ nb_age_class_jv; // number of age classes for each species [sp] IVECTOR sp_unit_age_class_jv; // time step used for the population of the species [sp] (0= pas de calcul de pop; 1=month;2=quarter) int max_age_class; // max number of age classes over all species

The documentation for this class was generated from the following files:

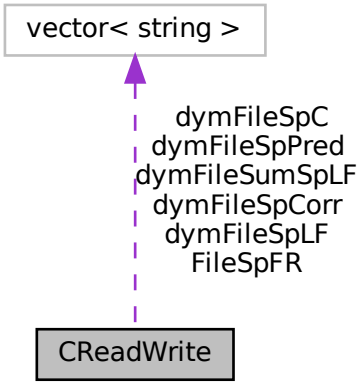
- src/Param.h
- src/dv_selectivity.cpp
- src/Param.cpp
- src/VarParamCoupled.cpp

3.6 CReadWrite Class Reference

IO class.

`#include <ReadWrite.h>`

Collaboration diagram for CReadWrite:



Public Member Functions

- void **rbin_headpar** (string file_in, int &nlong, int &nlat, int &nlevel)
- void **rtxt_headpar** (string file_in, int &nlong, int &nlat, int &nlevel)
- void **rwbin_minmax** (string file_io, double minvalstep, double maxvalstep)
- void **rtxt_mat2d** (string file_in, DMATRIX &mat2d, int &nlong, int &nlat)
- void **rbin_header** (string file_in, string &idformat, int &idfunc, double &minval, double &maxval, int nlong, int nlat, int nlevel, double &startdate, double &enddate, DMATRIX &xlon, DMATRIX &ylat, DVECTOR &zlevel, IMATRIX &msksp)
- void **wbin_header** (string file_out, string &idformat, int &idfunc, double &minval, double &maxval, int nlong, int nlat, int nlevel, double &startdate, double &enddate, const DMATRIX &xlon, const DMATRIX &ylat, const DVECTOR &zlevel, const IMATRIX &msksp)
- void **rtxt_header** (string file_in, int nlong, int nlat, int nlevel, double &startdate, double &enddate, DVECTOR &xlon, DVECTOR &ylat, DVECTOR &zlevel, IMATRIX &msksp)
- void **rbin_mat2d** (string file_out, PMap &map, DMATRIX &mat2d, int nlat, int nlong, int nbytetoskip)
- void **rbin_input2d** (string file_in, PMap &map, DMATRIX &mat2d, int nbi, int nbj, int nbytetoskip)
- void **wbin_mat2d** (string file_out, const DMATRIX &mat2d, int nlat, int nlong, bool FILEMODE)
- void **wbin_transpomat2d** (string file_out, const DMATRIX &mat2d, int nlong, int nlat, bool FILEMODE)
- void **wtxt_header** (string file_out, int nlong, int nlat, int nlevel, double &startdate, double &enddate, const DVECTOR &xlon, const DVECTOR &ylat, const DVECTOR &zlevel, const IMATRIX &msksp)
- void **wtxt_mat2d** (string file_out, const DMATRIX &mat2d, int nlat, int nlong, bool FILEMODE)
- void **rtxt_col_lonlat** (string file_in, DMATRIX &mat2d, int nlong, int nlat, DVECTOR &xlon, DVECTOR &ylat, int nbvar, int var)
- void **wbin_fishery** (string file_in, string file_out, int nbvar)
- void **rbin_fishery** (string file_in, DMATRIX &mat2d, CParam &param, int nbvar, int nvar, int yyyy, int mm)
- void **InitSepodymFileDym** (CParam &param, CMatrices &mat, int nb_mo, DVECTOR &zlevel, const IMA↩ TRIX &msksp)
- void **SaveSepodymFileDym** (CParam &param, PMap &map, CMatrices &mat)
- void **SaveDymFile** (PMap &map, CMatrices &mat, string file, const dmatrix &data, const int nlon, const int nlat)
- void **InitFluxesCohortsFileTxt** (CParam &param)
- void **SaveFluxesCohortsFileTxt** (CParam &param, CMatrices &mat, PMap &map, int day, int month, int year)
- void **InitSepodymFileTxt** (CParam &param)
- void **SaveSepodymFileTxt** (CParam &param, CMatrices &mat, PMap &map, dvector sumP, DVECTOR &sumF, DVECTOR &sumFprime, DVECTOR &sumF_area_pred, DVECTOR &sumF_required_by_sp, DV↩ ECTOR &mean_omega_sp, int day, int mois2, int yr2, int t_total, int qtr1, int qtr2, int nbi, int nbj)
- void **rbin_fishery_header** (CParam &param)
- void **rtxt_fishery_data** (CParam &param, const PMap &map, const int nbt, const int jday_spinup)
- void **set_effort_rm** (CParam &param, PMap &map, const int nbt, const int jday_spinup)
- void **degrade_fishery_reso** (CParam &param, PMap &map, const int nbt, const int jday_spinup)
- void **set_frec_rm** (CParam &param, const PMap &map, const int nbt, const int jday_spinup)
- void **set_frec_rm_no_effort_fisheries** (CParam &param, const PMap &map, const int nbt, const int jday↩ _spinup)
- void **delete_fisheries_rec** (void)
- void **get_catch** (CParam &param, dmatrix &catch_obs, const int f, int y, const int m, const int sp)
- void **get_effort** (CParam &param, dmatrix &effort, const int f, int y, const int m)
- void **get_effort_lonlat** (CParam &param, dmatrix &effort, dmatrix &efflon, dmatrix &efflat, const int f, int y, const int m)
- void **get_effort_rm** (CParam &param, dmatrix &effort, const int f, int y, const int m)
- void **get_fishery_data** (CParam &param, D3_ARRAY &effort, D4_ARRAY &catch_obs, int y, const int m)
- void **get_fishery_data** (CParam &param, D3_ARRAY &effort, D4_ARRAY &catch_obs, D3_ARRAY &efflon, D3_ARRAY &efflat, int y, const int m)
- void **get_average_effort** (CParam &param, D3_ARRAY &effort, D3_ARRAY &efflon, D3_ARRAY &efflat, const int nby, const int m)
- void **get_average_effort_rm** (CParam &param, dmatrix &effort, const int f, const int nby, const int m)

- void **get_average_selectivity** (PMap &map, CParam &param, dvector &swa, const ivector fisheries, const int nbf, const int nbt, const int nb_ages, const int sp, const int step_count)
- void **get_fishery_data_mpa** (PMap &, CParam &, d3_array &, d4_array &, d3_array &, d3_array &, int, int)
- void **mpa_areas_comp** (PMap &, CParam &)
- void **inc_obs_catch_mpa** (PMap &map, CParam &param, dmatrix &catch_obs, const int sp)
- int **get_numrec** (const int f, const int y, const int m)
- void read_lf_WCPO (CParam &param, string filename, const float startdate, const float enddate, const int sp)
- void read_lf_EPO (CParam &param, string filename, const float startdate, const float enddate, const int sp)
- void **read_lf_fine** (CParam &param, string filename, const float startdate, const float enddate, const int sp)
- void **read_frq_data** (CParam &param, PMap &map, const float startdate, const float enddate, const int sp)
- void **get_LF_qtr_data** (CParam &param, d4_array LF_qtr_obs, int y, const int q)
- void write_frq_data (CParam &param, int sp, int year, int qtr, d3_array frq, bool FILEMODE)
- void read_pred_frq_data (CParam &param, string filename, const float startdate, const float enddate, const int sp)

Public Attributes

- vector< string > **dymFileSpPred**
- vector< string > **dymFileSpC**
- vector< string > **dymFileSpLF**
- vector< string > **dymFileSumSpLF**
- vector< string > **dymFileSpCorr**
- vector< string > **FileSpFR**

Friends

- class **fishery_record**
- class **fishing_effort**

3.6.1 Detailed Description

IO class.

Class functions are accessible through all computational classes. All types of input data are read here, any new output writing routines must be placed here as well.

3.6.2 Member Function Documentation

3.6.2.1 read_lf_EPO()

```
void CReadWrite::read_lf_EPO (
            CParam & param,
            string filename,
            const float startdate,
            const float enddate,
            const int sp )
```

const int nb_ages = param.sp_nb_age_class_ad[sp];

3.6.2.2 read_lf_WCPO()

```
void CReadWrite::read_lf_WCPO (
            CParam & param,
            string filename,
            const float startdate,
            const float enddate,
            const int sp )
```

double L_pr = param.juv_length(sp,param.sp_nb_age_class_jv[sp]-1);

3.6.2.3 read_pred_frq_data()

```
void CReadWrite::read_pred_frq_data (
            CParam & param,
            string filename,
            const float startdate,
            const float enddate,
            const int sp )
```

int nb_ages = param.sp_nb_age_class_ad[sp];

3.6.2.4 write_frq_data()

```
void CReadWrite::write_frq_data (
            CParam & param,
            int sp,
            int year,
            int qtr,
            d3_array frq,
            bool FILEMODE )
```

int nb_ages = param.sp_nb_age_class_ad[sp];

The documentation for this class was generated from the following files:

- src/ReadWrite.h
- src/ReadWrite_DYM.cpp
- src/ReadWrite_fisheries.cpp
- src/ReadWrite_TXT.cpp

3.7 CSaveTimeArea Class Reference

The class to aggregate variables to the regional structure.

`#include <SaveTimeArea.h>`

Public Member Functions

- void **SumByArea** (const PMap &map, const dmatrix &mask_catch, const dmatrix &mat2d, dvector &sum↩ _area, const dvector cell_area, const int nb_reg, const int nbt)
- void **SumByEEZ** (const CParam &param, const PMap &map, const DMATRIX &mat2d, DVECTOR &sum↩ _EEZ, const dvector cell_area)
- double **SumByEEZ** (const PMap &map, const int EEZ_ID, const DMATRIX &mat2d, const dvector cell_area, const int nlon, const int nlat)
- int **NobsByEEZ** (const PMap &map, const int EEZ_ID, const DMATRIX &mat2d, const int nlon, const int nlat)
- double **SumByEEZ** (const PMap &map, const int EEZ_ID, const DMATRIX &C, const DMATRIX &E, const int nlon, const int nlat)
- double **StdCPUEByEEZ** (const PMap &map, const int EEZ_ID, const DMATRIX &C, const DMATRIX &E, const double mean, const int nobs, const int nlon, const int nlat)

3.7.1 Detailed Description

The class to aggregate variables to the regional structure.

The documentation for this class was generated from the following files:

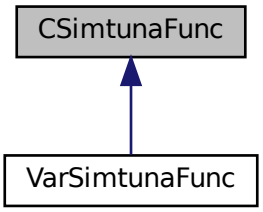
- src/SaveTimeArea.h
- src/SaveTimeArea.cpp

3.8 CSimtunaFunc Class Reference

The simulation function which do not use dvariables.

`#include <SimtunaFunc.h>`

Inheritance diagram for CSimtunaFunc:



Public Member Functions

- double **function_lambda** (CParam &param, CMatrices &mat, int n, int i, int j)
- double **daylength** (double lat, int jday)
- double **daylength_twilight** (double lat, int jday, const double p)
- double **grad_daylength** (double lat, int jday)
- double **f_accessibility_comp** (const double Od, const double On, const double Td, const double Tn, double twosigsq, double temp_mean, double oxy_teta, double oxy_cr, const double DL)

3.8.1 Detailed Description

The simulation function which do not use dvariables.

The documentation for this class was generated from the following files:

- src/SimtunaFunc.h
- src/SimtunaFunc.cpp

3.9 Date Class Reference

Class written by J.Jouanno to handle date format.

```
#include <Date.h>
```

Static Public Member Functions

- static void **init_time_variables** (CParam &param, int &Tr_step, int &nbt_spinup_tuna, int &jday_run, int &jday_spinup, int &nbstot, const int info, const int flagsimu)
- static void **update_time_variables** (const int t_count, const int deltaT, const int date_mode, const int jday↩ _spinup, int &jday, int &day, int &month, int &year, int &newyear)
- static int **get_nbstot** (const int ndat000, const int ndatfin, const int jdays_run, const int deltaT, const int date_mode, ivector &rundates)
- static int **get_nbt_before_first_recruitment** (const int first_recruitment_date, const int ndatini, const int deltaT, const int date_mode)
- static int **dym_startdate_run** (CParam &param, const dvector zlevel_dym, const int nbstot)
- static void **zlevel_run** (CParam &param, const dvector zlevel_dym, const int nbstot, dvector &zlevel, const int nbt_start_series)
- static int **leapYear** (int year)
- static int **dayWithinMonth** (int day, int month, int year)
- static unsigned long **julday** (int day, int month, int year)
- static unsigned long **clmjulday** (int day, int month, int year)
- static unsigned long **nlyjulday** (int day, int month, int year)
- static unsigned long **juldayy** (int day, int month, int year)
- static unsigned long **clmjuldayy** (int day, int month, int year)
- static unsigned long **nlyjuldayy** (int day, int month, int year)
- static void **dmy** (unsigned long julnum, int &d, int &m, int &y)
- static void **clmdmy** (unsigned long julnum, int &d, int &m, int &y)
- static void **nlydmy** (unsigned long julnum, int &d, int &m, int &y)
- static void **idatymd** (const int ndat, int &year, int &month, int &day)
- static string **MakeDate** (int yr, int mo, int jr)
- static string **MakeDate** (int yr, int mo)
- static string **MonthName** (int mo)
- static int **Update_now_time** (int yr, int month, int day)
- static string **Update_now_time_str** (int yr, int month, int day)
- static string **Update_now_time_str_spinup** (int month)

3.9.1 Detailed Description

Class written by J.Jouanno to handle date format.

The model supports three date formats depending on the calendar: 360-day year (most frequently used), 365-day year and standard calendar with leap years.

The documentation for this class was generated from the following files:

- src/Date.h
- src/Date.cpp

3.10 fishery_record Class Reference

Class that reads and stores all fishing data.

`#include <ReadWrite.h>`

Public Member Functions

- int get_i ()

  2015: catch for a single species, can be modified to multispecies if needed
- int **get_j** ()
- double **get_lon** ()
- double **get_lat** ()
- double **get_effort** (void)
- double **get_efflon** (void)
- double **get_efflat** (void)
- double **get_catch** (void)
- void **set_record** (double longitude, double latitude, int ii, int jj, double ee, double cc)
- void **change_coord** (double longitude, double latitude, int ii, int jj)

3.10.1 Detailed Description

Class that reads and stores all fishing data.

Fishing data to be stored in SEAPODYM: i,j indices, lon/lat coordinates (center of the fishing area), effort and catch

The documentation for this class was generated from the following file:

- src/ReadWrite.h

3.11 fishing_effort Class Reference

Class that reads and stores redistributed fishing effort data.

`#include <ReadWrite.h>`

Public Member Functions

- int get_i ()
  fishing effort
- int **get_j** ()
- double **get_effort** (void)
- void **set_effort** (int ii, int jj, double ee)

3.11.1 Detailed Description

Class that reads and stores redistributed fishing effort data.

Fishing effort redistributed to the model resolution will be read and used in Calpop class for each i,j to compute fishing mortality rates.

The documentation for this class was generated from the following file:

- src/ReadWrite.h

3.12 PMap Class Reference

Class managing spatial domain and grid: the land mask, the indexing and the boundaries.

`#include <Map.h>`

Public Member Functions

- void **lit_map** (CParam &param)
- void **delete_map** (const CParam &param)
- void **reg_indices** (CParam &param)

Public Attributes

- IMATRIX **bord_cell**
- IMATRIX **nbl_bord_cell**
- IMATRIX **carte**
- DMATRIX **itopo**
- IMATRIX **maskEEZ**
- IMATRIX **maskMPA**
- int **imin**
- int **imax**
- int **jmin**
- int **jmax**
- int **imin1**
- int **imax1**
- int **global**
- IVECTOR **iinf**
- IVECTOR **isup**
- IVECTOR **jinf**
- IVECTOR **jsup**
- IVECTOR **jinf1**
- IVECTOR **jsup1**
- ivector **regimin**
- ivector **regimax**
- ivector **regjmin**
- ivector **regjmax**

3.12.1 Detailed Description

Class managing spatial domain and grid: the land mask, the indexing and the boundaries.

This class reads land mask, EEZ mask (if exist) and topographic indices. The boundary conditions are defined here as well using the land mask information. Also, the ragged array indices are computed and stored in this class.

The documentation for this class was generated from the following files:

- src/Map.h
- src/Map.cpp

3.13 CParam::region Struct Reference

Structure defining the regional ID and boundaries.

`#include <Param.h>`

Public Attributes

- int **area_id**
- double **lgmin**
- double **lgmax**
- double **ltmin**
- double **ltmax**

3.13.1 Detailed Description

Structure defining the regional ID and boundaries.

The derived instance $**$area is used for regional extractions, mostly in the IO routines.

The documentation for this struct was generated from the following file:

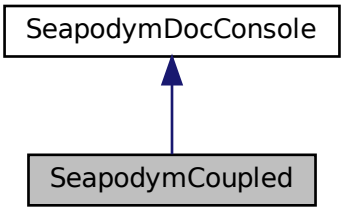
- src/Param.h

3.14 SeapodymCoupled Class Reference

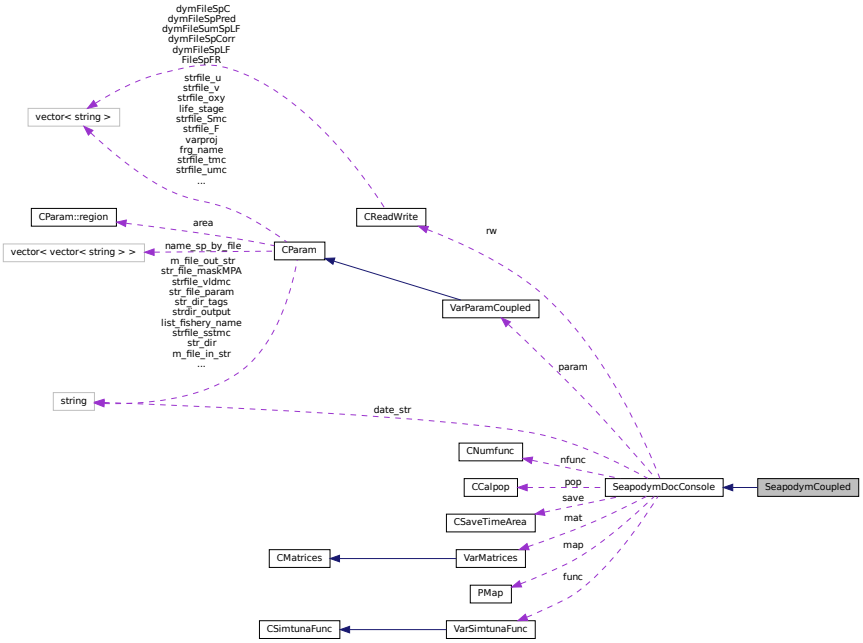
The main simulation class.

`#include <SeapodymCoupled.h>`

Inheritance diagram for SeapodymCoupled:



Collaboration diagram for SeapodymCoupled:



Public Member Functions

- **SeapodymCoupled** (const char ∗parfile)
- int **nvarcalc** () const
- void **xinit** (dvector &x, adstring_array &names)
- double **run_coupled** (dvar_vector x, const bool writeoutputfiles=false)

- double **run_habitat** (dvar_vector x, const bool writeoutputfiles=false)
- double **run_density** (dvar_vector x, const bool writeoutputfiles=false)
- dvariable **reset** (dvar_vector x)
- void **write** (const char ∗parfile)
- void **save_statistics** (const string dirout, const adstring_array x_names, double likelihood, dvector g, double elapsed_time, int status, int iter, int nvars)
- int **EditRunCoupled** (const char ∗parfile)
- double OnRunCoupled (dvar_vector x, const bool writeoutputfiles=false)

  The tuna population main loop is in this function.
- void **OnSimulationEnd** ()
- double OnRunHabitat (dvar_vector x, const bool writeoutputfiles=false)

  The main loop of habitat simulations.
- void **ReadHabitat** ()
- double OnRunDensity (dvar_vector x, const bool writeoutputfiles=false)

  The tuna population simulation without fishing and density fitting.
- void **ReadDensity** ()
- void **OnRunFirstStep** ()
- void **OnBuildForage** ()
- double **get_total_time_reading** ()
- int **get_maxfn** ()
- double **get_crit** ()

Friends

- class **tag_release**

Additional Inherited Members

3.14.1 Detailed Description

The main simulation class.

3.14.2 Member Function Documentation

3.14.2.1 OnRunCoupled()

```
double SeapodymCoupled::OnRunCoupled (
            dvar_vector x,
            const bool writeoutputfiles = false )
```

The tuna population main loop is in this function.

This is the main loop function including the calculation of biomass exchange between regions, based on the one time step simulations with non-zero biomass only in the donor region and quantification of biomass changes in all. See SeapodymCoupled_OnRunCoupled.cpp for the description of the main loop

This is the main loop function. It includes the following calls: 1- Initialising population density 2- Reading all forcing data (once in optimization mode, at every time step in simulation mode) 3- Reading fisheries data 4- Reading tagging data if tag_like is activated 5- Age/lifestage loop calling the ADRE solvers and ageing 6- Predicting observed variables (catch, LF and density of tags) 7- Likelihood computation 8- Writing outputs (in simulation mode only)

This is the main loop function for the SAVE-BEFORE-FISHING simulation mode. The default function includes the following calls: 1- Initialising population density 2- Reading all forcing data (once in optimization mode, at every time step in simulation mode) 3- Reading fisheries data 4- Reading tagging data if tag_like is activated 5- Age/lifestage loop calling the ADRE solvers and ageing 6- Predicting observed variables (catch, LF and density of tags) 7- Likelihood computation 8- Writing outputs (in simulation mode only) Here to the default function added the second ADRE solver in order to: 1) Solve the ADREs without fishing using the state vector of model with fishing at T-1 2) Save the outputs, which correspond to the model solution without fishing mortality 3) Restore the model-with-fishing state vector and get the ADRE solution for T. Note, to activate this simulation mode, use this file instead of SeapodymCoupled_OnRunCoupled in Makefile or Makefile.clt. If latter, only simulation mode is supported for this run.
if (t_count > nbt_spinup_forage + nt_jv){ SPINUP TO BE FIXED OR REMOVED!!!

}

if (t_count > nbt_spinup_forage + nt_yn){ TO BE FIXED!!! for (int age=0; age<=nb_age_built[sp]; age++){///TO BE FIXED!!!

3.14.2.2 OnRunDensity()

```
double SeapodymCoupled::OnRunDensity (
            dvar_vector x,
            const bool writeoutputfiles = false )
```

The tuna population simulation without fishing and density fitting.

This is the main loop for the model without fishing and fitting of density. Similar to the default function, it includes the following calls: 1- Initialising population density 2- Reading all forcing data (once in optimization mode, at every time step in simulation mode) 3- Age/lifestage loop calling the ADRE solvers and ageing 4- Computing model density -> used as predictions 5- Likelihood computation using input density as observations 6- Writing outputs (in simulation mode only) Note, there is no modelling of tagged cohorts here! if (t_count > nbt_spinup_forage + nt_yn){ TO BE FIXED!!! for (int age=0; age<=nb_age_built[sp]; age++){///TO BE FIXED!!!

The documentation for this class was generated from the following files:

- src/SeapodymCoupled.h
- src/dv_food_requirement_index.cpp
- src/dv_spawning.cpp
- src/dv_survival.cpp
- src/dv_total_pop.cpp
- src/fd_food_requirement_index.cpp
- src/fd_spawning.cpp
- src/fd_survival.cpp
- src/fd_total_pop.cpp
- src/food_requirement_index.cpp
- src/like.cpp
- src/Seapodym_OnRunDensity.cpp
- src/Seapodym_OnRunHabitat.cpp
- src/SeapodymCoupled_EditRunCoupled.cpp

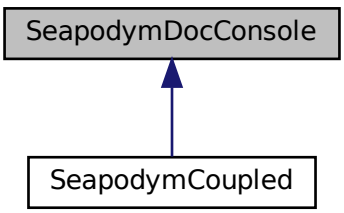
- src/SeapodymCoupled_Forage.cpp
- src/SeapodymCoupled_Funcs.cpp
- src/SeapodymCoupled_OnCompFluxes.cpp
- src/SeapodymCoupled_OnReadForcing.cpp
- src/SeapodymCoupled_OnRunCoupled.cpp
- src/SeapodymCoupled_OnRunFirstStep.cpp
- src/SeapodymCoupled_OnWriteOutput.cpp
- src/SeapodymCoupled_ReadTags.cpp
- src/SeapodymCoupled_SaveBeforeFishing.cpp
- src/spawning.cpp

3.15 SeapodymDocConsole Class Reference

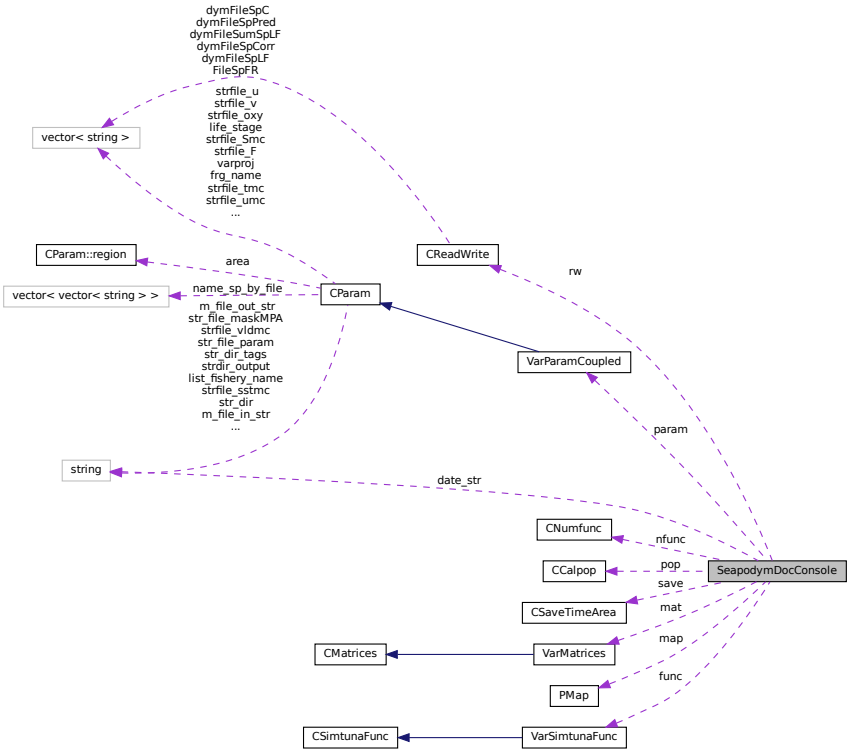
This class derives all necessary classes for the main simulation class.

```
#include <SeapodymDocConsole.h>
```

Inheritance diagram for SeapodymDocConsole:



Collaboration diagram for SeapodymDocConsole:



Public Attributes

- CReadWrite **rw**
- VarParamCoupled ∗ **param**
- VarMatrices **mat**
- PMap **map**
- VarSimtunaFunc **func**
- CNumfunc **nfunc**
- CSaveTimeArea **save**
- int **nbi**
- int **nbj**
- int **nlon**
- int **nlat**
- int **deltaT**
- int **nlon_input**
- int **nlat_input**
- double **deltaX**
- double **deltaY**
- double **SUM_CATCH**
- int **nb_fishery**
- int **nb_species**
- int **nb_forage**
- int **nb_layer**
- int **tuna_spinup**
- string **date_str**

- char **runtype**
- int **t_count**
- int **t_series**
- double **sumP**
- DVECTOR **sumF**
- DVECTOR **sumFprime**
- DVECTOR **sumF_area_pred**
- DVECTOR **sumF_required_by_sp**
- DVECTOR **mean_omega_sp**

Protected Member Functions

- void **UpdateDisplay** ()

Protected Attributes

- CCalpop **pop**

3.15.1 Detailed Description

This class derives all necessary classes for the main simulation class.

The documentation for this class was generated from the following files:

- src/SeapodymDocConsole.h
- src/SeapodymDocConsole_UpdateDisplay.cpp

3.16 tag_release Class Reference

Class handling tag releases.

`#include <SeapodymCoupled.h>`

Public Member Functions

- int **get_i** ()
- int **get_j** ()
- int **get_age** (void)
- void **set_release** (int ii, int jj, int aa)

3.16.1 Detailed Description

Class handling tag releases.

It stores and returns the information about release position and fish age at release.

The documentation for this class was generated from the following file:

- src/SeapodymCoupled.h

3.17 Utilities Class Reference

Old SEAPODYM class containing conversions and array handling functions.

`#include <Utilities.h>`

Static Public Member Functions

- static string **MakeDate** (int yr, int mo, int jr)
- static string **MakeDate** (int yr, int mo)
- static string **MonthName** (int mo)
- static string **itoa** (int i)
- static int **MyMax** (int a, int b)
- static double **MyMax** (double a, double b)
- static short **MyMax** (short a, short b)
- static char **MyMax** (char a, char b)
- static int **MyMin** (int a, int b)
- static double **MyMin** (double a, double b)
- static short **MyMin** (short a, short b)
- static char **MyMin** (char a, char b)
- static int * **create1d** (int *mat, const int n1, const int val=0)
- static double * **create1d** (double *mat, int n1, double val=0)
- static string * **create1d** (string *mat, int n1, string val="")
- static double ** **create2d** (double **mat, int n1, int n2, double val=0)
- static double ** **create2d** (double **mat, int n1, const IVECTOR &n2, double val=0)
- static string ** **create2d** (string **mat, int n1, const IVECTOR &n2, string val="")
- static int ** **create2d** (int **mat, int n1, int n2, int val=0)
- static int ** **create2d** (int **mat, int n1, const IVECTOR &n2, int val=0)
- static double *** **create3d** (double ***mat, int n1, int n2, int n3, double val=0)
- static double *** **create3d** (double ***mat, int n1, const IVECTOR &n2, const IVECTOR &n3, double val=0)
- static int *** **create3d** (int ***mat, int n1, int n2, int n3, int val=0)
- static double **** **create4d** (double ****mat, int n1, int n2, int n3, int n4, double val=0)
- static double **** **create4d** (double ****mat, int n1, const IVECTOR &n2, int n3, int n4, double val=0)
- static double **** **create4d** (double ****mat, int n1, const IVECTOR &n2, const IVECTOR &n3, const IVECTOR &n4, double val=0)
- static double ***** **create5d** (double *****mat, int n1, const IVECTOR &n2, int n3, const IVECTOR &n4, const IVECTOR &n5, double val=0)
- static void **delete1d** (string *mat)
- static void **delete1d** (const IVECTOR &mat)
- static void **delete1d** (double *mat)
- static void **delete2d** (double **mat, int n1)
- static void **delete2d** (int **mat, int n1)
- static void **delete2d** (string **mat, int n1)
- static void **delete3d** (double ***mat, int n1, int n2)
- static void **delete3d** (double ***mat, int n1, const IVECTOR &n2)
- static void **delete3d** (int ***mat, int n1, int n2)
- static void **delete4d** (double ****mat, int n1, int n2, int n3)
- static void **delete4d** (double ****mat, int n1, const IVECTOR &n2, int n3)
- static void **delete4d** (double ****mat, int n1, const IVECTOR &n2, const IVECTOR &n3)
- static void **delete5d** (double *****mat, int n1, const IVECTOR &n2, int n3, const IVECTOR &n4)

3.17.1 Detailed Description

Old SEAPODYM class containing conversions and array handling functions.

Most of the functions to handle multi-dimensional array of doubles (de)allocation are currently handled by Autodif classes and functions, so only functions for the arrays of string are used.

The documentation for this class was generated from the following file:

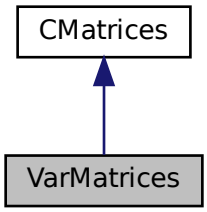
- src/Utilities.h

3.18 VarMatrices Class Reference

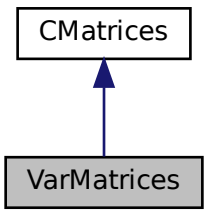
Seapodym DVAR matrices class.

`#include <VarMatrices.h>`

Inheritance diagram for VarMatrices:



Collaboration diagram for VarMatrices:



Public Member Functions

- void **CreateMatHabitat** (PMap &map, const int nb_species, const int nforage, const int nblayer, const int nb↩ _ages, int t0, int nbt, const int nbi, const int nbj, const ivector sp_adult_age0, const ivector sp_nb_age_class, const imatrix age_compute_habitat)
- void **CreateMatTransport** (PMap &map, const int nbi, const int nbj)
- void CreateMatSpecies (PMap &map, int t0, int nbt, int nbi, int nbj, int nb_species, const ivector a0_adult, const ivector &sp_nb_cohorts)

  void CreateMatSpecies(PMap& map, int nbi, int nbj, int nb_species, const ivector& sp_nb_age_class_jv, const ivector& sp_nb_age_class) {
- void **CreateMatCatch** (PMap &map, int nbi, int nbj, int nb_species, const IVECTOR &nb_fleet, const ivector a0_adult, const IVECTOR &nb_cohorts, const IVECTOR &nb_region)

Public Attributes

- dvar_matrix **dvarsU**
- dvar_matrix **dvarsV**
- DVAR4_ARRAY **dvarF_access**
- DVAR4_ARRAY **dvarZ_access**
- DVAR4_ARRAY **dvarDensity**
- DVAR4_ARRAY **dvarCatch_est**
- DVAR4_ARRAY **dvarLF_est**
- dvar4_array **dvarCtot_age_obs**
- dvar4_array **dvarCtot_age_est**
- dvar3_array **dvarSeasonSwitch**
- dvar3_array **dvarSigmaSeason**
- dvar_matrix **dvarsDiffusion_x**
- dvar_matrix **dvarsDiffusion_y**
- dvar_matrix **dvarsAdvection_x**
- dvar_matrix **dvarsAdvection_y**

3.18.1 Detailed Description

Seapodym DVAR matrices class.

The documentation for this class was generated from the following file:

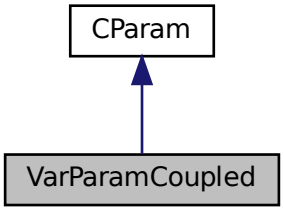
- src/VarMatrices.h

3.19 VarParamCoupled Class Reference

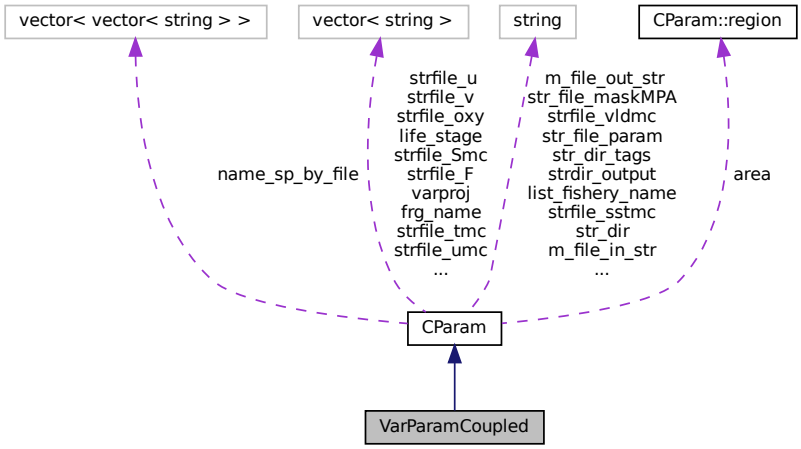
Seapodym DVAR parameter class.

`#include <VarParamCoupled.h>`

Inheritance diagram for VarParamCoupled:



Collaboration diagram for VarParamCoupled:



Public Member Functions

- int **nvarcalc** () const
- bool **gcalc** ()
- void **set_gradcalc** (bool flag)
- bool **scalc** ()
- void **set_scalc** (bool flag)
- void **xinit** (dvector &x, adstring_array &x_names)
- dvariable **reset** (dvar_vector x)
- void **getparam** (void)
- double **get_parval** (int idx)
- dvector **get_parvals** (void)
- void **outp_param** (adstring_array x_names, const int nvars)
- void **get_param_index** (ivector &ix, dmatrix &xy, dmatrix &pars)
- double **par_init_lo** (int ix, double eps)
- double **par_init_up** (int ix, double eps)

- double **par_init_step** (int ix, double delta)
- double **par_init_step_left** (int ix)
- double **par_init_step_right** (int ix)
- void **set_all_false** (string ∗pnames)
- int **set_var_parameters** (ivector phase_par_flags, string ∗pnames)
- bool read (const string &parfile)
- void **re_read_varparam** ()
- void **write** (const char ∗parfile)
- void **save_statistics** (const string dirout, const adstring_array x_names, double likelihood, dvector g, double elapsed_time, int status, int iter, int nvars)

Public Attributes

- double **Mp_mean_max_min**
- double **Mp_mean_max_max**
- dvar_vector **dvarsMp_mean_max**
- double **Mp_mean_exp_min**
- double **Mp_mean_exp_max**
- dvar_vector **dvarsMp_mean_exp**
- double **Ms_mean_max_min**
- double **Ms_mean_max_max**
- dvar_vector **dvarsMs_mean_max**
- double **Ms_mean_slope_min**
- double **Ms_mean_slope_max**
- dvar_vector **dvarsMs_mean_slope**
- double **M_mean_range_min**
- double **M_mean_range_max**
- dvar_vector **dvarsM_mean_range**
- double **a_sst_spawning_min**
- double **a_sst_spawning_max**
- dvar_vector **dvarsA_sst_spawning**
- double **b_sst_spawning_min**
- double **b_sst_spawning_max**
- dvar_vector **dvarsB_sst_spawning**
- double **a_sst_larvae_min**
- double **a_sst_larvae_max**
- dvar_vector **dvarsA_sst_larvae**
- double **b_sst_larvae_min**
- double **b_sst_larvae_max**
- dvar_vector **dvarsB_sst_larvae**
- double **alpha_hsp_prey_min**
- double **alpha_hsp_prey_max**
- dvar_vector **dvarsAlpha_hsp_prey**
- double **alpha_hsp_predator_min**
- double **alpha_hsp_predator_max**
- dvar_vector **dvarsAlpha_hsp_predator**
- double **beta_hsp_predator_min**
- double **beta_hsp_predator_max**
- dvar_vector **dvarsBeta_hsp_predator**
- double **a_sst_habitat_min**
- double **a_sst_habitat_max**
- dvar_vector **dvarsA_sst_habitat**
- double **b_sst_habitat_min**
- double **b_sst_habitat_max**

- dvar_vector **dvarsB_sst_habitat**
- double **T_age_size_slope_min**
- double **T_age_size_slope_max**
- dvar_vector **dvarsT_age_size_slope**
- dvector **thermal_func_delta_min**
- dvector **thermal_func_delta_max**
- dvar_matrix **dvarsThermal_func_delta**
- double **a_oxy_habitat_min**
- double **a_oxy_habitat_max**
- dvar_vector **dvarsA_oxy_habitat**
- double **b_oxy_habitat_min**
- double **b_oxy_habitat_max**
- dvar_vector **dvarsB_oxy_habitat**
- dvector **eF_habitat_min**
- dvector **eF_habitat_max**
- dvar_matrix **dvarsEF_habitat**
- double **hp_cannibalism_min**
- double **hp_cannibalism_max**
- dvar_vector **dvarsHp_cannibalism**
- double **sigma_species_min**
- double **sigma_species_max**
- dvar_vector **dvarsSigma_species**
- double **MSS_species_min**
- double **MSS_species_max**
- dvar_vector **dvarsMSS_species**
- double **MSS_size_slope_min**
- double **MSS_size_slope_max**
- dvar_vector **dvarsMSS_size_slope**
- double **c_diff_fish_min**
- double **c_diff_fish_max**
- dvar_vector **dvarsC_diff_fish**
- double **nb_recruitment_min**
- double **nb_recruitment_max**
- dvar_vector **dvarsNb_recruitment**
- double **a_adults_spawning_min**
- double **a_adults_spawning_max**
- dvar_vector **dvarsA_adults_spawning**
- double **spawning_season_peak_min**
- double **spawning_season_peak_max**
- dvar_vector **dvarsSpawning_season_peak**
- double **spawning_season_start_min**
- double **spawning_season_start_max**
- dvar_vector **dvarsSpawning_season_start**
- dmatrix **q_sp_fishery_min**
- dmatrix **q_sp_fishery_max**
- dvar_matrix **dvarsQ_sp_fishery**
- dmatrix **s_slope_sp_fishery_min**
- dmatrix **s_slope_sp_fishery_max**
- dmatrix **s_asympt_sp_fishery_min**
- dmatrix **s_asympt_sp_fishery_max**
- dvar_matrix **dvarsSslope_sp_fishery**
- dvar_matrix **dvarsSlength_sp_fishery**
- dvar_matrix **dvarsSasympt_sp_fishery**
- dvar_matrix **dvarsLike_param**
- dvar_matrix **dvarsProb_zero**

Additional Inherited Members

3.19.1 Detailed Description

Seapodym DVAR parameter class.

In this class we read the XML parameter file, initialize and reset variable parameters.

3.19.2 Member Function Documentation

3.19.2.1 read()

```
bool VarParamCoupled::read (
            const string & parfile )
```

create vectors of model parameters sp_unit_age_class_jv.allocate(0, nb_species - 1); sp_nb_age_class_jv.↩ allocate(0, nb_species - 1); juv_length.allocate(0, nb_species - 1); juv_weight.allocate(0, nb_species - 1); sp_↩ nb_age_class_ad.allocate(0, nb_species - 1); sp_unit_age_class_ad.allocate(0, nb_species - 1); sp_unit_age_↩ class.allocate(0, nb_species - 1);

The documentation for this class was generated from the following files:

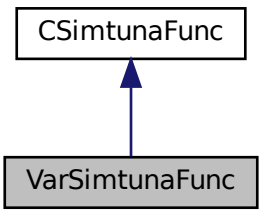
- src/VarParamCoupled.h
- src/VarParamCoupled.cpp
- src/VarParamCoupled_reset.cpp
- src/VarParamCoupled_xinit.cpp

3.20 VarSimtunaFunc Class Reference

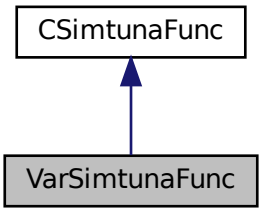
All SEAPODYM functions including DVAR parameters.

```
#include <VarSimtunaFunc.h>
```

Inheritance diagram for VarSimtunaFunc:



Collaboration diagram for VarSimtunaFunc:



Public Member Functions

- void Spawning_Habitat (VarParamCoupled &param, CMatrices &mat, const PMap &map, dvar_matrix &Hs, const double sigma_sp_var, int sp, const int t_count, const int jday)
- void **Hs_comp** (VarParamCoupled &param, CMatrices &mat, const PMap &map, dvar_matrix &Hs, double a, double b, double c, double d, double e, const double sigma_sp_var, const int jday, int t_count)
- double **Hs_comp_elem** (CMatrices &mat, dvector F, const double pp_transform, const double a, const double b, const double c, const double d, const double e, const double sigma_sp_var, const int nb_forage, ivector day_layer, ivector night_layer, const int jday, const int t, const int i, const int j)
- void Juvenile_Habitat (VarParamCoupled &param, CMatrices &mat, const PMap &map, dvar_matrix &Hs, int sp, const int t_count)
- void **Juvenile_Habitat_cannibalism** (VarParamCoupled &param, CMatrices &mat, const PMap &map, dvar_matrix &Hs, dvar_matrix &total_pop, int sp, const int t_count)
- void **Hj_comp** (VarParamCoupled &param, CMatrices &mat, const PMap &map, dvar_matrix &Hj, double a, double b, const int t)
- void **Hj_cannibalism_comp** (VarParamCoupled &param, CMatrices &mat, const PMap &map, dvar_matrix &Hj, const dmatrix &total_pop, double a, double b, double c, const int t)
- void Faccessibility (VarParamCoupled &param, VarMatrices &mat, const PMap &map, const int sp, const int jday, const int t_count, const int pop_built, const int tags_only, const ivector tags_age_solve)
- void **Vars_at_age_precomp** (CParam &param, const int sp)
- double **Topt_at_age_comp** (CParam &param, const double teta_min, const double teta_max, const int sp, const int age)
- void **Faccessibility_comp** (VarParamCoupled &param, VarMatrices &mat, const PMap &map, double teta↩ _max, double oxy_teta, double oxy_cr, const int sp, const int age, const int jday, const int t)
- void **Average_currents** (VarParamCoupled &param, VarMatrices &mat, const PMap &map, int age, const int t_count, const int pop_built)
- void **Average_currents_comp** (VarParamCoupled &param, VarMatrices &mat, const PMap &map, const int age, const int t)
- double **Tmean_comp** (VarParamCoupled &param, VarMatrices &mat, const PMap &map, const int sp, const int age, const int t)
- void **Feeding_Habitat** (VarParamCoupled &param, VarMatrices &mat, const PMap &map, dvar_matrix &Ha, int sp, int age, const int jday, const int t_count, const int migration_flag)
- void **Hf_comp** (VarParamCoupled &param, VarMatrices &mat, const PMap &map, dvar_matrix &Hf, const int sp, const int age, const int jday, const int t)
- void Feeding_Habitat_Index (VarParamCoupled &param, VarMatrices &mat, const PMap &map, dvar_matrix &Ha, int sp, int age, const int jday, const int t_count)
- void **Seasonal_Habitat_Index** (VarParamCoupled &param, VarMatrices &mat, const PMap &map, dvar_↩ matrix &Hs, dvar_matrix &Ha, int sp, int age, const int jday, const int t_count)

- void **Ha_comp** (VarParamCoupled &param, VarMatrices &mat, const PMap &map, const dmatrix Hs, dvar↩ _matrix &Ha, const int sp, const int jday)
- void Seasonal_switch (VarParamCoupled &param, VarMatrices &mat, const PMap &map, const int jday, int sp)
- void **Seasonal_switch_comp** (VarParamCoupled &param, VarMatrices &mat, const PMap &map, double season_peak, double season_start, const int jday, const int sp)
- void **Seasonal_switch_year_precomp** (CParam &param, CMatrices &mat, const PMap &map, double season_peak, double season_start, const int sp)
- void Mortality_Sp (VarParamCoupled &param, CMatrices &mat, const PMap &map, dvar_matrix &M, dvar↩ _matrix &H, int sp, double mean_age_in_dtau, const int age, const int t_count)
- void M_sp_comp (const PMap &map, dvar_matrix &M, const dmatrix &H, double, double, double, double, double, double, const int dtau)
- void **M_PH_juv_comp** (VarParamCoupled &param, const PMap &map, CMatrices &mat, dvar_matrix &M, const dmatrix &PH, double mean_age_in_dtau)
- void **allocate_dvmatr** (const int imin, const int imax, const ivector jinf, const ivector jsup)
- dvariable **adv_diff** (const double H, dvariable &c)
- void **time_reading_init** ()

Public Attributes

- double **elapsed_time_reading**

3.20.1 Detailed Description

All SEAPODYM functions including DVAR parameters.

3.20.2 Member Function Documentation

3.20.2.1 Faccessibility()

```
void VarSimtunaFunc::Faccessibility (
            VarParamCoupled & param,
            VarMatrices & mat,
            const PMap & map,
            const int sp,
            const int jday,
            const int t_count,
            const int pop_built,
            const int tags_only,
            const ivector tags_age_solve )
```

Forward main functions called in simulation mode only for: 1) accessibility to forage components (f_accessibility) or to their respective layers (f_accessibility_layer). 2) average currents given the accessibility to the layer. See accessibility.cpp

3.20.2.2 Feeding_Habitat_Index()

```
void VarSimtunaFunc::Feeding_Habitat_Index (
            VarParamCoupled & param,
            VarMatrices & mat,
            const PMap & map,
            dvar_matrix & Hf,
            int sp,
            int age,
            const int jday,
            const int t_count )
```

Forward main function called in simulation mode only for: feeding habitat for young and adult life stages, with or without seasonal switch between habitats depending on the migration flag. See feeding_habitat.cpp

3.20.2.3 Juvenile_Habitat()

```
void VarSimtunaFunc::Juvenile_Habitat (
            VarParamCoupled & param,
            CMatrices & mat,
            const PMap & map,
            dvar_matrix & Hj,
            int sp,
            const int t_count )
```

Forward main function called in simulation mode only for: juvenile habitat functions. See juvenile_habitat.cpp

3.20.2.4 M_sp_comp()

```
void VarSimtunaFunc::M_sp_comp (
            const PMap & map,
            dvar_matrix & M,
            const dmatrix & H,
            double Mp_max,
            double Ms_max,
            double Mp_exp,
            double Ms_slope,
            double range,
            double mean_age_in_dtau,
            const int dtau )
```

Forward functions for: mortality rates at age. These functions include fixed natural mortality rate and variable component, depending on habitat indices defined for the life stage

3.20.2.5 Mortality_Sp()

```
void VarSimtunaFunc::Mortality_Sp (
            VarParamCoupled & param,
            CMatrices & mat,
            const PMap & map,
            dvar_matrix & M,
            dvar_matrix & H,
            int sp,
            double mean_age_in_dtau,
            const int age,
            const int t_count )
```

Forward main function called in simulation mode only for: mortality rates at age. See mortality_sp.cpp

3.20.2.6 Seasonal_switch()

```
void VarSimtunaFunc::Seasonal_switch (
            VarParamCoupled & param,
            VarMatrices & mat,
            const PMap & map,
            const int jday,
            int sp )
```

Forward main function called in simulation mode only for: computing the seasonal switch function used to switch between habitats. See seasonal_switch.cpp

3.20.2.7 Spawning_Habitat()

```
void VarSimtunaFunc::Spawning_Habitat (
            VarParamCoupled & param,
            CMatrices & mat,
            const PMap & map,
            dvar_matrix & Hs,
            const double sigma_sp_var,
            int sp,
            const int t_count,
            const int jday )
```

Forward main function called in simulation mode only for: spawning habitat functions. See spawning_habitat.cpp

The documentation for this class was generated from the following files:

- src/VarSimtunaFunc.h
- src/accessibility.cpp
- src/dv_accessibility.cpp
- src/dv_feeding_habitat.cpp
- src/dv_juvenile_habitat.cpp
- src/dv_mortality_sp.cpp
- src/dv_seasonal_switch.cpp
- src/dv_spawning_habitat.cpp
- src/fd_accessibility.cpp
- src/fd_feeding_habitat.cpp
- src/fd_juvenile_habitat.cpp
- src/fd_mortality_sp.cpp
- src/fd_seasonal_switch.cpp
- src/fd_spawning_habitat.cpp
- src/feeding_habitat.cpp
- src/juvenile_habitat.cpp
- src/mortality_sp.cpp
- src/seasonal_switch.cpp
- src/spawning_habitat.cpp

Index