

## [Help](#)

```
#if !defined(DEFINEFILE)
#define DEFINEFILE

/* General Purpose macros */

/* Numerical Recipes p.942 */
#define NR_END 1
#define FREE_ARG char*
#define SWAP(a,b) tempr=(a);(a)=(b);(b)=tempr
#define M_PI2 6.28318530717959

#define KRO(i,j) ((i)==(j) ? 1:0)
#define TIMER_DEF      struct timeval temp_1, temp_2
#define TIMER_START    gettimeofday(&temp_1, (struct timezone*)0)
#define TIMER_STOP     gettimeofday(&temp_2, (struct timezone*)0)
#define TIMER_ELAPSED  ((temp_2.tv_sec-temp_1.tv_sec)+(temp_2.tv_usec-temp_1.tv_u

#define TNCONST 14
#define TNDELTA 0.0001
#define DIM1 1
#define DIM2 2
#define TEST1 1
#define TEST2 2

#define MAXINPUTLINE 150
#define MAXFILELENGTH 80

#define NARGS 1
#define OPTSTRING "hx:"

#define REAL double
#define RETURNOK 0

#define EXA 0
#define EXB 1

/* integral approximation methods */
#define TR 1
#define SIMP 2
```

```

#define NC4 3
#define NC6 4

/* jump density function */
#define GAUSS 1

/* problem parameter */
typedef struct
{
    double s;
    double K;
    double T;
    double r;
    double sigma;
    double divid;
    double lambda;
    double Eu;
} PARAM;

/* jump parameter */
typedef struct
{
    double par1;
    double par2;
    double Eu;
    double zmin;
    double zmax;
    double *d;
} DENSITY;

/* space and time mesh discratisation */
typedef struct
{
    double xmin;
    double xmax;
    double h; /* space step */
    double k; /* time step */
    int N; /* space nodes */
    int M; /* time nodes */
    int Index; /* price point index */

```

```

    double upwind_alphacoef;
} MESH;

```

```

typedef struct
{
    double p1;
    double p2;
    double p3;
} WEIGHT;

```

```

typedef struct
{
    int min;
    int max;
    int N;
} IMESH;

```

```

typedef struct
{
    double dif;
    double adv;
    double lin;
} EQ;

```

```

int Gaussian_data(double mu, double gamma2, DENSITY *g);
int Gaussian_vect(int l, int u, double h, DENSITY *g);
void freeDensity(DENSITY *g);
int set_parameter(double s, double K, double t, double r, double sigma, double d);
int equation(PARAM p, EQ *eq);
int mesh(double T, EQ eq, int N, MESH *m);
int set_mesh(EQ eq, int N, MESH *m);
int set_weights_espl(double T, EQ eq, MESH *m, WEIGHT *w);
int set_weights_impl(int M, double T, EQ eq, MESH *m, WEIGHT *w);
int initgrid_1Dbis(PARAM p, DENSITY g, EQ eq, int N, MESH *m, IMESH *Im);
int set_boundaryAA(int bound, MESH m, PARAM p, IMESH Im, double *in, double *out);
int tridiagsystem(double *a, double *b, double *c, double *r, double *u, int n);
int tridiag_bis(double *a, double *b, double *c, double *r, double *u, unsigned
double calc_int(int nodes, double *weight, double *pu);

```

```

int d1_intcomp(int nodes, double h, double *weight, double *jdensity, int int_me
int int_trapez(REAL h, REAL *d, REAL *p);
int int_simp(REAL h, REAL *d, REAL *p);
int int_nc4(REAL h, REAL *d, REAL *p);
int int_nc6(REAL h, REAL *d, REAL *p);

void dfour1(double *data, unsigned long nn, int isign);
void drealf1(double *data, unsigned long n, int isign);
/*
    void dconvlv (double *data, unsigned long n, double *respns, unsigned long m,
    int dcorrel(double *data1, double *data2, unsigned long n, double *ans);
*/
#endif

```