Help

```
#include "bs1d_pad.h"

int MOD_OPT(ChkMix)(Option *Opt, Model *Mod)
{
  TYPEOPT *ptOpt = (TYPEOPT *)(Opt->TypeOpt);
  TYPEMOD *ptMod = (TYPEMOD *)(Mod->TypeModel);
  int status = OK;

  if (ptOpt->Maturity.Val.V_DATE <= ptMod->T.Val.V_DATE)
    {
      Fprintf(TOSCREENANDFILE, "Current date greater than maturity!\ n");
      status += 1;
    };
  if ((ptOpt->MinOrElse).Val.V_BOOL == MINIMUM)
    {
      if ((ptOpt->PathDep.Val.V_NUMFUNC_2)->Par[4].Val.V_PDOUBLE > ptMod->S0.Val
        {
          Fprintf(TOSCREENANDFILE, "Minimum greater than spot!\ n");
          status += 1;
        };
    }
  if ((ptOpt->MinOrElse).Val.V_BOOL == MAXIMUM)
    {
      if ((ptOpt->PathDep.Val.V_NUMFUNC_2)->Par[4].Val.V_PDOUBLE < ptMod->S0.Val
        {
          Fprintf(TOSCREENANDFILE, "Maximum lower than spot!\ n");
          status += 1;
        };
    }
  return status;
}

extern PricingMethod MET(MC_FloatingAsian_Standard);
extern PricingMethod MET(CF_Fixed_CallLookBack);
extern PricingMethod MET(CF_Fixed_PutLookBack);
extern PricingMethod MET(CF_Floating_CallLookBack);
extern PricingMethod MET(CF_Floating_PutLookBack);
extern PricingMethod MET(AP_FixedAsian_Laplace);
extern PricingMethod MET(AP_FixedAsian_LaplaceFourier);
```

```
extern PricingMethod MET(AP_FixedAsian_Levy);
extern PricingMethod MET(AP_FixedAsian_TurnbullWakeman);
extern PricingMethod MET(AP_FixedAsian_Stratified_Lognormal);
extern PricingMethod MET(AP_FixedAsian_MilevskyPosner);
extern PricingMethod MET(AP_FixedAsian_ThompsonLow);
extern PricingMethod MET(AP_FixedAsian_ThompsonUp);
extern PricingMethod MET(AP_FixedAsian_LordLow);
extern PricingMethod MET(AP_FixedAsian_LordUp);
extern PricingMethod MET(AP_FixedAsian_FusaiTagliani);
extern PricingMethod MET(AP_FixedAsian_FusaiMeucci);
extern PricingMethod MET(AP_FixedAsian_Zhang);
extern PricingMethod MET(AP_FloatingAsian_Zhang);
extern PricingMethod MET(AP_FixedAsian_CarmonaDurlemann);
extern PricingMethod MET(FD_FixedAsian_RodgerShi);
extern PricingMethod MET(FD_FixedAsian_RodgerShi2);
extern PricingMethod MET(FD_ExplicitLookback);
extern PricingMethod MET(MC_FixedAsian_KemnaVorst);
extern PricingMethod MET(MC_FixedAsian_Glassermann);
extern PricingMethod MET(MC_FixedAsian_Stratification);
extern PricingMethod MET(MC_FixedAsian_StratificationAdaptive);
extern PricingMethod MET(MC_FixedAsian_RobbinsMonro);
extern PricingMethod MET(MC_FixedAsian_Exact);
extern PricingMethod MET(MC_LookBackMax_Andersen);
extern PricingMethod MET(MC_LookBackMin_Andersen);
extern PricingMethod MET(TR_Babbs_Call);
extern PricingMethod MET(TR_Babbs_Put);
extern PricingMethod MET(TR_Asian_FSG);
extern PricingMethod MET(TR_Asian_SingularPointsSup);
extern PricingMethod MET(MC_MovingAverage_BernhartTankovWarin);
/*extern PricingMethod MET(FD_FixedAsian_BenHameurBretonLecuyer);*/
PricingMethod *MOD_OPT(methods)[] =
{
  &MET(MC_FloatingAsian_Standard),
  &MET(CF_Fixed_CallLookBack),
  &MET(CF_Fixed_PutLookBack),
  &MET(CF_Floating_CallLookBack),
  &MET(CF_Floating_PutLookBack),
  &MET(AP_FixedAsian_Laplace),
  &MET(AP_FixedAsian_LaplaceFourier),
  &MET(AP_FixedAsian_Levy),
  &MET(AP_FixedAsian_TurnbullWakeman),
```

```
  &MET(AP_FixedAsian_Stratified_Lognormal),
  &MET(AP_FixedAsian_MilevskyPosner),
  &MET(AP_FixedAsian_ThompsonLow),
  &MET(AP_FixedAsian_ThompsonUp),
  &MET(AP_FixedAsian_LordLow),
  &MET(AP_FixedAsian_LordUp),
  &MET(AP_FixedAsian_FusaiTagliani),
  &MET(AP_FixedAsian_FusaiMeucci),
  &MET(AP_FixedAsian_Zhang),
  &MET(AP_FloatingAsian_Zhang),
  &MET(AP_FixedAsian_CarmonaDurlemann),
  &MET(FD_FixedAsian_RodgerShi),
  &MET(FD_FixedAsian_RodgerShi2),
  &MET(FD_ExplicitLookback),
  &MET(MC_FixedAsian_KemnaVorst),
  &MET(MC_FixedAsian_Glassermann),
  &MET(MC_FixedAsian_Stratification),
  &MET(MC_FixedAsian_StratificationAdaptive),
  &MET(MC_FixedAsian_RobbinsMonro),
  &MET(MC_FixedAsian_Exact),
  &MET(MC_LookBackMax_Andersen),
  &MET(MC_LookBackMin_Andersen),
  &MET(TR_Babbs_Call),
  &MET(TR_Babbs_Put),
  &MET(TR_Asian_FSG),
  &MET(TR_Asian_SingularPointsSup),
  /*&MET(FD_FixedAsian_BenHameurBretonLecuyer),*/
  &MET(MC_MovingAverage_BernhartTankovWarin),
  NULL
};

extern DynamicTest MOD_OPT(test);
DynamicTest *MOD_OPT(tests)[] =
{
  &MOD_OPT(test),
  NULL
};

Pricing MOD_OPT(pricing) =
{
  ID_MOD_OPT,
```

```
  MOD_OPT(methods),
  MOD_OPT(tests),
  MOD_OPT(ChkMix)
};



/* ------------------------------------------------------------------- */
/* Analytic formula used if K'<0. */
/* ------------------------------------------------------------------- */

int MOD_OPT(Analytic_KemnaVorst_lib)(double pseudo_stock, double pseudo_strike,
                                     double t, double r, double divid, double *p
{
  double b;

  b = r - divid;

  /* Put Case */
  if ((p->Compute) == &Put_OverSpot2)
    {
      *ptprice = 0.;
      *ptdelta = 0.;
    }
  /* Call case */
  else
    {
      /* Case r=d */
      if (b == 0.)
        {
          *ptprice = exp(-r * t) * (pseudo_stock - pseudo_strike);
          *ptdelta = exp(-r * t) * (1. - time_spent);
        }
      /* Case r <> d */
      else
        {
          *ptprice = exp(-r * t) * (pseudo_stock * (1.0 / (b * t)) * (exp(b * t)
          *ptdelta = exp(-r * t) * ((1 - time_spent) / (b * t) * (exp(b * t) - 1
        }
    }
  return OK;
```

5 pages

}