`smt` User Guide
a Matlab toolbox for structured matrices

Michela Redivo-Zaglia* Giuseppe Rodriguez†

The `smt` toolbox is described in the paper

> M. Redivo-Zaglia and G. Rodriguez. `smt`: a Matlab toolbox for structured matrices. Numer. Algorithms, 59 (2012), pp. 639–659. DOI: 10.1007/s11075-011-9527-9.

This document includes some additional information, mostly technical, which was not included there.

1 The toolbox

To install the toolbox, it is sufficient to uncompress the archive file containing the software. This creates the directory `smt` and its subtree. This directory must be added to the Matlab search path in order to be able to use the toolbox from any other directory. The command `smtcheck` should be run at this point, to verify that the installation was successful.

The toolbox installation procedure creates the directory tree sketched in Figure 1. The main directory contains a set of general purpose functions, described in detail in Section 2.3, and the following four subdirectories:

- `@smcirc` and `@smtoep`, which contain the functions to create and manipulate the objects of class `smcirc` and `smtoep`, i.e., circulant and Toeplitz matrices;
- `private`, containing some internal functions, discussed in Section 2.3, which are not directly accessible to the user;
- `demo`, which hosts an interactive tutorial on the basic use of the toolbox (see Section 4), the `tests` directory, which contains the scripts used for the numerical tests reported in the paper, and the `validate` directory, discussed in the following.

Full documentation for every function of the toolbox is accessible via the Matlab `help` command, and the code itself is extensively commented. Manual pages can be obtained by the usual Matlab means, i.e.,

---

*Dipartimento di Matematica, via Trieste 63, 35121 Padova, Italy (`Michela.RedivoZaglia@unipd.it`).

†Dipartimento di Matematica e Informatica, Università di Cagliari, viale Merello 92, 09123 Cagliari, Italy (`rodriguez@unica.it`).

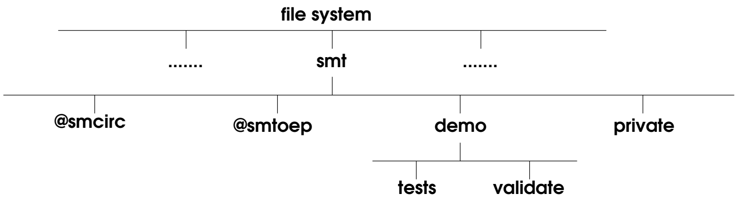

Figure 1: Directory tree of `smt`

`help` <func_name>	for the functions in the main directory,
`help` <class>/<func_name>	where <class> is either `@smcirc` or `@smtoep`,
`help private`/<func_name>	for the functions in the `private` subdirectory.

Notice that <func_name> may be `Contents` (except in conjunction with `private`), in which case a description of the entire directory content is displayed. For example, the command

```
help smtoep/Contents
```

displays the list of all the functions, operators and methods for `smtoep` objects (i.e., Toeplitz matrices), while

```
help @smtoep/mtimes
```

gives information about the matrix product operator for Toeplitz matrices.

The `validate` function, located in the directory `demo/validate` is a useful tool to check that the toolbox is working properly: it runs most of the possible tests on all operators and functions, to verify that the structured routines produce essentially the same results than Matlab standard routines. Enter the command

```
validate([],[],[],1)
```

to see which tests are performed; see the help page for more information.

Let us end this section by a brief explanation of how Matlab deals with new data types. When the user creates an object of class, say, `obj`, then the interpreter looks for the function with the same name in a directory called `@obj`, located in the search path. Similarly, when an expression involves a variable of class `obj`, or a function is applied to it, the corresponding operator or function defined for objects of this class is searched in the same directory.

More information on classes can be found on the chapter on object-oriented programming of Matlab online documentation.

2 Classes and methods

The classes `smcirc` and `smtoep` are intended to store circulant and Toeplitz matrices, respectively.

2.1 The `smcirc` class

A circulant matrix can be created by specifying its first column, with the command

```
C=smcirc([1;2;3;4])
```

and it is visualized either as a matrix

```
C =
     1     4     3     2
     2     1     4     3
     3     2     1     4
     4     3     2     1
```

or showing its record structure

```
C =
smcirc object with fields:
      c: [4x1 double]
    dim: 4
     ev: []
```

depending on how the configuration parameter `display` is set; see the function `smtconfig`.

Matrix operators			
`plus`	`A+B`	`power`	`A.^2`
`uplus`	`+A`	`mldivide`	`A\B`
`minus`	`A-B`	`mrdivide`	`A/B`
`uminus`	`-A`	`ldivide`	`A.\B`
`mtimes`	`A*B`	`rdivide`	`A./B`
`times`	`A.*B`	`transpose`	`A.'`
`mpower`	`A^2`	`ctranspose`	`A'`

Table 1: Overloaded operators

All the *overloaded* operators, or *methods*, for `smcirc` matrices are coded in a set of functions, whose names (fixed by the Matlab syntax) are reported in Table 1, together with the equivalent Matlab notations. Table 2 lists the standard functions which have been redefined for circulant matrices.

The list in Table 2 is surely incomplete, since, in principle, all Matlab matrix functions could be overloaded for circulant matrices. We implemented those functions which we consider useful, leaving an extension of this list, if motivated by real need, to future versions of the package. We note, however, that some functions are simply not essential; for example, if `C` is circulant, then the "`sin`" of its entries can be easily computed by

```
D=smcirc(sin(C.c));
```

without the need to overload the `sin` function for the `smcirc` class.

Let us add some comments on some of the functions listed in Table 2. When a new class is added to Matlab, there is a number of functions which must be defined so that the class conforms to Matlab syntax rules. The `get` method allows to extract a field from an object, while `display` defines how an object should be visualized on the screen; this can be customized in `smt`, as it will be shown in Section 2.3. Some other functions define the effect of subindexing on the new class. The function `subsref`, is a function which allows to access a field (`C.c`) or an element (`C(2,3)`) of a circulant matrix, and to use typical Matlab subindexing expressions like `C(:)` or `C(3:6,:)`. Notice that `C(1:3,4:7)` returns a Toeplitz matrix (i.e., a `smtoep` object; see Section 2.2), while `C([1,3,5],6:8)`

Elementary math functions			
`abs`	absolute value	`fix`	round towards zero
`angle`	phase angle	`floor`	round towards $-\infty$
`conj`	complex conjugate	`ceil`	round towards $+\infty$
`imag`	imaginary part	`round`	round argument
`real`	real part	`sign`	signum function
Basic array information			
`size`	size of array	`get`	get object fields
`length`	length of array	`isempty`	true for empty array
`display`	display array	`isequal`	true for equal arrays
Array operations and manipulation			
`diag`	diagonals of a matrix	`prod`	product of elements
`diff`	difference/approx. derivative	`reshape`	change size
`full`	convert to full matrix	`sum`	sum of elements
`max`	largest component	`tril`	lower triangular part
`min`	smallest component	`triu`	upper triangular part
Array utility functions			
`double`	convert to double	`subsasgn`	subscripted assignment
`single`	convert to single	`subsindex`	subscript index
`isa`	true if object is in a class	`subsref`	subscripted reference
`isfloat`	true for floating point	`end`	last index
`isreal`	true for real array		
Matrices and numerical linear algebra			
`cond`	condition number	`inv`	matrix inverse
`det`	determinant	`norm`	matrix norm
`eig`	eigenvalues and eigenvectors		

Table 2: Overloaded functions

returns a full matrix. The function `subsasgn` is called whenever the user modifies the content of the `.c` field (like in `C.c(1:3)=[4;5;6]`), while `subsindex` has not been implemented, since we consider it useless for circulant matrices.

We introduced an additional routine, `smtvalid`, which is called by other functions of the toolbox for determining if an object is a valid operand in an expression.

2.2 The `smtoep` class

A Toeplitz matrix can be created by specifying its first column and row, for example with the command

```
T=smtoep([4:7],[4:-1:1]),
```

or giving only the first column, in which case the resulting matrix is Hermitian. The matrix is displayed on the screen either as

```
T =

     4     3     2     1
     5     4     3     2
     6     5     4     3
     7     6     5     4
```

or

```
T =
smtoep object with fields:
       t: [7x1 double]
    dim1: 4
```

```
dim2: 4
 cev: []
```

depending on the `display` configuration parameter; see `smtconfig`.

The operators of Table 1 and the functions of Table 2 have been implemented also for `smtoep` matrices, with four exceptions: the functions `inv`, `det`, `eig`, and `cond` are missing, since there is no standard method to invert a Toeplitz matrix, or to compute its determinant or its eigenvalues. Moreover, the inverse of Toeplitz matrix is not Toeplitz itself, so it could not be stored in a variable of the same class.

2.3 Other functions

Besides the overloaded operators and methods located in the `@smcirc` and `@smtoep` directories, some general functions, listed in Table 3, are placed in the main toolbox directory, and are directly accessible to the user.

Among these functions, we find the `issmcirc` and `issmtoep` functions, which return logical values and check if the supplied parameter belongs to the corresponding class, and the `smtcheck` function, which verifies if the toolbox is correctly installed.

Preconditioners		**General functions**	
`smtcprec`	circulant preconditioners	`issmcirc`	true for `smcirc` object
`strang`	Strang preconditioner	`issmtoep`	true for `smtoep` object
`optimal`	optimal preconditioner	`smtcheck`	check toolbox installation
`superopt`	superoptimal precond.	`smtconfig`	toolbox configuration
		`smtgallery`	test matrices

Table 3: Computational and general functions

The `smtgallery` function gives access to a wide collection of structured test matrices, which are listed in Table 4; see the paper for more information.

Circulant matrices	
`crrand`	uniformly distributed random matrix
`crrandn`	normally distributed random matrix
Toeplitz matrices	
`algdec`	matrix with algebraic decay
`expdec`	matrix with exponential decay
`gaussian`	Gaussian matrix
`tchow`	Chow matrix
`tdramadah`	matrix of 0/1 with large determinant or inverse
`tgrcar`	Grcar matrix
`tkms`	Kac-Murdock-Szego matrix
`tparter`	Parter matrix
`tprand`	uniformly distributed random matrix
`tprandn`	normally distributed random matrix
`tprkdef`	rank deficient linear prediction matrix
`tprolate`	prolate matrix
`ttoeppd`	symmetric positive definite Toeplitz matrix
`ttoeppen`	pentadiagonal Toeplitz matrix
`ttridiag`	tridiagonal Toeplitz matrix
`ttriw`	upper triangular matrix discussed by Wilkinson

Table 4: Test matrices available in the `smtgallery` function

3 Implementation issues

As noted in the previous sections, a great effort has been devoted to catch all possible user's errors, and to reproduce the standard behaviour of Matlab, for example concerning the output of each function in the presence of scalars or empty arrays in input. Since these features are scarcely documented in Matlab manuals, our choices are mostly due to experimental tests. The overhead caused by using the toolbox functions in Matlab computations, with respect to directly inserting inline code in a program, is investigated in the numerical experiments; see Figure 2 in Section 3 of the paper, and the `test2.m` script in the `smt/demo/tests` directory.

Some of the toolbox functions use the `isfloat` command, which was introduced in version 7 of Matlab. For those who are using version 6.5, a patch for this function is included in the software; see the `README.txt` file in the main toolbox directory.

4 Tutorial

In this section, we report the output of the `tutorial.m` script, which illustrates the basic use of the toolbox, and which is available in the `smt/demo` directory.

```
tutorial

% CREATING CIRCULANT MATRICES
C = smcirc([5 4 3 2 1])

C =

     5     1     2     3     4
     4     5     1     2     3
     3     4     5     1     2
     2     3     4     5     1
     1     2     3     4     5

pause % press a key

D = smcirc(6:10)

D =

     6    10     9     8     7
     7     6    10     9     8
     8     7     6    10     9
     9     8     7     6    10
    10     9     8     7     6

whos C D
  Name      Size            Bytes  Class     Attributes

  C         5x5               576  smcirc
  D         5x5               576  smcirc

pause % press a key

% STRUCTURE OF SMCIRC OBJECTS
smtconfig display compact
C

C =

smcirc object with fields:
      c: [5x1 double]
    dim: 5
     ev: []

smtconfig display full

pause % press a key
```

```
% FIRST COLUMN OF THE CIRCULANT MATRIX
C.c

ans =

     5
     4
     3
     2
     1

pause % press a key

% CREATING TOEPLITZ MATRICES
A = smtoep(5:-1:1,5:9)

A =

     5     6     7     8     9
     4     5     6     7     8
     3     4     5     6     7
     2     3     4     5     6
     1     2     3     4     5

pause % press a key

B = smtoep([5 4 3 2 1])

B =

     5     4     3     2     1
     4     5     4     3     2
     3     4     5     4     3
     2     3     4     5     4
     1     2     3     4     5

whos A B
  Name      Size            Bytes  Class     Attributes

  A         5x5               792  smtoep
  B         5x5               792  smtoep

pause % press a key

% STRUCTURE OF SMTOEP OBJECTS
smtconfig display compact
B

B =

smtoep object with fields:
       t: [9x1 double]
    dim1: 5
    dim2: 5
     cev: []

smtconfig display full

pause % press a key
```

```
% THE TOEPLITZ MATRIX DATA
B.t

ans =

     1
     2
     3
     4
     5
     4
     3
     2
     1

pause % press a key

% OPERATORS AND TYPE OF THE RESULTS
E = A + B

E =

    10    10    10    10    10
     8    10    10    10    10
     6     8    10    10    10
     4     6     8    10    10
     2     4     6     8    10

whos A B E
  Name      Size            Bytes  Class     Attributes

  A         5x5               792  smtoep
  B         5x5               792  smtoep
  E         5x5               792  smtoep

pause % press a key

F = A .* B

F =

    25    24    21    16     9
    16    25    24    21    16
     9    16    25    24    21
     4     9    16    25    24
     1     4     9    16    25

whos A B F
  Name      Size            Bytes  Class     Attributes

  A         5x5               792  smtoep
  B         5x5               792  smtoep
  F         5x5               792  smtoep

pause % press a key
```

```
G = A * B

G =

    95   120   133   132   115
    80   102   114   114   100
    65    84    95    96    85
    50    66    76    78    70
    35    48    57    60    55

whos A B G
  Name      Size            Bytes  Class     Attributes

  A         5x5               792  smtoep
  B         5x5               792  smtoep
  G         5x5               200  double

pause % press a key

H = A + C

H =

    10     7     9    11    13
     8    10     7     9    11
     6     8    10     7     9
     4     6     8    10     7
     2     4     6     8    10

whos A C H
  Name      Size            Bytes  Class     Attributes

  A         5x5               792  smtoep
  C         5x5               576  smcirc
  H         5x5               792  smtoep

pause % press a key

% NOT ALL OPERATIONS ARE DEFINED
inv(C)

ans =

   2.1333e-01   1.3333e-02   1.3333e-02   1.3333e-02  -1.8667e-01
  -1.8667e-01   2.1333e-01   1.3333e-02   1.3333e-02   1.3333e-02
   1.3333e-02  -1.8667e-01   2.1333e-01   1.3333e-02   1.3333e-02
   1.3333e-02   1.3333e-02  -1.8667e-01   2.1333e-01   1.3333e-02
   1.3333e-02   1.3333e-02   1.3333e-02  -1.8667e-01   2.1333e-01

% INV(A) PRODUCES THE ERROR:
%
% ??? Error using ==> smtoep.inv at 14
% Not yet implemented, use inv(full(T)) instead.
pause % press a key
```

```
% ONLINE HELP IS AVAILABLE FOR ALL FUNCTIONS
help smtoep
 SMTOEP Toeplitz matrix class constructor.

    T = SMTOEP creates a default empty object.

    T = SMTOEP(col) creates a smtoep object from the vector col
        containing the first column of a square symmetric (or
        Hermitian) Toeplitz matrix.

    T = SMTOEP(col,row) creates a smtoep object from the vector
        col containing the first column and the vector row
        containing the first row of a Toeplitz matrix.

    T = SMTOEP(t,m,n) creates a smtoep object from the vector t
        of dimension (m+n-1) containing a (m-by-n) Toeplitz
        matrix in TPS (Toeplitz Packed Storage or "gnomon")
        format.

    T = SMTOEP(C) creates a smtoep object from the circulant
        matrix contained in the smcirc object C.

    See also smtoep/toeprem, SMCIRC/SMTOEP.

pause % press a key

help smtoep/mtimes
 *  Matrix multiply.

    T*S is the matrix product of T and S. Any scalar (a 1-by-1
    matrix) may multiply anything. Otherwise, the number of
    columns of T must equal the number of rows of S.

    A = MTIMES(T,S) is called for the syntax 'T * S' when T or
    S is an smtoep matrix. The result is a smtoep matrix only
    when one of the argument is of class smtoep and the other
    is a scalar or a smtoep matrix of dimension 1, otherwise
    it is a full array.

    When the same matrix T must be successively multiplied by
    many different arrays, the execution time can be reduced
    by 1/3 by calling T = TOEPREM(T), which precomputes the
    eigenvalues of the circulant matrix in which T is embedded.

    See also smtoep/times, toeprem.

pause % press a key

% FOR A LIST OF AVAILABLE FUNCTIONS, TRY:
% help smt
% help smcirc/Contents
% help smtoep/Contents
%
pause % press a key
```

```
% MEMORY STORAGE
v = rand(n,1);
R = toeplitz(v);   % THIS IS A FULL MATRIX
S = smtoep(v);     % THIS IS A SMTOEP MATRIX
whos R S
  Name         Size                 Bytes  Class     Attributes

  R         2000x2000            32000000  double
  S         2000x2000               32712  smtoep

pause % press a key

% TEST ON THE SPEED OF COMPUTATION:
% PERFORM 200 MATRIX-VECTOR PRODUCTS
x = rand(n,1);

tic, for i=1:200, y=R*x; end, toc % R IS A FULL MATRIX (WAIT A BIT ...)
Elapsed time is 0.498005 seconds.

tic, for i=1:200, y=S*x; end, toc % S IS A SMTOEP MATRIX
Elapsed time is 0.342400 seconds.

pause % press a key

% TEST FOR A VERY LARGE DIMENSION:
% MATRIX-VECTOR PRODUCT
w = rand(100000,1);
S = smtoep(w);
whos S
  Name           Size                  Bytes  Class     Attributes

  S         100000x100000            1600712  smtoep


x = rand(100000,1);
tic, y=S*x; toc
Elapsed time is 0.138585 seconds.

pause % press a key
```

```
% SUBINDEXING IS ALLOWED
% THE OUTPUT IS STRUCTURED WHEN POSSIBLE
C = smcirc(rand(5,1))

C =

   4.1922e-01   8.2641e-01   6.1509e-01   7.6043e-02   3.2441e-01
   3.2441e-01   4.1922e-01   8.2641e-01   6.1509e-01   7.6043e-02
   7.6043e-02   3.2441e-01   4.1922e-01   8.2641e-01   6.1509e-01
   6.1509e-01   7.6043e-02   3.2441e-01   4.1922e-01   8.2641e-01
   8.2641e-01   6.1509e-01   7.6043e-02   3.2441e-01   4.1922e-01

pause % press a key

T = C(2:5,1:4)
Warning: Result of subindexing is smtoep
> In smcirc.subsref at 64
  In tutorial at 144

T =

   3.2441e-01   4.1922e-01   8.2641e-01   6.1509e-01
   7.6043e-02   3.2441e-01   4.1922e-01   8.2641e-01
   6.1509e-01   7.6043e-02   3.2441e-01   4.1922e-01
   8.2641e-01   6.1509e-01   7.6043e-02   3.2441e-01


whos C T
  Name      Size            Bytes  Class     Attributes

  C         5x5               576  smcirc
  T         4x4               776  smtoep

pause % press a key

M = C([1 3 5],1:5);
Warning: Result of subindexing is unstructured
> In smcirc.subsref at 69
  In tutorial at 149

whos C M
  Name      Size            Bytes  Class     Attributes

  C         5x5               576  smcirc
  M         3x5               120  double

pause % press a key
```

```
% THE SMTGALLERY SUITE
%
% help smtgallery
%
pause % press a key

help smtgallery
 SMTGALLERY SMT test matrices.
    [out1,out2,...] = SMTGALLERY(matname, param1, param2, ...)
    takes matname, a string that is the name of a matrix family, and
    the family’s input parameters. See the listing below for available
    matrix families. Most of the functions take an input argument
    that specifies the order of the matrix, and unless otherwise
    stated, return a single output.
    For additional information, type "help private/matname", where matname
    is the name of the matrix family.

    crrand    uniformly distributed random circulant matrix
    crrandn   normally distributed random circulant matrix

    algdec    Toeplitz matrix with algebraic decay
    expdec    Toeplitz matrix with exponential decay
    gaussian  Gaussian Toeplitz matrix
    tchow     Chow matrix -- a singular Toeplitz lower Hessenberg matrix
    tdramadah matrix of ones and zeroes whose inverse has large integer entries
    tgrcar    Grcar matrix -- a Toeplitz matrix with sensitive eigenvalues
    tkms      Kac-Murdock-Szego Toeplitz matrix
    tparter   Parter matrix -- a Toeplitz matrix with singular values near PI
    tprkdef   rank deficient linear prediction Toeplitz matrix
    tprand    uniformly distributed random Toeplitz matrix
    tprandn   normally distributed random Toeplitz matrix
    tprolate  Prolate Toeplitz matrix -- symmetric, ill-conditioned matrix
    ttoeppd   symmetric positive definite Toeplitz matrix
    ttoeppen  pentadiagonal Toeplitz matrix (sparse)
    ttridiag  tridiagonal Toeplitz matrix (sparse)
    ttriw     upper triangular Toeplitz matrix discussed by Wilkinson and others

    This function is written following the gallery function of Matlab.

    See also gallery.

pause % press a key

help private/gaussian
 GAUSSIAN Gaussian Toeplitz matrix.

    T = SMTGALLERY(’GAUSSIAN’, N, SIGMA) is the N-by-N Toeplitz
    matrix such that
       T(i,j) = sqrt(SIGMA/(2*pi)) * exp(-SIGMA/2*(i-j)^2)
    for real positive SIGMA.  SIGMA defaults to 1.
    The matrix is symmetric positive definite.

    [T,ACOND] = SMTGALLERY(’GAUSSIAN’, N, SIGMA) returns also
    the asymptotic condition number.

pause % press a key
```

```
T = smtgallery('gaussian',1000);
T(1:5,1:5)

ans =

   3.9894e-01   2.4197e-01   5.3991e-02   4.4318e-03   1.3383e-04
   2.4197e-01   3.9894e-01   2.4197e-01   5.3991e-02   4.4318e-03
   5.3991e-02   2.4197e-01   3.9894e-01   2.4197e-01   5.3991e-02
   4.4318e-03   5.3991e-02   2.4197e-01   3.9894e-01   2.4197e-01
   1.3383e-04   4.4318e-03   5.3991e-02   2.4197e-01   3.9894e-01

pause % press a key

get(T)

T is a smtoep object with fields:
       t: [1999x1 double]
    dim1: 1000
    dim2: 1000
     cev: []

pause % press a key

% THE SMTCPREC SUITE
%
help smtcprec
 SMTCPREC Construction of circulant preconditioners.

    C = SMTCPREC(precname,A,options) returns a smcirc matrix
    containing the circulant preconditioner of matrix A, where
    precname is a string containing the preconditioner type:
       strang    Strang preconditioner (only for smtoep matrices)
       optimal   optimal preconditioner
       superopt  superoptimal preconditioner
    The third parameter, if present, is passed to the function
    which performs the computation.

    For additional information, type "help private/precname".

pause % press a key

help private/optimal
 OPTIMAL optimal circulant preconditioner.

    C = SMTCPREC('OPTIMAL',A) returns a smcirc matrix containing
    the optimal preconditioner of the matrix A.

    The computation is fast only if A is a smtoep matrix.
    If A is a smcirc matrix, C is A itself.

pause % press a key
```

```
A = smtgallery('gaussian',1000);
S = smtcprec('strang',A);         % STRANG PRECONDITIONER
O = smtcprec('optimal',A);        % OPTIMAL PRECONDITIONER
T = smtcprec('superoptimal',A);   % SUPEROPTIMAL PRECONDITIONER
whos A S O T
  Name         Size              Bytes  Class     Attributes

  A         1000x1000            16712  smtoep
  O         1000x1000             8536  smcirc
  S         1000x1000             8536  smcirc
  T         1000x1000            24536  smcirc

pause % press a key

% CONJUGATE GRADIENT TEST
S = smtgallery('gaussian',n);   % SMTOEP MATRIX
R = full(S);                    % FULL MATRIX
b = S*ones(n,1);

% CONJUGATE GRADIENT - FULL MATRIX
tic, [x flag res iter] = pcg(R,b,1e-8,100); toc, iter
Elapsed time is 0.157521 seconds.

iter =

    42

% CONJUGATE GRADIENT - SMTOEP MATRIX
tic, [x flag res iter] = pcg(S,b,1e-8,100); toc, iter
Elapsed time is 0.089216 seconds.

iter =

    42

pause % press a key
```

```
% CONJUGATE GRADIENT WITH OPTIMAL PRECONDITIONING
C = smtcprec('optimal',S);
whos C
  Name         Size              Bytes  Class     Attributes

  C         2000x2000            16536  smcirc


% PCG WITH FULL MATRIX
tic, [x flag res iter] = pcg(R,b,1e-8,100,C); toc, iter
Elapsed time is 0.053440 seconds.

iter =

     4

% PCG WITH SMTOEP MATRIX
tic, [x flag res iter] = pcg(S,b,1e-8,100,C); toc, iter
Elapsed time is 0.033338 seconds.

iter =

     4

pause % press a key

% A LARGER TEST
S = smtgallery('gaussian',50000);
b = S*ones(50000,1);
C = smtcprec('optimal',S);
whos S C
  Name          Size                Bytes  Class     Attributes

  C         50000x50000            400536  smcirc
  S         50000x50000            800712  smtoep

tic, [x flag res iter] = pcg(S,b,1e-8,100,C); toc, iter
Elapsed time is 0.267763 seconds.

iter =

     4

pause % press a key
```

```
% GMRES TEST
T = smtgallery('tparter',5000);
b = T*ones(5000,1);

% NO PRECONDITIONING
tic, [x flag res iter] = gmres(T,b,[],1e-8,100); toc, iter
Elapsed time is 0.596139 seconds.

iter =

     1    60

% STRANG PRECONDITIONING
C = smtcprec('strang',T);
tic, [x flag res iter] = gmres(T,b,[],1e-8,100,C); toc, iter
Elapsed time is 0.129959 seconds.

iter =

     1     9

pause % press a key
```