```
   Help
#include "timehes1d_std.h"
#include "pnl/pnl_cdf.h"
#include "pnl/pnl_finance.h"
#include "pnl/pnl_root.h"

#if defined(PremiaCurrentVersion) && PremiaCurrentVersion <
     (2010+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(AP_BGM_TimeHeston)(void *Opt, void *Mod)
{
  return NONACTIVE;
}
int CALC(AP_BGM_TimeHeston)(void*Opt,void *Mod,Pricing
    Method *Met)
{
  return AVAILABLE_IN_FULL_PREMIA;
}
#else
///////////////////////////////////////////////////////////
    //////

int  expansion_terms(double kappa, double v0, double theta,
     double t,
                     double T, double u, double *f0,
    double *f1,
                     double *f2, double *g1, double *g2,
    double *h1,
                     double *h2, double *wu, double *w0u,
    double *wuu)
{
  double kappaT, kappat;

  kappaT = kappa * T;
  kappat = kappa * t;


  *f0=exp(-2*kappaT)*(exp(2*kappat)*(theta-2*v0)+exp(2*kapp
    aT)*((-2*kappat+2*kappaT-5)*theta+2*v0)+4*exp(kappat+kappaT)
    *((-kappat+kappaT+1)*theta+(kappat-kappaT)*v0))/(4.*CUB(ka
    ppa));
  *f1=exp(-kappaT)*(exp(kappaT)*((-kappat+kappaT-2)*theta+
    v0)-exp(kappat)*((kappat-kappaT-2)*theta-kappat*v0+kappaT*
    v0+v0))/(SQR(kappa));

  *f2=exp(-kappa*(t+3*T))*(2*exp(kappa*(t+3*T))*((kappa*(T-
    t)-3)*theta+v0)+exp(2*kappa*(t+T))*((kappa*(kappa*(t-T)-4)*
    (t-T)+6)*theta-(kappa*(kappa*(t-T)-2)*(t-T)+2)*v0))/(2.*
    CUB(kappa));

  *g1=(2*exp(kappaT)*theta+exp(kappat)*(SQR(kappa)*SQR(t-T)
    *v0-(kappa*(kappa*(t-T)-2)*(t-T)+2)*theta))/(2*SQR(kappa))
    ;


  *g2=exp(-kappaT)*(exp(2*kappaT)*theta-exp(2*kappat)*(thet
    a-2*v0)+2*exp(kappa*(t+T))*(kappa*(t-T)*(theta-v0)-v0))/(2*
    SQR(kappa));

  *h1=(exp(kappaT)*theta+exp(kappat)*((kappat-kappaT-1)*th
    eta+kappa*(T-t)*v0))/kappa;

  *h2=(exp(kappat)-exp(kappaT))*(exp(kappat)*(theta-2*v0)-
    exp(kappaT)*theta)/(2*kappa);

  *wu=(-exp(u*t)+exp(u*T))/u;

  *w0u=(exp(T*u)*(-t*u+T*u-1)+exp(t*u))/SQR(u);

  *wuu=SQR(exp(t*u)-exp(T*u))/(2*SQR(u));


  return  0;

}

/*******************************************************
    *************
  Computation of the partial derivatives  given by formula
    (2.13) page 7
 *******************************************************
```

```
    **************/

int greeksBS(double x, double y, double K, double T,
    double r, double divid,
             double *Pxy, double *Pyy, double *Pxxy,
    double *Pxxyy )
{

  double f,g,fg;


  f= (log(K)-x-r*T+divid*T)/sqrt(y) + 0.5*sqrt(y);
  g=f-sqrt(y);
  fg=f*g;

  *Pxy=(0.5/(sqrt(2*M_PI)*y*sqrt(y)))*( exp(x)*exp(-divid*
    T)*( sqrt(y)*f+1-fg )*exp(-0.5*SQR(g))-K*exp(-r*T)*(1-fg)*
    exp(-0.5*SQR(f)) );

  *Pyy=(0.25/(sqrt(2*M_PI)*SQR(y)))*( exp(x)*exp(-divid*T)*
    (-2*f-g+SQR(f)*g)*exp(-0.5*SQR(g))-K*exp(-r*T)*(-2*g-f+SQ
    R(g)*f)*exp(-0.5*SQR(f)) );

  *Pxxy=(0.5/ (sqrt(2*M_PI)*y*sqrt(y)) )*( exp(x)*exp(-div
    id*T)*( ( sqrt(y)*f+1-fg )*(1-g/sqrt(y)) +1-(g+f)/sqrt(y) )
    *exp(-0.5*SQR(g)) -K*exp(-r*T)*(-(f+g)/sqrt(y)-(1-fg)*f/sq
    rt(y))*exp(-0.5*SQR(f)) );

  *Pxxyy=(0.25/(sqrt(2*M_PI)*CUB(y)))*
    ( exp(x)*exp(-divid*T)* ((sqrt(y)-g)* ((-2*f-g+SQR(f)*
    g)*(sqrt(y)-g)-6+4*fg+2*SQR(f))+6*f+3*g-SQR(f)*g)
      *exp(-0.5*SQR(g))-K*exp(-r*T)*(9*f+6*g-3*f*SQR(g)-6*
    SQR(f)*g-CUB(f)+CUB(f)*SQR(g))*exp(-0.5*SQR(f)));


  return  0;


}

/***********************************************************
```

```
    *****
  Pricing formula (2.13) page 7
 **********************************************************
    ******/

int ApBGMHeston(double S, double K, double T, double r,
    double divid,
                double v0, double kappa, double timestep,
    PnlVect *theta, PnlVect *vovol,
                PnlVect *rho, double *ptprice, double *ptde
    lta)
{
  int    i;
  double var,a1,a2,b0,b2,w1,w2,alpha,beta,v0t;
  double f0,f1,f2,g1,g2,h1,h2,wu,w0u,wuu;
  double Pxy, Pyy,  Pxxy, Pxxyy;
  double Pxyhu, Pyyhu, Pxxyhu, Pxxyyhu, Pxyhd, Pyyhd, Pxxyh
    d, Pxxyyhd;
  double BS_put_price,BS_put_delta;
  double h;

  a1    = 0.;
  a2    = 0.;
  b0    = 0.;
  w1    = 0.;
  w2    = 0.;
  alpha = 0.;
  beta  = 0.;
  v0t   = v0;
  var   = 0.;
  h     = 0.01;
  /********************************************************
    *************
    Explicit computations for Picewise  constant parameter
    case see page 7
   ********************************************************
    *************/

  for(i=0;i<(int)(T/timestep);i++)
    {
      expansion_terms(kappa, v0, GET(theta,i),((double)i*
    timestep),((double) (i+1)*timestep),
                      -kappa,&f0,&f1,&f2,&g1,&g2,&h1,&h2,&
    wu,&w0u,&wuu);

      a1+=wu*w1+GET(rho, i)*GET(vovol, i)*f1;
      a2+=wu*alpha+GET(rho, i)*GET(vovol, i)*w0u*w1+SQR(GET
    (rho, i)*GET(vovol, i))*f2;
      b0+=wu*beta+wuu*w2+SQR(GET(vovol, i))*f0;
      alpha+=GET(rho, i)*GET(vovol, i)*w1/4+SQR(GET(rho, i)
    *GET(vovol, i))*g1;
      beta+=wu*w2+SQR(GET(vovol, i))*g2;
      w1+=GET(rho, i)*GET(vovol, i)*h1;
      w2+=SQR(GET(vovol, i))*h2;
      v0t=exp(-kappa/4)*(v0t-GET(theta, i))+GET(theta, i);

      var+=v0t/4;
    }

  b2=0.5*SQR(a1);

  greeksBS(log(S), var, K, T, r,divid, &Pxy, &Pyy, &Pxxy, &
    Pxxyy);
  greeksBS(log(S*(1.+h)), var, K, T, r,divid, &Pxyhu, &Pyyh
    u, &Pxxyhu, &Pxxyyhu);
  greeksBS(log(S*(1.-h)), var, K, T, r,divid, &Pxyhd, &Pyyh
    d, &Pxxyhd, &Pxxyyhd);


  /*BS put price*/
  pnl_cf_put_bs(S,K,T,r,divid,sqrt(var/T),&BS_put_price,&
    BS_put_delta);

  /* Put Price given by formula (2.13) page 7*/
  *ptprice=BS_put_price+a1*Pxy+a2*Pxxy+b0*Pyy+b2*Pxxyy;
  /* Put Delta */
  *ptdelta=BS_put_delta+0.5*( a1*(Pxyhu-Pxyhd)+a2*(Pxxyhu-
    Pxxyhd)+b0*(Pyyhu-Pyyhd)+b2*(Pxxyyhu-Pxxyyhd) )/(S*h);

  return OK;
}
```

```
static int ApBGMTimeHeston(double S,NumFunc_1  *p, double
    T, double r, double divid, double v0,
                           double kappa, double timestep,
    PnlVect *theta, PnlVect *vovol,
                           PnlVect *rho, double *ptprice,
    double *ptdelta)
{
  double K;

  K=p->Par[0].Val.V_PDOUBLE;

  ApBGMHeston(S,K,T,r,divid,v0,kappa,timestep,theta,vovol,
    rho,ptprice,ptdelta);
  if ((p->Compute)==&Call)
    {
      *ptdelta =  1 + *ptdelta;
      *ptprice = (S - K * exp (-r * T)) + *ptprice;
    }
  return OK;
}

int CALC(AP_BGM_TimeHeston)(void *Opt, void *Mod, Pricing
    Method *Met)
{
  int status;
  double r,divid;
  PnlVect *theta, *vovol, *rho;
  PnlMat *all_params;
  TYPEOPT* ptOpt=(TYPEOPT*)Opt;
  TYPEMOD* ptMod=(TYPEMOD*)Mod;

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

  all_params = pnl_mat_create_from_file (ptMod->TimeDepPara
    meters.Val.V_FILENAME);
  theta = pnl_vect_create (0);
  vovol = pnl_vect_create (0);
  rho = pnl_vect_create (0);
  pnl_mat_get_col (theta, all_params, 0);
  pnl_mat_get_col (vovol, all_params, 1);
  pnl_mat_get_col (rho, all_params, 2);

  status = ApBGMTimeHeston(ptMod->S0.Val.V_PDOUBLE,
                           ptOpt->PayOff.Val.V_NUMFUNC_1,
                           ptOpt->Maturity.Val.V_DATE-pt
    Mod->T.Val.V_DATE,
                           r, divid, ptMod->Sigma0.Val.V_
    PDOUBLE,
                           ptMod->MeanReversion.hal.V_PDOUB
    LE,
                           ptMod->TimeStep.Val.V_PDOUBLE,
                           theta, vovol,  rho,
                           &(Met->Res[0].Val.V_DOUBLE),
                           &(Met->Res[1].Val.V_DOUBLE)
                          );

  /* Do not free all_params until you don't use theta, vov
    ol and rho */
  pnl_mat_free (&all_params);
  pnl_vect_free (&theta);
  pnl_vect_free (&vovol);
  pnl_vect_free (&rho);

  return status;
}

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

#endif //PremiaCurrentVersion
static int MET(Init)(PricingMethod *Met,Option *Opt)
{
  if ( Met->init == 0) Met->init=1;
  return OK;
}
```

```
PricingMethod MET(AP_BGM_TimeHeston)=
{
  "AP_BGM_TimeHeston",
  {{" ",PREMIA_NULLTYPE,{0},FORBID}},
  CALC(AP_BGM_TimeHeston),
  {{"Price",DOUBLE,{100},FORBID},
   {"Delta",DOUBLE,{100},FORBID} ,
   {" ",PREMIA_NULLTYPE,{0},FORBID}},
  CHK_OPT(AP_BGM_TimeHeston),
  CHK_ok,
  MET(Init)
};
```

References