Help

```
#ifndef _MOD_H
#define _MOD_H

#include "
href../../../../common/math/jlparser/include/jlparser/parser_h_src.pdfjlparser/p
#include "
href../../../../common/math/mcam/src/workspace_h_src.pdfworkspace.hpp"
#include "pnl/pnl_matrix.h"
#include "pnl/pnl_random.h"

namespace mcam {

class Model
{
public:
    double T; //!< maturity of the option
    int size; //!< size of the model
    int brownianSize; //!< size of the driving BM
    int dates;   //!< number of exercising dates
    int subticks; //!< number of sub ticks between two exercising dates
    int subdates; //!< number of dates on the finer grid
    double dt; //!< T / subdates
    double sqrt_dt; //!< sqrt(T / subdates)
    double r;     //!< instantaneous interest rate
    PnlVect *dividend;    //!< instantaneous dividend rate
    PnlVect *spot;   //!< spot values
    PnlMat_Workspace path_m; //!< It contains a path of the model after calling
    PnlMat_Workspace regressor_m; //!< It contains a path of the regressors afte

public:
    Model();
    Model(const Param &P);
    virtual ~Model();

    double discount(int t) const;
    virtual PnlVect* getMin() const = 0;
    virtual PnlVect* getMax() const = 0;

    /**
```

```
     * Compute a path of the model on the finer grid and store it in #path_m
     *
     * @param G is a matrix of std normal r.v. with size ((dates * subticks) x *
     */
    virtual void path(const PnlMat *G) = 0;
    virtual void print() const = 0;
    virtual PnlMat* getPath() const;
    virtual PnlMat* getRegressor() const;

protected:
    PnlMat_Workspace work;   /*!< Workspace for intermediate computations */
};

Model *instantiate_model(const Param &P);
}
#endif

/* vim: set ft=cpp: */
```