Help

```
/*
 * RfBm.h
 *
 * Generate samples from W1 and Wtilde jointly as provided in compute_Wtilde in
 *
 * n denotes the number N in rBergomi.cpp, i.e., the timestep is 1/n.
 *
 *  Created on: 30 Jan 2018
 *      Author: bayerc
 */

#ifndef RFBM_H_
#define RFBM_H_

#include "pnl/pnl_random.h"
#include <
href../../../common/math/highdim_solver/highdim_vector_h_src.pdfvector>

typedef std::vector<double> Vector;

class RfBm {
private:
int n;
double H;
double gamma;
PnlRng *rng_;
std::vector<Vector> L; // Cholesky factorization of the covariance matrix.
Vector x_; //  used to store a normal random vector
PnlVect wrap_x_; // a PnlVect wrapper on x_
double A(int i, int j) const; // covariance matrix
double G(double x) const; // function G as in Benjamin's write-up
// individual parts of covariance matrix
double cBm(int i, int j) const; // covariance of scaled Brownian increments
double cfBm(int i, int j) const; // covariance between fBm at time (i+1)/n and (
double cMixed(int i, int j) const; // covariance between \ sqrt{n} (W_{(i+1)/n}
public:
RfBm();
RfBm(int nI, double HI, PnlRng *rng);
void generate(Vector& W1, Vector& Wtilde);
```

```
void operator()(Vector& W1, Vector& Wtilde);
std::vector<Vector> GetL() const {return L;}
std::vector<Vector> GetA() const;
};

#endif /* RFBM_H_ */
```