

[Help](#)

```
#include <stdlib.h>
#include "copulas.h"

typedef struct {
    double          rho;
    double          g_rho;
    double          u_rho;
    double          factor;
    double          t1;
    double          t2;
    double*         tab1 ;
    double*         tab2 ;
} double_t_params;

/* Initiaalaiston des points sur la grille */

static double *init_points(copula *cop){
    int i;
    double a1,b1;
    int n=2*cop->size+100;
    double *tab;
    double_t_params *p;
    p=cop->parameters;
    tab=malloc(n*sizeof(double));

    a1=-6*sqrt((p->t1-2)/p->t1)*(p->rho)-6*sqrt((p->t2-2)/p->
        t2)*sqrt(1-(p->rho)*(p->rho));
    b1=6*sqrt((p->t1-2)/p->t1)*(p->rho)+6*sqrt((p->t2-2)/p->
        t2)*sqrt(1-(p->rho)*(p->rho));

    for(i=0;i<n;i++){
        tab[i]=a1+i*(b1-a1)*1./(n-1);
    }
    return tab;
}
```

```

static double f1(double rho,double t1,double x1){
    double u=exp((t1+1)*0.5*log(1+x1*x1/(rho*rho*(t1-2))));
    return 1/(u);
}

static double f2(double rho,double t2,double x2){
    double u=exp((t2+1)*0.5*log(1+x2*x2/((1-rho*rho)*(t2-2)))
    );
    return 1/(u);
}

double *init_cdf(copula *cop)
{
    double_t_params *p;
    int l;
    int n=0;
    double fval1,h1,x1;
    double a1=0;
    double b1=0;
    double a2=0;
    double pi=3.14159265;
    double s1=0;
    double *s2;
    double k;
    int i,j;
    double coefs=0;

    p=cop->parameters;
    n=2*cop->size+100;
    a1=-6*(p->rho)*sqrt((p->t1-2)/p->t1);
    b1=6*(p->rho)*sqrt((p->t1-2)/p->t1);
    a2=-6*sqrt(1-(p->rho)*(p->rho))*sqrt((p->t2-2)/p->t2);
    s2=malloc(n*sizeof(double));
    coefs=tgamma((p->t1+1)*0.5)*tgamma((p->t2+1)*0.5)/((tgam
        ma(p->t1*0.5))*(tgamma(p->t2*0.5))*pi*(p->rho)*sqrt((1-p->rh
        o*p->rho)*(p->t1-2)*(p->t2-2)));
    k=(b1-a1)*1./n;

    for(l=0;l<n;l++){
        s2[l]=0;
    }
}

```

```

    for(i=0;i<n;i++){
        s1=0;
        x1=p->tab1[1]-(a1+k*i);
        if(x1>a2){
            h1=(x1-a2)*1./n;
            for(j=0;j<n;j++){

                fval1=f2(p->rho,p->t2,a2+h1*j);

                s1+=fval1;
            }
            s1=s1*h1;
            s2[1]+=s1*coefs*k*f1(p->rho,p->t1,k*i+a1);

        }
    }
}

return s2;
}

double double_t_cdf( const copula *cop,double x)
{
    int n=0;
    int i=0;
    double result;
    double_t_params *p;
    n=2*cop->size+100;
    p=cop->parameters;

    if(p->tab1[0]>=x) return 0.00000;
    else if (p->tab1[n-1]<=x) return p->tab2[n-1];

    do{
        i=i+1;
    }while(p->tab1[i]<x);

    result=((p->tab2[i]-p->tab2[i-1])/(p->tab1[i]-p->tab1[i-1]
    ))*x +(p->tab1[i]*p->tab2[i-1]-p->tab1[i-1]*p->tab2[i])/(
    p->tab1[i]-p->tab1[i-1]);

```

```

    return result;
}

static double double_t_inv_cdf(const copula *cop,double x)
{
    int n=0;
    int i;
    double_t_params *p;
    int a;

    n=2*cop->size+100;
    p=cop->parameters;
    a=0;

    if(x==1) x=1-0.0001;

    if(x==0){
        do{
            a=a+1;
        }while(p->tab2[a]==0);

        return p->tab1[a];
    }

    i=1;
    a=0;

    if ((x<0)|| (x>1)) return 0;

    if(p->tab2[0]>=x) return p->tab1[0];
    else if (x>p->tab2[n-1]) return p->tab1[n-1];

    do{
        i=i+1;
    }while(p->tab2[i]<x);
    a=i-1;

    return p->tab1[a] +((x-p->tab2[a])*(p->tab1[a+1]-p->tab1[

```

```

        a]))/((p->tab2[a+1]-p->tab2[a]));
    }

static void double_t_generate(copula *cop)
{ double_t_params *p;
  p=cop->parameters;
  ((double_t_params *)cop->parameters)->factor = simulate_      student(p->t1);
}

static double  student_density(const copula *cop, double x)
{
  double_t_params *p;
  p=cop->parameters;
  return (tgamma((p->t1+1)*0.5)/((tgamma((p->t1)*0.5))*sq
    rt(M_PI*(p->t1))*exp((((p->t1)+1)*0.5)*log(1+x*x/(p->t1)))))
    ;
}

static double  double_t_density(const copula *cop, double x
){

  double eps=0.00001;

  return (double_t_cdf(cop,x+eps)-double_t_cdf(cop,x-eps))/
    (2*eps);

}

static double *double_t_compute_prob(const copula *cop,
  double f_t)
{
  double          *result;
  double_t_params *p;
  double          a;
  double          b;
  int             i;
  p = cop->parameters;
  result = malloc((cop->size)* sizeof(double));

  a=double_t_inv_cdf(cop,f_t);

```

```

b=sqrt(p->t2/(p->t2-2))*1./p->g_rho;

for (i = 0; i < cop->size; i++) {

    result[i] = student_cdf(p->t2,b*(a-p->rho*sqrt((p->t1-2
    )/(p->t1))*cop->points[i]));

}

return (result);
}

static int double_t_compute_dt(const copula *cop,const step
    _fun *H, double *time){
    double_t_params *p;
    double X;
    double zi;
    p = cop->parameters;
    X = (p->rho * p->factor)*sqrt((p->t1-2)/p->t1) +( p->g_rho
        o * simulate_student(p->t2))*sqrt((p->t2-2)/p->t2) ;
    zi = -log(1. - double_t_cdf(cop,X));
    if (zi >= H->data[H->size-1].y2) return ( 0 );
    else {
        *time = inverse_sf(H, zi);
        return ( 1 );
    }
}

copula *init_double_t_copula(const double
    rho,const double t1,const double t2)
{
    copula *cop;
    double_t_params *p;
    double h;
    double v0;
    int jv;
    int b;
    double x;
    cop = malloc(sizeof(copula));
    cop->name = "One-factor Double_T copula ";
    cop->nfactor = 1;

```

```
p = malloc(sizeof(double_t_params));
cop->parameters = p;
p->rho = rho;
p->g_rho = sqrt(1.0 - rho*rho);
p->u_rho = rho / p->g_rho;
p->t1=t1;
p->t2=t2;

cop->size = 200;
p->tab1=init_points(cop);
p->tab2=init_cdf(cop);

b=(cop->size);
cop->points = malloc(b*sizeof(double));
cop->weights = malloc(b*sizeof(double));

x=6;

h = 2*x/ (cop->size-1);
for (jv = 0, v0 = -x; jv < cop->size; jv++, v0 += h) {
    cop->points[jv] = v0;
    cop->weights[jv] = student\_density(cop,v0) * h;
}
cop->density = double_t_density;
cop->generate = double_t_generate;
cop->compute_default_time = double_t_compute_dt;
cop->compute_cond_prob = double_t_compute_prob;

return (cop);
}
```

References