

[Help](#)

```
#ifndef __finance_tool_box__
#define __finance_tool_box__
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include <math.h>
#include <assert.h>

#include "pnl/pnl_mathtools.h"
#include "pnl/pnl_vector.h"

extern double init_cond_with_dupire(const double x,
                                    const double S0,
                                    const double K0,
                                    const int dupire,
                                    const int product);

extern double init_cond(const double x,
                        const double S0,
                        const double K0,
                        const int product);

extern double bound_cond(const double x, const double S0,
                        const double K,
                        const double rebate, const double
                        barrier, const double ttm, const double r, const double div,
                        const int product, const int prod
                        uct_type);

typedef struct Option_Eqd
{
    int am; // 0 european 1 american
    int product;
    // 1 - Call ; 2 - Put ; 3 - forward
    int product_type;
    // 1- Vanilla; 2 Up-and-Out ;3 Down-and-Out ; 4 Double
    barrier ;5
    // parisian; 6 varswap ; 7 Forward start
    double S0;
```

```
double K;
double T;
double t_start;
double rebate;
double barrier;
double price;
double delta;
double implied_vol;

// Not really in option_eqd,
double rate;
double divid;

} Option_Eqd;

// 1 - Call ; 2 - Put ; 3 - forward
// 1- European ; 2 Up-and-Out ;3 Down-and-Out ; 4 Double
// barrier; 5 parisain
extern Option_Eqd *option_eqd_create(int am, int product_,
                                     int product_type_,
                                     double S0_,
                                     double K_, double T_,
                                     double rebate_,
                                     double barrier_);
extern Option_Eqd *option_eqd_create_forwardstart(int am,
int product_,
int product_type_, double S0_,
double K_, double T_,
double t_start_,
double rebate_,
double barrier_);
extern void option_eqd_set_rate(Option_Eqd *opt, double ra
te_, double divid_);
extern double option_eqd_init_cond(const Option_Eqd *Op,
const double x);
extern double option_eqd_bound_cond(const Option_Eqd *Op,
const double x,
double ttm);

extern int option_eqd_compute_implied_vol(Option_Eqd *op);
```

```

extern double Double_Primitive_Call_Put(const double K,
    const double S0, const double x, const int is_call);
extern double Compute_Projection_U0(const double K, const
    double S0, const double x, const double h);

typedef struct List_Option_Eqd
{
    int am; // 0 european 1 american
    PnlVectInt *product;
    // 1 - Call ; 2 - Put ; 3 - forward
    int product_type;
    // 1- European ; 2 Up-and-Out ;3 Down-and-Out ; 4
    Double barrier ;5
    // parisian ; 7 forward start options
    double S0;
    double rebate;
    int nb_maturity; //number of option_eqd maturity in the
        list
    int nb_options ; //number of option_eqd in the list
    PnlVectInt *index_maturity;
    PnlVect *K;
    PnlVect *T;
    PnlVect *t_start;
    PnlVect *price;
    PnlVect *implied_vol;

    // Not really in option_eqd,
    double rate;
    double divid;

} List_Option_Eqd;

extern List_Option_Eqd *list_option_eqd_create(int am_,
    double S0_);
extern List_Option_Eqd *list_option_eqd_create_with_data(
    int am_, double S0_, PnlVectInt *product_, PnlVectInt *index_
    matu, PnlVect *Matu, PnlVect *Strike);
extern List_Option_Eqd *list_option_eqd_create_forwardsta
    rt_with_data(int am_, double S0_, PnlVectInt *product_, PnlV
    ectInt *index_matu, PnlVect *Matu, PnlVect *Start_Date, Pn

```

```
    lVect *Strike);  
extern List_Option_Eqd *list_option_eqd_copy(const List_  
    Option_Eqd *op_in);  
  
extern void list_option_eqd_set_rate(List_Option_Eqd *lopt,  
    double rate_, double divid_);  
  
extern Option_Eqd list_option_eqd_get_value(List_Option_Eqd  
    *lopt, int it, int k);  
  
extern void list_option_eqd_free(List_Option_Eqd **op);  
extern void list_option_eqd_readmarketdata(List_Option_Eqd  
    *op, const char *file);  
extern void list_option_eqd_savemarketdata(List_Option_Eqd  
    *op, const char *file);  
extern void list_option_eqd_print(List_Option_Eqd *op);  
extern void list_option_eqd_print_nsp(List_Option_Eqd *op);  
extern int list_option_eqd_compute_implied_vol(List_Option_  
    Eqd *op);  
  
#endif
```

## References