Test of the linear systems from the convection-diffusion equations

This package provides implementations of comparing BICGSTAB, GMRES, DQGMRES, FOM, DIOM, SCG, SWI, RGMRES, RFOM, RSCG, and DSWI for solving the linear systems from the convection-diffusion equations:

$$Ax = b,$$

where $A \in \mathbb{R}^{n\times n}$ is unsymmetric positive definite.

Contents

BICGSTAB

Using Matlabs function "bicgstab" directly.

RGMRES

RGMRES (Restarted GMRES): Algorithm 6.11 in Yousef Saad's "Iterative Methods for Sparse Linear System (2nd Edition)"

```
function [x, k, res, resvec] = regmres(A, b, restart, tol, x0, maxit)
```

• [x] = regmres(A, b) attempts to find a solution x to the system of linear equations $Ax = b$. The n-by-n coefficient matrix A must be positive definite but need not be symmetric. The right hand side column vector b must have length n.
• [x] = regmres(A, b, restart) specifies the restarted number. If restart is [ ] then resgmres uses the default, n
• [x] = regmres(A, b, restart, tol) specifies the tolerance of the method. If tol is [ ] then resgmres uses the default, 1e-6.
• [x] = regmres(A, b, restart, tol, x0) specifies the initial guess. If x0 is [ ] then resgmres uses the default, an all zero vector.
• [x] = regmres(A, b, restart, tol, x0, maxit) specifies the maximum number of iterations. If maxit is [ ] then regmres uses the default, 10000.

• [x, k] = regmres(A, b, ...) returns the iteration number at which x was computed: $1 \leq k \leq$ maxit.
• [x, k, res] = regmres(A, b, ...) also returns the last relative residual norm $\|b - Ax\|/\|b\|$.
• [x, k, res, resvec] = regmres(A, b, ...) also returns a vector of estimates of the residual norms at each iteration, including $\|b - Ax\|$.

GMRES: set $restart = n$ in the function "regmres".

DQGMRES

DQGMRES (Direct Quasi-GMRES): Algorithms 6.6 and 6.13 in Yousef Saad's "Iterative Methods for Sparse Linear System (2nd Edition)"

```
function [x, k, res, resvec] = dqgmres(A, b, m, tol, x0, maxit)
```

• [x] = dqgmres(A, b) attempts to find a solution x to the system of linear equations $Ax = b$. The n-by-n coefficient matrix A must be positive definite but need not be symmetric. The right hand side column vector b must have length n.
• [x] = dqgmres(A, b, m) specifies the number of the sliding window. If m is [ ] then dqgmres uses the default, n.
• [x] = dqgmres(A, b, m, tol) specifies the tolerance of the method. If tol is [ ] then dqgmres uses the default, 1e-6.
• [x] = dqgmres(A, b, m, tol, x0) specifies the initial guess. If x0 is [ ] then dqgmres uses the default, an all zero vector.
• [x] = dqgmres(A, b, m, tol, x0, maxit) specifies the maximum number of iterations. If maxit is [ ] then dqgmres uses the default, 10000.
• [x, k] = dqgmres(A, b, ...) returns the iteration number at which x was computed: $1 \leq k \leq$ maxit.
• [x, k, res] = dqgmres(A, b, ...) also returns the last relative residual norm $\|b - Ax\|/\|b\|$.
• [x, k, res, resvec] = dqgmres(A, b, ...) also returns a vector of estimates of the residual norms at each iteration, including $\|b - Ax\|$.

RFOM

RFOM (Restarted FOM): Algorithm 6.5 in Yousef Saad's "Iterative Methods for Sparse Linear System (2nd Edition)"

```
function [x, k, res, resvec] = refom(A, b, restart, tol, x0, maxit)
```

• [x] = refom(A, b) attempts to find a solution x to the system of linear equations $Ax = b$. The n-by-n coefficient matrix A must be positive definite but need not be symmetric. The right hand side column vector b must have length n.

- [x] = refom(A, b, restart) specifies the restarted number. If restart is [ ] then refom uses the default, n
- [x] = refom(A, b, restart, tol) specifies the tolerance of the method. If tol is [ ] then refom uses the default, 1e-6.
- [x] = refom(A, b, restart, tol, x0) specifies the initial guess. If x0 is [ ] then refom uses the default, an all zero vector.
- [x] = refom(A, b, restart, tol, x0, maxit) specifies the maximum number of iterations. If maxit is [ ] then refom uses the default, 10000.
- [x, k] = refom(A, b, ...) returns the iteration number at which x was computed: $1 \leq k \leq$ maxit.
- [x, k, res] = refom(A, b, ...) also returns the last relative residual norm $\|b - Ax\|/\|b\|$.
- [x, k, res, resvec] = refom(A, b, ...) also returns a vector of estimates of the residual norms at each iteration, including $\|b - Ax\|$.

FOM: set $restart = n$ in the function "refom".

DIOM(m)

DIOM (Direct Incomplete Orthogonalization Method): Algorithms 6.6 and 6.8 in Yousef Saad's "Iterative Methods for Sparse Linear System (2nd Edition)"

```
function [x, k, res, resvec] = diom(A, b, mk, tol, x0, maxit)
```

- [x] = diom(A, b) attempts to find a solution x to the system of linear equations $Ax = b$. The n-by-n coefficient matrix A must be positive definite but need not be symmetric. The right hand side column vector b must have length n.
- [x] = diom(A, b, m) specifies the number of the sliding window. If m is [ ] then diom uses the default, n.
- [x] = diom(A, b, m, tol) specifies the tolerance of the method. If tol is [ ] then diom uses the default, 1e-6.
- [x] = diom(A, b, m, tol, x0) specifies the initial guess. If x0 is [ ] then diom uses the default, an all zero vector.
- [x] = diom(A, b, m, tol, x0, maxit) specifies the maximum number of iterations. If maxit is [ ] then diom uses the default, 10000.
- [x, k] = diom(A, b, ...) returns the iteration number at which x was computed: $1 \leq k \leq$ maxit.
- [x, k, res] = diom(A, b, ...) also returns the last relative residual norm $\|b - Ax\|/\|b\|$.
- [x, k, res, resvec] = diom(A, b, ...) also returns a vector of estimates of the residual norms at each iteration, including $\|b - Ax\|$.

SWI(m)

SWI: Sliding window implementation with pre-allocated memory

```
[x, k, res, resvec] = swi(A, b, m, tol, x0, maxit)
```

• [x] = swi(A, b) attempts to find a solution x to the system of linear equations $Ax = b$. The n-by-n coefficient matrix A must be positive definite but need not be symmetric. The right hand side column vector b must have length n.
• [x] = swi(A, b, m) specifies the number of the sliding window. If m is [ ] then swi uses the default, n.
• [x] = swi(A, b, m, tol) specifies the tolerance of the method. If tol is [ ] then swi uses the default, 1e-6.
• [x] = swi(A, b, m, tol, x0) specifies the initial guess. If x0 is [ ] then swi uses the default, an all zero vector.
• [x] = swi(A, b, m, tol, x0, maxit) specifies the maximum number of iterations. If maxit is [ ] then swi uses the default, 10000.
• [x, k] = swi(A, b, ...) returns the iteration number at which x was computed: $1 \le k \le$ maxit
• [x, k, res] = swi(A, b, ...) also returns the last relative residual norm $\|b - Ax\|/\|b\|$.
• [x, k, res, resvec] = swi(A, b, ...) also returns a vector of estimates of the residual norms at each iteration, including $\|b - Ax\|$.

SCG: set $m = n$ in the function "swiwp".

DSWI

DSWI: Sliding window implementation with choosing m dynamically

The approach to choose m dynamically is as follows:

$$m = \begin{cases} \max\{[0.8m], 2\}, & \text{if } \|r_k\| < 0.9\|r_{k-1}\|; \\ \min\{2m, n\}, & \text{if } \|r_k\| > 2\|r_{k-1}\|; \\ m, & \text{otherwise.} \end{cases}$$

```
[x, k, res, resvec] = dynswiwp(A, b, m, tol, x0, maxit)
```

• [x] = dynswiwp(A, b) attempts to find a solution x to the system of linear equations $Ax = b$. The n-by-n coefficient matrix A must be positive definite but need not be symmetric. The right hand side column vector b must have length n.
• [x] = dynswiwp(A, b, m) specifies the number of the sliding window. If m is [ ] then dynswiwp uses the default, n.
• [x] = dynswiwp(A, b, m, tol) specifies the tolerance of the method. If tol is [ ] then dynswiwp uses the default, 1e-6.
• [x] = dynswiwp(A, b, m, tol, x0) specifies the initial guess. If x0 is [ ] then dynswiwp uses the default, an all zero vector.
• [x] = dynswiwp(A, b, m, tol, x0, maxit) specifies the maximum number of iterations. If maxit is [ ] then dynswiwp uses the default, 10000.

• [x, k] = dynswiwp(A, b, ...) returns the iteration number at which x was computed: $1 \leq k \leq$ maxit
• [x, k, res] = dynswiwp(A, b, ...) also returns the last relative residual norm $\|b - Ax\|/\|b\|$.
• [x, k, res, resvec] = dynswiwp(A, b, ...) also returns a vector of estimates of the residual norms at each iteration, including $\|b - Ax\|$.

Test

1. Download the files: regmres.m, refom.m, rescg.m, dqgmres.m, diom.m, swi.m, dynswiwp.m, test_condiff.m, A.mat and b.mat;

2. Run the function "test_condiff" directly, i.e.,

```
>> test_condiff
```

```
BICGSTAB          130    0.0045        5.36e-08
GMRES              43    0.0114        4.17e-07
DQGMRES(36)        43    0.0132        5.81e-07
FOM                43    0.0232        4.47e-07
DIOM(2)          5751    0.2528             NaN
DIOM(5)            49    0.0119        8.25e-07
DIOM(10)           60    0.0226        2.33e-07
SCG                43    0.0052        4.47e-07
SWI(2)             71    0.0116        8.28e-07
SWI(5)             52    0.0090        1.58e-07
SWI(10)            63    0.0075        3.50e-07
RGMRES            123    0.0147        7.18e-07
RFOM               85    0.0190        9.29e-07
RSCG               84    0.0103        9.91e-07
DSWI               69    0.0158        6.37e-07
```