Help

```
#if defined(PremiaCurrentVersion) && PremiaCurrentVersion <
     (2008+2) //The "#else" part of the code will be freely av
    ailable after the (year of creation of this file + 2)
#else
/*************************************************************
    ******************/
/*                                   itersolv.h
                     */
/*************************************************************
    ******************/
/*
                     */
/* ITERative SOLVers for systems of linear equations
                     */
/*
                     */
/* Copyright (C) 1992-1995 Tomas Skalicky. All rights res
    erved.              */
/*
                     */
/*************************************************************
    ******************/
/*
                     */
/*        ANY USE OF THIS CODE CONSTITUTES ACCEPTANCE OF TH
    E TERMS           */
/*               OF THE COPYRIGHT NOTICE (SEE FILE copyrght.h
    )                */
/*
                     */
/*************************************************************
    ******************/

#ifndef ITERSOLV_H
#define ITERSOLV_H

#include "laspack/highdim_vector.h"
#include "laspack/qmatrix.h"
#include "laspack/precond.h"
#include "laspack/eigenval.h"
```

```
#include "laspack/copyrght.h"

typedef Vector *(*IterProcType)(QMatrix *, Vector *, Vector
     *, int,
        PrecondProcType, double);

/* classical iterative methods */

Vector *JacobiIter(QMatrix *A, Vector *x, Vector *b, int
    NoIter,
      PrecondProcType Dummy, double Omega);
Vector *SORForwIter(QMatrix *A, Vector *x, Vector *b, int
    NoIter,
            PrecondProcType Dummy, double Omega);
Vector *SORBackwIter(QMatrix *A, Vector *x, Vector *b, int
    NoIter,
            PrecondProcType Dummy, double Omega);
Vector *SSORIter(QMatrix *A, Vector *x, Vector *b, int NoI
    ter,
            PrecondProcType Dummy, double Omega);

/* semi-iterative methods */

Vector *ChebyshevIter(QMatrix *A, Vector *x, Vector *b,
    int MaxIter,
            PrecondProcType PrecondProc, double OmegaPrecon
    d);

/* CG and CG-like methods */

Vector *CGIter(QMatrix *A, Vector *x, Vector *b, int Max
    Iter,
            PrecondProcType PrecondProc, double OmegaPrecon
    d);
Vector *CGNIter(QMatrix *A, Vector *x, Vector *b, int Max
    Iter,
            PrecondProcType PrecondProc, double OmegaPrecon
    d);
Vector *GMRESIter(QMatrix *A, Vector *x, Vector *b, int Max
    Iter,
            PrecondProcType PrecondProc, double OmegaPrecon
```

```
    d);
Vector *BiCGIter(QMatrix *A, Vector *x, Vector *b, int Max
    Iter,
            PrecondProcType PrecondProc, double OmegaPrecon
    d);
Vector *QMRIter(QMatrix *A, Vector *x, Vector *b, int Max
    Iter,
            PrecondProcType PrecondProc, double OmegaPrecon
    d);
Vector *CGSIter(QMatrix *A, Vector *x, Vector *b, int Max
    Iter,
            PrecondProcType PrecondProc, double OmegaPrecon
    d);
Vector *BiCGSTABIter(QMatrix *A, Vector *x, Vector *b, int
    MaxIter,
            PrecondProcType PrecondProc, double OmegaPrecon
    d);
void SetGMRESRestart(int MaxSteps);

#endif /* ITERSOLV_H */

#endif //PremiaCurrentVersion
```

References