```
   Help
extern "C"{
#include "kou1d_pad.h"
}
#include"math/ap_kou_model/functions.h"

extern "C"{
#if defined(PremiaCurrentVersion) && PremiaCurrentVersion <
     (2008+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_Kou_Floating)(void *Opt, void *Mod)
{
  return NONACTIVE;
}
int CALC(AP_Kou_Floating)(void*Opt,void *Mod,PricingMethod
    *Met)
{
return AVAILABLE_IN_FULL_PREMIA;
}
#else
  static int Kou_Ap_Floating(double s_maxmin,NumFunc_2*P,
    double S0,double T,double r,double divid,double sigma,double lam
    bda,double lambdap,double lambdam,double p,double *ptprice,
    double *ptdelta)
  {
    long double  x[11];
    double ksi=p*lambdap/(lambdap-1)+(1-p)*lambdam/(lambdam
    +1)-1;

    /*Call Case*/
    if ((P->Compute)==&Call_StrikeSpot2)
      {
        x[0]=-((r-divid)-sigma*sigma/2-lambda*ksi);
        x[1]=sigma;
        x[2]=lambda;
        x[3]=1-p;
        x[4]=lambdam;
        x[5]=lambdap;
        x[6]=S0;
        x[7]=r;
        x[8]=T;
      x[9]=s_maxmin;
      x[10]=divid;

      *ptprice=CLB(x,T);
      *ptdelta=dCLB(x,T);

    }
  else
    if ((P->Compute)==&Put_StrikeSpot2)
      {
        x[0]=(r-divid)-sigma*sigma/2-lambda*ksi;
        x[1]=sigma;
        x[2]=lambda;
        x[3]=p;
        x[4]=lambdap;
        x[5]=lambdam;
        x[6]=S0;
        x[7]=r;
        x[8]=T;
        x[9]=s_maxmin;
        x[10]=divid;

        *ptprice=PLB(x,T);
        *ptdelta=dPLB(x,T);
      }
 return OK;
}
int CALC(AP_Kou_Floating)(void*Opt,void *Mod,Pricing
  Method *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  Kou_Ap_Floating((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->Sigma.Val.V_PDOUBLE,ptMod->Lambda.
```

```
    Val.V_PDOUBLE,ptMod->LambdaPlus.Val.V_PDOUBLE,ptMod->LambdaM
    inus.Val.V_PDOUBLE,ptMod->P.Val.V_PDOUBLE,&(Met->Res[0].Val.
    V_DOUBLE),&(Met->Res[1].Val.V_DOUBLE));
  }

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

    if ((strcmp(((Option*)Opt)->Name,"    LookBackCallFloatingEuro")==0) || (str
      return OK;
    return WRONG;
  }


#endif //PremiaCurrentVersion
  static int MET(Init)(PricingMethod *Met,Option *Mod)
  {
    return OK;
  }

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

References