

API Reference

Generated with ROBODoc Version 4.99.43 (Jan 26 2019)

January 27, 2019

Contents

1 Kohonen/distance_base_utilities	5
1.1 distance_base_utilities/distance_base	6
1.2 distance_base_utilities/distance_function1	7
2 Kohonen/euclidean_distance_utilities	8
2.1 euclidean_distance_utilities/calculate_euclidean_distance	9
2.2 euclidean_distance_utilities/euclidean_distance	10
3 Kohonen/factory_distance_utilities	11
3.1 factory_distance_utilities/factory_distance	12
4 Kohonen/influence_function_utilities	13
4.1 influence_function_utilities/calculate_influence_function	14
4.2 influence_function_utilities/influence_function	15
4.3 influence_function_utilities/sgn	16
5 Kohonen/kohonen_map_base_utilities	17
5.1 kohonen_map_base_utilities/kohonen_map_base	18
5.2 kohonen_map_base_utilities/kohonen_map_constructor	19
5.3 kohonen_map_base_utilities/kohonen_map_destructor	20
5.4 kohonen_map_base_utilities/kohonen_map_function1	21
5.5 kohonen_map_base_utilities/kohonen_map_function2	22
6 Kohonen/kohonen_pattern_utilities	23
6.1 kohonen_pattern_utilities/kohonen_pattern	24
6.2 kohonen_pattern_utilities/kohonen_pattern_accessor	25
6.3 kohonen_pattern_utilities/kohonen_pattern_create	26
6.4 kohonen_pattern_utilities/kohonen_pattern_destroy	27
6.5 kohonen_pattern_utilities/kohonen_pattern_mutator	28
6.6 kohonen_pattern_utilities/kohonen_pattern_print	29
7 Kohonen/kohonen_prototype_utilities	30
7.1 kohonen_prototype_utilities/kohonen_prototype	31
7.2 kohonen_prototype_utilities/kohonen_prototype_accessor	32
7.3 kohonen_prototype_utilities/kohonen_prototype_constructor	33
7.4 kohonen_prototype_utilities/kohonen_prototype_destructor	34

7.5	kohonen_prototype_utilities/kohonen_prototype_mutator	35
7.6	kohonen_prototype_utilities/kohonen_prototype_print	36
8	Kohonen/self_organized_map_utilities	37
8.1	self_organized_map_utilities/calculate_coordinates	38
8.2	self_organized_map_utilities/calculate_distance_between_prototypes	39
8.3	self_organized_map_utilities/calculate_distance_matrix	40
8.4	self_organized_map_utilities/calculate_sigma	41
8.5	self_organized_map_utilities/calculate_u_matrix	42
8.6	self_organized_map_utilities/create	43
8.7	self_organized_map_utilities/destroy	44
8.8	self_organized_map_utilities/external_predict_map	45
8.9	self_organized_map_utilities/external_train_map	46
8.10	self_organized_map_utilities/find_best_match_unit	47
8.11	self_organized_map_utilities/find_bmu_grid	48
8.12	self_organized_map_utilities/get_count	49
8.13	self_organized_map_utilities/get_prototypes	50
8.14	self_organized_map_utilities/index2position	51
8.15	self_organized_map_utilities/position2index	52
8.16	self_organized_map_utilities/predict	53
8.17	self_organized_map_utilities/print	54
8.18	self_organized_map_utilities/query_som	55
8.19	self_organized_map_utilities/read_som	56
8.20	self_organized_map_utilities/self_organized_map	57
8.21	self_organized_map_utilities/train_som_data	59
8.22	self_organized_map_utilities/update_weights	60
9	Kohonen/two_level_self_organized_map_utilities	61
9.1	two_level_self_organized_map_utilities/calculate_cluster_measures	62
9.2	two_level_self_organized_map_utilities/calculate_coordinates	63
9.3	two_level_self_organized_map_utilities/calculate_distance_between_prototypes	64
9.4	two_level_self_organized_map_utilities/calculate_distance_matrix	65
9.5	two_level_self_organized_map_utilities/calculate_sum2_clusters_grid	66
9.6	two_level_self_organized_map_utilities/create	67
9.7	two_level_self_organized_map_utilities/destroy	68
9.8	two_level_self_organized_map_utilities/evaluate_2lsom	69

9.9	two_level_self_organized_map_utilities/get_cluster_samples	70
9.10	two_level_self_organized_map_utilities/index2position	71
9.11	two_level_self_organized_map_utilities/position2index	72
9.12	two_level_self_organized_map_utilities/predict_2lsom	73
9.13	two_level_self_organized_map_utilities/print_2lsom	74
9.14	two_level_self_organized_map_utilities/query_som	75
9.15	two_level_self_organized_map_utilities/set_cluster_layer	76
9.16	two_level_self_organized_map_utilities/set_parameters	77
9.17	two_level_self_organized_map_utilities/train_2lsom	78
9.18	two_level_self_organized_map_utilities/train_cluster_layer	79
9.19	two_level_self_organized_map_utilities/train_grid_layer	80
9.20	two_level_self_organized_map_utilities/two_level_self_organized_map	81
9.20.1	two_level_self_organized_map/assign_input_to_clusters	83
9.20.2	two_level_self_organized_map/calculate_u_matrix	84
9.20.3	two_level_self_organized_map/find_best_match_unit	85
9.20.4	two_level_self_organized_map/read_som_layer	86
9.21	two_level_self_organized_map_utilities/update_weights	87
10	ROBODoc/ROBODoc Cascading Style Sheet	88

1 Kohonen/distance_base_utilities

[Modules]

NAME

MODULE distance_base_utilities

PURPOSE

This module defines an abstract class for distance

AUTHOR

Oscar Garcia-Cabrejo

NOTES

MODIFICATION HISTORY

1.1 distance_base_utilities/distance_base

[distance_base_utilities] [Classes]

NAME

distance_base

PURPOSE

Abstract Class to represent an abstract function to calculate distance

METHODS

```
procedure(distance_function1),deferred :: calculate
end type distance_base
```

1.2 distance_base_utilities/distance_function1

[distance_base_utilities] [Functions]

NAME

distance_function1

PURPOSE

Template for Function to calculate distance

SYNOPSIS

```
!=====
function distance_function1(distance,vector1,vector2) result(d)
!=====
import :: distance_base
class(distance_base) :: distance
real(kind=8),dimension(:,:),intent(inout) :: vector1,vector2
real(kind=8) :: d
```

2 Kohonen/euclidean_distance_utilities

[Modules]

NAME

MODULE euclidean_distance_utilities

PURPOSE

This module defines a class to calculate the Euclidean distance between kohonen prototypes

AUTHOR

Oscar Garcia-Cabrejo

NOTES

MODIFICATION HISTORY

2.1 euclidean_distance_utilities/calculate_euclidean_distance*[euclidean_distance_utilities] [Functions]***NAME**`calculate_euclidean_distance`**PURPOSE**

Function to calculate euclidean distance between vectors

SYNOPSIS

```

=====
function calculate_euclidean_distance(distance,vector1,vector2) result(d)
=====
class(euclidean_distance) :: distance
real(kind=8),dimension(:,:),intent(inout) :: vector1,vector2
real(kind=8) :: d

```

2.2 euclidean_distance_utilities/euclidean_distance

[euclidean_distance_utilities] [Classes]

NAME

euclidean_distance

PURPOSE

Class to calculate the euclidean distance

METHODS

procedure,public :: calculate => calculate_euclidean_distance

3 Kohonen/factory_distance_utilities

[Modules]

NAME

MODULE factory_distance_utilities

PURPOSE

This module defines a factory to create distance objects

AUTHOR

Oscar Garcia-Cabrejo

NOTES

MODIFICATION HISTORY

3.1 factory_distance_utilities/factory_distance

[factory_distance_utilities] [Classes]

NAME

factory_distance

PURPOSE

Class to represent a distance factory

METHODS

```
contains
  procedure,public :: create_distance
end type factory_distance
```

4 Kohonen/influence_function_utilities

[Modules]

NAME

MODULE influence_function_utilities

PURPOSE

This module defines a class to calculate the influence functions required in Robust SOM

AUTHOR

Oscar Garcia-Cabrejo

4.1 influence_function_utilities/calculate_influence_function

[influence_function_utilities] [Functions]

NAME

calculate_influence_function

PURPOSE

Calculates the influence function

SYNOPSIS

```
!=====
function calculate_influence_function(my_function,type_,r) result(v)
!=====
class(influence_function) :: my_function
character(len=*) :: type_
real(kind=8),intent(inout) :: r
real(kind=8) :: v
```

4.2 influence_function_utilities/influence_function

[influence_function_utilities] [Classes]

NAME

influence_function

PURPOSE

Class that represents an influence function

METHODS

contains

procedure,public :: calculate => calculate_influence_function

4.3 influence_function_utilities/sgn

[*influence_function_utilities*] [*Functions*]

NAME

sgn

PURPOSE

Sign function

SYNOPSIS

```
!=====
function sgn(x) result(v)
!=====
real(kind=8),intent(inout) :: x
real(kind=8) :: v
```


5 Kohonen/kohonen_map_base_utilities

[Modules]

NAME

MODULE kohonen_map_base_utilities

PURPOSE

This module defines an abstract class for kohonen maps

AUTHOR

Oscar Garcia-Cabrejo

NOTES

MODIFICATION HISTORY

5.1 kohonen_map_base_utilities/kohonen_map_base

[*kohonen_map_base_utilities*] [*Classes*]

NAME

kohonen_map_base

PURPOSE

Abstract Class to represent a template for a kohonen map

METHODS

```
procedure(kohonen_map_constructor),public,deferred :: create
procedure(kohonen_map_destructor),public,deferred :: destroy
procedure(kohonen_map_function1),public,deferred :: train
procedure(kohonen_map_function2),public,deferred :: predict
end type kohonen_map_base
```

5.2 kohonen_map_base_utilities/kohonen_map_constructor

[*kohonen_map_base_utilities*] [*Functions*]

NAME

kohonen_map_constructor

PURPOSE

Template function for the constructor of a kohonen map

SYNOPSIS

```
!=====
  subroutine kohonen_map_constructor(kohonen_map,training_parameters)
!=====
  import :: kohonen_map_base
  import :: kohonen_layer_parameters
  class(kohonen_map_base) :: kohonen_map
  type(kohonen_layer_parameters),dimension(:) :: training_parameters
```

5.3 kohonen_map_base_utilities/kohonen_map_destructor

[*kohonen_map_base_utilities*] [*Functions*]

NAME

kohonen_map_destructor

PURPOSE

Template function for the destructor of a kohonen map

SYNOPSIS

```
!=====
subroutine kohonen_map_destructor(kohonen_map)
!=====
import :: kohonen_map_base
class(kohonen_map_base) :: kohonen_map
```

5.4 kohonen_map_base_utilities/kohonen_map_function1

[*kohonen_map_base_utilities*] [*Functions*]

NAME

kohonen_map_function1

PURPOSE

Template function for the training function of a kohonen map

SYNOPSIS

```
!=====
  subroutine kohonen_map_function1(kohonen_map,input_data)
!=====
  import :: kohonen_map_base
  import :: kohonen_pattern
  class(kohonen_map_base) :: kohonen_map
  type(kohonen_pattern),dimension(:),intent(inout) :: input_data
!   real(kind=8),dimension(:,:),intent(inout),optional :: distances
```

5.5 kohonen_map_base_utilities/kohonen_map_function2[*kohonen_map_base_utilities*] [*Functions*]**NAME**

kohonen_map_function2

PURPOSE

Template function for the prediction function of a kohonen map

SYNOPSIS

```

=====
subroutine kohonen_map_function2(kohonen_map,input_data,map_output)
=====
import :: kohonen_map_base
import :: kohonen_pattern
class(kohonen_map_base) :: kohonen_map
type(kohonen_pattern),dimension(:),intent(inout) :: input_data
integer,dimension(:,:),intent(out) :: map_output

```

6 Kohonen/kohonen_pattern_utilities

[Modules]

NAME

MODULE kohonen_pattern_utilities

PURPOSE

This module defines a class for kohonen patterns (input data)

AUTHOR

Oscar Garcia-Cabrejo

NOTES

MODIFICATION HISTORY

6.1 kohonen_pattern_utilities/kohonen_pattern*[kohonen_pattern_utilities] [Classes]***NAME**`kohonen_pattern`**PURPOSE**`Class to represent a container for input data to a kohonen map`**ATTRIBUTES**

```
private
  type(kohonen_prototype) :: pattern
  character(len=50) :: pattern_name
contains
```

METHODS

```
procedure,public :: create => kohonen_pattern_create
procedure,public :: destroy => kohonen_pattern_destroy
procedure,public :: get => kohonen_pattern_accessor
procedure,public :: set => kohonen_pattern_mutator
procedure,public :: print => kohonen_pattern_print
procedure,public :: get_nrow => kohonen_pattern_nrow
procedure,public :: get_ncol => kohonen_pattern_ncol
!
end type kohonen_pattern
```


6.2 kohonen_pattern_utilities/kohonen_pattern_accessor

[*kohonen_pattern_utilities*] [*Functions*]

NAME

kohonen_pattern_accessor

PURPOSE

Kohonen pattern accessor

SYNOPSIS

```
!=====
subroutine kohonen_pattern_accessor(current_pattern,pattern_value)
!=====
class(kohonen_pattern) :: current_pattern
type(kohonen_prototype),intent(inout) :: pattern_value
```

6.3 kohonen_pattern_utilities/kohonen_pattern_create*[kohonen_pattern_utilities] [Functions]***NAME**`kohonen_pattern_create`**PURPOSE**`Kohonen pattern constructor`**SYNOPSIS**

```

=====
subroutine kohonen_pattern_create(current_pattern,input,name)
=====
class(kohonen_pattern) :: current_pattern
real(kind=8),dimension(:,:),intent(inout) :: input
character(len=*),optional :: name

```

6.4 kohonen_pattern_utilities/kohonen_pattern_destroy

[*kohonen_pattern_utilities*] [*Functions*]

NAME

kohonen_pattern_destroy

PURPOSE

Kohonen pattern destructor

SYNOPSIS

```
!=====
subroutine kohonen_pattern_destroy(current_pattern)
!=====
class(kohonen_pattern) :: current_pattern
```

6.5 kohonen_pattern_utilities/kohonen_pattern_mutator*[kohonen_pattern_utilities] [Functions]***NAME**`kohonen_pattern_mutator`**PURPOSE**`Kohonen pattern mutator`**SYNOPSIS**

```

=====
subroutine kohonen_pattern_mutator(current_pattern,pattern_value)
=====
class(kohonen_pattern) :: current_pattern
type(kohonen_prototype),intent(inout) :: pattern_value

```

6.6 kohonen_pattern_utilities/kohonen_pattern_print

[*kohonen_pattern_utilities*] [*Functions*]

NAME

kohonen_pattern_print

PURPOSE

Function to print a Kohonen pattern

SYNOPSIS

```
!=====
subroutine kohonen_pattern_print(current_pattern,unit_)
!=====
class(kohonen_pattern) :: current_pattern
integer,intent(inout),optional :: unit_
```

7 Kohonen/kohonen_prototype_utilities

[Modules]

NAME

MODULE kohonen_pattern_utilities

PURPOSE

This module defines a class for kohonen prototype (units inside kohonen layers)

AUTHOR

Oscar Garcia-Cabrejo

NOTES

MODIFICATION HISTORY

7.1 kohonen_prototype_utilities/kohonen_prototype

[*kohonen_prototype_utilities*] [*Classes*]

NAME

kohonen_prototype

PURPOSE

Class to store a prototype inside a Kohonen map

METHODS

```
contains
!
  procedure :: create => kohonen_prototype_constructor
  procedure :: destroy => kohonen_prototype_destructor
  procedure :: get_prototype => kohonen_prototype_accessor
  procedure :: set_prototype => kohonen_prototype_mutator
  procedure :: print => kohonen_prototype_print
  procedure :: distance => kohonen_prototype_distance
  procedure :: get_nrow => kohonen_prototype_nrow
  procedure :: get_ncol => kohonen_prototype_ncol
!
end type kohonen_prototype
```

7.2 kohonen_prototype_utilities/kohonen_prototype_accessor

[*kohonen_prototype_utilities*] [*Functions*]

NAME

kohonen_prototype_accessor

PURPOSE

Accessor

SYNOPSIS

```
!=====
subroutine kohonen_prototype_accessor(prototype,d)
!=====
class(kohonen_prototype) :: prototype
real(kind=8),dimension(prototype%number_rows,prototype%number_columns) :: d
```


7.3 kohonen_prototype_utilities/kohonen_prototype_constructor

[*kohonen_prototype_utilities*] [*Functions*]

NAME

kohonen_prototype_constructor

PURPOSE

Constructor

SYNOPSIS

```
!=====
subroutine kohonen_prototype_constructor(prototype,input_data)
!=====
class(kohonen_prototype) :: prototype
real(kind=8),dimension(:,:) :: input_data
```

7.4 kohonen_prototype_utilities/kohonen_prototype_destructor

[*kohonen_prototype_utilities*] [*Functions*]

NAME

kohonen_prototype_destructor

PURPOSE

Destructor

SYNOPSIS

```
!=====
subroutine kohonen_prototype_destructor(prototype)
!=====
class(kohonen_prototype),intent(inout) :: prototype
```

7.5 kohonen_prototype_utilities/kohonen_prototype_mutator

[*kohonen_prototype_utilities*] [*Functions*]

NAME

kohonen_prototype_mutator

PURPOSE

Mutator

SYNOPSIS

```
!=====
subroutine kohonen_prototype_mutator(prototype,new_data)
!=====
class(kohonen_prototype) :: prototype
real(kind=8),dimension(:,:),intent(inout) :: new_data
```

7.6 kohonen_prototype_utilities/kohonen_prototype_print

[*kohonen_prototype_utilities*] [*Functions*]

NAME

kohonen_prototype_print

PURPOSE

Function to print a kohonen prototype

SYNOPSIS

```
!=====
subroutine kohonen_prototype_print(prototype,unit_)
!=====
class(kohonen_prototype) :: prototype
integer,intent(inout),optional :: unit_
```

8 Kohonen/self_organized_map_utilities

[Modules]

NAME

MODULE self_organized_map_utilities

PURPOSE

This module defines a class for simple self_organized_map (one kohonen layer)

AUTHOR

Oscar Garcia-Cabrejo

NOTES

MODIFICATION HISTORY

8.1 self_organized_map_utilities/calculate_coordinates*[self_organized_map_utilities] [Functions]***NAME**`calculate_coordinates`**PURPOSE**

Subroutine to calculate the coordinates of the units inside a kohonen layer

SYNOPSIS

```

!=====
  subroutine calculate_coordinates(current_index,ix,iy,iz,nx,ny,nz,coordinates,node_type)
!=====
  integer,intent(inout) :: current_index,ix,iy,iz,nx,ny,nz
  real(kind=8),dimension(:,:),intent(out) :: coordinates
  character(len=*),intent(in) :: node_type

```

8.2 self_organized_map_utilities/calculate_distance_between_prototypes

[self_organized_map_utilities] [Functions]

NAME

calculate_distance_between_prototypes

PURPOSE

Subroutine to calculate the distance between the prototypes

SYNOPSIS

```
!=====
subroutine calculate_distance_between_prototypes(kohonen_map)
!=====
class(self_organized_map) :: kohonen_map
```

8.3 self_organized_map_utilities/calculate_distance_matrix*[self_organized_map_utilities] [Functions]***NAME**`calculate_distance_matrix`**PURPOSE**

Subroutine to calculate the distance between the units inside a kohonen layer

SYNOPSIS

```

!=====
  subroutine calculate_distance_matrix(coordinates,distance_matrix,grid_type,toroidal)
!=====
  real(kind=8),dimension(:,,:),intent(inout) :: coordinates,distance_matrix
  character(len=*) :: grid_type
  logical :: toroidal

```


8.4 self_organized_map_utilities/calculate_sigma

[self_organized_map_utilities] [Functions]

NAME

calculate_sigma

PURPOSE

Function to calculate the scaling factor sigma

SYNOPSIS

```
!=====
function calculate_sigma(kohonen_map,input_data,seed) result(sigma)
!=====
class(self_organized_map) :: kohonen_map
real(kind=8),dimension(:,:),intent(inout) :: input_data
integer,intent(inout),optional :: seed
real(kind=8) :: sigma
```

8.5 self_organized_map_utilities/calculate_u_matrix

[*self_organized_map_utilities*] [*Functions*]

NAME

calculate_u_matrix

PURPOSE

Subroutine to calculate the u_matrix

SYNOPSIS

```
!=====
subroutine calculate_u_matrix(kohonen_map)
!=====
class(self_organized_map) :: kohonen_map
```

8.6 self_organized_map_utilities/create

[*self_organized_map_utilities*] [*Functions*]

NAME

create

PURPOSE

Constructor for self_organized_map

SYNOPSIS

```
!=====
subroutine create_som(kohonen_map,training_parameters)
!=====
class(self_organized_map) :: kohonen_map
type(kohonen_layer_parameters),dimension(:) :: training_parameters
```

8.7 self_organized_map_utilities/destroy

[*self_organized_map_utilities*] [*Functions*]

NAME

destroy

PURPOSE

Destructor for self_organized_map

SYNOPSIS

```
!=====
subroutine destroy_som(kohonen_map)
!=====
class(self_organized_map) :: kohonen_map
```

8.8 self_organized_map_utilities/external_predict_map

[*self_organized_map_utilities*] [*Functions*]

NAME

external_predict_map

PURPOSE

Subroutine to connect this module to R

SYNOPSIS

```
!=====
subroutine external_predict_map(prot,nx,ny,new_pat,npat,nvar,node_index) &
  bind(C, name="predict_som_")
!=====
use, intrinsic :: iso_c_binding, only : c_double, c_int
integer(c_int),intent(in) :: nx,ny,npat,nvar
real(c_double),intent(in) :: prot(nx*ny,nvar),new_pat(npat,nvar)
integer(c_int),intent(out) :: node_index(npat,3)
```

8.9 self_organized_map_utilities/external_train_map

[self_organized_map_utilities] [Functions]

NAME

external_train_map

PURPOSE

Subroutine to connect the self_organizing_map module to R o C

SYNOPSIS

```
!=====
subroutine external_train_map(x,nvar,npat,nx,ny,nepoch,alpha,grid_type,&
    distance_type,neigh_type,toroidal,prot,distortion,&
    u_matrix,coords,number_patterns,node_index) bind(C, name="train_som_")
!=====
use, intrinsic :: iso_c_binding, only : c_double, c_int, c_char
real(kind=8),parameter :: version=0.1d0
character(len=*),parameter :: program_name="som_train"
integer(c_int), intent(in) :: nvar,npat,nx,ny,nepoch,toroidal
real(c_double),intent(out) :: prot(nx*ny,nvar),distortion(nepoch)
real(c_double),intent(out) :: u_matrix(2*nx-1,2*ny-1),coords(nx*ny,3)
integer(c_int),intent(out) :: number_patterns(nx,ny),node_index(npat,3)
real(c_double),intent(in) :: x(npat,nvar)
real(c_double),intent(in) :: alpha
integer(c_int),intent(in) :: grid_type,distance_type,neigh_type
```

8.10 self_organized_map_utilities/find_best_match_unit*[self_organized_map_utilities] [Functions]***NAME**`find_best_match_unit`**PURPOSE**

Subroutine to calculate the best match unit

SYNOPSIS

```

=====
subroutine find_best_match_unit(kohonen_map,current_prototype,ihit,jhit,khit,dist_hit)
=====
class(self_organized_map) :: kohonen_map
type(kohonen_prototype),intent(inout) :: current_prototype
integer,intent(out) :: ihit,jhit,khit
real(kind=8),intent(out) :: dist_hit

```

8.11 self_organized_map_utilities/find_bmu_grid

[self_organized_map_utilities] [Functions]

NAME

find_bmu_grid

PURPOSE

Subroutine to calculate the best match unit over the grid

SYNOPSIS

```
!=====
subroutine find_bmu_grid(kohonen_map,input_data)
!=====
class(self_organized_map) :: kohonen_map
type(kohonen_pattern),dimension(:),intent(inout) :: input_data
```


8.12 self_organized_map_utilities/get_count

[self_organized_map_utilities] [Functions]

NAME

get_count

PURPOSE

Function to get count matrix for self_organized_map

SYNOPSIS

```
!=====
subroutine get_count_som(kohonen_map,count_)
!=====
class(self_organized_map) :: kohonen_map
integer,dimension(:,:,:),intent(inout) :: count_
```

8.13 self_organized_map_utilities/get_prototypes

[*self_organized_map_utilities*] [*Functions*]

NAME

get_prototypes

PURPOSE

Subroutine to get SOM prototypes

SYNOPSIS

```
!=====
subroutine get_prototypes(kohonen_map,prototypes)
!=====
class(self_organized_map) :: kohonen_map
real(kind=8),dimension(:,:),intent(out) :: prototypes
```

8.14 self_organized_map_utilities/index2position

[self_organized_map_utilities] [Functions]

NAME

position2index

PURPOSE

Subroutine to calculate the position ix,iy,iz inside a rectangular grid from index

SYNOPSIS

```
!=====
subroutine index2position(index_,nx,ny,nz,cx,cy,cz)
!=====
integer,intent(inout) :: index_,nx,ny,nz
integer,intent(inout) :: cx,cy,cz
```

8.15 self_organized_map_utilities/position2index

[self_organized_map_utilities] [Functions]

NAME

position2index

PURPOSE

Function to calculate the index inside a rectangular grid from position ix,iy,iz

SYNOPSIS

```
!=====
function position2index(ix,iy,iz,nx,ny) result(index_)
!=====
integer,intent(inout) :: ix,iy,iz,nx,ny
integer ::index_
```

8.16 self_organized_map_utilities/predict*[self_organized_map_utilities] [Functions]***NAME**`predict`**PURPOSE**`Prediction function for self_organized_map`**SYNOPSIS**

```

=====
subroutine predict_som(kohonen_map,input_data,map_output)
=====
class(self_organized_map) :: kohonen_map
type(kohonen_pattern),dimension(:),intent(inout) :: input_data
integer,dimension(:,:),intent(out) :: map_output

```

8.17 self_organized_map_utilities/print

[*self_organized_map_utilities*] [*Functions*]

NAME

print

PURPOSE

Print function for self_organized_map

SYNOPSIS

```
!=====
subroutine print_som(kohonen_map,unit_)
!=====
class(self_organized_map) :: kohonen_map
integer,intent(inout),optional :: unit_
```

8.18 self_organized_map_utilities/query_som*[self_organized_map_utilities] [Functions]***NAME**`query_som`**PURPOSE**

Function to find the input samples associated with specific vector

SYNOPSIS

```

=====
subroutine query_som(kohonen_map,input_pattern,sample_index) !,output_patterns)
=====
class(self_organized_map) :: kohonen_map
real(kind=8),dimension(:,:),intent(inout) :: input_pattern
integer,allocatable :: sample_index(:)

```

8.19 self_organized_map_utilities/read_som

[self_organized_map_utilities] [Functions]

NAME

read_som

PURPOSE

Subroutine to read the prototypes to define a self_organized_map

SYNOPSIS

```
!=====
subroutine read_som(kohonen_map,som_fl)
!=====
class(self_organized_map) :: kohonen_map
character(len=*) :: som_fl
```


8.20 self_organized_map_utilities/self_organized_map

[self_organized_map_utilities] [Classes]

NAME

self_organized_map

PURPOSE

Class to represent a self_organized_map

ATTRIBUTES

```

type(kohonen_prototype),allocatable :: grid(:, :, :)
integer,allocatable :: number_patterns(:, :, :),cells_index(:, :)
real(kind=8),allocatable :: u_matrix(:, :, :),distance(:, :)
real(kind=8),allocatable :: cells_distances(:, :),coordinates(:, :)
type(kohonen_layer_parameters) :: parameters
type(factory_distance) :: factory
class(distance_base),allocatable :: distance_function
real(kind=8),allocatable :: distortion(:)
!
integer,allocatable :: grid_pattern_index(:, :, :),list_node_grid(:, :, :, :)
!
contains

```

METHODS

```

procedure,public :: create => create_som
procedure,public :: destroy => destroy_som
procedure,private :: train_som_data
procedure,public :: train => train_som_data
procedure,public :: predict => predict_som
procedure,public :: print => print_som
procedure,public :: read => read_som
procedure,public :: get_count => get_count_som
procedure,public :: query => query_som
procedure,public :: get_prototypes
!procedure,public :: get_index => get_index_som
!procedure,public :: get_u_matrix => get_u_matrix_som
procedure,private :: find_best_match_unit
procedure,private :: update_weights
! procedure,private :: update_weights1
procedure,private :: find_bmu_grid
procedure,private :: calculate_u_matrix
procedure,private :: calculate_sigma
!
procedure,nopass,private :: position2index
procedure,nopass,private :: index2position
procedure,nopass,private :: calculate_distance_matrix
procedure,nopass,private :: calculate_coordinates

```

```
    procedure,private :: calculate_distance_between_prototypes
!
    procedure,nopass,public :: external_train_map
    procedure,nopass,public :: external_predict_map
```

8.21 self_organized_map_utilities/train_som_data*[self_organized_map_utilities] [Functions]***NAME**`train_som_data`**PURPOSE**Training function for `self_organized_map`**SYNOPSIS**

```

=====
subroutine train_som_data(kohonen_map,input_data)
=====
class(self_organized_map) :: kohonen_map
type(kohonen_pattern),dimension(:),intent(inout) :: input_data

```

8.22 self_organized_map_utilities/update_weights

[self_organized_map_utilities] [Functions]

NAME

update_weights

PURPOSE

Subroutine to update the weights

SYNOPSIS

```
!=====
subroutine update_weights(kohonen_map,current_values,ihit,jhit,khit,&
    maximum_radius,iteration)
!=====
class(self_organized_map) :: kohonen_map
real(kind=8),dimension(:,:),intent(inout) :: current_values
integer,intent(inout) :: ihit,jhit,khit,iteration
real(kind=8),intent(inout) :: maximum_radius
```

9 Kohonen/two_level_self_organized_map_utilities

[Modules]

NAME

MODULE two_level_self_organized_map_utilities

PURPOSE

In this module the two-level SOM is defined

AUTHOR

Oscar Garcia-Cabrejo

NOTES

MODIFICATION HISTORY

9.1 two_level_self_organized_map_utilities/calculate_cluster_measures

[two_level_self_organized_map_utilities] [Functions]

NAME

calculate_cluster_measures

PURPOSE

Subroutine to calculate some clustering statistics of a two-level self_organized_map

SYNOPSIS

```
!=====
subroutine calculate_cluster_measures(kohonen_map,results)
!=====
class(two_level_self_organized_map) :: kohonen_map
real(kind=8),dimension(:,:,:) :: results
```

9.2 two_level_self_organized_map_utilities/calculate_coordinates*[two_level_self_organized_map_utilities] [Functions]***NAME**

calculate_coordinates

PURPOSE

Subroutine to calculate the coordinates of the units inside a kohonen layer

SYNOPSIS

```

!=====
  subroutine calculate_coordinates(current_index,ix,iy,iz,nx,ny,nz,coordinates,node_type)
!=====
  integer,intent(inout) :: current_index,ix,iy,iz,nx,ny,nz
  real(kind=8),dimension(:,:),intent(out) :: coordinates
  character(len=*),intent(in) :: node_type

```

9.3 two_level_self_organized_map_utilities/calculate_distance_between_prototypes

[two_level_self_organized_map_utilities] [Functions]

NAME

calculate_distance_between_prototypes

PURPOSE

Subroutine to calculate distance between prototypes

SYNOPSIS

```
!=====
subroutine calculate_distance_between_prototypes(kohonen_map)
!=====
class(two_level_self_organized_map) :: kohonen_map
```


9.4 two_level_self_organized_map_utilities/calculate_distance_matrix

[two_level_self_organized_map_utilities] [Functions]

NAME

calculate_distance_matrix

PURPOSE

Subroutine to calculate the distance between the units inside a kohonen layer

SYNOPSIS

```
!=====
subroutine calculate_distance_matrix(coordinates,distance_matrix,grid_type,toroidal)
!=====
real(kind=8),dimension(:,,:),intent(inout) :: coordinates,distance_matrix
character(len=*) :: grid_type
logical :: toroidal
```

9.5 two_level_self_organized_map_utilities/calculate_sum2_clusters_grid

[two_level_self_organized_map_utilities] [Functions]

NAME

calculate_sum2_clusters_grid

PURPOSE

Subroutine to calculate some clustering statistics of a two-level self_organized_map

SYNOPSIS

```
!=====
subroutine calculate_sum2_clusters_grid(kohonen_map,results)
!=====
class(two_level_self_organized_map) :: kohonen_map
real(kind=8),dimension(:),optional :: results
```

9.6 two_level_self_organized_map_utilities/create

[two_level_self_organized_map_utilities] [Functions]

NAME

create

PURPOSE

Constructor of a two_level self_organized_map

SYNOPSIS

```
!=====
subroutine create_2lsom(kohonen_map,training_parameters)
!=====
class(two_level_self_organized_map) :: kohonen_map
type(kohonen_layer_parameters),dimension(:) :: training_parameters
```

9.7 two_level_self_organized_map_utilities/destroy

[two_level_self_organized_map_utilities] [Functions]

NAME

destroy

PURPOSE

Destructor of a two_level self_organized_map

SYNOPSIS

```
!=====
subroutine destroy_2lsom(kohonen_map)
!=====
class(two_level_self_organized_map) :: kohonen_map
```

9.8 two_level_self_organized_map_utilities/evaluate_2lsom*[two_level_self_organized_map_utilities] [Functions]***NAME**

evaluate_2lsom

PURPOSE

Subroutine to calculate some clustering statistics of a two-level self_organized_map

SYNOPSIS

```

=====
subroutine evaluate_2lsom(kohonen_map,input_data,results)
=====
class(two_level_self_organized_map) :: kohonen_map
type(kohonen_pattern),dimension(:),intent(inout) :: input_data
real(kind=8),dimension(:),optional :: results

```

9.9 two_level_self_organized_map_utilities/get_cluster_samples

[two_level_self_organized_map_utilities] [Functions]

NAME

get_cluster_samples

PURPOSE

Accessor to cluster results obtained using a two-level self_organized_map

SYNOPSIS

```
!=====
subroutine get_cluster_samples(kohonen_map,clusters)
!=====
class(two_level_self_organized_map) :: kohonen_map
integer,dimension(:),intent(inout) :: clusters
```

9.10 two_level_self_organized_map_utilities/index2position

[two_level_self_organized_map_utilities] [Functions]

NAME

index2position

PURPOSE

Subroutine to calculate the position ix,iy,iz inside a rectangular grid from index

SYNOPSIS

```
!=====
  subroutine index2position(index_,nx,ny,nz,cx,cy,cz)
!=====
integer,intent(inout) :: index_,nx,ny,nz
integer,intent(inout) :: cx,cy,cz
```

9.11 two_level_self_organized_map_utilities/position2index

[two_level_self_organized_map_utilities] [Functions]

NAME

position2index

PURPOSE

Function to calculate the index inside a rectangular grid from position ix,iy,iz

SYNOPSIS

```
!=====
function position2index(ix,iy,iz,nx,ny) result(index_)
!=====
integer,intent(inout) :: ix,iy,iz,nx,ny
integer ::index_
```


9.12 two_level_self_organized_map_utilities/predict_2lsom*[two_level_self_organized_map_utilities] [Functions]***NAME**`predict_2lsom`**PURPOSE**

Subroutine to make a prediction from a trained two_level self_organized_map

SYNOPSIS

```

=====
subroutine predict_2lsom(kohonen_map,input_data,map_output)
=====
class(two_level_self_organized_map) :: kohonen_map
type(kohonen_pattern),dimension(:),intent(inout) :: input_data
integer,dimension(:,:),intent(out) :: map_output

```

9.13 two_level_self_organized_map_utilities/print_2lsom*[two_level_self_organized_map_utilities] [Functions]***NAME**`print_2lsom`**PURPOSE**

Subroutine to print the layers of a two_level self_organized_map

SYNOPSIS

```

=====
subroutine print_2lsom(kohonen_map,unit_)
=====
class(two_level_self_organized_map) :: kohonen_map
integer,optional :: unit_

```

9.14 two_level_self_organized_map_utilities/query_som*[two_level_self_organized_map_utilities] [Functions]***NAME**`query_som`**PURPOSE**

Function to find the input samples associated with specific vector

SYNOPSIS

```

=====
subroutine query_2lsom(kohonen_map,input_pattern,sample_index) !,output_patterns)
=====
class(two_level_self_organized_map) :: kohonen_map
real(kind=8),dimension(:,:),intent(inout) :: input_pattern
integer,allocatable :: sample_index(:)

```

9.15 two_level_self_organized_map_utilities/set_cluster_layer*[two_level_self_organized_map_utilities] [Functions]***NAME**`set_cluster_layer`**PURPOSE**

Subroutine to initialize the cluster layer of a Two Level Self-Organizing Map

SYNOPSIS

```

=====
subroutine set_cluster_layer(kohonen_map,seed)
=====
class(two_level_self_organized_map) :: kohonen_map
integer :: seed
!
integer :: ix,number_clusters,ierr
real(kind=8),allocatable :: input(:,,:)

```

9.16 two_level_self_organized_map_utilities/set_parameters

[two_level_self_organized_map_utilities] [Functions]

NAME

set_parameters

PURPOSE

Subroutine to set parameters

SYNOPSIS

```
!=====
subroutine set_parameters(kohonen_map,training_parameters)
!=====
class(two_level_self_organized_map) :: kohonen_map
type(kohonen_layer_parameters),dimension(2) :: training_parameters
```

9.17 two_level_self_organized_map_utilities/train_2lsom

[two_level_self_organized_map_utilities] [Functions]

NAME

train_2lsom

PURPOSE

Subroutine to train a two_level self_organized_map

SYNOPSIS

```
!=====
subroutine train_2lsom(kohonen_map,input_data)
!=====
class(two_level_self_organized_map) :: kohonen_map
type(kohonen_pattern),dimension(:),intent(inout) :: input_data
```

9.18 two_level_self_organized_map_utilities/train_cluster_layer

[two_level_self_organized_map_utilities] [Functions]

NAME

train_cluster_layer

PURPOSE

Subroutine to train the cluster layer of a two_level self_organized_map

SYNOPSIS

```
!=====
subroutine train_cluster_layer(kohonen_map)
!=====
class(two_level_self_organized_map) :: kohonen_map
```

9.19 two_level_self_organized_map_utilities/train_grid_layer

[two_level_self_organized_map_utilities] [Functions]

NAME

train_grid_layer

PURPOSE

Subroutine to train the grid layer of a two_level self_organized_map

SYNOPSIS

```
!=====
subroutine train_grid_layer(kohonen_map,input_data)
!=====
class(two_level_self_organized_map) :: kohonen_map
type(kohonen_pattern),dimension(:),intent(inout) :: input_data
```


9.20 two_level_self_organized_map_utilities/two_level_self_organized_map

[two_level_self_organized_map_utilities] [Classes]

NAME

two_level_self_organized_map

PURPOSE

Class to represent a two level self_organized_map

ATTRIBUTES

```

type(kohonen_prototype),allocatable :: grid(:,:,),cluster_layer(:)
real(kind=8),allocatable :: coordinates(:,)
integer,allocatable :: number_patterns(:,,:),cells_index(:,)
integer,allocatable :: cluster_number_patterns(),cluster_cells_index(:,)
integer,allocatable :: grid_cluster(:,,:),cluster_samples(:)
real(kind=8),allocatable :: u_matrix(:,,:),distance(:,),cells_distances(:,)
integer,allocatable :: number_cluster_samples(),index_cluster_samples(:,)
type(kohonen_layer_parameters),dimension(2) :: parameters
type(factory_distance) :: factory
class(distance_base),allocatable :: distance_function
integer :: number_variables,number_variables1,number_variables2,number_clusters
integer :: number_nodes
contains

```

METHODS

```

procedure,public :: create => create_2lsom
procedure,public :: destroy => destroy_2lsom
procedure,public :: train => train_2lsom
procedure,public :: predict => predict_2lsom
procedure,public :: train_grid_layer
procedure,public :: train_cluster_layer
procedure,public :: print => print_2lsom
procedure,public :: query => query_2lsom
procedure,public :: set_cluster_layer
procedure,public :: set_parameters
! procedure,public :: read => read_som
procedure,private :: query_2lsom
procedure,public :: read_som_layer
procedure,private :: calculate_u_matrix
procedure,private :: find_best_match_unit
procedure,private :: update_weights
procedure,private :: calculate_distance_between_prototypes
procedure,private :: assign_input_to_clusters
!procedure,public :: get_count => get_count_2lsom
!procedure,public :: get_index => get_index_som
!procedure,public :: get_u_matrix => get_u_matrix_som
procedure,public :: calculate_sum2_clusters_samples => evaluate_2lsom

```

```
procedure,public :: get_cluster_samples
procedure,public :: calculate_sum2_clusters_grid
procedure,nopass,private :: calculate_distance_matrix
procedure,nopass,private :: calculate_coordinates
```

9.20.1 two_level_self_organized_map/assign_input_to_clusters

[*two_level_self_organized_map*] [*Functions*]

NAME

assign_input_to_clusters

PURPOSE

Subroutine to assign input to clusters

SYNOPSIS

```
!=====
subroutine assign_input_to_clusters(kohonen_map,input_patterns)
!=====
class(two_level_self_organized_map) :: kohonen_map
type(kohonen_pattern),dimension(:),intent(inout) :: input_patterns
```

9.20.2 two_level_self_organized_map/calculate_u_matrix*[two_level_self_organized_map] [Functions]***NAME**`calculate_u_matrix`**PURPOSE**

Subroutine to calculate the u_matrix

SYNOPSIS

```
!=====
subroutine calculate_u_matrix(kohonen_map)
!=====
class(two_level_self_organized_map) :: kohonen_map
```

9.20.3 two_level_self_organized_map/find_best_match_unit

[*two_level_self_organized_map*] [*Functions*]

NAME

find_best_match_unit

PURPOSE

Subroutine to calculate the best match unit

SYNOPSIS

```
!=====
subroutine find_best_match_unit(kohonen_map,current_prototype,ihit,jhit,khit,dist_hit)
!=====
class(two_level_self_organized_map) :: kohonen_map
type(kohonen_prototype),intent(inout) :: current_prototype
integer,intent(out) :: ihit,jhit,khit
real(kind=8),intent(out) :: dist_hit
```

9.20.4 two_level_self_organized_map/read_som_layer

[*two_level_self_organized_map*] [*Functions*]

NAME

read_som_layer

PURPOSE

Subroutine to read the prototypes of the first/seconf layer of a two level
self_organized_map

SYNOPSIS

```
!=====
subroutine read_som_layer(kohonen_map,som_fl,layer_type)
!=====
class(two_level_self_organized_map) :: kohonen_map
character(len=*) :: som_fl,layer_type
```

9.21 two_level_self_organized_map_utilities/update_weights*[two_level_self_organized_map_utilities] [Functions]***NAME**

update_weights

PURPOSE

Subroutine to update the weights

SYNOPSIS

```

=====
subroutine update_weights(kohonen_map,current_values,ihit,jhit,khit,&
                        maximum_radius,iteration)
=====
class(two_level_self_organized_map) :: kohonen_map
real(kind=8),dimension(:,:),intent(inout) :: current_values
integer,intent(inout) :: ihit,jhit,khit,iteration
real(kind=8),intent(inout) :: maximum_radius

```

10 ROBODoc/ROBODoc Cascading Style Sheet

[Modules]