Help

```
#ifndef _CHAOS_HPP
#define _CHAOS_HPP

#include "pnl/pnl_matrix.h"
#include "pnl/pnl_basis.h"

#include "
href../../../../common/math/jlparser/include/jlparser/parser_h_src.pdfjlparser/p
#include "
href../../../../common/math/mcam/src/Martingale_h_src.pdfMartingale.hpp"

namespace mcam {
/**
 * @class Chaos.
 * @brief Winer chaos expansion
 *
 * Implementation of the Wiener chaos expansion using the Hermite polynomials, s
 * terms are orthogonal and computing the conditional expectation just boils dow
 * terms.
 */
class Chaos: public Martingale
{
public:

    int order; //!< order of the chaos expansion
    PnlBasis *Basis;

    Chaos();
    Chaos(const Param &P, Model *M, bool prune);
    /**
     * Constructor
     *
     * @param T time horizon in terms of ticks
     * @param size brownian motion size
     * @param order order of the decomposition
     */
    Chaos(int T, int size, int order);
    ~Chaos();
    void print() const;
```

```
    /**
     * Computes the path of the martingale and its gradient. The results are sto
     * Mgrad.
     *
     * @param G The Brownian increments with size (subdates x size). This choice
     * is essential to silently ignore the last lines of G corresponding to incr
     * time T
     * @param alpha the coefficients of the expansion
     */
    virtual void computePath(const PnlMat *G, const PnlVect *alpha);

private:
    int *colMax; //!< Hold the maximum index of non elements in Basis->T
    void initColMax();
};
}

#endif /* _CHAOS_HPP */
```