

[Help](#)

```

#include "bs1d_pad.h"

static int TurnbullWakeman_FixedAsian(double pseu
    do_stock,double pseudo_strike,NumFunc_2 *po,
    double t,double r,double divid,double sigma,double *pt
    price,double *ptdelta)
{
    double CTtK,PTtK,Dlt,Plt;
    double rt, sgt , m1, m2, m3, m4, meanlog, v, k1
        , k2, k3, k4;
    double as0, as1, as2, k1a, k2a, k3a, k4a;
    double m1a, m2a, m3a, m4a;
    double k;
    double d1,d2,esp,nd1,nd2,v1,levy_price;

    /*Scaling of the param*eters*/
    rt = (r-divid) * t;
    sgt = sigma * sqrt(t);
    k = t * pseudo_strike/pseudo_stock;

    /*Computation of the first four moments*/
    m1 = Moments(1, rt, sgt, 1.0);
    m2 = Moments(2, rt, sgt, 1.0);
    m3 = Moments(3, rt, sgt, 1.0);
    m4 = Moments(4, rt, sgt, 1.0);

    /*Computation of the cumulants of the arithmet
        ic average*/
    k1 = m1;
    k2 = m2 - m1 *m1;
    k3 = m3 - 3 * m1 * m2 + 2 * m1 * m1 * m1;
    k4 = m4 - 4 * m3 * m1 - 3*m2*m2+12*m2*m1*m1-6 *
        m1 * m1 * m1 * m1;
    /*k4 = m4 - 4 * m3 * m1 + 6 * m2 * m1 * m1 - 3
        * m1 * m1 * m1 * m1 - 3 * k2 * k2;*/

    /*Fit the parameters meanlog,v of lognormal dis
        tribution*/
    meanlog = 2.0 * log(m1) - log(m2) / 2.0;
    v = log(m2) - 2 * log(m1);

```

```

/*Computation of lognormal density and its derivatives*/
as0 = logdens(k, meanlog, sqrt(v));
as1 = Der1Logdens(k, meanlog, sqrt(v));
as2 = Der2Logdens(k, meanlog, sqrt(v));

/*Levy Formula*/
v1=sqrt(v);
d1=(log(pseudo_stock/pseudo_strike)+meanlog+SQ
  R(v1))/v1;

d2=d1-v1;
esp=meanlog+SQR(v1)/2.0-(r-divid)*t;
nd1=N(d1);
nd2=N(d2);
levy_price=pseudo_stock*exp(-divid*t)*exp(esp)*
  nd1-exp(-r*t)*pseudo_strike*nd2;

/*Edgeworth Adjustment : Computation of theoretical moments of the
lognormal density*/
m1a = momlog(1, meanlog, v);
m2a = momlog(2, meanlog, v);
m3a = momlog(3, meanlog, v);
m4a = momlog(4, meanlog, v);

/*Edgeworth Adjustment : Computation of theoretical cumulants of the
lognormal density*/
k1a = m1a;
k2a = m2a - m1a * m1a ;
k3a = m3a - 3 * m1a * m2a + 2 * m1a * m1a * m1a;
/*k4a = m4a - 4 * m3a * m1a + 6 * m2a * m1a *
  m1a - 3 * m1a * m1a * m1a * m1a - 3 * k2a * k2a;*/
k4a = m4a - 4 * m3a * m1a - 3*m2a*m2a+12*m2a*
  m1a*m1a-6 * m1a * m1a * m1a * m1a;

/* Call Price */
CTtK=levy_price+pseudo_stock*(exp(-r * t) * -(

```

```

    k3 - k3a) * as1/6.0 +((k4 - k4a) + 3.0 * (k2 - k
    2a)) * as2 / 24.0))/t;

/* 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)*(exp((r-divid)*t)-1.)/(t*(r-divid));

/*Delta for call option*/
Dlt=exp(-divid*t)*exp(esp)*nd1-(exp(-r * t) * (
    -(k3 - k3a) * as1/6.0 +((k4 - k4a) + 3.0 * (k2 -
    k2a)) * as2 / 24.0))/t;

/*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 ((po->Compute)==&Call_OverSpot2)
*ptprice=CTtK;
else
*ptprice=PTtK;

/*Delta */
if ((po->Compute)==&Call_OverSpot2)
*ptdelta=Dlt;
else
*ptdelta=Plt;

return OK;
}

int CALC(AP_FixedAsian_TurnbullWakeman)(void *
    Opt,void *Mod,PricingMethod *Met)

```

```

{
    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->Pat
            hDep.Val.V_NUMFUNC_2)->Par[0].Val.V_PDOUBLE)/(pt
            Opt->Maturity.Val.V_DATE-(ptOpt->PathDep.Val.V_
            NUMFUNC_2)->Par[0].Val.V_PDOUBLE);
        pseudo_spot=(1.-time_spent)*ptMod->S0.Val.V_
            PDOUBLE;
        pseudo_strike=(ptOpt->PayOff.Val.V_NUMFUNC_2
            )->Par[0].Val.V_PDOUBLE-time_spent*(ptOpt->PathD
            ep.Val.V_NUMFUNC_2)->Par[4].Val.V_PDOUBLE;

        if (pseudo_strike<=0.){
            Fprintf(TOSCREEN,"ANALYTIC FORMULA{n{n{n"});
            return_value=Analytic_KemnaVorst(pseudo_sp
            ot,pseudo_strike,time_spent,ptOpt->PayOff.Val.V_
            NUMFUNC_2,ptOpt->Maturity.Val.V_DATE-ptMod->T.Val.V_DA

```

```

    TE,r,divid,&(Met->Res[0].Val.V_DOUBLE),&(Met->Res[
    1].Val.V_DOUBLE));
}
else
return_value= TurnbullWakeman_FixedAsian(ps
eudo_spot,pseudo_strike,ptOpt->PayOff.Val.V_
NUMFUNC_2,ptOpt->Maturity.Val.V_DATE-ptMod->T.Val.V_DA
TE,r,divid,ptMod->Sigma.Val.V_PDOUBLE,&(Met->Res[0
].Val.V_DOUBLE),&(Met->Res[1].Val.V_DOUBLE));
}

return return_value;
}

int CHK_OPT(AP_FixedAsian_TurnbullWakeman)(void *
Opt, void *Mod)
{
if ( (strcmp(((Option*)Opt)->Name,"
AsianCallFixedEuro")==0) || (strcmp( ((Option*)Opt)->Name,"
AsianPutFixedEuro")==0) )
return OK;
return WRONG;
}

static int MET(Init)(PricingMethod *Met)
{
return OK;
}

PricingMethod MET(AP_FixedAsian_TurnbullWakeman)=
{
"AP_FixedAsian_TurnbullWakeman",
{" ",END,0,FORBID},
CALC(AP_FixedAsian_TurnbullWakeman),
{"Price",DOUBLE,100,FORBID},{"Delta",DOUBLE,10
0,FORBID} ,{" ",END,0,FORBID}},
CHK_OPT(AP_FixedAsian_TurnbullWakeman),
CHK_ok,
MET(Init)
};

```

## References