CODeME

Generated by Doxygen 1.8.11

Contents

Chapter 1

Bug List

File BUMDA.h
No known bugs.

File CMA_ES.h
No known bugs.

File Dynamic_Mechanism.h
No known bugs.

File Gauss.c
No known bugs.

File Gauss.h
No known bugs.

File Interpolation_Algorithm.c
No known bugs.

File Interpolation_Algorithm.h
No known bugs.

File Kinematic_Mechanisms.h
No known bugs.

File Manipulability_Functions.h
No known bugs.

File Matrix.c
No known bugs.

File Matrix.h
No known bugs.

File Numerical_Optimization.h
No known bugs.

File Objective_Functions.h
No known bugs.

File OMNI.h
No known bugs.

File Optimizers.h
No known bugs.

File R_pipeScript.c

No known bugs.

File R_pipeScript.h

No known bugs.

File Random.c

No known bugs.

File Random.h

No known bugs.

File SVD_Descomposition.c

No known bugs.

File SVD_Descomposition.h

No known bugs.

Chapter 2

Class Index

2.1 Class List

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

Chapter 3

File Index

3.1 File List

Here is a list of all documented files with brief descriptions:

Chapter 4

Class Documentation

4.1 BUMDA Struct Reference

Public Attributes

- int **Dimension**
- int **PopulationSize**
- double * **Max_Lim**
- double * **Min_Lim**
- double * **Var**
- double * **Mu**
- double ** **Population**
- double * **Best**
- double **Best_Fitness**
- int **gen**
- int **neval**
- int **Max_Evaluation**
- int **Max_Generations**
- int **n**
- double **Min_Var**
- double **Umbral**
- struct {
  int **flg**
  double **val**
  } **stStopFitness**

- TRAND **SEED**

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

- BUMDA.h

4.2 cmaes_boundary_transformation_t Struct Reference

Public Attributes

- double const * **lower_bounds**
- double const * **upper_bounds**
- unsigned long **len_of_bounds**
- double * **al**
- double * **au**

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

- boundary_transformation.h

4.3 CMAES_DATA Struct Reference

Public Attributes

- double * **Max_Lim**
- double * **Min_Lim**
- cmaes_t **CMA**
- int **gen**
- char **Filenames** [2][50]
- double *const * **pop**
- double * **arFunvals**
- double const * **Best**
- double **Best_Fitness**
- double * **inxstart**
- double * **inrgstddev**

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

- CMA_ES.h

4.4 cmaes_random_t Struct Reference

Public Attributes

- long int **startseed**
- long int **aktseed**
- long int **aktrand**
- long int * **rgrand**
- short **flgstored**
- double **hold**

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

- cmaes.h

4.5 cmaes_readpara_t Struct Reference

Public Attributes

- char * **filename**
- short **flgsupplemented**
- int **N**
- unsigned int **seed**
- double * **xstart**
- double * **typicalX**
- int **typicalXcase**
- double * **rgInitialStds**
- double * **rgDiffMinChange**
- double **stopMaxFunEvals**
- double **facmaxeval**
- double **stopMaxIter**
- struct {
  int **flg**
  double **val**
  } **stStopFitness**

- double **stopTolFun**
- double **stopTolFunHist**
- double **stopTolX**
- double **stopTolUpXFactor**
- int **lambda**
- int **mu**
- double **mucov**
- double **mueff**
- double * **weights**
- double **damps**
- double **cs**
- double **ccumcov**
- double **ccov**
- double **diagonalCov**
- struct {
  int **flgalways**
  double **modulo**
  double **maxtime**
  } **updateCmode**

- double **facupdateCmode**
- char * **weigkey**
- char **resumefile** [99]
- const char ** **rgsformat**
- void ** **rgpadr**
- const char ** **rgskeyar**
- double *** **rgp2adr**
- int **n1para**
- int **n1outpara**
- int **n2para**

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

- cmaes.h

4.6 cmaes_t Struct Reference

Public Attributes

- const char * **version**
- cmaes_readpara_t **sp**
- cmaes_random_t **rand**
- double **sigma**
- double * **rgxmean**
- double * **rgxbestever**
- double ** **rgrgx**
- int * **index**
- double * **arFuncValueHist**
- short **flgIniphase**
- short **flgStop**
- double **chiN**
- double ** **C**
- double ** **B**
- double * **rgD**
- double * **rgpc**
- double * **rgps**
- double * **rgxold**
- double * **rgout**
- double * **rgBDz**
- double * **rgdTmp**
- double * **rgFuncValue**
- double * **publicFitness**
- double **gen**
- double **countevals**
- double **state**
- double **maxdiagC**
- double **mindiagC**
- double **maxEW**
- double **minEW**
- char **sOutString** [330]
- short **flgEigensysIsUptodate**
- short **flgCheckEigen**
- double **genOfEigensysUpdate**
- cmaes_timings_t **eigenTimings**
- double **dMaxSignifKond**
- double **dLastMinEWgroesserNull**
- short **flgresumedone**
- time_t **printtime**
- time_t **writetime**
- time_t **firstwritetime**
- time_t **firstprinttime**

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

- cmaes.h

4.7 cmaes_timings_t Struct Reference

Public Attributes

- double **totaltime**
- double **totaltotaltime**
- double **tictoctime**
- double **lasttictoctime**
- clock_t **lastclock**
- time_t **lasttime**
- clock_t **ticclock**
- time_t **tictime**
- short **istic**
- short **isstarted**
- double **lastdiff**
- double **tictoczwischensumme**

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

- cmaes.h

4.8 CONTROL_DATA Struct Reference

Public Attributes

- int **DOF**
- int **Links**
- int **NTP**
- int **Articulations**
- double * **Time**
- double * **Initial_Position**
- double * **Initial_Velocity**
- double * **Initial_Acceleration**
- double * **Trayectory_Parameters**

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

- Complex_Dynamic_Control.h

4.9 CONTROL_PID Struct Reference

Public Attributes

- int **DOF**
- int **Links**
- int **NTP**
- int **Articulations**
- double * **Time**
- double * **Initial_Position**
- double * **Initial_Velocity**
- double * **Trayectory_Parameters**

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

- Complex_Kinematic_Control.h

4.10 CUBIC_DATA Struct Reference

Public Attributes

- int **DoF**
- int **Intervals**
- double * **Time**
- double * **Position**
- double * **Velocity**
- double ** **Parameter**

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

- Trayectory_Library.h

4.11 DATA_INTER Struct Reference

Public Attributes

- int **Dimension**
- int **Data_l**
- double * **Data_t**
- double ** **Data_p**
- double ** **Data_S**

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

- Interpolation_Algorithm.h

4.12 fcoords Struct Reference

Public Attributes

- float **x**
- float **y**
- float **z**

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

- defs_and_types.h

4.13 hist_rec Struct Reference

Public Attributes

- struct hist_rec * **prev**
- struct hist_rec * **next**
- float * **basis** [3]
- int **pos**

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

- defs_and_types.h

4.14 icoords Struct Reference

Public Attributes

- int **x**
- int **y**
- int **z**

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

- defs_and_types.h

4.15 individual Struct Reference

Public Attributes

- int **rank**
- double **constr_violation**
- double * **xreal**
- int ** **gene**
- double * **xbin**
- double * **obj**
- double * **constr**
- double **realvar_crowd_dist**
- double **binvar_crowd_dist**
- double **obj_crowd_dist**
- double **crowd_dist**

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

- global.h

4.16 lcoords Struct Reference

Public Attributes

- long **x**
- long **y**
- long **z**

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

- defs_and_types.h

4.17 lims Struct Reference

Public Attributes

- float **min**
- float **max**

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

- defs_and_types.h

4.18 lists Struct Reference

Public Attributes

- int **index**
- struct lists * **parent**
- struct lists * **child**

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

- global.h

4.19 MANIP_DATA Struct Reference

Public Attributes

- int **DOF**
- int **Num_Art_Constraints**
- double **Discretization**
- double **Min_Jacobian**
- double * **Upper_Constraints**
- double * **Lower_Constraints**

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

- Manipulability_Functions.h

4.20 OMNI_DATA Struct Reference

Public Attributes

- int **nreal**
- int **nobj**
- int **ncon**
- int **popsize**
- double **pcross_real**
- double **pmut_real**
- double **eta_c**
- double **eta_m**
- int **ngen**
- int **nrealmut**
- int **nrealcross**
- double * **min_realvar**
- double * **max_realvar**
- double * **epsilon**
- double **delta**
- double **seed**
- int **mate**
- int **input_type**
- int **var_option**
- int **obj_option**
- int **frequency**
- double * **Best**
- double **Best_Fitness**

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

- OMNI.h

4.21 OPT_PAR Struct Reference

Public Attributes

- int **Dimension**
- int **Max_Iterations**
- double **Tolerance**
- double * **Parameters**
- double * **Fun_Parameters**

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

- Numerical_Optimization.h

4.22 OPTIMIZATION_DATA Struct Reference

```
#include <Optimizers.h>
```

Public Attributes

- char * Type
- char * **Name**
- char * **InitFile**
- char * **OptSignals**
- int Dimension
- double ** **Limits**
- int stopMaxFunEvals
- int **stopMaxIter**
- double **stopFitness**
- double GA_seed
- double **GA_pcross_real**
- double **GA_pmut_real**
- double **GA_eta_c**
- double **GA_eta_m**
- double **GA_delta**
- int **GA_popsize**
- int **GA_ncon**
- int **GA_ngen**
- int **GA_nrealmut**
- int **GA_nrealcross**
- int **GA_mate**
- int **GA_input_type**
- int **GA_var_option**
- int **GA_obj_option**
- int **GA_frequency**

4.22.1 Detailed Description

Optimization Problem Data

4.22.2 Member Data Documentation

4.22.2.1 int OPTIMIZATION_DATA::Dimension

Problem Data

4.22.2.2 double OPTIMIZATION_DATA::GA_seed

GA

4.22.2.3 int OPTIMIZATION_DATA::stopMaxFunEvals

Stop Criterion

4.22.2.4 char* OPTIMIZATION_DATA::Type

Optimizer

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

- Optimizers.h

4.23 population Struct Reference

Public Attributes

- individual * **ind**

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

- global.h

4.24 ROBOT_DATA Struct Reference

Public Attributes

- int **DOF**
- double Density
- double * **Mass**
- double * **Radoius**
- double *** **Inertia**
- double ** Torque
- double ** **Position**
- double ** **Velocity**
- double ** **Acceleration**
- double * Gravity
- double * **Viscous_fric**
- double * **Dynamic_fric**

4.24.1 Member Data Documentation

4.24.1.1 double ROBOT_DATA::Density

Link Data

4.24.1.2 double* ROBOT_DATA::Gravity

Envirement Data

4.24.1.3 double** ROBOT_DATA::Torque

Limit Data

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

- Complex_Dynamic_Control.h

4.25 TRAND Struct Reference

Public Attributes

- unsigned **z1**
- unsigned **z2**
- unsigned **z3**
- unsigned **z4**

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

- Random.h

Chapter 5

File Documentation

5.1 BUMDA.h File Reference

Function prototypes for the BUMDA optimizers.

```
#include "Optimizers.h"
#include "Miscellaneous/Gauss.h"
```

Classes

- struct BUMDA

Typedefs

- typedef struct BUMDA **BUMDA_DATA**

Functions

- int **Compare** (double A, double B)
- void **BUMDA_Order_Population** (BUMDA_DATA *evo, double *Evaluations)
- int **BUMDA_CheckVar** (BUMDA_DATA *evo)
- void **BUMDA_Init** (OPTIMIZATION_DATA problem, BUMDA_DATA *evo)
- void **BUMDA_Init_FILE** (OPTIMIZATION_DATA problem, BUMDA_DATA *evo)
- int **BUMDA_TestForTermination** (BUMDA_DATA *evo)
- void **BUMDA_Set_Distribution** (BUMDA_DATA *evo, double *Evaluation)
- void **BUMDA_Get_New_Population** (BUMDA_DATA *evo, double *Eval)
- void **BUMDA_free_memory** (BUMDA_DATA *evo)
- void **Evolutive_BUMDA** (BUMDA_DATA *evo, void **Parameters, FILE *fileOut)

5.1.1 Detailed Description

Function prototypes for the BUMDA optimizers.

This contains the prototypes for the BUMDA optimizers and eventually any macros,constants, or global variables needed.

Author

Salvador Botello-Aceves

Bug No known bugs.

5.2 CMA_ES.h File Reference

Function prototypes for the trajectory routines.

```
#include "Optimizers.h"
#include "Miscellaneous/Gauss.h"
#include "CMAES/cmaes_interface.h"
```

Classes

- struct CMAES_DATA

Typedefs

- typedef struct CMAES_DATA **CMAES_DATA**

Functions

- void **CMAES_Init** (OPTIMIZATION_DATA problem, CMAES_DATA *evo)
- void **CMAES_Init_FILE** (OPTIMIZATION_DATA problem, CMAES_DATA *evo)
- void **CMAES_free_memory** (CMAES_DATA *evo)
- void **Evolutive_CMAES** (CMAES_DATA *evo, void **Parameters, FILE *fileOut)

5.2.1 Detailed Description

Function prototypes for the trajectory routines.

This contains the prototypes for the trajectory routine and eventually any macros,constants, or global variables needed.

Author

Salvador Botello-Aceves

Bug No known bugs.

5.3 Dynamic_Mechanism.h File Reference

Function prototypes for the trajectory routines.

```
#include "Complex_Dynamic_Control.h"
```

Functions

- double ** **Serial_midLink_Mass_Position** (int DoF, double *DH_Parameters, double *Theta)
- double *** **Serial_Linear_Cylindrical_Inertia_Tensor** (int DoF, double *DH_Parameters, double *Mass, double *Radious)
- double ** **Serial_Compute_Jacobian_Position** (int DoF, double *DH_Parameters, double *Theta)
- double ** **Serial_Compute_Jacobian_Orientation** (int DoF, double *DH_Parameters, double *Theta)
- double ** Serial_Set_Inertia_Matrix (int DoF, double *DH_Parameters, double *Theta, double ***Inertia, double *Mass)
- double *** **Serial_Get_Inertia_Derivative** (int DoF, double *DH_Parameters, double *Theta, double ***Inertia, double *Mass)
- double ** **Serial_Set_Coriolis_Matrix** (int DoF, double *DH_Parameters, double *Theta, double ***Inertia, double *Mass, double *Velocitys)
- double * **Serial_Set_Gravitational_Vector** (int DoF, double *DH_Parameters, double *Theta, double *Mass, double *Gravity)
- void **Serial_Direct_Dynamic** (int DoF, double *DH_Parameters, double **q, double *Torque, double ***Inertia, double *Mass, double *Gravity)
- void **Serial_Inverse_Dynamic** (int DoF, double *DH_Parameters, double **q, double *Torque, double ***Inertia, double *Mass, double *Gravity, double dt)
- int **Elbow_Set_Functions** ()
- double ** **Elbow_Set_Inertia_Matrix** (double *DH_Parameters, double **q, ROBOT_DATA Robot)
- double ** **Elbow_Set_Coriolis_Matrix** (double *DH_Parameters, double **q, ROBOT_DATA Robot)
- double * **Elbow_Set_Gravitational_Vector** (double *DH_Parameters, double **q, ROBOT_DATA Robot)
- void **Elbow_Direct_Dynamic** (double *DH_Parameters, double **q, double *Torque, ROBOT_DATA Robot)
- void **Elbow_Inverse_Dynamic** (double *DH_Parameters, double **q, double *Torque, ROBOT_DATA Robot, double dt)
- int **Two_Link_Set_Functions** ()
- double ** **Two_Link_Set_Inertia_Matrix** (double *DH_Parameters, double **q, ROBOT_DATA Robot)
- double ** **Two_Link_Set_Coriolis_Matrix** (double *DH_Parameters, double **q, ROBOT_DATA Robot)
- double * **Two_Link_Set_Gravitational_Vector** (double *DH_Parameters, double **q, ROBOT_DATA Robot)
- void **Two_Link_Direct_Dynamic** (double *DH_Parameters, double **q, double *Torque, ROBOT_DATA Robot)
- void **Two_Link_Inverse_Dynamic** (double *DH_Parameters, double **q, double *Torque, ROBOT_DATA Robot, double dt)
- int **Two_LinkMass_Set_Functions** ()
- double ** **Two_LinkMass_Set_Inertia_Matrix** (double *DH_Parameters, double **q, ROBOT_DATA Robot)
- double ** **Two_LinkMass_Set_Coriolis_Matrix** (double *DH_Parameters, double **q, ROBOT_DATA Robot)
- double * **Two_LinkMass_Set_Gravitational_Vector** (double *DH_Parameters, double **q, ROBOT_DATA Robot)
- void **Two_LinkMass_Direct_Dynamic** (double *DH_Parameters, double **q, double *Torque, ROBOT_↩ DATA Robot)
- void **Two_LinkMass_Inverse_Dynamic** (double *DH_Parameters, double **q, double *Torque, ROBOT_↩ DATA Robot, double dt)
- int **Elbow_3DOF_Set_Functions** ()

- double ** **Elbow_3DOF_Set_Inertia_Matrix** (double *DH_Parameters, double **q, ROBOT_DATA Robot)
- double ** **Elbow_3DOF_Set_Coriolis_Matrix** (double *DH_Parameters, double **q, ROBOT_DATA Robot)
- double * **Elbow_3DOF_Set_Gravitational_Vector** (double *DH_Parameters, double **q, ROBOT_DATA Robot)
- void **Elbow_3DOF_Direct_Dynamic** (double *DH_Parameters, double **q, double *Torque, ROBOT_DATA Robot)
- void **Elbow_3DOF_Inverse_Dynamic** (double *DH_Parameters, double **q, double *Torque, ROBOT_DATA Robot, double dt)
- int **Articulated_Set_Functions** ()
- double ** **Articulated_Set_Inertia_Matrix** (double *DH_Parameters, double **q, ROBOT_DATA Robot)
- double ** **Articulated_Set_Coriolis_Matrix** (double *DH_Parameters, double **q, ROBOT_DATA Robot)
- double * **Articulated_Set_Gravitational_Vector** (double *DH_Parameters, double **q, ROBOT_DATA Robot)
- void **Articulated_Direct_Dynamic** (double *DH_Parameters, double **q, double *Torque, ROBOT_DATA Robot)
- void **Articulated_Inverse_Dynamic** (double *DH_Parameters, double **q, double *Torque, ROBOT_DATA Robot, double dt)
- int **Anthro_Set_Functions** ()
- double ** **Anthro_Set_Inertia_Matrix** (double *DH_Parameters, double **q, ROBOT_DATA Robot)
- double ** **Anthro_Set_Coriolis_Matrix** (double *DH_Parameters, double **q, ROBOT_DATA Robot)
- double * **Anthro_Set_Gravitational_Vector** (double *DH_Parameters, double **q, ROBOT_DATA Robot)
- void **Anthro_Direct_Dynamic** (double *DH_Parameters, double **q, double *Torque, ROBOT_DATA Robot)
- void **Anthro_Inverse_Dynamic** (double *DH_Parameters, double **q, double *Torque, ROBOT_DATA Robot, double dt)

Variables

- double **(* **Serial_Get_Mass_Center_Position**)(int DoF, double *DH_Parameters, double *Theta)
- double **(* **Inertia_Matrix**)(double *DH_Parameters, double **q, ROBOT_DATA Robot)
- double **(* **Coriolis_Matrix**)(double *DH_Parameters, double **q, ROBOT_DATA Robot)
- double *(* **Gravitational_Vector**)(double *DH_Parameters, double **q, ROBOT_DATA Robot)
- void(* **Direct_Dynamic_Function**)(double *DH_Parameters, double **q, double *Torque, ROBOT_DATA Robot)
- void(* **Inverse_Dynamic_Function**)(double *DH_Parameters, double **q, double *Torque, ROBOT_DATA Robot, double dt)

5.3.1 Detailed Description

Function prototypes for the trajectory routines.

This contains the prototypes for the trajectory routine and eventually any macros,constants, or global variables needed.

Author

Salvador Botello-Aceves

Bug No known bugs.

5.3.2 Function Documentation

5.3.2.1 double** Serial_Set_Inertia_Matrix (int *DoF,* double * *DH_Parameters,* double * *Theta,* double *** *Inertia,* double * *Mass*)

Compute Matrix A

Compute Matrix C

5.4 Gauss.c File Reference

Implementation pseudorandom Gaussian distribution functions.

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include "Gauss.h"
#include "Random.h"
```

Macros

- #define **pi** 3.1416

Functions

- double **BoxMuller** ()
- void **BoxMuller_Vector** (int Dimension, double *Vector)
- double **Random_NormD** (double mean, double std_desv)
- void **Random_NormalDVector** (int Dimension, double *Mu, double **Covariance, double *Vector)
- double **get_Average** (int length, double *data)
- double **get_Variance** (int length, double *data)

5.4.1 Detailed Description

Implementation pseudorandom Gaussian distribution functions.

Implementation pseudorandom Gaussian distribution and statistics functions.

Author

Salvador Botello-Aceves

Bug No known bugs.

5.5 Gauss.h File Reference

Function prototypes for the pseudorandom Gaussian generators.

```
#include <time.h>
#include "Random.h"
```

Functions

- double **BoxMuller** ()
- void **BoxMuller_Vector** (int Dimension, double ∗Vector)
- double **Random_NormD** (double mean, double std_desv)
- void **Random_NormalDVector** (int Dimension, double ∗Mu, double ∗∗Covariance, double ∗Vector)
- double **get_Average** (int length, double ∗data)
- double **get_Variance** (int length, double ∗data)

Variables

- unsigned **x**

5.5.1 Detailed Description

Function prototypes for the pseudorandom Gaussian generators.

This contains the prototypes for the pseudorandom Gaussian generators and eventually any macros,constants, or global variables needed.

Author

Salvador Botello-Aceves

Bug No known bugs.

5.6 Interpolation_Algorithm.c File Reference

Implementation of Interpolation functions.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <math.h>
#include "Matrix.h"
#include "Interpolation_Algorithm.h"
```

Functions

- double * **Polinomial_Interpolation** (int length, double *x, double *y)
- double **Evaluate_Polinomial_Interpolation** (int length, double Value, double *DataA)
- void **Splines_C0** (DATA_INTER *Data)
- void **Splines_C1** (DATA_INTER *Data)
- void **Splines_C2** (DATA_INTER *Data)
- double * **Position_Splines_C0** (DATA_INTER Data, double value)
- double * **Position_Splines_C1** (DATA_INTER Data, double value)
- double * **Position_Splines_C2** (DATA_INTER Data, double value)
- double * **Velocity_Splines_C0** (DATA_INTER Data, double value)
- double * **Velocity_Splines_C1** (DATA_INTER Data, double value)
- double * **Velocity_Splines_C2** (DATA_INTER Data, double value)
- double * **Acceleration_Splines_C0** (DATA_INTER Data, double value)
- double * **Acceleration_Splines_C1** (DATA_INTER Data, double value)
- double * **Acceleration_Splines_C2** (DATA_INTER Data, double value)

5.6.1 Detailed Description

Implementation of Interpolation functions.

Implementation of Polynomial and Splines C0, C1, C2 interpolation functions. Position, Velocity and Acceleration functions are implemented for the C0, C1 and C2 interpolation.

Author

Salvador Botello-Aceves

Bug No known bugs.

5.7 Interpolation_Algorithm.h File Reference

Function prototypes for the Interpolation routines.

Classes

- struct DATA_INTER

Functions

- double * **Polinomial_Interpolation** (int length, double *x, double *y)
- double **Evaluate_Polinomial_Interpolation** (int length, double Value, double *DataA)
- void **Splines_C0** (DATA_INTER *Data)
- void **Splines_C1** (DATA_INTER *Data)
- void **Splines_C2** (DATA_INTER *Data)
- double * **Position_Splines_C0** (DATA_INTER Data, double value)
- double * **Position_Splines_C1** (DATA_INTER Data, double value)
- double * **Position_Splines_C2** (DATA_INTER Data, double value)
- double * **Velocity_Splines_C0** (DATA_INTER Data, double value)
- double * **Velocity_Splines_C1** (DATA_INTER Data, double value)
- double * **Velocity_Splines_C2** (DATA_INTER Data, double value)

Variables

- void(∗ **Interpolator_Function**)(DATA_INTER ∗Data)
- double ∗(∗ **Position_Interpolator_Function**)(DATA_INTER Data, double value)
- double ∗(∗ **Velocity_Interpolator_Function**)(DATA_INTER Data, double value)
- double ∗(∗ **Acceleration_Interpolator_Function**)(DATA_INTER Data, double value)

5.7.1 Detailed Description

Function prototypes for the Interpolation routines.

This contains the prototypes for the Interpolation routines and eventually any macros, constants, or global variables needed.

Author

Salvador Botello-Aceves

Bug No known bugs.

5.8 Kinematic_Mechanisms.h File Reference

Function prototypes for the trajectory routines.

Functions

- double ∗∗ **Transformation_Matrix** (double ∗DH_Parameter)
- double ∗∗ **Serial_Direct_Kinematic** (int DOF, double ∗∗DH_Parameters, double ∗∗Positions)
- void **Print_Manipulator_R** (int DOF, double ∗∗Position, double ∗lim, double Pause_Time)
- void **Serial_Inverse_Kinematic** (int DOF, double ∗DH_Parameters, double ∗Position, double ∗Theta)
- double ∗∗ **Anthro_Generate_DH_Table** (double ∗DH_Parameters, double ∗Theta)
- double ∗ **Anthro_Direct_Kinematic** (double ∗DH_Parameters, double ∗Theta)
- void **Anthro_Inverse_Kinematic** (double ∗DH_Parameters, double ∗Position, double ∗Theta)
- double ∗∗ **Anthro_Compute_Jacobian** (double ∗DH_Parameters, double ∗Thetas)
- void **Anthro_Snapshot_Architucture** (double ∗DH_Parameters, double ∗Position, double ∗Articulation, char ∗Name_picture)
- void **Anthro_Architucture_Plotter** (double ∗DH_Parameters, double ∗Position, double ∗Articulation, double alpha, FILE ∗pipeOut)
- double ∗∗ **Articulated_Generate_DH_Table** (double ∗DH_Parameters, double ∗Theta)
- double ∗ **Articulated_Direct_Kinematic** (double ∗DH_Parameters, double ∗Theta)
- void **Articulated_Inverse_Kinematic** (double ∗DH_Parameters, double ∗Position, double ∗Theta)
- double ∗∗ **Articulated_Compute_Jacobian** (double ∗DH_Parameters, double ∗Thetas)
- void **Articulated_Snapshot_Architucture** (double ∗DH_Parameters, double ∗Position, double ∗Articulation, char ∗Name_picture)
- void **Articulated_Architucture_Plotter** (double ∗DH_Parameters, double ∗Position, double ∗Articulation, double alpha, FILE ∗pipeOut)
- double ∗∗ **Elbow_Generate_DH_Table** (double ∗DH_Parameters, double ∗Theta)
- double ∗ **Elbow_Direct_Kinematic** (double ∗DH_Parameters, double ∗Theta)
- void **Elbow_Inverse_Kinematic** (double ∗DH_Parameters, double ∗Position, double ∗Theta)
- double ∗∗ **Elbow_Compute_Jacobian** (double ∗DH_Parameters, double ∗Thetas)

- void **Elbow_Snapshot_Architucture** (double *DH_Parameters, double *Position, double *Articulation, char *Name_picture)
- void **Elbow_Architucture_Plotter** (double *DH_Parameters, double *Position, double *Articulation, double alpha, FILE *pipeOut)
- double ** **Two_Link_Generate_DH_Table** (double *DH_Parameters, double *Theta)
- double * **Two_Link_Direct_Kinematic** (double *DH_Parameters, double *Theta)
- void **Two_Link_Inverse_Kinematic** (double *DH_Parameters, double *Position, double *Theta)
- double ** **Two_Link_Compute_Jacobian** (double *DH_Parameters, double *Theta)
- void **Two_Link_Snapshot_Architucture** (double *DH_Parameters, double *Position, double *Articulation, char *Name_picture)
- void **Two_Link_Architucture_Plotter** (double *DH_Parameters, double *Position, double *Articulation, double alpha, FILE *pipeOut)
- double ** **Two_LinkMass_Generate_DH_Table** (double *DH_Parameters, double *Theta)
- double * **Two_LinkMass_Direct_Kinematic** (double *DH_Parameters, double *Theta)
- void **Two_LinkMass_Inverse_Kinematic** (double *DH_Parameters, double *Position, double *Theta)
- double ** **Two_LinkMass_Compute_Jacobian** (double *DH_Parameters, double *Theta)
- void **Two_LinkMass_Snapshot_Architucture** (double *DH_Parameters, double *Position, double *Articulation, char *Name_picture)
- void **Two_LinkMass_Architucture_Plotter** (double *DH_Parameters, double *Position, double *Articulation, double alpha, FILE *pipeOut)
- double ** **Elbow_3DOF_Generate_DH_Table** (double *DH_Parameters, double *Theta)
- double * **Elbow_3DOF_Direct_Kinematic** (double *DH_Parameters, double *Theta)
- void **Elbow_3DOF_Inverse_Kinematic** (double *DH_Parameters, double *Position, double *Theta)
- double ** **Elbow_3DOF_Compute_Jacobian** (double *DH_Parameters, double *Theta)
- void **Elbow_3DOF_Snapshot_Architucture** (double *DH_Parameters, double *Position, double *Articulation, char *Name_picture)
- void **Elbow_3DOF_Architucture_Plotter** (double *DH_Parameters, double *Position, double *Articulation, double alpha, FILE *pipeOut)
- void **Delta_Inverse_Kinematic** (double *Delta_Parameters, double *Position, double *Theta)
- double ** **Delta_Compute_Jacobian** (double *Delta_Parameters, double *Theta)
- void **Delta_Snapshot_Architucture** (double *Delta_Parameters, double *Position, double *Articulation, char *Name_picture)
- void **Delta_Architucture_Plotter** (double *Delta_Parameters, double *Position, double *Articulation, double alpha, FILE *pipeOut)

Variables

- double **(* **Generate_DH_Table**)(double *DH_Parameters, double *Theta)
- double *(* **Direct_Kinematic_Function**)(double *Opt_Parameters, double *Parameter)
- void(* **Inverse_Kinematic_Function**)(double *Parameters, double *Position, double *Theta)
- double **(* **Jacobian_Function**)(double *Opt_Parameters, double *Parameter)
- void(* **Snapshot_Function**)(double *Parameters, double *Position, double *Articulation, char *Name_↩ picture)
- void(* **Plotter_Function**)(double *Parameters, double *Position, double *Articulation, double alpha, FILE *pipeOut)

5.8.1 Detailed Description

Function prototypes for the trajectory routines.

This contains the prototypes for the trajectory routine and eventually any macros,constants, or global variables needed.

Author

Salvador Botello-Aceves

Bug No known bugs.

5.9 Manipulability_Functions.h File Reference

Function prototypes for the trajectory routines.

Classes

- struct MANIP_DATA

Functions

- double ** **Level_Max_Work_Space_Limiter** (MANIP_DATA DATA, double *Opt_Parameters, double Level_z, double *Lim, int *counter)
- double ** **Level_Min_Work_Space_Limiter** (MANIP_DATA DATA, double *Opt_Parameters, double Level_z, double *Lim, int *counter)
- void **Total_Workspace_Visualization** (FILE *pipeOut, MANIP_DATA DATA, double *Opt_Parameters, double *Lim, double alpha)
- void **Workspace_Visualization** (MANIP_DATA DATA, double *Opt_Parameters, double *Lim, char *Filename)
- void **Regular_Workspace_Visualization** (double *Lim, double *Centroid, double Length, char *Filename)
- void **All_Workspace_Visualization** (MANIP_DATA DATA, double *Opt_Parameters, double *Lim, double *Centroid, double Length, char *Filename)
- double ** **Cubic_Work_Space_Disc** (double *Centroid, double length, int *Size)
- void **Cubic_Workspace_Visualization** (FILE *pipeOut, double *Lim, double *Centroid, double Length, double alpha)
- double * **Work_Space_Numeric_Centroid** (MANIP_DATA DATA, double *Opt_Parameters, double *Lim)
- double **Manipulability_Work_Space** (MANIP_DATA DATA, double *Opt_Parameters, double *Centroid)

Variables

- double **(* **Shape_Work_Space_Disc**)(double *Centroid, double length, int *Size)
- void(* **Shape_Workspace_Visualization**)(FILE *pipeOut, double *Lim, double *Centroid, double Length, double alpha)

5.9.1 Detailed Description

Function prototypes for the trajectory routines.

This contains the prototypes for the trajectory routine and eventually any macros,constants, or global variables needed.

Author

Salvador Botello-Aceves

Bug No known bugs.

5.10 Matrix.c File Reference

Implementation Vector-Matrix functions.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <math.h>
#include "defs_and_types.h"
#include "Matrix.h"
#include "SVD_Descomposition.h"
```

Functions

- double ** **create_Double_Matrix** (int rows, int cols)
- double ** **Identity_Double_Matrix** (int Dimension)
- int ** **create_Integer_Matrix** (int rows, int cols)
- int ** **Identity_Integer_Matrix** (int Dimension)
- char ** **create_Char_Matrix** (int rows, int cols)
- void **free_Double_Matrix** (int rows, double **Matrix)
- void **free_Inetger_Matrix** (int rows, int **Matrix)
- void **free_Char_Matrix** (int rows, char **Matrix)
- void **Copy_Double_Vector** (int length, double *Destiny, double *Sources)
- void **Copy_Double_Matrix** (int rows, int cols, double **Destiny, double **Sources)
- void **print_Double_Vector** (int length, double *Vector)
- void **print_Double_Matrix** (int rows, int cols, double **Matrix)
- double ** **Transpose_Matrix** (int rows, int cols, double **Matrix)
- void **Vector_Add_Vector** (int length, double *Source, double *Destiny)
- void **Vector_Subtract_Vector** (int length, double *Source, double *Destiny)
- void **Matrix_Multiplication_Scalar** (int rows, int cols, double **Matrix, double Scalar)
- double * Matrix_Vector_Multiplication (int rows, int cols, double **Matrix, double *Vector)
- double ** **Matrix_Matrix_Multiplication** (int rows, int cols, int intermedia, double **Matrix_1, double **Matrix_2)
- void **Matrix_Vector_Mult** (int rows, int cols, double **Matrix, double *Vector, double *Ans)
- void **Matrix_Matrix_Mult** (int rows, int cols, int intermedia, double **Matrix_1, double **Matrix_2, double **Ans)
- void **Matrix_Add_Matrix** (int rows, int cols, double **Source, double **Destiny)
- void **Matrix_Substract_Matrix** (int rows, int cols, double **Source, double **Destiny)
- void **Matrix_Complete_Symmetry** (int size, double **Matrix)
- double **Dot_Product** (int length, double *Vector_1, double *Vector_2)
- double **Norm_L2** (int length, double *Vector)
- double **Norm_Forbenius** (int rows, int cols, double **Matrix)
- double **Norm_One** (int rows, int cols, double **Matrix)
- double **Norm_Infinity** (int rows, int cols, double **Matrix)
- void **LU_Crout_Descomp** (int length, double **A)
- double * **LU_Crout_Solver** (int length, double **A, double *VO)
- void LU_Cholesky_Diagonal_Descomp (int length, double **A)
- double * LU_Cholesky_Diagonal_Solver (int length, double **A, double *VO)
- int **LU_Cholesky_Diagonal_Check** (int Dimension, double **LU, double min)
- int Check_PositiveDef_Matrix (int Dimension, double **Matrix, double min)
- void **Set_PositiveDef_Matrix** (int Dimension, double **Matrix)
- double ** **Inverse_Matrix** (int length, double **Matrix)

- double **Power_Method** (int length, double **Matrix, int Max_Iterations, double Tolerance, double *Eigenvector)
- double * **Power_Method_Def** (int length, int displacement, double **Matrix, int Max_Iterations, double Tolerance, double **Eigenvector)
- double **Inverse_Power_Method** (int length, double **Matrix, int Max_Iterations, double Tolerance, double *Eigenvector)
- double * **Inverse_Power_Method_Def** (int length, int displacement, double **Matrix, int Max_Iterations, double Tolerance, double **Eigenvector)
- double **Conditon_Number** (int length, double **Matrix)
- double **Inverse_Conditon_Number** (int length, double **Matrix)

5.10.1 Detailed Description

Implementation Vector-Matrix functions.

Implementation of Matrix and Vector operator as: Create, delete, dot-product, adding, subtracting, multiplying, norms, decomposition , inverse and linear equation solvers.

Author

Salvador Botello-Aceves

Bug No known bugs.

5.10.2 Function Documentation

5.10.2.1 int Check_PositiveDef_Matrix (int *Dimension,* double ** *Matrix,* double *min*)

RECIBE: Dimension del problema, Matriz a checar ENTREGA: Bandera de Matriz Definida Positiva

5.10.2.2 void LU_Cholesky_Diagonal_Descomp (int *length,* double ** *A*)

RECIBE: Tamaño, Matriz ENTREGA: –

5.10.2.3 double* LU_Cholesky_Diagonal_Solver (int *length,* double ** *A,* double * *VO*)

RECIBE: Tamaño, Matriz, Vector ENTREGA: Vector Solucion

5.10.2.4 double* Matrix_Vector_Multiplication (int *rows,* int *cols,* double ** *Matrix,* double * *Vector*)

RECIBE: Renglones, Columnas, Matriz, Vector ENTREGA: Vector resultante

5.11 Matrix.h File Reference

Function prototypes for the Vector-Matrix routines.

Functions

- double ** **create_Double_Matrix** (int rows, int cols)
- double ** **Identity_Double_Matrix** (int Dimension)
- int ** **create_Integer_Matrix** (int rows, int cols)
- int ** **Identity_Integer_Matrix** (int Dimension)
- char ** **create_Char_Matrix** (int rows, int cols)
- void **free_Double_Matrix** (int rows, double **Matrix)
- void **free_Inetger_Matrix** (int rows, int **Matrix)
- void **free_Char_Matrix** (int rows, char **Matrix)
- void **Copy_Double_Vector** (int length, double *Destiny, double *Sources)
- void **Copy_Double_Matrix** (int rows, int cols, double **Destiny, double **Sources)
- void **print_Double_Matrix** (int rows, int cols, double **Matrix)
- void **print_Double_Vector** (int length, double *Vector)
- double ** **Transpose_Matrix** (int rows, int cols, double **Matrix)
- void **Vector_Add_Vector** (int length, double *Source, double *Destiny)
- void **Vector_Subtract_Vector** (int length, double *Source, double *Destiny)
- void **Matrix_Multiplication_Scalar** (int rows, int cols, double **Matrix, double Scalar)
- double * Matrix_Vector_Multiplication (int rows, int cols, double **Matrix, double *Vector)
- double ** **Matrix_Matrix_Multiplication** (int rows, int cols, int intermedia, double **Matrix_1, double **Matrix_2)
- void **Matrix_Vector_Mult** (int rows, int cols, double **Matrix, double *Vector, double *Ans)
- void **Matrix_Matrix_Mult** (int rows, int cols, int intermedia, double **Matrix_1, double **Matrix_2, double **Ans)
- void **Matrix_Add_Matrix** (int rows, int cols, double **Source, double **Destiny)
- void **Matrix_Substract_Matrix** (int rows, int cols, double **Source, double **Destiny)
- void **Matrix_Complete_Symmetry** (int size, double **Matrix)
- double **Dot_Product** (int length, double *Vector_1, double *Vector_2)
- double **Norm_L2** (int length, double *Vector)
- double **Norm_Forbenius** (int rows, int cols, double **Matrix)
- double **Norm_One** (int rows, int cols, double **Matrix)
- double **Norm_Infinity** (int rows, int cols, double **Matrix)
- void LU_Cholesky_Diagonal_Descomp (int length, double **A)
- double * LU_Cholesky_Diagonal_Solver (int length, double **A, double *VO)
- void **LU_Crout_Descomp** (int length, double **A)
- double * **LU_Crout_Solver** (int length, double **A, double *VO)
- int **LU_Cholesky_Diagonal_Check** (int Dimension, double **LU, double min)
- int Check_PositiveDef_Matrix (int Dimension, double **Matrix, double min)
- void **Set_PositiveDef_Matrix** (int Dimension, double **Matrix)
- double ** **Inverse_Matrix** (int length, double **Matrix)
- double **Power_Method** (int length, double **Matrix, int Max_Iterations, double Tolerance, double *Eigenvector)
- double **Inverse_Power_Method** (int length, double **Matrix, int Max_Iterations, double Tolerance, double *Eigenvector)
- double * **Power_Method_Def** (int length, int displacement, double **Matrix, int Max_Iterations, double Tolerance, double **Eigenvector)
- double * **Inverse_Power_Method_Def** (int length, int displacement, double **Matrix, int Max_Iterations, double Tolerance, double **Eigenvector)
- double **Conditon_Number** (int length, double **Matrix)
- double **Inverse_Conditon_Number** (int length, double **Matrix)

5.11.1 Detailed Description

Function prototypes for the Vector-Matrix routines.

This contains the prototypes for the Vector-Matrix functions and eventually any macros, constants, or global variables needed.

Author

Salvador Botello-Aceves

Bug No known bugs.

5.11.2 Function Documentation

5.11.2.1 int Check_PositiveDef_Matrix (int *Dimension,* double ** *Matrix,* double *min*)

RECIBE: Dimension del problema, Matriz a checar ENTREGA: Bandera de Matriz Definida Positiva

5.11.2.2 void LU_Cholesky_Diagonal_Descomp (int *length,* double ** *A*)

RECIBE: Tamaño, Matriz ENTREGA: –

5.11.2.3 double* LU_Cholesky_Diagonal_Solver (int *length,* double ** *A,* double * *VO*)

RECIBE: Tamaño, Matriz, Vector ENTREGA: Vector Solucion

5.11.2.4 double* Matrix_Vector_Multiplication (int *rows,* int *cols,* double ** *Matrix,* double * *Vector*)

RECIBE: Renglones, Columnas, Matriz, Vector ENTREGA: Vector resultante

5.12 Numerical_Optimization.h File Reference

Function prototypes for the trajectory routines.

Classes

- struct OPT_PAR

Functions

- double * **Numerical_Gradient_Function** (OPT_PAR Parameters, double *Opt_Parameter)
- void **Check_MinimumLowering** (OPT_PAR Parameters, double *Opt_Parameter, double *alpha, double *Gradient, double *Direction)
- void **Line_Search_Methods** (OPT_PAR Parameters, double *Opt_Parameter)
- void **Newton_Method** (OPT_PAR Parameters, double *Opt_Parameter)

Variables

- double(∗ **Opt_Function**)(OPT_PAR Parameters, double ∗Opt_Parameter)
- double ∗(∗ **Gradient_Function**)(OPT_PAR Parameters, double ∗Opt_Parameter)
- double ∗∗(∗ **Hessian_Function**)(OPT_PAR Parameters, double ∗Opt_Parameter)

5.12.1 Detailed Description

Function prototypes for the trajectory routines.

This contains the prototypes for the trajectory routine and eventually any macros,constants, or global variables needed.

Author

Salvador Botello-Aceves

Bug No known bugs.

5.13 Objective_Functions.h File Reference

Function prototypes for the trajectory routines.

Functions

- double **Function_Link_Error** (double ∗Opt_Parameter, void ∗∗Parameter)
- double **Function_Int_Link_Error** (double ∗Opt_Parameter, void ∗∗Parameter)
- double **Function_Torque_Diff** (double ∗Opt_Parameter, void ∗∗Parameter)
- double **Function_Int_Torque** (double ∗Opt_Parameter, void ∗∗Parameter)
- double **Function_Link_Error_Control** (double ∗Opt_Parameter, void ∗∗Parameter)
- double **Function_Int_Link_Error_Control** (double ∗Opt_Parameter, void ∗∗Parameter)
- double **Function_Torque_Diff_Control** (double ∗Opt_Parameter, void ∗∗Parameter)
- double **Function_Int_Torque_Control** (double ∗Opt_Parameter, void ∗∗Parameter)
- double **Function_Simultaneous** (double ∗Opt_Parameter, void ∗∗Parameter)
- double **Function_Simultaneous_Control** (double ∗Opt_Parameter, void ∗∗Parameter)
- double **Function_Concurrent_Dynamic** (double ∗Opt_Parameter, void ∗∗Parameter)

Variables

- double(∗ **Fitness_Function**)(double ∗Opt_Parameter, void ∗∗Parameter)

5.13.1 Detailed Description

Function prototypes for the trajectory routines.

This contains the prototypes for the trajectory routine and eventually any macros,constants, or global variables needed.

Author

Salvador Botello-Aceves

Bug No known bugs.

5.14 OMNI.h File Reference

Function prototypes for the trajectory routines.

```
#include "OMNI/global.h"
#include "OMNI.h"
#include "BUMDA.h"
#include "CMA_ES.h"
#include "Optimizers.h"
```

Classes

- struct OMNI_DATA

Macros

- #define **INF** 1.0e14
- #define **EPS** 1.0e-10
- #define **E** 2.71828182845905
- #define **PI** 3.14159265358979
- #define **GNUPLOT_COMMAND** "gnuplot -persist"

Typedefs

- typedef struct OMNI_DATA **OMNI_DATA**

Functions

- void **OMNI_Init** (OPTIMIZATION_DATA problem, OMNI_DATA ∗evo)
- void **OMNI_Init_FILE** (OPTIMIZATION_DATA problem, OMNI_DATA ∗evo)
- void **OMNI_Input_Global** (OMNI_DATA ∗evo)
- void **OMNI_free_memory** (OMNI_DATA ∗evo)
- void **OMNI_evaluate_pop** (population ∗pop, void ∗∗Parameters)
- void **OMNI_evaluate_ind** (individual ∗ind, void ∗∗Parameters)
- void **Evolutive_OMNI** (OMNI_DATA ∗evo, void ∗∗Parameters, FILE ∗fileOut)

5.14.1 Detailed Description

Function prototypes for the trajectory routines.

This contains the prototypes for the trajectory routine and eventually any macros,constants, or global variables needed.

Author

Salvador Botello-Aceves

Bug No known bugs.

5.15 Optimizers.h File Reference

Function prototypes for the trajectory routines.

Classes

- struct OPTIMIZATION_DATA

Typedefs

- typedef struct OPTIMIZATION_DATA OPTIMIZATION_DATA

Functions

- void **Init_Optimizer** (OPTIMIZATION_DATA problem, void *OPTIMIZER_STRUCT)
- void **Run_Optimizer** (OPTIMIZATION_DATA problem, void *OPTIMIZER_STRUCT, void **Parameters, FILE *fileOut)
- void **Optimizer_Free** (OPTIMIZATION_DATA problem, void *OPTIMIZER_STRUCT)
- void **Set_Optimizer** (OPTIMIZATION_DATA *problem)
- double **Get** (OPTIMIZATION_DATA problem, void *OPTIMIZER_STRUCT, char *ptr)
- double * **Get_Ptr** (OPTIMIZATION_DATA problem, void *OPTIMIZER_STRUCT, char *ptr)

Variables

- void(* **Print_Function**)(double *Best_Fitness, void **Parameters, FILE *fileOut)

5.15.1 Detailed Description

Function prototypes for the trajectory routines.

This contains the prototypes for the trajectory routine and eventually any macros,constants, or global variables needed.

Author

Salvador Botello-Aceves

Bug No known bugs.

5.15.2 Typedef Documentation

5.15.2.1 typedef struct OPTIMIZATION_DATA OPTIMIZATION_DATA

Optimization Problem Data

5.16 R_pipeScript.c File Reference

Implementation of pipeline with Comprehensive R Archive Network.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
```

Functions

- FILE * **R_openPipe** ()
- void **R_closePipe** (FILE *pipe)
- void **R_Set_Library** (char *Lib, FILE *pipe)
- void **R_Set_Librarys** (int nLib, char **Lib, FILE *pipe)
- void **R_Set_Canvas** (double *Lim, char *xLabel, char *yLabel, char *main, FILE *pipe)
- void **R_Set_3dCanvas** (double *Lim, char *xLabel, char *yLabel, char *zLabel, char *main, FILE *pipe)
- void **R_plot_XY** (double x, double y, char *pch, char *color, FILE *pipe)
- void **R_plot_XYZ** (double x, double y, double z, char *pch, char *color, FILE *pipe)
- void **R_plot_Point** (double *point, char *pch, char *color, FILE *pipe)
- void **R_plot_3dPoint** (double *point, char *pch, char *color, FILE *pipe)
- void **R_plot_Set_Points** (int nPoints, double **Set, char *pch, char *color, FILE *pipe)
- void **R_plot_Set_Lines** (int nPoints, double **Set, char *Line_width, char *color, FILE *pipe)
- void **R_plot3d_Set_Points** (int nPoints, double **Set, char *alpha, char *pch, char *color, FILE *pipe)
- void **R_plot3d_Set_Lines** (int nPoints, double **Set, char *alpha, char *Line_width, char *color, FILE *pipe)
- void **R_sleepSystem** (FILE *pipe, double time)
- void **R_Set_viewSize3d** (int xPixel, int yPixel, FILE *pipe)
- void **R_Set_Figure** (char *Filename, double *size, FILE *pipe)
- void **R_Print_Figure** (FILE *pipe)
- void **R_Print_Figure3d** (char *Filename, FILE *pipe)
- void **R_Print_Snapshot3d** (char *Filename, char *fmt, FILE *pipe)
- void **R_Set_Legend** (char *Position, int size, char **legend, char **color, char **pch, char *cex, FILE *pipe)
- void **R_Set_EPS** (FILE *pipe)

5.16.1 Detailed Description

Implementation of pipeline with Comprehensive R Archive Network.

Implementation of the pipeline communication with Comprehensive R Archive Network, for data communication, illustrate and save 2D and 3D graphs, etc.

Author

Salvador Botello-Aceves

Bug No known bugs.

5.17 R_pipeScript.h File Reference

Function prototypes for the pipeline communication with R.

Functions

- FILE * **R_openPipe** ()
- void **R_closePipe** (FILE *pipe)
- void **R_Set_Library** (char *Lib, FILE *pipe)
- void **R_Set_Librarys** (int nLib, char **Lib, FILE *pipe)
- void **R_Set_Canvas** (double *Lim, char *xLabel, char *yLabel, char *main, FILE *pipe)
- void **R_Set_3dCanvas** (double *Lim, char *xLabel, char *yLabel, char *zLabel, char *main, FILE *pipe)
- void **R_plot_XY** (double x, double y, char *pch, char *color, FILE *pipe)
- void **R_plot_XYZ** (double x, double y, double z, char *pch, char *color, FILE *pipe)
- void **R_plot_Point** (double *point, char *pch, char *color, FILE *pipe)
- void **R_plot_3dPoint** (double *point, char *pch, char *color, FILE *pipe)
- void **R_plot_Set_Points** (int nPoints, double **Set, char *pch, char *color, FILE *pipe)
- void **R_plot_Set_Lines** (int nPoints, double **Set, char *Line_width, char *color, FILE *pipe)
- void **R_plot3d_Set_Points** (int nPoints, double **Set, char *alpha, char *pch, char *color, FILE *pipe)
- void **R_plot3d_Set_Lines** (int nPoints, double **Set, char *alpha, char *Line_width, char *color, FILE *pipe)
- void **R_sleepSystem** (FILE *pipe, double time)
- void **R_Set_viewSize3d** (int xPixel, int yPixel, FILE *pipe)
- void **R_Set_Figure** (char *Filename, double *size, FILE *pipe)
- void **R_Print_Figure** (FILE *pipe)
- void **R_Print_Figure3d** (char *Filename, FILE *pipe)
- void **R_Print_Snapshot3d** (char *Filename, char *fmt, FILE *pipe)
- void **R_Set_Legend** (char *Position, int size, char **legend, char **color, char **pch, char *cex, FILE *pipe)
- void **R_Set_EPS** (FILE *pipe)

5.17.1 Detailed Description

Function prototypes for the pipeline communication with R.

This contains the prototypes for the pipeline communication with R and eventually any macros,constants, or global variables needed.

Author

Salvador Botello-Aceves

Bug No known bugs.

5.18 Random.c File Reference

Implementation of pseudorandom numbers generators.

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include <time.h>
#include "Random.h"
```

Functions

- void **PrintStruct** (TRAND ∗Seed)
- void **Set_Seed** (TRAND ∗Seed, unsigned seed)
- unsigned **TausStep** (unsigned ∗z, int S1, int S2, int S3, unsigned M)
- unsigned **LCGStep** (unsigned ∗z, unsigned A, unsigned C)
- double **HybridTaus** (TRAND ∗Seed)

5.18.1 Detailed Description

Implementation of pseudorandom numbers generators.

Implementation of pseudorandom numbers generators, as: Taus Step, LCG Step and Hybrid Taus.

Author

Salvador Botello-Aceves

Bug No known bugs.

5.19 Random.h File Reference

Function prototypes for the pseudorandom numbers generators.

Classes

- struct TRAND

Functions

- void **PrintStruct** (TRAND ∗seed)
- void **Set_Seed** (TRAND ∗Seed, unsigned seed)
- unsigned **TausStep** (unsigned ∗z, int S1, int S2, int S3, unsigned M)
- unsigned **LCGStep** (unsigned ∗z, unsigned A, unsigned C)
- double **HybridTaus** (TRAND ∗Seed)

5.19.1 Detailed Description

Function prototypes for the pseudorandom numbers generators.

This contains the prototypes for the pseudorandom numbers generators and eventually any macros,constants, or global variables needed.

Author

Salvador Botello-Aceves

Bug No known bugs.

5.20 SVD_Descomposition.c File Reference

Implementation of SVD decomposition routine.

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include "defs_and_types.h"
```

Functions

- int **dsvd** (double **a, int m, int n, double *w, double **v)

5.20.1 Detailed Description

Implementation of SVD decomposition routine.

Implementation of SVD decomposition routine. Takes an mxn matrix a and decomposes it into udv, where u,v are left and right orthogonal transformation matrices, and d is a diagonal matrix of singular values.

This routine is adapted from svdecomp.c in XLISP-STAT 2.1 which is code from Numerical Recipes adapted by Luke Tierney and David Betz.

Author

-

Bug No known bugs.

5.21 SVD_Descomposition.h File Reference

Function prototypes for the SVD decomposition routine.

Functions

- int **dsvd** (double **a, int m, int n, double *w, double **v)

5.21.1 Detailed Description

Function prototypes for the SVD decomposition routine.

This contains the prototypes for the SVD decomposition routine and eventually any macros,constants, or global variables needed.

Author

Salvador Botello-Aceves

Bug No known bugs.

Index