

Help

```

#include <stdlib.h>
#include <math.h>
#include "pnl/pnl_vector.h"
#include "pnl/pnl_fft.h"
#include "math/wienerhopf.h"
#include "kou1d_pad.h"

#include "pnl/pnl_cdf.h"
#include "pnl/pnl_random.h"
#include "pnl/pnl_specfun.h"

#if defined(PremiaCurrentVersion) && PremiaCurrentVersion < (2012+2) //The "#els
static int CHK_OPT(AP_WH_KOU_FixedLookback)(void *Opt, void *Mod)
{
    return NONACTIVE;
}
int CALC(AP_WH_KOU_FixedLookback)(void *Opt, void *Mod, PricingMethod *Met)
{
    return AVAILABLE_IN_FULL_PREMIA;
}
#else
/*////////////////////////////////////*/

//=====

static int ap_wienerhopf_lookbackfixed_kou(double s_maxmin, NumFunc_2 *P, double
    double r, double divid, double sigma, double lambda, double lambdap, double

{
    int ifCall;
    double cp, cm, ptprice1, ptdelta1, mu, qu, omega, sig2, lp, lm;

    lp = lambdam;
    lm = -lambdap;

    /* if(ifCall==1)          //CALL//
        {omega=lm<-2. ? 2. : (-lm+1.)/2.; }
    else                      //PUT//

```

```

    {omega= lp>1. ? -1. : -lp/2.; }*/

    omega = 0;
    cp = (1 - P0) * lambda;
    cm = P0 * lambda;

    sig2 = sigma * sigma;

    mu = r - divid + cp / (lp + 1.0) + cm / (lm + 1.0) - sig2 / 2.0;

    qu = r - mu * omega - sig2 * omega * omega / 2 + cp + cm - cp * lp / (lp + 1.0);

//CALL
    if ((P->Compute) == &Call_OverSpot2)
    {
        ifCall = 1;
    }
//PUT
    if ((P->Compute) == &Put_OverSpot2)
    {
        ifCall = 0;
    }

    lookback_fxs(4, mu, qu, omega, ifCall, Spot, s_maxmin, lm, lp,
                sigma, sigma, cm, cp, r, divid, Strike,
                T, h, er, &ptprice1, &ptdelta1);

//Price
    *ptprice = ptprice1;
//Delta
    *ptdelta = ptdelta1;

    return OK;
}

//=====
int CALC(AP_WH_KOU_FixedLookback)(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 ap_wienerhopf_lookbackfixed_kou((ptOpt->PathDep.Val.V_NUMFUNC_2)->Par[
        ptOpt->PayOff.Val.V_NUMFUNC_2, ptMod->
        r, divid, ptMod->Sigma.Val.V_PDOUBLE,
        ptMod->LambdaMinus.Val.V_PDOUBLE, ptMo
        Met->Par[1].Val.V_SPDOUBLE, Met->Par[0
}

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

#endif //PremiaCurrentVersion
static int MET(Init)(PricingMethod *Met, Option *Opt)
{
    static int first = 1;

    if (first)
    {
        Met->HelpFilenameHint = "AP_FASTWH_KOU_lookbackfixed";
        Met->Par[0].Val.V_PDOUBLE = 2.0;
        Met->Par[1].Val.V_PDOUBLE = 0.001;

        first = 0;
    }
    return OK;
}

PricingMethod MET(AP_WH_KOU_FixedLookback) =
{
    "AP_FastWH",
    { {"Scale of logprice range", DOUBLE, {100}, ALLOW},
      {"Space Discretization Step", DOUBLE, {500}, ALLOW},

```

```
    {" ", PREMIA_NULLTYPE, {0}, FORBID}
},
CALC(AP_WH_KOU_FixedLookback),
{ {"Price", DOUBLE, {100}, FORBID},
  {"Delta", DOUBLE, {100}, FORBID},
  {" ", PREMIA_NULLTYPE, {0}, FORBID}
},
CHK_OPT(AP_WH_KOU_FixedLookback),
CHK_split,
MET(Init)
};
```