IGoR

Generated by Doxygen 1.8.13

Contents

Chapter 1

Deprecated List

Member GenModel::generate_sequences (int, bool)

This function used to store generated sequences in memory, and quickly overloaded it for large number of generated sequences.

Chapter 2

Bug List

Member Dinucl_markov::add_to_marginals (long double, Marginal_array_p &) const

Will only count realizations of unambiguous nucleotides (realization indices>=0 since they are set to -1 in iterate_common)

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

Class Documentation

5.1 Adjacency_list Struct Reference

IGoR's Bayesian Network adjacency list.

```
#include <Model_Parms.h>
```

Public Attributes

- std::list< std::shared_ptr< Rec_Event > > **children**
- std::list< std::shared_ptr< Rec_Event > > **parents**

5.1.1 Detailed Description

IGoR's Bayesian Network adjacency list.

Author

Q.Marcou

Version

1.0

Contains a list of smart pointers pointing to an event parents and children (i.e adjacent nodes)

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

- Model_Parms.h

5.2 Aligner Class Reference

A modified Smith-Waterman alignment class.

```
#include <Aligner.h>
```

Public Member Functions

- Aligner (Matrix< double >, int, Gene_class)
- std::forward_list< Alignment_data > **align_seq** (std::string, double, bool, int, int, bool=false)
- std::forward_list< Alignment_data > **align_seq** (std::string, double, bool, int, int, std::set< std::string >, bool=false)
- std::forward_list< Alignment_data > **align_seq** (std::string, double, bool, bool, int, int, bool=false)
- std::forward_list< Alignment_data > **align_seq** (std::string, double, bool, bool, int, int, std::set< std::string >, bool=false)
- std::forward_list< Alignment_data > **align_seq** (std::string, double, bool, std::unordered_map< std::string, std::pair< int, int >>, bool=false)
- std::forward_list< Alignment_data > **align_seq** (std::string, double, bool, std::unordered_map< std::string, std::pair< int, int >>, std::set< std::string >, bool=false)
- std::forward_list< Alignment_data > **align_seq** (std::string, double, bool, bool, std::unordered_map< std↩ ::string, std::pair< int, int >>, bool=false)
- std::forward_list< Alignment_data > **align_seq** (std::string, double, bool, bool, std::unordered_map< std↩ ::string, std::pair< int, int >>, std::set< std::string >, bool=false)
- std::unordered_map< int, std::forward_list< Alignment_data > > **align_seqs** (std::vector< std::pair< const int, const std::string >>, double, bool)
- std::unordered_map< int, std::forward_list< Alignment_data > > **align_seqs** (std::vector< std::pair< const int, const std::string >>, double, bool, bool)
- std::unordered_map< int, std::forward_list< Alignment_data > > **align_seqs** (std::vector< std::pair< const int, const std::string >>, double, bool, int, int, bool=false)
- std::unordered_map< int, std::forward_list< Alignment_data > > **align_seqs** (std::vector< std::pair< const int, const std::string >>, double, bool, bool, int, int, bool=false)
- std::unordered_map< int, std::forward_list< Alignment_data > > **align_seqs** (std::vector< std::pair< const int, const std::string >>, double, bool, std::unordered_map< std::string, std::pair< int, int >>, bool=false)
- std::unordered_map< int, std::forward_list< Alignment_data > > **align_seqs** (std::vector< std::pair< const int, const std::string >>, double, bool, bool, std::unordered_map< std::string, std::pair< int, int >>, bool=false)
- void **align_seqs** (std::string, std::vector< std::pair< const int, const std::string >>, double, bool)
- void **align_seqs** (std::string, std::vector< std::pair< const int, const std::string >>, double, bool, bool)
- void **align_seqs** (std::string, std::vector< std::pair< const int, const std::string >>, double, bool, int, int, bool=false)
- void **align_seqs** (std::string, std::vector< std::pair< const int, const std::string >>, double, bool, bool, int, int, bool=false)
- void **align_seqs** (std::string, std::vector< std::pair< const int, const std::string >>, double, bool, std↩ ::unordered_map< std::string, std::pair< int, int >>, bool=false)
- void **align_seqs** (std::string, std::vector< std::pair< const int, const std::string >>, double, bool, bool, std↩ ::unordered_map< std::string, std::pair< int, int >>, bool=false)
- void **write_alignments_seq_csv** (std::string, std::unordered_map< int, std::forward_list< Alignment_data >>)
- std::unordered_map< int, std::forward_list< Alignment_data > > **read_alignments_seq_csv** (std::string, double, bool)
- void **set_genomic_sequences** (std::vector< std::pair< std::string, std::string > >)
- int **incorporate_in_dels** (std::string &, std::string &, const std::forward_list< int >, const std::forward_list< int >, int)

5.2.1 Detailed Description

A modified Smith-Waterman alignment class.

Author

Q.Marcou

Version

1.0

The Aligner class allows to perform SW alignments according to the parameters (substitution matrix,gap penalty) supplied upon construction of the object. The SW alignments matrix has been altered for V and J in order to allow for deletions on the deleted side only. Alignments can be made in parallel using openMP

5.2.2 Constructor & Destructor Documentation

5.2.2.1 Aligner()

```
Aligner::Aligner (
            Matrix< double > sub_mat,
            int gap_pen,
            Gene_class gene )
```

Constructor for the Aligner class : substitution matrix : sets the gap penalty (the gap penalty is linear) : Gene class of the gene aligned. V gene allows for deletions on the 3' side of the genomic template, J gene on the 5' , D gene and undefined allow deletion on both sides

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

- Aligner.h
- Aligner.cpp

5.3 Alignment_data Class Reference

Stores information on the alignment of one genomic template against the target.

```
#include <Aligner.h>
```

Public Member Functions

- **Alignment_data** (std::string gene, int off)
- **Alignment_data** (int off, size_t five_p_off, size_t three_p_off, size_t align_len, std::forward_list< int > ins, std::forward_list< int > del, std::vector< int > mis, double alignment_score)
- **Alignment_data** (std::string gene, int off, size_t align_len, std::forward_list< int > ins, std::forward_list< int > del, std::vector< int > mis, double alignment_score)
- **Alignment_data** (std::string gene, int off, size_t five_p_off, size_t three_p_off, size_t align_len, std::forward_list< int > ins, std::forward_list< int > del, std::vector< int > mis, double alignment_score)

Public Attributes

- std::string **gene_name**
- int **offset**
- size_t **five_p_offset**
- size_t **three_p_offset**
- std::forward_list< int > **insertions**
- std::forward_list< int > **deletions**
- size_t **align_length**
- std::vector< int > **mismatches**
- double **score**

5.3.1 Detailed Description

Stores information on the alignment of one genomic template against the target.

Author

Q.Marcou

Version

1.0

Stores information on the alignment of one genomic template against the target. It contains:

- the gene name
- the offset of the alignment (index on the target sequence on which the first letter of the FULL genomic template aligns (can be negative or lie outside the target)
- 5' and 3' offset positions of the best alignment first and last aligned nucleotide
- insertions : indices on the TARGET of inserted nucleotides
- deletions : indices on the GENOMIC TEMPLATE of deleted nucleotides
- alignment length
- list of mismatches (that lie event outside the best alignment to allow IGoR to know mismatch positions in advance while exploring different deletions numbers)
- the alignment score

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

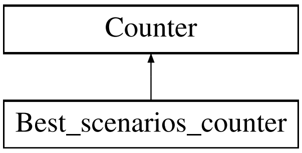
- Aligner.h

5.4 Best_scenarios_counter Class Reference

Records the N best scenarios realizations and mismatches.

```
#include <Bestscenarioscounter.h>
```

Inheritance diagram for Best_scenarios_counter:



Public Member Functions

- **Best_scenarios_counter** (size_t)
- **Best_scenarios_counter** (size_t, bool)
- **Best_scenarios_counter** (size_t, std::string)
- **Best_scenarios_counter** (size_t, std::string, bool)
- std::string **type** () const
- void **initialize_counter** (const Model_Parms &, const Model_marginals &)
- void **count_scenario** (long double, double, const std::string &, Seq_type_str_p_map &, const Seq_offsets↩ _map &, const std::unordered_map< std::tuple< Event_type, Gene_class, Seq_side >, std::shared_ptr< Rec_Event >> &, Mismatch_vectors_map &)
- void **count_sequence** (double, const Model_marginals &, const Model_Parms &)
- void **add_checked** (std::shared_ptr< Counter >)
- void **dump_sequence_data** (int, int)
- std::shared_ptr< Counter > **copy** () const

Public Attributes

- size_t **n_scenarios_counted**
- std::shared_ptr< std::ofstream > **output_scenario_file_ptr**
- std::queue< std::vector< int > > **single_scenario_realizations_queue**
- std::list< int > **single_scenario_mismatches_list**
- std::vector< std::tuple< double, std::queue< std::vector< int > >, std::list< int > > > **best_scenarios↩ _vec**
- std::forward_list< std::shared_ptr< const Rec_Event > > **event_fw_list**

Additional Inherited Members

5.4.1 Detailed Description

Records the N best scenarios realizations and mismatches.

Author

Q.Marcou

Version

1.0

Implementation of the Counter abstract class. Records the N most likely scenario realizations and mismatches and append it to a semicolon separated file.

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

- Bestscenarioscounter.h
- Bestscenarioscounter.cpp

5.5 CDR3SeqData Class Reference

Class to store CDR3 information of a sequence.

```
#include <CDR3SeqData.h>
```

Public Member Functions

- **CDR3SeqData** (const CDR3SeqData &orig)
- std::string **strData** ()

Public Attributes

- int **seq_index**
- int **v_anchor**
- int **j_anchor**
- std::string **CDR3nt**
- std::string **CDR3aa**

5.5.1 Detailed Description

Class to store CDR3 information of a sequence.

Author

C. Olivares

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

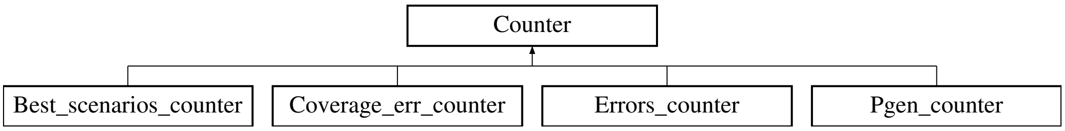
- CDR3SeqData.h
- CDR3SeqData.cpp

5.6 Counter Class Reference

Scenario statistics recording abstract class.

```
#include <Counter.h>
```

Inheritance diagram for Counter:



Public Member Functions

- **Counter** (const std::string &path="/tmp/", bool last_iter=false)
- virtual std::string **type** () const =0
- virtual void **initialize_counter** (const Model_Parms &, const Model_marginals &)=0
- virtual void **count_scenario** (long double, double, const std::string &, Seq_type_str_p_map &, const Seq_offsets_map &, const std::unordered_map< std::tuple< Event_type, Gene_class, Seq_side >, std::shared_ptr< Rec_Event >> &, Mismatch_vectors_map &)
- virtual void **count_sequence** (double, const Model_marginals &, const Model_Parms &)
- virtual void **add_to_counter** (std::shared_ptr< Counter >)
- virtual void **add_checked** (std::shared_ptr< Counter >)=0
- virtual void **dump_sequence_data** (int, int)
- virtual void **dump_data_summary** (int)
- bool **is_last_iter_only** () const
- std::string **get_path_to_files** () const
- void **set_path_to_files** (const std::string &new_path)
- virtual std::shared_ptr< Counter > **copy** () const =0

Protected Attributes

- std::string **path_to_file**
- bool **last_iter_only**
- bool **fstreams_created**

5.6.1 Detailed Description

Scenario statistics recording abstract class.

Author

Q.Marcou

Version

1.0

The Counter abstract class provides an interface to collect individual scenarios statistics and aggregate them in various ways.

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

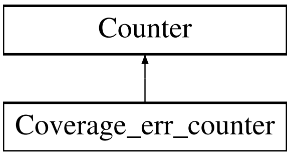
- Counter.h
- Counter.cpp

5.7 Coverage_err_counter Class Reference

Records the number of time each genomic position is observed with or without error.

```
#include <Coverageerrcounter.h>
```

Inheritance diagram for Coverage_err_counter:



Public Member Functions

- **Coverage_err_counter** (Gene_class)
- **Coverage_err_counter** (Gene_class, bool, bool)
- **Coverage_err_counter** (std::string, Gene_class, bool)
- **Coverage_err_counter** (std::string, Gene_class, size_t, bool, bool)
- std::string **type** () const
- void **initialize_counter** (const Model_Parms &, const Model_marginals &)
- void **count_scenario** (long double, double, const std::string &, Seq_type_str_p_map &, const Seq_offsets_map &, const std::unordered_map< std::tuple< Event_type, Gene_class, Seq_side >, std::shared_ptr< Rec_Event >> &, Mismatch_vectors_map &)
- void **count_sequence** (double, const Model_marginals &, const Model_Parms &)
- void **add_checked** (std::shared_ptr< Counter >)
- void **dump_sequence_data** (int, int)
- void **dump_data_summary** (int)
- std::shared_ptr< Counter > **copy** () const

Additional Inherited Members

5.7.1 Detailed Description

Records the number of time each genomic position is observed with or without error.

Author

Q.Marcou

Version

1.0

The Coverage_err_counter allows to record the number of times each genomic site is observed (coverage) and how many times a mismatch has been observed on it (error/mutation). The recording can be made at the single position level, as well as joint over positions duet,triplet etc (e.g the number of times two nucleotides were observed in the same scenario)

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

- Coverageerrcounter.h
- Coverageerrcounter.cpp

5.8 D_position_comparator Struct Reference

Public Member Functions

- bool **operator()** (std::tuple< std::string, int, int, double > position_1, std::tuple< std::string, int, int, double > position_2)

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

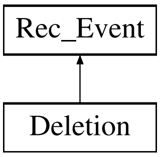
- Utils.h

5.9 Deletion Class Reference

Deletion recombination event.

`#include <Deletion.h>`

Inheritance diagram for Deletion:



Public Member Functions

- **Deletion** (Gene_class, Seq_side, std::pair< int, int >)
- **Deletion** (std::forward_list< int >)
- **Deletion** (Gene_class, Seq_side)
- **Deletion** (Gene_class, Seq_side, std::unordered_map< std::string, Event_realization > &)
- std::shared_ptr< Rec_Event > **copy** ()
- void iterate (double &, Downstream_scenario_proba_bound_map &, const std::string &, const Int_Str &, Index_map &, const std::unordered_map< Rec_Event_name, std::vector< std::pair< std::shared_ptr< const Rec_Event >, int >>> &, std::shared_ptr< Next_event_ptr > &, Marginal_array_p &, const Marginal_array_p &, const std::unordered_map< Gene_class, std::vector< Alignment_data >> &, Seq_type_str_p_map &, Seq_offsets_map &, std::shared_ptr< Error_rate > &, std::map< size_t, std::shared_ptr< Counter >> &, const std::unordered_map< std::tuple< Event_type, Gene_class, Seq_side >, std::shared_ptr< Rec_Event >> &, Safety_bool_map &, Mismatch_vectors_map &, double &, double &)
- void **add_realization** (int)
- std::queue< int > **draw_random_realization** (const Marginal_array_p &, std::unordered_map< Rec_Event_name, int > &, const std::unordered_map< Rec_Event_name, std::vector< std::pair< std::shared_ptr< const Rec_Event >, int >>> &, std::unordered_map< Seq_type, std::string > &, std::mt19937_64 &) const
- void **write2txt** (std::ofstream &)
- void **initialize_event** (std::unordered_set< Rec_Event_name > &, const std::unordered_map< std::tuple< Event_type, Gene_class, Seq_side >, std::shared_ptr< Rec_Event >> &, const std::unordered_map< Rec_Event_name, std::vector< std::pair< std::shared_ptr< const Rec_Event >, int >>> &, Downstream_scenario_proba_bound_map &, Seq_type_str_p_map &, Safety_bool_map &, std::shared_ptr< Error_rate >, Mismatch_vectors_map &, Seq_offsets_map &, Index_map &)
- void **add_to_marginals** (long double, Marginal_array_p &) const
- bool **has_effect_on** (Seq_type) const
- void **iterate_initialize_Len_proba** (Seq_type considered_junction, std::map< int, double > &length_best_proba_map, std::queue< std::shared_ptr< Rec_Event >> &model_queue, double &scenario_proba, const Marginal_array_p &model_parameters_point, Index_map &base_index_map, Seq_type_str_p_map &constructed_sequences, int &seq_len) const
- void **initialize_Len_proba_bound** (std::queue< std::shared_ptr< Rec_Event >> &model_queue, const Marginal_array_p &model_parameters_point, Index_map &base_index_map)

Friends

- class **Coverage_err_counter**
- class **Hypermutation_global_errorrate**
- class **Hypermutation_full_Nmer_errorrate**

Additional Inherited Members

5.9.1 Detailed Description

Deletion recombination event.

Author

Q.Marcou

Version

1.0

The Deletion RecEvent models deletions of one genomic fragment on a given side. Deletions can be either positive or negative (= Palindromic insertions)

By construction the corresponding GeneChoice must have been explored first.

5.9.2 Member Function Documentation

5.9.2.1 iterate()

```
void Deletion::iterate (
             double & ,
             Downstream_scenario_proba_bound_map & ,
             const std::string & ,
             const Int_Str & ,
             Index_map & ,
             const std::unordered_map< Rec_Event_name, std::vector< std::pair< std::shared_↩
ptr< const Rec_Event >, int >>> & ,
             std::shared_ptr< Next_event_ptr > & ,
             Marginal_array_p & ,
             const Marginal_array_p & ,
             const std::unordered_map< Gene_class, std::vector< Alignment_data >> & ,
             Seq_type_str_p_map & ,
             Seq_offsets_map & ,
             std::shared_ptr< Error_rate > & ,
             std::map< size_t, std::shared_ptr< Counter >> & ,
             const std::unordered_map< std::tuple< Event_type, Gene_class, Seq_side >, std↩
::shared_ptr< Rec_Event >> & ,
             Safety_bool_map & ,
             Mismatch_vectors_map & ,
             double & ,
             double & )  [inline], [virtual]
```

General: Loop over all possible number of deletions for a given gene on a given sequence side

Specific: -First check whether any of these number of deletions is possible given the current position and number of deletions on other genes -Loop over # of deletions in decreasing order THIS IS A TEMPORARY FIX// //FIXME

THIS IS A TEMPORARY FIX// //FIXME

THIS IS A TEMPORARY FIX// //FIXME

Implements Rec_Event.

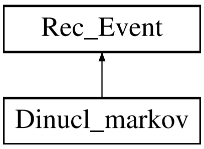
The documentation for this class was generated from the following files:

- Deletion.h
- Deletion.cpp

5.10 Dinucl_markov Class Reference

Dinucleotide insertion Markov model.

Inheritance diagram for Dinucl_markov:



Public Member Functions

- **Dinucl_markov** (Gene_class)
- std::shared_ptr< Rec_Event > **copy** ()
- int **size** () const
- void iterate (double &, Downstream_scenario_proba_bound_map &, const std::string &, const Int_Str &, Index_map &, const std::unordered_map< Rec_Event_name, std::vector< std::pair< std::shared_ptr< const Rec_Event >, int >>> &, std::shared_ptr< Next_event_ptr > &, Marginal_array_p &, const Marginal_array_p &, const std::unordered_map< Gene_class, std::vector< Alignment_data >> &, Seq_type_str_p_map &, Seq_offsets_map &, std::shared_ptr< Error_rate > &, std::map< size_t, std::shared_ptr< Counter >> &, const std::unordered_map< std::tuple< Event_type, Gene_class, Seq_side >, std::shared_ptr< Rec_Event >> &, Safety_bool_map &, Mismatch_vectors_map &, double &, double &)

  Evaluate all Event_Realization of a RecEvent for a given sequence.
- void **add_realization** (int)
- std::queue< int > **draw_random_realization** (const Marginal_array_p &, std::unordered_map< Rec_Event_name, int > &, const std::unordered_map< Rec_Event_name, std::vector< std::pair< std::shared_ptr< const Rec_Event >, int >>> &, std::unordered_map< Seq_type, std::string > &, std::mt19937_64 &) const
- void **write2txt** (std::ofstream &)
- void **ind_normalize** (Marginal_array_p &, size_t) const

- void **initialize_event** (std::unordered_set< Rec_Event_name > &, const std::unordered_map< std::tuple< Event_type, Gene_class, Seq_side >, std::shared_ptr< Rec_Event >> &, const std::unordered_map< Rec_Event_name, std::vector< std::pair< std::shared_ptr< const Rec_Event >, int >>> &, Downstream_scenario_proba_bound_map &, Seq_type_str_p_map &, Safety_bool_map &, std::shared_ptr< Error_rate >, Mismatch_vectors_map &, Seq_offsets_map &, Index_map &)
- void add_to_marginals (long double, Marginal_array_p &) const
- void update_event_internal_probas (const Marginal_array_p &, const std::unordered_map< Rec_Event_name, int > &)
- double ∗ **get_updated_ptr** ()
- void **initialize_crude_scenario_proba_bound** (double &, std::forward_list< double ∗> &, const std::unordered_map< std::tuple< Event_type, Gene_class, Seq_side >, std::shared_ptr< Rec_Event >> &)
- bool **has_effect_on** (Seq_type) const
- void **iterate_initialize_Len_proba** (Seq_type considered_junction, std::map< int, double > &length_best_proba_map, std::queue< std::shared_ptr< Rec_Event >> &model_queue, double &scenario_proba, const Marginal_array_p &model_parameters_point, Index_map &base_index_map, Seq_type_str_p_map &constructed_sequences, int &seq_len) const
- void **initialize_Len_proba_bound** (std::queue< std::shared_ptr< Rec_Event >> &model_queue, const Marginal_array_p &model_parameters_point, Index_map &base_index_map)

Additional Inherited Members

5.10.1 Detailed Description

Dinucleotide insertion Markov model.

Author

Q.Marcou

Version

1.0

Models a Markov chain dictating the identity of inserted nucleotides in the inserted region. We assume a low error frequency and almost flat dinucleotide model regime such that we use an euristic to extract the most likely realization. This choice has been made because the full handling through a forward algorithm would not be able to cope with e.g context dependent errors.

By construction the Insertion event must have been explored first

5.10.2 Member Function Documentation

5.10.2.1 add_to_marginals()

```
void Dinucl_markov::add_to_marginals (
            long double scenario_proba,
            Marginal_array_p & updated_marginals ) const  [virtual]
```

Bug Will only count realizations of unambiguous nucleotides (realization indices>=0 since they are set to -1 in iterate_common)

Implements Rec_Event.

5.10.2.2 iterate()

```
void Dinucl_markov::iterate (
             double & ,
             Downstream_scenario_proba_bound_map & ,
             const std::string & ,
             const Int_Str & ,
             Index_map & ,
             const std::unordered_map< Rec_Event_name, std::vector< std::pair< std::shared_↩
ptr< const Rec_Event >, int >>> & ,
             std::shared_ptr< Next_event_ptr > & ,
             Marginal_array_p & ,
             const Marginal_array_p & ,
             const std::unordered_map< Gene_class, std::vector< Alignment_data >> & ,
             Seq_type_str_p_map & ,
             Seq_offsets_map & ,
             std::shared_ptr< Error_rate > & ,
             std::map< size_t, std::shared_ptr< Counter >> & ,
             const std::unordered_map< std::tuple< Event_type, Gene_class, Seq_side >, std↩
::shared_ptr< Rec_Event >> & ,
             Safety_bool_map & ,
             Mismatch_vectors_map & ,
             double & ,
             double & ) [inline], [virtual]
```

Evaluate all Event_Realization of a RecEvent for a given sequence.

Author

Q.Marcou

Version

1.0

Parameters

in,out	*scenario_proba*	Probability of the currently explored (incomplete) scenario
in,out	*downstream_proba_map*	
in	*sequence*	The studied sequence in nucleotide code
in	*int_sequence*	The studied sequence in integer code
in,out	*base_index_map*	Dynamic map recording where probabilities should be read on the marginals.
in	*offset_map*	Tells the event by how much indices from the children events should be modified
in	*next_event_ptr_arr*	Indicates the next event to call iterate on
in	*updated_marginals_point*	Summary marginals on which complete scenario posteriors are recorded
in	*model_parameters_point*	Current recombination probability distribution
in	*allowed_realizations*	The set of genomic templates alignment
in,out	*constructed_sequences*	Map containing the (incomplete) scenario's resulting sequence
in,out	*seq_offsets*	Map containing the 3' and 5' offsets of each scenario sequence piece
in	*error_rate_p*	Pointer to the error model object
in	*counters_list*	The list of Counter to be counted

Parameters

in	*events_map*	A map containing all events contained in the Model_parms, accessible through their type, gene class and side.
in,out	*safety_set*	A map indicating whether checks on offsets overlap should be performed
in,out	*mismatches_lists*	A map containing the (incomplete) scenario mismatches
in	*seq_max_prob_scenario*	Most likely scenario's probability for the considered sequence
in	*proba_threshold_factor*	Threshold on probability ratio between most likely scenario and explored scenario

Returns

void

The iterate method is the heart of IGoR's scenario exploration. Model_Parms define an order in which the RecEvent should be processed. Upon call of iterate all EventRealization of the RecEvent are assessed and for each possible realization the iterate method is called recursively for the next event. Inside the iterate method a filtering on too improbable realizations is performed (tree prunning) using the downstream_proba_map.

The index map is used to read off the probability of the current RecEvent EventRealization at the correct location given the events parent's realizations. It is further modified to take into account the current event's realization when its children realization probabilities will be read.

Implements Rec_Event.

5.10.2.3 update_event_internal_probas()

```
void Dinucl_markov::update_event_internal_probas (
            const Marginal_array_p & ,
            const std::unordered_map< Rec_Event_name, int > & )  [virtual]
```

Update probability values contained in a matrix where coordinates >=4 indicates ambiguous nucleotides We simply take the average probability over the different possible nucleotides.

Reimplemented from Rec_Event.

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

- Dinuclmarkov.h
- Dinuclmarkov.cpp

5.11 Enum_fast_memory_dual_key_map< K1, K2, V > Class Template Reference

Public Member Functions

- **Enum_fast_memory_dual_key_map** (size_t Key1_range, size_t Key2_range)
- V & **at** (const K1 &key1, const K2 &key2)
- const V & **at** (const K1 &key1, const K2 &key2) const
- V & **at** (const K1 &key1, const K2 &key2, int memory_layer)
- const V & **at** (const K1 &key1, const K2 &key2, int memory_layer) const
- int **get_current_memory_layer** (const K1 &key1, const K2 &key2)
- void **request_memory_layer** (const K1 &key1, const K2 &key2)
- void **set_value** (const K1 &key1, const K2 &key2, V value, int memory_layer)

Protected Attributes

- V ∗ **value_ptr_arr**
- int ∗ **memory_layer_ptr**
- int **max_layer**
- size_t **range_key1**
- size_t **range_key2**
- size_t **total_range**

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

- Utils.h

5.12 Enum_fast_memory_map< K, V > Class Template Reference

Public Member Functions

- **Enum_fast_memory_map** (int defined_range)
- V & **operator[ ]** (const K &key)
- V & **at** (const K &key)
- V & **at** (const K &key, int memory_layer)
- const V & **at** (const K &key, int memory_layer) const
- int **get_current_memory_layer** (const K &key)
- void **get_all_current_memory_layer** (int ∗memory_layers_recipient)
- bool **exist** (const K &key)
- void **request_memory_layer** (const K &key)
- void **set_value** (const K &key, const V &value, int memory_layer)
- void **multiply_all** (double &prod_operand, int ∗memory_adresses)
- void **reset** ()
- void **init_first_layer** (V value)

Protected Attributes

- V ∗ **value_ptr_arr**
- int ∗ **memory_layer_ptr**
- int **max_layer**
- size_t **range**

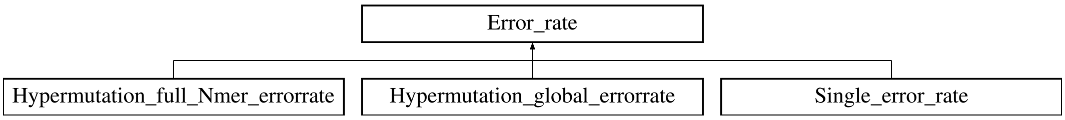
The documentation for this class was generated from the following file:

- Utils.h

5.13 Error_rate Class Reference

Abstract class for generic error models behavior.

Inheritance diagram for Error_rate:


Error_rate
Hypermutation_full_Nmer_errorrate
Hypermutation_global_errorrate
Single_error_rate

Public Member Functions

- virtual double **compare_sequences_error_prob** (double, const std::string &, Seq_type_str_p_map &, const Seq_offsets_map &, const std::unordered_map< std::tuple< Event_type, Gene_class, Seq_side >, std::shared_ptr< Rec_Event >> &, Mismatch_vectors_map &, double &, double &)=0
- virtual void **update** ()=0
- virtual void **initialize** (const std::unordered_map< std::tuple< Event_type, Gene_class, Seq_side >, std::shared_ptr< Rec_Event >> &)
- bool **is_updated** () const
- void **update_value** (bool update_status)
- virtual void **add_to_norm_counter** ()=0
- virtual void **clean_seq_counters** ()=0
- void **norm_weights_by_seq_likelihood** (Marginal_array_p &, const size_t, const double seq_weight=1)
- virtual void **write2txt** (std::ofstream &)=0
- virtual std::shared_ptr< Error_rate > **copy** () const =0
- virtual std::string **type** () const =0
- virtual Error_rate ∗ **add_checked** (Error_rate ∗)=0
- double **get_model_likelihood** () const
- double **get_seq_likelihood** () const
- double **get_seq_probability** () const
- double **get_seq_mean_error_number** () const
- virtual const double & **get_err_rate_upper_bound** (size_t, size_t)=0
- virtual void **build_upper_bound_matrix** (size_t, size_t)=0
- virtual int **get_number_non_zero_likelihood_seqs** () const =0
- virtual std::queue< int > **generate_errors** (std::string &, std::mt19937_64 &) const =0
- void **set_viterbi_run** (bool viterbi_like)

Public Attributes

- int **debug_number_scenarios**

Protected Attributes

- bool **updated**
- long double **model_log_likelihood**
- int **number_seq**
- long double **seq_likelihood**
- double **seq_mean_error_number**
- long double **scenario_new_proba**
- long double **seq_probability**
- bool **viterbi_run**
- Matrix< double > **upper_bound_proba_mat**
- size_t **max_err**
- size_t **max_noerr**

5.13.1 Detailed Description

Abstract class for generic error models behavior.

Author

Q.Marcou

Version

1.0

Base class for defining different error models such as additive or non-additive hypermutation models. Errors are assessed when all RecEvent iterate have been processed (terminal leaf of the scenario tree)

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

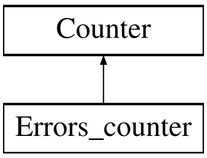
- Errorrate.h
- Errorrate.cpp

5.14 Errors_counter Class Reference

Counter recording the number of genomic nucleotides and errors/mismatch per scenario.

```
#include <Errorscounter.h>
```

Inheritance diagram for Errors_counter:



Public Member Functions

- **Errors_counter** (size_t)
- **Errors_counter** (size_t, bool)
- **Errors_counter** (size_t, std::string)
- **Errors_counter** (size_t, std::string, bool)
- std::string **type** () const
- void **initialize_counter** (const Model_Parms &, const Model_marginals &)
- void **count_scenario** (long double, double, const std::string &, Seq_type_str_p_map &, const Seq_offsets↩_map &, const std::unordered_map< std::tuple< Event_type, Gene_class, Seq_side >, std::shared_ptr< Rec_Event >> &, Mismatch_vectors_map &)
- void **count_sequence** (double, const Model_marginals &, const Model_Parms &)
- void **add_checked** (std::shared_ptr< Counter >)
- void **dump_sequence_data** (int, int)
- std::shared_ptr< Counter > **copy** () const

Additional Inherited Members

5.14.1 Detailed Description

Counter recording the number of genomic nucleotides and errors/mismatch per scenario.

Author

Q.Marcou

Version

1.0

Counter recording the number of genomic nucleotides and errors/mismatch per scenario. This information can either be recorded for the N best scenarios or be aggregated to extract individual sequence posterior error/mutation load.

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

- Errorscounter.h
- Errorscounter.cpp

5.15 Event_comparator Struct Reference

Public Member Functions

- bool **operator()** (std::shared_ptr< const Rec_Event > event_p1, std::shared_ptr< const Rec_Event > event_p2)

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

- Rec_Event.h

5.16 Event_realization Struct Reference

Unit that stores an event realization name, value and index.

```
#include <Rec_Event.h>
```

Public Member Functions

- **Event_realization** (std::string real_name, int val_int, std::string val_str, Int_Str val_str_int, int index_val)

Public Attributes

- const std::string **name**
- const int **value_int**
- const std::string **value_str**
- const Int_Str **value_str_int**
- int **index**

5.16.1 Detailed Description

Unit that stores an event realization name, value and index.

Author

Q.Marcou

Version

1.0

Depending on the RecEvent type to which it belongs, the Event_realization must supply either a string (both std↵ ::string and IntStr) or an integer value. Integers values are e.g the number of deletions or insertions of Insertion or Deletion RecEvent String values are e.g realization of a GeneChoice Rec_Event, and stands for the gene sequence.

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

- Rec_Event.h

5.17 ExtractFeatures Class Reference

Class to extract sequences features (e.g. CDR3) of sequences using alignment and V, J anchors information.

```
#include <ExtractFeatures.h>
```

Public Member Functions

- **ExtractFeatures** (const ExtractFeatures &orig)
- void load_VJgenomicTemplates (vector< pair< string, string >> v_genomic, vector< pair< string, string >> j_genomic)

  Function to map the templateID and the sequence for V and J.
- void load_VJanchors (string flnV_CDR3_anchors, string flnJ_CDR3_anchors)

  load data files into GeneFeatures functor class
- void load_VJanchors (unordered_map< string, size_t > flnV_CDR3_anchors, unordered_map< string, size_t > flnJ_CDR3_anchors)

  load data files into GeneFeatures functor class
- void **print_VgenomicTemplates** ()
- void **print_JgenomicTemplates** ()
- void **set_sorted_alignments** (unordered_map< int, pair< string, unordered_map< Gene_class, vector< Alignment_data >>>> *pointer)
- CDR3SeqData extractCDR3 (int seq_index)

  Get the CDR3 as an instance of CDR3SeqData given the V and J Alignment_data structs.
- int getVAnchor4Seq (string seq_str, Alignment_data v_alig)
- int getJAnchor4Seq (string seq_str, Alignment_data j_alig)
- string generateCDR3_csv_line (CDR3SeqData cdr3InputSeq)

  Function to generate a CDR3 string line to be printed on a file.

Public Attributes

- unordered_map< string, string > **UMap_v_genomic**
- unordered_map< string, string > **UMap_j_genomic**
- unordered_map< string, size_t > **UMap_v_CDR3_anchors**
- unordered_map< string, size_t > **UMap_j_CDR3_anchors**
- unordered_map< int, pair< string, unordered_map< Gene_class, vector< Alignment_data > > > > ∗ **p_sorted_alignments**

5.17.1 Detailed Description

Class to extract sequences features (e.g. CDR3) of sequences using alignment and V, J anchors information.

Author

C. Olivares

5.17.2 Member Function Documentation

5.17.2.1 extractCDR3()

```
CDR3SeqData ExtractFeatures::extractCDR3 (
              int seq_index )
```

Get the CDR3 as an instance of CDR3SeqData given the V and J Alignment_data structs.

Parameters

seq_index	
V_alignment	
J_alignment	

Returns

CDR3SeqData

5.17.2.2 generateCDR3_csv_line()

```
string ExtractFeatures::generateCDR3_csv_line (
              CDR3SeqData cdr3InputSeq )
```

Function to generate a CDR3 string line to be printed on a file.

Parameters

cdr3InputSeq	is CDR3SeqData instance

Returns

csvline line string to be printed on a file.

5.17.2.3 getJAnchor4Seq()

```
int ExtractFeatures::getJAnchor4Seq (
            string seq_str,
            Alignment_data j_alig )
```

Calculate the J anchor in reference to the input sequence

Parameters

seq_str	
j_alig	

Returns

cdr3_j_read_anch the anchor in reference to the input sequence.

5.17.2.4 getVAnchor4Seq()

```
int ExtractFeatures::getVAnchor4Seq (
            string seq_str,
            Alignment_data v_alig )
```

Calculate the V anchor in reference to the input sequence

Parameters

seq_str	
v_alig	

Returns

cdr3_v_read_anch the anchor in reference to the input sequence.

5.17.2.5 load_VJanchors() [1/2]

```
void ExtractFeatures::load_VJanchors (
            string flnV_CDR3_anchors,
            string flnJ_CDR3_anchors )
```

load data files into GeneFeatures functor class

Parameters

flnV_CDR3_anchors	CDR3 anchors filename for V genes
flnJ_CDR3_anchors	CDR3 anchors filename for J genes

5.17.2.6 load_VJanchors() [2/2]

```
void ExtractFeatures::load_VJanchors (
            unordered_map< string, size_t > v_CDR3_anchors,
            unordered_map< string, size_t > j_CDR3_anchors )
```

load data files into GeneFeatures functor class

Parameters

v_CDR3_anchors	unordered_map of sequence description and position for V genes CDR3 anchors.
j_CDR3_anchors	unordered_map of sequence description and position for J genes CDR3 anchors.

5.17.2.7 load_VJgenomicTemplates()

```
void ExtractFeatures::load_VJgenomicTemplates (
            vector< pair< string, string >> v_genomic,
            vector< pair< string, string >> j_genomic )
```

Function to map the templateID and the sequence for V and J.

Parameters

v_genomic	vector with the genomic templates for V genes
j_genomic	vector with the genomic templates for J genes

Returns

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

- ExtractFeatures.h
- ExtractFeatures.cpp

5.18 gen_CDR3_data Struct Reference

```
#include <GenModel.h>
```

Public Member Functions

- **gen_CDR3_data** (const std::unordered_map< std::string, size_t > &v_anchors_indices, const std::unordered_map< std::string, Event_realization > &v_reals, size_t v_event_pos, const std::unordered_map< std::string, size_t > &j_anchors_indices, const std::unordered_map< std::string, Event_realization > &j_reals, size_t j_event_pos, std::shared_ptr< std::ostream > output_stream_ptr=std::shared_ptr< std::ostream >(&std::cout, null_delete< std::ostream >()))

Public Attributes

- std::map< int, std::tuple< std::string, size_t, size_t, std::string > > **v_anchors**
- size_t **v_event_queue_position**
- std::map< int, std::tuple< std::string, size_t, size_t, std::string > > **j_anchors**
- size_t **j_event_queue_position**
- std::shared_ptr< std::ostream > **output_stream**
- bool **output_nt_CDR3** = true
- bool **output_anchors_found** = true
- bool **output_inframe** = true
- bool **output_aa_CDR3** = false
- bool **output_productive** = false

5.18.1 Detailed Description

Hardcode a data structure for the function extracting CDR3s in generated sequences

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

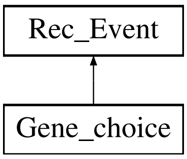
- GenModel.h

5.19 Gene_choice Class Reference

GeneChoice recombination event.

```
#include <Genechoice.h>
```

Inheritance diagram for Gene_choice:



Public Member Functions

- **Gene_choice** (Gene_class)
- **Gene_choice** (Gene_class, std::unordered_map< std::string, Event_realization > &)
- **Gene_choice** (Gene_class, std::vector< std::pair< std::string, std::string >>)
- std::shared_ptr< Rec_Event > **copy** ()
- void iterate (double &, Downstream_scenario_proba_bound_map &, const std::string &, const Int_Str &, Index_map &, const std::unordered_map< Rec_Event_name, std::vector< std::pair< std::shared_ptr< const Rec_Event >, int >>> &, std::shared_ptr< Next_event_ptr > &, Marginal_array_p &, const Marginal_array_p &, const std::unordered_map< Gene_class, std::vector< Alignment_data >> &, Seq_type_str_p_map &, Seq_offsets_map &, std::shared_ptr< Error_rate > &, std::map< size_t, std::shared_ptr< Counter >> &, const std::unordered_map< std::tuple< Event_type, Gene_class, Seq_side >, std::shared_ptr< Rec_Event >> &, Safety_bool_map &, Mismatch_vectors_map &, double &, double &)

  Evaluate all Event_Realization of a RecEvent for a given sequence.
- void **add_realization** (int)
- bool **add_realization** (std::string gene_name, std::string gene_sequence)
- void **set_genomic_templates** (const std::vector< std::pair< std::string, std::string >> &)
- std::queue< int > **draw_random_realization** (const Marginal_array_p &, std::unordered_map< Rec_Event_name, int > &, const std::unordered_map< Rec_Event_name, std::vector< std::pair< std::shared_ptr< const Rec_Event >, int >>> &, std::unordered_map< Seq_type, std::string > &, std::mt19937_64 &) const
- void **write2txt** (std::ofstream &)
- void **initialize_event** (std::unordered_set< Rec_Event_name > &, const std::unordered_map< std::tuple< Event_type, Gene_class, Seq_side >, std::shared_ptr< Rec_Event >> &, const std::unordered_map< Rec_Event_name, std::vector< std::pair< std::shared_ptr< const Rec_Event >, int >>> &, Downstream_scenario_proba_bound_map &, Seq_type_str_p_map &, Safety_bool_map &, std::shared_ptr< Error_rate >, Mismatch_vectors_map &, Seq_offsets_map &, Index_map &)
- void add_to_marginals (long double, Marginal_array_p &) const
- bool **has_effect_on** (Seq_type) const
- void **iterate_initialize_Len_proba** (Seq_type considered_junction, std::map< int, double > &length_best_proba_map, std::queue< std::shared_ptr< Rec_Event >> &model_queue, double &scenario_proba, const Marginal_array_p &model_parameters_point, Index_map &base_index_map, Seq_type_str_p_map &constructed_sequences, int &seq_len) const
- void **initialize_Len_proba_bound** (std::queue< std::shared_ptr< Rec_Event >> &model_queue, const Marginal_array_p &model_parameters_point, Index_map &base_index_map)

Friends

- class **Coverage_err_counter**
- class **Hypermutation_global_errorrate**
- class **Hypermutation_full_Nmer_errorrate**

Additional Inherited Members

5.19.1 Detailed Description

GeneChoice recombination event.

Author

Q.Marcou

Version

1.0

Models the gene choice recombination process. The event realizations are explored based on the sequence alignments that were provdided to the inference. Since D gene can be heavily deleted and might not be recognizable by sequence alignments, a special handling of the D gene choice exploring all D positions ranked by their likelihood has been implemented.

5.19.2 Member Function Documentation

5.19.2.1 add_to_marginals()

```
void Gene_choice::add_to_marginals (
            long double scenario_proba,
            Marginal_array_p & updated_marginals ) const  [virtual]
```

All add_to_marginals should take into account the possibility to perform viterbi runs(take only the most likely scenario into account)

Implements Rec_Event.

5.19.2.2 iterate()

```
void Gene_choice::iterate (
            double & ,
            Downstream_scenario_proba_bound_map & ,
            const std::string & ,
            const Int_Str & ,
            Index_map & ,
            const std::unordered_map< Rec_Event_name, std::vector< std::pair< std::shared_←
ptr< const Rec_Event >, int >>> & ,
            std::shared_ptr< Next_event_ptr > & ,
            Marginal_array_p & ,
            const Marginal_array_p & ,
            const std::unordered_map< Gene_class, std::vector< Alignment_data >> & ,
            Seq_type_str_p_map & ,
            Seq_offsets_map & ,
            std::shared_ptr< Error_rate > & ,
            std::map< size_t, std::shared_ptr< Counter >> & ,
            const std::unordered_map< std::tuple< Event_type, Gene_class, Seq_side >, std←
::shared_ptr< Rec_Event >> & ,
            Safety_bool_map & ,
            Mismatch_vectors_map & ,
            double & ,
            double &  )  [inline], [virtual]
```

Evaluate all Event_Realization of a RecEvent for a given sequence.

Author

Q.Marcou

Version

1.0

Parameters

in,out	*scenario_proba*	Probability of the currently explored (incomplete) scenario
in,out	*downstream_proba_map*	
in	*sequence*	The studied sequence in nucleotide code
in	*int_sequence*	The studied sequence in integer code
in,out	*base_index_map*	Dynamic map recording where probabilities should be read on the marginals.
in	*offset_map*	Tells the event by how much indices from the children events should be modified
in	*next_event_ptr_arr*	Indicates the next event to call iterate on
in	*updated_marginals_point*	Summary marginals on which complete scenario posteriors are recorded
in	*model_parameters_point*	Current recombination probability distribution
in	*allowed_realizations*	The set of genomic templates alignment
in,out	*constructed_sequences*	Map containing the (incomplete) scenario's resulting sequence
in,out	*seq_offsets*	Map containing the 3' and 5' offsets of each scenario sequence piece
in	*error_rate_p*	Pointer to the error model object
in	*counters_list*	The list of Counter to be counted
in	*events_map*	A map containing all events contained in the Model_parms, accessible through their type, gene class and side.
in,out	*safety_set*	A map indicating whether checks on offsets overlap should be performed
in,out	*mismatches_lists*	A map containing the (incomplete) scenario mismatches
in	*seq_max_prob_scenario*	Most likely scenario's probability for the considered sequence
in	*proba_threshold_factor*	Threshold on probability ratio between most likely scenario and explored scenario

Returns

void

The iterate method is the heart of IGoR's scenario exploration. Model_Parms define an order in which the RecEvent should be processed. Upon call of iterate all EventRealization of the RecEvent are assessed and for each possible realization the iterate method is called recursively for the next event. Inside the iterate method a filtering on too improbable realizations is performed (tree prunning) using the downstream_proba_map.

The index map is used to read off the probability of the current RecEvent EventRealization at the correct location given the events parent's realizations. It is further modified to take into account the current event's realization when its children realization probabilities will be read.

Implements Rec_Event.

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

- Genechoice.h
- Genechoice.cpp

5.20 GenModel Class Reference

High level V(D)J generative model.

```
#include <GenModel.h>
```

Public Member Functions

- **GenModel** (const Model_Parms &)
- **GenModel** (const Model_Parms &, const Model_marginals &)
- **GenModel** (const Model_Parms &, const Model_marginals &, const std::map< size_t, std::shared_ptr< Counter >> &)
- bool **infer_model** (const std::vector< std::tuple< int, std::string, std::unordered_map< Gene_class, std↩ ::vector< Alignment_data >>>> &sequences, const int iterations, const std::string path, bool fast_iter, double likelihood_threshold=1e-25, bool viterbi_like=false)
- bool **infer_model** (const std::vector< std::tuple< int, std::string, std::unordered_map< Gene_class, std↩ ::vector< Alignment_data >>>> &sequences, const int iterations, const std::string path, bool fast_iter=true, double likelihood_threshold=1e-25, double proba_threshold_factor=0.001)
- bool **infer_model** (const std::vector< std::tuple< int, std::string, std::unordered_map< Gene_class, std↩ ::vector< Alignment_data >>>> &sequences, const int iterations, const std::string path, bool fast_iter, double likelihood_threshold, bool viterbi_like, double proba_threshold_factor, double mean_number_seq_err_↩ thresh=INFINITY)
- std::forward_list< std::pair< std::string, std::queue< std::queue< int > > > > generate_sequences (int, bool)
- void **generate_sequences** (int, bool, std::string, std::string, std::list< std::pair< gen_seq_trans, std↩ ::shared_ptr< void >>>=std::list< std::pair< gen_seq_trans, std::shared_ptr< void >>>(), bool output↩ _only_func=false, int=-1)
- bool **load_genmodel** ()
- bool **write2txt** ()
- bool **readtxt** ()
- void **write_seq2txt** (std::string, std::forward_list< std::string >)
- void **write_seq_real2txt** (std::string, std::string, std::forward_list< std::pair< std::string, std::queue< std↩ ::queue< int >>>>)

5.20.1 Detailed Description

High level V(D)J generative model.

Author

Q.Marcou

Version

1.0

Highest level class to model the V(D)J recombination and subsequent processes. It contains the model's graph structure (Model_Parms), the associated probability distribution (Model_Marginals). The GenModel class provides high level functions to perform inference / sequence annotation as well as generating random sequences from the model.

5.20.2 Member Function Documentation

5.20.2.1 generate_sequences()

```
forward_list< pair< string, queue< queue< int > > > > GenModel::generate_sequences (
             int number_seq,
             bool generate_errors )
```

Deprecated This function used to store generated sequences in memory, and quickly overloaded it for large number of generated sequences.

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

- GenModel.h
- GenModel.cpp

5.21 std::hash< Event_safety > Struct Template Reference

Public Member Functions

- std::size_t **operator()** (const Event_safety ev_saf) const

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

- Utils.h

5.22 std::hash< Gene_class > Struct Template Reference

Public Member Functions

- std::size_t **operator()** (const Gene_class &gene) const

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

- Utils.h

5.23 std::hash< Int_Str > Struct Template Reference

Public Member Functions

- std::size_t **operator()** (Int_Str const &int_str) const

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

- IntStr.h

5.24 std::hash< Seq_type > Struct Template Reference

Public Member Functions

- std::size_t **operator()** (const Seq_type &seq_t) const

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

- Utils.h

5.25 std::hash< std::pair< Gene_class, Seq_side > > Struct Template Reference

Public Member Functions

- std::size_t **operator()** (const pair< Gene_class, Seq_side > &gene_pair) const

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

- Utils.h

5.26 std::hash< std::pair< Seq_type, Seq_side > > Struct Template Reference

Public Member Functions

- std::size_t **operator()** (const std::pair< Seq_type, Seq_side > seq_pair) const

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

- Utils.h

5.27 std::hash< std::tuple< Event_type, Gene_class, Seq_side > > Struct Template Reference

Public Member Functions

- std::size_t **operator()** (const std::tuple< Event_type, Gene_class, Seq_side > &event_triplet) const

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

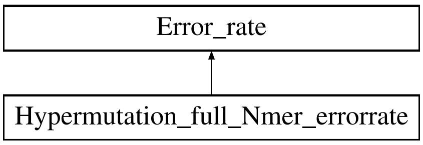
- Utils.h

5.28 Hypermutation_full_Nmer_errorrate Class Reference

A non-additive context dependent hypermutation model.

```
#include <HypermutationfullNmererrorrate.h>
```

Inheritance diagram for Hypermutation_full_Nmer_errorrate:



Public Member Functions

- **Hypermutation_full_Nmer_errorrate** (size_t, Gene_class, Gene_class, double, size_t=0)
- **Hypermutation_full_Nmer_errorrate** (size_t, Gene_class, Gene_class, std::vector< double >, size_t=0)
- **Hypermutation_full_Nmer_errorrate** (size_t, Gene_class, Gene_class, double, std::string, size_t=0)
- **Hypermutation_full_Nmer_errorrate** (size_t, Gene_class, Gene_class, std::vector< double >, std::string, size_t=0)
- double **compare_sequences_error_prob** (double, const std::string &, Seq_type_str_p_map &, const Seq_offsets_map &, const std::unordered_map< std::tuple< Event_type, Gene_class, Seq_side >, std::shared_ptr< Rec_Event >> &, Mismatch_vectors_map &, double &, double &)
- void **update** ()
- void **initialize** (const std::unordered_map< std::tuple< Event_type, Gene_class, Seq_side >, std::shared_ptr< Rec_Event >> &)
- void **add_to_norm_counter** ()
- void **clean_seq_counters** ()
- void **clean_all_counters** ()
- void **write2txt** (std::ofstream &)
- void **set_output_Nmer_stream** (std::string)
- std::shared_ptr< Error_rate > **copy** () const
- std::string **type** () const
- Hypermutation_full_Nmer_errorrate & **operator+=** (Hypermutation_full_Nmer_errorrate)
- Error_rate ∗ **add_checked** (Error_rate ∗)
- const double & **get_err_rate_upper_bound** (size_t, size_t)
- void **build_upper_bound_matrix** (size_t, size_t)
- int **get_number_non_zero_likelihood_seqs** () const
- std::queue< int > **generate_errors** (std::string &, std::mt19937_64 &) const
- uint64_t **generate_random_mutation_probas** (double, double)

Additional Inherited Members

5.28.1 Detailed Description

A non-additive context dependent hypermutation model.

Author

Q.Marcou

Version

1.0

A specialization of the ErrorRate class. Implements a context dependent hypermutation/error model with tunable context size. A different mutation probability is recorded for each context of size N, leading to 4^N parameters. This model is inspired from the S5F mutability model. The identity of the resulting nucleotide after mutation is assumed to follow a uniform distribution.

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

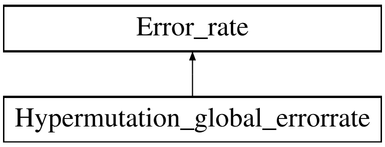
- HypermutationfullNmererrorrate.h
- HypermutationfullNmererrorrate.cpp

5.29 Hypermutation_global_errorrate Class Reference

An additive (independent site) context dependent hypermutation model.

```
#include <Hypermutationglobalerrorrate.h>
```

Inheritance diagram for Hypermutation_global_errorrate:



Public Member Functions

- **Hypermutation_global_errorrate** (size_t, Gene_class, Gene_class, double)
- **Hypermutation_global_errorrate** (size_t, Gene_class, Gene_class, double, std::vector< double >)
- **Hypermutation_global_errorrate** (size_t, Gene_class, Gene_class, double, std::string)
- **Hypermutation_global_errorrate** (size_t, Gene_class, Gene_class, double, std::vector< double >, std::string)
- double **compare_sequences_error_prob** (double, const std::string &, Seq_type_str_p_map &, const Seq_offsets_map &, const std::unordered_map< std::tuple< Event_type, Gene_class, Seq_side >, std::shared_ptr< Rec_Event >> &, Mismatch_vectors_map &, double &, double &)
- void **update** ()
- void **initialize** (const std::unordered_map< std::tuple< Event_type, Gene_class, Seq_side >, std::shared_ptr< Rec_Event >> &)
- void **add_to_norm_counter** ()
- void **clean_seq_counters** ()
- void **clean_all_counters** ()
- void **write2txt** (std::ofstream &)
- void **set_output_Nmer_stream** (std::string)
- std::shared_ptr< Error_rate > **copy** () const
- std::string **type** () const
- Hypermutation_global_errorrate & **operator+=** (Hypermutation_global_errorrate)
- Error_rate ∗ **add_checked** (Error_rate ∗)
- const double & **get_err_rate_upper_bound** (size_t, size_t)
- void **build_upper_bound_matrix** (size_t, size_t)
- int **get_number_non_zero_likelihood_seqs** () const
- std::queue< int > **generate_errors** (std::string &, std::mt19937_64 &) const
- uint64_t **generate_random_contributions** (double)

Additional Inherited Members

5.29.1 Detailed Description

An additive (independent site) context dependent hypermutation model.

Author

Q.Marcou

Version

1.0

A specialization of the ErrorRate class. Implements a context dependent hypermutation/error model with tunable context size. Nucleotide from the context are assumed to contribute independently to the mutability of the context through an additive logarithmic score. Such a model contains only 3N+1 parameters and allows to probe large context sizes. The identity of the resulting nucleotide after mutation is assumed to follow a uniform distribution.

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

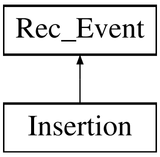
- Hypermutationglobalerrorrate.h
- Hypermutationglobalerrorrate.cpp

5.30 Insertion Class Reference

Insertion recombination events.

```
#include <Insertion.h>
```

Inheritance diagram for Insertion:



Public Member Functions

- **Insertion** (Gene_class, std::pair< int, int >)
- **Insertion** (Gene_class, std::forward_list< int >)
- **Insertion** (Gene_class)
- **Insertion** (Gene_class, std::unordered_map< std::string, Event_realization > &)
- std::shared_ptr< Rec_Event > **copy** ()
- void iterate (double &, Downstream_scenario_proba_bound_map &, const std::string &, const Int_Str &, Index_map &, const std::unordered_map< Rec_Event_name, std::vector< std::pair< std::shared_ptr< const Rec_Event >, int >>> &, std::shared_ptr< Next_event_ptr > &, Marginal_array_p &, const Marginal_array_p &, const std::unordered_map< Gene_class, std::vector< Alignment_data >> &, Seq_type_str_p_map &, Seq_offsets_map &, std::shared_ptr< Error_rate > &, std::map< size_t, std::shared_ptr< Counter >> &, const std::unordered_map< std::tuple< Event_type, Gene_class, Seq_side >, std::shared_ptr< Rec_Event >> &, Safety_bool_map &, Mismatch_vectors_map &, double &, double &)

  Evaluate all Event_Realization of a RecEvent for a given sequence.
- bool **add_realization** (int)
- std::queue< int > **draw_random_realization** (const Marginal_array_p &, std::unordered_map< Rec_Event_name, int > &, const std::unordered_map< Rec_Event_name, std::vector< std::pair< std::shared_ptr< const Rec_Event >, int >>> &, std::unordered_map< Seq_type, std::string > &, std::mt19937_64 &) const
- void **write2txt** (std::ofstream &)
- void **initialize_event** (std::unordered_set< Rec_Event_name > &, const std::unordered_map< std::tuple< Event_type, Gene_class, Seq_side >, std::shared_ptr< Rec_Event >> &, const std::unordered_map< Rec_Event_name, std::vector< std::pair< std::shared_ptr< const Rec_Event >, int >>> &, Downstream_scenario_proba_bound_map &, Seq_type_str_p_map &, Safety_bool_map &, std::shared_ptr< Error_rate >, Mismatch_vectors_map &, Seq_offsets_map &, Index_map &)
- void **add_to_marginals** (long double, Marginal_array_p &) const
- void **set_crude_upper_bound_proba** (size_t, size_t, Marginal_array_p &)
- void **initialize_crude_scenario_proba_bound** (double &, std::forward_list< double ∗> &, const std::unordered_map< std::tuple< Event_type, Gene_class, Seq_side >, std::shared_ptr< Rec_Event >> &)
- bool **has_effect_on** (Seq_type) const
- void **iterate_initialize_Len_proba** (Seq_type considered_junction, std::map< int, double > &length_best_proba_map, std::queue< std::shared_ptr< Rec_Event >> &model_queue, double &scenario_proba, const Marginal_array_p &model_parameters_point, Index_map &base_index_map, Seq_type_str_p_map &constructed_sequences, int &seq_len) const
- void **initialize_Len_proba_bound** (std::queue< std::shared_ptr< Rec_Event >> &model_queue, const Marginal_array_p &model_parameters_point, Index_map &base_index_map)

Additional Inherited Members

5.30.1 Detailed Description

Insertion recombination events.

Author

Q.Marcou

Version

1.0

The Insertion RecEvent models the distribution of junctional insertion length during the V(D)J recombination process.

5.30.2 Member Function Documentation

5.30.2.1 iterate()

```
void Insertion::iterate (
            double & ,
            Downstream_scenario_proba_bound_map & ,
            const std::string & ,
            const Int_Str & ,
            Index_map & ,
            const std::unordered_map< Rec_Event_name, std::vector< std::pair< std::shared_↩
ptr< const Rec_Event >, int >>> & ,
            std::shared_ptr< Next_event_ptr > & ,
            Marginal_array_p & ,
            const Marginal_array_p & ,
            const std::unordered_map< Gene_class, std::vector< Alignment_data >> & ,
            Seq_type_str_p_map & ,
            Seq_offsets_map & ,
            std::shared_ptr< Error_rate > & ,
            std::map< size_t, std::shared_ptr< Counter >> & ,
            const std::unordered_map< std::tuple< Event_type, Gene_class, Seq_side >, std↩
::shared_ptr< Rec_Event >> & ,
            Safety_bool_map & ,
            Mismatch_vectors_map & ,
            double & ,
            double & ) [inline], [virtual]
```

Evaluate all Event_Realization of a RecEvent for a given sequence.

Author

Q.Marcou

Version

1.0

Parameters

in,out	*scenario_proba*	Probability of the currently explored (incomplete) scenario
in,out	*downstream_proba_map*	
in	*sequence*	The studied sequence in nucleotide code
in	*int_sequence*	The studied sequence in integer code
in,out	*base_index_map*	Dynamic map recording where probabilities should be read on the marginals.
in	*offset_map*	Tells the event by how much indices from the children events should be modified
in	*next_event_ptr_arr*	Indicates the next event to call iterate on
in	*updated_marginals_point*	Summary marginals on which complete scenario posteriors are recorded
in	*model_parameters_point*	Current recombination probability distribution

Parameters

in	*allowed_realizations*	The set of genomic templates alignment
in,out	*constructed_sequences*	Map containing the (incomplete) scenario's resulting sequence
in,out	*seq_offsets*	Map containing the 3' and 5' offsets of each scenario sequence piece
in	*error_rate_p*	Pointer to the error model object
in	*counters_list*	The list of Counter to be counted
in	*events_map*	A map containing all events contained in the Model_parms, accessible through their type, gene class and side.
in,out	*safety_set*	A map indicating whether checks on offsets overlap should be performed
in,out	*mismatches_lists*	A map containing the (incomplete) scenario mismatches
in	*seq_max_prob_scenario*	Most likely scenario's probability for the considered sequence
in	*proba_threshold_factor*	Threshold on probability ratio between most likely scenario and explored scenario

Returns

void

The iterate method is the heart of IGoR's scenario exploration. Model_Parms define an order in which the RecEvent should be processed. Upon call of iterate all EventRealization of the RecEvent are assessed and for each possible realization the iterate method is called recursively for the next event. Inside the iterate method a filtering on too improbable realizations is performed (tree prunning) using the downstream_proba_map.

The index map is used to read off the probability of the current RecEvent EventRealization at the correct location given the events parent's realizations. It is further modified to take into account the current event's realization when its children realization probabilities will be read.

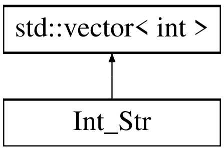
Implements Rec_Event.

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

- Insertion.h
- Insertion.cpp

5.31 Int_Str Class Reference

Inheritance diagram for Int_Str:



Public Member Functions

- Int_Str & **operator+=** (const Int_Str &)
- Int_Str & **operator+=** (const int &)
- Int_Str & **operator+=** (int &&)
- Int_Str & **append** (const Int_Str &)
- Int_Str & **append** (const int &)
- Int_Str **operator+** (const Int_Str &) const
- Int_Str **operator+** (const int &) const
- Int_Str **operator+** (int) const
- Int_Str **substr** (std::size_t pos=0, std::size_t len=npos) const
- Int_Str & **erase** (std::size_t pos, std::size_t len)

Static Public Attributes

- static const std::size_t **npos** = -1

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

- IntStr.h
- IntStr.cpp

5.32 inverse_offset_comparator Struct Reference

Public Member Functions

- bool **operator()** (const std::pair< std::shared_ptr< const Rec_Event >, int > &inv_offset_1, const std::pair< std::shared_ptr< const Rec_Event >, int > &inv_offset_2)

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

- Utils.h

5.33 Matrix< T > Struct Template Reference

Public Member Functions

- **Matrix** (int m, int n)
- **Matrix** (int m, int n, T arr[ ])
- **Matrix** (int m, int n, std::vector< T > vect)
- **Matrix** (const Matrix< T > &other)
- Matrix< T > & **operator=** (const Matrix &other)
- T & **operator()** (const int &i, const int &j)
- const T & **operator()** (const int &i, const int &j) const
- T **get_field** (const int &i, const int &j) const
- const int & **get_n_rows** () const
- const int & **get_n_cols** () const

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

- Utils.h

5.34 Model_marginals Class Reference

Encapsulates the marginal probabilities/posterior frequency for each recombination event's realization.

```
#include <Model_marginals.h>
```

Public Member Functions

- **Model_marginals** (const Model_Parms &)
- **Model_marginals** (const Model_marginals &)
- size_t **compute_size** (const Model_Parms &)
- size_t **get_event_size** (std::shared_ptr< const Rec_Event >, const Model_Parms &) const
- std::pair< std::list< std::pair< Rec_Event_name, size_t > >, std::shared_ptr< long double > > **compute_event_marginal_probability** (Rec_Event_name, const Model_Parms &) const
- std::pair< std::list< std::pair< Rec_Event_name, size_t > >, std::shared_ptr< long double > > **compute_event_marginal_probability** (Rec_Event_name, const std::set< Rec_Event_name > &, const Model_Parms &) const
- Model_marginals & **operator=** (const Model_marginals &)
- Model_marginals & **operator+=** (Model_marginals)
- Model_marginals & **operator-=** (Model_marginals)
- Model_marginals **operator+** (Model_marginals)
- Model_marginals **operator-** (Model_marginals)
- void **normalize** (std::unordered_map< Rec_Event_name, std::list< std::pair< std::shared_ptr< const Rec_Event >, int >>>, std::unordered_map< Rec_Event_name, int >, std::queue< std::shared_ptr< Rec_Event >>)
- void **uniform_initialize** (const Model_Parms &)
- void **null_initialize** ()
- void **random_initialize** (const Model_Parms &)
- void **flatten** (std::shared_ptr< const Rec_Event >, const Model_Parms &)
- void set_realization_proba (std::string, std::shared_ptr< const Rec_Event >, double, const Model_Parms &)
- void **add_pseudo_counts** (double)
- bool **add_to_marginals** (double event_proba, std::list< std::shared_ptr< Rec_Event >>, Model_Parms)
- void **copy_fixed_events_marginals** (const Model_marginals &, const Model_Parms &, const std::unordered_map< Rec_Event_name, int > &)
- std::unordered_map< Rec_Event_name, std::vector< std::pair< std::shared_ptr< const Rec_Event >, int > > > **get_offsets_map** (const Model_Parms &) const
- std::unordered_map< Rec_Event_name, std::vector< std::pair< std::shared_ptr< const Rec_Event >, int > > > **get_offsets_map** (const Model_Parms &, std::queue< std::shared_ptr< Rec_Event >>) const
- std::unordered_map< Rec_Event_name, std::list< std::pair< std::shared_ptr< const Rec_Event >, int > > > **get_inverse_offset_map** (const Model_Parms &) const
- std::unordered_map< Rec_Event_name, std::list< std::pair< std::shared_ptr< const Rec_Event >, int > > > **get_inverse_offset_map** (const Model_Parms &, std::queue< std::shared_ptr< Rec_Event >>) const
- std::unordered_map< Rec_Event_name, int > **get_index_map** (const Model_Parms &) const
- std::unordered_map< Rec_Event_name, int > **get_index_map** (const Model_Parms &, std::queue< std::shared_ptr< Rec_Event >>) const
- void **write2txt** (std::string, const Model_Parms &)
- void **txt2marginals** (std::string, const Model_Parms &)
- Model_marginals **empty_copy** ()
- Model_marginals & **invert_edge** (Rec_Event_name, Rec_Event_name, Model_Parms &)
- size_t **get_length** () const

Public Attributes

- std::string **debug_marg_name**
- std::unique_ptr< long double [ ]> **marginal_array_smart_p**

5.34.1 Detailed Description

Encapsulates the marginal probabilities/posterior frequency for each recombination event's realization.

Author

Q.Marcou

Version

1.0

Model_Marginals central component is a giant array containing all the marginal probabilities (or posterior frequencies during the inference) for each event realization. The class supplies various methods to navigate into this array and as well methods handling normalization and other various transformation.

5.34.2 Member Function Documentation

5.34.2.1 set_realization_proba()

```
void Model_marginals::set_realization_proba (
            std::string ,
            std::shared_ptr< const Rec_Event > ,
            double ,
            const Model_Parms & )
```

Sets the realization probability to the given value Note that the value will be set for all conditional dependences

//TODO recode this in order to be able to set several realizations probas at the same time //FIXME if the supplied new value is 1.0 there will be a zero division issue (this could be fixed by fixing all others to 0 instead of trying to set the supplied one to 1)

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

- Model_marginals.h
- Model_marginals.cpp

5.35 Model_Parms Class Reference

Implements IGoR's Bayesian Network structure.

```
#include <Model_Parms.h>
```

Public Member Functions

- **Model_Parms** (std::list< std::shared_ptr< Rec_Event >> event_list)
- **Model_Parms** (const Model_Parms &)
- std::list< std::shared_ptr< Rec_Event > > **get_children** (Rec_Event ∗) const
- std::list< std::shared_ptr< Rec_Event > > **get_children** (std::shared_ptr< Rec_Event >) const
- std::list< std::shared_ptr< Rec_Event > > **get_children** (Rec_Event_name) const
- std::list< std::shared_ptr< Rec_Event > > **get_parents** (Rec_Event ∗) const
- std::list< std::shared_ptr< Rec_Event > > **get_parents** (std::shared_ptr< Rec_Event >) const
- std::list< std::shared_ptr< Rec_Event > > **get_parents** (Rec_Event_name) const
- std::list< std::shared_ptr< Rec_Event > > **get_ancestors** (Rec_Event ∗) const
- std::list< std::shared_ptr< Rec_Event > > **get_ancestors** (std::shared_ptr< Rec_Event >) const
- std::list< std::shared_ptr< Rec_Event > > get_ancestors (Rec_Event_name) const

  Get all ancestors of the supplied event.
- bool **add_edge** (Rec_Event ∗, Rec_Event ∗)
- bool **add_edge** (std::shared_ptr< Rec_Event >, std::shared_ptr< Rec_Event >)
- bool **add_edge** (Rec_Event_name, Rec_Event_name)
- bool **remove_edge** (Rec_Event ∗, Rec_Event ∗)
- bool **remove_edge** (std::shared_ptr< Rec_Event >, std::shared_ptr< Rec_Event >)
- bool **remove_edge** (Rec_Event_name, Rec_Event_name)
- void **invert_edge** (Rec_Event ∗, Rec_Event ∗)
- void **invert_edge** (std::shared_ptr< Rec_Event >, std::shared_ptr< Rec_Event >)
- void **invert_edge** (Rec_Event_name, Rec_Event_name)
- bool **has_edge** (Rec_Event ∗, Rec_Event ∗) const
- bool **has_edge** (std::shared_ptr< Rec_Event >, std::shared_ptr< Rec_Event >) const
- bool **has_edge** (Rec_Event_name, Rec_Event_name) const
- std::list< std::shared_ptr< Rec_Event > > **get_roots** () const
- bool **add_event** (std::shared_ptr< Rec_Event >)
- bool **add_event** (Rec_Event ∗)
- std::queue< std::shared_ptr< Rec_Event > > **get_model_queue** () const
- std::shared_ptr< Rec_Event > **get_event_pointer** (const Rec_Event_name &) const
- std::shared_ptr< Rec_Event > **get_event_pointer** (const std::string &, bool by_nickname) const
- void **update_edge_event_name** (Rec_Event_name, Rec_Event_name)
- void **write_model_parms** (std::string)
- void **read_model_parms** (std::string)
- void **set_fixed_all_events** (bool)
- std::list< std::shared_ptr< Rec_Event > > **get_event_list** () const
- std::unordered_map< Rec_Event_name, Adjacency_list > **get_edges** () const
- const std::unordered_map< std::tuple< Event_type, Gene_class, Seq_side >, std::shared_ptr< Rec_Event > > **get_events_map** () const
- std::unordered_map< std::tuple< Event_type, Gene_class, Seq_side >, std::shared_ptr< Rec_Event > > **get_events_map** ()
- void **set_error_ratep** (Error_rate ∗Er_r)
- void **set_error_ratep** (std::shared_ptr< Error_rate > Er_r)
- std::shared_ptr< Error_rate > **get_err_rate_p** ()

5.35.1 Detailed Description

Implements IGoR's Bayesian Network structure.

Author

Q.Marcou

Version

1.0

Implements IGoR's Bayesian Network structure through an acyclic directed graph. Together with the recombination model topology it also contains the error model. This class implements various methods to extract information from the graph structure such as the order in which RecEvents must be processed provided the topological constraints.

5.35.2 Member Function Documentation

5.35.2.1 get_ancestors()

```
list< shared_ptr< Rec_Event > > Model_Parms::get_ancestors (
            Rec_Event_name event_name ) const
```

Get all ancestors of the supplied event.

Author

Q.Marcou

Version

1.0

Parameters

in	*event_name*	The event whom we seek the ancestors

Returns

The list of the event ancestors

Get all ancestors of the supplied event. Ancestors are all events higher than the considered event in the genealogy and with link of any degree to it

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

- Model_Parms.h
- Model_Parms.cpp

5.36 null_delete< T > Struct Template Reference

Declare a null_delete function.

```
#include <Utils.h>
```

Public Member Functions

- void **operator()** (T ∗&)

5.36.1 Detailed Description

template<class T>
struct null_delete< T >

Declare a null_delete function.

Author

> Q.Marcou This function is not performing any task, it's purpose is to supply a "null_delete" function to prevent shared pointer objects created when passing Rec_Event or Error_rate objects pointers to model_parms to be destroyed when the model_parms object is destroyed itself and the rec_event and error_rate objects might still be of used (and if not prevent from a segfault error by trying to delete them twice)

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

- Utils.h

5.37 offset_comp Struct Reference

Public Member Functions

- bool **operator()** (const std::pair< std::shared_ptr< const Rec_Event >, int > pair_1, const std::pair< std↩ ::shared_ptr< const Rec_Event >, int > pair_2)

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

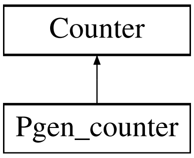
- Model_marginals.h

5.38 Pgen_counter Class Reference

Estimates sequences generation probability.

```
#include <Pgencounter.h>
```

Inheritance diagram for Pgen_counter:



Public Member Functions

- **Pgen_counter** (std::string)
- **Pgen_counter** (std::string, bool, bool do_output_sequences=false)
- std::string **type** () const
- void **initialize_counter** (const Model_Parms &, const Model_marginals &)
- void **count_scenario** (long double, double, const std::string &, Seq_type_str_p_map &, const Seq_offsets↵_map &, const std::unordered_map< std::tuple< Event_type, Gene_class, Seq_side >, std::shared_ptr< Rec_Event >> &, Mismatch_vectors_map &)
- void **dump_sequence_data** (int, int)
- void **add_checked** (std::shared_ptr< Counter >)
- std::shared_ptr< Counter > **copy** () const

Additional Inherited Members

5.38.1 Detailed Description

Estimates sequences generation probability.

Author

Q.Marcou

Version

1.0

This Counter implements an estimator for the generation probability of evaluated sequences. Alternatively the counter can record the probability of generation of putative ancestor (unmutated/error free) sequences and their associated posterior probability.

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

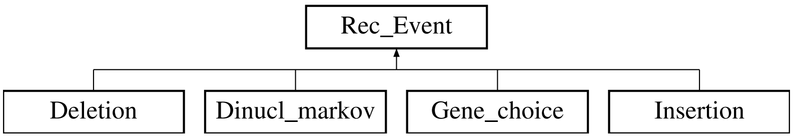
- Pgencounter.h
- Pgencounter.cpp

5.39 Rec_Event Class Reference

Recombination event class (IGoR's graph nodes)

```
#include <Rec_Event.h>
```

Inheritance diagram for Rec_Event:



Public Member Functions

- **Rec_Event** (Gene_class, Seq_side)
- **Rec_Event** (Gene_class, Seq_side, std::unordered_map< std::string, Event_realization > &)
- virtual std::shared_ptr< Rec_Event > **copy** ()=0
- virtual int **size** () const
- virtual void iterate (double &, Downstream_scenario_proba_bound_map &, const std::string &, const Int_Str &, Index_map &, const std::unordered_map< Rec_Event_name, std::vector< std::pair< std::shared_ptr< const Rec_Event >, int >>> &, std::shared_ptr< Next_event_ptr > &, Marginal_array_p &, const Marginal_array_p &, const std::unordered_map< Gene_class, std::vector< Alignment_data >> &, Seq_type_str_p_map &, Seq_offsets_map &, std::shared_ptr< Error_rate > &, std::map< size_t, std::shared_ptr< Counter >> &, const std::unordered_map< std::tuple< Event_type, Gene_class, Seq_side >, std::shared_ptr< Rec_Event >> &, Safety_bool_map &, Mismatch_vectors_map &, double &, double &)=0

  Evaluate all Event_Realization of a RecEvent for a given sequence.
- bool **set_priority** (int)
- const Gene_class **get_class** () const
- const Seq_side **get_side** () const
- const std::unordered_map< std::string, Event_realization > **get_realizations_map** () const
- const int **get_priority** () const
- const Rec_Event_name **get_name** () const
- const std::string **get_nickname** () const
- void **set_nickname** (std::string name)
- Event_type **get_type** () const
- int **get_len_max** () const
- int **get_len_min** () const
- bool **operator==** (const Rec_Event &) const
- void **update_event_name** ()
- virtual std::queue< int > **draw_random_realization** (const Marginal_array_p &, std::unordered_map< Rec_Event_name, int > &, const std::unordered_map< Rec_Event_name, std::vector< std::pair< std::shared_ptr< const Rec_Event >, int >>> &, std::unordered_map< Seq_type, std::string > &, std::mt19937_64 &) const =0
- virtual void **write2txt** (std::ofstream &)=0
- virtual void **ind_normalize** (Marginal_array_p &, size_t) const
- virtual void **initialize_event** (std::unordered_set< Rec_Event_name > &, const std::unordered_map< std::tuple< Event_type, Gene_class, Seq_side >, std::shared_ptr< Rec_Event >> &, const std::unordered_map< Rec_Event_name, std::vector< std::pair< std::shared_ptr< const Rec_Event >, int >>> &, Downstream_scenario_proba_bound_map &, Seq_type_str_p_map &, Safety_bool_map &, std::shared_ptr< Error_rate >, Mismatch_vectors_map &, Seq_offsets_map &, Index_map &)
- virtual void **initialize_crude_scenario_proba_bound** (double &, std::forward_list< double ∗> &, const std::unordered_map< std::tuple< Event_type, Gene_class, Seq_side >, std::shared_ptr< Rec_Event >> &)
- virtual void **add_to_marginals** (long double, Marginal_array_p &) const =0
- virtual void **set_crude_upper_bound_proba** (size_t, size_t, Marginal_array_p &)
- void **set_upper_bound_proba** (double)
- double **get_upper_bound_proba** () const
- virtual void update_event_internal_probas (const Marginal_array_p &, const std::unordered_map< Rec_Event_name, int > &)
- void **set_event_identifier** (size_t)
- int **get_event_identifier** () const
- void **set_event_marginal_size** (size_t ev_size)
- bool **is_updated** () const
- void **fix** (bool fix_status)
- bool **is_fixed** () const
- void **set_viterbi_run** (bool viterbi_like)
- virtual double ∗ **get_updated_ptr** ()
- void **compute_crude_upper_bound_scenario_proba** (double &)

- const std::vector< int > & **get_current_realizations_index_vec** () const
- virtual bool **has_effect_on** (Seq_type) const =0
- void **iterate_initialize_Len_proba_wrap_up** (Seq_type considered_junction, std::map< int, double > &length_best_proba_map, std::queue< std::shared_ptr< Rec_Event >> model_queue, double scenario_proba, const Marginal_array_p &model_parameters_point, Index_map &base_index_map, Seq_type_str_p_map &constructed_sequences, int seq_len) const
- virtual void **iterate_initialize_Len_proba** (Seq_type considered_junction, std::map< int, double > &length_best_proba_map, std::queue< std::shared_ptr< Rec_Event >> &model_queue, double &scenario_proba, const Marginal_array_p &model_parameters_point, Index_map &base_index_map, Seq_type_str_p_map &constructed_sequences, int &seq_len) const =0
- void **iterate_initialize_Len_proba** (Seq_type considered_junction, std::map< int, double > &length_best_proba_map, std::queue< std::shared_ptr< Rec_Event >> &model_queue, double &scenario_proba, const Marginal_array_p &model_parameters_point, Index_map &base_index_map, Seq_type_str_p_map &constructed_sequences) const
- virtual void **initialize_Len_proba_bound** (std::queue< std::shared_ptr< Rec_Event >> &model_queue, const Marginal_array_p &model_parameters_point, Index_map &base_index_map)=0

Protected Member Functions

- int **compare_sequences** (std::string, std::string)
- void **add_realization** (const Event_realization &)
- void **iterate_wrap_up** (double &scenario_proba, Downstream_scenario_proba_bound_map &downstream_proba_map, const std::string &sequence, const Int_Str &int_sequence, Index_map &index_map, const std::unordered_map< Rec_Event_name, std::vector< std::pair< std::shared_ptr< const Rec_Event >, int >>> &offset_map, std::shared_ptr< Next_event_ptr > &next_event_ptr_arr, Marginal_array_p &updated_marginal_array_p, const Marginal_array_p &model_parameters_point, const std::unordered_map< Gene_class, std::vector< Alignment_data >> &allowed_realizations, Seq_type_str_p_map &constructed_sequences, Seq_offsets_map &seq_offsets, std::shared_ptr< Error_rate > &error_rate_p, std::map< size_t, std::shared_ptr< Counter >> &counters_list, const std::unordered_map< std::tuple< Event_type, Gene_class, Seq_side >, std::shared_ptr< Rec_Event >> &events_map, Safety_bool_map &safety_set, Mismatch_vectors_map &mismatches_lists, double &seq_max_prob_scenario, double &proba_threshold_factor)

Protected Attributes

- std::unordered_map< std::string, Event_realization > **event_realizations**
- int **priority**
- Gene_class **event_class**
- Seq_side **event_side**
- Rec_Event_name **name**
- std::string **nickname**
- int **len_min**
- int **len_max**
- Event_type **type**
- int **event_index**
- std::forward_list< std::tuple< int, int, int > > **memory_and_offsets**
- bool **updated**
- bool **viterbi_run**
- bool **initialized**
- size_t **event_marginal_size**
- bool **fixed**
- double **event_upper_bound_proba**
- double **scenario_downstream_upper_bound_proba**
- double **scenario_upper_bound_proba**
- std::forward_list< double ∗ > **updated_proba_bounds_list**
- std::vector< int > **current_realizations_index_vec**
- const int ∗ **current_realization_index**
- int **current_downstream_proba_memory_layers** [6]

5.39.1 Detailed Description

Recombination event class (IGoR's graph nodes)

Author

Q.Marcou

Version

1.0

This class implements the recombination event object. Rec_Events are the nodes in IGoR's Bayesian Network structure. This is a purely abstract class and cannot be instanciated as is, only classes deriving from it and implementing the purely abstract methods can be.

Rec_Events contain the different Event_realization associated to it in a hashmap.

The RecEvents design is key to the way IGoR explore all possible scenarios (through the iterate method) and generate sequences (through the draw_random_realization)

5.39.2 Member Function Documentation

5.39.2.1 iterate()

```
virtual void Rec_Event::iterate (
             double & ,
             Downstream_scenario_proba_bound_map & ,
             const std::string & ,
             const Int_Str & ,
             Index_map & ,
             const std::unordered_map< Rec_Event_name, std::vector< std::pair< std::shared_↩
ptr< const Rec_Event >, int >>> & ,
             std::shared_ptr< Next_event_ptr > & ,
             Marginal_array_p & ,
             const Marginal_array_p & ,
             const std::unordered_map< Gene_class, std::vector< Alignment_data >> & ,
             Seq_type_str_p_map & ,
             Seq_offsets_map & ,
             std::shared_ptr< Error_rate > & ,
             std::map< size_t, std::shared_ptr< Counter >> & ,
             const std::unordered_map< std::tuple< Event_type, Gene_class, Seq_side >, std↩
::shared_ptr< Rec_Event >> & ,
             Safety_bool_map & ,
             Mismatch_vectors_map & ,
             double & ,
             double & )  [pure virtual]
```

Evaluate all Event_Realization of a RecEvent for a given sequence.

Author

Q.Marcou

Version

1.0

Parameters

`in,out`	*scenario_proba*	Probability of the currently explored (incomplete) scenario
`in,out`	*downstream_proba_map*	
`in`	*sequence*	The studied sequence in nucleotide code
`in`	*int_sequence*	The studied sequence in integer code
`in,out`	*base_index_map*	Dynamic map recording where probabilities should be read on the marginals.
`in`	*offset_map*	Tells the event by how much indices from the children events should be modified
`in`	*next_event_ptr_arr*	Indicates the next event to call iterate on
`in`	*updated_marginals_point*	Summary marginals on which complete scenario posteriors are recorded
`in`	*model_parameters_point*	Current recombination probability distribution
`in`	*allowed_realizations*	The set of genomic templates alignment
`in,out`	*constructed_sequences*	Map containing the (incomplete) scenario's resulting sequence
`in,out`	*seq_offsets*	Map containing the 3' and 5' offsets of each scenario sequence piece
`in`	*error_rate_p*	Pointer to the error model object
`in`	*counters_list*	The list of Counter to be counted
`in`	*events_map*	A map containing all events contained in the Model_parms, accessible through their type, gene class and side.
`in,out`	*safety_set*	A map indicating whether checks on offsets overlap should be performed
`in,out`	*mismatches_lists*	A map containing the (incomplete) scenario mismatches
`in`	*seq_max_prob_scenario*	Most likely scenario's probability for the considered sequence
`in`	*proba_threshold_factor*	Threshold on probability ratio between most likely scenario and explored scenario

Returns

void

The iterate method is the heart of IGoR's scenario exploration. Model_Parms define an order in which the RecEvent should be processed. Upon call of iterate all EventRealization of the RecEvent are assessed and for each possible realization the iterate method is called recursively for the next event. Inside the iterate method a filtering on too improbable realizations is performed (tree prunning) using the downstream_proba_map.

The index map is used to read off the probability of the current RecEvent EventRealization at the correct location given the events parent's realizations. It is further modified to take into account the current event's realization when its children realization probabilities will be read.

Implemented in Deletion, Dinucl_markov, Gene_choice, and Insertion.

5.39.2.2 update_event_internal_probas()

```
void Rec_Event::update_event_internal_probas (
            const Marginal_array_p & ,
            const std::unordered_map< Rec_Event_name, int > & )  [virtual]
```

Does nothing since in general events will not need to perform any operation on the marginal probabilities

Reimplemented in Dinucl_markov.

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

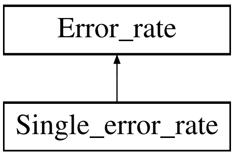
- Rec_Event.h
- Rec_Event.cpp

5.40 Single_error_rate Class Reference

Independent single nucleotide error model.

```
#include <Singleerrorrate.h>
```

Inheritance diagram for Single_error_rate:



Public Member Functions

- **Single_error_rate** (double)
- double **compare_sequences_error_prob** (double, const std::string &, Seq_type_str_p_map &, const Seq_offsets_map &, const std::unordered_map< std::tuple< Event_type, Gene_class, Seq_side >, std↩ ::shared_ptr< Rec_Event >> &, Mismatch_vectors_map &, double &, double &)
- void **update** ()
- void **add_to_norm_counter** ()
- void **clean_seq_counters** ()
- Single_error_rate **operator+** (Single_error_rate)
- Single_error_rate & **operator+=** (Single_error_rate)
- void **write2txt** (std::ofstream &)
- std::shared_ptr< Error_rate > **copy** () const
- std::string **type** () const
- Error_rate ∗ **add_checked** (Error_rate ∗)
- const double & **get_err_rate_upper_bound** (size_t, size_t)
- void **build_upper_bound_matrix** (size_t, size_t)
- int **get_number_non_zero_likelihood_seqs** () const
- std::queue< int > **generate_errors** (std::string &, std::mt19937_64 &) const

Additional Inherited Members

5.40.1 Detailed Description

Independent single nucleotide error model.

Author

Q.Marcou

Version

1.0

Simplest instance of the ErrorRate family. Models errors/mutations as a Bernouilli process with a global rate independent of position and context.

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

- Singleerrorrate.h
- Singleerrorrate.cpp

Index