Help

```
#include  "
href../../../mod/bs1d/bs1d_pad/bs1d_pad_h_src.pdfbs1d_pad.h"
#include "pnl/pnl_specfun.h"

#if defined(PremiaCurrentVersion) && PremiaCurrentVersion < (2016+2) //The "#els
static int CHK_OPT(AP_FixedAsian_Stratified_Lognormal)(void *Opt, void *Mod)
{
  return NONACTIVE;
}
int CALC(AP_FixedAsian_Stratified_Lognormal)(void *Opt, void *Mod, PricingMethod
{
  return AVAILABLE_IN_FULL_PREMIA;
}
#else

static int IsFiniteNumber(double x)
{
  return (x <= DBL_MAX && x >= -DBL_MAX);
}

static double phi(double x)
{
  double val;

  val=x/sqrt(2.0);

  return 0.5+0.5*pnl_sf_erf(val);
}

static double p(double sigma,double t, double z)
{
  return exp(-pow(sigma*sigma*t/2+log(z),2)/2/sigma/sigma/t)/sqrt(2*M_PI*sigma*s
}

static double q(double sigma,double t, double z)
{
  return exp(-pow(sigma*sigma*t+log(z),2)/2/sigma/sigma/t)/sqrt(2*M_PI*sigma*sig
}
```

```
static double a(double sigma,double t, double z)
{
      return (phi(log(z)/sigma/sqrt(t)+sigma*sqrt(t)/2.0)-phi(log(z)/sigma/sqrt(

}

static double b(double sigma,double t, double z)
{

    return (phi(log(z)/sigma/sqrt(t)+sigma*sqrt(t))-phi(log(z)/sigma/sqrt(t)-sig

}

static int Stratified_Lognormal_FixedAsian(double pseudo_stock, double pseudo_st
{
  int call_or_put;
  int i;
  double u,du,dz,sig,z,p,mu,d1,d2,mean;
  double CTtK, PTtK, Dlt, Plt;
  int N=500;

  if ((po->Compute) == &Call_OverSpot2)
call_or_put=1;
  else
call_or_put=0;

  p=1-2*(r-divid)/sigma/sigma;

  /*  Call Price */
  CTtK=0;
  du=12*sqrt(t)/N;
  for (i=1;i<N;i++){
    u=du*(i-N/2.0);
    z=pseudo_stock*exp(sigma*u-p*sigma*sigma*t/2);
    mean=pseudo_stock*a(sigma,t,z/pseudo_stock)/t;
    sig = sqrt((1/t)*log(2*(b(sigma,t,z/pseudo_stock)/a(sigma,t,z/pseudo_stock)-
mu=-(1 - 2*log(t*mean)/sig/sig/t)*sig*sig*t/2;
d1=(log(pseudo_stock*a(sigma,t,z/pseudo_stock)/pseudo_strike/t))/(sig*sqrt(t))+s
    d2=d1-sig*sqrt(t);
    CTtK=CTtK+exp(-divid * t)*(12*sqrt(t)/N)*(exp(-(r-divid)*t)*((1/t)*exp(mu+si
  }
```

```
  /*  Delta for call option*/
  Dlt=0.0;
  dz=( exp(r-divid+sigma*sigma/2)+12*sqrt(( exp(sigma*sigma/2) - 1 )*exp(2*( r-d
  for (i=1;i<N;i++){
z=dz*i;
mean=a(sigma,t,z)/t;
sig = sqrt((1/t)*log(2*(b(sigma,t,z)/a(sigma,t,z)-1-z)/(sigma*sigma*a(sigma,t,z)
if (IsFiniteNumber(sig)) {
  mu=-(1 - 2*log(t*mean)/sig/sig/t)*sig*sig*t/2;
    d1=(log(a(sigma,t,z)*pseudo_stock/pseudo_strike/t))/(sig*sqrt(t))+sig*sqrt(t
    d2=d1-sig*sqrt(t);
    Dlt=Dlt+exp(-divid * t)*dz*pseudo_stock*(exp(-(r-divid)*t)*((1/t)*exp(mu+sig
        Dlt=Dlt+exp(-divid * t)*dz*(exp(-(r-divid)*t)*((1/t)*exp(mu+sig*sig*t/2)
   };
  }
  /*  Put Price from Parity*/
  if (r == divid)
    PTtK = CTtK + pseudo_strike * exp(-r * t) - pseudo_stock * exp(-r * t);
  else
    PTtK = CTtK + pseudo_strike * exp(-r * t) - pseudo_stock * exp(-r * t) * (ex

  /*Delta for put option*/
  if (r == divid)
    Plt = Dlt - exp(-r * t);
  else
    Plt = Dlt - exp(-r * t) * (exp((r - divid) * t) - 1.0) / (t * (r - divid));

  /*Price*/
  if(call_or_put)
    *ptprice = CTtK;
  else
    *ptprice = PTtK;

  /*Delta */
  if(call_or_put)
    *ptdelta = Dlt;
  else
    *ptdelta = Plt;

  return OK;
```

```
}

int CALC(AP_FixedAsian_Stratified_Lognormal)(void *Opt, void *Mod, PricingMethod
{
  TYPEOPT *ptOpt = (TYPEOPT *)Opt;
  TYPEMOD *ptMod = (TYPEMOD *)Mod;

  int return_value;
  double r, divid, time_spent, pseudo_spot, pseudo_strike;
  double t_0, T_0;

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

  T_0 = ptMod->T.Val.V_DATE;
  t_0 = (ptOpt->PathDep.Val.V_NUMFUNC_2)->Par[0].Val.V_PDOUBLE;


  if (T_0 < t_0)
    {
      Fprintf(TOSCREEN, "T_0 < t_0, untreated case\ n\ n\ n");
      return_value = WRONG;
    }
  /* Case t_0 <= T_0 */
  else
    {
      time_spent = (ptMod->T.Val.V_DATE - (ptOpt->PathDep.Val.V_NUMFUNC_2)->Par[
      pseudo_spot = (1. - time_spent) * ptMod->S0.Val.V_PDOUBLE;
      pseudo_strike = (ptOpt->PayOff.Val.V_NUMFUNC_2)->Par[0].Val.V_PDOUBLE - ti

      if (pseudo_strike <= 0.)
        {
          Fprintf(TOSCREEN, "ANALYTIC FORMULA\ n\ n\ n");
          return_value = Analytic_KemnaVorst(pseudo_spot, pseudo_strike, time_sp
        }
      else
        return_value = Stratified_Lognormal_FixedAsian(pseudo_spot, pseudo_strik
    }
  return return_value;

}
```

```
static int CHK_OPT(AP_FixedAsian_Stratified_Lognormal)(void *Opt, void *Mod)
{
  if ((strcmp(((Option *)Opt)->Name, "AsianCallFixedEuro") == 0) || (strcmp(((Op
    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(AP_FixedAsian_Stratified_Lognormal) =
{
  "AP_FixedAsian_Stratified_Lognormal",
  {{" ", PREMIA_NULLTYPE, {0}, FORBID}},
  CALC(AP_FixedAsian_Stratified_Lognormal),
  {{"Price", DOUBLE, {100}, FORBID}, {"Delta", DOUBLE, {100}, FORBID} , {" ", PR
  CHK_OPT(AP_FixedAsian_Stratified_Lognormal),
  CHK_ok,
  MET(Init)
};
```