```
Help
#include "dps std.h"
#include "math/equity_pricer/levy_diffusion.h"
#include "math/equity pricer/carr.h"
#if defined(PremiaCurrentVersion) && PremiaCurrentVersion <</pre>
     (2010+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(CF_CarrDPS)(void *Opt, void *Mod)
  return NONACTIVE;
}
int CALC(CF_CarrDPS)(void*Opt,void *Mod,PricingMethod *Met)
return AVAILABLE_IN_FULL_PREMIA;
#else
int CALC(CF_CarrDPS)(void *Opt, void *Mod, PricingMethod *
    Met)
  TYPEOPT* ptOpt=(TYPEOPT*)Opt;
  TYPEMOD* ptMod=(TYPEMOD*)Mod;
  double jump_drift;
  NumFunc_1 *p;
  int option_type;
  int std=1;
  Option Eqd *op;
  DPS_diffusion *Process= DPS_diffusion_create(ptMod->Eta.
    Val.V_PDOUBLE,
                                                  ptMod->Ka
    ppa.Val.V PDOUBLE,
                                                  ptMod->Rh
    o.Val.V_PDOUBLE,
                                                  ptMod->Th
    eta.Val.V_PDOUBLE,
                                                  ptMod->Si
    gma0.Val.V_PDOUBLE,
                                                  ptMod->Mea
    nS.Val.V_PDOUBLE,
                                                  ptMod->Si
```

```
gmaS. Val. V PDOUBLE,
                                               ptMod->Lam
bdaS.Val.V_PDOUBLE,
                                              ptMod->Mea
nV.Val.V PDOUBLE,
                                               ptMod->Lam
bdaV.Val.V_PDOUBLE,
                                              ptMod->Mea
nSV.Val.V PDOUBLE,
                                              ptMod->Si
gmaSV.Val.V_PDOUBLE,
                                              ptMod->Mea
nVS.Val.V PDOUBLE,
                                              ptMod->Lam
bdaSV.Val.V_PDOUBLE,
                                               ptMod->Rh
oSV.Val.V PDOUBLE,
                                               &jump_drif
t);
  Levy diffusion * Levy =Levy diffusion create(Process,
&DPS diffusion characteristic exponent, &DPS diffusion ln
characteristic_function);
  p=ptOpt->PayOff.Val.V_NUMFUNC_1;
  if ((p->Compute) == &Call)
    option_type=1;
  else
    if((p->Compute) ==&Put)
      option_type=2;
    else
      option_type=3;
  op=option eqd create(ptOpt->EuOrAm.Val.V BOOL,option
type, std, ptMod->SO. Val. V PDOUBLE, p->Par[0]. Val. V DOUBLE, pt
Opt->Maturity.Val.V_DATE-ptMod->T.Val.V_DATE,0,0);
  option eqd set rate(op,log(1.+ptMod->R.Val.V DOUBLE/1
00.),log(1.+ptMod->Divid.Val.V DOUBLE/100.));
  CarrMethod Vanilla option LD(op,0.1,Levy);
  (Met->Res[0].Val.V DOUBLE)=op->price;
  (Met->Res[1].Val.V_DOUBLE)=op->delta;
```

```
free(op);
      free(Levy);
      free(Process);
      return OK;
}
static int CHK_OPT(CF_CarrDPS)(void *Opt, void *Mod)
{
  if ((strcmp( ((Option*)Opt)->Name, "CallEuro")==0)||(strc
    mp( ((Option*)Opt)->Name, "PutEuro")==0))
    return OK;
  return WRONG;
}
#endif //PremiaCurrentVersion
static int MET(Init)(PricingMethod *Met,Option *Opt)
{
  if (Met->init == 0)
    {
      Met->init=1;
    }
  return OK;
}
PricingMethod MET(CF_CarrDPS)=
  "CF Carr DPS",
  {{" ",PREMIA NULLTYPE,{0},FORBID}},
  CALC(CF_CarrDPS),
  {{"Price",DOUBLE,{100},FORBID},
   {"Delta",DOUBLE,{100},FORBID} ,
   {" ",PREMIA_NULLTYPE,{0},FORBID}},
  CHK_OPT(CF_CarrDPS),
  CHK_ok,
  MET(Init)
};
```

References