

[Help](#)

```

#ifndef __STRUCTS_H__
#define __STRUCTS_H__

/*value of a polynomial function h between x1 and x2,
  y1=h(x1), y2=h(x2). degree is the degree of the polynom
  and a contains its coefficients
*/

/**
 * \ defgroup Polynomial Step Functions
 */
/*@{*/

typedef struct
{
    double          x1;/*!< l.h.s. of an interval */
    double          x2;/*!< r.h.s. of an interval */
    double          y1;/*!< y1=h(x1), where h is a polynomial function */
    double          y2;/*!< y2=h(x2), where h is a polynomial function */
    int             degree;/*!< degree of the polynom h */
    double          *a;/*!< coefficients of the polynom. Be
                           careful a[0] corresponds to the
                           coefficient of x^1 : h(0)=0 */
} step_element;

/*description of a function h on 'size' intervals. On each
  interval h is described with data, a step_element
*/

typedef struct
{
    int             size;/*!< number of intervals */
    step_element    *data;/*!< data[i] contains the
                           description of the polynomial
                           function on the ith interval*/
} step_fun;

```

```

/*@{*/

/**
 * \ addtogroup Polynomial Step Functions
 */
/*@{*/
step_fun      *copy_sf(const step_fun      *sf);

step_fun      *init_constant_sf(int          size,
    const double *x,
    const double *y);

step_fun      *init_cont_linear_sf(int      size,
    const double *x,
    const double *y);

step_fun      *integrate_sf(const step_fun      *sf);

double         compute_sf(const step_fun      *sf,
    double      x);

double         inverse_sf(const step_fun      *sf,
    double      y);

void           free_step_fun(step_fun      **sf);
/*@}*/

/** Grid : contains 2 arrays of size 'size'. The first array
    contains 'data' and the second one contains 'delta': for i>=1
    delta(i)=data(i)-data(i-1), delta(0)=delta(1)
 */

/**
 * \ defgroup Grid
 */
/*@{*/
typedef struct
{
    int          size; /*!< size of the arrays*/
    double       *data; /*!< array of datas*/
    double       *delta; /*!< array of data differences : delta(i)=data(i)

```

```
} grid;
/*@{*/

/**
 * \ addtogroup Grid
 */
/*@{*/
grid          *create_grid(int  n);
grid          *init_grid_cdo(int      n,
                               const double *x);
grid          *init_hom_grid(double  x0,
                               double  xn,
                               double  delta);
grid          *init_fine_grid(const grid *gd_init,
                              int      n);
void          free_grid(grid          *gd);
/*@{*/

#endif /* __STRUCTS_H__ */
```