Help

```
#ifndef _BNSModel_H
#define _BNSModel_H

#include "math/ImportanceSampling_jl/src/Model.hpp"
#include "pnl/pnl_matrix.h"

/**
  @class BNSModel

  The volatility process follows
  \ f[
  d\ sigma_t^2 = - \ kappa \ sigma^2_t dt + dN_{\ kappa t}
  \ f]
  where \ f$ N \ f$ is a compound poissonMat process with jump intensity \ f$ \
  jump size following an exponential distribution with parameter \ f$ \ beta \ f
  is the mean reversion of the squared volatility process.

  The log price X is defined by
  \ f[
  dX_t = (interest - q - \ kappa k(-\ rho) - \ sigma^2/2) dt + \ sigma_t dW_t  +
  \ f]
  with \ f$k(u) = \ lambda u / (\ beta + u) \ f$.
*/
class BNSModel : public JumpModel
{
public:
  PnlVect *beta; /*!< the jump size has an exp(beta) distribution */
  PnlVect *kappa; /*!< mean reversion of the vol */
  PnlVect *leverage; /*!< rho dN_t */

  BNSModel();
  BNSModel(const Param &P);
  void print() const;
  virtual void path();

  ~BNSModel();

protected:
  virtual void pathMu_aux(PnlRng *rng, const PnlVect *mu);
```

```
  virtual void pathMuFull_aux(PnlRng *rng, const PnlVect *mu);

private:
  PnlVect *Vol; /*!< vector of squared volatility at a given time */
};

#endif
```