```
   Help
extern "C"{
#include "kou1d_std.h"
}
#include<iostream>
#include<cmath>
#include"math/ap_kou_model/functions.h"

#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)
#else
  static int Kou_EuAp(double S0,NumFunc_1  *P,double T,
    double r,double divid,double sigma,double lambda,double lambdap,
    double lambdam,double p,double *ptprice,double *ptdelta)
  {
    double K,rebate;
    long double ksi, cst1, cst2, dcst1, dcst2;
    long double* x=new long double[8];

     ksi=p*lambdap/(lambdap-1)+(1-p)*lambdam/(lambdam+1)-1;
    K=P->Par[0].Val.V_DOUBLE;
    rebate=P->Par[1].Val.V_DOUBLE;
    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]=log(K/S0);
    x[7]=T;
    cst1=psiVN(x);
    dcst1=-dpsiVN(x)/S0;
    /*Digit Case*/
    if ((P->Compute)==&Digit)
      {
        *ptprice = cst1*exp(-r*T)*rebate;
        *ptdelta = dcst1*exp(-r*T)*rebate;
        delete [] x;
        return OK;
      }
```

```
    /*Call Case*/
    else
      {
        x[0]=(r-divid)+sigma*sigma/2-lambda*ksi;
        x[2]=lambda*(ksi+1);
        x[3]=p*lambdap/((1+ksi)*(lambdap-1));
        x[4]=lambdap-1;
        x[5]=lambdam+1;
        cst2=psiVN(x);
        dcst2=-dpsiVN(x)/S0;
        *ptprice =S0*exp(-divid*T)*cst2-K*exp(-r*T)*cst1;
        *ptdelta =exp(-divid*T)*cst2-exp(-divid*T)*dcst2+K*
    exp(-r*T)*dcst1/S0;
        if ((P->Compute)==&Call)
          {
             delete [] x;
            return OK;
            }

        else
          if ((P->Compute)==&Put)
            {
              *ptprice+=K*exp(-r*T)-S0*exp(-divid*T);
              *ptdelta-=exp(-divid*T);
           delete [] x;
           return OK;
            }
      }
   return OK;
  }
#endif //PremiaCurrentVersion

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_Eu)(void *Opt, void *Mod)
{
  return NONACTIVE;
}
int CALC(AP_Kou_Eu)(void*Opt,void *Mod,PricingMethod *Met)
```

```
{
return AVAILABLE_IN_FULL_PREMIA;
}
#else
  int CALC(AP_Kou_Eu)(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  Kou_EuAp(ptMod->S0.Val.V_PDOUBLE,ptOpt->PayOff.
    Val.V_NUMFUNC_1,ptOpt->Maturity.Val.V_DATE-ptMod->T.Val.V_DA
    TE,r,divid,ptMod->Sigma.Val.V_PDOUBLE,ptMod->Lambda.Val.V_
    PDOUBLE,ptMod->LambdaPlus.Val.V_PDOUBLE,ptMod->LambdaMinus.
    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_Eu)(void *Opt, void *Mod)
  {
    if ( (strcmp( ((Option*)Opt)->Name,"CallEuro")==0) || (
    strcmp( ((Option*)Opt)->Name,"PutEuro")==0)|| (strcmp( ((
    Option*)Opt)->Name,"DigitEuro")==0) )
      return OK;

    return WRONG;
  }


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

  PricingMethod MET(AP_Kou_Eu)=
  {
```

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

# References