```
   Help
#include <stdlib.h>
#include <math.h>
#include  "temperedstable1d_lim.h"
#include "pnl/pnl_vector_double.h"
#include "pnl/pnl_fft.h"
#include "math/wienerhopf.h"

#if defined(PremiaCurrentVersion) && PremiaCurrentVersion <
     (2009+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_fastwhdownout)(void *Opt, void *Mod)
{
  return NONACTIVE;
}
int CALC(AP_fastwhdownout)(void*Opt,void *Mod,Pricing
    Method *Met)
{
return AVAILABLE_IN_FULL_PREMIA;
}
#else
/*////////////////////////////////////////////////////*/

 //=========================================================
    ========================
static int wh_tsl_downout(int am, int upordown, int ifCall,
     double Spot, double lm1, double lp1,
            double num,double nup, double cm,double cp,
            double r, double divid,
            double T, double h, double Strike1,
            double bar,double rebate,
            double er, long int step,
            double *ptprice, double *ptdelta)
{
  double cnup, cnum, lpnu, lmnu, ptprice1, ptdelta1, mu,
    qu, om;

  if(upordown==0)
   {om=lm1<-2. ? 2. : (-lm1+1.)/2.;  }
   else
   {om= lp1>1. ? -1. : -lp1/2.; }
```

```
  cnup=cp*tgamma(-nup);
  cnum=cm*tgamma(-num);

  lpnu=exp(nup*log(lp1));
  lmnu=exp(num*log(-lm1));

   mu= r - divid + cnup*(lpnu-exp(nup*log(lp1+1.0))) + cn
    um*(lmnu-exp(num*log(-lm1-1.0)));


  qu = r + (pow(lp1,nup) - pow(lp1+om,nup))*cnup + (pow(-
    lm1,num)-pow(-lm1-om,num))*cnum;


  fastwienerhopf(1, mu, qu, om, am, upordown, ifCall, Spo
    t, lm1, lp1,
            num, nup, cnum, cnup, r, divid,
            T, h, Strike1, bar, rebate,
            er, step, &ptprice1, &ptdelta1);

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

  return OK;
}

//=========================================================
    ==========================
int CALC(AP_fastwhdownout)(void *Opt,void *Mod,Pricing
    Method *Met)
{
  TYPEOPT* ptOpt=( TYPEOPT*)Opt;
  TYPEMOD* ptMod=( TYPEMOD*)Mod;
  double r,divid,limit, strike, spot,rebate;

  NumFunc_1 *p;
  int res;
  int upordown;
```

```
  int ifCall;

  r=log(1.+ptMod->R.Val.V_DOUBLE/100.);
  divid=log(1.+ptMod->Divid.Val.V_DOUBLE/100.);
  limit=((ptOpt->Limit.Val.V_NUMFUNC_1)->Compute)((ptOpt->    Limit.Val.V_NUMFUN
  p=ptOpt->PayOff.Val.V_NUMFUNC_1;
  strike=p->Par[0].Val.V_DOUBLE;
  spot=ptMod->S0.Val.V_DOUBLE;
  ifCall=((p->Compute)==&Call);

  rebate=((ptOpt->Rebate.Val.V_NUMFUNC_1)->Compute)((ptOpt-
    >Rebate.Val.V_NUMFUNC_1)->Par,ptMod->T.Val.V_DATE);


  if ((ptOpt->DownOrUp).Val.V_BOOL==DOWN)
    upordown=0;
  else upordown=1;

  res = wh_tsl_downout(ptOpt->EuOrAm.Val.V_BOOL,upordown,
    ifCall, spot, -ptMod->LambdaPlus.Val.V_PDOUBLE, ptMod->Lam
    bdaMinus.Val.V_PDOUBLE,
        ptMod->AlphaPlus.Val.V_PDOUBLE, ptMod->AlphaMi
    nus.Val.V_PDOUBLE,
        ptMod->CPlus.Val.V_PDOUBLE, ptMod->CMinus.Val.V_
    PDOUBLE,
        r, divid,
        ptOpt->Maturity.Val.V_DATE-ptMod->T.Val.V_DATE,
    Met->Par[1].Val.V_DOUBLE, strike,
                           limit,rebate,
        Met->Par[0].Val.V_DOUBLE, Met->Par[2].Val.V_INT2
    ,
                           &(Met->Res[0].Val.V_DOUBLE), &(
    Met->Res[1].Val.V_DOUBLE));

 return res;

}
static int CHK_OPT(AP_fastwhdownout)(void *Opt, void *Mod)
{
  Option* ptOpt=(Option*)Opt;
  TYPEOPT* opt=(TYPEOPT*)(ptOpt->TypeOpt);
```

```
  if ((opt->OutOrIn).Val.V_BOOL==OUT)
    if ((opt->Parisian).Val.V_BOOL==WRONG)
  if ((opt->EuOrAm).Val.V_BOOL==EURO)
  return  OK;

  return WRONG;
}


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

  if (first)
    {
      Met->Par[0].Val.V_PDOUBLE=2.0;
      Met->Par[1].Val.V_PDOUBLE=0.001;
      Met->Par[2].Val.V_INT2=100;

      first=0;
    }
  return OK;
}

PricingMethod MET(AP_fastwhdownout)=
{
  "AP_FastWH",
  { {"Scale of logprice range", DOUBLE, {100}, ALLOW},
    {"Space Discretization Step",DOUBLE,{500},ALLOW},
    {"TimeStepNumber",INT2,{100},ALLOW},
   {" ",PREMIA_NULLTYPE,{0},FORBID}},
  CALC(AP_fastwhdownout),
  {{"Price",DOUBLE,{100},FORBID},
   {"Delta",DOUBLE,{100},FORBID},
   {" ",PREMIA_NULLTYPE,{0},FORBID}},
  CHK_OPT(AP_fastwhdownout),
  CHK_split,
  MET(Init)
};
```

References