Help

```
#pragma once
#include <pnl/pnl_mathtools.h>
#include "pnl/pnl_specfun.h"


class Model_proj {
protected:
double r;  // interest rate
double q;  // divident yield

double om = 0.0;  // Represents a contour shift of the model

// Cumulants
double c1;
double c2;
double c4;

double RNmu; // risk neutral drift rate

public:
virtual ~Model_proj() = default;
Model_proj(double r, double q, double om=0.0)
: r(r), q(q), om(om)
{
}
virtual void rn_chf(double t, double u, dcomplex* res) const = 0;
double get_truncation_alpha(double L1, double T) const;
double get_c1() const { return c1; };
double get_interest_rate() const { return r; };
double get_div_yield() const { return q; };
double get_countour_shift_om() const { return om; };
double get_rn_drift() const { return RNmu; };  // Retrieve risk neutral drift ra
};

////////////////////////////////////////////////

class CGMY_Model_proj : public Model_proj {
private:
double C;
```

```
double G;
double M;
double Y;

double RePart;

void setParams(double C, double G, double M, double Y, double om = 0.0);

public:
void rn_chf(double t, double u, dcomplex* res) const;

CGMY_Model_proj(double r, double q, double C, double G, double M, double Y, doub
Model_proj(r, q, om)
{
setParams(C, G, M, Y, om);
}
};

////////////////////////////////////////////////

class BlackScholes_Model_proj : public Model_proj {
private:
double sigma;

public:
void rn_symbol(double u, dcomplex* res) const;
void rn_chf(double t, double u, dcomplex* res) const;
void bs_roots(double lam, double*bp, double*bm) const;
void cf_phim_BS(double u, double bm, dcomplex *res) const;
void setParams(double sigma, double om = 0.0);


BlackScholes_Model_proj(double r, double q, double sigma, double om = 0.0) :
Model_proj(r, q, om)
{
setParams(sigma, om);
}
};


////////////////////////////////////////////////
```

```
class Kou_Model_proj : public Model_proj {
private:
double sigma;
double lam;
double p_up;
double eta1;
double eta2;
public:
void rn_symbol(double u, dcomplex* res) const;
void rn_chf(double t, double u, dcomplex* res) const;
void setParams(double sigma, double lam, double p_up, double eta1, double eta2,

Kou_Model_proj(double r, double q, double sigma, double lam, double p_up, double
Model_proj(r, q, om)
{
setParams(sigma, lam, p_up, eta1, eta2, om);
}
};

////////////////////////////////////////////////

class NIG_Model_proj : public Model_proj {
private:
double alph;
double bet;
double delt;

double asq, bsq, sqd;  // Precomputed and stored for efficiency in evaluating ch

public:
void rn_symbol(double u, dcomplex* res) const;
void rn_chf(double t, double u, dcomplex* res) const;
void setParams(double alph, double bet, double delt, double om = 0.0);

NIG_Model_proj(double r, double q, double alph, double bet, double delt, double
Model_proj(r, q, om)
{
setParams(alph, bet, delt, om);
}
};
```