

## Help

```
#include "bs1d_doublim.h"
#define INC 1.0e-5 /*Relative Increment for Delta-Hedging*/

static int Call_KunitomoIkeda_91(double s, NumFunc_1 *L, NumFunc_1 *U, NumFunc_1
{
    double out_price, out_delta, price_plus, price_minus;

    CallOut_KunitomoIkeda_91(s, L, U, Rebate, PayOff, t, r, divid, sigma, &out_pri

    /*Price*/
    *ptprice = out_price;

    CallOut_KunitomoIkeda_91(s * (1. + INC), L, U, Rebate, PayOff, t, r, divid, si
    price_plus = out_price;
    CallOut_KunitomoIkeda_91(s * (1. - INC), L, U, Rebate, PayOff, t, r, divid, si
    price_minus = out_price;

    /*Delta*/
    *ptdelta = (price_plus - price_minus) / (2.*s * INC);

    return OK;
}

int CALC(CF_CallOut_KunitomoIkeda)(void *Opt, void *Mod, PricingMethod *Met)
{
    TYPEOPT *ptOpt = (TYPEOPT *)Opt;
    TYPEMOD *ptMod = (TYPEMOD *)Mod;
    double r, divid;

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

    return Call_KunitomoIkeda_91(ptMod->S0.Val.V_PDOUBLE, ptOpt->LowerLimit.Val.V
        ptOpt->Rebate.Val.V_NUMFUNC_1, ptOpt->PayOff.Val
        r, divid, ptMod->Sigma.Val.V_PDOUBLE, &(Met->Res

}

static int CHK_OPT(CF_CallOut_KunitomoIkeda)(void *Opt, void *Mod)
{
```

```

Option *ptOpt = (Option *)Opt;
TYPEOPT *opt = (TYPEOPT *) (ptOpt->TypeOpt);

if ((opt->Parisian).Val.V_BOOL == FALSE)
    if ((opt->RebOrNo).Val.V_BOOL == NOREBATE)
        return strcmp(((Option *)Opt)->Name, "DoubleCallOutEuro");
return WRONG;
}

```

```

static int MET(Init)(PricingMethod *Met, Option *Opt)
{
    if (Met->init == 0)
    {
        Met->init = 1;
    }
    return OK;
}

```

```

PricingMethod MET(CF_CallOut_KunitomoIkeda) =
{
    "CF_CallOut_KunitomoIkeda",
    {{" ", PREMIA_NULLTYPE, {0}, FORBID}},
    CALC(CF_CallOut_KunitomoIkeda),
    {{"Price", DOUBLE, {100}, FORBID}, {"Delta", DOUBLE, {100}, FORBID} , {" ", PR
    CHK_OPT(CF_CallOut_KunitomoIkeda),
    CHK_ok,
    MET(Init)
} ;

```