Dimensional Analysis Toolbox

For Use with MATLAB®

Steffen Brückner

Computation

Visualization

Programming

User's Guide

Version 1

How to contact the author

www.sbrs.net	Web
datool@sbrs.net	Email
comp.soft-sys.matlab	Newsgroup

+49 (711) 685 2482	Phone (daytime Europe)
+49 (711) 677 2153	Phone
+49 (711) 677 2154	Fax

Steffen Brückner
Hessenlauweg 19
D-70569 Stuttgart

Mail

Printing History: February 2002 First printing
February 2002 changed to V1.01

Table of contents

Dimensional Analysis Toolbox for Matlab

Version 1.01, 18-Feb-2002

Preface

Dimensional analysis is a powerful tool in engineering reducing the number of parameters of a given problem as well as to clarify the scaling behaviour of a system. For a given list of relevant parameters dimensional analysis gives the corresponding dimensionless groups (transformation groups, □s). This toolbox aims at engineers using Matlab who have to do a lot of dimensional analysis and the according data transformations.

The homepage of dimensional analysis at the Universität Stuttgart

```
http://www.isd.uni-stuttgart.de/pigroup/
```

Prerequisites

The toolbox was developed and tested with the following equipment

- Matlab 6.1 (R12.1)
- PC running MS Windows 2000 (P II 450 MHz, 512 MB)

In the freely available version I cannot guarantee the toolbox to work on any configuration, but I will do whatever possible to make it running on different systems. Please let me know when you encounter any problems with this toolbox.

Although the Dimensional Analysis Toolbox requires a Windows operating system for the `matedit` matrix editor the toolbox can also be used with other operating systems. Only the matrix editor does not work on these systems.

The graphical UI `datool` requires a Matlab 6 (R12).

The Dimensional Analysis Toolbox makes use of the symbolic math toolbox if available. Otherwise basic formatting algorithms are used.

Licensing terms

I. Conditions of use for the Dimensional Analysis Toolbox for Matlab

It order to use the Dimensional Analysis Toolbox for Matlab, or any of its constituent parts, a user must agree to abide by a set of conditions of use.

1. The package is freely available for "Academic" use. "Academic" use of the package is defined to be use of the package for academic, educational, research or other non-commercial and not-for-profit purposes under the following conditions. Any use of the package implies that these conditions have been understood, and that the user agrees to abide by all the listed conditions.

2. The toolbox comes with no guarantee, expressed or implied, neither that it is suit-able for any specific purpose nor that it is free of error. It should not be relied on as the basis to solve a problem whose incorrect solution could result in injury to person or property. If the toolbox is employed in such a manner, it is at the users' own risk and the author disclaims all liability for such misuse.

2. The package is distributed "as is". In particular, no maintenance, support, troubleshooting or subsequent upgrade is implied.

3. The use of the Dimensional Analysis Toolbox for Matlab must be acknowledged, in any publication which contains results obtained with the package or any of its parts. Use a citation in the like the following:
   Brückner, Steffen, Dimensional Analysis Toolbox for Matlab, Version 1.01, `http://www.sbrs.net/`, 2002

4. The toolbox, or any modification thereof, may not be sold or supplied, nor embedded within other software products which are subsequently sold, or supplied, to a third party without written consent from the author.

5. The toolbox may not be distributed in any form other than pointing to the toolbox website at http://www.sbrs.net/ without written consent from the author. The toolbox is also available at Mathwork's MatlabCentral. Any other form of distribution is considered "commercial use" and the respective licensing terms apply.

6. Users of the toolbox are encouraged to send examples of the problems they have solved using the package to the author. These examples may subsequently be passed into the public domain to allow others to test and compare new and existing software.

7. The toolbox source code may be changed to suit your own needs, but the changed version may not be distributed without written consent from the author. Users are encouraged to send their modifications and extensions to the author for evaluation of possible integration future versions.

8. It is the responsibility of the licensee to ensure that each user of the package is aware of, and agrees to abide by, all the conditions given above.

9. The toolbox contains LiteGrid which is a freeware ActiveX control for matrix manipulation in forms. This control was written by Andrew Ivannikov and is freely available.

II. Commercial use of the Dimensional Analysis Toolbox for Matlab

The use of the Dimensional Analysis Toolbox V1.01 in commercial environments is free. Only basic email support is granted. For further support please contact the author.

Conditions for redistribution of the Dimensional Analysis Toolbox for Matlab or any of its constituent parts have to be negotiated separately with

Steffen Brückner
Hessenlauweg 19
D-70569 Stuttgart
email: datool@sbrs.net

to whom all commercial enquiries should be addressed

Overview

The following printout is the help about the toolbox.

```
Dimensional Analysis
MATLAB Toolbox   Ver. 1.01   18-Feb-2002


Dimensional Analysis
  checkdm     - check dimension matrix for validity
  createab    - create A and B submatrices
  created     - create a D submatrix
  datool      - GUI for dimensional analysis
  rlist       - manage relevance list
  diman       - perform dimensional analysis
  dtrans      - transform data from x to pi
  numpi       - number of base variables

Output
  pretty      - pretty print of dimensionless groups
  latex       - LaTeX output of dimensionless groups
  texfile     - write TeX output of a pi set to file

Helper
  getdv       - get dependent variables
  matedit     - GUI matrix editor
  unit2si     - converts units to dimensional representation
  data2si     - converts data to SI basic units
  xsort       - resort data arrays

Demos
  beamdemo    - cantilever beam with tip load
  blastdemo   - energy in a nuclear explosion
  dguidemo    - starts GUI with demo input
  oscdemo     - simple oscillator (use of D submatrix)
  spheredemo  - sphere in a flow
  transfdemo  - sphere in a flow (data transformation)
```

Installation

Simple User Installation

Extract the file `DIMENSION.ZIP` to any directory. A subdirectory dimension is created in this directory. Add this subdirectory to your Matlab search path, e.g. using the Matlab Set Path utility.
Using this installation, the Dimensional Analysis Toolbox can be used on any system, regardless which operating systems and whether you have administrative rights or not. With this kind of installation, you cannot use the graphical matrix editor `matedit` which requires the ActiveX plugin LiteGrid to be installed.

Windows operating systems (32-bit):

Log in as administrator. Extract the contents of the `DIMENSION.ZIP` file to your `$MATLAB\toolbox` directory. Please make sure that "use path information" is turned on in your unzip utility. A subdirectory `dimension` is automatically created.
Then run the script file `dinst.m` in this directory to add the Dimensional Analysis Toolbox to your Matlab path and install the LiteGrid ActiveX plugin.
If `dinst` fails, use `addpath` in Matlab to add the toolbox path to your Matlab search path and copy the file `LiteGrid\lgrid.ocx` to your `$WINDIR\system32` directory and run the command `regsvr32 lgrid` in that directory (using a shell or the Matlab dos or system command).

Other operating systems:

To install the Dimensional Analysis Toolbox to your `local $MATLAB\toolbox` directory you need to have administrative rights (i.e. log on as root on Unix systems). Extract the archive `DIMENSION.ZIP` to your local `$MATLAB\toolbox` directory. Run the `dinst.m` script in that directory; see below to add the Dimensional Analysis Toolbox to your Matlab search path.

Search Path

Since Matlab does not provide a way to save the changed search path permanently from a script file, you must either use the `Set Path` utility found in the Matlab menu `File -> Set Path` and save the current search path, or edit the file `$MATLABROOT\toolbox\local\pathdef.m` manually to make the search path to the dimensional analysis toolbox permanent.

Note:

Toolboxes in `$MATLABROOT\toolbox` might be overwritten when installing a new version of Matlab. Please make sure to reinstall the Dimensional Analysis Toolbox for Matlab after upgrading your Matlab software.
If you choose not to install LiteGrid (either no administrative rights or non-Windows operating system), you can still use the graphical UI `datool` without technical restrictions.

Dimensional Analysis

Buckingham's Pi-Theorem states that for each dimensional homogeneous and complete relationship f of n physical variables x_i

$$f(x_1,\ldots,x_n) = 0 \tag{1}$$

there exists a corresponding relationship

$$F(\pi_1,\ldots,\pi_m) = 0 \tag{2}$$

of only $m \leq n$ dimensionless groups

$$\pi_j = x_{j+r}\prod_{i=1}^{r} x_j^{-\alpha_{ji}} \;\; ; \alpha_{ji} \in \mathbb{R} \tag{3}$$

with $m = n - r$, where r denotes the rank of the dimensional matrix.

The dimensional matrix can be established from the knowledge of the problems' relevance list, which is the list of all relevant parameters and their respective dimensions. The rows of the dimensional matrix correspond to the parameters while the columns correspond to the value of the dimension exponent of the variable. The elements of the dimensional matrix are therefore the exponents of the dimensions of all the relevant parameters.

	D_1 $\cdots$ D_k
x_1	
$\vdots$	D
x_n	

$\Rightarrow$

	$\overline{D}_1$ $\cdots$ $\overline{D}_r$
x_1	I
$\vdots$	
x_n	α_{ji}

(4)

The dimensional matrix ***D***, shown in eq. (4) on the left, is formed by the relevant parameters x_i as the rows and the corresponding dimensional exponents D_j as the columns of the matrix. Applying rank preserving column operations on the dimensional matrix, the original dimensional matrix is transformed into the matrix shown in eq. (4) on the right side. This matrix consists of an upper square identity matrix ***I*** of size *r x r* and a lower sub-matrix ***A*** of size *(n-r) x r*. The lower sub-matrix ***A*** contains the exponents a_{ji} used in the forming of the dimensionless groups in equation (3).

The rank r of the dimensional matrix ***D*** represents the number of independent base dimensions involved in a given problem. Mechanical problems e.g. can often be written in the two base dimensions force and distance instead of the three SI-unit system base dimensions length, mass, and time. In either case however, the rank of the dimensional matrix will be *r=2* after execution of the rank preserving dimensional matrix operations.

Demos

The Dimensional Analysis Toolbox for Matlab includes some demos for dimensional analysis for a sphere submerged in a flow. The list of relevant parameters is given by

Drag force	D	[N]
flow velocity	v	[m/s]
fluid viscosity	ν	[m^2/s]
fluid density	ρ	[kg/m^3]
sphere diameter	d	[m]

The Dimensional Analysis Toolbox now allows creating the relevance list in Matlab

```
% Define the variable names and the
% respective dimensions
N = {'D', 'v',   'nu',   'rho',   'd'};
u = {'N', 'm/2', 'm2/s', 'kg/m3', 'm'};

% create the relevance list
[d,f] = unit2si(u);
RL = rlist(N,d,f);
```

now the dimensional analysis can be done

```
% choose the base variables
bv = {'v','d','rho');

% do the dimensional analysis
piset = diman(RL,bv);
```

and now we take a look at the dimensionless groups

```
% pretty print the pis
pretty(piset);
```

With `piset` we can transform the data

```
% load demo data
load demodata/spheredata
XData = Kugel';

% tramsform the x-data to SI basic units
XData = data2si(XData,RL);

% and transform the data
PiData = dtrans(XData,piset);
```

Some demos are included with the toolbox. Simply run the script files by typing their name. Using the Matlab `type` command you can take a look at the source code.

`spheredemo`	-	dimensional analysis for the sphere and pretty print the resulting dimensionless groups
`beamdemo`	-	dimensional analysis for a cantilever beam with tip load
`transfdemo2`	-	dimensional analysis for the sphere and transform data (given by Prandtl) (demonstrates data transform)
`oscdemo`	-	differential equation of motion for a simple oscillator (demonstrates use of D submatrix)
`blastdemo`	-	energy in a nuclear explosion
`dguidemo`	-	sphere relevance list in GUI `datool`

Function Reference

CHECKDM

Purpose Check dimensional analysis for validity

Syntax

```
b = CHECKDM(A,B)
b = CHECKDM(D)
```

Description `checkdm` checks the dimensional matrix `[B A]` or `D` according to the calling syntax for validity

`b = checkdm(A,B)` takes the following inputs

`A`	-	submatrix `A` of dimensional matrix
`B`	-	submatrix `B` of dimensional matrix
`D`	-	dimensional matrix `D = [B A]`

and returns,

`b`	-	1 if valid, 0 if invalid

Examples Create a relevance list, build A and B sub matrices and check for validity

```
>> Name = {'q' ,'d','u'  ,'nu'  ,'D'};
>> Unit = {'Pa','m','m/s','m2/s','N'};
>> % create a new relevance list
>> [d,f] = unit2si(Unit);
>> RL = rlist(Name,d,f);
>> % specify base variables
>> bv = {'q','d','u'};
>> % check dimensional matrix for validity
>> checkdm(createab(RL,bv))
>> % and now do the analysis
>> piset = diman(RL,bv);
```

Algorithm `checkd` checks if the rank of the dimensional matrix corresponds to the number of rows and is equal to the rank of the A sub matrix (if A,B are given)

CREATEAB

Purpose Create A and B sub matrices

Syntax `[A,B,order] = CREATEAB(RL,BaseVars)`

Description `createab` forms a dimensional matrix and creates the sub matrices A and B needed for dimensional analysis

`b = chreateab(RL,BaseVars)` takes the following inputs

`RL`	-	a valid relevance list
`BaseVars`		Cell array of strings containing the names of the base variables

and returns,

`A`	-	sub matrix `A` of dimensional matrix
`B`	-	sub matrix `B` of dimensional matrix
`order`	-	index vector for the new order of the given variables from RL in the [B A] matrix

Examples Create a relevance list, build A and B sub matrices and check for validity

```
>> Name = {‘q’ ,’d’,’u’  ,’nu’  ,’D’};
>> Unit = {‘Pa’,’m’,’m/s’,’m2/s’,’N’};
>> % create a new relevance list
>> [d,f] = unit2si(Unit);
>> RL = rlist(Name,d,f);
>> % specify base variables
>> bv = {‘q’,’d’,’u’};
>> % check dimensional matrix for validity
>> checkdm(createab(RL,bv))
>> % and now do the analysis
>> piset = diman(RL,bv);
```

Algorithm `createab` builds the dimensional matrix from the dimensional information in the relevance list RL and deletes zero rows and linear dependent rows from the set.

CREATED

Purpose Create D submatrix

Syntax `D = CREATED(RL,DV,D1,D1NAMES)`

Description `created` forms the D sub matrix needed for dimensional analysis

`D = created(RL,DV,D1,D1NAMES)` takes the following inputs

`RL`	-	a valid relevance list
`DV`	-	Cell array of strings containing the names of the dependent variables
`D1`	-	planned D sub matrix
D1NAMES		variable names in the order corresponding to the columns in D1

and returns,

`D`	-	sub matrix `D` of dimensional set

Examples See `oscdemo`

See also `getdv`

Algorithm `created` resorts the rows of `D1` to reflect the order of variables in the relevance list.

DATA2SI

Purpose Transform data to SI basic units

Syntax `Y = DATA2SI(X,RL)`

Description `data2si` transforms the x-domain data X to x-domain data Y using SI basic units

`Y = data2si(X,RL)` takes the following inputs

`X`	-	x-domain data, rows corresponding to variables
`RL`	-	a valid relevance list

and returns,

`Y`	-	x-domain data in basic SI units

`data2si` is useful when using non-basic SI units, such as e.g. [mm] or [cal]. The data is then transformed to the respective basic SI units.

Examples See `transfdemo`

DATOOL

Purpose

Graphical user interface for dimensional analysis

Syntax

```
piset = DATOOL(Name,Unit)
piset = DATOOL
```

Description

`datool` is a graphical user interface which aids the user in performing dimensional analysis for a given problem

`piset = datool(Name,Unit)` takes the following inputs

`Name`	-	Cell array of strings containing the variable names
`Unit`	-	Cell array of strings containing the units of the variables

and returns,

`piset`	-	Dimensional Set as found by the dimensional analysis in the GUI

With the second syntax, `datool` is started with a dummy variable. This is for loading already existing sets.

Examples

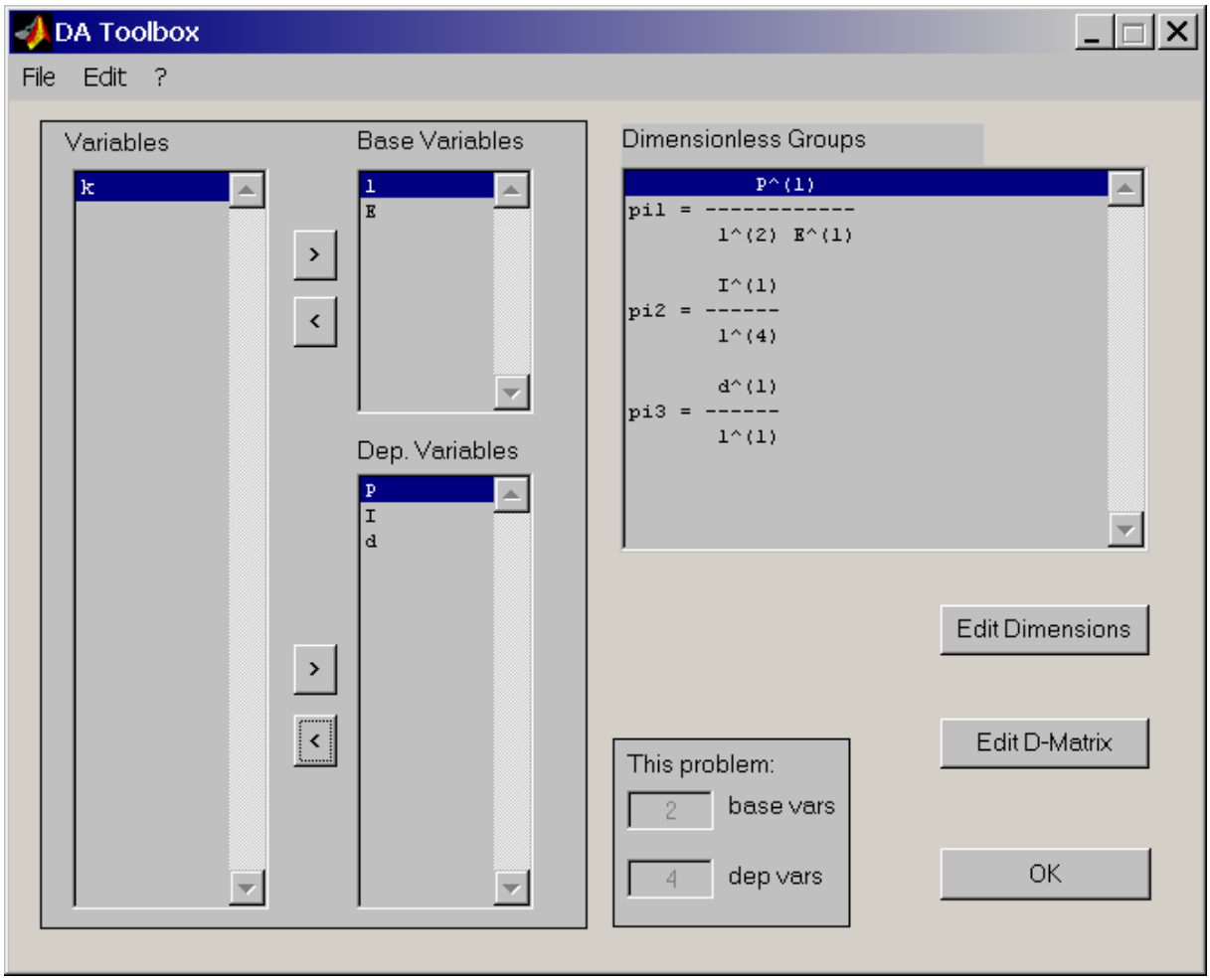
Performing dimensional analysis on a sphere submersed in a flow

```
Name = {'q' ,'d','u'  ,'nu'  ,'D'};
Unit = {'Pa','m','m/s','m2/s','N'};
piset = datool(Name,Unit);
```

Here the names of the variables are defined, the according units are defined and `datool` is called with these parameters.

Usage

The `datool` GUI provides a list of the Variables on the left side and two lists for the base and the dependent variables respectively. With the arrow buttons variables can be moved from the Variables list to the base or dependent variable list and vice versa. After the base and the dependent variables have been selected the dimensional information (dimensional matrix) should be check by clicking `Edit Dimensions`. If the information in the dimensional matrix is correct, the dimensional analysis is started automatically and the resulting dimensionless groups are displayed. The D matrix if not defined otherwise is assumed to be an identity matrix of the suitable size. The D matrix can be edited clicking on `Edit D-Matrix`.



The file menu allows saving and loading of dimensional sets. The sets are stored as Matlab data files (*.mat). Another option is the `LaTeX` output which generates a LaTeX (*.tex) file with the information from the dimensional set.

The edit menu allows to copy the `piset` variable including the sub matrices of the dimensional set to the workspace.

The number of "dep vars" displayed is a maximum, at least one dependent variable must be selected for dimensional analysis.

Restrictions

`datool` uses the `matedit` command of the dimensional analysis toolbox which in turn uses an Active X control LiteGrid. If this is not installed (e.g. when you are not using a Windows 32-bit operating system) the Matlab Array Editor is called (requires Java runtime). In this case, you must use the menu functions `Edit -> Reload Dimensions from WS` and `Edit -> Reload D-Matrix from WS` respectively to reload the data closing the array editor. Be careful only to use these two functions after editing a matrix since no checks are performed on the imported data!

The Matlab Array Editor does not work when `datool` is called with output parameters. This is due to the implantation of the Array Editor as a java class which is only executed when no other window is using `uiwait`....

When using the `matedit` matrix editor (using LiteGrid) you must exit the window by clicking the OK button. If you use the close window function of the window, a dispatch error is generated and your changes are discarded.

The number of base variables displayed corresponds to the full relevance list. Systems with less base variables are still possible choosing a smaller number of dependent variables with appropriate dimensions.

The number of dependent variables displayed corresponds to the maximum number of dependent variables for the full relevance list. Fewer dependent variables are always possible. At least one base variable has to be chosen for a valid dimensional set.

DIMAN

Purpose Perform the dimensional analysis

Syntax

```
piset = diman(RL,BaseVars,D)
```

Description `diman` calculates the C sub matrix from the given A, B (and optional D) sub matrices.

`piset = diman(RL,BaseVars,D)` takes the following inputs

`RL`	-	already existing relevance list with valid A and B sub matrices
`BaseVars`		Cell array of strings containing the names of the base variables
`D`		(optional) D sub matrix

and returns,

`piset`	-	A piset structure containing all sub matrices

If the optional parameter `D` is not given the D sub matrix is created as an identity matrix of suitable size.

Examples Creating a relevance list for the sphere problem and performing dimensional analysis

```
>> Name = {'q' ,'d','u'  ,'nu'  ,'D'};
>> Unit = {'Pa','m','m/s','m2/s','N'};
>> % create a new relevance list
>> [d,f] = unit2si(Unit);
>> RL = rlist(Name,d,f);
>> % and now do the analysis
>> piset = diman(RL,{'q','d','u'});
```

Algorithm `diman` uses an algorithm proposed by Szirtes. The dimensional matrix is divided into two sub matrices B for the dependent variables and A for the independent variables. A quadratic and regular matrix D of size *m x m,* where *m* is the number of dependent variables, is used to create different sets of dimensionless groups. The method of calculation for the C sub matrix is

$$C = -D\,(A^{-1}\,B)^{T}$$

DTRANS

Purpose Transform data from x to π domain

Syntax `PiData = dtrans(Xdata,piset)`

Description `dtrans` transforms data given in the x domain to the representation in dimensionless groups (π domain) using the transformation given by the valid `piset`.

`PiData = dtrans(Xdata,piset)` takes the following inputs

`XData`	-	Matrix of data in the x domain. The rows correspond to the variables, the columns to individual records.
`piset`	-	A valid dimensional set (after analysis)

and returns,

`PiData`	-	The dimensionless data. Row correspond to the dimensionless groups, columns to the individual records

Examples Creating a relevance list for the sphere problem

```
>> Name = {'q' ,'d','u'  ,'nu'  ,'D'};
>> Unit = {'Pa','m','m/s','m2/s','N'};
>> XData = [ [0.87 0.008 1.2 1.22e-5 2.04e-6]' , …
          [0.29 0.008 2.2 1.53e-5 7.20e-6]'];
>> % create a new relevance list
>> [d,f] = unit2si(Unit);
>> RL = rlist(Name,d,f);
>> % and now do the analysis
>> piset = diman(RL,{'q','d','u'});
>> % Transform data to SI basic units
>> XData = data2si(XData,RL);
>> % and now transform the data
>> PiData = dtrans(Xdata,piset);
```

GETDV

Purpose get list of dependent variables

Syntax `dv = GETDV(RL,BV)`

Description `getdv` determines the list of dependent variables for a given relevance list and set of base variables

`dv = getdv(RL,BV)` takes the following inputs

`RL`	-	a valid relevance list
`BV`	-	Cell array of strings containing the names of the base variables

and returns,

`DV`	-	Cell array of strings containing the names of the dependent variables

Examples See `oscdemo`

See also `created`

LATEX

Purpose Produce LaTeX formatting of dimensionless groups

Syntax `latex(piset)`

Description `latex` prints the LaTeX commands to visualize the dimensionless groups in piset

`latex(piset)` takes the following inputs

`piset` - A valid dimensional set

If the symbolic math toolbox is installed `latex` makes use of this toolbox, otherwise internal formatting algorithms are used (although the ones in the symbolic math toolbox produce much nicer results…)

Examples

```
>> latex(p)
{\it pi\_1}={\frac {\nu}{du}}
{\it pi\_2}={\frac {D}{q{d}^{2}}}
```

See also `texfile`

MATEDIT

Purpose GUI matrix editor

Syntax `Mout = MATEDIT(Min,RNames,CNames,FigName)`

Description `matedit` is a GUI matrix editor using the ActiveX control LiteGrid and only works on Windows operating systems

`Mout = matedit(Min)` takes the following inputs

`Min` - Matrix to edit

`RNames` - (optional) The titles of the rows

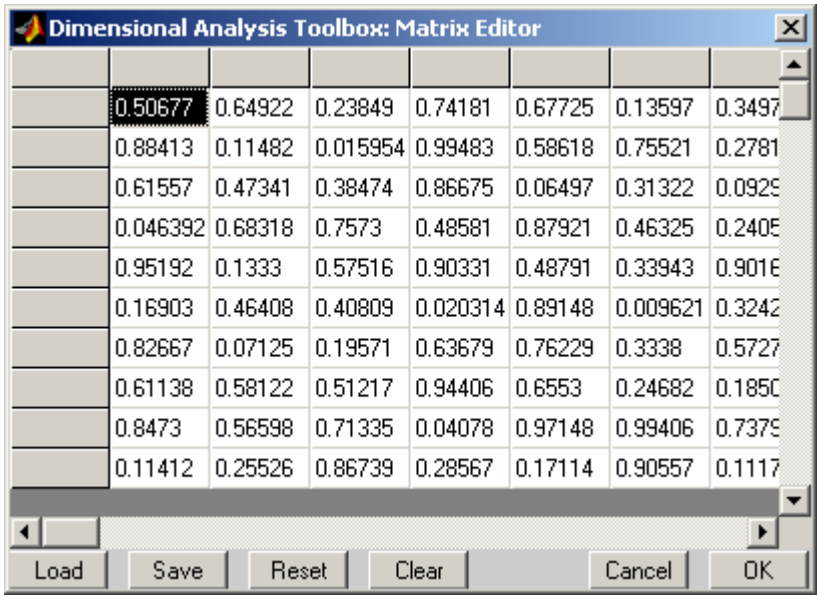
`CNames` - (optional) The titles of the columns

`FigName` - (optional) The names of the edit window

and returns,

`Mout` - The matrix after editing

It is not possible to change the matrix size in the matrix editor



If LiteGrid is not installed on your computer `matedit` returns an empty matrix.

`Load` and `Save` allow to store the matrix in a file.

`Reset` resets the matrix to a identity matrix or to a rectangular matrix with ones on the main diagonal.
`Clear` sets all matrix elements to zero.

`Ok` exits the matrix editor affirming changes.

`Cancel` exits the matrix editor aborting changes.

LiteGrid

LiteGrid is a freely available ActiveX control for matrix editing written by Andrew Ivannikov. The control lgrid.ocx is available on the Internet at

```
http://www.codeguru.com/controls/LiteGrid.shtml
```

for download.

The version of `lgrid.ocx` which is included in the Dimensional Analysis Toolbox package is recompiled with changes suggested by Qi Yang to change the selection highlighting behaviour.

To install the LiteGrid ActiveX control copy the file `lgrid.ocx` to your `$WINDIR\system32` directory (`$WINDIR` stands for your windows directory, e.g. `C:\WINDOWS` or `C:\WINNT`). Open a shell (`Start -> Run … -> "command.com"`) and type the following commands (remember to replace `$WINDIR` with your windows directory path) to register the control with your system

```
C:\> cd $WINDIR\system32
C:\:> regsvr32 lgrid.ocx
```

NUMPI

Purpose Determine the number of dimensionless groups and base variables for a given relevance list

Syntax `[numPi,numBase] = numpi(RL)`

Description `numpi` calculates the number of base variables needed for a given relevance list and the number of maximum possible dimensionless groups

`numpi(RL)` takes the following inputs

`RL` - A valid relevance list

And returns,

`numPi` - The number of possible dimensionless groups

`numBase` - The number of base variables needed to be selected for the given relevance list

Algorithm First the rank r of the dimensional matrix formed by the relevance list `RL` is calculated. The number of base variables for the given relevance list equals the number of independent dimensions, which in turn equals the rank r of the dimensional matrix. All variables which are not base variables are labelled dependent variables.

Given relevance list with n variables, the rank r of the dimensional matrix corresponding to the relevance list is calculated. The number of base variables `numBase` = r and the number of dependent variables `numPi` = $n - r$.

PRETTY

Purpose Pretty print dimensionless groups

Syntax `pretty(piset)`

Description `pretty` prints the dimensionless groups defined in the dimensional set piset in a fashion that resembles type-set formulas

`pretty(piset)` takes the following inputs

`piset` - A valid dimensional set

If the symbolic math toolbox is installed pretty makes use of this toolbox, otherwise internal formatting algorithms are used (although the ones in the symbolic math toolbox produce much nicer results…)

Examples Creating a relevance list for the sphere problem

```
>> pretty(p)
```

$$pi1 = \frac{nu}{d\ u}$$

$$pi2 = \frac{D}{q\ d^2}$$

RLIST

Purpose Create and manage a relevance list

Syntax

```
RL = rlist(Name,Dimension,Factor)
RL = rlist(RL,Name,Dimension,Factor)
```

Description `rlist` creates a relevance list or adds additional variables to a relevance list.

`RL = rlist(RL,Name,Dimension,Factor)` takes the following inputs

`RL`	-	already existing relevance list or []
`Name`	-	Cell array of strings containing the variable names
`Dimension`		Array of dimensional exponents, the rows rows correspond to the variables and the columns correspond to the base dimensions
`Factor`	-	Vector of conversion factors from given units to SI units

and returns,

`RL`	-	The relevance list

If `RL` is not given as a input parameter a new relevance list is created.

The input argument `Factor` is optional, the default is 1.

Examples Creating a relevance list for the sphere problem

```
>> Name = {‘q’ ,’d’,’u’  ,’nu’  ,’D’};
>> Unit = {‘Pa’,’m’,’m/s’,’m2/s’,’N’};
>> % create a new relevance list
>> [d,f] = unit2si(Unit);
>> RL = rlist(Name,d,f);
>> % add a variable
>> RL = rlist(RL,’v0’,’m/s’);
```

Algorithm RLIST creates and maintains a data structure RL for relevance lists

```
RL(ii).Name       % Name of the variable
RL(ii).Dimension  % row vector of dimensional exponents
RL(ii).Factor     % conversion factor to SI units
```

TEXFILE

Purpose Produce a TeX output file with dimensional set

Syntax `texfile(Filename,piset)`

Description `texfile` writes a TeX output file with the information (matrices and dimensionless groups) from a dimensional set

`texfile(Filename,piset)` takes the following inputs

`Filename`		A valid output file name, existing files are overwritten
`piset`	-	A piset

If `piset` is not a valid dimensional set, no dimensionless groups are output and the undefined or ill-defined matrices are printed as `[]`

Examples `texfile('latextest.tex',piset)`

Algorithm `texfile` uses the `latex` command from the dimensional analysis toolbox which in turn uses the symbolic math toolbox if it is installed. Otherwise internal display algorithms are used.

See also `latex`

UNIT2SI

Purpose Transform unit names to SI unit representation vectors

Syntax
```
[Dimension,Factor] = unit2si(unit)
TM = unit2si('help')
```

Description `unit2si` transforms the units in strings into a dimensional representation vector and a conversion factor for the data. Unknown units are transformed into empty (dimensionless) dimensional representations with factor 1.

`[Dimension,Factor] = unit2si(unit)` takes the following inputs

`unit` - Cell array of strings containing the unit names

and returns,

`Dimension` Matrix of the dimensional representations rows correspond to the input unit, columns to the 7 base dimensions of the SI system

`Factor` - Vector of conversion factors from the input data to the pi domain

If `piset` is not a valid dimensional set, no dimensionless groups are output and the undefined or ill-defined matrices are printed as `[]`.

The second form calling `unit2si` with the parameter `'help'` returns the cell array of dimensional information used in this function. The first column of `TM` contains the recognized units.

Examples See e.g. `RLIST`

Recognized Units

dimensionless	`'0','rad','grad','sr'`
mass	`'kg','g','mg','t','oz','lb'`
geometric	`'m','km','dm','cm','mm','AU','pc','LJ','Angst','in','ft', 'yd','mile','m2','km2','m3','dm3','l','floz','pt','qt', 'gal'`
time	`'ms','s','min','h','d','Hz','1/s','1/min'`
thermodynamics	`'K','St','Pas','J/K','J/m3','J/kg','J/mol','J/molK', 'W/mK', 'W/m2K','W/m2','W/sr', 'mol/l','l/mol', 'g/mol','J/molK','J/mol','kg/`
electricity	`'A','mA','W','VA','PS','W/m2','J','kWh','eV','erg', 'cal','kcal', 'm2/s','V','ohm','Ohm','S','C','Ah','C/m3', 'C/m2','F','F/m','V/m','Wb','T','H','H/m','A/m','C/kg'`
light	`'cd', 'cd/m2','sb','lm','lx'`
substance	`'mol','m3/kg'`
mechanics	`'m3/s','tex','kg/m2','kg/m3','kg/s','m/s','km/h','kn', 'm/s2','GU','N','dyn','p','Ns','Pa','N/m2','bar','mbar', 'mmHg','atm','at','Torr', 'lb/in2','psi','kg/s2','kg/s'`
radioactivity	`Bq','Ci','Gy','rd','Sv','mSv','Rem','Gy/s','rd/s', 'Sv/s','rem/s','R'`

At the moment only these dimensions are recognized. The dimensional representation is not parsed for prefixes or combinations of these units.

XSORT

Purpose Sort x data for transformation to π domain

Syntax

```
XDsorted = XSORT(XData,piset,Names)
```

Description `xsort` sorts the x data `D` with rows corresponding to the `Names` for the transformation with `dtrans` using `piset`. The analysis with `diman` changes the order of the variable names so the sort is essential.

`XDsorted = XSORT(XData,piset,Names)` takes the following inputs

`XData`	-	Matrix of the data in the x domain, rows correspond to the variables, columns to records
`piset`	-	A valid dimensional set
`Names`	-	Cell array of strings containing the variable names in the same order as they appear in `XData`

and returns,

`XDsorted`	Matrix of the data in the x domain, rows sorted to match the variable order in `piset`

Examples See `dtrans`

Private functions

The Dimensional Analysis Toolbox for Matlab also brings some private function which cannot be accessed from outside the toolbox. The private functions have the status “undocumented” and are subject to change any time.

`formdm` - form a dimensional matrix

`fzerom` - find zero rows/columns in a matrix

`lindep` - find linear dependent rows/columns

`rmlindep` - eliminate linear dependent rows/columns

The private sub functions are stored in the private subdirectory of the Dimensional Analysis Toolbox.

Version History

1.00 2002-02-09

- initial version (aka 1.0)

1.01 2002-02-18

- some bugs eliminated
- added `data2si`
- added `Load/Save/Clear/Cancel` to `matedit`
- changed algorithm for linear dependent row deletion
- the use of `unit2si` in the examples changed

Known problems

- The `pretty` and `latex` commands make use of the symbolic math toolbox. The identification if this toolbox is installed and a license is available relies on a try-catch statement which in some configuration produces long error messages in the base workspace. I'm looking for a workaround to check if a license is available.
- A good number of error messages and warnings might be displayed in the base workspace when using the GUI `datool`. Otherwise these messages should be displayed as message boxes which get really annoying. So check you base workspace windows for warnings and error messages…
- The unit information in `unit2si` is not parsed for combinations and prefixes. If this should be included, an inconvenient style for unit representation would be necessary. This is design for simplicity in use (a parser would maybe have been less code to type…)
- Using the windows close function in the matrix editor produces an error. Sorry, Matlab closes the ActiveX control before returning to the script, so all data is lost and a dispatch error is generated.
- The Matlab search path is not saved automatically by `dinst.m`. Who knows how to do that?
- I would like to add some help pages to the Matlab help system. Any ideas how this can be accomplished without writing all the pages manually?

Feedback

Any comments and bug-reports are welcome. Please check the Dimensional Analysis Toolbox homepage at `http://www.sbrs.net/` for known bugs, solutions and product updates.

You are always welcome to send me an email: `datool@sbrs.net`.

Literature

1. Buckingham, E., On Physically Similar Systems: Illustration of the Use of Dimensional Equations, Phys. Review 4, 345-376, 1914.

2. Huntley, H., Dimensional Analysis, MacDonald, London, 1952.

3. Szirtes, T., Applied Dimensional Analysis and Modeling, McGraw-Hill, New York, 1997.

4. Baker, W.E., Westine, P.S., Dodge, F.T., Similarity Methods in Engineering Dynamics, Elsevier, New York, 1991.

5. Brückner, S. and Rudolph, S.: Several Papers on the application of dimensional analysis, `http//www.isd.uni-stuttgart.de/~brueckner/` or `http://www.sbrs.net/`

6. Barenblatt, G.I., Dimensional Analysis; Gordon and Breach, New York, 1987

7. Douglas, J.F., An Introduction to Dimensional Analysis for Engineers; Sir Isaac Pitman & Sons, London, 1969

8. Focken, C.M., Dimensional Methods and their Application; Edward Arnold and Co, London, 1953

Visit `http://www.sbrs.net/` for a more elaborate list of dimensional analysis book.