Help

```
#include "cgmy1d_pad.h"
#include "pnl/pnl_fft.h"
#include "pnl/pnl_complex.h"
#include "pnl/pnl_vector.h"

#if defined(PremiaCurrentVersion) && PremiaCurrentVersion <
     (2011+2) //The "#else" part of the code will be freely av
    ailable after the (year of creation of this file + 2)
static int CHK_OPT(FFT_CGMY_FloatingLookback)(void *Opt,
    void *Mod)
{
  return NONACTIVE;
}
int CALC(FFT_CGMY_FloatingLookback)(void*Opt,void *Mod,
    PricingMethod *Met)
{
  return AVAILABLE_IN_FULL_PREMIA;
}
#else

// "resultat takes the product of a circulant matrix M(c)
    and a vextor x
static void circulante (PnlVectComplex *resultat, PnlVectC
    omplex *c, PnlVectComplex *x)
{
  PnlVectComplex *temp;
  int i,n;
  n = x->size;
  temp=pnl_vect_complex_create(n);

  pnl_fft (c, temp);
  pnl_fft (x,resultat);
  for (i=0;i<n;i++)
  {
    pnl_vect_complex_set (temp,i,Cmul (pnl_vect_complex_get
    (temp,i), pnl_vect_complex_get(resultat,i)));
  }
  pnl_ifft(temp,resultat);
  pnl_vect_complex_free(&temp);
}
```

```
// "r" takes the product of the toeplitz matrix   N(v,w)
    with holes and a vector x
static void toep (PnlVectComplex *v, PnlVectComplex *w, Pn
    lVectComplex *x, PnlVectComplex *r)
{
  int M, i;
  PnlVectComplex *temp;
  PnlVectComplex *temp2;
  PnlVectComplex *x2;
  M=v->size;
  temp=pnl_vect_complex_create(4*M);
  temp2=pnl_vect_complex_create(4*M);
  x2=pnl_vect_complex_create(4*M);
  pnl_vect_complex_set(temp,0,CRmul(CONE,0.5));
  for (i=1;i<2*M+1;i=i+2)
  {
    pnl_vect_complex_set(temp,i,pnl_vect_complex_get(v,(i-1
    )/2));
    pnl_vect_complex_set(temp,i+1,CZERO);
  }
  for ( i=2*M+1;i<4*M-1;i=i+2)
  {
    pnl_vect_complex_set(temp,i,pnl_vect_complex_get(w,(M-1
    )-((i-1)-2*M)/2));
    pnl_vect_complex_set(temp,i+1,CZERO);
  }
  pnl_vect_complex_set(temp,i,pnl_vect_complex_get(w,(M-1)-
    ((4*M-1-1)-2*M)/2));
  for ( i=0;i<2*M+1;i++)
    pnl_vect_complex_set(x2,i,pnl_vect_complex_get(x,i));
  for ( i=2*M+1;i<4*M;i++)
    pnl_vect_complex_set(x2,i,CZERO);
  circulante(temp2,temp,x2);
  for ( i=0;i<2*M+1;i++)
    pnl_vect_complex_set(r,i,pnl_vect_complex_get(temp2,i))
    ;

  pnl_vect_complex_free(& temp);
  pnl_vect_complex_free(& temp2);
  pnl_vect_complex_free(& x2);
```

```
}

// the characteristic function of the CGMY model
static dcomplex fi_CGMY (double t,double drift, double C,
    double G,double M,double Y,dcomplex w,double signe)
{
  dcomplex temp1,Gc,Mc,u;
  Gc=Complex(G,0);
  Mc=Complex(M,0);
  u=CRmul(w,(double)signe);
  temp1=RCmul(
    C*tgamma(-Y),
    Cadd(
      Csub(Cpow_real(Mc,Y),Cpow_real(Csub(Mc,Cmul(CI,u))
    ,Y)),
            Csub(Cpow_real(Gc,Y),Cpow_real(Cadd(Gc,Cmul(
    CI,u)),Y))
        )
    );
  temp1=Csub(temp1,CRmul(Cmul(CI,u),drift));
  temp1=CRmul(temp1,(-t));    //temp1=temp1*(-t);
  temp1=Cexp(temp1); //temp1=temp1.expon();

  return temp1;
}

// the characteristic function of the CGMY model after the
    Esscher transformation
static dcomplex fi_CGMY_star (double t,double drift,
    double C,double G,double M,double Y, dcomplex u,double x,double
    signe)
{
  dcomplex b;
  dcomplex a;
  dcomplex c;
  a=CRmul(CI,-1);
  c= fi_CGMY ( t,drift, C, G, M, Y, Cadd(u,CRmul(a,x)), si
    gne );
  b= fi_CGMY ( t,drift, C, G, M, Y, CRmul(a,x) , signe);
  a=Cdiv(c,b);
```

```
  return a;
}

//The diagonal matrix which attributes the corresponding
    coeficients to the calculation of F in the CGMY model
static void  F_diag_fi_CGMY (PnlVectComplex *v,double t,
    double drift, double C_MODEL,double G_MODEL,double M_MODEL,
    double Y_MODEL,dcomplex h,double x,int M,double signe)
{
  int com;

  for (com=0;com<2*M+1;com++)
    pnl_vect_complex_set(v,com,Cmul(pnl_vect_complex_get(v,
    com),fi_CGMY_star(t, drift,  C_MODEL, G_MODEL, M_MODEL, Y_
    MODEL,CRmul(h,(double)(M-com)),x,signe)));
}

//The diagonal matrix which attributes the corresponding
    coeficients to the calculation of G in the CGMY model
static void G_diag_fi_CGMY (PnlVectComplex *v,double t,
    double drift, double C_MODEL,double G_MODEL,double M_MODEL,
    double Y_MODEL,dcomplex h, double x,int M,double signe)
{
  int i;

  for (i=0;i<2*M+1;i++)
    pnl_vect_complex_set(v,i,Cmul(pnl_vect_complex_get(v,i)
    ,fi_CGMY_star(t, drift,  C_MODEL, G_MODEL, M_MODEL, Y_
    MODEL,Cadd(CRmul(h,(double)(M-i)),CRmul(CI,x)),x, signe)));
}
// the estimation's algorithm  in the  CGMY model with "n_
    point" maximum observations, and  M point of the Hilbert's
    estimation
static dcomplex estiation_CGMY (double Xmaxmin, PnlVectCom
    plex *G,double t,double drift, double C_MODEL,double G_
    MODEL,double M_MODEL,double Y_MODEL,dcomplex h,double x,int M,
    int n_points,double signe)
{
  int i,j;
 double Xmax;
  dcomplex C,cte,a,k;
```

```
PnlVectComplex *v,*w,*F,*tempg,*tempf;
F=pnl_vect_complex_create (2*M+1);
v=pnl_vect_complex_create (M);    //the first colomn vec
  tor of  the Hilbert's matrix
w=pnl_vect_complex_create (M);    //the first line vector
   of  the Hilbert's matrix
tempg=pnl_vect_complex_create (2*M+1);
tempf=pnl_vect_complex_create (2*M+1);
//initialisation of v and w
cte=CONE;
C=CRmul(CI,M_1_PI); // 1/pi
for (i=0;i<M;i=i+1)
  pnl_vect_complex_set (v,i,CRdiv(C,(double)(2*i+1)));
for (i=0;i<M;i=i+1)
  pnl_vect_complex_set (w,i,CRdiv(C,(double)(-(2*i+1))));
//intialisation  of G_1, F_1 and Cte_1
     Xmax=MAX(Xmaxmin,0);
 for(i=0;i<2*M+1;i++)
   {
     pnl_vect_complex_set (F,i,Cexp( Cmul(     CRmul( CR
  mul(h,(double)(i-M)) , Xmax ),CI)   ));
     pnl_vect_complex_set (G,i,Cexp(CRmul(CRadd(Cmul(CRmu
  l(h,(double)(i-M)),CI),x),Xmax)));
   }
     F_diag_fi_CGMY (F, t, drift,  C_MODEL, G_MODEL, M_
  MODEL, Y_MODEL, h,x, M, signe);//G=( fi_CGMY(1)F(1),..,fi_CGMY(
  i)F(i),..,fi_CGMY(2M+1)F(2M+1) )
     toep(v,w,F,tempf);
     pnl_vect_complex_clone(F,tempf);

     cte=CRmul(CRsub(pnl_vect_complex_get(F,M),1.0),-1);/
  /1-f(0)


     G_diag_fi_CGMY (G,t, drift,  C_MODEL, G_MODEL, M_
  MODEL, Y_MODEL, h,x, M, signe);//G=( fi_CGMY(1)G(1),..,fi_CGMY(
  i)G(i),..,fi_CGMY(2M+1)G(2M+1) )
     toep(v,w,G,tempg);//temp=C(H)*G
     pnl_vect_complex_clone (G,tempg);
```

```
     for (j=1;j<n_points;j++)
     {
       for(i=0;i<2*M+1;i++)
       {
          pnl_vect_complex_set (F,i,Cadd(cte,pnl_vect_
    complex_get(F,i)));//F=F+cte
          pnl_vect_complex_set (G,i,Cadd(cte,pnl_vect_
    complex_get(G,i)));//G=G+cte
       }
       F_diag_fi_CGMY (F, t, drift,  C_MODEL, G_MODEL, M_
    MODEL, Y_MODEL, h,x, M, signe);//G=( fi_CGMY(1)F(1),..,fi_CGMY(
    i)F(i),..,fi_CGMY(2M+1)F(2M+1) )
       toep(v,w,F,tempf);//
       pnl_vect_complex_clone(F,tempf);//F=temp=H.Df*F

       G_diag_fi_CGMY (G,t, drift,  C_MODEL, G_MODEL, M_
    MODEL, Y_MODEL, h,x, M, signe);//G=( fi_CGMY(1)G(1),..,fi_CGMY(
    i)G(i),..,fi_CGMY(2M+1)G(2M+1) )
       toep(v,w,G,tempg);//temp=C(H)*G
       pnl_vect_complex_clone (G,tempg);//G=temp=H.Dg*G

       cte=CRmul(CRadd(pnl_vect_complex_get(F,M),-1),-1);

     }

     k=RCmul(-1,CI);
     a=fi_CGMY ( n_points*t, drift,  C_MODEL, G_MODEL, M_
    MODEL, Y_MODEL,CRmul(k,x), signe);

     pnl_vect_complex_set (G,M,Cmul(Cadd(pnl_vect_
    complex_get (G,M),cte),a));

     pnl_vect_complex_free(&v);
     pnl_vect_complex_free(&w);
     pnl_vect_complex_free(&tempf);
     pnl_vect_complex_free(&tempg);
     pnl_vect_complex_free(&F);

     return pnl_vect_complex_get(G,M);
}
int fft_cgmy_lookbackfloating(double s_maxmin,NumFunc_2*P,
```

```
    double S0,double T,double r,double divid, double C_MODEL,double
    G_MODEL,double M_MODEL,double Y_MODEL,int n_points,long M,
    double *ptprice)
{
  double pas,d,c,drift,nu,h_temp,x_maxmin,signe;
  dcomplex h,res;
  PnlVectComplex *G;
  pas=T/n_points;
  //la largeur du  la bande d'estimation//
  d=MIN(M_MODEL,G_MODEL);
  // le h ideal en fonction de M : h(M)//
  nu=Y_MODEL;
  c=2*C_MODEL*fabs(tgamma(-Y_MODEL)*cos(M_PI*Y_MODEL/2));
  h_temp=pow(((M_PI*d)/(pas*c)),1.0/(1+nu))*pow((double)M,-
    nu/(1+nu));
  h=CRmul(CONE,h_temp);
  //condition martingale//
  drift=r-divid+(
    C_MODEL*tgamma(-Y_MODEL)*
       (
      pow(M_MODEL,Y_MODEL)-pow((M_MODEL-1),Y_MODEL)+pow(
    G_MODEL,Y_MODEL)-pow((G_MODEL+1),Y_MODEL)
        )
    );
  G=pnl_vect_complex_create (2*M+1);
//CALL
  if ((P->Compute)==&Call_StrikeSpot2)
  {
    signe=-1;
    x_maxmin=signe*log ((s_maxmin/S0));
    res=estiation_CGMY(x_maxmin,G,pas,drift, C_MODEL, G_
    MODEL, M_MODEL, Y_MODEL,h,signe*1.0,M,n_points, signe);
    res= CRmul(CRsub(CRmul(res,exp(-r*T)),exp(-divid*T)),-
    S0);
  }
//PUT
  if ((P->Compute)==&Put_StrikeSpot2)
  {
    signe=1;
    x_maxmin=log ((s_maxmin/S0));
    res=estiation_CGMY(x_maxmin,G,pas,drift, C_MODEL, G_
```

```
    MODEL, M_MODEL, Y_MODEL,h,signe*1.0,M,n_points, signe);
    res= CRmul(CRsub(CRmul(res,exp(-r*T)),exp(-divid*T)),S0
    );
  }
  *ptprice=Creal(res);

  pnl_vect_complex_free(&G);

  return OK;
}
int CALC(FFT_CGMY_FloatingLookback)(void*Opt,void *Mod,
    PricingMethod *Met)
{
  TYPEOPT* ptOpt=(TYPEOPT*)Opt;
  TYPEMOD* ptMod=(TYPEMOD*)Mod;
  double r,divid;

  r=log(1.+ptMod->R.Val.V_DOUBLE/100.);
  divid=log(1.+ptMod->Divid.Val.V_DOUBLE/100.);

  return  fft_cgmy_lookbackfloating((ptOpt->PathDep.Val.V_
    NUMFUNC_2)->Par[4].Val.V_PDOUBLE,ptOpt->PayOff.Val.V_NUMFUNC_2,pt
    Mod->S0.Val.V_PDOUBLE,ptOpt->Maturity.Val.V_DATE-ptMod->T.Val
    .V_DATE,r,divid,ptMod->C.Val.V_PDOUBLE,ptMod->G.Val.V_
    DOUBLE,ptMod->M.Val.V_SPDOUBLE,ptMod->Y.Val.V_PDOUBLE,Met->Par[0
    ].Val.V_PINT,Met->Par[1].Val.V_LONG,&(Met->Res[0].Val.V_
    DOUBLE));
}

static int CHK_OPT(FFT_CGMY_FloatingLookback)(void *Opt,
    void *Mod)
{
  if ((strcmp(((Option*)Opt)->Name,"    LookBackCallFloatingEuro")==0) || (strcm
    return OK;
  return WRONG;
}

#endif //PremiaCurrentVersion
static int MET(Init)(PricingMethod *Met,Option *Mod)
{
  if ( Met->init == 0)
```

```
    {
      Met->init=1;
       Met->HelpFilenameHint = "    ap_cgmy_lookbackfloating_fft";
      Met->Par[0].Val.V_PINT=252;
      Met->Par[1].Val.V_LONG=4096;
    }
  return OK;
}

PricingMethod MET(FFT_CGMY_FloatingLookback)=
{
  "FFT_CGMY_LookbackFloating",
  {{"Number of discretization steps",LONG,{100},ALLOW},{"N
    Truncation level (a power of 2)",LONG,{100},ALLOW},{" ",PREM
    IA_NULLTYPE,{0},FORBID}},
  CALC(FFT_CGMY_FloatingLookback),
  {{"Price",DOUBLE,{100},FORBID},{" ",PREMIA_NULLTYPE,{0},
    FORBID}},
  CHK_OPT(FFT_CGMY_FloatingLookback),
  CHK_ok,
  MET(Init)
} ;
```

References