

## Help

```

#include <stdlib.h>
#include "temperedstable1d_std.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_fastwhamer)(void *Opt, void *Mod)
{
    return NONACTIVE;
}
int CALC(AP_fastwhamer)(void*Opt,void *Mod,PricingMethod *
    Met)
{
    return AVAILABLE_IN_FULL_PREMIA;
}
#else

static int wh_tsl_american(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 er, long int step,
        double *ptprice, double *ptdelta)
{
    double cnup, cnum, lpnu, lmnu, ptprice1, ptdelta1, mu,
        qu, om;

    if(ifCall==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;

    fastwienerhopfamerican(1, mu, qu, om, ifCall, Spot, lm1,
        lp1,
            num, nup, cnum, cnup, r, divid,
            T, h, Strike1,
            er, step, &ptprice1, &ptdelta1);

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

    return OK;
}

//=====
=====
int CALC(AP_fastwhamer)(void *Opt,void *Mod,PricingMethod *
    Met)
{
    TYPEOPT* ptOpt=( TYPEOPT*)Opt;
    TYPEMOD* ptMod=( TYPEMOD*)Mod;
    double r,divid, strike, spot;

    NumFunc_1 *p;
    int res;

    int ifCall;

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

    p=ptOpt->PayOff.Val.V_NUMFUNC_1;
    strike=p->Par[0].Val.V_DOUBLE;
    spot=ptMod->S0.Val.V_DOUBLE;

```

```

    ifCall=((p->Compute)==&Call);

    res = wh_tsl_american(ifCall, spot, -ptMod->LambdaPlus.
        Val.V_PDOUBLE, ptMod->LambdaMinus.Val.V_PDOUBLE,
        ptMod->AlphaPlus.Val.V_PDOUBLE, ptMod->AlphaMinus.Val.
        V_PDOUBLE,
        ptMod->CPlus.Val.V_PDOUBLE, ptMod->CMinus.Val.V_PDOUB
        LE,
        r, divid,
        ptOpt->Maturity.Val.V_DATE-ptMod->T.Val.V_DATE, Met->
        Par[1].Val.V_DOUBLE, strike,
        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_fastwhamer)(void *Opt, void *Mod)
{
    // Option* ptOpt=(Option*)Opt;
    // TYPEOPT* opt=(TYPEOPT*)(ptOpt->TypeOpt);

    if ((strcmp( ((Option*)Opt)->Name,"PutAmer")==0) || (strc
        mp( ((Option*)Opt)->Name, "CallAmer")==0) )
        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_PDDOUBLE=0.01;
    Met->Par[2].Val.V_INT2=600;

    first=0;
}

return OK;
}

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

```

## References