Help

```
extern "C" {
#include "
href../../../mod/bsnd/bsnd_stdnd/bsnd_stdnd_h_src.pdfbsnd_stdnd.h"
#include "
href../../../common/optype_h_src.pdfoptype.h"
#include "
href../../../common/enums_h_src.pdfenums.h"
}

#include <
href../../../common/math/highdim_solver/highdim_vector_h_src.pdfvector>
#include "
href../../../common/math/mcam/src/Model_h_src.pdfmath/mcam/src/Model.hpp"
#include "
href../../../common/math/mcam/src/Option_h_src.pdfmath/mcam/src/Option.hpp"
#include "
href../../../common/math/mcam/src/Martingale_h_src.pdfmath/mcam/src/Martingale.h
#include "
href../../../common/math/mcam/src/workspace_h_src.pdfmath/mcam/src/workspace.hpp
#include "
href../../../common/math/mcam/src/optim_h_src.pdfmath/mcam/src/optim.hpp"
#include "
href../../../common/math/mcam/src/MonteCarlo_h_src.pdfmath/mcam/src/MonteCarlo.h
#include "
href../../../common/math/jlparser/include/jlparser/parser_h_src.pdfjlparser/pars


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

```
int CALC(MC_Lelong_dual)(void *Opt, void *Mod, PricingMethod *Met)
{
  TYPEOPT *ptOpt = (TYPEOPT *)Opt;
  TYPEMOD *ptMod = (TYPEMOD *)Mod;
  PnlVect *spot_, *divid_, *sig_;
  double r;
  int size;
  mcam::Model *mod;
  mcam::Option *opt;
  mcam::Martingale *mart;
  mcam::Optim *optim;

  size = ptMod->Size.Val.V_PINT;
  divid_ = pnl_vect_create(size);
  spot_ = ptMod->S0.Val.V_PNLVECT;
  sig_ = ptMod->Sigma.Val.V_PNLVECT;
  r = log(1. + ptMod->R.Val.V_DOUBLE / 100.);
  for (int i = 0; i < size; i++)
    pnl_vect_set(divid_, i,
                 log(1. + GET(ptMod->Divid.Val.V_PNLVECT, i) / 100.));


  Param P;
  std::vector<double> sig(sig_->array, sig_->array + size);
  std::vector<double> spot(spot_->array, spot_->array + size);
  std::vector<double> dividend_rate(divid_->array, divid_->array + size);

  P.insert("model type", T_STRING, std::string("bs"));
  P.insert("model size", T_INT, size);
  P.insert("spot", T_VECTOR, spot);
  P.insert("maturity", T_DOUBLE, ptOpt->Maturity.Val.V_DATE - ptMod->T.Val.V_DAT
  P.insert("volatility", T_VECTOR, sig);
  P.insert("interest rate", T_DOUBLE, r);
  P.insert("dividend rate", T_VECTOR, dividend_rate);
  P.insert("correlation", T_DOUBLE, ptMod->Rho.Val.V_DOUBLE);
  P.insert("option type", T_STRING, std::string("PremiaOption"));
  P.insert("martingale type", T_STRING, std::string("chaos"));
  P.insert("optim cost", T_STRING, std::string("MaxCost"));
  P.insert("MC iterations", T_INT, Met->Par[0].Val.V_INT);
  P.insert("SAA iterations", T_INT, Met->Par[0].Val.V_INT);
  P.insert("sub ticks", T_INT, 1);
```

```
  P.insert("chaos order", T_INT,  Met->Par[1].Val.V_INT);
  P.insert("dates", T_INT,  Met->Par[2].Val.V_INT);
  P.insert("payoff numfunc", T_PTR,  static_cast<void *>(ptOpt->PayOff.Val.V_NUM

  if ((mod = mcam::instantiate_model(P)) == NULL) abort();
  if ((opt = mcam::instantiate_option(P)) == NULL) abort();
  if ((mart = mcam::instantiate_martingale(mod, P, false)) == NULL) abort();
  if ((optim = mcam::instantiate_optim(mod, opt, mart, P)) == NULL) abort();

  PnlVect *alpha = pnl_vect_create_from_zero(mart->nbFreedom);
  mcam::PnlRng_Workspace rng;
  rng.set_seed(0);
  mcam::StepMethod *stepMethod = new mcam::LineSearchStep();
  double prix, var, grad_cost;
  optim->saa(rng, alpha, prix, var, stepMethod);
  std::cout << "SAA Price:" << prix << "\ n";
  optim->EGradCost(rng, alpha, grad_cost, prix, var, false);
  Met->Res[0].Val.V_DOUBLE = prix;

  delete mod;
  delete opt;
  delete mart;
  delete optim;
  delete stepMethod;
  pnl_vect_free(&alpha);
  pnl_vect_free(&divid_);
  return OK;
}

static int CHK_OPT(MC_Lelong_dual)(void *Opt, void *Mod)
{
  Option *ptOpt = (Option *)Opt;
  TYPEOPT *opt = (TYPEOPT *)(ptOpt->TypeOpt);
  if ((opt->EuOrAm).Val.V_BOOL == AMER)
    return OK;
  else
    return  WRONG;
}

#endif
```

```
static int MET(Init)(PricingMethod *Met, Option *Opt)
{
  if (Met->init == 0)
    {
      Met->init = 1;
      Met->HelpFilenameHint = "mc_lelong_dual";
      Met->Par[0].Val.V_INT = 5000;
      Met->Par[1].Val.V_INT = 3;
      Met->Par[2].Val.V_INT = 5;
    }
  return OK;
}

PricingMethod MET(MC_Lelong_dual) =
{
  "MC_Lelong_dual",
  { {"N iterations", INT, {100}, ALLOW, SETABLE, NULL},
    {"Chaos order", INT, {100}, ALLOW, SETABLE, NULL},
    {"Number of Exercise Dates", INT, {100}, ALLOW, SETABLE, NULL},
    {" ", PREMIA_NULLTYPE, {0}, FORBID, UNSETABLE, NULL}
  },
  CALC(MC_Lelong_dual),
  { {"Price", DOUBLE, {100}, FORBID, SETABLE, NULL},
    {" ", PREMIA_NULLTYPE, {0}, FORBID, SETABLE, NULL}
  },
  CHK_OPT(MC_Lelong_dual),
  CHK_mc,
  MET(Init),
  0,
  "mc_lelong_dual"
};
}
```