Help

```
#if defined(PremiaCurrentVersion) && PremiaCurrentVersion < (2008+2) //The "#els
#else
#ifndef FD_SOLVER_COMMON_H
#define FD_SOLVER_COMMON_H

#include <stdio.h>

#include <
href../../../common/math/highdim_solver/laspack/highdim_matrix_h_src.pdfmath/hig
#include <
href../../../common/math/highdim_solver/laspack/qmatrix_h_src.pdfmath/highdim_so
#include <
href../../../common/math/highdim_solver/laspack/highdim_vector_h_src.pdfmath/hig
#include <
href../../../common/math/highdim_solver/laspack/errhandl_h_src.pdfmath/highdim_s

//////////////////////////////////////////////////
//
// Implementation of a finite differences solver
// for PDEs.
//

#define FDSOLVERMAXDIM  20

typedef int FDBOOL;

#define TRUE 1
#define FALSE 0

struct _FDSolver;

struct _FDSolverVectorFiller;

typedef int (*FDSolverVectorFillerInit_t)(struct _FDSolver *,
    struct _FDSolverVectorFiller *);

typedef int (*FDSolverVectorFillerNextElem_t)(struct _FDSolver *,
    struct _FDSolverVectorFiller *,
    unsigned *, double *);
```

```
typedef int (*FDSolverVectorFillerFinish_t)(struct _FDSolver *,
    struct _FDSolverVectorFiller *);

typedef void (*FDSolverVectorFillerFree_t)(struct _FDSolver *,
    struct _FDSolverVectorFiller *);


typedef struct _FDSolverVectorFiller
{

  FDSolverVectorFillerInit_t init;
  FDSolverVectorFillerNextElem_t next_elem;
  FDSolverVectorFillerFinish_t finish;
  FDSolverVectorFillerFree_t free;

  void *data;

} FDSolverVectorFiller;


struct _FDSolverCoMatricesFiller;

typedef int (*FDSolverCoMatricesFillerInit_t)(
  struct _FDSolver *,
  struct _FDSolverCoMatricesFiller *
);

typedef int (*FDSolverCoMatricesFillerNextRow_t)(
  struct _FDSolver *,
  struct _FDSolverCoMatricesFiller *,
  unsigned *, unsigned *
);

typedef int (*FDSolverCoMatricesFillerNextElem_t)(
  struct _FDSolver *,
  struct _FDSolverCoMatricesFiller *,
  unsigned *, double *, unsigned *
);

typedef int (*FDSolverCoMatricesFillerFinish_t)(
```

```
  struct _FDSolver *,
  struct _FDSolverCoMatricesFiller *
);

typedef void (*FDSolverCoMatricesFillerFree_t)(
  struct _FDSolver *,
  struct _FDSolverCoMatricesFiller *
);

typedef struct _FDSolverCoMatricesFiller
{

  FDSolverCoMatricesFillerInit_t init;
  FDSolverCoMatricesFillerNextRow_t next_row;
  FDSolverCoMatricesFillerNextElem_t next_elem;
  FDSolverCoMatricesFillerFinish_t finish;
  FDSolverCoMatricesFillerFree_t free;

  void *data;

} FDSolverCoMatricesFiller;

typedef struct _FDSolverCoordWalkerData
{

  unsigned coord[FDSOLVERMAXDIM];
  unsigned *pl;
  unsigned *ph;
  unsigned *sh;
  unsigned *f;
  unsigned *first, *size, dim;

} FDSolverCoordWalkerData;


typedef struct _FDSolver
{
  // Common data
  unsigned dim;
  unsigned size[FDSOLVERMAXDIM];
  unsigned offsA[FDSOLVERMAXDIM];
```

```
  unsigned offsB[FDSOLVERMAXDIM];  // TODO: Is it useful?

  QMatrix Ac, An;
  Matrix Bc, Bn;
  Vector x1, x2, b1, b2;
  Vector *xc, *xn;
  Vector *bc, *bn;

  FDBOOL is_A_symmetric;
  FDBOOL is_fully_explicit;
  FDBOOL is_fully_implicit;

  // PDE state
  double t, deltaT;

  // PDE specific routines
  FDSolverVectorFiller *b_filler;    // Boundary condition

  // Problem-specific data
  void *data;

  // Internal data
  FDSolverCoordWalkerData xwd;
  unsigned xidx, bidx;

} FDSolver;


void FD_SLICE_WALKER_UPDATE(FDSolverCoordWalkerData *wd, unsigned *state, int *n
void FD_SLICE_WALKER_RESET(FDSolverCoordWalkerData *wd, unsigned idim, unsigned


#define FD_WALKER_RESET(s,wd) \
  FD_SLICE_WALKER_RESET(wd,(s)->dim,NULL,(s)->size)

#define FD_WALKER_UPDATE(wd) FD_SLICE_WALKER_UPDATE(wd,NULL,NULL)

#define FD_WALKER_ON_BOUNDARY(wd)                                    \
  (*((wd)->pl) == ((wd)->f ? *((wd)->f) : 0)                         \
   ||                                                                \
   *((wd)->ph) ==  ((wd)->f ? *((wd)->f) : 0) + *((wd)->sh) - 1)  \
```

```
#endif

#endif //PremiaCurrentVersion
```