```
   Help
#include  "bs2d_std2d.h"
#include "error_msg.h"


static int VFExplicit(int am,double s1,double s2,NumFunc_2
     *payoff,double K,double t,double r,double divid1,double
    divid2,double sigma1,double sigma2,double rho,int N, double
    *ptprice,double *ptdelta1,double *ptdelta2)
{
  int i,j,I,II1,II2,JJ2;
  double temps,hg,ee,l,la,lb,w,p,k1,k2,t1,t2,bb1,bb2,bb3,bb
    4,bb5,aa,bb,
    c0,c00,c1,c2,c3,aaa,bbb,ccc,ddd,eee,fff,ggg,hhh;
  double *u,*uap,*psi;
  double u1,u2,loc,sig1,sig2;
  double a2,c,s,b1,b2,b3,v1,v2,h,g,dt,alpha,c22,theta,k3;
  double delta1,delta2;

  /*Memory Allocation*/
  u=(double *)calloc(N*N,sizeof(double));
  if (u==NULL)
    return MEMORY_ALLOCATION_FAILURE;
  uap=(double *)calloc(N*N,sizeof(double));
  if (uap==NULL)
    return MEMORY_ALLOCATION_FAILURE;
  psi=(double *)calloc(N*N,sizeof(double));
  if (psi==NULL)
    return MEMORY_ALLOCATION_FAILURE;

  /* Initialisations */
  sig1=sigma1*sigma1;
  sig2=sigma2*sigma2;
  u1=log(s1);
  u2=log(s2);


  /*Space Localization*/
  p=0.2316419;
  bb1=0.319381530;
  bb2=-0.356563782;
```

```
bb3=1.781477937;
bb4=-1.821255978;
bb5=1.330274429;

la=0;
lb=10*K;
w=0;
for (j=0;j<20;j++)
  {
    for (i=0;i<200;i++)
{
  /*Homogeneous Discretization of [la,lb]*/
  l=la+i*(lb-la)*0.005;

  k1= ( l-fabs(t*(r-divid1-0.5*sigma1*sigma1)))/( sqrt(
  t)*sigma1);
  k2= ( l-fabs(t*(r-divid2-0.5*sigma2*sigma2)))/( sqrt(
  t)*sigma2);

  t1=1/(1+p*k1);
  t2=1/(1+p*k2);




  /*Put Minimum*/
  if((payoff->Compute)==&PutMin)
    k3=4*K;
  /*Call Maximum*/
  else if((payoff->Compute)==&CallMax)
    k3=1000*64*( exp(u1+fabs(r-divid1)*t +sig1*t)+ exp(
  u2+fabs(r-divid2)*t +sig2*t))*
      ( exp(u1+fabs(r-divid1)*t +sig1*t)+ exp(u2+fabs(r-
  divid2)*t +sig2*t));
  /*Exchange option*/
  else /*if((payoff->Compute)==&Exchange)*/
    k3=1000*64*exp(2*u1+2*fabs(r-divid1)*t +2*sig1*t);


  /*Precision of 1/1000*/
  w = 1/(2.506628274631)*
```

```
    (exp(-0.5*k1*k1)*
     (bb1*t1+bb2*t1*t1+bb3*t1*t1*t1+bb4*t1*t1*t1*t1+bb5*
  t1*t1*t1*t1*t1) +
     exp(-0.5*k2*k2)*
     (bb1*t2+bb2*t2*t2+bb3*t2*t2*t2+bb4*t2*t2*t2*t2+bb5*
  t2*t2*t2*t2*t2))-0.001/k3;


  if(w<0) {
    break;      }
}
    /*[la,lb] contains the solution w(l)=0 */
    la=l-(lb-la)*0.005;
    lb=l;
    /*Frame of the solution*/
    if(lb-la<0.00001) break;
  }
loc=l;


/* Rotation */

if (sig1==sig2) {

  theta = atan(1);
  alpha=sig1*0.5*(1+rho);
  a2=sig2*0.5*(1-rho);

  c = cos(theta);
  s = sin(theta);

  b1 = (r - divid1 - sig1*0.5)*c +  (r - divid2 - sig2*0.
  5)*s;
  b2 =  (r - divid2 - sig2*0.5)*c -  (r-divid1 - sig1*0.5
  )*s ;
}else{
  theta = atan(2*rho*sigma1*sigma2/(sig1-sig2))/2;

  c = cos(theta);
  c2 = cos(2*theta);
  s = sin(theta);
```

```
  alpha=((sig1+sig2)*c2+sig1-sig2)/(4*c2);
  a2= ((sig1+sig2)*c2+sig2-sig1)/(4*c2);
  b1 = (r - divid1 - sig1*0.5)*c +  (r - divid2 - sig2*0.
  5)*s;
  b2 =  (r - divid2 - sig2*0.5)*c -  (r-divid1 - sig1*0.5
  )*s ;

}

/*homothetie*/

b3 = sqrt(alpha/a2);

/*Space steps*/
h = (double)(2.0*loc/((double)1.0*N));
g = (double)(2.0*b3*loc/((double)1.0*N) );
hg = h*g;


/*Localization after rotation and homothetie*/

v1=u1*c+u2*s;
v2=(u2*c-u1*s)*b3;


/*Stability Condition*/

/*  if(2*a1 < (h,g)*MAX(fabs(b1),fabs(b2*b3)))  */
/*Pechlet Condition*/
if(2*alpha - MAX(h*fabs(b1),g*fabs(b2*b3))<0)
  {
    /*Upwind Scheme*/

    /*Stability Condition Time Step*/
    dt=hg/(r*hg+3*alpha*(h/g+g/h)+g*fabs(b1)+h*fabs(b2*b3
  ));
    /*Constants*/
    c0 = alpha*dt/(h*h);
    c00 = alpha*dt/(g*g);
    c1 = b1*dt/h;
```

```
    c22 = b2*b3*dt/g;
    c3 = r*dt;
    aaa=1-2*(c0+c00)-c3-(fabs(c1)+fabs(c22));
    bbb=c0+MAX(-c1,0);
    ccc=c0+MAX(c1,0);
    ddd=c00+MAX(-c22,0);
    eee=c00+MAX(c22,0);
    fff=aaa-c00;
    ggg=aaa-c0;
    hhh=aaa-c0-c00;


  }
else
  {

    /*Central Scheme*/

    /*Stability Condition Time Step*/

    dt= hg/(0.5*r*hg+(h/g+g/h)*2*alpha+0.5*(g*fabs(b1)+h*
  fabs(b2*b3)) );
    /*Constants*/
    c0 = alpha*dt/(h*h);
    c00 = alpha*dt/(g*g);
    c1 = 0.5*b1*dt/h;
    c22 = 0.5*b2*b3*dt/g;
    c3 = r*dt;
    aaa=1-2*(c0+c00)-c3;
    bbb=c0-c1;
    ccc=c0+c1;/*Deltas*/
    ddd=c00-c22;
    eee=c00+c22;
    fff=1-2*c0-3*c00-c3;
    ggg=1-3*c0-2*c00-c3;
    hhh=1-3*(c0+c00)-c3;

  }


/*Maturity Conditions*/
```

```
ee = N*0.5;
for(j=0;j<N;j++)
  {
    for (i=0;i<N;i++)
{
  I = i*N+j;

  aa=v1+i*h-h*ee;
  bb=v2+j*g-g*ee;
  psi[I] =(payoff->Compute)(payoff->Par,exp(aa*c-bb*s/b3
  ),exp(aa*s+bb*c/b3));
  /*psi[I] = Payoff(aa*c-bb*s/b3,aa*s+bb*c/b3,K,mu);*/

  uap[I] = psi[I];
  u[I] = psi[I];


}
  }


/*Explicit Finite Difference Cycle*/
temps = 0;
while (temps<t)
  {
    temps += dt;
    for (i=1;i<N-1;i++)
{
  for (j=1;j<N-1;j++)
    {
      I=i*N+j;
      u[I] = uap[I]*aaa+uap[I-N]*bbb+uap[I+N]*ccc+
  uap[I-1]*ddd+uap[I+1]*eee;

    }
}


    /*Homogeneous Dirichlet Conditions*/
    for (i=1;i<N-1;i++)
{
```

```
  II1=i*N;
  II2=i*N+(N-1);
  JJ2=(N-1)*N + i;

  u[II1] = uap[II1]*fff + uap[II1-N]*bbb + uap[II1+N]*cc
  c+
    uap[II1+1]*eee;

  u[II2]=uap[II2]*fff+uap[II2-N]*bbb+ uap[II2+N]*ccc+ua
  p[II2-1]*ddd;

  u[i] = uap[i]*ggg+uap[i+N]*ccc+uap[i-1]*ddd+uap[i+1]*
  eee;

  u[JJ2] = uap[JJ2]*ggg+uap[JJ2-N]*bbb+uap[JJ2-1]*ddd+ua
  p[JJ2+1]*eee;

}

    u[0] = uap[0]*hhh+uap[N]*ccc+uap[1]*eee;

    u[N-1] = uap[N-1]*hhh+uap[N+N-1]*ccc+
uap[N-2]*ddd;

    u[(N-1)*N+(N-1)]=uap[(N-1)*N + (N-1)]*
hhh+uap[(N-2)*N + (N-1)] *bbb+
uap[(N-1)*N + (N-2)]*ddd;

    u[(N-1)*N]= uap[(N-1)*N]*hhh+
uap[(N-2)*N]*bbb+uap[(N-1)*N+1]*eee;


    for (i=0;i<N;i++)
{
  for (j=0;j<N;j++)
    {
      I = i*N+j;
      /*Splitting for the american case*/
      uap[I] = MAX(u[I],psi[I]);
```

```
      }
  }
    }

  /*Price*/
  i=  ((int)(N/2))*N+(int)(N/2);
  *ptprice=uap[i];

  /*Deltas*/
  delta1=(uap[i+N]-uap[i-N])/(2*h);
  delta2=(uap[i+1]-uap[i-1])/(2*g);
  *ptdelta1=(delta1*c-delta2*s*b3)*exp(-u1);
  *ptdelta2=(delta2*b3*c+delta1*s)*exp(-u2);

  /*Memory desallocation*/
  free(u);
  free(uap);
  free(psi);

  return OK;
}

int CALC(FD_VFExplicit)(void *Opt,void *Mod,PricingMethod *
    Met)
{
  TYPEOPT* ptOpt=(TYPEOPT*)Opt;
  TYPEMOD* ptMod=(TYPEMOD*)Mod;
  double r,divid1,divid2;

  r=log(1.+ptMod->R.Val.V_DOUBLE/100.);
  divid1=log(1.+ptMod->Divid1.Val.V_DOUBLE/100.);
  divid2=log(1.+ptMod->Divid2.Val.V_DOUBLE/100.);

  return VFExplicit(ptOpt->EuOrAm.Val.V_BOOL,ptMod->S01.Val
    .V_PDOUBLE,
        ptMod->S02.Val.V_PDOUBLE,ptOpt->PayOff.Val.V_
    NUMFUNC_2,(ptOpt->PayOff.Val.V_NUMFUNC_2)->Par[0].Val.V_PDOUBLE,
        ptOpt->Maturity.Val.V_DATE-ptMod->T.Val.V_DATE,
        r,divid1,divid2,ptMod->Sigma1.Val.V_PDOUBLE,pt
    Mod->Sigma2.Val.V_PDOUBLE,ptMod->Rho.Val.V_RGDOUBLE,
        Met->Par[0].Val.V_INT,
```

```
        &(Met->Res[0].Val.V_DOUBLE),&(Met->Res[1].Val.V_
    DOUBLE),&(Met->Res[2].Val.V_DOUBLE) );
}

static int CHK_OPT(FD_VFExplicit)(void *Opt, void *Mod)
{

  if ( (strcmp(((Option*)Opt)->Name,"PutMinimumAmer")==0)
       || (strcmp( ((Option*)Opt)->Name,"CallMaximumAmer")=
    =0)
       ||(strcmp(((Option*)Opt)->Name,"ExchangeAmer")==0) )
    return OK;
  return WRONG;

}

static int MET(Init)(PricingMethod *Met,Option *Opt)
{
  if ( Met->init == 0)
    {
      Met->init=1;

      Met->Par[0].Val.V_INT2=100;


    }

  return OK;
}

PricingMethod MET(FD_VFExplicit)=
{
  "FD_FVExpl",
  {{"StepNumber",INT2,{100},ALLOW},{" ",PREMIA_NULLTYPE,{0}
    ,FORBID}},
  CALC(FD_VFExplicit),
  {{"Price",DOUBLE,{100},FORBID},{"Delta1",DOUBLE,{100},FO
    RBID} ,
   {"Delta2",DOUBLE,{100},FORBID} ,
   {" ",PREMIA_NULLTYPE,{0},FORBID}},
  CHK_OPT(FD_VFExplicit),
```

```
  CHK_ok,
  MET(Init)
};
```

References