

[Help](#)

```
#include "
href../../mod/bs1d/bs1d_doublim/bs1d_doublim_h_src.pdfdbl.h"

static NumFunc_1 call =
{
    Call,
    {"Strike", PDOUBLE, {100}, ALLOW, SETABLE}, {" ", PREMIA_NULLTYPE, {0}, FORBI
    CHK_call
};

static NumFunc_1 const_Re =
{
    Const,
    {"Const Rebate", DOUBLE, {100}, ALLOW, UNSETABLE}, {" ", PREMIA_NULLTYPE, {0}
    CHK_ok
};

static NumFunc_1 const_Low =
{
    Const,
    { {"Lower Limit", PDOUBLE, {100}, ALLOW, SETABLE},
      {"Delay", SPDOUBLE, {0}, ALLOW, SETABLE},
      {" ", PREMIA_NULLTYPE, {0}, FORBID, SETABLE}
    },
    CHK_call
};

static NumFunc_1 const_Up =
{
    Const,
    { {"Upper Limit", PDOUBLE, {100}, ALLOW, SETABLE},
      {"Delay", SPDOUBLE, {0}, ALLOW, UNSETABLE},
      {" ", PREMIA_NULLTYPE, {0}, FORBID, SETABLE}
    },
    CHK_call
};

static TYPEOPT ParisianDoubleCallInEuro =
```

```

{
    /*PayOff*/          {"PayOff", NUMFUNC_1, {0}, FORBID, SETABLE},
    /*Rebate*/          {"Const Rebate", NUMFUNC_1, {0}, FORBID, UNSETABLE},
    /*LowerLimit*/      {"Lower Limit", NUMFUNC_1, {0}, FORBID, SETABLE},
    /*UpperLimit*/      {"Upper Limit", NUMFUNC_1, {0}, FORBID, SETABLE},
    /*Maturity*/        {"Maturity", DATE, {0}, ALLOW, SETABLE},
    /*DateBetween0andMaturity*/ {"DateBetween0andMaturity", DATE, {0}, FORBID, UNSE
    /*OutOrIn*/         {"In", BOOL, {IN}, FORBID, UNSETABLE},
    /*Parisian*/        {"Parisian", BOOL, {TRUE}, FORBID, UNSETABLE},
    /*TwoDoubleStep*/   {"TwoDoubleStep", BOOL, {FALSE}, FORBID, UNSETABLE},
    /*RebNo*/           {"Rebate", BOOL, {REBATE}, FORBID, UNSETABLE},
    /*EuOrAm*/          {"Euro", BOOL, {EURO}, FORBID, UNSETABLE}

};

```

```

/* For double parisian options, the same delay must be used for lower and
 * upper barriers. The value of the delay is deduced from the one associated
 * to the Lower barrier Numfunc */

```

```

static int OPT(Init)(Option *opt, Model *mod)
{
    TYPEOPT *pt = (TYPEOPT *) (opt->TypeOpt);

    if (opt->init == 0)
    {
        opt->init = 1;
        opt->nvar = 11;
        opt->nvar_setable = 4;

        pt->PayOff.Val.V_NUMFUNC_1 = &call;
        pt->Rebate.Val.V_NUMFUNC_1 = &const_Re;
        pt->LowerLimit.Val.V_NUMFUNC_1 = &const_Low;
        pt->UpperLimit.Val.V_NUMFUNC_1 = &const_Up;

        (pt->EuOrAm).Val.V_BOOL = EURO;
        (pt->OutOrIn).Val.V_BOOL = IN;
        (pt->RebOrNo).Val.V_BOOL = NOREBATE;
        (pt->Maturity).Val.V_DATE = 1.0;

        (pt->PayOff.Val.V_NUMFUNC_1)->Par[0].Val.V_PDOUBLE = 100.0;
        (pt->Rebate.Val.V_NUMFUNC_1)->Par[0].Val.V_PDOUBLE = 0.0;
    }
}

```

```

(pt->LowerLimit.Val.V_NUMFUNC_1)->Par[0].Val.V_PDOUBLE = 90.0;
(pt->UpperLimit.Val.V_NUMFUNC_1)->Par[0].Val.V_PDOUBLE = 110.0;
(pt->LowerLimit.Val.V_NUMFUNC_1)->Par[1].Val.V_SPDOUBLE = 0.01;
(pt->UpperLimit.Val.V_NUMFUNC_1)->Par[1].Val.V_SPDOUBLE = 0.01;

/* test for setability */
if ((pt->RebOrNo).Val.V_BOOL == REBATE)
    pt->Rebate.Vsetable = SETABLE;
else
    pt->Rebate.Vsetable = UNSETABLE;

}

return OK;
}

MAKEOPT(ParisianDoubleCallInEuro);

```