Help

```
#include <iostream>
#include <cmath>
#include <cstdlib>

using namespace std;

#include "
href../../../../common/math/mcam/src/DupireModel_h_src.pdfDupireModel.hpp"
#include "
href../../../../common/math/jlparser/include/jlparser/parser_h_src.pdfjlparser/p

inline static double sq(double x)
{
    return x * x;
}

mcam::DupireModel::DupireModel()
    : Model()
{
}

mcam::DupireModel::~DupireModel()
{
    if (Cov_Chol) pnl_mat_free(&Cov_Chol);
}

mcam::DupireModel::DupireModel(const Param& P)
    : Model(P)
{
    brownianSize = size;
    P.extract("correlation", rho);
    /* test the value of the correlation */
    if (rho > 1 || rho < -1.0 / (size - 1))
    {
        perror("correlation out of range");
        exit(1);
    }
    Cov_Chol = pnl_mat_create(size, size);
    Rho(rho);
```

```
}

/* set the value of the correlation parameter and update the
 * covariance matrix and its derivative */
void mcam::DupireModel::Rho(double rho)
{
    pnl_mat_set_all(Cov_Chol, rho);
    pnl_mat_set_diag(Cov_Chol, 1., 0);
    pnl_mat_chol(Cov_Chol);
}

// /*
//  * Volatility function
//  * dS_t = interest S_t dt + sigma(t, S_t) dW_t
//  */
// double mcam::DupireModel::sigma (double maturity, double t, double St, int i)
// {
//   return 0.6*(1.2-exp(-.1*(t))*exp(-0.1e-2 * sq(St*exp(interest*(maturity-t))
//     *exp(-0.5e-1*sqrt(t)) * St;
// }

/*
 * Black Scholes Volatility function for testing purposes
 * dS_t = interest S_t dt + sigma(t, S_t) dW_t
 */
double mcam::DupireModel::sigma(double maturity, double t, double St, int i) con
{
    return 0.2 * St;
}

/*
 * Volatility function
 * dS_t = interest S_t dt + sigma(t, S_t) dW_t
 */
// double mcam::DupireModel::sigma (double maturity, double t, double St, int i)
// {
//   /* return (0.01 + 0.1*exp(-St/100)+0.01*t)*St; */
//   return 15.0;
// }

void mcam::DupireModel::path(const PnlMat* G)
```

```
{
    int j;
    double tj;
    // Time 0
    PnlMat* m = path_m();
    PnlMat* w = work();
    pnl_mat_resize(m, subdates + 1, size);
    pnl_mat_set_row(m, spot, 0);
    pnl_mat_resize(w, subdates, size);
    pnl_mat_dgemm('N', 'T', sqrt_dt, G, Cov_Chol, 0., w);

    for (j = 0, tj = 0; j < subdates; j++, tj += dt)
    {
        for (int i = 0; i < size; i++)
        {
            MLET(m, j + 1, i) = MGET(m, j, i) * (1 + (r - GET(dividend, i)) * dt
        }
    }
}

PnlVect * mcam::DupireModel::getMin() const
{
    PnlVect *min_v = pnl_vect_create (size);
    for (int i = 0; i < size; i++)
    {
        double sig = sigma(T, T, GET(spot,i), i) / GET(spot,i);
        LET(min_v, i) = GET(spot,i) * exp (MIN(r - sig*sig/2, 0) * T - sig * 4.0
    }
    return min_v;
}

PnlVect * mcam::DupireModel::getMax() const
{
    PnlVect *max_v = pnl_vect_create (size);
    for (int i = 0; i < size; i++)
    {
        double sig = sigma(T, T, GET(spot,i), i) / GET(spot,i);
        LET(max_v, i) = GET(spot,i) * exp (MAX(r - sig*sig/2, 0) * T + sig * 4.0
    }
    return max_v;
}
```

```
void
mcam::DupireModel::print() const
{
    cout << "**** Local Volatility Model Characteristics ****" << endl;
    mcam::Model::print();
}
```