Metapopulation Assessment System

Generated by Doxygen 1.8.9.1

Tue Mar 3 2015 10:32:44

Contents

Chapter 1

Metapopulation Assessment System<br>

Jon Brodziak, Matthew Supernaw, and Clay Porch
NOAA, National Marine Fisheries Service

1.1 Introduction

The scale and connectivity of spatial processes are key to understanding and predicting patterns in resource abundance in population and community ecology. The importance of spatial scale and connectivity is a paradigm for ecology (Levin 1992) and provides a strong rationale to focus on metapopulations, or populations of populations, as the basic unit of population ecology. To understand and predict impacts on populations, population assessments should increasingly focus on the influence of spatial variation on resources and the development of metapopulation assessments will require a shift towards more complex and hierarchical population models.

From a software design perspective, an object-oriented design paradigm where one can provide precise design specifications, have a shorter development phase for new code, and expect easier maintenance with consistency and reusability through time is a natural approach to developing more complex and hierarchical population models. To some extent, such features are partially implemented in some existing integrated assessment models, e.g. Stock Synthesis, but there is ample room for improvements in model development, selection, testing, uncertainty quantification, and assessment modeling components. In this context, rapid prototyping and testing of alternative models for a diversity of fishery systems is desirable and could be achieved through the ongoing development of a system library of tested modules and templates. The capacity to build new models from object-oriented templates will foster the ongoing development of structured assessment models and will also help to avoid some of the life cycle issues of maintaining a single omnibus assessment model.

1.2 Purpose

1.3 Overview

1.4 Overview

1.5 Examples

1.6 References

Chapter 2

Namespace Index

2.1 Namespace List

Here is a list of all namespaces with brief descriptions:

Chapter 3

Hierarchical Index

3.1 Class Hierarchy

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

Chapter 4

Class Index

4.1 Class List

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

Chapter 5

File Index

5.1 File List

Here is a list of all files with brief descriptions:

Chapter 6

Namespace Documentation

6.1 mas Namespace Reference

Classes

- class EvaluationObject
- class Information
- class Location
- class Model
- class Object
- class Observation
- struct Point
- class Polygon
- class Rectangle
- class Structure
- class Subpopulation

6.1.1 Detailed Description

Since MAS is using the standard c++ library as much as possible, vector and matrix operations are held here. Matrices are represented in std::vector containers with dimensions folded.

Chapter 7

Class Documentation

7.1 mas::EvaluationObject< REAL_T, EVAL_T > Class Template Reference

`#include <Object.hpp>`

Inherits mas::Object.

Inherited by mas::Location< REAL_T, EVAL_T >, mas::Model< REAL_T, EVAL_T >, mas::Structure< REAL_T, EVAL_T >, and mas::Subpopulation< REAL_T, EVAL_T >.

Protected Member Functions

- Information< REAL_T, EVAL_T > * GetInfo () const
- void SetInfo (Information< REAL_T, EVAL_T > *info)

Protected Attributes

- Information< REAL_T, EVAL_T > * info

Additional Inherited Members

7.1.1 Member Function Documentation

7.1.1.1 template<typename REAL_T , typename EVAL_T > Information<REAL_T, EVAL_T>* mas::EvaluationObject< REAL_T, EVAL_T >::GetInfo ( ) const `[inline], [protected]`

7.1.1.2 template<typename REAL_T , typename EVAL_T > void mas::EvaluationObject< REAL_T, EVAL_T >::SetInfo (Information< REAL_T, EVAL_T > * *info*) `[inline], [protected]`

7.1.2 Member Data Documentation

7.1.2.1 template<typename REAL_T , typename EVAL_T > Information<REAL_T,EVAL_T>* mas::EvaluationObject< REAL_T, EVAL_T >::info `[protected]`

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

- Object.hpp

7.2 mas::Information< REAL_T, EVAL_T > Class Template Reference

```
#include <Information.hpp>
```

Public Member Functions

- void RegisterEstimable (EVAL_T *parameter, int phase=0)

7.2.1 Detailed Description

template<typename REAL_T, typename EVAL_T>class mas::Information< REAL_T, EVAL_T >

Class to hold information about a models configuration.

7.2.2 Member Function Documentation

7.2.2.1 template<typename REAL_T, typename EVAL_T> void mas::Information< REAL_T, EVAL_T >::RegisterEstimable (EVAL_T * *parameter*, int *phase* = 0) [inline]

Register a model as estimable. Simply adds a pointer to the parameter to a list along with its phase. First phase is 0.

Parameters

parameter	
phase	

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

- Information.hpp

7.3 mas::Location< REAL_T, EVAL_T > Class Template Reference

```
#include <Location.hpp>
```

Inherits mas::EvaluationObject< REAL_T, EVAL_T >.

Additional Inherited Members

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

- Location.hpp

7.4 mas::Model< REAL_T, EVAL_T > Class Template Reference

```
#include <Model.hpp>
```

Inherits mas::EvaluationObject< REAL_T, EVAL_T >.

Additional Inherited Members

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

- Model.hpp

7.5 mas::Object Class Reference

```
#include <Object.hpp>
```

Inherited by mas::EvaluationObject< REAL_T, EVAL_T >, mas::Observation< REAL_T >, and mas::↩
Observation< EVAL_T, REAL_T >.

Public Member Functions

- virtual std::string ToString () const

Friends

- std::ostream & operator<< (std::ostream &os, const Object &obj)

7.5.1 Member Function Documentation

7.5.1.1 virtual std::string mas::Object::ToString ( ) const [inline], [virtual]

7.5.2 Friends And Related Function Documentation

7.5.2.1 std::ostream& operator<< (std::ostream & *os*, const Object & *obj*) [friend]

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

- Object.hpp

7.6 mas::Observation< REAL_T > Class Template Reference

```
#include <Observation.hpp>
```

Inherits mas::Object.

Additional Inherited Members

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

- Observation.hpp

7.7 mas::Point< REAL_T > Struct Template Reference

```
#include <Location.hpp>
```

Public Member Functions

- Point ()
- Point (const REAL_T &x, const REAL_T &y)

Public Attributes

- REAL_T x
- REAL_T y

7.7.1 Constructor & Destructor Documentation

7.7.1.1 template<typename REAL_T> mas::Point< REAL_T >::Point ( ) [inline]

7.7.1.2 template<typename REAL_T> mas::Point< REAL_T >::Point (const REAL_T & *x*, const REAL_T & *y*) [inline]

7.7.2 Member Data Documentation

7.7.2.1 template<typename REAL_T> REAL_T mas::Point< REAL_T >::x

7.7.2.2 template<typename REAL_T> REAL_T mas::Point< REAL_T >::y

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

- Location.hpp

7.8 mas::Polygon< REAL_T > Class Template Reference

```
#include <Location.hpp>
```

Public Member Functions

- void AddPoint (const Point< REAL_T > &p)
- REAL_T ComputeArea ()
- Point< REAL_T > Centroid ()

7.8.1 Member Function Documentation

7.8.1.1 template<typename REAL_T> void mas::Polygon< REAL_T >::AddPoint (const Point< REAL_T > & *p*) [inline]

7.8.1.2 template<typename REAL_T> Point<REAL_T> mas::Polygon< REAL_T >::Centroid ( ) [inline]

7.8.1.3 template<typename REAL_T> REAL_T mas::Polygon< REAL_T >::ComputeArea ( ) [inline]

Compute the area of this polygon. points must be entered in clockwise order.

Returns

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

- Location.hpp

7.9 mas::Rectangle< REAL_T > Class Template Reference

```
#include <Location.hpp>
```

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

- Location.hpp

7.10 mas::Structure< REAL_T, EVAL_T > Class Template Reference

```
#include <Structure.hpp>
```

Inherits mas::EvaluationObject< REAL_T, EVAL_T >.

Public Types

- enum StructureType { AGE = 0, LENGTH, STAGE }

Public Member Functions

- Structure (StructureType type=AGE)
- operator StructureType ()
- operator StructureType () const

Public Attributes

- StructureType structure_type

Additional Inherited Members

7.10.1 Member Enumeration Documentation

7.10.1.1 template<typename REAL_T , typename EVAL_T = REAL_T> enum mas::Structure::StructureType

Enumerator

AGE

LENGTH

STAGE

7.10.2 Constructor & Destructor Documentation

7.10.2.1 template<typename REAL_T , typename EVAL_T = REAL_T> mas::Structure< REAL_T, EVAL_T >::Structure (StructureType *type* = AGE) [inline]

Default constructor.

The default structure type is age.

7.10.3 Member Function Documentation

7.10.3.1 template<typename REAL_T , typename EVAL_T = REAL_T> mas::Structure< REAL_T, EVAL_T >::operator StructureType ( ) [inline]

7.10.3.2 template<typename REAL_T , typename EVAL_T = REAL_T> mas::Structure< REAL_T, EVAL_T >::operator StructureType ( ) const [inline]

7.10.4 Member Data Documentation

7.10.4.1 template<typename REAL_T , typename EVAL_T = REAL_T> StructureType mas::Structure< REAL_T, EVAL_T >::structure_type

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

- Structure.hpp

7.11 mas::Subpopulation< REAL_T, EVAL_T > Class Template Reference

```
#include <Subpopulation.hpp>
```

Inherits mas::EvaluationObject< REAL_T, EVAL_T >.

Public Member Functions

- const std::vector< EVAL_T > & GetBiomass () const
- void SetBiomass (std::vector< EVAL_T > B)
- const std::vector< EVAL_T > & GetDeaths () const
- void SetDeaths (std::vector< EVAL_T > D)
- const std::vector< EVAL_T > & GetGrowth () const
- void SetGrowth (std::vector< EVAL_T > G)
- const std::vector< EVAL_T > & GetImmigration () const
- void SetImmigration (std::vector< EVAL_T > I)
- const std::vector< EVAL_T > & GetEmigration () const
- void SetEmigration (std::vector< EVAL_T > O)
- const std::vector< EVAL_T > & GetRecruitment () const
- void SetRecruitment (std::vector< EVAL_T > R)
- const std::vector< EVAL_T > & GetYield () const
- void SetYield (std::vector< EVAL_T > Y)
- uint32_t GetAgeMax () const
- void SetAgeMax (uint32_t age_max)
- uint32_t GetGenders () const
- void SetGenders (uint32_t genders)
- Observation< REAL_T > ∗ GetObservations () const
- void SetObservations (Observation< REAL_T > ∗observations)
- uint32_t GetTimeMax () const
- void SetTimeMax (uint32_t time_max)

Additional Inherited Members

7.11.1 Detailed Description

template<typename REAL_T, typename EVAL_T = REAL_T>class mas::Subpopulation< REAL_T, EVAL_T >

The subpopulation class follows the well-established premise of conservation of biomass (Russell 1931) within each area of the model. The general model of the time change, indexed by t , in the biomass (a,t B) of the harvested subpopulation in each area, indexed by a , depends on biomass increases due to somatic growth (G(a,t)), recruitment (R(a,t)), and immigration (I(a,t)) and depends on biomass decreases due to natural deaths (D(a,t)), fishery yields (Y(a,t)), and emigration (O(a,t)) via

B(a,t+1) = B(a,t) + [G(a,t) + R(a,t) + I(a,t)] - [D(a,t) + Y(a,t) + O(a,t)]

7.11.2 Member Function Documentation

7.11.2.1 template<typename REAL_T , typename EVAL_T = REAL_T> uint32_t mas::Subpopulation< REAL_T, EVAL_T >::GetAgeMax ( ) const [inline]

7.11.2.2 template<typename REAL_T , typename EVAL_T = REAL_T> const std::vector<EVAL_T>& mas::Subpopulation< REAL_T, EVAL_T >::GetBiomass ( ) const [inline]

7.11.2.3 template<typename REAL_T , typename EVAL_T = REAL_T> const std::vector<EVAL_T>& mas::Subpopulation< REAL_T, EVAL_T >::GetDeaths ( ) const [inline]

7.11.2.4 template<typename REAL_T , typename EVAL_T = REAL_T> const std::vector<EVAL_T>& mas::Subpopulation< REAL_T, EVAL_T >::GetEmigration ( ) const [inline]

7.11.2.5 template<typename REAL_T , typename EVAL_T = REAL_T> uint32_t mas::Subpopulation< REAL_T, EVAL_T >::GetGenders ( ) const [inline]

7.11.2.6 template<typename REAL_T , typename EVAL_T = REAL_T> const std::vector<EVAL_T>& mas::Subpopulation< REAL_T, EVAL_T >::GetGrowth ( ) const [inline]

7.11.2.7 template<typename REAL_T , typename EVAL_T = REAL_T> const std::vector<EVAL_T>& mas::Subpopulation< REAL_T, EVAL_T >::GetImmigration ( ) const [inline]

7.11.2.8 template<typename REAL_T , typename EVAL_T = REAL_T> Observation<REAL_T>* mas::Subpopulation< REAL_T, EVAL_T >::GetObservations ( ) const [inline]

7.11.2.9 template<typename REAL_T , typename EVAL_T = REAL_T> const std::vector<EVAL_T>& mas::Subpopulation< REAL_T, EVAL_T >::GetRecruitment ( ) const [inline]

7.11.2.10 template<typename REAL_T , typename EVAL_T = REAL_T> uint32_t mas::Subpopulation< REAL_T, EVAL_T >::GetTimeMax ( ) const [inline]

7.11.2.11 template<typename REAL_T , typename EVAL_T = REAL_T> const std::vector<EVAL_T>& mas::Subpopulation< REAL_T, EVAL_T >::GetYield ( ) const [inline]

7.11.2.12 template<typename REAL_T , typename EVAL_T = REAL_T> void mas::Subpopulation< REAL_T, EVAL_T >::SetAgeMax (uint32_t *age_max*) [inline]

7.11.2.13 template<typename REAL_T , typename EVAL_T = REAL_T> void mas::Subpopulation< REAL_T, EVAL_T >::SetBiomass (std::vector< EVAL_T > *B*) [inline]

7.11.2.14 template<typename REAL_T , typename EVAL_T = REAL_T> void mas::Subpopulation< REAL_T, EVAL_T >::SetDeaths (std::vector< EVAL_T > *D*) [inline]

7.11.2.15 template<typename REAL_T , typename EVAL_T = REAL_T> void mas::Subpopulation< REAL_T, EVAL_T >::SetEmigration (std::vector< EVAL_T > *O*) [inline]

7.11.2.16 template<typename REAL_T , typename EVAL_T = REAL_T> void mas::Subpopulation< REAL_T, EVAL_T >::SetGenders (uint32_t *genders*) [inline]

7.11.2.17 template<typename REAL_T , typename EVAL_T = REAL_T> void mas::Subpopulation< REAL_T, EVAL_T >::SetGrowth (std::vector< EVAL_T > *G*) [inline]

7.11.2.18 template<typename REAL_T , typename EVAL_T = REAL_T> void mas::Subpopulation< REAL_T, EVAL_T >::SetImmigration (std::vector< EVAL_T > *I*) [inline]

7.11.2.19 template<typename REAL_T , typename EVAL_T = REAL_T> void mas::Subpopulation< REAL_T, EVAL_T >::SetObservations (Observation< REAL_T > * *observations*) [inline]

7.11.2.20 template<typename REAL_T , typename EVAL_T = REAL_T> void mas::Subpopulation< REAL_T, EVAL_T >::SetRecruitment (std::vector< EVAL_T > *R*) [inline]

7.11.2.21 template<typename REAL_T , typename EVAL_T = REAL_T> void mas::Subpopulation< REAL_T, EVAL_T >::SetTimeMax (uint32_t *time_max*) [inline]

7.11.2.22 template<typename REAL_T , typename EVAL_T = REAL_T> void mas::Subpopulation< REAL_T, EVAL_T >::SetYield (std::vector< EVAL_T > *Y*) [inline]

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

- Subpopulation.hpp

Chapter 8

File Documentation

8.1 Information.hpp File Reference

```
#include "Object.hpp"
```

Classes

- class mas::Information< REAL_T, EVAL_T >

Namespaces

- mas

8.2 Location.hpp File Reference

```
#include <vector>
#include "Object.hpp"
#include "Subpopulation.hpp"
```

Classes

- struct mas::Point< REAL_T >
- class mas::Rectangle< REAL_T >
- class mas::Polygon< REAL_T >
- class mas::Location< REAL_T, EVAL_T >

Namespaces

- mas

8.3 MAS.hpp File Reference

```
#include <vector>
#include "support/CSTAR/CSTAR.hpp"
#include "support/CSTAR/Population.hpp"
```

Namespaces

- mas

8.4 MathUtilites.hpp File Reference

Namespaces

- mas

8.5 Model.hpp File Reference

```
#include "Object.hpp"
#include "Subpopulation.hpp"
```

Classes

- class mas::Model< REAL_T, EVAL_T >

Namespaces

- mas

8.6 Object.hpp File Reference

```
#include <string>
#include "Information.hpp"
```

Classes

- class mas::Object
- class mas::EvaluationObject< REAL_T, EVAL_T >

Namespaces

- mas

8.7 Observation.hpp File Reference

```
#include "Object.hpp"
```

Classes

- class mas::Observation< REAL_T >

Namespaces

- mas

8.8 Structure.hpp File Reference

```
#include "Object.hpp"
```

Classes

- class mas::Structure< REAL_T, EVAL_T >

Namespaces

- mas

8.9 Subpopulation.hpp File Reference

```
#include <vector>
#include "Object.hpp"
#include "Observation.hpp"
```

Classes

- class mas::Subpopulation< REAL_T, EVAL_T >

Namespaces

- mas

Index